Doktora Tezi

Halit ŞENBERBER

Elektrik-Elektronik Mühendisliği

2018

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

ZEKİ OPTİMİZASYON TEKNİKLERİ İLE SİSTEMLERİ KESİRLİ DERECELİ MODELLEME VE DENETLEYİCİ TASARIMI

Hazırlayan
Halit ŞENBERBER

Danışman
Doç. Dr. Aytekin BAĞIŞ

Doktora Tezi

Haziran 2018
KAYSERİ

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

ZEKİ OPTİMİZASYON TEKNİKLERİ İLE SİSTEMLERİ KESİRLİ DERECELİ MODELLEME VE DENETLEYİCİ TASARIMI

(Doktora Tezi)

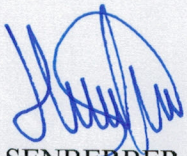
Hazırlayan
Halit ŞENBERBER

Danışman
Doç. Dr. Aytekin BAĞIŞ

Haziran 2018
KAYSERİ

BİLİMSEL ETİĞE UYGUNLUK

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



Halit ŞENBERBER

"**Zeki Optimizasyon Teknikleri ile Sistemleri Kesirli Dereceli Modelleme ve Denetleyici Tasarımı**" adlı Doktora tezi, Erciyes Üniversitesi Lisansüstü Tez Önerisi ve Tez Yazma Yönergesi 'ne uygun olarak hazırlanmıştır.

Tezi Hazırlayan

Halit ŞENBERBER

Danışman

Doç. Dr. Aytekin BAĞIŞ

Elektrik-Elektronik Mühendisliği ABD Başkanı

Prof. Dr. Necmi TAŞPINAR

Doç. Dr. Aytekin BAĞIŞ danışmanlığında **Halit ŞENBERBER** tarafından hazırlanan "**Zeki Optimizasyon Teknikleri ile Sistemleri Kesirli Dereceli Modelleme ve Denetleyici Tasarımı**" adlı bu çalışma, jürimiz tarafından Erciyes Üniversitesi Fen Bilimleri Enstitüsü **Elektrik-Elektronik Mühendisliği** Anabilim Dalında **Doktora** tezi olarak kabul edilmiştir.

07/06/2018

JÜRİ:

Danışman	: Doç. Dr. Aytekin BAĞIŞ
Üye	: Prof. Dr. Nurhan KARABOĞA
Üye	: Prof. Dr. Derviş KARABOĞA
Üye	: Doç. Dr. Murat BARUT
Üye	: Dr. Öğr. Üyesi Asuman SAVAŞÇIHABEŞ

ONAY:
Bu tezin kabulü Enstitü Yönetim Kurulunun 14/08/2018 tarih ve 2018/36-04 sayılı kararı ile onaylanmıştır.

14 / 08 / 2018

Prof. Dr. Mehmet AKKURT

Enstitü Müdürü

TEŞEKKÜR

Bu tez çalışması kapsamında bilgi ve tecrübesini hiçbir zaman esirgemeyen danışman hocam sayın Doç. Dr. Aytekin BAĞIŞ'a teşekkürlerimi sunarım. Tez izleme komitesinde bulunarak çalışmalarımı yönlendiren hocalarım sayın Prof. Dr. Nurhan KARABOĞA ve sayın Prof. Dr. Derviş KARABOĞA'ya ayrıca teşekkür ederim.

Ayrıca; çalışmalarıma doğrudan ya da dolaylı olarak katkıda bulunmuş tüm arkadaşlarıma, değerli hocalarıma ve çalışmalarım süresince sabır göstererek beni daima destekleyen eşime, kızıma ve aileme en içten teşekkürlerimi sunarım.

Halit ŞENBERBER

Kayseri, Haziran 2018

ZEKİ OPTİMİZASYON TEKNİKLERİ İLE SİSTEMLERİ KESİRLİ DERECELİ MODELLEME VE DENETLEYİCİ TASARIMI

Halit ŞENBERBER
Erciyes Üniversitesi, Fen Bilimleri Enstitüsü
Doktora Tezi, Haziran 2018
Danışman: Doç. Dr. Aytekin BAĞIŞ

ÖZET

Farklı özelliklere sahip sistemler için model ve denetleyici tasarımı, bu yapılara uygun sistem parametrelerinin belirlenmesi kontrol sistemlerinde her zaman karmaşıklığını koruyan ama aynı zamanda ilgi çekici problemler arasında yer almıştır. Özellikle klasik denetim yöntemleri ile çözüm üretilmesi zor olan karmaşık sistemler için esnek hesaplama yöntemleri kullanılarak yapılan model oluşturma, denetleyici tasarımı oldukça etkili sonuçlar sunabilmektedir. Yapay zekâ tabanlı algoritmalar kullanılarak yapılan model oluşturma ve denetleyici tasarımı gibi çalışmalar, son zamanlarda araştırmacılar tarafından sıkça kullanılan ve etkili yöntemler arasında yer almaktadır.

Bu tez çalışması iki temel bölümden oluşmaktadır. İlk bölümde literatürde yer alan farklı özelliklere sahip sistemler için model tasarımları gerçekleştirilmiş, model parametreleri yapay arı kolonisi algoritması (YAK) (Artificial bee colony, ABC) kullanılarak tahmin edilmiştir. Tamsayı dereceli model yapılarının yanı sıra kesir dereceli model yapıları da bu çalışmada kullanılmış, bu iki model yapısı arasında performans kıyaslaması gerçekleştirilmiştir. İkinci temel bölümde ise yine literatürde yer alan farklı sistemler için denetleyici tasarımları gerçekleştirilmiştir.

Model oluşturma çalışmasına benzer şekilde denetleyici tasarımında da kesir dereceli ve tamsayı dereceli kontrolörler üzerinde çalışılmıştır. Bu çalışmalarda YAK algoritmasının performansını kıyaslamak amacıyla literatürde sıkça adı geçen, diferansiyel gelişim algoritması (DGA)(Differential evolution, DE) ile parçacık sürüsü optimizasyonu (PSO) (Particial swarm optimization, PSO) yaklaşımları kullanılmıştır.
Çalışmalarda elde edilen sonuçların başarımını değerlendirebilmek için, performans indisleri ağırlıklı olarak ISE (Integral squared error) ve ITAE (Integral time absolute

error) tercih edilirken, bazı çalışmalarda ITSE (Integral time-square error) ve IAE (Integral absolute error) değerleri de performans indeksi olarak kullanılmıştır.

Elde edilen bulgu ve değerlendirmelere göre YAK algoritması ile yapılan modelleme ve denetleyici tasarımı sonuçlarının, DGA ve PSO kullanılarak elde edilen sonuçlara benzer ya da daha iyi olduğu görülmüştür. Ayrıca kesir dereceli model ve denetleyici yapılarının, tamsayı dereceli model ve denetleyici yapılarına göre daha başarılı sonuçlar ortaya koyduğu da yine bu çalışmaların bir başka sonucudur.

Anahtar Kelimeler: Sistem modelleme; Kesir dereceli denetleyiciler; Tamsayı dereceli denetleyiciler; Yapay arı kolonisi algoritması; Diferansiyel gelişim algoritması; Parçacık sürüsü optimizasyonu.

FRACTIONAL ORDER MODELLING OF SYSTEMS AND CONTROLLER DESIGN BY INTELLIGENT OPTIMIZATION TECHNIQUES

Halit SENBERBER
Erciyes University, Graduate School of Natural and Applied Sciences
Ph.D. Thesis, June 2018
Supervisor: Assoc. Prof. Dr. Aytekin BAGIS

ABSTRACT

The design of the model and controller for systems with different characteristics and the determination of the suitable system parameters for these designs have always been one of the problems of interest in the control systems, which maintains its complexity at the same time. Modeling by using soft computing methods and designing the controller can provide very effective results especially for complex systems which are difficult to be solved by classical control methods. Modeling and controller design using artificial intelligence based algorithms are among the most effective methods recently used by researchers.

This thesis study consists of two basic parts. In the first part, the model designs for the systems with different features in the literature is performed and the model parameters are estimated by using artificial bee colony algorithm (ABC). In addition to integer order model structures, fractional order model structures are also used in this study, and performance comparison between these two model structures are performed. In the second basic part, controller design for different systems in the literature has been realized.

Fractional and integer order controller design are also implemented in the controller design similar to the modeling work. In these studies, in order to compare the performance of the ABC algorithm, Differential evolution algorithm (DE) and Particle swarm optimization (PSO) approach which are frequently used in the literature have also been used.

ISE (Integral squared error) and ITAE (Integral time absolute error) are predominantly chosen as performance indices in order to evaluate the performance of the results obtained in the studies. ITSE (Integral time-square error) and IAE (Integral absolute error) errors are also used as performance indexes in some studies.

Based on the results of the obtained graphs and performance indices, it can be said that the results of modeling and controller design by using ABC algorithm have similar to or better than those obtained by using DE and PSO. There is also another result of these studies that fractional order model and controller structures give better results than integer order model and controller structures.

Key words: System modeling; fractional order controller design; integer order controller design; artificial bee colony algorithm; differential evolution algorithm; Particle swarm optimization.

İÇİNDEKİLER

ZEKİ OPTİMİZASYON TEKNİKLERİ İLE SİSTEMLERİ KESİRLİ DERECELİ MODELLEME VE DENETLEYİCİ TASARIMI

Sayfa No

BİLİMSEL ETİĞE UYGUNLUK i

YÖNERGEYE UYGUNLUK ii

KABUL ONAY iii

TEŞEKKÜR iv

ÖZET v

ABSTRACT vii

İÇİNDEKİLER ix

TABLOLAR LİSTESİ xi

ŞEKİLLER LİSTESİ xv

SİMGELER VE KISALTMALAR xix

GİRİŞ 1

1. BÖLÜM

GENEL BİLGİLER ve LİTERATÜR TARAMASI

1.1. Genel Bilgiler ve Literatür Taraması 4

2. BÖLÜM

ÇALIŞMADA KULLANILAN ALGORİTMA, MODEL VE KONTROLÖR YAPILARI

2.1. Yapay Arı Kolonisi (YAK) Algoritması 12

2.2. Diferansiyel Gelişim Algoritması (DGA) 16

2.3. Parçacık Sürüsü Optimizasyonu (PSO) 17

2.4. Kesir Dereceli Sistemlerin Matematiksel Yapısı 20

2.5. Tamsayı ve Kesir Dereceli Model Yapıları 23

2.6. Tamsayı ve Kesir Dereceli Kontrolör Yapıları 25

3. BÖLÜM

YAPAY ZEKA ALGORİTMALARI İLE SİSTEM MODELLEME

3.1 Modelleme çalışmaları 29

3.1.1 Tamsayı Dereceli Süreçler İçin Yapılan Modelleme Çalışmaları..... 29

3.1.2 Kesir Dereceli Süreçler İçin Yapılan Modelleme Çalışmaları..... 44

4. BÖLÜM

YAPAY ZEKA ALGORİTMALARI İLE KONTROLÖR TASARIMI

4.1 Tamsayı Dereceli Süreçler için Yapılan Kontrolör Tasarımları..... 55

4.1.1 Yüksek Mertebeli Tamsayı Dereceli Sistemlerin Kontrolü için Tamsayı Dereceli PID Kontrolör Tasarımları 55

4.1.2 Yüksek Mertebeli Tamsayı Dereceli Sistemlerin Kontrolü için Tamsayı ve Kesirli Dereceli PID Kontrolör Tasarımları 62

4.1.3 Tamsayı Dereceli Sistemlerin Kontrolü için Tamsayı ve Kesirli Dereceli PID Kontrolör Tasarımına Yönelik Diğer Çalışmalar..... 85

4.2. Kesir Dereceli Süreçler için Kontrolör Tasarım Örnekleri..... 116

4.3. Modellenen Sistemler için Kontrolör Tasarımı 145

5. BÖLÜM

TARTIŞMA, SONUÇ VE ÖNERİLER

5.1. Tartışma, Sonuç ve Öneriler..... 155

KAYNAKÇA..... 160

ÖZGEÇMİŞ..... 171

TABLOLAR LİSTESİ

Tablo 2.1. Sistem cevabı yönteminde Ziegler-Nichols ayarları. 27
Tablo 2.2. Sürekli titreşim yöntemine göre denetleyici ayar değerleri. 27
Tablo 3.1. Modellemede kullanılan algoritmalara ait parametre değerleri. 30
Tablo 3.2. G_1, G_2 ve G_3 süreçleri için algoritmalar ile elde edilen model parametreleri.. 30
Tablo 3.3. G_4 ve G_5 süreçleri için algoritmalar ile elde edilen model parametreleri..... 44
Tablo 4.1. T_1, T_2 ve T_3 süreçleri için algoritmalar ile elde edilen kontrolör parametreleri ve basamak cevap özellikleri. 56
Tablo 4.2. T_4 ve T_5 süreçleri için algoritmalara ait kontrol parametreleri. 63
Tablo 4.3. T_4 süreci için YAK algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 64
Tablo 4.4. T_4 sürecine ait YAK algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri 64
Tablo 4.5. T_4 süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 67
Tablo 4.6. T_4 sürecine ait DG algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri 68
Tablo 4.7. T_4 süreci için PSO algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 71
Tablo 4.8. T_4 sürecine ait PSO algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri 71
Tablo 4.9. T4 sürecine ait optimum denetleyiciler için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması. 74
Tablo 4.10. T_5 süreci için YAK algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 75
Tablo 4.11. T_5 sürecine ait YAK algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri 75
Tablo 4.12. T_5 süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 79
Tablo 4.13. T_5 sürecine ait DG algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri 79

Tablo 4.14. T_5 süreci için PSO algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 81
Tablo 4.15. T_5 sürecine ait PSO algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri..... 82
Tablo 4.16. T_5 sürecine ait optimum denetleyiciler için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması 84
Tablo 4.17. T_6 süreci için YAK algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 85
Tablo 4.18. T_6 sürecine ait YAK algoritmasına dayalı optimum denetleyiciler için kullanılarak elde edilen temel performans karakteristikleri 86
Tablo 4.19. T_6 süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 90
Tablo 4.20. T_6 sürecine ait DG algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri..... 90
Tablo 4.21. T_6 süreci için PSO algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 93
Tablo 4.22. T_6 sürecine ait PSO algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri..... 93
Tablo 4.23. T_6 sürecine ait optimum denetleyiciler için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması. 96
Tablo 4.24. T_7 süreci için YAK algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 97
Tablo 4.25. T_7 sürecine ait YAK algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri 97
Tablo 4.26. T_7 süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 100
Tablo 4.27. T_7 sürecin ait DG algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri..... 100
Tablo 4.28. T_7 süreci için PSO algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 103
Tablo 4.29. T_7 sürecine ait PSO algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri..... 103

Tablo 4.30. T_7 sürecine ait optimum denetleyiciler için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması. 105
Tablo 4.31. T_8 süreci için YAK algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 106
Tablo 4.32. T_8 sürecine ait YAK algoritmasına dayalı optimum denetleyiciler için kullanılarak elde edilen temel performans karakteristikleri 106
Tablo 4.33. T_8 süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 109
Tablo 4.34. T_8 sürecine ait DG algoritmasına dayalı optimum denetleyiciler için kullanılarak elde edilen temel performans karakteristikleri..... 110
Tablo 4.35. T_8 süreci için PSO algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 112
Tablo 4.36. T_8 sürecine ait PSO algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri..... 112
Tablo 4.37. T_8 sürecine ait optimum denetleyiciler için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması. 115
Tablo 4.38. T_9 süreci için YAK algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 117
Tablo 4.39. T_9 sürecine ait YAK algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri 117
Tablo 4.40. T_9 süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 120
Tablo 4.41. T_9 sürecine ait DG algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri..... 120
Tablo 4.42. T_9 süreci için PSO algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 123
Tablo 4.43. T_9 sürecine ait PSO algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri..... 123
Tablo 4.44. T_9 sürecine ait optimum denetleyiciler için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması 126
Tablo 4.45. T_{10} süreci için YAK algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 126

Tablo 4.46. T_{10} sürecine ait YAK algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri 127
Tablo 4.47. T_{10} süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 130
Tablo 4.48. T_{10} sürecine DG algoritmasına dayalı ait optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri..... 130
Tablo 4.49. T_{10} süreci için PSO algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 133
Tablo 4.50. T_{10} sürecine ait PSO algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri 133
Tablo 4.51. T_{10} sürecine ait optimum denetleyiciler için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması 135
Tablo 4.52. T_{11} süreci için YAK algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 136
Tablo 4.53. T_{11} sürecine ait YAK algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri 136
Tablo 4.54. T_{11} süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 139
Tablo 4.55. T_{11} sürecine ait DG algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri..... 139
Tablo 4.56. T_{11} süreci için PSO algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri..... 142
Tablo 4.57. T_{11} sürecine ait PSO algoritmasına dayalı optimum denetleyiciler için PSO algoritması kullanılarak elde edilen temel performans karakteristikleri 142
Tablo 4.58. T_{11} sürecine ait optimum denetleyiciler için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması 144
Tablo 4.59. Kontrolör tasarımı için kullanılan ve farklı algoritmalarla elde edilen model yapıları..... 146
Tablo 4.60. Kesir dereceli modeller için algoritmaları kullanarak elde edilen FOPID denetleyici parametreleri..... 147
Tablo 4.61. Tam sayı dereceli modeller için algoritmaları kullanarak elde edilen FOPID denetleyici parametreleri..... 149

ŞEKİLLER LİSTESİ

Şekil 2.1. YAK algoritmasına ait akış diyagramı. 15
Şekil 2.2. PSO algoritmasına ait akış diyagramı. 18
Şekil 2.3. f(x)=x doğrusunun 0.5 ve 1. dereceden türevleri 23
Şekil 2.4. Sistem modelleme blok yapısı 24
Şekil 2.5. Kapalı çevrim denetleyici temel yapısı 25
Şekil 2.6 Giriş Sinyali (a), Çıkış sinyalindeki değişim(b) 26
Şekil 2.7. Sürekli titreşim yöntemi 27
Şekil 2.8. λ-μ uzayında kesir dereceli kontrolör 28
Şekil 3.1. G_1 için kesir dereceli model (Gm2) yapısına ait birim basamak cevapları 32
Şekil 3.2. G_1 için tamsayı dereceli model (Gm1) yapısına ait birim basamak cevapları.. 32
Şekil 3.3. G_1 süreci için kesir dereceli model yapısına (Gm2) ait en düşük hata değerinin iterasyonlara göre değişim grafiği 34
Şekil 3.4. G_1 süreci için kesir dereceli model yapısına (Gm2) ait ortalama hata değerinin iterasyonlara göre değişim grafiği 34
Şekil 3.5.G_1 süreci için tamsayı dereceli model yapısına (Gm1) ait en düşük hata değerinin iterasyonlara göre değişim grafiği. 35
Şekil 3.6.G_1 süreci için tamsayı dereceli model yapısına (Gm1) ait ortalama hata değerinin iterasyonlara göre değişim grafiği 35
Şekil 3.7. G_2 için kesir dereceli model yapısına (Gm2) ait birim basamak cevapları 36
Şekil 3.8. G_2 için tamsayı dereceli model yapısına (Gm1) ait birim basamak cevapları.. 36
Şekil 3.9. G_2 süreci için kesir dereceli model yapısına (Gm2) ait en düşük hata değerinin iterasyonlara göre değişim grafiği 38
Şekil 3.10. G_2 süreci için kesir dereceli model yapısına (Gm2) ait ortalama hata değerinin iterasyonlara göre değişim grafiği 38
Şekil 3.11. G_2 süreci için tamsayı dereceli model yapısına (Gm1) ait en düşük hata değerinin iterasyonlara göre değişim grafiği 39
Şekil 3.12. G_2 süreci için tamsayı dereceli model yapısına (Gm1) ait ortalama hata değerinin iterasyonlara göre değişim grafiği 39
Şekil 3.13. G_3 için kesir dereceli model yapısına (Gm2) ait birim basamak cevapları 40
Şekil 3.14. G_3 için tamsayı dereceli model yapısına (Gm1) ait birim basamak cevapları.40

Şekil 3.15. G_3 süreci için kesir dereceli model yapısına (Gm2) ait en düşük hata değerinin iterasyonlara göre değişim grafiği 42
Şekil 3.16. G_3 süreci için kesir dereceli model yapısına (Gm2) ait ortalama hata değerinin iterasyonlara göre değişim grafiği 42
Şekil 3.17. G_3 süreci için tamsayı dereceli model yapısına (Gm1) ait en düşük hata değerinin iterasyonlara göre değişim grafiği 43
Şekil 3.18. G_3 süreci için tamsayı dereceli model yapısına (Gm1) ait ortalama hata değerinin iterasyonlara göre değişim grafiği. 43
Şekil 3.19. G_4 için tamsayı dereceli model yapısına (Gm1) ait birim basamak cevapları.46
Şekil 3.20. G_4 için kesir dereceli model yapısına (Gm2) ait birim basamak cevapları 46
Şekil 3.21. G_4 süreci için tamsayı model yapısına (Gm1) ait en düşük hata değerinin iterasyonlara göre değişim grafiği 48
Şekil 3.22. G_4 süreci için tamsayı model yapısına (Gm1) ait ortalama hata değerinin iterasyonlara göre değişim grafiği. 48
Şekil 3.23. G_4 süreci için kesir dereceli model yapısına (Gm2) ait en düşük hata değerinin iterasyonlara göre değişim grafiği 49
Şekil 3.24. G_4 süreci için kesir dereceli model yapısına (Gm2) ait ortalama hata değerinin iterasyonlara göre değişim grafiği 49
Şekil 3.25. G_5 için tamsayı dereceli model yapısına (Gm1) ait birim basamak cevapları 50
Şekil 3.26. G_5 için kesir dereceli model yapısına (Gm2) ait birim basamak cevapları 50
Şekil 3.27.G_5 süreci için tamsayı model yapısına (Gm1) ait anlık hata değerinin iterasyonlara göre değişim grafiği 52
Şekil 3.28. G_5 süreci için tamsayı model yapısına (Gm1) ait ortalama hata değerinin iterasyonlara göre değişim grafiği 52
Şekil 3.29. G_5 süreci için kesir dereceli model yapısına (Gm2) ait en düşük hata değerinin iterasyonlara göre değişim grafiği 53
Şekil 3.30. G_5 süreci için kesir dereceli model yapısına ait (Gm2) ortalama hata değerinin iterasyonlara göre değişim grafiği 53
Şekil 4.1 Farklı yöntemler için T_1 sürecine ait birim basamak cevapları 58
Şekil 4.2 T_1 süreci için algoritmalara ait hata gelişim grafikleri..... 58
Şekil 4.3 Farklı yöntemler için T_2 sürecine ait birim basamak cevapları 59
Şekil 4.4 T_2 süreci için algoritmalara ait hata gelişim grafikleri. 59
Şekil 4.5 Farklı yöntemler için T_3 sürecine ait birim basamak cevapları 61

Şekil 4.6 T_3 süreci için algoritmalara ait hata gelişim grafikleri 61
Şekil 4.7 T_4 süreci için YAK algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 65
Şekil 4.8. T_4 süreci için DG algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 69
Şekil 4.9. T_4 süreci için PSO algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 72
Şekil 4.10. T_5 süreci için YAK algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 77
Şekil 4.11. T_5 süreci için DG algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 80
Şekil 4.12. T_5 süreci için PSO algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 82
Şekil 4.13. T_6 süreci için YAK algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 88
Şekil 4.14. T_6 süreci için DG algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 91
Şekil 4.15. T_6 süreci için PSO algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 94
Şekil 4.16. T_7 süreci için YAK algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 98
Şekil 4.17. T_7 süreci için DG algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 101
Şekil 4.18. T_7 süreci için PSO algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 104
Şekil 4.19. T_8 süreci için YAK algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 107
Şekil 4.20. T_8 süreci için DG algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 110
Şekil 4.21. T_8 süreci için PSO algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 113
Şekil 4.22. T_9 süreci için YAK algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 118

Şekil 4.23. T_9 süreci için DG algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 121
Şekil 4.24. T_9 süreci için PSO algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 124
Şekil 4.25. T_{10} süreci için YAK algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 128
Şekil 4.26. T_{10} süreci için DG algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 131
Şekil 4.27. T_{10} süreci için PSO algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 134
Şekil 4.28. T_{11} süreci için YAK algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 137
Şekil 4.29. T_{11} süreci için DG algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 140
Şekil 4.30. T_{11} süreci için PSO algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri 143
Şekil 4.31. G_1 süreci için önerilen kesirli dereceli model yapılarına ait kontrolör tasarımı sonucu elde edilen basamak cevapları..... 150
Şekil 4.32. G_1 süreci için önerilen tamsayı dereceli model yapılarına ait kontrolör tasarımı sonucu elde edilen basamak cevapları 151
Şekil 4.33. G_2 süreci için önerilen kesirli dereceli model yapılarına ait kontrolör tasarımı sonucu elde edilen basamak cevapları..... 152
Şekil 4.34. G_2 süreci için önerilen tamsayı dereceli model yapılarına ait kontrolör tasarımı sonucu elde edilen basamak cevapları 153
Şekil 4.35. G_3 süreci için önerilen kesirli dereceli model yapılarına ait kontrolör tasarımı sonucu elde edilen basamak cevapları..... 154

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış bazı simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Kısaltmalar	**Açıklama**
YAK	Yapay Arı Kolonisi Algoritması
ABC	Artificial Bee Colony
DGA	Diferansiyel Gelişim Algoritması
DE	Differential Evolution
PSO	Parçacık Sürüsü Optimizasyonu
GA	Genetik Algoritma
ISE	Integral Squared Error
ITAE	Integral Time Absolute Error
IAE	Integral Absolute Error
CRONE	Commande Robuste D'ordre Non Entire
NINTEGER	A Non-Integer Control Toolbox For Matlab
FOMCON	Fractional-Order Modeling And Control
TSA	Tabu Araştırma Algoritması
FIR	Finite Impulse Response
F-LMS	Fractional Least Mean Square
CARMA	Controlled Autoregressive Moving Average
PID	Proportional–Integral–Derivative
FOPID	Fractional Order Proportional–Integral–Derivative
CFE	Continious Fraction Expansion

GİRİŞ

Sistemin sahip olduğu giriş-çıkış verileri yardımı ile matematiksel modelinin elde edilmesi işlemi sistem modelleme olarak adlandırılır[1-7]. Modellemenin en temel amacı oluşturulan matematiksel yapının, sistem üzerinde yapılacak diğer çalışmalarda da güvenle uygulanabilmesidir. Sistemler için model oluşturma ve denetleyici tasarımının gerçekleştirilmesi, bu tasarımlar için en uygun sistem parametrelerinin belirlenmesi üzerinde sıklıkla çalışılan önemli kontrol problemlerinden birisidir. Minimum hata değeri bilinmeyen parametrelerin elde edilmesi sistem modellemenin tek amacı değildir. Parametreler en doğru şekilde belirlenirken, bu sürecin en hızlı bir biçimde elde edilmesi de sistem modellemenin temel amaçlarından biridir.

Sisteme ait matematiksel modelin elde edilebilmesi, oluşturulan model yapısının başka sistemlere de uygulanabilmesi anlamına gelmektedir. Bu yüzden oluşturulan model yapısının iyi sonuç verebilmesi için ele alınan probleme uygun model yapısının belirlenmesi büyük önem taşımaktadır. Sistemlerin osilasyonlu olması, dışarıdan gürültü gibi bozucu bir etkiye maruz kalınması ya da ölü zamana sahip olması gibi durumlarda model yapısının oluşturulması daha zor hal almaktadır. Bu durumlarda yapay zekaya dayalı yöntemler hızlı ve etkili çözümler sunabilmektedir.

Klasik denetim yöntemlerinden olan, hız ve konum kontrolü amacıyla sıklıkla kullanılan PID (Proportional–Integral–Derivative) kontrol, hızlı ve etkili denetim sunan bilinen en iyi endüstriyel kontrol sistemlerinden biridir [8,9]. Başlangıcı 20. yüzyılın başına uzanan PID kontrol günümüzde hala hem endüstriyel hem de akademik alanda sıkça kullanılmaktadır. Bunun nedeni, matematiksel yapısı oluşturulan gerçek sistemlere kolayca ve güvenle uygulanabilmesidir. Tasarımının basit olması ve maliyetinin de düşük olması tercih edilmesinde önemli rol oynayan diğer nedenlerdendir. Ancak 21. yüzyılın başlarına kadar hem model hem de kontrolör tasarımında tamsayı dereceli sistemler kullanılmakta iken özellikle yazılım alanında kat

edilen mesafeler ve teknolojik gelişmelere paralel olarak; kesirli hesaplama, kesirli dereceli sistem modellerinin analizinde ve kesir dereceli kontrolör tasarımlarında kullanılmaya başlanmıştır [10-11]. Kesir dereceli sistemlere yönelişin nedeni gerçek sistemleri tanımlamada kesirli dereceli modellerin, tamsayı dereceli modellere göre daha yakın sonuçlar sağlamasıdır.

Denetleyici ve model parametrelerinin belirlenmesinde zaman ve/veya frekans cevabı özellikleri kullanılan bazı yaklaşımlar kullanıcılar için zaman alıcı, karmaşık veya kullanım kolaylığından uzaklaştırıcı olabilmektedir. Son yıllarda klasik model ve denetleyici yapılarının çözüm üretmekte zorlandığı sistemler için, esnek hesaplama yöntemlerine dayalı yapılar sunmuş oldukları daha etkili ve hızlı çözümler nedeniyle sıklıkla tercih edilmektedir. Yapay arı kolonisi (YAK), diferansiyel gelişim (DG), genetik algoritma (GA), karınca kolonisi algoritması (KKA) ve parçacık sürüsü optimizasyonu (PSO) algoritmaları ile yapılan modelleme ve denetleyici tasarımı çalışmalarına literatürde sıklıkla rastlanmaktadır. Tez çalışmasında bunlar içinde son zamanlarda geliştirilen ve popüler olanlardan nümerik problemler için geliştirilmiş, sürü zekası tabanlı PSO ve YAK ile evrimsel tabanlı DG algoritmaları tercih edilmiştir. Parametrelerin hızlı ve doğru bir şekilde elde edilmesinde bu algoritmaların performansları önemli bir rol oynamaktadır.

Farklı özelliklere sahip sistemler için model ve denetleyici yapıları sunmak, bu yapılara ait parametrelerin doğru ve hızlı bir şekilde elde edilmesi bu tez çalışmasının temel amaçlarından biridir. Bu çalışmalar gerçekleştirilirken farklı esnek hesaplama yöntemlerinin kullanımı, bu yöntemler arasındaki performans farklılıklarının derinlemesine incelenmesi ise bu tez çalışmasının bir diğer temel unsuru olarak değerlendirilmektedir. Ayrıca hem tamsayı hem de kesir dereceli model ve kontrolör yapılarının ele alındığı bu tez çalışması, her iki tip sistemin de performanslarını kıyaslama yönüyle literatüre detaylı bir çalışma olarak katkı sunacaktır.

Bu nedenle tez çalışmasının ilk bölümünde kesir ve tamsayı dereceli sistemler hakkında kısa bilgiler verilmiş, bu sistemler, model yapıları ve kontrolör tasarımı hakkında literatür incelemesi yapılmıştır. Kesir dereceli sistemlerin ortaya çıkışı, kontrol sistemlerinde kullanılışı ile ilgili bilgiler yine bu bölümde yer almıştır.

İkinci bölümde ise model ve kontrolör yapılarına ait parametrelerin elde edilmesi için kullanılan algoritma yapılarından bahsedilmiştir. YAK, DG ve PSO algoritmalarının çalışma şekilleri, akış diyagramları, algoritma parametreleri bu bölümde yer almıştır.

İkinci bölümde ayrıca model ve kontrolör yapıları hakkında bilgiler de verilmiştir. Klasik kontrolör olarak adlandırılan PID denetleyici yapısı bu bölümde yer almaktadır. Ayrıca kesir dereceli denetleyicilerin yapısı, denetleyiciye ait blok diyagram, denetleyici parametreleri gibi ayrıntılı bilgiler bu bölümde sunulmuştur.

Üçüncü bölümde tamsayı ve kesir dereceli model yapıları, algoritmalar ile bu modellere ait parametre tahminleri verilmiştir. Modelleme işlemi sonrasında elde edilen parametre değerleri ve hata değerleri tablolar halinde bu bölümde sunulmuştur. Ayrıca birim basamak cevapları ve hata-iterasyon eğrileri de bu bölümde yer almıştır.

Tez çalışmasının temel konularından olan kontrolör tasarım örnekleri dördüncü bölümde incelenmiştir. Literatürde yer alan örnek sistemler için, tamsayı ve kesir dereceli denetleyici yapıları bu bölümde sunulmuştur. Elde edilen parametre ve hata değerleri ile çalışma sonunda elde edilen sonuçlar bu bölümde sunulmuştur.

Modellenen sistemler için kontrolör tasarımına beşinci bölümde yer verilmiştir. YAK, DG ve PSO algoritmalarının kullanıldığı bu çalışmalarda, algoritmaların performansları ortaya konulmuş, aralarındaki farklar, birbirine üstünlükleri gibi özellikler bu bölümde detaylı bir şekilde incelenmiştir. İki farklı denetleyici yapısının kullanıldığı bu bölümde, tamsayı ve kesir dereceli kontrolör yapılarının sonuçları karşılaştırmalı olarak sunulmuştur. Performansların kıyaslanması amacıyla, literatüre uygun olarak, ISE, ITSE, IAE ve ITAE ölçütleri tercih edilmiştir. Ayrıca sadece hata değerinin yer almadığı, aynı zamanda oturma ve aşma sürelerinin dahil edildiği diğer bir amaç fonksiyonu da bu çalışmada kullanılmış, sonuçlar yukarıda belirtilen performans indisleri ile detaylı bir şekilde kıyaslanmıştır.

Tez çalışmasının son bölümünde tartışma, sonuç ve önerilere yer verilmiştir.

1. BÖLÜM

GENEL BİLGİLER ve LİTERATÜR TARAMASI

1.1. Genel Bilgiler ve Literatür Taraması

Sisteme ait bilinmeyen parametrelerin deneysel yöntemlerle elde edilmesi işlemi sistem modelleme olarak adlandırılır [1-3]. Bilinmeyen parametrelerin farklı yöntemler kullanılarak elde edilmesi süreci önemli kontrol problemlerinden biridir. Sistemlerin matematiksel modellerinin önemi bugün mühendislikteki tüm tasarımların yanı sıra biyoloji, ekoloji ve ekonomi gibi alanlarda da hızla artmaktadır [1]. Ancak son zamanlarda bilgisayar alanındaki gelişmelere bağlı olarak tamsayı dereceli model yapıları yerine, gerçek sistemlere ait karakteristik özellikleri daha iyi yansıtan kesir dereceli model yapıları kullanılmaktadır.

Kesir dereceli sistemler, transfer fonksiyonunun dereceleri tamsayı şartı olmaksızın herhangi bir gerçek sayı olan diferansiyel denklemler ile gösterilen sistemler olarak adlandırılabilir [12,13]. Kesir dereceli hesaplama yönteminin ilk kullanımı 17. yüzyılın sonlarında Leibnitz ile L'Hospital arasındaki yazışmalarda görülmüştür. L'Hospital'in 'türevin derecesi tamsayı yerine ½ olması durumunda ne olur?' sorusuna, Leibnitz'in verdiği 'bu ileride faydalı olacak paradoksa dönüşecektir' cevabı kesir dereceli hesaplama yönteminin başlangıcı olarak kabul edilmektedir [14,15]. Kesir dereceli sisteme ait transfer fonksiyonu Eşitlik 1.1'de verilmiştir.

$$G(s) = \frac{b_m s^{\beta_m} + b_{m-1} s^{\beta_{m-1}} + \cdots + b_1 s^{\beta_1} + b_0 s^{\beta_0}}{a_n s^{\alpha_n} + a_{n-1} s^{\alpha_{n-1}} + \cdots + a_1 s^{\alpha_1} + a_0 s^{\alpha_0}} \tag{1.1}$$

β_m ve α_n üs ifadeleri pay ve payda da yer alan katsayıların üsleri olarak tanımlanmaktadır. Katsayı dereceleri kesirli değil de tamsayı olarak seçildiğinde sistem tamsayı dereceli sistem olarak ifade edilmektedir. Kesir dereceli sistemlerin kaplamış

olduğu uzay aralığından dolayı gerçek sistemleri modellemede daha başarılı olacağı düşüncesi ile 20. yüzyıl sonlarında Caputo ve Mainardi tarafından kesirli-türev tabanlı modellerin temelleri atılmıştır. Daha sonra kesir dereceli türev ve integraller içeren durum eşitlikleri dikkate alan çalışmalar Bagley ve Calico (1991); Makroglou, Miller ve Skaar (1994) tarafından sunulmuştur [16,17].

Kesir dereceli modellerin kullanılması ile birlikte kesir dereceli denetleyici tasarımı da gerçekleştirilmiştir. Endüstriyel alanda sıkça kullanılan basit ama etkili bir denetleyici olarak bilinen PID kontrolörden yola çıkarak, kesir dereceli kontrolör yapısı ilk olarak Podlubny tarafından sunulmuştur [11]. Oluşturulan bu yapının, klasik denetleyici olarak adlandırılan PID denetleyiciden daha esnek ve etkin bir yapıya sahip olduğu iddia edilmektedir. Son yıllarda başta kontrol sistemleri olmak üzere endüstriyel alanda kesir dereceli hesaplama yöntemi daha iyi anlaşılmaya başlanmış ve teknolojik gelişmelere bağlı olarak daha yoğun şekilde kullanılmaya başlanmıştır. Özellikle akademik alanda yapılan çalışmalar için oluşturulan kesir dereceli matlab uygulamaları [18-20], bu alanda yapılan çalışmaları hızlandırmıştır. Oustaloup tarafından geliştirilen CRONE (Fransızca "commande robuste d'ordre non entire" kelimelerinin Fransızca kısaltması) [18], Valério tarafından geliştirilen NINTEGER ("A Non-Integer Control Toolbox For Matlab" kelimelerinin kısaltması) [19], ve Tepljakov tarafından geliştirilen FOMCON ("Fractional-order Modeling and Control") [20] bilinen en yaygın kesir dereceli uygulamalardır.

Literatürde model ve kontrolör tasarımı ile ilgili pek çok çalışma mevcuttur. Farklı özellikteki sistemler için tamsayı dereceli model ve denetleyici yaklaşımı ile ilgili çalışmalar literatürde oldukça yaygındır [6,7].

Mukherjee ve arkadaşları [6] zamanla değişmeyen doğrusal sistemleri, düşük dereceli model yapısı kullanarak modellemişlerdir. Yüksek mertebeli sistemler için düşük dereceli model yapılarının kullanıldığı çalışma Şenberber ve Bağış [7] tarafından sunulmuştur. Bu çalışmada modelleme işlemi YAK algoritması kullanılarak gerçekleştirilmiş, DG algoritması ve genetik algoritma (GA) ile de aynı süreçler modellenmiştir. Esnek hesaplama yöntemlerinden PSO kullanarak yüksek mertebeli sistemler için düşük dereceli model tasarımı ile ilgili çalışmalar Nagar ve Singh [21], Bağış [22] ve Gutman ve arkadaşları [23] tarafından literatüre sunulmuştur. Luitel ve

Venayagamoorthy [24]'ye ait çalışmada PSO yaklaşımı kullanarak lineer olmayan sistemler için modelleme yapılmıştır. Zorlu ve Özer [25] tarafından doğrusal olmayan sistem kimliklendirme için klonal seçme algoritması (clonal selection algorithm, CSA) kullanmışlardır. Genetik algoritma ile yapılan çalışmada Coelho and Pessoa [26] deneysel top-tüp sistemi için sistem modelleme çalışmasını literatüre sunmuştur.

Literatürde sıkça kullanılan Box-Jenkins gaz fırını problemi ile ilgili çalışmalar farklı araştırmacılar tarafından sunulmuştur [27-29]. Araştırmacılar bu çalışmalarda tabu araştırma algoritması (TSA), DG ve GA gibi algoritmalar ile lineer olmayan modeller için parametre tahmini gerçekleştirmişlerdir. Zaman gecikmeli sistemler [30] ile kararsız sistemler [31] için model yapılarının oluşturulması ile ilgili çalışmalar Şenberber ve Bağış tarafından gerçekleştirilmiştir. Bu çalışmalarda YAK algoritması kullanılmış, algoritmanın performans incelemesi için DGA ve GA kullanılmıştır. Şenberber [31] tarafından yapılan yüksek lisans tez çalışmasında farklı özelliklere sahip zaman gecikmeli, kararlı-kararsız, osilasyonlu vb. sistemler ele alınmış, bu sistemler için farklı model yapıları sunulmuştur. YAK algoritmasının performansını incelemek amacıyla gerçekleştirilen bu kapsamlı çalışmada elde edilen sonuçlar GA ve DGA ile karşılaştırılmıştır. DG algoritması ve YAK algoritmalarının kullanıldığı lineer olmayan ikinci dereceden Volterra model yapısı ve doğrusal bir FIR (finite impulse response) modeline dayanan diğer bir sistem modelleme çalışması Mete ve arkadaşları [32] tarafından sunmuştur.

Box-Jenkins problemlerinin sistem tanımlaması için Chaudhary ve Raja [33] bir tür kesirli en küçük kareler (fractional least mean square, F-LMS) algoritması geliştirmiştir. Chaudhary ve Raja [34] başka bir çalışmada, CARMA (controlled autoregressive moving average) sistemler için kesir dereceli sistem modellemesinde LMS algoritması kullanmışlardır. Esnek hesaplama yöntemi kullanılarak kesir dereceli sistem modellemesi ile yapılan bir başka çalışma Ming ve Dazi [35] tarafından literatüre kazandırılmıştır. Sabatier ve arkadaşları [36] kurşun asit pillerin şarj tahmini için kesir dereceli model yaklaşımı kullanmışlardır. Öte yandan, kesir dereceli sistemler için, kesir dereceli model yapısı tasarımının gerçekleştirildiği başka bir çalışma Aleksei Tepljakov ve arkadaşları [37] tarafından sunulmuştur. Sözü edilen bu çalışmada sistemler için hem modelleme hem de kesir dereceli kontrolör tasarımı gerçekleştirilmiştir. Zaman domeninde kesir dereceli sistemlerin modellenmesi ve tanımlanmasına dair diğer bir

çalışma Cois ve arkadaşları [38] tarafından gerçekleştirilmiştir. Malti ve arkadaşları [39] tarafından sunulan bir başka çalışmada ise, Monte Carlo yöntemini kullanarak gürültülü bir çıkış ortamında kesir dereceli modeller ile sürekli zamanlı sistem tanımlaması incelenmiştir.

Zaman gecikmeleri farklı ağ kontrol sistemleri için PID kontrolör tasarımı Tran ve arkadaşları [40] tarafından gerçekleştirilmiştir. Li [41] tarafından sunulan çalışmada belirli kazanç ve faz paylarına sahip kapalı döngü sistemler için PID denetleyici tasarımı yapılmıştır. PID denetleyici tasarımına ilişkin amaç ve yöntemlerin incelendiği detaylı bir çalışma Ang ve arkadaşları [42] tarafından sunulmuştur. Papadopoulos ve arkadaşları [43], basamak, rampa ve parabolik girişleri oldukça iyi biçimde izleyebilen Tip3 kontrol döngüleri için dijital PID kontrolör tasarlamışlardır. Kararsız birinci dereceden ölü zamanlı süreçler için, kazanç ve faz payı özelliklerine dayalı PID kontrolör tasarımı Paraskevopoulos ve arkadaşları [44] tarafından sunulmuştur.

Bağış [45] yapmış olduğu çalışmada modifiye edilmiş genetik algoritma yardımıyla PID denetleyici tasarımı yapmış, bunun için üç farklı örnekten yararlanmıştır. Literatürde yer alan örneklerle kıyaslama yapmak amacıyla ISE hata ölçütünü kullanmıştır. Taeib ve arkadaşlarına [46] ait çalışmada, modifiye edilmiş parçacık sürüsü algoritması yardımıyla lineer olmayan sistemler için PID denetleyici tasarımı gerçekleştirilmiştir. Esnek hesaplama yöntemleri yardımıyla yapılan bir diğer çalışmada Theja ve arkadaşları [47] modifiye edilmiş Philip-Heffron modeli kullanarak PID kontrolör tabanlı güç sistemi sabitleyici tasarımını yapay arı kolonisi algoritmasına dayalı olarak incelemişlerdir. Farahani ve arkadaşları [48] çoklu-alan yük frekans kontrolü için PID kontrolör tasarımında kaotik optimizasyon algoritmasından yararlanmışlardır. Gao ve arkadaşları [49] sunmuş oldukları çalışmada, kontrol sinyali, hata değeri ve basamak cevabına ait yükselme süresinin ağırlıklandırılmış formuna sahip bir amaç fonksiyonu kullanan YAK algoritmasına dayalı bir PID kontrolör tasarımı gerçekleştirmiştir.

Fırçasız doğru akım (DA) motor hız kontrol çalışması için PID kontrolörün parametrelerinin PSO algoritması ve bakteriyel besin arama algoritması (BBAA) ile belirlenmesi çalışması İbrahim ve arkadaşları [50] tarafından sunulmuştur. Hasanien [51] tarafından yapılan çalışmada, Taguchi metodu ile birleştirilmiş GA yardımıyla otomatik voltaj regülatörü sisteminin PID ile kontrolü gerçekleştirilmiştir. Diğer

yandan, Hwang ve arkadaşları [52] tarafından sunulan çalışmada ise PID denetleyici performansını artırmak için bulanık mantık kullanılmıştır. Tien ve Lie'ye ait çalışmada [53] CSA ile YAK algoritmasının birleşimi ile oluşturulan yeni bir algoritmanın kullanımına dayalı PID denetleyici örnekleri ele alınmıştır. Adaptif mutasyon operatörüne sahip PSO yaklaşımına dayalı PID denetim örnekleri Chen ve arkadaşları [54] tarafından sunulmuştur.

Ho ve arkadaşları[55] alt-sönümlü (underdamped) sistemlerin kontrolü için PID parametrelerini bulanık sinir ağ yapısı ve benzetilmiş tavlama (simulated annealing) algoritması kullanarak optimize etmişlerdir. YAK algoritmasına dayalı olarak PID parametrelerinin ayarlanması ile ilgili bir başka çalışma Abachizadeh ve arkadaşları [56] tarafından sunulmuştur. Karaboga ve Akay [57] tarafından yapılan çalışmada ise, PID tasarımı işleminde arılar (bees), armoni arama (harmony search) ve YAK algoritmalarının başarımları karşılaştırılmıştır. Değişken parametreli GA'ya dayalı PID kontrolörün performans karşılaştırması Korkmaz ve arkadaşları [58] tarafından yapılmıştır. Diğer yandan, Neath ve arkadaşları [59] yapmış oldukları incelemede çok amaçlı GA ile çok yönlü indüktif güç aktarma sistemi için optimal PID denetleyici tasarımı üzerinde durmuşlardır.

Yadaiah ve Malladi [60] integral ve türev sabiti arasındaki ilişkiyi kullanarak modifiye edilmiş Ziegler-Nichols (ZN) tekniği ile PID denetleyici tasarımı yapmışlar, performans kıyaslaması için PSO yaklaşımı ile GA'yı kullanmışlardır. Erçin ve Çoban [61] tarafından yapılan çalışmada ise, PID kontrol için YAK algoritması ile arılar algoritması kullanılmış, performans ölçütü olarak hataların karelerinin integrali (ISE) yöntemi tercih edilmiştir. Manyetik alana bağlı olarak çalışan ulaşım sistemleri (MAGLEV) için PSO algoritması ile gerçek zamanlı PID kontrolü Wai ve arkadaşları [62] tarafından gerçekleştirilmiştir. PID parametrelerinin belirlenmesine yönelik bir başka çalışmada Luo ve arkadaşları [63], yapay balık sürüsü (artificial fish swarm) algoritmasının performansını ortaya koymuşlardır.

Hsiao ve arkadaşları [64] PID kontrolörlerin tasarımı için karınca kolonisi optimizasyonunu kullanmışlar, algoritmanın performansını ZN ve bulanık mantıkla yapılan çalışmalarla kıyaslamışlardır. Otomatik voltaj regülatörü sisteminde PID'nin optimum tasarımı için PSO yaklaşımı Gaing [65] tarafından gerçekleştirilmiştir.

Otomatik voltaj regülatörü sistemi ile ilgili bir başka çalışmada ise Mukherjee ve Ghoshal [66], PSO ile bulanık PID kontrolör tasarımı gerçekleştirilmiştir.

PSO algoritması kullanarak ileri-geri tip ve bulanık mantığa dayalı güç sistem dengeleyici yapılarında parametre tahmini için yapılan bir çalışma Zonkoly ve arkadaşları [67] tarafından sunulmuştur. Wang [68] tarafından yapılan çalışmada ise zaman gecikmeli sistemlerde kazanç ve faz paylarının istenilen şekilde elde edilmesi için ileri-geri tip faz kompansatörü tasarımı ele alınmıştır. Tan [69] tarafından yapılan çalışmada, lineer zamanla değişmeyen sistemlerin stabilizasyonu için ileri-geri tip kontrolörlere ait parametre hesaplamalarında Hermite–Biehler teoremine dayalı analitik bir yaklaşım kullanılmıştır. Diğer bir çalışmada ise, yüksek mertebeli osilasyonlu sistemler için PID denetleyici tasarımı Bağış ve Şenberber [70] tarafından literatüre sunulmuştur. Kontrolöre ait parametreleri elde etmek için YAK algoritmasının kullanıldığı bu çalışmada, kıyaslama amacıyla PSO ve GA ile de aynı çalışmalar gerçekleştirilmiştir.

Yüksek dereceli, zaman gecikmeli sistemler için DGA ve GA kullanarak PID parametrelerinin ayarlanması ile ilgili bir başka çalışma Saad ve arkadaşları [71], tarafından gerçekleştirilmiştir. Chang ve Shih [72] tarafından yapılan çalışmada ise, doğrusal olmayan örnek sistemlerde kullanılan PID parametrelerinin belirlenmesinde hız operatörü değişikliğe uğratılan PSO yaklaşımı kullanılmıştır. Zhao ve arkadaşları [73] tarafından sunulan çalışma ise, sınırlı model belirsizliği şartları altında PID denetleyici performansını iyileştirmek için PSO'ya dayalı sağlam bir parametre ayarlama yöntemi önermektedir.

Kesirli dereceli basit bir modelin ve frekans cevabı özelliklerinin kullanımına dayalı yeni bir model indirgeme yöntemi ile kesirli derece zaman gecikmeli PID parametrelerinin bulunması çalışması Jin ve arkadaşları [74] tarafından sunulmuştur. Jin ve arkadaşları bu çalışmada performans indisi olarak ITAE hatasını kullanmışlardır. Maiti ve arkadaşları [75] yapmış oldukları çalışmada kesirli sistemlerin PID ile kontrolünü PSO kullanarak gerçekleştirmişlerdir. Parçalı ortogonal fonksiyonların kullanımına dayalı şekilde tamsayılı ve kesirli dereceli süreçler için kontrolör tasarımı konusu Bouafoura ve Braiek [76] tarafından yapılan çalışmada ele alınmıştır.

Kesirli dereceli sistemlerle ilgili yapılan başka bir çalışmada, ölü zamana sahip kesirli dereceli süreçler için ortaya konulan kesir dereceli hibrid bulanık PID denetleyicilerin karşılaştırılması Das ve arkadaşları [77] tarafından yapılmıştır. Debbarma ve arkadaşları [78] tarafından yapılan çalışmada ise, güç sistemlerinin otomatik üretim kontrolünde iki serbestlik dereceli kesirli PID denetim etkisi incelenmiştir. Diğer yandan, Luo ve arkadaşları [79] yapmış oldukları çalışmada kesirli dereceli sistemler için kesirli dereceli oransal-integral (FOPI) denetleyici tasarımı gerçekleştirmiştir.

Luo ve Chen [80] ile sunulan çalışmada, kesirli dereceli sistemler için kesirli oransal-türevsel (FOPD) denetleyici tasarımı gerçekleştirilmiştir. Kesirli bir kutba sahip zaman gecikmeli sistemler için kesirli oransal-integral denetleyici (FOPI) tanımlanması ve ayarlanmasının ele alındığı bir inceleme, Malek ve arkadaşları [81] tarafından sunulmuştur. Çok girişli-çok çıkışlı sistemler için GA'nın kullanımına dayalı kesir dereceli PID kontrolör tasarımı Moradi [82] tarafından gerçekleştirilmiştir. Yeroğlu ve Tan [83] tarafından yapılan çalışmada ise, kesir dereceli sistemler için klasik kontrolör tasarım teknikleri incelenmiştir.

Zaman gecikmeli kesirli sistemler için kontrolör tasarımı yapılan bir diğer çalışma Özbay ve arkadaşları [84] tarafından gerçekleştirilmiştir. Optimal PID ve kesirli dereceli PID kontrolörler için ayarlama kuralları ile ilgili bir diğer çalışma Padula ve Visioli [85] tarafından sunulmuştur. YAK algoritması kullanarak DC motor sürücü beslemeli helikopter için akıllı PID ve kesirli PID (FOPID) hız kontrol tasarımının karşılaştırılmasına yönelik bir çalışma Rajasekhar ve arkadaşları [86] tarafından ortaya konulmuştur. Aynı çalışmada, PSO ile elde edilen sonuçlar YAK ile ulaşılan değerlerle karşılaştırmalı olarak verilmiştir. Yeroğlu ve Ateş'e [87] ait çalışmada, kesirli dereceli oransal-integral-türev (FOPID) kontrolörlerin otomatik ayarlanmasında stokastik çoklu parametreli diverjans yöntemi ele alınmıştır. Kesirli dereceli sistemler için, YAK tabanlı kesirli denetleyici tasarımı Şenberber ve Bağış [88] tarafından incelenmiştir.

Literatürde yer alan çalışmalar dikkate alındığında, klasik kontrolör olarak adlandırılan PID denetleyici ile oldukça fazla sayıda çalışmanın yapıldığı görülmektedir. Yeni bir yaklaşım olarak adlandırılabilen kesirli dereceli PID denetleyici~~si~~ ile ilgili çalışmaların son yıllarda sık şekilde ortaya çıktığı da literatür taramasında verilen örneklerden anlaşılabilmektedir. Bu yüzden bu tez çalışmasında kontrol problemlerinde önemini

hala koruyan PID denetleyici ile kesirli dereceli PID denetleyici kıyaslanmıştır. Literatürde bu iki denetleyicinin birden kıyaslandığı çalışma sayısının oldukça az olması bu çalışmanın önemini daha fazla ortaya koymaktadır. Ayrıca farklı amaç fonksiyonlarının kullanıldığı bu detaylı çalışmanın, bundan sonra yapılacak çalışmalar için fikir vermesi açısından literatüre faydalı olacağı düşünülmektedir. Diğer yandan üç farklı algoritmanın bu çalışmada kullanılması ve performans karşılaştırmaları da yapılacak benzer çalışmalarda kullanıcılara yol göstereceği düşünülmektedir.

2. BÖLÜM

ÇALIŞMADA KULLANILAN ALGORİTMA, MODEL VE KONTROLÖR YAPILARI

Bu tez çalışmasında model ve denetleyici parametrelerini elde etmek için YAK algoritması kullanılmıştır. Algoritmanın performansını daha iyi gözlemleyebilmek için aynı çalışmalar DGA ve PSO ile de yapılmıştır. Ayrıca çalışmanın bir bölümünde denetleyici tasarımı için Matlab uygulamalarından Simulink programı kullanılmıştır [89]. Algoritmalar ile ilgili bilgiler bu bölümde yer almaktadır.

2.1. Yapay Arı Kolonisi (YAK) Algoritması

Yapay arı kolonisi algoritması bal arılarının yiyecek arama davranışlarını örnek alarak, 2005 yılında Karaboğa tarafından geliştirilen sürü zekasına dayalı bir algoritmadır [90-93]. Özellikle nümerik optimizasyon problemlerinin çözümünde iyi sonuçlar ortaya koyan YAK, oldukça az parametreye sahip basit ve esnek bir yapıya sahiptir.

Algoritma 3 temel birleşenden oluşmaktadır [7,15,30,57,70]. İlk bileşen yiyecek kaynakları ile ilgili bilgiyi kolonideki arılar ile paylaşan görevli arılardır. Hafızlarında yiyecek kaynağı ile ilgili bilgi olan bu arılar, kovan dışında aradıkları yiyecekler ile ilgili bilgiyi dans ederek diğer arılarla paylaşırlar. Görevli arıların iletmiş olduğu bu bilgiyi alan ve hangi yiyecek kaynağına gideceğine karar veren arı türü ise ikinci bileşen olan gözcü arılardır. Gözcü arılar, görevli arıların danslarını gözlemler ve yiyecek kaynağının miktarı ve kalitesini göz önünde bulundurarak belirlenen olasılık değerine göre bu seçim işlemini gerçekleştirirler. Son bileşen ise kaşif arılardır. Bu arılar diğer iki arı türünden farklı olarak rasgele yeni yiyecek kaynakları bulurlar ve bu kaynağı kullanmaya başlarlar. Kaynaktan faydalanırken ise görevli arı olarak çalışmaya devam ederler. Bu durum eldeki tüm kaynakların tüketilmesini aynı zamanda ise koloninin

yeni kaynaklarla ilgili bilgilendirilmesini sağlar. Yiyecek kaynaklarının tüketilmesi "limit" kontrol parametresi ile denetlenmektedir. Yiyecek kaynaklarının tamamı tükenmiş yani limit parametresi aşılmışsa, yiyecek kaynağı terk edilir ve rasgele yeni bir kaynak bulunur. Bu durum algoritmada olası çözümlerin tükendiği, yeni bir çözüm ile optimizasyona devam edilmesi gerektiğini göstermektedir.

YAK algoritmasının temel adımları aşağıdaki gibidir:

Başlangıç aşaması

Repeat

Görevli arı fazı

Gözcü arı fazı

Kaşif arı fazı

Until *(durdurma kriteri sağlanana kadar)*

Görevli arıların yiyecek kaynaklarına ilk kez gönderildiği aşama başlangıç aşaması olarak adlandırılmaktadır. Bu durum algoritmada başlangıç çözümlerinin belirlendiği, optimizasyon başlatıldığı kısımdır. Başlangıç rasgele Eşitlik 2.1'de gösterilen ifadeye göre yapılmaktadır.

$$x_{ij} = x_j^{\min} + rand(0,1)\ (x_j^{\max} - x_j^{\min}) \tag{2.1}$$

x_{ij} *i.* yiyecek kaynağının *j.* boyutunda yer alan değeri göstermektedir. i = 1…..SN, j = 1….D ve SN yiyecek kaynağı sayısı ve D optimize edilecek parametre sayısını göstermektedir. Parametre alt sınırı $x_j^{\min}$ ile ifade edilirken, üst sınır $x_j^{\max}$ ile ifade edilmektedir.

Görevli arılar hafızalarında yiyecek kaynağına ait bilgileri tutmaktadır. Hafızalarında tutmuş oldukları yiyecek kaynaklarından daha iyisini elde edebilmek için, kaynağın komşuluğunda sürekli olarak araştırma yapmaktadırlar. Elde edilen yeni kaynak eski kaynaktan kaliteli ise bu kaynak hafızaya alınır ve araştırma yeni kaynağın komşuluğunda yapılır. Görevli arı fazına ait bu araştırma Eşitlik 2.2 ile ifade edilmektedir.

$$v_{ij} = x_{ij} + \varphi_{ij} \ (x_{ij} - x_{kj}) \tag{2.2}$$

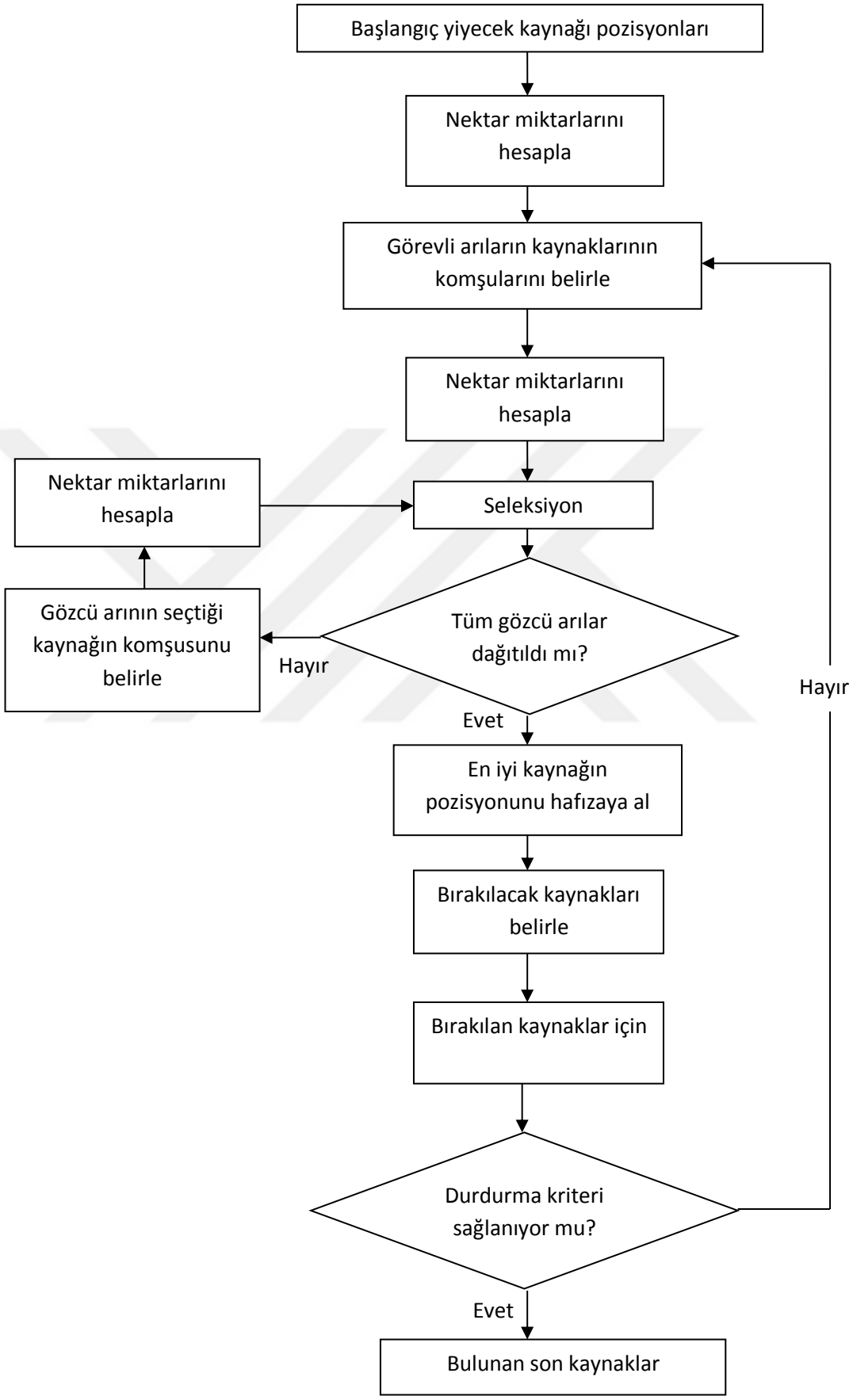
Burada v_{ij} yeni çözümü, x_{ij} mevcut çözümü, φ_{ij} [-1,1] aralığında seçilmiş rasgele bir değeri göstermektedir. x_{kj} ise popülasyondan seçilmiş rasgele yiyecek kaynağını ifade etmektedir. Yeni çözüm v_{ij} ile mevcut durumdaki çözüm x_{ij} arasında kaynak miktarı ve kalitesine göre seleksiyon uygulanmakta ve uygun olan çözüm hafızaya alınmaktadır. Bu işlem ise Eşitlik 2.3'de verilmiştir.

$$fitness_i = \begin{Bmatrix} 1/(1+f_i), & f_i \geq 0 \\ 1+abs(f_i), & f_i \leq 0 \end{Bmatrix} \tag{2.3}$$

Görevli arılar gittikleri nektarların miktarı ve kalitesi ile ilgili bilgileri kovanda bulunan gözcü arılara dans aracılığı ile bildirirler. Dans aracılığı ile aldıkları bilgileri kullanan gözcü arılar kaynakların kalitesine bağlı olarak belirli bir olasılıkla kaynaklar arasında tercih yaparlar. Bu olasılık Eşitlik 2.4 ile hesaplanır.

$$p_i = \frac{f(x_i)}{\sum_{n=1}^{FS} f(x_n)} \tag{2.4}$$

Eşitlik 2.4'de *FS* popülasyon sayısını belirtirken, *pi* yiyecek kaynağının seçim olasılığını belirtmektedir. $f(x_i)$ ise i. kaynağın kalitesini belirtmektedir. Bu aşamadan sonra bütün yiyecek kaynakları tüketilmiş ise, bu durum limit parametresinin tamamlandığı anlamına gelmektedir. Limit parametresinin tamamlanması ile kaşif arı, görevli arıya dönüşür ve Eşitlik 2.1'i kullanarak rasgele yeni çözümler üretmeye çalışır. Şekil 2.1'de YAK algoritmasına ait akış diyagramı verilmiştir [31].



Şekil 2.1. YAK algoritmasına ait akış diyagramı

2.2. Diferansiyel Gelişim Algoritması (DGA)

Diferansiyel gelişim algoritması (DGA) popülasyon tabanlı sezgisel bir algoritmadır [94-95]. Storn ve Price tarafından geliştirilen bu algoritma GA'ya benzer bir yapıya sahiptir. GA'nın yapısında bulunan çaprazlama, mutasyon ve seçim gibi operatörler DGA'da da bulunmaktadır. DGA' nın GA'dan farkı her bir operatör tüm popülasyona sırayla uygulanmamaktadır [96]. Yeni birey elde etme işlemi kromozomların her biri ele alınarak yapılır. Ele alınan kromozom haricinde rasgele seçilen 3 kromozom ile yeni birey elde edilir. Bu durum DGA'nın GA'ya göre daha kısa sürede optimizasyonu tamamlamasını ve daha düşük iterasyon değerlerinde sonuca ulaşmasını sağlar. Algoritmanın temel adımları aşağıdaki gibidir [94];

Başlangıç popülasyonunun oluşturulması
Uygunluk değerinin hesaplanması
Repeat
Mutasyon operatörü
Çaprazlama operatörü
Seçim operatörü
Until *(durdurma kriteri)*

Başlangıç popülasyonu üretilirken Eşitlik 2.5 kullanılır.

$$x_{j,i,G=0} = x_j^{(l_0)} + rand_j[0,1] \cdot \left(x_j^{(h_i)} - x_j^{(l_0)} \right)$$
$$i = (1,2,\ldots \qquad \ldots \qquad randj[0,1] \in [0,1] \tag{2.5}$$

Burada $x_{j,i,G}$ G**.** jenerasyona ait *i* kromozomunun *j*. parametresini, $x_j^{(l_0)}$ parametreye ait alt sınır değeri, $x_j^{(h_i)}$ ise parametreye ait üst sınır değeri ifade etmektedir. *D* optimize edilecek parametre sayısını belirtirken, *NP* üçten büyük sayıda kromozom sayısını ifade etmektedir.

Mevcut kromozom üzerinde rasgele değişiklikler yapmak için mutasyon operatörü kullanılmaktadır. Mutasyon işlemi için elde bulunan vektör dışında 3 vektör daha seçilir ve seçilen bu vektörlerden ikisinin farkı alınarak ölçekleme faktörü (F parametresi) ile çarpılır. Elde edilen yeni vektör üçüncü vektör ile toplanarak yeni vektör elde edilmiş olur. Mutasyon işlemi Eşitlik 2.6'da verilmiştir.

$$n_{j,i,G+1} = x_{j,r_3,G} + F \cdot (x_{j,r_1,G} - x_{j,r_2,G}) \tag{2.6}$$

Burada r_1, r_2, r_3 birbirinden farklı 3 vektörü göstermektedir. $n_{j,i,G+1}$ çaprazlama aşamasında kullanılacak yeni vektörü göstermektedir.

Çaprazlama işlemi mutasyon sonucu elde edilen yeni amaç vektörünü kullanarak yeni jenerasyona aday vektör üretmektir. Deneme vektörüne ait genler çaprazlama oranı (*CR*) olasılıkla mutasyon sonucu elde edilen kromozomdan, (*1-CR*) olasılıkla mevcut kromozomdan seçilir. Bu işlem Eşitlik 2.7'de verilmiştir.

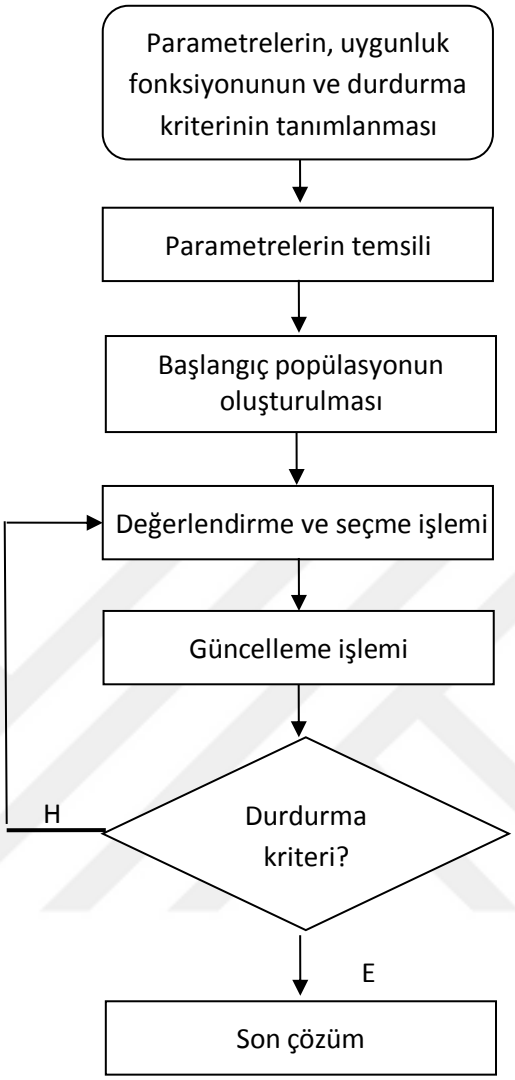
$$u_{j,i,G+1} = \begin{cases} n_{j,i,G} & \text{eger } \left(rand_j[0,1) < CR \vee j = j_{rand}\right) \\ x_{j,i,G} & \text{diger durumlarda} \end{cases} \tag{2.7}$$

Önce mutasyon daha sonra çaprazlama işleminin ardından deneme kromozomu elde edilir. Yeni jenerasyona geçecek birey uygunluk fonksiyonuna göre seçilir. Seçim işleminde ise eldeki bireyler ile üretilen yeni bireyler arasında uygunluğu yüksek olan birey yeni popülasyonun bireyi olarak atanır. Seçim işlemi Eşitlik 2.8'de verilmiştir.

$$\bar{x}_{i,G+1} = \begin{cases} \bar{u}_{i,G+1} & \text{eger } f(\bar{u}_{i,G+1}) \leq f(\bar{x}_{i,G}) \\ \bar{x}_{i,G} & \text{diger durumlarda} \end{cases} \tag{2.8}$$

2.3. Parçacık Sürüsü Optimizasyonu (PSO)

Kuş ve balık sürülerinin iki boyutlu hareketlerinden yola çıkarak Kennedy ve Eberhart tarafından geliştirilen PSO popülasyon tabanlı sezgisel bir algoritmadır [97]. PSO, çok parametreli ve değişkenli optimizasyon problemlerinin çözümünde kullanılan bu algoritma, başlangıçta doğrusal olmayan problemlerin çözümü için geliştirilmiştir [4,15,22,98]. Algoritmada çözümler parçacık, popülasyon ise sürü ile ifade edilir. PSO algoritmasına ait temel akış diyagramı aşağıda verilmiştir [15].



Şekil 2.2. PSO algoritmasına ait akış diyagramı

PSO algoritmasında diğer algoritmalara (GA, DGA vb.) benzer şekilde önce popülasyon rasgele oluşturulur ve çözüm iyileştirilmeye başlar. Ancak PSO algoritmasında çaprazlama ve mutasyon operatörleri yoktur. Her bir iterasyonda, tüm parçacıkların hızları ve pozisyonları güncellenir. Sürüdeki tüm parçacıklar hız vektörü ile en iyi pozisyona ulaşmaya çalışırlar. Elde edilen çözümden parçacığın uygunluk sayısal olarak belirlenir. En iyi uygunluk değerine sahip parçacık bölgesel en iyi olarak etiketlenir. İkinci iterasyonda yeni parçacık değeri belirlenir. İkinci ve daha sonraki iterasyonlarda en iyi uygunluk değerine sahip parçacık yerel en iyi olarak belirlenir ve yerel en iyi ile bölgesel en iyi arasında iyileştirme işlemi gerçekleştirilir. Bu şekilde bireyler arasında bilgi aktarımı olmaktadır.

Eşitlik 2.9'da bir sonraki adımda kullanılacak olan parçacıkları belirlemek için hız vektörü güncellemesi verilmiştir.

$$V_{id} = k\left[Iw_i \times V_{id} + c_1 \times rand_1(\mathrm{P}_{id} - X_{id}) + \mathrm{c}_2 \times rand_2 \times (\mathrm{P}_{gd} - X_{id}) \right] \tag{2.9}$$

Eşitlik 2.10 ise konum vektörüdür.

$$X_{id} = X_{id} + V_{id} \tag{2.10}$$

Burada X_{id} bireyin ilk konumu ve V_{id} anlık hız vektörünü vermektedir. rand_1 ve $rand_2$ değerleri [0,1] aralığında bulunan ve rastgele üretilmiş sayılardır. P_{id} 'i.' parçacığının en iyi olduğu konum ve P_{gd} ise problem için en iyi değere sahip konum vektörünü ifade etmektedir. Iw_i momentum katsayısı (ağırlık değeri) ve c_1 ile c_2 parametreleri hızlanma (cognitive) ve öğrenme (social) gibi kullanıcı tarafından belirlenen katsayıları göstermektedir. c_1 ile c_2 parametreleri uzun süreli denemeler sonucunda, deneysel tecrübelere göre belirlenmiş değerlerdir. k, PSO algoritmasının yakınsama hızını belirleyen parametredir ve Eşitlik 2.11'de ki gibi tanımlanır.

$$k = \frac{2}{\left| 2 - \varphi - \sqrt{\varphi^2 - 4\varphi} \right|} \tag{2.11}$$

Burada $\varphi = c_1 + c_2$ ile ifade edilir. PSO algoritmasının eylemsizlik ağırlığı (Iw_i) Eşitlik 2.12'de verilmiştir.

$$Iw_i = Iw_{\max} \frac{Iw_{\max} - Iw_{\min}}{iter_{\max}} .iter \tag{2.12}$$

Burada $iter_{max}$ maksimum iterasyon sayısını, *iter* ise mevcut durumdaki iterasyon sayısını vermektedir.

2.4. Kesir Dereceli Sistemlerin Matematiksel Yapısı

Giriş bölümünde belirtildiği gibi, kesir dereceli sistemlere yönelişin nedeni gerçek sistemleri tanımlamada kesir dereceli modellerin, tamsayı dereceli modellere göre daha yakın sonuçlar sağlamasıdır. Bu nedenle son yıllarda kesir dereceli matematiksel hesaplamaların geliştirilmesi ile ilgili çalışmalar artmıştır.

Kesir dereceli matematiği tanımlarken $D = \frac{d}{dx}$ diferansiyel operatörü kullanılmaktadır. *f* fonksiyonunun *n*'nci türevi (*n* poizitif tamsayı olursa) ;

$$D^n f(\mathrm{x}) = \frac{d^n f(\mathrm{x})}{dx^n} \tag{2.13}$$

şeklinde ifade edilebilir. Burada karmaşıklığa yol açan durum n değerinin tamsayı olmama durumudur. Bu durum L'Hopital tarafından 1695 yılında dile getirilmiş ve o tarihten beri Euler, Laplace, Fourier, Abel, Liouville, Riemann ve Laurent gibi birçok ünlü matematikçinin dikkatini çekmiştir. 1884 yılına kadar araştırmalardan sonuç elde edilememiş ancak bu tarihte türev derecesinin sadece tamsayı değil, rasyonel veya irrasyonel, pozitif veya negatif gibi rastgele bir sayı olabileceği belirtilmiştir [99]. Bu ifadeye göre kesir dereceli sistemler tamsayı dereceli sistemlerin genel bir ifadesi olarak düşünülebilir.

İntegral ve türev yapılarının ortak gösterimi olan, literatürde 'integro-diferansiyel' operatörü olarak adlandırılan yapı Eşitlik 2.14'de verilmiştir. Eşitlikte görüldüğü gibi üs derecesi pozitif olduğunda türev, negatif olduğunda ise integral anlamına gelmektedir.

$${}_a D_x^a = \begin{cases} \dfrac{d^a}{dx^a} & : a > 0 \\ 1 & : a = 0 \\ \displaystyle\int_a^x (\mathrm{d}\tau)^{-a} & : a < 0 \end{cases} \tag{2.14}$$

Literatürde birçok kesir dereceli integral-türevsel tanımlama mevcuttur. En çok kullanılan kesir dereceli integral-türevsel ifadeler Eşitlik 2.15, 2.16 ve 2.17'de verilmiştir [99-101].

Grünwald–Letnikov yöntemi;

$$ {}_{a}D_{t}^{r} f(\mathrm{t}) = \lim_{h \to o} h^{-r} \sum_{j=0}^{[t-a/h]} (-1)^{j} \binom{r}{j} f(\mathrm{t}-\mathrm{jh}) \tag{2.15} $$

$(-1)^{j}\binom{r}{j}$ ifadesi binomial açılımın katsayılarıdır.

Caputo yaklaşımı;

$$ {}_{a}D_{t}^{r} f(\mathrm{t}) = \frac{1}{\Gamma(r-n)} \int_{a}^{t} \frac{f^{n}(\tau)}{(t-\tau)^{r-n+1}} d\tau \tag{2.16} $$

Burada n-1<r<n and $\Gamma(.)$ bir Gamma fonksiyonudur.

Riemann – Liouville methodu;

$$ {}_{a}D_{t}^{r} f(\mathrm{t}) = \frac{1}{\Gamma(n-r)} \frac{d^{n}}{dt^{n}} \int_{a}^{t} \frac{f(\tau)}{(t-\tau)^{r-n+1}} d\tau \tag{2.17} $$

Burada n-1<r<n and $\Gamma(.)$ bir Gamma fonksiyonudur. Bu fonksiyon Eşitlik 2.18'de verilmiştir.

$$ \Gamma(\mathrm{z}) = \int_{0}^{\infty} e^{-u} u^{z-1} du \tag{2.18} $$

Bu denklemde z'nin tamsayı olması durumunda $\Gamma(\mathrm{z}+1) = z!$ olur.

