

**T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ**

**ÖLÜ ZAMANLI SİSTEMLERİN
MODELLENMESİ VE
DENETLENMESİ**

YÜKSEK LİSANS TEZİ

Murat Erhan ÇİMEN

**Enstitü Anabilim Dalı : ELEKTRİK ELEKTRONİK
MÜHENDİSLİĞİ**
Tez Danışmanı : Prof. Dr. Ali Fuat BOZ

Ocak 2018

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ÖLÜ ZAMANLI SİSTEMLERİN
MODELLENMESİ VE
DENETLENMESİ

YÜKSEK LİSANS TEZİ

Murat Erhan ÇİMEN


Enstitü Anabilim Dalı : ELEKTRİK ELEKTRONİK
MÜHENDİSLİĞİ

Bu tez 08./01./2019 tarihinde aşağıdaki jüri tarafından oybirliği / oyçokluğu ile kabul edilmiştir.

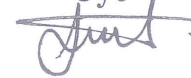
Prof. Dr.
Ali Fuat BOZ
Jüri Başkanı



Doç. Dr.
İrfan YAZICI
Üye



Yrd. Doç. Dr.
Mustafa DURSUN
Üye



BEYAN

Tez içindeki tüm verilerin akademik kurallar çerçevesinde tarafımdan elde edildiğini, görsel ve yazılı tüm bilgi ve sonuçların akademik ve etik kurallara uygun şekilde sunulduğunu, kullanılan verilerde herhangi bir tahrifat yapılmadığını, başkalarının eserlerinden yararlanılması durumunda bilimsel normlara uygun olarak atıfta bulunulduğunu, tezde yer alan verilerin bu üniversite veya başka bir üniversitede herhangi bir tez çalışmasında kullanılmadığını beyan ederim.

Murat Erhan ÇİMEN

19.12.2017

TEŐEKKÜR

Yüksek lisans eğitiminin boyunca değerli bilgi ve deneyimlerinden yararlandığım, her konuda bilgi ve desteğini almaktan çekinmediğim, araştırmanın planlanmasından yazılmasına kadar tüm aşamalarında yardımlarını esirgemeyen, teşvik eden, aynı titizlikte beni yönlendiren değerli danışman hocam Prof. Dr. Ali Fuat BOZ'a teşekkürlerimi sunarım.

Her zaman maddi manevi destek vererek beni bugünlere getiren annem Ayfer ÇİMEN'e, babam Fadıl ÇİMEN'e, kardeşlerime ve yeğenime, her konuda manevi desteklerini esirgemeyen arkadaşlarıma sevgilerimi sunarım.

Ayrıca bu çalışmanın maddi açıdan desteklenmesine olanak sağlayan Sakarya Üniversitesi Bilimsel Araştırma Projeleri (BAP) Komisyon Başkanlığına (Proje No: 2017-50-01-026) teşekkür ederim.

İÇİNDEKİLER

TEŞEKKÜR.....	i
İÇİNDEKİLER	ii
SİMGELER VE KISALTMALAR LİSTESİ.....	iv
ŞEKİLLER LİSTESİ	vi
TABLolar LİSTESİ	x
ÖZET.....	xiii
SUMMARY	xiv
BÖLÜM 1.	
GİRİŞ	1
1.1. Literatür Taraması.....	1
1.2. Tezin Amacı.....	12
1.3. Tezin Planı.....	12
BÖLÜM 2.	
OPTİMİZASYON ALGORİTMALARI.....	14
2.1. Parçacık Sürü Optimizasyonu	19
2.2. Guduk Kuşu Algoritması.....	21
2.3. Ateş Böceği Algoritması	23
BÖLÜM 3.	
ÖLÜ ZAMANLI SİSTEMLERİN MODELLENMESİ.....	26
3.1. Auto-tuning Ayar Tekniği.....	26
3.1.1. Tek röle testi	30
3.1.2. Çift röle testi.....	34
3.1.3. Ölü zamanlı röle testi	37
3.2. Optimizasyon Algoritmaları ile Parametre Optimizasyonu	38

3.3. Farklı Yöntemler	39
3.3.1. Fourier dönüşümü	39
3.3.1.1. Fourier serisi(limit cycle).....	39
3.3.1.2. Fourier serisi	40
3.3.2. En küçük kareler yöntemi	41
BÖLÜM 4.	
ÖLÜ ZAMANLI SİSTEMLERİN KONTROLÜ	44
4.1. Zeigler-Nichols Yöntemi.....	44
4.1.1. Basamak cevabı yöntemi.....	44
4.1.2. Frekans cevabı yöntemi.....	45
4.2. AMIGO Yöntemi	46
4.3. Smith Öngörücüsü.....	47
4.4. Optimizasyon Algoritmaları ile Denetleyici Katsayılarının Belirlenmesi	49
BÖLÜM 5.	
ÖLÜ ZAMANLI SİSTEMLERİN MODELLENMESİ VE DENETİMİ İÇİN TOOLBOX GELİŞTİRME	51
5.1. Parodta Arayüzü	51
5.1.1. Parametre optimizasyon arayüzü	53
5.1.2. Denetleyici tasarım arayüzü	58
BÖLÜM 6.	
SONUÇLAR	61
6.1. Analiz Sonuçları.....	61
6.1.1. Parametre optimizasyon sonuçları	62
6.1.2. Denetleyici tasarım sonuçları	67
6.2. Tartışma ve gelecek çalışmalar	106
KAYNAKLAR.....	109
EKLER	115
ÖZGEÇMİŞ	123

SİMGELER VE KISALTMALAR LİSTESİ

ATV	: Yaklaşık Transfer Fonksiyonu Yöntemi
CS	: Gugukkuşu Algoritması
d,a	: Genlik
Eİ	: En iyi Değer
F	: Eğrilik Fonksiyonu
FA	: Ateşböceği Algoritması
FFT	: Hızlı Fourier Dönüşümü
$G(i\omega)$: Frekans Cevabı
$G(s)$: Transer Fonksiyonu
IAE	: Mutlak Hataların Integrali
ISE	: Karesel Hataların Integrali
ITAE	: Mutlak Hataların Zamanla Çarpımının Integrali
ITSE	: Karesel Hataların Zamanla Çarpımının Integrali
K_c	: Kritik Kazanç
KH	: Kestirim Hatası
LSE	: En Küçük Kareler Yöntemi
N	: Rölenin Fonksiyonu
P_a	: Guguk Kuşu Algoritmasının Levy Flight Parametresi
PSO	: Parçacık Sürü Optimizasyonu
$r(t),u$: Referans İşareti
T_c	: Periyot
Y	: Ölçülen Çıkış Dizisi
X_i, Y_i	: Sonucun Reel ve Sanal Kısımları
Z	: Hesaplanan Çıkış Dizisi
ZOH	: Zero Order Holder Birinci Dereceden Tutucu
a_n, b_n, a_0	: Fourier Katsayıları

α	: Ateş Böceği Gauss Rastlantı Katsayısı
c_1	: PSO Korelasyon Parametresi
c_2	: PSO Korelasyon Parametresi
n	: Açılım Katsayısı
β_0	: Ateş böceği Çekicilik Parametresi
Δ	: Histerisiz Değeri
φ	: Değişken Dizisi
θ	: Bilinmeyen Değişken Dizisi
$\hat{\theta}$: Kestirilen Değişken Disizi
γ	: Ateş Böceği Işık Emilim Parametresi
ω	: Frekans
ω_c	: Kritik Frekans
w	: PSO Ağırlık Parametresi
Φ	: Değişken Matrisi
z	: En Küçük Kareler için Çıkış değeri

ŞEKİLLER LİSTESİ

Şekil 2.1. Bir fonksiyonda global ve lokal minimum noktalar.....	15
Şekil 2.2. Açık döngü kontrol sistemi.....	16
Şekil 2.3. Geri beslemeli kontrol sistemi.....	17
Şekil 2.4. ISE, IAE, ITAE, ITSE, ITSE2, ISE2, IAE2, ITAE2, ITSE2 ve IT2SE2'nin hesaplanması.....	19
Şekil 2.5. Parçacık sürü optimizasyonu [56].....	20
Şekil 2.6. Cuckoo search algoritması [31].....	22
Şekil 2.7. Ateş böceği algoritması [55].....	25
Şekil 3.1. Birinci dereceden bir sistemin kritik kazanç ve kritik frekansı.....	27
Şekil 3.2. Kapalı döngü sistem.....	28
Şekil 3.3. Zeigler Nichols yöntemine göre osilasyon cevabı.....	28
Şekil 3.4. Röle tekniği ile sistemi ile kapalı döngü sistem.....	29
Şekil 3.5. Röle tekniği ile kapalı döngü sistem cevabı.....	29
Şekil 3.6. Nyquist eğrisi.....	30
Şekil 3.7. Röle yapısı.....	30
Şekil 3.8. Sinüs sinyaline karşılık röle çıkışı.....	31
Şekil 3.9. Fourierin serisinin açılım etkisi.....	33
Şekil 3.10. Röle testinin uygulanması.....	34
Şekil 3.11. Tek Röle ve çift rölenin tanıma noktaları.....	35
Şekil 3.12. Çift rölenin bağlantı yapısı.....	35
Şekil 3.13. Ölü zamanlı rölenin yapısı.....	37
Şekil 3.14. Optimizasyon ile model parametrelerinin belirlenmesi.....	39
Şekil 4.1. Basamak yanıtı.....	44
Şekil 4.2. Frekans yanıtı.....	46
Şekil 4.3. Smith öngörücüsü.....	48
Şekil 4.4. Sezgisel algoritmalar ile denetleyici tasarımı bloğu.....	50

Şekil 5.1. PARODTA arayüzü.....	52
Şekil 5.2. PSO CS ve FA parametrelerini değiştirme arayüzü.....	52
Şekil 5.3. Parametre optimizasyonu arayüzü.....	54
Şekil 5.4. Tek-Çift kanallı röle ayarı.....	55
Şekil 5.5. Menu sekmesi.....	55
Şekil 5.6. Ayarlar sekmesi.....	55
Şekil 5.7. Basamak cevabı örnek uygulama.....	56
Şekil 5.8. Tek kanallı role testi.....	57
Şekil 5.9. Çift kanallı röle testi.....	58
Şekil 5.10. Diyagramlar alt menüsü.....	58
Şekil 5.11. Denetleyici tasarım uygulaması.....	59
Şekil 5.12. Denetleyici tasarım uygulaması.....	60
Şekil 6.1. PSO için PI denetleyici sonuçları.....	68
Şekil 6.2. Cuckoo Search için PI denetleyici sonuçları.....	68
Şekil 6.3. Ateş Böceği Algoritması için PI denetleyici sonuçları.....	68
Şekil 6.4. $G_1(s)$ Sistemi için farklı PI denetleyici sonuçları.....	69
Şekil 6.5. PSO için PID denetleyici sonuçları.....	70
Şekil 6.6. CS için PID denetleyici sonuçları.....	70
Şekil 6.7. FA için PID denetleyici sonuçları.....	71
Şekil 6.8. $G_1(s)$ Sistemi için farklı PID denetleyici sonuçları.....	72
Şekil 6.9. PSO için PI-P denetleyici sonuçları.....	73
Şekil 6.10. CS için PI-P denetleyici sonuçları.....	73
Şekil 6.11. FA için PI-P denetleyici sonuçları.....	73
Şekil 6.12. PSO için PI-PD denetleyici sonuçları.....	74
Şekil 6.13. CS için PI-PD denetleyici sonuçları.....	74
Şekil 6.14. FA için PI-PD denetleyici sonuçları.....	75
Şekil 6.15. PSO için PID-PD denetleyici sonuçları.....	76
Şekil 6.16. CS için PID-PD denetleyici sonuçları.....	76
Şekil 6.17. FA için PID-PD denetleyici sonuçları.....	76
Şekil 6.18. PSO için PI Denetleyici Sonuçları.....	77
Şekil 6.19. CS için PI denetleyici sonuçları.....	77
Şekil 6.20. FA için PI denetleyici sonuçları.....	77

