T.C.
ERCİYES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

FARKSAL GELİŞİM ALGORİTMASININ İNCELENMESİ VE İŞARET KESTİRİMİNDE KULLANILMASI

Tezi Hazırlayan
Canan Aslıhan KOYUNCU

Tezi Yöneten
Yrd. Doç. Dr. Nurhan KARABOĞA

Elektronik Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

Eylül 2006
KAYSERİ

T.C.
ERCİYES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

FARKSAL GELİŞİM ALGORİTMASININ İNCELENMESİ VE İŞARET KESTİRİMİNDE KULLANILMASI

Tezi Hazırlayan
Canan Aslıhan KOYUNCU

Tezi Yöneten
Yrd. Doç. Dr. Nurhan KARABOĞA

Elektronik Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

Bu çalışma Erciyes Üniversitesi Bilimsel Araştırma Projeleri Birimi tarafından FBT-06-19 nolu proje olarak desteklenmiştir.

Eylül 2006
KAYSERİ

Yrd. Doç. Dr. Nurhan KARABOĞA danışmanlığında **Canan Aslıhan KOYUNCU** tarafından hazırlanan **"Farksal Gelişim Algoritmasının İncelenmesi ve İşaret Kestiriminde Kullanılması"** adlı bu çalışma, jürimiz tarafından Erciyes Üniversitesi Fen Bilimleri Enstitüsü Elektronik Mühendisliği Anabilim Dalında **Yüksek Lisans** tezi olarak kabul edilmiştir.

18/09/2006

JÜRİ:

Başkan : Prof. Dr. Mustafa Alçı

Üye : Prof. Dr. Kerim Güney

Üye : Yrd. Doç. Dr. Nurhan Karaboğa

ONAY:

Bu tezin kabulü, Enstitü Yönetim Kurulunun 22/09/2006 tarih ve 2006/28-08 sayılı kararı ile onaylanmıştır.

22/09/2006

Prof. Dr. Nusret AYYILDIZ
Enstitü Müdürü
Mühür ve İmza

TEŞEKKÜR

Tez çalışmasının danışmanlığını yürüten ve yüksek lisans çalışmalarım boyunca bu çalışmanın gerçekleştirilmesinde bilgi ve tecrübelerini esirgemeyen saygı değer hocam Yrd. Doç. Dr. Nurhan Karaboğa'ya, sevgili arkadaşım Araştırma Görevlisi Bahriye Baştürk'e ve öğrenim hayatım boyunca maddi ve manevi desteğini esirgemeyen her zaman yanımda olan sevgili aileme teşekkürlerimi sunarım.

FARKSAL GELİŞİM ALGORİTMASININ İNCELENMESİ VE İŞARET KESTİRİMİNDE KULLANILMASI

Canan Aslıhan KOYUNCU
Erciyes Üniversitesi, Fen Bilimleri Enstitüsü
Yüksek Lisans Tezi, Eylül 2006
Tez Danışmanı: Yrd. Doç. Dr. Nurhan KARABOĞA

ÖZET

Optimizasyon problemleriyle karşılaşıldığında genellikle ilk yapılması gereken, probleme özel sezgisel bir yaklaşım tekniği belirlemektir. Bu şekildeki gelişime dayalı algoritmalar, zor optimizasyon problemleri için oldukça üstün başarım göstermektedirler. Algoritmada kullanılan basit mutasyon işlemi, algoritmanın başarımını geliştirmekte ve onu daha güçlü (robust) yapmaktadır. Farksal Gelişim Algoritması hızlı, basit, kolayca kullanılabilir ve değiştirilebilir, gürültülü ve zamana bağlı amaç fonksiyonları için kullanılabilir, özellikle doğrusal olmayan sınırlamalı optimizasyon problemlerinde etkili olma vb. özelliklere de sahiptir.

Sayısal süzgeç ayrık zamanlı işaretleri süzmek için kullanılan sayısal bir sistemdir. Süzgeç tasarımcıları, farklı bir çok süzgeç yapısını, örneğin, Sonlu Darbe Yanıtlı (Finite Impulse Response, FIR) ve Sonsuz Darbeli Yanıtlı (Infinite Impulse Response, IIR) süzgeçler, bunların doğrudan ya da kaskat şekilleri veya kafes yapılarını vs. incelemektedirler. Çoğu süzgeç tasarım işleminde tasarımcı, önce bir süzgeç yapısını ardından süzgeç katsayılarını ve sonra da seçilen süzgecin ihtiyaçları yerine getirip getirmediğini belirlemektedir. Giriş verisinin istatistiksel karakteristiği zamana bağlı olarak değişirse ya da giriş verisi hakkındaki gereken bilgi yeterli olmazsa adaptif süzgeçlere ihtiyaç duyulmaktadır. Bilgisayarlar, başarımlarının gelişimi ile evrimsel algoritmalara dayanan optimizasyon metotlarının yeni bir sınıfını kullanarak bazı teknik problemlerin çözümüne izin vermektedir. Bu çalışmada, bu algoritmalardan biri olan Farksal Gelişim algoritması kullanılarak adaptif FIR süzgeç tasarımı gerçekleştirilmiş ve işaret kestirimine yönelik yeni bir algoritma geliştirilmiştir.

Anahtar Kelimeler: Farksal Gelişim Algoritması, Adaptif Süzgeç, İşaret Kestirimi.

ANALYSING DIFFERENTIAL EVOLUTION ALGORITHM AND ITS USE FOR SIGNAL PREDICTION

Canan Aslıhan KOYUNCU
Erciyes University, Graduate School of Natural and Applied Sciences
MSc. Thesis, September 2006
Thesis Supervisor: Assistant Prof. Dr. Nurhan KARABOĞA

ABSTRACT

Facing with optimization problems, the first thing to do is determining a specific approximation technique for the problem. The evolutionary algorithms have high performance on the difficult optimization problems. The simple mutation process used in the algorithm makes differential evolution be an efficient, effective and robust evolutionary optimization method. Differential Evolution algorithm is also simple and easy to use and to modify and requires a few control variables. It is quite effective particularly on nonlinear and noisy problems.

Digital filter is a system that filters discrete time signals. Filter designers consider different filter structures: Finite Impulse Response(FIR) and Infinite Impulse Response (IIR) filters, direct and cascaded forms, lattice structures etc. Common to most filter design procedures is the fact that the designer has to specify a filter structure beforehand, compute the filter coefficients and then decide if the chosen filter fulfills the requirements. If the statistical characteristic of the input data varies with respect to time or the required knowledge about input data is not satisfactory, adaptive filters are needed. For the design of adaptive FIR filters, there exist several methods. Computer performance improvement allows solving some technical problems by using a new class of optimization methods based on the so-called ‘evolution’ algorithm.

In this study, one of these algorithms, which is called differential evolution has been used to design Adaptif FIR Filter and a new algorithm has been described for Signal Prediction.

Keywords: Differential Evolution Algorithm, Adaptive Filter, Signal Prediction.

İÇİNDEKİLER

KABUL ONAY i
TEŞEKKÜR ii
ÖZET iii
ABSTRACT iv
KISALTMALAR viii
TABLOLAR LİSTESİ ix
ŞEKİLLER LİSTESİ x

1. BÖLÜM 1
1. Giriş 1
1.1. Temel bilgiler 1
1.2. Literatür Araştırması 5

2. BÖLÜM 7
ADAPTİF SÜZGEÇLER 7
2.1. Giriş 8
2.2. Adaptif Süzgeçleme 9
2.3. Adaptif FIR Süzgeç ve Uygulamaları 10
2.4. Adaptif Süzgeçleme Algoritmaları 12

3. BÖLÜM 14
FARKSAL GELİŞİM ALGORİTMASI 14
3.1. Giriş 14
3.2. Farksal Gelişim Algoritması 15
3.3. Algoritmanın Temel Adımları 17
3.3.1. Kodlama 20
3.3.2. Popülasyon yapısı ve parametre sınırları 21
3.3.3. Mutasyon 22
3.3.4. Seleksiyon 22
3.3.5. Rekombinasyon 23
3.4. DE Metodu 23

3.4.1. DE/rand/1/bin 24
3.4.2. DE/best/2/bin 25

4. BÖLÜM 27
DE ALGORİTMASI KULLANILARAK ALT TASARIMI 27
4.1. DE Algoritması Kullanılarak ALT Tasarımı ve LMS Algoritmasıyla Başarımlarının Karşılaştırılması 27
4.1.1. Tasarım Hakkında Genel Bilgi 28
4.1.2. Simülasyon Sonuçları 29
4.1.2.1. Gürültüsüz Referans Giriş için ALT Tasarımı 29
4.1.2.1.1. LMS Algoritmasıyla ALT Tasarımı 30
4.1.2.1.2. DE/rand/1/bin ile ALT Tasarımı 32
4.1.2.1.3. DE/best/2/bin ile ALT Tasarımı 33
4.1.2.2. Gürültülü Referans Giriş için ALT Tasarımı 35
4.1.2.2.1. Gürültülü Referans Giriş için LMS Algoritması ile ALT Tasarımı 36
4.1.2.2.2. Gürültülü Referans Giriş için DE/rand/1/bin ile ALT Tasarımı 38
4.1.2.2.3. Gürültülü Referans Giriş için DE/best/2/bin ile ALT Tasarımı 40
4.2. Farklı Hata Fonksiyonları Kullanılarak DE Algoritması ile ALT Tasarımı 42
4.2.1. ALT Tasarımında Farklı Hata Fonksiyonları Kullanıldığı Durumda Elde Edilen Simülasyon Sonuçları ve Değerlendirme 43
4.2.2. DE/rand/1/bin için MSE, LMS ve MAE Hata Fonksiyonlarının Değerlendirilmesi 44
4.3. DE/rand/1/bin, DE/best/2/bin ve LMS Algoritmalarıyla Tasarlanan ALT'lerin İşaret Bant Genişliğine göre Gösterdikleri Davranışların Karşılaştırılması 47
4.3.1. Giriş İşareti Frekansının 25 kHz olduğu ALT Tasarımı 47
4.3.2. Giriş İşareti Frekansları 25 kHz ve 10 kHz olan iki işaretin toplamı olan Adaptif Süzgeç Tasarımı 52
4.3.3. Giriş işareti frekansları 25 kHz, 10 kHz ve 5 kHz olan üç işaretin toplamı olan ALT Tasarımı 57

5. BÖLÜM 63
İŞARET KESTİRİMİNE YÖNELİK ADAPTİF FIR SÜZGEÇ TASARIMI 63

5.1. Yöntem 63
5.2. Simülasyon Sonuçları 64

6. BÖLÜM 82
SONUÇ 82

KAYNAKLAR 84
EK-1 93
EK-2 98
ÖZGEÇMİŞ 100

KISALTMALAR

DEA	Farksal Gelişim Algoritması
LMSA	En Küçük Ortalama Kareler Algoritması
RLSA	Özyineli En Küçük Kareler Algoritması
FIR	Sonlu Darbe Yanıtlı Sayısal Süzgeçler
IIR	Sonsuz Darbe Yanıtlı Sayısal Süzgeçler
MSE	Ortalama Karesel Hata
LMS	En Küçük Ortalama Karesel Hata
MAE	Ortalama Mutlak Hata
A/D	Analog/Sayısal
SNR	İşaret Gürültü Oranı
CR	Çaprazlama sabiti
F	Ölçekleme sabiti
K	Kombinasyon faktörü
rand	Rasgele değişken
μ	LMS algoritmasının adım büyüklüğü

TABLOLAR LİSTESİ

Tablo 4.1. LMS Algoritmasının kontrol parametreleri…….……...…….31

Tablo 4.2. DE'nin kontrol parametreleri……...…..……...…...32

Tablo 4.3. DE/best/2/bin'nin kontrol parametreleri…….……...…..34

Tablo 4.4. LMS Algoritması, DE/rand/1/bin ve DE/best/2/bin ile ALT tasarımında elde edilen ağırlıklar…….....…..35

Tablo 4.5. LMS Algoritmasının kontrol parametreleri……..……..36

Tablo 4.6. DE'nin kontrol parametreleri……..……..……40

Tablo 4.7. Gürültülü referans giriş için LMS Algoritması, DE/rand/1/bin ve DE/best/2/bin ile elde edilen ağırlıklar…...……..…....…...42

Tablo 4.8. DE/rand/1/bin'nin kontrol parametreleri…..…….……...…..44

Tablo 4.9. MSE, LMS ve MAE ile elde edilen ağırlıklar…….....……..47

Tablo 4.10. DE/rand/1/bin'nin kontrol parametreleri……..…...48

Tablo 4.11. DE/best/2/bin'nin kontrol parametreleri…….……48

Tablo 4.12. LMS Algoritmasının kontrol parametreleri.....…….....…..…48

Tablo 4.13. Frekansı 25 kHz olan tek bir işaretin giriş işareti olduğu süzgeç için DE1/rand/1/bin, DE/best/2/bin ve LMS algoritmaları ile elde edilen ağırlıklar……...…….……..52

Tablo 4.14. DE1/rand/1/bin, DE/best/2/bin ve LMS algoritmaları ile elde edilen ağırlıklar…….……..…….…….…….57

Tablo 4.15. Üç sinüzoidal işaret toplamının giriş işareti olduğu durumda DE1/rand/1/bin, DE/best/2/bin ve LMS algoritmaları ile elde edilen ağırlıklar…….…..……..……62

Tablo 5.1. DE algoritmasında kullanılan kontrol parametreleri.....…...…..64

Tablo 5.2. İşaret kestirimine yönelik adaptif FIR süzgeç tasarımında DE ile elde edilen ağırlıklar…….……..……..……....66

Tablo 5.3. İşaret kestirimine yönelik Adaptif FIR Süzgeç tasarımında DE ile elde edilen ağırlıklar……..……..…....…….73

Tablo 5.4. Giriş işaretinin iki sinüzoidal işaret olduğu LMS-DE ile işaret kestiriminde elde edilen FIR süzgeç ağırlıkları …..…....……80

ŞEKİLLER LİSTESİ

Şekil 2.1. Geniş bantlı S(f) işareti içinde dar bantlı güçlü bozucu bir işaret N(f) 9
Şekil 2.2. Adaptif süzgeç için blok diyagram 10
Şekil 2.3. Adaptif sistem kimliklendirme 11
Şekil 2.4. Adaptif sistem modelleme 11
Şekil 2.5. Adaptif kestirim 12
Şekil 2.6. Gürültü giderme 12
Şekil 3.1. Yeni bir çözümün elde edilmesi 19
Şekil 3.2. Algoritmanın şematik diyagram formunda gösterimi 20
Şekil 3.3. DE/rand/1/bin'e göre $u_{j,i,G+1}$'i elde etmek için oluşan süreç ve kontür çizgileriyle amaç fonksiyonunun gösterimi. 24
Şekil 3.4. D=7, n=3 ve L=3 için çaprazlama örneği 25
Şekil 3.5. DE/best/2/bin'e göre $u_{j,i,G+1}$'i elde etmek için oluşan süreç ve kontür çizgileriyle amaç fonksiyonunun gösterimi. 26
Şekil 4.1. İki katsayılı adaptif lineer toplayıcı gösterimi 29
Şekil 4.2. Süzgeç girişi, x(n) 30
Şekil 4.3. Gürültüsüz referans işaret d(n) 30
Şekil 4.4. LMS algoritması sonucu y(n) ve gürültüsüz referans d(n), (a)İlk 100, (b) 450-550 arası, (c)Son 100 iterasyon 31
Şekil 4.5. LMS Algoritması için iterasyona göre hata fonksiyonunun gelişimi 31
Şekil 4.6. DE/rand/1/bin sonucu süzgeç çıkışı y(n) ve istenen gürültüsüz çıkış d(n). (a) İlk 100, (b)450-550 arası, (c) Son 100 iterasyon 33
Şekil 4.7. Gürültüsüz referans giriş için DE/rand/1/bin için iterasyonda hata 33
Şekil 4.8. DE/best/2/bin sonucu süzgeç çıkışı y(n) ve gürültüsüz istenen çıkış d(n). (a) İlk 100 (b) 450-550 arası, (c) Son 100 iterasyon 34
Şekil 4.9. Gürültüsüz referans için DE/best/2/bin'de iterasyona göre hata fonksiyonunun gelişimi 34
Şekil 4.10. Arzu edilen işaretin, gürültü işaretiyle sinüzoidal işaretin toplamından oluştuğu durumda iki katsayılı adaptif lineer toplayıcı gösterimi 35
Şekil 4.11. LMS algoritması ile yakalanacak gürültülü istenilen işaret, d(n) 37
Şekil 4.12. LMS algoritması sonucu süzgeç çıkışı y(n) ve istenen gürültülü çıkış d(n).

(a) İlk 100, (b)450-550 arası, (c) Son 100 iterasyon 37
Şekil 4.13. Gürültülü referans için LMS Algoritmasında iterasyona göre hata 38
Şekil 4.14. DE/rand/1/bin ile yakalanacak gürültülü istenilen işaret, d(n) 39
Şekil 4.15. DE/rand/1/bin sonucu süzgeç çıkışı y(n) ve istenen gürültülü çıkış d(n). (a) İlk 100, (b)450-550 arası, (c) Son 100 iterasyon 39
Şekil 4.16. Gürültülü referans için DE/rand/1/bin ile iterasyona göre hatanın gelişimi 39
Şekil 4.17. DE/best/2/bin ile yakalanacak gürültülü istenilen işaret, d(n) 40
Şekil 4.18. DE/best/2/bin sonucu süzgeç çıkışı y(n) ve istenen gürültülü çıkış d(n). (a) İlk 100, (b)450-550 arası, (c)son 100 iterasyon 41
Şekil 4.19. Gürültülü referans için DE/best/2/bin'de iterasyona göre hata 41
Şekil 4.20. Hata fonksiyonlarının başarımlarını değerlendirmede kullanılan Adaptif Lineer Toplayıcı girişi 44
Şekil 4.21. Hata fonksiyonlarının başarımlarını değerlendirmede kullanılan Adaptif Lineer Toplayıcı referans girişi 44
Şekil 4.22. Hata fonksiyonları için arzu edilen işaretle süzgeç çıkışının karşılaştırılması. (a) MSE (b)LMS, (c) MAE 45
Şekil 4.23. İterasyona göre hata değerleri. (a) MSE, (b)LMS, (c) MAE 46
Şekil 4.24. Tek bir sinüzoidal işaretin giriş işareti olduğu durumda adaptif lineer toplayıcı giriş işareti 49
Şekil 4.25. Tek bir sinüzoidal işaretin giriş işareti olduğu durumda adaptif lineer toplayıcı referans girişindeki arzu edilen işaret 49
Şekil 4.26. Tek bir sinüzoidal işaretin giriş işareti olduğu durumda adaptif süzgeç çıkışı y(n), arzu edilen çıkış d(n)'in ilk 200 örneği. (a)DE/rand/1/bin (b) DE/best/2/bin (c) LMS algoritması 49
Şekil 4.27. Tek bir sinüzoidal işaretin giriş işareti olduğu durumda hatanın gelişimi. (a)DE/rand/1/bin (b)DE/best/2/bin (c)LMS 51
Şekil 4.28. İki sinüzoidal işaret toplamının giriş işareti olduğu durumda adaptif lineer toplayıcı giriş işareti 53
Şekil 4.29. İki sinüzoidal işaret toplamının giriş işareti olduğu durumda adaptif lineer toplayıcı referans giriş işareti 53
Şekil 4.30. İki sinüzoidal işaret toplamının giriş işareti olduğu durumda adaptif süzgeç çıkışı y(n), arzu edilen işaret d(n)'in ilk 200 örneği. (a)DE/rand/1/bin

(b)DE/best/2/bin (c) LMS 53
Şekil 4.31. İki sinüzoidal işaretin toplamının giriş işareti olduğu durumda adaptif süzgeç çıkışı y(n), arzu edilen işaret d(n)'in son 200 örneği. (a)DE/rand/1/bin (b)DE/best/2/bin (c) LMS 55
Şekil 4.32. İki sinüzoidal işaretin toplamını için hatanın gelişimi. (a)DE/rand/1/bin (b) DE/best/2/bin (c)LMS 56
Şekil 4.33. Üç sinüzoidal işaretin giriş işareti olduğu durumdaki adaptif süzgeç giriş işareti 58
Şekil 4.34. Üç sinüzoidal işaretin giriş işareti olduğu durumdaki adaptif süzgecin referans giriş işareti 58
Şekil 4.35. Üç sinüzoidal işaretin toplamının giriş işareti olduğu durumda süzgeç çıkışı y(n), istenilen işaret d(n)'in ilk 200 örneği (a) DE/rand/1/bin (b) DE/best/2/bin (c) LMS 58
Şekil 4.36. Üç sinüzoidal işaretin toplamının giriş işareti olduğu durumda süzgeç çıkışı y(n), istenilen işaret d(n)'in son 200 örneği (a) DE/rand/1/bin (b) DE/best/2/bin (c) LMS 59
Şekil 4.37. Üç sinüzoidal işaretin toplamı için hatanın gelişimi. (a) DE/rand/1/bin (b) DE/best/2/bin (c) LMS 61
Şekil 5.1. İşaret kestirimi için genel bir gösterim 63
Şekil 5.2. İşaret kestirimi için DE ile tasarlanan adaptif FIR süzgeç çıkışındaki istenilen s(n) işareti 64
Şekil 5.3. İşaret kestirimi için DE ile tasarlanan adaptif FIR süzgeç giriş işareti 64
Şekil 5.4. İşaret kestirimi için DE ile tasarlanan (a) çıkış işareti y(n), (b) referans işaret d(n) 65
Şekil 5.5. DE ile işaret kestirimi için iterasyona göre hatanın gelişimi 65
Şekil 5.6. SNR=13 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç giriş işareti, x(n) 68
Şekil 5.7. SNR=13 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin çıkış işareti y(n), referans işaret d(n). 68
Şekil 5.8. SNR=13 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç için hata 69
Şekil 5.9. SNR=10 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı

LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin x(n) giriş işareti. 70
Şekil 5.10. SNR=10 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin (a) y(n) çıkış işareti, (b) d(n) referans giriş işareti. 70
Şekil 5.11. SNR=10 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç için hata. 71
Şekil 5.12. SNR=8 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin x(n) giriş işareti. 71
Şekil 5.13. SNR=8 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin (a) y(n) çıkış işareti, (b) d(n) referans giriş işareti. 72
Şekil 5.14. SNR=8 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç için hatanın gelişimi 72
Şekil 5.15. İki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç çıkışında istenilen s(n) işareti. 75
Şekil 5.16. SNR=13 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç giriş işareti, x(n) 75
Şekil 5.17. SNR=13 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin, (a) y(n) çıkış işareti, (b) d(n) referans giriş işareti. 75
Şekil 5.18. SNR=13 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç için hatanın gelişimi. 76
Şekil 5.19. SNR=10 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin x(n) giriş işareti. 76
Şekil 5.20. SNR=10 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin,

(a) y(n) çıkış işareti, (b) d(n) referans giriş işareti. 77
Şekil 5.21. SNR=10 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç için hatanın gelişimi. 77
Şekil 5.22. SNR=8 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin x(n) giriş işareti. 78
Şekil 5.23. SNR=8 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin, (a) y(n) çıkış işareti, (b) d(n) referans giriş işareti. 79
Şekil 5.24. SNR=10 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç için hatanın gelişimi. 79

1. BÖLÜM

GİRİŞ

Bu tez çalışmasında Farksal Gelişim (Differential Evolution, DE) Algoritması kullanılarak adaptif FIR süzgeç tasarımı gerçekleştirilmiş ve işaret kestirimine yönelik yeni bir algoritma geliştirilmiştir. DE Algoritması ilk olarak Adaptif Lineer Toplayıcı(ALT) tasarımında kullanılmıştır. DE algoritması, ALT tasarımında gürültülü ve gürültüsüz işaretler için ayrı ayrı yazılmıştır. DE Algoritması ile süzgeç tasarımında çeşitli hata fonksiyonları kullanılarak algoritmanın işlevselliği artırılmıştır. Farklı işaret/gürültü oranına sahip periyodik işaretler üzerinde çalışılmış ve DE Algoritmasının süzgeç tasarımındaki başarısı ölçülmüştür. Ayrıca ALT için farklı bant genişliğine sahip işaretler kullanılarak da yazılan algoritmanın başarımı ölçülmüştür. İşaret kestiriminin DE ile tasarımı incelenmiş ve işaret kestirimine yönelik yeni bir algoritma önerilmiştir. İşaret kestiriminde kullanılan işaretler, farklı işaret gürültü oranına sahip gürültülere maruz kalmış sinüzoidal işaretler olarak alınmıştır. Yine önerilen algoritmanın işlevselliğini göstermek için farklı bant genişliğine sahip işaretlerin farklı işaret gürültü oranına sahip işaretlere maruz kalması durumunda tasarlanan ALTnın başarımı ölçülmüştür.

Bu bölümde çalışmayla ilgili temel bilgiler verilmiştir.

1.1. Temel bilgiler

Fiziksel bir durum hakkında bilgi taşıyan, bir veya birden fazla değişkene bağlı fonksiyonlar işaret olarak adlandırılırlar. İstenilen nitelikteki işaretleri üretebilen veya istenilen çıkış işaretlerini üretecek şekilde verilen giriş işaretlerini işleyebilen düzenekler sistem olarak adlandırılır [1,2,3].

Genlik modülasyonlu (Amplitude Modulation, AM) radyoda olduğu gibi, analog iletişim sistemleri, bir yerden diğer bir yere, yani noktadan noktaya iletişim ya da bir

yerden birçok yere bant sınırlı analog bir işareti ucuz bir şekilde iletebilirler. Uyumlu bir alıcı demodüle olmuş bir mesaj için mümkün olabilecek en büyük işaret/gürültü oranını sağlar. Bu alıcı için yapılan analizler göstermektedir ki rezidüal(kalıntı) bir hata analog sistem çıkışında her zaman için var olacaktır [4,5].

Analog sistemler sayısal sistemlerden çok daha ucuz olmalarına rağmen sayısal sistemler çok daha verimli, iyi başarıma sahip ve esnektir [6]. Bu özellikleri şöyle açıklayabiliriz:

- **Verimlilik:** Kaynak Kodlama Teoremi gönderilen mesajın ne kadar karmaşık olduğunun ölçümüne ve kaynak kodlama tarafından meydana getirilmiş karmaşıklığın üstesinden gelmeye yani bastırmaya izin verir. Analog iletişimde, ilgili parametreler sadece mesajın bant genişliği ve genliğidir. İşaret yapısı bu durumda çok daha etkili bir iletişim sistemini elde etmek amacıyla kullanılamaz.
- **Başarım:** Gürültülü Kanal Kodlama Teoreminden dolayı, hata-doğrulama kodlarını formüle edecek özel bir kriter mevcuttur. İstenildiği kadar hata serbestliği olduğu bir yerde işaret taşınabilir. Bilgi, gürültülü bir kanal ile gönderilebilir. Analog sistemler, kanal bozulmalarının üstesinden gelemezken sayısal sistemler kanal bozulmalarında bile hatasız iletim yapabilirler.
- **Esneklik:** Sayısal iletişim sistemleri gerçek değerli ayrık zamanlı işaretleri iletebilirler. Analog olarak iletilen herhangi bir işaret sadece A/D (Analog/Digital, Analog/Sayısal) dönüşümünde kullanılan bitlerin sayısı artırılarak sayısal olarak da iletilebilir. Görüntüler ticari televizyonlarda olduğu gibi analog sistemler vasıtasıyla gönderilebilir, fakat HDTV (High-Definition Television-Yüksek Tanımlı Televizyon)'de olduğu gibi daha iyi iletişim başarımı sayısal sistemler kullanıldığında sağlanır. Sayısal sistemlerin analog sistemlerden daha geniş bir işaret çeşitliliğini iletebilmelerine ek olarak noktadan noktaya sayısal sistemler etkin ve esnek iletimi sağlayan küresel(global) sistemlerle de organize olabilirler. Bilgisayar ağları, böyle bir sistemdir.

Sayısal bilgisayarların hızla artması, etkin algoritmaların artarak gelişmesi ve iletişim altyapısı oluşturmak için bilgisayarları birbirine bağlayabilme ile sayısal iletişim çoğu durum için en iyi seçenek haline gelmiştir. Sayısal yöntemlerin yaygınlaşmasını sağlayan bir diğer neden de, analog sistemlerle gerçekleştiremediğimiz bazı işlevlerin

sayısal yöntemlerle kolaylıkla gerçekleştirilebilmesidir. Örneğin, sayısal yöntemler ile bilgi işaretinde kayıpsız sıkıştırma sayesinde bilgi kaybına neden olmadan bilginin bellekte kapladığı alanın veya bant genişliğinin azaltılması mümkündür. Adaptif süzgeçlerde olduğu gibi sistem karakteristiğini kendi kendine güncelleyen sayısal sistemlerin analog karşılığı yoktur [6].

Sayısal sistemler güvenilir ve ucuz olmalarının yanı sıra diğer bazı avantajlara da sahiptir. Sayısal işaretler sadece belirli değerleri aldıkları için bozulmalardan etkilenmeleri analog işaretlere göre daha azdır. Üstelik hata algılama ve düzeltme kodlamaları ile bozulmalara karşı sayısal işaretlerin dayanıklılığı arttırılabilmektedir. Diğer taraftan, analog sistemlerde kullanılan direnç, kapasitör gibi devre elemanlarının toleranslarından dolayı, analog yöntemlerde her zaman ideal değerden bir sapma payı mevcuttur. Sayısal yöntemlerde ise sistem cevabındaki kesinliğin çok yüksek olduğu görülmektedir. Ayrıca, sayısal sistemlerde yazılım programları sayesinde devre donanımı hiç değiştirilmeksizin sadece programda değişiklik yapılarak sistem güncellenebilir. Bu ise sistem tasarımı için büyük bir esneklik sağlamaktadır.

Sayısal süzgeç, ayrık zamanlı işaretleri süzgeçlemek için kullanılan sayısal bir sistemdir. Sayısal süzgeçler, işaretin örneklenmiş değerleri üzerinde nümerik hesaplamaları yapabilmek için sayısal işlemcileri kullanırlar. Sayısal süzgeçler, yazılımsal ya da donanımsal olarak veya her ikisinin birleştirilmesiyle gerçekleştirilebilirler. Yazılıma dayalı sayısal süzgeçler, düşük seviyeli diller kullanılarak genel amaçlı sayısal işaret işleme yongaları(chips) vasıtasıyla ya da yüksek seviyeli diller kullanılarak iş istasyonları (workstation) ile veya kişisel bilgisayarlar vasıtasıyla tasarlanabilirler. Donanımsal sayısal süzgeçler, özel geliştirilmiş VLSI(Very Large Scale Integration-Çok Büyük Çapta Tümleşim) yongaları kullanılarak tasarlanabilirler. Donanım sayısal süzgeçler yüksek hesap gücü ile hızlı işlem yetenekleri sayesinde gerçek-zamanlı sayısal işaret işlemede yaygın olarak kullanılırlar.

Süzgeçleme, bir işaretin frekans spektrumunun değiştirilerek tekrar şekillendirildiği veya bazı arzu edilen özel operasyonlara tabii tutulduğu bir işlemdir. Süzgeçleme, genelde ya gürültü gibi işaretteki bozulmaları gidermek veya iki ya da daha fazla işaretin kanal kullanımını daha verimli hale getirmek için karıştırıldığı yerlerde; bu işaretleri tekrar ayırmak, işaretleri frekans elemanlarına bölmek, modüleli bir işaretin

demodülasyonunu almak, ayrık zamanlı işaretleri sürekli zaman işaretlerine çevirmek gayeleri için kullanılmaktadır [4].

Bir taşıt stereosunda motordan kaynaklanan harmoniklerin bastırılması gibi süzgeçleme gerektiren hemen hemen bütün uygulamalarda, gerekli frekans cevabı önceden bilinememektedir ya da zamanla değişen bir işaret olmaktadır. Çoğu uygulamada, kendi kendini otomatik olarak tasarlayabilen ve zamana göre sistem değişikliklerini izleyebilen adaptif bir süzgeç oldukça yararlı olmaktadır. Adaptif süzgeçler birçok uygulamada özellikle de haberleşme alanında kullanılmaktadır.

Süzgeç tasarımcıları, FIR ve IIR süzgeçler gibi farklı birçok süzgeç yapısını, bunların doğrudan ya da kaskat şekillerini veya kafes yapılarını vs. incelemektedirler. Çoğu süzgeç tasarım işleminde tasarımcı, önce bir süzgeç yapısını ardından süzgeç katsayılarını ve sonra da seçilen süzgecin ihtiyaçları yerine getirip getirmediğini belirlemektedir [7]. Bu işlem eğer dikkatlice gerçekleştirilirse, tatmin edici sonuçlar alınabilir. Birçok tasarım algoritması daha az karmaşık yapılar üzerinde yoğunlaştığı için seçilen süzgeç yapısı yine de optimallığı sağlayamamaktadır [8]. Giriş verisinin istatistiksel karakteristiği zamana bağlı olarak değişirse ya da giriş verisi hakkındaki gereken bilgi yeterli olmazsa adaptif süzgeçlere ihtiyaç duyulmaktadır [9]. Adaptif süzgeçlerin tasarlanması için birçok metot vardır. Bilgisayarlar, başarımlarının gelişimi ile evrimsel algoritmalara dayanan optimizasyon metotlarının yeni bir sınıfını kullanarak bazı teknik problemlerin çözümüne izin vermektedirler. Çeşitli algoritmalar bu problemler için kullanılmış ve yeni algoritmalar da ortaya çıkmıştır. Bunlardan birisi de gelişime dayalı algoritmalardan DE algoritmasıdır.

DE algoritması ilk olarak 1995 yılında K. Price tarafından ortaya konmuştur [10]. DE algoritması basit ama güçlü popülasyon tabanlı bir algoritmadır [11]. Özellikle bütünüyle düzenlenmiş uzayda tanımlı ve gerçek değerli tasarım parametrelerini içeren fonksiyonları, küresel olarak optimize etmek amacıyla kullanılan bir direkt araştırma algoritmasıdır [12]. DE algoritmasının tek dezavantajı stokastik bir yapıya sahip olmasıdır. Bu yakınsama oranının tam olarak belirlenmesinin imkânsız olduğunu gösterir. Fakat, minimal araştırma süreci direkt amaçlı olduğunda ve minimizasyon parametreleri doğru seçildiğinde hızlıca yakınsama özelliğine sahiptir [13-14].

