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

ÇOK AMAÇLI OPTİMİZASYON ALGORİTMALARININ SIKIŞTIRILMIŞ ALGILAMADA KULLANILMASI

Hazırlayan
Murat Emre ERKOÇ

Danışman
Prof. Dr. Nurhan KARABOĞA

Doktora Tezi

Mart 2023
KAYSERİ

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

ÇOK AMAÇLI OPTİMİZASYON ALGORİTMALARININ SIKIŞTIRILMIŞ ALGILAMADA KULLANILMASI

(Doktora Tezi)

Hazırlayan
Murat Emre ERKOÇ

Danışman
Prof. Dr. Nurhan KARABOĞA

Bu çalışma, Türkiye Bilimsel ve Teknolojik Araştırma Kurumu tarafından Yurt İçi Öncelikli Alanlar Doktora Burs Programı (2211-C) ve Erciyes Üniversitesi Bilimsel Araştırma Projeleri Birimi tarafından FDK-2021-10946 kodlu proje ile desteklenmiştir.

Mart 2023
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.

Murat Emre ERKOÇ

İmza

“Çok Amaçlı Optimizasyon Algoritmalarının Sıkıştırılmış Algılamada Kullanılması” adlı Doktora tezi, Erciyes Üniversitesi Lisansüstü Tez Önerisi ve Tez Yazma Yönergesi’ ne uygun olarak hazırlanmıştır.

Hazırlayan	**Danışman**
Murat Emre ERKOÇ	Prof. Dr. Nurhan KARABOĞA
İmza	İmza

Elektrik Elektronik Mühendisliği ABD Başkanı

Prof. Dr. Ferhat DALDABAN

İmza

TEŞEKKÜR

Çalışmalarımın her aşamasında beni destekleyen, her türlü yardımı ve fedakârlığı sağlayan, bilgi birikimi ve tecrübesini benimle paylaşarak akademik çalışmalarımda ilerlememi sağlayan danışman hocam Sayın Prof. Dr. Nurhan KARABOĞA'ya teşekkürü bir borç bilirim.

Tez izleme komitesinde bulunarak rapor dönemlerinde fikir, öneri ve yapıcı geri bildirimleri ile tezimin gelişmesine katkı sağlayan Sayın Prof. Dr. İbrahim DEVELİ ve Sayın Prof. Dr. Enis GÜNAY'a teşekkür ederim.

Tez çalışmamı, 2211-C Yurt İçi Öncelikli Alanlar Doktora Burs Programı ile destekleyen TÜBİTAK Bilim İnsanı Destek Programları Başkanlığı'na teşekkür ederim.

FDK-2021-10946 numaralı proje ile Doktora tezimi destekleyen Erciyes Üniversitesi Bilimsel Araştırma Projeleri Birimi'ne teşekkürlerimi sunarım.

Hayatım boyunca maddi, manevi her türlü desteği vererek bugünlere gelmemi sağlayan anne ve babama, bu tez çalışmasını gerçekleştirmek için beni motive eden eşime ve her zaman ilham ve moral kaynağı olan kızıma teşekkür ederim.

Murat Emre ERKOÇ

Mart 2023, KAYSERİ

ÇOK AMAÇLI OPTİMİZASYON ALGORİTMALARININ SIKIŞTIRILMIŞ ALGILAMADA KULLANILMASI

Murat Emre ERKOÇ

Erciyes Üniversitesi, Fen Bilimleri Enstitüsü
Doktora Tezi, Mart 2023
Danışman: Prof. Dr. Nurhan KARABOĞA

ÖZET

Sıkıştırılmış algılama, bir sinyali belirli koşullar altında kendi uzunluğundan daha az sayıda örnek kullanarak geri oluşturabilen bir sinyal işleme yöntemidir. Bu yöntemde, sinyalin az sayıda örnekleri kullanılarak elde edilmiş olan ölçümlerinden orijinal sinyali geri oluşturmak için Seyrek Sinyal Geri Oluşturma (SSR) algoritmaları kullanılmaktadır. Literatürde çeşitli SSR yöntemleri önerilmiştir. Son yıllarda, SSR alanında Çok Amaçlı Optimizasyon Algoritmaları (MOOA) da kullanılmaktadır. SSR problemi seyreklik ve ölçüm hatası olarak iki amaç fonksiyonu şeklinde ifade edilebilmektedir. Bu iki amaçlı optimizasyon probleminin optimizasyonunda MOOA'ların kullanımı, düzenleme parametresinin algoritma performansı üzerindeki hassas etkisini ortadan kaldırmaktadır. Bu tez çalışmasında, öncelikle literatürde yer alan çok amaçlı seyrek geri oluşum metotları incelenmiş ve ardından MOABC algoritmasına dayalı bir SSR yöntemi önerilmiştir. Önerilen yöntem, seyrekliği ve ölçüm hatasını aynı anda optimize etmektedir. Ayrıca NSGA-II, MOEA/D ve MOPSO gibi diğer MOOA'lar da SSR problemine uygulanmıştır. Algoritmaların literatürdeki SSR algoritmalarıyla karşılaştırılmasında EKG, görüntü ve SOTF gibi çeşitli test sinyalleri kullanılmıştır. Elde edilen sonuçlara göre, önerilen yöntemin karşılaştırılan algoritmalara göre bazı üstünlükleri bulunmaktadır. Ayrıca bu tez çalışmasında, klasik SSR metotlarından olan OMP algoritmasının FPGA üzerinde donanım uygulaması yapılmıştır. Tasarlanan OMP algoritmasının en küçük kareler yaklaşımı aşamasında QR ayrışım metodu kullanılmıştır. Tasarlanan OMP algoritması, uzunluğu 128, seyreklik seviyesi 5 olan sinyali 32 adet ölçüm ile geri oluşturabilmektedir.

Anahtar Kelimeler: Sıkıştırılmış Algılama, Seyrek Geri Oluşum, Çok Amaçlı Optimizasyon; Sayısal Tasarım, FPGA.

MULTIOBJECTIVE OPTIMIZATION ALGORITHMS IN COMPRESSED SENSING

Murat Emre ERKOÇ

Erciyes University, Graduate School of Natural and Applied Sciences
PhD Thesis, March 2023
Supervisor: Prof. Dr. Nurhan KARABOĞA

ABSTRACT

Compressed sensing (CS) is a signal processing method that can reconstruct a signal using fewer samples than its length. Sparse Signal Reconstruction (SSR) algorithms are used to reconstruct the original signal from the measurements obtained from a small number of samples of the signal. Various SSR methods have been proposed in the literature. Recently, Multi-Objective Optimization Algorithms (MOOA) have been used in SSR. The fact that the sparse reconstruction problem can be expressed as two objective functions, sparsity and measurement error, makes it possible to use MOOA in this field. In addition, the use of MOOA eliminates the sensitive effect of the regulation parameter on the SSR algorithm's performance. In this thesis, first of all, multi-objective sparse reconstruction methods in the literature were examined and analyzed. Then, an SSR method based on the MOABC algorithm has been proposed. The proposed method optimizes the sparsity and measurement errors simultaneously. In addition, other MOOAs such as NSGA-II, MOEA/D and MOPSO have also been used in the optimization of the SSR problem. Various test signals such as ECG, image and SOTF were used to compare the algorithms. According to the obtained results, the proposed method has some advantages over the compared algorithms. Furthermore, an FPGA-based hardware implementation of the OMP algorithm was carried out, in which the QR decomposition method was utilized for the least square's solution of OMP. The OMP algorithm designed in this study is capable of recovering a signal with a length of 128 and a sparsity level of 5 using only 32 measurements.

Keywords: Compressed Sensing, Sparse Reconstruction, Multi-Objective Optimization, Digital Design, FPGA.

İÇİNDEKİLER

ÇOK AMAÇLI OPTİMİZASYON ALGORİTMALARININ SIKIŞTIRILMIŞ ALGILAMADA KULLANILMASI

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

SIKIŞTIRILMIŞ ALGILAMADA GENEL BİLGİLER

1.1 Seyreklik 5

1.2. Ölçüm İşlemi 7

1.3. Seyrek Sinyal Geri Oluşturma Algoritmaları 9

2. BÖLÜM

ÇOK AMAÇLI OPTİMİZASYON ALGORİTMALARININ SIKIŞTIRILMIŞ ALGILAMADA KULLANILMASI

2.1. Çok Amaçlı Optimizasyon Algoritmaları 13

2.1.1. NSGA II 15

2.1.2. MOEA/D 16

2.1.3. MOPSO 17

2.2. Yerel arama metodu 18
2.3 Diz noktası seçme yöntemleri 21
2.3.1. Ağırlıklı toplam yöntemi 22
2.3.2. Açıya dayalı yöntem 23
2.3.3. Kink Yöntemi 24
2.4. Literatürdeki MOSR Yöntemlerinin İncelenmesi 25
2.4.1. Seyrek Optimizasyon Problemleri için MOEA/D ile Beraber Yinelemeli Eşikleme Algoritması 25
2.4.2. SSR için Evrimsel Çok Amaçlı Yaklaşım 26
2.4.3. SSR için Ayrıştırmaya Dayalı Çok Aşamalı ve Çok Amaçlı bir Yaklaşım. 27
2.4.4. SSR için İki Aşamalı Evrimsel bir Yaklaşım 27
2.4.5. MOSR için Evrimsel Hibrit Algoritma 28
2.4.6. MOSR için Uyarlanabilir Ayrıştırma Tabanlı Evrimsel Yaklaşım 29
2.4.7. Seyrek Optimizasyon için Tercihe Dayalı bir MOEA Yaklaşımı 29
2.4.8. SSR için GMC'ye Dayalı Çok Amaçlı bir Yaklaşım 30
2.4.9. Seyrek optimizasyon için MOBSO Algoritması 30
2.4.10. SSR için Önerilen Yeni bir MOOA 31
2.5. Yöntemlerin Değerlendirilmesi 31

3. BÖLÜM

ÇOK AMAÇLI YAPAY ARI KOLONİ ALGORİTMASINA DAYALI SEYREK SİNYAL GERİ OLUŞUM YÖNTEMİ

3.1. ABC Algoritması 37
3.2. Seyrek Sinyal Geri Oluşumu için Çok Amaçlı Yapay Arı Koloni Algoritması 40
3.2.1. Bölgesel Arama Metodu 44
3.2.2. Nihai çözümün elde edilmesi 45

4. BÖLÜM

UYGULAMA VE SONUÇLAR

4.1. Algoritmaların SOTF Test Problemleri Kullanılarak Test Edilmesi 49
4.2. Nihai Çözüm Bulma Metotlarının Karşılaştırılması 57
4.3. Görüntü Geri Oluşturma Problemi 58
4.4. Tek Amaçlı ve Çok Amaçlı Optimizasyon Algoritmalarının SSR Probleminin Optimizasyonunda Karşılaştırılması 63

4.5. EKG Sinyalinin Geri Oluşturması 66

5. BÖLÜM

OMP ALGORİTMASININ FPGA ÜZERİNDE UYGULANMASI

5.1. Giriş 73

5.2. OMP Algoritması 75

5.3. OMP Algoritmasının FPGA Üzerinde Uygulanması 79

5.3.1. Veri Yolu 80

5.3.1.1. Hesaplama Birimi 80

5.3.1.2. Hafıza Birimleri 81

5.3.2. Kontrol Birimi 82

5.3.3. Tasarımın İşleyişi 82

5.4. Değerlendirme ve Sonuçlar 86

6. BÖLÜM

TARTIŞMA SONUÇ VE ÖNERİLER

TARTIŞMA SONUÇ VE ÖNERİLER 91

KAYNAKÇA 95

ÖZGEÇMİŞ 109

KISALTMALAR

ABC : Yapay Arı Koloni

ADEA : Uyarlanabilir Ayrıştırma Tabanlı Evrimsel Yaklaşım

BP : Temel Takip

CS : Sıkıştırılmış Algılama

CoSaMP : Sıkıştırılmış Örnekleme Eşleştirme Takibi

DE : Diferansiyel Gelişim

FISTA : Hızlı Yinelemeli Büzülme Eşiği Algoritması

FPGA : Alanda Programlanabilir Kapı Dizileri

FSM : Sonlu Durum Makinesi

GA : Genetik Algoritma

GMC : Gauss Karışım Kümelenmesi

GMM : Gauss Karışım Modeli

HDL : Donanım Tanımlama Dili

HLS : Yüksek Seviye Sentez

IHalfT : Yinelemeli Yarım Eşikleme

IHardT : Yinelemeli Sert Eşikleme

IST : Yinelemeli Yumuşak Eşikleme

LB : Doğrusal Bregman

LBEA : Doğrusal Bregman Evrimsel Algoritması

ME : Ölçüm Hatası

MOABC : Çok Amaçlı Yapay Arı Koloni Algoritması

MOABC-SR : Seyrek Geri Oluşumu için Çok Amaçlı Yapay Arı Koloni Algoritması

MOBSO-OS : Seyrek Optimizasyon için Çok Amaçlı Beyin Fırtınası Optimizasyon Algoritması

MOEA/D : Ayrıştırma Tabanlı Çok Amaçlı Evrimsel Algoritma

MOOA : Çok Amaçlı Optimizasyon Algoritması

MOOP : Çok Amaçlı Optimizasyon Problemi

MOPSO : Çok Amaçlı Parçacık Sürü Optimizasyonu

MOSR : Çok Amaçlı Geri Oluşum

NSGA : Baskılanmayanları Sıralayan Genetik Algoritma

PALM : İlkel Artırılmış Lagrange Çarpanı

PF : Pareto Cephesi

PGRASP : Olasılık Tabanlı Açgözlü Rastgele Uyarlamalı Arama Prosedürü

PSEA : Parçacık Sürüsünden Esinlenmiş Evrimsel Algoritma

PRD : Yüzde Ortalama Kare Farkı

PSO : Parçacık Sürü Optimizasyonu

RE : Geri Oluşturma Hatası

RIP : Kısıtlı İzometri Özelliği

RSE : Bağıl Seyreklik Hatası

SSIM : Yapısal Benzerlik İndeksi Ölçüsü

SSR : Seyrek Sinyal Geri Oluşumu

StEMO : Yumuşak Eşiklemeli Evrimsel Çok Amaçlı Algoritma

SPLS : Seyrek Tercih Tabanlı Yerel Arama

XSG : Xilinx Sistem Üreteci

TABLOLAR LİSTESİ

Tablo 2.1. MOSR yöntemlerinde kullanılan bazı yerel arama yöntemlerinin özeti. 20

Tablo 2.2. MOSR yöntemlerinin tasarım özellikleri. 35

Tablo 2.3. MOSR yöntemlerinin karşılaştırıldığı algoritmalar test sinyalleri ve hesaplama karmaşıklıkları. 36

Tablo 4.1. Seyrek optimizasyon test problemleri 50

Tablo 4.2. Bu tez çalışmasında kullanılan algoritma parametreleri. 50

Tablo 4.3a. Gürültüsüz sinyaller için geri oluşturma hatasının ortalama değerleri.... 51

Tablo 4.3b. Gürültülü sinyaller için geri oluşturma hatasının ortalama değerleri 52

Tablo 4.4a. Gürültüsüz sinyaller için ölçüm hatasının ortalama değerleri 53

Tablo 4.4b. Gürültülü sinyaller için ölçüm hatasının ortalama değerleri 54

Tablo 4.5a. Gürültüsüz sinyaller için bağıl seyreklik hatasının ortalama değerleri. .. 55

Tablo 4.5b. Gürültülü sinyaller için bağıl seyreklik hatasının ortalama değerleri. 56

Tablo 4.6. Son çözümü bulmak için kullanılan metotlara ait RE değerleri. 57

Tablo 4.7. Çeşitli geri elde etme algoritmalarına ait PSNR ve SSIM değerleri 61

Tablo 4.8. Tek amaçlı optimizasyon ve çok amaçlı optimizasyon algoritmalarının karşılaştırılmasında kullanılan NSGA-II, MOABC, GA ve ABC parametreleri. 63

Tablo 4.9. Çeşitli seyreklik seviyelerine sahip test problemleri için NSGA-II, MOABC, GA ve ABC ile geri oluşturulan sinyallerin ortalama RE ve MSE değerleri. 65

Tablo 4.10. Çeşitli ölçüm sayılarına sahip test problemleri için NSGA-II, MOABC, GA ve ABC ile geri oluşturulan sinyallerin ortalama RE ve MSE değerleri. 66

Tablo 4.11. Elde edilen performans sonuçlarını değerlendirme kriterleri [123]. 67

Tablo 4.12. Geri elde edilen EKG sinyallerinin PRD ve SNR değerleri 68

Tablo 5.1. FPGA tasarımı ve kaynak kullanımı ile ilgili özellikler. 87

Tablo 5.2. Literatürde yer alan bazı çalışmaların tasarım özellikleri. 88

ŞEKİLLER LİSTESİ

Şekil 1.1 *Cameraman* görüntü sinyali ve dalgacık katsayıları. 6
Şekil 1.2. Geri oluşturulan dalgacık katsayıları ve görüntü. 6
Şekil 1.3. (a) Geleneksel örnekleme, (b) CS. 7
Şekil 1.4. SSR algoritmalarının sınıflandırılması. 11
Şekil 2.1. Diz noktası içeren bir PF 21
Şekil 2.2. Amaç değerlerinin ağırlıklı toplamı. 22
Şekil 2.3 Açı tabanlı yöntem. 23
Şekil 3.1. ABC algoritmasının akış şeması 39
Şekil 4.1. Test görüntüleri: a) *Cameraman,* b) *Lena,* c) *Barbara,* d) *Houses,* e) *Peppers*. 60
Şekil 4.2. a) Orijinal *Cameraman* görüntüsü, b) görüntünün dalgacık dönüşümü, c) geri elde edilen dalgacık katsayıları, d) MOABC-SR tarafından geri elde edilen görüntü. 62
Şekil 4.3. Sırasıyla; KnMOPSO, IHT, OMP, NSGA II, Homotopy, FISTA, PALM ve MOEA/D algoritmaları kullanılarak geri elde edilen *Cameraman* görüntüsü. 62
Şekil 4.4. Farklı a) seyreklik ve b) ölçüm sayılarından oluşan SSR problemlerine ait ortalama RE değerleri. 64
Şekil 4.5. MIT-BIH-100 EKG Orijinal ve Geri Oluşturulan Sinyal 69
Şekil 4.6. MIT-BIH-101 EKG Orijinal ve Geri Oluşturulan Sinyal 69
Şekil 4.7. MIT-BIH-103 EKG Orijinal ve Geri Oluşturulan Sinyal 70
Şekil 4.8. MIT-BIH-117 EKG Orijinal ve Geri Oluşturulan Sinyal 70
Şekil 4.9. MIT-BIH-207 EKG Orijinal ve Geri Oluşturulan Sinyal 71
Şekil 5.1. ***Q, R*** matrisleri ve sinyalin sıfır olmayan elemanlarının hesaplandığı birinci kısım 83
Şekil 5.2. ***R*** matrisinin tersinin alındığı ve sinyalin geri oluşturulduğu tasarım. 85
Şekil 5.3. Test sinyalleri 88
Şekil 5.4. Geri elde edilen sinyaller ve değeri sıfırdan farklı olan indisler. 89

GİRİŞ

Teknolojinin gelişmesiyle birlikte büyük verilerin etkili ve verimli bir şekilde işlenmesi önem kazanmaktadır. Veri toplama ve işleme teknikleri teknolojik gelişmelerin hızını ve sınırını belirlemektedir. Donoho, Candès, Romberg ve Tao tarafından ortaya çıkarılan Sıkıştırılmış Algılama (Compressed Sensing, CS), belirli şartları sağlayan sinyallerin geleneksel yöntemlerin kullandığı örnek sayısından daha az sayıda örnek kullanarak geri elde edilebileceğini ifade eden bir sinyal işleme tekniğidir [1-4]. CS yöntemi sıkıştırma ve algılama işlemlerini aynı anda gerçekleştirmektedir. Büyük veri çağında artan bilgi miktarıyla birlikte yüksek boyutlu veri sistemlerinin karmaşıklığı ve işlem maliyeti kritik hale gelmiştir. CS, depolama birimlerinin kapasitelerini verimli kullanımının yanı sıra sinyal işleme yöntemlerinin çalışma sürelerinin kısaltılmasını ve enerji tüketimlerinin azaltılmasını sağlamaktadır. CS genel olarak görüntüleme sistemleri, biyomedikal uygulamalar, iletişim sistemleri, sensör ağı, radar hedefi, video işleme ve yüz tanıma alanlarında kullanılmaktadır. CS'nin kullanıldığı alanlara literatürden örnek olarak şunlar verilebilir [5]:

- **Görüntüleme**
  - Medikal Görüntüleme [6].
  - Spektral Görüntüleme [7, 8].
  - Radar Görüntüleme [9-11].
  - Görüntü İşleme [12, 13].
- **Biyomedikal Uygulamalar**
  - EEG Sinyal işleme [14].

- Genomik Algılama [15].

❖ **Haberleşme sistemleri**

- Kablosuz Sensor Ağları [16-20].
- Çoklu Giriş-Çoklu Çıkış Haberleşmesi [21-23].
- Ultra Geniş Bant Haberleşme [24, 25].
- Anten Dizileri [26, 27].

❖ **Desen Tanıma**

- Yüz Tanıma [28, 29].
- Konuşma Tanıma [30].
- Bilgisayarlı Görü ve Desen Tanıma [31].

❖ **Konuşma ve Ses İşleme**

- Konuşma Kodlama [32].
- Okyanus Seslerini İzleme [33].
- Ses Yerini Belirleme [34].

❖ **Video İşleme**

- Video Geri Oluşturma [35].
- Tek Piksel Algılayıcı Kullanarak 3-Boyutlu Video Algılama [36].

❖ **Mikro-Nano Elektronik ve VLSI Uygulamaları** [37-39].

CS yönteminde, bir sinyali kendi uzunluğundan daha küçük örnek sayısı kullanarak geri elde edebilmek için sinyalin ve ölçüm matrisinin bazı özelliklere sahip olması gerekmektedir. Bu özellikler sinyalin seyrek olması, ölçüm matrisinin tutarsızlık ve sınırlı izometri özelliğine (Restricted Isometry Property, RIP) sahip olmasıdır. Sıfırdan farklı olan eleman sayısı sıfır olan eleman sayısından daha az olan sinyale seyrek sinyal denir. Bazı sinyaller ya kendi alanında ya da Fourier, Dalgacık gibi dönüşüm alanlarında seyrektir. Gauss dağılımına sahip olarak rastgele üretilmiş ölçüm matrisi tutarsızlık ve RIP şartlarını büyük oranda sağlamaktadır.

CS yönteminin diğer aşaması ise sinyalin geri oluşturulmasıdır. Seyrek sinyal geri oluşum (Sparse Signal Reconstruction, SSR) işlemi için literatürde çeşitli algoritmalar önerilmiştir. Bunlar l_1-minimizasyonuna dayalı konveks yöntemler, l_0-minimizasyonuna dayalı açgözlü algoritmalar ve l_q-minimizasyonuna dayalı konveks olmayan yöntemler şeklinde sınıflandırılabilir. Özellikle l_1-minimizasyonuna dayalı konveks yöntemler bu

alanda yaygın olarak kullanılmaktadır. Bu metotların haricinde son yıllarda sezgisel optimizasyon metotları da sinyal geri oluşumunda kullanılmaya başlamıştır.

Tez çalışması kapsamında öncelikle çok amaçlı optimizasyon algoritmalarının CS'de kullanıldığı çalışmalar incelenmiştir. Bu incelemede, literatürdeki çalışmalar belirli özelliklerine göre sınıflandırılmış ve karşılaştırılmıştır. Baskılanmayanları Sıralayan Genetik Algoritma (Non-dominated Sorting Genetic Algorithm-II, NSGA-II), Ayrıştırma Tabanlı Çok Amaçlı Evrimsel Algoritma (Multi-Objective Evolutionary Algorithm based on Decomposition, MOEA/D), Çok Amaçlı Parçacık Sürü Optimizasyon (Multi-Objective Particle Swarm Optimization, MOPSO) algoritmalarına dayalı çok amaçlı seyrek geri oluşum (Multi-objective Sparse Reconstruction, MOSR) yöntemleri çeşitli SSR problemlerine uygulanarak incelenmiştir.

Tez çalışmasında Çok Amaçlı Yapay Arı Koloni (Multi-Objective Artificial Bee Colony, MOABC) algoritmasına dayalı bir MOSR yöntemi önerilmiştir. Bu yöntemde temel ABC algoritması SSR probleminin seyreklik ve ölçüm hatası terimlerini optimize edecek şekilde yeniden tasarlanmıştır. Önerilen yöntemin etkinliği çeşitli test problemleri kullanılarak kanıtlanmıştır. Önerilen yöntem seyrek görüntü geri oluşturulması ve EKG sinyali geri oluşturulması problemleri gibi CS'nin uygulama alanlarından bazılarında test edilmiştir. Bu testlerde, geri elde edilen sinyalin ölçüm hatası, seyreklik hatası ve geri oluşum hatası gibi kriterler kullanılmıştır. Elde edilen sonuçlara göre, önerilen çok amaçlı optimizasyon algoritması SSR işleminde karşılaştırılan diğer algoritmalara göre bazı avantajlara sahiptir.

Ayrıca bu tez çalışmasında klasik yöntemlerden olan Dik Eşleştirme Arayış (Orthogonal Matching Pursuit, OMP) algoritmasının Alanda Programlanabilir Kapı Dizileri (Field Programmable Gate Array, FPGA) üzerinde donanımı tasarlanmıştır. CS'nin donanım üzerinde uygulanması söz konusu olduğunda, CS'de kullanılan geri oluşturma algoritmasının hesaplama karmaşıklığı, yineleme sayısı ve geri oluşturmadaki doğruluk oranı gibi kriterler değerlendirilmektedir. Bu kriterler göz önünde bulundurulduğunda OMP algoritması diğer SSR algoritmalarına göre donanım tasarımı açısından daha uygundur. OMP'nin donanım uygulaması hakkında literatürde çeşitli çalışmalar mevcuttur. Bu tez çalışması kapsamında tasarlanan OMP algoritmasının etkinliği değerlendirilmiş ve literatürdeki benzer çalışmalarla karşılaştırılmıştır. Bu

karşılaştırmada tasarımda kullanılan blok RAM, lojik elemanları ve sayısal sinyal işlemci sayıları gibi kullanılan kaynak sayılarının yanı sıra tasarımın maksimum çalışma frekansı ve toplam saat çevrim sayısı gibi algoritmanın performansıyla alakalı kriterler değerlendirilmiştir.

Tez çalışmasının ilk bölümünde CS ile ilgili genel bilgiler yer almaktadır. CS'nin uygulanabilmesi için önemli olan seyreklik kavramı, ölçüm matrisinin sağlaması gereken kriterler ve ölçüm matrisi çeşitleri açıklanmıştır. SSR problemleri matematiksel olarak ifade edilmiş ve problemin çözümü için kullanılan metotlardan bahsedilmiştir. SSR probleminin çok amaçlı bir optimizasyon problemi olarak ifade edilmesi de ilk bölümde açıklanmıştır. İkinci bölümde, çok amaçlı optimizasyon algoritmalarıyla ilgili bilgiler verilmiş ve bu algoritmaların SSR probleminin çözümünde kullanılma yöntemleri açıklanmıştır. Ayrıca literatürdeki MOSR metotlarının detaylı olarak sınıflandırılması ve karşılaştırılmaları yapılmıştır. Yaygın olarak kullanılan MOSR metotlarının tasarım özellikleri değerlendirilmiştir. Üçüncü bölümde, parametre sayısı az olan MOABC algoritmasına dayalı bir MOSR yöntemi önerilmiştir. Ayrıca çok amaçlı algoritmanın oluşturduğu domine edilemeyen çözümlerin içinden nihai çözümün elde edilmesi için ölçüm hatasına dayalı seçim yapan bir yöntem uygulanmıştır. Dördüncü bölümde, tasarlanan MOABC algoritmasına dayalı metot çeşitli sinyallerle test edilmiştir. Ayrıca önerilen yöntem farklı sınıflardan SSR metotlarıyla karşılaştırılmıştır. NSGA-II, MOEA/D, MOPSO algoritmaları SSR probleminin optimizasyonunda kullanılmış ve çeşitli problemler üzerinde test edilmiştir. Bu bölümde karşılaştırma yapılırken kullanılan sinyaller yapay olarak üretilen sinyaller, görüntü sinyalleri ve EKG sinyallerinden oluşmaktadır. Beşinci bölümde klasik SSR metotlarından olan OMP algoritması FPGA üzerinde uygulanmıştır. Yapılan tasarımın performansı değerlendirilip literatürdeki benzer çalışmalarla karşılaştırılmıştır. Son bölümde ise çalışmalar ve elde edilen sonuçlar genel olarak özetlenmiştir.

1. BÖLÜM

SIKIŞTIRILMIŞ ALGILAMADA GENEL BİLGİLER

CS, bir sinyali kendi uzunluğundan daha az sayıda örnek kullanarak geri elde edebilen bir sinyal işleme metodudur. CS ile sinyal örneklenirken aynı zamanda sıkıştırılmış olur. CS'nin uygulanabilmesi için sinyalin seyrek olması ve ölçüm matrisinin bazı özellikleri sağlaması gerekmektedir. Temel olarak CS; sinyalin seyrek olarak ifade edilmesi, ölçüm işlemi ve seyrek sinyal geri oluşturulması olmak üzere üç aşamadan oluşmaktadır [40].

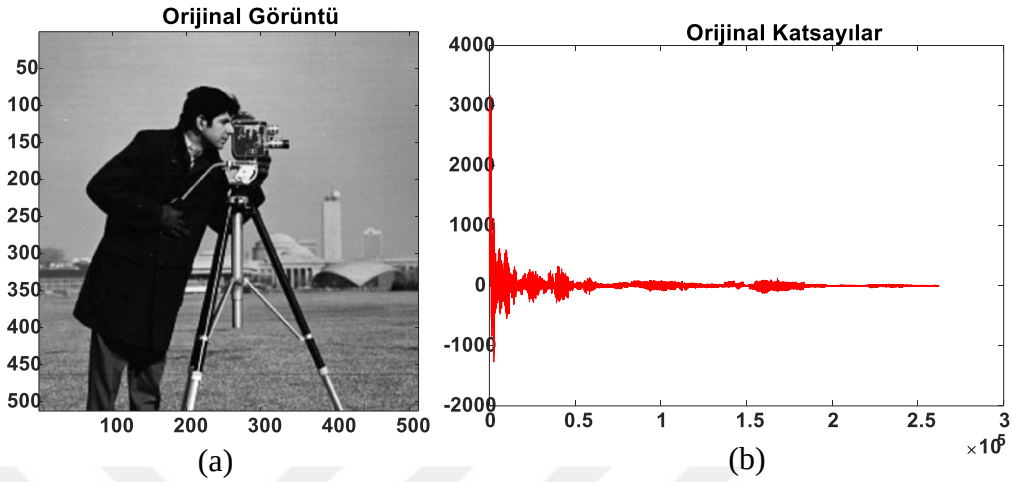
1.1 Seyreklik

CS'nin temel aşamalarından biri uygun bir dönüşüm veya özel bir yöntem kullanılarak sinyalin seyrek olarak ifade edilmesidir. CS geri oluşumunun uygulanabilmesi için bir sinyalin seyrek veya sıkıştırılabilir olması gerekmektedir. Eğer $\boldsymbol{x}$ sinyali kendi uzunluğundan çok daha küçük olan k adet sıfır olmayan elemana sahipse $\boldsymbol{x}$ sinyaline k-seyrek sinyal denir. Eğer $\boldsymbol{x}$ sinyalinin az sayıda elemanı diğer elemanlarından çok daha büyükse $\boldsymbol{x}$ sinyaline sıkıştırılabilir sinyal denir. Birçok sinyal orijinal bölgesinde veya bir dönüşüm bölgesinde seyrek olarak ifade edilebilir [3]. Bir $\boldsymbol{x}$ sinyali, taban vektörlerinin lineer kombinasyonu şeklinde Denklem (1.1)'deki gibi ifade edilebilir.

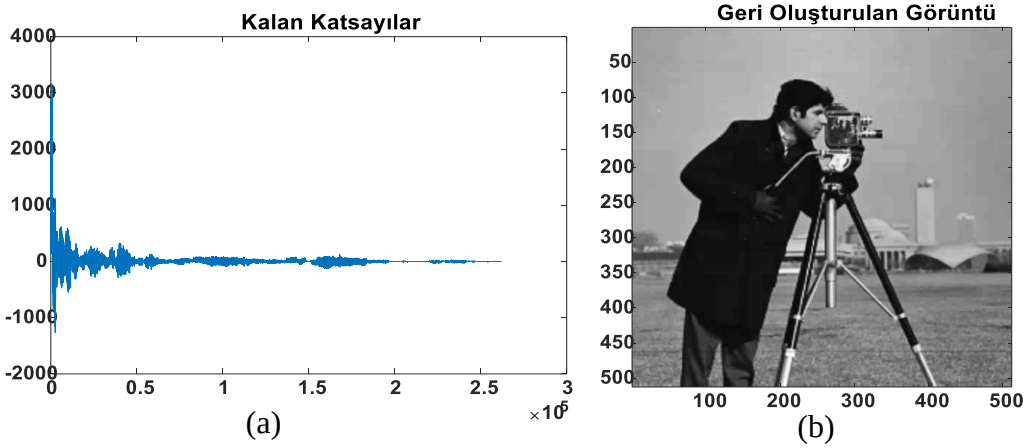
$$\boldsymbol{x} = \sum_{i=1}^{N} (\boldsymbol{s}_i \boldsymbol{\psi}_i) \tag{1.1}$$

Denklem (1.1)'de $\boldsymbol{s}$ vektörü $\boldsymbol{x}$ sinyalinin bir dönüşüm bölgesinde temsil edilmiş seyrek katsayıları, $\boldsymbol{\psi} \in R^{NxN}$ ise dönüşümü ifade eden ortonormal bir dönüşüm matrisidir. Bir sinyali seyrek olarak ifade etmek için sinyalin özelliğine göre Fourier, ayrık kosinüs veya dalgacık metodu gibi dönüşümler kullanılabilir [41]. Kullanılacak olan dönüşüm yöntemi, sinyalin özelliğine göre seçilir. Sinyal sinüzoidal parçalardan oluşuyorsa, sinyali seyrek olarak temsil etmek için Fourier dönüşümü, görüntü sinyallerini seyrek olarak ifade etmek için ise dalgacık veya ayrık kosinüs dönüşümü kullanılabilir.

Şekil 1.1'de *Cameraman* görüntü sinyali ve onun dalgacık katsayıları görülmektedir.



Şekil 1.1. Cameraman görüntü sinyali ve dalgacık katsayıları.



Şekil 1.2. Geri oluşturulan dalgacık katsayıları ve görüntü.

Şekil 1.1'de görüldüğü gibi, büyük katsayıların sayısı küçüklerinkinden daha azdır. Ayrıca büyük katsayılar, görüntünün oluşturulmasında önemli bir katkıya sahiptir. Küçük katsayıların çoğu atıldığında bile görüntü kalitesinde çok fazla kayıp olmayacaktır. Şekil 1.2'de katsayıların çoğu atıldıktan sonra geriye kalan katsayılar ve bu katsayılar kullanılarak geri oluşturulan görüntü yer almaktadır. Şekil 1.2'de katsayıların en küçük olanlarından yaklaşık %90,8'i sıfıra eşitlenmiş hali yer almaktadır ve katsayıların en büyük %9,2'si kullanılarak görüntü yeniden oluşturulmuştur. Şekil 1.1 ve 1.2'de görüldüğü gibi, bu katsayılar kullanılarak geri oluşturulmuş görüntü ile orijinal görüntü arasındaki fark çok azdır. Geri oluşturulan görüntünün SNR'si 37.1804'tür ve yapısal benzerlik indeksi ölçüsü (Structural Similarity, SSIM) [42] 0.9705'tir.

1.2. Ölçüm İşlemi

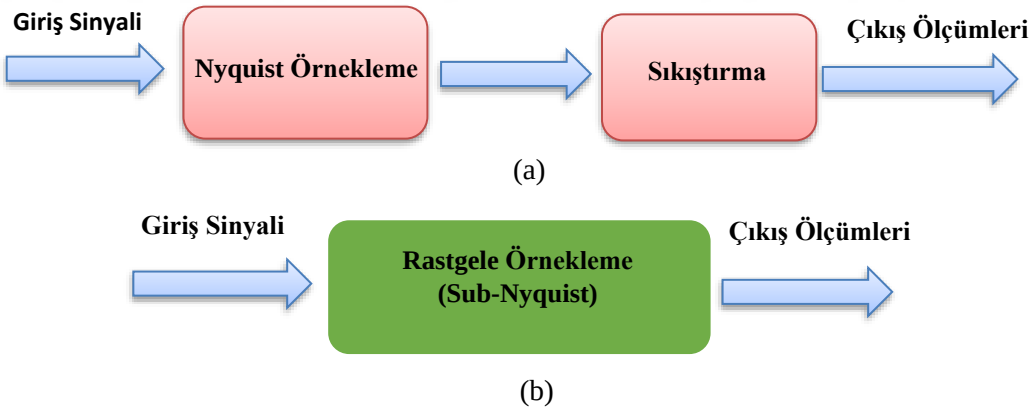
CS teoreminde ölçüm işlemi Denklem (1.2) kullanılarak ifade edilmektedir.

$$\mathbf{y} = \boldsymbol{Ax} \tag{1.2}$$

Denklem (1.2)'de, $\boldsymbol{x} \in R^N$ seyrek sinyali, $\boldsymbol{A} \in R^{MxN}$ $(M < N)$ ölçüm matrisini, $\boldsymbol{y} \in R^M$ gözlem vektörünü, M ölçüm sayısını ve N sinyal uzunluğunu temsil etmektedir. Eğer ölçüm aşamasında gürültünün var olduğu hesaba katılırsa Denklem (1.2), Denklem (1.3) şeklinde ifade edilebilir.

$$\boldsymbol{y} = \boldsymbol{Ax} + \boldsymbol{n},\ \boldsymbol{n} \in R^M \tag{1.3}$$

Ölçüm işlemi, N uzunluklu $\boldsymbol{x} \in R^N$ sinyalinden M adet örnek alınması ve $\boldsymbol{y} \in R^M$ gözlem vektörünün oluşturulması işlemidir. Burada ölçüm sayısı M, sinyal uzunluğu N'den küçüktür. CS'de ölçüm işleminde sinyal, kendi uzunluğundan daha az ölçüm sayısıyla örneklenmektedir. Geleneksel örnekleme ve CS tekniği arasındaki fark, Şekil 1.3'te gösterilmektedir [5].



Şekil 1.3. (a) Geleneksel örnekleme, (b) CS.

