İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

ÇOK - BOYUTLU KAOTİK SİSTEMLER İLE ŞİFRELEME

YÜKSEK LİSANS TEZİ

Müh. Asiye YİĞİT

Tezin Enstitüye Verildiği Tarih: 20 Ocak 1997

Tezin Savunulduğu Tarih : 6 Şubat 1997

Tez Danışmanı : Doç.Dr. Cüneyt GÜZELİŞ

Diğer Jüri Üyeleri: Prof.Dr. Ahmet DERVİŞOĞLU

Doç.Dr. Acar SAVACI

OCAK 1997

ÖNSÖZ

Son 35 yılda doğrusal olmayan dinamik sistemler ile ilgili çalışmalar oldukça arttı. Pek çok araştırmacı, bu dönemde gelişen geometrik ve kalitatif tekniklerin gücünün ve güzelliğinin farkına vardı. Daha da önemlisi onlar, bu teknikleri, fizik ve kimyadan, ekoloji ve ekonomiye kadar pek çok alanda varolan doğrusal olmayan problemlere uygulayabildiler. Elde edilen sonuçlar, bilime oldukça katkı sağlayan nitelikteydi: Analitik bir görüş açısından tamamen takip edilemez gibi görünen sistemler, geometrik ve kalitatif teknikler ile kolayca anlaşılabildi. Böylece deterministik sistemlerin çözümlerinin görünen rasgele davranışı, pek çok durum için anlamlandırıldı.

Kaos bugünlerde, hemen hemen tüm bilim dallarında üzerinde çalışılan bir konudur. Kaotik bir dinamik sistem, ilk koşullara duyarlığın bir sonucu olarak rasgele davranış sergileyen deterministik bir sistemdir. Pratikte ilk koşullar, sonsuz bir prezisyon ile belirlenemediğinden, tam olarak ifade edilemezler. Bu nedenle kaotik bir sistemin davranışı, tahmin edilemezdir. Böylece kaotik sistemler, rasgele doğasından dolayı haberleşme sistemleri ile ilgili çalışmalarda kullanılmaya başlandı. Bunun yanısıra ilk koşullara olan duyarlığı sebebiyle çok sayıda, tamamen farklı, rasgele benzeri ve yeniden üretilebilen diziler oluşturan kaotik sistemlerin, şifreleme sistemlerinde anahtar dizisi olarak kullanılabileceği de gösterildi. Biz bu çalışmada, çok boyutlu kaotik sistemlerin, bu konuda yapılan çalışmaları da göz önünde bulundurarak, tek zamanlı sistemlerde kullanılabilen rasgele sayı üreteci olabileceğini gösterdik.

Bu tezi hazırlamama, gerek yardımcı ve öğretici eleştirileri ile gerekse kütüphanesinin kapılarını bana açarak büyük katkılar sağlayan Sayın Hocam Doç. Dr. Cüneyt Güzeliş'e çok teşekkür ederim.

Ayrıca Devreler ve sistemler Anabilim Dalı araştırma görevlilerinden Sayın Müştak Yalçın'a, sorduğum soruları içtenlikle dinlediği ve cevapladığı için teşekkür etmek istiyorum. Bunun yanı sıra Sayın Prof. Dr. Nadir Yücel'e sağladığı yardımcı kaynaklardan dolayı teşekkür ederim.

Son olarak anneme, ablama, erkek kardeşime ve babama sağladıkları manevi destekten dolayı minnettarım.

Asiye YİĞİT

Ocak 1997

İÇİNDEKİLER

ÖNSÖZ ii

ŞEKİL LİSTESİ vii

TABLO LİSTESİ xii

TANIM LİSTESİ xiv

TEOREM LİSTESİ xvi

ÖZET xvii

SUMMARY xviii

BÖLÜM 1 GİRİŞ 1

BÖLÜM 2 DOĞRUSAL OLMAYAN DİNAMİK SİSTEMLER 7

2.1 Genel Tanımlar 8
2.2 Vektör Alanlarının Bazı Genel Özellikleri 11
2.3 Denge Noktaları 13
2.3.1 Liapunov'un Dolaysız Yöntemi 14
2.3.2 Liapunov'un Dolaylı Yöntemi (Doğrusallaştırma) 15
2.3.3 Hiperbolik Denge Noktaları 17
2.3.3.1 Dönüşümler 18
2.4 Periyodik Çözümler 19
2.5 Poincaré - Bendixson Teoremi 20
2.5.1 Limit Kümeler 20
2.5.2 Poincaré - Bendixson Teoremi 21
2.5.3 Limit Çevrimler 21
2.6 Poincaré Dönüşümleri 21
2.6.1 Periyodik Bir Orbit Civarındaki Poincaré Dönüşümü 21
2.6.2 Periyodik Çözümlerin Kararlılığı: Karkteristik Çarpanlar 22
2.7 Sanki Periyodik Çözümler 23

BÖLÜM 3 KAOS VE GARİP ÇEKİCİLER 24

3.1 Ön Tanımlar 24
3.2 Kaos 25
3.2.1 Kaotik Sistemlere Örnekler 28
3.2.1.1 Çadır ("Tent") Dönüşümü 29
3.2.1.2 Testere Dişi ("Sawtooth") Dönüşümü 29

3.2.1.3 Sinüs Dönüşümü 30
3.2.1.4 Öteleme Dönüşümü 30
3.3 Garip Çekiciler 31

BÖLÜM 4 LOJİSTİK DÖNÜŞÜM İÇİN DUYARLIK TOPOLOJİK GEÇİŞLİLİK VE PERİYODİK ORBİTLER 34

4.1 İlk Koşullara Duyarlık 35
4.1.1 Küçük Hataların Duyarlık Özelliğinden Dolayı Güçlenmesi 38
4.1.2 Doğrusal Dönüşüme Bir Örnek 39
4.1.3 Lorenz'in Yaklaşımı 40
4.1.4 Hata Gelişiminin Analizi 41
4.1.5 Liapunov Üstelleri 45
4.2 Topolojik Geçişlilik 47
4.3 Yoğun Periyodik Orbitler 49
4.3.1 Periyodik Orbitleri Üreten Mekanizma 50
4.4 Kaosu Yaratan Temel İşlem: Yoğurma 52
4.4.1 Yoğurma İşleminin Çekme ve Katlama Şeklinde Yorumlanması 53
4.4.2 Yoğurma İşleminin Çekme - Katlama ve Yapıştırma Şeklinde Yorumlanması 54
4.4.3 Lojistik Dönüşümde Yoğurma İşlemi 58
4.4.4 Testere Dişi Dönüşümü 59
4.5 Kaosun Analizi 61
4.5.1 İki Sembol Üzerine Öteleme Dönüşümü 61
4.5.1.1 İlk Koşullara Duyarlık 61
4.5.1.2 Periyodik Orbitler 62
4.5.1.3 Topolojik Geçişlilik 63
4.5.2 Çadır Dönüşümü 63
4.5.2.1 Yoğun Periyodik Orbitler 63
4.5.2.2 İlk Koşullara Duyarlık 66
4.5.2.3 Topolojik Geçişlilik 66
4.6 Lojistik Dönüşüm için Kaos 67
4.6.1 Çadır Dönüşümü ile Lojistik Dönüşümün Denkliği 69
4.6.1.1 Yoğun Periyodik Orbitler ve Topolojik Geçişlilik 73
4.6.1.2 Duyarlık 74

BÖLÜM 5 GARİP ÇEKİCİLER 76

5.1 Chua Çekicisi 77
5.1.1 Chua Devresinin Denge Noktaları 79
5.1.2 Denge Noktalarının Kararlılığı 80
5.2 Lorenz Çekicisi 90
5.2.1 Sistemin Kalıcı Durum Davranışları 94

BÖLÜM 6 KRİPTOGRAFİ VE SANKİ - RASGELE DİZİ ÜRETEÇLERİ 100

6.1 Kriptografi 101
6.1.1 Tek Zamanlı Sistemler 103
6.2 Güvenli Sanki - Rasgele Bit Üreteçleri 105

6.2.1 İnşa Blokları 105
6.2.1.1 Doğrusal Kongruens Üreteçler 106
6.2.1.2 Geribeslemeli Ötemeli Yazıcılar 106
6.2.1.3 Doğrusal Olmayan Geribeslemeli Ötemeli Yazıcılar 107
6.2.1.4 Doğrusal Geribeslemeli Ötemeli Yazıcılar 108
6.3 İstatistiksel Rasgelelik Testleri 111
6.3.1 İstatistik Hipotezlerin Kontrolu 111
6.3.2 İstatistiksel Rasgelelik Testleri 117
6.3.2.1 Frekans Testi 117
6.3.2.2 Seri Test 118
6.3.2.3 Poker Testi 118
6.3.2.4 Hareket Testi 121
6.3.2.5 Özilişki "otokorelasyon" Testi 123

BÖLÜM 7 KAOTİK SİSTEMLERİN ŞİFRELEMEDE KULLANIMINA İLİŞKİN YAPILAN ÇALIŞMALAR 126

7.1 Kaotik Bir Şifreleme Algoritmasının Türetilmesi 127
7.1.1 Lojistik Dönüşümün Kaotik Davranışı 128
7.1.2 Genelleştirilmiş Kaotik Dönüşüm 129
7.1.3 Kaotik Dönüşümün Kriptografide Kullanımı 133
7.1.4 Kaotik Dönüşümün Kriptografide Kullanım Güvenliğinin Tahmini Üst 135
7.1.5 Sonuç 136
7.2 Kaotik Şifreleme Sistemlerindeki Problemler 136
7.2.1 Matthews'in Genelleştirdiği Kaotik Dönüşüm 138
7.2.2 Ön Çalışmalar 139
7.2.3 Çevrim Uzunluklarıyla İlgili Özellikler 140
7.2.4 Sonuçlar 143
7.3 Doğrusal Olmayan Anahtar Üreteçleri 144
7.3.1 Monotonik Doğrusal Olmayan Üreteçler 145
7.3.2 Simülasyonlar 146
7.3.3 Sonuç 147
7.4 Kotik Bir Şifreleme Algoritmasının Süper Bilgisayarlar Yardımıyla İncelenmesi 147
7.4.1 Algoritma ve Onun Gerçekleştirilmesi 148
7.4.2 Yuvarlatma Hatalarından Kaynaklanan Çevrimler 149
7.4.3 Çevrim Uzunluklarının ve Diğer Özelliklerin Cray Y - MP ile İncelenmesi 151
7.4.4 Sonuç 153
7.5 Kriptografide Kaos: Garip Çekicilerden Kaçış 154
7.5.1 Yüksek Adım Aralığı ile Ayrıklaştırılmış Lorenz Sistemi 157
7.5.2 Rasgeleliği Ölçen Testler 161
7.5.3 Sonuç 161

BÖLÜM 8 ÇOK BOYUTLU KAOTİK SİSTEMLER İLE ŞİFRELEME 163

8.1 Chua ve Lorenz Sitemlerinin Şifrelemede Kullanılması 163

SONUÇLAR VE ÖNERİLER 168

KAYNAKLAR 171

EKLER 176

ÖZGEÇMİŞ 232

ŞEKİL LİSTESİ

		Sayfa No
Şekil 2.1	U sınırındaki vektör alanı	14
Şekil 2.2	V = Sabit eğrisinin sınırındaki bazı noktaların ∇V vektörleri	15
Şekil 3.1	Çadır dönüşümü	29
Şekil 3.2	Testere dişi dönüşümü	30
Şekil 4.1	$x_0 = .2027$ ilk koşulu ve a = 4 parametre değeri için lojistik dönüşümünün zaman serisi	35
Şekil 4.2	$x_0 = .2027$ ilk koşulu ve a = 3.742718 parametre değeri için lojistik dönüşümün zaman serisi	36
Şekil 4.3a	a = 2.75 değeri için $x_0 = 0.5$ ilk koşulunun ≈ 0.64 son durumuna yakınsaması	37
Şekil 4.3b	a = 2.75 değeri için $x_0 = 0.3$ ilk koşulunun ≈ 0.64 son durumuna yakınsaması	37
Şekil 4.4a	a = 2.75 parametre değeri ve $x_0 = 0.22$ ilk koşulu için dönüşümün kararlı tek bir noktaya yakınsaması	38
Şekil 4.4b	a = 2.75 parametre değeri, $x_0 - 0.015$ ve $x_0 + 0.015$ değerleri için dönüşümün tek bir noktaya yakınsaması	38
Şekil 4.5	Çok küçük bir aralığın tekrarlamalar sonucunda oldukça geniş bir bölgeye yayılması	39
Şekil 4.6	Doğrusal sistemlerdeki duyarlığın $x \mapsto 1.5x$ doğrusal dönüşümü için incelenmesi	40
Şekil 4.7a	$x_0 = 0.6$ ilk koşulunun ürettiği orbit	42
Şekil 4.7b	$y_0 = 0.600001$ ilk koşulunun ürettiği orbit	42
Şekil 4.7c	$x_0 = 0.6$ ve $y_0 = 0.600001$ ilk koşul değerlerinin oluşturduğu orbitler arasındaki farkın mutlak değeri	42

Sayfa No

Şekil 4.8a $x_0 = 0.6$ ilk koşulunun $x \mapsto 1.5x$ doğrusal dönüşümü altında ürettiği orbit 43

Şekil 4.8b $y_0 = 0.600001$ ilk koşulunun $x \mapsto 1.5x$ doğrusal dönüşümü altında ürettiği orbit 43

Şekil 4.8c $x_0 = 0.6$ ve $y_0 = 0.600001$ ilk koşul değerlerinin $x \mapsto 1.5x$ doğrusal dönüşümü altında oluşturduğu orbitlerin arasındaki farkın mutlak değeri 43

Şekil 4.9 $x \mapsto 1.5x$ doğrusal dönüşümü için $x_0 = 0.202$ ve $y_0 = 0.202001$ ilk koşulları ele alındığında $(y_n - x_n) / x_n$ değerinin grafiği 44

Şekil 4.10 Herhangibir J altaralığına ulaşan bir ilk koşulun I altaralığında var olması 49

Şekil 4.11 $\sin^2(\pi/7) = 0.1882550$ ilk koşulunun ürettiği gerçek periyodik orbitten bilgisayardan kaynaklanan yuvarlatma hatalarından dolayı uzaklaşılması 50

Şekil 4.12a 25 farklı noktanın lojistik dönüşümü altında aldığı değerler 51

Şekil 4.12b Gerçek parabol ve dönüşüm uygulanmış noktaların meydena getirmiş olduğu merdiven fonksiyonu 51

Şekil 4.13 Oldukça düşük doğrulukla aritmetiğin dört farklı uzun terim davranışına neden olması 52

Şekil 4.14 Yoğurma işleminin uzatma - katlama işlemi ile yorumlanması 53

Şekil 4.15 12 bölüme ayrılmış hamura iki kez çekme - katlama işleminin uygulanması 54

Şekil 4.16 Çekme - katlama işleminin çizgi ile ifade edilen hamura dört kez uygulanması 55

Şekil 4.17 Çekme - kesme ve yapıştırma işlemleri ile yorumlanan yoğurma işlemi 55

Şekil 4.18 12 parçaya ayrılmış hamura çekme - katlama işleminin uygulanmasının ardından çekme - kesme ve yapıştırma işleminin uygulanması ve bu sonucun aynı hamura iki kez çekme - katlama işlemi uygulanmış sonuçla karşılaştırılması 56

Şekil 4.19a $x = 9/10$ ilk koşuluna üç kez T işleminin uygulanması 57

Şekil 4.19b $x = 9/10$ ilk koşuluna S, S, T işlemlerinin uygulanması 57

Şekil 4.20 Çekme - katlama işlemine karşı gelen çadır dönüşümü 57

Şekil 4.21 Çekme - kesme ve yapıştırma işlemlerine karşı gelen testere dişi dönüşümü 58

Şekil 4.22 $x \mapsto 4x(1-x)$ lojistik dönüşümünün grafiği 58

Şekil 4.23 $h(x) = \sin^2(\pi x/2)$ dönüşümünün grafiği 69

Şekil 5.1 Doğrusal bir self (L), doğrusal kapasite (C_1, C_2), doğrusal bir direnç (R) ve gerilim kontrollu doğrusal olmayan bir dirençten (N_R) oluşan Chua devresi 77

Şekil 5.2 Doğrusal olmayan N_R direncinin $\pm$ E karakteristiği 78

Şekil 5.3 $\alpha = 4$, $\beta = 33$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin (1.5, 0, -1.5) denge noktasına yakınsamasının x - y düzleminde gözlemlenmesi 83

Şekil 5.4 $\alpha = 4$, $\beta = 33$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin (1.5, 0, -1.5) denge noktasına yakınsamasının x - z düzleminde gözlemlenmesi 83

Şekil 5.5 $\alpha = 9$, $\beta = 16.5$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x - y düzlemindeki davranışı 84

Şekil 5.6 $\alpha = 9$, $\beta = 16.5$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x - z düzlemindeki davranışı 84

Şekil 5.7 $\alpha = 9$, $\beta = 16$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x - y düzlemindeki davranışı 85

Şekil 5.8 $\alpha = 9$, $\beta = 16$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x - z düzlemindeki davranışı 85

Şekil 5.9 $\alpha = 9$, $\beta = 15.5$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x - y düzlemindeki davranışı 86

Şekil 5.10 $\alpha = 9$, $\beta = 15.5$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x - z düzlemindeki davranışı 86

Şekil 5.11 $\alpha = 9$, $\beta = 100/7$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x - y düzlemindeki davranışı 87

Şekil 5.12 $\alpha = 9$, $\beta = 100/7$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri

için sistemin x - z düzlemindeki davranışı 87

Şekil 5.13 $\alpha = 9$, $\beta = 100/7$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin y - z düzlemindeki davranışı 88

Şekil 5.14 $\alpha = 9$, $\beta = 15.5$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için x(t)'nin grafiği 88

Şekil 5.15 $\alpha = 9$, $\beta = 100/7$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için y(t)'nin grafiği 89

Şekil 5.16 $\alpha = 9$, $\beta = 100/7$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x - z düzlemindeki davranışı 89

Şekil 5.17 $\sigma = 10$, $b = 8/3$ ve $r = 0.5$ parametre değerleri için tüm ilk koşulların (0, 0, 0) denge noktasına yakınsaması 94

Şekil 5.18 $r = 0.5$ parametre değeri için Lorenz sisteminin (0, 0, 0) denge noktasına yakınsaması 95

Şekil 5.19 $r = 0.5$ parametre değeri için Lorenz sisteminin simetrik kararlı denge noktalarından birine yakınsaması 96

Şekil 5.20 $r = 0.5$ parametre değeri için Lorenz sisteminin x - z düzlemindeki kalıcı durum davranışı 96

Şekil 5.21 $r = 28$ parametre değeri için x - y düzlemindeki Lorenz çekicisi 97

Şekil 5.22 $r = 28$ parametre değeri için x - z düzlemindeki Lorenz çekicisi 97

Şekil 5.23 $r = 28$ parametre değeri için x değişkeninin zamana bağlı grafiği 98

Şekil 5.24 $r = 28$ parametre değeri için y değişkeninin zamana bağlı grafiği 98

Şekil 5.25 $r = 28$ parametre değeri için z değişkeninin zamana bağlı grafiği 99

Şekil 6.1 Bir şifre 101

Şekil 6.2 Bir blok şifre 102

Şekil 6.3 Bir dizi şifre 103

Şekil 6.4 Tek zamanlı bir sistem 104

Şekil 6.5 n - durumlu geribeslemeli ötemeli yazıcı 107

Şekil 6.6. Dört durumlu De Brujin dizi üreteci 108

Şekil 6.7 $\beta = \beta_0$ hipotezinin $\beta \neq \beta_0$ karşıt hipotezine göre kontrolu 112

Şekil 6.8 $\beta = \beta_0$ hipotezinin $\beta > \beta_0$ karşıt hipotezine göre kontrolu 113

Şekil 6.9 χ^2 - dağılımı 114

Şekil 6.10 N(0,1) dağılımı 115

Şekil 7.1 Kullanılabilir tekrarlama sayısı 152

Şekil 7.2 Çevrimle sonuçlanan sonlanmaların oranı 153

Şekil 7.3 Çevrim uzunlukları 153

Şekil 7.4 r = 28 için x - y düzlemindeki Lorenz çekicisi 156

Şekil 7.5 r = 28 için z - x düzlemindeki Lorenz çekicisi 156

Şekil 7.6 Küçük Δt aralıkları için çizilen grafik
($\Delta t = 0.01$ ——, $\Delta t = 0.024$ .....) 158

Şekil 7.7 Büyük Δt aralıkları için çizilen grafik 158
($\Delta t = 0.01$ ——, $\Delta t = 0.0245$ .....)

Şekil 7.8 Büyük Δt değerleri için Lorenz çekicisi
($\Delta t = 0.01$ ——, $\Delta t = 0.0245$ .....) 159

Şekil 7.9 Büyük Δt değerleri için tekrarlama sayısının
arttırılması durumunda Lorenz çekicisi
($\Delta t = 0.01$ ——, $\Delta t = 0.0245$ .....) 159

TABLO LİSTESİ

		Sayfa No
Tablo 2.1	İkinci dereceden doğrusal bir sistem için denge noktalarının sınıflandırılması	14
Tablo 4.1	$x(n+1) = 4x(n)(1-x(n))$ lojistik dönüşümü için hata yayılımı	44
Tablo 4.2	Farklı tekrarlama değerleri için ortalama hata güçlendirme faktörlerinden hesaplanan Liapunov üstelleri	47
Tablo 4.3	$I_2 = [0.1, 0.2]$ aralığının dört kez dönüşüm altında tekrarlanması sonucunda başlangıçta belirlenen tüm alt aralıklara ulaşılması	48
Tablo 4.4	25 farklı noktaya dönüşüm altında karşı gelen noktalar	51
Tablo 4.5	$x_0 = 8/25$ ilk koşulunun çadır dönüşümü altındaki ilk 9 tekrarlama sonucu, x'_0 değerleri ve bunların, $x'(k) = h(x(k))$ dönüşüm değerlerinin hesaplanarak kontrolunun yapılması	70
Tablo 6.1	χ^2 - dağılımı	116
Tablo 7.1	Kullanılabilir tekrarlama sayısı	142
Tablo 7.2	Çevrim ile sonuçlanan sonlanmaların oranı	144
Tablo 7.3	Çevrim uzunlukları	151
Tablo C.1	Frekans testi sonuçları	219
Tablo C.2	Seri testi sonuçları	220
Tablo C.3	m = 8 için poker testi sonuçları	220
Tablo C.4	m = 4 için poker testi sonuçları	220
Tablo C.5	m = 3 için poker testi sonuçları	221
Tablo C.6	m = 2 için poker testi sonuçları	221

		Sayfa No
Tablo C.7	Hareket testi sonuçları	221
Tablo C.8	d = 1 için özilişki testi sonuçları	222
Tablo C.9	d = 2 için özilişki testi sonuçları	222
Tablo C.10	d = 10 için özilişki testi sonuçları	223
Tablo C.11	d = 1000 için özilişki testi sonuçları	223
Tablo C.12	d = 10000 için özilişki testi sonuçları	223
Tablo D.1	Frekans testi sonuçları	224
Tablo D.2	Seri testi sonuçları	225
Tablo D.3	m = 8 için poker testi sonuçları	225
Tablo D.4	m = 4 için poker testi sonuçları	225
Tablo D.5	m = 3 için poker testi sonuçları	226
Tablo D.6	m = 2 için poker testi sonuçları	226
Tablo D.7	Hareket testi sonuçları	227
Tablo D.8	d = 1 için özilişki testi sonuçları	227
Tablo D.9	d = 2 için özilişki testi sonuçları	227
Tablo D.10	d = 10 için özilişki testi sonuçları	228
Tablo D.11	d = 1000 için özilişki testi sonuçları	228
Tablo D.12	d = 10000 için özilişki testi sonuçları	229

TANIM LİSTESİ

		Sayfa No
Tanım 2.1	Fonksiyon	8
Tanım 2.2	C^r Fonksiyonlar	8
Tanım 2.3	Homeomorfizm	8
Tanım 2.4	C^r Diffeomorfizm	8
Tanım 2.5	Çözüm	9
Tanım 2.6	Vektör Alanı	9
Tanım 2.7	İntegral Eğrisi	9
Tanım 2.8	Orbit	9
Tanım 2.9	Akış	10
Tanım 2.10	Otonom Olmayan Vektör Alanları	10
Tanım 2.11	Otonom Vektör Alanları	10
Tanım 2.12	Liapunov Anlamında Kararlılık	13
Tanım 2.13	Liapunov Anlamında Asimptotik Kararlılık	13
Tanım 2.14	Denge Noktası	13
Tanım 2.15	Hiperbolik Denge Noktası	17
Tanım 2. 16	Vektör Alanı için Periyodik Çözüm	19
Tanım 2.17	Dönüşüm için Periyodik Çözüm	20
Tanım 2.18	ω - Limit Noktası	20
Tanım 2.19	ω - Limit Kümesi	20

		Sayfa No
Tanım 2.20	α - Limit Noktası	20
Tanım 2.21	α - Limit Kümesi	20
Tanım 2.22	Limit Çevrim	21
Tanım 2.23	Sanki Periyodik Çözüm	23
Tanım 3.1	Değişmez Küme	24
Tanım 3.2	Gezgin Olmayan Nokta	24
Tanım 3.3	Çekici Küme	25
Tanım 3.4	Çekim Bölgesi	25
Tanım 3.5	Tuzak Bölgesi	25
Tanım 3.6	Topolojik Geçişlilik	25
Tanım 3.7	Çekici	26
Tanım 3.8	Vektör Alanları için İlk Koşullara Duyarlık	26
Tanım 3.9	Dönüşümler için İlk Koşullara Duyarlık	26
Tanım 3.10	Vektör Alanları için Kaotiklik	26
Tanım 3.11	Dönüşümler için Kaotiklik	26
Tanım 3.12	Dizi Uzayı	30
Tanım 3.13	Öteleme Dönüşümü	30
Tanım 3.14	Garip Çekici	31

TEOREM LİSTESİ

		Sayfa No
Teorem 2.1	Picard'ın Zamanda Yerel Varlık ve Teklik Teoremi	11
Teorem 2.2	Genel Varlık ve Teklik Teoremi	11
Teorem 2.3	Peano Varlık Teklik Teoremi	12
Teorem 2.4	Başlangıç Şartlarına Göre Varlık Teklik ve Türetilebilirlik	12
Teorem 2.5	Liapunov'un Dolaysız Yöntemi	14
Teorem 2.6	Denge Çözümünün Asimptotik Kararlılığı	16
Teorem 2.7	Bendixson Kriteri	19
Teorem 2.8	Poincaré - Bendixson Teoremi	21
Teorem 2.9	Periyodik Çözümlerin Kararlılığı	23
Teorem 5.1	Chua Devresinde Matematiksel Kaos	90

ÖZET

Günümüzde astronomi, biyoloji, biyofizik, kimya, mühendislik, jeoloji, matematik, tıp, meteoroloji, plazma, fizik ve hatta sosyal bilimlerde üzerinde çalışılan bir konu olan kaosun analizi oldukça zordur. Buna rağman, lojistik dönüşümün belli bir modelinden, kaotik davranışın açıklanması için yararlanılması oldukça iyi bir yoldur.

Kaosun üç karakteristiği vardır: İlk koşullara aşırı duyarlılık, topolojik geçişlilik ve yoğun periyodik orbitler. Hamurun yoğurulması işlemi, kaosun tüm bu matematiksel özelliklerine ulaşılmasını sezgisel olarak sağlar.

Bütün güvenli haberleşme sistemleri, asıl mesaj işaretinin yalnızca mesajın iletilmek istendiği alıcılar tarafından bilinen ve diğer alıcılar tarafından çözülmesi zor olan bir şifreyle karıştırılmasına dayanır. Doğrusal ve doğrusal olmayan ötemeli yazıcılar rasgele şifreleme işaretleri üretmede kullanılan donanımlara örnektirler. Kaotik sistemler çok az parametre ile tanımlı olmalarına rağmen ancak başlangıç koşullarının kesin olarak belirlenmesi durumunda nasıl davranacağı tamamıyla bilinebilen aksi halde ise rasgele davranış sergileyen sistemlerdir. Başlangıç koşullarına aşırı duyarlık özelliği bir çok araştırmacıyı, kaotik işaretleri şifreleme işaretleri olarak kullanmaya yöneltmiştir. Fakat gerçel sayı dizileri olarak karşımıza çıkan kaotik işaretler, sayısal gerçeklemede kuvantalama hatalarından dolayı periyodu kısa işaretlere dönüşürler. Buna rağmen, kaotik dönüşümlerin yüksek prezisyonlu bilgisayarlarda gerçekleştirilmesi sonucu meydana gelen çevrim uzunluklarının pratik uygulamalar için yeterli olduğu gösterildi. Tek boyutlu kaotik dönüşümlerin sifrelemede kullanılmasını öneren çalışmalar yanında, çok boyutlu kaotik sistemlerin kullanılmasını öneren çalışmalar da vardır. Fakat bu çalışmalarda sistemlerin kaotik yapısı, Δt zaman adımının büyük tutularak nümerik olarak çözülmesiyle yok edilmiştir.

Bu çalışmada Lorenz ve Chua sistemlerinin, kaotik durumda da rasgele sayılar üretebileceği iddia edilmektedir. Bu amaçla sistemin küçük zaman adımları için elde edilen nümerik çözümlerinin en düşük anlamlı rakamlarından yararlanılarak oluşturulan dizilere, rasgele olduklarını doğrulamak için istatistiksel rasgelelik testleri uygulandı. Sonuçlar yukarıda verilen iddiayı destekler yöndedir. Böylelikle Lorenz ve Chua sistemlerinin söz edilen metot ile, sayısal bir güvenli haberleşme sisteminde şifre olarak kullanılabileceği gösterilmiş oldu.

SUMMARY

In the last thirty five years, there has been a great deal of interest in the study of nonlinear dynamical systems. And, the result has been a better qualitative understanding of previously intractable systems. The apparent random behavior of solutions of deterministic systems is now very easily comprehended in many cases.

Generally speaking, the analysis of chaos is extremely difficult. However, in the specific model case of the logistic map there is a beautiful and illuminating way to really understand chaotic behavior through and through.

There are many possible definitions of chaos, ranging from measure theoretic notions of randomness in ergodic theory to the topological approach we will adopt here.

Definition 1. $f:J \to J$ is said to be topologically transitive if for any pair of open sets $U, V \subset J$ there exists $k > 0$ such that $f^k(U) \cap V \neq \varnothing$.

Definition 2. $f:J \to J$ has sensitive dependence on initial conditions if there exists $\delta > 0$ such that, for any $x \in J$ and any neighborhood N of x, there exists $y \in N$ and $n \geq 0$ such that $|f^n(x) - f^n(y)| > \delta$.

Definition 3. Let V be a set. $f:V \to V$ is said to be chaotic on V if

1. f has sensitive dependence on initial conditions.
2. f is topologically transitive.
3. Periodic points are dense in V.

To summarize, a chaotic map possesses three ingredients: unpredictability, indecomposability, and an element of regularity. A chaotic system is unpredictable because of the sensitive dependence on initial conditions. It cannot be broken down or decomposed into two subsystems (two invariant open subsets) which do not interact under f because of topological transitivity. And, in the midst of this random behavior, we nevertheless have an element of regularity, namely the periodic points which are dense.

Having discussed the phenomena of chaos and the routes leading to it in 'simple' one-dimensional settings, we continue with the exposition of chaos in dynamical systems of two or more dimensions. This the relevant case for models in the natural sciences since very rarely can processes be described by only one single state variable. They are strange attractors. Roughly speaking, an attractor is an invariant set to which all nearby orbits converge.

The first strange attractor ever recognized as such in the natural sciences is the Lorenz attractor, discovered in 1962. The Lorenz system is a model of thermal convection which, however includes not only a description of the motion of some viscous fluid or atmosphere but also the information about distribution of heat, the driving force of thermal convection.

Chaos is seen in the electrical circuits. Chua's circuit is the simplest electronic electronic circuit that satisfies the mathematical properties of chaos. In addition, this remarkable circuit is the only physical system for which the presence of

chaos has been proved mathematically. The circuit is readily constructed at low cost using standard electronic components and exhibits a rich variety of bifurcations and chaos.

The focus of this research is that Chua attractor and the Lorenz attractor can be used in cryptography as a secure pseudorandom generator.

Cryptography is set of techniques used to ensure the secrecy of messages when hostile personnel have the technical capability to intercept and correctly interpret an unprotected message, which is called the plaintext. After transformation to secret form, a message is called the ciphertext or a cryptogram. The process of transformation the plaintext into ciphertext is called encryption. Cryptanalysis is the unauthorized extraction of information from the ciphertext.

A practical cryptographic communication system must produce ciphertext that only resists cryptanalysis, but also can be efficiently deciphered by authorized personnel.

In a cryptographic system, a set of parameters that determines a specific cryptographic transformation or its inverse is called a key. In most applications, a single key determines both a cryptographic transformation and its inverse. Cryptographic systems are designed so that their security depends upon the inability of the cryptanalyst to determine the key even if she knows the structure of the cryptographic system.

An enciphering algorithm or transformation is called a cipher. Messages can be enciphered by a block cipher, a stream cipher, or a combination of these ciphers. A block cipher divides the plaintext into separate blocks of m bits and associates with each block $(x_i, x_{i+1}, \ldots, x_{i+m-1})$ of plaintext a block $(y_i, y_{i+1}, \ldots, y_{i+n-1})$ of n bits of ciphertext such that $y_k = f_k(x_i, x_{i+1}, \ldots, x_{i+m-1})$, $i \le k \le i+n-1$ where the f_k are functions. Messages are deciphered with inverse functions. For unambiguous deciphering, $n \ge m$ is necessary; for ease of automation, $n = m$ in nearly all practical block ciphers.

In stream ciphers, on the other hand, the plaintext is encrypted on a bit - by - bit basis. In encrypting the dataflow to be transmitted, the key is fed into an algorithm called the pseudorandom bit generator to create a long sequence of binary signals. This "key - stream" is then mixed with the plaintext sequence, usually by Exclusive - OR (XOR bit - wise modulo - 2 addition) gates, to produce the ciphertext.

Stream ciphers play an especially important role in cryptographic practices - both diplomatic and military - that protect communications in the very high frequency domain. The central problem in stream - cipher cryptography, however, is the difficulty of generating a long unpredictable sequence of binary signals from a short and random key. Unpredictable sequences are desirable in cryptography because it is impossible, given a reasonable segment of its signals and computer resources, to find out more about them. Pseudorandom bit generators have been widely used to construct these sequences.

There are two types of stream ciphers: those for which the keystream is independent of the plaintext and those for which the keystream is not. Independently keyed ciphers are often called synchronous ciphers because they require synchronization between the keystream and the ciphertext for successful deciphering. Stream ciphers for which the keystream is a function of the plaintext are often called auto - key ciphers.

The only unconditionally secure cipher is a synchronous cipher that generates a completely random keystream as long as the message. Because the key, which is the same as the keystream in this case, cannot be reused, this cipher is called the one - time tape or one - time pad. If the key is produced by a binary symmetric source such that the probability for any enciphering signal to be 1 is equal to 1/2 independently of the other signals, the key is random.

A known - plaintext attack is an attempted cryptanalysis based upon some knowledge of corresponding ciphertext and plaintext. An unconditionally secure cipher is one that is invulnerable to cryptanalysis no matter how much ciphertext is available. A computationally secure cipher is one that is vulnerable to cryptanalysis only if a prohibitively large amount of computation is performed. Most practical ciphers are at best computationally secure.

The necessity of long, frequently changed keys renders the one - time tape impractical for most applications. Therefore since the twenties, many cipher systems have operated with an approximate version of the one - time pad that uses long sequences of pseudorandom bits generated from short secret keys.

Generally speaking, we can formulate three requirements for cryptographically secure keystream generators.

(1) The period of the keystream must be large enough to accommodate the length of the transmitted message.

(2) The output bits must be easy to generate.

(3) The output bits must be hard to predict. Given the generator and the first n output bits, a(0), a(1),..., a(n-1), it should be computationally in feasible to predict the $(n+1)^{th}$ bit a(n) in the sequence with better than a 50 - 50 chance. That is, given a portion of the output sequence, the cryptanalyst should not generate other bits forward or backward.

Linear congruence generators (LCGs) widely discussed method for generating pseudorandom numbers is based on recurrences of the form

$$X(i+1) = aX(i) + b \mod m. \tag{1}$$

Here, (a,b,m) are the parameters describing the generator and can be utilized as secret keys. X_0, is the seed. If the parameters are chosen judiciously, the numbers X(i) will not repeat until all integers of the interval $[0, m-1]$ have occurred. Boyar shows that sequences generated by LCGs are not cryptographically secure, [45].

Strong cryptographic sequences can be designed on the basis of shift registers even when the feedback is linear. A feedback shift register consists of n flip - flops and a feedback function that expresses each new element a(t), when $t \geq n$, of the sequence in terms of the previously generated elements a(t - n), a(t - n + 1), ...,a(t -1). The feedback function must be nonsingular, that is, of the form

$$a(t) = g(a(t-1), a(t-2), \ldots, a(t-n+1)) \oplus a(t-n) \tag{2}$$

where $\oplus$ denotes the XOR operation.

Depending on the whether the function g is linear (implementable with XORs alone) or not, the generator is called a linear feedback shift register (LFSR) or a nonlinear feedback shift register (NLFSR). Each individual storage element of FSR

is called stage, and the binary signals a(0), a(1), a(2), ..., a(n-1) are loaded into them as initial condition.

The period of the sequence produced by an FSR depends both on the number of stages and on the details of the feedback connection. The maximal period of a sequence that can be generated by an n - stage FSR with nonsingular feedback is 2^n, the number of possible states an n - stage shift register may have.

The state diagram of a nonsingular FSR may have many small cycles, and the output sequence becomes insecure when the generator falls into one of them. A countermeasure is to design n - stage shift registers that generate sequences of the largest possible period 2^n, the so - called De Brujin sequences of degree n. The following is a De Brujin sequences of period 2^4, generated by the four - stage NLFSR,

1011001010000111....

LFSRs are also among the most important devices in building pseudorandom bit generators. They have feedback functions of the form

$$a(t) = c_1 a(t-1) \oplus c_2 a(t-2) \oplus ... \oplus c_{n-1} a(t-n+1) \oplus a(t-n), \quad (3)$$

with $a(t) = c_i \in \{0,1\}$. The feedback connection of an LFSR can be represented formally by so - called feedback polynomial

$$f(x) = 1 + c_1 x + c_2 x^{n-2} + ... + c_{n-1} x^{n-1} + x^n \quad (4)$$

in the indeterminate x, which has no other meaning than as a mathematical symbol. The feedback polynomial decides the period and the statistical behavior of the output sequence. To avoid trivial output, the zero state should be excluded from initial setting.

In general, to guarantee the largest possible period 2^{n-1} for the output sequence, the feedback polynomial f(x) of the LIFERS should be primitive. This means f(x) will be chosen so that the least positive integer T that makes X^T - 1 divisible by f(x) will be $T = 2^n - 1$ Maximality of the period simultaneously guarantees good statistics.

There are a lot of tests that have been applied to sequences in order to investigate their randomness. Here, we will present five statistical tests that are designed to determine if a sequence appears random. We will calculate a value, called a statistic, for a sequence and then compare that value with the corresponding distribution of values for random sequences. If the pseudorandom sequence exhibits a property that very few truly random sequences do, then we conclude that the pseudorandom sequence does not model a random sequence very well.

Suppose we wish to test a particular feature of a sequence. First we define a statistic to quantify that feature, and then we determine the distribution of that statistic for random sequences.

We conduct a statistical test by making a hypothesis, called the null hypothesis, and testing it against an alternative hypothesis. In our case the null hypothesis, H_0, will be the hypothesis that our sequence was produced by a random source. The alternative hypothesis, H_A, is the hypothesis that our sequence was

produced by one of some given set of nonrandom sources. Next we choose a value α, called the significance level, which is equal to the probability that we conclude our sequences was not produced by a random source, when in fact it was. That is,

$$\alpha = P(\text{reject } H_0 \mid H_0 \text{ is true}), \tag{5}$$

so $0 \le \alpha \le 1$. Common choices for α are 0.05 or 0.01. A random variable X has a χ^2 - distribution with k degrees of freedom if X is the sum of the squares of k independent random randoın variables that have a N(0,1) distribution. If we expect the statistic to take on a larger value under the alternative hypothesis, then we have a one - sided test. In this case we would determine a cut - off value $X_{k,1-\alpha}$ such that

$$P(X > X_{k,1-\alpha} \mid X \text{ is the statistic of a random sequence}) = \alpha. \tag{6}$$

If we expect the statistic under the alternative hypothesis to be either larger or smaller than that under the null hypothesis, instead of just larger, then we have a two sided test. A two sided test is usually used when the statistic of the random sequence follows a normal distribution. In this case we find two values $X_{\alpha/2}$ and $X_{1-\alpha/2}$, and we reject H_0 if $X>X_{1-\alpha/2}$ or $X<X_{\alpha/2}$. The cut - off points are determined so that

$$\begin{aligned} \alpha/2 &= P(X > X_{1-\alpha/2} \mid X \text{ is the statistic of a random sequence}) \\ &= P(X > X_{\alpha/2} \mid X \text{ is the statistic of a random sequence}). \end{aligned} \tag{7}$$

The first statistical test is frequency test to investigate the randomness of the sequences. Frequency test is designed to compare the number of 0's and the number of 1's in a sequences. Since our sequences should resemble one produced by a binary symmetric source (BSS), we would expect that number of 0's and 1's to be about the same. If n is the length of the sequence we are studying, n_0 is the number of 0's in the sequence, and n_1 is the number of 1's, then our statistic is

$$X = \frac{(n_0 - n_1)^2}{n}. \tag{8}$$

The statistic X follows the χ^2 - distribution with one degree of freedom.

The serial test determines if the number of occurrences of each pair of bits, 00, 01, 10, and 11, are approximately the same. If we let n_{00} be the number of times the pair 00 appears in a sequence of length n, and we define n_{01}, n_{10}, and n_{11} similarly, then our statistic is

$$X = \frac{4}{n-1}\left(n_{00}^2 + n_{01}^2 + n_{10}^2 + n_{11}^2\right) - \frac{2}{n}\left(n_0^2 + n_1^2\right) + 1 \tag{9}$$

where n_0 and n_1 are the number of 0's and 1's respectively. The statistic X approximately follows a χ^2 - distribution with two degrees of freedom.

bit patterns, the poker test studies the number of occurrences of each of the 2^m m-bit patterns, for some integer m.

First we partition a sequence of length n into k m-bit subsequences, where $k = \left\lfloor \frac{n}{m} \right\rfloor$, the greatest integer smaller then n/m. Since there are 2^m possible m-bit subsequences we will let n_i be the number of occurrences of the i^{th} subsequence for $i = 1,...,2^m$. Our statistic for the poker test is then

$$X = \frac{2^m}{k} \sum_{i=1}^{2^m} n_i^2 - k. \tag{10}$$

The statistic X follows a χ^2 - distribution with 2^m-1 degrees of freedom.

The fourth statistical test we will study is called runs test. A run is defined as a subsequences consisting of a repeated single bit. So a run of ones of length t is a subsequence of t consecutive ones that is not preceded or succeeded immediately by a one. In order to perform the runs test we need to know n, the length of the sequence, and m, the number of runs in the sequence. We also need to know k, the length k is at least 5. It is sufficient to estimate k using the formula,

$$k = \left\lfloor \log_2 \frac{n}{10} \right\rfloor. \tag{11}$$

Let n_{0i} and n_{1i} be the number of runs of zeros and ones of length i, for $i \le k-1$. Also let n_{0k} and n_{1k} be the number of runs of zeros and ones of length k or grater. Our test statistic is calculated as

$$X = \sum_{i=1}^{k} \left(\frac{(n_{0i} - y_{0i})^2}{y_{0i}} + \frac{(n_{1i} - y_{1i})^2}{y_{1i}} \right) \tag{12}$$

This statistic approximately follows a χ^2 - distribution with 2k-3 degrees of freedom.

The final test we look at is called the autocorrelation test. The purpose of this test is to see whether there is a correlation between the bits of the sequence as it is given and the bits of a shifted version of the same sequence. If the sequence we wish to test is $S = s_1 s_2 ... s_3$, then we will compare s_i with s_{i+d} if we are considering a shift of size d. If no correlation exists then we would expect s_i to be equal to s_{i+d} approximately half the time.

If we let $\oplus$ denote the XOR operator, or equivalently, modulo-2 addition, then the number of bits not equal to their shifts can be calculated as

$$A(d) = \sum_{i=1}^{n-d} s_i \oplus s_{i+d} \quad d = 1,...,\lfloor n/2 \rfloor. \tag{13}$$

A(d) should follow a binomial distribution.

Under certain conditions, even simple non - linear iterative functions are capable of generating "chaotic" sequences of random numbers. The apparent ability of such functions to provide a source of random numbers is clearly considerable

Under certain conditions, even simple non - linear iterative functions are capable of generating "chaotic" sequences of random numbers. The apparent ability of such functions to provide a source of random numbers is clearly considerable interest in cryptology. The provably unbreakable one - time system, in which plaintext is encrypted by modular addition to a key sequence of random numbers that is used only once, has never gained wide acceptance because of the difficulty of securely distributing the sequences of random numbers, the "one - time pads", to the members of communication network.

If a function were used to generate the random keys, instead of the pads, the key sequence could be generated when and where required, using specific values of the function's tunable parameters. These parameter values would then be the only numbers that have to be disseminated, a task posing a much lower security risk than the safe distribution of one - time pads.

Matthews [4] has recently proposed a new method of generating random numbers for cryptographic purposes using a chaotic function. He suggests that it may be useful "as a replacement for the one - time pad system", [4, p. 29]. The function he proposed is

$$g(x(n+1)) = (\beta+1)(1+1/\beta)^{\beta} x(n)(1-x(n))^{\beta} \qquad (14)$$

with the parameter β in the range $1 \leq \beta \leq 4$ and the initial value of x_0 in the range $0 < x_0 < 1$.

The values of β and x_0 serve as the cryptographic key. Each pair generates a unique sequence of x values when the function is repeatedly applied. The advantage of such a generating function is that its nonlinear form could make cryptanalysis much more difficult than in the case of a linear congruential generator. However, Wheeler [5] subsequently pointed out that Matthews' function may lead in practice to cycling with an unacceptably low period. Based on this idea , a monotonic nonlinear function for key generation was developed [6]. The purpose of Mitchell's paper [6] is to present a modification of Matthews' approach which is immune to wheeler's criticism. A nonlinear pseudorandom number generator based on chaos theory [4] was criticized for producing cycling keys if used with low - precision arithmetic [5]. However, this function was shown to generate suitable pseudorandom key sequences when implemented with high precision arithmetic [7]. Chaos theory provides many formulations that can be used as random generators. The chaotic nature of the Lorenz system of equations makes it a good candidate for pseudorandom number generation. Carroll, Verhagen, and Wong modified the equations to ensure that its sequences are not serially auto correlated. So that the modified system produces extremely long sequences with good random properties.

We took note of all of these papers which suggests that chaotic functions can produce long sequences with good statistical properties. Based on these ideas, we have investigated the suitability of two systems of equations from chaos theory, the Lorenz Strange Attractor and the Chua Strange Attractor. They generates a large set of random numbers in a repeatable fashion, which is, at least in principle, suitable for use as a replacement for one - time pads.

The Chua and the Lorenz equations cannot be solved analytically but require numerical solution. Because of their chaotic nature, the numerical solution depends

both on the numerical technique and the precision of the calculation. The simplest numerical method for solving the equations is the Euler method.

Carroll, Verhagen, and Wong say that from a cryptographer's point of view, the sequence of points might not be useful because sub-sequences of it are likely to become highly self correlated - each point differing from its predecessor by only an infinitesimal amount. But the system is on the verge of instability. They used this properties of the Lorenz system. In this case, if we set the differential time increment Δt to 0.0245 or more, the particle will escape from the gravity of the attractor and spin out into three-dimensional space. They say that if Δt is equal to 0.0245 or more than 0.0245, the modified system can produce three simultaneous number sequences that satisfy statistical tests for the characteristics of randomness. In this way, the chaotic nature of the Lorenz system is destroyed and the system becomes unstable. We claim that the Lorenz and the Chua equations in the chaotic mode can produce pseudorandom sequences. In fact, the point of the numerical solution of the systems obtained using Euler method for small Δt is different from its predecessor by only an infinitesimal amount. However after the third digit, digits begin being different from its predecessor. After this point, we notice that digits diverge the predecessor significantly. This cause us so as to think that the systems can use in order to produce pseudorandom sequences by taking the least significant digits or digits which are near close by the significant digits of the itaretes generated, and converting them to a convenient base, the result can be added to a plaintext to give a ciphertext consisting of characters that have been encrypted using a random keystream. To decrypted the ciphertext, it is only necessary to plug the same values of parameters, and subtract the resulting iterates from the ciphertext.

On the computer used for calculation (Cx486DX2-S), the number of significant digits after the point is 15. So that the numbers in three-dimensional obtained using Euler method were added to each other. then the least significant digits of the numbers reduced modulo the size of the character set (e.g 256).

The equations were iterated 131072 times to produce test sequences for five different initial conditions. Five tests (the frequency test, the serial test for pairs, the poker test, the runs test, and the autocorrelation test) are used to verify that the sequences are random. They satisfied statistical tests for the characteristics of randomness. Further investigations showed that the system produces long sequences with good random properties not only the least significant digits but also other digits except the first and the second digits after the point. The test results of the sequences which are consist of the tenth digit, the eleventh digit, and the twelfth digit were given in this research.

Chaos theory provides a great many other systems that deserve to be investigated so as to produce pseudorandom sequences.

BÖLÜM 1

GİRİŞ

Rasgele benzeri davranışlar içeren çeşitli doğal süreçlerin deterministik kaotik sistemler ile modellenebilmesi ve bu modellerin analizinde kuramsal gelişmelere parelel sayısal ve analog bilgisayar kullanımı, son yirmi yıla damgasını vuran araştırma alanlarından birisini oluşturmuştur. Kaosla ilgili çalışmalar, dinamik sistem kuramını zenginleştirmiş ve bu sayede insanoğlunun, anlaşılmaz görünen bazı doğal olguları kavrayabilmesini mümkün kılmıştır. Türbülans mekanizması, kristallerde büyüyen desenler ve beynin kendine benzer yapısı, bu doğal olaylara yalnızca birkaç örnektir. Kaos günümüzde; astronomi, biyoloji, biyofizik, kimya, mühendislik, jeoloji, matematik, tıp, meteoroloji, plazma, fizik ve hatta sosyal bilimlerde üzerinde çalışılan bir konu olmuştur.

Herhangibir fiziksel olaya bilimsel yaklaşımın temel öğeleri; olayı açıklayan bir modelin yaratılması ve önerilen modele dayalı deneylerle, olayın yinelenebilirliğidir. Bu anlamda, başlangıç koşulları ve tüm parametrelerin belirli olması durumunda, davranışı tam olarak açıklanabilen deterministik modellerin, bilimsel yaklaşımın doğal aygıtları olacakları açıktır. Yeniden oluşturma ve kabul edilebilir doğruluktaki ölçümün mümkün olmadığı fiziksel olaylar, genelde rassal modeller ile açıklanmaktadır. Böyle fiziksel olayları deterministik bir biçimde açıklamanın yolu ise, temel özelliklerinden birisi başlangıç şartlarına aşırı duyarlılık olan ve bu yüzden rasgele benzeri davranışlar gösteren, deterministik kaotik modeller kullanmaktır.

Örneğin Edward Lorenz, havanın karmaşık davranışını modelleyebilmek amacıyla üç - boyutlu kaotik bir sistem kullandı. Bunun yanısıra biyologlar, popülasyondaki düzensiz değişimleri matematiksel olarak modelleyebilmek için rasgele davranış sergileyebilen tek - boyutlu doğrusal olmayan lojistik dönüşümden yararlandılar. Bütün bunlara ilave olarak kaosla ilgili araştırmalar, kuramsal biyolojide

büyük ilerlemelere neden oldu. Çevre bilimciler ve salgın hastalıklarla uğraşan bilimciler, açıklayamadıkları ve bu nedenle rafa kaldırdıkları pek çok olayı, kaos ile ilgili çalışmalar sayesinde kavrayabildiler. Örneğin New York City'deki kızamık salgını ile ilgili kayıtlarda ve Kanada'daki vaşak popülasyonunda iki yüz yıl içinde görülen düzensiz değişimlerde deterministik kaos gözlemlendi. Yine kaos ile ilgili yapılan araştırmaların ışığında, moleküler biyologlar, proteinlere hareket halindeki sistemler gözüyle bakmaya başlarken; fizyologlar da organları durağan yapılar olarak değil, bazı düzenli ve düzensiz osilasyonların oluşturduğu bir sistem olarak düşünmeye başladılar, [1].

Deterministik kaotik modeller, yalnızca karmaşık dinamiklere sahip bu tür olay ve sistemleri açıklamak için kullanılmazlar, bunların yanı sıra mühendislik problemlerinin çözümü amacıyla da kullanılırlar. Örneğin geniş bantlı olmalarından dolayı gürültü işaretlerine benzeyen ve kanalda senkronizasyon sağlanmadan demodüle edilemeyen kaotik işaretler, bu özellikleri sayesinde güvenli haberleşme sistemlerinde yaygın bir kullanım alanı kazandı. Bu anlamda Heidari ve Bateni, kaotik dizilerin, yaygın - bantlı ("spread - spectrum") haberleşme sistemlerinde, sanki - gürültü ("pseudo - noise") dizilerinin yerine kullanılmalarını önerdiler, [2]. Bundan başka Kevin, Oppenheim, Steven ve Wornell tarafından mesaj işareti, Lorenz sisteminden elde edilen kaotik işaretle maskelendi ve böylece haberin güvenli bir şekilde iletimi için gerekli sistem oluşturuldu, [3]. Bu çalışmaların yanında, ancak başlangıç koşullarının kesin olarak belirlenmesi durumunda nasıl davranacağı tamamıyla bilinebilen aksi halde ise rasgele davranış sergileyen kaotik sistemlerin, başlangıç şartlarına aşırı duyarlık göstermesi ve bu sayede birbirinden farklı çok sayıda dizinin elde edilebilmesi ve kriptografide kısa bir anahtar ("key") yardımıyla uzun rasgele diziler üretebilen algoritmalara duyulan ihtiyaç, araştırmacıları kriptografide kullanılan anahtar dizilerinin ("key stream"), bu tür sistemlerin ürettiği dizilerle yerdeğiştirebileceği konusunda çalışmaya yöneltti. Kriptografide sanki - rasgele dizi üreteci olarak, doğrusal kongruens üreteçleri ("linear congruence generators"), doğrusal ve doğrusal olmayan geribeslemeli yazıcılar ve bu tekniklerin, özellikle doğrusal geribeslemeli ötemeli yazmaçların çeşitli şekillerde birleşimleriyle oluşturulan sistemler kullanılırken, kaos kuramının anlaşılmaya başlanması ile bu yöntemlerin

yerine yukarıda bahsedilen nedenlerden dolayı kaotik sistemleri öneren çalışmalar şu şekilde özetlenebilir.

Kaotik sistemlerin kriptografide kullanımına ilişkin ilk inceleme, Robert A. J. Matthews'in kırılamaz olduğu tanıtlanabilir tek - zamanlı sistemlerle ("one - time pad") yer değiştirebileceğini söylediği biyolojik popülasyonun modellenmesi amacıyla kullanılan lojistik dönüşümün genelleştirilmiş şeklini öneren çalışmasıdır, [4]. Matthews bu çalışmasında, anahtar dizilerinin, dönüşüm sonucu bulunan gerçel sayıların en anlamsız iki dijitinin oluşturduğu sayıların uygun moda çevrilmesiyle oluşturulmasını önerdi. Bu çalışmanın ardından Daniel D. Wheeler, Matthews'in önerdiği kaotik dönüşümü ufak çevrimler oluşturduğu için eleştirdi ve bu nedenle dönüşümün, ciddi kriptografik uygulamalarda kullanılamayacağını ifade etti, [5]. Daha sonra Douglas W. Mitchell, anahtar dizisi üretmek için kaotik dönüşümlerden çok monotonik dönüşümlerin kullanılması gerektiğini savundu ve bu yol ile üretilen anahtar dizilerinin de, sınırlı, çevrimsiz ve monotonik olmadığı için kaotik olduğunu iddia etti, [6]. Mitchell, önerdiği monotonik dönüşüm sonucu bulunan gerçel sayıların, Matthews'in önerdiği yöntemle anahtar dizileri üretebileceğini ifade etti. Daha sonra Daniel D. Wheeler ve Robert A. J. Matthews, Wheeler tarafından kısa periyotlu çevrimler oluşturduğu için eleştirilen Matthews'in kaotik dönüşümünü, Cray Y - MP ile gerçekleştirdiler ve elde ettikleri sonuçlara dayanarak, genelleştirilmiş kaotik dönüşümün pratik uygulamalar için yeterli çevrim uzunluğuna sahip diziler üretebileceğini söylediler, [7]. Onlar bu çalışmalarında, çevrimlerin ve kriptografik uygulamalar için uygun olmayan sabit dizilerin oluştuğu durumları kontrol eden bir altprogram içeren genelleştirilmiş kaotik algoritmanın ikili prezisyonla gerçekleştirilmesiyle, ciddi kriptografik uygulamalar için uygun diziler elde edilebileceğini belirttiler. Ardından Carroll, Jeff Verhagen ve Perry T. Wong, sanki - rasgele dizi üretmek için çok - boyutlu kaotik sistemlerin, tek - boyutlu kaotik sistemlerden daha iyi sonuçlar verebileceğini söylediler ve bu amaç ile Lorenz sistemini incelediler, [8]. Onlar bu çalışmalarında, Lorenz sisteminin nümerik çözümünün birbirine çok yakın sayılar üretmesinden dolayı kriptografik uygulamalar için bu şekliyle yararlı olamayacağını ve sistemin, Δt aralığının büyük tutulduğu uygun bir algoritma ile yeniden düzenlenmesiyle rasgele sayılar üretebileceğini söylediler.

Bu çalışmada, sanki - rasgele dizi üretmek için kullanılan doğrusal kongruens üreteçler, doğrusal ve doğrusal olmayan geribeslemeli yazıcılar yerine, çok - boyutlu sürekli - zamanlı kaotik sistemlerin kullanılması önerildi. Sanki - rasgele diziler, rasgele görünüşlü olmalarına rağmen, kullanıcılar tarafından aynı parametre değerlerinin ve aynı makinelerin kullanılmasıyla tekrar üretilebilirler. Rasgele ikili bir dizi, her bir sembolünün 1/2 olasılıkla +1 veya 1/2 olasılıkla -1 değerini aldığı istatistiksel olarak bağımsız sembollerden oluşan stokastik bir süreçtir. Sanki - rasgele bir dizinin özilinti ("autocorrelation") fonksiyonu, rasgele ikili bir dizinin, örneğin sınırlı bantlı beyaz gürültünün özilintisine benzerdir. Belirli bir algoritma ile üretilmelerine rağmen bu diziler, 0 ve 1 rakamlarının dizi içinde dengeli dağılması gibi rasgele dizilerin sahip olduğu pek çok özelliği bünyelerinde barındırırlar. Sanki - rasgele diziler, yaygın - bantlı haberleşmede, şifrelemede, gürültü üretiminin gerektiği benzetimlerde ve daha birçok alanda kullanılmaktadır.

Gerçek mesajın, rasgele sayılardan oluşan ve mesaj ile eş uzunluklu bir anahtar dizisiyle XOR işlemi kullanılarak şifrelenmesi anlamına gelen ve kırılamaz olduğu tanıtlanabilir, Matthews (1989), tek - zamanlı sistemler, tek seferlik kullanılan uzun rasgele dizilerin güvenli bir şekilde dağıtılmasından kaynaklanan zorluklardan dolayı yaygın bir kullanım alanı kazanamamıştır. Kaotik sistemler bu probleme pratik bir çözüm getirebilirler. Çünkü kaotik sistemlerin parametrelerine ve başlangıç koşullarına duyarlığından dolayı, ayar parametrelerindeki ufak değişimlerle birbirinden tamamen farklı çok sayıda dizi üretmek mümkündür. Buna ilave olarak kaotik sistemlerin çok küçük hatalara duyarlığından dolayı, dönüşümün tekrarlama değeri tam olarak bilinmediği müddetçe, anahtar dizisinin tümünün doğru olarak yeniden hesaplanmasına imkan yoktur.

Doğrusal sistemlerin çözümü, özdeğerler ve özvektörler cinsinden üstel fonksiyonlarla temsil edilebilirk, doğrusal olmayan çok az sayıda sistemin, bilinen fonksiyonlar cinsinden ifade edilebilen çözümleri vardır. Bu nedenle bu tür sistemler, bilgisayarlar yardımıyla nümerik yaklaşımlarla çözülebilirler. Bir gerçel sayının bilgisayarda sınırlı sayıda rakamla ifade edilebilmesinden dolayı, kuvantalama hataları meydana gelir. Nümerik yaklaşımlarla çözülerek kriptografide kullanılan kaotik sistemler ile, bilgisayarın sınırlı prezisyonundan dolayı, ilk bakışta kriptografik uygulamalar için yeterince uzun periyotlu dizilerin üretilemeyeceği düşünülebilir.

Matthews'in önerdiği genelleştirilmiş kaotik dönüşümün Cray Y - MP ile gerçekleştirildiği çalışma, Wheeler (1991), çevrim uzunluğunun, bilgisayarın prezisyonu ile üstel olarak arttığını ve böylece bu problemin, prezisyonun arttırılmasıyla ortadan kaldırılabileceğini gösterdi.

şimdiye kadar yapılan çalışmaların, tek - boyutlu kaotik bir sistem olan lojistik dönüşümün genelleştirilmiş şeklinin, kriptografide kullanımını destekler yönde olması, kısa periyotlu çevrimlerin kullanılan bilgisayarın prezisyonunun arttırılmasıyla üstesinden gelinebileceğinin gösterilmesi ve değiştirilmiş olmasına rağmen Lorenz sisteminden elde edilen dizilerin istatistiksel rasgelelik testlerini geçmesi, biz iki veya daha fazla boyutlu kaotik sistemlerin, kriptografik uygulamalar için daha yararlı olabileceği fikrine götürdü. Bu amaç ile Chua ve Lorenz çekicileri Bölüm 5'de ele alındı ve bu sistemlerin davranışları genişçe incelendi. Bölüm 6'da, kriptografi ve kriptografide sanki - rasgele dizi üretiminde yararlanılan üreteçler açıklandıktan sonra, rasgeleliğin belirlenmesi amacıyla kullanılan istatistiksel testlerin ve bu testlerin değerlendirilmesi için göz önüne alınması gereken istatistiksel hipotezlerin üzerinde duruldu. Bölüm 7'de, kaotik sistemlerin kriptografide kullanımına ilişkin yapılan araştırmalar sunulduktan sonra, Bölüm 8'de Chua ve Lorenz çok - boyutlu kaotik sistemlerinin Matthews'in önerdiği yöntem ile kriptografide sanki - rasgele dizi üreteci olarak kullanılması ele alındı. Bu sistemlerin nümerik olarak çözülmesiyle elde edilen üç farklı dizideki gerçel sayıların en anlamsız üç dijitiyle meydana getirilen sayıların, kullanılan karakter sayısına (256) göre modülünün alınmasıyla oluşturulan anahtar dizilerinin rasgeleliğinin belirlenmesi amacıyla, dizilere rasgele istatistiksel testler (frekans testi, seri test, poker testi, hareket testi ve özilinti testi) uygulandı ve sonuçların iyi olduğu görüldü. Biz kaos kuramının bu anlamda araştırmayı hak eden oldukça fazla çok - boyutlu kaotik sistemi bünyesinde barındırdığına inanıyoruz.

Çalışmanın ana konusunu, bu inceleme teşkil etmekle birlikte, kaos kuramını temel alarak yapılan gerek bu incelemeden gerekse daha önceden yapılan çalışmalardan dolayı, kaos kuramını oluşturan tanımların ve teoremlerin verilmesinde de yarar görüldü. Bu amaç ile Bölüm 2'de doğrusal olmayan dinamik sistemlerin genel özellikleri, tanımlar ve teoremler yardımıyla sunulduktan sonra, Bölüm 3 ile "kaos" olgusuna gelindi ve burada, kaotikliğe neden olan üç özellik açıklandı. Bölüm 4'te, sanki - rasgele sayılar üretmek için yapılan kriptografik araştırmalarda temel

olarak alınıp üzerinde bir takım değişiklikler yapılan ve çok - boyutlu kaotik sistemlerdeki kaos üretme mekanizmasının açıklanması amacıyla kullanılan, tek - boyutlu doğrusal olmayan lojistik dönüşümün, gerçekte yoğurma işlemine denk olduğu gösterildi.

BÖLÜM 2

DOĞRUSAL OLMAYAN DİNAMİK SİSTEMLER

Bölüm 2'de kaos teorisini anlaşılması için gerekli tanımlar ve teoremler sunulacaktır. Genel tanımlar altbölümünde, fonksiyon ile başlayan bu tanımlar zinciri, C^r fonksiyonlar, homeomorfizm, C^r diffeomorfizm, otonom sistem otonom olmayan sistem, vektör alanları, çözüm, orbit, yörünge ve integral eğrisi tanımları ile son bulacaktır. Vektör alanlarını bazı genel özellikleri başlığı altında ise,
$\dot{x}(t) = f(x,t),\ t \geq 0,\ x(0) = x_0$ diferansiyel denklemi için Peano varlık teoremi, Picard'ın (zamanda) yerel varlık ve teklik teoremi, genel varlık ve teklik teoremi ve C^r varlık teklik teoremi incelenecektir. Ayrıca yine bu altbölümde, herhangibir çözümün Liapunov anlamında kararlılığı ve asimptotik kararlılığı tanımları verilecektir.

$\dot{x}(t) = f(x,t)$ sistemi için denge noktası tanımı ve denge noktasının fiziksel anlamı, denge noktaları alt bölümünde irdelenecektir. Yine bu altbölümde, x^* denge noktasının kararlılığını ve asimptotik kararlılığını doğrudan belirlemeye imkan sağlayan Liapunov'un dolaysız yönteminden söz edilecektir. Ayrıca doğrusal sistem davranışlarının incelenerek doğrusal olmayan sistem davranışları hakkında yorum yapılabilmesini olanaklı kılan ve Liapunov'un ilk metotu olarak da bilinen Liapunov'un dolaylı yöntemi açıklanmakla kalmayıp hiperbolik denge noktası tanımı yine bu altbölümde verilecek ve denge noktaları da, vektör alanları ve dönüşümler için sınıflandırılacaktır.

