T.C.
ERCİYES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

ÇOK FAZLI SÜZGEÇ YAPILARININ META SEZGİSEL ALGORİTMALAR KULLANILARAK İNCELENMESİ VE GELİŞTİRİLMESİ

Hazırlayan
Ahmet LOĞOĞLU

Danışman
Prof. Dr. Nurhan KARABOĞA

Doktora Tezi

Şubat 2024
KAYSERİ

T.C.
ERCİYES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

ÇOK FAZLI SÜZGEÇ YAPILARININ META SEZGİSEL ALGORİTMALAR KULLANILARAK İNCELENMESİ VE GELİŞTİRİLMESİ

(Doktora Tezi)

Hazırlayan
Ahmet LOĞOĞLU

Danışman
Prof. Dr. Nurhan KARABOĞA

İkinci Danışman
Doç. Dr. Serdar KOÇKANAT

Bu çalışma, Erciyes Üniversitesi Bilimsel Araştırma Projeleri Birimi tarafından FDK-2020-9663 kodlu proje ile desteklenmiştir.

Şubat 2024
KAYSERİ

BİLİMSEL ETİĞE UYGUNLUK

Bu çalışmadaki tüm bilgilerin, akademik ve etik kurallara uygun bir şekilde elde edildiğini beyan ederim. Aynı zamanda bu kural ve davranışların gerektirdiği gibi, bu çalışmanın özünde olmayan tüm materyal ve sonuçları tam olarak aktardığımı ve referans gösterdiğimi belirtirim.

Ahmet Loğoğlu

İmza

YÖNERGEYE UYGUNLUK

“Çok Fazlı Süzgeç Yapılarının Meta Sezgisel Algoritmalar Kullanılarak İncelenmesi ve Geliştirilmesi” adlı Doktora tezi, Erciyes Üniversitesi Lisansüstü Tez Önerisi ve Tez Yazma Yönergesi’ ne uygun olarak hazırlanmıştır.

Hazırlayan	**Danışman**
Ahmet LOĞOĞLU	Prof. Dr. Nurhan KARABOĞA
İmza	İmza

Elektrik Elektronik Mühendisliği ABD Başkanı

Prof. Dr. Enis GÜNAY

İmza

KABUL VE ONAY

Prof. Dr. Nurhan KARABOĞA danışmanlığında **Ahmet LOĞOĞLU** tarafından hazırlanan **"Çok Fazlı Süzgeç Yapılarının Meta Sezgisel Algoritmalar Kullanılarak İncelenmesi ve Geliştirilmesi"** adlı bu çalışma jürimiz tarafından Erciyes Üniversitesi Fen Bilimleri Enstitüsü **Elektrik Elektronik Mühendisliği** Anabilim Dalında **Doktora** tezi olarak kabul edilmiştir.

09.02.2024

JÜRİ:

Danışman : Prof. Dr. Nurhan KARABOĞA

Üye : Prof. Dr. Mustafa ALÇI

Üye : Prof. Dr. Fatma LATİFOĞLU

Üye : Doç. Dr. Asuman SAVAŞÇIHABEŞ

Üye : Doç. Dr. Ali DURMUŞ

ONAY:

Bu tezin kabulü Enstitü Yönetim Kurulunun ……...… tarih ve ……..……sayılı kararı ile onaylanmıştır.

Prof. Dr. Emel KIZILKAYA AYDOĞAN

Enstitü Müdürü

TEŞEKKÜR

Bu çalışmanın yürütülmesi ve tamamlanması sürecinde pozitif enerjisi ile bana her zaman yardım ve destek sağlayan, süreklilik isteyen akademik hayatın detaylarını öğreten değerli danışmanım Prof. Dr. Nurhan KARABOĞA'ya, ikinci danışmanım Doç. Dr. Serdar KOÇKANAT'a

Bilgi ve deneyimleriyle desteklerini esirgemeyen Değerli Hocalarım Prof. Dr. Derviş KARABOĞA'ya, Prof. Dr. Alper BAŞTÜRK'e

Tez çalışmalarım boyunca her rapor dönemi yorum, öneri ve geri bildirimleri ile tezimin gelişmesine katkıda bulunan Tez İzleme Komitesinin değerli üyeleri Prof. Dr. Mustafa ALÇI ve Prof. Dr. Fatma LATİFOĞLU'na

Bu tez çalışmasına maddi destek veren Erciyes Üniversitesi Bilimsel Araştırma Projeleri Birimi'ne (Proje No: FDK-2012-4156) teşekkür ederim.

Tez savunmama katılarak yorum ve önerileri ile değerli görüşlerini sunma nezaketi gösteren jüri üyelerine, çalışmalarımda emeği geçen tüm çalışma arkadaşlarıma,

İlgisi, desteği ve sabrını her zaman hissettiren değerli eşim, Zeynep Figen LOĞOĞLU'na, yaşlarından beklenmeyen olgunluğu gösterip onlarla geçireceğim zamanlarda çalışmamı anlayışla karşılayan oğullarım Mustafa ve Hasan'a ve kızım Fatma Zehra'ya, bugünlere gelmemde büyük pay sahibi olan anneme ve rahmetli babama teşekkürü bir borç bilirim.

Ahmet LOĞOĞLU

Şubat 2024, KAYSERİ

ÇOK FAZLI SÜZGEÇ YAPILARININ META SEZGİSEL ALGORİTMALAR KULLANILARAK İNCELENMESİ VE GELİŞTİRİLMESİ

Ahmet LOĞOĞLU

Erciyes Üniversitesi, Fen Bilimleri Enstitüsü
Doktora Tezi, Şubat 2024
Danışman: Prof. Dr. Nurhan KARABOĞA

ÖZET

Çok fazlı süzgeçler, girişlerine uygulanan işareti daha etkili bir şekilde süzgeçlemek için bu işareti farklı örnekleme oranlarına sahip farklı frekans bantlarına ayrıştırır ve ardından orijinal işareti tekrar elde etmek için bu alt bantları sentezlemek suretiyle yeniden orijinal işarete dönüştürür. İşareti farklı frekans aralıklarına ayırarak her kanalda farklı işlem yapılmasına olanak sağlayan çok fazlı süzgeç bankaları ses ve video kodlama, çoklu hızlı işaret işleme, sayısal iletişim ve işaret analizi gibi çeşitli alanlarda kullanılır. Ayrıca çok fazlı süzgeç bankaları daha az hesaplama karmaşıklığı, sistem kaynaklarının verimli kullanımı ve geliştirilmiş süzgeç tepki karakteristikleri gibi avantajlar sunmaktadır.

Bu tez çalışmasında meta sezgisel algoritmalarla çok fazlı süzgeç bankası tasarımı gerçekleştirilmiştir. Yapay Arı Koloni Algoritması (Artificial Bee Colony Algorithm, ABC) İyileştirilmiş Yapay Arı Koloni Algoritması (Modified Artificial Bee Colony Algorithm, mABC), Hızlı Yapay Arı Koloni Algoritması (Quick Artificial Bee Colony Algorithm, qABC), Parçacık Sürü Optimizasyonu Algoritması (Particle Swarm Optimization, PSO) ve Genetik Algoritma (Genetic Algorithm, GA) gibi meta sezgisel algoritmalarla tasarlanan sayısal süzgeçler kullanılarak gerçekleştirilen çok fazlı süzgeç bankası tasarımlarının performansları karşılaştırılmıştır.

Sonuç olarak, ABC algoritması diğer algoritmalardan daha iyi ortalama değere sahiptir ve daha kısa sürede çözüme ulaşmaktadır. qABC algoritmasının da standart sapma ve en iyi hata değeri açısından ABC ve mABC algoritmalarından daha iyi sonuçlar vermesi sebebiyle çok fazlı süzgeç bankalarının tasarımında kullanılabileceği gösterilmiştir.

Anahtar Kelimeler: Çok fazlı süzgeç bankası, ABC algoritması, mABC algoritması, qABC algoritması, PSO algoritması, GA, sayısal süzgeç, optimizasyon.

ANALYZING AND IMPROVING POLYPHASE FILTER STRUCTURES USING META HEURISTIC ALGORITHMS

Ahmet LOĞOĞLU

Erciyes University, Graduate School of Natural and Applied Sciences
PhD Thesis, February 2024
Supervisor: Professor Doctor Nurhan KARABOĞA

ABSTRACT

Polyphase filters effectively decompose signals at their inputs into various frequency bands, each with its own sampling rate. This process enables more efficient filtering, followed by a synthesis of these sub-bands to reconstruct the original signal. Such segmentation into different frequency bands allows polyphase filter banks to facilitate distinct processing in each channel. These filters find extensive applications in numerous domains, including audio and video coding, multi-rate signal processing, digital communication, and signal analysis. Key advantages of polyphase filter banks include reduced computational complexity, efficient resource utilization, and enhanced filter response characteristics.

In this thesis, a polyphase filter bank design has been implemented using metaheuristic algorithms. The performances of polyphase filter bank designs, which are realized using digital filters designed with metaheuristic algorithms such as the Artificial Bee Colony Algorithm (ABC), Modified Artificial Bee Colony Algorithm (mABC), Quick Artificial Bee Colony Algorithm (qABC), Particle Swarm Optimization (PSO), and Genetic Algorithm (GA), have been compared.

In conclusion, the ABC algorithm exhibits a superior average value compared to other algorithms and reaches a solution in a shorter period of time. The qABC algorithm has also shown to provide better results in terms of standard deviation and best error value compared to ABC and mABC algorithms, demonstrating its suitability for the design of polyphase filter banks.

Keywords: Polyphase filter bank, ABC algorithm, mABC algorithm, qABC algorithm, PSO algorithm, GA, digital filter, optimization.

İÇİNDEKİLER

ÇOK FAZLI SÜZGEÇ YAPILARININ META SEZGİSEL ALGORİTMALAR KULLANILARAK İNCELENMESİ VE GELİŞTİRİLMESİ

BİLİMSEL ETİĞE UYGUNLUK ii

YÖNERGEYE UYGUNLUK iii

KABUL VE ONAY iv

TEŞEKKÜR v

ÖZET vi

ABSTRACT vii

İÇİNDEKİLER viii

KISALTMALAR xi

TABLOLAR LİSTESİ xiii

ŞEKİLLER LİSTESİ xiv

GİRİŞ 1

1. BÖLÜM

GENEL BİLGİLER ve LİTERATÜR ÇALIŞMASI

1.1. Sayısal İşaret İşleme 5

1.2. Sayısal Süzgeçler 6

1.3. Çoklu Hızlı Sayısal İşaret İşleme 6

1.4. Süzgeç Bankası 7

1.4.1. Analiz ve Sentez Süzgeç Bankası 8

1.4.2. Mükemmel Yeniden Yapılandırma 8

1.5. Süzgeç Bankalarının Sınıflandırılması 10

1.5.1. Aşağı Örnekleme 12

1.5.2. Yukarı Örnekleme 13

1.5.3. Ayrıştırma 13

1.5.4. Ara değerleme 14

1.6. Çok Fazlı Süzgeç Bankası 14

1.6.1. Örnek Azaltıcı/Çoğaltıcılar 17

1.6.2. Optimizasyon Algoritmaları ve Çok Fazlı Süzgeç Bankası Tasarımı 19

1.6.3. Çok Fazlı Süzgeç Bankası Tasarımında Sezgisel Algoritmaların Kullanılması 20

2. BÖLÜM

SÜZGEÇ TASARIMINDA KULLANILAN ALGORİTMALAR

2.1. Yapay Arı Koloni Algoritması 31

2.2. İyileştirilmiş Yapay Arı Koloni Algoritması 37

2.3. Hızlı Yapay Arı Koloni Algoritması 40

2.4. Parçacık Sürü Optimizasyon Algoritması 43

2.5. Genetik Algoritma 44

3. BÖLÜM

SAYISAL SÜZGEÇ TASARIMI

3.1. Sayısal Süzgeç Tasarımı 51

3.2. ABC Algoritması ile Süzgeç Tasarımı 53

3.2.1. Popülasyon Parametresinin İncelenmesi 53

3.2.2. Limit Parametresinin İncelenmesi 55

3.3. mABC Algoritması ile Süzgeç Tasarımı 57

3.3.1. MR Parametresinin İncelenmesi 57

3.4. qABC Algoritması ile Süzgeç Tasarımı 60

3.4.1. RR Parametresinin İncelenmesi 60

3.5. Farklı Derecelerden Süzgeç Tasarımı 64

3.5.1. ABC, mABC ve qABC Algoritmaları ile Tasarım 64

3.5.2. En İyi Ortalama Değer için Süzgeç Tasarımı 69

3.6. Farklı Geçirme Bandına Sahip Süzgeçlerin Tasarımı 72

3.6.1. ABC, mABC ve qABC Algoritmaları ile Tasarım 73

3.6.2. ABC, GA ve PSO Algoritmaları ile Tasarım 77

3.7. Tasarlanan Süzgeçlerin Değerlendirilmesi..... 83

4. BÖLÜM

ÇOK FAZLI SÜZGEÇ BANKASI TASARIMI

4.1. ABC Algoritması ile Çok Fazlı Süzgeç Bankası Tasarımı 86

4.2. qABC Algoritması ile Çok Fazlı Süzgeç Bankası Tasarımı..... 89

4.3. mABC Algoritması ile Çok Fazlı Süzgeç Bankası Tasarımı..... 91

4.4. GA ile Çok Fazlı Süzgeç Bankası Tasarımı..... 93

4.5. PSO ile Çok Fazlı Süzgeç Bankası Tasarımı..... 95

4.6. Önerilen Süzgeç Bankalarının Değerlendirilmesi..... 97

5. BÖLÜM

TARTIŞMA, SONUÇ ve ÖNERİLER

5.1. Tartışma 99

5.2. Sonuç ve Öneriler..... 100

KAYNAKLAR..... 102

6. BÖLÜM

EKLER

EKLER..... 108

EK 1. Farklı Süzgeç Derecelerinde ABC Algoritması ile Elde Edilen Süzgeç Katsayıları..... 108

ÖZGEÇMİŞ..... 110

KISALTMALAR

ABC : Yapay arı kolonisi (Artificial bee colony)
AI : Yapay bağışıklık (Artificial immune)
BA : Yarasa algoritması (Bat algorithm)
BFO : Bakteri besin arama optimizasyonu (Bacterial foraging optimization)
CD : Yoğun disk (Compact disc)
CMOS : Bütünleyici metal oksit yarı iletken (Complementary metal oxide semiconductor)
CSA : Klonal arama algoritması (Clonal search algorithm)
CT : Bilgisayarlı tomografi (Computed tomography)
CIC : Kaskat toplayıcı tarama (Cascaded integrator–comb)
EEG : Elektroensefalogram (Electroencephalogram)
EKG : Elektrokardiyogram
FBMC : Çok taşıyıcılı süzgeç bankası (Filter bank multi carrier)
FPGA : Programlanabilir kapı dizisi (Field programmable gate array)
FIR : Sonlu dürtü yanıtlı (Finite impulse response)
GA : Genetik algoritma (Genetic algorithm)
GSA : Yerçekimi arama algoritması (Gravitational search algorithm)
HB : Yarı bant (Half-band)
HS : Harmoni arama (Harmony search)
IIR : Sonsuz dürtü yanıtlı (Infinite impulse response)
Mbps : Saniyede megabit (Megabits per second)
MR : Modifikasyon oranı (Modification rate)
MR : Manyetik rezonans görüntüleme (Magnetic resonance imaging)
MSE : Ortalama karesel hata (Mean square error)
mABC : İyileştirilmiş yapay arı koloni (Modified artificial bee colony)
NCO : Sayısal kontrollü osilatör (Numerically controlled oscillator)
NPR : Mükemmele yakın yeniden oluşturma (Near perfect reconstruction)
PFB : Çok fazlı süzgeç bankası (Polyphase filter bank)
PRE : Yeniden oluşturma tepe hatası (Peak reconstruction error)

PR : Mükemmel yeniden oluşturma (Perfect reconstruction)
PSO : Parçacık sürü optimizasyonu (Particle swarm optimization)
qABC : Hızlı yapay arı koloni (Quick artificial bee colony)
RF : Radyo Frekans (Radio frequency)
RR : Rastgelelik oranı (Randomization rate)
SA : Benzetilmiş tavlama (Simulated annealing)
SN : Besin sayısı (Source number)
s : Saniye (Second)
TMUX : Çoklu erişimli programlanabilen çoğullayıcı (Transmultiplexer)
TS : Tabu arama (Tabu search)
USG : Ultrasonografi (Ultra-sonography)

TABLOLAR LİSTESİ

Tablo 3.1. ABC algoritmasının popülasyon büyüklüğünün incelenmesi 54
Tablo 3.2. Sabit popülasyon büyüklüğüne bağlı olarak limit parametresinin etkisinin incelenmesi 56
Tablo 3.3. mABC algoritmasında MR parametresinin incelenmesi 58
Tablo 3.4. qABC algoritmasında RR parametresinin incelenmesi 61
Tablo 3.5. ABC, mABC ve qABC algoritmalarının süzgeç derecelerine göre karşılaştırmalı performans analizi 66
Tablo 3.6. N=32 için ABC algoritmasıyla elde edilen değerler 69
Tablo 3.7. Geçirme ve durdurma bandı frekanslarına göre ABC, mABC ve qABC algoritmaları ile elde edilen veriler 74
Tablo 3.8. ABC, PSO ve GA algoritmalarının süzgeç derecelerine göre karşılaştırmalı performans analizi 78
Tablo Ek 1.1. Süzgeç derecesi 8 olduğunda elde edilen katsayılar 108
Tablo Ek 1.2. Süzgeç derecesi 16 olduğunda elde edilen katsayılar 108
Tablo Ek 1.3. Süzgeç derecesi 32 olduğunda elde edilen katsayılar 108
Tablo Ek 1.4. Süzgeç derecesi 64 olduğunda elde edilen katsayılar 108
Tablo Ek 1.5. Süzgeç derecesi 128 olduğunda elde edilen katsayılar 109

ŞEKİLLER LİSTESİ

Şekil 1.1. Analiz ve sentez süzgeç bankası 8
Şekil 1.2. Basit bir yeniden yapılandırma süzgeç bankası 9
Şekil 1.3. Düzgün karmaşık süzgeç bankalarında çift (a) ve tek (b) alt kanal yığını 11
Şekil 1.4. Aşağı örnekleyici 13
Şekil 1.5. Yukarı örnekleyici 13
Şekil 1.6. Örtüşme önleyici süzgeç ve aşağı örnekleyiciden oluşan örnek azaltıcı 14
Şekil 1.7. Bir yukarı örnekleyici ve örtüşme önleyici süzgeçten oluşan ara değerleyici 14
Şekil 1.8. Çok fazlı süzgeç bankası yapısı 15
Şekil 1.9. Üç katlı çok fazlı süzgeç yapısı 17
Şekil 1.10. Üç katlı devrik çok fazlı süzgeç yapısı 18
Şekil 1.11. Birbirine denk iki örnek azaltıcı sistem 18
Şekil 1.12. Birbirine denk iki örnek artırıcı sistem 18
Şekil 1.13. Çok fazlı süzgeç bankası tasarımı hakkında 2000-2023 yılları arasında yapılan yayın sayıları 28
Şekil 1.14. Çok fazlı süzgeç bankası tasarımı hakkında 2000-2023 yılları arasında yapılan yayınların ülkelere göre dağılımı 29
Şekil 1.15. Çok fazlı süzgeç bankası tasarımı hakkında 2000-2023 yılları arasında yapılan yayınların kullanılan yöntemlere göre dağılımı 29
Şekil 2.1. ABC Algoritmasının sözde kodu 32
Şekil 2.2. ABC Algoritmasının akış diyagramı. 36
Şekil 2.3. mABC Algoritmasının sözde kodu 37
Şekil 2.4 mABC Algoritmasının akış diyagramı 39
Şekil 2.5. qABC Algoritmasının sözde kodu 40
Şekil 2.6. qABC Algoritması akış diyagramı 42
Şekil 2.7. PSO Algoritmasının sözde kodu 44
Şekil 2.8. GA'nın sözde kodu 45
Şekil 2.9. GA'nın akış diyagramı 47
Şekil 3.1. Çok fazlı süzgeç bankası tasarımında önerilen modelin iş akışı 51

Şekil 3.2. Farklı MR değerleri için mABC algoritması kullanılması durumunda iterasyona bağlı MSE'nin değişimi 59
Şekil 3.3. mABC algoritmasının MR parametresinin farklı değerlerine göre performansı 60
Şekil 3.4. Farklı RR değerleri için qABC algoritması ile iterasyona bağlı MSE'nin değişimi 62
Şekil 3.5. qABC algoritmasının RR parametresinin farklı değerlerine göre performansı 63
Şekil 3.6. Süzgeç derecelerine göre ABC, mABC, qABC algoritmalarının ortalama değerler açısından karşılaştırılması 67
Şekil 3.7. Süzgeç derecelerine göre ABC, mABC, qABC algoritmalarının standart sapma değerleri açısından karşılaştırılması 67
Şekil 3.8. Süzgeç derecelerine göre ABC, mABC, qABC algoritmalarının en iyi hata değerleri açısından karşılaştırılması 68
Şekil 3.9. Süzgeç derecelerine göre ABC, mABC, qABC algoritmalarının zaman değerleri açısından karşılaştırılması 69
Şekil 3.10. N=32 için ABC qABC ve mABC algoritmalarıyla iterasyona bağlı olarak MSE'nin değişimi 71
Şekil 3.11. Geçiş bandı frekansı $w_p = 0,1$ ve durdurma bandı frekansı $w_s = 0,9$ alındığında ortalama karesel hata grafiği 75
Şekil 3.12. Geçiş bandı frekansı $w_p = 0,4$ ve durdurma bandı frekansı $w_s = 0,6$ alındığında ortalama karesel hata grafiği 76
Şekil 3.13. ABC, PSO ve GA algoritmalarının N=32 için yakınsama eğrileri 80
Şekil 3.14. ABC, PSO ve GA algoritmalarının süzgeç derecelerine göre elde edilen ortalama değerlerinin karşılaştırmalı analizi 81
Şekil 3.15. ABC, PSO ve GA algoritmalarının süzgeç derecelerine göre elde edilen standart sapma değerlerinin karşılaştırmalı analizi 82
Şekil 3.16. ABC, PSO ve GA algoritmalarının süzgeç derecelerine göre elde edilen en iyi hata değerlerinin karşılaştırmalı analizi 83

Şekil 4.1. N=32 için ABC algoritması ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği 87
Şekil 4.2. N=256 için ABC algoritması ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği 88
Şekil 4.3. N=32 için qABC algoritması ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği 90
Şekil 4.4. N=256 için qABC algoritması ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği 91
Şekil 4.5. N=32 için mABC algoritması ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği 92
Şekil 4.6. N=256 için mABC algoritması ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği 93
Şekil 4.7. N=32 için GA ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği..... 94
Şekil 4.8. N=256 için GA ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği 95
Şekil 4.9. N=32 için PSO ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği 96
Şekil 4.10. N=256 için PSO ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği 97

GİRİŞ

Bir işaret, genellikle zamanla değişen bir nicelik veya parametredir. İşaretler çeşitli biçimlerde olabilir. Bunlar ses, görüntü, radyo dalgaları, ısı veya hatta bir hisse senedinin zamanla değişen fiyatı gibi büyüklük ve veriler olabilir. İşaretler genellikle bir bilgi kaynağından bir alıcıya bilgi taşırlar. İşaret işleme, bu işaretleri analiz etmek, değiştirmek veya işlemek için kullanılan bir dizi teknik ve algoritma içermektedir. İşlem süreci, genellikle işaretin türüne ve hangi bilginin çıkarılması gerektiğine bağlı olarak değişir.
Sayısal İşaret İşleme (Digital Signal Processing, DSP), bilginin matematik olarak işlenmesi işlemidir. Bu bilgi, ses, görüntü, sıcaklık veya diğer her türlü işaret veya ölçüm olabilir [1, 2]. En basit şekliyle, bir işaret zamanla değişen bir niceliktir ve bu değişimler genellikle periyodik veya rastgele olabilir. Sayısal işaret işleme, analiz, sentez ve değişiklikler yapmak için bu işaretleri sayısal olarak genellikle de ikili gösterimle kullanır. Analog işaretler genellikle bir dönüştürücü (Analog to Digital Converter, ADC) kullanılarak sayısal formata dönüştürülür, işlemler yapılır ve daha sonra gerektiğinde bir dönüştürücü (Digital to Analog Converter, DAC) kullanılarak orijinal haline geri dönüştürülür. Sayısal işaret işlemenin uygulama alanları aşağıda verilmektedir:

- Ses ve Görüntü İşleme: İşaretleri analiz etmek, iyileştirmek ve sentezlemek için işaret işleme yöntemlerinin kullanıldığı en geniş alanlardan biridir. Bu alanda genellikle ses sıkıştırma, görüntü iyileştirme, ses tanıma ve görüntü tanıma gibi teknolojiler bulunmaktadır.
- Telekomünikasyon: Sayısal işaret işleme, analog işaretleri sayısal hale getirme ve bunları daha sonra tekrar analog hale dönüştürme özelliği nedeniyle telekomünikasyonda önemli bir yer tutar. Bu, ses ve görüntü verilerinin daha verimli bir şekilde iletilmesini ve depolanmasını sağlar.

- Biyomedikal Sistemler: Elektrokardiyografi (EKG), elektroensefalogram (EEG), manyetik rezonans görüntüleme (Magnetic Resonance, MR), bilgisayarlı tomografi (Computed Tomography, CT), ultrasonografi (Ultra-Sonography, USG) vb. gibi biyomedikal işaretlerin işlenmesinde DSP geniş bir uygulama alanına sahiptir.
- Radyo ve Televizyon Yayını: DSP, radyo ve televizyon yayınlarının kalitesini artırmak için de kullanılmaktadır. Örneğin, sayısal televizyon ve radyo, analog işaretlerin sayısal hale getirilmesi ve işlenmesi prensibine dayanmaktadır.
- Radar ve Sonar Sistemleri: Radar ve sonar sistemleri, elektromanyetik veya ses dalgalarının yankılarını analiz ederek nesneleri ve ortamları haritalamak için sayısal işaret işleme tekniklerini kullanır.
- Sismoloji: Bu sahada da deprem verilerinin analizi ve yorumlanması için DSP teknikleri kullanılır.
- Ses ve Görüntü Sıkıştırma: Bu alanda DSP teknikleri, ses ve görüntü verilerini sıkıştırmak ve depolamak için kullanılır. Bu teknolojiler, özellikle internet üzerinden büyük veri aktarımlarında ve depolamada çok önemlidir.

Bu geniş kullanım alanları, DSP'nin ne kadar önemli ve etkili bir teknoloji olduğunu göstermektedir. Son yıllarda DSP, çok çeşitli sektörlerde verimliliği ve güvenilirliği artırma özellikleri nedeniyle elektronik ve haberleşme alanında yaygın bir teknik haline gelmiştir [3].

Bir işaretin sayısallaştırılması sürecini gerçek rakamlarla göstermek için, ses işareti örneği kullanabilir. Bu süreç genellikle üç ana adımdan oluşur: örnekleme, kuantalama ve kodlama:

1. Örnekleme

Bir ses işaretini sayısallaştırmak için örnekleme hızı çok önemli bir ölçüttür. Örneğin bu ses işareti CD (Compact Disc, Yoğun Disk) kalitesinde olacaksa, CD kalitesindeki ses için standart örnekleme hızı 44.1 kHz olduğundan, bu saniyede 44100 örnek alındığı anlamına gelir. Ses dalgası, 44.1 kHz hızında örneklenir. Yani, her saniyede ses dalgasının 44100 farklı noktası kaydedilir.

2. Kuantalama

CD kalitesi ses için standart bit derinliği 16 bit'tir. Bu, her örneğin 2^{16}, yani 65536 farklı değer alabileceği anlamına gelmektedir. Her örneklenen değer, bu 65536 değer arasından

birine dönüştürülür. Bu, analog işaretin genliğinin sayısal değerlere dönüştürülmesi demektir.

3. Kodlama

Her kuantalanmış örnek, belirli bir sayıda bit kullanılarak sayısal bir değere dönüştürülür. 16 bit derinlikte, her örnek için 16 bit kullanılır. Örneğin, bir ses işaretinin ilk örneği 0.5 genliğinde olsun. Bu, kuantalama sürecinde belirli bir sayısal değere dönüştürülür örneğin, 32768 ve sonra bu değer 16 bit bir sayı olarak kodlanır.

Bir saniyelik stereo ses yani iki kanal için veri hızı şöyle hesaplanmaktadır:

44100 (örnek/s) × 16 (bit/örnek) × 2 (kanal) = 1411200 (bit/s) veya yaklaşık 1.4 Mbps olarak bulunur. Bu sayısal veriler, daha sonra saklama, iletim veya işleme için kullanılabilir. Örneğin, ses dosyası olarak saklanabilir, internet üzerinden iletilmek üzere sıkıştırılabilir veya ses tanıma yazılımında işlenebilir. Bu örnek, bir analog ses işaretinin nasıl sayısallaştırıldığını ve sonuçta ortaya çıkan verinin nasıl işlenebileceğini açıklamaktadır. Benzer süreçler, görüntü ve diğer tür işaretler için de uygulanabilir, ancak örnekleme hızları, bit derinlikleri ve kodlama yöntemleri farklılık gösterebilmektedir [4].

Bu tez çalışmasında, Karaboğa [5] tarafından 2005 yılında önerilmiş olan yapay arı koloni (Artificial Bee Colony, ABC) algoritması literatürde ilk kez çok fazlı süzgeç bankalarının tasarımında ve optimizasyonunda kullanılmış, sonuçlar performans açısından literatürde sıkça kullanılan diğer sezgisel algoritmalarla karşılaştırmalı olarak değerlendirilmiştir.

Tez çalışmasında ilk olarak, Çok Fazlı Süzgeç Bankası (Polyphase Filter Bank, PFB) tasarım problemlerinin sezgisel algoritmalarla optimizasyonu hakkında bir literatür taraması yapılmıştır. Bu inceleme, 1990-2023 yılları arasında IEEE Xplore, Web of Science, Taylor ve Francis, Science Direct, Springer Link veri tabanlarını kapsayacak şekilde gerçekleştirilmiştir. Bu veri tabanlarından farklı olarak Google Scholar web arama motorundan da faydalanılmıştır.

Tez çalışmasının birinci bölümünde, DSP, sayısal işaretler, sayısal süzgeçler ve uygulama alanları hakkında temel bilgiler verilmiştir. Ayrıca, literatürde sezgisel algoritmalar kullanılarak gerçekleştirilen çok fazlı sayısal süzgeç tasarımları ayrıntılı olarak incelenmiş ve elde edilen sonuçlar bu ilk bölümde verilmiştir.

Tez çalışmasının ikinci bölümünde, çok fazlı sayısal süzgeç bankası tasarımlarında kullanılan ABC Algoritması, Modifiye Yapay Arı Koloni (Modified Artificial Bee Colony, mABC) Algoritması, Hızlı Yapay Arı Koloni (Quick Artificial Bee Colony, qABC) Algoritması, Parçacık Sürü Optimizasyonu (Particle Swarm Optimization, PSO)

Algoritması ve Genetik Algoritma (Genetic Algorithm, GA) gibi literatürde oldukça geniş uygulama alanına sahip meta sezgisel algoritmalar anlatılmıştır.

Tez çalışmasının üçüncü bölümünde PFB tasarımında kullanılacak olan sayısal süzgeçlerin sezgisel algoritmalarla tasarımı konusu ayrıntılı olarak incelenmiştir. Bu bölümde kullanılan algoritmaların performansları da mukayeseli olarak değerlendirilmiştir. Ayrıca, bölümün özgünlüğünün ön plana çıkması için ABC, mABC, qABC algoritmalarına ait kontrol parametre değerlerinin nasıl seçilmesi gerektiği konusu araştırılmış ve ek olarak değişen geçiş bandı durdurma bandı frekansları ile süzgeç dereceleri için süzgeç tasarımlarının değerlendirilmesi gerçekleştirilmiştir. Dördüncü bölümde önerilen yöntemle tasarlanan PFB tasarımları incelenmiştir.

Son olarak beşinci bölümde tartışma, sonuç ve önerilere yer verilmiştir.

1. BÖLÜM

GENEL BİLGİLER VE LİTERATÜR ÇALIŞMASI

1.1. Sayısal İşaret İşleme

DSP, işaretlerin sayısal değerlere, tercihen ikili sayılara dönüştürülüp işlendiği bir süreçtir. Bu süreç sırasıyla aşağıdaki adımlardan oluşur:

1. Örnekleme: Analog bir işaret sürekli olduğundan, sayısal olarak işlenmeden önce belirli zaman aralıklarında örneklenir. Bu işlem, işaretin sürekli zaman formunun ayrık zaman formuna dönüştürülmesini sağlar.
2. Kuantalama: Örnekleme sonrası, her bir örneğin genlik değeri belirli bir aralığa sınırlanır.
3. Kodlama: Sınırlandırılan değerler daha sonra sayısal bir biçime dönüştürülür ve bu durum için genellikle ikili kodlama kullanılır.
4. İşleme: Sayısal işaret artık çeşitli algoritmalar ve matematik işlemler kullanılarak işlenebilir. Bu, işaret üzerinde süzgeçleme, frekans analizi, sıkıştırma veya diğer değişikliklerle yapılır.
5. Sayısal Analog Dönüşüm: Eğer sonuç analog bir biçimde gerekiyorsa, sayısal işaret bir DAC kullanılarak tekrar bir analog işarete dönüştürülür.

Bu işlemler, işaretleri analiz etmek, tanımlamak ve değiştirmek için mühendislik, tıp ve diğer alanlarda bilim adamları tarafından yaygın olarak kullanılmaktadır.

Elektronik devrelerin orta ölçekli ve büyük ölçekli birleşimi işleminden çok büyük ölçekli birleşimine kadar tümleşik devre tekniklerindeki hızlı ilerlemeler, etkili, küçültülmüş, hızlı ve ucuz, özel amaçlı sayısal işlemcilerin üretimini teşvik etmiş ve böylece normalde herhangi bir analog devre veya analog işaret işleme sistemi için çok karmaşık veya çok maliyetli olan karmaşık sayısal işaret işleme görevlerini yerine getirebilen son derece gelişmiş sayısal donanım sistemlerinin geliştirilmesine yol açmıştır.

DSP, çeşitli türdeki işaretleri süzgeçleme, ölçme veya sıkıştırma ve analog sinyaller üretme amacıyla kullanılmaktadır. Bilgi değişken genlikte elektrik darbelerine dönüştürülürken, sayısal işaret bilgisi her bir veri bitini yansıtan iki ayırt edilebilir genlikte olacak şekilde ikili formata dönüştürülür [1].

1.2. Sayısal Süzgeçler

İşaret işlemenin birçok uygulamasında, bir işaretin göreceli genlik ve frekans bölümlerinin değiştirilmesi istenir, bu işlem genellikle süzgeçleme olarak adlandırılır. Çıkışın Fourier dönüşümü, girişin Fourier dönüşümü ve sistemin frekans yanıtının çarpımı olduğundan, uygun frekans yanıtı kullanılmalıdır. Bir analog süzgeç, gerekli süzgeçleme etkisini oluşturabilmek için dirençler, kapasitörler ve işlemsel yükselteçler (Operation Amplifiers, Op-Amp.) gibi elektronik devre elemanlarını ve sayısal mantığı kullanır. Sayısal süzgeç, örnek değerler üzerinde sayısal hesaplamalar yapmak için sayısal işlemciler kullanır. Bir işlemci, kişisel bilgisayar gibi genel amaçlı bir bilgisayar veya özel bir sayısal işaret işlemci çipi olabilir. Bu süzgeçlerin başlıca avantajları şunlardır: Sayısal bir süzgeç programlanabilir, yani işlemcinin belleğinde saklanan bir program tarafından çalıştırılır. Bu, sayısal süzgeçlerin donanımı etkilemeden kolayca değiştirilebileceği anlamına gelir. Sayısal süzgeçler genel amaçlı bir bilgisayarda kolayca tasarlanabilir, test edilebilir ve uygulanabilir. Analog süzgeç devrelerin özellikleri, özellikle aktif bileşenler içerenler, kararlılıktan sapabilirler ve sıcaklığa bağlı olarak değişim gösterebilirler. Sayısal süzgeçler hem zaman hem de sıcaklık ile kararlılık göstermektedir. Analog muadillerinin aksine, sayısal süzgeçler düşük frekanslı işaretleri doğru bir şekilde işleyebilir. DSP teknolojisinin hızı arttığından, sayısal süzgeçler radyo frekansı (Radio Frequency, RF) alanındaki yüksek frekanslı işaretler için kullanılmaya başlanmıştır ki bu işaretler son zamanlarda analog teknolojinin özel koruması altındaydı. Sayısal süzgeçler, işaretleri çeşitli şekillerde işleyebilme yetenekleri açısından oldukça kullanışlıdır; buna bazı sayısal süzgeç türlerinin işaret özelliklerindeki değişiklikleri optimize etme yeteneği de dahildir [6].

1.3. Çoklu Hızlı Sayısal İşaret İşleme

Çoklu hızlı sayısal işaret işleme (Multirate Digital Signal Processing), konuşma, ses ve videonun alt bant kodlamasındaki uygulamalar, çoklu taşıyıcı veri iletimi vb. uygulamalar nedeniyle son yıllarda çok dikkat çekmiştir. Çoklu hızlı işaret işlemenin önemli bir

özelliği, yüksek işlem gücüdür. Çoklu hızlı bir sistem, orijinal işaretlerin örnekleme hızını, onları işlemeden önce veya işlerken artırabilir veya azaltabilir. Bu işaretler daha sonra farklı örnekleme hızına sahip çoklu hızlı sistemin çeşitli bölümlerinde eş zamanlı olarak işlenebilir. Çoklu hızlı işaret işlemenin birçok avantajı bulunmaktadır ve çoklu hızlı işaret işleme teknikleri birçok alanda uygulanmaktadır. Bu alanlardan bazıları konuşma ve görüntü sıkıştırma, sayısal ses kodlama/kod çözme, istatistiksel ve uyarlamalı işaret işleme, ayrık zamanlı çok boyutlu işaret işleme, diferansiyel eşitliklerin sayısal yöntemlerle çözümü, jeofiziksel işaret işleme, yüksek çözünürlüklü görüntü elde etme ve rasyonel örnekleme hızı dönüştürme olarak verilebilir [7-9]. Sayısal süzgeç bankaları, çok oranlı DSP'nin en önemli uygulamalarıdır [10]. Son yıllarda çeşitli sayıda farklı süzgeç bankası geliştirilmiştir. Bu süzgeç bankaları arasında, Kosinüs Modülasyonlu süzgeç bankaları (Cosine-Modulated Filter Banks, CMFB) çok popülerdir çünkü uygulanması kolaydır ve mükemmel yeniden yapılandırma (Perfect Reconstruction, PR) sağlayabilirler. Ayrık Fourier Dönüşümü (Discrete Fourier Transform, DFT), PFB yüksek hesaplama verimliliği sağlayan bir başka popüler süzgeç bankasıdır, ancak alt bant işaretlerinin alt örneklenmesinden kaynaklanan diğer örtüşme bileşenlerini iptal edememesi bir dezavantajdır. DFT süzgeç bankasına belirli bir değişiklik getirerek bu dezavantajının üstesinden gelinebilir. Değiştirilmiş DFT (Modified DFT) süzgeç bankaları da PR sağlayabilir.

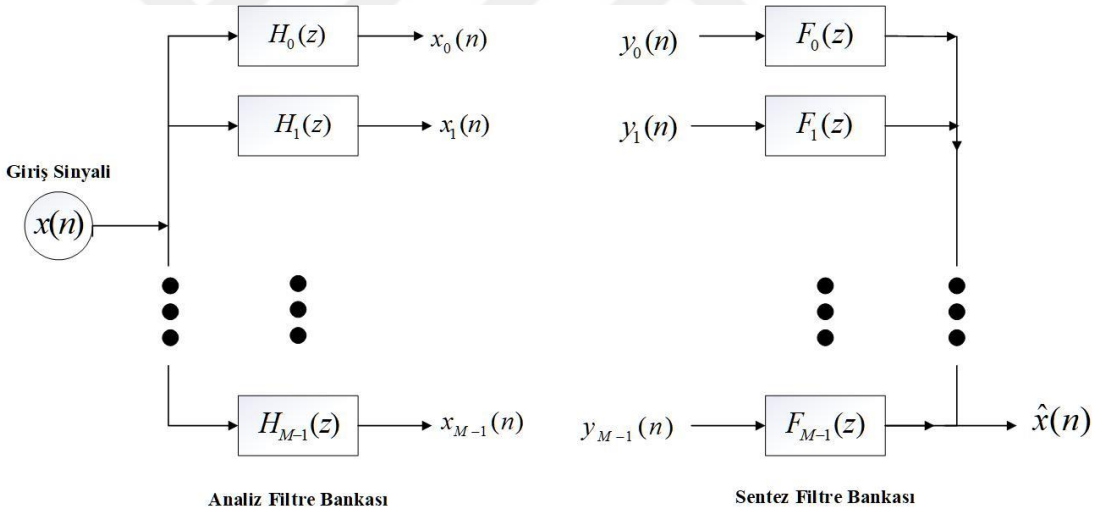
1.4. Süzgeç Bankası

Süzgeç bankaları, son yıllarda son derece önemli bir yapı haline gelmiştir. DSP eğitiminde önemli ölçüde aritmetik kullanımı gerekmektedir ki materyalin doğası, yöntemleri kesin olarak belirlemek, özelliklerini ve performanslarını sağlam bir şekilde oluşturmak için matematiği kullanmayı zorunlu kılmaktadır. Sayısal işaret işlemede birçok uygulama alanı örnekleme hızının değiştirilmesini gerektirir. Oranın değişmesiyle ilgili süreçler, ara değer ekleme ve değer eksiltmedir. İki temel örnekleme hızı değiştirme işlemi vardır: aşağı örnekleme ve yukarı örnekleme. Ara değerleme ve eksiltme işlemlerini uygulamak için en önde gelen maliyeti azaltan yapılardan biri noktasal zaman yapısıdır. Örnek azaltma, son derece işaret işleme sisteminde örnek oranı F_s'yi azaltma yöntemidir ve ara değerleme bunun tersidir, bir işaret işleme sisteminde amaç örnekleme hızı F_s'yi artırmaktır. Bu işlemler, işaret işleme sistemlerinde oldukça yaygındır ve

neredeyse sürekli olarak bir sonlu dürtü yanıtlı (Finite Impulse Response, FIR) süzgecinden yararlanılarak gerçekleştirilir [11].

1.4.1. Analiz ve Sentez Süzgeç Bankası

İki ana süzgeç bankası türü vardır. Birincisi analiz süzgeç bankası (Analysis Filter Bank, AFB) diğeri sentez süzgeç bankası (Synthesis Filter Bank, SFB) olarak adlandırılır. Bir analiz süzgeç bankası, giriş işaretini $X_k(n)$ M alt bant işaretlerine bölen bir $H_k(n)$ analiz süzgeçleri takımıdır ve bir sentez süzgeç bankası, M işareti $Y_k(n)$ 'i birleştiren bir M sentez süzgeçleri $F_k(z)$ takımıdır. Şekil 1.1'de gösterildiği gibi yeniden oluşturulmuş bir $\hat{x}(n)$ işaretine dönüştürülür. Analiz süzgeç bankası, giriş işaretini farklı bir frekans spektrumuna sahip farklı bir alt banda ayrıştırır ve sentez süzgeç bankası, farklı alt bant işaretini yeniden yapılandırır ve orijinal işaretin değiştirilmiş bir versiyonunu üretir.



Şekil 1.1. Analiz ve sentez süzgeç bankası

1.4.2. Mükemmel Yeniden Yapılandırma

Analiz ve sentez bankalarının sırasıyla analiz ve sentez transfer matrisleri $A(z)$ ve $S(z)$ ile basamaklanmasını göz önünde bulundurursak; sabit kazanç c ve tamsayı gecikmesi d için, Eşitlik 1.1'deki gibi olup,

$$\hat{x}(n) = cx(n-d) \tag{1.1}$$

kaskat bağlı sistemin çıktısı, girdinin ölçeklenmiş ve gecikmeli bir versiyonu ise; bu süzgeç bankası mükemmel yeniden yapılandırma özelliğine sahiptir [12-15]. Şekil 1.3'de

basit bir yeniden yapılandırma süzgeç bankası görülmektedir. Ölçeklendirme ve gecikmenin gerçekleşmesi için, en genel durumda, Eşitlik 1.2'deki gibi

$$S(z)A(z) = cz^{-d}I_M \tag{1.2}$$

I_M'nın $M \times M$ birim matrisi olduğu Eşitlik 1.3'teki koşul sağlanmalıdır.

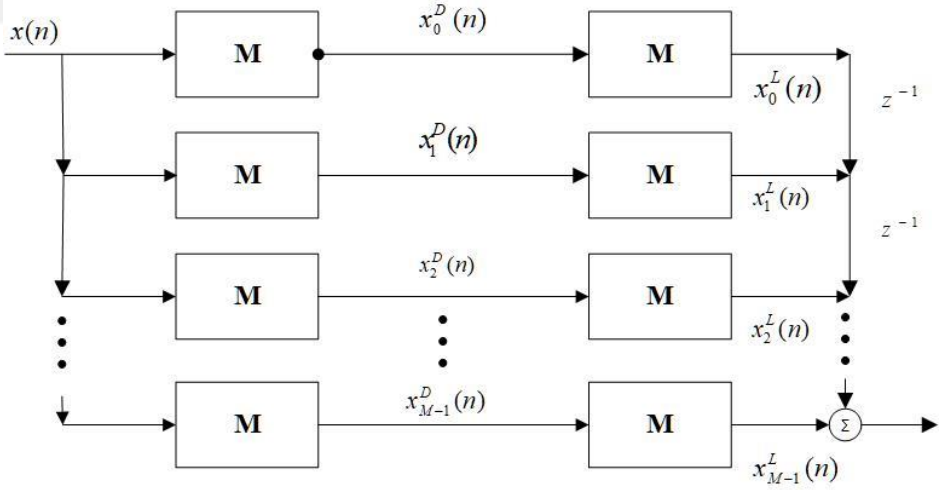
$$Y_M(z) = \frac{1}{M}\sum_{k=0}^{M-1} X\left(z^{\frac{1}{M}}W_M^k\right) \tag{1.3}$$

Örnek olarak, Şekil 1.2'de gösterilen basit mükemmel yeniden yapılandırma süzgeç bankası (Perfect Reconstruction Filter Bank, PRFB) için, bir alt örnekleyicinin çıktısı için, $X_k^D(z)$ Eşitlik 1.4'deki gibi yazılabilir:

$$X_k^D(z) = \frac{1}{M}\sum_{i=0}^{M-1}\left(z^{\frac{-k}{M}}W^{-i}\right)X\left(z^{\frac{1}{M}}W^{-i}\right) \tag{1.4}$$

Yukarı örnekleyicilerin çıkışları da Eşitlik 1.5'teki gibi yazılabilir:

$$X_k^L(z) = \frac{1}{M}\sum_{i=0}^{M-1}\left(z^{-k}W^{-i}\right)X\left(zW^{-i}\right) \tag{1.5}$$



Şekil 1.2. Basit bir yeniden yapılandırma süzgeç bankası

Süzgeç bankasının çıktısı, Eşitlik 1.5'teki *k* değerlerinin gecikmeli toplamıdır, bu nedenle;

$\hat{X}(z) = \frac{1}{M}\sum_{k=0}^{M-1} z^{-(M-1-k)}z^{-k}\sum_{i=0}^{M-1} W^{-i}X\left(zW^i\right)$ alınırsa Eşitlik 1.6 ve 1.7 elde edilir.

$$\hat{X}(z) = \frac{1}{M}z^{-(M-1)}\sum_{i=0}^{M-1} X(z) \tag{1.6}$$

$$\hat{X}(z) = z^{-(M-1)}X(z) \tag{1.7}$$

Zaman bölgesinde $\hat{x}(n) = x(n - M + 1)$ ki bu Eşitlik 1.1'deki $c = 1$ve $d = M - 1$ ile aynıdır. Yukarı/aşağı örnekleyiciler yapıdan alınıp yalnızca gecikmeler bırakılırsa, sonucun yine de bir mükemmel yeniden yapılandırma süzgeç bankası olduğu görülmektedir.