Geleneksel örneklemede sinyal kendi uzunluğunun en az iki katı büyüklüğünde örnek sayısıyla örneklenir, daha sonra sıkıştırma işlemi uygulanır. CS metodunda ise seyrek olan sinyaller kendi uzunluğundan daha az sayıda ölçüm kullanılarak sıkıştırılmış bir şekilde algılanmış olur. Tam bir geri oluşum için ölçüm matrisinin $\boldsymbol{x}$ sinyalindeki bilgileri koruması gerekmektedir. CS tekniğinin uygulanabilmesi ve seyrek sinyalin doğru bir şekilde geri elde oluşturulabilmesi için ölçüm matrisinin RIP ve tutarsızlık özelliklerini

sağlaması gerekmektedir [3, 41]. RIP için ölçüm matrisi ve sınırlı izometri sabitinin (δ_k), Denklem (1.4)' teki eşitsizliği sağlaması gerekmektedir.

$$(1-\delta_k)||\boldsymbol{x}||_2^2 \leq \left||\boldsymbol{A}\boldsymbol{x}|\right|_2^2 \leq (1+\delta_k)||\boldsymbol{x}||_2^2 \qquad (1.4)$$

Denklem (1.4)'te $\boldsymbol{x}$, k adet sıfır olmayan elamana sahip bir vektör, $\delta_k(0,1)$ ise sınırlı izometri sabitidir. Bu eşitsizlik, ölçüm matrisinin iki k-seyrek vektör arasındaki mesafeyi koruması gerektiğini belirtir. Bu eşitsizlik sağlandığında, herhangi bir k-seyrek vektör, ölçüm matrisinin sıfır uzayında olamaz ve sinyali geri oluşturmak mümkündür. Başka bir ifadeyle, ölçüm matrisinden elde edilen tüm k-kolon alt kümeleri neredeyse diktir [3]. Ancak hem k-seyrek hem de sıkıştırılabilir sinyallerin kararlı bir şekilde geri oluşturulabilmesi için yeterli koşul, 3k-seyrekliğinde rastgele bir $\boldsymbol{x}$ vektörü için ölçüm matrisinin Denklem (1.4)'ü sağlamasıdır [41].

Ölçüm matrisi ve temsil matrisi ile ilgili bir diğer ilişki ise tutarsızlıktır. Tutarlılık, ölçüm matrisinin herhangi iki öğesi ile temsil matrisi arasındaki korelasyonun ölçüsüdür. Ölçüm matrisi ve temsil matrisi arasındaki tutarlılık Denklem (1.5)'teki gibi hesaplanmaktadır.

$$\mu(\boldsymbol{A},\boldsymbol{\psi}) = \sqrt{n}.max_{1\leq k,j\leq n}|<\boldsymbol{A}_k,\boldsymbol{\psi}_j>| \qquad (1.5)$$

Denklem (1.5)'te yer alan $\mu(A,\psi)$ değeri 1 ile $\sqrt{n}$ arasındadır. Bu değer 1 olduğunda iki matris arasındaki tutarlılık minimum, $\sqrt{n}$ olduğunda ise maksimumdur. Bu matrisler ilişkili öğeler içeriyorsa, tutarlılık büyüktür. CS teorisinde, seyrek sinyalin geri oluşturulabilmesi için ölçüm matrisi ile temsil matrisi arasındaki tutarlılığın minimum olması gerekir. Bağımsız ve aynı şekilde dağılmış (independent and identically distributed, iid) rastgele Gauss tipi dağılıma sahip matrisler, RIP ve tutarsızlık özelliğini sağlamaktadırlar [41].

Alıcı tarafında aynı rastgele matris üretilemediğinden, rastgele üretilen matrislerin sinyalle birlikte iletilmesi gerekmektedir. Bu da gerçek zamanlı uygulamalar için verimli değildir. Dolayısıyla, literatürde rastgele matrislerin yanı sıra, sirkülasyon ve Toeplitz gibi bazı etkili deterministtik ölçüm matrisleri de önerilmiştir [5]. Ayrıca son zamanlarda, etkili bir ölçüm matrisi tasarlamak için derin öğrenme yöntemleri de kullanılmaktadır [43]. CS işleminin başarısını doğrudan etkilediğinden dolayı ölçüm matrisinin

belirlenmesi işlemi CS alanında önemli konulardan biridir. Literatürde çeşitli ölçüm matrislerinin performanslarının incelendiği çalışmalar da mevcuttur [44, 45].

1.3. Seyrek Sinyal Geri Oluşturma Algoritmaları

Denklem (1.2)'de verilen eksik belirtilmiş sistemden sinyalin geri oluşturulması için SSR algoritmaları kullanılır. SSR probleminin çözümünde bu algoritmalar genel olarak gözlem vektörü, seyreklik ve ölçüm matrisi gibi bilgileri kullanırlar. Literatürde çeşitli SSR algoritmaları önerilmiştir. Bu algoritmalar genel olarak konveks, konveks olmayan ve açgözlü algoritmalar olarak üç ana kategoride sınıflandırılırlar. Bunların dışında, sezgisel algoritmalar ve yapay zekâ teknikleri de SSR probleminin çözümünde kullanılmaktadır [43, 46-48]. Bu yöntemlerden bazıları klasik yöntemlerle beraber hibrit bir şekilde kullanılarak SSR problemini optimize etmektedirler.

CS geri elde edilmesindeki amaç Denklem (1.2)'de verilen ifadeyi sağlayan en seyrek çözümü bulmaktır. Sinyal seyrekliği l_0-normu ile ifade edilmektedir. CS'de SSR problemi Denklem (1.6)'daki gibi ifade edilmektedir.

$$\min_{x} ||x||_0 \text{ öyle ki } y=Ax \tag{1.6}$$

Denklem (1.6)'da $||x||_0$, l_0-normudur ve $\boldsymbol{x}$ vektörünün sıfır olmayan elemanların sayısını ifade etmektedir. Bu problem, Denklem (1.7)'deki l_0-norm problemine dönüştürülerek çözülebilir.

$$\min_{\boldsymbol{x}\in R^N} \{||\boldsymbol{y}-Ax||_2^2+\lambda||\boldsymbol{x}||_0\} \tag{1.7}$$

Denklem (1.7)'de düzenleme parametresi olan λ, seyreklik seviyesi ve ölçüm hatası terimleri arasında dengeyi sağlar. L_0-norm probleminin çözümünde OMP [49], Sıkıştırılmış Örnekleme Eşleştirme Takibi (Compressed Sampling Matching Pursuit-CoSaMP) [50] ve Yinelemeli Sert Eşik (Iterative Hard Thresholding, IHardT) yöntemi [51] gibi yöntemler kullanılır. Bu yöntemler genellikle ölçüm sayısı seyreklik seviyesinden çok daha fazla olduğu zaman iyi performans gösterirler. L_0-norm problemi NP-zor sınıfında olan bir problemdir. Dolayısıyla sinyal uzunluğu arttıkça problemi çözmek oldukça zorlaşmaktadır [52]. Bu dezavantajı ortadan kaldırmak için, l_0-norm

düzenlenmesi Denklem (1.8)'de olduğu gibi l_1-norm optimizasyon problemine dönüştürülebilir.

$$\min_{\boldsymbol{x}\in R^N} \{||\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}||_2^2+\lambda||\boldsymbol{x}||_1\} \tag{1.8}$$

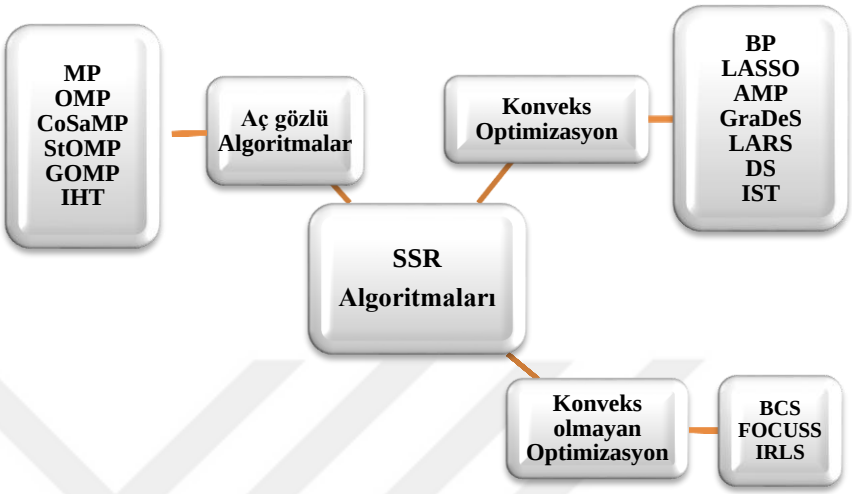
Konveks optimizasyon problemi olan l_1-norm düzenlemesi, doğrusal programlama yöntemleri ile çözülebilir. L_1-norm optimizasyon sınıfına ait olan algoritmalar, SSR probleminin çözümünde en yaygın kullanılan metotlardır. Bu metotlar gürültüye karşı dayanıklıdırlar. Dışbükey olmayan bir yaklaşım olan l_q-normu ise, l_1-norm düzenlemesinin Denklem (1.9)'da verilen ifadeye dönüştürülmüş şeklidir.

$$\min_{\boldsymbol{x}\in R^N} \{||\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}||_2^2+\lambda||\boldsymbol{x}||_q^q\}, 0<q<1 \tag{1.9}$$

Konveks olmayan yaklaşım, konveks yaklaşıma göre çok daha az sayıda ölçüm kullanarak seyrek çözümü elde edebilir. Ayrıca, konveks olmayan yaklaşımda mükemmel bir geri oluşum için RIP koşulunun daha zayıf bir koşulu sağlanması yeterli olmaktadır [5]. Ancak büyük boyutlu problemler için konveks olmayan yaklaşımın uygulanması zordur. SSR algoritmalarının sınıflandırılmasına bir örnek Şekil 1.4'te gösterilmektedir [53].

SSR metotlarında yer alan düzenleme parametresi, algoritmaların performansını önemli ölçüde etkilemektedir. Bu parametrenin performans üzerindeki etkisi ile ilgili çeşitli çalışmalar yapılmıştır [54]. Belirtilen çalışmada l_1-norm optimizasyon algoritmalarından yinelemeli yumuşak eşikleme (Iterative Soft Thresholding, IST) algoritmasının kullanıldığı deneyde optimum performansın sağlandığı parametre değeri yaklaşık 0,01

olarak bulunmuştur. Düzenleme parametresinin diğer değerlerinde performansın ani bir şekilde değiştiği belirtilmiştir.



Şekil 1.4. SSR algoritmalarının sınıflandırılması.

Başka bir çalışmada yinelemeli yarım eşikleme algoritmasının (Iterative Half Thresholding, IHalfT) uygulanmasında düzenleme parametresinin belirlenmesi ve performansa etkisi ile alakalı teorik çalışma yapılmıştır [52]. SSR algoritmalarının performansı düzenleme parametresine oldukça bağlıdır. Uygun olmayan bir seçim, algoritmaların performansını önemli ölçüde etkileyeceği için düzenleme parametresini optimum bir şekilde belirlemek zordur. Düzenleme parametresinin belirlenmesinden kaçınmak için SSR problemi, Denklem (1.10)'daki gibi çok amaçlı seyrek geri oluşturma problemine dönüştürülmüştür [54].

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) = \min_{\boldsymbol{x}} \left(||\boldsymbol{x}||_0,\ ||\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}||_2^2\right) \tag{1.10}$$

Denklem (1.10)'da ilk terim seyreklik, ikincisi ölçüm hatası terimidir. SSR probleminin çözümü bu iki terimin aynı anda optimize edilmesine bağlıdır. Çok amaçlı optimizasyon algoritmaları (Multi-Objective Optimization Algorithm- MOOA) birden fazla amaç fonksiyonuna sahip olan optimizasyon problemlerinin çözümü için kullanılırlar. MOOA'lar tüm farklı amaç fonksiyonlarını aynı anda optimize etmeye çalışır ve amaç fonksiyonları arasında çeşitli takas çözümleri sağlayabilirler. SSR probleminin optimize edilmesi gereken iki amaç fonksiyonu olduğu için MOOA'ların SSR probleminin optimizasyonunda kullanılması uygundur. Ayrıca bu yöntemler çeşitli klasik yöntemlerle

beraber kullanılarak yakınsama hızının geliştirilmesi gibi avantajlara sahip olabilirler. Bu yöntemlerde düzenleme parametresinin belirlenmesine ve sinyal seyrekliğinin önceden bilinmesine gerek yoktur. Literatürde çeşitli MOSR çözüm metotları önerilmiştir. Literatürde önerilen MOSR metotları bir sonraki bölümde detaylı olarak incelenmiştir.

2. BÖLÜM

ÇOK AMAÇLI OPTİMİZASYON ALGORİTMALARININ SIKIŞTIRILMIŞ ALGILAMADA KULLANILMASI

MOOA'lar 2012'den beri SSR problemlerinin çözümünde kullanılmaktadır. Bu algoritmaların CS alanında sinyalin geri elde edilmesinde kullanılmasıyla ortaya çıkan yöntem, literatürde yer alan çalışmalarda genelde MOSR olarak adlandırılmaktadır. Literatürdeki MOSR yöntemlerinde çoğunlukla NSGA II, MOEA, MOEA/D ve MOPSO algoritmaları kullanılmıştır. Bir MOSR metodu; MOOA, yerel arama metodu ve diz noktası seçim yöntemlerinden oluşmaktadır. Tez çalışmasının bu bölümünde bu kavramlarla ilgili genel bilgiler açıklandıktan sonra literatürdeki MOSR algoritmalarının tasarım ve performansları analiz edilip kullandıkları MOOA, yerel arama ve diz noktası seçim yöntemlerine göre karşılaştırılması verilmiştir [55]. Ayrıca, literatürdeki MOSR metotları ile ilgili bazı çıkarımlar ve yorumlar yapılmış ve önerilen yöntemlerin hesaplama karmaşıklıkları özetlenmiştir.

2.1. Çok Amaçlı Optimizasyon Algoritmaları

Çok amaçlı optimizasyon yöntemi, birden fazla amaç fonksiyonu olan bir optimizasyon problemine en uygun çözümleri bulma işlemidir. Çok amaçlı bir optimizasyon problemi Denklem (2.1)'deki gibi ifade edilebilir [56].

$$min\ F(a) = (f_1(a), f_2(a), \dots, f_n(a))^T,\ a \in \Omega \qquad (2.1)$$

Denklem (2.1)'de, Ω bir karar uzayıdır ve $a \in \Omega$ bir karar vektörüdür. $F(a)$ n tane amaç fonksiyonundan oluşur $f_i : \Omega \rightarrow R,\ i = 1, \dots, n$, R^n amaç uzayıdır. Çok amaçlı optimizasyon problemlerinin (Multi-Objective Optimization Problem, MOOP) genellikle çelişen amaç fonksiyonları vardır ve bu amaç fonksiyonları arasında bir takas çözümü söz konusudur.

Amaçlardan birinin gelişmesi diğer amacın gerilemesine neden olabilir. Amaç fonksiyonları çelişmiyorsa, MOOP tek bir optimizasyon problemine dönüştürülebilir [56].

Tanım 1: $u = (u_1, \dots, u_n)^T$ ve $v = (v_1, \dots, v_n)^T$ birer vektör olmak üzere, eğer $\forall i \in \{1, 2, \dots n\}$ için $u_i \leq v_i$ ve $u \neq v$ ise u vektörü v vektörünü domine etmektedir.

Tanım 2: Eğer $a' < a$ şartını sağlayan başka bir $a' \in \Omega$ yoksa, a çözümü Pareto optimum çözüm olarak adlandırılır. Pareto optimal çözümlerin kümesi Pareto optimal küme ve Pareto optimal kümenin arama uzayındaki görüntüsü ise Pareto cephesi (Pareto Front-PF) olarak tanımlanır.

MOOP'ların çözümü için 1950'lerden beri çeşitli yöntemler geliştirilmektedir. Literatürde bu problemin çözümü için çeşitli matematiksel programlama teknikleri mevcuttur [57, 58]. Bu teknikler MOOP'ların çözümünde kullanılırken bazı kısıtlamalara sahiptir. Genellikle bu yöntemler PF'nin şekline duyarlı olduğundan dolayı PF içbükey veya bağlantısız olduğu zaman doğru çalışmayabilirler. Ayrıca, her koşmada yalnızca tek bir çözüm ürettiğinden Pareto optimum çözüm kümesinin birkaç elemanını bulmak için bu yöntemlerin oldukça fazla sayıda koşulması gerekmektedir. Buna karşılık evrimsel algoritmalar tek bir çalışmada Pareto optimal kümesinin birden fazla üyesinin eşzamanlı bulunmasını sağlamaktadır. Ayrıca evrimsel algoritmalar PF'nin şekline veya sürekliliğine daha az duyarlıdırlar [59]. Literatürde MOOP'ları çözmek için önerilen algoritmalardan bazıları şunlardır:

Vektör Değerlendirmeli Genetik Algoritma [60], Çok Amaçlı Genetik Algoritma [61], Güç Pareto Evrimsel Algoritması [62], NSGA [63], NSGA II [64], NSGA III [65], MOEA/D [66], MOPSO [67], Arşivlenmiş Çok Amaçlı Benzetilmiş Tavlama [68].

Geniş bir kullanım alanına sahip olan MOOA'ların kullanıldığı alanlardan bazıları şunlardır: veri madenciliği [69], anten dizisi tasarımı [70], kablosuz sensör ağı [71], biyoinformatik [72], kontrol sistemleri ve robotik [73], görüntü işleme [74], desen tanıma [75] ve yapay sinir ağları [76].

Ayrıca son zamanlarda MOOA'lar CS'de SSR metodu olarak kullanılmaktadır. SSR probleminin her iki terimini de aynı anda optimize etmek ve düzenleme parametresinin

geri oluşturma performansına olan hassas etkisini azaltmak için CS'de seyrek sinyal geri elde edilmesinde MOOA'lar tercih edilebilir. Literatürde bu alanda diğer algoritmalar da kullanılmasına rağmen NSGA II, MOEA/D ve MOPSO gibi algoritmalar yaygın olarak kullanılmaktadır.

2.1.1. NSGA II

NSGA II, Siinivas ve Deb tarafından önerilen NSGA [63] algoritmasının geliştirilmesiyle, 2002 yılında Deb ve ark. tarafından ortaya konulmuştur [64]. Pareto optimal cephede daha geniş bir dağılım ve daha az karmaşıklığa sahip olan NSGA-II, hızlı ve elit prensibine dayanan sezgisel bir algoritmadır. NSGA II'de ilk olarak rastgele bir popülasyon oluşturulur. Daha sonra bu popülasyon domine edilemeyen ilkesine göre sıralanır. NSGA II'de her bir çözümü kaç tane çözümün domine ettiği ve o çözümün domine ettiği çözümler kümesini belirlemek gerekir. Her çözüm domine edilemeyen seviyesine eşit bir uygunluk değerine atanır ve *Npop* boyutunda bir yavru popülasyonu oluşturmak için kalabalık karşılaştırma işlemine dayalı ikili turnuva seçimi, rekombinasyon ve mutasyon operatörleri uygulanır. NSGA-II'nin temel adımları şöyledir:

Adım 1: Başlangıç popülasyonun oluşturulması.

Adım 2: Uygunluk değerlendirmesi.

Adım 3: Domine edilemeyen sıralama yöntemiyle popülasyonun sıralanması.

Adım 4: Yeni birey oluşturulması

Adım 4.1: Seleksiyon işleminin uygulanması

- Rastgele iki bireyden domine edilememiş sıralamaya göre iyi olanın seçilmesi.
- Eğer domine edilememiş sıralamaları aynı ise komşuluk yoğunluğuna göre seçim yapılması.

Adım 4.2: Çaprazlama işleminin uygulanması

Adım 4.3: Mutasyon işleminin uygulanması

Adım 5: Oluşan yeni bireylerin uygunluklarının değerlendirilmesi.

Adım 6: Eski ve yeni bireylerin birleştirilmesi.

Adım 7: Eski ve yeni bireylerin bulunduğu popülasyondan, en iyi N_{pop} bireyin seçilmesi. (1. kriter domine edilemeyen sıralama, 2. kriter ise komşuluk uzaklığı.)

Adım 8: Durdurma kriteri sağlanıyorsa dur. Sağlanmıyorsa Adım 4'e git.

2.1.2. MOEA/D

MOEA/D çok amaçlı bir optimizasyon problemini bir dizi alt optimizasyon problemlerine ayrıştırır ve bunları aynı anda optimize eder. Her alt problem yalnızca birkaç komşu alt problemden gelen bilgiler kullanılarak optimize edilir. MOEA/D Denklem (2.1)'deki MOOP problemini *Npop* adet optimizasyon alt problemlerine ayrıştırır ve bir çözüm popülasyonu geliştirerek bu alt problemleri aynı anda çözer. Her jenerasyonda popülasyon, her bir alt problem için algoritmanın çalıştırılmasının başlangıcından itibaren bulunan en iyi çözümden oluşur. Alt problemler arasındaki komşuluk ilişkileri, kümeleme katsayı vektörleri arasındaki uzaklıklara göre tanımlanır. İki komşu alt problemin optimal çözümleri çok benzer olmalıdır. MOEA/D'nin temel adımları şöyledir [66]:

Adım 1: Başlangıç

Adım 1.1: Dış popülasyonun (EP) boş kümeye eşitlenmesi.

Adım 1.2: Her bir alt problemin komşuluğunun hesaplanması.

Adım 1.3: Başlangıç popülasyonun üretilmesi.

Adım 1.4: Referans noktalarının başlatılması.

Adım 2: Güncelleme

Adım 2.1: Genetik operatörler kullanarak rastgele yeni çözümler üretilmesi.

Adım 2.2: Üretilen yeni çözümü geliştirmek için probleme özgü sezgisel bir geliştirme yöntemin uygulanması.

Adım 2.3: En iyi amaç değerinin güncellenmesi.

Adım 2.4: Komşu çözümlerin güncellenmesi.

Adım 2.5: EP kümesinin güncellenmesi.

Adım 3: Durdurma kriteri sağlanıyorsa dur ve EP'yi algoritmanın çıkışı olarak tanımla, sağlanmıyorsa Adım 2'ye git.

2.1.3. MOPSO

Belirli bir problemin uygunluk ortamını keşfetmek için bir parçacık sürüsü kullanmak parçacık tabanlı algoritmalardaki temel prensiplerden biridir. Her parçacık, aday çözümü tanımlayan bir vektördür ve eşdeğer olarak birkaç uygunluk fonksiyonu ile değerlendirilir. Algoritma yinelemelidir ve her yinelemede parçacıklar mevcut uygunluk değerlerinin yanı sıra yakındaki parçacıkların ve bir bütün olarak sürünün uygunluk değerlerine göre uygunluk ortamında hareket eder. Çok amaçlı problemlerin çözümü için PSO'da bazı değişiklikler yapılmalıdır. Çok amaçlı yöntemlerde amaç tek bir küresel en iyi çözümü değil Pareto cephesini içeren bir dizi çözüm bulmaktır. Bunu yapmak için, her yinelemede bulunan tüm domine edilemeyen çözümlerin depolandığı domine edilemeyen çözümlerin bir arşivi tutulur. MOPSO algoritmasında öncelikle sürü ve arşiv değerlerinin başlangıç değerleri atanır. Sürüdeki her bir parçacık için sırasıyla arşivden bir lider seçilir; hız ve pozisyon güncellenir. Son olarak da domine edilemeyen çözümlerin arşivi güncellenir ve bu adımlar durdurma kriteri sağlanıncaya kadar devam eder. MOPSO algoritmasının temel adımları şöyledir [77]:

Adım 1: Başlangıç popülasyonun oluşturulması (POP)

Adım 2: Her parçacık için başlangıç hızlarının oluşturulması.

Adım 3: Popülasyondaki parçacıkların değerlendirilmesi.

Adım 4: Depodaki (REP) domine edilemeyen vektörleri temsil eden parçacıkların pozisyonlarının kaydedilmesi.

Adım 5: Şimdiye kadar keşfedilen arama uzayına ait hiperküplerin oluşturulması ve bu hiperküpleri, her bir parçacığın koordinatlarının amaç fonksiyonlarının değerlerine göre tanımlandığı bir koordinat sistemi şeklinde kullanarak parçacıkların konumlandırılması.

Adım 6: Parçacık hafızalarının başlangıç değerlerinin atanması.

Adım 7: Ana Döngü

Adım 7.1: Parçacık hızlarının hesaplanması

Adım 7.2: Önceki adımda hesaplanan hızlar da eklenerek parçacıkların yeni konumlarının hesaplanması

Adım 7.3: Parçacıkların araştırma uzayı içinde tutulması

Adım 7.4: Popülasyondaki parçacıkların değerlendirilmesi

Adım 7.5: REP içeriğinin güncellenmesi

Adım 7.6: Eğer parçacığın mevcut konumu hafızasındaki konumundan daha iyi ise parçacığın konumunun güncellenmesi

Adım 7.7: Maksimum yineleme sayısına ulaşılmadıysa Adım 7.1'e git.

2.2. Yerel arama metodu

Literatürdeki MOSR yöntemleri incelendiğinde, MOOA'ların SSR problemlerinin seyreklik ve ölçüm hatası terimlerini aynı anda optimize etmeye çalıştığı görülmektedir. MOOA'lar seyreklik seviyesinin optimizasyonunda başarılı olmasına rağmen, geri oluşturma performansları klasik yöntemler kadar başarılı değildir. Ayrıca, büyük gürültü seviyelerine sahip seyrek problemler için verimli değildirler [78]. SSR problemlerinin optimizasyonunda mevcut MOOA'ların yakınsama hızı düşüktür. Bu nedenle literatürdeki çoğu MOSR yöntemi bir yerel arama yöntemi kullanmaktadır. MOSR metotlarının çözüme yakınsama hızı uygun ve etkili bir yerel arama yöntemi kullanılarak geliştirilebilir. Literatürde yer alan MOSR yöntemlerinde çeşitli yerel arama yöntemleri kullanılmıştır. Yaygın olarak kullanılan yerel arama yöntemleri yinelemeli eşikleme algoritmalarıdır. Gradyan inişi, düzenleme parametresinin ayarlanması ve eşik kesme işlemleri eşikleme yöntemlerinin temel üç adımlarındandır [79]. IST [80], IHardT [81] ve IHalfT [52] algoritmaları yaygın olarak kullanılan yinelemeli eşikleme yöntemlerindendir. Bu yöntemler son iki adımda birbirinden farklıdır. Adımlar aşağıdaki gibi açıklanmıştır [79].

Adım 1: Gradyan inişi: $\boldsymbol{x}^{(n)}$ vektörü güncel bir çözüm olsun, deneme çözümü olan $\widetilde{\boldsymbol{x}} \in R^N$ Denklem (2.2)'deki gibi hesaplanmaktadır.

$$\widetilde{\boldsymbol{x}} = \boldsymbol{x}^{(n)} + \mu \times \boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{x}^{(n)} - \boldsymbol{y}) \tag{2.2}$$

Adım 2: Düzenleme parametresinin ayarlanması: IST metodu için düzenleme parametresi ya deneysel bir sabit değer olarak alınabilir ya da seyreklik seviyesinin bir önceki tahmininden yararlanılarak elde edilebilir. IHardT ve IHalfT metotları için düzenleme parametresi sırasıyla $|\widetilde{\boldsymbol{x}}_{i_{k+1}}|$ ve $\frac{\sqrt{96}}{9}|\widetilde{\boldsymbol{x}}_{i_{k+1}}|^{3/2}$ dir.

Adım 3: Eşik kesme: Seyrek sinyali elde edebilmek için sinyalin bazı bileşenleri aşağıdaki gibi sıfıra eşitlenir.

a) Sert Eşik:

$$\boldsymbol{x}_i^{(n+1)} = \begin{cases} \widetilde{\boldsymbol{x}}_i & |\widetilde{\boldsymbol{x}}_i| > \lambda_n \\ 0 & diğer \end{cases} \tag{2.3}$$

b) Yumuşak Eşik:

$$\boldsymbol{x}_i^{(n+1)} = \begin{cases} \boldsymbol{\psi}_1(\widetilde{\boldsymbol{x}}_i, \lambda_n) & |\widetilde{\boldsymbol{x}}_i| > 0.5\lambda_n \\ 0 & diğer \end{cases} \tag{2.4}$$

$$\boldsymbol{\psi}_1(\widetilde{\boldsymbol{x}}_i, \lambda_n) = sgn(\widetilde{\boldsymbol{x}}_i)(\widetilde{\boldsymbol{x}}_i - 0.5\lambda_n) \tag{2.5}$$

c) Yarım Eşik:

$$\boldsymbol{x}_i^{(n+1)} = \begin{cases} \boldsymbol{\psi}_2(\widetilde{\boldsymbol{x}}_i) & |\widetilde{\boldsymbol{x}}_i| > \frac{\sqrt[3]{54}(\lambda_n)^{2/3}}{4} \\ 0 & diğer \end{cases} \tag{2.6}$$

$$\boldsymbol{\psi}_2(\widetilde{x}_i) = \left(\frac{2}{3}\right)\widetilde{\boldsymbol{x}}_i\left(1 + cos\left(\left(\frac{2\pi}{3}\right) - \left(\frac{2\phi(\widetilde{\boldsymbol{x}}_i)}{3}\right)\right)\right) \tag{2.7}$$

$$\phi(\widetilde{\boldsymbol{x}}_i) = \arccos\left(\left(\frac{\lambda_n}{8}\right)\left(\frac{|\widetilde{\boldsymbol{x}}_i|}{3}\right)^{-\frac{3}{2}}\right) \tag{2.8}$$

IST, IHardT ve IHalfT algoritmalarının Xu ve ark. tarafından yapılan bir çalışmada karşılaştırmaları mevcuttur [52]. Belirtilen çalışmada karşılaştırma parametresi olarak

geri elde edilen sinyal ile orijinal sinyal arasındaki MSE ve algoritmanın çalışma süresi kullanılmıştır. Bu algoritmaları karşılaştırmak için N=512 uzunluğunda, k=130 seyrekliğine sahip bir seyrek sinyal ve rastgele Gauss dağılımına sahip bir ölçüm matrisi kullanılmıştır. Elde edilen sonuçlara göre, ölçüm sayısı azaltıldığında, IHalfT algoritmasının geri oluşturma doğruluğu açısından diğer karşılaştırılan algoritmalardan daha iyi olduğu sonucuna varılmıştır. Örneğin, ölçüm sayısı M=240 olduğunda, seyrek sinyali yalnızca IHalfT algoritması geri elde edebilmiştir. Ölçüm matrisinde gürültü söz konusu olduğunda, IHalfT algoritması aynı gürültü seviyesinde en iyi sinyal geri oluşturma performansına sahiptir. Ayrıca en az sayıda ölçüme ihtiyaç duyan metot IHalfT'tir. Bu durum, IHalfT algoritmasının gürültüye karşı daha kararlı olduğunu göstermektedir. Tablo 2.1'de, literatürde yer alan MOSR metotlarında yaygın olarak kullanılan yerel arama yöntemleri özetlenmektedir.

Tablo 2.1. MOSR yöntemlerinde kullanılan bazı yerel arama yöntemlerinin özeti.

Yerel Arama Metodu			
	IHardT	**IST**	**IHalfT**
Minimizasyon Yöntemi	L_0	L_1	$L_{1/2}$
Eşik Değeri	λ_n	$0.5\lambda_n$	$\frac{\sqrt[3]{54}\lambda_n^{3/2}}{4}$
Geri Oluşturma Performansı	En iyi geri oluşturma süresine ve en düşük geri oluşturma doğruluğuna sahiptir.	Geri oluşturma hatası konusunda IHalfT'den daha kötü ve geri oluşturma süresi konusunda IHalfT'den daha iyidir.	En yüksek geri oluşturma doğruluğuna ve en yüksek geri oluşturma süresine sahiptir.
Literatürde Kullanıldığı MOSR çalışmaları	Li ve ark. [84] Li ve ark. [79]	Li ve ark. [54] Yan ve ark. [85] Li ve ark. [79]	Li ve ark. [84] Li ve ark. [78] Li ve ark. [79] Liang ve ark. [86] Li ve ark. [87] Yue ve ark. [88]

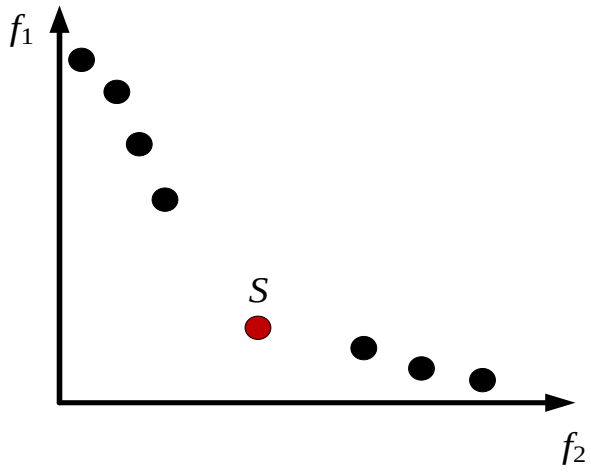
Bu yerel arama yöntemlerine ek olarak, bir çalışmada Doğrusallaştırılmış Bregman (Linearized Bregman, LB) yöntemi kullanılmaktadır [82]. LB yöntemi belirtilen çalışmada kullanılırken bir parametre matrisi tanımlanır. Bu matris, yerel aramada

kullanılan bireyler için (0,1) aralığında düzgün dağılım sağlar ve düzenleme parametresinin önceden tanımlanması gerekliliğini ortadan kaldırır.

Zhou ve ark. tarafından yapılan bir çalışmada farklı bir yerel arama yöntemi kullanılmıştır [83]. Belirtilen çalışmadaki yerel arama yöntemi daha iyi bir giriş bulduğu zaman sıfır olmayan giriş kümesinin seçilen girişini değiştirir. Dolayısıyla sıfırdan farklı uygun girdilerin bulunmasını destekler ve yineleme sayısını azaltmaya yardımcı olur. Bu yöntemin performansı, ilk çözüme ve komşuluk yapısına bağlıdır. Komşuluk yapısı, *k*-ortalama kümelemesi kullanılarak oluşturulmuştur.

2.3 Diz noktası seçme yöntemleri

Tek amaçlı optimizasyon yöntemlerinin aksine çok amaçlı optimizasyon yöntemleri bir dizi çözüm bulmaya çalışır. Çözümler kümesinden nihai bir çözüm bulmak için bir karar vericinin amaç fonksiyonlarının önemine göre son çözümü seçmesi gerekmektedir. Diz bölgeleri, bir amaçtaki küçük bir iyileşmenin diğerinde ciddi bir bozulmaya neden olacağı bölgelerdir. Diz noktası Şekil 2.1'de gösterilmektedir. Şekilde S noktası diz noktasıdır ve diz noktasının olduğu bölge genellikle PF üzerinde çözüm anlamında en önemli bölgedir. Karar vericinin özel bir tercihi yoksa diz noktası PF'de ilk tercih edilen çözümdür.



Şekil 2.1. Diz noktası içeren bir PF

MOSR yöntemleri çeşitli optimum ölçüm hatası ve seyreklik değerleri kullanarak bir dizi takas çözümü elde eder. Bu takas çözümlerinin içinden nihai bir çözüm seçmek gerekmektedir. Kullanıcı tercihleri belirtilmediğinde nihai çözüm olarak diz noktası

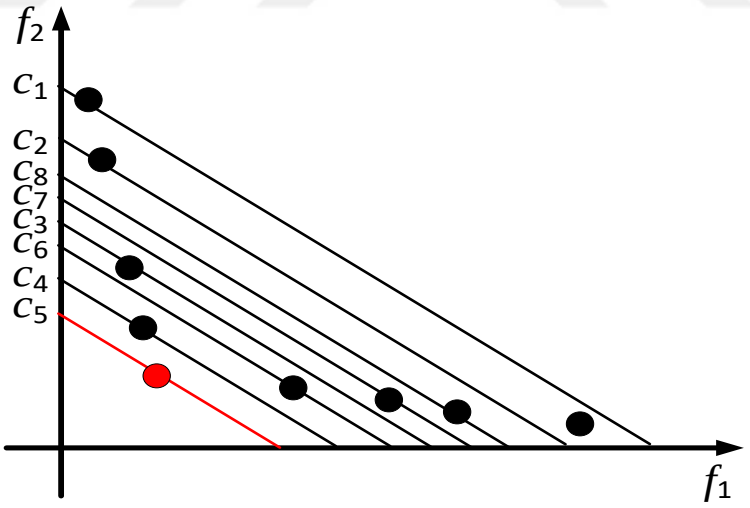
tercih edilmektedir [89]. MOSR yöntemlerinde nihai çözümün elde edilmesinde, amaç değerlerinin ağırlıklı toplamına dayalı yöntem [78], açı tabanlı yöntem [90] ve Kink yöntemi [91] yaygın olarak kullanılmaktadır.

2.3.1. Ağırlıklı toplam yöntemi

Ağırlıklı toplam yöntemi, Denklem (2.9)'da doğrunun kesişmesine göre diz bölgesini elde eder.

$$f_2 = mf_1 + c \tag{2.9}$$

Denklem (2.9)'da f_1 birinci amaç fonksiyonu yani seyrekliği, f_2 ise ikinci amaç fonksiyonu olan ölçüm hatasını ifade eder. Denklem (2.9)'da m eğimi, c ise kesişimi temsil etmektedir. Bu yöntemde minimum kesişime sahip çözüm diz noktası olarak tanımlanır. Ağırlıklı toplam yönteminin temel prensibi Şekil 2.2'de gösterilmektedir. Şekil 2.2'de c_1-c_8 noktaları incelendiğinde 5. noktanın kesişimi minimum olduğu görülmektedir. Bu nedenle 5. nokta diz noktası olarak tanımlanmaktadır [92].



Şekil 2.2. Amaç değerlerinin ağırlıklı toplamı.

Ağırlıklı toplam yöntemi, Li ve ark. tarafından yapılan bir çalışmada kullanılmıştır [78]. Belirtilen çalışmada, Denklem (2.10)'daki ölçüm hatası ve seyrekliğin ağırlıklı toplamının değerleri hesaplanarak diz noktası yaklaşık olarak bulunmaktadır.

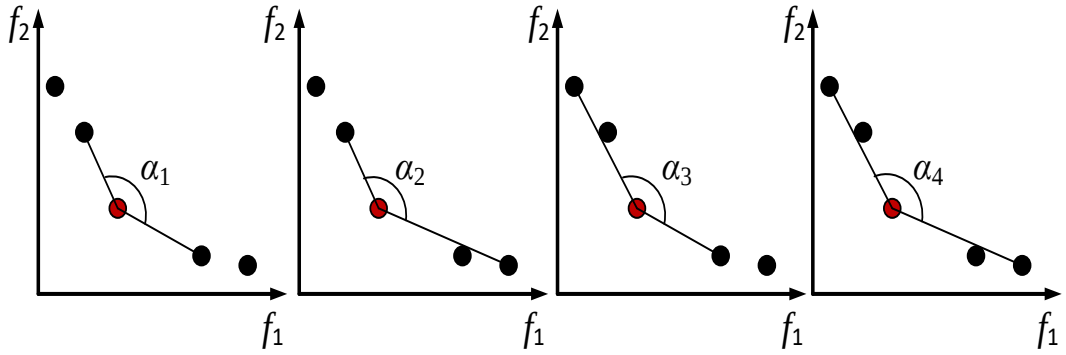
$$||\boldsymbol{y} - \boldsymbol{Ax}||_2^2 + \alpha||\boldsymbol{x}||_0 \tag{2.10}$$

Ağırlıklı parametre α çok küçük bir sayıdır ve gerçek seyreklik bulgusu α'ya oldukça duyarlıdır. Ayrıca bu yöntem Liang ve ark. tarafından yapılan bir çalışmada kullanılmıştır [86].

2.3.2. Açıya dayalı yöntem

Açıya dayalı yöntemde mevcut çözümü PF üzerindeki en yakın komşularına bağlamak için düz çizgiler çizilir. Çizgilerin açıları, her bir amaç fonksiyonundaki ödünleşimleri sembolize eder. Her iki yöndeki en yakın iki komşu dikkate alınır ve Şekil 2.3'teki gibi dört açı (α1∼α4) belirlenir. Bunların arasındaki en küçük açı, o çözümün ödünleşimi olarak kabul edilir. PF üzerindeki her noktanın açıları karşılaştırılarak en küçük takas açısına sahip çözüm son çözüm olarak kabul edilir [90].

