



**T.C.
ONDOKUZ MAYIS ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ
İNŞAAT MÜHENDİSLİĞİ ANA BİLİM DALI**

**META-SEZGİSEL ALGORİTMALAR İLE KARAYOLU
BOYKESİT OPTİMİZASYONU**

Doktora Tezi

Sina ASHERLOU

Danışman
Doç. Dr. Erhan Burak PANCAR

II. Danışman
Prof. Dr. Şeref ORUÇ

SAMSUN
2022

**T.C.
ONDOKUZ MAYIS ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ
İNŞAAT MÜHENDİSLİĞİ ANA BİLİM DALI**



**META-SEZGİSEL ALGORİTMALAR İLE KARAYOLU
BOYKESİT OPTİMİZASYONU**

Doktora Tezi

Sina ASHERLOU

Danışman

Doç. Dr. Erhan Burak PANCAR

II. Danışman

Prof. Dr. Şeref ORUÇ

SAMSUN
2022

TEZ KABUL VE ONAYI

Sina ASHERLOU tarafından, Doç. Dr. Erhan Burak PANCAR danışmanlığında hazırlanan “META-SEZGİSEL ALGORİTMALAR İLE KARAYOLU BOYKESİT OPTİMİZASYONU” başlıklı bu çalışma, jürimiz tarafından 27.12.2022 tarihinde yapılan sınav sonucunda oy birliği ile başarılı bulunarak Doktora Tezi olarak kabul edilmiştir.

	Unvanı Adı Soyadı Üniversitesi Ana Bilim/Ana Sanat Dalı	Sonuç
Başkan	Prof. Dr. Atakan AKSOY Karadeniz Teknik Üniversitesi İnşaat Mühendisliği Ana Bilim Dalı	☒ Kabul ☐ Ret
Üye	Doç. Dr. Erhan Burak PANCAR Ondokuz Mayıs Üniversitesi İnşaat Mühendisliği Ana Bilim Dalı	☒ Kabul ☐ Ret
Üye	Prof. Dr. Faik Ahmet SESLİ Ondokuz Mayıs Üniversitesi Harita Mühendisliği Ana Bilim Dalı	☒ Kabul ☐ Ret
Üye	Doç. Dr. Ufuk KIRBAŞ Ondokuz Mayıs Üniversitesi İnşaat Mühendisliği Ana Bilim Dalı	☒ Kabul ☐ Ret
Üye	Dr. Öğr. Üyesi Seda SELÇUK Çankaya Üniversitesi İnşaat Mühendisliği Ana Bilim Dalı	☒ Kabul ☐ Ret

Bu tez, Enstitü Yönetim Kurulunca belirlenen ve yukarıda adları yazılı jüri üyeleri tarafından uygun görülmüştür.

Prof. Dr. Ali BOLAT
Enstitü Müdürü

BİLİMSEL ETİĞE UYGUNLUK BEYANI

Hazırladığım Yüksek Lisans tezinin bütün aşamalarında bilimsel etiğe ve akademik kurallara riayet ettiğimi, çalışmada doğrudan veya dolaylı olarak kullandığım her alıntıya kaynak gösterdiğimi ve yararlandığım eserlerin Kaynaklar'da gösterilenlerden oluştuğunu, her unsurun enstitü yazım kılavuzuna uygun yazıldığını ve TÜBİTAK Araştırma ve Yayın Etiği Kurulu Yönetmeliği'nin 3. bölüm 9. maddesinde belirtilen durumlara aykırı davranılmadığını taahhüt ve beyan ederim.

Etik Kurul Gerekli mi ?

Evet ☐ (Gerekli ise ekler kısmına ekleyiniz)

Hayır ☒

06 /11 / 2022
Sina ASHERLOU

TEZ ÇALIŞMASI ÖZGÜNLÜK RAPORU BEYANI

Tez Başlığı: META-SEZGİSEL ALGORİTMALAR İLE KARAYOLU
BOYKESİT OPTİMİZASYONU

Yukarıda başlığı belirtilen tez çalışması için şahsım tarafından 06.11.2022 tarihinde intihal tespit programından alınmış olan özgünlük raporu sonucunda;

Benzerlik oranı : % 6

Tek kaynak oranı : % 4 çıkmıştır.

06 /11 / 2022
Doç. Dr. Erhan Burak PANCAR

ÖZET

META-SEZGİSEL ALGORİTMALAR İLE KARAYOLU BOYKESİT OPTİMİZASYONU

Sina ASHERLOU

Ondokuz Mayıs Üniversitesi

Lisansüstü Eğitim Enstitüsü

İnşaat Mühendisliği Ana Bilim Dalı

Doktora, Aralık/2022

Danışman: Doç. Dr. Erhan Burak PANCAR

Karayolu boykesit tasarımında en önemli problemlerden biri kırmızı çizginin kontrolünün sağlıklı ve kısa sürede yapılamamasıdır. Bu sebeple hem zaman kaybedilmekte hem de her noktada tam kontrol mümkün olmadığından kontrol dışı uygulamalar gerçekleşmektedir. Kırmızı çizgi optimizasyonu tasarım kriterlerini karşılamının yanı sıra toplam hafriyat miktarının en aza indirilmesini de içermektedir. Günümüzde kırmızı çizginin tasarımı deneme yanılma ile yapılmaktadır. Ancak fazlasıyla matematiksel işlemleri olduğundan önemli ölçüde zaman kaybına yol açmaktadır. Son yıllarda meta-sezgisel algoritma yöntemleri kullanılarak kırmızı çizgi optimizasyonu ile ilgili çalışmalar yapılmıştır. Bir karayolu projesinde kırmızı çizginin optimizasyonu doğrudan hafriyat optimizasyonunu etkilemektedir. Bu kapsamda pik noktaların otomatik bulunması aynı zamanda eğim sınır şartlarına da uyumlu olması ve en önemlisi hafriyat miktarı hesaplama sırasında arazi üzerinde siyah kotların otomatik bulunması minimum maliyete neden olan optimum kırmızı çizgi tasarımında yadsınamaz bir gerçektir. Bu kapsamda çeşitli tasarım kısıtlamalarına da dikkat ederek minimum maliyete neden olan optimum kırmızı çizgi bulunmalıdır. Ancak literatürde bulunan eksikliklerin giderilmesi, sadece çok aşamalı bir optimizasyon tekniği ile mümkün olabilecektir. Bu çalışmada hafriyat dengelemesi, yani yarma dolgu miktarını eşitleyecek ve aynı zamanda sıfır denilecek kadar az bir değer bulabilecek bir yazılım kodlaması hazırlanacaktır. Tasarlanan yazılımda tek amaçlı optimizasyon yöntemlerinden Yapay Arı Koloni (ABC), Ateşböceği Algoritması (FA) ve Parçacık Sürü Optimizasyonu (PSO) kullanılmaktadır. Aynı şekilde çok amaçlı optimizasyon algoritmalarından Kuvvet Pareto Evrimsel Algoritması (SPEA2), Parçacık Sürü Optimizasyonu (MOPSO) ve Bastırılmamış Sınıflandırılmalı Genetik Algoritma (NSGA III) kullanılmış ve yazılıma entegre edilerek karşılaştırılmıştır. Bu bağlamda, sunulan algoritmanın kırmızı çizgi optimizasyonunda başarılı bir ölçekte kullanılabileceğinin temel çalışması gerçekleştirilmiştir. Son olarak yöntemler üzerinden elde edilen sonuçlar çok çok tatmin edici bir seviyededir. Ancak, bu bağlamda genel olarak tek amaçlı yöntemlerden PSO yöntemi hafriyat dengelemesini neredeyse sıfıra yakın bir değer bulmuştur. Çok amaçlı yöntemlerden SPEA2 yönteminin sonuçlarının daha güvenilir ve istikrarlı olduğu görülmüş ve sonuçların tatmin edici seviyede olduğu belirlenmiştir.

Anahtar Sözcükler: Karayolu, Kırmızı çizgi, Optimizasyon, Meta-sezgisel algoritma

ABSTRACT

OPTIMIZATION OF HIGHWAY VERTICAL ALIGNMENT BY META-HEURISTIC ALGORITHMS

Sina ASHERLOU

Ondokuz Mayıs University

Institute of Graduate Studies

Department of Civil Engineering

Ph.D., December/2022

Supervisor: Assoc. Prof. Dr. Erhan Burak PANCAR

One of the most important problem in highway vertical alignment design is the inability to control the grade line in a healthy and short time. For this reason, both time is lost and uncontrolled applications take place because full control is not possible at every point. Grade line optimization includes meeting the design criteria while minimizing the total amount of earthwork. Today, the grade line is designed through heuristic approaches. However, due to the excessive number of mathematical procedures, it causes a considerable loss of time. In recent years, researchers have conducted studies on grade line optimization through meta-heuristic algorithm methods. The optimization of the grade line in a highway project directly affects the excavation optimization. In this context, automatic detection of the peak points, their compatibility with the slope boundary conditions, and most importantly, the automatic detection of ground elevations on the land during the computation of the earthwork amount are the facts that cannot be ignored in terms of the optimum grade line design as it brings minimum cost. The optimum grade line with the minimum cost should be determined by considering various design restrictions. However, it will only be possible to eliminate the deficiencies in the literature with a multi-stage optimization technique. In this study, a software code will be prepared to balance the amount of excavation, that is cleavage filling, and to find a value that is less than zero. The designed software was integrated into Artificial Bee Colony (*ABC*), Firefly Algorithm (*FA*) and Particle Swarm Optimization (*PSO*), which are one of the single-objective optimization methods, and with Force Pareto Evolutionary Algorithm (*SPEA2*), Particle Swarm Optimization (*MOPSO*) and Unqualified Classified Genetic Algorithm (*NSGA III*), which are among the multi-objective optimization algorithms, and the basic study showing that the presented algorithm can be used in a successful scale in grade line optimization was carried out. Finally, the results obtained through the methods are at a very satisfactory level. However, in this context, the *PSO* method, one of the single-objective methods, found excavation balancing to be almost zero and the results of the *SPEA2* method, one of the multi-objective methods, were made more reliable and stable and the results were determined at a very satisfactory level.

Keywords: Highway, Grade line, Optimization, Meta heuristic algorithm

ÖN SÖZ VE TEŞEKKÜR

Doktora tez danışmanlığımı üstlenerek çalışmalarımın yürütülmesi sırasında yönlendirmelerini ve desteğini benden hiçbir zaman esirgemeyen, bu çalışma kapsamında elde edilen verilerden yayın hazırlamamıza vesile olan, üzerimde çok büyük emeği olup her zaman yol gösteren ve hayatıma çok kıymetli izler bırakan saygı değer hocam Doç. Dr. Erhan Burak PANCAR'a sonsuz teşekkür eder, saygı ve şükranlarımı sunarım. Aynı şekilde, bu tez kapsamında ikinci danışmanlığımı üstlenen ve yüksek lisans eğitimimden itibaren benim bir akademisyen olma yolunda yürümeme vesile olan saygı değer hocam Prof. Dr. Şeref ORUÇ'a saygılarımı sunar ve teşekkür ederim.

Kendilerinden ders aldığım, aynı zamanda tez savunma sınavımda yer alarak bana önemli katkılar sağlayan değerli hocalarım sayın Prof. Dr. Faik Ahmet SESLİ ve Doç. Dr. Ufuk KIRBAŞ'a ayrıca görüş ve önerileriyle tezime değer katan ve tüm karmaşık matematiksel işlemlerde bana yol gösteren saygı değer Prof. Dr. Ziya YAPAR'a saygı ve şükranlarımı sunarım.

Tezin hazırlanması ve yazılım algoritmalarında bana destek olan, bilgi ve tecrübelerini benimle paylaşan saygı değer Dr. Öğr. Üyesi Aref YELGHİ'ye tezime olan katkıları ve tüm destekleri için teşekkür ederim.

Son olarak, hayatım boyunca yanımda olan, bana güven veren, destek ve yardımları ile bugünlere gelmemi sağlayan çok değerli babam Akbar ASHERLOU, annem Fakhri BAKHTYARI'ye ve sevgili ablam Elena ASHERLOU'ya minnet ve şükranlarımı sunmayı hayatımın sonuna kadar borç bilirim.

Tez çalışmam sırasında ihmal ettiğim çok değerli eşim Homeira HAJİHOSSEİNLOU'ya, bana göstermiş olduğu anlayıştan dolayı çok teşekkür ediyorum. Ayrıca, bu tezimi sevgili kızım Aryana'ya armağan ediyorum.

Sina ASHERLOU

İÇİNDEKİLER

TEZ KABUL VE ONAYI	i
BİLİMSEL ETİĞE UYGUNLUK BEYANI	ii
TEZ ÇALIŞMASI ÖZGÜNLÜK RAPORU BEYANI	ii
ÖZET	iii
ABSTRACT	iv
ÖNSÖZ VE TEŞEKKÜR	v
İÇİNDEKİLER	vi
ŞEKİLLER DİZİNİ	viii
TABLolar DİZİNİ	xi
1. GİRİŞ	1
2. KURAMSAL TEMELLER VE KAYNAK ÖZETLERİ	6
2.1. Karayolu Projelerinin Uygulanması	6
2.1.1. Altyapı Çalışmaları	6
2.1.2. Karayolu Üst Yapı Tasarımı	7
2.2. Karayolu ve Elemanları	7
2.2.1. Güzergâh-Plan-Boykesit	7
2.2.2. Toprak İşİ	8
2.2.3. Altyapı ve Üstyapı Tanımı	8
2.2.4. Alt Temel Tabakası	8
2.2.5. Temel Tabakası	9
2.2.6. Şev	10
2.2.7. Boyuna ve Enine Eğimler	10
2.2.8. Yolun Eksen Çizgisi	11
2.2.9. Başlangıca Olan Uzaklık (Kilometre)	11
2.3. Plan Tasarımı	12
2.3.1. Doğru Parçalar	12
2.3.2. Yatay Kurplar	12
2.4. Boykesit Tasarımı	13
2.4.1. Siyah Kot	13
2.4.2. Siyah Çizgi	13
2.4.3. Kırmızı Kot	14
2.4.4. Kırmızı Çizgi	14
2.4.5. Enkesit	14
2.4.6. Tip Enkesit	14
2.4.7. Yarma-Dolgu İşleri	15
2.4.8. Boykesitin Hazırlanması	16
2.4.9. Boyuna Eğimin Sınır Değerleri	17
2.4.10. Kırmızı Çizginin Geçirilmesi	18
2.4.11. Düşey Kurbalar	18
2.4.12. Dairesel Düşey Kurbalar	18
2.5. Optimizasyon Yöntemleri	19
2.5.1. Tek Amaçlı Parçacık Sürü Optimizasyonu (PSO)	20
2.5.2. Tek Amaçlı Ateşböceği Algoritması (Firefly Algorithm- FA)	21
2.5.3. Tek Amaçlı Yapay Arı Koloni (Artificial Bee Colony- ABC)	22
2.5.4. Çok Amaçlı Optimizasyon Yöntemi NSGAIİ	23
2.5.5. Çok Amaçlı Optimizasyon Yöntemi MOPSO	24
2.5.6. Çok Amaçlı Optimizasyon Yöntemi SPEA2	25
2.6. Kaynak Özetleri	25
3. MATERYAL VE YÖNTEM	32
3.1. Boykesit Tasarımı	33
3.1.1. Siyah Çizgi	33
3.1.2. Kırmızı Çizgi	34

3.1.3. Some Noktalarının Otomatik Olarak Belirlenmesi	34
3.1.4. Zorunlu Some Noktalarının Eklenmesi.....	36
3.1.5. Yasak Bölgelerdeki Some Noktaların Kaldırılması	38
3.1.6. Kırmızı Çizgi Eğimleri	39
3.1.7. Yarma-Dolgu Alanları	40
3.1.8. Some Noktasının Araziye Göre Durumu	42
3.2. Tek-Amaçlı Meta-Sezgisel Optimizasyon Algoritmaları.....	44
3.2.1. Kırmızı Çizgiye Ait Some Noktaların Optimizasyonu	45
3.2.2. Y Eksen Üzerinde Optimizasyon	48
3.2.3. X ve Y Eksen Üzerinde Optimizasyon	49
3.2.4. Kırmızı Çizgiye Ait Tek-Amaçlı Meta-Sezgisel Akış Şeması.....	51
3.3. Çok-Amaçlı Meta-Sezgisel Optimizasyon Algoritmaları	53
3.3.1. Çok-Amaçlı Meta-Sezgisel Algoritması Kullanarak Boykesit Tasarımı	53
3.3.2. Araziye ait Siyah Çizgi	54
3.3.3. Kırmızı Çizgi Tasarımı	54
3.3.4. Yarma – Dolgu Alanların Belirlenmesi	55
3.3.5. Çok-Amaçlı Algoritmalar Kullanarak X ve Y Eksen Üzerinde Optimizasyon ..	56
3.3.6. Kırmızı Çizgiye Ait Çok-Amaçlı Meta-Sezgisel Akış Şeması	58
3.4. Kırmızı Çizgiye Ait Düşey Kurp Optimizasyonu	60
3.4.1. Düşey Kurp Kodlaması	61
3.5. Kırmızı Çizgiye Ait Optimizasyonun Akış Şeması	61
4. BULGULAR VE TARTIŞMA.....	63
4.1. Tek-Amaçlı Optimizasyon Algoritmaları ile İlgili Bulgular.....	63
4.1.1. Some Noktalarının Siyah Çizgi Üzerindeki Koordinatı.....	65
4.1.2. Y Eksen Üzerinde Tek-Amaçlı Meta-Sezgisel Algoritmaların Uygulanması	68
4.1.3. X-Y Eksen Üzerinde Tek-Amaçlı Meta-Sezgisel Algoritmaların Uygulanması	71
4.1.4. Geliştirilmiş Tek-Amaçlı Meta-Sezgisel Algoritma Çıktıları.....	75
4.1.5. Y ve X-Y Eksenleri Optimizasyonu İçin Karşılaştırma Diyagramları	78
4.2. Çok-Amaçlı Optimizasyon Algoritmaları ile İlgili Bulgular	80
4.2.1. Kırmızı Çizgi ile İlgili NSGA3 Optimizasyon Sonuçları	81
4.2.2. Kırmızı Çizgi ile İlgili MOPSO Optimizasyon Sonuçları.....	83
4.2.3. Kırmızı Çizgi ile İlgili SPEA2 Optimizasyon Sonuçları.....	85
4.2.4. Çok-amaçlı meta-sezgisel optimizasyonların karşılaştırma diyagramları	87
4.3. Meta-Sezgisel Algoritmaları Kullanarak Düşey Kurp Optimizasyonu.....	89
4.3.1. Meta-Sezgisel Algoritmalar ile Düşey Kurp Uygulaması	89
4.3.2. Düşey kurp optimizasyonu.....	91
4.3.3. Düşey Kurban Siyah Çizgi Üzerindeki Yükseklik Kotları	93
4.3.4. Optimum Düşey Kurbanın Seçilmesi	94
5. SONUÇ	95
KAYNAKLAR	101
EKLER.....	105
ÖZ GEÇMİŞ.....	121

ŞEKİLLER DİZİNİ

Şekil 2.1. Yol enkesiti ve elemanları (Yayla, 2008)	10
Şekil 2.2. Geçki yatay geometrisi üzerinde kilometre tanımlanması (Baykal, 2009)	12
Şekil 2.3. Boykesit elemanları (Baykal, 2009)	14
Şekil 2.4. Tip enkesitte kırmızı ve siyah çizgi kavramları (Baykal, 2009)	15
Şekil 2.5. Tepe ve dere düşey kurp tipleri (Yayla, 2008)	17
Şekil 2.6. Dairesel düşey kurba (Yayla, 2008)	19
Şekil 2.7. Parçacık sürü optimizasyonun akış diyagramı (Hassan et al., 2005)	21
Şekil 2.8. FA'nın akış diyagramı (Yang, 2010)	22
Şekil 2.9. ABC'nın akış diyagramı	23
Şekil 3.1. Örnek bir siyah çizgi kotları ve kilometrajları	33
Şekil 3.2. Örnek kırmızı çizgi	34
Şekil 3.3 Siyah çizgi	35
Şekil 3.4. Some noktalarının belirlenmesi	35
Şekil 3.5. Zorunlu some nokta eklenmesi	36
Şekil 3.6. Zorunlu some noktası eklenmesinin yazılım kodlaması	37
Şekil 3.7. Some noktası eklenmesinin yazılım kodlaması	38
Şekil 3.8. Some noktası bulunmaması gereken aralık	38
Şekil 3.9. Some noktası kaldırımı için gereken yazılım kodlaması	39
Şekil 3.10. Maksimum ve minimum eğimler	40
Şekil 3.11. Maksimum ve minimum eğimlerin yazılım kodlaması	40
Şekil 3.12. Yarma – dolgu alanları	41
Şekil 3.13. Yarma – dolgu alanlarındaki kırılma noktaları	42
Şekil 3.14. Some noktasının arazi üzerindeki durumu	43
Şekil 3.15. Some noktasının arazi üzerindeki durumu	44
Şekil 3.16. Some noktaları (Özkan, 2013)	45
Şekil 3.17. Her some noktası için farklı kırmızı çizgi (Özkan, 2013)	46
Şekil 3.18. Her some noktası için matris oluşumu (Özkan, 2013)	46
Şekil 3.19. Her some noktası için matris oluşumu	47
Şekil 3.20. Her some noktası için değişkenlerin kodlaması	47
Şekil 3.21. Y ekseninde denetleme	48
Şekil 3.22. X ve Y ekseninde denetleme	50
Şekil 3.23. Tek-amaçlı meta-sezgisel algoritmaya ilişkin akış şeması	52
Şekil 3.24. Siyah çizgi	54
Şekil 3.25. Kırmızı çizgi	54
Şekil 3.26. Yarma – Dolgu Alanları	55
Şekil 3.27. Tüm verilerin parçalar haline getirilmesi	56

Şekil 3.28. X ve Y ekseninde denetleme.....	57
Şekil 3.29. Çok-amaçlı meta-sezgisel algoritmaya ilişkin akış şeması.....	59
Şekil 3.30. Some noktasına düşey kurba yerleştirilmesi.....	60
Şekil 3.31. Some noktasına düşey kurp yerleştirilmesi.....	60
Şekil 3.32. Düşey kurp yazılım kodlaması.....	61
Şekil 3.33. Kırmızı çizgiye ait optimizasyon akış şeması.....	62
Şekil 4.1. Doğru parçaların optimizasyonu.....	63
Şekil 4.2. Some noktasının belirlenmesi.....	64
Şekil 4.3. Y ekseninde açılımı.....	65
Şekil 4.4. X-Y ekseninde açılımı.....	65
Şekil 4.5. Doğru çizginin belirlenmesi.....	66
Şekil 4.6. Siyah kotun bulunması.....	66
Şekil 4.7. Siyah kotun bulunması.....	67
Şekil 4.8. Some noktalarının siyah kotları.....	67
Şekil 4.9. PSO yöntemi Y ekseninde.....	69
Şekil 4.10. FA yöntemi Y ekseninde.....	70
Şekil 4.11. ABC yöntemi Y ekseninde.....	71
Şekil 4.12. PSO yöntemi X-Y ekseninde.....	73
Şekil 4.13. FA yöntemi X-Y ekseninde.....	74
Şekil 4.14. ABC yöntemi X-Y ekseninde.....	75
Şekil 4.15. Y ekseninde için yapılan denemeler.....	76
Şekil 4.16. X-Y ekseninde için yapılan denemeler.....	76
Şekil 4.17. PSO yöntemi ile en optimum kırmızı çizgi.....	77
Şekil 4.18. FA yöntemi ile en optimum kırmızı çizgi.....	77
Şekil 4.19. ABC yöntemi ile en optimum kırmızı çizgi.....	77
Şekil 4.20. Y ekseninde yapılan karşılaştırma grafiği.....	78
Şekil 4.21. X-Y ekseninde yapılan karşılaştırma grafiği.....	79
Şekil 4.22. Doğru parçaların ayrı ayrı optimizasyonu.....	80
Şekil 4.23. Some noktalarının parçalanma durumu.....	81
Şekil 4.24. X-Y ekseninde açılımı.....	81
Şekil 4.25. NSGA3 yönteminin sonuçları.....	83
Şekil 4.26. MOPSO yönteminin sonuçları.....	85
Şekil 4.27. SPEA2 yönteminin sonuçları.....	87
Şekil 4.28. NSGA3 ile ilgili karşılaştırma grafiği.....	88
Şekil 4.29. MOPSO ile ilgili karşılaştırma grafiği.....	88
Şekil 4.30. SPEA2 ile ilgili karşılaştırma grafiği.....	88
Şekil 4.31. Some noktasına düşey kurba yerleştirilmesi.....	89

Şekil 4.32. Some noktasına düşey kurp yerleştirilmesi.....	90
Şekil 4.33. Düşey kurp yazılım kodlaması	91
Şekil 4.34. Kırmızı çizgilere uygulanan düşey kurbalar	92
Şekil 4.35. Her some noktası için ayrı bir düşey kurp uygulaması.....	92
Şekil 4.36. X ve Y ekseninde some noktalarının yer değişimi ve düşey kurba	92
Şekil 4.37. Düşey kurba ve siyah çizgi ilişkisinin yazılım kodlaması	93
Şekil 4.38. Düşey kurbanın siyah çizgi üzerindeki yüksekliği	93
Şekil 4.39. Örnek bir optimum tepe düşey kur	94



TABLÖLAR DİZİNİ

Tablo 2.1. Dairesel düşey kurba yarıçapları (Yayla, 2008)	19
Tablo 3.1. Y eksenli değışkenleri.....	49
Tablo 3.2. X-Y eksenli değışkenleri	50
Tablo 4.1. Y eksenli için tek-amaçlı meta-sezgisel optimizasyon sonuçları.....	68
Tablo 4.2. X-Y eksenli için optimizasyon sonuçları.....	72
Tablo 4.3. <i>NSGA3</i> yöntemi ile yarma-dolgu alanlarının farkı	82
Tablo 4.4. <i>MOPSO</i> yöntemi ile yarma-dolgu alanlarının farkı.....	84
Tablo 4.5. <i>SPEA2</i> yöntemi ile yarma-dolgu alanlarının farkı	86



1. GİRİŞ

Karayolu yapım maliyetinin yüksek olmasından dolayı, projelendirme aşamasında bir dizi mühendislik çalışması gerekmektedir. Bu bağlamda ülkemizde ve genel olarak dünyada birçok çalışma yapılmıştır. Bu bağlamda, çok sayıda optimizasyon yöntemi geliştirilmiştir. Karayolu tasarımı yeniliklere açık olduğundan günümüzde hala çalışmalar tüm hızıyla devam etmektedir.

Bir karayolu projesi altyapı ve üstyapıdan oluşmaktadır. Toprak işleri, yani yarma ve dolgu işleri sonunda oluşturulan tesviye yüzeyinin altında kalan taban zeminine altyapı çalışmaları adı verilmektedir. Üstyapının performansı altyapının fiziksel özellikleri ile doğrudan ilişkilidir. Bu bakımdan her zaman tüm şartların sağlanması gerekmektedir. Üstyapı ise tabakalı bir yapıya sahip olup ve trafik yüklerini doğrudan taban zeminine aktarmaktadır. Üst yapı, ekonomik ömrü boyunca büyük bozulmalar ve çatlama ile karşı karşıya kalmadan, yolun trafik yüklerini taşıyabilecek kısımdır (Sütaş ve Öztaş, 1983).

Mühendislik eğitiminde bilgisayar programlama dili sayısal hesaplamalara büyük ölçüde yardımcı olacaktır. Böylece problemin matematik modeli kurularak programlama tekniğine ulaşılmaktadır. Yazılımda birkaç değer girilerek çok hızlı bir şekilde sonuç alınmaktadır.

Tüm mühendislik alanlarının tasarımdan uygulamasına kadar optimizasyonun her yerde olduğu yadsınamaz bir gerçektir. Matematiksel optimizasyon veya programlama tüm tasarım planlama problemlerinde kullanılmaktadır. Bu bağlamda tüm faktörleri göz önünde bulundurarak en ekonomik veya en yüksek erişilebilir performansa sahip bir sonuç elde edilmektedir. Bir optimizasyonun kullanım amacı, üretim verimliliğini en üst düzeye çıkarmaktır (Yang and Press, 2010).

Mühendislik tasarımı ve buna bağlı olarak, iş planlaması ile daha detaylı çalışma rotası belirlenmesi ve hatta tatil planlaması için optimizasyonun her yerde olduğu yadsınamaz bir gerçektir. Genel olarak tüm bu faaliyetlerde, bazı hedeflere ulaşmaya ya da kalite, kâr ve zaman gibi bir kavramı optimize etmeye çalışılmıştır. Günümüzde tüm uygulamalarda kaynaklar olarak, para ve zaman daima sınırlıdır. Bu değerli kaynaklar, çeşitli kısıtlamalar altında çözümler bulunarak en iyi şekilde kullanılmalıdır. Matematiksel optimizasyon veya programlama, tasarım ile ilgili problemlerde kullanılmaktadır. Tüm kısıtlamalara rağmen, istenmeyen faktörleri en

aza indirgeyerek ve istenen faktörleri en üst düzeye çıkararak en ekonomik veya en yüksek performansa sahip bir seçenek bulunmalıdır. Bir optimizasyonun tasarımında, tasarım amacı basitçe üretim verimliliğini en üst düzeye çıkarmak veya üretim kârını en aza indirmektir. Optimizasyon uygulamalarında hangi bilgilerin eksik olması ve hangi bilgilerin mevcut olduğunun değerlendirmesinde zaman sınırlamaları söz konusudur. Günümüzde bilgisayar simülasyonlarının çözümü için çeşitli arama algoritmaları kullanılmaktadır. (Yang and Press, 2010).

Karayolu uygulaması sırasında projenin denetimi için bilgisayar yazılımı yardımıyla bazı hızlı çözümlerin üretilmesi gerekmektedir. Bu çözümler sayesinde karayolu projelerinin uygulaması daha hızlı yapılmaktadır. Karayolu tasarımı, inşaat maliyetini doğrudan etkilemektedir. 1960'lardan beri optimum tasarım çalışmaları başlatılmıştır (Özkan, 2013).

Teorik olarak, optimizasyon konusunda sonsuz sayıda alternatif bulunmaktadır. Bu bağlamda, mühendislik sektöründeki uygulamaları optimize etmek için problemlerin formüle edilmesi gerekmektedir. (Kim et.al., 2005).

Özkan et al. (2021), yaptıkları çalışmada birkaç alternatif yatay güzergâh oluşturmuşlardır. Karayolu tasarımı sırasında bu alternatif yatay güzergâhlar için ayrı ayrı geometrik tasarımlar gerçekleştirmiş ve seçilen bir yatay güzergâha uygun binlerce düşey güzergâh olabileceğinden, bir optimizasyon metodu geliştirerek en ekonomik tasarıma ulaşmaya çalışılmıştır. Düşey güzergâhın ana maliyet olarak toprak işleri maliyeti kabul edilmiştir. Optimizasyon işlemini hızlı bir şekilde uygulamak için AutoCAD Civil 3D yazılımı ile Visual Basic programlama dili kullanılarak bir program modülü hazırlanmıştır. Böylece, toprak işleri maliyetini %27'ye varan oranlarda düşürebilmişlerdir.

Bir karayolu projesi tasarım ve uygulamadan ibarettir. Karayolu projelerinin uygulaması ne kadar ince detaylardan oluşuyorsa tasarım aşamasına bir o kadar dikkat edilmelidir. Başka bir deyişle uygulama sırasında nasıl ki yanlış uygulamalar kabul edilmeyecek sonuçlar oluşturuyorsa, proje tasarım aşamasında da gerekli şartnamelere göre uygun tasarlanmayan projelerde, uygulama maliyetlerini artıracak ve ekonomik olmamasına neden olacaktır.

Karayolları inşaatı sırasında önce projenin tasarımı el ile veya bir bilgisayar kullanarak yapılabilir. Bu nedenle tasarlanması gereken yolun tüm detayları

yani tüm kırmızı çizgi eğimleri ve güzergâhlar ve aynı zamanda tüm yatay ve düşey kurpların detayları karayolları şartnamesine göre seçilmelidir.

Bir karayolu projesinin hazırlanmasında, tasarım aşamasına başlamadan önce ilk aşama o projeye ait arazi durumunun incelenmesi ve bu bağlamda gereken en önemli verilerden biri olan arazinin topografisinin dikkatli bir şekilde çıkartılmasıdır ve buna bağlı olarak tüm yol sınıflarının kriterlerini de göz önünde bulundurarak en iyi güzergahın seçilmesidir.

Bir sonraki aşama, arazi verilerinin hatasız bir şekilde toplanmasından sonra bu verilere göre güzergahların dikkatle seçilmesidir. Tabi ki güzergâh seçimi ne kadar dikkatli yapılırsa projenin ekonomik olmasına o kadar yardımcı olacaktır. Ancak, güzergâh seçimi, bir karayolu projesinin ekonomik olup olmaması için önemli bir faktör sayılabilir ama tam olarak bir karayolu projesinin ekonomik olmasında yeterli olmayacaktır.

Bir karayolu projesinin tam olarak ekonomik olması isteniliyorsa, o projeye ait kazı işlemleri en az miktara indirilmelidir ve aynı zamanda yarma-dolgu hacimleri birbirine eşitlenmelidir. Bu durumu özetlemek gerekirse bir karayolu projesini en ekonomik hale getirmek için o projenin yarma-dolguları üzerinde çalışmalar yapılmalıdır. Bu bağlamda kazı işlemlerini eşitlemek ve aynı zamanda en aza indirmek için boykesit tasarımı üzerinde optimizasyon yapılmalıdır.

Boykesit üzerinde bulunan veriler, sadece siyah çizgi ve kırmızı çizgiye ait verilerdir. Ancak, arazinin doğal durumunu gösteren siyah çizgiye müdahale etmek pek mümkün olmayacaktır. Bu durumda, geriye kalan ve müdahale edebilmemizi sağlayacak bir tek seçenek vardır ve o da kırmızı çizgi veya proje çizgisi dediğimiz faktördür. Bu bağlamda, bir karayolu projesinin ekonomik olması için o projeye ait kırmızı çizginin optimizasyonunun yapılması gerekmektedir.

Literatürdeki çalışmaların optimizasyon yöntemleri kendi kapsamlarında çok faydalı olabilir. Ancak, hiçbiri yeterli olmamaktadır. Yapılan çalışmalarda, yarma ve dolgu miktarları hesaplanmış, bu kapsamda bir model oluşturulmuş ancak söz konusu modellerde some noktalarının otomatik bulunması ve eğim gibi parametreler göz önünde bulundurulmamıştır. Kırmızı çizgi tasarımı için hafriyat miktarının en az olması ve yarma-dolgu miktarının eşit çıkarılabilmesi amacıyla bazı optimizasyon yöntemlerin kullanılması yadsınamaz bir gerçektir. Ancak, şimdiye kadar kırmızı çizgi

ile ilgili yapılan çalışmalarda, meta-sezgisel algoritmalarından sadece bazı tek amaçlı optimizasyon algoritmaları kullanılmıştır. Bu bağlamda kullanılan yöntemler genel olarak sadece yatay güzergâh optimizasyonu için kullanılmışlardır.

Bu tez çalışmasında; Optimum kırmızı çizgi tasarımı için öncelikle üç tane farklı tek amaçlı meta sezgisel yöntem ile bir kırmızı çizgi tasarımı gerçekleştirilecektir. Tasarlanan kırmızı çizgilerin siyah çizgiye göre yukarı ve aşağısında kalan bölgelerin alanları hesaplanacaktır. Bu çalışmada, MATLAB programı üzerinden tek-amaçlı optimizasyon algoritmalarından en popüler olanlarından üç tanesi (PSO, ABC ve FA) kullanılmıştır. Bu yöntemler sayesinde kırmızı çizgi, siyah çizgi üzerinden geçirilecektir ve çizginin yukarısında yani yarma bölgelerinin alanları tek tek hesaplanacaktır ve aynı zamanda çizginin altında kalan bölgeleri yani dolgu kısımlarının da alanları hesaplanmış olacaktır. Ayrıca, yarma-dolgu farkı da hesaplanıp optimizasyon yöntemlerine göre belirli sayıda tekrarlanacaktır ve en son bu fark sıfıra yakın bir değer olana kadar devam edilecektir. Bu bağlamda, bulunan çizgi en optimum kırmızı çizgi olarak kabul edilecektir. Başka bir deyişle, optimizasyon yöntemleri binlerce deneme yanılma yapılarak alan farklarının sıfıra yakın bir değer bulunmasına yardımcı olacaklardır. Bu şekilde bulunan çizgi aslında yöntemlerin alana göre optimizasyon yazılımı sayesinde gerçekleşecektir. Bu çalışmada hazırlanan algoritma, tek amaçlı meta sezgisel algoritmalara entegre edilmiş olacaktır. Daha sonra her üç yöntemi de ayrı ayrı hem Y ekseninde hem de X-Y ekseninde 2000 defa ile 30 kez tekrarlayıp ve karşılaştırması yapılacaktır. Bu bağlamda, üst sezgisel yöntemler kullanılarak alan farklarının sıfıra yakın bir değer bulunmasına yardımcı olacaktır. Bu şekilde bulunan çizgi, aslında yöntemlerin alana göre optimizasyon yazılımı sayesinde gerçekleşmektedir. Bu çalışma kapsamında daha önce dikkate alınmamış tüm kırmızı çizgi tasarım parametreleri dikkate alınmaktadır. Bu bağlamda, bu parametrelerden en önemlileri yani eğim sınır şartları, some noktalarının otomatik bulunması, arazinin durumuna göre pik noktaların ekleme çıkartması ve doğal zemine ait siyah kotların bulunması da bu çalışmada mevcut olan algoritmaya entegre edilmiştir.

Bu çalışmada, tasarlanan kırmızı çizgilerin siyah çizgiye göre optimizasyonu yapılmaktadır. Bu bağlamda, tek-amaçlı meta-sezgisel algoritmaların dışında, çok-amaçlı meta-sezgisel optimizasyon algoritmalarını da kullanarak kırmızı çizgi optimizasyonu yapılabilmektedir. Ancak, günümüzde kırmızı çizgi optimizasyonunda

tek-amaçlı yöntemlerden GA ve PSO yöntemi tüm parametreleri göz önünde bulundurmadan kullanılmaktadır. Ama, kırmızı çizgi tasarımı fazlasıyla karmaşık matematiksel işlemlerden oluştuğu için optimizasyon yöntemlerinden çok-amaçlı optimizasyon algoritmalarının kullanımı daha mantıklı olacaktır. Bu bağlamda, çok-amaçlı optimizasyon algoritmalarından en popüler olanlarından üç tanesi (*NSGA3*, *MOPSO* ve *SPEA2*) mercek altına alınacaktır. Böylece, kırmızı çizginin optimizasyonu için hazırlanan yazılım ile çok-amaçlı optimizasyon algoritmaları entegre edilerek daha az hafriyat miktarı ve yarma-dolgu dengelemesi yapılacaktır. Bu bağlamda, çok amaçlı yöntemlerde iki faktör veya iki hedef üzerinden yola çıkarak daha detaylı inceleme yapılmış olacaktır. Böylece, tasarlanan algoritma sayesinde sonuçlar daha istikrarlı ve güvenilir olabilecektir. Tabi ki bu işlemler her 3 meta-sezgisel optimizasyon yöntemi için de ayrı ayrı denenecektir. Daha sonra her üç yöntemi de ayrı ayrı X-Y ekseninde 2000 iterasyonla 30 kez tekrarlayıp ve karşılaştırması yapılacaktır.

2. KURAMSAL TEMELLER VE KAYNAK ÖZETLERİ

2.1. Karayolu Projelerinin Uygulanması

Karayolu uygulama projelerinin aplikasyon işlemlerinin organize edilmesi, jeodezi mühendisinin görevleri kapsamındadır. Projenin işleyişinin denetimi Jeodezi mühendisleri tarafından titizlikle yapılmaktadır (Baykal, 2009; Tunç, 2004).

2.1.1. Altyapı Çalışmaları

Bir karayolu projesi için tasarlanmış olan güzergâh, yatay ve düşey dönemeçler ile enkesitlerin araziye applike edilme işlemidir. Bu bağlamda, uygulanması gereken yolun arazi şeridinde, bazı kısımlar kazılır ve bazı kısımlar doldurulur; bu yarma ve dolgu işlemlerinin tümüne toprak işleri adı verilmektedir (Evren, 1994; Umar ve Yayla, 1997).

Yol projelerinin uygulamasında önce şevlerin kazıkları çakılır. Yarma ve dolgu enkesitlerinin doğal zeminle kesişen noktaları olan bu kazıklar aslında iş makinelerinin yarma-dolgu yapacağı yani toprak işlerinin arazi şeridini belirlemiş olacaklardır (Baykal, 2009; Tunç, 2004).

Karayolu eksen hattı arazide kazıklarla işaretlenir. Bu işlem genelde her 20 metre de bir yapılmaktadır. Ancak, inşaat uygulama sırasında arazinin tamamına hakim olma açısından daha fazla sayıda yani daha sık kazıkların aplikasyonu istenmektedir. Bu durumda ekstradan eklenen kazık noktalarından enkesit çıkarılması gerekmektedir. Bu işlem altyapı çalışması bitene kadar tekrarlanabilir (Nostrand, 1992).

Yol projesinde her eksen hattının enkesit doğrultuları uygun şekilde belirlenir ve şev eğimleri arazide düzgün bir şekilde ayarlanır. İnşaatın uygulamasında, kırmızı kotların arazide gerçekleşmesi için projenin bazı aşamalarında yükseklik ölçmeleri düzgün ve dikkatli bir şekilde yapılmalıdır (Baykal, 2009; Wright and Ashford, 1989).

Bir karayolu projesinde kazı işleri sonunda elde ettiğimiz yüzeye tesviye yüzeyi adı verilir. Projede tasarlanan boyuna ve enine eğimler greyder kullanarak reglaj yapılır ve uygulamanın son kontrolü için kırmızı çizginin kotları incelenir ve altyapı çalışmaları bitmiş olur. Toprak işi uygulamalarında istenilen doğrulukların aplikasyonu yapılırken, yatayda ve düşeyde $\pm (1-2)$ cm mertebesi hata payı olarak

kabul edilmektedir (Baykal, 2009; Müller, 1988; Shepherd, 1983; Umar ve Yayla, 1997).

2.1.2. Karayolu Üst Yapı Tasarımı

Demir yollarında alt yapı, temel, balast tabakası, travers ve raylardan oluşur. Ancak, karayolları alt yapısı, alt temel, temel ve kaplama tabakalarından oluşmaktadır. Bu tabakaların kalınlığı yolun trafik akışına, zeminin tipine, altyapının taşıma gücüne ve diğer faktörlere bağlı olarak belirlenir. Kara yollarında yuvarlanma yüzeyinin oluşturulmasını kaplama tabakası ve demir yollarında ise bu yüzeyi raylar oluşturmaktadır. Yolculuğun güvenliğini ve konforunu sağlamak için bu yüzeylerin düzgün olması gerekmektedir. Karayollarında kaplama tabakasından talep edilen sınır değerleri, otoyollarda $\pm (1-3)$ mm, normal karayollarında $\pm (5-6)$ mm, verilebilir. Ray uygulamasında ise talep edilen sınır değerleri proje hızına bağlı olup örnek olarak 120 km/s için yatay ve düşey kısımlarında $\pm (0.6-0.8)$ mm kabul edilmektedir (Darke, 1967).

2.2. Karayolu ve Elemanları

Karayolu mühendisliğinde genel olarak kullanılan bazı önemli terimler ve yol elemanlarının tanımları aşağıda kısaca açıklanmıştır.

2.2.1. Güzergâh-Plan-Boykesit

Bir karayolun arazideki izlediği doğrultuya güzergâh denilir. Dolayısıyla geçki bir bakıma yolun izdüşümüdür.

Yolun yatay bir düzlem üzerindeki izdüşümüne plan adı verilir. Bir yolun eksen hattı, planda aliymanlar ile bunları birbirine bağlayan eğri kısımlar yani yatay kurplardan oluşur. Bir yatay kurp; aliyman dediğimiz doğru parçaları, ve bunları izleyen geçiş eğrisi ile dairesel yaydan oluşur.

Plan üzerindeki yol ekseninin düşey bir düzlemde izdüşümü alınır boykesiti ya da Başka bir deyişle profili elde edilmiş olacaktır. Boykesit çıkışlardan ve inişlerden yani eğimleri farklı düz kısımlardan ve bunları birbirine bağlayan dairesel eğri kısımlardan oluşur. Boykesit üzerindeki eğri kısımlara düşey kurba ve eğimli düz kısımlara rampa adı verilir. Düşey kurbalar parabolik veya dairesel, rampalar ise iniş veya çıkış eğimli olur (Bruce and Clarkeson, 1950; Hickerson, 1959; Yayla, 2008).

2.2.2. Toprak İşi

Taşıt ve yayaların karayolunu daha konforlu kullanabilmeleri için doğal zeminin buna uygun hale getirilmesi gerekir. Bu nedenle yol enkesitlerinin bazı kısımlar kazılır, bazı kısımlar ise doldurulur. Bu kazı işlemine toprak işi adı verilir. Doğal zemin üzerinde yapılan bu düzeltilmeye uygulamada tesviye yüzeyi adı verilir. Başka bir deyişle, toprak işi sonunda ulaşılan veya ortaya çıkan yüzey tesviye yüzeyidir. Daha sonra tesviye yüzeyine daha ince tesviye (reglaj) işlemi yapılmak üzere bir greyder yardımı ile uygun enine ve boyuna eğim verilmektedir. Bu reglaj işlemi yapıldıktan sonra yol üstyapı inşasına artık başlanabilir (Bruce and Clarkeson, 1950; Hickerson, 1959; Yayla, 2008).

2.2.3. Altyapı ve Üstyapı Tanımı

Yolun, toprak işleri bittiğinde, o yola ait belirlenmiş yükseklik ve enkesit tiplerinin yapılması demektir. Bu da altyapı işlerinin bitmesi anlamına gelmektedir. Köprü, viyadük, drenaj tesisleri, tünel, menfez ve istinat duvarı gibi sanat yapıları da altyapı içine girer. Altyapı yolun esas taşıyıcı kısmıdır. Ancak, altyapı üzerine başka tabakalar da inşa edilerek taşıyıcılık görevini iyi bir şekilde yapabilmesine yardımcı olacaktır.

Yolun, trafik yüklerini taşıyıp ve bu yükleri taban yüzeyine dağıtması gerekmektedir. Ancak, bu yükler zemininin taşıma gücünü aşmayacaktır. Altyapı üzerine inşa edilen ve alt temel, temel ve kaplama tabakalarından oluşan bu kısma üstyapı denilmektedir (Bruce and Clarkeson, 1950; Hickerson, 1959).

2.2.4. Alt Temel Tabakası

Toprak işleri sonunda ortaya çıkan tesviye yüzeyine serilen alt temel tabakasının genel olarak bir granülometrisi vardır. Genellikle, ince malzemesi az, çakıl, kırma taş, kum, yüksek fırın cürufu gibi granüler malzemeden oluşmaktadır. Kaplama yüzeyinden gelen trafik yükleri taban zemin üzerine yayılmaktadır. Bu nedenle, yol üzerinde bulunan temel tabakasına yardımcı olmanın yanı sıra, su ve soğuk havalarda oluşan don tesirlerine karşı tampon bölge görevi de gören bu tabakanın teşkili ile daha ekonomik malzemelerden inşa olunan temel tabakası kalınlığı azaltılmış olacaktır. Böylece, taban zemininin durumuna göre, az miktarda olsa bile, bazı durumlarda bu tabakanın yapılışından vazgeçilebilir (Bruce and Clarkeson, 1950; Hickerson, 1959; Kumbasar vd., 1969).

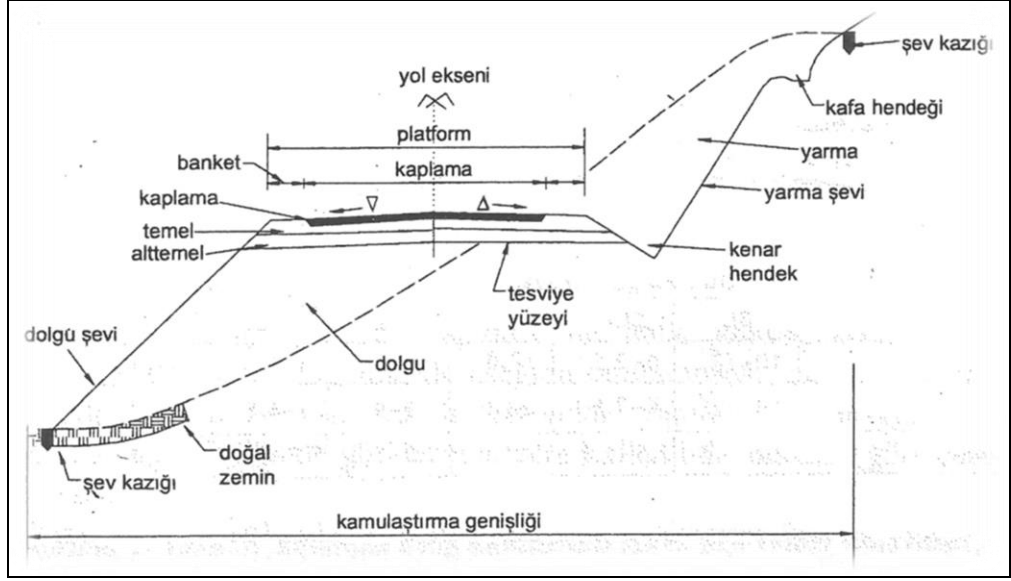
2.2.5. Temel Tabakası

Temel tabakası, alt temel ve kaplama tabakası arasına yerleştirilmektedir. Bu tabakanın alt temel tabakasına göre granülometrisi ile malzemesinin fiziki özellikleri daha iyi olmaktadır. Bu tabakada kullanılan malzemeler, doğal kum, kırmataş veya doğal çakıl ile az miktarda bağlayıcı ince malzemedir. Yol gövdesinde çok önemi olan bu tabakanın ilk görevi kaplamadan gelen trafik yükünü, önce alt temel tabakasına ve daha sonra taban üzerine yaymakla beraber trafiğin darbe tesirlerini önemli miktarda azaltmaktır.