1.5. Süzgeç Bankalarının Sınıflandırılması

Haberleşmede işaret işleme bağlamında ayrık Fourier dönüşümü süzgeç bankaları (Discrete Fourier Transform Filter Bank, DFT-FB) çok kanallı vericiler ve alıcılarda kanallaştırma kanal süzgeçleme amaçları için kullanışlıdır. Ancak, PR veya Mükemmele Yakın Yeniden Yapılandırma (Near Perfect Reconstruction, NPR) süzgeç bankası sistemleri için doğrudan uygulanabilir değildir. Süzgeç bankalarının bazı temel terminolojisi ve genel sınıflandırmaları aşağıdaki gibidir.

- **Tek tip süzgeç bankası**

Tek tip süzgeç bankası tüm kanallar için eşit bant genişliğine sahiptir. DFT-FB ve geliştirilmiş varyantları, alçak geçiren bir prototipten frekans kaydırma ve modülasyon ile elde edilen alt bant süzgeçlerine sahiptir. Gerçek tek tip süzgeç bankaları genellikle simetrik olarak yerleştirilmiş karmaşık frekans kaydırma tabanlı alt kanalların ikili olarak birleştirilmesiyle elde edilir.

- **Ağaç yapılı süzgeç bankası**

Ağaç yapılı süzgeç bankaları, yarı bant alçak geçiren veya yüksek geçiren süzgeç çiftlerinin büyük bir kısmını kesen veya ara değerleme yapan bir ağaç yapılı basamaklandırma yoluyla elde edilir.

- **Gerçek ve karmaşık süzgeç bankası**

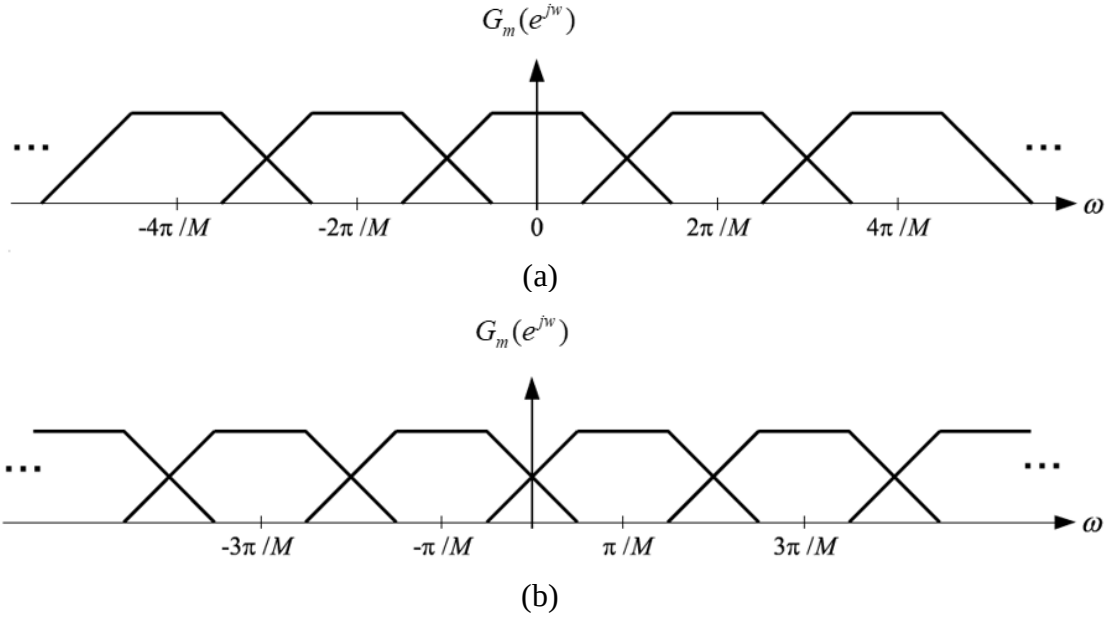
DFT-FB ve türevleri hem alt bant işaretleri hem de birleştirilmiş yüksek oranlı işaret için karmaşık işaret modellerine sahiptir. Böyle bir model, taşıyıcı modülasyonlu iletişim sistemlerinde dalga biçimi işleme için tercih edilir. Gerçek yüksek oranlı işaret modeli temel bant iletim sistemlerinde, örneğin kablolu iletişimde uygulanabilir ve buna alt taşıyıcı işaret seviyesi eşlenik simetrisi uygulanarak ulaşılabilir. Alt bant tabanlı kaynak kodlama gibi diğer çeşitli uygulamalarda hem alt bant hem de birleşik işaretler için gerçek işaret formatına ihtiyaç duyulur. Yaygın olarak kullanılan CMFB'ları ayrık Fourier dönüşümü süzgeç bankaları ile de yakın bir bağlantıya sahiptir.

- **Kritik örneklemeli süzgeç bankası**

DFT-FB gibi, alt bant işaretlerindeki toplam örnek sayısı birleştirilmiş yüksek oranlı işaretteki ile aynıdır, yani alt bant örnekleme oranı f_s/M' dir. Çeşitli uygulamalarda, alt bant işaretlerinin tipik olarak 2 gibi küçük bir tamsayı faktörü ile aşırı örneklenmesi gerekli veya faydalıdır. Özellikle, Ofset Kuadratür Genlik Modülasyonu (Offset Quadrature Amplitude Modulation, OQAM) alt taşıyıcı modülasyonu (FBMC/OQAM) dalga formu ile çoklu taşıyıcı süzgeç bankası (Filter Bank Multi Carrier OQAM) için kullanılan OQAM işaret modeli, iki kat fazla örneklenmiş modelin kullanılmasını gerektirir.

- **Çift / Tek yığınlı alt kanallar**

DFT-FB çift yığınlı alt bant düzenlemesine sahiptir, bu da alt bantlardan birinin sıfır frekans doğru akım merkezli olduğu anlamına gelir. Tek yığınlı durumda, iki alt kanal Şekil 1.3'te gösterildiği gibi DC etrafında simetrik olarak konumlandırılmıştır.



Şekil 1.3. Düzgün karmaşık süzgeç bankalarında çift (a) ve tek (b) alt kanal yığını

- **Analiz / Sentez yapılandırması**

Bir süzgeç bankasının yüksek hızlı bir işareti alt bantlara böldüğü ve uygulamaya özel alt bant işlemenin ardından yüksek hızlı geniş bant işaretin bir sentez süzgeç bankası kullanılarak yeniden üretildiği bir süzgeç bankası sistemidir. Uygulamalar arasında alt bant tabanlı kaynak kodlama ve frekans alanı eşitleme yer alır.

- **Çoklu erişimli programlanabilen çoğullayıcı yapılandırması**

Çoklu erişimli programlanabilen çoğullayıcı (Transmultiplexer, TMUX) yapılandırması çoklu taşıyıcı süzgeç bankası (Filter Bank Multi Carrier, FBMC) sistemlerinde uygulanır. Verici tarafında, QAM modülasyonlu alt taşıyıcılardan çok taşıyıcılı işaret üretmek için bir sentez süzgeç bankası kullanılır. Alıcı tarafında, bir analiz süzgeç bankası alt taşıyıcıları ayırmak için kullanılır.

- **Mükemmel yeniden yapılandırma**

PR TMUX veya analiz sentez sistemi, SFB ve AFB arka arkaya bağlantısında orijinal alt bant işaretlerini veya yüksek oranlı işareti tam olarak yeniden üretir. TMUX bağlamında, bu aynı zamanda bir dikey sistemdir. NPR süzgeç bankası sisteminde, yeniden yapılandırma iyi bir yaklaşım olarak elde edilir.

- **Paraüniter süzgeç bankası**

Sentez süzgeç bankası ve analiz süzgeç bankası için aynı prototip süzgeci kullanan ancak diğer tarafta karmaşık eşlenik formda olan bir PR sistemidir.

- **Biortogonal süzgeç bankası**

SFB ve AFB için farklı prototip süzgeçler kullanan bir mükemmel yeniden yapılandırma sistemidir. Süzgeç seçicilik gereksinimleri farklıysa bu tür tasarımlar yararlı olabilir ve süzgeç bankası sistemlerinin gecikme süresini azaltmaya da yardımcı olabilir [16].

1.5.1. Aşağı Örnekleme

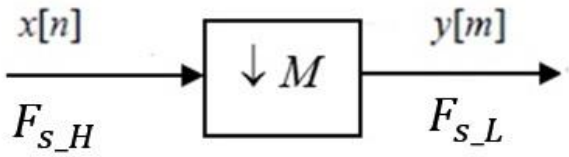
Aşağı örnekleme, örnekleme oranını azaltma işlemidir. Örnekleme hızı işaretin bant genişliğinden önemli ölçüde büyükse bir işareti aşağı örneklemek faydalı olabilir. Bir sayısal işaret işleme sistemini uygulamak için gereken hesaplama maliyetini veya ihtiyaç duyulan belleği azaltabilir. Bir M faktörü ile aşağı örnekleme işlemi Şekil 1.4'te gösterilmiştir, burada aşağı örnekleyici yüksek örnekleme oranı F_{s_H} ile bir giriş işareti $x[n]$ alır ve her M 'inci örneği tutarak ve geri kalanını atarak düşük örnekleme oranı F_{s_L} (çıkış dizisi) üretir. Bu aşağı örnekleme işlemi Eşitlik 1.8'deki gibi yazılabilir:

$$y[m] = x[n]\big|_{n=mM} n = 0,1,2,\cdots N; m = 0,1,2,\cdots \frac{N}{M} \tag{1.8}$$

F_{s_H} ve F_{s_L} örnekleme hızları arasındaki ilişki Eşitlik 1.9'daki gibi tanımlanabilir:

$$F_{s_L} = F_{s_H}/M \tag{1.9}$$

Şekil 1.4'te M aşağı örnekleme faktörü olarak adlandırılır ve basitçe giriş hızının çıkış hızına oranıdır.



Şekil 1.4. Aşağı örnekleyici

1.5.2. Yukarı Örnekleme

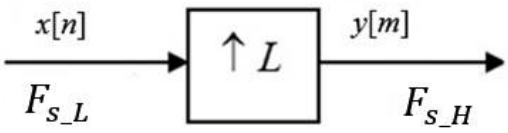
Yukarı örnekleme, orijinal örneklerin arasına sıfır değerli örnekler ekleyerek işaretin örnekleme hızını artırır. Yukarı örneklemenin ana nedeni, bir sistemin çıkışındaki örnekleme oranını artırmaktır, böylece daha yüksek bir örnekleme oranında çalışan başka bir sistem işareti işleyebilir.

Şekil 1.5'te bir işaretin L faktörü ile yukarı örneklenmesi işlemi gösterilmektedir, F_{s_L} düşük örnekleme oranına sahip bir giriş işareti $x[m]$ alır ve giriş işaretinin her örneği arasına $L-1$ sıfır ekleyerek F_{s_H} yüksek örnekleme oranına sahip çıkış dizisi $y[n]$ üretir. Yukarı örnekleme işlemi şu şekilde yazılabilir:

$$y[n] = x[m]|_{m=n/L} m \in Z, y[n] = 0 \; diğer \; durumlarda \qquad (1.10)$$

$$n = 0,1,2,\cdots N; m = 0,1,2,\cdots \frac{N}{L} \qquad (1.11)$$

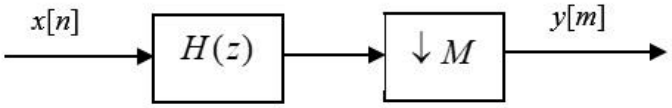
Bu durumda F_{s_L} ve F_{s_H} örnekleme hızları, çıkış hızının giriş hızına oranı olan bir üst örnekleme faktörü L ile ilişkilidir.



Şekil 1.5. Yukarı örnekleyici

1.5.3. Ayrıştırma

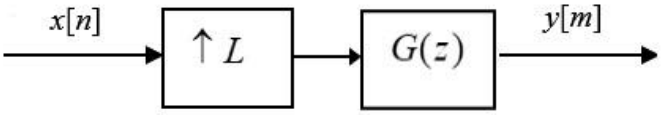
Ayrıştırma, bir işaretin etkin örnekleme oranını düşürmek için süzgeçleme ve alt örnekleme işlemidir. Eğer bir işareti sadece ara örnekleri atarak alt örnekleme yaparsak, örtüşme elde ederiz. Aşağı örneklemeden kaynaklanan bu örtüşmeyi önlemek için, şekil 1.6'da gösterildiği gibi aşağı örnekleyiciden önce bir örtüşme önleyici alçak geçiren süzgeç $H(z)$ kullanılır.



Şekil 1.6. Örtüşme önleyici süzgeç ve aşağı örnekleyiciden oluşan örnek azaltıcı

1.5.4. Ara değerleme

Ara değerleme, Şekil 1.7'de gösterildiği gibi etkin örnekleme oranını artırmak için bir işareti yukarı örnekleme ve süzgeçleme işlemidir. Yukarı örneklemeden kaynaklanabilecek ekstra spektral kopyaları önlemek için, yukarı örnekleyiciden sonra alçak geçiren süzgeç $G(z)$ kullanılır [17].



Şekil 1.7. Bir yukarı örnekleyici ve örtüşme önleyici süzgeçten oluşan ara değerleyici

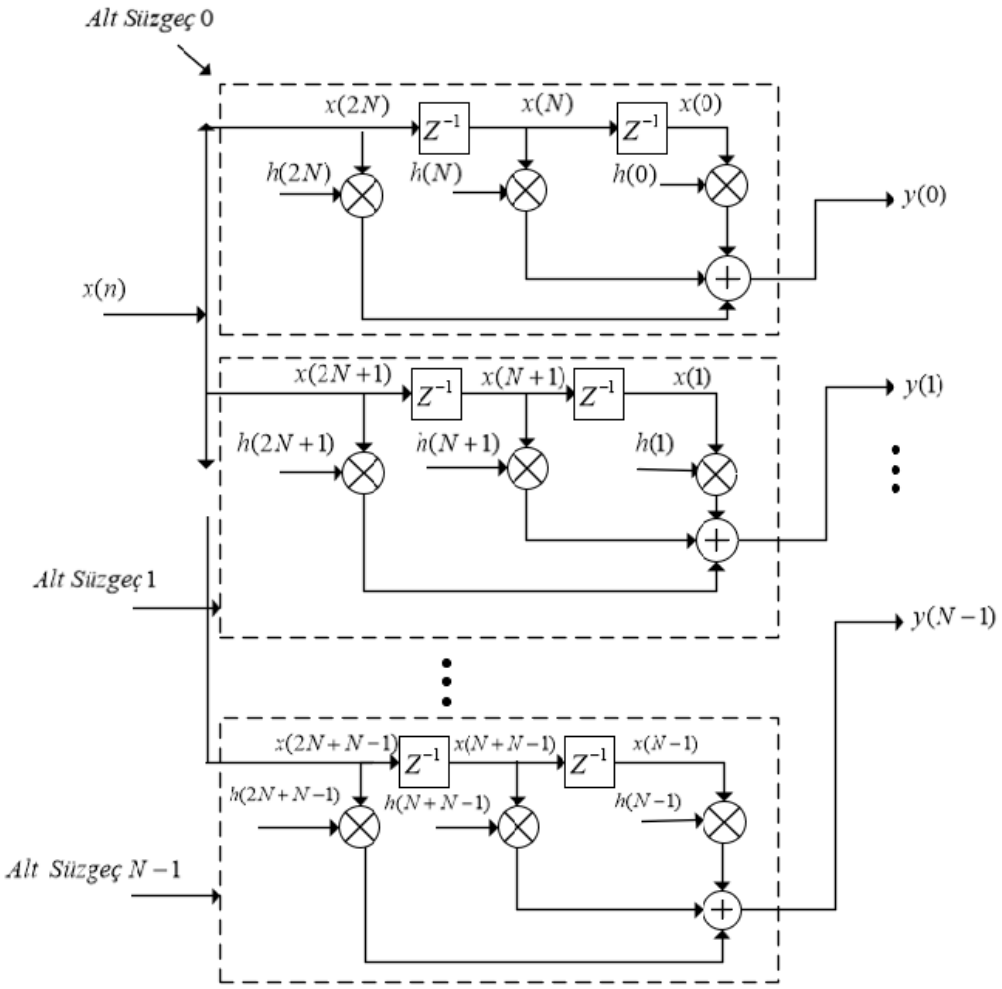
1.6. Çok Fazlı Süzgeç Bankası

PFB, bir giriş işaretinden alt bantları çıkarmak için bir DFT ile birleştirilmiş çoklu hızlı bir süzgeç yapısıdır. Bir işarete yeniden örnekleme ve süzgeçleme uygulamak için basit bir hesaplama yapısı içermektedir. Görüntü veya işaret işlemede, bir cihazın giriş işaretlerinde DFT yapması gerekir. Bir giriş işareti üzerindeki bu DFT'nin iki dezavantajı vardır, biri kaçak, ikincisi ise dalgalı kayıptır. PFB'ler, DFT'nin dezavantajlarının üstesinden gelmek için kullanılan bir tekniktir. PFB, kanal boyunca düz bir yanıt oluşturmak için kullanılır ve ayrıca bant dışı işaretlerin mükemmel şekilde bastırılmasını sağlar. PFB, bir prototip süzgeci modüle etmek için DFT'yi uygular ve DFT aşamalarının sayısından toplama ve süzgeçlenmiş zaman alanı verilerini hesaplar. Matematik olarak, ağırlıklandırma penceresi Eşitlik 1.12'deki gibi yazılabilir,

$$y(n) = \sum_{p=0}^{P-1} x(n+pN)h(n+pN) \qquad (1.12)$$

Burada $h(n+pN)$, P-kanallı çok fazlı alt süzgeçlere karşılık gelen alt süzgeç katsayılarıdır. Bu tür N çok fazlı alt süzgeç, DFT aşamasının sayısıyla birlikte topluca

PFB olarak adlandırılır. P =3 kanallı ve N alt süzgeçli çok fazlı süzgeç bankasının yapısı Şekil 1.8'de gösterilmektedir [18].



Şekil 1.8. Çok fazlı süzgeç bankası yapısı

Çok fazlı süzgeç yapıları FIR süzgeçler için etkin örnekleme hızı dönüştürücüler için geliştirilmiştir. Gerçi bu yapılar diğer uygulamalarda da kullanılabilir. Çok fazlı yapı herhangi bir sistem fonksiyonunun Eşitlik 1.13'teki gibi ayrıştırılabilir.

$$\begin{aligned} H(z) = \ldots + h(0) \quad & + h(M)z^{-M} + \ldots \\ \ldots + h(1)z^{-1} \quad & + h(M+1)z^{-(M+1)} + \ldots \\ & \vdots \\ \ldots + h(M-1)z^{-(M-1)} & + h(2M-1)z^{-(2M-1)} + \ldots \end{aligned} \tag{1.13}$$

İkinci adım olarak terimleri her bir *i* satır için çekilir ise

$$\begin{aligned} H(z) = \Big[&\ldots + h(0) && + h(M) z^{-M} \Big] \\ &+ z^{-1} \Big[\ldots + h(1) && + h(M+1) z^{-M} + \ldots \Big] \\ & && \vdots \\ &+ z^{-(M-1)} \Big[\ldots + h(M-1) + h(2M-1) z^{-M} + \ldots \Big] \end{aligned} \tag{1.14}$$

Eşitlik 1.14 elde edilir. Eşitlik 1.14 kısaltılarak aşağıdaki gibi Eşitlik 1.15'teki

$$H(z) = \sum_{i=0}^{M-1} z^{-i} P_i(z^M) \tag{1.15}$$

biçime dönüştürülebilir. Buradan $P_i(z)$ değeri Eşitlik 1.16'daki gibi elde edilir.

$$P_i(z) = \sum_{n=-\infty}^{\infty} h(nM+i) z^{-n} \tag{1.16}$$

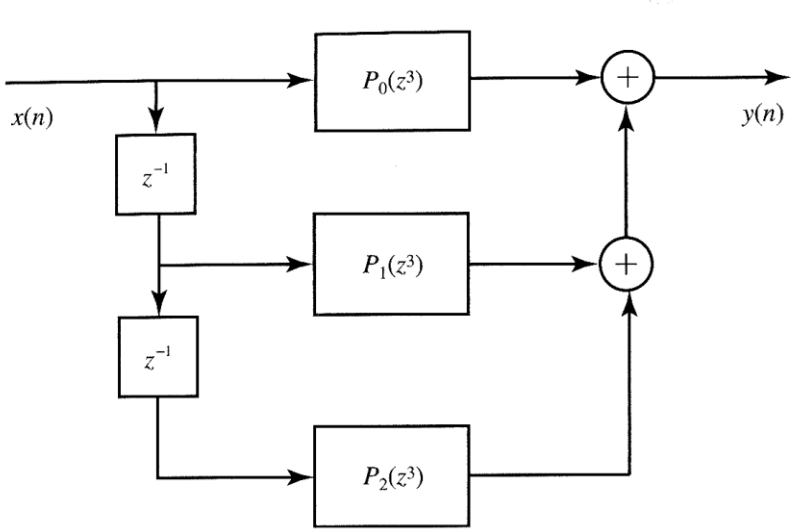
Eşitlik 1.15 ilişkisi *M* Bileşen çok fazlı ayrıştırma olarak adlandırılır ve $P_i(z)$ $H(z)$ 'nin çok fazlı bileşenleridir. Her bir alt dizi

$$p_i(n) = h(nM+i), \qquad i = 0,1,\ldots,M-1 \tag{1.17}$$

Orijinal dürtü tepkisinin geciktirilmiş yani faz ötelenmiş şeklinin örnek azaltılması yoluyla elde edilir. M bileşenli çok fazlı süzgeç yapısı elde etmek için, çıkış dizisinin *z*-dönüşümünü Eşitlik 1.15'te *M*=3 alınıp kullanılarak Eşitlik 1.18 elde edilebilir ve bu eşitlik Şekil 1.9'daki çok fazlı yapıya götürür.

$$\begin{aligned} Y(z) &= H(z)X(z) \\ &= P_0(z^3)X(z) + z^{-1}P_1(z^3)X(z) + z^{-2}P_2(z^3)X(z) \\ &= P_0(z^3)X(z) + z^{-1}\Big\{P_1(z^3)X(z) + z^{-1}\Big[P_2(z^3)X(z)\Big]\Big\} \end{aligned} \tag{1.18}$$

Benzer bir şekilde Eşitlik 1.18'deki son eşitlik Şekil 1.11'de gösterilen çok fazlı yapıyı üretir. Bu yapı, devrik FIR süzgeç gerçekleştirmeye benzediğinden, devrik çok fazlı yapısı olarak bilinir. Elde edilen çok fazlı yapılar FIR ya da IIR herhangi bir süzgeç için ve herhangi bir M değeri için geçerlidir. Şekil 1.9'da çok fazlı süzgeç yapısının blok şeması verilmektedir.



Şekil 1.9. Üç katlı çok fazlı süzgeç yapısı

1.6.1. Örnek Azaltıcı/Çoğaltıcılar

Genel olarak, örnekleme hızı dönüştürücüler ve doğrusal zamanla değişmeyen sistemlerin sıraları birbirleri ile değiştirilemez. Aşağıda, bir süzgeç ile bir örnek azaltıcı ya da örnek artırıcının pozisyonlarının değişimine, süzgecin uygun şekilde değiştirilmesi ile imkân sağlayan ve asil özdeşlikler olarak bilinen iki özdeşlik çıkartılacaktır. Şekil 1.10'da 3 katlı devrik çok fazlı süzgeç yapısı görülmektedir.

Eşitlik 1.19'u elde etmek için Şekil 1.11'deki yapıya göre bir örnek azaltıcının giriş/çıkış tanımından

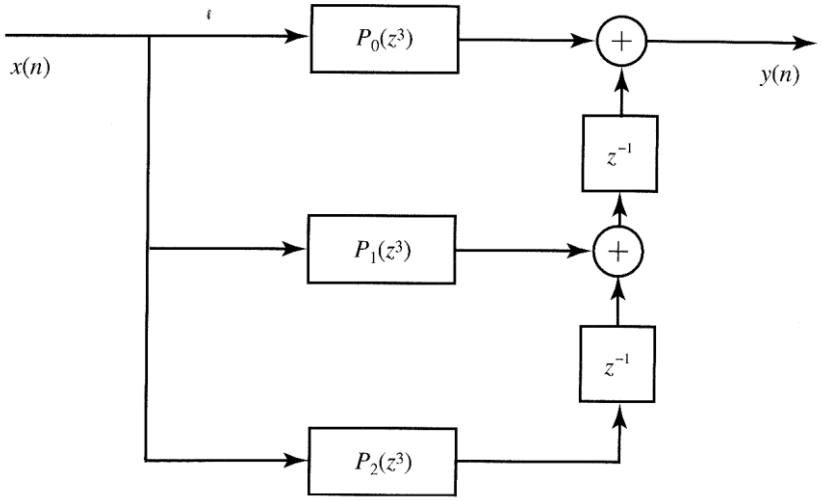
$$y(n) = x(nD) \xleftrightarrow{Z} Y(z) = \frac{1}{D}\sum_{i=0}^{D-1} X(z^{1/D} W_D^i) \qquad (1.19)$$

olduğu görülmektedir. Burada $W_D = e^{-j2\pi D}$ 'dir. Şekil 1.11-a'da birbirine denk iki örnek azaltıcı sistem görülmektedir. Burada gösterilen sistem çıkışı $V_1(z) = H(z^D)X(z)$ olduğundan Eşitlik 1.20 yazılır.

$$Y(z) = \frac{1}{D}\sum_{i=0}^{D-1} V_1(z^{1/D} W_D^i) = \frac{1}{D}\sum_{i=0}^{D-1} H(zW_D^{iD}) X(z^{1/D} W_D^i) \qquad (1.20)$$

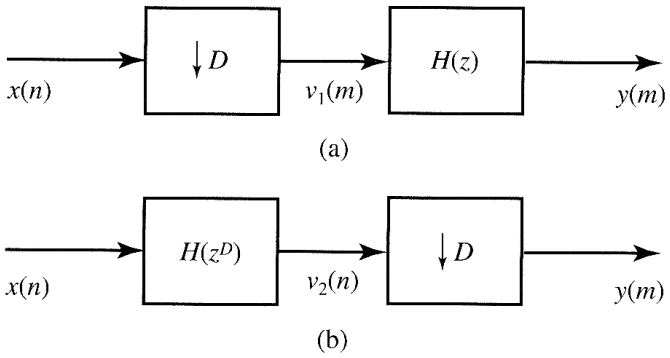
$W_D^i = 1$ olduğu ve Şekil 1.11-b'de birbirine denk iki örnek azaltıcı sistem görülmektedir, bu yapı göz önüne alınırsa Eşitlik 1.20 Eşitlik 1.21'deki gibi yazılabilir.

$$Y(z) = \frac{1}{D} H(z) \sum_{i=0}^{D-1} X(z^{1/D} W_D^i) = H(z) V_2(z) \qquad (1.21)$$

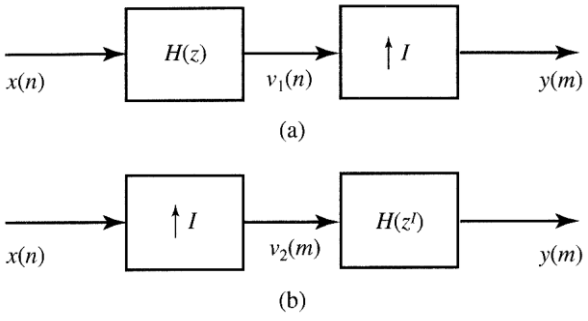


Şekil 1.10. Üç katlı devrik çok fazlı süzgeç yapısı

Bu eşitliklerden Şekil 1.12'de verilen iki yapının denk olduğu açıkça görülmektedir.



Şekil 1.11. Birbirine denk iki örnek azaltıcı sistem



Şekil 1.12. Birbirine denk iki örnek artırıcı sistem

Benzer bir özdeşliğin örnek artırma için doğru olduğu gösterilebilir. Bir örnek artırıcının giriş/çıkış tanımlaması,

$$y(n) = \begin{cases} x(\frac{n}{I}), & n = 0, \pm I, \pm 2I, \ldots \\ 0, & \end{cases} \xleftrightarrow{z} Y(z) = X(z^I) \qquad (1.22)$$

şeklinde yazılabilir. Şekil 1.12-a'da birbirine denk iki örnek artırıcı sistem görülmektedir ve bu sistemin çıkışı $V_I(z) = X(z^I)$ olduğu için $Y(z)$

$$Y(z) = H(z^I)V_1(z) = H(z^I)X(z^I) \qquad (1.23)$$

gibi yazılabilir. Şekil 1.12-b'de verilen sistemin çıkışı

$$Y(z) = V_2(z^I) = H(z^I)X(z^I) \qquad (1.24)$$

şeklindedir ve bu Eşitlik 1.23'e eşittir. Bu Şekil 1.12'de görülen birbirine denk iki örnek artırıcı sistemi göstermektedir.

Bu bilgiler doğrultusunda, doğrusal süzme ve örnek azaltıcı ya da örnek artırıcı sıralarının, süzgecin sistem fonksiyonu uygun şekilde düzenlenir ise, değiştirilmesinin mümkün olduğu gösterilmiş bulunmaktadır [7, 19].

1.6.2. Optimizasyon Algoritmaları ve Çok Fazlı Süzgeç Bankası Tasarımı

PFB tasarımında kullanılan alt süzgeçler için önceden belirlenen genlik ve frekans cevabı özelliklerine göre tasarım yapmak için farklı yöntemler mevcuttur. Fourier serilerine bağlı olan pencereleme teknikleri, dönüşüm teknikleri ve optimizasyon teknikleri en çok tercih edilen süzgeç tasarım yaklaşımlarıdır [20]. PFB tasarımı optimizasyonu yaklaşımında amaç, arzulanan özelliklere sahip süzgecin olası tasarımları arasından en iyisini bulmaktır. En iyiyi bulma işlemi yapılırken önce arzulanan sisteme ait bir model kurulmakta, ardından da bu model, en iyi çözümü üretmek için kullanılmaktadır. Çalışmalar ve uygulamalar üzerine yapılan incelemelere göre tasarımlar genellikle, PFB tasarımının optimizasyonunda donanımsal ve yazılımsal iyileştirme olmak üzere iki şekilde yapılmaktadır. Bu nedenle, çok fazlı süzgeç bankası tasarımlarının uygulanabilirliği için bu problemleri çözebilecek uygun optimizasyon algoritmalarının kullanımına ihtiyaç vardır. Bu algoritmalar bilgisayarlı hesaplamada ve problem çözümünde zaman ve hız bakımından kolaylık sağlamaktadır. PFB tasarımında düşük güç tüketimi, daha az donanım maliyeti ve tasarım için harcanan sürenin kısa olması, yüksek

hesaplama hızı gibi önemli özellikleri bulunması istenir. Sezgisel algoritmalar sahip oldukları küresel araştırma yetenekleri sayesinde çok modlu yani çok parametreli hata yüzeylerinde küresel minimumlara veya yakındaki çözümlere yüksek yakınsama hızları ile ulaşabilmekte ve arzu edilen sistem tasarımlarını etkin şekilde gerçekleştirebilmektedirler. Sezgisel algoritmalar kendi içerisinde popülasyon tabanlı, iteratif tabanlı, stokastik ve deterministik tabanlı algoritmalar olmak üzere farklı gruplar içerisinde değerlendirilmektedir. Türeve dayalı algoritmalar tek modlu problemlerin çözümünde etkin olmakla birlikte, düşük yakınsama hızı ve adım büyüklüğü parametreleri nedeniyle etkinliğini dezavantaja çevirmektedirler.

1.6.3. Çok Fazlı Süzgeç Bankası Tasarımında Sezgisel Algoritmaların Kullanılması

PFB tasarımlarında istenen giriş ve çıkış ilişkisini verecek çok sayıda alternatif çözüm bulunmaktadır. Bu çözümlerin içerisinden en iyisinin en kısa sürede bulunması optimizasyon algoritmalarının kullanılmasıyla elde edilebilmektedir. Son yıllarda, doğada var olan canlıları ve onların ilişkilerini temel alan ve sezgisel algoritmalar olarak da adlandırılan optimizasyon algoritmaları geliştirilmiştir. Bu algoritmalar, klasik algoritmalardan farklı olarak, bilimin çoğu alanında her tür optimizasyon problemine uygulanabilmekte, esnek ve yüksek performans özellikleri göstermekte, çok modlu problemler için etkin şekilde sonuç vermekte ve problemlerin modellenmesindeki matematik yükü hafifletmektedirler [21]. Popülasyon tabanlı yani tanımlanan bir probleme ait çözüm kümesi üzerinde sürekli yeni ve bir öncekinden daha iyi çözümler geliştiren algoritmalar araştırmacılar tarafından çok daha fazla ilgi görmüştür. Popülasyon tabanlı algoritmalar da kendi içerisinde evrimsel tabanlı ve sürü tabanlı algoritmalar olmak üzere incelenmektedirler [22]. Sezgisel algoritmaların yakınsama hızı, doğruluk, hassasiyet, esneklik ve hesaplama maliyeti gibi ölçütler açısından birbirinden farklı oldukları bilinmektedir. Bu farklılıklar, literatürde çeşitli bilim alanlarında karşılaşılan optimizasyon problemlerinin çözümlerinde bu algoritmaların performanslarının karşılaştırılmasına neden olmaktadır.

Günümüze kadar, araştırmacılar sayısal süzgeç ve sayısal süzgeç bankası tasarımları için birçok meta sezgisel algoritmayı kullanmışlardır [23, 24]. Bu tasarım problemleri, genetik algoritma (Genetic Algorithm, GA), ısıl işlem algoritması (Simulated Annealing, SA), tabu araştırma algoritması (Tabu Search, TS), yapay bağışıklık algoritması (Artificial

Immune, AI), parçacık sürü optimizasyon algoritması (Particle Swarm Optimization, PSO), diferansiyel gelişim algoritması (Differential Evolution, DE), armoni arama algoritması (Harmony Search, HS), klonal seçim algoritması (Clonal Search Algorithm, CSA), ABC algoritması, arı algoritması (Bees Algorithm, BA), mABC algoritması, qABC algoritması, yerçekimi arama algoritması (Gravitational Search Algorithm, GSA), yarasa algoritması (Bat Algorithm, BA), bakteriyel toplayıcılık optimizasyon (Bacterial Foraging Optimization, BFO) algoritması ve bahsedilen bu algoritmaların hibrit veya iyileştirilmiş versiyonlarını içermektedirler [25-44]. Bahsedilen bu sezgisel algoritmalar popülasyon tabanlı olmakla beraber, evrimsel tabanlı ve sürü tabanlı algoritmalar olmak üzere ikiye ayrılabilmektedirler. ABC ve PSO algoritması gibi algoritmalar sürü tabanlı algoritmalardır ve doğada yaşayan canlı topluluklarından ve bunların sosyal davranışlarından esinlenilerek ortaya çıkmışlardır. GA ve DE algoritması gibi algoritmalar ise evrimsel tabanlı algoritmalardır ve evrimsel yaşamdan ve süreçlerden esinlenilerek ortaya çıkmışlardır.

Çok fazlı süzgeç bankaları, modern sayısal işaret işlemede önemli bir yere sahiptir ve geniş bantlı işaretlerin alt bantlara ayrılmasında ya da birden çok işaretin tek bir bantta birleştirilmesinde kilit bir rol oynarlar. Bu süzgeç bankaları, telekomünikasyon, radyo frekansı işleme ve ses kodlama gibi çeşitli uygulamalarda yaygın olarak kullanılmaktadır. Optimal süzgeç bankası tasarımı, genellikle genlik ve faz yanıtlarının yanı sıra, grup gecikmesi ve enerji dağılımı gibi çeşitli performans kriterlerine bağlıdır. Bu karmaşık çok amaçlı optimizasyon problemi, sezgisel algoritmaların kullanımını gerektirir, çünkü geleneksel yöntemler genellikle yerel optimumlara takılıp kalabilir veya hesaplama açısından çok maliyetli olabilir. Sezgisel algoritmalar, özellikle donanım uygulamalarında kısıtlı kaynakların etkin bir şekilde kullanılmasını sağlamak için kullanılmaktadır. Literatürde genellikle PFB tasarımında, donanımsal tasarım ve yazılımsal iyileştirme ile ilgili yapılan çalışmalar öne çıkmaktadır. Tez çalışmasında literatüre yeni bir yaklaşım modeli olarak ABC, qABC, mABC, PSO ve GA algoritmalarına dayalı PFB tasarımı önerilmiştir. Bu meta sezgisel algoritmaların performansı birbirleriyle karşılaştırılarak detaylı bir şekilde incelenmiştir.

Alt bölümde konuyla ilgili gerçekleştirilen literatür çalışmaları hakkında bilgi verilmektedir.

Wu Ying ve arkadaşları, 2008 yılında çok fazlı süzgeçlere dayalı yüksek verimli bir yönlendirilmiş algoritma sunmakta ve deneysel sonuçlar önerilen modelin doğruluğunu

teyit etmektedir. Bu makalede, çok fazlı yönlendirilmiş algoritma simüle edilmiş, avantaj ve dezavantajları analiz edilmiştir. İlk olarak, kör bölge problemini hedef alarak, çoklu azaltılmasına dayanarak, algoritma geliştirilmiş, kör bölge olmayan yönlendirilmiş alıcıların matematiksel bir modeli türetilmiştir. Daha sonra, modelden dört kanallı alıcıları gerçekleştiren bir simülasyon sistemi tasarlanmıştır. Tüm deneysel sonuçlar önerilen modelin doğru olduğunu, çalışma sonuçlarının fizibilitesini ve kullanılabilirliğini doğrulamaktadır [45].

Mark L. Fowler ve arkadaşları, 2011 yılında DSP'nin matematiksel olarak ağır bir konu olduğunu ve öğrencilerin bunu tam olarak anlayabilmeleri için sayısal işaret işleme konularının altında yatan matematiksel gelişmeleri anlamaları gerektiğini incelemiştir. Ancak, yalnızca matematiksel gelişmelere güvenmek genellikle bir sonucun temelinin gerçek doğasını bulanıklaştırır. Aritmetiğe hâkim olan öğrencilerin bir konunun temel fikirlerini gerçekten kavramaması mümkündür. Dahası, DSP konularını sadece "matematik üzerinden giderek" öğretmek, öğrencileri onları iyi araştırmacılar yapacak keşif sanatını öğrenmekten mahrum bırakır. Bu makale örnek olarak çok fazlı yok etme ve ara değerleme konusunu kullanmaktadır [46].

P. P. Vaidyanathan ve arkadaşları, 2013 yılında bir analog sentez analiz süzgeç bankasını ele almış ve model işaret $x(t)$'nin sonlu bir yenilik oranına sahip olduğu bilgisine dayanarak bunun nasıl sayısal bir süzgeç bankasına dönüştürüleceğini göstermiştir. Bu dönüşüm, aşırı örnekleme fikrine dayanmaktadır. Bunun avantajı, tüm tasarımların ve uygulamaların sayısal süzgeç bankası teorisinden faydalanacak olması ve yarı sonsuz matrislerin söz konusu olmamasıdır. Aşırı örnekleme sadece uygun bir teorik gerekçedir ve pratikte örneklemeden önce uygulanır, gerçekte aşırı örnekleme söz konusu değildir. Bu durumun bir avantaj olabileceği bu çalışmada gösterilmiştir [47].

Shilian Zheng ve arkadaşları, 2014 yılında gerçekleştirdikleri çalışmada grup testi fikrine dayanan geniş bantlı bir spektrum algılama prosedürü önermiştir. Çıkış alt bantlarının aynı bant genişliğine sahip olduğu ve sürekli frekans bileşenleri içeren yeni bir PFB önerilmiştir. Bu PFB, iki aşamalı bir geniş bant spektrum algılama yöntemi oluşturmak için kullanılmıştır. Bilişsel radyo, birincil kullanıcılara lisanslı mevcut kullanılmayan spektruma fırsatçı bir şekilde erişerek spektrum yetersiz kullanım sorununa bir çözüm olarak önerilmiştir. Bilişsel radyolar spektrumun ikincil kullanıcıları olarak kabul edildiğinden, birincil kullanıcılara zararlı girişimlere neden olmamalıdırlar. Bu hedefe ulaşmak için, ikincil kullanıcıların iletişim için spektrum boşluklarını tespit etmek üzere

geniş bir spektral aralığı algılamaları gerekir. Bu süreç geniş bant spektrum algılama olarak adlandırılır. Simülasyonlar, işgal edilen birincil kanal sayısı düşük olduğunda önerilen yöntemin ikili test sayısını büyük ölçüde azaltabileceğini göstermektedir [48].

Ling, Bingo Wing-Kuen Ho ve arkadaşları, 2014 yılında, optimizasyon problemini çözmek için ortak bir kısıtlama dönüşümü ve değiştirilmiş dolgulu fonksiyon yöntemi kullanarak, süzgeç bankasının maksimum büyüklük bozulmasına ve hem geçiş bandı hem de durdurma bandı üzerindeki prototip süzgecin maksimum dalgalanma büyüklüklerine ilişkin özelliklere tabi olarak süzgeç bankasının toplam örtüşme hatasını en aza indirerek elde edilen, mevcut tasarımlardan daha iyi performans gösteren optimum bir kosinüs modülasyonlu tekdüze olmayan doğrusal fazlı FIR süzgeç bankası tasarımı sunmaktadır. Çalışmada araştırmacılar kosinüs modülasyonlu, düzgün olmayan doğrusal fazlı FIR bir süzgeç bankası tasarlamaktadır. Tüm analiz süzgeçlerinin ve süzgeç bankasının sentez süzgeçlerinin frekans yanıtları, tek bir prototip süzgecin frekans yanıtının hem esnetilmesi hem de kaydırılması temelinde türetilmiştir. Süzgeç bankasının toplam örtüşme hatası, süzgeç bankasının maksimum büyüklük bozulmasına ve prototip süzgecin hem geçiş bandı hem de durdurma bandı üzerindeki maksimum dalgalanma büyüklüklerine ilişkin özelliklere tabi olarak en aza indirilir. Bu makale, optimizasyon problemini çözmek için ortak bir kısıtlama dönüşümü ve değiştirilmiş doldurulmuş fonksiyon yöntemi önermektedir. Özellikle, fonksiyonel eşitsizlik kısıtları, kısıt dönüştürme yöntemi ile ayrık kısıtlara dönüştürülür. Konveks olmayan optimizasyon probleminin küresel optimal çözümü, modifiye edilmiş doldurulmuş fonksiyon yöntemi ile verimli bir şekilde bulunabilir. Bilgisayar sayısal simülasyon sonuçları, tasarımın mevcut tasarımlardan daha iyi performans gösterdiğini ortaya koymaktadır [49].

Fred Harris ve arkadaşları, 2015 yılında, çok fazlı süzgeç bankalarının kullanımına dayanan geniş bant genişlikli süzgeçleri uygulamak için verimli mimarilerin bir incelemesini sunmuştur. Daha önceki bir seçenek, değişken bant genişlikli süzgeçleri uygulamak için iki katmanlı birçok fazlı süzgeç kullanırken, daha yeni seçenek, süzgeç boyunca azaltılmış gecikme ile değişken bant genişlikli süzgeçleri uygulamak için serpiştirilmiş çok fazlı süzgeç bankalarını kullanır. Hassas bant genişliği ayarı olmadan ve iç katman işlemenin kullanıldığı veya serpiştirilmiş işlemenin kullanıldığı hassas bant genişliği ayarlı mükemmel yeniden yapılandırmalı, maksimum düzeyde azaltılmamış süzgeç bankalarının karışımları, verimli çok çözünürlüklü geniş bant yazılım tanımlı radyolar oluşturmak için esnek mimariler sunar. Bu tasarımların tasarım parametreleri

arasında Ters Hızlı Fourier Dönüşümü (Inverse Fast Fourier Transform, IFFT) boyutu, seçilebilir geçiş bant genişliği ve dalgalanma performansı ile ilgili uzunluğa sahip çok fazlı süzgeç, iç katman süzgeci veya çok fazlı iç katman süzgeci seçeneği ve değişken bant genişliği kanalizatörleri bulunmaktadır. Tüm bu seçenekler, belirli bir süzgeçleme görevini gerçekleştirmek için önemli ölçüde daha az işlem gücü gerektirerek çevreci olduklarını kanıtlamaktadır [50].

Rahul M. Deshmukh ve arkadaşları, 2015 yılında çoklu hızlı süzgecin birçok DSP uygulamasında yaygın olarak kullanıldığını sunmuştur. DSP sisteminin karmaşıklığını azaltmak için çok verimli mimariler tasarlanmıştır. Toplayıcı ve çarpıcı, süzgecin alan, güç ve gecikme parametrelerinin azaltılmasına katkıda bulunan süzgecin ana temel bloklarıdır. Bu çalışma, atlamalı besleme doğrudan çarpanı kullanan çok fazlı FIR süzgeci ve kaydırma ve ekleme çarpanı kullanan çok fazlı FIR süzgeç sunmaktadır. Karşılaştırma yapıldığında, önerilen tasarımın geleneksel tasarıma göre alan ve gecikme açısından daha verimli olduğu görülmüştür [51].