Açı tabanlı yöntem, MOSR metotlarında ilk olarak yumuşak eşiklemeli evrimsel çok amaçlı algoritmada (Soft Thresholding Evolutionary Multiobjective Algorithm, StEMO) kullanılmıştır [54]. Araştırmaya göre, MOSR problemlerinde diz bölgesi bulma konusunda dikkat edilmesi gereken bazı hususlar vardır. Bunların birincisi algoritma tarafından üretilen tahmini PF'nin gerçek PF'ye yakınsadığından emin olmak zordur.



Şekil 2.3. Açı tabanlı yöntem.

Bir popülasyonun çözümlerinden bazıları gerçekten PF'de yer almıyorsa, diz bölgesini bulmak için kullanılan hesaplamaları bozabilir ve küresel olarak optimal çözümün yanlış bir tahminine yol açabilir. Bir diğer husus PF üzerinde yer alan amaç fonksiyonlarının büyüklükleri birbirinden çok farklı aralıklarda olabilir. Bu durumda, karşılaştırılan açıların arasındaki farkların önemi kalmayabilir. Bahsedilen çalışmada bu sorunları ortadan kaldırmak için açı tabanlı yöntem uygulanmadan önce iki amaç fonksiyonu

normalize edilmiştir ve PF'nin mevcut olmayan elemanları için ara değerleme yapılmıştır.

Yue ve ark. tarafından yapılan bir çalışmada gelişmiş açı tabanlı bir yöntem kullanılmıştır [88]. Belirtilen çalışmaya göre, sıradan açı tabanlı yöntem gürültüye karşı dayanıklı değildir. Gürültüye karşı daha iyi bir metot geliştirmek için maksimum ödünleşimlere sahip olan belirli bir sayıda çözüm seçilmiştir. Daha sonra, seçilen bu çözümler arasından ölçüm hatası en az olan çözüm son çözüm olarak belirlenmiştir.

2.3.3. Kink Yöntemi

Kink yönteminde en büyük eğim değişimine sahip olan çözüm nihai çözüm olarak kabul edilir. Amaç fonksiyonları $\left(f_1(x_i), f_2(x_i)\right)$ olan bir x_i çözümünün eğim değişimi $\Delta\beta_i$, Denklem (2.11) ile hesaplanmaktadır.

$$\begin{cases} \beta_0 = \frac{2}{\pi} \\ \beta_i = arctan\left(\frac{f_1(x_i)-f_1(x_{i+1})}{f_2(x_i)-f_2(x_{i+1})}\right) \\ \Delta\beta_i = \frac{\beta_i}{\beta_{i-1}} \end{cases} \tag{2.11}$$

Denklem (2.11)'deki en büyük eğim değişimi değerine sahip olan çözüm optimal çözüm olarak tanımlanır [85, 91].

Yan ve ark. tarafından yapılan çalışmalarda domine edilemeyen çözümler oluşturulup normalleştirme işlemi yapıldıktan sonra etkili bir nihai çözüm elde edilmesinde diz bölgesini bulmak için Kink yöntemi kullanılmıştır [82, 85].

Bunların dışında MOSR problemlerinde nihai bir çözüm elde etmek için kullanılan başka yöntemler de vardır. Bir çalışmada ölçüm hatasının önceden tanımlanmış olan alt ve üst sınır aralığında bulunan çözümler tercih edilen çözümler olarak seçilmektedir [83]. Seyrek sinyalin sıfır olmayan girişleri elde edildikten sonra en küçük kareler yöntemi kullanılarak nihai çözüm elde edilmiştir.

Diğer bir çalışmada önerilen Seyrek Tercih Tabanlı Yerel Arama (Sparse Preference based Local Search-SPLS) adlı MOSR metodunda gerçek *k*-seyrek çözüm üzerinde tercihe dayalı yatay eşik çizgisi yöntemi kullanılmıştır [79]. SPLS, popülasyonu bir parametre ve amaç fonksiyonlarına bağlı yatay bir eşik çizgisi aracılığıyla iki alt kümeye

böler. Doğrusal kısım yatay eksene paralel olduğunda, SPLS gerçek seyreklik seviyesine yakın diz bölgesini tespit edebilmektedir [87].

Li ve diğerleri tarafından yapılan bir çalışmada Gauss Karışım Modellerine (Gaussian Mixture Model, GMM) dayalı bir nihai çözüm bulma yöntemi kullanılmıştır [87]. PF üzerinden seyrek çözümleri bulabilmek için iki fazlı yinelemeli bir arama prosedürü önerilmiştir. İlk aşamada yaklaşık olarak PF'nin tamamı zincir tabanlı bir arama yöntemi aracılığıyla elde edilir. İkinci aşamada ise rastgele bir şekilde tercihe dayalı olarak diz bölgesi değerlendirilir. Belirtilen iki faz arasında tüm PF'nin yaklaşımını iki alt kümeye bölen Gauss Karışım Modelleri aracılığıyla kümelenerek diz bölgesi tespit edilir. Belirtilen çalışmaya göre GMM, MOSR için PF'nin doğrusal bir dağılımının elde edilmesinde gürültü söz konusu olduğunda SPLS'deki yöntemden daha üstündür. Ayrıca, SPLS'nin aksine bu yöntem seyreklik tespiti için herhangi bir hassas parametreye bağlı değildir [87].

2.4. Literatürdeki MOSR Yöntemlerinin İncelenmesi

Tez çalışmasının bu bölümünde literatürdeki her bir MOSR algoritması ayrı bir başlık altında özetlenmiştir. Aşağıdaki alt başlıklar özetlenen çalışmaların başlıklarıdır. Bu bölümde MOSR metotları incelenirken bu yöntemlerde yer alan MOOA, yerel arama yöntemleri, diz noktası seçim yöntemlerinin nasıl kullanıldıkları üzerinde durulmuştur. Özellikle MOSR algoritmalarının performans farklılıkları bu tasarım özelliklerinden kaynaklanmaktadır. Bu algoritmaları analiz ederek daha verimli MOSR yöntemleri tasarlamak mümkündür. Ayrıca bu bölümde literatürde var olan MOSR algoritmalarının test edildiği sinyal türleri belirtilmiş ve bazı performans karşılaştırmalarından ve değerlendirmelerinden bahsedilmiştir.

2.4.1. Seyrek Optimizasyon Problemleri için MOEA/D ile Beraber Yinelemeli Eşikleme Algoritması

Li ve ark. tarafından 2012 yılında SSR problemi için çok amaçlı optimizasyon algoritmalarının kullanılması önerilmiştir [84]. Önerilen yöntem MOEA/D'yi kullanarak Denklem (1.10)'daki SSR problemini optimize etmektedir. Ayrıca bu yöntem, yerel arama yöntemi olarak yinelemeli eşikleme yöntemi kullanmaktadır. Belirtilen yöntemde,

yerel arama yöntemi olarak sert eşikleme metodu kullanıldığında MOEA/D-IHardT, yarım eşikleme metodu kullanıldığında ise MOEA/D-IHalfT olarak adlandırılmıştır.

Önerilen metodun performansı, klasik yöntem olan IHardT ve IHalfT algoritmaları ile karşılaştırılmıştır. Karşılaştırmada kullanılan test sinyalleri yapay olarak üretilmiştir. MOEA/D-iHardT ve MOEA/D-iHalfT'nin geri oluşturma performansları, orijinal test sinyalleri ve geri oluşturulma işleminin MSE değerleri kullanılarak incelenmiştir. Belirtilen çalışmadaki elde edilen sonuçlara göre, MOEA/D-iHardT ve MOEA/D-iHalfT önceden seyreklik bilgisine gerek olmaksızın seyrek geri oluşturma işlemi için etkili yöntemlerdir. Ayrıca MOEA/D-iHalfT yönteminin MOEA/D-iHardT yöntemine göre daha iyi geri oluşturma performansına sahip olduğu belirtilmiştir.

2.4.2. SSR için Evrimsel Çok Amaçlı Yaklaşım

Li ve ark. 2014 yılında, NSGA II kullanarak StEMO adlı bir MOSR yöntemi önermişlerdir [54]. Önerilen yöntem Denklem (1.10)'da verilen optimizasyon problemini PF'nin tamamını bularak optimize etmektedir. StEMO yöntemi, PF'nin diz noktasını bulmak için açı tabanlı olan yöntemi ve yerel arama yöntemi olarak yumuşak eşikleme yöntemini kullanmaktadır. Belirtilen çalışmaya göre önerilen yöntemin ana katkısı, seyrek geri oluşturma probleminin çözümü açısından çok amaçlı bir yöntemin tek amaçlı yönteme göre daha üstün olabileceğini göstermektir.

StEMO metodunun SSR performansı; OMP, BP [93], Homotopy [94] ve IST algoritmaları gibi yaygın olarak kullanılan klasik yöntemlerle karşılaştırılmıştır. Belirtilen çalışmada test sinyali olarak; yapay sinyaller, SPARCO araç kutusu ve çeşitli görüntü sinyalleri kullanılmıştır. Önerilen metodun etkinliğini göstermek için yapılan deneylerin ilk örneği olarak, N=2000 uzunluğunda yapay bir sinyal üretilmiştir. Ölçüm sayısı M=800-1800 aralığında değiştirilerek yöntemlerin performansları analiz edilmiştir. Bu test sonucuna göre, OMP'nin 1200-1600 ölçüm aralığında en düşük geri oluşum hatasına sahip olması dışında, StEMO diğer karşılaştırılan algoritmalara göre daha düşük geri oluşum hatasına sahiptir. Ayrıca algoritmalar, seyreklik oranı 0.1-0.5 aralığında değiştirilerek karşılaştırılmışlardır. Bu karşılaştırmanın sonucuna göre StEMO en düşük geri oluşum hatası değerine sahiptir [54]. Ayrıca, StEMO yönteminin diğer yöntemlerle karşılaştırılmasında SPARCO yazılımı kullanılmıştır. Bu karşılaştırmada yukarıda bahsedilen yöntemlerin geri oluşturma hataları sunulmuştur. Sonuç olarak StEMO,

karşılaştırılan algoritmaların çoğundan daha iyi performansa sahiptir. StEMO algoritmasının SSR'de MOSR yöntemini kullanan ilk metotlardan biri olması nedeniyle, diğer MOSR algoritmaları ile StEMO algoritmasının karşılaştırılması çalışmada mevcut değildir. Ancak, yeni bir MOSR yöntemi öneren diğer makaleler çalışmalarını genelde StEMO algoritması ile karşılaştırmışlardır.

2.4.3. SSR için Ayrıştırmaya Dayalı Çok Aşamalı ve Çok Amaçlı bir Yaklaşım

Li ve ark. MOEA/D-L1/2 adlı bir MOSR yöntemi önermiştir [78]. Önerilen yöntem, SSR problemini çözmek için MOEA/D algoritmasını ve yerel arama yöntemi olarak yinelemeli yarım eşikleme yöntemini kullanmaktadır. MOEA/D-L1/2, PF'den nihai çözümü elde etmek için ağırlıklı toplam yöntemini kullanmaktadır.

Belirtilen çalışmada yazarlar MOEA/D-L1/2'nin geri oluşturma performansını MOEA/D-old (MOEA/D'nin eski versiyonu) ve yarım eşikleme algoritmaları ile karşılaştırmıştır. Bu üç algoritmanın geri oluşturma performanslarını karşılaştırmak için geri oluşturulmuş sinyal ile gerçek seyrek sinyal arasındaki MSE değerleri kullanılmıştır. Bu karşılaştırmada, geri oluşturma performansını incelemek için yapay test sinyalleri kullanılmıştır. Test örneklerinin konfigürasyonu bahsedilen çalışmada gösterilmektedir [78]. Karşılaştırılan algoritmalar, makalede belirtilen P1-P6 problemleri üzerinde 100 kez çalıştırılmıştır. Yeniden oluşturulan sinyal ile gerçek sinyal arasındaki MSE 10^{-6}'nın altındaysa, seyrek bir sinyalin geri oluşturulması bahsedilen makalede başarılı olarak kabul edilmektedir. Bu test sinyalleri ile yapılan deneylere göre MOEA/D-L1/2 başarı oranı açısından MOEA/D-old ve yarım eşikleme algoritmalarından daha iyidir. Belirtilen çalışmaya göre MOEA/D-L1/2, daha az sayıda gözlemle büyük boyutlu sinyali geri oluşturma ve gürültülü ölçümlerden seyrek sinyali geri oluşturma konularında avantajlıdır.

2.4.4. SSR için İki Aşamalı Evrimsel bir Yaklaşım

Zhou ve ark. iki aşamalı evrimsel yaklaşıma dayalı bir MOSR yöntemi önermiştir [83]. Birinci aşamada, Denklem (2.12)'yi optimize ederek bir grup çözüm üretmek için MOEA/D kullanılmıştır.

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) = \min_{\boldsymbol{x}} \left(||\boldsymbol{x}||_1,\ ||\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}||_2^2\right) \tag{2.12}$$

Gürültünün etkisini ortadan kaldırmak için çözümlere öznitelik çıkarma yöntemi uygulanmış ve *k*-ortalamalar kümeleme yöntemi kullanılarak sıfırdan farklı bir başlangıç kümesi oluşturulmuştur. İkinci aşamada, sıfır olmayan girişlerin konumlarını geliştirmek için ileriye dayalı bir seçim yöntemi uygulanmıştır. Önerilen yöntem, sinyalin sıfır olmayan girişlerini bulup ardından en küçük kareler yöntemini kullanarak seyrek sinyali geri oluşturmaktadır. Yerel arama yöntemi olarak olasılığa dayalı açgözlü, rastgele uyarlanabilir bir arama prosedürü kullanılan bu yöntem MOEA/D-PGRASP olarak adlandırılmaktadır.

Belirtilen çalışmada, MOEA/D-PGRASP algoritmasının performansının karşılaştırıldığı yöntemler OMP, düzleştirilmiş L_0 [95], Elastic-net (EN) [96], M+GRASP ve MOEA/D-best metotlarıdır. Önerilen yöntemin geri oluşturma performansını değerlendirmek için yapay olarak üretilen sinyaller kullanılmıştır. Yapılan çalışmadan elde edilen sonuçlara göre, önerilen yöntemin karşılaştırılan algoritmalardan daha iyi olduğu sonucuna varılmıştır. Örnekleme oranının 0.2 olduğu durum göz önüne alındığında, iki boyutlu sinyalin geri oluşturulmasında MOEA/D-PGRASP hem sayısal hem de görsel olarak iyi sonuçlara sahiptir.

2.4.5. MOSR için Evrimsel Hibrit Algoritma

Yan ve ark., MOSR için melez bir evrimsel yöntem önermiştir [82]. Bu yönteme Doğrusal Bregman Evrimsel Algoritması (Linear Bregman Evolutionary Algorithm-LBEA) denilmektedir. MOOA olarak Diferansiyel Gelişim algoritması kullanılmıştır. Önerilen yöntemin yerel aramasını gerçekleştirmek için LB tekniği uygulanmıştır. Nihai çözüm Kink yöntemiyle elde edilmiştir.

Yöntemin geri oluşturma performansını değerlendirmek için görüntü sinyalleri ve yapay test sinyalleri kullanılmıştır. OMP, Hızlı Yinelemeli Büzülme Eşiği Algoritması (Fast Iterative Shrinkage-Thresholding Algorithm, FISTA) [97], Homotopy ve StEMO yöntemleri LBEA ile karşılaştırılmıştır. Seyreklik seviyesi, ölçüm sayısı ve gürültünün StEMO ve LBEA'nın geri oluşum performansı üzerindeki etkileri analiz edilmiştir. Ayrıca algoritmalar, görüntü sinyalinin geri oluşturulması problemi üzerinde test edilmiştir. Elde edilen sayısal sonuçlara göre LBEA, gürültüye karşı gürbüzlük ve yakınsama açısından klasik algoritmalardan daha iyidir. Ayrıca LBEA, çeşitlilik,

yakınsama ve gürbüzlük açısından karşılaştırılan MOSR algoritmalarına göre belirli bir üstünlüğe sahiptir.

2.4.6. MOSR için Uyarlanabilir Ayrıştırma Tabanlı Evrimsel Yaklaşım

Yan ver ark. uyarlanabilir ayrıştırma tabanlı evrimsel yaklaşımı (Adaptive Decomposition-based Evolutionary Approach, ADEA) kullanan bir MOSR yöntemi tasarlamışlardır [85]. ADEA, yerel arama olarak iki aşamalı yinelemeli yumuşak eşik yöntemini kullanmaktadır. Bu çalışmada PF üzerindeki diz noktasını bulmak için Kink yöntemi uygulanmıştır. Belirtilen çalışmada ayrıca, uyarlamalı ayrıştırmaya dayalı bir çevresel seçim yaklaşımı tasarlanmıştır. Bu yaklaşımın, diz bölgesinden bir çözümün seçilme şansını artırdığı ve algoritma karmaşıklığını azalttığı belirtilmiştir.

Çalışmada ADEA yöntemi ile karşılaştırılan algoritmalar; OMP, FISTA, Homotopy, BP, LBEA ve StEMO'dur. Çalışmadaki yöntemleri karşılaştırmak için SPARCO araç kutusu, yapay olarak oluşturulmuş sinyaller ve görüntü sinyalleri kullanılmıştır. MOSR yöntemlerinden olan StEMO, LBEA ve ADEA'nın çalışma süresi, geri oluşum hataları, gürültü seviyeleri, değişken seyreklik ve sinyal uzunlukları gibi farklı test koşulları altında karşılaştırılmıştır. Elde edilen sonuçlar ADEA'nın farklı test durumları için SSR performansı açısından StEMO ve LBEA'ya göre belirli bir üstünlüğe sahip olduğunu göstermektedir. ADEA metodu StEMO ve LBEA'dan daha küçük koşma süresine sahiptir. Ancak, klasik yöntemlerin ADEA'dan ve diğer karşılaştırılan MOSR yöntemlerinden daha hızlı olduğu çalışmada belirtilmiştir.

2.4.7. Seyrek Optimizasyon için Tercihe Dayalı bir MOEA Yaklaşımı

Li ve ark. SSR için, SPLS adı verilen tercihe dayalı bir yöntem önermişlerdir [79]. Önerilen yöntem, MOOA olarak MOEA/D algoritmasını kullanmaktadır ve Denklem (1.10)'da verilen problemi optimize etmektedir. Bu yöntemde, yerel arama için yumuşak, yarım ve sert eşikleme yöntemlerinin hepsinin kullanılabileceği belirtilmiştir. Yerel arama metodunda algoritmanın her bir iterasyonu için başlangıç çözümünün seçiminde sadece diz noktasına yakın bir çözüm tercih edilmektedir. Bu durum kademeli olarak seyrek sinyal elde etme ihtimalini artırmaktadır.

SPLS'nin SSR performansını değerlendirmek için yapay sinyaller ve MRI görüntü sinyalleri kullanılmıştır. Ayrıca yazarlar SPLS'yi IHT, OMP, StEMO ve MOEA/D

algoritmaları ile karşılaştırmışlardır. Elde edilen sonuçlara göre SPLS'nin geri oluşum başarı oranı açısından OMP, StEMO ve MOEA/D'ye göre bazı üstünlüklerinin olduğu söylenebilir. Yazarlara göre SPLS, sinyalin uzunluğunun yüksek veya ölçüm sayısının düşük olduğu durumlarda makalede karşılaştırılan diğer algoritmalardan daha üstün bir SSR performansına sahiptir. SPLS, diz noktasının bulunması sürecinde eşikleme algoritmalarına kıyasla daha az hesaplama süresi harcar. Ayrıca SPLS algoritmasının performansı dalgacık dönüşümüyle seyrek olarak ifade edilen MRI görüntü sinyalleri üzerinde test edilmiştir. Test edilen MRI görüntü sinyallerindeki büyük bileşenlerin çoğu, SPLS algoritması tarafından başarılı bir şekilde geri oluşturulmuştur.

2.4.8. SSR için GMC'ye Dayalı Çok Amaçlı bir Yaklaşım

Li ve ark. Gauss karışım kümelenmesine (Gaussian Mixture Clustering, GMC) dayalı bir MOSR yöntemi önermiştir [87]. Önerilen yöntemde Gauss karışım modelleri kullanılarak nihai çözüm elde edilmiştir. MOEA/D-GMC metodundaki yerel arama prosedürü iki aşamalı bir yinelemeli arama yöntemine dayanmaktadır. MOEA/D-L1/2 ve SPLS/HALF gibi diğer MOSR algoritmalarının aksine, bu yöntemde seyreklik tespiti için ekstra bir parametre gerekli değildir. SPLS metodunda kontrol parametresinin hassasiyeti ve ölçümlerdeki gürültü seviyesi seyreklik seviyesinin yanlış tespit edilmesine neden olabilir. Seyreklik tespiti konusundaki güvenilmezliği ortadan kaldırmak için Gauss karışım modeli tabanlı yöntem önerilmiştir. Belirtilen çalışmada elde edilen sonuçlara göre, yöntemin etkinliği gürültülü durumlarda görülmektedir.

Önerilen yöntem, yapay test sinyalleri kullanılarak MOEA/D-L1/2, SPLS ve HALF yöntemi ile karşılaştırılmıştır. Bu karşılaştırmalarda performans parametresi olarak, karşılaştırılan algoritmaların geri oluşturmalarına ait MSE ve seyreklik hataları incelenmiştir. Performans parametrelerinin ayrıntılı bir karşılaştırması bahsedilen çalışmada yer almaktadır [87].

2.4.9. Seyrek optimizasyon için MOBSO Algoritması

Liang ve ark. Seyrek Optimizasyon için Çok Amaçlı Beyin Fırtınası Optimizasyon Algoritmasına (Multi-objective Brainstorm Optimization Algorithm for Sparse Optimization, MOBSO-OS) dayalı bir MOSR tekniği önermiştir [86]. MOBSO-OS metodu, yerel arama yöntemi olarak IHalfT operatörünü kullanır. Önerilen bu MOSR

yönteminde PF üzerindeki diz bölgesini bulmak için ağırlıklı toplam yöntemi uygulanmıştır.

MOBSO-OS yöntemi; Homotopy, MP, OMP ve KnMOPSO algoritmaları ile karşılaştırılmıştır. MOBSO-OS algoritmasını test etmek için yapay test sinyalleri kullanılmıştır. Makalede MOBSO-OS'un etkinliği, seyrek sinyalin geri oluşturma işleminin düşük hata ve düşük çalışma süresiyle kanıtlandığı belirtilmiştir. Makaledeki elde edilen sonuçlar MOBSO-OS'un aynı test koşulları altında MP, OMP, Homotopy ve KnMOPSO'dan daha iyi SSR performansına sahip olduğunu göstermektedir. Ancak, algoritmanın çalışma süresi bakımından MOBSO-OS'un bazı klasik yöntemlere göre belirgin bir avantajının olmadığı görülmektedir.

2.4.10. SSR için Önerilen Yeni bir MOOA

Yue ve ark. çok amaçlı parçacık sürü optimizasyon algoritmasına dayalı, MOPSO-iHalfT adı verilen bir MOSR tekniği önermiştir [88]. MOPSO-iHalfT'de diz bölgesini elde etmek için açı tabanlı yaklaşıma dayalı bir yöntem geliştirilmiştir. Yerel arama prosedürü olarak IHalfT kullanılmıştır.

MOPSO-iHalfT'nin geri oluşturma performansını incelemek için yapay olarak oluşturulmuş sinyaller, SPARCO araç kutusu ve görüntü sinyalleri kullanılmıştır. MOPSO-iHalfT'nin performansı IST, MP, OMP, IHalfT, StEMO ve MOEA/D-iHalfT yöntemleriyle karşılaştırılmıştır. Belirtilen çalışmaya göre MOPSO-iHalfT, elde edilen ölçüm hatası açısından değerlendirildiğinde diğer karşılaştırılan yöntemlere göre üstünlüğe sahiptir. Hesaplama süresi açısından değerlendirildiğinde bazı test sinyalleri için IHalfT ve IST'nin performansı, MOPSO-iHalfT'den daha iyi olmasına rağmen MOPSO-iHalfT; StEMO, OMP ve MP algoritmalarından daha küçük hesaplama süresine sahiptir. Ayrıntılı performans sonuçları ve karşılaştırmalar belirtilen çalışmada gösterilmektedir [88].

2.5.Yöntemlerin Değerlendirilmesi

Bu alanda yapılan çalışmalar incelendiğinde MOEA/D'nin bu alanda yaygın olarak kullanıldığı görülmektedir. MOEA/D, SSR problemini çözmek için uygun bir algoritmadır çünkü MOSR problemini ortak bir şekilde optimize edilecek birkaç skalar alt probleme böler. MOEA/D'nin kullanıldığı MOSR yöntemleri; MOEA/D-iHardT,

MOEA/D-iHalfT, MOEA/D-L1/2, MOEA/D-PGRASP ve MOEA/D-GMC metotlarıdır. NSGA II, DE, EA, MOBA ve son olarak MOPSO gibi diğer MOOA'lar sırasıyla StEMO, LBEA, ADEA, SPLS, MOBSO-OS, MOPSO-iHalfT gibi MOSR yöntemlerinde kullanılmıştır. MOSR yöntemlerinin çoğu, MOOP amaç fonksiyonları olarak Denklem (1.10)'u kullanır. Ancak, MOEA/D-PGRASP algoritması Denklem (2.12)'yi amaç fonksiyonu olarak kullanmaktadır. MOEA/D-PGRASP yöntemi ile MOEA/D-iHalfT, MOEA/D-iHardT, StEMO gibi diğer MOSR yöntemlerinde kullanılan optimizasyon algoritmalarının görevi birbirlerinden farklıdır. MOEA/D-PGRASP metodunda kullanılan optimizasyon algoritmasının görevi, algoritmanın ikinci aşamasında daha fazla özellik çıkarma ve sıfır olmayan giriş taraması için çoklu Pareto çözümleri sağlayarak genel bir geliştirme sürecini optimize etmektir.

Literatürde yer alan MOSR yöntemlerinin neredeyse tamamı bir yerel arama metoduna sahiptir. MOOA ile beraber yerel arama yöntemini kullanmanın amacı, bu algoritmaların yakınsama hızını artırmak ve algoritmanın geri oluşturma performansını geliştirmektir. IHalfT ve IHardT, sırasıyla MOEA/D-iHalft ve MOEA/D-iHardt yöntemlerinde yerel arama operatörü olarak kullanıldığından dolayı MOEA/D-iHalft, MOEA/D-iHardt, IHalfT ve IHardT algoritmalarının karşılaştırılması, yerel bir arama yöntemi kullanmanın performans üzerindeki etkilerini göstermektedir. Li ve ark.'nın yapmış olduğu çalışmada yer alan sonuçlara göre, MOEA/D-iHalft ve MOEA/D-iHardt, seyrek sinyalin yeniden yapılandırılmasında IHalfT ve IHardT metotlarından daha iyi performansa sahiptir [84]. Literatürde, IHalfT algoritmasının MOSR yöntemlerinde yerel arama yöntemi olarak yaygın bir şekilde kullanıldığı görülmektedir. Yerel arama yöntemi olarak IHalfT yöntemini kullanan MOSR algoritmaları, MOEA/D-iHalfT, SPLS, MOEA/D-L1/2, MOBSO-OS ve MOPSO-iHalfT'dir. Bunun yanı sıra, StEMO algoritmasında IST, LBEA'da LB yöntemi, ADEA'da iki aşamalı bir IST yöntemi, MOEA/D-GMC'de iki aşamalı yinelemeli arama prosedürü ve MOEA/D-iHardT'de ise IHardT yöntemi kullanılmaktadır. SPLS gibi bazı MOSR yöntemlerinde ise, IHardT, IHalfT ve IST gibi birden çok yerel arama metodu kullanabilmektedir.

Literatürdeki MOSR yöntemlerinden bazıları PF'nin tamamını bulmaya yönelik iken bazıları da PF'nin yerel bir kısmını bulmaktadır. MOEA/D-iHardT, MOEA/D-iHalfT ve SPLS algoritmaları PF'yi kısmi olarak bulmaya çalışırken LBEA, StEMO, MOEA/D-

L1/2, ADEA ve MOEA/D-GMC metotları PF'nin tamamını bulmaya yönelik tasarlanmışlardır.

Tasarımda MOSR algoritmalarının performansını etkileyen bir diğer kavram da PF üzerinden nihai çözümü seçme yöntemidir. Ağırlıklı toplam yöntemi, açıya tabanlı yöntem ve Kink yöntemleri nihai çözümü elde etmek için kullanılan literatürdeki temel yöntemlerdendir. StEMO ve MOPSO-IHalfT'de açıya dayalı yöntem; MOEA/D-L1/2 ve MOBSO-OS'ta ağırlıklı toplam yöntemi; LBEA ve ADEA'da ise Kink yöntemi kullanılmaktadır. Bunun yanı sıra, SPLS'de gerçek *k*-seyrek çözüm üzerinde tercihe dayalı bir yöntem kullanılırken MOEA/D-GMC'de ise Gauss karışım modelleri aracılığıyla kümeleme yöntemi kullanılmaktadır. MOEA/D-PGRASP metodunda önceden tanımlanmış sınırlar arasında bulunan ölçüm hatasına sahip çözüm, nihai çözüm olarak seçilmektedir. Liang ve ark.'nın yaptığı çalışmada amaç değerlerin ağırlıklı toplamı, açıya dayalı ve uç çizgiye olan mesafe yöntemleri incelenmiştir [92]. Bu üç yöntemin MOEA/D-iHalft algoritması temelinde sinyali geri oluşturma performansları, belirtilen çalışmadaki test fonksiyonları kullanılarak karşılaştırılmıştır. Elde edilen sinyallerin ölçüm ve seyreklik hataları değerlendirilmiştir. Belirtilen çalışmadaki karşılaştırmaya göre ağırlıklı toplam yöntemi test fonksiyonlarının çoğunda, karşılaştırılan diğer yöntemlerden üstündür. Ağırlıklı toplam yönteminin hesaplama maliyeti, uç çizgiye olan mesafe ve açı tabanlı yöntemin hesaplama maliyetinden daha küçüktür. Dolayısıyla ağırlıklı toplam yöntemi tüm test fonksiyonlarında, karşılaştırılan diğer yöntemlerden daha az hesaplama süresine sahiptir.

Literatürde yer alan MOSR yöntemlerinin hesaplama karmaşıklığına ilişkin bilgiler, ilgili çalışmalardan elde edilerek bu bölümde özetlenmiş ve Tablo 2.3'te verilmiştir. MOEA/D-PGRASP algoritmasının hesaplama karmaşıklığı $O(TMN)$'dir. Burada, M ölçüm sayısını, N sinyal uzunluğunu ve T yineleme sayısını göstermektedir.

Yan ve ark. tarafından yapılan çalışmada belirtilenlere göre, LBEA metodunun karmaşıklığı DE algoritmasının mutasyon, çaprazlama, bölgesel arama metodu ve seçme operatörüne göre belirlenmektedir [82]. Belirtilen operatörlerin karmaşıklıkları sırasıyla $O(2(Np))$, $O((Np)N_{ls}MN)$ ve $O(2(Np)^2)$ şeklindedir. Belirtilen çalışmada popülasyon sayısı Np, sinyal uzunluğu N'den çok küçük ve bölgesel arama metodunun yineleme

sayısı $Nls = 15$ olduğundan, LBEA'nın genel karmaşıklığı yaklaşık olarak $O(15 \times NpMN)$, StEMO'nun karmaşıklığı ise yaklaşık olarak $O(2N^2Np)$ şeklinde hesaplanmıştır.

ADEA'nın hesaplama karmaşıklığı esas olarak uyarlanabilir ayrıştırma tabanlı çevresel seçim ve yerel arama yöntemleri tarafından belirlenmektedir. Çevresel seçim yönteminde elitizm, seçim ve referans vektör uyarlamasının hesaplama karmaşıklıkları, en kötü duruma göre sırasıyla $O(2\times(Np + 2\Lambda r)^2)$, $O((Np + 2\Lambda r)^2)$ ve $O(2\times\Lambda r)$ şeklindedir. ADEA'da kullanılan yerel arama yönteminin hesaplama karmaşıklığı, $O(3/2(Np+2\Lambda r)N^2)$ ile üst sınırlıdır. Buradaki Λr, diz-yaklaşım bölgesindeki eklenen referans vektörlerinin sayısıdır. Popülasyon sayısı sinyal uzunluğundan çok küçük olduğu için ilgili makalede ADEA'nın hesaplama karmaşıklığı, en kötü durumda yaklaşık olarak $O(3/2(Np+2\Lambda r)N^2)$ şeklinde hesaplandığı belirtilmiştir.

MOEA/D-GMC, SPLS ve MOEA/D-L1/2 metotları yerel arama için aynı sayıda yineleme içermektedir. MOEA/D-GMC metodu diğer iki metottan farklı olarak Gauss karışım modelini kullanarak, $O(PDM^3)$ karmaşıklık seviyesiyle sınıflandırmaktadır. Burada D ve P sırasıyla kümelerin ve nesnel vektörlerin sayısıdır. MOEA/D-L1/2 ve MOEA/D-GMC yeni popülasyonun oluşturulması işleminde aynı hesaplama karmaşıklığına sahiptir. SPLS metodu, hesaplama süresinin çoğunu k-seyrek bir çözüme yakın diz noktasına yakınsamaya harcarken; StEMO, MOEA/D-iHardT ve MOEA/D-iHalfT bir k-seyrek çözümden uzakta çok fazla hesaplama yüküne sahiptir.

Ayrıca MOPSO-iHalfT metodunun hesaplama karmaşıklığının hesaplanmasında PSO algoritmasının ana döngüsü, yerel arama yönteminin yinelemeleri ve tek boyut ağırlıklı yöntemdeki döngü hakimdir ve MOPSO-iHalfT algoritmasının genel hesaplama karmaşıklığı $O(Np(Nls+2Np))$ olarak belirtilmiştir [88]. MOSR yöntemlerinin tasarım özellikleri Tablo 2.2'de özetlenmiştir ve bazı MOSR yöntemlerinin hesaplama karmaşıklığı ve MOSR yöntemleriyle karşılaştırılan algoritmalar Tablo 2.3'te belirtilmiştir.

Bu bölümde MOSR yöntemleri incelenmiştir. Öncelikle, temel tasarım özellikleri değerlendirilmiştir. Literatürde yer alan metotların kullandıkları algoritmaların ve tasarım özelliklerinin farklılıkları vurgulanmış ve orijinal makaleleri göz önünde bulundurularak performansları hakkında bazı çıkarımlarda bulunulmuştur. Ayrıca literatürde yaygın kullanılan MOOA'lar, lokal arama metotları ve son çözümü seçme metotları kullanılarak

bazı MOSR yöntemleri tasarlanmıştır. Bu bölümde adımları verilen NSGA II, MOEA/D ve MOPSO algoritmaları MOSR probleminin optimizasyonunda kullanılmıştır. IHalfT metodu ise bu algoritmalarda yerel arama metodu olarak kullanılmıştır. IHalfT algoritması bu algoritmalarda oluşturulan popülasyonların MOSR probleminin çözümüne yönelik geliştirilmelerine katkı sağlamaktadır. NSGA II, MOEA/D ve MOPSO algoritmaları ve IHalfT algoritması literatürde yaygın kullanılan metotlardır. Buradaki amaç, yaygın kullanılan çok amaçlı tasarım yöntemlerinin SSR probleminin çözümündeki performansı üzerine olan etkisini incelemektir. NSGA II, MOEA/D ve MOPSO algoritmalarına dayalı geri oluşturma metotlarının performansları çeşitli test sinyalleri kullanılarak test edilmiştir. Bu testlere ait sonuçlar Bölüm 4'te verilmiştir

Tablo 2.2. MOSR yöntemlerinin tasarım özellikleri.

MOSR Yöntemi	**MOOA**	**Son Çözümü Seçme Yöntemi**	**Yerel Arama Metodu**
MOEA/D-iHardT, MOEA/D-iHalfT	MOEA/D	*	IHalfT IHardT
StEMO	NSGA II	Açı tabanlı yöntem	IST
MOEA/D-L1/2	MOEA/D	Ağırlıklı toplam fonksiyonu	IHalfT
MOEA/D-PGRASP	MOEA/D	Önceden tanımlanmış sınırlar arasında bulunan ölçüm hatasına sahip çözüm, nihai çözüm olarak seçilir.	En küçük kareler optimizasyonuna dayalı bir yöntem.
LBEA	DE	Kink yöntemi	LB yöntemi.
ADEA	EA	Kink yöntemi	İki aşamalı IST
SPLS	MOEA/D	Gerçek *k*-seyrek çözümde tercihe dayalı bir yöntem.	IHalfT, IHardT, IST. (Bu yöntemlerden herhangi biri uygulanabilir.)
MOEA/D-GMC	MOEA/D	Gauss karışım modelleri ile kümeleme.	İki aşamalı IHalfT
MOBSO-OS	MOBA	Ağırlıklı toplam fonksiyonu	IHalfT
MOPSO-iHalfT	MOPSO	Gürbüz açı tabanlı yöntem	IHalfT

***ilgili çalışmada belirtilmemiştir.**

Tablo 2.3. MOSR yöntemlerinin karşılaştırıldığı algoritmalar test sinyalleri ve hesaplama karmaşıklıkları.

MOSR Yöntemi	Karşılaştırılan Algoritmalar	Test Sinyalleri	Hesaplama Karmaşıklığı
MOEA/D-iHardT MOEA/D-iHalfT	**Klasik Yöntemler:** IHardT, IHalfT.	Yapay olarak üretilmiş sinyaller.	*
StEMO	**Klasik Yöntemler:** OMP, BP, Homotopy, IST, ADM.	SPARCO, yapay sinyaller, görüntü sinyalleri.	$O(2N^2Np)$
MOEA/D-L1/2	**Klasik Yöntemler:** IHalfT **MOSR Yöntemleri:** MOEA/D-old.	Yapay olarak üretilmiş sinyaller	*
MOEA/D-PGRASP	**Klasik Yöntemler:** OMP, SPGL1, BPDN, Smoothed L0 (SL0), Elastic-net (EN). **MOSR Yöntemleri:** M+GRASP, MOEA/D-best.	SPARCO yazılımı, yapay olarak üretilmiş sinyaller	$O(TMN)$
LBEA	**Klasik Yöntemler:** OMP, Homotopy, L1LS, FISTA. **MOSR Yöntemleri:** StEMO.	Yapay olarak üretilmiş sinyaller, görüntü sinyalleri.	$O(NpN_{ls}MN)$
ADEA	**Klasik Yöntemler:** BP, OMP, Homotopy, FISTA. **MOSR Yöntemleri**: StEMO, LBEA, ADEA.	SPARCO yazılımı, yapay olarak üretilmiş sinyaller, görüntü sinyalleri.	$O(3/2(Np+2\Lambda r)N^2)$
SPLS	**Klasik Yöntemler:** Yinelemeli eşik algoritmaları, OMP. **MOSR Yöntemleri**: StEMO (StEMO/L0.5, StEMO/L0, StEMO/L1), MOEA/D.	Yapay olarak üretilmiş sinyaller, görüntü sinyalleri.	*
MOEA/D-GMC	**Klasik Yöntemler**: SPLS, MOEA/D-L1/2, IHalfT **MOSR Yöntemleri**: STEMO.	Yapay olarak üretilmiş sinyaller	$O(PDM^3)$
MOBSO-OS	**Klasik Yöntemler:** Homotopy, MP, OMP. **MOSR Yöntemleri**: KnMOPSO.	SPARCO yazılımı, yapay olarak üretilmiş sinyaller	*
MOPSO-iHalfT	**Klasik Yöntemler:** IHalfT, OMP, MP, IST. **MOSR Yöntemleri**: MOEA/D-iHalfT, StEMO.	SPARCO yazılımı, yapay olarak üretilmiş sinyaller, görüntü sinyalleri	$O(Np(N_{ls}+2Np))$

***ilgili çalışmada belirtilmemiştir.**

.