Şekil 6.21. $G_2(s)$ Sistemi için Farklı PI Denetleyici Sonuçları.....	78
Şekil 6.22. PSO için PID denetleyici sonuçları.....	79
Şekil 6.23. CS için PID denetleyici sonuçları.....	79
Şekil 6.24. FA için PID denetleyici sonuçları.....	80
Şekil 6.25. $G_2(s)$ Sistemi için Farklı PID Denetleyici Sonuçları.....	81
Şekil 6.26. PSO için PI-P denetleyici sonuçları.....	82
Şekil 6.27. CS için PI-P denetleyici sonuçları.....	82
Şekil 6.28. FA için PI-P denetleyici sonuçları.....	82
Şekil 6.29. PSO için PI-PD denetleyici sonuçları.....	83
Şekil 6.30. CS için PI-PD denetleyici sonuçları.....	83
Şekil 6.31. FA için PI-PD denetleyici sonuçları.....	84
Şekil 6.32. PSO için PID-PD denetleyici sonuçları.....	84
Şekil 6.33. CS için PID-PD denetleyici sonuçları.....	85
Şekil 6.34. FA için PID-PD denetleyici sonuçları.....	85
Şekil 6.35. PSO için PI denetleyici sonuçları.....	86
Şekil 6.36. CS için PI denetleyici sonuçları.....	86
Şekil 6.37. FA için PI denetleyici sonuçları.....	86
Şekil 6.38. $G_3(s)$ Sistemi için Farklı PI Denetleyici Sonuçları.....	87
Şekil 6.39. PSO için PID denetleyici sonuçları.....	88
Şekil 6.40. CS için PID denetleyici sonuçları.....	88
Şekil 6.41. FA için PID denetleyici sonuçları.....	89
Şekil 6.42. $G_3(s)$ Sistemi için Farklı PID Denetleyici Sonuçları.....	90
Şekil 6.43. PSO için PI-P denetleyici sonuçları.....	91
Şekil 6.44. CS için PI-P denetleyici sonuçları.....	91
Şekil 6.45. FA için PI-P denetleyici sonuçları.....	91
Şekil 6.46. PSO için PI-PD denetleyici sonuçları.....	92
Şekil 6.47. CS için PI-PD denetleyici sonuçları.....	92
Şekil 6.48. FA için PI-PD denetleyici sonuçları.....	93
Şekil 6.49. PSO için PID-PD denetleyici sonuçları.....	94
Şekil 6.50. CS için PID-PD denetleyici sonuçları.....	94
Şekil 6.51. FA için PI-PD denetleyici sonuçları.....	94
Şekil 6.52. PSO için PID denetleyici sonuçları.....	95

Şekil 6.53. CS için PID denetleyici sonuçları.....	95
Şekil 6.54. FA için PID denetleyici sonuçları.....	96
Şekil 6.55. $G_4(s)$ Sistemi için farklı PID denetleyici sonuçları.....	97
Şekil 6.56. PSO için PI-P denetleyici sonuçları.....	98
Şekil 6.57. CS için PI-P denetleyici sonuçları.....	98
Şekil 6.58. FA için PI-P denetleyici sonuçları.....	98
Şekil 6.59. PSO için PI-PD denetleyici sonuçları.....	99
Şekil 6.60. CS için PI-PD denetleyici sonuçları.....	99
Şekil 6.61. FA için PI-PD denetleyici sonuçları.....	99
Şekil 6.62. PSO için PID-PD denetleyici sonuçları.....	100
Şekil 6.63. CS için PID-PD denetleyici sonuçları.....	100
Şekil 6.64. FA için PID-PD denetleyici sonuçları.....	101
Şekil 6.65. PSO için PI Denetleyici Sonuçları.....	101
Şekil 6.66. CS için PI denetleyici sonuçları.....	102
Şekil 6.67. FA için PI denetleyici sonuçları.....	102
Şekil 6.68. $G_4(s)$ sistemi için Farklı PI denetleyici sonuçları.....	103
Şekil 6.69. PSO için PI-P denetleyici sonuçları.....	104
Şekil 6.70. CS için PI-P denetleyici sonuçları.....	104
Şekil 6.71. FA için PI-P denetleyici sonuçları.....	105
Şekil 6.72. PSO için PI-PI denetleyici sonuçları.....	105
Şekil 6.73. CS için PI-PI denetleyici sonuçları.....	106
Şekil 6.74. FA için PI-PI denetleyici sonuçları.....	106

TABLolar LİSTESİ

Tablo 4.1. Zeigler Nichols basamak yanıtına göre denetleyici tasarımı [73].....	45
Tablo 4.2. Zeigler Nichols frekans yanıtına göre denetleyici tasarımı.....	46
Tablo 4.3. AMIGO basamak cevabına göre PI PID[73, 76, 77].....	47
Tablo 4.4. AMIGO frekans cevabına göre PI PID Denetleyici [73, 76,77].....	47
Tablo 6.1. Algoritmaların parametreleri.....	62
Tablo 6.2. $G_1(s)$ için gerçek sistem ile algoritmaların basamak cevabına göre bulduğu sonuçların benzerlikleri.....	62
Tablo 6.3. $G_1(s)$ için gerçek sistem ile algoritmaların tek röle cevabına göre bulduğu sonuçların benzerlikleri.....	63
Tablo 6.4. $G_1(s)$ için gerçek sistem ile algoritmaların çift röle cevabına göre bulduğu sonuçların benzerlikleri.....	63
Tablo 6.5. FOPDT sisteminin en iyi ve en kötü parametreleri ve literatürdeki sonuçlar.....	63
Tablo 6.6. $G_2(s)$ için gerçek sistem ile algoritmaların basamak cevabına göre bulduğu sonuçların benzerlikleri.....	64
Tablo 6.7. $G_2(s)$ için gerçek sistem ile algoritmaların tek röle cevabına göre bulduğu sonuçların benzerlikleri.....	64
Tablo 6.8. $G_2(s)$ için gerçek sistem ile algoritmaların çift röle cevabına göre bulduğu sonuçların benzerlikleri.....	64
Tablo 6.9. $G_3(s)$ için gerçek sistem ile algoritmaların basamak cevabına göre bulduğu sonuçların benzerlikleri.....	65
Tablo 6.10. $G_3(s)$ için gerçek sistem ile algoritmaların tek röle cevabına göre bulduğu sonuçların benzerlikleri.....	65
Tablo 6.11. $G_3(s)$ için gerçek sistem ile algoritmaların çift röle cevabına göre bulduğu sonuçların benzerlikleri.....	65
Tablo 6.12. $G_4(s)$ için gerçek sistem ile algoritmaların basamak cevabına göre	

bulduğu sonuçların benzerlikleri.....	66
Tablo 6.13. $G_4(s)$ için gerçek sistem ile algoritmaların tek röle cevabına göre bulduğu sonuçların benzerlikleri.....	66
Tablo 6.14. $G_4(s)$ için gerçek sistem ile algoritmaların çift röle cevabına göre bulduğu sonuçların benzerlikleri.....	66
Tablo 6.15. $G_5(s)$ için gerçek sistem ile algoritmaların basamak cevabına göre bulduğu sonuçların benzerlikleri.....	67
Tablo 6.16. $G_5(s)$ için gerçek sistem ile algoritmaların tek röle cevabına göre bulduğu sonuçların benzerlikleri.....	67
Tablo 6.17. $G_5(s)$ için gerçek sistem ile algoritmaların çift rölecevabına göre bulduğu sonuçların benzerlikleri.....	67
Tablo 6.18. G_1 Sisteminde PI için PSO CS ve FA sonuçları.....	68
Tablo 6.19. $G_1(s)$ Sistemi için Farklı PI denetleyici katsayıları.....	69
Tablo 6.20. G_1 Sisteminde PID için PSO CS ve FA sonuçları.....	70
Tablo 6.21. $G_1(s)$ Sistemi için Farklı PID denetleyici katsayıları.....	71
Tablo 6.22. G_1 Sisteminde PI-P için PSO CS ve FA sonuçları.....	72
Tablo 6.23. G_1 Sisteminde PI-PD için PSO CS ve FA sonuçları.....	74
Tablo 6.24. G_1 Sisteminde PID-PD için PSO CS ve FA sonuçları.....	75
Tablo 6.25. G_2 Sisteminde PI için PSO CS ve FA sonuçları.....	77
Tablo 6.26. $G_2(s)$ Sistemi için Farklı PI Denetleyici Katsayıları.....	78
Tablo 6.27. G_2 Sisteminde PID için PSO CS ve FA sonuçları.....	79
Tablo 6.28. $G_2(s)$ Sistemi için Farklı PID denetleyici katsayıları.....	80
Tablo 6.29. G_2 Sisteminde PI-P için PSO CS ve FA sonuçları.....	81
Tablo 6.30. G_2 Sisteminde PI-PD için PSO CS ve FA sonuçları.....	83
Tablo 6.31. G_2 Sisteminde PID-PD için PSO CS ve FA sonuçları.....	84
Tablo 6.32. G_3 Sisteminde PI için PSO CS ve FA sonuçları.....	85
Tablo 6.33. $G_3(s)$ Sistemi için Farklı PI Denetleyici Katsayıları.....	87
Tablo 6.34. G_3 Sisteminde PID için PSO CS ve FA sonuçları.....	88
Tablo 6.35. $G_3(s)$ Sistemi için Farklı PID denetleyici katsayıları.....	89
Tablo 6.36. $G_3(s)$ Sisteminde PI-P için PSO CS ve FA sonuçları.....	90
Tablo 6.37. G_3 Sisteminde PI-PD için PSO CS ve FA sonuçları.....	92
Tablo 6.38. G_3 Sisteminde PID-PD için PSO CS ve FA sonuçları.....	93

Tablo 6.39. G ₄ Sisteminde PID için PSO CS ve FA sonuçları.....	95
Tablo 6.40. G ₄ (s) Sistemi için Farklı PID denetleyici katsayıları.....	96
Tablo 6.41. G ₄ Sisteminde PI-P için PSO CS ve FA sonuçları.....	97
Tablo 6.42. G ₄ Sisteminde PI-PD için PSO CS ve FA sonuçları.....	99
Tablo 6.43. G ₄ Sisteminde PID-PD için PSO CS ve FA sonuçları.....	100
Tablo 6.44. G ₅ Sisteminde PI için PSO CS ve FA sonuçları.....	101
Tablo 6.45. G ₄ (s) Sistemi için Farklı PI denetleyici katsayıları.....	103
Tablo 6.46. G ₅ Sisteminde PI-P için PSO CS ve FA sonuçları.....	104
Tablo 6.47. G ₅ Sisteminde PI-PI için PSO CS ve FA sonuçları.....	105

ÖZET

Anahtar kelimeler: Ölü Zamanlı Sistemler, PSO, Ateşböceği, Gugukkuşu, Auto-tunning

Bu çalışmada literatürde bulunan ölü zamanlı sistemlerin modellenmesi ve denetiminde kullanılan bazı matematiksel tabanlı teknikler tanıtılmıştır. Ardından bu tekniklerin yapmış olduğu işlemler, son zamanlarda doğadan esinlenilerek geliştirilmiş olan ve sezgisel algoritma olarak adlandırılan optimizasyon teknikleri ile yeniden yapılmıştır.

Bu çalışmada, literatürde kullanılan bu sezgisel algoritmalarından parçacık sürü optimizasyonu, gugukkuşu ve ateşböceği algoritmaları tanıtılmış ve akış diyagramları verilmiştir. Ayrıca bu algoritmalarda kullanılan IAE, ISE, ITSE gibi amaç fonksiyonları tanıtılmış ve bu amaç fonksiyonlarının öneminden bahsedilmiştir.

Ayrıca bu çalışmada, optimizasyon algoritmalarının daha rahat kullanılabilmesi için kullanıcı dostu bir arayüz tasarımı yapılmıştır. Bu arayüze, ister veri isterse sistem parametreleri girilebilmektedir. Bu arayüz, girilen bilgilere ve kullanıcı seçeneklerine göre(amaç fonksiyonu ve algoritma gibi) hem model parametrelerini, hemde sistem için belirlenen denetleyici parametrelerini belirleyebilmektedir. Ayrıca seçilen bu algortimaların parametleri değiştirilerek, performansları karşılaştırılabilmektedir.

Oluşturuluş olan bu arayüz ile literatürde yaygın olarak kullanılan beş farklı sistemin hem parametre optimizasyonu, hemde denetleyici tasarımları yapılmıştır. Elde edilen simulasyon sonuçlarından, bu beş sistem için ateşböceği ve parçacık sürü optimizasyonunun daha başarılı sonuçlar verdiği görülmüştür. Ayrıca tanıma işleminde elde edilen sonuçlardan bazıları literatürde olan sonuçlarla kıyaslanmış. Denetleyici tasarımında da elde edilen denetleyici katsayıları Zeigler Nichols, AMIGO ve Smith Öngürücüsü gibi farklı denetleyici tipleri ile karşılaştırılmış ve elde edilen denetleyici katsayıları tablolar halinde verilirken ve sistem yanıtları grafik halinde verilmiştir.

MODELLING AND CONTROL OF TIME DELAYED SYSTEMS

SUMMARY

Keywords: Time Delayed Systems, PSO, Firefly, Cuckoo Search, Auto-tuning

In this study, some mathematical based techniques used in the identification and control of dead time systems in the literature are introduced. Then the processes that these techniques use have been reworked with the optimization techniques, which are recently developed by inspiration from the nature and called the heuristic algorithm.

Some of these heuristic algorithms cited in the literature such as Particle Swarm Optimization, Cuckoo Search and Firefly algorithms are introduced and their flow diagrams are given in this study.

In addition to these, in this study a user-friendly interface design has been made to make the optimization algorithms easier to use. Either the system parameters or data can be entered to this interface. This interface then can determine both the model parameters and the controller parameters according to the entered information and user options (such as the objective function and the algorithm). In addition, the parameters of these selected algorithms can be changed and their performances can be compared.