Yaprak Eminağa ve arkadaşları, 2015 yılında Elektrokardiyogram (EKG) toplama sistemlerinde kullanılabilecek düşük karmaşıklığa sahip yüksek verimli bir örnekleme oranını azaltıcı süzgeci literatüre sunmuşlardır. Çalışmada EKG işaret toplama sistemleri için tüm geçiş tabanlı çok fazlı IIR süzgeçleri kullanan yüksek verimli bir ayrıştırma süzgecinin tasarımı ve uygulaması ele alınmaktadır. Önerilen ayrıştırma süzgeci dört aşamadan oluşmaktadır. Tasarım, süzgeç katsayılarının ve çarpımların sayısını azaltarak düşük güç tüketimi elde etmeyi, böylece donanım karmaşıklığını ve güç tüketimini düşürmeyi amaçlamaktadır. Önerilen ayrıştırıcının performansı, daha düşük güç tüketimi ile daha iyi performans gösteren bir ayrıştırıcı ile karşılaştırılmıştır. Simülasyon sonuçları, EKG işaretlerinin yüksek verimlilik ve düşük donanım karmaşıklığı ile işlenmesinde önerilen ayrıştırma süzgecinin etkinliğini doğrulamaktadır. Performans, bu alanda en sık başvurulan veri tabanlarındaki kalp durumu bilgilerinin işlenmesiyle doğrulanmış ve karşılaştırmalı simülasyonlar gerçekleştirilmiştir [52].

Zhao Kongrui ve arkadaşları, 2015 yılında donanımdaki işlem hızının sınırlandırılması nedeniyle, geleneksel aşağı dönüşümün GHz yüksek hızlı örnekleme işareti için işe yaramadığını incelemiştir. Bu makalede, GHz yüksek hızlı örnekleme işaretinin aşağı dönüşümünü sağlamak için çok fazlı ayrışımı ile yeni bir yaklaşım önerilmiştir. Bu yöntemde, giriş yüksek hızlı örnekleme işareti ve karşılık gelen salınım işareti, çok fazlı

ayrıştırma olarak görülebilen gecikme ve ayrıştırma yaklaşımı ile düşük hızlı örneklemenin *M* alt kanalı olacak şekilde paralelleştirilir. Aşağı dönüştürme işlemi giriş verisinin her kanalı için gerçekleştirilir. Daha sonra yüksek hızlı örnekleme işaretinin aşağı dönüşümü, her kanalın aşağı dönüşüm sonuçlarıyla birlikte gelir. Simülasyonlar önerilen yöntemin etkinliğini doğrulamaktadır [53].

Baderia, K. ve arkadaşları, 2015 yılında optimize edilmiş çok fazlı bileşenler ve küresel sürü optimizasyonu tekniklerini kullanarak, çeşitli performans özellikleriyle değerlendirilen ve yakın zamanda geliştirilen tekniklere göre üstünlük gösteren iki kanallı bir çeyrek ayna süzgeç bankasının optimum tasarımını önermektedir. Bu makale, çok verimli küresel sürü optimizasyon teknikleri guguk kuşu arama (Cuckoo Search, CS) ve modifiye guguk kuşu arama (Modified Cuckoo Search, MCS) tabanlı kesirli türev kısıtlamalarını zorlayan optimize edilmiş çok fazlı bileşen kullanarak iki kanallı çeyrek ayna süzgeç (Quadrature Mirror Filter, QMF) bankasının optimum tasarımını önermektedir. Amaç fonksiyonu, frekans alanında, geçiş bandı ve durdurma bandındaki hataların enerjisinin toplamı olarak yapılandırılmıştır. Bu amaç fonksiyonu, CS ve MCS tabanlı kesirli türev kısıtlamalarını ve iki kanallı QMF bankasının mükemmel yeniden yapılandırma koşulunu bir kısıtlama olarak zorlayan Lagrange çarpanı yöntemi olarak adlandırılan iyi bilinen bir gerçek en küçük kare yöntemi kullanılarak optimize edilir [54].

Tong Liu ve arkadaşları, 2016 yılında Sayısal Aşağı Dönüşümün (Digital Down Converter, DDC), DSP cihazları tarafından daha sonra gerçek zamanlı işlemeyi kolaylaştırmak için yüksek frekanslı veri akışlarını düşük frekanslı veri akışlarına dönüştüren yazılım radyosu için anahtar teknolojilerden biri olduğunu sunmuştur. Bu makale, çok fazlı süzgeçlemeye dayalı bir sayısal frekans dönüştürme yöntemi sunmaktadır. Yüksek hızlı sayısal frekans dönüşümü altında Alan Programlanabilir Kapı Dizisi (Field Programmable Gate Array, FPGA) kullanılarak uygulanmış ve Sayısal Kontrollü Osilatör (Numerically Controlled Oscillator, NCO), frekans karıştırma modülü, kademeli entegratör tarak (Cascaded Integrator–Comb, CIC) süzgeç, yarım bant (Half-Band, HB) süzgeçleri ve FIR süzgeç dahil olmak üzere ayrıntılı olarak tartışılmıştır [55].

Gopal S. Gawande ve arkadaşları, 2016 yılında, çoklu hızlı süzgeçleme tekniğinin farklı sistemlerin örnekleme oranlarını karşılamak için yaygın olarak kullanıldığını ve DSP'nin sayısal süzgeçlerin düşük maliyetli uygulamalarına neden olan güçlü bir teknik olduğunu sunmuştur. Çok fazlı ayrıştırma tekniği, çok katmanlı sayısal süzgeçlerde paralelliği benimseyerek hesaplama karmaşıklığını azaltır. Güç tüketimi genellikle yeniden

yapılandırılabilir donanım üzerinde çok katmanlı sayısal süzgeçler uygularken önemli bir biçim kısıtlamasıdır. Sayısal uygulamalardaki dinamik güç tüketimi, bütünleyici metal oksit yarı iletken (Complementary Metal Oxide Semiconductor, CMOS) kapılarının çıkışındaki anahtar aktivitesini en aza indirerek azaltılacaktır. Çalışmaların çoğu, sıfır olmayan her bit donanım uygulamasında fazladan bir toplayıcıya karşılık geldiğinden, süzgeç katsayıları içindeki 2'nin kuvveti veya sıfır olmayan terimlerin aralığını en aza indirgemek etrafında dönmektedir. Bu çalışma, süzgeç katsayıları içindeki sıfır olmayan terimlerin miktarını azaltarak CMOS kapısının çıkışındaki anahtar etkinliğini azaltmak için tamamen benzersiz bir kural önermektedir. Önerilen algoritma çok fazlı yapılara uygulanmış ve dinamik güç tüketimi üzerindeki etkisi analiz edilmiştir [56].

David B. H. Tay ve arkadaşları, 2016 yılında aşağı örneklemeli alanda, oldukça hassas örneklenmiş iki parçalı grafik süzgeç bankaları için hesaplama açısından verimli olan çok fazlı süzgeç yapısını önermişlerdir. Aşağı örneklenmiş alandaki işaretlerin eşdeğer alt grafikler üzerinde tanımlandığı ve süzgeçlemenin bu alt grafiklere göre spektral süzgeçler olarak tanımlanabileceği gösterilmiştir. Çok fazlı analizden elde edilen sonuçlar, grafik süzgeç bankalarının tek boyutlu süzgeç bankaları ile benzerliklerini ortaya koymaktadır, ancak bazı farklılıklar da mevcuttur. Teorik sonuçlar, süzgeç bankasının iki parçalı yönlendirilmiş grafiklere ve daha genel taban matrislerine sahip süzgeçlere genelleştirilmesine izin vermiştir. Alternatif taban matrislerinin kullanımının doğrusal olmayan yaklaşım uygulamalarında iyileşmeye yol açabileceği de gösterilmiştir. Gelecekteki çalışmalar, rastgele yönlendirilmiş bir grafiği bir dizi iki parçalı yönlendirilmiş grafiğe ayrıştırmak için teknikler geliştirmeyi içerebilir [57].

Yingying Du ve arkadaşları, 2016 yılında bant geçiren örnekleme yapısına dayanan sayısal aşağı dönüştürme teknolojisinin yazılım radyo alıcı sisteminin temel teknolojilerinden biri olduğunu sunmuştur. Çarpan kullanan geleneksel sayısal karıştırma dikey demodülasyonu, bir dereceye kadar hesaplama karmaşıklığını artırmaktadır. Bu çalışma, çok fazlı süzgeçleme yapısına dayanan ve hesaplama karmaşıklığını etkili bir şekilde azaltan bir tür sayısal aşağı dönüştürme tasarımı önermektedir. İlk olarak, tek-çift çıkarma ve işaret düzeltme işlemi gerçekleştirilir; ikinci olarak, güncelleme için zaman geciktirme süzgeçleri kullanılır; son olarak, temel bant işareti çıkarıldıktan sonra çıkış olarak verilir. Uygulanabilirliğini kanıtlamak için yazılım simülasyonu uygulanmıştır. Sayısal aşağı dönüştürme, çok fazlı süzgeçleme yapısı benimsenerek tamamlanır; çıkış veri oranı on defa çıkarıldıktan sonra azaltılır. Bu, ana bant işlemenin baskısını etkili bir

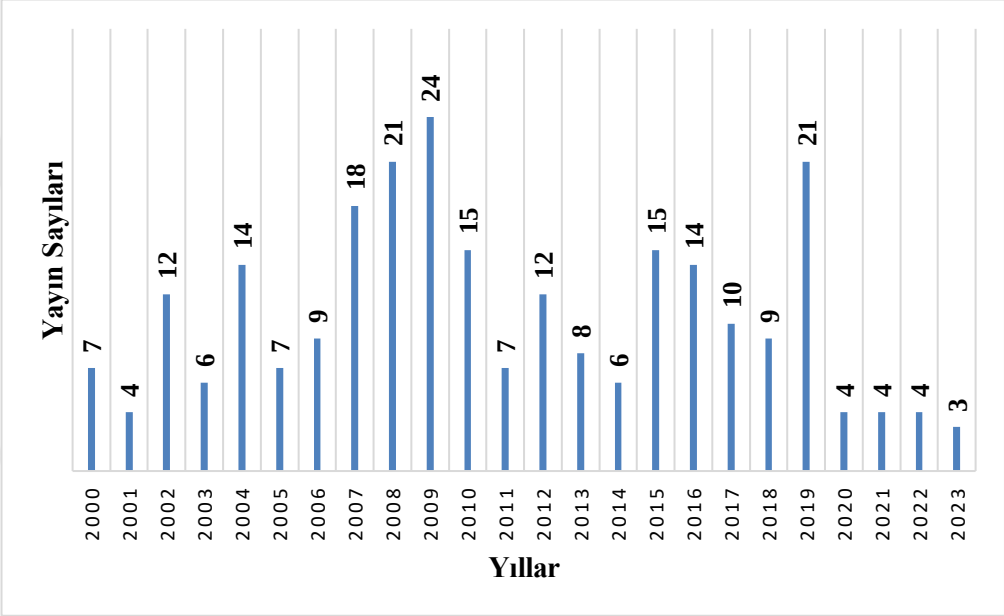
şekilde hafifletebilir ve sistem kaynak tüketimleri nispeten düşüktür, bu nedenle tasarım şeması pratiktir [58].

Yuichi Tanaka ve arkadaşları, 2017 yılında spektral grafik dalgacıklarının ve süzgeç bankalarının çok fazlı bir yapısını ele almıştır. Burada iki kanallı kritik örneklemeli grafik süzgeç bankaları ele alınmıştır. Klasik işaret işlemede, süzgeç bankalarının çok fazlı yapısı çok kullanışlıdır, çünkü aşağı örnekleyici ve yukarı örnekleyici analiz süzgeçlemesinden önce veya sentez süzgeçlemesinden sonra yerleştirilebilir. Bu çalışma, benzer bir yapının spektral grafik süzgeç bankaları için de mümkün olduğunu teorik olarak göstermektedir. Bu yapı, altta yatan grafik iki parçalı olduğu sürece herhangi bir iki kanallı kritik örneklemeli spektral grafik süzgeç bankası için kullanılabilir. Klasik işaret işlemeye benzer şekilde, aşağı örnekleme ve yukarı örnekleme operatörlerini sırasıyla analiz ve sentez süzgeçleme işlemlerinden önce ve sonra hareket ettirebilir. Alçak geçiren ve yüksek geçiren kanallardaki dönüştürülmüş katsayıların sayısı $N/2$ olmak zorunda değildir [59]. Şekil 1.16'da çok fazlı süzgeç tasarımı hakkında 2000-2023 yılları arasında yapılan yayınların kullanılan yöntemlere göre dağılımları görülmektedir. Şekil 1.16'da incelendiğinde çok fazlı süzgeç tasarımı hakkında 2000-2023 yılları arasında yapılan yayınların kullanılan yöntemlere göre dağılımında donanımsal iyileştirme yöntemlerinin kullanımının yazılımsal iyileştirme yöntemlerinin kullanımından üç kattan fazla olduğu görülmektedir.

Loğoğlu ve arkadaşları 2022 yılında yapay arı koloni ABC algoritması kullanarak PFB tasarımına yeni bir yaklaşım gerçekleştirmişlerdir. PFB'ler, DSP ve haberleşmede alt bant kodlama için en çok tercih edilen çok katmanlı yapılardır. PFB tasarımı için süzgeç derecesi ve frekans seçiciliği gibi birçok önemli tasarım parametresi vardır. Ayrıca, tasarımlarda istenen frekans cevabını gerçekleştirmek için, durdurma bandı ve geçiş bandı zayıflaması büyük önem taşımaktadır. PFB tasarımında araştırmacılar ve uygulayıcılar sıklıkla iteratif ve meta sezgisel optimizasyon yöntemlerini kullanmaktadır. Sezgisel teknikler sürekli ve ayrık çözüm uzayında önemli bir problem çözme yeteneğine sahiptir. Bu nedenle, önerilen yöntem diğer yöntemlerden daha iyi sonuçlar vermiştir ve algoritmanın performansı kontrol parametrelerine bağlıdır. Yazarlar bu çalışmada, önerilen PFB tasarımı probleminin çözümü için, ABC algoritması kullanmışlardır. İlk aşamada, önerilen PFB tasarımının performansını artırmak için ABC algoritmasının kontrol parametreleri incelenmişlerdir. İkinci aşamada, 8-256 arasında alınan farklı süzgeç dereceleri ve farklı geçirme bantları için analizler gerçekleştirilmiştir. Elde edilen

tüm sonuçlar PSO ve GA algoritmaları ile de karşılaştırılmıştır. Son olarak, farklı süzgeç dereceleri ve frekansları için ABC algoritması tarafından elde edilen en iyi sonuçlara göre çok fazlı süzgeç bankasının sayısal işaret işleme uygulaması gerçekleştirilmiştir [60].

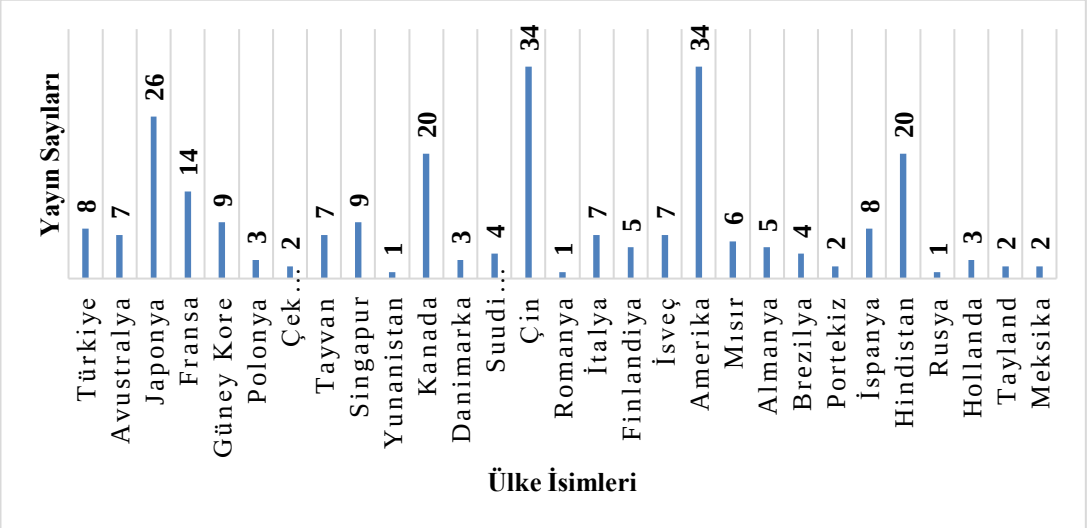
Şekil 1.13'te çok fazlı süzgeç bankası tasarımı hakkında 2000-2023 yılları arasında yapılan yayın sayıları verilmektedir. Şekil incelendiğinde 2009 yılında tasarım konusunun zirve yaptığı öncesi ve sonrasında ise bilim dünyasında bu konuyla ilgili çeşitli sayılarda yayınlar yapılmaya devam edildiği anlaşılmaktadır.



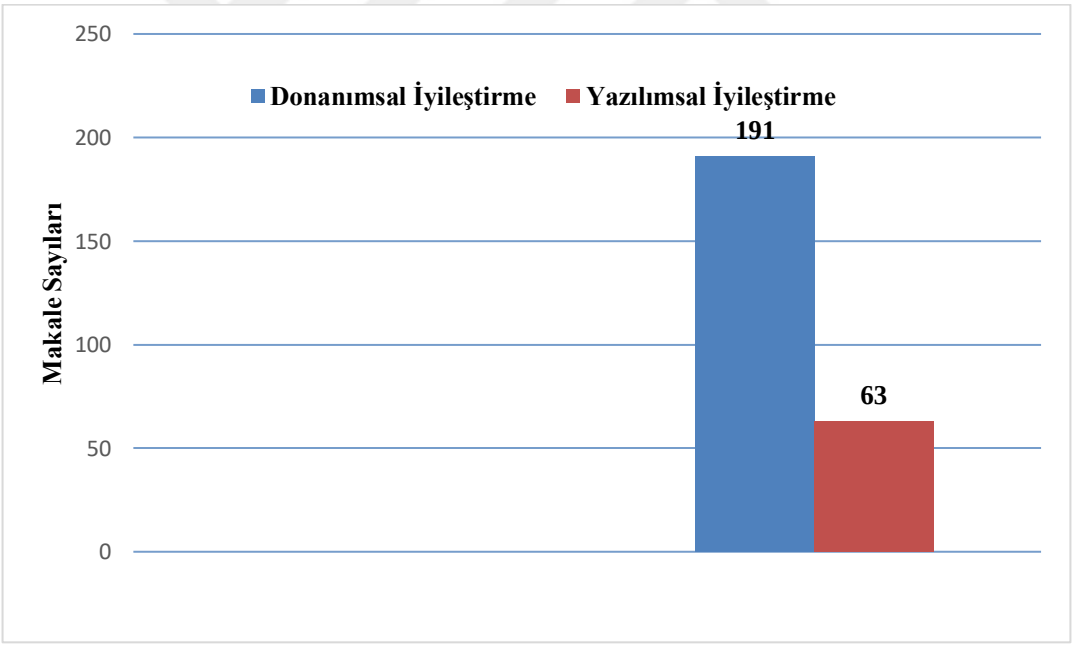
Şekil 1.13. Çok fazlı süzgeç bankası tasarımı hakkında 2000-2023 yılları arasında yapılan yayın sayıları

Şekil 1.14'te çok fazlı süzgeç bankası tasarımı hakkında 2000-2023 yılları arasında yapılan yayınların ülkelere göre dağılımı görülmektedir. Şekil 1.14'teki çubuk grafik incelendiğinde birçok ülkeden araştırmacının çok fazlı süzgeç tasarımı hakkında çeşitli sayıda yayınlar yaptığı görülmektedir. Özellikle Çin ve Amerika'da 2000-2023 zaman aralığında çok fazlı süzgeç tasarımı hakkında en yüksek sayıda yayın yapılmış bu iki ülkeyi Japonya, Kanada ve Hindistan takip etmiştir. Aralarında ülkemizin de 8 adet yayınla katıldığı bu konuda diğer ülkeler de çeşitli sayılarda yayınlarla konu hakkında bilim dünyasına katkıda bulunmuşlardır.

Şekil 1.16'da PFB hakkında 2000-2023 yılları arasında yapılan yayınların kullanılan yöntemlere göre dağılımı çizdirilmiştir.



Şekil 1.14. Çok fazlı süzgeç bankası tasarımı hakkında 2000-2023 yılları arasında yapılan yayınların ülkelere göre dağılımı



Şekil 1.15. Çok fazlı süzgeç bankası tasarımı hakkında 2000-2023 yılları arasında yapılan yayınların kullanılan yöntemlere göre dağılımı

2. BÖLÜM

SÜZGEÇ TASARIMINDA KULLANILAN ALGORİTMALAR

Sezgisel algoritmalar, karmaşık ve zorlu optimizasyon problemlerinin çözümlerini bulmak için tasarlanmış, genellikle doğadan ilham alan ve sezgiye dayalı yaklaşımlardır. Bu algoritmaların temel amacı, arama uzayındaki en iyi veya yeterince iyi çözümleri etkin ve verimli bir şekilde keşfetmektir. Algoritmaların her biri, belirli bir doğal mekanizmayı matematiksel ve bilgisayar bilimleri bağlamında modelleyerek karmaşık problemlerin çözümüne uygulamaktadır. Sezgisel algoritmaların uygulanabilirliği, genellikle problemin doğası ve gereksinimlerine göre değişiklik gösterir. Örneğin, GA, evrimin genetik varyasyon ve doğal seleksiyon ilkelerini taklit ederek çözüm popülasyonunu iterasyonlar boyunca geliştirir. PSO, kuş sürülerinin veya balık sürülerinin toplu hareketlerinden esinlenir ve bir çözüm uzayında bireylerin/parçacıkların deneyimlerinden öğrenerek küresel optimuma ulaşmayı amaçlar. ABC algoritması da arıların besin kaynaklarını arama davranışını modelleyerek optimizasyon problemine uygulanmaktadır. Algoritmaların güçlü yanları arasında, bölgesel optimumlara takılmadan küresel optimuma yakınsama ve türev bilgisi gerektirmeyen problemler için uygun olmaları yer almaktadır. Ayrıca, paralel işleme kabiliyetleri ve yüksek boyutlu arama uzaylarında etkili olabilmeleri de problemlerin çözümünde avantajlar sağlamaktadır. Ancak, bu algoritmaların parametre ayarlaması ve algoritmanın probleme uygunluğunun değerlendirilmesi gibi zorlukları da vardır. Bilimsel araştırmalarda ve endüstriyel uygulamalarda, meta sezgisel algoritmalar özellikle çok amaçlı optimizasyon, sayısal işaret işleme temelli tasarımlar, zamanlamalı karar verme problemleri ve mühendislik tasarımı gibi alanlarda yaygın olarak kullanılmaktadır. Sonuç olarak, meta sezgisel algoritmalar, özellikle geleneksel optimizasyon yöntemlerinin yetersiz kaldığı veya uygulanamadığı durumlarda, çeşitli disiplinlerdeki karmaşık problemlere yenilikçi ve etkili çözümler sunmaktadır.

Sezgisel algoritmalar, optimizasyon problemlerinin çözümünde esnek özellikleri sayesinde uygulama kolaylığı, hızlı yakınsama, düşük parametre hassasiyeti, küresel araştırma yeteneği gibi üstün avantajlarından dolayı birçok araştırmacı tarafından farklı bilim dallarında yaygın bir şekilde kullanılmaktadır [61]. PFB tasarım problemleri daha önceki bölümde de bahsedildiği gibi, çok parametreli ve çok boyutlu optimizasyon problemleridir. Bu nedenle, PFB tasarım problemlerinin çözümünde sezgisel algoritmaların kullanılması daha uygundur. Bu tez çalışmasında PFB tasarımı için ABC, mABC, qABC, PSO ve GA gibi meta sezgisel algoritmalar kullanılmıştır. Kullanılan algoritmalar ortak bir çözüm kümesi üzerinde çalışarak daha iyi çözümler geliştiren popülasyon tabanlı algoritmalar olmakla birlikte, literatürde en çok kullanılan ve karşılaştırılan sezgisel algoritmalardır [62-67].

2.1. Yapay Arı Koloni Algoritması

ABC doğada bir arada sürü halinde yaşayan bal arılarının besin arama davranışlarından esinlenen ve bu davranışları modelleyen ve optimizasyon problemlerinin çözümünde kolaylık ve etkinlik sağlaması amacıyla geliştirilen bir sezgisel algoritmadır. Karaboğa tarafından 2005 yılında ortaya konulmuştur ve kuş, balık ve yarasa gibi hayvan sürülerinin belirli bir amaca yönelik ortak hareketlerinden esinlenilerek geliştirilen sürü zekâsına dayalı sezgisel algoritmaların en başarılısıdır [5, 42]. ABC algoritması, çok boyutlu, çok modlu, sınırlı veya sınırsız birçok sayısal optimizasyon problemine uygulanmış ve GA, PSO, DE gibi literatürde çok sık kullanılan sezgisel optimizasyon algoritmalarıyla karşılaştırılarak etkinliği ortaya konulmuştur [22, 68, 69]. Ayrıca, algoritma gürültü gidermeden sistem modellemeye, parametre belirlemeden süzgeç bankası tasarımına kadar birçok alanda başarıyla kullanılmıştır [23, 60, 70-77].

ABC algoritması bal arılarının besin arama davranışlarını modellerken bazı varsayım ve kabulleri yapmaktadır. Algoritmada görevli arıların sayısı toplam besin kaynağının sayısına eşittir. İşçi arıların sayısı aynı zamanda gözcü arıların da sayısına eşittir. Nektarı tükenmiş kaynağın arısı kâşif arı haline dönüşmektedir. Besin kaynaklarının pozisyonları çözülmesi istenen optimizasyon problemine ait olası çözümlere ve kaynakların nektar miktarları ise o kaynaklarla ilgili çözümlerin kalitesine (fitness, uygunluk) karşılık gelmektedir. ABC algoritması en fazla nektara sahip kaynağın yerini bulmaya çalışarak çözülmesi istenen optimizasyon probleminin minimumunu veya maksimumunu veren

çözümü elde etmeyi amaçlar. ABC algoritmasının sözde kodu Şekil 2.1'de gösterilmiştir [67].

Kâşif arıların başlangıçtaki besin kaynaklarına gönderilmesi
REPEAT
İşçi arıların besin kaynaklarına gönderilmesi
Gözcü arılar için kaynakların olasılık değerinin hesaplanması
Gözcü arıların seçilen besin kaynaklarına gönderilmesi
Arılar tarafından terk edilen kaynakların kullanımı sürecinin durdurulması
Kâşif arıların yeni besin kaynakları için rastgele arama alanına gönderimi
UNTIL (durdurma kriterleri sağlanana kadar)

Şekil 2.1. ABC Algoritmasının sözde kodu

Algoritmada çözüm uzayı kovan çevresi olarak kabul edilir ve başlangıçta rasgele üretilmiş çözüm popülasyonu Eşitlik 2.1'den belirlenir.

$$x_{ij} = x_j^{\text{min.}} + rand(0,1) \times (x_j^{maks.} - x_j^{\text{min.}}) \qquad (2.1)$$

Burada, x_i popülasyonda bir çözümü göstermektedir. Eşitlik 2.1'de:

i=1……SN , SN: yiyecek kaynağı sayısı, j=1…D, D: optimize edilecek parametre sayısı, $x_j^{\text{min.}}$, j. parametrenin alt sınırı, $x_j^{\text{maks.}}$ da j. parametrenin üst sınır değerini göstermektedir. Algoritma başlangıcında her kaynağa ait sayaç sıfırlanmaktadır. Maksimum çevrim sayısı (Maximum Cycle Number, MCN) veya tanımlanmış bir durdurma kriteri, algoritmanın durdurma kriteri olarak kullanılabilir [69, 78, 79] .

ABC algoritmasının başlangıç aşamasından sonra görevli arı, gözcü arı ve kâşif arı süreçlerinden geçirilerek elde edilen çözümler iyileştirilmeye çalışılır. Algoritmanın durdurulması için belirlenen bir maksimum çevrim sayısı, hata değeri veya diğer algoritmalardaki kullanımlara benzer bir durdurma kriteri kullanılabilmektedir. ABC algoritmasında, her bir kaynağa sadece bir görevli arı atandığından yiyecek kaynaklarının sayısı görevli arıların sayısına eşit olmaktadır. Görevli arı yiyecek kaynağı üzerinde çalışırken hem kaynağın pozisyonunu hem kaynağın nektar miktarını hem de yeni kaynağa ait bilgileri de hafızasında tutmaktadır. Görevli arı çalışırken yiyecek kaynağı komşuluğunda yeni bir yiyecek kaynağı belirler ve bunun kalitesini de değerlendirir. Eğer yeni kaynaktaki nektar miktarı eskisinde daha çok ise arı hafızasındaki eski kaynak

bilgisini siler ve yeni kaynak ile ilgili bilgiler hafızasına alır. Yeni kaynağın mevcut olan kaynak komşuluğunda belirlenmesi Eşitlik 2.2 ile tanımlanmaktadır.

$$V_{ij} = x_{ij} + \phi_{ij} \times (x_{ij} - x_{kj}) \tag{2.2}$$

Burada j, [1, D] aralığında rasgele üretilen bir tamsayı olarak ifade edilmektedir. Rasgele seçilen j parametresi değiştirilirken, rasgele seçilen x_k komşu çözümünün $\left(k \in \{1,2,...SN\}\right)$ $j.$parametresi ile mevcut çözümün $j.$ parametresinin farkları alınıp [-1, 1] arasında rasgele değer alan ϕ_{ij} değeri ile ağırlıklandırılmakta ve ardından mevcut çözümün $j.$ parametresine eklenmektedir. Eşitlik 2.2'ye göre x_{ij} ile x_{kj} arasındaki fark azaldıkça, x_{ij} pozisyonundaki değişim de azalacaktır. Böylece, çözüme yaklaştıkça değişim miktarı uyarlayıcı bir şekilde azalacak ve ince bir ayarlama yapılacaktır.

Görevli arıların yeni çözümler üretmesi işleminde elde edilmiş yeni v_{ij} değerlerinin önceden belirlenen parametre sınırlarını veya önceden belirlenen çözüm uzayını aşması durumunda $j.$ parametreye ait olan alt ve üst sınır değerlerine yeniden ayarlama yapılarak ötelenmesi gerekmektedir. Bu işlem Eşitlik 2.3'de verilmektedir.

$$v_{ij} = \begin{cases} x_j^{\min}, v_{ij} < x_j^{\min} \\ v_{ij}, x_j^{\min} \le v_{ij} \le x_j^{\max} \\ x_j^{\max}, v_{ij} > x_j^{\max} \end{cases} \tag{2.3}$$

Görevli arıların üretmiş oldukları bu yeni çözümler yeni bir kaynağı temsil etmektedirler. Bu yeni kaynakların kalite değerleri hesaplanır ve Eşitlik 2.4 ile uygunluk değerleri atanır.

$$fitness_i = \begin{cases} \dfrac{1}{1+f_i} & f_i \ge 0 \\ 1+abs(f_i) & f_i \le 0 \end{cases} \tag{2.4}$$

Burada f_i değeri, v_i kaynağının yani çözümünün maliyet değeridir. Görevli arılar mevcut x_i çözümleri ile muhtemel v_t çözümleri arasında nektar miktarlarına yani uygunluk değerlerine göre bir açgözlü seçme işlemi gerçekleştirmektedirler. Yeni bulunan v_i çözümü daha iyi ise görevli arı hafızasından eski çözümü silerek v_i çözümünü hafızasına alır ve geliştirememe sayacını sıfırlar. Mevcut x_i çözümü v_i çözümünden daha iyi ise x_i çözümünü hafızasında tutmaya devam eder ve x_i çözümü ile ilgili sayacını bir artırır.

Bütün görevli arılar bir çevrim sonunda kovana döndüklerinde elde ettikleri bilgileri kovanda bekleyen gözcü arılarla paylaşmaktadırlar. Bu bilgi aktarma işlemi görevli arıların dans alanında sergiledikleri davranışlarla gerçekleşmektedir. ABC

algoritmasında görevli arıların danslarını izleyerek yiyecek kaynaklarının kalitesi hakkında bilgi sahibi olan ve bu bilgilere göre kaynak seçen serbest arılar gözcü arılar olarak adlandırılmaktadırlar. Gözcü arılar tüm görevli arıların verdiği bilgileri izledikten sonra kaynakları nektar miktarlarının oranlarına bağlı olarak Eşitlik 2.5 aracılığıyla hesaplanan olasılık değerlerine göre seçerler.

$$p_i = fitness_i / \sum_{i=1}^{SN} fitness_i \qquad (2.5)$$

Burada, fit_i i. kaynağın nektar miktarı yani uygunluk değeri, SN ifadesi de yiyecek kaynağı sayısını göstermektedir. Bu olasılık hesaplama işlemine göre bir kaynağın uygunluk değeri arttıkça bu kaynak bölgesini seçecek gözcü arı sayısı da artmaktadır. Böylece algoritmanın pozitif geri besleme özelliği ortaya çıkmaktadır. Algoritmada gözcü arılar kaynakları rulet tekerleği olarak adlandırılan bir olasılıksal seçme işlemi ile seçerler. Gözcü arılar gidecekleri kaynakları belirleme işlemini gerçekleştirdikten sonra aday yiyecek kaynağını görevli arıların komşuluk arama işlemine benzer şekilde Eşitlik 2.2 aracılığı ile üretmektedirler.

Gözcü arılar yeni üretilen her bir v_i kaynağının pozisyonu ile daha önceki x_i kaynağının pozisyonunu performans açısından değerlendirmekte ve aralarında bir açgözlü seçme işlemi uygulamaktadırlar. Yeni çözüm daha iyi ise hafızadaki eski çözüm silinerek yeni çözüm kaydedilmekte ve geliştirememe sayacı sıfırlanmaktadır. Aksi takdirde, eski çözüm hafızadaki yerini korumakta ve geliştirememe sayacı bir artırılmaktadır. Bu süreç tüm gözcü arılar yiyecek kaynağı bölgelerine dağılana kadar devam etmektedir. ABC algoritmasının akış diyagramı Şekil 2.2'de gösterilmektedir.

Algoritmanın her bir çevrimin sonunda tüm görevli ve gözcü arılar arama işlemlerini tamamladıktan sonra çözüm geliştirememe sayaçları kontrol edilmektedir. Çözüm geliştirememe sayaçları aracılığıyla bir arının çözümü temsil eden kaynaktan yeterince faydalanıp faydalanamadığını görülmektedir. Eğer bir kaynak için çözüm geliştirememe sayacı belirli bir eşik değerin üzerine çıkmışsa, artık bu kaynağın görevli arısı için kaynak tükenmiş olarak kabul edilmekte ve görevli arı kâşif arı haline dönüşmektedir. Kâşif arı haline gelen arı için de rasgele çözüm arama süreci yeniden başlamaktadır. ABC algoritmasında kaynağın tükendiğinin belirlenmesi için kullanılan eşik değer, *"limit"* parametresi olarak ifade edilmekte ve algoritma için önemli bir kontrol parametresi olarak kabul edilmektedir [22].

Algoritmanın temel adımları aşağıda verilmektedir:

1. Adım: Bütün x_{ij}, $i=1,2,...SN$, $j=1...D$ çözümlerine Eşitlik 2.1 kullanılarak başlangıç değerlerini ata ve çözüm geliştirememe sayaçlarını sıfırla,

2. Adım: Her bir x_i çözümünün fonksiyon ve uygunluk değerlerini ($fitness_i$) hesapla

REPEAT

3. Adım: Eşitlik 2.2 kullanarak mevcut çözüm x_i için yeni bir kaynak üret,

4. Adım: Eşitlik 2.4 kullanarak kaynağın uygunluk derecesini hesapla,

5. Adım: Yeni üretilen çözüm ve mevcut çözümü uygunluk değerine göre karşılaştırarak en iyi olanı tut. Mevcut çözüm ve yeni çözüm için sayacı ayarla,

6. Adım: Gözcü arıların yeni kaynak araştırmaları için Eşitlik 2.5 kullanarak olasılık değerlerini hesapla,

7. Adım: Rulet tekerleği tekniğini kullanarak [0,1] aralığında rastgele bir sayı üret,

8. Adım: Rastgele üretilen sayı, p_i olasılık değerinden küçük ise Eşitlik 2.2'yi kullanarak yeni bir kaynak üret. Üretilen kaynağın mevcut çözümün komşuluğunda olması gereklidir.

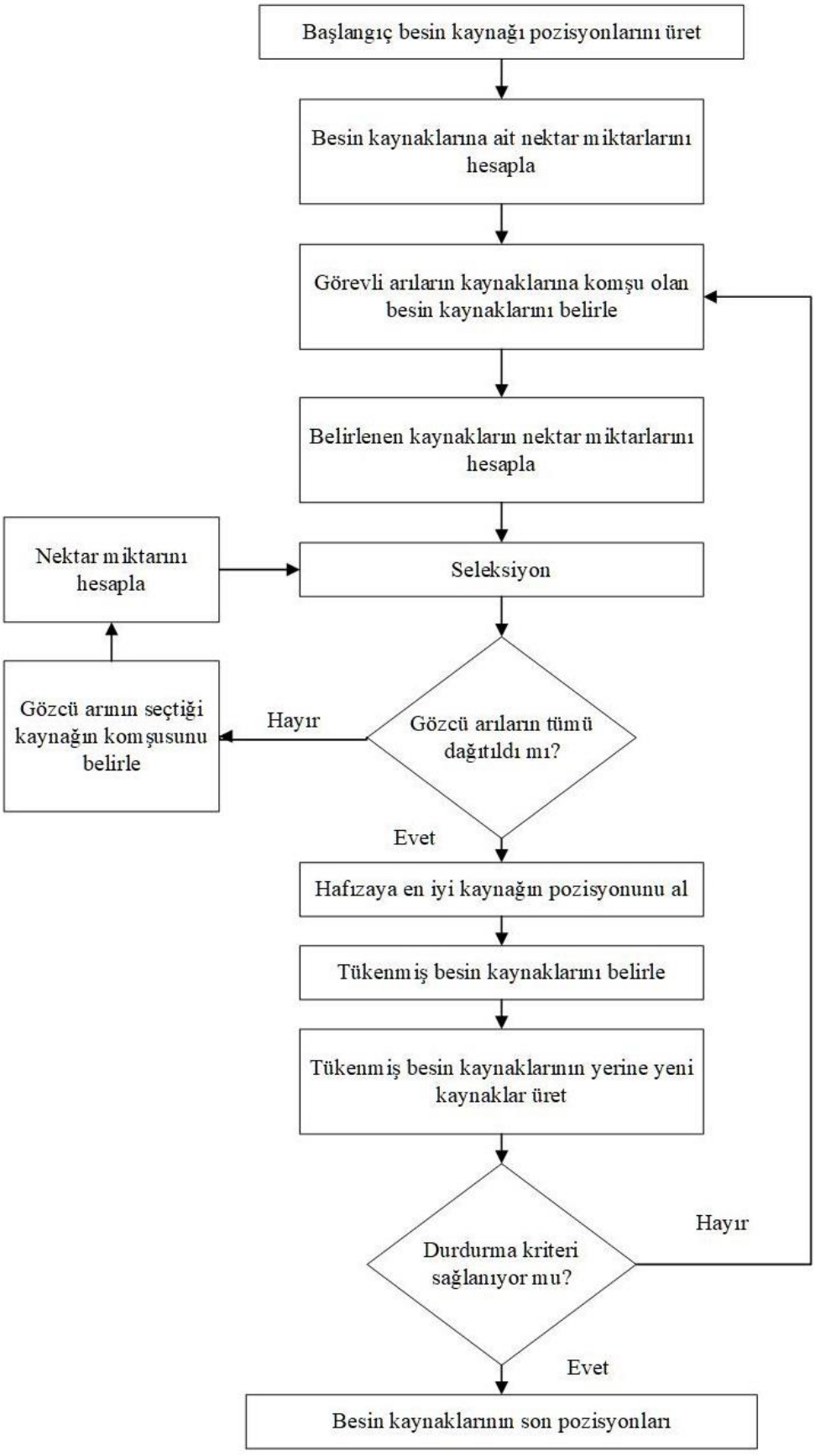
9. Adım: Yeni üretilen ve mevcut çözümleri karşılaştırarak en iyi olanı hafızada tut. Mevcut çözüm ve yeni çözüm için sayacı ayarla,

10. Adım: Her kaynak için tutulan çözüm geliştirememe sayacını kontrol et,

11. Adım: Eğer sayaç limit değerini aşmış ise o kaynakla ilgilenen arı kâşif arı olarak belirleyerek, mevcut x_i çözümünü Eşitlik 2.1 kullanarak yenile.

UNTİL (maksimum çevrim sayısı)

Başlangıçta algoritma için yapılan temel kabullere göre her çevrimde sadece bir kâşif arının çıkmasına izin verilmektedir. Bu durumda 3 temel kontrol parametresi karşımıza çıkmaktadır. Bunlar sırasıyla koloni sayısı, limit değeri ve maksimum çevrim sayısıdır. ABC algoritmasının uygulanacağı optimizasyon problemine göre bu kontrol parametrelerinin araştırılması ve uygun değerlerin seçilmesi yakınsama hızını, kalite çözümlere ulaşma performansını pozitif yönde etkileyecektir.



Şekil 2.2. ABC Algoritmasının akış diyagramı.

2.2. İyileştirilmiş Yapay Arı Koloni Algoritması

ABC algoritması, araştırma uzayının araştırılmasını etkili ve hızlı bir biçimde tamamlayarak en uygun değerini hesaplayabilmektedir. Bununla birlikte algoritma özellikle sınırlı optimizasyon problemlerinde, birleşik veya bileşenlerine ayrılamayan fonksiyonların optimizasyonunda yetersiz kalmaktadır. Karaboğa ve Akay tarafından önerilen yerel araştırma özelliğinde ya da gözcü arılar için yeni kaynak seçim işleminde değişiklik yapılması prensibine dayalı olan ABC algoritması, iyileştirilmiş ABC algoritması olarak adlandırılmaktadır [22, 43, 68, 69, 71, 74, 80-85].

mABC algoritmasının sözde kodu Şekil 2.3'te verilmiştir [86]:

Başlangıç popülasyonunun oluşturulması (besin kaynaklarının üretilmesi)
DO
İşçi arıların kaynaklara gönderilmesi
Gözcü arılar vasıtasıyla olasılık değerleri kullanılarak kaynak seçimi
Gözcü arıların kaynağa gönderilmesi
Geliştirilemeyen kaynakların terk edilmesi
Kâşif arıların rastgele kaynak aramaya gönderilmesi
WHİLE (durdurma kriterleri sağlanana kadar)

Şekil 2.3. mABC Algoritmasının sözde kodu

mABC algoritmasında, ABC algoritmasında olduğu gibi işçi, gözcü ve kâşif arı olmak üzere üç tip arı bulunmaktadır. Her bir çözüm bir besin kaynağını ifade etmektedir. Başlangıç için besin kaynaklarının üretilmesi ve her kaynak için bir arı seçimi işlemi aynı olup Eşitlik 2.2 kullanılarak gerçekleştirilmektedir. Mevcut kaynağın komşuluğunda yeni bir kaynak oluşturma işlemi mABC algoritmasında farklı gerçekleştirilmektedir. Eşitlik 2.6'da ifade edildiği gibi iyileştirme oranı (Modification Rate, MR) olarak kabul edilen yeni bir parametre değerlendirme için kullanılmıştır. Eğer rastgele üretilen bir sayı $(rand_{ij})$, önceden belirlenmiş olan MR değişim oranı parametresinden küçük ise yeni komşuluk seçilmekte üretilen sayı daha büyük ise yeni komşuluk seçilmemektedir [87].

$$u_{ij} = \begin{cases} x_{ij} + \phi_{ij}(x_{ij} - x_{kj}), & rand_{ij} < MR \\ x_{ij} & ,diğer \end{cases} \tag{2.6}$$

Eşitlik 2.6'da rastgele sayı ve MR parametreleri 0 ve 1 aralığındaki sayılardır,

$k \in \{1,2,...,SN\}$ ise rastgele seçilen bir değerdir.

Üretilen komşuluklar yine Eşitlik 2.4 kullanılarak kalite değerleri hesaplanmakta, mevcut ve yeni çözümlerin kaliteleri belirlenerek kalitesi daha iyi olan çözümler seçilmektedir.

Her bir çevrim sonunda kovana gelen arıların kaynak bilgilerinden kaynağın kalite değerine göre gözcü arıların kaynak seçme olasılıkları Eşitlik 2.5 ile gerçekleştirilmektedir. Gözcü arılar seçilen kaynağı araştırırken seçilen kaynağın komşuluğunda yeni bir kaynak bulma işlemini yine Eşitlik 2.6 ile gerçekleştirirler. Bulunan kaynak kalite değerine göre değerlendirilerek tutulur ya da göz önüne alınmaz.

Her çevrim sonucunda mevcut kaynakların geliştirilmesi ile ilgili bir sayaç tutulur. Kaynak geliştirilememiş ise bu sayaç bir artırılır. Limit olarak da adlandırılan bu parametre ile mevcut kaynak belirli bir çevrim boyunca geliştirilemediği durumda o kaynak tükenmiş olarak kabul edilir ve görevli arı kâşif arı olarak kabul edilir. Kâşif arı da rastgele araştırma işlemine yeniden başlar.

Şekil 2.4'te mABC algoritmasının akış diyagramı görülmektedir. Algoritma başlatma, değerlendirme ve sıralama, çözüm paylaşımlı gözcü arılar, işçi arılar çözüm paylaşımı, anlık güncelleme yerleşik seçim ile, kaşif arılar, işbirlikçi strateji ve popülasyon yöneticisi aşamalarından oluşmaktadır [88].

mABC algoritmasının temel adımları aşağıda verilmiştir.