Periyodik çözümler altbölümünde vektör alanları ve dönüşümler için periyodik çözüm tanımı ve hangi şartlar altında basit bağlantılı bir D bölgesinin periyodik orbitler içermeyeceğini ifade eden Bendixson kriteri sunulacaktır. Bölüm 2.5'te Poincaré - Bendixson teoreminin anlaşılabilmesi için gerekli α ve ω limit noktaları ile α ve ω limit kümelerinin tanımları verildikten sonra, limit çevrim tanımı ve bir düzlemde

ile α ve ω limit kümelerinin tanımları verildikten sonra, limit çevrim tanımı ve bir düzlemde denge noktası içermeyen sıkı bir limit kümenin, kapalı bir orbit olduğunu ifade eden Poincaré - Bendixson teoremi sunulacaktır. Sürekli zamanlı sistemler ile ayrık zamanlı sistemler arasında bir köprü görevi gören Poincaré fonksiyonu ve periyodik çözümlerin kararlılığının karakteristik ("floquet") çarpanlarla belirlenmesi Bölüm 2.6'da, sanki periyodik çözüm tanımı ise Bölüm 2.7'de sunulacaktır.

2.1 Genel Tanımlar

Tanım 2.1 (Fonksiyon) Eğer değer bölgesindeki her bir x değeri ile bir y değerini bağdaştıran bir kural veya metod verilmişse, y değerine başka bir x değişkeninin fonksiyonu denir.

Değeri verilen x değişkeninin değerine bağlı olan y değişkenine bağlı değişken, x değişkenine ise bağımsız degişken denir. Kural veya metod, karşılıklı değerlerin bir tablosu, bir grafik veya bir denklem olabilir.

Tanım 2.2 (C^r Fonksiyonlar) Eğer bir fonksiyon r kez türetilebiliyorsa ve türevlerin her biri sürekli ise; bu fonksiyon C^r sınıfındandır.

Tanım 2.3 (Homeomorfizm) Eğer $f(\cdot)$ fonksiyonu birebir, örten ve sürekli ise ve $f^{-1}(\cdot)$ de sürekli ise; $f(\cdot)$ fonksiyonu, bir homeomorfizmdir.

Tanım 2.4 (C^r diffeomorfizm) $g(x)$ dönüşümü evrilebilir ise ($g^{-1}(x)$ varsa) ve $g(x)$ ile $g^{-1}(x)$ dönüşümlerinin her ikiside C^r sınıfından ise; $g(x)$ dönüşümü, C^r diffeomorfizmdir.

$$\dot{x} = f(x,t), \tag{2.1}$$

$$x \mapsto g(x) \tag{2.2}$$

denklemlerinde U, R^n'de açık bir küme olmak üzere, $x \in U \subset R^n$, $t \in R^1$ dir. (2.1), vektör alanı veya bayağı diferansiyel denklem olarak; (2.2) ise dönüşüm veya fark denklemi olarak adlandırılır. (2.1) denkleminin çözümü demekle $I \subset R^1$ aralığından R^n'e bir dönüşüm kast edilmektedir;

$$x:I \to R^n, \quad t \mapsto x(t) \tag{2.3}$$

öyleki x(t), (2.1) denklemini sağlar. Yani $\dot{x}(t) = f(x(t),t)$ dir. Bunların ışığı altında çözümün tanımı, şu şekilde verilebilir.

Tanım 2.5 (Çözüm) t_0 anında x_0 başlangıç şartıyla başlayan ve $\dot{x} = f(x,t)$ ifadesini, $x(t_0,t_0,x_0) = x_0$ olmak üzere, sağlayan $x(\cdot\ , t_0\ , x_0)$: $I \to R^n$ fonksiyonuna, t_0 anında x_0 ile başlayan çözüm denir.

x dönüşümü R^n'de geometrik yorumu olan bir eğriye sahiptir. (2.1) denklemi, bu eğrinin her bir noktasında tanjant vektörünü verir. Bundan dolayı (2.1) denklemine vektör alanı olarak da bakılır. Vektör alanının tanımı şu şekilde verilir.

Tanım 2.6 (Vektör Alanı) $\dot{x} = f(x,t)$ denkleminde, $f(\cdot\ ,\cdot)$ fonksiyonu sürekli ise;

$f(\cdot\ ,\cdot)$: $Ux\ I \to R^n$ fonksiyonu bir vektör alanı oluşturur,

$$VA = \{ y \in R^n \mid f(x,t) = y \}. \tag{2.4}$$

(2.1) denklemindeki bağımlı değişkenlerin uzayı (yani R^n), (2.1) denkleminin faz uzayı olarak adlandırılır.

Çözüm eğrileri, yörünge ("trajectory") veya orbit olarak adlandırılır. $x(t,t_0,x_0)$'ın varolduğu her t için grafiği ise, integral eğrisi olarak adlandırılır ve tanımı şu şekilde verilir.

Tanım 2.7 (İntegral Eğrisi) I, çözümün varolduğu zaman aralığı olmak üzere,

$$x(t,t_0,x_0) = \{(x,t) \in R^n x R^1 \mid x = x(t,t_0,x_0),\ t \in I\} \tag{2.5}$$

grafiği, integral eğrisini oluşturur.

x_0, (2.1) denkleminin faz uzayında bir nokta olsun. x_0'dan geçen ve $Or(x_0)$ ile işaret edilen orbit, x_0'dan geçen eğri üzerindeki noktalar kümesidir. Daha açık ve kesin bir tanım şu şekildedir.

Tanım 2.8 (Orbit) $x_0 \in U \in R^n$ olmak üzere bir t_0 anında x_0 noktasından geçen orbit, çözümün bütün t anlarında aldığı değerler kümesidir,

$$Or(x_0) = \{x \in R^n \mid x = x(t, t_0, x_0),\ t \in I\}. \tag{2.6}$$

Herhangibir $T \in I$ için $Or(x(T, t_0, x_0)) = Or(x_0)$ olduğuna dikkat ediniz.

Çözüm, integral eğrileri ve orbitler arasındaki farkı izah etmek amacıyla,

$$\begin{aligned} \dot{u} &= v, \\ \dot{v} &= u \end{aligned} \qquad (u, v) \in R^1 x R^1. \tag{2.7}$$

sistemi göz önüne alınsın. $t = 0$ anında $(u, v) = (1,0)$ noktasından geçen çözüm, $(u(t), v(t)) = (\cos t, -\sin t)$ şeklindedir. $(u, v) = (1,0)$ noktasından geçen integral eğrisi,

$$\{(u, v, t) \in R^1 x R^1 x R^1 \mid (u(t), v(t)) = (\cos t, -\sin t),\ \forall t \in R^1\} \tag{2.8}$$

ifadesiyle verilir. $(u, v) = (1,0)$ noktasından geçen orbit ise, $u^2 + v^2 = 1$ şeklinde tanımlanan bir dairedir.

Tanım 2.9 (Akış) t_0 anındaki x_0 başlangıç şartını, t anındaki x'e dönüştüren,

$$\phi_t(\cdot) = x(t, t_0, \cdot): U \to R^n \tag{2.9}$$

fonksiyonuna akış denir.

Tanım 2.10 (Otonom Olmayan Vektör Alanları) Zamana bağlı vektör alanları, otonom olmayan vektör alanları olarak adlandırılır.

Tanım 2.11 (Otonom Vektör Alanları) Zamandan bağımsız vektör alanları, otonom vektör alanları olarak adlandırılır.

İki - boyutlu, otonom olmayan vektör alanları ile iki - boyutlu otonom vektör alanlarının dinamikleri arasında büyük bir fark vardır: kaosun meydana gelmesi, otonom olmayan durumda mümkünken, otonom durumda mümkün değildir.

2.2 Vektör Alanlarının Bazı Genel Özellikleri

Teorem 2.1 (Picard'ın Zamanda Yerel Varlık ve Teklik Teoremi)
$f:R_+xR^n \to R^n$ ve $x(t) \in R^n$ olmak üzere,

$$\dot{x}(t) = f(x(t),t) \quad t \geq 0, \quad x(0) = x_0 \tag{2.10}$$

denklemi ele alınsın. f fonksiyonu t ve x'de sürekli olsun ve aşağıdaki şartları sağlasın:
$B = \{x \in R^n : \|x - x_0\| \leq r\}$ ($\|.\|$ normu ifade eder) olmak üzere,

$$\|f(x,t) - f(y,t)\| \leq k\|x - y\|, \quad \forall x,y \in B, \quad \forall t \in [0,T] \tag{2.11}$$

$$\|f(x_0,t)\| \leq h, \quad \forall t \in [0,T] \tag{2.12}$$

ifadelerini sağlayan sonlu T, r, h, k sabitleri vardır. Bu durumda (2.10)'un $[0,\delta]$ aralığında tek bir çözümü vardır. Burada $\rho > 1$ olmak üzere,

$$h\delta e^{k\delta} \leq r, \tag{2.13}$$

$$\delta \leq \min\left(T, \frac{\rho}{k}, \frac{r}{h+kr}\right) \tag{2.14}$$

ifadeleri sağlanır.
Teoremin tanıtı için,[9]'a bakınız.
$\|f(x,t) - f(y,t)\| \leq k\|x - y\|$, $\forall x,y \in B$, $\forall t \in [0,T]$ ifadesi lokal Lipschitz koşulu; k ise Lipschitz sabiti olarak bilinir.

Teorem 2.2, eğer f genel Lipschitz koşulunu sağlarsa, (2.10) denkleminin $[0,\infty)$ aralığında tek bir çözüme sahip olduğunu söyler.

Teorem 2.2 (Genel Varlık ve Teklik Teoremi) $\forall t \in [0,\infty)$ için, aşağıdaki koşulları sağlayan sonlu k_T ve h_T sabitlerinin olduğu varsayılsın:

$$\|f(x,t)-f(y,t)\| \le k_T \|x-y\|, \quad \forall x,y \in R^n, \quad \forall t \in [0,T] \tag{2.15}$$

$$\|f(k_0,t)\| \le h_T, \quad \forall t \in [0,T]. \tag{2.16}$$

Bu durumda $\forall T \in [0,\infty)$ için (2.10) ile tanımlanan sistem, $[0,T]$ üzerinde tek bir çözüme sahiptir.

Teoremin tanıtı için,[9]'a bakınız.

Teorem 2.3 (Peano Varlık Teklik Teoremi) (2.10) ile tanımlanan sistem, (x_0,t_0)'da sürekli olsun. Bu durumda öyle bir $\varepsilon > 0$ sayısı vardırki; (2.10), ε'nun belirlediği $t \in [t_0-\varepsilon, t_0+\varepsilon]$ zaman aralığında en az bir çözüme sahiptir.

Tanıt için [10]'a bakınız.

Teorem 2.4, başlangıç değerlerine göre varlık, teklik ve türetilebilirlik üzerinedir.

$f(x,t)$, $U \subset R^n x R^1$ açık setinde C^r, $r \ge 1$, sınıfından olmak üzere; aşağıdaki vektör alanı düşünülsün,

$$\dot{x} = f(x,t). \tag{2.17}$$

Teorem 2.4 (Başlangıç Değerlerine Göre Varlık, Teklik ve Türetilebilirlik) $(\mathbf{x}_0,\mathbf{t}_0) \in \mathbf{U}$ olsun. $|t-t_0|$ ifadesinin yeterince küçük değerleri için, (2.17) eşitliğinin t_0 anında x_0 noktasından geçen ve $\mathbf{x}(t_0,t_0,x_0) = x_0$ olmak üzere $\mathbf{x}(t,t_0,x_0)$ ile gösterilen bir çözümü vardır. Bu çözüm tektir. Yani (2.17) sisteminin t_0 anında x_0'dan geçen başka bir çözümü, her iki çözümün var olduğu ortak bir aralıkta $\mathbf{x}(t,t_0,x_0)$ çözümüyle aynı olmalıdır. Bunlara ilave olarak $\mathbf{x}(t,t_0,x_0)$, t, t_0 ve x_0'ın C^r sınıfından bir fonksiyonudur.

Tanıt için [11], [12] veya [13]'e bakınız.

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \quad \mathbf{x} \in \mathbf{R}^n \tag{2.18}$$

sistemi ele alınsın.

Tanım 2.12 (Liapunov Anlamında Kararlılık) x(t), (2.18) sisteminin $x^*(t)$'den farklı herhangibir çözümü olmak üzere, verilen rasgele bir $\varepsilon > 0$ sayısı için $\delta = \delta(\varepsilon) > 0$ sayısı varsa, öyleki bu sayının $\|x^*(t_0) - x(t_0)\| < \delta$ ifadesiyle belirlediği ilk koşullarla başlayan $x^*(t)$ ve x(t) çözümleri $t > t_0$ anları için $\|x^*(t_0) - x(t_0)\| < \varepsilon$ ifadesini sağlıyorsa; $x^*(t)$ çözümüne, Liapunov anlamında kararlıdır denir.

Tanım 2.13 (Liapunov Anlamında Asimptotik Kararlılık) Eğer $x^*(t)$ çözümü Liapunov anlamında kararlıysa ve $\|x^*(t_0) - x(t_0)\| < r$ olduğunda,

$\lim_{t\to\infty} \|x^*(t) - x(t)\| = 0$ ifadesini sağlayan bir $r < \infty$ sabiti varsa, $x^*(t)$ çözümüne, Liapunov anlamında asimptotik kararlıdır denir.

2.3 Denge Noktaları

$f: R_+ x R^n \to R^n$ olmak üzere,

$$\dot{x}(t) = f(x,t) \tag{2.19}$$

vektör alanı düşünülsün.

Tanım 2.14 (Denge Noktası) $x_0 \in R^n$ vektörü, $f(x_0, t) = 0, \forall t \geq t_0$ ifadesini sağlıyorsa, (2.19) ile verilen sistemin $t_0 \in R_+$ anında bir denge noktasıdır.

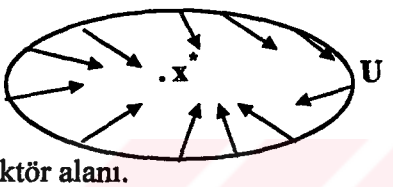
Eğer x_0, (2.19)'in t_0 anında bir denge noktasıysa; tüm $t_1 \geq t_0$ anları için de (2.19)'in bir denge noktasıdır. Denge noktasının fiziksel önemi şu şekilde ifade edilebilir. $x_0 \in R^n$, (2.19)'in t_0 anında bir denge noktası ise, $t_1 \geq t_0$ olduğunda,

$$\dot{x}(t) = f(x,t),\ t \geq t_1;\ x(t_1) = x_0, \tag{2.20}$$

sistemi tek bir çözüme sahiptir: $x(t) = x_0 \quad \forall t \geq t_1$.

2.3.1 Liapunov'un Dolaysız Yöntemi

Denge noktası $\mathbf{x}^*$ ile verilen bir vektör alanı düşünülsün ve sabit noktaların kararlı olup olmadığı belirlenmek istensin. Kabaca $\mathbf{x}^*$ sabit noktası civarında çözümlerin başlayıp ve yine tüm pozitif zamanlar için orada kaldığı bir U bölgesi varsa ; $\mathbf{x}^*$ sabit noktası kararlıdır. Vektör alanı U bölgesinin sınırına teğet ise veya $\mathbf{x}^*$ noktasına doğru yönelmişse, bu koşul sağlanır ve bu koşul, U bölgesi $\mathbf{x}^*$ üzerine daralsa bile doğru kalır.



Şekil 2.1. U sınırındaki vektör alanı.

$$\begin{aligned}\dot{x}_1 &= f(x_1,x_2)\\ \dot{x}_2 &= g(x_1,x_2)\end{aligned} \qquad (x_1,x_2)\in \mathbf{R}^2 \tag{2.21}$$

denge noktası (x_1^*,x_2^*) ile gösterilen (2.21) vektör alanı göz önüne alınsın. $V(x_1,x_2)$fonksiyonu, $V(x_1^*,x_2^*)=0$ koşulunu sağlayan skaler bir fonksiyon olsun, $V:\mathbf{R}^2 \to \mathbf{R}^1$. Öyleki $V(x_1,x_2)=c=$ **sabit** ifadesini sağlayan noktaların geometrik yeri, c'nin farklı değerleri için, (x_1^*,x_2^*) denge noktasını $V(x_1,x_2)>0$ koşulu ile çevreleyen kapalı eğrilerdir. Vektör alanı, (x_1^*,x_2^*) noktasını çevreleyen her bir eğriye teğetse veya içeri doğru yönelmişse, $[\nabla V(x_1,x_2)]^T\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix}\le 0$ koşulu sağlanır. Bu ifade, V'nin yörüngeler boyunca türevini işaret eder. Bu düşünceleri açık ve kesin bir şekilde ifade eden genel bir teorem şu şekilde verilir.

Teorem 2.5 (Liapunov'un Dolaysız Yöntemi) Aşağıdaki vektör alanı düşünülsün,

$$\dot{x}=f(x),\ x\in \mathbf{R}^n. \tag{2.22}$$

$\mathbf{x}^*$, bu vektör alanının sabit bir noktası ve V, $\mathbf{x}^*$'in U komşuluğunda tanımlı C^1 sınıfından bir fonksiyon olsun, $\mathbf{V:U \to R}$. Öyleki V fonksiyonu, aşağıdaki koşulları sağlar.

i) $\mathbf{V(x^*)=0}$ ve eğer $\mathbf{x \neq x^*}$ ise $\mathbf{V(x)>0}$.

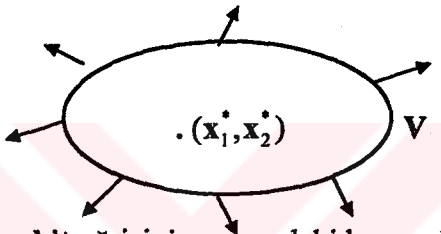
ii) $\mathbf{x \in U-\{x^*\}}$ için $\mathbf{\dot{V}(x) \leq 0}$.

Bu durumda $\mathbf{x}^*$ noktası Liapunov kararlıdır. Eğer

iii) $\mathbf{x \in U-\{x^*\}}$ için $\mathbf{\dot{V}(x)<0}$.

koşulu da sağlanırsa, $\mathbf{x}^*$ Liapunov anlamında asimptotik kararlıdır.

Tanıt için [14]'e bakınız.



Şekil 2.2. $\mathbf{V=}$ **sabit** eğrisinin sınırındaki bazı noktaların $\nabla \mathbf{V}$ vektörleri.

2.3.2 Liapunov'un Dolaylı Yöntemi (Doğrusallaştırma)

$$\dot{\mathbf{x}}=\mathbf{f(x)},\ \mathbf{x} \in \mathbf{R}^n \tag{2.23}$$

(2.23) sisteminin herhangibir $\mathbf{x}^*(t)$ çözümünün kararlı olup olmadığı hakkında bir yorum yapılabilmesi için, $\mathbf{x}^*(t)$'ye yakın çözümlerin doğası anlaşılmalıdır. x ifadesi aşağıdaki gibi verilsin,

$$\mathbf{x=x^*(t)+y} \tag{2.24}$$

(2.23) ifadesinin, (2.24) ifadesinde yerine yazılmasıyla elde edilen ifade, $\mathbf{x}^*(t)$ civarında Taylor formülüne açılabilir,

$$\dot{\mathbf{x}}=\dot{\mathbf{x}}^*+\dot{\mathbf{y}}=\mathbf{f(x^*(t))+Df(x^*(t))y+O(\|y\|^2)}. \tag{2.25}$$

Df, f fonksiyonunun türevini; $\|.\|$ ise R^n, de tanımlı bir normu gösterir. $\dot{x}^*(t) = f(x^*(t))$ ifadesi kullanılarak (2.25) ifadesi (bu ifadenin yazılabilmesi için f, en azından iki kez türetilebilmelidir) aşağıdaki eşitliğe indirgenir,

$$\dot{y} = Df(x^*(t))y + O(\|y\|^2). \tag{2.26}$$

Kararlılık problemine bir çözüm getirilebilmesi için, $x^*(t)$'ye gelişigüzel yakın çözümlerin davranışlarıyla ilgilenilmelidir. Bu problemin, aşağıda verilen doğrusal sistemin incelenerek çözümlenmesi oldukça mantıklı görünmektedir,

$$\dot{y} = Df(x^*(t))y. \tag{2.27}$$

$x^*(t)$ çözümünün kararlılığı hakkında bir yorum getirilebilir. Aşağıda verilen iki maddenin incelenmesi ile başka şekilde de yorum yapılabilir:

1. (2.27) eşitliğiyle tanımlanan sistemin $y = 0$ çözümünün kararlı olup olmadığının belirlenmesi,
2. $y = 0$ çözümünün kararlılığının (veya kararsızlığının), $x^*(t)$ çözümünün kararlılığını (veya kararsızlığını) doğurmasının gösterilmesi.

Zamana bağlı katsayılar içeren doğrusal bir diferansiyel denklemin çözümünün bulunması için genel bir metot yoktur. Bu nedenle 1. adımın cevaplanması, orjinal sorunun cevaplanması kadar zor olabilir. Buna rağmen $x^*(t)$, bir denge noktası ise, yani $x^*(t) = x^*$ ise, $Df(x^*(t)) = Df(x^*)$ matrisi sabit katsayılı bir matristir ve (2.27) sisteminin $t = 0$ anında $y_0 \in R^n$ noktasından geçen çözümü kolayca yazılabilir,

$$y(t) = e^{Df(x^*)t}y_0 \tag{2.28}$$

Teorem 2.6 (Denge Çözümünün Asimptotik Kararlılığı) $Df(x^*)$ matrisinin tüm özdeğerlerinin negatif reel kısımlı olduğu varsayılsın. Bu durumda (2.23) eşitliğiyle verilen doğrusal olmayan otonom durum denklemlerinin, $x = x^*$ denge çözümü asimptotik kararlıdır.

Tanıtı için [14]'e bakınız.

2.3.3 Hiperbolik Denge Noktaları

Doğrusal vektör alanına ilişkin tüm özdeğerlerin gerçel kısmı sıfırdan farklıysa; doğrusal olmayan vektör alanın denge çözümü civarındaki dinamiği, doğrusal vektör alanın dinamiği ile aynıdır. Bu tür denge noktalarına özel bir isim verilir.

Tanım 2.21 (Hiperbolik Denge Noktası) $\mathbf{x} \in \mathbf{R}^n$ olmak üzere, $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ sisteminin bir sabit çözümü, $\mathbf{x} = \mathbf{x}^*$ olsun. $\mathbf{Df}(\mathbf{x}^*)$'ın özdeğerlerinin hiçbiri sıfır gerçel kısımlı değilse; $\mathbf{x}^*$ denge noktasına, hiperboliktir denir.

Eğer doğrusallaştırılmış sistemin özdeğerlerinin bazıları sıfırdan büyük gerçel kısımlı; geri kalanları ise sıfırdan küçük gerçel kısımlıysa, vektör alanın hiperbolik sabit noktası, eyer noktası olarak adlandırılır. Doğrusallaştırılmış sistemin tüm özdeğerleri negatif gerçel kısımlıysa; vektör alanın hiperbolik sabit noktası, kararlı düğüm veya çukur ("sink") olarak adlandırılır. Doğrusallaştırılmış sistemin tüm özdeğerleri pozitif gerçel kısımlıysa; vektör alanın hiperbolik sabit noktası, kararsız düğüm veya kaynak ("source") olarak adlandırılır. Doğrusallaştırılmış sistemin özdeğerleri saf sanalsa ve sıfırdan farklıysa; hiperbolik olmayan sabit nokta, merkez olarak adlandırılır.

Tablo 2.1, $\dot{\mathbf{x}}(\mathbf{t}) = \mathbf{Ax}(\mathbf{t}),\ \mathbf{t} \geq 0$ doğrusal vektör alanı için denge noktasının sınıflandırılmasını gösterir.

Tablo 2.1. İkinci dereceden doğrusal bir sistem için denge noktalarının sınıflandırılması.

A'nın Özdeğerleri	Denge Noktasının Tipi
λ_1, λ_2 gerçel, $\lambda_1 < 0, \lambda_2 < 0$	Kararlı Düğüm ("Stable Node")
λ_1, λ_2 gerçel, $\lambda_1 > 0, \lambda_2 > 0$	Kararsız Düğüm ("unstable Node")
λ_1, λ_2 gerçel, $\lambda_1 . \lambda_2 < 0$	Eyer Noktası ("Saddle Point"
λ_1, λ_2 kompleks eşlenik, Re $\{\lambda_1\} > 0$	Karasız Odak ("Unstable Focus")
λ_1, λ_2 kompleks eşlenik, Re $\{\lambda_1\} < 0$	Kararlı Odak ("Stable Focus")
λ_1, λ_2 sanal	Merkez ("Center")

2.3.3.1 Dönüşümler

C^r $(r \geq 1)$ sınıfından bir dönüşüm (fark denklemi) göz önüne alınsın,

$$x \mapsto g(x), \quad x \in R^n \tag{2.29}$$

ve bu dönüşümün $x = x^*$ noktasında sabit bir noktası yani $x^* = g(x^*)$ olsun. Bu dönüşümün doğrusallaştırılmışı, $A \equiv Dg(x^*)$ olmak üzere,

$$y \mapsto Ay, \quad y \in R^n \tag{2.30}$$

şeklinde verilir. $y_0 \in R^n$ noktasının (2.30) altındaki orbiti, (eğer dönüşüm $C^r, r \geq 1$ diffeomorfizmse) iki yönlü sonsuz dizi şeklinde,

$$\{...,A^{-n}y_0,...,A^{-1}y_0,y_0,Ay_0,...,A^n y_0,...\} \tag{2.31}$$

veya sonsuz dizi şeklinde (eğer dönüşüm $C^r, r \geq 1$ sınıfındansa; fakat evrilir değilse),

$$\{y_0,Ay_0,...,A^n y_0\} \tag{2.32}$$

şeklinde verilir. (2.31) ve (2.32) eşitliklerinden kolayca görülebileceği gibi, (2.30) sisteminin $y = 0$ sabit noktası; A'nın tüm özdeğerlerinin modülü birden küçükse asimptotik kararlıdır.

Eğer (2.30) eşitliğiyle verilen sistemin bazı özdeğerlerinin modülü birden küçük; gerikalanların modülü birden büyükse, (2.29) ile verilen dönüşümün hiperbolik sabit noktası, eyer noktası olarak adlandırılır. Eğer doğrusal dönüşümün tüm özdeğerlerinin modülü birden küçükse; dönüşümün hiperbolik sabit noktası, kararlı düğüm veya çukur olarak adlandırılır. Doğrusal dönüşümün özdeğerlerinin tümünün modülü birden büyükse; dönüşümün hiperbolik sabit noktası kararsız düğüm veya kaynak olarak adlandırılır. Eğer doğrusal dönüşümün özdeğerlerinin

modülü birse; dönüşümün hiperbolik olmayan sabit noktası, merkez olarak adlandırılır.

2.4 Periyodik Çözümler

(2.33) ile verilen vektör alanı ve (2.34) ile verilen dönüşüm göz önüne alınsın,

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}),\ \mathbf{x} \in \mathbf{R}^n, \tag{2.33}$$

$$\mathbf{x} \mapsto \mathbf{g}(\mathbf{x}),\ \mathbf{x} \in \mathbf{R}^n. \tag{2.34}$$

Tanım 2.16 (Vektör Alanı için Periyodik Çözüm) Bütün $\mathbf{t} \in \mathbf{R}$ değerleri için $\mathbf{x}(\mathbf{t}, \mathbf{t}_0) = \mathbf{x}(\mathbf{t}+\mathbf{T}, \mathbf{x}_0)$ eşitliğini sağlayan bir $\mathbf{T} > 0$ sayısı varsa; (2.33) denkleminin x_0'dan geçen çözümü T periyoduyla periyodiktir.

Tanım 2.17 (Dönüşüm için Periyodik Çözüm) Eğer $\mathbf{x}_0 \in \mathbf{R}^n$ noktasından geçen orbit, $\mathbf{k} > 0$ olmak üzere, $\mathbf{g}^k(\mathbf{x}_0) = \mathbf{x}_0$ koşulunu sağlıyorsa; bu orbitin, periyodu k olan periyodik bir orbit olduğu söylenir.

Teorem 2.7, bir bölgenin hangi koşullar altında periyodik bir çözüm içermeyeceğini ifade eder. Başka bir deyişle, periyodik bir çözümün varolmamasına dair yeterli bir koşul verir.

f ve g fonksiyonları, en azından C^1 sınıfından olmak üzere aşağıdaki vektör alanı düşünülsün,

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x},\mathbf{y}) \\ \dot{\mathbf{y}} &= \mathbf{g}(\mathbf{x},\mathbf{y}). \end{aligned} \qquad (\mathbf{x},\mathbf{y}) \in \mathbf{R}^2 \tag{2.35}$$

Teorem 2.7 (Bendixson Kriteri) $\dfrac{\partial \mathbf{f}}{\partial \mathbf{x}} + \dfrac{\partial \mathbf{g}}{\partial \mathbf{y}}$ ifadesinin özdeş olarak sıfır olmadığı ve işaret değiştirmediği basit bağlantılı bir $\mathbf{D} \subset \mathbf{R}^2$ bölgesi varsa; (2.35), D içinde kapalı orbit içermez.

Tanıt için [9]'a bakınız.

Bağlantılı bölge, tek bir parça içinde olan bir küme olarak düşünülebilir. Yani, küme içindeki her iki nokta tamamen küme içinden geçen bir eğri ile birbirine bağlanır. Eğer herhangibir küme bağlantılı ve bu kümenin sınırı da bağlantılı ise; bu küme, basit bağlantılıdır.

2.5 Poincaré - Bendixson Teoremi

Poincaré - Bendixson teoremi, düzlemde limit çevrimin saptanabilmesi için bir kriter verir. Bu teorem, düzlemsel dinamik sistemleri anlamak için yararlanılan temel bir araç olmasına rağmen, daha yüksek boyutlu diferansiyel denklemler için bir genellemeye sahip değildir. Bu altbölüme, Poincaré - Bendixson teoreminin anlaşılması için gerekli görülen yardımcı tanımların verilmesi ile başlanacaktır.

2.5.1 Limit Kümeler

$\mathbf{f}:\mathbf{W}\to\mathbf{R}^n$, $\mathbf{W}\subset\mathbf{R}^n$ açık kümesi üzerinde $\mathbf{C}^1$ sınıfından bir fonksiyon olmak üzere (2.36) sistemi ele alınsın,

$$\dot{\mathbf{x}}=\mathbf{f}(\mathbf{x}). \tag{2.36}$$

Tanım 2.18 (ω - Limit Noktası) $\lim_{n\to\infty}\phi_{t_n}(\mathbf{x})=\mathbf{y}$ olmasını sağlayan bir $t_n\to\infty$ dizisi varsa; $\mathbf{y}\in\mathbf{W}$, $\mathbf{x}\in\mathbf{W}$'nın bir ω - limit noktasıdır.

Tanım 2.19 (ω - Limit Kümesi) ω - limit nokta olan tüm y değerlerinin oluşturduğu küme, ω - limit kümesi olarak adlandırılır, $\mathbf{L}_\omega(\mathbf{y})$.

Tanım 2.20 (α - Limit Noktası) $\lim_{n\to\infty}\phi_{t_n}(\mathbf{x})=\mathbf{y}$ olmasını sağlayan bir $t_n\to-\infty$ dizisi varsa; $\mathbf{y}\in\mathbf{W}$, $\mathbf{x}\in\mathbf{W}$'nın bir α - limit noktasıdır.

Tanım 2.21 (α - Limit Kümesi) α - limit nokta olan tüm y değerlerinin oluşturduğu küme, α - limit küme olarak adlandırılır, $\mathbf{L}_\alpha(\mathbf{y})$.

Limit kümelere bazı örnekler verilebilir. Örneğin $\mathbf{x}^*$, asimptotik kararlı bir denge noktası ise, çekim bölgesindeki her bir noktanın ω - limit kümesidir. Her denge noktası, kendisinin ω - limit kümesi ve α - limit kümesidir. Kapalı bir orbit,

üzerindeki her noktanın ω - limit kümesi ve α - limit kümesidir. Poincaré - Bendixson teoremi, düzlemde denge içermeyen sıkı ("compact": kapalı ve sınırlı) bir limit kümenin, kapalı bir orbit olduğunu söyler.

2.5.2 Poincaré - Bendixson Teoremi

Teorem 2.8 (Poincaré - Bendixson Teoremi) Denge noktası içermeyen C^1 sınıfından düzlemsel dinamik sistemin boş olmayan sıkı ("compact": kapalı ve sınırlı) kümesi, kapalı bir orbittir.
Tanıt için [12]'ye bakınız.

2.5.3 Limit Çevrimler

Tanım 2.22 (Limit Çevrim) Limit çevrim kapalı bir orbittir. Öyleki bazı $x \notin \gamma$ için, $\gamma \subset L_{\omega}(x)$ veya $\gamma \subset L_{\alpha}(x)$ dir. İlk durumda γ, ω - limit çevrim; ikinci durumda ise α - limit çevrim olarak adlandırılır.

2.6 Poincaré Dönüşümleri

Poincaré dönüşümü ile, n boyutlu sürekli zamanlı sistemin incelenmesi, (n-1) ayrık zamanlı sistemin incelenmesiyle yerdeğiştirir. Bu yöntemin sağladığı avantajlar şu şekilde sıralanabilir.
1) En azından bir değişken elendiğinden, bu dönüşüm daha düşük boyutlu bir sistemin incelenmesine neden olur.
2) Periyodik çözümün varlığı, Poincaré fonksiyonunun sabit noktasının varlığına indirgenir.
Periyodik çözümlerin kararlılığı ile ilgili bir yorum getiren karakteristik çarpanlardan (karakteristik çarpanlar "Floquet " çarpanları olarak da bilinir) bahsedebilmek için, periyodik bir orbit civarındaki Poincaré dönüşümünden kısaca bahsedilecektir.

2.6.1 Periyodik Bir Orbit Civarındaki Poincaré dönüşümü

$\mathbf{f}:U \to \mathbf{R}^n$, $U \subset \mathbf{R}^n$ açık seti üzerinde C^r olmak üzere,

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \qquad \mathbf{x} \in \mathbf{R}^n \tag{2.37}$$

diferansiyel denklemi göz önüne alınsın ve $\phi(t,\cdot)$, bu denklemin ürettiği akışı göstersin. $\mathbf{x}_0 \in \mathbf{R}^n$, periyodik orbitin geçtiği herhangibir nokta olmak üzere; (2.37)'nin, periyodu T olan ve $\phi(t,\mathbf{x}_0)$ ile gösterilen periyodik bir çözümünün olduğu varsayılsın (yani $\phi(t+T,\mathbf{x}_0) = \phi(t,\mathbf{x}_0)$). Σ, x_0 noktasında vektör alanına çapraz ("transverse") (n-1) boyutlu bir yüzey olsun. "Çapraz" kelimesinin anlamı: n(x), Σ'nın x noktasında normal vektörü olmak üzere ve ., vektör çarpanını göstermek üzere; $\mathbf{f}(\mathbf{x}).\mathbf{n}(\mathbf{x}) \neq 0$ olmasıdır. $V \subset \Sigma$ olan bir açık küme bulunabilir; Öyleki V açık kümesinde başlayan çözümler, değeri T'ye yakın bir zaman süresinden sonra Σ'ya döner. Poincaré dönüşümü,

$$\begin{aligned} &\mathbf{P}:V \to \Sigma, \\ &\mathbf{x} \mapsto \phi(\tau(\mathbf{x}),\mathbf{x}) \end{aligned} \tag{2.38}$$

şeklinde tanımlanır. $\tau(\mathbf{x})$, x'in Σ'ya ilk dönüş zamanıdır. $\mathbf{P}(\mathbf{x}_0) = \mathbf{x}_0$ ve $\tau(\mathbf{x}_0) = T$ dir. P'nin sabit noktası, (2.37)'nin periyodik orbitine karşı gelir ve bu sabit noktanın periyodu, k dır.

Oldukça kapsamlı olan Poincaré dönüşümleri, bu çalışmanın ana konusunu teşkil etmediğinden; yukarıda verilenler bu konu için yeterli görüldü. Bu konu ile daha fazla bilgi edinmek için, [15], [16], [17], [13], [12] ve [14] yayınları incelenebilir.

2.6.2 Periyodik Çözümlerin Kararlılığı: Karakteristik Çarpanlar

$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},t)$'nin periyodik bir çözümüne ilişkin Poincaré fonksiyonu, P(·) olsun. $\mathbf{x} \mapsto \mathbf{P}(\mathbf{x})$ dönüşümünün $\mathbf{x}^*$ sabit noktasının kararlılığı, P(·)'nin $\mathbf{x}^*$'da

doğrusal eşdeğerine bakılarak incelenebilir;

$$\delta \mathbf{x}_{k+1} = \mathbf{DP}(\mathbf{x}^*).\delta \mathbf{x}_k = \mathbf{D}_P.\delta \mathbf{x}_k. \tag{2.39}$$

"

$\mathbf{D}_P$'nin özdeğerleri $\mathbf{m}_i$, periyodik çözümün karakteristik çarpanları olarak adlandırılır. Denge noktasın özdeğerlerine benzer şekilde, karakteristik çarpanların kompleks düzlemdeki yeri, sabit noktanın kararlılığını belirler.

Teorem 2.9 (Periyodik Çözümlerin Kararlılığı) Eğer $|\mathbf{m}_i| < 1$ ve Σ, periyodik çözüme çapraz ise; periyodik çözüm asimptotik kararlıdır.

2.7 Sanki Periyodik Çözümler

Tanım 2.23 (Sanki Periyodik Çözüm) Sanki periyodik bir çözüm, periyodik fonksiyonların sayılabilir toplamı olarak ifade edilmelidir,

$$\mathbf{x}(t) = \sum_i \mathbf{h}_i(t). \tag{2.40}$$

($h_i(t)$, T_i periyoduna sahiptir) ve aşağıda verilen iki özelliği sağlayan sonlu sayıda taban frekansından oluşan bir küme olmalıdır, $\{\hat{\mathbf{f}}_1,...,\hat{\mathbf{f}}_p\}$. **1)** Bu küme lineer bağımsızdır. Yani $\mathbf{k}_1\hat{\mathbf{f}}_1+...+\mathbf{k}_p\hat{\mathbf{f}}_p = 0$ ifadesini sağlayan sıfırdan farklı bir tamsayılar kümesi yoktur. **2)** $\{\mathbf{k}_1,...,\mathbf{k}_p\}$ tamsayılar kümesi olmak üzere, f_i, her i için $\mathbf{f}_i = |\mathbf{k}_1\hat{\mathbf{f}}_1+...+\mathbf{k}_p\hat{\mathbf{f}}_p|$ şeklinde verilir, [18].

Yani sanki periyodik bir çözüm, her birinin frekansı, sonlu bir küme oluşturan taban frekanslarının çeşitli toplam ve farkları cinsinden yazılabilen periyodik dalgaların toplamı şeklindedir. Sanki periyodik bir çözüm, p taban frekanslarının sayısı olmak üzere, p - periyodik olarak adlandırılır.

Pratik bir görüş açısından kaos, yukarıdaki dinamik sistem davranışlarından hiçbiridir. Yani ne bir denge noktasıdır, ne periyodiktir ve ne de sanki periyodiktir.

Bölüm 3'te kavramsal kaos tanımı, bu tanımın anlaşılması için gerekli matematiksel araçların incelenmesinin ardından sunulacaktır.

BÖLÜM 3

KAOS VE GARİP ÇEKİCİLER

Bu bölüm, kaotiklik tanımının anlaşılması için gerekli değişmez küme, gezgin olmayan ("nonwandering") nokta, çekici küme, çekici kümenin çekim bölgesi ve tuzak bölgesi ön tanımlarının 3.1'de verilmesi ile başlayacaktır. Bu tanımların ardından kaos altbölüm başlığı altında, vektör alanları ve dönüşümler için topolojik geçişlilik, çekici ("attractor") ve ilk koşullara duyarlık tanımları verildikten sonra, kaotiklik tanımı vektör alanları ve dönüşümler için sunulacak ve kaotik sistemlere birkaç örnek verilecektir. Bu irdelemelerin ardından, Bölüm 3.3'te garip çekicinin ("strange attractor") tanımı üzerinde durulacaktır.

3.1 Ön Tanımlar

Tanım 3.1 (Değişmez Küme) $S, S \subset \mathbf{R}^n$ ifadesini sağlayan bir küme olsun.

a) (Sürekli Zamanda) Eğer herhangi bir $\mathbf{x}_0 \in \mathbf{S}$ için, $\mathbf{x}(t,0,\mathbf{x}_0) \in \mathbf{S}$ ifadesi tüm $t \in \mathbf{R}$ değerleri için sağlanıyorsa; S'nin $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ vektör alanı altında değişmez olduğu söylenir.

b) (Ayrık Zamanda) Eğer herhangi bir $\mathbf{x}_0 \in \mathbf{S}$ için, $\mathbf{g}^n(\mathbf{x}_0) \in \mathbf{S}$ ifadesi tüm n değerleri için sağlanıyorsa; S'nin $\mathbf{x} \mapsto \mathbf{g}(\mathbf{x})$ dönüşümü altında değişmez olduğu söylenir.

Tanım 3.2 (Gezgin Olmayan Nokta) Eğer aşağıdaki koşul sağlanırsa, $\mathbf{x}_0$'a gezgin olmayan (nonwandering) nokta denir.

a) (Akış) $\mathbf{x}_0$'ın herhangi bir U komşuluğu için, $t \neq 0$ olmak üzere, bazı t değerleri vardır. Öyleki bu değerler, $\phi(t,U) \cap U \neq \varnothing$ koşulunu sağlar.

b) (Dönüşüm) $\mathbf{x}_0$'ın herhangi bir U komşuluğu için, $n \neq 0$ olmak üzere, bazı n değerleri vardır. Öyleki bu değerler, $\mathbf{g}^n(U) \cap U \neq \varnothing$ koşulunu sağlar.

Tanım 3.3 (Çekici Küme) A'nın aşağıdaki koşulu sağlayan bazı U komşulukları varsa; kapalı değişmez küme $A \subset R^n$, çekici bir kümedir.

a) (Akış) $\forall x \in U,\ \forall t \geq 0,\ \phi(t,x) \in U$ ve $\phi(t,x) \underset{t\uparrow\infty}{\rightarrow} A$ (3.1)

b) (Dönüşüm) $\forall x \in U,\ \forall n \geq 0,\ g^n(x) \in U$ ve $g^n(x) \underset{n\uparrow\infty}{\rightarrow} A$. (3.2)

Tanım 3.4, faz uzayındaki hangi noktaların, çekici kümeye asimptotik olarak yaklaşdığını ifade eder.

Tanım 3.4 (Çekim Bölgesi) A'nın çekim bölgesi aşağıdaki şekilde verilir,

a) (Akış) $\bigcup_{t\leq 0} \phi(t,U)$ (3.3)

b) (Dönüşüm) $\bigcup_{n\leq 0} g^n(U)$. (3.4)

İfadelerde görülen U, Tanım 3.3'te tanımlandı.

Tanım 3.5 (Tuzak Bölgesi) Kapalı, bağlantılı M kümesi, $\forall t \geq 0$ için $\phi(t,M) \subset M$ koşulunu sağlıyorsa, tuzak bölgesi olarak adlandırılır. $\forall t \geq 0$ için $\phi(t,M) \subset M$ koşulunun sağlanmasıM'nin sınırındaki vektör alanın (∂M),M'nin iç bölgesine doğru yönelmesine eşdeğerdir. Tuzak bölgesi, çekim bölgesi tarafından içerilmelidir. Bu tanım kullanılarak A kümesi, $\bigcap_{t>0} \phi(t,M) = A$ şeklinde verilir. Tuzak bölgesi tarafından yakalanabilen orbitleri oluşturan noktalar kümesi, çekim bölgesini oluşturur.

3.2 Kaos

Tanım 3.6 (Topolojik Geçişlilik) Herhangi iki $U,V \subset A$ açık setleri için, aşağıdaki koşullar sağlanıyorsa; kapalı değişmez A kümesinin topolojik geçişli olduğu söylenir,

a) (Akış) $\exists t \in R \ni \phi(t,U) \cap V \neq \varnothing$ (3.5)

b) (Dönüşüm) $\exists n \in Z \ni g^n(U) \cap V \neq \varnothing$. (3.6)

Tanım 3.7 (Çekici) Bir çekici, topolojiksel geçişli olan çekici bir kümedir.

R^n üzerinde C^r $(r \geq 1)$ sınıfından otonom vektör alanlar ve dönüşümler düşünülsün,

Vektör Alanı $\dot{x} = f(x)$, (3.7)

Dönüşüm $x \mapsto g(x)$. (3.8)

(3.7) sisteminin ürettiği akış, $\phi(t,x)$ ile gösterilsin ve bu akışın tüm $t > 0$ değerleri için varolduğu varsayılsın. Yine $\Lambda \subset R^n$, $\phi(t,x)$ altında değişmez kompakt bir küme olsun. Yani tüm $t \in R$ değerleri için, $\phi(t,\Lambda)$ ifadesi sağlansın. Ve yine dönüşümler içinde $\Lambda \subset R^n$, $g(x)$ altında değişmez kompakt bir küme olsun. Yani tüm $n \in Z$ değerleri için ($g(x)$ tersinir değilse, $n \geq 0$ alınmalıdır), $g^n(\Lambda) \subset \Lambda$ ifadesi sağlansın.

Tanım 3.8 (Vektör Alanları için İlk Koşullara Duyarlık) Herhangibir $x \subset \Lambda$ değeri ve x'in herhangi bir U komşuluğu için, $|\phi(t,x) - \phi(t,y)| > \varepsilon$ koşulunu gerçekleyen $y \in U$ ve $t > 0$ sayılarının bulunabileceği bir $\varepsilon > 0$ değeri varsa, $\phi(t,x)$ akışının Λ üzerinde ilk koşullara duyarlı olduğu söylenir.

Tanım 3.9 (Dönüşümler için İlk Koşullara Duyarlık) Herhangibir $x \subset \Lambda$ değeri ve x'in herhangibir U komşuluğu için, $|g^n(x) - g^n(y)| > \varepsilon$ koşulunu gerçekleyen $y \in U$ ve $n > 0$ sayılarının bulunabileceği bir $\varepsilon > 0$ değeri varsa, $g(x)$'in Λ üzerinde ilk koşullara duyarlı olduğu söylenir.

Tanım 3.10 (Vektör Alanları için Kaotiklik) Λ, aşağıdaki koşullar sağlandığı takdirde kaotiktir,

1) $\phi(t,x)$, Λ üzerinde ilk koşullara duyarlıdır,

2) $\phi(t,x)$, Λ üzerinde topolojik geçişlidir.

Bazı yazarlar, örneğin [19], bu tanıma 3. şartıda koşmuştur,

3) $\phi(t,x)$'in periyodik orbitleri Λ içinde yoğundur.

Tanım 3.11 (Dönüşümler için Kaotiklik) Λ, aşağıdaki koşullar sağlandığı takdirde kaotiktir,

1) $g(x)$, Λ üzerinde ilk koşullara duyarlıdır,

2) $\mathbf{g(x)}$, Λ üzerinde topolojik geçişlidir.

Bazı yazarlar, örneğin [19], bu tanıma 3. şartıda koşmuştur.

3) $\mathbf{g(x)}$'in periyodik orbitleri Λ içinde yoğundur.

[14], her iki tanım içinde 3. maddeyi şart koşmamıştır; buna rağmen bu maddenin önemini ve kaosla olan ilişkisini açıklamıştır.

(3.10) ve (3.11) tanımlarında verilen özelliklerin hepsini değil; fakat birkaçını sağlayan bir örnek şu şekilde verilebilir.

Örnek 1) $\mathbf{a} > 1$ olmak üzere, $\mathbf{R^1}$ üzerinde (3.9) ile verilen vektör alanı göz önüne alınsın,

$$\dot{\mathbf{x}} = \mathbf{ax}, \quad \mathbf{x} \in \mathbf{R}^1. \tag{3.9}$$

(3.9) tarafından üretilen akış,

$$\phi(\mathbf{t},\mathbf{x}) = \mathbf{e}^{\mathbf{at}}\mathbf{x} \tag{3.10}$$

şeklindedir. (3.10) denkleminden aşağıdaki yargılara varılır.

1) $\phi(\mathbf{t},\mathbf{x})$, peiryodik orbite sahip değildir.

2) $\phi(\mathbf{t},\mathbf{x})$, sıkı olmayan $(0,\infty)$ ve $(-\infty,0)$ kümelerinde topolojik geçişlidir.

3) $\phi(\mathbf{t},\mathbf{x})$, $\mathbf{R^1}$ üzerinde ilk koşullara aşırı duyarlıdır, [14]. Çünkü $\mathbf{x}_0 \neq \mathbf{x}_1$ olmak üzere herhangibir $\mathbf{x}_0, \mathbf{x}_1 \in \mathbf{R}^1$ için,

$$\left|\phi(\mathbf{t},\mathbf{x}_0) - \phi(\mathbf{t},\mathbf{x}_1)\right| = \mathbf{e}^{\mathbf{at}}\left|\mathbf{x}_0 - \mathbf{x}_1\right| \tag{3.11}$$

sağlanır. Böylece herhangi iki nokta arasındaki fark, zamanda üstel olarak artar.

Verilen örnek, kaotiklik için Tanım (3.10) ve Tanım (3.11)'in tamamen sağlanması gerektiğinin önemini vurgular.

3.2.1 Kaotik Sistemlere Örnekler

Kaos, pek çok farklı araştırma alanlarında gözlemlenmiştir. Kaos sergileyen dinamik bir sisteme örnek verilmek istenirse, belli bir tür popülasyonunun uzun terim davranışını inceleyen popülasyon biyolojisi göz önüne alınabilir. Biyologlar, popülasyon değişimlerini sürekli veya ayrık zamanda ifade edebilmek için matematiksel modeller oluştururlar. Popülasyonun bazı limit değerleri olduğunu farz eden böyle bir model [19]'da

$$\frac{dP}{dt} = rP(L-P) \tag{3.12}$$

şeklinde verilir. P, popülasyonun büyüklüğünü; L, P'nin alabileceği maksimum değeri ve r, değeri çevre şartlarına bağlı olarak değişen bir parametreyi gösterir. Bu diferansiyel denklemin çözümü, (3.13)'te verilmiştir.

$$P(t) = \frac{LP_0e^{Lrt}}{L-P_0+P_0e^{Lrt}} \tag{3.13}$$

P_0, başlangıçdaki popülasyon değerini gösterir. Bu model periyodik bir davranış göstermediği gibi başka herhangibir düzensiz değişim de göstermez. Buna rağmen bu modele paralel

$$P(n+1) = rP(n)(L-P(n)) \tag{3.14}$$

fark denklemi , sürekli karşılığı kadar basit bir dinamiğe sahip değildir. Hakikaten bu fark denklemin dinamiği henüz tam olarak anlaşılamamıştır. Bundan başka bu sistem, çok boyutlu dinamik sistemlerin sergilediği patolojilere sahiptir. Bu nedenle $P(n+1) = rP(n)(L-P(n))$ fark denklemi, en temel doğrusal olmayan dinamik sistem olarak düşünülür. (3.14) eşitliği $x(n) = P(n)/L$ kullanılarak normalize edilebilir,

$$x(n+1) = rx(n)(1-x(n)). \tag{3.15}$$

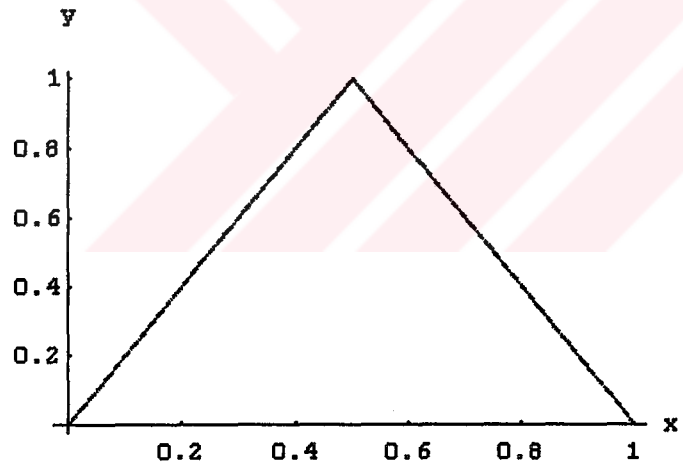
Bu çalışmada lojistik dönüşüm olarak bilinen (3.15) eşitliğinin dinamik özellikleri üzerinde geniş çapta durulmuştur.

Lojistik dönüşümden başka kaos sergileyen dönüşümler de vardır.

3.2.1.1 Çadır ("Tent") Dönüşümü

$$x(n+1) = \begin{cases} 2rx(n), & 0 \le x(n) \le 1/2 \\ 2r(1-x(n)), & 1/2 \le x(n) \le 1 \end{cases} \tag{3.16}$$

şeklinde tanımlanan çadır dönüşümünün grafiği, $r = 1$ değeri için Şekil 3.1'de verilmiştir.

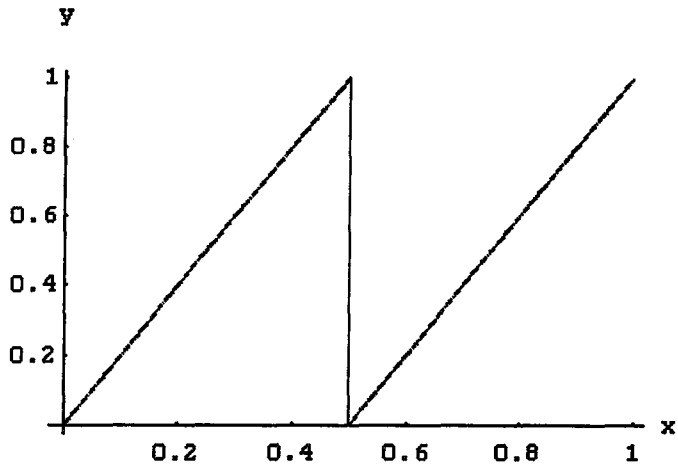


Şekil 3.1 Çadır dönüşümü.

3.2.1.2 Testere Dişi ("Sawtooth") Dönüşümü

$$x(n+1) = \begin{cases} rx(n), & 0 \le x(n) \le 1/r \\ \frac{r}{r-1}(x(n) - \frac{1}{r}), & 1/r \le x(n) \le 1 \end{cases} \tag{3.17}$$

şeklinde verilen testere dişi dönüşümünün grafiği $r = 2$ değeri için Şekil 3.2'de verilmiştir.



Şekil 3.2 Testere dişi dönüşümü.

3.2.1.3 Sinüs Dönüşümü

Belli parametre değerleri için kaos sergileyen sinüs dönüşümü,

$$\mathbf{x(n+1) = a[x(n)]^2 \sin(\pi x(n))} \tag{3.18}$$

şeklinde verilir.

3.2.1.4 Öteleme Dönüşümü

Tanım 3.12 (Dizi Uzayı) $\Sigma_2 = \{s = (s_0 s_2 s_3 ...) | s_j = 0$ **veya** $1\}$ şeklinde tanımlanan Σ_2, 0 ve 1 sembolleri üzerine dizi uzayı olarak adlandırılır. Σ_2'nin elemanları, (000...) veya (0101...)'e benzeyen sonsuz dizilerdir. Daha genel olarak Σ_n, 0 ve n-1 arasındaki tamsayılardan oluşan sonsuz dizilerden meydana gelir.

Tanım 3.13 (Öteleme Dönüşümü) $\sigma: \Sigma_2 \to \Sigma_2$ öteleme dönüşümü, $\sigma(s_0 s_1 s_2 ...) = (s_1 s_2 s_3 ...)$ şeklinde tanımlanır ve süreklidir.

s_0, 0 veya 1 olabilir. Bundan dolayı σ, Σ_2'nin ikiye - bir dönüşümüdür. Öteleme dönüşümü şu özelliklere sahiptir:

i) Mümkün olan tüm periyotlara sahip sayılabilir sonsuzlukta periyodik orbitler içerir;

ii) Sayılamayan sonsuzlukta periyodik olmayan orbitler içerir ve

iii) Yoğun orbit içerir. Yani bir sembol dizisinin bazı tekrarlamaları, Σ_2'de verilen herhangibir sembol dizisine gelişigüzel yakındır.

Yani Σ_2, σ için kaotiktir.

Testere dişi dönüşümü, ikili açılım kullanılarak yorumlandığında, öteleme dönüşümü olarak adlandırılır. Öteleme dönüşümünün Tanım 3.11'de verilen özellikleri sağladığı, Bölüm 4.6'da gösterilecektir. Yerdeğiştirme özelliği yardımıyla, testere dişi dönüşümünün tekrarlamalarına indirgenebilen çadır dönüşümü için; kaosun özellikleri, öteleme dönüşümünün ikili açılım kullanılarak ifade edildiğinde testere dişi dönüşümüne olan özdeşliğinden dolayı, öteleme donüşümü yardımıyla ortaya çıkarılacaktır. Böylece lojistik dönüşümün, çadır dönüşümüne özdeşliğinin Bölüm 4.6'da gösterilmesiyle; kaosun özelliklerinin lojistik dönüşüm için de sağlandığı gerçeği ortaya çıkarılmış olacaktır.

3.2 Garip Çekiciler

Garip çekicilerin varlığından söz edebilmek için yitirgen (dissipative) dinamik sistemler göz önüne alınmalıdır. Bu sistemlerin en belirgin özelliği enerji kaybetmeleridir. Garip çekiciler, oldukca kompleks ve kaosun tüm işaretlerini sergileyen yitirgen sistemlerin son durumunu karakterize eden desenlerdir.

Tanım 3.14 (Garip Çekici) $\mathbf{A} \in \mathbf{R}^n$'nin bir çekici olduğu farz edilsin. Bu durumda A kaotikse, garip çekici olarak çağrılır.

Buna benzer formal tanımlar, [20] ve[21]'de bulunabilir. Salt matematiksel olarak sunulan bu ifadelerin anlaşılabilirliğini arttırmak hedefiyle garip çekicilerin tanımı, matematiksel araçları sözle ifade eden cümlelerle aşağıdaki gibi tekrar verilmiştir.

Eğer aşağıdaki özellikleri sağlayan bir **M** kümesi varsa; düzlemin sınırlı bir **A** altkümesi, **g** dönüşümü için kaotiktir ve garip çekicidir.

i) Çekici: **M**, **A**'nın komşuluğundadır. Yani **A**'nın her **(x,y)** noktası için, **M**'de içerilen **(x,y)** merkezli küçük bir disk vardır. Bu, **A**'nın **M** içinde olmasını gerektirir. **M** bir tuzak bölgesidir. Öyleki **M**'de başlayan her orbit, tüm iterasyonlar için yine **M**'dedir. Bundan başka orbit **A**'ya yakın kalır ve istenildiği kadar **A**'ya yaklaştırılabilir. Yani **A**, bir çekicidir.

ii) Duyarlık: M'de başlayan orbitler ilk koşullara duyarlık gösterir. Bu özellik, A'yı kaotik bir çekici yapar.

iii) Fraktal: Çekici, parçalı bir yapıya sahip olduğundan; garip çekici olarak adlandırılır.

iv) Topolojik Geçişlilik: A, iki farklı çekiciye parçalanamaz. M'de öyle başlangıç koşulları vardırki; bunların ürettiği orbitler, A'nın herhangibir başka noktasına gelişigüzel şekilde yakındır.

Eğer verilen bir sistemin garip bir çekiciye sahip olduğu tanıtlanmak isteniyorsa, aşağıdaki adımlar takip edilebilir.

Adım 1) Faz uzayında bir M tuzak bölgesi bulunur.

Adım 2) M'nin kaotik değişmez bir Λ kümesi içerdiği gösterilir.

Adım 3) Bu iki adım doğrulandığı takdirde, $\bigcap_{t>0}\phi(t,M)=A$ çekici bir kümedir ve daha fazlası $\Lambda \subset A$ ifadeside sağlanır. Böylece A, ilk koşullara duyarlığa sebebiyet veren bir mekanizma içerir. Yani A'nın garip bir çekici olduğu sonucuna ulaşmak için yalnızca aşağıda verilen iki özelliğin doğruluğunun gösterilmesi yeterlidir.

1) Λ üzerindeki ilk koşullara duyarlık, A'ya genişler.

2) A, topolojik geçişlidir.

Garip çekicilerle ilgili, aşağıda verilen alanlarda oldukça detaylı çalışmalar sonunda ortaya çıkarılmış özellikler vardır.

1) Tek Boyutlu Tersinir Olmayan Dönüşümler Bu dönüşümlere, μ bir parametre olmak üzere,

$$x \mapsto \mu x(1-x) \tag{3.19}$$

veya

$$x \mapsto x^2 - \mu \tag{3.20}$$

örnek olarak verilebilir. Okuyucu, [22], [23], [24] ve [25], ve yayınlarına başvurabilir.

2) İki Boyutlu Dönüşümlerin Hiperbolik Çekicileri [26], [27] ve [29], Tanım 3.11'i sağlayan hiperbolik çekici kümelerle ilgili yapay örnekler oluşturmuşlardır.

3) Lorenz Benzeri Sistemler Son birkaç yılda Lorenz denklemlerinin garip bir çekiciye sahip olduğunu tanıtlamaya dair büyük ilerlemeler sağlanmıştır. Okuyucu,[29], [30], [31] ve [32] eserlerine başvurabilir.

4) Hénon Dönüşümü Bu dönüşüm, μ ve ε parametreler olmak üzere,

$$\begin{aligned} \mathbf{x} &\mapsto \mathbf{y}, \\ \mathbf{y} &\mapsto -\varepsilon \mathbf{x} + \mu - \mathbf{y}^2, \end{aligned} \tag{3.21}$$

şeklinde verilir. Küçük ε değerleri için bu dönüşümün garip bir çekiciye sahip olduğu [33]'te tanıtlanmıştır.

Garip çekici problemi, hala tam anlamıyla çözümlenememiştir. Problemin çözümlenebilmesi için, çok boyutlu dinamik sistemlerle (vektör alanları için üçten büyük boyutlu; dönüşümler içinse ikiden büyük boyutlu sistemlerle) ilgili detaylı çalışmalara ihtiyaç vardır. Çok boyutlu sistemlerde kaotik davranış sergileyen çeşitli örnekler,[34]'te bulunabilir.

Garip çekici tanmı ile sınırlanan bu altbölümün, Chua ve Lorenz sistemlerinin Bölüm 5'te irdelenmesiyle daha iyi kavranacağına inanıyoruz.

BÖLÜM 4

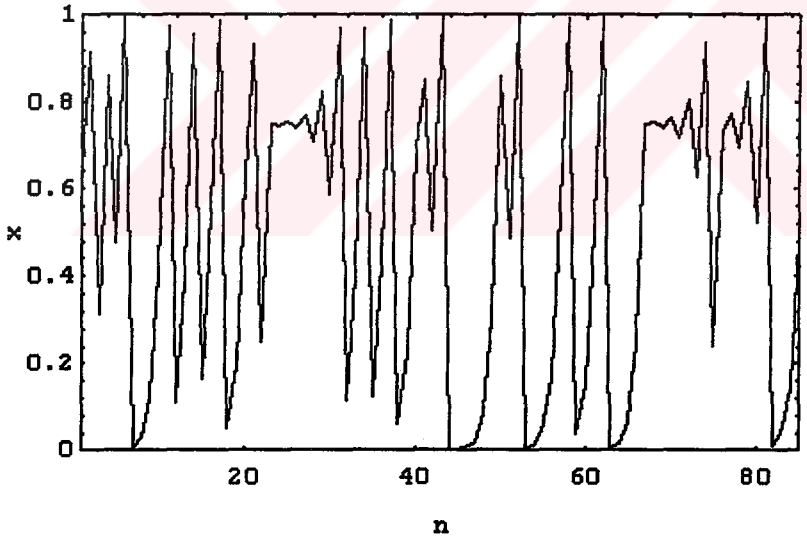
LOJİSTİK DÖNÜŞÜM İÇİN DUYARLIK, TOPOLOJİK GEÇİŞLİLİK VE PERİYODİK ORBİTLER

Kaotik davranışın önemli bir özelliği olan duyarlığın, $x \mapsto 4x(1-x)$ lojistik dönüşümü için, nümerik olarak elde edilen grafikler yardımıyla açıklanmasıyla başlayacak olan Bölüm 4, yalnızca parametrelere duyarlığın, kaotik davranışa neden olamayacağının, $x \mapsto 1.5x$ doğrusal dönüşümü için gösterilmesiyle devam edecektir. Daha sonra hata gelişiminin, $x \mapsto 4x(1-x)$ lojistik dönüşümü için analiz edilmesiyle varılacak olan ve kaotik davranışın ölçülmesini olanaklı hale getiren Liapunov üstellerinden söz edilecektir. Bu incelemenin ardından, $x \mapsto 4x(1-x)$ dönüşümünün topolojik geçişli olduğu ve yoğun periyodik orbitlere sahip olduğu, nümerik olarak verilen tekniklerle sunulacaktır. Nümerik olarak bulunan periyodik orbitlerin aslında sınırlı prezisyondan kaynaklandığı, prezisyon arttırıldığında, düşük prezisyonda varolan periyodik orbitlerin kaybolduğu ve yenilerinin oluştuğu belirtildikten sonra, nümerik çalışmalarda nasıl bir mekanizmanın yapay periyodik orbitlere neden olduğu üzerinde durulacaktır. Nümerik olarak bulunan sonuçların açıklanmasının ardından, kaotikliğin üç özelliğini sağlayan, başka bir deyişle kaosu yaratan temel işlem olarak yorumlanan "yoğurma" olayından söz edilecektir. Yoğurma işleminin, çekme - katlama şeklinde yorumlanması, çadır dönüşümünün tanımının yapılmasına; bu işlemin, çekme - kesme ve yapıştırma şeklinde yorumlanması ise testere - dişi dönüşümünün tanımının yapılmasına neden olacaktır. Ardından, çadır dönüşümünün kaosun üç özelliğini sağladığı, kaosu yaratan temel mekanizmalardan biri olan ve testere - dişi dönüşümünün, kendine argüman olan sayının ikili açılımı düşünülerek yorumlanmasıyla tanımlanacak olan öteleme dönüşümü yardımıyla gösterilecektir. Bu incemelerin ardından, düzgün yoğurma işlemine karşı gelen çadır dönüşümü ile düzgün olmayan yoğurma işlemine karşı gelen lojistik dönüşümün eşdeğerliği, çadır

dönüşümünün, testere-dişi dönüşümü kullanılarak ikili sayılar için yeniden tanımlanmasıyla verilecektir.

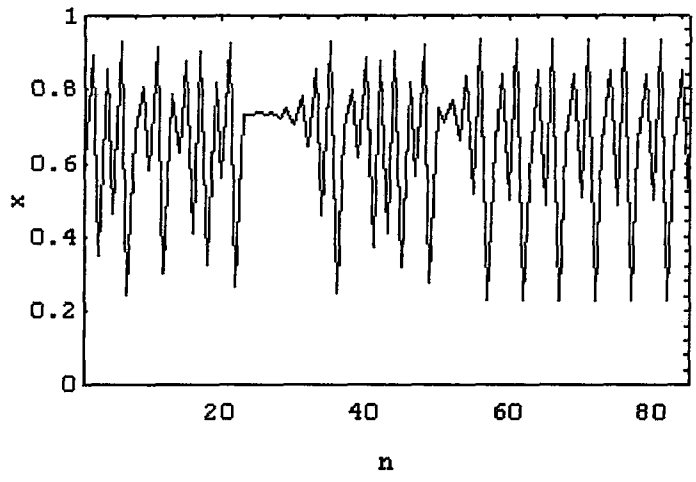
4.1 İlk Koşullara Duyarlık

Lojistik dönüşümün üç farklı şekli, $x \mapsto x^2 + c$, $p \mapsto p + rp(1-p)$ ve $x \mapsto ax(1-x)$ olan bu denklemler, $c = -2$, $r = 3$ ve $a = 4$ değerleri için kaotik bir davranış sergilerler. Şekil 4.1, $a = 4$ ve $x_0 = .2027$ değerleri için $x \mapsto ax(1-x)$ lojistik dönüşümünün davranışını gösterir. Şekilde yatay eksen, dönüşümün tekrarlanma sayısını; düşey eksen ise x değerini gösterir. Küçük bir aralık dışındaki genlik değerlerinin tahmininin oldukca zor olduğu görülmektedir. Grafiğe bakılarak emin olunulan tek şey, genliğin 0 ile 1 arasında kalmasıdır; $0 \le x \le 1$ ve $0 \le a \le 4$ olduğu müddetçe $x \mapsto ax(1-x)$ ifadesinin genliği 0 ile 1 arasında kalır.