Bu tez çalışmasında, ilk olarak bir yapay zeka optimizasyon algoritması olan DE algoritmasının DE/rand/1/bin ve DE/best/2/bin olarak adlandırılan iki şekli kullanılarak adaptif süzgeç tasarımı gerçekleştirilmiş ve elde edilen sonuçlar değerlendirilmiştir. Daha sonra, MSE, LMS ve MAE hata fonksiyonları, ALT tasarımında LMS algoritmasına ve DE/best/2/bin algoritmasına göre istenilen işarete daha iyi yakınsayan DE/rand/1/bin algoritmasına adaptif süzgeç tasarımı için ayrı ayrı uygulanmıştır. Çalışmanın devamında, DE algoritmasının adaptif süzgeç tasarımındaki başarımını ölçmek için farklı bant genişliğine sahip üç sinüzoidal işaret üzerinde çalışılmış ve DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarının başarımları karşılaştırılmıştır. Tez çalışmasının son bölümünde, adaptif süzgeç tasarımında ilk defa kullanılan DE algoritması, LMS algoritması ile birlikte kullanılarak yeni bir algoritma geliştirilmiştir. LMS-DE olarak adlandırdığımız bu algoritma işaret kestirimine uygulanmış ve başarımı elde edilen simülasyon sonuçları vasıtasıyla değerlendirilmiştir.

1.2. Literatür Araştırması

Price ve Storn DE'nin çalışma prensibini ortaya koymuşlardır [15]. Daha sonraki çalışmalarında, DE'nin 10 farklı stratejisini önermişlerdir. Yine yaptıkları çalışmalarla DE'nin anahtar parametrelerinin seçimi için çeşitli kurallar belirlemişlerdir [16]. DE çeşitli alanlarda birçok probleme uygulanmış ve başarılı olmuştur. DE'nin çeşitli uygulamalarına örnek olarak:

- Sayısal süzgeç tasarımı [17],
- Ethanol yakıtı üretimi için fuzzy karar verici problemlerinde DE'nin kullanılması [18],
- DE ile bulanık mantık kontrolörü tasarımı [19],
- Toplu fermantasyon işleminde DE'nin kullanılması [20-21],
- DE kullanılarak multi sensör füzyonu [22],
- DE ile üç yataklı reaktördeki ısınmanın transfer parametreleri tahmini [23],
- DE kullanılarak optimal ısı değiştirici tasarımı [24-25],
- DE ile birleşik damıtma sistemi ısısının sentezi ve optimizasyonu [26],
- DE ilenonlineer fonksiyonların optimizasyonu [27],
- DE kullanarak ısıl kırıcı çalışması optimizasyonu [28],
- DE ile histeretik(kesik) sistemlerin kimliklendirmesi [29],

- Model optimizasyonu için DE algoritması [30],
- Çoklu optimal kontrol problemleri için DE'nin kullanılması [31],
- DE ile görev özelliklerine göre optimum robot tasarımı [32],
- Mühendislik Problemlerinde DE'nin kullanılması [33],
- Sabit noktalı sayısal süzgeçlemede DE'nin kullanılması [34],
- Kontrol parametrelerinin deneysel çalışmasında DE'nin kullanılması [35],
- DE ile gerçek parametreli optimizasyon [36],
- DE gelişiminin üçüncü adımı [37],
- Jenerasyon Gap Modeliyle DE'nin karşılaştırılması [38],
- DE kullanarak adaptif kontrol parametreleriyle RBF(Radial Basis Function-Radyal temelli Fonksiyon) ağlarının eğitilmesinde [39],
- DE ile RBF ağları için artımlı(incremental) eğitim metodu [40],
- Bulanık adaptif DE [41], RBF Ağlarını geliştirmede [42],
- Küresel optimizasyona pratik bir yaklaşım olarak DE [43],
- DE ile sınırlı çok amaçlı optimizasyon [44],
- DE ve PSO(Particle Swarm Optimization, Parçacık Sürüsü Optimizasyonu) için durdurma kriterine göre yürütüm süresi analizleri [45],
- DE açısından kodlu CDMA(Coded Division Multiple Access, Kod Bölmeli Çoklu Erişim) sistemlerde girişim bastırma optimizasyonu [46],
- Kurallı DE [47],
- Çok amaçlı optimizasyon için DE [48],
- Genetik algoritma DE'nin Marcov zinciri Monte Carlo versiyonu [49],
- DE ile kuramlı optimizasyon problemi eğitim metodu [50]

verilebilir.

2. BÖLÜM

ADAPTİF SÜZGEÇLER

Adaptif süzgeçleme teknikleri çok geniş alanlarda uygulanmaktadır. Bunlar arasında yankı bastırma, adaptif dengeleme, adaptif gürültü giderme ve işaret kestirimi gibi uygulamalar bulunmaktadır. Bu uygulamalar karakteristikleri önceden bilinmeyen sistemlerin ürettiği işaretleri işleyen süreçleri içermektedir. Bu şartlar altında, sabit katsayılı süzgeçlerin yerine adaptif süzgeçlerin kullanılmasıyla çok önemli bir başarım elde edilebilir. Adaptif bir süzgeç "kendini tasarlamak için" tekrarlanan bir algoritma kullanan adaptasyon algoritması ya da adaptif süzgeçleme algoritması olarak bilinen kendi kendini tasarlayan bir süzgeç çeşididir. Algoritma sisteme göre mevcut bir önceki bilgiye göre seçilen tahmini bir başlangıçla başlar ve bundan sonra ardı ardına iterasyonlarla bu tahmini yeniler ve sonunda da istatistiksel bir algıyla optimal sonuca yakınsar.

Bir adaptif süzgeç algoritmasının başarımı aşağıdaki faktörlerden biri ya da bir kaçına bağlı olarak değerlendirilir [51]:

- **Yakınsama hızı:** Bu nicelik algoritmanın geçici davranışını göstermektedir. Sabit şartlar altında, optimuma "yeteri kadar yakın" yakınsamak amacıyla algoritma için gerekli iterasyonların sayısını tanımlar.
- **Hesaplama gereklilikleri:** Bu pratik açıdan önemli bir parametredir. İlgili parametreler algoritmanın bütün bir iterasyonu için gerekli operasyonların

sayısını, veriyi depolamak için gerekli hafıza miktarını ve aynı zamanda da programı içerir. Bu nicelikler adaptif süzgeci gerçekleştirmek için kullanılan bilgisayarın fiyatını etkilemektedir.

- **Nümerik sağlamlık:** Sonlu kelime uzunluğunu kullanarak çalışan sayısal bir bilgisayar üzerinde adaptif süzgeçleme algoritmalarının gerçekleştirilmesi kuantalama hatalarıyla sonuçlanmaktadır. Bu hatalar bazen adaptasyon algoritmasının nümerik kararsızlığına neden olabilmektedir. Bir adaptif süzgeçleme algoritmasının sonlu kelime uzunluklu işlemlerini kullanan sayısal uygulamalarının sabit olduğu durumlarda nümerik olarak güçlü(robust) olması beklenir.

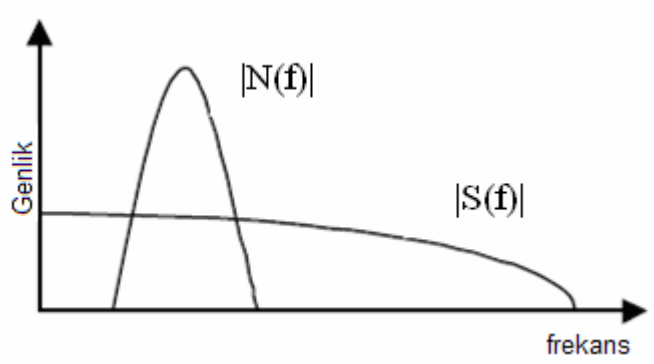
Başarımın diğer bir pratik ölçümü de adaptif süzgecin kararlı duruma ulaşabilmesi için gerekli hesaplamaların sayısıdır. Bu ölçüm yakınsama hızını ve hesaplama gereksinimlerini içerir ve optimum çözüm için yeteri kadar yakın yakınsamak amacıyla algoritmanın ihtiyaç duyduğu iterasyonların sayısının üretimidir ve her bir iterasyon için gerekli hesaplamaların sayısıdır.

İdeal olarak, yüksek yakınsama hızıyla ve düşük toleransla bilgisayar üzerinde kolayca gerçekleştirilebilen kolaylıkla hesaplanabilir ve nümerik olarak güçlü olan bir adaptif süzgeç yapısı istenir. Çoğu mühendislik problemlerinde olduğu gibi, bu arzu edilen karakteristikler çoğu durumda bir diğeriyle uyuşmamakta ve bir çeşit takas gerektirmektedir. Örneğin, genellikle kullanılan en küçük ortalama kareler (Least Mean Square, LMS) algoritması hesaplanabilir kolaylığı olan ve nümerik olarak güçlü olan bir algoritmadır fakat yakınsama hızı açısından bir dezavantajı vardır. Diğer taraftan, iyi bilinen özyineli en küçük kareler (Recursive Least Squares, RLS) algoritması hızlı yakınsama göstermektedir fakat çok karmaşık bir yapıya sahiptir. Bu algoritmalar şu anda kullanıma hazır olan adaptasyon algoritmalarının örnek olarak verilebilir.

2.1. Giriş

Pratikte, işaret aynı frekans bandını kaplayan diğer işaretlerle ya da gürültüyle bozulur. İşaret ve gürültü farklı frekans bantlarında bulunduklarında bilinen lineer(doğrusal) süzgeçler arzu edilen işareti elde edebilirler [52]. Fakat işaretle gürültü arasında spektral bir örtüşme varsa ya da işaretin veya karışan işaretin istatistikleri zamanla değişiyorsa

sabit katsayılı süzgeçler uygun değillerdir. Şekil 2.1 geniş bantlı S(f) işaretinin Fourier spektrumunun dar bantlı bozucu işaret N(f) örtüşmesini göstermektedir [53].

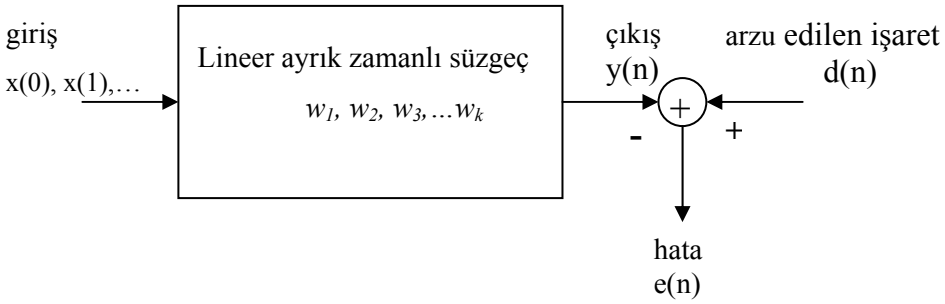


Şekil 2.1. Geniş bantlı S(f) işareti içinde dar bantlı güçlü bozucu bir işaret N(f)

Bu durum aynı frekans sahasında çalışan çeşitli modülasyon teknolojilerinin olması durumunda sık sık meydana gelebilir. Aslında, ısıl ya da diğer gürültü kaynaklarından daha çok mobil radyo sistemlerindeki ortak kanal karışması genellikle sınırlama faktörüdür [54]. Bu aynı zamanda kasıtlı işaret bastırmanın bir sonucu da olabilir. Örneğin, genellikle askeri operasyonlarda mücadele eden taraflar düşmanın haberleşmesini kesmek için kasıtlı olarak işaret yayarlar. Eğer gürültünün istatistikleri önceden bilinmiyorsa ya da zamanla değişiyorsa süzgeç katsayıları önceden belirlenemez. Bu durumlarda, sürekli olarak katsayıları yenileyen adaptif algoritmalara ihtiyaç duyulur.

2.2. Adaptif Süzgeçleme

Normal sabit bir süzgeç işaretin ve istenmeyen gürültünün her ikisinin de istatistiklerinin önceden bilinmesiyle tasarlanır. Oysa adaptif süzgeç yinelemeli algoritmalar vasıtasıyla değişen çevre parametrelerine göre sürekli ayar yapan bir süzgeçtir. Adaptif süzgeçler, zamanla değişen ve önceki değerleri bilinmeyen işaretler için tasarlanmıştır.



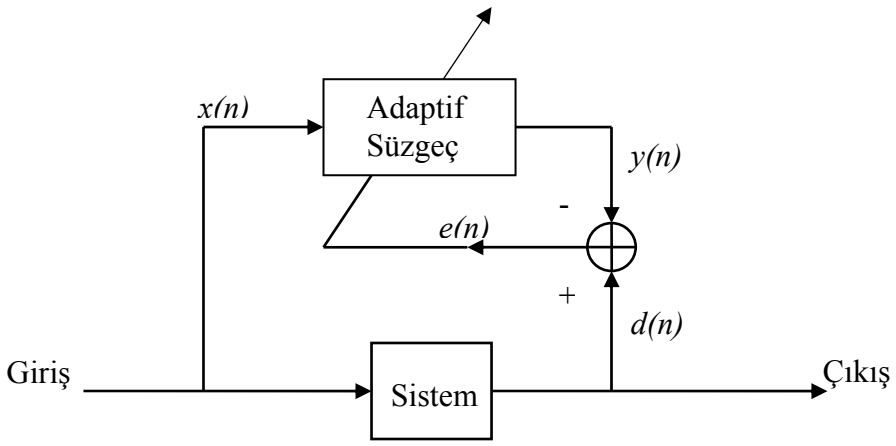
Şekil 2.2. Adaptif süzgeç için blok diyagram

Şekil 2.2'den de görüldüğü gibi adaptif süzgeçlerde işaret x(n) ve arzu edilen işaret olan referans giriş d(n) olmak üzere iki giriş gerekmektedir ve süzgeç katsayıları w_k'lar vasıtasıyla üretilen bir de y(n) çıkışı vardır. Arzu edilen işaret tahmini hata e(n)'i elde etmek için süzgeç çıkışındaki işaretle karşılaştırılır. Bu hata işareti yeni zaman sabitine göre süzgeç ağırlıklarını ayarlamakta kullanılır. Ağırlıkları ayarlamak amacıyla kullanılan birçok algoritma mevcuttur. Eğitim algoritmasının seçimi yakınsama hızına ve hesaplama karmaşıklığına bağlıdır.

2.3. Adaptif FIR Süzgeç ve Uygulamaları

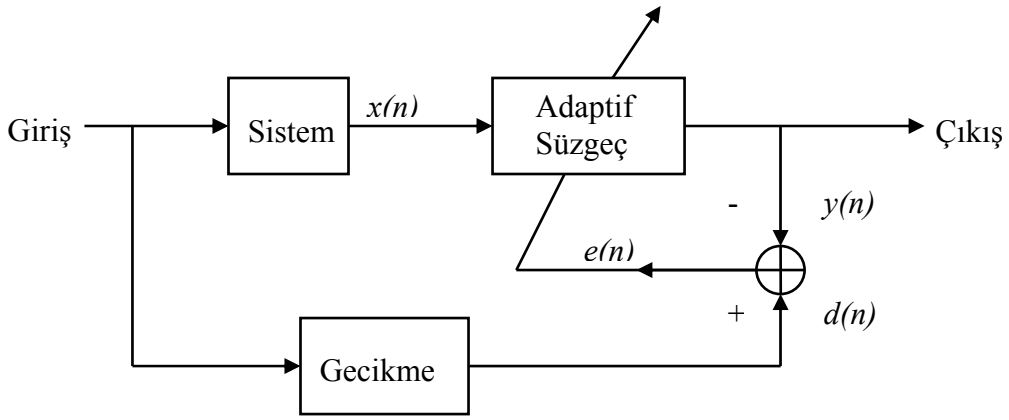
Bilinmeyen çevrelerde iyi davranış göstermesi ve istatistiksel zaman değişimlerini izleyebilmesinden dolayı adaptif süzgeçler geniş bir sahada kullanılmaktadır. Temel olarak dört önemli uygulama mevcuttur [55]. Bunlar; kimliklendirme, sistem modelleme, kestirim ve gürültü giderme uygulamalarıdır. Bunların arasındaki fark arzu edilen cevabın çıkarılması durumudur. Bunlar aşağıda açıklanarak sırasıyla Şekil 2.3, Şekil 2.4, Şekil 2.5 ve Şekil 2.6'da gösterilmektedir.

Adaptif Sistem Kimliklendirme: Adaptif süreç hata işaretinin gücünü indirgemeye çalışırken bilinmeyen sistemin darbe cevabını tahmin eden bir yöntemle süzgeç katsayıları ayarlanmaktadır.



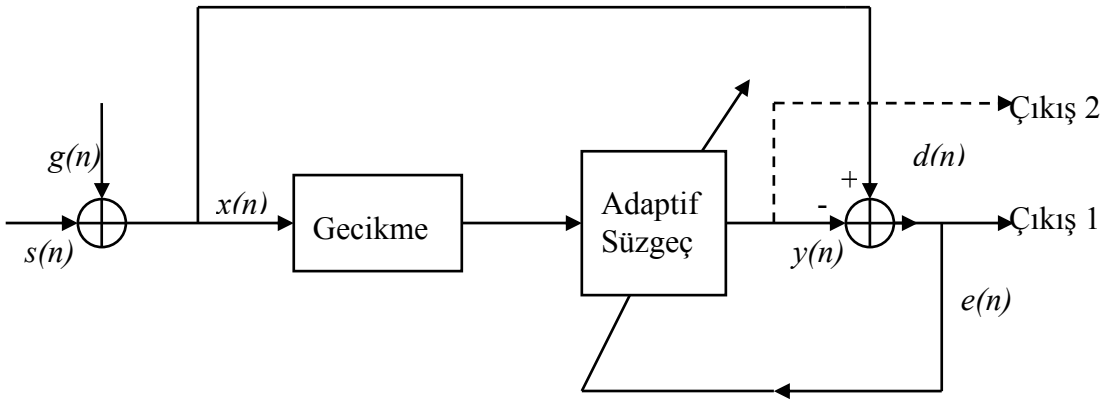
Şekil 2.3. Adaptif sistem kimliklendirme

Adaptif Sistem Modelleme: Burada Şekil 2.2'dekine benzer bir prensip kullanılmaktadır. Kimliklendirmenin tersine, ters darbe cevabı tahmin edilmektedir.



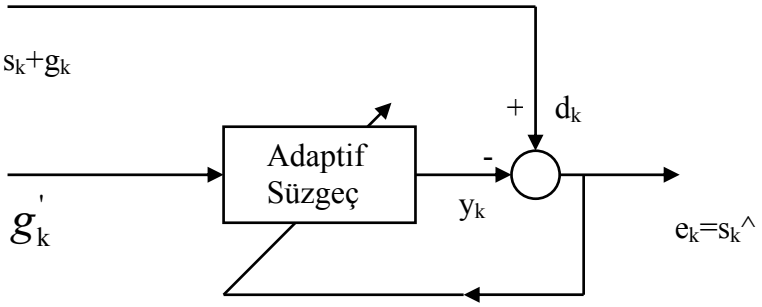
Şekil 2.4. Adaptif sistem modelleme

Adaptif Kestirim: Gürültü g(n) periyodik işaret s(n)'e eklenmiştir. Adaptif süzgecin y(n) çıkışı s(n)'nin optimal tahminidir. Hata işareti e(n), gürültü işareti g(n)'nin bir tahminidir.



Şekil 2.5. Adaptif kestirim

Gürültü giderme: Referans işaret s_k işareti ile eklenebilir g_k gürültüsünün toplamı ve giriş işareti de gürültü işareti g_k ile arasında korelasyon olan g_k' işaretidir. Adaptif süzgeç gürültü işareti g_k'ya benzeyen bir y_k çıkış işareti üretecektir. Toplam çıkış e_k işaret s_k'nın optimal bir tahminidir.



Şekil 2.6. Gürültü giderme

Süzgeç yapısının FIR ya da IIR olması, ne tür algoritma kullanılacağı ve öğrenme oranının ne olacağı eldeki uygulamaya bağlıdır. Bu çalışmada, bunlardan DE algoritması ile FIR süzgeç tasarımı üzerine odaklanılmıştır.

2.4. Adaptif Süzgeçleme Algoritmaları

Adaptif bir süzgecin ağırlıklarını yenilemek için birçok algoritma mevcuttur. LMS

algoritması en yaygın kullanılan algoritmadır. Şekil 2.2'de genel bir adaptif süzgeç gösterimi verilmişti. Burada amaç *e(n) = d(n)-y(n)* eşitliğini minimize edecek şekilde hata işaretine bağlı bir algoritma fonksiyonuyla lineer süzgeci tasarlamaktır. İyi bilinen Wiener süzgeçleme problemi için, amaç fonksiyonu $E\{e^2(n)\}$'dir [19]. Çıkış işareti *y(n)* şu şekilde hesaplanır:

$$y(n) = W_n^T X_n \tag{2.1}$$

Burada,

$$W_n = \left[w_n(0), w_n(1), ..., w_n(N-1) \right]^T \tag{2.2}$$

W_n n. iterasyonda FIR süzgecin katsayı vektörüdür ve N süzgeç uzunluğudur.

$$X_n = \left[x_n, x_{n-1}, x_{n-2}, ..., x_{n-N+1} \right]^T \tag{2.3}$$

Burada, X_n n. iterasyonda adaptif süzgecin giriş vektörüdür.

LMS algoritması W_n katsayılarını yenileyen en basit algoritmadır. Bu algoritma süzgeç katsayılarını aşağıdaki gibi yeniler:

$$W_{n+1} = W_n + \mu e(n) X_n \tag{2.4}$$

Burada, μ>0 adım büyüklüğüdür. Amaç her bir iterasyonda süzgeç katsayılarını elde etmektir [56].

3. BÖLÜM

FARKSAL GELİŞİM ALGORİTMASI

Farksal gelişim algoritması evrime dayalı bir algoritma olduğundan, öncelikle evrimsel algoritmalara kısa bir giriş yapılacak ve ardından basit DE algoritmasının avantajları vurgulanarak daha sonra algoritmanın temel adımları açıklanacaktır. Aynı zamanda algoritmanın başarımı bazı popüler genetik algoritmalarla karşılaştırılacaktır.

3.1. Giriş

Evrim prensibine dayalı teknikler bilinen optimizasyon metotlarının noksanlıklarının üstesinden gelmek için kullanılmaktadır [57-61]. Özellikle geçen 20-30 yılda evrim prensibine dayalı algoritmalara ilgi oldukça artmış ve bu konularla ilgili araştırmacıların ve literatürdeki çalışmaların sayısı hızla çoğalmıştır. Son zamanlarda bu prensibe dayalı tekniklerin hepsini temsilen ortak bir terim olarak evrimsel hesaplama terimi yaygın olarak kullanılmaya başlanmıştır. Bu sınıfa giren algoritmalara örnek olarak genetik algoritmalar, evrimsel programlama (evolutionary programming), evrimsel stratejiler (evolution strategies), ve DE algoritması vs. sayılabilir [10,15]. Ayrıca literatürde bu paradigmaların çeşitli özelliklerini birlikte kullanan çok sayıda hibrid sistemler de mevcuttur. Bundan dolayı sınıflandırma yapmak oldukça zordur. Bir problemi çözmede kullanılacak herhangi bir evrimsel algoritma aşağıdaki beş elemana ihtiyaç duymaktadır:

- Problem için çözümlerin genetik temsili,
- Çözümlerin başlangıç popülasyonunu oluşturacak bir yöntem,
- Çözümleri uygunluk açısından değerlendirmeye tabii tutacak değerlendirme fonksiyonu yani çevre,
- Genetik kompozisyonu değiştirecek operatörler,

- Kontrol parametrelerinin değerleri (popülasyon büyüklüğü, operatörleri uygulama ihtimalleri vs.).

Evrimsel algoritma tek bir bireyle (çözümle) değil, bireylerin popülasyonu P(t) ile ilgilenir. Her birey mevcut problem için muhtemel bir çözümü temsil eder ve bir veri yapısı (S) olarak tanımlanır. Her birey uygunluk (fitness) ölçüt değerini belirlemek amacıyla değerlendirilir ve daha uygun bireyleri seçmek suretiyle yeni popülasyon (P(t+1)) oluşturulur (seçme adımı). Yeni popülasyonun bazı bireyleri yeni çözümler oluşturmak için bazı operatörler vasıtasıyla transformasyona tabi tutulurlar (değiştirme adımı). Bir evrimsel algoritmanın temel adımları aşağıda verilmiştir.

Adım 1. P(t) yi başlangıç durumuna ayarla.
Adım 2. P(t) yi değerlendir.
Adım 3. Durdurma kriteri sağlanıncaya kadar aşağıdaki işlemleri gerçekleştir
  3.1. Önceki popülasyonda bulunan bireylere seçme işlemi uygulayarak yeni popülasyon oluştur.
  3.2. Yeni popülasyonu değiştir.
  3.3. Yeni popülasyonu değerlendir.

Evrimsel hesaplama teknikleri arasında oldukça benzerlikler olmasına rağmen birçok farklılıklar da mevcuttur. Örneğin, çözümlerin temsili için farklı veri yapıları kullanırlar ve dolayısıyla kullanılan genetik operatörler de farklı olabilir. Yine yukarıda verilen temel adımlarda 3.1 ve 3.2 adımları yer değiştirebilir. Örneğin, evrimsel stratejilerde ilk önce popülasyon değiştirilir ve sonra yeni popülasyon seçme işlemi ile oluşturulur. Aynı teknikte bile değişik alternatifler mevcuttur. Örneğin, DE algoritmasında bireylerin seçimi için kullanılabilecek çok sayıda seçme metodu bulunmaktadır.

3.2. Farksal Gelişim Algoritması

DE algoritması basit, ama güçlü popülasyon tabanlı bir algoritmadır [11]. Özellikle tamamen düzenlenmiş uzayda tanımlı ve gerçek değerli tasarım parametrelerini içeren fonksiyonları küresel olarak optimize etmek amacıyla kullanılan bir direkt araştırma algoritmasıdır. Gerçek parametreli optimizasyon, bilimde, mühendislikte ve iş ortamlarında karşılaşılan pratik problemlerin önemli ve geniş bir sınıfını oluşturmaktadır. Aerodinamik şekillerin optimizasyonunda, IIR süzgeç tasarımında,

Sinir ağlarının öğrenmesi gibi farklı uygulamalarda dinçliği ve gücü gösterilmiştir [17, 62-63]. Zor optimizasyon problemleriyle karşılaşıldığında genellikle ilk yapılması gereken, probleme özel sezgisel bir yaklaşım tekniği belirlemek olacaktır. Uzman bilgisinin tasarıma ilave edilmesine olanak sağlayan böyle teknikler, gelişime dayalı algoritmalar gibi genel metotlarla karşılaştırıldığında oldukça üstün başarım göstermektedirler. Örneğin, amaç fonksiyonu lineer polinom formunda ise bu tür problemi çözmek amacıyla geliştirilmiş Simpleks metodu, Karmakar algoritması gibi başarılı yöntemler mevcuttur. Bu yöntemler, bu tip problemlerde yüzlerce hatta binlerce tasarım değişkeninin bulunması durumunda bile küresel optimal çözümleri bulabilme kabiliyetine sahiptir. Amaç fonksiyonu lineer bir polinom şeklinde değilse ve probleme özel metotlar yoksa o zaman genellikle Levenberg-Marquardt algoritması, Branch and Bound metodu gibi yöntemlere başvurulmaktadır.

Maalesef, doğrusal olmama, gürültü, yüksek boyutluluk, çok modluluk, sınırlamalar, değişmeme(düz olma) ve türev alınamama gibi özellikler tek başlarına veya birlikte bu tür deterministik metotların başarımlarını çok olumsuz etkilemektedirler. Stokastik algoritmalar gibi gelişime dayalı algoritmalar, bu tür sınırlamaların çoğunun üstesinden gelebilmektedirler. Ancak bu tür genel metotlar, probleme özel bilgiyi kullanarak tasarımlarına aktaramazlarsa bunların da başarımı nispeten düşmektedir. Optimizasyon işleminin tamamen düzenlenmiş uzay domeninde sınırlandırılma bilgisi, kendi başına gelişime dayalı algoritmaların başarımını artırmak için kullanılabilecek probleme özel önemli bir bilgidir. Tipik bir gelişime dayalı algoritmanın temel basamakları aşağıdaki gibidir:

Başlangıç popülasyonunun oluşturulması
Değerlendirme
TEKRARLA
    Mutasyon
    Yeniden birleştirme
    Değerlendirme
    Seçme
DURDURMA KRİTERİ SAĞLANINCAYA KADAR

Gelişime dayalı algoritmaların genel amaçlı nümerik optimizasyon algoritmaları olarak

incelenmesi ve değerlendirilmesi ile ilgili çalışmalara literatürde oldukça sık rastlanmaktadır. Ancak bu tartışmaların çoğu, orijinalinde bu amaç için tasarlanmamış tümleşik(combinatorial) tip algoritmalarla elde edilen sonuçlar üzerinedir. DE algoritması ise bir ayrık optimizasyon algoritması değil özellikle nümerik optimizasyon için geliştirilmiş bir gelişim algoritmasıdır. Bu algoritma yeni, ama sadece basit olmayan aynı zamanda oldukça da etkili olan bir mutasyon işlemi uygulamaktadır. Daha önce tanımlanmış, olasılık dağılım fonksiyonuna dayalı olarak çalışan genetik algoritma gibi gelişim tabanlı algoritmaların tersine DE algoritması rastgele olarak seçilmiş amaç vektör çiftlerinin farklarına dayalı bir mutasyon işlemi kullanır. Vektörlerin dağılım formu temelde, amaç fonksiyonunun topografisine göstermiş oldukları cevaplara bağlıdır.

DE algoritmasında kullanılan basit mutasyon işlemi, algoritmanın başarımını geliştirmekte ve onu daha güçlü yapmaktadır. Bu özelliğinin yanı sıra diğer özellikleri için de şunlar söylenebilir: Hızlı, basit, kolayca kullanılabilir ve değiştirilebilir, etkili küresel optimizasyon kabiliyetli, doğal olarak paralel, kayan nokta formatına bağlı hassasiyet sınırlamalı, matris çarpımları ve sıralama işlemleri olmadığı için hesaplama maliyeti açısından avantajlı, daha önceden tanımlanmış herhangi bir olasılık dağılımlı mutasyon kullanmamaktadır. Tamsayı, ayrık ve karışık parametre optimizasyonuna kolaylıkla uyarlanabilir, amaç fonksiyonunun veya sınırlama fonksiyonlarının türevine gerek duymaz, düz yüzeylerde çalışabilir, gürültülü ve zamana bağlı amaç fonksiyonları için kullanılabilir, tek bir koşmada alternatif çözümler üretebilir ve özellikle doğrusal olmayan sınırlamalı optimizasyon problemlerinde etkilidir.

3.3. Algoritmanın Temel Adımları

DE algoritmasının temel adımları açık bir şekilde aşağıda verilmektedir. Çoğu açıdan bu algoritma tipik bir gelişim algoritmasıdır. Adım 2'de görüleceği üzere algoritma başlangıçta, NP (Number of Population -Popülasyon Büyüklüğü) tane D(Dimension-Boyut) boyutlu amaç değişken vektörlerinin x_{jiG} rastgele dağıtılmış başlangıç popülasyonunu (P_0) üretir. Randj [0,1] terimi, 1 ile 0 arasında uniform şekilde dağılmış rastgele değişkeni temsil etmektedir. *j* indisi *j*'nin her bir değeri için yeni rastgele sayı üretildiğini göstermektedir. *i* indisi ise rastgele değerin her bir değeri için yeni bir rastgele sayı üretildiğini göstermektedir. h_i ve l_o indisleri üst ve alt başlangıç parametre

sınırlarını temsil etmektedir. Oluşturulan bu popülasyon G_{max} jenerasyonu(Maximum Generation-Maksimum Jenerasyon, İterasyon) boyunca mutasyon, çaprazlama ve seleksiyon operatörleri uygulanarak başarılı bir şekilde geliştirilir [64,65]. DE algoritmasının en önemli özelliği deneme vektörünün üretilme şeklidir [28,66].

Adım 1. Kontrol parametrelerinin

D, G_{max}, NP≥4, F∈(0,1+), CR∈ [0,1], değerlerini ve parametre sınırlarını $\bar{x}^{(lo)}, \bar{x}^{(hi)}$ ata.