1. Adım: Bütün $x_{ij,}\, i = 1,2,...SN,\, j = 1...D$ çözümlerine Eşitlik 2.1 kullanılarak başlangıç değerlerini ata ve çözüm geliştirememe sayaçlarının sıfırla,

2. Adım: Her bir x_i çözümünün fonksiyon ve uygunluk değerlerini hesapla,

REPEAT

3. Adım: Mevcut çözüm x_i için Eşitlik 2.6 kullanarak yeni bir kaynak üret ve Eşitlik 2.4 kullanarak kaynağın uygunluk derecesini hesapla,

4. Adım: Aç gözlü seçme işlemini kullanarak mevcut ve yeni çözümden en iyi olanı hafızada tut. Mevcut çözüm için sayacı bir artır, yeni çözüm için sayacı sıfırla,

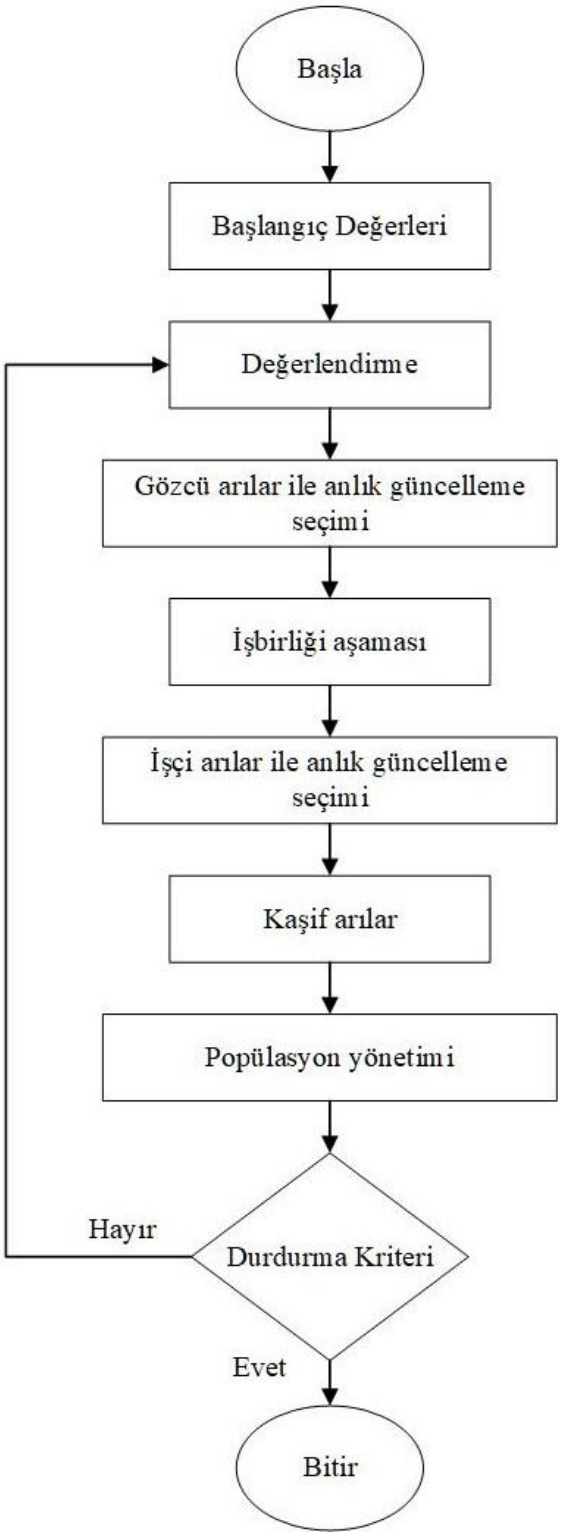
5. Adım: Eşitlik 2.5 ile p_i, olasılık değerlerini gözcü arılar için hesapla,

6. Adım: Eğer rastgele üretilen sayı p_i'den küçük ise mevcut çözümün komşuluğunda Eşitlik 2.2'yi kullanarak yeni bir kaynak üret,

7. Adım: Aç gözlü seçme işlemini kullanarak mevcut ve yeni çözümden en iyi olanı hafızada tut. Mevcut çözüm için sayacı bir artır, yeni çözüm için sayacı sıfırla,

8. Adım: Eğer sayaç limiti aşmış ise mevcut x_i çözümünü Eşitlik 2.1 kullanarak yenile.

UNTİL (maksimum çevrim sayısı)



Şekil 2.4 mABC Algoritmasının akış diyagramı

2.3. Hızlı Yapay Arı Koloni Algoritması

Karaboğa ve Görkemli tarafından önerilen hızlı ABC algoritmasının, ABC algoritmasına göre yakınsama hızı daha yüksektir böylece daha az adımda sonuca yakınsamaktadır [43, 85]. Yani, yeni çözüm geliştirilmesi işleminde mevcut komşuların en iyisinin seçilmesi prensibi göz önüne alınarak algoritmaya hızlı yakınsama özelliği kazandırılmıştır. qABC algoritmasının sözde kodu Şekil 2.5'te verilmiştir:

Başlangıç popülasyonunun oluşturulması
REPEAT
İşçi arıların besin kaynaklarına gönderilmesi
Gözcü arıların en yakın ve iyi kaynağı seçimi
Terk edilecek kaynakların hesaplanması
Terk edilen kaynaklardaki arıların kâşif arı olarak yeni kaynak araması
UNTİL (maksimum çevrim sayısı veya hata kriteri sağlanana kadar**)**

Şekil 2.5. qABC Algoritmasının sözde kodu

Bu algoritmada da ABC algoritmasında olduğu gibi işçi, gözcü ve kâşif arılar bulunmaktadır. Başlangıçta besin kaynağı kadar işçi arı bulunmaktadır. Her bir besin kaynağı olası birer çözümü ifade etmektedir. Besin kaynaklarının oluşturulması Eşitlik 2.1'de verildiği gibi gerçekleştirilir. Besin kaynaklarının araştırılmasından sonra Eşitlik 2.2'den farklı olarak mevcut çözümün komşuluğundaki en iyi çözüm seçilir. Kullanılan Eşitlik 2.7'de verilmiştir.

$$u_{N,i}^{best} = x_{N,i}^{best} + \phi_i (x_{N,i}^{best} - x_{ki}) \tag{2.7}$$

Eşitlik 2.7'de $x_{N,i}^{best}$, x komşuluğundaki en iyi çözümü ifade etmektedir ve x_{ij} ve u_{ij} arasındaki mesafe $d(m, i)$ Öklid mesafesidir. Mevcut çözüm x için komşuluğunda üretilecek yeni çözümler md_m Eşitlik 2.8 ile verilen Öklid uzaklığında olmalıdır.

$$md_m = \frac{\sum_{n=1}^{S} d(m,i)}{SN-1} \tag{2.8}$$

Eğer yeni çözümün uzaklığı md_m mesafesinden küçükse mevcut çözümün bir komşuluğu olarak kabul edilir ve kalite değeri incelenebilir. Bu işlem Eşitlik 2.9 ile açıklanabilir.

$$x_j = \begin{cases} d(m, j) < r * md_m, \ x_i \ komşusu \\ d(m, j) \geq r * md_m, \ değil \end{cases} \quad (2.9)$$

Eşitlik 2.9'da r komşuluk yarıçapı olup $r \geq 0$ şartını sağlaması gerekir. Yarıçap sıfır olarak alınırsa algoritma, ABC algoritması gibi çalışmaktadır. Komşulukta seçilen yeni olası kaynaklar Eşitlik 2.4 ile verildiği gibi bir kalite değerlendirme kriteri ile değerlendirilir. Mevcut kaynak ve komşuluğundan seçilen yeni kaynaktan hangisinin kalite değeri yüksek ise o kaynak saklanır, diğeri silinir. Her çevrim sonucu kovandaki gözcü arıların besin kaynaklarını seçme olasılığı Eşitlik 2.5 ile ifade edildiği gibi kaynakların kalite değerleri ile orantılı bir şekilde rulet tekerleği yöntemi ile seçilir. Her kaynağın geliştirilmeme sayacı olarak da ifade edilen Limit değerine bağlı olarak kaynak araştırılmaya devam edilir, limit değeri aşılınca kaynak bırakılır. Bu durumda bırakılan kaynağın görevli arısı, kâşif arı haline gelir ve Eşitlik 2.1'i kullanarak rastgele araştırma yapılmaya başlanır.

Şekil 2.6'da qABC algoritmasının akış diyagramı verilmiştir [89].

Algoritmanın temel adımları aşağıda açıklanmaktadır:

1. Adım: Bütün $x_{ij}, i = 1, 2, ...SN, \ j = 1...D$ çözümlerine Eşitlik 2.1 kullanılarak başlangıç değerlerinin ata ve çözüm geliştirememe sayaçlarının sıfırla,

2. Adım: Her bir x_i çözümünün fonksiyon ve uygunluk değerlerini hesapla,

REPEAT

3. Adım: İkili optimizasyon yöntemleri için Eşitlik 2.7 ile komşu bir kaynak üret. Öklid mesafe uzunluğunu Eşitlik 2.8'den hesapla. Bu uzunluk içerisinde yeni komşuluğu üret,

4. Adım: Yeni üretilen çözüm komşuluk mesafesi ve r komşuluk yarıçapı ile kabul edilip edilmeyeceğini Eşitlik 2.9'dan hesapla,

5. Adım: Yeni çözümün kalite değerini hesapla,

6. Adım: Mevcut çözüm ve yeni çözümden kalite değerleri en iyi olanı seç, diğerini dikkate alma. Mevcut çözüm için çözüm araştırma sayacını artır,

7. Adım: Gözcü arılar için Eşitlik 2.5 yardımıyla yeni olası çözümleri belirle,

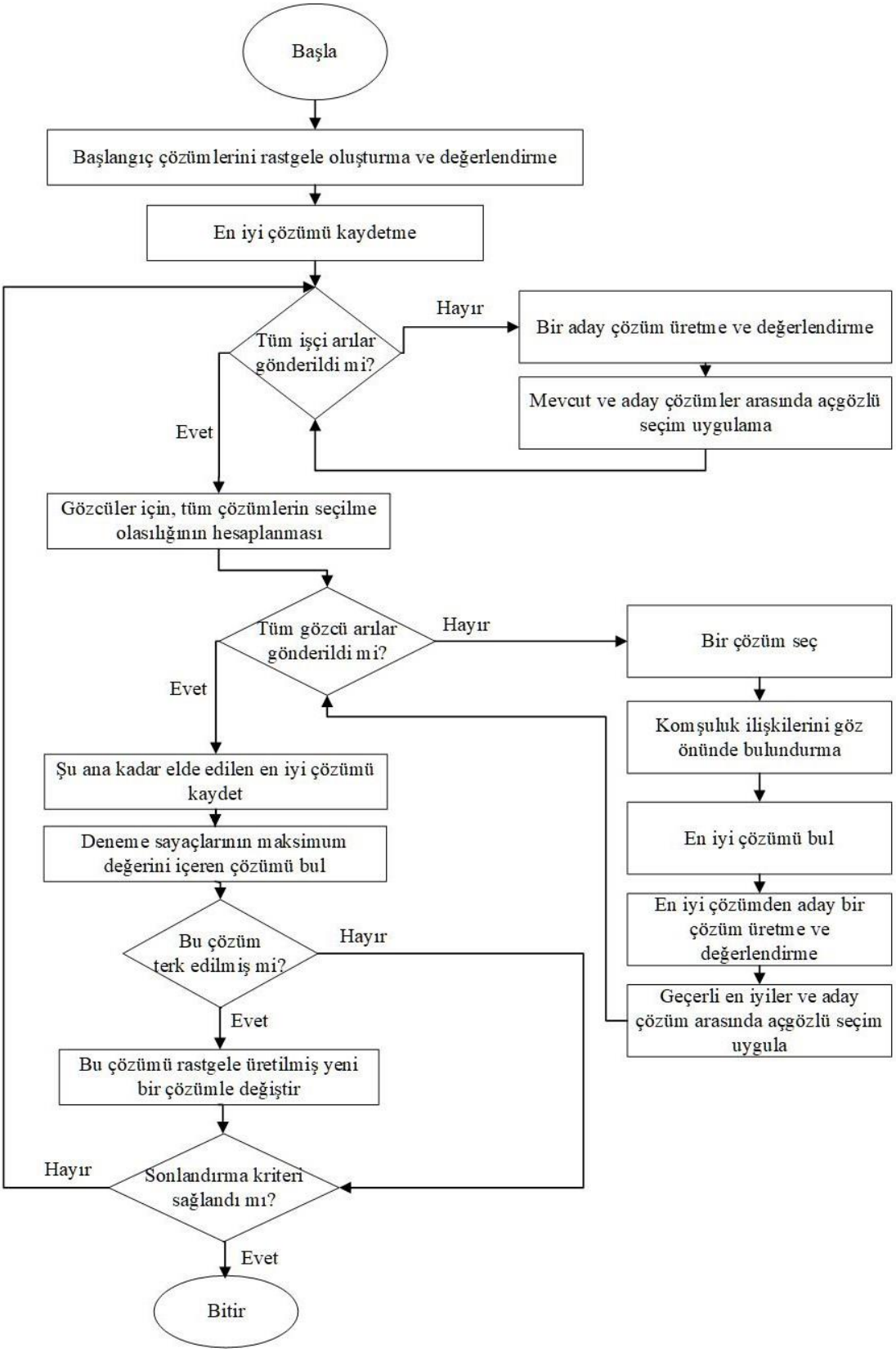
8. Adım: Eşitlik 2.2 ile yeni bir kaynak üret,

9. Adım: Mevcut çözüm ve yeni çözümden kalite değerleri en iyi olanı seç, diğerini dikkate alma. Mevcut çözüm için çözüm araştırma sayacını artır,

10. Adım: Eğer sayaç mevcut çözüm araştırması limit değerini aşmış ise o kaynağı terk

ederek kâşif arı haline gelen işçi arı ile rastgele kaynak araştırmasına başla.

UNTİL (maksimum çevrim sayısı)



Şekil 2.6. qABC Algoritması akış diyagramı

2.4. Parçacık Sürü Optimizasyon Algoritması

PSO algoritması, kuş ve balık gibi canlı sürülerinin sosyal davranışlarını özellikle de besin arama esnasında bireylerin göstermiş oldukları etkileşimleri ve kolektif davranışları modelleyen popülasyon tabanlı sürü zekasına dayanan sezgisel bir algoritmadır. 1995 yılında Kennedy ve Eberhart tarafından ortaya konulmuştur [34]. Başta çok parametreli problemlerin optimizasyonunda olmak üzere literatürde pek çok alanda kullanılmaktadır [90].

Algoritmada her bir parçacık optimizasyon problemine ait olası çözümü temsil etmekte ve parçacıkların oluşturduğu popülasyonda sürü olarak değerlendirilmektedir. Başlangıçta, tüm parçacıklar çözüm uzayında rasgele üretilir. Parçacıkların pozisyonları pozisyon veya konum vektörleriyle, hızları ise hız vektörleriyle gösterilmektedir. Sürü içerisinde yer alan tüm parçacıklar çözüm uzayında rasgele bir konumdan harekete başlamaktadır. Hareket esnasında, optimizasyon problemine ait olan fonksiyonda parçacığın pozisyonu yerine konularak çözümün kalitesi değerlendirilmektedir. Parçacıklar sürü olarak bu hareketi gerçekleştirirken, her bir parçacık kendisine ait olan en iyi pozisyon bilgisini tutmakta ve bununla birlikte sürü içerisindeki tüm parçacıkların en iyi pozisyon bilgisi de sürünün yönlendirilmesinde kullanılmaktadır. PSO algoritmasının modelinde birinci öncelik parçacıkların arasındaki sosyal bilgi paylaşımının hızlanması ve yaygınlaşmasıdır. Sürü içerisindeki tüm parçacıkların uygunluğu iteratif olarak hesaplanır ve en iyi çözümün pozisyonu hafızaya alınır. Her bir parçacık bir sonraki pozisyonunu Eşitlik 2.6'da gösterildiği gibi bir önceki pozisyonuna ve hızına göre ayarlamaktadır.

$$\vec{x}(t+1) = \vec{x}(t) = \vec{v}(t+1) \tag{2.10}$$

Burada $\vec{x}(t)$ konum vektörünü ve $\vec{v}(t)$ hız vektörünü temsil etmektedir. Her bir parçacık hız vektörünü güncellerken, kendisinin sahip olduğu en iyi değer $\vec{p}(t)$ ile tüm sürü içerisindeki parçacıkların sahip olduğu en iyi yani en iyi küresel değer $\vec{g}(t)$'yi kullanmaktadır. Böylece parçacık çevresi ile etkileşerek hem kendi en iyi pozisyonunu hem de küresel en iyi pozisyonu kullanarak arama uzayında hareket etmektedir. Her bir parçacık kendi hız vektörünü Eşitlik 2.11'de verildiği gibi güncellemektedir.

$$\vec{v}(t+1) = w\vec{v}(t) + c_1 rand(0,1)(\vec{p}(t) - \vec{x}(t)) + c_2 rand(0,1)(\vec{g}(t) - \vec{x}(t)) \tag{2.11}$$

Algoritmanın sözde kodu Şekil 2.7'de verilmiştir.

Başlangıç popülasyonunun oluşturulması
REPEAT
Her parçacık için uygunluk değerini hesapla
Hesaplanan ve sürüdeki en iyi uygunluk değerini sakla
En iyi uygunluk değerine sahip parçacığı seç
Her parçacık için konum ve hızını hesapla
UNTIL (maksimum çevrim sayısı ya da hata kriterine ulaşana kadar)

Şekil 2.7. PSO Algoritmasının sözde kodu

Burada *w* eylemsizlik ağırlığı olarak adlandırılmakta ve yeni hız değerinin hesaplanmasında önceki hız değerinin $\vec{v}(t)$ ağırlığını göstermektedir. $c_1 ve\ c_2$, $\vec{p}(t)$ ve $\vec{g}(t)$ vektörlerinin ağırlıkları veya bilişsel ve sosyal bileşenler olarak da ifade edilmektedir. Herhangi bir parçacığın sahip olduğu hız vektörünün değeri V_{max} ile sınırlandırılabilmektedir.

PSO algoritmasına ait temel adımlar aşağıdaki gibi ifade edilmektedir [67].

1. Adım: Başlangıç popülasyonunun üretilmesi oluşturulması,

REPEAT

2. Adım: Sürüdeki her bireyin uygunluk değerinin hesaplanması,
3. Adım: Sürü içerisindeki en iyi bireyin konumunun değiştirilmesi,
4. Adım: Eşitlik 2.10 kullanılarak sürüdeki her bireyin hızlarının güncellenmesi,
5. Adım: Eşitlik 2.11 kullanılarak sürüdeki her bireyin konumlarının güncellenmesi,
6. Adım: Elde edilen en iyi sonuçların kaydedilmesi,

UNTIL (çevrim=maksimum çevrim sayısı)

PSO algoritmasının temel kontrol parametreleri koloni sayısı, ω eylemsizlik ağırlığı, c_1 ve c_2 bilişsel ve sosyal bileşenler ve maksimum çevrim sayısı veya belirlenen bir durdurma kriteri olarak tanımlanmaktadır.

2.5. Genetik Algoritma

GA, popülasyon tabanlı algoritma grubunda değerlendirilen ve 1970'li yılların başında J. Holland tarafından keşfedilen sezgisel bir araştırma yöntemidir [91]. Canlılarda bulunan doğal seleksiyon ve genetik gelişim benzerliğini gerçekleştirmektedir. GA, arama

uzayında çözümler tarafından oluşturulan başlangıç koşullarını kullanır. Daha sonra çaprazlama, mutasyon ve doğal seçimler bir sonraki nesil için yeni çözümler üretmek için kullanılır. Bir sonraki nesil oluşturulurken her bir bireye uygunluk değerlendirme süreci uygulanır. Birbirini izleyen çözümlerin değerlendirilmesi ve geliştirilmesine dayanan bu döngü nesiller, optimal bir çözüm bulunana kadar devam eder [25, 92].

GA, doğal evrim sürecini taklit ederek arama ve optimizasyon problemlerini çözen bir yöntemdir. Çok boyutlu karmaşık arama alanlarında en uygun bireylerin hayatta kalmasına dayalı bir mantıkla çalışan ve küresel optimum çözümleri bulmaya yönelik bir algoritmadır. GA, diğer optimizasyon yöntemleri gibi, tek bir çözüm yapısını geliştirmek yerine, çeşitli çözüm yapılarının oluşturduğu bir küme üzerinde işlem yapar. Bu küme, GA terminolojisinde popülasyon olarak adlandırılır ve bir problemin farklı çözümlerini temsil eder. Popülasyonda vektör, birey veya kromozom olarak isimlendirilen sayı dizileri bulunur. Bu bireylerin her bir elemanına gen adı verilir. Popülasyonun bireyleri GA operatörleri tarafından evrimsel süreç içinde belirlenir [26, 66, 93-95]. Klasik türev temelli optimizasyon yöntemlerine göre, GA'lar, başlangıçtaki popülasyonun yeterli seçilmesi durumunda yerel optimumlara yakalanma riski taşımazlar. GA'lar, çok değişkenli ve çok amaçlı problemlere kolayca uyum sağlayabilmesi sebebiyle süzgeç optimizasyonunda kullanılmaktadır [96]. GA'nın sözde kodu Şekil 2.8'deki gibi ifade edilmektedir [67].

```
    Başlangıç
Başlangıç popülasyonunu oluştur.
Popülasyonu değerlendir
    WHILE
        Seçme
        Çaprazlama
        Mutasyon
        Popülasyonu değerlendir
    UNTIL
    END
```

Şekil 2.8. GA'nın sözde kodu

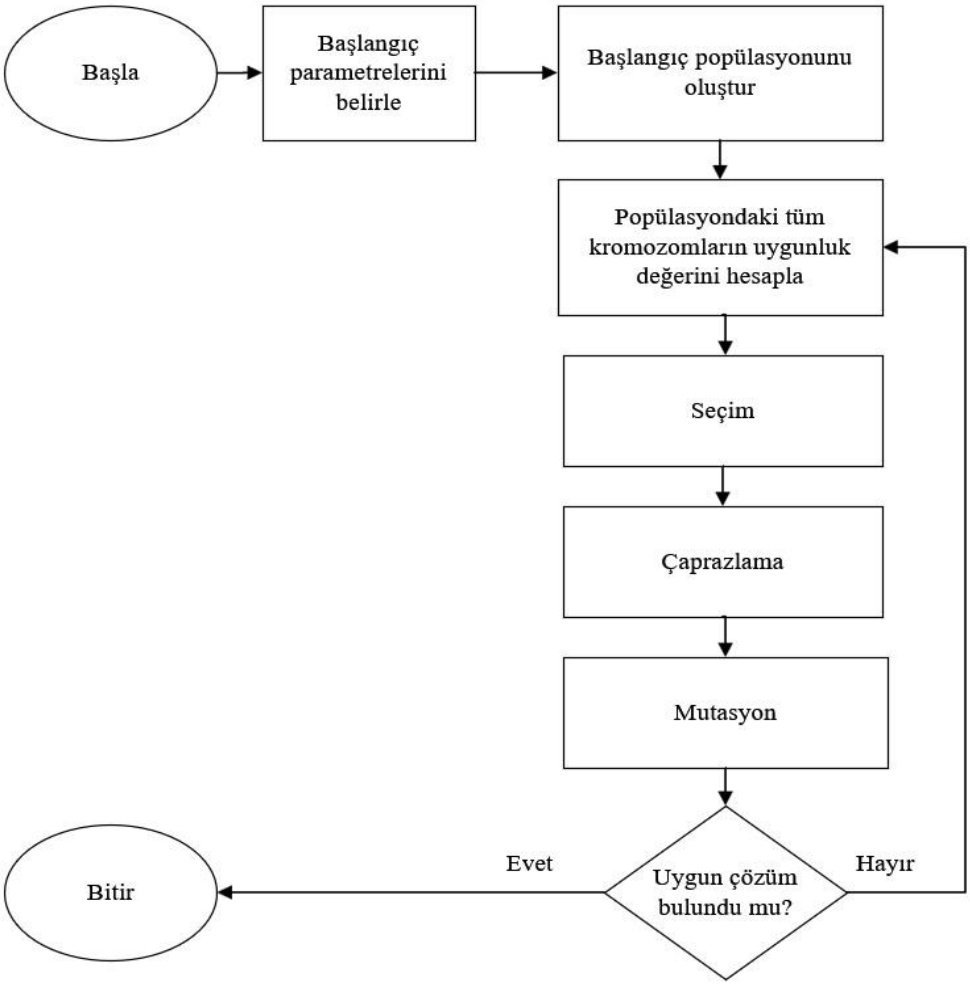
GA'da, çok sayıda olası çözüm adayının bir arada bulunduğu bir popülasyon oluşturulur. Bu aday çözümler, probleme uygun bir şekilde kodlanmış dizilerde saklanır. Bu dizideki her bir öğeye birey adı verilir. Her bir birey, arama alanındaki belirli bir bölgeyi ifade eder. Başlangıç bireyleri rastgele oluşturulabilir, ancak bu zorunlu değildir. Özellikle çok kısıtlı optimizasyon problemlerinin çözümünde, başlangıç bireylerini tanımlanan kısıtlardan bazılarına göre atamalar yaparak daha iyi adaylar elde etmek mümkün olabilir. Bireyler, uygunluk fonksiyonuna göre değerlendirilerek, çözümün optimal çözüme ne kadar yakın olduğunu gösteren bir uygunluk değeri hesaplanır. Bir GA, başlangıç popülasyonu oluşturulduktan sonra üç evrim operatörü ile çalışır. Bu operatörler; seçim, çaprazlama ve mutasyondur. Genel olarak bu operatörlerin her biri, yeni nesildeki popülasyonun her bireyi için ayrı ayrı uygulanır. Seçim işlemi, popülasyondaki bireyin uygunluk değerine bağlı olarak, yeni bireyler üretmek için, ebeveyn bireyi seçme işlemi olarak tanımlanır. Çaprazlama operatörü, seçim işleminden sonra gerçekleştirilir. Ebeveyn bireylerinin kromozomlarının belirli bölümlerinin karşılıklı olarak değiştirilmesi sonucunda yeni niteliklere sahip bireyleri oluşturmaktan sorumludur. Mutasyon, yeni nitelikteki bireylerde kromozomlardan herhangi birinde bulunan genin mutasyon olasılığına göre değiştirilmesidir.

GA parametresinde belirlenen popülasyondaki birey sayısı kadar birey üretildiğinde, yeni bir nesil ortaya çıkar. Bu yeni nesil önceki neslin yerine geçer. Her nesilde bireylerin temsil ettikleri çözüm değerlendirilerek, popülasyondaki en iyi birey seçilir. Bulunan en iyi birey, önceki nesillerdeki en iyi bireylerden daha iyiyse onun yerini alır. GA işlemini bitirme işlemi için farklı yöntemler vardır. Bu yöntemler; algoritma çalışırken istenilen çözüme ulaşıldığında, GA'nın başında tanımlanmış olan toplam iterasyon sayısına gelindiğinde veya uygunluk değerinin sürekli sabit kaldığı durumlarda, bulunan en iyi bireyi temsil eden çözüm, problem için en uygun çözüm olarak kabul edilir. GA'nın çalışmasını etkileyen faktörler parametrelerdir. Genel olarak bu parametreler popülasyon büyüklüğü, mutasyon tipi ve olasılığı, çaprazlama tipi ve olasılığı, nesil sayısı ve seçim operatörünün tipidir.

Şekil 2.9'da GA'nın genel akış şeması verilmiştir.

Genetik algoritmanın performansını etkileyen faktörler aşağıdaki gibi sıralanabilir [26]:
Kodlama şekli: GA'nın performansını belirleyen önemli bir faktördür, ancak kodlama

şekli problem ve programa bağlı olduğu için tüm problemlerde geçerli olan uygun kodlama şeklini belirtmek mümkün değildir. Michalewicz belirli bir problem tipi için yaptığı çalışmada, gerçek sayı gösteriminin daha hızlı sonuç verdiğini belirtmiştir [97]. GA, yerel en iyi çözümü bulmak yerine geniş ölçüde çoklu arama yeteneği sunar [98].



Şekil 2.9. GA'nın akış diyagramı

Kromozom sayısı: Kromozom sayısı arttıkça çalışma süresi de artar, sayıyı azaltma işlemi ise kromozomların çeşitliliğini engeller. GA'ların temel özelliği, kromozomlara sahip bireylerin karakterize ettiği topluluğun yenilenmesidir. Her bir kromozom, optimizasyon probleminde uygun bir çözüm olarak düşünülür. Sembollerin her biri gen olarak adlandırılır ve gen hangi parametreyi ifade edecekse sıra ile dizilmesi sonucu kromozomlar oluşturulur.

Mutasyon oranı: Kromozomların yapısı birbirine benzer hale geldiğinde hala çözüm değerlerinin uzağında oldukları durumda, bu durumu mutasyon işlemi ile aşmak mümkündür. GA'nın çıkmaza girdiği yerden kurtulmak için tek yol olarak da söylenebilir. Ancak yüksek bir değer vermek GA'yı kararlı bir noktaya ulaşmasını zorlaştıracaktır. Mutasyon oranı P_m, algoritmanın başında ya da her jenerasyon başında belirlenmektedir. Bir popülasyonda yaklaşık olarak $P_m \times N \times \ell$ adet karakterde mutasyon meydana gelmektedir. Burada, N popülasyon büyüklüğü ve ℓ kromozom uzunluğudur. GA'nın tamamen rastgele olmasından kaçınmak için mutasyon oranının çok yüksek seçilmemesi gerekir. Mutasyon oranının çok yüksek seçilmesi, arama işleminde aşırı rastgeleliğe sebep olur ve bu durum çözüm uzayının da aşırı farklılaşma riskini artıracaktır. Benzer şekilde düşük mutasyon oranı ise düşük farklılaşma ve yerel optimuma yakın çözümler üretilmesine neden olacaktır. Literatürdeki çalışmalarda uygun P_m oranının $1/N \le P_m$ ve $P_m \ge 1/\ell$ olması önerilmektedir [99, 100].

Birçok araştırmacı, uygun P_m oranının $1/N \le P_m$ ve $P_m \ge 1/\ell$ olmasını önermektedir [101]. Mutasyon ile değişime maruz kalan bir gende yani değişkende, λ gen uzunluğu olmak üzere, beklenen değişim;

$$E(G) = \frac{1}{\lambda}\sum_{i=1}^{\lambda}\frac{1}{2^i} \tag{2.12}$$

Eşitlik 2.12 ile hesaplanmaktadır ve değişken değerinde çok fazla bir değişime neden olmamaktadır. Örneğin gen uzunluğu λ=4 bit ise $E(G)$=1/4(0.5+0.25+0.125+0.0675) =0.23563 kadar bir değişim beklenmektedir. Böylece mutasyon ile çözüm uzayı dışındaki farklı noktalara sıçrayarak, algoritmanın daha ayrıntılı araştırma yapmasına olanak tanır. Pratik çalışmalarda, algoritmanın tamamen rastgele olmasını engellemek amacıyla mutasyon oranı, P_m değerinin 0.01 ile 0.001 arasında alındığı görülmektedir. Uygunluk değeri amaç fonksiyonu ile ters orantılı olarak belirlenmelidir. Buna göre t döngüsündeki x^t vektörü bireylerin dış yapılarına karşılık gelecek olursa $u(x^t)$ uygunluk değeri Eşitlik 2.14'te verilmiştir.

$$\partial = \max\left\{-1 \times \min\left[\, f(x_1^t), \ldots, f(x_N^t)\,\right] + 1, 0\right\} \tag{2.13}$$

$$u(x_i^t) = \frac{1}{\partial + f(x_i^t)}, i = 1, ..., N \tag{2.14}$$

Eşitlik 2.13'teki ∂ değişkeni sayesinde uygunluk değeri asla negatif olmayacak ve sıfıra bölme hatası ile karşılaşılmayacaktır. Böylece bireyin çaprazlama için seçilme olasılığı 2.15 eşitliği ile hesaplanmaktadır.

$$p_i^t = \frac{u(x_i^t)}{\sum_{j=1}^{N} u(x_j^t)} \tag{2.15}$$

Kaç noktalı çaprazlama yapılacağı: Genellikle çaprazlama tek noktada gerçekleştirilirse de bazı problemlerde çok noktalı çaprazlama da kullanılmaktadır.

Parametre kodlamasının nasıl yapıldığı: GA'da parametrenin doğrusal veya logaritmik kodlanması algoritmanın performansını da etkilemektedir.

Popülasyon büyüklüğü: Popülasyonun büyüklüğü problemin çözüm süresinde etkilidir. Popülasyonda bulunan bireylerin sayısı gereğinden fazla ise çözüm süresi uzar, bireylerin sayısı gereğinden az ise popülasyon istenen çözüme ulaşamayabilir. Problemin yapısına göre popülasyondaki birey sayısının iyi belirlenmesi gerekir.

GA'nın temel adımları aşağıdaki gibi ifade edilmektedir:

1. Adım: Uygunluk fonksiyonunun tanımlanması,
2. Adım: Kodlama,
3. Adım: Rastgele bireylerle başlangıç popülasyonunun seçilmesi,

REPEAT

4. Adım: Popülasyondaki bütün bireylerin uygunluk fonksiyonunun hesaplanması,
5. Adım: Yeni nesil için en iyi bireylerin seçilmesi,
6. Adım: Çaprazlama ve mutasyon ile yeni neslin oluşturulması,
7. Adım: Yeni nesli (kromozomları) popülasyona ekle,

UNTIL (yeterince iyi bir çözüm bulunana kadar**)**

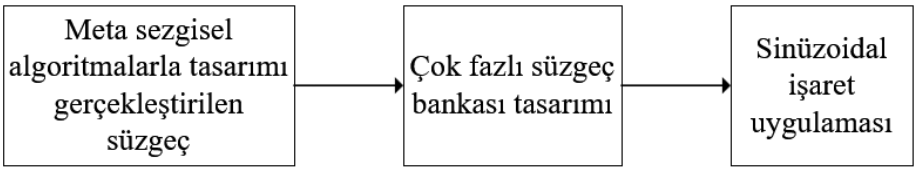
3. BÖLÜM

SAYISAL SÜZGEÇ TASARIMI

Süzgeç bankaları sayısal işaret işleme, haberleşme sistemleri, biyomedikal işaret işleme ve alt bant kodlama gibi alanlarda geniş oranda kullanılır. Son zamanlarda, çok fazlı süzgeç bankası tasarımına olan ilgi hem araştırmacılar hem de uygulayıcılar tarafından önemli ölçüde artmıştır. Süzgeç tasarımında ideal frekans seçiciliği, süzgecin farklı frekans bantlarındaki performansını belirleyen bir ölçüttür. İdeal bir süzgeç, belirli bir frekans aralığındaki işaretleri tamamen geçirirken, diğer frekanslardaki işaretleri tamamen engeller. Ancak, pratikte bu tür bir "ideal" süzgeç gerçekleştirilemez. Gerçek süzgeçler genellikle belirli bir geçiş bandına sahiptir, bu da süzgecin tamamen geçirdiği ve tamamen durdurduğu frekanslar arasında bir geçiş bölgesine sahip olduğu anlamına gelir. İdeal bir alçak geçiren süzgeç, kesme frekansına kadar olan tüm frekansları değiştirmeden geçirir ve diğer tüm frekansları da engeller. Ancak, gerçek dünyada süzgeçler genellikle kesme frekansının hemen üzerindeki ve altındaki frekanslarda tam anlamıyla geçirmez veya durdurmazlar. Bu nedenle, ideal frekans seçiciliği elde etmek için süzgeç tasarımında çeşitli teknikler kullanılır. Bu teknikler genellikle süzgecin durdurma bandındaki zayıflamayı artırmayı ve geçiş bandını daraltmayı hedeflemektedir. Bu durum, istenmeyen frekansların daha etkin bir şekilde engellenmesini ve istenen frekansların daha net bir şekilde geçirilmesini sağlar. Süzgeç bankası tasarımında kritik bir unsur olan frekans seçiciliği, süzgeç bankasının farklı frekans bantlarındaki performansını belirleyen bir ölçüttür. İdeal frekans seçiciliğini elde etmek için, süzgeç bankasının yüksek durdurma bandı zayıflamasına ve dar bir geçiş bandına sahip olması gereklidir [102-104].

3.1. Sayısal Süzgeç Tasarımı

Süzgeç bankası tasarımında sayısal ve sezgisel optimizasyon teknikleri yaygın olarak kullanılmaktadır. Çok fazlı süzgeç bankaları, çeşitli süzgeç yapılarının tasarımında, süzgeçlemenin maliyetini ve karmaşıklığını azaltan önemli bir bileşen olarak ön plana çıkmaktadır. Bu bölümde, ABC, qABC ve mABC, PSO ve GA algoritmaları ile PFB tasarımında kullanılacak sayısal süzgeç uygulamaları gerçekleştirilmiştir. Sezgisel algoritmaların kontrol parametreleri elde edilen sonuçları doğrudan etkilemektedir. Bu sebeple süzgeç tasarımında öncelikle ABC tabanlı algoritmaların kontrol parametreleri ayrıntılı olarak incelenmiş ve en iyi prototip süzgeci veren parametre değerleri belirlenmiştir. Belirlenen bu sayısal süzgeç diğer bölümde PFB'nin tasarımına temel teşkil etmiştir. PFB tasarımı için bu tez çalışmasında önerilen model Şekil 3.1'de gösterilmiştir. Süzgeç bankasının performansı, girişine uygulanan bir sinüzoidal işarete karşılık nasıl bir çıkış işareti ürettiği temel alınarak değerlendirilmiştir. Blok yapısı çizdirilen Şekil 3.1'de bu durum ifade edilmektedir.



Şekil 3.1. Çok fazlı süzgeç bankası tasarımında önerilen modelin iş akışı

Meta sezgisel algoritmaların performansını etkileyen en önemli faktörlerden biri, algoritmanın kontrol parametrelerinin belirlenmesidir. Dolayısıyla ~~a~~lt bölümlerde, PFB tasarımında ABC, qABC ve mABC algoritmalarının kontrol parametrelerinin değişiminin tasarım performansı üzerindeki etkisi incelenmiştir. Prototip süzgeçlerin tasarımında her zaman kararlı bir yapıya sahip olan FIR süzgeçlerin kullanımı tercih edilmiştir. FIR süzgeçler, sayısal işaret işleme uygulamalarında kullanılmak üzere özelleştirilmiş cihazlarda, Eşitlik 3.1'in doğrudan uygulanması sayesinde, tekrarlamaya gerek kalmadan ve programlaması daha basit bir şekilde hayata geçirilebilir [105]. Bu yapılar her zaman programlanabilir olduğundan, yineleme olmadan, yuvarlama ve taşma hatalarının kolayca üstesinden gelinebildiğinden ve FIR süzgeçlerinin tümü doğrusal bileşenler olduğundan, veri iletimi, biyomedikal, sayısal işaret işleme ve sayısal ses gibi birçok uygulamada önemlidirler. Bir FIR süzgeç tasarlanmadan önce, performans gereksinimleri belirlenmelidir. Tipik bir süzgeç belirli frekansları geçirmeli ve diğer

frekansları zayıflatmalıdır; dolayısıyla, söz konusu frekansların yanı sıra gerekli kazanç ve zayıflamaların tam olarak tanımlanması gerekir. Tez çalışmasında prototip süzgeç için alçak geçiren FIR süzgeç kullanılmıştır. Alçak geçiren süzgeçler düşük frekansları, sıfırdan belirli bir kesme frekanslarını geçirmek ve yüksek frekansları bloke etmek üzere kullanılırlar [19]. Bu amaçla, tasarım problemi, bir prototip süzgecin çok fazlı bileşenleri kullanılarak frekans alanında gerçek yanıt ile ideal yanıt arasındaki ortalama karesel hata (Mean Squared Error, MSE) olarak formüle edilmiştir. MSE, gözlemlenen gerçek değerler ile modelin ürettiği tahmini değerler arasındaki sapmaların karelerinin ortalama değerinin alınmasıyla elde edilir ve bu sapmaların büyüklüğüne dair bir fikir vermektedir. Düşük bir MSE değeri, modelin verileri yüksek doğrulukla tahmin ettiğini ve dolayısıyla güvenilir olduğunu göstermektedir. Yüksek MSE değerleri ise modelin verileri yeterince iyi tahmin edemediğini, hatalı veya düşük performanslı olabileceğini işaret etmektedir. Optimizasyon ve makine öğrenmesi gibi alanlarda, modelin veya algoritmanın geliştirilmesi sürecinde MSE, bir kayıp fonksiyonu olarak kullanılır ve bu değerin minimize edilmesi amaçlanır; böylece, modelin ya da algoritmanın parametreleri en doğru tahminleri gerçekleştirecek biçimde ayarlanabilir. Hangi hata ölçütünün tercih edileceği, başvurulan modelin niteliklerine ve incelenen veri setinin yapısına göre değişiklik göstermektedir.

Prototip süzgeç tasarımında kullanılan FIR sayısal süzgecin matematik ifadesi Eşitlik 3.1.'de verilmiştir,

$$y(k) = \sum_{m=0}^{M-1} w_m x(k-m) \tag{3.1}$$

Eşitlikte giriş işareti $x(k)$, çıkış işareti $y(k)$, w_m süzgeç katsayılarından oluşan bir vektördür ve M süzgecin derecesini belirtmektedir. Süzgeç katsayı vektörü Eşitlik 3.2'deki gibidir.

$$w = \left[b_0 \; b_1 \; ... b_{M-1}\right] \tag{3.2}$$

Eşitlik 3.3'te $e(k)$ hata işaretidir ve süzgeç tasarımında kullanılan algoritmaya geri beslenir. FIR sayısal süzgecin katsayı vektörü algoritma tarafından ayarlanır ve amaç fonksiyonunun minimizasyonu gerçekleştirilir. Süzgeç tasarımında, bir süzgecin başarımını değerlendirmede ortalama karesel hata kullanılabilir. Bu durumda süzgecin ürettiği çıkışın, arzu edilen çıkışa ne kadar yakın olduğunu belirlemek için karesel

farkların ortalaması alınır. Bu hesaplama, süzgecin etkinliğini ölçmede temel bir kriter olarak işlev görmektedir. Eşitlikte *E* beklenen değeri temsil etmektedir.

$$J(w) = \frac{1}{K}\sum_{k=1}^{K}\left[d(k) - y(k)\right] = E\left[\left|e(k)\right|^2\right] \tag{3.3}$$

3.2. ABC Algoritması ile Süzgeç Tasarımı

ABC Algoritmasının kontrol parametresi değişikliklerinin çok fazlı süzgeç bankası tasarımının performansı üzerindeki etkileri araştırıldığında; algoritmanın kontrol parametrelerinin belirlenmesinin sezgisel algoritmaların çalışma performansını doğrudan etkileyen en önemli faktörlerden biri olduğu sonucuna varılmıştır. Bu sebeple algoritmanın temel parametrelerinden olan popülasyon ve limit değerleri alt bölümlerde detaylı olarak incelenmiştir.

3.2.1. Popülasyon Parametresinin İncelenmesi

Algoritmanın kontrol parametrelerini incelemeye yönelik olarak ilk gerçekleştirilen çalışmada, süzgeç derecesinin literatürde tasarımda sıklıkla tercih edilen bir değer olması dolayısıyla 32 olarak alınması tercih edilmiştir.-Bu duruma göre popülasyon büyüklüğü, sınır değeri ve maksimum çevrim sayısı gibi kontrol parametrelerine farklı değerler verilerek algoritma 50 kez çalıştırılmış ve elde edilen parametre değerleri Tablo 3.1'de verilmiştir [104-107]. Tabloda ABC Algoritması için popülasyon büyüklüğünün optimizasyon performansı üzerindeki etkisi değerlendirilmektedir. Çalışma sırasına göre altı farklı senaryo sunulmuş ve her biri için popülasyon büyüklüğü, maksimum çevrim sayısı, ortalama, standart sapma, minimum, maksimum ve MSE değerleri verilmiştir. Algoritmanın en uygun kontrol parametreleri araştırılırken, öncelikle en uygun koloni sayısının seçilmesi göz önüne alınmıştır. Bu amaçla koloni sayısı sırasıyla 10, 30, 50 ve 100 olarak seçilmiştir. Limit değeri bu durumda rasgele olarak 600 seçilmiş ve sırasıyla çevrim sayıları koloni sayıları için 600, 300, 200, 150, 120, 60 olarak sabitlenmiştir. Her bir koloni sayısı için toplam 600 değerlendirme yapılmış ve algoritma da 50 kez koşulmuştur.

Gerçekleştirilen incelemeye göre aşağıdaki sonuçlar elde edilmiştir.

- Popülasyon Büyüklüğü: Algoritmanın her bir çalıştırılmasında kullanılan çözüm sayısını gösterir. Popülasyon büyüklüğü 10'dan başlayarak 100'e kadar artmaktadır.

- Maksimum Çevrim Sayısı: Algoritmanın her bir senaryoda kaç iterasyon çalıştırıldığını belirtir. Bu değer arttıkça maksimum çevrim sayısı azalmıştır, bu da daha büyük popülasyonlarla daha az iterasyona ihtiyaç duyulduğunu gösterir.
- Ortalama Değer: Her bir senaryodaki optimizasyon sonucunun ortalama değeridir. Popülasyon büyüklüğü arttıkça ortalama değer genel olarak artmıştır.
- Standart Sapma: Sonuçların ne kadar dağıldığını gösteren istatistiksel bir ölçüttür. Yüksek standart sapma değerleri, sonuçların ortalama etrafında daha fazla varyasyona sahip olduğunu gösterir. Tablo 3.1'de, popülasyon büyüklüğü arttıkça standart sapmanın genelde arttığı görülmektedir.
- Minimum ve Maksimum Değer: Her senaryoda elde edilen en düşük ve en yüksek optimizasyon sonuçlarını gösterir. Bu değerler arasındaki farkın büyük olması, algoritmanın tutarlılığının düşük olduğunu gösterebilir. Popülasyon büyüklüğü arttıkça hem minimum hem de maksimum değerler artmıştır.
- Ortalama Karesel Hata: Optimizasyonun başarısını ölçen bir değerdir. Düşük MSE değerleri, algoritmanın problemin optimum çözümüne daha yakın sonuçlar ürettiğini gösterir. Tablo 3.1'de, popülasyon büyüklüğü arttıkça MSE'nin genel eğiliminin yükseldiği görülmekte, bu durum da algoritmanın daha büyük popülasyonlarla çalıştırıldığında daha az kararlı olduğunu veya belirli bir popülasyon büyüklüğünden sonra daha az etkili olduğunu göstermektedir.

Tablo 3.1, genel olarak, ABC algoritmasının farklı popülasyon büyüklüklerine ve iterasyon sayılarına nasıl tepki verdiğini ve performansının bu parametrelerle nasıl değiştiğini göstermektedir. Bu bulgular, ABC algoritması ile süzgeç bankası optimizasyon problemi çözmek isteyen araştırmacılar için de faydalı olacaktır.

Tablo 3.1. ABC algoritmasının popülasyon büyüklüğünün incelenmesi

Popülasyon Büyüklüğü	**10**	30	50	100
Maksimum Çevrim Sayısı	600	200	120	60
Ortalama	**0,0923**	0,1915	0,3305	0,7908
Standart Sapma	**0,012**	0,0276	0,0671	0,1555
Minimum	**0,0666**	0,137	0,1893	0,4928
Maksimum	**0,1306**	0,2636	0,4715	1,3041
MSE	**0,0087**	0,0374	0,1136	0,6491

Ancak, bu sonuçların genelleştirilebilirliği ve hangi senaryonun en iyi olduğu hakkında kesin bir yargıya varmak için daha fazla analiz yapmak gerekmektedir. Tablo 3.1'de gösterildiği gibi popülasyon büyüklüğünü artırmak ABC algoritmasının performansını önemli ölçüde artırmamıştır. Popülasyon büyüklüğü 10 olarak seçildiğinde ortalama, standart sapma ve en iyi hata değerleri sırasıyla 0.0923, 0.012 ve 0.0087 olup diğer popülasyon sayılarına göre daha küçük bulunmuştur. Popülasyon büyüklüğünün 10 olarak seçilmesi durumunda algoritma performans değerlerinin daha iyi sonuç verdiği görülmüştür. Bu çalışmada elde edilen sonuçlara göre popülasyon büyüklüğünün diğer çalışmalarda da 10 olarak alınmasına karar verilmiştir.