Şekil 4.1. $x_0 = .2027$ ilk koşulu ve a = 4 parametre değeri için lojistik dönüşümün zaman serisi.

Aynı formül ile çizilen Şekil 4.2'de, a değeri 3.742718 olmak üzere, yine aynı $x_0 = .2027$ başlangıç koşulu ele alındı. Grafikten x değerinin, her 5. veya 10. tekrarlamada aynı genliği aldığı görülmektedir. Bilgisayarla yapılan nümerik hesaplarla periyodun gerçekte 20 olduğu bulundu. Bu farklılık, grafik çizilerken yapılan kaçınılmaz yuvarlatma hatalarından kaynaklanmaktadır. Burada gözlemlenilen, nümerik hesaplar ile teorik hesapların birbirinden farklı olduğudur.

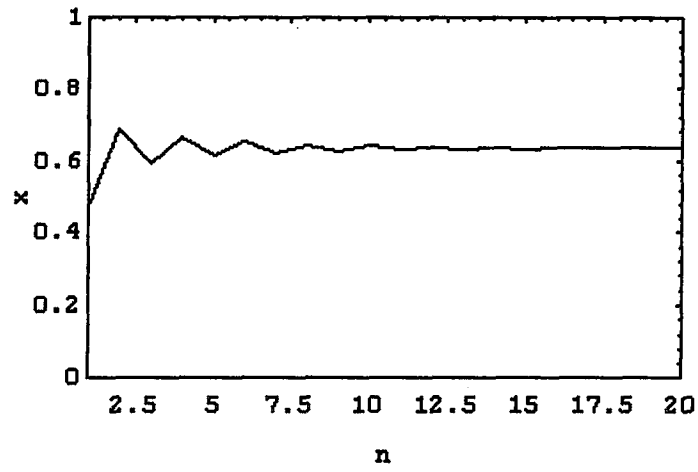


Şekil 4.2. x_0 =.2027 ilk koşulu ve a = 3.742718 parametre değeri için lojistik dönüşümün zaman serisi.

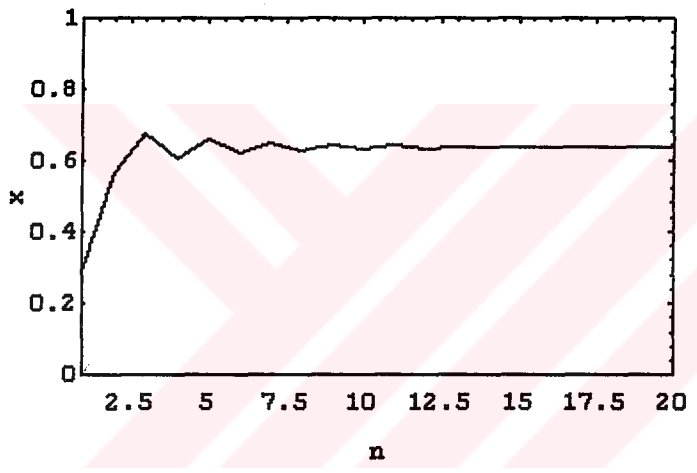
Şekil 4.3'te, a = 2.75 değeri ve $x_0 = 0.22$ ve $x_0 = 0.12$başlangıç koşulları için dömüşümün aynı son duruma yakınsadığı görülmektedir. Bu, parametrelerine duyarlı olmayan, kararlı bir sisteme örnek olarak verilebilir.

Tek bir noktaya yakınsama olayı, grafiksel tekrarlamayla da görülebilir. Grafiksel tekrarlamanın nasıl yapıldığına dair bir açıklama yapmak yararlı olur. x_0 değerinin bir sonraki tekrarlamada aldığı x_1 değeri, x_0'dan parabole dik çıkılarak bulunur. Fonksiyonun, x_1 değerini nereye götürdüğünü bulabilmek için, x_1 değerinin x eksenine taşınması gereklidir. Bunu yapmanın en iyi yolu ise $y = x$ eksenini kullanmaktır. Yani x_0'ın bir sonraki tekrarlamada aldığı x_1 değeri, x_0'dan parabole dik çıkılarak bulunur. Bulunan bu x_1 değerinden $y = x$ eksenine gidilirse, x_1 değerinin x eksenine taşınması gerçekleştirilmiş olur. Daha sonra x_1 değerinden parabole dik çıkılarak x_2 değeri bulunur. İşleme bu şekilde devam edildiği takdirde limit küme elde edilir.

Şekil 4.4'te, yalnız $x_0 = 0.22$ ilk koşuluyla üretilen orbitin değil, aynı zamanda $x_0 - 0.015$ ve $x_0 + 0.015$ aralığında ürettilen tüm orbitlerin de tek bir noktaya yakınsadığı görülmektedir. Bu grafiklerde gözlemlenilen şey, belirli bir aralıktaki tüm ilk koşullarla tek bir değere yakınsandığı olayıdır. Bununla birlikte aralıktaki tüm değerlerin, aynı son duruma yakınsadığı gözlemlenebilir. Bu, kararlılığa bir örnektir. Böyle bir olay kaotik sistemlerde görülmez.



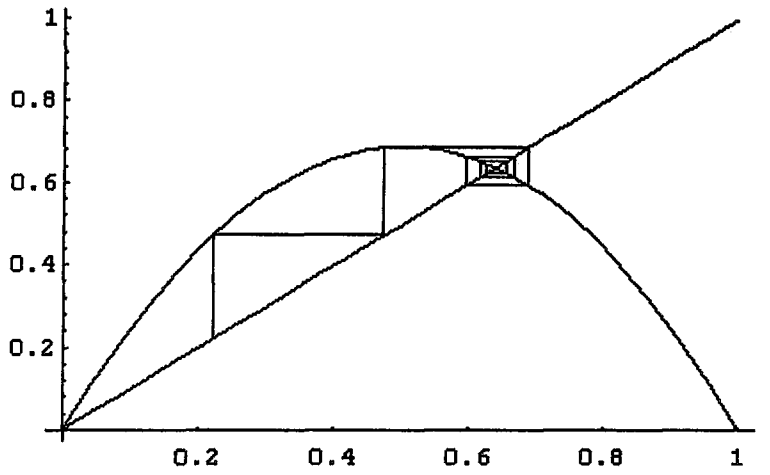
(a)



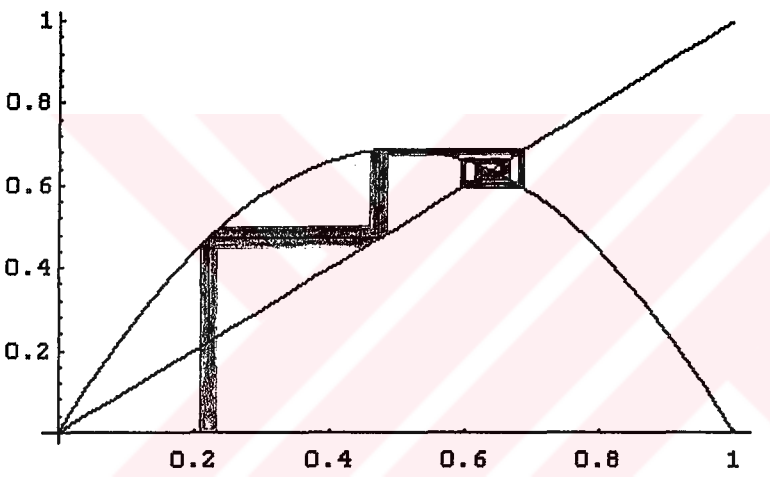
(b)

Şekil 4.3. a = 2.75 değeri için 0 ve 1 arasındaki tüm ilk koşullar aynı son duruma yakınsar. (a) $x_0 = 0.5$ ilk koşulunun 0.64 noktasına yakınsaması. (b) $x_0 = 0.3$ ilk koşulunun 0.64 noktasına yakınsaması.

Lojistik dönüşümde a parametresi, 3'ten büyük olacak şekilde değiştirildiğinde, sistemin davranışı oldukça farklılaşır. a parametresi 3'ten küçük olduğunda tek bir değere yakınsayan dönüşüm, a = 3 değerinden itibaren, hangisine varılacağını başlangıç koşullarının belirlediği iki farklı değere yakınsar. Sistemin kalıcı durum davranışı bu durumda iki nokta iken, a parrametresi bu noktadan sonra arttırıldığında, sistemin kalıcı durumu 4, 8, 16, 32,... farklı değerden oluşur. a parametresi 3.8'den büyük olduğunda periyot ikilenmesi kaybolur ve rasgele seçilen bir x_0 başlangıç değerinin lojistik dönüşüm altında ürettiği orbit rasgele görünüşlü olur. Yani sistemin bu noktadan sonra davranışı tamamen kaotiktir.



(a)

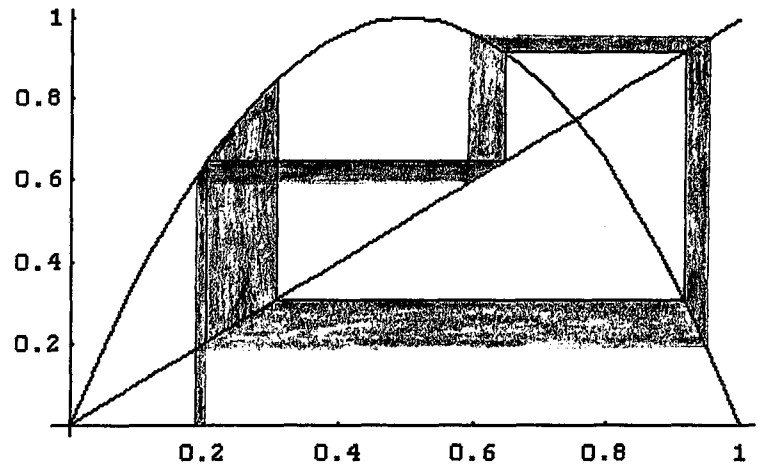


(b)

Şekil 4.4. (a) a = 2.75 parametre değeri ve x_0 = 0.22 ilk koşulu için dönüşümün kararlı tek bir noktaya yakınsaması. (b) a = 2.75 parametre değeri, x_0 - 0.015 ve x_0 + 0.015 değerleri için dönüşümün kararlı tek bir noktaya yakınsaması.

4.1.1 Küçük Hataların Duyarlık Özelliğinden Dolayı Güçlenmesi

Kaotik davranışın önemli bir özelliği olan duyarlık, a parametresinin 4 yapılmasıyla açıklanacaktır. Parametrelere karşı olan duyarlık, en küçük hataların bile büyümesine neden olur. Şekil 4.5'den görüldüğü gibi çok küçük bir aralık, dönüşümün daha birkaç tekrarı sonucunda gözardı edilemeyecek derecede büyümüştür. Tekrarlama sayısı arttırıldığında, [0,1] aralığının tümünün kapsandığı görülür.



Şekil 4.5. Çok küçük bir aralığın tekrarlamalar sonucunda oldukça geniş bir bölgeye yayılması.

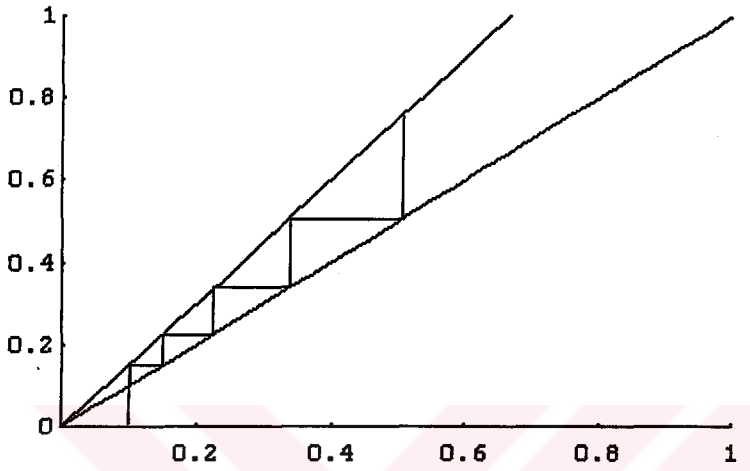
Yukarıda yazılanlar özetlenecek olursa, eğer parametrelere duyarlık söz konusuysa, gelişigüzel seçilen çok küçük bir aralık, dönüşümün tekrarlanması sonucunda önemli derecede genişler ve bu, ilk koşullara aşırı duyarlık olarak adlandırılır. Daha ileri giderek burada tartışılan $\mathbf{x} \mapsto 4\mathbf{x}(1-\mathbf{x})$ lojistik dönüşüm için, seçilen herhangibir aralığın, tekrarlamalar sonucunda tüm [0,1] aralığına genişleyeceği söylenebilir.

4.1.2 Doğrusal Dönüşüme Bir Örnek

Duyarlık, kaotik davranışın gözlenebilmesi için gerekli bir özelliktir. Buna rağmen yalnızca bu özellik otomatik olarak kaosa neden olmaz. Başka bir deyişle parametrelerine duyarlı fakat kaotik olmayan sistemler vardır. $\mathbf{x} \mapsto \mathbf{cx}$ basit ifadesi, $\mathbf{c} > 1$ olduğunda parametrelerine duyarlı, fakat kaotik olmayan bir sistemi tanımlar. Verilen bir x_0 ilk koşulu, dönüşümün n kez tekrarı sonunda $\mathbf{x(n)} = \mathbf{c^n x_0}$ değerine eşit olur. İşleme $\mathbf{u_0} = \mathbf{x_0} + \mathbf{E_0}$ ilk koşuluyla başlandığında, n tekrarlamadan sonra dönüşümün aldığı değer, $\mathbf{u(n)} = \mathbf{c^n}(\mathbf{x_0} + \mathbf{E_0})$ olur. $E_0 = \varepsilon$ olarak alınırsa, hatanın n tekrarlama sonunda ulaştığı değer,

$$E(n) = u(n) - x(n) = c^n(x_0 + \varepsilon) - c^n x_0 = c^n \varepsilon \qquad (4.1)$$

şeklindedir. Herhangibir ε sapması, dömüşümün her tekrarı sonunda $c > 1$ çarpanı ile büyür. Yani sistem duyarlıdır; fakat kaotik değildir. $x \mapsto 1.5x$ doğrusal sisteminin duyarlığı, Şekil 4.6'dan görülmektedir.



Şekil 4.6. Doğrusal sistemlerdeki duyarlığın $x \mapsto 1.5x$ doğrusal dönüşümü için incelenmesi.

4.1.3 Lorenz'in Yaklaşımı

Ünlü bir meterolojist olan Lorenz; kaotik sistemlerin duyarlığını, lojistik dönüşüm ile doğrusal dönüşüm arasındaki farkı kullanarak tanımlamıştır,[37]. Kaotik bir dönüşüm altında, verilen ilk koşuldan çok küçük bir sapma ile işleme başlandığında elde edilen işaret, gerçek işaret ile aynı genlikte olacaktır. Bu olayı başka şekilde ifade etmek gerekirse; dönüşümün birkaç tekrarından sonra, hatanın genliği, gerçek işaretinin genliği ile aynı derecede olacaktır.

$x(n+1) = 4x(n)(1-x(n))$ lojistik dönüşümü için, ilk koşul $x_0 = 0.202$ olarak alınsın ve sapma değeride $\varepsilon = 10^{-6}$ olarak seçilsin. Bu sapma değeri ile .202 ilk koşuluna çok yakın olan .202001 değeri elde edilir. Şekil 4.7'de $|u(n)-x(n)|$'e karşı gelen seriler görülmektedir. İlk 15 tekrarlama için bu fark, sıfırdan ayırt edilemeyecek kadar küçüktür. Fakat belli bir kritik genlikten sonra aradaki farkta bir artma olmakta ve bu fark, tekrarlama sayısı arttıkça, orjinal işaret ile aynı genliğe ulaşmaktadır. Bu, kaotik sistemlerde gözlemlenen tipik bir davranıştır.

Bu hata gelişimine zıt bir karakteristik gösteren $\mathbf{x(n+1)=cx(n)}$ doğrusal dönüşümünde hata, gerçek değerlerle uyumlu olarak artar ve bağıl hata,

$$\frac{E_n}{x_n}=\frac{c^n\varepsilon}{c^n x_0}=\frac{\varepsilon}{x_0}=\text{sabit} \tag{4.2}$$

şeklinde verilir.

4.1.4 Hata Gelişiminin Analizi

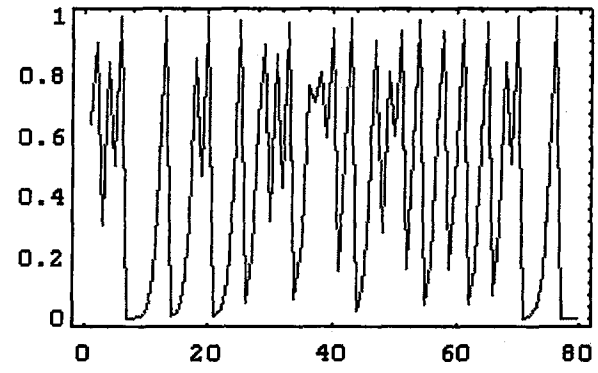
Yukarıda tanımlanan ilk koşullara aşırı duyarlık, tüm kaotik sistemlerin kesinlikle sahip olduğu bir özelliktir. Keşfedilmiş nicelikleri ölçebilmek için bir takım teknikler geliştirmek bilimin önemli bir konusudur. Hata yayılımının, burada tartışıldığı gibi olması durumunda bu ölçme nasıl yapılır?

Başlangıçta, basit bir doğrusal sistemi tanımlayan $\mathbf{x(n+1)=cx(n)}$ ifadesindeki hata yayılımı irdelensin. Hata, tekrarlama başına c faktörü ile artar,

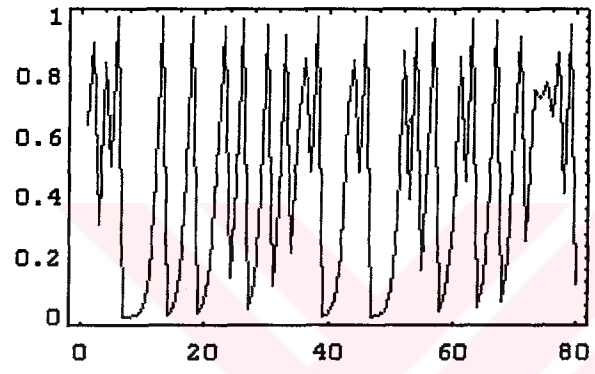
$$\left|\frac{E_n}{E_0}\right|=c^n. \tag{4.3}$$

Doğrusal sistem için bulunan bu basit kanunun, $\mathbf{x(n+1)=4x(n)(1-x(n))}$ ifadesi için geçerli olmadığı Şekil 4.7'de görülmektedir.

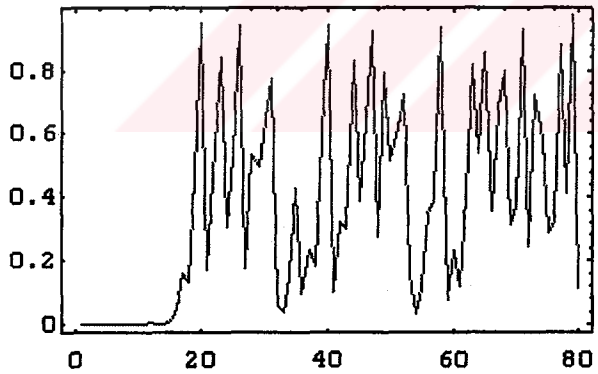
Hatanın 0.1 değerini aştığı ilk tekrarlama sayısı, çeşitli ilk koşullar ve farklı hata değerleri için belirlensin ve sonuçlar Tablo 4.1'de toplu olarak gösterilsin. Tablodaki ilk satırdan anlaşılması gereken şey: $x_0=0.202$ gerçek değeri ve $x_{nh}=0.203$ hatalı değeri ile işleme başlandığında, dönüşüm sonuçlarında aradaki farkın, 0.1 değerini geçtiği ilk tekrarlama sayısı, 9 ve bu tekrarlama sayısı sonucunda $\mathbf{E(n)}$ değerinin, $\mathbf{E(n)}=0.25622$ olduğudur.



(a)

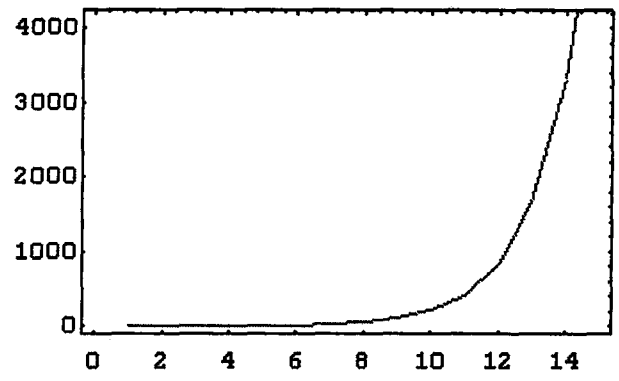


(b)

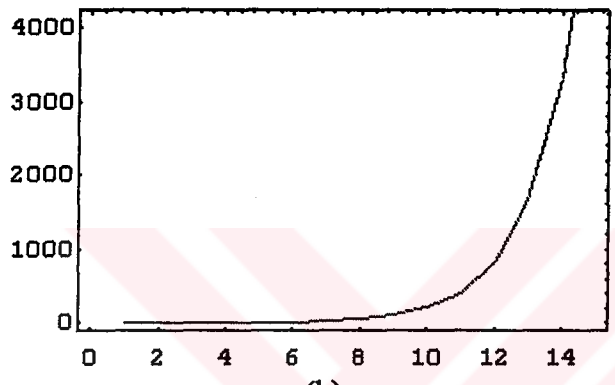


(c)

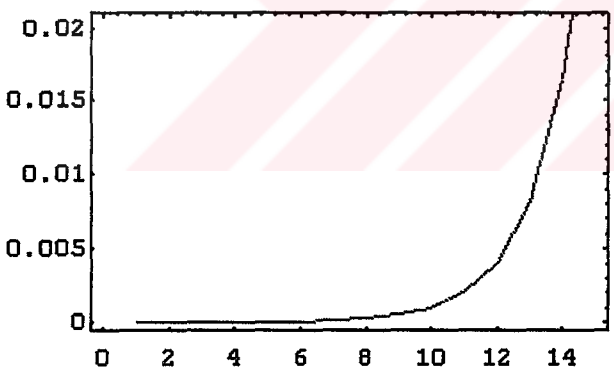
Şekil 4.7. (a) $x_0 = 0.6$ ilk koşulunun ürettiği orbit. (b) $y_0 = 0.600001$ ilk koşulunun ürettiği orbit. (c) $x_0 = 0.6$ ve $y_0 = 0.600001$ ilk koşul değerlerinin oluşturduğu orbitler arasındaki farkın mutlak değeri



(a)



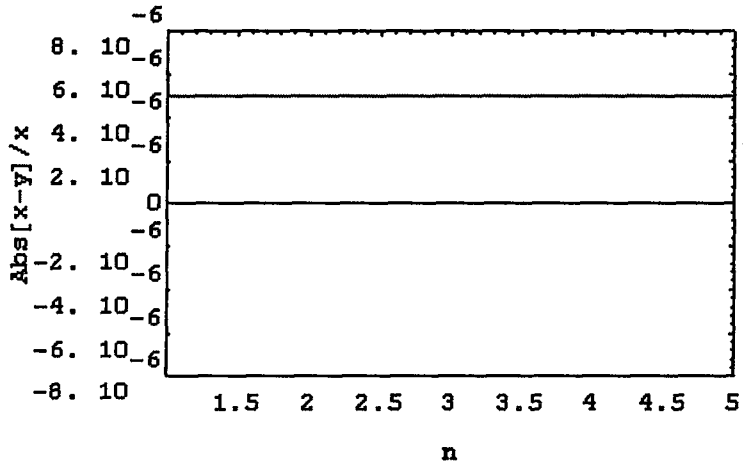
(b)



(c)

Şekil 4.8. (a) $x_0 = 0.6$ ilk koşulunun $x \mapsto 1.5\ x$ dönüşümü altında ürettiği orbit. (b) $y_0 = 0.600001$ ilk koşulunun $y \mapsto 1.5\ y$ dönüşümü altında ürettiği orbit. (c) $x_0 = 0.6$ ve $y_0 = 0.600001$ ilk koşul değerlerinin $x \mapsto 1.5\ x$ dönüşümü altında oluşturduğu orbitler arasındaki farkın mutlak değeri.

Bağıl hatanın tüm zamanlar için aynı kaldığı şekil 4.9'da görülmektedir.



Şekil 4.9. $x \mapsto 1.5x$ doğrusal dönüşümü için, $x_0 = 0.202$ ve $y_0 = 0.202001$ ilk koşulları ele alındığında (y(n) -x(n)) / x(n) değerinin grafiği.

Tablo 4.1. $x(n+1) = 4x(n)(1-x(n))$ lojistik dönüşümü için hata yayılımı.

x_0	Hata E_0	Adım n	Hata E(n)	Üstel
0.202	0.001000	9	0.25622	0.61623
0.202	0.000100	11	- 0.12355	0.64720
0.202	0.000010	15	0.25730	0.67703
0.202	0.000001	17	0.15866	0.70438
0.347	0.001000	7	-0.12331	0.68781
0.347	0.000100	11	-0.18555	0.68417
0.347	0.000010	15	0.31390	0.69028
0.347	0.000001	18	0.19490	0.67668
0.869	0.001000	8	-0.25072	0.69054
0.869	0.000100	10	0.14068	0.72491
0.869	0.000010	13	0.11428	0.71876
0.869	0.000001	18	0.32095	0.70439

Hatanın, tekrarlama başına ne hızla arttığına dair bir ölçü türetmek amacıyla, $\left|\frac{E(n)}{E_0}\right| = c^n$ ifadesine değişik bir açıdan bakılsın. Yukarıdaki deneyde **E(n)** ve E_0 değerleri sanki doğrusal sistemden geliyormuş gibi düşünülebilir. Bu, hata gelişimini nümerik olarak karakterize eden **c** faktörünü belirlemeye imkan sağlar. (4.1)

ifadesinin her iki tarafının doğal logaritmasının alınmasıyla, bu işlem için uygun bir formül bulunur,

$$\ln \mathbf{c} = \left| \frac{\mathbf{E(n)}}{\mathbf{E_0}} \right|. \tag{4.4}$$

Tablo 4.1'de "üstel" başlığı altında verilen ifadenin sağ tarafı, hata gelişimini gösteren **c** sabitinin logaritmasını verir. İlginç olan bu üstel değerlerin tümünün 0.7 civarında olmasıdır. Dolayısıyla $c \approx e^{0.7} \approx 2$ olur. Yapılan bu deneyin sonucu şu şekilde ifade edilebilir: $\mathbf{x(n+1) = 4x(n)(1-x(n))}$ lojistik dönüşümünde, ufak hatalar, her tekrarlamada yaklaşık olarak 2 ile çarpılır.

Bu ifade yeterince küçük hatalar için geçerlidir. Yani birim aralık içinde hataların artmadığı **x** değerleri vardır. Örneğin 0.5 noktası civarında parabol oldukça düzdür ve hatalar bastırılır. Diğer taraftan birim aralığın uç noktalarında hatalar 4 çarpanına varıncaya kadar artar.

4.1.5 Liapunov Üstelleri

Deneyle gözlemlenen yukardaki olay, Liapunov üstellerinin tanımlanmasına neden olur. Bu üsteller, x_0'ın içindeki çok küçük hataların ortalama büyümesini niteler. Yukarıda verilen metodun, x_0 ilk koşulundan bağımsız gibi görünmesi ilginçtir. Hesaplama, daha doğru olarak nasıl yapılabilir? Hatadaki ortalama büyümeyi hesaplamak için, Tablo 4.1'de verilen tekrarlama sayısından daha fazla sayıda tekrarlamalar düşünmek bir avantaj olabilir. Bu durumda daha ufak E_0 hatasıyla başlama zorunluluğu doğmaktadır. Çünkü hata, her tekrarlama başına 2 ile çarpılır. Bu, o kadar iyi bir yaklaşım değildir. Çünkü, makine ile ifade edilebilecek mümkün en küçük hatayla bile işleme başlansa, ancak biraz daha fazla tekrarlama yapılacaktır. Yani farklı bir metod düşünülmelidir. Gelişigüzel seçilmiş E_0 ilk hatasıyla işleme başlandığı varsayılsın. Bu durumda toplam hata kuvvetlendirme faktörü $\left|\mathbf{E(n)/E_0}\right|$ aşağıdaki gibi yazılır,

$$\left|\frac{\mathbf{E(n)}}{\mathbf{E_0}}\right| = \left|\frac{\mathbf{E(n)}}{\mathbf{E(n-1)}}\right|\left|\frac{\mathbf{E(n-1)}}{\mathbf{E(n-2)}}\right|\cdots\left|\frac{\mathbf{E(1)}}{\mathbf{E_0}}\right|. \tag{4.5}$$

$\left|\mathbf{E(n)/E_0}\right|$ değerini hesaplamak için, tüm bu tek ifadelerin çarpılması gerekir. Bu ise bilgisayarda taşma hatasına neden olur. Bu durumdan kurtulmak için, logaritmik toplamlar düşünülebilir,

$$\begin{aligned}\frac{1}{\mathbf{n}}\ln\left|\frac{\mathbf{E(n)}}{\mathbf{E_0}}\right| &= \frac{1}{\mathbf{n}}\ln\left|\frac{\mathbf{E(n)}}{\mathbf{E(n-1)}}\frac{\mathbf{E(n-1)}}{\mathbf{E(n-2)}}\cdots\frac{\mathbf{E(1)}}{\mathbf{E_0}}\right| \\ &= \frac{1}{\mathbf{n}}\sum_{\mathbf{k=1}}^{\mathbf{n}}\ln\left|\frac{\mathbf{E(k)}}{\mathbf{E(k-1)}}\right|.\end{aligned} \tag{4.6}$$

Tekrarlama başına hata güçlendirme faktörleri nasıl tahmin edilebilir? $\mathbf{E(k+1)/E(k)}$ oranı, k. tekrarlamada x_k içinde olan E(k) ufak hatasının, sonraki tekrarlamada ne kadar büyüyeceğini veya azlacağını tanımlar. $\mathbf{E(k+1)/E(k)}$ hata güçlendirme faktörü, ε ufak olduğu müddetçe, $E(k)=\varepsilon$ hatasından bağımsızdır. Böylece gelişigüzel seçilen ufak ε hatası kullanılarak $\mathbf{E(k+1)/E(k)}$ hata güçlendirme faktörü, çok küçük ilk koşul hataları E_0 için, $\left|\tilde{E}(k+1)\right|/\varepsilon$ yardımıyla hesaplanır,

$$\left|\tilde{E}(k+1)\right| = f(x(k)+\varepsilon) - f(x(k)). \tag{4.7}$$

Sonuç olarak, Liapunov üstellerini hesaplamak için, gelişmiş bir yöntem elde edildi,

$$\frac{1}{\mathbf{n}}\ln\left|\frac{\mathbf{E(n)}}{\mathbf{E_0}}\right| = \frac{1}{\mathbf{n}}\sum_{\mathbf{k=1}}^{\mathbf{n}}\ln\left|\frac{\tilde{\mathbf{E}}(\mathbf{k})}{\varepsilon}\right|. \tag{4.8}$$

Tablo 4.1'de kullanılan ilk koşul değerleri için, üstteki formül kullanılarak bulunan sonuçlar Tablo 4.2'de verilmiştir. Her tekrarlama için, 0.001 hatası

kullanılmıştır. Tekrarlama sayısı arttıkça; üstel, $\lambda(x_0) = \ln 2 = 0.69314$ sayısına yakınsar.

Tablo 4.2. Farklı tekrarlama değerleri için, ortalama hata güçlendirme faktörlerinden hesaplanan Liapunov üstelleri.

Tekrarlamalar	$x_0 = 0.202$	$x_0 = 0.347$	$x_0 = 0.869$
10	0.62475	0.68873	0.72803
100	0.69435	0.69360	0.69708
1000	0.69368	0.69199	0.69004
10000	0.69322	0.69290	0.69337
100000	0.69307	0.69306	0.69327

Bu üstellerin daha dikkatli bir analizle ölçülmesi sonucunda, $\lambda(x_0) = 0.693$ sayısı elde edildi. Böylece lojistik dönüşüm için, ilk koşullara duyarlık, ölçülmüş oldu.

$\lambda(x_0)$ Liapunov üsteli, kararsız, kaotik davranıştan; kararlı ve tahmin edilebilir davranışı ayırmak ve bu özellikleri ölçmek için kullanılır. Eğer $\lambda(x_0) > 0$ koşulunu sağlayan $\lambda(x_0)$ büyük bir sayı ise, ilk koşullardaki ufak değişimlere olan duyarlık da büyüktür. Sonuç olarak Liapunov üstelleri, kaotik davranışı ölçmek, hesaplamak ve belirlemek için gerekli araçlardan biridir, [35].

4.2 Topolojik Geçişlilik

Şekil 4.5'te ufak hataların, tekrarlama sayısı arttıkça ne şekilde büyüdüğü görüldü. Şekil 4.5, biraz farklı bir görüş açısıyla da yorumlanabilir: küçük bir aralıktaki başlangıç koşulları, tekrarlama sayısı arttıkça, tüm [0,1] aralığına yayılır.

Topolojik geçişliliği yorumlamanın sezgisel bir yolu da, birim aralığı alt aralıklara ayırmak ve bu aralıklardan herhangibirini başlangıç, diğerlerini ise hedef olarak seçerek; belirli bir tekrarlama sayısından sonra tüm alt aralıklara ulaşılıp ulaşılmadığını kontrol etmektir. Eğer sonlu sayıda olan bu alt aralıkların tamamına ulaşılmışsa, sistemin topolojik geçişli olduğu söylenir.

Yukarıda ifade edilen işlem, bir örnek üzerinde irdelenebilir. I = [0,1] aralığı, 1/10 uzunluklu 10 alt bölgeye ayrılsın,

$\mathbf{I_k} = \left[\frac{\mathbf{k-1}}{10}, \frac{\mathbf{k}}{10}\right], \mathbf{k} = 1,...,10$ ve $\mathbf{I_2} = [0.1,0.2]$ aralığı da başlangıç olarak seçilsin.

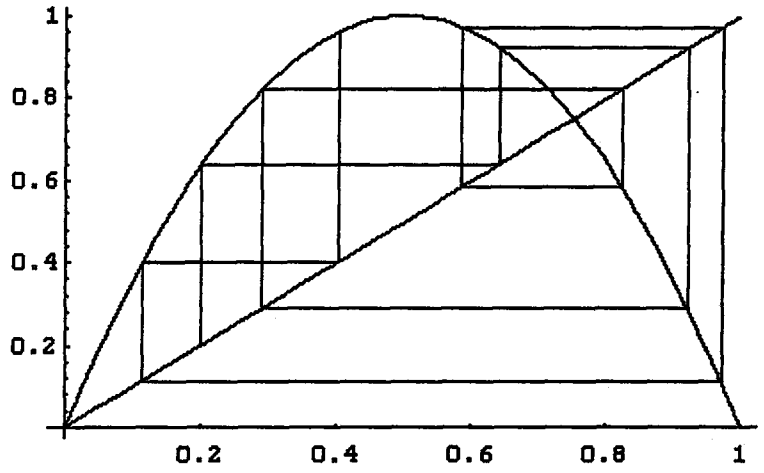
$\mathbf{I_2} = [0.1,0.2]$ aralığı ile işleme başlandığında ilk tekrarlamadan sonra, I_4, I_5, I_6 ve I_7 alt aralıklarını kapsayan [0.360, 0.640] aralığı bulunmuştur. İkinci tekrarlamadan sonra ise, [0.922, 1.000] aralığı elde edilir, yani I_{10} aralığına ulaşılır. Seçilen başlangıç aralığının üç kez tekrarlanması sonucu, I_1, I_2 ve I_3 alt aralıklarını simgeleyen [0.000,0.289] alt aralığı bulunur. Başlangıç koşulunun 4. tekrarında ise [0.000, 0.822] aralığı bulunur. Böylelikle [0.1, 0.2] ilk koşulu ile, 4 tekrarlamadan sonra 1/10 uzunluklu 10 alt aralığa ulaşılmıştır. Tablo 4.12'de, söylenilenlerin özeti verilmiştir. $\mathbf{I_2}$ yerine başka bir aralıktan işleme başlansaydı veya [0,1] birim aralığı, 10 alt aralığa değilde 100, 1000 veya bir milyon alt aralığa bölünseydi, yine aynı sonuç ile karşılaşılırdı.

Topolojik geçişlilik daha formel olarak şu şekilde ifade edilir: sıfırdan farklı olması koşuluyla gelişi güzel ufaklıkta seçilen herhangi iki I ve J açık aralığı için; I altaralığı içinde, itere edildiğinde nihayi olarak J altaralığına düşecek noktalar üreten bir ilk koşul değeri, bulunabilir.

Şekil 4.10, a = 4 için, lojistik dönüşümünün topolojik geçişli olduğunu gösterir.

Tablo 4.3. I_2 = [0.1,0.2] aralığı 4 kez tekrarlanması ile, başlangıçta belirlenen tüm alt aralıklara ulaşılmıştır.

Tekrarlama n	Aralık	Ulaşılan Aralıklar I_1 I_2 I_3 I_4 I_5 I_6 I_7 I_8 I_9 I_{10}
0	[0.100,0.200]	*
1	[0.100,0.200]	* * * *
2	[0.100,0.200]	*
3	[0.100,0.200]	* * *
4	[0.100,0.200]	* * * * * * * *

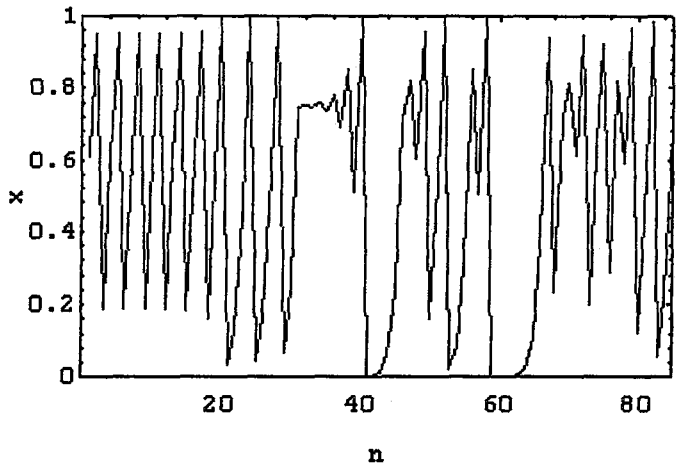


Şekil 4.10. Herhangibir J alt aralığına ulaşan bir ilk koşulun, I alt aralığında varolması.

4.3 Yoğun Periyodik Orbitler

Periyodik orbite neden olan bir ilk koşul ile işleme başlandığında, birkaç tekrarlama sonunda, tekrar seçilen ilk koşul değerine ulaşılır. Teorik olarak, her aralıkta periyodik bir orbite neden olan sonsuz sayıda değer vardır. Verilen bir alt aralıkta duyarlığın, topolojik geçişliliğin ve yoğun periyodik orbitlerin varlığı, kaos için gerekli koşullardır.

Periyodu 3 olan orbit üreten $\sin^2(\pi/7) = 0.1882550...$ ilk koşuluyla işleme başlansın. Bu sayı, bilgisayar aritmetiği ile tam olarak ifade edilemez. Bu sayıyı tek doğruluklu kayan nokta aritmetiği ile ifade ederken yapılan hata, 0.00000005 dir (aslında bu sayı kullanılan makineye bağlıdır). Bu durumda, dönüşümün ilk koşullara olan duyarlığını temel alarak, ne olacağı tahmin edilebilir. Yapılan hesaplamalar, en fazla 20 tekrarlamadan sonra, periyodu 3 olan çevrime yakın olunacağını ve daha sonra duyarlığın etkisinden dolayı çevrimden uzaklaşılacağını göstermiştir, [37]. Yani yuvarlatma hatalarından ve dönüşümün ilk koşullara olan duyarlığından dolayı, orbit, gerçek periyodik çevrimden uzaklaşır. Bu, şekil 4.14'te görülüyor. Aynı problem, tüm periyodik orbitlere neden olan değerler için geçerlidir. Periyodik bir orbit üreten değer, bilgisayarla tam olara ifade edilse bile, ilk tekrarlamadan sonra meydana gelen yuvarlatma hatasından ve duyarlılığın yıkıcı etkisinden dolayı, bilgisayarla hesaplanan orbit, gerçek periyodik orbitten uzaklaşır.



Şekil 4.11. $\sin^2(\pi/7) = 0.1882550$ ilk koşulunun ürettiği gerçek periyodik orbitten bilgisayardan kaynaklanan yuvarlatma hatalarından dolayı uzaklaşılması.

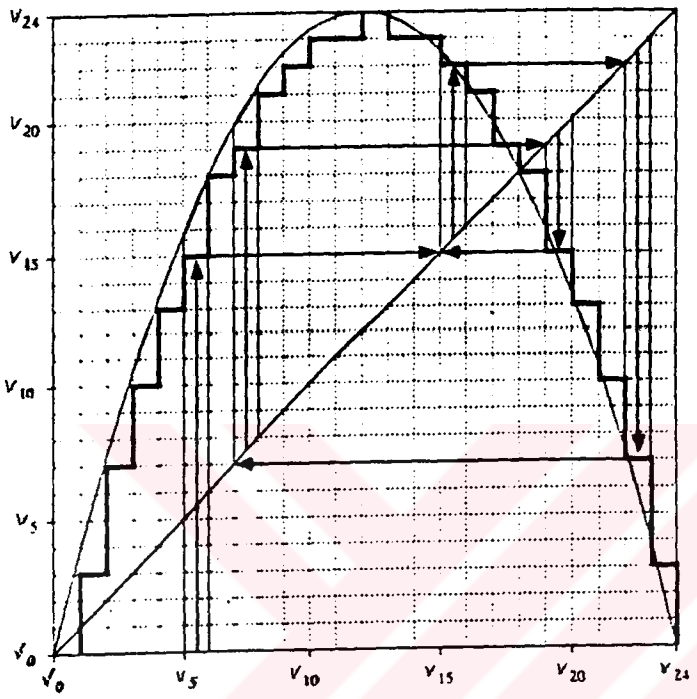
Sonuç olarak, parametrelerine duyarlı bir sistemin periyodik orbitlerini bilgisayarla keşfetmek mümkün değildir.

4.3.1 Periyodik Orbitleri Üreten Mekanizma

Periyodik orbitlere nasıl bir mekanizmanın neden olduğunu kavrayabilmek amacıyla, kullanılan aritmetiğin doğruluğu dramatiksel olarak azaltılsın. Şekil 4.12'de, [0,1] aralığının 25 alt aralığa bölündüğü görülmektedir. Bu örnekte, v_n ile v_{n+1} arasındaki değerler, v_n değerine kesilmiştir. v_0 ,...,v_{24} noktalarının, kesme hataları dikkate alınarak, lojistik dönüşüm altında hangi değerleri aldığı hesaplandığında, Şekil 4.12'de görülen yaklaşık parabol elde edilmiş ve lojistik dönüşümün , v_0,...,v_{24} noktalarını dönüştürdüğü değerler Tablo 4.4'te listelenmiştir.

5/24 ≈ 0.2083 den 6/24 = 0.25'e kadar olan reel sayıları ifade eden v_5 tekrarlama değeri şu şekilde açıklanır. Lojistik dönüşüm bu bölgeyi, sınırları 95/144≈ 0.6697 ve 3/4 = 0.75 sayıları ile belirlenen bölgeye dönüştürür. Yani bu durumda aralığın genişliği, 13/6≈2.17 faktörü ile artmıştır. v_5 değeri, lojistik dönüşümü altında v_{15} değerine dönüşür. v_5 ile ifade edilen gerçel sayıların bölgesi, v_{15}'e daralır. Bu noktadan sonra işleme devam edildiğinde v_{22}, v_1 , v_{19} ve tekrar v_{15} değeri bulunur. Dolayısıyla periyodu 4 olan bir dizi elde edilir. Sonuçta, v_n ile v_{n+1} arasındaki değerlerin v_n'e kesilmesi sonucu periyodu 3 olan, v_3, v_{10} ve v_{23} dizisi, v_0

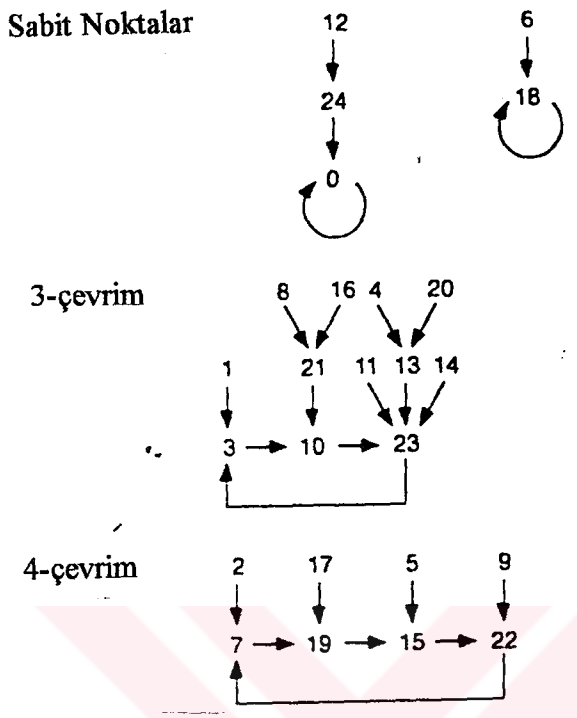
v_{18} sabit noktaları ve periyodu 4 olan v_{22} , v_1 , v_{19} , v_{15} dizisi bulunmuştur. Bu gerçekler, Şekil 4.13'te özetlenmiştir.



Şekil 4.12. Gerçek parabol ile dönüşüm uygulanmış noktaların meydana getirmiş olduğu merdiven fonksiyonunu.

Tablo 4.4. 25 farklı noktaya, dönüşüm altında karşılık gelen noktalar.

0	0	5	15	10	23	15	22	20	13
1	3	6	18	11	23	16	21	21	10
2	7	7	19	12	24	17	19	22	7
3	10	8	21	13	23	18	18	23	3
4	13	9	22	14	23	19	15	24	0



Şekil 4.13. Oldukca düşük doğruluklu aritmetiğin dört farklı uzun terim davranışına neden olması.

Yani bilgisayarla gözlemlenen orbitler, sınırlı prezisyondan kaynaklanan yapay orbitlerdir. Kullanılan aritmetiğin doğruluğu arttırıldığında, varolan periyodik orbitler kaybolur ve yenileri oluşur. Periyodik orbitlere neden olduğu söylenilen mekanizma, kayan nokta aritmetiği kullanıldığında da geçerlidir. $x \mapsto 4x(1-x)$ lojistik dönüşümünün tekrarlamaları sonucu meydana gelen orbitler, kullanılan sınırlı prezisyonlu aritmetikten dolayı birbiri üzerine düşer ve birkaç periyodik çevrim hayatta kalır.

4.4 Kaosu Yaratan Temel İşlem: Yoğurma

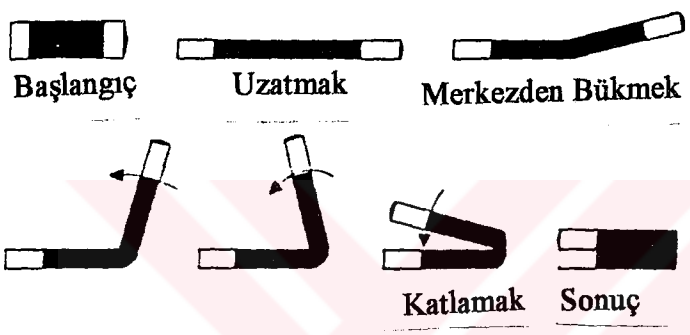
Lojistik dönüşümün belirli bir modelinin, kaotik davranışın anlaşılması için kullanılması oldukça güzel bir yoldur. Bununla birlikte hamurun yoğurulması işlemi, kaos teorisinin anlaşılması için seçilen iyi bir örnektir

Yoğurma işleminin kendisiyle ilgili hiçbir rasgeleliği yoktur. Aksine belirli bir hareket devamlı olarak tekrar edilmektedir. Yoğurma işlemi, çekme, uzatma ve

katlama işlemlerinin söylendikleri sıra ile peşpeşe uygulanması şeklinde düşünülebilir. Bu belirli tanıma rağmen, işlem sonunda elde edilen sonuçlar rasgele olayların sahip olduğu pek çok özelliği içerir.

4.4.1 Yoğurma İşleminin Çekme ve Katlama Şeklinde Yorumlanması

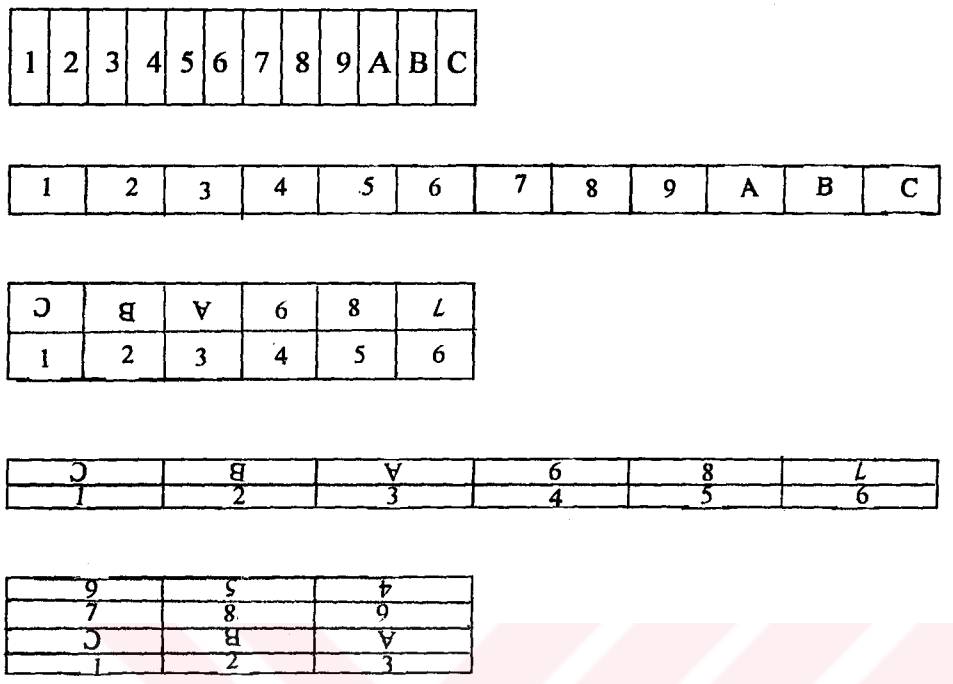
Bu işlem, hamurun uzunluğunun iki katına çekilmesi ve merkezinden kendi üzerine katlanması şeklinde düşünülür. Şekil 4.14, bu işlemi göstermektedir.



Şekil 4.14. Yoğurma işleminin uzatma - katlama işlemi ile yorumlanması.

Acaba hamurun değişik bölümleri üzerinde bu yoğurma işlemi nasıl bir etki yapmaktadır? Bunu görebilmek amacıyla, hamur 12 bölüme ayrılarak, uzatma ve çekme işlemleri uygulansın. Bu işlemlerin iki kez uygulanmasıyla elde edilen hamurun, bir miktar karıştığı Şekil 4.15'te görülmektedir.

Bu adımdan sonra artık ideal bir durum ele alınsın. Yani sonsuz incelikte hamur tabakalarıyla çalışıldığı ve katlama işlemi sonucu hamurun kalınlığının değişmediği varsayılsın. Dolayısıyla hamur, Şekil 4.16'da olduğu gibi bir çizgiyle ifade edilebilir. Şekilde, başlangıçta birbirlerine oldukça yakın olan farklı iki meteryalin birkaç tekrarlamadan sonra nerede bulunacağının tahmin edilemeyeceği görülmektedir. Bu, yoğurma işleminin topolojik geçişliliği olarak verilir. Başka bir deyişle, başlangıçta birbirine yakın olan iki farklı madde, bir süre sonra komşu olacak kadar yakın olamazlar. Bu, ilk koşullara duyarlığın bir etkisidir. Yani başlangıçta ilk koşullarda olan ufak sapmalar, işlem süresince büyük sapmalar şeklini alır.

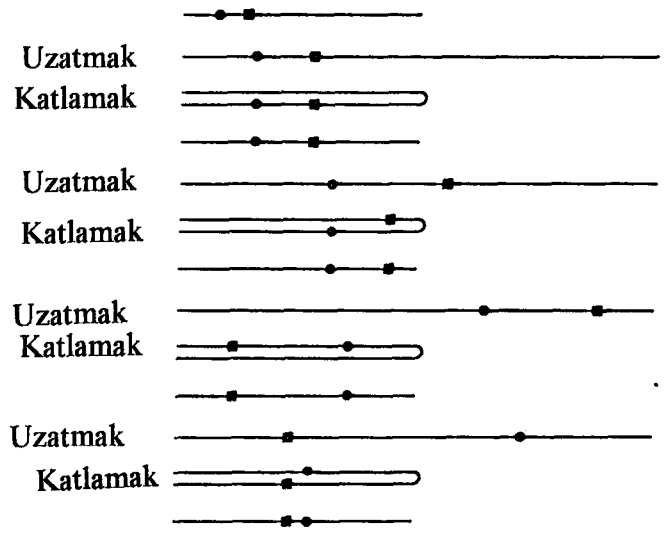


Şekil 4.15. 12 bölüme ayrılmış hamura iki kez çekme - katlama işleminin uygulanması.

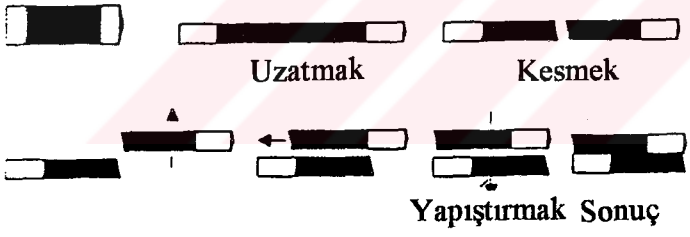
4.4.2 Yoğurma işleminin Çekme - Katlama ve Yapıştırma Şeklinde Yorumlanması

Şimdi ise çekme - kesme ve yapıştırma işlemleri şeklinde yorumlanan ikinci bir yoğurma işlemi ele alınacaktır, Şekil 4.17. Hamur, diğer işlemin analizinde olduğu gibi tekrar 12 parçaya bölünsün ve çekme - kesme ve yapıştırma işlemlerinden sonra uzatma - katlama işlemi uygulansın. Bu işlem sonunda elde edilen sonuçla, iki çekme - katlama isleminin peşpeşe uygulanmasıyla elde edilen sonuç karşılaştırılsın. Parçaların yatay sıraları göz önüne alınmaz ise, sonuçların aynı olduğu Şekil 4.18'de görülmektedir.

Hamur, yoğurma işleminin matematiksel olarak modellenmesi amacıyla, [0,1] aralığını temsil eden bir doğru şeklinde alınır. Bu durumda, iki farklı yoğurma işleminin özellikleri daha kolay anlaşılabilir. Çekme -katlama işlemi için T sembolü, çekme - kesme ve yapıştırma işlemleri içinse S sembolü kullanılsın.



Şekil 4.16. Çekme - katlama işleminin, çizgi ile ifade edilen hamura dört kez uygulanması.

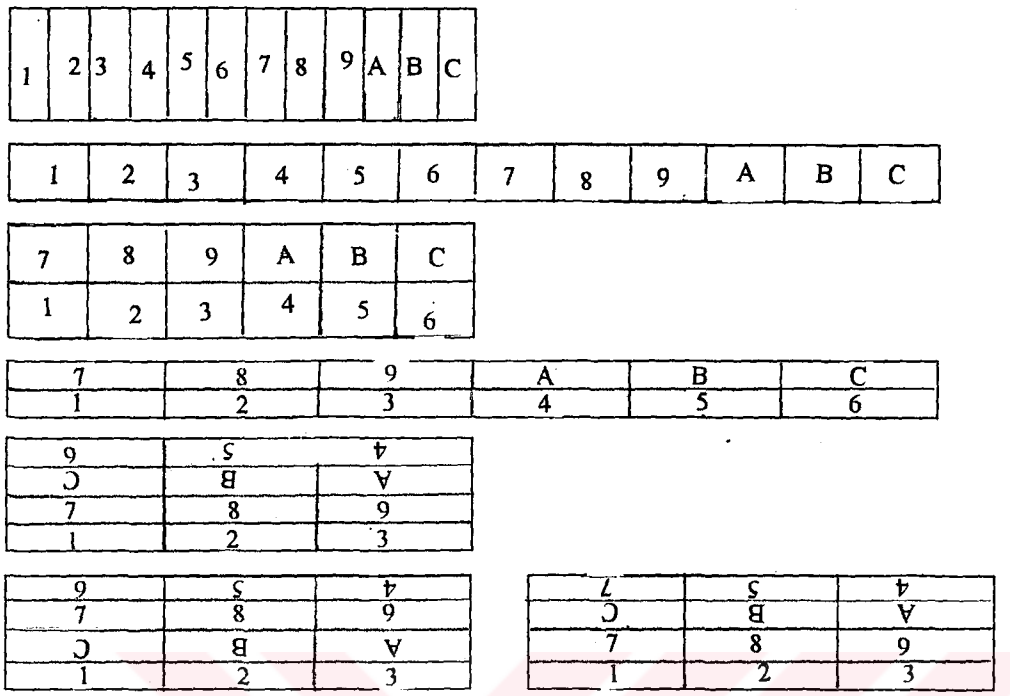


Şekil 4.17. Çekme - kesme ve yapıştırma işlemleri ile yorumlanan yoğurma işlemi.

Örneğin T işlemi, hamuru modelleyen [0,1] aralığına üç kez uygulansın ve yine aynı aralığa iki kez S işlemi ve daha sonrada T işlemi uygulansın. Gözlemlenilen, başlangıçtaki parçacığın, her iki farklı işlem sonucunda da aynı yerde olmasıdır. Yani parçacık başlangıçta bir x konumundaysa, aşağıdaki durum elde edilir,

$$T(T(T(x))) = T(S(S(x))). \qquad (4.9)$$

Her iki işlemin meydana getirdiği etki, Şekil 4.19'da görülmektedir.



Şekil 4.18. 12 parçaya ayrılmış hamura çekme - katlama işleminin uygulanmasının ardından, çekme - kesme ve yapıştırma işleminin uygulanması ve bu sonucun, aynı hamura iki kez çekme - katlama işlemi uygulanmış sonuçla karşılaştırılması.

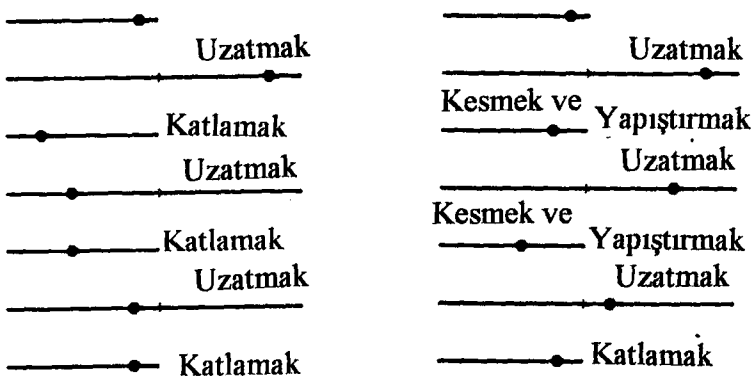
Çekme - katlama işleminin (T), N kez uygulanması, çekme - kesme ve yapıştırma işlemlerinin (S), N-1 kez ve ardındanda bir kez T işleminin uygulanması ile özdeştir,

$$T^{N} = T\,S^{N-1} \tag{4.10}$$

Hamurun yoğurulma işlemi, matematiksel olarak bir fonksiyonla modellenebilir. Çekme - katlama işlemi,

$$T(x) = \begin{cases} 2x & , x \geq 0.5 \\ -2x + 2, & x > 0.5 \end{cases} \tag{4.11}$$

şeklinde ifade edilir ve Şekil 4.20, bu dönüşümün grafiğini gösterir. Bu dönüşüm, şeklinden dolayı çadır dönüşümü olarak adlandırılır.

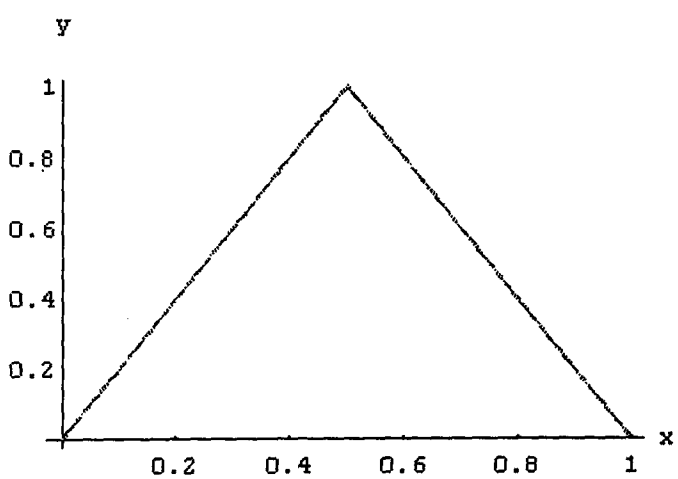


Şekil 4.19. x=9/10 ilk koşuluna, üç kez T işleminin uygulanması solda, S, S, T işlemlerinin uygulanması ise sağda gösterilmiştir.

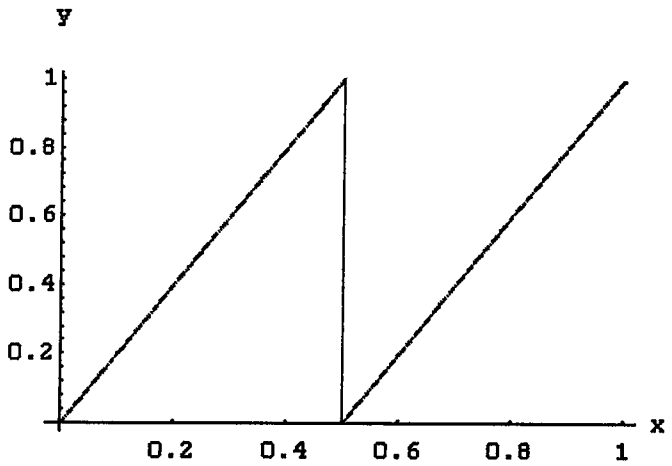
İkinci yoğurma işlemini modellemek için, başka bir temel matematiksel dönüşüm kullanılır. Testere dişi (saw-tooth) dönüşümü olarak çağrılan bu S dönüşümü, [0,1] aralığındaki x ler için

$$S(x) = \begin{cases} 2x & , x < 0.5 \\ 2x - 1, & x \geq 0.5 \end{cases} \qquad (4.12)$$

şeklinde tanımlanır ve Şekil 4.21, bu dönüşümün grafiğini gösterir.



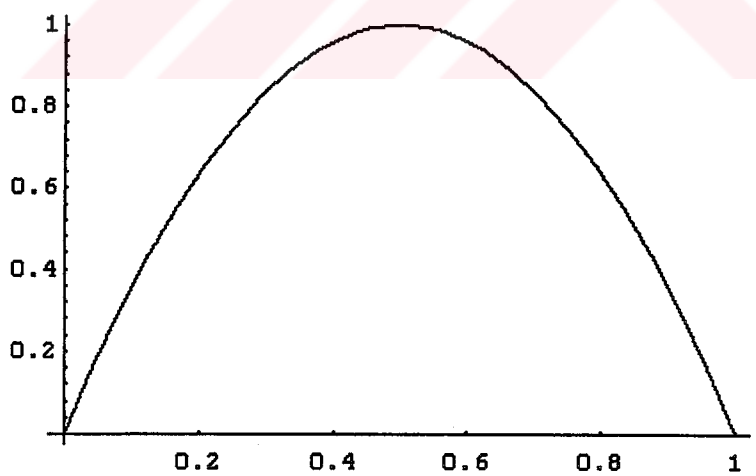
Şekil 4.20 Çekme - katlama işlemine karşı gelen çadır dönüşümü.



Şekil 4.21 Çekme - kesme ve yapıştıma işlemlerine karşı gelen testere dişi dönüşümü.

4.4.3 Lojistik Dönüşümde Yoğurma İşlemi

Burada $x \mapsto 4x(1-x)$ dönüşümü ile hamurun yoğurulması işlemi arasındaki ilişki anlaşılmaya çalışılacaktır. Lojistik dönüşüm, x-y düzleminde ifade edildiğinde, Şekil 4.22'deki parabol elde edilir.



Şekil 4.22. $x \mapsto 4x(1-x)$ lojistik dönüşümünün grafiği.

Lojistik dönüşüm için, [0,1] aralığındaki değerlere karşı gelen y değerleri de [0,1] aralığındadır. Buna ilave olarak y değerleri, $x < 1/2$ koşulunu gerçekleyen değerler için monotoniksel olarak artarken, $x > 1/2$ değerleri için monotoniksel olarak azalır. x ekseni üzerindeki [0,1/2] aralığı, y ekseni üzerinde [0,1] aralığına

çekilir. Aynı şey, [1/2, 1] aralığı içinde geçerlidir. Başka bir deyişle lojistik dönüşüm, her iki aralığı, uzunluklarının iki katına çeker.

Buna rağmen, uzama işlemi homojen ("uniform") değildir. 1/2 noktasına yakın ufak aralıklar bastırılırken, 0 veya 1'e yakın ufak aralıklar büyük miktarda uzatılır. Şekil 4.22'de, eşit aralıklarla seçilen 5 ilk koşulun dönüşümleri sonucu, düzenli olmayan uzamaların oluştuğu görülüyor.

0, 1/2 ve 1 noktalarının dönüşümleri düşünüldüğünde, 0→0, 1/2→ 1, 1→0 değerleri bulunur. O halde lojistik dönüşümün [0,1] aralığına uygulanması, çekme ve katlama işlemlerinin birleşimi olarak düşünülmelidir.

Lojistik dönüşüm için kaosun özelliklerini anlamak amacıyla, çadır dönüşümünün tekrarları düşünülecek ve her iki dönüşümün birbirine özdeş olduğu gösterilecektir. Bu amaçla tüm ilk koşul değerleri ve tüm tekrarlama sayıları için, $T^k(x_0)$ değerinin veren kapalı bir formül bulunacaktır. Çadır dönüşümü ile testere dişi dönüşümü arasında,

$$\mathbf{T^k(x_0) = TS^{k-1}(x_0)} \tag{4.13}$$

bağıntısının olduğu görülmüştü. Bu ifadeden görüldüğü gibi, çadır dönüşümü için kapalı bir formülün bulunabilmesi için, $\mathbf{S^{k-1}(x_0)}$ ifadesinin hesaplanması gerekmektedir.