Kesir dereceli sistemlerin ifade edilebilmesi için bazı yaklaşımların kullanılması gerekmektedir. Nümerik çözüm için diğer bir yaklaşım da hem kutupları hem de sıfırları olabilen fonksiyon için Sürekli Kesirli Açılım (Continious Fraction Expansion - CFE) yöntemidir [80]. Burada amaç kesirli integral-diferansiyel operatörün eşdeğerini elde etmeye çalışmaktır. Bu yapı Eşitlik 2.19'da verilmiştir.

$$G(\mathrm{s}) \approx a_0(\mathrm{s}) + \cfrac{b_1(\mathrm{s})}{a_1(\mathrm{s}) \cfrac{b_2(\mathrm{s})}{a_2(\mathrm{s}) \cfrac{b_3(\mathrm{s})}{a_3(\mathrm{s}) + \ldots}}} \tag{2.19}$$

Burada a_i ve b_i ifadeleri rasyonel ya da sabit ifade olabilir. Bütün bu bilgiler göz önünde bulundurularak konunun daha iyi anlaşılabilmesi için bir fonksiyonun yarı türevinin nasıl elde edildiği aşağıda verilmiştir. Örnek fonksiyon olarak $f(x)=x^k$ alınırsa;

$$f'(\mathrm{x}) = \frac{d}{dx} f(\mathrm{x}) = \mathrm{k}x^{k-1} \tag{2.20}$$

$$\frac{d^a}{dx^a} \mathrm{x}^k = \frac{\mathrm{k}!}{(\mathrm{k}-a)!} x^{k-a} \tag{2.21}$$

genelleştirilmiş ifadesi elde edilir. Faktöriyel yerine gama fonksiyonu kullanılırsa;

$$\frac{d^a}{dx^a} \mathrm{x}^k = \frac{\Gamma(\mathrm{k}+1)}{\Gamma(\mathrm{k}-a+1)} x^{k-a} \tag{2.22}$$

elde edilir. x'in yarı türevi;

$$\frac{d^{\frac{1}{2}}}{dx^{\frac{1}{2}}} \mathrm{x} = \frac{\Gamma(1+1)}{\Gamma(1-\frac{1}{2}+1)} x^{1-\frac{1}{2}} = \frac{\Gamma(2)}{\Gamma(\frac{3}{2})} x^{\frac{1}{2}} = 2\pi^{-\frac{1}{2}} x^{\frac{1}{2}} = \frac{2x^{\frac{1}{2}}}{\sqrt{x}} \tag{2.23}$$

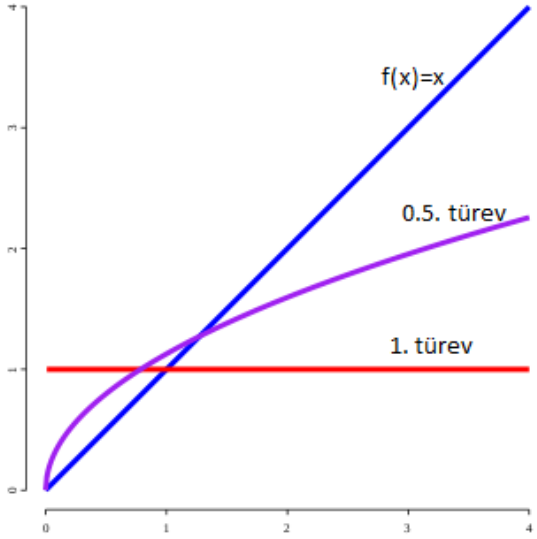
olarak elde edilir. Bir kez daha yarım türev alınırsa;

$$2\pi^{-\frac{1}{2}} x^{\frac{1}{2}} = 2\pi^{-\frac{1}{2}} \frac{\Gamma(1+\frac{1}{2})}{\Gamma(\frac{1}{2}-\frac{1}{2}+1)} x^{\frac{1}{2}-\frac{1}{2}} = 2\pi^{-\frac{1}{2}} \frac{\Gamma(\frac{3}{2})}{\Gamma(1)} x^0 = \frac{1}{\Gamma(1)} = 1 \tag{2.24}$$

elde edilir. Yani;

$$\left(\frac{d^{\frac{1}{2}}}{dx^{\frac{1}{2}}} \frac{d^{\frac{1}{2}}}{dx^{\frac{1}{2}}} \right) x = \frac{d}{dx} x = 1 \tag{2.25}$$

olarak elde edilir. Bir fonksiyonun iki kez yarım türevinin alınması 1 tam türev anlamına gelmektedir. *f(x)=x* için yarım türev ve tam türev grafiği Şekil 2.3'te verilmiştir.



Şekil 2.3. *f(x)=x* doğrusunun 0.5 ve 1. dereceden türevleri

2.5. Tamsayı ve Kesir Dereceli Model Yapıları

Sistemin sahip olduğu giriş-çıkış verilerini kullanarak minimum hata değeri ile matematiksel model oluşturma işlemi modelleme olarak adlandırılır. Sistem modelleme matematiksel ve deneysel olmak üzere iki farklı şekilde gerçekleştirilebilir. Matematiksel modellemede fiziksel kurallar ön planda iken, deneysel modellemede birkaç kez tekrarlanan deneysel çalışma sonucunda elde edilen veriler yardımıyla model oluşturma süreci gerçekleştirilir.

Sistem modelleme yapılırken karşılaşılan bazı durumlarda modelleme işlemi zorlaşmaktadır. Bu zorluklar aşağıdaki gibi sıralanabilir:

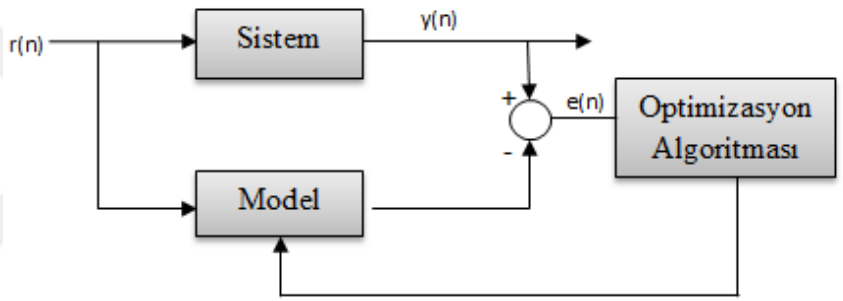
- Modelleme için uygun model yapısının bulunması gerekmektedir. Ele alınan sisteme uygun yapıya sahip olmayan model yapılarında başarılı sonuçlar elde etmek güçtür.
- Dışarıdan maruz kalınan gürültü gibi bozucu etkiler sistem modellemeyi güçleştirmektedir.

- Modelin karakteristiğini yansıtan bazı değişkenleri ölçmek zor hatta bazen imkansızdır.

Sistem modellemeye ait basamakları aşağıdaki gibi ifade etmek mümkündür.

- ❖ Sistemin girişine referans işaret verilir.
- ❖ Sisteme ait uygun model yapısı oluşturulur.
- ❖ Elde edilen modelin parametreleri ile belirlenir.
- ❖ Model girişi ile çıkış işareti arasındaki fark bulunur ve aradaki fark minimize edilmeye çalışılır. Eğer fark azaltılamıyorsa yeni bir model yapısı önerilir.

Modelleme işlemine ait temel blok yapısı Şekil 2.4'te verilmiştir.



Şekil 2.4. Sistem modelleme blok yapısı

Burada *r(n)* giriş (referans değer) sinyali, *y(n)* çıkış sinyali, *e(n)* ise hata değerini göstermektedir. Sisteme girilen referans değer ile model arasındaki hata değeri optimizasyon algoritması yardımı ile minimize edilmeye çalışılır. Algoritma istenilen model yapısına ait parametreleri giriş ile çıkış sinyali arasındaki hata değerine göre belirler.

Sistemlerin osilasyonlu, zaman gecikmeli, doğrusal-doğrusal olmayan gibi durumlarına göre farklı model yapıları tercih edilmektedir. Bu noktada önemli olan daha düşük dereceli model yapıları kullanarak minimum hata değeri ile sistemi modellemektir. Tamsayı ve kesir dereceli örnek model yapıları Eşitlik 2.26 ve 2.27'de verilmiştir.

$$Gmi = \frac{b_0}{a_2 s^2 + a_1 s + a_0} \exp(\mathrm{Ls}) \tag{2.26}$$

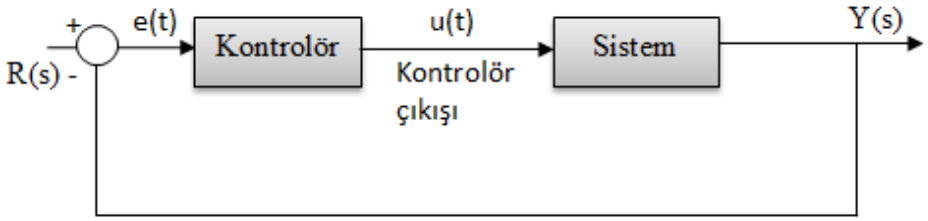
$$Gmf = \frac{b_0}{a_2 s^{k_2} + a_1 s^{k_1} + a_0} \exp(\mathrm{Ls}) \tag{2.27}$$

Eşitlik 2.26 tamsayı dereceli ve ikinci dereceden gecikmeli bir model yapısını gösterirken, Eşitlik 2.27 kesir dereceli gecikmeli model yapısını göstermektedir. Her iki model yapısı da birbirine oldukça benzemektedir. Ancak *Gmi* olarak belirtilen tamsayı dereceli model yapısı için 5 parametre ($b_0, a_2, a_1, a_0 veL$) tahmini yapılırken, *Gmf* (kesir dereceli) model yapısı için 7 ($b_0, a_2, k_2, a_1, k_1, a_0 veL$) parametre tahmini yapılmaktadır. Kesir dereceli modele ait iki farklı parametre k_1 ve k_2 modelleme süresinin zaman alıcı olmasına neden olurken, arama uzayında daha esnek hareket kabiliyetinden dolayı daha başarılı sonuçlar sunmaktadır.

Bu tez çalışmasında ele alınan örnekler göz önünde bulundurularak farklı model yapıları önerilmiştir. Bu modeller ile ilgili çalışmalar üçüncü bölümde detaylı olarak verilmiştir. Bu bölümde sadece sistem modelleme kavramı ve model yapıları üzerinde durulmuştur.

2.6. Tamsayı ve Kesir Dereceli Kontrolör Yapıları

Belirli bir amaç doğrultusunda bir sistemi kontrol etmek, yönlendirmek veya ayarlamak için tasarlanmış yapılar kontrol sistemi olarak adlandırılabilir. İstenilen giriş değerine bağlı olarak denetim büyüklüğünü uygun hale getirmek kontrol sisteminden beklenen, arzu edilen bir durumdur. Temel kapalı çevrim kontrolör yapısı Şekil 2.5'te verilmiştir.



Şekil 2.5. Kapalı çevrim denetleyici temel yapısı

Burada *R(s)* girişe uygulanan referans sinyal, *Y(s)* çıkış işareti, *u(t)* kontrolör çıkışı ve *e(t)* hata değeridir.

PID kontrolör hızlı ve etkili kontrol sunan, bilinen en iyi endüstriyel kontrolörlerden biridir [8]. Oransal-İntegral-Türevsel denetleyicilerin bir araya gelmesi ile oluşan PID kontrolör endüstriyel alanda özellikle hız ve konum kontrol uygulamalarında sıklıkla kullanılmaktadır. 3 parametreye sahip PID kontrolör yapısının algoritması Karl Astrom'a göre şu şekildedir [3];

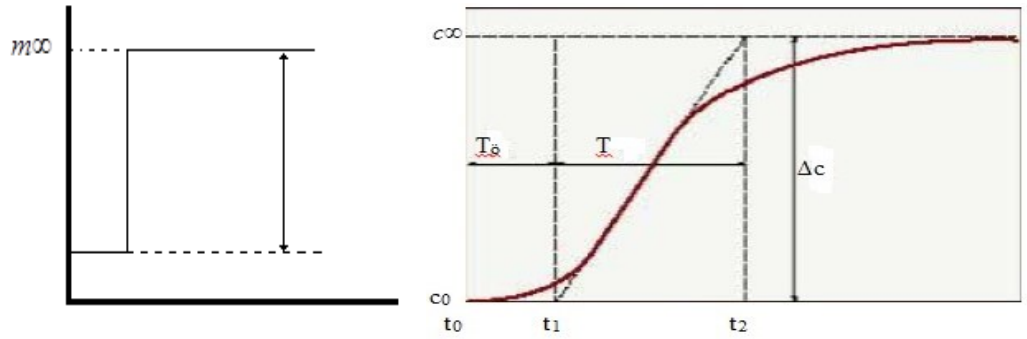
$$u(t) = K[e(t) + 1/T_i \int_0^t e(\tau)d\tau + T_d \frac{de(t)}{dt} \quad (2.28)$$

Klasik PID olarak adlandırılan bu kontrolör yapısına ait transfer fonksiyonu Eşitlik 2.29'da verilmiştir.

$$Gpid(s) = \frac{U(s)}{E(s)} Kp + \frac{Ki}{S} + KdS = Kp + (1 + \frac{1}{TiS} + TdS) \quad (2.29)$$

Literatürde PID denetleyiciye ait parametrelerin belirlenmesi için kullanılan birçok yöntem mevcuttur. Bunlardan en bilinenleri Ziegler-Nichols [102], Cohen ve Coon [103], ISE, IAE, ITAE, ITSE gibi bazı performans kriterlerinin kullanılmasına dayanan yöntemler [104, 105] ile faz-kazanç payı [106] gibi yöntemlerdir.

Literatürde en çok kullanılan Ziegler-Nichols yöntemi ile parametre ayarı iki şekilde yapılmaktadır. Bu yaklaşımlardan ilki olan geçiş cevabı (transient response) yöntemine göre, sistemin açık çevrim cevabı Eşitlik 2.30'daki gibi verilen bir modele göre tanımlanarak Tablo 2.1' de yer alan değerler hesaplanır. Bu durum Şekil 2.6'da gösterilmiştir.



a) b)

Şekil 2.6 (a) Giriş Sinyali, (b) Çıkış sinyalindeki değişim

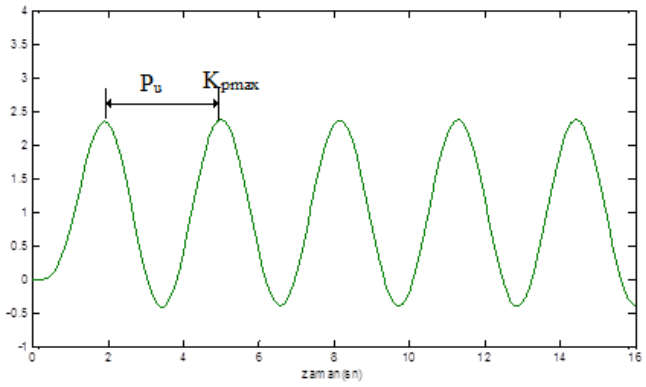
$$K_0 = \frac{\Delta c}{\Delta m}, T_{ö} = t_1 - t_0, T = t_2 - t_1 \quad (2.30)$$

Burada K_0 sistem kazancını, $T_ö$ ölü zaman sabiti, T zaman sabiti, Δm kontrol işaretinin değişim miktarını ve Δc çıkış sinyalindeki değişimi göstermektedir.

Tablo 2.1. Sistem cevabı yönteminde Ziegler-Nichols ayarları [107]

Denetim Türü	K_p	T_i	T_d
P	$\frac{T}{K_0 T_{Ö}}$	-	-
PI	$0.9\frac{T}{K_0 T_{Ö}}$	$3.34T_{ö}$	-
PID	$1.2\frac{T}{K_0 T_{Ö}}$	$2T_{ö}$	$0.5T_{ö}$

PID denetleyicisinin ayarlanmasında bir diğer yöntem ise sürekli salınım veya titreşim yöntemidir. Ziegler-Nichols tarafından geliştirilen bu yöntemde, geri beslemeli ve kapalı döngü bir sistemin sürekli salınımda test edilmesine dayanır. Başlangıçta integral zaman sabiti sonsuza, türev zaman sabiti ise sıfıra ayarlanarak sadece oransal etki aktif hale getirilir. Sisteme birim basamak girişi uygulanarak K_p sürekli artırılır ve Şekil 2.7'de görüldüğü gibi sistem sürekli salınımlı (osilasyonlu) duruma getirilir. Sürekli salınımda K_{pmax} değeri ve sürekli titreşim periyodu P_u elde edilir. K_{pmax} ve P_u değerleri bulunduktan sonra Tablo 2.2 yardımıyla T_i ve T_D değerleri hesaplanır.



Şekil 2.7. Sürekli titreşim yöntemi

Tablo 2.2. Sürekli titreşim yöntemine göre denetleyici ayar değerleri [107]

Etki Türü	K_p	T_i	T_d
P	$0.5\ K_{pmax}$	-	-
PI	$0.45\ K_{pmax}$	$0.825P_u$	-
PID	$0.6\ K_{pmax}$	$0.5P_u$	$0.125P_u$

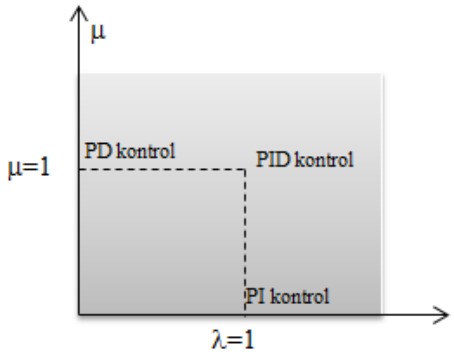
Günümüzde denetleyici tasarımında deneysel ayarların dışında, analitik ayarlar büyük önem taşımaktadır. Analitik ayar, matlab gibi bilgisayar programları yardımıyla

algoritmalar tarafından gerçekleştirilen tasarım yöntemidir. Bu tasarım türü seçilen bir kriterin minimize edilmesi yolu ile gerçekleştirilmektedir. Literatürde yapılan çalışmalarda kullanılan kriter genellikle hatanın minimize edilmesi yöntemidir. Bunun için kullanılan hata ölçütlerinden ISE, IAE, ITAE ve ITSE başlıca olanlarıdır.

Tez çalışmasının giriş bölümünde de belirtildiği gibi kesir dereceli sistemlerin teknolojik gelişmelere bağlı olarak sık bir şekilde kullanılmaya başlanması ile kesir dereceli denetleyici tasarımı da son yıllarda yaygın şekilde kullanılmaya başlanmıştır. Kesir dereceli kontrolörler için klasik kontrolörlerin daha özel bir yapısı olduğu söylenebilir. $PI^{\lambda}D^{\mu}$ şeklinde tanımlanan kesir dereceli PID, klasik PID'den farklı olarak iki farklı parametreye sahiptir. Bu parametreler λ ve μ olarak adlandırılmaktadır. Bu iki parametrenin kontrolöre esneklik ve dayanıklılık sağladığı iddia edilmektedir. İlk kez Podlubny tarafından literatüre sunulan kesir dereceli kontrolöre ait transfer fonksiyonu Eşitlik 2.31'de verilmiştir [11].

$$Gfopid(s) = K_p + \frac{K_i}{s^{\lambda}} + K_d.s^{\mu} = K_p + K_i.s^{-\lambda} + K_d.s^{\mu} \tag{2.31}$$

Burada ($\lambda, \mu > 0$) olmalıdır. $\lambda = \mu = 1$ olduğunda klasik PID, $\lambda = 1$ ve $\mu = 0$ olduğunda PI, $\lambda = 0$ ve $\mu = 1$ olduğunda ise PD kontrolör elde edilir. Bu durum Şekil 2.8.'de açıkça görülebilir.



Şekil 2.8. λ-μ uzayında kesir dereceli kontrolör

Şekil 2.8.'de görüldüğü gibi klasik PID denetleyici türev-integral düzleminde λ ve μ parametrelerinin '0' ya da '1' olma durumuna göre dört nokta ile ifade edilirken, kesir dereceli PID düzlemde sonsuz nokta ile temsil edilir. Bu durum kesir dereceli PID yapısına esneklik kazandırmaktadır.

3. BÖLÜM

YAPAY ZEKA ALGORİTMALARI İLE SİSTEM MODELLEME

Tez çalışmasının bu bölümünde tamsayı dereceli ve kesir dereceli model yapılarının tasarımı gerçekleştirilmiştir. Literatürde yer alan kesir ve tamsayı dereceli örnekler için bazı model yapıları önerilmiş, model parametreleri YAK, DG ve PSO algoritmaları kullanılarak elde edilmeye çalışılmıştır. Bu şekilde hem model yapıları arasındaki başarım kıyaslanırken, bir yandan da algoritmaların modelleme performansları incelenmiştir. Performans kriteri için bu bölümde ISE tercih edilmiştir.

3.1 Modelleme çalışmaları

3.1.1 Tamsayı Dereceli Süreçler İçin Yapılan Modelleme Çalışmaları

Bu bölümde literatürde yer alan tamsayı dereceli süreçler için hem tamsayı hem de kesir dereceli modeller önerilmiştir. $G_1(s)$, $G_2(s)$ ve $G_3(s)$ süreçleri literatürde sırasıyla [108], [109] ve [22] nolu kaynaklardan alınmıştır. Modelleme sürecinde YAK, DG ve PSO algoritmaları kullanılmıştır. Bu sistemler için Eşitlik 3.4 ve Eşitlik 3.5'de verilen model yapıları önerilmiştir. İki model yapısı da ölü zamanlı forma sahiptir. İlk model tamsayı dereceli ve beş parametreye sahip iken, ikinci model yapısı kesir dereceli ve 7 parametreye sahiptir. Algoritmalara ait kontrol parametreleri Tablo 3.1'de verilmiştir.

$$G_1(\mathrm{s}) = \frac{1}{(s+1)(\mathrm{s}^2+0.2s+0.1)} \tag{3.1}$$

$$G_2(\mathrm{s}) = \frac{9}{\mathrm{s}^2+1.2s+9} e^{-0.5s} \tag{3.2}$$

$$G_3(s) = \frac{46.21s+206.1}{0.9372s^3+2.656s^2+75.87s+112.1} \tag{3.3}$$

$$Gm1 = \frac{b_0}{a_2 s^2 + a_1 s + a_0} \exp(\mathrm{Ls}) \tag{3.4}$$

$$Gm2 = \frac{b_0}{a_2 s^{k_2} + a_1 s^{k_1} + a_0} \exp(\mathrm{Ls}) \tag{3.5}$$

Tablo 3.1. Modellemede kullanılan algoritmalara ait parametre değerleri

Kontrol Parametresi	**YAK**	**DGA**	**PSO**
Popülasyon Boyutu	20		
Çaprazlama Oranı		0.9	
Ölçekleme Faktörü		0.8	
Bilişsel ve Sosyal Bileşen			c1,c2=2
Limit	Limit*		
İterasyon Sayısı	100		

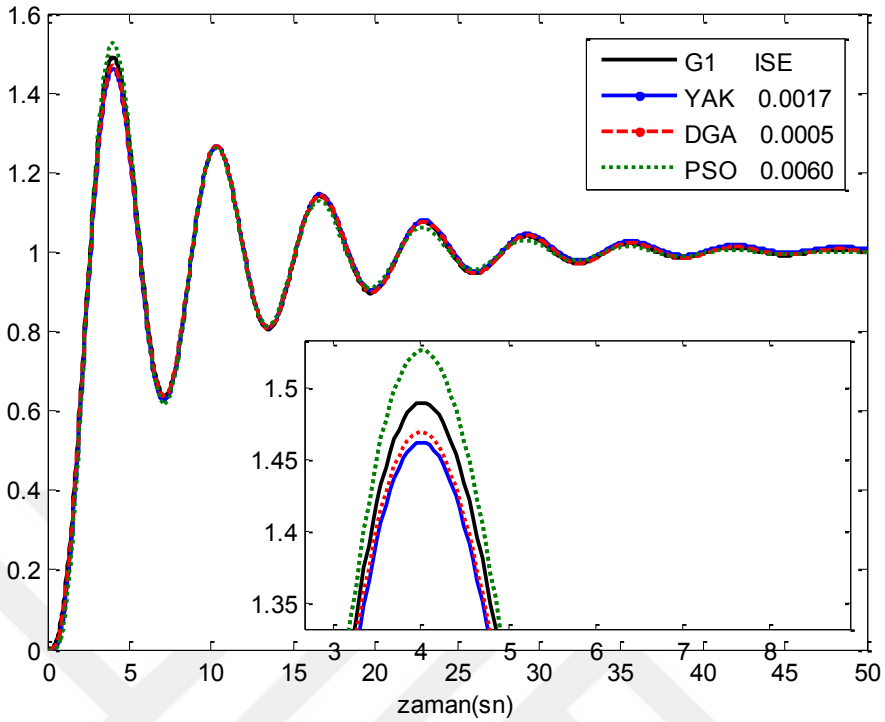
*Limit parametresi kullanılan parametre ve koloni büyüklüğüne göre değişmektedir. Çalışmada kullanılan limit parametresi için, literatürdeki çalışmalara uygun olarak, (koloni büyüklüğü x parametre sayısı)/2 ifadesi kullanılmıştır. Örneğin 5 parametreye sahip model için limit büyüklüğü 50 iken, 7 parametreye sahip model yapısı için 70 olarak seçilmiştir.

Tablo 3.2. G_1, G_2 ve G_3 süreçleri için algoritmalar ile elde edilen model parametreleri

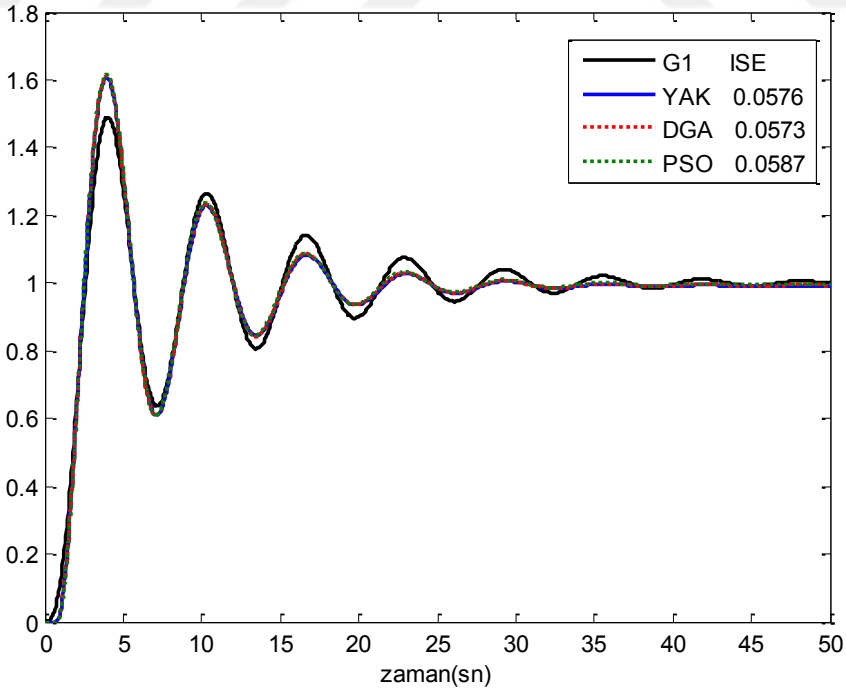
Süreç	**Model**	**Algr.**	**b_0**	**a_2**	**a_1**	**a_0**	**L**	**k_2**	**k_1**	**ISE**	**SS**
$G_1(s)$	Gm1	YAK	60.49	61.17	18.34	60.98	0.757	-	-	0.0576	0.0025
		DGA	67.04	67.48	20.05	67.51	0.757	-	-	**0.0573**	0.0064
		PSO	69.356	70.859	20.651	69.965	0.730	-	-	0.0587	0.0033
	Gm2	YAK	43.519	61.725	50.753	43.527	0.2095	2.4869	1.0492	0.0017	0.0005
		DGA	19.413	26.555	23.195	19.506	0.2242	2.500	1.0977	**0.0005**	0.0057
		PSO	58.618	81.951	58.501	58.143	0.3149	2.3912	0.965	0.0063	0.0075
$G_2(s)$	Gm1	**YAK**	90.601	10.299	12.441	90.380	0.501	-	-	**0.0010**	0.0512
		DGA	87.840	9.386	11.560	88.261	0.510	-	-	0.0022	0.0121
		PSO	32.944	3.6902	4.7131	32.9115	0.4909	-	-	**0.0010**	0.1298
	Gm2	**YAK**	96.778	9.7217	12.291	96.937	0.5081	2.031	1.0867	**0.0001**	0.2487
		DGA	81.345	9.9972	7.7785	81.263	0.5183	1.929	0.9024	0.0006	0.0200
		PSO	13.495	0.4566	2.8068	13.5847	0.4996	2.500	1.4323	0.0009	0.1837
$G_3(s)$	Gm1	**YAK**	98.72	0.0001	20.03	54.32	0.0001	-	-	**0.1608**	0.001
		DGA	75.67	0.0007	15.32	41.71	0.0005	-	-	0.1609	0
		PSO	76.351	0.0481	13.734	41.999	0.0289	-	-	0.1650	0.004
	Gm2	**YAK**	63.605	0.3961	15.417	32.223	0.0073	2.4254	0.6679	**0.0229**	0.0381
		DGA	100	0.6365	18.702	51.737	0.0232	2.2969	0.6454	0.0451	0.0101
		PSO	63.196	11.134	0.3423	34.428	0.0010	0.7636	2.4495	0.0459	0.0017

Tablo 3.2.'de G_1, G_2 ve G_3 süreçleri için önerilen model yapılarına ait parametre değerleri yer almaktadır. Üç algoritma kullanılarak yapılan bu çalışmada hem tam sayı dereceli hem de kesir dereceli model yapıları aynı anda verilmiştir. Bu şekilde hem algoritmaların performansları hem de farklı model yapılarının performans kıyaslaması gerçekleştirilmiştir. Model parametrelerinin yanında ISE hata indisi ile standart sapma (SS) değerleri de yine aynı tabloda yer almaktadır. Hata değeri göz önünde bulundurulduğu zaman 3 süreç için de kesir dereceli model yapısının süreçleri modellemede daha başarılı olduğu söylenebilir. Özellikle G_3 süreci için kesir dereceli model yapısının başarımının, tamsayı dereceli model yapısına göre oldukça üstün olduğu söylenebilir.

Algoritmaların performansları kıyaslandığı takdirde YAK algoritmasının DG ve PSO algoritmalarına göre süreçleri modellemede daha başarılı olduğu ifade edilebilir. Gm1 modeli ile yapılan çalışmalarda algoritmaların performansları birbirine oldukça yakınken, Gm2 modeli ile yapılan çalışmalarda performans farklılıkları ortaya çıkmaktadır. G_1 sürecinde DG algoritması Gm2 modelinde 0.0005 hata değeri ile süreci modellemede diğer algoritmalara göre başarılı performans sergilerken, YAK ve PSO sırası ile 0.0017 ve 0.0063 hata değerlerini elde etmiştir. G_2 ve G_3 süreçlerinde ise YAK algoritmasının yine Gm2 modeli kullanarak sırasıyla 0.0001 ve 0.0229 hata değeri ile sürecin karakteristik özelliklerini en yakın yansıtan çalışmalara sahip olduğu söylenebilir. Gm1 modelinin G_3 sürecini modellemede yetersiz kaldığı Tablo 3.2.'den çıkarılabilecek başka bir sonuçtur.



Şekil 3.1. G_1 için kesir dereceli model (Gm2) yapısına ait birim basamak cevapları



Şekil 3.2. G_1 için tamsayı dereceli model (Gm1) yapısına ait birim basamak cevapları

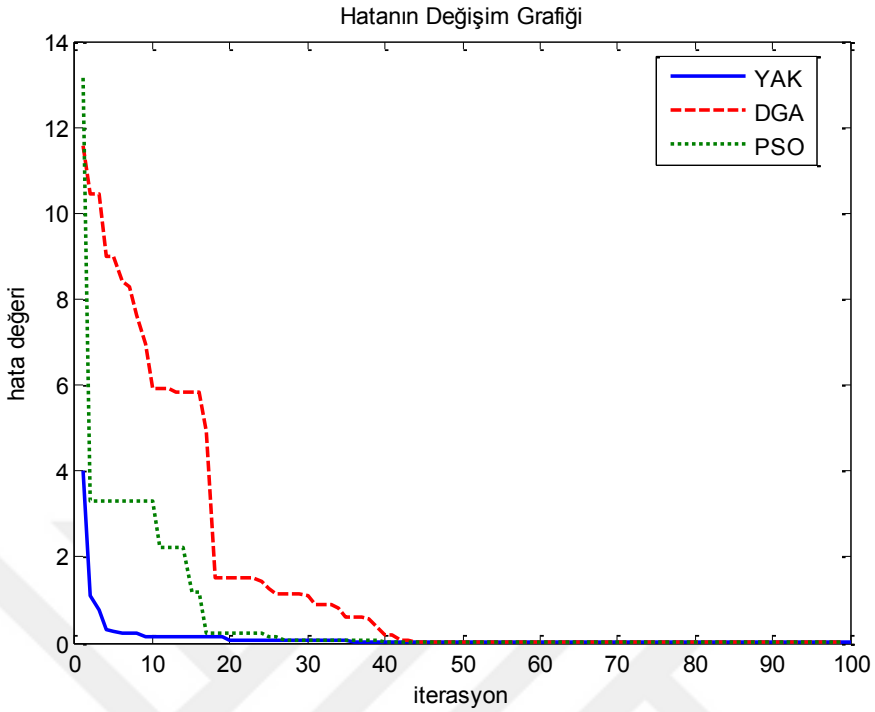
Standart sapma değerleri göz önünde bulundurulduğunda algoritmaların birbirine yakın performans sergilediği ancak PSO algoritmasının daha düşük standart sapma değeri ile

minimum hata değerine ulaştığı yine Tablo 3.2.'den söylenebilir. Gm1 model ile yapılan çalışmalarda neredeyse bütün süreçler için daha düşük standart sapma değeri elde edilmesine rağmen, bu durumun tek başına değerlendirilmesi yerine hata değeri ile birlikte değerlendirilmesi daha anlamlı olacaktır. Özellikle G_3 sürecinde tamsayı dereceli model yapısının başarımı oldukça düşük iken elde edilen standart sapma değeri yanıltıcı olacaktır.

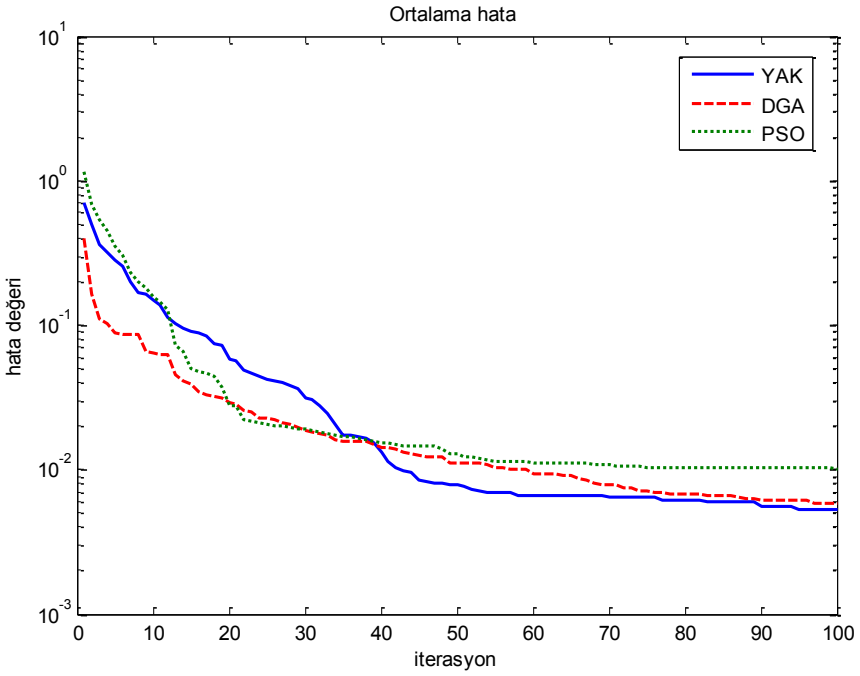
Şekil 3.1. kesir dereceli model yapısı kullanılarak G_1 süreci için elde edilen birim basamak cevaplarını göstermektedir. Aynı süreç için tamsayı dereceli model yapısına ait basamak cevapları ise Şekil 3.2.'de verilmiştir. Algoritmaların süreci modellemede oldukça başarılı ve birbirine oldukça yakın performanslar sergilemesi nedeniyle grafiklerin örtüştüğü görülmektedir. Bu nedenle şekiller içerisinde farklılıkların ortaya çıktığı alanlar ayrıntılı olarak verilmiştir. Bu durum tez çalışmasının diğer aşamalarında farklı örneklerde de verilecektir.

Osilasyonlu olması nedeniyle modellenmesi zor olan G_1 sürecine ait basamak grafiklerinde algoritmaların başarılı performans sergiledikleri söylenebilir. 0-50 saniye aralığında neredeyse grafiğin tamamında üç algoritma ile elde edilen basamak cevapları, G1 sürecine ait basamak cevabı ile örtüştüğü Şekil 3.2.'den görülmektedir. Gm1 model yapısına ait çıkış grafiklerinde ise bu durum biraz daha farklıdır. Bu model yapısının G_1 sürecini modellemede Gm2 model yapısına göre başarımı daha düşüktür. Şekil 3.2.' ye göre neredeyse şeklin tamamında sürece ait grafik ile algoritmalar ile elde edilen grafikler arasında farklılıklar ortaya çıkmaktadır.

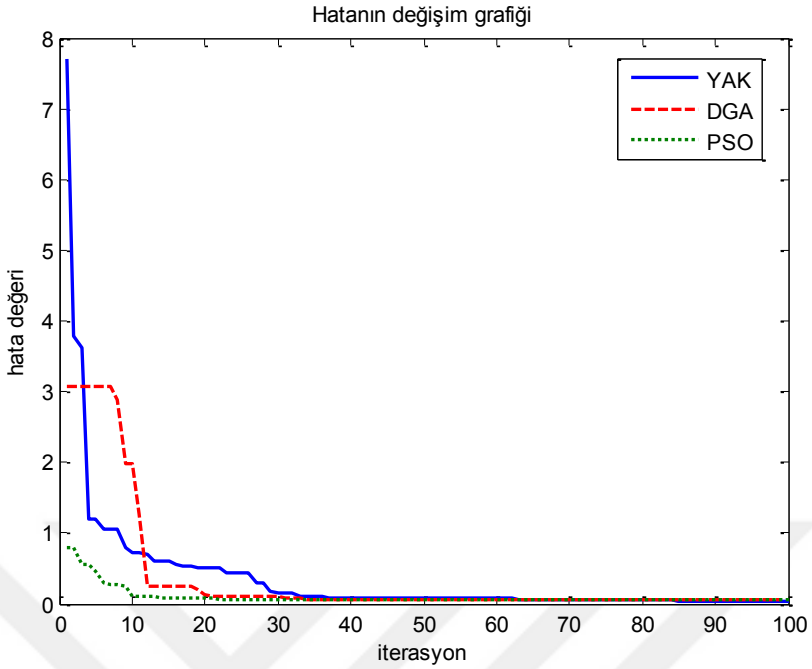
Şekil 3.3. G_1 süreci için kesir dereceli model yapısına ait en düşük hata değerinin iterasyonlara göre değişimini gösterirken, Şekil 3.4. ise yine aynı sürece ait hata değerlerinin ortalamasını göstermektedir. En düşük hata değerinin yer aldığı grafikte bütün algoritmalar neredeyse ilk 40 iterasyonda hatayı minimize etmiştir. YAK algoritması ilk 10 iterasyonda elde edilen minimum hata değerine oldukça yakınsamıştır. Ortalama hata değerleri göz önünde bulundurulduğunda ise YAK algoritmasının 30 koşturma sonunda diğer iki algoritmaya göre daha düşük hata değerini elde ettiği Şekil 3.4.'den görülmektedir.



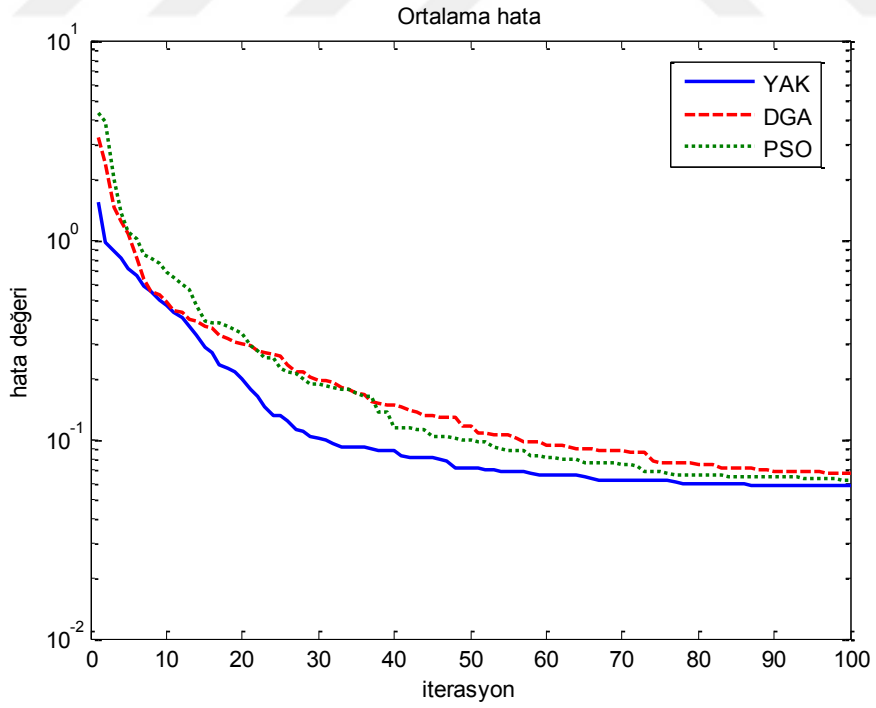
Şekil 3.3. G_1 süreci için kesir dereceli model yapısına (Gm2) ait en düşük hata değerinin iterasyonlara göre değişim grafiği



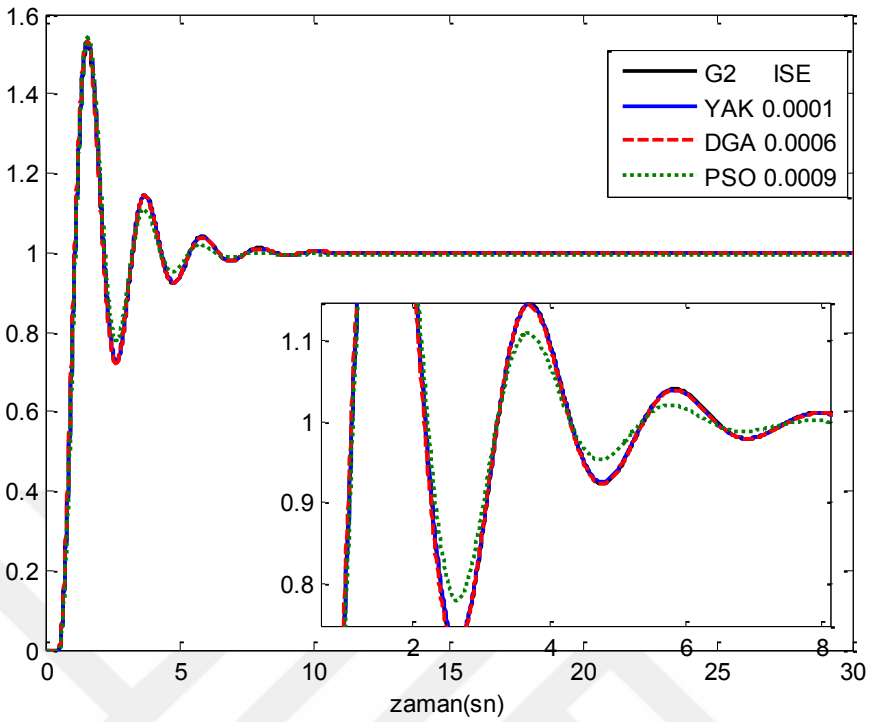
Şekil 3.4. G_1 süreci için kesir dereceli model yapısına (Gm2) ait ortalama hata değerinin iterasyonlara göre değişim grafiği



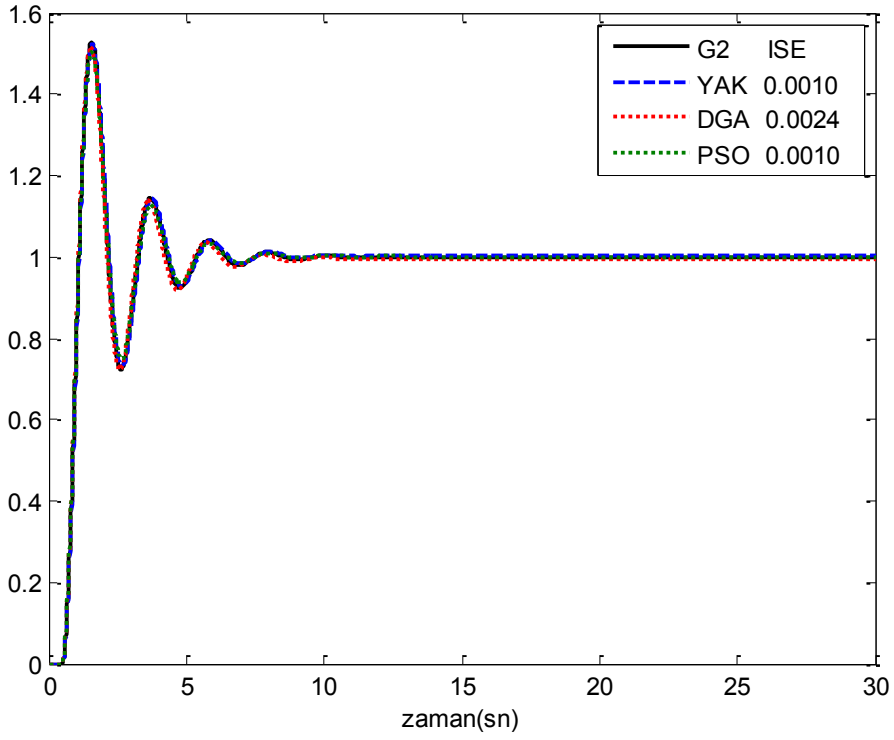
Şekil 3.5. G_1 süreci için tamsayı dereceli model yapısına (Gm1) ait en düşük hata değerinin iterasyonlara göre değişim grafiği



Şekil 3.6. G_1 süreci için tamsayı dereceli model yapısına (Gm1) ait ortalama hata değerinin iterasyonlara göre değişim grafiği



Şekil 3.7. G_2 için kesir dereceli model yapısına (Gm2) ait birim basamak cevapları



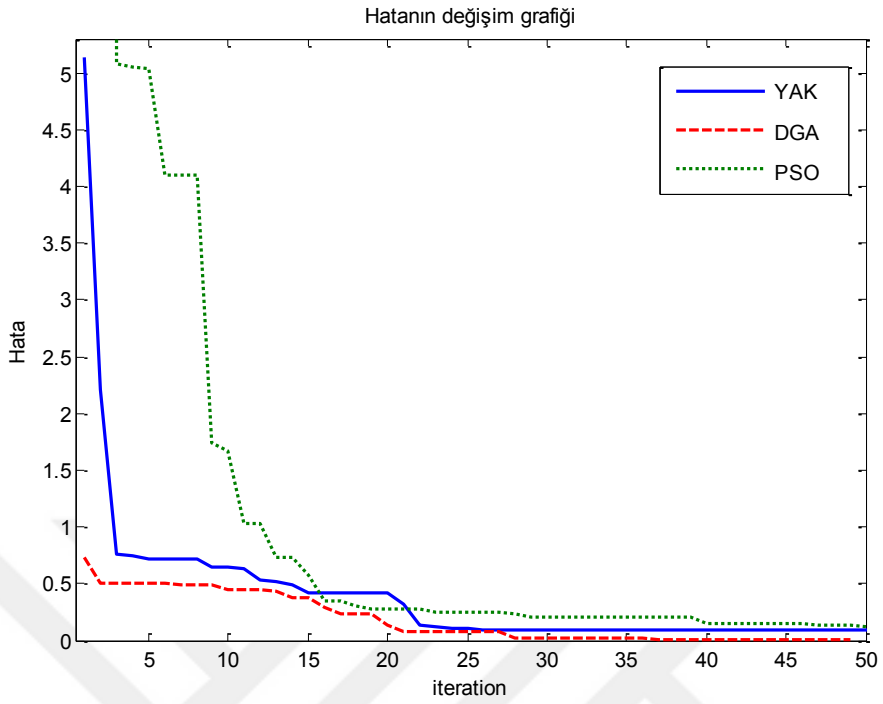
Şekil 3.8. G_2 için tamsayı dereceli model yapısına (Gm1) ait birim basamak cevapları

G_2 süreci için önerilen model yapılarına ait birim basamak cevapları Şekil 3.7 ve 3.8'de yer almaktadır. Basamak cevaplarının yer aldığı ilk şekil kesir dereceli model yapısına ait iken, ikinci şekil tamsayı dereceli model yapısına ait birim basamak cevaplarını göstermektedir. Şekil 3.7' ye göre YAK algoritmasının 0.0001 hata değeri ile en başarılı model parametrelerini elde ettiği görülürken, DGA 0.0006 ve PSO 0.0009 hata değeri ile YAK algoritmasına yakın sonuçlar elde etmiştir. Tamsayı dereceli model yapısına ait Şekil 3.8 incelendiği zaman yine üç algoritmanın da yakın sonuçlar elde ettiği PSO ve YAK algoritmasının çok küçük farkla daha başarılı sonuç elde ettiği söylenebilir. İki model yapısı ile yapılan çalışmalar kıyaslandığında ise kesir dereceli model yapısı ile elde edilen sonuçların tamsayı dereceli model yapısına göre daha başarılı olduğu yine bu iki grafikten açıkça görülmektedir.

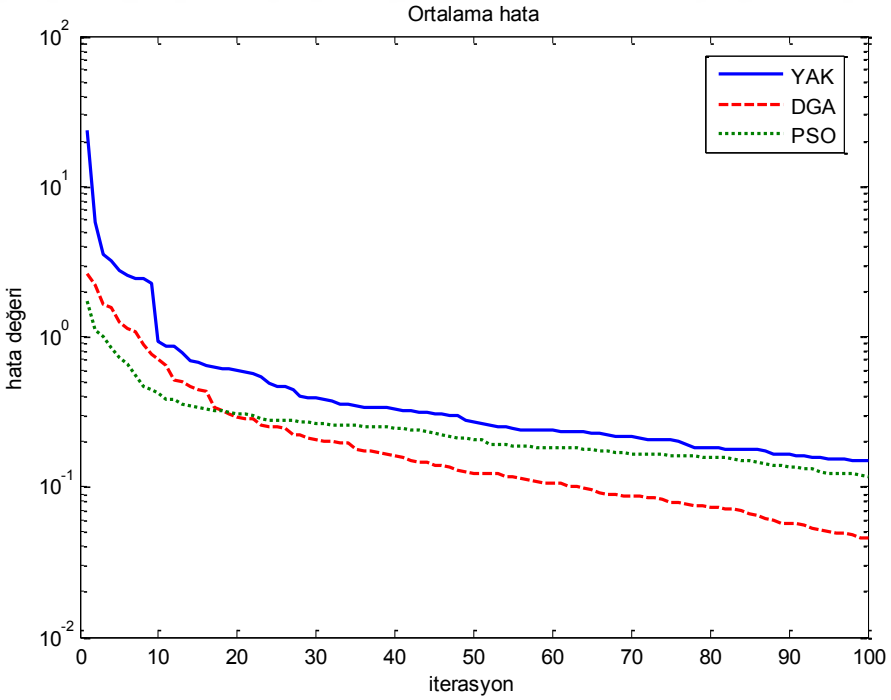
Kesir dereceli model yapısı için en iyi sonuçlara ait hata değişim grafiğini gösteren Şekil 3.9'a göre YAK algoritması hızlı bir şekilde hatayı minimize etmiş, yaklaşık ilk 25 iterasyonda minimum değere ulaşmıştır. DGA'nın da benzer performans sergilediği bu süreçte, PSO algoritması ilk 50 iterasyonda hata değerini minimize etmiştir. Ortalama hata değerlerinin yer aldığı Şekil 3.10'a göre YAK algoritması daha yüksek hata değerinden başlayıp hatayı minimize ederken, PSO ve DGA daha düşük hata değerinden başlamıştır. Ortalama hata değerinde en düşük sonuç DG algoritması kullanılarak elde edilmiştir.

Tamsayı dereceli model yapısına ait hata gelişim grafikleri Şekil 3.11 ve Şekil 3.12'de yer almaktadır. En iyi sonuçlara ait grafik olan Şekil 3.11'e göre bütün algoritmalar neredeyse ilk 15 iterasyonda hatayı minimize etmiştir. Ortalama hata değerlerinin yer aldığı grafiğe göre algoritmalar birbirine yakın hata değerinden optimizasyona başlamış, YAK algoritması bu algoritmalar arasında en düşük ortalama hata değerini elde etmiştir.

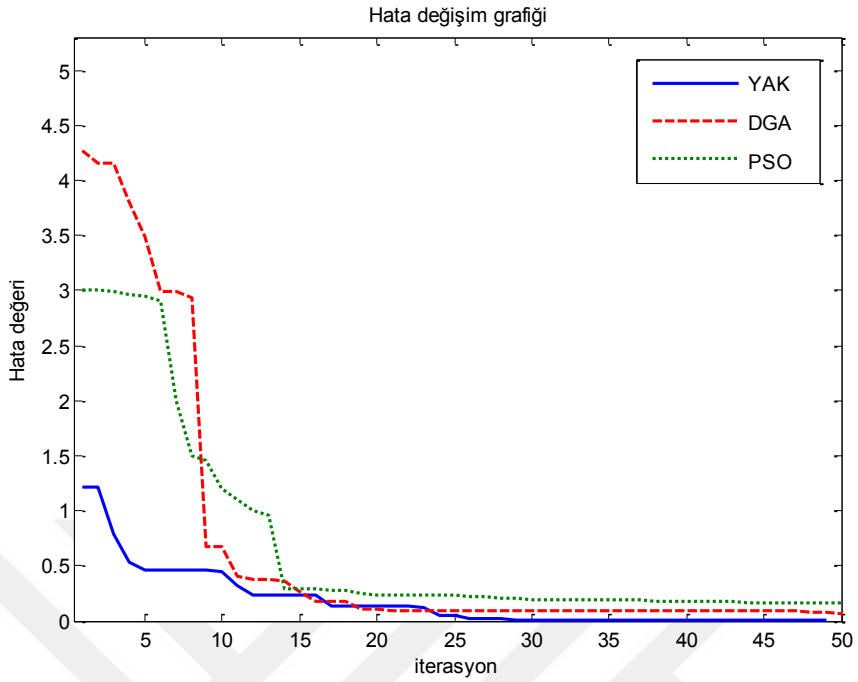
Osilasyonlu ve aynı zamanda gecikmeli olan G_2 sürecini modellemede üç algoritmanın da başarılı sonuçlar ortaya koyduğu söylenebilir. Algoritmaların düşük iterasyon sayılarında hatayı minimize etmeleri yine grafiklerden açıkça görülmektedir. Modellemesi oldukça zor olan bir sonraki örnekte ele alınan süreç algoritmaların ve farklı model yapılarının performans kıyaslamalarında daha bilgi verici olacaktır.



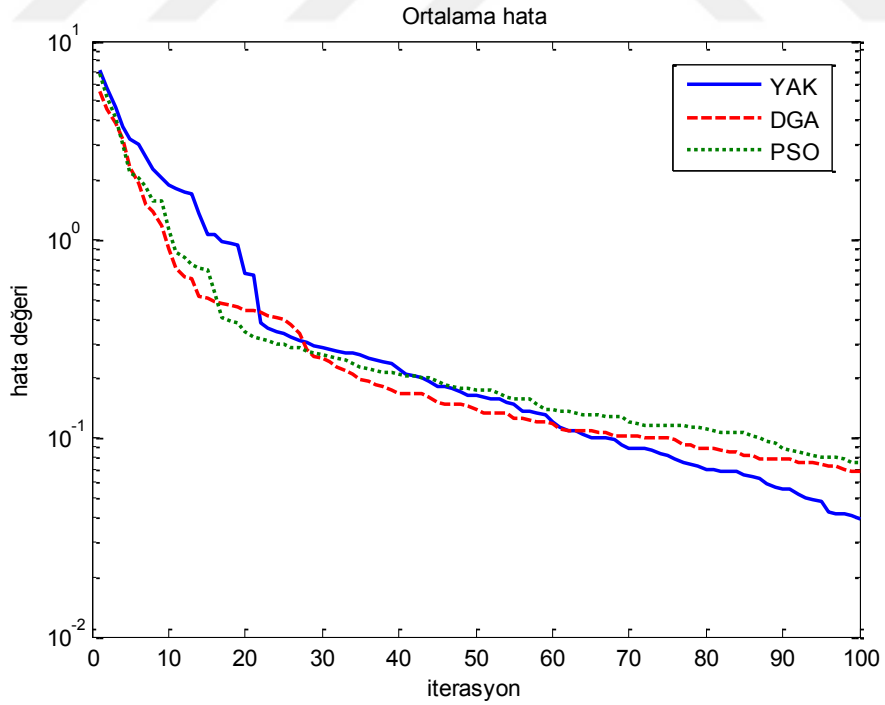
Şekil 3.9. G_2 süreci için kesir dereceli model yapısına (Gm2) ait en düşük hata değerinin iterasyonlara göre değişim grafiği



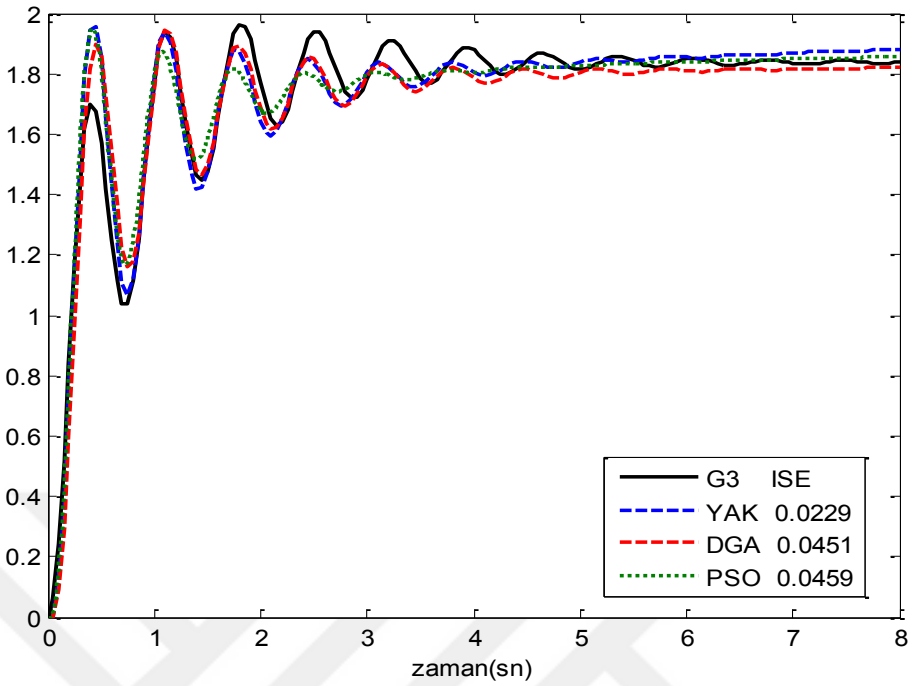
Şekil 3.10. G_2 süreci için kesir dereceli model yapısına (Gm2) ait ortalama hata değerinin iterasyonlara göre değişim grafiği



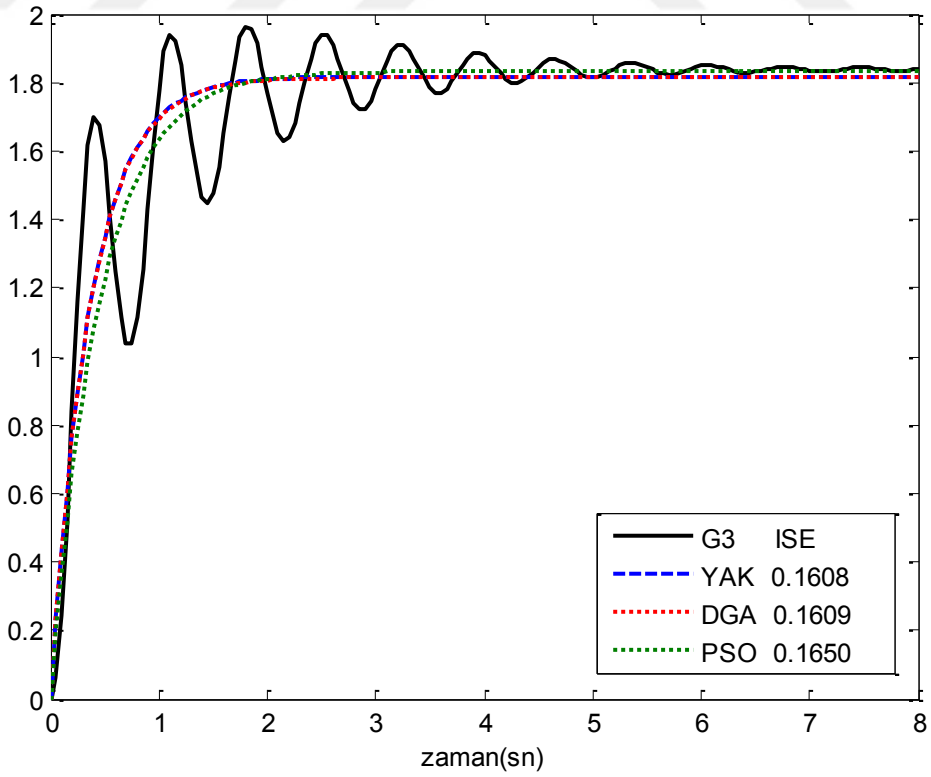
Şekil 3.11. G_2 süreci için tamsayı dereceli model yapısına (Gm1) ait en düşük hata değerinin iterasyonlara göre değişim grafiği



Şekil 3.12. G_2 süreci için tamsayı dereceli model yapısına (Gm1) ait ortalama hata değerinin iterasyonlara göre değişim grafiği



Şekil 3.13. G_3 için kesir dereceli model yapısına (Gm2) ait birim basamak cevapları



Şekil 3.14.G_3 için tamsayı dereceli model yapısına (Gm1) ait birim basamak cevapları

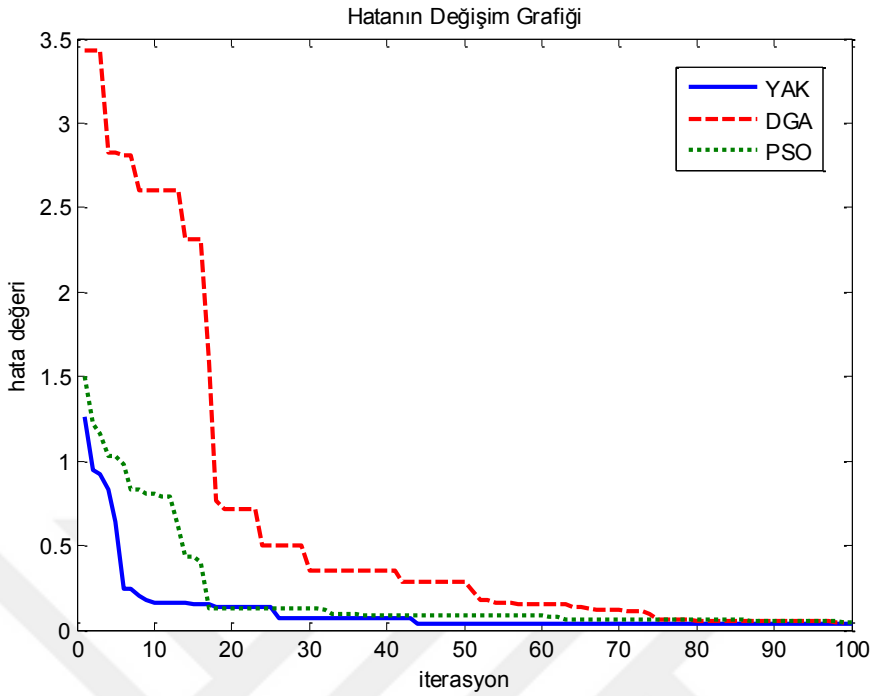
Şekil 3.13'de G_3 süreci için önerilen kesir dereceli model yapısına ait basamak cevapları yer almaktadır. Modellemesi oldukça zor olan bu süreçte üç algoritmanın da model parametrelerini düşük hata değeri ile elde ederken başarılı performans sergiledikleri söylenebilir. YAK algoritması 0.0229 hata değeri ile en iyi performansı elde ederken, DGA 0.0451 ve PSO 0.0459 hata değeri ile YAK algoritmasına göre daha düşük performans sergilediği yine aynı grafikten çıkarılabilecek bir sonuçtur. Ayrıca önerilen model yapısının süreci modellemede başarılı olduğu yine Şekil 3.13'den söylenebilir.

Tamsayı dereceli model yapısına ait basamak cevaplarının yer aldığı Şekil 3.14'de ise üç algoritmanın süreci modellemede başarılı olamadığı açıkça görülmektedir. Bu durum algoritmalardan ziyade önerilen ikinci derece model yapısının başarımı ile ilgilidir. Tam sayılı model yapısı süreci modelleme de yetersiz kalmış, sisteme ait osilasyonlu çıkışları elde edememiştir. Giriş kısmında da belirtilen kesir dereceli model yapısının tam sayı dereceli model yapısına göre daha esnek yapıya sahip olduğu ifadesi bu iki grafikte açıkça görülmektedir.

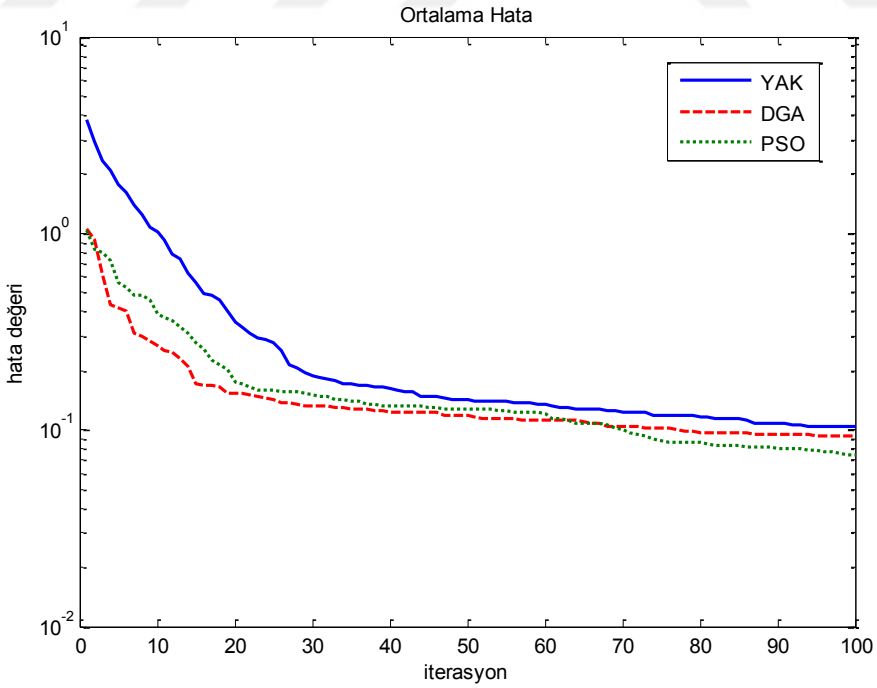
Şekil 3.15 G_3 sürecinde kesir dereceli model yapısı için elde edilen en iyi sonuçlara ait hata gelişim grafiklerini göstermektedir. Hata değerini YAK ilk 10 iterasyonda, PSO ilk 20 iterasyonda ve DGA 75. iterasyona kadar indirgemiştir. Ortalama hata değerinin yer aldığı Şekil 3.16 dikkate alındığında ise algoritmalar oldukça yakın performans sergilemiş, PSO en düşük ortalama hata değerini elde etmiştir.

Tamsayı dereceli model yapısına ait en iyi sonuçlara ait hata gelişim grafiğinde ise YAK algoritması diğer iki algoritmaya göre daha yüksek değerden optimizasyona başlamış ve yine diğer iki algoritmaya göre hatayı çok daha hızlı bir şekilde minimize etmiştir. YAK yaklaşık ilk 20 iterasyonda minimum hata değerine ulaşırken, PSO ve DGA ilk 40 iterasyonda minimum hata değerini elde etmiştir.

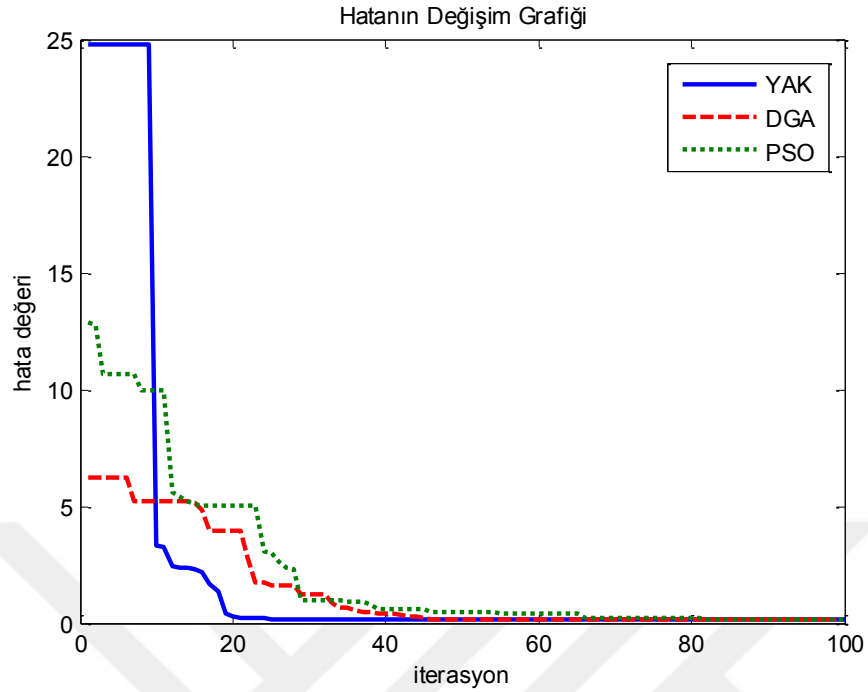
Şekil 3.18'de yer alan ortalama hata değerine ait hata gelişim grafiğine göre üç algoritma da oldukça yakın performanslar sergilemiştir. YAK algoritması çok az farkla en düşük ortalama hata değerini elde etmiştir. Üç algoritmada yaklaşık olarak ilk 50 iterasyondan sonra hatayı düşürememiştir. Bu durum tamsayı dereceli model yapısının süreci modellemede başarılı olamadığını ortaya koymaktadır.



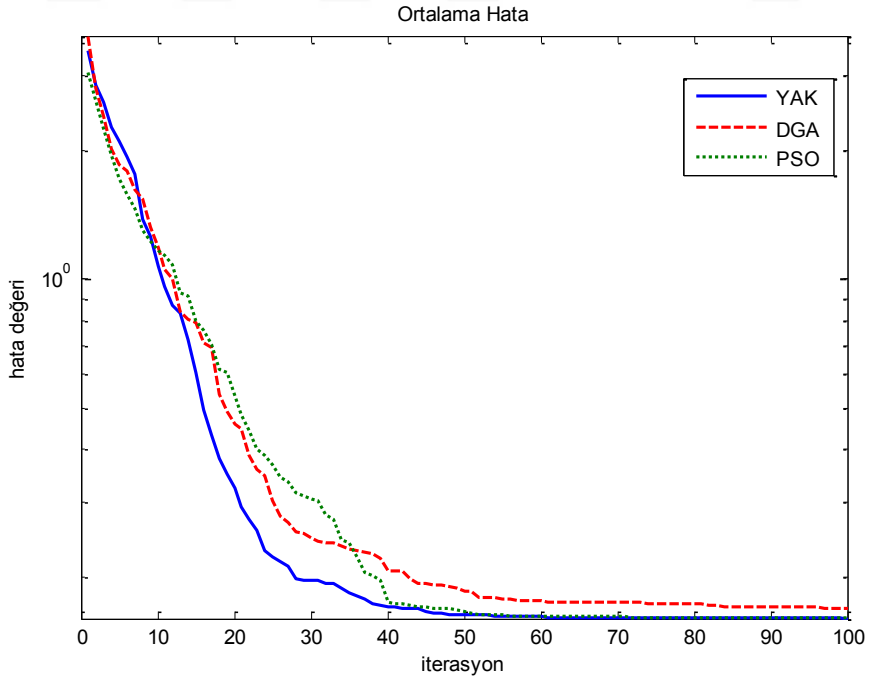
Şekil 3.15. G_3 süreci için kesir dereceli model yapısına (Gm2) ait en düşük hata değerinin iterasyonlara göre değişim grafiği



Şekil 3.16. G_3 süreci için kesir dereceli model yapısına (Gm2) ait ortalama hata değerinin iterasyonlara göre değişim grafiği



Şekil 3.17. G_3 süreci için tamsayı dereceli model yapısına (Gm1) ait en düşük hata değerinin iterasyonlara göre değişim grafiği



Şekil 3.18. G_3 süreci için tamsayı dereceli model yapısına (Gm1) ait ortalama hata değerinin iterasyonlara göre değişim grafiği

3.1.2 Kesir Dereceli Süreçler İçin Yapılan Modelleme Çalışmaları

Bu bölümde literatürde yer alan kesir dereceli süreç yapıları ele alınmış, bu süreçler için önceki çalışmalarda kullanılan Gm1 ve Gm2 model yapıları önerilmiştir. Bir önceki çalışma tamsayı dereceli süreçler için hem kesir hem de tamsayı dereceli model yapıları önerilmişti. Ayrıca YAK, DG ve PSO algoritmaları kullanılarak modellere ait parametre tahmini yapılmıştı. Bu bölümde ise kesir dereceli süreçler ele alınarak model yapılarının ve algoritmaların performanslarının kıyaslanması için örnek sayısı çeşitlendirilmiştir. Bu şekilde model yapıları ve algoritmaların performansları açısından daha net veriler elde edilmesi beklenmektedir. Aşağıda G_4 [75,110] ve G_5 [111] süreçleri yer almaktadır. Yapılan çalışmalardan, bu süreçler için önerilen model yapılarına ait olarak elde edilen parametre, hata ve standart sapma değerleri Tablo 3.3'de verilmiştir.

$$G_4 = \frac{1}{0.8s^{2.2} + 0.5s^{0.9} + 1} \tag{3.6}$$

$$G_5 = \frac{1}{s^{2.6} + 2.2s^{1.5} + 2.9s^{1.3} + 3.32s^{0.9} + 1} \tag{3.7}$$

Tablo 3.3. G_4 ve G_5 süreçleri için algoritmalar ile elde edilen model parametreleri

Süreç	Model	Algr.	b_0	a_2	a_1	a_0	k_2	k_1	ISE	SS
$G_4(s)$	Gm1	YAK	81.510	60.460	21.370	82.690	-	-	0.0454	0.0005
		DGA	68.1622	66.9236	49.3147	14.0797	-	-	0.0643	0.0008
		PSO	32.6006	24.0227	6.8567	33.2038	-	-	0.0643	0.0004
	Gm2	YAK	68.692	53.456	28.052	69.020	2.1552	0.9020	0.0083	0.0380
		DGA	39.2725	29.7080	22.7742	39.4805	2.2756	1.002	0.0025	0.0002
		PSO	70.000	51.1407	30.3985	69.998	2.1787	1.001	0.0114	0.033
$G_5(s)$	Gm1	YAK	15.300	72.880	93.690	15.350	-	-	0.0028	0.367
		DGA	15.0678	50.3624	93.5711	14.7594	-	-	0.0022	0.1913
		PSO	18.5195	98.0837	100.00	18.9020	-	-	0.0140	0.0961
	Gm2	YAK	22.2994	98.001	87.202	22.878	0.998	1.3839	0.0008	0.1244
		DGA	19.3210	40.7512	99.000	19.5246	1.2276	1.0269	0.0033	0.0755
		PSO	26.811	100	99.607	26.584	1.4075	0.8951	0.0053	0.0732

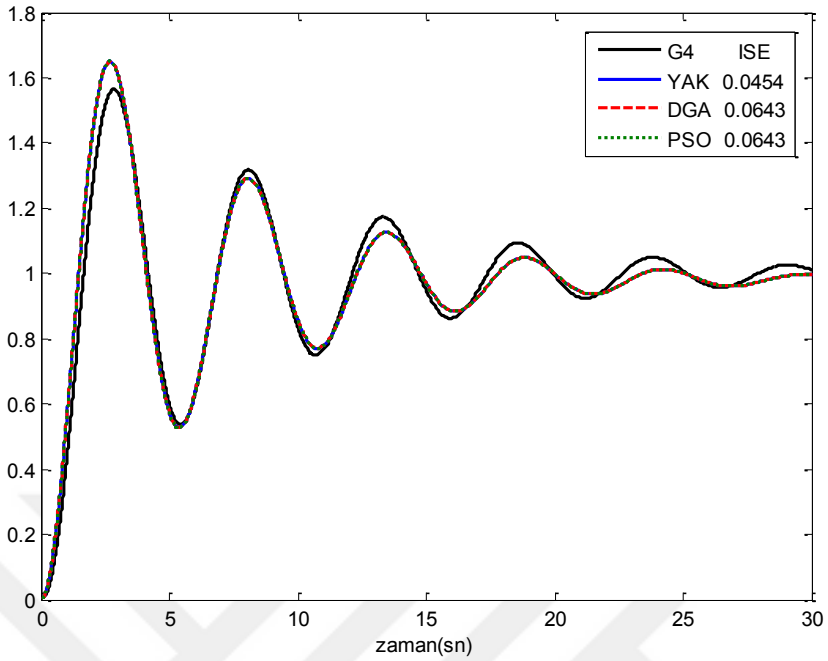
Tablo 3.3'te en düşük hata değerine sahip algoritmalar koyu renkle gösterilmiştir. Buna göre G_4 süreci için Gm1 modeli kullanılarak yapılan çalışmada 0.0454 hata değeri ile en başarılı sonuç YAK algoritması ile elde edilmiştir. DG ve PSO algoritmaları 0.0643 hata değeri ile YAK algoritmasına göre süreci modellemede daha az başarılı oldukları söylenebilir. Standart sapma değerleri ise birbirine oldukça yakındır. Gm2 model yapısı kullanılarak yapılan çalışma dikkate alındığında ise DG algoritmasının 0.0025 hata değeri ile en düşük hata değerine ulaştığı görülmektedir. Bu algoritmayı sırasıyla 0.0083

ve 0.0114 hata değeri ile YAK ve PSO algoritmaları izlemektedir. DG algoritması ayrıca 0.0002 standart sapma değeri ile sonuca ulaşması dikkat çekici bir durumdur. G_4 süreci için model yapıları ile elde edilen sonuçlar göz önünde bulundurulursa üç algoritma içinde daha düşük hata değerine kesir dereceli model yapısı kullanılarak ulaşılmıştır.