Adım 2. Başlangıç popülasyonunu oluştur

$$\forall i \le NP \wedge \forall J \le D : x_{j,i,G=0} = x_j^{(lo)} + rand_j[0,1].(x_j^{(hi)} - x_j^{(lo)})$$
$$i = (1,2,...,NP), j = (1,2,...,D), G = 0, rand_j[0,1] \in [0,1]$$

Belirli bir durdurma kriteri sağlanıncaya kadar aşağıdaki adımları tekrarla

Adım 3. Mutasyon ve rekombinasyon

$$r_1, r_2, r_3 \in \{1,2,...,NP\}, r_1 \neq r_2 \neq r_3 \neq i \text{(rasgele seçilmiş)}$$
$$j_{rand} \in \{1,2,...,D\}, \text{(rastgele seçilmiş)}$$
$$\forall j \le D, u_{j,i,G+1} = \begin{cases} x_{j,r_3,G} + F.(x_{j,r_1,G} - x_{j,r_2,G}) \\ \quad \text{eğer } (rand_j[0,1) < CR \vee j = j_{rand}) \\ x_{j,i,G} \quad \text{diğer durumlarda} \end{cases}$$

Adım 4. Seleksiyon

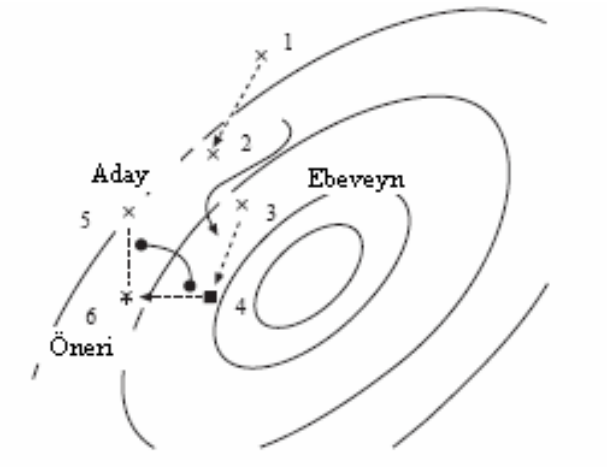
$$\bar{x}_{i,G+1} = \begin{cases} \bar{u}_{i,G+1} & \text{eğer } f(\bar{u}_{i,G+1}) \le f(\bar{x}_{i,G}) \\ \bar{x}_{i,G} & \text{diğer durumlarda} \end{cases}$$

DE algoritması, bir ebeveyn vektöründen x_{jiG}, bir deneme vektörü u_{jiG+1} üretmek için mutasyon ve rekombinasyon işlemlerinin her ikisini birlikte kullanmaktadır. r_1, r_2, r_3 indisleri, i indisinden farklı olan ve aynı zamanda birbirlerinden farklı rastgele seçilmiş popülasyon indisleridir. Burada i mevcut amaç vektörünü (ebeveyn) temsil etmektedir. Dolayısıyla Adım 1'de belirtildiği gibi popülasyon büyüklüğü NP her zaman 3'ten büyük olmalıdır. Hem CR(Crossover Rate-Çaprazlama Oranı) hem de F değerleri kullanıcı tarafından belirlenecek olan kontrol parametreleridir. CR olasılığı temsil ettiği için 0 ile 1 arasında değerler almaktadır. F(Fitness Scaling-Uygunluk Ölçekleme) ise bir ölçekleme faktörüdür. $Rand_j$ [0,1] CR'den küçükse veya $j=j_{rand}$ olduğunda yeni (child,

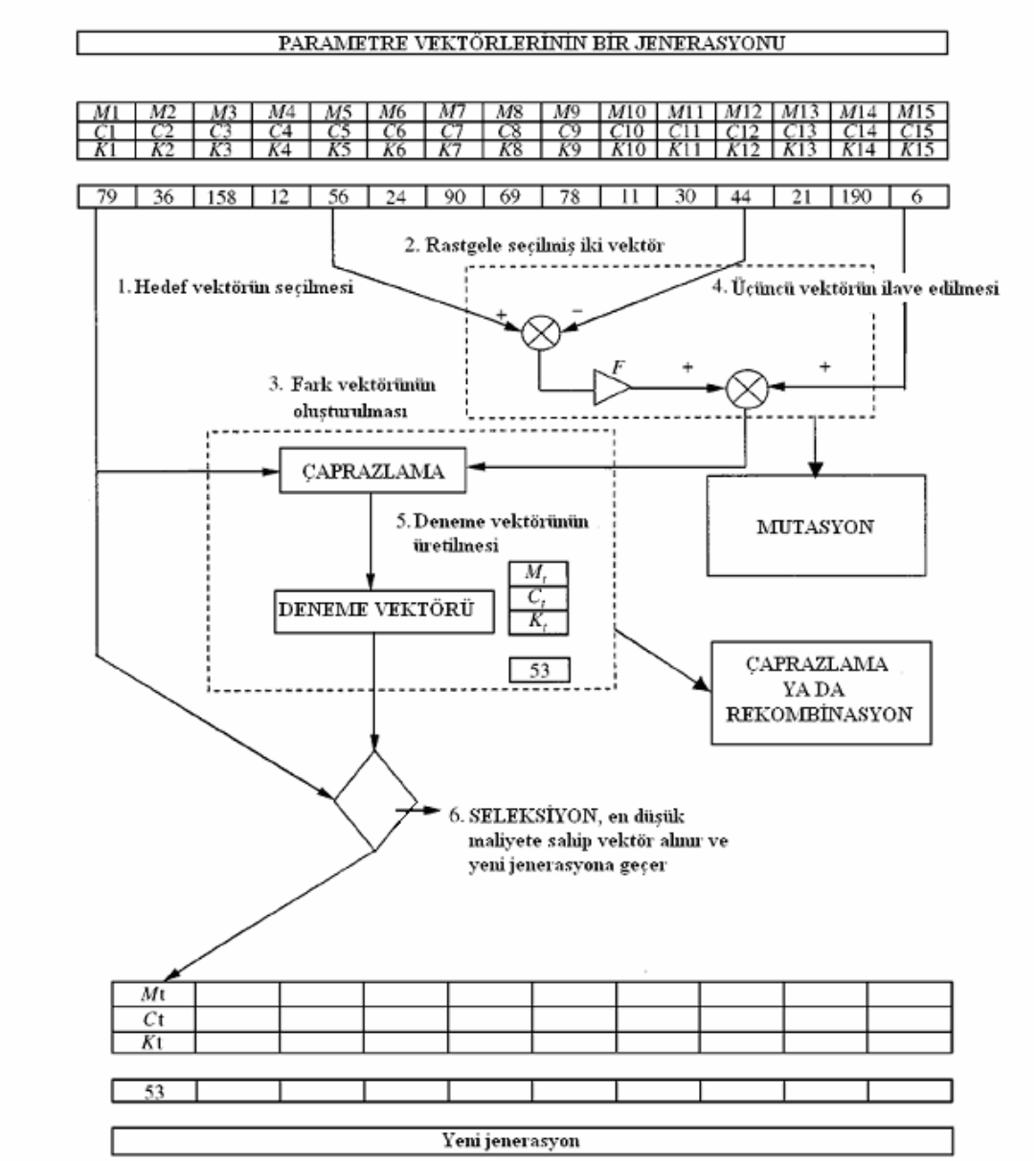
çocuk) parametre, rastgele seçilmiş üç tane vektörün doğrusal kombinasyonu olacaktır. Değilse, yeni parametre doğrudan kendi ebeveyninden alınacaktır. $\vee j=j_{rand}$ şartı, yeni vektörlerin en az bir parametrede de olsa ebeveyn vektörlerin farklı olmasını garanti etmek amacıyla kullanılmaktadır.

Adım 4'te görüldüğü gibi her yeni vektör, amaç fonksiyonu vasıtasıyla değerlendirildikten sonra maliyeti ebeveyninin maliyeti ile karşılaştırılır. Şayet yeni vektör eşit veya daha düşük bir maliyete sahip olursa yeni vektör popülasyondaki ebeveyn vektörü ile yer değiştirir. Yoksa ebeveyn vektör yeni vektörle yer değiştirmeyip popülasyonda muhafaza edilir. Bir sonraki jenerasyonun üyeleri bu seleksiyon işlemiyle seçildikten sonra gelişim çevrimi ya problem çözülene kadar ya da tüm vektörler bir noktaya yakınsayana kadar tekrar etmektedir. Algoritmanın şematik diyagramı Şekil 3.2'de verilmektedir. Diyagram algoritmanın işleyişini daha rahat anlamak için alternatif bir yöntemdir.

Yeni bir çözümün üretilmesi Şekil 3.1'de gösterilmiştir. İki popülasyon üyesi (1,2) arasındaki fark üçüncü popülasyon üyesine (3) ilave edilir. Sonuç (4), aday üyeyle (5) çaprazlama işlemine tutulmakta ve yeni çözüm (6) elde edilmektedir. Şayet yeni çözüm, adaydan daha iyiyse adayla yer değiştirmektedir.



Şekil 3.1. Yeni bir çözümün elde edilmesi



Şekil 3.2. Algoritmanın şematik diyagram formunda gösterimi

3.3.1. Kodlama

Nümerik optimizasyonu amaçlayan çoğu gelişim algoritmaları nümerik parametreleri kodlamak amacıyla ikili tamsayıları kullanmaktadır. Ancak nümerik optimizasyon işlemlerinde tamsayı formatla parametre değerlerinin geniş dinamik sahasını verimli olarak temsil etmek pek mümkün olmamaktadır. Bunun için Gray kodlama gibi değişik kodlama türleri kullanılsa bile, hala bu tür yaklaşımlar geniş dinamik sahayı tanımlama kabiliyetinden yoksun kalmaktadır. Bu yüzden amaç vektörlerini kayan-noktalı sayılar kullanarak kodlama tercih edilen bir yöntem olmaktadır. Bundan dolayı DE algoritması,

gerçek parametreleri bilinen kayan-noktalı sayılar kullanarak kodlamakta ve bu parametreler arasındaki işlemleri, standart kayan-noktalı aritmetik mantığına göre gerçekleştirmektedir.

3.3.2. Popülasyon yapısı ve parametre sınırları

DE algoritması, sabit büyüklükte popülasyon kullanmaktadır. Popülasyon, NPxD kayan-noktalı dizi formunda tanımlanır. Burada NP, popülasyondaki amaç vektörlerinin sayısına ve D de bir amaç vektöründeki parametrelerin sayısına karşılık gelmektedir. Gelişim algoritmasını paralel formda gerçekleştirirken, amaç vektörlerinin iki dizisi hafızada tutulur. Birinci dizi mevcut ebeveyn popülasyonu için diğeri ise yeni vektörleri ihtiva eden gelecek jenerasyonu oluşturmak amacıyla kullanılmaktadır. Başlangıç amaç vektörleri iyi tanımlanmış sınırlamalara sahip bir araştırma uzayından uniform dağılımla rastgele elemanlar seçilerek oluşturulur. Pratikte, başlangıç parametre sınırları genellikle fiziksel şartlara göre belirlenir. Fiziksel şartlar belirleyici değilse o zaman başlangıç alt ve üst parametre sınırları, küresel optimayı ihtiva eden bölgeleri kapsayacak kadar büyük seçilmelidir. Alt ve üst parametre sınırları başlangıç parametreleri için kullanıldıktan sonra algoritmanın başlangıçta verilen sınırlar ötesinde araştırma yapmasına imkân vermek amacıyla ihmal edilebilir. Bununla birlikte sınırlamalar, özellikle düzgün ya da düzgüne yakın ani değişim göstermeyen yüzeyler ile çalışıldığında sonsuza gidişin engellenmesi için gereklidir. Başlangıçta eğer iyi bir çözüm tespit edilebilmişse veya zaten mevcutsa, algoritma bu nominal çözüm etrafında dağılmış amaç vektörlerinden oluşan popülasyonla araştırmaya başlayabilir. Ancak bu şekildeki bir uygulama araştırmayı bölgeselleştireceğinden pek tavsiye edilmez. Hatırlanması gereken önemli bir nokta, başlangıç parametre değerlerinin bir şekilde rasgele dağıtılmasıdır. Çünkü asıl araştırmayı yönlendiren, amaç vektörleri arasındaki farklardır. Başlangıç popülasyonu oluşturulurken uygun olmayan bir amaç vektörü kabul edilmeyip reddedilebilir ve tekrar üretilebilir.

Yeni bir parametre değeri, katı bir sınırlamayı aşarsa o zaman parametreye müsaade edilen herhangi bir değeri atamak için çeşitli yöntemler mevcuttur. Bunların en basiti, parametreye değer olarak aşmış olduğu sınır değerini atamaktır.

3.3.3. Mutasyon

Parametre optimizasyonu açısından mutasyon işlemi, var olan bir amaç vektörünün bir veya daha fazla parametreleri üzerine, rasgele üretilmiş bir artışın ilave edilme işlemine karşılık gelir. İyi bir mutasyon işleminin amacı, var olan amaç vektörlerini doğru zamanda doğru miktar ile doğru yönde hareket ettirecek artışları başka bir ifadeyle adımları üretmektir. Bu hedefi başarmak esas olarak adımları üretecek dağılımın karakteristiklerine bağlıdır.

DE algoritması, bir amaç vektörüne mutasyon işlemini, rasgele seçilmiş amaç vektörler çiftinin ağırlıklaştırılmış farkının bu amaç vektörüne ilave edilmesiyle gerçekleştirir. Algoritmanın basit mutasyon işlemi sık sık aritmetik çaprazlama işlemiyle karıştırılır. Aritmetik çaprazlama, iki vektörün genel doğrusal kombinasyonu iken mutasyon üç vektörün doğrusal kombinasyonunun özel bir halidir. Her iki durumda da mevcut amaç vektörlerinin doğrusal kombinasyonu olan yeni vektörler üretilmektedir. Kısacası DE algoritması, gerçek parametre optimizasyonuna rasgele amaç vektörlerinin doğrusal kombinasyonlarını örnekleyen, mutasyon ve rekombinasyon gibi iki basit operasyon kullanan normal gelişime dayalı bir yaklaşımdır. Aritmetik çaprazlama ile mutasyon birbirlerine benzemelerine rağmen amaç vektörleri uzayını oldukça farklı şekilde araştırırlar. Örneğin aritmetik çaprazlama, mevcut vektörü popülasyondaki diğer bir vektöre daha çok benzetmeye çalışırken mutasyon işlem vektörleri herhangi bir referans vektör (mevcut vektör) ihtiva etmemektedir. Referans vektörü yerine, araştırma eksenlerinin sayısını artıran ikinci rasgele seçilmiş bir amaç vektörü kullanmaktadır.

3.3.4. Seleksiyon

Seleksiyon işlemi, yeni üretilen vektörlerin hangi şartlar altında popülasyona girebileceğini tanımlayan bir kriterdir. Örneğin, turnuva seleksiyon işlemi rasgele seçilmiş vektör çiftleri arasında bir dizi yarışma gerçekleştirmek suretiyle yeni jenerasyonun üyesini belirlemektedir. Tipik olarak yarışan vektörler ebeveyn-aday popülasyonundan seçilir ve en fazla kazanan vektörlerin gelişmesine müsaade edilir. Gelişim stratejisi, alternatif olarak birleştirilmiş ebeveyn-çocuk popülasyonunun en iyi başarım gösteren üyelerinin gelişimine müsaade edildiği deterministik bir seleksiyon işlemi kullanmaktadır. Her iki seleksiyon işleminin dezavantajı, her jenerasyon sonunda popülasyonun kalite değerlerine göre sıraya dizilme maliyetidir. DE algoritmasının

seleksiyon işleminde, yeni üretilen vektör ebeveynine göre daha gelişmiş veya en azından aynı gelişme seviyesinde değilse ebeveyn vektör en az bir jenerasyon daha popülasyonda kalmaya devam etmekte ve başka bir vektörle yer değiştirmemektedir.

3.3.5. Rekombinasyon

Mutasyon işlemi, temel olarak bir popülasyonu güçlü yapmaktan ve yeni araştırma bölgelerinin araştırılmasından sorumludur. Rekombinasyon veya çaprazlama işlemi ise tamamlayıcı bir işlemdir ve amacı, var olan amaç vektör parametrelerinden faydalanarak yeni vektörleri oluşturmak suretiyle araştırmanın başarılı olması için yardımcı olmaktır. Rekombinasyonun etkisi üzerine oldukça ciddi çalışmalar ve tartışmalar mevcuttur. Ama kesin olan bir sonuç, uniform rekombinasyon işleminin araştırmanın hesaplama maliyetini önemli miktarda artırdığıdır. Ancak önemli miktarda da araştırmaya hızlılık sağlamaktadır.

DE algoritmasının ilk versiyonları özel uniform olmayan ayrık rekombinasyon işlemi kullanmıştır. Bu işlemde, ebeveynle değiştirilme ihtimali olan yeni çözüm oluşturulurken çözümün bazı elemanları ebeveynden bazıları ise mutasyona tabii tutulmuş vektörden alınmaktadır. Yeni aday çözümün oluşumunda, hangi vektörden daha çok elemanın alınacağını belirleyen frekansı kontrol etmek amacıyla hem üstel hem de binomial işlemler kullanılmıştır. Bu iki işlem, kullanıcı tanımlı, optimizasyon işlemi boyunca sabit tutulan ve çaprazlama oranı (CR) olarak adlandırılan kontrol parametresi aracılığıyla ayarlanır. Binomial uygulamada, 0 ile 1 arasında rasgele üretilen bir sayının CR'den küçük olması durumlarında elemanlar mutasyona uğramış çözümden, değilse ebeveynden alınmaktadır. Bunun tersine üstel uygulamada ise 0 ile 1 arasında rasgele üretilen bir sayının CR'den küçük olması anına kadar elemanlar ebeveynden, büyük olduğu andan itibaren de mutasyona uğramış çözümden alınmaktadır. Her iki durumda da yeni üretilen vektörün en az bir parametresinin ebeveynden farklı olması sağlanmaktadır [12].

3.4. DE Metodu

Rasgele çözümler üretmek için popülasyondan fark ve bilginin yönünü çıkarmak son derece önemli yakınsama özelliklerine sahip adaptif bir şekil ile sonuçlanır. DE algoritması mutasyon ve çaprazlama operatörlerinin özelliklerine göre ayrılabilir. Bu

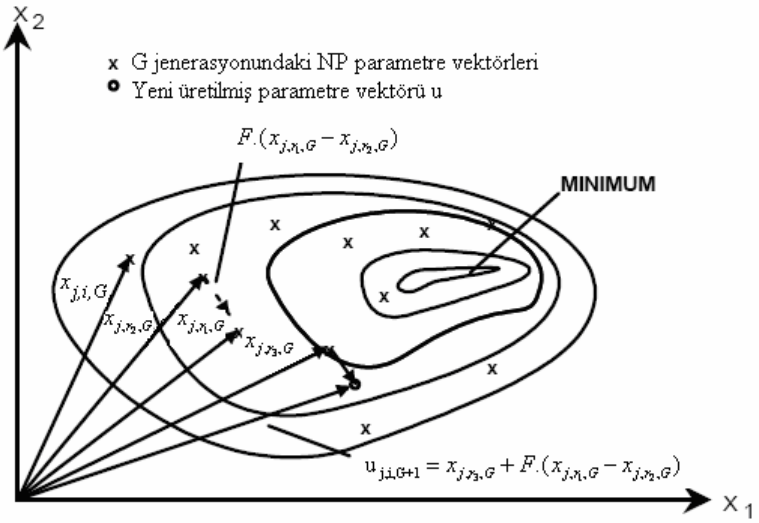
çalışmada iki en etkili DE algoritması kullanılmıştır. İlki, DE/rand/1/bin olarak gösterilen ve değişim vektörünün popülasyondan rasgele seçildiği sadece fark vektörünün mutasyona tabi tutulduğu algoritmadır [10,67,68]. İkinci DE ise kısaca DE/best/2/bin olarak gösterilen algoritmadır [10,68].

3.4.1. DE/rand/1/bin

DE'nin ilk şekli olan DE/rand/1/bin daha önce de ele alındığı gibi şöyle çalışmaktadır: her bir deneme vektörü $u_{i,j,G+1}$ aşağıdaki gibi elde edilmektedir:

$$r_1, r_2, r_3 \in \{1, 2, ..., NP\}, r_1 \neq r_2 \neq r_3 \neq i \text{(rasgele seçilmiş)}$$
$$j_{rand} \in \{1, 2, ..., D\}, \text{(rastgele seçilmiş)}$$
$$u_{j,i,G+1} = x_{j,r_3,G} + F.(x_{j,r_1,G} - x_{j,r_2,G})$$

F>0 olmak zorundadır. r_1, r_2, r_3 rasgele seçilmiştir, i'den farklı değerler almak zorundadır ve [1,NP] aralığındadır. F gerçel bir sayıdır ve $(x_{j,r_1,G} - x_{j,r_2,G})$ arasındaki farkı kontrol altına alan sabit bir faktördür. Şekil 3.3, DE/rand/1/bin'de rol oynayan fark vektörlerini gösteren iki boyutlu bir örneği ele almaktadır.



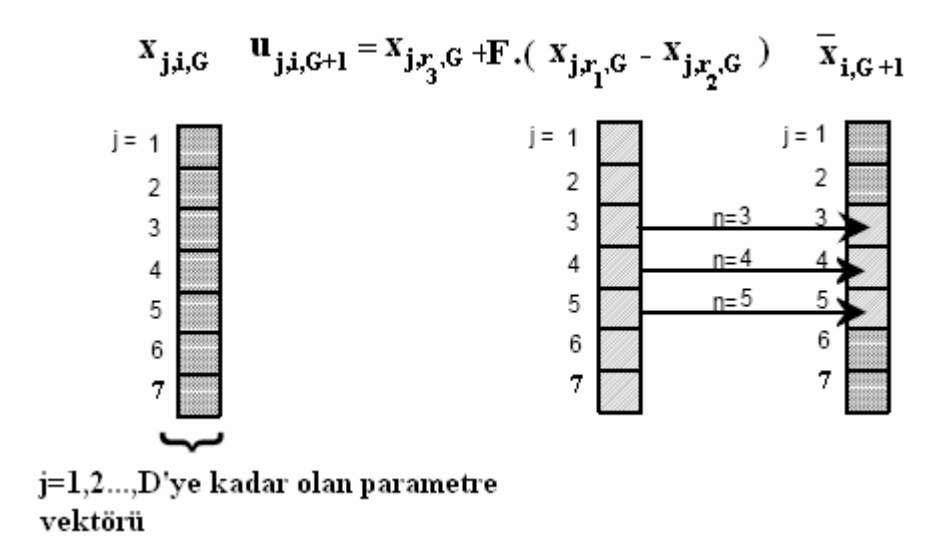
Şekil 3.3. DE/rand/1/bin'e göre $u_{j,i,G+1}$'i elde etmek için oluşan süreç ve dışhat(contour) çizgileriyle amaç fonksiyonunun gösterimi.

Parametre vektörleri arasındaki çeşitliliği artırmak için, vektör

$$\bar{x}_{i,G+1} = \begin{cases} \bar{u}_{i,G+1} & \text{eğer j=}\langle n \rangle_D, \langle n+1 \rangle_D, ..., \langle n+L-1 \rangle_D \\ \bar{x}_{i,G} & \text{diğer durumlarda} \end{cases}$$

şeklinde tanımlanır. Burada $\langle \ \rangle_D$ işareti D modunda olduğunu göstermektedir.

Yani $\bar{x}_{i,G+1}$ vektör elemanlarının belirli bir dizisi $u_{j,i,G+1}$'e özdeştir, diğer elemanları ise $x_{j,i,G}$'nin orijinal değerlerini kazanır. Mutasyon işlemi gelişime dayalı algoritmalardaki çaprazlamayla aynıdır. Bu fikir Şekil 3.4'te D=7, n=3 ve L=3 için gösterilmektedir. n [1,D] aralığında rasgele seçilmiş bir tamsayıdır. L [1,D] aralığında seçilmiş bir tamsayıdır. Bu rasgele sayılar n ve L yeni bir deneme vektörü elde etmek için kullanılmaktadır [10].



Şekil 3.4. D=7, n=3 ve L=3 için çaprazlama örneği

3.4.2. DE/best/2/bin

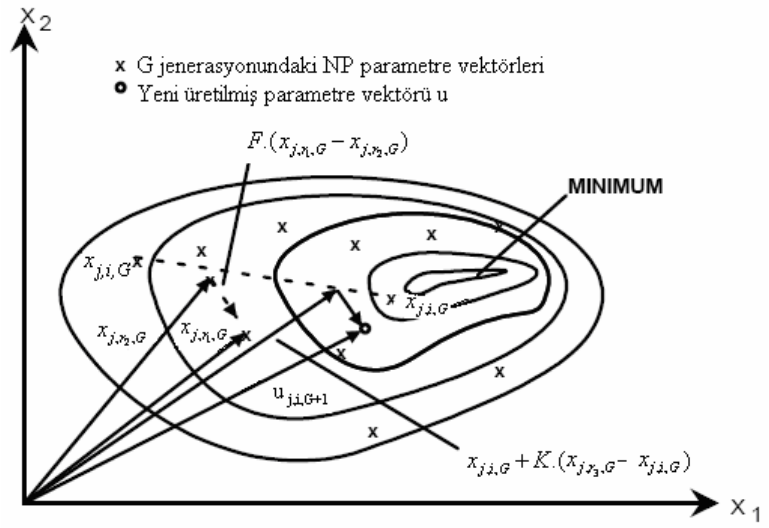
Temel olarak, DE/best/2/bin DE/rand/1/bin ile aynı şekilde çalışır fakat $u_{j,i,G+1}$'i aşağıdaki gibi üretir:

$r_1, r_2, r_3 \in \{1, 2, ..., NP\}, r_1 \neq r_2 \neq r_3 \neq i$(rasgele seçilmiş)

$j_{rand} \in \{1, 2, ..., D\}$, (rastgele seçilmiş)

$$\forall j \leq D, u_{j,i,G+1} = \begin{cases} x_{j,i,G} + K.(x_{j,r_3,G} - x_{j,i,G}) + F.(x_{j,r_1,G} - x_{j,r_2,G}) \\ \qquad \text{eğer } (\text{rand}_j[0,1) < CR \vee j = j_{rand}) \\ x_{j,i,G} \quad \text{diğer durumlarda} \end{cases}$$

Kontrol değişkeni K ile yöntem biraz değiştirilmiş olmaktadır. K kombinasyon faktörünü kullanmadaki ana fikir mevcut en iyi vektör $x_{j,i,G}$'yi de hesaba katarak DE/rand/1/bin'i geliştirmektir. Bu özellik kritik olmayan amaç fonksiyonlarında faydalı olabilir. Şekil 3.5 yukarıdaki eşitlikle tanımlanan vektör-jenerasyon ilişkisini göstermektedir. Bütün sabitler DE/rand/1/bin'e özdeştir [10].



Şekil 3.5. DE/best/2/bin'e göre $u_{j,i,G+1}$'i elde etmek için oluşan süreç ve kontür çizgileriyle amaç fonksiyonunun gösterimi.

4. BÖLÜM

DE ALGORİTMASI KULLANILARAK ALT TASARIMI

Bu bölüm üç ana alt bölümden oluşmaktadır. Birinci alt bölümde DE/rand/1/bin, DE/best/2/bin ve LMS algoritmasıyla tasarlanan ALT'nin başarımları değerlendirilecek ve karşılaştırılacaktır. İkinci alt bölümde farklı hata fonksiyonları kullanılarak DE algoritmasının ALT üzerindeki başarımı değerlendirilecek ve sonuçları karşılaştırılacaktır. Son bölümde ise bant genişliği farklı işaretler kullanılarak DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarıyla tasarlanan ALT'nın farklı bant genişliklerinde ayrı ayrı başarımları değerlendirilecek ve karşılaştırılacaktır.

4.1. DE Algoritması Kullanılarak ALT Tasarımı ve LMS Algoritmasıyla Başarımlarının Karşılaştırılması

ALT'ler, otomatik ayar yapabilen sayısal süzgeçlerdir. Bu süzgeçler katsayılarını giriş işaretlerine göre değiştirirler ve değişken işaret durumlarına karşılık süzgeç karakteristiklerinin değişimini gerektiren uygulamalarda kullanılırlar [69].

ALT'lerde iki giriş gerekmektedir: işaret ve referans girişi. Bir adaptif süzgeç kendi katsayılarını, yani ağırlıklarını yenileme kabiliyetine sahiptir. Yeni katsayılar bir katsayı üretecinden süzgece gönderilir. Katsayı üreteci gelen bir işarete cevaben katsayıları değiştiren adaptif bir algoritmadır. Çoğu uygulamalarda katsayı üretecinin amacı süzgeç katsayılarını referans girişe eşleştirmektir, bundan dolayı adaptif süzgeç referans girişi işaretten çıkararak aradaki farka göre çalışır.

İdeal olarak, yüksek yakınsama hızıyla ve düşük toleransla bilgisayar üzerinde basitçe gerçekleştirilebilen kolaylıkla hesaplanabilir ve nümerik olarak güçlü olan bir adaptif süzgeç yapısı istenir. Çoğu mühendislik problemlerinde olduğu gibi, bu arzu edilen karakteristikler çoğu durumda bir diğeriyle uyuşmamakta ve bir çeşit takas

gerektirmektedir. Örneğin, genellikle kullanılan en küçük ortalama kareler (LMS) algoritması hesaplanabilir kolaylığı olan ve nümerik olarak güçlü olan bir algoritmadır fakat yakınsama hızı açısından bir dezavantajı vardır.

4.1.1. Tasarım Hakkında Genel Bilgi

Bu çalışmada DE/rand/1/bin ve DE/best/2/bin olarak adlandırılan iki yeni DE algoritması ALT tasarımı için geliştirilmiş ve kullanılmıştır. Daha önce de anlatılmış olduğu gibi DE algoritmasının popülasyonu başlangıç parametre sınırları içerisinde rasgele başlatılır. Her bir G jenerasyonunda, popülasyon pertürbasyona uğrar. Birbirinden farklı üç birey popülasyondan rasgele seçilir. DE/rand/1/bin olarak isimlendirdiğimiz algoritmada popülasyondan rasgele seçilen üç vektörden ikisi arasındaki fark alınıp F ölçekleme faktörüyle adım büyüklüğü ölçeklendirildikten sonra üçüncü vektörle toplanarak deneme vektörü bulunur. DE/best/2/bin olarak adlandırılan algoritmada ise mevcut birey de hesaplamaya katılır. K katsayısı burada rasgele seçilen bir vektörle mevcut birey arasında meydana gelen birleşmenin seviyesini temsil eder.

Tipik olarak, eğer yeni birey halen seçili olan bireyden daha iyiyse bu durumda hali hazırdaki birey yeni olanla yer değiştirir. En iyi birey, algoritmada kullanılan hata fonksiyonunun iyilik değeriyle seçilir.

Buradaki seçilen en iyi vektör tasarımda kullanılacak süzgeç katsayılarıdır. Yani rasgele seçilen her bir birey mutasyon ve rekombinasyon işlemlerine maruz kaldıktan sonra süzgeç katsayılarının oluşumuna yardım etmektedir. Bu çalışmada, adaptif süzgeçlemede yaygın olarak kullanılan ve Eşitlik 4.1'de verilen MSE(Mean Square Error-Ortalama Karesel Hata) hata fonksiyonu kullanılmaktadır. MSE hata fonksiyonu, y(n) çıkış işaretinin d(n) arzu edilen işaretten çıkarılarak karelerinin alınması prensibine dayalıdır.

$$MSE= (1/N).\sum_{n=1}^{N} [\, d(n) - y(n) \,]^2 \tag{4.1}$$

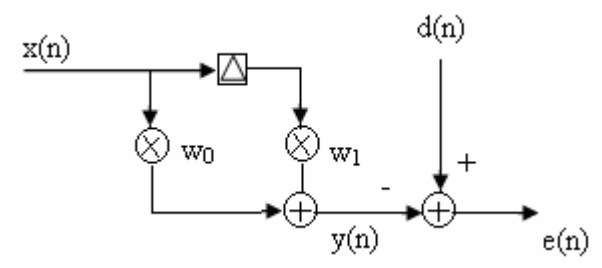
Gelişime dayalı birçok çalışma göstermektedir ki rekombinasyon ve mutasyon birbirlerinin etkisini artırmaktadır. DE'de, rekombinasyon CR katsayısı ile kontrol edilir. Price 1999'da CR'nin alabileceği değerleri listelemiştir [35]. Buna göre

mükemmel fakat yavaş bir optimizasyon için CR'nin değerinin 0.1, hızlı fakat riskli bir yakınsama için 1.0 olarak alınabileceği gösterilmiştir. Burada başlangıç için önerilen CR değeri 0.5'tir. Ayrıca ölçekleme faktörü F katsayısı için Storn ve Price tarafından önerilen değer $0.4 \le F \le 1.0$ arasındadır [70,71,72]. Cruz ve arkadaşlarının çalışmalarından da görülmektedir ki ölçekleme faktörü $0.4 \le F \le 0.6$ arasındaki değerlerde en iyi başarımı sergilemektedir [71].

4.1.2. Simülasyon Sonuçları

Simülasyon sonuçları iki şekilde değerlendirilmiştir. İlk olarak adaptif lineer süzgecin referans girişi $d(n) = 2\cos(2\pi n 25/N)$ olarak alındığında elde edilen sonuçlar değerlendirilecektir. Daha sonra ise referans işarete normal dağılımlı rasgele bir gürültü işareti eklendiği durumda elde edilen sonuçlar değerlendirilecektir.

4.1.2.1. Gürültüsüz Referans Giriş için ALT Tasarımı



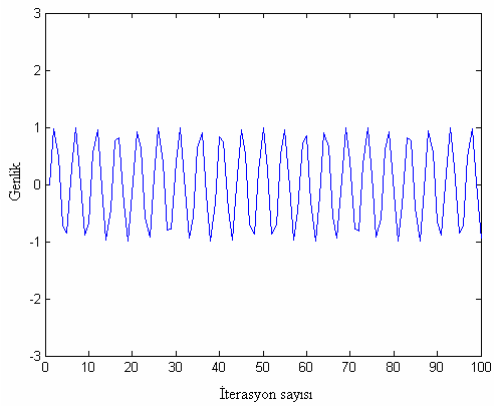
Şekil 4.1. İki katsayılı ALT gösterimi

Şekil 4.1'de, 2 ağırlıklı tek girişli ALT yapısı gösterilmektedir. Burada d(n) referans işareti herhangibir gürültüye maruz kalmamış 25 kHz'lik frekansa ve 500 kHz'lik bir örnekleme frekansına sahip kosinüs işaretini temsil etmektedir. Adaptif süzgeç girişi x(n), 25 kHz'lik bir frekansa ve N=500 kHz'lik bir örnekleme frekansına sahip sinüzoidal bir işaret olarak alınmıştır [55]. Bu çalışmada kullanılan işaretler, Eşitlik 4.2'te verilmektedir. Şekil 4.2'de ALT girişi olan sinüzoidal x(n) giriş işareti, Şekil 4.3'de ise arzu edilen işaret gösterilmektedir. Çalışmayla ilgili LMS algoritması ve DE algoritmasının temel adımları Ek-1'de verilmektedir.

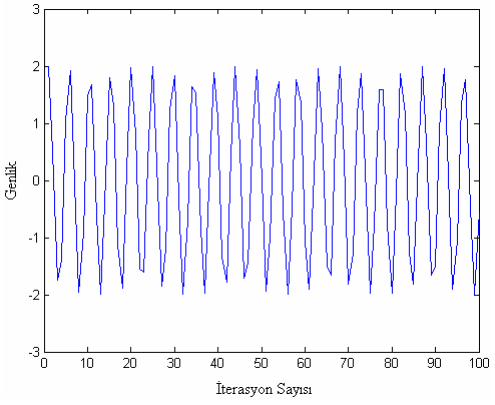
$$x(n) = \sin(2\pi n 25/N + n) \qquad \text{N=500} \qquad (4.2)$$
$$d(n) = 2\cos(2\pi n 25/N + n) \qquad \text{n=0,1,...,999}$$

Bu çalışmada kullanılan MSE sıfıra sürülecek hata fonksiyonu, y(n) izlenecek gerçek işaret ve d(n) arzu edilen işarettir [74-77].

İlk olarak literatürde çokça denenmiş ve kabul görmüş olan LMS algoritmasıyla ALT tasarımı gerçekleştirilmiştir. Daha sonra DE/rand/1/bin ve DE/best/2/bin algoritmalarıyla ALT tasarımı gerçekleştirilerek simülasyon sonuçları gösterilmiştir.



Şekil 4.2. Süzgeç girişi, x(n)



Şekil 4.3. Gürültüsüz referans işaret d(n)

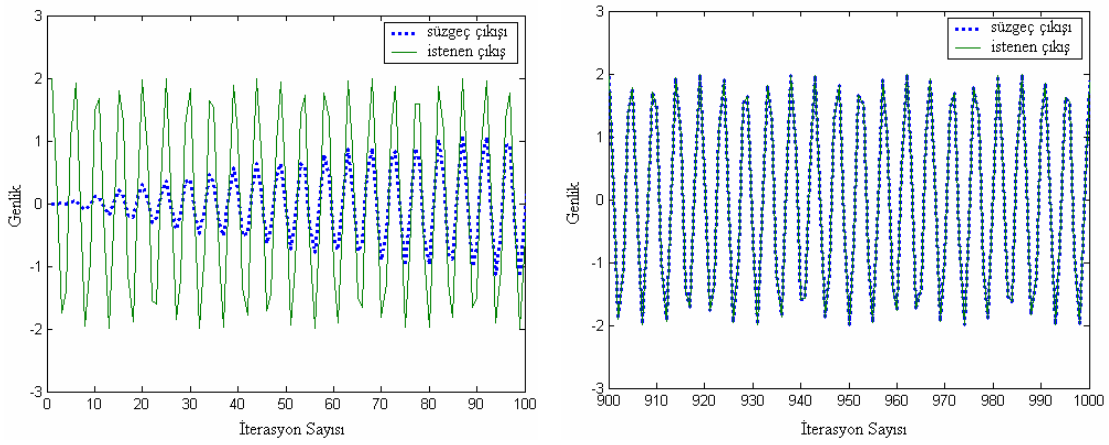
4.1.2.1.1. LMS Algoritmasıyla ALT Tasarımı

Gürültüsüz referans giriş işareti için ALT tasarımında kullanılan LMS algoritmasının kontrol parametreleri Tablo 4.1'deki gibi seçilmiştir.