4.4.4 Testere Dişi Dönüşümü

Testere dişi dönüşümü, Frac(x) dönüşümü kullanılarak yeniden tanımlanabilir. Frac(x) dönüşümü, k bir tamsayı olmak üzere,

$$\mathbf{eğer\ k \le x < k+1\ ise\ Frac(x) = x - k} \tag{4.14}$$

şeklinde tanımlanır. Bazı sayıların bu dönüşüm altında aldığı değerler aşağıda görülmektedir,

Frac(0.4) = 0.4

Frac(5.123) = 0.123

Frac(18) = 0

Frac(24/7) = 3/7.

Testere dişi dönüşümü, Frac(x) dönüşümü kullanılarak

$$0 \le x < 1 \text{ için } \mathbf{S(x) = Frac(x)} \tag{4.15}$$

şeklinde yeniden tanımlanabilir.

$0 \le x < 1$ koşulunu sağlayan x_0 değerinin, Frac(x) dönüşümü altındaki ilk k+1 tekrarlaması,

$$\begin{aligned} \mathbf{x}(1) &= \mathbf{Frac}(2\mathbf{x}_0) \\ \mathbf{x}(2) &= \mathbf{Frac}(2\mathbf{x}(1)) \\ &\vdots \\ \mathbf{x}(\mathbf{k}+1) &= \mathbf{Frac}(2\mathbf{x}(\mathbf{k})), \quad \mathbf{k} = 0,1,2,3,... \end{aligned} \tag{4.16}$$

şeklinde hesaplanır. x(k) değeri, yalnızca x_0 değerine bağlı olarakta ifade edilebilir,

$$\mathbf{x}(\mathbf{k}) = \mathbf{Frac}(2^{\mathbf{k}}\mathbf{x}_0). \tag{4.17}$$

Bu bölümde şimdiye kadar iki önemli araç bulundu. Bunlardan biri, yoğurma işleminin yerdeğiştirme özelliği, diğeri ise testere dişi dönüşümünün k. tekrarlama değerinin veren kapalı formüldür. Böylece, x_0 ilk koşulunun k tane çekme - katlama işleminden sonra aldığı x(k) değeri bilinmek istenirse, yerdeğiştirme özelliği kullanılarak ilk k-1 iterasyon, çekme - kesmeme ve yapıştırma işlemlerinin (S) kapalı formülü kullanılarak hesaplanır, $\mathbf{y} = \mathbf{S}^{k-1}(\mathbf{x}_0) = \mathbf{Frac}(2^{k-1}\mathbf{x}_0)$ ve ardından bu işlem sonunda elde edilen değere bir kez çekme - katlama işlemi (T) uygulanır,

$$\mathbf{T(y)} = \begin{cases} 2\mathbf{y}, & \mathbf{y} \le 0.5 \\ -2\mathbf{y}+2, & \mathbf{y} > 0.5. \end{cases} \tag{4.18}$$

4.5 Kaosun Analizi

$0 \le x < 1$ için $S(x) = \mathbf{Frac}(2x)$ olarak tanımlanan testere dişi dönüşümü, 0 ve 1 arasındaki x gerçel sayısının ikili gösterimi ele alınarak farklı bir açıdan yorumlanmaya çalışılsın. Birim aralıktaki herhangibir sayı, $x = 0.a_1\ a_2\ a_3...$ şeklinde yazılabilir. İfadede görülen a_k sembolü ikili dijitleri ifade eder. Yani a_k ifadesi 0 veya 1 ve x sayısının onluk karşılığı, $x = a_1\ 2^{-1} + a_2\ 2^{-2} + a_3\ 2^{-3} + ...$ şeklindedir. Örneğin 1/2 = 0.100..., 3/4 = 0.1100..., 1/3 = $0.\overline{01}$, 1/7 = $0.\overline{001}$. Birim aralıkta ikili açılımı $x = 0.a_1\ a_2\ a_3...$ ve $y = 0.b_1\ b_2\ b_3...$ olan iki sayı ele alınsın. Eğer x ve y sayılarının ilk k dijiti aynı ise, $|x - y| \le 2^{-k}$ koşulu sağlanır.

4.5.1 İki Sembol Üzerine Öteleme Dönüşümü

$0.a_1a_2a_3...$ sayısının 2 ile çarpılması, tüm dijitlerin bir sola kaymasına neden olur. Yani $0.a_1\ a_2\ a_3...$ sayısı, 2 ile çarpıldıktan sonra, $a_1.a_2a_3...$ şekline gelir. Bu nedenle $0.a_1\ a_2\ a_3...$ sayısına S dönüşümü uygulandıktan sonra, $0.a_2\ a_3\ a_4...$ sayısı elde edilir,

$$x = 0.a_1\ a_2\ a_3... \rightarrow S(x) = 0.a_2\ a_3\ a_4 \qquad (4.19)$$

Yani S dönüşümü, dijitlerin bir sola kaymasına ve nokta önündeki dijitin silinmesine neden olmaktadır. Bu dönüşüm, ikili açılım kullanılarak yorumlandığında öteleme dönüşümü olarak da adlandırılır.

4.5.1.1 İlk Koşullara Duyarlık

N dijiti belli olan $x_0 = 0.a_1\ a_2\ a_3...$ sayısı göz önüne alınsın, örneğin N = 100 olsun. Gerçek sayı, N dijitle belirlenmiş olandan en fazla 2^{-100} kadar farklıdır. Pek çok olayda ölçme hatası olarak yorumlanan bu kadar küçük bir farkın, hiç önemli olmadığı düşünülebilir. a_{101}, a_{102}, a_{103} ve daha sonraki dijitler bilinmediği için, hesaplama, her adımında sanki birisi bir demir para atıp bu dijitleri belirliyormuş gibi düşünülebilir (yazı = 1 ve tura = 0). İlk koşul değeri, 101. ve daha sonraki dijitleri etkiler.

$x_0 = 0.a_1\ a_2\ a_3...$ başlangıç değeri ile S(x) işlemine başlandığında başlangıçta herşey olağan gözükür; fakat işleme devam edildiğinde, sonuçlar, tamamıyla rasgele şekil alır. Bu olay, ilk koşullara duyarlık olarak adlandırılır.

Yoğurma işlemi ile farklı maddenin hamura düzenli bir şekilde yayıldığı söylenebilir. Eğer madde, hamurun üzerine bir grup şeklinde gelirse, parçacıkların koordinatları, $x_0 = 0.a_1\ a_2\ a_3...a_k\ a_{k+1}...$ şeklinde verilir. Burada parçacıkların grup şeklinde olduğu varsayıldığı için ilk k dijit aynıdır. Diğer dijitler, topluluk içindeki maddelerin rasgele karışımını ifade edecek şekilde, düzgün olarak dağılmışlardır. k işlemden sonra, ortak koordinatlar gitmiş olur ve geriye rasgele olanlar kalır. Kalan dijitler, maddenin tüm hamur içinde düzgün dağılımını verir.

Duyarlıkla ilgil bir tanım şu şekilde verilebilir. 0 ve 1 arasında verilen herhangibir x_0 değeri için, x_0 noktasına gelişigüzel yakın bir y_0 noktası vardır. Öyleki öteleme dönüşümünün x_0 ve y_0 noktalarında başlayan tekrarlamaları, nihayi durumda birbirlerinden belirli bir eşik değeri kadar farklıdır. Bu eşik değeri, birim aralıktaki tüm x_0 değerleri için aynıdır ve duyarlık sabiti olarak çağrılır. x_0'a yakın başlayan tüm orbitlerin bu eşik değerini aşması, duyarlık özelliği için gerekli değildir.

4.5.1.2 Periyodik Orbitler

Eğer x_0 ilk koşulu, $x_0 = 0.\overline{a_1 a_2 a_3 ... a_k}$ şeklinde ise, nasıl bir durumla karşılaşılacağı bu bölümde irdelenecektir. k dijitte bir kendini tekrar eden sonsuz uzunluklu bir dizi ele alındığında, öteleme dönüşümünün her k tekrarlamasında bir x_0 ile karşılaşılır. Yani uzunluğu k olan bir çevrim oluşur. Bu durumda x_0 ilk koşulu, öteleme dönüşümüme göre periyodik, olarak adlandırılır. Açıkça görülen, herhangibir uzunlukta çevrim üretilebileceğidir. Fakat daha önemli olan, verilen herhangi bir x_0 noktasına gelişigüzel yakın ve periyodik orbit üreten bir ω_0 değeri bulunabilir.

$x_0 = 0.a_1\ a_2\ a_3...$ olsun ve ω_0, $\omega_0 = 0.\overline{a_1 a_2 a_3 ... a_k}$ olarak seçilsin. Bu durumda x_0 ve ω_0 birbirlerinden en fazla 2^{-k} kadar farklıdır ve ω_0, periyodik bir orbit üretir. Bu, periyodik orbitlerin yoğun olması anlamına gelir.

4.5.1.3 Topolojik Geçişlilik

Gelişigüzel seçilen iki ufak I ve J aralıkları için, dönüşümün belli bir tekrarlama sayısından sonra J alt aralığına gidecek bir x_0 başlangıç koşulu, I alt aralığında bulunabiliyorsa, bu alt aralıkların topolojik geçişli olduğu söylenir. Öteleme dönüşümü için, bu özelliğin de sağlandığı görülecektir.

Herhangi iki tane I ve J aralıkları seçilsin ve n, I aralığının uzunluğu $1/2^{n-1}$ sayısından büyük olacak şekilde belirlenen bir sayı olsun. İlk aralığın orta noktasının ikili gösterimi, $0.a_1\ a_2\ a_3...$ ve ikinci aralıktaki bir y noktasının ikili açılımı da $0.b_1b_2b_3...$ olarak verilsin. Bu durumda öteleme dönüşümünün n kez tekrarlanmasından sonra, y sayısına eşit olacak yani J hedef aralığında bulunacak bir x_0 başlangıç koşulu bulunmaya çalışılacaktır. Bu amaçla, x_0 başlangıç koşulu, I alt aralığının orta noktasının ilk n dijiti ve buna ilave olarakda y hedef noktasının dijitleri olarak tanımlanır, $x_0 = 0.a_1a_2a_3...a_nb_1b_2b_3...$ x_0 başlangıç koşulu, I aralığının orta noktasından en fazla 2^{-n} kadar farklıdır. Yani belirlenen bu ilk koşul, I aralığı tarafından içerilir ve bu ilk koşulun, öteleme dönüşümü altında n kez tekrarından sonra,

$x(n) = 0.b_1\ b_2\ b_3... = y$ noktası elde edilir.

4.5.2 Çadır Dönüşümü

Testere dişi dönüşümünün (öteleme dönüşümünün veya çekme - kesme ve yapıştırma işleminin), kaosun üç özelliğini sağladığı görüldü. Bu bölümde ise çadır dönüşümü için kaosun üç özelliği tartışılacaktır. T dönüşümü, yerdeğiştirme özelliği yardımıyla, S dönüşümünün tekrarlamalarına indirgenebilir. x_0 sayısının k. tekrarlaması, k-1 tane öteleme dönüşümünün ardından bir kez çekme - katlama işleminin uygulanmasıyla hesaplanır.

4.5.2.1 Yoğun Periyodik Orbitler

Bu bölümde, öteleme dönüşümü için periyodik bir orbit üreten noktanın, çekme - katlama işlemi içinde periyodik bir orbit ürettiği anlaşılmaya çalışılacaktır.

Verilen bir n tamsayısı için, $x(n) = x_0$ eşitliğini sağlayan bir x_0 ilk koşulunun bulunması, bu ilk koşulunun n periyotlu bir orbit üretmesi anlamına gelir. Öteleme dönüşümü için, n periyotlu bir orbit üreten ω_0 noktasının, çadır dönüşümü altında, $x_0 = T(\omega_0)$, periyodik bir orbit üreteceği iddia edilmektedir.

ω_0, S dönüşümü altında periyodik bir orbit üretsin, $\omega_0 = S^n(\omega_0)$. Bu durumda x_0 ilk koşulunun tanım bağıntısı ve yerdeğiştirme özelliği kullanılarak, $x(n) = x_0$ eşitliğinin gerçeklendiği gözlemlenir,

$$x(n) = T^n(x_0) = T^n(T(x_0)) = T^{n+1}(\omega_0)$$
$$= T(S^n(\omega_0)) = T(\omega_0) = x_0. \qquad (4.20)$$

Böylece eğer ω_0, öteleme dönüşümü için n periyotlu bir orbit üretirse, x_0 noktası da ($x_0 = T(\omega_0)$) çekme - katlama işlemi (T) için, n periyotlu bir orbit üretir.

T dönüşümünün, Kaosun üç özelliğini de sağladığı, bu dönüşümün ikili açılımı kullanılarak incelenecektir. Çadır dönüşümünün matematiksel ifadesi tekrar göz önüne alınsın,

$$T(x) = \begin{cases} 2x & , x \le 0.5 \\ -2x+2, & x > 0.5. \end{cases} \qquad (4.21)$$

$x \in [0,1]$ olmak üzere, $x = 0.a_1a_2a_3...$ şeklinde verilsin. Eğer $x < 1/2$ ise, çadır dönüşümü, testere dişi dönüşümüne özdeştir. Yani, $x < 1/2$ ise T(x),

$T(x) = 0.a_2a_3a_4...$ şeklinde ifade edilir. Eğer $x \ge 1/2$ ise, $S(x) = 2x - 1$ olduğundan T(x),

$$T(x) = -2x + 2 = 1 - (2x - 1)$$
$$= 1 - S(x) = 1 - 0.a_2a_3a_4... \qquad (4.22)$$

şeklinde ifade edilir. Eşlek ("dual") ikili dijit tanımı şu şekilde verilir,

$$a^* = \begin{cases} 1, a = 0 \\ 0, a = 1. \end{cases} \tag{4.23}$$

Bu durumda, $0.a_2a_3a_4...+0.a_2^*a_3^*a_4^*...= 0.111...= 1$ olduğundan,

$x \geq 1/2$ için T(x),

$$T(x) = 0.a_2^*a_3^*a_4^*... \tag{4.24}$$

şeklinde tanımlanır. Böylece çadır dönüşümünün ikili karşılığını veren ifade,

$$T(0.a_1a_2a_3...) = \begin{cases} 0.a_2a_3a_4..., & a_1 = 0 \\ 0.a_2^*a_3^*a_4^*..., & a_1 = 1 \end{cases} \tag{4.25}$$

şeklindedir.

Testere dişi dönüşümünün $S^n(\omega) = \omega$ eşitliğini sağlayan $\omega \in [0,1)$ noktasının, çadır dönüşümü için periyodik orbit üreten $x = T(\omega)$ periyodik noktasına neden olduğu görüldü. Periyodik orbitlerin yoğun olduğu, ikili açılımı gelişigüzel $a_1 ... a_n$ dizisi ile başlayan ve periyodik orbit üreten noktaların bulunabileceğinin gösterilmesi ile tanıtlanır. $\omega = 0.\overline{0a_1...a_n} < 1/2$ noktası, S dönüşümü altında (n + 1) periyotlu orbit üretir. T dönüşümü, ilk dijit 0 ise, öteleme dönüşümüne özdeştir. Bu durumda $x = T(\omega) = 0.\overline{a_1...a_n 0}$ noktası bulunur ve bu nokta, T dönüşümü altında (n + 1) periyotlu orbit üretir .

Örneğin ikili açılımı 0.11011 ile başlayan sayıları içeren I = [27/32,28/32] aralığı ele alındığında, aralığın orta noktası, 55/64 = 0.110111 dir. Yukarıda söylenilenler dikkate alınarak x, şu şekilde seçilir,

$$x = T(0.\overline{0110111}) = 0.\overline{1101110} = \frac{110}{127} \in I. \tag{4.26}$$

Hakikaten x, T dönüşümü altında periyodu 7 olan bir orbit üretir.

4.5.2.2 İlk Koşullara Duyarlık

$x_0 = 0.a_1a_2a_3...$, birim aralıktan gelişigüzel seçilen bir sayının ikili açılımı olsun. Verilen bir n sayısı için, $n > 0$ olmak üzere, x_0 noktasına yakın bir z_0 noktası araştırılacaktır. Öyleki bu değerler, hem $|z_0 - x_0| < 2^{-n}$ eşitsizliğini sağlamalı hem de bu noktaların dönüşüm altındaki bazı tekrarlamaları, birbirlerinden $\varepsilon = 1/2$ değeri kadar farklı olmalıdır. z_0 değeri $z_0 = 0.\mathbf{a}_1...\mathbf{a}_n\mathbf{a}^*_{n+1}\mathbf{a}_{n+1}\mathbf{a}_{n+2}\mathbf{a}_{n+3}...$ şeklinde seçilsin. İyi bir kestirim ile $|z_0 - x_0| < 2^{-n}$ olduğu elde edilir. Bu durumda dönüşümün n. tekrarlamasında $|z(n) - x(n)| = 1/2$ eşitliğinin gerçeklendiği iddia edilsin. Tanıt için, $a_n = 0$ ve $a_n = 1$ durumları ayrı ayrı düşünülecektir.

$a_n = 0$ durumu:

$$\begin{aligned}
\mathbf{x(n)} &= \mathbf{T^n(x_0) = T(S^{n-1}(x_0)) = T(0.a_n a_{n+1} a_{n+2}...)} \\
&= \mathbf{T(0.0a_{n+1} a_{n+2} a_{n+3}...) = 0.a_{n+1} a_{n+2} a_{n+3}...} \\
\mathbf{z(n)} &= \mathbf{T^n(z_0) = T(S^{n-1}(z_0)) = T(0.a_n a^*_{n+1} a_{n+2} a_{n+3}...)} \\
&= \mathbf{T(0.0a^*_{n+1} a_{n+2} a_{n+3}...) = 0.a^*_{n+1} a_{n+2} a_{n+3} a_{n+4}...}
\end{aligned} \tag{4.27}$$

elde edilir. Böylece ilk durum için iddianın tanıtlandığı görülmektedir.

$a_n = 1$ durumu:

$$\begin{aligned}
\mathbf{x(n)} &= \mathbf{T(0.1a_{n+1} a_{n+2} a_{n+3}...) = 0.a^*_{n+1} a^*_{n+2} a^*_{n+3}...} \\
\mathbf{z(n)} &= \mathbf{T(0.1a^*_{n+1} a_{n+2} a_{n+3}...) = 0.a_{n+1} a^*_{n+2} a^*_{n+3}...}
\end{aligned} \tag{4.28}$$

elde edilir. Yani ikinci durum için de iddianın tanıtlandığı görülmektedir.

4.5.2.3 Topolojik Geçişlilik

Birim aralık içinde gelişigüzel iki açık **I** ve **J** aralıkları verilsin. İkili açılımı $a_1...a_n$ ile başlayan sayılar, I aralığı içinde; $b_1...b_n$ ile başlayanlar ise J aralığında kalacak şekilde, $a_1...a_n$, $b_1...b_n$ bitlerini ve n sayısını yeterince büyük seçmek daima mümkündür. Burada I aralığı içinde, n tekrarlamadan sonra J içinde kalan bir x_0 ilk

koşulu bulunmaya çalışılsın. Yani $x_0 \in I$ olmak üzere, $x(n) = T^n(x_0) \in J$ eşitliğini sağlayan x_0 noktası belirlenecektir. Yine x_0 noktası, $a_n = 0$ ve $a_n = 1$ durumlarının ayrı ayrı ele alınmasıyla saptanacaktır.

$a_n = 0$ durumu:

$x_0 = 0.a_1 \ldots a_n b_1 \ldots b_n$ seçilsin. Bu durumda $x_n \in J$ olduğu şu şekilde doğrulanabilir,

$$\begin{aligned} \mathbf{x(n) = T^n(x_0) = T\left(S^{n-1}(x_0)\right) = T(0.a_n b_1 \ldots b_n)} \\ \mathbf{= T(0.0 b_1 \ldots b_n) = 0.b_1 \ldots b_n \in J.} \end{aligned} \tag{4.29}$$

$a_n = 1$ durumu:

$\mathbf{x_0 = 0.a_1 \ldots a_n b_1^* \ldots b_n^*}$ olarak seçilsin. Bu durumda $x(n) \in J$ olduğu aşağıdaki gibi doğrulanabilir,

$$\begin{aligned} \mathbf{x(n) = T^n(x_0) = T\left(S^{n-1}(x_0)\right) = T(0.a_n b_1^* \ldots b_n^*)} \\ \mathbf{= T(0.0 b_1^* \ldots b_n^*) = 0.b_1 \ldots b_n 111 \in J.} \end{aligned} \tag{4.30}$$

Yani öteleme dönüşümü ve onun çadır dönüşümü ile olan ilişkisi, kaosun üç özelliğinin, çadır dönüşümü için anlaşılmasını sağlamıştır.

4.6 Lojistik Dönüşüm İçin Kaos

Bu bölümde düzgün yoğurma işleminin fonksiyonel gösterimi olan çadır dönüşümü ile düzgün olmayan yoğurma işlemine karşı gelen $\mathbf{x \mapsto 4x(1-x)}$ lojistik dönüşümün özdeş olduğu gösterilecektir. Yani öteleme dönüşümü ve çadır dönüşümü için gösterilen tüm karmaşık davranış, lojistik dönüşümde de gözlenir.

Çadır dönüşümü ile lojistik dönüşümün özdeşliği, koordinatlarda doğrusal olmayan değişimi sağlayan,

$$\mathbf{x' = h(x) = \sin^2\left(\frac{\pi x}{2}\right)} \tag{4.31}$$

eşitliğinden yararlanılarak gösterilecektir. Şekil 4.23, h fonksiyonunun S şeklindeki grafiğini göstermektedir. h fonksiyonu, [0,1] birim aralığını kendi üzerine dönüştürmektedir. Yani $x' \in [0,1]$ aralığındaki tüm değerlerin her birine karşı, $x \in [0,1]$ aralığında $x' = h(x)$ eşitliğini sağlayan tek bir değer vardır. Bu fonksiyon, çadır dönüşümünün dinamiğini, lojistik dönüşümün dinamiğine nasıl çevirmektedir? Bu dinamik dönüşümünün nasıl gerçeklendiği aşağıda ifade edilecek olan matematiksel işlemler ile sunuldu. x_0 ilk koşulunun ürettiği x_0, x_1, x_2,... orbitinin çadır dönüşümü altındaki durumu ele alınsın,

$$x(1) = T(x_0), x(2) = T^2(x_0), \ldots, x(k) = T^k(x_0), \ldots \tag{4.32}$$

Dönüşüm uygulanmış ilk koşul, $x_0' = h(x_0)$ şeklindedir. $y_0 = x_0'$ değerinin $x \mapsto f(x) = 4x(1-x)$ lojistik dönüşümü altındaki tekrarlamaları,

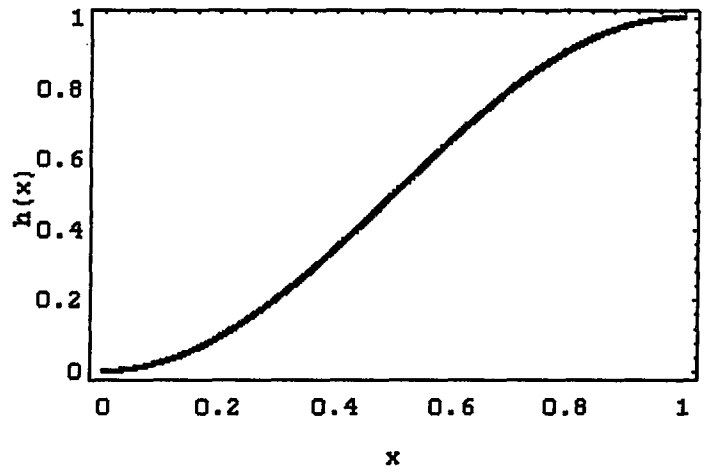
$$y_1 = f(y_0), y_2 = f^2(y_0), \ldots, y_k = f^k(y_0), \ldots \tag{4.33}$$

şeklindedir.

Yukarıdaki iddia, (4.32) eşitliğinde görülen orbit ile (4.33) eşitliğindeki orbitin aynı olduğunu ifade etmektedir. Yalnızca $y_0 = h(x_0)$ değildir. Aynı zamanda,
$y(1) = h(x(1)), y(2) = h(x(2)), \ldots, y(k) = h(h(k)), \ldots$ dır. Yani x_0'ın T altında ürettiği orbit ile x_0'ın dönüşüm uygulanmış değerinin, $y = x_0' = h(x_0)$, f lojistik dönüşümü altında ürettiği orbit aynıdır. Bu denklik, f ve T dönüşümleri yardımıyla, $x \in [0,1]$ koşulunu sağlayan tüm x ler için, şu denklemle verilir,

$$f^k(h(x)) = h(T^k(x)), \quad k = 1,2,\ldots. \tag{4.34}$$

Bu eşdeğerlik, bir örnek ile kontrol edilebilir. $x_0 = 8/25$ ilk koşulunun ilk 9 tekrarlamasının ("iterations") sonucu, Tablo 4.5'te verilmiştir. Çadır dönüşümünün h(x) uygulanmış koordinatları ile lojistik dönüşümün x_0' noktasından başlayan orbiti, tamamen aynıdır.



Şekil 4.23. $\mathbf{h(x) = \sin^2(\pi\, x/2)}$ dönüşümünün grafiği.

Fakat bu sonuç, dikkatle yorumlanmalıdır. En sağdaki iki sütunun pek çok tekrarlama için özdeş olmasını matematiksel tanımlar garantilemesine rağmen; bu teorik sonucun pratikte de sağlanacağı yorumu yapılmamalıdır. Bunun nedeni, fonksiyonun parametrelerine olan duyarlığıdır. $x_0 = 8/25$ sayısı bilgisayarda tam olarak ifade edilmesine rağmen, x_0' sayısı bilgisayarda tam olarak ifade edilemez ve dahası her tekrarlama sonunda ufak hatalar birbirine eklenir. Bunun sonucu olarak değeri makine prezisyonuna bağlı sonlu tekrarlama sayısından sonra, nümerik hesapla bulunan orbit, x_0' değerinin ürettiği gerçek orbitten farklıdır. Yani, bu eşitliğin nümerik hesapla gözlemi, kaosun etkisinden dolayı mümkün değildir.

4.6.1 Çadır Dönüşümü ile Lojistik Dönüşümün Denkliği

Bu bölümde, çadır dönüşümü ile lojistik dönüşümün eşdeğerliğinin arkasındaki matematik sunulacaktır. Böylece bu özdeşliğin sağlandığı, matematiksel olarak tanıtlanacaktır Bunun için başlangıç olarak iyi bilinen iki trigonometrik özdeşlik kullanılacaktır,

$$\cos^2\alpha = 1 - \sin^2\alpha, \qquad \sin^2 2\alpha = 2\sin\alpha\cos\alpha. \tag{4.35}$$

Tablo 4.5. $\mathbf{x_0 = 8/25}$ ilk koşulunun çadır dönüşümü altındaki ilk 9 tekrarlama sonucu, $\mathbf{x'_0}$ değerinin f altındaki değerleri ve bunların, $\mathbf{x'(k) = h(x(k))}$ dönüşüm değerlerinin hesaplanarak, kontrolünün yapılması.

k	$\mathbf{x(k) = T^k(x_0)}$	$\mathbf{x'(k) = \sin^2(\pi\, x(k)/2)}$	$\mathbf{y(k) = f^k(y_0)}$
0	8/25	0.232	0.232
1	16/25	0.713	0.713
2	18/25	0.819	0.819
3	14/25	0.594	0.594
4	22/25	0.965	0.965
5	6/25	0.136	0.136
6	12/25	0.469	0.469
7	24/25	0.996	0.996
8	2/25	0.016	0.016
9	4/25	0.062	0.062

x_0 ilk koşulunun çadır dönüşümü altındaki tekrarları ve dönüşüm uygulanmış $\mathbf{x'_0 = \sin^2(\pi\, x/2)}$ ilk koşulunun, lojistik dönüşüm altındaki tekrarlamaları birbirine eşdeğerdir.

Bu olay cebirsel olarak ifade edilecektir. Çadır dönüşümü için, x_0 başlangıç koşulu; lojistik dönüşüm için ise $\mathbf{y_0 = x'_0}$ başlangıç koşulu kullanılsın. Yani, $\mathbf{x_0, x(1), x(2),...}$, dizisi çadır dönüşümü altındaki orbiti; $\mathbf{x_0, y(1), y(2),...}$, dizisi ise lojistik dönüşüm altındaki orbiti göstersin. Tüm k lar için, k = 0, 1, 2,...,

$$\mathbf{y(k) = x'(k) = h(x(k))} \tag{4.36}$$

koşulunun sağlandığı gösterilirse, denkliğin varlığı tanıtlanmış olur.

$\mathbf{0 \le x_0 \le 1}$ olmak üzere, $\mathbf{y_0 = \sin^2(\pi\, x_0/2)}$ ilk koşulunun lojistik dönüşüm altındaki tekrarlaması göz önüne alınsın,

$$\mathbf{y(1) = 4y_0(1-y_0) = 4\sin^2(\pi\, x_0/2)(1-\sin^2(\pi\, x_0/2))}. \tag{4.37}$$

$\cos^2\alpha = 1 - \sin^2\alpha$ trigonometrik özdeşliği kullanıldığında,

$$y(1) = 4\sin^2(\pi x_0/2)\cos^2(\pi x_0/2) \tag{4.38}$$

ifadesi elde edilir. Bu ifade, $\sin 2\alpha = 2\sin\alpha\cos\alpha$ özdeşliğinin kullanılmasıyla şu şekilde basitleştirilir,

$$y(1) = \sin^2(\pi x_0). \tag{4.39}$$

x_0 ilk koşulunun çadır dönüşümü altındaki ilk tekrarlaması, $x(1) = T(x_0)$ dır. Burada Koordinat değişiminden sonra, y_1 değerinin x_1 değerine özdeş olduğu; yani

$$x'(1) = h(x(1)) = y(1) \tag{4.40}$$

eşitliği gösterilecektir. İlk olarak $0 \le x_0 \le 1/2$ durumu göz önüne alınsın. Bu durumda

$$x(1) = T(x_0) = 2x_0, \tag{4.41}$$

$$x'(1) = \sin^2\left(\frac{\pi x(1)}{2}\right) = \sin^2(\pi x_0) = y(1) \tag{4.42}$$

değerleri bulunur.

İkinci olaraksa $1/2 < x_0 \le 1$ durumu ele alınsın. $x(1) = T(x_0) = 2 - 2x_0$ eşitliği yerine yazıldığında,

$$x'(1) = \sin^2\left(\frac{\pi x(1)}{2}\right) = \sin^2(\pi - \pi x_0) \tag{4.43}$$

bulunur. İfade,

$$\sin^2(\alpha + \pi) = \sin^2(\alpha), \tag{4.44}$$

$$\sin^2(-\alpha) = \sin^2\alpha \tag{4.45}$$

özellikleri kullanılarak, şu şekilde basitleştirilir,

$$\mathbf{x'(1) = \sin^2(-\pi x_0) = \sin^2(\pi x_0) = y(1).} \qquad (4.46)$$

Sonuç, $\mathbf{x'(1) = y(1)}$ olduğunu gösterir. Dolayısıyla tümevarım yöntemi ile tüm k değerleri için $\mathbf{x'(k) = y(k)}$ olduğu sonucu çıkarılır. Yani $\mathbf{x'(k) = h(T^k(x_0))}$ ve $\mathbf{y(k) = f^k(h(x_0))}$ olduğundan, (4.34) eşitliği tanıtlanmış oldu.

Üstteki denklemler yardımıyla lojistik dönüşümün periyodik orbit üreten noktaları kolayca belirlenebilir. Çadır dönüşümü için x_0 periyodik orbit üreten bir nokta ise, bu ilk koşula $\sin^2\left(\frac{\pi x_0}{2}\right)$ dönüşümü uygulmasıyla, lojistik dönüşüm altında periyodik orbit üreten ilk koşul değeri bulunur. Örneğin $x_0 = 2/7$ ilk değeri çadır dönüşümü altında periyodu 3 olan bir orbit üretir, $2/7 \to 4/7 \to 6/7 \to 2/7$. Bu nedenle $\sin^2(\pi/7) = 0.188255099...$ noktası, lojistik dönüşüm içinde periyodu 3 olan bir noktadır.

Yani çadır dönüşümü ile lojistik dönüşümün tekrarlamaları, tümel eşdeğerdir. Bu nedenle kaosun tüm belirtileri, $\mathbf{x \mapsto f(x) = 4x(1-x)}$ lojistik dönüşümde de bulunur.

- Çadır dönüşümü için periyodik orbit üreten noktalar, lojistik dönüşüm için de periyodik orbit üretirler.
- Çadır dönüşümü için topolojik geçişlikik gösteren noktalar, lojistik dönüşüm için de aynı özelliği gösterirler.
- Çadır dönüşümü için duyarlık gösteren noktalar, lojistik dönüşüm içinde aynı özelliği gösterirler.

4.6.1.1 Yoğun Periyodik Orbitler ve Topolojik Geçişlilik

T sembolü çadır dönüşümünü, $\mathbf{x \mapsto f(x) = 4x(1-x)}$ ifadesi lojistik dönüşümü ve $\mathbf{h(x) = \sin^2(\pi x/2)}$ eşitliği ise koordinat değişimini sağlayan dönüşümü belirtmektedir. $\mathbf{x \in [0,1]}$ ve $\mathbf{k = 1,2,...}$ için $\mathbf{f^k(h(x)) = h(T^k(x))}$ eşitliğinin sağlandığı üstte gösterildi. Daha fazlası, lojistik dönüşümünün periyodik noktaları [0,1] aralığında yoğundur ve yine bu dönüşüm topolojik geçişlidir.

(a) Lojistik dönüşümün periyodik noktalarının, [0,1] aralığında yoğun olduğu iddia edilsin. $y \in [0,1]$ olsun. Burada lojistik dönüşümün periyodik noktalarından oluşan ve limiti y olan bir dizinin var olduğu gösterilecektir. x değeri, y'nin h altındaki öngörüntüsü olarak seçilebilir. Yani h örten fonksiyon olduğu için, $h(x) = y$ dir. T'nin periyodik noktaları, [0,1] aralığında yoğun olduğu için, limiti x olan bir $x(1), x(2), \ldots$ dizisi bulunabilir ve dizideki her bir x(k) noktası, T dönüşümü altında p(k) periyotlu orbit üretir. Yani $k = 1,2,\ldots$ için $T^{p(k)}(x(k)) = x(k)$ dır. $y(k) = h(x(k))$ ile tanımlanan $y(1), y(2), \ldots, y(k)$ dizisinin limitinin y olduğu ve bu dizinin lojistik dönüşümün periyodik orbit üreten noktalarından oluştuğu iddia edilecektir. İlk iddia doğrudur. Çünkü h sürekli bir fonksiyondur. İkinci iddianın doğruluğu, $f^k h = hT^k$ eşitliğinden takip edilir. Gerçekten ikinci iddianın da sağlandığı görülmektedir,

$$f^{p(k)}(y(k)) = f^{p(k)}(h(x(k))) = h(T^{p(k)}x(k)) = h(x(k)) = y(k). \tag{4.47}$$

(b) Lojistik dönüşümün topolojik geçişli olduğu iddia edilsin. Bu amaçla U ve V, [0,1] aralığında iki açık aralık olsun. Bu durumda $f^k(y) \in V$ koşulunu sağlayan, bir $y \in U$ noktası ve bir k sayısı bulunmalıdır. Bu nedenle öncelikle U ve V nin öngörüntüleri göz önüne alınmalıdır,

$$\begin{aligned} A &= h^{-1}(U) = \{x \in [0,1] \mid h(x) \in U\}, \\ B &= h^{-1}(V). \end{aligned} \tag{4.48}$$

h sürekli bir fonksiyon olduğundan, A ve B açık aralıklardır. Yani T topolojik geçişli olduğundan, $T^k(x) \in B$ olacak k sayısı ve $x \in A$ sayısı vardır. $y = h(x)$ olarak alınsın. Bu durumda fonksiyonel eşitlik kullanılarak, $f^k(y) = f^k(h(x)) = h(f^k(x))$ elde edilir. Ve $f^k(x) \in B$ olduğundan, $h(f^k(x)) \in h(B) = V$ olduğu sonucuna varılır. Yani lojistik dönüşüm topolojik geçişlidir.

Çadır dönüşümü için yoğun periyodik noktaların varlığı ve T ile f dönüşümlerinin eşdeğerliği, f dönüşümünün de yoğun periyodik orbitlere sahip

olduğu sonucunu verir. T'nin topolojik geçişlilik özelliğine sahip olması ve f ile T arasındaki eşdeğerlik, f'nin de topolojik geçişlilik özelliğine sahip olduğu sonucunu verir. Bu yaklaşım, kaosun üçüncü özelliği olan duyarlık için çalışmaz. Yani ilk koşullara duyarlık, bir dinamik sistemden, koordinat değişimiyle tekrarlamaları bu dinamik sisteme eşit olan başka bir dinamik sisteme taşınmaz. Bunun tersine, karıştırma ve yoğun periyodik noktalar, eşdeğer sisteme geçer.

Bu nedenle, lojistik dönüşümün duyarlık özelliğinin ortaya çıkarılabilmesi için, T'nin duyarlık özelliğine sahip olmasından ve T ile f'nin denkliğinden daha fazla bilgi gerekmektedir.

4.6.1.2 Duyarlık

f fonksiyonunun ilk koşullara duyarlı olduğu iddia edilsin. y, [0,1] aralığında gelişigüzel seçilen bir nokta olsun. Burada limiti y olan $\mathbf{y}(1),\mathbf{y}(2),\ldots$ ilk koşullarıyla başlayan öyle bir dizinin var olduğu gösterilecektirki bu her bir ilk koşula karşılık gelen orbit, y'den en azından belirli bir $\delta_f > 0$ sayısı kadar uzaktır. x sayısı, y'nin h altındaki öngörüntüsü, $\mathbf{h}(\mathbf{x}) = \mathbf{y}$ olarak seçilir. T dönüşümü duyarlı olduğundan, yani $\delta_T > 0$ koşulunu sağlayan bir δ_T sabiti ve T için limiti x olan $\mathbf{x}(1),\mathbf{x}(2),\ldots$ ilk koşullarından oluşan bir dizi vardır. Öyleki bu diziye T altında karşı gelen orbitler, x noktasından en azından δ_T sayısı kadar uzaktır.

Yani bu dizideki her x(k) ilk koşul değeri için, öyle bir n(k) değeri vardırki, **x(k)'nın n(k).** tekrarlama değeri, **x'in n(k).** tekrarlam değerinden en azından δ_T kadar uzaktır,

$$\left|\mathbf{T}^{\mathbf{n(k)}}(\mathbf{x(k)}) - \mathbf{T}^{\mathbf{n(k)}}(\mathbf{x})\right| \geq \delta_T \tag{4.49}$$

$$\delta_f = \left\{\left|\mathbf{h}(\mathbf{x}) - \mathbf{h}(\mathbf{y})\right| \middle| \left|\mathbf{x} - \mathbf{y}\right| \geq \delta_T,\ \ \mathbf{x},\mathbf{y} \in [0,1]\right\} \tag{4.50}$$

ve bu durumda δ_f sayısı tanımlanabilir. $\mathbf{h}(\mathbf{x}) = \sin^2(\pi\,\mathbf{x}/2)$ monotonik olarak arttığı için, δ_f, $\delta_T \leq \mathbf{x} \leq 1$ aralığı için tanımlanan $\mathbf{h}(\mathbf{x}) - \mathbf{h}(\mathbf{x} - \delta_T)$ sürekli fonksiyonunun minimum noktasıdır. Burada ele alınan durumda, $\delta_f = \mathbf{h}(\delta_T) > 0$ olur.

$\mathbf{y(k) = h(x(k))}$ ile tanımlı $\mathbf{y(1), y(2), \ldots}$ dizisi düşünülsün. Yani koordinat değişimi, **T** dönüşümü için ilk koşulları oluşturan diziye uygulanır. **h** dönüşümü sürekli olduğu için, bu dizi bir limite sahiptir,

$$\lim_{k\to\infty} \mathbf{y(k)} = \lim_{k\to\infty} \mathbf{h(x(k))} = \mathbf{h(x)} = \mathbf{y}. \tag{4.51}$$

$\mathbf{f^{n_k} h = h T^{n_k}}$ fonksiyonel denklemi kullanılarak aşağıdaki ifade bulunur,

$$\begin{aligned} \left|\mathbf{f^{n(k)}(y(k)) - f^{n(k)}(y)}\right| &= \left|\mathbf{f^{n(k)}(h(x(k))) - f^{n(k)}(h(x))}\right| \\ &= \left|\mathbf{h(T^{n(k)}(x(k))) - h(T^{n(k)}(x))}\right|. \end{aligned} \tag{4.52}$$

(4.49) eşitsizliği ve δ_f tanımı göz önüne alarak,

$$\left|\mathbf{f^{n(k)}(y(k)) - f^{n(k)}(y)}\right| \geq \delta_f \tag{4.53}$$

elde edilir. Yani, **y(k)** değeri ile başlayan orbit, **n(k)** tekrarlamadan sonra, δ_f sayısına eşit veya ondan daha büyük uzaklığa erişir. $\mathbf{y(k)} \to \mathbf{y}$ olduğundan, **y** noktasına gelişigüzel yakın ve bu özelliğe sahip ilk koşullar böylece bulunmuş olur. Yani, **f** dönüşümü, **y** noktasında duyarlık özelliğine sahiptir.

Duyarlık özelliğini yalnızca lojistik dönüşüm için değil, aynı zamanda benzer durumlar için de ortaya çıkaran iyi bir çözüm, 1992 yılında beş Avustralyalı matematikçinin oluşturduğu bir grup tarafından yayınlanmıştır, [36].

BÖLÜM 5

GARİP ÇEKİCİLER

Günümüzde garip çekiciler, yalnızca fizikçi ve matematikçilerin değil, aynı zamanda doğa bilimcilerin ve hatta sosyal bilimcilerin dikkatini çeken popüler bir konudur. Araştırmacıların bu konuya olan ilgileri, doğada varolan fakat Kepler veya Newton gibi kanunlarla açıklanamayan olayların anlaşılabilmesi için garip çekicilerden umulan beklentilerden dolayıdır, [37, s 656]. Bilim adamları, kuramsal fizikte çözülemeyen en büyük problemlerden biri olan türbülans mekanizması anlayabilmek istedikleri kadar, dünyanın ikliminin nasıl bir sistemle yönetildiğini veya insan beyninde faaliyetlerin ne şekilde cereyan ettiğini de kavrayabilmek isterler.

Doğada varolan ilk garip çekici Lorenz tarafından 1962 yılında keşfedildi ve Lorenz'in basınçlı yitirgen hidrodinamik akışın, deterministik doğrusal olmayan bayağı diferansiyel denklemlerle temsil edilebileceğini ifade ettiği bu çalışması, "Journal of the Atmospher Science" da yayınlandı. Lorenz çekicisi, bilinen garip çekicilerin en eskisi olmasına rağmen, pek çok temel soruyu cevaplayabildiği için, önemini hala korumaktadır.

Chua çekicisi ise, otonom bir devrede kaosun oluşabilmesi için gerekli kriterleri içeren en basit elektronik devredir. Bununla beraber, kaosun varlığının matematiksel olarak tanıtlandığı tek fiziksel sistem olan bu devre, düşük fiyatlı devre parçalarıyla inşa edilebilir, [38, s 657].

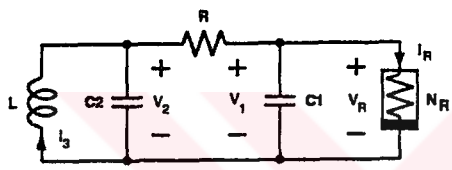
Chua ve Lorenz çekicisi söylendikleri sıra ile bu bölümde ele alınacaktır.

5.1 Chua Çekicisi

Kapasite, direnç ve kondansatörden oluşan otonom bir devrenin kaos sergileyebilmesi için,

1) En azından doğrusal olmayan bir eleman,
2) En azından yerel olarak aktif bir direnç ve
3) En azından üç tane enerji depolayan eleman

içermelidir. Chua devresi, bu kriterleri sağlayan en basit elektronik devredir.



Şekil 5.1. Doğrusal bir self (L), doğrusal iki kapasite (C_1,C_2), doğrusal bir direnç (R) ve gerilim kontrollü doğrusal olmayan bir dirençten (N_R) oluşan Chua devresi.

Chua devresinin oldukça zengin bir dinamiğe sahip olduğu, bilgisayar simülasyonu ile, [39] ve deney ile, [40] doğrulandı. Bu çalışmalardan sonra, devrenin dinamiklerinin tümünü anlamaya ilişkin oldukça yoğun çabalar ve çalışmalar oluştu ve bu devreye, doğrusal olmayan dinamikleri ve kaosu öğrenmek, anlamak ve öğretmek için bir örnek gözüyle bakılmaya başlandı. Bu devrenin, Shilnikov'un bakış açısıyla kaos sergilediği; 1986 yılında Chua *et al* tarafından tanıtlandı, [41].

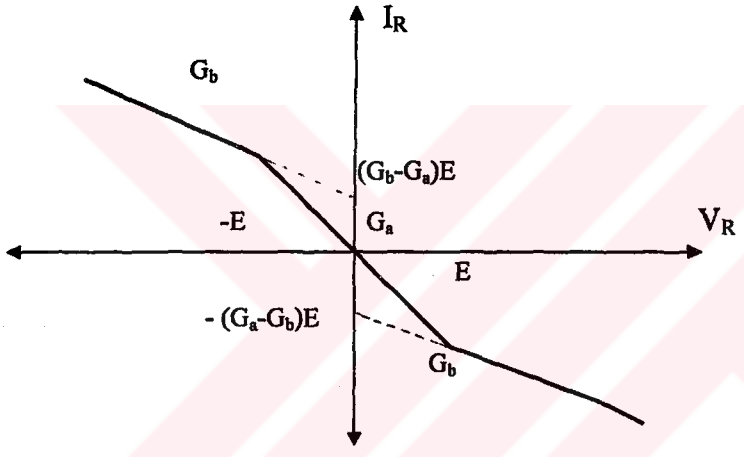
Şekil 5.1'de verilen Chua devresi için, I_3, V_2 ve V_1 durum değişkenleri olarak seçildiğinde, $G = 1/R$, $G_a' = G + G_a$ ve $G_b' = G + G_b$ olmak üzere, aşağıdaki durum denklemleri yazılır.

$$\frac{dI_3}{dt} = -\frac{1}{L}V_2 \tag{5.1}$$

$$\frac{dV_2}{dt} = \frac{1}{C_2}I_3 - \frac{G}{C_2}(V_2 - V_1) \tag{5.2}$$

$$\frac{dV_1}{dt}=\frac{G}{C_1}(V_2-V_1)-\frac{1}{C_1}f(V_1)=\begin{cases}\dfrac{G}{C_1}V_2-\dfrac{G'_b}{C_1}V_1-\left(\dfrac{G_b-G_a}{C_1}\right)E, & V_1<-E\\ \dfrac{G}{C_1}V_2-\dfrac{G'_a}{C_1}V_1, & -E\le V_1\le E\\ \dfrac{G}{C_1}V_2-\dfrac{G'_b}{C_1}V_1-\left(\dfrac{G_b-G_a}{C_1}\right)E, & V_1>-E\end{cases} \quad (5.3)$$

N_R'nin parçalı doğrusal yapısından dolayı, Chua devresinin vektör alanı, üç ayrı bölgeye ayrılır: $V_1<-E$, $|V_1|\le -E$ ve $V_1>E$.



Şekil 5.2. Doğrusal olmayan N_R direncinin karakteristiği, $\pm E$ kırılma noktalarına ve iç ve dış bölgelerdeki G_a ve G_b eğimlerine sahiptir.

Şimdiye kadar Chua devresi, L, C_2, G, C_1, E, G_a ve G_b parametrelerine göre tanımlandı. Doğrusal olmayan direncin normalize edilmesiyle parametre sayısı azaltılabilir. Yani doğrusal olmayan direncin kırılma noktaları olan $\pm E$ yerine $\pm 1V$ alınarak normalize işlemi gerçekleştirilir. Daha fazlası, (5.1) - (5.3) denklemleri, aşağıda verilen değişken değişimleri yardımıyla boyutsuz olarak normalize edilebilir: $x=V_1/E$, $y=V_2/E$, $z=I_3/(EG)$ ve $\tau=tG/C_2$. Bu kabuller ışığında (5.4) - (5.6) denklemleri elde edilir.

$$\frac{dz}{d\tau}=-\beta y \quad (5.4)$$

$$\frac{dy}{d\tau}=x-y+z \quad (5.5)$$

$$\frac{dx}{d\tau}=\alpha\left(y-h(x)\right)$$

$$=\begin{cases}\alpha\left(y-bx-(b-a)\right) & x<-1\\ \alpha\left(y-ax\right) & -1\le x\le 1\\ \alpha\left(y-bx-(a-b)\right) & x>1\end{cases} \qquad (5.6)$$

Denklemlerde, $a=G_a'/G=1+G_a/G$, $b=G_b'/G=1+G_b/G$, $\alpha=C_2/C_1$ ve $\beta=C_2/(LG^2)$ olarak verilirler. Yani 7 parametreye sahip olan Chua devresi, $\{a,b,\alpha,\beta\}$ boyutsuz parametreleriyle temsil edilebildi. Chua diyodunun (doğrusal olmayan direnç) G_a, G_b eğimlerine karşı gelen a ve b değerleri sabit tutulursa; Chua devresinin kalıcı durum dinamikleri, iki boyutlu $\alpha-\beta$ diyagramında özetlenebilir. $\alpha-\beta$ diyagramı için, a ve b, $a=-1/7$ ve $b=2/7$ değerlerinde sabit tutuldu.

5.1.1 Chua Devresinin Denge Noktaları

$x<-1$ **için:**

$$\begin{aligned}-\beta y&=0\\ x-y+z&=0\\ \alpha(y-bx-(b-a))&=0\end{aligned} \qquad (5.7)$$

denklemlerinin çözümü, $\left(\frac{a-b}{b},0,-\frac{(a-b)}{b}\right)$ denge noktasını verir.

$-1\le x\le 1$ **için:**

$$\begin{aligned}-\beta y&=0\\ x-y+z&=0\\ \alpha(y-ax)&=0\end{aligned} \qquad (5.8)$$

denklemlerinin çözümü, $(0,0,0)$ denge noktasını verir.

$x>1$ **için:**

$$\begin{aligned}-\beta y&=0\\ x-y+z&=0\\ \alpha(y-bx-(a-b))&=0\end{aligned} \qquad (5.9)$$

denklemlerinin çözümü, $\left(\frac{-a+b}{b},0,-\frac{(-a+b)}{b}\right)$ denge noktasını verir.

5.1.2 Denge Noktalarının Kararlılığı

Denge noktalarının kararlılığı,

$$\frac{dz}{d\tau} = -\beta y = f_3(x,y,z) \tag{5.10}$$

$$\frac{dy}{d\tau} = x - y + z = f_2(x,y,z) \tag{5.11}$$

$$\frac{dx}{d\tau} = \alpha(y - h(x)) = f_1(x,y,z) \tag{5.12}$$

denklemlerinin, kararlılığı incelenecek olan denge noktalarında doğrusallaştırılması sonucu elde edilen D_F matrislerinin özdeğerleri yani karakteristik polinomun sıfırları tarafından belirlenir.

Routh - Hurwitz kriterine göre, denge noktalarının kararlılığı hakkında bilgi, karakteristik polinomun katsayıları tarafından belirlenir. Bu kriterin ışığında oluşturulan tablonun ilk sütunundaki tüm terimlerin pozitif olması, karakteristik polinomun tüm sıfırlarının (köklerinin) sol yarı düzlemde olması için gerekli ve yeterli bir koşuldur. Dahası, ilk sütundaki katsayıların işaret degiştirme sayısı, pozitif gerçel kısımlı köklerin sayısını verir.

$\left(\frac{a-b}{b},0,-\frac{(a-b)}{b}\right)$ ve $\left(\frac{-a+b}{b},0,-\frac{(-a+b)}{b}\right)$ denge noktalarının kararlılığı:

Bu denge noktaları simetrik olduğu için, birinin kararlılığının incelenmesi yeterlidir.

$$\begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} & \frac{\partial f_1}{\partial z} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} & \frac{\partial f_2}{\partial z} \\ \frac{\partial f_3}{\partial x} & \frac{\partial f_3}{\partial y} & \frac{\partial f_3}{\partial z} \end{bmatrix} = D_{F_a} = \begin{bmatrix} -\alpha b & \alpha & 0 \\ 1 & -1 & 1 \\ 0 & -\beta & 0 \end{bmatrix} \tag{5.13}$$

matrisinin karakteristik polinomu,

$$s^3+s^2(1+\alpha b)+s(\beta+\alpha b-\alpha)+\alpha b\beta \tag{5.14}$$

şeklindedir. Klasik kararlılık kuramının Routh - Hurwitz teoremi, bu denge noktalarının kararlılığı için gerekli ve yeterli koşulları verir. Bu teorem için gerekli tablo, karakteristik polinomun dikkate alınmasıyla şu şekilde oluşturulur:

$$\begin{array}{lcc} s^3 & 1 & \beta+\alpha b-\alpha \\ s^2 & 1+\alpha\beta & \alpha b\beta \\ s^1 & \dfrac{(1+\alpha b)(\beta+\alpha b-\alpha)-\alpha b\beta}{(1+\alpha\beta)} & 0 \\ s^0 & \alpha b\beta & 0 \end{array} . \tag{5.15}$$

Denge noktalarının kararlı olması için,

$$1+\alpha\beta>0, \tag{5.16}$$

$$(1+\alpha b)(\beta+\alpha b-\alpha)>\alpha b\beta \tag{5.17}$$

ve

$$\alpha b\beta>0 \tag{5.18}$$

eşitsizliklerinin sağlanması gerekir.

$(0,0,0)$ denge noktasının kararlılığı:

Bu denge noktasının kararlılığı için,

$$\begin{bmatrix} -\alpha a & \alpha & 0 \\ 1 & -1 & 1 \\ 0 & -\beta & 0 \end{bmatrix} \tag{5.19}$$

matrisinin, $s^3+s^2(1+\alpha a)+s(\beta+\alpha a-\alpha)+\alpha a\beta$ karakteristik polinomunun tüm kökleri açık sol yarı düzlemde olmalıdır. Bunun için gerekli ve yeterli koşullar, yine Routh - Hurwitz teoremi kullanılarak belirlenebilir.

$$\begin{array}{lll} s^3 & 1 & \beta+a\alpha-\alpha \\ s^2 & 1+a\alpha & \alpha a\beta \\ s^1 & \dfrac{(1+\alpha a)(\beta+a\alpha-\alpha)-\alpha a\beta}{1+\alpha a} & 0 \\ s & \alpha a\beta & 0 \end{array} \qquad (5.20)$$

düzenlenen bu tabloya göre denge noktasının kararlılığı için,

$$1+a\alpha>0, \qquad (5.21)$$

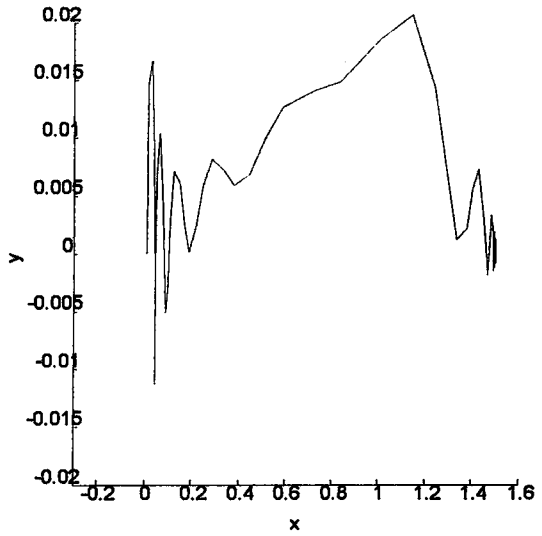
$$(1+\alpha a)(\beta+a\alpha-\alpha)>\alpha a\beta \qquad (5.22)$$

ve

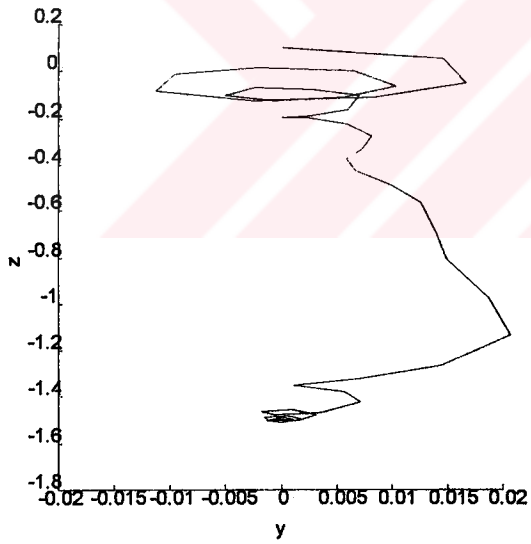
$$\alpha a\beta>0 \qquad (5.23)$$

koşulları sağlanmalıdır.

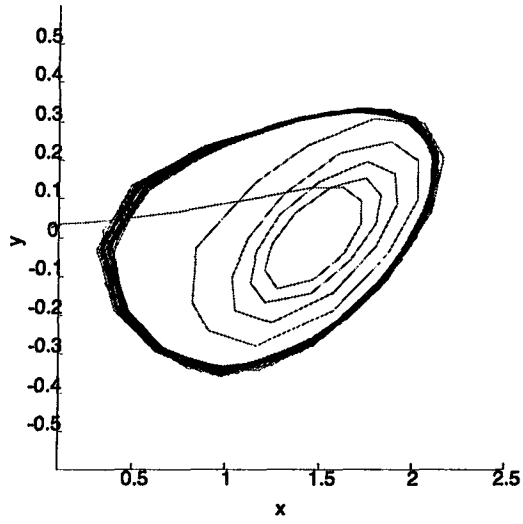
Chua devresi ile ilgili verilecek tüm şekiller için, $a=-\frac{1}{7}$ ve $b=\frac{2}{7}$ olarak alındı. Bu durumda sistemin denge noktaları, $(-3/2,0,3/2)$, $(0,0,0)$ ve $(3/2,0,-3/2)$ olarak elde edilir.



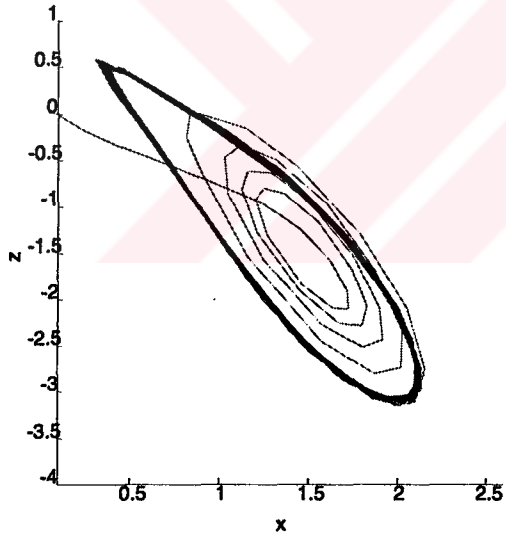
Şekil 5.3. $\alpha = 4$, $\beta = 33$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x - y düzleminde (1.5, 0, -1.5) denge noktasına yakınsaması.



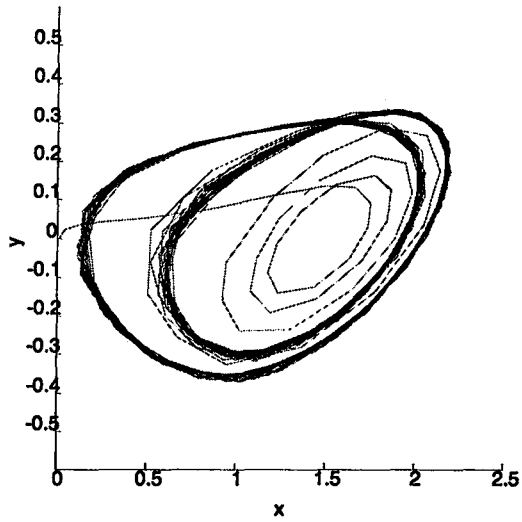
Şekil 5.4. $\alpha = 4$, $\beta = 33$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin y - z düzleminde (1.5, 0, -1.5) denge noktasına yakınsaması.



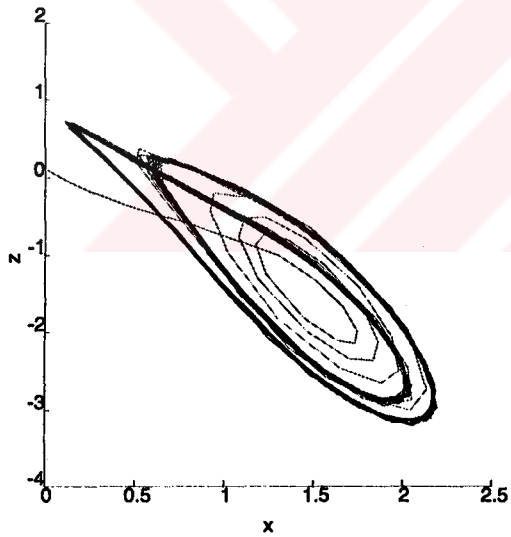
Şekil 5.5. $\alpha = 9,\ \beta = 16.5$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x - y düzlemindeki davranışı.



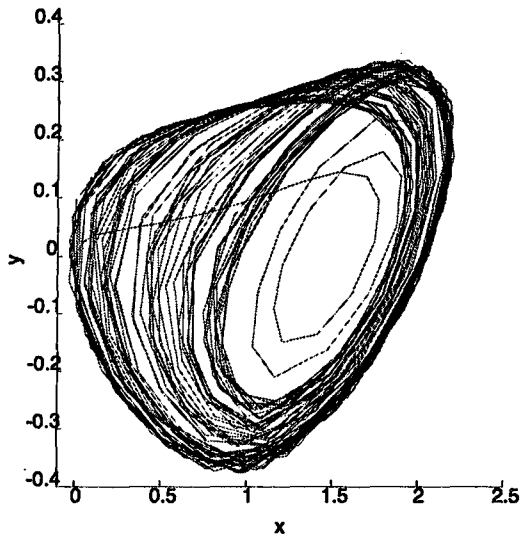
Şekil 5.6. $\alpha = 9,\ \beta = 16.5$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x -z düzlemindeki davranışı.



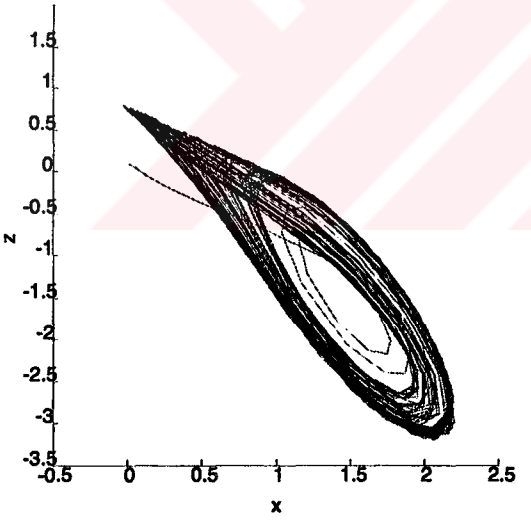
Şekil 5.7. $\alpha = 9$, $\beta = 16$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x - y düzlemindeki davranışı.



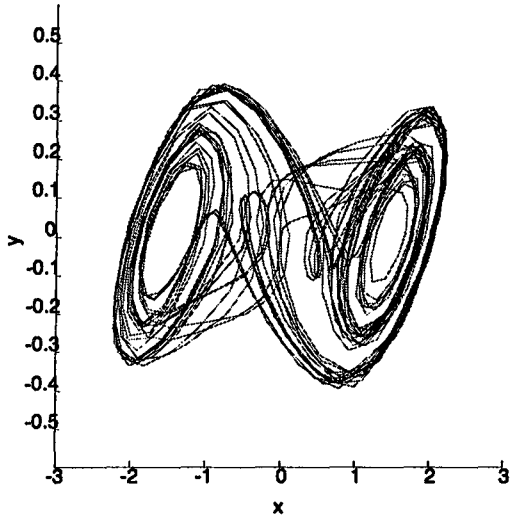
Şekil 5.8. $\alpha = 9$, $\beta = 16$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x -z düzlemindeki davranışı.



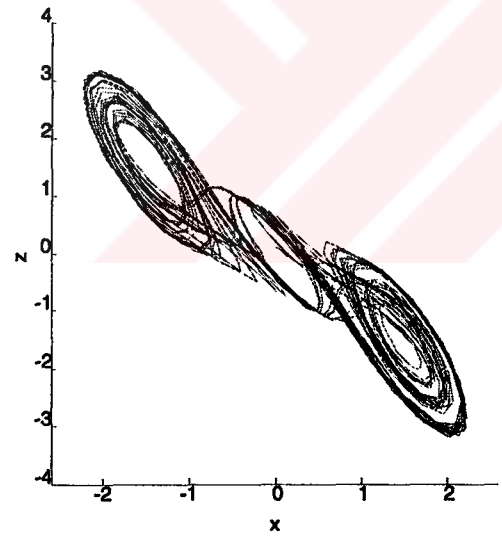
Şekil 5.9. $\alpha = 9, \beta = 15.5$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x - y düzlemindeki davranışı.



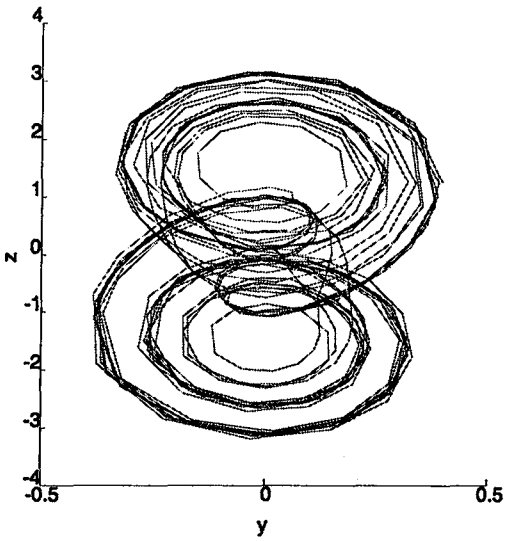
Şekil 5.10. $\alpha = 9, \beta = 15.5$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x - z düzlemindeki davranışı.



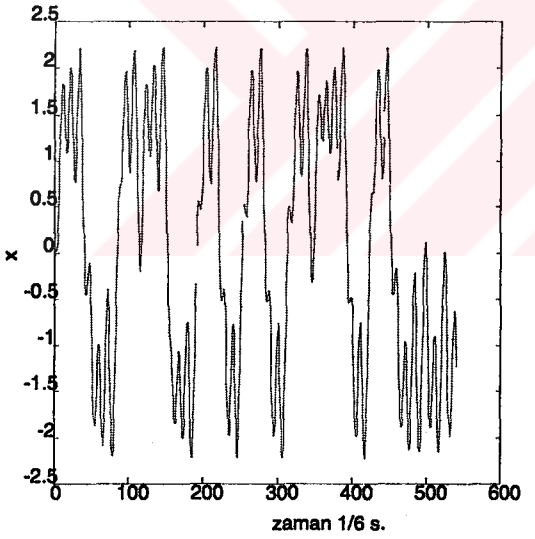
Şekil 5.11. $\alpha = 9, \beta = 100/7$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x -y düzlemindeki davranışı.