3.2.2. Limit Parametresinin İncelenmesi

ABC algoritmasında "limit" parametresi, bir çözümün yani bir besin kaynağının ne kadar süre boyunca iyileştirilemediğini belirleyen bir sayıdır. Bir çözüm belirli bir "limit" değerine ulaştığında, bu çözüm terk edilir ve keşif aşamasında yeni bir çözüm aranmaya başlanır. Bu parametre, algoritmanın yerel minimumlarda takılıp kalmamasını sağlamak ve arama alanında çeşitliliği korumak için önemlidir. Yani, bir çözüm belirli bir süre boyunca iyileşmiyorsa, arama başka bölgelere yönlendirilir. Dolayısıyla bu parametre de algoritmanın performansını doğrudan etkilemektedir. Çok düşük limit değeri, iyi çözümlerin erken terk edilmesine, çok yüksek limit değeri ise algoritmanın kötü çözümlerde fazla zaman harcamasına yol açmaktadır. Bu yüzden, limit değerinin probleme ve algoritmanın diğer parametrelerine uygun şekilde ayarlanması gerekmektedir.

ABC Algoritması için popülasyon büyüklüğü analizinden sonra ikinci parametre analizi limit değeri için yapılmıştır. Tablo 3.2'de popülasyon büyüklüğünün sabit tutularak limit değerinin incelenmesi durumunda elde edilen değerler verilmektedir. Tabloda popülasyon büyüklüğünün 10 ve maksimum çevrimin 1200 olması durumunda elde edilen değerler görülmektedir. Limit parametresinin incelemesine yönelik olarak gerçekleştirilen çalışmada aşağıdaki sonuçlar elde edilmiştir:

- Çalışma Sırası: Farklı limit değerleri için 10 farklı çalışmayı temsil etmektedir.
- Popülasyon Büyüklüğü: Her çalışmada kullanılan işçi arı ve gözcü arı sayısının toplamını gösterir ve sabit bir değer olarak 10 seçilmiştir.
- Limit: ABC algoritmasında, bir çözümün iyileştirilmediği maksimum iterasyon sayısını temsil eder. Bu değer 32'den başlayarak 320'ye kadar arttırılmıştır.

- Maksimum Çevrim Sayısı: Her bir senaryoda algoritmanın kaç iterasyon çalıştırıldığını gösterir ve deneysel analizler sonucunda tüm çalışmalarda sabit olarak 1200 olarak belirlenmiştir.
- Ortalama Değer: Her bir çalışma sırasında elde edilen sonuçların ortalamasıdır. Limit değeri arttıkça ortalama değerlerde genel bir düşüş trendi gözlenmemektedir; bu, limit artışının ortalama üzerinde belirgin bir etkiye sahip olmadığını gösterebilir.
- Standart Sapma: Sonuçların dağılımının bir ölçüsüdür. Limit değeri arttıkça standart sapma genel bir dalgalanma gösterse de analizlere etki eden ayırt edici bir eğilim görülmemektedir.
- Minimum ve Maksimum Değer: Her çalışma için elde edilen en düşük ve en yüksek performans değerlerini gösterir. Bu değerlerin geniş bir aralıkta olması, algoritmanın farklı limit değerlerine karşı tutarlılığının değişebileceğine işaret edebilir. Limit değeri arttıkça minimum ve maksimum değerlerde belirgin bir farklılık gözlemlenmemiştir.

Tablo 3.2. Sabit popülasyon büyüklüğüne bağlı olarak limit parametresinin etkisinin incelenmesi

Limit	Ortalama	Standart Sapma	Minimum	Maksimum	MSE
32	2.2252	0.3431	1.3389	2.848	5.0668
64	2.1871	0.3874	1.2692	2.957	4.9306
96	2.2079	0.3244	1.4922	2.7634	4.9778
128	2.321	0.2919	1.7449	2.754	5.4707
160	2.2769	0.29	1.5171	2.7594	5.2667
192	2.1944	0.3119	1.0593	2.7093	4.9107
224	2.2649	0.3909	1.2671	3.0759	5.2796
256	2.2553	0.3425	1.4605	2.8516	5.2013
288	2.3178	0.2777	1.2809	2.9188	5.4478
320	**0.0646**	**0.0069**	**0.0491**	**0.0779**	**0.0042**

Tablo 3.2'deki son satır dışındaki veriler, limit artışı ile genel bir performans değişikliği olmadığını gösteriyor gibi görünürken, son sütunda ortalama ve standart sapmanın önemli ölçüde düştüğü ve minimum ve maksimum değerlerin belirgin şekilde iyileştiği görülmektedir. Ancak son sütundaki değerlerin diğerlerinden önemli ölçüde düşük olması, bu limit değerinde algoritmanın çok daha iyi performans gösterdiğine işaret etmektedir. 10 farklı çalışmada sonuncu sıradaki parametre değerleri algoritma popülasyon sayısı 10 olarak sabit tutulduğunda limit değeri 320 iken daha iyi sonuçlar

üretmiştir. Böylece popülasyon ve limit parametreleri üzerinde gerçekleştirilen incelemelerden limit değerinin 320 ve popülasyon büyüklüğünün de 10 olarak alınmasına karar verilmiştir.

3.3. mABC Algoritması ile Süzgeç Tasarımı

Bu bölümde mABC Algoritması, prototip süzgeç tasarımında, süzgeç katsayılarını optimize etmek için kullanılmaktadır. Süzgeçler, sayısal işaret işlemede yaygın olarak kullanılır ve belirli bir frekans yanıtını gerçekleştirmek üzere tasarlanmış katsayılara sahiptir. mABC algoritması süzgeçlerin tasarımında genellikle frekans yanıtı, grup gecikmesi, kazanç ve diğer özellikler açısından belirli gereksinimleri karşılayacak şekilde katsayıları ayarlamak için kullanılır. Bu katsayılar, optimizasyon problemi için değişkenler olarak tanımlanır. Algoritma süzgeç katsayıları setleriyle başlar. Bu setler, potansiyel çözüm adayları olarak görülür ve başlangıçta rastgele seçilir. Daha sonra süzgeç katsayıları seti değerlendirilir ve yerel arama teknikleri kullanarak bu set iyileştirmeye çalışılır. Bu aşamada, süzgeç tasarımının performansını ölçen bir hedef fonksiyon kullanılır. Bu süreçte, daha iyi performans gösteren süzgeç katsayıları tercih edilir. Eğer tasarımı gerçekleştirilen süzgeç belirli bir süre boyunca iyileşmezse, rastgele yeni bir set oluşturulur, bu da algoritmanın çeşitliliğini artırır ve yerel optimumlardan kaçınmayı sağlar. Sonuçta, mABC algoritması en uygun süzgeç katsayılarını bulmuş olur. Bu katsayılar, istenen frekans yanıtını sağlayan ve diğer performans kriterlerini karşılayan bir süzgeç oluşturmak için kullanılabilir. Bu yaklaşım, özellikle karmaşık veya çoklu kriterlere dayalı tasarım gereksinimleri söz konusu olduğunda geleneksel yöntemlere göre daha esnek ve etkili bir çözüm sağlamaktadır. Tasarımlarda algoritmanın optimal MR değerinin belirlenmesi algoritmanın performansını maksimize etmek için kritik öneme sahiptir. Alt bölümde algoritmanın MR kontrol parametresini değerlendirmeye yönelik gerçekleştirilen inceleme verilmektedir.

3.3.1. MR Parametresinin İncelenmesi

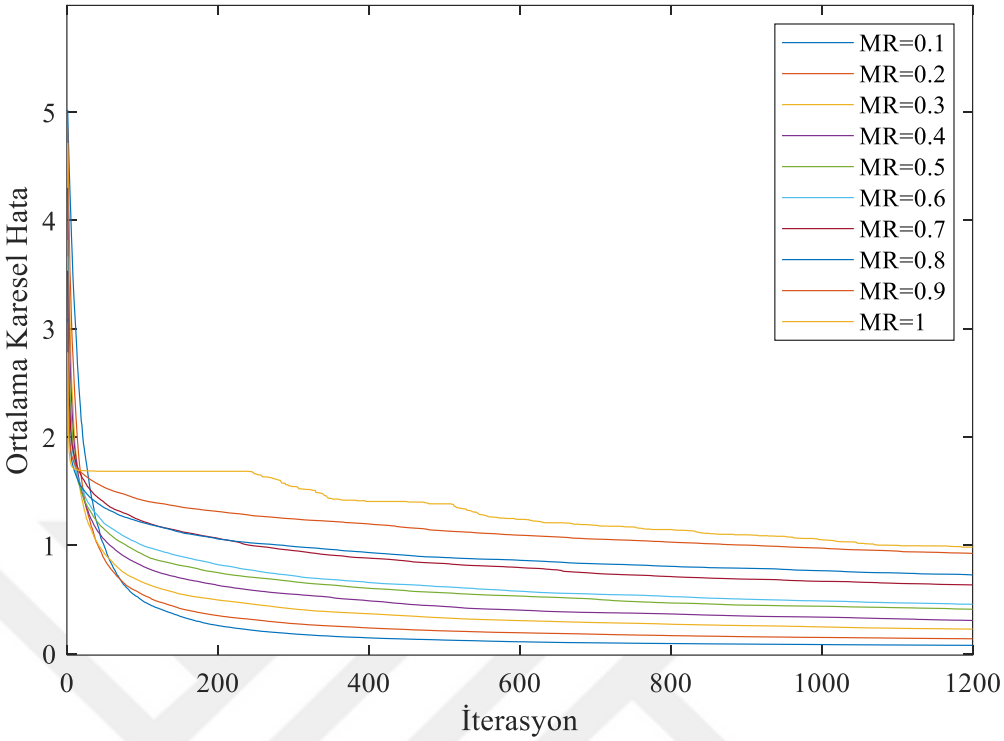
Bu bölümde farklı MR değerleri için mABC algoritmasının performansı değerlendirilmiştir. Tablo 3.3'te elde edilen sonuçlar verilmektedir. Burada mABC algoritması için popülasyon sayısı 10 (NP=10), Limit değeri 320 ve Maksimum çevrimde 1200 olarak alınmıştır.

Tablo 3.3. mABC algoritmasında MR parametresinin incelenmesi

Parametre	**Algoritmanın performans değerleri**									
MR	**0.1**	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
Ortalama Değer	**0.0763**	0.1368	0.2260	0.3061	0.4100	0.4557	0.6345	0.7265	0.9253	0.9828
Standart Sapma	**0.0124**	0.0466	0.0887	0.1061	0.1610	0.1415	0.2305	0.2409	0.3037	0.1822
En İyi Değer	**0.0557**	0.0742	0.1097	0.1590	0.1490	0.2161	0.3038	0.3653	0.4332	0.6332
Zaman (s)	**4.7078**	5.3687	5.3463	5.3556	5.3187	5.3465	5.333	5.3412	5.3249	5.3334

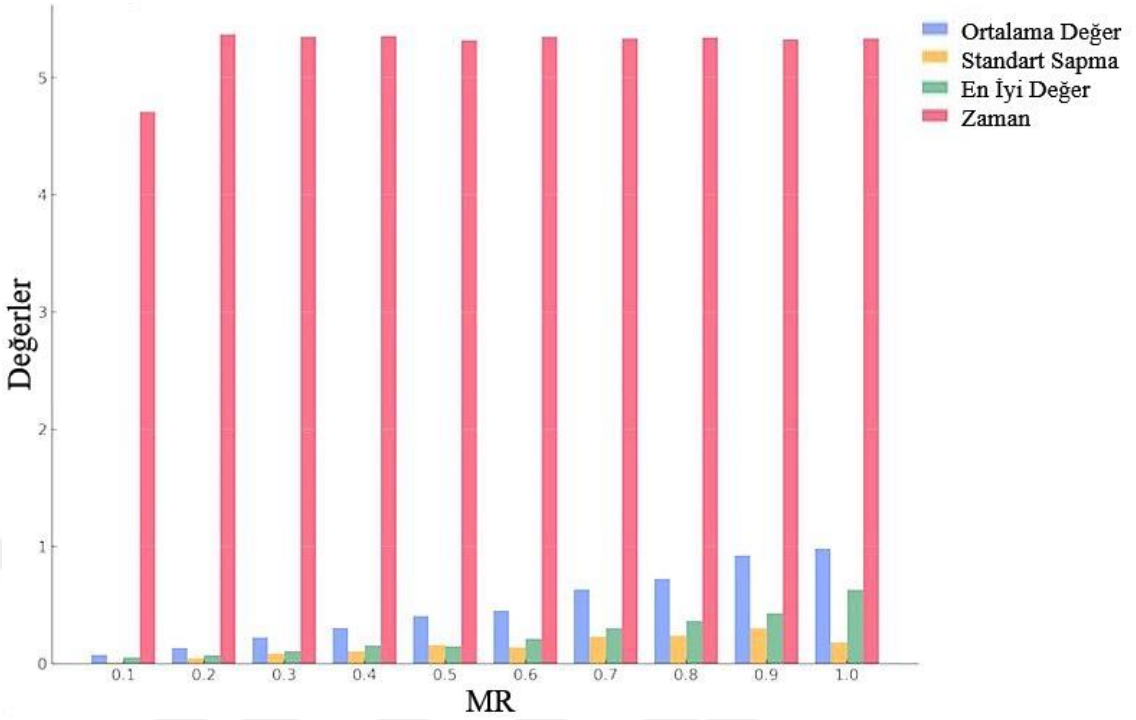
Tablo 3.3'e göre MR değeri arttıkça, ortalama, standart sapma ve zaman değerleri de artmaktadır. Buna göre, algoritmanın daha fazla değişim ile daha uzun süre çalıştığını, ancak daha kötü ve daha değişken sonuçlar verdiğini göstermektedir. Bu nedenle, MR değerinin çok yüksek veya çok düşük olmaması gerekmektedir. Tablodaki verilere göre, MR=0.5 için en iyi değerler elde edilmiştir. Burada elde edilen sonuç, algoritmanın bu değer için en iyi çözüm kalitesine ulaştığını göstermektedir. Ancak, bu değerin elde edilmesi için gereken zaman da oldukça fazladır. Algoritmanın hız performans dengesini sağlamak için MR değerinin daha düşük bir seviyeye ayarlanması istenebilir. Örneğin, MR=0.1 için geçirilen zaman değeri daha düşüktür, ancak en iyi değer çok fazla değişmemiştir. MR'nin 0.1 olarak seçilmesi, bu değerin daha iyi bir seçim olabileceğini göstermektedir.

Şekil 3.2'de mABC algoritmasının farklı MR değerleri için elde edilen ortalama hata değerlerinin iterasyona göre değişimi çizdirilmiştir. Şekil 3.2'den anlaşılacağı gibi MR değeri arttıkça, ortalama hata değeri de artmaktadır. Bu durum, algoritmanın daha fazla mutasyon ile daha kötü sonuçlar verdiğini göstermektedir. Ayrıca, MR değeri arttıkça, ortalama hata değerinin değişimi de daha yavaş olmaktadır. Bu değişim, algoritmanın daha fazla mutasyon ile daha yavaş yakınsadığını göstermektedir. Bu nedenle, MR değerinin çok yüksek veya çok düşük olmaması gerektiği sonucuna varılabilir. MR=0.1 için en düşük ortalama hata değeri elde edilmiştir. Bu durum, algoritmanın en iyi yakınsama performansına ulaştığını göstermektedir.



Şekil 3.2. Farklı MR değerleri için mABC algoritması kullanılması durumunda iterasyona bağlı MSE'nin değişimi

Şekil 3.3'te mABC algoritmasının MR değerlerine göre performansı verilmiştir. Şekil 3.3'e göre MR değeri arttıkça, algoritmanın ortalama hatası da artmaktadır. En düşük ortalama hata MR değeri 0.1 iken, en yüksek ortalama hata MR'nin 1 olduğu durumda elde edilmiştir. Yani MR arttıkça algoritmanın tutarlılığı azalmaktadır. Standart sapma, sonuçların yayılımını ve tutarlılığını gösterir. MR değeri arttıkça standart sapma genel bir artış eğilimi göstermiş, bu da sonuçların tutarlılığının azaldığını göstermiştir. En düşük standart sapma MR değeri 0.1'de, en yüksek standart sapma ise MR değeri 0.9'da iken görülmüştür. En iyi değer algoritmanın elde ettiği en iyi performansı gösterir. En iyi değer genel olarak MR arttıkça artmıştır. Ancak, MR'nin 1 olması durumunda en yüksek değere ulaşmıştır, bu veriler de bu değerde algoritmanın en kötü performansı sergilediğini göstermiştir. MR değeri arttıkça algoritmanın çalışma süresi yani zaman da artmıştır, ancak bu artış doğrusal değildir ve MR değeri 0.7'den sonra koşma zamanı da azalmıştır. Bu durum, algoritmanın belirli MR değerlerinde daha verimli çalıştığını gösterebilir.



Şekil 3.3. mABC algoritmasının MR parametresinin farklı değerlerine göre performansı

Genel olarak, MR değerinin artması ile algoritmanın hata oranları ve çalışma süresi de artmıştır. Ancak, bu eğilimler her performans değeri için farklılık göstermektedir ve algoritmanın MR değerine duyarlılığını göstermektedir.

3.4. qABC Algoritması ile Süzgeç Tasarımı

Karaboğa ve Görkemli'nin sunduğu qABC yöntemi, standart ABC algoritmasına kıyasla daha hızlı bir yakınsama sürecine sahiptir, bu da sonuca daha az sayıda adımda ulaşılmasını sağlar. Algoritmanın hızlandırılması, mevcut komşulardan en uygun olanının tercih edilmesi ilkesine dayanarak gerçekleştirilmiştir, bu sayede algoritma daha çabuk sonuca ulaşacaktır [43, 85]. Belirli bir uygulama için Rastgelelik oranı (Randomization Rate, RR) parametresinin belirlenmesi önemlidir ve optimal RR değerinin dikkatli bir şekilde seçilmesi gerekir. Alt bölümde algoritmanın RR kontrol parametresinin belirlenmesine yönelik gerçekleştirilen çalışma anlatılmaktadır.

3.4.1. RR Parametresinin İncelenmesi

qABC algoritması ABC algoritmasından farklı olarak, ABC algoritmasının rastgelelik oranını dinamik olarak ayarlayan bir mekanizma ile çalışmaktadır. Bu mekanizma, algoritmanın arama alanını daha iyi keşfetmesini ve daha hızlı yakınsamasını

sağlamaktadır. qABC algoritmasının RR parametresinin değerlendirilmesine odaklanılan bu çalışmada N=32 için elde edilen sonuçlar Tablo 3.4'te verilmiştir. Tabloda qABC algoritmasının tasarımda kullanılan diğer parametre değerleri sabitken sadece RR değerleri değiştirilerek elde edilen sonuçlar gösterilmektedir. Bu çalışmada aşağıdaki sonuçlar elde edilmiştir:

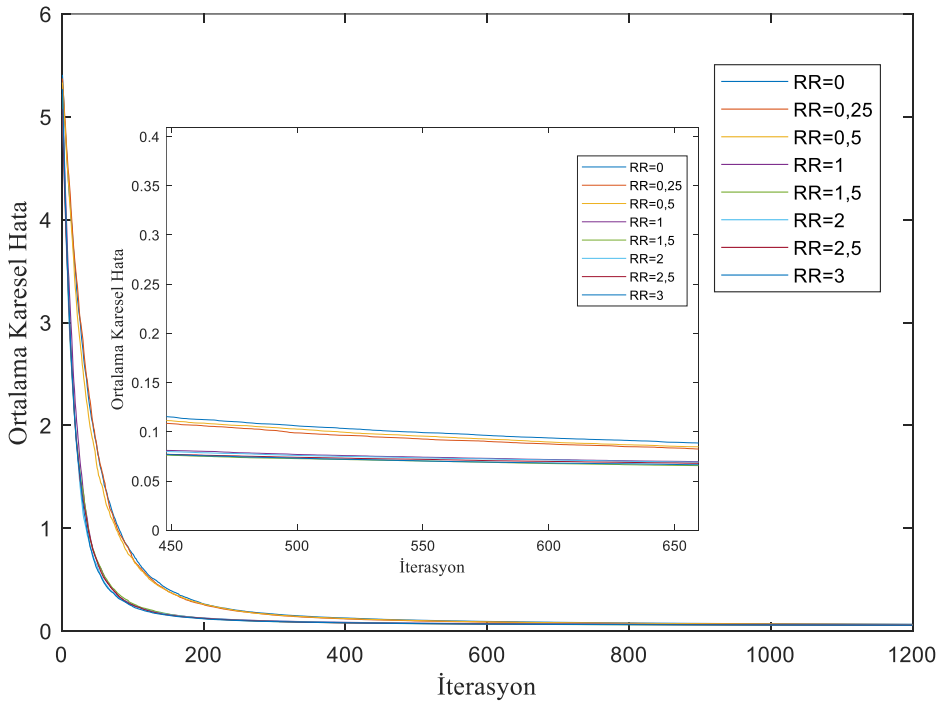
- NP: Arı kolonisinin boyutu, yani arama alanındaki çözüm adayı sayısı.
- Limit: Bir kaynağın terk edilmesi için izin verilen maksimum deneme sayısı.
- Maksimum Çevrim Sayısı: Algoritmanın durdurma kriteri, yani maksimum iterasyon sayısı.
- RR: Rastgelelik oranı, algoritmanın arama alanını ne kadar rastgele keşfedeceğini belirleyen bir parametre.
- Ortalama Değer: Farklı RR değerleri için ortalama uygunluk değeri, yani optimizasyon problemine göre çözüm kalitesi.
- Standart Sapma: Farklı RR değerleri için uygunluk değerlerinin standart sapması, yani çözüm kalitesinin değişkenliği.
- En İyi Değer: Farklı RR değerleri için elde edilen en iyi uygunluk değeri, yani en iyi çözüm kalitesi.
- Zaman: Farklı RR değerleri için algoritmanın çalışma süresi, bu süreler saniye cinsinden verilmiştir.

Tablo 3.4'e göre RR değeri arttıkça, ortalama değer, standart sapma ve zaman değerleri de artmaktadır. Bu, algoritmanın daha fazla rastgelelik ile daha uzun süre çalıştığını, ancak daha kötü ve daha değişken sonuçlar verdiğini göstermektedir. Bu nedenle, RR'nin çok yüksek veya çok düşük olmaması gerektiği sonucuna varılmıştır. Burada NP=10, Limit=320 ve Maksimum çevrim sayısı=1200 şeklinde ABC algoritmasında belirlenen parametre değerleriyle aynı olarak alınmıştır.

Tablo 3.4. qABC algoritmasında RR parametresinin incelenmesi

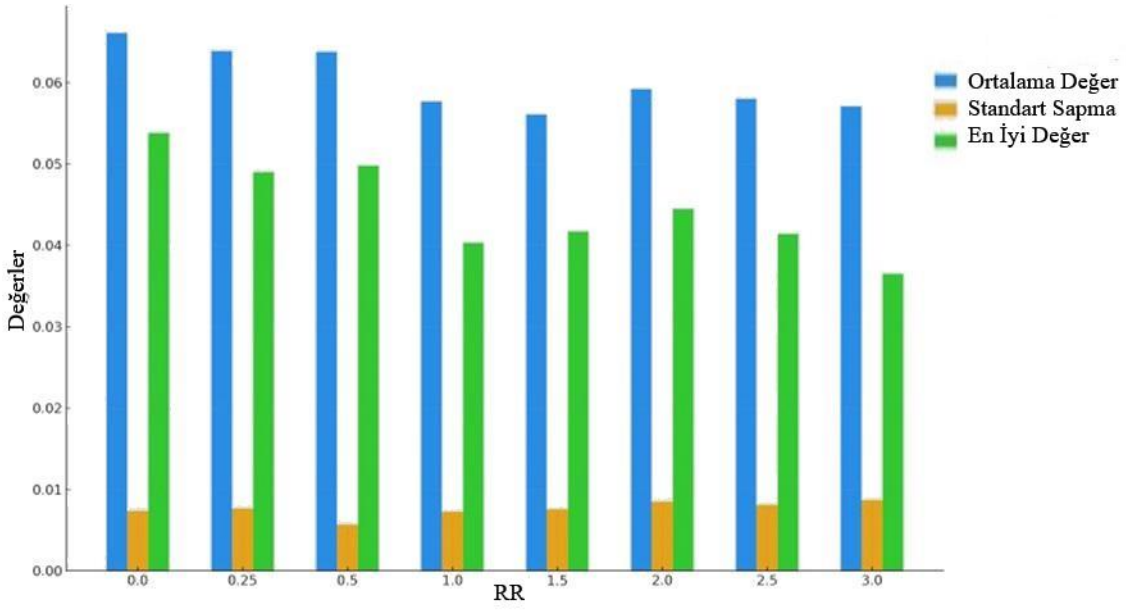
Parametre	**Parametrelerin almış olduğu değerler**							
RR	0	0.25	**0.5**	1	**1.5**	2	2.5	**3**
Ortalama Değer	0.0661	0.0639	0.6380	0.0577	**0.0561**	0.0592	0.0580	0.0571
Standart Sapma	0.0074	0.0077	**0.0057**	0.0073	0.0076	0.0085	0.0081	0.0087
En İyi Değer	0.0538	0.0490	0.0498	0.0403	0.0417	0.0445	0.0414	**0.0365**
Zaman (s)	5.7000	5.7434	5.7151	5.6973	**5.6806**	5.7972	5.991	5.6978

Bu sonuçlara göre RR'nin değeri 0.5 olarak alındığında standart sapma açısından en iyi sonuç elde edilmiştir. En kısa çalışma süresi ile en iyi ortalama değere RR'nin 1.5 alınması durumunda ulaşılmıştır. Tablo 3.4'e göre, RR=1 için en iyi değer elde edilmiştir. Bu, algoritmanın bu değer için en iyi çözüm kalitesine ulaştığını göstermektedir. Ancak, bu değer için geçirilen zaman değeri de oldukça yüksektir. Bu nedenle, algoritmanın hız performans dengesini sağlamak için RR değerinin daha düşük bir seviyeye ayarlanması daha iyi sonuçlar üretebilecektir. Örneğin, RR=1.5 için zaman değeri daha düşük, ancak en iyi değer çok fazla değişmemiştir. Bu durum, RR'nin 0.25 olmasının daha iyi bir seçim olabileceğini göstermektedir. RR parametresinin, rastgelelik oranı anlamına geldiği düşünülürse alacağı değer de algoritmanın arama alanını ne kadar rastgele keşfedeceğini belirlemektedir. RR çok yüksekse, algoritma daha fazla çeşitlilik sağlar, ancak bu durum yakınsama hızını yavaşlatır. RR değeri çok düşükse, algoritma daha hızlı yakınsar, ancak yerel optimumlara takılma riski artar. Bu nedenle, bu değerin uygun bir şekilde ayarlanması, algoritmanın performansını etkiler. RR'nin nasıl belirleneceği, optimizasyon problemine ve arama alanının özelliklerine bağlıdır. Genellikle, RR değeri 0 ile 1 arasında bir sayı olarak seçilir [43].



Şekil 3.4. Farklı RR değerleri için qABC algoritması ile iterasyona bağlı MSE'nin değişimi

Şekil 3.4'te qABC algoritmasının farklı RR değerleri için ortalama hata değerleri çizdirilmiştir. qABC algoritması ile elde edilen ortalama hata değerlerinin farklı RR değerleri için iterasyon sayısına göre değişiminin gösterildiği bu şekilde farklı RR değerlerinin algoritmanın yakınsaması üzerindeki etkisi gösterilmiştir. Şekil 3.4 incelendiğinde, RR değeri arttıkça, ortalama hata değeri de artmaktadır. Bu durum, algoritmanın daha fazla rastgelelik ile daha kötü sonuçlar verdiğini göstermektedir. Ayrıca, RR değeri arttıkça, ortalama hata değerinin değişimi de daha yavaş olmaktadır. Bu değişim, algoritmanın daha fazla rastgelelik ile daha yavaş yakınsadığını göstermektedir. Bu nedenle, RR değerinin çok yüksek veya çok düşük olmaması gerektiği sonucu çıkmaktadır. RR'nin 0 ve 1 olması durumunda en düşük ortalama hata değeri elde edilmiştir. Şekil 3.5'te qABC algoritmasının RR parametresinin farklı değerlerine göre performansı çizdirilmiştir. Şekil 3.5'e göre RR değeri arttıkça, ortalama hata değerlerinde genel bir düşüş eğilimi görülmektedir. En düşük ortalama hata, RR değeri 1.5 olduğunda elde edilmiş, bu değer algoritmanın belirli bir RR değerinde en iyi performansını sergilediğini göstermiştir. Standart sapma, sonuçların ne kadar tutarlı olduğunu gösterir. En düşük standart sapma 0.5 RR değerinde görülmektedir, bu da bu RR değerinde algoritmanın sonuçlarının en tutarlı olduğu anlamına gelmektedir. RR arttıkça, standart sapma genel olarak artmış, bu durum da sonuçların daha az tutarlı hale geldiğini ifade etmektedir. En iyi değer algoritmanın ulaşabileceği en iyi performansı gösterir.



Şekil 3.5. qABC algoritmasının RR parametresinin farklı değerlerine göre performansı

RR değeri 3'te en iyi değer elde edilmiş, bu da algoritmanın bu RR değerinde en iyi potansiyel sonuca ulaştığını göstermiştir. Zaman, algoritmanın çalışma süresini gösterir ve genel olarak, RR değeri arttıkça çalışma süresi de artmıştır. En kısa süre RR'nin sıfır olması durumunda elde edilmişken, en uzun süre RR'nin 2.5 olduğu durumda görülmektedir. Ancak, zaman değerlerinin birbirine yakın değerler olarak elde edildiği, büyük dalgalanmaların olmadığı da gözlemlenmiştir. Genel olarak, RR değerinin artması bazı ölçütlerde iyileşme gösterirken ortalama hata ve en iyi değer, standart sapma ve zaman performansın azaldığını veya tutarlılığın korunmadığını göstermiştir.

3.5. Farklı Derecelerden Süzgeç Tasarımı

ABC tabanlı algoritmaların etkinliğini değerlendirmek için farklı derecelerden süzgeç tasarımı konusu bu bölümde incelenmiştir. Algoritmaların süzgeç derecelerine göre çalışma süresi, ortalama standart sapma, en iyi hata değeri ve zaman ölçütleri açısından incelemeler gerçekleştirilmiştir. En iyi ortalama değere sahip süzgeç derecesi için ayrıntılı incelemeler konunun devamında ayrı bir alt başlıkta değerlendirilmiştir. Ayrıca qABC ve mABC algoritmasının farklı RR ve MR değerleri ile elde ettiği sonuçlar incelenmiştir.

Simülasyonlar Intel(R) Core (TM) i3-1005G1 CPU 1.20GHz, 8192 MB RAM özelliklerine sahip bir bilgisayarda Matlab ortamında gerçekleştirilmiştir.

3.5.1. ABC, mABC ve qABC Algoritmaları ile Tasarım

Bu alt bölümde ABC, mABC ve qABC gibi optimizasyon algoritmalarının, kontrol parametrelerinin değerlendirilmesine yönelik olarak önceki bölümlerde gerçekleştirilen uygulamalarda elde edilen en iyi kontrol parametreleriyle, sayısal süzgeç tasarımındaki etkinliği incelenmiştir. Gerçekleştirilen çalışmada tasarlanmak istenen süzgece ait parametreler süzgeç derecesi, ortalama, en iyi hata değeri, standart sapma ve zaman olarak kullanılmıştır. Performans metrikleri arasında kullanılan "en iyi hata değeri" algoritmanın çalışma süresi boyunca bulunan en iyi çözümün değerini almaktadır. Burada her bir çalıştırma için elde edilen en iyi çözümün objektif fonksiyon değerini saklar. En iyi çözümün objektif fonksiyon değeri "en iyi hata değeri" değişkenine atanır ve bu değer, algoritmanın tüm çalışma süresi boyunca bulduğu en iyi çözümün kalitesini gösterir.

Tablo 3.5'te kullanılan sezgisel algoritmalarla tasarlanan farklı dereceden sayısal süzgeçlerle elde edilen sonuçlar verilmektedir. Deneysel çalışmalar, ABC'nin geçiş bandı frekansı, durdurma bandı frekansı ve süzgeç derecesi açısından en iyi başarıya sahip olduğunu göstermektedir. mABC algoritması, artan performansa yol açan geliştirmeleri içeren ABC algoritmasının bir varyasyonudur. Bununla birlikte, simülasyon çalışmalarında mABC'nin ABC'ye göre farklılıklar içerdiği fakat çok daha başarılı sonuçlar elde edilememiştir. Bu mABC algoritmasının belirli durumlarda daha etkili olduğunu göstermiştir. qABC'nin ABC'den daha düşük bir performans derecesine sahip olduğu da anlaşılmıştır.

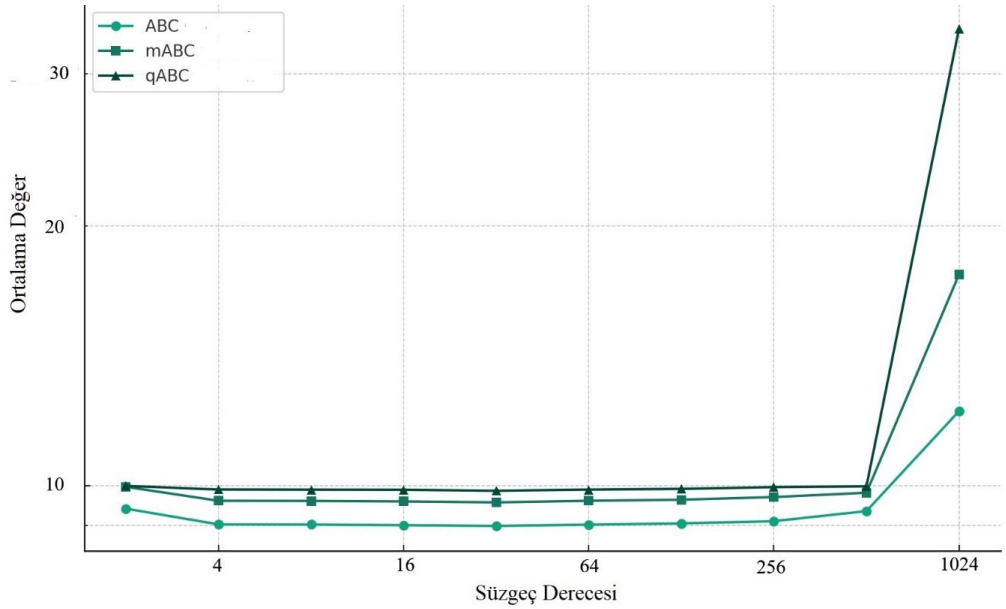
Böylece, ABC algoritmasının çalışma süresi, ortalama karesel hata, geçiş bandı frekansı, durdurma bandı frekansı ve süzgeç derecesi optimizasyonunda en etkili olduğu sonucuna varılmıştır. qABC ve mABC algoritmaları elde edilen sonuçlar ışığında en küçük hata değerlerine ulaşma bakımından qABC algoritması mABC algoritmasına göre daha iyi sonuçlar vermiştir. Örneğin süzgeç derecesi 64 alındığında qABC algoritması kullanıldığında ortalama, standart sapma ve en iyi hata değeri sırasıyla 0.0602, 0.0084 ve 0.0454 olarak elde edilmiştir. Aynı süzgeç derecesinde sadece ABC algoritması en kısa zamanda çözüme ulaşmış, diğer tüm değerlerde qABC algoritması en iyi sonuçları vermiştir.

Şekil 3.6'da ABC, mABC ve qABC algoritmalarının süzgeç derecelerine göre ortalama değer bakımından karşılaştırmalı performans analizi çizdirilmiştir. Şekle göre daha düşük süzgeç derecelerinde, tüm algoritmalar kabul edilebilir sonuçlar verirken, süzgeç derecesi arttığında qABC ve ABC'nin mABC'ye göre daha iyi performans gösterdiği görülmüştür.

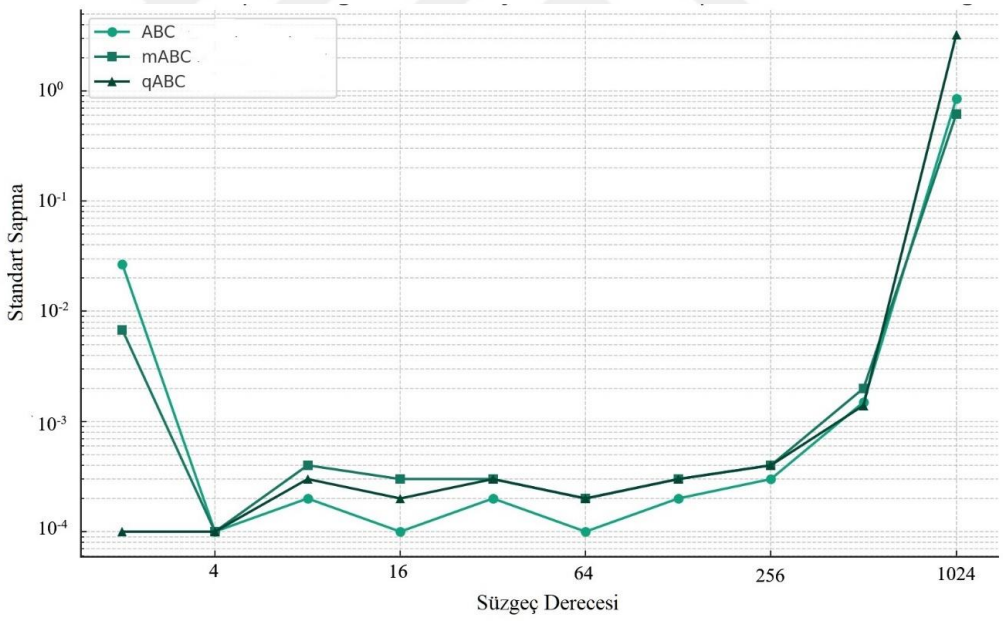
Şekil 3.7'de ABC, mABC ve qABC algoritmaları için süzgeç derecelerine göre standart sapma açısından karşılaştırmalı bir performans analizi çizdirilmiştir. Şekil 3.7'ye göre süzgeç derecesinin düşük olduğu durumlarda, kullanılan tüm meta sezgisel algoritmalar kabul edilebilir değerler vermişlerdir. Süzgeç derecesinin artmasına bağlı olarak ortalama değer ile incelenen duruma benzer şekilde standart sapma açısından da qABC ve ABC algoritması ile elde edilen sonuçların mABC algoritması ile elde edilen sonuçlara göre daha iyi olduğu anlaşılmıştır.

Tablo 3.5. ABC, mABC ve qABC algoritmalarının süzgeç derecelerine göre karşılaştırmalı performans analizi

Süzgeç Derecesi	Algoritmaların Performans Değerleri	ABC	mABC	qABC
2	Ortalama	0.9409	0.9972	0.9996
	Standart Sapma	0.0268	0.0068	0.0001
	En İyi Hata Değeri	0.9077	0.9711	0.9996
	Zaman (s)	**5.4280**	5.7279	5.7758
4	Ortalama	0.3363	0.3363	0.3363
	Standart Sapma	**0.0001**	**0.0001**	**0.0001**
	En İyi Hata Değeri	0.3363	0.3363	0.3363
	Zaman (s)	5.4336	**5.7200**	**5.4669**
8	Ortalama	0.0920	0.0902	0.0905
	Standart Sapma	0.0081	0.0047	0.0039
	En İyi Hata Değeri	0.0805	0.0828	0.0826
	Zaman (s)	5.4691	6.2666	5.6253
16	Ortalama	0.0718	**0.0714**	0.0695
	Standart Sapma	0.0051	0.0063	0.0068
	En İyi Hata Değeri	0.0605	**0.0558**	0.0562
	Zaman (s)	5.4533	5.8048	5.6024
32	Ortalama	**0.0658**	0.0809	**0.0579**
	Standart Sapma	0.0065	0.0219	0.0084
	En İyi Hata Değeri	**0.0498**	0.0586	0.0407
	Zaman (s)	5.4577	5.8943	5.6229
64	Ortalama	0.0908	0.2970	0.0602
	Standart Sapma	0.0100	0.1012	0.0084
	En İyi Hata Değeri	0.0637	0.1212	**0.0454**
	Zaman (s)	5.4755	5.8515	5.6637
128	Ortalama	0.2133	1.6362	0.0913
	Standart Sapma	0.0531	0.4105	0.0140
	En İyi Hata Değeri	0.1458	0.9704	0.0673
	Zaman (s)	5.5435	5.8716	5.6967
256	Ortalama	1.7407	8.9614	0.2900
	Standart Sapma	0.5318	1.4493	0.0429
	En İyi Hata Değeri	0.7627	5.9800	0.2067
	Zaman (s)	5.5171	5.9102	5.6728
512	Ortalama	29.7486	31.6977	3.4043
	Standart Sapma	3.6347	2.1835	0.5696
	En İyi Hata Değeri	23.0279	26.1841	2.3860
	Zaman (s)	5.5373	6.0184	5.7937
1024	Ortalama	147.6231	88.5218	45.3785
	Standart Sapma	8.2465	5.6146	3.2353
	En İyi Hata Değeri	122.8353	75.7604	38.3211
	Zaman (s)	5.6641	6.1540	5.913



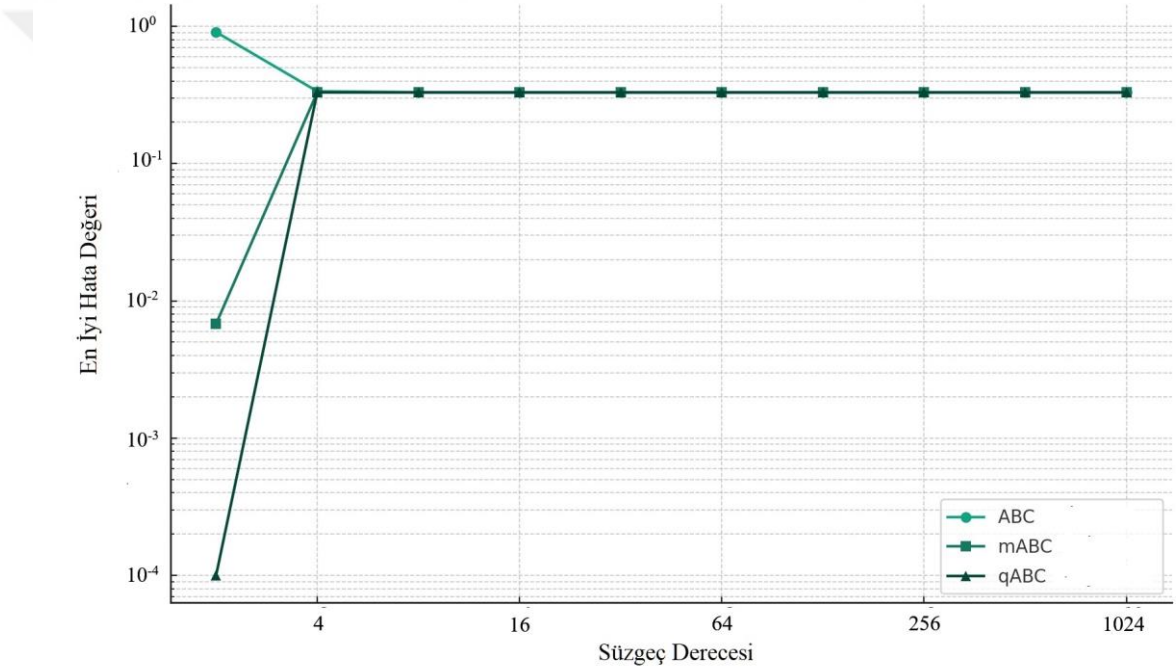
Şekil 3.6. Süzgeç derecelerine göre ABC, mABC, qABC algoritmalarının ortalama değerler açısından karşılaştırılması



Şekil 3.7. Süzgeç derecelerine göre ABC, mABC, qABC algoritmalarının standart sapma değerleri açısından karşılaştırılması

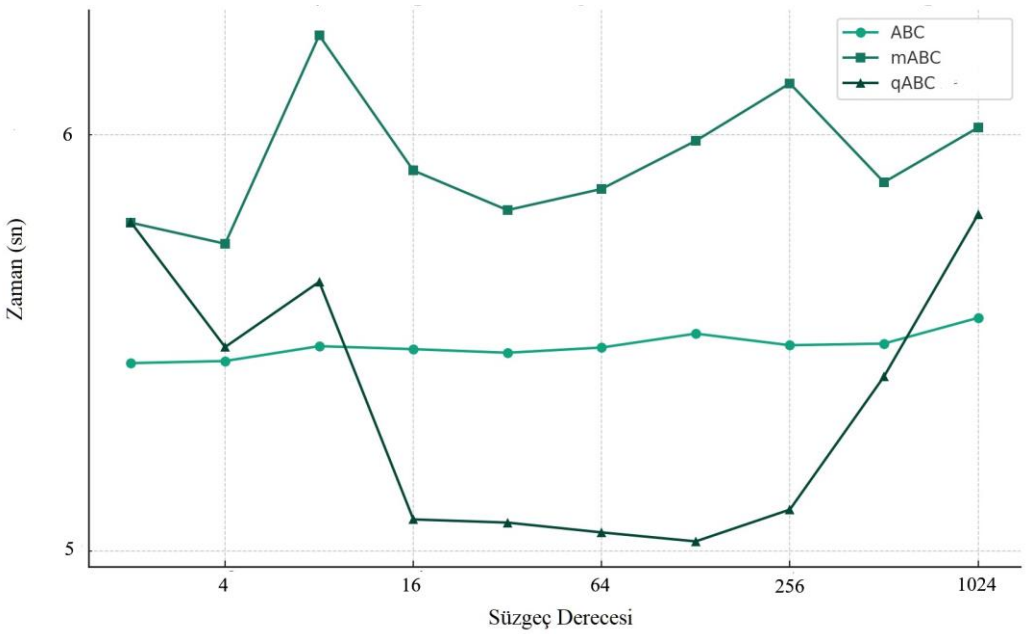
Şekil 3.8'de ABC, mABC ve qABC algoritmalarının süzgeç derecesinin artmasına bağlı olarak en iyi hata değeri açısından çizimi yapılmıştır. Şekilden daha düşük süzgeç derecelerinde, tüm algoritmaların sonuçlarının kabul edilebilir değerler olduğu ifade edilebilir. Fakat süzgeç derecesi arttıkça qABC ve ABC algoritmalarının mABC

algoritması ile elde edilen değerlerden daha iyi sonuçlar elde etmektedir. Tablo 3.5'teki verilere göre süzgeç derecesi 16 olduğunda üç algoritma içerisinde standart sapma değeri 0.0051 ve 5.4533 saniye ile en kısa zamanda ABC algoritması en iyi sonucu vermiştir. En iyi hata değeri 0.0558 ile mABC algoritmasıdır. Süzgeç derecesi 32 olduğunda da benzer bir performansla üç algoritma içerisinde standart sapma değeri 0.0065 ve 5.4577 saniye ile en kısa zamanda ABC algoritması en iyi sonucu vermiş ve en iyi hata değeri de 0.0407 olduğundan qABC algoritması ile elde edilmiştir. Bu şekildeki bir değerlendirmeden de anlaşılacağı üzere algoritmanın kontrol parametrelerinin doğru belirlenmesi, süzgecin performansını arttıran önemli faktörlerden birisidir.