Tablo 4.1. LMS Algoritmasının kontrol parametreleri

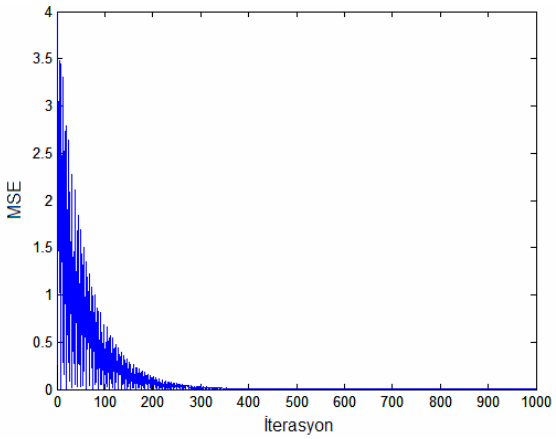
Adım büyüklüğü (μ)	*0.01*
İterasyon sayısı	*1000*

Tablo 4.1'den de anlaşılacağı gibi algoritmada kullanılan μ adım büyüklüğünün değeri 0.01, iterasyon sayısı ise 1000 olarak alınmıştır. Kullanılan bu kontrol parametreleriyle MSE hassaslığında hata minimize edilmeye çalışılmıştır. Şekil 4.4 LMS algoritması ile elde edilen çıkışla istenilen çıkışın karşılaştırmasını vermektedir.



(a) (b)

Şekil 4.4. LMS algoritması sonucu y(n) ve gürültüsüz referans d(n), (a)İlk 100, (b)Son 100 iterasyon



Şekil 4.5. LMS Algoritması için iterasyona göre hata fonksiyonunun gelişimi

Şekil 4.4.(a) ve (b)'de gürültüsüz referans giriş işareti için tasarlanan ALT çıkışında elde edilen sonuçlar verilmektedir. Şekil 4.4.(a)'da gürültüsüz referans giriş için tasarlanan ALT çıkışı ilk arzu edilen işaretle 100 iterasyon için verilmektedir ve Şekil 4.4.(b)'de elde edilen işaretin son 100 iterasyondaki şekliyle arzu edilen işaret verilmektedir. Şekil 4.4'ten de görüldüğü gibi ilk 100 iterasyonda süzgeç çıkışı arzu edilen işareti yakalayamamakta ve son 100 iterasyonda istenilen çıkışı sağlamaktadır. Şekil 4.5'te iterasyona göre hata fonksiyonu verilmektedir. Buradan da görüldüğü üzere LMS algoritmasıyla hatanın minimize edildiği en iyi sonuç 350 iterasyondan sonra elde edilmektedir. Hata 350 iterasyondan sonra sıfıra çok yakın bir değerde kalmış fakat hiçbir zaman sıfır değerine ulaşmamıştır.

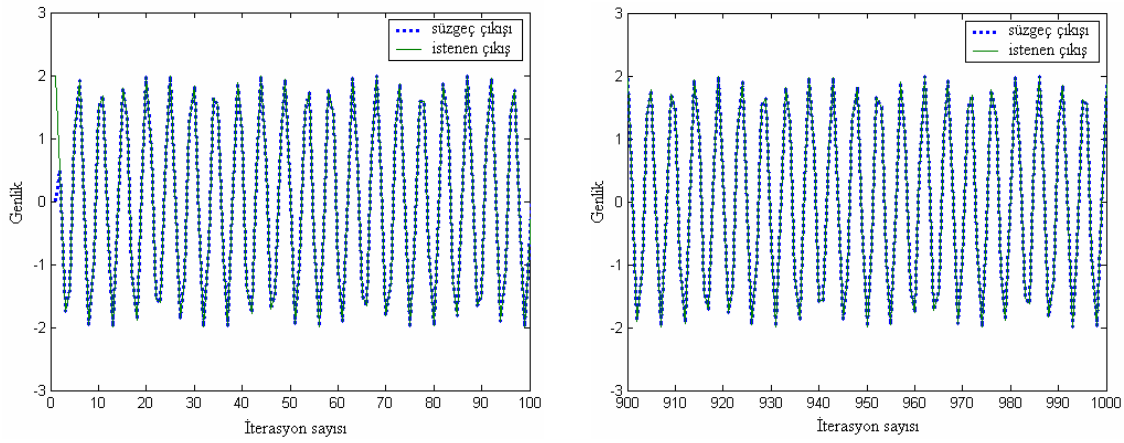
4.1.2.1.2. DE/rand/1/bin ile ALT Tasarımı

Gürültüsüz referans giriş işareti için tasarlanan ALT bu kez DE/rand/1/bin ile gerçekleştirilmiştir. Tasarımda kullanılan DE/rand/1/bin'nin kontrol parametreleri Tablo 4.2'deki gibi seçilmiştir.

Tablo 4.2. DE'nin kontrol parametreleri

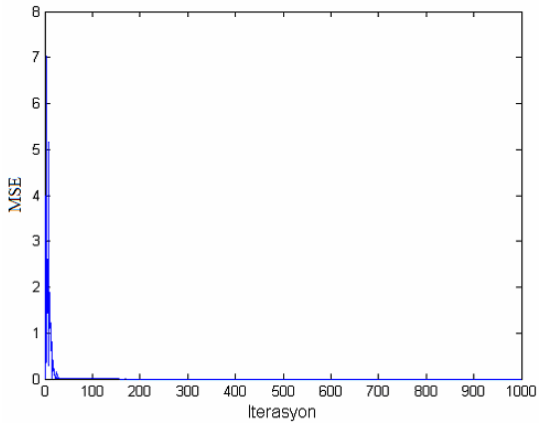
Popülasyon büyüklüğü (NP)	*100*
Çaprazlama oranı (CR)	*0.5*
Ölçekleme faktörü (F)	*0.4*
İterasyon sayısı (Gmax)	*1000*

Şekil 4.6 (a)'da gürültüsüz referans işaret için tasarlanmış ALT çıkışı ile arzu edilen işaretin ilk 100 örneği verilmektedir. Şekil 4.6 (b)'de gürültüsüz işaret için tasarlanan ALT çıkışında elde edilen işaretle istenilen işaretin son 100 örneği verilmektedir. Şekil 4.7'de ise ALT tasarımında kullanılan ve elde edilen katsayıların iyilik değerini belirleyen MSE hata fonksiyonunun iterasyona göre gelişimi verilmektedir.



(a) (b)

Şekil 4.6. DE/rand/1/bin sonucu süzgeç çıkışı y(n) ve istenen gürültüsüz çıkış d(n). (a) İlk 100, (b) Son 100 iterasyon



Şekil 4.7. Gürültüsüz referans giriş için DE/rand/1/bin için iterasyonda hata

Gürültüsüz referans giriş işaret için tasarlanan ALT çıkışıyla arzu edilen işaret Şekil 4.6 (a) ve (b)'de verilmektedir. Şekil 4.6.(a)'dan görüldüğü gibi ilk 100 iterasyonda süzgeç çıkışı arzu edilen işarete çok yaklaşmış, Şekil 4.6 (b)'den de görüldüğü gibi son 100 iterasyonda istenilen çıkışı sağlamıştır. Şekil 4.7'de iterasyona göre hata fonksiyonu verilmektedir. Buradan da görüldüğü üzere DE/rand/1bin algoritmasıyla hatanın sıfıra inmesinden dolayı en iyi sonuç 150 iterasyondan sonra elde edilmektedir.

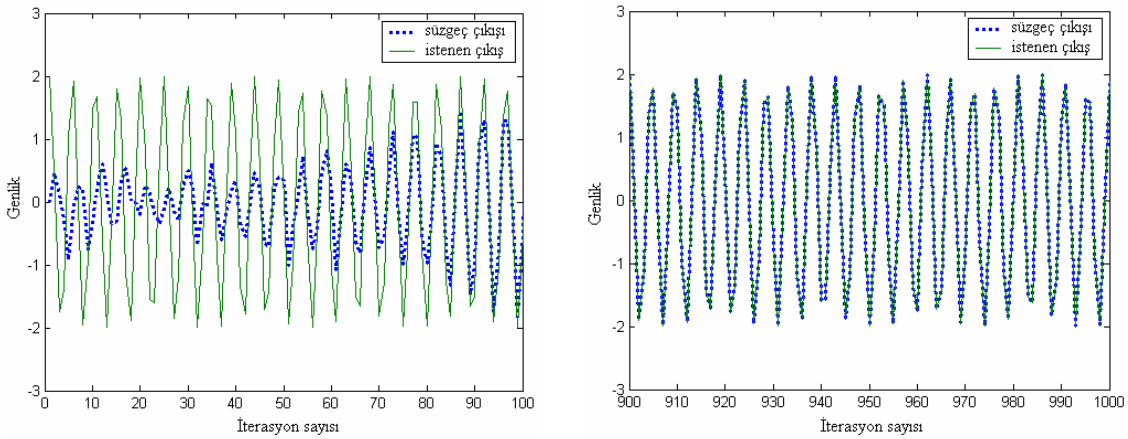
4.1.2.1.3. DE/best/2/bin ile ALT Tasarımı

Burada gürültüsüz referans giriş için ALT tasarımı DE/best/2/bin algoritması ile

yapılmaktadır. Tasarımda kullanılan DE/best/2/bin'nin kontrol parametreleri Tablo 4.3'deki gibi seçilmiştir.

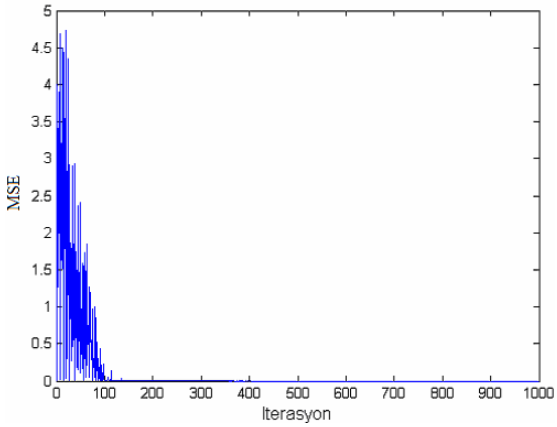
Tablo 4.3. DE/best/2/bin'nin kontrol parametreleri

Popülasyon büyüklüğü(NP)	*100*	*Ölçekleme faktörü (F)*	*0.4*
Çaprazlama oranı (CR)	*0.5*	*İterasyon sayısı (Gmax)*	*1000*
Birleşme faktörü (K)	*0.2*		



(a) (b)

Şekil 4.8. DE/best/2/bin sonucu süzgeç çıkışı y(n) ve gürültüsüz istenen çıkış d(n). (a) İlk 100, (b) Son 100 iterasyon



Şekil 4.9. Gürültüsüz referans için DE/best/2/bin'de iterasyona göre hata fonksiyonunun gelişimi

Şekil 4.8'de adaptif süzgeç çıkışında istenilen işaretle DE/best/2/bin ile tasarlanan süzgeç çıkışındaki işaretin karşılaştırılması verilmektedir. Elde edilen sonuçlar, (a) da

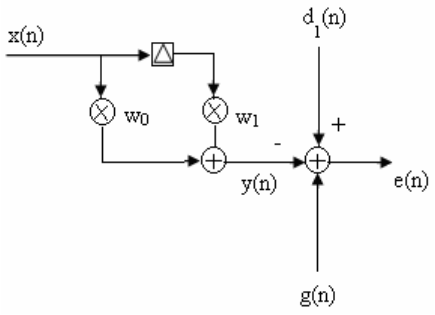
ilk 100 iterasyonda işaretin henüz yakalanamadığın ve (b)'de son 100 iterasyonda istenilen çıkışın sağlandığını göstermektedir. Şekil 4.9'da iterasyona göre hata fonksiyonu verilmektedir. Buradan da görüldüğü üzere DE/best/2/bin algoritmasıyla hatanın sıfıra inmesiyle en iyi sonuç 400 iterasyondan sonra elde edilmektedir.

Tablo 4.4. LMS Algoritması, DE/rand/1/bin ve DE/best/2/bin ile Adaptif Lineer Toplayıcı tasarımında elde edilen ağırlıklar

Ağırlıklar	Elde Edilen Ağırlıklar		
	LMS Algoritması	DE/rand/1/bin Algoritması	DE/best/2/bin Algoritması
w_0	0.5241	0.5248	0.5248
w_1	-2.0670	-2.0677	-2.0677

Tablo 4.4 simülasyon sonucu LMS, DE/rand/1/bin ve DE/best/2/bin algoritmaları ile elde edilen süzgeç ağırlıklıklarını göstermektedir. Görüldüğü her iki DE algoritmasıyla elde edilen katsayılar aynı ve LMS algoritmasıyla elde edilen katsayılar da bu katsayılara çok yakın değerlerdir. Buradan da anlaşılmaktadır ki her iki DE algoritmasında da hata sıfırlanmış ve ağırlıklar olması gereken değerlerini almışlardır. Bu nedenle her iki DE algoritması ile tasarlanan ALT ağırlıkları birbirine eşit çıkmıştır. LMS algoritması ise hatayı sıfırlayamamış fakat çok küçülmüştür. Bu nedenle de DE algoritmasıyla elde edilen ağırlıklara çok yakın bir değerde kalmıştır.

4.1.2.2. Gürültülü Referans Giriş için ALT Tasarımı



Şekil 4.10. Arzu edilen işaretin, gürültü işaretiyle sinüzoidal işaretin toplamından oluştuğu durumda iki katsayılı ALT gösterimi

Daha önce verilen referans işarete, bu çalışmada g(n) gürültü işareti eklenmiştir. g(n)

normal dağılımlı rasgele bir gürültü işaretidir ve SNR değeri 10 dB olarak alınmıştır.

Şekil 4.10, arzu edilen işaret d(n)'in, gürültü işareti g(n) ile sinüzoidal işaret $d_1(n)$'in toplamından oluştuğu durumdaki iki katsayılı ALT gösterimini vermektedir. Şekilde x(n) olarak gösterilen işaret sinüzoidal bir giriş işaretidir. y(n), gürültülü referans giriş için tasarlanan ALT çıkışında elde edilen işareti ve e(n) de arzu edilen işaretle, süzgeç çıkışındaki işaret arasındaki hatayı göstermektedir.

$$x(n) = \sin(2\pi n 25/N + n) \qquad \text{N=500} \qquad (4.3)$$
$$d_1(n) = 2\cos(2\pi n 25/N + n) \qquad \text{n=0,1,...,999}$$
$$d(n) = d_1(n) + g(n)$$

x(n) giriş işareti, yine 25 kHz'lik bir frekansa ve N=500 kHz'lik bir örnekleme frekansına sahip sinüzoidal bir işaret olarak alınmıştır. Hata fonksiyonu olarak burada da MSE hata fonksiyonu kullanılmıştır. Çalışmada kullanılan gürültü işaretlerinin yazılımı Ek-1'de verilmektedir.

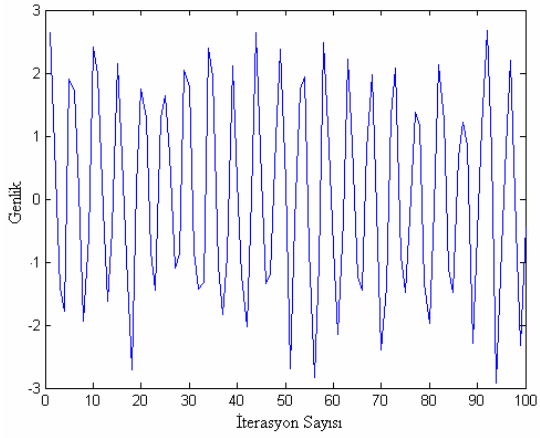
Gürültülü referans giriş kullanıldığı durumlarda LMS algoritması, DE/rand/1/bin ve DE/best/2/bin algoritmalarının ALT tarasımındaki başarımları değerlendirilmiştir. Gürültü işareti rasgele dağılımlı bir işaret olduğu için bütün algoritmaların arzu edilen işareti farklı olacaktır.

4.1.2.2.1. Gürültülü Referans Giriş için LMS Algoritması ile ALT Tasarımı

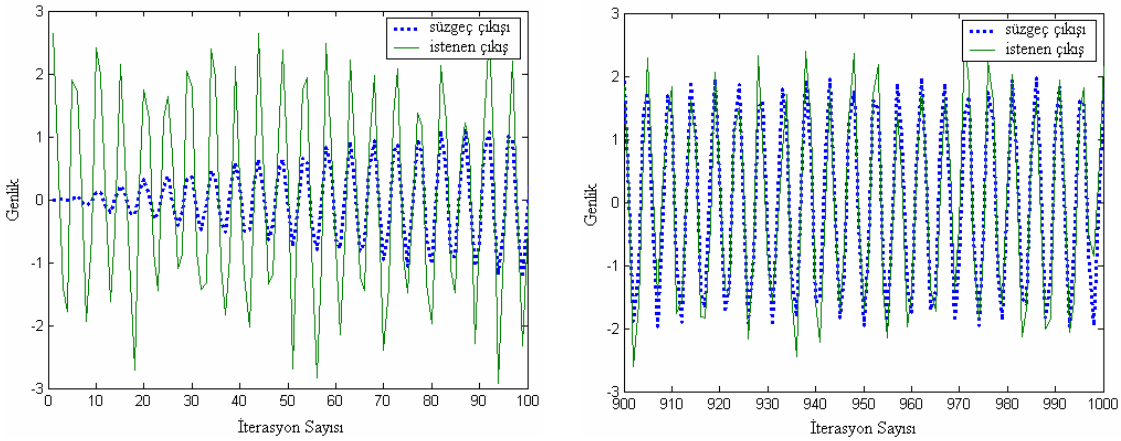
Bu çalışmada LMS algoritması gürültülü referans giriş için ALT tasarımında kullanılmıştır. Bu çalışmada kullanılan LMS algoritmasının kontrol parametreleri Tablo 4.5'teki gibi seçilmiştir.

Tablo 4.5 LMS Algoritmasının kontrol parametreleri

Adım büyüklüğü (μ)	*0.01*
İterasyon sayısı	*1000*



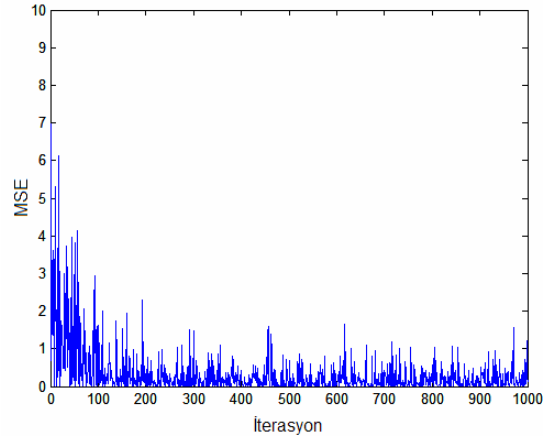
Şekil 4.11. LMS algoritması ile yakalanacak gürültülü istenilen işaret, d(n)



(a) (b)

Şekil 4.12. LMS algoritması sonucu süzgeç çıkışı y(n) ve istenen gürültülü çıkış d(n). (a) İlk 100, (b) Son 100 iterasyon

Şekil 4.12 (a), (b)'de gürültülü referans giriş işareti için tasarlanan ALT çıkışında elde edilen sonuçlar verilmektedir. Şekil 4.12 (a)'da gürültülü referans giriş için tasarlanan ALT çıkışı ilk arzu edilen işaretle 100 iterasyon için verilmektedir. Şekil 4.12 (b)'de elde edilen işaretin son 100 iterasyondaki şekliyle arzu edilen işaret verilmektedir. Şekil 4.12'den de görüldüğü gibi ilk 100 iterasyonda süzgeç çıkışı arzu edilen işareti yakalayamamakta, son 100 iterasyonda istenilen çıkışa yaklaşmaktadır.



Şekil 4.13. Gürültülü referans için LMS Algoritmasında iterasyona göre hata

Şekil 4.13'te iterasyona göre MSE hata fonksiyonunun gelişimi verilmektedir. Şekilden de görülmektedir ki LMS algoritması gürültülü işareti iyi yakınsayamamıştır.

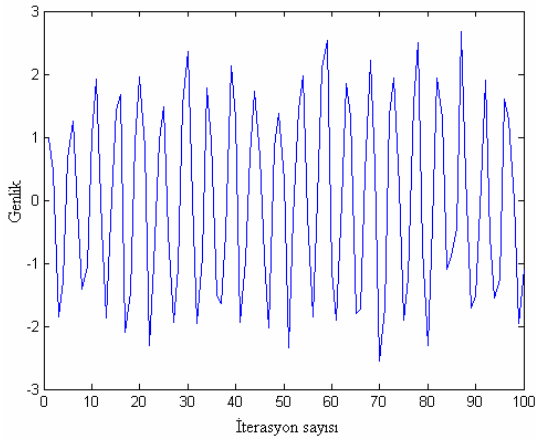
4.1.2.2.2. Gürültülü Referans Giriş için DE/rand/1/bin ile ALT Tasarımı

DE/rand/1/bin bu kez gürültülü referans giriş işareti için ALT tasarımında kullanılmıştır. Tasarımda kullanılan DE/rand/1/bin'nin kontrol parametreleri Tablo 4.5'deki gibi seçilmiştir.

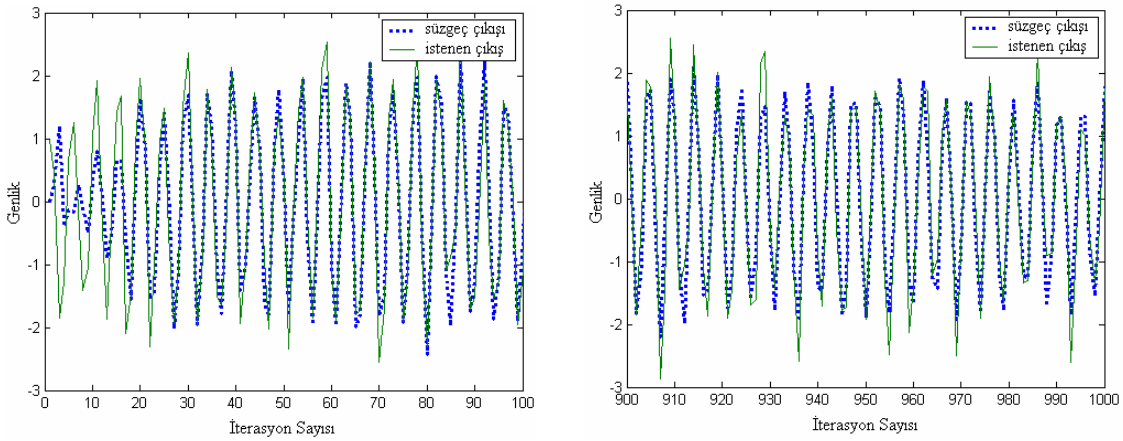
Tablo 4.5 DE'nin kontrol parametreleri

Popülasyon büyüklüğü (NP)	*100*
Çaprazlama oranı (CR)	*0.5*
Ölçekleme faktörü (F)	*0.4*
İterasyon sayısı (Gmax)	*1000*

Şekil 4.14'te ALT'ye uygulanan gürültülü referans işaret verilmektedir. Şekil 4.15 (a)'da gürültülü referans işaret için tasarlanmış ALT çıkışı ile arzu edilen işaretin ilk 100 örneği verilmektedir. Şekil 4.15 (b)'de gürültülü işaret için tasarlanan ALT çıkışında elde edilen işaretle istenilen işaretin son 100 örneği verilmektedir. Şekil 4.16'da ise ALT tasarımında kullanılan MSE hata fonksiyonunun iterasyona göre gelişimi verilmektedir.

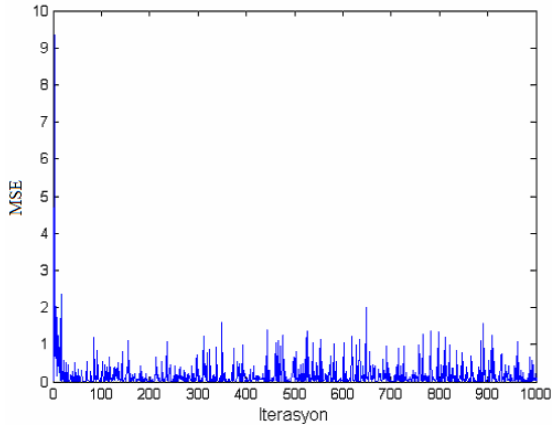


Şekil 4.14. DE/rand/1/bin ile yakalanacak gürültülü istenilen işaret, d(n)



(a) (b)

Şekil 4.15. DE/rand/1/bin sonucu süzgeç çıkışı y(n) ve istenen gürültülü çıkış d(n). (a) İlk 100, (b) Son 100 iterasyon



Şekil 4.16. Gürültülü referans için DE/rand/1/bin ile iterasyona göre hatanın gelişimi

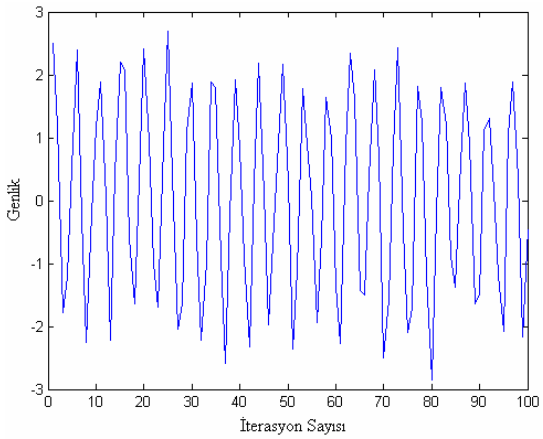
Şekil 4.15 (a), (b)'de gürültülü referans giriş işareti için tasarlanan ALT çıkışında DE/rand/1/bin ile elde edilen sonuçlar verilmektedir. Şekil 4.15'den de görüldüğü gibi ilk 100 iterasyonda süzgeç çıkışı arzu edilen işarete yaklaşmakta, son 100 iterasyonda istenilen çıkışa yakınsamaktadır. Şekil 4.16'te iterasyona göre DE/rand/1/bin'deki hata fonksiyonu verilmektedir. Buradan hatanın zaman zaman sıfırı yakaladığı fakat gürültüden dolayı süreklilik gösteremediği görülmektedir.

4.1.2.2.3. Gürültülü Referans Giriş için DE/best/2/bin ile ALT Tasarımı

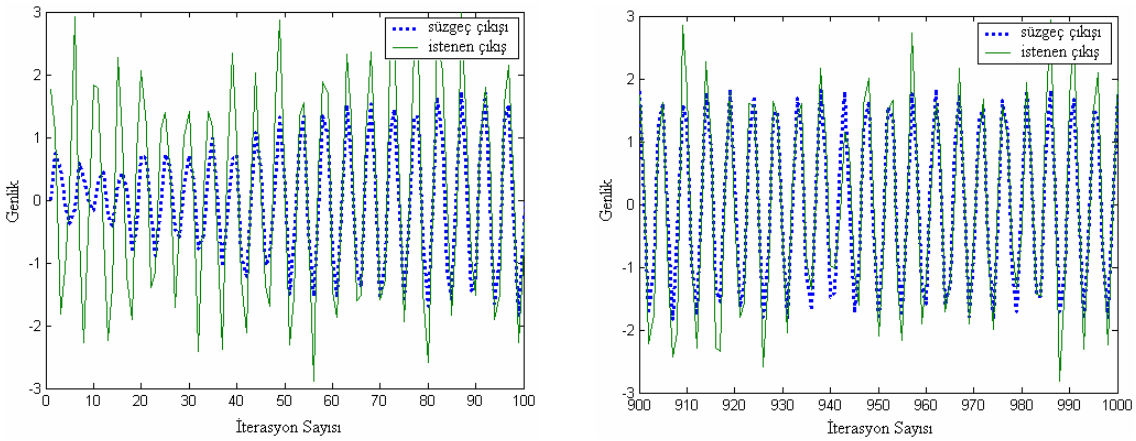
DE/best/2/bin bu kez gürültülü referans giriş işareti için ALT tasarımında kullanılmıştır. Tasarımda kullanılan DE/best/2/bin'nin kontrol parametreleri Tablo 4.6'deki gibi seçilmiştir.

Tablo 4.6 DE'nin kontrol parametreleri

Popülasyon büyüklüğü(NP)	*100*	*Ölçekleme faktörü (F)*	*0.4*
Çaprazlama oranı (CR)	*0.5*	*İterasyon sayısı (Gmax)*	*1000*
Birleşme faktörü (K)	*0.2*		

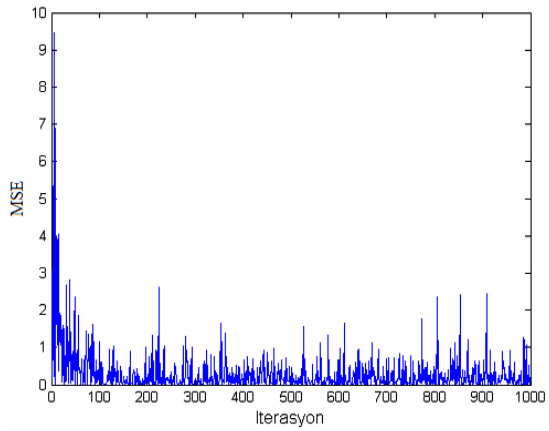


Şekil 4.17. DE/best/2/bin ile yakalanacak gürültülü istenilen işaret, d(n)



(a) (b)

Şekil 4.18. DE/best/2/bin sonucu süzgeç çıkışı y(n) ve istenen gürültülü çıkış d(n). (a) İlk 100, (b)son 100 iterasyon



Şekil 4.19. Gürültülü referans için DE/best/2/bin'de iterasyona göre hata

Şekil 4.18'te ALT'ye uygulanan gürültülü referans işaret verilmektedir. Şekil 4.18 (a)'da gürültülü referans işaret için tasarlanmış ALT çıkışı ile arzu edilen işaretin ilk 100 örneği verilmektedir. Şekil 4.18 (b)'de gürültülü işaret için tasarlanan ALT çıkışında elde edilen işaretle istenilen işaretin son 100 örneği verilmektedir. Şekil 4.19'da ise ALT tasarımında kullanılan MSE hata fonksiyonunun iterasyona göre gelişimi verilmektedir.

Tablo 4.7'de LMS algoritması, DE/rand/1/bin ve DE/best/2/bin algoritmaları kullanılarak gürültülü referans girişe göre tasarlanan ALT için ağırlıklar verilmektedir.

Bu kısımda, önceki kısımdaki gürültüsüz referans giriş işaretin olduğu durumda tasarlanan ALT yapısına gürültülü bir referans işareti uygulanmış ve sonuçları verilmiştir.

Tablo 4.7. Gürültülü referans giriş için LMS Algoritması, DE/rand/1/bin ve DE/best/2/bin ile elde edilen ağırlıklar

Ağırlıklar	Elde Edilen Ağırlıklar		
	LMS Algoritması	DE/rand/1/bin Algoritması	DE/best/2/bin Algoritması
w_0	0.5328	0.6283	0.4326
w_1	-2.0376	-1.9991	-1.8938

Şekil 4.12, 4.15 ve 4.18'de görüldüğü gibi LMS, DE/rand/1/bin ve DE/best/2/bin algoritmaları birbirine yakın başarımlar göstermişlerdir. Şekil 4.13, Şekil 4.16 ve Şekil 4.19'da hata fonksiyonlarına bakıldığında hata değeri LMS algoritmasında ilk 100 iterasyondan sonra 1'in altına düşebilmiş, DE/rand/1/bin'de hata değeri ilk 10 iterasyondan sonra ve DE/best/2/bin'de ise ilk 100 iterasyondan sonra 1'in altına düşmeyi başarmıştır. Bu şekillerden hatanın gelişimine bakıldığında ise bütün algoritmalarda zaman zaman hata değeri 1'in üstüne çıkmış fakat sıfıra en çok yaklaşan algoritma DE/rand/1/bin olmuştur.

4.2. Farklı Hata Fonksiyonları Kullanılarak DE Algoritması ile ALT Tasarımı

Bu bölümde, farklı hata fonksiyonları ALT tasarımında kullanılarak en iyi değeri verecek hata fonksiyonu simülasyon sonuçlarıyla birlikte değerlendirilmiştir.

ALT tasarımında, yaygın olarak kullanılan hata fonksiyonları aşağıda tanımlanmıştır.

Ortalama Karesel Hata (MSE): Eşitlik 4.4'te tanımlandığı gibi MSE hata fonksiyonu çıkış işareti y(n)'in, arzu edilen işaret d(n)'den çıkarılarak karelerinin alınması ve ortalamasının kullanılması fikrine dayalıdır.

$$MSE = \frac{1}{N}\sum_{n=1}^{N}\left[d(n)-y(n)\right]^2 \tag{4.4}$$

En Küçük Karesel Hata (LMS): Eşitlik 4.5'te tanımlandığı gibi En Küçük Karesel Hata fonksiyonu, arzu edilen işaret d(n)'den tasarlanan süzgecin çıkışı y(n)'in çıkarılarak elde edilmiş olan hataların kareleri alınarak toplanması ve toplamın karekökünün minimize edilmesi amacını taşımaktadır.

$$LMS = 1/N \left\{ \sum_{n=1}^{N} [d(n) - y(n)]^2 \right\}^{1/2} \tag{4.5}$$

Ortalama Mutlak Hata (MAE): Eşitlik 4.6'da tanımlandığı gibi Ortalama Mutlak Hata fonksiyonu arzu edilen işaret d(n) ile süzgeç çıkışı y(n) arasındaki farkın, mutlak değerler toplamının en küçük yapılmasını amaçlamaktadır.

$$MAE = 1/N \sum_{n=1}^{N} |d(n) - y(n)| \tag{4.6}$$

4.2.1. ALT Tasarımında Farklı Hata Fonksiyonları Kullanıldığı Durumda Elde Edilen Simülasyon Sonuçları ve Değerlendirme

Adaptif süzgeç girişi x(n), 25 kHz'lik bir frekansa ve N=500 kHz'lik bir örnekleme frekansına sahip sinüzoidal bir işaret olarak alınmıştır. Bu çalışmada kullanılan işaretler, Eşitlik 4.7'de verildiği gibi alınmıştır.

$$x(n) = \sin(2\pi n 25 / N) \qquad \text{N=500} \tag{4.7}$$
$$d(n) = 2\cos(2\pi n 25 / N) \qquad \text{n=0,1,...,999}$$

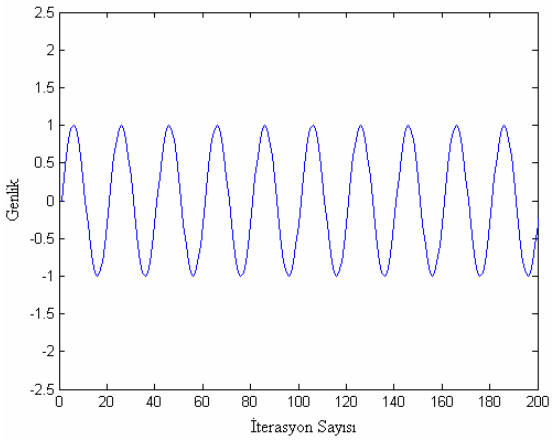
Bu çalışmada, önceki çalışmalardaki simülasyon sonuçlarıyla iterasyona göre hata fonksiyonlarının başarımı değerlendirildiğinde en iyi algoritma olarak belirlenen DE/rand/1/bin, hata fonksiyonlarının başarımını değerlendirmede kullanılmıştır. Hata fonksiyonlarının sonuçlardaki etkisini değerlendirmek için MSE, LMS ve MAE hata fonksiyonları DE/rand/1/bin algoritmasıyla bu uygulamada ayrı ayrı kullanılmış ve değerlendirilmiştir.