With this interface that has been created, both parameter optimization and controller designs of five different systems, which are widely used in the literature have been made. From the obtained simulation results, it was seen that firefly and particle swarm optimization gave more successful results for these five systems. In addition, some of the results obtained in the identification process are compared with those in the literature. The controller coefficients obtained in the controller design were compared with different controller types such as Zeigler Nichols, AMIGO and Smith Predictor, and the obtained controller coefficients were tabulated and the system responses were plotted graphically.

BÖLÜM 1. GİRİŞ

Ölü zamanlı sistemler endüstri de en çok karşılaşılan sistemlerdir. Zaman gecikmeli olan bu sistemlere haddeleme, ısı eşanjörü, nükleer reaktörler, zincir kasnak sistemleri, kimyasal süreç olarak petrol damıtma, bakterilerin çoğalması veya son zamanlarda gelişen bilişim sektörleri örnek olarak verilebilmektedir.

Ölü zamanlı sistemlerin modellenmesi ve kontrolünde şu zamana kadar pek çok teknik geliştirilmiştir ve bilgisayar teknolojisindeki ilerleme ile bu gelişme devam etmektedir. Bu teknikler analitik olduğu gibi sayısal yöntemlerle de geliştirilmiştir. Analitik olarak geliştirilen tekniklerde sistem modeli üzerinde tanıma ve kontrol yapılmaya çalışılmaktadır. Sayısal olarak geliştirilen yöntemler de ise, belirlenen hata fonksiyonu iteratif olarak minimum yapılmaya çalışılarak yapılmaktadır. Bu optimizasyon tekniklerine bakıldığında yine iteratif olarak çalışan klasik teknikler kullanılabildiği gibi, bilgisayar teknolojisinin gelişmesi ile uygulama imkanı artan sezgisel algoritmalarda kullanılabilmektedir.

Bu tez çalışmasında Parçacık Sürü Optimizasyonu, Guguk Kuşu Algoritması ve Ateş Böceği Algoritmaları kullanılarak sistem modelleme ve denetleyici tasarımı yapılmıştır. Yine bu algoritmaların modelleme ve denetleyici tasarımı problemleri için performansları karşılaştırılmıştır. Ardından bu algoritmaların daha kolay kullanılabilmesi için, kullanıcı dostu bir arayüz tasarımı gerçekleştirilmiştir.

1.1. Literatür Taraması

Kontrol teorisindeki gelişmeler Helenizmden gelen su saatlerindeki çalışmalara kadar dayanmaktadır. Su saatinde, şamandıra vasıtası ile sabit debili su akışı regüle edilmeye çalışılmaktadır. İlerleyen dönemlerde ise bu düzenek biraz daha geliştirilerek zamanı

ölçmek için kullanılmıştır. Kontrol alanının temellerini atılmasında Maxwell ve Routh'un yaptığı çalışmalarda governor aletinin matematiksel olarak incelenmesi önemli rol oynamıştır [1, 2]. Tabii sanayi devriminde James Watt'ın buhar makinası ile birleştirdiği governor aleti ile buharlı makinelerdeki hız kontrolünün üretimdeki önemi, sanayiye uygulanmaya başlandığı zaman çok daha fazla anlaşılmıştır. Bu çalışmalar ile hız kazanan bir alan olan kontrol sistemleri, Zeigler-Nichols'un yaptığı çalışmalar ile farklı yön almaya başlamıştır. Zeigler-Nichols ilk olarak düşük dereceli sistemleri basamak cevap eğrisine göre birinci dereceden ölü zamanlı sistem gibi modellemiş, ardından elde ettikleri sistem modeline uygun denetleyici tasarımı yapmışlardır. Daha sonra bunu geliştirerek kapalı döngü kazancını arttırıp, sistemin osilasyona girmesi sağlanmış ve bu osilasyon cevabını kullanarak sistemin kritik kazanç ve kritik frekansına göre denetleyici tasarımı yapmışlardır [3-5].

Tytskin ise kapalı döngüde genellikle lineer ve nonlineer sistemler için röleyi kullanmıştır. Bu röle tekniği ile sistem osilasyona girmektedir. Ayrıca bu teknik sistemi kararlı bölgede tutarak, sistem çıkışını sınırlandırmıştır. Ayrıca osilasyon cevabı sayesinde sistemin frekans cevabı bulunmuştur [6, 7].

Aström ve Hagglund yapmış oldukları çalışmada, bu röle tekniğini kullanarak sistemin kritik kazanç ve kritik noktasını bulmuşlar ve Nyquist eğrisinde sistemin çalışmasını istedikleri noktaya taşıyan denetleyici tasarımını önermişlerdir [8].

Luyben röle tekniğini ilk defa nonlineer bir sistem olan damıtma sistemlerinde uygulamıştır. Bu uygulamada sistemin çalıştığı bölgeye göre transfer fonksiyonunun bilindiği ve bununda belirli bölgelerde birinci dereceden ölü zamanlı sistem olduğu kabulü yapılmıştır. Dolayısı ile bu bölgede röle tekniği kullanılarak, sistem parametreleri belirlenmeye çalışılmıştır. Bu yöntemde de yaklaşık transfer fonksiyon (ATV) yöntemi denilmiştir. Fakat bu yöntem kullanılırken sisteme çok büyük bozucular etki ettiğinde, bu bozucuların sistemi farklı bir noktaya taşıdığı ve bu yüzden elde edilen bu transfer fonksiyonunun sistemi temsil edemediği ifade edilmiştir [9].

Wei Li ve ark. yapmış oldukları çalışmada, Luyben'in 1990 yılında önermiş olduğu ATV ile Muhrelin yöntemini birleştirerek geliştirilmiş bir Auto-tuning tekniği önermişlerdir. İki aşamadan oluşan işlemde ilk aşamada Auto-tuning testi yapılmaktadır. İkinci aşamada ise röleye ölü zaman ekleyerek bir test yapılmaktadır. Yapılan testlerden sonra en küçük kareler metoduna uygun bir şekilde düzenlenmiş denklemler sayesinde, sistemin farklı tipte modeller için parametreleri belirlenmektedir. Sisteme en uygun model seçimi için, derecesi en küçük olan ve parametreleri negatif olmayan tercih edilmektedir. Ayrıca önermiş oldukları yöntemi gerçek zamanlı bir uygulama üzerinde denemişler ve sonuçlarını paylaşmışlardır [10].

Eşref ve ark. yapmış oldukları çalışmada, parametrik yöntemlerle sistem modellemeye yönelmişlerdir. Hata öngörü algoritması ile sistem modelleme işlemi yapmışlardır. Sistemi modellemek için geri beslemeli röle kullanmışlar ve geri beslemeye Zero Order Hold (ZOH) ekleyerek sistemi ayırıklaştırmışlardır. Bu işlemi yaparken, kendi belirledikleri ARMAX modelini temsil eden bir transfer fonksiyonuna göre hata öngörü algoritması ile ayırık modelin parametrelerini bulmuşlardır. Ardından modellenmiş olan sisteme, Dinamik Matris Kontrol yöntemi ile denetleyici tasarlamışlar ve sonuçları gerçek bir sistem üzerinde karşılaştırmışlardır [11].

Kennedy ve Eberhart yapmış oldukları çalışmada, ilk olarak parçacık sürü optimizasyonunu önermişler ve nasıl çalıştığını açıklamışlardır. Ayrıca PSO algoritmasının çalışmasını etkileyen iki faktör olduğunu, bunlardan birisinin en iyi olan parametre çarpanı, diğerinin ise yaklaşmayı belirleyen ve hızların çarpımı olan rastlantı sayısı olduğunu belirtmişlerdir. Dolayısı ile bu algoritmanın hem sürü hem evrimleşmeyle başarılı sonuçlar elde edilebileceğinden bahsetmişlerdir [12]. Yine aynı kişiler, çözümlenmesi zor problemlerde PSO algoritmasının kullanılabilir olduğunu belirtmişler ve bunun ispatı için yapay sinir ağlarını eğitmeye yönelik çalışmalarda bulunmuşlardır [13].

Sung ve ark. yapmış oldukları çalışmada, normal röle testinin sadece 1. Harmoniği baskın olan sistemlerde daha doğru sonuçlar verdiğini, dolayısı ile bu duruma uymayan sistemlerde harmoniklerin etkisini azaltarak, 1. harmoniğin etkisini arttırmak

için normalde iki seviye olan röle çıkışını altı seviye yaparak, çıkış sinyalinin daha fazla sinüs sinyaline benzemesini sağlamışlardır. Bu sayede ilk harmonik daha baskın hale gelmiştir ve ölü zamanın artması ile kalan harmoniklerin etkisinin artması engellenmiştir. Ayrıca bu işlemin daha hassas sonuç verdiği görülmüştür [14].

Ho ve ark. yapmış oldukları çalışmada, asimetrik ve histeresizli röle ile birinci dereceden ölü zamanlı sistem modellemesi yapmışlardır. Ayrıca integratör içeren bir sistemin önüne bir türev operatörü getirerek, uygulamış oldukları tekniğin integratör içeren sistemler içinde uygulanabileceğini göstermişlerdir. Ardından sisteme Zeigler-Nichols'sün kapalı döngü PID tekniği ile PID denetleyici tasarımı yapmışlardır. Ayrıca önermiş oldukları tekniği sıvı tank sisteminde uygulayarak, başarılı sonuçlar elde etmişlerdir [15].

Tan ve ark. yapmış oldukları çalışmada, röle tekniği ile sistemi farklı iki noktada tanıma işlemi yapmışlardır. Ayrıca tanıma işlemi sırasında, transfer fonksiyonu olarak FOFDT ve SOPDT modelleri kullanmışlar ve hangisi daha iyi sonuç veriyorsa iyi olanı seçmişlerdir. Ardından modellenen sistem için denetleyici tasarımı yapmışlardır [16].

Shen ve ark. yapmış oldukları çalışmada, asimetrik röle kullanarak sistem parametrelerini tahmin etmeye çalışmışlardır. Yaptıkları bu çalışmalarda, asimetrik röle ile parametrelerin doğru kestirilmesinin ancak düşük frekanslar için mümkün olduğunu göstermişlerdir. Ayrıca Bias arttıkça kritik frekanstan sapmalar olduğunu ve tekniğin yanlış sonuçlar bulduğunu ifade etmişlerdir. Yine sistemin gürültü altında çalışabilmesi için uzun süreli bir osilasyon yapması gerektiğini, aksi takdirde sapmaların olabileceğini ifade etmişlerdir. Nitekim doğru bir kestirim yapabilmek için ilk olarak normal röle ve asimetrik röle ile ayrı ayrı elde edilen sistem kritik frekans ve kazanç değerlerinin karşılaştırılıp doğrulanması gerektiği, ardından sistemin gerçek kazancının integral formülünden doğru bir şekilde bulunabileceği ifade edilmektedir. Burada sistem parametrelerini bulmak için En küçük Kareler Yöntemi (LSE) kullanmışlar ve ATV metodundan farklı olarak sistemin ölü zamanını hesaplamışlar ve ölü zaman gerçek zamandan farklı çıktığı takdirde, sistemin derecesini yükselterek parametre belirleme işlemi yapmışlardır [17].

Wang ve ark. yapmış oldukları çalışmada, frekans cevabı ile sistem parametrelerini kestirmeye çalışmışlardır. Bu metodun hem robust olduğunu, hemde zaman kazandıran ve kolay bir teknik olduğunu belirtmişlerdir. Model kestirim işleminde genelde önemli olan modelin sistemi iyi temsil edebilmesidir. Yine modellemede kestirim tek noktaya göre yapıldığında, denetleyici tasarımında problemler oluşabilmektedir. Bu nedenle Wang Q. ve arkadaşları yapmış oldukları çalışmada, çoklu noktaya göre kestirim işlemi yapmışlar ve sinüsoidal transfer fonksiyonu olarak bilinen tekniğin Wellstead tarafından ilk defa kullanıldığını belirtmişlerdir. Fakat sinüsoidal transfer fonksiyonu ile kestirim işleminin, uzun süre gerektirdiğini belirtmişlerdir. Ayrıca bu frekans yöntemi ile yapılan kestirim işleminin geçici durumu da içerdiğini belirterek, bunun için bazı değişikliklerin yapılması gerektiğini ifade etmişlerdir. Yine parametre kestiriminde gürültünün çok büyük bir sorun olduğunu, röle testinin gürültüyü azalttığını, gürültü eğer farklı bantlarda ise filtre kullanılabileceğini, ayrıca gürültülü ortamda rölenin histeresiz bant genişliğinin gürültünün bant genişliğinin 2 katı olması gerektiğini belirtmişlerdir. Frekans cevabı ile elde ettikleri verilere göre en küçük kareler metodu ile parametrelerin kestirimini yaparak, ölü zamanın da kestirimini yapmışlardır. Diğer taraftan önerdikleri modelle ilgili olarak, eğer modelin derecesi düşük ise o zaman indirgenmiş modelin ölü zamanının büyüdüğünü belirtmişlerdir. Bu metodun avantajları (i) tek röle testi ile kestirimin mümkün olduğu, (ii) kritik kazancın çok hassas sonuç verdiğini ve (iii)denetleyici tasarımında kolay bir metot olduğunu belirtmişlerdir [18].