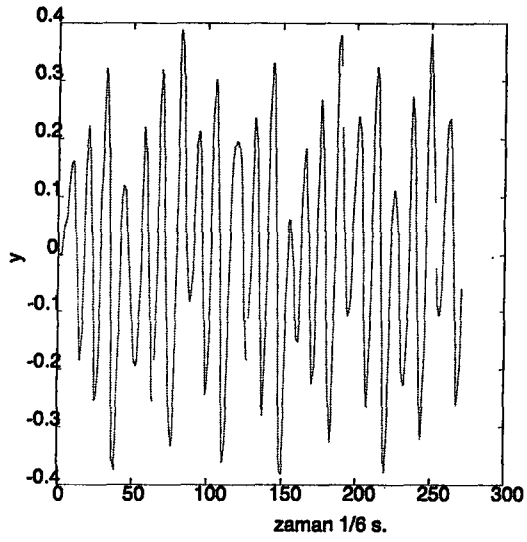
Şekil 5.12. $\alpha = 9, \beta = 100/7$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için sistemin x - z düzlemindeki davranışı.



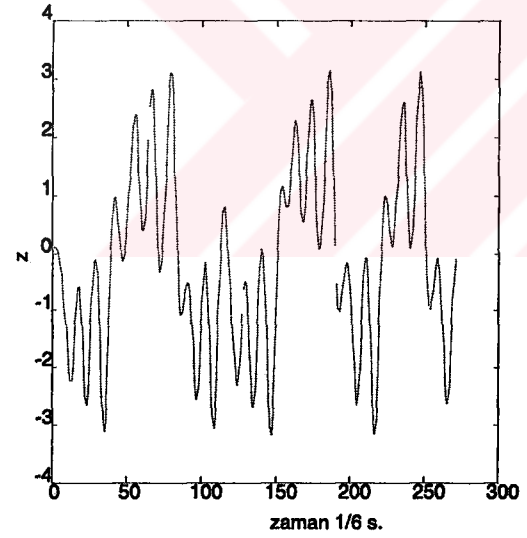
Şekil 5.13. $\alpha = 9, \beta = 100/7$ ve $(\mathbf{x}_0, \mathbf{y}_0, \mathbf{z}_0) = (0.01,0,0.1)$ değerleri için sistemin y -z düzlemindeki davranışı.



Şekil 5.14. $\alpha = 9, \beta = 100/7$ ve $(\mathbf{x}_0, \mathbf{y}_0, \mathbf{z}_0) = (0.01,0,0.1)$ değerleri için x(t)'nin grafiği.



Şekil 5.15. $\alpha = 9, \beta = 100/7$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için y(t)'nin grafiği.



Şekil 5.16. $\alpha = 9, \beta = 100/7$ ve $(x_0, y_0, z_0) = (0.01, 0, 0.1)$ değerleri için z(t)'nin grafiği.

Bölüm 3 ve Bölüm 4'de kaotik sistemlerin ilk koşullara aşırı duyarlık gösterdiği belirtildi. Chua devresinin simülasyonu sonucu bulunan nümerik çözüm, (5.1) - (5.4) denklemlerinin gerçek çözümü değildir, [40]. Bu nedenle sistemin kaotik olduğu iddiası, matematiksel tanıtlar yardımıyla desteklenebilmelidir.

Sürekli zamanlı sistemlerde, kaosun varlığını tanıtlamak oldukça zordur. Buna rağmen Chua devresi için böyle bir tanıt, Chua *et al* tarafından verildi, [41]. Bu çalışmadan elde edilen sonuç, [38]'de şu şekilde verilmiştir.

Teorem (Chua devresinde matematiksel kaos) (α, β) parametre uzayında öyle bir bölge vardırki, bu bölgede Chua çekicisinin hareketi, yazı - tura atmanın matematiksel modeli (Bernouilli ötelemesi) ile üretilir.

5.2 Lorenz Çekicisi

Akışkanlar mekaniği ile ilgili kısmi türevli Napier - Stokes denklemlerinden türetilen Lorenz denklemleri, Saltzman denklemlerinin basit konveksiyon için basitleştirilmişidir, [42]. İki boyutlu akışkan hücre alttan ısıtıldığında ve üstten soğutulduğunda oluşan konveksiyon hareketini tanımlayan Lorenz denklemleri,

$$\begin{aligned} \dot{x} &= -\sigma x + \sigma y \\ \dot{y} &= -xz + rx - y \\ \dot{z} &= xy - bz \end{aligned} \qquad (5.24)$$

şeklinde verilir.

Alt ve üst yüzeyler arasındaki sıcaklık farkı sabit olduğunda, düzgün derinlikteki akışkan yüzeyinde meydana gelen akış, Rayleigh tarafından incelendi,[43]. Bu sistem, eğer sıcaklık derinlikle doğrusal olarak değişirse; konveksiyon içermeyen kalıcı çözümüne sahiptir.

Bu çözüm kararsızsa, konveksiyon oluşur. x değişkeni, konveksiyon hareketinin şiddetiyle orantılıdır. y ise artan ve azalan akımlar arasındaki sıcaklık farkı ile orantılıdır. x ve y'nin benzer işaretleri, ısınan akışkanın yükseldiğini ve soğuyan akışkanın indiğini gösterir. z değişkeni, düşeydeki sıcaklık değişimini ölçer. Bu değişkenin pozitif değeri, en büyük sapmanın sınırlar civarında meydana geldiğini gösterir. Prandtl sayısı olarak bilinen boyutsuz olan σ parametresi, kinematik viskozitenin, termik iletkenliğe oranıdır. r parametresi, Rayleigh sayısı ile orantılı sıcaklık gradyanıdır. $b = 4(1+a^2)^{-1}$, incelenen bölge ile orantılı geometrik bir çarpandır. σ, r ve b parametreleri, pozitif değerli olarak alınırlar.

Sistemin kararlılığının belirlenmesi için, denge noktalarının özel bir öneme sahip olduğu, Bölüm 2'de incelendi. Sistemin denge noktaları,

$$\begin{aligned} -\sigma x+\sigma y&=0\\ -xz+rx-y&=0\\ xy-bz&=0 \end{aligned} \tag{5.25}$$

denklemlerinin çözümleridir.

Lorenz denklemlerinin,

$$(0,0,0), \tag{5.26}$$

$$(-\sqrt{b(-1+r)},-\sqrt{b(-1+r)},(-1+r)) \tag{5.27}$$

ve

$$(\sqrt{b(-1+r)},\sqrt{b(-1+r)},(-1+r)) \tag{5.28}$$

olmak üzere üç denge noktası vardır. Bir denge noktasının kararlılığı, sistemin kararlılığı incelenecek olan denge noktasında doğrusallaştırılması sonucu elde edilen matrisin özdeğerleri, yani matrisin karakteristik polinomunun sıfırları (kökleri) tarafından belirlenir.

(0,0,0) denge noktasının kararlılığı,

$$\begin{aligned} \dot{x}&=-\sigma x+\sigma y=f_1(x,y,z)\\ \dot{y}&=-xz+rx-y=f_2(x,y,z)\\ \dot{z}&=xy-bz=f_3(x,y,z) \end{aligned} \tag{5.29}$$

olmak üzere,

$$\begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} & \frac{\partial f_1}{\partial z} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} & \frac{\partial f_2}{\partial z} \\ \frac{\partial f_3}{\partial x} & \frac{\partial f_3}{\partial y} & \frac{\partial f_3}{\partial z} \end{bmatrix} = \begin{bmatrix} -\sigma & \sigma & 0 \\ -z+r & -1 & -x \\ y & x & -b \end{bmatrix} \tag{5.30}$$

matrisinde bu denge noktasının yazılması sonucu, $(x,y,z)=(0,0,0)$, elde edilen

$$\begin{bmatrix} -\sigma & \sigma & 0 \\ r & -1 & 0 \\ 0 & 0 & -b \end{bmatrix} \tag{5.31}$$

matrisinin özdeğerleri, yani

$$(s+b)[s^2+(\sigma+1)s-\sigma(-1+r)] \tag{5.32}$$

karakteristik polinomun sıfırları (kökleri) tarafından belirlenir. Basitlik açısından bu adımdan sonra, $\lambda=(-1+r)$ alındı.

$$(s+b)[s^2+(\sigma+1)s-\sigma\lambda]=0 \tag{5.33}$$

karakteristik denkleminde tüm kökler gerçeldir ve $\Delta=(\sigma+1)^2+4\sigma\lambda$ olmak üzere aşağıdaki gibi verilirler,

$$s_1=-b \tag{5.34}$$

$$s_2=\frac{-(\sigma+1)+\sqrt{\Delta}}{2} \tag{5.35}$$

$$s_3=\frac{-(\sigma+1)-\sqrt{\Delta}}{2}. \tag{5.36}$$

Bu özdeğerlerden herhangibiri pozitif olduğunda, (0,0,0) dengenoktası, kararsız olacaktır.

Modelin simetrisinden dolayı, diğer iki denge noktası özdeş özelliklere sahiptir. Bu nedenle yalnızca birinin detaylı çalışılması yeterlidir. Bunlardan pozitif koordinatlısı ele alınsın,

$$\left(\sqrt{b\lambda},\sqrt{b\lambda},(-1+r)\right). \tag{5.37}$$

Sistemin bu denge noktasında doğrusallaştırılmış modeli,

$$\begin{bmatrix} -\sigma & \sigma & 0 \\ 1 & -1 & -\sqrt{b\lambda} \\ \sqrt{b\lambda} & \sqrt{b\lambda} & -b \end{bmatrix} \tag{5.38}$$

matrisi ile verilir. Bu matrisin karakteristik polinomu,

$$s^3+(\sigma+b+1)s^2+b(\sigma+\lambda+1)s+2\sigma b\lambda \tag{5.39}$$

olarak elde edilir. Routh - Hurwitz teoremine göre,

$$\begin{array}{lcc} s^3 & 1 & b(\sigma+\lambda+1) \\ s^2 & (\sigma+b+1) & 2\sigma b\lambda \\ s^1 & \dfrac{(\sigma+b+1)b(\sigma+\lambda+1)-2\sigma b\lambda}{(\sigma+b+1)} & 0 \\ s^0 & 2\sigma b\lambda & 0 \end{array} \tag{5.40}$$

$$(\sigma+b+1)b(\sigma+\lambda+1)>2\sigma b\lambda \tag{5.41}$$

kararlılık için gerekli ve yeterli bir koşuldur. Bu ifade,

$$\lambda(b+1-\sigma)+(\sigma+1)(\sigma+b+1)>0 \tag{5.42}$$

şeklinde yeniden yazılabilir. Yani bu koşul sağlandığında orjinden uzakla olan her iki denge noktası asimptotik kararlıdır. Diğer taraftan,

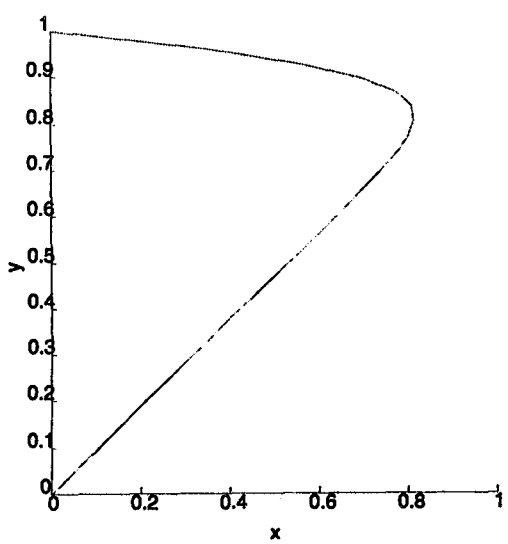
$$\sigma > b+1 \tag{5.43}$$

$$\lambda > \frac{(\sigma+1)(\sigma+b+1)}{(\sigma-b-1)} \quad \text{veya} \quad r > \frac{\sigma(\sigma+b+3)}{(\sigma-b-1)} \tag{5.44}$$

olduğunda tüm denge noktaları kararsızdır. Bu koşullar altında sistem oldukça karmaşık bir davranış sergiler. Denge noktalarından birine yakın başlayan yörünge, bazı anlar için dışarıya doğru sarmal şekilde hareket eder ve daha sonra zıt denge noktasına gider ve benzer hareketi bu denge noktası etrafında icra eder. Hareket bu şekilde devam eder. Bu özel örnek için, simülasyon çalışmaları parametrelerin belli değer bölgesinde, denge noktaları için asimptotik kararlılık koşulu sağlansa bile, garip çekicinin varolabileceğini göstermiştir, [44].

5.2.1 Sistemin Kalıcı Durum Davranışları

$0 < r < 1$ olduğunda, Lorenz denklemleri tek bir kalıcı durum çözümüne sahiptir, $(x,y,z) = (0,0,0)$. Bu çözüm, hiçbir konveksiyon olmadığının ve herhangibir çözümün bu noktaya yakınsadığını ifade eder. Şekil 5.17 ve Şekil 5.18, $(x_0,y_0,z_0) = (0.0,1.0,0.0)$, $\sigma = 10.0$, $r = 0.5$ ve $b = 8.0/3.0$ değerleri için bu durumu gösterir.

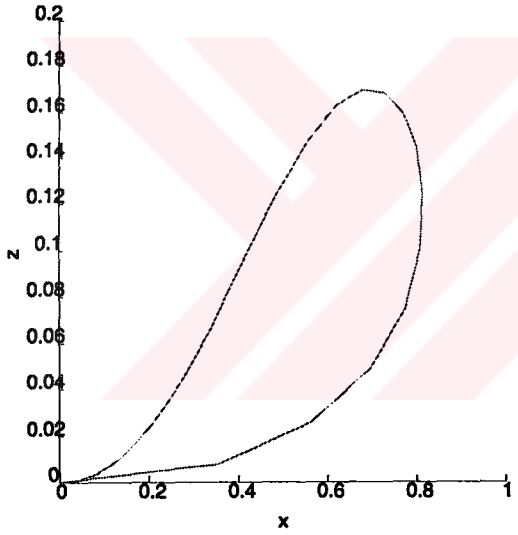


Şekil 5.17. r = 0.5, b = 8.0/3.0 ve σ = 10 parametre değerleri için tüm ilk koşulların, (0,0,0) denge noktasına yakınsaması.

Konveksiyon ilk kez r = 1 olduğunda oluşur. r > 1 olduğunda, Lorenz denklemleri (0,0,0) kalıcı - durum çözümüne ek olarak iki simetrik kalıcı - durum çözümüne daha sahiptir:

$$(\mathbf{x},\mathbf{y},\mathbf{z}) = \left(+\sqrt{\mathbf{b}(\mathbf{r}-1)}, +\sqrt{\mathbf{b}(\mathbf{r}-1)}, \mathbf{r}-1\right)$$
$$(\mathbf{x},\mathbf{y},\mathbf{z}) = \left(-\sqrt{\mathbf{b}(\mathbf{r}-1)}, -\sqrt{\mathbf{b}(\mathbf{r}-1)}, \mathbf{r}-1\right). \qquad (5.45)$$

Şekil 5.19 ve Şekil 5.20, $(\mathbf{x},\mathbf{y},\mathbf{z}) = (0.0,1.0,0.0), \sigma = 10.0$, $\mathbf{r} = 15.0$ ve $\mathbf{b} = 8.0/3.0$ değerleri için sistemin, bu simetrik kararlı noktalardan birine yakınsadığını gösterir.

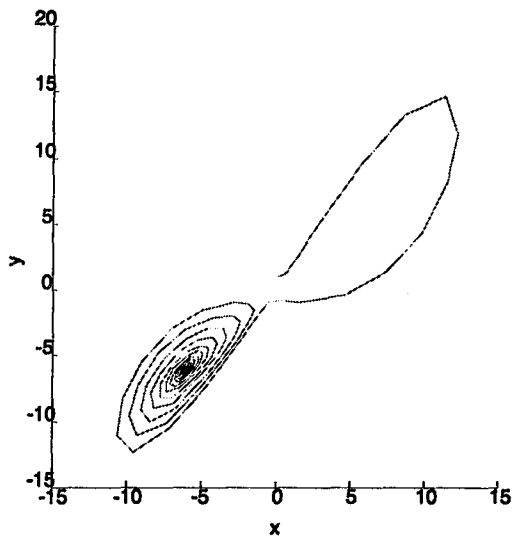


Şekil 5.18. r = 0.5 için, Lorenz sistemi (0,0,0) denge noktasına yakınsaması.

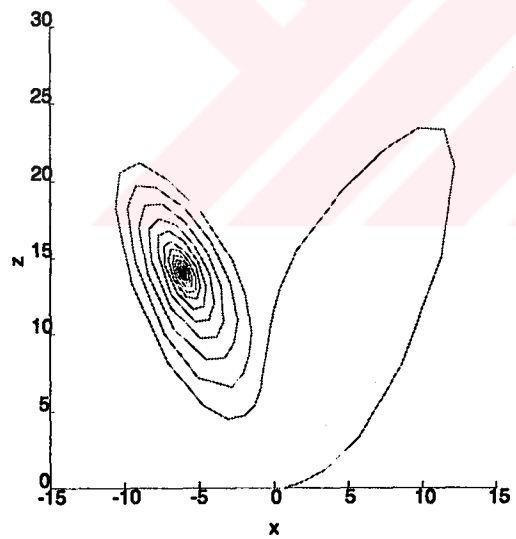
r, r_c ile gösterilen r'nin kritik değerinden büyük olduğunda kalıcı - durum konveksiyonunda kararsızlık başlar,

$$\mathbf{r}_c = \frac{\sigma(\sigma+\mathbf{b}+3)}{(\sigma-\mathbf{b}-1)}, \quad \mathbf{r} > 1. \qquad (5.46)$$

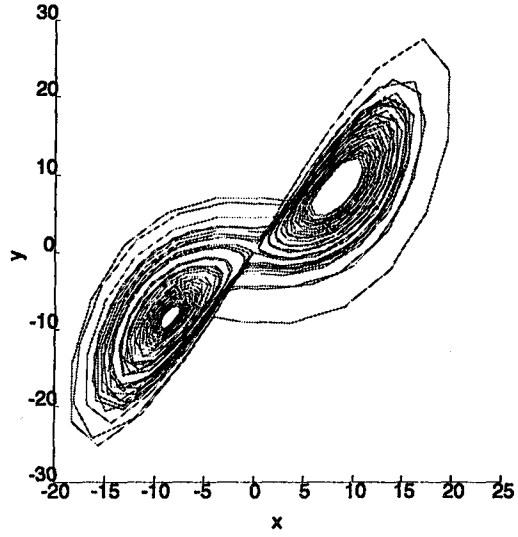
Eğer $(\sigma-\mathbf{b}-1) > 0$ ise, $\mathbf{r} > \mathbf{r}_c$ için kalıcı konveksiyon kararsızdır ve sonuç kaotiktir. Şekil 5.21 ve Şekil 5.22, $(\mathbf{x}_0,\mathbf{y}_0,\mathbf{z}_0) = .(0.0,1.0,0.0)$, $\sigma = 10.0$, $\mathbf{r} = 28.0$ ve $\mathbf{b} = 8.0/3.0$ değerleri için, x - y ve x - z düzlemindeki Lorenz çekicisini gösterir.



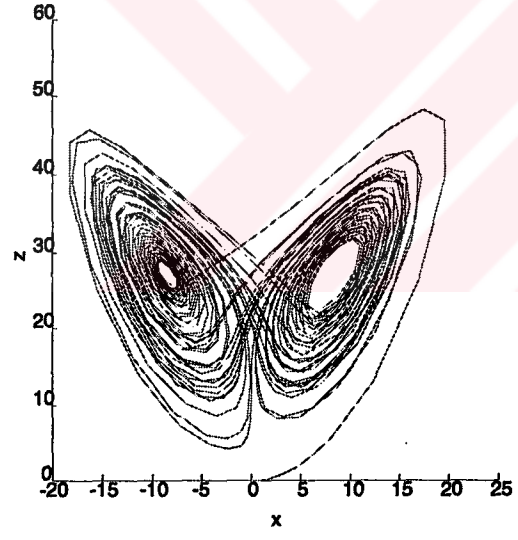
Şekil 5.19. Sistemin, r = 15.0 için simetrik kararlı denge noktalarından birine yakınsaması.



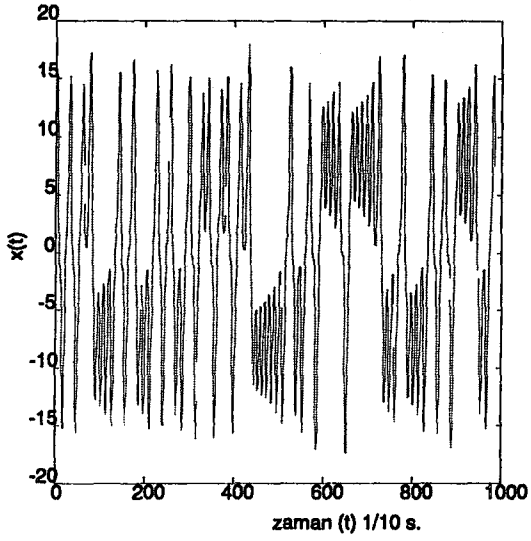
Şekil 5.20. r =5 parametre değeri için Lorenz sisteminin x - z düzlemindeki kalıcı - durum davranışı.



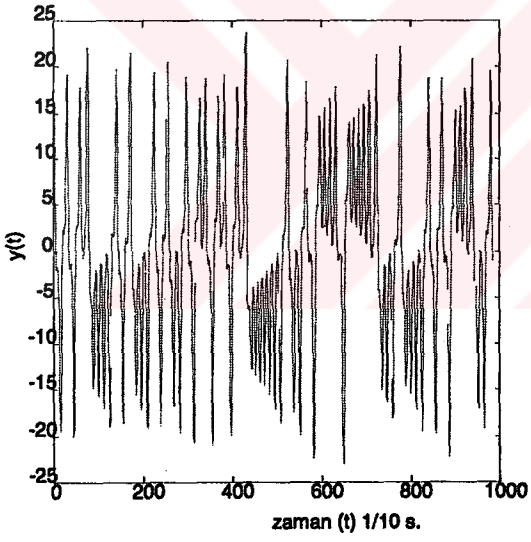
Şekil 5.21. r = 28 için, x - y düzlemindeki Lorenz çekicisi.



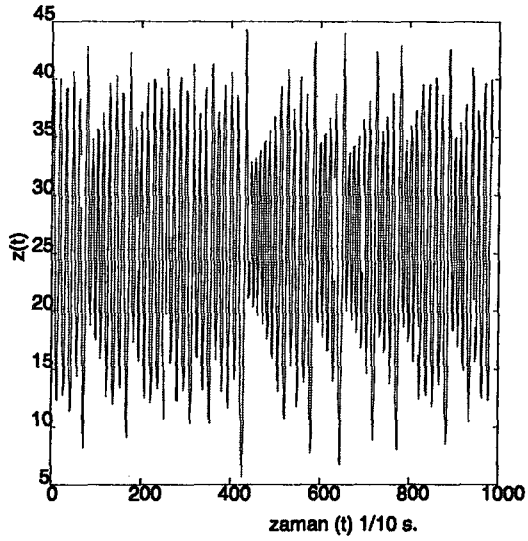
Şekil 5.22. r = 28 paremetre değeri için, x - z düzlemindeki Lorenz çekicisi.



Şekil 5.23. r = 28 paremetre değeri için x değişkeninin zamana bağlı grafiği.



Şekil 5.24. r = 28 paremetre değeri için y değişkeninin zamana bağlı grafiği.



Şekil 5.25. r = 28 paremetre değeri için z değişkeninin zamana bağlı grafiği.

BÖLÜM 6

KRİPTOGRAFİ VE SANKİ - RASGELE DİZİ ÜRETEÇLERİ

Bu bölüm özel bir bilginin izinsiz girişlere karşı korunması anlamına gelen Kriptografi altbölümü ile başlayacaktır. Bu altbölümde, kriptografi, şifreli mesaj, şifreleme, şifre çözme ve kriptoanaliz tanımları sunulduktan sonra, bilgilerin şifrelenerek anlaşılmaz şekle getirildiği kriptosistemler üzerinde durulacaktır. Bu altbölüm tek - zamanlı sistemlerin ("one - time pad") ve bunların kullanılmasındaki güçlüklerin açıklanması ile son bulacaktır. Bu sistemlerin kullanım güçlüklerinden dolayı, tek - zamanlı sistemlerin yaklaşık uyarlamaları şeklinde tasarlanan algoritmalarda, sanki - rasgele dizilere oldukça fazla ihtiyaç vardır. Bu nedenle bu altbölümün ardından, kriptografide kullanılan sanki - rasgele dizi üreteçleri anlatılacaktır.

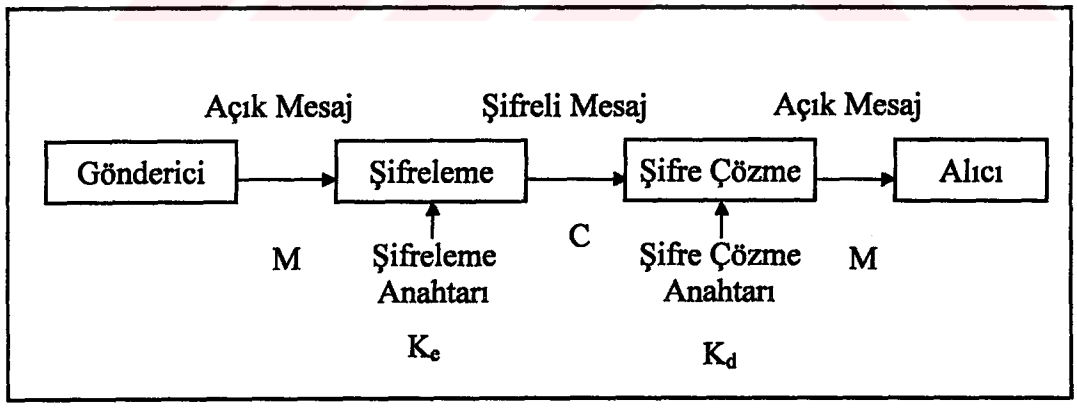
Kriptografinin temel kabullerinden biri, kriptografik algoritmaların gizli tutulamayacağıdır. Bu sebeple algoritmanın güvenliği, kullanılan anahtar dizisine bağlıdır. Bir anahtar dizisi, kullanım süresi boyunca gizli tutulmalı ve korunan bilgi önemini yitirene kadar kriptoanalize karşı dayanıklı olmalıdır. Anahtar dizisinin kriptoanalize karşı dayanıklı olmasının sağlamanın bir yolu; dizinin, mümkün olduğunca rasgele görünüşlü olmasının garantilemektir. Bu nedenle deterministik olarak üretilen anahtar dizilerinin, kriptosistemlerde kullanılmasından önce istatistiksel rasgelelik testlerini geçmesi gerekmektedir. Bu yüzden Bölüm 6, dizilerin rasgeleliğinin belirlenmesi için yararlanılan istatistiksel rasgelelik testlerinin birkaçının, bu testlerin değerlendirilmesinde kullanılan istatistiksel hipotezlerin kontrolunun açıklanmasının ardından incelenmesiyle noktalanacaktır.

6.1 Kriptografi

Kriptografi ("Cryptography"), düşman personelin açık mesaj ("plaintext") olarak çağrılan korunmasız mesajı yakalamak ve doğru olarak yorumlamak için teknik yeteneğe sahip olduğu durumlarda, mesajların gizliliğini garantilemek amacıyla kullanılan tekniklerin tümüdür. Yani özel bir bilgiyi izinsiz girişlere karşı korumak anlamına gelen kriptografi, yazının başlangıcı kadar eskidir. Herhangibir mesaj güvenli hale getirildikten sonra, şifreli mesaj ("ciphertext") veya kriptogram ("cryptogram") olarak adlandırılır. Açık mesajı şifreli mesaj şekline getiren dönüşüm sürecine, şifreleme ("encryption") denir. Şifreli mesajı açık mesaj haline getiren ters dönüşüm sürecine ise şifre çözme ("decryption veya deciphering") denir. Kriptoanaliz ("Cryptanalysis") de, şifreli mesajdan yetkisizce bilgi çıkarılmasıdır.

Pratik bir kriptografik iletişim sisteminin ürettiği mesaj, kriptoanalize dayanıklı olmakla kalmayıp yetkili insanlar tarafından da çözülebilmelidir.

Şifreleme algoritması veya dönüşümü, şifre olarak adlandırılır. Yani bir şifre, açık mesajı (M) bir dönüşüm yardımıyla şifreli mesaj (C) olarak adlandırılan karışık bir şekle getirerek, M'nin gizlenmesini sağlar. Şekil 6.1, bir şifre tanımlar.



Şekil 6.1. Bir şifre.

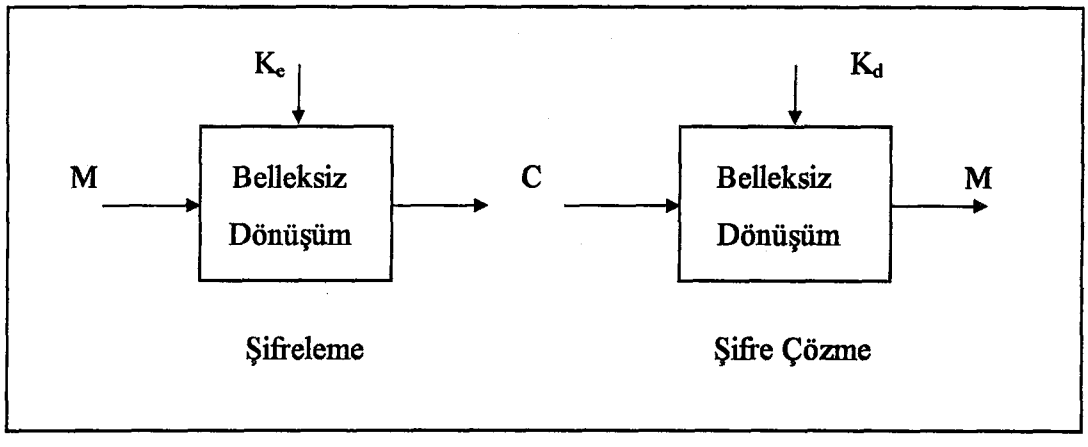
Açık mesajın yetkisiz kişiler tarafından kolayca açığa çıkarılmasının önlenmesi amacıyla, gönderici bu açık mesajı belirli bir parametre yardımıyla çok sayıda mümkün şifreli mesaja dönüştürülebilmelidir. Bu belirli parametre, şifreleme anahtarı ("encryption key") (K_e) olarak adlandırılır. Bu işlemlerin ardından alıcı, güvenli bir kanal vasıtası ile iletilen şifreli mesajı çözmek amacıyla, şifre çözme

anahtarını ("decryption key") (K_d) kullanır. Genel anahtarlı kriptosistemlerinde , K_e genelken; K_d gizli tutulur. Hesaplama yoluyla K_e anahtarından K_d anahtarını ortaya çıkarmak mümkün değildir. Özel anahtarlı kripto sistemlerinde, gönderici ve alıcı çoğunlukla hem şifrelemede hem şifre çözmede kullanılan ortak bir anahtarı (K) paylaşırlar. Değiştirilebilen bu K anahtarı daima gizli tutulur.

Özel anahtarlı kriptosistemler, blok şifreleme sistemleri ve dizi şifreleme sistemleri olmak üzere iki kısma ayrılır. Blok şifreleme sistemlerinde açık mesaj m bitlik bloklara ayrılır, $(x_i, x_{i+1}, \ldots, x_{i+m-1})$ ve her bir bloğa açık mesajda n bitlik blok karşı getirilir, $(y_i, y_{i+1}, \ldots, y_{i+n-1})$. Bu dönüşüm,

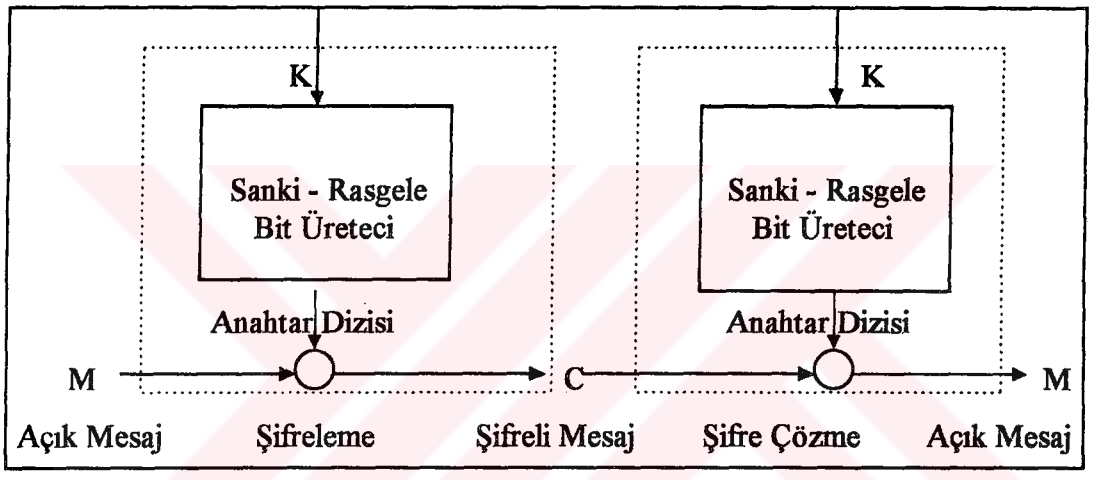
$$y_k = f_k(x_i, x_{i+1}, \ldots, x_{i+m-1}), \quad i \le k \le i+n-1 \tag{6.1}$$

şeklinde tanımlanır. Bu şifreleme, f_k fonksiyonlarının tersi yardımıyla çözülür. Şifrenin tek olarak çözülebilmesi için, $n \ge m$ şartı gereklidir. Hemen hemen tüm pratik blok şifreleme sistemlerinde, $n = m$ olarak alınır. Her bir bloğun birbirinden bağımsız olarak sabit bir anahtar kontrolü altında şifrelendiği blok şifreleme sistemlerinde, farklı yerlerde görülen aynı açık mesaj bloklarına, aynı şifreli blok karşı getirilir. Yani şifreli mesajdaki farklı desenlerin ("pattern") gözlenme sıklığını analiz ederek gizli bilgiyi ortaya çıkarmaya çalışan kriptoanalizstin ("cryptanalyst") ataklarını boşa çıkarmak için blok, yeterince uzun olmalıdır. Şekil 6.2, bir blok şifreyi gösterir.



Şekil 6.2. Bir blok şifre

Diğer taraftan dizi şifreleme sistemlerinde açık mesaj, bit - bit şifrelenir. Bu tür sistemlerde iletilen verinin şifrelenmesi için, kısa K anahatarı, uzun ikili diziler üretmek amacıyla sanki - rasgele bit üreteci olarak adlandırılan bir algoritmadan geçirilir. Anahtar dizisi ("key - stream") olarak adlandırılan bu uzun dizi, açık mesajla karıştırılır. Bu karıştırma işlemi genellikle modülo - 2'ye göre toplamı ifade eden dışlamalı veya (Exclusive - Or) kapılarıyla gerçekleştirilir. Şekil 6.3, bu ilkeyi ifade eder.



Şekil 6.3. Bir dizi şifre.

6.1.1 Tek Zamanlı Sistemler ("The One - Time Pad ")

Dizi şifreleme sistemleri, anahtar dizisinin açık mesajdan bağımsız olduğu sistemler ve anahtar dizisinin açık mesajdan bağımsız olmadığı sistemler olmak üzere ikiye ayrılır. Bağımsız anahtarlı şifreler, eşzamanlı ("synchronous") şifreler olarak adlandırılır. Çünkü bu tür sistemlerde şifreli mesajın doğru olarak çözülebilmesi için, anahtar dizisi ile şifreli mesaj eşzamanlı olmalıdır. Anahtar dizisinin, açık mesajın fonksiyonu olduğu dizi şifreleri, kendinden eşzamanlı şifreler ("self - synchronizing") veya oto - anahtar şifreler ("auto - key ciphers") olarak adlandırılır.

Şartsız olarak güvenli tek sistem, gerçek rasgele bir dizi üreten eşzamanlı bir şifreleme sistemidir. Bu şifre, anahtar dizisinin bir kereye mahsus olmak üzere kullanılmasından dolayı, tek zamanlı sistem ("one - time pad veya one - time tape") veya Vernam sistem olarak adlandırılır. Bu sistemde, anahtar dijitleri rasgele olarak

seçilir ve anahtar dizisi mesajla aynı uzunlukta olmakla beraber kendisini asla tekrar etmez.

Bu sistemde şifreli mesaj, n uzunluklu açık mesajın, yukarıda özellikleri verilen n uzunluklu gizli bir anahtar dizisi ile XORlanması sonucunda elde edilir. Şekil 6.4, bu ilkeyi gösterir. Alıcı, şifreli mesajı, güvenli bir kanal vasıtasıyla iletilen aynı gizli anahtar dizisiyle XORlayarak açık mesaj haline getirir.

Kriptoanalizst, tüm olası anahtarların araştırıldığı kaba kuvvet atağı ("brute - force attack") ile bu tür sistemi kırmak isterse, tüm anlamlı açık mesajları elde eder. Örneğin şifrelenen kelime "Ayşe" olsun. Kriptoanalizst kaba kuvvet atağı sonunda "Ayşe" isminin de dahil olduğu türkçede dört harfle ifade edilebilecek tüm isimleri elde eder. Fakat onun elinde, elde ettiği hangi ismin doğru olduğuna dair hiçbir ipucu yoktur.

Açık Mesaj M: 011000111111101... Gizli Anahtar K: 100110010001011...
Şifreli Mesaj C: 111110101110110...

Şekil 6.4. Tek zamanlı bir sistem.

Eğer K anahtarı, ikili simetrik bir kaynak tarafından üretiliyorsa, öyleki şifreli mesajın hehangibir sembolünün 1 olma olasılığı diğer sembollerden bağımsız olarak 1/2 dir; tek zamanlı sistem, kriptoanalizstin açık mesaj hakkında hiçbir bilgisinin olmadığı atak türüne (ciphertext - only attack) karşı mükemmel güvenlidir. Pratikte açık mesaj hakkında bir miktar bilgi elde mevcuttur. Bazı şifreli metinlere karşılık gelen açık metinlerin bilinmesi bilgisini temel alarak yapılan bu atak türü, bilinen açık mesaj atağı ("known - plaintext attack") olarak adlandırılır. Şartsız olarak güvenli bir şifre, elde mevcut olan açık mesaj - şifreli mesaj çiftlerinin sayısına bağlı olmaksızın kriptoanalizden yara almayan tek sistemdir.

Bu sistem, iki kez kullanılmayan uzun rasgele anahtar dizilerine ihtiyaç duyduğu için pratikte kullanışlı değildir.

Tek zamanlı sistemlerin kullanımının güçlüğünden dolayı, yirmili yıllardan beri pek çok şifreleme sistemi, gizli tutulan kısa bir anahtarın bir algoritma yardımıyla ürettiği uzun sanki - rasgele dizileri kullanan tek zamanlı sistemlerin yaklaşık uyarlamaları şeklinde tasarlanmaktadır.

6.2 Güvenli Sanki - Rasgele Bit Üreteçleri

Kriptolojik uygulamalar için güvenli anahtar dizi üreteçlerinin üç özelliği kesin ve açık olarak belirtilebilir.

1) Anahtar dizisinin periyotu, iletilmek istenen mesajın uzunluğu ile uzlaşacak kadar büyük olmalıdır.

2) Çıkış bitlerinin üretimi kolay olmalıdır.

3) Çıkış bitlerinin tahmini güç olmalıdır. Bir üretecin ilk n çıkış biti kullanılarak, dizinin (n+1). biti hesap yolu ile %50'den daha iyi bir olasılıkla tahmin edilememelidir. Yani çıkış dizisinin verilen bir parçası için, kriptoanalizst ne bu noktadan önceki ne de bu noktadan sonraki diğer bitleri üretememelidir.

Verilen belirli bir anahtar dizi üretecinin hangi niteliği temel alınarak, üretecin oluşturduğu çıkış işaretinin tahmin edilemez olduğu sonucu çıkarılır? Bit üreteçlerinin güvenliğini doğrulayan her zaman ve her yerde uygulanabilir ve pratik olarak kontrol edilebilir bir kriter geliştirilmemiş durumdadır.

6.2.1 İnşa Blokları

Sezgisel olarak, rasgele oluşun manasının tahmin edilemezlik olduğu söylenir. Bir dizinin rasgele olabilmesi için, periyodu büyük olmalı ve çeşitli uzunluktaki desenler, tüm dizi boyunca düzgün olarak dağılmalıdır. Bu bölümde sanki - rasgele dizi üreten bazı basit yöntemlerden, doğrusal geribeslemeli ötemeli yazıcılar, doğrusal olmayan ötemeli yazıcılar ve doğrusal kongruens üreteçlerinden söz edilecektir.

6.2.1.1 Doğrusal Kongruens Üreteçler ("Linear Congruence Generators ")

Sanki - rasgele sayılar üretmek amacıyla oldukça fazla tartışılan yöntemlerden biri,

$$\mathbf{X(i+1) = aX(i) + b \mod m} \tag{6.2}$$

ifadesiyle verilir. Gizli anahtar olarak kullanılabilen $(\mathbf{a,b,m})$, üreteci tanımlayan parametrelerdir. $\mathbf{X_0}$, başlangıç değeridir. Eğer parametre değerleri akıllıca seçilirse, $\mathbf{X(i)}$ sayıları $[0, m-1]$ aralığındaki tüm tamsayılar oluşana kadar kendilerini tekrarlamazlar. Bu üretece örnek olarak $\mathbf{X_0 = 1}$ olmak üzere,

$$\mathbf{X(i) = 5X(i-1) + 3 \mod 16} \tag{6.3}$$

verilebilir. Üretecin ürettiği dizi değerleri, $\{1,8,11,10,5,12,15,14,9,0,3,2,13,4,7,6,\ 1,8,...\}$ şeklindedir.

[45], doğrusal kongruens üreteç dizilerinin kriptolojik uygulamalar için güvenli olmadığını gösterdi. Bu tür bir dizinin verilen yeterince uzun bir parçasından, m, a ve b parametreleri yeniden inşa edilebilir.

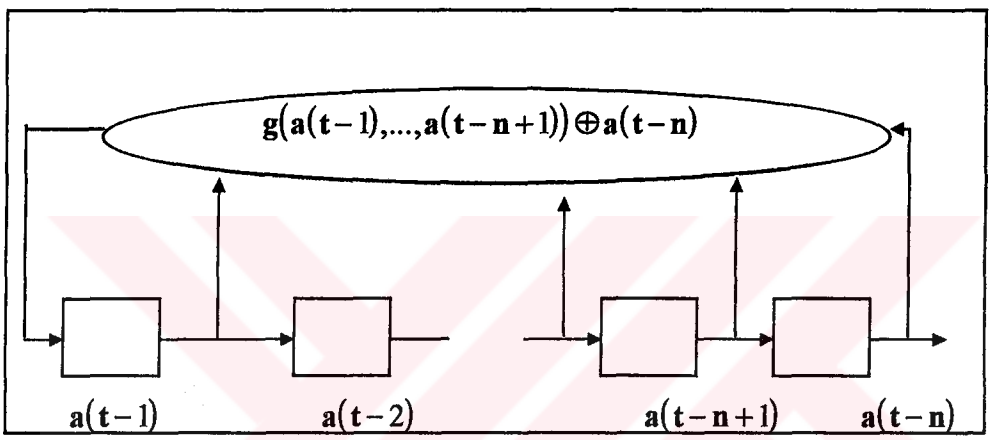
6.2.1.2 Geribeslemeli Ötemeli Yazıcılar

Güçlü kriptografik diziler, geribesleme devresi doğrusal olsa bile ötemeli yazıcılar temel alınarak oluşturulabilir. Geribeslemeli ötemeli bir yazıcı, n tane flip - flop ve her yeni elemanı $\mathbf{a(t)}$, $\mathbf{t \geq n}$ olmak üzere, daha önceden üretilen elemanlar $\mathbf{a(t-n), a(t-n+1), ..., a(t-1)}$ cinsinden ifade eden bir geribesleme fonksiyonundan meydana gelir. Şekil 6.5, n - durumlu geribeslemeli bir yazıcıyı gösterir. Tekil olmayan geribesleme fonksiyonu,

$$\mathbf{a(t) = g\big(a(t-1), a(t-2), a(t-n+1)\big) \oplus a(t-n)} \tag{6.4}$$

şeklinde verilir. ⊕ sembolü, XOR işlemini ifade eder.

g fonksiyonunun doğrusal olup olmadığına bağlı olarak, üreteç doğrusal geribeslemeli ötemeli yazıcı (DGÖY) veya doğrusal olmayan geribeslemeli ötemeli yazıcı (DOGÖY) olarak adlandırılır. Geribeslemeli ötemeli yazıcının (GÖY) her bir elemanı, durum olarak adlandırılır. Bir dizi üretmek için $a(0), a(1)\, a(2), \ldots, a(n-1)$ ikili sayıları, başlangıç değerleri olarak GÖY'ya yüklenir.



Şekil 6.5. n - durumlu geribeslemeli ötemeli yazıcı.

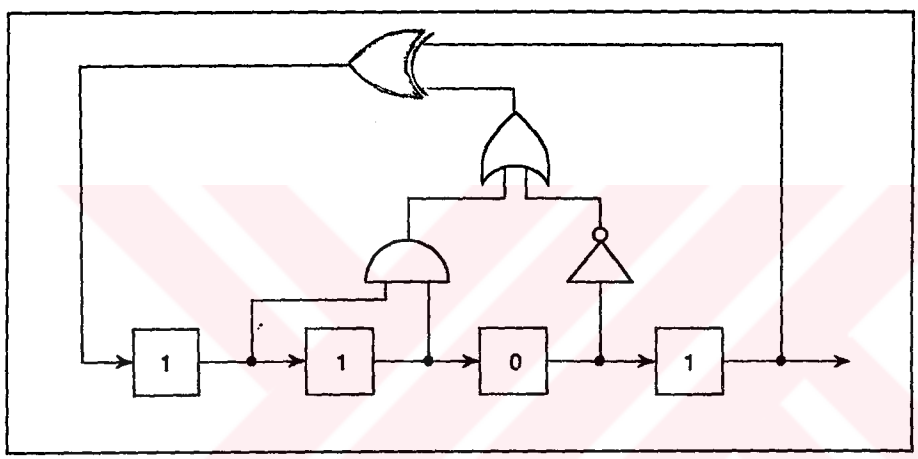
GÖY'nın ürettiği dizinin periyotu, hem durum sayısına hem geribesleme bağlantılarına bağlıdır. Geribesleme fonksiyonu tekil olmayan n - durumlu GÖY tarafından üretilen dizinin maksimum periyotu, 2^n dir. 2^n, n - durumlu ötemeli yazmacın sahip olabileceği mümkün tüm durumların sayısısır.

6.2.1.3 Doğrusal Olmayan Geribeslemeli Ötemeli Yazıcılar

Tekil olmayan geribeslemeli ötemeli yazıcının durum diyagramı, ufak çevrimlere sahip olabilir. Eğer üreteç bu durumlardan herhangibirine düşerse, üretilen çıkış dizisi güvenli olmaz. Bu duruma karşı tedbirler almak amacıyla, n.dereceden De Brujin dizileri olarakta çağrılan mümkün en büyük periyotlu (2^n) dizilerini üreten n - durumlu ötemeli yazıcılar dizayn edilir. 1011001010000111..., Şekil 6.4'de verilen DOGÖY tarafından üretilen periyotu 2^4 olan bir De Brujin dizisidir. n - durumlu De Brujin dizilerinin sayısı, $2^{2^{n-1}-1}$ dir ve bu diziler ideal desen dağılımına sahiptirler.

Örneğin 4 uzunluklu her bir desen, üreteç tarafından bir periyotta yalnızca bir kez üretilir.

Bilinen algoritmalarla üretilen De Brujin dizilerinin ya hızlı üretilmelerini gerçekleştirmede tekniksel güçlükler vardır ya da öz - ilişkisel karakteristikleriyle ilgili zayıflıkları vardır. Örneğin hızlı gerçekleştirmeye uyarlanabilen bu dizilerin geniş bir sınıfı için, $a(t)$ ve $a(t-n)$ arasındaki çakışma olasılığı ("coincidence probability"), 1/2'den büyüktür. Kriptoanalizst, bu özelliği çok iyi bir şekilde kullanabilir.



Şekil 6.6. Dört durumlu De Brujin dizi Üreteci

6.2.1.4 Doğrusal Geribeslemeli Ötemeli Yazıcılar

DGÖY devreleri, uzun süredir hata denetim kodlamasında, VLSI testlerinde, geniş bantlı iletişim sistemlerinde ve bunlara benzer uygulama alanlarında kullanılmaktadır. Bu devreler aynı zamanda sanki - rasgele bit üreteçleri oluşturmada kullanılan en önemli araçlardan biridir. $c_i \in \{0,1\}$ olmak üzere, DGÖY devrelerinin geribesleme fonksiyonları,

$$a(t) = c_1 a(t-1) \oplus c_2 a(t-2) \oplus \ldots c_{n-1} a(t-n+1) \oplus a(t-n) \qquad (6.5)$$

şeklindedir. DGÖY'nın geribesleme bağlantıları, geribesleme polinomu ile ifade edilebilir,

$$f(x) = 1 + c_1 x + c_2 x^{n-2} + \ldots + c_{n-1} x^{n-1} + x^n. \qquad (6.6)$$

Denklemde görülen x'in matematiksel sembol olmasından başka bir anlamı yoktur. Çıkış dizisinin periyotunu ve istatistiksel davranışını, geribesleme polinomu belirler. Aşikar çıkışdan kaçınmak için sıfir durumu başlangıç olarak alınmaz. Örneğin dört durumlu bir DGÖY'nın geribesleme polinomu

$$f(x) = 1 + x + x^2 + x^3 + x^4 \qquad (6.7)$$

şeklinde ise, başlangıç koşuluna bağlı olarak DGÖY,

1) 1111011110...,

2) 1000110001... ve

3) 0100101001...

dizilerinden birini üretir. Periyotu 5 olan bu diziler, oldukça kötü istatistiklere sahiptirler. Aksine eğer DGÖY'nın geribesleme polinomu $f(x) = 1 + x + x^4$ şeklinde ise, üreteç oldukça iyi istatistikte periyotu 15 olan tek bir dizi üretir,

101100100011110....

Çıkış dizisinin mümkün en büyük periyotlu olmasını garantilemek için, DGÖY'nın geribesleme fonksiyonu $f(x)$ ilkel olmalıdır. Yani $f(x)$, fonksiyonu $(x^T - 1)$ ifadesine tam olarak bölünebilir yapan en küçük pozitif tamsayı T,

$$T = 2^n - 1 \qquad (6.8)$$

olacak şekilde seçilir. Bu biçimde seçilen $f(x)$ fonksiyonunun ilkel olduğu söylenir. Hazırda polinomların ilkel olup olmadığını test eden algoritmalar vardır. n. dereceden ilkel polinomların sayısı,

$$\frac{\phi(2^n - 1)}{n} \qquad (6.9)$$

şeklinde verilir. Euler fonksiyonu olarak bilinen $\phi(x)$, x tamsayısından küçük ve x ile aralarında asal olan pozitif tamsayıların sayısını verir. İlkel polinoma sahip bir DGÖY tarafından üretilen dizi, maksimum uzunluklu DGÖY dizisi veya basitçe m -dizisi olarak adlandırılır. Periyotun maksimum uzunluklu olması, aynı zamanda dizinin iyi istatistikte olmasını garanti eder.

Büyük periyotlara, ideal istatistik özelliklere ve ilkel polinomların çokluğuna rağmen, m - dizileri kriptografik dönüşümlerden geçirilmeksizin anahtar dizisi olarak kullanılamaz. Gerçekte, n durumlu DGÖY'nın ürettiği çıkış dizisinin 2n biti gözlenerek, başlangıç durumu ve geribesleme bağlantıları belirlenebilir. Bunu görmek için,

$$a(t),a(t+1),...,a(t+n-1),a(t+n),...,a(t+2n-2),a(t+2n-1) \qquad (6.10)$$

dizi parçasının verildiği varsayılsın. $a(t+n),...,a(t+2n-2),a(t+2n-1)$ bitlerinin her biri, soldaki n bite dayalı olarak yazılabilir,

$$\begin{aligned}
&a(t)c_0 \quad \oplus a(t+1)c_1 \oplus ... \oplus a(t+n-1)c_{n-1} = a(t+n)\\
&a(t+1)c_0 \quad \oplus a(t+2)c_1 \oplus ... \oplus a(t+n)c_{n-1} = a(t+n+1)\\
&\vdots\\
&a(t+n-1)c_0 \oplus a(t+n)c_1 \oplus ... \oplus a(t+2n-2)c_{n-1} = a(t+2n-1).
\end{aligned} \qquad (6.11)$$

Geribesleme sabitleri, bu doğrusal cebirsel denklemlerin, $c_1,...,c_{n-1}$ katsayılarının bilinmeyen olarak alınıp çözülmesiyle belirlenir.

Genel olarak her periyodik dizi, tekil olmayan DGÖYlar tarafından üretilebilir. Bu tür bir diziyi üretebilecek en kısa uzunluklu DGÖY'nın geribesleme fonksiyonunu bulan etkili bir sentez işlemi vardır, [46]. Böyle bir DGÖY'nın uzunluğu, a dizisinin doğrusal karmaşıklığı ("linear complexity") olarak adlandırılır ve $DK(a)$ ("$LC(a)$") olarak gösterilir. Sonuç olarak kriptografik kullanımlara uygun bir üreteç dizayn etmek için; Üretecin meydana getirdiği dizilerin doğrusal karmaşıklığının altsınırının yeterince büyük olmasını garantilemek gerekmektedir, [47].

6.3 İstatistiksel Rasgelelik Testleri

Bu bölümde dizilerin rasgeleliğinin belirlenmesi için oluşturulmuş istatistiksel testler üzerinde durulacaktır. Bu amaçla rasgeleliği belirlenmek istenen dizi için istatistik olarak adlandırılan bir değer hesaplanacak ve elde edilen bu değer, rasgele dizinin bu değere karşı gelen dağılımıyla karşılaştırılacaktır. Eğer sanki - rasgele bir dizi, oldukça az sayıda rasgele dizinin sahip olduğu bir özelliği sergilerse, sanki - rasgele dizinin, rasgele diziyi iyi bir şekilde modellemediği sonucuna varılacaktır.

6.3.1 İstatistik Hipotezlerin Kontrolu

Dizinin belirli bir özelliğinin test edilmesi istensin. Bu durumda ilk olarak bu özelliği ölçen bir istatistik tanımlanır ve ardından bu istatistiğin rasgele diziler için izlediği dağılım belirlenir.

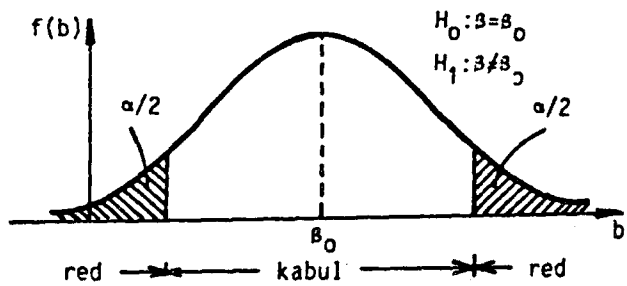
Bir istatistiksel test, sıfır hipotezi ("null hypothesis") olarak adlandırılan bir hipotezin, karşıt hipoteze ("alternative hypothesis") karşı kontrolü ile gerçekleştirilir. β gibi bir parametre değerinin, seçilen bir β_0 değerine eşit olduğunun kabul edilip edilemeyeceğine karar vermek için yapılan işlemlere, $\beta = \beta_0$ istatistik hipotezinin kontrolu (test) denir. Burada β, ortalama,standart sapma,... gibi herhangibir parametre ve β_0, bu parametrenin bilinmeyen değeri için incelenen probleme göre seçilen bir değerdir.

H_0 diye adlandırılan $\beta = \beta_0$ (β_0, X istatistiği için kabul edilen bir değerdir) hipotezini kontrol etmek için önce şunların belirlenmesi gerekir:

1) Hipotezin reddedilmesi halinde kabul edildiği varsayılacak olan H_A karşıt hipotezi: İncelenen problemin yapısına göre $\beta \neq \beta$, ya da $\beta > \beta_0$ (veya $\beta < \beta_0$) şeklinde olabilir. Karşıt hipotezin ne şekilde seçileceği çalışmanın amacına bağlıdır. İlgilenilen sadece β'nın β_0'a eşit olup olmadığı ise karşıt hipotez $\beta \neq \beta$ şeklinde alınabilir. Buna karşılık bazı problemlerde bizi özellikle β'nın β_0'a eşit mi, yoksa β_0'dan özellikle büyük mü (ya da küçük mü) olduğu ilgilendirebilir. Bu durumda karşıt hipotez $\beta > \beta_0$ (ya da $\beta < \beta_0$) şeklinde seçilmelidir.

2) Hesaplanan β istatistiğinin β_0'dan ne kadar farklı olması halinde H_0 hipotezinin reddedileceği seçilecek α anlamlılık düzeyine ("significance level") bağlıdır. α'nın değeri seçildikten sonra $\beta = \beta_0$ değeri merkez olmak üzere çizilen X istatistiğinin örnekleme dağılımı, kabul bölgesi ve red bölgesi olarak ikiye ayrılır. Gözlenen X değerinin içinde kalması halinde H_0 hipotezinin kabul edileceği kabul bölgesi β_0 değerine yakın, H_0 hipotezinin reddedileceği kritik bölge β_0 değerine uzak olup red bölgesinin alanı, α'ya eşit alınır. Red bölgesinin, örnekleme dağılımının her iki ucunda mı, yoksa sadece bir ucunda mı yer alacağı H_A karşıt hipotezinin tipine bağlı olup buna göre iki uçlu ya da bir uçlu test uygulanır:

a) İki Uçlu Test: $\mathbf{H}_0: \beta = \beta_0$ hipotezini, $\mathbf{H}_A: \beta \neq \beta_0$ hipotezine göre kontrol etmek isteniyorsa, gözlenen X değerinin gerek bu istatistik için kabul edilen β_0'dan fazla küçük, gerekse fazla büyük olması hallerinde, H_0 hipotezini reddetmek gerekeceğinden, red bölgesi örnekleme dağılımının her iki ucuna simetrik olarak yerleştirilir. Buna göre, örnekleme dağılımında aşılması olasılığı $\alpha/2$ ve $1-\alpha/2$ olan değerlerin arasında kalan bölge kabul bölgesi, bu değerlerin dışında kalan iki bölge red bölgesidir (Şekil 6.7).

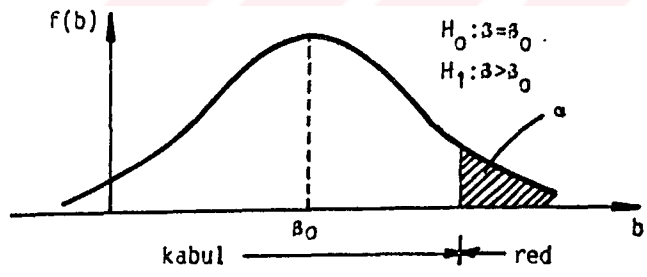


Şekil 6.7. $\beta = \beta_0$ hipotezinin, $\beta \neq \beta_0$ karşıt hipotezine göre kontrolu.

b) Bir Uçlu Test: $\mathbf{H}_0: \beta = \beta_0$ hipotezinin kontrolunde karşıt hipotez $\mathbf{H}_A: \beta > \beta_0$ (ya da $\beta < \beta_0$ şeklinde ise gözlenen X değerinin sadece β_0'dan fazla büyük (fazla küçük) olması halinde H_0 hipotezi reddedileceğinden, red bölgesi, örnekleme dağılımının sağ

(sol) ucuna yerleştirilir. Buna göre, $\beta = \beta_0$ merkezi çevresinde çizilen X istatistiğinin, örnekleme dağılımında aşılma olasılığı olan α, kritik bölgenin sınırını belirler, Şekil 6.8.

İstatistik hipotezler (H_0 ve H_A hipotezleri), rasgele değişkenin dağılımının özellikleri ile ilgili bir problemde mantıklı bir temele dayanılarak yapılan kabullerdir. Hipotezlerin kontrolu için, hipotezde yer alan parametreye karşı gelen istatistiğin örnekleme dağılımını bilmek gerektiği görülmektedir. Anlamlılık düzeyi seçildikten sonra, karşıt hipotezin tipine göre, örnekleme dağılımının ya her iki ucunda ya da sadece bir ucunda bulunan kritik bölgenin sınırları belirlenmektedir. Gözlenen örnekten hesaplanan istatistik değerinin red bölgesine düşmesi halinde H_0 hipotezi red, aksi halde kabul edilmektedir. Böylelikle bir rasgele değişkenin herhangibir parametresinin bilinmeyen değeri için yapılan bir seçimin doğruluğunun, eldeki örnekten o parametre için tahmin edilen değerle karşılaştırılarak, kabul edilip edilmeyeceğine karar verilmektedir.



Şekil 6.8. $\beta = \beta_0$ hipotezinin, $\beta > \beta_0$ karşıt hipotezine göre kontrolu.

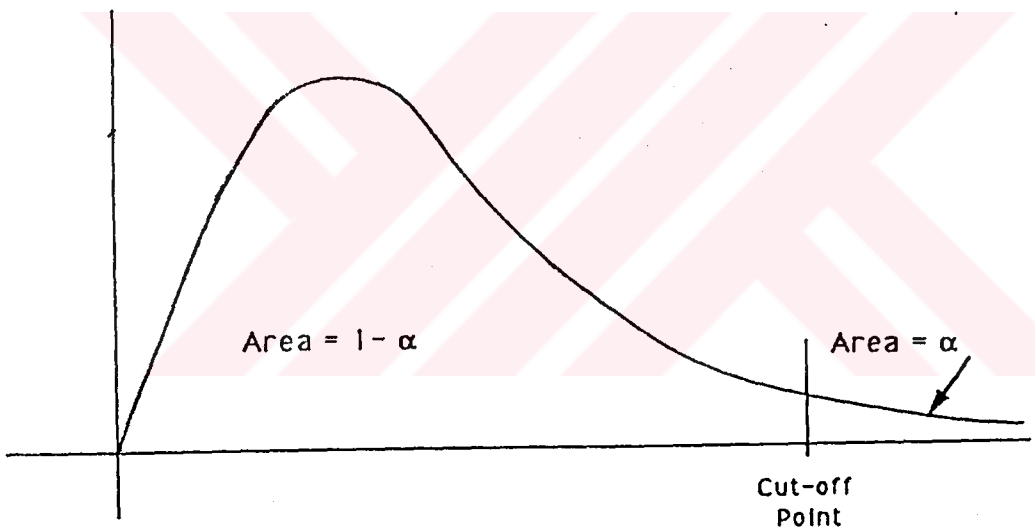
Buradaki incelemede sıfır hipotezi, H_0, rasgeleliği belirlenecek olan dizilerin, rasgele bir kaynak tarafından üretildiğidir. Karşıt hipotez H_A ise, dizilerin rasgele olmayan bir kaynak tarafından üretildiğidir. Sıfır hipotezinin belirlenmesinin ardından anlamlılık düzeyi ("significance level"), α, belirlenir. α'nın, H_0'ın doğru olduğu halde test sonunda reddedilmesi olasılığını gösterdiği görüldü. Yani α,

$\alpha = P(H_0$'ı reddet $|$ H_0 doğru) (6.12)

şeklinde tanımlanır. $0 \leq \alpha \leq 1$ aralığında olan α, 0.05 veya 0.01 olarak seçilir.

Rasgele bir kaynak tarafından üretilen bir dizinin test sonucunda reddedilmesi istenmezken, rasgele bir dizinin karakteristiklerini içermeyen dizinin test sonucunda kabul edilmesi de kriptografik uygulamalar için tehlikelidir.

Rasgele bir değişken, X, N(0,1) dağılımına (ortalaması 0, varyansı 1 olan normal dağılım) sahip k tane bağımsız değişkenin karelerinin toplamı ise, serbestlik derecesi ("degrees of fredom") k olan χ^2 - dağılımına sahiptir. Şekil 6.9, bu dağılımın grafiğini göstermektedir.



Şekil 6.9. x^2 - dağılımı.

Tablo 6.1, hesaplanan istatistiğin, seçilen bir X_0 değerini aşması olasılığını verir.

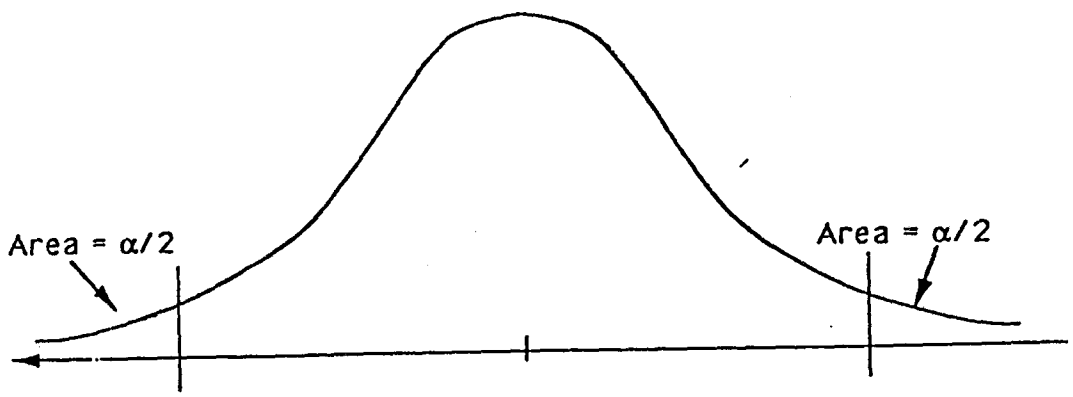
Hesaplanan istatistiğin, gerçek rasgele diziler için serbestlik derecesi k olan χ^2 dağılımını izlediği varsayılsın. Eğer istatistiğin, bu parametre için seçilen bir istatistikten büyük olması bekleniyorsa, tek uçlu test uygulanması gerektiği daha önceden görüldü. Bu durumda $X_{k,1-\alpha}$ kesim noktası,

$P(X>X_{k,1-\alpha} |$ X, rasgele bir dizinin istatistiğidir) (6.13)

şeklinde ve $X_{k,1-\alpha}$ değeri de, Tablo 6.1'deki istatistiksel dağılımdan belirlenir. Örneğin $k = 5$ ve $\alpha = 0.05$ olduğunda, $X_{k,1-\alpha} = 11.07$ olarak belirlenir. Yani $X > X_{k,1-\alpha}$ bölgesi kritik bölgedir. Rasgele olmayan bir dizi için belirlenen istatistiğin kritik bölgeye düşmesinin, rasgele bir dizininkine göre daha olası olduğu beklendiği için, $X > X_{k,1-\alpha}$ ise, H_0 reddedilir. Dolayısıyla H_0 doğru olduğu halde rededilme olasılığı α'dır,

$$\begin{aligned}\alpha &= P(X > X_{k,1-\alpha} \mid X, \text{rasgele bir dizinin istatistiğidir}) \\ &= P(X > X_{k,1-\alpha} \mid H_0, \text{doğrudur}) \\ &= P(H_0\text{'ı reddet} \mid H_0, \text{doğrudur}).\end{aligned} \tag{6.14}$$

Eğer karşıt hipotez altında hesaplanan istatistiğin, bu parametre için seçilen istatistikten yalnızca büyük olması yerine, daha büyük ya da daha küçük olması bekleniyorsa, iki uçlu test uygulanması gerektiği daha önceden görüldü. İki uçlu test genellikle rasgele dizinin istatistiği normal dağılım izlediğinde kullanılır. Normal dağılım, merkez limit teoremi("central limit theorem") ile açıklanabilir. Bu teoreme göre, $X_i (i = 1,2,\ldots,n)$ birbirinden bağımsız rasgele değişkenler ise, $X = \sum_{i=1}^{n} c_i X_i$ şeklinde, X_i lerin sabit sayılarla çarpılıp toplanmalarıyla elde edilen (X_i lerin doğrusal bir kombinezonu olan) X rasgele değişkenini dağılımı, X_i lerin ayrı ayrı dağılımları ne olursa olsun, n büyüdükçe hızla normal dağılıma yaklaşır. Şekil 6.10'de N(0,1) dağılımı verildi.