4.2.2. DE/rand/1/bin için MSE, LMS ve MAE Hata Fonksiyonlarının Değerlendirilmesi

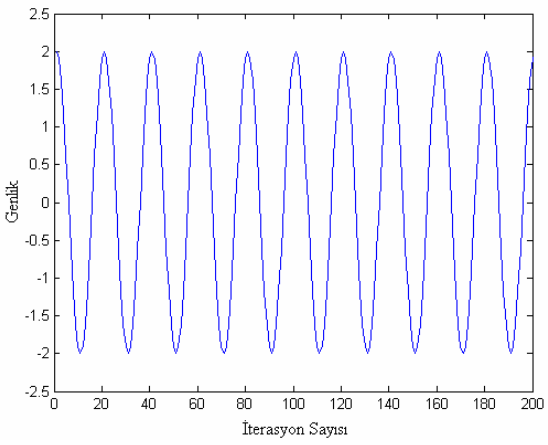
Şekil 4.20 ve Şekil 4.21'de ALT girişine uygulanan 25 kHz frekansa ve 500 kHz örnekleme frekansına sahip sinüzoidal işaretle referans girişe uygulanan arzu edilen işaretin ilk 100 örneği gösterilmektedir. Tablo 4.8'de de DE/rand/1/bin için kullanılan kontrol parametreleri verilmektedir.

Tablo 4.8. DE/rand/1/bin'nin kontrol parametreleri

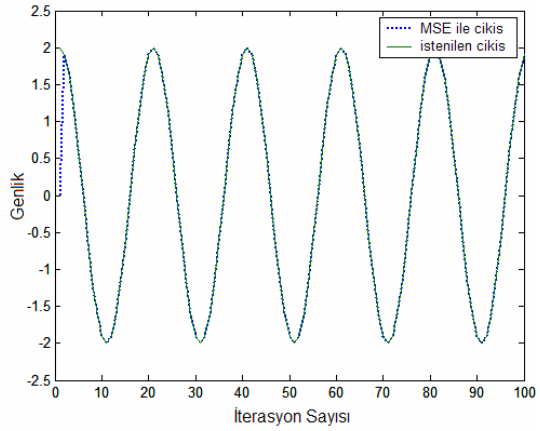
Popülasyon büyüklüğü	*100*
Çaprazlama oranı	*0.5*
Ölçekleme faktörü	*0.4*
İterasyon sayısı	*1000*



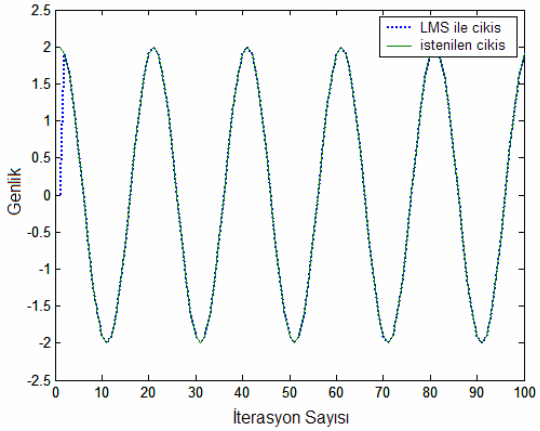
Şekil 4.20. Hata fonksiyonlarının başarımlarını değerlendirmede kullanılan ALT girişi



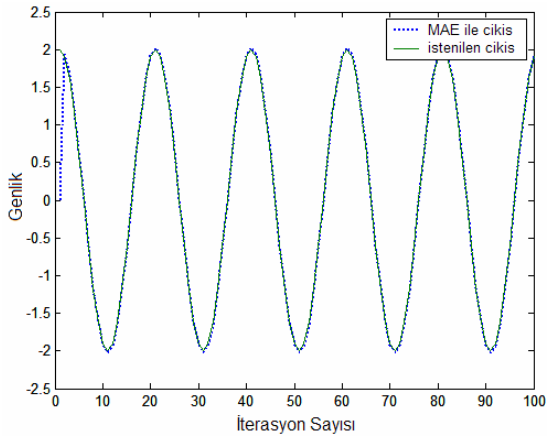
Şekil 4.21. Hata fonksiyonlarının başarımlarını değerlendirmede kullanılan ALT referans girişi



(a)



(b)

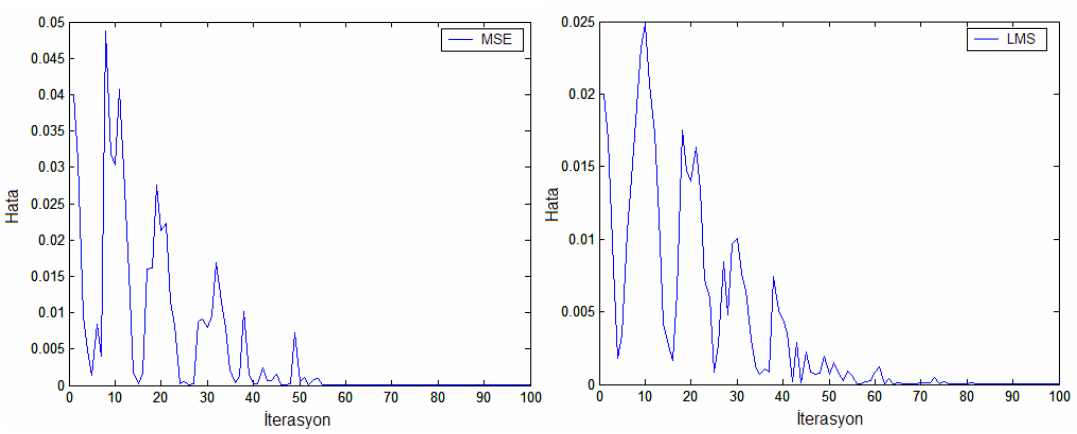


(c)

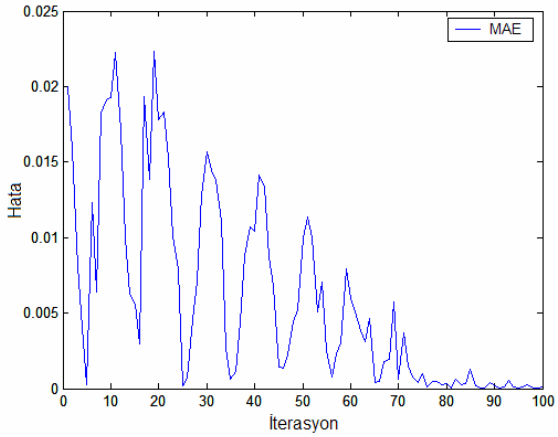
Şekil 4.22. Hata fonksiyonları için arzu edilen işaretle süzgeç çıkışının karşılaştırılması. (a) MSE (b)LMS, (c) MAE

Şekil 4.22 (a)'da MSE hata fonksiyonunun kullanıldığı DE/rand/1/bin algoritmasıyla tasarlanan ALT çıkış işareti y(n) ile tasarım sonucunda elde edilmek istenen referans işaret d(n)'in ilk 100 örneği verilmektedir. Şekil 4.22 (b)'de LMS hata fonksiyonunun kullanıldığı DE/rand/1/bin algoritmasıyla elde edilmek istenen referans işaret d(n) ve tasarlanan ALT çıkışı y(n)'nin ilk 100 örneği verilmektedir. Şekil 4.22 (c)'de ise DE/rand/1/bin algoritması kullanılarak tasarlanan ALT çıkışı y(n) ile istenen çıkış

d(n)'in ilk 100 örneği MAE hata fonksiyonunun kullanıldığı durum için verilmektedir.



(a) (b)



(c)

Şekil 4.23. İterasyona göre hata değerleri. (a) MSE, (b)LMS, (c) MAE

Şekil 4.20 ALT girişini ve Şekil 4.21'de ALT'nin referans girişi olan arzu edilen işareti göstermektedir. Şekil 4.22 (a),(b),(c) de ALT çıkışı ile istenilen çıkış yani arzu edilen işaretin ilk 100 örneğinin karşılaştırmasını vermektedir. Şekil 4.23 (a),(b),(c) ise tasarımda kullanılan hata fonksiyonlarının iterasyona göre gelişimini göstermektedir.

Şekil 4.23 (a)'da görülmektedir ki MSE hata fonksiyonu 55 iterasyondan sonra hatayı

minimize etmiş ve bu değeri kalan 45 iterasyon boyunca tamamen sıfıra düşürmeye çalışmıştır. Şekil 4.23 (b)'de LMS hata fonksiyon 75 iterasyondan sonra hatayı minimize etmeyi başarmış ve bu değeri kalan 25 iterasyon boyunca sıfıra yaklaştırmaya devam etmiştir. Şekil 4.23 (c)'de ise MAE hata fonksiyonuyla tasarlanan süzgeçte hatanın 100 iterasyon sonunda minimize edildiği görülmektedir. Hata fonksiyonlarının gelişiminden de anlaşılmaktadır ki adaptif lineer toplayıcı tasarımında kullanılabilecek en uygun hata fonksiyonu MSE hata fonksiyonudur.

100 iterasyon sonunda elde edilen ALT ağırlıkları Tablo 4.9'de gösterilmektedir.

Tablo 4.9. MSE, LMS ve MAE hata fonksiyonları ile elde edilen ağırlıklar

Ağırlıklar	Elde Edilen Ağırlıklar		
	MSE	LMS	MAE
w_0	6.1601	6.1554	6.2550
w_1	-6.4767	-6.4722	-6.5484

4.3. DE/rand/1/bin, DE/best/2/bin ve LMS Algoritmalarıyla Tasarlanan ALT'lerin İşaret Bant Genişliğine göre Gösterdikleri Davranışların Karşılaştırılması

Bu bölümde, adaptif süzgecin giriş ve referans giriş işaretlerinin bantgenişliğine göre LMS, DE/ran/1/bin ve DE/best/2/bin algoritmaları ile tasarımı gerçekleştirilecek ve başarımları karşılaştırılacaktır.

4.3.1. Giriş İşareti Frekansının 25 kHz olduğu ALT Tasarımı

Bu uygulamada 6 ağırlıklı bir ALT yapısı ele alınacaktır. İlk olarak giriş işaretimiz 25 kHz'lik bir frekansa ve 500 kHz'lik bir örnekleme frekansına sahip sinüzoidal bir dalga olarak alınmıştır. Giriş işareti ve arzu edilen işaret Eşitlik 4.8'de verilmektedir. Burada x(n) olarak gösterilen işaret ALT girişindeki sinüzoidal işareti, d(n) olarak verilen işaret ise adaptif süzgeç referans girişi olan arzu edilen işareti göstermektedir. n, 0'dan 999'a kadar olan örnek sayısını ve N de kHz olarak örnekleme frekansını göstermektedir.

$$x(n) = \sin(2\pi n 25/N) \qquad \text{N=500} \qquad (4.8)$$
$$d(n) = 2\cos(2\pi n 25/N) \qquad \text{n=0,1,...,999}$$

Tablo 4.10, Tablo 4.11 ve Tablo 4.12'de sırasıyla DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarının bu çalışmada kullanılan kontrol parametreleri verilmektedir.

Tablo 4.10. DE/rand/1/bin'nin kontrol parametreleri

Popülasyon büyüklüğü	*100*
Çaprazlama oranı	*0.5*
Ölçekleme faktörü	*0.4*
İterasyon sayısı	*1000*

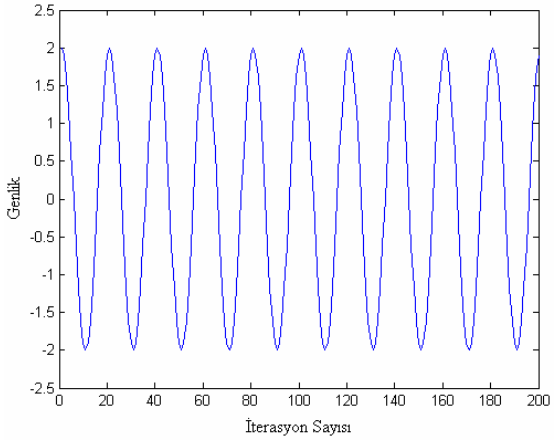
Tablo 4.11. DE/best/2/bin'nin kontrol parametreleri

Popülasyon büyüklüğü	*100*	*İterasyon sayısı*	*1000*
Çaprazlama oranı	*0.5*	*Birleşme faktörü*	*0.2*
Ölçekleme faktörü	*0.4*		

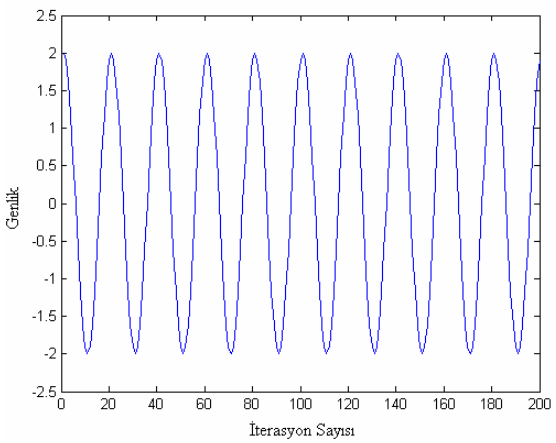
Tablo 4.12. LMS Algoritmasının kontrol parametreleri

Adım büyüklüğü (μ)	*0.01*
İterasyon sayısı	*1000*

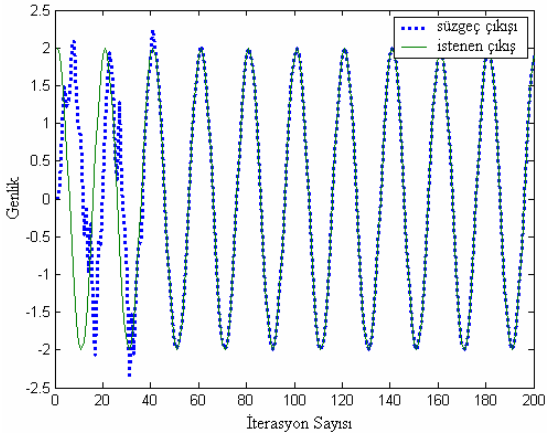
Şekil 4.24 ALT girişi olarak alınan ve frekansı 25 kHz olan sinüzoidal işareti göstermektedir. Şekil 4.25 ise adaptif süzgeç referans girişi olarak alınan ve 25 kHz frekansa sahip kosinüs işaretini vermektedir. Şekil 4.26'da sırayla DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarıyla tasarlanan adaptif süzgeç çıkışı y(n) ile süzgeç çıkışında istenilen işaret d(n) verilmektedir. Şekil 4.27'de ise sırasıyla DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarıyla tasarlanan adaptif süzgecin MSE hata fonksiyonunun iterasyona göre gelişimi verilmektedir.



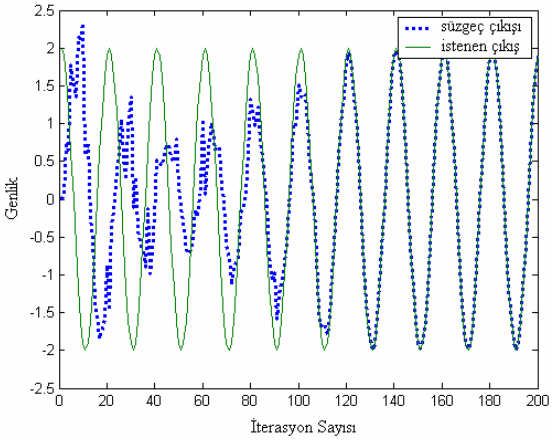
Şekil 4.24. Tek bir sinüzoidal işaretin giriş işareti olduğu durumda ALT giriş işareti



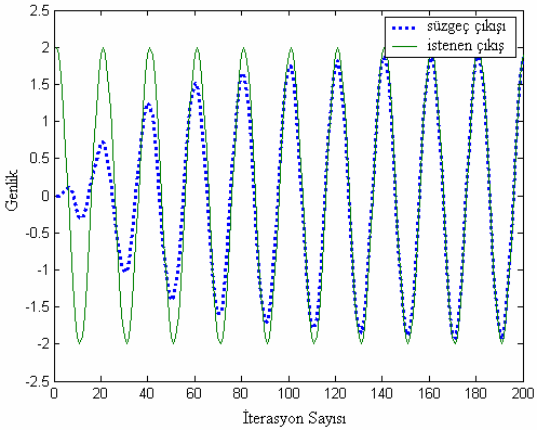
Şekil 4.25. Tek bir sinüzoidal işaretin giriş işareti olduğu durumda ALT referans girişindeki arzu edilen işaret



(a)



(b)

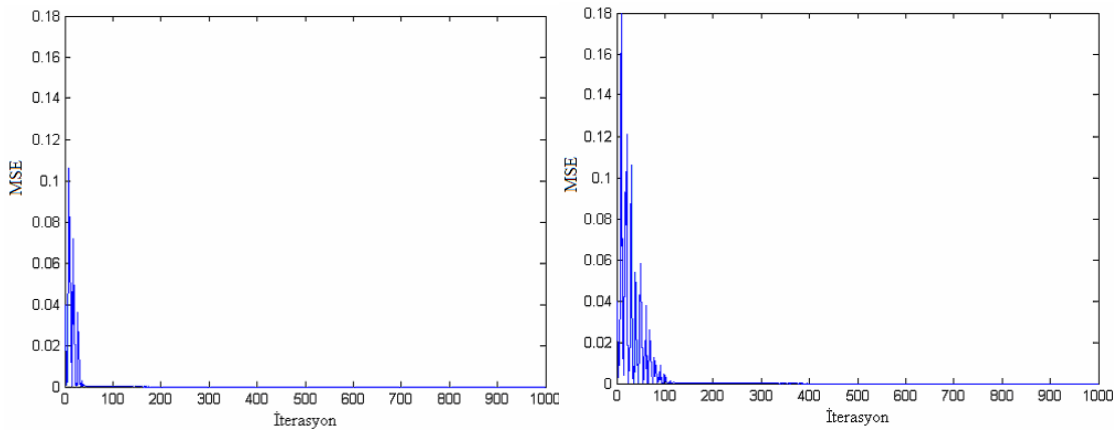


(c)

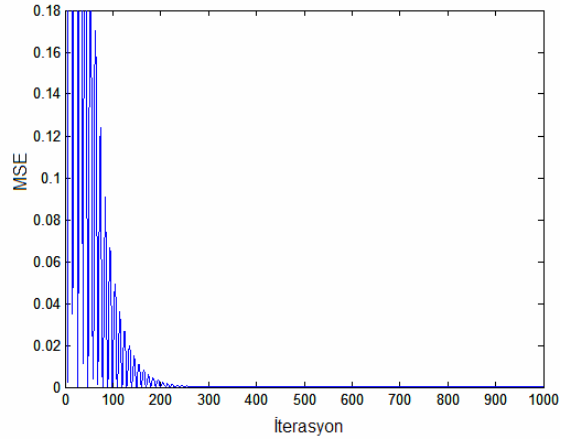
Şekil 4.26. Tek bir sinüzoidal işaretin giriş işareti olduğu durumda adaptif süzgeç çıkışı y(n), arzu edilen çıkış d(n)'in ilk 200 örneği. (a)DE/rand/1/bin, (b) DE/best/2/bin, (c) LMS algoritması

Şekil 4.26 (a)'da giriş işaretinin 25 kHz frekansa sahip olduğu tek bir sinüzoidal işaret olan DE/rand/1/bin algoritmasıyla tasarlanan ALT çıkış işareti y(n) ile tasarım sonucunda elde edilmek istenen referans işaret d(n)'in ilk 200 örneği verilmektedir. Şekil 4.26 (b)'de giriş işaretinin 25 kHz frekansa sahip olduğu tek bir sinüzoidal işaret olan DE/best/2/bin algoritmasıyla elde edilmek istenen referans işaret d(n) ve tasarlanan ALT çıkışı y(n)'nin ilk 200 örneği verilmektedir. Şekil 4.26 (c)'de ise LMS algoritması kullanılarak tasarlanan ALT çıkışı y(n) ile istenen çıkış d(n)'in ilk 200 örneği verilmektedir.

Şekil 4.26'da arzu edilen işaretle DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarıyla elde edilen sonuçlar verilmiştir. Elde edilen sonuçlar, Şekil 4.26 (a)'da DE/rand/1/bin kullanıldığında ALT'nin istenilen işareti 40 iterasyondan sonra, (b)'de DE/best/2/bin kullanıldığında ALT'nin istenilen çıkışı 120 iterasyondan sonra yakaladığını ve (c)'de ise LMS algoritmasıyla tasarlanan adaptif süzgecin istenilen işareti ilk 200 iterasyonda yakalamayadığını göstermektedir.



(a) (b)



(c)

Şekil 4.27. Tek bir sinüzoidal işaretin giriş işareti olduğu durumda hatanın gelişimi. (a)DE/rand/1/bin (b)DE/best/2/bin (c)LMS

Şekil 4.24'de Adaptif süzgecin giriş işareti olan 25 kHz frekansına sahip x(n) işareti gösterilmektedir. Şekil 4.25'de ise adaptif süzgecin referans giriş işareti verilmektedir.

Şekil 4.26 (a), (b), (c)'de sırasıyla DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarıyla elde edilen çıkışların ilk 200 örnekleri istenilen çıkışla karşılaştırılmıştır. Şekil 4.27 (a), (b), (c)'de algoritmalarda kullanılan hata fonksiyonlarının iterasyona göre gelişimleri verilmektedir.

Şekil 4.27 (a), (b), (c)'den de anlaşılacağı gibi DE/rand/1/bin algoritması ilk 100 iterasyonda hatayı sıfıra çok yaklaştırmış ve 100-150 arası iterasyonda hatayı sıfıra çekmiştir. DE/best/2/bin algoritması ise hatayı 100-400 iterasyon arası sıfıra yaklaştırmış ve 400'den sonra da sıfıra çekmeyi başarmıştır. LMS algoritması ise hatayı ancak 200-300 iterasyon arasında sıfıra yaklaştırmış ama hiçbir zaman tam sıfırı sağlayamamıştır. Bu durumda hata fonksiyonlarının iterasyona göre gelişimi dikkate alındığında en iyi başarımı DE/rand/1/bin sergilemiştir. Tablo 4.13'te DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarıyla elde edilen adaptif süzgeç ağırlıkları verilmektedir.

Tablo 4.13. Frekansı 25 kHz olan tek bir işaretin giriş işareti olarak kullanıldığı süzgeç için DE1/rand/1/bin, DE/best/2/bin ve LMS algoritmaları ile elde edilen ağırlıklar

Ağırlıklar	Elde Edilen Ağırlıklar		
	DE/rand/1/bin	DE/best/2/bin	LMS
w_0	1.2202	0.5872	-0.7086
w_1	-0.3375	0.2563	-0.6302
w_2	-1.0429	-0.0478	-0.4903
w_3	0.9092	-0.8676	-0.2988
w_4	-1.9088	-0.9139	-0.0709
w_5	-0.2029	-0.4800	0.1741

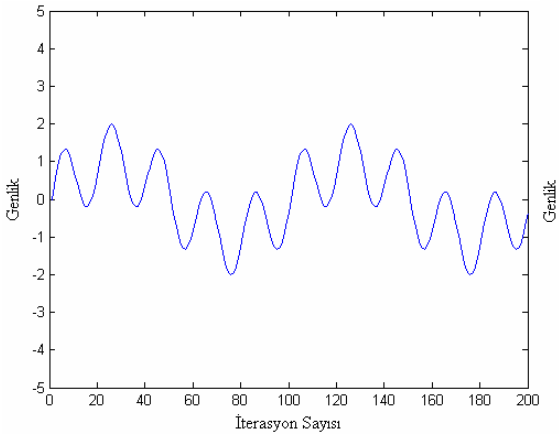
4.3.2. Giriş İşareti Frekansları 25 kHz ve 10 kHz olan iki işaretin toplamı olan Adaptif Süzgeç Tasarımı

Bu çalışmada giriş işareti olarak Eşitlik 4.9'da verilen işaret kullanılmıştır. Anlaşılacağı üzere işaretin bant genişliği önceki çalışmaya göre artmıştır.

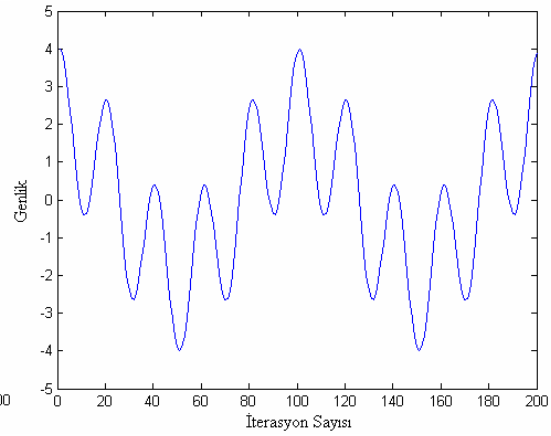
$$x(n) = \sin(2\pi n 25/N) + \sin(2\pi n 10/N) \qquad \text{N=500} \qquad (4.9)$$
$$d(n) = 2\cos(2\pi n 25/N) + 2\cos(2\pi n 10/N), \qquad \text{n=0,1,...,999}$$

Şekil 4.28'de Adaptif süzgecin giriş işareti olan 25 kHz ve 10 kHz frekanslara sahip iki işaretin toplamı olan x(n) işareti gösterilmektedir. Şekil 4.29'da ise adaptif süzgecin referans giriş işareti verilmektedir.

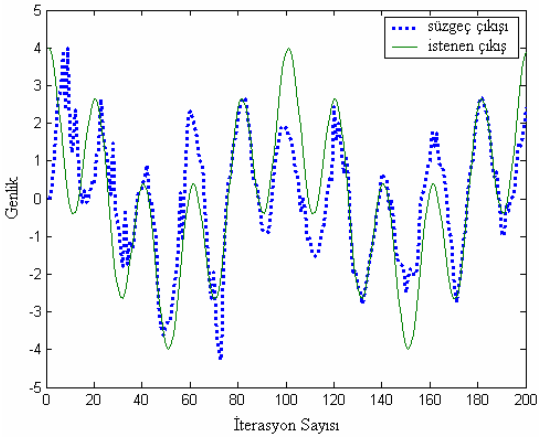
Şekil 4.30'da DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarıyla elde edilen çıkışların ilk 200 örnekleri istenilen çıkışla karşılaştırılmıştır. Şekil 4.31' de ise bu çıkışların son 200 örnekleri istenilen çıkışla karşılaştırılmıştır. Şekil 4.32'de algoritmalarda kullanılan hata fonksiyonlarının iterasyona göre gelişimleri verilmektedir.



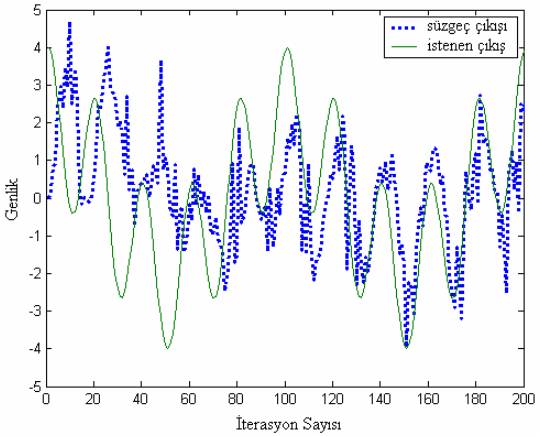
Şekil 4.28. İki sinüzoidal işaret toplamının giriş işareti olduğu durumda ALT giriş işareti



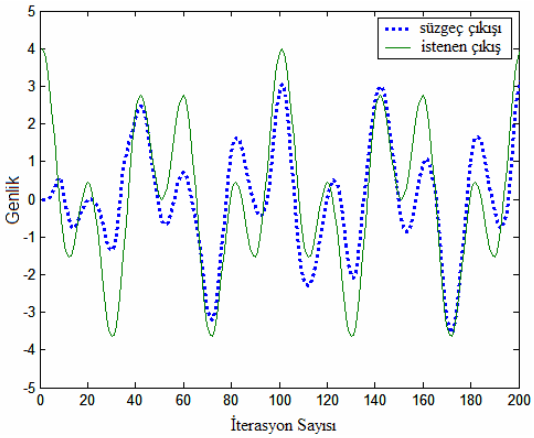
Şekil 4.29. İki sinüzoidal işaret toplamının giriş işareti olduğu durumda ALT referans giriş işareti



(a)



(b)

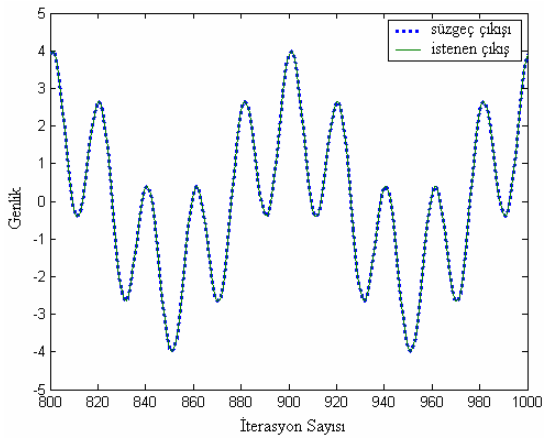


(c)

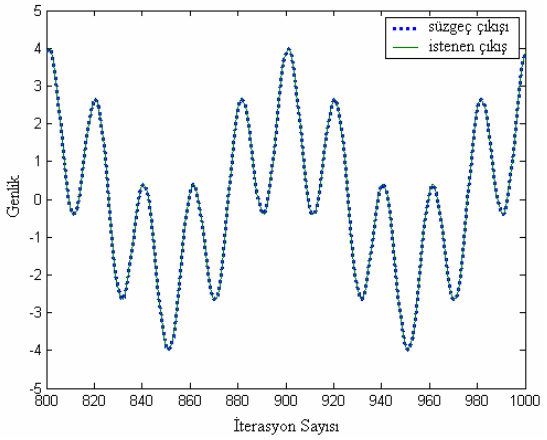
Şekil 4.30. İki sinüzoidal işaret toplamının giriş işareti olduğu durumda adaptif süzgeç çıkışı y(n), arzu edilen işaret d(n)'in ilk 200 örneği. (a)DE/rand/1/bin (b)DE/best/2/bin (c) LMS

Şekil 4.30 (a)'da giriş işaretinin 25 kHz ve 10 kHz frekansa sahip olan iki sinüzoidal işaretin toplamından oluştuğu DE/rand/1/bin algoritmasıyla tasarlanan ALT çıkış işareti y(n) ile tasarım sonucunda elde edilmek istenen referans işaret d(n)'in ilk 200 örneği verilmektedir. Şekil 4.30 (b)'de giriş işaretini 25 kHz ve 10 kHz frekansa sahip iki sinüzoidal işaretin toplamının oluşturduğu DE/best/2/bin algoritmasıyla elde edilmek istenen referans işaret d(n) ve tasarlanan ALT çıkışı y(n)'nin ilk 200 örneği verilmektedir. Şekil 4.30 (c)'de ise LMS algoritması kullanılarak tasarlanan ALT çıkışı y(n) ile istenen çıkış d(n)'in ilk 200 örneği verilmektedir.

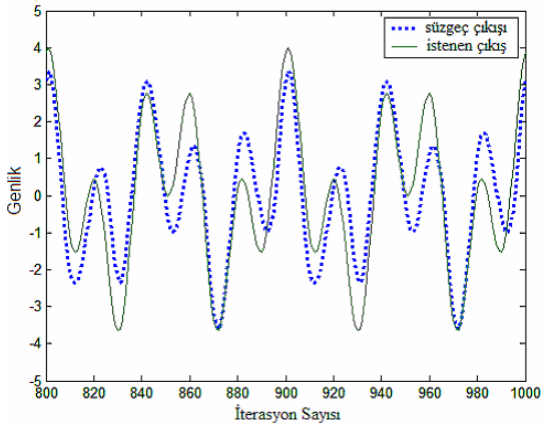
Şekil 4.30'da arzu edilen işaretle DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarıyla elde edilen sonuçlar verilmiştir. Elde edilen sonuçlar, Şekil 4.30 (a), (b) ve (c)'de görülmektedir ki bu üç algoritmayla tasarlanan adaptif süzgeç ilk 200 iterasyonda arzu edilen işareti yakalayamamaktadır. Şekil 4.31'de DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarıyla sağlanan sonuçların son 200 iterasyonu verilmektedir.



(a)



(b)



(c)

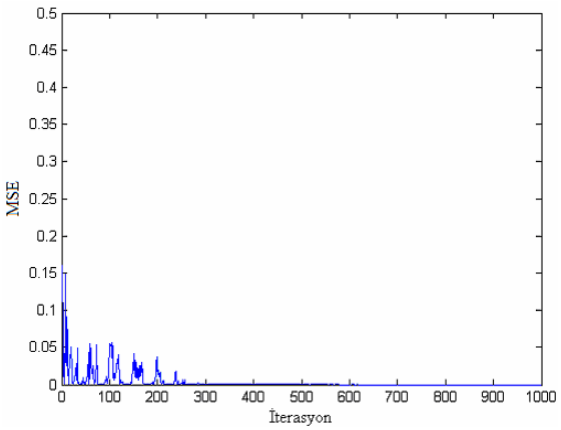
Şekil 4.31. İki sinüzoidal işaretin toplamının giriş işareti olduğu durumda adaptif süzgeç çıkışı y(n), arzu edilen işaret d(n)'in son 200 örneği. (a)DE/rand/1/bin (b)DE/best/2/bin (c) LMS

Şekil 4.31 (a)'da giriş işaretinin 25 kHz ve 10 kHz frekansa sahip olan iki sinüzoidal işaretin toplamından oluştuğu DE/rand/1/bin algoritmasıyla tasarlanan ALT çıkış işareti y(n) ile tasarım sonucunda elde edilmek istenen referans işaret d(n)'in son 200 örneği verilmektedir. Şekil 4.31 (b)'de giriş işaretini 25 kHz ve 10 kHz frekansa sahip iki sinüzoidal işaretin toplamının oluşturduğu DE/best/2/bin algoritmasıyla elde edilmek istenen referans işaret d(n) ve tasarlanan ALT çıkışı y(n)'nin son 200 örneği verilmektedir. Şekil 4.31 (c)'de ise LMS algoritması kullanılarak tasarlanan ALT çıkışı y(n) ile istenen çıkış d(n)'in son 200 örneği verilmektedir.