Kaplama tabakası, trafiğin doğrudan temas ettiği, temel tabakası üzerine inşa edilmektedir. Bu tabaka genellikle bitümlü karışımlar, beton, parke vb. Malzemeler ile yapılmaktadır. Bir ya da birkaç tabaka halinde inşa edilen bu tabakanın asıl görevi düzgün bir yuvarlanma yüzeyi temin etmektir. Bu tabakalar en yaygın olarak bitümlü karışımlar kullanarak beton asfaltı halinde yapılmaktadır. Bu tabakanın trafiğin ve iklim koşulların bozucu etkilerine karşı koymaları için en üst tabakada aşınma tabakası yapılmaktadır. Bu tabakanın altındaki tabaka ise binder tabakası adı verilmektedir.

Karayolu üstyapısı kaplama, temel ve alt temel tabakalarından oluşmaktadır. Bu tabakaların kalınlıkları aslında geçecek olan trafiğin miktarına ve cinsine bağlı olmakla birlikte iklim koşulları ve taban zemininin taşıma gücü gibi faktörler ile de değişmektedir.

Karayolu elemanlarının bazı tanımları yukarıda yapıldı ve yapılacak olan diğer yol elemanlar ise Şekil 2.1’de gösterilmiştir (Bruce and Clarkeson, 1950; Yayla, 2008).



Şekil 2.1. Yol enkesiti ve elemanları (Yayla, 2008)

2.2.6. Şev

Bir karayolunun yarma kısmında, kenar hendek tabanı ile arazi arasındaki eğri yüzey, dolgu kısmında ise platformun dış kenarı ile doğal zemin arasında kalan eğik kısma şev adı verilir. Şev eğimi belirlemede iki önemli faktör vardır, birincisi dolguların veya yarmaların yükseklikleridir ve ikincisi zeminin kendini tutma özelliğidir. Şev eğiminin, kendisini tutan zeminlerde dik olmasına bir sakınca yoktur. Yüksek dolgularda şev eğimini az tutarak daha geniş bir arazi şeridi uygulanır bu da trafiğin güvenlik ve konforu yönünden daha çok tercih edilmektedir. Kaya yarmalarda dik şev kullanılır. Dolgu şevleri için en çok kullanılan eğimler 3/1, 3/2, ve 4/1 olarak kabul edilmektedir. Yarma şevlerinde ise en çok 1/1, 2/1, 1/2 ve 3/2 değerleri kullanılmaktadır. Örnek olarak 1/3 şev; yatayda bir, düşeyde üç birim demektir (Meyer, 1968; Yayla, 2008).

2.2.7. Boyuna ve Enine Eğimler

Yağmur sularının yol yüzeyine geldiğinde platformu çok hızlı bir şekilde terk edebilmeleri gerekmektedir. Bunun için yol enkesitine eksenden etrafa doğru enine eğim verilmelidir. Enine eğim miktarının belirlenmesinde kaplama cinsi, çok önemli bir faktördür. Yağmur sularının kısa bir süre içinde akabilmeleri için 0.01 ile 0.02 arasında bir eğimli ince taneli asfalt kaplamalar seçilebilir. Toprak yollarda 0,04 ile 0.06, kaba agregalardan oluşan kaplamalarda ise enine eğim 0.03 ile 0.04, mertebesinde olmaktadır. Beton yollarda minimum eğim 0.015 ve maksimum eğim ise 0.02 olmalıdır. Yol projelerinde banketler drenaj amacı ile yapılmaktadır ve genellikle

kaplama eğiminden daha fazla tutulur. Kırmataş ve çakıl kaplamalı banketlerde 0.06 ve 0.04 aynı şekilde toprak olarak bırakılmış banketlerde ise 0.08 enine eğim önerilmektedir.

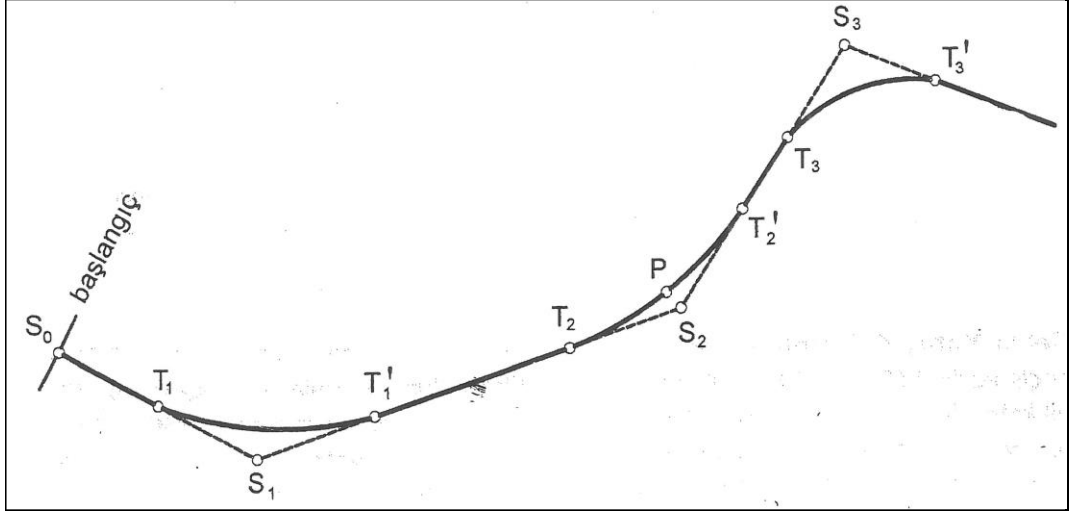
Yol projelerinde boyuna eğim adı verilen eğim yol boyunca boyuna doğrultuda verilmektedir. Yolun sınıfı ile geçtiği arazinin topografik durumu boyuna eğimi sınırlayan başlıca faktörlerdendir. Düz arazilerde ve standartları daha yüksek olan yollarda boyuna eğim daha az tutulmaktadır. Arazinin durumu uygun olsa bile yüzey suların drenajı için yola en az 0,003 ile 0,005 arasında bir boyuna eğim verilmektedir (Monejem, 2005; Oglesby and Hicks, 1984; Tyler, 1987; Yayla, 2008).

2.2.8. Yolun Eksen Çizgisi

Yol eksen hattı yola ait kaplamanın tam ortasından geçtiği varsayılan doğrultu demektir. Bölünmemiş yollarda yol platformunu iki kısma ayırarak karşı yönlerden gelen trafiğin daha güvenilir olması sağlanmaktadır. Başka bir deyişle her iki yöndeki trafiğin kullanabilecekleri yol kısmının gösterilmesi için yol üzerine boyuna doğrultuda bir çizgi çizilmektedir bu çizgiye eksen çizgisi adı verilmektedir. Eksen çizgisi kesin olmama koşulu ile genel olarak yol platformunu ortalamaktadır (Yayla, 2008).

2.2.9. Başlangıca Olan Uzaklık (Kilometre)

Rastgele bir geçki noktasının, başlangıç noktasına olan mesafesi ve geçki yatay geometrisi boyunca ölçülen yatay uzaklığıdır. Örnek olarak Şekil 2.2’de P noktasının kilometresi $\overline{S_0, T_1}$ doğru parça + $\overline{T_1, T'_1}$ eğri parça + $\overline{T'_1, T_2}$ doğru parça + $\overline{T_2, P}$ eğri parçasıdır. Karayolu projelerinde geçki dışındaki noktaların, örnek olarak some noktalarının kilometresine gerek yoktur. Karayollarında ve aynı şekilde demir yollarında nokta kilometreleri: $KM_P = \text{kilometre} + \text{metre}$ (örneğin 14+034,673) şeklinde yazılır (Baykal, 2009; Behbahani, 2011).



Şekil 2.2. Geçki yatay geometrisi üzerinde kilometre tanımlanması (Baykal, 2009)

2.3. Plan Tasarımı

2.3.1. Doğru Parçalar

Eğrilik yarıçapının her noktasında eğri miktarı sıfıra sabitlenirse bu parçaya doğru parçası denilir. Doğru uzunluğu sonsuzdur. Geçki elemanında belirli uzunluğa sahip olma şartıyla doğru parçalar kullanılır. Türkçe kaynaklara göre bu doğru parçasına tanjant, aliyman ve teğet uzunluğu da denmektedir (Bıçakcı, 1997; KGM, 2005; Kiper, 1988; Tunç, 2004; Umar ve Yayla, 1997).

2.3.2. Yatay Kurplar

Bir yol geçkisi düz kısımlar denilen aliymanlar ile bunları birbirine bağlayan eğri kısımlar yani yatay kurplardan oluşmaktadır. Doğru parçaları değiştirmeye faydalı olan yatay kurbalar kapasite ve yolculuk konforunu ve en önemlisi güvenliği arttırmaktadır. İyi bir yatay kurba projelendirmesi yapılmazsa, işletme hızı iyice düşeceği için yolun kapasitesi azalacaktır. Buna bağlı olarak, merkezkaç kuvvetin savrulma ve devrilmelere yol açtığı ve kazaları da artırmaya sebebiyet verecektir. Ayrıca, yolculuk konforu de kötüleşebilir.

Arazi durumu her ne kadar elverişli olsa bile bir yolun uzun bir kısmının düz olarak devam etmesini de istemeyiz. Çok uzun aliymanlarda yani yaklaşık 8~10 km'den daha uzun doğru parçalarda sürücünün dikkati dağılabilir. Ayrıca, yol boyunca orta ayırıcı yoksa geceleri far etkisinden dolayı gözlerin kamaşması artacaktır. Bu bağlamda, yolun yönü doğu-batı yönünde ise güneş etkisi söz konusu olabilir. Yukardaki durumlar sürücülerin kazaya karışma olasılığını artırabilir. Buna karşın,

uzun aliymanlar yerine daha kısa aliymanlar tasarlayıp ve aynı zamanda bunları büyük yarıçaplı kurbalarla birbirine bağlamak daha güvenilir olacaktır.

Karayolu projesinde mevcut olan yatay kurbalar genelde daire yaylardan oluşmaktadır. Ancak, kurba yarıçapları küçük uygulandığı takdirde eğer hız fazla ise aliymandan kurbaya geçiş noktalarında merkezkaç kuvveti oluştuğundan taşıtların dengelerinin bozulmasına sebep olacaktır. Ayrıca, bu denge bozukluğu yolcuları fazla derecede rahatsız edecektir. Bu yüzden yüksek standartlı karayollarında aliyman ile yatay kurba arasına özel eğrili geçişler yerleştirilir. Bu şekilde tasarlanan yollarda, karşılaşılan merkezkaç kuvvetin etkisini kurbaya giriş yapıldığında belirli bir uzunluk boyunca dağıtılarak azaltılacaktır. Böylece, daha güvenilir ve hızlı bir şekilde çok konforlu bir geçiş sağlanacaktır. Yatay kurbalarda, kurbanın içine doğru enine bir eğim verilerek merkezkaç kuvvetin etkisini daha fazla önlemek mümkün olacaktır (Yayla, 2008).

2.4. Boykesit Tasarımı

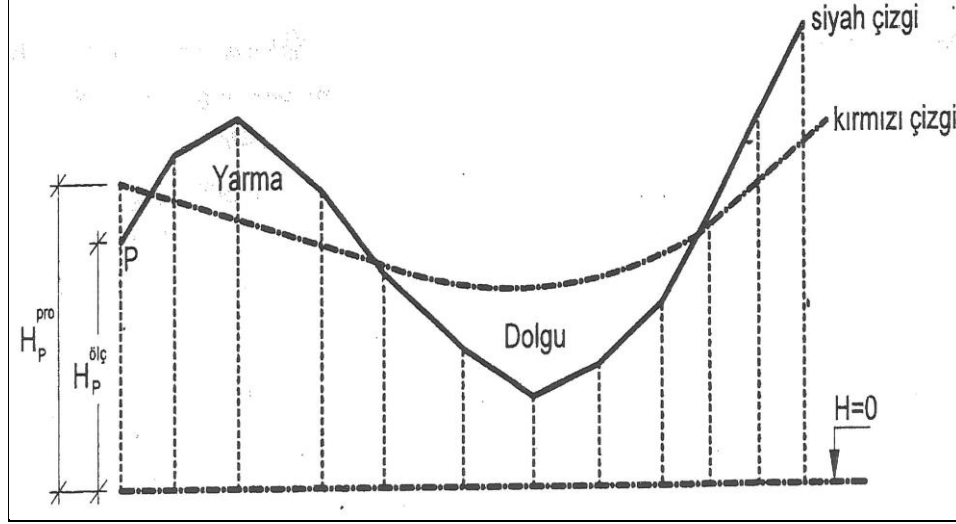
Geçkinin yatay geometrisi boyunca fiziksel yeryüzünün düşey yüzeylerle arakesitidir. Boykesit tasarımı için geçkinin yatay geometrisinin hesaplamaları ve tasarımı tamamlanmış olmalıdır (Baykal, 2009).

2.4.1. Siyah Kot

Boykesit üzerinde eksen kazığı noktalarının ve enkesitlerde arazinin karakteristik noktalarının yükseklikleri, arazinin doğal halini yani topografik yapısını göstermektedir.

2.4.2. Siyah Çizgi

Boykesit ve enkesitler üzerinde siyah kotu bilinen arazi noktalarının doğru parçaları kırık çizgiler ile birleşik olarak bir kırık çizgiler dizisi olarak Şekil 2.3'te gösterilmektedir. Başka bir deyişle, boykesit ve enkesit doğrultusu boyunca arazinin doğal halini grafik olarak göstermektedir (Baykal, 2009; Ceylan, 1993; Ceylan ve Baykal, 2005; Ceylan ve Baykal, 2008).



Şekil 2.3. Boykesit elemanları (Baykal, 2009)

2.4.3. Kırmızı Kot

Boykesit üzerinde tasarlanmış kırmızı çizgi noktalarının proje yükseklikleri olup hesapla bulunmasıdır (Baykal, 2009).

2.4.4. Kırmızı Çizgi

Boykesit üzerinde tasarlanan proje çizgisi veya geçki düşey geometrisinin grafik gösterimidir. Boykesit üzerinde arazinin doğal halini gösteren siyah çizgi üzerinden geçirilen ve yarma dolgu bölgelerini oluşturan çizgiye kırmızı çizgi denilmektedir (Baykal, 2009).

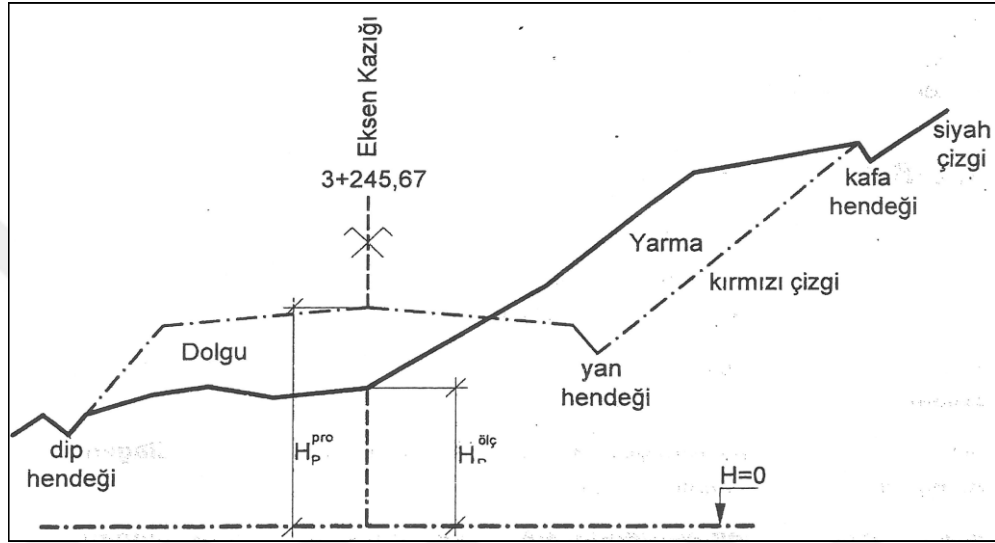
2.4.5. Enkesit

Karayolu projelerinde her eksen kazığından geçkiye dik doğrultuda çıkarılan kesitlere enkesit adı verilmektedir. Böylece, eksen kazığının kilometresi mutlaka enkesitlerin üzerine yazılmalıdır (Evren, 1994; Müller, 1988).

2.4.6. Tip Enkesit

Yatay geometrinin eğri kısımlarında enkesitlerin kırmızı çizgisini oluştururken dever uygulaması dikkatle yapılmalıdır. Ulaştırma yapısı boyunca genellikle tek bir tip enkesit yeterli olmayacaktır. Yüksek yarma ve dolgulara, köprü, tünellerde ve menfezlerde, istasyonlarda, kara yollarının tırmanma şeridi uygulanan bölgelerinde vd. istinat duvarı gerekli olabilir, bu durumda bazen özel tip enkesitler uygulanmaktadır (Evren, 1994; Müller, 1988).

Tip enkesit dolgu ve yarma şevlerinin eğimini, kara yollarında bölünmüş yol olup olmadığı, yan, kafa ve dip hendeklerinin boyutlarını ve banket ile şerit sayısı, kaplama ve orta refüj genişliklerini, hat genişliği ve hat eksenleri arasındaki uzaklık, demir yollarında hat sayısı gibi bilgileri göstermektedir. Bu bağlamda, tip enkesit, proje hızı ile düşünülmesi gereken bir karayolu yapısının standardını belirleyebilen en önemli elemanlardan biridir (Şekil 2.4) (Baykal, 2009; KGM, 2002; KGM, 2005; Manns, 1985).



Şekil 2.4. Tip enkesitte kırmızı ve siyah çizgi kavramları (Baykal, 2009)

2.4.7. Yarma-Dolgu İşleri

Boykesitte ve enkesitlerde yapılması gereken toprak işlerini, siyah ve kırmızı çizginin oluşturduğu kapalı bölgelerin alanları belirlemektedir. Kırmızı kotların siyah kotlardan büyük veya küçük olması o bölgenin yarma veya dolgu olduğunu göstermektedir. Karışık enkesitler hem dolgu hem de yarma alanına sahiptirler. Bazı enkesitler ise ya tam yarma ya da tam dolgudan oluşmaktadır.

Yol projelerinde toprak işleri sonunda yol platformu ile doğal zemin arasında oluşturulan eğri parçalara yarma veya dolgu şevleri denilmektedir.

Yatay geometri ve buna ilişkin düşey geometrinin tasarımı esnasında yatay geometrinin belli kısımlarının değiştirilmesi gerekmektedir. Enkesitler arasında kalan yatay geometrinin değişikliklerine ripaj, enkesitlerin dışına çıkan değişikliklere ise varyant denir. (Baykal, 2009).

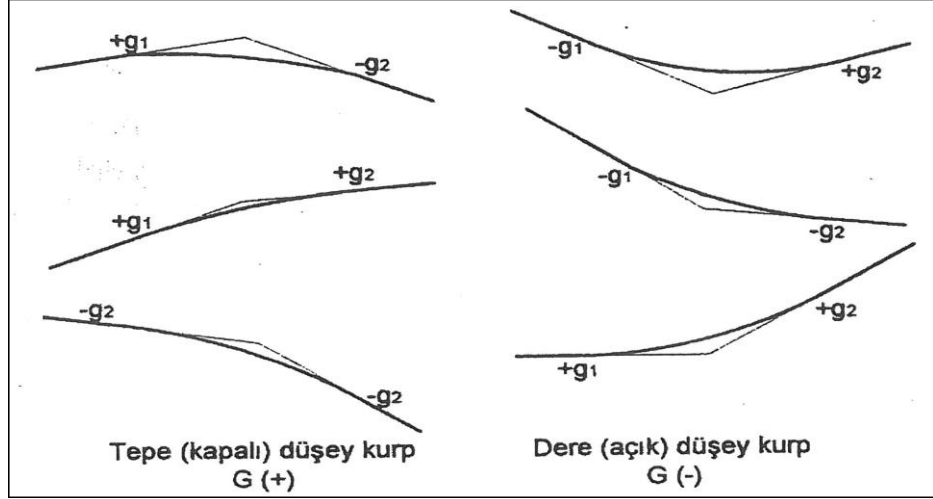
2.4.8. Boykesitin Hazırlanması

Yol projelerinin boykesit hazırlanmasındaki ilk iş; aliymenlar yani doğru parçalar ve kurbalardan oluşan geçki ekseninin plan üzerindeki izdüşümünü belli bir ölçek ile bir doğru boyunca açılarak tasarlanmasıdır. Buna bağı olarak, bu aliymen üzerinde geçkinin aplike edilmesinde, kazık çakılan noktalarda kilometreler belirtilir ve bu sırada alınan her noktaya ait siyah kotlar, belirli bir ölçekle, bir kıyas hattı üzerinde düşey bir düzlem olarak işaretlenir. İşaretlenmiş olan bu noktalar birleştirilir ve bir kırık çizgi oluşturur. Bu geçki, ekseni boyunca arazinin doğal halini göstermekte ve siyah çizgi adı verilmektedir. Siyah çizgi üzerindeki herhangi bir noktanın kotu o noktanın siyah kotu demektir. Bu durumda, boykesit üzerindeki siyah kot, doğal arazinin kotu olmaktadır (Nicholson, 1988; Schofield, 2001; Wright and Paquette, 1985).

Geçkinin siyah çizgisi geçirildikten sonra bazı esaslar çerçevesinde kırmızı çizgi tasarlanır. Kırmızı çizgi, yolun toprak işi sonunda ulaştığı tesviye yüzeyinin durumunu boykesit üzerinde gösterilen hat olarak tanımlanır ve bu hat üzerindeki tüm noktaların yükseklikleri o noktanın kırmızı kotunu göstermektedir. Başka bir deyişle, yolun tesviye yüzeyi hazırlandığında kırmızı çizgi üzerinde kalan kısımlara yarma ve altında kalan kısımlar ise dolgu yapılacaktır. Yolun tesviye yüzeyi hazırlandıktan sonra üzerine inşa edilen üstyapı tamamlanır (Uren and Price, 2006).

Doğru parçalar ve bu doğruları birbirine bağlayan eğri kısımlar sürekli bir çizgi oluşturur bu çizgiye kırmızı çizgi adı verilir. Düşey kurp adı verilen bu eğri kısımlar daire veya parabolik yaylarıdır. Doğru parçalara ait kırmızı kotlar bu kısımdaki boyuna eğim ve yolun başlangıcına olan uzaklık ile kolaylıkla hesaplanabilir. Buna karşılık olarak düşey kurp bölgelerine denk gelen noktalara ait kırmızı kotlar ise matematiksel hesaplamalar ile bulunur.

Boykesitte kırmızı çizginin iniş yapan kesimlerine iniş, çıkışta olan kesimlerine ise rampa adı verilir. Bir inişi daha dik eğimli bir iniş veya bir rampayı daha az eğimli bir rampa izliorsa aradaki düşey kurbaya kapalı düşey kurp veya tepe düşey kurp adı verilmektedir. Buna karşın, rampayı daha dik eğimli bir rampa, bir inişi bir rampa veya bir inişi daha az eğimli bir iniş izliorsa aralarındaki düşey kurbaya açık düşey kurp veya dere düşey kurp adı verilmektedir. Şekillerin altında gösterilen G değeri, g_1 ve g_2 eğimlerinin cebrik farkıdır (Şekil 2.5) (AASHTO, 1994).



Şekil 2.5. Tepe ve dere düşey karp tipleri (Yayla, 2008)

Karayolu uygulamalarında bazı bölgelerin boyuna eğimi sıfır olduğundan bu kısımlara palye adı verilir. Palye ancak, yolun enine eğimi yeterli ve arazi durumu elverişli ise yapılabilir. Ayrıca, yolun kenarları yani banketin dış kenarları açık olduğunda kısa kesimler için yapılabilir. Aksi takdirde yola boyuna eğim verilerek yüzey suların drenajı gerçekleşmiş olacaktır.

Boykesitin tasarımı sırasında, genelde düşey ölçek için 1/200 veya 1/100 ölçekleri ve yatay ölçek için 1/2000 veya 1/1000 kabul edilmektedir. Bu ölçeklere göre hazırlanan boykesit üzerinde, her noktanın kilometreleri ve her kesite ait kırmızı ve siyah kotlar, kesitler arası mesafe, proje başlangıcına olan uzaklıklar, yatay kurbaların tüm verileri ve kurba elemanları, boyuna eğimlerin değiştiği noktalar ve bu noktalar arasındaki eğimler, ayrıca, her kesitteki yarma ve dolgu miktarları yani kırmızı kotlar ile siyah kotların farkları gösterilmektedir. Ölçü ve cinsleri ile belirtilecek istinat ve iksa duvarları ve menfezler gibi sanat yapılar da boykesit üzerinde gözükmelidir (Yayla, 2008).

2.4.9. Boyuna Eğimin Sınır Değerleri

Kırmızı çizginin tasarımı yapılırken her şeyden önce eğim sınır değerlerine dikkat etmeliyiz. Bu bağlamda eğim seçiminde en önemli faktörler; arazinin topografik durumu ve yolun sınıfı ve en önemlisi faydalanan olan trafikteki hâkim taşıt cinsidir. Yüksek standart ile tasarlanan yollarda proje hızının büyük olması gerekmektedir. Bu yüzden, öngörülen yüksek standartlı işletme hızına ulaşmak için boyuna eğim mümkün olduğunca az tutulabilir, tabi ki bu durum sadece arazi durumunda elverişli ise mümkün olacaktır. Aynı şekilde, yüksek standartlı ve ağır taşıt

trafiğinin çok fazla olduğu takdirde, bu yollarda küçük eğimler tercih edilerek taşıtların işletme maliyeti azaltılmış olacaktır. Buna karşın, boyuna eğimi yüksek tutarak dağlık bölgelerde yapım zorlukları daha aza indirilmiş olacaktır. Bu gibi yüksek eğimli kesimlerde eğimi düşürmek mümkün olamıyorsa rampa uzunluğunu en az tutarak hiç olmazsa işletme maliyetinin artmasını önlemeye çalışılacaktır (Yayla, 2008).

2.4.10. Kırmızı Çizginin Geçirilmesi

Geçki eksenindeki siyah çizgi belirlendikten sonra boyuna eğim için KGM tarafından kabul görülen eğim sınır değerlerinden minimum ve maksimum değerler arasında kalacak bir şekilde kırmızı çizgi belirlenir. Bu çizgi önce açık poligon olarak tasarlanır daha sonra dere ve tepe noktalarına düşey kurbalar yerleştirilir. Ancak, bu düşey kurbalar, zorunlu minimum görüş uzunluklarına göre tasarlanmalıdır (Yayla, 2008).

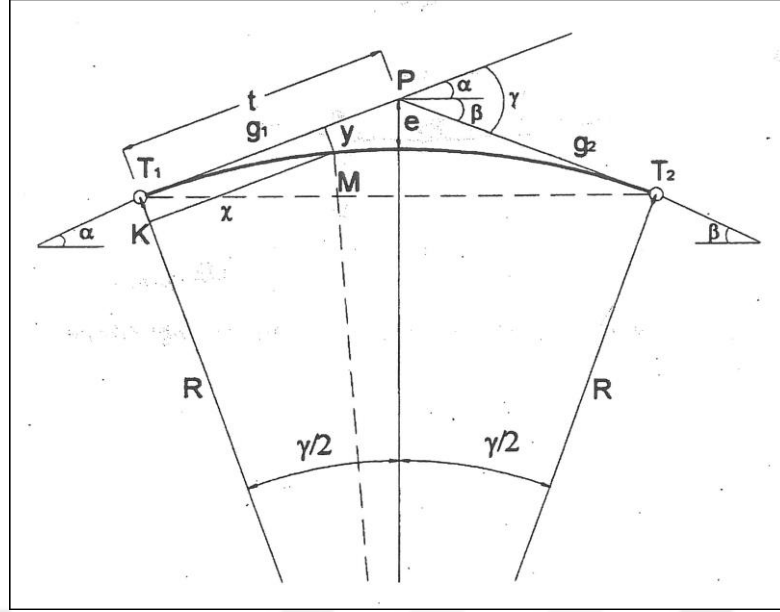
2.4.11. Düşey Kurbalar

Kırmızı çizginin eğimi değiştirildiği takdirde görüş uzunluğu kısaldığından güvenlik azalacaktır. Bu durumlarda, özellikle eğer hız fazla ise, ani eğim değişmesinin düşey ivme ve sarsıntıya yol açtığı ve yolculuk konforunu de düşürmeye neden olacaktır. Bu nedenle belirtilen yerlerde farklı eğimli olan kırmızı çizginin iki kolu arasına uygun uzunlukta düşey kurbalar yerleştirilerek yukarıda adı geçen sakıncaların giderilmesine çalışılır.

Genel olarak birbirini takip eden iki kırmızı çizgi kolları arasındaki eğimlerin cebrik farkının %0.5'den büyük olması takdirde bu iki kolun arasına düşey kurba yerleştirilir. Düşük standartlı yollarda bu fark %1 olarak kabul edilmektedir (Yayla, 2008).

2.4.12. Dairesel Düşey Kurbalar

Düşey kurbalar genel olarak dairesel geçirilmektedir. Şekil 2.6'da gösterildiği gibi eğer birbirini takip eden iki kırmızı çizgi kolunun eğimlerini g_1 ve g_2 olarak kabul edersek, bunların arasına yerleştirilen R yarıçapındaki daire yayı olan düşey kurba aşağıda gösterilmektedir.



Şekil 2.6. Dairesel düşey kurba (Yayla, 2008)

Dairesel kurbaların hesaplamalarında R yarıçapı için arazinin durumu ve yolun sınıfına göre minimum değerler aşağıdaki Tablo 2.1’de verilmektedir (Yayla, 2008).

Tablo 2.1. Dairesel düşey kurba yarıçapları (Yayla, 2008)

Düşey kurba tipi ve arazi durumu	Yol sınıfına göre minimum kurba yarıçapları			
	I	II	III	IV
a- Tepe düşey kurba				
Düz arazi	25.000	15.000	10.000	5.000
Engelibeli arazi	15.000	10.000	5.000	2.500
Dağlık arazi	5.000	2.500	1.500	1.000
b- Dere düşey kurba				
Düz arazi	8.000	5.000	3.000	2.000
Engelibeli arazi	5.000	3.000	2.000	1.500
Dağlık arazi	2.000	1.500	1.200	1.000

2.5. Optimizasyon Yöntemleri

Mühendislik tasarımından iş planlamasına kadar optimizasyon yöntemlerinin her yerde kullandığı yadsınamaz bir gerçektir. Hemen hemen tüm bu çalışmalarda, belirli hedeflere ulaşmak için örneğin kâr, kalite ve en önemlisi zaman gibi bir şeyi optimize etmekte kullanılmaktadır. Günümüzde dünya uygulamalarında daima sınırlı olan kaynaklar, zaman ve paradır. Buna bağlı olarak çeşitli kısıtlamalar altında çözümler bularak değerli kaynakları en iyi şekilde kullanmak gerekmektedir. Matematiksel optimizasyon veya programlama, mevcut kısıtlamalar etkisinde,

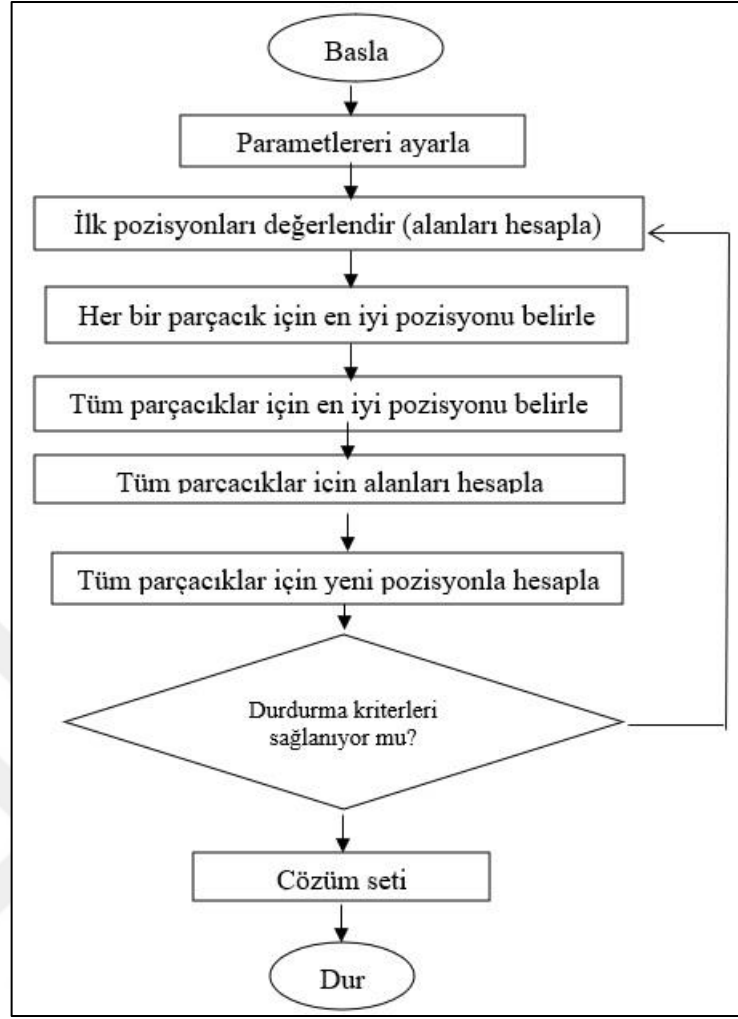
istenilen faktörleri maksimum seviyeye çıkartarak ve istenilmeyen faktörleri ise minimuma indirerek en ekonomik maliyetli veya en yüksek performansa sahip bir seçenek bulmaktır. Bir optimizasyon tasarımı, tasarım amacı genel olarak üretim verimliliğini en üst düzeye veya üretim kârını en aza indirmektir. Günümüzde bilgisayar yardımıyla simülasyon çözümü gerektiğinde farklı faydalı arama algoritmaları ile bu tür optimizasyon problemleri yadsınamaz bir seçenek haline gelmektedir (Yang and Press, 2010).

Optimizasyon yöntemleri çok farklı algoritmalara sahip olmaktadır. Bunlardan en güvenilir ve çözümü garanti edebilecek olanlar meta-sezgisel yöntemleridir. Bu yöntemlerde kendi içlerinde tek amaçlı ve çok amaçlı yöntemler olarak ikiye ayrılmaktadır. Bu yöntemlerin en önemli farkı ise çözüme çalışmak istedikleri kriterlerin amaç fonksiyon sayısıdır. Aşağıda, bu çalışmada kullanılan tek ve çok amaçlı optimizasyon yöntemleri açıklanmaktadır.

2.5.1. Tek Amaçlı Parçacık Sürü Optimizasyonu (PSO)

Kennedy (bir sosyal psikolog) ve Eberhart'ın (bir elektrik mühendisi) parçacık sürüleri için ilk değişik hesaplama yöntemi ile zekâ üretiminde tüm bireysel yeteneklerin yanında daha basit sosyal etkileşimlerin analoglarını kullanılmasını önermişlerdir (Kennedy and Eberhart, 1995; Reynolds, 1987; Rao, 2019).

Parçacık ifadesi, örneğin, bir sürüdeki kuşu veya bir kolonideki arıyı ifade etmektedir. Sürüdeki her parçacık veya birey, grup zekasını veya kendi zekasını kullanarak dağıtılan bir biçimde davranmaktadır. Bu sebeple eğer bir parçacık gıdaya giden yolu iyi bir şekilde bulursa, hızlıca sürülerin geri kalanı ve hatta sürüdeki en uzak olanlar da o iyi yolu seçebilir. PSO yöntemi genel olarak problemler üzerinde daha yüksek ve kaliteli performans göstermektedir. Her parçacığın iki tane özelliği vardır: Birisi konumu diğeri ise hızıdır. Her parçacık tasarım bölgesinde gezerek bulunduğu en iyi konumu hatırlar. Parçacıklar birbirlerine bilgi aktarır veya iyi konumlar birbirleriyle haberleşer ve böylece iyi konumlarda alınan bilgilere dayanarak kendi konumlarını ve hızlarını ayarlarlar (Reynolds, 1987). Bu algoritma ile ilgili akış şeması Şekil 2.7'de gösterilmektedir.



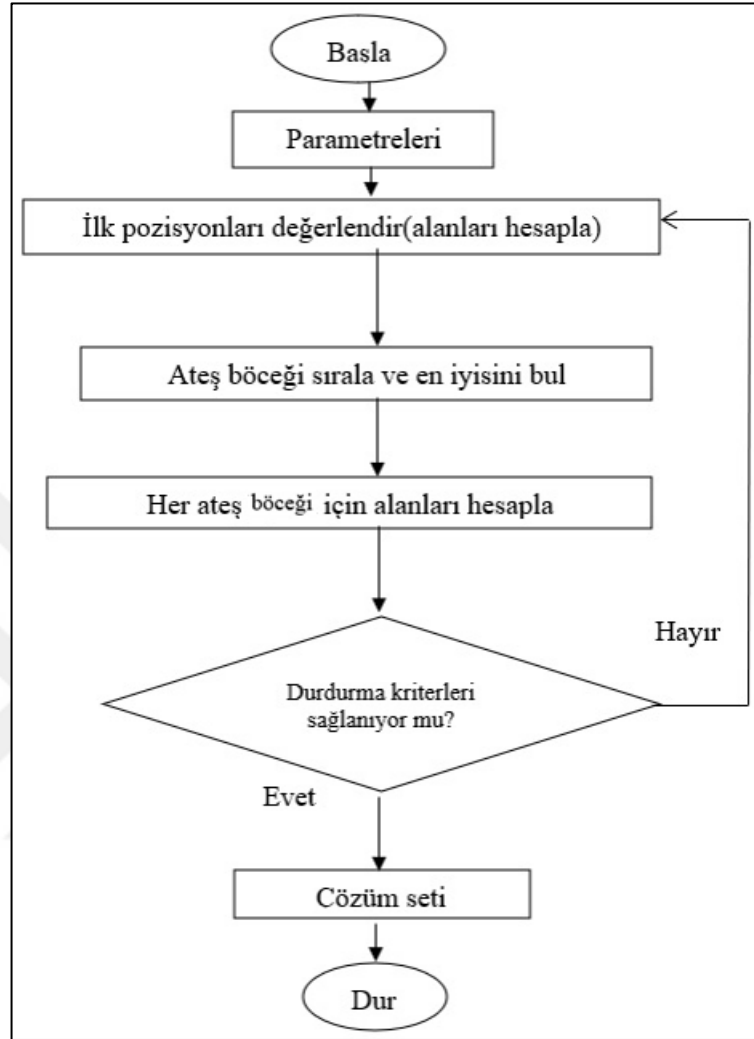
Şekil 2.7. Parçacık sürü optimizasyonun akış diyagramı (Hassan et al., 2005)

Daha sonra metasezgisel temelli ateşböceği algoritması üzerinde değişiklikler yapılarak algoritmanın parametreleri ayarlanır hale gelmiştir ve sinir ağları gibi karmaşık problemleri çözmüştür (Yelghi ve Köse, 2018).

2.5.2. Tek Amaçlı Ateşböceği Algoritması (Firefly Algorithm- FA)

Ateşböceği algoritması aslında ateşböceklerin birbirlerine olan parlaklık karakteristiklerine göre göserdikleri performansı modelleyen bir algoritmadır. Algoritma genel olarak üç tane temelden ibarettir, birincisi, daha az parlak olan daha çok parlak olana doğru hareket eder yani daha az parlak olan daha çok parlak olana doğru hareket eder. İki ateşböceğinin arasındaki mesafe her ne kadar artarsa çekicilik o kadar azalır. Eğer ateşböceği, kendisinden daha parlak bir ateşböceği bulamazsa rastgele hareket eder. İkincisi ise tüm ateşböcekleri tek cinstir. Bu nedenle cinsiyette bakılmaksızın birbirlerine doğru hareket ederler. Üçüncüsü de Bir

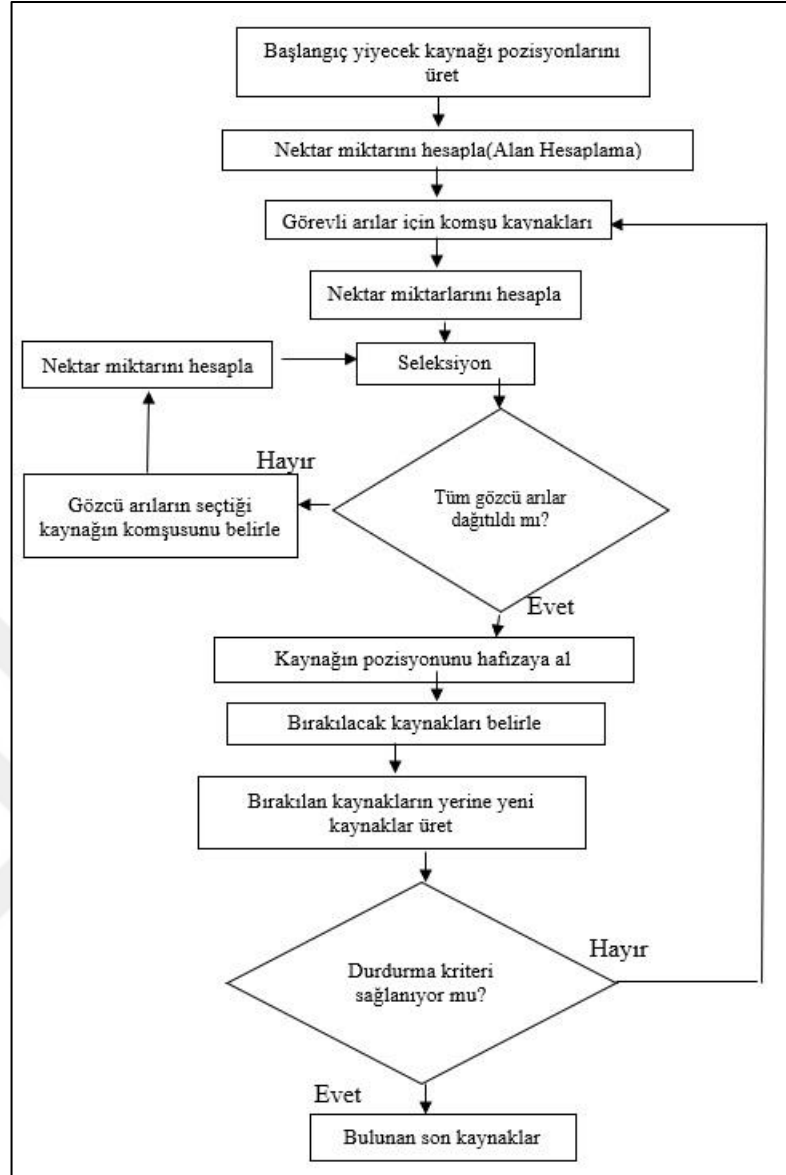
ateşböceğinin çekiciliği amaç fonksiyon değeriyle belirlenir (Yang, 2010). Bu algoritma ile ilgili akış şeması Şekil 2.8’de gösterilmektedir.



Şekil2.8. FA’nın akış diyagramı (Yang, 2010).

2.5.3. Tek Amaçlı Yapay Arı Koloni (*Artificial Bee Colony- ABC*)

Bal arı sürülerinin kolektif zekâsı ortaya çıkmasına yol açan yem seçiminin minimal modeli üç temel bileşenden oluşur. Bu bağlamda bir koloni içinde üç grup arı mevcuttur: Kâşif arılarıdır, görevli arılar ve gözcü arılarıdır (Karaboga et al., 2014). Önerilen modelde koloninin yarısı gözcü arıya ait ve diğer yarısı ise görevli arıya aittir. Başka bir deyişle görevli arıların sayısı nektar miktarına eşittir (Karaboga et al., 2007). *ABC* optimizasyon algoritmasının genel yapısı Şekil 2.9’da göstermektedir.



Şekil2.9. ABC'nın akış diyagramı

2.5.4. Çok Amaçlı Optimizasyon Yöntemi NSGAII

NSGA-II (Non-dominated Sorting Genetic Algorithm) çok amaçlı bir yöntem olduğundan, doğrudan çözen optimizasyon algoritması olarak popülasyon elemanlarını bu yöntem ile sıralayabilmektedir. Bu yöntemde kıyaslama işlemi için bir kriter vardır yani x ve y arasında seçim yapılırsa galip olan seçilir ve mağlup olan elenir (Coello et al., 2007; Deb et al., 2002; Engelbrecht, 2007).

Çok amaçlı optimizasyon yöntemleri genel olarak iki veya üzeri faktör üzerinde optimizasyon işlemi gerçekleştirmektedir. İyi bir algoritmanın bulduğu cevaplarda önce kalitesi ve sonra düzeni mercek altına alınmaktadır. NSGA'nın ilk versiyonunda

yani NSGA-I algoritmasında düzen kriteri kullanılmamıştır. Daha etkin bir yöntem için NSGA-II uygun ve güçlü bir algoritma olarak tasarlanmıştır (Deb, 1999).

NSGA-II en popüler çok amaçlı optimizasyon algoritmalarından biridir. Bu algoritmaya dayalı yazılmış olan makaleler diğerlerinden daha fazla ve kullanıldığı fikir çok yaygındır. Tek amaçlı meta-sezgisel algoritmalar, NSGA-II fikrinden esinlenerek çok amaçlı algoritmaya dönüşebilmektedirler. Başka bir deyişle, PSO (particle swarm optimization), arı kolonisi Algoritması (bee colony), karınca kolonisi algoritması (ant colony) gibi algoritmaların da çok amaçlı versiyonları vardır ve NSGA-II fikrini kullanarak çalışmaktadırlar.

Tek amaçlı genetik algoritma ve çok amaçlı genetik algoritmanın tek farkı yeni popülasyonların seçimindedir. Bu algoritmalar önce cevapların kalitesini göz önünde bulundurarak sıralama işlemini yapar ve daha sonra eğer kalite faktörü belirleyici olmuyorsa ikinci kriter olarak düzeni esas alacaktır. Başka bir deyişle, örneğin: Futbol maçlarında takım eleme konusunda bazen sadece puan belirleyici olmayabilir bu durumda gol sayısı ikinci kriter olarak kullanılmaktadır. Bu yüzden bu yöntemde birleştirme, sıralama ve kesme senaryoları göz önünde bulundurulmaktadır (Coello and Coello, 2001; Coello et al., 2007)

2.5.5. Çok Amaçlı Optimizasyon Yöntemi MOPSO

Coello vd. (2004), yaptıkları çalışmalarda MOPSO (*Multiple Objective Particle Swarm Optimization*) algoritmasını ileri sürmüşlerdir. Tek amaçlı optimizasyonu problemlerini PSO yöntemi ile ve çok amaçlı optimizasyon problemlerini ise PSO yöntemini geliştirerek MOPSO yöntemi ile çözebilmekteyiz. PSO algoritmasına bazı değişiklikler uygulayarak MOPSO'ya dönüştürmek mümkündür. Aynı şekilde, GA (*Genetic Algorithm*) genetik algoritmasını NSGA-II algoritmasına dönüştürmek için birtakım yazılım değişikliği yapılmalıdır. Bu değişimin en önemli nedeni problemin çok amaçlı olmasıdır. MOPSO algoritmasını daha iyi çözebilmek için önce PSO algoritmasını gözden geçirmeliyiz ve daha sonra MO teriminin üzerinde durulacaktır.

Parçacık Sürü Optimizasyon algoritmasının temelinde sürü zekâsı vardır. Sürü zekasında etmenler tek başına etkili olamazlar. Bu nedenle sürü içerisinde yer alarak ve iş birliği yaparak sonuca ulaşırlar. Sürü şeklinde ulaştıkları sonuca tek başına ulaşmaları mümkün değildir. Arama bölgesinde yeni cevapların bulunması için birtakım kurallar tanımlanmalıdır. Bu bağlamda, en önemli kavram hareket

kavramıdır. İlk hareket bir nevi atalet sayılmaktadır. İkinci hareket ise nostaljik olarak sayılır ve tek başına bir işe yaramaz. Algoritmanın ana çekirdeği üçüncü aşaması sayılmaktadır. *PSO* ve *MOPSO* arasındaki fark en iyi parçacığın seçilmesindedir yani çok amaçlı optimizasyon algoritma konusunda en iyi seçeneğin tanımı üzerine odaklanmıştır. Çok amaçlı optimize algoritmalarına göre mağlup olmayan seçeneğin birinci derecede kalitesi sonra da düzeni önemlidir (De Oca et al., 2009).