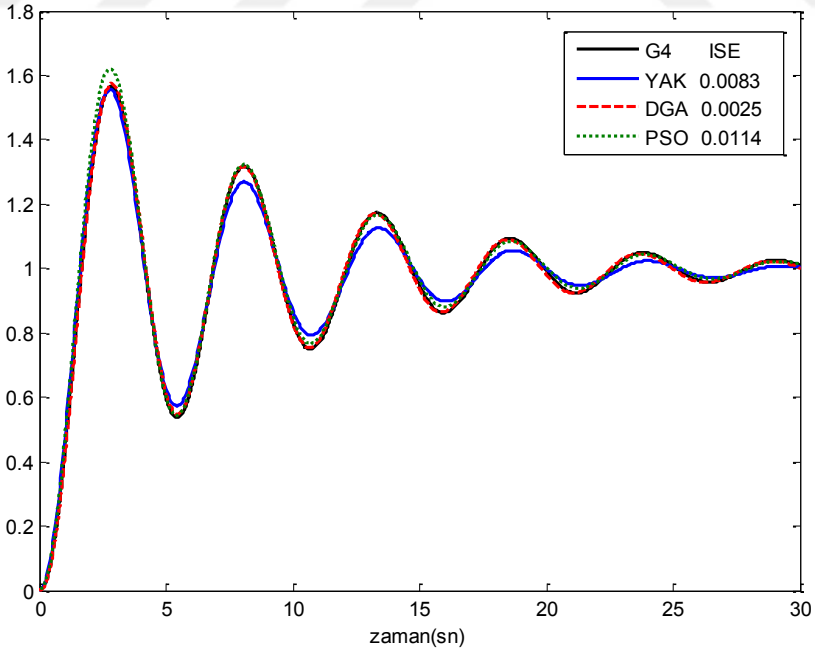
G_5 süreci için yapılan çalışmalar dikkate alındığında ise G_4 sürecine benzer sonuçların elde edildiği Tablo 3.3'ten görülmektedir. Tamsayı dereceli model yapısının kullanıldığı çalışmada DG algoritması 0.0022 ISE değeri ile süreci modellemede en başarılı sonucu elde etmiştir. DG algoritması için 30 koşturma sonunda 0.1913 standart sapma değeri elde edilmiştir. YAK algoritmasının 0.0028 hata değeri ile en başarılı ikinci sonucu elde ettiği çalışmada PSO algoritması ise 0.0140 hata değeri ile diğer iki algoritmaya göre daha başarısız olduğu Tablo 3.3'ten söylenebilir.

Kesir dereceli model yapısı kullanılarak elde edilen sonuçlara göre YAK algoritması ile yapılan çalışmada 0.0008 hata değeri ile sürecin karakteristik özelliklerini yansıtması açısından oldukça düşük hata değerine ulaşılmıştır. YAK algoritmasına ait standart sapma değeri 0.1244 olarak Tablo 3.3'te yer almaktadır. DG ve PSO algoritmalarının sırasıyla 0.0033 ve 0.0053 hata değeri ile sürecin modellenmesinde başarılı olduğu yine tablodan söylenebilir.

Her iki süreç göz önünde bulundurulduğunda kesir dereceli model yapısı ile yapılan çalışmalarda tamsayı dereceli model yapısı ile yapılan çalışmalara göre daha başarılı sonuçlar elde edildiği söylenebilir. Ayrıca kullanılan üç farklı algoritma yapısından YAK algoritmasının diğer iki algoritmaya göre başarılı performans sergilediği yine Tablo 3.3'ten çıkarılabilecek bir sonuçtur. Standart sapma değerleri göz önüne alındığında ise DG ve PSO algoritmalarının YAK algoritmasına göre daha başarılı olduğu yine tablodan ifade edilebilir. Süreçlere ait basamak cevapları ve hata gelişim grafikleri karşılaştırmalı olarak aşağıda sunulmuştur.



Şekil 3.19. G_4 için tamsayı dereceli model yapısına (Gm1) ait birim basamak cevapları



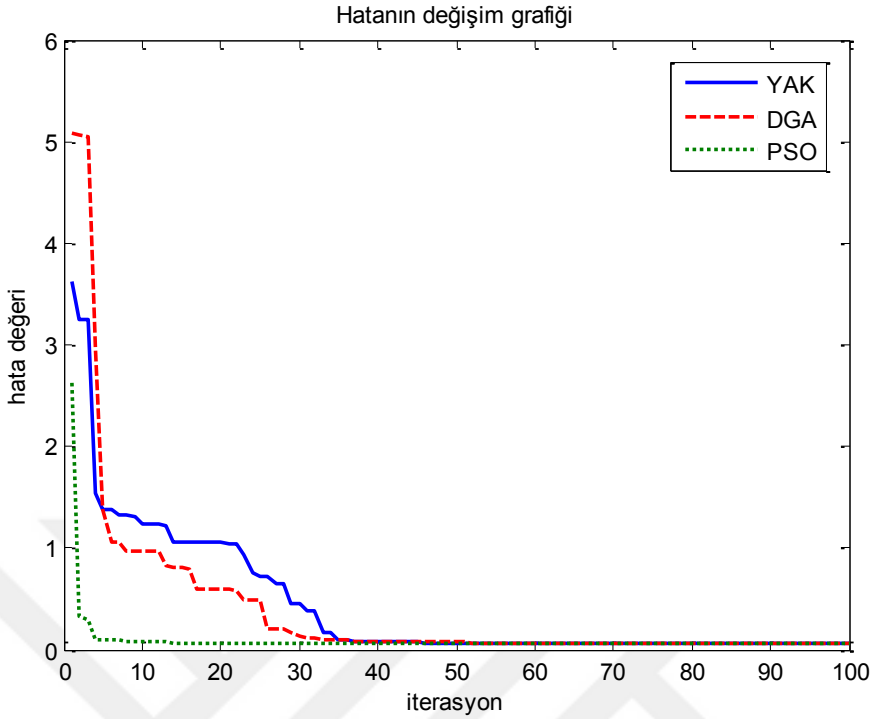
Şekil 3.20. G_4 için kesir dereceli model yapısına (Gm2) ait birim basamak cevapları

Şekil 3.19'da G_4 sürecine ait tamsayı dereceli model yapısı ile elde edilen basamak cevapları verilmiştir. YAK algoritması diğer iki algoritmaya göre daha başarılı performans sergilemiş, 0.0454 hata değeri ile süreci modellemiştir. DG ve PSO algoritmalarını ise 0.0643 hata değeri süreci modellemede ile aynı başarıyı göstermiştir.

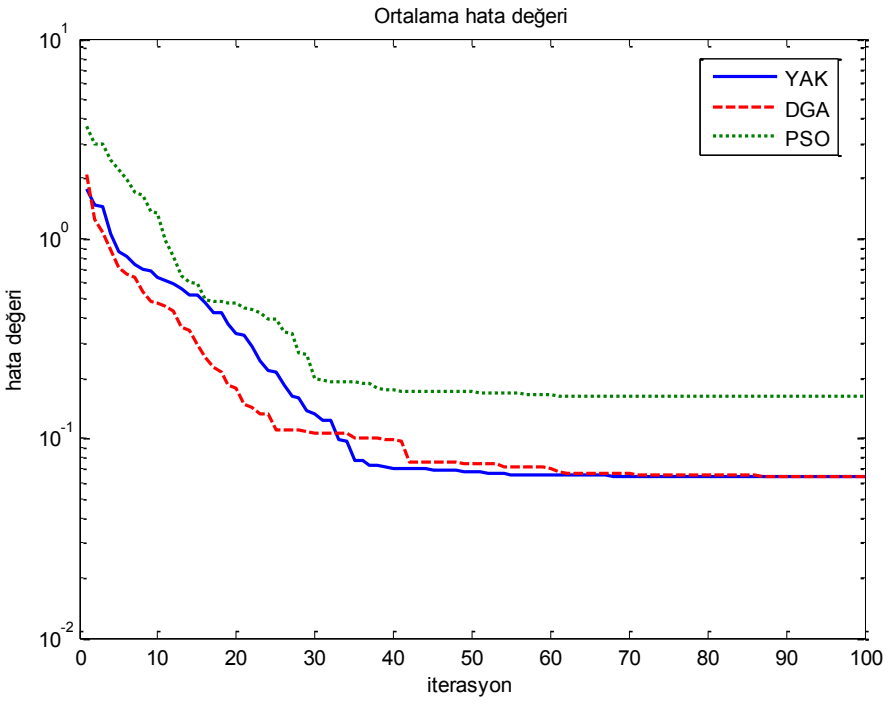
Şekil 3.20'de ise G_4 süreci için kesir dereceli model yapısına ait basamak cevapları yer almaktadır. Şekle göre DG algoritması 0.0025 hata değeri ile en iyi performansa sahip algoritma olurken, bu algoritmayı 0.0083 ISE değeri ile YAK ve 0.0114 ISE değeri ile PSO izlemiştir. İki model yapısı arasındaki başarıma göre, kesir dereceli modelin üstünlüğü bu iki şekilde net bir şekilde ortaya çıkmaktadır. YAK ve DG algoritmalarının iki model yapısında da başarılı olduğu PSO algoritmasının ise bu iki algoritmaya göre daha başarısız olduğu söylenebilir.

G_4 sürecinde tam sayı dereceli model yapısı için en iyi sonuca ait hata gelişim grafikleri Şekil 3.21'de, ortalama hata gelişim grafikleri ise Şekil 3.22'de yer almaktadır. En iyi sonuçların yer aldığı şekle göre PSO algoritmasının çok düşük iterasyon sayısında hatayı minimize ettiği dikkat çekicidir. DG ve YAK algoritmaları ise yaklaşık ilk 30 iterasyonda büyük ölçüde elde edilen hata değerine yaklaşmışlardır. Ortalama hata değerine ait grafiğe göre ise DG ve YAK algoritmaları benzer performans sergilemiş, birbirine oldukça yakın hata değeri elde etmiştir. PSO algoritmasının ise bu iki algoritmaya göre daha başarısız olduğu söylenebilir.

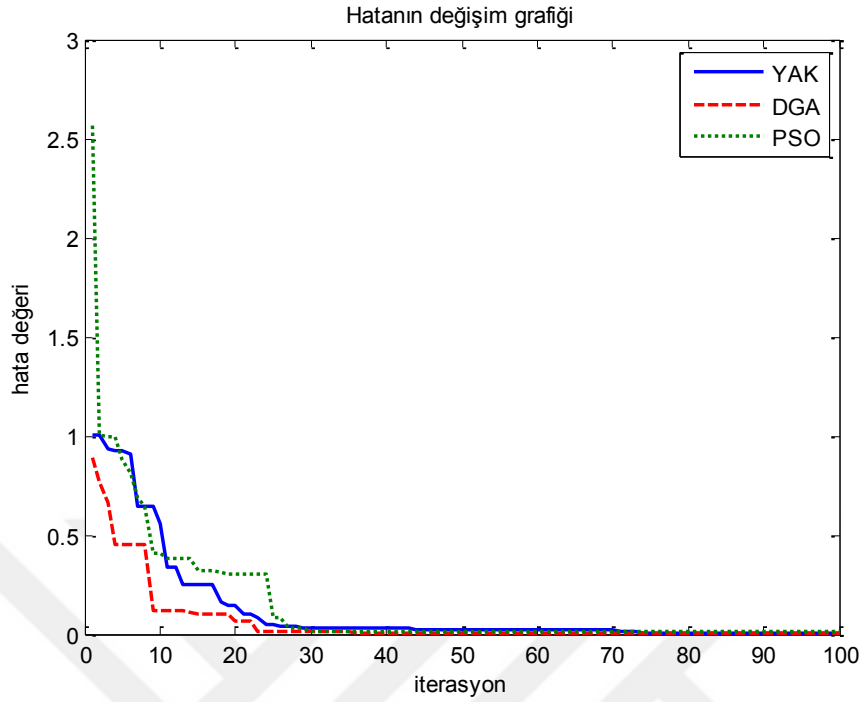
G_4 süreci için önerilen kesir dereceli model yapısına ait hata gelişim grafikleri Şekil 3.23 ve Şekil 3.24'de yer almaktadır. İlk grafik en iyi performansa ait hata indirgemeyi gösterirken, ikinci grafik ortalama hata indirgemeyi göstermektedir. Üç algoritma da yaklaşık olarak ilk 25 iterasyonda hatayı indirgemeyi başarmıştır. PSO algoritması DG ve YAK algoritmasına göre daha büyük hata değerinden optimizasyona başladığı yine grafikten çıkarılabilecek bir diğer sonuçtur. Şekil 3.24'e göre ortalama hatalar dikkate alındığında en düşük hata değeri DG algoritmasına aittir. YAK algoritması en düşük ikinci ortalama hata değerini elde ederken, PSO en yüksek ortalama hata değerini elde etmiştir.



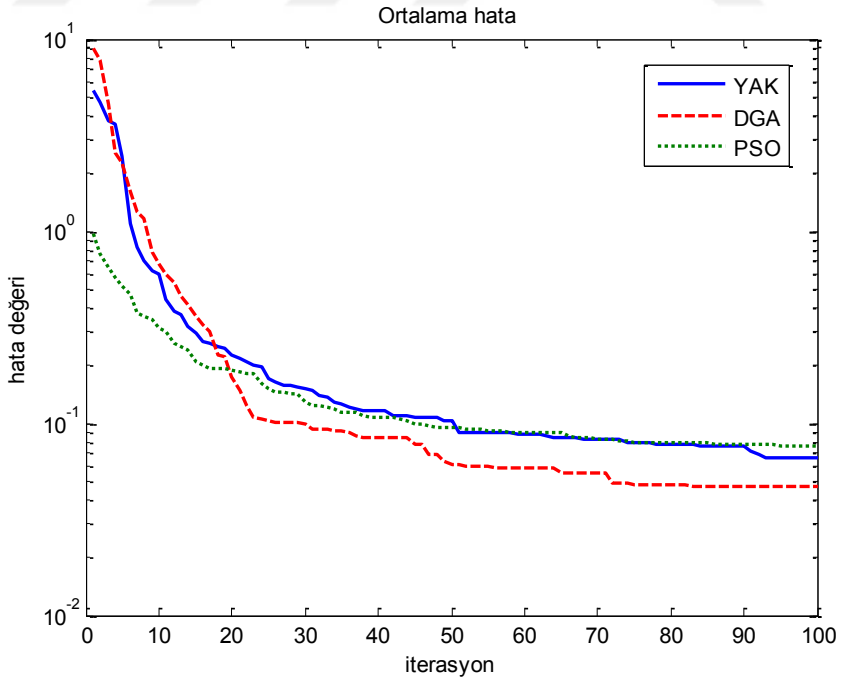
Şekil 3.21. G_4 süreci için tamsayı model yapısına (Gm1) ait en düşük hata değerinin iterasyonlara göre değişim grafiği



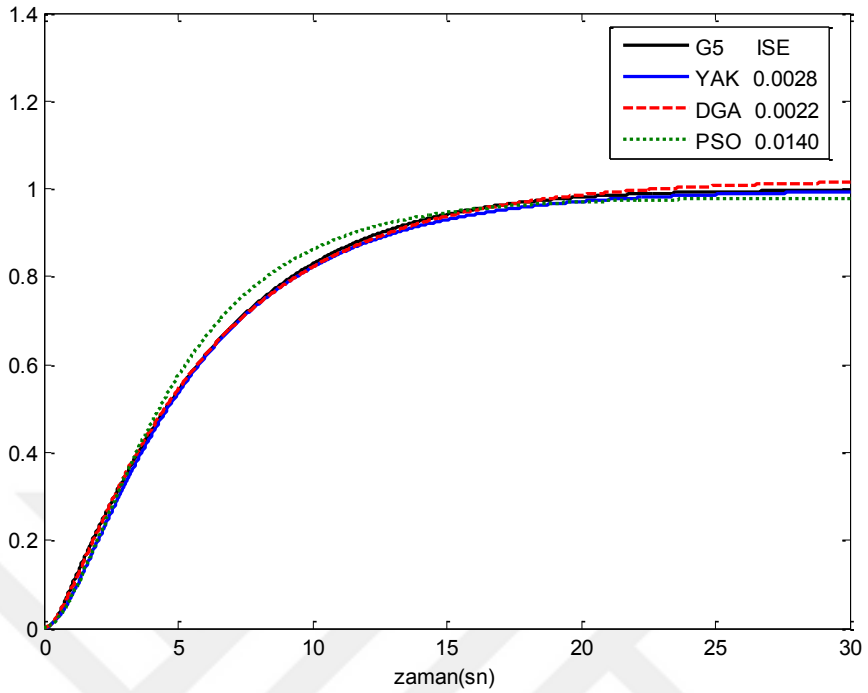
Şekil 3.22. G_4 süreci için tamsayı model yapısına (Gm1) ait ortalama hata değerinin iterasyonlara göre değişim grafiği



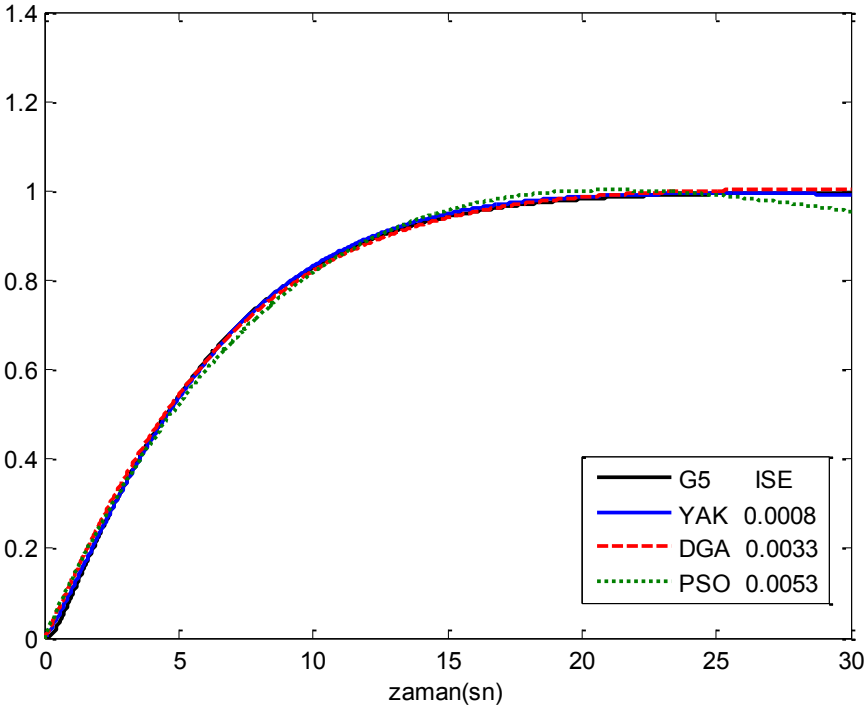
Şekil 3.23. G_4 süreci için kesir dereceli model yapısına (Gm2) ait en düşük hata değerinin iterasyonlara göre değişim grafiği



Şekil 3.24. G_4 süreci için kesir dereceli model yapısına (Gm2) ait ortalama hata değerinin iterasyonlara göre değişim grafiği



Şekil 3.25. G_5 için tamsayı dereceli model yapısına (Gm1) ait birim basamak cevapları



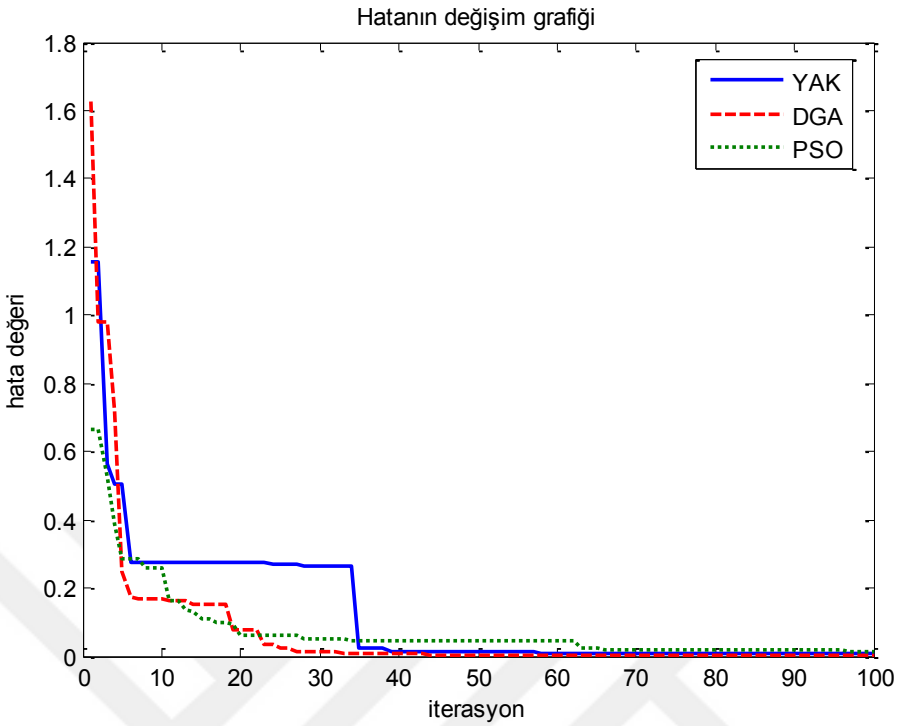
Şekil 3.26. G_5 için kesir dereceli model yapısına (Gm2) ait birim basamak cevapları

Şekil 3.25'de G_5 sürecine ait tamsayı modeller ile elde edilen birim basamak cevapları yer almaktadır. Algoritmaların süreci modelleme performanslarına göre DG algoritmasının 0.0022 ISE değeri ile en başarılı algoritma olduğu söylenebilir. YAK algoritmasının 0.0028 hata değeri ile DGA'nın başarımına yakın performans sergilemiş, PSO algoritması ise 0.0140 hata değeri ile bu iki algoritmanın başarımına uzak performans sergilemiştir. Algoritmaların performans farklılıkları basamak cevaplarının yer aldığı şekilde 5-15. saniyeler arasında açıkça görülmektedir.

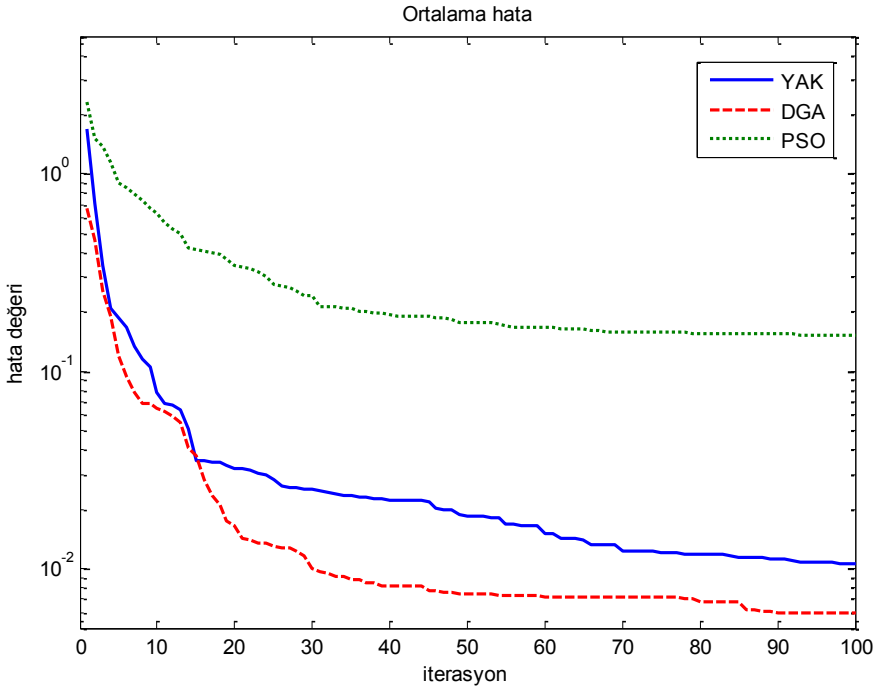
Kesir dereceli model yapılarına ait basamak cevapları ise Şekil 3.26'da yer almaktadır. Bu model yapısında elde edilen sonuçlara göre YAK algoritması 0.0008 gibi düşük hata değeri süreci modellemede oldukça başarılı sonuç elde etmiştir. DG algoritması 0.0033 hata değeri ile en iyi ikinci performansa sahipken, PSO algoritması 0.0053 ISE değeri ile performans açısından son sırada yer almaktadır. YAK ve PSO algoritmaları için kesir dereceli model yapısı ile elde edilen sonuçların, tam sayı dereceli model yapısı ile elde edilen sonuçlara göre daha başarılı olduğunu söylemek mümkündür. Ancak DG algoritmasının kullanıldığı çalışmada tam sayı dereceli modelin kesir dereceli model yapısına göre üstünlüğü dikkat çekici bir durumdur.

Tam sayı dereceli modeller için algoritmaların hata gelişim grafiklerinin yer aldığı Şekil 3.27 ve 3.28 sırasıyla en iyi sonuçları ve ortalama hata değerlerini göstermektedir. Ortalama hata değerleri göz önünde bulundurulduğunda PSO algoritmasının en yüksek hata değerini, DGA'nın ise en düşük hata değerini elde ettiği Şekil 3.28'den görülmektedir. En iyi performansların yer aldığı hata gelişim grafiğinde ise PSO ve DG algoritmalarının YAK algoritmasına göre daha hızlı olduğu ifade edilebilir. Üç algoritmanın da minimum hata değerine 100 iterasyondan önce ulaştığını söylemek mümkündür.

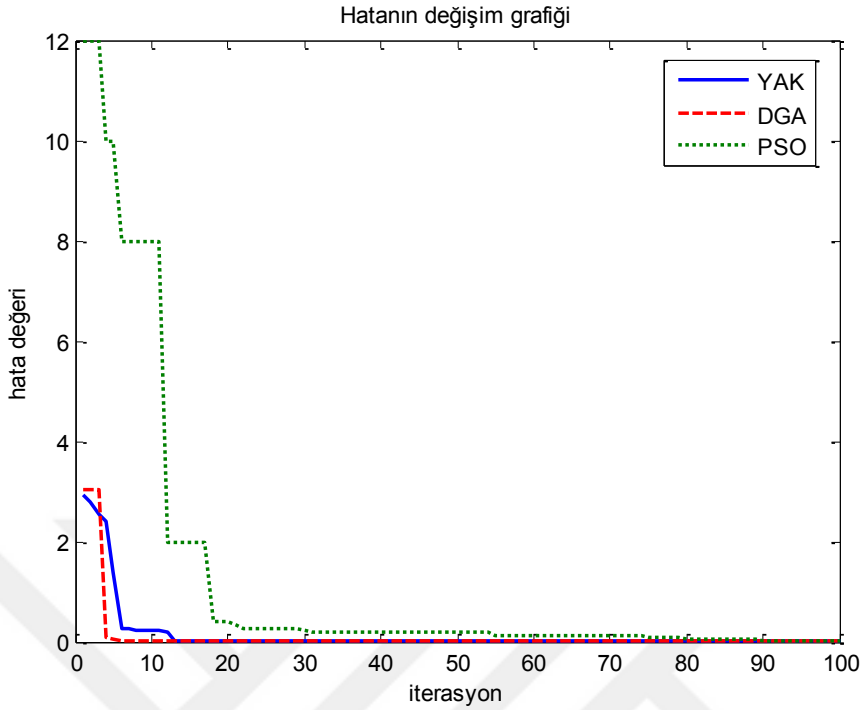
Kesir dereceli model sonuçlarının yer aldığı Şekil 3.29'da DG ve YAK algoritmalarının benzer performans sergiledikleri, hata değerini neredeyse ilk 10 iterasyonda minimize ettikleri görülmektedir. PSO algoritması daha yüksek hata değerinden başlayarak hatayı ilk 25 iterasyonda büyük ölçüde azalttığı yine aynı şekilden çıkarılabilecek başka bir sonuçtur. Ortalama hata değerlerinin yer aldığı Şekil 3.30'da ise üç algoritmanın da birbirine oldukça yakın başarım sergiledikleri, YAK algoritmasının çok az farkla daha düşük ortalama hata değerine ulaştığı görülmektedir.



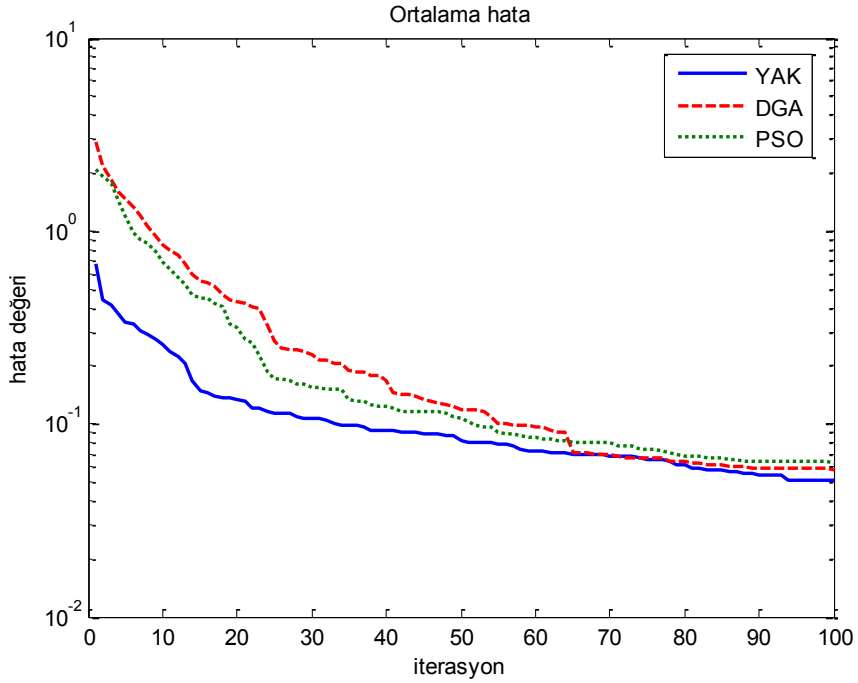
Şekil 3.27. G_5 süreci için tamsayı model yapısına (Gm1) ait anlık hata değerinin iterasyonlara göre değişim grafiği



Şekil 3.28. G_5 süreci için tamsayı model yapısına (Gm1) ait ortalama hata değerinin iterasyonlara göre değişim grafiği



Şekil 3.29. G_5 süreci için kesir dereceli model yapısına (Gm2) ait en düşük hata değerinin iterasyonlara göre değişim grafiği



Şekil 3.30. G_5 süreci için kesir dereceli model yapısına ait (Gm2) ortalama hata değerinin iterasyonlara göre değişim grafiği

4. BÖLÜM

YAPAY ZEKA ALGORİTMALARI İLE KONTROLÖR TASARIMI

Tez çalışmasının bu bölümünde kontrolör tasarımı örnekleri yer almaktadır. Tez çalışması bu bölümünde tamsayı ve kesir dereceli süreçler için ayrı ayrı hem tamsayı hem de kesir dereceli denetleyici tasarımları gerçekleştirilmiştir. Denetleyicilere ait parametreler tahmin edilirken modelleme çalışmalarında kullanılan algoritmalar tercih edilmiştir. Bu şekilde hem algoritmaların performansları hem de denetleyici yapıları arasındaki farklılıklar ortaya koyulmaya çalışılacaktır. Ayrıca modelleme sürecinden farklı olarak bu bölümde bazı örneklerde dört farklı performans indisi kullanılmıştır. ISE (OF_1) ve ITAE (OF_2) hata indislerinin yanı sıra, bu hatalar ile oturma süreleri (Ts) ve aşma miktarlarının (OS) da hesaba katıldığı OF_3 ve OF_4 şeklinde tanımlanan amaç fonksiyonları da kontrolör tasarımlarında kullanılmıştır. Böylece denetleyici tasarımında 4 farklı amaç fonksiyonu kullanımının etkisinin gözlenmesine imkan sağlanmıştır. Amaç fonksiyonları OF_1, OF_2, OF_3 ve OF_4, Eşitlik 4.1- 4.4'de verilmiştir.

$$OF_1 = ISE = \int_0^\infty e^2(\mathrm{t})\,\mathrm{dt} \tag{4.1}$$

$$OF_2 = ITAE = \int_0^\infty \mathrm{t}\,|\,e(\mathrm{t})\,|\,\mathrm{dt} \tag{4.2}$$

$$OF_3 = 10 \cdot ISE + 5 \cdot Ts + 3 \cdot OS \tag{4.3}$$

$$OF_4 = 10 \cdot ITAE + 5 \cdot Ts + 3 \cdot OS \tag{4.4}$$

4.1 Tamsayı Dereceli Süreçler İçin Yapılan Kontrolör Tasarımları

4.1.1 Yüksek Mertebeli Tamsayı Dereceli Sistemlerin Kontrolü için Tamsayı Dereceli PID Kontrolör Tasarımı

Bu bölümde tamsayı dereceli süreçler için kontrolör tasarımları gerçekleştirilmiştir. T_1 [70,112,113], T_2 [70,112] ve T_3 [70,112,114] örneklerinin ele alındığı ilk bölümde YAK, DG ve PSO algoritmaları kullanılarak klasik PID denetleyici parametreleri tahmin edilmeye çalışılmıştır. Performans kriteri olarak OF_2 amaç fonksiyonu kullanılmış, sonuçlar karşılaştırmalı olarak tablolar halinde verilmiştir. Tablodaki son sütunda, ITAE değeri ile birlikte sistem cevabına ait (%5) oturma süresi (Ts) ve aşım miktarı değerlerini de (OS%) içeren ve Eşitlik 4.4 ile verilen OF_4 fonksiyonuna ait hesaplanan sonuçları da belirtilmiştir.

$$T_1(s) = \frac{1}{\left(s^2+s+1\right)\left(s+2\right)^2} e^{-0.1s} \tag{4.5}$$

$$T_2(s) = \frac{1}{\left(s+1\right)^6\left(s^2+2\right)} \tag{4.6}$$

$$T_3(s) = \frac{1}{\left(s^2+2s+3\right)^3\left(s+3\right)} e^{-0.3s} \tag{4.7}$$

Eşitlik 4.5'de yer alan T_1 süreci dördüncü mertebeden ve ölü zamana sahip bir sistemdir. Literatürde yer alan [112] ve [113] numaralı kaynaklardan alınan bu örnek için elde edilen PID parametreleri, hata değerleri ve basamak özelliklerine ait veriler Tablo 4.1'de sunulmuştur. Tabloya göre hata değeri göz önünde bulundurulduğunda en başarılı sonuçlar YAK ve PSO kullanılarak yapılan çalışmalarda elde etmiştir. DG algoritması da 48.990 OF_4 değeri ile bu iki algoritmaya oldukça yakın performans sergilemiştir. Üç algoritmanın da literatürde yer alan çalışmalara göre çok daha başarılı olduklarını söylemek yine Tablo 4.1'den mümkündür. Benzer performans sergileyen bu algoritmaların literatürdeki çalışmalara göre daha düşük oturma süresi elde ettikleri ve ITAE hata değerinin de oldukça düşük olduğu gözden kaçırılmaması gereken bir durumdur. Şekil 4.1'de T_1 süreci için elde edilen birim basamak cevapları yer almaktadır. YAK ve PSO algoritmalarının performanslarının birbirine oldukça yakın

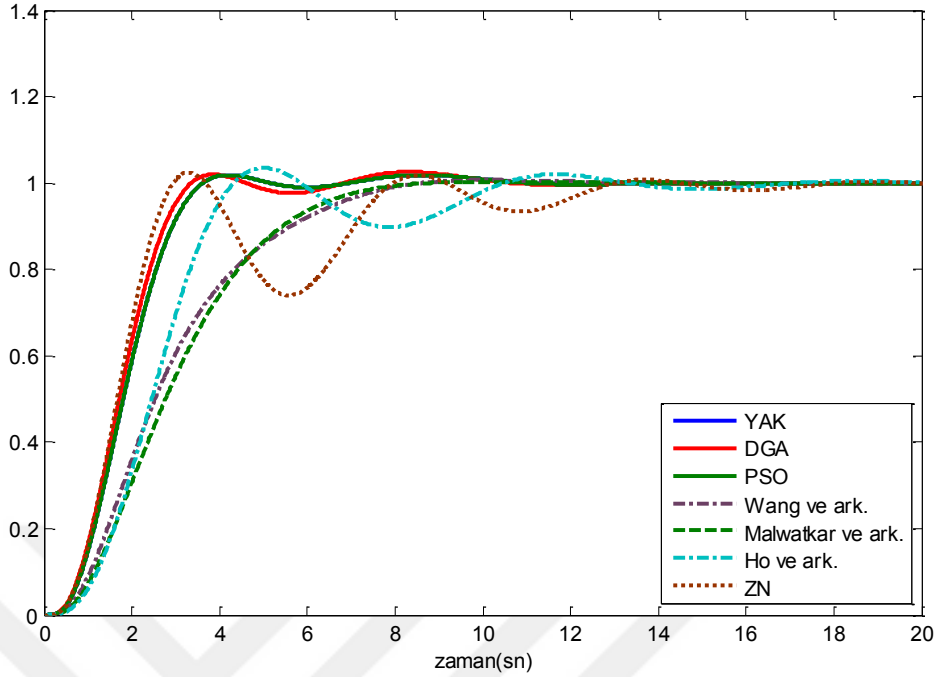
olması nedeniyle elde edilen grafiklerde üst üste oturmaktadır. Bu nedenle sadece YAK algoritmasına ait birim basamak cevabına grafikte yer verilmiştir. Şekil 4.1'e göre YAK algoritmasının yükselme ve oturma sürelerinin diğer çalışmalara göre daha düşük olduğu söylenebilir. Bu yorum, Tablo 4.1'de yer alan değerlerle de örtüşmektedir. Algoritmalarla elde edilen birim basamak cevaplarının tamamında ise küçük olsa da aşma miktarının olduğu görülmektedir. Kontrolü zor bu süreç için küçük miktarda aşmanın olması kaçınılmazdır. Bu nedenle üç algoritma için de sürecin kontrolünde oldukça başarılı performanslara sahip oldukları söylenebilir.

Tablo 4.1. T_1, T_2 ve T_3 süreçleri için algoritmalar ile elde edilen PID kontrolör parametreleri ve basamak cevabı özellikleri

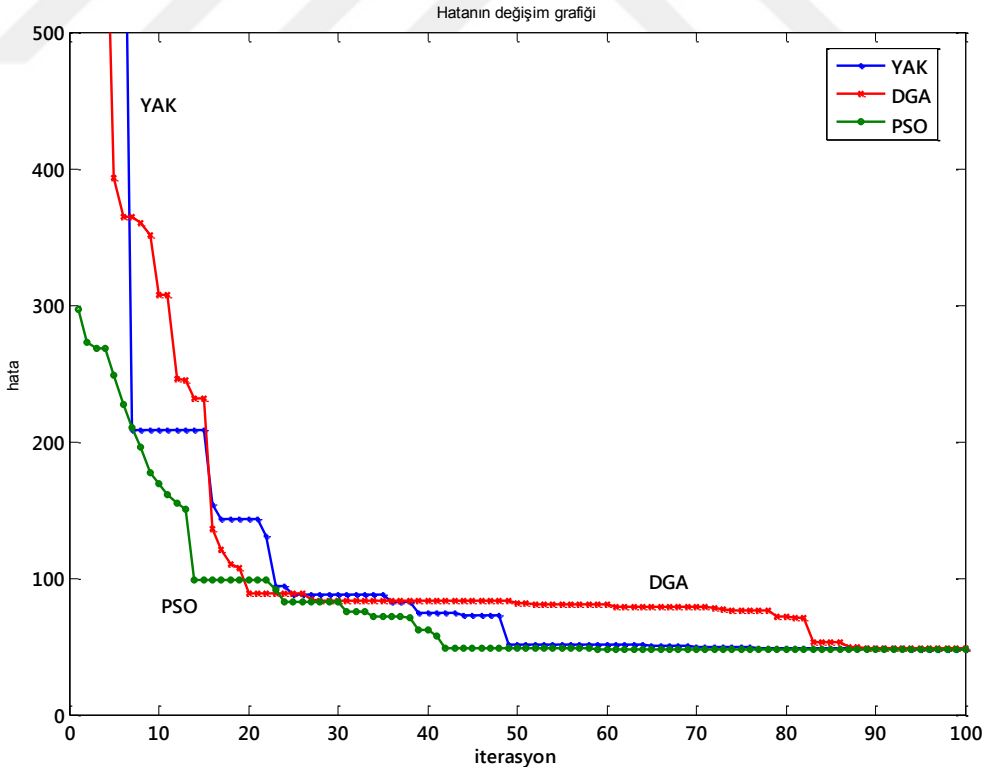
Süreç	**Yöntem**	**PID Parametreleri**			**Basamak cevabı özelikleri**			
		Kp	**Ki**	**Kd**	**ITAE**	**Ts(sn)**	**OS%**	**OF_4**
T_1	**Malwatkar [112]**	1.3039	1.3104	1.3351	6.1999	6.2875	0.3143	94.3796
	Wang [113]	1.5030	1.3660	1.7150	6.1833	6.6230	0.6808	96.9898
	Ho [112]	2.1470	1.4840	0.7770	7.2006	9.4044	3.5306	129.6200
	ZN	4.2331	1.6277	2.7522	8.0067	11.6359	2.4391	145.5636
	PSO [70]	2.7106	2.2147	2.8134	2.6440	3.1886	1.8618	47.9686
	DGA	2.9251	2.3271	3.1413	2.6426	2.9806	2.5539	48.9901
	YAK [70]	2.7121	2.2136	2.8118	2.6452	3.1883	1.8582	**47.9683**
T_2	**Malwatkar [112]**	0.78	0.3708	0.8964	69.269	20.782	16.257	845.37
	Wang [112]	0.5996	0.6299	2.281	216.52	53.313	48.14	2576.16
	Ho [112]	1.959	0.5727	1.676	48.045	23.778	30.106	689.66
	ZN	2.3702	0.4356	3.2245	29.329	21.716	14.466	445.27
	PSO [70]	1.9742	0.49193	3.033	23.575	13.058	4.9997	**316.04**
	DGA	1.9077	0.3912	2.8708	23.6589	13.0878	4.8458	316.565
	YAK [70]	1.9738	0.49229	3.0306	23.576	13.055	5.0021	**316.04**
T_3	**Kappa-Tau [114]**	4.5	10.9756	0.1485	38.0459	16.6627	0	463.7730
	Karimi [114]	4.93	15.6013	0.6162	15.6880	8.6716	0.6982	202.3331
	Wang [114]	17.562	22.485	14.13	10.0070	9.1222	5.1766	161.2102
	ZN	43.1985	14.37	32.465	35.1072	22.8617	0.6884	467.446
	PSO [70]	29.177	25.729	20.632	6.0179	4.6798	1.3326	**87.576**
	DGA	29.2155	25.8843	20.9660	5.3545	4.6608	1.4625	87.6693
	YAK [70]	29.148	25.706	20.573	6.0205	4.6845	1.3162	**87.576**

Şekil 4.2'de ise algoritmalara ait en düşük hata değeri için elde edilen hata gelişim grafikleri yer almaktadır. YAK ve DGA'nın PSO algoritmasına göre çok daha yüksek hata değerinden optimizasyona başladığı ancak hatayı ilk 10 iterasyonda oldukça düşürdüğü görülmektedir. YAK yaklaşık ilk 50 iterasyonda minimum hata değerini elde ederken, DGA ilk 80 ve PSO ilk 40 iterasyonda minimum hata değerine ulaşmıştır. Bu sürece ait hata gelişim grafiklerine göre YAK algoritmasının diğer iki algoritmaya göre hatayı daha düşük iterasyonlarda hızlı biçimde azalttığını söylemek yanlış olmayacaktır. Tablo 4.1'de T_2 süreci için elde edilen verilere göre algoritmaların performanslarının T_1 sürecine benzer olduğu görülür. Ancak hem literatürde yer alan çalışmalar, hem de tez çalışmasında yer alan bu üç örnek arasında en büyük aşma miktarına bu örnekte rastlanmaktadır. T_2 sürecinin yüksek mertebeli olması nedeniyle kontrolü de oldukça zordur. Bu nedenle oturma süreleri de diğer süreçlere göre oldukça uzundur. Ancak YAK, DGA ve PSO ile yapılan çalışmalarda hem aşma miktarı hem de oturma süreleri literatürde yer alan çalışmalarla kıyaslandığında daha küçüktür. ITAE hata değeri dikkate alındığında da yine algoritmaların performanslarının literatüre göre oldukça iyi olduğu söylenebilir. YAK ve PSO algoritmaları bu sürecin kontrolünde 316.04 OF_4 değeri ile en başarılı sonuca ulaşırken, DG algoritması 316.565 OF_4 değeri ile benzer başarım sergilemiştir. Literatürde yer alan çalışmalarda Malwatkar 845.37, Wang 2576.16 ve Ho 689.66 OF_4 değeri ile algoritmaların performanslarından oldukça uzak bir başarım sergilemiştir.

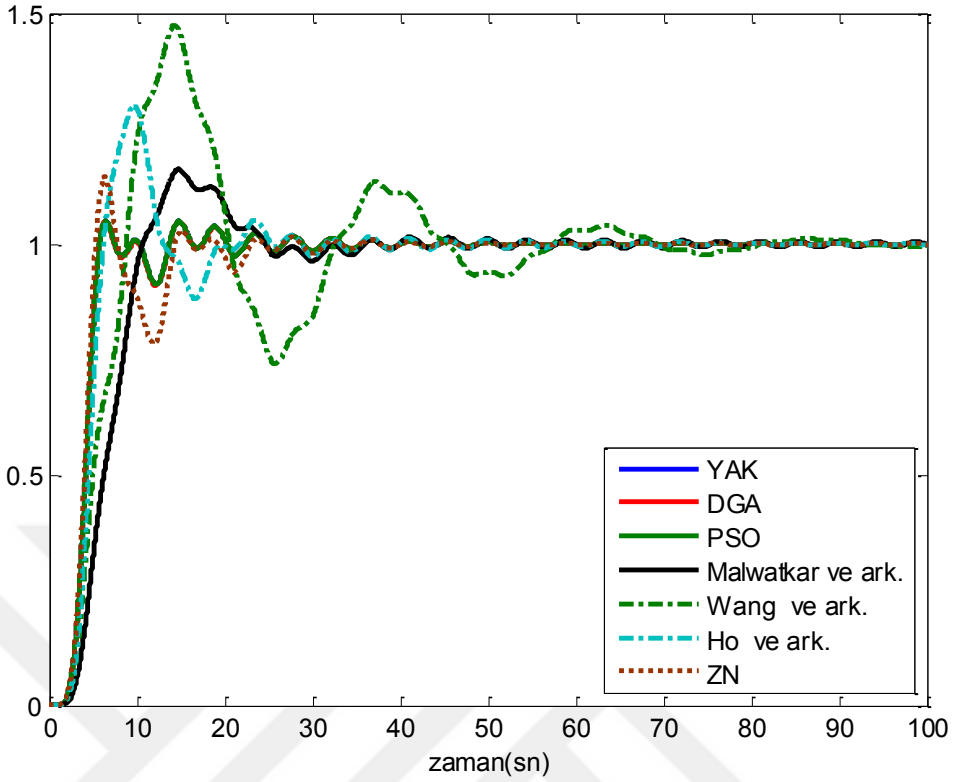
T_2 sürecine ait birim basamak cevaplarına Şekil 4.3'de yer verilmiştir. Tablo 4.1.'de yer alan sayısal ifadeler doğrultusunda algoritmalar ve literatürde yer alan çalışmaların performans farklılıkları bu şekil üzerinde daha net bir şekilde görülmektedir. YAK algoritması ile DG algoritmasının oldukça yakın performans sergilemesi nedeniyle basamak cevaplarına ait çizimlerin neredeyse üst üste geldiği görülmektedir. Aşma miktarı göz önünde bulundurulduğunda bu iki çizimin literatürdeki çalışmalara olan üstünlüğü göze çarpmaktadır. Özellikle Wang ve Ho'ya ait çalışmalarla yapılan kıyaslamada bu farklılık oldukça belirgindir. Oturma sürelerinde de benzer ifadeleri söylemek yanlış olmayacaktır.



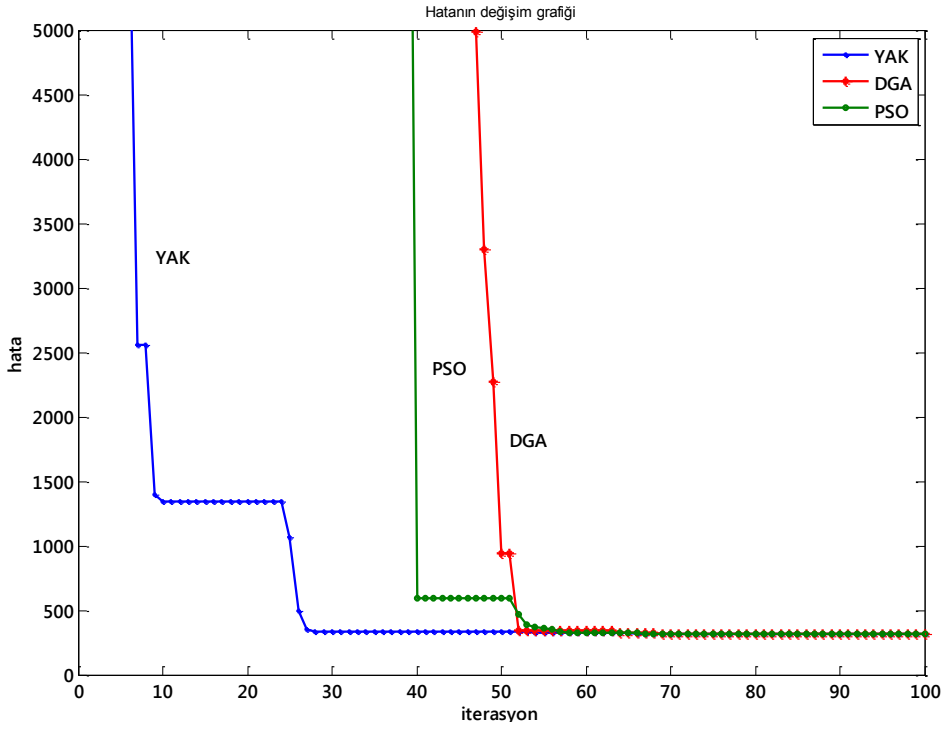
Şekil 4.1 Farklı yöntemler için T_1 sürecine ait birim basamak cevapları



Şekil 4.2 T_1 süreci için algoritmalara ait hata gelişim grafikleri



Şekil 4.3 Farklı yöntemler için T_2 sürecine ait birim basamak cevapları

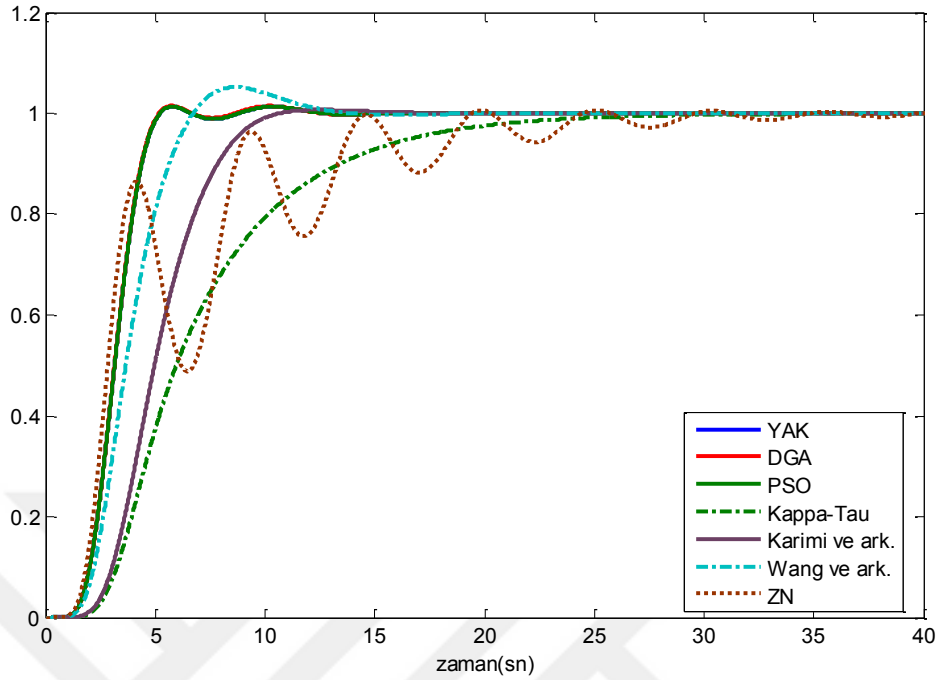


Şekil 4.4 T_2 süreci için algoritmalara ait hata gelişim grafikleri

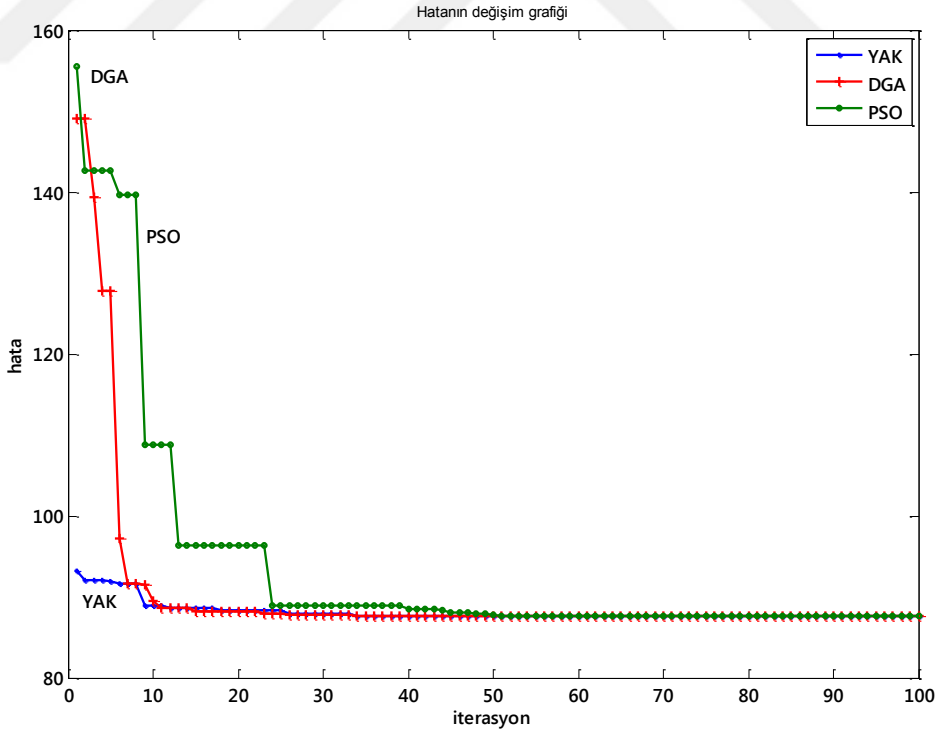
Şekil 4.4 ise algoritmaların sürecin kontrolünde sergiledikleri performansa bağlı olarak elde ettikleri hata gelişim grafiklerini vermektedir. Üç algoritmada oldukça yüksek hata değerinden optimizasyona başlamış, ve hızlı bir şekilde sonuca ulaşmıştır. İterasyon sayısının 100 olarak seçilmesine rağmen, YAK algoritmasının ilk 30, PSO ve DG algoritmalarının ilk 50 iterasyonda nihai çözüme ulaşmaları algoritmaların gücünün açık bir göstergesi olarak ifade edilebilir. YAK algoritmasının diğer iki algoritmaya göre T_1 sürecinde olduğu gibi daha hızlı hata indirgemeye sahip olduğu da yine Şekil 4.4'ten açıkça görülmektedir. Bu örnek için yakınsama performansları açısından en kötü performansa sahip algoritma DGA'dır.

T_3 süreci için basamak cevap özellikleri ile PID denetleyici parametre değerleri Tablo 4.1'de yer almaktadır. Bu süreçte elde edilen sonuçlar ilk iki süreçte yer alan sonuçlara benzer özellikler taşımaktadır. Yüksek mertebeli ve gecikmeli süreç olan T_3 süreci için algoritmalar kullanılarak elde edilen denetleyici performansları birbirine oldukça yakındır. PSO ve YAK algoritmaları 85.576 OF_4 değeri ile en başarılı denetleyici yapısını elde ederken, DGA 87.669 OF_4 değeri ile bu algoritmalara oldukça yakın performans sergilemiştir. Tez çalışmasında önerilen algoritmaya dayalı denetleyici yapılarının literatürdeki örneklerden daha başarılı sonuçlar sergilediği Tablo 4.1'den de açıkça görülmektedir. Özellikle ITAE hata değeri ile oturma süreleri göz önüne alındığında bu farklılık oldukça açık bir şekilde görülmektedir.

Bu süreç için YAK ve PSO algoritmalarının basamak cevapları oldukça yakın elde edildiği için sadece YAK ve DG algoritmalarına ait basamak cevapları dikkate alınarak Şekil 4.5'de grafik olarak sunulmuştur. Şekil 4.5'e göre DG ve YAK algoritmalarının kullanımı ile benzer basamak cevaplarının elde edildiği, aşma miktarının ve oturma sürelerinin düşük olduğu ifade edilebilir. Özellikle literatürde yer alan Kappa-Tau, Karimi ve Wang'a ait yöntemlerle elde edilen denetleyici yapıları göz önünde bulundurulduğunda, bu çalışmada elde edilen PID denetleyici yapılarının oldukça başarılı sonuçlar ortaya koyduklarını söylemek yanlış olmayacaktır.



Şekil 4.5 Farklı yöntemler için T_3 sürecine ait birim basamak cevapları



Şekil 4.6 T_3 süreci için algoritmalara ait hata gelişim grafikleri

Şekil 4.6'da T_3 süreci için optimizasyon sonucunda elde edilen hata-iterasyon değişim grafikleri yer almaktadır. Şekle göre YAK algoritmasının diğer algoritmalara göre daha düşük hata değerinden optimizasyona başladığı ve hata değerini ilk 10 iterasyon içerisinde indirgediği görülmektedir. DG ve PSO algoritmalarının ise daha yüksek ve birbirine yakın bir değerden optimizasyona başladığı ve hata değerini 10-25. iterasyonlar arasında minimize ettikleri görülebilmektedir. Bu durum üç algoritmanın da bu süreçteki hatayı gelişim hızının oldukça başarılı olduğunu göstermektedir.

Bu bölüme ilişkin genel bir değerlendirme olarak,T_1, T_2 ve T_3 süreçleri ile ilgili yapılan denetleyici çalışmalarında kullanılan üç algoritmanın da başarılı sonuçlar verdiği, özellikle hata gelişim hızlarının oldukça iyi olduğu söylenebilir. Ayrıca süreçler için önerilen OF_4 amaç fonksiyonun literatürde yer alan ve sadece hata değerinin kullanıldığı çalışmalara göre daha başarılı ve arzu edilir sistem cevabı özellikleri sunduğu da görülmüştür. Tez çalışmasında yer alan diğer kontrolör tasarımı örneklerinde dört farklı amaç fonksiyonu kullanımı ile elde edilen avantaj ve dezavantajlar ayrıntılı olarak incelenecektir.

4.1.2 Yüksek Mertebeli Tamsayı Dereceli Sistemlerin Kontrolü için Tamsayı ve Kesirli Dereceli PID Kontrolör Tasarımları

Tez çalışmasının bu aşamasında T_4 [15,115] ve T_5 [15,71] olarak tanımlanan Eşitlik 4.8 ve Eşitlik 4.9'da yer alan yüksek mertebeli iki sistem ele alınmıştır. İlk üç örnekten farklı olarak bu sistemler için tamsayı dereceli PID denetleyicinin yanı sıra kesirli dereceli PID (FOPID) denetleyici yapıları da önerilmiştir. OF_1, OF_2, OF_3 ve OF_4 olmak üzere dört farklı amaç fonksiyonu bu örnekler için kullanılmıştır. Bu şekilde hem denetleyici yapılarının hem de farklı amaç fonksiyonlarının kullanımında oluşan avantajlar-dezavantajlar hakkında detaylı bir inceleme sunulmuştur.

Denetleyici parametrelerini tahmin etmek için diğer çalışmalarda olduğu gibi YAK, DG ve PSO algoritmaları tercih edilmiştir. Kullanılan denetleyici ve amaç fonksiyonu sayısının fazla olması nedeniyle elde edilen birim basamak cevapları ve hata gelişim grafikleri her algoritma için ayrı ayrı verilmiştir. Bu şekilde grafiklerin daha net bir şekilde analiz edilebileceği düşünülmüştür. Tez çalışmasının bundan sonraki aşamalarında yer alan (modellenen yapılar için kontrolör tasarımı bölümüne kadar) çalışmalarda bu yöntem izlenmiştir.

$$T_4(s) = \frac{27}{(s+1)(s+3)^3} \tag{4.8}$$

$$T_5(s) = \frac{25.2s^2 + 21.2s + 3}{s^5 + 16.58s^4 + 25.41s^3 + 17.18s^2 + 11.70s + 1} \tag{4.9}$$

Yapılan çalışmalarda, literatüre uygun olarak T_4 süreci için birim basamak cevabı [0-10] saniye aralığında elde edilirken T_5 için [0-2] saniye aralığında elde edilmiştir. Her iki örnek için de K_p, K_i ve K_d için arama aralığı [0-100], λ ve μ için arama aralığı [0-2] olarak belirlenmiştir. Bu süreçler için algoritmaların parametre değerleri objektif bir kıyaslama yapabilmek amacıyla eşit seçilmiştir. Parametre değerleri Tablo 4.2'de verilmiştir.

Tablo 4.2 T_4 ve T_5 süreçleri için algoritmalara ait kontrol parametreleri

Kontrol Parametreleri	**PSO**	**DGA**	**YAK**
Popülasyon(koloni) büyüklüğü	10		
Hızlanma ve öğrenme katsayıları	c1=c2=2	-	-
Limit	-	-	PID:15 FOPID:25
Çaprazlama ve mutasyon oranı	-	0.9 / 0.3	-
İterasyon sayısı	100		
Parametre aralığı	[0-100]		

T_4 sürecine ait YAK algoritması ile elde edilen FOPID ve PID denetleyici parametreleri Tablo 4.3'de sunulmuştur. Süreç için YAK algoritması kullanılarak elde edilen optimum performans karakteristikleri ise Tablo 4.4'de verilmiştir.

Tabloya göre bütün amaç fonksiyonları için FOPID denetleyici yapısının tamsayı dereceli denetleyici yapısına göre daha başarılı sonuçlar elde ettiğini söylemek mümkündür. FOPID denetleyicinin sadece amaç fonksiyonları göz önüne alındığında değil, aşma ve oturma süreleri göz önüne alındığında da PID denetleyiciye olan üstünlüğü dikkat çekicidir. Örneğin Tablo 4.4'de OF_3 amaç fonksiyonuna ait performans karakteristikleri incelendiğinde FOPID kontrolör 6.3913 değeri elde ederken, PID kontrolör 13.2966 değerini elde etmiştir. FOPID denetleyiciye ait ISE hata değeri, oturma süresi ve aşma miktarı sırasıyla 0.2904, 0.6975 ve 0 iken, PID denetleyiciye ait bu değerler sırasıyla 0.5890, 1.4078 ve 0.1225 olarak elde edilmiştir. Standart sapma değerlerinde de FOPID denetleyicinin genel olarak daha başarılı görülmektedir. Tablo 4.4'de ve diğer algoritmalara ait benzer tablolarda yer alan OF

değeri, "OF=(10×hata)+(5×ts)+(3×OS)" ifadesi ile hesaplanan ve "hata" değeri olarak da ilgili çalışmada elde edilen ISE ya da ITAE değerinin kullanıldığı bir karakteristik değerdir.

Tablo 4.3 T_4 süreci için YAK algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

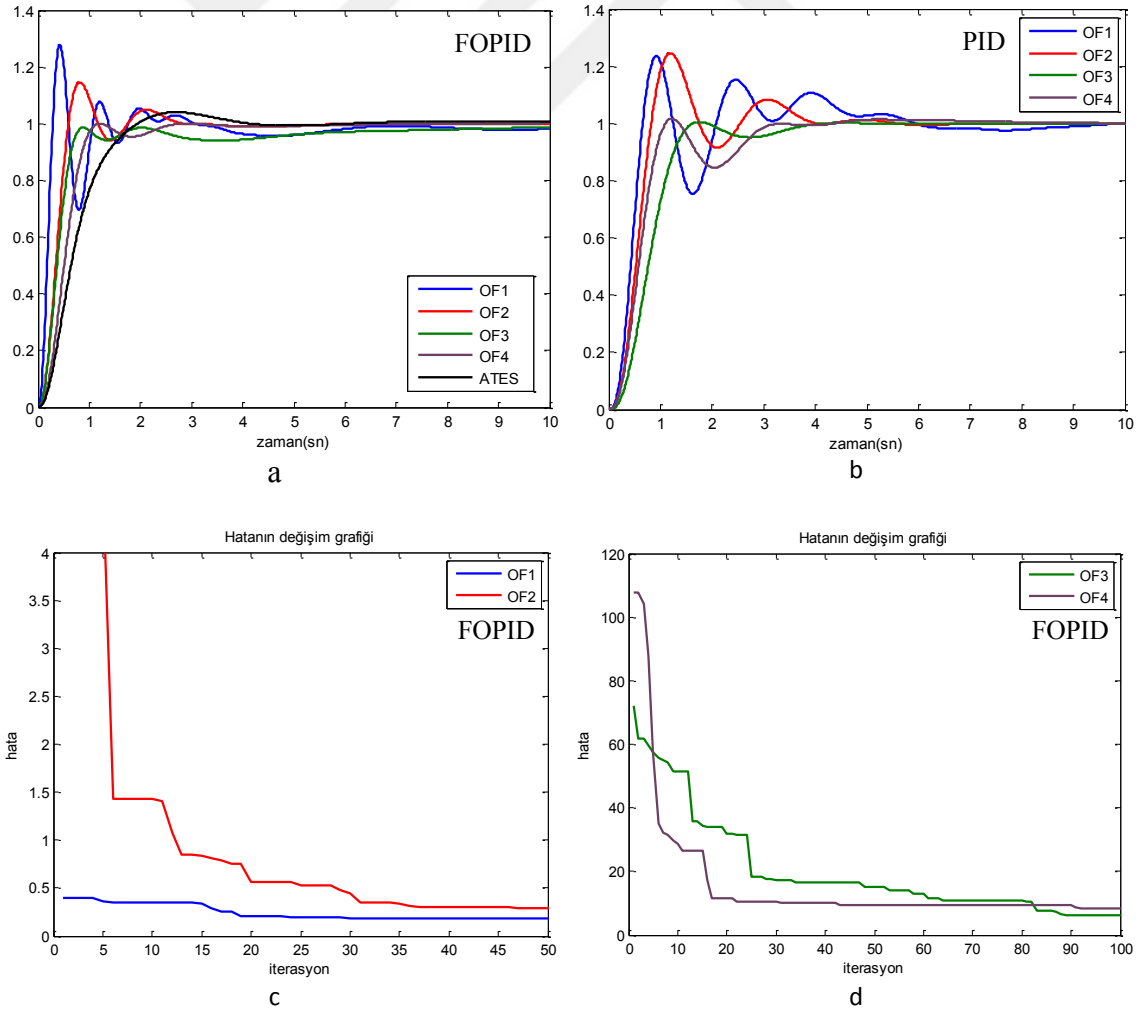
Kontrolör	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	8.5559	9.5539	8.1297	1.4960	0.519
		OF_2	7.204	3.224	3.108	0.981	1.336
		OF_3	6.4561	1.2572	2.298	1.4510	1.0264
		OF_4	1.248	1.604	1.775	1.029	1.248
	Ateş [115]		2.937	1.277	1.080	1.070	1.340
PID	Tez Çalışması	OF_1	2.8673	3.9968	3.6422	-	-
		OF_2	3.5602	2.9363	2.3302	-	-
		OF_3	2.1109	1.1951	1.0706	-	-
		OF_4	2.5211	1.4519	2.0540	-	-

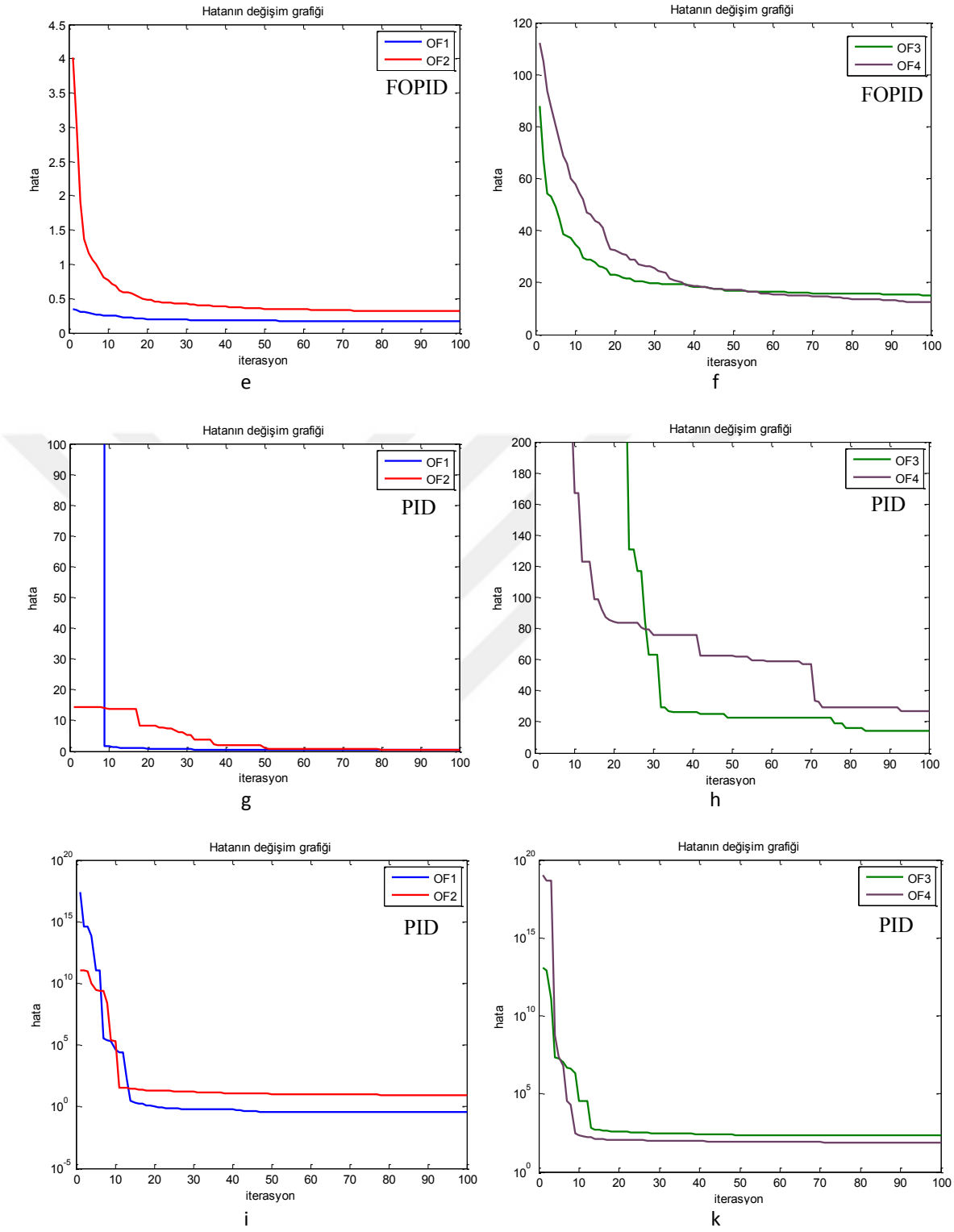
Tablo 4.4. T_4 sürecine ait YAK algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

Kontr.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.1738		105.865	0.1810	2.7304	30.1586	0.0005
		OF_2		0.4198	57.083	0.3761	1.5176	15.099	0.0009
		OF_3	0.2904		6.3913	0.4804	0.6975	0	0.0028
		OF_4		0.3882	8.6418	0.6459	0.9519	0	0.0035
	Ateş [115]		-	0.7481	23.392	1.1393	1.6312	3.4320	-
PID	Tez Çalışması	OF_1	0.3759	-	88.4883	0.4072	4.4980	20.7462	0.0041
		OF_2		0.6984	92.7732	0.5114	3.4730	22.8081	0.0654
		OF_3	0.5890		13.2966	0.5847	1.4078	0.1225	0.9401
		OF_4	-	0.7868	26.4829	0.6469	2.7360	1.6448	5.4798

Şekil 4.7'de T_4 süreci için elde edilen birim basamak cevapları ve hata gelişim grafikleri yer almaktadır. FOPID denetleyici kullanılarak elde edilen birim basamak cevapları (a), PID denetleyiciye ait birim basamak cevapları (b) olarak gösterilmiştir. İki grafiğe göre sadece hata değerinin yer aldığı OF_1 ve OF_2 amaç fonksiyonlarının yükselme sürelerinin OF_3 ve OF_4 amaç fonksiyonlarına göre daha hızlı olduğu ancak aşma oranlarının da daha fazla ve oturma sürelerinin de daha yüksek olduğu görülmektedir. FOPID denetleyiciye ait oturma sürelerinin PID denetleyiciye ait oturma sürelerine göre neredeyse yarısı kadar olduğu ve FOPID denetleyicinin daha hızlı cevap verebildiği Şekil 4.7'den açıkça görülmektedir.

Şekil 4.7'de yer alan c,d,e,f grafikleri FOPID denetleyiciye, g,h,i,k grafikleri ise PID denetleyiciye ait hata gelişim grafiklerini göstermektedir. Şekildeki c,d ve g,h grafikleri minimum hata için elde edilen değişimi gösterirken, e,f ve i,k grafikleri ise ortalama hataya ait değişimi sunmaktadır. Amaç fonksiyonlarının optimizasyona başladığı ve bitirdiği değerlerin çok farklı olması nedeniyle OF_1, OF_2 ile birlikte, OF_3 ve OF_4 ise bu iki fonksiyondan farklı şekilde gruplandırılmıştır. Bu şekilde hatanın iterasyona bağlı değişiminin daha açık bir şekilde görülebilmesi amaçlanmıştır. Ayrıca elde edilen en düşük hata değerine bağlı grafikler ile ortalama hata grafikleri alt alta verilerek çalışmanın analizi daha görsel hale getirilmiştir. Şekil 4.7.c'deki grafik FOPID denetleyici için en düşük hataya ait çalışmadaki gelişim grafiğini verirken, (e) grafiği aynı denetleyici için ortalama hatanın gelişim grafiğini göstermektedir. Benzer şekilde (h) grafiği PID denetleyici için en düşük hatanın gelişim grafiğini, (k) ise ortalama hatanın gelişim grafiğini göstermektedir.





Şekil 4.7 T_4 süreci için YAK algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

Bu grafiklere göre FOPID denetleyici PID denetleyiciye göre çok daha küçük değerlerden optimizasyona başlamış ve daha düşük hata değerlerini elde etmiştir. Özellikle PID kontrolöre ait (k) grafiğinde hata değerinin 10^{13} ve 10^{19} gibi çok yüksek değerlerden başlayarak 10^{2} gibi değerlere kadar indirgemesi dikkat çekicidir. Bu durum kuşkusuz algoritmanın gücünü de göstermektedir. OF_1 ve OF_2 hata indislerinin kullanıldığı çalışmalarda ise doğal olarak OF_3 ve OF_4 indislerine göre daha düşük hata değerinden başlamıştır. Fonksiyonların hesabında kullanılan katsayılar göz önüne alındığında bu durumun kaçınılmaz olduğu gayet açıktır. Ortalama hata değerlerinin yer aldığı hata gelişim grafiklerinin ise en düşük hatanın yer aldığı grafiklere benzer değişimler sergilediği görülmektedir.

DG algoritması kullanılarak T_4 süreci için önerilen FOPID ve PID denetleyicileri için elde edilen parametre değerleri Tablo 4.5'de sunulmuştur. Tablo 4.6'da ise denetleyicilerin performansları hakkında detaylı bilgi veren hata, aşma ve oturma süresi gibi karakteristik değerler yer almaktadır. Tabloda verilen değerler 30 kez koşma sonunda elde edilen en iyi hata değerlerine ait verilerdir. Buna göre. OF_1 ve OF_2 performans indislerinin kullanıldığı çalışmalarda PID denetleyicinin daha başarılı sonuçlar elde ettiği, ancak sadece hatanın değil aynı zamanda oturma süresi ve aşma miktarının da optimize edildiği OF_3 ve OF_4 indislerinde FOPID denetleyicinin üstünlüğü göze çarpmaktadır. Bu durumun nedeni OF_1 ve OF_2 indislerinde FOPID denetleyicinin aşma miktarının PID denetleyiciye göre yüksek olmasıdır. Ancak aşmanın sıfırlandığı OF_3 ve OF_4 indislerinde ise kesir dereceli denetleyici ile oldukça başarılı sonuçlar elde etmiştir.