Şekil 6.10. N(0,1) dağılımı.

Bu durumda $X_{1-\alpha/2}$ ve $X_{\alpha/2}$ değerleri bulunur. Eğer $X > X_{1-\alpha/2}$ veya $X < X_{\alpha/2}$ ise H_0 reddedilir. Kesim noktaları,

$$\alpha/2 = P(X < X_{k,1-\alpha/2} \mid X, \text{rasgele bir dizinin istatistiğidir})$$
$$= P(X < X_{k,1-\alpha/2} \mid X, \text{rasgele bir dizinin istatistiğidir}) \quad (6.15)$$

olacak şekilde belirlenir.

Tablo 6.1. χ^2 - dağılımı

	0.99	0.95	0.75	0.50	0.25	0.05	0.01
v = 1	0.00016	0.00393	0.1015	0.4549	1.323	3.841	6.635
v = 2	0.02010	0.1026	0.5753	1.386	2.773	5.991	9.210
v = 3	0.1148	0.3518	1.213	2.366	4.108	7.815	11.34
v = 4	0.2971	0.7107	1.923	3.357	5.385	9.488	13.28
v = 5	0.5543	1.1455	2.675	4.351	6.626	11.07	15.09
v = 6	0.8720	1.635	3.455	5.348	7.841	12.59	16.81
v = 7	1.239	2.167	4.255	6.346	9.037	14.07	18.48
v = 8	1.646	2.733	5.071	7.344	10.22	15.51	20.09
v = 9	2.088	3.325	5.899	8.343	11.39	16.92	21.67
v = 10	2.558	3.940	6.737	9.342	12.55	18.31	23.21
v = 11	3.053	4.575	7.584	10.34	13.70	19.68	24.73
v = 12	3.571	5.226	8.438	11.34	14.84	21.03	26.22
v = 15	5.229	7.261	11.04	14.34	18.25	25.00	30.58
v = 20	8.260	10.85	15.45	19.34	23.83	31.41	37.57
v = 30	14.95	18.49	24.48	29.34	34.80	43.77	50.89
v = 50	29.71	34.76	42.94	49.33	56.33	67.50	76.15
v >30							
x_p	-2.33	-1.64	-.675	0.00	0.675	1.64	2.33

$X_{\alpha/2}$ ve $X_{1-\alpha/2}$ değerleri, N(0,1) dağılımına ait tablodan belirlenebilir. Örneğin $\alpha = 0.05$ ise, $X_{\alpha/2} = -1.96$ ve $X_{1-\alpha/2} = 1.96$ dır.

Tablo 6.1'de $v > 30$ olduğunda istatistik değeri,

$$v + (2v)^{1/2} x_p + 2/3\, x_p\, x_p - 2/3 + O(1/v^{1/2}) \qquad (6.16)$$

olarak belirlenir.

6.3.2 İstatistiksel Rasgelelik Testleri

Burada beş istatistiksel test sunulacaktır. Burada verilecek örnekler, rasgeleliğin belirlenmesi için her bir testin, kendi başına yeterli olmadığını, fakat diğer testlerle bir arada kullanılması gerektiğini gösterecektir.

6.3.2.1 Frekans Testi

Frekans testi, dizideki 1 ve 0'ların sayısını karşılaştırmak amacıyla geliştirilmiştir. Dizilerin ikili simetrik bir kaynak (BSS) tarafından üretildiği düşünüldüğünden, 0 ve 1'lerin yaklaşık aynı sayıda olduğu beklenir. Çalışılan dizinin uzunluğu n, dizideki sıfırların sayısı n_0 ve dizideki 1'lerin sayısı n_1 ise, istatistik,

$$X = \frac{(n_0 - n_1)^2}{n} \qquad (6.17)$$

şeklinde belirlenir. X istatistiği, serbestlik derecesi 1 olan χ^2 - dağılımını takip ettiği için, Tablo 6.1 yardımıyla $X_{1,1-\alpha}$ kesim noktası bulunabilir. Eğer α, 0.05 alınırsa, 3.841'den büyük olan bölge kritik bölgedir. Yani hesaplanan X istatistiği, 3.841'den büyükse, dizi reddedilir. Bu testi geçen, fakat rasgele olmayan diziler bulmak oldukça kolaydır. Bu testen geçen, fakat rasgele olmayan 12 uzunluklu bazı dizi örnekleri şunlardır, 010101010101 ve 000000111111.

6.3.2.2 Seri Test

Seri test ile dizideki her bir çiftin, 00, 01, 10, 11, meydana geliş sayısının yaklaşık olarak aynı olup olmadığı belirlenir. Yani rasgele bir dizi için her bir çiftin oluş sayısı, yaklaşık olarak $(n-1)/4$ olmalıdır. Dizideki 00'ların meydana geliş sayısının, 01'lerin meydana geliş sayısından daha fazla olduğu varsayılsın. Bu durum, 0'takip eden bitin, sıfır olma olasılığının bir olma olasılığından daha yüksek olması gerektiğini söyler. Yani, bir bitin meydana gelişi, önceki bitten bağımsız değildir. Bu nedenle böyle bir dizi, rasgele bir kaynak tarafından üretilemez.

n_{00}, n uzunluklu bir dizideki 00 çiftinin sayısını göstersin ve n_{01}, n_{10} ve n_{11} da benzer şekilde tanımlansın, bu durumda X istatistiği,

$$X = \frac{4}{n-1}\left(n_{00}^2 + n_{01}^2 + n_{10}^2 + n_{11}^2\right) - \frac{2}{n}\left(n_0^2 + n_1^2\right) + 1. \qquad (6.18)$$

n_0 ve n_1, söylendikleri sıra ile 0 ve 1'lerin sayısıdır.

I. J. Good, X istatistiğinin yaklaşık olarak serbestlik derecesi 2 olan χ^2 - dağılımını izlediğini göstermiştir. Eğer $\alpha = 0.05$ alınırsa, Tablo 6.10 yardımıyla sınırı 5.991 olan kritik bölge belirlenir. Böylece eğer $X > 5.991$ ise, dizi reddedilir.

Bu testi geçen, fakat rasgele olmayan bir dizi, 00110011001100110 şeklinde verilebilir.

6.3.2.3 Poker testi

Poker testi, frekans ve seri testlerinin genelleştirilmiş şeklidir. Frekans testinin 1'li desenlerin oluş sayını verdiği, seri testinin ise, ikili desenlerin her birinin oluş sayısını verdiği görüldü. Poker testi ise, 2^m tane m - bitlik desenlerin her birinin oluş sayısını inceler.

Bu test için ilk olarak n uzunluklu dizi, k tane m - bitlik altdizilere ayrılır. Burada k, n/m'den küçük olan en büyük tamsayı olarak tanımlanır, $k = \left\lfloor \frac{n}{m} \right\rfloor$. 2^m

tane m - bitlik altdizi varolduğundan, n_i, i. altdizinin meydana geliş sayısını, $i = 1,...,2^m$ için göstersin. Bu değerler yardımıyla poker testi için istatistik,

$$X = \frac{2^m}{k}\sum_{i=1}^{2^m} n_i^2 - k \qquad (6.19)$$

şeklinde tanımlanır. Bu istatistik, serbestlik derecesi 2^m-1 olan χ^2 - dağılımını gösterir. Böylece tekrar Tablo 6.1 yardımıyla, kritik bölge belirlenir.

Poker testi, m'nin farklı değerleri için tekrar edilebilir ve dizi m'nin farklı değerleri için tekrarlanan bu testlerin çoğunu geçmelidir. Buna rağmen m'nin büyüklüğüne dair bir sınır vardır. Bu sınır, her bir nesnenin meydana gelişinin beklenen sayısının en az 5 olmasıdır. Buradaki durum için, 2^m tane altdizinin her birinin meydana geliş sayısı en az 5 olmalıdır. Yani $\frac{k}{2^m} \geq 5$ veya $\left\lfloor \frac{n}{m} \right\rfloor \geq 5(2^m)$ eşitsizliği sağlanmalıdır.

Eğer m üzerine bu şekilde bir kısıtlama getirilirse, poker testini geçen, fakat rasgele olmayan bir dizi kolaylıkla bulunur. 000 111 101 011 001 100 010 110 altdizisini 5 kez tekralayan 120 uzunluklu bir dizi, rasgele değildir.

Bu diziden $m = 3$ için elde edilen istatistik,

$$\begin{aligned} X &= \frac{2^3}{40}\sum_{i=1}^{8} n_i^2 - 40 \\ &= \frac{1}{5}(8)(5^2) - 40 \\ &= \frac{200}{5} - 40 \\ &= 0 \end{aligned}$$

şeklindedir. Bu arada $m = 3$ için, istatistik, serbestlik derecesi 7 olan χ^2 dağılımını izlediği için, kritik bölge, $\alpha = 0.05$ olmak üzere, 14.067'den büyük değerler içeren bölgedir.

Aynı dizinin $m = 2$ için istatistiği,

$$X = \frac{2^2}{60}\sum_{i=1}^{4} n_i^2 - 60$$
$$= \frac{1}{15}(4)(15^2) - 60$$
$$= 0$$

şeklindedir. İstatistik, $m = 2$ için serbestlik derecesi 3 olan χ^2 dağılımını izlediğinden, kritik bölge, $\alpha = 0.05$ olmak üzere, 7.815'den büyük değerleri içeren bölgedir.

Bu dizinin $m = 1$ için istatistiği,

$$X = \frac{2}{n}\sum_{i=1}^{2} n_i^2 - n$$
$$= \frac{2}{120}(60^2 + 60^2) - 120$$
$$= 0$$

şeklindedir. Bu durumda $\alpha = 0.05$ için, kritik bölge 3.841'den büyük değerleri içeren bölgedir. Bu gözlemler, 000 111 101 011 001 100 010 110 altdizisini 5 kez tekrarlayan dizinin, poker testini kolayca geçtiğini gösterir.

poker testinin m'nin daha büyük değerlerine izin veren değiştirilmiş bir şekli vardır. Değiştirilmiş bu poker testinde, m - bitlik altdizilerin her birinin meydana geliş sayısını saymak yerine, altdiziler her birinde görünen 1'lerin sayısına bağlı olarak gruplara ayrılır. İlk grup, içinde sıfır tane 1 olan tüm dizileri, ikinci grup, içinde bir tane 1 olan tüm dizileri içerir ve grupların içerdiği tüm diziler bu şekilde düşünülerek düzenlenir. i tane 1 içeren alt dizilerin grup içindeki sayısının beklenen değeri,

$$y_i = \binom{m}{i}\frac{k}{2^m}, \qquad i = 0,\ldots,m \tag{6.20}$$

şeklindedir. İfadede $\binom{m}{i} = \frac{m!}{i!(m-i)!}$ dir.

Hala grup sayısının beklenen değerinin 5'den az olacağı mümkünse, küçük değerde olanlar, her bir grubun beklenen büyüklüğü en az 5 olacak şekilde bir araya getirilir. Bunları gruplamanın bir yolu şu şekilde olabilir:

$$Y_i = \begin{cases} y_0 + y_1 + \ldots + y_d & i = d; \\ y_i, & i = d+1,\ldots,m-d-1; \\ y_{m-d} + y_{m-d+1} + \ldots + y_m, & i = m-d. \end{cases} \qquad (6.21)$$

İfadede görülen d, tüm $i = d,\ldots,m-d$ için $Y_i \geq 5$ olacak şekilde tanımlanır. Bu durumda $m-2d+1$ tane grup vardır: d - grup, d tane 1'den daha fazla 1 içermeyen tüm dizileri içerir, (m - d) - grup, (m-d) tane 1'den daha az 1 içermeyen tüm dizileri içerir ve i - grup, $i = d+1,\ldots,m-d-1$ olmak üzere, i tane 1 içeren tüm dizileri içerir. Bu test için belirlenen istatistik,

$$X = \sum_{i=d}^{m-d} \frac{(n_i - Y_i)^2}{Y_i} \qquad (6.22)$$

şeklindedir. n_i, yukarıda tanımlandığı gibi, incelenen dizide i - grubuna düşen altdizilerin sayısıdır. X istatistiği, serbestlik derecesi $m-2d$ olan χ^2 - dağılımını izler.

6.3.2.4 Hareket Testi

Bir hareket ("a run"), tek bir bitin tekrarlamalarını içeren bir altdizi olarak tanımlanır. 01100010111101 dizisinde, 0, 11, 000, 1, 0, 1111, 0, 1 hareketleri gözlemlenir. Yani bu dizide, 1 tane 3 sıfırdan oluşan hareket, 3 tane 1 sıfırdan oluşan hareket, 1 tane 4 birden oluşan hareket, 1 tane 2 birden oluşan hareket ve 2 tane 1 birden oluşan hareket, gözlemlenir.

Hareket testi, incelenen dizideki çeşitli uzunluklarda değişen hareketlerin sayısını, rasgele bir dizideki bu hareketlerin beklenen değerleriyle karşılaştırarak, kontrol eder. Hareket testini gerçekleştirmek için, dizinin, uzunluğu (n) ve içerdiği hareket sayısı (m) bilinmelidir. Ayrıca dizide en büyük uzunluklu hareketin uzunluğu

(k) da bilinmelidir. k'nın beklenen değeri en az 5'dir. Babbage, k değerini tahmin etmek amacıyla,

$$k = \left\lfloor \log_2 \frac{n}{10} \right\rfloor \tag{6.23}$$

eşitliğini kullandı.

$i \le k-1$ olmak üzere, n_{0i} ve n_{1i}, söylendikleri sıra ile i uzunluklu dizideki sıfır ve birlerin sayısı olsun. n_{0k} ve n_{1k} ise, k uzunluklu veya k'dan daha büyük uzunluklu dizedeki söylendikleri sıra ile sıfır ve birlerin sayısı olsun. y_{0i} ve y_{1i}, n_{0i} ve n_{1i} değerlerinin beklenen değeri olsun. Bu durumda y_{0i} ve y_{1i},

$$y_{0i} = y_{1i} = \frac{m(m-1)(n-m)(n-m-1)\ldots(n-m-i+2)}{2(n-1)(n-2)(n-i)} \quad i = 1,\ldots,k \tag{6.24}$$

şeklinde, y_{0k} ve y_{1k} ise

$$y_{0k} = \frac{m}{2} - \sum_{i=1}^{k-1} y_{0i}$$

$$y_{1k} = \frac{m}{2} - \sum_{i=1}^{k-1} y_{1i}$$

(6.25)

şeklindedir. Hareket testi için istatistik,

$$X = \sum_{i=1}^{k} \left(\frac{(n_{0i} - y_{0i})^2}{y_{0i}} + \frac{(n_{1i} - y_{1i})^2}{y_{1i}} \right) \tag{6.26}$$

şeklindedir. İstatistik, serbestlik derecesi $2k-3$ olan χ^2 - dağılımını izler.

11111 00000 111 000 11 00 11 00 10 10 10 10 altdizisini 5 kez tekrarlayan bir dizi, rasgele olmadığı halde haraket testini kolayca geçer. Babbage'nin k'nın tahmin üzerine yaklaşımı kullanılarak, k,

$$\mathbf{k} = \left\lfloor \log_2 \frac{\mathbf{n}}{10} \right\rfloor = \log_2 16 = 4$$

olarak belirlenir. Bu dizideki hareketlerin sayısı,

$$\mathbf{n}_{01} = 20 \quad \mathbf{n}_{11} = 20$$
$$\mathbf{n}_{02} = 10 \quad \mathbf{n}_{12} = 10$$
$$\mathbf{n}_{03} = 5 \quad \mathbf{n}_{13} = 5$$
$$\mathbf{n}_{04} = 5 \quad \mathbf{n}_{14} = 5$$

şeklinde bulunur. y_{0i} ve y_{1i} değerleri üstte verilen eşitlik yardımıyla,

$$\mathbf{y}_{01} \cong 19.87 \quad \mathbf{y}_{11} \cong 19.87$$
$$\mathbf{y}_{02} \cong 10.06 \quad \mathbf{y}_{12} \cong 10.06$$
$$\mathbf{y}_{03} \cong 5.06 \quad \mathbf{y}_{13} \cong 5.06$$
$$\mathbf{y}_{04} \cong 5.01 \quad \mathbf{y}_{14} \cong 5.01$$

olarak elde edilir. Hesaplanan bu değerlerin ışığında X istatistiği,

$$\mathbf{X} = \sum_{i=1}^{k} \left(\frac{(\mathbf{n}_{0i} - \mathbf{y}_{0i})^2}{\mathbf{y}_{0i}} + \frac{(\mathbf{n}_{1i} - \mathbf{y}_{1i})^2}{\mathbf{y}_{1i}} \right)$$
$$\cong 2\left(\frac{(20-19.87)^2}{19.87} + \frac{(10-10.06)^2}{10.06} + \frac{(5-5.06)^2}{5.06} + \frac{(5-5.01)^2}{5.01} \right)$$
$$\cong 0.004$$

olarak bulunur. $2k-3=5$ serbestlik derecesi olan χ^2 - dağılımından $\alpha = 0.05$ için kritik bölge, 11.07'den büyük değerleri içeren bölge olarak belirlenir. X istatistiğinin bu değerden oldukça lüçük olduğu görülüyor. Böylece dizi rasgele olmadığı halde bu testi kolaylıkla geçer.

6.3.2.5 Özilişki Testi

Bu test, rasgeleliği incelenen dizinin bitleri ile aynı dizinin ötelenmiş şekli arsında bir ilişki olup olmadığını belirlemek amacıyla oluşturulmuştur. Eğer

incelenmek istenen dizi, $S = s_1 s_2 ... s_n$ şeklinde ise ve d büyüklüğünde bir öteleme gerçekleştirilmiş ise, bu, s_i ile s_{i+1} bitlerinin karşılaştırılacağı anlamına gelir.

Ötelenmişlerine eşit olmayan bitlerin sayısı, $\oplus$ XOR işlemini göstermek üzere,

$$A(d) = \sum_{i=1}^{n-d} s_i \oplus s_{i+d} \qquad d = 1,...,\lfloor n/2 \rfloor \tag{6.27}$$

şeklindedir. Rasgele bir dizi çin A(d), (n - d) tane birbirinden bağımsız olarak düzenli bir şekilde dağılmış 0/1 rasgele değişkenlerinin toplamı şeklinde olduğundan, A(d), $d \neq 0$ olmak üzere, ortalaması (n - d)/2 ve varyansı (n - d)/4 olan binom dağılımını izler. Eğer (n - d) büyükse, binom dağılımının yaklaşık olarak normal dağılımı izlediği söylenilebilir, $N\left(\frac{n-d}{2}, \frac{n-d}{4}\right)$. Bu test için, A(d)'nin hem çok ufak değerleri hem de çok büyük değerleri gözlenmemelidir. Bu nedenle iki uçlu test kullanılmalıdır. Normal dağılımda, $\alpha = 0.05$ için kesim noktaları daha öncedende söylendiği gibi, $X_{1-\alpha/2} = 1.96$ ve $X_{\alpha/2} = -1.96$ şeklindedir. Yani

$$\left| \frac{A(d) - \frac{n-d}{2}}{\frac{\sqrt{n-d}}{2}} \right| > 1.96 \tag{6.28}$$

durumunda dizi rededilir. Bu söylenilene özdeş olarak

$$\begin{aligned} A(d) &> \frac{n-d}{2} + 1.96\frac{\sqrt{n-d}}{2} \\ A(d) &< \frac{n-d}{2} - 1.96\frac{\sqrt{n-d}}{2} \end{aligned} \tag{6.29}$$

eşitsizliklerinin gerçekleşmesi halinde dizi reddedilir.

Dizilerin rasgeleliğinin belirlenmesinde kullanılan testler, yalnızca yukarıda verilen beş istatistiksel test değildir. Bu testlerin yanı sıra aralık testi, kupon toplayıcı testi, permütasyon testi, seri korelasyon testi, t'lerin maksimumum testi, ikili türev

testi, değişim noktası testi, çarpışma testi gibi testler de rasgeleliğin belirlenmesinde kullanılır, [48]. Buna rağmen rasgele olmadığı halde istatistiksel testleri geçen dizilerin varlığı, rasgele testleri ölçen başka birtakım testlere ihtiyaç duyulduğunu göstermiştir. Bu nedenle dizilerin öngörülemezliğini ölçen karmaşıklık testleri türetilmiştir.

Rasgeleliğin ölçülmesi amacıyla, teknik bilgilerin artması ve bilgisayar yazılım ve donanımının gelişmesi paralelinde ileride daha pek çok testin türetileceği açıktır.

BÖLÜM 7

KAOTİK SİSTEMLERİN ŞİFRELEMEDE KULLANIMINA İLİŞKİN YAPILAN ÇALIŞMALAR

Bu bölümde kaotik sistemlerin, kriptolojide sanki - rasgele dizi üreteci olarak kullanımına ilişkin yapılan çalışmalar tanıtılacaktır.

Biyolojik popülasyonu modellemek için kullanılan lojistik dönüşümün kaotik özellikler sergilemesinden yararlanarak tek - zamanlı sistemlerle yer değiştirebileceğini söylediği yeni bir genelleştirilmiş dönüşüm türeten Robert A. J. Matthews, [4], ile başlayacak bu çalışmalar, Daniel D. Wheeler'ın küçük çevrimler oluşturduğu için bu dönüşümü eleştirdiği çalışmasıyla, [5], devam edecektir. Ardından Douglas W. Mitchell'in anahtar dizisi üretmek için kaotik dönüşümlerden çok monotonik dönüşümlerin kullanılması gerektiğini savunan ve bu yol ile üretilen anahtar dizilerinin, sınırlı, çevrimsiz ve monotonik olmadığı için kaotik olduğunu iddia ettiği çalışması, [6], verilecektir. Daniel D. Wheeler ve Robert A. J. Matthews'in, Wheeler tarafından küçük çevrimler oluşturduğu için eleştirilen Matthews'in genelleştirilmiş kaotik dönüşümünün, Cray Y - MP ile gerçekleştirilmesiyle bu tür sorunların ortadan kalkacağını gösterdikleri çalışmalarının, [7], ardından, bu ana kadar tek - boyutlu dönüşümlerle sınırlanan kaos kuramına dayalı sistemlerden sanki - rasgele dizi üretme fikri, John M. Carroll, Jeff Verhagen ve Perry T. Wong ile çok - boyutlu sistemlere taşınacaktır, [8].

7.1 Kaotik Bir Şifreleme Algoritmasının Türetilmesi

Belirli koşullar altında, lojistik dönüşüm rasgele sayılardan oluşan kaotik diziler üretebilir. Matthews çalışmasında, bu tür bir kaotik dönüşüm türettti. Ayrıca çalışmasında, bu dönüşümün, kriptografik uygulamalar için uygun olabileceğini ve tek zamanlı sistemlerle yer değiştirilebileceğini ifade etti,[4]. Matthews'in bu çalışması şu şekilde özetlenebilir.

Lojistik dönüşüm, ayar parametrelerinin uygun seçimi ile, rasgele sayılardan oluşan diziler üretebilir. Kaotik fonksiyonların ayar parametrelerindeki ufak değişimler, birbirinden tamamen farklı dizilerin üretilmesine neden olur. Dolayısıyla kaotik dönüşümler, bir anahtarda olması gereken karakteristiklere sahiptirler. Kaotik dönüşümler, rasgele sayılar üretebilme özelliğinden dolayı, şifreleme biliminde oldukça ("cryptology") ilgi uyandırmıştır. Gerçek mesajın, rasgele sayılardan oluşan bir anahtar dizisi ile modüler toplam vasıtasıyla şifrelenmesi anlamına gelen ve kırılamaz olduğu tanıtlanabilir tek zamanlı sistemler, rasgele sayılardan oluşan dizilerin güvenli dağıtılmasından kaynaklanan güçlüklerden dolayı yaygın bir kullanım alanı kazanamamıştır.

Rasgele anahtar dizileri üretmek için kaotik bir dönüşüm kullanılırsa, ayar parametrelerinin uygun değerleriyle istenilen yerde ve zamanda anahtar dizisi üretilebilir ve böylece uzun rasgele dizilerin güvenli bir şekilde dağıtılması yerine, yalnızca bu parametre değerlerinin güvenli şekilde dağıtılması yeterli olur.

Tek boyutlu doğrusal olmayan dönüşümlerin en çok çalışılanı lojistik dönüşümdür,

$$\mathbf{x} \mapsto \mathbf{g}(\mathbf{x}) = \lambda \mathbf{x}(1-\mathbf{x}), \quad 0 < \mathbf{x} < 1. \tag{7.1}$$

Ayar parametreleri λ ve x_0 olan bu dönüşüm, biyolojik popülasyonu modellemek için kullanılır.

Lojistik dönüşüm, λ parametresinin 3'ten büyük olmayan değerleri için dikkat çekici bir davranışta bulunmaz: x_0 başlangıç değerinin $g(\cdot)$ dönüşümü altındaki tekrarlamaları ("iterations"), tek bir değere yakınsar. Kalıcı durum tek bir noktadan oluşur. Buna rağmen, λ parametresinin 3'ten büyük değerleri için oluşan diziler,

periyodiklik gösterir. Başlangıçta iki farklı değer arasında salınım yapan dizi, λ arttıkça 4, 8, 16, 32,... değerleri arasında salınım yapar. Fakat λ'nın 3.6'dan büyük bazı değerleri için, dönüşümün ürettiği diziler tamamen rasgele görünüşlüdür. λ, 3.8 değerini aştığında ise, periyot ikilenmesi tamamen kaybolur ve verilen bir rasgele x_0 başlangıç değeri için üretilen diziler rasgele görünüşlü olur.

7.1.1 Lojistik Dönüşümün Kaotik Davranışı

Bir dönüşümün kaotik davranışını anlamaya ilişkin anahtar kavram, dönüşümün sabit noktalarıdır. $\mathbf{x}_{s_1} = 0$, lojistik dönüşümün aşikar bir sabit noktasıdır. Diğer sabit nokta,

$$\mathbf{x}_{s_2} = \lambda \mathbf{x}_{s_2}(1-\mathbf{x}_{s_2}) \tag{7.2}$$

denkleminin çözülmesiyle,

$$\mathbf{x}_{s_2} = 1-1/\lambda \tag{7.3}$$

olarak bulunur. Dönüşümün sabit noktaları, λ'nın belli değerleri için kararsız olur. Yani başlangıç değerinin bu kararsız sabit noktalara ne derece yakın olduğuna bakılmaksızın; bu başlangıç değerinin g(·) dönüşümü altındaki tekrarlamaları, kararsız sabit noktalardan uzaklaşırlar. Feigenbaum, dönüşümün sabit noktadaki eğiminin mutlak değeri 1'den büyükse; bu sabit noktanın kararsız olduğunu gösterdi,[49].

g(·) dönüşümünün herhangibir x noktasındaki eğimi,

$$\frac{\mathbf{dg(x)}}{\mathbf{dx}} = \lambda(1-2\mathbf{x}) \tag{7.4}$$

şeklindedir. Ohalde $0 < \lambda < 1$ olduğunda, $\mathbf{x}_{s_1} = 0$ sabit noktası kararlıdır.

$\mathbf{x}_{s_2} = 1-1/\lambda$ sabit noktasının kararlılığı da λ'ya bağlıdır,

$$\frac{dg(x_{s_2})}{dx} = 2 - \lambda. \tag{7.5}$$

Böylece $\lambda > 3$ olduğunda, lojistik dönüşümün sabit noktalarının her ikiside kararsızdır. $\lambda > 3$ olduğunda herhangibir değere yakınsamayan g(·) dönüşümü, periyot ikilenmesi ve kaos sergilemeye başlar. Buna rağmen, $\lambda > 3$ olduğunda; (7.1) denkleminin analitik tarifi oldukça karmaşıktır, [50].

Feigenbaum, en azından bazı periyot katlanmalarının bu λ değerinden sonra sonsuz uzunlukta olacağı, λ_∞ ile gösterilen bir λ değeri hesaplamak için yaklaşık bir teknik geliştirdi.

Fakat λ_∞'un üzerindeki λ değerleri için bile, kararlılık pencereleri vardır. Bu pencerelerde üretilen diziler periyodik olacağından, anahtar dizisi olarak güvenli değildirler.

7.1.2 Genelleştirilmiş Kaotik Dönüşüm

Matthews, kriptolojik uygulamalar için, x ve λ değişkenlerine sahip olan lojistik dönüşüm yerine; $(\lambda, \alpha, \beta, \mathbf{x})$ değişkenleriyle tanımlanan genelleştirilmiş lojistik dönüşümü önerdi,

$$\mathbf{x} \mapsto \mathbf{g}(\lambda, \alpha, \beta, \mathbf{x}) = \lambda \mathbf{x}(\alpha - \mathbf{x})^{\beta}. \tag{7.6}$$

Dört değişken içeren bu dönüşüm ilk koşullara aşırı duyarlı olduğundan, oldukça fazla sayıda farklı rasgele dizi üretebilir. Dönüşümün rasgele sayılar üretebilmesini garantilemek amacıyla ayar parametreleri üzerine birtakım koşullar türetildi.

İlk olarak dönüşüm, z. dereceden türetilebili maksimuma sahip olmalıdır. Öyleki küçük $|\mathbf{x}|$'ler için,

$$\mathbf{g}(0) - \mathbf{g}(\mathbf{x}) \propto |\mathbf{x}^{\mathbf{z}}|, \quad \mathbf{z} \geq 1 \tag{7.7}$$

sağlanır,[50]. Bunu garantilemenin en iyi yolu, λ, α, ve β değerlerinin pozitif olmasıdır.

İkinci olaraksa, dönüşüm periyodik olmayan diziler üretebilmelidir. Feigenbaum, çalışmasında: dönüşümün kaotik davranıp davranamayacağını belirlemek için en önemli özelliğin; dönüşümün, aşikar olmayan sabit noktadaki eğiminin olduğunu söyledi.

Feigenbaum, en azından periyodik olmayan dizilerin görünmeye başladığı λ_∞ değerinin, dönüşümün aşikar olmayan sabit noktadaki eğiminin -1.6012 değerinden büyük olması gerektiği koşulundan hesaplanabileceğini gösterdi. Buradaki genel ilke, sabit noktadaki eğim değeri ne kadar büyükse; dönüşümün davranışının o derece garip olduğudur. g(·) dönüşümünün herhangibir x noktasındaki eğimi,

$$\frac{dg(x)}{dx} = \lambda(\alpha - x)^{\beta} - \lambda x \beta (\alpha - x)^{\beta - 1} \tag{7.8}$$

şeklindedir. g(·)'nin aşikar olmayan sabit noktası, $g(x_s) = x_s$ denkleminin çözülmesiyle,

$$x_s = \left(\alpha - 1/\lambda^{1/\beta}\right) \tag{7.9}$$

şeklinde bulunur ve bu sabit noktadaki eğim,

$$\frac{dg(x_s)}{dx} = 1 + \beta - \alpha\,\beta\,\lambda^{1/\beta} \tag{7.10}$$

şeklindedir. Dönüşümü kaotik davranışa zorlamak için, (7.5) ifadesinin sağ tarafi, maksimize edilmelidir. Eğer x_s 'in değeri, $\frac{dg(x)}{dx} = 0$ ifadesini sağlayan x değerinden büyükse, $\frac{dg(x_s)}{dx}$ mümkün olduğunca negatif yapılmalıdır.

$\mathbf{x} = \alpha/(1+\beta)$ olduğunda, $\dfrac{\mathbf{dg(x)}}{\mathbf{dx}} = 0$ dır. Yani bu x değeri, g(·)'yı maksimum yapan değerdir. g(·) dönüşümünün maksimum değeri g_m ile gösterilsin,

$$\mathbf{g_m} = \lambda\left[\alpha^{\beta+1}/(\beta+1)\right](1+1/\beta)^{-\beta}. \tag{7.11}$$

Bu ifadenin yeniden düzenlenmesi ile λ,

$$\lambda = (\beta+1)\left[\mathbf{g_m}/\alpha^{\beta+1}\right](1+1/\beta)^{\beta} \tag{7.12}$$

şeklinde bulunur. $\mathbf{x_s} > \alpha/(1+\beta)$ yani $\alpha - 1/\lambda^{1/\beta} > \alpha/(1+\beta)$ eşitsizliği sağlanırsa; x_s, dönüşümün sağ bölümündedir. Bu eşitsizlikte, λ'nın yerine yazılmasıyla, sabit noktanın dönüşümün sağ tarafında olmasını garantileyen kriter bulunur,

$$\mathbf{g_m} > \alpha/(1+\beta). \tag{7.13}$$

Gerçel değerli g(·) için, α en azından g_m kadar olmalıdır,

$$\alpha = \mathbf{k g_m}, \quad \mathbf{k} \geq 1. \tag{7.14}$$

Bu ifadenin (7.13)'de yazılması ile; $\beta > 0$ eşitsizliğinin sağlanmasının, sabit noktanın dönüşümün sağ tarafında olması için yeterli olacağı bulunur. Buradaki hedef, dömüşümün sabit noktadaki eğimini mümkün olduğunca negatif yapılmasıdır. Bu koşulun sağlanması için, aşağıda yazılan ifadeden görüldüğü gibi, $\mathbf{g_m} >> \alpha$ olmalıdır,

$$\frac{\mathbf{dg(x_s)}}{\mathbf{dx}} = (1+\beta) - \left(\mathbf{g_m}/\alpha\right)^{1/\beta}(1+\beta)^{1+1/\beta}. \tag{7.15}$$

Fakat g(·) dönüşümünün gerçel değerli olması için, $\alpha \geq \mathbf{g_m}$ olmalıdır. Dolayısıyla en iyi çözüm,

$$\alpha = g_m \tag{7.16}$$

eşitliğinin sağlanmasıdır. (7.16)'in, (7.12)'de yerine yazılmasıyla elde edilen λ, (7.6)'da yerine yazıldığında; sabit noktada eğimi en büyük olan g(·) dönüşümü, herhangibir x değişkeni için aşağıdaki gibi verilir,

$$g(x) = \left[(\beta+1)/g_m^{\beta}\right](1+1/\beta)^{\beta} x(g_m - x)^{\beta}. \tag{7.17}$$

Kriptolojik uygulamalar için, g_m üzerine bir üst sınır verilmelidir. Böyle yapılmadığı takdirde, β büyük olduğunda g(·) dönüşümünün herhangibir x değeri için küçük olma tehlikesi vardır. Böyle bir tehlike gerçekleştiğinde, dönüşüm tarafından üretilen anahtar dizisinde sıfırlar baş gösterir. Bu tür bir dizinin şifrelemede hiçbir güvenlik sağlayamayacağı açıktır. Bu risk, $g_m = 1$ alınarak ortadan kaldırılabilir. $g_m = 1$ alındığında (7.16)'den dolayı, $\alpha = 1$ olur. Bu değerler ışığında, kriptolojik uygulamalar için uygun bir dönüşüm elde edilir,

$$g(x) = (\beta+1)(1+1/\beta)^{\beta} x(1-x)^{\beta} \tag{7.18}$$

α ve λ'nın gerçekte ayar parametreleri olmadığı görülüyor. Bu dönüşüm için başlangıç değerleri, 0 ile 1 arasında alınmalıdır.

β üzerindeki kısıtlamalar, sabit noktada kaosun görünmeye başladığı eğim yardımıyla belirlenebilir. Daha öncede söylenildiği gibi, bu değer -1.6012 değerinden büyük olmalıdır.

β'nın altsınırı, Ulam ve Von Neumann'ın araştırmaları kullanılarak belirlenebilir, [51]. Onlar, $x \mapsto f(x) = 4x(1-x)$ dönüşümünün ürettiği dizilerin, f(·)'in rasgele sayı üreteci olarak kullanımına olanak sağlayan uygun istatistiksel özelliklere sahip olduğunu buldular. Bu dönüşümün aşikar olmayan sabit noktası, 3/4 tür. Dönüşümün bu noktadaki eğimi, -2 dir. Bu değer $dg(x_s)/dx$ için kullanılırsa, $\beta \geq 1$ altsınırı bulunur. Yani bu koşul sağlandığında, g(·) dönüşümünün aşikar olmayan x_s sabit noktasındaki eğimi en az -2 dir.

β'nın üst sınırını hesaplamada kullanılan makinenin özellikleri belirler. Makinenin, en fazla D dijit gösterebildiği ve değer bölgesinin $10^{-n} \le x \le 10^{n}$ olduğu varsayılsın. Dönüşümün herhangibir değeri, $(1-x)^{\beta} \le 10^{-n}$ olacak şekilde 1'e çok yakınsa; makineden "bölgenin dışında" diye bir mesaj gelir. bu durumdan kurtulabilmek için, β üzerine,

$$\left(10^{-D}\right)^{\beta} > 10^{-n} \tag{7.19}$$

olacak şekilde bir üst sınır getirilir. Bu eşitsizlik, β'nın üst sınırının **n/D** olduğunu söyler.

Burada, $n = 38$ ve $D = 8$ olduğu Amstrad PCW 8256 için, β üzerine aşağıdaki gibi bir sınır verildi,

$$g(x) = (\beta+1)(1+1/\beta)^{\beta} x(1-x)^{\beta} \qquad 1 \le \beta \le 4. \tag{7.20}$$

7.1.3 Kaotik Dönüşümün Kriptografide Kullanımı

Kriptografik uygulamalar için, genelleştirilmiş kaotik dönüşümün nasıl kullanılabileceği güçlü özelliklere sahip olmayan bir cihaz kullanarak aşağıdaki gibi tarif edilebilir.

İlk olarak kullanılan makinenin özelliğine bağlı olan D dijite göre g(·)'nin değer bölgesi, x_0 ve β belirlenir. β belirlenirken, yukarıda açıklanan alt ve üst sınırlar dikkate alınmalıdır. β'nın tamsayı değerinde olmama zorunluğunun olmaması önemlidir. Gerçekte kaotik dönüşümün güzelliği, birbirine oldukça yakın tamsayı olmayan β değerleri için birbirinden tamamen farklı diziler üretebilmesidir. Nümerik simülasyonlar, uygun anahtar dizilerinin, β'nın (D-2) tane dijitle belirlenmesi durumunda üretilebileceğini göstermektedir. Dönüşümün ürettiği dizideki sayıların en düşük anlamlı iki dijiti, uygun bir tabana çevrilerek açık mesaja eklenir ve bunun sonucunda şifreli mesaj elde edilir. Şifreli mesajı açmak için gerekli olan, g(·)

dönüşümünün aynı β ve x_0 değerleri için çalıştırılıp; yukarıda verilen aynı metotla elde edilen sonucun şifreli mesajdan çıkarılmasıdır.

$\beta = 2.53$ ve $x_0 = 0.45$ değerleri kullanılarak dönüşümün ilk 5 değeri,

0.812980077
0.095875139
0.609135158
0.463757308
0.785823223

şeklinde verilir. Bu değerler yuvarlatma ve benzeri hatalardan dolayı makineden makineye değişir. Bu nedenle iletişim ağındaki her üyenin aynı makineyi kullanması gerekir.

Yukarıda verilen 5 değerin son iki dijitinin, 25'e göre modülü alındığında 2, 14, 8, 8, ve 23 dizisi elde edilir. Rotor devresi kullanılarak, [52], bu dizi CHAOS yalın mesajına eklenirse, EVIWQ elde edilir. Dönüşümün parametrelerine olan aşırı duyarlığının vurgulanması amacıyla, $\beta = 2.5300001$ alınarak yukardaki işlem tekrarlandığında, 23, 1, 3, 12, 23 dizisi bulunur.

Şifreli mesajı açmak için, β ve x_0 bilgisi tamamen bilinmelidir. $g(\cdot)$ dönüşümünün ürettiği dizilerin sıfırlardan oluşma tehlikesi vardır. Bu durum, dönüşümün değeri 10^{-D}'den küçükken meydena gelir.

Örneğin D = 8 olsun ve herhangibir dönüşüm değeri 10^{-10} prezisyonla oluşsun. Bu dönüşüm değeri birkaç tekrarlamadan sonra 00'lardan oluşan bir anahtar dizisi üretir. Bu problemden, 10^{-10} prezisyonla meydana gelen dönüşüm değerinin logaritması ardından mutlak değeri alınarak; ufak bir sayıyla, örneğin 0.001, ile çarpılarak yeni başlangıç değerinin belirlenmesi ile kurtulabilinir.

7.1.4 Kaotik Dönüşümün Güvenliğinin Tahmini Üst Sınırı

Kaotik dönüşümün tahmini üst sınırı, bilinmeyen metin üzerinde uygulanan ayrıntılı anahtar araştırma atağında ("exhaustive key search attack") denenebilecek olası tüm anahtarların sayısı ile verilir.

Dönüşüm, β ve x_0 değerlerine aşırı duyarlıdır. Böylece anahtar dizilerinin yaklaşık olarak sayısı, farklı anahtar dizilerinin üretilmesine neden olan bu parametre değerlerinin sayısının çarpımı ile verilir. Tamamen farklı dizilerin üretilmesi için en iyi çözümün, β ve x_0 parametre değerlerinin çıkışdan iki az dijitle belirlenmesi olduğu, önceden vurgulanmışdı. Bu durumda eğer makine D dijitlik göstergeye sahipse, anahtar sayısı,

$$\mathbf{K} = 10^{2D-4} \tag{7.21}$$

şeklindedir. Tüm anahtarların eşit olasılıklı olduğu rasgele bir şifreleme modelinde, anahtar entropisi,

$$2^{H(K)} = 10^{(2D-4)} \tag{7.22}$$

ifadesini sağlar, [53]. Bu ifade 10 tabanında yazıldığında,

$$\mathbf{H(K)} = 6.6(\mathbf{D}-2) \tag{7.23}$$

bulunur. Bu, kaotik şifreleme sisteminin karmaşıklık fonksiyonunun zamanla üstel olduğu anlamına gelir, ([53, s 30]). Böylece şifreye yapılacak kaba kuvvet atağı ("brute force attack"), D dijit sayısının arttırılmasıyla önlenebilir.

Şifreyi kırmak için kullanılacak makinenin saniyede R işlem yapabildiği varsayılsın. Bu atak türü kullanılarak şiyreyi kırma süresi yıl olarak Y ile verilir,

$$\mathbf{Y} = 2^{6.6(D-2)} / 3.10^7 \mathbf{R}. \tag{7.24}$$

Her birinin saniyede yaptığı işlem sayısı 2.10^6 olan 1000 tane T - 800 Inmos Transputers içeren bir makinede, 12 göstergelik kaotik bir şifreleme sistemini bu atak türü ile kırmak için gerekli süre, yaklaşık 1000 yıldır. Dahası, Y, D dijit sayısına aşırı duyarlıdır. Her fazla dijit sayısı, Y değerini 100 kat arttırır.

Bu etkileyici sonuca rağmen, kaotik sistemi kırmak için kullanılacak tek atak türü, bu değildir. Doğrusal modüler kongruens üreteçlerinden elde edilen dizilere dayanan şifreler, bu dizilerin analizi ve ayar parametrelerinin türetilmesi ile kırılabilir. g(·)'nin doğrusal olmayan yapısı, bu atak türüne karşı daha dayanıklı olabilir. Buna rağmen dayanıklılığın kontrol edilebilmesi için, **g(·)** dönüşümünün yapısının dikkatle çalışılmasına ihtiyaç vardır.

7.1.5 Sonuç

Sonuç olarak Matthews bu çalışmasında, en azından prensip olarak tek zamanlı sistemlerle yerdeğiştirebilecek kaotik bir şifreleme algoritması türetti. Yazar, sözü edilen çalışmasında bu algoritma için güçlü özelliklerle donatılmış makinelere ihtiyaç duyulmadığını da belirtti. Bunlara ilave olarak, Matthews, istenilen yerde ve zamanda rasgele dizi üretebilme gücünün, tek zamanlı sistemlerde uzun rasgele dizileri güvenli bir şekilde dağıtmaya duyulan gereksinimin üstesinden gelebileceğini de yine bu çalışmasında ifade etti.

7.2 Kaotik Şifreleme Sistemlerindeki Problemler

Wheeler, Matthews'in [4]'te kaos kuramına dayalı olarak oluşturduğu ve tek zamanlı sistemlerle yerdeğiştirebileceğini söylediği kaotik şifreleme algoritmasının, sayısal bilgisayarlarla gerçekleştirildiğinde birbirini tekrar eden çevrimler oluşturduğunu söyledi. Oluşan bu çevrimlerin tahmin edilemezlik ilkesini dışladığını ve oldukça kısa uzunlukta gerçekleştiğini belirtti. Kriptoanalizstin bu kısa çevrim uzunluklarından yararlanarak şifreli mesajı kırabileceği tekniklerin varolduğunu ve bu nedenden dolayı, Matthews'in önerdiği kaotik dönüşümün kriptografik uygulamalar için uygun olamayacağını söyleyen Wheeler'ın [5]'deki bu çalışması, şu şekilde özetlenebilir.

Matthews'in tek zamanlı sistemlerle yerdeğiştirebileceğini önerdiği kriptografik metodun daha dikkatle incelenmesi gerekir. O, önerdiği metodun yaralanamaz olduğuna dair etraflı bir çalışma yapmadı.

Açık mesajın şifrelendiği gerçek rasgele anahtara bağlı olan tek zamanlı sistemler, tamamen güvenli olan tek kriptografik sistemdir. Bu sistemin güvenliği, açık mesajın gerçek rasgele bir anahtarla aynı uzunlukta bir şifreli mesaja çevrilmesinden ileri gelir. Tek zamanlı sistemlerin kullanışlılığını sınırlayan iki işlemsel ihtiyaç şu şekilde ifade edilebilir. İlk olarak bu sistemler, uzun rasgele diziler üretebilen kaynaklara ihtiyaç duyar. İkinci olaraksa bu uzun dizilerin tam kopyaları, iletişim ağını kullanan her kullanıcıya dağıtılmalıdır.

Pek çok bilgisayar programlama dilinde, verilen bir başlangıç değerinden rasgele sayılar üretebilen dönüşümlerin varolması, ilk bakışta sözü edilen her iki problemi çözebilirmiş gibi bir izlenim uyandırmaktadır. Çünkü bu tür bir durumda, uzun rasgele dizilerin iletişim ağını paylaşan her kullanıcıya dağıtılması yerine, başlangıç koşulunun dağıtılması yeterli olur.

Doğrusal kongruens ve ötelemeli yazmaç dizilerine dayanan rasgele sayı üreteçleri, [54], [55]'te detayla olarak incelenmiştir. Bu algoritmaların ürettiği sayılar, sanki - rasgele sayılar olarak adlandırılır. Çünkü rasgele görünüşlü olan bu sayılar, deterministik algoritmalarla üretilmelerinden kaynaklanan birbirini tekrar eden çevrimlere rağmen; rasgelelik için geliştirilmiş istatistiksel testleri geçer.

Bu dizileri üreten dönüşümler, bazı kriptografik uygulamalar için kullanışlıdır. Fakat bu dönüşümler, tek zamanlı sistemler ve benzerlerinde rasgele sayı üreten bir kaynak olarak kullanılamaz. Bu tür diziler, muhtemel kelimeleri kullanma metoduna göre oldukça hassastır. Olası kelimenin, şifreli mesajın özel bir yerinde olduğu varsayılır. Daha sonra olası kelimeyi, şifreli mesajdaki şekle getiren anahtar rakamları belirlenir. Bu kısa diziden, üretecin nerede çevrime girdiği belirlenebilir. Bu ise, tüm anahtar dizisinin üretilebilmesini olanaklı hale getirir. Bu işlem sonunda, açık mesaj elde edilemez ise; olası kalimenin başka bir özel yerde olduğu varsayılır ve yukarıda verilen metod aynen tekrar edilir.

7.2.1 Matthews'in Genelleştirdiği Kaotik Dönüşüm

Kaos kuramı, basit doğrusal olmayan denklemlerin belli parametre değerleri için kaotik diziler üretebileceğini gösterdi. Matthews, belli bir değer bölgesindeki parametre değerleri için kaotik diziler üreten bir dönüşüm önerdi,

$$g(x) = (\beta + 1)(1 + 1/\beta)^{\beta} x(1-x)^{\beta}. \tag{7.25}$$

x_0 başlangıç değerinin ve β parametresinin değer bölgesi,

$$1 \leq \beta \leq 4 \tag{7.26}$$

$$0 < x_0 < 1 \tag{7.27}$$

şeklindedir. x_0 ve β, kriptografik anahtarlardır. Bu ikilini her bir çifti, $g(\cdot)$ dönüşümü altında tek bir dizi üretir. Matthews, örnek olarak $\beta = 2.53$ ve $x_0 = 0.45$ değerlerini kullandı. Bu değerlerle, dönüşüm herhangibir x değeri için,

$$g(x) = 8.198790355x(1-x)^{2.53} \tag{7.28}$$

şeklindedir. Matthews, $x_0 = 0.45$ başlangıç değeri için ilk 5 iterasyon değerini verdi, 0.812980077, 0.095875139, 0.609135158, 0.463757308. O, bu sayıların en sağdaki iki rakamının anahtar dizisi olarak kullanılmasıyla; genelleştirdiği kaotik dönüşümün tek zamanlı sistemlerle yerdeğiştirebileceğini ifade etti.

Matthews, bu dönüşümün tekrar etmeyen diziler üreteceğini iddia etti. Gerçekte önerdiği dönüşüm, bir hesap makinesinde veya bir bilgisayarda gerçekleştirildiğinde; tekrarlı diziler üretmelidir. Gerçel sayılar için sabit uzunluklu gösterim, sonlu sayıda farklı değere neden olur. Dönüşümün ürettiği bir değer, yuvarlatma hatalarından dolayı, önceden varolan bir değere yuvarlandığında; dönüşüm, önceki diziyle tamamen aynı olan bir dizi üretir. Yani, kaotik dönüşümün sayısal bilgisayarlarla gerçeklenmesi durumunda; diğer sanki - rasgele dizi üreteçlerine benzer şekilde, kaotik dönüşüm de sonlu uzunluklu bir çevrim oluşturacaktır.

Buna rağmen bilgisayarlarla, tek prezisyonda bile, ifade edilebilecek oldukça çok sayıda gerçel sayı vardır. Dolayısıyla bu dönüşümle, gönderilecek tüm mesajların uzunluğundan daha uzun bir çevrim uzunluğuna sahip diziler üretilebilir. Fakat kaotik dönüşümlere dayanan sistemlerin, kriptografik uygulamalar için kullanılmadan önce çevrim uzunlukları belirlenmelidir.

7.2.2 Ön Çalışmalar

Kaotik sistemlerin özelliklerinden biri, ilk koşullara olan aşırı duyarlıktır. Bu, kelebek etkisi olarak adlandırılır: Pekinde bir kelebeğin uçarak meydana getirdiği havadaki dakikalık bir değişim, bir ay sonra New York'taki hava durumunu etkileyebilir. Gleick'in raporuna göre, Edward Lorenz bu etkiyi, hava desenlerinin bilgisayarla simülasyonu esnasında meydana gelen yuvarlatma hatalarının, hava durumunu tamamen değiştirdiğini anladığında fark etti,[1].

Yani çevrim uzunluğunun, gerçel sayıların bilgisayarda ifade ediliş biçimine ve bilgisayarın nümerik sonuçları ne çeşit bir metoda göre yuvarladığına kritik bir şekilde bağlı olduğu beklenir. Bu çalışmadaki tüm incelemeler, 8087 matematik işlemci ile donatılmış bir MS - DOS bilgisayarında gerçekleştirildi. Programlar, Turbo Pascal 5.0'da yazıldı. Değişkenler ise, ikili prezisyonla (64 bit) saklandı.

Farklı seviyedeki doğrulukların etkisini inceleyebilmek amacıyla, yuvarlatılmış sayıları daha düşük prezisyona çevirmek için iki farklı metod verildi. İlk yöntemde, sayıları onlu rakamlardan oluşan değişkenlere çeviren Pascal fonksiyonu kullanıldı. Sayıyı üstel formdaki ASCII dizisine çeviren bu metod, mantis kısmın istenen sayıdaki rakamdan sonra keser ve bu şekilde elde edilen sayıyı tekrar ikili doğruluklu gerçel biçime çevirir. Örneğin yuvarlatma kümesi anlamlı 3 rakamdan oluşacaksa; 0.095875139 sayısı, "9.5875139E-0002" dizisine çevrilir, ardından "9.58E-0002" dizisine kesilir ve daha sonra ikili prezisyonlu 0.095800000 gerçel sayı biçimine çevrilir. İkinci yuvarlatma yöntemi, sayının geçici olarak tek prezisyonlu gerçel değişken olarak saklanmasıdır.

Matthews, oldukça küçük değerler sonucunda oluşturulan dizilerle ilgiliydi. 10 rakamlık bir sayının en sağdaki iki rakamının alınması bile, sıfırlardan oluşan dizilere neden oldu. Matthews, bu probleme neden olacak kadar küçük x değerleri

oluştuğunda; logaritma dönüşümünü önerdi. En iyi seçenek, sayıların normalize üstel gösterimindeki en sağdaki iki rakamının alınmasıdır. x değeri çok ufak olduğunda, sayının üstel kısım negatif olacaktır; fakat kriptografik anahtar dizisi olarak, artık sıfırlara neden olmayan anlamlı rakamlar alınacaktır.

Daha ciddi problem, dönüşüm sonucunda çıkan değerlerin 1'e çok yakın olması durumunda yuvarlatma değerinin 1 olmasıdır. Çünkü bu durumda dönüşüm, (1-x) ifadesinden dolayı tam olarak sıfır değerini üretir.

Bu nedenlerden dolayı, sayıyı en yakın sayıya yuvarlayan fonksiyonlar yerine; yukarıda, istenilen rakamda sayıyı kesen Paskal fonksiyonu tanımlandı. Matthews bu güçlükten söz etmediğine göre, onun kaotik dönüşümü gerçekleştirdiği makinenin kesme yöntemini kullanıyor olması mümkündür.

Aynı problem, tek doğruluklu gerçel değişkenler için de meydana gelir. Turbo Pascal, ikili prezisyonlu bir sayıyı tek prezisyonlu bir sayı şeklinde saklamak için; bu sayıyı, tek prezisyonla ifade edilebilecek en yakın sayıya yuvarlar . Bu şekildeki sınırlı prezisyon kullanıldığında; dönüşümün verdiği 1.0 değerini kontrol eden bir blok, kaotik dönüşüm algoritmasına ilave edilmelidir. 1.0 değeri meydana geldiğinde, önceki x değeri 0.9 ile çarpılır ve bu işlem sonucunda elde edilen sayı için dönüşüm değeri yeniden hesaplanır.

7.2.3 Çevrim Uzunluklarıyla İlgili Özellikler

İlk olarak, Matthews'in kullandığı örnek parametre değerleri için prezisyonun anlamlı üç rakamla sınırlandığı durumda çevrim uzunluğu kontrol edildi. Bu işlem sonucunda elde edilen

0.8120 0.0970 0.6140 0.4520 0.8090
0.1000 0.6280 0.4210 0.8660 0.0439

dizi değerleri, birbirini tekrar eden çevrimler içeren aşağıdaki dizi tarafından takip edilir:

0.32100 0.98800000 0.00011100 0.0009090 0.007430
0.05970 0.41800000 0.87100000 0.0401000 0.296000

0.99800 0.00000121 0.00000992 0.0000813 0.000666
0.00545 0.04400000 0.32100000 ....

Sonuçlardan, dönüşümün 10 kez tekrarından sonra, değerlerin birbirini tekrar ettiği, çevrim uzunluğunun 17 olduğu ve dizinin, Matthews'in 9 anlamlı rakamla hesapladığı diziden uzaklaştığı görüldü. Yuvarlatma işlemindeki değişimler, ktiptografik diziyi tamamen değiştirebilir.

Prezisyonun 8 anlamlı rakamla sınırlandığı hesaplama ile üretilen dizilerin çevrim uzunlukları belirlendi. Çevrimler, sanki - rasgele dizi üreteçlerinin kendilerine karşı iki kat hızla çalıştırılmalarıyla belirlenir. Bir dönüşümün ürettiği dizinin çevrimini ortaya çıkaran ve bu çevrimin uzunluğunu ölçen Pascal kodu,

```
repeat
        Xn := G(Xn);
        Yn := G(Yn);
        Yn := G(Yn)
until  Xn = Yn;
CycleLength := 0;
repeat
        CycleLength := CycleLength + 1;
        Xn := G(Xn)
until  Xn = Yn;
```

şeklinde verilir.

Tablo 7.1, β ve x_0 değerlerinin 100 farklı değeri için elde edilen çevrim uzunluklarını gösterir. Tablo 7.1, Matthews'in kullandığı parametre değerlerinide içerir ($\beta = 2.53$ ve $x_0 = 0.45$). Bu parametre değerleri için, kaotik dönüşüm kendini, 13511 tekrarlamadan sonra tekrar eden bir çevrime girdi.

Tablo 7.1, çevrim uzunluğunun β tarafından belirlendiği fikrini verir. Her β değeri için, kaotik dönüşümün girebileceği küçük bir çevrim kümesi vardır. x_0 değeri, bu çevrimlerden hangisine girileceğini belirler.

Tablo 7.1. Prezisyonun 8 anlamlı rakamla sınırlandığı durumda elde edilen çevrim uzunlukları.

$\beta =$	1.03	1.33	1.63	1.93	2.23	2.53	2.83	3.13	3.43	3.73
x_0										
0.05	3116	15838	2982	2177	2136	13511	16456	11140	4720	1049
0.15	3116	15838	5357	2177	2136	13511	16456	11140	4720	1049
0.25	3116	15838	2982	2177	2136	13511	16456	11140	4720	1049
0.35	3116	15838	2982	2177	2136	13511	16456	11140	4720	1049
0.45	3116	15838	2982	2177	2136	13511	16456	11140	4720	1049
0.55	3116	15838	2982	2495	2136	13511	16456	11140	4720	1049
0.65	3116	15838	2982	2177	2136	13511	16456	11140	4720	1049
0.75	3116	15838	2982	2495	2136	1451	16456	11140	4720	1049
0.85	3116	15838	2982	2177	2136	13511	16456	11140	4720	1049
0.95	3116	15838	2982	2177	2136	13511	16456	11140	4720	1049

8 anlamlı rakamlı hesaplama ile belirlenen çevrim uzunlukları, ciddi kriptografik uygulamalar için kullanılamayacak kadar kısadır. Daha kısa çevrim uzunluklarının olup olamayacağını bilmek için, bir yol yoktur. Burada yalnızca 100 farklı parametre değeri için çevrim uzunlukları kontrol edildi. Belki araştırma daha fazla β ve x_0 değerleri için yapılsaydı, çevrim uzunluğu 10 olan diziler bile bulunabilirdi. Böyle bir çevrim gerçek kullanımda meydana geldiğinde, şifreli mesaj bilgisayar kullanımına ihtiyaç duymayan kağıt ve kalem yöntemleriyle kolayca kırılabilir.

Tablo 7.1'de görülen oldukça kısa çevrim uzunlukları, prezisyonun 8 rakamla sınırlandırılmasından kaynaklanan yapay uzunluklar olabilirdi. Tablo 7.2, hesaplamaların Turbo Pascalda tek prezisyonlu gerçel değişkenlerle yapılması durumundaki çevrim uzunluklarını gösterir. Tablo 7.2'de elde edilen ortalama çevrim uzunluğu 2993 tür. Bu değer, Tablo 7.1'deki ortalama çevrim uzunluğu 7181'den bile daha küçüktür. Farklı yuvarlatma hataları için, farklı çevrim uzunlukları bulundu; fakat kısa çevrim uzunlukları, kesme metodunun bir yapısı değildir.

uygun olmadığını söyledi. O, gerçel sayıların sonlu gösteriminden kaynaklanan yuvarlatma hatalarının, dönüşümün kısa çevrimlere girmesine neden olduğunu belirtti.

Mesaj uzunluğu, çevrim uzunluğunun bir çarpanı ise, mesaj vigeneré ile kırılabilir. Kısa çevrim uzunluklarının, kriptoanalizstlere mesajı kırmak için açık kapılar bıraktığı bilinmektedir. Çevrim uzunluğunun bilinmesi, araştırılması gereken değer uzayını önemli derecede azaltacağı mümkün gibi görünmektedir. Ötemeli yazıcılar ile oluşturulan dizilerde bu bilginin kullanılarak dizilerin yeniden inşa edilebilmesi mümkün iken, kaotik dönüşümlerin bu yapısal güçsüzlüğü içermeyeceği daha olasıdır.

Buna rağmen Wheeler, tüm hesaplamaların ikili prezisyonla gerçekleştirilmesi durumunda, çevrim uzunluğunun pratik kullanımlar için yeterince uzun olabileceğini ve Turbo Pascal ile yapılan testlerin bu fikri desteklediğini ifade etti. O, Tablo 7.1 ve Tablo 7.2'deki hiçbir parametre değerinin, 100,000 tekrarlama için çevrim oluşturmadığını ve belirlenen dört çevrim uzunluğunun, 38,053,252 ile 72,475,155 değerleri arasında olduğunu belirtti. Fakat bu sonuçların diğer sistemler için genelleştirilemeyeceğini söyleyen Wheeler, kaotik dönüşümlerin kriptografik uygulamalar için kullanılmadan önce algoritmanın, kullanılacağı yazılım ve donanımlar için test edilmesi gerektiğini belirtti. O, test ederek bile sistemin tüm parametre değerleri için iyi sonuçlar verebileceğini garantilemenin zor olduğunu söyledi.

Kaotik dönüşümün Matthews'in önerdiği şekilde kullanılmasının güvenli olamayacağını ifade eden Wheeler, bu dönüşümün diğer kriptografik uygulamalar için uygun olabileceğini, örneğin dönüşümün standart sanki - rasgele dizi üreteçleriyle birleştirilerek kullanılabileceğini, belirtti. Rasgele sayı üretecinin çıkışına uygulanan kaotik dönüşüm, anahtar dizisinin ufak bir parçası kullanılarak gerçekleştirilen "olası kelime" atağını önleyebilir.

Tablo 7.2. Tek prezisyonlu gerçel sayılarla bulunan çevrim uzunlukları.

$\beta =$	1.03	1.33	1.63	1.93	2.23	2.53	2.83	3.13	3.43	3.73
x_0										
0.05	640	565	421	10964	7514	2241	738	1215	2395	2320
0.15	2557	2843	421	10964	7514	1674	859	1215	2395	1248
0.25	2557	2843	2718	10964	203	1709	859	1215	2395	149
0.35	640	2843	421	10964	7514	2241	859	1215	2395	1248
0.45	2557	2843	421	10964	7514	1674	859	1215	2395	1248
0.55	2557	2843	421	10964	7514	2241	859	1215	2395	149
0.65	2557	671	421	10964	455	1988	1183	1215	2395	1248
0.75	2557	2843	421	10964	7514	2241	1183	1215	2395	1248
0.85	640	2843	421	10964	7514	2241	859	1215	2395	149
0.95	2557	2843	2718	10964	7514	2241	1183	1215	2395	1248

7.3 Doğrusal Olmayan Anahtar Üreteçleri

Mitchell, [6]'daki çalışmasında, doğrusal olmayan anahtar dizileri üreten bir yöntem sundu. Wheeler, [6]'da Matthews'in [6]'da önerdiği kaotik dönüşümü, ortaya çıkarılmamış tehlikeler içerdiği için eleştirirken; Mitchell, önerdiği yöntemin bu eleştirilerden yara almayacağını ve daha başka istenilen özelliklere de sahip olduğunu söyledi. Mittchell'in bu çalışması şu şekilde özetlenebilir.

Wheeler, Matthews'in önerdiği

$$x \mapsto g(x) = (\beta + 1)(1 + 1/\beta)^{\beta} x(1-x)^{\beta}, \quad 1 \le \beta \le 4, \quad 0 < x < 1 \tag{7.29}$$

kaotik dönüşümün, değer bölgesi 149'dan 16456'ya değişen uzunluklarda çevrim oluşturduğunu gösterdi. Gerçekte çevrim oluşturmayan dönüşüm değerleri, yuvarlatma hatalarından dolayı çevrim içeren dizilere neden olurlar. Anahtar dizisinde meydana gelen bu problem, hesaplamada kullanılan prezisyonun derecesine bağlıdır. Sınırlı ve monotonik olmayan dönüşümlerin, yuvarlatma hatalarından dolayı çevrime girme olasılıkları vardır. Bu nedenle anahtar dizileri üretmek için monotonik dönüşümleri ele almak iyi bir seçimdir.

7.3.1 Monotonik Doğrusal Olmayan Üreteçler

Matthews'in önerdiği dönüşüm, $\beta = 1$ için çok iyi bilinen lolistik dönüşüme ($\lambda = 4$ için) eşdeğerdir,

$$x \mapsto 4x(1-x), \qquad 0 < x < 1. \tag{7.30}$$

Bu dönüşüm, θ, x_0 koşuluna bağlı olmak üzere $x(n) = \sin^2(2^n \theta\, \pi)$ çözümüne sahiptir. Bu çözüme benzer; fakat monotonik olan bir çözüm göz önüne alınabilir,

$$x(n) = an^b, \quad a > 0, \quad b > 1. \tag{7.31}$$

a ve b değerlerini kullanıcı seçer ve b'nin 1'den büyük alınmasının nedeni, x değerinin tek bir sayıda takılıp kalmasını önlemek içindir. x(n), n ile monotonik olarak artar ve x(n)'nin (D-1). ve D. rakamları, anahtarın n. elemanını belirler.