2.5.6. Çok Amaçlı Optimizasyon Yöntemi *SPEA2*

Kuvvet Pareto Evrimsel Algoritması (*SPEA*), eski bilinen yöntemlerle yenilerini birleştirmektedir. (Zitzler, 1999).

SPEA'daki bazı potansiyel zayıflıkları iyileştirilerek *SPEA2* geliştirilmiştir. Bu durumda *SPEA* rasgele araştırma algoritması gibi çalışmaktadır. *SPEA*'dan farklı olarak, *SPEA2* yoğunluk bilgisini kullanan ince-taneli uygunluk atama stratejisini kullanır. Ayrıca, arşiv boyutu da sabit tutulmaktadır. Başka bir deyişle, bastırılmamış bireylerin sayısı eğer önceden tanımlanmış arşiv boyutundan küçük ise, arşiv tamamen bu bireylerle doldurulur. *SPEA*'da genel olarak arşiv boyutu değişebilir. Ayrıca, bastırılmamış yüzey eğer arşiv boyutunu aşıyorsa kullanılan kümeleme tekniği, benzer özellikleri olan ama sınır noktalarını kaybetmeyen alternatif bir küçültme yöntemiyle değiştirilir. Son olarak, *SPEA2*'nin seçim işleminde sadece arşiv üyeleri kullanılmaktadır (Zitzler, 1999; Zitzler et al., 2000).

2.6. Kaynak Özetleri

Easa (1988), yaptığı çalışmada yol sınıfını tüm geometrik özelliklerini göz önünde bulundurarak ve hafriyat miktarını en aza indirerek maliyet hesabını yapmıştır. Bu işlemi gerçekleştirmek adına bir model hazırlanmıştır. Bu modelde hafriyat maliyeti en aza indirilip ve geometrik özellikleri karşılayan yol sınıfının seçimi yapılmaktadır. Bu bağlamda, düşey ve yatay kurp ilişkileri ve tüm eğimler kontrol edilmektedir. Model, teknik olarak mümkün olan tüm yol sınıflarını sıralar ve depolama ve ödünç alma maliyetlerini de düşünerek doğrusal programlama problemini çözmeye çalışmaktadır.

Goh vd. (1988), 1988 yılında iki boyutlu otoyol konum problemlerini çözmek için formüle edilmiş bir dinamik programlama modeli ve durum parametrelere modelini tanımlamışlar. Bu bağlamda, bu iki modeli kullanarak elde edilen çözümleri karşılaştırmak için sayısal bir örnek sunmuşlardır. Bu çalışmada, optimum yatay ve

düşey kurpları aynı anda hesaplamak için üç boyutlu bir modelin temeli oluşturulmuştur.

Chew vd. (1989), üç boyutlu bir karayolu güzergâhını modelleyerek çok çeşitli olası kısıtlamalara tabi tutarak optimum siyah çizgi ve kırmızı çizginin bulunmasına yönelik sayısal araştırmalar gerçekleştirilmişlerdir. Bu bağlamda önerilen modelin uygulanabilirliğini göstermek için Singapur'daki gerçek zemin koşullarına dayanan iki sayısal örnek sunmuşlardır.

Fwa (1989), bu çalışmada önceden seçilmiş bir yatay çizgi için optimum bir düşey çizgi profili üretmek üzere geliştirilmiş bir optimizasyon programı sunmaktadır. Programın amacı, singapur'daki yerel otoyol mühendislerine ve planlayıcılarına otoyol geometrik tasarımı ve konum analizi için pratik bir yardım sağlamaktır. Hafriyat, malzeme ve araç işletme maliyetlerini içeren toplam maliyeti en aza indirmek için dinamik bir programlama formülasyonu benimsenmiştir. Programın çok yönlülüğü, otoyol yapımında önemli olan maliyet kalemlerinin çoğunu modellemesine olanak tanır ve farklı geometrik tasarım parametrelerinin proje maliyetleri üzerindeki etkileri hakkında hızlı bilgi sağlamak ve farklı maliyet kategorilerinin önemini belirlemek için kullanılabilir. Program uygulamasını göstermek için gerçek hayattan sayısal bir örnek sunulmuştur.

Moreb (1996), karayolu uygulamasındaki hafriyat maliyetini belirlemek için tasarımcı tarafından seçilen yol sınıfına bağlı olduğu üzerinde çalışmalar yapmıştır. Tasarımcının seçebileceği neredeyse sonsuz sayıda uygulanabilir alternatif vardır ve bunların hepsi de yolun geometrik özelliklerini karşılamaktadır. Tabii ki bu seçeneklerin hepsine hafriyat hesaplaması da yapılmalıdır. Bu çalışmada, karayolu eğim seçimi ve hafriyat taşımalarını tek bir doğrusal programlama probleminde birleştirerek incelenmiştir.

Fwa vd. (2002), yaptıkları çalışmada genetik algoritmaların yapay zekâ tekniği sayesinde karayolu tasarımındaki eğim, sabit yükseklik noktaları ve yatay-dikey kurpların karmaşık matematiksel çözümleri yaparak, güzergâh optimizasyon analizlerini gerçekleştirmişler. Bu üç kısıtlama göz önünde bulundurularak ilgili inşaat maliyetleri incelenmiştir.

Goktepe ve Lav (2003), çalışmalarında inşaat maliyetini önemli ölçüde azaltabilme üzerinde çalışmalar yapmışlardır. Bu bağlamda, hafriyat işlerini en aza

indirmenin yanı sıra yarma-dolgu dengelemesini de dikkate almışlardır. Uygulamadaki hafriyat miktarının azaltılması için kırmızı çizginin zemin çizgisine mümkün olduğunca yakın geçirilmelidir. Ancak, eksen hattı dengeleme açısından yanıltıcı olabildiğinden, bu makalede eksen hattı yükseklikleri yerine zemin yüksekliğindeki enine değişikliklerin ortalamasını temsil eden varsayımsal bir ağırlıklı zemin kotunu dikkate alan bir yöntem önerilmektedir. Sonuç olarak, kırmızı çizginin ağırlıklı zemin çizgisine uygun hale getirilmesi ve dengeli bir yarma-dolgu tasarımı ve daha az hafriyat miktarı ile çok daha uygun maliyetli bir sonuç sağlamışlardır.

Jha ve Schonfeld (2004), yaptıkları çalışmada maliyeti göz önünde bulundurarak bir coğrafi bilgi sistemi ile genetik algoritmaları bütünleştiren, çeşitli maliyetlerin güzergâh seçimi üzerindeki etkilerini inceleyen ve yol iyileştirme projelerindeki sınırları gerçekçi bir şekilde yansıtan kısıtlı alanlarda optimizasyonu araştıran bir güzergahın modelini sunmuşlardır. Sonuçlar, karayolu tasarımı sırasındaki güzergâh seçimindeki sıklıkla ihmal edilen seyahat süresi maliyetinin güzergâh optimizasyonunu önemli ölçüde etkilediğini göstermektedir.

Goktepe ve Lav (2004), çalışmalarında enkesitin eksen hattı üzerindeki yükseklik kotunun yanıltıcı olabileceğinden bir enkesit üzerinde zemin kotu olarak, ağırlıklı zemin hattı yöntemini kullanarak olabildiğince daha yakın ayarlamaktadır. Ayrıca, bu çalışmada bir enkesitteki orijinal hacim ile bir dolgudaki sıkıştırma sonrası hacim arasındaki fark nedeniyle ve buna bağlı olarak, tüm hafriyat malzemesinin dolgu malzemesi olarak kullanılamayacağına göre ağırlıklı zemin kotlarının hesaplanmasında modifiye edilmiş kazı ve dolgu hacimleri kullanılmıştır. Sonuç olarak bu makalenin amacı, büzülme ve şişme faktörlerini göz önünde bulundurarak yarma-dolgu hacmini dengelemek ve hafriyat miktarını en aza indirmek ve böylece çok daha uygun maliyetli proje tasarımını sağlamaktır.

Goktepe vd. (2005), yaptıkları çalışmada dinamik programlama yaklaşımını kullanarak bir karayolu kırmızı çizgisinin optimizasyonu için bir metodoloji tanımlayarak çeşitli tasarım kısıtlamalarını göz önünde bulundurarak minimum maliyete neden olan optimum kırmızı çizgiyi bulmuşlardır. Bu bağlamda, tanıtılan algoritmanın kırmızı çizgi optimizasyonunda başarıyla kullanılabileceğini göstermişlerdir.

Kim vd. (2005), yaptıkları çalışmada hesaplama verimliliğini ve çözümlerin kalitesini arttırmak için genetik algoritma kullanımını önermişlerdir. Bu bağlamda,

karayolu tasarımının optimize etme aşamasında, coğrafi bilgi sistemi (CBS) veri tabanında hesaplanacak verilerin yükünün önemli bir sorun olacağı gözlenmiştir. Başka bir deyişle, CBS aracılığıyla doğrudan gerçek haritalar üzerinde çalışılmıştır. Ayrıca, hesaplama süresinden tasarruf etmek ve oldukça büyük ve detaylı projelerin daha hassas özellikler içerdiği kavşaklar, köprüler ve tünellerinin incelemesi için bir modelin benimsene bilirliliği artmaktadır. Pek çok optimizasyon sürecinde büyük problemleri daha küçük parçalara ayırmanın hesaplama süresini azaltabileceği ve daha iyi bir çözüm üretebileceği çok iyi bilinmektedir. Bu araştırmada, çalışma alanları uygun bir şekilde alt bölümlere ayrılarak genetik algoritmalar kullanarak mevcut yöntemlere göre daha verimli sonuçlar alınmıştır.

Kim vd. (2007), yaptıkları çalışmalarda köprülerin veya tünellerin dolgu bölgelerine tercih edilip edilmediğini otomatik olarak belirleyen üç boyutlu bir optimizasyon metodu geliştirmişlerdir. Ancak, bu tür optimizasyonlar maliyet karşılaştırmasıyla yapılmaktadır. Bu bağlamda, hafriyat miktarı, araç işletimi ve diğer tüm maliyetlerin ayrıntıları coğrafi bilgi sistemi üzerinden elde edilebilir. Böylece, genetik algoritma yöntemini geliştirip ve yapay bir alan oluşturarak karayolu güzergâh optimizasyonunu gerçekleştirmişlerdir.

Goktepe vd. (2009), karayolu optimizasyonu ile ilgili yaptıkları bu çalışmada hafriyat optimizasyonu, projelerin toplam maliyetinde çok önemli bir rolü olacağından, ağırlıklı zemin hattı metodunu geliştirerek yarma-dolgu hacimlerini dengeleyen ve toplam hafriyat miktarını en aza indiren bir zemin hattına dayanan bir hafriyat optimizasyon tekniği geliştirmişlerdir. Bu çalışmada, belirli bir ağırlıklı toprak çizgisi için çok sayıda kırmızı çizgi alternatifi olacağından hafriyat optimizasyonu açısından karayollarının en iyi kırmızı çizgiyi elde etmek için dinamik programlamayı baz alarak bir yöntem sunmuşlardır.

Kang vd. (2012), yaptıkları çalışmada tasarımcılara kırmızı çizgi tercihinde akıllı bir optimizasyon aracı sunmuşlardır. Karayolu optimizesi için genetik algoritmaları bir coğrafi bilgi sistemi (GIS) ile bütünleştirmişlerdir. Oluşturulan model karmaşık arazi tiplerini birleştirerek, kırmızı çizgiyi optimize ederek bir model sunmaktadır. Model tarafından optimize edilen tasarımlar gerçeğe oldukça yakındır. Ancak, gereken süreyi büyük ölçüde azaltabilmenin yanında daha düşük maliyetli çözümler de üretebilmektedir.

Bosurgi vd. (2013), karayolu optimizasyonu ile ilgili bazı çalışmalar yapmışlardır. Karayolu optimizasyonu için algoritmaların geliştirilmesi hızla artmaktadır. Bu konu çok karmaşık olduğundan ve çok faktörden oluştuğu için mühendisler farklı ve ilgili hedeflere ulaşmak zorundadırlar. İnşaat ve hafriyat maliyetlerin en aza indirilmesi, çevresel kısıtlamalara uyum sağlamak ve kullanıcılara güvenlik ve konforun en üst düzeye çıkarılması çok ince bir şekilde aynı anda düşünülmelidir. Bu karmaşık hesaplamaları sadece modern yapay zekâ teknikleri basitleştirip ve hızlandırabilir. Bu çalışmada karayolu optimizasyonu için parçacık sürü optimizasyon yöntemi (PSO) kullanılarak bir arama algoritması önerilmiştir. Bu algoritma, geometrik ve çevresel kısıtlamalar ile ilgili farklı türde inşaat maliyetlerinden oluşan belirli bir maliyet fonksiyonun minimizasyonu ile ilgilidir ve böylece yanlış tasarımları önleyebilmektedir. Bu algoritmayı kullanmak için Matlab yazılım programı kullanılmıştır.

Al-Sobky (2014), kırmızı çizginin yerini hafriyat miktarına göre tespit edilmesi üzerinde çalışmalar yapmıştır. Bu bağlamda, kırmızı çizgi tasarımı yol maliyetini, güvenliğini ve trafik performansını etkilemektedir. Kırmızı çizgi optimizasyonu tasarım kriterlerini karşılamadan yanı sıra toplam hafriyat miktarının en aza indirilmesin de sağlar. Genel olarak, optimum kırmızı çizginin belirlenmesi, deneme yanılma sürecini başlatmaktadır. Son yıllarda, doğrusal programlama ve genetik algoritmalar kullanılarak kırmızı çizgi optimizasyonu yapılmıştır. Bu bağlamda, kırmızı çizgi parçaları eşit mesafelerde varsayılmıştır. Ancak, bu eşit mesafeler düşük hafriyata yol açmayacaktır. Bu bildiride, hafriyat dengelemesi için yani yarma-dolgu eşitlemesi için bir sayısal yaklaşımda bulunmuştur. Ancak, önerilen yaklaşım sadece kırmızı çizginin tek bir parçası üzerinde sağlanmıştır.

Casal vd. (2017), yaptıkları çalışmada yatay güzergahın optimizasyonu için geçiş eğrileri ile uygun şekilde bağlantılı teğet uzunlukları ve dairesel kurplardan oluşan genel bir formülasyon sunmuşlardır.

Babapour vd. (2018), yaptıkları çalışmada meta-sezgisel algoritmalar kullanarak kırmızı çizginin optimizasyonuna odaklanmışlardır. GA ve PSO yöntemleri iki akıllı optimizasyon aracı olarak, orman yolu boykesit tasarımı ve maliyet değerlendirmesi için uç noktaları birbirine bağlayarak bir optimize orman yolu bulmak için kullanılmıştır. Uygulanan yöntemler arasından GA yöntemi daha düşük bir hesaplama

maliyeti ile daha iyi çözümlerde optimum konumu kaydedebilmesi nedeniyle en uygun yöntem olarak sunulmuştur.

Ghanizadeh ve Heidarabadizadeh (2018), yaptıkları çalışmada ilk olarak kırmızı çizginin optimizasyonunu formüle etmişlerdir. Bu amaçla her bir some noktası seçiminde yükseklik ve uzunluk değişken olarak kabul edilmiştir. Bu optimizasyon problemini çözmek için genetik algoritma (GA) ve parçacık sürü optimizasyonu (PSO) kullanarak karşılaştırılmıştır. Her bir karayolu optimizasyonu için önce bir alternatif güzergâh tasarımı yapılır sonra farklı optimizasyon algoritmaları ile en uygun düşey güzergâh belirlenir ve daha sonra maliyet hesabı yapılarak karşılaştırılır.

Ghanizadeh vd. (2020), yaptıkları çalışmada kırmızı çizgi optimizasyonu için çeşitli fonksiyonları karşılaştırmışlardır. Bu bağlamda problemi formüle ettikten sonra amaç fonksiyonu için, kırmızı çizgi ile mevcut zemin arasındaki varyansın mutlak değerinin toplamı, yarma ve dolgular için farklı ağırlıklara dayalı olarak kırmızı çizgi ile mevcut zemin arasındaki varyansın mutlak değerinin toplamı, yarma ve dolgu hacimlerinin toplamı ve ilk üç durum için hafriyat maliyeti hesaplanarak en doğru olanların karşılaştırması yapılmıştır. Ortaya çıkan problemi optimize etmek için ise genetik algoritma (GA) ve grup arama optimizasyonu (GSO) uygulanmış ve bu iki optimizasyon algoritmasının performansları karşılaştırılmıştır. Bu araştırma, kırmızı çizgi tasarımında yaygın olarak kullanılan mevcut zemin ile kırmızı çizgi arasındaki mutlak varyans değerinin toplamının minimizasyonunun, hafriyat maliyeti açısından optimum kırmızı çizgiyi hiçbir şekilde sağlayamayacağını kanıtlamıştır.

Karayolu projelerinin tasarım sırasındaki ilk aşamada genellikle birkaç alternatif yatay güzergâh oluşturulur, seçilmiş olan her bir alternatif yatay güzergâh için o yolun düşey güzergahı oluşturulur. Seçilmiş olan bir yatay güzergâh için tüm tasarım kurallarına uygun sınırsız sayıda düşey güzergâh olabileceğinden, en ekonomik tasarımı elde etmek için tasarımcı tarafından bir optimizasyon metodu oluşturularak düşey güzergahı optimize etmeye odaklanmıştır. Tasarımcı tarafından oluşturulmuş bu yöntem ile kırmızı çizgi tasarımını optimize etmeye odaklanmıştır. Problemin çok değişkenli ve karmaşık matematiksel işlemlerden oluştuğu için doğrudan bir arama tekniği geliştirilmiştir. Optimizasyonun verimli ve hızlı bir şekilde uygulanabilmesi için AutoCAD Civil 3D yazılımı kullanarak kırmızı çizgiyi optimize etmek için Visual Basic programlama dilinde bir modül tasarlanmıştır. AutoCAD Civil 3D yazılımına entegre edilen program modülü, gerçek bir proje için de uygulanmıştır ve sonuçlarda

gözüktüğü gibi tasarım özelliklerini deęiřtirmeden kırmızı çizgiyi hafifçe ayarlayarak projenin hafriyat maliyetlerinde inanılmaz bir azalma sağlanmıştır (özkan. 2013, ozkan et al., 2021).

Finans ve yazılım mühendislięi alanında, karmařık problemleri çözmek için optimizasyon algoritmaları kullanılmaktadır. Başka bir deyiřle, bu problemlerin çözümleri ile dięer gerçek problemlerin daha istikrarlı ve hızlı bir şekilde çözömleri mümkün olmaktadır (Yelghi and Tavangari, 2022; Yelghi et al., 2021)



3. MATERYAL VE YÖNTEM

Bir karayolu projesi tasarım ve uygulamadan ibarettir. Karayolu projelerinin uygulaması ne kadar ince detaylardan oluşuyorsa bir o kadar da tasarım aşamasına dikkat edilmelidir. Başka bir deyişle uygulama sırasında nasıl ki yanlış uygulamalar kabul edilmeyecek sonuçlar oluşturuyorsa, proje tasarım aşamasında da gerekli şartnamelere göre uygun tasarlanmayan projelerinde, uygulama maliyetleri artacak ve ekonomik olmamasına neden olacaktır.

Karayolları inşaatı sırasında bilindiği üzere önce projeye ait tasarım elle veya bir bilgisayar programı kullanarak yapılmaktadır. Bu bağlamda tasarlanması istenilen yolun tüm bilgileri yani kırmızı çizgi eğimleri, güzergâhlar ve yatay ve düşey kurpların verileri karayolları şartnamesine uygun olarak seçilmelidir.

Bir karayolu projesinin hazırlanmasında, tasarım aşamasına başlamadan önce o projeye ait arazi durumunun incelenmesi ve bu bağlamda gereken en önemli verilerden biri olan arazinin topografisinin dikkatli bir şekilde çıkartılmasıdır ve buna bağlı olarak tüm yol sınıflarının kriterlerini de göz önünde bulundurarak en iyi güzergahın seçilmesidir.

Sıradaki işlem, arazi verilerinin hatasız bir şekilde toplanmasından sonra bu verilere göre güzergahların dikkatle seçilmesidir. Tabi ki güzergâh seçimi ne kadar dikkatli yapılırsa projenin ekonomik olmasına o kadar yardımcı olacaktır. Ancak, güzergâh seçimi, bir karayolu projesinin ekonomik olup olmaması için önemli bir faktör sayılabilir ama tam olarak bir karayolu projesinin ekonomik olmasında yeterli olmayacaktır.

Bir karayolu projesinin tam olarak ekonomik olması isteniliyorsa, o projeye ait kazı işlemleri en az miktara indirilmelidir ve aynı zamanda yarma-dolgu hacimleri birbirine eşitlenmelidir. Bu durumu özetlemek gerekirse bir karayolu projesini en ekonomik hale getirmek için o projenin yarma-dolgusu üzerinde çalışmalar yapılmalıdır. Bu bağlamda kazı işlemlerini eşitlemek ve aynı zamanda en aza indirmek için boykesit tasarımı üzerinde optimizasyon yapılmalıdır.

Boykesit üzerinde bulunan veriler, sadece siyah çizgi ve kırmızı çizgidir. Ancak, arazinin doğal durumunu gösteren siyah çizgiye müdahale etmek pek mümkün olmayacaktır. Bu durumda, geriye kalan ve müdahale edebilmemizi sağlayacak bir tek seçenek vardır ve o da kırmızı çizgi veya proje çizgisi dediğimiz faktördür. Bu

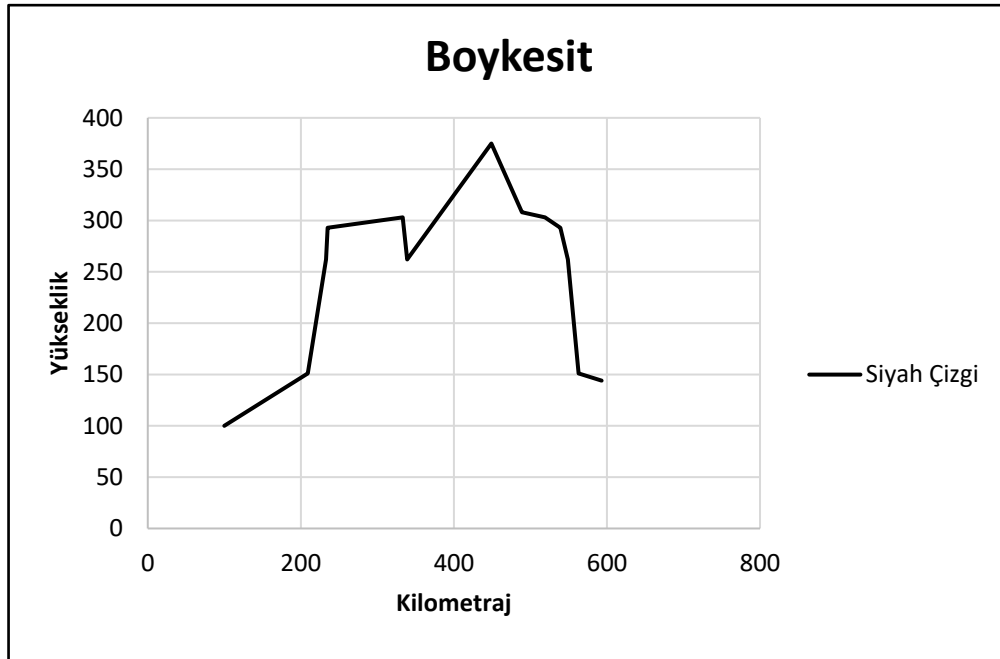
bağlamda, bir karayolu projesinin ekonomik olması için o projeye ait kırmızı çizginin optimizasyonunun yapılması gerekmektedir.

3.1. Boykesit Tasarımı

Karayolları projelerinde boykesit tasarımı, kırmızı çizginin belirlendiği kısım olduğu için çok önemli bir aşamadır. Esas olarak bu aşama, yol inşaatı maliyetini, yol güvenliğini ve performansını etkileyen en önemli aşamadır. Aslında boykesit tasarımı siyah çizgi, siyah kot, some noktaları ve kırmızı çizginin düz kısımları ve düşey kurbalarını içermektedir. Bu bağlamda, yukarıda adı geçen kriterlerin her birinin ayrı ayrı doğru ve ince tasarımı sayesinde, hafriyat miktarını en aza indirerek, maliyeti de düşürmek mümkün olacaktır. Başka bir deyişle, boykesit tasarımında adı geçen kriterlerin her birinin ayrı ayrı optimizasyonu yapılarak direkt maliyetin düşürmesinde önemli bir etki sağlayacaktır. Böylece, boykesit elemanlarının optimum tasarımı, hafriyat miktarını doğrudan etkileyeceği yadsınamaz bir gerçektir.

3.1.1. Siyah Çizgi

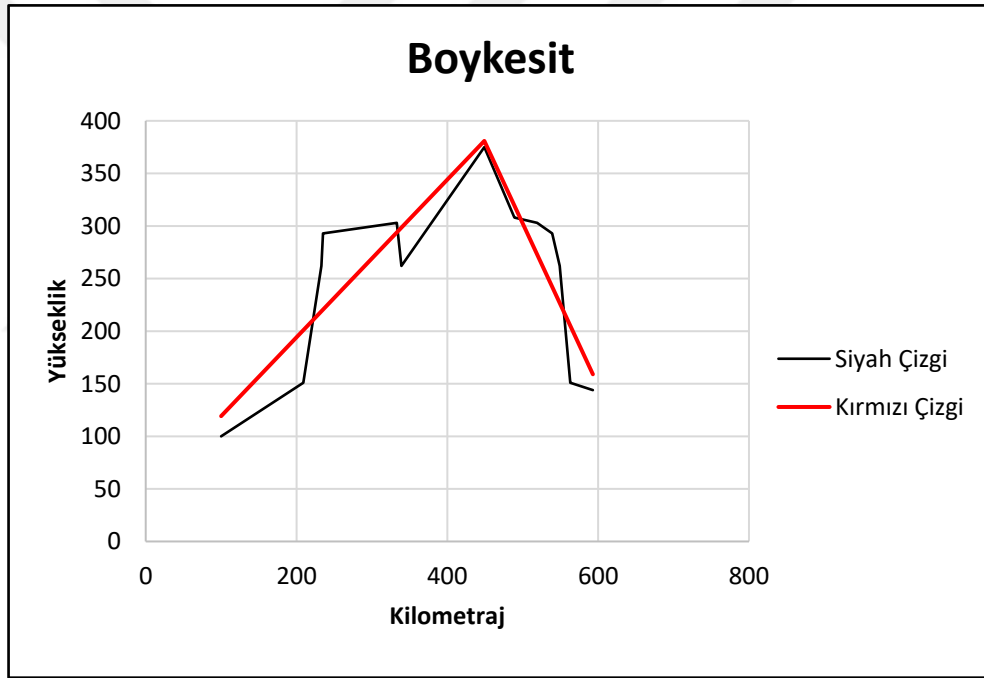
Boykesit tasarımına başlamadan önce plan üzerinde çizilmiş ve güzergâhları belirlenmiş karayolu projesinin, eksen hattı üzerindeki zemin kotları ve enkesit kilometreleri belirlenecektir. Böylece, boykesitin X ve Y ekseninde sırasıyla mesafeler ve yükseklik kotları gösterilerek boykesit tasarımı gerçekleştirilecektir. Bu bağlamda, örnek bir siyah çizgi tasarımı Şekil 3.1'de gösterilmektedir.



Şekil 3.1. Örnek bir siyah çizgi kotları ve kilometrajları

3.1.2. Kırmızı Çizgi

Günümüzde boykesit tasarımı yapılırken some noktaları belirlenerek kırmızı çizgi geçirilmektedir. Ancak, kırmızı çizgi tasarımında en önemli faktör, kırmızı çizginin geçirileceği yerlerdir yani bu çizginin tasarımında dikkat etmemizi gerektiren en önemli konu some noktalarının dikkatli seçilmesidir. Başka bir deyişle some noktaları seçilirken aynı zamanda kırmızı çizgide çizilmiş olacaktır. Bu durumda, siyah çizgi ve kırmızı çizgi aralarında kalan bölgelerde ortaya çıkmış olacaktır. Özet olarak kırmızı çizgi zemin hattını farklı noktalardan keserek iki çizgi arasında oluşan bölgeler yani yarma-dolgu bölgeleri meydana gelecektir. Kırmızı çizgi tasarımı, örnek siyah çizgi üzerinde yani Şekil 3.2'de gösterilmiştir. Bu tasarımda X ekseninde mesafeyi yani kilometrajları ve Y ekseninde yükseklik kotları gösterilmektedir.

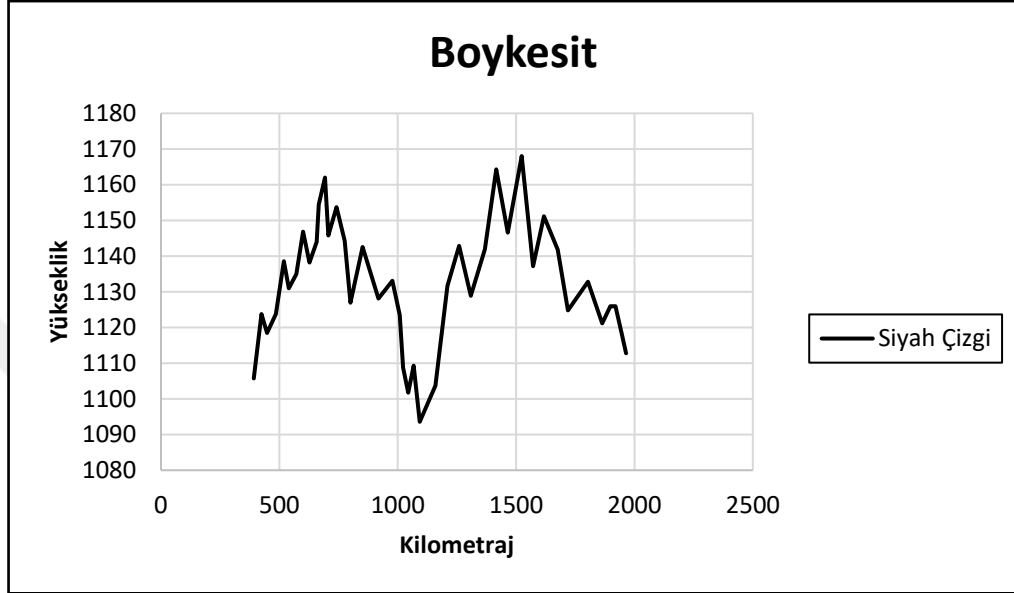


Şekil 3.2. Örnek kırmızı çizgi

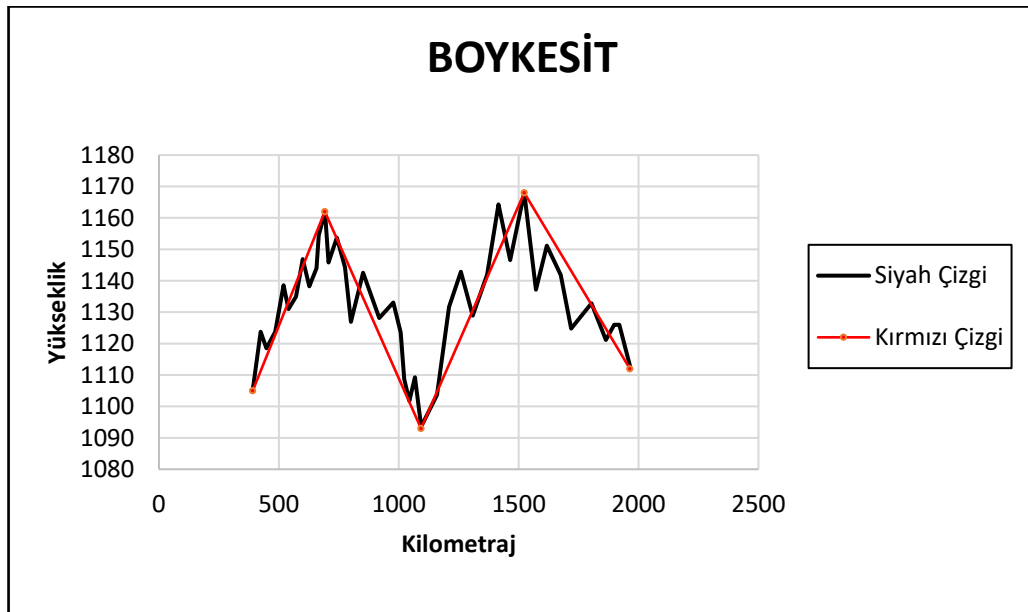
3.1.3. Some Noktalarının Otomatik Olarak Belirlenmesi

Kırmızı çizginin oluşturduğu yarma dolgu miktarlarını eşit tutmak için öncelikle boykesit üzerinde some noktaların belirlenmesi ve bu noktaların proje üzerinde nerelerden geçirileceği en önemli hususlardan biri haline gelip ve kesinlikle göz önünde bulundurulmalıdır. Günümüzde some noktalarının belirlenmesi en gelişmiş bilgisayar programlarında bile ya tasarlayan kişi tarafından elle seçiliyor veya basit regresyon yöntemleri kullanılarak program tarafından belirleniyor ki bu da optimum yarma dolgu elde etmemiz için kabul edilecek bir some noktası seçimi olmayacaktır.

Bu çalışma kapsamında, arazinin doğal halini gösteren verileri siyah çizgi olarak Şekil 3.3'te gösterilmiştir. some noktaları yazılım tarafından yükselen tepelere göre ve düşen derelere göre otomatik olarak belirlenmesi gerekmektedir. Şekil 3.4'te gösterildiği gibi kırmızı çizginin some noktaları yazılım tarafından otomatik seçildiği takdirde yarma-dolgu miktarlarının eşit tutulma durumu yüksek olacaktır.



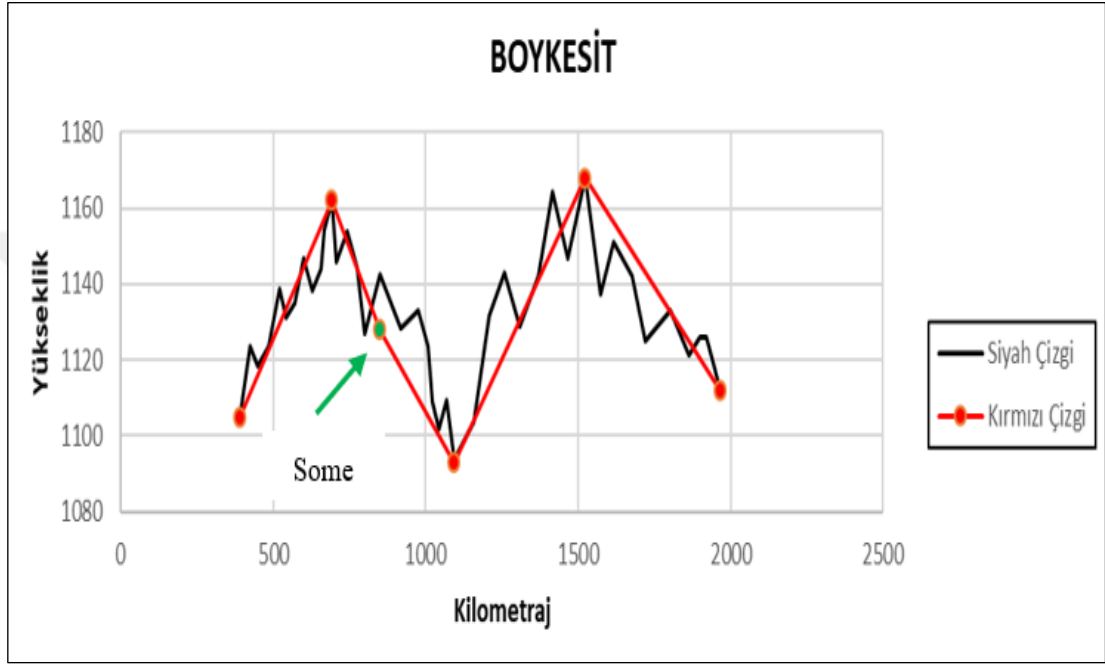
Şekil 3.3 Siyah çizgi



Şekil 3.4. Some noktalarının belirlenmesi

3.1.4. Zorunlu Some Noktalarının Eklenmesi

Yukarıda da gösterildiği gibi yol boyunca some noktalarının sadece yazılım tarafından otomatik seçilmesi tam olarak doğru olmayacaktır yani bazı projelerde projenin işleyişi kapsamında zorunlu some noktaları Şekil 3.5'te de görüldüğü gibi tasarımcı tarafından elle eklenmesi gerekmektedir. Tabii ki bu durumu yazılım ile göstermemiz gerekiyorsa satırlarca kodlama sayesinde programa entegre edebiliriz.



Şekil 3.5. Zorunlu some nokta eklenmesi

Şekil 3.6'da bu kodlamadan sadece bir kısmı gösterilmektedir. Bu kodlama sayesinde önce yazılım tarafından kırmızı çizginin some noktalarıyla ilgili bulunan otomatik some noktalarına ek olarak eklenmesi gereken zorunlu some noktasının olup olmadığı sorulmaktadır. Eklenmesi gereken some noktası sorusuna evet cevabı verildiği takdirde proje kapsamında toplam eklenmesi gereken some noktasının sayısı yazılım tarafından sorulmaktadır. Toplam eklenmesi gereken some noktalarının sayısı girildikten sonra o some noktalarına ait koordinatlar istenmektedir. Tabi ki X eksenini sorulduğunda some noktasının bulunduğu kilometrajı ve Y eksenini sorulduğunda o some noktasına ait yükseklik kodu istenmektedir. Böylece ekstradan eklememiz gereken some noktalarının koordinatları yazılıma verilerek artık otomatik bulunan some noktalarına ek olarak zorunlu some noktalarda eklenmiş olacaktır.

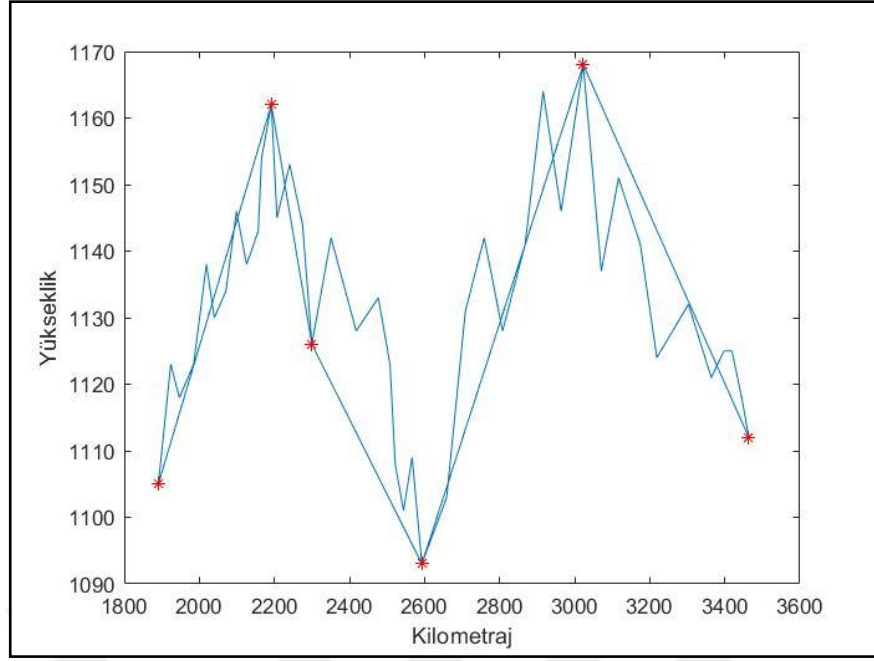
```

prompt = 'Do you want to insert peak Y/N: ';
yesno = input(prompt, "s");
if yesno=='y'
number_range = input('Please input the insert number :', "s");
number_range=str2num(number_range);
user_range_insert=[];
for i=1:number_range
prompt = "please input x "+num2str(i)+" :";
number_range = input(prompt, "s");
number_rangex=str2num(number_range);
user_range_insert(i,1)=number_rangex;
prompt = "please input y "+num2str(i)+" :";
number_range = input(prompt, "s");
number_rangey=str2num(number_range);
user_range_insert(i,2)=number_rangey;
end
else
    user_range_insert=[];
end

```

Şekil 3.6. Zorunlu some noktası eklenmesinin yazılım kodlaması

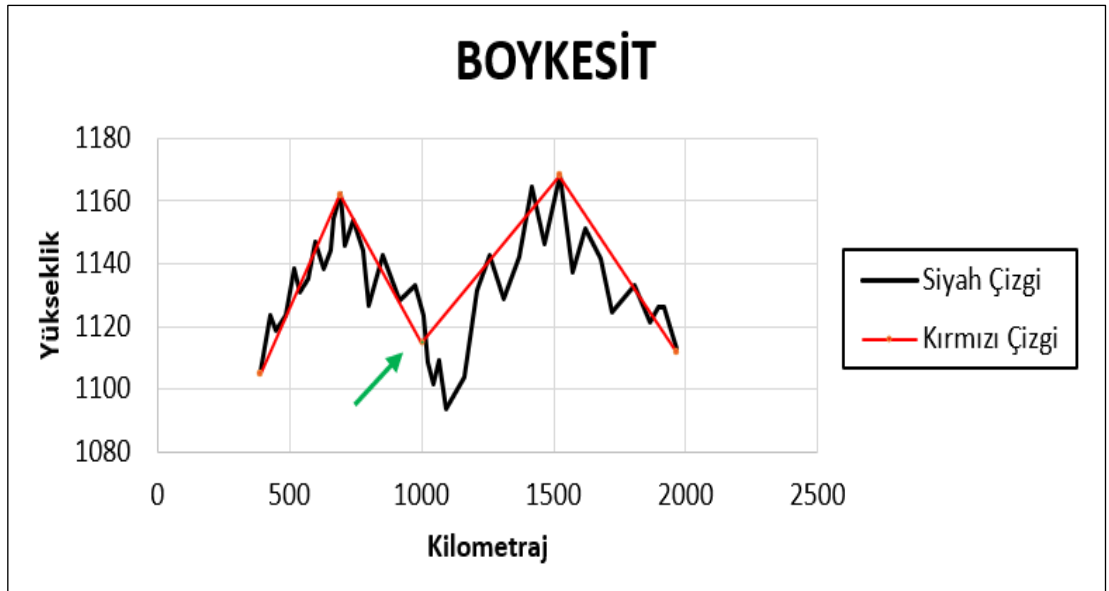
Örnek olarak yukarıdaki projenin some noktalarına ek olarak, bir tane some noktası koordinatı eklenmiş olup ve yazılıma entegre edilmiştir ve buna bağlı olarak örnek proje sonuçları Şekil 3.7’de görülmektedir.



Şekil 3.7. Some noktası eklenmesinin yazılım kodlaması

3.1.5. Yasak Bölgelerdeki Some Noktaların Kaldırılması

Proje işleyişi için Şekil 3.8'deki gibi, bazı aralıklarda hiç some noktası bulundurulmamalıdır. Bu bağlamda yazılımın eksik kalmaması adına belli aşamada, yazılım tarafından tasarımcıya, some noktası bulunmaması gereken aralıklar sorulacaktır.



Şekil 3.8. Some noktası bulunmaması gereken aralık

Yazılım tarafından Şekil 3.9'da görüldüğü gibi önce kaldırılması gereken some noktalarının olup olmadığı tasarımcıya sorulacaktır. Bu sayede proje gereği some

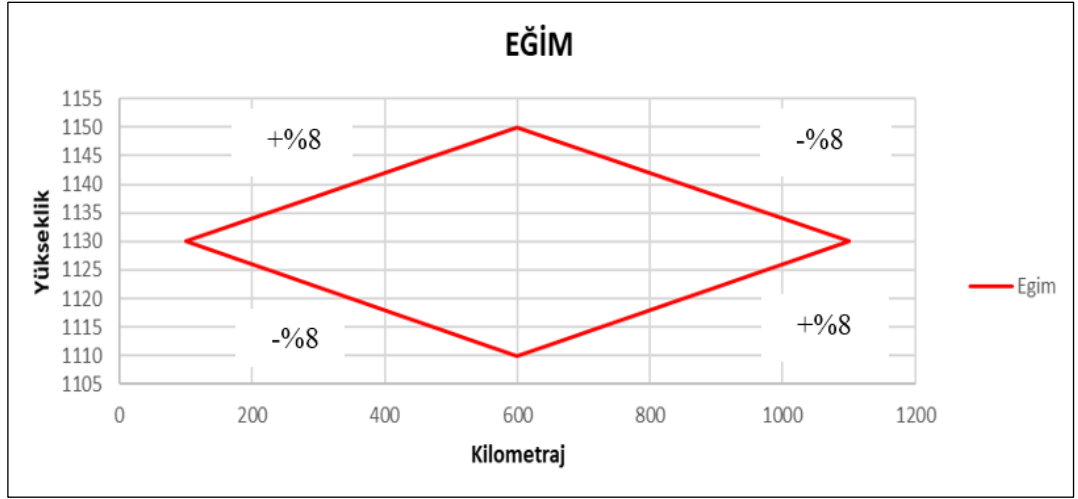
noktalarının bulunmaması gereken aralıkları var ise tasarımcı tarafından girilecektir. Tabi ki bu aşamada yazılım, some noktası bulunmaması gereken aralıkların başlangıç ve bitiş kilometrajlarına ihtiyaç duyacak ve bu aralıklar girildiği takdirde some noktası bulunuyorsa kaldırılmış olacaktır.

```
prompt = 'Do you want to remove the range data for peak Y/N:';
yesno = input(prompt, "s");
if yesno=='y'
    number_range = input('please input the number of Range:', "s");
    number_range=str2num(number_range);
    user_range=[];
    for i=1:number_range
        prompt = "please input x start "+num2str(i)+" :";
        number_range = input(prompt, "s");
        number_rangex=str2num(number_range);
        user_range(i,1)=number_rangex;
        prompt = "please input x end "+num2str(i)+" :";
        number_range = input(prompt, "s");
        number_rangey=str2num(number_range);
        user_range(i,2)=number_rangey;
    end
else
    user_range=[];
end
end
```

Şekil 3.9. Some noktası kaldırımı için gereken yazılım kodlaması

3.1.6. Kırmızı Çizgi Eğimleri

Karayolu şartnamesinde her yol sınıfı için eğim sınırları belirlenmiştir. Bu bağlamda, kırmızı çizginin yukarı ve aşağı yönündeki maksimum ve minimum eğimleri Şekil 3.10'da gösterildiği gibi belirlenmelidir. Başka bir deyişle, bir yol projesi maksimum ve minimum eğimlere göre tasarlanıp ve proje boyunca kırmızı çizginin eğimi bu sınırları geçmemelidir.



Şekil 3.10. Maksimum ve minimum eğimler

Tasarlanan yazılım tarafından maksimum ve minimum eğimler Şekil 3.11’de gösterildiği gibi tasarlayıcıdan sorulacaktır. Bu çalışma kapsamında projenin maksimum eğimi için %8 ve minimum eğim içinde %0,5 kabul edilerek kırmızı çizgi tasarımı belirlenen eğim sınırları arasında gerçekleştirilecektir. Buna bağlı olarak da eğim sınırları dışında kalan kırmızı çizgi parçaları devre dışı bırakılarak hesaplamalara dahil edilmeyecektir.

```
slope_valuemax = input('please input the max slope:', "s");
% slope_valuemax=0.08;
slope_valuemax=str2num(slope_valuemax);

slope_valuemin = input('please input the min slope:', "s");
% slope_valuemin=0.005
slope_valuemin=str2num(slope_valuemin);
```

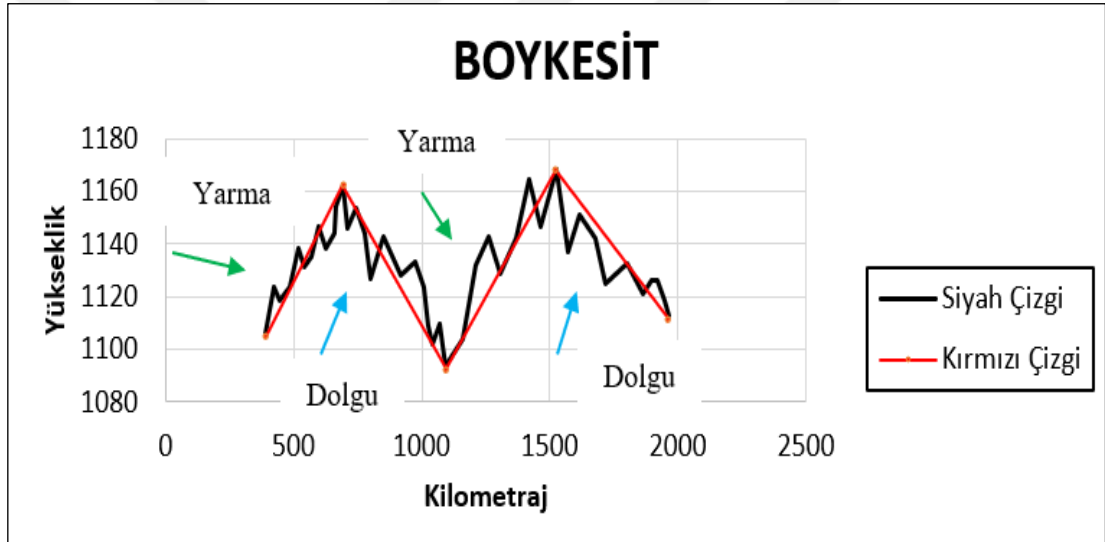
Şekil 3.11. Maksimum ve minimum eğimlerin yazılım kodlaması

3.1.7. Yarma-Dolgu Alanları

Kırmızı çizgi elemanları kaba bir değişle düz çizgilerden ve some noktalarında bulunan düşey kurbalardan oluşmaktadır. Yukarıda da anlatıldığı gibi kırmızı çizginin her elemanının optimize bir şekilde tasarımı yapıldığı taktirde, toplam hafriyat miktarını en aza indirme konusunda önemli bir rolü olacaktır. Bu bağlamda kırmızı çizginin düz olan kısımları ile siyah çizgi arasında kalan yarma-dolgu bölgelerinin eşit olması ve aynı zamanda bu miktarın en aza indirilmesi yapıldığı takdirde, proje

maliyeti bakımından çok daha ekonomik olmasına neden olacaktır. Bu sayede, yol projesine ait en uygun kırmızı çizginin tasarımı için yeni bir yöntem geliştirilmiştir.