Şekil 3.8. Süzgeç derecelerine göre ABC, mABC, qABC algoritmalarının en iyi hata değerleri açısından karşılaştırılması

Şekil 3.9'da süzgeç tasarımında kullanılan ABC tabanlı algoritmaların süzgeç derecelerine göre zaman açısından performansı çizdirilmiştir. ABC algoritması ile tasarımda tüm süzgeç derecelerinde yaklaşık aynı zamanlarda tasarımları gerçekleştirdiği gözlemlenirken qABC algoritmasının ABC'ye göre daha uzun zamanda tasarımları gerçekleştirdiği mABC algoritmasının ise ABC ve qABC'ye göre daha daha uzun zamanda süzgeçleri tasarladığı anlaşılmaktadır.



Şekil 3.9. Süzgeç derecelerine göre ABC, mABC, qABC algoritmalarının zaman değerleri açısından karşılaştırılması

3.5.2. En İyi Ortalama Değer için Süzgeç Tasarımı

Süzgeç derecelerinin incelendiği bölümde *N*=32 için ideal süzgeç değerlerine en yakın sonuçlar elde edildiğinden bu alt bölümde bu süzgeç derecesi için süzgeç tasarımı temel alınmış ve bu durum için ABC, mABC ve qABC algoritmalarının performansları karşılaştırılmıştır. Elde edilen en iyi kontrol parametreleriyle gerçekleştirilen tasarımlarda algoritmaların çözüm kalitesi ve yakınsama hızı üzerindeki etkileri incelenmiştir. Tablo 3.6'da, her üç algoritma için belirlenen parametre değerleri ve performans metriklerinin değerleri verilmektedir.

Tablo 3.6. *N*=32 için ABC algoritmasıyla elde edilen değerler

Parametre adı	**Parametrenin almış olduğu değerler**
Ortalama Değer	0.0633
Standart Sapma	0.0071
En İyi Değer	0.0498
Zaman (s)	5.7090

Tablo 3.6'da yer alan "en iyi değer" algoritmanın çalışma süresi boyunca bulunan en iyi çözümün değerini alır. Burada her bir çalıştırma için elde edilen en iyi çözümün objektif fonksiyon değerini saklar. En iyi çözümün objektif fonksiyon değeri "en iyi değer"

değişkenine atanır ve bu değer, algoritmanın tüm çalışma süresi boyunca bulduğu en iyi çözümün kalitesini gösterir. ABC algoritması çalıştırılırken daha önce belirlenen parametre değerleri kullanılmıştır: NP=10, Limit=320, Maksimum çevrim=1200.

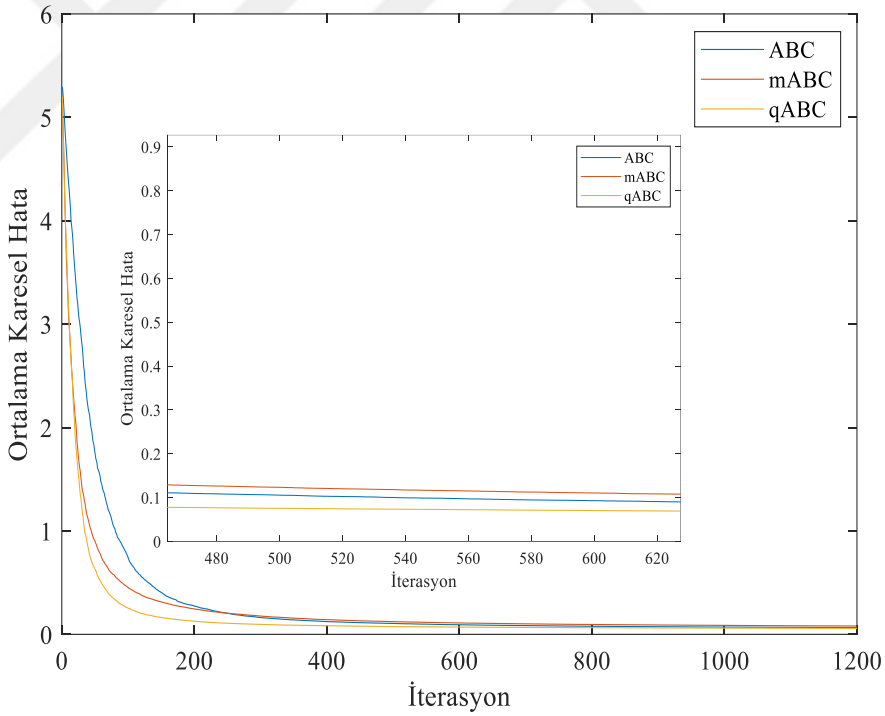
Gerçekleştirilen değerlendirmeler göre ABC algoritması ile qABC ve mABC algoritmalarından daha iyi bir ortalama değer elde etmiştir: 0.0633. Bu durum, ABC algoritmasının bu problem için daha iyi bir çözüm kalitesine ulaştığını göstermektedir. Ancak, ABC algoritmasının çalışma süresi, qABC ve mABC algoritmalarından daha uzundur. Bu nedenle, ABC algoritmasının hız performans dengesini sağlamak için parametre değerleri üzerinde daha farklı karşılaştırmaları içeren çalışmalar yapılmalıdır.

Tablo 3.3 ve Tablo 3.4'te elde edilen sonuçlar, qABC ve mABC algoritmasının farklı RR ve MR değerleri ile elde ettiği sonuçları göstermektedir. RR ve MR değerleri, algoritmanın performansını etkileyen önemli parametrelerdir. RR ve MR değerleri çok yüksekse, algoritma daha fazla çeşitlilik sağlar, ancak yakınsama hızını yavaşlatır. RR ve MR değerleri çok düşükse, algoritma daha hızlı yakınsar, ancak yerel optimumlara takılma riski artar. Bu nedenle, RR ve MR değerlerinin uygun bir şekilde ayarlanması, algoritmanın performansını doğrudan etkilemektedir. Tablolara göre, RR=1 ve MR=0.5 için en iyi değerler elde edilmiştir. Bu, algoritmanın bu değerler için en iyi çözüm kalitesine ulaştığını göstermektedir. Ancak, bu değerler için zaman da oldukça yüksektir. Bu nedenle, algoritmanın hız performans dengesini sağlamak için RR ve MR değerlerini daha düşük bir seviyeye ayarlamak isteyebiliriz. Örneğin, RR=0.25 ve MR=0.1 için zaman daha düşük, ancak en iyi değer çok fazla değişmemiştir. Bu, bu değerlerin daha iyi bir seçim olabileceğini göstermektedir.

Şekil 3.7'de N=32 için iterasyon sayısına bağlı ortalama karesel hata grafiği verilmiştir. Tüm algoritmalar başlangıçta yüksek MSE değerleriyle başlamış ve ilk 200 iterasyon içinde hızlı bir düşüş göstermiştir. Bu, algoritmaların başlangıçta hızlı bir iyileşme kaydettiğini ve incelenen performans değeri açısından hızla başarıya ulaşmıştır. İlk büyük düşüşten sonra MSE değerleri neredeyse sabit bir seviyeye yerleşmiştir. mABC ve qABC algoritmalarının MSE değerleri neredeyse örtüşmektedir ve ABC algoritmasına göre hafifçe daha düşük bir MSE değerine sahiptir. Bu, mABC ve qABC algoritmalarının bu problem için biraz daha iyi performans gösterdiğini göstermiştir. Grafikte yer alan küçük iç grafik, 500 ile 600 iterasyonlar arasında algoritmaların MSE değerlerindeki minimal değişiklikleri daha detaylı göstermektedir. Bu iç grafikte, tüm algoritmaların MSE

değerlerinde herhangi bir önemli iyileşme veya bozulma olmadan kararlı bir seviyede kaldığı görülmektedir. Genel olarak, grafikteki veriler, üç algoritmanın da 32. derece bir süzgeç tasarımı problemi için benzer ve olası çözümler sağladığını göstermektedir.

Şekil 3.10'da süzgeç derecesi 32 olduğunda algoritmaların performansları karşılaştırılmış ve ABC, mABC, qABC algoritmaları içerisinde standart sapma ve çözüm için gereken süre bakımından ABC algoritması en iyi sonucu vermiştir. En iyi hata değeri açısından qABC algoritması daha başarılı olmuştur. Şekil 3.10'da N=32 için aynı parametre değerlerinde ABC, qABC ve mABC algoritmalarıyla iterasyona bağlı MSE'nin değişimi verilmiştir. Şekilden iterasyon sayısının algoritmanın yakınsama davranışı üzerindeki etkisi değerlendirildiğinde iterasyon sayısı arttıkça, ortalama hata değerinin de azaldığı gözlemlenmiştir. Bu değişim, algoritmanın daha iyi sonuçlara yakınsadığını göstermektedir. Ayrıca, şekle göre, ortalama hata değeri 100 iterasyondan sonra çok az değişmektedir.



Şekil 3.10. N=32 için ABC qABC ve mABC algoritmalarıyla iterasyona bağlı olarak MSE'nin değişimi

Bu durum, algoritmanın bu noktadan sonra daha fazla iyileşme sağlayamadığını göstermektedir. Bu nedenle, iterasyon sayısının çok yüksek olmasının algoritmanın performansını artırmayacağı görülmüştür. Grafiğe göre, ortalama hata değeri 500

iterasyonda yaklaşık 0.5'e ulaşmaktadır. Bu durum algoritmanın bu değer için en iyi yakınsama performansına ulaştığını göstermektedir.

3.6. Farklı Geçirme Bandına Sahip Süzgeçlerin Tasarımı

Sayısal süzgeçler, istenmeyen frekans bileşenlerini ayıklamak ve yararlı işaret bölümlerini vurgulamak için kullanılmaktadır. Süzgeç tasarımı süreci, genellikle belirli bir frekans aralığını geçirirken diğerlerini bloke edecek şekilde frekans yanıtının şekillendirilmesini içermektedir. Analog ve sayısal bölgelerde tasarım için farklı teknikler ve yaklaşımlar bulunmaktadır. Analiz sırasında, düşük geçiren, yüksek geçiren, bant geçiren ve bant durduran süzgeçler gibi çeşitli süzgeç tiplerinin karakteristik özellikleri, belirli bir uygulamanın gereksinimlerini karşılayacak şekilde incelenmesi gerekmektedir. Süzgeç performansının frekans aralığına göre değerlendirilmesi, süzgeç tasarım parametrelerinin seçimini ve işaretin karakteristiklerini anlamak için kritik önem taşımaktadır. Farklı frekans aralıklarında süzgeçlerin performanslarını incelemek, işaret işleme ve haberleşme sistemleri tasarımında temel bir öneme sahiptir. Özellikle, frekans yanıtı, bir süzgecin ne kadar iyi süzgeçlediğini yani istenmeyen frekansları ne kadar etkin bir şekilde reddettiğini ve istenilen frekansları ne kadar az bozulma ile ilettiğini gösterir. İşaret gürültü oranı, faz kayması, grup gecikmesi ve genlik yanıtı gibi çeşitli metrikler ile ölçülür. Gürültü bastırma, harmonik bozulma, faz hatası ve geçici yanıt gibi faktörler, süzgecin performansını etkileyen diğer önemli parametrelerdir. İdeal süzgeç davranışına yaklaştıkça, pratikte karşılaşılan zorluklar ve tasarım zorlukları ortaya çıkar. Bu zorluklar arasında, süzgeç devrelerinin karmaşıklığı, maliyeti ve fiziksel boyutu ile ilgili teknik ve ekonomik sınırlamalar bulunmaktadır. Ayrıca, farklı frekans bantlarındaki performans özelliklerinin incelenmesi, süzgeç tasarımı için çeşitli optimizasyon algoritmalarının etkinliğini test etmek ve bu algoritmalar arasındaki performans farklılıklarını değerlendirmek için de kullanılmaktadır.

Süzgeçlerin performanslarını farklı frekans aralıklarında incelemek, daha verimli ve etkili sistemlerin geliştirilmesinde kilit bir rol oynamaktadır. Çalışmanın bu bölümünde farklı geçirme ve durdurma bandı frekanslarına göre ABC, mABC ve qABC algoritmalarının performansları incelenmiştir.

3.6.1. ABC, mABC ve qABC Algoritmaları ile Tasarım

Süzgeç derecelerinin incelendiği kısımlarda N=32 için ideal süzgeç değerlerine en yakın sonuçlar elde edildiğinden bu bölümdeki incelemelerde bu değer kullanılmıştır. Farklı geçirme ve durdurma bandı frekanslarına göre ABC, mABC ve qABC algoritmaları ile elde edilen veriler bu alt bölümde incelenmiştir. Tablo 3.6'da, ABC, mABC ve qABC algoritmalarının farklı frekans aralıklarındaki performans değerleri verilmiştir. Her bir frekans aralığı için, algoritmaların performans değeri olarak ortalama, standart sapma, en iyi hata ve çalışma süresi temel alınmıştır.

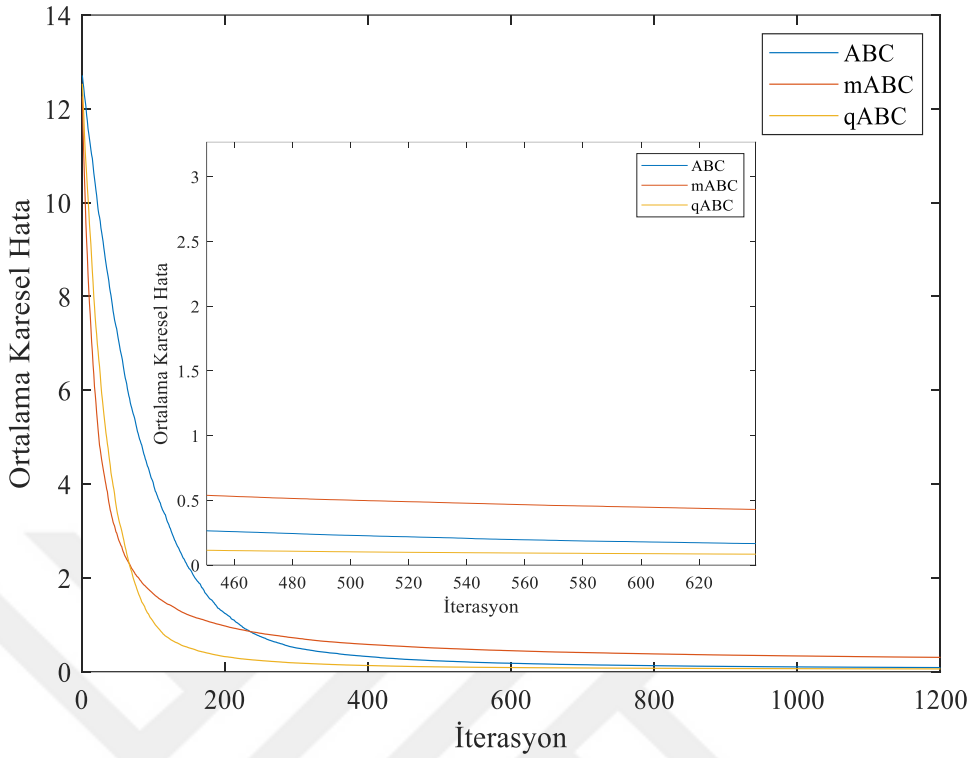
qABC algoritması, tüm frekans aralıklarında en düşük ortalama değerlerine sahiptir. Bu qABC algoritmasının süzgeç tasarımında farklı frekans aralıkları açısından diğer algoritmalara göre daha tutarlı ve genel olarak daha iyi performans sergilediğini göstermektedir. Yalnızca 0.45-0.55 frekans aralığında en küçük ortalama 0.0107 değeri ile ABC algoritmasında elde edilmiştir. Standart sapma, algoritmanın sonuçlarının tutarlılığını gösterir. qABC algoritması, dört farklı frekans aralığında en düşük standart sapma değerlerine sahiptir, bu durum da qABC'nin diğer algoritmalarla karşılaştırıldığında daha tutarlı sonuçlar ürettiğini gösterir. En iyi hata ölçütü algoritmanın en iyi performansını gösterir. Burada, qABC algoritması dört frekans aralığının tamamında en iyi hata değerini sunarak en iyi performansı göstermiştir. Bu durum, qABC algoritmasının çoğu durumda en iyi çözüme ulaşma konusunda mABC ve ABC algoritmalarından üstün olduğunu göstermektedir. Çalışma süresi açısından, qABC algoritması, 0.3-0.7 frekans aralığı için 4.2527 saniye, 0.45-0.55 frekans aralığı için 5.2979 saniye ile en düşük çalışma süresine sahiptir.

Diğer frekans aralıklarında en düşük çalışma süresi değerleri ABC algoritmasında elde edilmiştir. Bu verilere göre, genel olarak ABC algoritmasının hız açısından başarılı olduğu görülmüştür. Genel olarak, qABC algoritması en düşük ortalama ve standart sapmaya sahip olması ve çoğu frekans aralığında en düşük çalışma süresine sahip olmasıyla, tabloda gösterilen verilere dayanarak diğer iki algoritmaya kıyasla üstün performans sergilemiştir. Ancak, ABC algoritması da belirli durumlarda en kısa çalışma süresini sağlayarak güçlü bir performans sergilemiştir. Bu sonuçlar, süzgeç tasarımında kullanılan algoritmaların seçiminde önemli bir rol oynamaktadır.

Tablo 3.7. Geçirme ve durdurma bandı frekanslarına göre ABC, mABC ve qABC algoritmaları ile elde edilen veriler

Süzgeç Frekans Aralıkları $(w_p - w_s)$	Algoritmaların Performans Değerleri	ABC	mABC	qABC
0.1 - 0.9	Ortalama	0.0881	0.3054	0.0594
	Standart Sapma	0.0101	0.1205	0.0070
	En İyi Hata Değeri	0.0669	0.1460	0.0474
	Zaman (s)	5.2442	5.3640	5.3297
0.2 - 0.8	Ortalama	0.0853	0.2818	0.0557
	Standart Sapma	0.0115	0.0843	0.0079
	En İyi Hata Değeri	0.0587	0.1299	**0.0395**
	Zaman (s)	**5.1817**	5.3456	5.3416
0.3 - 0.7	Ortalama	0.0830	0.2981	0.0569
	Standart Sapma	0.0100	0.1041	0.0081
	En İyi Hata Değeri	0.0645	0.1400	0.0401
	Zaman (s)	5.3036	5.4197	4.2527
0.4 - 0.6	Ortalama	0.0912	0.2908	0.0596
	Standart Sapma	0.0105	**0.0991**	0.0068
	En İyi Hata Değeri	0.0618	0.1496	0.0479
	Zaman (s)	5.1859	5.3434	5.2965
0.45 - 0.55	Ortalama	0.0107	0.3159	0.0708
	Standart Sapma	0.0099	0.1041	0.0071
	En İyi Hata Değeri	0.0787	0.1181	0.0549
	Zaman (s)	5.5998	5.3541	5.2979

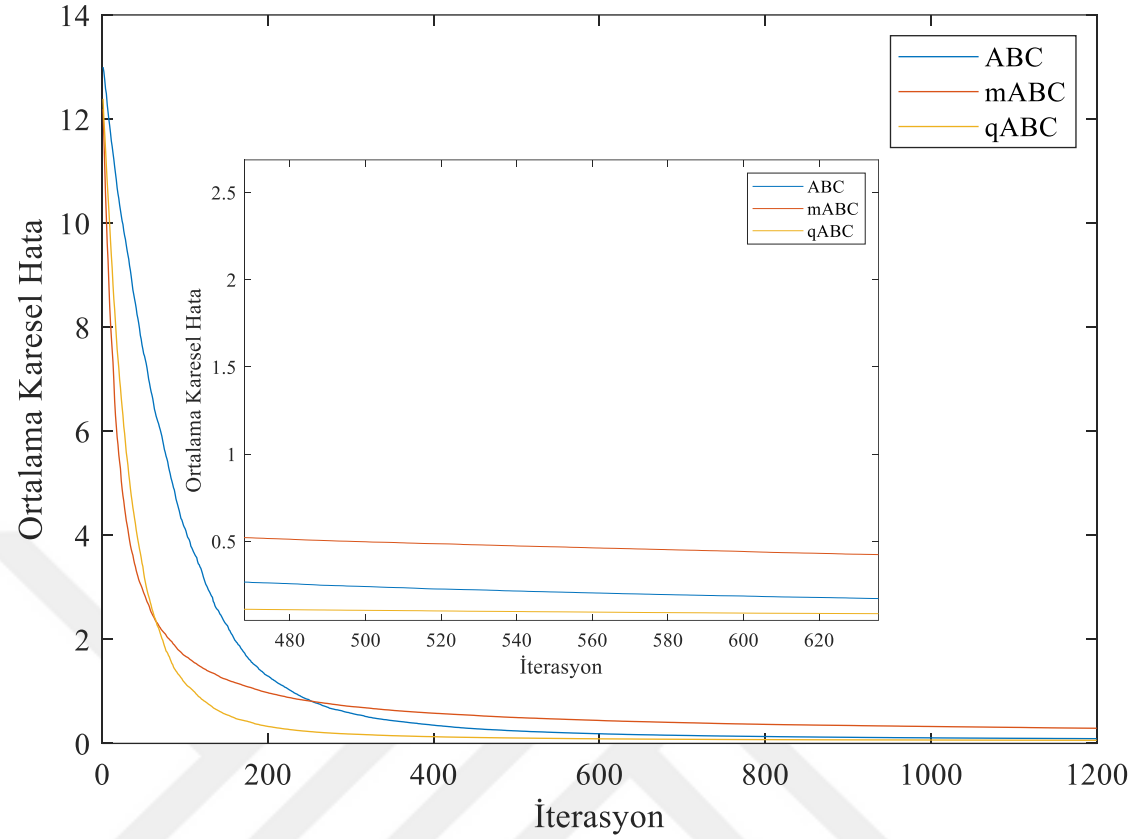
Elde edilen verilerin yorumlanmasıyla algoritmanın kontrol parametrelerinin doğru belirlenmesi, performansı artırmış ve daha doğru sonuçların elde edilmesinde önemli faktörlerden birisi olmuştur. Tablo 3.7'de yer alan "en iyi hata değeri" algoritmanın çalışma süresi boyunca bulunan en iyi çözümün değerini alır. Şekil 3.11'de Geçiş bandı frekansı 0.1ve durdurma bandı frekansı 0.9 alındığında ABC, mABC ve qABC algoritmalarıyla N'nin olması durumunda tasarlanan sayısal süzgeç için iterasyonlara bağlı olarak MSE'nin değişimi çizdirilmiştir. Tüm algoritmaların MSE değerleri başlangıçta yüksektir ve ilk 200 iterasyon içinde hızlı bir düşüş görülmüştür. Buna göre, her bir algoritmanın süzgeç tasarımı problemine hızla yakınsadığını göstermektedir. ABC algoritması, başlangıçta en yüksek MSE değerine sahiptir ve hızlı bir iyileşme gösterdikten sonra, MSE değerleri diğer algoritmaların altında kararlı bir seviyeye yerleşmiştir. Bu veriden ABC algoritmasının bu özel frekans aralığında en iyi performansı sergilediği görülmüştür.



Şekil 3.11. Geçiş bandı frekansı $w_p = 0{,}1$ ve durdurma bandı frekansı $w_s = 0{,}9$ alındığında ortalama karesel hata grafiği

mABC algoritması, başlangıçta orta seviye bir MSE ile başlamış ve hızlı bir düşüşten sonra qABC algoritmasının biraz üzerinde bir MSE değeri ile kararlı seviyeye ulaşmıştır. Bu durumda, diğer algoritmalara göre mABC orta derecede bir performans göstermiştir. qABC algoritması, ilk iterasyonlarda en düşük MSE değeri ile başlamış ve en düşük MSE değerini koruyarak ilerlemiştir. Ancak, 200 iterasyondan sonra ABC algoritması tarafından geçilmiş ve en yüksek MSE değerlerinden biri olarak kalmıştır. Tablo 3.7'ye göre zaman açısından değerlendirildiğinde sonuca en kısa sürede ABC algoritması ulaşmıştır. Tablo 3.7'deki incelendiğinde ortalama değer, standart sapma ve en iyi hata değeri açısından da en başarılı sonuçları qABC algoritması vermiştir.

Şekil 3.12'de süzgeç parametrelerinden. geçiş bandı frekansı w_p=0.4 ve durdurma bandı frekansı w_s=0.6 olarak alındığında ortalama karesel hata grafiği verilmiştir. ABC, mABC ve qABC algoritmaları başlangıçta yüksek MSE değerlerine sahiptir ve ilk 200 iterasyon içinde hızlıca düşüş göstermişlerdir. Bu veriler, her bir algoritmanın bu frekans aralığı için hızla yakınsadığını göstermiştir.



Şekil 3.12. Geçiş bandı frekansı $w_p = 0,4$ ve durdurma bandı frekansı $w_s = 0,6$ alındığında ortalama karesel hata grafiği

ABC algoritması, iterasyonların başında diğer algoritmalara göre daha hızlı düşen bir MSE değerine sahiptir ve bu avantajını iterasyon süresince koruyarak en düşük MSE değerlerini sürdürmüştür. Bu kararlı duruma göre, ABC algoritması bu özel frekans aralığında en iyi performansı göstermiştir. mABC algoritması, başlangıçta düşük bir MSE ile başlamış ve hızlıca düşerek ABC algoritmasına yakın bir değere ulaşmıştır. Ancak, ABC'nin ulaştığı değerin altında kalmaya devam etmiş ve qABC'ye kıyasla daha iyi bir performans sergilemiştir. qABC algoritması ise başlangıçta en düşük MSE değeri ile başlamasına rağmen, ABC ve mABC algoritmalarına göre daha yüksek bir MSE değeri ile devam etmiş ve grafikte en üstte yer almıştır. Bu durumda, qABC algoritmasının bu frekans aralığında diğer algoritmalara göre daha az etkili olduğu görülmüştür. Grafikteki eğilimler genel olarak incelendiğinde, ABC algoritması bu özel frekans aralığında en iyi performansı sergilemiştir. mABC algoritması da rekabetçi bir performans gösterirken, qABC algoritması bu durumda daha yüksek MSE değerleri ile biraz geride kalmıştır.

3.6.2. ABC, GA ve PSO Algoritmaları ile Tasarım

Önceki bölümlerde ABC tabanlı algoritmalarla sayısal süzgeç tasarımı konusu hem algoritmaların kontrol parametreleri hem süzgeç derecesi ve hem de geçirme ve durdurma bantları temel alınarak ayrıntılı bir şekilde incelenmiştir. Bu alt bölümde ise incelenen bölümlerde ABC algoritmasının iyi bir performans sergilemesi sebebiyle literatürde çok kullanılan PSO ve GA algoritmalarıyla sayısal süzgeç tasarımı konusu incelenmiştir. ABC, PSO ve GA algoritmalarının süzgeç derecelerine göre ortalama, ortalama karesel hata ve en iyi hata değeri açısından elde edilen sonuçlar yorumlanarak popülasyon büyüklüğü, maksimum iterasyon sayısı gibi kontrol parametreleri ABC, PSO ve GA için aynı değerlerde alınarak performans analizlerinin eşit koşullarda yapılabilmesi için karşılaştırmalar yapılmıştır. En uygun parametrenin bulunması için ABC, PSO ve GA algoritmaları değerlendirilen her bir süzgeç derecesi için 50 kez koşturulmuştur. Tablo 3.8'de, 8, 16, 32, 64, 128, 256, 512, 1024 olarak alınan farklı süzgeç derecelerinde ABC, PSO ve GA algoritmaları ile elde edilen veriler performans metrikleri verilmektedir.
Süzgeç derecesi 8 ve 16 alındığında, ABC algoritması en küçük standart sapma değerine sahiptir. PSO, ortalama hata değerleri için 0.088 ve 0.0583, en iyi hata değerleri için ise 0.0793 ve 0.0476 olarak daha düşük sonuçlar elde etmiştir. GA, süzgeç derecesi 8 ve 16 olduğunda hata değerleri sırasıyla 0.1263. 0.0226, 0.0838 ve 0.1378, 0.0316, 0.0881 olarak gerçekleşmiştir ki bu da daha yüksek hata değerlerine karşılık gelmektedir. ABC algoritması ile süzgeç derecesi 32, 64 ve 128 olmak üzere, sırasıyla 0.0633 ve 0.007; 0.0921 ve 0.0103; 0.2013 ve 0.0358 ortalama ve standart sapma değerlerinde en küçük sonuçları elde etmiştir. PSO, süzgeç derecesi 32 ve 64 için en iyi hata değerlerini 0.0318 ve 0.0692 olarak belirlerken, süzgeç derecesi 128'de ABC algoritması 0.1358 ile en düşük en iyi hata değerini elde etmiştir.

GA algoritmasının hata değerleri diğer algoritmalara göre daha yüksek değerlere sahiptir. Süzgeç derecesi 256, 512, 1024 alındığında ortalama hata ve standart sapma değerleri önemli ölçüde artmıştır. Süzgeç derecesi 256 olduğunda, ABC algoritması ortalama, standart sapma ve en iyi hata değeri için sırasıyla 1.9369, 0.6466 ve 0.6209 ile en düşük sonuçları elde etmiştir. Süzgeç derecesi 512 ve 1024'te ise, PSO ile en düşük ortalama değerleri, sırasıyla 28.1046 ve 87.8376 olarak elde edilmiştir. Süzgeç derecesi 512 ve 1024 için en küçük standart sapma değerleri GA ile elde edilmiştir.

Tablo 3.8. ABC, PSO ve GA algoritmalarının süzgeç derecelerine göre karşılaştırmalı performans analizi

Süzgeç Derecesi	Algoritmaların Performans Değerleri	ABC	PSO	GA
8	Ortalama	0.0906	0.0880	**0.1263**
	Standart Sapma	**0.0055**	0.0157	**0.0226**
	En İyi Hata	0.0816	0.0793	**0.0838**
16	Ortalama	0.0707	**0.0583**	0.1378
	Standart Sapma	0.0073	**0.0083**	0.0316
	En İyi Hata	0.0551	0.0476	0.0881
32	Ortalama	**0.0633**	0.0664	0.2679
	Standart Sapma	0.0070	0.1134	0.0613
	En İyi Hata	**0.0433**	**0.0318**	0.1283
64	Ortalama	0.0921	0.1707	0.2509
	Standart Sapma	0.0103	0.1533	0.0531
	En İyi Hata	0.0711	0.0692	0.1616
128	Ortalama	0.2013	1.0946	1.0972
	Standart Sapma	0.0358	0.5729	0.2515
	En İyi Hata	0.1358	0.2777	0.6994
256	Ortalama	1.9369	6.5348	6.2444
	Standart Sapma	0.6466	1.3681	0.7326
	En İyi Hata	0.6209	3.5657	4.5867
512	Ortalama	29.1556	28.1046	30.0897
	Standart Sapma	3.9335	3.1901	1.4146
	En İyi Hata	20.2331	21.4484	26.8783
1024	Ortalama	146.8958	87.8376	106.744
	Standart Sapma	7.6304	5.1977	3.0615
	En İyi Hata	122.0045	72.7469	100.7144

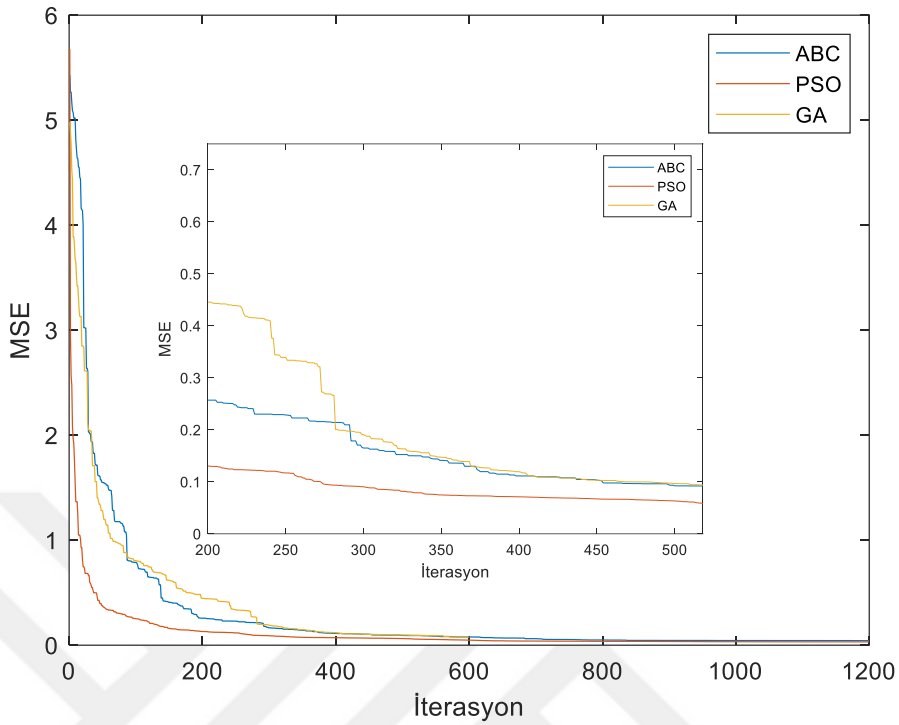
Genel olarak, süzgeç derecesinin artışıyla birlikte algoritmaların performansında bir düşüş gözlemlenmektedir. Bu değerler, algoritmaların daha yüksek süzgeç derecelerinde optimum çözümden uzaklaştığını göstermektedir. Özellikle, süzgeç derecesi 1024 için ortalama hata ve standart sapma değerleri çok yüksektir, bu koşullar altında

algoritmaların ürettiği değerlerin optimizasyona katkı sunmayacağı görülmüştür. Tablo 3.8'e göre, tüm değerler incelendiğinde ABC algoritmasının, sayısal süzgeç tasarımında daha iyi bir optimizasyon yöntemi olduğu görülmektedir. GA, sürekli olarak en yüksek hata değerlerine sahip olmuştur, bu durum da GA'nın sayısal süzgeç tasarımında daha az etkili olduğunu göstermektedir. En iyi hata değeri, süzgeç bankasının en iyi performansını göstermektedir. Tablo 3.8'e göre, ortalama, standart sapma ve en iyi hata değerleri, ABC algoritması için en düşük, GA algoritması için en yüksek, PSO algoritması için ise her iki algoritmaya göre orta seviyededir. Bu sonuca göre, ABC algoritmasının, PSO ve GA algoritmalarından daha iyi bir arama performansına sahip olduğu görülmüştür. Algoritmalar süzgeç derecesine bağlı olarak 6 farklı süzgeç derecesi için eşit koşullarda çalıştırıldıktan sonra her bir süzgeç derecesi için en küçük standart sapma değerlerinin ABC Algoritmasında elde edildiği görülmüştür.

ABC algoritması, süzgeç bankası tasarımında en iyi ortalama ve standart sapma değerleri açısından diğer algoritmalardan az bir farkla da olsa daha iyidir. Süzgeç derecesi arttıkça, ortalama, standart sapma ve en iyi hata değerleri en düşük seviyeye inmektedir.

ABC algoritmasının, PFB tasarımında en iyi optimizasyon yöntemlerinden biri olduğunu göstermektedir. PSO algoritması, PFB tasarımında GA algoritmasından iyi ABC algoritmasından daha kötü performans gösteren algoritmadır. Süzgeç derecesi arttıkça, ortalama, standart sapma ve en iyi hata değerleri azalmakta, ancak ABC algoritmasının değerlerine ulaşamamaktadır. Bu verilere göre, PSO algoritmasının, GA algoritmasından daha iyi, ancak ABC algoritmasından daha kötü bir arama performansına sahip olduğunu göstermektedir. GA algoritması süzgeç derecesi arttıkça, ortalama, standart sapma ve en iyi hata değerleri azalmakta, ancak ABC ve PSO algoritmalarının değerlerinden çok uzak kalmaktadır. Bu durumda GA algoritmasının, PFB tasarımında çok iyi bir performans göstermediği söylenebilir.

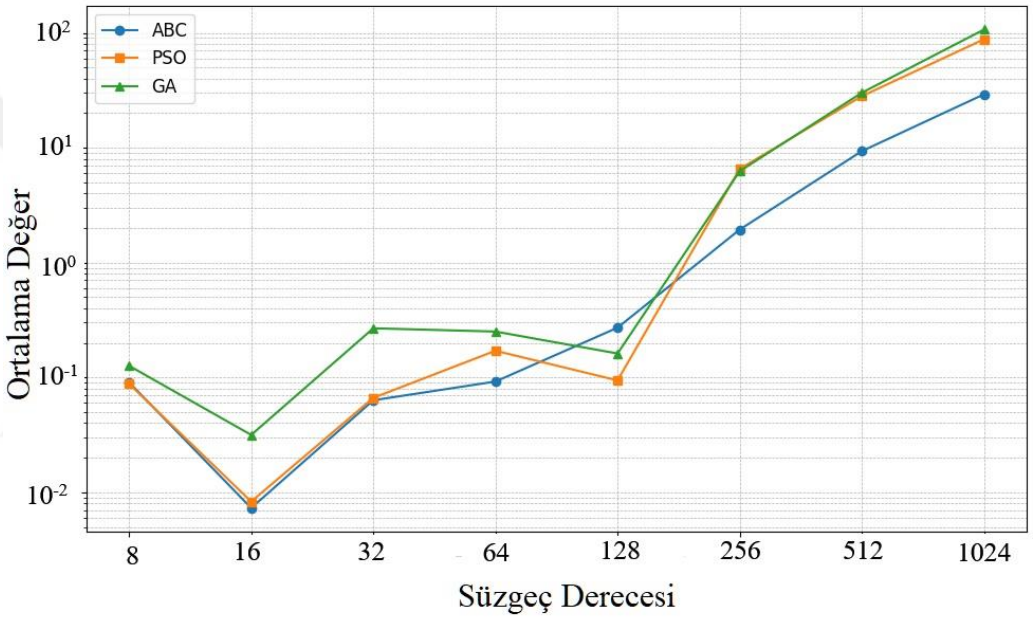
Tablo 3.8'de PSO algoritması kullanılarak yapılan tasarımların 50 koşma için hesaplanan ortalama ve ortalama karesel hata ve en iyi değerleri GA ile elde edilen değerlerden daha küçük çıkmıştır. Genel olarak değerlendirildiğinde PSO ve GA algoritmaları ile en iyi hata değerleri için benzer sonuçlar bulunurken, ABC algoritmasının en iyi ortalama ve ortalama karesel hata değerleri diğer algoritmalardan az bir farkla da olsa daha iyidir. Bu sonuçlara bakıldığında, daha düşük ortalama ve ortalama karesel hata değerleri ABC algoritması ile elde edildiğinden, ABC algoritması diğer algoritmalara göre daha başarılıdır.



Şekil 3.13. ABC, PSO ve GA algoritmalarının N=32 için yakınsama eğrileri

Şekil 3.13'te ABC, PSO ve GA algoritmalarının süzgeç derecesi değeri 32 iken iterasyon sayısına göre MSE değerlerinin nasıl değişim gösterdiği görülmektedir. MSE değeri genellikle bir algoritmanın performansını değerlendirmek için kullanılır. Yapılan analizlerde gerçek değerler ile tahmin edilen değerler arasındaki farkların karelerinin ortalaması alınarak hesaplanmaktadır. MSE değerinin düşük olması, modelin verileri iyi bir şekilde temsil ettiği ve tahminlerin gerçek değerlere yakın olduğu anlamına gelmektedir. Yüksek bir MSE değeri ise, modelin verilerden önemli ölçüde sapmalar ürettiğini ve tahminlerin gerçek değerlerden uzak olduğunu göstermektedir. Şekil 3.13'te ABC algoritması, başlangıçta hızlı bir düşüş göstermiş ve daha sonra MSE değeri oldukça düşük bir seviyeye ulaşmıştır. İlk iterasyonlardan itibaren kararlı bir yakınsama eğilimi göstermesi, algoritmanın hızlı ve etkili bir şekilde optimal çözüme ulaşabildiğini göstermektedir. PSO algoritması, ABC'ye kıyasla biraz daha yavaş başlamış, fakat sonrasında benzer bir hızda düşüş göstermiş ve ABC ile benzer bir düzeye ulaşmıştır. PSO'nun da etkili bir yakınsama performansı sergilediği, ancak ABC kadar hızlı olmadığı görülmektedir. GA algoritması, diğer iki algoritmaya göre daha yüksek MSE ile başlamış ve yavaş bir iyileşme göstermiştir. Ancak, diğerlerine kıyasla daha yüksek bir MSE değeri ile sonlanmaktadır. Bu veriler, GA'nın süzgeç tasarımı için etkili bir algoritma olmadığını

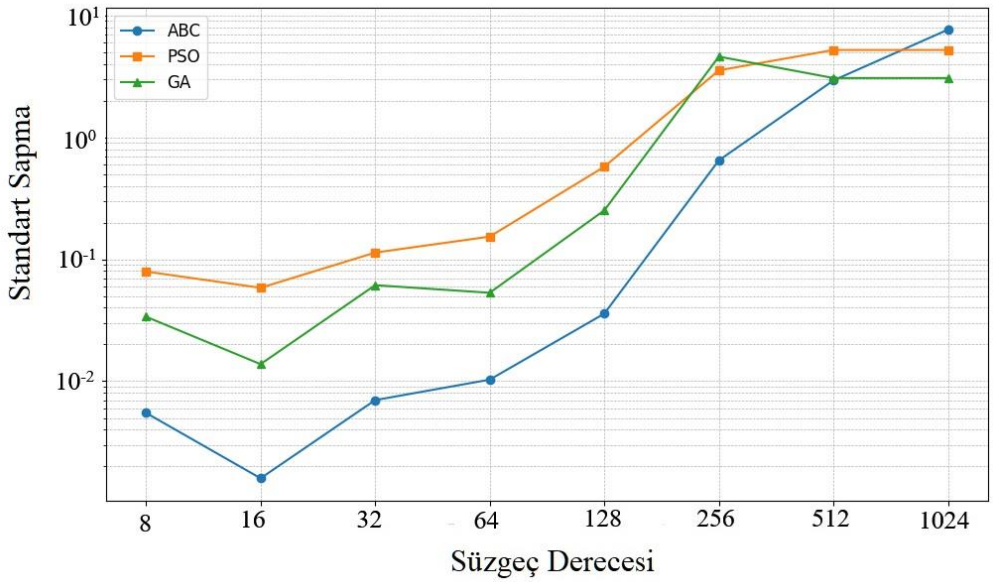
göstermektedir. Şekil 3.13'te, yaklaşık 200 iterasyon sonrasında, ABC ve PSO algoritmalarının MSE değerlerinde belirgin bir iyileşme olmamıştır. GA ise daha yavaş iyileşme potansiyeli göstermektedir. Bu bilgiler ışığında, ABC ve PSO'nun süzgeç tasarım problemi için GA'ya göre daha iyi sonuçlar verdiği söylenebilir. Ayrıca, belirli bir iterasyon sayısından sonra algoritmaların daha fazla iyileşme göstermediği ve bu noktadan sonra iterasyon sayısını artırmanın fayda sağlamadığı değerlendirilmiştir. Şekil 3.14'te ABC, PSO ve GA algoritmalarının süzgeç derecelerine göre elde edilen ortalama değerlerinin karşılaştırmalı grafiği çizdirilmiştir.



Şekil 3.14. ABC, PSO ve GA algoritmalarının süzgeç derecelerine göre elde edilen ortalama değerlerinin karşılaştırmalı analizi

Şekil 3.14'e göre daha düşük süzgeç derecelerinde, tüm algoritmalar kabul edilebilir sonuçlar verirken, süzgeç derecesi arttıkça ortalama değerler de artış göstermiştir. Şekilden PSO ve ABC algoritmasının GA'ya göre daha iyi performans gösterdiği söylenebilir. Süzgeç derecesi arttıkça, ortalama değerler de artmaktadır ki bu durum daha yüksek süzgeç derecelerindeki artışın daha karmaşık optimizasyon problemleri oluşturduğunu göstermektedir. Düşük süzgeç derecelerinde PSO algoritması en düşük ortalama değere sahiptir. Süzgeç derecesinin 32 ve 64 olduğu durumda ABC diğerlerine göre daha düşük bir ortalama değeri göstermiştir yani ABC algoritması 32. ve 64. derecelerde süzgeç tasarımında daha iyi bir performans sergilemektedir. 256, 512 ve 1024. dereceden gibi daha yüksek süzgeç dereceleri için tüm algoritmaların ortalama

değerleri yakınsamış ve benzer bir eğilim göstermiştir. Ancak GA ile genel olarak biraz daha yüksek ortalama değer elde edilmiştir.

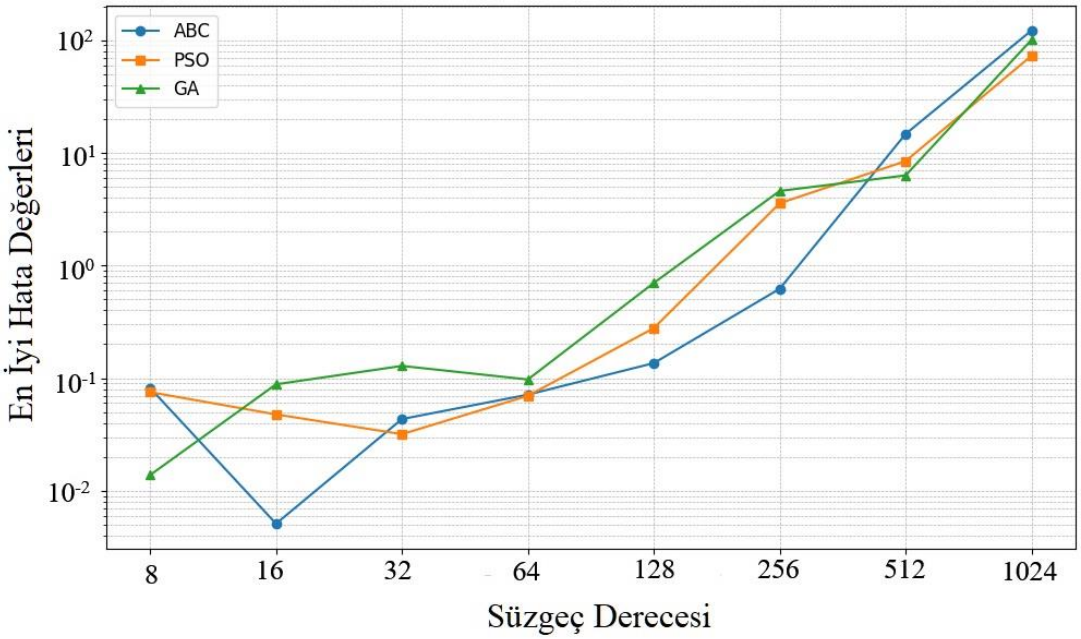


Şekil 3.15. ABC, PSO ve GA algoritmalarının süzgeç derecelerine göre elde edilen standart sapma değerlerinin karşılaştırmalı analizi

Şekil 3.15'te ABC, PSO ve GA'nın süzgeç derecelerine göre elde edilen standart sapma değerlerinin karşılaştırmalı analizi verilmiştir. Şekil 3.15'e göre daha düşük süzgeç derecelerinde, ABC ve PSO kabul edilebilir sonuçlar vermektedirler. Süzgeç derecesi arttıkça PSO ve ABC algoritmalarının GA'ya göre daha iyi sonuçlar verdiği görülmüştür. 8 ve 16 gibi daha düşük süzgeç derecelerinde, PSO algoritmasının çok daha yüksek standart sapmalara sahip olduğu görülmüştür. 32. ve 64. süzgeç derecelerinde, ABC algoritmasının standart sapması önemli ölçüde düşmüş ve bu süzgeç derecelerinde daha tutarlı sonuçlar elde edilmiştir. Tüm algoritmalar için standart sapma, süzgeç derecesi arttıkça genel olarak artış göstermiştir. 1024. Dereceden bir süzgeç tasarımında PSO ve GA benzer standart sapmalara sahipken, ABC algoritması daha düşük bir standart sapmaya sahiptir.