3. BÖLÜM

ÇOK AMAÇLI YAPAY ARI KOLONİ ALGORİTMASINA DAYALI SEYREK SİNYAL GERİ OLUŞUM YÖNTEMİ

SSR, sıkıştırma ve algılama işlemlerini aynı anda gerçekleştiren bir sinyal işleme yöntemi olan CS işleminin performansını doğrudan etkileyen önemli aşamalardan biridir. Genel olarak SSR algoritmaları; dışbükey optimizasyon, dışbükey olmayan optimizasyon ve açgözlü algoritmalar olarak sınıflandırılabilir. Son yıllarda çok amaçlı optimizasyon algoritmaları SSR'de kullanılmaya başlanmıştır. Tez çalışmasının bu bölümünde, seyrek sinyal geri oluşum problemleri için seyrekliği ve ölçüm hatasını aynı anda optimize eden MOABC algoritmasına dayalı bir SSR yöntemi önerilmiştir. Önerilen yönteme seyrek geri oluşumu için çok amaçlı yapay arı koloni algoritması (Multi-Objective Artificial Bee Colony Algorithm for Sparse Signal Reconstruction, MOABC-SR) adı verilmiştir [98]. Çalışmanın bu bölümünde ilk olarak ABC algoritmasının temel prensiplerinden bahsedilmiştir. Daha sonra, SSR için tasarlanan MOABC-SR algoritmasının detayları Algoritma 3.1'de açıklanmıştır. Ayrıca, önerilen yöntemin yakınsamasını iyileştirmek için kullanılan yerel arama yöntemi Algoritma 3.2'de verilmiştir. Son olarak, PF'den nihai bir çözüm seçmek için kullanılan yöntem Algoritma 3.3'te açıklanmıştır. Bu bölümde önerilen yöntem çeşitli test sinyalleri kullanılarak değerlendirilmiş ve diğer SSR algoritmaları ile karşılaştırılmıştır. Karşılaştırmalara ait performans verileri ve yorumları 4. Bölümde yer almaktadır.

3.1. ABC Algoritması

Yapay Arı Koloni (Artificial Bee Colony, ABC) algoritması, optimizasyon problemlerinin çözümü için 2005 yılında Derviş Karaboğa tarafından önerilmiştir [99, 100]. ABC, gerçek bal arılarının akıllı yiyecek arama davranışlarının benzetimini yapmaktadır. Koloni boyutu, sınır değeri ve maksimum döngü sayısı gibi az sayıda kontrol parametresine sahip olan ABC, etkili ve verimli bir optimizasyon algoritmasıdır.

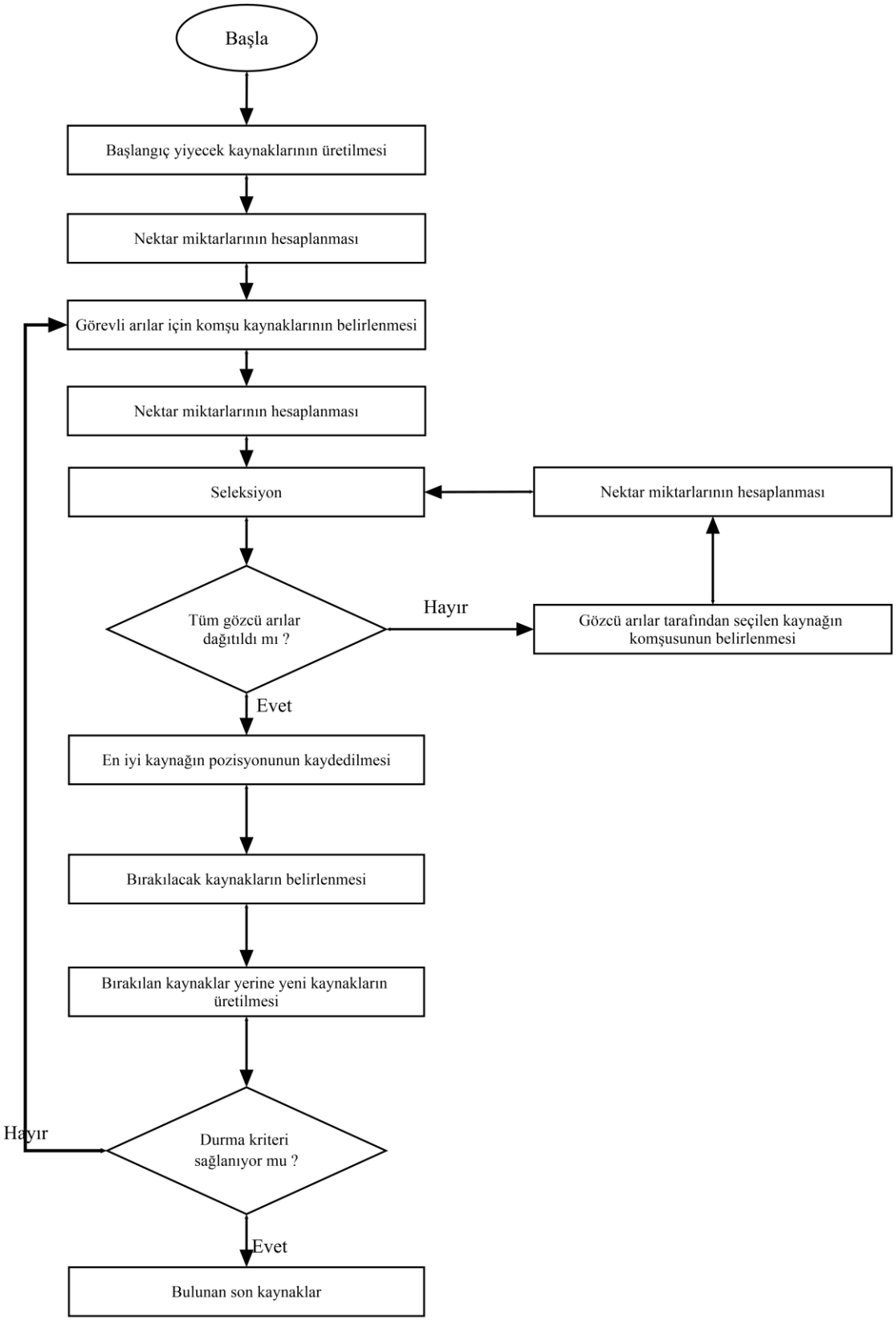
ABC algoritması popülasyona dayalı bir arama prosedürü sağlar. Bu işlemde yiyecek kaynaklarının yerleri adı verilen popülasyon zamanla yapay arılar tarafından güncellenir. Algoritmada arılar, yüksek miktarda nektar içeren yiyecek kaynaklarının ve son olarak en yüksek nektar içeren yiyecek kaynaklarının yerlerini keşfetmeyi amaçlar. ABC algoritmasının genel yapısı şu şekildedir:

Başlangıç Evresi

REPEAT
İşçi arı evresi
Gözcü arı evresi
Kâşif arı evresi
Elde edilen en iyi yiyecek kaynağı konumunun kaydedilmesi

UNTIL (güncel döngü sayısı = maksimum döngü sayısı)

ABC algoritmasının akış şeması Şekil 3.1'de gösterilmiştir [101]. Bazı çalışmalarda ABC algoritmasının performansı değerlendirilmiş ve yaygın olarak kullanılan algoritmalarla karşılaştırılmıştır. Karaboğa ve ark. ABC'nin performansını Genetik Algoritma (GA), Parçacık Sürü Optimizasyonu (PSO) ve Parçacık Sürüsünden Esinlenmiş Evrimsel Algoritma (PS-EA) ile karşılaştırmıştır [102]. Karşılaştırılan algoritmalar, sürü zekâsı ve ABC algoritması gibi popülasyon tabanlı algoritmalardır. ABC algoritmasının performansını değerlendirmek için PSO, PS-EA, GA ve ABC algoritmaları, çok modlu olan beş adet yüksek boyutlu test fonksiyonu üzerinde test edilmiştir. Elde edilen sonuçlara göre ABC algoritmasının yerel minimumdan başarılı bir şekilde çıkabileceği ve çok modlu optimizasyon fonksiyonları için verimli bir şekilde kullanılabileceği sonucuna varılmıştır. Karaboğa ve Baştürk'ün yapmış olduğu bir çalışmada, ABC algoritmasının performansı DE, EA ve PSO algoritmaları ile çok boyutlu sayısal problemler kullanılarak karşılaştırılmıştır [103]. Bahsedilen çalışmanın sonuçlarına göre ABC algoritmasının performansı; DE, PSO, EA ile rekabet edebilmektedir ve yüksek boyutlu mühendislik problemlerinin çözümü için verimli bir şekilde kullanılabilmektedir. ABC çeşitli alanlarda farklı optimizasyon problemlerinin çözümünde kullanılmaktadır [104, 105].



Şekil 3.1. ABC algoritmasının akış şeması

3.2. Seyrek Sinyal Geri Oluşumu için Çok Amaçlı Yapay Arı Koloni Algoritması

Bu çalışmada, MOSR probleminin çözümü için MOABC algoritmasına dayalı MOABC-SR adı verilen bir yöntem tasarlanmıştır [98]. Bu yöntemde ABC algoritması, çok amaçlı optimizasyon problemlerinin çözümü için geliştirilmiş ve MOSR probleminin çözümü için kullanılmıştır. Algoritmanın arama hızını ve çözümün doğruluğunu artırmak için MOABC algoritmasıyla beraber yerel arama metodu kullanılmıştır. Ayrıca domine edilemeyen çözümlerin arasından son çözümü elde etmek için ölçüm hatasına bağlı bir son çözüm seçme yöntemi kullanılmıştır.

Tasarlanan MOABC-SR yönteminin adımları Algoritma 3.1'de gösterilmektedir. Yöntemde öncelikle, seyrek sinyal ve sinyallerin seyreklik seviyesi için yiyecek kaynakları pozisyonları rastgele bir şekilde üretilir. Yiyecek kaynaklarının nektar miktarları Denklem (1.10)'da ifade edilen amaç fonksiyonlarına göre hesaplanmaktadır. Amaç değerleri hesaplandıktan sonra, her bir yiyecek kaynağının domine edilip edilememe durumları belirlenir. Domine edilen ve edilemeyen yiyecek kaynakları Bölüm 2'de verilen dominantlık tanımına göre belirlenmektedir. MOABC algoritmasının işçi arı aşamasında, yeni çözüm yani yiyecek kaynağı üretme işlemi ABC algoritmasının yeni çözüm üretme prensibine göre yapılmaktadır. Yeni çözümler üretildikten sonra, çözümlerin uygunluk değerleri hesaplanır. Çözümlerin birbirlerine baskın olup olmama durumları değerlendirilir. Bu değerlendirmeden sonra yeni üretilen çözümle mevcut çözüm arasında aç gözlü bir seçme işlemi uygulanır ve daha iyi olan seçilir. Eğer üretilen çözüm mevcut çözüme baskın ise, mevcut çözüm ile değiştirilir. Her baskın olma durumu kontrolünde gelişemeyen çözümün baskın olamama sayaçları bir artırılır ve sonraki arı evrelerinde kullanılmak üzere kaydedilir.

MOABC algoritmasının gözcü arı aşamasında, SSR probleminde daha etkili bir arama için yeni çözümler mevcut çözümün seyreklik değerine göre üretilmektedir. Mevcut çözümün normalleştirilmiş seyreklik değeri, 0 ile 1 arasında üretilmiş rastgele bir sayıdan daha büyükse yeni bir çözüm üretilir. Üretilen çözümün uygunluk değerleri hesaplanarak baskın olup olmama durumları belirlenir. İşçi arı evresinde olduğu gibi çözümler ve çözümlerin gelişip gelişmeme durumları güncellenir. İşçi arı evresinden farklı olarak, minimum mutlak değerine sahip olan bir çözümün sadece bir girişi sıfıra eşitlenir. Bu şekilde, üretilen çözümün seyreklik seviyesi desteklenerek algoritmanın SSR problemi için yakınsama hızı ve bölgesel arama performansı geliştirilmektedir.

MOABC algoritmasının üçüncü aşaması kâşif arı evresidir. Bu aşamada işçi arı evresinde ve gözcü arı evresinde belirli bir süre gelişmeyen çözümler yeni çözümlerle değiştirilir. Gelişmemiş çözümün normalleştirilmiş seyreklik değeri, 0 ile 1 arasında üretilen rastgele bir sayıdan büyükse yeni çözüm yerel arama yöntemi kullanılarak üretilir. Bu çalışmada kullanılan yerel arama yönteminin adımları Algoritma 3.2'de gösterilmiştir. Kâşif arı evresindeki yerel arama yöntemi, baskın olmayan kümeden minimum seyreklik seviyesine sahip olan yiyecek kaynağını ilk çözüm olarak kullanır ve yeni çözümü üretir. Üretilen çözüm diğer evrelerde belirli bir süre gelişemeyen yani limit değerini aşan çözüm ile değiştirilir. Temel ABC algoritmasında her döngüde sadece bir kâşif arı sayısı belirlenir. Bu yöntemde gelişememiş çözümün seyreklik değerine ve sınır değerlerine göre kâşif arı sayısı belirlenerek algoritmanın yakınsama hızı geliştirilir. MOABC'nin ana adımlarından sonra, her bir yiyecek kaynağı konumunu iyileştirmek için yerel arama yöntemi kullanılır. Son olarak elde edilen yiyecek kaynakları değerlendirilir. Domine edilen ve domine edilemeyen yiyecek kaynağı kümeleri güncellenir ve maksimum yineleme sayısı şartı sağlanıncaya kadar işçi arı evresinden itibaren bu döngüler devam eder.

Maksimum yineleme sayısı şartı sağlandığında, nihai çözüm domine edilemeyen çözümler kümesinden seçilir. Nihai çözümün seçiminde domine edilemeyen çözümlerin sıfır olmayan konumları kullanılır. Domine edilemeyen çözümler ve ölçüm matrisinin sözde tersi kullanılarak gözlem vektörleri oluşturulur. Oluşturulan gözlem vektörleriyle gerçek gözlem vektörü arasındaki en küçük hataya sahip olan gözlem vektörünün hesaplandığı sinyal, geri elde edilmiş sinyal olarak seçilir. Nihai çözüm seçme yöntemi Algoritma 3.3'te gösterilmektedir.

Algoritma 3.1: MOABC-SR Algoritmasının adımları

Adım 1: Başlangıç

- Başlangıç değerlerinin atanması ve sayaçların sıfırlanması.
- Başlangıç yiyecek kaynakları olarak çeşitli seyreklik seviyelerinde rastgele seyrek sinyaller üretilmesi.

$$Yiyecek\ Kaynakları, P = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)} \ldots\ldots\ldots x^{(t)}\} \qquad (3.1)$$

Adım 2: Başlangıç yiyecek kaynaklarının değerlendirilmesi.

- Seyrek geri oluşturma probleminin iki amacı olan ölçüm hatasının ve seyrekliğinin hesaplanması.
- Domine edilen ve domine edilemeyen yiyecek kaynakları kümesinin belirlenmesi.

Adım 3: Ana Döngü

Adım 3.1: *İşçi Arı Evresi:* Denklem (3.2)'yi kullanarak işçi arılar için mevcut yiyecek kaynaklarının komşuluğunda yer alan yeni yiyecek kaynaklarının üretilmesi.

FOR i=1 to POP (Yiyecek kaynaklarının ($P_{i,j}$) her *i* üyesi için)

- Değiştirilecek *j* parametresinin rastgele bir şekilde belirlenmesi.
- Mevcut yiyecek kaynağının komşuluğunda olan *k* parametresinin belirlenmesi.
- Denklem (3.2)'yi kullanarak yeni çözüm geliştirilmesi

$$T_{i,j} = P_{i,j} + \phi_{i,j}\left(P_{i,j} - P_{k,j}\right) \quad (3.2)$$

Not: Denklem (3.2)'de yer alan *k*, *i*'nin komşuluğunda yer alan bir çözüm ve ϕ, [-1, 1] aralığında yer alan rastgele üretilmiş bir sayıdır.

- Üretilen yeni çözümün değerlendirilmesi.
- Eğer Denklem (3.2) kullanılarak elde edilen $T_{i,j}$ çözümü (yiyecek kaynağı), $P_{i,j}$ 'yi domine ederse mevcut yiyecek kaynaklarından olan $P_{i,j}$, $T_{i,j}$ ile değiştirilir.

END FOR

Adım 3.2: *Gözcü Arı Evresi:* Seyreklik seviyesine bağlı olarak P_i çözümlerinden seçilen gözcü arılar için yeni pozisyonların üretilmesi ve üretilen yeni pozisyonların değerlendirilmesi.

FOR *i*=1 to *Pop* (Yiyecek kaynaklarının ($P_{i,j}$) her *i* üyesi için)

IF (*i*. üyenin normalize edilmiş seyreklik değeri 0 ile 1 arasında üretilmiş olan rastgele bir sayıdan büyükse)

- Değiştirilecek *j* parametresinin rastgele bir şekilde belirlenmesi.
- Mevcut yiyecek kaynağının komşuluğunda olan *k* parametresinin belirlenmesi.
- Denklem (3.2)'yi kullanarak yeni çözüm geliştirilmesi.
- Üretilen yeni çözümde min(abs($T_{i,j}$)) ifadesini sağlayan *j* indisinin bulunması ve $T_{i,j}$' nin sıfıra eşitlenmesi.
- Eğer elde edilen $T_{i,j}$, $P_{i,j}$'yi domine ederse mevcut yiyecek kaynaklarından olan $P_{i,j}$, $T_{i,j}$ ile değiştirilir.

END IF

END FOR

Adım 3.3: *Kâşif Arı Evresi*: Her bir yiyecek kaynağının gelişim göstermediği döngüler bir sayaca (*trial_counter*) kaydedilir. Gelişim göstermeyen döngü sayısı, belirlenen "*limit*" değerini aşan yiyecek kaynakları belirlenir ve bunlar yerel arama yöntemiyle oluşturulan yeni yiyecek kaynaklarıyla değiştirilir.

FOR *i*=1 to gelişim göstermeyen kaynak sayısı

IF (*trial_counter* > *limit*)

- *k''* (domine edilmeyen gıda kaynağı setinin minimum seyrekliği) değerini bul.
- *Temp* (*k''* değerine sahip besin kaynağı) değerini bul.

IF (*k''* > *rand*(0,1))

- Yeni gıda kaynağı= *Bölgesel arama metodu* (***A***, ***y***, *Temp, k'', Ls*). Not: *Ls*, yerel arama yönteminin yineleme sayısıdır.
- Yeni yiyecek kaynağının genel yiyecek kaynağına eklenmesi.

END IF

END IF

END FOR

Adım 4: Yiyecek kaynaklarının, yerel arama yöntemiyle iyileştirilmesi

- *Yiyecek kaynakları=Bölgesel arama metodu* (***A***, ***y****, yiyecek kaynakları, yiyecek kaynaklarını seyreklik seviyeleri, Ls*)

Adım 5: Gıda kaynaklarının değerlendirilmesi

- Amaç fonksiyonlarının değerlerinin hesaplanması
- Domine edilen ve domine edilmeyen yiyecek kaynaklarının belirlenmesi

Adım 6: Eğer algoritmanın durma koşulu sağlanıyor ise durulur, sağlanmıyorsa Adım 3'e gidilir.

Adım 7: Geri elde edilen sinyal, Algoritma 3'te yer alan nihai çözümü seçme metoduyla domine edilemeyen küme içinden seçilir.

Önerilen yöntemin ana bölümleri ABC'nin ana döngüsü, yerel arama yöntemi ve son çözüm seçim yönteminden oluşmaktadır. Bu nedenle, MOABC-SR'nin hesaplama karmaşıklığı, ABC'nin yiyecek kaynağı sayısı *Nf*, yerel arama yönteminin döngü sayısı *Nls* ve son çözüm seçim yönteminin döngü sayısı tarafından belirlenmektedir. Yerel arama yöntemi ABC'nin ana döngüsünde iki kez kullanılırken, son çözüm seçim yöntemi önerilen yöntemin ana döngüsünün dışında sadece bir kez kullanılır. Son seçim yönteminin döngü sayısı, son domine edilemeyen kümenin eleman sayısına (*Nnd*) eşittir ve bu sayı yiyecek kaynağı sayısından çok daha azdır. Dolayısıyla önerilen yöntemin hesaplama karmaşıklığı yaklaşık olarak $O(N(Nf(2\times Nls)+Nnd))$ şeklindedir.

3.2.1. Bölgesel Arama Metodu

MOSR algoritmalarının yakınsama hızını iyileştirmek için bölgesel arama yöntemleri kullanılır [85]. Ayrıca popülasyondaki her çözüm, MOOA ile beraber bölgesel arama yöntemi kullanılarak geliştirilebilir. Bu geliştirmeler yakınsama hızına ve çözümün doğruluğuna etki eder. Bu çalışmada Algoritma 3.2'de gösterilen yinelemeli eşikleme algoritması, sinyalin geri elde edilmesi için önerilen yöntemin yerel arama yeteneğini geliştirmek için kullanılmıştır.

Algoritma 3.2: *Yinelemeli Yarım Eşikleme Algoritması*
Girişler

***A*:** ölçüm matrisi

***y*:** gözlem vektörü

$\boldsymbol{x_{başlangıç}}$**:** Başlangıç çözümü

k: seyreklik seviyesi

ls: maksimum iterasyon sayısı

n: güncel iterasyon sayısı

μ*:* adım büyüklüğü

Algoritma Adımları

Adım 1: Başlangıç değerlerinin atanması ($\boldsymbol{x^{(n)} = x_{başlangıç}}$) ve sayaçların sıfırlanması (n = 0)

Adım 2: Denklem (2.2)'de verilen Gradyan iniş yöntemini kullanarak bir deneme çözümü $\widetilde{\boldsymbol{x}} \in R^N$ üretilmesi.

Adım 3: $\widetilde{\boldsymbol{x}}$ sinyalinin bileşenlerinin Denklem (3.3)'teki gibi sıralanması.

$$|\widetilde{\boldsymbol{x}}_{i_1}| \geq |\widetilde{\boldsymbol{x}}_{i_2}| \geq \cdots \geq |\widetilde{\boldsymbol{x}}_{i_n}| \tag{3.3}$$

Adım 4: Düzenleme parametresinin Denklem (3.4)'te verilen ifadeye göre seçilmesi.

$$\lambda_n = \frac{\sqrt{96}}{9}|\widetilde{\boldsymbol{x}}_{\boldsymbol{i}_{\bar{k}+1}}|^{3/2} \tag{3.4}$$

Adım 5: Seyrek sinyali elde edebilmek için $\widetilde{\boldsymbol{x}}$ sinyalinin en küçük bileşenlerinden bazılarının Denklem (2.6)-(2.8) kullanılarak sıfıra eşitlenmesi.

Adım 6: Eğer *n>ls* ise ***x*** sinyali çıkış olarak tanımlanır, aksi takdirde döngü sayısı artırılır (*n=n+1*) ve Adım 2'ye gidilir.

3.2.2. Nihai çözümün elde edilmesi

MOOA'lar tek amaçlı optimizasyon algoritmalarının aksine, tek seferde bir dizi çözüm bulmaktadırlar. Elde edilen çözümler birbiriyle çelişen çözümlerdir ve aynı anda geliştirilemezler. Genel olarak bir karar vericinin, farklı hedeflerin önemini göz önünde bulundurarak PF'deki tüm çelişen çözümlerden bir çözüm seçmesi gerekir. Birbiriyle çelişen hedeflerin sayısı arttığında, en iyi çözümü seçmek karar vericiler için zordur. Literatürdeki nihai çözüm tanımlama yöntemlerinden olan amaç değerlerinin ağırlıklı

toplamı [78], açıya dayalı yöntem [90] ve Kink yöntemi [91] mevcut MOSR algoritmalarında yaygın olarak kullanılmaktadır. Bu çalışmada bunlardan farklı bir nihai çözüm tanımlama yöntemi kullanılmıştır. Ödünleşim çözümler elde edildikten sonra bu çözümler içinden bir optimum çözüm seçilmesi gerekmektedir. Gerçek sinyal hakkında var olan bilgi gözlem vektörüdür. Domine edilemeyen çözümlerden gerçek gözlem vektörüne en yakın olan gözlem vektörünü oluşturan çözüm optimum çözüm olarak kabul edilir. Temel olarak bu yöntem, domine edilemeyen çözümlerin sıfır olmayan girişlerini bulma, ölçüm matrisinin karşılık gelen sütunlarının sözde tersinin hesaplanması ve gerçek gözlem vektörü ile karşılaştırmak için bir deneme gözlem vektörünün hesaplanması gibi ana adımları içerir. Nihai seçim yönteminin adımları Algoritma 3.3'te verilmiştir.

Algoritma 3.3: *Nihai Çözümün Seçimi*

FOR (her bir $\boldsymbol{x'} \in \boldsymbol{P_{non_dominated}}$)

Adım 1: $\boldsymbol{x'}$ vektörü için sıfır olmayan elemanlarını Denklem (3.5)'de belirtilen aralıkta bulunması.

$$1 < I_1, I_2 \dots \dots I_k < N \tag{3.5}$$

Adım 2: Ölçüm matrisi olan $\boldsymbol{A}$ matrisi sütunlarının Denklem (3.6)'da belirtildiği gibi seçilmesi.

$$\boldsymbol{C} = \boldsymbol{A}_{(I_1, I_2 \dots\dots I_k)} \tag{3.6}$$

Adım 3: Denklem (3.7)'yi kullanarak $\boldsymbol{A}$ matrisinin seçilen sütunlarının sözde tersinin hesaplanması.

$$\boldsymbol{D} = (\boldsymbol{C^T C})^{-1} \boldsymbol{C^T} \tag{3.7}$$

Adım 4: Denklem (3.8)'i kullanarak, Denklem (3.7)'den elde edilen $\boldsymbol{D}$ matrisinin orijinal gözlem vektörü olan $\boldsymbol{y}$ ile çarpılması.

$$\boldsymbol{x'} = \boldsymbol{Dy} \tag{3.8}$$

Adım 5: Denklem (3.8)'den elde edilen $\boldsymbol{x'}$ vektörü ve $\boldsymbol{A}$ matrisini kullanarak Denklem (3.9)'daki gibi bir gözlem vektörü hesaplanması.

$$\boldsymbol{b'} = \boldsymbol{Ax'} \tag{3.9}$$

Adım 6: Denklem (3.10)'u kullanarak gerçek gözlem vektörü ve elde edilen gözlem vektörü arasındaki hatanın hesaplanması.

$$e = \|\boldsymbol{b} - \boldsymbol{b'}\|_2 = \sqrt{(\boldsymbol{b}_1 - \boldsymbol{b'}_1)^2 + (\boldsymbol{b}_2 - \boldsymbol{b'}_2)^2 + \cdots (\boldsymbol{b_m} - \boldsymbol{b'}_m)^2} \tag{3.10}$$

END FOR

Adım 7: Domine edilemeyen kümenin en küçük *e* değerine sahip üyesi nihai çözüm olarak belirlenir.

Tez çalışmasının bu bölümünde SSR problemleri için MOABC algoritmasına dayalı bir yöntem önerilmiştir. MOABC-SR, seyreklik ve ölçüm hatası hedeflerini aynı anda optimize etmektedir. Önerilen yöntemin diğer algoritmalara göre daha az kontrol parametresi vardır ve verimlidir. Ayrıca, yerel arama yöntemi olarak IHalfT algoritması kullanılmıştır. Nihai çözümün seçimi, elde edilen domine edilemeyen çözümlerin sözde tersini temel alan bir yöntemle yapılır. Dördüncü bölümde MOABC-SR algoritması ile diğer sık kullanılan diz noktası seçim yöntemleri de test edilmiş ve bu bölümde tasarlanan nihai çözüm elde etme yöntemi ile karşılaştırılmıştır.

4. BÖLÜM

UYGULAMA VE SONUÇLAR

Tez çalışmasının bu bölümünde MOABC, MOEA/D, NSGA II ve MOPSO algoritmalarına dayalı MOSR yöntemlerinin SSR performansları çeşitli test problemleri kullanılarak test edilmiştir. Bu test problemleri yapay olarak üretilen sinyaller, SOTF test sinyalleri [106], görüntü sinyalleri ve EKG sinyallerinden oluşmaktadır. Tablo 4.1'de yer alan SOTF test sinyalleri literatürde önerilen diğer MOSR yöntemlerinin test edilmesinde de kullanılmıştır [78, 79, 88]. Dolayısıyla bu test sinyallerinin kullanılması literatürdeki diğer MOSR yöntemleriyle karşılaştırma imkânı sağlamaktadır. Ayrıca, MOABC-SR metodu diğer SSR algoritmaları ile karşılaştırılmıştır. Karşılaştırılan algoritmalar KnMOPSO [107], IHT, OMP, Homotopy, FISTA ve İlkel Artırılmış Lagrange Çarpanı (Primal Augmented Lagrange Multiplier, PALM) [108]'dir. Bu algoritmalara ek olarak, Bölüm 2'de belirtilen MOEA/D, NSGA II ve MOPSO algoritmaları SSR probleminin optimizasyonu için kullanılmış ve SOTF problemleri üzerinde test edilmiştir. Bölgesel arama yöntemi, son seçim yöntemleri ve ortak algoritma parametreleri MOABC, MOEA/D, NSGA II ve MOPSO için aynıdır. Bu çalışmada karşılaştırılan algoritmalar, çeşitli SSR algoritma sınıflarını kapsayacak şekilde seçilmiştir. KnMOPSO algoritması, SSR probleminin optimize edilmesinde MOPSO algoritmasını kullanmaktadır. KnMOPSO metodunun kodu yazarlar tarafından sağlanan web sayfasından [109] elde edilmiştir. IHalfT ve FISTA yinelemeli eşikleme algoritmaları sınıfına aitken, Homotopy ve PALM ise l_1-minimizasyon algoritmaları sınıfındandır. OMP ise açgözlü algoritmalar sınıfına dahil olmaktadır. SSR işleminin performans ölçütlerinde geri oluşturma hatası (Reconstruction Error, RE), ölçüm hatası (Measurement Error, ME) ve bağıl seyreklik hatası (Relative Sparsity Error, RSE) değerleri sırasıyla Denklem (4.1)-(4.3)'e göre hesaplanmıştır.

$$RE = \frac{||\boldsymbol{x}' - \boldsymbol{x}||_2}{||\boldsymbol{x}||_2} \tag{4.1}$$

$$ME = ||\boldsymbol{Ax}' - \boldsymbol{y}||_2^2 \quad (4.2)$$

$$RSE = |k' - k|/k \quad (4.3)$$

RE ve RSE gerçek sinyal ile geri elde edilen sinyal arasındaki hata ilişkisini, ME ise gerçek gözlem vektörü ile geri oluşturulan sinyalin gözlem vektörü arasındaki hata ilişkisini göstermektedir. ME, geri elde edilen sinyal hatalı da olsa, yüksek seyreklik düzeyine sahip geri oluşturulmuş sinyal için çok düşük olabilir. Dolayısıyla, SSR işleminin performansına karar vermek için bu üç parametreyi birlikte değerlendirmek gerekir. Bu bölümdeki simülasyon sonuçları, 3.30 GHz hızında i5 işlemciye ve 4 GB RAM hafıza birimine sahip bir bilgisayarda MATLAB 2015a kullanılarak elde edilmiştir.

4.1. Algoritmaların SOTF Test Problemleri Kullanılarak Test Edilmesi

Simülasyon çalışmalarının bu bölümünde, algoritmaların SSR performanslarını incelemek için Tablo 4.1'de yer alan SOTF sinyalleri kullanılmıştır. MOABC-SR, KnMOPSO, MOEA/D, NSGA II ve MOPSO algoritmalarının ortak olan kontrol parametreleri; popülasyon sayısı, maksimum yineleme sayısı ve bölgesel arama metodunun yineleme sayısıdır. Bu çalışmada kullanılan MOOA parametreleri Tablo 4.2'de yer almaktadır. Tüm algoritmaların parametreleri tüm test problemleri için aynıdır. Algoritmalar tüm test durumları için 30 kez çalıştırılmıştır. Algoritmaların SSR performanslarını gösteren RE, ME ve RSE parametrelerinin ortalama değerleri sırasıyla Tablo 4.3-4.5'te verilmiştir. OMP ve IHalfT algoritmalarının seyreklik seviyesi bilgisi önceden verildiğinden dolayı RSE değerleri bu iki algoritma için Tablo 4.5'te verilmemiştir.

Tablo 4.3'te elde edilen RE değerlerine göre, MOABC-SR algoritması seyrek SOTF1, SOTF2, SOTF4, SOTF5, SOTF6, SOTF7 ve SOTF8 sinyallerini tam bir şekilde yeniden yapılandırabilmiştir. MOABC-SR tarafından geri elde edilme işleminin ME ve RSE değerleri de bunu desteklemektedir. MOABC-SR'nın elde ettiği diğer test sinyallerinde bir miktar hata vardır. Bununla birlikte, MOABC-SR'nin geri elde etme performansı, genel olarak diğer karşılaştırılan algoritmalardan üstündür. Bazı sinyaller için OMP ve IHT algoritması, MOABC-SR'den daha küçük RE hatasına sahip olması OMP ve IHT algoritması için seyrekliğin önceden bilinmesinden kaynaklanabilir. KnMOPSO

algoritması değerlendirildiğinde Tablo 4.3-4.5'te görüldüğü gibi tüm SOTF test sinyalleri için önemli bir hata değerine sahiptir.

Tablo 4.1. Seyrek optimizasyon test problemleri

Test Fonksiyonu	**Sinyal Uzunluğu** (N)	**Gözlem Sayısı** (M)	**Seyreklik Değeri** (k)	**Gürültünün Varyansı** (Ω)
SOTF1	50	30	13	0
SOTF2	50	27	13	0
SOTF3	50	24	13	0
SOTF4	512	300	130	0
SOTF5	512	270	130	0
SOTF6	512	240	130	0
SOTF7	5120	3000	1300	0
SOTF8	5120	2700	1300	0
SOTF9	5120	2400	1300	0
SOTF10	50	30	13	0.01
SOTF11	50	27	13	0.01
SOTF12	50	24	13	0.01
SOTF13	512	300	130	0.01
SOTF14	512	270	130	0.01
SOTF15	512	240	130	0.01

Tablo 4.2. Bu tez çalışmasında kullanılan algoritma parametreleri.

Parametre/Algoritma	**MOABC**	**MOEA/D**	**NSGA II**	**MOPSO**	**KnMOPSO**
Popülasyon Sayısı	40	40	40	40	40
Yineleme Sayısı	200	200	200	200	200
Bölgesel arama yineleme sayısı	10	10	10	10	10
Limit	80	-	-	-	-
Çaprazlama Oranı	-	0.5	0.7	-	-
Mutasyon Oranı	-	-	0.4	-	-
Komşu Sayısı	-	4	-		-
W(atalet ağırlığı)	-	-	-	0.50	0.50
W_{damp} (atalet ağırlığı sönümleme oranı)	-	-	-	0.72	0.72
C1 (Bireysel öğrenme katsayısı)				2.05	2.05
C2 (Küresel öğrenme katsayısı)				2.05	2.05
Arşiv sayısı	-	25	-		-

Tablo 4.3a**.** Gürültüsüz sinyaller için geri oluşturma hatasının ortalama değerleri

Test Fonksiyonu	Algoritmalar RE Değerleri											
	MOSR					Klasik						
	MOABC-SR	MOEA/D	NSGA II	MOPSO	KnMOPSO	OMP	IHT	Homotopy	FISTA	PALM		
SOTF1	1.2883e-16	1.0002e-16	1.0600e-16	0.2141	0.3169	1.2972e-15	9.7350e-17	4.0264e-16	1.7245e-06	2.3506e-16		
SOTF2	2.9811e-16	0.3210	2.8810e-16	3.6961e-16	0.2818	1.1981e-15	0.5221	0.0486	0.0486	0.0486		
SOTF3	0.2694	0.6237	0.7554	0.5907	0.3744	0.2779	0.6140	0.4646	0.4646	0.4646		
SOTF4	4.1379e-16	3.7565e-16	3.7574e-16	4.6611e-15	0.3652	2.2843e-15	4.3618e-16	1.2224e-15	1.6616e-06	6.0378e-16		
SOTF5	6.0458e-16	5.5764e-16	1.6083e-15	0.0568	0.3819	5.0688e-04	4.2629e-16	0.1336	0.1272	0.1336		
SOTF6	7.3646e-15	3.1874e-14	1.7972e-14	0.3141	0.4574	0.0181	0.4560	0.4140	0.4144	0.4141		
SOTF7	1.2900e-15	0.1347	1.2766e-15	1.4884e-05	0.0888	9.7591e-06	1.1786e-15	0.7417	4.1779	0.0335		
SOTF8	2.6462e-12	0.1383	2.1147e-14	0.6145	0.2517	0.0012	1.3540e-15	0.7550	4.6065	0.1906		
SOTF9	0.6097	0.1280	0.6729	0.6091	0.3485	0.3800	0.4127	0.7554	5.4720	0.3926		

Tablo 4.3b**.** Gürültülü sinyaller için geri oluşturma hatasının ortalama değerleri

Test Fonksiyonu	Algoritmalar RE Değerleri									
	MOSR					Klasik				
	MOABC-SR	MOEA/D	NSGA II	MOPSO	KnMOPSO	OMP	IHT	Homotopy	FISTA	PALM
SOTF10	0.0073	0.0086	0.5114	0.3523	0.2188	0.0036	0.1399	0.0650	0.0270	0.0650
SOTF11	0.0137	0.0157	0.6878	0.5824	0.2439	0.9532	0.0107	0.2260	0.2067	0.2260
SOTF12	0.0120	0.4987	0.5875	0.3493	0.1642	0.0139	0.4525	0.0440	0.0442	0.0440
SOTF13	0.0089	0.2832	0.6112	0.4767	0.0197	0.0091	0.0090	0.0488	0.0454	0.0489
SOTF14	0.0112	0.0137	0.5999	0.5082	0.1774	0.0112	0.0088	0.2417	0.2387	0.2418
SOTF15	0.6513	0.7208	0.6964	0.6417	0.3032	0.6693	0.4489	0.4531	0.4538	0.4532

Bu test sonuçlarına göre, seyrek sinyalin geri elde edilmesinde KnMOPSO'nun diğer algoritmalara göre zayıf performansa sahip olduğu görülmektedir. Ancak KnMOPSO'nun RSE değeri SOTF9, SOTF10, SOTF13 ve SOTF15 problemleri için MOABC-SR'nin elde ettiği değerden daha iyidir. IHT algoritması SOTF1, SOTF4, SOTF5, SOTF7 ve SOTF8 sinyallerini doğru bir şekilde geri oluşturabilmiştir. Ayrıca, OMP algoritması SOTF1, SOTF2, SOTF4, SOTF5 ve SOTF7 sinyallerini düzgün bir şekilde geri oluşturabilirken, diğer sinyallerin geri oluşturulmasında belli bir hataya sahiptir ve tam olarak yeniden oluşturamamıştır. Homotopy, PALM ve FISTA, SOTF1 ve SOTF4 sinyallerini doğru bir şekilde geri oluşturabilmiştir. Bu algoritmalar diğer test sinyalleri için dikkate değer oranda hata büyüklüğüne sahiptir. Ayrıca RE değerleri de birbirine yakındır.