Bi ve ark. yapmış olduğu çalışmada, sisteme standart röleye ilaveten bir parazitik röle eklemişlerdir. Bu parazit içeren röle sayesinde Hızlı Fourier Dönüşümünü(FFT) kullanarak kritik frekans etrafındaki noktalara göre tanıma işlemi yapabildiklerini ifade etmişlerdir. Genelde sistemleri tanımanın iki temel sorunu olduğunu, bunlardan ilkinin Describing Function yaklaşımı olduğunu ve diğer sorununda kritik noktaya göre yapılan tanıma işlemi ile kaba bir kontrol yapıldığını belirtmişlerdir. Ayrıca sistem birden fazla noktaya göre tanınmak isteniyorsa, sisteme fazladan komponent eklenmesi gerektiği belirtilmektedir. Burada parazitik rölenin genliği öyle bir alfa değerinde seçilmeli ki, sistemi harekete geçirebilecek bir oranda olmalı fakat sistemin osilasyonunu bozmamalıdır Sonuç olarak denemeler yaparak, başarılı sonuçlar

almışlar ve bu metodun (i) tek röle testi ile birden fazla noktanın tanınabilmesine imkan sağladığı, (ii) yaklaşım olmadığı için çok hassas olduğu ve (iii) gürültülere ve bozuculara karşı duyarsız olduğu ifade edilmiştir [19].

Wang ve ark. yapmış oldukları çalışmada, sistemleri.n FOPDT modellerini biaslı ve biassız röle tekniği ile elde ettikleri zaman ve frekans cevaplarına göre yapmışlardır. Röle tekniği ile parametre kestirim işleminde muhakkak model bilgisine ihtiyaç duyulduğunu ve eğer model sisteme tam uyuyor ise kesin olarak parametrelerin bulunabileceğini ifade etmişlerdir. Histeresizli ve asimetrik röle ile tanıma işleminde ilk olarak kazanç bulunmalıdır. Ardından ölü zaman ile zaman sabiti arasındaki oran bulunmalıdır. Bu ikisi bulunduktan sonra, sistemin zaman sabiti bulunmalıdır. Ardından bu bilgiler kullanılarak sistemin ölü zamanı hesaplanabilmektedir. Histeresizli röle ile tanıma yapıldığında bir denkleme daha ihtiyaç duyulduğunu veya kazanç için ikinci bir testin yapılması gerektiğini ifade etmişlerdir. Bu yöntemle tanıma yapıldığı takdirde ilk olarak çıkış genliği bulunmalı ve buna bağlı olarak iteratif olarak zaman sabiti ve ardından kazanç ve ölü zaman hesaplanmalıdır. Bu çalışmada sunulan FOPDT tanıma işleminin hem hassas hemde uygulanabilir sonuçlar verdiğini belirtmişlerdir [20].

Friman yapmış olduğu çalışmada kapalı döngüye PI ekleyerek, sistemin istenilen noktada osilasyona girmesini sağlamıştır. Ayrıca sistemin kritik kazanç ve kritik frekansını, bu osilasyon cevabından sağlayan denklemleri vermiştir [21].

Friman ve ark. yaptıkları çalışmada sistem tanımada çift kanallı röle testini önermişlerdir. Bu test sayesinde sistemin faz açısı 180 dereceden farklı bir noktaya kaydırılarak, sistemin asıl çalışacağı bölgeye yakın bir noktada frekans ve fazı bulunmuştur. Ayrıca bu tanıma işleminden sonra elde edilen faz açısı ve frekansa bağlı olarak, kullanılması gereken kontrol tipleri için öneride bulunmuşlardır [22].

Mitsukura ve ark. yapmış oldukları çalışmada birinci dereceden ölü zamanlı bir sistemin parametrelerini en küçük kareler yöntemi ile belirlemişlerdir. Aynı zamanda

belirledikleri sisteme genetik algoritma ile PID denetleyici tasarımını da gerçekleştirmişlerdir [23].

Luyben yapmış olduğu çalışmada, genelde sistem hakkında daha fazla bilgi elde etmek için sisteme ilave bileşenler katıldığını ifade etmiştir. Yaptığı çalışmada, sistemi daha karmaşık hale getiren bu işlemler yerine sistemin cevaplarına bakılarak, Ölü Zaman ve Zaman sabiti oranına göre sonuçlar çıkarılabileceğini ifade etmiştir. Örnek olarak FOPDT sisteminin eğer zaman sabiti çok büyük ise ölü zamanlı integratörlü bir sisteme benzediğini, eğer zaman sabiti çok küçük ise o zaman kare dalgaya benzediğini belirtmiştir. Dolayısı ile sistem cevabının bu oranına bağlı olarak, bir eğrilik katsayısı (F) önerisi yapmıştır. Sonuç olarak sistem cevabının bilindiği durumda eğrilik katsayısı ve buradan da ölü zaman ve zaman sabiti oranı hesaplanabilmektedir. Buna bağlı olarak çalışmada, yöntemin kullanım algoritması verilmiştir. Diğer taraftan, bu yöntem ile yüksek dereceli sistemler ve kararsız sistemlerin tanınmasının çok etkin olmadığı ifade edilmiştir [24].

Kealy ve O'Dwyer yapmış oldukları çalışmada, FOPDT bir sistem için parametrelerini kendilerinin belirledikleri bir PI denetleyici kullanmışlardır. Bu sayede sistemi ikinci dereceden bir sistem gibi modellemişlerdir. Basamak cevabı sayesinde tepe noktalarını kullanmışlar ve bu noktalar sayesinde sistemin 5 farklı parametresini belirlediklerini ifade etmişlerdir. Önerdikleri yöntem ilede bir ısı sistem üzerinde deneysel çalışmalar yaparak, sistem parametrelerini belirlemişlerdir [25].

Gaing ise yapmış olduğu çalışmada, otomatik voltaj regülatörünün kontrolü için PSO algoritmasını kullanmıştır. Zaman domeninde tanımladığı amaç fonksiyonu ile bu algoritmayı kullanarak, optimum PID tasarımı gerçekleştirmiştir ve sonuçlarını Genetik Algoritmalar ile karşılaştırmıştır [26].

Wilson yapmış olduğu çalışmada röle tekniği ile elde edilen kritik kazanç ve kritik frekansın, Zeigler-Nichols ayar yöntemine uygulanabildiğini söylemiş ve histeresizi değiştirerek, Nyquist Eğrisi üzerinde birden fazla noktanın osilasyon cevabına göre elde edebileceğini belirleterek, simülasyon çalışmaları yapmıştır [27].

Kaya yapmış olduğu çalışmada, integratör içeren birinci dereceden ölü zamanlı bir sistem parametrelerini, asimetrik ve histeresizli bir röle ve A function metodunu kullanarak elde etmiştir. Ayrıca asimetrik röle kullanarak sistemin daha çok parametresinin belirlenebildiğini ifade etmiştir. Modellenen sistem integratör içerdiğinden, sistem sonuna türev operatörü yerleştirerek integratörü yok etmiş ve sistemi birinci dereceden sistem gibi modellemiştir. Yine bu çalışmada, modellenen sistem derecesinin yüksek olması durumunda, modellemenin integratörlü bir sistem olarak da yapılabileceği ifade edilmiştir. Yapılan çalışma sonucu elde edilen sonuçlar, farklı çalışmalarla kıyaslanmış ve ardından sisteme denetleyici tasarımı yapılmıştır [28].

Boz ve Sarı yapmış oldukları çalışmada, belirledikleri performans kriterlerini kullanarak farklı sistemler için en optimum PI-PD ve PI-PID tasarımları gerçekleştiren Standart Form yapılarını ve sonuçlarını vermişlerdir [29, 30].

Yang ve Deb 2009 yılında guguk kuşlarından esinlenerek modelledikleri Guguk Kuşu Algoritmasını önermişlerdir. Yapısı itibari ile hem global hem de lokal aramayı dengeleyerek iteratif olarak çalışan bu algoritmayı, literatürde bulunan ve benchmark olarak kabul edilen problemlerin çözümünde kullanmışlar ve başarısını diğer algoritmalar ile kıyaslamışlardır [31].

Yang tarafından önerilen Ateş Böceği Algoritması, ateş böceklerinin avlanma ve karşı cinsi etkileme gibi davranışlarından esinlenilerek geliştirilmiştir. Ateş böceklerinin parlaklığına, çekiciliğine ve birbirlerine uzaklığına bağlı olarak modellenmiş olan bu algoritma ile, çok boyutlu optimizasyon problemlerinde başarılı sonuçlar elde edildiği görülmüştür [32].

Soltesz ve ark. yapmış oldukları çalışmada iki kanallı röle kullanarak, sistemin giriş ve çıkış sinyallerini toplamış ve ardından sistemi birinci dereceden bir sistem olarak modelleyerek, sistem parametrelerini klasik yöntem olan Newton-Raphson metodu ile belirlemişlerdir. Ayrıca bu tanıma yönteminin tüm ölü zamanlı sistemlere uygulanabileceğini ifade ederek, gelecek çalışmaları için denetleyici tasarımı

gerçekleştireceklerini ve bunların performanslarını değerlendireceklerini belirtmişlerdir [33].

Civicioğlu ve Besdok yaptıkları çalışmada guguk kuşu, parçacık sürü optimizasyonu, evrimsel algoritma ve yapay arı kolonisi algoritmalarını, literatürde bulunan 50 benchmark problemde kullanmışlar ve elde ettikleri performansları birbirleri ile karşılaştırmışlardır [34].

Bobál ve ark. yapmış oldukları çalışmada sistem parametrelerini Yinelemeli En Küçük Kareler Metodu ile tanıyarak, modelin parametrelerini sistem çalışırken sürekli güncellemişlerdir. Ardından sürekli güncellenen bu model ile sistemi kontrol edebilmek için, Smith Kestirimcisi (Smith Predictor) kullanmışlardır. Denetleyici tasarımında kutup atama metodunu kullanmışlardır. Bu çalışmayı hem simülasyon ortamında, hemde deneysel olarak bir ısı sistem üzerinde kullanmışlardır [35].

Kaya ve Nalbantoğlu yapmış oldukları çalışmada, geri beslemeli röle tekniğini kullanarak ikinci dereceden ters cevaplı ölü zamanlı bir sistemin parametrelerini belirlemeye çalışmışlardır. Kendi belirledikleri amaç fonksiyonuna göre genetik algoritma kullanarak sistemin ölü zamanını, kazancını, sistem kutuplarını ve sistem sıfırını belirlemişlerdir [36].

Mirzal ve ark. ise ölü zamanı ve parametreleri sürekli değişen birinci dereceden ölü zamanlı bir sistemin denetimi üzerine çalışmışlardır. Bu sistem için iteratif yöntem, Zeigler Nichols ve Genetik Algoritma olmak üzere 3 farklı yöntem ile denetleyici tasarımı gerçekleştirmişler ve en iyi sonucun Genetik Algoritma ile tasarlanan PID olduğunu görmüşlerdir [37].

Misal ve ark. yapmış oldukları çalışmada bir nükleer santral parametrelerini MATLAB Toolbox'ını kullanarak belirlemişlerdir. Elde ettikleri modeli kullanarak geri beslemeli röle yardımıyla kritik kazanç ve kritik frekansı belirlemişler ve denetleyici tasarımı için Zeigler Nichols metodunu kullanmışlardır. Elde ettikleri sonuçları literatürde olan başka bir çalışma ile kıyaslamışlardır [38].

Gandomi ve ark. yapmış oldukları çalışmada Guguk Kuşu Algoritmasını bazı kısıtlı yapı problemlerin çözümünde kullanmışlardır. Sonuçların daha önceden var olan çözümlerden daha başarılı olduğu ifade edilmiştir [39].

Berner ve arkadaşları yapmış oldukları çalışmada adaptif asimetrik ve histeresizli röle kullanmayı önermişlerdir. Ardından sisteme göre ayarlanan bu adaptif röle tekniği ile çok fazla deneme yaparak, farklı sistemlere uygulamışlar ve rölenin parametrelerini almışlardır. Akabinde bu verilerle eğriler oluşturmuşlardır. Bu adaptif röle, herhangi bir sistem üzerinde uygulandığında sisteme adapte olan rölenin parametreleri alınarak, daha önceden çizilmiş olan eğrilerle FOPDT olarak sistem parametreleri elde edilmiştir [40].

Hassaan yapmış olduğu çalışmada çok osilasyonlu ikinci dereceden bir sistem için PD-PI denetleyici tasarımı gerçekleştirmiş ve klasik PID ye göre daha başarılı olduğunu ifade etmiştir [41].

Prokop ve arkadaşları röle tekniği ile sistem parametrelerini belirlemişler ve ardından belirlenen sisteme, ikinci dereceden serbestlik derecesine sahip denetleyici tasarımı uygulaması yapmışlardır [42].

Karagül yapmış olduğu çalışmada Guguk Kuşu Algoritmasını kullanarak atık toplama problemini çözümlenmiş ve farklı bir teknikle de karşılaştırarak, Guguk Kuşunun daha başarılı sonuçlar verdiğini göstermiştir [43].