Şekil 3.16'da ABC, PSO ve GA'nın süzgeç derecelerine göre elde edilen en iyi hata değerleri çizdirilmiştir. Şekilden daha düşük süzgeç derecelerinde, ABC ve PSO algoritmaları ile kabul edilebilir sonuçlar elde edilirken süzgeç derecesi arttıkça en iyi hata değerleri de artış göstermiştir. PSO ve ABC algoritmalarının GA'ya göre daha iyi sonuçlar verdiği söylenebilir. Tüm algoritmalar için, süzgeç derecesinin artmasına bağlı

olarak hata değerinin de arttığı Şekil 3.16'dan anlaşılmaktadır. Bu durumda, problemin zorluğunun süzgeç derecesiyle doğru orantılı olarak arttığı ifade edilebilir.



Şekil 3.16. ABC, PSO ve GA algoritmalarının süzgeç derecelerine göre elde edilen en iyi hata değerlerinin karşılaştırmalı analizi

Süzgeç derecesi düşükken algoritmalar arasında performans açısından belirgin farklar bulunmaktadır. Süzgeç derecesi arttıkça, özellikle 128'den sonra, algoritmaların hata değerleri birbirine yaklaşmaya başlamaktadır. Bu durum, belirli bir karmaşıklık seviyesinden sonra algoritmaların benzer performans gösterdiğini göstermektedir. En yüksek süzgeç derecesi olan 1024'te, ABC ve PSO algoritmalarının neredeyse aynı hata değerine sahip olduğu, GA'nın ise biraz daha yüksek bir hata değeri sergilediği gözlemlenmiştir.

3.7. Tasarlanan Süzgeçlerin Değerlendirilmesi

Bu bölümde sayısal süzgeç tasarımında kullanılan ABC tabanlı algoritmaların yani ABC, qABC ve mABC algoritmalarının kontrol parametreleri incelenmiştir. Algoritmalar için popülasyon sayısı, ortalama değer, standart sapma, minimum maksimum değer ve MSE parametreleri açısından değerlendirmeler gerçekleştirilmiştir. Bu durumda ABC algoritmasının kontrol parametreleri popülasyon büyüklüğü için 10 ve limit değeri için de 320 olarak belirlenmiştir. Daha sonra mABC algoritmasının MR değeri incelenmiştir.

mABC algoritmasındaki incelemelerde ABC için belirlenen popülasyon büyüklüğü ve limit parametresinin değerleri önceki çalışmada belirlendiği gibi 10 ve 320 olarak kullanılmıştır. MR'nin 0.1 ile 0.1 artışlarla 1'e kadar olan tüm değerlerde incelemesi, ortalama değer, standart sapma, algoritmada elde edilen en iyi değer ve zaman temel alınarak değerlendirilmiş ve MR'nin tüm değerlerde 0.1 olması gerektiği belirlenmiştir. Son olarak qABC algoritması için RR parametre hangi değerde en iyi tasarımı verdiği araştırılmıştır. qABC algoritması için de popülasyon büyüklüğü ve limit parametresi değerleri ABC'nin kontrol parametrelerinin belirlendiği ilk çalışmadaki gibi 10 ve 320 olarak alınmıştır. mABC algoritması için kullanılan tasarım metriklerinin temel alındığı çalışmada RR'nin: 0, 0.25, 0.5, 1, 1.5, 2, 2.5, 3 olması durumunda değerlendirmeler gerçekleştirilmiştir. RR'nin değerinin 0.5 olarak alınması durumunda standart sapmanın en küçük olduğu belirlenirken RR'nin 1.5 olması durumunda daha kısa zamanlarda tasarımlar gerçekleştirilmiştir. Ayrıca RR değeri arttıkça, ortalama hata değeri de artmaktadır.

ABC, PSO ve GA ile yüksek dereceden süzgeç tasarımlarında tüm algoritmaların ortalama değerlerinin benzer bir eğilim göstermesine rağmen GA biraz daha yüksek ortalama değerler elde edilmiştir. Bu algoritmalar için süzgeç derecesi artarken standart sapma değerleri de artmaktadır. Örneğin 1024. dereceden bir süzgeç tasarımında yani yüksek dereceden süzgeç tasarımlarında PSO ve GA algoritmaları yaklaşık aynı standart sapma değerlerine sahipken ABC algoritması ile daha düşük standart sapma elde edilmiştir. Süzgeç derecesinin düşük olduğu durumda Kullanılan bu algoritmalar arasında performans açısından belirgin farklar bulunurken derece arttıkça algoritmaların hata değerlerinin birbirine yaklaşmaya başladığı görülmüştür. 1024. Dereceden bir süzgeç tasarımında ABC ve PSO algoritmalarının neredeyse aynı hata değerine sahip olduğu tespit edilirken GA'nın biraz daha yüksek bir hata değeri sergilediği gözlemlenmiştir.

4. BÖLÜM

ÇOK FAZLI SÜZGEÇ BANKASI TASARIMI

Son yıllarda sayısal işaret işleme ve iletişim sistemleri, veri aktarımının ve işlemenin karmaşıklığı nedeniyle yüksek performanslı ve verimli süzgeç bankalarına ilgiyi artırmıştır. Bu ilgi, özellikle çok fazlı süzgeç bankalarının tasarımı ve optimizasyonunda kendini göstermektedir. Çok fazlı süzgeç bankaları, DSP'de temel bir araç olarak, işaretleri birden fazla frekans bileşenine ayırma yeteneği sayesinde geniş bir uygulama alanına sahiptir. Meta sezgisel algoritmalar geniş çözüm alanlarını etkili bir şekilde tarayabilen ve karmaşık optimizasyon problemlerini çözebilen güçlü araçlardır. Bu algoritmalar, belirlenen performans kriterlerini karşılayacak şekilde süzgeç katsayılarının ayarlanmasında kullanılabilirler. DFT Süzgeç Bankaları, her bir bant geçiren süzgecin bir demodülatör ve ardından çalışan toplamlı bir alçak geçiren süzgeç olarak uygulandığı N uzunluklu FIR bant geçiren süzgeçler kullanılarak oluşturulabilen Fourier dönüşüm süzgeç bankasıdır. Bir alt bant sistemindeki önemli parçalardan biri analiz sentez süzgeç bankasıdır. Analiz süzgeç bankası giriş frekansını farklı alt bantlara bölmek için kullanılırken, sentez süzgeç bankası tüm alt bant işaretlerini birleştirerek bir tam bant işareti oluşturur. Farklı alt bant süzgeç bankası yapıları arasında modüle edilmiş süzgeç bankaları, kolay tasarımları ve kolay uygulanabilirlikleri nedeniyle en popüler olanlarıdır. Genel olarak, DFT çok fazlı süzgeç bankası ve kosinüs modülasyonlu süzgeç bankası olmak üzere iki sınıf modülasyonlu süzgeç bankası vardır [106]. Bu uygulamada aşırı örnekleme kullanılmıştır. İşaret işlemede aşırı örnekleme, bir işaretin Nyquist oranından önemli ölçüde daha yüksek bir örnekleme frekansında örneklenmesi işlemidir. Teorik olarak, bant genişliği sınırlı bir işaret Nyquist hızında veya bunun üzerinde örneklenirse mükemmel bir şekilde yeniden yapılandırılabilir. Nyquist oranı işaretin bant genişliğinin iki katı olarak tanımlanır. Aşırı örnekleme çözünürlüğü ve işaret gürültü oranını iyileştirebilir ve örtüşme önleyici süzgeç performans gereksinimlerini azaltarak örtüşme ve faz bozulmasını önlemede yardımcı olabilir. Bir işaret Nyquist oranının N katında

örnekleniyorsa, N faktörü kadar aşırı örneklendiği anlamına gelmektedir. Kanal sayısı ikinin katı olmalıdır. Kanal başına katsayı sayısı serbestçe seçilebilmektedir. Kök yükseltilmiş hata fonksiyonuna (root-raised-error-function) dayanan çok fazlı süzgeç bankasındaki prototip süzgeç doğrusal faza sahiptir [107, 108]. Giriş ve çıkış işaretleri arasındaki ortalama yeniden yapılandırma hatası desibel cinsinden hesaplanmaktadır.

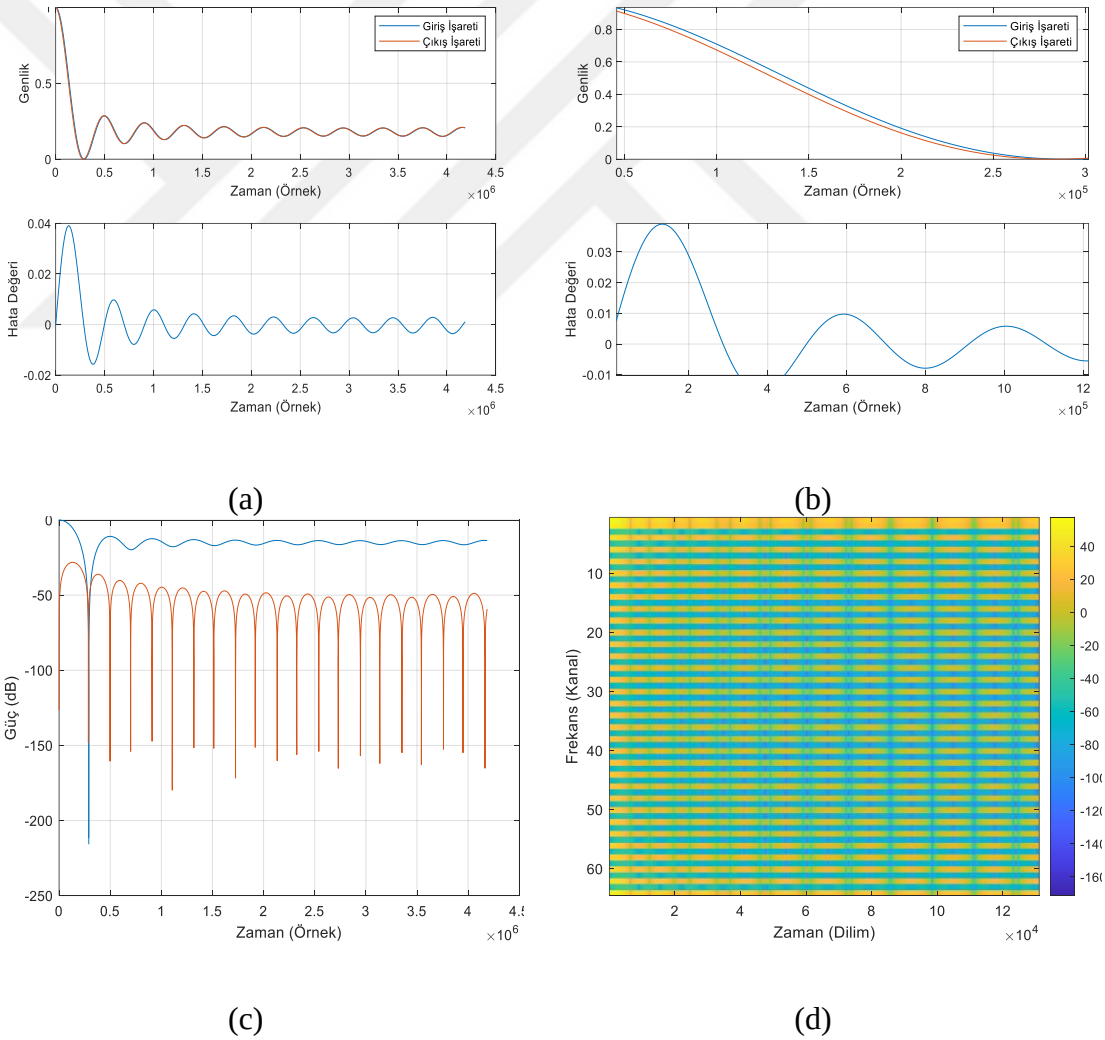
Bu çalışmada kullanılan ABC, qABC, mABC, PSO ve GA algoritmalarının problem çözmedeki üstün yönleri birleştirilerek mükemmele yakın bir yeniden yapılandırma ile çok fazlı bir süzgeç bankası oluşturulmuştur. Tez çalışması kapsamında kullanılan sezgisel algoritmaların sayısal süzgeçlerin tasarımında nasıl kullanıldığı bir önceki bölümde detaylı bir şekilde ele alınarak farklı derecelerden ve farklı geçirme bantlarına ait süzgeçler için ayrıntılı olarak incelenmiştir. Algoritmaların süzgeç tasarımı üzerindeki etkisi karşılaştırmalı olarak analiz edildikten sonra bu bölümde süzgeç bankası tasarımı için önerilen yöntem tanıtılacaktır. ABC, qABC, mABC, PSO ve GA algoritmalarının PFB tasarımında kullanımına dayalı olan yöntemle elde edilen tasarımlar verilerek çok fazlı süzgeç bankasının genel performansı gösterilecektir. Üçüncü bölümde Şekil 3.1'de PFB tasarımı için önerilen model blok yapı olarak verilmiştir. Üçüncü bölümde tasarlanan en iyi süzgeç derecesi yani süzgeç derecesinin 32 olması durumu temel alınarak gerçekleştirilen tasarımlarda derecenin yükselmesi durumundaki değişimi göstermek adına da N=256 için de elde edilen grafikler de verilmektedir. Tüm tasarımlarda iki katlı süzgeç bankası tasarımları gerçekleştirilmiştir.

Çok fazlı süzgeç bankasının performansını değerlendirmek için süzgeç bankasının girişine sinüzoidal bir işaret uygulanmıştır. Süzgeç bankası çıkışında elde edilen çıkış işaretinin giriş işaretine ne derece uyduğu gösterilmiştir. Çalışmalarda giriş ve çıkış işaretleri arasındaki farkın yani hata değerlerinin grafiği elde edilmiş, giriş ve çıkış işaretleri arasındaki güç farkının farklı bir ifadeyle yeniden yapılandırma hatasının desibel cinsinden grafiği çizdirilmiştir. Alt bant işaretlerinin spektrogramı, desibel cinsinden renkli bir harita olarak görselleştirilmiştir. Önerilen PFB tasarımlarında ABC algoritması ile 8. 16. 32. 64. ve 128. derecelerden tasarlanan sayısal süzgeç katsayıları Ek 1'de verilmiştir.

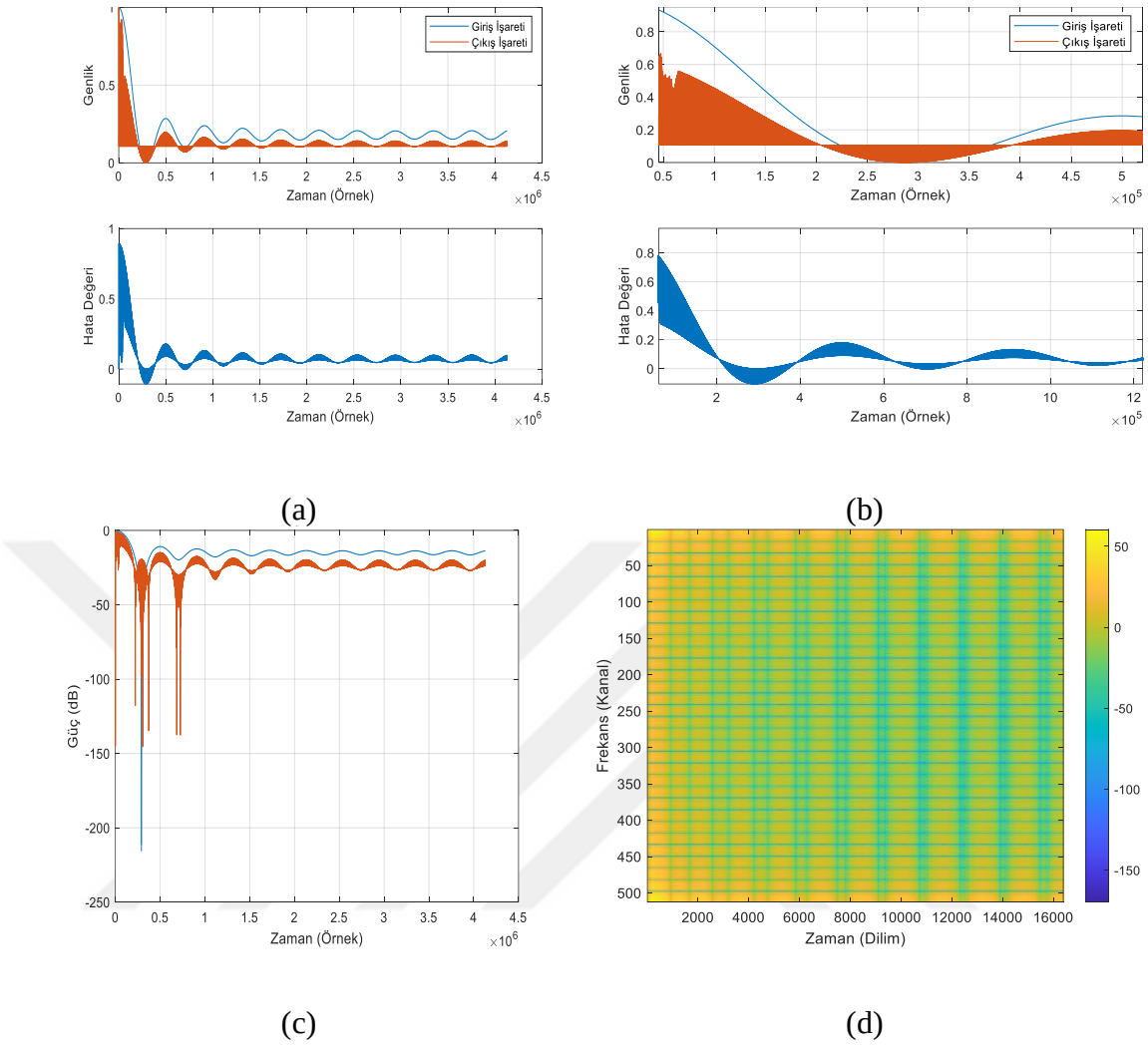
4.1. ABC Algoritması ile Çok Fazlı Süzgeç Bankası Tasarımı

Bu alt bölümde ABC ile N'nin 32 ve 256 olması durumunda iki katlı PFB tasarımları gerçekleştirilmiştir. Süzgeç derecesi arttıkça gerçekleştirilen tasarımın uygulanabilirliği

performans kriterleri açısından fayda sağlamamaktadır. Bu durumun anlaşılabilmesi amacıyla çalışmaya 256. dereceden PFB çıktıları eklenmiştir. Şekil 4.1'de *N*'nin 32 olması durumunda önerilen yöntemle tasarlanmış çok fazlı süzgeç bankasının (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği verilmektedir. (a) ve (b) şıklarında giriş ve çıkış işareti zamana göre genlik değeri açısından birbiriyle büyük oranda örtüştüğü gözlemlenmektedir. (c) şıkkında zamanla giriş işaretinin dalgalanması çıkış işaretine göre daha az olmuştur. (d) şıkkında da renk dağılımında mavi/yeşil dağılımı yoğunluk gösterdiğinden işaretin düşük genlikli olduğu görülmektedir. Belirli zaman aralıklarında farklı frekansta enerji yoğunlukları gözlemlenmiştir.



Şekil 4.1. N=32 için ABC algoritması ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği



Şekil 4.2. N=256 için ABC algoritması ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği

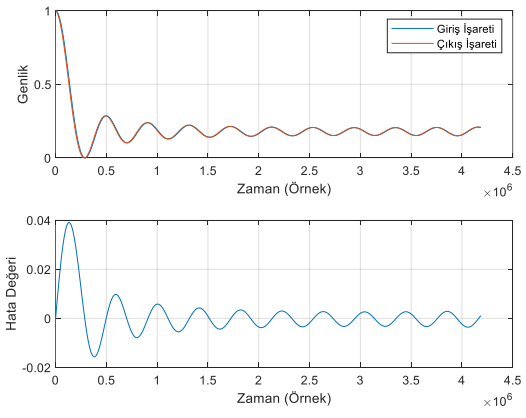
Süzgeçlerde derece artışının süzgeç bankası tasarımı üzerindeki etkisini göstermek için diğer bir örnek olarak *N*, 256 olarak seçilmiştir. Şekil 4.2'de 256. dereceden süzgeç için ABC algoritması ile tasarlanmış PFB'nin (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği çizdirilmiştir. Şekil 4.2 (a) ve (b)'de çıkış işareti giriş işaretine göre zamanla genlik değeri açısından çok fazla dalgalanmaya uğradığı ve örtüşmenin neredeyse tamamen bozulduğu gözlemlenmektedir. Şekil 4.2 (c)'de zamanla giriş işaretinde daha az çıkış işaretinde çok daha fazla dalgalanma meydana gelmiştir. Yeniden yapılandırma hatası, giriş işareti ile çıkış işaretlerinin arasındaki farktan kaynaklandığından, süzgeç bankasının işaretteki bilgiyi kaybetmesi veya bozması nedeniyle bu durum oluşmuştur. Böylelikle N'nin 256 olması durumunda PFB tasarımının uygulanabilir olmadığı anlaşılmıştır. (d) şıkkındaki

spektogram grafiğinden renk dağılımında mavi/yeşil dağılımı işaretin düşük genlikli sarı renk yoğunluğu olan bölgelerde de işaretin yüksek genlikli olduğu görülmektedir. Süzgeç derecesi arttığında çıkış işareti giriş işaretini işleyememektedir. Yani bozulma artmaktadır.

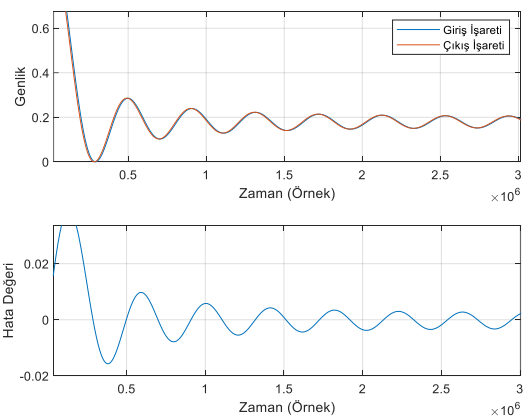
4.2. qABC Algoritması ile Çok Fazlı Süzgeç Bankası Tasarımı

PFB tasarımında qABC'nin kullanımı, süzgeçlerin frekans ve faz yanıtlarını iyileştirmeye yönelik katsayıları belirlemede önemli avantajlar sağlamaktadır.

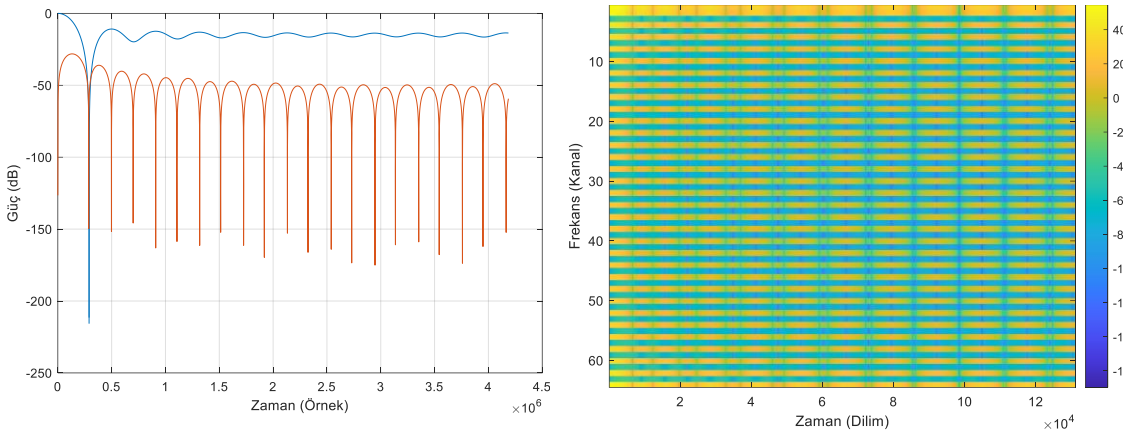
Bu alt bölümde qABC ile tasarlanan çok fazlı süzgeç bankaları sayısal süzgecin derecesinin 32 ve 256 olması durumunda incelenmiştir. Süzgeç derecesi arttıkça tasarımın uygulanabilirliği performans kriterleri açısından başarılı bulunmamıştır. Şekil 4.3 ve 4.4'te sırasıyla 32. ve 256. derece için qABC algoritması ile tasarlanmış PFB'nin (a) şıkkında giriş ve çıkış işaretleri (b) şıkkında giriş ve çıkış işaretlerinin detaylı gösterimi (c) şıkkında yeniden yapılandırma hatası ve (d) şıkkında spektrogram grafikleri verilmektedir. Şekil 4.3 (a) ve (b)'de giriş ve çıkış işareti zamana göre genlik değeri açısından birbiriyle çok iyi bir şekilde örtüştüğü gözlemlenmektedir. Şekilde (c) şıkkından zamanla giriş işaretindeki dalgalanmanın çıkış işaretine göre daha az gerçekleştiği görülmektedir. Şekil 4.3 (d)'de de mavi/yeşil dağılımı yoğunluk gösterdiğinden işaretin düşük genlikli olduğu görülmektedir. Ayrıca belirli zaman aralıklarında farklı frekansta enerji yoğunlukları da gözlemlenmektedir.



(a)



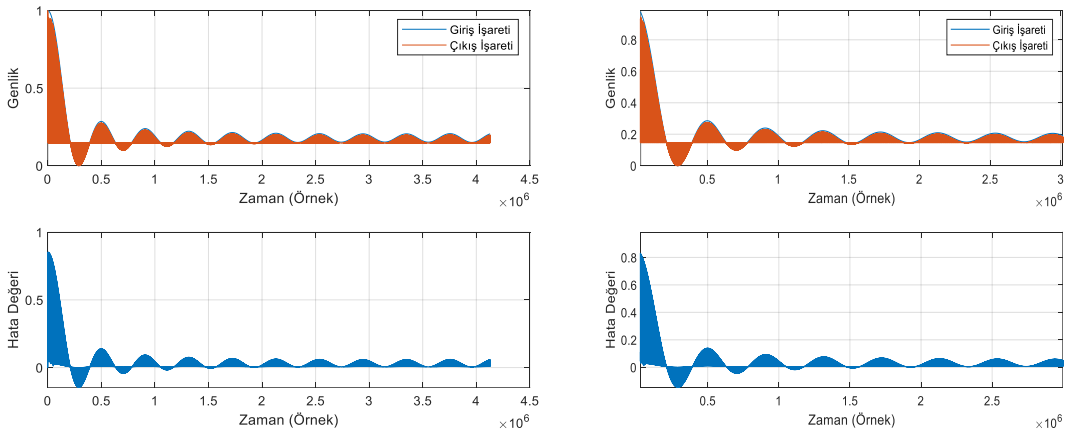
(b)



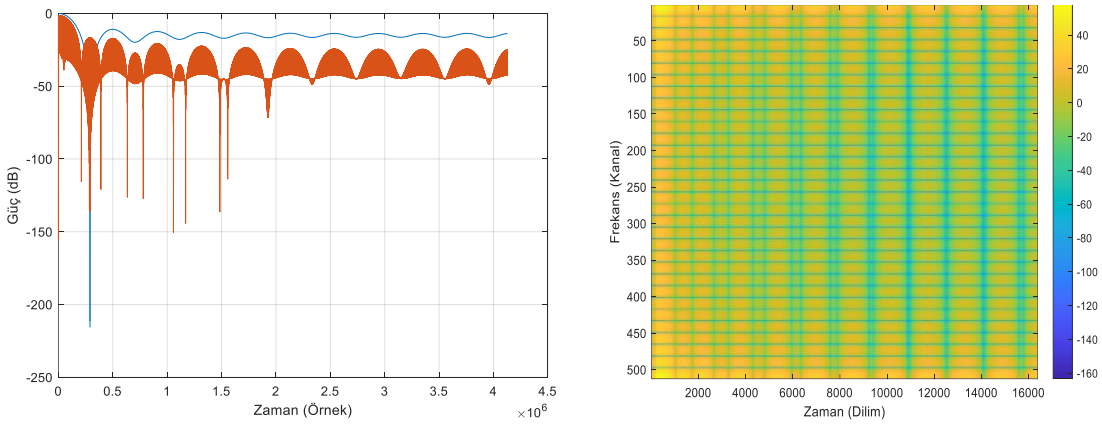
(c) (d)

Şekil 4.3. N=32 için qABC algoritması ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği

Şekil 4.4'ün (a) ve (b) şıkları değerlendirildiğinde ABC algoritması ile belirlenen duruma yakın bir sonuç elde edildiği görülmüştür. Yani çıkış işaretinin giriş işaretine göre zamanla genlik değeri açısından daha fazla dalgalanmaya uğradığı ve örtüşmenin neredeyse tamamen bozulduğu gözlemlenmiştir. Şekil 4.4 (c)'de zamanla giriş işaretinde daha az çıkış işaretinde çok daha fazla dalgalanma meydana gelmiştir. Süzgeç bankasının işaretteki bilgiyi kaybetmesi veya bozması nedeniyle bu durum oluşmaktadır. (d) şıkkından da sarı renk yoğunluğu olan bölgelerde işaretin yüksek genlikli olduğu anlaşılmaktadır.



(a) (b)



(c) (d)

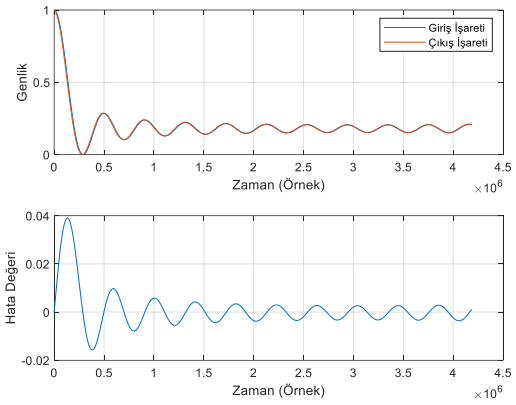
Şekil 4.4. N=256 için qABC algoritması ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği

4.3. mABC Algoritması ile Çok Fazlı Süzgeç Bankası Tasarımı

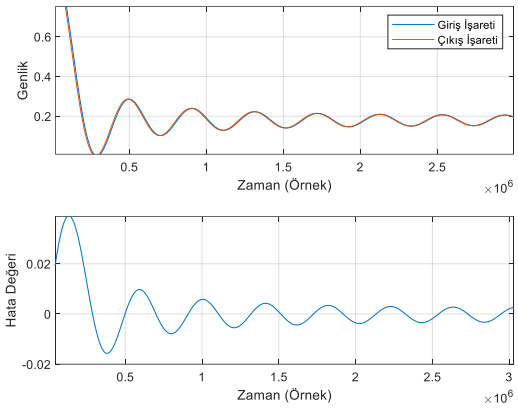
Bu alt bölümde mABC ile 32. ve 256. dereceden sayısal süzgeçlerle tasarlanan PFB tasarımlarının sonuçları incelenmiştir. Süzgeç derecesi arttıkça gerçekleştirilen tasarımlar uygulanabilirlikten uzaklaşmaktadır. Önerilen yöntemle elde edilen sonuçlar Şekil 4.5-6'da sırasıyla 32. ve 256. dereceler için verilmektedir.

Şekil 4.5'te (a), (b) ve (c)'de verilen giriş ve çıkış işaretlerinin değerlendirilmesi göz önüne alındığında işaretlerin örtüşmesi dolayısıyla ABC ve qABC ile tasarımlardakine benzer şekilde 32. dereceden tasarlanan bir süzgeçle PFB tasarımının uygun olduğu anlaşılmaktadır. Şekilde (d) şıkkında mavi/yeşil dağılımından bu bölgelerde düşük genlikli işaretlerin olduğu anlaşılmıştır.

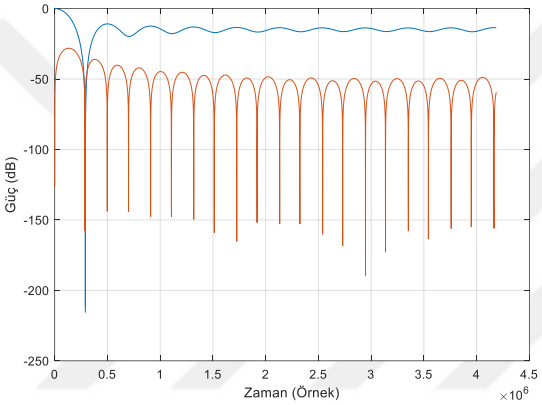
32. dereceye göre daha yüksek bir derece olan 256. dereceden süzgeçlerle gerçekleştirilen tasarımlara ait grafiklerin verildiği Şekil 4.6 incelendiğinde şimdiye kadar incelenen diğer sonuçlara benzer bir durumla karşılaşılmıştır. Şekilde örtüşmenin tamamen bozulduğu anlaşıldığından elde edilen çözümün kullanılmamasının uygun olacaktır. Şekil 4.6 (c)'de zamanla giriş işaretinde daha az çıkış işaretinde daha fazla bozulma meydana geldiği belirlenmiştir. (d) şıkkında da sarı renk yoğunluğu olan bölgelerde yüksek genlikli, mavi/yeşil dağılımlı bölgelerde de düşük genlikli işaretlerin bulunduğu görülmektedir.



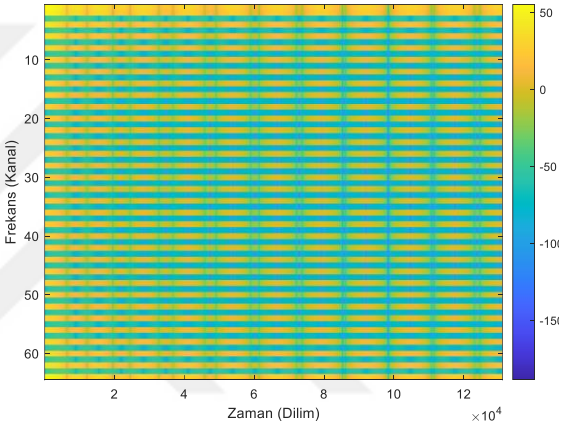
(a)



(b)

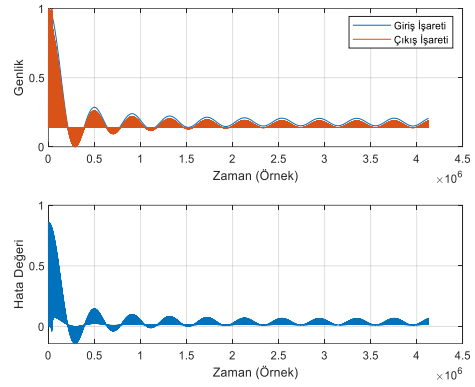


(c)

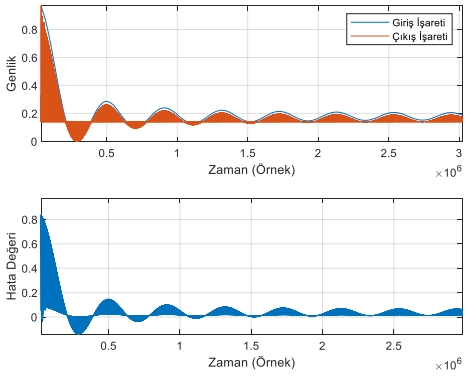


(d)

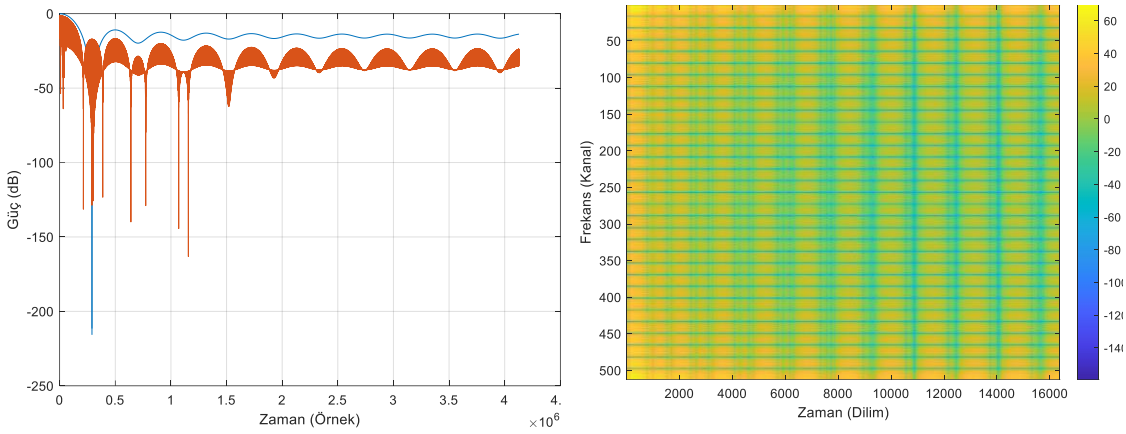
Şekil 4.5. N=32 için mABC algoritması ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği



(a)



(b)

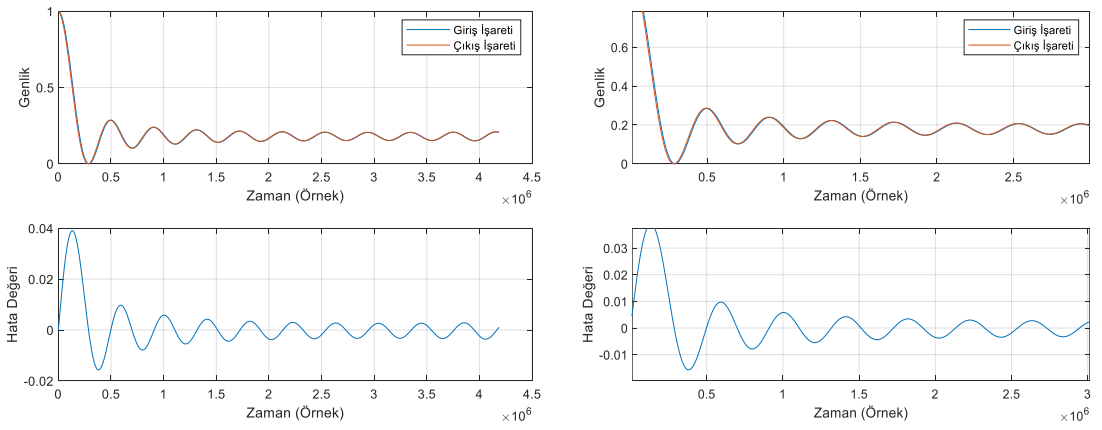


(c) (d)

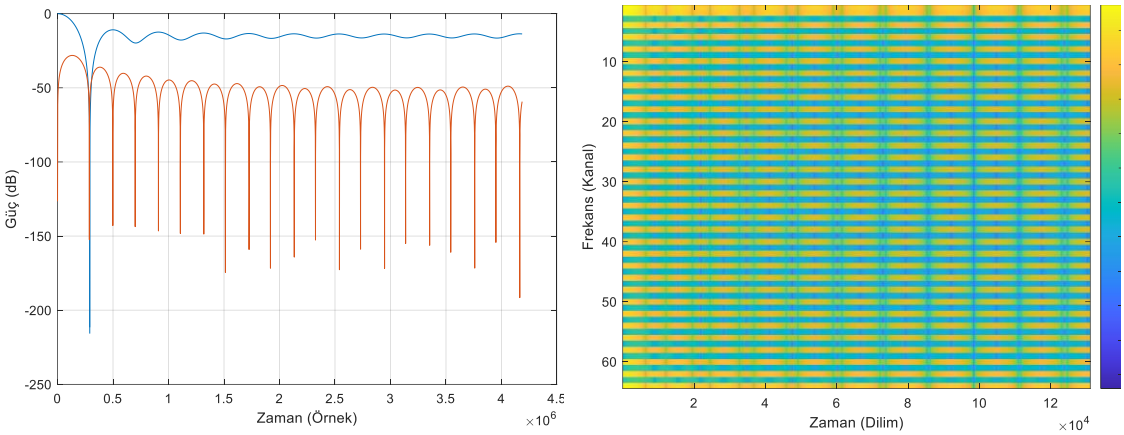
Şekil 4.6. N=256 için mABC algoritması ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği

4.4. GA ile Çok Fazlı Süzgeç Bankası Tasarımı

Bu bölümde GA ile 32. ve 256. dereceden PFB tasarımları gerçekleştirilmiştir. Önerilen yöntemle ilk olarak 32 daha sonra 256. dereceden süzgeçle gerçekleştirilen uygulamalarda diğer algoritmalara benzer şekilde 32. dereceden süzgeçle gerçekleştirilen tasarımlar daha iyi sonuç üretmiştir. N=32 için elde edilen sonuçlar Şekil 4.7'de, N=256 için elde edilen sonuçlar Şekil 4.8'de verilmektedir.



(a) (b)

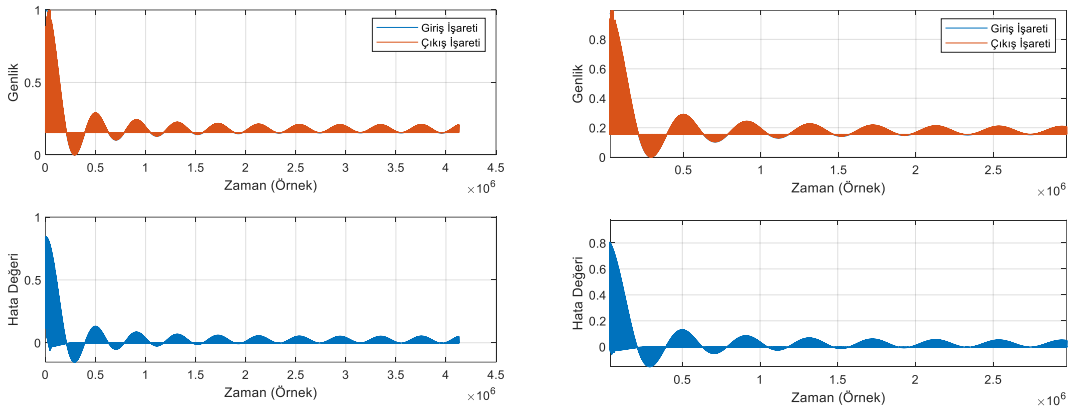


(c) (d)

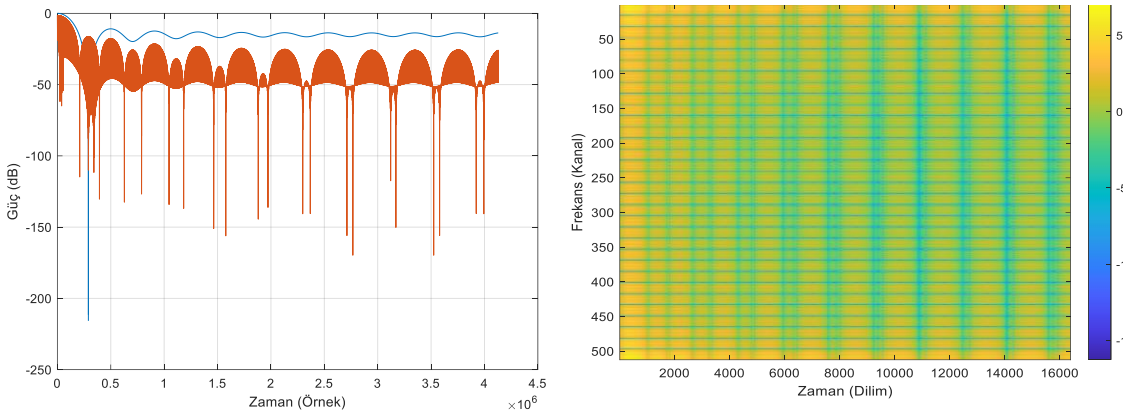
Şekil 4.7. N=32 için GA ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği

Şekilde (a) ve (b) şıklarından giriş ve çıkış işaretinin tamamen örtüştüğü gözlenmektedir. (c) şıkkından zamanla giriş işareti dalgalanmasının çıkış işaretine göre daha az gerçekleştiği görülürken (d) şıkkından işaretin düşük genlikli olduğu anlaşılmaktadır.

Şekil 4.8'te çıkış işaretinin giriş işaretine göre zamanla genlik değerinin bozulduğu anlaşılmaktadır yani süzgeç derecesinin artması süzgeç bankası tasarımına olumsuz olarak yansımaktadır. (c) şıkkında çizdirilen yeniden yapılandırma hatasından da 32. dereceye göre daha fazla bozulma meydana geldiği ifade edilebilir. (d) şıkkında da sarı renk yoğunluğu olan bölgelerde işaretin yüksek genlikli olduğu anlaşılmıştır.