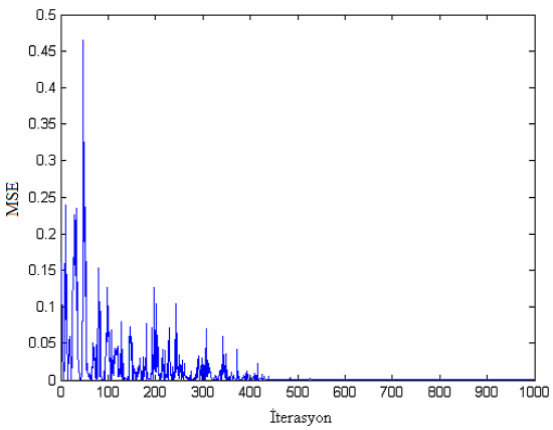
Şekil 4.31'de arzu edilen işaretle DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarıyla elde edilen sonuçlar verilmiştir. Elde edilen sonuçlar, Şekil 4.31 (a) DE/rand/1/bin algoritması kullanılarak giriş işaretinin 25 ve 10 kHz frekanslara sahip iki sinüzoidal işaretten oluşan ALT çıkışındaki işaretin arzu edilen işareti yakaladığını, (b) DE/best/2bin algoritmasıyla tasarlanan süzgeç çıkışının da arzu edilen işareti yakaladığını ve (c) LMS algoritması kullanılarak giriş işaretinin 25 ve 10 kHz frekansa sahip iki işaretin toplamından oluşan adaptif süzgeç çıkışının arzu edilen işareti son 200 iterasyonda bile yakalayamadığını göstermektedir.

Şekil 4.32 (a)'da giriş işaretinin 25 kHz ve 10 kHz frekanslara sahip olan iki sinüzoidal işaretin toplamından oluştuğu DE/rand/1/bin algoritmasıyla tasarlanan ALT'nin

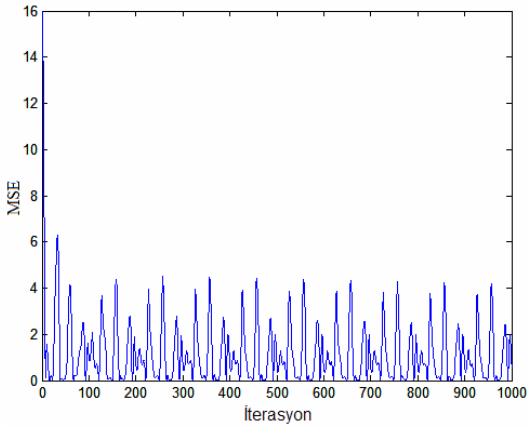
iterasyona göre MSE hata fonksiyonunun gelişimi verilmektedir. Şekil 4.32 (b)'de giriş işaretini 25 kHz ve 10 kHz frekansa sahip iki sinüzoidal işaretin toplamının oluşturduğu DE/best/2/bin algoritmasında MSE hata fonksiyonun iterasyona göre gelişimi gösterilmektedir. Şekil 4.32 (c)'de ise giriş işaretinin 25 kHz ve 10 kHz frekanslara sahip iki sinüzoidal işaretin toplamından oluşan LMS algoritması kullanılarak tasarlanan ALT'nin iterasyona göre MSE hata fonksiyonunun gelişimi verilmektedir.



(a)



(b)



(c)

Şekil 4.32. İki sinüzoidal işaretin toplamı için hatanın gelişimi. (a)DE/rand/1/bin, (b) DE/best/2/bin, (c)LMS

Şekil 4.32 (a), (b), (c)'den de anlaşılacağı gibi DE/rand/1/bin algoritması 250 iterasyondan sonra hatayı sıfıra çok yaklaştırmış ve 600 iterasyondan sonra hatayı sıfıra çekmiştir. DE/best/2/bin algoritması ise hatayı 400-500 iterasyon arası sıfıra yaklaştırmış ama 1000 iterasyonda mutlak sıfıra çekmeyi başaramamıştır. LMS algoritması ise hatayı minimize edememiştir. LMS algoritmasında MSE çok yüksek değerlerde kalmıştır.

Tablo 4.14. DE1/rand/1/bin, DE/best/2/bin ve LMS algoritmaları ile elde edilen ağırlıklar

Ağırlıklar	Elde Edilen Ağırlıklar		
	DE1/rand/1/bin	DE/best/2/bin(*10^5)	LMS
w_0	69.3539	-0.3650	1.1107
w_1	-56.2246	1.0927	-0.0086
w_2	-125.5606	-0.8261	-0.6788
w_3	96.1441	-0.4991	-0.9252
w_4	94.4525	0.9301	-0.8171
w_5	-78.4349	-0.3330	-0.4606

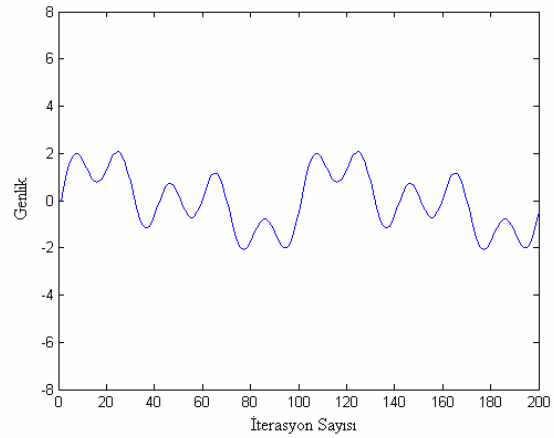
Tablo 4.14'te DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarıyla elde edilen adaptif süzgeç ağırlıkları verilmektedir.

4.3.3. Giriş işareti frekansları 25 kHz, 10 kHz ve 5 kHz olan üç işaretin toplamı olan ALT Tasarımı

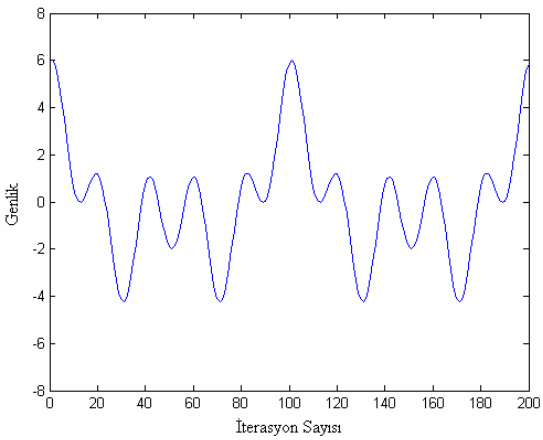
Bu çalışmada giriş işaretleri olarak Eşitlik 4.10'da verilen işaretler kullanılmıştır. Anlaşılacağı üzere işaretin bant genişliği önceki çalışmaya göre artmıştır.

$$x(n) = \sin(2\pi n25/N) + \sin(2\pi n10/N) + \sin(2\pi n5/N) \qquad (4.10)$$
$$d(n) = 2\cos(2\pi n25/N) + 2\cos(2\pi n10/N) + 2\cos(2\pi n5/N)$$

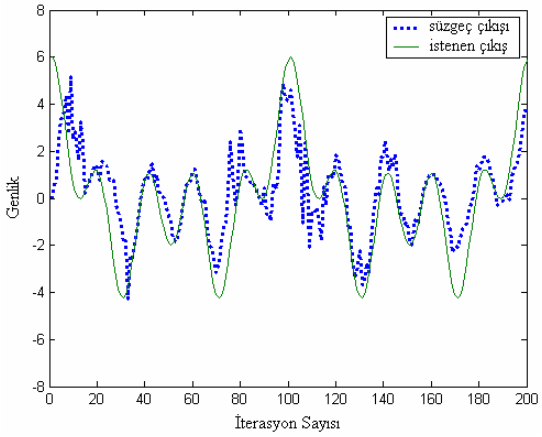
Şekil 4.33'de adaptif süzgecin giriş işareti olan 25 kHz, 10 kHz ve 5 kHz frekanslara sahip üç işaretin toplamı olan x(n) işareti gösterilmektedir. Şekil 4.34'de ise adaptif süzgecin referans giriş işareti verilmektedir.



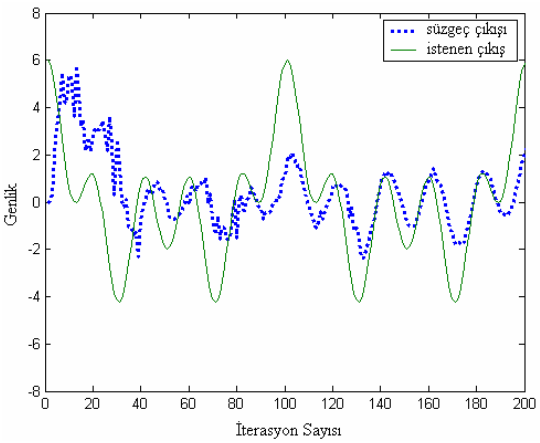
Şekil 4.33. Üç sinüzoidal işaretin giriş işareti olduğu durumdaki adaptif süzgeç giriş işareti



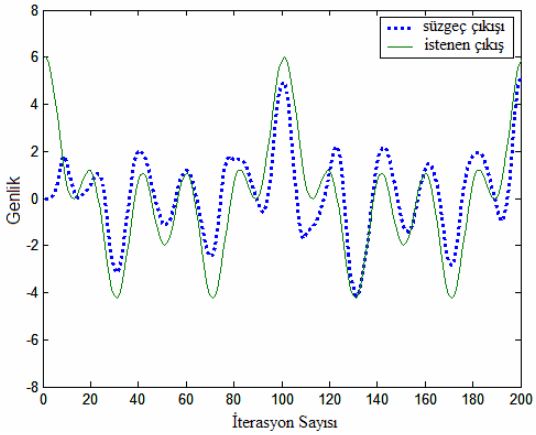
Şekil 4.34. Üç sinüzoidal işaretin giriş işareti olduğu durumdaki adaptif Süzgecin referans giriş işareti



(a)



(b)

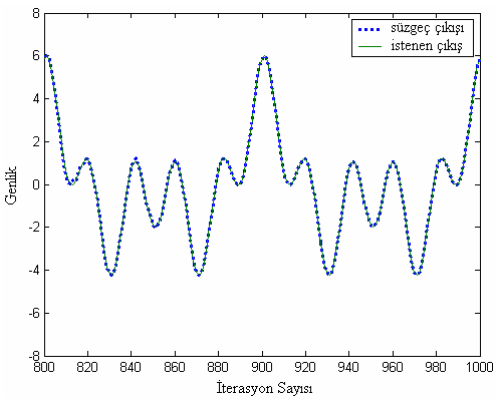


(c)

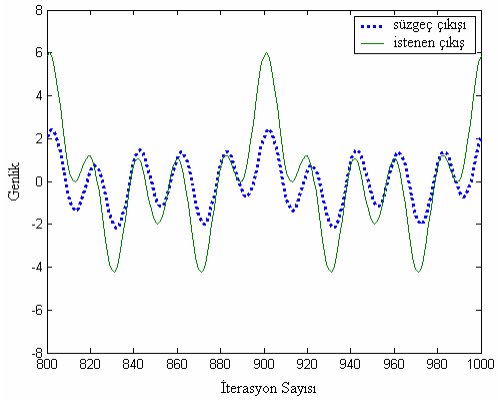
Şekil 4.35. Üç sinüzoidal işaretin toplamının giriş işareti olduğu durumda süzgeç çıkışı y(n), istenilen işaret d(n)'in ilk 200 örneği (a) DE/rand/1/bin, (b) DE/best/2/bin, (c) LMS

Şekil 4.35 (a)'da giriş işaretinin 25 kHz, 10 kHz ve 5 kHz frekanslara sahip olan üç sinüzoidal işaretin toplamından oluştuğu DE/rand/1/bin algoritmasıyla tasarlanan ALT çıkış işareti y(n) ile tasarım sonucunda elde edilmek istenen referans işaret d(n)'in ilk 200 örneği verilmektedir. Şekil 4.35 (b)'de giriş işaretini 25 kHz, 10 kHz ve 5 kHz frekanslara sahip üç sinüzoidal işaretin toplamının oluşturduğu DE/best/2/bin algoritmasıyla elde edilmek istenen referans işaret d(n) ve tasarlanan ALT çıkışı y(n)'nin ilk 200 örneği verilmektedir. Şekil 4.35 (c)'de ise LMS algoritması kullanılarak tasarlanan ALT çıkışı y(n) ile istenen çıkış d(n)'in ilk 200 örneği verilmektedir.

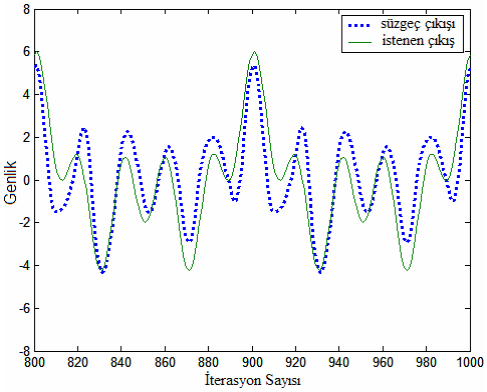
Şekil 4.35'te arzu edilen işaretle DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarıyla elde edilen sonuçlar verilmiştir. Elde edilen sonuçlar, Şekil 4.35 (a), (b) ve (c)'de görülmektedir ki bu üç algoritmayla tasarlanan adaptif süzgeç ilk 200 iterasyonda arzu edilen işareti yakalayamamıştır. Şekil 4.36'da DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarıyla sağlanan sonuçların son 200 iterasyonu verilmektedir.



(a)



(b)



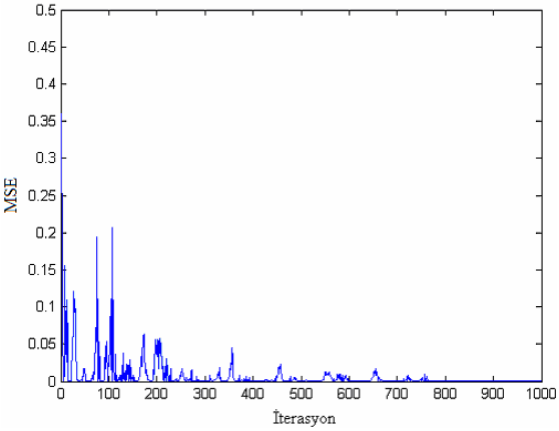
(c)

Şekil 4.36. Üç sinüzoidal işaretin toplamının giriş işareti olduğu durumda süzgeç çıkışı y(n), istenilen işaret d(n)'in son 200 örneği, (a) DE/rand/1/bin (b) DE/best/2/bin (c) LMS

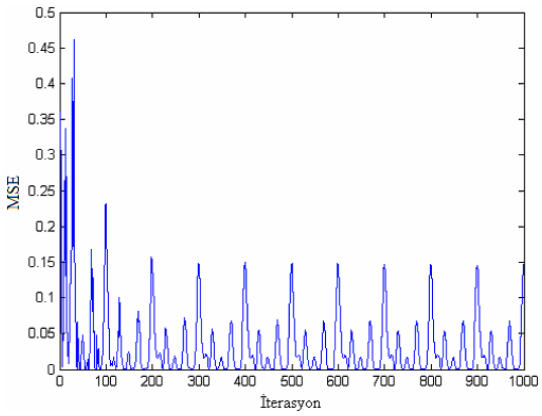
Şekil 4.36 (a)'da giriş işaretinin 25 kHz, 10 kHz ve 5 kHz frekanslarına sahip olan üç sinüzoidal işaretin toplamından oluştuğu DE/rand/1/bin algoritmasıyla tasarlanan ALT çıkış işareti y(n) ile tasarım sonucunda elde edilmek istenen referans işaret d(n)'in son 200 örneği verilmektedir. Şekil 4.36 (b)'de giriş işaretini 25 kHz, 10 kHz ve 5 kHz frekanslarına sahip üç sinüzoidal işaretin toplamının oluşturduğu DE/best/2/bin algoritmasıyla elde edilmek istenen referans işaret d(n) ve tasarlanan ALT çıkışı y(n)'nin son 200 örneği verilmektedir. Şekil 4.36 (c)'de ise LMS algoritması kullanılarak tasarlanan ALT çıkışı y(n) ile istenen çıkış d(n)'in son 200 örneği verilmektedir.

Şekil 4.36'da arzu edilen işaretle DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarıyla elde edilen sonuçlar verilmiştir. Elde edilen sonuçlar, Şekil 4.36 (a) DE/rand/1/bin algoritması kullanılarak giriş işaretinin 25 kHz, 10 kHz ve 5 kHz frekanslara sahip üç sinüzoidal işaretten oluşan ALT çıkışındaki işaretin arzu edilen işareti yakaladığını, (b) DE/best/2bin algoritmasıyla tasarlanan süzgeç çıkışının da arzu edilen işareti yakalayamadığını ve (c) LMS algoritması kullanılarak giriş işaretinin 25, 10, 5 kHz frekanslarına sahip üç işaretin toplamından oluşan adaptif süzgeç çıkışının arzu edilen işareti son 200 iterasyonda bile yakalayamadığını göstermektedir.

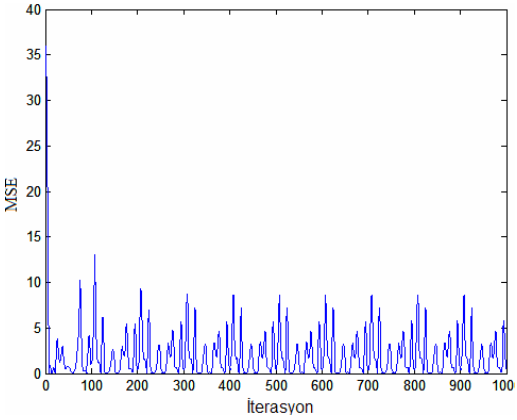
Şekil 4.37 (a)'da giriş işaretinin 25 kHz, 10 kHz ve 5 kHz frekanslara sahip olan iki sinüzoidal işaretin toplamından oluştuğu DE/rand/1/bin algoritmasıyla tasarlanan ALT'nin iterasyona göre MSE hata fonksiyonunun gelişimi verilmektedir. Şekil 4.37 (b)'de giriş işaretini 25 kHz, 10 kHz ve 5 kHz frekanslarına sahip üç sinüzoidal işaretin toplamının oluşturduğu DE/best/2/bin algoritmasında MSE hata fonksiyonun iterasyona göre gelişimi gösterilmektedir. Şekil 4.37 (c)'de ise giriş işaretinin 25 kHz, 10 kHz ve 5 kHz frekanslara sahip üç sinüzoidal işaretin toplamından oluşan LMS algoritması kullanılarak tasarlanan ALT'nin iterasyona göre MSE hata fonksiyonunun gelişimi verilmektedir.



(a)



(b)



(c)

Şekil 4.37. Üç sinüzoidal işaretin toplamı için hatanın gelişimi.
(a) DE/rand/1/bin (b) DE/best/2/bin (c) LMS

DE/best/2/bin algoritmasının ve LMS algoritmasının işaretin bant genişliği arttıkça başarımlarının da düştüğü görülmüştür. Şekil 4.37 (a), (b), (c)'den de anlaşılacağı gibi DE/rand/1/bin algoritması 750 iterasyondan sonra hatayı sıfıra çok yaklaştırmış fakat mutlak sıfırı yakalayamamıştır. DE/best/2/bin algoritması ve LMS algoritması ise hatayı minimize edememiştir. LMS algoritmasının MSE hata değeri çok yüksek değerler almıştır. Tablo 4.15'te üç sinüzoidal işaretin toplamından oluşan giriş işaretinin olduğu durumda DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarıyla elde edilen adaptif süzgeç ağırlıkları verilmektedir.

Tablo 4.15. Üç sinüzoidal işaret toplamının giriş işareti olduğu durumda DE1/rand/1/bin, DE/best/2/bin ve LMS algoritmaları ile ağırlıklar

Ağırlıklar	Elde Edilen Ağırlıklar		
	DE1/rand/1/bin(*10^5)	DE/best/2/bin	LMS
w_0	0.1331	1.3661	-1.2527
w_1	-0.6474	0.1621	-0.8106
w_2	1.2755	-0.6727	-0.5713
w_3	-1.2724	-0.4106	-0.3795
w_4	0.6426	-0.3138	-0.0734
w_5	-0.1314	-0.3544	0.4982

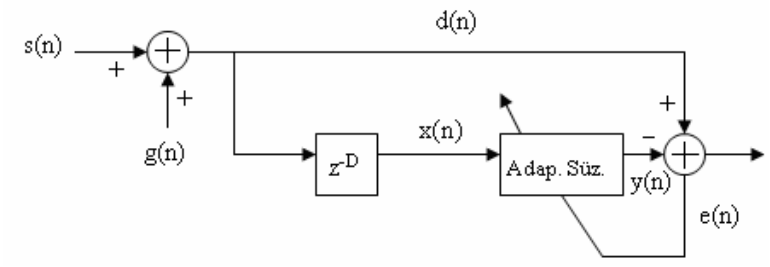
5. BÖLÜM

İŞARET KESTİRİMİNE YÖNELİK ADAPTİF FIR SÜZGEÇ TASARIMI

Bu bölüm iki ana alt bölümden oluşmaktadır. Birinci bölümde işaret kestirmine yönelik adaptif FIR süzgeç tasarım yöntemi açıklanmış ikinci alt bölümde de öncelikle DE algoritması kullanılarak daha sonra da yeni önerilen LMS-DE kombinasyonundan oluşan algoritmayla işaret kestirimine yönelik adaptif FIR süzgeç tasarımı yapılmış ve simülasyon sonuçlarına yer verilmiştir. Simülasyon yapılırken farklı SNR oranlarına sahip gürültü işaretleri farklı bant genişliğine sahip sinüzoidal işaretlere eklenmiş ve önerilen algoritmanın işaret kestirimine yönelik adaptif FIR süzgeç tasarımındaki başarımı ölçülmüştür.

5.1. Yöntem

Bir işaretin geçmişteki değerlerini kullanarak gelecekteki değerlerini bulma problemi işaret kestirimi olarak bilinmektedir. x(n) girişi, gerçek işaret s(n) ile bu işareti bozan gürültü işareti g(n)'nin toplamının gecikmiş şeklidir. Referans giriş ise bozulmuş işarettir. Adaptif süzgecin başarılı olabilmesi için, adaptif süzgeç girişi x(n) ile referans giriş d(n) arasında bir korelasyon olmalıdır. Şekil 5.1 işaret kestiriminin genel bir gösterimini vermektedir. Şekil 5.1'de y(n) çıkış işaretini ve e(n) de arzu edilen işaretle adaptif süzgeç çıkışı arasındaki farkı temsil etmektedir.



Şekil 5.1. İşaret kestirimi için genel bir gösterim

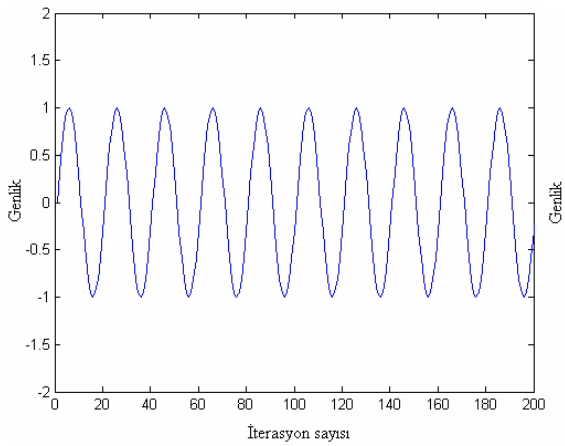
Bu çalışmada öncelikle DE algoritmasıyla işaret kestirimine yönelik adaptif FIR süzgeç tasarlanmış ve sonuçları değerlendirilmiştir. Aynı zamanda gürültülü ortamlarda işaret kestiriminin başarısı ölçülmüştür.

5.2. Simülasyon Sonuçları

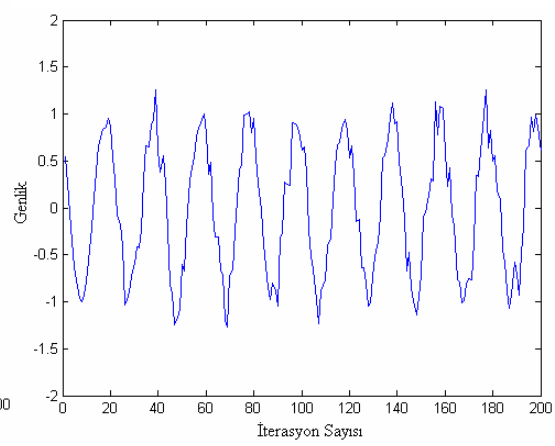
İlk olarak 25 kHz frekansında, 500 kHz'lik örnekleme frekansı olan ve SNR=13 dB oranına sahip bir gürültüyle bozulmuş sinüzoidal bir işaretin kestirimi yapılmaktadır. Adaptif FIR süzgeç uzunluğu 30 olarak alınmaktadır. Önceki bölümde DE/rand/1/bin'in adaptif FIR süzgeç tasarımındaki başarımının DE/best/2/bin'in başarımından daha iyi olduğu görülmüştü. Bu nedenle işaret kestiriminde temel olması açısından burada DE algoritması olarak sadece DE/rand/1/bin algoritması kullanılmaktadır. Algoritmada kullanılan parametreler Tablo 5.1'de verilmektedir. DE algoritması kullanılarak tasarlanan adaptif süzgeç ağırlıkları Tablo 5.2'de verilmektedir.

Tablo 5.1. DE algoritmasında kullanılan kontrol parametreleri

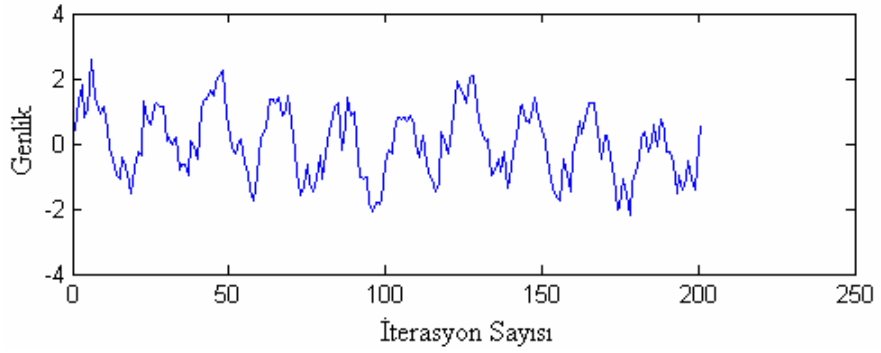
Popülasyon büyüklüğü (NP)	*100*	*Ölçekleme faktörü (F)*	*0.2*
Çaprazlama oranı (CR)	*0.5*	*İterasyon sayısı (Gmax)*	*1000*



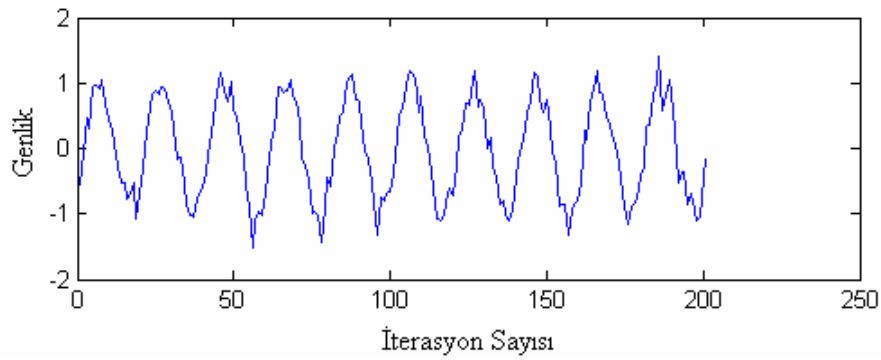
Şekil 5.2. İşaret kestirimi için DE ile tasarlanan adaptif FIR süzgeç çıkışındaki istenilen s(n) işareti



Şekil 5.3. İşaret kestirimi için DE ile tasarlanan adaptif FIR süzgeç giriş işareti

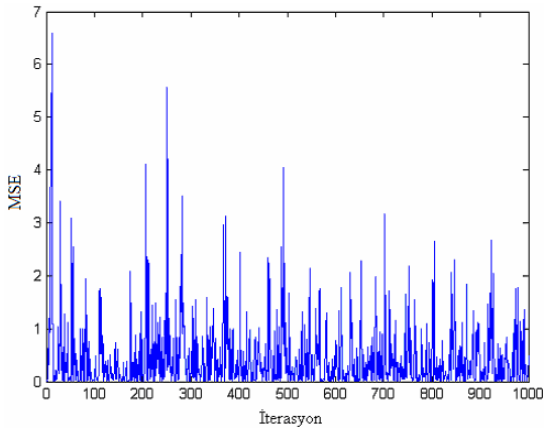


(a)



(b)

Şekil 5.4. İşaret kestirimi için DE ile tasarlanan (a) çıkış işareti y(n), (b) referans işaret d(n)



Şekil 5.5. DE ile işaret kestirimi için iterasyona göre hatanın gelişimi

Tablo 5.2. İşaret kestirimine yönelik adaptif FIR süzgeç tasarımında DE ile elde edilen ağırlıklar

Ağırlıklar	DE Algoritmasıyla Elde Edilen Ağırlıklar				
	s(n)=sin(2*pi*n*25/N)				
w_0	0.8205	w_{10}	0.3933	w_{20}	0.2165
w_1	1.0136	w_{11}	0.3329	w_{21}	0.5551
w_2	-0.1363	w_{12}	-0.0882	w_{22}	0.9553
w_3	0.3834	w_{13}	1.0538	w_{23}	0.7535
w_4	0.0735	w_{14}	1.1868	w_{24}	0.8842
w_5	-0.4032	w_{15}	1.0667	w_{25}	0.7182
w_6	0.0216	w_{16}	0.1641	w_{26}	-0.0878
w_7	0.3290	w_{17}	-0.0204	w_{27}	-0.0507
w_8	0.7702	w_{18}	1.0233	w_{28}	0.7352
w_9	1.5677	w_{19}	0.5688	w_{29}	0.0030

Şekil 5.2'de süzgeç çıkışında elde edilmek istenen işaret verilmektedir. Şekil 5.3'te ise DE algoritması kullanılarak tasarlanan adaptif FIR süzgeç girişi verilmektedir. Şekil 5.4'te süzgeç çıkışında elde edilen işaretle, süzgeç referans giriş işareti verilmektedir. Şekil 5.5'te ise iterasyona göre hata fonksiyonunun gelişimi gösterilmektedir. Şekillerden de anlaşılacağı gibi DE algoritmasının gürültülü ortamda işaret kestirimi uygulamasında başarılı olduğunu söyleyemeyiz. Bu nedenle bu bölümdeki çalışmanın ikinci kısmında LMS algoritmasıyla DE algoritmasının birleşiminden oluşan LMS-DE algoritması olarak adlandırdığımız yeni bir algoritma geliştirilmiş ve yine işaret kestiriminde uygulaması gerçekleştirilmiştir.

LMS algoritması burada DE'nin öncülüğünü yapmaktadır. Yani LMS algoritmasıyla DE algoritmasından önce birkaç örnek koşulmuş ve süzgeç katsayıları her bir iterasyon için ayarlanmıştır. DE algoritması da NP popülasyon büyüklüğünü oluşturacaktır. DE algoritmasının, başlangıç popülasyonunu rasgele oluşturduğu daha önceki bölümlerde anlatılmıştır. Fakat bu algoritmada LMS sonucu elde edilen katsayı vektörü, DE için popülasyonun ilk vektörü olarak kullanılacaktır. Diğer vektörler rasgele belirlenecektir. Algoritmanın diğer adımları DE algoritmasıyla aynı şekilde devam etmektedir. Bu yeni çalışmayla ilgili program Ek-2'de verilmektedir.

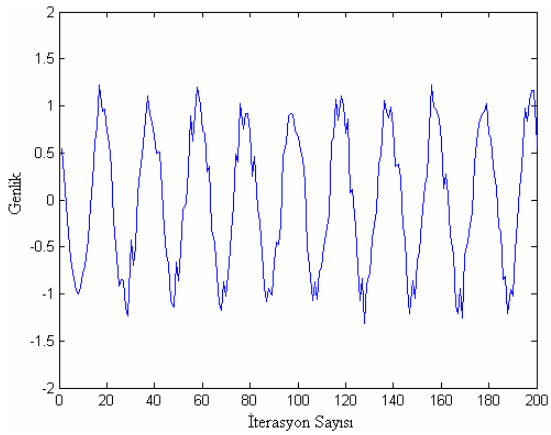
Bu tez çalışmasında LMS-DE Algoritmasıyla işaret kestirimi uygulamasında iki farklı giriş işareti (s(n)) için inceleme gerçekleştirilmiştir.

Örnek 1.