Roeva ve Slalov yapmış oldukları çalışmada ise, nonlinear bir sistem için geliştirilmiş Smith Kestirimcisi ve Ateş Böceği Algoritması sayesinde PID denetleyicisi tasarımı gerçekleştirmişlerdir. Sonucu literatürde olan ve Genetik Algoritma ile yapılmış bir çalışma ile kıyaslayarak, daha başarılı sonuçlar verdiğini ifade etmişlerdir [44].

Batık ve arkadaşları yapmış oldukları çalışmada, denetleyici tasarımı için Windows tabanlı bir arayüz geliştirmişlerdir. Bu arayüz sayesinde, girilen sistem için Genetik Algoritma kullanılarak P, PI veya PID denetleyici katsayıları bulunabilmektedir [45].

Benner ve ark. daha önceden önerdikleri adaptif röleden elde ettikleri parametreler ile sisteme farklı modeller önermişlerdir. Ardından elde ettiği modele AMIGO denetleyici tasarımı ile denetleyici tasarımı gerçekleştirmişlerdir [46, 47].

Qibing ve arkadaşları yapmış olduğu çalışmada Geliştirilmiş Guguk Kuşu Algoritmasını önermişlerdir. Önerdikleri algortmada, Guguk Kuşu Algoritmasının yapısında olan olasılık katsayısını ve adım uzunluğunu adaptif hale getirmişlerdir. Bu yeni algoritma ile birinci ve ikinci derece ölü zamanlı sistemler için IAE kriterine göre PID denetleyici tasarımı gerçekleştirmişler ve farklı tekniklerle karşılaştırmışlardır [48].

Meena ve arkadaşının yapmış olduğu çalışmada da Geliştirilmiş Ateş Böceği Algoritması önerilmiştir. Bu önerdikleri algortmada, genelde sabit olan çekicilik katsayısının ateş böceklerinin uzaklıklarına bağlı olarak değişmesi sağlanmıştır. Bu şekilde modelledikleri algoritma ile belirledikleri beş farklı ölü zamanlı sistem için PID denetleyici tasarımı gerçekleştirmişlerdir [49].

Nema ve Pady yapmış oldukları çalışmada ölü zamanlı sistem parametrelerini belirledikten sonra, PI-PD denetleyicisi tasarımı gerçekleştirmişlerdir. İlk olarak röle testi ile elde edilen kritik kazanç ve frekansa göre sistem parametreleri belirlenmiştir. Ardından PD denetleyici ile sistem kararlı hale getirilip, guguk kuşu algoritması ile PI denetleyici tasarımı gerçekleştirilmiştir [50].

Shikanth ve ark. yapmış oldukları çalışmada çift ters sarkaç sistemini denge noktasında lineerleştirmişler ve Ateş Böceği Algoritması ile kutup atama metodunu kullanarak sisteme denetleyici tasarımı gerçekleştirmişlerdir [51].

Luo ve Lv yapmış oldukları çalışmada Geliştirilmiş Guguk Kuşu Algoritmasını önermişlerdir. Önerdikleri algortmada, ilk olarak lokal aramayı belli bir iterasyon sayısı ile yaptırmışlar ve belirledikleri iterasyon sayısından sonra global ve lokal aramayı birlikte çalıştırmışlardır. Bu yeni algoritma ile termik santral modelleri için

kendi belirledikleri amaç fonksiyonuna göre PID denetleyici tasarımı gerçekleştirmişlerdir [52].

Karaarslan ve Zengin ise, Ateş Böceği Algoritmasını kullanan ve okullar da verilen haftalık ders programını belirlenen amaç fonksiyonuna göre minimize eden bir arayüz tasarımı gerçekleştirmişlerdir [53].

Gupta ve Padhy yapmış olduğu çalışmada, Ateş Böceği Algoritmasına adım uzunluğunu üstel (exponansiyel) olarak azaltan bir parametre eklemişler ve rastlantı katsayısının iki ateş böceği arasındaki uzaklığa bağlı olarak ölçeklendirilmesini sağlamışlardır. Bu önerdikleri Geliştirilmiş Ateş Böceği Algoritması ile, ölü zamanlı kararsız ve integratörlü sistemlere ISE kriterine göre denetleyici tasarımı gerçekleştirmişlerdir [54].

Bu çalışmada da PSO, CS ve FireFly Algoritmaları ile sistem modelleme ve denetleyici tasarımı yapılmış ve kullanım kolaylığı olması için bir arayüz programı geliştirilmiştir.

1.2. Tezin Amacı

Tezin amacı literatürde sık kullanılan farklı tipteki ölü zamanlı sistemlerin Parçacık Sürü Optimizasyonu, Guguk Kuşu Algoritması ve Ateş Böceği Algoritmaları ile modellenmesi ve ardından modeli elde edilen bu sistemlere uygun denetleyici tasarımı yapmaktır. Yine modellenen ve denetlenen sistemlerin performans karşılaştırmalarını yapmakta tezin amaçları arasındadır. Bu işlemleri daha kolay yapabilmek ve sonuçları karşılaştırabilmek için ayrıca kullanıcı dostu bir arayüz programı tasarımı da yapılmıştır.

1.3. Tezin Planı

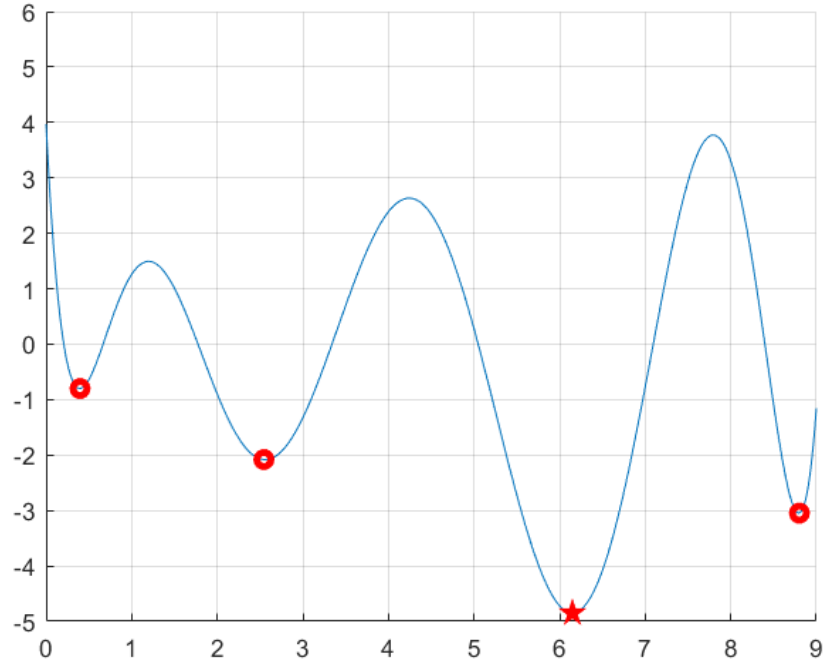
İlk bölümde tez içeriği verilerek giriş yapılmış ve ardından bu konuda literatürde yapılmış çalışmalar hakkında bilgiler verilmiştir. Bölüm 2’de tezde kullanılan sezgisel

optimizasyon algoritmaları tanıtılmış ve nasıl çalıştıkları hakkında bilgi verilmiştir. Ayrıca bu bölümde kullanılan amaç fonksiyonları hakkında da bilgi verilmiştir. Bölüm 3’de ölü zamanlı sistemlerin modellenmesinde kullanılan bazı tekniklerden bahsedilmiştir. Bölüm 4’de ölü zamanlı sistemlerin denetlenmesinde kullanılan tekniklerden bazıları verilmiştir. Bölüm 5’de de ölü zamanlı sistemlerin PSO, CS ve FA algoritmaları ile modellenmesi ve denetlenmesi için oluşturulan bir toolbox tanıtılmış ve örnek uygulamalar verilmiştir. Bölüm 6’da ölü zamanlı sistemlerin farklı model yapıları için modelleme bazı sistemler diğer çalışmalar ile kıyaslanmıştır. Ardından optimizasyonla elde edilen denetlenmiş olan sistemin cevapları verilmiştir. Bununla birlikte sistem için tasarlanan Zeigler Nichols, AMIGO ve Smith öngörücülerinden elde edilen sonuçlar tablo ve sistem yanıtları halinde verilerek sonuçları optimizasyonla elde edilen PI ve PID denetleyiciler ile karşılaştırılmıştır. Ardından bu tezle alakalı ve daha sonraki yapılabilecek araştırmalar için öneriler verilmiştir.

BÖLÜM 2. OPTİMİZASYON ALGORİTMALARI

Optimizasyon, verilen koşullar altında en iyi sonucu elde etme eylemidir. Herhangi bir mühendislik probleminden yol planlamaya kadar her alanda bir optimizasyon mevcuttur. Dolayısı ile optimizasyon, belirli kısıtlar altında (bu kısıtlar genellikle zaman, kaynak veya kalite olabilmektedir) en iyi çözümü elde etme sanatıdır da denilebilir. Kâinata bakıldığında en iyi optimizasyonun kendisinde olduğu görülmektedir. Örneğin iki nokta arasındaki sıvı farkına bakıldığında en az enerji harcayacak şekilde sıvı denge noktasına geldiği görülmektedir. Kısacası optimizasyon, suyun gidebileceği farklı alternatif yollar arasında en iyi yolu seçmesi işlemidir. Analitik olarak geliştirilen Newton, Lagrange, Lineer programlama, dinamik programlama, nonlinear programlama gibi optimizasyon metodlarının yanında, son zamanlarda doğadaki canlı veya cansız varlıkların davranışları modellenerek geliştirilmiş algoritmalarda bulunmaktadır. Günümüzde ise geliştirilen bu algoritmalara örnek olarak Genetik algoritmalar (GA), Simulated Annealing (SA), Ant Colony Optimization (ACO), Partical Swarm Optimization (PSO), Differential Evolution (DE), Harmony Search (HS), Artificial Bee Colony (ABC), FireFly Algorithm (FA) ve Cuckoo Search (CS) örnek olarak verilebilir. Bu algoritmaların birbirlerine göre üstünlükleri olduğu gibi dezavantajları da vardır. Bu üstünlükler ve dezavantajlar uygulanan probleme göre değişmektedir. Literatürde herhangi bir optimizasyon algoritmasının başarısını test etmek için birçok benchmark problem olarak isimlendirilen ve çözülmesi zor olan test problemleri kullanılmıştır. Dolayısı ile her algoritma her test probleminde yapısı itibari ile başarılı olamayabilmektedir. Bu nedenle çözülmesi istenilen probleme uygun optimizasyon tekniği seçilmelidir. Bu da problemin çözülmesini daha hızlı ve daha hassas hale getirecek ve en önemlisi daha doğru sonuçlar üretecektir [55].

Optimizasyonda kullanılan amaç fonksiyonları her probleme göre deęişebilmektedir. Ama genel olarak bakıldığında bunun en önemli kısmı minimum veya maksimum yapılmak istenen amaç fonksiyonunun belirlenmesidir. Örnek olarak vermek gerekirse bir firma için maliyet veya kar fonksiyonu olarak düşünülebilir. Maliyet minimum yapılmaya çalışılırken, kar maksimum yapılmaya çalışılmaktadır. Bunlar yapılırken böyle bir işyerinde hammadde tedarigi, lojistik maaliyetler, çalışan sayısı, çalışanların giderleri, işyerinin sabit giderleri, üretiminin kapasitesi gibi kısıtlar mevcuttur. Burada kontrol edilen deęişken olarak pek çok kısıt kullanılarak minimum maliyet elde edilebilir. Bu yapılırken önemli olan global olarak en minimumu elde etmektir. Şekil 2.1.'de görülen grafik üzerinde birden çok minimum noktası olmasına rağmen, sadece yıldızla gösterilen bir global minimum noktası bulunmaktadır. Dolayısı ile burada önemli olan bu grafięi veren fonksiyonu minimum yapar iken bu global minimum noktasının bulunmasıdır.



Şekil 2.1. Bir fonksiyonda global ve lokal minimum noktalar

Genel olarak optimizasyon problemi tanımlanırken belirli amaç ve kısıtlardan oluşmaktadır. Kimi uygulamalarda amaç yakıtı minimum yapacak araç hızı olurken, kiminde ise zaman göre malzeme üzerindeki gerilimi veya veri akışını maksimum

yapan bir fonksiyondur. Bunlar minimum yapılmak istenen amaç fonksiyonları iken, kısıtlar ise bu fonksiyon minimum yapılırken sağlanması gereken şartlardır. Örnek olarak aracın minimum yakıt harcaması bir amaç fonksiyonu iken, sürekli aynı hızda gitmesi fonksiyonun bir eşitlik kısıtı olmaktadır. Ya da veri akışını maksimum yapmak bir amaç fonksiyonu iken, veri yolunun kapasitesi bir eşitsizlik kısıtı olmaktadır. Bir optimizasyon probleminin genel tanımı Eşitlik (2.1)'de verilmiştir. Burada minimum yapılmak istenen $f(x)$ fonksiyonu iken, $h(x)$ eşitlik kısıtları ve $g(x)$ eşitsizlik kısıtları olarak tanımlanmaktadır.