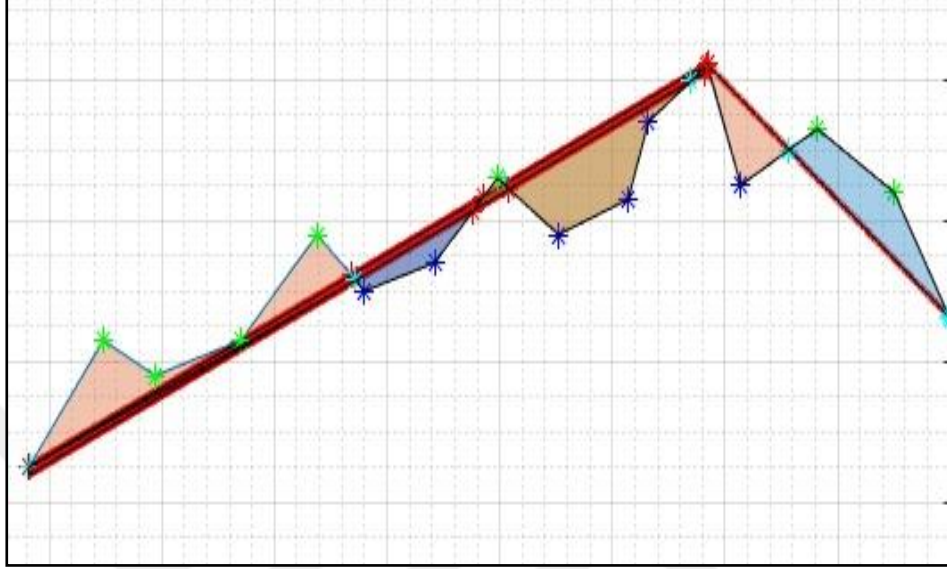
Kırmızı çizgi tasarımı yapılırken önce kırmızı çizginin düz kısımları siyah çizgi üzerinden rastgele geçirilmelidir. Ancak, bu işlem gerçekleştiğinde bu iki çizgi arasında kalan kapalı bölgelerin eşit tutulmasında fayda vardır. Başka bir deyişle, tasarımın optimize olması için kırmızı çizginin aşağı ve yukarısında kalan bölgeleri eşit olmalıdır. Ancak bu aşamada geçirilecek olan düz çizgi her ne kadar da göz ile ayarlanmış olursa olsun yine de hafriyat miktarını eşitlemek ve aynı zamanda en aza indirmek konusunda fazlasıyla eksik kalacaktır. Buna bağlı olarak, Şekil 3.12’de gösterildiği gibi kırmızı çizgi ile siyah çizgisinin kesiştiği noktaları dikkate alacak olursak bu iki çizgi arasında oluşan yarma-dolgu alanları gösterilmiştir.



Şekil 3.12. Yarma – dolgu alanları

Kırmızı çizginin altında ve yukarısında kalan noktalar yani arazinin kırılma noktaları, yazılım tarafından bulunarak ayrı ayrı veriler halinde toplanacaktır. Bu noktalar ile kırmızı çizgi arasında kapalı alanlar oluşturulduktan sonra, kırmızı çizginin yukarısında ve aşağısında oluşan bölgeler sırasıyla yarma ve dolgu alanları olarak kabul edilecektir. Bulunan bölgelerin alanları yazılım tarafından hesaplanacak ve olabildiğince eşitlenmeye çalışılacaktır. Bu çalışma kapsamında, hafriyat miktarında en aza indirmek için oluşan yarma-dolgu alanlarının miktarlarını da minimum seviyeye indirmiş olacağız. Tabi ki kırmızı çizgi tasarımında bu eşitlenme ve aynı zamanda en düşük miktara yaklaştırmak binlerce deneme yanılma yapılarak mümkün olacaktır. Bu bağlamda, kırmızı çizgi geçirilip ve tekrar tekrar yarma-dolgu

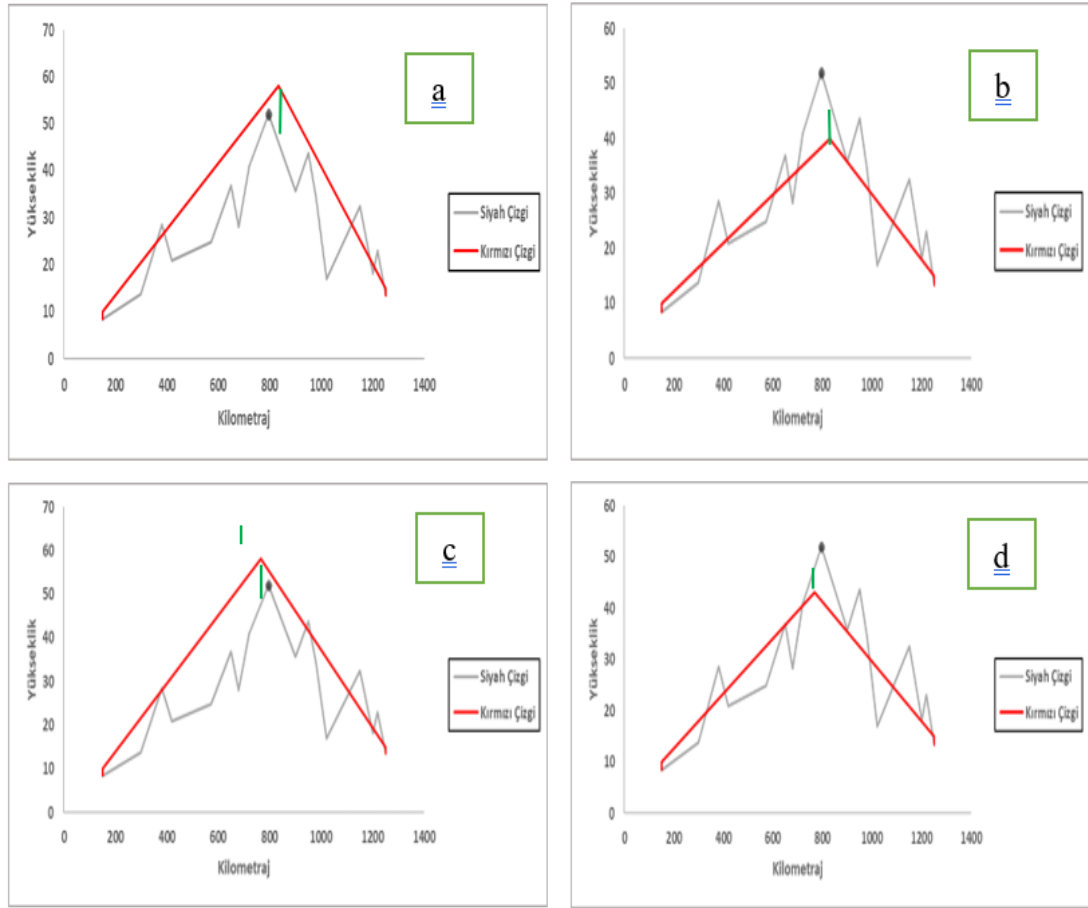
alanları hesaplanması gerekmektedir. Bu binlerce çizilen kırmızı çizgilerden en optimum miktara denk gelen kırmızı çizgi seçilip ve nihai kırmızı çizgi olarak kabul edilecektir. Örnek olarak Şekil 3.13'te gösterilmiş olan alanların eşit olması halinde en uygun kırmızı çizgi tasarımı gerçekleştirilmiş olacaktır.



Şekil 3.13. Yarma – dolgu alanlarındaki kırılma noktaları

3.1.8. Some Noktasının Araziye Göre Durumu

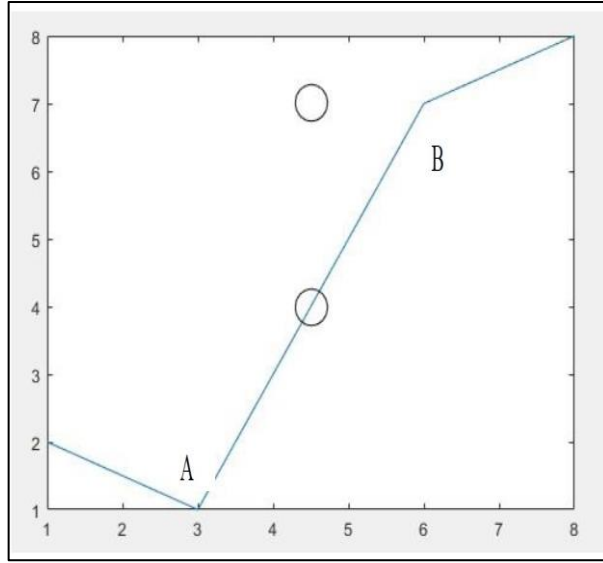
Kırmızı çizgiye ait some noktaları arazinin durumuna göre yer değiştirmektedir. Buna bağlı olarak, bu yer değiştirmeler yapıldığında some noktalarının yükseklik kotu da değişmektedir. Ayrıca, bu some noktaları siyah çizgi ile kesişerek, yarma-dolgu bölgelerini oluşturmaktadır. Some noktaların her yer değişiminde bu bölgelerin alanları tekrar hesaplanmaktadır. Kırmızı çizgi ile siyah çizgi arasında oluşan bölgelerin alan hesabı için aslında bu bölgelerin kapalı olması şarttır. Dolayısıyla her some noktası yazılım tarafından oluşturulduğunda o some noktası ile zemin çizgisinin arasında bir farazi çizgi geçirilerek o bölgeyi kapalı hale getirmemiz gerekmektedir. Şekil 3.14'te görüldüğü gibi pik noktası siyah bir nokta ile gösterilmiştir. Ancak, kırmızı çizgi üzerindeki some noktası bu pik noktasına göre yani siyah çizgiye göre 4 farklı durumda yer alabilir. Bu durumlar Şekil 3.14(a-d)'te 4 farklı durum ile gösterilmektedir.



Şekil 3.14. Some noktasının arazi üzerindeki durumu

Aşağıdaki formül 3.1'in sayesinde herhangi bir some noktasının koordinatı siyah çizgi üzerinde bulunacaktır. Böylece Şekil 3.15'te gösterildiği gibi some noktasının arazi üzerindeki koordinatı hesaplanmış olacaktır. A (a, b) ve B (c, d) noktalarının koordinatları arazi kotları olduğundan proje başından bilinmektedir. Some noktasının siyah çizgi üzerinde nereye denk geleceği yani X ve Y koordinatları aşağıdaki Formül 3.1'den bulunmaktadır. Tabi ki bu aşamada X koordinatı boykesit üzerinde kilometrajı göstermektedir. Başka bir deyişle, X koordinatı her some noktası için siyah çizgi ve kırmızı çizgi için aynıdır.

$$Y = \frac{X(d-b) - ad + cb}{c-a} \quad (3.1)$$



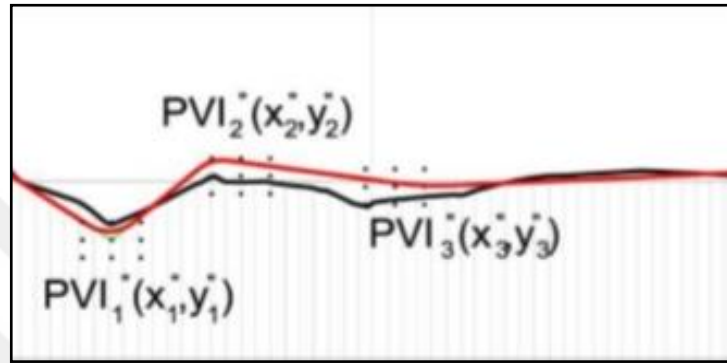
Şekil 3.15. Some noktasının arazi üzerindeki durumu

3.2. Tek-Amaçlı Meta-Sezgisel Optimizasyon Algoritmaları

Optimum kırmızı çizgi tasarımı için öncelikle farklı üç tane tek-amaçlı meta-sezgisel yöntem ile bir kırmızı çizgi tasarımı gerçekleştirilecektir. Tasarlanan kırmızı çizgilerin siyah çizgiye göre yukarı ve aşağısında kalan bölgelerinin alanları hesaplanacaktır. Bu bağlamda, 1995 tarihinde kenedy ve eberhart tarafından geliştirilen *PSO* yöntemi, 2005 tarihinde karaboğa tarafından önerilmiş *ABC* yöntemi ve 2012 tarihinde Xin Shi Yang tarafından önerilmiş *FA* yöntemleri kullanılmıştır. Bu yöntemler sayesinde kırmızı çizgi, siyah çizgi üzerinde çizilecek ve çizginin yukarısında yani yarma bölgelerinin alanları tek tek hesaplanacaktır ve aynı zamanda çizginin altında kalan bölgeler de yani dolgu kısımlarının da alanları hesaplanacaktır. Ayrıca Yarma-Dolgu farkı hesaplanıp optimizasyon yöntemlerine göre belirli iterasyonda tekrarlanacaktır. Çizilen binlerce kırmızı çizgi arasından bu farkı sıfıra yakın bir değer olanı seçip ve o çizgiyi en optimum kırmızı çizgi olarak kabul edecektir. Bu bağlamda yöntemler sayesinde binlerce deneme yanılma yapılarak alan farklarının sıfıra yakın bir değer bulunmasına yardımcı olacaktır. Bu şekilde bulunan çizgi aslında yöntemlerin alana göre optimizasyon yazılımı sayesinde gerçekleşecektir. Bu bağlamda yazılım programında kullanılan alan hesaplama yöntemi cross yöntemi ile hesaplanmaktadır. Bu şekilde bulunan alanların optimizasyonu ile ilgili yazılım kodlaması tasarlanıp ve meta-sezgisel algoritma yöntemlerine entegre edilerek daha tatmin edici sonuçlar elde edilmektedir.

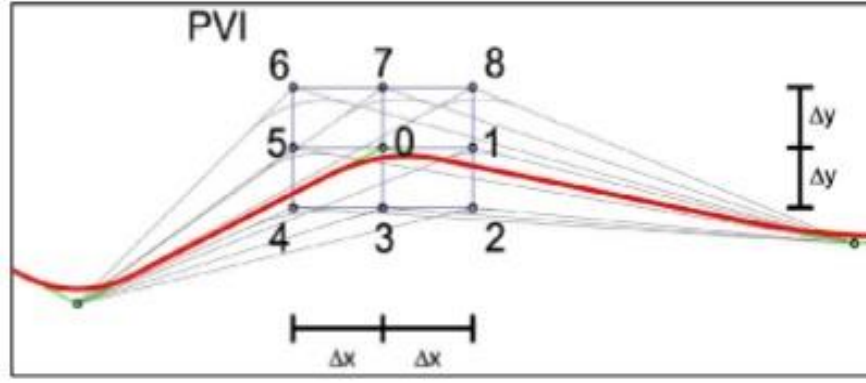
3.2.1. Kırmızı Çizgiye Ait Some Noktaların Optimizasyonu

Yukarıda anlatılan alanlar üzerinden optimizasyon, yarma-dolgulara ait alanlar farkının sıfır olduğu durumunu incelemektedir ki bu da hafriyatın en önemli noktası, yani kazılan yarma miktarını, dolgu alanlarında kullanarak ve geride yaklaşık sıfır fazla malzemeyle yapılmasıdır. Bu çalışma kapsamında, tasarlanan kırmızı çizgiye ait some noktaları, Şekil 3.16'da görüldüğü gibi koordinat sisteminin X ve Y eksenini üzerinde incelenmiştir.



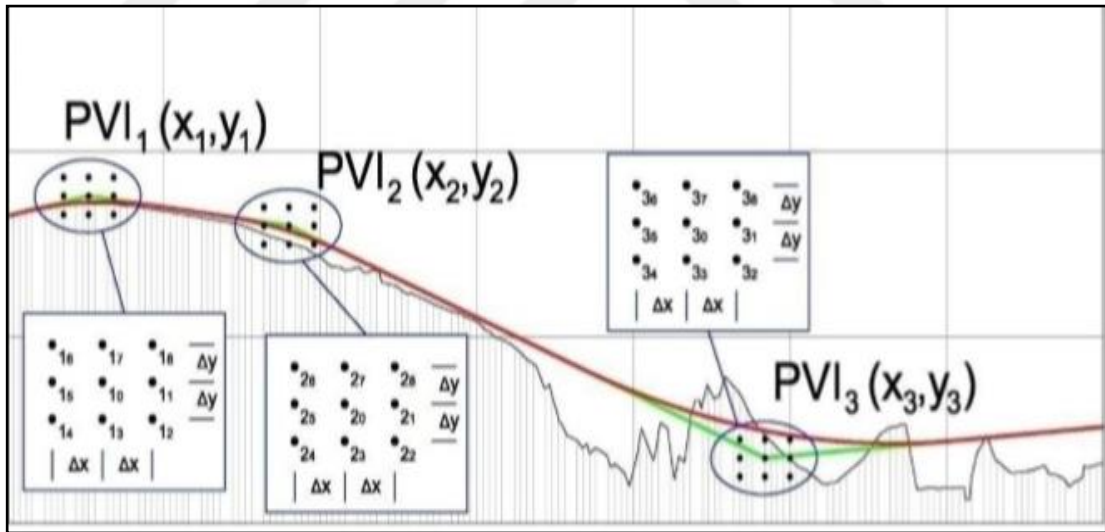
Şekil 3.16. Some noktaları (Özkan, 2013)

Some noktalarının optimizasyonu için her some noktasına ait bir matris oluşturularak koordinat sisteminin X ve Y yönünde incelemeler yapılmaktadır. Başka bir deyişle, Şekil 3.17'de görüldüğü gibi kabaca sıfır adı verilen noktayı yazılım tarafından seçilen otomatik some noktası kabul edersek, bu noktayı yukarıya veya aşağıya doğru hareket ettirdiğimiz takdirde belki de yarma dolgu miktarları değişmiş olacaktır ve hatta daha optimum bir kırmızı çizgi geçirmemize neden olabilecektir. Bu bağlamda, etrafında oluşan diğer 8 noktayı algoritma tarafından oluşturulan rastgele some noktaları olarak kabul edilmektedir. Tabiki bu rastgele noktaların sayısı ne kadar fazla olursa o kadar kırmızı çizgimiz daha optimize edilmiş olarak tasarlanmış olacaktır. Bu X ve Y yönünde değişmekte olan some noktaları her bir değişim yapıldığında yarma-dolgu alanları ona göre değiştiği için tekrar bu alanlar hesaplanacaktır. Böylece, bu yöntemin gerçekleşmesinde fazlasıyla matematiksel işlem olduğundan, bu karmaşık hesaplamaların yapılması için bir optimizasyon yönteminin kullanılması yadsınamaz bir gerçek haline gelmektedir.



Şekil 3.17. Her some noktası için farklı kırmızı çizgi (Özkan, 2013)

Genel olarak proje boyunca tasarlanmış bir kırmızı çizgi, çok fazla sayıda some noktasından geçebilmektedir. Bunun yanı sıra bu some noktalarının her birine ayrı ayrı matrisler oluşturularak ve her some noktasını binlerce değişken hale getirdiğimiz takdirde aslında projenin her yerine hâkim olacak sonsuzca some noktası ortaya çıkacaktır. Şekil 3.18’de örnek olarak some noktaları PVI ile gösterilmiş ve bu some noktalara ait oluşturulan matris sayesinde her some noktasının yeri 9 değişken hale getirilerek kırmızı çizgi tasarımı gerçekleştirilmiş olacaktır.



Şekil 3.18. Her some noktası için matris oluşumu (Özkan, 2013)

Some noktalarına ait matrislerin yatay ve dikey yönünde ne kadar bir açıklığı kapsayacağı yazılıma tanımlanmalıdır. Başka bir deyişle some noktalara, yer değiştirme konusunda bir aralık belirtilmelidir. Bu koordinat sisteminde her some noktası için X ve Y eksenini üzerinde hangi aralıklarda değişken olması Şekil 3.19’da yazılım kodlaması ile gösterilmiştir. Yazılım kodlamasında görüldüğü gibi başlangıç-bitiş ve orta some noktalarında değişkenlik aralığı ayrı ayrı girilmektedir. Bu

bağlamda, yazılım tarafından otomatik bulunan her some noktasının X ve Y koordinatına belirli bir mesafe ekleyerek, o mesafe kadar sağ, sol, yukarı ve aşağı tarafa bir açıklık tanımlanıp ve bu aralıklar içerisinde some noktasının yer değiştirmesine imkân sağlanacaktır.

```
VarMin(1)=y(1)-10;           % Lower Bound of Variables
VarMax(1)=y(1)+10;           % Upper Bound of Variables
VarMin(2)=y(cutterpeak_idx)-10; % Lower Bound of Variables
VarMax(2)=y(cutterpeak_idx)+10; % Upper Bound of Variables
VarMin(3)=y(end)-10;          % Lower Bound of Variables
VarMax(3)=y(end)+10;          % Upper Bound of Variables

VarMin(4)=x(1)-10;           % Lower Bound of Variables
VarMax(4)=x(1)+10;           % Upper Bound of Variables
VarMin(5)=x(cutterpeak_idx)-10; % Lower Bound of Variables
VarMax(5)=x(cutterpeak_idx)+10; % Upper Bound of Variables
VarMin(6)=x(end)-10;          % Lower Bound of Variables
VarMax(6)=x(end)+10;          % Upper Bound of Variables
```

Şekil 3.19. Her some noktası için matris oluşumu

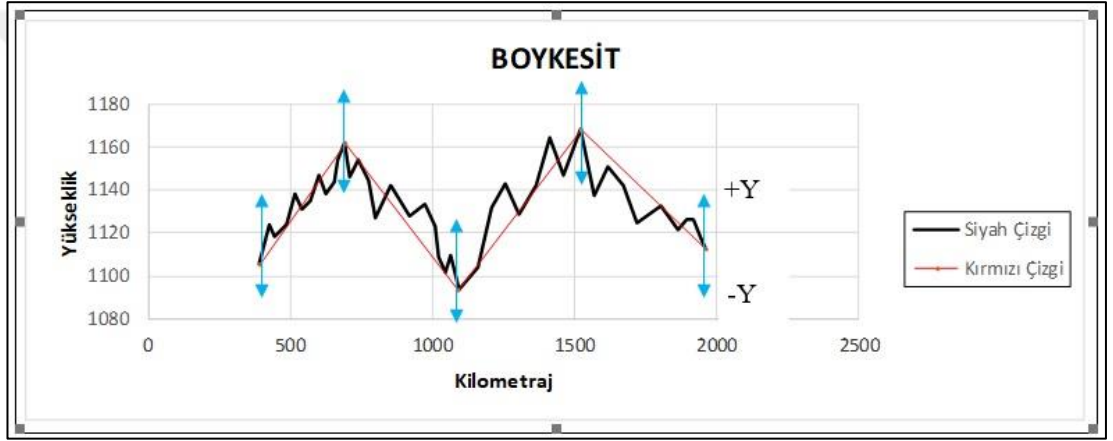
Ancak, yukarıda anlatıldığı gibi her some noktasının ne kadar bir değişkenlik gösterebileceği ayrı ayrı yazılıma girilmelidir. Bu durumu ortadan kaldırmak amacıyla, Şekil 3.20'de görüldüğü gibi proje tasarımcısından değişkenliğin kaç olması gerektiğini direkt sorarak ve tüm some noktalarına aynı değerleri uygulayarak netliğe kavuşturulmuştur. Tabi ki bu açıklıklar her iki yönde aynı olmak zorunda değildir. Bu çalışma kapsamında, some noktalarının yerlerini önce sadece tek bir yönde değiştirip ve sonuçları incelenecektir, Daha sonra some noktalarının yerlerini her iki yönde değiştirip sonuçları karşılaştırılacaktır.

```
% % xs = input('what is the size of change for x:', "s");
% % xs=str2num(xs);
xs=2;
% % ys = input('what is the size of change for y:', "s");
% % ys=str2num(ys);
ys=2;
change_size=[xs;ys];
```

Şekil 3.20. Her some noktası için değişkenlerin kodlaması

3.2.2. Y Ekseninde Optimizasyon

Bu aşamada, kullanılan örnek zemin kotları, bir karayolu projesinin eksen hattı olarak düşünülmüştür. Kırmızı çizgi, başlangıç ve bitiş noktalarından ve aynı zamanda orta some noktalarından oluşturulmuştur. Kırmızı çizginin düz kısımlarının kesiştiği noktalar yani some noktaların belirlenmesi ilk önce rastgele ve yaklaşık olarak seçilecektir. Ancak bu noktaların Y eksenine yani Şekil 3.18'deki gibi belirli bir aralıkta matris oluşturularak, değişken halinde denetimi yapılacaktır. Böylece, some noktalarının yerini değiştirerek ve binlerce nokta üzerinde deneme yanılma işlemi uygulayarak optimum kırmızı çizgi tasarımı elde edilecektir. Şekil 3.21'de some noktalarının yer değiştirme aralığı gösterilmiştir.



Şekil 3.21. Y ekseninde denetleme

Bu bağlamda, kullanılan meta-sezgisel yöntemlere entegre etme şartıyla, tasarlanacak olan kırmızı çizginin başlangıç, orta ve bitiş noktadaki some noktalarının Y eksenine, yukarı ve aşağı doğrultuda istediğimiz bir miktarı tanımlamalıyız. Mevcut değişkenler Tablo 3.1'de gösterilmiştir. Ayrıca, değişkenlerin pozitif ve negatif yönde ne kadar bir aralıkta değişim yapabilmesine ilişkin oluşturulan denklem, Formül 3.2'de gösterilmiştir. Bu formülde görüldüğü gibi Y_S sembolü ile gösterilmiş olan başlangıç noktasının gerçekte olan koordinatına A miktarı kadar artı veya eksi yaparak Y ekseninde yer değiştirmesine izin verilecektir. Aynı şekilde Y_P sembolü ile gösterilen orta some noktaları ve Y_E ile gösterilen bitiş noktasındaki some noktasında A değeri kadar bir aralık verilerek yer değiştirmelerine imkân verilmektedir. Böylece kırmızı çizginin some noktaları Y ekseninde binlerce yer değiştirecektir. Bu yer değiştirmelerinden oluşan yarma dolgu bölgelerinin alanları,

fazlasıyla karmaşık matematiksel işlemden sonra hesaplanacak ve aralarından en optimum miktara sahip olan kırmızı çizgi son olarak kabul edilecektir.

Tablo 3-1. Y eksenı deęiřkenleri

Nokta	Deęiřkenler		
	Bařlangıç	Some	Bitiř
Eksen	Y_S	Y_P	Y_E

$$Y_{b-Min} = Y_S - A \quad (3.2)$$

$$Y_{b-Max} = Y_S + A$$

$$Y_{s-Min} = Y_P - A$$

$$Y_{s-Max} = Y_P + A$$

$$Y_{b-Min} = Y_E - A$$

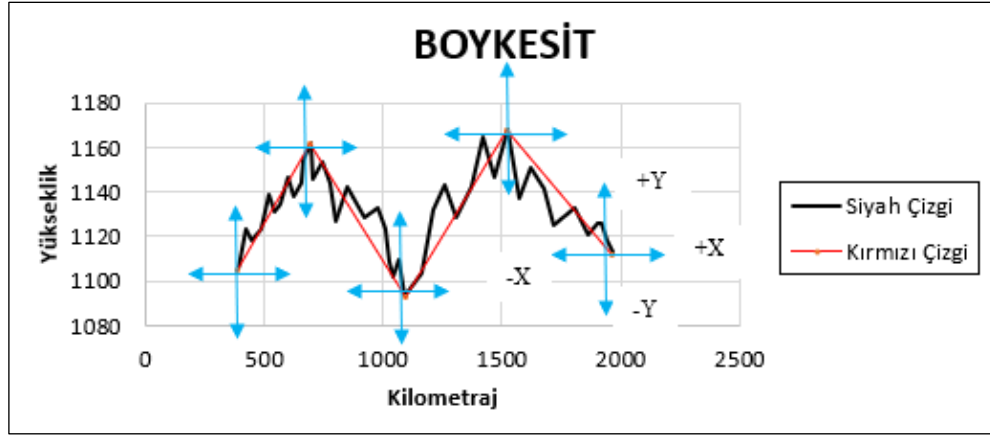
$$Y_{b-Max} = Y_E + A$$

3.2.3. X ve Y Eksenı Üzerinde Optimizasyon

Bu ařamada, kullanılan örnek zemin kotları, bir karayolu projesinin eksen hattı olarak düşünölmüřtür. Kırmızı çizgi, bařlangıç ve bitiř noktalarından ve aynı zamanda orta some noktalarından oluřmaktadır.

Kırmızı çizginin düz kısımlarının keřiřtięi noktalar yani some noktaların belirlenmesi, ilk önce rastgele ve yaklaşık olarak seçilecektir. Ancak, bu noktaların X ve Y eksenine yani řekil 3.18'deki gibi belirli bir aralıkta matris verilerek, deęiřken halde denetimi yapılacaktır. Böylece, some noktalarının yerini deęiřtirerek ve binlerce nokta üzerinde deneme yanılma iřlemi uygulayarak optimum kırmızı çizgi tasarımı elde edilecektir. řekil 3.22'de some noktalarının yer deęiřtirme aralıęı gösterilmiřtir.

Bu ařamada kırmızı çizgi tasarımı ve optimizasyonu, koordinat sisteminin her iki ekseninde yani hem X hem de Y eksenı üzerinde aynı zamanda yapılacaktır. Bu durumda, yarma-dolgu miktarı en aza indirilerek ve aynı zamanda sıfıra yakın bir deęer bulunarak kırmızı çizgi tasarımı gerçekteřtirilecektir.



Şekil 3.22. X ve Y ekseninde denetleme

Bu bağlamda, kullanılan meta-sezgisel yöntemlere entegre etme şartıyla, tasarlanacak olan kırmızı çizginin başlangıç, orta ve bitiş noktadaki some noktalarının X ve Y eksenine, yukarı-aşağı ve sağ-sol doğrultusunda istediğimiz bir miktarı tanımlamalıyız. Mevcut değişkenler Tablo 3.2'de gösterilmiştir. Ayrıca, değişkenlerin pozitif ve negatif yönde ne kadar bir aralıkta değişim yapabilmesine ilişkin oluşturulan denklem, Formül 3.3'te gösterilmiştir. Bu formülde görüldüğü gibi Formül 3.2'in dışında X ekseninde de bir aralık verilecektir. Böylece, bir some noktasının her iki eksen üzerinde yer değiştirilmesi sağlanacaktır. X_S Sembolü ile gösterilmiş olan başlangıç noktasının gerçekte olan koordinatına A miktarı kadar artı veya eksi yaparak X ekseninde yer değiştirmesine izin verilecektir. Aynı şekilde X_P sembolü ile gösterilen orta some noktaları ve X_E ile gösterilen bitiş noktasındaki some noktasında A değeri kadar bir aralık verilerek yer değiştirmelerine imkân verilmektedir. Böylece kırmızı çizginin some noktaları X ve Y ekseninde binlerce yer değiştireceklerdir. Bu yer değişmelerinden oluşan yarma dolgu bölgelerinin alanları, fazlasıyla karmaşık matematiksel işlemle sonra hesaplanacak ve aralarından en optimum miktara sahip olan kırmızı çizgi son olarak kabul edilecektir.

Her iki eksen üzerinde denetleme için değişkenler Tablo 3.2'de gösterilmektedir. Bu bağlamda her iki yönde belirli bir miktarda gösterildiği gibi değişiklik yapılacaktır.

Tablo 3.2. X-Y ekseninde değişkenleri

Nokta	Değişkenler		
	Başlangıç	Some	Bitiş
1. Eksen	X_S	X_P	X_E
2. Eksen	Y_S	Y_P	Y_E

Y ekseninde kullanılan denklemler Formül 3.2'te ve X ekseninde için de gereken denklemler Formül 3.3'te gösterilmiştir.

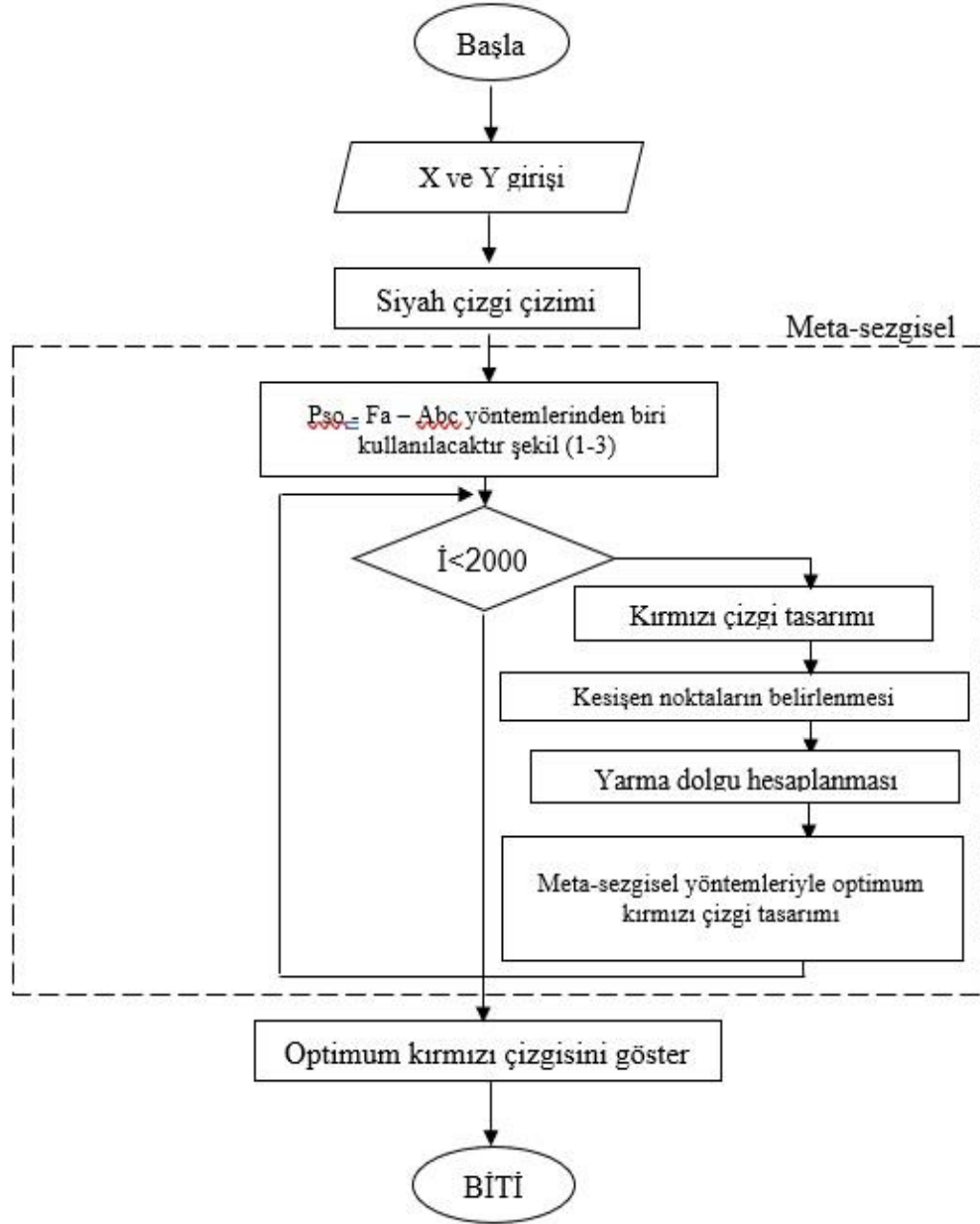
$$\begin{aligned}
 X_{b-Min} &= X_S - A \\
 X_{b-Max} &= X_S + A \\
 X_{S-Min} &= X_P - A \\
 X_{S-Max} &= X_P + A \\
 X_{b-Min} &= X_E - A \\
 X_{b-Max} &= X_E + A
 \end{aligned}
 \tag{3.3}$$

Yukarıda anlatılan yöntemler uygulandığında tasarlanacak olan kırmızı çizgide, hafriyat miktarı en aza indirilecek ve aynı zamanda yarma-dolgu miktarının eşit olmasına çalışılacaktır.

3.2.4. Kırmızı Çizgiye Ait Tek-Amaçlı Meta-Sezgisel Akış Şeması

Bu çalışma kapsamında, hafriyat miktarını en aza indirerek ve aynı zamanda yarma-dolgu miktarında eşitleyerek, maliyet miktarının en aza düşürülmesi sağlanacaktır. Bu bağlamda, önce Şekil 3.23'te gösterilmekte olan akış şemasına göre kırmızı çizgi hesaplama algoritması tasarlanacaktır. Daha sonra ise, bu algoritma sayesinde karayolu boykesit tasarımı aşamasında dikkate alınan herhangi bir some noktasını binlerce değişken hale getirerek en optimum kırmızı çizgi tasarımı gerçekleştirilecektir. Böylece, kırmızı çizgi optimizasyonu ile ilgili en önemli adımlardan biri ortaya konulmuş ve optimum bir kırmızı çizgi yapılmasına olanak sağlayacak düzenlemenin altyapısı (akış şeması) gerçekleştirilmiştir.

Bu akış şemasında belirtildiği gibi önce araziden topladığımız eksen hattının koordinatı belirlenecektir. Daha sonra bu koordinatlara göre siyah çizgi geçirilecektir. Meta-sezgisel optimizasyon yöntemlerinden PSO, FA ve ABC yöntemleri kullanılarak kırmızı çizgi optimizasyonu gerçekleştirilecektir. Bu bağlamda, önce some noktaları otomatik olarak yazılım tarafından rastgele seçilecek ve daha sonra bulunan some noktalarına Y yönünde ve daha sonra X ve Y yönünde bir aralık verilerek some noktalarına yer değiştirme olanağı sağlanacaktır. Tabi ki yazılım tarafından bu yer değiştirme 2000 defa tekrar edilecek ve her defasında yarma-dolgu alanları hesaplanacaktır. Bu hesaplamalardan hangisi en az ve aynı zamanda eşit ise yazılım tarafından seçilecek ve son olarak gösterilecektir.



Şekil 3.23. Tek-amaçlı meta-sezgisel algoritmaya ilişkin akış şeması

Ancak, işlemlerin daha sağlıklı olmasından dolayı yukarıda anlatılan algoritma, yazılım tarafından 30 kez daha tekrarlanacak ve her defasında yazılım tarafından optimum kırmızı çizgi tasarlanacaktır. Dolayısıyla, bu 30 tane bulunan optimum kırmızı çizgi arasından, yarma-dolgu miktarı en az olan ve aynı zamanda neredeyse sifıra yakın yarma-dolgu farkı gösteren kırmızı çizgi seçilecektir. Tabi ki bu işlemler her 3 meta-sezgisel optimizasyon yöntemi için de ayrı ayrı denenecektir.

3.3. Çok-Amaçlı Meta-Sezgisel Optimizasyon Algoritmaları

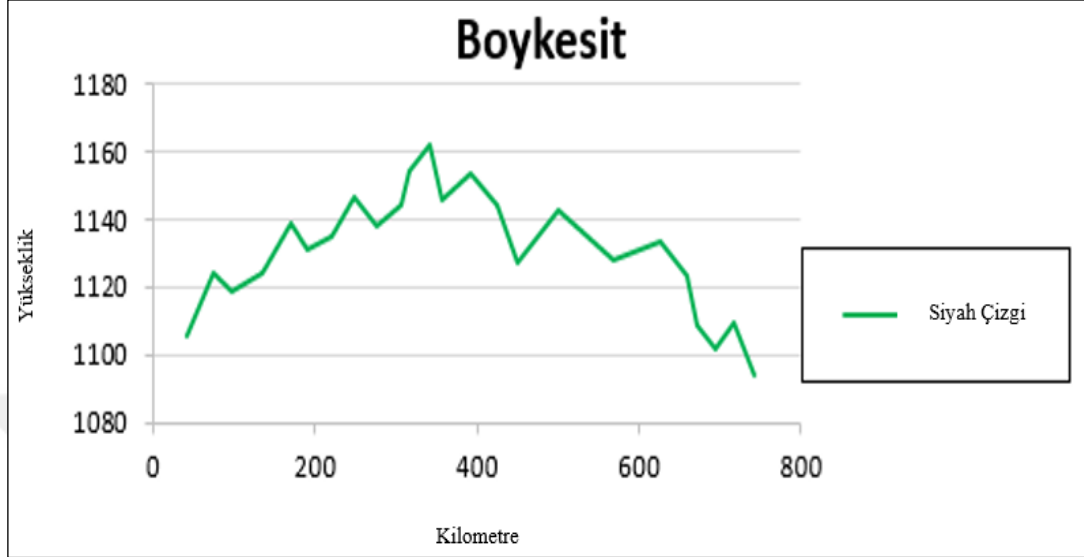
Bu çalışmada, tasarlanan kırmızı çizgilerin siyah çizgiye göre optimizasyonu yapılmaktadır. Bu bağlamda, tek-amaçlı meta-sezgisel algoritmaların dışında, çok-amaçlı meta-sezgisel optimizasyon algoritmalarını da kullanarak kırmızı çizgi optimizasyonu yapılabilmektedir. Böylece kırmızı çizginin yukarısında ve aşağısında kalan kapalı alanlar hesaplanacaktır. Ayrıca, yarma-dolgu farkı da hesaplanıp, optimizasyon yöntemleri ile binlerce deneme yapıldıktan sonra bu fark, sıfıra yakın bir değer olana kadar devam edilecektir. Tüm bu işlemler sonucunda bulunan çizgi, en optimum kırmızı çizgi olarak kabul edilecektir. Bu bağlamda, meta-sezgisel yöntemlerine entegre edilecek yazılım sayesinde oluşan yarma-dolgu bölgelerinin alanları hesaplanacak ve alan farklarının sıfıra yakın bir değer bulunmasına yardımcı olacaktır. Bu şekilde bulunan çizgi, aslında meta-sezgisel yöntemlerine entegre edilmiş kırmızı çizgi optimizasyon yazılımı sayesinde gerçekleşecektir. Başka bir deyişle, hazırlanan algoritma, meta-sezgisel algoritmalara entegre edilerek, cross yöntemi ile alanların hesaplamasıdır. Ancak, günümüzde kırmızı çizgi optimizasyonunda genellikle sadece tek-amaçlı yöntemler kullanılmaktadır. Ama, kırmızı çizgi tasarımı fazlasıyla karmaşık matematiksel işlemlerden oluştuğu için optimizasyon yöntemlerinden çok-amaçlı optimizasyon algoritmalarının kullanımı daha mantıklı olacaktır. Bu bağlamda, çok-amaçlı optimizasyon algoritmalarından en popüler olanlarından üç tanesi (*NSGA3*, *MOPSO* ve *SPEA2*) mercek altına alınacaktır. Böylece, kırmızı çizginin optimizasyonu için hazırlanan yazılım ile çok-amaçlı optimizasyon algoritmaları entegre edilerek daha az hafriyat miktarı ve yarma-dolgu dengelemesi için iki faktör veya iki hedef üzerinden yola çıkarak daha detaylı inceleme yapılmış olacaktır. Bu bağlamda, tasarlanan algoritmanın sayesinde sonuçlar daha istikrarlı ve güvenilir olabilecektir. Tabi ki bu işlemler her 3 meta-sezgisel optimizasyon yöntemi için de ayrı ayrı denenecektir.

3.3.1. Çok-Amaçlı Meta-Sezgisel Algoritması Kullanarak Boykesit Tasarımı

Karayolları projelerinin en önemli kısmı boykesit tasarımının yapılmasıdır. Bu bağlamda, maliyeti düşürmek için hafriyat miktarını en aza indirmeliyiz. Böylece, boykesit elemanlarının optimum tasarımı, hafriyat miktarını doğrudan etkileyeceği yadsınamaz bir gerçektir.

3.3.2. Araziye ait Siyah Çizgi

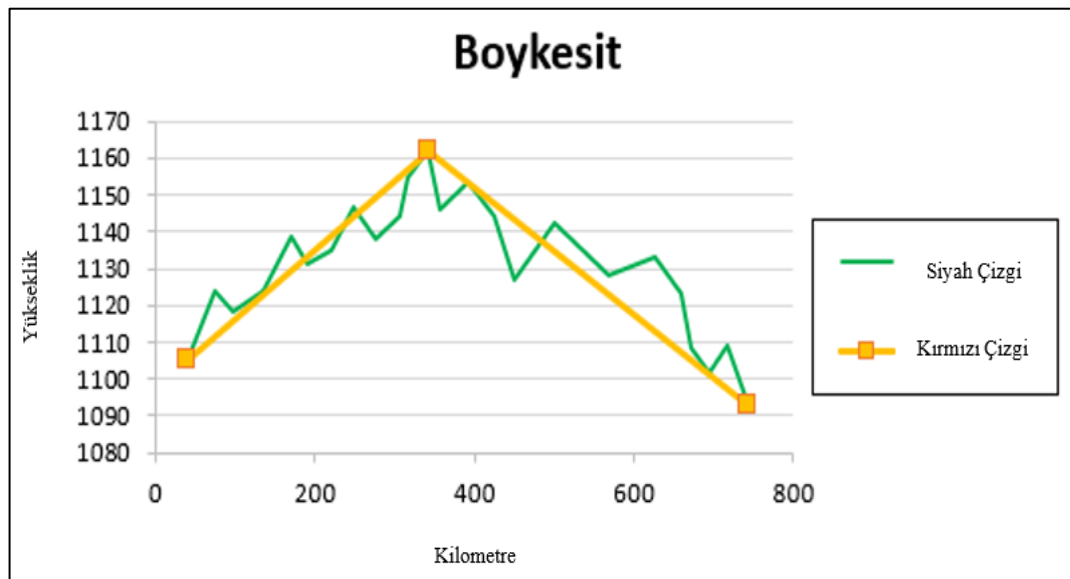
Boykesit tasarımında X eksenı mesafeyi ve Y eksenı yükseklięi göstermektedir. Bu bağlamda örnek bir siyah çizgi Şekil 3.24'te gösterilmektedir.



Şekil 3.24. Siyah çizgi

3.3.3. Kırmızı Çizgi Tasarımı

Günümüzde boykesit tasarımında some noktaları belirlenerek kırmızı çizgi geçirilmektedir. Zemin hattı ve kırmızı çizgi arasında kalan yarma ve dolgunun miktarının en az olması ve aynı zamanda eşit olması gerekmektedir. Kırmızı çizgi tasarımı, örnek siyah çizgi üzerinde yani Şekil 3.25'te gösterilmektedir.

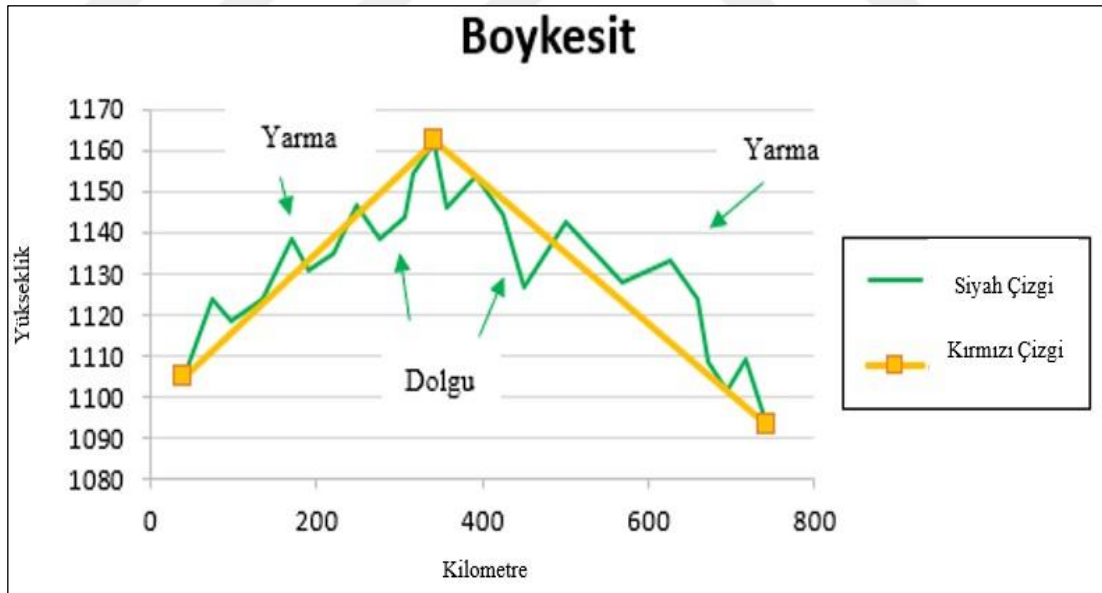


Şekil 3.25. Kırmızı çizgi

3.3.4. Yarma – Dolgu Alanların Belirlenmesi

Kırmızı çizgi genel olarak doğru parçalardan ve düşey kurplardan oluşmaktadır. Bu kapsamda, kırmızı çizginin doğru parçaları, zemin durumuna göre yarma-dolguyu eşit tutacak bir şekilde geçirilmelidir.