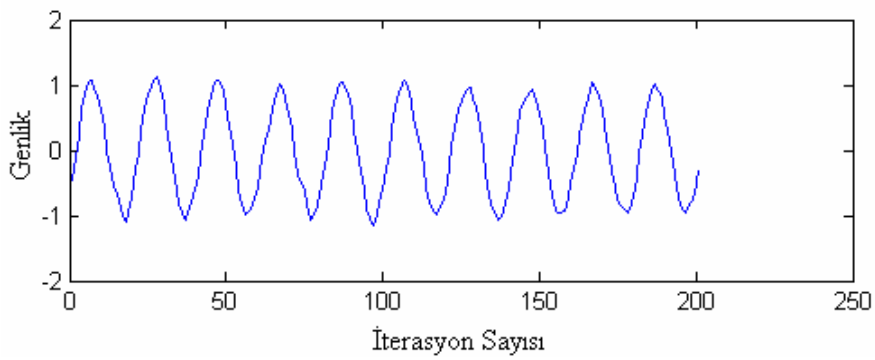
s(n) giriş işareti 25 kHz frekansına ve 500 kHz'lik örnekleme frekansına sahip sinüzoidal bir dalga olarak alınmıştır. d(n) referans giriş işareti ise s(n) işaretinin gürültüye maruz kalmış şeklidir. x(n) olarak tanımlanan adaptif FIR süzgeç girişi ise gürültüye maruz kalmış s(n) işaretinin gecikmeye uğramış versiyonudur. Giriş işareti x(n) ve referans giriş işareti d(n) Eşitlik 5.1'deki gibidir. s(n) işaretine sırasıyla SNR=13 dB, SNR=10 dB ve SNR=8 dB olan g(n) gürültüsü eklenmiştir. Adaptif FIR süzgeç uzunluğu yani ağırlık sayısı 30 olarak seçilmiştir. DE algoritmasında hata fonksiyonu olarak MSE hata fonksiyonu kullanılmıştır. Tablo 5.1'deki DE'nin kontrol parametreleri bu çalışmada da kullanılmıştır. LMS algoritmasının μ adım büyüklüğü değeri 0.01 ve iterasyon sayısı da 1000 olarak alınmıştır.

$$\begin{aligned} s(n) &= \sin(2\pi n 25/N) \\ d(n) &= s(n) + g(n) \\ x(n) &= d(n-D) \end{aligned} \tag{5.1}$$

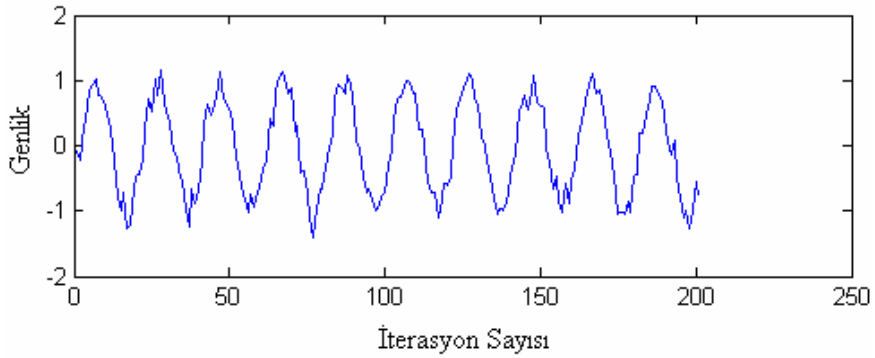
Şekil 5.6, Şekil 5.7 ve Şekil 5.8'de SNR=13 dB hata oranına sahip gürültüye maruz kalmış işaretin kestirimiyle ilgili simülasyon sonuçları verilmektedir. Gürültüye maruz kalmadan önceki sinüzoidal işaret Şekil 5.2'de gösterilmektedir. Şekil 5.6 gürültüye maruz kalmış ve gecikmeye uğramış işareti göstermektedir. Şekil 5.7'de ise adaptif süzgeç çıkışında elde edilen işaretle, gürültüye uğramış ve adaptif süzgecin referans girişine uygulanmış işaret gösterilmektedir. Şekil 5.8 iterasyona göre hatanın gelişimini göstermektedir.



Şekil 5.6. SNR=13 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç giriş işareti, x(n).

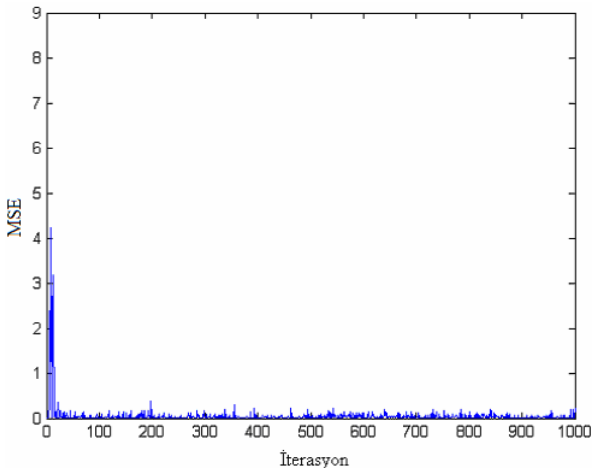


(a)



(b)

Şekil 5.7. SNR=13 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin çıkış işareti y(n), referans işaret d(n).

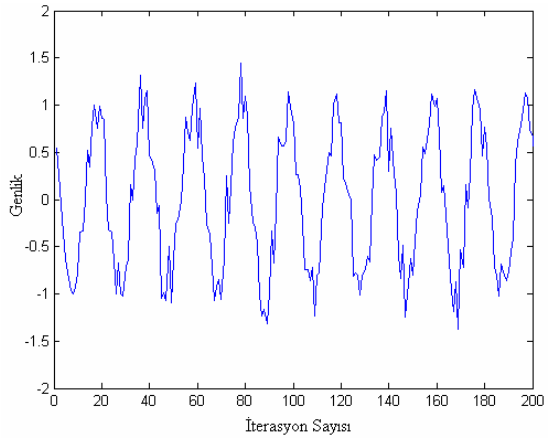


Şekil 5.8. SNR=13 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç için hata

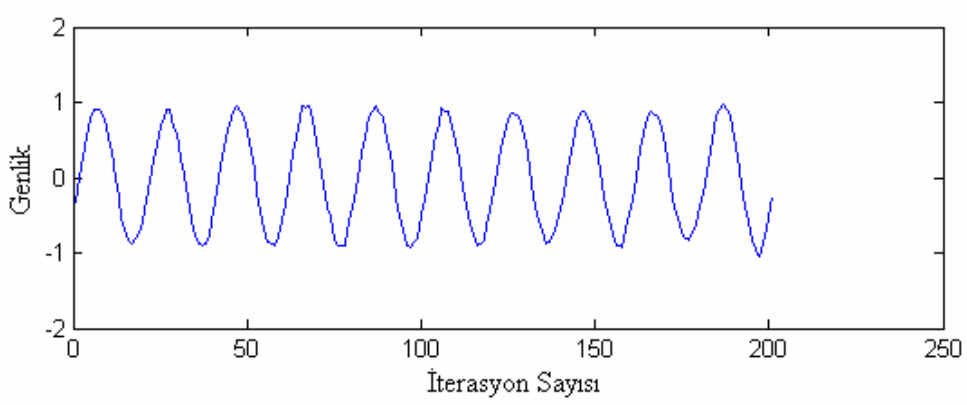
Şekil 5.6, Şekil 5.7 ve Şekil 5.8'den de görüldüğü gibi SNR=13 dB'lik gürültüye maruz kalan işaret kestirilmiştir. SNR=13 dB için LMS-DE algoritması kullanılarak tasarlanan adaptif FIR süzgeç katsayıları Tablo 5.3'de verilmektedir.

Şekil 5.9, Şekil 5.10 ve Şekil 5.11'de SNR=10 dB'lik gürültüye maruz kalmış işaretin kestirimiyle ilgili simülasyon sonuçları verilmektedir. Gürültüye maruz kalmadan önceki sinüzoidal işaret Şekil 5.2'de gösterilmektedir. Şekil 5.9'da gürültüye maruz kalmış ve gecikmeye uğramış işaretin son 200 örneği gösterilmektedir. Şekil 5.10'da ise adaptif süzgeç çıkışında elde edilen işaretle gürültüye uğramış ve adaptif süzgecin referans girişine uygulanmış işaret gösterilmektedir. Şekil 5.11 iterasyona göre hatanın gelişimini göstermektedir.

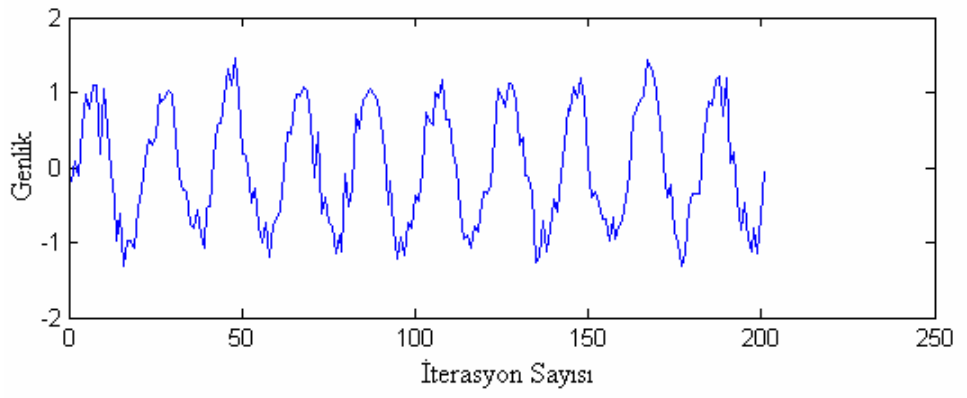
Şekil 5.9, Şekil 5.10 ve Şekil 5.11'den de görüleceği gibi SNR=10 dB'lik gürültüye maruz kalan işaret kestirilmiştir. SNR=10 dB için LMS-DE algoritması kullanılarak tasarlanan adaptif FIR süzgeç katsayıları Tablo 5.3'de verilmektedir.



Şekil 5.9. SNR=10 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin x(n) giriş işareti.

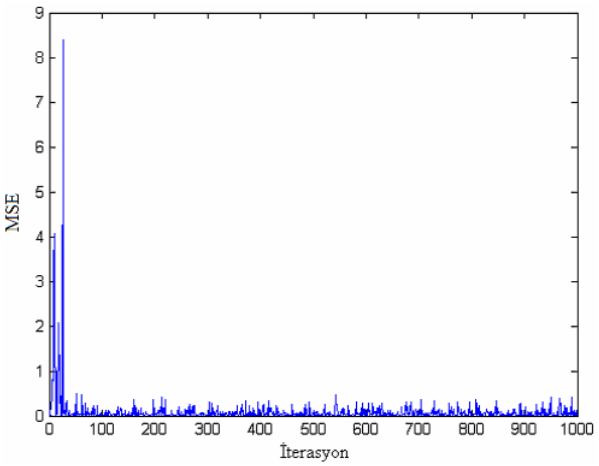


(a)



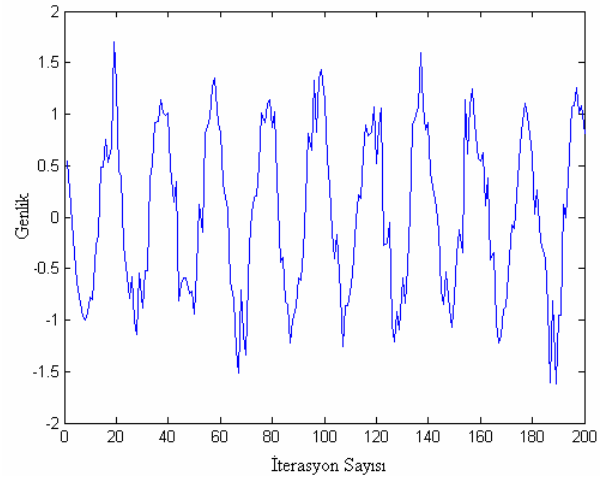
(b)

Şekil 5.10. SNR=10 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin (a) y(n) çıkış işareti, (b) d(n) referans giriş işareti.

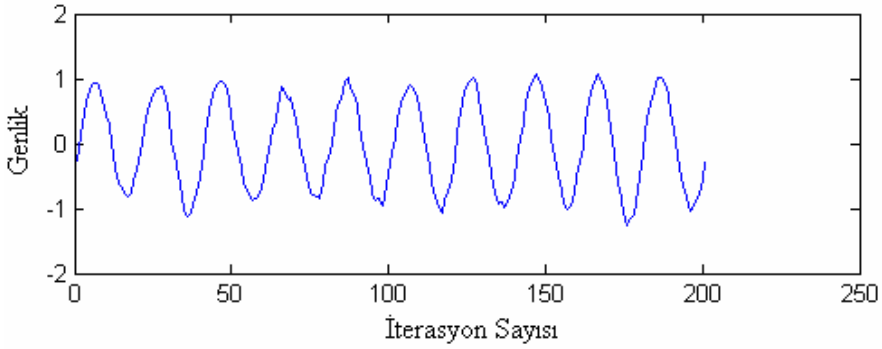


Şekil 5.11. SNR=10 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç için hata.

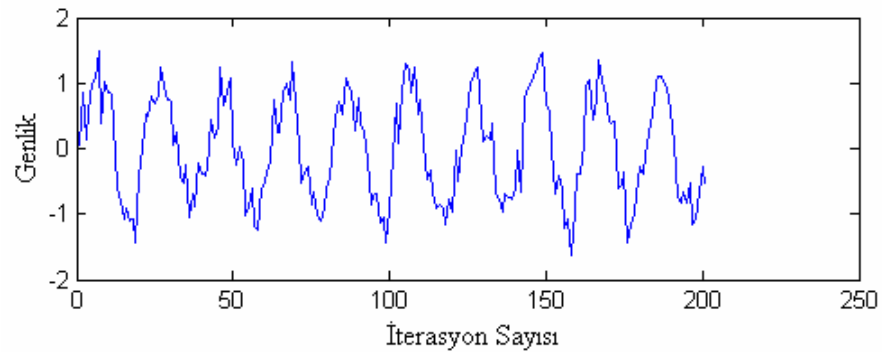
Şekil 5.12, Şekil 5.13 ve Şekil 5.14'te SNR=8 dB'lik gürültüye maruz kalmış işaretin kestirimiyle ilgili simülasyon sonuçları verilmektedir. Gürültüye maruz kalmadan önceki sinüzoidal işaret Şekil 5.2'de gösterilmektedir. Şekil 5.12'de gürültüye maruz kalmış ve gecikmeye uğramış işaretin son 200 örneği gösterilmektedir. Şekil 5.13'te ise adaptif süzgeç çıkışında elde edilen işaretle gürültüye uğramış ve adaptif süzgecin referans girişine uygulanmış işaret gösterilmektedir. Şekil 5.14 iterasyona göre hatanın gelişimini göstermektedir.



Şekil 5.12. SNR=8 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin x(n) giriş işareti

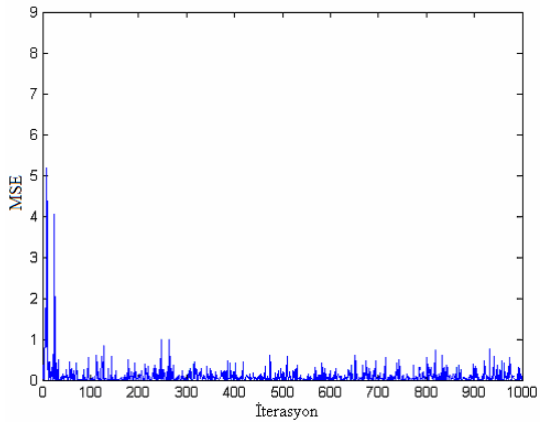


(a)



(b)

Şekil 5.13. SNR=8 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin (a) y(n) çıkış işareti, (b) d(n) referans giriş işareti.



Şekil 5.14. SNR=8 dB için tek bir sinüzoidal işaretin giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç için hatanın gelişimi

Şekil 5.12, Şekil 5.13 ve Şekil 5.14'ten de görüldüğü gibi SNR=8 dB'lik gürültüye maruz kalan işaret kestirilmiştir. SNR=8 dB için LMS-DE algoritması kullanılarak tasarlanan süzgeç katsayıları Tablo 5.3'te verilmektedir.

Tablo 5.3. LMS-DE ile işaret kestiriminde elde edilen FIR süzgeç ağırlıkları

Ağırlıklar					
SNR=13 dB		SNR=10 dB		SNR=8 dB	
w_0	-0.1000	w_0	0.0147	w_0	0.0116
w_1	-0.1311	w_1	-0.0263	w_1	-0.0152
w_2	-0.0798	w_2	-0.0003	w_2	-0.0249
w_3	-0.1354	w_3	-0.0048	w_3	0.0001
w_4	-0.0062	w_4	0.0595	w_4	0.0316
w_5	0.0741	w_5	0.0112	w_5	0.0012
w_6	0.1273	w_6	0.0894	w_6	0.0348
w_7	0.0762	w_7	0.0618	w_7	0.1063
w_8	0.0767	w_8	0.0625	w_8	0.0842
w_9	0.0260	w_9	0.1128	w_9	0.0692
w_{10}	0.0783	w_{10}	0.0605	w_{10}	0.0907
w_{11}	-0.0467	w_{11}	0.0563	w_{11}	0.0422
w_{12}	-0.1195	w_{12}	0.0174	w_{12}	-0.0058
w_{13}	-0.1291	w_{13}	-0.0545	w_{13}	0.0029
w_{14}	-0.0589	w_{14}	-0.0424	w_{14}	-0.0870
w_{15}	-0.0953	w_{15}	-0.0308	w_{15}	-0.1121
w_{16}	-0.0072	w_{16}	-0.0750	w_{16}	-0.0767
w_{17}	-0.0106	w_{17}	-0.0820	w_{17}	-0.0673
w_{18}	-0.0328	w_{18}	-0.0893	w_{18}	-0.0812
w_{19}	-0.0535	w_{19}	-0.0604	w_{19}	-0.1518
w_{20}	-0.2305	w_{20}	-0.0161	w_{20}	-0.1159
w_{21}	-0.1888	w_{21}	0.0245	w_{21}	-0.0388
w_{22}	-0.0443	w_{22}	-0.0258	w_{22}	-0.0304
w_{23}	0.0640	w_{23}	0.0151	w_{23}	-0.0399
w_{24}	0.0764	w_{24}	0.0230	w_{24}	0.0325
w_{25}	-0.1000	w_{25}	0.0338	w_{25}	0.0116
w_{26}	-0.1311	w_{26}	0.0365	w_{26}	-0.0152
w_{27}	-0.0798	w_{27}	0.0191	w_{27}	-0.0249
w_{28}	-0.1354	w_{28}	0.0427	w_{28}	0.0001
w_{29}	-0.0062	w_{29}	0.0153	w_{29}	0.0316

Şekil 5.8, Şekil 5.11 ve Şekil 5.14'deki hata gelişiminden de görüldüğü gibi işaret/gürültü oranı küçüldükçe adaptif FIR süzgecin hata oranı büyümektedir. İşaret kestirimine yönelik adaptif FIR süzgeç işaret/gürültü oranları da dikkate alındığında LMS-DE algoritmasıyla başarıyla tasarlanmıştır.

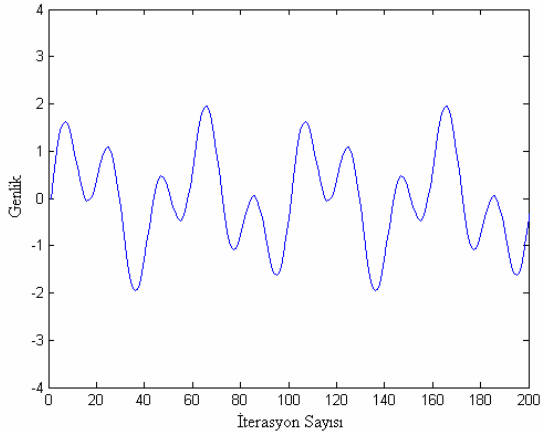
Örnek 2.

İşaret kestirimi için ikinci bir uygulama olarak biri 25 kHz diğeri ise 10 kHz frekansa ve 500 kHz'lik örnekleme frekansına sahip iki işaretin toplamının, s(n) istenilen işaret olarak ele alınması durumunda incelenmiştir. d(n) referans giriş işareti ise s(n) işaretinin gürültüye maruz kalmış şeklidir. x(n) olarak tanımlanan adaptif FIR süzgeç girişi ise gürültüye maruz kalmış s(n) işaretinin gecikmeye uğramış versiyonudur. Eşitlik 5.2'de adaptif FIR süzgecin giriş işaretleri verilmektedir. DE'nin ve LMS algoritmasının kontrol parametreleri Örnek 1'de kullanılan parametrelerle aynı şekilde alınmıştır.

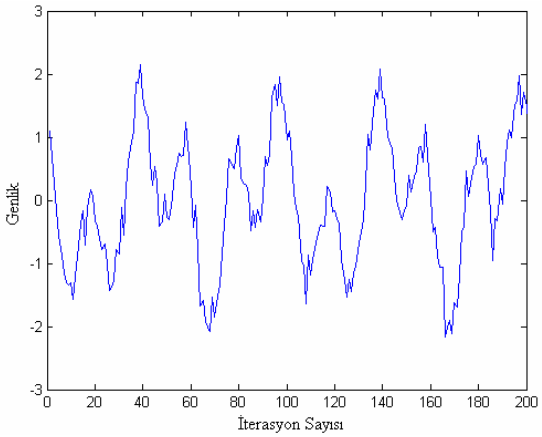
$$s(n) = \sin(2\pi n 25/N) + \sin(2\pi n 10/N) \tag{5.2}$$
$$d(n) = s(n) + g(n)$$
$$x(n) = d(n - D)$$

Burada n=0,1,…,999 olup N=500'dür. Şekil 5.15 ve Şekil 5.2'den de görüldüğü gibi Şekil 5.15'teki sinüzoidal işaret Şekil 2'deki işarete göre daha büyük bir bant genişliğine sahiptir. Şekil 5.2'de ve Şekil 5.15'te bu iki işaretin de ilk 200 örneği verilmektedir. Bu örnek içinde, önceki örnekte olduğu gibi sırasıyla SNR=13 dB, SNR=10 dB ve SNR=8 dB'lik işaret gürültü orana sahip gürültülere maruz kalan işaret kestirimi gerçekleştirilmiştir.

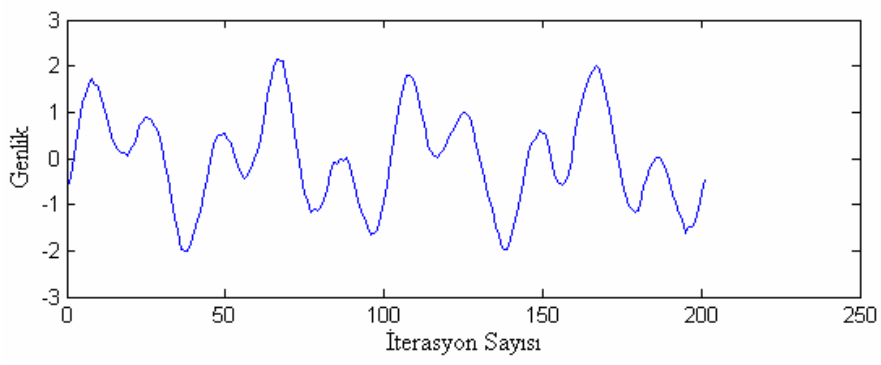
Şekil 5.15'te gürültüye maruz kalmamış istenilen işaret gösterilmektedir. Şekil 5.16, Şekil 5.17 ve Şekil 5.18'de SNR=13 dB hata oranına sahip gürültüye maruz kalmış işaretin kestirimiyle ilgili simülasyon sonuçları verilmektedir. Gürültüye maruz kalmadan önceki sinüzoidal işaret Şekil 5.15'de gösterilmektedir. Şekil 5.16 gürültüye maruz kalmış ve gecikmeye uğramış işareti göstermektedir. Şekil 5.17'de ise adaptif süzgeç çıkışında elde edilen işaretle gürültüye uğramış ve adaptif süzgecin referans girişine uygulanmış işaret gösterilmektedir. Şekil 5.18 iterasyona göre hatanın gelişimini göstermektedir.



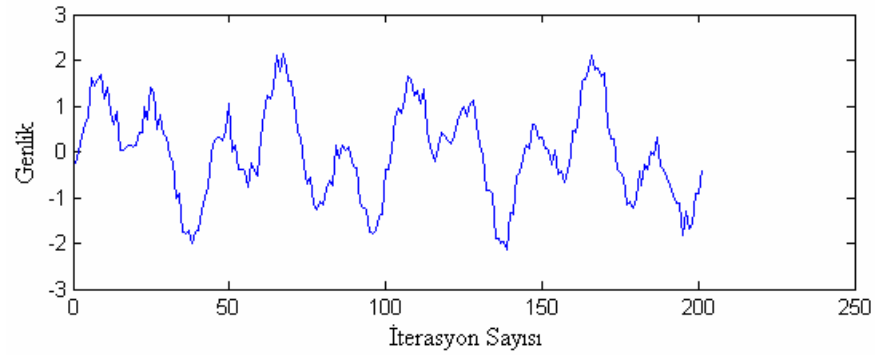
Şekil 5.15. İki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç çıkışında istenilen s(n) işareti.



Şekil 5.16. SNR=13 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç giriş işareti, x(n)

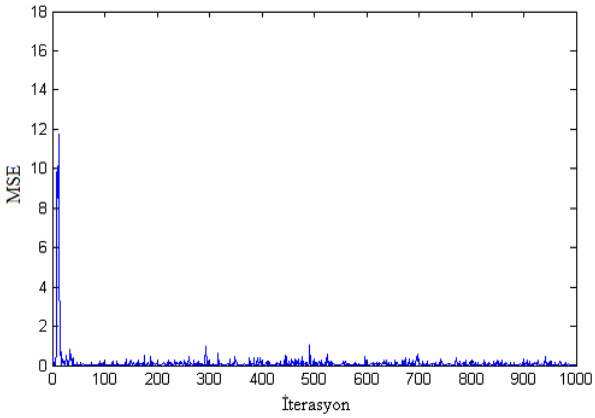


(a)



(b)

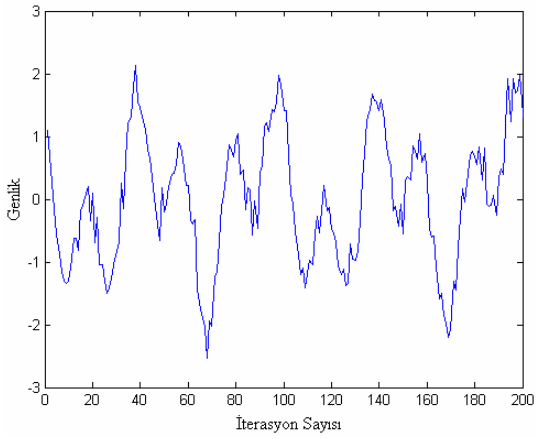
Şekil 5.17. SNR=13 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin, (a) y(n) çıkış işareti, (b) d(n) referans giriş işareti.



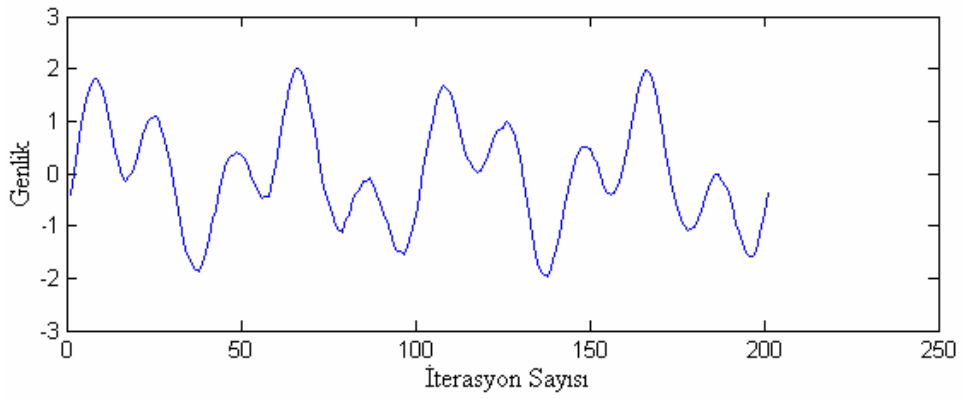
Şekil 5.18. SNR=13 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç için hatanın gelişimi.

Şekil 5.16, Şekil 5.17 ve Şekil 5.18'den de görüldüğü gibi SNR=13 dB'lik gürültüye maruz kalan geniş bantlı işaret de başarıyla kestirilmiştir. SNR=13 dB için LMS-DE algoritması kullanılarak tasarlanan adaptif FIR süzgeç katsayıları Tablo 5.4'te verilmektedir.

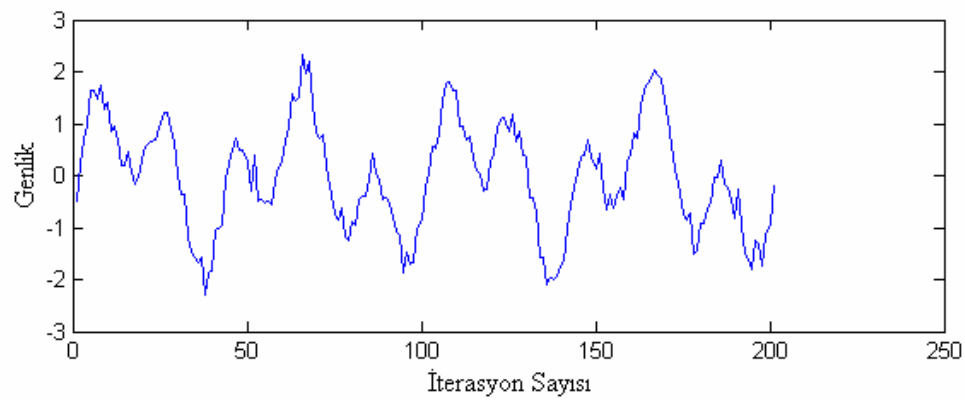
Şekil 5.19, Şekil 5.20 ve Şekil 5.21'de SNR=10 dB'lik gürültüye maruz kalmış geniş bantlı işaretin kestirimiyle ilgili simülasyon sonuçları verilmektedir. Gürültüye maruz kalmadan önceki sinüzoidal işaret Şekil 5.15'te gösterilmektedir. Şekil 5.19'da gürültüye maruz kalmış ve gecikmeye uğramış işaretin son 200 örneği gösterilmektedir. Şekil 5.20'de ise adaptif süzgeç çıkışında elde edilen işaretle gürültüye uğramış ve adaptif süzgecin referans girişine uygulanmış işaret gösterilmektedir. Şekil 5.21 iterasyona göre hatanın gelişimini göstermektedir.



Şekil 5.19. SNR=10 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin x(n) giriş işareti.

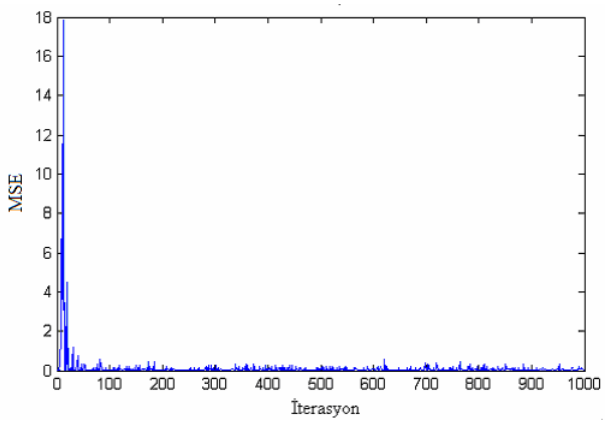


(a)



(b)

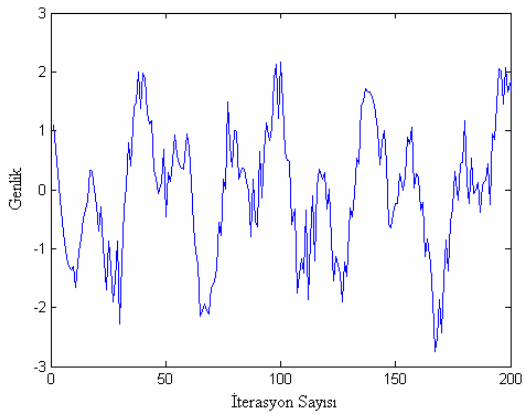
Şekil 5.20. SNR=10 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin, (a) y(n) çıkış işareti, (b) d(n) referans giriş işareti.



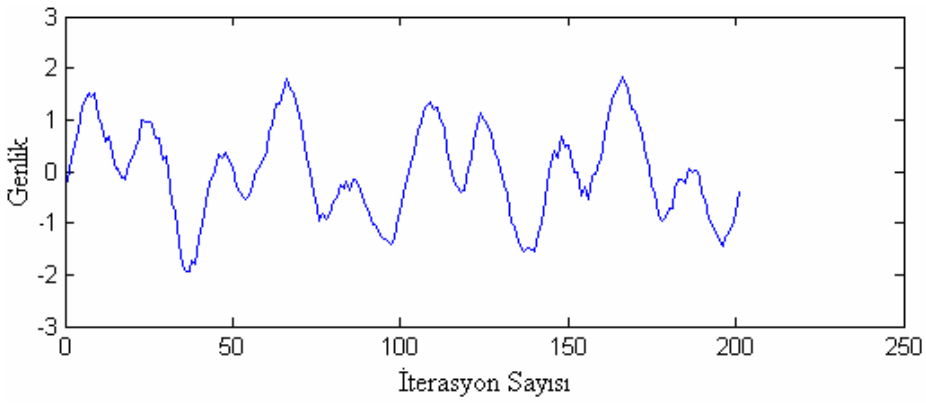
Şekil 5.21. SNR=10 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç için MSE'nin gelişimi.

Şekil 5.19, Şekil 5.20 ve Şekil 5.21'den de görüldüğü gibi SNR=10 dB'lik gürültüye maruz kalan geniş bantlı işaret kestirilmiştir. SNR=10 dB için LMS-DE algoritması kullanılarak tasarlanan adaptif FIR süzgeç katsayıları Tablo 5.4'de verilmektedir.

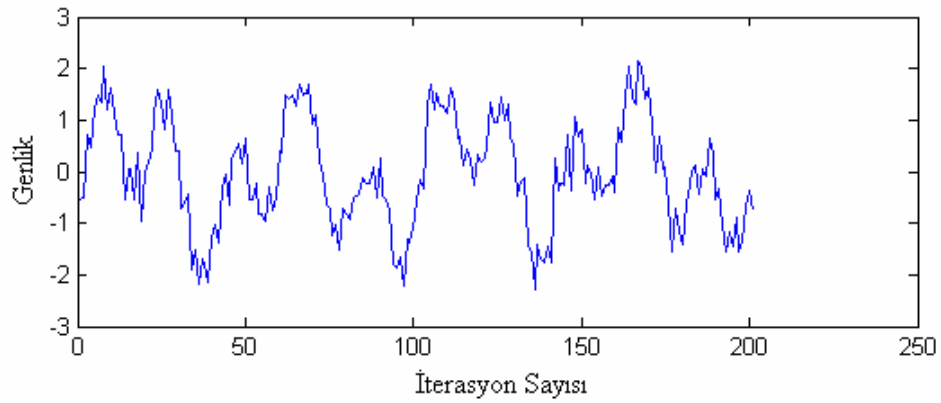
Şekil 5.22, Şekil 5.23 ve Şekil 5.24'de SNR=8 dB'lik gürültüye maruz kalmış işaretin kestirimiyle ilgili simülasyon sonuçları verilmektedir. Gürültüye maruz kalmadan önceki sinüzoidal işaret Şekil 5.15'te gösterilmektedir. Şekil 5.22'de gürültüye maruz kalmış ve gecikmeye uğramış işaretin son 200 örneği gösterilmektedir. Şekil 5.23'de ise adaptif süzgeç çıkışında elde edilen işaretle gürültüye uğramış ve adaptif süzgecin referans girişine uygulanmış işaret gösterilmektedir. Şekil 5.24 iterasyona göre hatanın gelişimini göstermektedir



Şekil 5.22. SNR=8 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin x(n) giriş işareti.