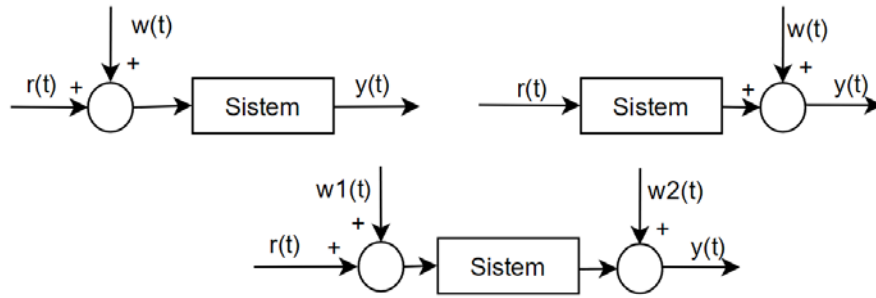
$\min f(x)$

$$h_i(x) = 0 \quad i = 1, 2, 3, \dots$$

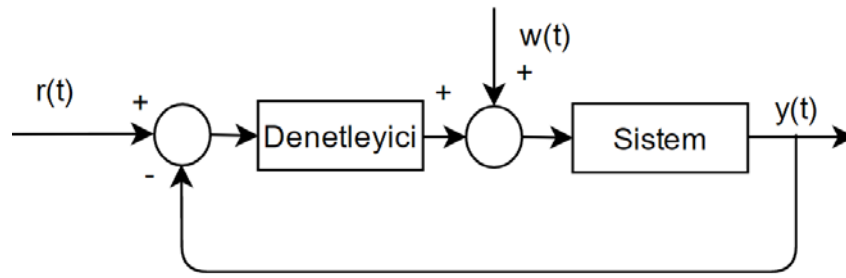
$$g_i(x) \leq 0 \quad i = 1, 2, 3, \dots$$

(2.1)

Bu çalışmada iki farklı problem çözümü üzerinde durulmuştur. İlki Şekil 2.2.'de görülen sistemlerin girişi ve çıkışı arasındaki hatayı minimize ederek, sistem modelindeki parametreleri belirleyecek fonksiyondur. Diğeri ise Şekil 2.3.'de görülen sistemin, istenilen performansa bağlı olarak girişi ile çıkışı arasındaki hatayı minimize ederek, sisteme uygun denetleyici parametrelerini belirleyecek fonksiyondur.



Şekil 2.2. Açık döngü kontrol sistemi



Şekil 2.3. Geri beslemeli kontrol sistemi

Genellikle denetleyici tasarımında yükselme zamanı, yüzde aşım, oturma zamanı gibi performans değerleri dikkate alınmaktadır. Tasarım yapılır iken bu kriterler arasında seçim yapmak gerekmektedir. Diğer taraftan bu kriterlerden bazılarında iyileştirmeler yapılırken, bir diğerinin değerinde kötüleşmeler olabilmektedir. Örneğin yükselme zamanını düşürmek için sistem kazancı artırıldığında, cevabın yüzde aşım miktarı artmaktadır. Dolayısı ile bu kriterler arasında en optimum noktayı bulmak büyük öneme sahiptir. Literatüre bakıldığında çözüm olarak sistemin giriş ile çıkış sinyalleri arasındaki hata sinyalinin yukarıda değinilen tüm performans ölçütlerini taşıdığı, dolayısı ile bu hata fonksiyonunun minimum olduğu noktanın tüm performans değerleri için optimum nokta olacağı belirtilmiştir. Dolayısı ile bunu kolay hale getiren, her sistem için kullanışlı, ayarlanabilir ve pratik açıdan bakıldığında numerik veya analitik olarak hesaplanabilen performans indeksleri önerilmiştir. Yaklaşık 70 yıl önce önerilen ve sistem cevabının geçici ve kalıcı davranışını düzenleyen Hatanın Karesel İntegrali (ISE) performans indeksi Eşitlik (2.3)'de verilmiştir. Bu performans indeksinin kontrol sistemlerinde geçici durum ve kalıcı durum için iyi sonuç verdiği görülse de, hatayı minimum yapan salınımlı bir çıkış verdiği bilinmektedir. Bunun gibi çalışan Hatanın Mutlak İntegrali (IAE) Eşitlik (2.2)'de verilmiştir. Bu performans indeksi salınımlı çıkış verdiği gibi numerik olarak işlem yapmayı daha zorlaştırmaktadır. Bunun için salınımları azaltan ve zamanla artan bir fonksiyon olan Hatanın Zamanla Karesel İntegrali (ITSE) Eşitlik (2.5)'de ve Hatanın Zamanla Mutlak İntegrali (ITAE) Eşitlik (2.4)'de verilmiştir. Bu fonksiyonlar zaman ilerlediğinde bir hata meydana geliyorsa amaç fonksiyonunun değerini arttırmaktadır. Yani bu hatayı genele yaymaktadır. Çünkü ilk başlarda ISE için hata çok büyük, lakin ilerleyen zamanlarda salınımlar olsada bu hata ilk başa göre yine küçük kalmaktadır. Bu sonradan meydana gelen hataları minimum yapacak, zamanın moment fonksiyonları

da kullanılmaktadır. Bu hata daha fazla zamana bağılı olması istendiğinde, Eşitlik (2.6) 'de Hatanın ve Zaman'ın Karesel İntegrali (IT2SE) kullanılabilir. Bu amaç fonksiyonları için örnek davranışlar Şekil 2.4.'de verilmiştir. Dolayısı ile bu amaç fonksiyonları kullanılarak en optimum sistem parametreleri elde edilmeye çalışılmaktadır. Verilen bu amaç fonksiyonları sistem tanıma için uygun olurken, denetleyici tasarımında pek uygun olamamaktadır. Bu amaç fonksiyonları sistem hatasını düşük yaparken, ölü zamandan ötürü keskin geçişlere neden olabilmektedir. Bu amaçla bu fonksiyonlara türevsel bir ifade eklenerek, keskin değişimlerin önüne geçilmeye çalışılmıştır. Bu amaç fonksiyonlarına ilave edilen bu türevsel hata Eşitlik (2.7)'da verilmiştir.

$$IAE = \int |e(t)|dt = \int |y(t) - r(t)|dt \quad (2.2)$$

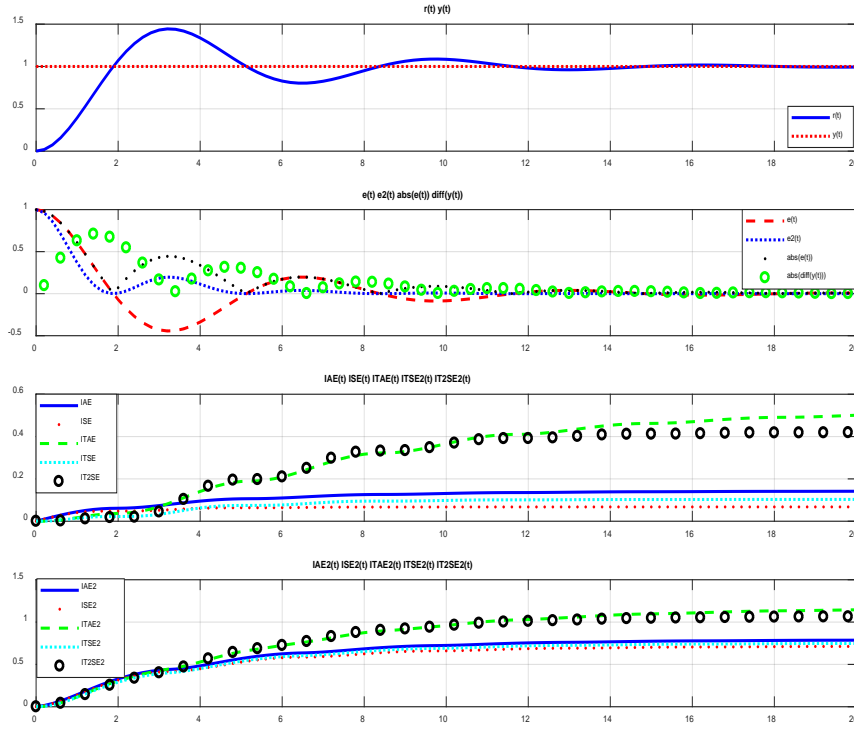
$$ISE = \int (e(t))^2 dt = \int (y(t) - r(t))^2 dt \quad (2.3)$$

$$ITAE = \int t |e(t)|dt = \int t |y(t) - r(t)|dt \quad (2.4)$$

$$ITSE = \int t (e(t))^2 dt = \int t (y(t) - r(t))^2 dt \quad (2.5)$$

$$IT2SE = \int (te(t))^2 dt = \int (t(y(t) - r(t)))^2 dt \quad (2.6)$$

$$\int \text{Abs}\left(\frac{de(t)}{dt}\right) \quad (2.7)$$



Şekil 2.4. ISE, IAE, ITAE, ITSE, ITSE2, ISE2, IAE2, ITAE2, ITSE2 ve IT2SE2'nin hesaplanması

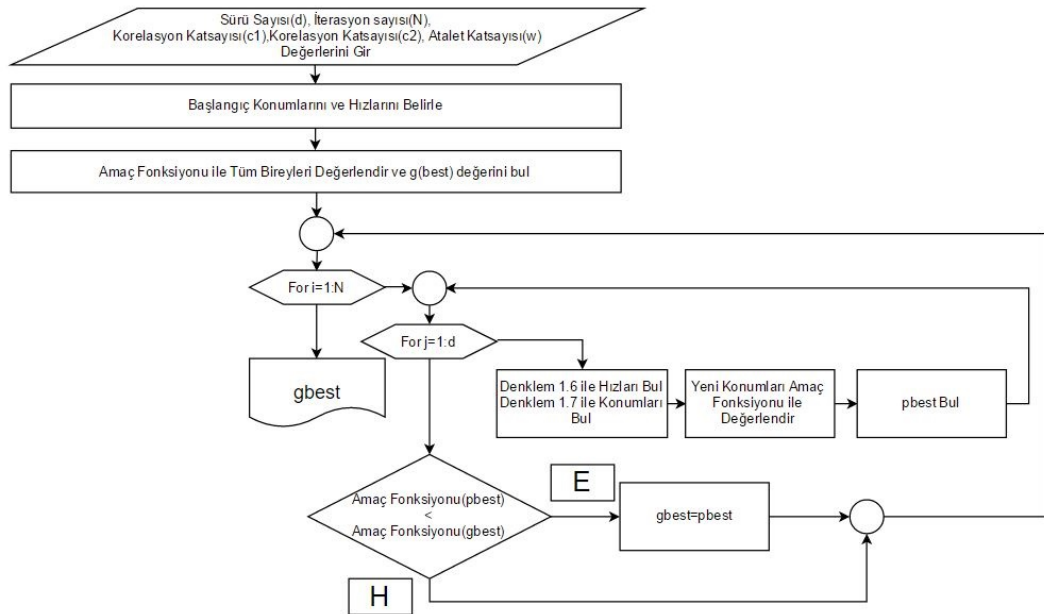
2.1. Parçacık Sürü Optimizasyonu

Kennedy ve Eberhart tarafından geliştirilen Parçacık Sürü Optimizasyonu (PSO), sürülerin davranışlarından esinlenmiştir [12, 13, 56]. Örnek olarak bir balık sürüsünde her bir balığın yiyecek bulabilmesi, avcılardan kaçabilmesi veya çevreye uyum sağlaması gibi davranışlarının modellenerek geliştirildiği bir algoritmadır. Popülasyon temelli olan bu algoritma, sürü içerisinde bilgi paylaşımı ile durumlarını güncellemektedir. Bu sayede neredeyse bütün arama uzayı araştırılarak, en iyi çözüme yakın bir çözüm bulunabilir. Sürü içerisindeki her bir bireyin kendi hafızası bulunmaktadır ve ayrıca en iyi konum değerini de bilmektedirler. Sürü içerisindeki her birey, parçacık olarak adlandırılmakta ve her parçacığın sürü içerisinde konum (p) ve hız (v) bilgileri bulunmaktadır. Bu konum ve hız bilgileri, parçacığın iterasyon içindeki en iyi pozisyonuna (p_{best}), sürü içerisindeki en iyi pozisyonuna (g_{best}) ve o andaki hız bilgisine göre her iterasyon da güncellenmektedir. Bu pozisyon ve hız bilgileri Eşitlik (2.8) ve (2.9)'daki denklemler ile güncellenir [12, 13, 56].

$$v_{t+1} = wv_t + c_1r_1(p_{best} - p_t) + c_2r_2(g_{best} - p_t) \quad (2.8)$$

$$p_{t+1} = p_t + v_{t+1} \quad (2.9)$$

Eşitlik (2.8)'de, r_1 ve r_2 her iterasyon da 0-1 arasında üretilen rastgele sayılardır. c_1 ve c_2 1,8-2 arasında değer alan öğrenme faktörüdür. Eski hıza yakınlığını sağlayan atalet katsayısı w ise 0'a yaklaştıkça eski konumu unutmaya başlamaktadır. Aynı zamanda iterasyon sayısı arttıkça, ataletin değeri genellikle küçültülmektedir. Atalet katsayısının çok küçülmesi de parçacıkların hareket edememesine neden olmaktadır [34, 56]. Nedeni ise iterasyon ilerlediğinde eski konuma göre çok küçük değişimlerin olması beklenirken, eski konuma göre büyük değişim çok iyi sonuç vermemektedir. PSO algoritması Şekil 2.5.'de verilmiştir. Ayrıca temel Matlab kodu Ekler 1'de verilmiştir.