Tablo 4.5. T_4 süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

Kontrolör	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	8.7890	6.1266	7.7409	1.4881	0.7314
		OF_2	9.999	4.0466	3.9068	1.4051	1.0329
		OF_3	6.9169	1.6499	3.1338	1.4751	1.0297
		OF_4	5.2860	1.6020	2.0285	1.3646	1.0930
PID	Tez Çalışması	OF_1	2.9126	4.0428	4.2622	-	-
		OF_2	2.7202	1.5592	1.4473	-	-
		OF_3	2.3602	1.3072	1.8842	-	-
		OF_4	2.478	2.0200	1.1356	-	-

Tablo 4.6. T_4 sürecine ait DG algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

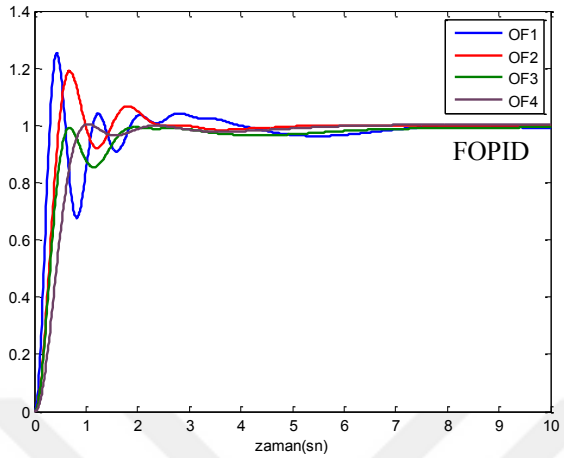
Kontr.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.1803	-	95.8495	0.1898	2.9278	26.4690	0.0003
		OF_2	-	0.2815	69.6772	0.3063	2.0013	18.9519	0.0060
		OF_3	0.2426	-	10.1494	0.3821	1.5448	0	0.0057
		OF_4	-	0.4389	8.4571	0.5572	0.8136	0	0.0060
PID	Tez Çalışması	OF_1	0.3746	-	94.0272	0.3919	4.3599	22.8274	0.0066
		OF_2		0.5278	45.1831	0.7082	2.8342	8.5779	0.0029
		OF_3	0.5556		31.2477	0.8322	2.9258	3.6876	0.7114
		OF_4		1.0594	26.4147	0.6732	3.1373	0.0447	0.8651

Şekil 4.8(a)-(b)'de T_4 süreci için DG algoritması kullanılarak elde edilen FOPID-PID denetleyicilere ait basamak cevapları yer almaktadır. Şekildeki c,d ve g,h grafikleri minimum hata için elde edilen değişimi gösterirken, e,f ve i,k grafikleri ise ortalama hataya ait değişimi sunmaktadır. FOPID denetleyicinin yer aldığı (a) grafiğine göre OF_3 ve OF_4 amaç fonksiyonları kullanılarak elde edilen basamak cevaplarında aşma miktarının olmadığı, oturma sürelerinin ise daha kısa olduğu açık bir şekilde görülmektedir. PID denetleyiciye ait (b) grafiğinde de benzer bir durum söz konusudur. OF_1 ve OF_2 amaç fonksiyonları ile elde edilen grafiklerin aşma miktarları daha fazla, oturma süreleri daha uzundur. Özellikle OF_1'e ait basamak cevabı dikkatle incelendiğinde daha osilasyonlu bir çıkış elde edildiği görülmektedir.

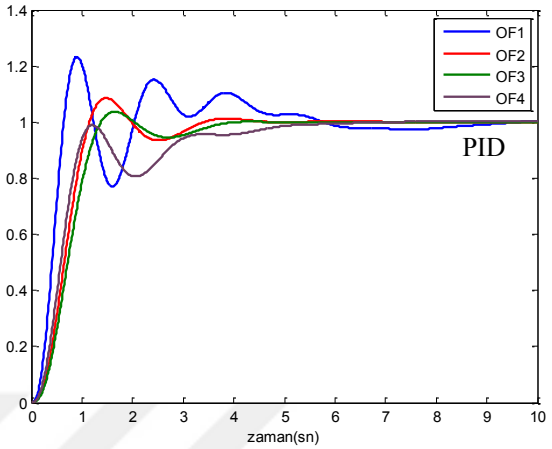
İki denetleyici yapısı kıyaslandığı zaman FOPID kontrolörün cevap süresinin ve oturma sürelerinin kısa oluşu göze çarpmaktadır. Her ne kadar PID kontrolörün bazı indislerde aşma miktarı daha küçük olsa da bu örnek için FOPID denetleyicinin daha başarılı olduğunu söylemek yanlış bir ifade olmayacaktır. Şekil 4.8'de verilen c,d,e,f grafikleri FOPID denetleyiciye ait hata gelişim grafiklerini göstermektedir. OF_1 ve OF_2 kullanılarak elde edilen hata gelişim grafiklerine göre DGA'nın minimum hata değerine olukça yakın bir değerden optimizasyona başladığı görülmektedir. 30 koşma sonunda elde edilen ortalama hata değerinin, en düşük hata değerine ait grafikle benzediği görülmektedir. Bu durum Tablo 4.6'da verilen standart sapma değeri ile örtüşmektedir. OF_3 ve OF_4 indislerinin yer aldığı (d) grafiğinde ise yüksek hata değerinden optimizasyona başlayan DGA, hatayı çok hızlı bir şekilde minimize edebilmiştir.

PID denetleyiciye ait hata gelişim eğrileri g,h,i,k grafiklerinde sunulmuştur. FOPID denetleyicinin aksine PID denetleyicide hata daha yüksek değerlerden optimizasyona başlamış daha büyük iterasyon değerlerinde minimum değerine ulaşmıştır. Özellikle (k)

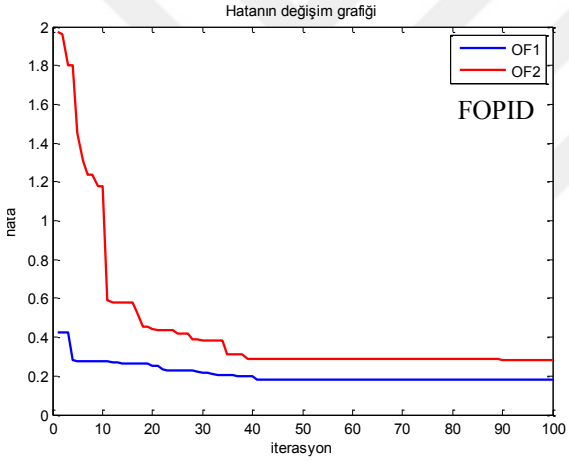
grafiği dikkate alındığında neredeyse 100. iterasyona kadar hata değeri DGA tarafından minimize edilmeye çalışılmıştır.



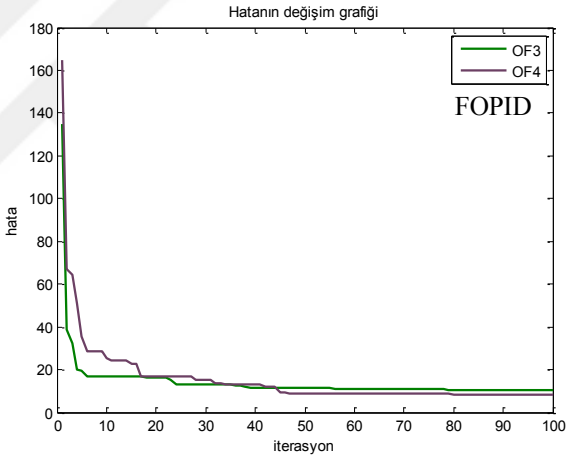
a



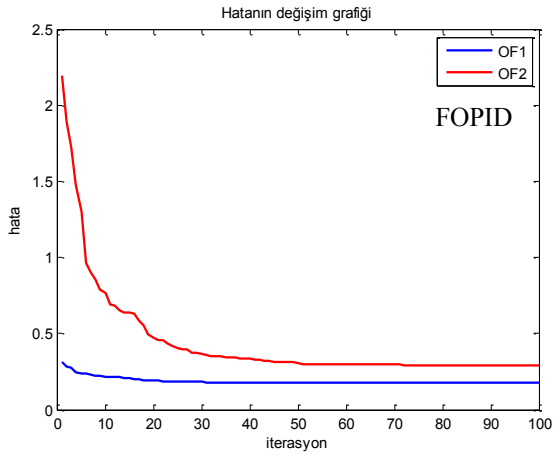
b



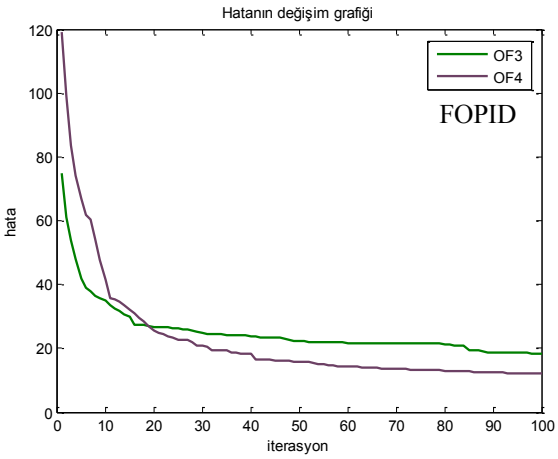
c



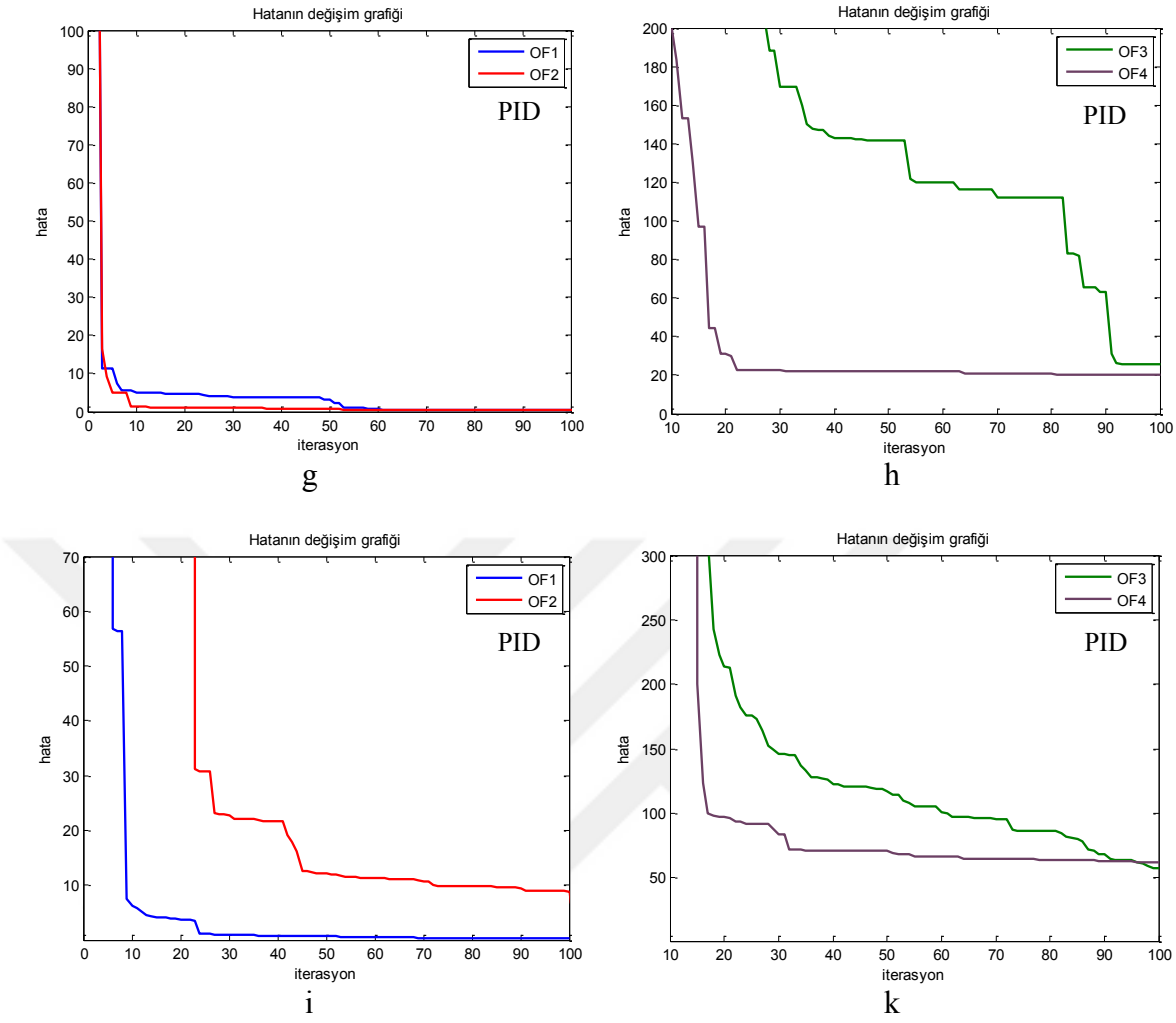
d



e



f



Şekil 4.8. T_4 süreci için DG algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

$T_4(s)$ süreci için PSO algoritması ile elde edilen denetleyici parametreleri ve temel performans karakteristikleri Tablo 4.7 ve Tablo 4.8'de sunulmuştur. PSO algoritması ile elde edilen sonuçlar için ilk iki algoritmada belirtilen sonuçlara benzer ifadeler olduğu söylenebilir. İki denetleyici arasında kıyaslamada FOPID denetleyicinin OF_2 amaç fonksiyonu dışında PID denetleyiciye göre daha başarılı olduğu tablodan görülebilmektedir. OF_2 amaç fonksiyonunda FOPID denetleyici 95.9135, PID denetleyici ise 71.3350 hata değeri elde etmiştir. Bu sonucun ortaya çıkmasına PID denetleyicinin ITAE değeri ile, aşma miktarını FOPID denetleyiciye göre daha düşük değerde elde etmesinin neden olduğu söylenebilir.

Diğer iki algoritmada olduğu gibi OF_3 ve OF_4 amaç fonksiyonlarının aşma miktarını ve yükselme süresini azaltmadaki başarısı dikkat çekicidir. FOPID denetleyicide iki fonksiyon için aşma miktarı 0 iken, PID denetleyicide 2.3252 ve 0.9760 olarak elde edilmiştir. Sadece hata indisinin kullanıldığı çalışmalarda aşma miktarı neredeyse %30 olarak elde edilmiştir. Standart sapma değerlerinde de FOPID denetleyicinin daha başarılı olduğu yine Tablo 4.8'den görülmektedir.

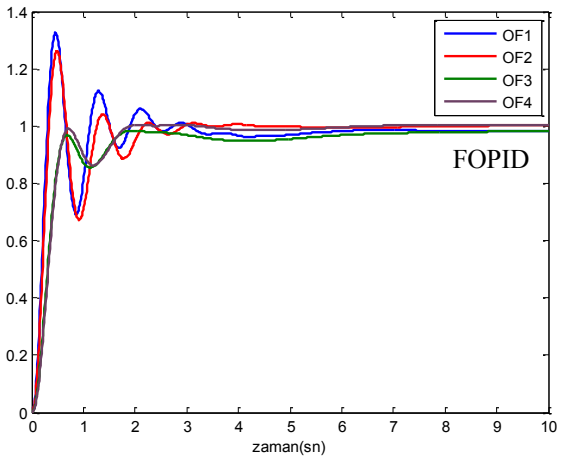
Tablo 4.7. T_4 süreci için PSO algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

Kontrolör	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	8.912	9.6124	8.0937	1.4315	0.4990
		OF_2	10.00	2.652	7.038	1.086	7.038
		OF_3	6.8917	1.3557	2.9716	1.4960	0.9948
		OF_4	6.567	1.784	2.999	1.115	1.451
PID	Tez Çalışması	OF_1	2.8068	6.0776	5.4288	-	-
		OF_2	3.3582	2.1339	1.9767	-	-
		OF_3	2.2976	1.4041	1.4933	-	-
		OF_4	2.1156	1.2818	1.1400	-	-

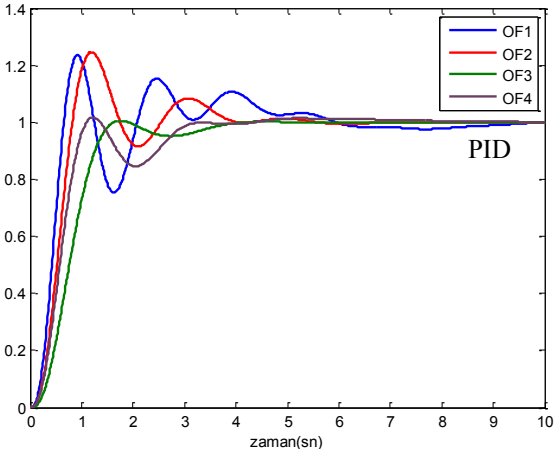
Tablo 4.8. T_4 sürecine ait PSO algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

Kontr.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.1946		117.6916	0.1906	2.2712	34.7965	0.0001
		OF_2		0.8333	95.9135	0.2133	2.0115	25.841	0.0021
		OF_3	0.2506		10.1234	0.3936	1.5235	0	0.0074
		OF_4		0.2507	10.580	0.4093	1.6145	0	0.0041
PID	Tez Çalışması	OF_1	0.3849	-	128.8391	0.3445	7.4663	29.2193	0.0021
		OF_2		0.6130	71.3350	0.5657	2.5992	17.4028	0.0086
		OF_3	0.5238	-	18.1828	0.7959	1.1939	2.3252	1.298
		OF_4	-	0.5750	15.5255	0.9149	1.3696	0.9760	0.6549

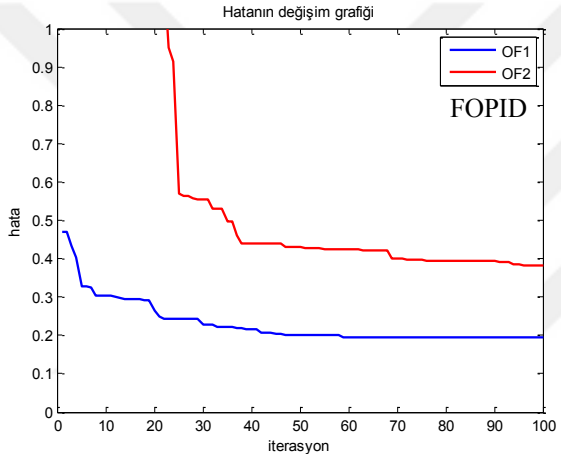
Şekil 4.9'da T_4 süreci için denetleyici yapılarına ait PSO algoritması kullanılarak elde edilen basamak cevapları ve hata iterasyon değişimleri yer almaktadır. Şekildeki c,d ve g,h grafikleri minimum hata için elde edilen değişimi gösterirken, e,f ve i,k grafikleri ise ortalama hataya ait değişimi sunmaktadır. FOPID denetleyiciye ait (a) grafiğinde OF_1 ve OF_2 'ye ait basamak cevaplarının oldukça yakın olduğu, iki hata indisinde de aşmanın yüksek olduğu görülmektedir. OF_3 ve OF_4 hata indislerine ait basamak cevaplarında ise aşma miktarının sıfıra indirildiği açıkça görülmektedir. PID denetleyiciye ait (b) grafiği ise OF_1 ve OF_2'ye ait basamak cevaplarında aşma miktarının FOPID denetleyiciye nispeten daha az olduğu ancak oturma ve yükselme sürelerinin daha uzun olduğu görülmektedir.


OF1
OF2
OF3
OF4
FOPID
zaman(sn)

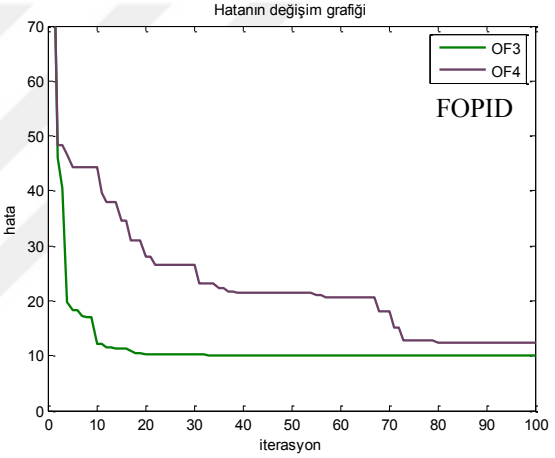
a


OF1
OF2
OF3
OF4
PID
zaman(sn)

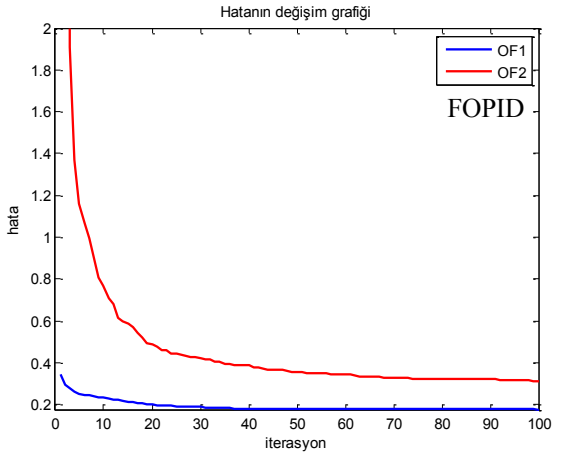
b


Hatanın değişim grafiği
OF1
OF2
FOPID
hata
iterasyon

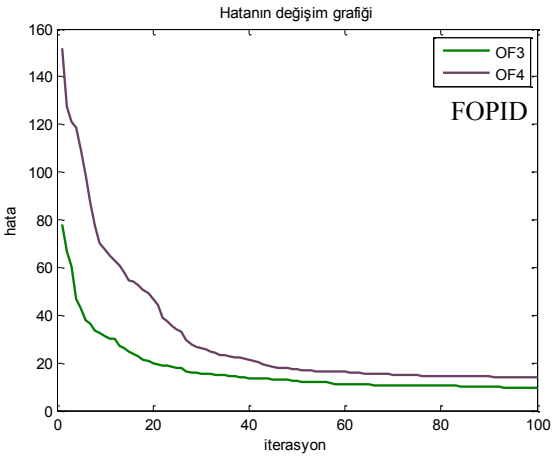
c


Hatanın değişim grafiği
OF3
OF4
FOPID
hata
iterasyon

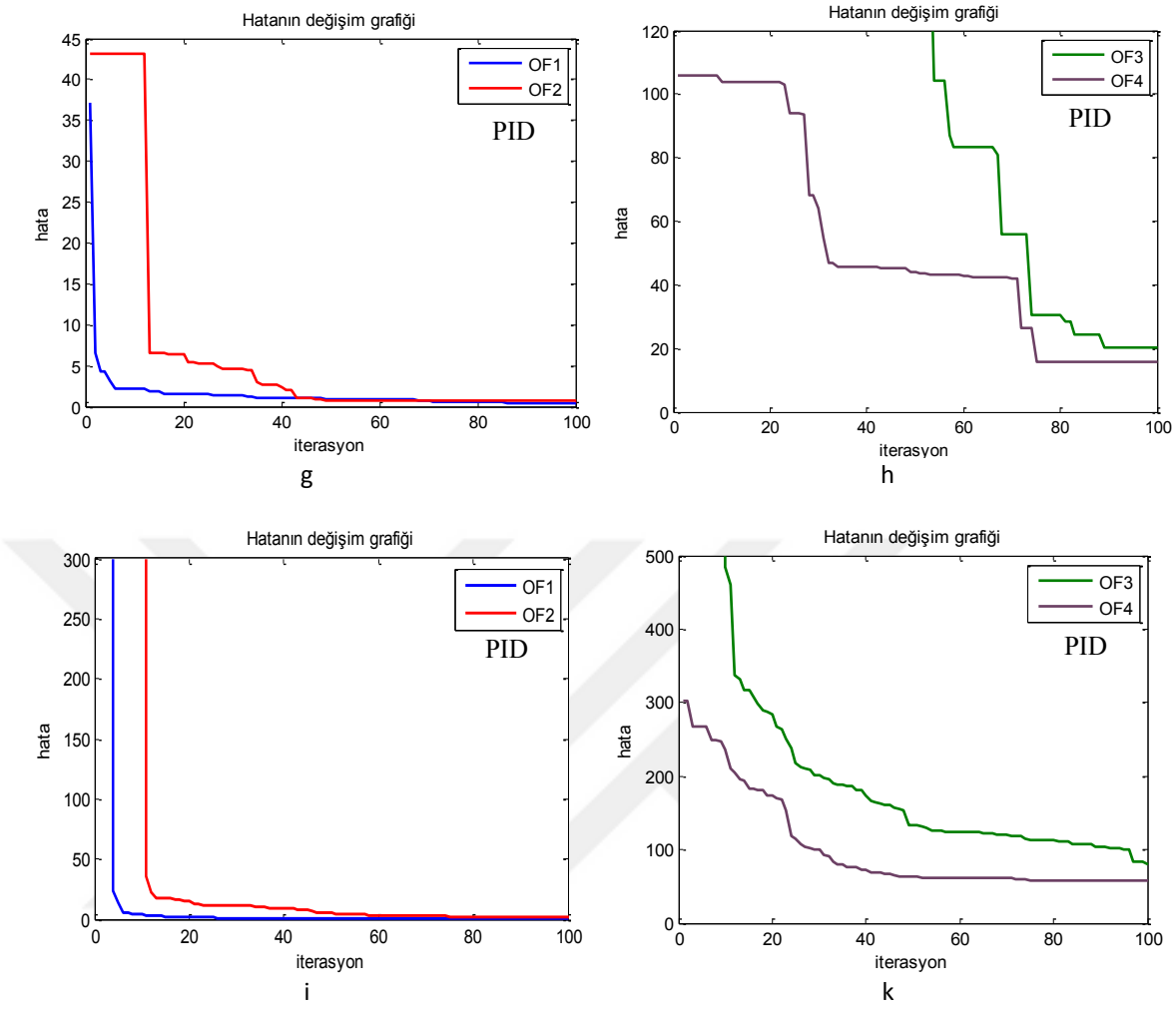
d


Hatanın değişim grafiği
OF1
OF2
FOPID
hata
iterasyon

e


Hatanın değişim grafiği
OF3
OF4
FOPID
hata
iterasyon

f



Şekil 4.9. T_4 süreci için PSO algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

FOPID denetleyiciye ait c,d,e,f hata gelişim grafiklerinde OF_4 dışında üç amaç fonksiyonunda hata gelişim oranının oldukça hızlı olduğu görülmektedir. OF_4'de ise hata oranının 70-80 iterasyonlar arasında minimum değerine ulaştığı (d) grafiğinden anlaşılmaktadır. FOPID denetleyici için ortalama değerlerin yer aldığı (e) ve (f) grafiklerine göre dört amaç fonksiyonunda da hata değeri 40-50. iterasyonlar arasında oldukça azaltılmıştır.

PID denetleyici tasarımında PSO algoritmasına ait hata değişimleri ise g,h,i,k grafiklerinde verilmiştir. OF_1 ve OF_2 için hatanın değeri ilk 20-40. iterasyonlar arasında oldukça azaltılabilirken, OF_3 ve OF_4 için ise hatanın düşürülmesi uzun iterasyon sayılarında mümkün olabilmiştir. Özellikle OF_3 için hata değerinin oldukça yüksek bir değerden başladığı (h) grafiğinden açıkça görülmektedir. Ortalama hata değerlerinin yer

aldığı (i) ve (k) grafiklerinde ise çok yüksek değerlerden optimizasyona başlandığı görülmektedir. OF_1 ve OF_2 fonksiyonlarında ortalama hata değerinin, en düşük hata değerine göre oldukça yüksek bir değerden başlamasının nedeninin, algoritmanın rastgele olarak bir kaç kere yeniden çalıştırılması sırasında hata değerinin oldukça yüksek değerlerden başlaması ve hızlı bir şekilde ilk birkaç iterasyonda hatayı azaltması olarak düşünülmektedir.

Tablo 4.9. T_4 sürecine ait optimum denetleyiciler için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması

Kontr.	Algoritma	ISE	OF_3	Tr(sn)	Ts(sn)	OS%	SS
FOPID	**YAK**	0.2904	**6.3913**	0.4804	0.6975	0	0.0028
	DGA	0.2426	10.1494	0.3821	1.5448	0	0.0057
	PSO	0.2506	10.1234	0.3936	1.5235	0	0.0074
PID	**YAK**	0.5890	**13.2966**	0.5847	1.4078	0.1225	0.9401
	DGA	0.5556	31.2477	0.8322	2.9258	3.6876	0.7114
	PSO	0.5238	18.1828	0.7959	1.1939	2.3252	1.298

Denetleyicilerin uygulanması sonucunda elde edilen grafiklerin karmaşık bir görüntü oluşturmaması adına her algoritma ve denetleyici için ayrı ayrı verilmişti. Algoritmaların performanslarını değerlendirmek amacıyla Tablo 4.9'da T_4 süreci için OF_3 amaç fonksiyonuna ait çıkış değerleri karşılaştırmalı olarak yeniden sunulmuştur. Bu amaç fonksiyonu aynı anda hatayı, oturma süresini ve aşma miktarını optimize etmesi nedeniyle kıyaslamada tercih edilmiştir. Tabloya göre hem FOPID hem de PID denetleyiciye ait çalışmada YAK algoritmasının daha başarılı performans sergilediği görülmektedir. FOPID denetleyici için YAK algoritması 6.3913 OF değeri elde ederken, DGA ile 10.1494 ve PSO ile 10.1234 OF değeri elde edilmiştir. YAK algoritması kullanılarak daha yüksek ISE değeri elde edilmesine rağmen, 0.6975 sn olarak bulunan oturma süresi değeri ile diğer iki algoritmaya üstünlük sağlamıştır. Diğer yandan, PID denetleyicide algoritmalar arasındaki performans farklılığı daha net görülmektedir. Çalışmalar sonucunda, YAK, DG ve PSO algoritmaları ile sırasıyla 13.2966, 31.2477 ve 18.1828 OF değerleri elde edilmiş ve YAK yaklaşımının etkinliği daha belirgin olarak görülmüştür.

T_4 süreci için farklı algoritmalarla yapılan çalışmaların sonuçlarına benzer olarak T_5 süreci için de ilgili çalışmalar tekrar edilmiş ve sonuçları aşağıda sunulmuştur. Böylece, T_5 süreci için YAK algoritması ile elde edilen parametre değerleri Tablo 4.10'da, temel performans karakteristikleri ise Tablo 4.11'de verilmiştir.. Tabloya göre OF_1 ve OF_2 amaç fonksiyonu göz önünde bulundurulduğunda 0.006 ve 0.0001 hata değerleri ile

FOPID denetleyici tasarımı gerçekleştirilmiştir. PID denetleyicide aynı fonksiyonlar için bu değerler sırasıyla 0.0610 ve 0.0113 olarak elde edilmiştir. OF_3 ve OF_4 amaç fonksiyonları için durum benzer olsa da aradaki başarım farkı çok büyük değildir. YAK algoritması ile FOPID denetleyici için 0.5221 OF_3 ve 1.7630 OF_4 değeri elde edilirken, PID denetleyici de 2.4224 OF_3 ve 1.9834 OF_4 değeri elde edilmiştir.

Tabloda dikkat çekici başka bir detay ise oturma sürelerinin oldukça düşük olmasıdır. İki denetleyici için elde edilen en yüksek oturma süresi 0.3343 sn ile OF_2 hata indisine ait iken, en düşük oturma süresi 0.0499 sn ile OF_3 hata indisine aittir. Aşma miktarlarına göre en dezavantajlı hata indisinin OF_2 olduğu söylenebilir. İki denetleyici içinde en yüksek aşma miktarı bu hata indisi kullanarak elde edilmiştir. Hem FOPID hem de PID denetleyici yapısının T_5 süreci için kontrolör tasarımında oldukça hızlı cevap verdiği yine Tablo 4.11'den çıkarılabilecek başka bir sonuçtur.

Tablo 4.10. T_5 süreci için YAK algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

Kontrolör	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	90.177	96.325	95.541	1.4955	0.512
		OF_2	100.0	51.98	99.99	1.019	1.192
		OF_3	29.6254	9.0857	16.8473	1.4958	1.4230
		OF_4	4.187	3.303	5.828	1.065	1.112
PID	Tez Çalışması	OF_1	2.1474	7.4913	10	-	-
		OF_2	5.4539	6.0922	9.9405	-	-
		OF_3	1.2458	3.4414	5.0839	-	-
		OF_4	1.8729	5.0542	5.3945	-	-

Tablo 4.11. T_5 sürecine ait YAK algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

Kontr.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.006		11.7377	0.0162	0.0193	3.8599	0.0002
		OF_2		0.0001	50.088	0.0235	0.0748	16.571	0.0031
		OF_3	0.0140		0.5221	0.0399	0.0499	0.0443	0.0064
		OF_4		0.0689	1.7630	0.1591	0.2147	0	0.0088
PID	Tez Çalışması	OF_1	0.0610		29.3201	0.1176	0.3010	9.0683	0.0002
		OF_2		0.0113	39.7714	0.1132	0.3343	12.6622	0.0001
		OF_3	0.0945	-	2.4224	0.2150	0.2954	0	0.512
		OF_4		0.0547	1.9834	0.2089	0.2874	0	0.7715

Şekil 4.10'da T_5 süreci için FOPID ve PID denetleyici tasarımında YAK algoritması kullanılarak elde edilen basamak cevapları ve hata değişim grafikleri yer almaktadır.

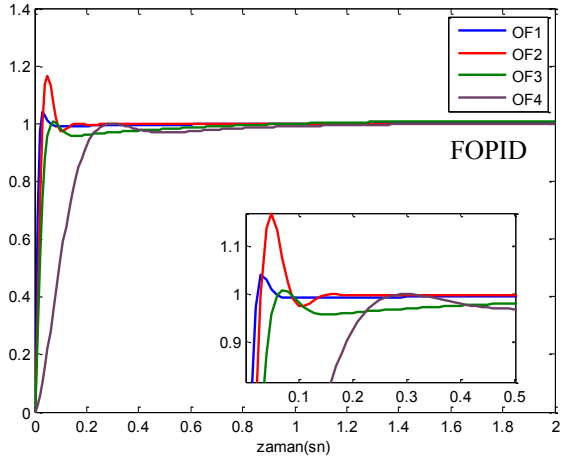
Şekildeki c,d ve g,h grafikleri minimum hata için elde edilen değişimi gösterirken, e,f ve i,k grafikleri ise ortalama hataya ait değişimi sunmaktadır. Grafiklerin oldukça benzer olması nedeniyle [0-0.5] sn aralığındaki değişim büyütülerek şekil içerisinde ayrıca verilmiştir.

FOPID denetleyiciye ait basamak cevaplarının yer aldığı (a) grafiğine göre OF_2 hata indisinin yüksek aşma miktarı ile, OF_4 hata indisinin yükselme süresinin büyük olması dikkat çekicidir. PID denetleyiciye ait (b) grafiğinde ise OF_1 ve OF_2 hata indislerinin aşma miktarlarının, OF_3 ve OF_4 hata indislerinin ise yükselme sürelerinin yüksek olduğu görülmektedir. Ancak basamak cevaplarının [0-2] saniye aralığında elde edildiği göz önünde bulundurulduğunda YAK algoritmasının sürecin kontrolünde oldukça hızlı cevap verdiği ifade edilebilir.

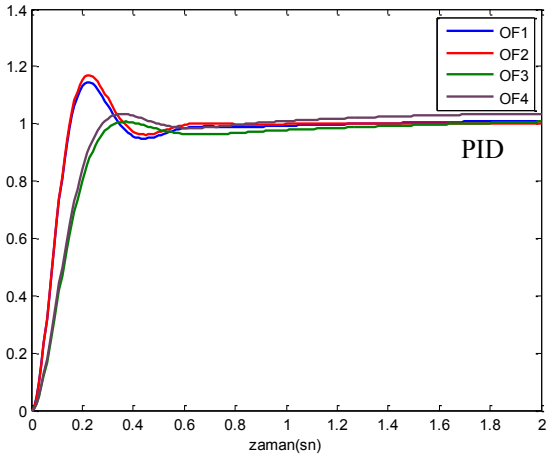
İki denetleyici yapısı içinde sadece ISE ve ITAE hata değerlerinin optimize edildiği çalışmalarda, hata değerinin oldukça düşük değerden başladığı ilk 10-20 iterasyon arasında minimum değere ulaşıldığı hata gelişim grafiklerinden görülmektedir. OF_3 ve OF_4 hata indislerinin kullanıldığı çalışmalarda ise hata değeri daha yüksek değerlerden başlamış ancak 10-30. iterasyonlar arasında hata değeri minimize edilmiştir.

FOPID denetleyici tasarımı sonucunda elde edilen (e) grafiğinde ISE ve ITAE ortalama hata değerlerinin ilk 20 iterasyonda hızlı bir şekilde indirgendiği görülmektedir. Bu grafiğe göre T_5 süreci için yapılan algoritmanın 30 defa yeniden çalıştırılmasında sonuçların birbirinden çok farklı çıkmadığı görülmektedir. Bu durum Tablo 4.11'de yer alan düşük standart sapma değerini doğrulamaktadır. OF_3 ve OF_4 performans indisleri için elde edilen ortalama grafiği (f)'ye göre, hata değeri 30-40. iterasyonlar arasında oldukça azaltılmıştır.

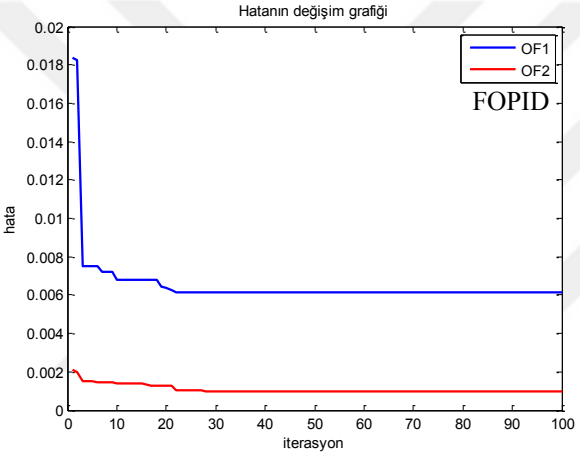
PID denetleyici ile elde edilen ortalama hata grafiklerinde benzer durum söz konusudur. (i) ve (k) grafiklerinde hata değerinin ilk 10 iterasyonda minimize edildiği, bu itrerasyondan sonra çok küçük değişimlerin olduğun görülmektedir. YAK algoritması kullanılarak yapılan denetleyici çalışmalarında hem FOPID hem de PID denetleyici yapısının başarılı sonuçlar sunduğu ifade edilebilir.


OF1
OF2
OF3
OF4
FOPID
zaman(sn)

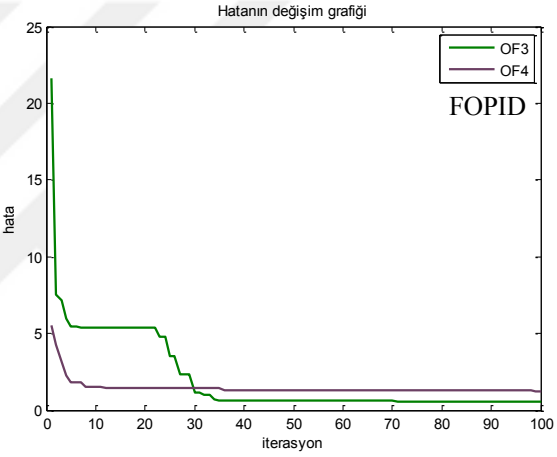
a


OF1
OF2
OF3
OF4
PID
zaman(sn)

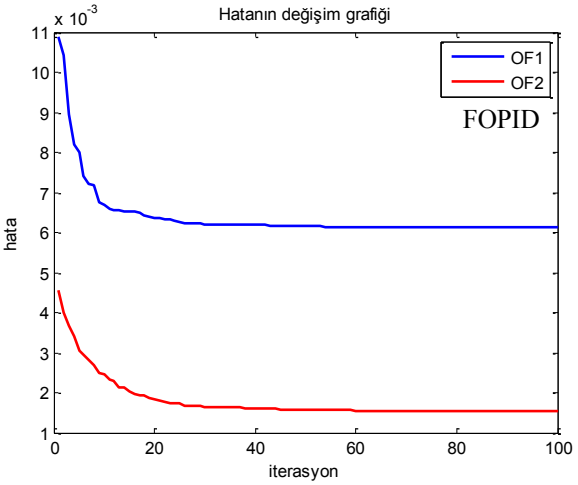
b


Hatanın değişim grafiği
OF1
OF2
FOPID
hata
iterasyon

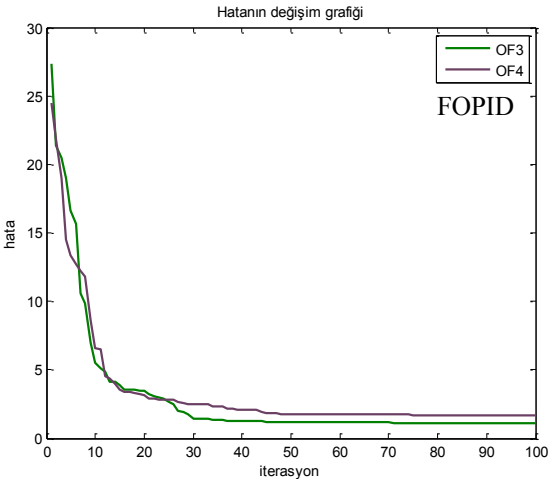
c


Hatanın değişim grafiği
OF3
OF4
FOPID
hata
iterasyon

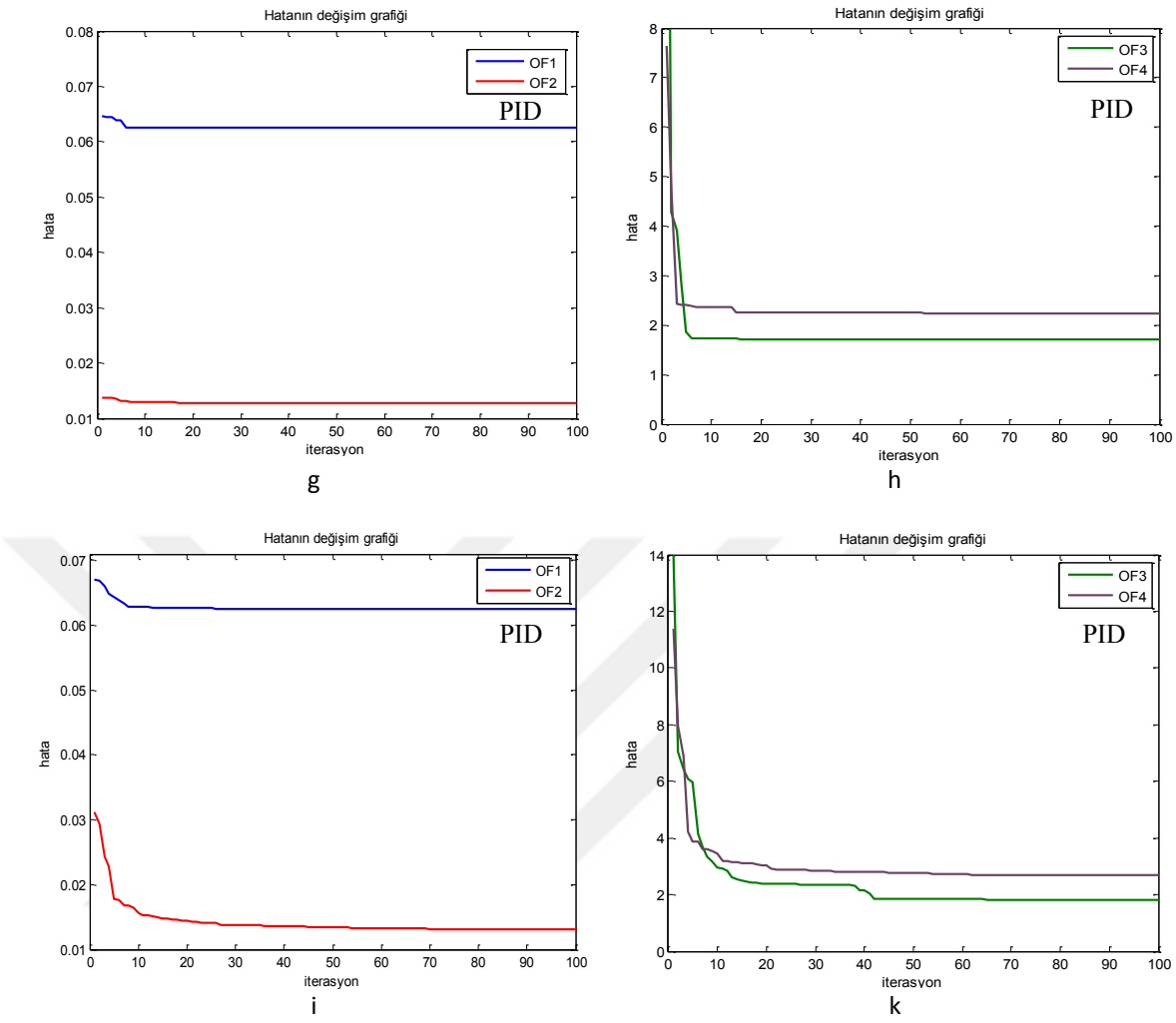
d


x 10^-3
Hatanın değişim grafiği
OF1
OF2
FOPID
hata
iterasyon

e


Hatanın değişim grafiği
OF3
OF4
FOPID
hata
iterasyon

f



Şekil 4.10. T_5 süreci için YAK algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

T_5 süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri Tablo 4.12'de yer almaktadır. Tablo 4.13' te ise aynı süreç için elde edilen hata değerleri, aşma miktarları ve oturma sürelerine ait veriler bulunmaktadır. Tabloya göre amaç fonksiyonları tek tek ele alındığında FOPID denetleyici yapısının başarısı göze çarpmaktadır. Bütün amaç fonksiyonları için FOPID denetleyici yapısı PID denetleyici yapısına göre daha üstün başarı sergilemiştir. Ayrıca bütün amaç fonksiyonları dikkate alındığında ise OF_3 ve OF_4 fonksiyonlarının başarımı dikkat çekicidir.

Tablo 4.12. T_5 süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

Kontrolör	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	95.784	25.002	94.747	1.328	1.021
		OF_2	85.577	35.218	95.246	1.201	1.101
		OF_3	19.8287	10.8793	17.1418	1.4961	1.3454
		OF_4	38.677	31.162	20.9594	1.4833	1.4221
PID	Tez Çalışması	OF_1	3.214	6.687	9.254	-	-
		OF_2	4.6700	5.1778	8.4098	-	-
		OF_3	1.8197	4.8384	5.3698	-	-
		OF_4	2.0440	2.7545	4.1634	-	-

Tablo 4.13. T_5 sürecine ait DG algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

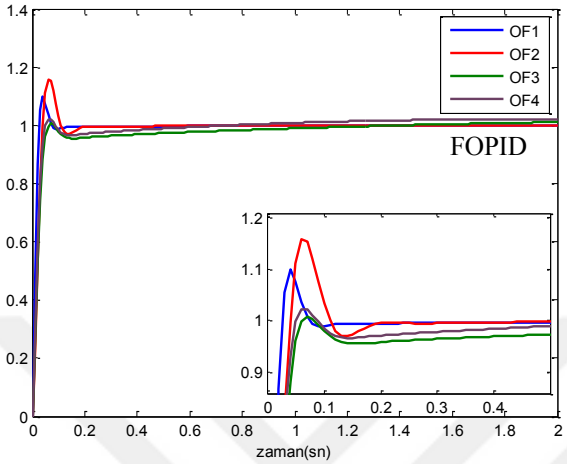
Kontr.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.0081		30.1218	0.0197	0.0567	9.9192	0.0021
		OF_2	-	0.0026	48.9831	0.0237	0.0751	16.1940	0.0033
		OF_3	0.0141		1.3117	0.0398	0.2342	0	0.0851
		OF_4		0.0307	1.9116	0.0356	0.1761	0.2412	0.0054
PID	Tez Çalışması	OF_1	0.0637	-	29.7409	0.1230	0.3206	9.1668	0.0474
		OF_2		0.0123	32.6032	0.1285	0.3589	10.2286	0.0023
		OF_3	0.0905	-	2.3427	0.2091	0.2875	0	0.0697
		OF_4		0.0262	1.9837	0.2509	0.3443	0.0001	0.1768

Tablo 4.13'e göre FOPID denetleyicide OF_1 hata indisi 0.0081, PID denetleyici de ise 0.0637 olarak elde edilmiştir. OF_2 hata indisinde ise FOPID için 0.0026, PID için 0.0123 değerleri elde edilmiştir. Sadece ISE ve ITAE hata indislerinin yer aldığı denetleyici çalışmalarında yüksek aşma miktarı ve düşük oturma süresi elde edilmiştir. T_5 süreci için DG algoritmasına dayalı denetleyici tasarımına ait sistem cevapları ve hata değişimleri Şekil 4.11'de sunulmuştur

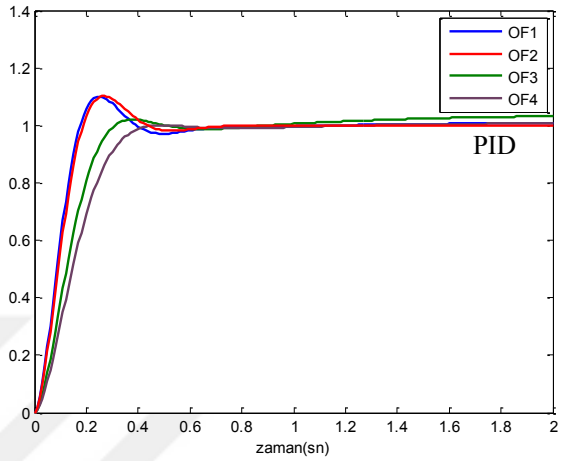
Şekil 4.11'de T_5 süreci için FOPID ve PID denetleyici tasarımında DG algoritması kullanılarak elde edilen basamak cevapları ve hata değişim grafikleri yer almaktadır. Şekildeki c,d ve g,h grafikleri minimum hata için elde edilen değişimi gösterirken, e,f ve i,k grafikleri ise ortalama hataya ait değişimi sunmaktadır. Şekil 4.11'de FOPID denetleyiciye ait basamak cevaplarının yer aldığı (a) grafiğinde ISE ve ITAE hatalarının performans indisi olarak kullanıldığı çalışmalarda oluşan aşma miktarının yüksekliği göze çarpmaktadır. PID denetleyici yapısına ait (b) grafiğinde bu miktar düşük olsa da benzerlik göstermektedir.

OF_1 ve OF_2 hata değerlerine ait grafiklerde her iki iki denetleyici yapısında da optimizasyon küçük değerlerden başladığı (c) ve (f) grafiklerinden görülmektedir. OF_3

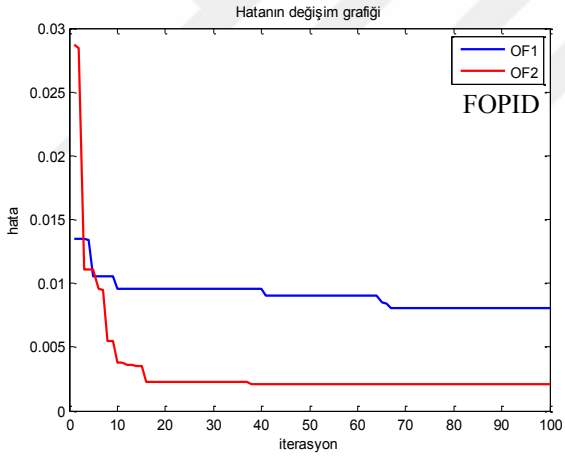
ve OF_4 hata indislerinin kullanıldığı çalışmalarda ise DGA'nın daha yüksek hata değerlerinden başlayarak ilk 30 iterasyonda hatayı minimize ettiği söylenebilir. Ortalama hata değerlerinin yer aldığı grafiklerde ise en düşük hata değerinde elde edilen eğrilere benzer eğriler elde edilmiştir.



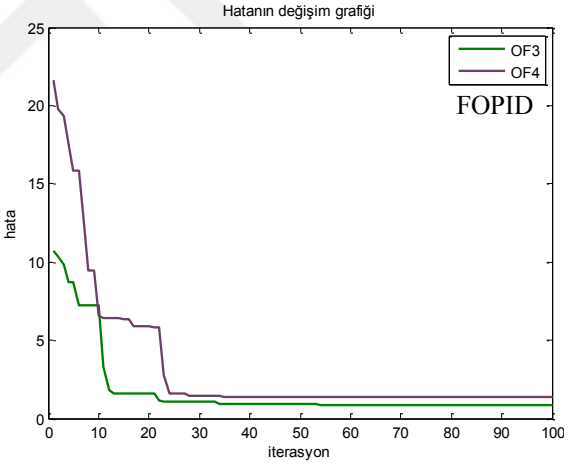
a



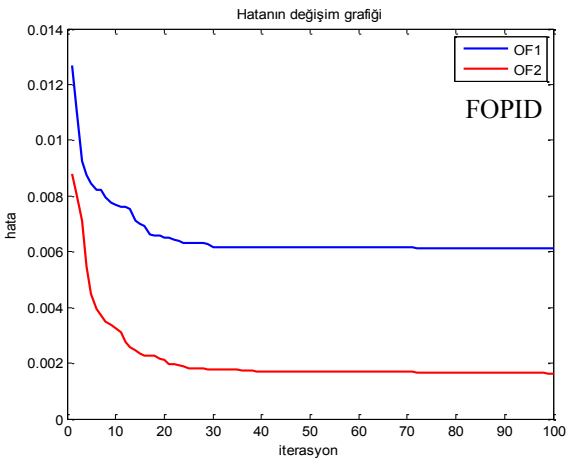
b



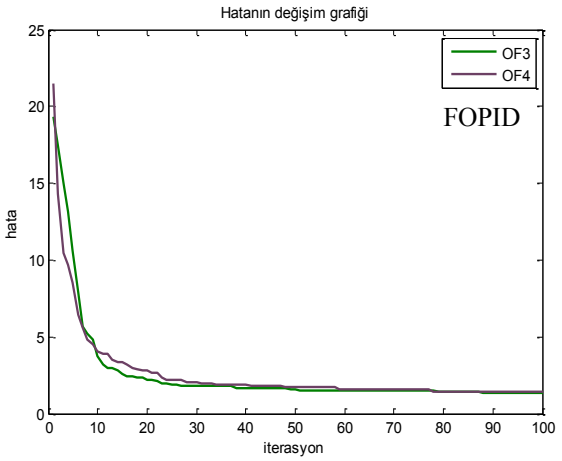
c



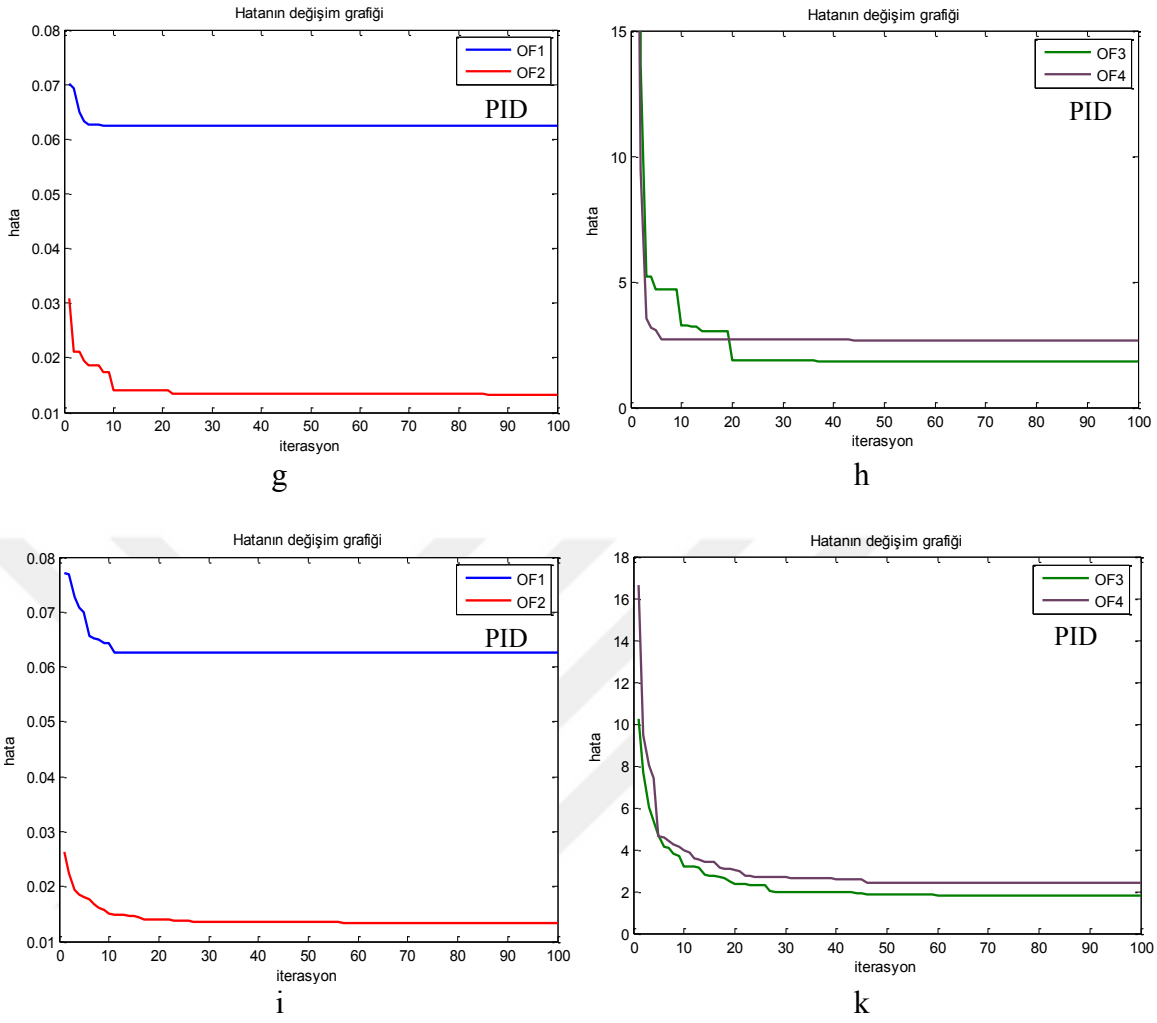
d



e



f



Şekil 4.11. T_5 süreci için DG algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

Tablo 4.14. T_5 süreci için PSO algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

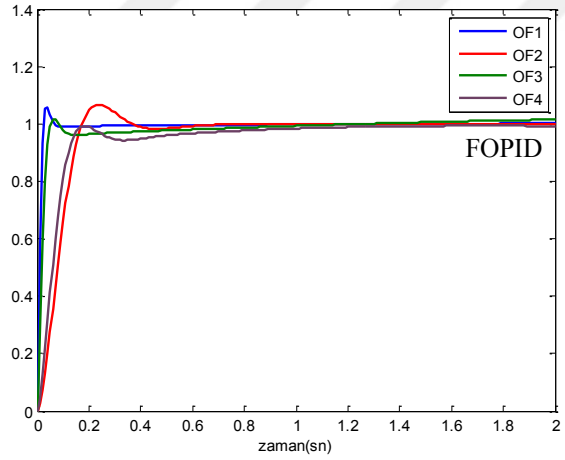
Kontrolör	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	72.414	65.963	94.747	1.414	1.214
		OF_2	7.598	4.170	8.038	1.213	1.093
		OF_3	22.276	17.2563	20.3174	1.4955	1.247
		OF_4	3.647	5.802	8.081	0.511	1.235
PID	Tez Çalışması	OF_1	3.114	6.254	8.241	-	-
		OF_2	5.5271	6.1075	9.9905	-	-
		OF_3	1.5491	3.9604	5.1874	-	-
		OF_4	2.056	6.3456	5.6564	-	-

T5 süreci için PSO algoritmasının denetleyici tasarımında kullanıldığı çalışmalarda elde edilen parametre değerleri Tablo 4.14'te yer almaktadır. Aynı süreç için elde edilen temel performans karakteristikleri ise Tablo 4.15'te yer almaktadır. Diğer iki algoritmada olduğu gibi bu süreç için önerilen denetleyici yapılarında FOPID

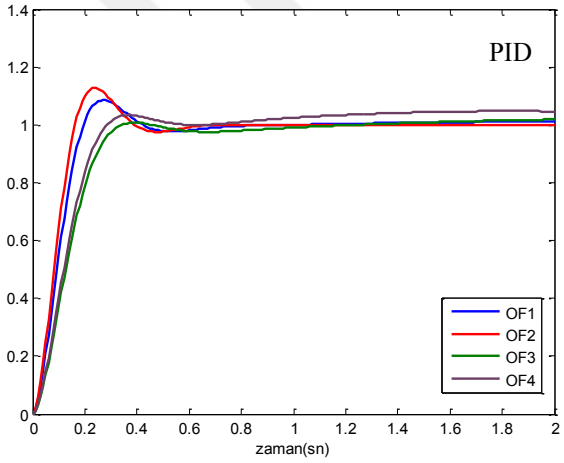
denetleyici yapısının performansının PID denetleyiciye göre daha başarılı olduğu söylenebilir. Şekil 4.12'de T_5 süreci için FOPID ve PID denetleyici tasarımında PSO algoritması kullanılarak elde edilen basamak cevapları ve hata değişim grafikleri sunulmuştur. Diğerlerinde olduğu gibi, bu şekildeki c,d ve g,h grafikleri minimum hata için elde edilen değişimi, e,f ve i,k grafikleri ise ortalama hataya ait değişimi göstermektedir.

Tablo 4.15. T_5 sürecine ait PSO algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

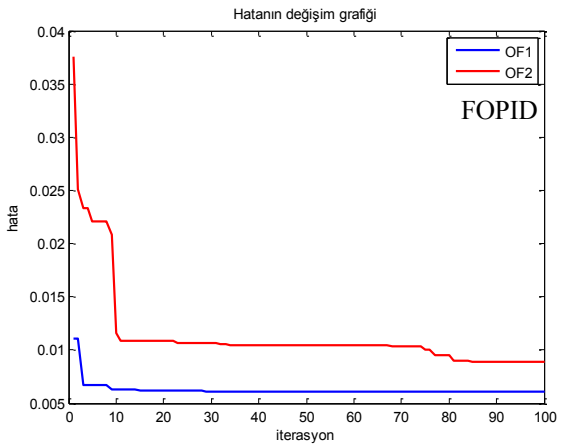
Kontr.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.0069	-	17.4114	0.0174	0.0426	5.7098	0.0025
		OF_2		0.0089	21.8908	0.1184	0.2848	6.7926	0.0088
		OF_3	0.0126		1.1516	0.0355	0.2001	0.0083	0.0564
		OF_4		0.0435	1.3700	0.1018	0.1347	0.0873	0.0454
PID	Tez Çalışması	OF_1	0.0683	-	24.3789	0.1352	0.3284	7.3514	0.0002
		OF_2		0.0113	40.1168	0.1127	0.3340	12.778	0.0021
		OF_3	0.0926		2.3900	0.2131	0.2927	0	0.851
		OF_4		0.0796	2.4474	0.2029	0.2788	0.0859	0.991

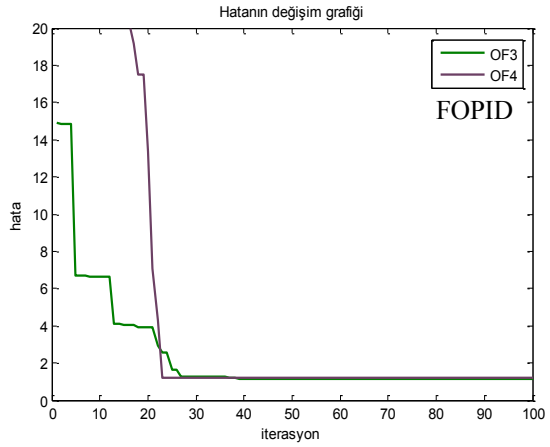


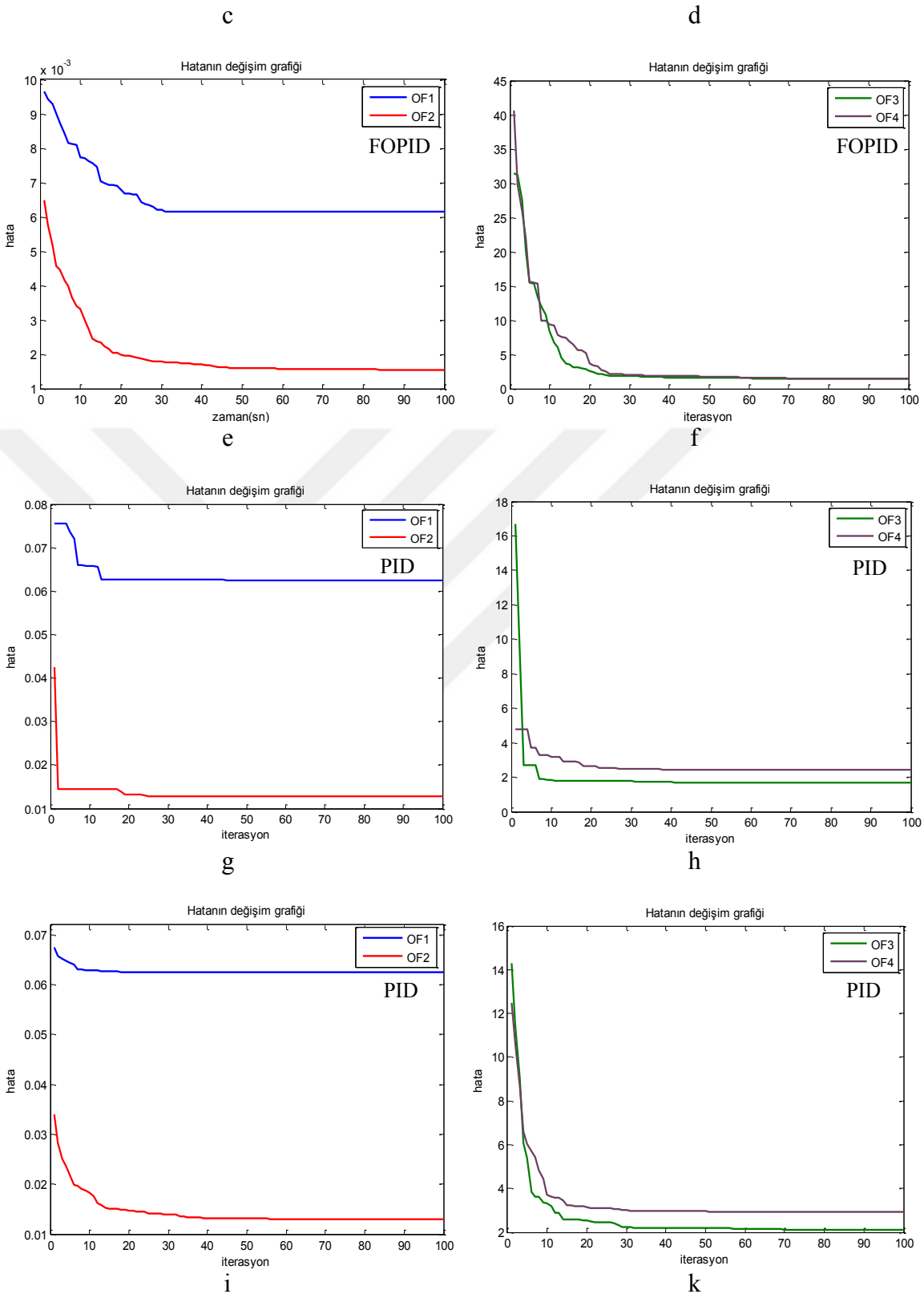
a



b







Şekil 4.12. T_5 süreci için PSO algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

OF_1 ve OF_2 fonksiyonlarında aşma miktarlarının yüksekliği, OF_3 ve OF_4 fonksiyonlarında ise genel olarak oturma sürelerinin düşüklüğü Şekil 4.12'de görülmektedir. Bu algoritma ile yapılan denetleyici tasarımında aşma miktarlarının diğer iki algoritmaya göre düşük olduğu Şekil 4.12.(a) ve (b) grafiklerinden anlaşılmaktadır. PSO algoritmasının diğer iki algoritmada olduğu gibi hatayı hızlı bir şekilde indirgediği yine aynı şekilden çıkarılabilecek başka bir sonuçtur. Özellikle OF_3 ve OF_4 amaç fonksiyonlarının kullanıldığı çalışmada FOPID denetleyici tasarımında PSO diğer iki algoritmaya göre hata değerini daha yüksek değerden başlayarak minimize etmeye çalışmış ve ilk 20 iterasyonda hata değerini oldukça azaltmıştır. PID denetleyicinin kullanıldığı çalışmada ise yine aynı fonksiyonlar için hata değeri ilk 10-15 iterasyon arasında oldukça azaltılmıştır.

Tablo 4.16'da ise algoritmaların denetleyici tasarımındaki performans kıyaslaması için OF_3 fonksiyonuna ait temel performans karakteristikleri verilmiştir. Tablodaki değerlerden, FOPID denetleyici için en iyi sonucun YAK algoritması kullanılarak elde edildiği görülmektedir. YAK algoritması ile 0.5221 OF_3 değerine ulaşılırken, 0.0399sn yükselme süresi, 0.0499sn oturma süresi ve %0.0443 aşma miktarı elde edilmiştir. Standart sapma değeri ise YAK için 0.0064 olarak elde edilmiştir. PSO algoritması 1.1516 OF_3 değeri ile ikinci en iyi performansı sunarken, DGA kullanıldığında 1.3117 ile en başarısız sonuç elde edilmiştir. PID denetleyici örneğinde ise FOPID denetleyicidekine benzer sonuçlar mevcuttur. DG algoritması 2.3427 OF_3 değeri ile en başarılı algoritma olmuştur. Bu algoritmayı 2.3900 OF ile PSO ve 2.4224 ile YAK algoritması takip etmektedir. Aşma miktarının %0 olarak elde edildiği DG algoritmasında 0.2875sn oturma süresi ve 0.2091sn yükselme süresi elde edilmiştir. Algoritmaların 30 defa yeniden çalıştırılması sonunda 0.0697 standart sapma değeri ile DGA diğer iki algoritmaya göre daha düşük değer elde etmiştir.

Tablo 4.16. T_5 sürecine ait optimum denetleyiciler için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması

Kontr.	Algoritma	ISE	OF_3	Tr(sn)	Ts(sn)	OS%	SS
FOPID	**YAK**	0.0140	**0.5221**	0.0399	0.0499	0.0443	0.0064
	DGA	0.0141	1.3117	0.0398	0.2342	0	0.085
	PSO	0.0126	1.1516	0.0355	0.2001	0.0083	0.0564
PID	YAK	0.0945	2.4224	0.2150	0.2954	0	0.512
	DGA	0.0905	**2.3427**	0.2091	0.2875	0	0.0697
	PSO	0.0926	2.3900	0.2131	0.2927	0	0.851

4.1.3 Tamsayı Dereceli Sistemlerin Kontrolü için Tamsayı ve Kesirli Dereceli PID Kontrolör Tasarımına Yönelik Diğer Çalışmalar

Bu bölümde, literatürdeki [116-118] numaralı referanslardan alınan ve tamsayı dereceli sistemleri içeren örnekler denetleyici tasarımında dikkate alınmıştır. Böylece, incelenen sistem sayısı arttırılarak daha detaylı bir inceleme ve karşılaştırma yapma imkanı ortaya konulmaya çalışılmıştır. Yüksek mertebeli bu süreçler için FOPID ve PID denetleyici tasarımları önerilmiş, elde edilen sonuçlar literatürde yer alan sonuçlarla karşılaştırmalı olarak sunulmuştur. Denetleyici tasarımında dikkate alınan ve T_6 [116, 117], T_7 [116] ve T_8 [118] olarak ifade edilen süreçler Eşitlik 4.10-4.12'de verilmiştir.

$$T_6(\mathrm{s}) = \frac{1}{(1+s)(1+0.1\mathrm{s})^2} \tag{4.10}$$

$$T_7(\mathrm{s}) = \frac{1}{(1+s)(1+0.6\mathrm{s})(1+0.6^2\,\mathrm{s})(1+0.6^3\,\mathrm{s})} \tag{4.11}$$

$$T_8(\mathrm{s}) = \frac{9}{(\mathrm{s}+1)(\mathrm{s}^2+2\mathrm{s}+9)} \tag{4.12}$$

YAK algoritması kullanarak yapılan çalışmalarda elde edilen T_6 sürecine ait FOPID ve PID denetleyicilerin parametreleri Tablo 4.17'de verilmiştir. Ayrıca literatürde Liu ve arkadaşları [116] ile Das ve arkadaşlarının [117] yapmış olduğu çalışmalardaki denetleyici parametre değerlerine de bu tabloda yer verilmiştir. Tablo 4.18'de ise YAK algoritması ile elde edilen en iyi sonuçlara ait temel performans karakteristikleri yer almaktadır.

Tablo 4.17. T_6 süreci için YAK algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

Kontr.	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez çalışması	OF_1	57.025	86.999	89.378	0.266	1.338
		OF_2	56.998	87.445	17.025	0.386	1.254
		OF_3	9.916	87.218	50.640	0.388	1.481
		OF_4	6.279	60.726	36.352	0.473	1.357
	Das ve ark. [116]		1.051	1.169	0.064	0.997	0.109
	Liu ve ark.. [117]		19.977	16.654	2.979	1.002	1.064
PID	Tez çalışması	OF_1	6.977	6.416	3.950	--	--
		OF_2	7.805	6.675	1.132	--	--
		OF_3	9.268	7.375	1.666	--	--
		OF_4	9.798	7.637	2.072	--	--
	Das ve ark. [116]		0.755	0.793	0.043	--	--
	Liu ve ark. [117]		9.909	8.441	1.474	--	--

Tablo 4.18. T_6 sürecine ait YAK algoritmasına dayalı optimum denetleyiciler için kullanılarak elde edilen temel performans karakteristikleri

Kontr.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez çalışması	OF_1	0.0104	--	6.9972	0.0177	0.0276	2.2517	0.0006
		OF_2	--	0.0174	11.1399	0.0353	0.0451	3.5801	0.0031
		OF_3	0.0108	--	0.2596	0.0185	0.0304	0.0000	0.0022
		OF_4	--	0.0466	0.6436	0.0277	0.0351	0.000643	0.0087
	Das ve ark. [116]		0.5465	0.5545	15.5901 (OF_3) 15.6710 (OF_4)	1.4230	2.2051	0.0000	-
	Liu ve ark. [117]		0.0482	0.0071	26.0158 (OF_3) 25.6043 (OF_4)	0.0967	0.2395	8.1122	-
PID	Tez çalışması	OF_1	0.0599	--	7.4725	0.1153	1.2814	0.000172	0.0014
		OF_2	--	0.0179	1.6279	0.2087	0.2898	0.000049	0.0009
		OF_3	0.0799	--	1.9175	0.1605	0.2237	0.0000	0.0011
		OF_4	--	0.0273	3.0480	0.1399	0.5550	0.0000	0.0096
	Das ve ark. [116]		0.7346	1.1023	19.9645 (OF_3) 23.6414 (OF_4)	1.9418	2.5238	0.0000	-
	Liu ve ark. [117]		0.0835	0.0139	8.7678 (OF_3) 8.0717 (OF_4)	0.1765	0.2382	2.2474	-

Tabloya göre tez çalışmasında tasarlanan denetleyici yapılarının literatürde yer alan örneklere göre daha başarılı olduğunu söylemek mümkündür. Örneğin Das ve arkadaşlarının yapmış olduğu FOPID denetleyici tasarımında 0.5465 ISE, 0.5545 ITAE hata değeri elde edilmiştir. Tez çalışmasında önerilen FOPID denetleyici de ise 0.0104 ISE ve 0.0174 ITAE değerleri elde edilmiştir. Liu ve arkadaşlarına ait FOPID denetleyici tasarımında bu hata değerleri sırasıyla 0.0482 ve 0.0071 olarak elde edilmiştir. Bu değerlerle birlikte aşma miktarı ve oturma sürelerini de hesaplayarak elde edilen OF_3 değeri [116] numaraları referansta 15.5901, OF_4 değeri 15.6710 olarak bulunmuştur. Aynı amaç fonksiyonları [117] numaralı referansta sırasıyla 26.0158 ve 25.6043 olarak elde edilmiştir. Tez çalışmasında ise OF_3 0.2596, OF_4 ise 0.6436 olarak bulunmuştur. Tablo 4.18'e göre FOPID denetleyici yapısında literatürdeki örneklere göre oldukça başarılı sonuçlar elde edildiği ifade edilebilir.

PID denetleyici için de elde edilen sonuçlar benzer özellikler taşımaktadır. Das ve arkadaşları OF_3 ve OF_4 değerlerini sırasıyla 19.9645, 23.6414, Liu ve arkadaşları 8.7678, 8.0717 olarak bulmuş tez çalışmasında ise bu sonuçlar 1.9175 ve 3.0480 olarak elde edilmiştir. Bu sonuçlara göre iki denetleyici yapısı için de literatürdeki örneklere göre daha başarılı sonuçlar elde edildiği açıkça ifade edilebilir. Özellikle aşma ve oturma sürelerinde oluşan farklılıklar dikkat çekicidir. FOPID denetleyici yapısı ile PID denetleyici yapısı kıyaslandığında ise FOPID denetleyicinin daha başarılı sonuçlar elde

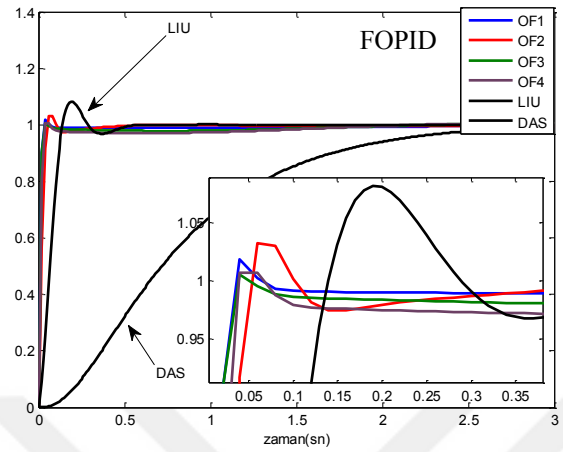
ettiği yine aynı tablodan anlaşılmaktadır. Elde edilen bu sonuçlara ait grafikler Şekil 4.13'te daha net şekilde görünmektedir. Hem denetleyici yapısı arasındaki başarım farkları hem de literatürdeki çalışmalardan daha üstün performans sonuçlarına ulaşıldığı şekilden açıkça anlaşılmaktadır. T_6 süreci için YAK algoritmasına dayalı FOPID ve PID denetleyici tasarımında elde edilen basamak cevapları ve hata değişim grafikleri Şekil 4.13'de verilmiştir. Şekildeki c,d ve g,h grafikleri minimum hata için elde edilen değişimleri, e,f ve i,k grafikleri ise ortalama hataya ait değişimleri göstermektedir.

Şekil 4.13'te literatür sonuçları ile FOPID ve PID denetleyici yapılarına ait birim basamak cevapları birlikte verilmiştir. Şekil 4.13(a) grafiğine göre tez çalışmasında FOPID denetleyici tasarımında bütün amaç fonksiyonlarında elde edilen aşma miktarının Liu ve arkadaşları [117] tarafından yapılan çalışmadaki aşma miktarına göre daha düşük olduğu, aynı zamanda yükselme ve oturma sürelerinin de aynı çalışmaya göre oldukça düşük olduğu söylenebilir. Das ve arkadaşlarının [116] yapmış olduğu çalışmalara göre aşma miktarı tez çalışmasında daha yüksek olsa da oturma ve yükselme süreleri arasındaki ciddi fark göze çarpmaktadır.