Tablo 4.4a**.** Gürültüsüz sinyaller için ölçüm hatasının ortalama değerleri

Test Fonksiyonları	Algoritmalar ME Değerleri									
	MOSR					Klasik				
	MOABC-SR	MOEA/D	NSGA II	MOPSO	KnMOPSO	OMP	IHT	Homotopy	FISTA	PALM
SOTF1	6.9039e-31	2.9067e-31	4.7710e-31	3.6927e-07	1.6098	5.4364e-29	6.7662e-31	2.8317e-30	2.5321e-11	2.4925e-30
SOTF2	1.4418e-30	8.6214e-23	7.4326e-31	1.8428e-30	1.5658	8.5052e-29	1.0621	8.7257e-29	1.1426e-09	3.0011e-12
SOTF3	1.5555e-04	1.6417e-21	4.4777e-31	9.2706e-10	0.4758	0.1186	0.1397	1.5022e-30	3.8877e-11	1.8371e-10
SOTF4	5.0338e-29	3.5384e-29	3.6335e-29	8.5936e-27	5.5480	1.2110e-27	5.3578e-29	2.2150e-28	2.6526e-10	1.0705e-28
SOTF5	6.9031e-29	3.4868e-29	1.1288e-28	5.5454e-20	4.5083	2.3886e-05	3.5278e-29	1.5482e-26	3.9409e-10	2.2392e-09
SOTF6	1.5570e-27	3.8680e-26	1.2219e-26	6.2070e-20	4.5271	0.0234	1.4003	3.6795e-27	3.7133e-10	2.7240e-08
SOTF7	5.7903e-27	4.4550e-08	5.5546e-27	6.5921e-07	10.8827	1.5526e-07	7.1769e-14	1.468e+03	1.7445e+03	0.0078
SOTF8	7.1116e-21	25.4973	4.0022e-25	5.5503e-19	0.1726	0.0434	7.0086e-14	1.525e+03	2.0460e+03	0.0018
SOTF9	0.0032	10.4388	1.0473e-22	1.6162e-07	20.5866	3.9125	2.9995	1.150e+03	1.5664e+03	4.7396e-04

Tablo 4.4b. Gürültülü sinyaller için ölçüm hatasının ortalama değerleri

Test Fonksiyonları	Algoritmalar ME Değerleri									
	MOSR					Klasik				
	MOABC-SR	MOEA/D	NSGA II	MOPSO	KnMOPSO	OMP	IHT	Homotopy	FISTA	PALM
SOTF10	7.7517e-05	2.6452e-05	8.4703e-28	6.3781e-06	1.5096	0.0020	0.1414	1.7891e-28	4.4545e-11	1.0207e-08
SOTF11	3.1974e-06	5.0128e-06	2.2442e-25	1.1208e-09	0.6068	0.1565	7.3170e-04	1.7383e-29	4.2359e-11	4.0137e-11
SOTF12	8.1706e-07	2.3588e-06	1.8971e-31	1.1193e-08	0.6496	0.0012	0.1715	1.4574e-29	3.3470e-11	7.7223e-13
SOTF13	0.0123	1.2773e-04	2.4493e-27	6.0098e-06	0.0523	0.0152	0.0155	8.7907e-27	4.1183e-10	4.2124e-09
SOTF14	0.0195	0.0020	2.3869e-23	3.3705e-04	0.8007	0.0183	0.0148	3.9694e-26	3.8407e-10	4.4410e-08
SOTF15	2.7228e-08	9.4894e-30	6.9451e-24	2.6203e-06	0.0588	0.7140	1.1223	2.3068e-26	3.7382e-10	3.0197e-09

Homotopy, PALM ve FISTA algoritmalarının performansları büyük sinyal uzunluğu ve yüksek seyrekliğine sahip test sinyallerini seyrek geri oluşturmak için pek uygun değildir. MOABC, MOEA/D ve NSGA II ye dayalı MOSR metotlarının geri elde etme performansları göz önüne alındığında MOABC-SR; SOTF3 ve SOTF9-15 test sinyallerinin geri elde edilmesinde NSGA II'den daha iyi performansa sahiptir. Bunların haricindeki test sinyalleri için benzer performans sonuçlarına sahiptirler. MOABC-SR, SOTF2, SOTF3, SOTF7-15 test sinyallerinin geri elde edilmesinde MOEA/D tabanlı yöntemden daha iyi performans sonuçlarına sahiptir.

Tablo 4.5a. Gürültüsüz sinyaller için bağıl seyreklik hatasının ortalama değerleri.

Test Fonksiyonları	Algoritmalar RSE Değerleri									
	MOSR					Klasik				
	MOABC-SR	MOEA/D	NSGA II	MOPSO	KnMOPSO	OMP	IHT	Homotopy	FISTA	PALM
SOTF1	0.0000	0.0000	0.0000	1.1358	0.8333	-	-	0.0000	0.3076	0.0000
SOTF2	0.0000	1.1538	0.0000	0.0000	0.6230	-	-	1.0769	2.7692	1.0769
SOTF3	0.8462	0.8461	2.0000	1.2948	0.9974	-	-	0.8461	1.2307	0.8461
SOTF4	0.0000	0.0000	0.0000	0.0000	0.5133	-	-	0.0000	0.4538	0.0000
SOTF5	0.0000	0.0000	0.0000	0.1782	0.7082	-	-	1.0769	1.7999	1.0769
SOTF6	0.0000	0.0000	0.0000	0.7217	0.6287	-	-	0.8461	1.7076	0.84615
SOTF7	0.0000	2.8107	0.0000	0.0015	0.1754	-	-	0.7026	0.7969	1.0093
SOTF8	0.0000	0.2615	0.0000	1.4915	1.4915	-	-	0.6840	0.5992	1.0315
SOTF9	0.7538	0.2092	1.4915	1.5823	0.0469	-	-	0.6844	0.4892	0.8325

Tablo 4.5b. Gürültüsüz sinyaller için bağıl seyreklik hatasının ortalama değerleri.

Test Fonksiyonları	Algoritmalar RSE Değerleri									
	MOSR					Klasik				
	MOABC-SR	MOEA/D	NSGA II	MOPSO	KnMOPSO	OMP	IHT	Homotopy	FISTA	PALM
SOTF10	0.8872	0.8461	1.7692	1.6128	0.7276	-	-	1.3076	2.3846	1.3076
SOTF11	0.8462	0.6153	1.4615	1.7153	0.9256	-	-	1.0769	2.2307	1.0769
SOTF12	0.8462	0.6923	1.6923	1.4615	0.8276	-	-	0.8461	1.1538	0.8461
SOTF13	0.0615	1.1384	2.1153	1.7456	0.0308	-	-	1.3076	1.93076	1.3076
SOTF14	0.0153	0.2461	1.6769	1.4997	0.0615	-	-	1.0769	1.76153	1.0769
SOTF15	0.9000	2.2230	1.5000	1.3594	0.7000	-	-	0.8461	1.66923	0.8461

MOABC-SR; SOTF2, SOTF4, SOTF7 ve SOTF15 problemleri haricinde MOPSO tabanlı algoritmadan daha düşük RE hatasına sahiptir. NSGA-II tabanlı yöntem SOTF3, SOTF9-15 için seyrek sinyalleri tam bir şekilde geri oluşturamamasına rağmen, Tablo 4.4'e göre tüm SOTF test sinyalleri için çok küçük ME değerlerine sahiptir. Ayrıca, NSGA-II'nin RE ve RSE değerleri sırasıyla Tablo 4.3 ve Tablo 4.5'te görüldüğü gibi yüksektir. Bu durum, NSGA II tabanlı yöntemin bu test koşulları için ödünleşim çözümleri bulamadığı, bunun yerine çok küçük ME değerine neden olan seyreklik değeri yüksek olan bir çözümü bulduğu anlamına gelir. Çünkü seyreklik değeri yüksek olan sinyallerin ME değerleri küçüktür. Ancak bu geri oluşturulan sinyal orijinal sinyale oldukça uzak olduğundan dolayı RE değerleri de yüksektir. Yue ve diğerleri tarafından yapılan bir çalışmada bazı SOTF sinyallerinin geri oluşturulmasının ölçüm hataları,

bahsedilen çalışmada verilmektedir [88]. Ancak bahsedilen çalışmada seyreklik hatası verilmemiştir. Daha yüksek seyreklik değerleri için çok küçük ölçüm hataları olabileceğinden, elde edilen sonuçlara göre algoritmanın performansı hakkında doğru bir çıkarım yapabilmek için diğer amaç değeri olan seyreklik hatası da verilmelidir.

Özetle, MOABC-SR metodu SOTF ile yapılan bu testlerde elde edilen performans değerleri göz önünde bulundurulduğunda diğer karşılaştırılan algoritmalara göre bazı üstünlüklere sahiptir. Diğer algoritmalardan daha fazla sayıda SOTF test sinyalini daha küçük oranda hata ile geri elde edebilmiştir.

4.2. Nihai Çözüm Bulma Metotlarının Karşılaştırılması

Çalışmanın bu bölümünde son çözümün elde edilmesi için bu çalışmada kullanılan yöntem; açı temelli yöntem ve Kink yöntemi ile karşılaştırılmıştır. MOSR metodu olarak MOABC-SR algoritması kullanılmıştır ve test koşulları her üç yöntem için de aynıdır. Bu yöntemler her bir SOTF problemi için 30 kez çalıştırılmış ve elde edilen ortalama RE değerleri Tablo 4.6'da gösterilmiştir.

Tablo 4.6. Son çözümü bulmak için kullanılan metotlara ait RE değerleri.

Test Fonksiyonu	**MOABC-SR Son çözümü seçme metodu**	**MOABC-SR Açı tabanlı metot**	**MOABC-SR Kink metodu**
SOTF1	1.2883e-16	1.3363e-04	1.1273e-16
SOTF2	2.9811e-16	3.3744e-16	3.4403e-16
SOTF3	0.2694	0.5419	0.5348
SOTF4	4.1379e-16	4.3840e-16	4.4826e-16
SOTF5	6.0458e-16	5.3395e-04	6.9322e-14
SOTF6	7.3646e-15	1.3853e-13	2.0115e-13
SOTF7	1.2900e-15	0.2137	1.4441e-15
SOTF8	2.6462e-12	0.1585	0.0008
SOTF9	0.6097	0.0335	0.6208
SOTF10	0.0073	0.0083	0.0084
SOTF11	0.0137	0.0137	0.0138
SOTF12	0.0120	0.1155	0.2007
SOTF13	0.0089	0.0122	0.0111
SOTF14	0.0112	0.1586	0.0110
SOTF15	0.6513	0.4370	0.3838

Tablo 4.6'daki verilere göre bu çalışmada kullanılan yöntem SOTF1, SOTF3, SOTF5-SOTF8, SOTF10, SOTF12, SOTF13 ve SOTF14 sinyalleri için açı tabanlı yöntemden daha düşük bir RE'ye sahiptir. Ayrıca yöntem, SOTF3 ve SOTF8-14 sinyallerinin geri oluşturulmasında Kink yönteminden daha üstündür. Ancak, SOTF15 sinyali için diğerlerinden daha yüksek RE değerine sahiptir. Üç yöntemin RE değerleri, SOTF sinyallerinin geri kalanı için birbirine yakındır. Özetle, bazı test sinyalleri için önerilen nihai çözüm seçim yöntemi, açı temelli yöntem ve Kink yöntemine göre daha düşük bir hataya sahiptir. Ayrıca yapılan bu çalışmaya göre, MOABC-SR algoritmasının diğer nihai çözüm seçim yöntemleriyle de uyumlu çalıştığı görülmektedir.

4.3. Görüntü Geri Oluşturma Problemi

Görüntü sinyalleri gibi daha büyük boyutlu sinyaller söz konusu olduğunda, ölçüm matrisinin depolanması için hafıza gereksinimi ve SSR için hesaplama karmaşıklığı artmaktadır. Bu artış, CS'nin pratik uygulamalarda kullanılmasını sınırlandırmaktadır. Büyük boyutlu görüntü işaretlerini daha etkili biçimde sıkıştırabilmek için blok tabanlı sıkıştırılmış algılama (BCS- Block Compressed Sensing) yöntemi geliştirilmiştir [110]. Bu yöntemde görüntü işareti birbiriyle örtüşmeyen küçük bloklara ayrılır. Her blok aynı ölçüm operatörü tarafından bağımsız olarak örneklenir. Blok tabanlı olmayan CS yöntemlerine kıyasla BCS yönteminde ölçüm matrisi boyutunun küçük olması, örnekleme işleminin basit ve verimli bir şekilde uygulanmasını sağlar. Bu sayede hafıza gereksinimi ve kodlayıcı biriminin karmaşıklığı önemli ölçüde azalarak, CS'de SSR işlemi önemli ölçüde hızlanır. Blok tabanlı ölçüm gerçek zamanlı uygulamalar için daha avantajlıdır çünkü kodlayıcı birimi görüntünün tamamının algılanmasını beklemek zorunda değildir. Kodlayıcı birimi doğrusal izdüşümü hesaplanan her bir bloğu diğer bir sinyal işleme birimine gönderebilir [110].

CS yönteminin $N \times N$ piksel boyutunda bir görüntü sinyaline uygulandığında, blok tabanlı olmayan CS yönteminde işlenecek olan görüntü sinyalinin boyutu $\boldsymbol{x} \in R^{N^2}$, ölçüm matrisinin boyutu $M \times N^2$ ve sinyalin seyrek olarak ifade edilmesi için uygulanan dönüşüm matrisinin boyutunun $N^2 \times N^2$ olması gerekmektedir. Güncel olarak kullanılan yüksek çözünürlüklü kameralarda N'nin değer aralığı 2000-4000 arasındadır [111]. N'nin bu değer aralığı için CS yöntemini uygulamak, ölçüm ve dönüşüm matrislerini kaydetmek için kullanılacak hafıza açısından pek mümkün değildir. Örnek olarak N=3000 ve

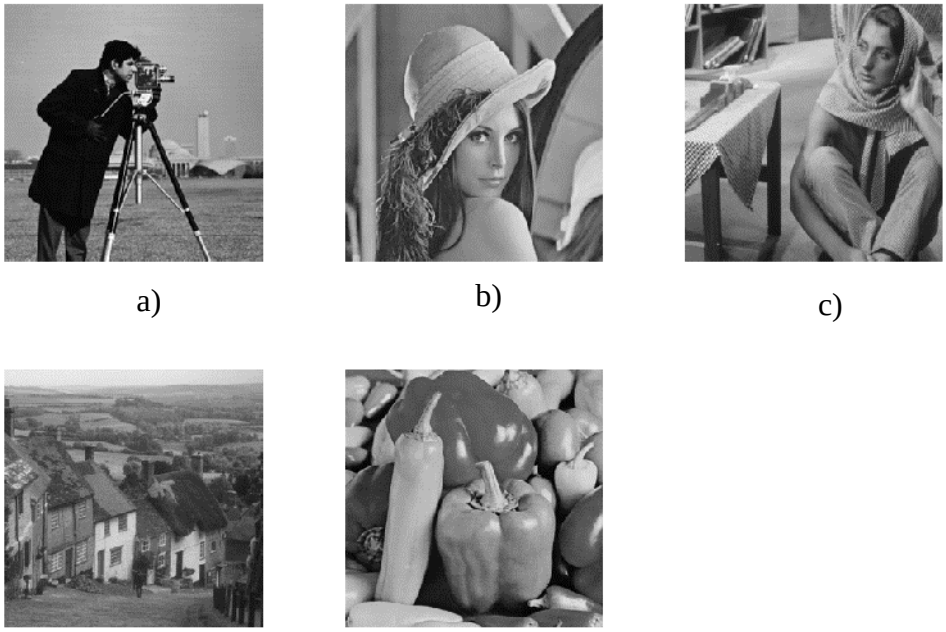
algılama oranı R=0.1 için, ölçüm matrisinin boyutu 900.000×9.000.000, dönüşüm matrisinin boyutu ise 9.000.000×9.000.000 olacaktır. Ayrıca CS geri elde etme problemini çözmek için gereken hesaplama maliyeti oldukça yüksek olacaktır. BCS yönteminde ise bir görüntü birbiriyle örtüşmeyen BxB piksel şeklinde bloklara ayrılır ve her blok bağımsız bir şekilde sıkıştırılmış olarak örneklenir. Tam boyutlu görüntü, seyrek geri elde edilmiş olan her bir görüntü bloğunun yeniden yapılandırılmış görüntü tuvalindeki yerine yerleştirilerek elde edilir. Daha sonra, bloklar arasındaki hataları gidermek için uygun filtreleme işlemi yapılır. Blokların ölçüm matrisi olan $\boldsymbol{A}_B$'nin boyutları B^2RxB^2 ve i. bloğun gözlem vektörü Denklem (4.4)'teki gibidir [111].

$$\boldsymbol{y}_i = \boldsymbol{A}_B \boldsymbol{x}_i \tag{4.4}$$

Bu zamana kadar BCS için çeşitli yöntemler sunulmuştur. Bu alanda yapılan bazı çalışmaların özetlendiği ve karşılaştırıldığı çalışmalar mevcuttur [112]. BCS alanında yapılan ilk çalışmalardan biri Gan, L. tarafından yapılmıştır [110]. Aynı örnekleme operatörü tarafından bağımsız bloklar şeklinde işlenen bu yöntem ilk olarak doğal görüntüler için önerilmiştir. Gan, L.' nin önermiş olduğu BCS yöntemi, diğer CS yöntemlerinden daha basit ve etkili olmasının yanında karmaşık geometriye sahip doğal görüntü işaretlerini başarılı bir şekilde işleyebilmektedir. Ayrıca bu yöntem, Wiener filtresi, konveks bir küme üzerine izdüşüm ve sert eşikleme gibi doğrusal ve doğrusal olmayan işlemler içermektedir. Bu alanda yapılan çalışmalardan biri de Mun, S. ve Fowler, J. E. tarafından yapılmıştır [113]. Bu çalışmada görüntü işareti için olan genel BCS çerçevesi, yineleme tabanlı yeniden oluşturma algoritmasıyla birleştirilmiştir. Ayrıca bu yöntemde, sinyalin seyrek olarak ifade edilmesinde sık kullanılan dalgacık dönüşümü yerine karmaşık değerli, çift-ağaç yapılı dalgacık dönüşümü kullanılmıştır. Bu alanda yapılan diğer bir çalışma ise Unde, A. S. ve Deepthi, P. P. tarafından yapılmıştır [114]. BCS için önerilen bu yöntem yinelemeli yeniden-ağırlıklandırılmış l_1-norm minimizasyonuna dayanmaktadır. L_1-norm minimizasyonu istenilen seyreklik değerini elde etmek için kullanılırken, Wiener filtresi de bloklara ayırma işleminden dolayı görüntüde ortaya çıkan pürüzleri düzeltmek için kullanılmaktadır.

Tez çalışmasının bu kısmında CS, görüntü sinyallerine uygulanmıştır. Algoritmalar görüntü sinyallerinin geri oluşturulması problemine uygulanarak karşılaştırılmıştır. Algoritmaların görüntü sinyallerine uygulanmasında BCS yöntemi kullanılmıştır. Test

görüntüleri, Şekil 4.1'de gösterilen 256×256 boyutundaki *Cameraman, Lena, Barbara, Houses* ve *Peppers'*tır.



a) b) c)

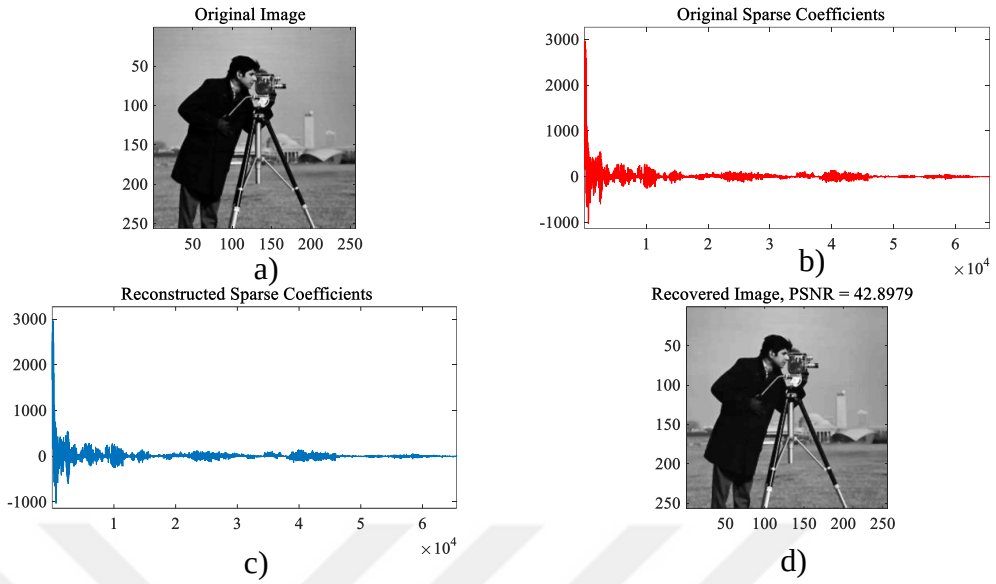
d) e)

Şekil 4.1. Test görüntüleri: a) *Cameraman,* b) *Lena*, c) *Barbara,* d) *Houses,* e) *Peppers*.

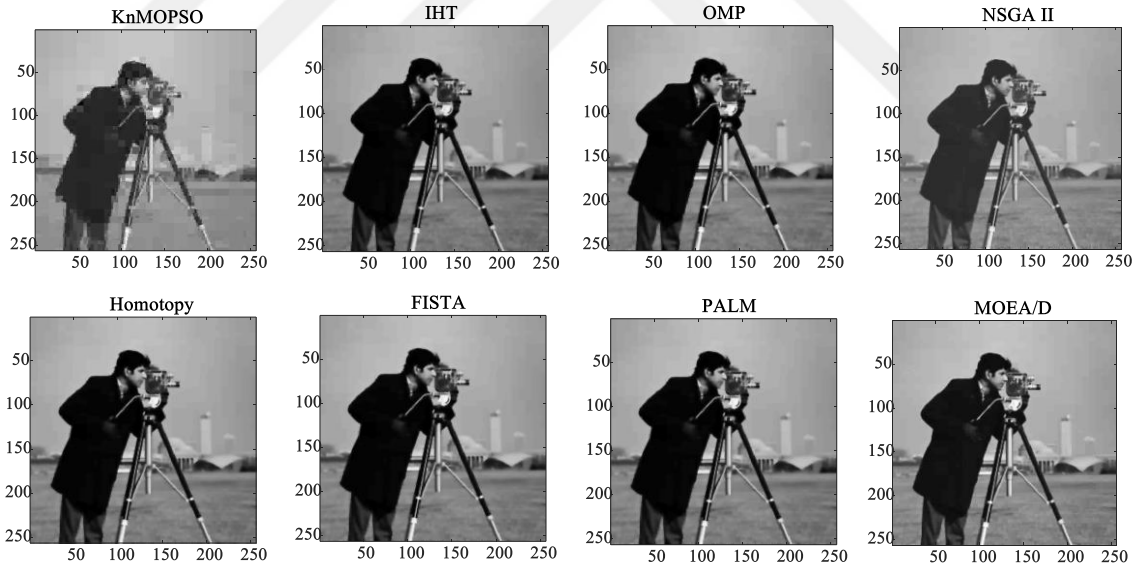
Görüntü sinyalinin seyrek bir temsilini elde etmek için "Db1" dalgacık ailesi kullanılmıştır. Bu deneyde, örnekleme oranı (*M/N*) tüm testler için 0.7'dir. Algoritmaların SSR performansları, PSNR ve SSIM parametreleri kullanılarak değerlendirilmiştir. Algoritmaların SSR performansları Tablo 4.7'de gösterilmektedir. Şekil 4.2'de orijinal *Cameraman* görüntüsü, orijinal görüntünün dalgacık katsayıları, geri oluşturulmuş dalgacık katsayıları ve MOABC-SR için geri oluşturulmuş görüntü yer almaktadır. Dalgacık Dönüşümü, orijinal sinyalin seyrek olarak ifade edilmesi için kullanılmıştır. Seyrek temsil elde edildikten sonra seyrek katsayılara CS uygulanmıştır ve bu katsayılar SSR algoritmaları kullanılarak geri oluşturulmuştur. Ters Dalgacık Dönüşümü, Şekil 4.2'de gösterildiği gibi görüntüyü geri elde etmek için uygulanmaktadır. KnMOPSO, IHT, OMP, Homotopy, FISTA, PALM, NSGA II ve MOEA/D tabanlı metotlar tarafından geri oluşturulmuş *Cameraman* test görüntüleri Şekil 4.3'te görülmektedir. Tablo 4.7'de elde edilen sonuçlara göre MOABC-SR; *Cameraman, Lena, Barbara* ve *Houses* görüntülerinin geri oluşturulmasında en yüksek PSNR ve SSIM'ye sahiptir. Homotopy yöntemi ise *Peppers* test görüntüleri için en yüksek PSNR ve SSIM değerlerine sahiptir.

Tablo 4.7. Çeşitli geri elde etme algoritmalarına ait PSNR ve SSIM değerleri

		PSNR/SSIM									
Algoritmalar		**Cameraman**		**Lena**		**Barbara**		**Houses**		**Peppers**	
MOSR	MOABC-SR	**42.8979**	**0.98150**	**43.2349**	**0.98201**	**39.7710**	**0.97282**	**38.6358**	**0.96132**	38.0909	0.96431
	KnMOPSO	25.4954	0.81925	25.9752	0.75492	24.6923	0.71089	27.0379	0.71032	25.0991	0.75952
	NSGA II	37.7054	0.97516	36.2881	0.94183	36.4009	0.96166	38.2259	0.95370	33.5452	0.96256
	MOEA/D	39.2215	0.96680	38.7508	0.96194	36.4095	0.95021	36.2562	0.94029	36.3520	0.95899
	MOPSO	40.1740	0.96930	39.2067	0.96296	38.5120	0.96340	38.2263	0.95370	38.8252	0.96450
Klasik	IHT	38.6647	0.97188	38.4089	0.97079	36.9024	0.96778	37.0930	0.95382	37.5647	0.97931
	OMP	38.7614	0.97196	38.5867	0.97144	37.2970	0.96931	37.4320	0.95567	38.4916	0.97966
	Homotopy	38.9063	0.97217	38.7223	0.97202	37.7711	0,97154	37.7459	0.95775	**39.4223**	**0.98043**
	FISTA	38.9004	0.97216	38.7213	0.97205	37.7639	0.97145	37.7142	0.95754	39.3777	0.98039
	PALM	38.9070	0.97220	38.7226	0.97208	37.8109	0.97161	37.7317	0.95761	39.1443	0.98037



Şekil 4.2. a) Orijinal *Cameraman* görüntüsü, b) görüntünün dalgacık dönüşümü, c) geri elde edilen dalgacık katsayıları, d) MOABC-SR tarafından geri elde edilen görüntü.



Şekil 4.3. Sırasıyla; KnMOPSO, IHT, OMP, NSGA II, Homotopy, FISTA, PALM ve MOEA/D algoritmaları kullanılarak geri elde edilen *Cameraman* görüntüsü.

Homotopy, FISTA ve PALM'in görüntü geri oluşturma performansları birbirine çok yakındır. PALM algoritması; *Cameraman, Lena* ve *Barbara* görüntülerinin geri oluşturulması için Homotopy ve FISTA'dan biraz daha iyidir. *Houses* ve *Peppers* görüntüleri için Homotopy algoritması FISTA ve PALM algoritmalarından daha yüksek PSNR ve SSIM'ye sahiptir. *Cameraman* ve *Lena* görüntülerinin geri oluşturulması için

MOEA/D tabanlı yöntem; MOABC-SR ve MOPSO tabanlı yöntem hariç karşılaştırılan diğer algoritmalardan daha iyi PSNR'ye sahiptir. Diğer görüntülerin geri oluşturulmasında ise KnMOPSO dışındaki algoritmalar MOEA/D tabanlı yönteme göre üstünlüğe sahiptir. NSGA II tabanlı SSR yöntemi, KnMOPSO hariç karşılaştırılan diğer algoritmalardan daha düşük görüntü geri oluşturma performansına sahiptir.

Özetle, elde edilen bu sonuçlara göre MOABC-SR, *Peppers* hariç diğer görüntülerin CS uygulandıktan sonra geri oluşturulmasında karşılaştırılan diğer algoritmalardan daha iyi performansa sahiptir.

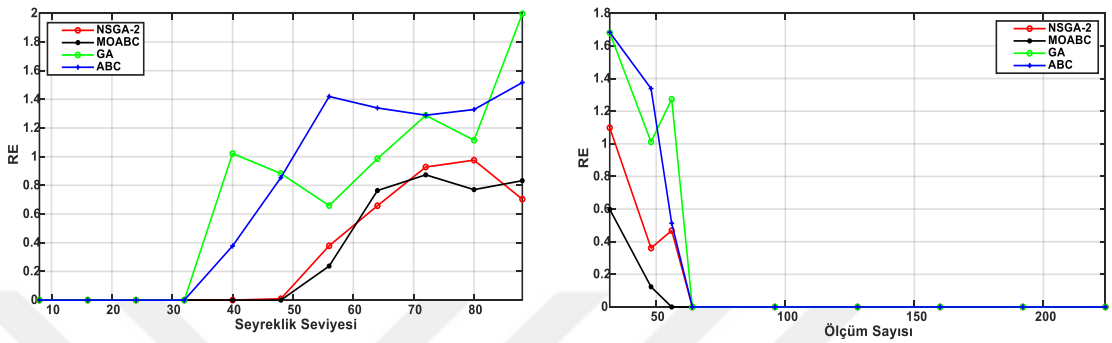
4.4. Tek Amaçlı ve Çok Amaçlı Optimizasyon Algoritmalarının SSR Probleminin Optimizasyonunda Karşılaştırılması

Çalışmanın bu bölümünde, MOABC ve NSGA-II'ye dayalı MOSR yöntemleri [115]'da önerilen tek amaç fonksiyonlu ABC algoritmasına dayalı SSR yöntemi ve [46]'da önerilen Genetik Algoritma (GA) tabanlı yöntem ile karşılaştırılmıştır. Bu algoritmaların performansları; çeşitli seyreklik değerleri ve ölçüm sayıları ile SSR problemleri kullanılarak değerlendirilmiştir. SSR algoritmalarının karşılaştırmasının ilk durumu için test sinyallerinin uzunluğu 256, ölçüm sayısı 96'dır ve seyreklik değeri 8'den 88'e kadar değişmektedir. Bu testin amacı, sinyal uzunluğu ve ölçüm sayısı sabit olmak üzere SSR algoritmalarının sinyali geri oluşturabileceği seyreklik değerini incelemektir. Seyreklik değeri yükseldikçe geri oluşturma hatalarının da yükselmesi beklenmektedir. MOABC, NSGA-II, GA ve ABC algoritmaları her bir seyreklik düzeyi için 30 kez çalıştırılmıştır. Bu bölümde karşılaştırılan algoritmaların parametreleri Tablo 4.8'de gösterilmektedir.

Tablo 4.8. Tek amaçlı optimizasyon ve çok amaçlı optimizasyon algoritmalarının karşılaştırılmasında kullanılan NSGA-II, MOABC, GA ve ABC parametreleri.

Parametre	**NSGA-II**	**MOABC**	**GA [46]**	**ABC [115]**
Popülasyon Büyüklüğü	40	40	40	40
Çaprazlama Oranı	0.7	-	0.7	-
Mutasyon Oranı	0.4	-	0.05	-
Limit Değeri	-	80	-	2500
Maksimum Döngü Sayısı	200	200	3000	3000

Algoritmaların geri oluşturduğu sinyallerin ortalama RE değerleri Şekil 4.4a'da gösterilmektedir. Şekil 4.4a'daki her nokta bir seyreklik değerine karşılık gelmektedir. Ayrıca Tablo 4.9, bu geri oluşturulan sinyallerin ortalama RE ve MSE değerlerini göstermektedir.



a) b)

Şekil 4.4. Farklı a) seyreklik ve b) ölçüm sayılarından oluşan SSR problemlerine ait ortalama RE değerleri.

Şekil 4.4a ve Tablo 4.9'a göre, MOABC yöntemi SSR problemini seyreklik değerinin 48 olduğu noktaya kadar yüksek bir doğrulukla çözebilirken, ABC ve GA algoritmalarının kullanıldığı yöntemler seyreklik değerinin 32 olduğu noktaya kadar sinyali geri oluşturabilmektedirler. NSGA-II algoritmasının kullanıldığı yöntem ise seyreklik değerinin 40 olduğu noktaya kadar sinyali tam olarak geri elde edebilmektedir. Ayrıca seyreklik değerinin 48 olduğu noktada NSGA-II'nin elde ettiği sinyalin hata değeri GA ve ABC algoritmalarının elde ettiğinden daha küçüktür. Sinyalin seyreklik değeri arttıkça SSR probleminin çözümü de zorlaşmaktadır. Bu çalışmaya göre MOABC-SR yöntemi, NSGA-II, ABC ve GA tabanlı yöntemlerin çözebileceğinden daha yüksek seyreklik düzeyine sahip SSR problemlerini çözebilmektedir. MOABC-SR yönteminin seyrek sinyali geri oluşturabileceği seyreklik seviyesi aralığı, ABC ve GA'nın geri oluşturabileceği aralıktan daha geniştir. Çok-amaçlı ve tek-amaçlı SSR metotlarının karşılaştırılmasının ikinci durumunda, ölçüm sayısı 32'den 224'e kadar değişirken, sinyal uzunluğu ve seyreklik seviyesi sırasıyla sabit olarak 256 ve 24'tür. Bu testin amacı algoritmaların SSR performanslarının ölçüm sayısına göre nasıl değiştiğini incelemektir. Ölçüm sayısı düştükçe geri elde edilen sinyalin hatasının artması beklenmektedir. MOABC, NSGA-II, GA ve ABC algoritmaları, her ölçüm sayısı için 30 kez çalıştırılmıştır.

Tablo 4.9. Çeşitli seyreklik seviyelerine sahip test problemleri için NSGA-II, MOABC, GA ve ABC ile geri oluşturulan sinyallerin ortalama RE ve MSE değerleri.

Sinyal Uzunluğu	Ölçüm Sayısı	Seyreklik Değeri	NSGA-II		MOABC-SR		GA [46]		ABC [115]	
			RE	MSE	RE	MSE	RE	MSE	RE	MSE
256	96	16	1.0e-15	1.3e-30	**1.2e-16**	**5.8e-34**	1.1e-15	1.0e-31	1.3e-15	1.5e-31
		32	1.7e-16	4.2e-33	**1.6e-16**	**2.4e-33**	1.2e-15	2.0e-31	1.9e-15	5.2e-31
		40	**5.5e-15**	**5.1e-29**	5.7e-14	7.9e-28	1.0221	0.1205	0.3779	0.0345
		48	0.0093	4.2e-04	**2.8e-12**	**3.1e-24**	0.8824	0.1862	0.8540	0.1628
		56	0.3789	0.0361	**0.2373**	**0.0166**	0.6589	0.1100	1.4196	0.5106
		64	**0.6580**	0.1488	0.7637	**0.1468**	0.9861	0.2788	1.3397	0.5146
		72	0.9280	0.2265	**0.8735**	**0.1598**	1.2878	0.4930	1.2890	0.4939
		80	0.9758	0.2628	**0.7702**	**0.2620**	1.1149	0.4123	1.3288	0.3886
		88	**0.7036**	**0.1783**	0.8327	0.2547	1.9964	1.5833	1.5169	0.9141

Şekil 4.4b, algoritmaların ortalama RE değerlerini göstermektedir ve Şekil 4.4b'deki her nokta bir ölçüm sayısına karşılık gelmektedir. Ayrıca Tablo 4.10, bu test problemleri için SSR algoritmalarına ait ortalama RE ve MSE değerlerini göstermektedir. Şekil 4.4b ve Tablo 4.10'a göre, MOABC yöntemi, ölçüm sayısının 56 olduğu noktadan itibaren seyrek sinyalleri yüksek doğrulukla geri elde edebilirken, NSGA-II, ABC ve GA'ya dayalı metotlar ölçüm sayısının 64 olduğu noktadan itibaren sinyali geri oluşturabilmektedirler. Bu çalışmaya göre MOABC-SR; NSGA-II, ABC ve GA'nın sinyali geri oluşturabildiği ölçüm sayısından daha düşük ölçüm sayısıyla geri oluşturabilmektedir.

Tablo 4.10. Çeşitli ölçüm sayılarına sahip test problemleri için NSGA-II, MOABC, GA ve ABC ile geri oluşturulan sinyallerin ortalama RE ve MSE değerleri.

Sinyal Uzunluğu	Seyreklik Değeri	Ölçüm Sayısı	NSGA-II		MOABC-SR		GA [46]		ABC [115]	
			RE	MSE	RE	MSE	RE	MSE	RE	MSE
256	24	32	1.0984	0.0838	**0.5986**	**0.0306**	1.6804	0.3239	1.68327	0.3249
		48	0.3610	0.0127	**0.1244**	**0.0020**	1.0116	0.1173	1.3386	0.2055
		56	0.4682	0.0322	**3.1e-16**	**6.8e-33**	1.2733	0.1859	0.5129	0.0301
		64	**1.9e-16**	3.9e-33	2.2e-16	**3.6e-33**	1.3e-15	1.3e-31	1.6e-15	3.0e-31
		96	2.9e-15	1.4e-29	**2.0e-16**	**3.3e-33**	1.4e-15	1.0e-31	2.0e-15	4.8e-31
		128	1.1e-16	1.3e-33	**8.9e-17**	**8.3e-34**	1.3e-15	9.6e-32	2.0e-15	4.5e-31
		160	**3.5e-17**	**9.7e-35**	1.1e-16	1.5e-33	1.4e-15	2.3e-31	2.5e-15	7.3e-31
		192	1.2e-16	1.1e-33	**1.2e-16**	**1.0e-33**	1.1e-15	1.5e-31	1.6e-15	3.0e-31
		224	1.4e-16	1.7e-33	**8.8e-17**	**9.0e-34**	2.2e-15	4.8e-31	1.0e-15	1.3e-31

4.5. EKG Sinyalinin Geri Oluşturması

EKG veri sıkıştırma işlemi uzun süreli kaydedilmiş olan kalp sinyallerinden anlamlı bilgi elde etmek için gereklidir. Ayrıca basit görevleri yerine getiren ve enerji açısından verimli olması gereken giyilebilir EKG kaydeden cihazlar da bulunmaktadır. Bazı uygulamalarda ise EKG verileri bir haberleşme kanalı üzerinden iletilir. Dolayısıyla EKG veri sıkıştırma, bant genişliğinin ve depolama kapasitesinin verimli kullanımı için gerekli bir işlemdir. Literatürde önerilen çok sayıda EKG veri sıkıştırma tekniği bulunmaktadır [116-118]. Son yıllarda EKG veri sıkıştırmasında CS yöntemleri de kullanılmaya başlanmıştır. EKG veri sıkıştırmasında CS yönteminin kullanılması ile hesaplama yükü kodlayıcıdan kod

çözücüye aktarılmış olur ve böylece kodlayıcı için donanım uygulamalarının daha basit olmasını sağlar. Ayrıca, CS ile EKG sıkıştırılmasında dalgacık bölgesindeki en büyük katsayıların konumunun kodlanmasına gerek yoktur [119]. Bu çalışmada EKG sinyaline CS yöntemi uygulanmış ve MOSR yöntemleriyle geri oluşturulmuştur [120]. Farklı özelliklerdeki EKG sinyalleri üzerinde test edebilmek için, MIT–BIH Aritmi veri tabanından 100, 101, 103, 117 ve 207 numaralı 5 farklı örnek kullanılmıştır [121]. Bu örnekteki EKG sinyallerinin uzunluğu 3328 olarak alınmıştır. Test sinyallerini seyrek olarak ifade etmek için dalgacık dönüşümü uygulanmış ve dalgacık katsayıları elde edilmiştir. Elde edilen seyrek EKG katsayılarına $M/N = 0{,}7$ olacak şekilde CS metodu uygulanmıştır. Sıkıştırılan EKG sinyal katsayıları, CS geri oluşturma yöntemleri kullanılarak tekrar geri elde edilmiştir. CS geri oluşturma yöntemi olarak MOSR yöntemleri kullanılmıştır. Kullanılan MOSR yöntemleri; NSGA-2, MOEA/D, MOPSO ve MOABC algoritmalarına dayalı olan yöntemlerdir. Algoritmaların SSR performanslarını değerlendirmek için yüzde ortalama kare farkı (Percentage root mean square difference, PRD) ve SNR kullanılmıştır. PRD ve SNR değerleri sırasıyla Denklem (4.5) ve Denklem (4.6) kullanılarak elde edilmiştir.

$$PRD = 100 \times \sqrt{\frac{\sum_{i=0}^{N-1}(\boldsymbol{x}(n)-\tilde{\boldsymbol{x}}(n))^2}{\sum_{i=0}^{N-1}\boldsymbol{x}^2(n)}} \tag{4.5}$$

$$SNR = -20\,log_{10}(PRD/100) \tag{4.6}$$

EKG sinyalinin geri elde edilmesi işleminin performans verileri Tablo 4.11'deki gibi değerlendirilip yorumlanabilir [122, 123].