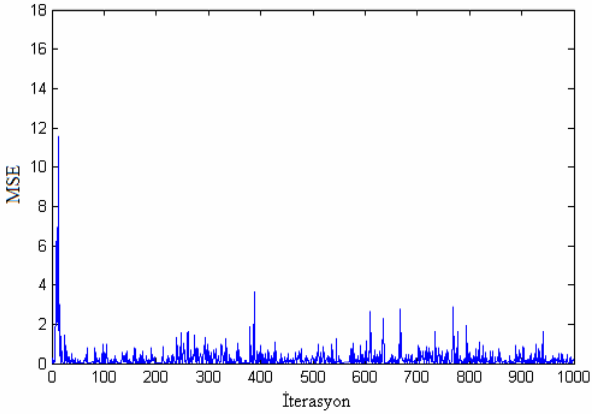


(a)



(b)

Şekil 5.23. SNR=8 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgecin, (a) y(n) çıkış işareti, (b) d(n) referans giriş işareti.



Şekil 5.24. SNR=10 dB için iki sinüzoidal işaret toplamının giriş işareti olarak kullanıldığı LMS-DE algoritmasıyla tasarlanan adaptif FIR süzgeç için hatanın gelişimi.

Şekil 5.22, Şekil 5.23 ve Şekil 5.24'ten de görüldüğü gibi SNR=8 dB'lik gürültüye maruz kalan geniş bantlı işaret kestirilmiştir. SNR=8 dB için LMS-DE algoritması kullanılarak tasarlanan adaptif FIR süzgeç katsayıları Tablo 5.4'te verilmektedir.

Şekil 5.18, Şekil 5.21 ve Şekil 5.24'deki hata gelişiminden de görüldüğü gibi işaret/gürültü oranı küçüldükçe adaptif FIR süzgecin başarımı düşmektedir. İşaret kestirimine yönelik adaptif FIR süzgeç işaret/gürültü oranları da dikkate alındığında LMS-DE algoritmasıyla başarıyla tasarlanmıştır.

Tablo 5.4. Giriş işaretinin iki sinüzoidal işaret olduğu LMS-DE ile işaret kestiriminde elde edilen FIR süzgeç ağırlıkları

Elde Edilen Ağırlıklar					
SNR=13 dB		SNR=10 dB		SNR=8 dB	
w_0	-0.1956	w_0	-0.1262	w_0	-0.0706
w_1	-0.1754	w_1	-0.1291	w_1	-0.0776
w_2	-0.1789	w_2	-0.0700	w_2	-0.1428
w_3	-0.0678	w_3	-0.0125	w_3	-0.0317
w_4	-0.0749	w_4	0.0088	w_4	-0.1552
w_5	0.0133	w_5	0.0735	w_5	0.0300
w_6	0.0833	w_6	0.0933	w_6	0.0444
w_7	0.1054	w_7	0.0929	w_7	0.0496
w_8	0.1661	w_8	0.1184	w_8	0.0887
w_9	0.1324	w_9	0.1645	w_9	0.1162
w_{10}	0.0989	w_{10}	0.0536	w_{10}	0.1669
w_{11}	0.1101	w_{11}	0.0644	w_{11}	0.0831
w_{12}	0.0571	w_{12}	0.0295	w_{12}	0.0747
w_{13}	0.0225	w_{13}	0.0213	w_{13}	0.0292
w_{14}	-0.0019	w_{14}	0.0086	w_{14}	0.0277
w_{15}	-0.0951	w_{15}	0.0180	w_{15}	0.0331
w_{16}	-0.0062	w_{16}	0.0201	w_{16}	-0.0622
w_{17}	-0.0103	w_{17}	-0.0022	w_{17}	-0.1219
w_{18}	-0.0208	w_{18}	0.0121	w_{18}	-0.1112
w_{19}	-0.0100	w_{19}	0.0589	w_{19}	-0.0480
w_{20}	0.0194	w_{20}	0.0395	w_{20}	0.0674
w_{21}	0.0839	w_{21}	0.0439	w_{21}	0.0670
w_{22}	0.1098	w_{22}	0.0702	w_{22}	0.1458
w_{23}	0.1723	w_{23}	0.1209	w_{23}	0.2181
w_{24}	0.2033	w_{24}	0.1211	w_{24}	0.2258
w_{25}	-0.1956	w_{25}	0.1339	w_{25}	-0.0706
w_{26}	-0.1754	w_{26}	0.0781	w_{26}	-0.0776
w_{27}	-0.1789	w_{27}	0.0614	w_{27}	-0.1428
w_{28}	-0.0678	w_{28}	0.0524	w_{28}	-0.0317
w_{29}	-0.0749	w_{29}	0.0180	w_{29}	-0.1552

Şekil 5.6-Şekil 5.24'ten görülmektedir ki LMS-DE algoritması farklı bant genişliğine sahip işaretler kullanıldığında ve farklı SNR oranlarına sahip gürültülerin bozduğu durumlarda bile işaret kestirimini başarıyla gerçekleştirmiştir. Tablo 5.4'te SNR=13 dB, SNR=10 dB ve SNR=8 dB oranlarına sahip gürültülerin bozmuş olduğu işaretlerin kestirilmek istenen işaretler olduğu işaret kestirimine yönelik adaptif FIR süzgecin LMS-DE algoritmasıyla elde edilen ağırlıklıkları verilmektedir.

6. BÖLÜM

SONUÇ

Bu tez çalışmasında, ilk olarak bir yapay zeka optimizasyon algoritması olan DE algoritmasının DE/rand/1/bin ve DE/best/2/bin olarak adlandırılan iki şekli kullanılarak adaptif süzgeç tasarımı gerçekleştirilmiş ve elde edilen sonuçlar değerlendirilmiştir. Yazılım, bütün programlama dillerine uyarlanabilecek basitlikte yapılmıştır. Öncelikle adaptif süzgeç tasarımında yaygın olarak kullanılan LMS algoritmasıyla adaptif süzgeç tasarımı gerçekleştirilerek DE/rand/1/bin ve DE/best/2/bin algoritmalarıyla elde edilmiş tasarımla başarımları karşılaştırılmıştır. DE/rand/1/bin ve DE/best/2/bin'in LMS algoritmasına göre daha yavaş olmasına rağmen çok daha hassas sonuçlar verdiği gözlemlenmiştir. DE/rand/1/bin ile DE/best/2/bin algoritmalarının başarımları değerlendirildiğinde ise DE/rand/1/bin algoritmasının DE/best/2/bin algoritmasına göre adaptif süzgeç tasarımında daha iyi sonuçlar verdiği görülmüştür. ALT'nin, SNR=13 dB, SNR=10 dB ve SNR=8 dB işaret gürültü oranına sahip gürültü işaretleriyle sinüzoidal işaretin toplamından oluşan gürültülü referans giriş ve gürültüsüz referans giriş işaretleri için hem LMS hem de DE/rand/1/bin ve DE/best/2/bin algoritmalarıyla ayrı ayrı tasarımı gerçekleştirilmiştir. Başarımları değerlendirildiğinde DE/rand/1/bin algoritmasının DE/best/2/bin algoritmasına ve LMS algoritmasına göre daha iyi yakınsadığı görülmüştür.

Yukarıda anlatılan çalışmalara ek olarak, MSE, LMS ve MAE hata fonksiyonları, ALT tasarımında LMS algoritmasına ve DE/best/2/bin algoritmasına göre istenilen işarete daha iyi yakınsayan DE/rand/1/bin algoritmasına adaptif süzgeç tasarımı için ayrı ayrı uygulanmıştır. Elde edilen simülasyon sonuçlarından adaptif süzgeç tasarımında MSE hata fonksiyonunun LMS ve MAE hata fonksiyonlarına göre hatayı daha erken minimize ettiği görülmüştür.

Çalışmanın devamında, DE algoritmasının adaptif süzgeç tasarımındaki başarımını ölçmek için farklı bant genişliğine sahip üç sinüzoidal işaret üzerinde çalışılmış ve DE/rand/1/bin, DE/best/2/bin ve LMS algoritmalarının başarımları karşılaştırılmıştır. Simülasyon sonuçlarından, daha büyük bant genişliğine sahip işaretler için adaptif süzgeç tasarımında DE/rand/1/bin algoritmasının DE/best/2/bin algoritmasına ve LMS algoritmasına göre daha iyi başarım sergilediği görülmüştür. DE/best/2/bin algoritması ise LMS algoritmasına göre daha iyi başarım sergilemiştir. İşaret bant genişliği büyüdükçe LMS algoritmasının ve DE/best/2/bin algoritmasının başarımlarının düştüğü ve arzu edilen işareti yakalayamadığı görülmüştür. DE/rand/1/bin algoritması ise daha büyük bant genişliğine sahip işaretlerin olduğu durumlarda bile iyi sonuçlar vermiştir.

Tez çalışmasının son bölümünde, adaptif süzgeç tasarımında ilk defa kullanılan DE algoritması, LMS algoritması ile birlikte kullanılarak yeni bir algoritma geliştirilmiştir. LMS-DE olarak adlandırdığımız bu algoritma işaret kestirimine uygulanmış ve başarımı elde edilen simülasyon sonuçları vasıtasıyla değerlendirilmiştir. Elde edilen sonuçlardan LMS-DE algoritmasının işaret kestiriminde iyi başarım sergilediği görülmüştür. SNR=8 dB, 10 dB ve 13 dB'lik farklı işaret/gürültü oranına sahip gürültü işaretlerinin periyodik işaretleri bozduğu durumlar üzerinde çalışılmış ve DE algoritması kullanılarak tasarlanan adaptif süzgecin, işaret kestirimindeki başarısı ölçülmüştür. Gürültü işaretinin işaret/gürültü oranı düştükçe hata oranının arttığı görülmüştür. Aynı zamanda bant genişliği birbirinden farklı iki işaret, işaret kestirimine yönelik uygulamada kullanılmış ve LMS-DE algoritmasının büyük bant genişliğine sahip işaretlerin olduğu durumlarda da işaret kestiriminde başarılı olduğu görülmüştür. Simülasyon sonuçlarından işaret bant genişliği büyüdükçe algoritmanın başarımının düştüğü görülmüştür.

Sonuç olarak adaptif süzgeç tasarımı için yapılacak yeni çalışmalarda, DE algoritması gürültü giderme, kimliklendirme ve modelleme gibi çeşitli adaptif süzgeç yapılarını tasarlamak için kullanılabilir. Ayrıca farklı yapay zeka optimizasyon algoritmaları kullanılarak da bu tez çalışmasında önerilen tasarımlar gerçekleştirilebileceği gibi bu tasarımlar donanım bazında da gerçekleştirilebilirler.

KAYNAKLAR

1. Chen, C. T., Digital Signal Processing: Spectral Computation and Filter Design, Oxford University Press, New York, 2001.

2. Ertürk, S., Sayısal İşaret İşleme, Kocaeli Üniversitesi, Birsen Yayınevi, Kocaeli.

3. Çetinkaya, B., Genetik Algoritma Kullanarak Optimum Genlik Cevaplı ve Minimum Fazlı Sayısal Süzgeçlerin Tasarımı, Yüksek Lisans Tezi, Erciyes Üniversitesi, 2004.

4. Karaboğa, N., Sayısal Filtre Katsayılarının Genetik Algoritma Kullanılarak Yuvarlatılması, Doktora Tezi, Erciyes Üniversitesi, 1996.

5. Lathi, B.P., Signal Processing and Linear Systems, Berkeley-Cambridge Press, 1998.

6. http://cnx.org/content/m0074/latest/

7. Gorne T, Schneider M: Design of Digital Filters with Evolutionary Algorithms, ICNNGA 93, Ref. No. 155-1, pp. 1, Berlin, 1993.

8. Lohmann R, Gorne T, Schneider M and Warstat M: Digital Systems and Structure Evolution, Prepr. 94th Conv. Audio Eng. Soc., D6-2, Berlin (1993)

9. Adem Kalinli, Nurhan Karaboga: A New Method for Adaptive IIR Filter Design Based on Tabu Search Algorithm, AEÜ, 2004.

10. Storn, R., Price, K.: Differential Evolution - a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization, Vol. 11. Kluwer Academic Publishers (1997) 341 – 359

11. Price, K. V. , : Differential Evolution: a fast and simple numerical optimizer. In: Smith, M., Lee, M., Keller, J., (eds.): Biennial Conference of the North American Fuzzy Information Processing Society, NAFIPS. IEEE Press, New York (1996) 524-527.

12. Derviş Karaboğa, Yapay Zeka Optimizasyon Algoritmaları, Nobel Yayınevi,

İstanbul, 2004.

13. Storn, R.: On the Usage of Differential Evolution for Function Optimization. NAFIPS 1996 Berkeley, 1996 pp. 519-523.

14. Vondraš, J.: Different Evolution Algorithm Application in Optimized Filter Design. In Proceedings of Workshop 2001. Prague: CTU, vol. A, pp. 50-51, 2001.

15. Price, K., Storn, R.: A simple evolution strategy for fast optimization. Dr. Dobb's Journal, 264, 18-24, 1997.

16. Price, K, Storn, R, : http://www.ICSI.Berkeley.edu/~storn/code.html

17. Storn, R.: Differential Evolution Design of an IIR-Filter with Requirements for Magnitude and Group Delay. In: Proceedings IEEE International Conference on Evolutionary Computation ICEC'96. IEEE Press, New York 268 – 273, 1996.

18. Wang, F. S., Jing, C. H. and Tsao, G. T., Fuzzy Decision-Making Problems of Fuel Ethanol Production Using a Genetically Engineered Yeast. Industrial & Engineering Chemistry Research. 37, 3434-3443, 1998.

19. Sastry, K.K.N, L. Behera and I.J. Nagrath, A Self Organizing Fuzzy Controller Design Using Differential Evolution. In Sixth Symposium on Intelligent Systems, IISc Bangalore, 166-177, 1997.

20. Chiou, J. P. And Wang, F. S., Hybrid method of evolutionary algorithm for static and dynamic optimization problems with application to a fed-batch fermentation process. Computers&Chemical Engineering, 23, 1277-1291, 1999.

21. Wang, F. S. And Cheng, W. M., Simultaneous optimization of feding rate and operation parameters for fed-batch fermentation process. Biotechnology Progress, 15(5), 949-952, 1999.

22. Joshi, R., and Sanderson, A. C., Minimal representation multisensor fusion using differential evolution. IEEE Transactions on Systems, Man, and

Cybernetics, Part A 29(1), pp. 63-67, 1999.

23. Babu, B. V., and Sastry, K. K. N., Estimation of heat-transfer parameters in a trickle-bed reactor using differential evolution and orthogonal collocation. Computers&Chemical Engineering, 23, 327-339, 1999.

24. Babu, B. V., and Munawar, S. A., Differential Evolution for the optimal design of heat exchangers. Proceedings of All-India seminar on Chemical Engineering Progress on Resource Development: A Vision 2010 and Beyond, IE (I), Bhubaneswar, India, March 11, 2000.

25. Babu, B. V., and Munawar, S. A., Optimal Design of Shell&Tube Heat Exchanger by Different strategies of Differential Evolution. PreJournal.com-The Faculty Lounge, Article No. 003873, posted on website Journal http://www.Prejournal.com 2001.

26. Babu, B. V., and Singh, R. P., Synthesis&optimization of Heat Integrated Distillation Systems Using Differential Evolution. Proceedings of All-India seminar on Chemical Engineering Progress on Resource Development: A Vision 2010 and Beyond, IE (I), Bhubaneswar, India, March 11, 2000.

27. Babu, B.V. and Angira, R., Optimization of Non-Linear Functions Using Evolutionary Computation. Proceedings of 12th ISME Conference on Mechanical Engineering, Crescent Engineering College, Chennai, January 10-12, Paper No. CT07, pp 153-157, 2001.

28. Babu, B. V., and Angira, R., Optimization of thermal cracker operation using Differential Evolution. Proceedings of International Symposium & 54th Annual Session of IIChE (CHEMCON-2001). Chennai, December 19-22, 2001.

29. Kyprianou, A., Worden, K., Panet, M., Identification of hysteretic systems using the differential evolution algorithm. Journal of Sound and Vibration, Vol. 248, No. 2, 289-314, Nov. 2001.

30. Mayer, D. G., Kinghorn, B. P. and Archer, A. A. Differential evolution – an easy and efficient evolutionary algorithm for model optimization.

Agricultural Systems, 83, pp.315-328. 2005.

31. Cruz, L. L. I, Willigenburg, V. L. G., Straten, V. G., Efficient differential evolution algorithms for multimodal optimal control problems. Applied Soft Computing, Vol. 3, pp. 97-122, 2003.

32. Shiakolas, P. S., Koladiya, D., and Kebrle, J., Optimum Robot Design Based on Task Specifications Using Evolutionary Techniques and Kinematic, Dynamic, and Structural Constraints. International Journal of Inverse Problems in Engineering, Volume 10, Number 4, 359-375, 2002.

33. Karaboga, D. and Okdem, S., A simple and global optimization algorithm for engineering problems: Differential evolution algorithm, Turkish Journal of Electrical Engineering and Computer Sciences, Vol.12, No.1, pp.53-60, 2004.

34. Nurhan Karaboğa, Bahadır Çetinkaya, Efficient Design of Fixed Point Digital FIR Filters by Using Differential Evolution Algorithm. Lecture Notes in Computer Science, vol.3512, 2005.

35. Kukkonen, S., Lampinen, J., An Emprical Study of Control Parameters for Generalized Differential Evolution, International Conference on Evolutionary and Deterministic Methods for Design, Optimization and Control with Applications to Industrial and Societal Problems (Eurogen 2005), Sept 12-14, Munich, Germany, 2005.

36. Rönkkönen, J., Kukkonen, S. and Price, K., Real-Parameter Optimization with differential evolution, IEEE Congress on Evolutionary Computation, Edinburg, Scotland, Vol 1, pp. 506-513, 2-5 September, 2005.

37. Kukkonen, S. and Lampinen, J., GDE3:The third Evolution Step of Generalized Differential Evolution, 2005, p. 443-450, IEEE Congress on Evolutionary Computation (CEC 2005), Edinburgh, United Kingdom, Sept 2-5, 2005.

38. Rönkkönen, J., Kukkonen, S. and Lampinen, J., A Comparison of Differential Evolution and Generelized Generation Gap Model, Journal of

Advanced Computational Intelligence and Intelligent Informatics, vol. 9, nro. 5, p. 549-555, 2005.

39. Liu, J. and Lampinen, J., Trainin RBF Networks Using Differential Evolution with Adaptive Control Parameters, 2005, p. 145-150, In: Randomil Matosek and Pavel Osmera(eds.)(2005), Proceedings of MENDEL 2005, 11th International Conference on Soft Computing, Brno, Czech Republic, June 15-17, 2005.

40. Junghong, L. and Lampinen, J., A Differential Evolution Based Incremental Traning Method for RBF Networks, 2005, p. 881-888, In: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2005, 25-29 June 2005, Washington, DC(USA), Vol. 1, ACM Press, New York(USA).

41. Liu, J. and Lampinen, J., A Fuzzy Adaptive Differential Evolution Algorithm Soft Computing, 2005, Vol. 9, nro. 6, p. 448-462, ISSN 1432-7643, 2005.

42. Liu, J. and Lampinen, J., Approximation by Growing Radial Basis Function Networks Using the Differential-Evolution-Based Algorithm, Journal of Advanced Computational Intelligence and Intelligent Informatics, vol. 9, nro. 5, p. 540-548, 2005.

43. Price, K., Storn, R., Lampinen, J., Differential Evolution: A Practical Approach to Global Optimization, Springer-Verlag, Berlin (Germany), 538 s., 2005.

44. Zielinski, K., Peters, D., Laur, R., Constrained Multi-Objective Optimization Using Differential Evolution, Proceedings of the Third Initernational Conference on Computational Intelligence, Robotics and Autonomuous Systems, Singapore, 2005.

45. Zielinski, K., Peters, D., Laur, R., Run Time Analysis Regarding Stopping Criteria for Differential Evolution and Particle Swarm Optimization, Proceedings of the 1st International Conference on Experiments/Process/System Modelling/Simulation/Optimization, Athens, 2005.

46. Weitkemper, P., Zielinski, K., Kammeyer, K. D., Laur, R., Optimization of Interference Cancellation in Coded CDMA Systems by means of Differential Evolution, 2005.

47. O'neill, M. and Brabazon, A., Grammatical Differential Evolution, 2006 International Conference on Artificial Intelligence (ICAI'06) CSEA Press Las Vegas, Nevada, 2006.

48. Robic T, Filipic B., DEMO: Differential evolution for multiobjective optimization, Evolutionary Multi-Criterion Optimization Lecture Notes In Computer Science, Vo. 3410, p. 520-533, 2005.

49. Braak, C. J. F., Ter, A. Marcov Chain Monte Carlo Version of the Genetic Algorithm Differential Evolution: Easy Bayesian Computing for Real Parameter Spaces. Statistics and Computing 16 (2006), 3, ISSN:1573-1375,2006.

50. Besson, P. ; Vesin, J. ; Popovici, V. ; Kunt, M., Differential Evolution Applied to a Multimodal Information Theoretic Optimization Problem, Lecture Notes in Computer Science, Vol. 3907-2006, p. 505-509, 2006.

51. Sankaran, Sundar G. : On Ways to Improve Adaptive Filter Performance, PhD Theses, Virginia Tech, India, 1999.

52. E. C. Ifeachor and B.W. Jervis, Digital Signal Processing, A Practical Approach, Prentice Hall, 2002.

53. Petrone, Joseph G.: Adaptive Filter Architectures For FPGA Implementation, Master of Science, Theses, Department of Electrical and Computer Engineering, Florida State University, 2004.

54. S. W. Wales: Co-channel Interference Suppression for TDMA Mobile Radio Systems. IEEE Colloquium on Mobile Communications Towards the Year 2000, , pp 9/1-9/6,Oct 17th 1994.

55. Bose, T. , Venkatachalam, A. And Thamvichai, R.: Multiplierless Adaptive Filtering, Digital Signal Processing 12, pp 107-118, 2002.

56. S. Haykin, Adaptive Filter Theory, Prentice Hall, Upper Saddle River, NJ, 2002.

57. Shiakolas, P. S., Koladiya, D., Kebrle, J.: Optimum Robot Design Based on Task Specifications Using Evolutionary Techniques and Kinematic, Dynamic, and Structural Constraints, ASME IMECE 2002, Advances in Robot Dynamics and Controls, No. DSC-9B-2, New Orleans, Nov. 17-22, 2002.

58. Yang, G., Chen, I-Ming: Task based optimization of modular robot configurations: minimized degree-of-freedom approach. Mechanism and Machine Theory, Vol. 35, 517-540, 2000.

59. Han, J., Chung, W. K., Youm, Y., Kim, S. H.: Task Based design of modular Robot Manipulator using efficient genetic Algorithm. Proceedings of IEEE International Conference on Robotics and Automation, Vol. 1, 507-512, 1997.

60. Parker, J. K., Khoogar, A. R., Gol dBerg, D. E.: Inverse kinematics of redundant robots using Genetic Algorithms. Proceedings of IEEE International Conference on Robotics and Automation, Vol. 1, 271-276, 1989.

61. Yun, W. M., Xi, Y. G.: Optimum motion planning in joint space for robots using Genetic Algorithm, Robotics and Autonomous Systems, Vol. 18(4), 373-393, 1996.

62. Ilonen, J., Kamarainen, J. -K., Lampinen, J.: Differential Evolution Training Algorithm for Feed-Forward Neural Networks. In: Neural Processing Letters Vol. 7, No. 1 (2003), 93-105, 2003.

63. Rogalsky, T., Derksen, R.W. and Kocabiyik, S.: Differential Evolution in Aerodynamic Optimization. In: Proceedings of the 46th Annual Conference of the Canadian Aeronautics and Space Institute. 29-36, 1999.

64. Dervis Karaboga, Selcuk Okdem,: A Simple and Global Optimization Algorithm for Engineering Problems: Differential Evolution Algorithm.

Turk J Elec Engin, Vol.12, No.1, 2004.

65. Nurhan Karaboga,: Digital IIR Filter Design Using Differential Evolution Algorithm. EURASIP Applied Signal Processing (2005) No.8, 1269-1276, 2005.

66. Babu**,** B. V. and Angira, R.: Optimization of Non-Linear Chemical Processes Using Evolutionary Algorithm, Proceedings of International Symposium & 55th Annual Session of IIChE (CHEMCON-2002), OU, Hyderabad, December 19-22, 2002.

67. Price, K. V., An introduction to differential evolution. In David Corne, Marco Dorigo, and Fred Glover, editors, New Ideas in Optimization, Advanced Topics in Computer Science, chapter 6, pages 79--108. McGraw-Hill, Maidenhead, Berkshire, England, 1999.

68. Storn, R., Price, K.: Minimizing the Real Functions of the ICEC'96 contest by Diffrential Evolution, in: IEEE Conference on Evolutionary Computation, Nagoya, pp. 842-844, 1996.

69. Allan V. Oppenheim: Adaptive Signal Processing. Prentice Hall Signal Processing Series, Prentice Hall, 22-23, 1989.

70. Mayer, D.G., Kinghorn, B.P. and Archer A. A. Differential evolution – An easy and efficient evolutionary algorithm for model optimisation, Agricultural Systems, 83, 315-328, 2005.

71. Lopez Cruz I.L., Van Willigenburg L.G., Van Straten, G. (2001). Parameter Control Strategy in Differential Evolution Algorithm for Optimal Control. In: M.H. Hamza (ed.) (2001). Proceedings of the IASTED International Conference Artificial Intelligence and Soft Computing (ASC 2001), Cancun , Mexico, pp. 211-216, May 21-24, 2001.

72. Kyprianou A., Worden K., Panet M. "Identification of hysteretic systems using the differential evolution algorithm". Journal of Sound and Vibration Vol. 248, No. 2, 289-314, Nov. 2001.

73. Karaboğa, N., Koyuncu, C. A., Differential Evolution Algorithm for

Designing Optimal Adaptive Linear Combiners, Lecture Notes in Computer Science, Volume 3809 / 2005, pp. 1063-1067, 2005.

74. Karaboğa, N., Koyuncu, C. A., Optimal Determination of Adaptive Linear Combiner Weights Using Differantial Evolution Algorithm. INISTA, Yildiz Technical University, Istanbul, 71-74, 2005.

75. Karaboğa, N., Koyuncu, C. A., Farksal Gelişim Algoritması Kulanılarak ALT Tasarımı. EMO Denizli Otomasyon Sempozyumu, Denizli, 2005.

76. Karaboğa, N., Koyuncu, C. A., İşaret Kestirimi için Farksal Gelişim Algoritması Kullanılarak Adaptif SDY süzgeç Tasarımı. İTUSEM 2005, Adana, 2005.

EK-1

```
clc
close all
clear all

birey=100;
firderece=2;

katsayi=0.2;
f=0.4;
CR=0.5;

maxgener=1000;

x=sin(2*pi*n*25/N);

d=2*cos(2*pi*n*25/N);
x1=[[t];x]

%Gürültünün eklenmesi

SNR = 13 ;
si = std(d);
sg = si * 10^(-SNR/20);
g = sg * randn(size(d));
%d=d+g;


    w=rand(firderece,birey);

    w1(:,:)=w(:,:);
    w2(:,:)=w(:,:);
    w3(:,:)=w(:,:);

for j=1:maxgener


    eniyi1=1;

        for k=1:birey

        r11=round((rand*(birey-1))+1);
        r21=round((rand*(birey-1))+1);
        r31=round((rand*(birey-1))+1);

            while (k==r11|k==r21|k==r31|r11==r21|r21==r31|r11==r31)
            r11=round((rand*(birey-1))+1);
            r21=round((rand*(birey-1))+1);
            r31=round((rand*(birey-1))+1);
            end
%DE/best/2/bin

        %yenicozum1=(w1(:,k)+katsayi*(w1(:,r31)-w1(:,k))+f*(w1(:,r11)-
w1(:,r21)));
```

```
% DE/rand/1/bin

        yenicozum1=((w1(:,r11)+f*(w1(:,r21)-w1(:,r31))));

%çaprazlama,yeniden birleşme

        rs=rand;

        if rs<CR
            adaycozum1=yenicozum1;

        else
            adaycozum1=w1(:,k);

        end


        if sum((abs(cikisyeni1(j)-d(j))).^2)<sum((abs(cikiseski1(j)-
d(j))).^2);
            w1(:,k)=adaycozum1;
        end
        if  sum(abs((cikisyeni1(j)-d(j)).^2))<sum(abs((cikiseski1(j)-
d(j)).^2));
            eniyi1=k;
        end


    %MSE Hata


      e1(j)=(sum((d(j)-cikis1(j)).^2))/birey;

  end % for k

  cikis1=cikis1(:);

end;

      %LMS icin

      for j=1:maxgener

          eniyi2=1;

          for k=1:birey

        r12=round((rand*(birey-1))+1);
        r22=round((rand*(birey-1))+1);
        r32=round((rand*(birey-1))+1);

            while (k==r12|k==r22|k==r32|r12==r22|r22==r32|r12==r32)
            r12=round((rand*(birey-1))+1);
            r22=round((rand*(birey-1))+1);
            r32=round((rand*(birey-1))+1);
            end


% DE/best/2/bin
```

```
       % yenicozum2=(w2(:,k)+katsayi*(w2(:,r32)-w2(:,k))+f*(w2(:,r12)-
w2(:,r22)));


% DE/rand/1/bin

        yenicozum2=((w2(:,r12)+f*(w2(:,r22)-w2(:,r32))));

        %Çaprazlama ve yeniden birleşme

        rs=rand;

        if rs<CR
            adaycozum2=yenicozum2;

        else
            adaycozum2=w2(:,k);

        end

%Seçme

        if sqrt(sum((abs(cikisyeni2(j)-
d(j))).^2))<sqrt(sum((abs(cikiseski2(j)-d(j))).^2));
            w2(:,k)=adaycozum2;
        end
        if  sqrt(sum(abs((cikisyeni2(j)-
d(j)).^2)))<sqrt(sum(abs((cikiseski2(j)-d(j)).^2)));
            eniyi2=k;
        end


%LMS Hata

      e2(j)=sqrt(sum((d(j)-cikis2(j)).^2))/birey;


      end %%for k

       cikis2=cikis2(:);

end;


    %MAE icin

for j=1:maxgener

    eniyi3=1;

      for k=1:birey


        r13=round((rand*(birey-1))+1);
        r23=round((rand*(birey-1))+1);
        r33=round((rand*(birey-1))+1);

            while (k==r13|k==r23|k==r33|r13==r23|r23==r33|r13==r33)
            r13=round((rand*(birey-1))+1);
            r23=round((rand*(birey-1))+1);
```

```
            r33=round((rand*(birey-1))+1);
            end

% DE/best/2/bin

       % yenicozum3=(w3(:,k)+katsayi*(w3(:,r33)-w3(:,k))+f*(w3(:,r13)-
w3(:,r23)));


% DE/rand/1/bin

        yenicozum3=((w3(:,r13)+f*(w3(:,r23)-w3(:,r33))));

%Çaprazlama ve yeniden birleşme

        rs=rand;

        if rs<CR
            adaycozum3=yenicozum3;

        else
            adaycozum3=w3(:,k);

        end

%Seçme


        if sum((abs(cikisyeni3(j)-d(j))))<sum((abs(cikiseski3(j)-
d(j))));
            w3(:,k)=adaycozum3;
        end
        if  sum(abs((cikisyeni3(j)-d(j))))<sum(abs((cikiseski3(j)-
d(j))));
            eniyi3=k;
        end

    %MAE Hata


      e3(j)=(sum(abs(d(j)-cikis3(j))))/birey;

  end %for k

        cikis3=cikis3(:);

    end;

w1(:,eniyi1)
w2(:,eniyi2)
w3(:,eniyi3)


end

e1=e1(:);
e2=e2(:);
e3=e3(:);
```

```
figure(8);plot(e1);xlabel('mse hata fonksiyonu');
figure(9);plot(e2);xlabel('lms hata fonksiyonu');
figure(10);plot(e3);xlabel('mae hata fonksiyonu');
```

EK-2

```
%Sinyal kestirimi

clc
close all;
clear all;

birey=100;
firderece=30;
katsayi=0.2;
f=0.2;
CR=0.8;


w=(rand(firderece,birey));

maxgener=1000;


% iterasyon
T=1000;
n=[0:T-1]';

mu=0.01;

N=500;

X=sin(2*pi*n*25/N)+sin(2*pi*n*10/N)+sin(2*pi*n*50/N);
d=sin(2*pi*n*25/N)+sin(2*pi*n*10/N)+sin(2*pi*n*50/N);

%Gürültü
SNR = 10 ;
si = std(d);
sg = si * 10^(-SNR/20);
g = sg * randn(size(d));
d=d+g;

xg=sin(2*pi*n*25/N-gecikme)+g1+sin(2*pi*n*10/N-
gecikme)+sin(2*pi*n*50/N-gecikme);


for n=1:T,

   y(n) = wlms' * xgor;
   e(n) = d(n) - y(n);
   wlms = wlms + 2*mu*e(n)*xgor;

end;

y=y(:);
```

```
for j=1:maxgener

    eniyi=1;

      for k=1:birey

          w(:,1)=wlms;

        r1=round((rand*(birey-1))+1);
        r2=round((rand*(birey-1))+1);
        r3=round((rand*(birey-1))+1);

            while(k==r1|k==r2|r1==r2)
            r1=round((rand*(birey-1))+1);
            r2=round((rand*(birey-1))+1);
            r3=round((rand*(birey-1))+1);
            end

        yenicozum=w(:,r1)+f*(w(:,r2)-w(:,r3));

        rs=rand;
        if rs<CR
            adaycozum=yenicozum;

        else
            adaycozum=w(:,k);

        end


        if sum((abs(cikisyeni(j)-d(j))).^2)<sum((abs(cikiseski(j)-
d(j))).^2);
            w(:,k)=adaycozum;
        end
        if  sum(abs((cikisyeni(j)-d(j)).^2))<sum(abs((cikiseski(j)-
d(j)).^2));
            eniyi=k;
        end


      e(j)=(sum((d(j)-cikis1(j)).^2));

      end %%for k

       cikis=cikis1(:);

end;

w(:,eniyi)
```

ÖZGEÇMİŞ

Canan Aslıhan Koyuncu 1980 yılında Kayseri'nin Yahyalı İlçesi'nde doğdu. İlkokulu Yahyalı Atatürk İlkokulu'nda, ortaokulu Sami Yangın Anadolu Lisesi'nde, liseyi Melikgazi Lisesi'nde bitirdikten sonra 1998 yılında Erciyes Üniversitesi Elektronik Mühendisliği Bölümünü kazandı. 2003 yılında bu lisans programından mezun olduktan sonra 2004 yılında Erciyes Üniversitesi Fen Bilimleri Enstitüsü Elektronik Mühendisliği Anabilim Dalı'ında yüksek lisans eğitimine başladı.

Adres: Melikgazi Mahallesi
Olgunlar Caddesi
No:33/8
38030 Melikgazi
KAYSERİ

Tel: +90 352 224 33 99
Email: cananaslihan@gmail.com