(7.31) ile tanımlanan üretecin, $a = 987.0$, $b = 1.1$ ve n = 11, 12, 13, 14, 15 değerleri için TI - 30 hesap makinesinde gerçekleştirilmesiyle, 13799.047, 15185.060, 16582.685, 17991.111, 19409.643 değerleri elde edilir. K anahtarı, bu sayıların 7. ve 8. rakamları alınarak 47, 60, 85, 11, 43 şeklinde oluşturulur. Tekrarlamalı olmayan bu üreteç,

$$x(n+1) = a\left[a^{-1/b} x(n)^{1/b} + 1\right]^b \tag{7.32}$$

şeklindeki tekrarlamalı üretece dönüştürülebilir. a parametresi, n'nin maksimum değeri için, taşma oluşturmayacak şekilde seçilir. Örneğin n'nin maksimum değeri 2000 ise ve 8 göstergelik bir makine kullanılıyorsa; $a < 2499$ koşulu sağlanmalıdır.

x(n), n ile monotonik arttığı için, bu dönüşümle üretilen dizinin bir çevrim oluşturması mümkün değildir. x(n) kaotik değilken; sınırlı, monotik olmayan ve çevrimsiz olmayan k(n) dizisi kaotiktir.

(7.31) ve (7.32), parametrelerin monotonikliği ve üretecin bir değerde takılıp kalmasını garantileyecek şekilde seçilmesiyle,

$$x(n)=\sum_{i=1}^{k}a(i)n^{b(i)},\quad a(i),b(i)>0,\quad b(1)>1 \tag{7.33}$$

şeklinde tekrarlamalı olmayan şekle genelleştirilebilir. (7.32)'ye dayanan tekrarlamalı üreteç ise, parametrelerin uygun seçilmesiyle

$$x(n+1)=\sum_{i=1}^{k}a(i)\left[d(i)x(n)^{g(i)}+c(i)\right]^{b(i)} \tag{7.34}$$

şeklinde verilir.

Daha güvenli bir yöntem, farklı fonksiyonel şekildeki ve farklı parametre değerlerindeki birden fazla monotonik doğrusal olmayan üretecin birlikte kullanılması olabilir. Bu birden fazla farklı üreteçlerin çıkışlarının toplanması ile elde edilen sayıların (D-1). ve D. rakamları alınarak anahtar dizileri oluşturulabilir.

7.3.2 Simülasyonlar

$$x(n)=an^{b},\quad a>0,\quad b>1,\quad n\in[1,20000] \tag{7.35}$$

üreteci,

$a = 1234.56,\ b = 1.070$
$a = 1234.57,\ b = 1.070$
$a = 1234.56,\ b = 1.071$
$a = 1234.57,\ b = 1.071$

parametre değerleri için incelendi. Elde edilen 4 dizinin herhangi ikisinin birbirleriyle karşılaştırılması, anahtar dizilerinin parametrelerine olan duyarlıklarının kontrol edilmesine olanak sağlar.

Yukarıda sunulan yöntemle bu parametre değerleri için üretilen anahtar dizileri, çevrim içermediklerinin doğrulanması amacıyla test edildi. Ardından ilk 10 gecikme için, her dizinin öz - ilişki fonksiyonu hesaplandı. Hiçbir durum için öz - ilişki fonksiyonun büyüklüğü, 0.015'den büyük çıkmadı. Bu sonuçlar en azından ilk 10 gecikme için seri ilişkinin olmadığını gösterir. Daha sonra her dizi için, Kolmogorov - Smirnov testi [56 s 663] uygulanarak; [00,99] aralığındaki tamsayıların dizi içinde düzgün dağılıp dağılmadığı kontrol edildi. Bu test sonucunda düzgün olmayan dağılım gözlenemedi.

Son olarak herhangi iki dizi için ilişki katsayıları göz önüne alınarak, anahtar dizisinin ufak değişimlere karşı olan duyarlığı kontrol edildi. Kontrol sonucunda, ilişki katsayılarının büyüklüğünün hiçbir durum için 0.011'den daha büyük olmadığı görüldü. Yani kriptoanalizci, parametrelerin yaklaşık olarak elde edilen değerlerini kullanarak anahtar dizisini yeniden inşa edemez. Çünkü anahtar dizisi parametrelerdeki çok ufak değişimlere karşı aşırı duyarlıdır.

7.3.3 Sonuç

Yukarıda özetlenen bu çalışmada Mitchell, Matthews'in önerdiği kaotik dönüşüm yerine monotonik bir dönüşüm önerdi ve bu dönüşümün periyodik çevrimler oluşturmadığını belirtti. O, önerdiği kaotik olmayan fakat monotonik olan dönüşümden elde edilen sayı değerlerinin 7. ve 8. rakamlarının alınmasıyla oluşturulan anahtar dizilerinin kaotik olduğunu ifade etti.

7.4 Kaotik Bir Şifreleme Algoritmasının Süper Bilgisayar Yardımıyla İncelenmesi

Wheeler, [5]'te Matthews'in [4]'te önerdiği kaos kuramına dayalı doğrusal olmayan sanki - rasgele dizi üretecini, düşük prezisyonlu aritmetik ile gerçekleştirildiğinde çevrimler oluşturduğu için eleştirdi. Kısa çevrim uzunluklarının ciddi kriptografik uygulamalar için kullanılamayacağı açıktır. Bu zayıflığı kullanan atak türleri literatürde mevcuttur. Wheeler ve Matthews'in Cray Y - MP makinesini kullanarak kaotik şifreleme algoritmasının özelliklerini inceledikleri ve elde ettikleri

sonuçlar ışığında üretecin kriptolojik uygulamalar için hala ilgi odağı olabileceği sonucuna vardıkları [7]'deki çalışmaları şu şekilde özetlenebilir.

Bir algoritma yardımıyla sanki - rasgele sayılardan oluşan diziler üreten üreteçler, kriptolojide önemli bir yer tutar. Bu tür üreteçler, kısa bir kullanıcı anahtarından uzun anahtar dizileri üretebilirler. Eğer anahtar dizisi kendini tekrar etmiyorsa ve yeterince rasgele ise, dizinin periyodik doğasını kullanan kriptografik teknikler artık etkin değildir. Buna rağmen parametre değerlerini elde eden diğer tekniklerle şifreleme sistemi kırılabilir. Bu tür ataklara karşı dayanıklı anahtar dizileri üretebilen algoritmalar, kriptolojik uygulamalar için çok önemlidir.

Matthews, kendisinin geliştirdiği doğrusal olmayan tekrarlamalı kaotik algoritmanın bu tür ataklara karşı daha az hassas olduğunu söyledi. Buna rağmen Wheeler, bu algoritmanın, düşük prezisyonlu bir sayısal bilgisayarla gerçekleştirilmesi sonucu, kriptoanalizstin yararlanabileceği yeterince küçük çevrimler oluşturduğunu söyledi. Burada, bu problemden yüksek prezisyonlu aritmetik kullanıldığında kurtulunabileği gerçeği üzerinde duruldu. Cray Y - MP süper bilgisayarıyla gerçekleştirilen incelemeler, çevrim uzunluğunun prezisyon ile üstel olarak arttığını gösterdi.

7.4.1 Algoritma ve Onun Gerçekleştirilmesi

Matthews, $\lambda = 4$ olduğunda kaotik diziler üretebilen

$$\mathbf{x} \mapsto \mathbf{g}(\mathbf{x}) = \lambda\, \mathbf{x}(1-\mathbf{x}) \qquad (0 < \mathbf{x}_0 < 1) \tag{7.36}$$

lojistik dönüşümünü kullanarak; ayar parametrelerinin sonsuz değeri için kaos üretebilen bir dönüşüm türetti:

$$\mathbf{x} \mapsto \mathbf{g}(\mathbf{x}) = (\beta + 1)\left(1 + \frac{1}{\beta}\right)^{\beta} \mathbf{x}(1-\mathbf{x})^{\beta} \,. \tag{7.37}$$

İfadede β, $1 \le \beta \le 4$ şartını sağlar. β'nın alt sınırı, (7.36) ifadesinde λ'nın 4 alınmasıyla belirlenirken; üst sınır, hesaplamalarda kullanılan makineler tarafından

belirlenir. x_0 değeri, 0 ile 1 arasında kalmasına rağmen, dejenere diziler oluşturan patolojik değerler vardır. Dejenere dizi demekle, $\mathbf{x(n+1)=x(n)}$ olması kast edilmektedir. Bu tür diziler oluşturan x değerleri,

$$\mathbf{x} = 1 - \frac{\beta}{(\beta+1)^{1+\frac{1}{\beta}}} \tag{7.38}$$

şeklinde tanımlanır. Yani $\beta = 1$ olduğunda, $\mathbf{x_0 = 3/4}$ değeri dejenere bir dizi üretir. Buna ilave olarak, nihayi olarak (7.38) ile verilen patolojik değere neden olan sonsuz sayıda ilk koşul değeri vardır. Bu durum, takip eden bölümde detaylı olarak incelenecektir.

Bu dönüşümün kriptografik bir algoritma olarak kullanılması için Matthews, β ve x_0 değerlerinin D rakamla belirlenmesini ve sonuç değerinin D+2 rakamla hesaplanmasını önerdi.

Doğrusal olmayan dönüşümün ufak hatalara olan aşırı duyarlığından dolayı, dönüşümün nümerik değeri tam ve doğru olarak bilinmediği müddetçe, anahtar dizisinin tamamını elde etmek mümkün değildir.

7.4.2 Yuvarlatma Hatalarından Kaynaklanan Çevrimler

Eğer kriptolojik uygulamalar için uygun diziler üretilmek isteniyorsa, belli patolojik değerler başlangıç olarak seçilmemelidir. Fakat dejenere dizilere neden olan bu patolojik değerlere, dönüşümün daha sonraki takrarlama değerlerinde de ulaşılması mümkündür. Li ve Yorke, (7.16) ve (7.17) gibi ifadelerin, nihayi olarak çevrim oluşturacağı sonsuz sayıda başlangıç değeri içerdiğini gösterdiler, [57].

Daha fazlası, aslında patolojik olmayan bir değer kullanılan düşük prezisyondan kaynaklanan yuvarlatmalardan dolayı, patolojik bir değere dönüşebilir. Gerçekte gerçel sayıların sonlu gösteriminden dolayı, üretilen her dizi nihayi olarak bir çevrime girer. Gerek dejenere dizilerin gerekse çevrim içeren dizilerin, kriptolojik uygulamalar için kullanılması oldukça tehlikelidir.

Buna rağmen çalışmada, bu tür problemlerin üstesinden yeterince yüksek prezisyon ve detektör kullanımıyla gelinilebileceği gösterildi.

Gerçel sayılar, sayısal bir bilgisayarda her birine sabit miktarda yer ayrılan gezer nokta gösterimiyle ("floating - point representation") saklanırlar. Gezer nokta gösterimi, kesirli ("fractional") ve üstel ("exponent") olmak üzere iki kısımdan oluşur. Makinede 01 15000000 şeklinde verilen gezer noktalı bir sayını değeri, $.15000000X10^{01}$ şeklinde yorumlanır.

Sayısal bilgisayarlar, ikili gösterim sistemini kullanırlar. Buna rağmen pek çok bilgisayarda bu gösterim sistemi oldukça farklı detaylara sahiptir. Yine bilgisayarlar, kesirli kısımları N rakam içeren iki sayının çarpımı sonunda elde edilen 2N rakamı azaltmak için de farklı yöntemler kullanırlar. Sonucu tekrar N rakama azlatmak için, bazı bakineler N rakamdan fazlasını keserken; bazıları kesirli kısmı en yakın sayıya yuvarlar.

Bilgisayarların kullandığı metotların tümü, kaotik algoritmanın yararlı olup olmayacağı konusunda yorum yapabilmek için göz önüne alınması gereken çok önemli noktalardır. Çok uç bir örnek ele alınsın. Örneğin bilgisayar tüm hesaplamaları, değeri 0.5'in üzerinde olan tüm sayıları 1'e yuvarlayarak gerçekleştirdiğinde, kaotik dönüşüm, prensip olarak kabul edilebilir $\beta = 1$ değeri için uygunsuz bir dizi üretir.

Buna rağmen kullanılabilir takrarlama sayısının, prezisyon ile üstel olarak artacağını beklemek için iyi bir neden vardır. Eğer prezisyon D rakamla sınırlanırsa, dönüşümün alabileceği 10^D farklı değer vardır. Genel olarak istenmeyen diziyi üreten değere ilk tekrarlamadan sonra; fakat 10^D tane tekrarlamadan önce ulaşılacaktır. Problemli değere ulaşılmadan önceki tekrarlama sayısı, bu iki uç sayının geometrik ortalamasıdır. Böylece problemler oluşmadan önceki tekrarlama sayısı $\langle N \rangle$, $1x10^D$'nin kare köküyle orantılı olacaktır,

$$\langle N \rangle \propto 10^{0.5D}. \qquad (7.39)$$

Yani çevrimden ve dejenere dizilerden kaçınılamamasına rağmen, bu problemlerin meydana gelmediği bir D değeri bulmak mümkündür.

7.4.3 Çevrim ve Diğer Özelliklerin Cray Y - MP ile İncelenmesi

Cray süper bilgisayarı, gezer noktalı sayıları ifade etmek için 64 bit kullanır. Kesirli kısım, bir tanesi işaret biti olmak üzere 49 bitten oluşur. Yani üstel, bir tanesi işaret biti olmak üzere 15 bitten meydana gelir.

Kaotik dönüşümü, bir çevrime girmesi veya $0 < x < 1$ bölgesi dışında bir değer üretmesi durumuna kadar tekrarlayan bir test programı, Cray Pascal'da yazıldı. Bu program ile 9 farklı prezisyon için, kaotik dönüşüm test edildi. Testler, Tablo 7.3'te görülen 5 farklı x_0 ve 10 farklı β değerleri için gerçekleştirildi. Bu değerler, x_0 ve β değerlerinin maksimum ve minimum değerleri arasından doğrusal interpolasyon ile seçildi.

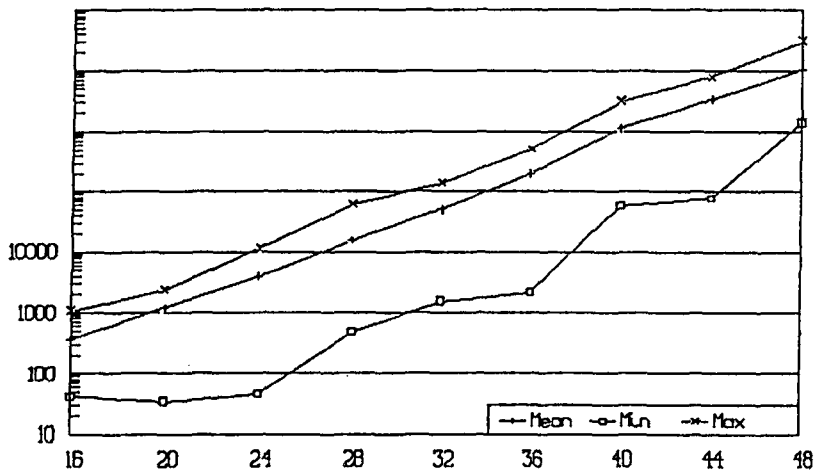
Tablo 7.3. Test değerleri.

x_0	β	
0.169982	1.317226	2.624208
0.333033	1.578623	2.885605
0.496085	1.840019	3.147001
0.659137	2.101415	3.408397
0.822188	2.362812	3.669794

Her bir prezisyon için, 50 test yapıldı. Böylece toplam 450 test icra edildi. Her test değeri için, program ne şekilde sonlandığını yazılı olarak ifade etmektedir.

Şekil 7.1, her prezisyon seviyesinde yapılan 50 deney için elde edilen kullanılabilir tekrarlama sayısının ortalama, minimum ve maksimum değerlerini gösterir. Ortalama kullanılabilir tekrarlama sayısı, 16 bit için 357 iken; 48 bit için 10 milyonun üzerindedir. Şekil 7.1, kaotik dönüşümün yüksek prezisyon ile gerçekleştirilmesi durumunda, kriptografik uygulamalar için yeterince uzun çevrim uzunluğuna sahip diziler üretebileceğini göstermektedir.

Bu gözlem, kısa çevrimler oluşturduğu için eleştirilen kaotik dönüşümün, yüksek prezisyon ile gerçekleştirilmesi durumunda hala ilgi odağı olabileceğini göstermektedir.



Şekil 7.1. Kullanılabilir tekrarlama sayısı.

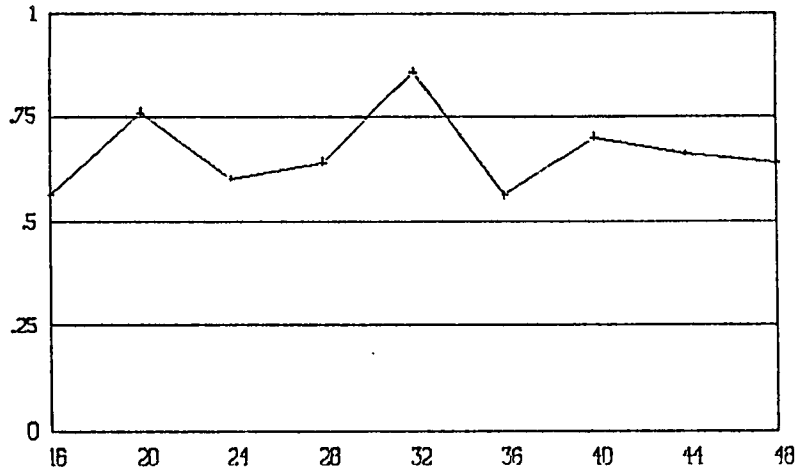
Sonuçların en küçük kareler yöntemi ile analizi, P bit sayısını göstermek üzere, kullanılabilir tekrarlama sayısının ortalama değerinin yaklaşık olarak

$$\langle \mathbf{N(P)} \rangle \approx 1.8 \times 10^{0.142\mathbf{P}} \tag{7.40}$$

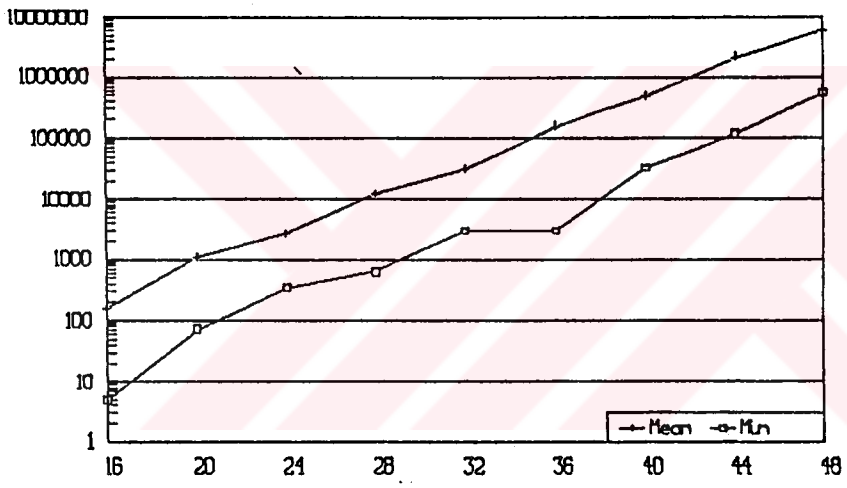
olduğunu gösterdi. (7.40), P bitin onlu tabanda kaç rakama denk düştüğünü gösteren ve P cinsinden $\mathbf{D} = \log_{10} 2\mathbf{P} = 0.301\mathbf{P}$ şeklinde yazılan D'ye bağlı olarak ifade edilebilir,

$$\langle \mathbf{N(D)} \rangle \approx 1.8 \times 10^{0.47\mathbf{D}} . \tag{7.41}$$

Yani (7.41) ifadesinin üstel kısmı, tahmin edildiği gibi 0.5D'ye yakındır. Şekil 7.2, çevrimle sonuçlanan dizilerin, prezisyona göre grafiğini gösterir. Şekil 7.3 ise, ortalama ve minumum çevrim uzunluklarının prezisyona bağlı grafiğini gösterir. Kullanılabilir tekrarlama sayısı, çevrim uzunluğundan büyük olabilir. Örneğin prezisyonun 16 bitle sınırlandığı durum için yapılan testlerden biri, dönüşümün periyodu 5 olan çevrime girmeden önce, 795 değer ürettiğini gösterdi. Fakat diziyi bir çevrim içinde değerlendirmek daha yaygın olduğundan, çevrim uzunluğu ile kullanılabilir tekrarlama sayısı aynıdır.



Şekil 7.2. Çevrimle sonuçlanan sonlanmaların oranı.



Şekil 7.3. Çevrim uzunlukları.

7.4.4 Sonuç

Wheeler ve Matthews'in yukarıda özetlenen bu çalışmasında gerçekleştirilen testler, Wheeler'in Matthews'in önerdiği kaotik dönüşümde işaret ettiği çevrimlerin kriptorafik uygulamalar için bir kusur olmadığını gösterdiler. Onlar, test sonuçlarının herhangibir prezisyonda kriptografik uygulamalar için kabul edilemez dizilerin üretilme riskini tahmin etmeye olanak sağladığını ifade ettiler.

Wheeler ve Matthews, sonlanma riski bulunan C uzunluğundaki mesaj sayısını 10^R'den 1'e indirmek için gerekli bit sayısını,

$$\mathbf{P} = 7(\log_{10}\mathbf{C} + \mathbf{R}) - 1.8 \tag{7.42}$$

şeklinde ve prezisyonla ilişkili olan R'yi ise,

$$\mathbf{R} = 0.142\mathbf{P} + 0.26 - \log_{10}\mathbf{C} \tag{7.43}$$

olarak tanımladılar. C karakter içeren bir mesajın güvenli bir şekilde şifrelenmesi için, $\mathbf{P} >> 7\log_{10}\mathbf{C}$ koşulunun sağlanması gerektiğini belirten Wheeler ve Matthews, K anahtar sayısını, P bit için $\mathbf{K} = 10^{0.602\mathbf{P}-4}$ şeklinde genelleştirdiler.

Onlar, prezisyonun 48 bitle sınırlandığı durumda sonlanma riski bulunan 5000 karakterli dizi sayısının, $10^{3.37}$ (≈ 2300)'den 1'e indiğini ve anahtar uzayının 10^{25} olduğunu belirttiler. Bütün bu söylenenlerin ışığında, ikili prezisyonun pek çok uygulama için yeterli olabileceğini ifade ettiler.

Sonuç olarak Wheeler ve Matthews, Matthews'in önerdiği kaotik dönüşümün kriptografik uygulamalar için göz önüne alınmaya değer basit bir sistem olduğunu ifade ettiler.

7.5 Kriptografide Kaos: Garip Çekicilerden Kaçış

Carroll, Verhagen ve Wong, Lorenz sisteminin kaotik yapısından dolayı sanki - rasgele dizi üretmek için kullanılabileceğini söylediler. Buna rağmen Lorenz sisteminden elde edilen dizilerin seri ilişkili olmadığını garantilemek için, bu sistemin değiştirilmesi gerektiğini vurguladılar. Değiştirilmiş Lorenz sisteminin iyi istatistiksel özellikler içeren oldukça uzun diziler üretebileceğini ve bu üretme mekanizmasının hızlı ve verimli olduğunu ifade ettikleri Carroll, Verhagen ve Wong'un [8]'deki bu çalışmaları şu şekilde özetlenebilir.

Kaotik sistemlerin çözümleri kısa bir zaman aralığı dışında tam olarak tahmin edilemediği için rasgele olarak adlandırılırlar. Lorenz sistemi üç boyutta sanki - rasgele diziler üretebildiğinden, kriptografide lojistik dönüşümden daha kullanışlı olabilir. Lorenz denklemleri, σ, r, b sabitler olmak üzere,

$$\frac{dx}{dt} = -\sigma x + \sigma y$$
$$\frac{dy}{dt} = -xz + rx - y \qquad (7.44)$$
$$\frac{dz}{dt} = xy - bz$$

şeklinde verilir. Bu çalışmada σ, r, b sabitleri söylendikleri sıra ile 10, 28 ve 8/3 olarak alınacaktır. Bu değerlerle sistem kaotiktir.

Analitik olarak çözülemeyan bu denklemler, nümerik olarak çözülebilirler. Denklemlerin kaotik yapısından dolayı, nümerik çözüm hem kullanılan nümerik tekniklere hem hesaplamanın gerçekleştirildiği prezisyona bağlıdır. Denklemleri çözmek için en basit yöntem, Euler metodudur:

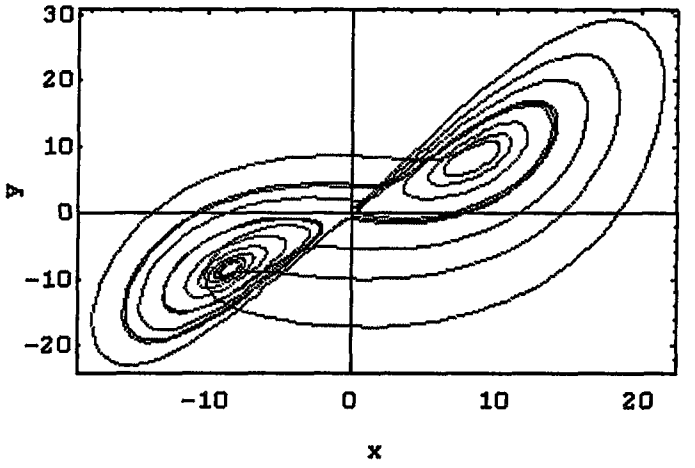
$$x(n+1) = x(n) + \left(-\sigma x(n) + \sigma y(n)\right)\Delta t$$
$$y(n+1) = y(n) + \left(-x(n)z(n) + rx(n) - y(n)\right)\Delta t \qquad (7.45)$$
$$z(n+1) = z(n) + \left(x(n)y(n) - bz(n)\right)\Delta t.$$

Eğer $(\sigma - b - 1) > 0$ sağlanırsa, sistemin davranışı

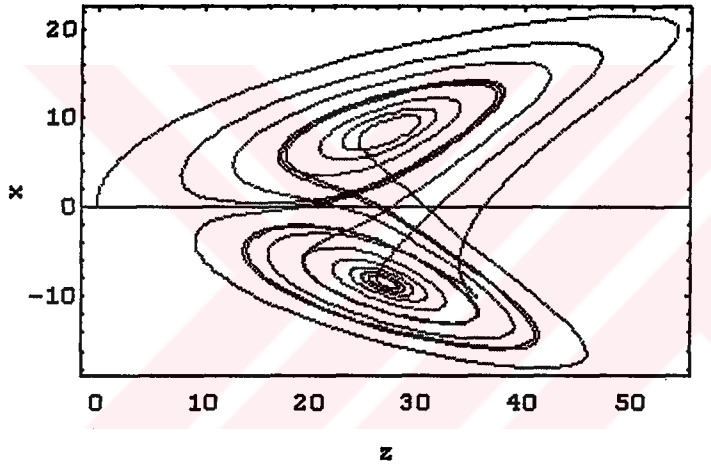
$$r > r_c = \frac{\sigma(\sigma + b + 3)}{(\sigma - b - 1)}, \quad r > 1 \qquad (7.46)$$

için kaotiktir. $\sigma = 10.0$, $r = 28.0$ ve $b = 8.0/3.0$ değerleri için bu üç koşul sağlanır. Şekil 7.4 ve Şekil 7.5, başlangıç değeri $(x_0, y_0, z_0) = (0.0, 1.0, 0.0)$ olmak üzere, bu parametre dağerleri için Lorenz sisteminin davranışını gösterir.

Şekil 7.4 ve Şekil 7.5'de görrüldüğü gibi sistem, yukarrıdaki parametre değerleri için üç tane kararsız denge noktasına sahiptir. Bu simetrik denge noktalarından birine yakın başlayan yörünge, denge noktası etrafında dışa doğru spiraller çizdikten sonra diğer denge noktası civarına atlamakta ve aynı hareketi orada icra etmekte ve hareket, bu şekilde devam etmektedir. Yani sistem bu parametre değerleri için oldukça karmaşık bir davranış sergilemektedir.



Şekil 7.4. r = 28.0 için x - y düzlemindeki Lorenz çekicisi.



Şekil 7.5. r = 28.0 için z - x düzlemindeki Lorenz Çekicisi.

Bu nedenle Lorenz sisteminin sanki - rasgele dizi üreteci olarak kullanılabilmesi için σ, r ve b parametreleri,

$$1)\ (\sigma - b - 1) > 0 \tag{7.47}$$

$$2)\ r > \frac{\sigma(\sigma + b + 3)}{(\sigma - b - 1)} \tag{7.48}$$

$$3)\ r > 1 \tag{7.49}$$

koşullarını sağlayacak şekilde seçilmelidir. Bu parametrelerin seçimi, kriptografik anahtar yerine geçer.

Lorenz denklemlerinin kaotik ve bu denklemlerin çözümlerinin tahmin edilemez olmalarına rağmen, basit nümerik çözüm rasgele diziler üretmez. Çünkü çözüm sonucu bulunan değerler, önceki değerlere çok yakındır.

7.5.1 Yüksek Adım Aralığı ile Ayrıklaştırılmış Lorenz Sistemi

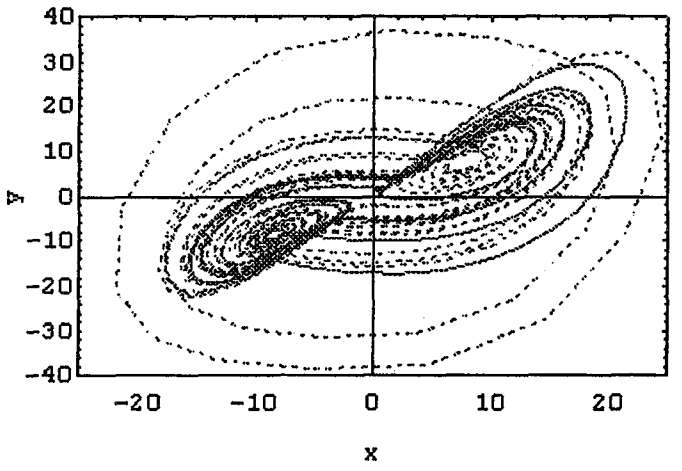
Kriptoloğun görüş açısından olaya bakıldığında, alt dizilerin dizinin kendisiyle oldukça ilişkili olması muhtemeldir. Çünkü her noktanın, bir önceki değerle arasında çok az fark vardır. Fakat sistem kararsızlığa çok yakındır. Eğer differansiyel artım Δt, 0.0245 veya daha fazla alınırsa; parçacık garip çekicinin etkisinden kurtulacak ve üç boyutlu uzayda dolaşacaktır.

Büyük Δt değerleri için bulunan nümerik çözüm, Lorenz sisteminin gerçek çözümü değildir. Çünkü doğrusal olmayan differansiyel denklemleri çözmek amacıyla gerçekleştirilen nümerik yaklaşımlar, yalnızca küçük Δt'ler için geçerlidir. Büyük Δt değerleri alınarak yapılan bu işleme, bir çözüm gözüyle bakılamamasına rağmen, Lorenz denklemlerinin kullanılarak rasgele sayılar üretilmesi gözüyle bakılabilir.

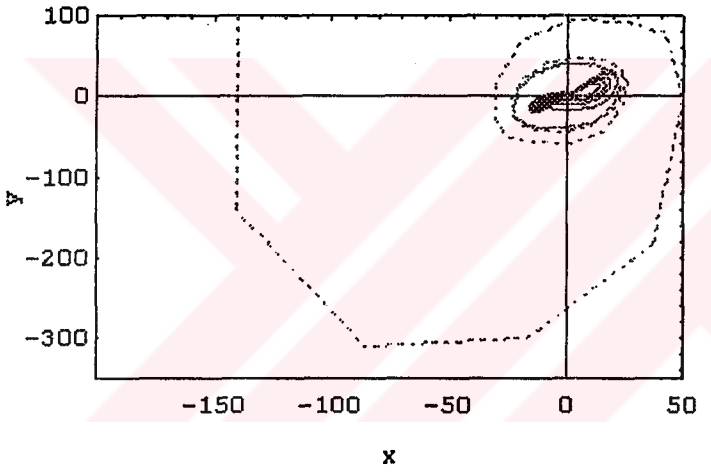
Küçük Δt değerleri için nümerik çözüm garip çekiciyi takip ederken; Δt değeri arttırıldığında başlangıçta garip çekiciye yakın olan nümerik çözüm birkaç adım sonra sonsuza doğru gider.

Δt değeri 0.01'den artım yönünde uzaklaştığında, nümerik çözüm, garip çekici etrafında başlangıçta daireler çizer daha sonra ise garip çekicinin içine düşer. Bu durum, $\Delta t = 0.01$ ve $\Delta t = 0.24$ değerleri için x - y düzleminde çizilen Şekil 7.6'dan görülmektedir. Buna rağmen Δt değeri, 0.0244 ile 0.0245 arasında iken davranış tamamen değişir. Başlangıçta garip çekici etrafında daireler çizen nümerik çözüm, daha sonra garip çekicinin içine düşmek yerine sonsuza doğru uzaklaşır. Şekil 7.7, $\Delta t = 0.01$ ve $\Delta t = 0.0245$ için nümerik çözümün x - y düzlemindeki yerini göstermektedir.

Bu durumda sistem, kaotik durumdan uzaklaşmış ve başıboş şekilde üç boyutlu uzayda dolaşmaya başlamıştır. Böylelikle ufak zaman adımları için birbirine oldukça yakın seyreden x, y ve z değerleri, önemli derecede birbirinden farklılaşır.



Şekil 7.6. Küçük Δt aralıkları için çizilen grafik ($\Delta t = 0.01$, $\Delta t = 0.024$).



Şekil 7.7. Büyük Δt aralıkları için çizilen grafik ($\Delta t = .01$, $\Delta t = .0245$).

Şekil 7.8 ve Şekil 7.9, Δt değerleri büyük olduğunda parçacığın z - x düzlemindeki haraketini göstermektedirler.

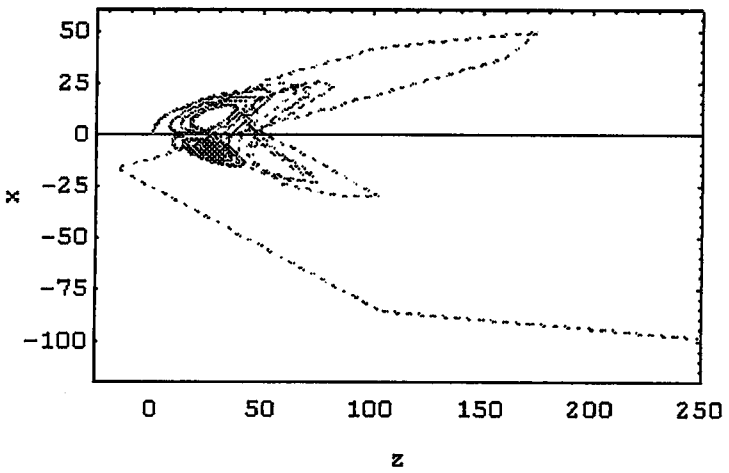
x, y, z değerleri, $\Delta t = 0.0245$ değeri için hızla sonsuza doğru giderler. Bu nedenle değerlerin bir bölge içinde sınırlandırılması gerekir. Burada $m = 2^{32} - 1$ olarak alındı.

Eğer (x, y, z) için başlangıç değerleri küçük olarak seçilirse, Δx, Δy ve Δz değerleride küçük olacakatır. Bu durumda üretilen diziler kendileriyle oldukça ilişkili olacaklardır. Bu problem aşağıda verilen iki yöntemin uygulanmasıyla engellenebilir:

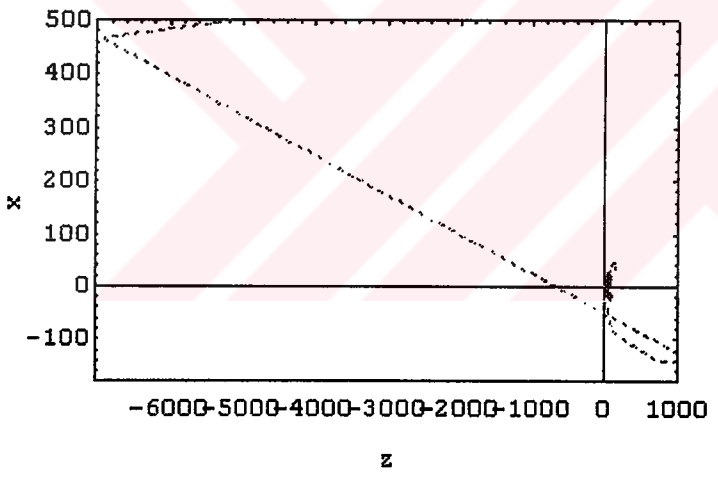
1) x_0 değeri büyük; y_0 ve z_0 değerleri ise küçük seçilerek.

2) x_0 değeri küçük; y_0 ve z_0 değerleri ise büyük seçilerek.

Büyük başlangıç değerlerinin seçimi, üretilen dizilerin rasgelelik karakterini etkilemez.



Şekil 7.8. Büyük Δt değerleri için Lorenz çekicisi, $\Delta t = 0.01$, $\Delta t = 0.0245$.



Şekil 7.9. Büyük Δt değerleri için tekrarlama sayısının arttırılması durumundaLorenz çekicisi, $\Delta t = 0.01$, $\Delta t = 0.0245$.

Lorenz sisteminin σ, r ve b parametre değerleri için üç tane denge noktası vardır:

$$(x,y,z) = (0,0,0) \tag{7.50}$$

$$(x,y,z) = \left(+\sqrt{b(r-1)}, +\sqrt{b(r-1)}, r-1\right) \tag{7.51}$$

$$(x,y,z) = \left(-\sqrt{b(r-1)}, -\sqrt{b(r-1)}, r-1\right). \tag{7.52}$$

Eğer işlem esnasında bir denge noktası üretilirse, bu noktadan sonra üretilen her değer aynı noktayı üretecektir.

x ve y değerleri eş zamanlı olarak sıfır olduğunda, x ve y dizilerinde sıfırlar baş gösterir. Bu durumda z dizisi hemen sabit bir dizi haline dönüşmez; fakat tahmin edilebilir bir tarzda davranır. Bu dizinin değeri, $(1-b\Delta t)$ sabit çarpanı ile değişir.

Lorenz denklemlerinin rasgele olmayan etkisinden dolayı dizilerin bozulma riskini yok etmek amacıyla, bu denklemler değiştirildi. Δx, Δy ve Δz

$$\begin{aligned}\Delta x &= \left(-\sigma x(n)+\sigma y(n)\right)\Delta t\\ \Delta y &= \left(-x(n)z(n)+rx(n)-y(n)\right)\Delta t\\ \Delta z &= \left(x(n)y(n)-bz(n)\right)\Delta t\end{aligned} \tag{7.53}$$

şeklinde olmak üzere, değiştirilmiş bu denklemler

$$\begin{aligned}x(n+1) &= \left(\left(\left(\text{round}\left(x(n)+\Delta x\right)\bmod m\right)+m\right)\bmod m\right)+1\\ y(n+1) &= \left(\left(\left(\text{round}\left(y(n)+\Delta y\right)\bmod m\right)+m\right)\bmod m\right)+1\\ z(n+1) &= \left(\left(\left(\text{round}\left(z(n)+\Delta z\right)\bmod m\right)+m\right)\bmod m\right)+1\end{aligned} \tag{7.54}$$

şeklindedir.

Hesaplamalar $\left(2^{32}-1\right)$ alanı ile sınırlandığı için, sistem periyodiktir. Olası tüm değerler tüketildiğinde, sistemin bu noktadan sonra ürettiği değer daha önce varolduğu için periyodik diziler oluşur. $m=\left(2^{32}-1\right)$ olmak üzere, sistemin erişebileceği mümkün maksimum periyot, m^3 dür. Buna rağmen Lorenz üretecinin rasgele olmayan bazı etkilerinden dolayı, daima mümkün maksimum çevrim uzunluğunu elde etmek mümkün değildir. Gerçek çevrim uzunluğu, nümerik olarak belirlenebilir. Bu amaç için en basit yöntem R. P. Brent tarafından verilir, [48]. Bunun yanısıra R. W. Floyd, R.W. Gosper [48]'de; Sedgewick ve Szymanski ise [58]'de periyotun belirlenmesi için daha farklı yöntemler verdiler.

$(x,y,z)=(0.0,1.0,0.0)$, $\sigma=10.0$, $r=28.0$, $b=8.0/3.0$ ve $\Delta t=0.1$ için sistemin 2^{31} uzunluktan sonra bile çevrime girmediği bulundu. Eğer sistemin periyotu

makinenin prezisyonundan büyükse, Brent algoritması ile sistemin periyotu belirlenemez. Prezisyonun 64 bitle sınırlandığı durum için bu sayı, $2^{52}-1$ dir.

Genel olarak x, y ve z dizilerinin her birinin periyotu m'den büyüktür. Karma eşleşik üreteçler ("multiplicative congruential generators") için periyotun m ile sınırlandığı kuramı, her dizinin diğerine bağlı olduğu bu sisteme uygulanamaz.

7.5.2 Rasgeleliği Ölçen Testler

Sayılardan oluşan dizilerin rasgeleliğini saptamak için geliştirilmiş pek çok test vardır, [48], [59], [60] ve [61]. Burada frekans, ikililer, üçlüler ve dörtlüler için seri, aralık ve aşağı ve yukarı doğru haraket testleri uygulandı. Bu dört test rasgelelik için klasik testlerdir, [62] ve [63].

Denklemler, test dizileri üretmek için 100,000 kez tekrar edildi. Test sonuçları, [8]'de bulunabilir.

7.5.3 Sonuç

Carroll, Verhagen ve Wong, yukarıda verilen bu çalışmada parametre ve başlangıç değerlerinin uygun seçimi ile değiştirilmiş Lorenz denklemlerinin üç boyutta istatistiksel testleri geçen diziler ürettiğini gösterdiler.

Onlar parametre ve başlangıç şartlarının ne şekilde seçilmesi gerektiğini şu şekilde verdiler.

1) Parametrelerin Seçimi

- σ, b ve r pozitif olmalıdır.
- $(\sigma-b-1)>0$, $r>1$ ve $r>\sigma(\sigma+b+3)/(\sigma-b-1)$ eşitsizlikleri sağlanmalıdır.
- b veya $\sqrt{b(r-1)}$, tamsayı olmamalıdır.
- Modül büyük bir asal sayı olmalıdır.

2) Başlangıç Koşullarının Seçimi

- $\Delta t>0.0245$ ve $\Delta t\sigma\geq 1$ koşulları sağlanmalıdır.

- Δx, Δy ve Δz değerlerinin her tekrarlamada sabit kalmaması için (x_0, y_0, z_0), yeterince büyük alınmalıdır. x_0'ın basamak sayısının, y_0 ve z_0'ın basamak sayılarının 3 katı veya 3'e bölümü olarak alınmasıyla bu durum sağlanır.

BÖLÜM 8

ÇOK BOYUTLU KAOTİK SİSTEMLER İLE ŞİFRELEME

Şimdiye kadar yapılan kriptografik çalışmaların, tek boyutlu kaotik bir sistem olan lojistik dönüşümün genelleştirilmiş şeklinin, kriptografide sanki rasgele dizi üreteci olarak kullanımını destekler yönde olması, kısa periyotlu çevrimlerin, kullanılan bilgisayarın prezisyonunun arttırılmasıyla üstesinden gelinebileceğinin gösterilmesi ve değiştirilmiş olmasına rağmen Lorenz sisteminden elde edilen dizilerin istatistiksel rasgelelik testlerini geçmesi; bizi, iki veya daha fazla boyutlu kaotik sistemlerin, kriptografik uygulamalar için daha yararlı olabileceği fikrine götürdü.

Bu bölümde Chua ve Lorenz sistemlerinin kaotik modda, Matthews'in kodlama yöntemi ile rasgele sayılar üretebileceği gösterildi.

8.1 Chua Ve Lorenz Sistemlerinin Şifrelemede Kullanılması

Tek zamanlı sistemlerin tek seferlik kullanılan uzun rasgele dizilere olan ihtiyacından dolayı, bu sistemlerin pek çok pratik uygulama için elverişsiz olduğu Bölüm 6 ve Bölüm 7'de sunuldu. Anahtar dizisi üretmek amacıyla pedlerin yerine bir dönüşüm kullanılırsa, uygun parametre değerleri ile istenilen yerde ve zamanda anahtar dizileri üretilebilir. Böyle bir durumda uzun rasgele dizilerin, iletişim ağını kullanan her kullanıcıya dağıtılması yerine, parametrelerin dağıtılması yeterli olur. Böylelikle tek-zamanlı sistemlerdeki dizilerin güvenli bir şekilde dağıtılması gerekliliğindeki problem ortadan kalkar.

Bilgisayar programlama dillerinin çoğunda rasgele sayı üreten dönüşümlerin varolması ilk bakışta, tek zamanlı sistemlerin kullanımını güç bir hale getiren etkenleri ortadan kaldırabilirmiş gibi gözükebilir. Fakat bu dönüşümlerin meydana getirdikleri dizilerin çevrim uzunlukları, ciddi kriptografik uygulamalar için çok

[54] ve[55]'te geniş olarak incelenen temeli doğrusal kongruens ve ötemeli yazıcılara dayanan üreteçler, Bölüm 6'da verildi. Bu üreteçlere dayanan algoritmalar ile üretilen diziler rasgele görünüşlüdür, fakat deterministik olarak oluşturulduklarından, kendilerini belirli bir uzunluktan sonra tekrar ederler. Bu nedenle bu tür algoritmalar ile üretilen diziler, sanki rasgele diziler olarak adlandırılırlar. Bazı kriptografik uygulamalar için yararlı olan bu üreteçler, tek zamanlı sistemlerde anahtar dizisi olarak kullanılamazlar.

Kaotik sistemlerin ilk koşullara aşırı duyarlı olduğu Bölüm 3'te irdelendi. Bu özelliği sayesinde kaotik sistemler, ayar parametrelerindeki ufak değişimler ile birbirlerinden tamamen farklı diziler üretebilirler. Bunun yanısıra, ilk koşullar pratikte kesin olarak ifade edilemediğinden; bu tür sistemlerin davranışı tahmin edilemezdir.

Kriptografik uygulamalarda kullanılan anahtar dizilerinde olması gereken bu özelliklerin, kaotik sistemlerin özünde içerilmesi, bu tür sistemlere rasgele sayı üreteci gözüyle bakılmasına ve arştırmaların bu yönde ilerletilmesine neden olmuştur.

Matthews'in lojistik dönüşümden yararlanarak genelleştirdiği kaotik dönüşümün tek zamanlı sistemlerle yer değiştirebileceğini söylediği çalışmasının ardından, [4], bu sistemi ufak çevrimlere girdiği için eleştiren Wheeler, daha sonra, sistemin, yüksek prezisyonda gerçekleştirilmesi durumunda, oldukça iyi sonuçlar verdiğini söyledi, [7].

Carroll, Verhagen ve Wong, Lorenz sistemini büyük Δt zaman adımları için Euler yöntemi ile çözerek sistemi kaotik davranıştan uzaklaştırdılar. Onlar Lorenz sisteminin bu şekilde güvenli rasgele sayılar üretebileceğini çalışmalarında ifade ettiler, [8]. Bu durumda sistem, küçük Δt zaman adımları için gözlenen Lorenz çekicisinin çekim etkisinden kurtulur ve (x, y, z) uzayında sonsuza doğru hareket eder.

Biz bu çalışmada, çok boyutlu kaotik sistemlerin kaotik yapısının yok edilmeden de kriptografide sanki rasgele dizi üreteci olarak kullanılabileceğini iddia ettik.

Bu amaçla bu iki sistem, kaotikliği sağlayan parametre değerleri ve küçük Δt zaman adımı için, 0.01, en basit nümerik yöntem olan Euler metotu ile çözüldü. Bu durumda Lorenz ve Chua sistemlerinin gerçekten birbirine oldukça yakın sayılar ürettiği gözlendi. Yani x, y ve z değerleri, bir sonraki adımda üretilen x, y ve z değerleri ile oldukça ilişkilidir. Bu durum, Ek A'da verilen Chua ile ilgili algoritmanın

vc1 = .1, vc2 = 0.2 ve il = -.1 başlangıç değerleri için ürettiği ilk 9 değere bakılarak fark edilebilir,

0.120635714285714 0.198000000000000 -0.128560000000000
0.141635327040816 0.195940757142857 -0.156834400000000
0.163003097732106 0.193829358841837 -0.184814740120000
0.184744035960954 0.191672948829539 -0.212493572562614
0.206863926303440 0.189478723975227 -0.239864469655473
0.229369353518780 0.187253931301955 -0.266922031439135
0.252267728153701 0.185005865209732 -0.293661892829054
0.275567312571771 0.182741864910881 -0.320080730381004
0.299277247437963 0.180469312083680 -0.346176268690278.

Yine Ek A'da verilen Lorenz sistemini kullanan algoritmanın x = .1, y = -.193 ve z = .701 başlangıç değerleri için ürettiği ilk 9 değer,

0.070700000000000 -0.163771000000000 0.682113666666671
0.047252900000000 -0.142819544362333 0.663808182791898
0.028245655563767 -0.128474205535516 0.646039144874307
0.012573669453838 -0.119463157913975 0.628775145962746
-0.000630013282943 -0.114826958996222 0.611992787834482
-0.012049707854271 -0.113851237489629 0.595673703583994
-0.022229860817807 -0.116014866372882 0.579802790229931
-0.031608361373314 -0.120950189384854 0.564367172433792
-0.040542544174468 -0.128412641460198 0.549355611541837

şeklindedir. Bu nümerik simülasyonlardan gözlenen, noktadan sonra ilk üç rakam için görülen yakınlığın ve tahmin edilebilirliğin, en düşük anlamlı rakamlara doğru ortadan kalktığıdır. Bu nedenle ilk adımda her iki sistemin, Euler yöntemi ile üç koordinatta ürettiği sayıların en düşük anlamlı üç rakamının, kullanılan karakter sayısına göre modunun alınmasıyla diziler oluşturuldu. Üç koordinatta ayrı ayrı üretilen bu dizilerin rasgeleliklerinin belirlenmesi amacıyla; dizilere Ek B'de verilen istatistiksel testler uygulandı. Fakat sonuçların iyi olmadığı görüldü. Bunun üzerine üç koordinatta üretilen x, y ve z sayıların ayrı ayrı ele alınması yerine, her üçünün veya her ikisinin en düşük anlamlı üç rakamının oluşturduğu üç basamaklı sayıların birbiriyle toplanması vasıtasıyla elde edilen değerin, kullanılan karakter sayısına göre, 256, modunun

alınmasıyla oluşturulan diziler göz önüne alındı. Bu yöntem ile üretilen diziler genelde istatistiksel rasgelelik testlerini geçti. Aynı algoritma ile noktadan sonra 13, 14 ve 15 rakamlarının alınması yerine (bu rakamlar kullanılan Cx486DX2-S CPU için en düşük anlamlı rakamlara karşılık gelmektedir), noktadan sonra 10, 11 ve 12, 12, 13 ve 14, 8, 9 ve 10 rakamları alınarak 5 farklı ilk koşul değeri için diziler oluşturuldu. Oluşturulan dizilerin 5'te üçü istatistiksel rasgelelik testlerini geçti. Daha fazlası bu algoritma ile her iki sistem için de noktadan sonra 4. dijitten itibaren, Ek B'de verilen istatistiksel rasgelelik testlerini geçen ilk koşul değerleri, rasgele alınan 5 farklı ilk koşul içinde gözlenmiştir.

.vc1 = 0.099910, vc2 = 0.199910 ve il = -0.099910 ilk koşul değerlerinden başlanmak üzere, her biri öncekinden 1E-5 kadar farklı olan 5 ilk koşul değerinin her biri için, Ek A. 1'de verilen Chua sistemini kullanan algoritmada denklemler 131072 kez tekrarlandı. Bu şekilde elde edilen x, y ve z değerlerinin (algoritmada bu değişkenler vc1, vc2 ve il'ye karşı gelir), noktadan sonra 10, 11 ve 12. rakamlarının oluşturduğu üç basamaklı sayıların birbiriyle toplanması şeklinde belirlenen değerin, 256'ya göre modunun alınmasıyla oluşturulan diziler, Ek B'de verilen testlerden geçirildi. Üretilen dizilerden yalnızca birisi d =2 için özilişki testinden kaldı. Bu dizi, vc1 = 0.099920, vc2 = 0.199920 ve il = -0.099920 başlangıç koşullarıyla belirlenen dizidir. Bu durumda d = 2 için belirlenen istatistik 525384'tür. Bu istatistik, 525290.519'dan büyük olduğu için, dizi reddedildi. Sonuçlar Ek C'de görülmektedir.

Benzer şekilde, Ek A. 2'de verilen Lorenz sistemini kullanarak sayılar üreten algoritmadaki denklemler de beş farklı başlangıç değeri için 131072 kez tekrar edildi. İlk dizi x = 0.1, y = -0.193 ve z = 0.701 başlangıç değeri ile, diğer dört dizi ise, x = x - 1E-5, y = y + 1E-5 ve z = z - 1E-5 işleminin belirlediği ilk koşullar ile üretildi. Üretilen bu diziler, Ek D'deki test sonuçlarından da görüldüğü gibi, gerçekleştirilen tüm testleri geçti.

Birbirinden 1E-5 kadar farklı ilk koşullar ile üretilen bu dizilerin birbirinden tamamen farklı olduğunu göstermek amacıyla, dizilere özilişki testi uygulandı. Chua sistemi için, vc1 = 0.099920, vc2 = 0.199920 ve il = -0.099920 başlangıç değerleri ile oluşturulan dizinin, vc1 = 0.099910, vc2 = 0.199910 ve il = -0.099910 ilk koşullarıyla oluşturulan diziden farklı olduğu, özilişki testinin d = 0, 1, 2 ve 3 için uygulanmasıyla

belirlendi. Aynı şekilde Lorenz sisteminde de, birbirinden 1E-5 kadar farklı olan ilk koşulların birbirinden tamamen farklı diziler ürettiği özilişki testi ile doğrulandı.

Lorenz ve Chua sistemlerinin, Euler yöntemi ile çözümlenmesi sonucu elde edilen x, y ve z dizilerinin, çevrim uzunlukları Ek E'de verilen C kodu ile bulunmaya çalışıldı. Denklemlerin $2^{32}-1$ kez tekrarlanması sonucunda bir çevrim gözlenmedi.

Aralarındaki fark 1E-5 olan ilk koşul değerleri ile üretilen dizilerin birbirlerinden tamamen farklı olması, rasgele sayı üretimi için noktadan sonraki elverişli rakamların çokluğu, x, y ve z dizilerinin çok farklı şekillerde kullanılmasıyla istatistiksel rasgelelik testlerini geçen diziler üretilebilmesi ve oluşturulan bu dizilerin çevrim uzunluklarının oldukça büyük olması; Chua ve Lorenz sistemlerinin tek zamanlı sistemlerde rasgele sayı üreteci olarak kullanılabileceğini destekler yöndedir.

Chua çekicisi için, $-2 < x < 2$, $-4 < z < 4$ ve $-0.6 < z < 0.6$, Lorenz çekicisi için de $-20 < x < 20$, $-30 < z < 30$ ve $0 < z < 50$ eşitsizliklerinin varolması, yani parçacığın oldukça geniş bir uzayda hareket etmesi, periyodik çevrimlerin pratik uygulamalar için oldukça büyük olacağı anlamına gelebilir.

Gerçekte kaotik bir sistemde sayılabilir sonsuzlukta yoğun periyodik orbitler ve sayılamayan sonsuzlukta periyodik olmayan orbitler vardır. Böyle bir durumun meydana gelme riskinden, kullanılan kaotik algoritmaya sistemin çevrim oluşturup oluşturmadığını belirleyen bir parçanın ilave edilmesiyle sakınılabilir.

Chua ve Lorenz sistemlerinin, kaotik davranışa neden olan parametre değerleri için üç tane kararsız denge noktasına sahip olduğu Bölüm 5'te sunuldu. Yani tam olarak bu noktalar başlangıç değeri olarak alınmadığı müddetçe; sistem, işlem süresi boyunca bu noktalardan herhangibirine ulaşarak sabit diziler oluşturamaz.

Bölüm 5'te verilen Chua sistemi, kaotik parametre değerlerinde bazı ilk koşul değerleri için sonsuza gider. Yani parçacık Chua çekisinden uzaklaşır. Rasgele sayı üreteci olarak bu sistem kullanıldığında, algoritmaya bu tür ilk koşul değerlerini bulan ve onları göz önüne almayan bir kısım eklenebilir.

Biz Ek A. 2'de Bölüm 5'teki üç parçalı Chua denklemleri yerine, sistemin sonsuza değilde bir limit çevrime gitmesini garantileyen beş bölgeli Chua sistemi kullanıldı.

SONUÇLAR VE ÖNERİLER

Başlangıç şartlarının kesin olarak kesin olarak belirlenmesi durumunda nasıl davranacağı tamamıyla bilinebilen aksi halde ise rasgele davranış sergileyen kaotik sistemlerin, başlangıç şartlarına aşırı duyarlık göstermesi ve bu sayede birbirinden farklı çok sayıda dizinin elde edile bilmesi ve krriptografide kısa bir anahtar yardımıyla "key" uzun rasgele diziler üretebilen algoritmalara duyulan ihtiyaç, araştırmacıları kriptografide kullanılan anahtar dizilerinin ("key stream"), bu tür sistemlerin ürettiği dizilerle yerdeğiştirebileceği konusunda çalışmaya yöneltti.

Gerçek mesajın, rasgele sayılardan oluşan ve mesaj ile eş uzunluklu bir anahtar dizisiyle XOR işlemi kullanılarak şifrelenmesi anlamına gelen ve kırılamaz olduğu tanıtlanabilir tek zamanlı sistemler, tek seferlik kullanılan uzun rasgele dizilerin güvenli bir şekilde dağıtılmasından kaynaklanan zorluklardan dolayı yaygın bir kullanım alanı kazanamadı. Kaotik sistemler bu probleme pratik bir çözüm getirebilirler. Çünkü kaotik sistemlerin parametrelerine ve başlangıç koşullarına duyarlığından dolayı, ayar parametrelerindeki ufak değişimlerle birbirinden tamamen farklı çok sayıda dizi üretmek mümkündür. Buna ilave olarak kaotik sistemlerin çok küçük hatalara duyarlığından dolayı, dönüşümün tekrarlama değerinin tam olarak bilinmemesi durumunda, anahtar dizisinin tümünün doğru olarak yeniden hesaplanmasına imkan yoktur. Daha fazlası şimdiye kadar yapılan çalışmalar, kaotik sistemlerin kriptografide sanki rasgele dizi üreteci olarak kullanımını destekler yöndedir.

Üstte verilen kriterler göz önüne alındığında, çok boyutlu kaotik sistemlerin, daha fazla boyutta rasgele sayı üretebileceği için daha yararlı olabileceği düşünülebilir. Bu amaçla Bölüm 5'te davranışı r parametresine göre incelenen Lorenz sistemi ve davranışı (α,β) parametrelerine göre incelenen Chua çekicisi göz önüne alındı. Sistemlerin, bu parametrelerin belirli değerleri için kaotik davrandığı gözlemlendi. Bu sistemler Euler yöntemi kullanılarak nümerik olarak çözüldüğünde, x, y ve z

değerlerinin ilk üç rakamı kendi aralarında oldukça ilişkilidir. Carroll, Verhagen ve Wong'un Bölüm 7'de ifade ettikleri gibi, birbirine oldukça yakın seyreden sayıların oluşturduğu bir dizi, kriptografik uygulamalar için hiç güvenli değildir. Fakat sistemlerin ufak zaman adımları için Euler yöntemi ile elde edilen sonuçların bir önceki sonuçla ilişkisi, noktadan sonra 4. rakamdan itibaren kaybolmaktadır.

Bu nedenle gerek Chua sistemi gerek ise Lorenz sisteminin Bölüm 8'de ifade edilen algoritma yardımıyla, rasgele sayı üreteci olarak kullanıldı. Lorenz sistemi için $x = 0.1$, $y = -0.193$ ve $z = 0.701$ başlangıç değerleri ile, Chua sistemi için ise $vc1 = 0.1$, $vc2 = 0.2$ ve $il = -0.1$ başlangıç değerleriyle başlanmak üzere her biri bir önceki değerden 1E-5 kadar farklı olan 5 farklı başlangıç değeriyle diziler oluşturuldu. Bu dizilerin rasgele olduklarının doğrulanması amacıyla; dizilere, Bölüm 6'da teorik olarak sunulan Ek B'de ise C kodu ile yazılan frrekans testi, seri testi, $m = 2, 3, 4$ ve 8 için poker testi, hareket testi ve $d = 1, 2, 10, 1000$ ve 10000 için özilişki testi uygulandı. Chua sistemi ile elde edilen dizilerin istatistiksel rasgelelik test sonuçları Ek C'de, Lorenz sistemi ile elde edilen dizilerin test sonuçları ise Ek D'de verildi. Chua sisteminde bir dizi yalnızca $d = 2$ için özilişki testinden kaldı. Lorenz sistemi için 5 farklı ilk koşul değeriyle elde edilen dizilerin tamamı yukarıda ifade edilen testleri geçti. Yapılan nümerik incelemeler noktadan sonra oldukça fazla rakam ile rasgele sayılar üretilebileceğini gösterdi. Yani noktadan sonra (13,14,15) (10,11,12), (8,9,10) ve (6,7,8) rakamlarının alınarak Bölüm 8'de ifade edilen algoritma ile diziler oluşturulduğunda, elde edilen dizilerin istatistiksel rasgelelik testlerini geçtiği gözlendi. Üç boyutta üretilen rakamların toplamını öneren bu algoritma yerine, $(x + y)$, $(x + z)$, $(y + z)$ değerlerini kullanarak sayılar üreten algoritma da kullanılabilir. Bu yol ile üretilen dizilerde rasgelelik testlerini geçti. Buna rağmen üç boyutta üretilen x, y ve z sayılarının ayrı ayrı ele alınmasıyla oluşturulan diziler, istatistiksel rasgelelik testlerini geçemedi.

Biz Ek A. 1'de ve Ek A. 2'de verilen algoritmalar ile en azından 5 farklı ilk koşul için noktadan sonra 10, 11 ve 12. rakamların alınması ile üretilen dizilerin diğerlerinden daha iyi olduğunu gözlemledik. Buna rağmen hangi rakamların alınmasıyla daha güvenli diziler üretilebileceği, 5 tane dizi için elde edilen sonuçlara dayanılarak söylenemez. Bunu için, en azından 1000 farklı ilk koşul değeri için

oluşturulan dizilerin istatistiksel rasgelelik test sonuçları değerlendirilerek bir karara varılmalıdır.

Her iki algoritma ile aralarındaki fark 1E-5 olan başlangıç değerleriyle oluşturulan dizilerin tamamen farklı olduğu, bu dizilere d = 1, 2, 3 ve 4 için özilişki testi uygulanarak doğrulandı.

Chua ve Lorenz sistemlerinin kotik davrandıkları parametre değerleri için oluşturdukları garip çekicilerin sınırları, söylendikleri sıra ile,

$-2 < x < 2$, $-4 < z < 4$ ve $-0.6 < y < 0.6$

$-30 < y < 30$, $0 < z < 50$ ve $-20 < x < 20$

şeklindedir. Yani $x(n+1) = 4x(n)(1-x(n))$ lojistik dönüşümü için meydana gelen çevrim uzunlukları, $0 < x(n) < 1$ olduğundan, Chua ve Lorenz sistemi ile oluşturulan dizilerin çevrim uzunluklarından oldukça kısa olmalıdır. Çünkü Chua ve Lorenz sistemlerinde parçacık oldukça geniş bir uzayda dolaşmaktadır.

Chua ve Lorenz sistemleri ile birbirlerinden 1E-5 kadar farklı olan ilk koşul değerleriyle oluşturulan dizilerin birbirlerinden tamamen farklı olması, rasgele sayı üretmek için noktadan sonraki elverişli rakamların çokluğu, dizilerin meydana getirebilecekleri çevrim uzunluklarının oldukça büyük olması ve x, y ve z dizilerinin farklı şekillerdeki pek çok kombinasyonu için rasgelelik testlerini geçen dizilerin oluşturulabilmesi; bu sistemlerin tek zamanlı sistemlerde rasgele sayı üreteci olarak kullanılmasını destekler yöndedir. Bütün bu söylenilenlere rağmen, dizilerin rasgeleliğinin belirlenmesi amacıyla kullanılan tüm testler bunlar değildir. Bölüm 6'da sunulan testlerden başka, kupon toplayıcı test, permütasyon testi, seri korelasyon testi, t'lerin maksimum testi, ikili türev testi, değişim noktası testi, çarpışma testi gibi istatistiksel testler de vardır.

Ancak rasgele olmadığı bilinen bir takım dizilerin istatistiksel testlerden geçtiği gözlenmiştir. Bu durum anahtar dizilerini test eden başka testlere olan gereksinimi ortaya çıkarmıştır. Bu amaç ile dizilerin öngörülemezliğinin ölçülmesinde kullanılan karmaşıklık testleri türetilmiştir.

Ek A. 1 ve Ek A. 2'de verilen algoritmaların güvenli rasgele diziler üretilebileceği, tüm bu testlerin uygulanması ile daha net olarak ifade edilebilmelidir.

KAYNAKLAR

[1] **GLEICK, J.,** Chaos, New York: Penguin Books, p 79, (1988).

[2] **HEIDARI - BATENI, GHOBAD,** Ph.D., Purdue University, Chaotic Seguences for Secure Digital Communications, December (1992).

[3] **CUOMO, K. M., OPPENHEIM, A. V. and STROGATZ, S. H.,** Synchronization of Lorenz - Based Chaotic Circuit with Application to Communications, IEEE Trans. Cir. and Sys., Vol. 40, No. 10, October (1993).

[4] **MATTHEWS, R.,** On the Derivation of a Chaotic Encryption Algorithm, Cryptologia, Vol. 13, No. 1, pp. 29 - 42, January (1989).

[5] **WHEELER, D. D.,** Problems with Chaotic Cryptosystems, Cryptologia, Vol. 13, No. 3, pp. 243 - 250, July (1989).

[6] **MITCHELL, D. W.,** Nonlinear Key Generators, Cryptologia, Vol. 14 No. 4, pp. 350 - 354, October (1990).

[7] **WHEELER, D. D. and MATTHEWS, R. A. J.,** Supercomputer Investigations of a Chaotic Encryption Algorithm, Vol. 15, No. 2, pp. 140 - 152, April (1991).

[8] **CARROLL, J. M., VERHAGEN, J. and WONG, P. T.,** Chaos in Crptography: The Escape from the Strange Attractor, Cryptologia Vol. 16, No. 1, pp. 52 - 71, January (1992).

[9] **VIDYASAGAR, M.,** Nonlinear System Analysis, Prentice - Hall, Inc., Englewood Cliffs, N. J., (1978).

[10] **GÜZELİŞ, C.,** Doğrusal Olmayan Devreler, Sistemler ve Kaos Ders Notları, İ. T. Ü., p 38, (1989).

[11] **ARNOLD, V. I.,** Ordinary Differential Equations, M. I. T., Press: Cambridge, MA, (1973).

[12] **HIRSCH, M. W. and SMALE, S.,** Differential Equations, Dynamical Systems and Linear Algebra, Acadamic Press: New York, (1974).

[13] **HALE, J.,** Ordinary Differential Equations, Robert E. Krieger Publishing Co., Inc.: Malabar, Florida, (1980).

[14] **WIGGINS, S.,** Introduction to Applied Nonlinear Dynamical Systems and Chaos, Springer - Verlag: New York, Inc., (1990).