Tablo 4.11. Elde edilen performans sonuçlarını değerlendirme kriterleri [123].

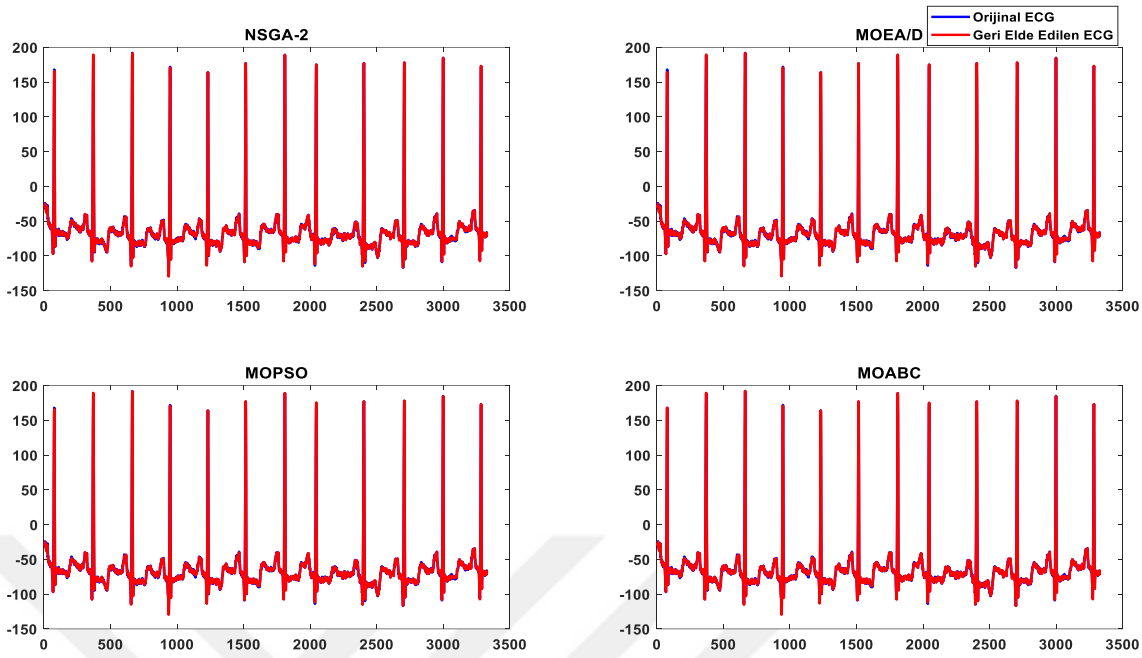
Sonuç	*Çok iyi*	*İyi*	*Kararsız*
PRD	0<PRD<%2	%2<PRD<%9	PRD≥%9
SNR	SNR>33 dB	20 dB<SNR<33 dB	SNR≤20 dB

MOSR metotları tarafından gerçekleştirilen geri elde etme işlemlerine ait performans değerleri Tablo 4.12'de gösterilmektedir.

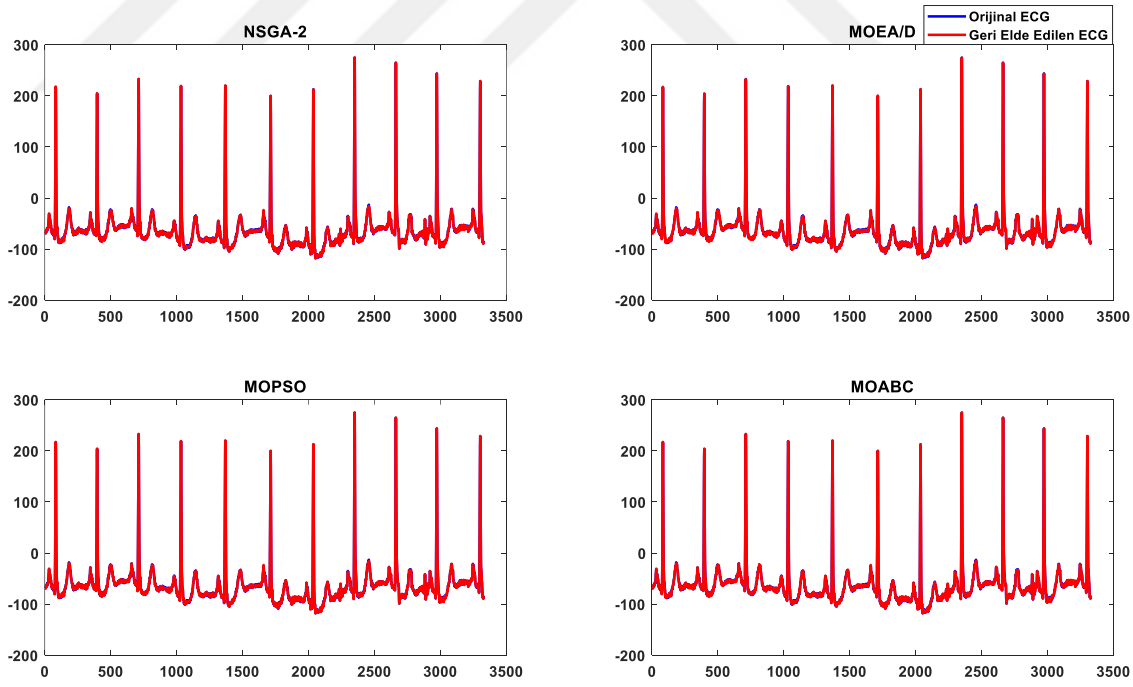
Tablo 4.12. Geri elde edilen EKG sinyallerinin PRD ve SNR değerleri

	PRD				**SNR (dB)**			
Örnek Numarası	**MOABC**	**NSGA II**	**MOPSO**	**MOEA/D**	**MOABC**	**NSGA II**	**MOPSO**	**MOEA/D**
100	**1.3003**	1.4138	1.4146	1.4539	**37.7185**	36.9925	36.9874	36.7494
101	**1.2970**	1.3842	1.3671	1.4078	**37.7407**	37.1760	37.2837	37.0291
103	1.3908	1.3908	1.3908	1.4585	37.1345	37.1345	37.1345	36.7271
117	**0.5663**	0.6261	0.6261	1.5609	**44.9375**	44.0668	44.0668	36.1326
207	**1.5315**	1.5794	1.5341	1.5732	**36.2974**	36.0304	36.2830	36.0643

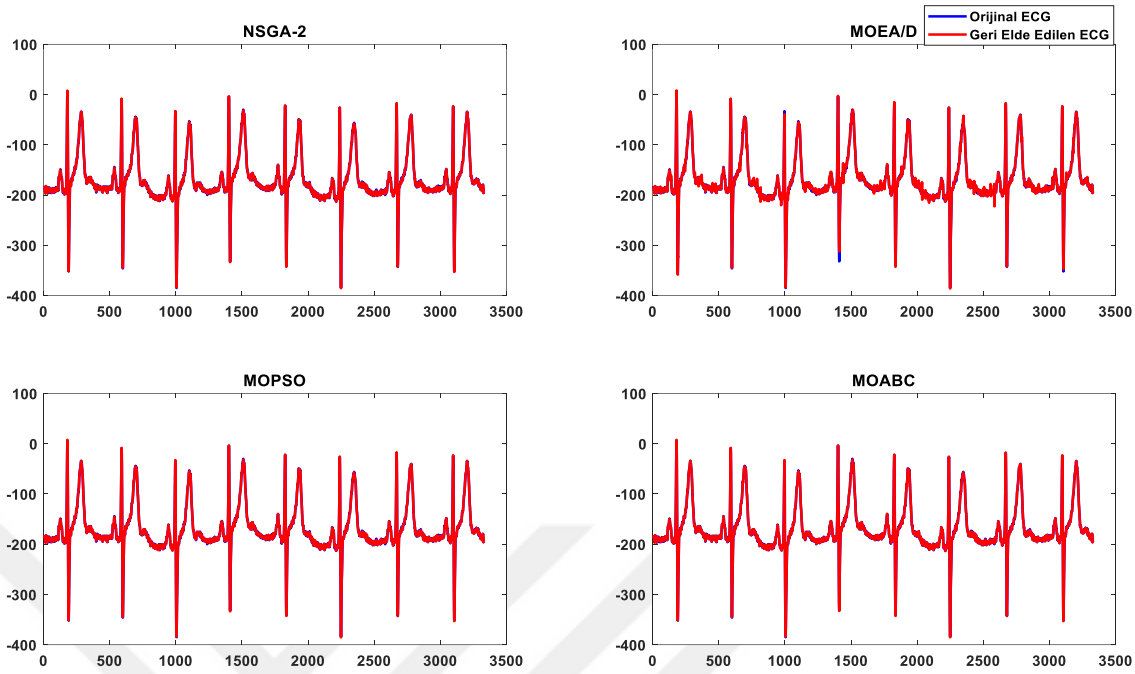
Tablo 4.11 ve Tablo 4.12'ye göre, EKG geri oluşturma performansı dört algoritma için de '*çok iyi'* olarak yorumlanabilir çünkü PRD değerleri 0 ile 2 arasında ve SNR değerleri ise 33dB'den büyüktür. Ayrıca PRD ve SNR verileri değerlendirildiğinde; 100, 101, 117 ve 207 numaralı EKG örnekleri için MOABC'nin SSR performansı diğer algoritmalardan daha iyidir. 103 numaralı örnek için ise MOABC, NSGA II ve MOPSO algoritmalarına dayalı yöntemlerin SSR performansları aynıdır. Bu SSR işlemlerine ait orijinal EKG ve geri elde edilmiş EKG sinyalleri 100, 101, 103, 117 ve 207 numaralı örnekler için sırasıyla Şekil 4.5-4.9'da gösterilmektedir. Belirtilen şekillerde görüldüğü gibi orijinal sinyalle geri elde edilen sinyal birbirine oldukça yakındır.



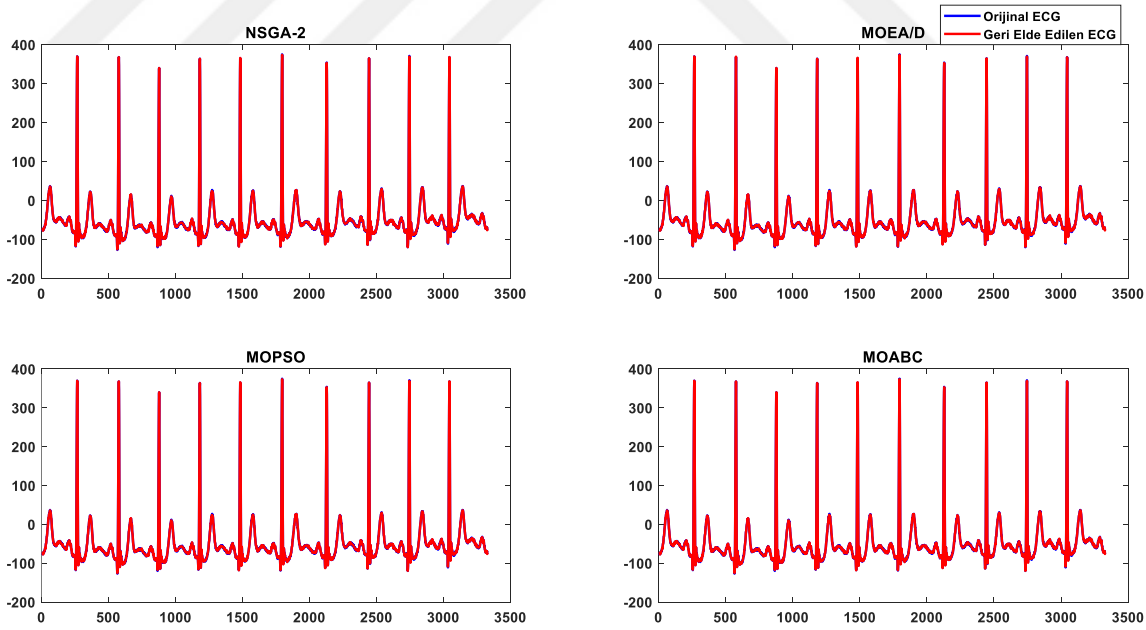
Şekil 4.5. MIT-BIH-100 EKG Orijinal ve Geri Oluşturulan Sinyal



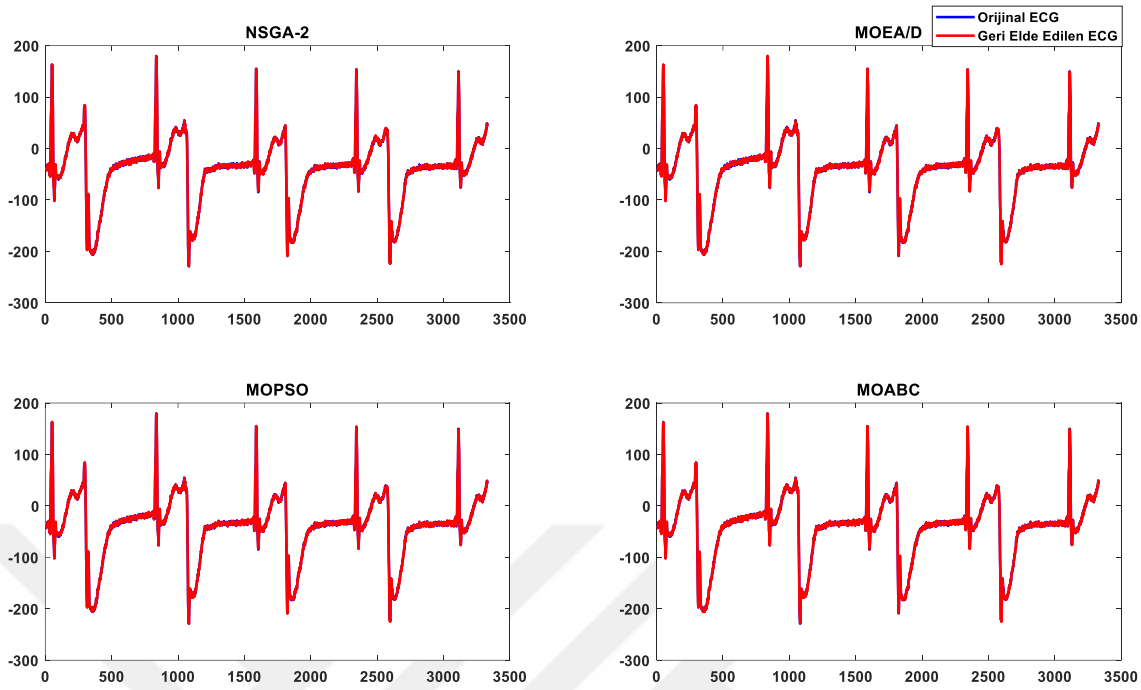
Şekil 4.6. MIT-BIH-101 EKG Orijinal ve Geri Oluşturulan Sinyal



Şekil 4.7. MIT-BIH-103 EKG Orijinal ve Geri Oluşturulan Sinyal



Şekil 4.8. MIT-BIH-117 EKG Orijinal ve Geri Oluşturulan Sinyal



Şekil 4.9. MIT-BIH-207 EKG Orijinal ve Geri Oluşturulan Sinyal

Tez çalışmasının bu bölümünde, önerilen MOSR algoritmalarının SSR performansları çeşitli problemlerle test edilmiştir. Bu test problemleri pratik uygulamalar için uygundur. Ayrıca bu algoritmalar literatürdeki çeşitli SSR algoritmalarıyla karşılaştırılmıştır. IHalfT ve OMP gibi bazı algoritmalar için seyreklik seviyesi bilgisinin SSR işleminden önce bilinmesi gerekirken, MOSR metotları seyreklik seviyesini ve ölçüm hatalarını aynı anda optimize eder ve seyreklik seviyesinin önceden bilinmesine gerek yoktur. Elde edilen test sonuçları değerlendirildiğinde SOTF, görüntü ve EKG sinyallerinin geri oluşturulmasında MOSR algoritmaları etkili bir yere sahiptir. Ayrıca MOSR yöntemlerinin sinyali geri elde edebildiği seyreklik seviyesi ve ölçüm sayısı aralığı tek amaçlı optimizasyon algoritmalarından daha geniştir. Bu bölümde çeşitli test problemlerine ait RE, ME, RSE, SNR, PRD değerleri incelendiğinde önerilen yöntemin karşılaştırılan diğer yöntemlerden daha iyi seyrek geri oluşturma performansına sahip olduğu görülmektedir.

5. BÖLÜM

OMP ALGORİTMASININ FPGA ÜZERİNDE UYGULANMASI

Tez çalışmasının bu bölümünde OMP algoritması Altera Quartus II programı üzerinde VHDL dili ve şematik kullanılarak tasarlanmıştır. Tasarlanan OMP algoritması, uzunluğu 128, seyreklik seviyesi 5 olan sinyali 32 adet ölçüm kullanarak geri oluşturabilmektedir. Temel olarak tasarım; kontrol ünitesi ve veri yolu olmak üzere iki kısımdan oluşmaktadır. Kontrol birimi, Sonlu Durum Makinesi (Finite State Machine, FSM) kullanılarak tasarlanmıştır. Genel olarak kontrol birimi algoritmanın içindeki veri yapılarının durumlarını, veri akışını, işlemlerin ne zaman ve hangi durumda gerçekleşeceğini belirler. Hafızaların okuma yazma durumları, sayıcıların aktif olma, çarpma, bölme, toplama işlemlerinin aktif olma zamanlarının belirlenmesi gibi durumların kontrol edilmesini sağlar. İşlem yapılacak olan sayıların bulunduğu hafıza birimi okuma durumuna getirilerek matris ve vektörler üzerinde işlemler yapılmaktadır. İşlemlerin paralelliği, yapılan işlemlerin verimliliği, veri yolundaki yapıların birbiri ile koordineli ve verimli çalışması kontrol biriminde yapılan tasarıma büyük oranda bağlıdır. Veri yolunda ise gerekli olan işlemlerin probleme göre tasarlanmış birimleri yer almaktadır. Buradaki birimler temel olarak 32 adet paralel dört işlem, karşılaştırıcı, hafızalar, hafızadaki adreslerin yerini tutan sayıcılardır. Tasarım Cyclone FPGA ailesi üzerinde test edilmiştir. Sayısal tasarımda paralel işlem ile kullanılan lojik elemanları sayısı ve sistem karmaşıklığı arasında doğru orantı varken paralel işlem sayesinde tasarımın saat çevrim sayısı azalır ve algoritmanın çalışma süresi kısalır. Ancak aynı zamanda sistem karmaşıklığı ve kullanılan lojik elemanları sayısı artar. Literatürdeki çalışmalar bu ikisini optimum seviyede tutarak, kullanılan kartın özelliklerine ve uygulamanın gereksinimine göre en uygun tasarımı bulmaya yöneliktir. Bu tasarımda birbirine bağlı olmayan işlemler aynı anda yapılarak algoritmanın çalışma süresinde optimizasyon yapılmıştır. Bir sonlu durumda bir sonraki adım olan iki vektörün çarpma işlemi yapılırken bir önceki işlemin sonucunun hafızaya kaydedilmesi veya bir önceki işlemin sonucu kaydedilirken ondan

sonraki işlem için hafızaların okuma durumuna getirilmesi sağlanarak zamandan tasarruf edilmesi bu optimizasyonlara örnek olarak verilebilir. Bu çalışmada gerçekleştirilen tasarım literatürdeki benzer çalışmalarla değerlendirilmiş ve bazı çalışmalardaki tasarımlardan üstün olduğu belirlenmiştir.

5.1. Giriş

CS, belirli koşullar altında bir sinyali kendi büyüklüğünden daha az örnek sayısıyla geri elde edebilen bir sinyal işleme metodudur. CS, öncelikle seyrek sinyali belirli koşullara göre örnekleyip ardından bu örnekleri kullanarak seyrek sinyali geri oluşturur. SSR yöntemlerinin kullanıldığı bu geri oluşturulma işlemi CS'nin önemli aşamalarından biridir. Belirli bir sürede daha az örnek sayısı kullanarak bir sinyali geri oluşturmak, SSR algoritmaları için çok fazla hesaplama gücü ve kaynağı gerektirir. Dolayısıyla verimli bir SSR algoritması tasarlamak önemlidir. OMP algoritması yaygın olarak kullanılan SSR yöntemlerinden biridir. Algoritmanın hesaplama karmaşıklığı büyük oranda ölçüm matrisinin boyutuna ve sinyalin seyrekliğine bağlıdır. Hesaplama karmaşıklığı ve SSR performansı göz önünde bulundurulduğunda gerçek zamanlı SSR optimizasyonunda donanım açısından uygun bir yöntemdir. Literatürdeki donanım uygulamaları geri oluşturma süresi ve kaynak kullanımı için optimum performans elde etmeyi amaçlamaktadırlar. Önerilen mimariler sinyalin özelliklerine göre, kullanılan algoritmalara ve hedef cihaza göre farklılık gösterirler. Ölçüm matrisinin boyutu, sinyalin boyutu ve seyrekliği, devrenin performansını belirleyen önemli parametrelerdir. Son zamanlarda, literatürde OMP algoritması için çeşitli donanım mimarileri önerilmiştir. OMP algoritmasının ilk donanım uygulamalarından biri Septimus ve ark. tarafından Donanım Tanımlama Dili (Hardware Description Language, HDL) kullanılarak tasarlanmıştır [124]. Belirtilen çalışmada, verimli bir donanım uygulaması elde etmek için modifiye edilmiş Gram-Schmidt metodunun kullanımı tercih edilmiş ve donanım karmaşıklığını azaltmak için karekök kullanılmamıştır. Tasarımda VHDL dili kullanılmış ve Xilinx Virtex 5 FPGA üzerinde gerçekleştirilmiştir. Tasarlanan OMP algoritmasının donanımı; sinyal uzunluğu $N = 128$, ölçüm sayısı $M = 32$ ve seyreklik seviyesi $k = 5$ olan SSR problemine göre tasarlanmıştır. Tasarım 930 saat çeviriminden oluşmaktadır ve saat frekansı 39 Mhz'dir. Stanislaus ve ark. yaptığı bir çalışmada QR ayrıştırma yöntemine dayalı OMP algoritması kullanılmıştır [125]. Belirtilen çalışmada tasarlanan

OMP algoritmasının donanımı; sinyal uzunluğu $N = 256$, ölçüm sayısı $M = 64$ ve seyreklik seviyesi $k = 8$ olan SSR problemini optimize edecek şekilde tasarlanmıştır. Tasarım 85 MHz ve 69 MHz olarak iki farklı saat frekansında çalışmaktadır ve Xilinx Virtex-5 FPGA ailesi üzerinde %15 dilim ve %80 DSP çekirdeğinden oluşan bir alanı kapladığı belirtilmiştir.

Rabah ve ark., Newton-Raphson yinelemesine dayalı matrisin tersinin alınması işlemini geliştirerek ve matris-vektör çarpımları için donanımı yeniden kullanarak OMP algoritmasının çalışma süresinin azaltıldığı bir çalışma önermiştir [126]. Önerilen tasarım Simulink kullanılarak modellenmiş ve Xilinx Sistem Üreteci (Xilinx Sytem Generator, XSG) tasarım aracı kullanılarak FPGA üzerinde gerçekleştirilmiştir. Önerilen mimari; seyreklik seviyesi 36, sinyal uzunluğu 1024 olan bir sinyali 256 adet ölçüm kullanarak geri oluşturmaktadır. Polat ve Kayhan HDL kullanarak standart OMP algoritmasının sinyal boyutlarına göre ayarlanabilir bir tasarımını yapmışlardır [127]. Önerilen mimari, Verilog HDL ile tasarlanmış ve Xilinx ailesinden Virtex FPGA'lar; Altera ailesinden Cyclone ve Stratix FPGA'lar için sentezlenmiştir. Tasarım, seyrekliği 5 olan 128 uzunluklu sinyali 7.75 µs içinde geri elde edebilirken; uzunluğu 256, seyrekliği 8 ve 12 olan sinyalleri sırasıyla 23.27 ve 39.56 µs içinde geri elde edebilmektedir. Ge ve ark. karekök içermeyen QR ayrışımına dayalı bir OMP algoritmasının Xilinx Kintex-7 FPGA üzerinde tasarımını yapmışlardır [128]. Çalışmada elde edilen sonuçlar, önerilen mimarinin uzunluğu 1024 ve seyrekliği 36 olan bir sinyali 238 µs sürede geri oluşturabilirken, maksimum çalışma frekansı 210 MHz'dir.

Bu çalışmada ise VHDL dili ve şematik kullanılarak OMP algoritmasının tasarımı yapılmıştır. Bu tasarımda en küçük kareler çözümü için QR ayrışma metodu kullanılmıştır. Ayrıca Öklid normunun hesaplanmasında ise Ge ve ark. çalışmasında yapmış olduğu gibi karekök kullanılmamıştır [128]. Sıfırdan farklı olan indekslerin bulunması işlemi sabit nokta formatına göre, matrisin tersinin alınması işlemi ise kayar nokta formatına göre tasarlanmıştır. Tasarım Intel Cyclone FPGA ailesi üzerinde gerçekleştirilmiştir. Bu çalışmanın ikinci kısmında genel OMP algoritması ve donanım tasarımı yapılan OMP algoritmasının temel prensipleri açıklanmıştır. Daha sonra tasarımın detayları açıklanmıştır. Yapılan tasarımının değerlendirilmesi ve karşılaştırılması dördüncü kısımda gerçekleştirilmiştir. Son olarak; yapılan çalışma özetlenmiş ve gelecek planı açıklanmıştır.

5.2. OMP Algoritması

MP algoritması, sinyal ölçümleri ile ölçüm matrisinin en çok ilintili olan sütununu gelişmiş bir sinyal tahmini kullanarak bulmaya çalışan basit bir güncelleme yöntemine sahiptir [129]. MP düşük hesaplama karmaşıklığına sahip olsa da yineleme sayısı seyreklik seviyesine büyük ölçüde bağlıdır. Tropp ve Gilbert tarafından geliştirilen bir açgözlü algoritma olan OMP algoritmasında ise MP algoritmasından farklı olarak yineleme sayısı seyreklik seviyesi kadar sürmektedir [49]. MP algoritmasında en küçük kareler problemi en sonda yer alırken OMP metodunda her yinelemede yer almaktadır. Ancak OMP algoritması genel olarak daha iyi SSR performansına sahiptir. OMP algoritmasında öncelikle gözlem vektörü ile en çok ilintili olan ölçüm matrisinin sütun vektörü seçilir ve her yinelemede seçilen sütunun katkısı ilk başta gözlem vektörüne eşit olan *kalan* (residue) vektörden çıkarılır. *Kalan* vektör sıfır veya belli bir değerin altına düştüğü zaman doğru sütun kümesi bulunmuştur. OMP algoritması k yineleme sürmektedir ve her yinelemede bir indis değeri bulunmaktadır. Son olarak da bu k tane indisler en küçük kareler yaklaşımıyla beraber kullanılarak seyrek $\boldsymbol{x}$ sinyali geri oluşturulur. Standart OMP algoritmasının adımları Algoritma 5.1'deki gibidir [49].

Standart OMP algoritmasında, Adım 1'de ölçüm matrisinin sütun vektörleri ile *kalan* vektörü çarpılır. İlk yinelemede kalan vektörü gözlem vektörüne eşittir. Sütun vektörlerine ait olan vektör iç çarpımlarının mutlak değerleri alınarak bunların içinden maksimum olanı bulunur. Her yinelemede, maksimum mutlak değeri sağlayan ölçüm matrisinin sütun vektörünün indisi bulunur. Bulunan bu indis Adım 2'de diğer yinelemelerde bulunan indislere eklenerek maksimum değeri sağlayan sütun vektörlerinin indis kümesi oluşturulur.

Algoritma 5.1

Girişler:

- $\boldsymbol{A}$: $M \times N$ boyutlu ölçüm matrisi.
- k: x sinyalinin seyreklik seviyesi.
- $\boldsymbol{y}$: $M \times 1$ boyutlu gözlem vektörü.

Çıkışlar:

- Λ_K: sıfırdan farklı elemanların indislerinin kümesi
- $\boldsymbol{A}_K$: ölçüm matrisinin seçilmiş sütun matrisleri.

- $\boldsymbol{a}_k$: yaklaşık olarak hesaplanan gözlem vektörü.
- $\boldsymbol{r}_k = y - a_k$: *kalan* vektör

Başlangıç Değerleri:

- $\boldsymbol{r}_0 = \boldsymbol{y}$
- $\Lambda_K = \emptyset$
- $j = 1$ İterasyon sayacı
- $\boldsymbol{A}_0 = \emptyset$

Not: $\boldsymbol{\varphi}_{t=1,\ldots,N}$, ölçüm matrisinin sütun vektörü

Adımlar:

Adım 1: Denklem (5.1)'deki optimizasyon problemini çözen λ_j indislerini bul. λ_j, A matrisinin $\boldsymbol{r}_{j-1}$ ile en çok ilintili olan sütunudur.

$$\lambda_j = argmax_{t=1..N} |\langle r_{j-1}, \varphi_t \rangle| \tag{5.1}$$

Adım 2: İndis kümesinin güncellenmesi.

$$\Lambda_K = \Lambda_K \cup \{\lambda_j\} \tag{5.2}$$

Adım 3: Seçilen sütunların eklenmesi.

$$\boldsymbol{A}_j = [\boldsymbol{A}_{j-1} \quad \boldsymbol{\varphi}_{\lambda_j}] \tag{5.3}$$

Adım 4: Denklem (5.4)'teki en küçük kareler probleminin çözülmesi.

$$\boldsymbol{x}_j = argmin_{x'} ||\boldsymbol{y} - \boldsymbol{A}_j \boldsymbol{x}'||_2 \tag{5.4}$$

Adım 5: Yaklaşık gözlem vektörünün hesaplanması.

$$\boldsymbol{a}_j = \boldsymbol{A}_j \times \boldsymbol{x}_j \tag{5.5}$$

Adım 6: *Kalan* vektörünün güncellenmesi.

$$r_j = \boldsymbol{y} - \boldsymbol{a}_j \tag{5.6}$$

Adım 7: Eğer $j < K$ ise j'yi artır ve yeni döngü için Adım 1'e git.

Her yinelemede nokta çarpımların mutlak değerlerinden maksimum olanı sağlayan sütun vektörü de seçilerek bir matris oluşturulması Adım 3'te gerçekleştirilir. Adım 4'te ise oluşturulan bu matris ve gözlem vektörü ile en küçük kareler yaklaşımı kullanılarak tahmini bir $\boldsymbol{x}$ sinyali hesaplanır. Hesaplanan $\boldsymbol{x}$ sinyali ve ölçüm matrisi kullanılarak tahmini bir gözlem vektörü oluşturulur. Oluşturulan bu gözlem vektörü Adım 6'da gerçek gözlem vektöründen çıkarılarak ilk adımda gerçek gözlem vektörüne eşit olan *kalan* vektörü güncellenmiş olur. Toplam yineleme sayısı seyreklik seviyesine eşit oluncaya

kadar bu adımlar devam eder. En son döngüde Adım 4'te elde edilen $\boldsymbol{x}$ sinyali geri oluşturulmuş sinyaldir.

Bu çalışmadaki OMP algoritmasının uygulanmasında, standart OMP algoritmasındaki en küçük kareler probleminin çözümü için QR ayrışım metodu kullanılmıştır. QR ayrışımına dayalı OMP, Algoritma 5.2'de verilmiştir. Algoritma 5.1 ve Algoritma 5.2'nin giriş çıkış sinyalleri aynıdır.

Algoritma 5.2: QR Ayrışımına Dayalı OMP Algoritması

FOR (j=1 to k)

Adım 1: Denklem (5.1)'de ölçüm matrisinin sütun vektörlerinin *kalan* vektör ile nokta çarpımları yapılmaktadır. Nokta çarpımların mutlak değeri maksimum olmasını sağlayan sütun vektörünün indisi seçilmektedir.

Adım 2: Denklem (5.2)'deki indis kümesi güncellenir ve bir önceki adımda seçilen indis numarasına denk gelen sütun vektörü Denklem (5.7)'deki gibi $\boldsymbol{C}$ vektörüne atanır.

$$\boldsymbol{C}_{(:,j)} = \boldsymbol{A}_{(:,\lambda_j)} \tag{5.7}$$

Adım 3: Aşağıdaki döngünün içinde QR ayrışımı için $\boldsymbol{R}$ matrisinin köşegen olmayan elemanları Denklem (5.8) kullanılarak hesaplanmaktadır ve ölçüm matrisinin seçilen sütunu Denklem (5.9)'daki gibi güncellenmektedir.

FOR (i=1 to j-1)

IF (j>1)

$$\boldsymbol{R}_{(i,j)} = \boldsymbol{Q}_{(:,i)}^{T} \times \boldsymbol{C}_{(:,j)} \tag{5.8}$$

$$\boldsymbol{C}_{(:,j)} = \boldsymbol{C}_{(:,j)} - \boldsymbol{R}_{(i,j)} \times \boldsymbol{Q}_{(:,i)} \tag{5.9}$$

END IF

END FOR

Adım 4: Seçilen vektörlerin normları $\boldsymbol{R}$ matrisinin köşegen elemanlarını oluşturmaktadır.

$$\boldsymbol{R}_{(j,j)} = ||\boldsymbol{C}_{(:,j)}||_2 = \sqrt{(\boldsymbol{C}_{(1,j)}{}^2 + \boldsymbol{C}_{(2,j)}{}^2 + \cdots \ldots + \boldsymbol{C}_{(M,j)}{}^2)} \tag{5.10}$$

Adım 5: $\boldsymbol{Q}$ matrisi Adım 2'de ölçüm matrisinden seçilen sütun vektörlerinin birleşiminden oluşur.

$$\boldsymbol{Q}_{(:,j)} = \boldsymbol{C}_{(:,j)} / \boldsymbol{R}_{(j,j)} \tag{5.11}$$

Adım 6: *Kalan* vektörü Denklem (5.12) kullanılarak güncellenir.

$$\boldsymbol{r}_j = \boldsymbol{r}_j - \boldsymbol{Q}_{(:,j)} \times (\boldsymbol{Q}_{(:,j)}^T \boldsymbol{r}_j) \quad (5.12)$$

END FOR

Adım 7: $\boldsymbol{R}$ matrisinin tersi aşağıdaki döngünün içinde yer alan denklemler kullanılarak hesaplanmaktadır.

FOR (j=1 to k)

$$\boldsymbol{Rinv}_{(j,j)} = \frac{1}{\boldsymbol{R}_{(j,j)}} \quad (5.13)$$

IF (j>1)

FOR (i=1 to j-1)

$$\boldsymbol{Rinv}_{(i,j)} = -\boldsymbol{Rinv}_{(j,j)} \times (\boldsymbol{Rinv}_{(i,1..j-1)} \times \boldsymbol{R}_{(1..j-1,j)}) \quad (5.14)$$

END FOR

END IF

END FOR

Adım 8: Geri elde edilen sinyalin sıfırdan farklı olan değerleri Denklem (5.15) kullanılarak hesaplanmaktadır.

$$\boldsymbol{x}' = \boldsymbol{Rinv} \times \boldsymbol{Q}^T \times \boldsymbol{y} \quad (5.15)$$

Adım 9: Denklem (5.15)'de elde edilen sıfırdan farklı olan değerlerin, Adım 2'de bulunan indislere yerleştirilmesiyle $\boldsymbol{x}$ sinyali geri oluşturulmuş olur.

$$\boldsymbol{x}_{\Lambda_K} = \boldsymbol{x}' \quad (5.16)$$

QR ayrışım metodunda yer alan Öklid normunun hesaplanması işleminde karekök işlemi yer almaktadır. Algoritma 5.2'nin 4. Adımında yer alan Denklem (5.10)'da, Öklid normunun hesaplanmasında karekök alınması gerekmektedir. Karekök işleminin donanımda çalışma süresi diğer işlemlere göre daha uzundur ve lojik elemanlarının sayısı açısından değerlendirildiğinde çok yer kaplamaktadır. Ayrıca sabit noktalı tasarımda karekök işleminden sonra sonucun doğruluğu da ondalık kısmını temsil eden bit sayısına bağlıdır. Bit sayısı azaldığında kesinlik azalır ve geri oluşturulan sinyalin hata oranı artabilir. Donanımda bu dezavantajları gidermek için OMP algoritmasının sayısal tasarımı için önerilen bazı çalışmalarda OMP algoritması karekök kullanılmadan

tasarlanmıştır [128], [130]. Karekök kullanmadan yapılan bu tasarımlardan biri Algoritma 5.2'deki Denklem (5.9) - (5.12)'leri sırasıyla Denklem (5.17) - (5.20)'deki gibi değiştirerek Denklem (5.10)'da karekök kullanmadan QR ayrışımını gerçekleştirmektedir [128].

$$\boldsymbol{C}_{(:,j)} = \boldsymbol{C}_{(:,j)} - \boldsymbol{R}_{(i,j)}\boldsymbol{Q}_{(:,i)}/\boldsymbol{R}_{(i,i)} \tag{5.17}$$

$$\boldsymbol{R}_{(j,j)} = ||\boldsymbol{C}_{(:,j)}||_2^2 = (\boldsymbol{C}_{(1,j)}{}^2 + \boldsymbol{C}_{(2,j)}{}^2 + \cdots \ldots + \boldsymbol{C}_{(M,j)}{}^2) \tag{5.18}$$

$$\boldsymbol{Q}_{(:,j)} = \boldsymbol{C}_{(:,j)} \tag{5.19}$$

$$\boldsymbol{r}_j = \boldsymbol{r}_j - \boldsymbol{Q}_{(:,j)}(\boldsymbol{Q}_{(:,j)}^T\boldsymbol{r}_j)/\boldsymbol{R}_{(j,j)} \tag{5.20}$$

OMP algoritmasındaki bu değişim sonucunda yaklaşık olarak aynı sonuçlar elde edilmektedir. Bu değişimle alakalı teorik açıklama Ge ve ark. çalışmasında yer almaktadır [128]. Karekök kullanılmaması donanımda uzun süren bir işlemin yapılmasına gerek olmadığı için avantajlı görünmesine rağmen Algoritma 5.2'den ve Denklem (5.17)-(5.20)'den görüleceği üzere bölme işlemi sayısını artırmaktadır. Bölme işlemi de donanımda uzun süren bir işlemdir ve sabit noktalı tasarımda sonucun doğruluğu ondalık kısmını temsil eden bit sayısına bağlıdır. Tez çalışmasının bu bölümünde karekök kullanılmadan uygulanan OMP algoritmasının FPGA üzerinde tasarımı yapılmıştır. En küçük kareler yaklaşımında kareköksüz QR metodunun kullanıldığı diğer çalışmalar da vardır. Bu OMP tasarımları teorik olarak büyük oranda birbirine benzemektedirler. OMP'de yapılan değişikliklerden donanımda algoritmanın çalışma süresi açısından avantaj sağlamak için tasarımın verimli bir şekilde yapılması gerekmektedir. Önerilen metotlardaki özgünlük ve verimliliğin sağlanması takip edilen tasarım prensiplerine bağlıdır. Tasarımın paralel işlem kabiliyeti, kullanılan lojik elemanı sayısı, hesaplama birimlerinin işlem kabiliyetleri ve kullanılan FPGA tipinin özellikleri tasarımın performansını etkileyen bazı faktörlerdir.