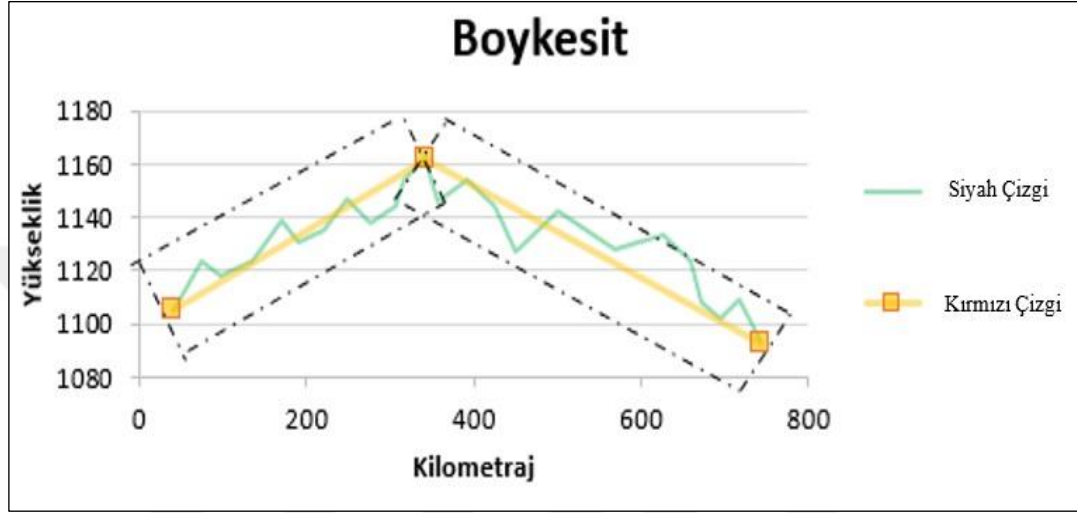
Kırmızı çizgi tasarımında önce kırmızı çizginin doğru parçaları siyah çizginin aşağı ve yukarısında kalan bölgeleri eşitleyecek bir şekilde geçirilmelidir. Ancak, bu aşamada geçirilecek olan doğru parçalar, çok-amaçlı meta-sezgisel optimizasyon algoritmaları kullanılarak gerçekleştirilecektir. Bu durumda, bir önceki bölümde anlatılan tek-amaçlı meta-sezgisel optimizasyon algoritmalarına göre proje boyunca tüm verilerin tek parça halinde göz önünde bulundurmak yerine, projeyi some noktalarından parçalar haline getirerek optimum kırmızı çizgi tasarımı gerçekleştirilecektir. Buna bağlı olarak, bu yazılım sayesinde kırmızı çizginin yukarısında (Yarma) ve aşağısında (Dolgu) bulunan bölgelerin alanları hesaplanıp ve eşitlenecektir. Aynı zamanda bu alanların miktarlarını da minimum seviyede tutarak hafriyat miktarı en aza düşürülecektir. Şekil 3.26’da gösterilen alanların eşit olması halinde en ekonomik kırmızı çizgi tasarımı gerçekleştirilecektir.



Şekil 3.26. Yarma – Dolgu Alanları

Bu çalışma kapsamında tasarlanan kırmızı çizgiler, yukarıda anlatılan tek-amaçlı meta-sezgisel yöntemler gibi tüm veriler üzerinde değil belki tasarlanan çok-amaçlı meta-sezgisel optimizasyon algoritmalarını kullanarak kırmızı çizginin doğru parçalarını, some noktalarından parçalayarak tasarımcıya daha detaylı ve ince tasarım

imkânı sunacaktır. Başka bir deyişle, tasarlamamız gereken projeyi parçalar haline getirerek böylece, çok-amaçlı optimizasyon algoritması daha az veri üzerinde işlem yapmış olacaktır. Dolayısıyla, bu algoritmalar proje üzerinde binlerce deneme yanılma yaparak aslında daha hızlı ve daha gerçekçi sonuçlar elde etmemize olanak sağlayacaklardır. Kırmızı çizginin some noktalarını baz alarak parçalama işlemi Şekil 3.27’de gösterilmektedir.



Şekil 3.27. Tüm verilerin parçalar haline getirilmesi

3.3.5. Çok-Amaçlı Algoritmalar Kullanarak X ve Y Eksenini Üzerinde Optimizasyon

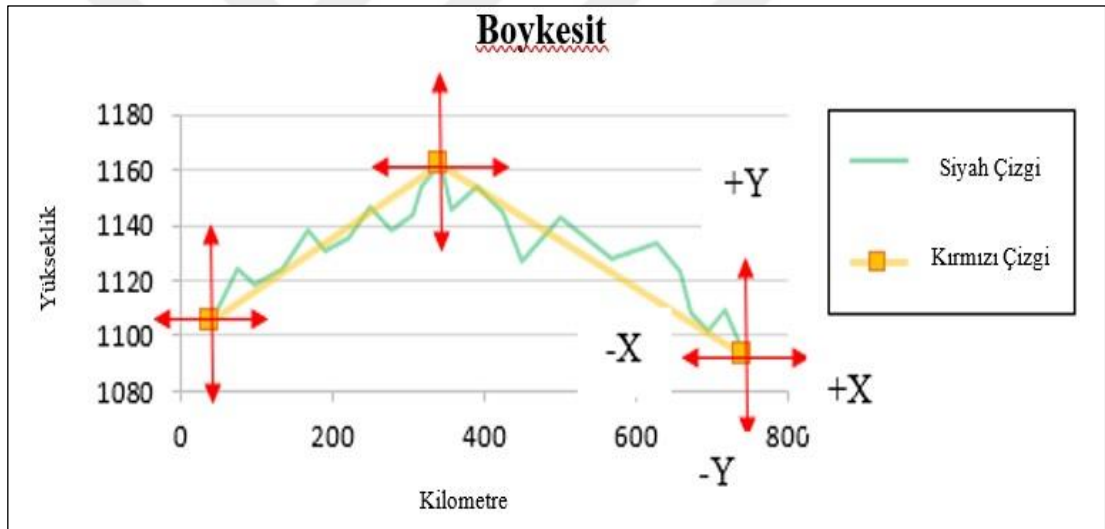
Bu aşamada, kullanılan örnek zemin kotları, bir karayolu projesinin eksen hattı olarak düşünülmüştür. Kırmızı çizgi, başlangıç ve bitiş noktalarından ve aynı zamanda orta some noktalarından oluşmaktadır.

Kırmızı çizginin düz kısımlarının kesiştiği noktalar yani some noktaların belirlenmesi, ilk önce rastgele ve yaklaşık olarak seçilecektir. Daha sonra, siyah çizgiye ait koordinatlar tam olarak some noktalarından, Şekil 3.27’deki gibi parçalar haline getirilecektir. Dolayısıyla, tüm incelemeler her parça için ayrı ayrı yapılacak ve en son birleştirilecektir. Bu aşamada birleştirme, ilk parçanın optimizasyonu yapılacak ve bitiş noktası diğer parçanın başlangıcı olarak kabul edilecektir. Ancak, bu noktaların X ve Y eksenine yani Şekil 3.28’deki gibi belirli bir aralıkta matris verilerek, değişken halde denetimi yapılacaktır. Böylece, some noktalarının yerini değiştirerek ve binlerce nokta üzerinde deneme yanılma işlemi uygulayarak optimum kırmızı çizgi

tasarımı elde edilecektir. Şekil 3.28'de some noktalarının yer değiştirme aralığı gösterilmiştir.

Bu aşamada kırmızı çizgi tasarımı ve optimizasyonu, koordinat sisteminin her iki ekseninde yani hem X hem de Y ekseninde aynı zamanda yapılacaktır. Bu durumda, yarma-dolgu miktarı en aza indirilerek ve aynı zamanda sıfıra yakın bir değer bulunarak kırmızı çizgi tasarımı gerçekleştirilecektir.

Bu bağlamda, kırmızı çizgi tasarım ve optimizasyonu X ve Y ekseninde ayrı ayrı parçalar haline getirilerek ve aynı zamanda her some noktasından veriler kesilip ve toplanarak yapılmaktadır. Başka bir deyişle, çok-amaçlı meta-sezgisel optimizasyon algoritması kullanarak sadece o parçanın incelemesi gerçekleştirilecektir. Bu parçalar halinde bulunan kırmızı çizgiler birleşip ve tek bir optimum kırmızı çizgi elde edilecektir.



Şekil 3.28. X ve Y ekseninde denetleme

Bu bağlamda, kullanılan meta-sezgisel yöntemlere entegre etme şartıyla, tasarlanacak olan kırmızı çizginin başlangıç, orta ve bitiş noktasındaki some noktalarının X ve Y eksenine, yukarı-aşağı ve sağ-sol doğrultusunda istediğimiz bir miktarı tanımlamalıyız. Ayrıca, değişkenlerin pozitif ve negatif yönde ne kadar bir aralıkta değişim yapabilmesinde gereken denklemlerle yazılıma tanımlamalıyız. Böylece, bir some noktasının her iki eksen üzerinde yer değiştirmesi sağlanacaktır. Başka bir deyişle, kırmızı çizginin some noktaları X ve Y ekseninde binlerce yer değiştireceklerdir. Bu yer değiştirmelerden oluşan yarma dolgu bölgelerinin alanları,

fazlasıyla karmaşık matematiksel işlemten sonra hesaplanacak ve aralarından en optimum miktara sahip olan kırmızı çizgi son olarak kabul edilecektir.

Yukarıda anlatılan yöntem uygulandığında tasarlanan kırmızı çizgide, hafriyat en az miktarda ve yarma-dolgu miktarı eşit olma olanağı vardır.

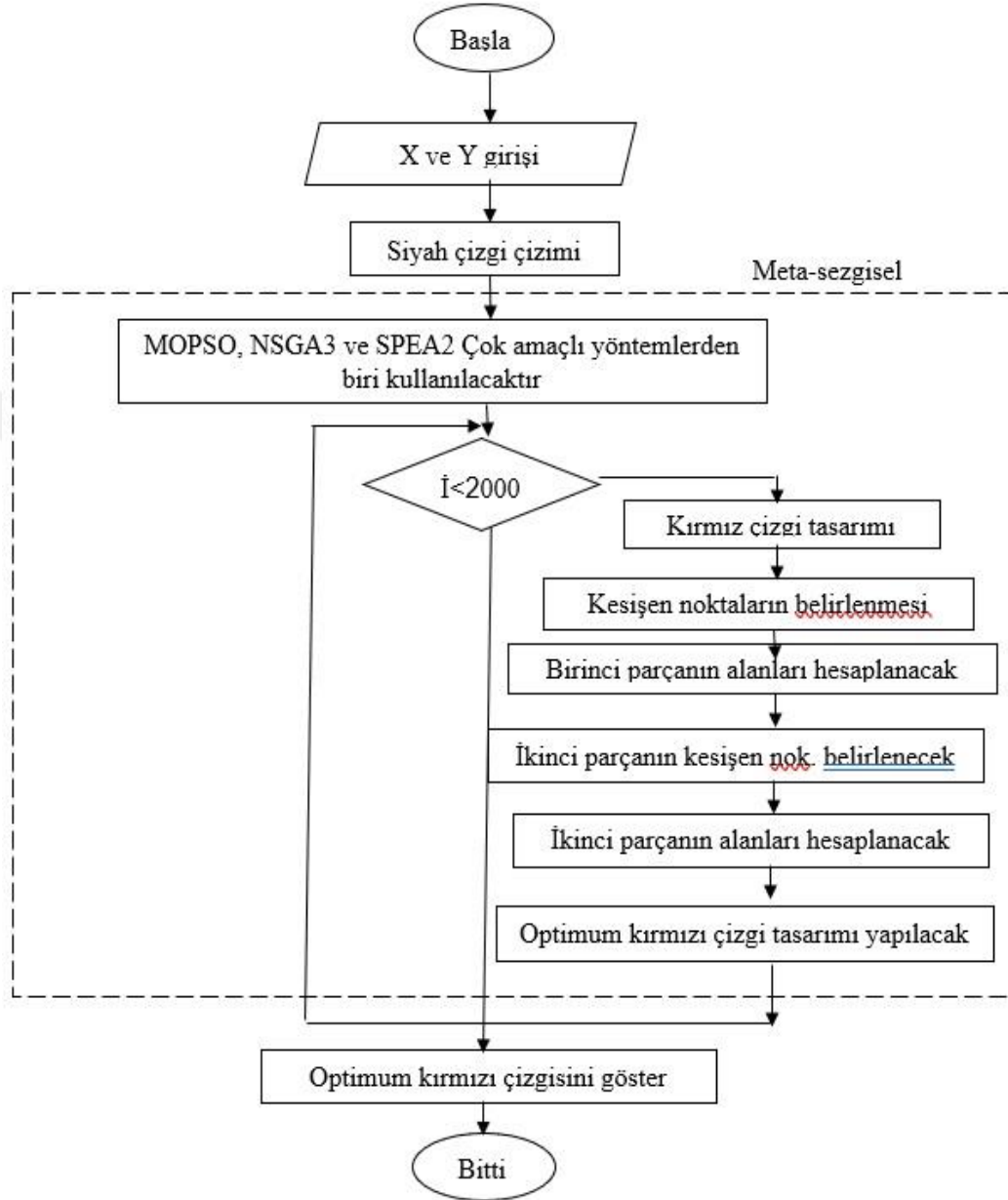
3.3.6. Kırmızı Çizgiye Ait Çok-Amaçlı Meta-Sezgisel Akış Şeması

Bu çalışma kapsamında, hafriyat miktarını en aza indirerek ve aynı zamanda yarma-dolgu miktarında eşitleyerek, maliyet miktarının en aza düşürülmesi sağlanacaktır. Bu bağlamda, önce Şekil 3.29'da gösterilmekte olan akış şemasına göre kırmızı çizgi hesaplama algoritması tasarlanacaktır. Daha sonra ise, bu algoritma sayesinde karayolu boykesit tasarımı yapıldığında dikkate alınan herhangi bir some noktasını binlerce değişken hale getirerek en optimum kırmızı çizgi tasarımı gerçekleştirilecektir. Böylece, kırmızı çizgi optimizasyonu ile ilgili en önemli adımlardan biri ortaya konulmuş ve optimum bir kırmızı çizgi yapılmasına olanak sağlayacak düzenlemenin altyapısı (akış şeması) gerçekleştirilmiştir.

Bu akış şemasında belirtildiği gibi önce araziden topladığımız eksen hattının koordinatı belirlenecektir. Daha sonra bu koordinatlara göre siyah çizgi geçirilecektir. Çok-amaçlı meta-sezgisel optimizasyon yöntemlerinden *MOPSO*, *NSGA3* ve *SPEA2* yöntemleri kullanılarak kırmızı çizgi optimizasyonu gerçekleştirilecektir. Bu bağlamda, önce some noktaları otomatik olarak yazılım tarafından rastgele seçilecek ve daha sonra bulunan some noktalarına X ve Y yönünde bir aralık verilerek some noktalarına yer değiştirme olanağı sağlanacaktır. Tabi ki yazılım tarafından bu yer değiştirme 2000 defa tekrar edilecek ve her defasında yarma-dolgu alanları hesaplanacaktır. Bu hesaplamalardan hangisi en az ve aynı zamanda eşit ise yazılım tarafından seçilecek ve nihai kırmızı çizgi olarak gösterilecektir.

Bu çalışma kapsamında, hafriyat miktarını en aza indirip ve aynı zamanda yarma-dolgu miktarını da eşitlemek mümkün olacaktır. Bu sayede maliyet miktarının en aza düşürülmesini sağlayabilmek amacıyla kırmızı çizgi algoritması tasarlanmıştır. Tasarlanan kırmızı çizgi hesaplama algoritmasına ilişkin doğru parçaların akış şeması Şekil 3.29'da verilmektedir. Daha sonra ise, bu algoritma sayesinde karayolu projesinin tüm verileri some noktalarına göre parçalara ayrılarak herhangi bir some noktasını binlerce değişik nokta arasından bulunabilmesinin en önemli adımlarından biri ortaya konulmuş ve böylece optimum bir kırmızı çizgi yapılmasına olanak

sağlayacak düzenlemenin altyapısı (akış şeması) gerçekleştirilmiştir. Bu algoritma, çok-amaçlı meta-sezgisel optimizasyon yöntemlerinden *MOPSO*, *NSGA3* ve *SPEA2* yöntemine entegre edilerek ve 30 kez deneme yapılarak mercek altına alınacaktır.

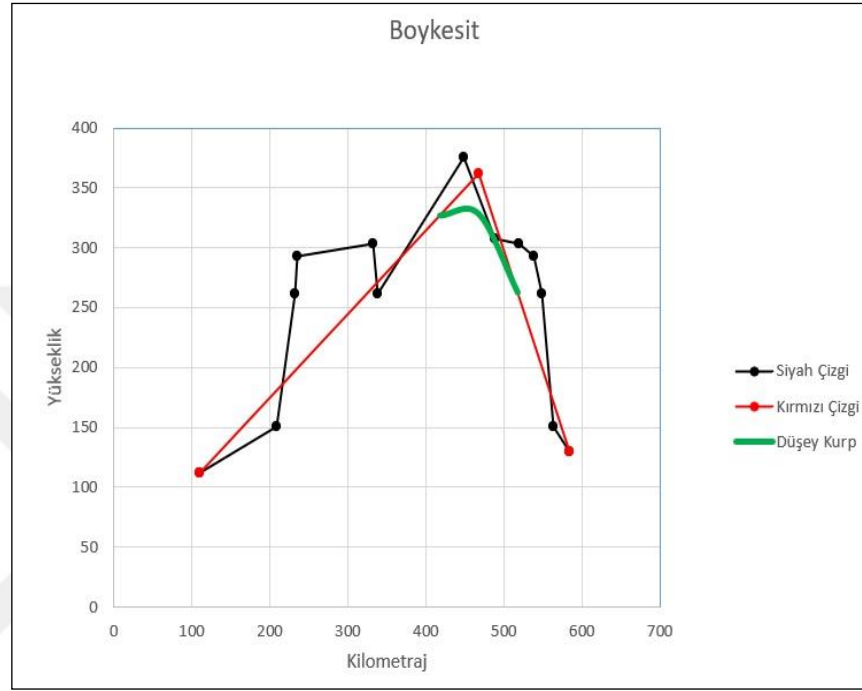


Şekil 3.29. Çok-amaçlı meta-sezgisel algoritmaya ilişkin akış şeması

Ancak, işlemlerin daha sağlıklı olmasından dolayı yukarıda anlatılan algoritma, yazılım tarafından 30 kez daha tekrarlanacak ve her defasında yazılım tarafından optimum kırmızı çizgi tasarlanacaktır. Dolayısıyla, bu 30 tane optimum kırmızı çizgi arasından, yarma-dolgu miktarı en az olan ve aynı zamanda neredeyse sıfıra yakın yarma-dolgu farkı gösteren kırmızı çizgi seçilecektir. Tabi ki bu işlemler her 3 meta-sezgisel optimizasyon yöntemi için de ayrı ayrı denenecektir.

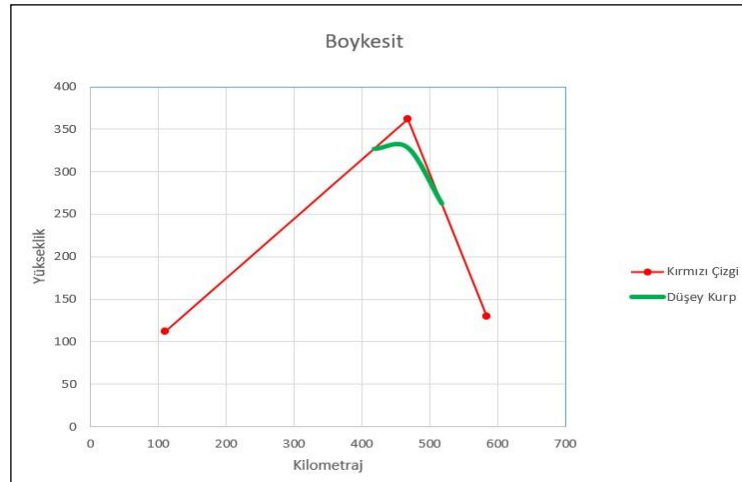
3.4. Kırmızı Çizgiye Ait Düşey Kurp Optimizasyonu

Yol projelerinde siyah çizgi üzerinden kırmızı çizginin doğru parçaları geçirilir ve bu doğru parçaların kesiştiği noktalarda yani some noktalarında düşey kurp yapılması gerekmektedir. Şekil 3.30'da gözüktüğü gibi siyah çizgiyi kesen doğru parçaların kesiştiği noktaya bir düşey kurp yerleştirilmektedir.



Şekil 3.30. Some noktasına düşey kurba yerleştirilmesi

Yukarıda gözükmekte olan kırmızı çizgi ve aynı zamanda some noktasında mevcut olan düşey kurba, daha net bir şekilde, Şekil 3.31'de gösterilmektedir.



Şekil 3.31. Some noktasına düşey kurp yerleştirilmesi

3.4.1. Düşey Kurp Kodlaması

Bu bağlamda, aynı düşey kurbayı karmaşık matematiksel işlemleri ile yazılıma entegre edebilmek için yüzlerce satır kodlama yazılmalıdır. Şekil 3.32’de bu kodlamaya ilişkin bir takım yazılım, sırasıyla gösterilmektedir. Örnek olarak yapılmış olan bu kodlamada önce kırmızı çizgiye farazi olarak bir koordinat verilmektedir. Daha sonra bu kırmızı çizgiye ait eğimler hesaplanmış olacaktır. Eğim hesabı yapıldıktan sonra düşey kurba uzunluğu tasarımcıdan sorularak yazılıma girilmiş olacaktır. Bu veriler ile düşey kurp hesaplama işlemleri, yazılım tarafından her ne kadarda karmaşık işlemlerden oluşsa bile çok hızlı bir şekilde gerçekleşmiş olacaktır.

```
xa=110,70
ya=111,80
xb=468,10
yb=361,40
xc=584,40
yc=130,20

x=[xa;xb;xc];
y=[ya;yb;yc];
plot(x,y)

g1=(yb-ya)/(xb-xa);
g2=(yc-yb)/(xc-xb);

G=abs(g1-g2);
prompt = 'What is the original value? ';
L = input(prompt);
T=L/2
E=(L*G)/8
KMT1=xb-T
KMT2=xb+T
KOTT1=yb-(g1*T)
KOTT2=yb+(g2*T)
KOTS=yb-E
xy=[KMT1 KOTT1;xb KOTS;KMT2 KOTT2];

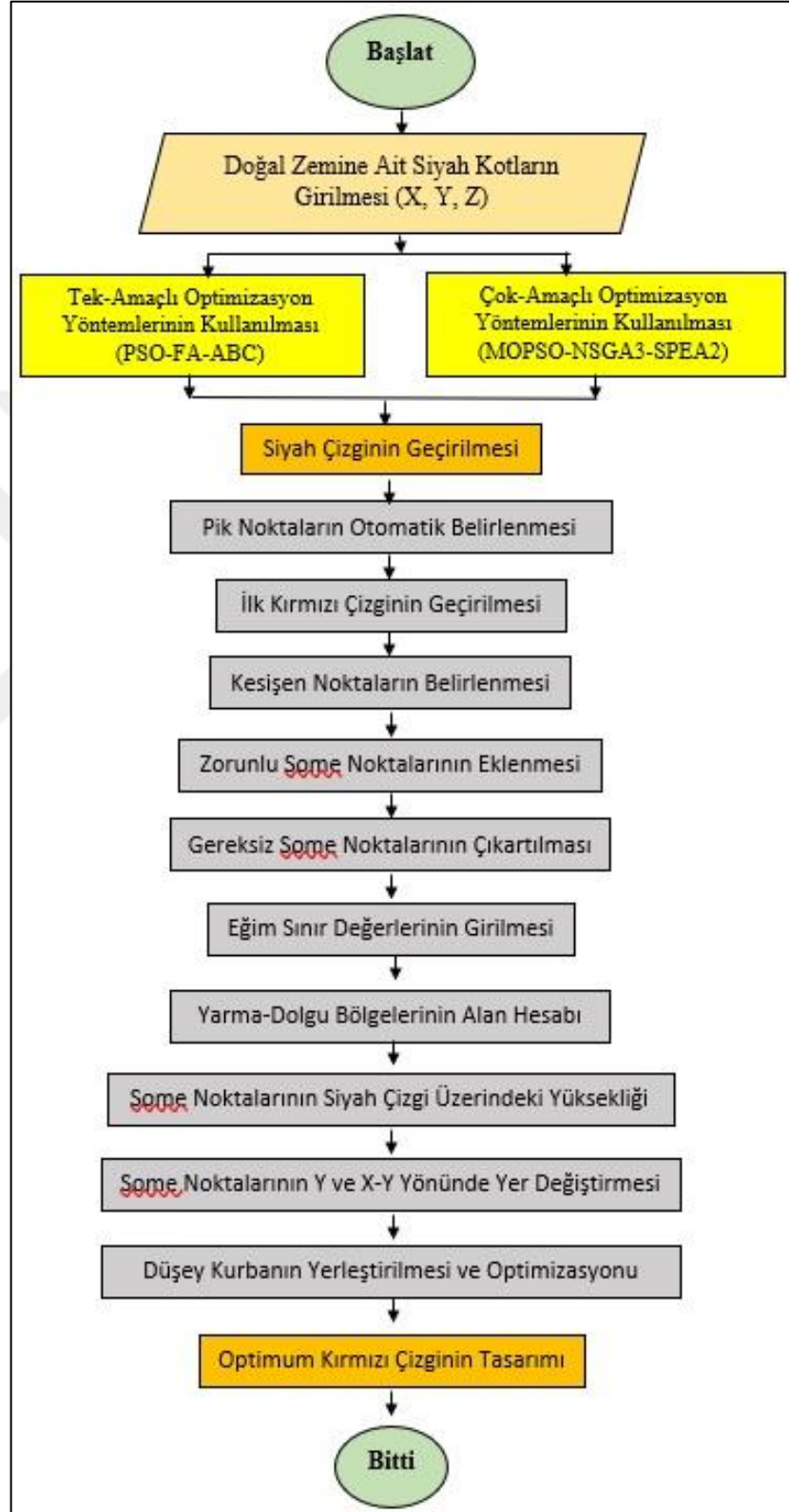
hold on
fplot(cscvn(xy),'r',2)
hold off
```

Şekil 3.32. Düşey kurp yazılım kodlaması

3.5. Kırmızı Çizgiye Ait Optimizasyonun Akış Şeması

Çalışma kapsamında tarafımızdan hazırlanan kırmızı çizgiye ait yazılım kodlaması, meta sezgisel optimizasyon algoritmalarından tek amaçlı olan PSO, FA ve ABC yöntemlerine ve aynı zamanda çok amaçlı optimizasyon algoritmalarından NSGA3, MOPSO ve SPEA2 yöntemlerine entegre edilerek denetimi yapılmıştır. Bu algoritmalar ile denetimi gereken noktalara ilişkin verdiği sonuçlar çok kabul edici

düzeyde bulunmuştur. Hazırlanan bu algoritma ile ilgili akış şeması aşağıdaki gözükmemektedir. Bu çalışmada kullanılan Matlab tabanlı yazılımın esas teşkil eden akış şeması Şekil 3.33'te verilmiştir.



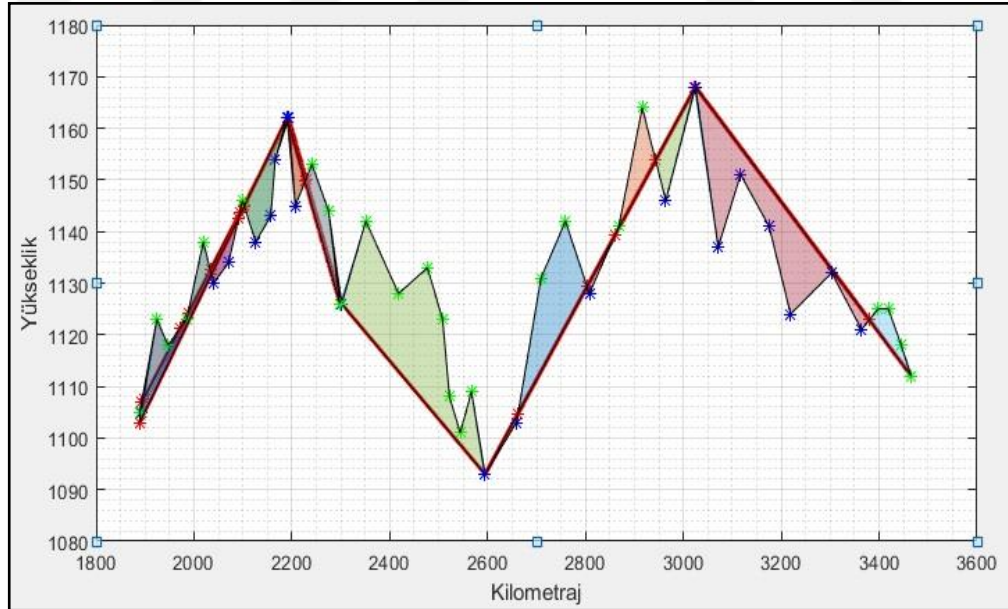
Şekil 3.33. Kırmızı çizgiye ait optimizasyon akış şeması

4. BULGULAR VE TARTIŞMA

4.1. Tek-Amaçlı Optimizasyon Algoritmaları ile İlgili Bulgular

Günümüzde kullanılan karayolu tasarım programları, tasarım konusunda çok profesyonel programlar olmalarına rağmen, kırmızı çizgi tasarımı ile ilgili optimum bir kırmızı çizgi tasarımı için bir yöntem sunmamaktadırlar. Bu çalışma kapsamında, tasarlanan algoritma sayesinde optimum kırmızı çizgi tasarımına ilişkin ön adımlar gerçekleştirilmiştir.

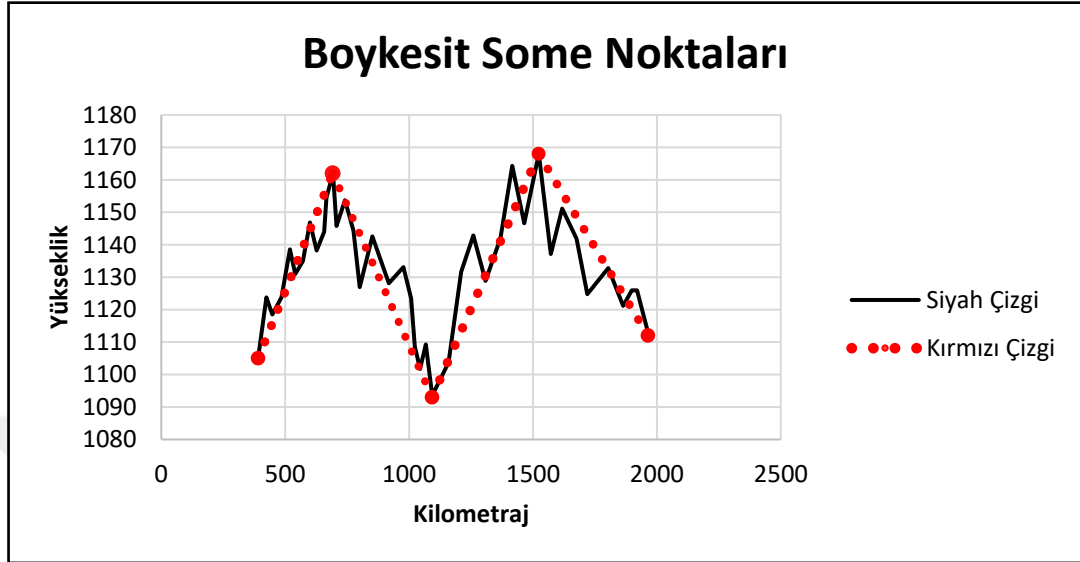
Bu çalışma kapsamında, örnek olarak bir karayolu projesinin bir kısmı baz alınmaktadır. Tasarlanan optimizasyon yazılımı ile elde edilen tüm detaylar adım adım aşağıda açıklanmaktadır. Örnek olarak dikkate alınan bu karayolu projesi tamamen rastgele mesafelerden ve yükseklik kotlarından oluşmaktadır. Yani başlangıç, bitiş ve ayrıca bir tepe düşey kurp some noktası bulunan bu projede sadece iki doğru parça yer almaktadır. Şekil 4.1'de kilometraj olarak daha uzun bir karayolu projesine ait siyah çizgi ve aynı zamanda kırmızı çizgi gösterilmektedir. Ayrıca, bu çizgiler arasında oluşan yarma-dolgu alanları da net bir şekilde gösterilmektedir. X eksenini mesafeyi ve Y eksenini yükseklik kotlarını göstermektedir.



Şekil 4.1. Doğru parçaların optimizasyonu

Örnek proje üzerinde kırmızı çizginin some noktası, yazılım tarafından otomatik olarak seçilmektedir. Sonuç olarak some noktası hazırlanan yazılım sayesinde, X ve Y yönünde binlerce yer değiştirme kabiliyeti olacaktır. Başka bir deyişle, bu yazılımı

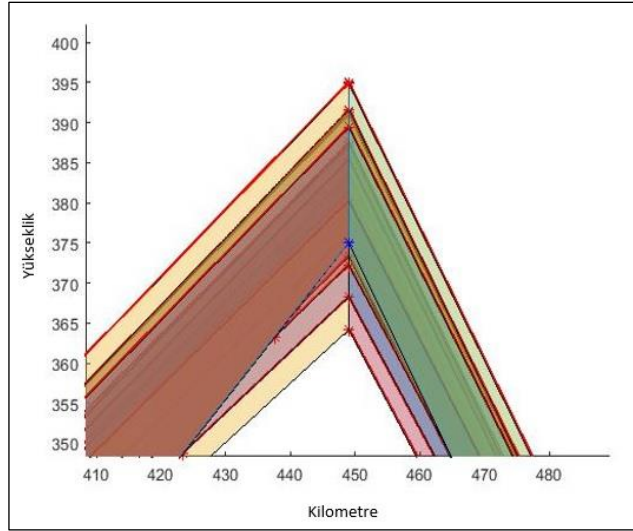
kullanarak some noktalarına binlerce yer deęiřtirmelerine imkân saęlanacaktır. Buna baęlı olarak, some noktasının seildięi yerin ok da nemli olmadığı yadsınamaz bir hale gelecektir. řekil 4.2'de some noktasının seildięi yer gsterilmiřtir.



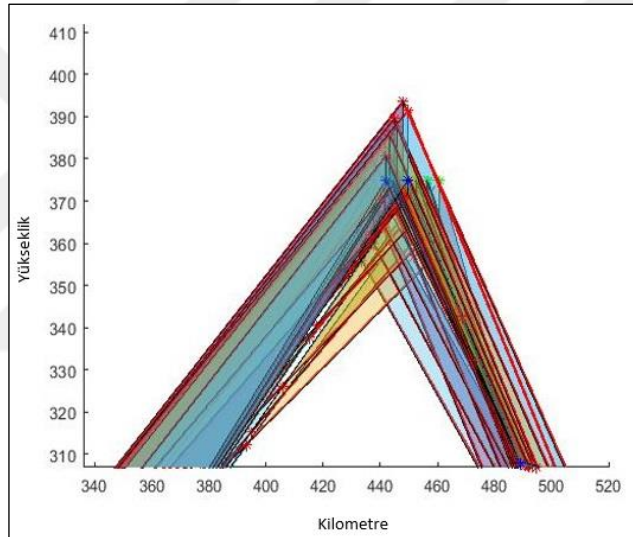
řekil 4.2. Some noktasının belirlenmesi

Tasarlanan algoritma kodlaması, 3 farklı tek-amalı meta-sezgisel optimizasyon algoritması ile denenerek sonuçları kaydedilmektedir. Kullanılan yöntemler (*PSO-FA-ABC*) literatürde akış diyagramları ile gsterilmektedir. Bu alıřma kapsamında, hazırlanan yazılım, yukarıdaki yöntemler ile entegre edilerek mercek altına alınmıřtır. Bu inceleme iki farklı bakış aısı ile karřılařtırılmaktadır. Bařka bir deyiřle, some noktasını nce Y eksenini zerinde ve daha sonra X ve Y eksenini zerinde bir matris oluřturarak yer deęiřtirmesine imkân verilecektir.

Bu baęlamda, Y eksenini zerinde yapılan uygulama řekil 4.3'te ve X-Y eksenini zerinde yapılan uygulama ise řekil 3.4'te ayrı ayrı inceleme yapılarak gsterilmektedir. Grldęü gibi Y eksenini zerinde yer deęiřtirmeler sadece o eksen zerinde belli bir mesafe arasında sadece yukarı ve ařaęı ynde yapılmaktadır. Ancak, X ve Y eksenini zerinde yer deęiřtirmeler ise her drt ynde rastgele yer deęiřtirme yapabilmektedir.



Şekil 4.3. Y eksenli açılımı

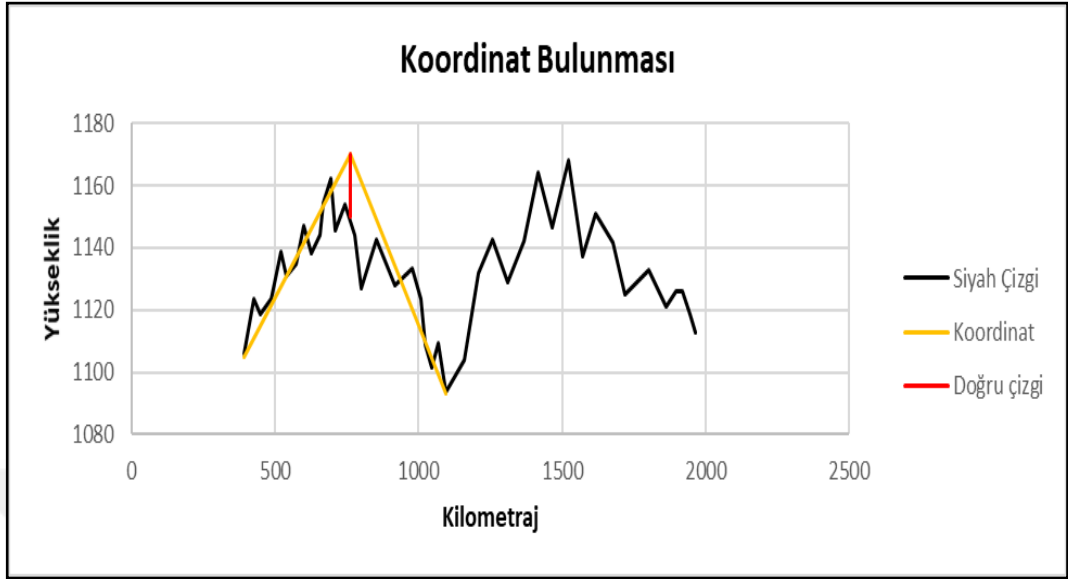


Şekil 4.4. X-Y eksenli açılımı

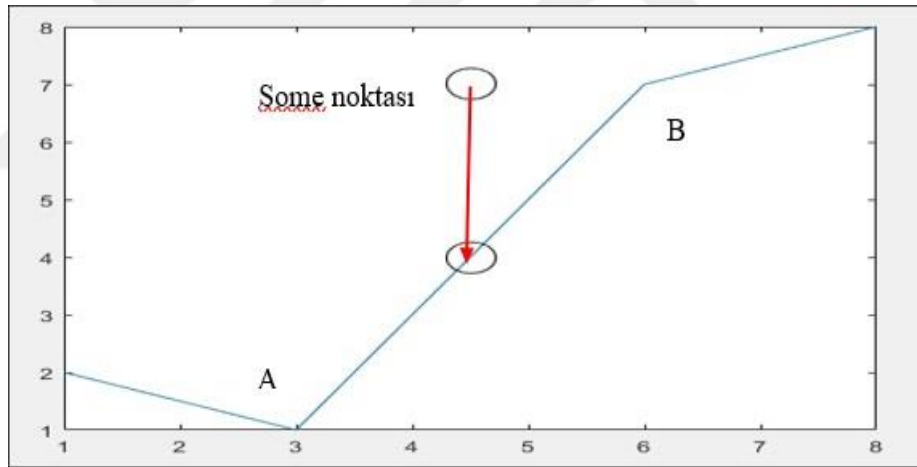
4.1.1. Some Noktalarının Siyah Çizgi Üzerindeki Koordinatı

Boykesit tasarımında some noktaları tasarlanan optimizasyon yazılımı sayesinde Şekil 4.5'te görüldüğü gibi hem X hem de Y eksenli üzerinde binlerce yer değiştirmektedirler. Bu bağlamda, some noktalarının her yer değişiminde kırmızı çizgi ile siyah çizgi arasında oluşan kapalı yarım-dolgu bölgelerinin alanları hesaplanacaktır. Başka bir deyişle, kırmızı çizginin her doğru parçasının başlangıç ve bitiş noktası siyah çizgiye bağlanarak bu kapalı alanların oluşumu için zaruri bir şarttır. Some noktaları yazılım sayesinde devamlı yer değiştirmelerine göre, bu noktaların siyah çizgi üzerinde, farklı yerlere denk gelecekleri yadsınamaz bir gerçek haline gelmektedir. Bu durumu özetleyeceksek Şekil 4.6'da gösterildiği gibi her some noktası

için farklı siyah kot bulunmaktadır. Bu bağlamda yol boyunca farklı some noktaları için siyah kotun bulunmasında Formül 4.1'den yardım alınacaktır.



Şekil 4.5. Doğru çizginin belirlenmesi



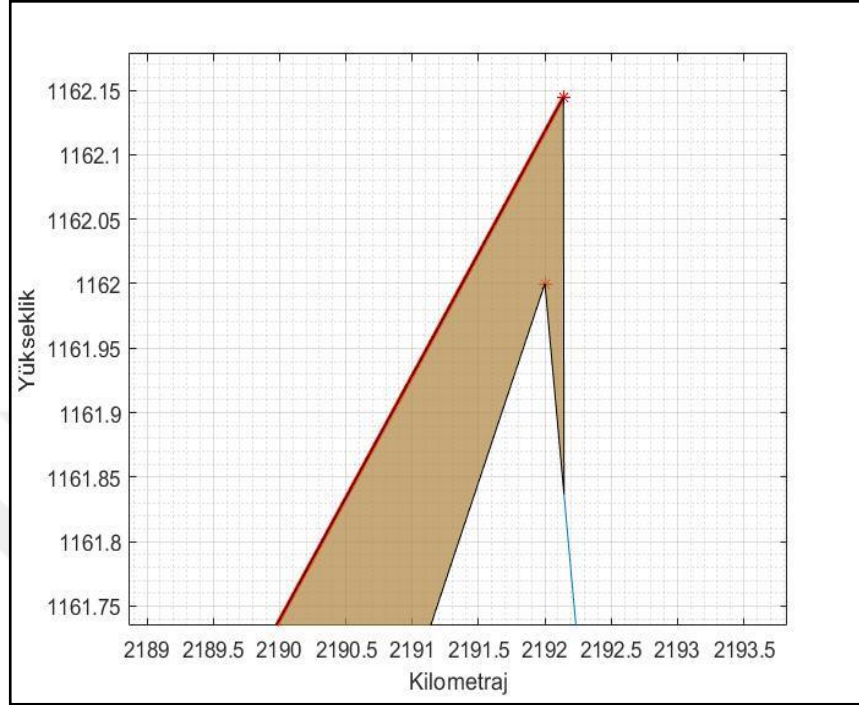
Şekil 4.6. Siyah kotun bulunması

A (a, b) ve B (c, d) noktalarının koordinatları arazi kotları olduğundan proje başından bilinmektedir. Some noktasının siyah çizgi üzerinde nereye denk geleceği yani X ve Y koordinatları aşağıdaki Formül 4.1'den bulunmaktadır. Tabi ki bu aşamada X koordinatı boykesit üzerinde kilometrajı göstermektedir. Başka bir deyişle, X koordinatı her some noktası için siyah çizgi ve kırmızı çizgi için aynıdır.

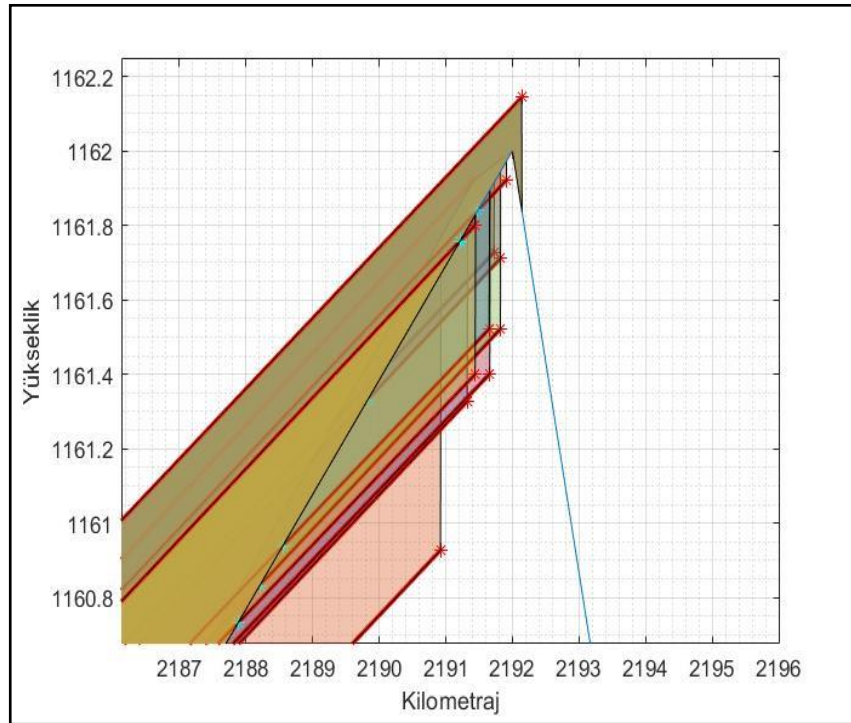
$$Y = \frac{X(d-b) - ad + cb}{c-a} \quad (4.1)$$

Yukarıdaki formül sayesinde, yazılımın bulduğu herhangi bir some noktasının siyah çizgi yani zemin üzerinde tam olarak dik kestiği noktanın koordinatı

hesaplanacaktır. Şekil 4.7’de tek bir some noktasının siyah çizgiye dik durumu ve Şekil 4.8’de optimizasyon sayesinde onlarca some noktasının zemin üzerindeki yükseklik kotu gösterilmiştir.



Şekil 4.7. Siyah kotun bulunması



Şekil 4.8. Some noktalarının siyah kotları

4.1.2 Y Eksenine Üzerinde Tek-Amaçlı Meta-Sezgisel Algoritmaların Uygulanması

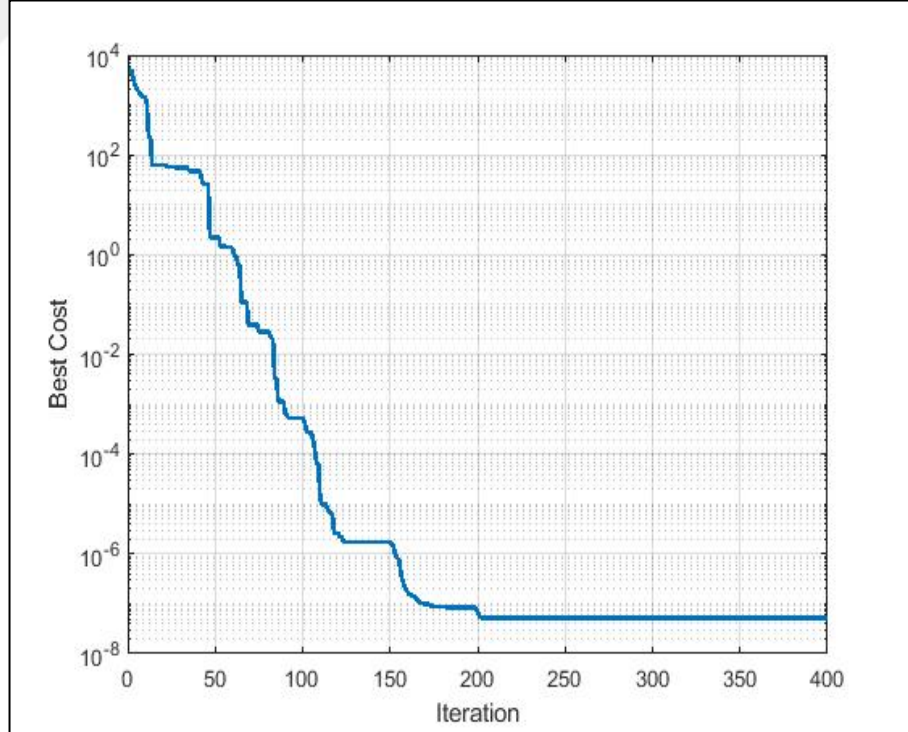
Örnek bir karayolu projesine ait tasarlanan kırmızı çizginin some noktaları sadece Y eksenine üzerinin yukarı ve aşağısına doğru bir aralık verilerek, üç meta-sezgisel yöntem (*PSO-FA-ABC*) ile denetimi yapılmaktadır. Bu bağlamda, optimizasyon yöntemleri uygulanan örnek projenin sonuçları aşağıdaki Tablo 4.1'de gösterilmektedir. Bu tabloda gözüktüğü gibi örnek bir karayolu projesi üzerinde uygulanmış kırmızı çizgi optimizasyon sonuçları daha gerçekçi ve güvenilir olması gerektiğinden bu deneme işlemi 30 kez yapılmıştır. Bu defalarca deneme sonucuna göre *PSO* yöntemi ortalama yarma-dolgu farkını neredeyse sıfır denilecek kadar bir değer bulana kadar sürdürmüştür. Bu fark *FA* yönteminde ortalama olarak 0.075 ve *ABC* yönteminde ise bu fark ortalama olarak 80 birim olarak gözükmektedir.