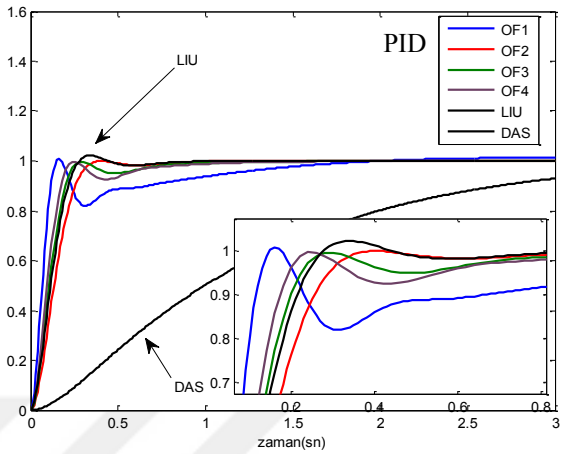
PID denetleyici tasarımında yine bütün amaç fonksiyonlarında tez çalışmasında önerilen denetleyici yapılarının üstünlüğü göze çarpmaktadır. Özellikle Das ve arkadaşlarının [116] yapmış olduğu çalışmada elde edilen basamak cevabının aşırı sönümlü olduğu, oturma ve süresinin oldukça yüksek olduğu Şekil 4.13(b) grafiğinden açıkça görülmektedir. Ayrıca yine aynı grafiğe göre Liu ve arkadaşlarının [117] kullanmış olduğu denetleyici yapısında az da olsa aşma miktarının olduğu ve tez çalışmasındaki denetleyici yapıları ile benzer oturma ve yükselme sürelerine sahip olduğu söylenebilir. Literatürde yer alan çalışmalarda dahil olmak üzere bütün amaç fonksiyonlarında FOPID denetleyici yapısının PID denetleyici yapısına üstünlüğü Şekil 4.13 (a) ve (b) grafiklerinden çıkarılabilecek başka bir sonuçtur.

FOPID denetleyiciye ait hata gelişim grafiklerinin yer aldığı Şekil 4.13.c,d,e,f grafiklerine göre OF_1 ve OF_2 amaç fonksiyonlarının kullanıldığı çalışmalarda YAK algoritması minimuma çok yakın değerden optimizasyona başlamış ve kısa sürede hatayı indirgemiştir. OF_3 ve OF_4 amaç fonksiyonlarında ise hata değeri nispeten daha yüksek değerlerden başlamış ilk 30 iterasyonda nihai değere ulaşılmıştır. Ortalama hata değeri göz önüne alındığında ise hata değeri bütün amaç fonksiyonları için ilk 30 iterasyonda oldukça indirgenmiştir. PID denetleyici için elde edilen Şekil 4.13.g,h,i,k

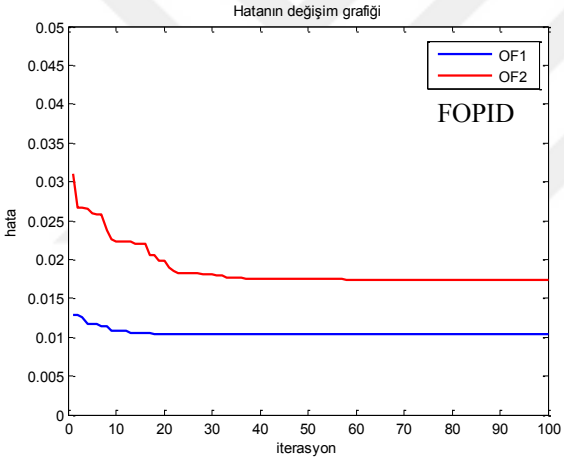
hata gelişim grafiklerinde ise YAK algoritması özellikle OF_3 ve OF_4 amaç fonksiyonlarında yüksek hata değerini ilk 50-60 iterasyonda minimize etmiştir.



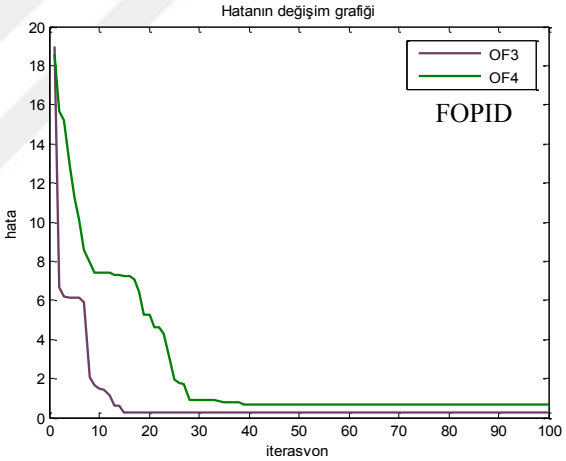
a



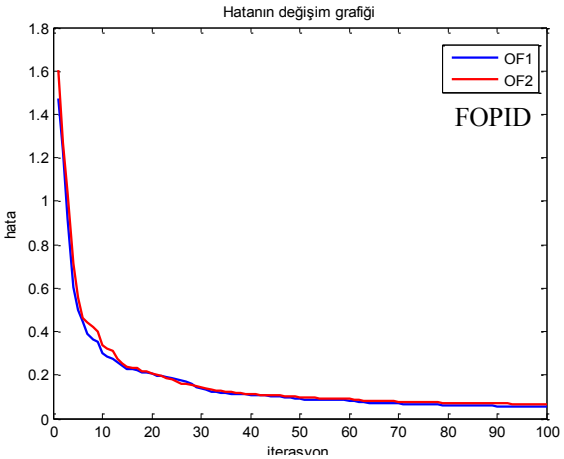
b



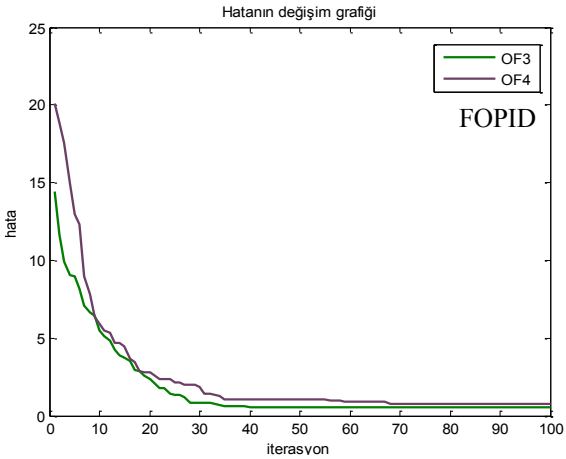
c



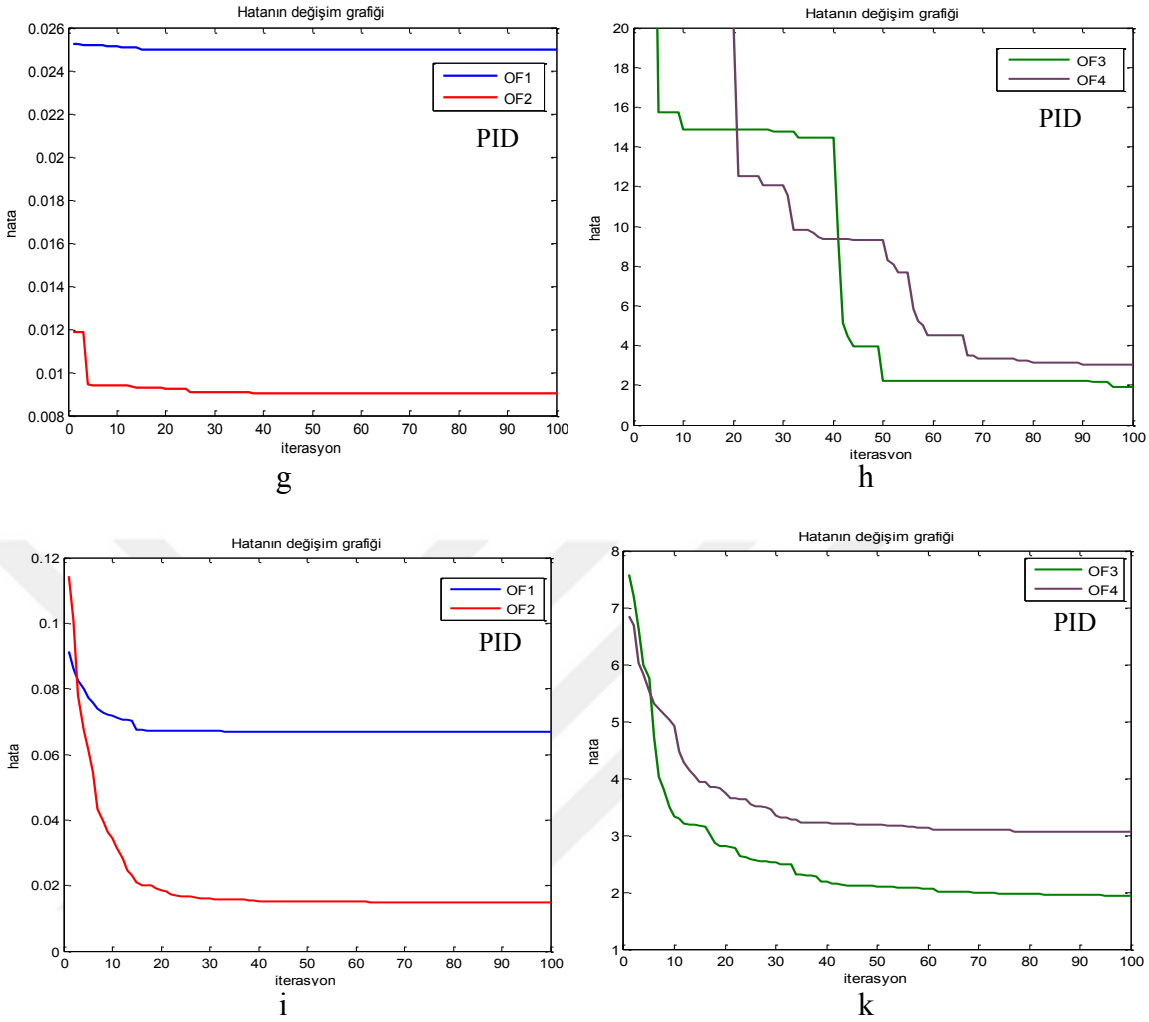
d



e



f



Şekil 4.13. T_6 süreci için YAK algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

DG algoritması kullanarak T_6 süreci için tasarlanan FOPID denetleyici yapısına ait parametre değerleri Tablo 4.19'da sunulmuştur. Bu denetleyici yapısı kullanılarak elde edilen hata, aşma miktarı, yükselme ve oturma süreleri ile standart sapma değerleri ise Tablo 4.20'de verilmiştir. Tabloya göre FOPID denetleyici ile yapılan çalışmalarda OF_2 amaç fonksiyonunun kullanımında elde edilen aşma miktarı dikkat çekicidir. % 22.4042 değeri ile oldukça yüksek aşma miktarının elde edildiği çalışmada 0.0026 ITAE değeri elde edilmiştir. Aynı amaç fonksiyonu için PID denetleyici ile % 1.6024 gibi oldukça düşük aşma miktarı elde edilmiştir. Bu denetleyici yapısında OF_2 için elde edilen ITAE hata değeri ise 0.0153'tür. Standart sapma değerlerinde ise DGA'nın performansının oldukça başarılı olduğunu söylemek mümkündür.

Tablo 4.19. T_6 süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

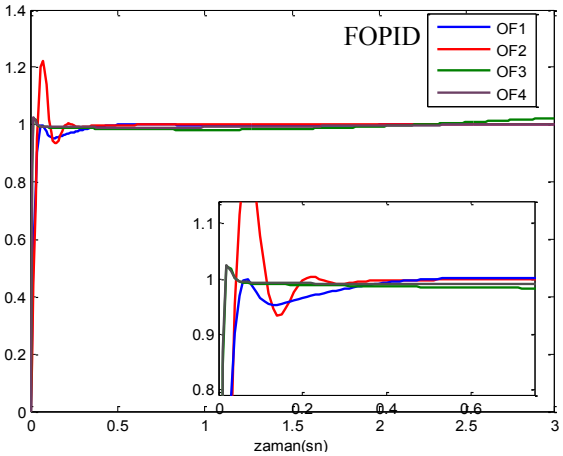
Kontr.	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	69.1541	82.0212	68.7849	1.4500	0.5892
		OF_2	100	19.8986	85.1416	1.0580	0.9721
		OF_3	33.306	99.281	80.411	1.451	0.8377
		OF_4	25.589	38.9980	65.250	1.333	0.678
PID	Tez Çalışması	OF_1	6.256	6.6987	10.143	-	-
		OF_2	8.8008	7.5915	1.2318	-	-
		OF_3	9.2585	7.6214	1.6373	-	-
		OF_4	9.0905	7.2851	1.6340	-	-

Tablo 4.20. T_6 sürecine ait DG algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

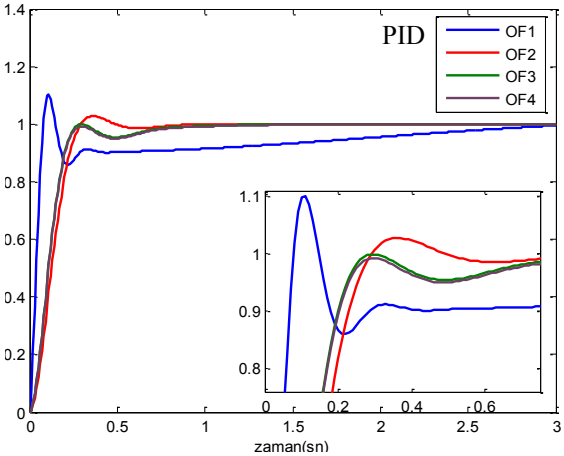
Kontr.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.0068	-	18.2861	0.0146	0.0333	6.0173	0.0005
		OF_2	-	0.0026	68.0537	0.0322	0.1629	22.4042	0.0006
		OF_3	0.0057	-	0.1442	0.0132	0.0172	0	0.0464
		OF_4	-	0.0214	0.333	0.0130	0.0166	0	1.3182
PID	Tez Çalışması	OF_1	0.0409	-	41.0985	0.0545	1.7831	10.5915	0.0003
		OF_2	-	0.0153	6.3196	0.2008	0.2720	1.6024	0.0082
		OF_3	0.0810	-	2.0526	0.1803	0.2486	0	0.0024
		OF_4	-	0.0255	1.5282	0.1830	0.2546	0	0.0099

Şekil 4.14'te ise DG algoritması kullanarak elde edilen denetleyici yapılarına ait çıkış grafikleri yer almaktadır. Şekildeki c,d ve g,h grafikleri minimum hata için elde edilen değişimi gösterirken, e,f ve i,k grafikleri ise ortalama hataya ait değişimi sunmaktadır. FOPID denetleyici yapısının PID'ye göre oldukça kısa yükselme ve oturma süresine sahip olduğu Şekil 4.14.a ve b grafiklerinden açıkça görülmektedir. Ayrıca yukarıda bahsedilen OF_2 amaç fonksiyonuna ait yüksek aşma miktarı da Şekil 4.14.a grafiğinde net bir şekilde görülmektedir.

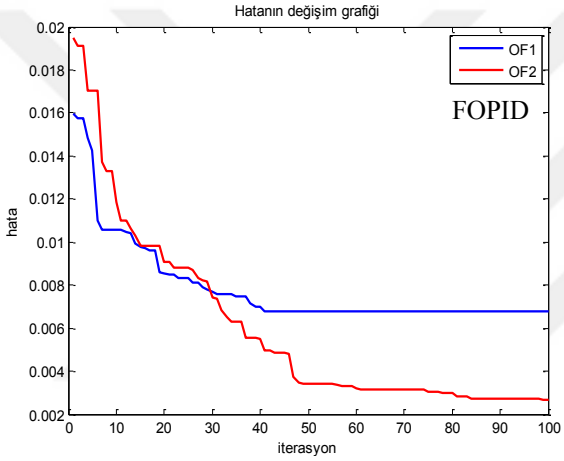
FOPID denetleyicide hatanın iterasyonlara göre değişimini gösteren Şekil 4.14.c ve e grafiklerinde hata değeri 40-50 iterasyonların arasında minimuma oldukça yakınsamış, Şekil 4.14.d ve f grafiklerinde ise ilk 20 iterasyonda minimuma yakınsamıştır. PID denetleyici yapısına ait Şekil 4.14.g,h,i ve k grafiklerinde de benzer sonuçlara rastlanmaktadır. Hata değeri maksimum iterasyon sayısına ulaşmadan indirgenmiştir. Ortalama hatanın değişim grafiklerinde ise hata ilk 20 iterasyonda azaltılmıştır. DG algoritmasının T_6 süreci için yapılan denetleyici tasarımı çalışmasında oldukça hızlı olduğunu bu grafiklere göre söylemek mümkündür.


FOPID
OF1
OF2
OF3
OF4
zaman(sn)

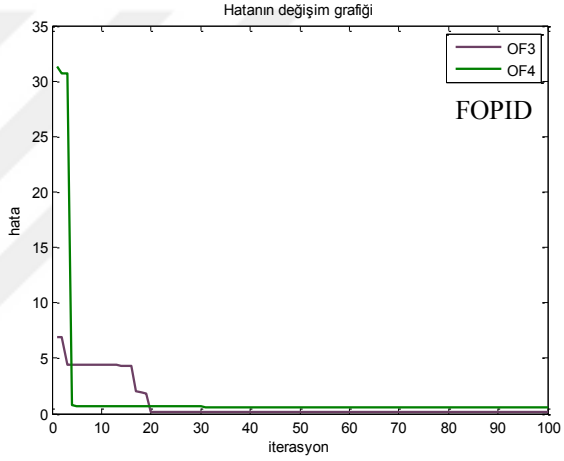
a


PID
OF1
OF2
OF3
OF4
zaman(sn)

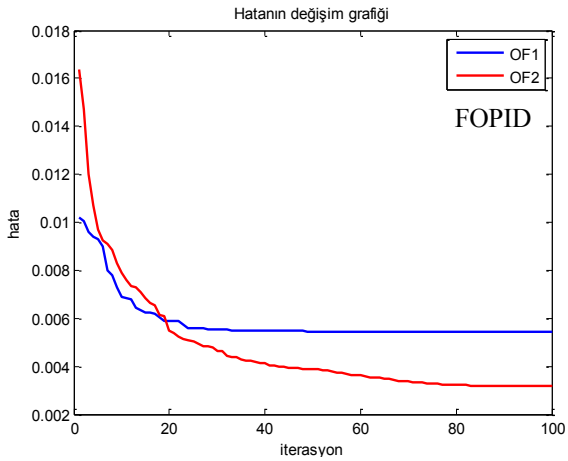
b


Hatanın değişim grafiği
FOPID
OF1
OF2
hata
iterasyon

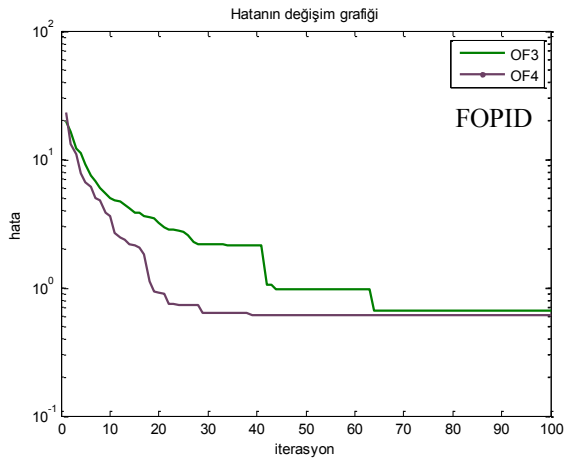
c


Hatanın değişim grafiği
FOPID
OF3
OF4
hata
iterasyon

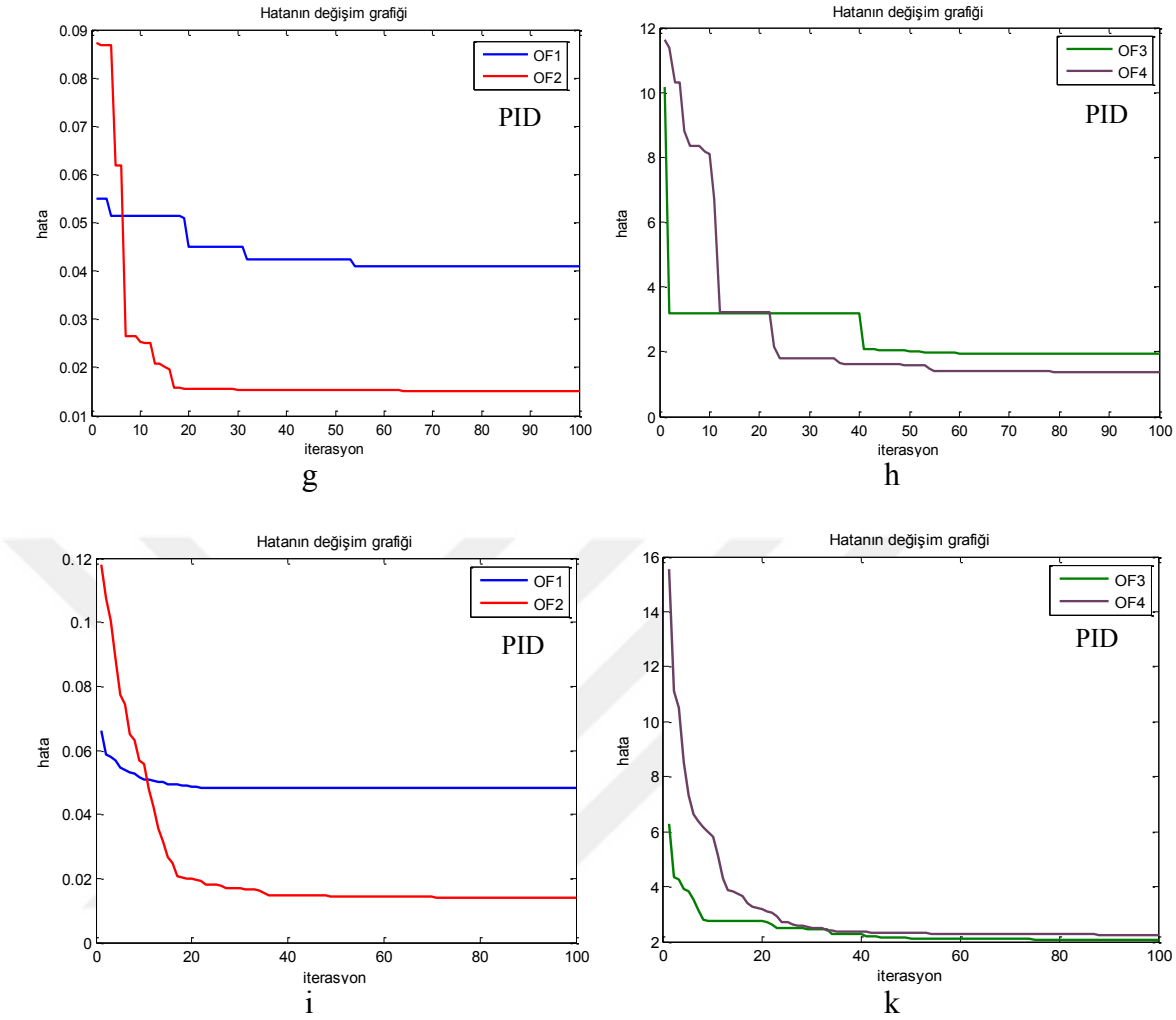
d


Hatanın değişim grafiği
FOPID
OF1
OF2
hata
iterasyon

e


Hatanın değişim grafiği
FOPID
OF3
OF4
hata
iterasyon

f



Şekil 4.14. T_6 süreci için DG algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

T_6 süreci için denetleyici tasarımı gerçekleştirilen son algoritma PSO algoritmasıdır. Tablo 4.21'de PSO algoritması kullanılarak elde edilen FOPID ve PID denetleyici yapılarına ait parametre değerleri verilmiştir. Bu denetleyici yapılarına ait temel performans karakteristikleri ise Tablo 4.22'de sunulmuştur. Tabloya göre her iki denetleyici yapısında da OF_2 fonksiyonunu dikkate alan çalışmanın en yüksek yükselme süresine sahip olduğu, OF_1 fonksiyonunu kullanan çalışmanın ise en yüksek oturma süresine sahip olduğu görülmektedir. Hata değerleri göz önüne alındığında FOPID denetleyici yapısının klasik PID yapısına göre üstün olduğu yine tablodan ifade edilebilir. Standart sapma değerlerinde ise T_6 süreci için PSO algoritmasının en iyi performansa sahip olduğunu söylemek yanlış bir ifade olmayacaktır.

Tablo 4.21. T_6 süreci için PSO algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

Kontrolör	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	35.8890	45.5787	31.666	1.4870	0.879
		OF_2	45.477	39.3904	7.5372	1.0780	0.9736
		OF_3	61.9798	99.7290	76.6259	1.4250	1.4206
		OF_4	53.4827	97.5815	76.0063	1.4250	1.3373
PID	Tez Çalışması	OF_1	6.5140	7.8740	7.1200	-	-
		OF_2	6.5571	5.7476	0.8231	-	-
		OF_3	8.6942	5.9067	1.3395	-	-
		OF_4	9.5993	8.7496	1.8139	-	-

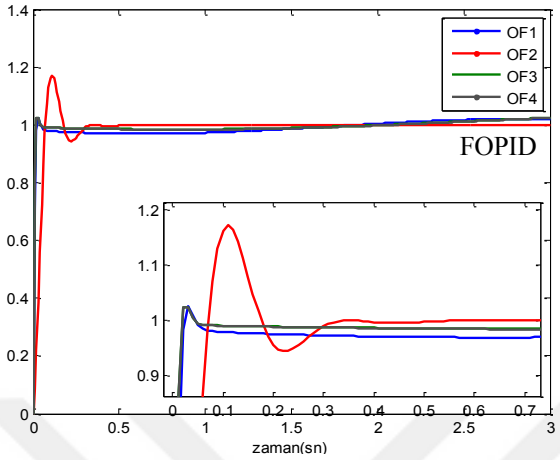
Tablo 4.22. T_6 sürecine ait PSO algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

Kontr.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.0071	-	5.2806	0.0162	0.7689	0.4550	0.00001
		OF_2	-	0.0050	52.6833	0.0518	0.2418	17.1415	0.00001
		OF_3	0.0059	-	0.1462	0.0138	0.0175	0	0.00121
		OF_4	-	0.0536	0.6241	0.0139	0.0176	0	0.00268
PID	Tez Çalışması	OF_1	0.0454		16.9245	0.0722	1.6869	2.6787	0.0098
		OF_2	-	0.0227	2.9994	0.2752	0.3770	0.2958	0.0009
		OF_3	0.0908	-	2.2933	0.2010	0.2770	0	0.0003
		OF_4	-	0.0249	2.3003	0.1687	0.2320	0.2971	0.0068

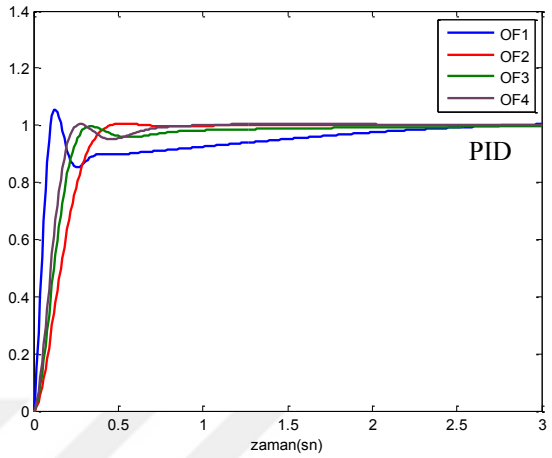
T_6 süreci için PSO algoritması kullanarak elde edilen basamak cevapları ve hata gelişim grafikleri Şekil 4.15'te yer almaktadır. Şekildeki c,d ve g,h grafikleri minimum hata için elde edilen değişimi gösterirken, e,f ve i,k grafikleri ise ortalama hataya ait değişimi sunmaktadır. FOPID denetleyici yapısının bulunduğu Şekil 4.15.a grafiğine göre OF_2 kullanıldığında aşma miktarının yüksek oluşu dikkat çekicidir. Bütün amaç fonksiyonlarında oldukça hızlı cevap hızının olduğu yine Şekil 4.15.a grafiğinden açıkça görülmektedir. PID denetleyiciye ait Şekil 4.15.b grafiğine göre yükselme ve oturma sürelerinin FOPID denetleyiciye göre daha geç olduğu söylenebilir. OF_1 dışında diğer üç amaç fonksiyonunda neredeyse aşma miktarının sıfır olduğu Şekil 4.15.b'den yine açıkça görülebilir.

FOPID denetleyiciye ait Şekil 4.15.c ve d hata gelişim grafiklerinde hata hızlı bir şekilde minimize edilmiştir. Ortalama hata değerlerinin ye aldığı Şekil 4.15.e ve f grafiklerinde hata değeri PSO ile ilk iterasyonlarda hızlı bir şekilde indirgense de son iterasyona kadar hatadaki minimizasyon işleminin sürdüğü görülmüştür. PID denetleyiciye ait Şekil 4.15.g ve h grafiklerinde hata ilk 10 iterasyonda oldukça

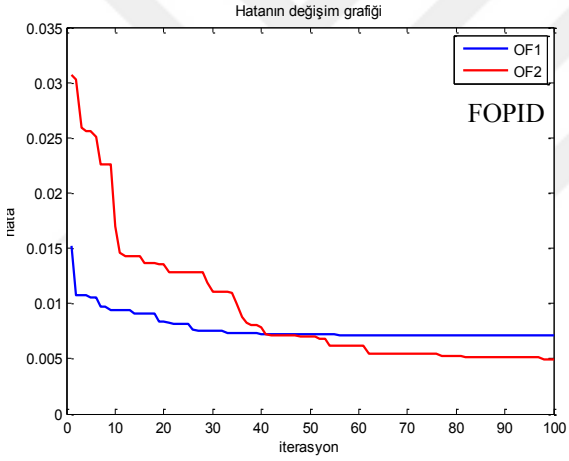
azaltılmış, Şekil 4.15.i ve k grafiklerinde ise bu sürecin daha yüksek iterasyonlara kadar sürdüğü gözlenmiştir.



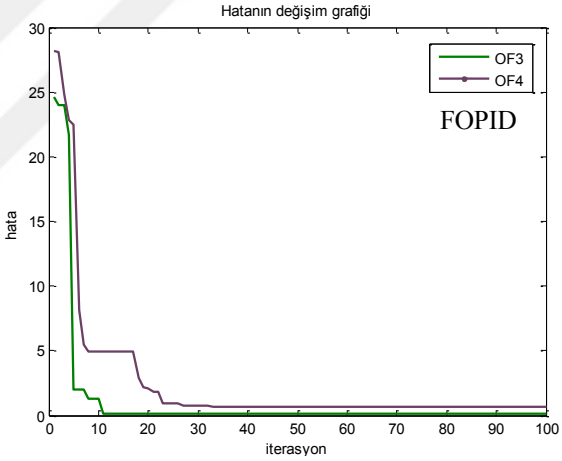
a



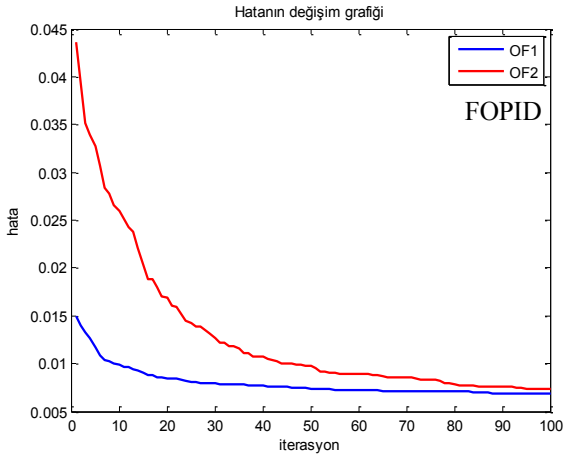
b



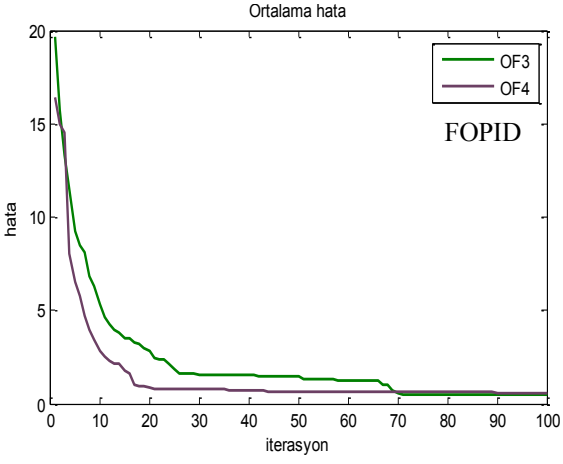
c



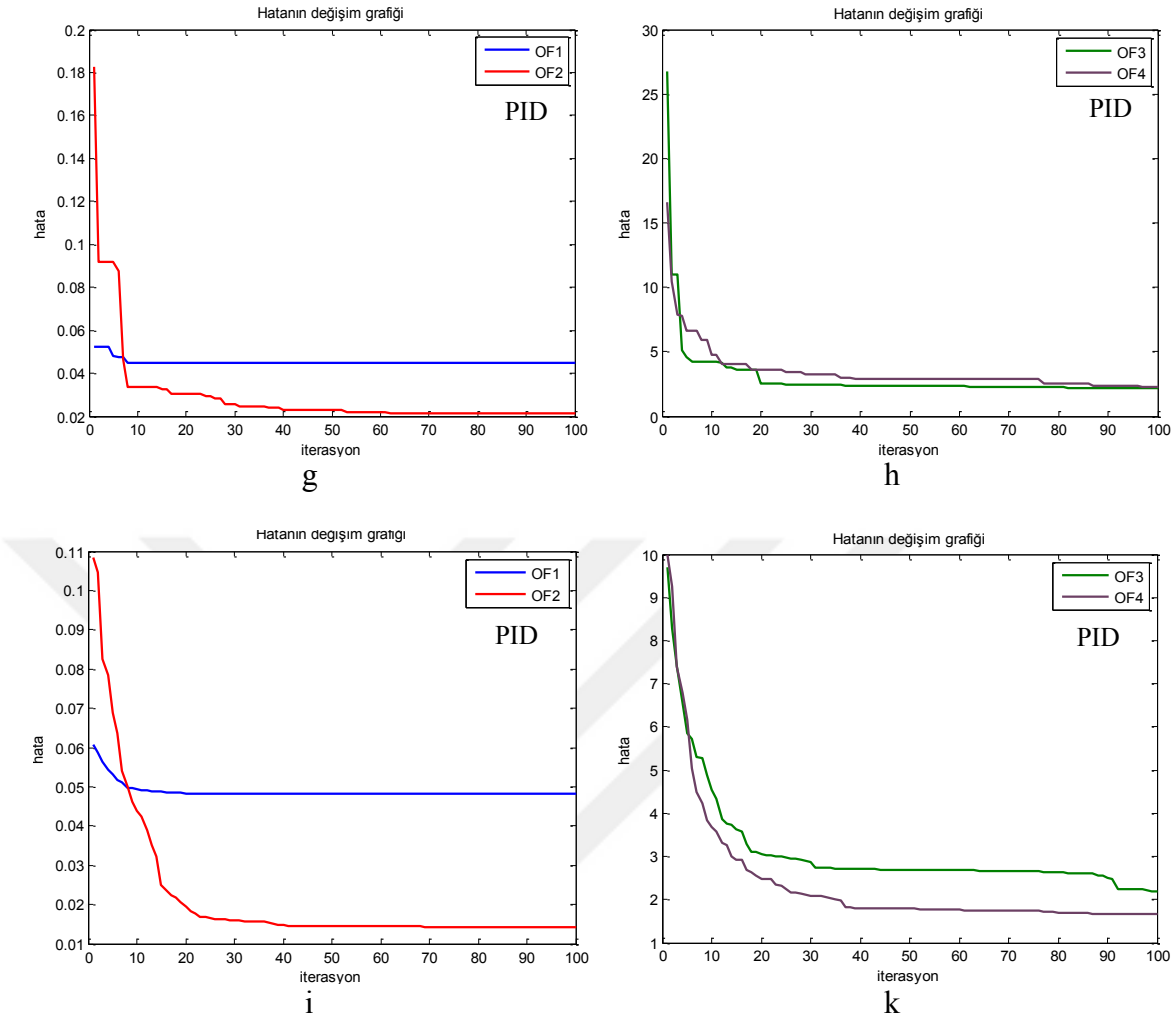
d



e



f



Şekil 4.15. T_6 süreci için PSO algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

Genel bir karşılaştırma yapabilmek için, Tablo 4.23'te T_6 süreci için algoritmaların genel performansını gösteren çıktılara yer verilmiştir. OF_3 amaç fonksiyonuna ait değerlerin yer aldığı bu tabloya göre T_6 süreci için FOPID denetleyici yapısında DG algoritması, PID denetleyicide ise YAK algoritması ile en başarılı sonuç elde edilmiştir. Bu algoritmalara ait OF_3 değerleri sırasıyla 0.1442 ve 1.9175'tir. FOPID denetleyici yapısında DGA sadece OF_3 değerinde değil, diğer bütün performans kriterlerinde de en başarılı algoritmadır. Aynı durum PID denetleyicide YAK algoritması için de geçerlidir. OF_3 amaç fonksiyonun kullanıldığı bütün algoritmalarda aşma miktarının sıfır olarak elde edildiği gözden kaçırılmaması gereken bir durumdur. Standart sapma değerlerinde ise PSO algoritmasının diğer iki algoritmaya göre daha üstün olduğunu söylemek mümkündür.

Tablo 4.23. T_6 sürecine ait optimum denetleyiciler için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması

Kontr.	Algoritma	ISE	OF_3	Tr(sn)	Ts(sn)	OS%	SS
FOPID	YAK	0.0108	0.2596	0.0185	0.0304	0	0.0022
	DGA	0.0057	**0.1442**	0.0132	0.0172	0	0.0464
	PSO	0.0059	0.1462	0.0138	0.0175	0	0.00121
PID	**YAK**	0.0799	**1.9175**	0.1605	0.2237	0	0.0011
	DGA	0.0810	2.0526	0.1803	0.2486	0	0.0024
	PSO	0.0908	2.2933	0.2010	0.2770	0	0.0003

Dördüncü mertebeden bir sistem olan T_7 süreci için YAK algoritması ile elde edilen FOPID ve PID denetleyici parametreleri Tablo 4.24'te verilmiştir. Tablo 4.25'te bu denetleyici yapılarına ait performans değerleri yer almaktadır. Ayrıca kıyaslama amacıyla literatürde yer alan Das ve arkadaşlarının [116] yapmış oldukları çalışmada elde edilen sonuçlar da aynı tabloda gösterilmiştir. Diğer süreçlerle kıyaslandığında T_7 süreci için kontrol tasarımının daha zor olduğunu söylemek Tablo 4.25'ten mümkündür. Elde edilen hata değeri, yükselme ve oturma süreleri ile maksimum aşma miktarının diğer örneklere göre oldukça yüksek olduğu tablodan açıkça görülmektedir. FOPID denetleyici için literatürde Das ve arkadaşlarının yapmış olduğu çalışmada 1.1919 ISE hata değeri elde edilmiştir. Elde edilen performans sonuçlarına göre aynı çalışmada OF_3 değeri 28.3893, OF_4 değeri 34.6439 olarak hesaplanmıştır. Tez çalışmasında bu değerler sırasıyla 8.0155 ve 17.1173 olarak bulunmuştur.

PID denetleyici yapısında ise Das ve arkadaşları 1.1807 ISE değeri ve 2.4618 ITAE değerine ulaşmışlardır. Bu değerler dikkate alınarak hesaplanan OF_3 ve OF_4 değerleri 30.8061 ve 43.6163'tür. Tez çalışmasında ISE 0.4031, ITAE 0.5816, OF_3 14.1888 ve OF_4 24.9692 olarak bulunmuştur. Bu sonuçlar ışığında,YAK algoritması ile elde edilen bütün sonuçların literatürde yer alan çalışmaya göre oldukça başarılı olduğunu söylemek mümkündür. Öte yandan, dikkate alınan süreç için kontrolör tasarımının zor olması nedeniyle standart sapma değerlerinin diğer örneklerde elde edilenlere göre daha yüksek olduğunu da söylemek gereklidir.

Tez çalışmasında iki denetleyici yapısı için her ne kadar ISE ve ITAE hata değerleri hem literatüre hem de diğer amaç fonksiyonlarına göre düşük elde edilse de, aşma miktarları ve oturma sürelerinin oldukça yüksek olması gözden kaçırılmaması gereken bir durumdur. Bu durum OF_3 ve OF_4 fonksiyonlarının kontrolör tasarımındaki başarısını

ortaya koymaktadır. Bu fonksiyonların başarımı Şekil 4.16'dan daha net anlaşılmaktadır.

Tablo 4.24. T_7 süreci için YAK algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

Kontr.	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	1.475	17.258	10.754	0.451	1.328
		OF_2	50.367	71.790	17.935	0.500	1.499
		OF_3	5.148	1.467	2.330	1.022	1.350
		OF_4	3.715	1.841	2.224	0.949	1.211
	Das ve ark. [116]		0.635	0.668	0.509	0.999	0.344
PID	Tez Çalışması	OF_1	2.975	2.743	4.613	--	--
		OF_2	2.744	1.413	1.643	--	--
		OF_3	2.157	1.063	1.154	--	--
		OF_4	2.517	1.206	1.969	--	--
	Das ve ark. [116]		0.973	0.554	0.312	--	--

Tablo 4.25. T_7 sürecine ait YAK algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

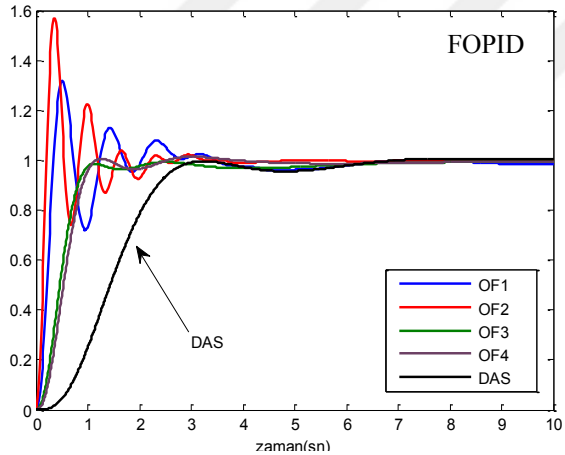
Kontr.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.2088	--	117.7537	0.2132	2.5919	34.2354	0.0571
		OF_2	--	0.3802	185.6713	0.1332	2.0736	57.1670	0.0882
		OF_3	0.3524	--	8.0155	0.6137	0.8983	0	0.9998
		OF_4	--	0.6350	17.1173	0.6611	0.9723	1.9682	1.2541
	Das ve ark. [116]		1.1919	1.8174	28.3893(OF_3) 34.6439(OF_4)	1.8393	2.8171	0.7949	-
PID	Tez Çalışması	OF_1	0.4031	--	101.3454	0.4137	6.1132	22.2495	0.0081
		OF_2	--	0.5816	43.3862	0.7622	3.0518	7.4371	0.0096
		OF_3	0.6396	--	14.1888	1.0472	1.5585	0	0.0771
		OF_4	--	0.8346	24.9692	0.7623	3.1683	0.2607	0.0832
	Das ve ark. [116]		1.1807	2.4618	30.8061 (OF_3) 43.6163 (OF_4)	2.4984	3.7997	0	-

Şekil 4.16.a grafiği FOPID denetleyici yapısı kullanılarak elde edilen T_7 sürecine ait basamak cevaplarını göstermektedir. Grafiğe göre OF_1 ve OF_2 amaç fonksiyonlarının çıkış cevaplarının OF_3 ve OF_4'e göre osilasyonlu ve yüksek aşma miktarına sahip olduğu açıkça görülmektedir. Bu osilasyonlu çıkışlar kontrolör tasarımında çok istenilen değişimler değildir. Das ve arkadaşlarının [116] çalışmasında aşma olmamasına rağmen, kontrolörün cevap hızının OF_3 ve OF_4'e göre oldukça yavaş olduğu görülebilmektedir. PID denetleyici yapısında ise yine en yüksek aşma miktarına OF_1 ve OF_2 amaç fonksiyonları kullanıldığında ulaşılırken, OF_3 ve OF_4 kullanımında ise bu miktarın azaltıldığı görülmektedir. Bu sonuçlar sadece hata değerlerinin göz önünde bulundurulduğu denetleyici tasarımları yerine, aynı zamanda aşma miktarı ve yükselme

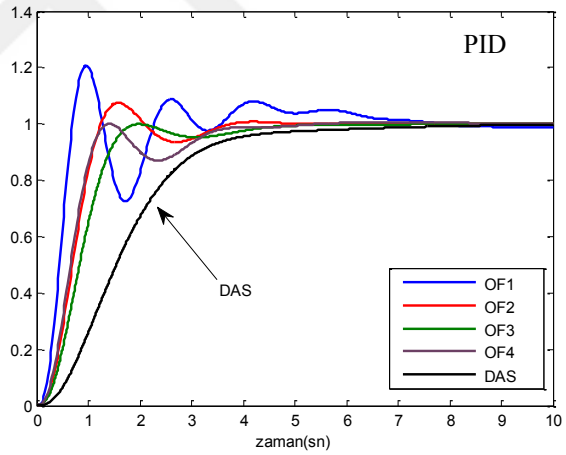
süresi gibi ilave performans özelliklerinin de optimize edildiği çalışmaların başarımını göstermektedir.

FOPID denetleyiciye ait hata gelişim grafiklerinin yer aldığı 4.16.c ve d'ye göre, hata değerinin diğer süreçler göz önünde bulundurulduğunda daha yüksek değerden optimize edilmeye çalışıldığı görülmektedir. Buna rağmen hata değeri neredeyse ilk 30-50 iterasyon arasında oldukça azaltılmıştır. Ortalama hata değerlerinin yer aldığı Şekil 4.16.e ve f grafiklerinde de benzer yorumları yapmak mümkündür.

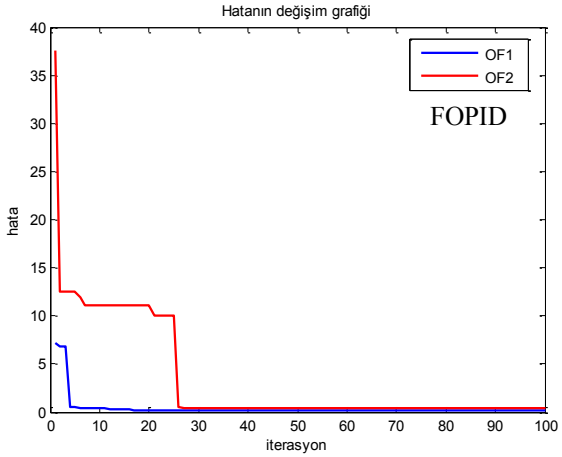
PID denetleyiciye ait Şekil 4.16.g ve h grafiklerinde en iyi sonuçlara ait hata gelişim grafikleri verilmiştir. OF_3 süreci haricinde YAK algoritmasının PID denetleyici tasarımında oldukça yüksek değerlerden hatayı gelişimye çalıştığı görülmektedir. OF_4 fonksiyonunda hata ilk 60 iterasyonda, OF_1 ve OF_2'de ise ilk 30 iterasyonda hata minimize edilmiştir. Ortalama hata değerlerinde bütün amaç fonksiyonlarında hata değerinin ilk 20 iterasyonda oldukça düşürüldüğü görülmektedir.



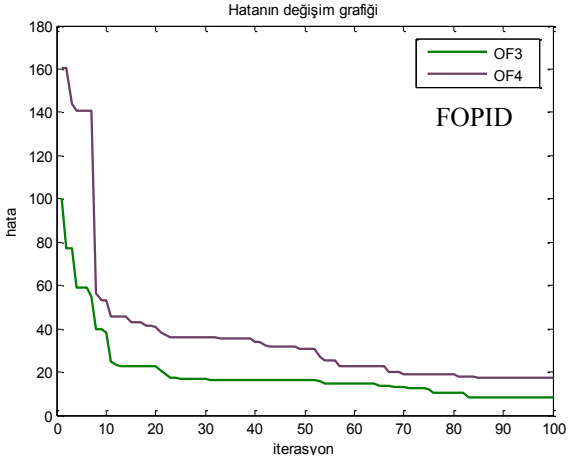
a



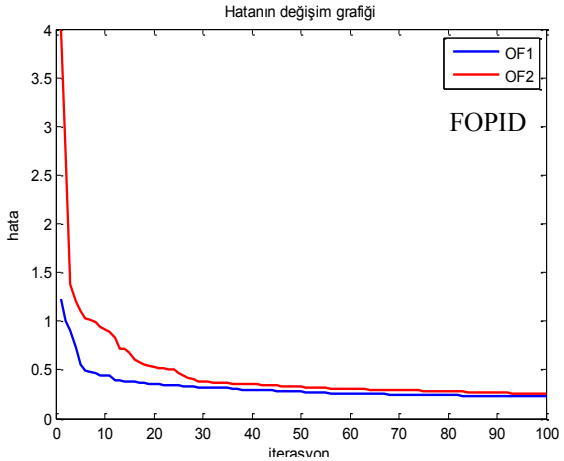
b



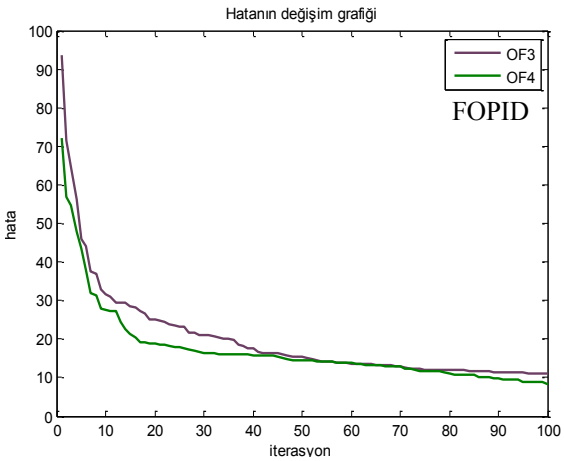
c



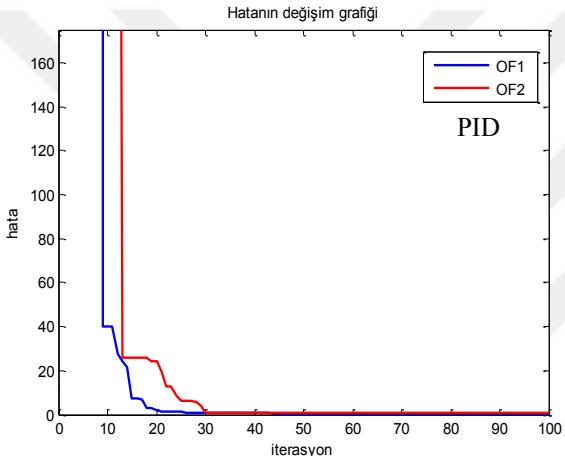
d



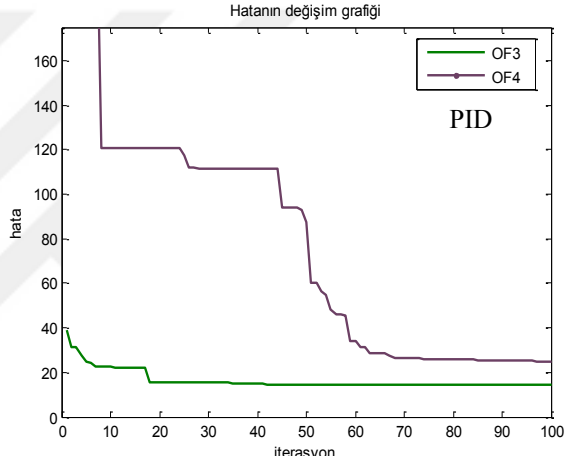
e



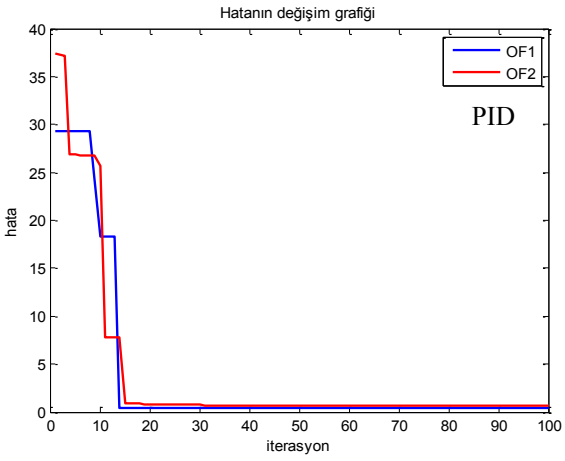
f



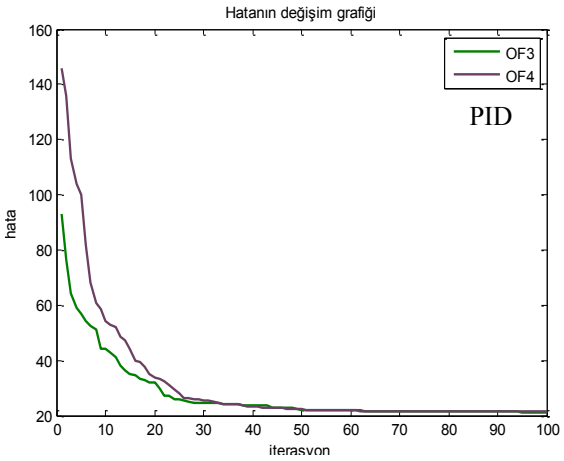
g



h



i



k

Şekil 4.16. T_7 süreci için YAK algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

Tablo 4.26'da DG algoritması kullanılarak T_7 süreci için elde edilen FOPID ve PID denetleyici yapılarına ait parametre değerleri verilmiştir. Denetleyicilere ait performans karakteristikleri ise Tablo 4.27'de verilmiştir. Tabloya göre FOPID denetleyicinin kullanıldığı kontrolör tasarımı çalışmalarında PID denetleyiciye göre daha başarılı sonuçlar elde edildiği görülmektedir. Özellikle aşma miktarı ile oturma sürelerindeki farklılıklar tabloda göze çarpmaktadır. PID denetleyici yapısında kullanılan OF_3 ve OF_4 amaç fonksiyonlarının aşma miktarını yeterince azaltamadığı, FOPID denetleyicide ise aynı fonksiyonların aşma değerini sıfırladığı görülmektedir. Tabloda dikkat çeken başka bir ayrıntı ise standart sapma değerlerindeki farklılıklardır. FOPID denetleyicide ilk iki amaç fonksiyonunda standart sapma değeri tatmin edici seviyede iken, OF_3 ve OF_4 amaç fonksiyonlarında yüksek standart sapma değerleri elde edilmiştir. PID denetleyici için bütün amaç fonksiyonlarında kabul edilebilir standart sapma değerleri elde edilmiştir.

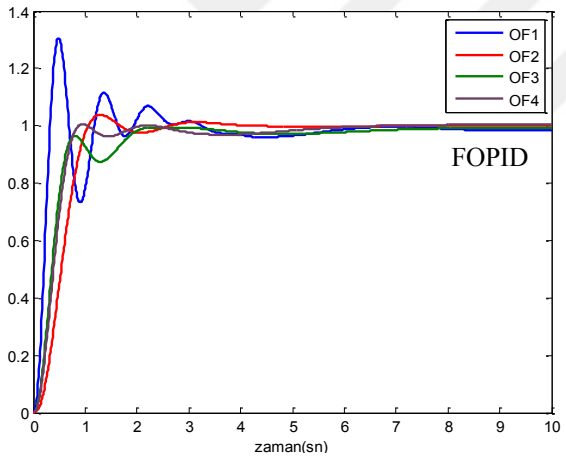
Tablo 4.26. T_7 süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

Kontr.	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	8.0565	12.000	9.3350	1.4248	0.522
		OF_2	4.0430	1.7560	2.1336	1.2068	1.0250
		OF_3	6.1252	1.5805	3.2739	1.4250	1.0193
		OF_4	6.9208	1.4200	2.7587	1.4250	1.1627
PID	Tez Çalışması	OF_1	3.2025	4.6670	4.5200	-	-
		OF_2	2.9143	1.5129	1.7759	-	-
		OF_3	2.3543	1.3483	1.5765	-	-
		OF_4	2.1922	1.1376	1.2872	-	-

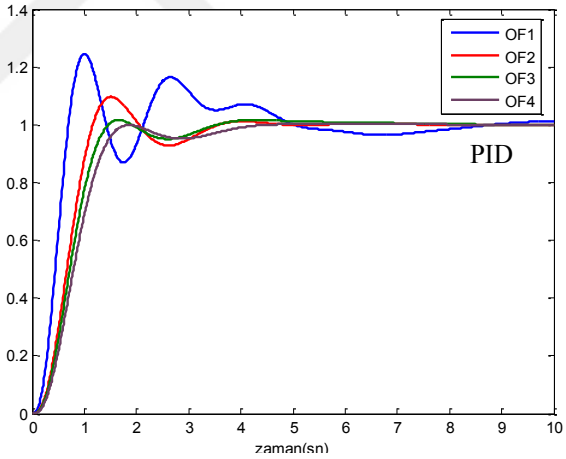
Tablo 4.27. T_7 sürecin ait DG algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

Kontr.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.1961	-	111.487	0.2031	2.4465	32.4312	0.0005
		OF_2	-	0.3117	19.0196	0.6528	0.9575	3.7050	0.0547
		OF_3	0.2820	-	11.5347	0.4709	1.7430	0	1.8793
		OF_4	-	0.5369	9.0969	0.5136	0.7456	0	2.0089
PID	Tez Çalışması	OF_1	0.3931	-	94.5457	0.4306	4.3384	22.9744	0.0001
		OF_2		0.5767	47.0247	0.7316	2.9941	8.7624	0.0003
		OF_3	0.5586	-	16.8306	0.8769	1.3093	1.5661	0.0034
		OF_4	-	0.6492	14.3800	1.0042	1.4994	0.1303	0.0022

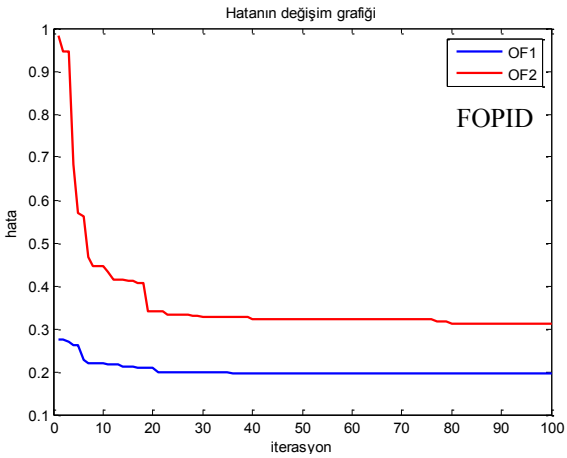
Şekil 4.17'de DG algoritması ile gerçekleştirilen denetleyici yapılarına ait basamak cevapları ve hata gelişim grafikleri yer almaktadır. Şekildeki c,d ve g,h grafikleri minimum hata için elde edilen değişimi gösterirken, e,f ve i,k grafikleri ise ortalama hataya ait değişimi sunmaktadır. FOPID denetleyiciye ait Şekil 4.17.a grafiği dikkatle incelendiğinde OF_1 amaç fonksiyonuna ait yüksek aşma değeri ilk dikkat çeken detaydır. OF_1 ile elde edilen denetleyicide hızlı yükselme süresine ulaşmasına rağmen, hem daha osilasyonlu hem de yüksek aşma değerli bir çıkış cevabı elde etmiştir. Diğer amaç fonksiyonlarında aşma miktarı çok az ya da sıfır olarak elde edilmiştir. PID denetleyici yapısı için elde edilen basamak cevaplarının bulunduğu Şekil 4.17.b grafiğinde de benzer bir durum söz konusudur. Bütün amaç fonksiyonlarında FOPID denetleyici yapısına göre performans farklılıkları görülmektedir. Maksimum aşma değeri, yükselme süresi ve oturma süresi gibi performans özelliklerinin PID denetleyicide daha yüksek elde edildiği Şekil 4.17.b grafiğinden açıkça görülmektedir.



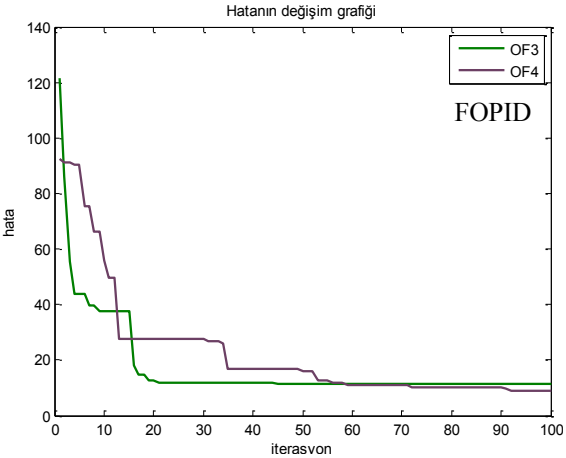
a



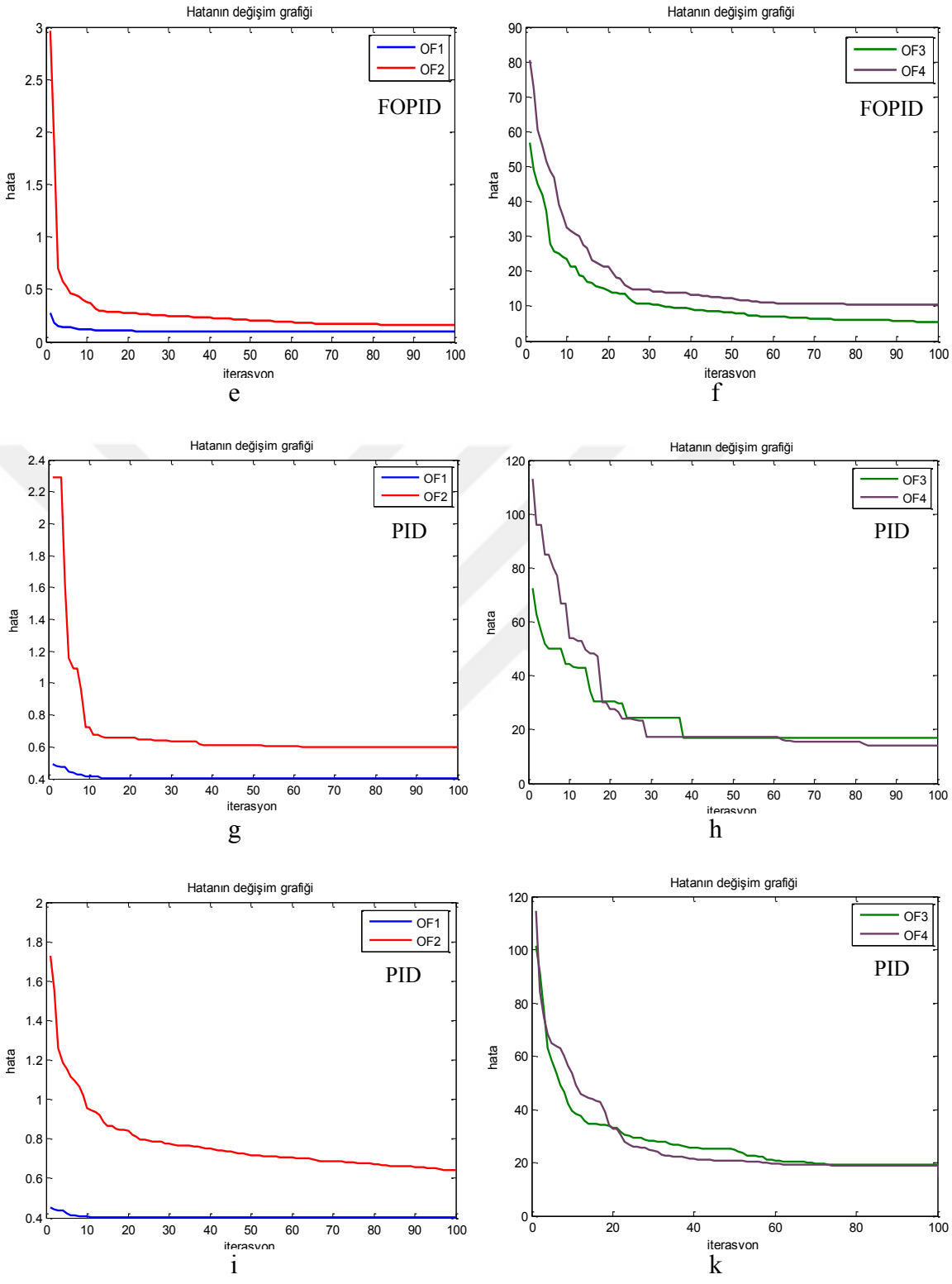
b



c



d



Şekil 4.17. T_7 süreci için DG algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

T_7 süreci için FOPID ve PID denetleyici tasarımı gerçekleştirilen bir diğer algoritma PSO'dur. Algoritma ile elde edilen denetleyici parametreleri Tablo 4.28'de, performans karakteristikleri ise Tablo 4.29'da yer almaktadır. Tablodaki değerlerden, diğer iki algoritmada elde edilen sonuçlara göre OF_3 amaç fonksiyonu haricinde PSO algoritmasının sürecin kontrolünde daha başarısız olduğu söylenebilir. Özellikle OF_1 ve OF_2 amaç fonksiyonları kullanılarak yapılan iki denetleyici tasarımında da yüksek aşma miktarları ve oturma süreleri dikkat çekicidir. Yukarıda bahsedilen yüksek aşma miktarı sistemlerin basamak cevabı değişimleri Şekil 4.18.a ve b'den açıkça görülmektedir.

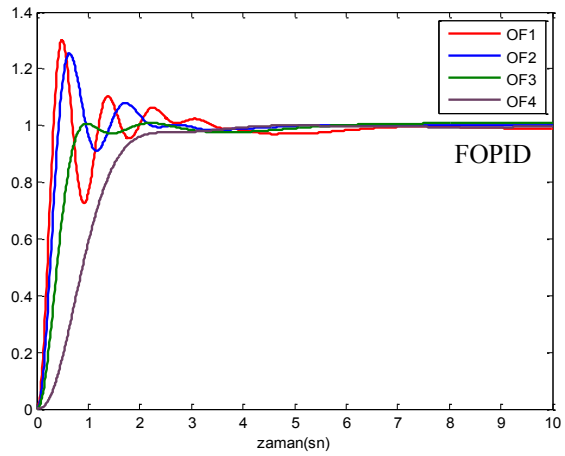
Şekil 4.18'de T_7 sürecine ait çıkış grafikleri yer almaktadır. Yukarıda bahsedilen yüksek aşma miktarı şekilde Şekil 4.18.a ve b grafiklerinden açıkça görülmektedir. Ayrıca FOPID denetleyici yapısının daha hızlı tepki cevabı da bu iki grafikten izlenmektedir. Hata gelişim grafiklerinde ise FOPID denetleyicinin OF_1 ve OF_2 amaç fonksiyonlarında hatayı ilk 20 iterasyonda, OF_3 ve OF_4 ve amaç fonksiyonlarında ise neredeyse son iterasyon değerine kadar minimize ettiği grafik Şekil 4.18.c ve d'den görülmektedir. PID denetleyici yapısında ise ilk 20 iterasyonda bütün amaç fonksiyonları için hata değeri büyük oranda minimize edilmiştir.

Tablo 4.28. T_7 süreci için PSO algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

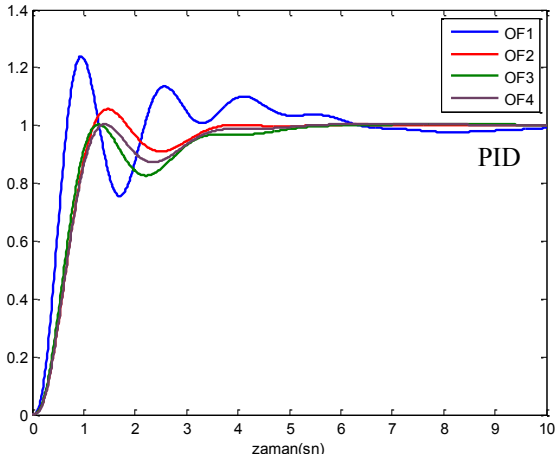
Kontr.	**Yöntem**		**Kp**	**Ki**	**Kd**	**λ**	**μ**
FOPID	Tez Çalışması	OF_1	10.7067	8.3016	9.0612	1.4249	0.6922
		OF_2	14.000	5.0123	5.5967	1.4230	1.0462
		OF_3	6.8496	1.4993	2.6925	1.4250	1.1844
		OF_4	1.7899	1.0764	1.0707	1.0019	0.967
PID	Tez Çalışması	OF_1	3.0425	3.5596	4.7700	-	-
		OF_2	2.832	1.399	1.8921	-	-
		OF_3	2.5733	1.1604	2.2788	-	-
		OF_4	2.5346	1.2197	1.9693	-	-

Tablo 4.29. T_7 sürecine ait PSO algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

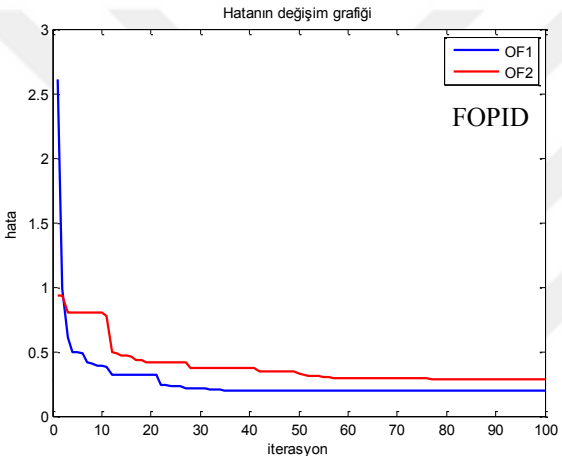
Kontr.	**Yöntem**		**ISE**	**ITAE**	**OF**	**Tr(sn)**	**Ts(sn)**	**OS%**	**SS**
FOPID	Tez Çalışması	OF_1	0.1972	-	108.6366	0.2067	2.4481	31.4749	0.0030
		OF_2	-	0.2872	88.6150	0.2739	1.9340	25.3577	0.1589
		OF_3	0.2990	-	6.8180	0.5269	0.7656	0	7.8432
		OF_4	-	0.8501	19.2169	1.2335	1.8597	0.4724	9.5135
PID	Tez Çalışması	OF_1	0.3858	-	95.6042	0.4162	4.9282	22.3684	0
		OF_2	-	0.5984	38.0158	0.7328	3.0510	5.5923	0.0001
		OF_3	0.5006	-	21.6800	0.7237	3.2377	0.1619	0.028
		OF_4		0.8173	24.8558	0.7815	3.1730	0.2726	0.0035


OF1
OF2
OF3
OF4
FOPID
zaman(sn)

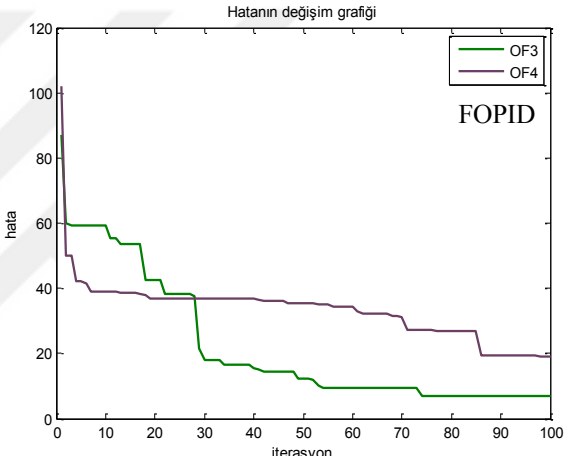
a


OF1
OF2
OF3
OF4
PID
zaman(sn)

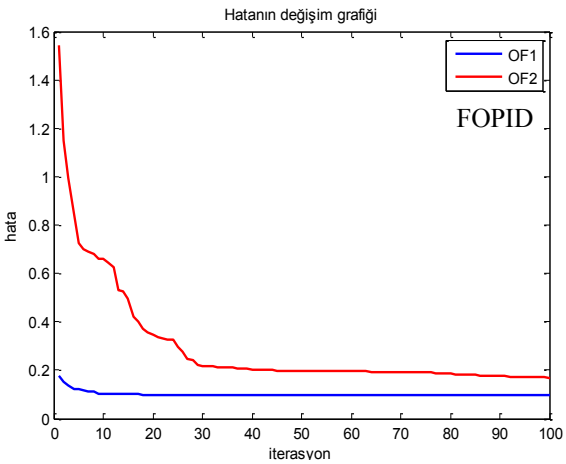
b


Hatanın değişim grafiği
OF1
OF2
FOPID
hata
iterasyon

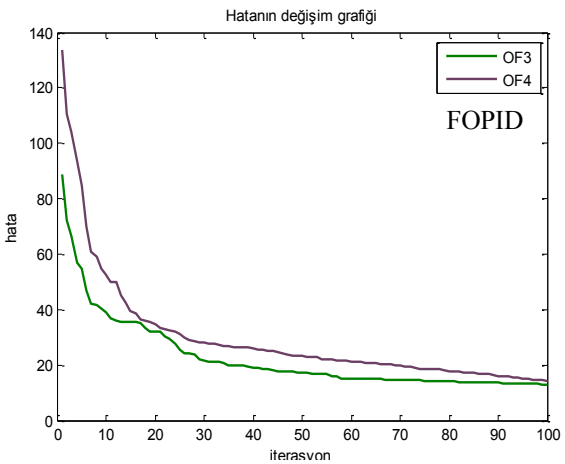
c


Hatanın değişim grafiği
OF3
OF4
FOPID
hata
iterasyon

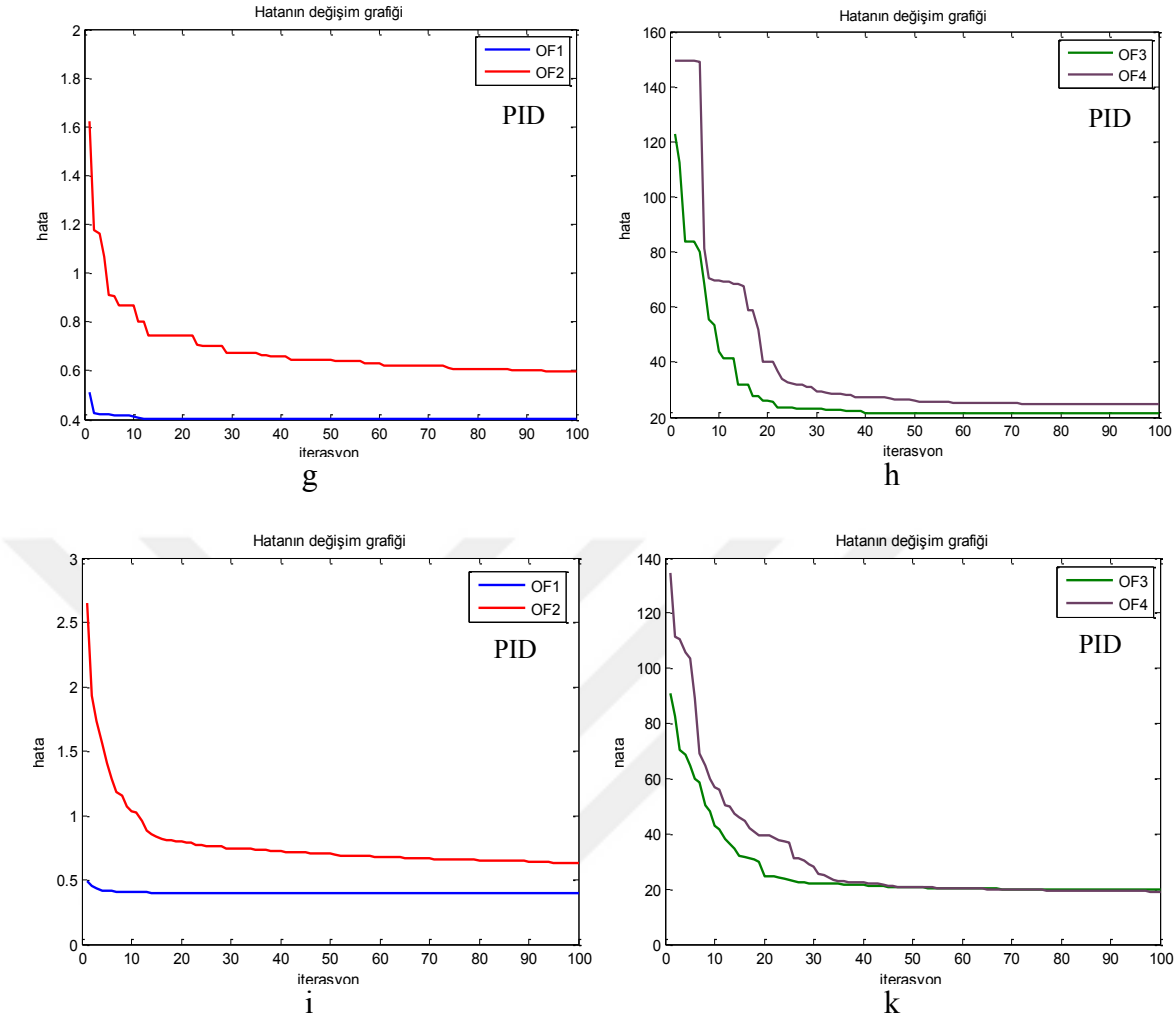
d


Hatanın değişim grafiği
OF1
OF2
FOPID
hata
iterasyon

e


Hatanın değişim grafiği
OF3
OF4
FOPID
hata
iterasyon

f



Şekil 4.18. T_7 süreci için PSO algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

Genel bir karşılaştırma yapabilmek adına, Tablo 4.30'da T_7 süreci için OF_3 fonksiyonu ile algoritmaların elde ettiği çıkış değerleri bir arada yer almaktadır. FOPID denetleyici yapısında PSO algoritmasının 6.8118, PID denetleyici için ise YAK algoritmasının 14.1888 değeri ile diğer algoritmalardan daha başarılı olduğu söylenebilir. PSO algoritmasının FOPID denetleyici tasarımın 7.8432 standart sapma değeri ile sonuca ulaşması gözden kaçırılmaması gereken olumsuz bir durumdur.