5.3. OMP Algoritmasının FPGA Üzerinde Uygulanması

Tez çalışmasının bu bölümünde Altera Quartus II programı üzerinde VHDL dili ve şematik kullanılarak OMP algoritmasının tasarımı yapılmıştır. Temel olarak tasarım; veri

yolu ve kontrol ünitesi olmak üzere iki kısımdan oluşmaktadır. Veri yolu; hesaplama ve hafıza birimlerinden oluşurken, kontrol birimi FSM kullanılarak tasarlanmıştır.

5.3.1. Veri Yolu

5.3.1.1. Hesaplama Birimi

Hesaplama birimi OMP algoritmasında yer alan matematiksel işlemleri içeren; dört işlem, mutlak değer alma, kaydırma ve karşılaştırma işlemlerinden oluşan bir birimdir. Algoritmada vektör-skalar, vektör iç çarpımları, iç çarpımlardaki toplama işlemleri, kalan vektörü ve ölçüm matrisinin en çok ilintili sütun vektörünün güncellenmesi gibi işlemler burada gerçekleşir. Hesaplama birimi aşağıdaki alt birimlerden oluşmaktadır:

Çarpma işlemi birimi: Bu birim 24 bitlik *M* adet paralel çarpma işlemlerinden oluşmaktadır. OMP algoritmasındaki iç ve vektör-skalar çarpım işlemleri bu birimde yapılmaktadır. OMP algoritmasının donanımında en çok bulunan işlem çarpma işlemidir. Çarpma işleminin verimli bir şekilde yapılması algoritmanın geri oluşturma süresini önemli derecede etkilemektedir. Algoritmanın genel saat frekansını optimum seviyede tutmak için bu birim içinde yer alan her bir çarpma işlemi için çıkış gecikmesi optimize edilmiştir. Bu optimizasyonda çarpma işleminin gerçekleştiği saat çevrim sayısı ve genel saat frekansı algoritmanın genel çalışma süresini minimum seviyede tutacak şekilde belirlenmiştir. Eğer işlem tek saat çevrimi sürseydi sistemin genel saat çevrim süresi artacağı için çalışma frekansı düşerdi ve OMP algoritmasının geri elde etme süresi artardı. Ayrıca çarpma işlemi biriminin ölçüm matrisinin maksimum indeksli değerinin bulunması sırasında toplama işlemi ve karşılaştırıcı ile paralel çalıştığı durumlar da vardır.

Paralel toplama işlemi: Matrislerin nokta çarpımında kullanılır. Quartus II programında yer alan paralel toplama yerleşik fonksiyonundan *M* adet sayıyı paralel olarak toplayacak şekilde oluşturulmuştur. Çarpma işlemi biriminde olduğu gibi paralel toplama işleminin saat çevrim sayısı da sistemin frekansını optimum seviyede tutacak şekilde ayarlanmıştır. Eklenen saat çevrimlerinin algoritmanın genel çalışma süresine etkisi OMP algoritmasında Denklem (5.1)'de yer alan optimizasyonda $\boldsymbol{A}\times\boldsymbol{res}$ çarpma işlemiyle ve karşılaştırma işlemiyle mümkün olduğu kadar paralel çalışarak minimuma indirilmiştir.

Bölme işlemi ve çıkarma işlemi: Çarpma işlemindeki gibi M adet paralel olarak tasarlanmışlardır. *Kalan* vektörü ve seçilen sütun vektörlerinin güncellenmesinde kullanılırlar. Bölme işlemi diğer üç işleme göre daha uzun sürmektedir. Ancak algoritmanın içinde çok fazla yer almadığı için işlemin tamamlanması için geçen çevrim sayısı artırılmıştır. Toplam saat çevrim sayısı artmıştır ancak sistemin saat frekansı düşmediği için bölme işleminin OMP algoritmasının geri elde etme süresine etkisi azaltılmıştır.

Karşılaştırma Birimi: Denklem (5.1)'de yer alan $\boldsymbol{A}\times\boldsymbol{res}$ iç çarpımında her sütunun hesaplanmasından sonra toplama işleminden elde edilen sonucun mutlak değerini bir önceki maksimum değerle karşılaştırarak maksimum değeri ve ona karşılık gelen sütunun indeks değerini kaydeder. Bu birim karşılaştırma işlemi yaparken aynı saat çevriminde çarpma ve toplama işlemleri diğer sütunlar için devam etmektedir. Maksimum değer değiştiği zaman hafızadaki maksimum değer ve indeksi güncellenir. Karşılaştırma birimi, matris çarpma işlemi bittikten sonra en son elde edilen değeri de karşılaştırır ve bu işlemle birlikte en çok ilintili olan sütun bulunmuş olur.

5.3.1.2. Hafıza Birimleri

Hafıza birimleri; ölçüm matrisinin depolandığı ROM hafıza ve $\boldsymbol{Q}$, $\boldsymbol{R}$, $\boldsymbol{C}$ ve *kalan* verilerinin depolandığı RAM hafıza birimlerinden oluşur. Hafıza birimleri VHDL kullanılarak tasarlanmıştır. Hafıza birimlerinin adreslerine ulaşmak için sayıcılar kullanılmıştır. Standart olarak bir saat çevriminde bir adresteki veri okunur veya kaydedilir. OMP algoritmasında $M\times N$ boyutunda bir ölçüm matrisinin gözlem vektörüyle en çok ilintili bir sütununu bulmak için öncelikle $M\times N$ adet verinin hafızadan okunması gerekir bu da sadece okuma işleminin $M\times N$ adet saat çevrimi sürmesi demektir ve algoritmanın çalışma süresini önemli ölçüde artırmaktadır. Bu tasarımda bir hafıza adresine ölçüm sayısı kadar sayı ikilik sistemde birleştirilip tek bir sayı olarak kaydedilmiştir. Dolayısıyla ölçüm matrisinin depolandığı hafıza $M\times N$ yerine N adresten oluşmaktadır ve M adet verinin hafızadan okunması veya hafızaya kaydedilmesi M yerine 1 saat çevrimi kadar sürmektedir. $\boldsymbol{Q}$, $\boldsymbol{C}$ matrisi ve *kalan* vektörünün tutulduğu hafıza birimleri bu şekilde kullanılmıştır. Genelde birbirinden bağımsız olarak işleme girecek eleman sayısı M olduğu için hafızaların bu şekilde blok olarak kullanılması işlemlerin paralel olarak kullanılmasını kolaylaştırmaktadır. Dört işlem birimlerinin hepsi M adet

paralel işlemlerden oluşmaktadır. Hafızaların bu şekilde kullanılması işlem sırasında sayıları doğru şekilde işleme alabilmek için ayrıştırma ve tekrar birleştirerek hafızaya kaydedilmesini sağlayacak tasarım gerektirmektedir. Bu tasarımda hafızaya kaydedilmeden önce veriler tek bir veri yapısı içerisinde birleştirilmektedir ve hafıza birimlerine bu şekilde kaydedilmektedir. Hafızadan veri okunup işleneceği zaman bu veriler ayrıştırılmaktadır. Vektör iç çarpımı sonuçlarının ve *R* matrisinin kaydedildiği hafıza birimleri ise her bir sayıya bir adres gelecek şekilde normal olarak tasarlanmıştır. Çünkü bu verilerin algoritma içindeki kullanımı sırasıyla bir sütunun veya satırın elamanlarına ulaşılmasını değil, her defasında farklı bir elemanına ulaşılmasını gerektirmektedir. Dolayısıyla, bu şekilde olan matrislerin veya vektörlerin her bir elemanını farklı adreslere kaydetmek daha verimli ve ulaşması kolay olacaktır. Ancak bu durum ***R*** matrisinin kaydedildiği hafıza biriminin adresine ulaşım sağlayan sayıcı tasarımını diğerlerinden farklı ve daha karmaşık hale getirmektedir.

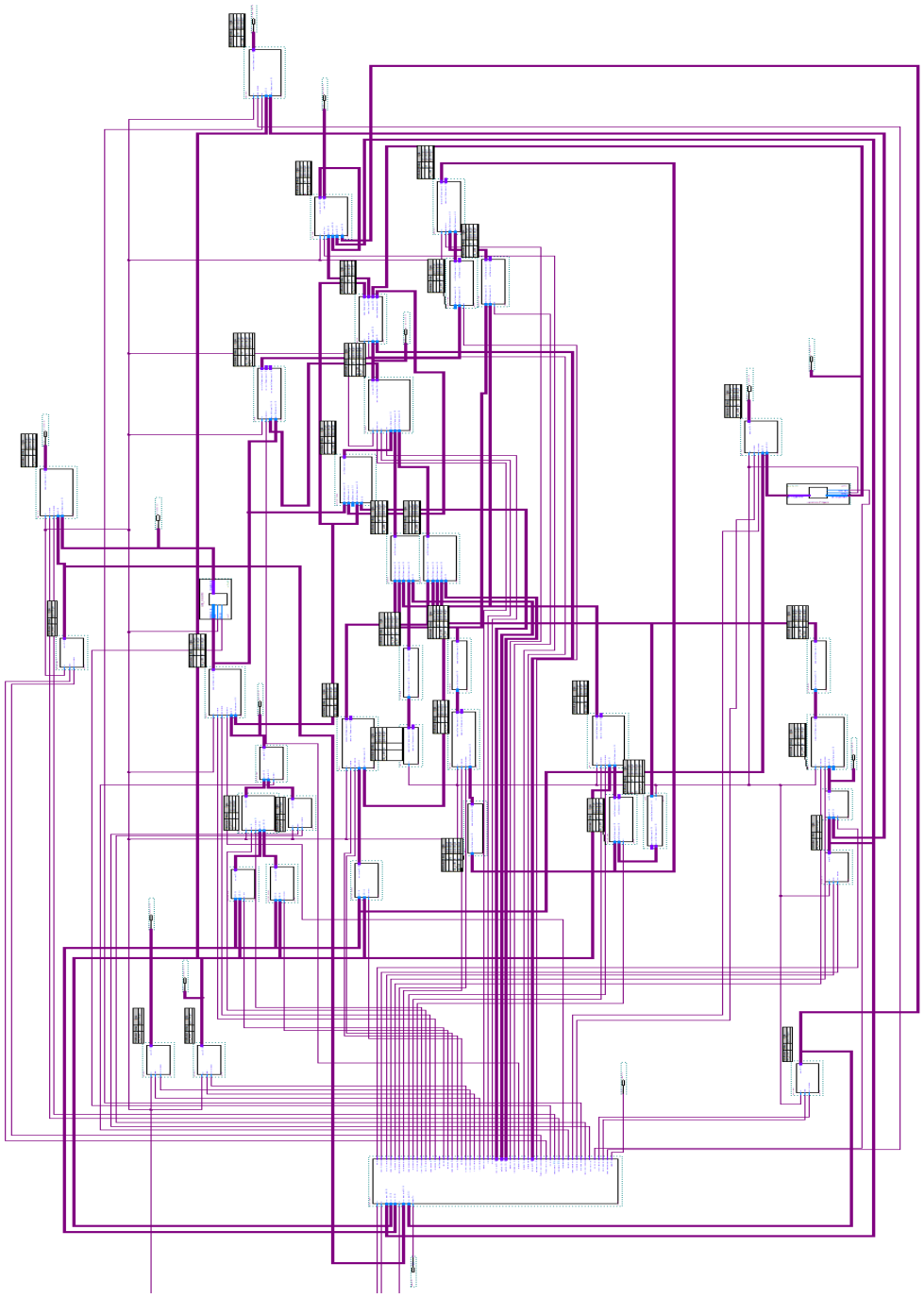
5.3.2. Kontrol Birimi

Kontrol birimi FSM kullanılarak tasarlanmıştır. Kontrol biriminin girişinde temel olarak saat, sıfırlama ve hafızaların adreslerini belirleyen sayıcıların çıkış sinyalleri yer almaktadır. Kontrol biriminin çıkışında ise hesaplama ve hafıza biriminde yer alan bütün birimlerin çalışma, sıfırlama ve seçme gibi fonksiyonlarının kontrolünü sağlayan sinyaller bulunmaktadır. Kontrol birimi her bir saat çevriminde giriş sinyallerine ve mevcut duruma göre çıkış sinyallerini ve gelecek durumu belirleyerek sistemi kontrol eder. Kontrol biriminin bir sonraki durumu ve çıkış sinyali, giriş sinyaline göre ve o andaki durumunda yapılan işleme göre belirlenir. Genellikle durumlar hafızadan veri okuma, gerekli işlemlerin aktif edilmesi, işlemlerden sonra verinin hazır olması için beklenmesi, beklerken diğer işlemin verilerinin ilgili hafızadan okunması, işlemler sonunda elde edilen verilerin hafızaya kaydedilmesi, algoritmanın döngü sayısının kontrol edilmesi ve algoritmanın çalışmasının durdurulması vb. durumları içermektedir. Bazı durumlarda ise işlemlerin paralel olarak yapılması sağlanmaktadır.

5.3.3. Tasarımın İşleyişi

Tasarımın içinde yer alan birimler VHDL kullanılarak oluşturulmuştur. Bu birimler VHDL kodundan blok şeması oluşturularak gerekli bağlantılar yapılmış ve VHDL ile tasarlanan kontrol birimleri kullanılarak kontrol edilmiştir. Algoritma 5.2'de adımları

verilen algoritmanın tasarımı iki bölümden oluşmaktadır. İlk bölümde paralel dört işlem, karşılaştırıcı, ölçüm matrisi ve gözlem vektörünün depolandığı hafıza birimleri vb. birimler bulunmaktadır. ***Q***, ***R*** matrislerinin hesaplanması, ***x*** sinyalinin sıfırdan farklı olan elemanlarının indislerinin bulunması ve matrislerin güncellenmesi gibi işlemlerin yapıldığı tasarım Şekil 5.1'de gösterilmektedir.

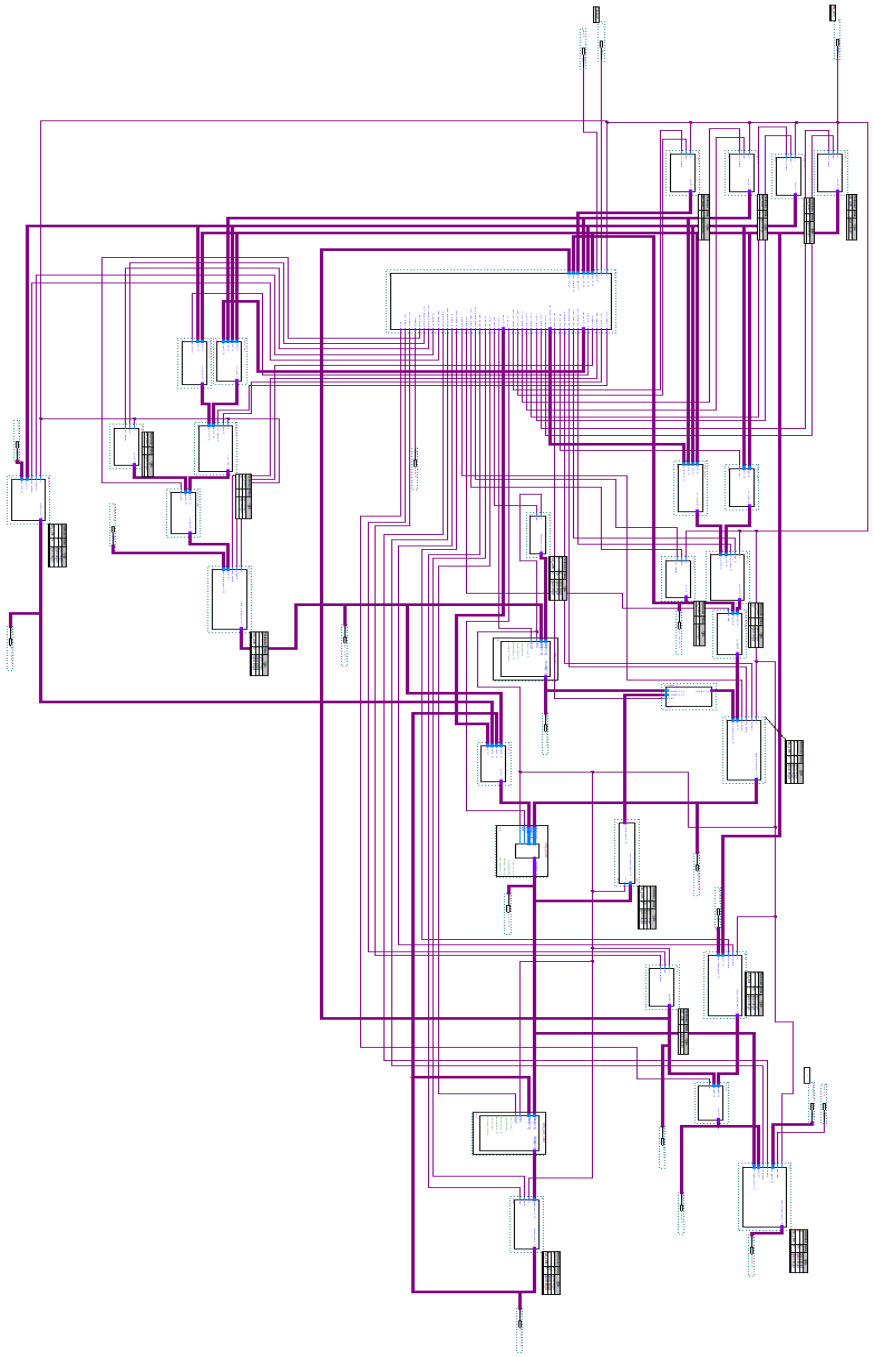


Şekil 5.1. ***Q, R*** matrisleri ve sinyalin sıfır olmayan elemanlarının hesaplandığı birinci kısım

Bu birimde öncelikle ***A*** matrisinin ilk sütunu ve *kalan* vektörü hafızadan okunur. Bu okuma işlemi 1 saat çevirimi kadar sürer. Okunan veriler paralel çarpma işlemi birimine aktarılır. İç çarpımın sonucu çıktığında karşılaştırma birimi ilk değerin mutlak değerini maksimum olarak kabul eder. Karşılaştırma işlemi devam ederken aynı zamanda matrisin diğer sütunu hafızadan okunmaktadır ve ondan önce okunan verilerin de iç çarpımı devam etmektedir. En son okunan ölçüm matrisi sütununun iç çarpımı tamamlandıktan sonra karşılaştırma birimi son değeri de karşılaştırır ve maksimum mutlak değere ait indeksi kaydeder (Adım 1). Bu işlemden sonra, ölçüm matrisinin tutulduğu hafızadan maksimum değerin indeksinin olduğu sütun okunarak ***C*** matrisinin hafızasına kaydedilir (Adım 2). Çarpma işlemi kullanılarak *C* matrisinin vektör normu hesaplanır ancak karekök alınmaz (Adım 4). Elde edilen sonuç *R* matrisinin hafızasına kaydedilirken aynı FSM durumu içinde Adım 6'da yer alan ***Q*** matrisinin sütun vektörü ve *kalan* vektörlerinin iç çarpımı gerçekleşir. Buradaki adımda Denklem (5.19) kullanıldığı için ***Q*** matrisinin çarpılacak olan sütun vektörü ***C*** matrisinin sütun vektörünün o andaki değerine eşittir. Dolayısıyla Adım 6'daki iç çarpımda ***Q*** matrisi yerine ***C*** matrisinin sütun vektörü okunur aynı zamanda da Adım 5'te belirtilen ***C***'nin sütun vektörünün değeri ***Q*** matrisinin tutulduğu hafızaya aktarılır. İç çarpım sonucunda çıkan sonuç ile ***Q*** matrisinin o anki sütun vektörüyle vektörel çarpım yapmak için vektörel çarpım durumuna geçilir. Daha sonra sırasıyla bölme ve çıkarma işlemlerinin olduğu durumlar tamamlanır ve *kalan* vektör güncellenir (Adım 6). Burada çıkarma işleminin girişi bölme işleminden elde edilen sonuca bağlı olduğu için paralellik yoktur. Bu işlemler farklı FSM durumlarında gerçekleşmektedir. Döngüden çıkıp çıkılmayacağının kontrolü bir sayıcı ile kontrol edilmektedir. İkinci döngüden itibaren ***R***'nin köşegen olmayan değerlerinin hesaplanması için ***Q*** ve ***C*** matrislerinin döngü sayısına karşılık gelen sütunlarının iç çarpımları yapılır. ***R***'nin elemanı hesaplandıktan sonra *kalan* vektörü güncellenmesinde olduğu gibi sırasıyla çarpma, bölme ve çıkarma işlemlerinden sonra ***C*** matrisi güncellenmiş olur (Adım 3). Başka bir sayıcı iç döngüyü kontrol eder ve sayıcının o andaki değeri gelecek durumun ne olacağını belirler. Genel algoritmanın döngü sayısı seyreklik seviyesine ulaştığında ***Q*** ve ***R*** matrisleri hesaplanmış olur (Algoritma 5.2, Adım 1-6).

Q ve ***R*** matrislerinin hesaplanmasından sonra birinci kısımda yer alan kontrol birimi (FSM1), ***R*** matrisinin tersinin alındığı tasarımın kontrol birimine (FSM2) ***R*** verisinin

hazır olduğu bilgisini göndererek matrisin tersinin alınması işlemini başlatır. FSM1 bu sinyali gönderdikten sonra Denklem (5.15)'de yer alan $\boldsymbol{Q^T \times y}$ matris vektör çarpımını birinci kısımda gerçekleştirir. $\boldsymbol{R}$ matrisinin tersinin alındığı ve geri oluşturulan $\boldsymbol{x}$ sinyalinin hesaplandığı ikinci kısmın şematik tasarımı Şekil 5.2'de gösterilmektedir.



Şekil 5.2. $\boldsymbol{R}$ matrisinin tersinin alındığı ve sinyalin geri oluşturulduğu tasarım.

Matris tersinin alındığı bu bölüm kayar nokta olarak tasarlanmıştır. İlk bölümden gelen veriler aktarım sırasında sabit noktadan kayar noktaya çevrilmiştir. İkinci kısım, Algoritma 5.2'deki Adım 7-9 arasını gerçekleştirmektedir. Tasarımda çarpma, bölme, toplama işlemleri, hafıza birimleri ve kontrol birimi yer almaktadır. Buradaki kontrol birimi ilk kısımdaki kontrol birimiyle koordineli bir şekilde çalışarak birbirine bağlı olmayan dataların hesaplanması mümkün olduğunca paralel bir şekilde yapılmaktadır**.** $\boldsymbol{R}$ matrisi hazır olduktan sonra FSM1, FSM2'ye başlatma sinyali göndererek ikinci bölümün çalışmasını başlatır. İkinci kısımda matrisin tersinin alınması işlemiyle, birinci kısımda $\boldsymbol{Q^T \times y}$ matris çarpma işlemi aynı anda başlar ve bunlar paralel bir şekilde farklı FSM'ler kullanılarak kontrol edilir. İlk kısımda hesaplanan $\boldsymbol{Q^T \times y}$ çarpımının sonucu ikinci kısma gönderilir İkinci kısımda $\boldsymbol{R}$ matrisinin tersinin alınması işlemi bittiğinde, Denklem (5.15)'in hesaplamasına başlanır. Denklem (5.15)'den elde edilen sonuçlar maksimum indislerin tuttuğu adreslere sırasıyla kaydedilerek $\boldsymbol{x}$ sinyali geri elde edilmiş olur.

5.4. Değerlendirme ve Sonuçlar

Bu çalışmada önerilen OMP tasarımının bazı özellikleri literatürdeki benzer sinyal boyutları için tasarlanan çalışmalarla karşılaştırılmıştır. Bu karşılaştırma Tablo 5.1 ve Tablo 5.2'de verilmiştir. Bu tablolardaki çalışmalarda karşılaştırılan tasarımlar için *M, N, k* sırasıyla 32, 128 ve 5'tir. Maksimum çalışma frekansı, tasarımda tek saat çevriminde en uzun süren yola ve kullanılan FPGA ailesine göre değişmektedir. Tasarımda tek saat çevriminde en uzun süren işlem eğer mümkünse farklı saat çevrimlerine bölünerek tasarımın maksimum frekansı artırılabilir. Ancak bu durumda algoritmanın toplam saat çevrim sayısı artacaktır. Dolayısıyla tasarımın gereksinim ve öğelerine göre optimum seviyeyi sağlamak gerekmektedir. Literatür incelendiğinde Xilinx FPGA ailesine ait olan kartlarda yapılan çalışmaların daha fazla sayıda olduğu görülmektedir. Tabloda Intel FPGA ailesinden bir çalışma mevcuttur [127]. Cyclone V kartında yapılan tasarımlar karşılaştırıldığında bu çalışmada önerilen tasarımın maksimum frekansı daha yüksektir. Ancak algoritmanın toplam saat çevrim sürelerini de göz önünde bulundurmak gerekir. Aynı frekansta saat çevrim sayısı küçük olan tasarım geri oluşturma süresi açısından daha avantajlıdır. Algoritmanın çalışma süresi; saat periyodu ve toplam saat çevrim sayısının çarpımına eşittir. Tasarımdaki diğer etken kaynak kullanımıdır. Aynı işi yapan bir tasarım için kullanılan blok RAM sayısı, yazmaç sayısı ve sayısal işaret işlemci sayısı bize tasarımın verimliliği hakkında bilgi vermektedir. Önerilen tasarım değerlendirildiğinde,

Tablo 5.1'deki Intel FPGA ailesinde yapılan tasarımlara göre bu çalışmadaki yöntemde kullanılan yazmaç sayıları daha fazla, blok RAM ve DSP sayıları daha azdır. Tablo 5.1'de Xilinx FPGA ailesindeki tasarımlarla karşılaştırıldığında ise genel olarak kaynak kullanımının önerilen metotta daha az olduğu görülmektedir.

Tablo 5.1. FPGA tasarımı ve kaynak kullanımı ile ilgili özellikler.

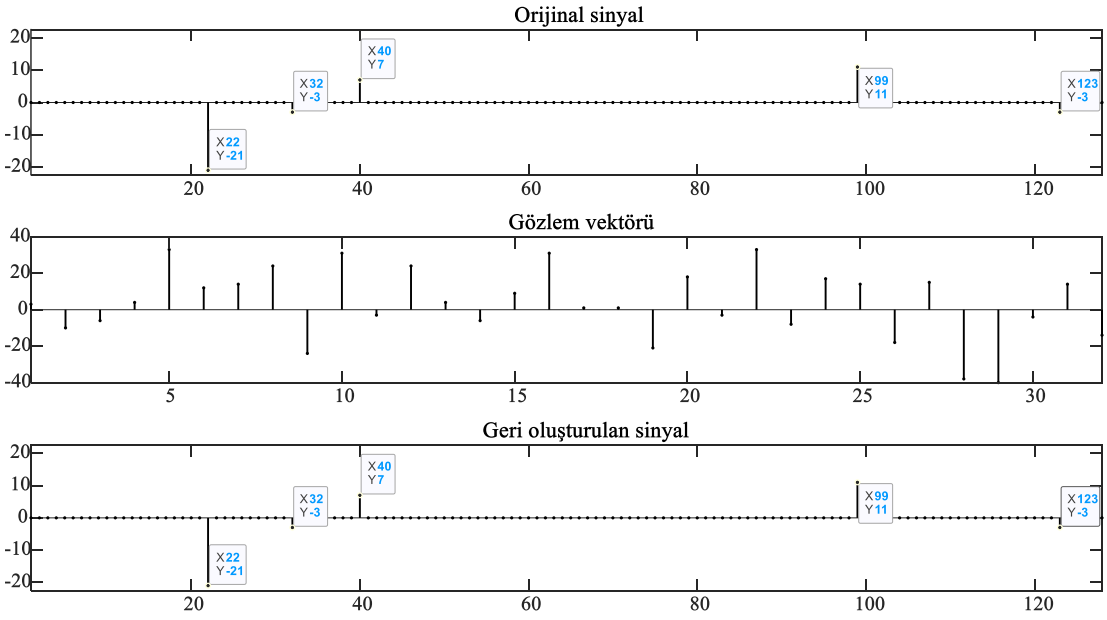
Gerçekleştirilen Çalışma	**FPGA Ailesi**	**Maksimum Frekans Mhz**	**Blok RAM**		**DSP Blok**	**Bileşik ALUT**	**Özgül Yazmaç**
			Sayısı	Bitleri	Sayısı		
Önerilen OMP Tasarımı	Cyclone-V	109	-	96852	39	23794	16248
Polat ve Kayhan [127]	Cyclone-V	99	-	230400	42	9692	6086
	Stratix IV	140	-	189908	40	9210	6350
	Stratix V	162	-	230400	42	9696	6086
Septimus ve Steinberg [124]	Virtex-5	39	-	-	-	-	-
Knoop ve ark. [131]	Virtex-7	100	4	-	130	22.3K	13.7K
Yu ve ark. [130]	Kintex-7	100	-	-	-		
Polat ve Kayhan [127]	Virtex-5	115	42	-	43	14.9K	8.2K
Kim ve ark. [132]	Kintex Ultrascale	250	>307	-	>2K	>153K	>210K
Li ve ark. [133]	Zynq ultrascale	113	68	-	254	61.1K	55.7K
			70	-	254	61.3K	55.1K

Tablo 5.2'de bazı çalışmalarda tasarlanan OMP algoritmasının en küçük kareler yaklaşımında kullanılan metotlar, saat çevrim sayıları, kullanılan tasarım aracı gibi özellikler özetlenmiştir. Tablo 5.2'den görüldüğü gibi OMP algoritmasında kullanılan metotlar bazı çalışmalarda teorik olarak benzer olmasına rağmen tasarımın verimliliği, zaman performansı, kaynak kullanımı gibi kriterler takip edilen donanım tasarlama tekniklerine büyük oranda bağlıdır. OMP algoritmalarından bazıları VHDL ve Verilog gibi genel HDL kullanılarak tasarlanırken, bazıları ise Xilinx tarafından sağlanan XSG, C/C++ veya başka HLS imkânı sağlayan programlar kullanılarak tasarlanmıştır. Genel olarak HDL kullanımı esneklik ve yüksek seviye boru hattı tasarımı sağlamaktadır. HDL'de yapılan tasarımın performansı tasarımcıya büyük oranda bağlıdır ve tasarlama süresi genelde daha uzun sürmektedir. HLS ise sunduğu hazır bloklar veya yüksek seviyeli tanımlama imkânı sayesinde bir tasarımın HDL ile tasarlanma süresine göre daha kısa olmasını sağlayabilir, ancak HDL kadar esnek değildir sadece tanımlı bloklar

kullanılabilir. Tablo 5.2'den HLS veya HDL kullanımının performansa etkisi ile ilgili bir çıkarım yapılamamaktadır.

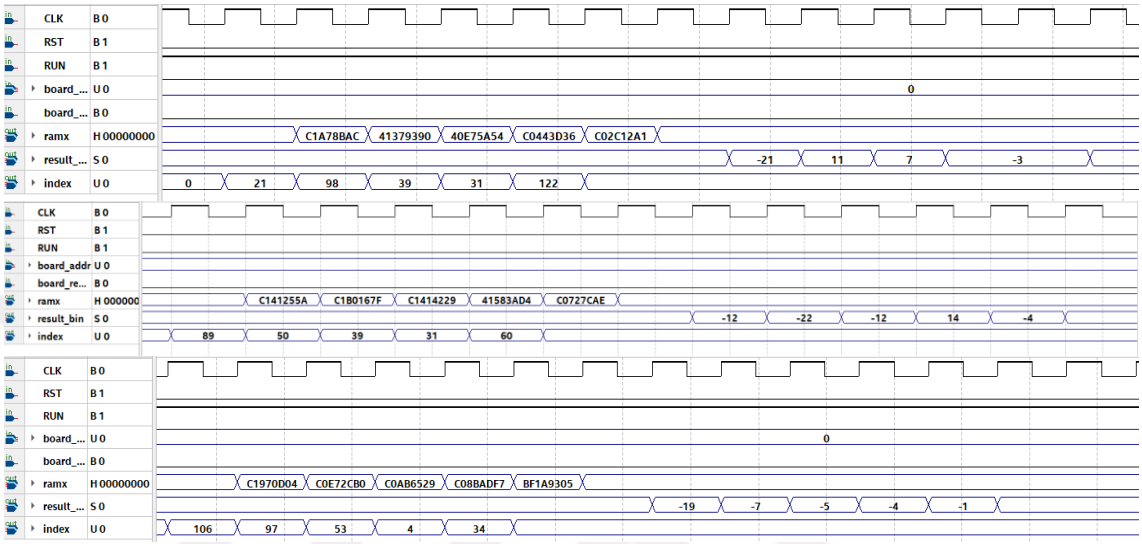
Tablo 5.2. Literatürde yer alan bazı çalışmaların tasarım özellikleri.

Çalışma	FPGA Ailesi	Kullanılan LS Metodu	Karekök	Çalışma Frekansı (Mhz)	Saat Çevrim Sayısı	Geri Oluşturma Süresi	HDL/HLS
Önerilen OMP Tasarımı	Cyclone V	QR	Yok	109	2400	22µs	HDL
Polat ve Kayhan [127]	Virtex-5	ACD	Yok	118	915	7.75µs	HDL
Septimus ve Steinberg [124]	Virtex 5	ACD	Yok	39	930	24µs	HDL
Knoop ve ark. [131]	Virtex 7	QR	Var	107	1800*	16.9µs	HLS
Yu ve ark. [130]	Virtex 5	QR	Yok	100	20599	205.985µs	HLS
Li ve ark. [133]	Zynq Ultrascale	ACD	Yok	113	1500*	13.2µs	HLS



Şekil 5.3. Test sinyalleri

Tasarlanan OMP algoritmasının etkinliği bazı test sinyalleri ile test edilmiştir. Bu tasarımın performansı $N = 128$ uzunluğuna $k = 5$ seyreklik seviyesine sahip sinyaller ve $M = 32$ örnek sayısına sahip test koşulları altında değerlendirilmiştir. Şekil 5.3'te kullanılan test sinyallerinden biri gösterilmektedir.



Şekil 5.4. Geri elde edilen sinyaller ve değeri sıfırdan farklı olan indisler.

Bu tasarım 3 aşamada test edilmiştir. Birincisi, tasarım birimlerinin doğru işlemleri yapıp yapmadığının kontrolünü sağlayan fonksiyonel simülasyon yöntemidir. Fonksiyonel simülasyonun sonucu lojik kapılarındaki gecikmeleri veya başka bir zaman gecikmesini dikkate almaz. Yapılan tasarımın, belirlenen bir işlemci üzerinde seçilen giriş ve çıkışlarının her bir saat çevrimindeki değerini gösterir. Tasarıma ait uygun giriş çıkışların kontrol edilmesiyle tasarımın doğruluğu kontrol edilir. Bu simülasyonda *clock*, *reset* ve *run* gibi giriş sinyalleri kullanılarak algoritmanın çalışması test edilmiştir. Bu simülasyon sonucunda algoritma akışındaki güncel durum (*pr_state*), gelecek durum (*nx_state*) ve bazı hafızaların giriş-çıkış sinyalleri görülmektedir. Fonksiyonel simülasyon sonucunda ortaya çıkan tasarım hataları giderildikten sonra, tasarımın zaman gecikmelerini de hesaba katan zamanlama simülasyonu yapılmaktadır. Bu simülasyon, tasarımda yer alan lojik kapılarının pratiğe yakın bir şekilde seçilen kart için zaman gecikmelerini de hesaplayarak sinyallerin saat çevrimine göre değişimini göstermektedir. Eğer fonksiyonel simülasyonda sonuçlar doğru olup zamanlama simülasyonunda sonuçlar yanlış ise algoritmada yer alan veriler planlanan zamanda doğru yere gitmiyor olabilir. Bunu düzeltmek için tasarımdaki en uzun süren birimi farklı saat çevirimlerine bölmek veya

farklı boru hattı evreleri eklemek gibi işlemlerle tasarımda değişiklikler yapmak gerekebilir. İkinci kısım sonunda geri oluşturulan sinyal Şekil 5.4'te gösterilmektedir.

Maksimum indeks değerleri ve geri oluşturulan sinyal değerleri incelendiğinde bu tasarımın doğru çalıştığı görülmektedir. Zamanlama simülasyonu kontrol edildikten sonra kart üzerinde doğruluğunu göstermek için sinyalin seyrek olmayan elemanlarının indislerini gösteren hafıza biriminin çıkışları ve geri elde edilen sinyalin kaydedildiği hafızanın çıkış sinyali kart üzerindeki LED'lerle kontrol edilerek doğrulanmıştır. Genel olarak seçilen bir kart çeşidi için tasarım zamanlama simülasyonuna göre doğruysa pratikte de doğru çalışmaktadır. Ancak bazı durumlarda fonksiyonel simülasyon ve zamanlama simülasyonuna göre doğru olduğu halde kart üzerinde çalışmamaktadır. Örneğin simülasyonda saat frekansını istediğimiz gibi ayarlayabiliriz. Ancak kart üzerindeki saat frekansı genelde sabittir ve tasarımın frekansıyla uyuşmadığı durumda zamanlama problemleri oluşacaktır. Böyle durumlarda PLL veya frekans bölücü kullanarak tasarıma göre saat frekansı ayarlanabilir.

Özetle bu bölümde, klasik CS algoritmalarından olan OMP algoritmasının VHDL dili ve şematik kullanarak FPGA üzerinde sayısal tasarımı yapılmıştır. Öncelikle tasarımı genel olarak doğrulamak için fonksiyonel simülasyon yapılmıştır. Fonksiyonel simülasyon sonucunda ortaya çıkan eksiklikler tamamlandıktan sonra, bir sonraki adım olan zamanlama simülasyonu yapılmıştır. Yapılan simülasyonlar sonucunda tasarımdaki hatalar düzeltilmiş ve FPGA üzerinde test edilmiştir. Tasarımda kullanılan lojik elemanlarının sayısı; sinyalin uzunluğu, seyrekliği ve ölçüm sayısıyla doğru orantılıdır. İşlemlerin verimliliği üzerinde çalışmalar yapılarak bu tasarımdaki saat frekansı ve kaynak sayısı optimize edilebilir. Ayrıca tasarımdaki boru hattı evreleri geliştirilerek de tasarımın saat çevrim sayıları optimum seviyeye getirilebilir. Tasarımın saat frekansının artması ve saat çevrim sayısının azalması algoritmanın çalışma süresinin kısalmasına neden olacak ve daha verimli bir tasarım ortaya çıkacaktır.