Tablo 4-1. Y eksenine için tek-amaçlı meta-sezgisel optimizasyon sonuçları

No	ABC	FA	PSO
1	20,67176	0,075076	1,81899E-12
2	80,19236	0,082699	4,54747E-12
3	29,33940	0,006993	9,61336E-10
4	68,03176	0,078852	4,13002E-05
5	217,61493	0,124834	1,23978E-07
6	39,77752	0,184907	2,72848E-12
7	84,80049	0,067062	2,72848E-12
8	164,88047	0,047276	1,00044E-11
9	8,84874	0,065172	3,54794E-09
10	67,84740	0,188054	1,72804E-11
11	61,93330	0,082672	6,36646E-12
12	60,45301	0,093912	9,09495E-13
13	75,75453	0,089407	7,27596E-12
14	83,71918	0,025089	1,09139E-11
15	10,02935	0,034653	4,93469E-06
16	10,01355	0,037043	2,72848E-12
17	62,79250	0,074691	1,91494E-08
18	41,26310	0,207178	2,91038E-11
19	46,82386	0,053098	1,81899E-12
20	138,25034	0,029284	5,78439E-10
21	87,15463	0,110539	1,81899E-12
22	170,36249	0,034956	1,19039E-12
23	169,29690	0,116562	8,18545E-12
24	96,76463	0,140001	1,90796E-05
25	40,70000	0,031051	6,57201E-09
26	70,78025	0,055689	2,27374E-11
27	154,19159	0,022818	1,90994E-11
28	142,42456	0,060135	1,81899E-12
29	142,67609	0,039112	2,72848E-12
30	118,89080	0,082078	7,27763E-06

Tablo 4.1'de gözüktüğü gibi tüm veriler her üç yöntemle ayı ayrı 2000 defa ve 30 kez deneme yapılarak elde edilmiştir. Bu sonuçlar kapsamında elde edilen diyagramlardan sadece örnek bir tanesinin iterasyon ve maliyet ilişkisine dikkat çekme adına şekil üzerinde gösterilmektedir. *PSO* yöntemi için Şekil 4.9. ve *FA* yöntemi için Şekil 4.10. ve son olarak *ABC* yöntemi için Şekil 4.11'de gösterilmektedir. Bu diyagramların X eksenini iterasyonu ve Y eksenini en iyi sonucu göstermektedir. Tabi ki bu çalışmaya, yarma-dolgu miktarlarını en aza düşürmek ve aynı zamanda aralarındaki farkı sıfıra yakın bir değer bulmak için girilmiştir.

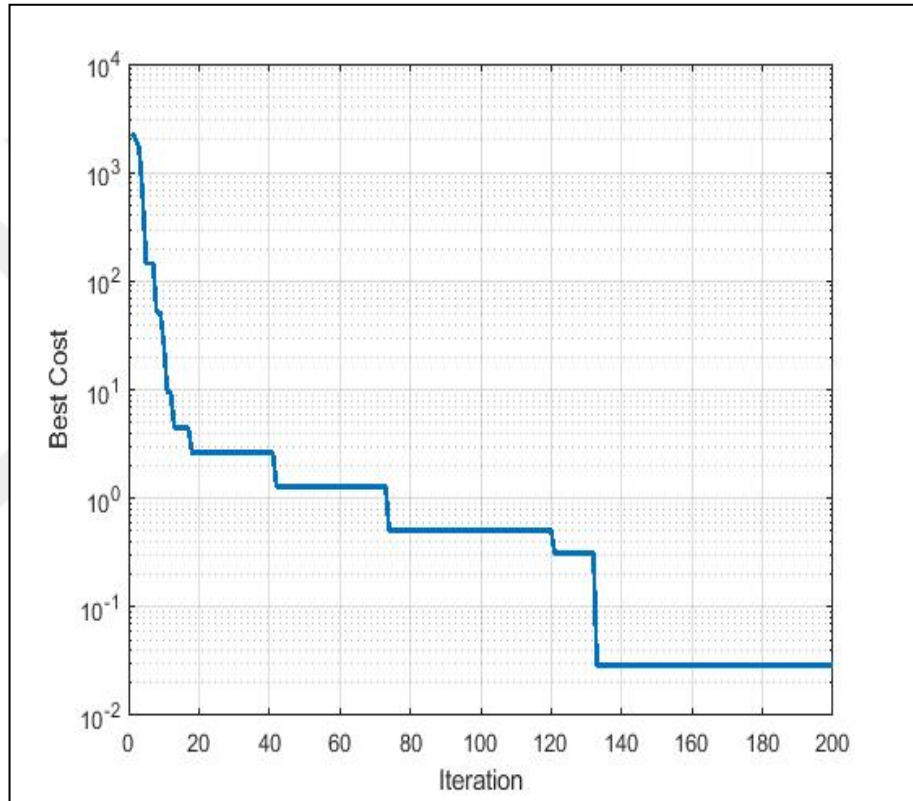
Şekil 4.9'da gözüktüğü gibi *PSO* yöntemi ile optimizasyon yapıldığında, birinci kırmızı çizgi geçirildiği zaman yarma-dolgu miktarları arasındaki fark çok fazla ve yaklaşık 10000 birim iken 200'üncü iterasyondan itibaren bu fark neredeyse sıfıra yakın bir değer bulunmaktadır. Bu yöntem ile tasarlanan kırmızı çizgi diyagrama göre çok hızlı ve etkin bir şekilde sonuçlar elde etmemizi sağlamaktadır. Bu optimizasyon yöntemine göre diyagramda gözüktüğü gibi bulunan her kırmızı çizgide yarma-dolgu farkı bir öncekine göre daha azdır. Başka bir deyişle bulunan her kırmızı çizgi bir öncekine göre daha ekonomiktir.



Şekil 4.9. *PSO* yöntemi Y ekseninde

Şekil 4.10'da gözüktüğü gibi *FA* yöntemi ile optimizasyon yapıldığında, birinci kırmızı çizgi geçirildiği zaman yarma-dolgu miktarları arasındaki fark *PSO*

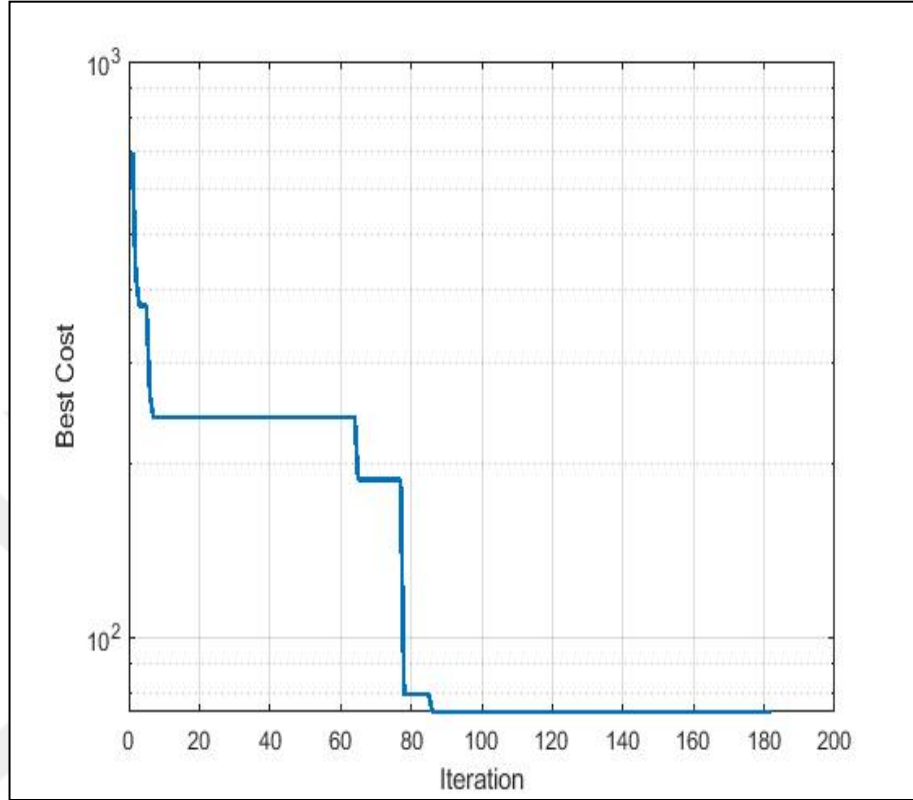
yönteminin ilk denemesi kadarda fazla olmasa bile yaklaşık 1000 birim olarak bulunmaktadır. Ancak, 135'inci iterasyondan itibaren bu fark ortalama olarak 0.1 birime yakın bir değere kadar düşmektedir. Bu yöntem ile tasarlanan kırmızı çizgi diyagramına göre çok hızlı ve etkin bir şekilde sonuçlar elde etmemiz sağlanmaktadır. Bu optimizasyon yöntemine göre diyagramda gözüktüğü gibi bulunan her kırmızı çizgide yarma-dolgu farkı bir öncekine göre daha az ve hatta büyük bir kısımda tekrarlanarak aynı değerleri bulmaktadır. Başka bir deyişle bulunan her kırmızı çizgi bir öncekine göre daha ekonomiktir.



Şekil 4.10. FA yöntemi Y ekseninde

Şekil 4.11'de gözüktüğü gibi ABC yöntemi ile optimizasyon yapıldığında, birinci kırmızı çizgi geçirildiği zaman yarma-dolgu miktarları arasındaki fark diğer iki yönteminin ilk denemesi kadarda fazla olmasa bile 1000 birimin altında bulunmaktadır. Ancak, 80'inci iterasyondan itibaren bu fark ortalama 80 birime yakın bir değere kadar düşmektedir. Bu yöntem ile tasarlanan kırmızı çizgi diyagrama göre çok hızlı ve etkin bir şekilde sonuçlar elde etmemizi sağlamaktadır. Bu optimizasyon yöntemine göre diyagramda gözüktüğü gibi bulunan her kırmızı çizgide yarma-dolgu farkı bir öncekine göre daha az ve hatta büyük bir kısımda tekrarlanarak aynı değerleri bulmaktadır. Başka bir deyişle bulunan her kırmızı çizgi bir öncekine göre daha

ekonomiktir. Ancak, bu yöntemin diyagramında görüldüğü gibi her ne kadarda en son bulunan kırmızı çizgi değerleri sıfırdan fazla olsa da ama diğer iki yöneme göre daha hızlı sabit bir değere ulaşmaktadır.



Şekil 4.11. ABC yöntemi Y ekseninde

4.1.3. X-Y Ekseninde Tek-Amaçlı Meta-Sezgisel Algorimaların Uygulanması

Örnek bir karayolu projesine ait tasarlanan kırmızı çizginin some noktaları her iki eksen üzerinde yani X ve Y eksenlerinin yukarı ve aşağısı aynı zamanda sağ ve sol taraflarında bir aralık verilerek, üç meta-sezgisel yöntem (*PSO-FA-ABC*) ile denetimi yapılmaktadır. Bu bağlamda, optimizasyon yöntemleri uygulanan örnek projenin sonuçları aşağıdaki Tablo 4.2'de gösterilmektedir. Bu tabloda gözüktüğü gibi örnek bir karayolu projesi üzerinde uygulanmış kırmızı çizgi optimizasyon sonuçları daha gerçekçi ve güvenilir olması gerektiğinden bu deneme işlemi 30 kez yapılmıştır. Bu defalarca deneme sonucuna göre *PSO* yöntemi ortalama yarma-dolgu farkını neredeyse sıfır denilecek kadar bir değer bulana kadar sürdürmüştür. Bu fark *FA* yönteminde ortalama olarak 0.091 ve *ABC* yönteminde ise bu fark ortalama olarak 88 birim olarak gözükmektedir.

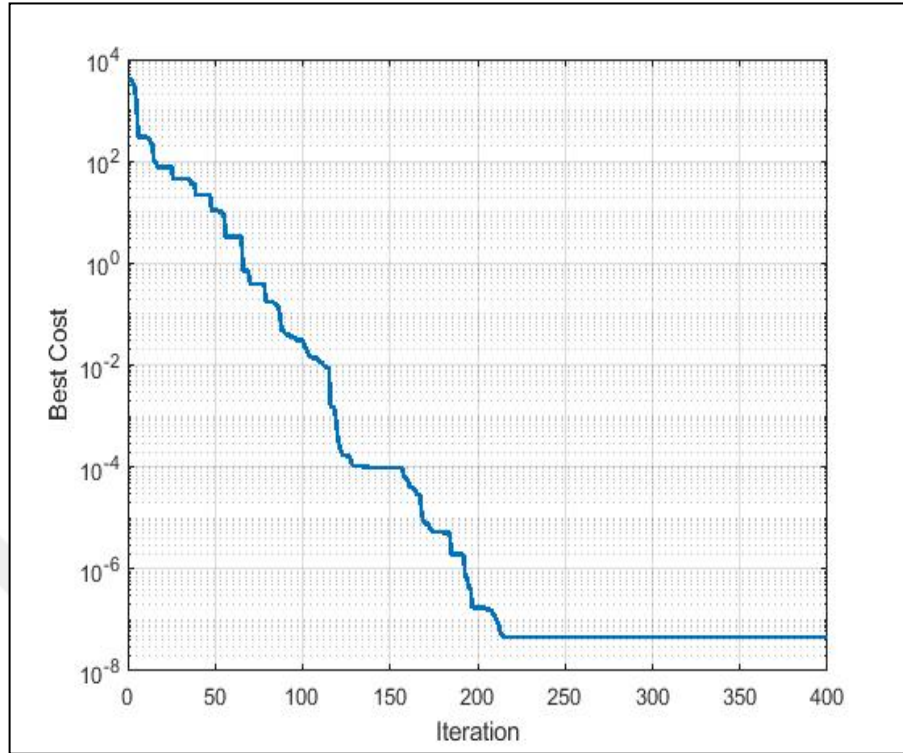
Tablo 4-2. X-Y eksenini için optimizasyon sonuçları

No	ABC	FA	PSO
1	100,47288	0,137311504	4,56229E-08
2	100,94008	0,131411504	7,18501E-11
3	52,96572	0,063822737	8,54016E-10
4	125,07930	0,089033701	4,98488E-07
5	130,20926	0,011212183	1,01707E-05
6	46,14494	0,071324928	3,63798E-12
7	60,64144	0,063426511	9,44761E-07
8	74,35408	0,034137526	8,76947E-06
9	109,45475	0,139822237	9,09495E-13
10	53,17350	0,116618169	1,63709E-11
11	115,26616	0,039844997	1,96645E-06
12	45,18449	0,096529908	9,12542E-07
13	95,96233	0,032811239	1,36424E-12
14	144,02244	0,147914396	9,09495E-12
15	181,69628	0,075952284	1,00044E-11
16	51,45774	0,068572561	1,19473E-05
17	93,19141	0,131428807	3,37183E-05
18	5,83415	0,120685159	9,09495E-13
19	39,86531	0,037289308	4,00423E-06
20	54,09034	0,083760008	5,79348E-10
21	26,82585	0,112385389	4,78394E-10
22	136,58899	0,10716315	1,14814E-05
23	44,11678	0,208777256	8,0945E-10
24	145,15820	0,223801018	6,98583E-09
25	109,98181	0,081351008	8,66805E-05
26	97,37403	0,065885935	1,61435E-11
27	127,51752	0,096527621	7,78703E-05
28	31,72404	0,022312261	2,73876E-08
29	115,01165	0,111732998	2,81943E-11
30	110,57428	0,119187257	4,42325E-05

Tablo 4.2'de gözüktüğü gibi tüm veriler her üç yöntemle ayı ayrı 2000 defa ve 30 kez deneme yapılarak elde edilmiştir. Bu sonuçlar kapsamında elde edilen diyagramlardan sadece örnek bir tanesinin iterasyon ve maliyet ilişkisine dikkat çekme adına şekil üzerinde gösterilmektedir. PSO yöntemi için Şekil 4.12 ve FA yöntemi için Şekil 4.13 ve son olarak ABC yöntemi için Şekil 4.14'te gösterilmektedir. Bu diyagramların X eksenini iterasyonu ve Y eksenini en iyi sonucu göstermektedir. Tabi ki bu çalışmaya, yarma-dolgu miktarlarını en aza düşürmek ve aynı zamanda aralarındaki farkı sıfıra yakın bir değer bulmak için girilmiştir.

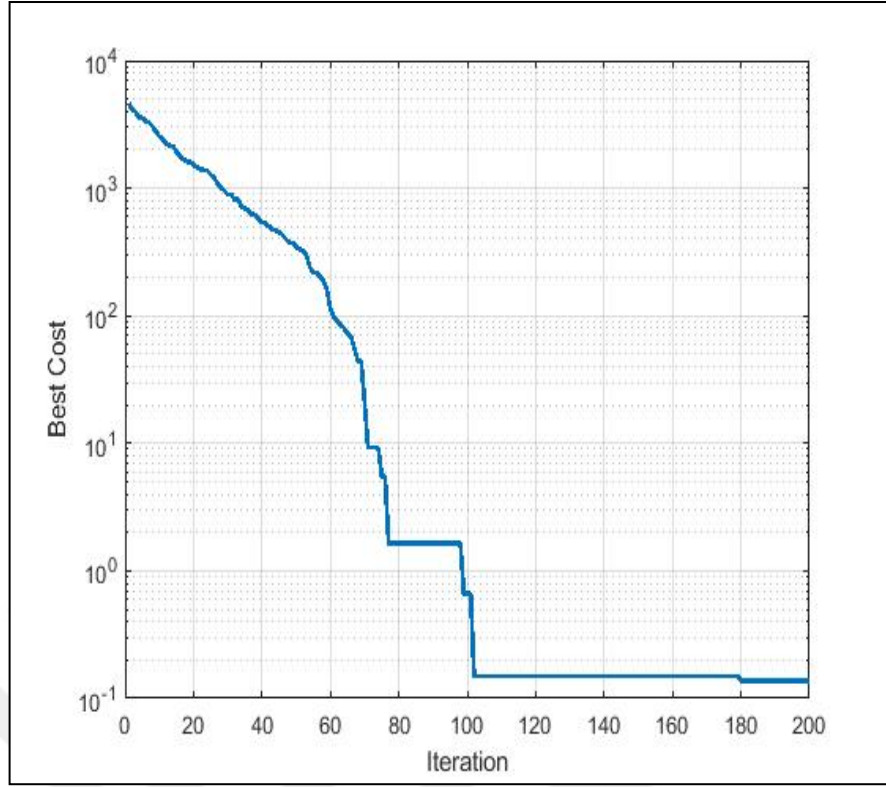
Şekil 4.12'de gözüktüğü gibi PSO yöntemi ile optimizasyon yapıldığında, birinci kırmızı çizgi geçirildiği zaman yarma-dolgu miktarları arasındaki fark çok fazla ve yaklaşık 10000 birim iken 215'inci iterasyondan itibaren bu fark neredeyse sıfıra yakın bir değer bulunmaktadır. Bu yöntem ile tasarlanan kırmızı çizgi diyagrama göre çok hızlı ve etkin bir şekilde sonuçlar elde etmemizi sağlamaktadır. Bu optimizasyon yöntemine göre diyagramda da gözüktüğü gibi bulunan her kırmızı çizgide yarma-

dolgu farkı bir öncekine göre daha azdır. Başka bir deyişle bulunan her kırmızı çizgi bir öncekine göre daha ekonomiktir.



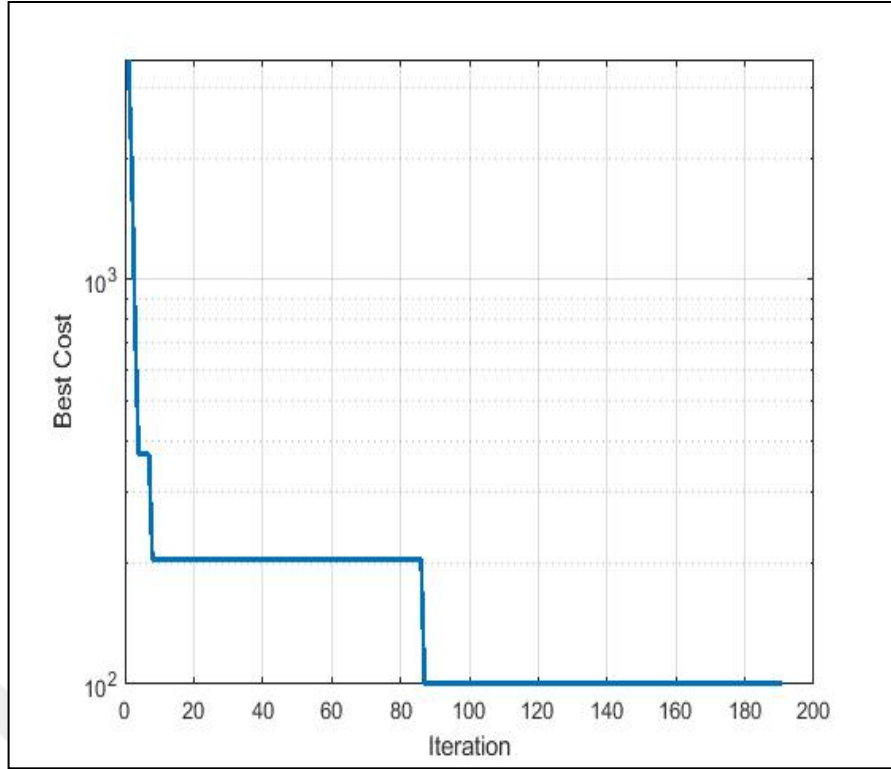
Şekil 4.12. PSO yöntemi X-Y ekseninde

Şekil 4.13'te gözüktüğü gibi FA yöntemi ile optimizasyon yapıldığında, birinci kırmızı çizgi geçirildiği zaman yarma-dolgu miktarları arasındaki fark PSO yönteminin ilk denemesi kadar fazla olmasa bile yaklaşık 8000 birim olarak bulunmaktadır. Ancak, 105'inci iterasyondan itibaren bu fark ortalama olarak 0.09 birime yakın bir değere kadar düşmektedir. Bu yöntem ile tasarlanan kırmızı çizgi diyagrama göre çok hızlı ve etkin bir şekilde sonuçlar elde etmemizi sağlamaktadır. Bu optimizasyon yöntemine göre diyagramda gözüktüğü gibi bulunan her kırmızı çizgide yarma-dolgu farkı bir öncekine göre daha az ve hatta büyük bir kısımda tekrarlanarak aynı değerleri bulmaktadır. Başka bir deyişle bulunan her kırmızı çizgi bir öncekine göre daha ekonomiktir.



Şekil 4.13. FA yöntemi X-Y ekseninde

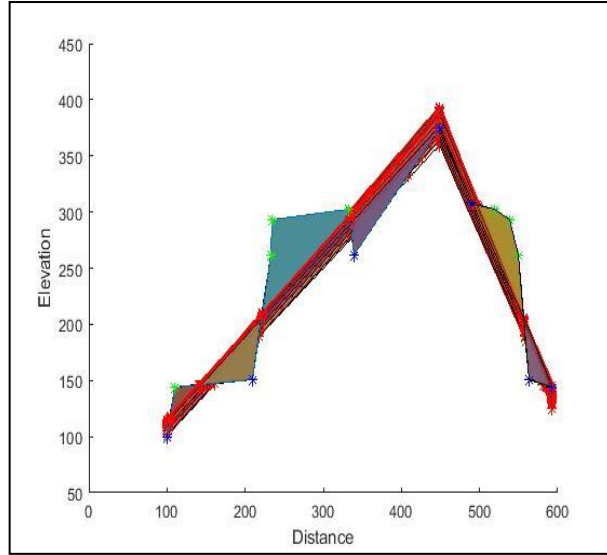
Şekil 4.14’te gözüktüğü gibi ABC yöntemi ile optimizasyon yapıldığında, birinci kırmızı çizgi geçirildiği zaman yarma-dolgu miktarları arasındaki fark diğer iki yönteminin ilk denemesi kadarda fazla olmasa bile 5000 birimin altında bulunmaktadır. Ancak, 85’inci iterasyondan itibaren bu fark ortalama 98 birime yakın bir değere kadar düşmektedir. Bu yöntem ile tasarlanan kırmızı çizgi diyagrama göre çok hızlı ve etkin bir şekilde sonuçlar elde etmemizi sağlamaktadır. Bu optimizasyon yöntemine göre diyagramda gözüktüğü gibi bulunan her kırmızı çizgide yarma-dolgu farkı bir öncekine göre daha az ve hatta büyük bir kısımda tekrarlanarak aynı değerleri bulmaktadır. Başka bir deyişle bulunan her kırmızı çizgi bir öncekine göre daha ekonomiktir. Ancak, bu yöntemin diyagramında görüldüğü gibi her ne kadarda en son bulunan kırmızı çizgi değerleri sıfırdan fazla olsa da ama diğer iki yöneme göre daha hızlı ve hatta 10’uncü deneme yapıldığında bile yarma-dolgu farkını düşürerek sabit bir değere ulaşmaktadır. Başka bir deyişle, bu yöneme göre her ne kadar da en son bulunan kırmızı çizgi ekonomik olmasa da elde etmek istediğimiz kırmızı çizgiye daha hızlı ve erken ulaşmamız gözlenmektedir.



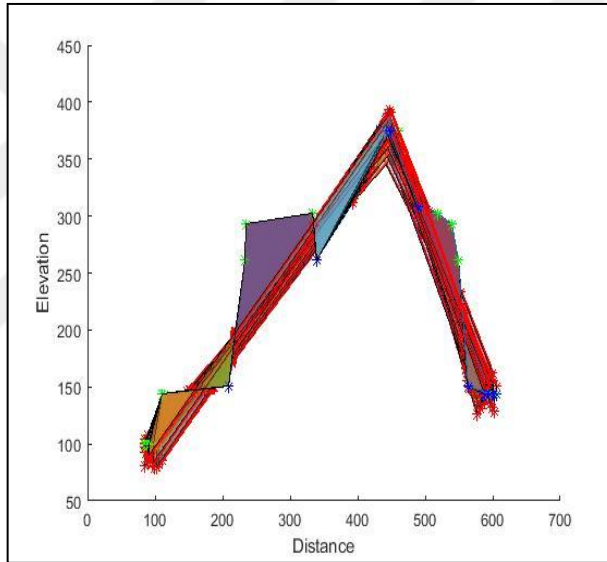
Şekil 4.14. ABC yöntemi X-Y ekseninde

4.1.4. Geliştirilmiş Tek-Amaçlı Meta-Sezgisel Algoritma Çıktıları

Bu çalışma kapsamında üç farklı meta-sezgisel optimizasyon algoritması kullanılmıştır. Bu algoritmalar sırasıyla PSO-FA-ABC adı verilen en popüler optimizasyon yöntemleridir. Dolayısıyla bu adı geçen yöntemlere entegre edilmiş optimum kırmızı çizgi yazılımı sayesinde en ekonomik kırmızı çizgi tasarımı gerçekleştirilmektedir. Geliştirilmiş olan bu algoritmalar hem Y ekseninde hem de X-Y ekseninde ayrı ayrı inceleme yapmaktadır. Tabi ki yukarıda anlatıldığı gibi her yöntem 2000 iterasyonla yapılmasına rağmen daha güvenilir ve gerçekçi sonuçlar elde etmemiz için bu işlem 30 kez aynı yöntemle tekrarlanmaktadır. Başka bir deyişle, her yöntem 30 kez denenmiş ve her bir deneme içinde 2000 iterasyonla tekrarlanmıştır. Bu bağlamda, Y ekseninde yapılan denemeler Şekil 4.15'te ve X-Y ekseninde ise Şekil 4.16'da gösterilmektedir.



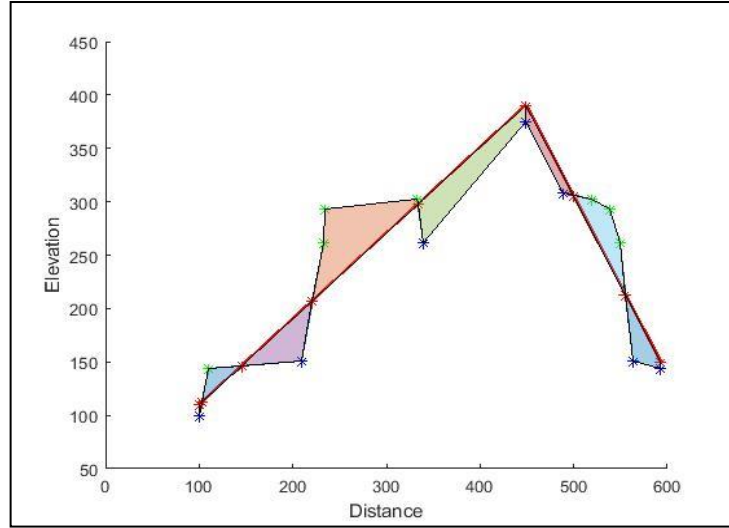
Şekil 4.15. Y eksenini için yapılan denemeler



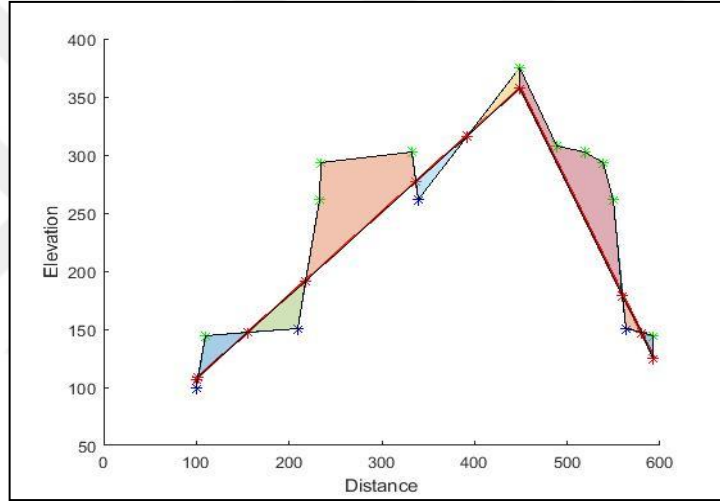
Şekil 4.16. X-Y eksenini için yapılan denemeler

Her bir denemede yapılan tüm tekrarlamalar arasından yarma-dolgu miktarı en az ve aynı zamanda eşit olan işlem yazılım tarafından en optimum kırmızı çizgi olarak önerilmektedir. Dolayısıyla, yazılım tarafından en ekonomik kırmızı çizgi, boykesit üzerinde tüm detaylarıyla birlikte çizilerek gösterilmektedir.

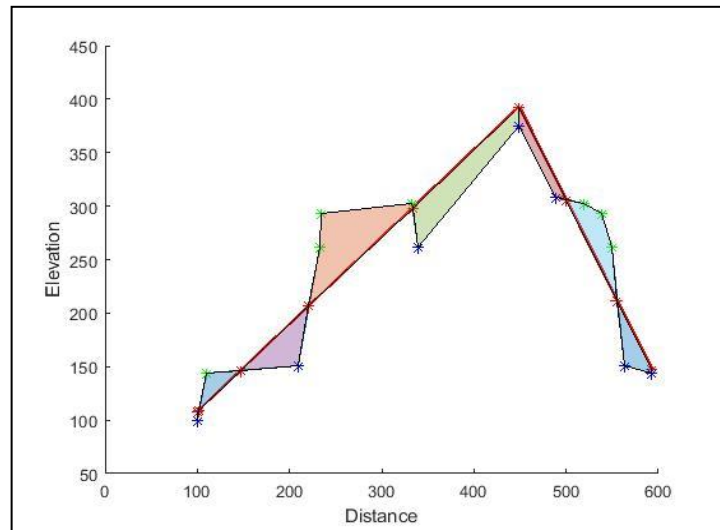
Geliştirilmiş olan her üç optimizasyon yönteminin en ekonomik bulduğu kırmızı çizgi, yazılım tarafından tüm karmaşık hesaplamalara rağmen bulunacaktır. Bu bağlamda, *PSO* yönteminin en optimum tasarladığı kırmızı çizgi Şekil 4.17'de ve *FA* yönteminin tasarladığı kırmızı çizgi Şekil 4.18'de ve *ABC* yöntemi ise Şekil 4.19'da gösterilmiştir.



Şekil 4.17. PSO yöntemi ile en optimum kırmızı çizgi



Şekil 4.18. FA yöntemi ile en optimum kırmızı çizgi

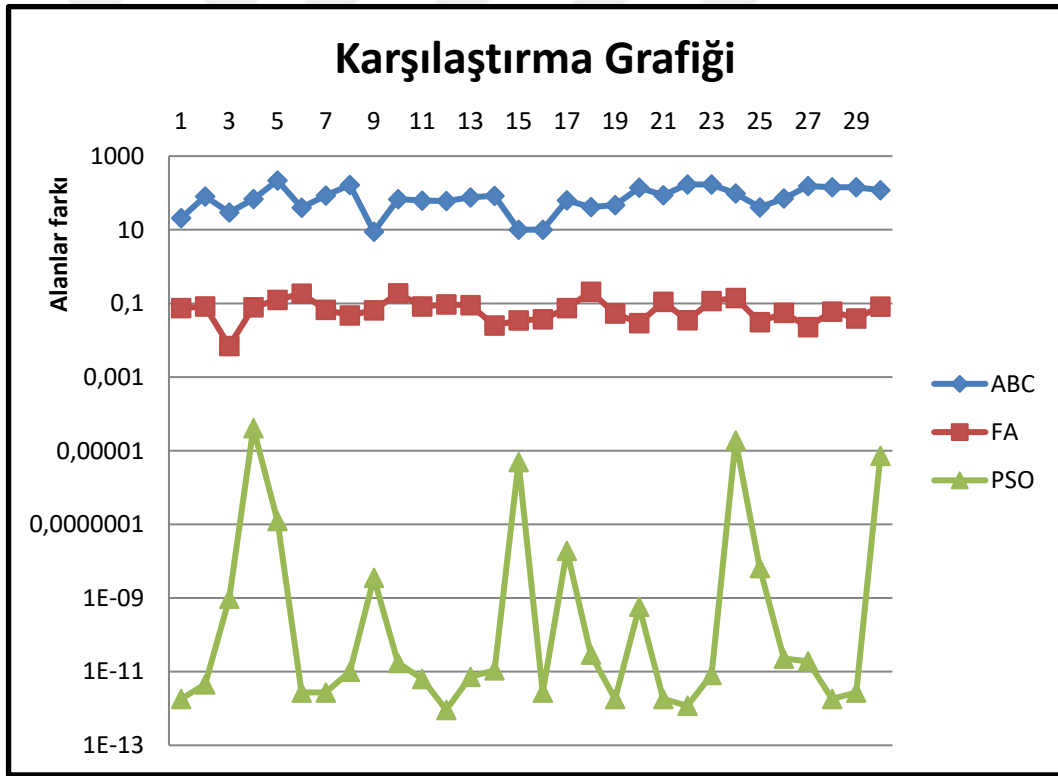


Şekil 4.19. ABC yöntemi ile en optimum kırmızı çizgi

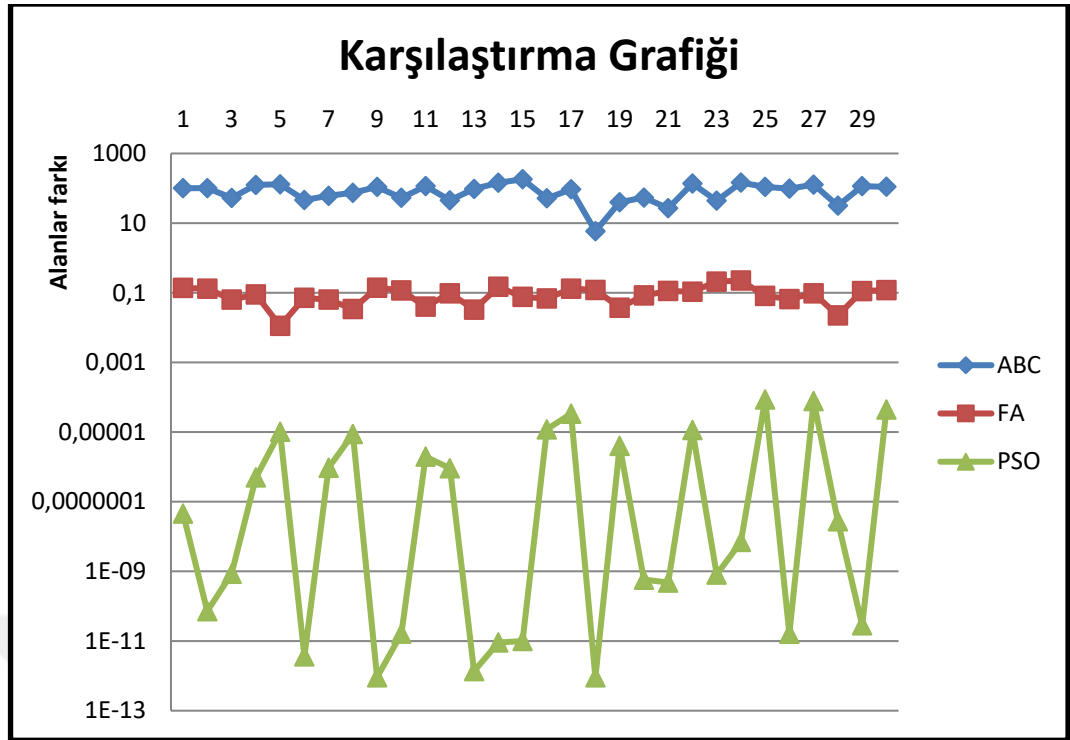
4.1.5. Y ve X-Y Eksenleri Optimizasyonu İçin Karşılaştırma Diyagramları

Bu çalışma kapsamında kullandığımız yöntemler (*ABC-FA-PSO*) ile elde ettiğimiz sonuçlar çok tatmin edici seviyededir. Ancak, bu yöntemlerden *ABC* yönteminin diğerlerine göre sonuçları çok az istikrarlı ve aksine *PSO* yöntemi ile elde ettiğimiz değerler çok istikrarlı ve neredeyse en optimum değere sahiptir.

Bu bağlamda, kırmızı çizgi optimizasyonu ile ilgili hazırlanan yazılım, meta-sezgisel algoritma optimizasyon yöntemlerine entegre edilmiştir. Daha sonra ise her üç yöntemi de ayrı ayrı hem Y eksenı üzerinde hem de X-Y ekseninde, 30 kez deneme yaparak değerlendirip ve karşılaştırması yapılmıştır. Bu karşılaştırma kapsamında, Y eksenı için kullandığımız yöntemlerden elde ettiğimiz diyagramlar Şekil 4.20'de ve X-Y eksenı için kullandığımız yöntemlerden elde ettiğimiz diyagramlar ise Şekil 4.21'de gösterilmektedir.



Şekil 4.20. Y eksenı üzerinde yapılan karşılaştırma grafiği



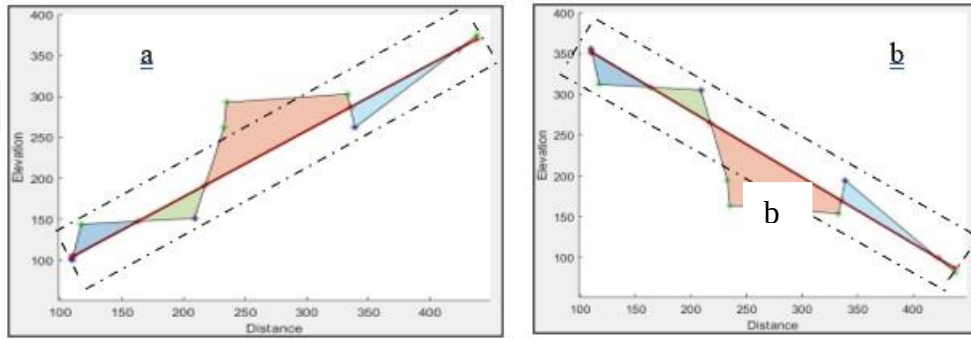
Şekil 4.21. X-Y eksenini üzerinde yapılan karşılaştırma grafiği

Yukarıdaki diyagramlar da gösterildiği gibi örnek üzerinde kullandığımız *PSO* yöntemi ile tasarlanan kırmızı çizgide toprak işleri dengelemesi gayet iyi sonuçlanmaktadır. Başka bir deyişle, bu yöntem kullanıldığında, toplam yarma-dolgu farkı neredeyse sıfır denilecek kadar bir değere ulaşmaktadır. Ama *FA* yönteminde bu fark sıfır ile bir rakamları arasında değişmektedir ve *ABC* yönteminde ise ortalama 90 civarında bir fark ile değişmektedir. Bu sayısal verilere göre bu çalışmada *PSO* değerleri daha çok güvenilir ve istikrarlı görünmektedir. Ayrıca, *PSO* yöntemi ile elde edilen verilere göre Y eksenini ve X-Y eksenini üzerindeki incelemelerin sonuçları neredeyse birbirlerine çok yakındır. Ancak, kullandığımız üç farklı meta-sezgisel algoritmalar ayrıca, Y eksenini ve X-Y eksenini üzerinde ayrı ayrı denenmesine göre demek ki toplamda 6 kez bu deneme tekrarlanmış olacaktır. Bu bağlamda, her deneme 2000 iterasyonla yapıldığına göre bu yöntemler arasından *ABC* yöntemi diğer iki yöntemle göre her ne kadar da ekonomik kırmızı çizgi bulamıyor olsa da diğer yöntemlere göre daha hızlı sonuca ulaşması gözlenmektedir. Buna bağlı olarak görüldüğü gibi optimizasyon yapıldığında ekonomik olması ve hızlı sonuca ulaşma konusu iki farklı konu olarak incelenmelidir. Ancak, bu çalışma yapıldığında süre bakımından tüm yöntemler yaklaşık aynı süre içinde sonuçlanmıştır. Buda bize daha ekonomik olma seçeneğini daha ön plana çıkartmıştır.

4.2. Çok-Amaçlı Optimizasyon Algoritmaları ile İlgili Bulgular

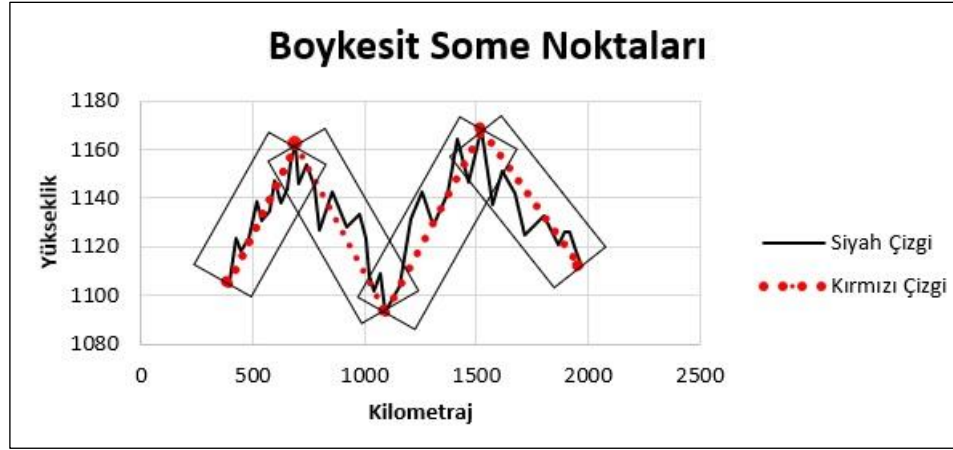
Günümüzde kullanılan karayolu tasarım programları, optimum bir kırmızı çizgi tasarımı ile ilgili bir yöntem sunmamaktadır. Bu çalışmada, tasarlanan algoritma sayesinde optimum kırmızı çizgi tasarımına ilişkin ön adımlar atılmıştır.

Bu çalışma kapsamında, örnek olarak bir karayolu projesinin bir kısmı göz önünde bulundurularak alınan bu karayolu projesi tamamen rastgele mesafelerden ve yükseklik kotlarından oluşmaktadır. Bu çalışma kapsamında, sadece doğru parçaların optimizasyonu yapılmaktadır. Bu parçalar Şekil 4.22’de gözüktüğü gibi a ve b halinde gösterilmektedir. X eksenini mesafeyi ve Y eksenini yükseklik kotlarını göstermektedir.



Şekil 4.22. Doğru parçaların ayrı ayrı optimizasyonu

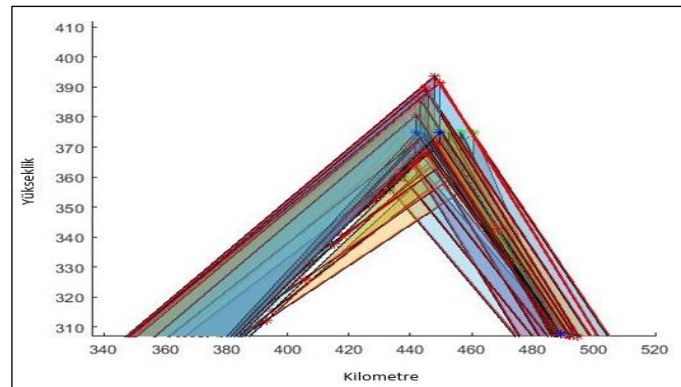
Örnek proje üzerinde kırmızı çizginin some noktalarını zemin hattının pik noktaları olarak varsayarsak tüm veriler bu pik noktalara göre parçalar haline getirilecektir. Böylece, algoritmamız daha az veri üzerinden en optimum kırmızı çizginin bulunmasına yardımcı olmakla birlikte daha az bir süre içinde yarma-dolgu alanları eşitlenebilecektir. Sonuç olarak some noktaları hazırlanan yazılım sayesinde, X ve Y yönünde binlerce yer değiştirme kabiliyetine sahip olduğuna göre, some noktasının seçildiği yerin çok da önemli olmadığı yadsınamaz bir gerçektir. Şekil 4.23'te some noktasının seçildiği yer ve tüm verilerin nerelerden parçalandığı gösterilmektedir. Bu çalışma kapsamında örnek projeye ait ilk iki doğru parçası baz alınarak bu çalışmaya girişilmiştir.



Şekil 4.23. Some noktalarının parçalanma durumu

Tasarlanan algoritma kodlaması, 3 farklı çok-amaçlı meta-sezgisel optimizasyon algoritması ile denenerek sonuçları kaydedilmektedir. Kullanılan yöntemler (*NSGA3-MOPSO-SPEA2*) ile ilgili genel bilgiler literatürde açıklanmaktadır. Bu çalışma kapsamında, hazırlanan yazılım, yukarıdaki yöntemler ile entegre edilerek mercek altına alınmıştır. Başka bir deyişle, some noktasını X ve Y eksenı üzerinde bir matris oluşturarak yer deęiřtirmesine imkân saęlanacaktır.

Bu bağlamda, X-Y eksenı üzerinde yapılan uygulama Şekil 4.24'te detaylı bir şekilde gösterilmektedir. Bu şeklin yatay kısmı kilometrajı ve dikey kısmı ise yükseklięi göstermektedir. Görüldüęü gibi some noktalarının yer deęiřtirmeleri X ve Y eksenı üzerinin her dört yönünde rastgele yapılabilmektedir.



Şekil 4.24. X-Y eksenı açılımı

4.2.1. Kırmızı Çizgi ile İlgili NSGA3 Optimizasyon Sonuçları

Örnek bir karayolu projesine ait tasarlanan kırmızı çizginin some noktaları her iki eksen üzerinde yani X ve Y eksenlerinin yukarı ve aşağısı aynı zamanda saę ve sol taraflarında bir aralık verilerek, çok-amaçlı meta-sezgisel yöntemlerinden, *NSGA3* ile

denetimi yapılmaktadır. Bu bağlamda, optimizasyon yöntemi uygulanan örnek projenin sonuçları aşağıdaki Tablo 4.3'te gösterilmektedir. Bu tabloda gözüktüğü gibi örnek bir karayolu projesi üzerinde uygulanmış kırmızı çizgi optimizasyon sonuçları daha gerçekçi ve güvenilir olması gerektiğinden bu deneme işlemi 30 kez yapılmıştır. Tabi ki bu yöntemde, örnek proje iki parça halinde incelendiği için buna bağlı olarak, tabloda önce her parçanın sonuçları yani yarma-dolgu farkı ayrı ayrı verilmektedir. Buna bağlı olarak, en son bu projeyi tek bir parça düşünerek bu ayrı ayrı yapılmış olan incelemenin toplamı alınarak tablonun son sütünün da gösterilmiştir.