Şekil 2.5. Parçacık sürü optimizasyonu [56]

2.2. Guguk Kuşu Algoritması

Xin-She ve Suash-Deb tarafından geliştirilmiş olan Guguk kuşu Algoritması (Cuckoo Search Algoritması CS), guguk kuşlarının yumurtalarını başka yuvalara bırakarak, kendi yumurtalarını başka guguk kuşlarının büyütmesini sağlamalarından esinlenerek geliştirilmiş bir algoritmadır. Guguk kuşlarının bu içgüdüsel davranışlarına kuluçka parazitliği (broodparasitism) denilmektedir [31]. Bu içgüdüsel davranışları modellemek için belirli kurallar belirlenmiştir [31, 43, 55];

- Bir guguk kuşu rastgele seçtiği bir yuvaya bir tane yumurta bırakabilir.
- Yuva içerisindeki kaliteli yumurta bir sonraki jenerasyona aktarılmaktadır.
- Yuvanın sahibi guguk kuşu, yuvaya geri döndüğünde bırakılmış olan yabancı yumurtayı belirli bir p_a (0,1) olasılığı ile tanıyabilmektedir. Eğer ev sahibi guguk kuşu bırakılmış olan yabancı yumurtayı tanırsa ya kendine yeni yuva kurmak üzere yuvayı terk edecek yada yabancı yumurtayı aşağı atacaktır.

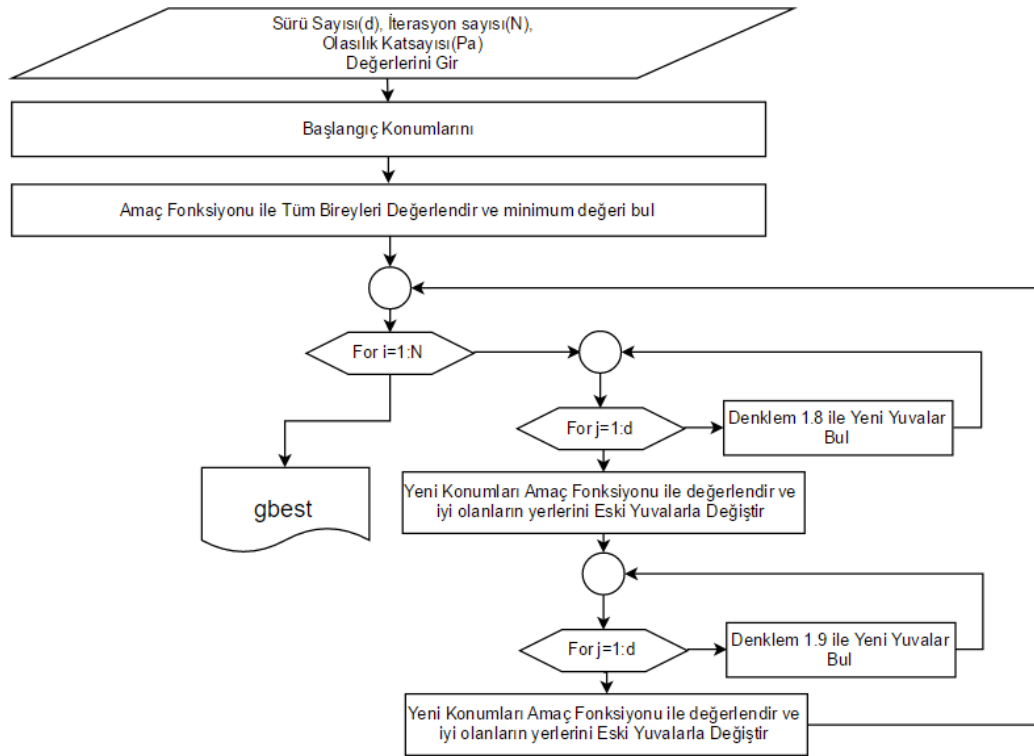
Cuckoo Search algoritması, global ve lokal rassal yürüyüşün dengelenerek kullanıldığı bir algoritmadır. Bu dengeleme de global ve lokal rassal yürüyüşlerin kontrol parametreleri bulunmaktadır ve bu parametreler ayarlanmaktadır. Cuckoo Search algoritmasında küresel rassal yürüyüş, Levy Flight ile gerçekleştirilmektedir [31]. Eşitlik (2.10)'da verilen Levy Flight ile üretilen rastgele değer, α gibi bir değişkenle ağırlıklandırılmaktadır. Ardından bu eski konum bilgisi ile toplanıp yeni konumun bulunması sağlanmaktadır. Bu çalışmada α parametresi 1 olarak alınmıştır. Ayrıca Levy Flight içinde Xin-She'nin önermiş olduğu kontrol parametreleri λ 'dır. Eşitlik (2.10)'da verilen ve küresel rassal yürüyüşü sağlayan λ parametresi genellikle 1 ile 3 arasında alınmaktadır [31]. Bu çalışmada 1,5 olarak alınmıştır.

$$x_i^{t+1} = x_i^t + \alpha \otimes \text{Levy}(\lambda) \quad (2.10)$$

Küresel rassal yürüyüş ile yerel aramayı sağlayan denklem, Eşitlik (2.11)'de verilmiştir. Bu denklemdeki α Eşitlik (2.10)'deki ile aynıdır. s adım uzunluğu olup, değerinin belirlenmesi önem arz etmektedir. Çünkü büyük bir değer seçildiğinde yakınsama olmayabilir. Küçük seçilirse çok fazla iterasyon yapılması gerekebilir [43]. $H(u)$ adım fonksiyonu olup, ϵ parametresi ise gauss dağılımdan gelen rassal reel bir sayıdır. x_j^t ve x_k^t ise çözüm uzayındaki vektörlerinin birbiri ile rasgele değişmesini sağlayan permütasyonlarıdır.

$$x_i^{t+1} = x_i^t + \alpha s \otimes H(p_a - \epsilon) \otimes (x_j^t - x_k^t) \quad (2.11)$$

Cuckoo Search Algoritmasının adımları Şekil 2.6.'da verilmiştir. Ayrıca temel Matlab kodu Ek 2'de verilmiştir.



Şekil 2.6. Cuckoo search algoritması [31]

2.3. Ateş Böceği Algoritması

Xin-She Yang tarafından 2008 yılında önerilen Ateşböceği (FireFly, FA) algoritması, ateşböceklerinin davranışlarından esinlenilerek geliştirilmiş bir algoritmadır. Ateş böcekleri geceleri ışıkları yanıp sönen ılıman ve tropik bölgelerde yaşayan canlılardır. Ateş böcekleri bu biyolojik esanslı olan ışıklarını iki temel amaç için yakıp söndürmektedir. Dişileri ve avlarını cezbetmek için ve ayrıca başka canlılara karşı korunmak içinde ışık yaymaktadırlar. Dolayısıyla bu algoritma, ateş böceklerinin parlaklık ve hareket yönleri modellenilerek geliştirilmiş bir algoritmadır. Ateşböceklerinin bu davranışları, algoritma için kurallaştırıldığında aşağıdaki sonuçlar ortaya çıkar;

- Ateş böceklerinin cinsiyeti olmadığı varsayımı yapılır ve her biri birbirini etkileyebilir.
- Ateş böceklerinin parlaklığı, çekiciliği ile doğru orantılıdır. Parlak olan ateş böceği diğerlerini kendine doğru çeker.
- Ateş böceğinin parlaklığı amaç fonksiyonundan elde edilecek olan değere göre belirlenmektedir.

Ateş böceklerinin parlaklığı, ışık yoğunluğu ve çekiciliği birbirileri ile doğrudan ilişkilidir. Eşitlik (2.12)'de verilen Işık yoğunluğu (I) denklemi, başlangıç ışık yoğunluğuna (I_0), ışığın sabit emilim katsayısına (γ) ve uzaklığa (r) bağlı olarak değişebilmektedir.

$$I = I_0 e^{-\gamma r} \quad (2.12)$$

Ateş böceklerinin birbirlerine olan çekiciliği (β), Eşitlik (2.13)'daki gibi uzaklık ile artıp azalabilmektedir. Denklemdaki başlangıç değeri (β_0), iki ateşböceğinin arasındaki uzaklığın sıfır olduğu andaki değeri göstermektedir. Ayrıca çekicilik Eşitlik (2.14)'deki denklem vasıtasıyla da hesaplanabilmektedir [32].

$$\beta = \beta_0 e^{-\gamma r^2} \quad (2.13)$$

$$\beta = \frac{\beta_0}{1 + \gamma r^2} \quad (2.14)$$

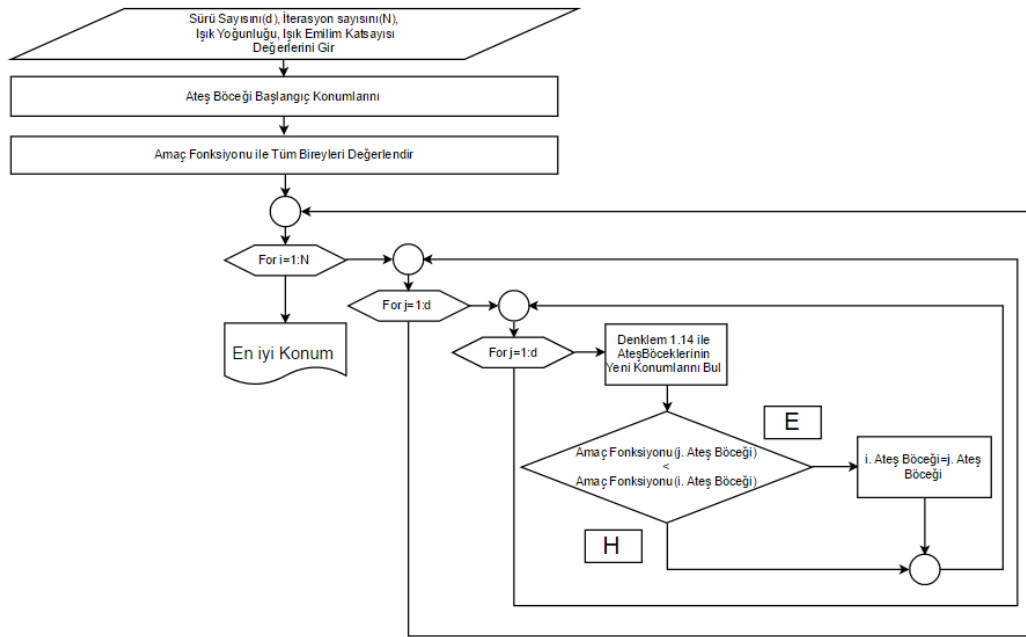
Ateş böceklerinin birbirine olan uzaklıkları hem ışık yoğunluğunu hemde çekiciliği değiştirmektedir. Bu da ateş böceklerinin yapacakları hareketi belirlemektedir. Dolayısı ile ateş böceklerinin birbirleri arasındaki uzaklıkların hesaplanabilmesi için Eşitlik (2.15) kullanılmaktadır.

$$r_{ij} = \|x_i - x_j\| = \sqrt{\sum_{k=1}^d (x_{ik} - x_{jk})^2} \quad (2.15)$$

Ateş böceklerinin bu davranışlarının modellemek için Eşitlik (2.16) kullanılmaktadır. Bu denklemdeki son ifade α ağırlıklıken, ϵ gauss dağılmış rastgele bir değişkendir. Ortadaki ifade ise iki ateş böceği arasındaki farkı çekicilik ile çarparak hareket yönünü tayin etmektedir. İlk ifade ise bir önceki konum olup, bu son iki ifade bunun üzerine eklenerek ateş böceğinin yeni konumu belirlenir [55, 57].

$$x_i^{t+1} = x_i + \beta_0 e^{-\gamma r_{ij}^2} (x_j - x_i) + \alpha \epsilon_i^t \quad (2.16)$$

Ateş böceği algoritmasının adımları, Şekil 2.7.'de verilmiştir. Ayrıca temel Matlab kodu Ek 3'de verilmiştir.