6. BÖLÜM

TARTIŞMA SONUÇ VE ÖNERİLER

Sezgisel algoritmalar çeşitli alanlarda yaygın olarak kullanıldığı gibi CS yönteminde de SSR probleminin çözümünde kullanılmaktadırlar. SSR problemi optimize edilmesi gereken iki amaç fonksiyonuna sahip olduğu için MOOA'ların bu alanda kullanılması uygundur. Dolayısıyla bu tez çalışmasında SSR probleminin çözümündeki düzenleme parametresinin algoritma performansına etkisini en aza indirgemek için SSR problemi MOSR problemine dönüştürülmüştür.

Tez çalışmasında öncelikle, literatürdeki MOSR yöntemleri ve başlıca tasarım özellikleri incelenmiştir. İncelenen metotlarda kullanılan algoritmalar ve metotlardaki tasarım özellikleri değerlendirilmiş ve bazı çıkarımlarda bulunulmuştur. Ayrıca NSGA II, MOEAD ve MOPSO algoritmaları SSR probleminin optimizasyonunda kullanılmış ve çeşitli sinyaller üzerinde test edilmiştir.

Tez çalışması kapsamında, literatürde daha önce CS alanında kullanılmayan MOABC algoritmasına dayalı SSR yöntemi önerilmiştir. MOABC-SR adı verilen bu yöntem, seyreklik ve ölçüm hatası hedeflerini aynı anda optimize etmektedir. Kullanılan ABC algoritmasının diğer algoritmalara göre daha az kontrol parametresine sahip olması ve gelişmiş bir arama stratejisine sahip olması önerilen yöntemin avantajlı olmasını sağlamaktadır. Önerilen yöntemde, yerel arama metodu olarak IHalfT algoritması kullanılmıştır ve domine edilemeyen çözümlerin ölçüm hatasını hesaplayarak en küçük hataya sahip olan çözüm son çözüm olarak belirlenmiştir.

Önerilen yöntem EKG ve görüntü sinyallerinin geri oluşturulması problemlerine uygulanmıştır. Çeşitli SSR algoritması kategorilerinden on algoritma farklı sinyaller üzerinde test edilmiş ve karşılaştırılmıştır. Ayrıca, çok amaçlı optimizasyon ve tek amaçlı optimizasyon metotlarının SSR probleminin çözümündeki performanslarının

değerlendirilmesi üzerine de çalışma yapılmıştır. Elde edilen verilere göre sonuçlar şu şekilde özetlenebilir:

- MOSR yöntemlerinin SSR performansları SOTF sinyalleri kullanılarak değerlendirildiğinde bu metotların karşılaştırılan diğer algoritmalardan daha fazla SOTF sinyalini daha az hata ile geri oluşturabildiği görülmektedir. Genel olarak MOABC-SR yöntemi, SOTF sinyallerinin geri oluşturulmasında karşılaştırılan diğer algoritmalardan daha üstündür. Ancak, bazı SOTF sinyalleri için istisnalar vardır.

- Seyreklik bilgisi bu çalışmada karşılaştırılan bazı algoritmalar için gerekliyken, MOSR metotları sinyal seyrekliğini ve ölçüm hatasını aynı anda optimize ettiğinden dolayı seyreklik bilgisinin önceden bilinmesine gerek yoktur. Literatürde SOTF sinyallerinin kullanıldığı bazı çalışmalarda ME hatası verilmiş ancak seyreklik hatası verilmemiştir. Daha küçük seyreklik değerleri için çok düşük ölçüm hataları olabileceğinden dolayı algoritmanın performansı hakkında doğru bir çıkarım yapabilmek için diğer amaç değeri olan seyreklik hatası da verilmelidir.

- Simülasyon sonuçları bölümünde MOABC-SR algoritması ile diğer sık kullanılan diz noktası seçim yöntemleri kullanılmış ve tasarlanan nihai çözüm elde etme yöntemi ile karşılaştırılmıştır. Bazı test sinyalleri için, önerilen nihai çözüm seçim yöntemi, açı temelli yöntem ve Kink yöntemine göre daha düşük bir hataya sahiptir. Ayrıca yapılan bu çalışmaya göre, MOABC-SR algoritmasının açı temelli yöntem ve Kink yöntemleriyle de uyumlu çalıştığı görülmektedir.

- MOEA/D, NSGA II ve MOPSO algoritmaları, MOABC ile aynı şartlarda karşılaştırılmıştır. Elde edilen sonuçlara göre, MOABC'ye dayalı yöntem, SOTF, görüntü ve EKG gibi test sinyalleri için genel olarak bu algoritmalara göre daha iyi performansa sahiptir.

- MOSR metotları, bu çalışmada karşılaştırılan tek amaçlı optimizasyon algoritmalarına göre daha yüksek seyreklik değerine sahip sinyalleri daha düşük ölçüm sayıları kullanarak geri oluşturabilmektedirler. Bu durum sinyalin ve problemin özelliklerine göre değişkenlik gösteren seyreklik ve ölçüm sayısı aralıklarının MOSR metotlarında daha geniş olduğunu, dolayısıyla MOSR

metotlarının performanslarının bu değişimlerden daha az etkilendiğini göstermektedir.

- Görüntü sinyalinin geri oluşturulmasında elde edilen SNR ve SSIM sonuçlarına göre, MOABC ye dayalı yöntem karşılaştırılan diğer algoritmalardan daha üstündür.

- EKG sinyalinin CS uygulandıktan sonra geri oluşturulmasında elde edilen PRD ve SNR verilerine göre, MOSR metotları bu alanda etkili bir yöntemdir.

Ayrıca tez çalışmasında, OMP algoritması FPGA üzerinde tasarlanmıştır. OMP algoritmasının temel prensibi gözlem vektörü ile en çok ilintili olan ölçüm matrisi sütunlarının her yinelemede bulunmasına dayanmaktadır. OMP algoritmasının hesaplama karmaşıklığı ve SSR performansı optimum bir seviyede olduğu için CS uygulamalarında yaygın olarak kullanılmaktadır. Tasarım, VHDL dili ve şematik kullanılarak gerçekleştirilmiştir. Tasarımı genel olarak doğrulamak için fonksiyonel simülasyon yapılmıştır. Fonksiyonel simülasyon sonucunda ortaya çıkan eksiklikler tamamlandıktan sonra, bir sonraki adım olan zamanlama simülasyonu yapılmıştır. Yapılan simülasyonlar sonucunda tasarımdaki hatalar düzeltilmiş ve FPGA kartı üzerinde deneme yapılmıştır. Algoritmanın kart üzerinde çalışması, sinyalin seyrek olmayan elemanlarının indislerini gösteren hafıza biriminin çıkışları ve geri elde edilen sinyalin kaydedildiği hafızanın çıkışı kart üzerindeki LED'lerle kontrol edilerek doğrulanmıştır. Tasarım, literatürdeki aynı boyutlu sinyaller için tasarlanan OMP çalışmalarının bazı performans sonuçlarıyla kıyaslanmıştır. Tasarımın saat frekansı, saat çevrim sayısı ve geri elde etme süresi bakımından literatürdeki çalışmalarla kıyaslandığında rekabetçidir. Tam bir performans kıyaslaması için tasarlanan platformun ve FPGA ailesinin aynı olması gerekmektedir.

Bu çalışmadaki OMP tasarımı, HDL dili kullanılarak tasarlanmıştır. HDL kullanımı, boru hattı tasarımında ve birimlerin tasarımında esneklik sağlayarak optimum saat frekansının elde edilmesi gibi konularda avantajlara sahiptir. Ancak, HDL ile tasarım uzun zaman alabilmektedir. Eğer mümkünse tasarımda yüksek seviyeli programlar kullanılarak tasarım süresi kısaltılabilir. Tasarımı geliştirmek için yapılan işlemlerin verimliliği üzerine çalışma yapılarak bu tasarımdaki saat frekansı artırılabilir. Ayrıca tasarımdaki boru hattı evreleri geliştirilerek de tasarımın saat çevrim sayıları optimum seviyeye getirilebilir. Tasarımın saat frekansının artması ve saat çevrim sayısının azalması

algoritmanın çalışma süresinin kısalmasına neden olacak ve daha verimli bir tasarım ortaya çıkacaktır.

Derin öğrenme yöntemleri son zamanlarda birçok alanda olduğu gibi CS alanında da kullanılmaya başlanmıştır. CS uygulamalarında kullanmak için etkili ve verimli bir derin öğrenme metodu geliştirilebilir. Geliştirilen bu metot FPGA üzerinde tasarlanarak gerçek zamanlı uygulamalarda kullanılabilir.

Bu çalışmada kullanılan test sinyalleri gerçek değerli sinyallerdir. Karmaşık değerli sinyallerde SSR probleminin optimizasyonu için verilerin gerçek ve sanal kısımlarını tek bir gerçek değerli dizide düzenleyip bu diziye CS uygulayarak değiştirilebilir. Bir SSR yöntemini karmaşık değerli CS problemlerinde kullanmak için, gözlemlenen sinyalin gerçek ve sanal kısımları ayrıştırılır ve bir sütun vektöründe birleştirilir. Ayrıca, sinyalin orijinal boyutunun dört katı büyüklüğünde bir matris oluşturmak için ölçüm matrisinin gerçek ve sanal olmak üzere alt matrislere ayrılması gerekir. Karmaşık değerli bir sinyalin gerçek ve sanal kısımlarını optimize etmek için MOSR algoritmalarının popülasyon sayıları ve bazı parametrelerinin değiştirilmesi gerekmektedir.

KAYNAKÇA

1. Donoho, D.L., 2006. Compressed sensing. **IEEE Transactions on Information Theory, 52** (4): 1289-1306.

2. Candes, E.J., Romberg, J., Tao, T., 2006. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. **IEEE Transactions on Information Theory**, **52** (2): 489-509.

3. Candes, E.J., Wakin, M.B., 2008. An introduction to compressive sampling. **IEEE Signal Processing Magazine, 25** (2): 21-30.

4. Candes, E.J., 2008. The restricted isometry property and its implications for compressed sensing. **Comptes Rendus Mathematique**, **346** (9-10): 589-592.

5. Rani, M., Dhok, S.B., Deshmukh, R.B., 2018. A systematic review of compressive sensing: concepts, implementations and applications. **IEEE Access, 6**: 4875-4894.

6. Otazo, R., Candes, E.J, Sodickson, D.K., 2015. Low-rank plus sparse matrix decomposition for accelerated dynamic MRI with separation of background and dynamic components. **Magnetic Resonance in Medicine**, **73** (3): 1125-1136.

7. Huang, W., Xiao, L. Liu, H. Wei, Z., 2015. Hyperspectral imagery super-resolution by compressive sensing inspired dictionary learning and spatial-spectral regularization. **Sensors**, **15** (1): p. 2041-2058.

8. Cao, P., Wu, E.X., 2015. Accelerating phase-encoded proton MR spectroscopic imaging by compressed sensing. **Journal of Magnetic Resonance Imaging, 41** (2): 487-495.

9. Patel, V.M., Easley, G. R., Healy, D. M., Chellappa, R., 2010. Compressed synthetic aperture radar. **IEEE Journal of Selected Topics in Signal Processing,**. **4** (2): 244-254.

10. Zhang, L., Xing, M. D., Qiu, C. W., Li, J., Sheng, J. L., Li, Y. C., Bao, Z., 2010. Resolution enhancement for inversed synthetic aperture radar imaging under low

snr via improved compressive sensing. **IEEE Transactions on Geoscience and Remote Sensing**, **48** (10): 3824-3838.

11. Huang, Q., Qu, L. L., Wu, B. H., Fang, G. Y., 2010. UWB through-wall imaging based on compressive sensing. **IEEE Transactions on Geoscience and Remote Sensing, 48** (3): 1634-1634.

12. Metzler, C.A., Maleki, A., Baraniuk, R.G, 2016. From denoising to compressed sensing. **IEEE Transactions on Information Theory, 62** (9): 5117-5144.

13. Li, S.T., Yang, B., 2011. A New pan-sharpening method using a compressed sensing technique. **IEEE Transactions on Geoscience and Remote Sensing**, **49** (2):738-746.

14. Poh, K.K., Marziliano, P., 2010. Compressive sampling of EEG signals with finite rate of innovation. **Eurasip Journal on Advances in Signal Processing**.

15. Tang, W.L., Cao, H. B., Duan, J. B., Wang, Y. P., 2011. A compressed sensing based approach for subtyping of leukemia from gene expression data. J**ournal of Bioinformatics and Computational Biology, 9** (5): 631-645.

16. Liu, X.Y., Zhu, Y. M., Kong, L. H., Liu, C., Gu, Y., Vasilakos, A. V., Wu, M. Y., 2015. CDC: Compressive data collection for wireless sensor networks. **IEEE Transactions on Parallel and Distributed Systems**, **26** (8):. 2188-2197.

17. Xin, K.F., Cheng, P, Chen, J.M., 2015. Multi-target localization in wireless sensor networks: a compressive sampling-based approach. **Wireless Communications & Mobile Computing, 15** (5): 801-811.

18. Hong, J.P., Choi, W., Rao, B.D., 2015. Sparsity controlled random multiple access with compressed sensing. **IEEE Transactions on Wireless Communications, 14** (2): 998-1010.

19. Okada, H., 2017. Link quality information sharing by compressed sensing and compressed transmission for arbitrary topology wireless mesh networks. **IEICE Transactions on Communications, E100b** (3): 456-464.

20. Aderohunmu, F.A., 2015. A data acquisition protocol for a reactive wireless sensor network monitoring application. **Sensors, 15** (5): 10221-10254.

21. Bajwa, W.U., Haupt, J., Sayeed, A. M., Nowak, R., 2010. Compressed channel sensing: a new approach to estimating sparse multipath channels. **Proceedings of the IEEE, 98** (6): 1058-1076.

22. Eltayeb, M.E., Al-Naffouri, T.Y., Bahrami, H.R., 2014. Compressive sensing for feedback reduction in MIMO broadcast channels. **IEEE Transactions on Communications, 62** (9): 3209-3222.

23. Shen, J.C., Zhang, J., Alsusa, E., Letaief, K. B., 2016. Compressed CSI acquisition in FDD massive MIMO: How much training is needed ? **IEEE Transactions on Wireless Communications, 15** (6): 4145-4156.

24. Shi, G.M., Lin, J., Chen, X. Y., Qi, F., Liu, D. H., Zhang, L., 2008. UWB echo signal detection with ultra-low rate sampling based on compressed sensing. **IEEE Transactions on Circuits and Systems Ii-Express Briefs, 55** (4): 379-383.

25. Wang, W.D., Wang, S. F., Yang, J. A., Liu, H., 2015. Under-sampling of PPM-UWB communication signals based on CS and AIC. **Circuits Systems and Signal Processing,**. **34** (11): 3595-3609.

26. Edelmann, G.F., Gaumond, C.F., 2011. Beamforming using compressive sensing. **Journal of the Acoustical Society of America**, **130** (4): El232-El237.

27. Hawes, M., Liu, W., Mihaylova, L., 2015. Compressive sensing based design of sparse tripole arrays. **Sensors**, **15** (12): 31056-31068.

28. Nagesh, P. Li, B.X., 2009. A compressive sensing approach for expression-invariant face recognition, 1518-1525.**IEEE Conference on Computer Vision and Pattern Recognition**.

29. Qiao, L.S., Chen, S.C., Tan, X.Y, 2010. Sparsity preserving projections with applications to face recognition. **Pattern Recognition, 43** (1): 331-341.

30. Gemmeke, J.F., Van Hamme, H., Cranen, B., Boves, L., 2010. Compressive Sensing for Missing Data Imputation in Noise Robust Speech Recognition. **IEEE Journal of Selected Topics in Signal Processing**, **4** (2): 272-287.

31. Wright, J., Ma, Y., Mairal, J., Sapiro, G., Huang, T. S., Yan, S.C., 2010. **Sparse Representation for Computer Vision and Pattern Recognition.** Proceedings of the IEEE, 98 (6): 1031-1044.

32. Asaei, A., Cernak, M., Bourlard, H., 2015. On compressibility of neural network phonological features for low bit rate speech coding, 418-422. *16th Annual Conference of the International Speech Communication Association* (Interspeech 2015).

33. Harris, P., Philip, R., Robinson, S., Wang, L., 2016. Monitoring Anthropogenic Ocean Sound from Shipping Using an Acoustic Sensor Network and a Compressive Sensing Approach dagger. **Sensors**, **16** (3).

34. Lei, Z.X., Yang, K. D., Duan, R., Xiao, P., 2015. Localization of low-frequency coherent sound sources with compressive beamforming-based passive synthetic aperture. **Journal of the Acoustical Society of America, 137** (4): El255-El260.

35. Mun, S., Fowler, J.E., 2011. *Residual Reconstruction for Block-Based Compressed Sensing of Video,*183-192. Data Compression Conference.

36. Edgar, M.P., Sun, M. J., Gibson, G. M., Spalding, G. C., Phillips, D. B., Padgett, M. J.., 2016. Real-time 3D video utilizing a compressed sensing time-of-flight single-pixel camera. **Optical Trapping and Optical Micromanipulation**, **9922**.

37. Liao, C.H., Tao, J., Zeng, X., Su, Y. F., Zhou, D., Li, X., 2016. Efficient Spatial Variation Modeling of Nanoscale Integrated Circuits Via Hidden Markov Tree. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems**, **35** (6): 971-984.

38. Zhang, W.Y., Li, X., Liu, F., Acar, E., Rutenbar, R. A., Blanton, R. D., 2011. Virtual Probe: A Statistical Framework for Low-Cost Silicon Characterization of

Nanoscale Integrated Circuits. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems**, **30** (12): 1814-1827.

39. Huang, H.T., Yu, H., Zhuo, C., Ren, F. B., A compressive-sensing based testing vehicle for 3d tsv pre-bond and post-bond testing data, p. 19-25. 2*016 International Symposium on Physical Design (Ispd'16)*, 2016.

40. Arjoune, Y., Kaabouch, N., El Ghazi, H., Tamtaoui, A., 2017. Compressive sensing: Performance comparison of sparse recovery algorithms, pp. 1-7. *IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, 2017, Las Vegas, NV, USA.

41. Baraniuk, R.G., 2007. Compressive sensing [Lecture Notes]. **IEEE Signal Processing Magazine**, **24** (4): 118-121.

42. Wang, Z., Bovik, A. C., Sheikh, H. R., Simoncelli, E. P., 2004. Image quality assessment: From error visibility to structural similarity. **IEEE Transactions on Image Processing**, **13** (4): 600-612.

43. Shi, W.Z., Jiang, F., Liu, S. H., Zhao, D. B., 2020. Image Compressed Sensing Using Convolutional Neural Network. **IEEE Transactions on Image Processing, 29**: 375-388.

44. Yin, W.T., Morgan, S., Yang, J. F., Zhang, Y., 2010. Practical Compressive Sensing with Toeplitz and Circulant Matrices. **Visual Communications and Image Processing**, **7744**.

45. Do, T.T., Gan, L., Nguyen, N. H., Tran, T. D., 2012. Fast and Efficient Compressive Sensing Using Structurally Random Matrices. **IEEE Transactions on Signal Processing, 60** (1): 139-154.

46. Erkoc, M.E., Karaboga, N., 2019. Evolutionary algorithms for sparse signal reconstruction. **Signal Image and Video Processing, 13** (7): 1293-1301.

47. Du, X.P., Cheng, L.Z., Liu, L.F., 2013. A Swarm Intelligence Algorithm for Joint Sparse Recovery. **IEEE Signal Processing Letters**, **20** (6): 611-614.

48. Zhao, Y., He, X., Huang, T. W., Huang, J. J., 2018. Smoothing inertial projection neural network for minimization Lp-q in sparse signal reconstruction. **Neural Networks**, **99**: 31-41.

49. Tropp, J.A., Gilbert, A.C., 2007. Signal recovery from random measurements via orthogonal matching pursuit. **IEEE Transactions on Information Theory, 53** (12): 4655-4666.

50. Needell, D., Tropp, J.A., 2009. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. **Applied and Computational Harmonic Analysis**, **26** (3): 301-321.

51. Blumensath, T., Davies, M.E., 2008. Iterative Thresholding for Sparse Approximations. **Journal of Fourier Analysis and Applications**, **14** (5-6): 629-654.

52. Xu, Z.B., Chang, X. Y., Xu, F. M., Zhang, H., 2012. L-1/2 Regularization: A Thresholding Representation Theory and a Fast Solver. **IEEE Transactions on Neural Networks and Learning Systems,**. **23** (7): 1013-1027.

53. Marques, E.C., Maciel, N., Naviner, L., Cai, H., Yang, J., 2019. A Review of Sparse Recovery Algorithms. **IEEE Access**, **7**: 1300-1322.

54. Li, L., Yao, X., Stolkin, R., Gong, M. G. He, S., 2014. An Evolutionary Multiobjective Approach to Sparse Reconstruction. **IEEE Transactions on Evolutionary Computation**, **18** (6): 827-845.

55. Erkoc, M.E., Karaboga, N., 2022. A comparative study of multi-objective optimization algorithms for sparse signal reconstruction. **Artificial Intelligence Review, 55** (4): 3153-3181.

56. Zhou, A.M., Qu, B. Y., Li, H., Zhao, S. Z., Suganthan, P. N., Zhang, Q. F.., 2011. Multiobjective evolutionary algorithms: A survey of the state of the art. **Swarm and Evolutionary Computation**, **1**(1): 32-49.

57. Miettinen, K., 1999. Nonlinear multiobjective optimization. **International series in operations research & management science.** Boston: Kluwer Academic Publishers. xvii, 298.

58. Ehrgott, M., Multicriteria Optimization. 2nd ed. 2005, Berlin ; New York: Springer. xiii, 323 p.

59. Coello, C.A.C., 2006. Evolutionary multi-objective optimization: A historical view of the field. **IEEE Computational Intelligence Magazine**, **1**(1): 28-36.

60. Schaffer, J.D. Multiple objective optimization with vector evaluated genetic algorithms. in *1st International Conference on Genetic Algorithms*. 1985.

61. Fonseca, C.M. and P.J. Fleming, 1993. Genetic algorithms for multiobjective optimization - formulation, discussion and generalization, p. 416-423. *Proceedings of the Fifth International Conference on Genetic Algorithms*, 1993.

62. Zitzler, E., Thiele, L., 1999. Multiobjective evolutionary algorithms: A comparative case study and the Strength Pareto approach. **IEEE Transactions on Evolutionary Computation, 3** (4): 257-271.

63. Siinivas, N., Deb, K., 1994. Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. **Evolutionary Computation**, **2** (3): 221-248.

64. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. **IEEE Transactions on Evolutionary Computation**, **6** (2): 182-197.

65. Deb, K., Jain, H., 2014. An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems With Box Constraints. **IEEE Transactions on Evolutionary Computation, 18** (4): 577-601.

66. Zhang, Q.F., Li, H., 2007. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. **IEEE Transactions on Evolutionary Computation, 11** (6): 712-731.

67. Coello, C.A.C., Lechuga, M.S., *MOPSO: A proposal for multiple objective particle swarm optimization.* Cec'02: Proceedings of the 2002 Congress on Evolutionary Computation, Vols 1 and 2, 2002: p. 1051-1056.

68. Bandyopadhyay, S., Saha, S.,Maulik, U., Deb, K., 2008. A simulated annealing-based multiobjective optimization algorithm: AMOSA. **IEEE Transactions on Evolutionary Computation, 2008. 12** (3): 269-283.

69. Alatas, B., Akin, E., Karci, A., 2008. Modenar: Multi-objective differential evolution algorithm for mining numeric association rules. **Applied Soft Computing,**. **8** (1): 646-656.

70. Panduro, M.A., Brizuela, C. A., Covarrubias, D., Lopez, C. 2006. A trade-off curve computation for linear antenna arrays using an evolutionary multi-objective approach. **Soft Computing**, **10** (2): 125-131.

71. Konstantinidis, A., Charalambous, C., Zhou, A. M., Zhang, Q. F., 2010. Multi-objective mobile agent-based sensor network routing using MOEA/D, 2010, *IEEE Congress on Evolutionary Computation (Cec)*.

72. Janson, S., Merkle, D., Middendorf, M., 2008. Molecular docking with multi-objective particle swarm optimization. **Applied Soft Computing**, **8** (1): 666-675.

73. Wozniak, P., 2011. Preferences in multi-objective evolutionary optimisation of electric motor speed control with hardware in the loop. **Applied Soft Computing**, **11** (1): 49-55.

74. Lazzerini, B., Marcelloni, F., Vecchio, M., 2010. A multi-objective evolutionary approach to image quality/compression trade-off in JPEG baseline algorithm. **Applied Soft Computing**, **10** (2): 548-561.

75. Reynolds, A.P., de la Iglesia, B., 2009. A multi-objective GRASP for partial classification. **Soft Computing, 13** (3): 227-243.

76. Qasem, S.N., Shamsuddin, S.M., 2011. Radial basis function network based on time variant multi-objective particle swarm optimization for medical diseases diagnosis. **Applied Soft Computing, 11** (1): 1427-1438.

77. Coello, C.A.C., Pulido, G.T., Lechuga, M.S., 2004. Handling multiple objectives with particle swarm optimization. **IEEE Transactions on Evolutionary Computation, 8** (3): 256-279.

78. Li, H., Fan, Y. Y., Zhang, Q. F., Xu, Z. B. Deng, J. D., 2016. A multi-phase multiobjective approach based on decomposition for sparse reconstruction, p. 601-608 *IEEE Congress on Evolutionary Computation (Cec)*. 2016.

79. Li, H., Zhang, Q. F., Deng, J. D., Xu, Z. B., 2018. A Preference-Based Multiobjective Evolutionary Approach for Sparse Optimization. **IEEE Transactions on Neural Networks and Learning Systems, 29** (5): 1716-1731.

80. Donoho, D.L., 1995. De-Noising by Soft-Thresholding. **IEEE Transactions on Information Theory**, **41** (3): 613-627.

81. Blumensath, T., Davies, M.E., 2009. Iterative hard thresholding for compressed sensing. **Applied and Computational Harmonic Analysis**, **27** (3): 265-274.

82. Yan, B., Zhao, Q., Wang, Z. H., Zhao, X. Y., 2017. A hybrid evolutionary algorithm for multiobjective sparse reconstruction. **Signal Image and Video Processing, 11** (6): 993-1000.

83. Zhou, Y., Kwong, S., Guo, H. N., Zhang, X., Zhang, Q. F., 2017. A Two-Phase Evolutionary Approach for Compressive Sensing Reconstruction. **IEEE Transactions on Cybernetics,** 2017. **47** (9): 2651-2663.

84. Li, H., et al., 2012. MOEA/D with Iterative Thresholding Algorithm for Sparse Optimization Problems. Berlin, Heidelberg: **Springer Berlin Heidelberg**.

85. Yan, B., Zhao, Q., Wang, Z. H., Zhang, J. A., 2018. Adaptive decomposition-based evolutionary approach for multiobjective sparse reconstruction. **Information Sciences**, **462**: 141-159.

86. Liang, J.J., Wang, P., Yue, C. T., Yu, K. J., Li, Z. H., Qu, B. Y., 2018. Multi-objective Brainstorm Optimization Algorithm for Sparse Optimization. *IEEE Congress on Evolutionary Computation (Cec).* 2018. p. 2211-2218.

87. Li, H., Sun, J. Y., Meng, D. Y., Zhang, Q. F., 2019. A Multiobjective Approach Based on Gaussian Mixture Clustering for Sparse Reconstruction. **IEEE Access**, **7**: 22684-22697.

88. Yue, C.T., Liang, J., Qu, B. Y., Han, Y. H., Zhu, Y. S., Crisalle, O. D., 2020. A novel multiobjective optimization algorithm for sparse signal reconstruction. **Signal Processing, 167**.

89. Zhang, X.Y., Tian, Y., Jin, Y.C., 2015. A Knee Point-Driven Evolutionary Algorithm for Many-Objective Optimization. **IEEE Transactions on Evolutionary Computation, 19** (6): 761-776.

90. Branke, E., Deb, K., Dierolf, H., Osswald, M., 2004. Finding knees in multi-objective optimization. **Parallel Problem Solving from Nature 3242**: 722-731.

91. Mierswa, I., Wurst, M., Information preserving multi-objective feature selection for unsupervised learning, p. 1545-1552. *Genetic and Evolutionary Computation Conference*, Temmuz 2006, Washington, Seattle, USA.

92. Liang, J.J., Zhu, X. P., Yue, C. T., Li, Z. H., Qu, B. Y., 2018. Performance analysis on knee point selection methods for multi-objective sparse optimization problems, p. 2507-2514. *2018 IEEE Congress on Evolutionary Computation (Cec)*. 2018.

93. Chen, S.S.B., Donoho, D.L., Saunders, M.A., 2001. Atomic decomposition by basis pursuit. **Siam Review**, **43** (1): 129-159.

94. Malioutov, D.M., Cetin, M., Willsky, A.S., *Homotopy continuation for sparse signal representation.* 2005 Ieee International Conference on Acoustics, Speech, and Signal Processing, Vols 1-5, 2005: p. 733-736.

95. Mohimani, H., Babaie-Zadeh, M., Jutten, C., 2009. A Fast Approach for Overcomplete Sparse Decomposition Based on Smoothed l(0) Norm. **IEEE Transactions on Signal Processing, 57** (1): 289-301.

96. Zou, H., Hastie, T., 2005. Regularization and variable selection via the elastic net. **Journal of the Royal Statistical Society Series B-Statistical Methodology, 67**: 768-768.

97. Beck, A. Teboulle, M., 2009. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. **Siam Journal on Imaging Sciences**, **2** (1): 183-202.

98. Erkoc, M.E., Karaboga, N., 2021. A novel sparse reconstruction method based on multi-objective Artificial Bee Colony algorithm. **Signal Processing**, **189**.

99. Karaboga, D., 2005. An Idea Based on Honey Bee Swarm for Numerical Optimization, in Technical report-TR06. **Erciyes university, Engineering Faculty, Computer Engineering Department.**

100. Karaboga, D., 2010. Artificial bee colony algorithm. **Scholarpedia**, **5** (3): 6915.

101. Karaboga, D., *Yapay Zeka Optimizasyon Algoritmaları*. 2014. 231.

102. Karaboga, D., Basturk, B., 2007. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. **Journal of Global Optimization**, **39** (3): 459-471.

103. Karaboga, D., Basturk, B., 2008. On the performance of artificial bee colony (ABC) algorithm. **Applied Soft Computing**, **8** (1): 687-697.

104. Karaboga, N., 2009. A new design method based on artificial bee colony algorithm for digital IIR filters. **Journal of the Franklin Institute-Engineering and Applied Mathematics, 346** (4): 328-348.

105. Karaboga, D., Gorkemli, B., Ozturk, C., Karaboga, N., 2014. A comprehensive survey: artificial bee colony (ABC) algorithm and applications. **Artificial Intelligence Review, 42** (1): 21-57.

106. Liang, J., Gong, M., Li, H., Qu, B., ., Problem Definitions and Evaluation Criteria for the CEC Special Session on Evolutionary Algorithms for Sparse Optimization, January 2018. DOI: 10.13140/RG.2.2.12568.70403, Report number: 2018001.

107. Yue, C., Liang, J., Qu, B., Song, H., Li, G., Han, Y. ,2017. A Knee Point Driven Particle Swarm Optimization Algorithm for Sparse Reconstruction. *Simulated Evolution and Learning*. SEAL 2017, Springer, Cham.

108. Gill, P.E., Robinson, D.P., 2012. A primal-dual augmented Lagrangian. **Computational Optimization and Applications, 51** (1): 1-25.

109. http://www5.zzu.edu.cn/ecilab/Benchmark/Sparse_Opt.htm (Erişim Tarihi: 2018.

110. Gan, L., 2007. Block compressed sensing of natural images, pp. 403-406. *15th International Conference on Digital Signal Processing, 2007,* Cardiff, UK.

111. Adler, A., D. Boublil, and M. Zibulevsky, *Block-based Compressed Sensing of Images via Deep Learning*, in *2017 Ieee 19th International Workshop on Multimedia Signal Processing (Mmsp)*. 2017.

112. Erkoc, M.E., Karaboga, N., 2019. Blok Tabanlı Sıkıştırılmış Algılama Metotlarının İncelenmesi ve Karşılaştırılması, *EEMKON*, 2019, İstanbul, Türkiye.

113. Mun, S., Fowler, J.E., 2009. Block Compressed Sensing of Images Using Directional Transforms, p. 3021-3024. *16th IEEE International Conference on Image Processing*, 2009.

114. Unde, A.S., Deepthi, P.P., 2017. Block compressive sensing: Individual and joint reconstruction of correlated images. **Journal of Visual Communication and Image Representation, 44**: 187-197.

115. Erkoc, M.E., Karaboga, N., 2021. Sparse signal reconstruction by swarm intelligence algorithms. **Engineering Science and Technology-an International Journal-Jestech**, **24** (2): 319-330.

116. Mamaghanian, H., Khaled, N., Atienza, D., Vandergheynst, P., 2011. Compressed sensing for real-time energy-efficient ecg compression on wireless body sensor nodes. **IEEE Transactions on Biomedical Engineering**, **58** (9): 2456-2466.

117. Singh, A., Dandapat, S., 2017. Block sparsity-based joint compressed sensing recovery of multi-channel ECG signals. **Healthcare Technology Letters, 4** (2): 50-56.

118. Lu, Z.T., Kim, D.Y., Pearlman, W.A., 2000. Wavelet compression of ECG signals by the set partitioning in hierarchical trees algorithm. **IEEE Transactions on Biomedical Engineering**, **47** (7): 849-856.

119. Polania, L. F., Carrillo, R. E., Blanco-Velasco, M., Barner, K. E., 2011. Compressed sensing based method for ECG compression, p. 761-764. *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),* 2011, Prague, Czech Republic.

120. Erkoc, M.E., Karaboga, N., 2021. Multi-objective sparse signal reconstruction in compressed sensing p. 373-396. *In:* Nature-Inspired Metaheuristic Algorithms for Engineering Optimization Applications (Eds: Carbas, S., Toktas, A., Ustun, D.). Springer Singapore.

121. Goldberger, A.L., Amaral, L. A. N., Glass, L., Hausdorff, J. M., Ivanov, P. C., Mark, R. G., Mietus, J. E., Moody, G. B., Peng, C. K., Stanley, H. E., 2000. PhysioBank, PhysioToolkit, and PhysioNet - Components of a new research resource for complex physiologic signals**. Circulation**, **101** (23): E215-E220.

122. Zigel, Y., Cohen, A., Katz, A., 2000. The weighted diagnostic distortion (WDD) measure for ECG signal compression. **IEEE Transactions on Biomedical Engineering, 47** (11): 1422-1430.

123. Izadi, V., Shahri, P.K. Ahani, H., 2020. A compressed-sensing-based compressor for ECG. Biomedical Engineering Letters, **10** (2): 299-307.

124. Septimus, A., Steinberg, R., 2010. *Compressive Sampling Hardware Reconstruction.* 2010 IEEE International Symposium on Circuits and Systems, 2010: p. 3116-3119.

125. Stanislaus, J.L.V.M., Mohsenin, T., *Low-Complexity FPGA Implementation of Compressive Sensing Reconstruction.* 2013 International Conference on Computing, Networking and Communications (Icnc), 2013.

126. Rabah, H., Amira, A., Mohanty, B. K., Almaadeed, S., Meher, P. K., 2015. FPGA implementation of orthogonal matching pursuit for compressive sensing

reconstruction. **IEEE Transactions on Very Large Scale Integration Systems**, **23** (10): 2209-2220.

127. Polat, O., Kayhan, S.K., 2018. High-speed FPGA implementation of orthogonal matching pursuit for compressive sensing signal reconstruction. **Computers & Electrical Engineering, 71**: 173-190.

128. Ge, X., Yang, F., Zhu, H. L., Zeng, X., Zhou, D., 2019. An Efficient FPGA Implementation of Orthogonal Matching Pursuit With Square-Root-Free QR Decomposition. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems**, **27** (3): 611-623.

129. Mallat, S.G., Zhang, Z.F, 1993. Matching Pursuits with Time-Frequency Dictionaries. **IEEE Transactions on Signal Processing, 41** (12): 3397-3415.

130. Yu, Z.L., et al., 2016. Fast compressive sensing reconstruction algorithm on FPGA using Orthogonal Matching Pursuit, pp. 249-252. *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2016, Montreal, QC, Canada

131. Knoop, B., Rust, J., Schmale, S., Peters-Drolshagen, D., Paul, S., 2016. Rapid digital architecture design of orthogonal matching pursuit, pp. 1857-1861. *24th European Signal Processing Conference (EUSIPCO)*, 2016, Budapest, Hungary.

132. Kim, S., Yun, U., Jang, J., Seo, G., Kang, J., Lee, H. N., Lee, M., 2018. Reduced Computational Complexity Orthogonal Matching Pursuit Using a Novel Partitioned Inversion Technique for Compressive Sensing. **Electronics**, 2018. **7** (9).

133. Li, J., Chow, P., Peng, Y. X., Jiang, T., 2021. FPGA Implementation of an Improved OMP for Compressive Sensing Reconstruction. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems**, **29** (2): 259-272.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı: Murat Emre ERKOÇ
Uyruğu: Türkiye (T.C)

EĞİTİM

Derece	**Kurum**	**Mezuniyet Tarihi**
Yüksek Lisans	Erciyes Üniversitesi, Elektrik-Elektronik Mühendisliği	2018
Lisans	Orta Doğu Teknik Üniversitesi, Elektrik Elektronik Mühendisliği	2015
Lise	Eskişehir Atatürk Lisesi	2007

İŞ DENEYİMLERİ

Yıl	**Kurum**	**Görev**
2015-Halen	Erciyes Üniversitesi, Mühendislik Fakültesi	Araştırma Görevlisi

YABANCI DİL

İngilizce

YAYINLAR

- Erkoc, M.E., Karaboga N., 2022. A comparative study of multi-objective optimization algorithms for sparse signal reconstruction. **Artificial Intelligence Review, 55** (4): 3153-3181.

- Erkoc, M.E., Karaboga, N., 2021. A novel sparse reconstruction method based on multi-objective Artificial Bee Colony algorithm. **Signal Processing**, **189**.
- Erkoc, M.E., Karaboga, N. Multi-objective Sparse Signal Reconstruction in Compressed Sensing, in Nature-Inspired Metaheuristic Algorithms for Engineering Optimization Applications, S. Carbas, A. Toktas, and D. Ustun, Editors. 2021, Springer Singapore: Singapore. p. 373-396.
- Erkoc, M.E., Karaboga, N., 2021. Sparse signal reconstruction by swarm intelligence algorithms. **Engineering Science and Technology-an International Journal-Jestech, 24** (2): 319-330.
- Erkoc, M.E., Karaboga, N., 2019. Evolutionary algorithms for sparse signal reconstruction**. Signal Image and Video Processing, 13** (7): 1293-1301.
- Erkoc, M.E., Karaboga, N., 2019. Blok Tabanlı Sıkıştırılmış Algılama Metotlarının İncelenmesi ve Karşılaştırılması, *EEMKON*, 2019, İstanbul.