Tablo 4.30. T_7 sürecine ait optimum denetleyiciler için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması

Kontr.	Algoritma	ISE	OF_3	Tr(sn)	Ts(sn)	OS%	SS
FOPID	YAK	0.3524	8.0155	0.6137	0.8983	0	0.9998
	DGA	0.2820	11.5347	0.4709	1.7430	0	1.8793
	PSO	0.2990	**6.8180**	0.5269	0.7656	0	7.8432
PID	**YAK**	0.6396	**14.1888**	1.0472	1.5585	0	0.0771
	DGA	0.5586	16.8306	0.8769	1.3093	1.5661	0.0034
	PSO	0.5006	21.6800	0.7237	3.2377	0.1619	0.028

Literatürde yer alan Das ve arkadaşlarının [118] yapmış olduğu çalışmadan örnek alınan T_8 sürecinin kontrolünde diğer örneklerde olduğu gibi dört farklı amaç fonksiyonu kullanılmıştır. Denetleyiciye ait parametre değerleri YAK, DG ve PSO algoritmaları kullanılarak elde edilmiştir. Algoritmaların performanslarını ortaya koyan tablo değerleri ve grafikler ayrıntılı olarak sunulmuştur. Tablo 4.31'de T_8 süreci için YAK algoritması kullanarak elde edilen denetleyici parametreleri yer almaktadır. Tablo 4.32'de ise hata değerleri, aşma miktarı, yükselme ve oturma süreleri ile standart sapma değerlerine yer verilmiştir.

Tablo 4.31. T_8 süreci için YAK algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

Kontr.	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	2.267	1.605	0.688	1.004	1.268
		OF_2	3.700	2.001	1.262	1.020	1.449
		OF_3	2.001	1.429	0.641	1.010	1.359
		OF_4	3.120	1.999	1.057	1.002	1.354
	Das [118] (ISE)		5.324	1.480	0.764	1.544	1.561
	Das 118](ITAE)		1.172	1.046	1.028	1.001	0.803
PID	Tez Çalışması	OF_1	1.916	0.892	1.585	--	--
		OF_2	1.972	1.177	0.913	--	--
		OF_3	2.065	1.189	1.269	--	--
		OF_4	2.074	1.307	1.193	--	--

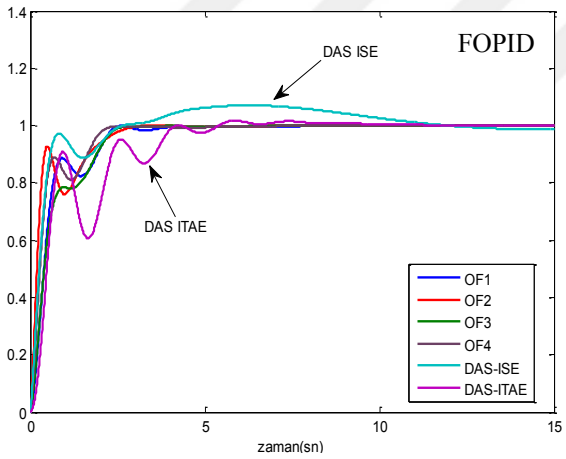
Tablo 4.32. T_8 sürecine ait YAK algoritmasına dayalı optimum denetleyiciler için kullanılarak elde edilen temel performans karakteristikleri

Kontrolör	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez çalışması	OF_1	0.3121	--	13.7313	1.7878	2.1221	0	0.0652
		OF_2	--	0.4526	13.7109	1.5050	1.8342	0.0045	0.0456
		OF_3	0.3356	--	14.0660	1.7845	2.1391	0.0049	1.0214
		OF_4	--	0.3960	13.1942	1.5526	1.8469	0	1.1871
	Das[118] (ISE)		0.2147	--	70.4510	2.6700	9.4008	7.1000	-
	Das[118] (ITAE)		--	1.7622	41.7030	4.1900	3.7962	1.7000	-
PID	Tez çalışması	OF_1	0.4067	--	30.7050	0.4051	5.3376	0	0.0044
		OF_2	--	1.1105	29.2232	0.5298	3.6092	0.0241	0.0097
		OF_3	0.3641	---	20.8080	0.4385	3.3594	0.1235	0.5684
		OF_4	--	0.9498	27.2835	0.4555	3.3520	0.3418	0.6711

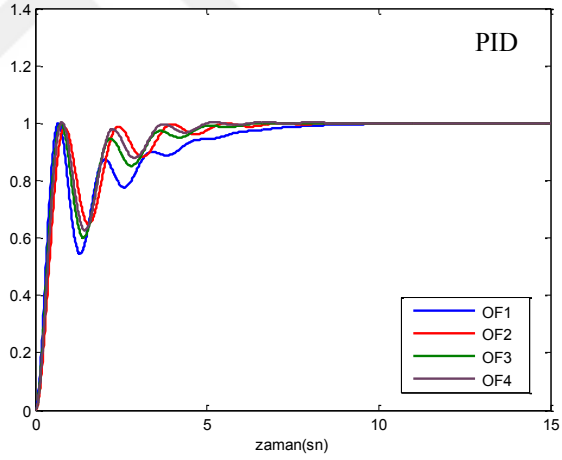
Tablo 4.31' de dört farklı amaç fonksiyonu için elde edilen en düşük hata değerine ait parametre değerleri verilmiştir. Bu değerler YAK algoritmasının toplamda 30 defa yeniden çalıştırılması sonucunda elde edilen en iyi değeri göstermektedir. Tablo 4.32'de ise en iyi denetleyici parametrelerine ait yükselme ve oturma süreleri ile aşma ve hata değerleri yer almaktadır.

Tabloya göre amaç fonksiyonları göz önünde bulundurulduğunda FOPID denetleyici kullanılarak yapılan kontrol işleminde daha başarılı sonuçlar elde edilmiştir. Bu tez çalışmasında bütün amaç fonksiyonlarında elde edilen değerlerin literatürde yer alan Das ve arkadaşlarının [118] yapmış olduğu çalışmaya göre daha başarılı olduğu yine Tablo 4.32'den görünmektedir. Hem FOPID hem de PID kontrolör yapısı dikkate alındığında standart sapma değerlerinin OF_1 ve OF_2 amaç fonksiyonları için oldukça düşük olduğu gözden kaçırılmaması gereken bir durumdur.

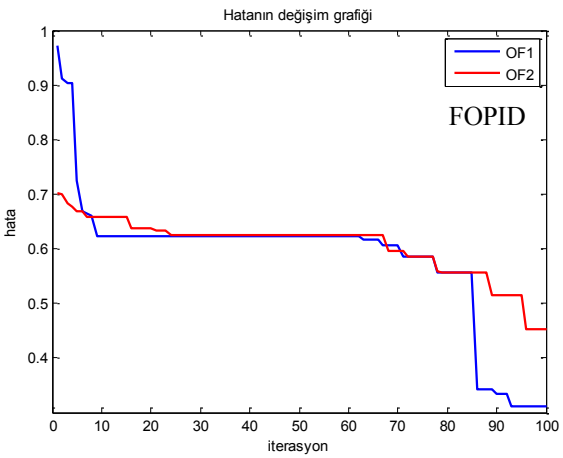
Şekil 4.19'da T_8 süreci için FOPID ve PID kontrölör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri yer almaktadır. YAK algoritmasının kullanıldığı denetleyici tasarım örneğinde Şekil 4.19.a FOPID, Şekil 4.19.b ise PID denetleyiciye ait birim basamak cevabını göstermektedir. Şekildeki c,d ve g,h grafikleri minimum hata için elde edilen değişimi gösterirken, e,f ve i,k grafikleri ise ortalama hataya ait değişimi sunmaktadır.



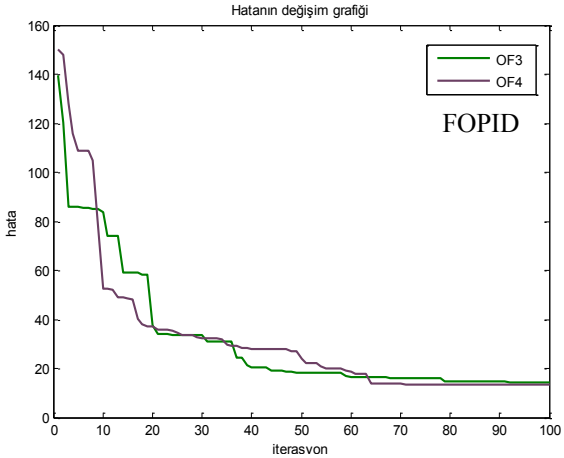
a



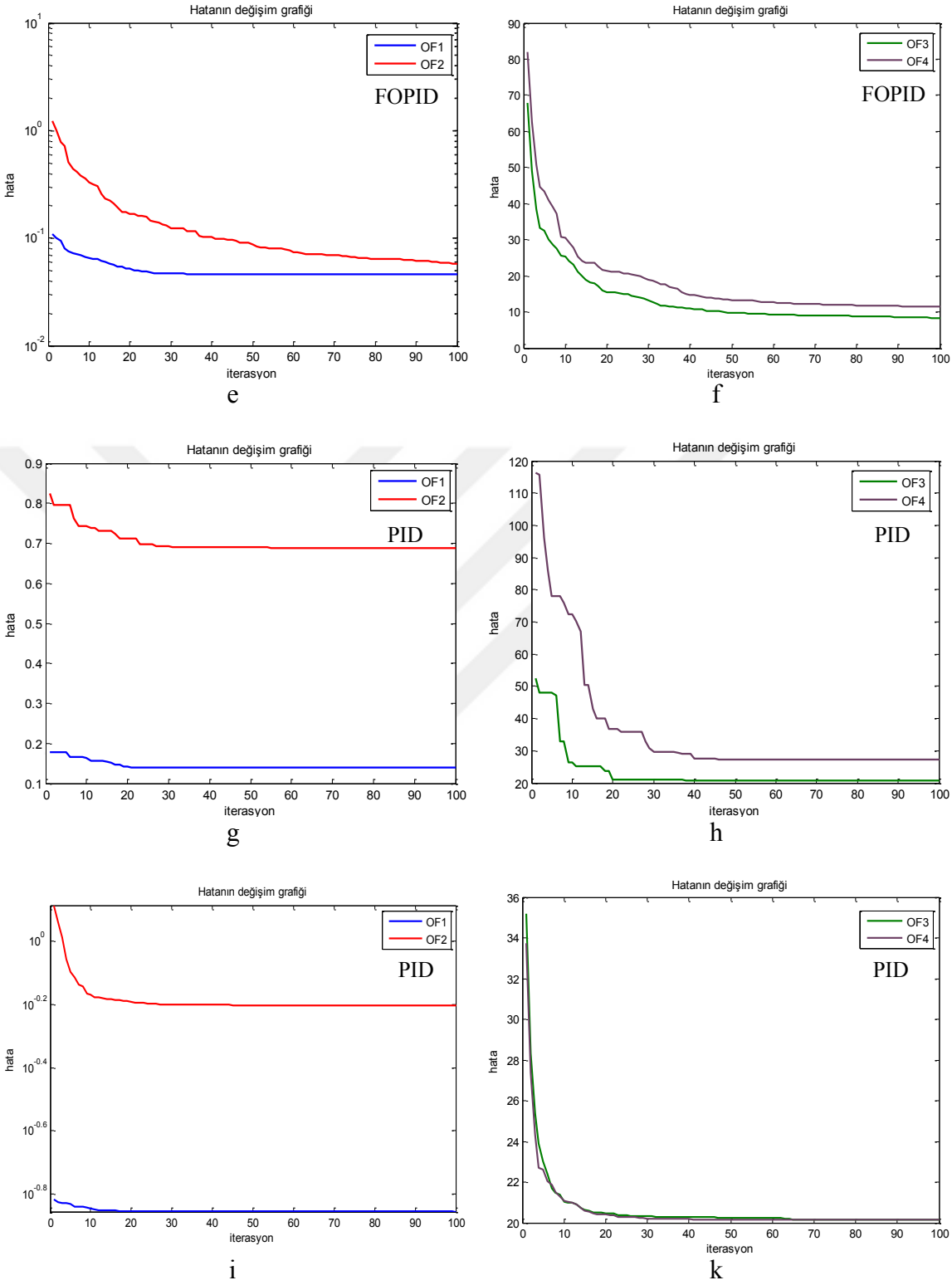
b



c



d



Şekil 4.19. T_8 süreci için YAK algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

Birim basamak cevaplarının yer aldığı bu grafiklerde özellikle yükselme ve oturma süreleri dikkate alındığında, FOPID denetleyicinin PID denetleyiciye göre daha başarılı sonuçlar elde ettiği görülür. Ayrıca literatürde DAS ve arkadaşlarının [118] yapmış olduğu çalışmadaki sonuçlara göre bu tez çalışmasında daha başarılı sonuçlar elde edildiği de yine bu şekillerden görülebilmektedir. Özellikle aşma miktarının azaltılmasında kullanılan OF_3 ve OF_4 amaç fonksiyonlarının hem literatürdeki örneğe göre hem de OF_1 ve OF_2 ile elde edilen sonuçlara göre daha başarılı olduğu söylenebilir.

FOPID kontrolöre ait hata gelişim grafikleri Şekil 4.19.c,d,e,f'de, PID kontrolöre ait grafikler ise Şekil 4.19.g,h,i,k'de yer almaktadır. En düşük ve ortalama hata değerlerinin yer aldığı bu grafiklerde YAK algoritmasının ilk 30 iterasyonda minimum hata değerine oldukça yakınsama sağladığı görülmektedir. Diğer yandan, OF_1 ve OF_2 amaç fonksiyonlarının kullanıldığı PID kontrolöre ait Şekil 4.19.d grafiğinde hata değerinin son iterasyonlara kadar indirgendiği izlenebilmektedir.

T_8 süreci için DG algoritması kullanılarak tasarlanan denetleyici yapılarına ait parametre değerleri Tablo 4.33'te ve optimum denetleyicilere ait performans verileri ise Tablo 4.34'de verilmiştir. Bu bölümde elde edilen sonuçların, YAK algoritması kullanarak yapılan denetleyici çalışmasında elde edilen sonuçlara benzerlik gösterdiği Tablo 4.34'ten söylenebilir. FOPID denetleyici yapısının klasik PID denetleyici yapısına göre daha düşük hata değeri ile sonuca ulaştığı yine aynı tablodan görülmektedir. DG algoritması için OF_1ve OF_2 amaç fonksiyonlarının kullanımının aşma oranını ve oturma süresini dikkate değer bir biçimde düşürdüğü, ancak standart sapma değerini de artırdığı tablodan ifade edilebilecek bir diğer önemli noktadır. Şekil 4.20'de T_8 süreci için DGA ile elde edilen basamak cevapları ve hata gelişim grafikleri sunulmuştur.

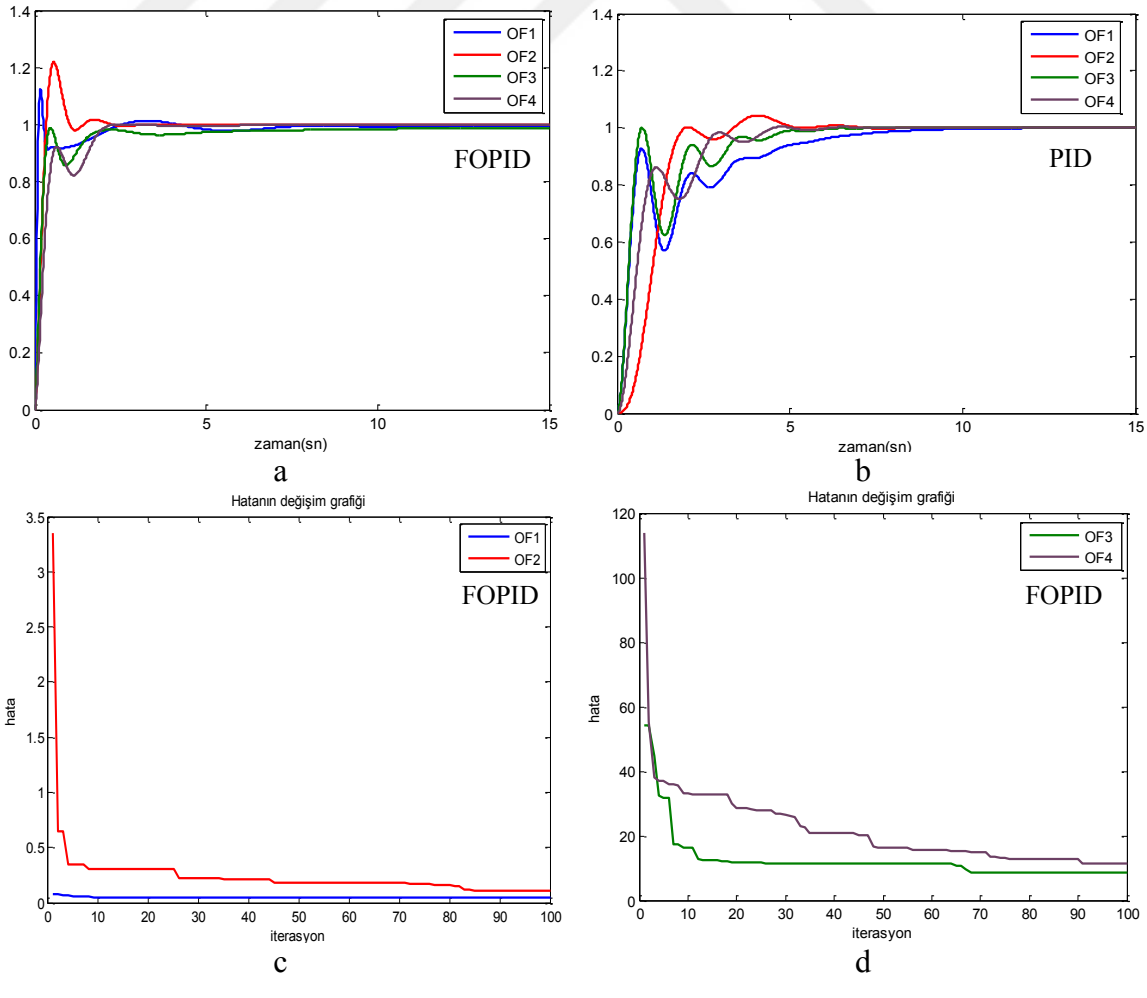
Tablo 4.33. T_8 süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

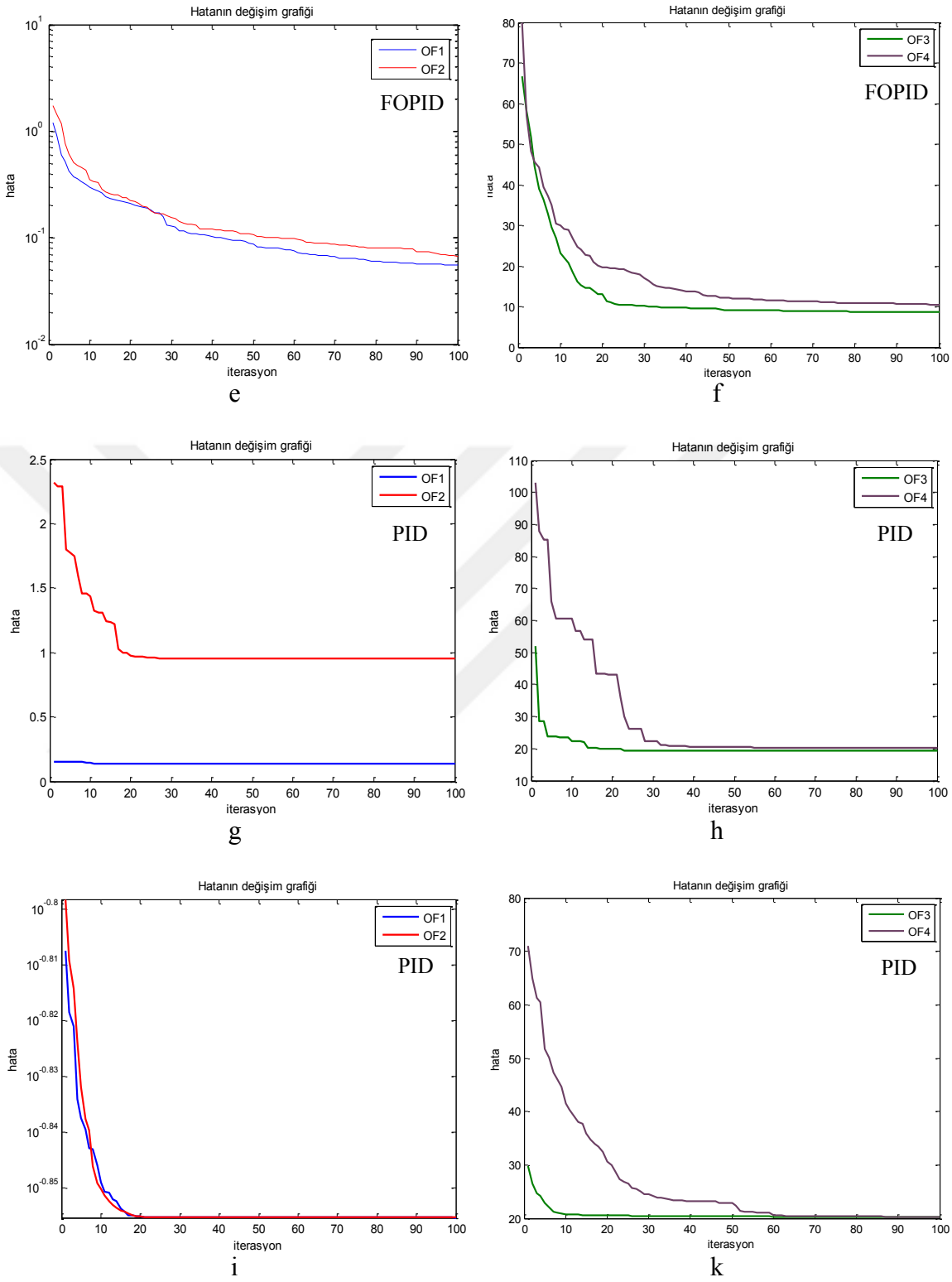
Kontrolör	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	6.5480	12.5770	9.8740	1.4580	0.5409
		OF_2	8.5215	12.1240	1.9079	1.3681	0.9998
		OF_3	5.4406	3.5308	1.9807	1.4210	0.7571
		OF_4	4.2520	2.6528	1.5058	1.3148	0.955
PID	Tez Çalışması	OF_1	1.725	0.810	1.415	-	-
		OF_2	0.6761	1.0000	0.0040	-	-
		OF_3	2.1616	1.2487	1.4079	-	-
		OF_4	1.3807	1.0787	0.4807	-	-

Tablo 4.34. T_8 sürecine ait DG algoritmasına dayalı optimum denetleyiciler için kullanılarak elde edilen temel performans karakteristikleri

Kontr.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.0480	-	46.8142	0.0800	1.4959	12.9516	0.0047
		OF_2	-	0.1070	71.0706	0.2417	0.8941	21.8432	0.0086
		OF_3	0.1331	-	8.8402	0.2688	1.4234	0.1308	2.1489
		OF_4	-	0.6495	10.8906	0.3456	1.6525	0.2799	1.4196
PID	Tez Çalışması	OF_1	0.4393		32.1269	0.4918	5.5467	0	0.0589
		OF_2	-	0.9547	31.1378	1.0717	4.6955	4.3659	0.1254
		OF_3	0.3421	-	20.3500	0.4311	3.3350	0.0848	0.0913
		OF_4		0.9866	24.2899	2.2247	2.6329	0.4197	0.3448

DG algoritmasına dayalı FOPID denetleyici cevabı Şekil 4.20.a ve PID denetleyici cevabı ise Şekil 4.20.b grafiğinde gösterilmiştir. Bu grafiklere göre çalışmada kullanılan FOPID denetleyicilerin daha hızlı yükselme süresine sahip olduğu ve oturma sürelerinin de PID denetleyicilere göre oldukça kısa olduğu söylenebilir. Her iki grafik için de OF_1 ve OF_2 amaç fonksiyonlarının kullanıldığı denetleyici yapılarında aşma miktarının yüksekliği görülmektedir.





Şekil 4.20. T_8 süreci için DG algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

FOPID kontrolöre ait en düşük hatanın gelişim grafiklerinin bulunduğu Şekil 4.20.c,d'de hata oranının ilk 10 iterasyonda çok hızlı bir şekilde düşüş sergilediği ifade edilebilir. Ortalama hata değerlerinin yer aldığı Şekil 4.20.e,f'de ise OF_3 ve OF_4 için hata değeri nihai değerine ilk 40 iterasyonda ulaşmış, OF_1 ve OF_2 için yer alan hata grafiğinde ise hata değerinin son iterasyonlara kadar düşüşüne devam ettiği görülmüştür. PID kontrolöre ati grafiklerde Şekil 4.20.g,h,i,k grafiklerinde ise OF_1 oldukça düşük hata değerinden başlamış, OF_2, OF_3 ve OF_4 ilk 30 iterasyonda nihai değerine ulaşmıştır.

T_8 sürecine ait kontrolör tasarımında kullanılan bir diğer algoritma PSO ile ilgili elde edilen sonuçlar Tablo 4.35 ve Tablo 4.36'da sunulmuştur.

Tablo 4.35. T_8 süreci için PSO algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

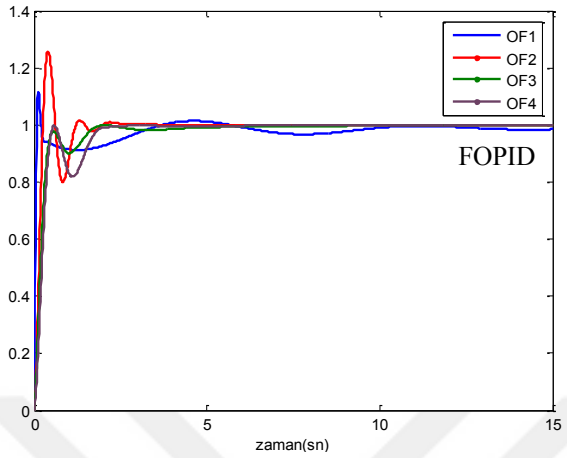
Kontrolör	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	6.9870	9.5470	15.0450	1.4980	0.5629
		OF_2	9.8156	6.9045	3.6917	1.2220	1.0011
		OF_3	5.7103	3.3089	1.4708	1.4580	0.9593
		OF_4	3.5570	2.2520	1.1988	1.3602	1.0032
PID	Tez Çalışması	OF_1	1.598	0.752	1.666	-	-
		OF_2	2.125	1.246	1.413	-	-
		OF_3	1.6415	1.3214	0.7541	-	-
		OF_4	2.1763	1.2600	1.3549	-	-

Tablo 4.36. T_8 sürecine ait PSO algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

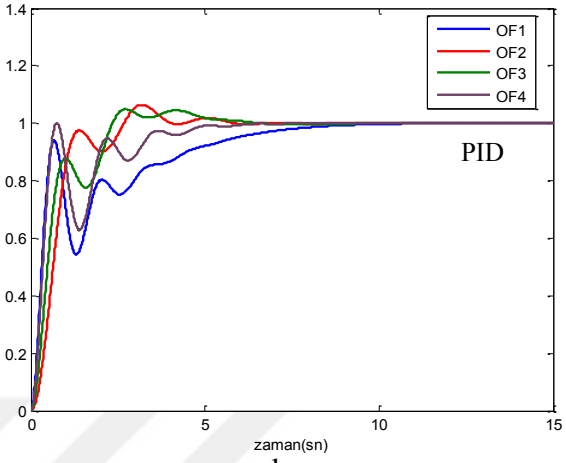
Kontr.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.0426	-	52.0117	0.0573	2.2274	13.4829	0.0009
		OF_2	-	0.1441	84.5160	0.1802	1.1080	25.8449	0.0032
		OF_3	0.1400	-	8.4590	0.3341	1.4118	0	1.2819
		OF_4	-	0.3286	12.1584	0.4649	1.7535	0.0351	1.1325
PID	Tez Çalışması	OF_1	0.4615	-	34.1539	0.4456	5.9013	0.0107	0.0654
		OF_2	-	0.7234	43.7524	0.8249	3.4674	6.3939	0.0547
		OF_3	0.3969	-	29.5098	1.8798	2.1867	4.8693	0.0897
		OF_4	-	1.0778	27.6852	0.4383	3.3432	0.0636	0.0996

Tablo 4.35'de PSO algoritması ile elde edilen denetleyici parametreleri, Tablo 4.36'da ise denetleyicilere ait performans değerleri yer almaktadır. Tablolardaki değerlerden, FOPID kontrolör yapısının kullanıldığı tasarımda elde edilen amaç fonksiyonu değerlerinin PID denetleyici ile elde edilenlere göre oldukça başarılı olduğu bu algoritma için de söylenebilir. Özellikle OF_3 ve OF_4 fonksiyonları için bu durum daha

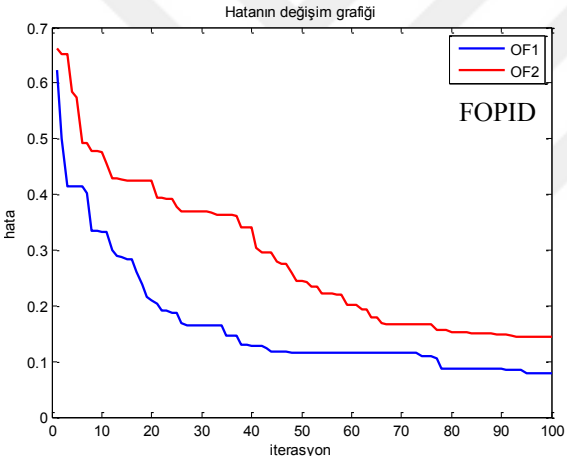
net görülmektedir. Aşma miktarı ve oturma sürelerinin de FOPID denetleyici için oldukça düşük olduğu yine Tablo 4.36'da açıkça görülmektedir.



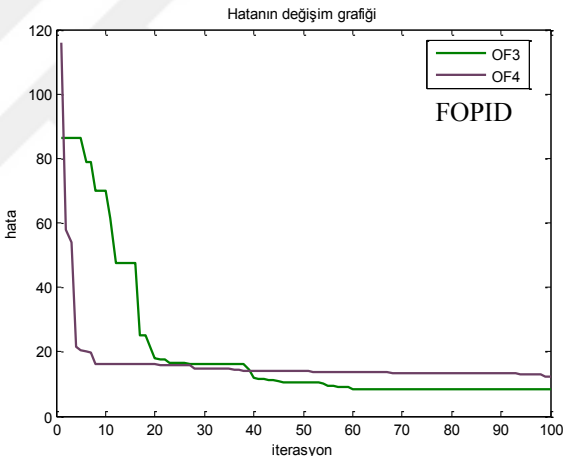
a



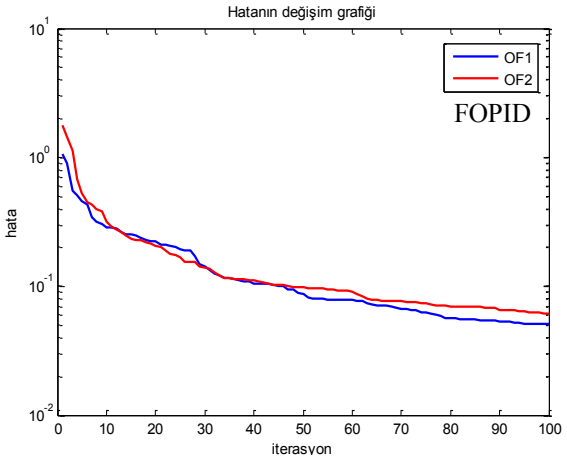
b



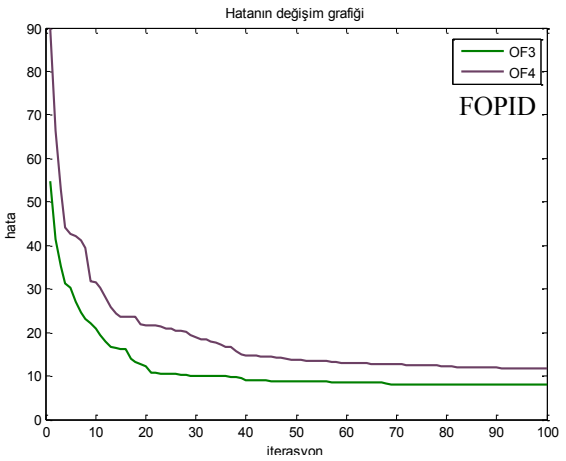
c



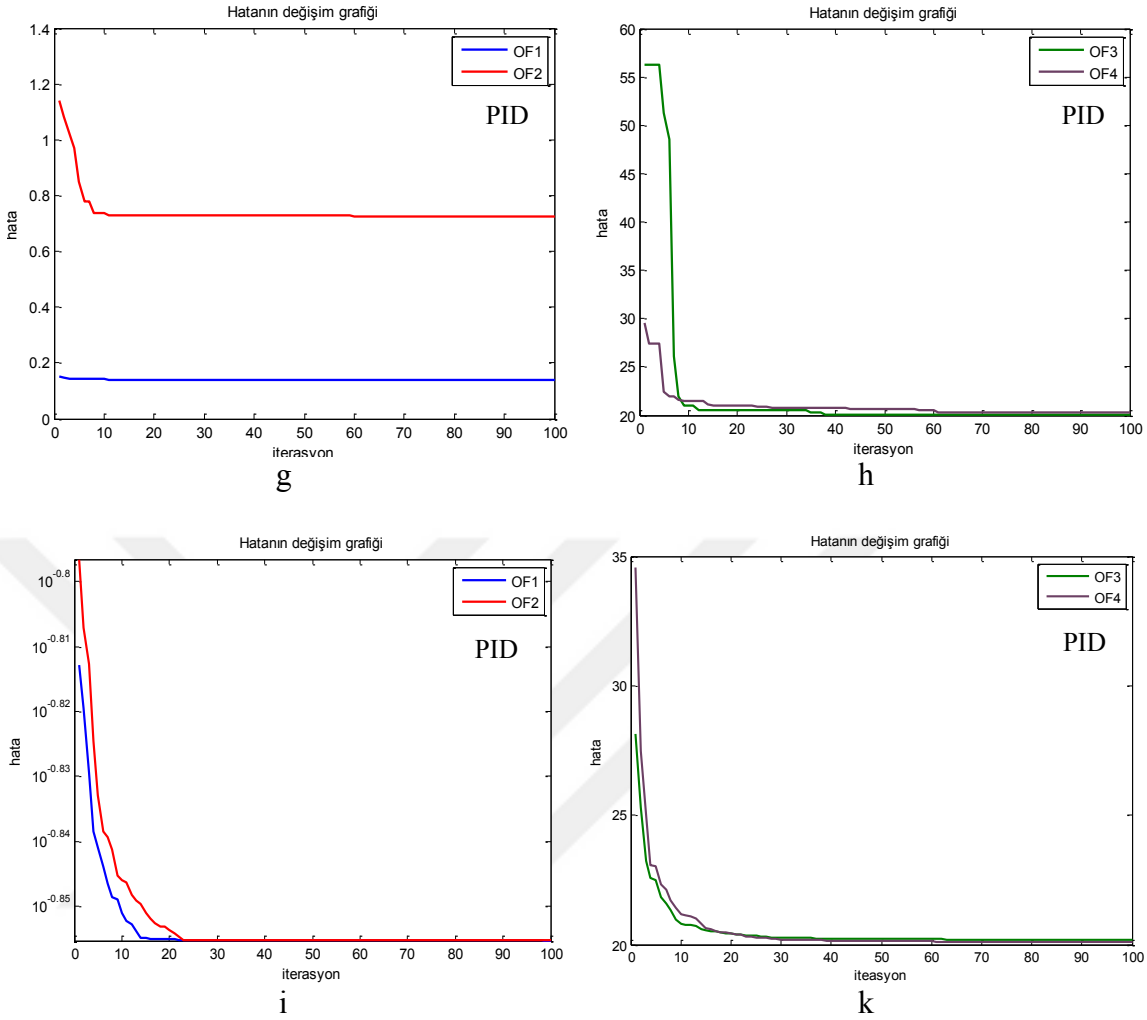
d



e



f



Şekil 4.21. T_8 süreci için PSO algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

Şekil 4.21'de yer alan birim basamak cevapları incelendiğinde FOPID denetleyiciye ait grafikte Şekil 4.21.a grafiğindeki oturma ve yükselme sürelerinin Şekil 4.21.b'dekine göre oldukça düşük oluşu gözden kaçırılmaması gereken bir durumdur. OF_1 ve OF_2 için FOPID denetleyicide aşma değeri oldukça yüksek iken, PID denetleyicide daha düşük bir aşma miktarı gözlenmektedir.

Bu bölümde dikkate alınan T_8 süreci için genel bir değerlendirmek yapabilmek adına, üç algoritmanın kullanımına dayalı olan ve yukarıda sunulan çalışmaların sonuçları toplu olarak Tablo 4.37'de yeniden verilmiştir. Bu sonuçlara göre, sadece ISE ve ITAE gibi tek bir göstergenin yer aldığı amaç fonksiyonu kullanımında YAK ve DG algoritmalarının benzer performans gösterdikleri, PSO algoritmasının ise bu iki algoritmaya göre daha az başarılı performans sergilediği söylenebilir. FOPID denetleyici yapısının PID denetleyiciden daha başarılı olduğu da bu süreç için

söylenebilecek başka bir önemli ifadedir. Birden fazla performans göstergesinin yer bulduğu OF_3 ve OF_4 fonksiyonlarının kullanımının aşma değerini düşürdüğü, yükselme ve oturma sürelerini ise azalttığı tablolardan ve şekillerden açıkça görülmektedir. Standart sapma değerleri göz önünde bulundurulduğunda ise PSO ve DG algoritmalarının YAK algoritmasına göre daha başarılı olduğu söylenebilir.

Tablo 4.37. T_8 sürecine ait optimum denetleyiciler için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması

Kontr.	Algoritma	ISE	OF_3	Tr(sn)	Ts(sn)	OS%	SS
FOPID	YAK	0.3356	14.0660	1.7845	2.1391	0.0049	1.0214
	DGA	0.1331	8.8402	0.2688	1.4234	0.1308	2.1489
	PSO	0.1400	**8.4590**	0.3341	1.4118	0	1.2819
PID	YAK	0.3641	20.8080	0.4385	3.3594	0.1235	0.5684
	DGA	0.3421	**20.3500**	0.4311	3.3350	0.0848	0.0913
	PSO	0.3969	29.5098	1.8798	2.1867	4.8693	0.0897

T8 süreci için, OF_3 amaç fonksiyonu kullanılması durumunda, algoritmalarla elde edilen performans karakteristiklerinin yer aldığı Tablo 4.37'ye göre, PSO algoritmasının 8.4590 OF_3 değeri ile FOPID denetleyici tasarımında en iyi performansa sahip olduğu, DGA'nın ise 8.8402 OF_3 değeri ile bu algoritmaya oldukça yakın bir başarı sağladığı görülmektedir. Diğer süreçler için oldukça başarılı performansa sahip YAK algoritması T8 sürecinde 14.0660 OF_3 değeri ile en başarısız FOPID kontrolör tasarımına sahip algoritma olmuştur. Aşmanın hemen hemen hiç olmadığı FOPID kontrolör tasarımında üç algoritma için de standart sapma değerleri diğer çalışmalara göre oldukça yüksektir.

PID denetleyici yapısının performansı göz önüne alındığında ise DG algoritması 20.3500 hata değeri ile en iyi sonucu elde etmiştir. PSO'da 29.5098, YAK'de ise 20.8080 OF_3 değerleri değerleri, PSO ve YAK'nin bu algoritmaya oldukça yakın performans sergilediğini göstermektedir. DG algoritması 0.0913, PSO 0.0897 ve YAK algoritması da 0.5684 standart sapma değeri ile sonuçlara ulaşım sağlamıştır. Böylece, standart sapma değeri dikkate alındığında PSO algoritmasının, diğer bütün performans karakteristiklerinde ise DGA'nın üstünlüğü göze çarpmaktadır.

4.2. Kesir Dereceli Süreçler için Kontrolör Tasarım Örnekleri

Kontrolör tasarım örneklerinin yer aldığı bu bölüme kadar tam sayı dereceli sekiz süreç ele alınmıştı. Bu süreçler için farklı amaç fonksiyonları kullanılarak üç algoritma ile denetleyici tasarım örnekleri karşılaştırmalı olarak sunulmuştu. Bu aşamadan sonra üç farklı kesir dereceli örnek yapısı ele alınmış ve bunlar Eşitlik 4.13-4.15'de sunulmuştur. Literatürde yer alan T_9[75,119], T_{10}[117], T_{11} [120] süreçleri için diğer örneklere benzer şekilde üç algoritma kullanılmış ve dört farklı amaç fonksiyonu ile elde edilen sonuçlar tablolar halinde verilmiştir.

$$T_9(s) = \frac{1}{0.8s^{2.2} + 0.5s^{0.9} + 1} \tag{4.13}$$

$$T_{10}(s) = \frac{1}{s^{2.6} + 2.2s^{1.5} + 2.9s^{1.3} + 3.32s^{0.9} + 1} \tag{4.14}$$

$$T_{11}(s) = \frac{8.97s^{1.39} - 4.79s^{1.16} + 5.86s^{0.45} + 0.09s^{0.15}}{0.28s^{3.32} + 0.52s^{2.71} + 0.97s^{2.64} + 1.55s^{1.79} + 1.44s^{1.31} + 0.12s^{0.70} + 0.45s^{0.46}} \tag{4.15}$$

Tablo 4.38'de YAK algoritması ile T_9 süreci için tasarlanan denetleyici yapılarına ait parametre değerleri verilmiştir. Tablo 4.39'da ise aynı sürece ait denetleyici yapılarının karakteristik özelliklerini gösteren sayısal veriler yer almaktadır. Tablo 4.39'a göre OF_1 amaç fonksiyonu ile yapılan çalışmada Maiti ve arkadaşlarının [75] YAK algoritmasına göre daha başarılı olduğunu söylemek mümkündür. Maiti'nin çalışmasında 0.0055 ISE değeri elde edilirken, YAK ile yapılan bu çalışmada 0.0065 ISE değeri ile literatürdeki çalışmaya oldukça yakın bir performans sergilenmiştir. Oturma süresi ve aşma miktarını da dikkate alarak hesaplanan OF_3'e göre ise YAK algoritması 0.1939, Maiti ise 1.5483 OF değeri elde etmiştir. Zhao ve arkadaşlarının [119] çalışmasında ise ISE değeri 0.0614, OF3 değeri ise 68.5459 olarak elde edilmiştir. Performans karakteristiklerinin tamamı dikkate alındığında YAK algoritmasının üstünlüğü göze çarpmaktadır.

PID denetleyici tasarımında ise Maiti 0.0142 ISE, 28.8296 OF_3değeri elde etmiştir. Bu denetleyici için YAK algoritması kullanılarak 0.0120 ISE ve 12.2249 OF_3 değerine ulaşılmış ve böylece literatürde yer alan çalışmadan daha iyi sonuç elde edilmiştir. Denetleyiciler arasındaki kıyaslamada ise FOPID denetleyicinin PID denetleyiciye göre daha başarılı olduğunu söylemek mümkündür. Özellikle aşma miktarları dikkate

alındığı zaman aradaki performans farkı oldukça dikkat çekicidir. FOPID denetleyicide yüzde aşım miktarı bütün amaç fonksiyonları için 0.5 değerinin altında iken, PID denetleyicide yine bütün amaç fonksiyonları için 3 değerinin üzerindedir. Hem literatürdeki çalışmalar ile tez çalışması arasındaki hem de denetleyicilerin kendi arasındaki performans farklılıkları, basamak cevaplarının ve hata değişimlerinin sunulduğu Şekil 4.22'den daha net anlaşılmaktadır.

Tablo 4.38. T_9 süreci için YAK algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

Kontr.	Method		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	23.300	85.121	94.498	0.326	1.320
		OF_2	83.316	55.123	58.551	0.468	1.431
		OF_3	61.507	50.406	95.401	0.359	1.500
		OF_4	11.658	90.102	92.338	0.500	1.499
	Maiti ve ark. [75]		442.68	324.03	115.27	1.500	1.410
	Zhao ve ark.[119]		233.423	22.397	18.527	0.100	1.150
PID	Tez Çalışması	OF_1	10.101	50.599	98.499	-	-
		OF_2	7.743	47.527	32.885	-	-
		OF_3	3.091	18.343	30.454	-	-
		OF_4	3.123	50.605	93.119	-	-
	Maiti ark. [75]		214.84	361.57	76.76	-	-

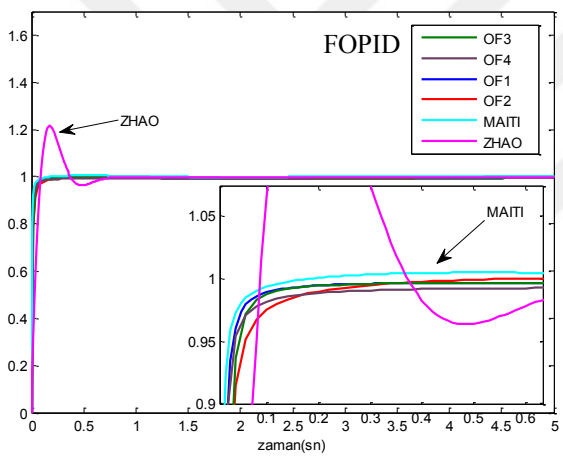
Tablo 4.39. T_9 sürecine ait YAK algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

Kontr.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.0065		0.6171	0.0229	0.0345	0.1265	0.0092
		OF_2		0.0609	2.1184	0.0267	0.0471	0.4245	0.0001
		OF_3	0.0057		0.1939	0.0157	0.0267	0.0011	0.0052
		OF_4		0.0394	0.5462	0.0165	0.0293	0.0018	0.0061
	Maiti ve ark. [75]		0.0055	0.0098	1.5483(OF_3) 1.5911(OF_4)	0.0170	0.0276	0.4516	-
	Zhao ve ark.[119]		0.0291	0.0614	68.5459(OF_3) 68.8689(OF_4)	0.0665	0.3247	22.2105	-
PID	Tez Çalışması	OF_1	0.0120		12.2249	0.0373	0.0472	3.9562	0.0078
		OF_2		0.0093	18.8996	0.0789	0.2257	5.8928	0.0011
		OF_3	0.0286		10.7249	0.0880	0.1100	3.2963	0.0046
		OF_4		0.0636	12.1218	0.0393	0.0490	3.7468	0.0019
	Maiti ve ark. [75]		0.0142	0.0103	28.8296(OF_3) 28.7907(OF_4)	0.0402	0.1763	9.2688	-

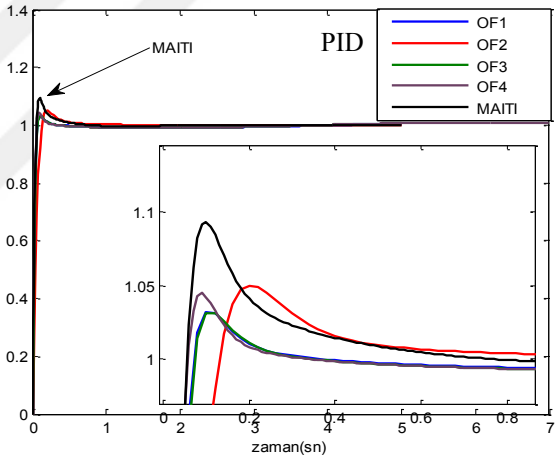
Şekil 4.22.a grafiğinde FOPID denetleyici için Şekil 4.22.b grafiğinde ise PID denetleyici için elde edilen basamak cevapları yer almaktadır. Şekildeki c,d ve g,h grafikleri minimum hata için elde edilen değişimi gösterirken, e,f ve i,k grafikleri ise ortalama hataya ait değişimi sunmaktadır. Özellikle Zhao'nun yapmış olduğu çalışmaya [119] göre YAK algoritmasının performansının daha başarılı olduğu Şekil 4.22.a'dan

grafiğinden açıkça görülmektedir. . FOPID'e ilişkin tüm çalışmalarda YAK algoritması ile elde edilen aşım miktarı neredeyse sıfır iken, Zhao'nun yapmış olduğu çalışmada [119] yaklaşık %20 gibi bir aşma söz konusudur. PID denetleyiciye ait Şekil 4.22.b grafiğinde ise bütün tasarımlar için dikkate değer bir aşma miktarı söz konusu iken, bu oran en fazla Maiti'ye ait çalışmada [75] ortaya çıkmıştır.

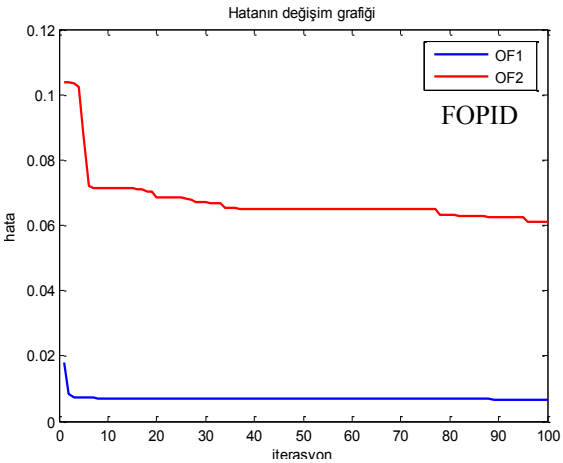
Diğer örneklerde olduğu gibi OF_1 ve OF_2 amaç fonksiyonlarında optimizasyon minimum hataya oldukça yakın bir değerden başlamış ve kısa süre içerisinde hata değeri minimize edilmiştir. OF_3 ve OF_4 fonksiyonlarında ise nispeten daha yüksek bir değerden optimizasyona başlayan YAK algoritması neredeyse son iterasyona kadar hatayı minimize etmiştir. Ortalama hata grafiklerinin yer aldığı Şekil 4.22.i,k incelendiğinde benzer bir ilerleyişin söz konusu olduğu görülmektedir.



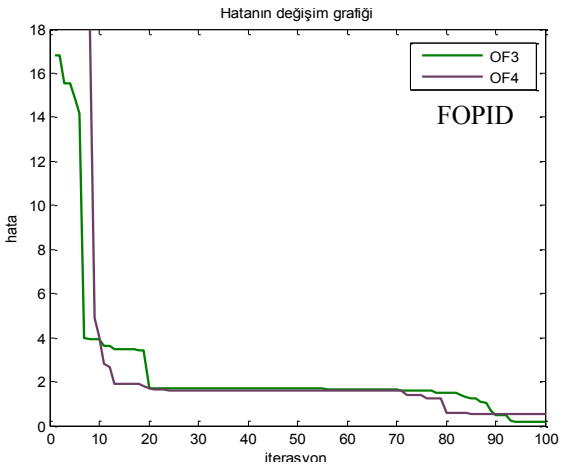
a



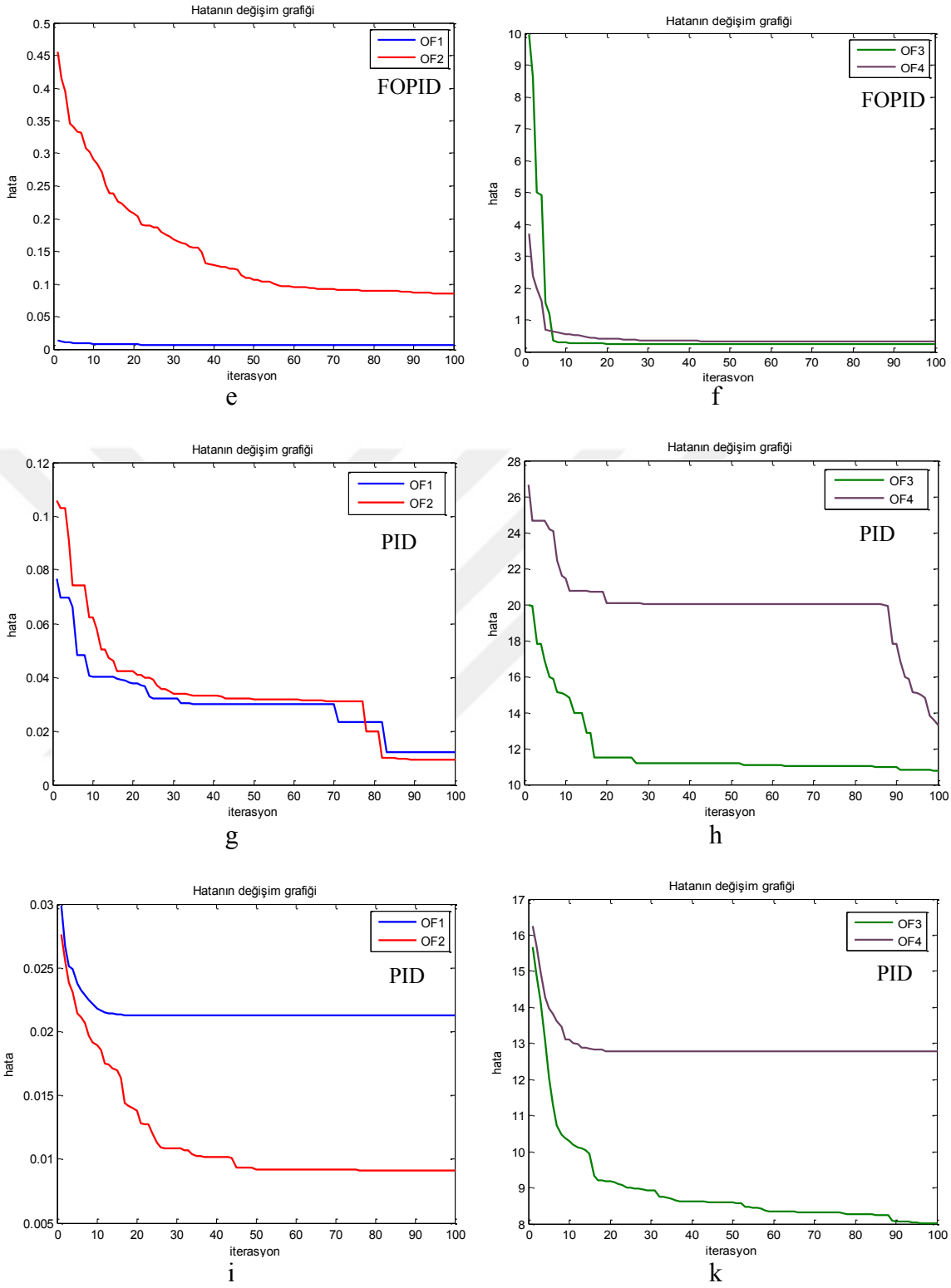
b



c



d



Şekil 4.22. T_9 süreci için YAK algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

DG algoritması kullanarak T_9 süreci için elde edilen FOPID ve PID denetleyici yapılarına ait parametre değerleri ve performans karakteristikleri Tablo 4.40 ve Tablo

4.41'de yer almaktadır. Bu algoritma ile elde edilen sonuçlara göre ISE ve ITAE hata değerlerinin PID denetleyici çalışmasında daha düşük elde edildiği tablodan görülmektedir. Şimdiye kadar yer alan örneklerin neredeyse tamamında FOPID denetleyicinin üstünlüğü göze çarpmakta iken bu süreçte farklı bir durumun ortaya çıktığı görülmektedir. Ancak OF_3 ve OF_4 amaç fonksiyonları kullanarak yapılan denetleyici çalışmalarında ise FOPID denetleyici daha başarılı sonuçlar elde etmiştir. Aşma miktarı FOPID denetleyicide OF_1 dışında kalan üç fonksiyonda sıfırlanırken, PID denetleyicide en düşük 3.5412 değeri elde edilmiştir.

Tablo 4.40. T_9 süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

Kontr.	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	25.5455	17.5775	12.2580	1.3330	1.3776
		OF_2	41.9557	38.3306	35.897	1.4699	1.0545
		OF_3	61.2347	33.5624	75.280	1.4508	0.511
		OF_4	72.0821	65.0520	65.3587	1.4878	1.0050
PID	Tez Çalışması	OF_1	3.987	15.547	65.5489	-	-
		OF_2	13.3728	85.555	59.4229	-	-
		OF_3	0.001	49.7681	83.365	-	-
		OF_4	9.357	56.6061	90.205	-	-

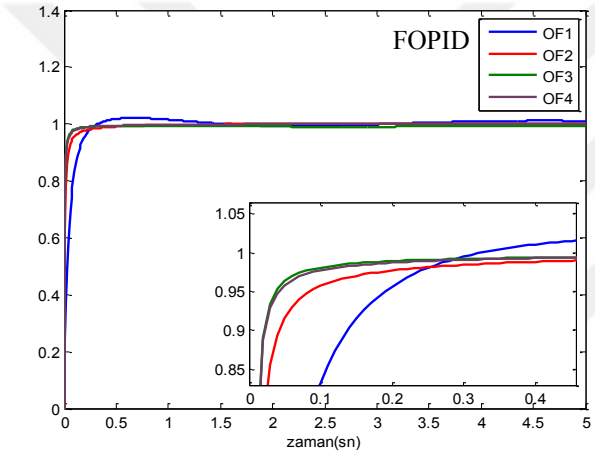
Tablo 4.41. T_9 sürecine ait DG algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

Kontr.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.0250		4.4164	0.1407	0.2057	1.0461	0.0086
		OF_2		0.0068	0.4970	0.0415	0.0859	0	0.0002
		OF_3	0.0062	-	0.2403	0.0198	0.0357	0	0.0007
		OF_4	-	0.0073	0.2860	0.0219	0.0426	0	0.0004
PID	Tez Çalışması	OF_1	0.0174	-	18.7693	0.0474	0.1418	5.9619	0.0006
		OF_2	-	0.0051	16.8366	0.0519	0.1370	5.3667	0.0001
		OF_3	0.0134	-	11.0249	0.0427	0.0533	3.5412	0.0371
		OF_4	-	0.0526	12.7694	0.0401	0.0496	3.9984	0.0001

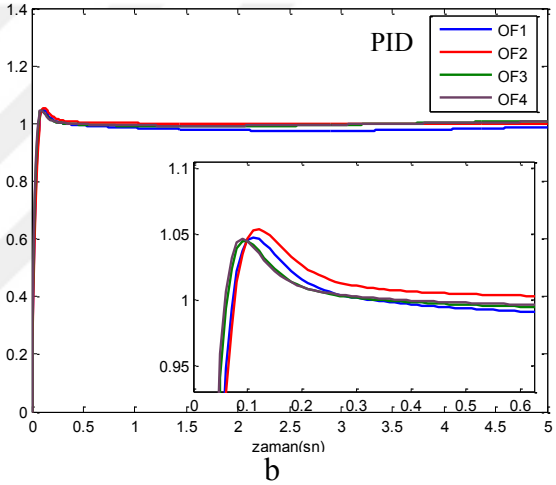
Şekil 4.23'te FOPID denetleyicinin basamak cevaplarının yer aldığı (a) grafiğinde bütün fonksiyonların birbirine yakın basamak cevapları elde ettiği görülmektedir. Sadece OF1 amaç fonksiyonun dikkate alındığı çalışmada yükselme ve oturma süresinin yüksek, aşma miktarının diğer fonksiyonların kullanımlarında elde edilenden fazla olduğu görülmektedir.

PID denetleyiciye ait Şekil 4.23.b grafiğinde ise OF_2 fonksiyonun çok az da olsa aşma miktarının diğer fonksiyonların kullanımlarında ulaşılandan yüksek olduğu

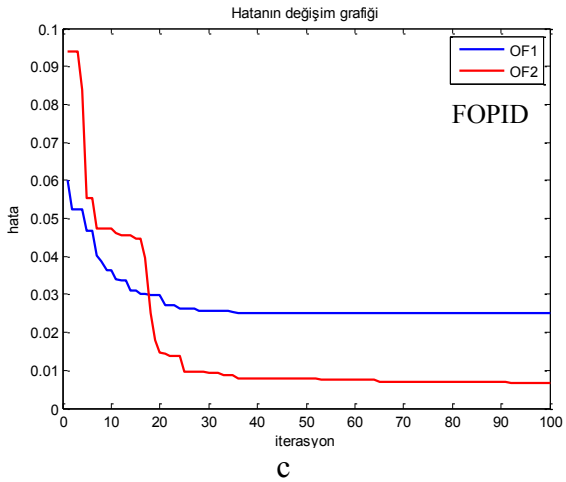
görülmektedir. FOPID denetleyiciye ait Şekil 4.23.c,d,e,f hata gelişim grafiklerinde hatanın OF_1 ve OF_2 için ilk 20 iterasyonda, OF_3 ve OF_4 içinse 10-15 iterasyonlar arasında çok hızlı bir şekilde minimize edildiği görülmektedir. PID denetleyiciye ait hata gelişim grafiklerinde ise OF_1 ve OF_2 için hata değeri ilk 30 iterasyonda oldukça azaltılmış, OF_3'te ilk 40 iterasyon ve OF_4'te ise ilk birkaç iterasyonda minimize edilmiştir. Diğer yandan, ortalama hata değerlerinin yer aldığı grafiklerde ise kesir dereceli denetleyici yapısının tam sayı dereceli denetleyiciye göre daha hızlı bir şekilde hatayı indirgediği söylenebilir.



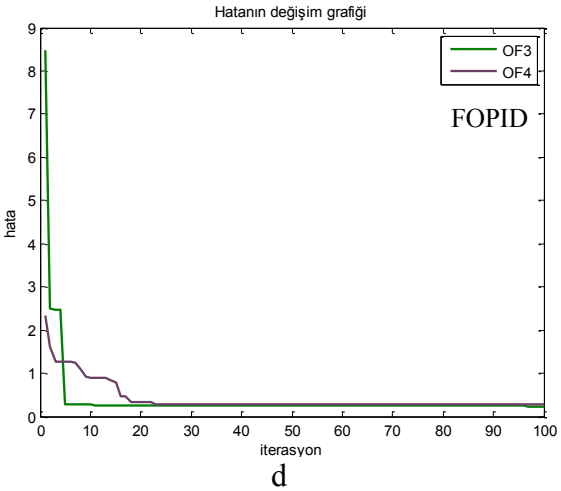
a



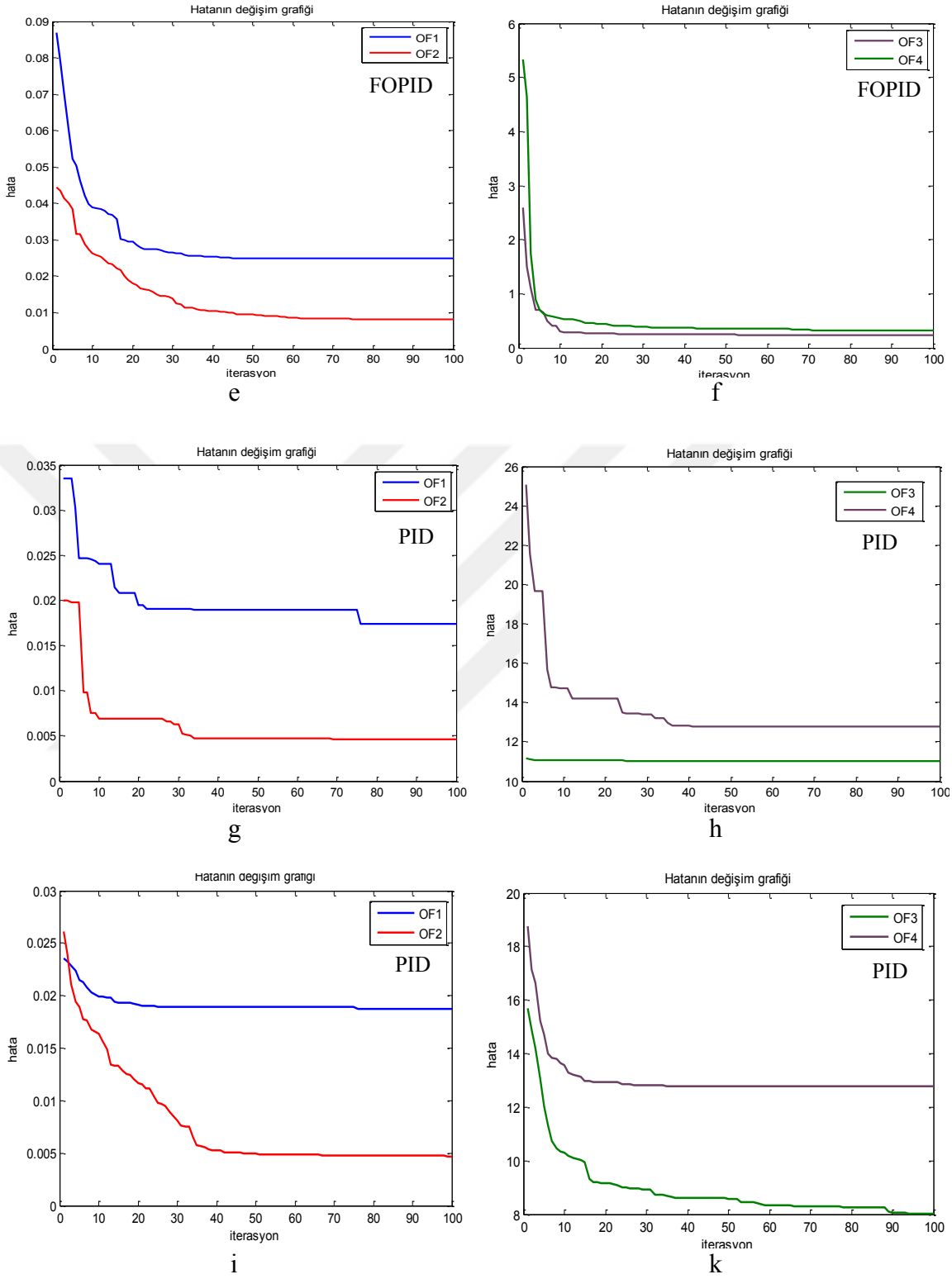
b



c



d



Şekil 4.23. T_9 süreci için DG algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

Tablo 4.42'de T_9 süreci için PSO algoritması kullanılarak elde edilen FOPID ve PID denetleyici yapılarına ait parametre değerlerine yer verilmiştir. Tablo 4.43'te ise bu

denetleyici yapılarının uygulanması ile elde edilen basamak cevapları yer almaktadır. Tabloya göre FOPID denetleyici yapısının OF_2 amaç fonksiyonu haricinde PID denetleyiciye göre daha başarılı performans sergilediği söylenebilir. Aşma miktarının sadece FOPID denetleyicide yok edildiği tabloda dikkat çeken bir diğer unsurdur. Ayrıca FOPID denetleyici yapısında hesaplanan bütün OF değerleri birin altında iken, PID denetleyicide OF değerleri sekizin üzerinde çıkmıştır. İki denetleyici yapısı arasındaki performans farkı basamak cevabı ve hata değişim grafiklerinin sunulduğu Şekil 4.24'ten görülmektedir.

Tablo 4.42. T_9 süreci için PSO algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

Kontr.	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	78.249	97.509	75.258	1.3750	1.0566
		OF_2	48.7005	63.2144	52.7046	1.2140	1.0535
		OF_3	60.4843	7.7187	74.980	1.4624	1.3953
		OF_4	87.6656	70.5684	63.9767	1.4890	1.1355
PID	Tez Çalışması	OF_1	26.2008	89.358	80.366	-	-
		OF_2	14.0483	86.547	61.822	-	-
		OF_3	0.0812	10.1857	8.3751	-	-
		OF_4	0.0002	19.2910	27.4135	-	-

Tablo 4.43. T_9 sürecine ait PSO algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

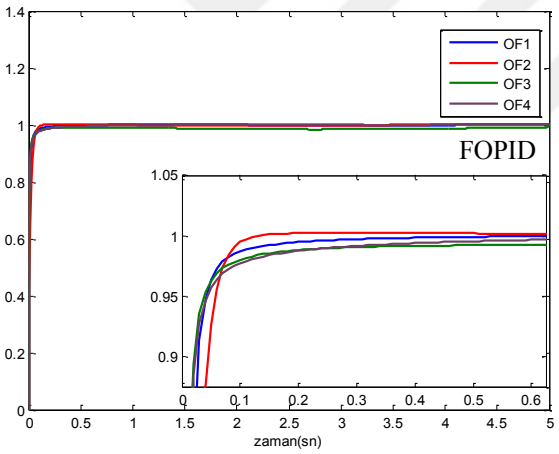
Kont.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.0064	-	0.4777	0.0251	0.0405	0.0705	0.0014
		OF_2	-	0.0151	0.6952	0.0426	0.0586	0.0836	0.0021
		OF_3	0.0066	-	0.2371	0.0187	0.0343	0	0.0001
		OF_4		0.0042	0.2620	0.0224	0.0440	0	0.0021
PID	Tez Çalışması	OF_1	0.0136	-	15.3671	0.0424	0.0525	4.9896	0.0025
		OF_2	-	0.0060	16.5915	0.0505	0.1320	5.2905	0.0030
		OF_3	0.0786	-	10.2218	0.2407	0.3045	2.6377	0.0077
		OF_4	-	0.1944	8.8825	0.0978	0.1236	2.1070	0.0003

T_9 süreci için PSO algoritmasına dayalı FOPID ve PID denetleyici tasarımında elde edilen basamak cevapları ve hata değişim grafikleri Şekil 4.24'de verilmiştir. Şekildeki c,d ve g,h grafikleri minimum hata için elde edilen değişimleri, e,f ve i,k grafikleri ise ortalama hataya ait değişimleri göstermektedir. Şekil 4.24.a'da yer alan basamak cevaplarının oldukça yakın olduğu görülmektedir. OF_1 ve OF_2 amaç fonksiyonları kullanıldığında çok az da olsa aşma miktarının olduğu görülmektedir. Şekil 4.24.b'ye göre PID denetleyici tasarımında kullanılan bütün fonksiyonlarda aşmanın meydana

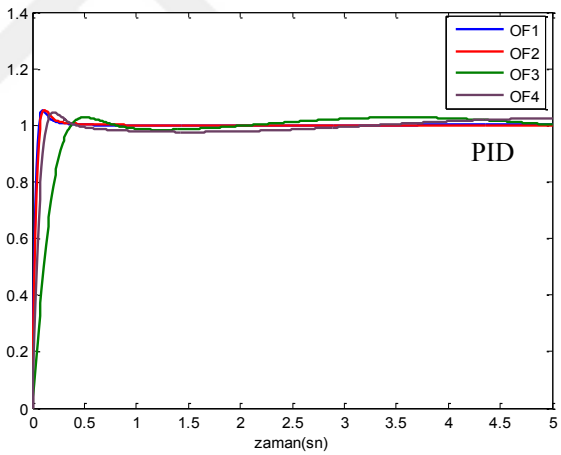
geldiği, OF_3 fonksiyonunun ise diğer fonksiyonlara göre daha yavaş tepki hızı oluşmasına neden olduğu görülmektedir.

FOPID hata gelişim grafiklerinin verildiği Şekil 4.24.c ve d grafiklerinde hata değerinin 10-40. iterasyonlar arasında oldukça azaltıldığı görülmektedir. Ortalama hata değerinin yer aldığı Şekil 4.24.e ve f grafiklerinde ise hata değeri bazı fonksiyonlar için 50-60 iterasyon arasına kadar indirgenmiştir.

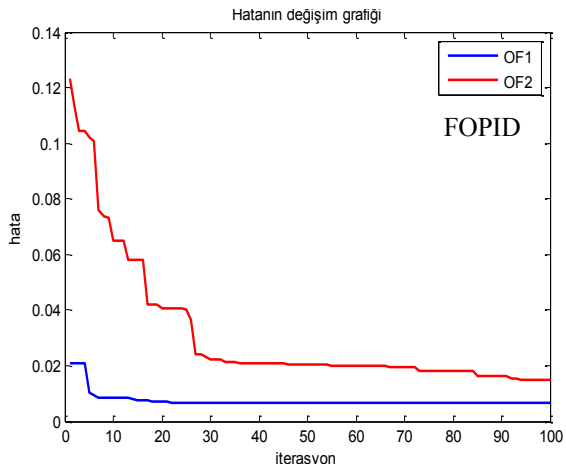
PID denetleyiciye ait hata gelişim grafikleri Şekil 4.24.g,h,i ve k'da verilmiştir. OF_1 ve OF_2 amaç fonksiyonlarında optimizasyon oldukça küçük bir değerden başlamış ve hızlı bir şekilde minimize edilmiştir. OF_3 ve OF_4 amaç fonksiyonlarına ait Şekil 4.24.h grafiğine göre hata değeri ilk 50 iterasyonda oldukça azaltılmıştır. Ortalama hata değeri ise OF_3 dışında bütün amaç fonksiyonlarında neredeyse ilk 40 iterasyonda minimuma indirgenmiştir.



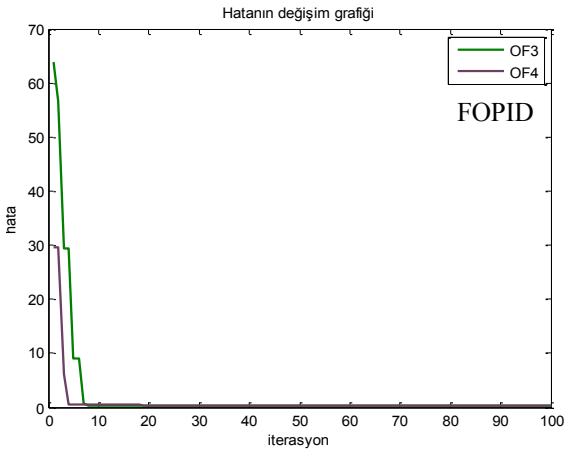
a



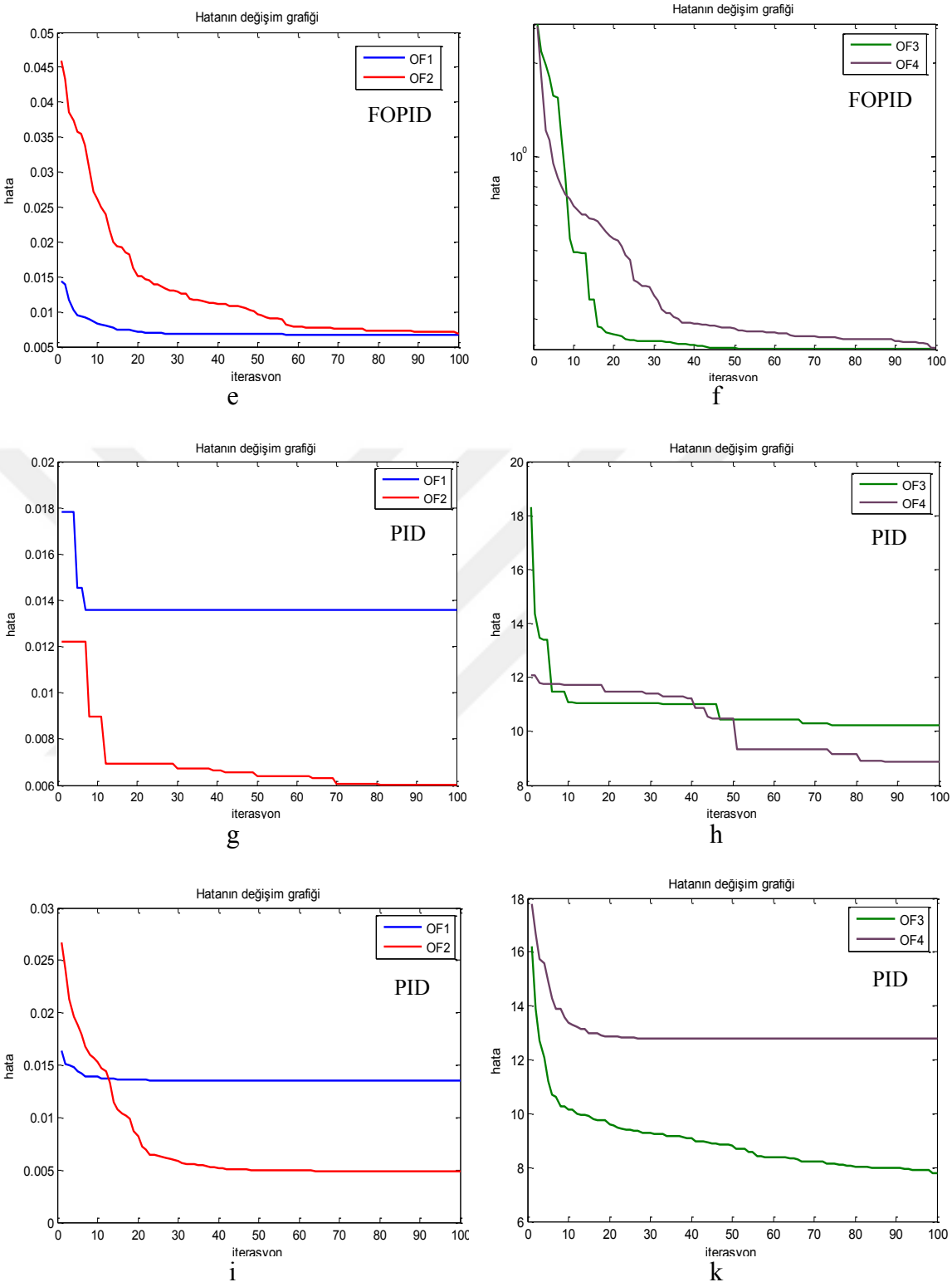
b



c



d



Şekil 4.24. T_9 süreci için PSO algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

Daha önce olduğu gibi T9 süreci için de farklı algoritmalarla elde edilen sonuçların bir arada görülebilmesi bakımından, Tablo 4.44'te OF_3 amaç fonksiyonu dikkate alınarak

hazırlanmış karşılaştırmalı performans tablosu yer almaktadır. Bu tabloya göre, FOPID denetleyici yapısında YAK algoritması 0.1939 OF değeri ile en iyi sonucu ortaya koyarken, PID denetleyicide PSO 10.2218 ile en başarılı sonucu kullanıcıya sunmuştur. Standart sapma değerleri bakımından bütün amaç fonksiyonlarında bu değerin oldukça düşük olarak hesaplandığı izlenmiştir.

Tablo 4.44. T_9 sürecine ait optimum denetleyiciler için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması

Kontr.	Algoritma	ISE	OF_3	Tr(sn)	Ts(sn)	OS%	SS
FOPID	**YAK**	0.0057	**0.1939**	0.0157	0.0267	0.0011	0.0052
	DGA	0.0062	0.2403	0.0198	0.0357	0	0.0007
	PSO	0.0066	0.2371	0.0187	0.0343	0	0.0001
PID	YAK	0.0286	10.7249	0.0880	0.1100	3.2963	0.0046
	DGA	0.0134	11.0249	0.0427	0.0533	3.5412	0.0371
	PSO	0.0786	**10.2218**	0.2407	0.3045	2.6377	0.0077

YAK algoritması kullanarak T_{10} süreci için önerilen denetleyici yapılarına ait parametre değerleri Tablo 4.45'te, hata değeri, aşma miktarı ve oturma süresini gösteren veriler ise Tablo 4.46'da verilmiştir. Literatürde Liu ve arkadaşlarının [117] yapmış olduğu çalışmada elde edilen parametreler ve performans karakteristiklerini gösteren veriler de yine bu tablolarda verilmiştir. Tez çalışmasında FOPID denetleyici için OF_1 amaç fonksiyonu kullanarak 0.0176 ISE değeri, OF_2 fonksiyonu ile 0.0509 ITAE değeri elde edilmiştir. OF_3 ve OF_4 ise sırasıyla 3.1586 ve 2.6303 olarak elde edilmiştir. Literatürde Liu ve arkadaşlarına ait çalışmada [117] ISE 0.3206, ITAE 0.3619, OF_3 37.6299 ve OF_4 ise 38.0431 olarak elde edilmiştir. Bütün amaç fonksiyonlarında elde edilen sonuçların literatürde yer alan çalışmaya göre daha başarılı olduğu açıkça görülmektedir.