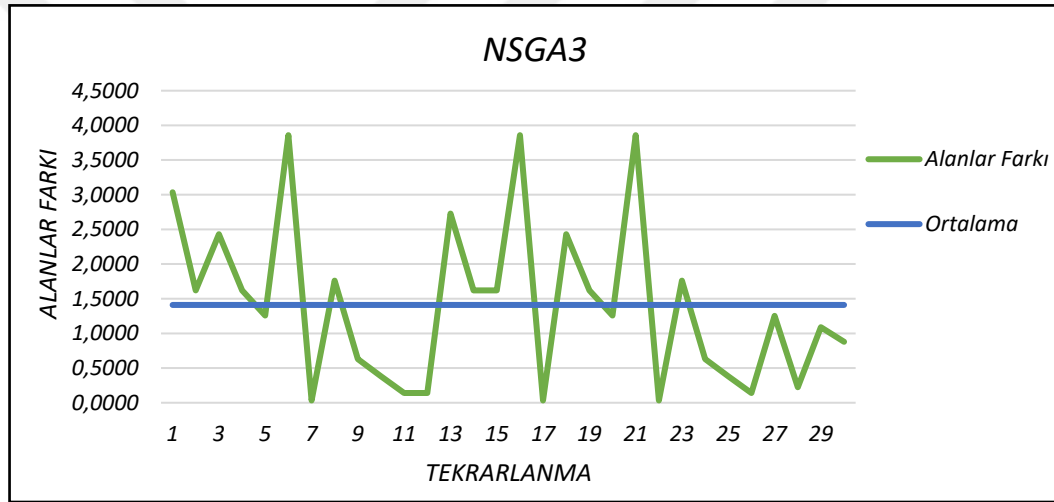
Tablo 4-3. NSGA3 yöntemi ile yarma-dolgu alanlarının farkı

NO	1'İNCİ PARÇANIN DEĞERLERİ	2'NCİ PARÇANIN DEĞERLERİ	TOPLAM
1	0,1688	2,8658	3,0346
2	0,8560	0,7646	1,6206
3	0,1688	2,2601	2,4289
4	0,8560	0,7646	1,6206
5	0,8674	0,3911	1,2585
6	2,8128	1,0456	3,8584
7	0,0087	0,0240	0,0327
8	1,2889	0,4722	1,7611
9	0,2291	0,4007	0,6298
10	0,1208	0,2586	0,3794
11	0,0792	0,0603	0,1395
12	0,0792	0,0603	0,1395
13	0,1688	2,5601	2,7289
14	0,8560	0,7646	1,6206
15	0,8560	0,7646	1,6206
16	2,8128	1,0456	3,8584
17	0,0087	0,0240	0,0327
18	0,1688	2,2601	2,4289
19	0,8560	0,7646	1,6206
20	0,8674	0,3911	1,2585
21	2,8128	1,0456	3,8584
22	0,0087	0,0240	0,0327
23	1,2889	0,4722	1,7611
24	0,2291	0,4007	0,6298
25	0,1208	0,2586	0,3794
26	0,0792	0,0603	0,1395
27	1,0149	0,2383	1,2532
28	0,1380	0,0850	0,2230
29	0,5910	0,4978	1,0888
30	0,6986	0,1809	0,8795

Tablo 4.3'te gözüktüğü gibi tüm veriler bu yöntemle ayı ayrı 2000 defa ve 30 kez deneme yapılarak elde edilmiştir. Bu sonuçlar kapsamında elde edilen ortalama alanlar farklarına ilişkin diyagram Şekil 4.25'te gösterilmektedir. Bu diyagramın X

ekseni tekrarlanma sayısını ve Y eksenini o deneme için en optimum yarma-dolgu miktarına denk gelen alanlar farkını göstermektedir. Tabi ki bu çalışmaya, yarma-dolgu miktarlarını en aza düşürmek ve aynı zamanda aralarındaki farkı sıfıra yakın bir değer bulmak için girilmiştir.

Tablo 4.3'te gözüktüğü gibi tüm veriler *NSGA3* yönteminin 2000 iterasyonla 30 kez deneme yapılarak elde edilen sonuçlarıdır. Bu çalışmada arazi verileri iki parça haline getirilerek incelemesi yapılmıştır. Bu sayede yarma-dolgu alanlarının dengelemesi için her parçaya ait ayrı bir değer bulunduğundan bu değerlerin toplamı alınarak denetim sonlandırılmıştır. Bu sonuçlar kapsamında elde edilen diyagram, *NSGA3* yöntemi için Şekil 4.25'te gösterilmiştir. Bu defalarca deneme sonucuna göre *NSGA3* yönteminde ortalama alanlar farkı 1.41 birim olarak görülmektedir.



Şekil 4.25. *NSGA3* yönteminin sonuçları

4.2.2. Kırmızı Çizgi ile İlgili *MOPSO* Optimizasyon Sonuçları

Örnek bir karayolu projesine ait tasarlanan kırmızı çizginin some noktaları her iki eksen üzerinde yani X ve Y eksenlerinin yukarı ve aşağısı aynı zamanda sağ ve sol taraflarında bir aralık verilerek, çok-amaçlı meta-sezgisel yöntemlerinden, *MOPSO* ile denetimi yapılmaktadır. Bu bağlamda, optimizasyon yöntemi uygulanan örnek projenin sonuçları aşağıdaki Tablo 4.4'te gösterilmektedir. Bu tabloda gözüktüğü gibi örnek bir karayolu projesi üzerinde uygulanmış kırmızı çizgi optimizasyon sonuçları daha gerçekçi ve güvenilir olması gerektiğinden bu deneme işlemi 30 kez yapılmıştır. Tabi ki bu yöntemde, örnek proje iki parça halinde incelendiği için buna bağlı olarak, tabloda önce her parçanın sonuçları yani yarma-dolgu farkı ayrı ayrı verilmektedir.

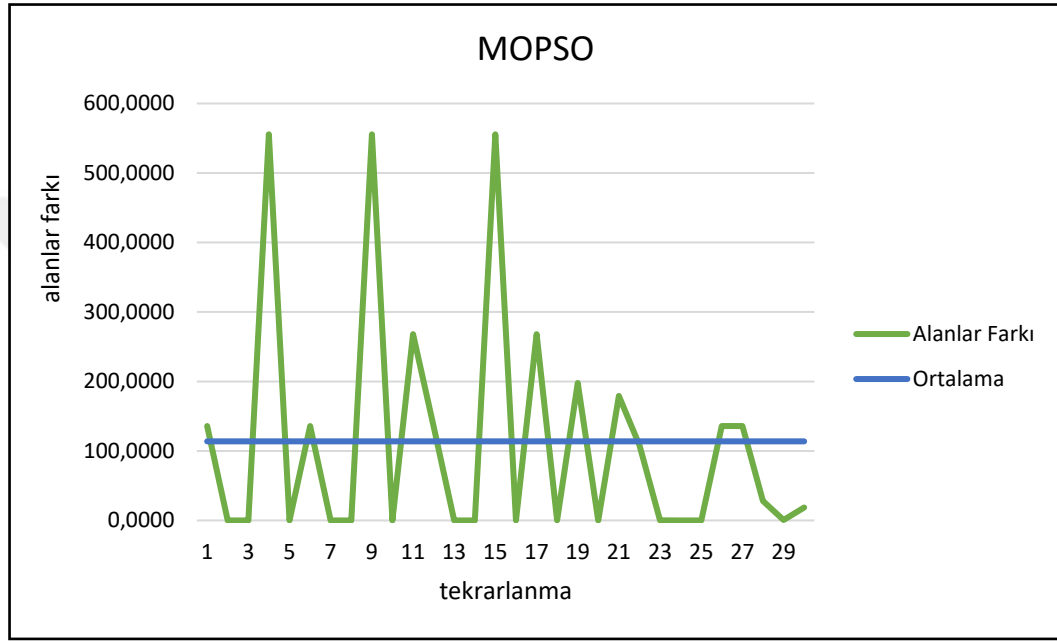
Buna bağılı olarak, en son bu projeyi tek bir parça düşünerek bu ayrı ayrı yapılmış olan incelemenin toplamı alınarak tablonun son sütünün da gösterilmiştir.

Tablo 4-4. MOPSO yöntemi ile yarma-dolgu alanlarının farkı

NO	1'İNCİ PARÇANIN DEĞERLERİ	2'NCİ PARÇANIN DEĞERLERİ	TOPLAM
1	49,1854	86,7756	135,9610
2	1,29E-06	1,05E-04	0,0001
3	0,0218	0,0697	0,0915
4	136,9353	418,6839	555,6192
5	0,0258	0,0089	0,0347
6	49,1854	86,7756	135,9610
7	1,29E-06	1,05E-04	0,0001
8	0,0218	0,0697	0,0915
9	136,9353	418,6839	555,6192
10	0,0258	0,0089	0,0347
11	193,2634	74,7027	267,9661
12	49,1854	86,7756	135,9610
13	1,29E-06	1,05E-04	0,0001
14	0,0218	0,0697	0,0915
15	136,9353	418,6839	555,6192
16	0,0258	0,0089	0,0347
17	193,2634	74,7027	267,9661
18	0,0221	0,0773	0,0994
19	136,5090	61,3534	197,8624
20	1,29E-06	1,05E-04	0,0001
21	92,5244	86,5656	179,0900
22	107	0,7304	108,0205
23	5,71E-08	0,24	0,2411
24	0,0014	0,0016	0,0030
25	1,29E-06	1,82E-11	0,0000
26	49,1854	86,7756	135,9610
27	49,1854	86,7756	135,9610
28	1,82E-12	28,0000	28,0000
29	0,0567	0,6419	0,6986
30	7,0057	11,7013	18,7070

Tablo 4.4'te gözüktüğü gibi tüm veriler bu yöntemle ayı ayrı 2000 defa ve 30 kez deneme yapılarak elde edilmiştir. Bu sonuçlar kapsamında elde edilen ortalama alanlar farklarına ilişkin diyagram Şekil 4.26'da gösterilmektedir. Bu diyagramın X eksenini tekrarlanma sayısını ve Y eksenini o deneme için en optimum yarma-dolgu miktarına denk gelen alanlar farkını göstermektedir. Tabi ki bu çalışmaya, yarma-dolgu miktarlarını en aza düşürmek ve aynı zamanda aralarındaki farkı sıfıra yakın bir değer bulmak için girişilmiştir.

Tablo 4.4'te gözüktüğü gibi tüm veriler *MOPSO* yönteminin 2000 iterasyon ile 30 kez deneme yapılarak elde edilen sonuçlarıdır. Bu çalışmada arazi verileri iki parça haline getirilerek incelemesi yapılmıştır. Bu sayede yarma-dolgu alanlarının dengelemesi için her parçaya ait ayrı bir değer bulunduğundan bu değerlerin toplamı alınarak denetim sonlandırılmıştır. Bu sonuçlar kapsamında elde edilen diyagram, *MOPSO* yöntemi için Şekil 4.26'da gösterilmiştir. Bu defalarca deneme sonucuna göre *MOPSO* yönteminde ortalama alanlar farkı 113.85 birim olarak gözükmektedir.



Şekil 4.26. *MOPSO* yönteminin sonuçları

4.2.3. Kırmızı Çizgi ile İlgili *SPEA2* Optimizasyon Sonuçları

Örnek bir karayolu projesine ait tasarlanan kırmızı çizginin some noktaları her iki eksen üzerinde yani X ve Y eksenlerinin yukarı ve aşağısı aynı zamanda sağ ve sol taraflarında bir aralık verilerek, çok-amaçlı meta-sezgisel yöntemlerinden, *SPEA2* ile denetimi yapılmaktadır. Bu bağlamda, optimizasyon yöntemi uygulanan örnek projenin sonuçları aşağıdaki Tablo 4.5'te gösterilmektedir. Bu tabloda gözüktüğü gibi örnek bir karayolu projesi üzerinde uygulanmış kırmızı çizgi optimizasyon sonuçları daha gerçekçi ve güvenilir olması gerektiğinden bu deneme işlemi 30 kez yapılmıştır. Tabi ki bu yöntemde, örnek proje iki parça halinde incelendiği için buna bağlı olarak, tabloda önce her parçanın sonuçları yani yarma-dolgu farkı ayrı ayrı verilmektedir. Buna bağlı olarak, en son bu projeyi tek bir parça düşünerek bu ayrı ayrı yapılmış olan incelemenin toplamı alınarak tablonun son sütünün da gösterilmiştir.

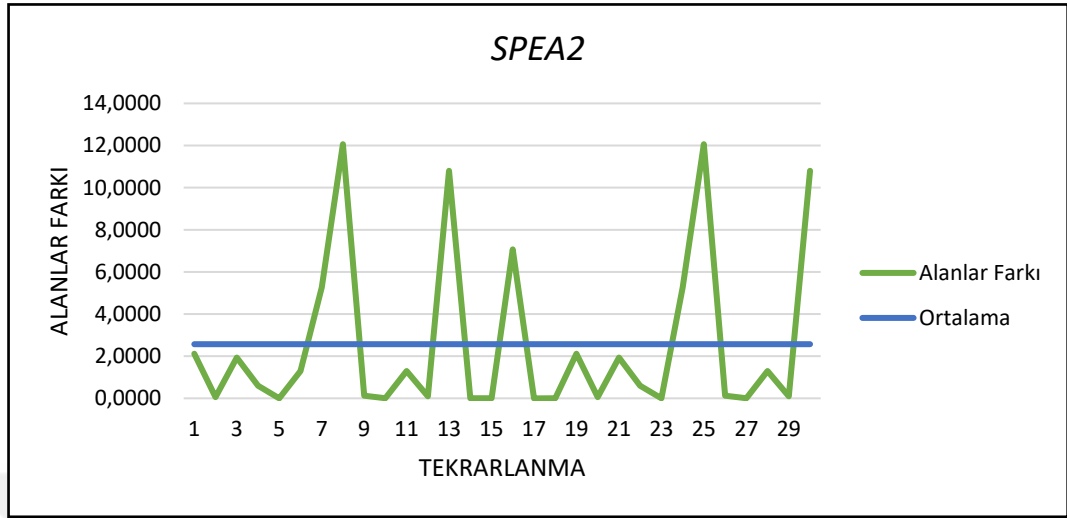
Tablo 4-5. *SPEA2* yöntemi ile yarma-dolgu alanlarının farkı

NO	1'İNCİ PARÇANIN DEĞERLERİ	2'NCİ PARÇANIN DEĞERLERİ	TOPLAM
1	1,2040	0,9097	2,1137
2	3,08E-04	0,0542	0,0545
3	0,892	1,0454	1,9374
4	0,0999	0,4966	0,5965
5	9,04E-07	1,31E-07	0,0000
6	0,7184	0,5791	1,2975
7	5,2694	9,84E-07	5,2694
8	9,0276	3,0361	12,0637
9	0,082	0,0411	0,1231
10	4,09E-04	0,0075	0,0079
11	0,7184	0,5791	1,2975
12	1,01E-07	0,0937	0,0937
13	8,7242	2,0784	10,8026
14	0,0056	0,0012	0,0068
15	0,0057	1,58E-05	0,0057
16	7,0383	0,0291	7,0674
17	3,88E-08	6,79E-06	0,0000
18	4,37E-11	0,0045	0,0045
19	1,2040	0,9097	2,1137
20	3,08E-04	0,0542	0,0545
21	0,892	1,0454	1,9374
22	0,0999	0,4966	0,5965
23	9,04E-07	1,31E-07	0,0000
24	5,2694	9,84E-07	5,2694
25	9,0276	3,0361	12,0637
26	0,082	0,0411	0,1231
27	4,09E-04	0,0075	0,0079
28	0,7184	0,5791	1,2975
29	1,01E-07	0,0937	0,0937
30	8,7242	2,0784	10,8026

Tablo 4.5'te gözüktüğü gibi tüm veriler bu yöntemle ayı ayrı 2000 defa ve 30 kez deneme yapılarak elde edilmiştir. Bu sonuçlar kapsamında elde edilen ortalama alanlar farklarına ilişkin diyagram Şekil 4.27'de gösterilmektedir. Bu diyagramın X eksenini tekrarlanma sayısını ve Y eksenini o deneme için en optimum yarma-dolgu miktarına denk gelen alanlar farkını göstermektedir. Tabi ki bu çalışmaya, yarma-dolgu miktarlarını en aza düşürmek ve aynı zamanda aralarındaki farkı sıfıra yakın bir değer bulmak için girilmiştir.

Tablo 4.5'te gözüktüğü gibi tüm veriler *SPEA2* yönteminin 2000 iterasyon ile 30 kez deneme yapılarak elde edilen sonuçlarıdır. Bu çalışmada arazi verileri iki parça haline getirilerek incelemesi yapılmıştır. Bu sayede yarma-dolgu alanlarının dengelemesi için her parçaya ait ayrı bir değer bulunduğundan bu değerlerin toplamı alınarak denetim sonlandırılmıştır. Bu sonuçlar kapsamında elde edilen diyagram,

SPEA2 yöntemi için Şekil 4.27’de gösterilmiştir. Bu defalarca deneme sonucuna göre *SPEA2* yönteminde ortalama alanlar farkı 2.57 birim olarak gözükmektedir.

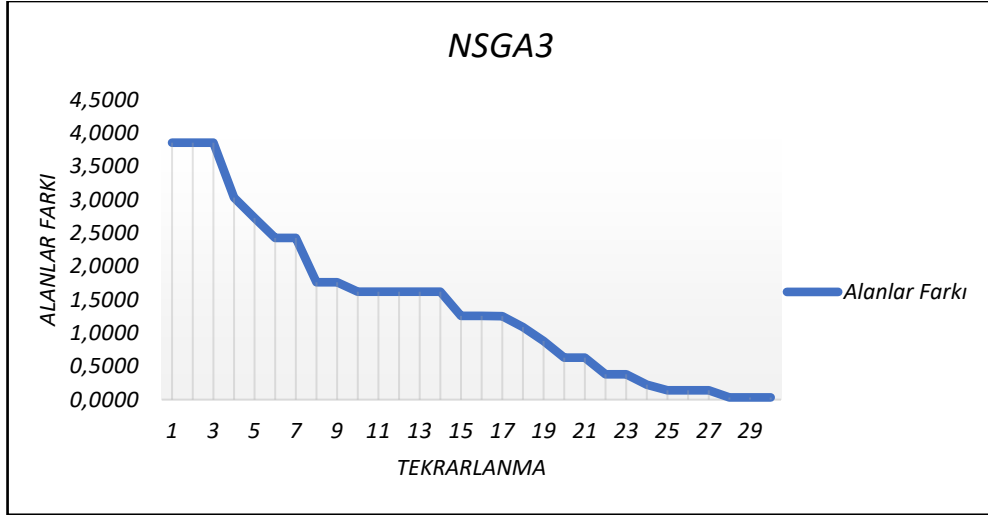


Şekil 4.27. *SPEA2* yönteminin sonuçları

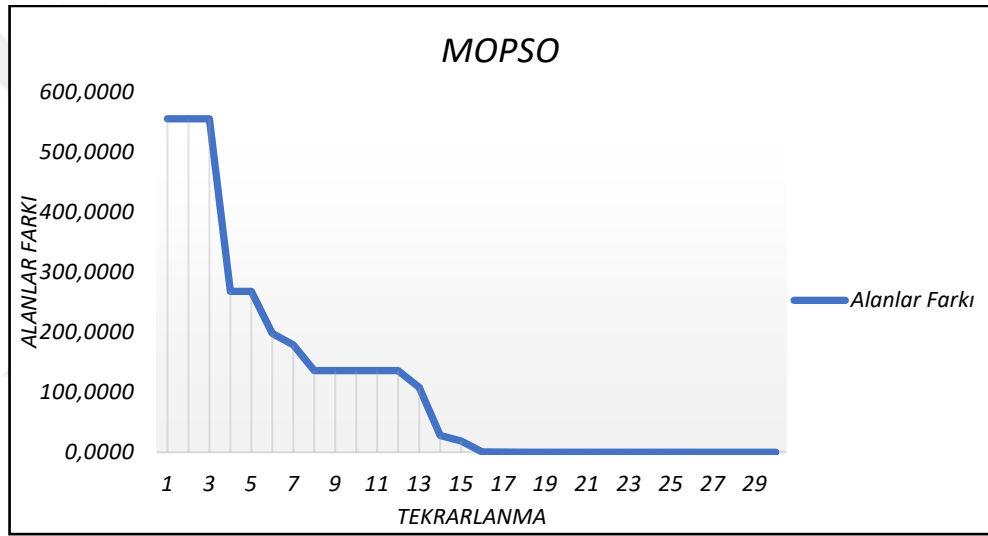
4.2.4. Çok-amaçlı meta-sezgisel optimizasyonların karşılaştırma diyagramları

Bu çalışma kapsamında kullandığımız yöntemler (*NSGA3-MOPSO-SPEA2*) ile elde ettiğimiz sonuçlar çok tatmin edici seviyededir. Ancak, bu yöntemlerden *MOPSO* yönteminin sonuçları diğerlerine göre çok az istikrarlı ve güvenilirdir. Ama, aksine *NSGA3* yöntemi ile elde ettiğimiz değerler çok istikrarlı ve neredeyse en optimum değere sahiptir.

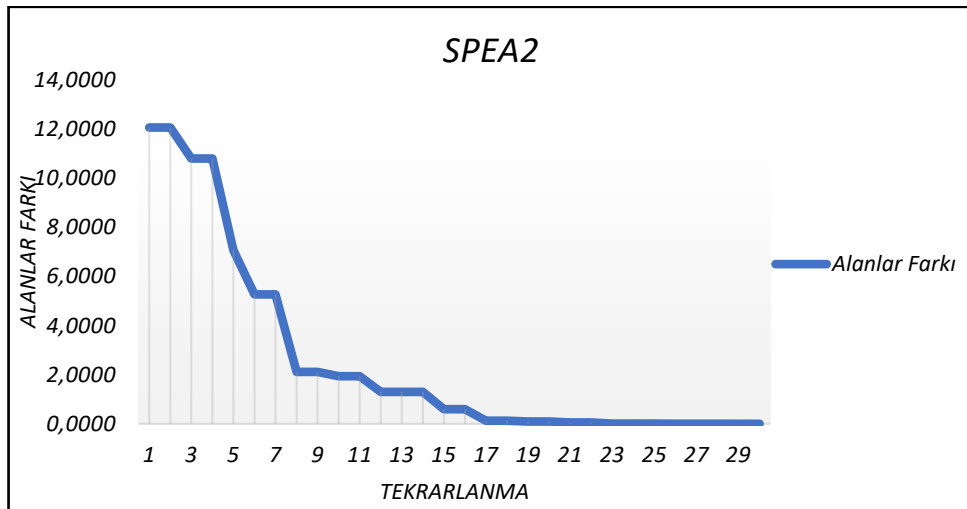
Bu bağlamda, kırmızı çizgi optimizasyonu ile ilgili hazırlanan yazılım, çok-amaçlı meta-sezgisel algoritma optimizasyonlarına entegre edilmiştir. Daha sonra ise her üç yöntemi X-Y ekseninde, 30 kez tekrarlayarak değerlendirip ve karşılaştırması yapılmıştır. Başka bir deyişle, bu 30 defa tekrarlamadan, 30 tane sonuç oluşturulmaktadır. Tabi ki bu sonuçlar içinden en az miktarı gösteren en optimum yarma-dolgu miktarını göstermektedir. Böylece, tasarlanan algoritmalar tekrarlandığı takdirde daha verimli sonuçlar sunmaktadır. Bu karşılaştırma kapsamında, X-Y ekseninde kullandığımız *NSGA3*, *MOPSO* ve *SPEA2* yöntemleri ile elde ettiğimiz diyagramlar sırasıyla Şekil 4.28, 4.29 ve 4.30’da gösterilmektedir.



Şekil 4.28. NSGA3 ile ilgili karşılaştırma grafiği



Şekil 4.29. MOPSO ile ilgili karşılaştırma grafiği

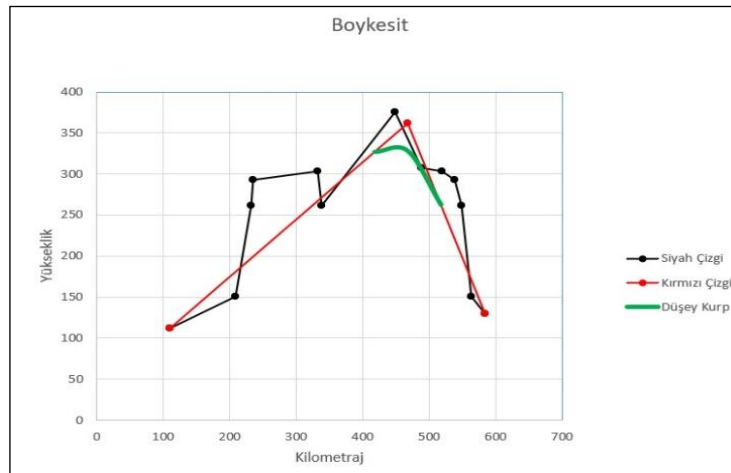


Şekil 4.30. SPEA2 ile ilgili karşılaştırma grafiği

Yukarıdaki diyagramlar da gösterildiği gibi örnek üzerinde kullandığımız *NSGA3*, *MOPSO* ve *SPEA2* yöntemleri ile tasarlanan kırmızı çizgide toprak işleri dengelemesi gayet iyi sonuçlanmaktadır. Başka bir deyişle, bu yöntemler kullanıldığında, toplam yarma-dolgu farkı neredeyse sıfır denilecek kadar bir değere ulaşmaktadır. Ancak, bu diyagramlardan da gözüktüğü gibi her yöntem 30 defa tekrarlanmıştır ama her üç yöntem de sonunda sıfır değere ulaşsalar bile *NSGA3* yönteminin maksimum bulduğu alan farkı 4 olduğu halde sadece 28'inci denemesinden itibaren sıfır değere ulaşmıştır. *MOPSO* yöntemi ise her ne kadar maksimum denemesinde bulduğu değer yaklaşık 555 olsa da yaklaşık 16'ncı denemesinden itibaren sıfır değerini bulmaktadır. *SPEA2* yöntemine göre ise maksimum bulduğu değer 12 civarında olduğu gibi 17'nci denemeden itibaren sıfır değerine ulaşmaktadır. Bu bağlamda, *SPEA2* yöntemi diğer iki yönteme göre tüm denemelerinde sıfıra yakın olmasının yanı sıra daha erken sıfıra ulaşmış ve daha fazla denemesinde sıfır değerini bulabilmiştir.

4.3. Meta-Sezgisel Algoritmaları Kullanarak Düşey Kurp Optimizasyonu

Yol projelerinde siyah çizgi üzerinden kırmızı çizginin doğru parçaları geçirilir ve bu doğru parçaların kesiştiği noktalarda yani some noktalarında düşey kurp yapılması gerekmektedir. Şekil 4.31'de gözüktüğü gibi siyah çizgiyi kesen doğru parçaların kesiştiği noktaya bir düşey kurp çizilmektedir.

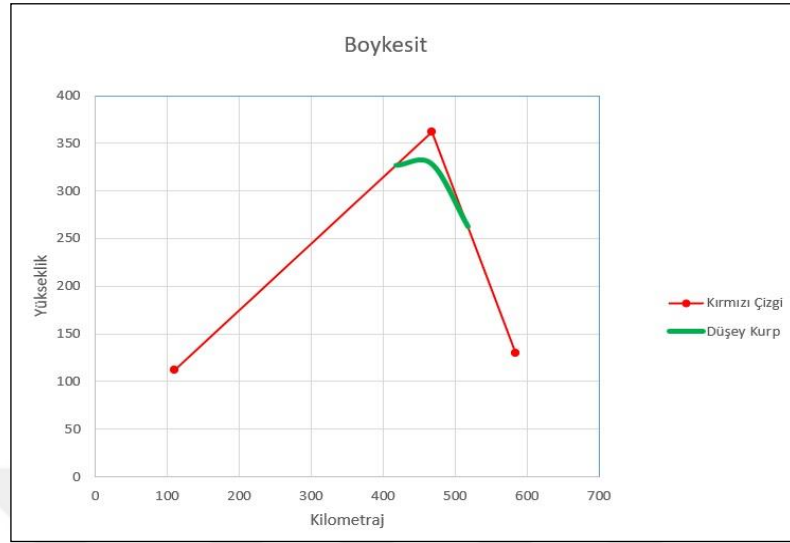


Şekil 4.31. Some noktasına düşey kurba yerleştirilmesi

4.3.1. Meta-Sezgisel Algoritmalar ile Düşey Kurp Uygulaması

Yukarıda gözükmemekte olan kırmızı çizgi ve aynı zamanda some noktasında mevcut olan düşey kurba tek başına Şekil 4.32'de gösterilmiştir. Bu bağlamda, düşey

kurbayı meta-sezgisel optimizasyon algoritmalarına entegre edebilmek için satırlarca kodlama gerekmektedir.



Şekil 4.32. Some noktasına düşey kurp yerleştirilmesi

Bu bağlamda, aynı düşey kurbayı karmaşık matematiksel işlemleri ile yazılıma entegre edebilmek için satırlarca kod yazılmalıdır. Şekil 4.33'te bu kodlamaya ilişkin bir takım yazılım, sırasıyla gösterilmektedir. Örnek olarak yapılmış olan bu kodlamada önce kırmızı çizgiye farazi olarak bir koordinat verilmektedir. Daha sonra bu kırmızı çizgiye ait eğimler hesaplanmış olacaktır. Eğim hesabı yapıldıktan sonra düşey kurba uzunluğu tasarımcıdan sorularak yazılıma girilmiş olacaktır. Bu veriler ile düşey kurp hesaplama işlemleri, yazılım tarafından her ne kadar da karmaşık işlemlerden oluşsa bile çok hızlı bir şekilde yazılım tarafından gerçekleştirilmiş olacaktır.

```

xa=110,70
ya=111,80
xb=468,10
yb=361,40
xc=584,40
yc=130,20

x=[xa;xb;xc];
y=[ya;yb;yc];
plot(x,y)

g1=(yb-ya)/(xb-xa);
g2=(yc-yb)/(xc-xb);

G=abs(g1-g2);
prompt = 'What is the original value? ';
L = input(prompt);
T=L/2
E=(L*G)/8
KMT1=xb-T
KMT2=xb+T
KOTT1=yb-(g1*T)
KOTT2=yb+(g2*T)
KOTS=yb-E
xy=[KMT1 KOTT1;xb KOTS;KMT2 KOTT2];

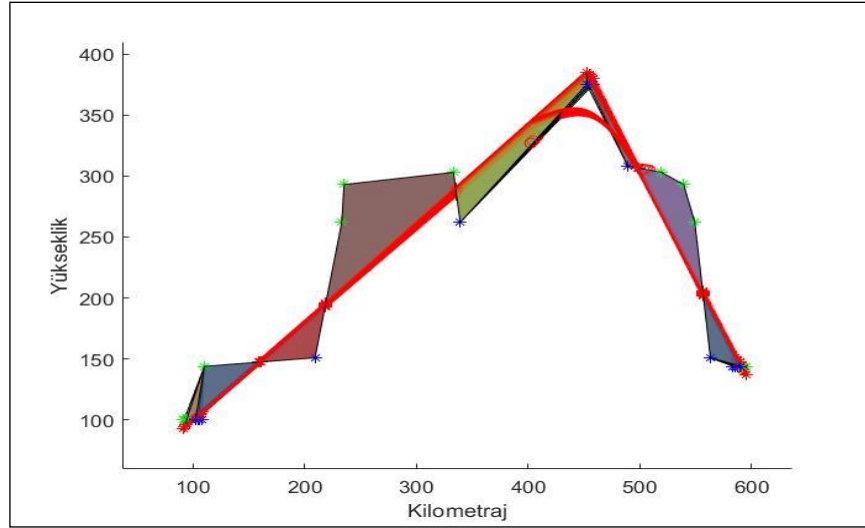
hold on
fplot(cscvn(xy),'r',2)
hold off

```

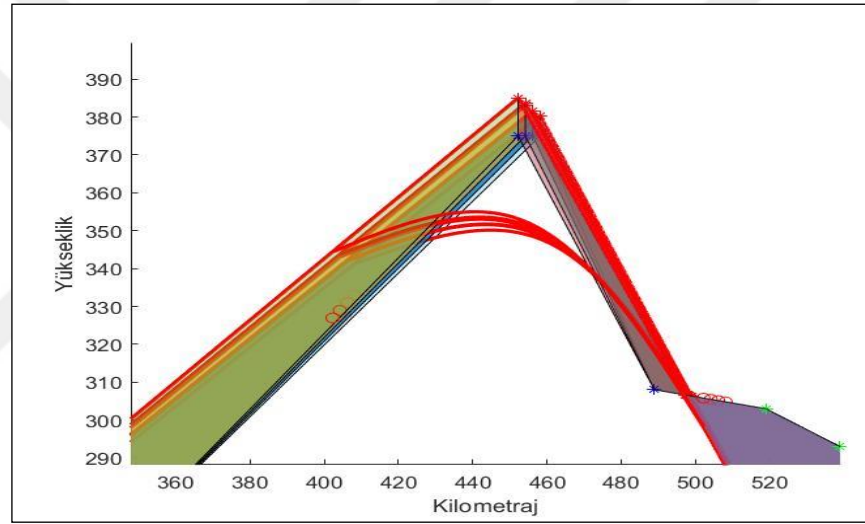
Şekil 4.33. Düşey kurp yazılım kodlaması

4.3.2. Düşey kurp optimizasyonu

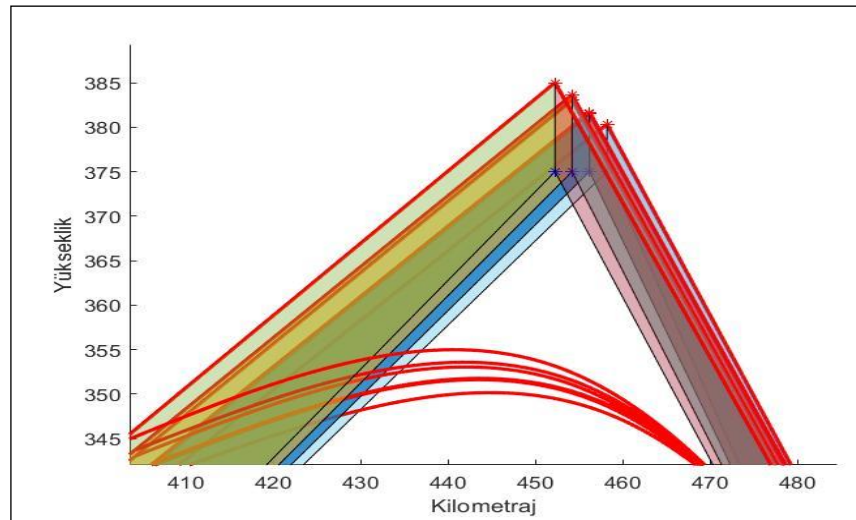
Yukarıda bahsi geçen kırmızı çizginin doğru parçalarına düşey kurp kodlaması entegre edildiği takdirde Şekil 4.34'te ve tam olarak Şekil 4.35'te gözükmekte olan kırmızı çizgiye bir düşey kurp geçirilmektedir. Bu yazılım sayesinde kırmızı çizginin doğru parçaları arasında düşey kurp yerleştirilmesi mümkün olmuştur. Herhangi bir optimizasyon yöntemi ile tasarım sırasında çizilen kırmızı çizgilere birer tane düşey kurp, yazılım tarafından yapılmaktadır ve bu yazılım sayesinde some noktasının X ve Y yönünde binlerce yer değiştirmesine rağmen her some noktası için ayrı ayrı bir düşey kurp geçirilmektedir. Şekil 4.36'da tam olarak some noktasının yer değiştirmesi ve buna bağlı olarak düşey kurpların değiştiği gözlenmektedir.



Şekil 4.34. Kırmızı çizgilere uygulanan düşey kurbalar



Şekil 4.35. Her some noktası için ayrı bir düşey kurp uygulaması



Şekil 4.36. X ve Y ekseninde some noktalarının yer değişimi ve düşey kurba

4.3.3. Düşey Kurban Siyah Çizgi Üzerindeki Yükseklik Kotları

Karayolu tasarımı, kırmızı çizginin optimizasyonu yapılırken düşey kurbanın siyah çizgi ile kapalı bölgeler oluşturması gerekmektedir. Tabi ki bu kapalı bölgelerin oluşumu için kurp başlangıç ve bitiş noktalarından siyah çizgiye dikler indirilerek bu durum sağlanmalıdır. Bu dik çizgiler ile siyah çizginin kesiştiği noktaların yükseklik kotları Şekil 4.37’de gözükmekte olan kodlama sayesinde bulunacaktır. Ayrıca, bu kesişme noktaların yerleri Şekil 4.38’de gösterilmektedir.

```
[KMT1,KOTT1,xs,KOTS,KMT2,KOTT2]=curve3point(xyvector(4),xyvector(1),xyvector(5),xyvector(2),xyvector(6),xyvector(3))

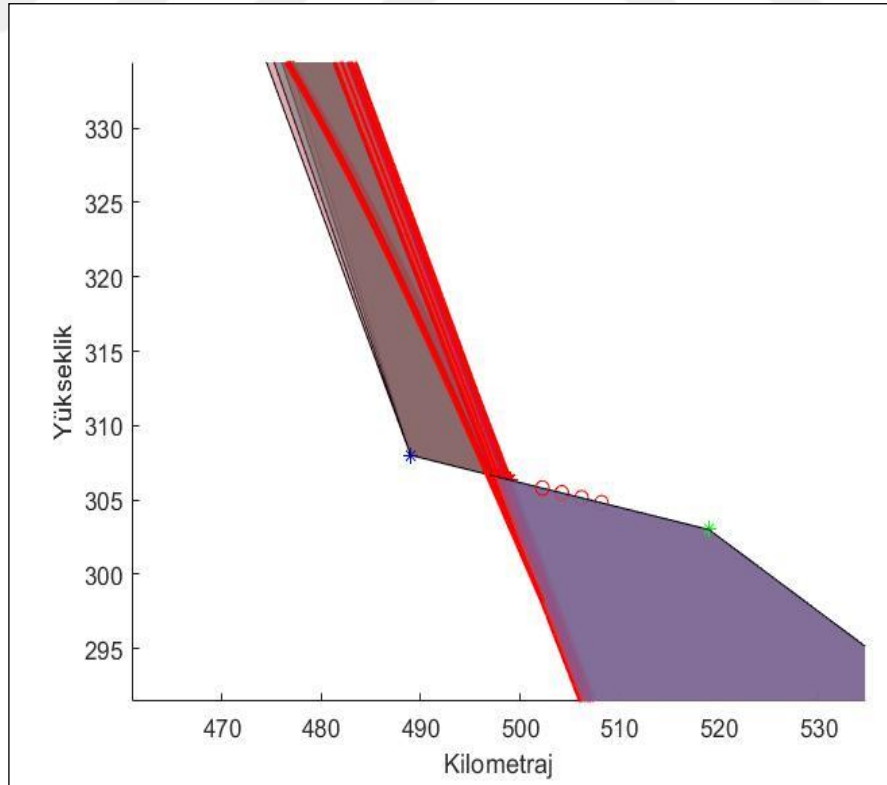
[pointyhat1,pointx1] = ziyaaft(y,x,[KOTT1 KMT1] )
[pointyhat2,pointx2] = ziyaaft(y,x,[KOTT2 KMT2] )
plot(pointx1,pointyhat1,'or')
plot(pointx2,pointyhat2,'or')

lx= x((pointx1<x) & (x<pointx2))
ly= y((pointx1<x) & (x<pointx2))

point_curv_x=[pointx1,pointx2]
point_curv_y=[pointyhat1,pointyhat2]

plot(point_curv_x,point_curv_y,'or')
```

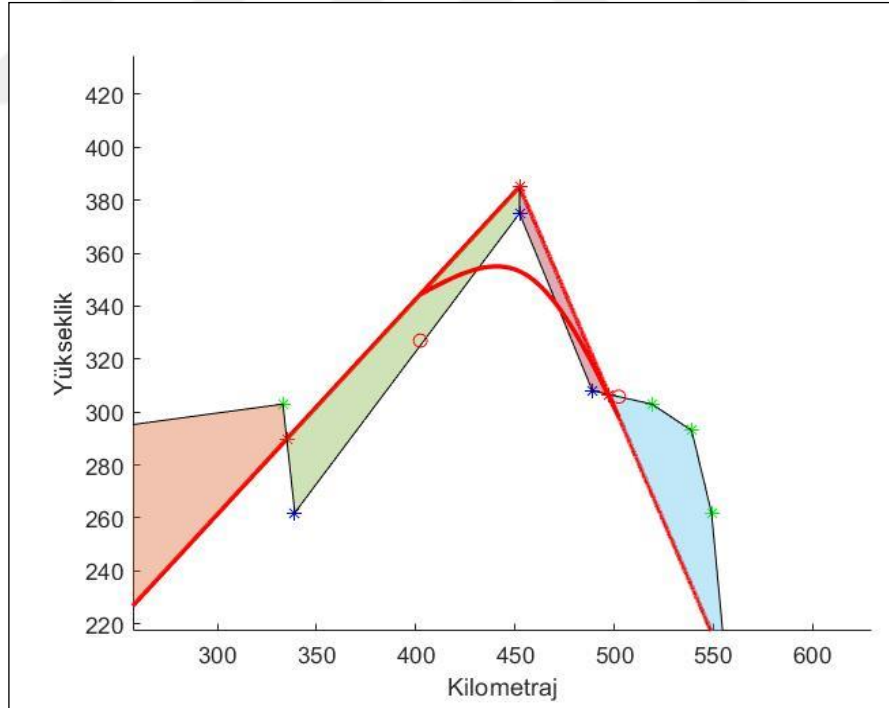
Şekil 4.37. Düşey kurba ve siyah çizgi ilişkisinin yazılım kodlaması



Şekil 4.38. Düşey kurbanın siyah çizgi üzerindeki yüksekliği

4.3.4. Optimum Düşey Kurbanın Seçilmesi

Kırmızı çizginin optimizasyonu sırasında bilindiği gibi, kırmızı çizgi sadece doğru parçalardan oluşmamaktadır. Başka bir deyişle bu doğru parçaların arasına yani some noktalarının bulunduğu yerlere kesinlikle bir düşey karp yerleştirilmelidir. Ancak, optimizasyon sırasında binlerce düşey karp geçirileceğine göre en optimum yarma dolguya sahip olan düşey karp seçilmelidir. Bu bağlamda, kırmızı çizginin some noktalarından geçirilen düşey kurbalarda duruma göre ya dere düşey karp veya tepe düşey karp olabilmektedir. Optimum düşey karp seçiminde tepe düşey karp olması gereken yerlerde yazılım tarafından geçirilmiş olan binlerce düşey kurbadan sadece en üstte olan yani some noktasına en yakın olan düşey karp ve aynı zamanda dere düşey karp olması gereken yerlerde ise en altta olan yani some noktasına en yakın bulunan düşey karp seçilmelidir. Böylece, kırmızı çizgi tasarımında projenin tamamının optimizasyonu gerçekleşmesi için proje boyunca tüm düşey kurbalarda optimizasyon yapılmalıdır. Örnek olarak bir optimum tepe düşey karp Şekil 4.39'da gösterilmektedir.



Şekil 4.39. Örnek bir optimum tepe düşey kur

5. SONUÇ

Bu çalışma kapsamında, tasarlanan algoritma sayesinde optimum kırmızı çizgi tasarımı gerçekleştirilmiştir. Örnek olarak dikkate alınan bu karayolu projesi tamamen rastgele mesafelerden ve yükseklik kotlarından oluşmaktadır. Başlangıç, bitiş ve ayrıca bir tepe düşey kurbası bulunan bu projede sadece iki doğru parça yer almaktadır. Optimizasyonun gerçekleştirilmesi için önce tasarım kriterlerinden, some noktası tayini, eğim sınır koşulları ve arazi üzerindeki yükseklik kotlarının bulunmasına ilişkin bir algoritma geliştirilmiştir ve meta sezgisel algoritmalarına entegre edilmiştir. Tasarlanan algoritma kodlaması, 3 farklı tek-amaçlı meta-sezgisel optimizasyon algoritması (*PSO-FA-ABC*) ile önce sadece Y ekseninde ve daha sonra X ve Y ekseninde toplamda 6 kez bu deneme tekrarlanmıştır. Bu bağlamda, her deneme 2000 iterasyonla yapılarak sonuçları kaydedilmiştir. Tasarlanan algoritma kodlaması, sonuçların daha detaylı incelenmesi için daha sonra 3 farklı çok-amaçlı meta-sezgisel optimizasyon algoritması (*NSGA3-MOPSO-SPEA2*) ile denenerek sonuçları kaydedilmiştir. Her iki yöntemle elde edilen veriler 30 kez program tarafından denenerek karşılaştırılmış ve daha güvenilir ve istikrarlı olan sonuçlar aşağıda özetlenmiştir. Son olarak yöntemler üzerinden elde edilen sonuçlar çok çok tatmin edici bir seviyededir. Ancak, bu bağlamda genel olarak tek amaçlı yöntemlerden *PSO* yöntemi hafriyat dengelemesini neredeyse sıfıra yakın bir değer bulmuştur. Çok amaçlı yöntemlerden *SPEA2* yönteminin sonuçlarının daha güvenilir ve istikrarlı olduğu görülmüş ve sonuçların tatmin edici seviyede olduğu belirlenmiştir.

- Örnek proje üzerinde kırmızı çizginin some noktası, yazılım tarafından otomatik olarak seçilmektedir. Sonuç olarak some noktası hazırlanan yazılım sayesinde, X ve Y yönünde binlerce yer değiştirme kabiliyeti olmaktadır. Başka bir deyişle, bu yazılımı kullanarak some noktalarına binlerce yer değiştirmelerine imkân sağlanmıştır. Bu yazılım kullanılarak projenin pik noktalarına göre otomatik some noktaları seçilmiş ve kırmızı çizgi optimizasyonuna uygun oldukları gözlenmiştir.
- Seçilen some noktaları, optimizasyon algoritmaları tarafından binlerce yer değiştirme imkânı olacağından seçildiği yerin çok da önemli olmadığı anlaşılmaktadır.
- Some noktalarının sadece yazılım tarafından otomatik seçilmesi tam olarak doğru olmayacaktır yani bazı projelerde projenin işleyişi kapsamında zorunlu

some noktaları tasarımcı tarafından elle eklenmesi veya istenmeyen yerlerde some noktalarının çıkartılması gerekmektedir. Bu durum satırlarca kodlama sayesinde programa entegre edilmiştir. Dolayısıyla tasarım esnasında istenilen bölgelere ekstradan some noktası eklenmesi veya istenilmeyen bölgelerden some noktalarının çıkartılması bu çalışmada gerçekleştirilmiş ve rahatlıkla yazılım tarafından işlemin yapıldığı tamamlanmıştır.

- Karayolu şartnamesinde her yol sınıfı için eğim sınırları belirlenmiştir. Bu bağlamda, kırmızı çizginin yukarı ve aşağı yönündeki maksimum ve minimum eğimleri belirlenmelidir. Başka bir deyişle, bir yol projesi maksimum ve minimum eğimlere göre tasarlanıp ve proje boyunca kırmızı çizginin eğimi bu sınırları geçmemelidir. Tasarlanan yazılım tarafından maksimum ve minimum eğimler tasarlayıcıdan sorulacaktır. Bu çalışma kapsamında projenin maksimum eğimi için %8 ve minimum eğim içinde %0,5 kabul edilerek kırmızı çizgi tasarımı belirlenen eğim sınırları arasında gerçekleşmiştir. Buna bağlı olarak da eğim sınırları dışında kalan kırmızı çizgi parçaları devre dışı bırakılarak hesaplamalara dahil edilmediği gözlenmiştir.
- Boykesit optimizasyonu yapılırken hazırlanan yazılım sayesinde kırmızı çizginin her doğru parçasının başlangıç ve bitiş noktası siyah çizgiye bağlanarak kapalı alanların oluşturması için zaruri bir şarttır. Some noktaları yazılım sayesinde devamlı yer değiştirmelerine göre, bu noktaların siyah çizgi üzerinde, farklı yerlere denk gelecekleri yadsınamaz bir gerçektir. Bu durumu özetleyeceksek her some noktası için yazılım tarafından farklı siyah kot bulunmaktadır. Bu bağlamda yol boyunca farklı some noktaları için siyah kotun doğru bir şekilde hesaplanması için gereken formül üretilmiş ve siyah kotların hesaplanması tam olarak gerçekleşmiştir.
- Boykesit optimizasyonu yapılırken hazırlanan yazılım sayesinde some noktaların her yer değişiminde kırmızı çizgi ile siyah çizgi arasında oluşan kapalı yarma-dolgu bölgelerinin alanları hesaplanmıştır. Some noktalarının yazılım tarafından devamlı yer değiştirmelerine göre farklı yarma dolgu bölgeleri oluşmaktadır. Bu farklı bölgeler için yazılım tarafından her seferinde yol boyunca farklı alan hesaplanmasının yapıldığı gözükmemektedir.
- Tek-amaçlı optimizasyon algoritmalarından *PSO* yöntemi sadece *Y* ekseninde bir aralık verilerek 30 kez denenmiştir. Bu defalarca deneme

sonucuna göre *PSO* yöntemi ortalama yarma-dolgu farkını neredeyse sıfır denilecek kadar bir küçük değer bulana dek sürdürmüştür.