(a) (b)

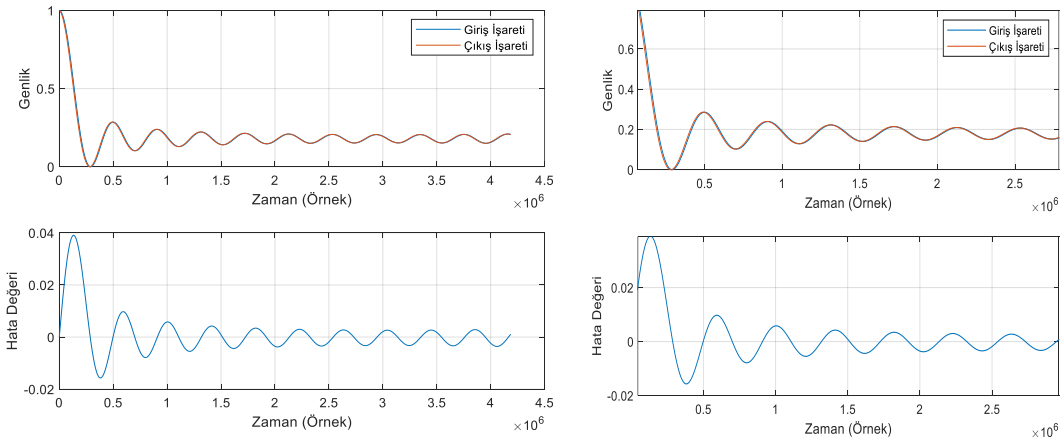


(c) (d)

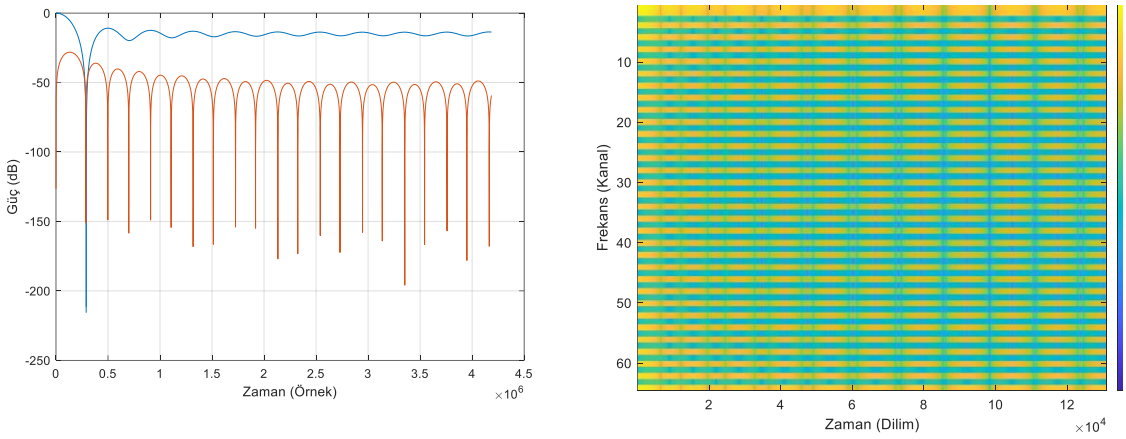
Şekil 4.8. N=256 için GA ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği

4.5. PSO ile Çok Fazlı Süzgeç Bankası Tasarımı

Bu alt bölümde PSO ile 32. ve 256. dereceden sayısal süzgeçlerle tasarlanan PFB tasarımları verilmektedir. Diğer algoritmalarla gerçekleştirilen tasarımlara benzer şekilde süzgecin derecesi arttıkça tasarımın başarımı kabul edilebilirlikten uzaklaşmaktadır. Önerilen yöntemle tasarlanmış PFB'ye ait grafik değerlendirmeler Şekil 4.9 ve 4.10'da verilmektedir. Diğer çalışmalara uygun olarak ilk olarak 32 daha sonra da 256. dereceden süzgeçler kullanılarak gerçekleştirilen çalışmalarda süzgeç bankası tasarımında kullanılan diğer sezgisel algoritmalarla benzer değerlendirmeler elde edilmiştir.



(a) (b)

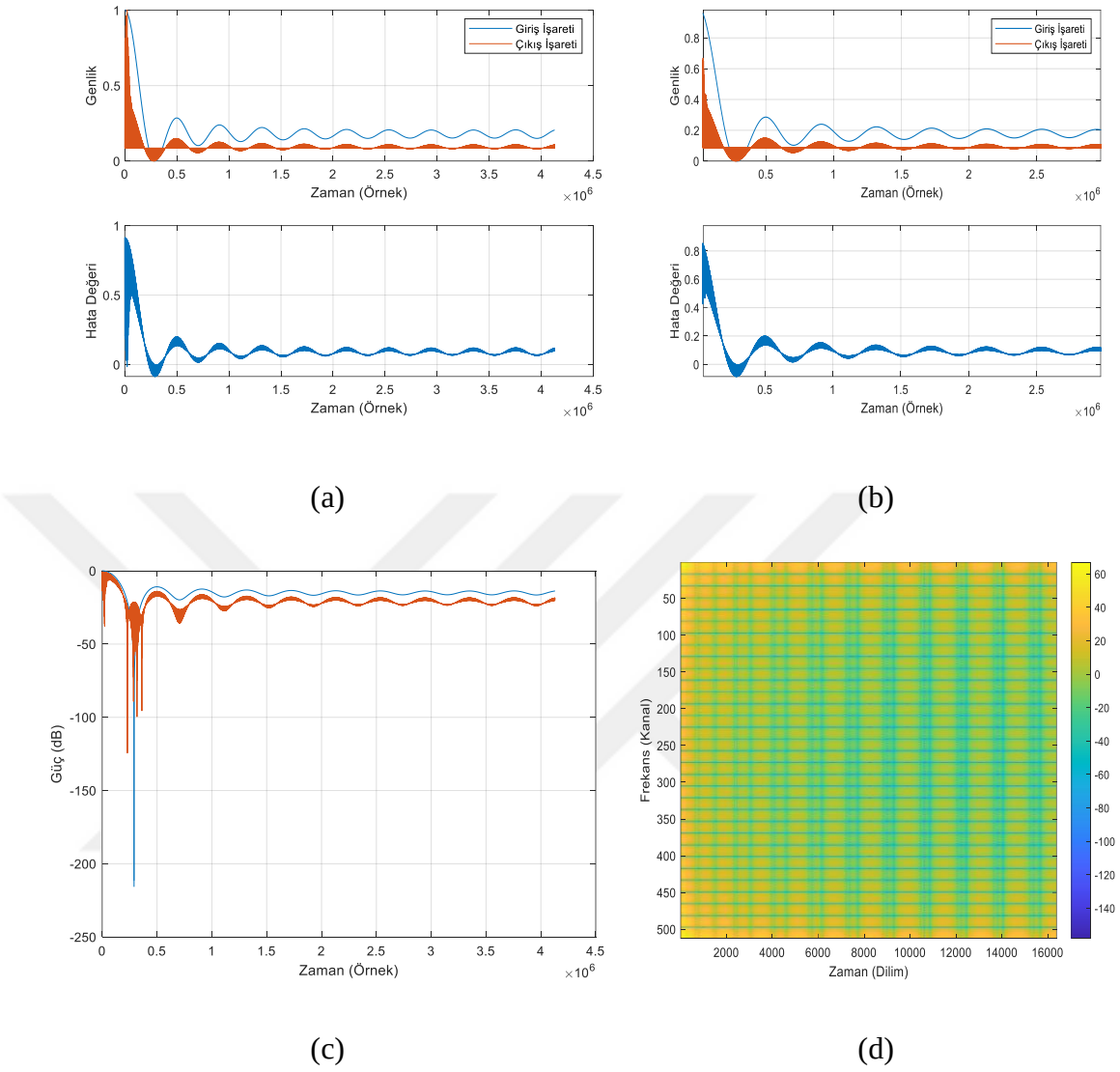


(c) (d)

Şekil 4.9. N=32 için PSO ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği

Şekil 4.9 (a) ve (b)'de üst üste çizdirilen giriş ve çıkış işaretleri değerlendirildiğinde genliğin büyük ölçüde örtüştüğü gözlemlenmektedir. (c) şıkkında zamana bağlı olarak giriş işaretinin dalgalanmasının çıkış işaretine göre kısmen daha az gerçekleştiği anlaşılırken d şıkkından işaretin daha ziyade mavi/yeşil dağılım göstermesi sebebiyle düşük genlikli bir işaret olduğu anlaşılmaktadır.

Şekil 4.10'da 256. dereceden sayısal süzgeç kullanılması durumunda süzgeç bankasının çıkışında elde edilen işaretler verilmektedir. 256. derece için diğer algoritmalarla tespit edilen durum PSO algoritması ile içinde geçerlidir. Şekilde (a) ve (b) şıklarında bozulmanın olduğu örtüşmenin tam anlamıyla gerçekleşmediği görülürken (c) şıkkından da zamanla çıkış işaretinde daha fazla, giriş işaretinde daha az bozulma meydana geldiği görülmüştür. (d) şıkkından da sarı renk yoğunluğu olan bölgelerde yüksek genlikli işaretin bulunduğu görülmüştür.



Şekil 4.10. N=256 için PSO ile tasarlanmış çok fazlı süzgeç bankası (a) giriş ve çıkış işaretleri (b) giriş ve çıkış işaretlerinin detaylı gösterimi (c) yeniden yapılandırma hatası (d) spektrogram grafiği

4.6. Önerilen Süzgeç Bankalarının Değerlendirilmesi

Süzgeç derecesi, süzgeç bankasının karmaşıklığını ve performansını belirleyen bir faktördür. Bu bölümde süzgeçlerin derecelerinin incelendiği üçüncü bölümdeki en iyi ortalama değere sahip olan 32. dereceden süzgeçlerin faz süzgeç bankası tasarımında kullanıldığı durum temel olarak göz önüne alınmıştır. Süzgeç derecesinin artmasının süzgeç bankasının performansını nasıl etkilediğini görmek açısından 256. dereceden süzgeç kullanıldığı durumda elde edilen sonuçlarla da karşılaştırma yapılmıştır. Tasarımda kullanılan ABC, qABC, mABC, GA ve PSO algoritmaların hepsinde 32.

dereceden bir sayısal süzgeç kullanılması durumunda süzgeç bankasının daha iyi performans gösterdiği belirlenmiştir. 32. derece için ABC ve PSO algoritmaları diğer tasarım algoritmalarına göre daha başarılı bulunmuştur. 256. dereceden yani 32. dereceye göre daha yüksek bir süzgeç kullanıldığında süzgeç bankasının performansında düşüş gözlemlenmiştir. Bu giriş ve çıkış işaretleri arasındaki örtüşmenin azalması ve çıkış dalga formlarında bozulmaların ortaya çıkması şeklinde kendini göstermektedir. Bu durum elbette süzgeç bankalarının tasarımında yüksek dereceden süzgeçlerin kullanımını sınırlandırmaktadır.

Böylece farklı süzgeç derecelerinde faz süzgeç bankasının performansını değerlendiren bu şekildeki bir çalışma ile belirli algoritmaların ki bu tez çalışmasında ABC ve PSO algoritmalarının belirli süzgeç derecelerinde daha başarılı olduğu, tespit edilmiştir.

5. BÖLÜM

TARTIŞMA, SONUÇ VE ÖNERİLER

5.1. Tartışma

Çok fazlı süzgeç bankası tasarımı son yıllarda araştırmacılar tarafından farklı yönleriyle ele alınan popüler konulardan birisi olmuştur. Çoklu hızlı DSP alt bant kodlama, alt bantlara ayrıştırma, işaret ayrıştırma ve işaret birleştirme gibi çeşitli alanlarda kullanılmaktadır. Çoklu hıza sahip DSP'nin en temel elemanlarından birisi olan çok fazlı süzgeç bankaları da yine aynı işlemler için kullanıldığı gibi DSP'de bulunan çok fazlı süzgeç bankalarında sıkça kullanılmaktadır. Aşağı veya yukarı örnekleme sonucu ortaya çıkan faz, genlik ve örtüşme bozulmalarının giderilmesi için çok fazlı süzgeç bankalarında doğrusal fazlı FIR süzgeçler kullanılmaktadır. Doğrusal ve FIR süzgeçlerin kullanımıyla faz ve örtüşme bozulmaları giderilmektedir. PFB'deki süzgeçler, prototip süzgeç adı verilen süzgecin dönüşümü ile elde edilmektedirler. Bu nedenle çok fazlı süzgeç bankasının tasarımı da prototip süzgecin en uygun şekilde tasarlanması problemi olarak ortaya çıkmaktadır. Prototip süzgeç parametrelerinin belirlenmesi, çok parametreli çok boyutlu bir optimizasyon problemidir. Bu tür problemlerin çözümünde küresel araştırma yeteneğine ve yüksek yakınsama hızına sahip sezgisel algoritmalar kullanılmaktadır.

Bu tez çalışmasında literatürde oldukça geniş uygulama alanına sahip ABC, mABC, qABC, PSO ve GA sezgisel algoritmalarının kullanımı tercih edilmiştir Gerçekleştirilen tez çalışmasında PFB tasarımı için süzgeç bankasında kullanılacak sayısal süzgeçlerin sezgisel algoritmalarla tasarlanması önerilmiştir.

Tez çalışmasının ilk bölümünde genel bilgiler ve literatür çalışması kapsamında DSP, sayısal süzgeçler, çoklu hızlı sayısal işaret işleme ve PFB yapısı hakkında temel bilgiler verilmiştir. IEEE Xplore, Web of Science, Science Direct, Google Scholar ve Springer Link veri tabanları kullanılarak gerçekleştirilen araştırma sonuçları

bu bölümde detaylandırılmıştır.

Tez çalışmasının ikinci bölümünde, süzgeç bankası tasarımında kullanılan ABC, mABC, qABC, PSO ve GA algoritmaları hakkında bilgi verilmiş ve bu algoritmaların sözde kodları, temel adımları ve çalışma prensipleri tek tek gösterilmiştir.

Tezin üçüncü bölümünde ABC tabanlı algoritmaların kontrol parametreleri detaylı bir şekilde incelenerek en iyi tasarımı veren parametre değerleri belirlenmiş ve bu parametre değerleri ile sayısal süzgeçlerin derecelerinin değişmesi durumu ayrıntılı olarak irdelenmiştir. ABC algoritması popülasyon sayısı ve limit parametresi, mABC ile MR ve qABC ile RR parametresi incelenmiştir. Ayrıca ABC tabanlı algoritmalara ek olarak sayısal süzgeç tasarımında literatürde çok kullanılan PSO ve GA algoritmaları da kullanılmış ve süzgeçlerin derecelerinin değişmesi durumunda nasıl bir performansa sahip olduğu ayrıntılı olarak irdelenmiştir. Farklı geçirme ve durdurma bantlarında sayısal süzgeçlerin tasarımında kullanılan algoritmaların performansları değerlendirildiğinde 32. dereceden bir FIR sayısal süzgeç kullanılması durumunda ABC algoritmasıyla tasarlanan sayısal süzgecin daha iyi sonuç verdiği belirlenmiştir.

Tezin dördüncü bölümünde önerilen yöntemle elde edilen sonuçlar verilmektedir. ABC, qABC, mABC, GA ve PSO ile tasarlanan sayısal süzgeçlerle tasarlanan çok fazlı süzgeç bankasının girişine sinüzoidal bir işaret uygulanması durumunda süzgeç bankasının çıkışında elde edilen giriş/çıkış işaretleri ve yeniden elde etme hatası çizdirilmiştir. Dördüncü bölümde üçüncü bölümde tespit edilen en optimal süzgeç derecesi temel alınmıştır. N=32 için gerçekleştirilen çalışmalara ek olarak süzgeç derecesinin artmasının performans üzerindeki etkisini göstermek adına 256. dereceden sayısal süzgeç kullanılarak da iki katlı süzgeç bankası tasarımları incelenmiştir. Tüm tasarımlarda 32. dereceden bir sayısal süzgeç kullanılması durumunda daha başarılı sonuçlar elde edilirken süzgeç derecesi arttıkça gerçekleştirilen tasarımlarda çıkış işareti giriş işaretini yakalayamamıştır. ABC algoritmasının diğer algoritmalara göre ortalama değer, standart sapma ve en iyi hata değerleri açısından daha iyi sonuçlar vermesi nedeniyle çok fazlı süzgeç bankalarının tasarımında kullanılabileceği gösterilmiştir.

5.2. Sonuç ve Öneriler

Bu tez çalışmasında prototip süzgecin genlik cevabının ideal süzgecin genlik cevabına yakınsaması prensibine dayalı bir tasarım gerçekleştirilmiştir. Önerilen yönteme göre ABC, mABC, qABC, PSO ve GA algoritmaları ile tasarlanan sayısal süzgeçlerle

tasarlanan iki katlı PFB tasarımlarının başarımları değerlendirilmiştir. Önerilen PFB'nin gerçek zamanlı uygulamalarda başarıyla kullanılabileceği değerlendirilmiştir. Ortalama değer, standart sapma ve en iyi hata değerinin kullanılarak gerçekleştirilen tasarımlarda süzgeç derecesinin etkisi detaylı bir şekilde incelenmiştir.

ABC ve PSO algoritmaları farklı süzgeç derecelerinde GA'ya göre daha başarılı sonuçlar üretmiştir Standart sapma açısından ABC algoritması, ortalama ve en iyi hata değeri açısından da PSO algoritması GA'dan daha iyi sonuçlar vermiştir. PFB tasarımının seçilen süzgeç parametrelerinin değişimiyle ve ABC, mABC ve qABC algoritmaları kullanılarak iyileştirilmesi yapılmıştır. Gerçekleştirilen çalışmalar tasarımda, ABC algoritmasının geçiş bandı frekansı, durdurma bandı frekansı ve süzgeç derecesi açısından en iyi başarıma sahip olduğunu göstermektedir. Doğal arı kolonilerinin davranışını taklit ederek potansiyel çözümler açısından başarılı bir arama stratejisi kullanana ABC algoritması mABC ve qABC algoritmalarına göre daha üstün bir performans sergilemiştir.

Değerlendirme için hata değeri, ortalama, standart sapma, en iyi hata değeri ve çözüme ulaşma süresini temsilen zaman değeri kullanılmıştır. Süzgeç derecesi 8, 16, 32 ve 64 değerlerine ulaşıncaya kadar tasarımı yapılan süzgeçlerin değerlendirme kriterlerine göre başarısını artırdığı ya da mevcut durumunu koruduğu görülmüştür. Özellikle 128, 256, 512 ve 1024 süzgeç derecelerine ulaşıldığında tasarlanan süzgeçlerin başarılarının çok fazla düşüş gösterdiği gözlemlenmiştir. ABC ve qABC algoritmaları da farklı süzgeç derecelerinde mABC'ye göre daha başarılı sonuçlar üretmiştir. Ortalama ve çözüm için geçirilen zaman değeri açısından ABC algoritması, standart sapma ve en iyi hata değeri açısından da qABC algoritması mABC'den daha iyi sonuçlar vermiştir. Ancak gelecekte yapılacak araştırmalarda, bu algoritmaların farklı konu ve parametreler üzerindeki etkinliğinin değerlendirilmesi için daha kapsamlı çalışmalara ihtiyaç duyulmaktadır.

Örneğin, ABC diğer optimizasyon teknikleriyle birleştirilerek hibrit bir yaklaşım oluşturulabilir. ABC ve PSO birleştirilmesi, her iki yöntemin üstünlüklerini birleştirerek daha iyi tasarım sonuçları elde etme konusunda çalışmalar yapılabilir. Gelecekteki çalışmalarda elde edilen sonuçlar ve gerçekleştirilen tasarımlar, gerçek zamanlı işaret işleme uygulamalarında kullanılabilecektir.

KAYNAKLAR

1. Goyal, D., C. Mongia, and S. Sehgal, *Applications of digital signal processing in monitoring machining processes and rotary components: a review.* **IEEE Sensors Journal, 2021. 21**(7): p. 8780-8804.
2. Diniz, P.S., E.A. Da Silva, and S.L. Netto, *Digital signal processing: system analysis and design*. 2010: Cambridge University Press.
3. Tan, L. and J. Jiang, *Digital signal processing: fundamentals and applications*. 2018: Academic Press.
4. Liu, C.-M. and W.-W. Chang, *Audio coding standards.* Preuzeto, 1999. **8**: p. 2016.
5. Karaboga, D., *An idea based on honey bee swarm for numerical optimization*. 2005, Technical report-tr06, Erciyes university, engineering faculty, computer engineering
6. Fliege, N.J., *Multirate digital signal processing: multirate systems, filter banks, wavelets*. 1994: John Wiley & Sons, Inc.
7. Proakis, J.G. and D.G. Manolakis, *Power spectrum estimation.* Digital signal processing; principles, algorithms, and applications, 3rd edn. Prentice Hall, New Jersey, 1996.
8. Miao, G.J. and M.A. Clements, *Digital signal processing and statistical classification*. 2002: Artech House.
9. Shenoy, R.G., D. Burnside, and T.W. Parks, *Linear periodic systems and multirate filter design.* **IEEE Transactions on Signal Processing, 1994. 42**(9): p. 2242-2256.
10. Uesaka, K. and M. Kawamata. *Synthesis of low-sensitivity second-order digital filter using genetic programming with automatically defined functions*. 2000. IEEE *International Symposium on Circuits and Systems (ISCAS)*
11. Sheikh, I.A. and N.K. Pannu, *Review of Polyphase Filtering Technique in Signal Processing.* I**nternational Journal of Computer Science and Mobile Computing, 2018. 7**(12): p. 213.
12. Mitra, S.K., *Digital signal processing: a computer-based approach*. 2001: McGraw-Hill Higher Education.
13. Jayant, N.S. and P. Noll, *Digital coding of waveforms: principles and applications to speech and video*. Vol. 2. 1984: Prentice-Hall Englewood Cliffs, NJ.
14. Vetterli, M. *Perfect transmultiplexers*. in *ICASSP'86. IEEE International Conference on Acoustics, Speech, and Signal Processing*. 1986.
15. Bellanger, M., G. Bonnerot, and M. Coudreuse, *Digital filtering by polyphase network: Application to sample-rate alteration and filter banks.* **IEEE Transactions on Acoustics, Speech, and Signal Processing, 1976. 24**(2): p. 109-114.
16. Renfors, M., et al., *Orthogonal waveforms and filter banks for future communication systems*. 2017: Academic Press.
17. Milic, L., *Multirate Filtering for Digital Signal Processing: MATLAB Applications: MATLAB Applications*. 2009: IGI Global.
18. Chennamangalam, J., *The polyphase filter bank technique*. 2011, CASPER Memo 41, The Berkeley Wireless Res. Cent., The UC Berkeley Radio
19. Proakis, J.G., *Digital signal processing: principles, algorithms, and applications, 4/E*. 2007: Pearson Education India.
20. Lu, W. and A. Antoniou, *Two-dimensional digital filters: Marcel Deckker.* New York, 1992.

21. Karaboğa, D. and Y.Z.O. Algoritmaları, *Atlas Yayın Dağıtım.* İstanbul, 2004.
22. Karaboga, D. and B. Akay, *A comparative study of artificial bee colony algorithm.* Applied mathematics and computation, 2009. **214**(1): p. 108-132.
23. Kockanat, S. and N. Karaboga, *The design approaches of two-dimensional digital filters based on metaheuristic optimization algorithms: a review of the literature.* **Artificial Intelligence Review, 2015. 44**: p. 265-287.
24. Yang, X.-S., *Metaheuristic optimization.* Scholarpedia, 2011. **6**(8): p. 11472.
25. Holland, J., *Adaptation in natural and artificial systems, univ. of mich. press.* Ann Arbor, 1975. **7**: p. 390-401.
26. Goldberg, D., *Genetic Algorithms in Search, Optimization and Machine Learning. Reading, MA: Addison-Wesley Professional.* 1989.
27. Kirkpatrick, S., C.D. Gelatt Jr, and M.P. Vecchi, *Optimization by simulated annealing.* science, 1983. **220**(4598): p. 671-680.
28. Glover, F., *Future paths for integer programming and links to artificial intelligence.* Computers & operations research, 1986. **13**(5): p. 533-549.
29. Glover, F., *Tabu search—part II.* ORSA **Journal on computing, 1990. 2**(1): p. 4-32.
30. Glover, F., *(1989a). Tabu Search-Part I,* ***ORSA*** **Journal on Computing, 1** (3), 190-206. 1989.
31. Glover, F., E. Taillard, and E. Taillard, *A user's guide to tabu search.* Annals of operations research, 1993. **41**(1): p. 1-28.
32. Farmer, J.D., N.H. Packard, and A.S. Perelson, *The immune system, adaptation, and machine learning.* **Physica D: Nonlinear Phenomena, 1986. 22**(1-3): p. 187-204.
33. Hunt, J.E. and D.E. Cooke, *Learning using an artificial immune system.* **Journal of network and computer applications, 1996. 19**(2): p. 189-212.
34. Kennedy, J. and R. Eberhart. *Particle swarm optimization*. in *Proceedings of ICNN'95-international conference on neural networks*. 1995. IEEE.
35. Storn, R. and K. Price, *Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces.* **Journal of global optimization, 1997. 11**: p. 341-359.
36. Corne, D., et al., *New ideas in optimization*. 1999: McGraw-Hill Ltd., UK.
37. Geem, Z.W., J.H. Kim, and G.V. Loganathan, *A new heuristic optimization algorithm: harmony search.* simulation, 2001. **76**(2): p. 60-68.
38. De Castro, L.N. and F.J. Von Zuben, *Learning and optimization using the clonal selection principle.* **IEEE transactions on evolutionary computation, 2002. 6**(3): p. 239-251.
39. Pham, D., et al., *The bees algorithm.* Technical Note, Manufacturing Engineering Centre, Cardiff University, UK, 2005: p. 44-48.
40. Rashedi, E., H. Nezamabadi-Pour, and S. Saryazdi, *GSA: a gravitational search algorithm.* **Information sciences, 2009. 179**(13): p. 2232-2248.
41. Passino, K.M., *Biomimicry of bacterial foraging for distributed optimization and control.* **IEEE control systems magazine, 2002. 22**(3): p. 52-67.
42. Logoglu, A., et al. *Examination of success of the finite impulse response digital filter designed using the bat algorithm*. in *2018 26th Signal Processing and Communications Applications Conference (SIU)*. 2018. IEEE.
43. Karaboga, D. and B. Gorkemli, *A quick artificial bee colony (qABC) algorithm and its performance on optimization problems.* **Applied Soft Computing, 2014. 23**: p. 227-238.

44. Gorkemli, B. and D. Karaboga, *A quick semantic artificial bee colony programming (qsABCP) for symbolic regression.* **Information Sciences, 2019. 502**: p. 346-362.
45. Zhou, X. and Y. Wu, *The research and improvement of a high efficient channelized algorithm based on polyphase filters.* **Signal Processing, 2008. 24**(1).
46. Fowler, M.L. *Polyphase filters-A model for teaching the art of discovery in DSP*. 2011. IEEE.
47. Vaidyanathan, P. P. (2013, August). Analog filter banks for sampling: Discretization, polyphase form, and role in compressive sensing. In *2013 IEEE Digital Signal Processing and Signal Processing Education Meeting (DSP/SPE)* (pp. 24-29). IEEE.
48. Zheng, S., & Yang, X. (2014, August). Wideband spectrum sensing based on group testing utilizing polyphase filter banks. In *2014 19th International Conference on Digital Signal Processing* (pp. 592-596). IEEE.
49. Ling, B.W.-K., et al., *Optimal design of cosine modulated nonuniform linear phase FIR filter bank via both stretching and shifting frequency response of single prototype filter*. **IEEE transactions on signal processing, 2014. 62**(10): p. 2517-2530.
50. Harris, F., Venosa, E., Chen, X., & Dick, C. (2015, July). Interleaving different bandwidth narrowband channels in perfect reconstruction cascade polyphase filter banks for efficient flexible variable bandwidth filters in wideband digital transceivers. In *2015 IEEE International Conference on Digital Signal Processing (DSP)* (pp. 1111-1116). IEEE.
51. Deshmukh, R.M. and R. Keote. *Design of polyphase FIR filter using bypass feed direct multiplier*. in *2015 International Conference on Communications and Signal Processing (ICCSP)*. 2015. IEEE.
52. Eminaga, Y., Coskun, A., Moschos, S. A., & Kale, I. (2015, June). Low complexity all-pass based polyphase decimation filters for ECG monitoring. In *2015 11th Conference on Ph. D. Research in Microelectronics and Electronics (PRIME)* (pp. 322-325). IEEE.
53. Kongrui, Z., & Chao, Z. (2015, July). A polyphase digital down converter method for GHz high speed sampling signal. In *2015 12th IEEE International Conference on Electronic Measurement & Instruments (ICEMI)* (Vol. 2, pp. 907-910). IEEE.
54. Baderia, K., A. Kumar, and G.K. Singh, *Design of quadrature mirror filter bank using polyphase components based on optimal fractional derivative constraints.* AEU-**International Journal of Electronics and Communications, 2015. 69**(9): p. 1254-1264.
55. Liu, T., Han, J., & Li, Z. (2016, August). Broadband DDC based on polyphase filter and its FPGA implementation. In *2016 IEEE International Conference on Electronic Information and Communication Technology (ICEICT)* (pp. 170-173). IEEE.
56. Gawande, G. S., & Khanchandani, K. B. (2016, February). Design, implementation and analysis of power efficient polyphase multirate filters. In *2016 3rd International Conference on Signal Processing and Integrated Networks (SPIN)* (pp. 353-358). IEEE.
57. Tay, D.B. and A. Ortega, *Bipartite graph filter banks: Polyphase analysis and generalization.* **IEEE Transactions on Signal Processing, 2017. 65**(18): p. 4833–4846.

58. Du, Y., Ye, X., Li, Y., & Cai, Z. (2016, August). Digital down-conversion design based on polyphase filtering structure. In *2016 IEEE International Conference on Signal and Image Processing (ICSIP)* (pp. 622-625). IEEE.
59. Tanaka, Y., & Sakiyama, A. (2017, March). A "polyphase" structure of two-channel spectral graph wavelets and filter banks. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 4144-4148). IEEE.
60. Logoglu, A., S. Kockanat, and N. Karaboga, *A Novel Approach for Polyphase Filter Bank Design Using ABC Algorithm.* **Elektronika Ir Elektrotechnika, 2022.**
61. Yang, X.-S., *Nature-inspired metaheuristic algorithms*. 2010: Luniver press.
62. Karaboga, D. and B. Basturk. *Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems*. in *International fuzzy systems association world congress*. 2007. Springer.
63. Basturk, B. *An artificial bee colony (ABC) algorithm for numeric function optimization*. in *IEEE Swarm Intelligence Symposium, Indianapolis, IN, USA, 2006*. 2006.
64. Karaboga, N., *A new design method based on artificial bee colony algorithm for digital IIR filters.* **Journal of the Franklin Institute, 2009. 346**(4): p. 328-348.
65. Karaboga, D. and B. Basturk. *An artificial bee colony (ABC) algorithm for numeric function optimization*. in *IEEE swarm intelligence symposium*. 2006. IEEE Press Indiana.
66. Karaboga, N. and B. Cetinkaya. *Performance comparison of genetic and differential evolution algorithms for digital FIR filter design*. in *International conference on advances in information systems*. 2004. Springer.
67. Ozdemir, G. and N. Karaboga, *A review on the cosine modulated filter bank studies using meta-heuristic optimization algorithms.* **Artificial Intelligence Review, 2019. 52**: p. 1629-1653.
68. Karaboga, D. and B. Akay, *A modified artificial bee colony (ABC) algorithm for constrained optimization problems.* **Applied soft computing, 2011. 11**(3): p. 3021-3031.
69. Karaboga, D. and B. Basturk, *On the performance of artificial bee colony (ABC) algorithm.* **Applied soft computing, 2008. 8**(1): p. 687-697.
70. Kockanat, S. and N. Karaboga, *A novel 2D-ABC adaptive filter algorithm: a comparative study.* **Digital Signal Processing, 2015. 40**: p. 140-153.
71. Kockanat, S., T. Koza, and N. Karaboga, *Cancellation of noise on mitral valve doppler signal using iir filters designed with artificial bee colony algorithm.* **Current Opinion in Biotechnology, 2011(22)**: p. S57.
72. Karaboga, N., S. Kockanat, and H. Dogan, *The parameter extraction of the thermally annealed Schottky barrier diode using the modified artificial bee colony.* **Applied Intelligence, 2013. 38**: p. 279-288.
73. Karaboga, N., Kockanat, S., & Dogan, H. (2011, June). Parameter determination of the schottky barrier diode using by artificial bee colony algorithm. In *2011 International Symposium on Innovations in Intelligent Systems and Applications* (pp. 6-10). IEEE.
74. Koza, T., Karaboğa, N., & Koçkanat, S. (2012, July). Aort valve Doppler signal noise elimination using IIR filter designed with ABC algorithm. In *2012 International Symposium on Innovations in Intelligent Systems and Applications* (pp. 1-5). IEEE.

75. Kamişlıoğlu, B., Koza, T., Koçkanat, S., & Karaboğa, N. (2013, June). Noise cancellation on mitral valve Doppler signal with IIR digital filter using harmony search algorithm. In *2013 IEEE INISTA* (pp. 1-5). IEEE.
76. Koçkanat, S., Karaboğa, N., & Koza, T., *ABC Algoritması İle Tasarlanan Fır Süzgeçlerle Mitral Kapak Doppler İşaretindeki Gürültü Giderimi* in *Elektrik Elektronik Mühendisliği Günleri*. 2011: Ankara. p. 45-48.
77. Karaboğa, N. and M.B. Cetinkaya, *A novel and efficient algorithm for adaptive filtering: artificial bee colony algorithm.* **Turkish Journal of Electrical Engineering and Computer Sciences, 2011. 19**(1): p. 175-190.
78. Akay, B., Nümerik Optimizasyon Problemlerinde Yapay Arı Kolonisi (artificial bee colony) Algoritmasının Performans Analizi. Erciyes Üniversitesi Fen Bilimleri Enstitüsü, Doktora Tezi, 2009:Kayseri
79. Koçkanat, S., Yapay Arı Koloni Algoritması Kullanılarak İki Boyutlu Sayısal Süzgeçlerin Tasarımının İncelenmesi ve Geliştirilmesi, Erciyes Üniversitesi Fen Bilimleri Enstitüsü Elektrik Elektronik Müh. ABD Doktora Tezi. 2015: Kayseri.
80. Koza, T., et al. *SSA analysis of noise eliminated transcranial doppler signals with iIR filters*. in *2015 23nd Signal Processing and Communications Applications Conference (SIU)*. 2015. IEEE.
81. Koçkanat, S., T. Koza, and N. Karaboğa. *IIR filter paramater estimation with ABC algorithm for mitral valve Doppler signal denoising*. in *2015 23nd Signal Processing and Communications Applications Conference (SIU)*. 2015. IEEE.
82. Kockanat, S., N. Karaboga, and T. Koza. *Image denoising with 2-D FIR filter by using artificial bee colony algorithm*. in *2012 International Symposium on Innovations in Intelligent Systems and Applications*. 2012. IEEE.
83. Karaboga, N., F. Latifoglu, and T. Koza, *Ssa analysis of transcranial doppler signal using iir filters designed with abc algorithm.* **Current Opinion in Biotechnology, 2011. 22**(1): p. 5.
84. Akay, B. and D. Karaboga, *A modified artificial bee colony algorithm for real-parameter optimization.* **Information sciences, 2012. 192**: p. 120-142.
85. Karaboga, D. and B. Gorkemli. *A quick artificial bee colony-qABC-algorithm for optimization problems*. in *2012 International symposium on innovations in intelligent systems and applications*. 2012. IEEE.
86. Kabalci, Y., S. Kockanat, and E. Kabalci, *A modified ABC algorithm approach for power system harmonic estimation problems.* **Electric Power Systems Research, 2018. 154**: p. 160-173.
87. Tan, R.K. and Ş. Bora, *Adaptive modified artificial bee colony algorithms (AMABC) for optimization ofcomplex systems.* **Turkish Journal of Electrical Engineering and Computer Sciences, 2020. 28**(5): p. 2602-2629.
88. Liang, J.-H. and C.-H. Lee, *A modification artificial bee colony algorithm for optimization problems.* **Mathematical Problems in Engineering, 2015.**
89. Gorkemli, B. and Z. Al-Dulaimi, *On the performance of quick artificial bee colony algorithm for dynamic deployment of wireless sensor networks.* **Turkish Journal of Electrical Engineering and Computer Sciences, 2019. 27**(6): p. 4038-4054.
90. Imran, M., R. Hashim, and N.E. Abd Khalid, *An overview of particle swarm optimization variants.* **Procedia Engineering, 2013. 53**: p. 491-496.
91. Holland, J.H., *Genetic algorithms and the optimal allocation of trials.* **SIAM Journal on Computing, 1973. 2**(2): p. 88-105.
92. Holland, J.H., *Genetic algorithms.* **Scientific American, 1992. 267**(1): p. 66-73.

93. Pham, D. and D. Karaboga, *Intelligent optimisation techniques: genetic algorithms, tabu search, simulated annealing and neural networks*. 2012: Springer Science & Business Media.
94. Erkoc, M.E. and N. Karaboga, *Evolutionary algorithms for sparse signal reconstruction.* **Signal, Image and Video Processing, 2019. 13**: p. 1293-1301.
95. Srinivas, M. and L.M. Patnaik, *Genetic algorithms: A survey.* computer, 1994. **27**(6): p. 17-26.
96. Loğoğlu, A., *Su Dağıtım Şebekesinin Optimizasyonunda Genetik Algoritmanın Kullanılması*, in *Elektrik Elektronik Mühendisliği*. Yüksek Lisans Tezi, 2006, Erciyes Üniversitesi: Kayseri. p. 77.
97. Zbigniew, M., *Genetic algorithms+ data structures= evolution programs.* **Comput Stat, 1996**: p. 372-373.
98. Wang, C.-H. and J.-Z. Lu, *A hybrid genetic algorithm that optimizes capacitated vehicle routing problems.* **Expert Systems with Applications, 2009. 36**(2): p. 2921-2936.
99. Şen, Z., *Genetik algoritmalar ve en iyileme yöntemleri*. 2004: Su Vakfı.
100. Nearchou, A.C., *Path planning of a mobile robot using genetic heuristics.* **Robotica, 1998. 16**(5): p. 575-588.
101. Yeo, M.F. and E.O. Agyei, *Optimising engineering problems using genetic algorithms.* **Engineering Computations, 1998. 15**(2): p. 268-280.
102. Oppenheim, A.V., et al., *Signals and systems*. Vol. 2. 1997: Prentice hall Upper Saddle River, NJ.
103. Oppenheim, A.V., R.W. Schafer, and J. Buck, *The discrete Fourier transform*, in *Discrete-Time Signal Processing*. 1999, Prentice Hall. p. 564-575.
104. Karaboğa, N. and B. Kamışlıoğlu, *Çeyrek Ayna Süzgeç Bankası Tasarımı İçin Yeni Bir Yöntem.* **Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, 2015. 30**(2).
105. Ifeachor, E.C. and B.W. Jervis, *Digital Signal Processing: A Practical Approach*. 2002: Prentice Hall.
106. Bellanger, M. and J. Daguet, *TDM-FDM transmultiplexer: Digital polyphase and FFT.* **IEEE Transactions on Communications, 1974. 22**(9): p. 1199-1205.
107. Lubberhuizen, W. *Near Perfect Reconstruction Polyphase Filterbank*. 2022 ; Web page: https://www.mathworks.com/matlabcentral/fileexchange/15813-near-perfect-reconstruction-polyphase-filterbank. (Erişim Tarihi: Aralık 2023)
108. Moazzen, I. *Filter Bank Design*. 2023; Web page: https://www.mathworks.com/matlabcentral/fileexchange/40128-filter-bank-design.

EKLER

EK 1. Farklı Süzgeç Derecelerinde ABC Algoritması ile Elde Edilen Süzgeç Katsayıları

Tablo Ek 1.1. Süzgeç derecesi 8 olduğunda elde edilen katsayılar

0.0178838877053427	0.417675932488215	0.0118564401940498	0.846819549059568
-0.0380649301769743	-0.305552668080392	0.00948624926784254	0.0881373380621081

Tablo Ek 1.2. Süzgeç derecesi 16 olduğunda elde edilen katsayılar

0.158521314446805	0.528748468546029	0.489877503983286	-0.0216564110748927
-0.178818852759353	0.0342402050354916	0.166828137672670	-0.361540153512452
0.423632560552408	-0.257556869330310	0.0253202302685653	0.0781870553893945
-0.0724767793650063	0.0317343122410023	0.0281803617753145	-0.0456170989915066

Tablo Ek 1.3. Süzgeç derecesi 32 olduğunda elde edilen katsayılar

-0.0444845220155277	-0.0226898677219153	0.0633106885826592	0.0899909411699788
-0.155178235709356	0.0465604237887962	-0.00128499852450455	0.133719257615825
-0.0566605133299175	0.0148291209378902	-6,71E+09	0.0132240654170025
-0.479556691848363	0.476835837471769	-0.102158187333724	-0.197126842429550
-0.0513264944778494	-0.138126137807327	0.311926196867722	0.397408340436380
0.376438802584474	0.0575235490305800	0.110216490120762	0.0594976242363078
-0.0659396341670633	0.0375922301404361	0.0202329654546419	0.0731046532115891
0.0922381088966494	0.0741314259541506	-0.0199017442060139	-0.0421109191992474

Tablo Ek 1.4. Süzgeç derecesi 64 olduğunda elde edilen katsayılar

0.0590966563078658	0.00865721049094175	-0.0155338646655796	0.103589807183596
-0.0613319429630139	-0.0525023445459635	0.143643293478894	0.0476710242752215
0.0661148888502405	0.0598158233865105	-0.0665376162541813	-0.0898526440501394
-0.0419544262219784	0.127449181034383	-0.00979780331859856	-0.122619805335922
0.0521455151205561	-0.138189017377438	-0.193450025946056	0.0631323626384276
-0.0518994534348069	-0.114349652359685	0.309856344605731	0.101143294680964
-0.132004101679965	0.136907229749626	-0.112394701361472	0.239452883139812
0.374660433171818	0.117605438043804	0.178328689595121	-0.154068017759959
0.0673889546451107	0.0737250012168407	0.0902257958743829	0.0814698693479454
-0.128514369904941	0.110166650100216	-0.102972391935215	0.0618707340885203
0.0525688260019078	-0.0909730615588612	0.106443183603380	-0.00577541986200678
0.191594045338870	0.0542094978648790	0.197089929295112	-0.133918571596748
0.0403319207054532	-0.351816132705929	0.000608786673390718	-0.0973737485943630

0.0790125416408791	-0.0865240774140919	0.0977367073821286	0.174723112462951
0.0424579329277972	-0.0493889503011097	-0.0314919509593670	-0.00160168693489260
-0.0941638779286217	-0.143548650244841	-0.0324147073199981	0.0973583943190761

Tablo Ek 1.5. Süzgeç derecesi 128 olduğunda elde edilen katsayılar

-0.128837845401157	0.0368765514086975	0.00646707567473570	0.0199192823049304
-0.0811745191538448	-0.00426253000846905	0.0663914874562997	-0.0439172225001276
0.0130002985850099	-0.0280815405983176	-0.00980230680430196	0.0848607429937403
-0.0107769000685585	0.00426905421478518	-0.141356519098847	0.0898963978593696
0.00273071794657437	0.163451706364653	-0.0897921916034997	0.00947163003769129
-0.117006675514546	0.127731599094227	0.0704782647303882	0.00210598261420364
-0.117612070090550	-0.00728991340736722	-0.175776807727357	0.137381022335580
0.0639975973827804	0.0164050156554699	-0.182035412545750	-0.0467791099730852
-0.00288449103874448	0.0517368171484564	0.0484630338072188	0.0600233200057652
-0.0622442137512778	-0.0426793304881060	0.172277146148800	0.0150273131997364
0.162985296058322	-0.0358276992255767	0.0669824159915241	-0.138867879871336
0.0334001018715657	-0.115175762102435	-0.0808747733044596	-0.00566685516021583
-0.0755200342690224	0.0322003684723253	-0.188020634691245	0.147214494331903
-0.0167342288257320	0.0658053930274538	0.165254693201561	-0.0272431791605028
0.0666541458406602	0.0203444350327869	0.0572204815943851	0.0832626476274547
0.0803386435841029	0.0681334122993323	-0.0237320947901908	0.00637286779239337
0.0216157774823716	-0.0701491933953733	-0.0548613906034893	0.00245135405867802
-0.0126044833423318	-0.129998151011255	0.0553871615742443	0.0518731302737386
0.0112169747376714	0.256683397529023	0.0199662974231749	0.106437483942417
0.0460261271558902	-0.0705245770988168	0.0338890308428010	-0.0446308986236706
0.0943166949437031	0.0505285390841616	-0.0195035738110643	0.167487477567377
0.135259487280267	0.0552678864079429	0.238821916565946	0.0862296906749685
0.0875453986019198	0.0784631509091033	-0.217079718944010	-0.0356133456456784
-0.0158190438248100	-0.00896061868503123	0.0883022765538066	0.140742814484257
-0.00562477468460862	-0.123647304041167	0.0397412326953713	-0.223447468723303
-0.0953832191781234	0.149896268401600	-0.121606964348794	0.0667386584029612
0.0777511295521009	-0.194193137077328	-0.0701249466656378	-0.148758282075666
-0.0942118825469530	0.0133200649183651	0.0412623801581108	0.0629003119839458
0.0593213706856075	0.0382306532887301	0.00990352522367593	0.00579130632368272
-0.0439309034378706	0.0149586686272571	0.00167477923932499	0.0272381201876377
0.00372364646288916	-0.00460621724446253	0.0491049754643056	-0.000607209914857042
0.0482235195125004	0.000307379950552073	0.00339510557288126	0.0169151498887303

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı: Ahmet LOĞOĞLU
Uyruğu: Türkiye (T.C)
Doğum Tarihi ve Yeri:
Medeni Durum:
e-mail:
Yazışma Adresi:

EĞİTİM

Derece	Kurum	Mezuniyet Tarihi
Yüksek Lisans	Erciyes Üniversitesi, Elektronik Mühendisliği	2006
Lisans	Erciyes Üniversitesi, Elektronik Mühendisliği	1998
Lise	Kadirli Lisesi, Kadirli-Osmaniye	1993

İŞ DENEYİMLERİ

Yıl	Kurum	Görev
2000-Halen	Osmaniye Korkut Ata Üniversitesi	24

YABANCI DİL

İngilizce

YAYINLAR

1. Logoglu, A., Kockanat, S., & Karaboga, N. (2022). A Novel Approach for Polyphase Filter Bank Design Using ABC Algorithm. Elektronika ir Elektrotechnika,28(6), 27-34.

2. Kockanat, S., Koza, T., Karaboga, N., & Logoglu, A. (2018). Adaptive FIR filtering using ABC algorithm: a noise reduction application on mitral valve doppler signal. Elektronika ir Elektrotechnika, 24(5), 62-68.

3. Logoglu, A., Koza, T., Kockanat, S., & Karaboga, N. (2018, May). Examination of success of the finite impulse response digital filter designed using the bat algorithm. In 2018 26th Signal Processing and Communications Applications Conference (SIU) (pp. 1-4). IEEE.

4. Logoglu A., K.S., Karaboga N., Performance analysis of ga, pso and abc algorithms for polyphase filter bank design. , in International conference on engineering, natural and applied science. 2021: Osmaniye. p. 132.

5. Loğoğlu A., Koçkanat S., Karaboğa N. ABC Algoritmasının Kontrol Parametrelerinin Çok fazlı Süzgeç Bankası Tasarımına Etkisi ARACONF 2021 - 2nd International Conference On Access To Recent Advances In Engineering And Digitalization, Kayseri, Türkiye, 10 - 11 Mart 2021, ss.111

6. Karaboğa N., Loğoğlu A. Su Dağıtım Şebekesinin Optimizasyonunda Genetik Algoritmanın Kullanılması ELECO-2006 Elektrik-Elektronik-Bilgisayar Mühendisliği Sempozyumu ve Fuarı, Bursa, Türkiye, 1 - 04 Aralık 2006, ss.1