Tablo 4.45. T_{10} süreci için YAK algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

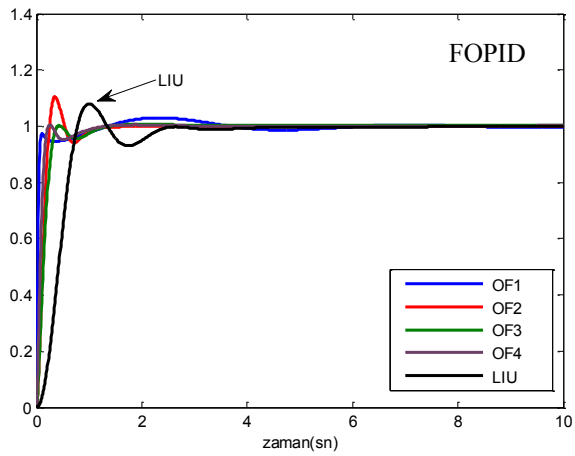
Kont.	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	72.445	89.754	75.458	0.511	1.458
		OF_2	61.931	5.5524	24.445	1.174	1.057
		OF_3	47.158	4.0555	15.783	1.297	1.165
		OF_4	82.444	5.435	29.462	1.319	1.256
	Liu ve ark. [117]		10.000	2.3088	8.981	0.988	0.429
PID	Tez Çalışması	OF_1	89.754	23.102	75.458	-	-
		OF_2	39.754	6.2106	18.819	-	-
		OF_3	25.402	2.9269	10.091	-	-
		OF_4	25.545	3.4027	10.105	-	-
	Liu ve ark.[117]		3.9474	0.6679	1.517	-	-

Tablo 4.46. T_{10} sürecine ait YAK algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

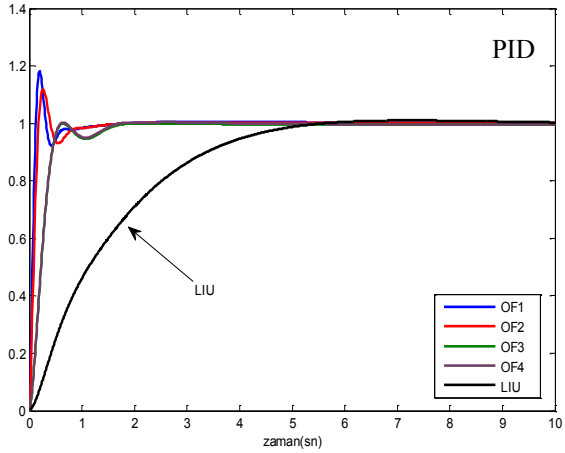
Kont.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.0176	-	12.2476	0.0539	0.4299	3.3071	0.0020
		OF_2	-	0.0509	35.5942	0.1749	0.8108	10.344	0.0031
		OF_3	0.0936	-	3.1586	0.2425	0.3175	0.2116	0.0044
		OF_4	-	0.0694	2.6303	0.1396	0.1796	0.3460	0.0069
	Liu ve ark. [117]		0.3206	0.3619	37.6299(OF_3) 38.0431(OF_4)	0.5017	2.0047	8.1334	-
PID	Tez Çalışması	OF_1	0.0450	-	57.1244	0.0907	0.5200	18.015	0.0006
		OF_2	-	0.0950	28.3056	0.2255	0.9889	7.4704	0.0008
		OF_3	0.1633	-	5.3049	0.3618	0.4831	0.4186	0.0061
		OF_4	-	0.1627	6.2724	0.3610	0.4817	0.6313	0.0085
	Liu ve ark. [117]		0.8284	1.8779	27.6388(OF_3) 38.1336(OF_4)	3.0108	3.8710	0	-

PID denetleyici kullanılarak elde edilen sonuçlarda yine tez çalışmasında elde edilen sonuçların literatürdeki çalışmaya göre oldukça başarılı olduğu Tablo 4.46'da görülmektedir. FOPID denetleyici yapısının PID denetleyiciye göre üstünlüğü bu çalışmada da görülmektedir. Ayrıca OF_3 ve OF_4 amaç fonksiyonlarının aşma miktarı ve yükselme süresinin düşürülmesindeki etkili performansı da bu örnekte açıkça görülmektedir.

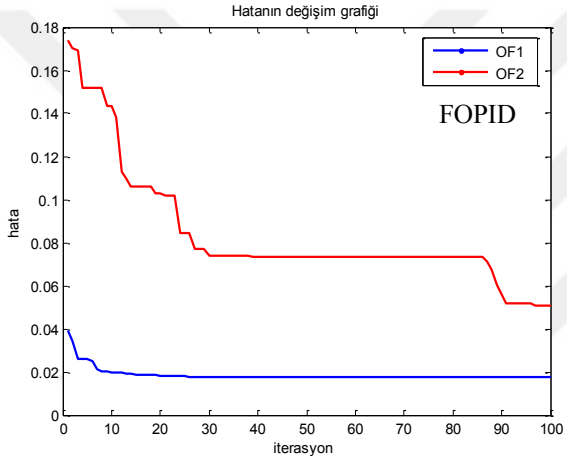
Şekil 4.25 T_{10} süreci için YAK algoritmasına dayalı olarak elde edilen denetleyici çıkışlarını göstermektedir. FOPID denetleyiciye ait basamak cevaplarının yer aldığı Şekil 4.25.a grafiğinde literatürde yapılan çalışma ile tez çalışmasında elde edilen sonuçların performans farkı görülmektedir. Tez çalışmasında bütün amaç fonksiyonları kullanılarak elde edilen basamak cevaplarının tepki hızının literatürdeki çalışmalara göre daha iyi olduğu görülmektedir. OF_2 fonksiyonu dışında diğer bütün fonksiyonlarda aşma yok denecek kadar az olduğu yine aynı grafikte görülmektedir. PID denetleyiciye ait Şekil 4.25.b grafiğinde ise OF_4 dışında diğer fonksiyonların aşma miktarlarının yüksek oluşu göze çarpmaktadır. Ancak bütün amaç fonksiyonları ile elde edilen basamak cevaplarının literatürde Liu ve arkadaşlarının [117] çalışmasına göre oldukça hızlı olduğu görülmektedir. Şekildeki c,d ve g,h grafikleri minimum hata için elde edilen değişimi gösterirken, e,f ve i,k grafikleri ise ortalama hataya ait değişimi sunmaktadır.


FOPID
LIU
OF1
OF2
OF3
OF4
LIU
zaman(sn)

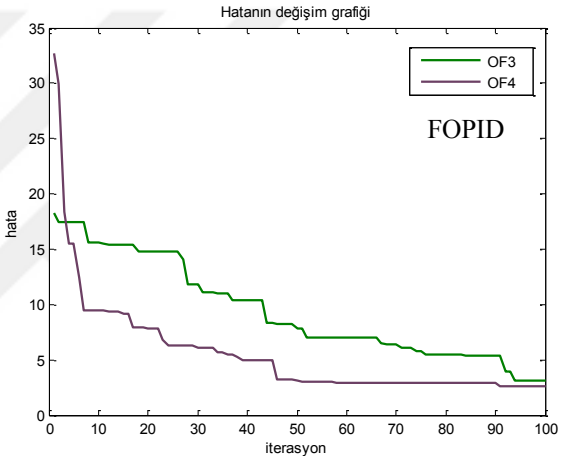
a


PID
LIU
OF1
OF2
OF3
OF4
LIU
zaman(sn)

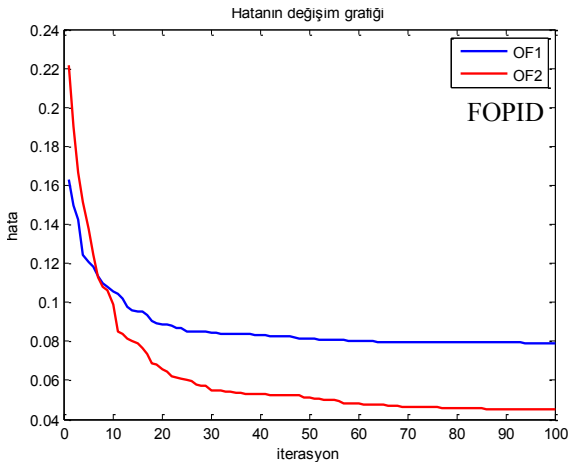
b


Hatanın değişim grafiği
FOPID
OF1
OF2
hata
iterasyon

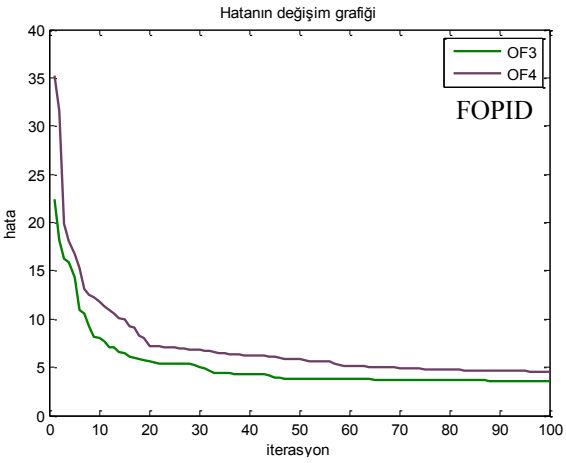
c


Hatanın değişim grafiği
FOPID
OF3
OF4
hata
iterasyon

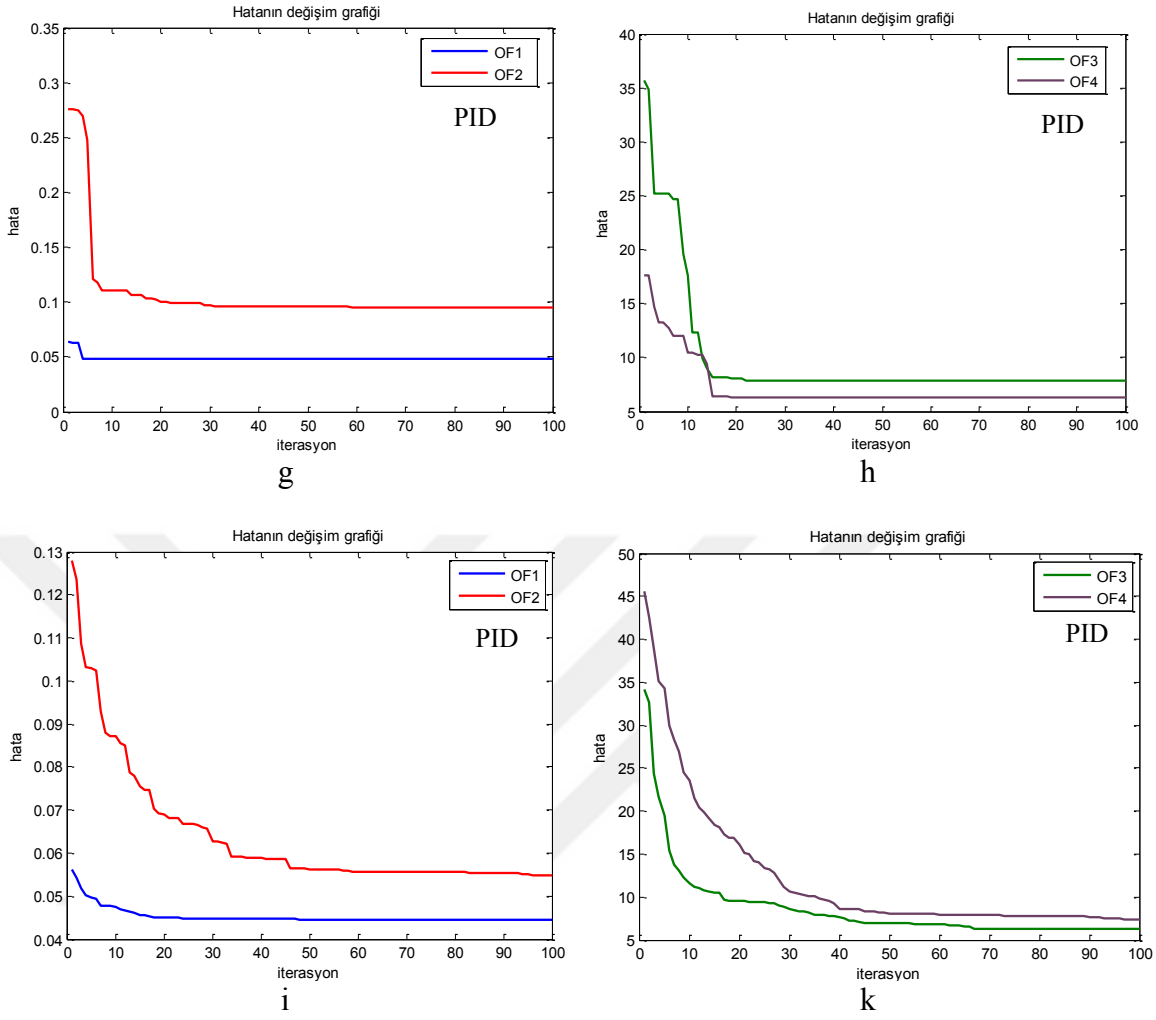
d


Hatanın değişim grafiği
FOPID
OF1
OF2
hata
iterasyon

e


Hatanın değişim grafiği
FOPID
OF3
OF4
hata
iterasyon

f



Şekil 4.25. T_{10} süreci için YAK algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

FOPID denetleyici kullanılarak elde edilen hata gelişim grafiklerinde hatanın OF_1 dışında kalan incelemelerde son iterasyona kadar indirgendiği, PID denetleyici de ise ilk 20 iterasyonda oldukça azaltıldığı görülmektedir. Ortalama hata değerlerinde ise her iki denetleyicide de hata değeri ilk 50 iterasyonda minimize edilmiş, bu iterasyondan sonra çok küçük değişiklikler meydana gelmiştir.

T_{10} süreci için denetleyici tasarımının yapıldığı diğer algoritma olan DGA'ya ait sonuçlar Tablo 4.47 ve Tablo 4.48'de verilmiştir.

Tablo 4.48'e göre FOPID denetleyici yapısının bütün amaç fonksiyonlarında PID denetleyiciye göre başarılı olduğu ve daha arzu edilir sonuçlar sunduğu söylenebilir. Özellikle OF_3 ve OF_4 performans indisleri dikkate alındığında bu farklılık daha belirgin

olarak görülmektedir. Bu iki indisin aşma miktarını diğer iki performans indisine göre oldukça düşürdüğü gözden kaçırılmaması gereken bir durumdur. Her iki denetleyici yapısında kullanılan amaç fonksiyonları için elde edilen standart sapma değerlerinin düşük olduğu yine tablodan çıkarılabilecek başka bir sonuçtur.

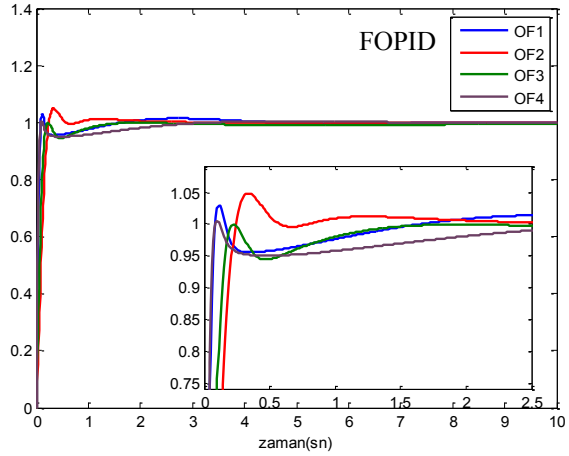
Tablo 4.47. T_{10} süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

Kontrolör	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	67.4770	71.653	93.824	1.2890	0.506
		OF_2	91.5470	11.2127	21.2440	1.2494	1.0837
		OF_3	85.7480	1.1495	34.0156	1.2738	1.4230
		OF_4	87.5650	6.0306	97.0783	1.3705	1.2302
PID	Tez Çalışması	OF_1	89.916	28.0310	89.916	-	-
		OF_2	74.367	11.1197	41.6378	-	-
		OF_3	26.3765	2.3518	16.923	-	-
		OF_4	23.4719	3.1061	9.1386	-	-

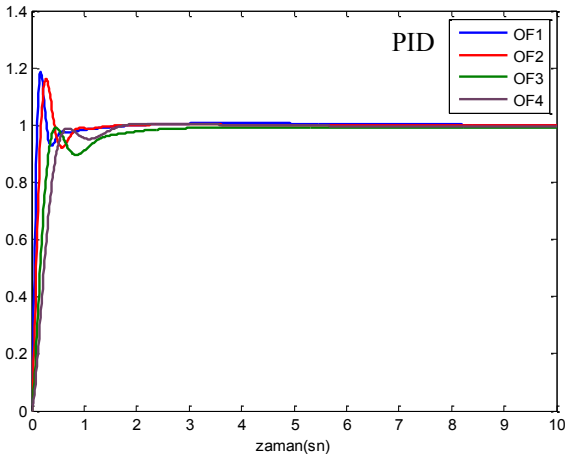
Tablo 4.48. T_{10} sürecine DG algoritmasına dayalı ait optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

Kont.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.0210	-	9.7810	0.0570	0.0711	3.0720	0.0001
		OF_2	-	0.0344	15.7573	0.1657	0.2094	4.7888	0.0002
		OF_3	0.0441	-	2.8286	0.1218	0.1563	0.5355	0.0044
		OF_4	-	0.1459	2.9254	0.0502	0.0641	0.3818	0.0026
PID	Tez Çalışması	OF_1	0.0419	-	58.0892	0.0794	0.4585	18.4592	0.0001
		OF_2	-	0.0563	52.2780	0.1298	0.7002	16.0715	0.0002
		OF_3	0.1257	-	7.3552	0.2689	1.2197	0	0.0039
		OF_4	-	0.2224	6.9055	0.3954	0.5329	0.6722	0.0024

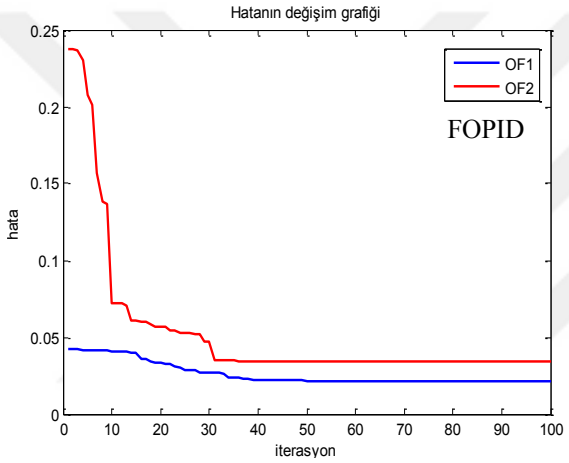
T10 süreci için DGA ile elde edilen denetleyicilere ait basamak cevabı ve hatanın değişimini sunan Şekil 4.26.a grafiğinde FOPID, 4.26.b grafiğinde ise PID denetleyici için elde edilen basamak cevaplarına yer verilmiştir. Her iki denetleyicide de amaç fonksiyonları kullanımında birbirine yakın çıkışlar elde edildiği görülmektedir. Ancak aşma miktarının Şekil 4.26.a'da 4.26.b daha düşük olduğu söylenebilir. Ayrıca OF_1 ve OF_2 amaç fonksiyonları kullanıldığında meydana gelen yüksek aşma miktarı yine 4.26.a ve b grafiğinden açıkça görülmektedir.


FOPID
OF1
OF2
OF3
OF4
zaman(sn)

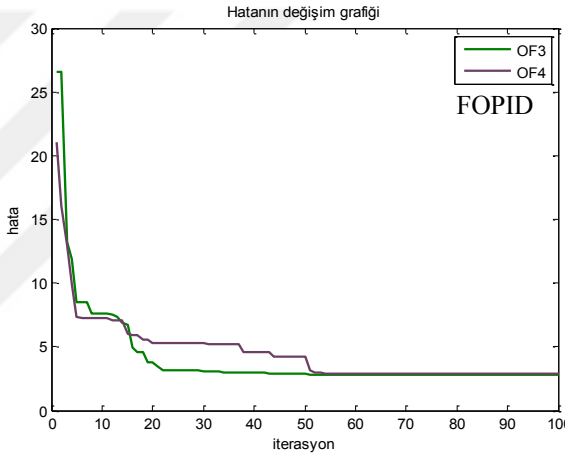
a


PID
OF1
OF2
OF3
OF4
zaman(sn)

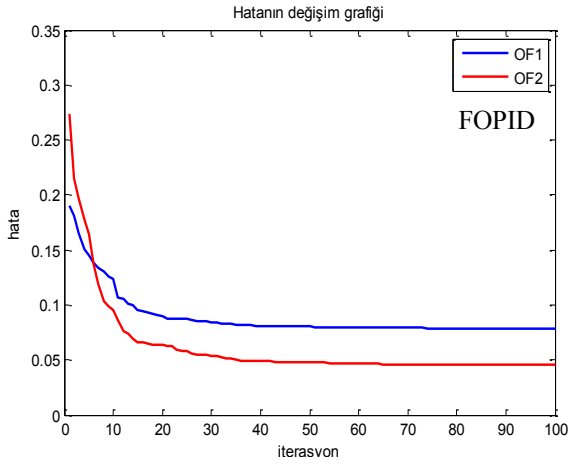
b


Hatanın değişim grafiği
FOPID
OF1
OF2
hata
iterasyon

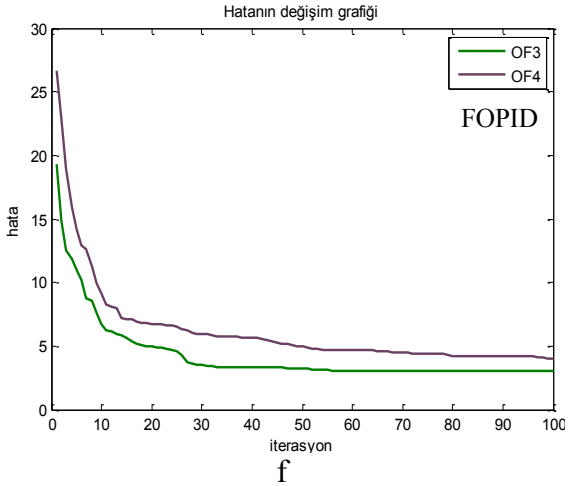
c


Hatanın değişim grafiği
FOPID
OF3
OF4
hata
iterasyon

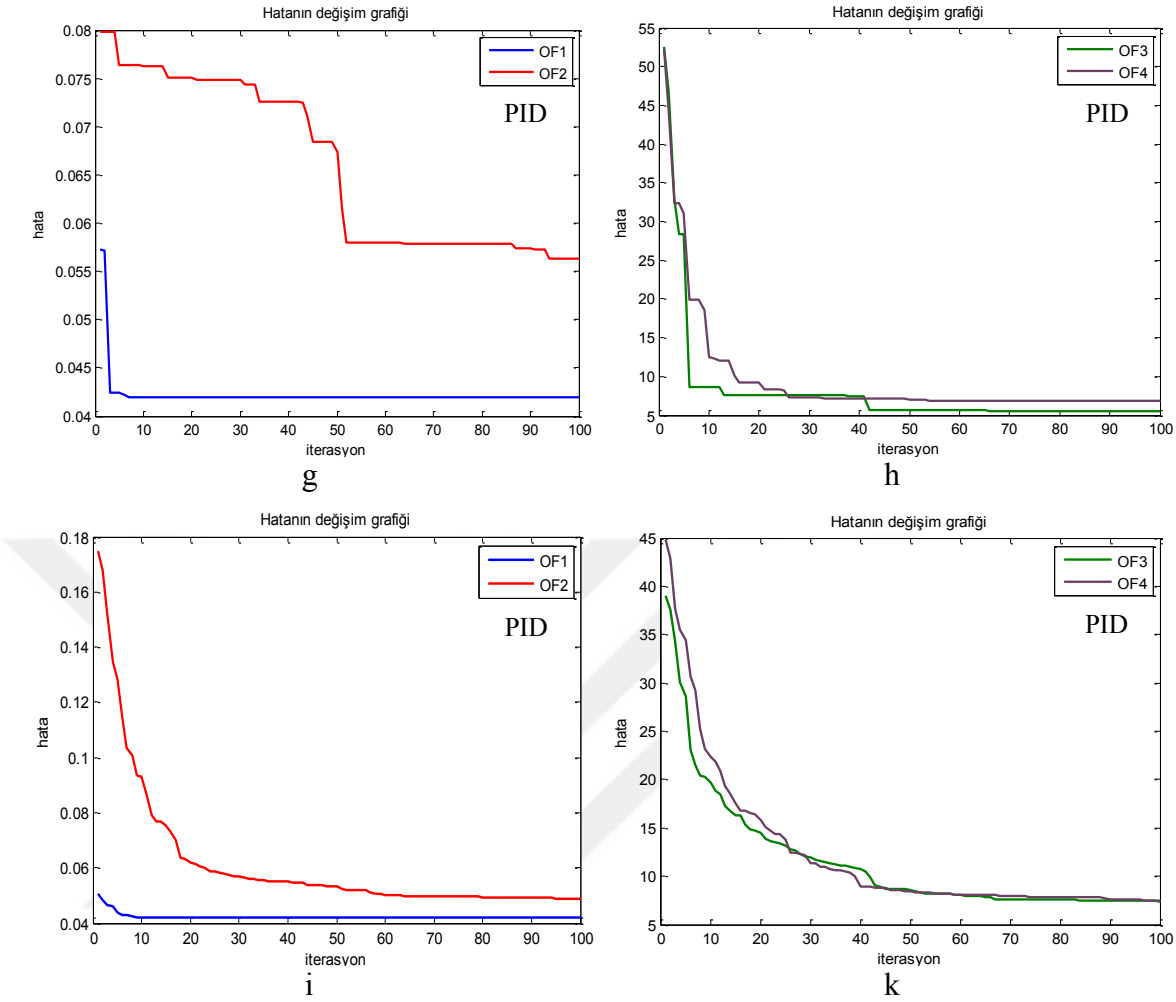
d


Hatanın değişim grafiği
FOPID
OF1
OF2
hata
iterasyon

e


Hatanın değişim grafiği
FOPID
OF3
OF4
hata
iterasyon

f



Şekil 4.26. T_{10} süreci için DG algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

Hata gelişim grafiklerinin yer aldığı şekillerde hata değeri OF_1 ve OF_2 için ilk 30 iterasyonda, OF_3 ve OF_4 için ilk 50 iterasyonda indirgendiği görülmektedir. DG algoritmasının hata değerini ilk 5-10 iterasyon arasında çok hızlı bir şekilde indirgediği Şekil 4.26.d grafiği algoritmanın gücünü göstermektedir. PID denetleyici için elde edilen hata gelişim grafiklerinde ise hata değerinin FOPID denetleyiciye nispeten daha yüksek değerden başladığı ancak benzer şekilde hızlıca indirgendiği Şekil 4.26.g ve h grafiklerinden anlaşılmaktadır. Ayrıca iki denetleyiciye ait grafiklere göre, OF_1 amaç fonksiyonunda hatanın minimuma oldukça yakın değerden başladığı ve çok az iterasyon sayısında azaltıldığı da açıkça görülmektedir. Ortalama hata değerlerinde ise benzer sonuçların elde edildiği söylenebilir. Hata değeri bütün amaç fonksiyonlarında ilk 40 iterasyonda minimize edilmiş, daha sonraki iterasyonlarda oldukça küçük değişiklikler meydana gelmiştir.

T_{10} süreci için önerilen denetleyici tasarımlarında son olarak PSO algoritması kullanılmıştır. Bu algoritma ile elde edilen denetleyici yapılarına ait parametreler Tablo 4.49'da, temel performans karakteristikleri ise Tablo 4.50'de sunulmuştur. Tablolardaki değerlerden, FOPID denetleyici için önerilen OF_2 amaç fonksiyonunda elde edilen %27.4150 aşma miktarı oldukça dikkat çekmektedir. Benzer bir sonuç %25.4342 ile PID denetleyici için önerilen aynı fonksiyonda da görülmektedir. Genel olarak OF değerlerine göre FOPID denetleyici yapısının PID'ye göre daha üstün performansa sahip olduğu, standart sapma değerlerinde ise PID denetleyicinin daha başarılı sonuçlar sunduğu tablodan söylenebilir.

Şekil 4.27'de T_{10} süreci için PSO algoritması kullanılarak elde edilen basamak cevapları ve hata gelişim grafiklerine yer verilmiştir. Şekildeki c,d ve g,h grafikleri minimum hata için elde edilen değişimi gösterirken, e,f ve i,k grafikleri ise ortalama hataya ait değişimi sunmaktadır. Yukarıda bahsedilen OF_2 amaç fonksiyonuna ait aşma değerinin yüksek oluşu Şekil 4.27.a ve b grafiklerinde açıkça görülmektedir. T_{10} süreci için FOPID denetleyicinin PID denetleyiciye göre daha hızlı tepki cevabına sahip olduğu yine aynı grafiklerden görülmektedir.

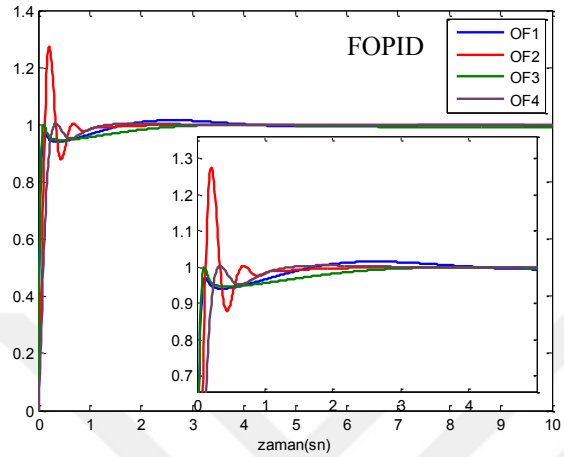
Tablo 4.49. T_{10} süreci için PSO algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

Kontrolör	**Yöntem**		**Kp**	**Ki**	**Kd**	**λ**	**μ**
FOPID	Tez Çalışması	OF_1	91.5470	28.500	66.668	1.444	0.577
		OF_2	91.222	19.7909	88.648	0.8842	0.991
		OF_3	80.3867	9.4502	87.7599	1.3708	0.510
		OF_4	61.2631	4.2766	21.2398	1.2075	1.2470
PID	Tez Çalışması	OF_1	95.547	9.998	94.147	-	-
		OF_2	86.522	69.358	26.2359	-	-
		OF_3	26.8921	4.2926	19.854	-	-
		OF_4	26.6333	3.9171	10.5209	-	-

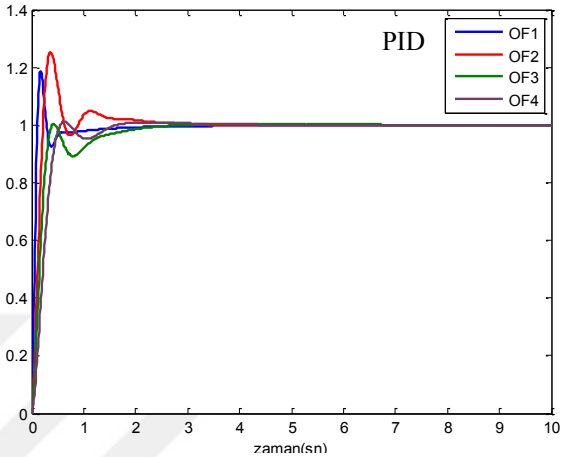
Tablo 4.50. T_{10} sürecine ait PSO algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

Kont.	**Yöntem**		**ISE**	**ITAE**	**OF**	**Tr(sn)**	**Ts(sn)**	**OS%**	**SS**
FOPID	Tez Çalışması	OF_1	0.0195	-	8.8534	0.0607	0.6217	1.8500	0.2254
		OF_2		0.0505	85.8995	0.0923	0.5571	27.4150	0.2333
		OF_3	0.0199	-	2.7196	0.0529	0.0673	0.7281	0.2154
		OF_4	-	0.0641	2.9685	0.1866	0.2418	0.3729	0.0587
PID	Tez Çalışması	OF_1	0.0423	-	59.1970	0.0794	0.4593	18.8258	0.0001
		OF_2		0.1135	80.3021	0.1565	0.5730	25.4342	0.0002
		OF_3	0.1131	-	8.2885	0.2436	1.2063	0.3755	0.0251
		OF_4	-	0.1519	8.0079	0.3473	0.4615	1.3937	0.0014

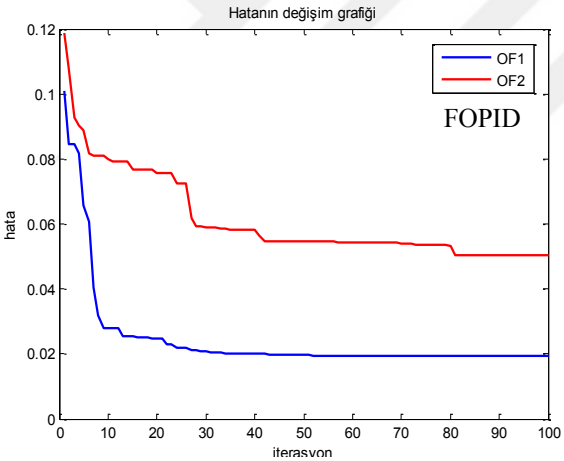
PSO algoritmasına ait hata gelişim grafiklerinde ise OF_2 dışında diğer fonksiyonlarda hızlı bir şekilde hatanın azaltıldığı görülmektedir. OF_1 fonksiyonunda diğer algoritmalarda olduğu gibi hatanın oldukça küçük bir değerden başladığı da yine bu grafiklerden açıkça görülmektedir.



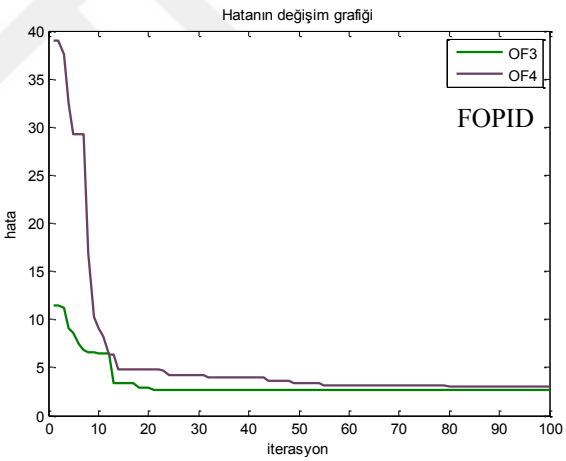
a



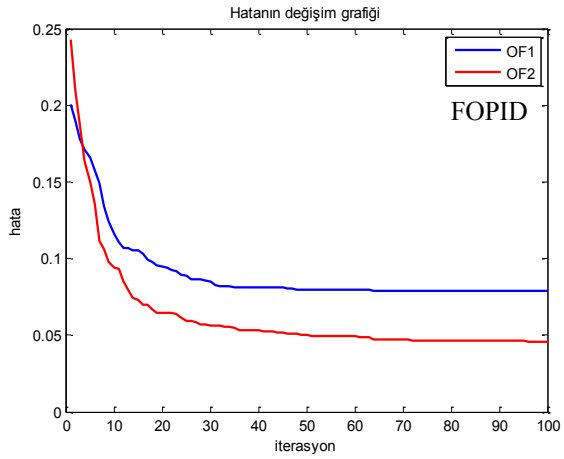
b



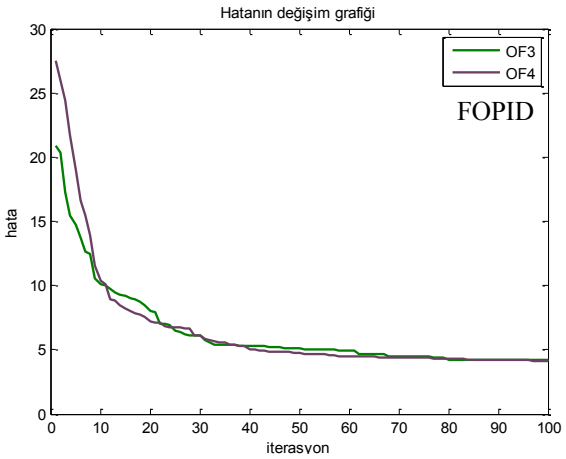
c



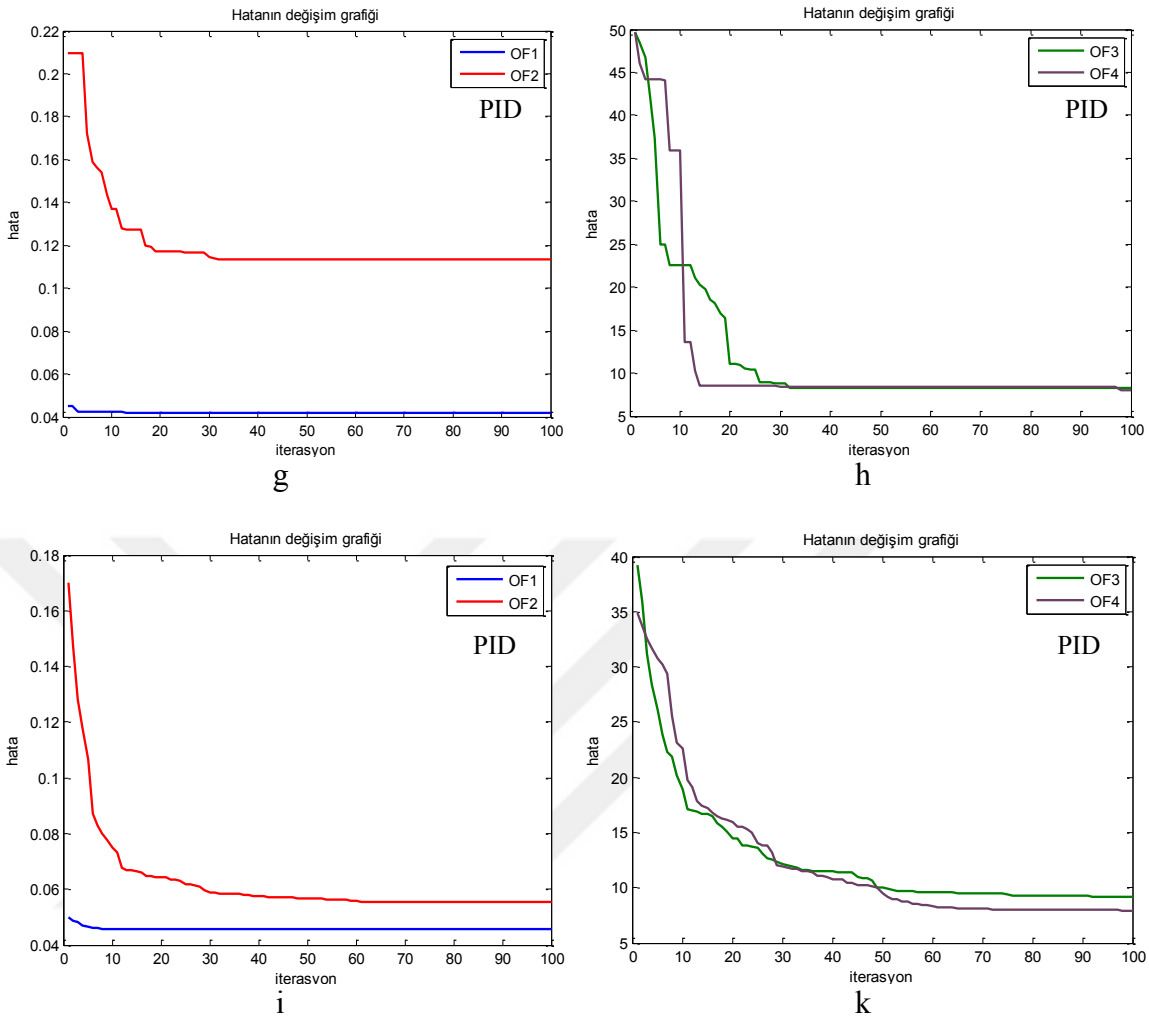
d



e



f



Şekil 4.27. T_{10} süreci için PSO algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

Algoritmalar arasındaki farklılıkların daha iyi görülebilmesi amacıyla, T_{10} süreci için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması Tablo 4.51'de sunulmuştur. Buna göre, FOPID denetleyicide 0.0199, PID denetleyicide ise 0.1131 değeri ile PSO algoritması en iyi sonuçları elde etmiştir. Standart sapma değerlerinde YAK algoritmasının üstünlüğü yine tablodan açıkça görülmektedir. Ayrıca aşma değerinin ise sadece DG algoritması ile yapılan çalışmada sıfırlandığı ifade edilebilir.

Tablo 4.51. T_{10} sürecine ait optimum denetleyiciler için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması

Kontr.	Algoritma	ISE	OF_3	Tr(sn)	Ts(sn)	OS	SS
FOPID	YAK	0.0936	3.1586	0.2425	0.3175	0.2116	0.0020
	DGA	0.0441	2.8286	0.1218	0.1563	0.5355	0.0044
	PSO	0.0199	**2.7196**	0.0529	0.0673	0.7281	0.2154
PID	YAK	0.1633	5.3049	0.3618	0.4831	0.4186	0.0006
	DGA	0.1257	7.3552	0.2689	1.2197	0	0.0039
	PSO	0.1131	**8.2885**	0.2436	1.2063	0.3755	0.0251

Kontrolör tasarımı için ele alınan son örnek T_{11} sürecidir. Bu sürece ait yapılan çalışmalar sonucunda elde edilen kontrolör yapılarına ait parametre değerleri Tablo 4.52'de, performans karakteristikleri Tablo 4.53'te verilmiştir. Literatürde yer alan Itik ve arkadaşlarının [120] yapmış olduğu çalışmadan ele alınan bu süreç için FOPID denetleyici yapısının oldukça başarılı performans sergilediği tablodan görülmektedir. Ayrıca FOPID denetleyici yapısı ile elde edilen sonuçlar literatürde yer alan örneğe göre daha başarılı olduğu da yine Tablo 4.53'ten anlaşılmaktadır. Tez çalışmasında ISE değeri 0.0051 olarak elde edilirken, literatürde bu değer 0.3162 olarak elde edilmiştir. Yükselme ve oturma süresi ile aşma miktarı da literatürdeki çalışmadan oldukça düşük elde edilmiştir. Sırası ile bu değerler literatürde 1.09sn, 2.36sn ve %5.45 olarak elde edilirken, tez çalışmasında YAK algoritması kullanarak OF_3 amaç fonksiyonu ile 0.0086sn, 0.0164sn ve %0 olarak elde edilmiştir.

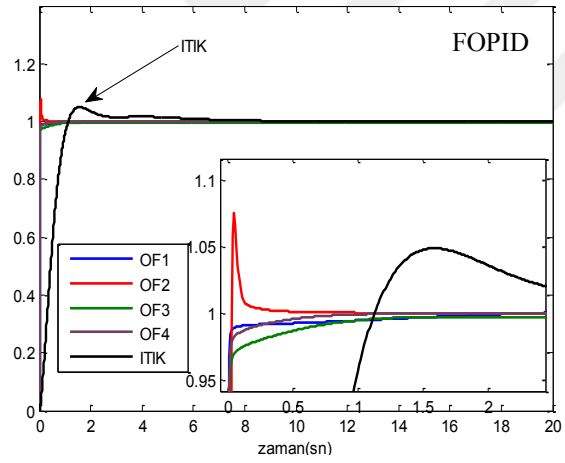
Tablo 4.52. T_{11} süreci için YAK algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

Kont.	Method		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	20.225	30.991	38.441	0.579	1.211
		OF_2	70.365	80.561	20.539	0.883	0.512
		OF_3	32.022	0.195	10.541	0.511	1.317
		OF_4	69.102	14.489	20.541	0.958	1.231
	Itik ve ark.[120]		0.68	0.328	0.0091	0.908	0.755
PID	Tez Çalışması	OF_1	1.991	2.3679	1.111	-	-
		OF_2	5.872	12.548	0.4946	-	-
		OF_3	1.874	0.475	0.2308	-	-
		OF_4	5.876	1.0726	0.949	-	-
	Itik ve ark. [120]		0.599	0.332	0.0018	-	-

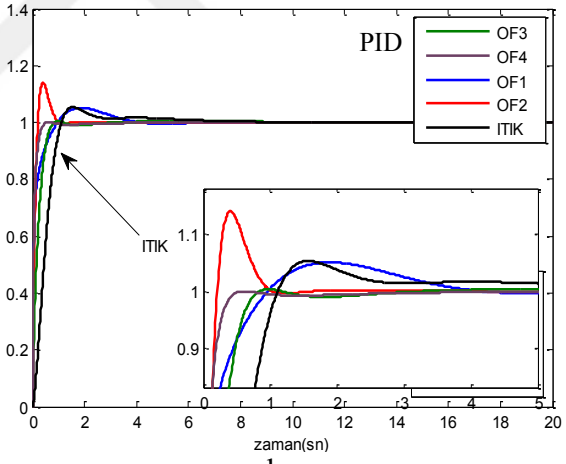
Tablo 4.53. T_{11} sürecine ait YAK algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

Kont.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.0051	-	0.6478	0.0083	0.0098	0.1826	0.0002
		OF_2	-	0.0038	23.3369	0.0236	0.0704	7.6328	0.0032
		OF_3	0.0054	-	0.1364	0.0086	0.0164	0	0.0008
		OF_4	-	0.0037	0.0956	0.0085	0.0117	0.0001	0.0053
	Itik ve ark. [120]		0.3162	0.6077	31.3120(OF_3) 34.2270(OF_4)	1.09	2.36	5.45	-
PID	Tez Çalışması	OF_1	0.04154	-	25.9367	0.4422	2.0518	5.0874	0.0891
		OF_2	-	0.0484	46.2572	0.1540	0.7265	14.046	0.0624
		OF_3	0.1052	-	4.7978	0.4609	0.5803	0.2814	0.0999
		OF_4	-	0.0917	2.2541	0.1866	0.2660	0.0023	0.0784
	Itik ve ark. [120]		0.3212	0.6094	58.7520(OF_3) 61.6340(OF_4)	1.172	5.96	8.58	-

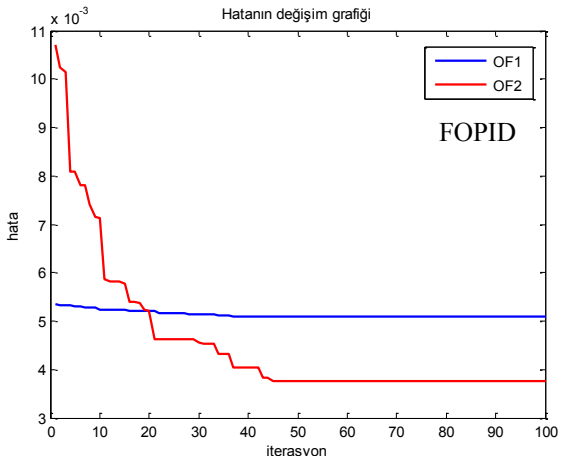
Şekil 4.28'de T_{11} sürecine ait YAK algoritması kullanarak elde edilen denetleyici yapılarına ait çıkış grafikleri yer almaktadır. FOPID denetleyiciye ait basamak cevaplarının yer aldığı Şekil 4.28.a'da bütün fonksiyonların birbirine oldukça yakın performans sergilenmesine neden olduğu görülmektedir. OF_2 dışında kalan amaç fonksiyonlarında aşmanın neredeyse sıfıra yakın olduğu yine bu grafikten anlaşılmaktadır. Ayrıca Itik ve arkadaşlarına [120] ait çalışmada aşma miktarının olduğu ve oturma süresinin tez çalışmasındaki sonuçlara göre oldukça yüksek değerde gerçekleştiği ifade edilebilir. Tez çalışmasında kullanılan amaç fonksiyonlarının kullanımına dayalı denetleyicilerle ulaşılan sistem çıkışlarında tepki hızlarının oldukça iyi olduğu da yine grafikten çıkarılabilecek başka bir sonuçtur. PID denetleyici yapısına ait basamak cevaplarının yer aldığı Şekil 4.28.b grafiğinde sadece OF_4 ile yapılan çalışmada aşmanın sıfıra oldukça yakın olduğu, diğer üç amaç fonksiyonu kullanımında ise belirli bir aşmanın bulunduğu görülmektedir.



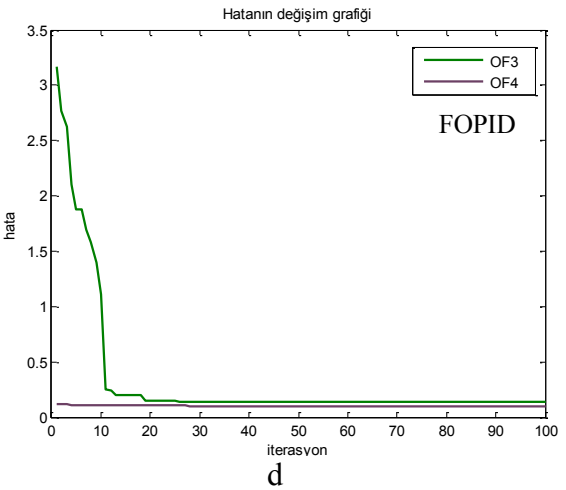
a



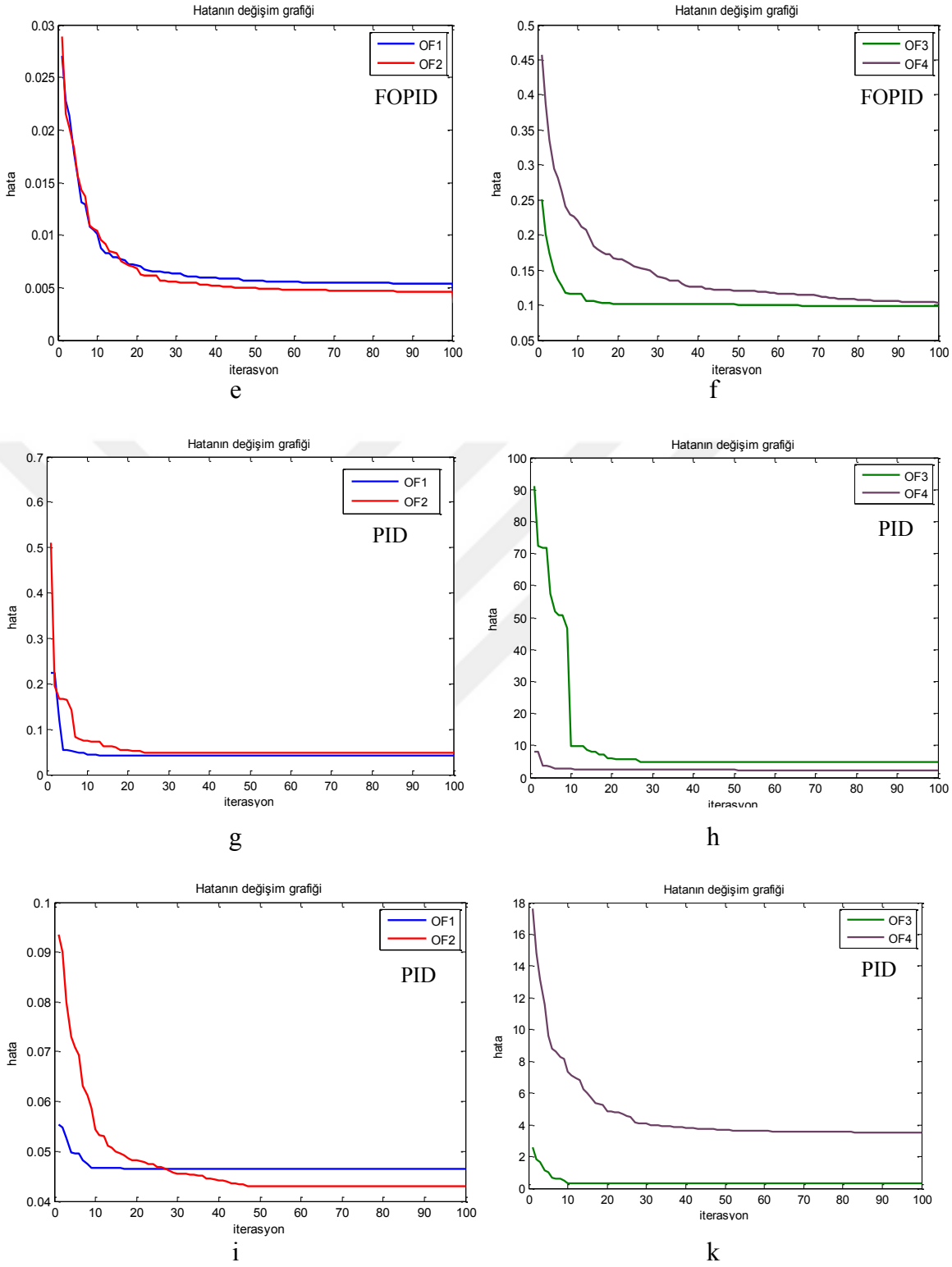
b



c



d



Şekil 4.28. T_{11} süreci için YAK algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

Ayrıca PID denetleyici yapısının FOPID denetleyiciye göre daha yavaş tepki hızının olduğu Şekil 4.28.b grafiğinden anlaşılmaktadır. FOPID denetleyici yapısına ait hata

gelişim grafiklerinin tamamında hata değerinin oldukça düşük değerlerden başladığı görülmektedir. PID denetleyicide ise OF_3 fonksiyonunda hata yüksek değerden başlamış ve hızlı bir şekilde düşürülmüştür.

DG algoritması kullanarak T_{11} süreci için tasarlanan denetleyici yapılarına ait parametre değerleri Tablo 4.54'te sunulmuştur. Aynı süreçte dört farklı amaç fonksiyonu için elde edilen hata değeri, aşma miktarı, yükselme ve oturma süreleri gibi performans karakteristikleri ise Tablo 4.55'te verilmiştir. Tabloya göre FOPID denetleyici kullanarak yapılan çalışmalarda düşük amaç fonksiyonu değerleri göze çarpmaktadır. OF_1 0.005, OF_2 0.0055, OF_3 0.0987 ve OF_4 0.1385 olarak elde edilirken PID denetleyicide bu değerler sırasıyla 0.0453, 0.0892, 0.2934 ve 1.8888 olarak elde edilmiştir. Standart sapma değerleri her iki denetleyici yapısı için de oldukça düşük olarak elde edilmiştir. Ayrıca aşma miktarı FOPID denetleyicide bütün amaç fonksiyonlarında hemen hemen sıfır olarak elde edilmiştir.

Tablo 4.54. T_{11} süreci için DG algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

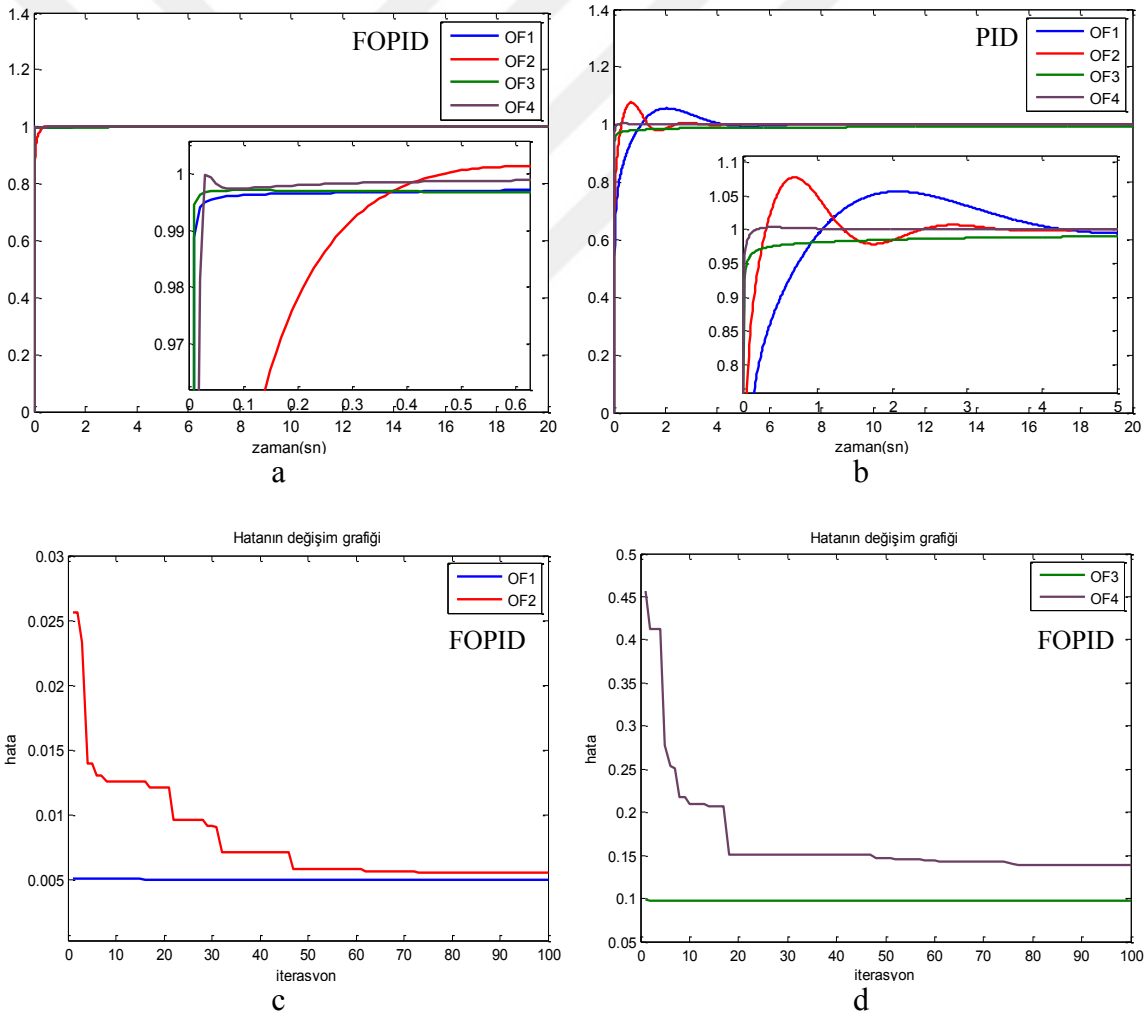
Kontrolör	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	82.245	45.530	86.511	1.2776	0.5985
		OF_2	20.4891	5.3033	3.6340	1.0728	0.9348
		OF_3	91.547	32.230	99.576	1.0916	0.577
		OF_4	82.245	30.0301	62.2824	0.7026	0.8775
PID	Tez Çalışması	OF_1	1.724	2.120	1.100	-	-
		OF_2	5.123	19.962	1.744	-	-
		OF_3	9.211	0.0011	9.9987	-	-
		OF_4	85.401	89.188	13.5470	-	-

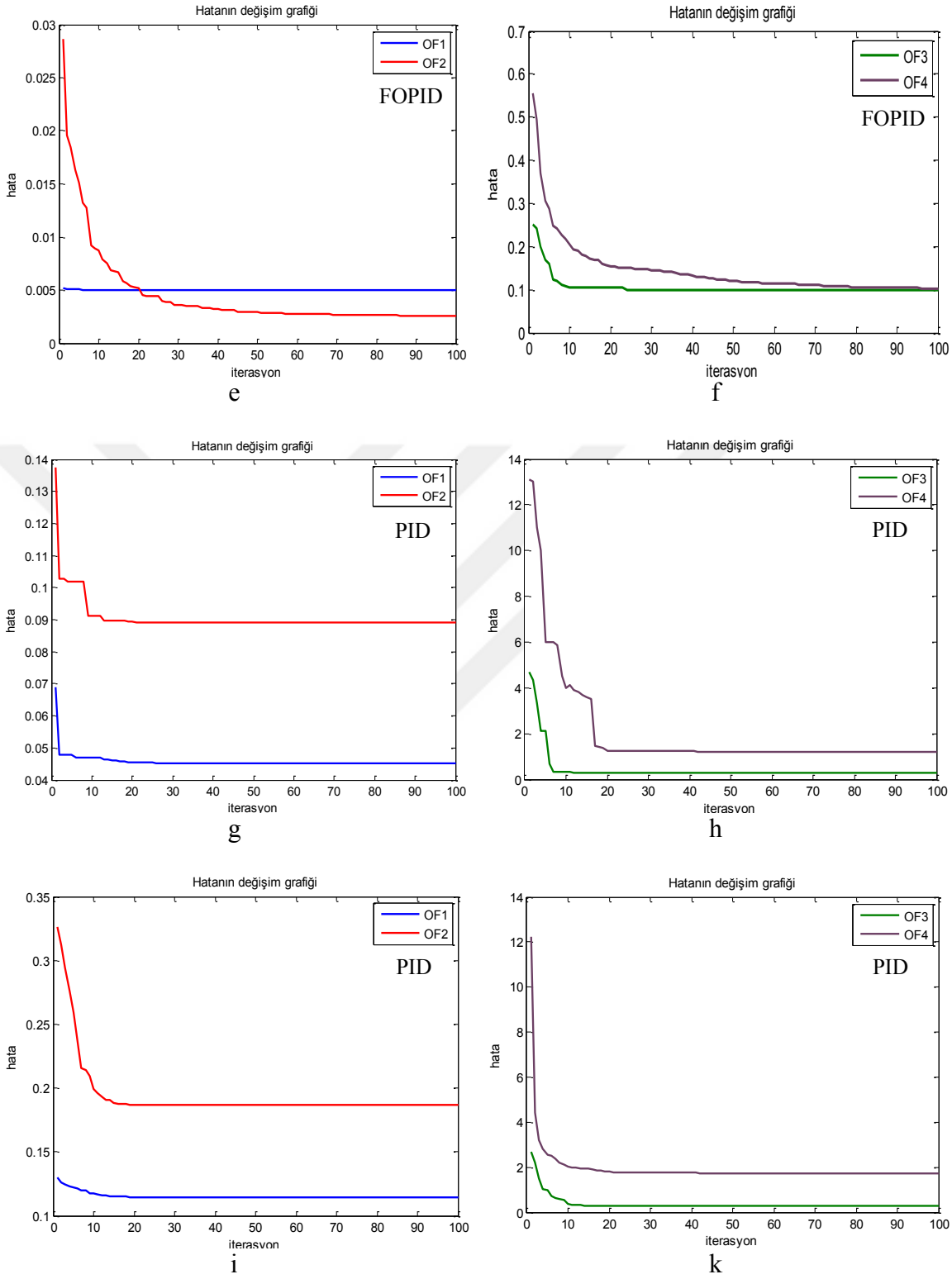
Tablo 4.55. T_{11} sürecine ait DG algoritmasına dayalı optimum denetleyiciler kullanılarak elde edilen temel performans karakteristikleri

Kont.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.0050	-	0.2096	0.0081	0.0096	0.0372	0
		OF_2	-	0.0055	1.0297	0.0498	0.1105	0.1406	0.0003
		OF_3	0.0050	-	0.0987	0.0082	0.0097	0	0.0001
		OF_4	-	0.0040	0.1385	0.0135	0.0180	0.030	0.0001
PID	Tez Çalışması	OF_1	0.0453	-	29.8198	0.5037	2.5124	5.6016	0.0987
		OF_2	-	0.0892	28.9100	0.1477	0.9932	7.6842	0.0058
		OF_3	0.0085	-	0.2934	0.0198	0.0416	0	0.0003
		OF_4	-	0.0050	1.1888	0.0162	0.0260	0.3363	0.0087

T11 süreci için elde edilen basamak cevabı ve hatanın değişimi grafiklerini içeren Şekil 4.29.a grafiğinde FOPID denetleyici yapısına ait birim basamak cevaplarında bütün

amaç fonksiyonlarında birbirine oldukça yakın sonuçlar elde edildiği görülmektedir. Sadece OF_2 fonksiyonunda çok küçük miktarda aşma olduğu, oturma süresinin de bu fonksiyon için daha uzun olduğu söylenebilir. PID denetleyici yapısına ait basamak cevaplarının yer aldığı Şekil 4.29.b grafiğinde ise ISE ve ITAE hata indislerinin kullanıldığı çalışmalarda aşmanın fazla ve oturma süresinin uzun olduğu görülmektedir. FOPID denetleyicide hata gelişim grafiklerine göre OF_1 ve OF_3 kullanımında neredeyse minimum hata değerinden optimizasyona başlandığı görülmektedir. PID denetleyicide de bu iki amaç fonksiyonu için benzer sonuçlar elde edilmiştir. Ortalama hata grafiklerinde ise DGA'nın çok hızlı şekilde hatayı minimize ettiği, algoritmanın bütün farklı çalışmalarında ilk 20 iterasyonda sonuca ulaştığı söylenebilir.





Şekil 4.29. T_{11} süreci için DG algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

Tablo 4.56'da PSO algoritması kullanarak T_{11} süreci için elde edilen denetleyici parametreleri yer almaktadır. Denetleyici tasarımı sonucunda elde edilen sistem cevabına ilişkin sonuçlar Tablo 4.57'de verilmiştir. Elde edilen sonuçların diğer iki algoritmadakine benzer olduğu ve FOPID denetleyici yapısının PID'ye göre daha başarılı olduğu söylenebilir. Aşma miktarı FOPID denetleyici ile yok edilmiş, yükselme süresi ve oturma süreleri de düşürülmüştür. PSO algoritması ile elde edilen denetleyicilerle ulaşılan sistem cevapları ve hatanın değişim grafikleri Şekil 4.30'da sunulmuştur. Şekildeki c,d ve g,h grafikleri minimum hata için elde edilen değişimi gösterirken, e,f ve i,k grafikleri ise ortalama hataya ait değişimi sunmaktadır.

Şekil 4.30'da FOPID denetleyici performansını gösteren (a) grafiğe göre sadece OF_2 fonksiyonunda az miktarda aşma olduğu, PID denetleyiciye ait (b) grafiğinde ise OF_2'nin yanı sıra OF_1 amaç fonksiyonunda da aşma olduğu görülmektedir. En iyi sonuçlara ait hata gelişim grafiklerinde FOPID denetleyici için ilk 20-30 iterasyonda hata oldukça azaltılmış, PID denetleyicide ise bazı amaç fonksiyonları için bu değer 50 iterasyona kadar çıkmıştır.

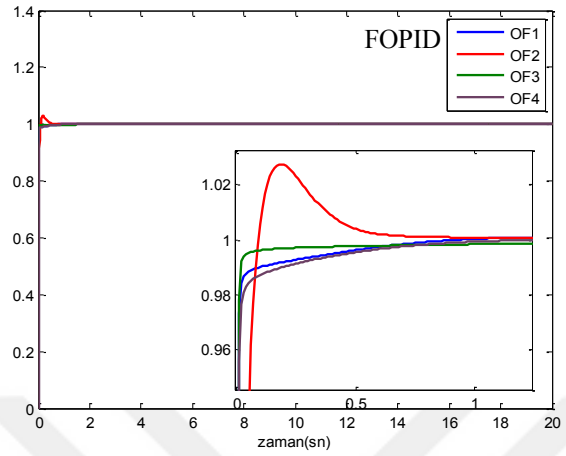
Tablo 4.56. T_{11} süreci için PSO algoritması kullanarak elde edilen FOPID ve PID denetleyici parametreleri

Kontrolör	Yöntem		Kp	Ki	Kd	λ	μ
FOPID	Tez Çalışması	OF_1	84.541	35.522	26.684	1.3357	0.600
		OF_2	59.7199	94.006	2.8058	1.1474	0.9460
		OF_3	89.4114	3.5086	94.1358	1.4151	0.8869
		OF_4	71.8787	13.1182	21.5362	1.2688	0.980
PID	Tez Çalışması	OF_1	19.741	73.221	1.101	-	-
		OF_2	6.999	23.001	0.969	-	-
		OF_3	35.211	9.203	3.119	-	-
		OF_4	13.8503	55.1075	27.8926	-	-

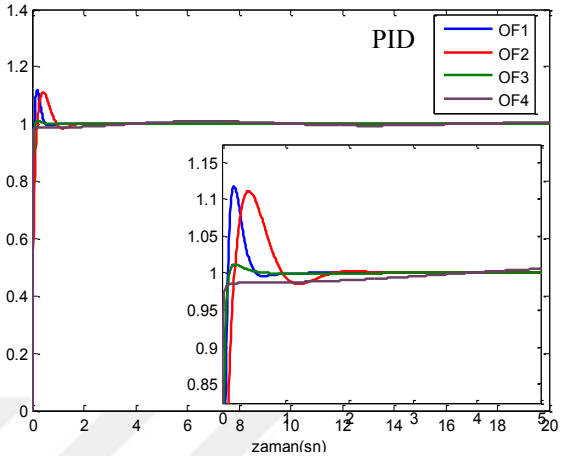
Tablo 4.57. T_{11} sürecine ait PSO algoritmasına dayalı optimum denetleyiciler için PSO algoritması kullanılarak elde edilen temel performans karakteristikleri

Kont.	Yöntem		ISE	ITAE	OF	Tr(sn)	Ts(sn)	OS%	SS
FOPID	Tez Çalışması	OF_1	0.0050	-	0.7035	0.0082	0.0098	0.0953	0.0001
		OF_2	-	0.0046	8.5948	0.0348	0.0542	2.7592	0.0002
		OF_3	0.0050	-	0.0980	0.0080	0.0095	0	0.0002
		OF_4	-	0.0056	0.1065	0.0084	0.0099	0.0001	0.0001
PID	Tez Çalışması	OF_1	0.0190	-	36.9909	0.0662	0.3434	11.6947	0.0069
		OF_2	-	0.0436	37.2234	0.1302	0.7123	11.0752	0.0047
		OF_3	0.0090	-	3.8834	0.0433	0.0641	1.1578	0.0065
		OF_4	-	0.1101	3.2921	0.0103	0.0182	0.7001	0.0233

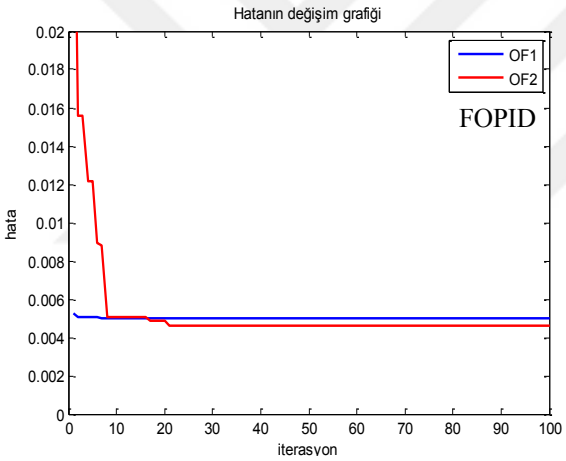
T_{11} süreci için bütün algoritmaların yer aldığı ortak bir değerlendirme Tablo 4.58'de verilmiştir. Bu değerlendirme için sadece OF_3 amaç fonksiyonu dikkate alınmıştır.



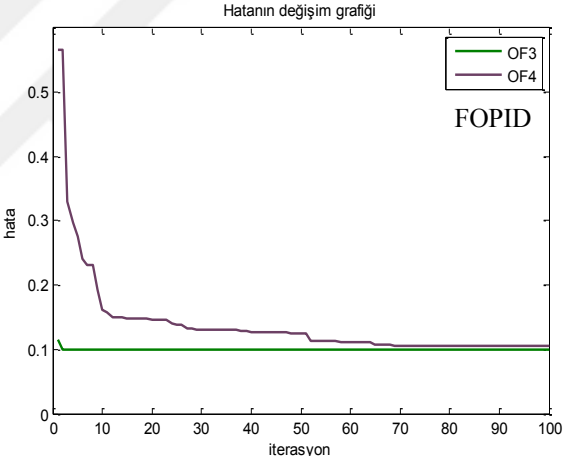
a



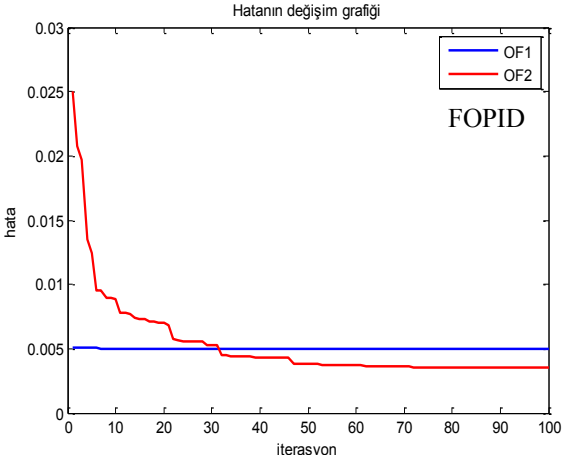
b



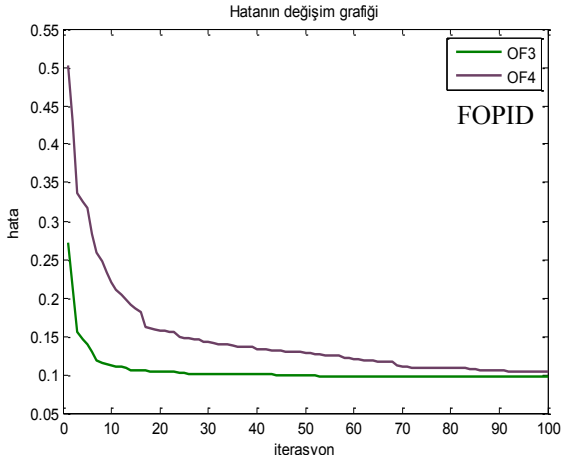
c



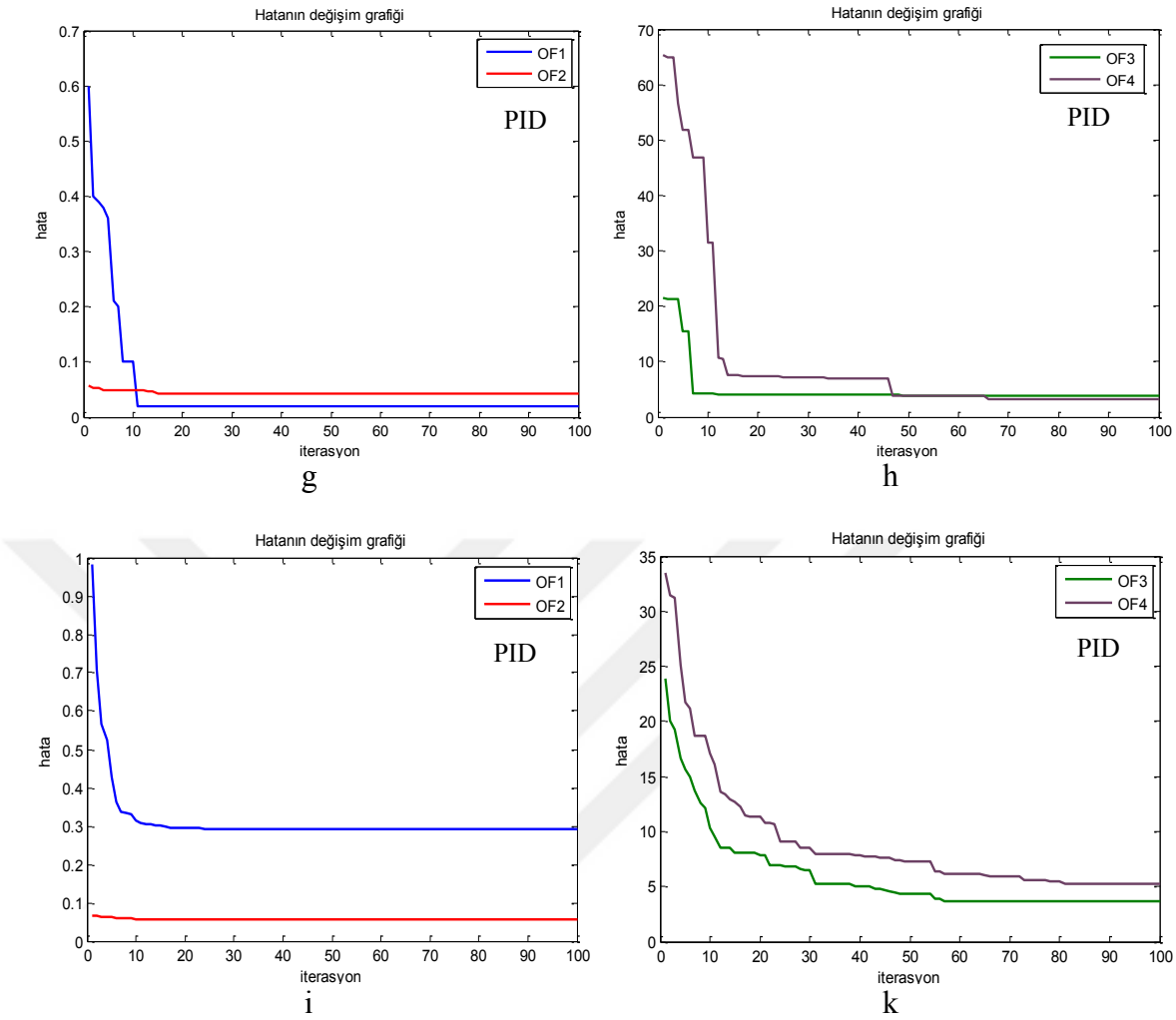
d



e



f



Şekil 4.30. T_{11} süreci için PSO algoritmasına dayalı FOPID ve PID kontrolör tasarımları sonucu elde edilen basamak cevapları ve hata gelişim grafikleri

Tablo 4.58'e göre T_{11} süreci için en iyi denetleyici yapısı FOPID için PSO algoritması kullanılarak, PID denetleyici için DG algoritması kullanarak elde edilmiştir. Ancak FOPID denetleyici için elde edilen değerlerin üç algoritma için de birbirine oldukça yakın olduğu da söylenebilir. Üç algoritmanın da FOPID denetleyicide sıfır aşma değeri ile sonuca ulaşması dikkat çekicidir. Standart sapma değerlerinde ise DGA'nın oldukça başarılı olduğunu söylemek mümkündür.