- Tek-amaçlı optimizasyon algoritmalarından *FA* yöntemi sadece *Y* eksenini üzerinin bir aralık verilerek 30 kez denenmiştir. Bu defalarca deneme sonucuna göre *FA* yöntemi ortalama yarma-dolgu farkı 0.075 olarak hesaplanmıştır.
- Tek-amaçlı optimizasyon algoritmalarından *ABC* yöntemi sadece *Y* eksenini üzerinin bir aralık verilerek 30 kez denenmiştir. Bu defalarca deneme sonucuna göre *ABC* yöntemi ortalama yarma-dolgu farkı 80 birim olarak hesaplanmıştır.
- *PSO* yöntemi ile *Y* ekseninde optimizasyon yapıldığında, birinci kırmızı çizgi geçirildiği zaman yarma-dolgu miktarları arasındaki fark çok fazla ve yaklaşık 10000 birim iken 200'üncü iterasyondan itibaren bu fark neredeyse sıfıra yakın bir değer bulunmaktadır. Bu yöntem ile tasarlanan kırmızı çizgi çok hızlı ve etkin bir şekilde sonuçlar elde etmemizi sağlamaktadır. Bu optimizasyon yöntemine göre bulunan her kırmızı çizgide yarma-dolgu farkı bir öncekine göre daha azdır. Başka bir deyişle bulunan her kırmızı çizgi bir öncekine göre daha ekonomiktir.
- *FA* yöntemi ile *Y* ekseninde optimizasyon yapıldığında, birinci kırmızı çizgi geçirildiği zaman yarma-dolgu miktarları arasındaki fark yaklaşık 1000 birim olarak bulunmaktadır. Ancak, 135'inci iterasyondan itibaren bu fark ortalama olarak 0.1 birime yakın bir değere kadar düşmektedir. Bu yöntem ile tasarlanan kırmızı çizgi çok hızlı ve etkin bir şekilde sonuçlar elde etmemizi sağlamaktadır. Bu optimizasyon yöntemine göre diyagramda gözüktüğü gibi bulunan her kırmızı çizgide yarma-dolgu farkı bir öncekine göre daha az ve hatta büyük bir kısımda tekrarlanarak aynı değerleri bulmaktadır. Başka bir deyişle bulunan her kırmızı çizgi bir öncekine göre daha ekonomiktir.
- *ABC* yöntemi ile *Y* ekseninde optimizasyon yapıldığında, birinci kırmızı çizgi geçirildiği zaman yarma-dolgu miktarları arasındaki fark diğer iki yönteminin ilk denemesi kadarda fazla olmasa bile 1000 birimin altında bulunmaktadır. Ancak, 80'inci iterasyondan itibaren bu fark ortalama 80 birime yakın bir değere kadar düşmektedir. Bu yöntem ile tasarlanan kırmızı çizgi diyagrama göre çok hızlı ve etkin bir şekilde sonuçlar elde etmemizi sağlamaktadır. Bu optimizasyon yöntemine göre diyagramda gözüktüğü gibi bulunan her kırmızı çizgide yarma-dolgu farkı bir öncekine göre daha az ve

hatta büyük bir kısımda tekrarlanarak aynı değerleri bulmaktadır. Başka bir deyişle bulunan her kırmızı çizgi bir öncekine göre daha ekonomiktir. Ancak, bu yöntemin diyagramında görüldüğü gibi her ne kadarda en son bulunan kırmızı çizgi değerleri sıfırdan fazla olsa da ama diğer iki yöntemle göre daha hızlı sabit bir değere ulaşmaktadır.

- Tek-amaçlı optimizasyon algoritmalarından *PSO* yöntemi X ve Y ekseninde ayrı ayrı bir aralık verilerek 30 kez denenmiştir. Bu defalarca deneme sonucuna göre *PSO* yöntemi ortalama yarma-dolgu farkını neredeyse sıfır denilecek kadar bir küçük değer bulana dek sürdürmüştür.
- Tek-amaçlı optimizasyon algoritmalarından *FA* yöntemi X ve Y ekseninde ayrı ayrı bir aralık verilerek 30 kez denenmiştir. Bu defalarca deneme sonucuna göre *FA* yöntemi ortalama yarma-dolgu farkı 0.091 olarak hesaplanmıştır.
- Tek-amaçlı optimizasyon algoritmalarından *ABC* yöntemi X ve Y ekseninde ayrı ayrı bir aralık verilerek 30 kez denenmiştir. Bu defalarca deneme sonucuna göre *ABC* yöntemi ortalama yarma-dolgu farkı 88 olarak hesaplanmıştır.
- *PSO* yöntemi ile X ve Y ekseninde optimizasyon yapıldığında, birinci kırmızı çizgi geçirildiği zaman yarma-dolgu miktarları arasındaki fark çok fazla ve yaklaşık 10000 birim iken 215'inci iterasyondan itibaren bu fark neredeyse sıfıra yakın bir değer bulunmaktadır. Bu yöntem ile tasarlanan kırmızı çizgi diyagrama göre çok hızlı ve etkin bir şekilde sonuçlar elde etmemizi sağlamaktadır. Bu optimizasyon yöntemine göre bulunan her kırmızı çizgide yarma-dolgu farkı bir öncekine göre daha azdır. Başka bir deyişle bulunan her kırmızı çizgi bir öncekine göre daha ekonomiktir.
- *FA* yöntemi ile X ve Y ekseninde optimizasyon yapıldığında, birinci kırmızı çizgi geçirildiği zaman yarma-dolgu miktarları arasındaki fark *PSO* yönteminin ilk denemesi kadarda fazla olmasa bile yaklaşık 8000 birim olarak bulunmaktadır. Ancak, 105'inci iterasyondan itibaren bu fark ortalama olarak 0.09 birime yakın bir değere kadar düşmektedir. Bu yöntem ile tasarlanan kırmızı çizgi çok hızlı ve etkin bir şekilde sonuçlar elde etmemizi sağlamaktadır. Bu optimizasyon yöntemine göre bulunan her kırmızı çizgide yarma-dolgu farkı bir öncekine göre daha az ve hatta büyük bir kısımda

tekrarlanarak aynı değerleri bulmaktadır. Başka bir deyişle bulunan her kırmızı çizgi bir öncekine göre daha ekonomiktir.

- *ABC* yöntemi ile *X* ve *Y* eksenini üzerinde optimizasyon yapıldığında, birinci kırmızı çizgi geçirildiği zaman yarma-dolgu miktarları arasındaki fark diğer iki yönteminin ilk denemesi kadarda fazla olmasa bile 5000 birimin altında bulunmaktadır. Ancak, 85'inci iterasyondan itibaren bu fark ortalama 98 birime yakın bir değere kadar düşmektedir. Bu yöntem ile tasarlanan kırmızı çizgi diyagrama göre çok hızlı ve etkin bir şekilde sonuçlar elde etmemizi sağlamaktadır. Bu optimizasyon yöntemine göre diyagramda gözüktüğü gibi bulunan her kırmızı çizgide yarma-dolgu farkı bir öncekine göre daha az ve hatta büyük bir kısımda tekrarlanarak aynı değerleri bulmaktadır. Başka bir deyişle bulunan her kırmızı çizgi bir öncekine göre daha ekonomiktir. Ancak, bu yöntemin diyagramında görüldüğü gibi her ne kadarda en son bulunan kırmızı çizgi değerleri sıfırdan fazla olsa da ama diğer iki yönteme göre daha hızlı ve hatta 10'uncü deneme yapıldığında bile yarma-dolgu farkını düşürerek sabit bir değere ulaşmaktadır. Başka bir deyişle, bu yönteme göre her ne kadar da en son bulunan kırmızı çizgi ekonomik olmasa da elde etmek istediğimiz kırmızı çizgiye daha hızlı ve erken ulaşmamız gözlenmektedir.
- *PSO* yöntemi ile tasarlanan kırmızı çizgide toprak işleri dengelemesi gayet iyi sonuçlanmaktadır. Başka bir deyişle, bu yöntem kullanıldığında, toplam yarma-dolgu farkı neredeyse sıfır denilecek kadar bir değere ulaşmaktadır. Ama *FA* yönteminde bu fark sıfır ile bir rakamları arasında değişmektedir ve *ABC* yönteminde ise ortalama 90 civarında bir fark ile değişmektedir. Bu sayısal verilere göre bu çalışmada *PSO* değerleri daha çok güvenilir ve istikrarlı görünmektedir.
- *PSO* yöntemi ile elde edilen verilere göre *Y* eksenini ve *X-Y* eksenini üzerindeki incelemelerin sonuçları neredeyse birbirlerine çok yakındır. Ancak, kullandığımız üç farklı tek amaçlı meta-sezgisel algoritmalar ayrıca, *Y* eksenini ve *X-Y* eksenini üzerinde ayrı ayrı denenmesine göre demek ki toplamda 6 kez bu denenme tekrarlanmıştır. Bu bağlamda, her deneme 2000 iterasyonla yapıldığına göre bu yöntemler arasından *ABC* yöntemi diğer iki yönteme göre her ne kadar da ekonomik kırmızı çizgi bulamıyor olsa da diğer yöntemlere göre daha hızlı sonuca ulaşması gözlenmektedir. Buna bağlı olarak görüldüğü gibi optimizasyon yapıldığında ekonomik olması ve hızlı sonuca ulaşma

konusu iki farklı konu olarak incelenmektedir. Ancak, bu çalışma yapıldığında süre bakımından tüm yöntemler yaklaşık aynı süre içinde sonuçlanmıştır. Buda bize daha ekonomik olma seçeneğini daha ön plana çıkartmıştır.

- *NSGA3* yöntemi X ve Y ekseninde 30 kez tekrarlanmıştır. Bu çalışmada arazi verileri iki parça haline getirilerek incelemesi yapılmıştır. Bu sayede yarma-dolgu alanlarının dengelemesi için her parçaya ait ayrı bir değer bulunduğundan bu değerlerin toplamı alınarak denetim sonlandırılmıştır. Bu sonuçlar kapsamında *NSGA3* yöntemi için ortalama alanlar farkı 1.41 birim gözükmektedir.
- *MOPSO* yöntemi X ve Y ekseninde 30 kez tekrarlanmıştır. Bu çalışmada arazi verileri iki parça haline getirilerek incelemesi yapılmıştır. Bu sayede yarma-dolgu alanlarının dengelemesi için her parçaya ait ayrı bir değer bulunduğundan bu değerlerin toplamı alınarak denetim sonlandırılmıştır. Bu sonuçlar kapsamında *MOPSO* yöntemi için ortalama alanlar farkı 113.85 birim gözükmektedir.
- *SPEA2* yöntemi X ve Y ekseninde 30 kez tekrarlanmıştır. Bu çalışmada arazi verileri iki parça haline getirilerek incelemesi yapılmıştır. Bu sayede yarma-dolgu alanlarının dengelemesi için her parçaya ait ayrı bir değer bulunduğundan bu değerlerin toplamı alınarak denetim sonlandırılmıştır. Bu sonuçlar kapsamında *SPEA2* yöntemi için ortalama alanlar farkı 2.57 birim gözükmektedir.
- Örnek olarak yapılmış olan bu kodlamada önce kırmızı çizgiye farazi olarak bir koordinat verilmiştir. Daha sonra bu kırmızı çizgiye ait doğru parçaların optimizasyonu gerçekleşmiştir. Ancak, bu doğru parçaların arasına tepe ve dere düşey kurplar eklenmelidir. Bu bağlamda, düşey kurba uzunluğu yazılıma girilmelidir. Bu veriler ile düşey kurp hesaplama işlemleri, yazılım tarafından her ne kadar da karmaşık işlemlerden oluşsa bile çok hızlı bir şekilde gerçekleşmiştir. Böylece, tape düşey kurbalarda some noktalarına en yakın olan düşey kurba seçilmelidir ve aynı şekilde dere düşey kurp bulunan yerlerde some noktasına en yakın olan düşey kurp en optimum düşey kurp olarak yazılım tarafından kabul edilecektir.

KAYNAKLAR

- AASHTO. (1994). A Policy on Geometric Design of Highways and Streets, American Association of State Highway and Transportation Officials.
- Al-Sobky, A. S. (2014). An optimization approach for highway vertical alignment using the earthwork balance condition. *World Applied Sciences Journal*, 29(7), 884-891.
- Babapour, R., Naghdi, R., Ghajar, I., & Mortazavi, Z. (2018). Forest road profile optimization using meta-heuristic techniques. *Applied Soft Computing*, 64, 126-137.
- Baykal, O. (2009). Mühendislik Ölçmeleri 1: Kara ve demir Yollarında Geçki Geometrisi Tasarımı ve Aplikasyonu: cilt 2 Sayısal örnekler. Birsan Yayınevi.
- Behbahani, H. (2011). Tarhe Hendsiye Rah, 17. Baskı, Tahrn: Markaze Naşre Daneşgahi.
- Bıçakçı, O. (1997). Demiryollarında Yeni Bir Birleştirme Eğrisinin Araştırılması. Basılmamış Doktora Tezi, İ.T.Ü. İnşaat Fakültesi, İstanbul.
- Bosurgi, G., Pellegrino, O., & Sollazzo, G. (2013). A PSO highway alignment optimization algorithm considering environmental constraints. *Advances in Transportation Studies*, (31).
- Bruce, A. G., & Clarkeson, J. (1950). Highway design and construction. International Textbook.
- Casal, G., Santamarina, D., & Vázquez-Méndez, M. E. (2017). Optimization of horizontal alignment geometry in road design and reconstruction. *Transportation Research Part C: Emerging Technologies*, 74, 261-274.
- Ceylan, A. (1993). Presizyonlu Nivelman Yerine Trigonometric Nivelman Yönteminin Kullanılabilirliği Üzerine Bir Çalışma. SÜ Fen Bil. Enst.
- Ceylan, A., & Baykal, O. (2006). Precise height determination using leap-frog trigonometric leveling. *Journal of surveying engineering*, 132(3), 118-123.
- Ceylan, A., & Baykal, O. (2008). Precise height determination using simultaneous-reciprocal trigonometric levelling. *Survey Review*, 40(308), 195-205.
- Chew, E. P., Goh, C. J., & Fwa, T. F. (1989). Simultaneous optimization of horizontal and vertical alignments for highways. *Transportation Research Part B: Methodological*, 23(5), 315-329.
- Coello, C. A. C., Lamont, G. B., & Van Veldhuizen, D. A. (2007). Evolutionary algorithms for solving multi-objective problems (Vol. 5, pp. 79-104). New York: Springer.
- 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.
- Coello, C., & Coello, C. A. (2001, March). A short tutorial on evolutionary multiobjective optimization. In *International Conference on evolutionary multi-criterion optimization* (pp. 21-40). Springer, Berlin, Heidelberg.
- De Oca, M. A. M., Stutzle, T., Birattari, M., & Dorigo, M. (2009). Frankenstein's PSO: a composite particle swarm optimization algorithm. *IEEE Transactions on Evolutionary Computation*, 13(5), 1120-1132.
- Deb, K. (1999). Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary computation*, 7(3), 205-230.
- Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. A. M. T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, 6(2), 182-197.

- Drake, J. (1967). Taschenbuch für Vermessungsingenieure. Verlag für Bauwesen.
- Easa, S. M. (1988). Selection of roadway grades that minimize earthwork cost using linear programming. *Transportation Research Part A: General*, 22(2), 121-136.
- Engelbrecht, A. P. (2007). Computational intelligence: an introduction. John Wiley & Sons.
- Evren, G. (1994). Toprak İşleri. İstanbul: İ.T.Ü. Rektörlüğü, İ.T.Ü. İnşaat Fakültesi Matbaası.
- Fwa, T. F. (1989). Highway vertical alignment analysis by dynamic programming. *Transportation Research Record*, 1239(1-9), 2-3.
- Fwa, T. F., Chan, W. T., & Sim, Y. P. (2002). Optimal vertical alignment analysis for highway design. *Journal of transportation engineering*, 128(5), 395-402.
- Ghanizadeh, A. R., & Heidarabadizadeh, N. (2018). Optimization of vertical alignment of highways in terms of earthwork cost using colliding bodies optimization algorithm. *دانشگاه علم و صنعت ایران*, 8(4), 657-674.
- Ghanizadeh, A. R., Heidarabadizadeh, N., & Mahmoodabadi, M. J. (2020). Effect of objective function on the optimization of highway vertical alignment by means of Metaheuristic algorithms. *Civil Engineering Infrastructures Journal*, 53(1), 115-136.
- Goh, C. J., Chew, E. P., & Fwa, T. F. (1988). Discrete and continuous models for computation of optimal vertical highway alignment. *Transportation Research Part B: Methodological*, 22(6), 399-409.
- Goktepe, A. B., & Lav, A. H. (2003). Method for balancing cut-fill and minimizing the amount of earthwork in the geometric design of highways. *Journal of transportation engineering*, 129(5), 564-571.
- Goktepe, A. B., & Lav, A. H. (2004). Method for optimizing earthwork considering soil properties in the geometric design of highways. *Journal of surveying engineering*, 130(4), 183-190.
- Göktepe, A. B., Altun, S., & Ahmedzade, P. (2009). Optimization of vertical alignment of highways utilizing discrete dynamic programming and weighted ground line. *Turkish Journal of Engineering and Environmental Sciences*, 33(2), 105-116.
- Goktepe, A. B., Lav, A. H., & Altun, S. (2005). Dynamic optimization algorithm for vertical alignment of highways. *Mathematical and Computational Applications*, 10(3), 341-350.
- Hassan, R., Cohanım, B., De Weck, O., & Venter, G. (2005, April). A comparison of particle swarm optimization and the genetic algorithm. In 46th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics and materials conference (p. 1897).
- Hickerson, T. F. (1959). Route surveys and design. McGraw-Hill.
- Jha, M. K., & Schonfeld, P. (2004). A highway alignment optimization model using geographic information systems. *Transportation Research Part A: Policy and Practice*, 38(6), 455-481.
- Kang, M. W., Jha, M. K., & Schonfeld, P. (2012). Applicability of highway alignment optimization models. *Transportation Research Part C: Emerging Technologies*, 21(1), 257-286.
- Karaboga, D., Akay, B., & Ozturk, C. (2007, August). Artificial bee colony (ABC) optimization algorithm for training feed-forward neural networks. In *International conference on modeling decisions for artificial intelligence* (pp. 318-329). Springer, Berlin, Heidelberg.
- 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.

- Kennedy, J., & Eberhart, R. (1995, November). Particle swarm optimization. In Proceedings of ICNN'95-international conference on neural networks (Vol. 4, pp. 1942-1948). IEEE.
- KGM. (2005). Proje ve Mühendislik Hizmetleri Kriter Raporları. Ankara: Karayolları Genel Müdürlüğü Otoyolları Dairesi Başkanlığı.
- KGM. (2002). Karayolları Ön ve Kesin Projeleri Mühendislik Hizmetleri Teknik Şartnamesi. Ankara: Karayolları Genel Müdürlüğü.
- KGM. (2005). Otoyolu Proje ve Mühendislik Hizmetleri Teknik Şartnamesi (Ön ve Kesin Proje). Ankara: Karayolları Genel Müdürlüğü Otoyolları Dairesi Başkanlığı.
- Kim, E., Jha, M. K., & Son, B. (2005). Improving the computational efficiency of highway alignment optimization models through a stepwise genetic algorithms approach. *Transportation Research Part B: Methodological*, 39(4), 339-360.
- Kim, E., Jha, M. K., Schonfeld, P., & Kim, H. S. (2007). Highway alignment optimization incorporating bridges and tunnels. *Journal of Transportation Engineering*, 133(2), 71-81.
- Kiper, T. (1988). Proje Mühendisleri için Karayolu Geometrik Standartları Esasları. Karayolları Genel Müdürlüğü, ANKARA.
- Kumbasar, V., Kumbasar, F., & Önalp, A. (1969). Yol Mühendisleri için Zemin Mekaniği. İTÜ İnşaat Fakültesi, İstanbul.
- Manns, K. (1985). Querbeseleunigung und Querruck in der Übergangsbogenbemessung. Universität Darmstadt.
- Meyer, F.C. (1968). Route Surveying. International Textbook Comp.
- Monejem, S. (2005). Rahsazi. 2.Baskı. Tehran: Angize Basımevi.
- Moreb, A. A. (1996). Linear programming model for finding optimal roadway grades that minimize earthwork cost. *European Journal of Operational Research*, 93(1), 148-154.
- Müller, G. (1988). Ingenieur-Geodäsie: Verkehrsbau. Strassenbau. Verlag für Bauwesen.
- Nicholson, A. (1988). Superelevation, Side Friction and Road Consistency, *ASCE Journal of Transportation Engineering*, 411-418.
- Nostrand, V. (1992). Highway. Usa: an Architectural Approach. Reinhold.
- Oglesby, H.C. and Hicks, R.G. (1984). Highway Engineering. John Wiley Inc.
- Özkan, E. (2013). Optimization of highway vertical alignment by direct search technique (Master's thesis, Middle East Technical University).
- Ozkan, E., Tuydes-Yaman, H., & Acar, S. O. (2021, February). Vertical alignment optimisation in highway design by means of mesh adaptive direct search. In Proceedings of the Institution of Civil Engineers-Transport (pp. 1-12). Thomas Telford Ltd.
- Rao, S. S. (2019). Engineering optimization: theory and practice. John Wiley & Sons.
- Reynolds, C. W. (1987, August). Flocks, herds and schools: A distributed behavioral model. In Proceedings of the 14th annual conference on Computer graphics and interactive techniques (pp. 25-34).
- Schofield, W. (2001). Engineering surveying: theory and examination problems for students. Elsevier.
- Shepherd, F. A. (1983). Engineering surveying: problems and solutions. Arnold.
- Sütaş, İ., ve Öztaş, G. (1983). Karayolu inşaatında uygulama ve projelendirme. Teknik Kitaplar Yayınevi.

- Tunç, A. (2004). Yol Mühendisliğinde Geoteknik ve Uygulamaları, Atlas Yayın - Dağıtım.
- Tyler G.H.. (1987). Civil Engineering Calculation Refrence Guide. New York: Mcgraw-Hill Book.
- Umar, F., ve Yayla, N. (1997). Yol İnşaatı, V. Baskı. İstanbul: İ.T.Ü. Rektörlüğü.
- Uren, J., & Price, W. F. (2006). Surveying for engineers: Traversing and coordinate calculations. Hampshire, United Kingdom: Macmillian.
- Wright, P. H., & Ashford, N. J. (1989). Transportation engineering, Planning and Design. 3rd Edition, Wiley, New York.
- Wright, P.H. and Paquette, R.J. (1985). Highway Engineering. Fifth Edition. New York, Usa: John Wiley and Sons.
- Yang, X. S., & Press, L. (2010). Nature-inspired metaheuristic algorithms second edition.
- Yelghi, A., & Köse, C. (2018). A modified firefly algorithm for global minimum optimization. Applied Soft Computing, 62, 29-44.
- Yelghi, A., & Tavangari, S. (2022). Features of metaheuristic algorithm for integration with ANFIS model (No. 7594). EasyChair.
- YELGHİ, A., GÜRSOY, M., & YELGHİ, A. (2021). Bankacılık piyasasında kredi türüne göre belirlenen faiz oranlarının enflasyon ve döviz kuru ilişkisi. Journal of Empirical Economics and Social Sciences, 3(1), 21-42.
- Zitzler, E. (1999). Evolutionary algorithms for multiobjective optimization: Methods and applications (Vol. 63). Ithaca: Shaker.
- Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. Evolutionary computation, 8(2), 173-195.

EKLER

PSO

% Publisher: Yarpiz (www.yarpiz.com)

%

% Developer: S. Mostapha Kalami Heris (Member of Yarpiz Team)

%

% Contact Info: sm.kalami@gmail.com, info@yarpiz.com

%

clc;

clear;

close all;

%% Problem Definition

for kol=1:1

nVar=6; % Number of Decision Variables

VarSize=[1 nVar]; % Size of Decision Variables Matrix

% % x=[4757.065 4767.065 4866.538 4890.310 4892 4990 4996.062 5106.062]';

% % y=[3387.466 3431.496 3438.321 3549.419 3580 3590 3549.419 3662.887]';

% %

% % x2=[5146.062 5176.062 5196.062 5206.062 5220.062 5250.062];

% % y2=[3431.496 3438.321 3549.419 3580 3590 3549.419 3662.887];

% % % x=flipr(x);

% % y2=flipr(y2);

% % x2=x2';

% % y2=y2';

% % x=[x;x2];

% % y=[y;y2];

%

x=[4757;4767;4866;4890;4892;4990;4996;5106;5146;5176;5196;5206;5220;5250];

%

y=[3387;3431;3438;3549;3580;3590;3549;3662;3595;3590;3580;3549;3438;3431];

%

% %%%%%%%%%%%%%%% finde peaks

%%%%%%%%%%%%%%%

% peaks=[5106 3662];

x=[100;110;209;233;235;333;339;449;489;519;539;549;563;593];

y=[100;144;151;262;293;303;262;375;308;303;293;262;151;144];

%%%%%%%%%%%%%%%

finde

peaks

%%%%%%%%%%%%%%%

% peaks=[5106 3662];

peaks=[449 375];

cutterpeak_idx=8;

%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%

VarMin(1)=y(1)-20; % Lower Bound of Variables

VarMax(1)=y(1)+20; % Upper Bound of Variables

```

VarMin(2)=y(cutterpeak_idx)-20;    % Lower Bound of Variables
VarMax(2)=y(cutterpeak_idx)+20;    % Upper Bound of Variables
VarMin(3)=y(end)-20;              % Lower Bound of Variables
VarMax(3)=y(end)+20;              % Upper Bound of Variables

VarMin(4)=x(1)-5;                 % Lower Bound of Variables
VarMax(4)=x(1)+5;                 % Upper Bound of Variables
VarMin(5)=x(cutterpeak_idx)-5;    % Lower Bound of Variables
VarMax(5)=x(cutterpeak_idx)+5;    % Upper Bound of Variables
VarMin(6)=x(end)-5;               % Lower Bound of Variables
VarMax(6)=x(end)+5;               % Upper Bound of Variables

%                               CostFunction=@(x,y,yvector,peaks,cutterpeak_idx)
Linezoptimum(x,y,yvector,peaks,cutterpeak_idx);    % Cost Function
%% Generate Range
%% PSO Parameters

global MaxIt;    % Maximum Number of Iterations
MaxIt=2000;
nPop=5;          % Population Size (Swarm Size)

% PSO Parameters
w=1;             % Inertia Weight
wdamp=0.99;      % Inertia Weight Damping Ratio
c1=1.5;          % Personal Learning Coefficient
c2=2.0;          % Global Learning Coefficient

% If you would like to use Constriction Coefficients for PSO,
% uncomment the following block and comment the above set of parameters.

% % Constriction Coefficients
% phi1=2.05;
% phi2=2.05;
% phi=phi1+phi2;
% chi=2/(phi-2+sqrt(phi^2-4*phi));
% w=chi;         % Inertia Weight
% wdamp=1;       % Inertia Weight Damping Ratio
% c1=chi*phi1;   % Personal Learning Coefficient
% c2=chi*phi2;   % Global Learning Coefficient

% Velocity Limits
VelMax=0.1*(VarMax-VarMin);
VelMin=-VelMax;

%% Initialization

empty_particle.Position=[];
empty_particle.Cost=[];
empty_particle.Velocity=[];
empty_particle.Best.Position=[];

```

```

empty_particle.Best.Cost=[];

particle=repmat(empty_particle,nPop,1);

GlobalBest.Cost=inf;

for i=1:nPop

    % Initialize Position
    particle(i).Position=unifrnd(VarMin,VarMax,VarSize);
    particle(i).zh=0;
    % Initialize Velocity
    particle(i).Velocity=zeros(VarSize);

    % Evaluation
    [particle(i).Cost
    Linezoptimum(x,y,particle(i).Position,peaks,cutterpeak_idx);
    particle(i).zh]=

    % Update Personal Best
    particle(i).Best.Position=particle(i).Position;
    particle(i).Best.Cost=particle(i).Cost;

    % Update Global Best
    if particle(i).Best.Cost<GlobalBest.Cost

        GlobalBest=particle(i).Best;

    end

end

end

BestCost=zeros(MaxIt,1);

%% PSO Main Loop
it=0;
it2=1;
% for it=1:MaxIt
while it<MaxIt
    for i=1:nPop

        % Update Velocity
        particle(i).Velocity = w*particle(i).Velocity ...
            +c1*rand(VarSize).*(particle(i).Best.Position-particle(i).Position) ...

        % Apply Velocity Limits
        particle(i).Velocity = max(particle(i).Velocity,VelMin);
        particle(i).Velocity = min(particle(i).Velocity,VelMax);

        % Update Position
        particle(i).Position = particle(i).Position + particle(i).Velocity;

```

```

% Velocity Mirror Effect
IsOutside=(particle(i).Position<VarMin | particle(i).Position>VarMax);
particle(i).Velocity(IsOutside)=-particle(i).Velocity(IsOutside);

% Apply Position Limits
particle(i).Position = max(particle(i).Position,VarMin);
particle(i).Position = min(particle(i).Position,VarMax);

% Evaluation
[particle(i).Cost particle(i).zh]=
it=it+1;
% Update Personal Best
if particle(i).Cost<particle(i).Best.Cost

particle(i).Best.Position=particle(i).Position;
particle(i).Best.Cost=particle(i).Cost;

% Update Global Best
if particle(i).Best.Cost<GlobalBest.Cost

GlobalBest=particle(i).Best;

end

end

end

end

BestCost(it2)=GlobalBest.Cost;
disp(['Iteration ' num2str(it2) ': Best Cost = ' num2str(BestCost(it2))]);

end

BestSol = GlobalBest;

%% Results
% close all
figure(2);
% close all
% hold on
semilogy(BestCost,'LineWidth',2);
xlabel('Iteration');
ylabel('Best Cost');
grid on;
figure(3)
% close all
[GlobalBest.Cost GlobalBest.zh] =

```

```
save(['PSO' num2str(kol) '.mat']);
end
```

INPUT PEAK

```
function user_peak= input_peak()
prompt = 'peak Y/N: ';
yesno = input(prompt, "s");
if yesno=='y'
number_peak = input('please input the number of peaks:', "s");
number_peak=str2num(number_peak);
user_peak=[];
for i=1:number_peak
number_peakx=str2num(number_peak);
user_peak(i,1)=number_peakx;
prompt = 'please input y axis: ';
user_peak(i,2)=number_peaky;
end
else
user_peak=[];
end
end
```

LINEOPTIMUM

```
function [z z0 aa]=lineoptimum(x,y,yvector,xvector)
global MaxIt;
aa=zeros(20,2);
z0=0;
yg=y;
yg(1)=yvector(1);
yg(end)=yvector(2);

x(1)=xvector(1);
x(end)=xvector(end);
x=sort(x);
xa=min(x):0.01:max(x); %%%%%%%%%%xa
t=numel(xa);
p1=abs(yvector(2))-abs(yvector(1));
p=p1/(t-1);
yg2=yvector(1):p:yvector(2);
yg2=yg2';
line_small(i,:)=polyfit(x(i:i+1), y(i:i+1),1);
end
% line_intersect=zeros(n-1,2);
line_intersect=[];
xa=min(x):0.01:max(x); %%%%%%%%%%xa
ya=xa*a(:,1)+a(:,2);
itt=1;
for i=1:n-1
b_line = polyfit(x(i:i+1), y(i:i+1),1); % Fit 'line'
```

```

ui=find(xa>=x(i));
uii=find(xa<=x(i+1));
inter=intersect(ui,uii);
y_line2 = polyval(b_line, xa(inter)) ;
% Evaluate 'line' At 'x_curve'
hold on
x_int = interp1((y_line2-ya(inter)), xa(inter), 0); % X-Value At Intercept
y_int = polyval(b_line,x_int);% Y-Value At Intercept

% ~(isnan(a))
if (~(isnan(x_int))& ~(isnan(y_int)))
    if ((x_int~=0)& (y_int~=0))
        line_intersect(itt,1) =x_int;
        line_intersect(itt,2)=y_int;
        itt=itt+1;
    end
end
end

% line_intersect(1,:)=[]; % hazf noghteye aval
gModel=min(x):0.01:max(x);
%=====outline and inline=====
if ((all(isnan(line_intersect(:,1))~=1)) && (all(isnan(line_intersect(:,2))~=1)))
    tModelled=polyval(a,gModel);
    plot(gModel,tModelled,'y-','LineWidth',2);
    hold on
    z=500000
    for i=1:n

        plot(line_intersect(i,1) ,line_intersect(i,2) , 'r*');
    end
else
    tModelled=polyval(a,gModel);
    plot(gModel,tModelled,'r-','LineWidth',2);
    hold on
    %%%%%%%%%%%%%% polygon build
    closed%%%%%%%%%%%%%

x=[gModel(1); x];
x=[x ;gModel(end)];
x=sort(x);

y=[tModelled(1); y];
y=[y ;tModelled(end)];

line_intersect=[[x(1) ya(1)]; line_intersect ;[x(end) ya(end)]];
n=numel(x);
ns=numel(line_intersect(:,1));
% line(x(i:i+1), y(i:i+1));
hold on;

```

```

line(x,y);
% for i=1:ns
plot(line_intersect(:,1),line_intersect(:,2),'r*');
% hold on
% end
cut=[];
fill=[];
ns=numel(line_intersect(:,1));
ff=1; ccc=1;
sw=0;

for i=2:n-1
    sw=0;
    for k=2:ns
        % plot(x(i),y(i),'c*')
        if (x(i)<=line_intersect(k,1) && sw==0)
            hold on
            li=a(1)*x(i)+a(2);
            if li<=y(i)

                else li>=y(i)
                    fill(ff,1)=x(i);
                    fill(ff,2)=y(i);
                    hold on
                    plot(fill(ff,1),fill(ff,2),'b*');
                    ff=ff+1;
                    sw=1;
                end
            end
        end
    end
    end
    %%%%%%%%%%%%%% categorize ploygon
    nsc=numel(cut(:,1));
    nsf=numel(fill(:,1));
    a=1:ns;
    evens=a(mod(a,2)==0);
    nc=even_ns;
    ncs=1;
    nf=odds_ns;
    nfs=2;
    end
    % cu=1;
    i=1;
    cc=1;
    fillg=[];
    cutg=[];
    for k=nfs:2:nf-1
        for s=1:ccc-1
            if ((line_intersect(k,1)<=cut(s,1)) && (cut(s,1)<=line_intersect(k+1,1)))

```

```

        [cutg]=[cutg;cut(s,:)];
    end
    end
    pol_low{cc}=[[line_intersect(k,1) line_intersect(k,2)];cutg;[line_intersect(k+1,1)
line_intersect(k+1,2)]];
    polarea_lo(cc)=polyarea(pol_low{1,cc}(:,1),pol_low{1,cc}(:,2));
% end
cc=cc+1;
% i=i+1;
cutg=[];
end
cc=1;
i=1;

    end
    end
    pol_upper{cc}=[[line_intersect(k,1)
line_intersect(k,2)];fillg;[line_intersect(k+1,1) line_intersect(k+1,2)]];
    polarea_up(cc)=polyarea(pol_upper{1,cc}(:,1),pol_upper{1,cc}(:,2));

cc=cc+1;
fillg=[];
% i=i+1;
end
lo=numel(pol_low);
up=numel(pol_upper);
% figure(ip+1)
hold on
for i=1:lo
polyin = polyshape(pol_low{1,i});
plot(polyin);
hold on
end

hold on
for i=1:up
polyin = polyshape(pol_upper{1,i});
plot(polyin);
hold on
end

%%%%%%%%%%%%%%calculate area
accum(ip)=(sum(polarea_lo)-sum(polarea_up));
z0=(accum);
z=min(abs(accum));
end
end

```

LINEZOPTIMUM

function [z zh] = Linezoptimum(x,y,xyvector,peaks,cutterpeak_idx)

```

z=0;z1=[];zh=0;
n=numel(peaks(:,1));
if n==0
z =lineoptimum(x,y,yvector);
else if (n==1)
N=2; % tedad khat
nc=1; %%%%%%%%% drost shavad
first=1;
for i=1:N

[z1
aa]=lineoptimum(x(first:cutterpeak_idx(nc)),y(first:cutterpeak_idx(nc)),xyvector(i:i+
1),xyvector(i+3:i+4));

z=z+z1;
zh=zh+z0;
first=cutterpeak_idx(nc);
cutterpeak_idx(nc)=14;

end
end
end

PEAK EARTH
function [data_peak line_peak]= peak_earth(real_data)
data_peak=[];

data=real_data;
local_max=islocalmax(data(2,:));
local_min=islocalmin(data(2,:));

hold on
plot(data(1,:),data(2,:),data(1,local_min),data(2,local_min),'b*');
data_peak_inx=[find(local_max==1) find(local_min==1)];
datapeak2dim(1,:)=data(1,data_peak_inx);
datapeak2dim(2,:)=data(2,data_peak_inx);
[~,ix]=sort(datapeak2dim(1,:)); % get the indices for sorting 2nd row
datapeak2dim=datapeak2dim(:,ix);

%%%%%%%%%%%%
[input_peak_user]=input_peak();
input_peak_user=input_peak_user';
%%%%%%%%%%%% if user added the peaks it will be show on the desktop
if (~isempty(input_peak_user))
datapeak_total=[datapeak2dim ,input_peak_user];
% datapeak_total=sortcols(datapeak_total);
[~,ix]=sort(datapeak_total(1,:)); % get the indices for sorting 2nd row
datapeak_total=datapeak_total(:,ix); % rearrange all by the column order
end

```

```

datapeak_total=[[data(1,1);data(2,1)] ,datapeak_total, [data(1,end);data(2,end)]];
% x_peak=[1 data_peak_inx numel(data)];
plot(datapeak_total(1,:),datapeak_total(2,:), 'r*');
hold on
line_peak=[];
n=numel(datapeak_total(1,:))-1;
for i=1:n
line_peak(i,:)=polyfit(datapeak_total(1,i:i+1), datapeak_total(2,i:i+1),1);
hold on;
line(datapeak_total(1,i:i+1),datapeak_total(2,i:i+1));
hold on;
end
data_peak=datapeak_total;
line_peak=line_peak;
end

```

SLOPE CONTROL

```

function [data_peak line_peak correct]= slope_control(data_peak,line_peak)

slope_valuemax = input('please input the max slope:', "s");
slope_valuemax=str2num(slope_valuemax);

slope_valuemin = input('please input the min slope:', "s");
slope_valuemin=str2num(slope_valuemin);
k=1;
correct=1;
while(correct)

    [data_peak line_peak
correct]=slope_controliter(data_peak,line_peak,slope_valuemin,slope_valuemax);
    for j=1:numel(line_peak(:,1))
        line_peak(j,:)=polyfit(data_peak(1,j:j+1), data_peak(2,j:j+1),1);
    end
    k=k+1
end
end

```

INPUT RANGE DELETE

```

function [user_range]= input_range_delete()
prompt = 'Do you want to remove the range data for peak Y/N: ';
yesno = input(prompt, "s");
if yesno=='y'
number_range = input('please input the number of Range:', "s");
number_range=str2num(number_range);
user_range=[];
for i=1:number_range
prompt = "please input x start "+num2str(i)+" : ";
number_range = input(prompt, "s");

```

```

number_rangex=str2num(number_range);
user_range(i,1)=number_rangex;
prompt = "please input x end "+num2str(i)+" :";
number_range = input(prompt, "s");
end
else
    user_range=[];
end
end
end

```

INPUT RANGE INSERT

```

function user_range_insert= input_range_insert()
prompt = 'Do you want to insert peak Y/N:';
yesno = input(prompt, "s");
if yesno=='y'
    number_range = input('Please input the insert number :', "s");
    number_range=str2num(number_range);
    user_range_insert=[];
    for i=1:number_range
        prompt = "please input x "+num2str(i)+" :";
        number_range = input(prompt, "s");
        number_rangex=str2num(number_range);
        user_range_insert(i,1)=number_rangex;
        prompt = "please input y "+num2str(i)+" :";
        number_range = input(prompt, "s");
        number_rangey=str2num(number_range);
        user_range_insert(i,2)=number_rangey;
    end
else
    user_range_insert=[];
end

end

```

ZIYAAFN

```

function [pointyhat,pointx] = ziyaafr(yy,xx,yx1,yx2)

pointx=yx2(2);
pointy=yx2(1);
[idxx1]=max(find(pointx>xx))
shekasteAx=xx(idxx1);
shekasteAy=yy(idxx1);
[idxx2 ]=min(find(pointx<xx));
shekasteBx=xx(idxx2);
shekasteBy=yy(idxx2);
pointyhat=((pointx*(shekasteBy-shekasteAy
% % plot(xx,yy)

```

```

% % hold on
% % plot(pointx,pointy,'-mo','MarkerEdgeColor','b','MarkerSize',20)
% % hold on
% % plot(pointx,pointyhat,'-mo','MarkerEdgeColor','k','MarkerSize',20)

```

End

LINEOPTIMUM

```

function [pointyhat,pointx] = ziyaaFn(yy,xx,yx1,yx2)

pointx=yx2(2);
pointy=yx2(1);
[idxx1]=max(find(pointx>xx))
shekasteAx=xx(idxx1);
shekasteAy=yy(idxx1);
[idxx2 ]=min(find(pointx<xx));
shekasteBx=xx(idxx2);
shekasteBy=yy(idxx2);
pointyhat=((pointx*(shekasteBy-shekasteAy

% % plot(xx,yy)
% % hold on
% % plot(pointx,pointy,'-mo','MarkerEdgeColor','b','MarkerSize',20)
% % hold on
% % plot(pointx,pointyhat,'-mo','MarkerEdgeColor','k','MarkerSize',20)

end

```

LINEOPTIMUM2

```

function [z aa xvector yvector old_yvector
old_xvector]=lineoptimum2(x,y,yvector,xvector,count,N,old_yvector,old_xvector)
global MaxIt;

aa=zeros(20,2);
if count==1
xvector(1)=0;
elseif count==N
xvector(end)=x(end);
else
yvector(1)=old_yvector;
xvector(1)=old_xvector;
end
xa=min(x):0.01:max(x); %%%%%%%%%%%xa
gModel=min(x):0.01:max(x);
t=numel(xa);
p1=abs(yvector(2))-abs(yvector(1));
p=p1/(t-1);
yg2=yvector(1):p:yvector(2);

```

```

yg2=yg2';
xa=xa';
aa(1,:)=polyfit(xa,yg2,1);

ip=1;
a=aa(ip,:);
n=numel(x);
line_small=zeros(n-1,2);

line_intersect=[];
xa=min(x):0.01:max(x); %%%%%%%%%%%xa
ya=xa*a(:,1)+a(:,2);
itt=1;
old_yvector=xvector(end);%%%%%%%%%%
old_xvector=xvector(end);

for i=1:n-1
    b_line = polyfit(x(i:i+1), y(i:i+1),1);    % Fit 'line'

    x_int = interp1((y_line2-ya(inter)), xa(inter), 0);    % X-Value At Intercept
    y_int = polyval(b_line,x_int);% Y-Value At Intercept

    % ~(isnan(a))
    if (~(isnan(x_int))& ~(isnan(y_int)))
        if ((x_int~=0)& (y_int~=0))
            line_intersect(itt,1)=x_int;
            line_intersect(itt,2)=y_int;
            itt=itt+1;
        end
    end
end

% line_intersect(1,:)=[]; % hazf noghteye aval

if (isempty(line_intersect) && (numel(x)==2))
    z=0;
    xvector=x;
    yvector=y;
    vyend=yvector(end);
    line(xvector(i:i+1), yvector(i:i+1));
    hold on

elseif (isempty(line_intersect) )
    sprintf('çizmeyi çal??iyor....');
    % pause(10)
    z=500000;

elseif ((all(isnan(line_intersect(:,1))==1)) && (all(isnan(line_intersect(:,2))==1)))
    tModelled=polyval(a,gModel);
    plot(gModel,tModelled,'y-','LineWidth',2);

```

```

hold on
z=600000;
for i=1:n
    line(x(i:i+1), y(i:i+1));
    hold on;
    line(x,y);
    plot(line_intersect(i,1) ,line_intersect(i,2) , 'r*');
end

else
vyend=line_intersect(end,2);

tModelled=polyval(a,gModel);
plot(gModel,tModelled,'r-','LineWidth',2);
hold on
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% polygon build
closed%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
x=[x(1); x];
y=[ya(1); y];
x=[x ;x(end)];
y=[y ;ya(end)];

line_intersect=[[x(1) ya(1)];
line_intersect ;[x(end) ya(end)]];

% for i=1:ns
plot(line_intersect(:,1) ,line_intersect(:,2) , 'r*');
% hold on
% end
cut=[];
fill=[];
ns=numel(line_intersect(:,1));
ff=1; ccc=1;
sw=0;

for i=2:n-1
    sw=0;
    for k=2:ns
        % plot(x(i) ,y(i) , 'c*')
        if ( x(i)<=line_intersect(k,1) && sw==0)
            hold on
            li=a(1)*x(i)+a(2);
            if li<=y(i)
                cut(ccc,1)=x(i);
                cut(ccc,2)=y(i);
                hold on
                plot(cut(ccc,1) ,cut(ccc,2) , 'g*')
                ccc=ccc+1;
                sw=1;
            end
        end
    end
end

```

```

end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% categorize ploygon
nsc=numel(cut(:,1));
nsf=numel(fill(:,1));
a=1:ns;
evens=a(mod(a,2)==0);
even_ns=evens(end);
odds=a(mod(a,2)~=0);
odds_ns=odds(end);
k=1;
if (line_intersect(k,2)<y(2))
nf=even_ns;
nfs=1;
nc=odds_ns;
ncs=2;
else
nc=even_ns;
ncs=1;
nf=odds_ns;
nfs=2;
end

% cu=1;
i=1;
cc=1;
cutg=[];
for k=nfs:2:nf-1
    for s=1:ccc-1
        if ((line_intersect(k,1)<=cut(s,1)) && (cut(s,1)<=line_intersect(k+1,1)))
            [cutg]= [cutg;cut(s,:)];
        end
    end
    pol_low{cc}=[line_intersect(k,1) line_intersect(k,2);cutg;line_intersect(k+1,1)
line_intersect(k+1,2)];
    polarea_lo(cc)=polyarea(pol_low{1,cc}(:,1),pol_low{1,cc}(:,2));
% end
cc=cc+1;
% i=i+1;
cutg=[];
end
cc=1;
i=1;
for k=ncs:2:nc-1
    for s=1:fff-1
        if ((line_intersect(k,1)<=fill(s,1)) && (fill(s,1)<=line_intersect(k+1,1)))
            [fillg]=[fillg;fill(s,:)];
        end
    end
end

```

```

    pol_upper{cc}=[line_intersect(k,1)
line_intersect(k,2)];fillg;[line_intersect(k+1,1) line_intersect(k+1,2)]];

cc=cc+1;
fillg=[];
% i=i+1;
end
lo=numel(pol_low);
up=numel(pol_upper);
figure(ip+1)
hold on

hold on
for i=1:up
polyin = polyshape(pol_upper{1,i});
plot(polyin)
hold on
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%calculate area
accum(ip)=(sum(polarea_lo)-sum(polarea_up));
z=min(abs(accum))
end
end

```

ÖZ GEÇMİŞ

Sina ASHERLOU, Khoy Behrouz Lisesi'ni bitirdikten sonra İran Azad Üniversitesi Mühendislik Fakültesi, İnşaat Mühendisliği Bölümünden 22.07.2010 tarihinde mezun oldu. 2016 yılında KTÜ İnşaat Mühendisliği Ana Bilim Dalı Yüksek Lisans programını bitirdi, 2017 yılında OMÜ İnşaat Mühendisliği doktora programına başladı. Mezuniyetinden bu yana farklı Metro ve Karayolu projelerinde çalışmış ve yaklaşık 6 senedir Avrasya Üniversitesinde Öğretim Görevlisi olarak görev yapan Sina ASHERLOU, iyi derecede İngilizce ve Farsça bilmektedir. Temel ilgi alanları, Ulaştırma ve Haritacılıktır.

İletişim Bilgileri

ORCID ID : 0000-0002-3950-2636

Yayınlar:

1. Asherlou, S., Yelghi, A., Pancar, E. B., & Oruç, Ş. (2022). An optimization approach for highway alignment using metaheuristic algorithms. *Meta-heuristic Optimization Techniques: Applications in Engineering*, 10, 11.
2. ASHERLOU, Sina and YELGHI, Aref and Burak PANCAR, Erhan and ORUÇ, Şeref, 2022, Highway Algorithm with Slope Parameters, Ground Line Coordinates and Peak Points, Fourth International Conference and Fifth National Conference on Civil Engineering, Architecture, Art and Urban Design, Tabriz, <https://civilica.com/doc/1427833>
3. Asherlou, S., Yelghi, A., ve Pancar, E. B. (2022). Çok Amaçlı Meta-Sezgisel Algoritma Kullanarak Karayolu Kırmızı Çizgi Optimizasyonu. *Mühendislik Alanında Teori ve Araştırmalar: Serüven Yayınevi*, 9, 135-154.
4. Asherlou, S., Yelghi, A., Pancar, E. B., & Oruç, Ş. (2022). Highway Vertical Curve Optimization with Meta- Heuristic Algorithm. *Current Debates on Natural and Engineering Sciences 5: Bidge Yayınları*, 443-459.