[15] **POINCARé, H.,** Les Méthodes Nouvelles de la Mécanique Céleste, 3 Vols., Gauthier - Villars: Paris, (1899).

[16] **GUCKENHEIMER, J. and HOLMES, P. J.,** Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields, Springer - Verlag: New York, Heidelberg, Berlin, (1983).

[17] **LICHTENBERG, A. J., and LIEBERMAN, M. A.,** Regular and Stochastic Motion, Springer - Verlag: New York, Heidelberg, Berlin, (1982).

[18] **PARKER, T. S. and CHUA, L. O.,** Practical Numerical Algorithms for Chaotic Systems, Springer - Verlag: New York, Berlin, Heidelberg, London, Paris, Tokyo, Honk Kong, (1989).

[19] **DEVANEY, R. L.,** An Introduction to Chaotic Dynamical Systems Second Edition, Addison - Wesley Publishing Company, (1989).

[20] **ECKMANN, J. P.,** Roads to Turbulence in Dissipative Dynamical Systems, Rev. Mod. Phys. No. 53, p 643, (1981).

[21] **RUELLA, D.,** Strange Attractors, Math. Intelligencer No. 2, p 126, (1980).

[22] **JAKOBSEN, M. V.,** Absolutely Continuous Invariant Measures for One Parameter families of One - Dimensional Maps, Comm. Math. Phys. No. 81, pp. 39 - 88, (1981).

[23] **MISIUREWICZ, M.,** The Structure of Mapping of on Interval with zero Entropy, Publ. Math. IHES, No. 53, pp. 5 - 16, (1981).

[24] **JOHNSON, R. A.,** m - Functions and Floquet Exponents for Linear Differential Systems, Ann. Math. Pura Appl., (4) Vol CXLVII, pp. 211 - 248, (1987).

[25] **GUCKENHEIMER, J. and JOHNSON, S.,** Distortion of s - Unimodal Maps, Preprint (1989).

[26] **PLYKIN, R.,** Sources and Sinks for a - Diffeomorphisms, Math., USSSR - Sb., No. 23, pp. 233 - 253, (1974).

[27] **NEMYTSKII, V. V. and STEPANOV, V. V.,** Qualitative Theory of

Differential Equations, Dover: New York (1989).

[28] **NEWHOUSE, S. E.,** Lectures on Dynamical Systems, in Dynamical Systems, C. I. M. E. Lectures, Bressanone, Italy, pp. 1 - 114, Birkhauser: Boston, June (1978).

[29] **SINAI, J. G. and VUL, E.,** Hyperbolicity Conditions for the Lorenz Model, Physica 2D, pp. 3 - 7, (1981).

[30] **AFRAIMOVICH, V. S., BYKOV, V. V., and SILNIKOV, L. P.,** On Structrully Unstable Attracting Limit Sets of Lorenz Attractor Type, Trans. Moscow Math. Soc. No. 2, pp. 153 - 216, (1983).

[31] **RYCHLIK, M.,** Lorenz Attractors Through Silnikov Type Bifurcations Part I, Preprint, (1987).

[32] **ROBINSON, C.,** Bifurcation to a Transitive Attractor of Lorenz Type Preprint, (1988)

[33] **BENEDICKS, M. and CARLESON, L.,** The Dynamics of the Hénon Map, Preprint, (1988)

[34] **WIGGINS, S.,** Global Bifurcations and Chaos: Analytical Methods, Springer - Verlag: New York, Heidelberg, Berlin, (1988)

[35] **SCHUSTER, H. G.,** Deterministic Chaos, Physik - Verlag, Weinheim and VCH Publishers, New York, (1984)

[36] **BANKS, J., BROOKS, J., CAIRNS, G., DAVIS, G., and STACEY, P.,** On Devaney's Definition of Chaos, American Math., Monthly No. 99, April (1992)

[37] **PEITGEN, H. O., JÜRGENS, H., and SAUPE D.,** Chaos and Fractal, Springer - Verlag: New York Berlin Heidelberg, (1992)

[38] **KENNEDY, M. P.,** Three Steps to Chaos - Part II, a Chua's Circuit, Primer, IEEE Trans on Circuits Syst. - I: Fundamental Theory and Applications, Vol. 40, No. 10, pp. 657 - 674, October (1993)

[39] **MATSUMOTO, T.,** A Chaotic Attractor from Chua's Circuit, IEEE Trans. Circuits Syst., Vol. CAS - 31, Vol. 12, pp. 1055 - 1058, (1984)

[40] **ZHONG G. O. and AYROM, F.,** Experimental Confirmation of Chaos from Chua's Circuit, Int. J. Circuit Theory Appl. Vol. 13, No. 11,

pp. 93 - 98, (1985)

[41] **CHUA, L. O., KOMURO, M., and MATSUMOTO, T.,** The Double Scroll Family Parts I and II, IEEE Trans. Circuits Sysy., Vol. CAS - 33, No. 11, pp. 1073 - 1118, (1986)

[42] **SALZMAN, B.,** Finite Amplitude Free Convection as an Initial Value Problem - 1, Journal of the Atmospheric Sciences, Vol. 19, No. 4, pp. 329 - 341, (1962)

[43] **RAYLEIGH, LORD.,** On Convention Currents in a Horizontal Layer of Fluid, when the Higher Temperature is on the under Side, Philosophical Magazine and Journal of Science, Vol. 32, No. 192, pp. 529 - 546, (1916)

[44] **COOK, P. A.,** Nonlinear Dynamical Systems, Prentice - Hall International (UK) Ltd., (1986)

[45] **BOYAR, J.,** Inferring Sequences Produced by Pseudorandom Number Generators, J. ACM, Vol. 36, No. 1, pp. 129 - 141, January (1989)

[46] **MASSEY, J. L.,** Shift - Register Synthesis and BCH Decoding, IEEE Trans. Information Theory, Vol. IT - 15, No. 1, pp. 122 - 127, January (1969)

[47] **ZENG, K., YANG, C. H., WEI, D. Y., RAO, R. R. N.,** Pseudorandom Bit Generators in Stream - Cipher Cryptography, IEEE Computer pp. 8 -17, February (1991)

[48] **KNUTH, D. E.,** The Art of Computer Programming, Vol. 2, 2nd Ed., Don Mills Canada: Addison - Wesley, (1981)

[49] **FEIGENBAUM, M. J.,** Universal Behaviour in Non - Linear Systems, Los Alamos Science, No. 1, pp. 4 - 27, (1980)

[50] **FEIGENBAUM, M. J.,** The Universal Metric Properties of Non - Linear Transformation, Journal of Statistical Physics, pp. 669 - 706, (1979)

[51] **ULAM, S. M. and VON NEUMANN,** On Combinations of Stochastic and Deterministic Processes, Bull. Am. Math. Soc. No. 53, p 1120, (1947)

[52] **MATTHEWS, R. A. J.,** A Rotor Devices for Periodic and Random - key Encryption, Cryptologia, (in the Press), (1988)

[53] **DENNING, D. E. R.,** Cryptography and Data Security, Reading MA: Addison - Wesley, (1983)

[54] **KNUTH, D. E.,** The art of Computer Programming, Vol. 2: Seminumerical Algorithms, Reading MA: Addison - Wesley, (1969)

[55] **GOLOMB, S. W.,** Shift Register Sequences (Revised Edition), Laguna Hills CA: Aegean Park Press, (1982)

[56] **LEVIN, R. I.,** Statistics for Management (4th Edition), Englewood Cliffs NJ: Prentice - Hall, (1987)

[57] **Lİ, T. Y. and YORKE, J. A.,** Period Three İmplies Chaos, American Mathematical Monthly, No. 82, pp. 985 - 992, (1975)

[58] **SEDGEWICK, R. and SZYMANSKI, T. G.,** The Complexity of Finding Periods, In Proc. ACM Symposium on Theory of Computing, No. 11, pp.74 - 80, (1979)

[59] **MITRANI, I.,** Simulation Techniques for Discrete Event Systems, New York: Cambridge University Press, (1982)

[60] **BRATLEY, P., FOX, B. C. and SCHRAGE, L. E.,** A guide to Simulation, 2nd Ed. New York: Springer - Verlag, (1987)

[61] **DAGPUNAR, J.,** Principle of Random Variate Generation, New York: Oxford University Press, (1988)

[62] **KENDALL, M. G. and BABINGTON - SMITH, B.,** Randomness and Random Sampling Numbers, Journal of the Royal Statistical Society, No. 101, pp. 147 - 166, (1938)

[63] **KENDALL, M. G. and BABINGTON - SMITH, B.,** Second Paper on Random Sampling Numbers, Supplement to the Jour., Royal Statistical Society, Vol. 6, No. 1, pp. 51 - 61, (1939)

EK A

ÇOK - BOYUTLU KAOTİK SİSTEMLER İLE RASGELE SAYILAR ÜRETEN ALGORİTMALAR

A.1 Chua Sistemini Kullanarak Rasgele Sayılar Üreten Algoritma

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<conio.h>
FILE *fp1;
main()
{
double vc10=.1;
double il0=-.1;
double vc20=.2;
double vc1;
double il;
double vc2;
double alfa=9.0;
double beta=14.28;
double delta=.85;
int iz;
int a[16];
double m0=2.14285714285714285714;
double m1=1.71428571428571428571;
double m2=-1;
double h=.01;
double k1;
char str1[35];
char str2[35];
char str3[35];
char st1[4];
char st2[4];
char st3[4];
double r1;
double r2;
double r3;
double r;
double i;
```

```
int as;
int ps=0;
int pt=0;
int pk=0;
clrscr();
fp1=fopen("as.dat","wb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
vc10=vc10 -0.00001;
il0=il0 +0.00001;
vc20=vc20 -0.00001;
printf("\n %lf %lf %lf",vc10,vc20,il0);
for(i=0.0;i<131072.0;i++){
        if((vc10<=1.0)&&(vc10>=-1.0)){
                k1=-alfa*(1+delta)*vc10 +alfa*vc20 + alfa*m0*vc10;
                vc1=vc10+h*k1;
                }
        if((vc10<-1.0)&&(vc10>=-20.0)){
                k1=-alfa*(1+delta)*vc10 +alfa*vc20 + alfa*(m1*(vc10+1)-m0);
                vc1=vc10+h*k1;
                }
        if((vc10 < -20)&&(vc10 >= -54.714285)){
                k1=-alfa*(1+delta)*vc10 +alfa*vc20 + alfa*(m2*(vc10+20)-
34.714285);
                vc1=vc10+h*k1;
        }
        if((vc10>1)&&(vc10<=20)){
                k1=-alfa*(1+delta)*vc10 +alfa*vc20 + alfa*(m1*(vc10-1)+m0);
                vc1=vc10+h*k1;
                }
        if((vc10>20)&&(vc10<=54.714285)){
                k1=-alfa*(1+delta)*vc10 +alfa*vc20 + alfa*(m2*(vc10-
20)+34.714285);
                vc1=vc10+h*k1;
                }
        vc2=vc20+h*(vc10-vc20+il0);
        il=il0-h*beta*vc20;
        sprintf(str1,"%.20lf",vc1);
        sprintf(str2,"%.20lf",vc2);
        sprintf(str3,"%.20lf",il);
        while(str1[ps]!='.')
                ps++;
        if(str1[ps]=='.')
                ps++;
        st1[0]=str1[ps+10];
        st1[1]=str1[ps+11];
        st1[2]=str1[ps+12];
```

```
        st1[3]='\0';
        while(str2[pt]!='.')
                pt++;
        if(str2[pt]=='.')
                pt++;
        st2[0]=str2[pt+10];
        st2[1]=str2[pt+11];
        st2[2]=str2[pt+12];
        st2[3]='\0';
        while(str3[pk]!='.')
                pk++;
        if(str3[pk]=='.')
                pk++;
        st3[0]=str3[pk+10];
        st3[1]=str3[pk+11];
        st3[2]=str3[pk+12];
        st3[3]='\0';
        r1=atof(st1);
        r2=atof(st2);
        r3=atof(st3);
        r=r1+r2+r3;
        r=fmod(r,256.0);
        as=(int)(r);
        for(iz=15;iz>=0;iz--){
                if((as>>iz<<15))
                        a[iz]=1;
                else
                        a[iz]=0;
                }
         for(iz=7;iz>=0;iz--){
                fprintf(fp1,"%d",a[iz]);
             }
/*      fprintf(fp1,"%c",as);       */
        vc10=vc1;
        il0=il;
        vc20=vc2;
        ps=0;
        pt=0;
        pk=0;
        }
fclose(fp1);
return 0;
}
```

A.2 Lorenz Sistemini Kullanarak Rasgele Sayılar Üreten Algoritma

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<conio.h>
FILE *fp1;
main(){
double sigma=10.0;
double r=28.0;
double b=2.666666666666;
double h=.01;
double x0=.1;
double y0=-.193;
double z0=.701;
double x1;
double y1;
double z1;
double i;
char str1[35];
char str2[35];
char str3[35];
char st1[4];
char st2[4];
char st3[4];
double p1;
double p2;
double p3;
double p4;
int p;
int ps=0;
int pt=0;
int pk=0;
int iz;
int a[16];
clrscr();
x0=x0-0.01;
y0=y0+0.01;
z0=z0-0.01;
fp1=fopen("as.dat","wb");
if(fp1==NULL){
        printf("\nCannot open this file");
        exit(1);
        }
for(i=0.0;i<131072.0;i++){
        x1=x0+h*(-sigma*x0 + sigma*y0);
        y1=y0+h*(-x0*z0 + r*x0 - y0);
```

```
z1=z0+h*(x0*y0-b*z0);
sprintf(str1,"%.20lf",x1);
sprintf(str2,"%.20lf",y1);
sprintf(str3,"%.20lf",z1);
while(str1[ps]!='.')
        ps++;
if(str1[ps]=='.')
        ps++;
st1[0]=str1[ps+10];
st1[1]=str1[ps+11];
st1[2]=str1[ps+12];
st1[3]='\0';
while(str2[pt]!='.')
        pt++;
if(str2[pt]=='.')
        pt++;
st2[0]=str2[pt+10];
st2[1]=str2[pt+11];
st2[2]=str2[pt+12];
st2[3]='\0';
while(str3[pk]!='.')
        pk++;
if(str3[pk]=='.')
        pk++;
st3[0]=str3[pk+10];
st3[1]=str3[pk+11];
st3[2]=str3[pk+12];
st3[3]='\0';
p1=atof(st1);
p2=atof(st2);
p3=atof(st3);
p4=p1+p2+p3;
p4=fmod(p4,256.0);
p=(int)(p4);
for(iz=15;iz>=0;iz--){
        if((p>>iz<<15))
                a[iz]=1;
        else
                a[iz]=0;
        }
/* for(iz=7;iz>=0;iz--)
    fprintf(fp1,"%d",a[iz]);*/
fprintf(fp1,"%c",p);
x0=x1;
y0=y1;
z0=z1;
ps=0;
pt=0;
pk=0;
```

```
        }
fclose(fp1);
 return 0;
}
```

EK B

İSTATİSTİKSEL RASGELELİK TESTLERİ

B.1 Frekans Testi

```
/*Bu program karaktarlerden oluşan dosyaya frekans testini uygular*/
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
FILE *out_file;
main()
{
clrscr();
int i;
int a[16];
double sfr=0.0;
double bir=0.0;
char string1[100];
char string2[200];
char ch1[8];
int j;
double sayac=0.0;
double top;
clrscr();
out_file=fopen("as.dat","rb");
if(out_file==NULL){
        printf("\nCan not open %s",string1);
        exit(1);
        }
do{
        fread((char *)ch1,sizeof(char),8,out_file);
 /* bu kısım sayıyı ikili olarak hesaplıyor*/
        for(j=0;j<=7;j++){
                for(i=15;i>=0;i--){
                        if((ch1[j]>>i<<15))
                                a[i]=1;
                        else
                                a[i]=0;
                       }
```

```
                for (i=7;i>=0;i--){
                        if(a[i]==1)
                                bir++;
                        else
                                sfr++;
                        }
                }
        sayac++;
        }while(sayac<16384.0);
fclose(out_file);
top=(double)(sfr-bir)*(sfr-bir)/(sfr+bir);
printf("\nbir=%lf sifir=%lf ist=%lf",bir, sfr,top);
return 0;
}
```

B.2 Seri Test

```
/*1 ve 0 sayılarından oluşan dosyaya seri testi uygular/
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
FILE *fp1;
main()
{
double n0=0.0,n1=0.0,n2=0.0,n3=0.0;
char ch1,ch2;
double sayac=1.0;
double ist1,ist2,ist3;
double bir=0.0,sfr=0.0;
clrscr();
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048576.0){
        if(sayac==1.0){
                ch1=getc(fp1);
                ch2=getc(fp1);
                }
        if(ch1-'0'==0)
                sfr++;
        if(ch1-'0'==1)
                bir++;

        if(sayac!=1.0)
                ch2=getc(fp1);
```

```
        if((ch1-'0'==0)&&(ch2-'0'==0))
                n0++;
        if((ch1-'0'==0)&&(ch2-'0'==1))
                n1++;
        if((ch1-'0'==1)&&(ch2-'0'==0))
                n2++;
        if((ch1-'0'==1)&&(ch2-'0'==1))
                n3++;
        ch1=ch2;
        sayac++;
        }/*while sonu*/
fclose(fp1);
if(ch2-'0'==0)
        sfr++;
if(ch2-'0'==1)
        bir++;
ist1=n0*n0+n1*n1+n2*n2+n3*n3;
ist2=bir*bir+sfr*sfr;
ist3=ist1*4.0/1048575.0-ist2*2.0/1048576.0+1;
printf("\nn0=%lf  n1= %lf n2=%lf n3=%lf",n0,n1,n2,n3);
printf("\nbir=%lf sfr=%lf ist=%lf",bir,sfr,ist3);
return 0;
}/*main sonu*/
```

B.3 Hareket Testi

```
/* 1 ve 0 sayılarından oluşan dosyaya hareket testini uygular*/
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
FILE *fp1;
main()
{
double dizi[16],dizi1[16],dizi2[16];
double top1=0.0;
double ist1=0.0,ist2=0.0,ist;
int say;
double uzun=1048576;
double sayac=0.0;
char ch1,ch2,ch3;
char c1,c2,c3,c4,c5,c6,c7,c8,c9,c10,c11;
char c12,c13,c14;
char c15,c16;
char c17;
char c18;
double n01=0.0;
double n11=0.0;
```

```
double n02=0.0,n12=0.0;
double n03=0.0,n13=0.0;
double n04=0.0,n14=0.0;
double n05=0.0,n15=0.0;
double n06=0.0, n16=0.0;
double n07=0.0,n17=0.0;
double n08=0.0,n18=0.0;
double n09=0.0,n19=0.0;
double n010=0.0,n110=0.0;
double n011=0.0,n111=0.0;
double n012=0.0,n112=0.0;
double n013=0.0,n113=0.0;
double n014=0.0,n114=0.0;
double n015=0.0,n115=0.0;
double n016=0.0,n116=0.0;
double hareket1;
double hareket2;
double hareket;
clrscr();
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("can not open as.dat");
        exit(1);
        }
/*1 uzunluklu*/
while(sayac<1048574.0){
        if(sayac==0.0){
                ch1=getc(fp1);
                ch2=getc(fp1);
                ch3=getc(fp1);
                }
        else
                ch3=getc(fp1);
        if((ch1=='1')&&(ch2=='0')&&(ch3=='1'))
                n01++;
        if((sayac==4093.0)&&(ch1=='1')&&(ch2=='1')&&(ch3=='0'))
                n01++;
        if((sayac==4093.0)&&(ch1=='0')&&(ch2=='1')&&(ch3=='0'))
                n01++;
        ch1=ch2;
        ch2=ch3;
        sayac++;
        }/*while sonu*/
fclose(fp1);
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("can not open as.dat");
```

```
            exit(1);
            }

    while(sayac<1048574.0){
            if(sayac==0.0){
                    ch1=getc(fp1);
                    ch2=getc(fp1);
                    ch3=getc(fp1);
                    }
            else
                    ch3=getc(fp1);
            if((ch1=='0')&&(ch2=='1')&&(ch3=='0'))
                    n11++;
            if((sayac==4093.0)&&(ch1=='0')&&(ch2=='0')&&(ch3=='1'))
                    n11++;
            if((sayac==4093.0)&&(ch1=='1')&&(ch2=='0')&&(ch3=='1'))
                    n11++;
            ch1=ch2;
            ch2=ch3;
            sayac++;
            }/*while sonu*/
  fclose(fp1);
  /*2 uzunluklu*/
  sayac=0.0;
  fp1=fopen("as.dat","rb");
  if(fp1==NULL){
          printf("\nCan not open as.dat");
          exit(1);
          }
  while(sayac<1048573.0){
          if(sayac==0.0){
                  c1=getc(fp1);
                  c2=getc(fp1);
                  c3=getc(fp1);
                  c4=getc(fp1);
                  }
          else
                  c4= getc(fp1);
          if((c1=='1')&&(c2=='0')&&(c3=='0')&&(c4=='1'))
                  n02++;
          c1=c2;
          c2=c3;
          c3=c4;
          sayac++;
          } /*while sonu*/
fclose(fp1);
sayac=0.0;
fp1=fopen("as.dat","rb");
```

```
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }

while(sayac<1048573.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                }
        else
                c4= getc(fp1);
        if((c1=='0')&&(c2=='1')&&(c3=='1')&&(c4=='0'))
                n12++;
        c1=c2;
        c2=c3;
        c3=c4;
        sayac++;
        } /*while sonu*/
fclose(fp1);
/*3 uzunluklu*/
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048572.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                }
        else
                c5= getc(fp1);
        if((c1=='1')&&(c2=='0')&&(c3=='0')&&(c4=='0')&&(c5=='1'))
                n03++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        sayac++;
        } /*while sonu*/
fclose(fp1);
```

```
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048572.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                }
        else
                c5= getc(fp1);
        if((c1=='0')&&(c2=='1')&&(c3=='1')&&(c4=='1')&&(c5=='0'))
                n13++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        sayac++;
        } /*while sonu*/
fclose(fp1);
/*4 uzunluklu*/
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048571.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                }
        else
                c6= getc(fp1);
        if((c1=='1')&&(c2=='0')&&(c3=='0')&&(c4=='0')&&(c5=='0')&&(c6=='1'))
                n04++;
        c1=c2;
        c2=c3;
        c3=c4;
```

```
        c4=c5;
        c5=c6;
        sayac++;
        } /*while sonu*/
fclose(fp1);
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }

while(sayac<1048571.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                }
        else
                c6= getc(fp1);
        if((c1=='0')&&(c2=='1')&&(c3=='1')&&(c4=='1')&&(c5=='1')&&(c6=='0'))
                n14++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        sayac++;
        } /*while sonu*/
fclose(fp1);
/*5 uzunluklu*/
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048570.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
```

```
                c7=getc(fp1);
                }
        else
                c7= getc(fp1);
        if((c1=='1')&&(c2=='0')&&(c3=='0')&&(c4=='0')&&(c5=='0')&&(c6=='0')&
&(c7=='1'))
                n05++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        sayac++;
        } /*while sonu*/
fclose(fp1);
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048570.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                }
        else
                c7= getc(fp1);
        if((c1=='0')&&(c2=='1')&&(c3=='1')&&(c4=='1')&&(c5=='1')&&(c6=='1')&
&(c7=='0'))
                n15++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        sayac++;
        } /*while sonu*/
fclose(fp1);
/*6 uzunluklu*/
sayac=0.0;
```

```
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048569.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                }
        else
                c8= getc(fp1);
        if((c1=='1')&&(c2=='0')&&(c3=='0')&&(c4=='0')&&(c5=='0')&&(c6=='0')&
&(c7=='0')&&(c8=='1'))
                n06++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        sayac++;
        } /*while sonu*/
fclose(fp1);
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048569.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                }
```

```
        else
                c8= getc(fp1);
        if((c1=='0')&&(c2=='1')&&(c3=='1')&&(c4=='1')&&(c5=='1')&&(c6=='1')&
&(c7=='1')&&(c8=='0'))
                n16++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        sayac++;
        } /*while sonu*/
fclose(fp1);
/*7 uzunluklu*/
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048568.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                }
        else
                c9= getc(fp1);
        if((c1=='1')&&(c2=='0')&&(c3=='0')&&(c4=='0')&&(c5=='0')&&(c6=='0')&
&(c7=='0')&&(c8=='0')&&(c9=='1'))
                n07++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        sayac++;
```

```
        } /*while sonu*/
fclose(fp1);
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048568.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                }
        else
        c9= getc(fp1);
        if((c1=='0')&&(c2=='1')&&(c3=='1')&&(c4=='1')&&(c5=='1')&&(c6=='1')&
&(c7=='1')&&(c8=='1')&&(c9=='0'))
                n17++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        sayac++;
        } /*while sonu*/
fclose(fp1);
/*8 uzunluklu*/
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048567.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
```

```
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                }
        else
                c10= getc(fp1);
        if((c1=='1')&&(c2=='0')&&(c3=='0')&&(c4=='0')&&(c5=='0')&&(c6=='0')&
&(c7=='0')&&(c8=='0')&&(c9=='0')&&(c10=='1'))
                n08++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
        sayac++;
        } /*while sonu*/
fclose(fp1);
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048567.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                }
        else
                c10= getc(fp1);
        if((c1=='0')&&(c2=='1')&&(c3=='1')&&(c4=='1')&&(c5=='1')&&(c6=='1')&
&(c7=='1')&&(c8=='1')&&(c9=='1')&&(c10=='0'))
```

```
                n18++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
        sayac++;
        } /*while sonu*/
fclose(fp1);
/*9 uzunluklu*/
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048566.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                c11=getc(fp1);
                }
        else
                c11= getc(fp1);
        if((c1=='1')&&(c2=='0')&&(c3=='0')&&(c4=='0')&&(c5=='0')&&(c6=='0')&
&(c7=='0')&&(c8=='0')&&(c9=='0')&&(c10=='0')&&(c11=='1'))
                n09++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
```

```
        c10=c11;
        sayac++;
        } /*while sonu*/
fclose(fp1);
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048566.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                c11=getc(fp1);
                }
        else
                c11= getc(fp1);
        if((c1=='0')&&(c2=='1')&&(c3=='1')&&(c4=='1')&&(c5=='1')&&(c6=='1')&
&(c7=='1')&&(c8=='1')&&(c9=='1')&&(c10=='1')&&(c11=='0'))
                n19++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
        c10=c11;
        sayac++;
        } /*while sonu*/
fclose(fp1);
/*10 uzunluklu*/
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
```

```
        }
while(sayac<1048565.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                c11=getc(fp1);
                c12=getc(fp1);
                }
        else
                c12= getc(fp1);
        if((c1=='1')&&(c2=='0')&&(c3=='0')&&(c4=='0')&&(c5=='0')&&(c6=='0')&
&(c7=='0')&&(c8=='0')&&(c9=='0')&&(c10=='0')&&(c11=='0')&&(c12=='1'))
                n010++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
        c10=c11;
        c11=c12;
        sayac++;
        } /*while sonu*/
fclose(fp1);
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048565.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
```

```
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                c11=getc(fp1);
                c12=getc(fp1);
                }
        else
                c12= getc(fp1);
        if((c1=='0')&&(c2=='1')&&(c3=='1')&&(c4=='1')&&(c5=='1')&&(c6=='1')&
&(c7=='1')&&(c8=='1')&&(c9=='1')&&(c10=='1')&&(c11=='1')&&(c12=='0'))
                n110++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
        c10=c11;
        c11=c12;
        sayac++;
        } /*while sonu*/
fclose(fp1);
/*11 uzunluklu*/
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048564.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                c11=getc(fp1);
                c12=getc(fp1);
```

```
                c13=getc(fp1);
                }
        else
                c13= getc(fp1);
        if((c1=='1')&&(c2=='0')&&(c3=='0')&&(c4=='0')&&(c5=='0')&&(c6=='0')&
&(c7=='0')&&(c8=='0')&&(c9=='0')&&(c10=='0')&&(c11=='0')&&(c12=='0')&&(c1
3=='1'))
                n011++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
        c10=c11;
        c11=c12;
        c12=c13;
        sayac++;
        } /*while sonu*/
fclose(fp1);
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048564.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                c11=getc(fp1);
                c12=getc(fp1);
                c13=getc(fp1);
                }
        else
                c13= getc(fp1);
```

```
        if((c1=='0')&&(c2=='1')&&(c3=='1')&&(c4=='1')&&(c5=='1')&&(c6=='1')&
&(c7=='1')&&(c8=='1')&&(c9=='1')&&(c10=='1')&&(c11=='1')&&(c12=='1')&&(c1
3=='0'))
                n111++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
        c10=c11;
        c11=c12;
        c12=c13;
        sayac++;
        } /*while sonu*/
fclose(fp1);
/*12 uzunluklu*/
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048563.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                c11=getc(fp1);
                c12=getc(fp1);
                c13=getc(fp1);
                c14=getc(fp1);
                }
        else
        c14= getc(fp1);
        if((c1=='0')&&(c2=='1')&&(c3=='1')&&(c4=='1')&&(c5=='1')&&(c6=='1')&
&(c7=='1')&&(c8=='1')&&(c9=='1')&&(c10=='1')&&(c11=='1')&&(c12=='1')&&(c1
3=='1')&&(c14=='0'))
```

```
                n112++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
        c10=c11;
        c11=c12;
        c12=c13;
        c13=c14;
        sayac++;
        } /*while sonu*/
fclose(fp1);
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048563.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                c11=getc(fp1);
                c12=getc(fp1);
                c13=getc(fp1);
                c14=getc(fp1);
                }
        else
                c14= getc(fp1);
        if((c1=='1')&&(c2=='0')&&(c3=='0')&&(c4=='0')&&(c5=='0')&&(c6=='0')&
&(c7=='0')&&(c8=='0')&&(c9=='0')&&(c10=='0')&&(c11=='0')&&(c12=='0')&&(c1
3=='0')&&(c14=='1'))
                n012++;
        c1=c2;
        c2=c3;
```

```
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
        c10=c11;
        c11=c12;
        c12=c13;
        c13=c14;
        sayac++;
        } /*while sonu*/
fclose(fp1);
/*13 uzunluklu*/
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048562.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                c11=getc(fp1);
                c12=getc(fp1);
                c13=getc(fp1);
                c14=getc(fp1);
                c15=getc(fp1);
                }
        else
                c15= getc(fp1);
        if((c1=='0')&&(c2=='1')&&(c3=='1')&&(c4=='1')&&(c5=='1')&&(c6=='1')&
&(c7=='1')&&(c8=='1')&&(c9=='1')&&(c10=='1')&&(c11=='1')&&(c12=='1')&&(c1
3=='1')&&(c14=='1')&&(c15=='0'))
                n113++;
        c1=c2;
        c2=c3;
        c3=c4;
```

```
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
        c10=c11;
        c11=c12;
        c12=c13;
        c13=c14;
        c14=c15;
        sayac++;
        } /*while sonu*/
fclose(fp1);
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048562.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                c11=getc(fp1);
                c12=getc(fp1);
                c13=getc(fp1);
                c14=getc(fp1);
                c15=getc(fp1);
                }
        else
                c15= getc(fp1);
        if((c1=='1')&&(c2=='0')&&(c3=='0')&&(c4=='0')&&(c5=='0')&&(c6=='0')&
&(c7=='0')&&(c8=='0')&&(c9=='0')&&(c10=='0')&&(c11=='0')&&(c12=='0')&&(c1
3=='0')&&(c14=='0')&&(c15=='1'))
                n013++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
```

```
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
        c10=c11;
        c11=c12;
        c12=c13;
        c13=c14;
        c14=c15;
        sayac++;
        } /*while sonu*/
fclose(fp1);
/*14 uzunluklu*/
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048561.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                c11=getc(fp1);
                c12=getc(fp1);
                c13=getc(fp1);
                c14=getc(fp1);
                c15=getc(fp1);
                c16=getc(fp1);
                }
        else
                c16= getc(fp1);
        if((c1=='0')&&(c2=='1')&&(c3=='1')&&(c4=='1')&&(c5=='1')&&(c6=='1')&
&(c7=='1')&&(c8=='1')&&(c9=='1')&&(c10=='1')&&(c11=='1')&&(c12=='1')&&(c1
3=='1')&&(c14=='1')&&(c15=='1')&&(c16=='0'))
                n114++;
        c1=c2;
        c2=c3;
        c3=c4;
```

```
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
        c10=c11;
        c11=c12;
        c12=c13;
        c13=c14;
        c14=c15;
        c15=c16;
        sayac++;
        } /*while sonu*/
fclose(fp1);
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048561.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                c11=getc(fp1);
                c12=getc(fp1);
                c13=getc(fp1);
                c14=getc(fp1);
                c15=getc(fp1);
                c16=getc(fp1);
                }
        else
                c16= getc(fp1);
        if((c1=='1')&&(c2=='0')&&(c3=='0')&&(c4=='0')&&(c5=='0')&&(c6=='0')&
&(c7=='0')&&(c8=='0')&&(c9=='0')&&(c10=='0')&&(c11=='0')&&(c12=='0')&&(c1
3=='0')&&(c14=='0')&&(c15=='0')&&(c16=='1'))
                n014++;
        c1=c2;
        c2=c3;
```

```
            c3=c4;
            c4=c5;
            c5=c6;
            c6=c7;
            c7=c8;
            c8=c9;
            c9=c10;
            c10=c11;
            c11=c12;
            c12=c13;
            c13=c14;
            c14=c15;
            c15=c16;
            sayac++;
            } /*while sonu*/
fclose(fp1);
/*15 uzunluklu*/
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
            printf("\nCan not open as.dat");
            exit(1);
            }
while(sayac<1048560.0){
            if(sayac==0.0){
                        c1=getc(fp1);
                        c2=getc(fp1);
                        c3=getc(fp1);
                        c4=getc(fp1);
                        c5=getc(fp1);
                        c6=getc(fp1);
                        c7=getc(fp1);
                        c8=getc(fp1);
                        c9=getc(fp1);
                        c10=getc(fp1);
                        c11=getc(fp1);
                        c12=getc(fp1);
                        c13=getc(fp1);
                        c14=getc(fp1);
                        c15=getc(fp1);
                        c16=getc(fp1);
                        c17=getc(fp1);
                        }
            else
                        c17= getc(fp1);
            if((c1=='0')&&(c2=='1')&&(c3=='1')&&(c4=='1')&&(c5=='1')&&(c6=='1')&
&(c7=='1')&&(c8=='1')&&(c9=='1')&&(c10=='1')&&(c11=='1')&&(c12=='1')&&(c1
3=='1')&&(c14=='1')&&(c15=='1')&&(c16=='1')&&(c17=='0'))
```

```
                n115++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
        c10=c11;
        c11=c12;
        c12=c13;
        c13=c14;
        c14=c15;
        c15=c16;
        c16=c17;
        sayac++;
        } /*while sonu*/
fclose(fp1);
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048560.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                c11=getc(fp1);
                c12=getc(fp1);
                c13=getc(fp1);
                c14=getc(fp1);
                c15=getc(fp1);
                c16=getc(fp1);
                c17=getc(fp1);
                }
        else
                c17= getc(fp1);
```

```
        if((c1=='1')&&(c2=='0')&&(c3=='0')&&(c4=='0')&&(c5=='0')&&(c6=='0')&
&(c7=='0')&&(c8=='0')&&(c9=='0')&&(c10=='0')&&(c11=='0')&&(c12=='0')&&(c1
3=='0')&&(c14=='0')&&(c15=='0')&&(c16=='0')&&(c17=='1'))
                n015++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
        c10=c11;
        c11=c12;
        c12=c13;
        c13=c14;
        c14=c15;
        c15=c16;
        c16=c17;
        sayac++;
        } /*while sonu*/
fclose(fp1);
/*16 uzunluklu*/
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048559.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                c11=getc(fp1);
                c12=getc(fp1);
                c13=getc(fp1);
                c14=getc(fp1);
                c15=getc(fp1);
                c16=getc(fp1);
```

```
                c17=getc(fp1);
                c18=getc(fp1);
                }
        else
                c18= getc(fp1);
        if((c1=='0')&&(c2=='1')&&(c3=='1')&&(c4=='1')&&(c5=='1')&&(c6=='1')&
&(c7=='1')&&(c8=='1')&&(c9=='1')&&(c10=='1')&&(c11=='1')&&(c12=='1')&&(c1
3=='1')&&(c14=='1')&&(c15=='1')&&(c16=='1')&&(c17=='1')&&(c18=='0'))
                n116++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
        c10=c11;
        c11=c12;
        c12=c13;
        c13=c14;
        c14=c15;
        c15=c16;
        c16=c17;
        c17=c18;
        sayac++;
        } /*while sonu*/
fclose(fp1);
sayac=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<1048559.0){
        if(sayac==0.0){
                c1=getc(fp1);
                c2=getc(fp1);
                c3=getc(fp1);
                c4=getc(fp1);
                c5=getc(fp1);
                c6=getc(fp1);
                c7=getc(fp1);
                c8=getc(fp1);
                c9=getc(fp1);
                c10=getc(fp1);
                c11=getc(fp1);
```

```
                c12=getc(fp1);
                c13=getc(fp1);
                c14=getc(fp1);
                c15=getc(fp1);
                c16=getc(fp1);
                c17=getc(fp1);
                c18=getc(fp1);
                }
        else
                c18= getc(fp1);
        if((c1=='1')&&(c2=='0')&&(c3=='0')&&(c4=='0')&&(c5=='0')&&(c6=='0')&
&(c7=='0')&&(c8=='0')&&(c9=='0')&&(c10=='0')&&(c11=='0')&&(c12=='0')&&(c1
3=='0')&&(c14=='0')&&(c15=='0')&&(c16=='0')&&(c17=='0')&&(c18=='1'))
                n016++;
        c1=c2;
        c2=c3;
        c3=c4;
        c4=c5;
        c5=c6;
        c6=c7;
        c7=c8;
        c8=c9;
        c9=c10;
        c10=c11;
        c11=c12;
        c12=c13;
        c13=c14;
        c14=c15;
        c15=c16;
        c16=c17;
        c17=c18;
        sayac++;
        } /*while sonu*/
fclose(fp1);
hareket1=n01+n02+n03+n04+n05+n06+n07+n08+n09+n010+n011+n012+n013+n01
4+n015+n016;
hareket2=n11+n12+n13+n14+n15+n16+n17+n18+n19+n110+n111+n112+n113+n11
4+n115+n116;
hareket=hareket1+hareket2;
dizi[0]=hareket*(hareket-1.0)/(2*uzun-2.0);
dizi[1]=dizi[0]*(uzun-hareket)/(uzun-2.0);
dizi[2]=dizi[1]*(uzun-hareket-1.0)/(uzun-3.0);
dizi[3]=dizi[2]*(uzun-hareket-2.0)/(uzun-4.0);
dizi[4]=dizi[3]*(uzun-hareket-3.0)/(uzun-5.0);
dizi[5]=dizi[4]*(uzun-hareket-4.0)/(uzun-6.0);
dizi[6]=dizi[5]*(uzun-hareket-5.0)/(uzun-7.0);
dizi[7]=dizi[6]*(uzun-hareket-6.0)/(uzun-8.0);
dizi[8]=dizi[7]*(uzun-hareket-7.0)/(uzun-9.0);
```

```
dizi[9]=dizi[8]*(uzun-hareket-8.0)/(uzun-10.0);
dizi[10]=dizi[9]*(uzun-hareket-9.0)/(uzun-11.0);
dizi[11]=dizi[10]*(uzun-hareket-10.0)/(uzun-12.0);
dizi[12]=dizi[11]*(uzun-hareket-11.0)/(uzun-13.0);
dizi[13]=dizi[12]*(uzun-hareket-12.0)/(uzun-14.0);
dizi[14]=dizi[13]*(uzun-hareket-13.0)/(uzun-15.0);
for(say=0;say<15;say++)
        top1=top1+dizi[say];
dizi[15]=hareket/2-top1;
dizi1[0]=n01;
dizi1[1]=n02;
dizi1[2]=n03;
dizi1[3]=n04;
dizi1[4]=n05;
dizi1[5]=n06;
dizi1[6]=n07;
dizi1[7]=n08;
dizi1[8]=n09;
dizi1[9]=n010;
dizi1[10]=n011;
dizi1[11]=n012;
dizi1[12]=n013;
dizi1[13]=n014;
dizi1[14]=n015;
dizi1[15]=n016;
dizi2[0]=n11;
dizi2[1]=n12;
dizi2[2]=n13;
dizi2[3]=n14;
dizi2[4]=n15;
dizi2[5]=n16;
dizi2[6]=n17;
dizi2[7]=n18;
dizi2[8]=n19;
dizi2[9]=n110;
dizi2[10]=n111;
dizi2[11]=n112;
dizi2[12]=n113;
dizi2[13]=n114;
dizi2[14]=n115;
dizi2[15]=n116;
for(say=0;say<16;say++){
        ist1=ist1+(dizi1[say]-dizi[say])*(dizi1[say]-dizi[say])/dizi[say];
        ist2=ist2+(dizi2[say]-dizi[say])*(dizi1[say]-dizi[say])/dizi[say];
        }
ist=ist1+ist2;
printf("\nn01=%lf",n01);
printf("\nn11=%lf",n11);
```

```
printf("\nn02=%lf",n02);
printf("\nn12=%lf",n12);
printf("\nn03=%lf",n03);
printf("\nn13=%lf",n13);
printf("\nn04=%lf",n04);
printf("\nn14=%lf",n14);
printf("\nn05=%lf",n05);
printf("\nn15=%lf",n15);
printf("\nn06=%lf",n06);
printf("\nn16=%lf",n16);
printf("\nn07=%lf",n07);
printf("\nn17=%lf",n17);
printf("\nn08=%lf",n08);
printf("\nn18=%lf",n18);
printf("\nn09=%lf",n09);
printf("\nn19=%lf",n19);
getchar();
printf("\nn010=%lf",n010);
printf("\nn110=%lf",n110);
printf("\nn011=%lf",n011);
printf("\nn111=%lf",n111);
printf("\nn012=%lf",n012);
printf("\nn112=%lf",n112);
printf("\nn013=%lf",n013);
printf("\nn113=%lf",n113);
printf("\nn014=%lf",n014);
printf("\nn114=%lf",n114);
printf("\nn015=%lf",n015);
printf("\nn115=%lf",n115);
printf("\nn016=%lf",n016);
printf("\nn116=%lf",n116);
getchar();
printf("\nhareket=%lf  ist=%lf",hareket,ist);
return 0;
}/*main sonu*/
```

B. 4 m = 2 İçin Poker Testi

```
/*1 ve 0 sayılarından oluşan dosyaya, m = 2 için poker testi uygular*/
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
FILE *fp1;
main()
{
double n00=0.0,n01=0.0;
double n10=0.0,n11=0.0;
```

```
double ist;
double sayac=1.0;
char ch1,ch2;
clrscr();
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<=1048575){
        ch1=getc(fp1);
        ch2=getc(fp1);
        if((ch1-'0'==0)&&(ch2-'0'==0))
                n00++;
        if((ch1-'0'==0)&&(ch2-'0'==1))
                n01++;
        if((ch1-'0'==1)&&(ch2-'0'==0))
                n10++;
        if((ch1-'0'==1)&&(ch2-'0'==1))
                n11++;
        sayac++;
        }    /*while sonu*/
ist=(n00*n00+n01*n01+n10*n10+n11*n11)*2.0*2.0/(512.0*1024.0)-
(512.0*1024.0);
printf("n00=%lf n01=%lf n10=%lf n11=%lf ist=%lf",n00,n01,n10,n11,ist);
fclose(fp1);
return 0;
}/*main sonu*/
```

B. 5 m = 3 İçin Poker Testi

```
/* 1 ve 0 sayılarından oluşan dosyaya m = 3 için poker testi uygular*/
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
FILE *fp1;
main()
{
double n000=0.0,n001=0.0,n010=0.0,n011=0.0;
double n100=0.0,n101=0.0,n110=0.0,n111=0.0;
double ist,ist1;
double ist2;
double sayac=1.0;
char ch1,ch2;
char ch3;
clrscr();
fp1=fopen("as.dat","rb");
```

```
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<=1048573){
        ch1=getc(fp1);
        ch2=getc(fp1);
        ch3=getc(fp1);
        if((ch1-'0'==0)&&(ch2-'0'==0)&&(ch3-'0'==0))
                n000++;
        if((ch1-'0'==0)&&(ch2-'0'==0)&&(ch3-'0'==1))
                n001++;
        if((ch1-'0'==0)&&(ch2-'0'==1)&&(ch3-'0'==0))
                n010++;
        if((ch1-'0'==0)&&(ch2-'0'==1)&&(ch3-'0'==1))
                n011++;
        if((ch1-'0'==1)&&(ch2-'0'==0)&&(ch3-'0'==0))
                n100++;
        if((ch1-'0'==1)&&(ch2-'0'==0)&&(ch3-'0'==1))
                n101++;
        if((ch1-'0'==1)&&(ch2-'0'==1)&&(ch3-'0'==0))
                n110++;
        if((ch1-'0'==1)&&(ch2-'0'==1)&&(ch3-'0'==1))
                n111++;
        sayac++;
        }    /*while sonu*/
ist1=(n000*n000+n001*n001+n010*n010+n011*n011);
ist2=(n100*n100+n101*n101+n110*n110+n111*n111);
ist=(ist1+ist2)*8.0/349525.0-349525.0;
printf("\nn000=%lf n001=%lf n010=%lf n011=%lf ",n000,n001,n010,n011);
printf("\nn100=%lf n101=%lf n110=%lf n111=%lf ",n100,n101,n110,n111);
printf("\nist=%lf",ist);
fclose(fp1);
return 0;
}/*main sonu*/
```

B. 6 m = 4 İçin Poker Testi

```
/*1 ve 0 sayılarından oluşan dosyaya, m = 4 için poker testi uygular*/
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
FILE *fp1;
main()
{
double n0=0.0,n1=0.0;
double n2=0.0,n3=0.0;
```

```
double n4=0.0,n5=0.0,n6=0.0,n7=0.0,n8=0.0;
double n9=0.0,n10=0.0,n11=0.0,n12=0.0;
double n13=0.0,n14=0.0,n15=0.0;
double ist,ist1,ist2,ist3,ist4;
double sayac=1.0;
char ch1,ch2,ch3,ch4;
clrscr();
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open as.dat");
        exit(1);
        }
while(sayac<=1048573){
        ch1=getc(fp1);
        ch2=getc(fp1);
        ch3=getc(fp1);
        ch4=getc(fp1);
        if((ch1-'0'==0)&&(ch2-'0'==0)&&(ch3-'0'==0)&&(ch4-'0'==0))
                n0++;
        if((ch1-'0'==0)&&(ch2-'0'==0)&&(ch3-'0'==0)&&(ch4-'0'==1))
                n1++;
        if((ch1-'0'==0)&&(ch2-'0'==0)&&(ch3-'0'==1)&&(ch4-'0'==0))
                n2++;
        if((ch1-'0'==0)&&(ch2-'0'==0)&&(ch3-'0'==1)&&(ch4-'0'==1))
                n3++;
        if((ch1-'0'==0)&&(ch2-'0'==1)&&(ch3-'0'==0)&&(ch4-'0'==0))
                n4++;
        if((ch1-'0'==0)&&(ch2-'0'==1)&&(ch3-'0'==0)&&(ch4-'0'==1))
                n5++;
        if((ch1-'0'==0)&&(ch2-'0'==1)&&(ch3-'0'==1)&&(ch4-'0'==0))
                n6++;
        if((ch1-'0'==0)&&(ch2-'0'==1)&&(ch3-'0'==1)&&(ch4-'0'==1))
                n7++;
        if((ch1-'0'==1)&&(ch2-'0'==0)&&(ch3-'0'==0)&&(ch4-'0'==0))
                n8++;
        if((ch1-'0'==1)&&(ch2-'0'==0)&&(ch3-'0'==0)&&(ch4-'0'==1))
                n9++;
        if((ch1-'0'==1)&&(ch2-'0'==0)&&(ch3-'0'==1)&&(ch4-'0'==0))
                n10++;
        if((ch1-'0'==1)&&(ch2-'0'==0)&&(ch3-'0'==1)&&(ch4-'0'==1))
                n11++;
        if((ch1-'0'==1)&&(ch2-'0'==1)&&(ch3-'0'==0)&&(ch4-'0'==0))
                n12++;
        if((ch1-'0'==1)&&(ch2-'0'==1)&&(ch3-'0'==0)&&(ch4-'0'==1))
                n13++;
        if((ch1-'0'==1)&&(ch2-'0'==1)&&(ch3-'0'==1)&&(ch4-'0'==0))
                n14++;
        if((ch1-'0'==1)&&(ch2-'0'==1)&&(ch3-'0'==1)&&(ch4-'0'==1))
```

```
                n15++;
        sayac++;
        }    /*while sonu*/
ist1=(n0*n0+n1*n1+n2*n2+n3*n3);
ist2=(n4*n4+n5*n5+n6*n6+n7*n7);
ist3=(n8*n8+n9*n9+n10*n10+n11*n11);
ist4=(n12*n12+n13*n13+n14*n14+n15*n15);
ist=(ist1+ist2+ist3+ist4)*16.0/262144.0-262144.0;
printf("\nn0=%lf n1=%lf n2=%lf n3=%lf n4=%lf ",n0,n1,n2,n3,n4);
printf("\nn5=%lf n6=%lf n7=%lf n8=%lf ",n5,n6,n7,n8);
printf("\nn9=%lf n10=%lf n11=%lf n12=%lf ",n9,n10,n11,n12);
printf("\nn13=%lf n14=%lf n15=%lf ",n13,n14,n15);
printf("\nist=%lf",ist);
fclose(fp1);
return 0;
}/*main sonu*/
```

B. 7 m = 8 İçin Poker Testi

```
/*Karakterlerden oluşan dosyaya m = 8 için poker testi uygular*/
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<math.h>
FILE *fp1;
main()
{
int ch[256];
double n[256];
clrscr();
int i;
double sayac=1.0;
int c;
double top=0.0;
double ist;
double gist;
for(i=0;i<=255;i++)
        ch[i]=i;
for(i=0;i<=255;i++)
        n[i]=0.0;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCan not open chua.dat");
        exit(1);
        }
while(sayac<=131072.0){
```

```
        c=getc(fp1);
        for(i=0;i<=255;i++){
                if(c==ch[i])
                        n[i]++;
                }
        sayac++;
        }/*while sonu*/
fclose(fp1);
for(i=0;i<=255;i++)
        top=top+n[i]*n[i];
ist=256.0*top/131072.0-131072.0;
gist=255.0+sqrt(2.0*255.0)*1.64+ 2*1.64*1.64/3.0-2.0/3.0;
printf("\nist=%lf gist=%lf",ist,gist);
return 0;
}/*main sonu*/
```

B. 8 Öz - İlişki Testi

```
/* d'nin farklı değerleri için 1 ve 0 sayılarından oluşan aynı iki dosyayı kullanarak,
diziye öz - ilişki testi uygular */
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<math.h>
FILE *fp1,*fp2;
main()
{
char c1,c2;
double a1=0.0,a2,a3;
int xor;
double d=10.0;
double n=1048576.0;
double sayac=1.0;
double temp = 1.0;
int i;
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCannot open as.dat");
        exit(1);
        }
fp2=fopen("as1.dat","wb");
if(fp2==NULL){
        printf("\nCannot open as1.dat");
        exit(1);
        }
while(temp <= 1048576.0){
```

```
        c1=getc(fp1);
        putc(c1,fp2);
        temp++;
        }
fclose(fp1);
fclose(fp2);
fp1=fopen("as.dat","rb");
if(fp1==NULL){
        printf("\nCannot open as.dat");
        exit(1);
        }
fp2=fopen("as1.dat","rb");
if(fp2==NULL){
        printf("\nCannot open as1.dat");
        exit(1);
        }
while(sayac<=(n-d)){
        c1=getc(fp1);
        if(sayac==1.0){
                for(i=1;i<=d+1;i++)
                        c2=getc(fp2);
                }
        else
                c2=getc(fp2);
        xor=(c1-'0')^(c2-'0');
        a1=a1+(double)(xor);
        sayac++;
        }
fclose(fp1);
fclose(fp2);
a2=(n-d)/2.0+1.96*sqrt(n-d)/2.0;
a3=(n-d)/2.0-1.96*sqrt(n-d)/2.0;
printf("\n a1=%lf  a2=%lf  a3=%lf",a1,a2,a3);
if((a1>a2)||(a1<a3))
        printf("diziyi reddet");
return 0;
}/*main sonu*/
```

EK C

CHUA DEVRESİ İLE ELDE EDİLEN 5 DİZİNİN İSTATİSTİKSEL RASGELELİK TEST SONUCLARI

C. 1 Frekans Testi

n0, n uzunluğundaki dizide gözlemlenen sıfırların sayısı, n1 ise n uzunluğundaki dizide gözlemlenen birlerin sayısıdır. n = 1,048,576 dır ve dizinin rededilmesi için, hesaplanan istatistiğin 3.841'den büyük olması gerekmektedir. İlk koşul değerleri, vc1= 0.099910, vc2 = 0.199910 ve il = -0.099910 şeklinde alınarak ilk dizi oluşturuldu. Daha sonraki diziler, vc1 + 1E-5, vc2 + 1E-5 ve il = il - 1E-5 koşullarına uyularak oluşturuldu. İcra edilen tüm testlerde bu ilk koşullarla meydane getirilen diziler kullanıldı.

Tablo C.1. Frekans testi sonuçları.

	n0	n1	İst
1.Dizi	524497	524079	0.166630
2.Dizi	524900	523676	1.428772
3.Dizi	523935	524641	0.475346
4.Dizi	524501	524075	0.173069
5.Dizi	524356	524220	0.017639

C. 2 Seri Test

n0, dizide gözlenen 00 çiftlerinin sayısı, n1, dizide gözlenen 01 çiftlerinin sayı, n2, dizide gözlenen 10 çiftlerinin sayısı ve n3, dizide gözlenen 11 çiflerinin sayısıdır. Hesaplanan istatistik 5.991'den büyükse dizi reddedilir.

Tablo C. 2. Seri test sonuçları.

n0	n1	n2	n3	İst
262415	262081	262082	261997	0.225761
262893	262007	262007	261668	1.718794
261744	262191	262190	262450	0.508697
261902	262598	262598	261477	3.319994
262340	262075	262015	262205	0.270061

C. 3 Poker Testi

C. 3. 1 m = 8 için Poker Testi

Hesaplanan istatistik 293.162815'den büyükse dizi reddedilir.

Tablo C. 3. m = 8 için poker testi sonuçları.

	İst.
1.Dizi	231.710938
2.Dizi	268.511719
3.Dizi	230.765625
4.Dizi	253.386719
5.Dizi	276.562500

C. 3. 2 m = 4 için Poker Testi

Hesaplanan istatistik 25.00'dan büyük ise dizi reddedilir.

Tablo C. 4. m = 4 için poker testi sonuçları.

	İst.
1.Dizi	17.899
2.Dizi	19.122
3.Dizi	16.374
4.Dizi	15.845
5.Dizi	6.402

C. 3. 3 m = 3 için Poker Testi

Hesaplanan istatistik 14.067'den büyükse dizi reddedilir.

Tablo C. 5. m = 3 için poker testi sonuçları.

	İst.
1.Dizi	2.217
2.Dizi	4.494
3.Dizi	3.282
4.Dizi	8.408
5.Dizi	2.644

C. 3. 3 m = 2 için Poker Testi

Hesaplanan istatistik 7.815'den büyük ise, dizi reddedilir.

Tablo C. 6. m = 2 için poker testi sonuçları.

	İst.
1.Dizi	0.232
2.Dizi	1.738
3.Dizi	1.929
4.Dizi	7.127
5.Dizi	0.121

C. 4 Hareket Testi

Hesaplanan istatistik 42.56'dan büyükse dizi reddedilir.

Tablo C. 7. Hareket testi sonuçları.

	İst.	Hareket
1.Dizi	13.939	524152
2.Dizi	13.948	524004
3.Dizi	19.721	524374
4.Dizi	12.896	525188
5.Dizi	21.286	524023

C. 5 Özilişki Testi

C. 5. 1 d = 1 için Özilişki Testi

d = 1 için hesaplanan istatistik 525291.019'dan büyük ise veya 523283.98'den küçük ise, dizi reddedilir

Tablo C. 8. d = 1 için özilişki testi sonuçları.

	İst.
1.Dizi	524251
2.Dizi	524014
3.Dizi	524381
4.Dizi	525196
5.Dizi	524030

C. 5. 2 d = 2 için Özilişki Testi

d = 2 için hesaplanan istatistik 525290.519'dan büyük ise veya 523283.48'den küçük ise, dizi reddedilir

Tablo C. 9. d = 2 için özilişki testi sonuçları.

	İst.
1.Dizi	523919
2.Dizi	525384
3.Dizi	524335
4.Dizi	523817
5.Dizi	523827

C. 5. 3 d = 10 için Özilişki Testi

d = 2 için hesaplanan istatistik 525286.515'den büyük ise veya 523279.48'den küçük ise, dizi reddedilir

Tablo C. 10. d = 10 için özilişki testi sonuçları.

	İst.
1.Dizi	523941
2.Dizi	524421
3.Dizi	524156
4.Dizi	523954
5.Dizi	524108

C. 5. 4 d = 1000 için Özilişki Testi

d = 1000 için hesaplanan istatistik 524791.041'den büyük ise veya 522784.95'ten küçük ise, dizi reddedilir

Tablo C. 11. d =1000 için özilişki testi sonuçları.

	İst.
1.Dizi	524054
2.Dizi	523489
3.Dizi	523336
4.Dizi	523549
5.Dizi	523826

C. 5. 5 d = 10000 için Özilişki Testi

d = 10000 için hesaplanan istatistik 520286.723'ten büyük ise veya 518289.276'dan küçük ise, dizi reddedilir

Tablo C. 12. d =10000 için özilişki testi sonuçları.

	İst.
1.Dizi	519080
2.Dizi	518669
3.Dizi	518927
4.Dizi	519528
5.Dizi	518767

EK D

LORENZ SİSTEMİ İLE ELDE EDİLEN 5 DİZİNİN İSTATİSTİKSEL RASGELELİK TEST SONUCLARI

D. 1 Frekans Testi

n0, n uzunluğundaki dizide gözlemlenen sıfırların sayısı, n1 ise n uzunluğundaki dizide gözlemlenen birlerin sayısıdır. n = 1,048,576 dır ve dizinin rededilmesi için, hesaplanan istatistiğin 3.841'den büyük olması gerekmektedir. İlk koşul değerleri, x = 0.1, y = -0.193 ve il = 0.701 şeklinde alınarak ilk dizi oluşturuldu. Daha sonraki diziler, x - 1E-5, y + 1E-5 ve z = z - 1E-5 koşullarına uyularak oluşturuldu. İcra edilen tüm testlerde bu ilk koşullarla meydane getirilen diziler kullanıldı.

Tablo D.1. Frekans testi sonuçları.

	n0	n1	İst
1.Dizi	524914	523662	1.4948
2.Dizi	523865	524711	0.682560
3.Dizi	524031	524545	0.251957
4.Dizi	523628	524948	1.661682
5.Dizi	523957	524619	0.417942

D. 2 Seri Test

n0, dizide gözlenen 00 çiftlerinin sayısı, n1, dizide gözlenen 01 çiftlerinin sayı, n2, dizide gözlenen 10 çiftlerinin sayısı ve n3, dizide gözlenen 11 çiflerinin sayısıdır. Hesaplanan istatistik 5.991'den büyükse dizi reddedilir.

Tablo D. 2. Seri test sonuçları.

n0	n1	n2	n3	İst
262644	262269	262270	261392	1.736181
261579	262285	262285	262426	0.990227
261743	262287	262288	262257	0.567268
261149	262479	262478	262469	3.371552
261969	261988	261988	262630	0.785568

D. 3 Poker Testi

D. 3. 1 m = 8 için Poker Testi

Hesaplanan istatistik 293.162815'den büyükse dizi reddedilir.

Tablo D. 3. m = 8 için poker testi sonuçları.

	İst.
1.Dizi	231.660156
2.Dizi	231.492188
3.Dizi	258.449219
4.Dizi	257.070312
5.Dizi	231.464844

D. 3. 2 m = 4 için Poker Testi

Hesaplanan istatistik 25.00'dan büyük ise dizi reddedilir.

Tablo D. 4. m = 4 için poker testi sonuçları.

	İst.
1.Dizi	17.383
2.Dizi	14.758
3.Dizi	14.611
4.Dizi	16.054
5.Dizi	18.166

D. 3. 3 m = 3 için Poker Testi

Hesaplanan istatistik 14.067'den büyükse dizi reddedilir. Tablo D.5'ten görüldüğü gibi beş farklı başlangıç değeri için Lorenz sistemi ile elde edilen dizilerin hiçbiri, m = 3 için poker testinden kalmamıştır.

Tablo D. 5. m = 3 için poker testi sonuçları.

	İst.
1.Dizi	6.144
2.Dizi	7.844
3.Dizi	5.180
4.Dizi	8.289
5.Dizi	5.805

D. 3. 3 m = 2 için Poker Testi

Hesaplanan istatistik 7.815'den büyük ise, dizi reddedilir. Tablo D.6'dan görüldüğü gibi Lorenz sistemi için beş farklı başlangıç değeri ile elde edilen hiçbir dizi m = 2 için poker testinden kalmamıştır.

Tablo D. 6. m = 2 için poker testi sonuçları.

	İst.
1.Dizi	4.121
2.Dizi	2.769
3.Dizi	7.450
4.Dizi	4.813
5.Dizi	1.313

D. 4 Hareket Testi

Hesaplanan istatistik 42.56'dan büyükse dizi reddedilir. Tablo D.7'de görüdüğü gibi bu beş farklı başlangıç değeri ile elde edilen hiç bir dizi hareket testinden kalmamıştır.

Tablo D. 7. Hareket testi sonuçları.

	İst.	Hareket
1.Dizi	32.514	524531
2.Dizi	12.137	524556
3.Dizi	19.380	524569
4.Dizi	24.066	524951
5.Dizi	14.135	523972

D. 5 Özilişki Testi

D. 5. 1 d = 1 için Özilişki Testi

d = 1 için hesaplanan istatistik 525291.019'dan büyük ise veya 523283.98'den küçük ise, dizi reddedilir

Tablo D. 8. d = 1 için özilişki testi sonuçları.

	İst.
1.Dizi	524539
2.Dizi	524570
3.Dizi	524575
4.Dizi	524957
5.Dizi	523976

D. 5. 2 d = 2 için Özilişki Testi

d = 2 için hesaplanan istatistik 525290.519'dan büyük ise veya 523283.48'den küçük ise, dizi reddedilir

Tablo D. 9. d = 2 için özilişki testi sonuçları.

	İst.
1.Dizi	524288
2.Dizi	524052
3.Dizi	524950
4.Dizi	525046
5.Dizi	523561

D. 5. 3 d = 10 için Özilişki Testi

d = 2 için hesaplanan istatistik 525286.515'den büyük ise veya 523279.48'den küçük ise, dizi reddedilir

Tablo D. 10. d = 10 için özilişki testi sonuçları.

	İst.
1.Dizi	524191
2.Dizi	523420
3.Dizi	524323
4.Dizi	524204
5.Dizi	524819

D. 5. 4 d = 1000 için Özilişki Testi

d = 1000 için hesaplanan istatistik 524791.041'den büyük ise veya 522784.95'ten küçük ise, dizi reddedilir. Tablo D.11'den görüldüğü gibi Lorenz sistemi ile elde edilen beş farklı dizi, bu testi geçmiştir.

Tablo D. 11. d =1000 için özilişki testi sonuçları.

	İst.
1.Dizi	523494
2.Dizi	523730
3.Dizi	524656
4.Dizi	523749
5.Dizi	523228

D. 5. 5 d = 10000 için Özilişki Testi

d = 10000 için hesaplanan istatistik 520286.723'ten büyük ise veya 518289.276'dan küçük ise, dizi reddedilir. Tablo D.12'den görüldüğü gibi Lorenz sistemi ile elde edilen beş farklı dizi, d = 10000 için öz-ilişki testinden geçmiştir.

Tablo D. 12. d =10000 için özilişki testi sonuçları.

	İst.
1.Dizi	518585
2.Dizi	519047
3.Dizi	519281
4.Dizi	518899
5.Dizi	519067

EK E

LİMİT ÇEVRİMİ BELİRLEYEN ALT PROGRAM

```
/** periyot **/
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<conio.h>
main(){
double sigma=10.0;
double r=28.0;
double b=2.666666666666;
double h=.01;
double x0=.12;
double y0=-.193;
double z0=.701;
double xx0=.12;
double yy0=-.193;
double zz0=.701;
double x1;
double y1;
double z1;
double xx;
double yy;
double zz;
double pulsar=0.0;
double cycle=0.0;
double a1;
clrscr();
do{
        pulsar++;
        x1=x0 +h*(-sigma * x0 + sigma*y0);
        y1= y0 + h *(-x0*z0 + r*x0 -y0);
        z1 = z0 + h*(x0*y0 - b*z0);
        x0=x1;
        y0=y1;
        z0=z1;
        xx=xx0 + h*(-sigma * xx0 + sigma*yy0);
        yy= yy0 + h*( -xx0*zz0 + r*xx0 -yy0);
```

```
        zz = zz0 + h*(xx0*yy0 - b*zz0);
        xx0=xx;
        yy0=yy;
        zz0=zz;
        xx=xx0 + h*(-sigma * xx0 + sigma*yy0);
        yy= yy0 + h*( -xx0*zz0 + r*xx0 -yy0);
        zz = zz0 + h*(xx0*yy0 - b*zz0);
        xx0=xx;
        yy0=yy;
        zz0=zz;
        a1=fmod(pulsar,10000000.0);
        if(a1==0.0)
                printf("\nx1=%lf itsy=%lf",x1,pulsar);
        }while(x1!=xx);
printf("\nx1=xx oldugu andaki x1 degeri=%lf " ,x1);
printf("\nx1=xx olana kadarki iter.sayisi=%e  ",pulsar);
x0=xx;
y0=yy;
z0=zz;
do{
        cycle++;
        x1=x0 +h*(-sigma * x0 + sigma*y0);
        y1= y0 + h *(-x0*z0 + r*x0 -y0);
        z1 = z0 + h*(x0*y0 - b*z0);
        x0=x1;
        y0=y1;
        z0=z1;
        }while(xx!=x0);
printf("\nPeriyot=%lf ",cycle);
return 0;
}
```

ÖZGEÇMİŞ

1972 yılında Erzurum'da doğdu. İlkokulu Saadetdere İlkokulunda pekiyi derece ile 1983 yılında tamamladı. Ortaokul ve Lise öğrenimini Avcılar 50. Yıl İnsa Lisesinde bitirdikten sonra 1989 yılında İstanbul Üniversitesi Elektronik Mühendisliği bölümüne girdi. Lise son sınıfta ve dört senelik lisans öğrenimi boyunca Vaksa Hacı Ömer Sabancı başarı bursu aldı. Lisans öğrenimini, 1993 yılında 86.42/100 not ortalaması ile dönem ikincisi olarak tamamladı. Aynı yıl İstanbul Teknik Üniversitesi Elektronik ve Haberleşme yüksek lisans programına yerleşti. Bir senelik İngilizce hazırlık kursunu tamamladıktan sonra 1994 yılında yüksek lisans derslerini almaya başladı.