Tablo 4.58. T_{11} sürecine ait optimum denetleyiciler için algoritmaların OF_3 fonksiyonuna göre performans kıyaslaması

Kontr.	Algoritma	ISE	OF_3	Tr(sn)	Ts(sn)	OS%	SS
FOPID	YAK	0.0054	0.1364	0.0086	0.0164	0	0.0008
	DGA	0.0050	0.0987	0.0082	0.0097	0	0.0001
	PSO	0.0050	**0.0980**	0.0080	0.0095	0	0.0002
PID	YAK	0.1052	4.7978	0.4609	0.5803	0.2814	0.0999
	DGA	0.0085	**0.2934**	0.0198	0.0416	0	0.0003
	PSO	0.0090	3.8834	0.0433	0.0641	1.1578	0.0065

4.3. Modellenen Süreçler için Kontrolör Tasarımı

Tez çalışmasının ilk aşamasında Bölüm 3'de yer alan ve Eşitlik 3.1-3.3'de sunulan G_1, G_2 ve G_3 tamsayı dereceli süreçler için tamsayı ve kesir dereceli model yapıları önerilmişti. Bu süreçleri tanımlamak için YAK, DG ve PSO algoritmaları kullanılarak hem tamsayı hem de kesir dereceli yeni modeller elde edilmişti. Elde edilen bu yeni süreçler için FOPID kontrolör tasarımı ise tez çalışmasının bu aşamasında gerçekleştirilmiştir.

Tablo 3.1'de yer alan model parametreleri kullanılarak elde edilen yeni süreçler için FOMCON toolbox [20] kullanılarak denetleyici tasarımları gerçekleştirilmiştir. Performans indisi olarak ISE, ITSE, IAE ve ITAE hataları tercih edilmiştir. G_1 ve G_2 süreçleri için hem FOPID hem de PID denetleyici tasarımı gerçekleştirilmiş ancak G_3 sürecinin model tasarım aşamasında tamsayı dereceli modelin yeterince başarılı olamaması nedeniyle bu süreç model yapısı ile elde edilen yeni süreçler için denetleyici tasarımı gerçekleştirilmemiştir. Modellenen süreçlere ait transfer fonksiyonları Eşitlik 4.16-4.18'de verilmiştir [121].

$$G_1(\mathrm{s}) = \frac{1}{(s+1)(\mathrm{s}^2 + 0.2s + 0.1)} \tag{4.16}$$

$$G_2(\mathrm{s}) = \frac{9}{\mathrm{s}^2 + 1.2s + 9} e^{-0.5s} \tag{4.17}$$

$$G_3(s) = \frac{46.21s + 206.1}{0.9372s^3 + 2.656s^2 + 75.87s + 112.1} \tag{4.18}$$

Bu sistemler için önerilen modeller ile oluşan yeni süreçler ise algoritma ve model yapılarına uygun olarak kodlamalar yapılarak Tablo 4.59'da sunulmuştur. Örneğin G_1 süreci için YAK algoritması kullanılarak elde edilen tamsayı dereceli model için G_{1TY}, aynı algoritma ile elde edilen kesir dereceli model için G_{1KY} şeklinde adlandırmalar yapılmıştır. Aynı süreçte DG algoritması için G_{1TD}, G_{1KD} ve PSO algoritması için G_{1TP}, G_{1KP} kısaltmaları tercih edilmiştir. Ayrıca Tablo 4.59'da verilen yeni süreçler için kararlılık analizleri yapılmış, bütün süreçlerin kararlı yapıda oldukları tespit edilmiştir.

Tablo 4.59. Kontrolör tasarımı için kullanılan ve farklı algoritmalarla elde edilen model yapıları

Süreç	Model yapısı türü	Elde Edilen Modele ait Transfer Fonksiyonu
G_1	**Tam sayı dereceli**	$G_{1TY}(s)=\frac{60.49}{(61.17s^2+18.34s+60.98)}e^{0.757}$
		$G_{1TD}(s)=\frac{67.04}{(67.48s^2+20.05s+67.51)}e^{0.757}$
		$G_{1TP}(s)=\frac{69.356}{(70.859s^2+20.651s+69.965)}e^{0.730}$
	Kesir dereceli	$G_{1KY}(s)=\frac{43.519}{(61.725s^{2.4869}+50.753s^{1.0492}+43.527)}e^{0.2095}$
		$G_{1KD}(s)=\frac{19.413}{(26.555s^{2.5}+23.195s^{1.0977}+19.506)}e^{0.2242}$
		$G_{1KP}(s)=\frac{58.618}{(81.951s^{2.3912}+58.501s^{0.965}+58.143)}e^{0.3149}$
G_2	**Tam sayı dereceli**	$G_{2TY}(s)=\frac{90.601}{(10.299s^2+12.441s+90.380)}e^{0.501}$
		$G_{2TD}(s)=\frac{87.840}{(9.386s^2+11.560s+88.261)}e^{0.510}$
		$G_{2TP}(s)=\frac{32.944}{(3.6902s^2+4.7131s+32.9115)}e^{0.4909}$
	Kesir dereceli	$G_{2KY}(s)=\frac{96.778}{(9.7217s^{2.031}+12.291s^{1.0867}+96.937)}e^{0.5081}$
		$G_{2KD}(s)=\frac{81.345}{(9.9972s^{1.929}+7.7785s^{0.9024}+81.263)}e^{0.5183}$
		$G_{2KP}(s)=\frac{13.495}{(0.4566s^{2.5}+2.8068s^{1.4323}+13.5847)}e^{0.4996}$
G_3	**Kesir dereceli**	$G_{3KY}(s)=\frac{63.605}{(0.3961s^{2.4254}+15.417s^{0.6679}+32.223)}e^{0.0073}$
		$G_{3KD}(s)=\frac{100}{(0.6365s^{2.2969}+18.702s^{0.6454}+51.737)}e^{0.0232}$
		$G_{3KP}(s)=\frac{63.196}{(0.3423s^{2.4495}+11.134s^{0.7636}+34.428)}e^{0.0010}$

Tablo 4.60'da kesir dereceli modeller için elde edilen FOPID denetleyicilere ait parametre değerleri ve hata değerleri sunulmuştur. Tabloda her bir süreç için elde edilen en düşük hata değeri kalın yazı stili ile belirtilmiştir. Bu şekilde ele alınan süreç için hangi performans indisinde hangi algoritmanın daha başarılı sonuçlar elde ettiği ortaya koyulmuştur.

Tablo 4.60. Kesir dereceli modeller için algoritmaları kullanarak elde edilen FOPID denetleyici parametreleri

Süreç	Model.	Perf. indisi	Kp	Ki	Kd	λ	μ	Hata
G_1	G_{1KY}	**ISE**	3.8501	2.9101	4.5779	1.0022	1.500	**0.3451**
		ITSE	2.8663	3.1407	4.3803	0.9002	1.4362	**0.0888**
		IAE	3.8931	3.1547	4.6854	1.0089	1.4683	**0.5337**
		ITAE	3.6784	3.4021	4.7923	0.9973	1.4332	0.3925
	G_{1KD}	ISE	3.4662	2.7697	4.2347	0.9697	1.500	0.3770
		ITSE	1.3832	2.6944	3.2059	0.8315	1.3020	0.1818
		IAE	2.1207	3.2355	4.0289	0.8832	1.3501	0.7888
		ITAE	4.5558	2.5582	3.9845	0.9993	1.4593	**0.3492**
	G_{1KP}	ISE	2.7328	2.3378	3.4775	0.86269	1.5	0.4511
		ITSE	3.1606	1.8466	3.0341	1.0678	1.475	0.1077
		IAE	2.8606	1.6697	2.7899	1.0664	1.499	0.5902
		ITAE	4.9955	3.7391	3.692	1.3832	1.3917	0.8880
G_2	G_{2KY}	**ISE**	0.6575	1.4069	0.1725	0.9038	1.4435	**0.5888**
		ITSE	0.7667	1.3663	0.1470	1.0568	1.4946	0.2905
		IAE	0.5797	1.3054	0.1359	1.0340	1.4064	**0.8048**
		ITAE	0.4382	1.2591	0.1284	1.0087	1.3050	0.4394
	G_{2KD}	ISE	0.6623	1.4979	0.1600	0.9898	1.4196	0.5963
		ITSE	0.6811	1.4591	0.1489	1.0500	1.4254	**0.2807**
		IAE	0.3105	1.2002	0.1247	0.9970	1.1974	0.8704
		ITAE	0.4297	1.2842	0.1300	1.0192	1.2906	**0.4028**
	G_{2KP}	ISE	0.1226	1.1704	0.1289	0.8654	1.1163	0.8130
		ITSE	0.0794	1.0514	0.1280	0.9436	0.9706	0.4103
		IAE	0.3811	0.9674	0.0859	1.0074	1.3818	1.0547
		ITAE	0.2659	0.9533	0.0902	1.0085	1.2263	0.7355
G_3	G_{3KY}	ISE	43.275	30.761	0.5527	0.2838	1.498	0.0170
		ITSE	6.505	87.496	0.6621	0.9337	1.3193	0.0001
		IAE	17.918	99.949	0.5447	1.1105	1.3688	0.0846
		ITAE	22.373	76.527	1.2204	0.6417	1.3159	**0.0018**
	G_{3KD}	ISE	3.2118	19.666	0.1134	0.9912	1.5	0.0359
		ITSE	2.7692	16.653	0.1063	0.9549	1.4991	0.0005
		IAE	2.2452	16.657	0.1137	0.9442	1.4329	**0.0474**
		ITAE	1.9545	19.941	0.1568	0.8937	1.3555	0.0022
	G_{3KP}	**ISE**	10	9.999	1.1942	0.7299	1.4182	**0.0119**
		ITSE	9.9998	9.9144	3.3344	0.7548	1.4065	**$2x10^{-5}$**
		IAE	9.7815	9.9852	0.4179	1.2405	1.4718	0.0559
		ITAE	5.0712	10	0.1926	1.0001	1.4871	0.0156

Tablo 4.59'da verilen tamsayı dereceli modeller için farklı algoritmalarla elde edilen FOPID denetleyicilere ait parametre değerleri ise Tablo 4.60'de sunulmuştur. Tablolarda verilen "Hata" ifadesi, ilgili çalışmada dikkate alınan performans indisinin (ISE, ITSE vb.) değerini belirtmektedir.

Tablo 4.60'a göre G_1 süreci için kesir dereceli modellerle elde edilen yeni süreçlerde YAK algoritması ile elde edilen yeni sürecin daha başarılı sonuçlar sergilediği söylenebilir. ISE, ITSE ve IAE performans indislerinde G_{1KY} 0.3451, 0.0888 ve 0.5337 hata değerleri ile en iyi sonuçları elde etmiştir. DG algoritmasına ait G_{1KD} ise ITAE hata değerinde 0.3492 değeri ile en iyi sonuca ulaşırken, bu süreçte PSO algoritması hiçbir performans indisinde diğer algoritmalardan daha başarılı sonuç ortaya koyamamıştır.

G_2 süreci için YAK ve DG algoritmalarına ait yeni süreçler olan G_{2KY} ve G_{2KD} ikişer performans indisinde en iyi sonuçlara ulaşırken, PSO yine bu süreçte hiçbir performans indisinde en iyi sonuca ulaşamamıştır. G_{2KY} 0.5888 ISE ve IAE 0.8048 hata değerleri ile bu performans indislerinde en iyi sonuçları elde ederken, G_{2KD} 0.2807 ITSE ve 0.4028 ITAE değeri ile bu indislerde en başarılı sonuca ulaşmıştır. G_{2KP} için önerilen denetleyici yapısında ise hiçbir hata indisinde en iyi sonuç elde edilememiştir.

G_3 sürecinde diğer iki süreçten daha farklı sonuçlar elde edildiği tablodan görülmektedir. PSO algoritması ile elde edilen G_{3KP} sürecinde 0.0119 ISE, 0.0000002 ITSE değeri ile en iyi sonuçlar elde edilmiştir. G_{3KY} 'de ise 0.0018 ITAE değeri ve G_{3KD} sürecinde 0.0474 IAE değeri ile en sonuçlara ulaşılmıştır. Bütün süreçler bir bütün olarak değerlendirildiğinde FOPID denetleyici tasarımında en başarılı sonuçların YAK algoritması ile elde edilen sürece ait olduğunu ifade etmek mümkündür.

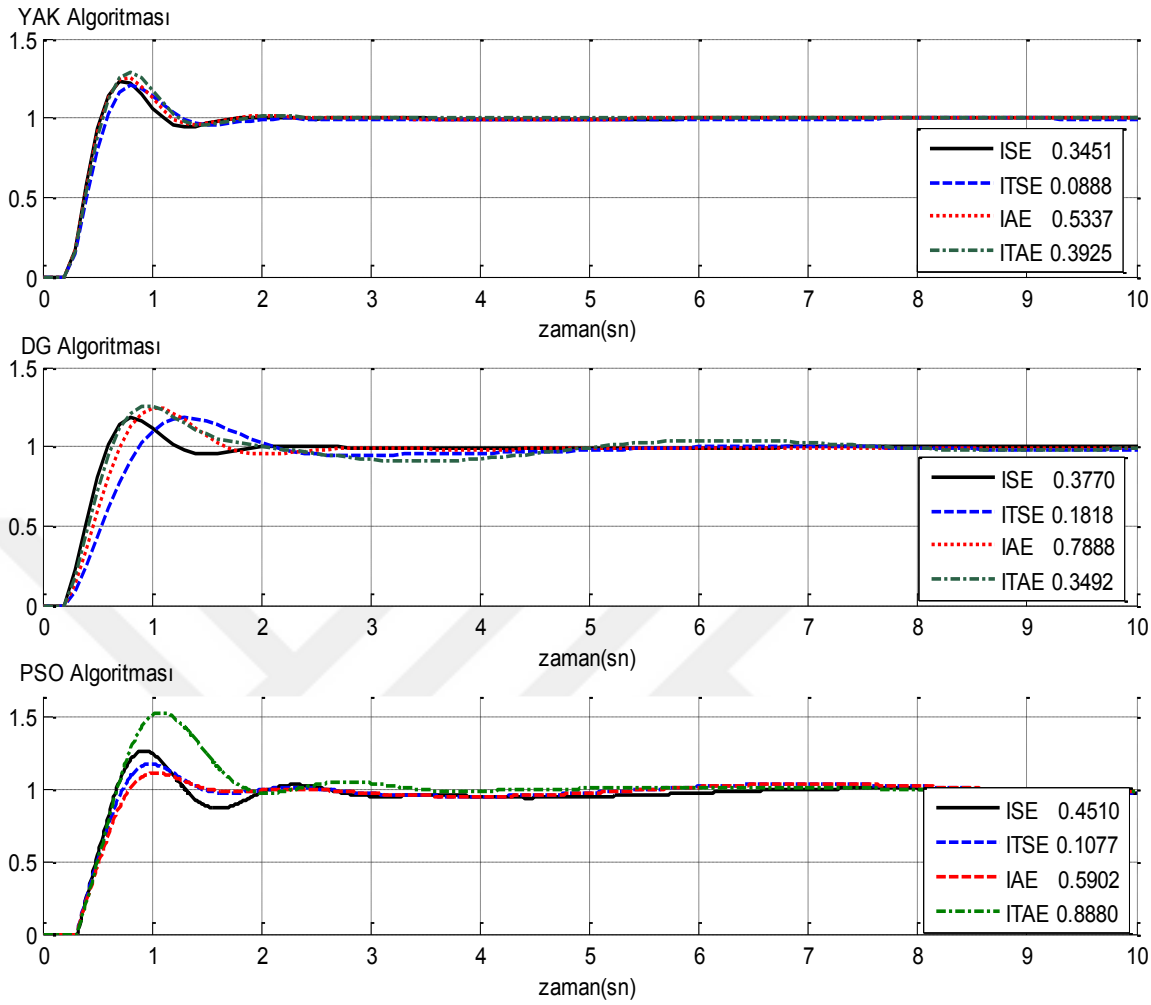
Tablo 4.61, farklı algoritmalarla elde edilen tam sayı dereceli modellere ait parametre ve hata değerlerini göstermektedir. Bu tablodaki değerlerden, G_1 sürecine ait sonuçlarda, G_{1TP} modeli için önerilen denetleyici yapısının en başarılı sonuçları ortaya koyduğu söylenebilir. G_{1TP} modeli için yapılan çalışmada 0.8803 ISE, 0.4616 ITSE ve 1.3199 IAE hata değeri elde edilirken, G_{1TY}'ye ait çalışmada 1.3717 ITAE değeri ile sadece bu süreçte en başarılı sonuca ulaşılmıştır.

G_2 sürecinde de G_1 sürecine benzer bir durum söz konusudur. PSO algoritması ile elde edilen G_{2TP} modelinde 0.5741 ISE, 0.1803 ITSE ve 0.7866 IAE değeri ile bu indislerde en iyi sonuçlar elde edilmiştir. DGA'ya ait model yapısına dayalı çalışmada herhangi bir performans indisinde başarı sağlanamazken, G_{2TY} 0.4174 ITAE değeri ile en iyi sonuca ulaşmayı sağlamıştır.

Tam sayı dereceli model yapıları ile elde edilen, PSO'ya dayalı denetleyici yapılarında diğerlerine göre daha başarılı hata değerlerine ulaşıldığı Tablo 4.61'den açıkça görülmektedir. Elde edilen bu sonuçlara ait basamak cevaplarını gösteren grafikler G_1, G_2 ve G_3 için Şekil 5.1-5.5'te verilmiştir.

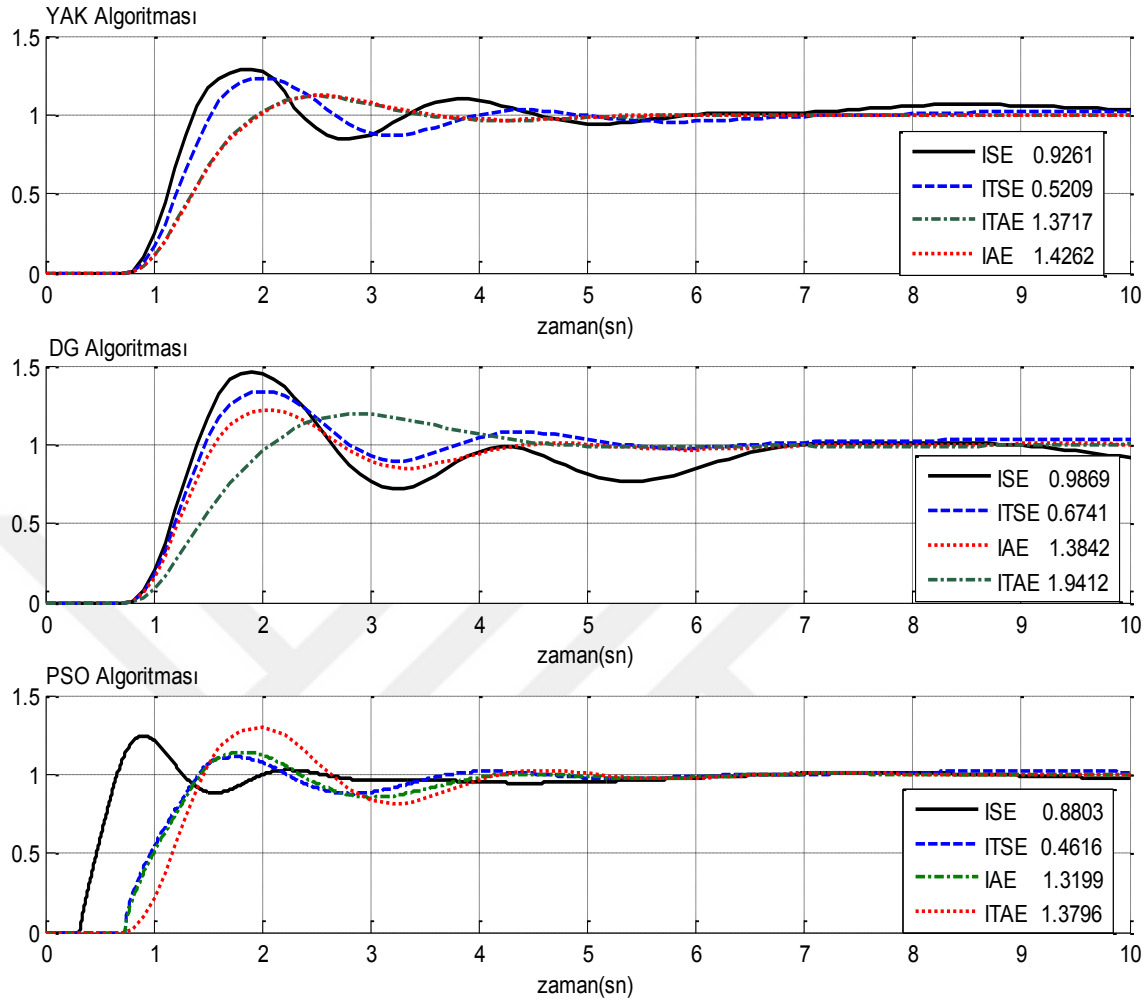
Tablo 4.61. Tam sayı dereceli modeller için algoritmaları kullanarak elde edilen FOPID denetleyici parametreleri

Süreç	Model.	Perf. indisi	Kp	Ki	Kd	λ	μ	Hata
G1	G_{1TY}	ISE	1.3334	0.7312	1.0963	1.1666	1.4994	0.9261
		ITSE	0.9531	0.7797	1.0379	1.0424	1.3881	0.5209
		IAE	0.4430	0.7684	0.8298	0.9988	1.1531	1.4262
		ITAE	0.4663	0.7630	0.8303	0.9999	1.1654	**1.3717**
	G_{1TD}	ISE	0.9168	1.3117	1.2810	0.7772	1.3857	0.9869
		ITSE	1.1186	0.8540	1.0816	1.1297	1.3898	0.6741
		IAE	0.8354	0.7880	1.0250	1.011	1.3383	1.3842
		ITAE	0.2058	0.8437	0.8353	0.9996	0.9794	1.9412
	G_{1TP}	**ISE**	1.5158	0.79305	1.2163	1.1429	1.4782	**0.8803**
		ITSE	1.0266	0.8147	1.101	1.0667	1.3812	**0.4616**
		IAE	0.9055	0.8656	1.1417	1.0148	1.3237	**1.3199**
		ITAE	0.8956	0.8650	1.1347	1.0161	1.3183	1.3796
G2	G_{2TY}	ISE	0.8485	1.4591	0.1659	1.0558	1.4999	0.5855
		ITSE	0.8181	1.3900	0.1577	1.0611	1.4912	0.1870
		IAE	0.5918	1.3152	0.1460	1.0222	1.3937	0.8008
		ITAE	0.4639	1.2734	0.1360	1.0202	1.3023	**0.4174**
	G_{2TD}	ISE	0.8318	1.4886	0.1560	1.0578	1.500	0.5852
		ITSE	0.7997	1.3877	0.1459	1.0637	1.4994	0.1857
		IAE	0.5804	1.3312	0.1368	1.0215	1.3974	0.7984
		ITAE	0.4073	1.2245	0.1228	1.0019	1.2812	0.4497
	G_{2TP}	**ISE**	0.8791	1.4962	0.1668	1.0648	1.5	**0.5741**
		ITSE	0.8380	1.4011	0.1573	1.4011	1.4961	**0.1803**
		IAE	0.6095	1.336	0.1463	1.0227	1.3952	**0.7866**
		ITAE	0.4980	1.2911	0.1390	1.0083	1.3123	0.4190



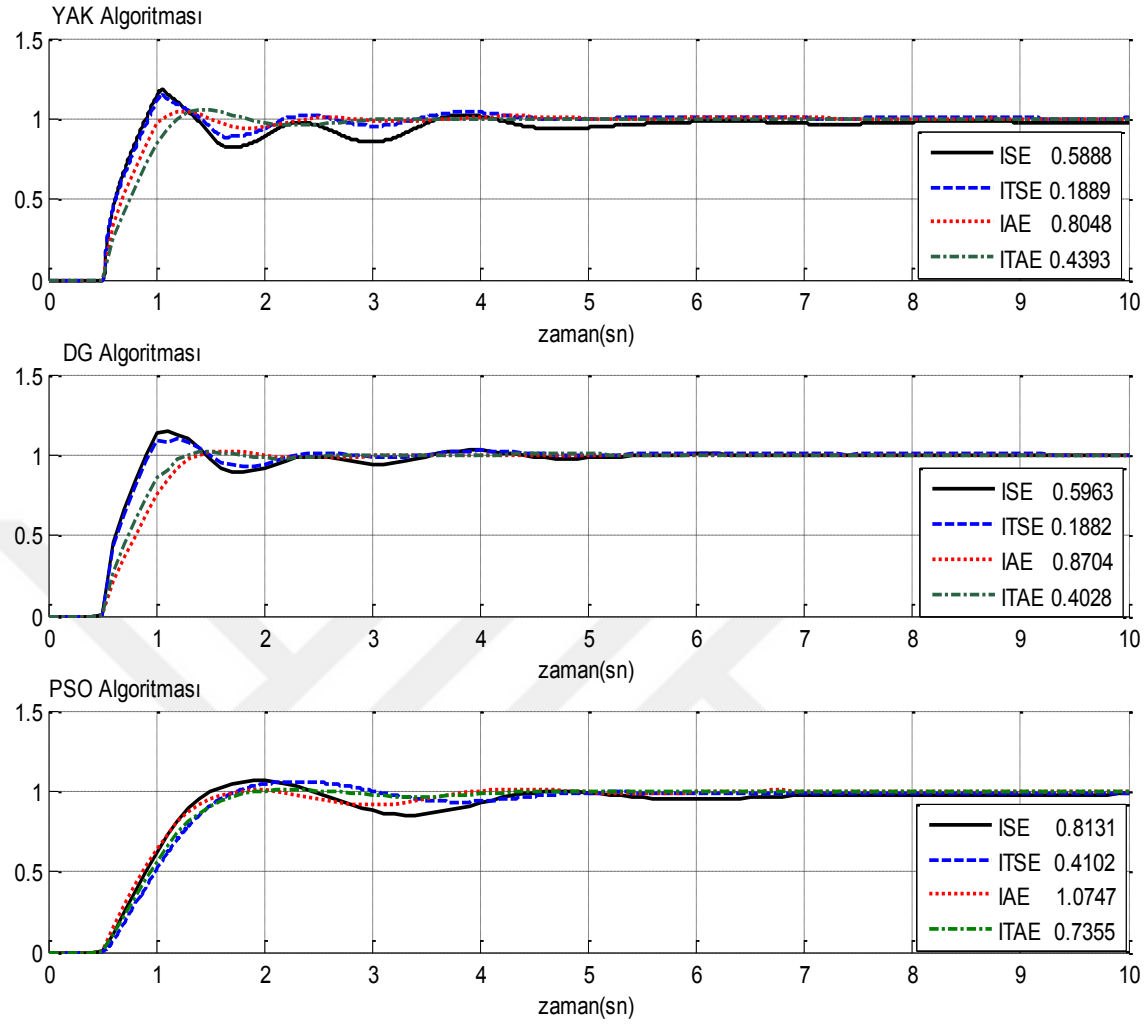
Şekil 4.31. G_1 süreci için önerilen kesirli dereceli model yapılarına ait kontrolör tasarımı sonucu elde edilen basamak cevapları

G_1 sürecinde kesir dereceli model yapısına ait kontrolör tasarımı ile elde edilen basamak cevapları Şekil 4.31'de sunulmuştur. Şekilde en üstte yer alan grafik YAK algoritmasına ait model yapısı ile elde edilmiş çıkışları gösterirken, ortadaki Şekil DGA'ya ve en altta yer alan grafik ise PSO algoritmasına aittir. YAK algoritmasına dayalı yeni süreç için önerilen kontrolöre ait basamak cevaplarında bütün hata indislerinin birbirine yakın performans sergiledikleri görülmektedir. DG algoritmasına dayalı yeni süreç için önerilen kontrolöre ait basamak cevaplarında, ISE hata indisinin kullanılması durumunda elde edilen denetim sistemi cevabının daha az aşma miktarına sahip olduğu, yükselme süresi ve oturma süresinin de diğer indislerin kullanımında elde edilene göre az da olsa daha küçük olduğu Şekil 4.31'den anlaşılmaktadır. PSO algoritmasına dayalı model yapısında ise ITAE hata indislerinin kullanımının sistem cevabında oldukça yüksek aşma değerine sebep olması dikkat çekicidir.



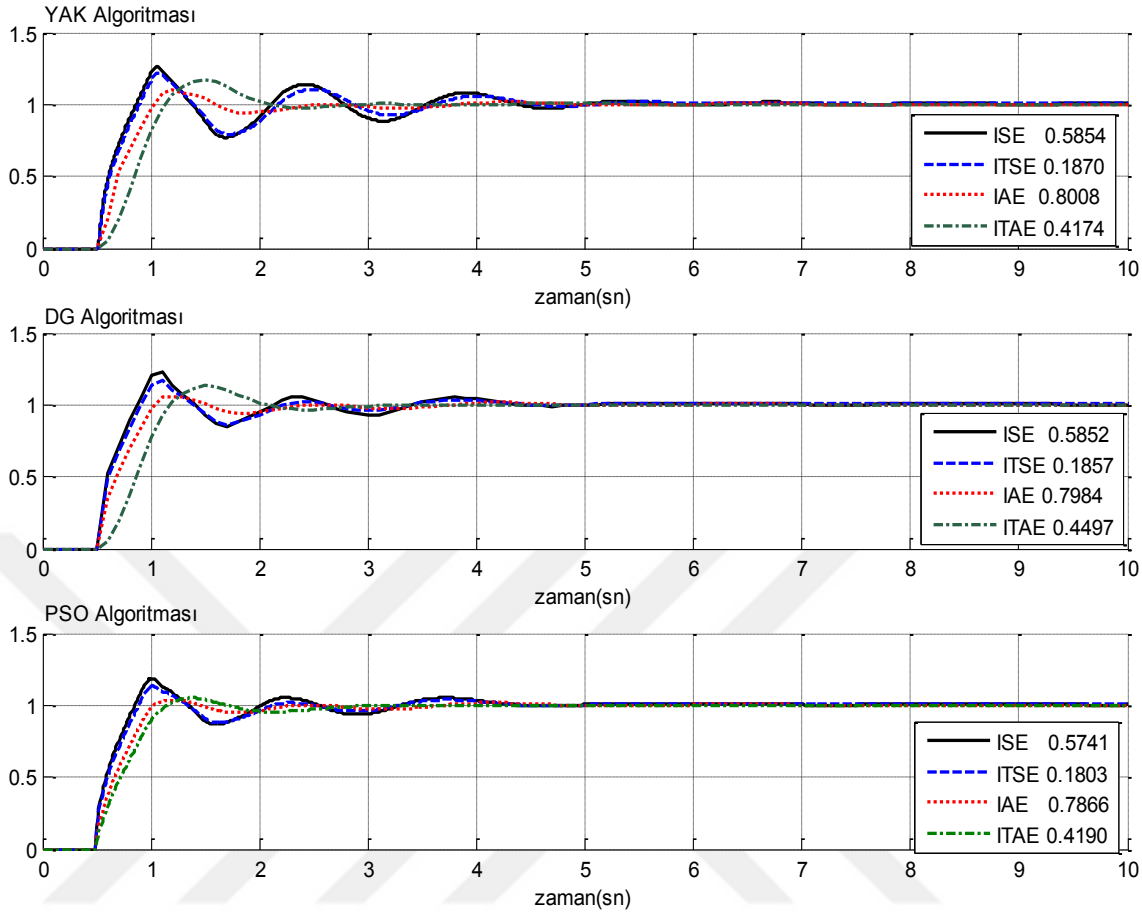
Şekil 4.32. G_1 süreci için önerilen tamsayı dereceli model yapılarına ait kontrolör tasarımı sonucu elde edilen basamak cevapları

Şekil 4.32'de tamsayı dereceli model yapısı dikkate alınarak tasarlanan kontrolör etkisi altındaki sistemlerin cevabı verilmiştir. YAK algoritması ile elde edilen basamak cevaplarına göre en yüksek aşma değeri ile en uzun oturma süresi ISE performans indisinde elde edilmiştir. IAE ve ITAE performans indislerine ait basamak cevaplarının birbirine oldukça yakın olduğu yine Şekil 4.32'den açıkça görülmektedir. DG algoritması ile elde edilen basamak cevaplarında ise ITSE ve ISE performans indislerindeki yüksek aşma miktarı dikkat çekicidir. ITAE hata indisinde ise diğer üç indise göre daha düşük yükselme süresi ve daha düşük aşma miktarı ile minimum hata değeri elde edilmiştir. PSO algoritmasına ait son grafikte ISE hata indisinin diğer indislere göre oldukça hızlı tepki ve oturma süresi sağladığı açıkça görülmektedir.



Şekil 4.33. G_2 süreci için önerilen kesirli dereceli model yapılarına ait kontrolör tasarımı sonucu elde edilen basamak cevapları

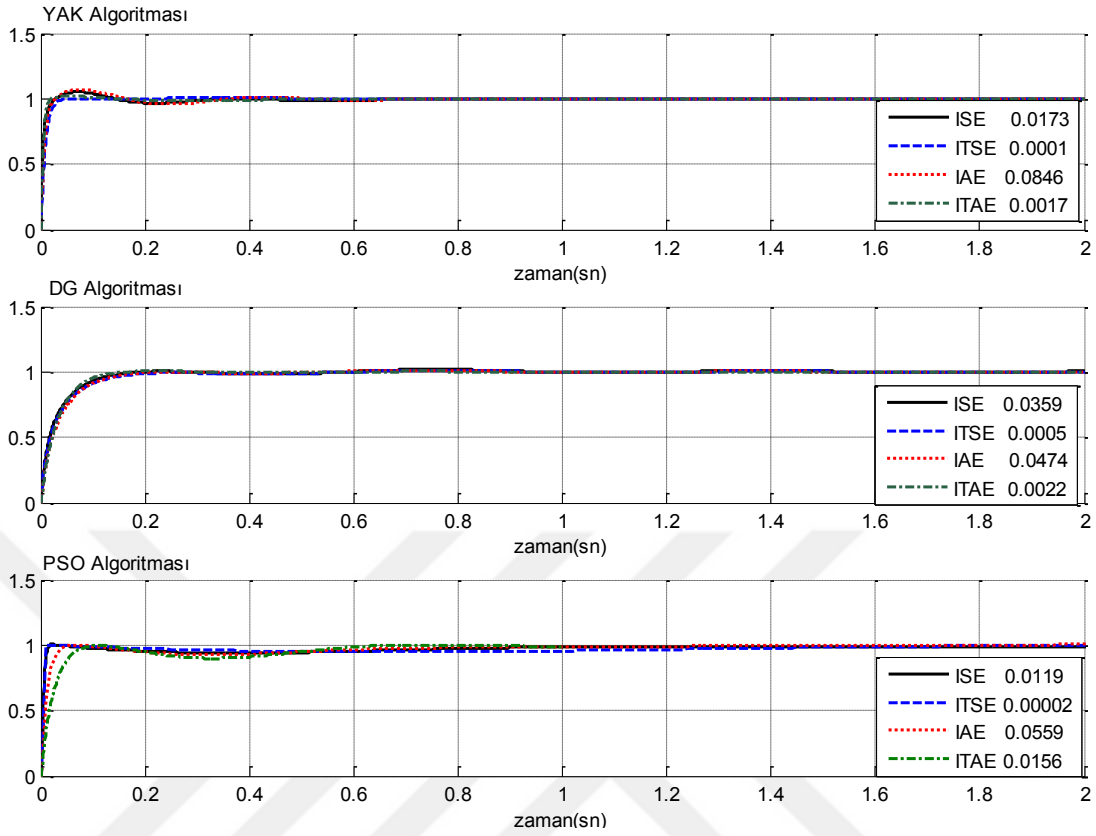
G2 süreci için elde edilen kesir dereceli model yapılarına dayalı kontrolör tasarımı ile elde edilen sistem cevapları Şekil 4.33'te sunulmuştur. Bu şekle göre YAK algoritmasına ait çıkış cevaplarında ISE ve ITSE hata indislerine ait çıkışlar ile IAE ve ITAE hata indislerine ait çıkışların benzer değişimlere sahip olduğu görülmektedir. ISE ve ITSE hata indislerine ait basamak cevaplarının aşma miktarı diğer indislere göre yüksek, yükselme süresi ise daha düşüktür. DG algoritmasına ait basamak cevaplarında da benzer sonuçlar görülmektedir. YAK algoritması ile aynı şekilde ISE ve ITSE hata indislerinde yüksek aşma ve tepki süresinin oldukça iyi olduğu görülmektedir. PSO algoritmasında ise bu iki algoritmaya göre daha farklı sonuçlar elde edilmiştir. Bütün performans indislerinin birbirine oldukça yakın çıkışlar verdiği bu algoritmada aşma miktarı oldukça düşüktür. Ancak tepki hızı diğer iki algoritma ile elde edilen sonuçlara göre daha yavaştır.



Şekil 4.34. G_2 süreci için önerilen tamsayı dereceli model yapılarına ait kontrolör tasarımı sonucu elde edilen basamak cevapları

G2 sürecinde tamsayı dereceli model yapıları kullanılarak tasarlanan kontrolörler ile elde edilen sistem cevapları Şekil 4.34'te verilmiştir. Şekil 4.34'deki değişimler, YAK algoritmasına ait tamsayı dereceli model yapısı için elde edilen kontrolör çıkışlarının G1 sürecine benzer sonuçlar sunduğunu anlatmaktadır. ISE ve ITSE performans indislerinin hızlı tepki cevabı ve yüksek aşma miktarı ortaya koyduğu da Şekil 4.34'ten görülmektedir. Ayrıca YAK algoritması için yüzde aşımın en düşük gerçekleştiği performans indisinin IAE olduğu yine aynı şekil üzerinde görülmektedir.

DG ve PSO algoritmalarında da YAK algoritmasına benzer sonuçlar elde edilmiştir. Bu iki algoritma için, IAE ve ITAE performans indislerinin düşük aşma değeri ve düşük tepki hızına sahip sistemler ortaya koyduğu şekilden görülmektedir. ISE ve ITSE performans indislerinde ise aşmanın diğer iki performans indisine göre daha yüksek olduğu da gözlenebilmektedir. Elde edilen hata değerlerine göre, PSO algoritmasına dayalı model yapısı ile tasarlanan kontrolöre sahip sistemde, ITAE dışında kalan indislerde en düşük hata değerlerine ulaşıldığı da yine Şekil 4.34'ten görülmektedir.



Şekil 4.35. G_3 süreci için önerilen kesirli dereceli model yapılarına ait kontrolör tasarımı sonucu elde edilen basamak cevapları

Şekil 4.35'te G3 süreci için elde edilen kesir dereceli model yapıları dikkate alınarak tasarlanan kontrolörlere sahip sistemlerin basamak cevaplarına yer verilmiştir. Bu süreçte tamsayı dereceli model yapılarının başarısız sonuçlar vermesi nedeniyle sadece kesir dereceli modellerin dikkate alındığı çalışmaların sonuçları sunulmuştur. Şekilde yer alan sistem cevabı değişimlerinden, üç algoritma içinde bütün hata indislerinde oldukça yakın sonuçlar elde edildiği sadece PSO algoritmasına ait model yapısında elde edilen sonuçlarda çok az da olsa farklılıkların mevcut olduğu görülmüştür. Şekil 4.35'e göre YAK algoritması ve PSO algoritmasına ait modeller için önerilen yeni süreçler için tasarlanan denetleyici yapılarının tepki hızlarının DGA'ya göre daha iyi olduğu görülmektedir. Ayrıca üç algoritma arasında aşmanın en yüksek olduğu denetleyici yapılarının YAK algoritmasına dayalı çalışmalara ait olduğu görülmektedir.

Performans indislerine göre en başarılı sonuçların PSO algoritmasına dayalı denetleyici tasarımlarında elde edildiği görülebilmektedir. ISE, ITSE ve IAE performans indislerinde en iyi sonuçlar bu algoritmaya ait model yapıları için elde edilirken, sadece ITAE performans indisi için YAK algoritmasına ait çalışmalarda başarılı sonuç ve düşük bir hata değeri elde edilmiştir.

5. BÖLÜM

TARTIŞMA, SONUÇ VE ÖNERİLER

5.1. Tartışma, Sonuç ve Öneriler

Sistemlerin analizi ve tasarımında kullanılan modellemenin doğru ve aynı zamanda etkili bir şekilde yapılması kontrol sistemi tasarımında önemli bir faktördür. Sistemlerin tanımlanmasında seçilecek model yapısının uygunluğu, modellemenin başarılı bir şekilde yapılması açısından oldukça önemlidir.

Ayrıca arzu edilen bir sistem tepkisini elde etmek için kontrolör parametreleri uygun bir ayar yaklaşımı ile dikkatlice belirlenmelidir. Sistemlerin kontrolü için önerilen denetleyici yapılarına ait parametrelerin doğru ve hızlı bir şekilde tahmin edilmesi arzu edilen sistem cevaplarını elde edebilmek için önem taşımaktadır. Uygun çözümün elde edilmesinde sezgisel algoritmaların kullanımı klasik yöntemlere göre oldukça hızlı ve etkili çözümler sunabilmektedir.

Bu tez çalışmasının iki temel amacı vardır. Bunlardan ilki kesir ve tamsayı dereceli yapılar kullanarak sistemlerin modellenmesidir. İkinci temel amaç ise istenilen sistem davranışının elde edilebilmesi için kullanılabilecek kesir ve tamsayı dereceli uygun denetleyici yapılarının yapay zekâya dayalı algoritmalar yardımıyla tasarlanmasıdır.

Bu amaçlara uygun olarak tez çalışmasında öncelikle kesir dereceli ve tamsayı dereceli sistemlerin yapıları ve matematiksel modelleri hakkında kısa bilgiler ilk bölümde verilmiştir. Ayrıca literatürde yer alan çalışmalardan da kısaca bu bölümde bahsedilmiştir.

İkinci bölümde ise çalışmalarda kullanılan algoritmalar tanıtılmış, model ve kontrolör yapıları hakkında bilgiler verilmiştir. YAK, DG ve PSO algoritmaları ve çalışma prensipleri bu bölümde sunulmuştur. Sistemleri modellemek ve kontrol etmek için

kullanılan yapılardan ve matematiksel karşılıklarından da yine bu bölüm içinde bahsedilmiştir.

Tez çalışmasının üçüncü bölümünde modellemeye ait çalışmalar sunulmuştur. İlk önce tamsayı dereceli örnek sistemler için hem tam sayı hem de kesir dereceli model yapıları sunulmuş, model yapılarına ait parametreler YAK, DG ve PSO algoritmaları kullanılarak elde edilmiştir. Daha sonra kesir dereceli süreçler için aynı aşamalar kullanılarak modelleme işlemi gerçekleştirilmiştir.

Literatürde yer alan örnek süreçler için kesir ve tamsayı dereceli denetleyici tasarımları dördüncü bölümde incelenmiştir. Yukarıda sözü edilen algoritmalar yardımıyla elde edilen denetleyici yapılarına ait parametreler ve denetleyicilerin etkisindeki sistemlere ait basamak cevabı değişimleri bu bölüm içinde birbiriyle ve literatürle karşılaştırmalı olarak sunulmuştur.

Tez çalışmasının beşinci bölümünde, daha önce üçüncü bölüm içinde modellemesi yapılan sistemler için kontrolör tasarımları yapılmıştır. Yine algoritmaların kullanımına dayalı biçimde elde edilen denetleyici yapılarının sistem üzerindeki etkilerinin incelenmesinde, literatürde kullanılan FOMCON programı ara yüzü Matlab/Simulink programında çalıştırılarak sonuçlara ulaşılmış ve karşılaştırmalı olarak verilmiştir.

Son bölümde ise tartışma sonuç ve önerilere yer verilmiştir. Buna göre aşağıda verilen yorumların yapılması mümkündür;

a) Yapılan modelleme çalışmalarında;
   I. Kesir dereceli model yapılarının tamsayı dereceli model yapılarına göre daha başarılı sonuçlar elde ettiğini söylemek mümkündür. Özellikle G_3 süreci için yapılan modelleme çalışmalarında elde edilen veriler, bu sonucu oldukça destekler niteliktedir. Modellemesi zor olan osilasyonlu bu sistemi kesir dereceli model yapısının oldukça başarılı bir şekilde modellediği görülmektedir.
   II. Model parametrelerinin tahmini için kullanılan YAK, DG ve PSO algoritmalarının sonuçlarının birbirine oldukça yakın olduğunu söylemek yanlış bir ifade olmayacaktır. Ancak elde edilen ISE hata değerleri dikkate alındığında YAK ve DG algoritmalarının PSO algoritmasına göre daha başarılı olduğunu söylemek mümkündür. Ayrıca YAK

algoritmasının hata gelişim hızının oldukça iyi olduğu da elde edilen verilerden çıkarılabilecek bir sonuçtur.

III. Algoritmalar için tercih edilen parametre araştırma sınır değerlerinin seçiminin süreçlerin modellenmesinde yeterli olduğu görülmüştür. Sadece G_3 sürecinde parametre sınır değerlerinin artırılmasının sonucu olumlu yönde etkileyebileceği ancak diğer yandan hesaplama süresini oldukça uzatabileceği öngörülmüştür.

IV. Modelleme sürecinde kullanılan ve literatürde sıklıkla tercih edilen ISE hata indisinin performans analizinde yeterli olduğunu söylemek mümkündür.

b) Denetleyici tasarım çalışmalarında;

I. Kesir dereceli denetleyici yapısının tamsayı dereceli denetleyici yapısına göre daha başarılı performans sergilediğini söylemek mümkündür. Neredeyse bütün süreçler için denetleyicilere ait sonuçların algoritmaların tümünde yapılan çalışmalarda daha başarılı ve istenilen sistem cevabı özellikleri sağladığı görülmüştür.

II. YAK, DG ve PSO algoritmalarının performansları genelde birbirine yakın olsa da birbirine üstün oldukları süreçler ve amaç fonksiyonları bulunmaktadır. Her örnek için çalışma sonunda verilen OF_3 amaç fonksiyonuna ait sonuçların verildiği tablolara göre YAK algoritmasının daha başarılı kontrolör tasarımı sonuçları sunduğunu söylemek mümkündür.

III. Çalışmalardan elde edilen başka bir önemli sonuç da, amaç fonksiyonu olarak sadece tek hata indisinin kullanılmasının yeterli faydayı sağlayamadığıdır. ISE, ITSE, IAE ya da ITAE gibi hata indislerinin yanı sıra bunlarla birlikte aynı zamanda aşma miktarı ve oturma süreleri gibi farklı sistem cevabı karakteristiklerinin de yer aldığı farklı amaç fonksiyonlarının kullanımı daha başarılı ve arzu edilir sonuçlar sunmaktadır.

IV. Standart sapma değerleri dikkate alındığında ise DG algoritmasının başarısı dikkat çekicidir. Özellikle bazı örneklerde sıfıra yakın standart sapmanın elde edilmesi algoritmanın başarısını ortaya koymaktadır.

V. Hata gelişim grafiklerine göre değerlendirme yapıldığında, kesir dereceli denetleyici yapılarında bütün algoritmaların oldukça hızlı bir şekilde düşük iterasyon sayılarında hatayı minimize ettiği görülmektedir. YAK algoritmasının bu konuda diğer iki algoritmaya göre biraz daha öne çıktığı da çalışmalardan izlenmiştir.

VI. YAK, DG ve PSO algoritmaları ile elde edilen sonuçların tamamının literatürde yer alan çalışmalara göre daha başarılı olduğu görülmüştür. Özellikle algoritmalarla yapılan T_1, T_2 ve T_3 süreçlerine ait çalışmaların literatürde yer alan klasik yöntemlere göre daha başarılı sonuçlar verdiği Tablo 4.1'de yer alan verilerden açıkça görülmektedir. Malwatkar [112], Wang [113], Ho [112], Kappa-Tau [114], Karimi [114] ve Wang [114]'e ait çalışmalarda elde edilen OF_4 değerleri, tez çalışmasında algoritmalar ile yapılan çalışmalardaki sonuçlardan çok daha yüksektir. Elde edilen bu sonuçlar sezgisel algoritmaların klasik yöntemlere karşı üstünlüğünü açıkça göstermektedir. Ayrıca algoritmalar için seçilen kontrol parametrelerinin de yapılan çalışmalar için uygun olduğu görülmüştür.

VII. Tez çalışmasının dördüncü bölümünde yer alan diğer süreçler için elde edilen sonuçların da literatürden alınan örneklere göre oldukça başarılı oldukları çalışmalarda elde edilen başka bir sonuçtur. T_4 sürecinde elde edilen hata değerleri, Ateş ve arkadaşlarına ait çalışmada [115] elde edilen hata değerlerinden oldukça düşüktür. T_6 sürecinde sonuçların kıyaslandığı Das ve arkadaşları[116] ile Liu ve arkadaşlarına ait çalışmalarda elde edilen OF_3 ve OF_4 değerlerinin de tez çalışmasında elde edilen sonuçlardan daha büyük olduğu Tablo 4.18'de belirtilmiştir. Benzer sonuçlar çalışmanın diğer süreçlerinde de mevcuttur. Tez çalışmasındaki sonuçların, T_7 sürecinde Das ve arkadaşlarının [116], T_8 sürecinde yine Das ve arkadaşlarının [118], T_9 süreci için Maiti [75] ve Zhao'nun [119], T_{10} sürecinde Liu'nun [117], T_{11} sürecinde ise Itik ve arkadaşlarının [120] çalışmasındaki sonuçlardan daha başarılı olduğu elde edilen sonuçlardan görülmektedir.

VIII. Tamsayı dereceli denetleyicilerde üç, kesir dereceli denetleyicilerde beş parametrenin tahmin edilmesi nedeniyle hesaplama süresi kesir dereceli denetleyiciler için biraz daha uzundur. Bu durumun alt programlarda

yapılacak düzenlemeler ile daha da iyileştirilebileceği değerlendirilmiştir.

c) Modellenen yapıların kontrolü çalışmasında;
   I. ISE, ITSE, IAE ve ITAE performans indislerinin başarılı sonuçlar sunduğu ancak ISE ve ITSE indislerinin kullanıldığı çalışmalarda yüksek aşma miktarının meydana geldiği görülmüştür. Bu nedenle ITAE ya da IAE performans indislerinin kullanımının aşmanın istenmediği durumlarda kullanılmasının daha başarılı sonuçlar verebildiği değerlendirilmiştir.
   II. Bu çalışma için kullanılan FOMCON programı arayüzünün, optimizasyon algoritmalarına göre daha yavaş çalıştığı ve geliştirilmesi gerektiği düşünülmektedir.

Bu tez çalışması, algoritmalara dayalı olarak elde edilen sonuçlara göre modelleme ve denetleyici tasarımı gibi farklı kontrol problemlerinde bu algoritmaların sergileyeceği performans konusunda önemli bilgiler vermektedir. Yapılan bu çalışmanın gelecekte yapılacak çalışmalara fikir sağlaması açısından dikkate değer ve önemli sonuçlar içerdiği düşünülmektedir. Özellikle literatürde son yıllarda yer alan kesir dereceli sistemler ile çok daha uzun süredir çalışmalara konu olan tamsayı dereceli sistemlerin farklılıkları, avantajları ve dezavantajları gibi konularda birçok örnek için elde edilen sonuçların araştırmacılara değerli ve önemli katkılar sağlayacağı ortadadır.

Tez çalışmasında T_1, T_2 ve T_3 süreci için elde edilen sonuçlar SCI-E (Science Citation Index Expanded) kapsamında yer alan Elektronika Ir Elektrotechnika dergisinde yayınlanmıştır. Ayrıca G_1 ve G_2 süreçlerinin modellenmesi ve kontrolör tasarımı sonrası elde edilen sonuçlar yine aynı dergide yayınlanmak üzere kabul edilmiştir [121]. T_4 ve T_5 süreçlerinde elde edilen sonuçlar TOK 2017 (Otomatik Kontrol Ulusal Toplantısı 2017) konferansında, kesir dereceli süreçler için kontrolör tasarımının gerçekleştirildiği T_9, T_{10} ve T_{11} süreçleri kapsayan çalışmalar 2017 Electronics konferansında sözlü bildiri olarak sunulmuştur.

Gelecek çalışmalarda farklı model ve denetleyici yapılarının farklı algoritmalar ile kullanımının sağlayacağı fayda veya avantajların araştırılması, teorik olarak elde edilen sonuçların pratik hayata geçirilmesi üzerinde durulması planlanmaktadır.

KAYNAKÇA

1. Söderström T., Stoica P., 1989, System Identification, Prentice-Hall.
2. Ljung L., 1987, System Identification: Theory for The User, Prentice Hall.
3. Åström K.J., Hägglund T., 2004. Revisiting the Ziegler–Nichols step response method for PID control. **Journal of Process Control, 14**: 635–650.
4. Şenberber H., Bağiş A., 2012. Kararsız sistemlerin model parametrelerinin yapay arı kolonisi algoritması kullanılarak belirlenmesi, ss.812-816. *Otomatik Kontrol Türk Milli Komitesi 2012 Ulusal Toplantısı(TOK 2012)*, Niğde, Türkiye, 11-13 Ekim 2012.
5. Wang Q. G., Lee T. H., Fung H. W., Bi Q., Zhang Y., 1999. PID tuning for improved performance. **IEEE Transactions on Control Systems Technology, Vol. 7**, (4): 457-465.
6. S. Mukherjee, Satakshi, Mittal R.C., 2007. Linear time invariant system order reduction using multipoint step response matching. **International Journal of Systems Science 38**(3): 211-217.
7. Bagis A., Senberber H., 2016. Modeling of higher order systems using artificial bee colony algorithm. **An International Journal of Optimization and Control: Theories & Applications, Vol.6**,(2):129-139.
8. Bağış A.,2011. Tabu search algorithm based PID controller tuning for desired system specifications. **Journal of the Franklin Institute, Vol. 348**: 2795–2812.
9. Ogata K., 1987. Modern control systems, University of Minnesota, Prentice Hall.
10. Podlubny I., Dorcak L., Kostial I., 1997. On fractional derivatives, fractional-order dynamic systems and $PI^{\lambda}D^{\mu}$ controllers, pp.4985-4990. *In: Proceedings of the 36th Conference on Decision & Control*.
11. Podlubny I. 1999. Fractional-order systems and $PI^{\lambda}D^{\mu}$ controllers, **IEEE Transactions on Automatic Control 44**(1): 208–214.
12. Vinagre B. M., Podlubny I., Dorcak L, Feliu V., 2000. On fractional PID controllers: A frequency domain approach, pp. 53–58. *IFAC workshop on digital control. Past, present and future of PID control*, Terrasa, Spain.
13. Podlubny I., 1999. Fractional Differential Equations, Academic Press, San Diego, California.

14. Lokenath D., 2006. A brief historical introduction to fractional calculus. **International Journal of Mathematical Education in Science and Technology, 35(**4): 487-501.
15. Şenberber H, Bağış A., Yüksek mertebeli sistemler için yapay arı kolonisi algoritması kullanarak kesir dereceli PID denetleyici tasarımı, ss. 201-205. *Otomatik Kontrol Ulusal Toplantısı* (*TOK2017)*, İSTANBUL.
16. Podlubny, I. J., 1994. Fractional–order systems and fractional order controllers. Tech. Rep. UEF-03-94, Slovak Academy of Sciences, Institute of Experimental Physics, Department of Control Engineering, Faculty of Mining, University of Technology, Kosice.
17. Oldham K. B., Spanier J., 2006. The Fractional Calculus: Theory and Applications of Differentiation and Integration to Arbitrary Order, Dover Books on Mathematics.
18. Oustaloup A.,1995. La Commande CRONE: Commande Robuste d'Ordre Non Entier, Hermes, Paris.
19. Valério D, Costa J. Sá. 2004. Ninteger: a non-integer control toolbox for MatLab, pp. 208-213. *Proceedings of the First IFAC Workshop on Fractional Differentiation and Applications*, Bordeaux, France.
20. Tepljakov A., Petlenkov E., Belikov J., 2011. FOMCON: Fractional-Order Modeling and Control Toolbox for MATLAB, pp. 684-689. *18th International Conference Mixed Design of Integrated Circuits and Systems(MIXDES 2011)*, June 16-18, 2011, Gliwice, Poland.
21. Nagar S. K., Singh S. K., 2004, An algorithmic approach for system decomposition and balanced realized model reduction, **Journal of Franklin Inst., Vol. 341**:. 615-630.
22. Bağış A., 2010. Yüksek mertebeli osilasyonlu sistemlerin modellenmesinde parçacık sürüsü optimizasyonu performansının incelenmesi, ss. 168-173. *Otomatik Kontrol Ulusal Toplantısı (TOK'2010*), 21-23 Eylül 2010, Gebze, KOCAELİ.
23. Gutman P.O., Mannerfelt C.F., and Molander P., 1982. Contributions to the model reduction problem, **IEEE Trans. Automat. Control, Vol. AC-27**(2):454-455.

24. Luitel B., Venayagamoorthy G.K., 2010. Particle swarm optimization with quantum infusion for system identification. **Engineering Applications of Artificial Intelligence, Vol 23**:635-649.
25. Zorlu, H., Özer Ş., 2009. Identification of nonlinear systems by using clonal selection algorithm, *IEEE 17th Signal Processing and Communications Applications Conference (SIU-2009)*, Proceedings CD, Antalya, Turkey, (in Turkish).
26. Coelho L. S., Pessoa M. W., 2009. Nonlinear model identification of an experimental ball-and-tube system using a genetic programming approach. **Mechanical Systems and Signal Processing, 23**(5): 1434–1446.
27. Wang, L., Langari, R., 1996. Complex systems modeling via fuzzy logic. **IEEE Transactions on Systems, Man, And Cybernetics. Part B, Cybernetics: A Publication of the IEEE Systems, Man, and Cybernetics Society, 26**(1): 100- 106.
28. Bagis, A., 2008. Fuzzy rule base design using tabu search algorithm for nonlinear system modeling. **ISA Transactions, 47**(1): 32-44.
29. Ozer, S., Zorlu, H., 2011. Identification of bilinear systems using differential evolution algorithm", **Sadhana-Academy Proceedings in Engineering Sciences, Vol. 36** (3): 281–292.
30. Şenberber H., Bağış A., 2012, Gecikmeli sistemlerin modellenmesinde yapay arı kolonisi algoritmasının performansının incelenmesi, ss. 164-168. *Akıllı Sistemlerde Yenilikler ve Uygulamaları Sempozyumu (ASYU 2012),* Trabzon.
31. Şenberber H. 2012, Yapay Ari Kolonisi Algoritmasinin Sistem Modelleme Performansinin İncelenmesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, Kayseri.
32. Mete S., Zorlu H., Ozer Ş., 2016. System identification using Hammerstein model optimized with differential evolution algorithm. **AEU - International Journal of Electronics and Communications 70** (12): 1667-1675.
33. Chaudhary N.I., Raja M.A.Z., 2015. Design of fractional adaptive strategy for input nonlinear Box–Jenkins systems. **Signal Process 116**: 141-151.
34. Raja M.A.Z., Chaudhary N.I., 2015. Two–stage fractional least mean square identification algorithm for parameter estimation of CARMA systems. **Signal Process 107**: 327-339.

35. Ming Z., Dazi L., 2012. Fractional System Identification Based on Improved NLJ Algorithm, pp. 1057-1061. *Control and Decision Conference (CCDC)*, 24th Chinese 2012.
36. Sabatier J., Aoun M., Oustaloup A., Grégoire G., Ragot F., Roy P., 2006. Fractional system identification for lead acid battery state of charge estimation. **Signal Processing, 86(**10): 2645-2657.
37. Tepljakov A., Petlenkov E., Belikov J., 2011. FOMCON: a MATLAB Toolbox for fractional-order system identification and control. **International Journal and Microelectronics and Computer Science, 2**(2): 51-62.
38. Cois O., Oustaloup A., Poinot T., Battaglia J.-L., 2001. Fractional state variable filter for system identification by fractional model, pp. 2484-2486. *European Control Conference(ECC)*, Portugal.
39. Malti R., Victor S., Oustaloup A., Garnier H., 2008. An optimal instrumental variable method for continuous-time fractional model identification, pp. 14379-14384. *IFAC Proceedings Vol.* 41(2).
40. Tran H.D., Guan Z.H., Dang X.K., Cheng X.M., Yuan F.S., 2013. A normalized PID controller in networked control systems with varying time delays. **ISA Transactions, Vol. 52**, pp. 592-599, 2013.
41. Li K., 2013. PID tuning for optimal closed-loop performance with specified gain and phase margins. **IEEE Transactions on Control Systems Technology, Vol. 21**,(3): 1024-1030.
42. Ang K.H., Chong G., Yun L.Y., 2005. PID control system analysis, design and technology. **IEEE Transactions on Control Systems Technology, Vol. 13**(4):559-576.
43. Papadopoulos K.G., Tselepis N.D., Margaris N.I., 2013. Type-III closed loop control systems-digital PID controller design. **Journal of Process Control, Vol. 23**: 1401-1414.
44. Paraskevopoulos P.N., Pasgianos G.D., Arvanitis K.G., 2006. PID-type controller tuning for unstable first order plus dead time processes based on gain and phase margin specifications. **IEEE Transactions on Control Systems Technology, Vol. 14**(5): 926-936.

45. Bağış A., 2007. Determination of the PID controller parameters by modified genetic algorithm for improved performance. **Journal of Information Science and Engineering, Vol. 23**:pp. 1469-1480.

46. Taeib A., Ltaief A., Chaari A., 2013. PID control based on modified particle swarm optimization for nonlinear process, pp. 1-5. *Computer and Information Technology (WCCIT)*, 2013 World Congress.

47. Theja B.S., Kothari D.P., Das S., 2012. Design of PID controller based power system stabilizer using modified Philip-Heffron's Model: An artificial bee colony approach, pp. 606-611. *Communication Systems and Network Technologies (CSNT 2012*) International Conference.

48. Farahani M., Ganjefar S., Alizadeh M., 2012. PID controller adjustment using chaotic optimization algorithm for multi-area load frequency control. **IET Control Theory Appl., Vol. 6**,(13): 1984–1992.

49. Gao F., Qi Y., Yin Q., Xiao J., 2010. An novel optimal PID tuning and on–line tuning based on artificial bee colony algorithm, pp 1-4. *Computational Intelligence and Software Engineering (CiSE, 2010*) International Conference.

50. Ibrahim H.E.A., Hassan F.N., Shomer A.O., 204. Optimal PID control of a brushless DC motor using PSO and BF techniques. **Ain Shams Engineering Journal, 5**(2): 391-398.

51. Hasanien H.M., 2013. Design optimization of PID controller in automatic voltage regulator system using taguchi combined genetic algorithm method **IEEE System Journal, Vol. 7(**4): 825 – 831.

52. Hwang H.S., Choi J.N., Lee W.H., Kim J.K., 1999. A tuning algorithm for the PID controller utilizing fuzzy theory, pp: 2210 – 2215. *Neural Networks IJCNN '99*. International Conference, Vol. 4.

53. Tien J.P., Li T.H.S., 2012. Hybrid clonal selection algorithm and the artificial bee colony algorithm for a variable PID-like fuzzy controller design, pp. 87-94. *Proceedings of 2012 International Conference on Fuzzy Theory and Its Applications*.

54. Chen J.,Ren Z., Fan X., 2006. Particle swarm optimization with adaptive mutation and its application research in tuning of PID parameters. *Systems and Control in Aerospace and Astronautics, ISSCAA 2006. 1st International Symposium*.

55. Ho S.J., Shu L.S., Ho S.Y., 2006. Optimizing fuzzy neural networks for tuning PID controllers using an orthogonal simulated annealing algorithm OSA. **IEEE Transactions on Fuzzy Systems, Vol. 14(**3): 421-434.

56. Abachizadeh M., Yazdi M.R.H., Koma A.Y., 2010. Optimal tuning of PID controllers using artificial bee colony algorithm, pp. 379-384. *2010 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*.

57. Karaboga D., Akay B., 2010. Proportional-integral-derivative controller design by using artificial bee colony, harmony search, and the bees algorithms. **Proc. IMechE Part I: J. Systems and Control Engineering, vol.224**, pp.869-883.

58. Korkmaz M., Aydoğdu O., Doğan H., 2012. Design and performance comparison of variable parameter nonlinear PID controller and genetic algorithm based PID controller, pp. 1-5. *Innovations in Intelligent Systems and Applications (INISTA), 2012 International Symposium*.

59. Neath M.J., Swain A.K., Madawala U.K., Thrimawithana D.J., 2014. An optimal PID controller for a bidirectional inductive power transfer system using multiobjective genetic algorithm. **IEEE Transactions on Power Electronics, Vol. 23(**3): 1523-1531.

60. Yadaiah N., Malladi S., 2013. An optimized relation between Ti and Td in modified Ziegler Nichols PID controller tuning, pp. 1275-1280. *IEEE International Conference on Control Applications (CCA) Part of 2013 IEEE Multi-Conference on Systems and Control Hyderabad.*

61. Erçin O., Çoban R., 2011. Comparison of the artificial bee colony and the bees algorithm for PID controller tuning. pp. 595-598. *Innovations in Intelligent Systems and Applications (INISTA), 2011 International Symposium*.

62. Wai R.J., Lee J.D., Chuang K.L., 2011. Real-time PID control strategy for maglev transportation system via particle swarm optimization. **IEEE Transactions on Industrial Electronics, Vol. 58(**2): 629-646.

63. Luo Y., Zhang J., Li X., 2007. The optimization of PID controller parameters based on artificial fish swarm algorithm, pp. 1058-1062. *Proceedings of the IEEE International Conference on Automation and Logistics*.

64. Hsiao Y.T., Chuang C.L., Chien C.C., 2004. Ant colony optimization for designing of PID controllers, pp. 321-326. *IEEE lnternational Symposium on Computer Aided Control Systems Design*.

65. Gaing Z.L., 2004. A particle swarm optimization approach for optimum design of PID controller in AVR system. **IEEE Transactions on Industrial Electronics, Vol. 19(**2):384-391.
66. Mukherjee V., Ghoshal S.P., 2007. Intelligent particle swarm optimized fuzzy PID controller for AVR system. **Electric Power Systems Research, Vol. 77:** 1689-1698.
67. El-Zonkoly A.M., Khalil A.A., Ahmied N.M., 2009. Optimal tunning of lead-lag and fuzzy logic power system stabilizers using particle swarm optimization. **Expert Systems with Applications, Vol. 36**,: 2097-2106.
68. Wang D.J., 2009. Synthesis of phase-lead/lag compensators with complete information on gain and phase margins. **Automatica, Vol. 45**: 1026-1031.
69. Tan N., 2003. Computation of stabilizing Lag/Lead controller parameters. **Computers and Electrical Engineering, Vol. 29**: 835-849.
70. Bagis A., Senberber H., 2017. ABC algorithm based PID controller design for higher order oscillatory systems, **Elektronika Ir Elektrotechnika, Vol. 23**(6):3-9.
71. Saad M.S., Jamaluddin H., Darus I.Z.M., 2012. Implementation of PID controller tuning using differential evolution and genetic algorithms. **Int. Journal of Innovative Computing, Information and Control, Vol.8**: 7761-7779.
72. Chang W.D., Shih S.P., 2010. PID controller design of nonlinear systems using an improved particle swarm optimization approach. **Communications in Nonlinear Science and Numerical Simulation, Vol.15**: 3632–3639.
73. Zhao J., Li T., Qian J., 2005. Application of particle swarm optimization algorithm on robust PID controller tuning. **Lecture Notes in Computer Sciences, Vol.3612**: 948-957.
74. Jin, J. Y., Ryu K. H., Sung S. W., Lee J., Lee I., 2014. PID auto-tuning using new model reduction method and explicit PID tuning rule for a fractional order plus time delay model. **Journal of Process Control, Vol. 24**: 113–128.
75. Maiti D., Biswas S., Konar A., 2008. Design of a fractional order PID controller using particle swarm optimization technique, pp. 457-462. *2nd National Conference on Recent Trends in Information Systems (ReTIS-08)*.

76. Bouafoura M. K., Braiek N. B., 2010. PI$^{\lambda}$D$^{\mu}$ controller design for integer and fractional plants using piecewise orthogonal functions. **Commun Nonlinear Sci Numer Simulat, Vol 15**: 1267–1278.

77. Das S., Pan I., Das S, 2013. Performance comparison of optimal fractional order hybrid fuzzy PID controllers for handling oscillatory fractional order processes with dead time. **ISA Transactions, Vol. 52**: 550–566.

78. Debbarma S., Saikia L.C., Sinha N., 2014. Automatic generation control using two degree of freedom fractional order PID controller. **Electrical Power and Energy Systems, Vol. 58**: 120–129.

79. Luo Y., Chen C.Q., Wang C.Y., Pi Y.G., 2010. Tuning fractional order proportional integral controllers for fractional order systems. **Journal of Process Control,Vol. 20**: 823–831.

80. Luo Y., Chen Y. Q., 2009. Fractional order [proportional derivative] controller for a class of fractional order systems. **Automatica, Vol. 45**: 2446-2450.

81. Malek H., Luo Y., Chen Y. Q., 2013. Identification and tuning fractional order proportional integral controllers for time delayed systems with a fractional pole. **Mechatronics, Vol. 23**: 746–754.

82. Moradi M., 2014. A genetic-multivariable fractional order PID control to multi-input multi-output processes. **Journal of Process Control, Vol. 24**: 336–343.

83. Yeroğlu C., Tan N., 2011. Classical controller design techniques for fractional order case. **ISA Transactions, Vol. 5**: 461-472.

84. Ozbay H., Bonnet C., Fioravanti A. R., 2012. PID controller design for fractional-order systems with time delays. **Systems & Control Letters, Vol. 61:** 18–23.

85. Padula F., Visioli A., 2011. Tuning rules for optimal PID and fractional-order PID controllers. **Journal of Process Control, Vol. 21**: 69–81.

86. Rajasekhar A., Jatoth R. K., Abraham A., 2014. Design of intelligent PID/ PI$^{\lambda}$D$^{\mu}$ speed controller for chopper fed DC motor drive using opposition based artificial bee colony algorithm. **Engineering Applications of Artificial Intelligence, Vol. 29**: 13–32.

87. Yeroğlu C., Ateş A., 2014. A stochastic multi-parameters divergence method for online auto-tuning of fractional order PID controllers. **Journal of the Franklin Institute, Vol. 351**: 2411–2429.

88. Senberber H., Bagis A., 2017. Fractional PID controller design for fractional order systems using ABC algorithm, pp. 1-7. *2017 Electronics*, Palanga, Lithuania, June 19th–21th.
89. MathWorks, Matlab&Simulink Getting Started Guide. R2014a. The MathWorks, Inc., 2014.
90. Karaboga, D., 2005, An idea based on honey bee swarm for numerical optimization, Tecnical Report TR06, Erciyes University, Engineering Faculty, Computer Engineering Department.
91. Karaboga, D., Oztürk, C., 2011, A novel clustering approach: Artificial Bee Colony (ABC) algorithm. **Applied Mathematics and Computation , Vol. 11**: 652–657.
92. Karaboga , D., 2010. Artificial Bee Colony Algorithm. **Scholarpedia, 5**(3):6915.
93. Karaboga, D., Akay, B., 2009. A comparative study of Artificial Bee Colony algorithm. **Applied Mathematics and Computation ,Vol 214:** 108–132.
94. Storn R., 1999. System design by constraint adaptation and differential evolution," **IEEE Transactions on Evolutionary Computation Vol. 3:** 5–9.
95. Storn R. and Price K., 1997. Differential evolution a simple and efficient heuristic for global optimization over continuous spaces. **Journal of Global Optimization 11:** 41–59.
96. Kyprianou A., Worden K., Panet M., 2001. Identification of hysteretic systems using the differential evolution algorithm. **Journal of Sound and Vibration 248**(2): 289–314.
97. Kennedy, J., Eberhart, R., 1995. Particle swarm optimization. *In Proceedings of IEEE Internatiol Conference on Neural Networks*, pp. 1942-1948.
98. He, J., Guo, H., 2013. A modified particle swarm optimization algorithm. **Telkomnika, 11** (10): 6209-6215.
99. Caponetto R., Dongola G., Fortuna L., Petras I., 2010. Fractional Order Systems Modeling and Control Applications, World Scientific Publishing Co. Pte. Ltd.
100. Aguila-Camacho N., Duarte-Mermoud M.A., 2013. Fractional adaptive control for an automatic voltage regulator, **ISA Transactions Vol. 52(**6): 807–815.
101. Murio D.A., 2009. Stable numerical evaluation of Grünwald–Letnikov fractional derivatives applied to a fractional IHCP. **Inverse Problems in Science and Engineering Vol. 17**(2): 229–243.

102. Ziegler J.G., Nichols N.B., 1942. Optimum settings for automatic controllers. **Transactions of ASME, Vol. 64:** 759–768.

103. Cohen G.H., Coon G.A., 1953. Theoretical consideration of retarded control, **Transactions of the ASME Vol. 75:** 27–834.

104. Zhuang M., Atherton D.P., 1993. Automatic tuning of optimum PID controllers. **IEE Proceedings on Control and Applications 140**(3): 216–224.

105. Pessen D.W., 1994. A new look at PID-controller tuning, **Transactions of the ASME, Journal of Dynamical Systems Measures and Control 116:** 553–557.

106. Ho W.K., Hang C.C., Cao L.S., 1995. Tuning of PID controllers based on gain and phase margin specifications. **Automatica 31**(3): 497–502.

107. Yüksel İ., 2011. Otomatik Kontrol Sistem Dinamiği Ve Denetim Sistemi. Nobel Yayınları.

108. Puangdownreong D., Sujitjorn S., 2007. Obtanining an optimum PID controller via adaptive tabu search, pp. 747–755. *ICANNGA, LNCS*, vol. 4432, Warsaw, Poland, April 11-14, 2007.

109. Srivastava S., Misra A., Thakur S.K., Pandit V.S., 2016. An Optimal PID controller via LQR for standard second order plus time delay systems. **ISA Transactions 60**: 244–253.

110. Xue C.Z.D., Chen Y., 2005. A Fractional order PID tuning algorithm for a class of fractional order plants. *Proceedings of the IEEE International Conference on Mechatronics & Automation,* Niagara Falls, Canada.

111. Liu L., Pan F., Xue D., 2015. Variable-order fuzzy fractional PID controller. **ISA Transactions, Vol. 55**: 227–233.

112. Malwatkar G.M., Sonawane S.H., Waghmare L.M., 2009. Tuning PID controllers for higher-order oscillatory systems with improved performance. **ISA Transactions, Vol.48**: 347–353.

113. Wang Q.G., Lee T.H., Fung H.W., Bi Q., Zhang Y., 1999. PID tuning for improved performance. **IEEE Transactions on Control Systems Technology, Vol.7**:457-465.

114. Karimi A., Garcia D., Longchamp R., 2003. PID controller tuning using Bode's integrals. **IEEE Transactions on Control Systems Technology, Vol.11**:812-821.

115. Ateş A., Yeroglu C., 2016. Optimal fractional order PID design via tabu search based algorithm. **ISA Transactions, Vol.60**:109–118.

116. Das S., Pan I., Das S., Gupta A., 2012. Improved model reduction and tuning of fractional order PID controllers for analytical rule extraction with genetic programming. **ISA Transactions, Vol. 51**: 237-261.

117. Liu L., Pan F., Xue D., 2015. Variable-order fuzzy fractional PID controller. **ISA Transactions, 55**: 227–233.

118. Das S., Saha S., Das S., Gupta A., 2011. On the selection of tuning methodology of FOPID controllers for the control of higher order processes. **ISA Transactions 50**(3): 376–388.

119. Zhao C., Xue D., Chen Y., 2005. A fractional order PID tuning algorithm for a class of fractional order plants, pp.216-221. *Proceedings of the IEEE International Conference on Mechatronics & Automation* Niagara Falls, Canada, July 2005.

120. Itik M., Sahin E., Ayas M.S., 2015. Fractional order control of conducting polymer artificial muscles. **Expert Systems with Applications, vol.42**: 8212–8220.

121. Senberber H., Bagis A., ABC Algorithm based System Modelling for Tuning the Fractional Order PID Controllers, **Elektronika Ir Elektrotechnika** (Kabul edilmiştir)**.**

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı, Soyadı	: Halit ŞENBERBER
Uyruğu	: Türkiye Cumhuriyeti
Doğum Tarihi ve Yeri	: 01 Haziran 1986, Kayseri
Medeni Durumu	: Evli
Telefon	: +90 352 207 66 66
Faks	: +90 352 437 37 84
Email	: halitsenberber@erciyes.edu.tr
Yazışma Adresi	: Erciyes Üniversitesi, Mühendislik Fakültesi, Elektrik-Elektronik Mühendisliği Bölümü, 38039, Melikgazi/KAYSERİ

EĞİTİM

Derece	Kurum	Mezuniyet Tarihi
Yüksek Lisans	ERÜ Fen Bilimleri Enstitüsü	2012
Lisans	ERÜ Mühendislik Fak., Elektrik-Elektronik Müh.	2009
Lise	Sami Yangın Anadolu Lisesi, Kayseri	2004

İŞ DENEYİMLERİ

Yıl	Kurum	Görev
2010- Halen	ERÜ Mühendislik Fak., Elektrik-Elektronik Müh	Araştırma Görevlisi

YABANCI DİL

İngilizce

YAYINLAR

SCI İndekslerine Giren Dergilerde Yayınlanan Makaleler

1. Senberber H, Bagis A. ABC Algorithm based System Modelling for Tuning the Fractional Order PID Controllers. **Elektronika Ir Elektrotechnika**, (**05.04.2018 tarihinde kabul edildi**).
2. Bagis A., Senberber H., 2017. ABC Algorithm Based PID Controller Design for Higher Order Oscillatory Systems. **Elektronika Ir Elektrotechnika, 23**(6): 3-9.

Diğer Dergilerde Yayımlanan Makaleler

1. Bagis A., Senberber H., 2016. Modeling of higher order systems using artificial bee colony algorithm. **An International Journal of Optimization and Control: Theories & Applications, 6**(2): 129-139.

Hakemli Kongre / Sempozyumların Bildiri Kitaplarında Yer Alan Yayınlar

1. Senberber H., Bagis A., 2017. Fractional PID controller design for fractional order systems using ABC algorithm", *2017 Electronics*, LITHUANIAN.

2. Şenberber H., Bağiş A., 2017. Yüksek Mertebeli Sistemler için Yapay Arı Kolonisi Algoritması Kullanarak Kesir Dereceli PID Denetleyici Tasarımı. *Otomatik Kontrol Ulusal Toplantısı 2017*, İstanbul, TÜRKIYE, 21-23 Eylül 2017, ss.201-205.

3. Şenberber H., Bağiş A., 2012. Kararsız Sistemlerin Model Parametrelerinin Yapay Arı Kolonisi Algoritması Kullanılarak Belirlenmesi. *Otomatik Kontrol Türk Milli Komitesi 2012 Ulusal Toplantısı(TOK 2012)*, Niğde, TÜRKİYE, 11-13 Ekim 2012, cilt.2, ss.812-816.

4. Şenberber H., Bağiş A., 2012. Gecikmeli Sistemlerin Modellenmesinde Yapay Arı Kolonisi Algoritmasının Performansının İncelenmesi. *Akıllı Sistemlerde Yenilikler ve Uygulamaları Sempozyumu (ASYU),* Trabzon, TÜRKİYE, 3-4 Temmuz 2012, ss.164-168.