Şekil 2.7. Ateş böceği algoritması [55]

BÖLÜM 3. ÖLÜ ZAMANLI SİSTEMLERİN MODELLENMESİ

Sistem model parametrelerinin belirlenmesinde kullanılan pek çok metot mevcuttur [58]. Gaus'un gezegenlerin yörüngelerini tahmin etmek için geliştirdiği en küçük kareler yöntemi, modeli bilinen bir sistemin parametrelerini hesaplamak için kullanılan en eski ama yaygın olan bir metottur. Bununla birlikte parametrik olmayan ve frekans domeninde tanıma yapmak için Fourier'in geliştirdiği, Fourier serisi veya Fourier dönüşümü gibi teknikler kullanılarak sistemin frekans spektrumu çıkarılabilmektedir [58, 59]. Buna benzer fakat Tynskin'in önerdiği röle tekniği ile nonlinear röle modelinden faydalanılarak, sistem modelinin parametreleri belirli çalışma noktaları için bulunabilmektedir [6]. Bununla birlikte iteratif olarak parametrelerin belirlenmesi de mümkündür. Buna en çok kullanılan Newton tipi arama klasik optimizasyon algoritmaları ile yapmak mümkündür [60]. Hatta bu gradyan yöntemi ile sistemden elde edilen giriş ve çıkış verilerine göre yapay sinir ağları bile eğitilmektedir [61]. Bu klasik gradyan tabanlı algoritmaların lokal noktalara takıldığı ve başlangıç durumuna duyarlı olması nedeniyle buna alternatif olarak önerilen sezgisel algoritmalar kullanılmaya başlanmıştır. Dolayısıyla bu süreç Holland'ın önerdiği genetik algoritmalar ile başlamış ve Kennedy ve Ebheart'de PSO algoritmasını ilk defa yapay sinir ağlarının eğitiminde kullanmışlardır [12]. Dolayısıyla lineer ve nonlinear sistemlerin modellenmesinde bu tip metasezgisel algoritmalar kullanılarak sistemi temsil etmesi sağlanabilmektedir [62-70].

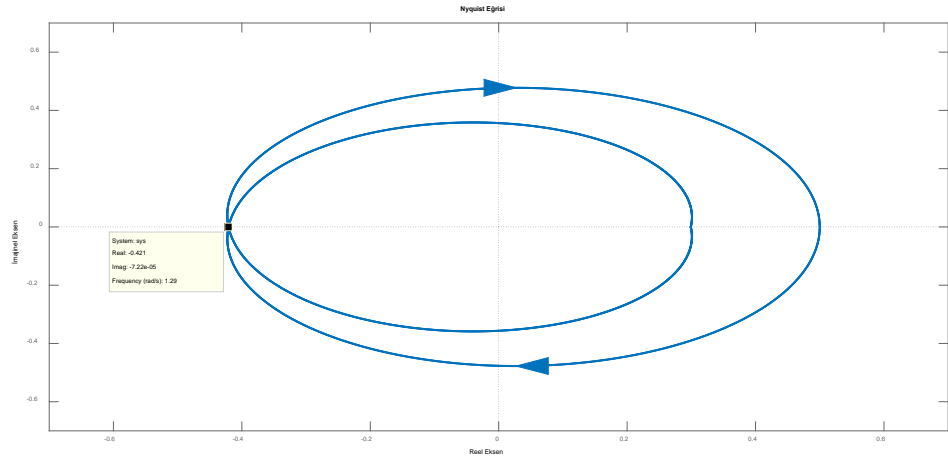
3.1. Auto-tuning Ayar Tekniği

Auto-tuning Tekniği'nin amacı, sistemi Nyquist Eğrisi üzerinde reel olarak -1, sanal (imajinel) olarak 0 noktasına taşıyarak, osilasyona girmesinin sağlanması ve buradan sistemin Kritik Kazanç ve Kritik frekans değerlerinin bulunmasıdır. Ardından, bulunan bu noktaya göre de denetleyici tasarımı yapmaktır. En popüler yöntemlerden biri olan

bu teknik, Astrom ve Hagglund tarafından önerilmiştir. Normal Zeigler-Nichols yönteminde, sistemin kritik kazanç ve kritik frekansını ölçmek için sistemin kazancı arttırılmaktadır. Sistem kazancı arttığında geri besleme ile sistemin osilasyona girmesi sağlanmaktadır. Yada belirli bir orana kadar arttırılarak, sistemin iki salınımı arasındaki orana bağlı olarak denetleyici tasarımı yapmaktır.

Örnek olarak birinci dereceden ölü zamanlı bir sistemin yapısı Eşitlik (3.1)'de verilmiştir. Bu modelde parametre değerlerinin $K = 1$, $\tau = 1$ ve $L = 0.2$ olarak alındığı durumda sistemin Nyquist eğrisi çizildirilir ise, kritik kazanç ve kritik frekans değerlerinin Şekil 3.1.'de görüldüğü gibi reel eksen kestiği noktadaki frekans ve kazanç değerlerine eşit olduğu görülebilir. Auto-tuning tekniği, sistemin bu kritik frekans ve kritik noktasının bulunabilmesini sağlamaktadır.

$$G(s) = \frac{1}{\tau s + 1} e^{-Ls} \quad (3.1)$$

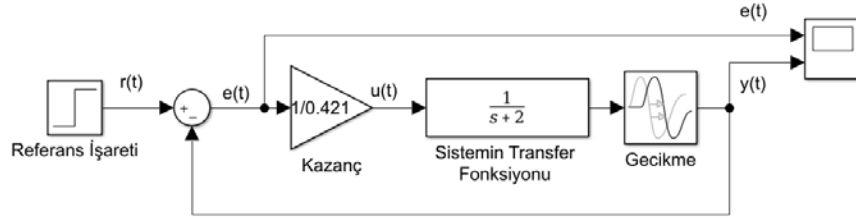


Şekil 3.1. Birinci dereceden bir sistemin kritik kazanç ve kritik frekansı

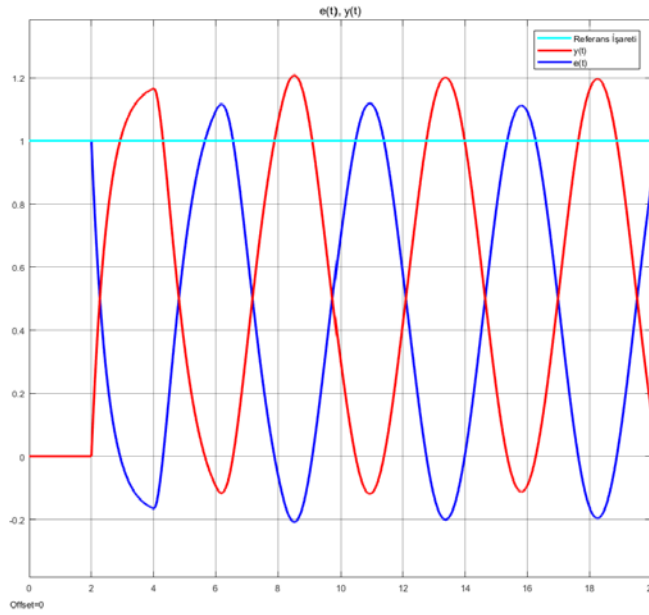
Aynı sistemin kapalı döngüde kritik kazanç ve kritik frekansı bulunmak istenirse, Şekil 3.2.'deki sistem kazancı arttırılarak, sistemin osilasyona girmesi sağlanır. Buradan elde edilen kazanç, Nyquist Eğrisindeki reel eksen üzerindeki noktanın, orjine uzaklığının tersi olacaktır. Sistem kazancının arttırıldığı durumdaki sistemin osilasyon cevabı Şekil 3.3.'de görülmektedir. Sistem cevabının periyodu, iki dalga arasında

geçen süredir. Osilasyon periyodu ile kritik frekans arasındaki ilişki Eşitlik (3.2)'de verilmiştir.

$$\omega_c = \frac{1}{T_c} \quad (3.2)$$



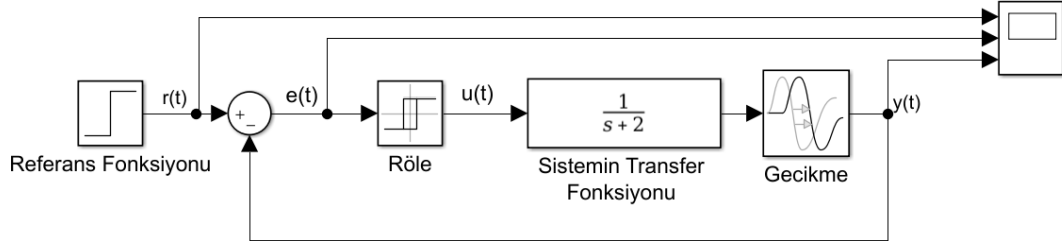
Şekil 3.2. Kapalı döngü sistem



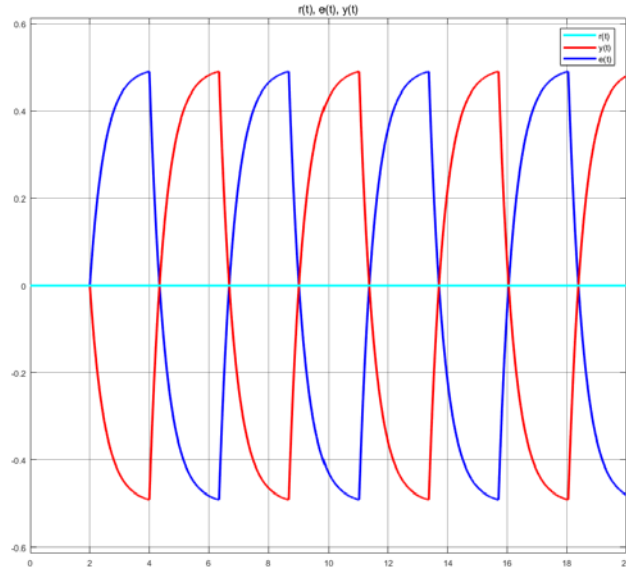
Şekil 3.3. Zeigler Nichols yöntemine göre osilasyon cevabı

Kapalı döngüde kazancı arttırmak her sistem için uygun olmayabilmektedir. Sistem osilasyona girdiğinde sistem kontrol edilemeyip kararsızlığa gidebilmekte ve dolayısı ile zarar görebilmektedir. Yine sistemi bu kritik noktaya taşımak, sistemi gürültü veya sistem belirsizliklerine karşı korumasız hale getirmek ile eş anlamlı olmaktadır. Bu etkileri daha aza indiren bir teknik olan röle tekniği ile sistemin istenilen sınırlar arasında osilasyona girmesi sağlanabilmektedir. Yine bu osilasyon cevabından elde edilen bilgiye göre, sistem parametreleri belirlenebilmektedir. Bu teknikle kullanılan

röle, Şekil 3.4.'de kapalı döngü içerisine yerleştirilmiş şekilde görülmektedir. Referans işareti 0 olmasına rağmen, sistemin ölü zaman içermesinden dolayı röle kullanarak osilasyona gittiği Şekil 3.5.'de görülmektedir.



Şekil 3.4. Röle tekniği ile sistemi ile kapalı döngü sistem



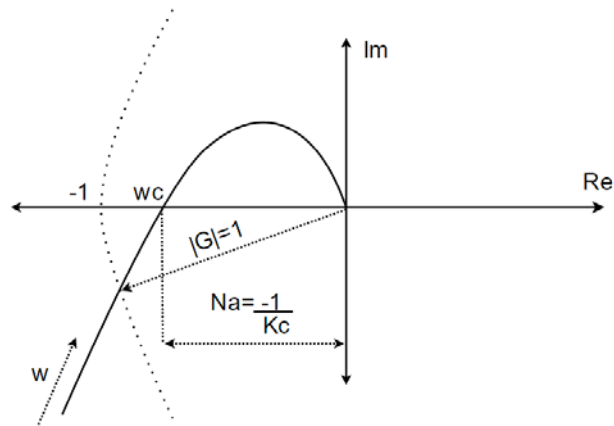
Şekil 3.5. Röle tekniği ile kapalı döngü sistem cevabı

Daha öncede değinildiği gibi röle kullanarak bir sistemin kritik kazanç ve frekansı belirlenebilmektedir. Bunun için sistemin kapalı döngü karakteristik fonksiyonunun belirlenmesi gerekmektedir. Şekil 3.4.'deki sistemin kapalı döngü karakteristik fonksiyonu, Eşitlik (3.3)'de verilmiştir. Buradaki RÖLE(s) rölenin transfer fonksiyonudur. Burada rölenin yapısı değiştirildiğinde, transfer fonksiyonu değişecektir. Dolayısı ile istenilen röle kullanarak sistemin çalışacağı noktaya göre kritik kazanç ve frekans değerleri bulunabilmektedir.

$$RÖLE(s)G(s)+1=0 \quad (3.3)$$

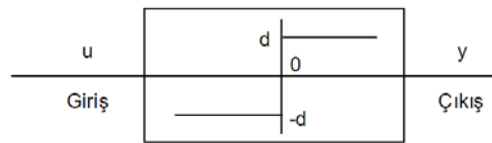
3.1.1. Tek röle testi

Tek röle kullanmak sistemi Şekil 3.6.'daki reel eksene üzerindeki noktaya taşıyarak, sistemin osilasyona girmesini sağlamaktadır. Bu nokta aslında sistemin çalışacağı kritik kazancı vermektedir. Dolayısı ile frekans yönünden bakıldığında, rölenin belirli frekanstaki cevabı önem arz etmektedir.



Şekil 3.6. Nyquist eğrisi

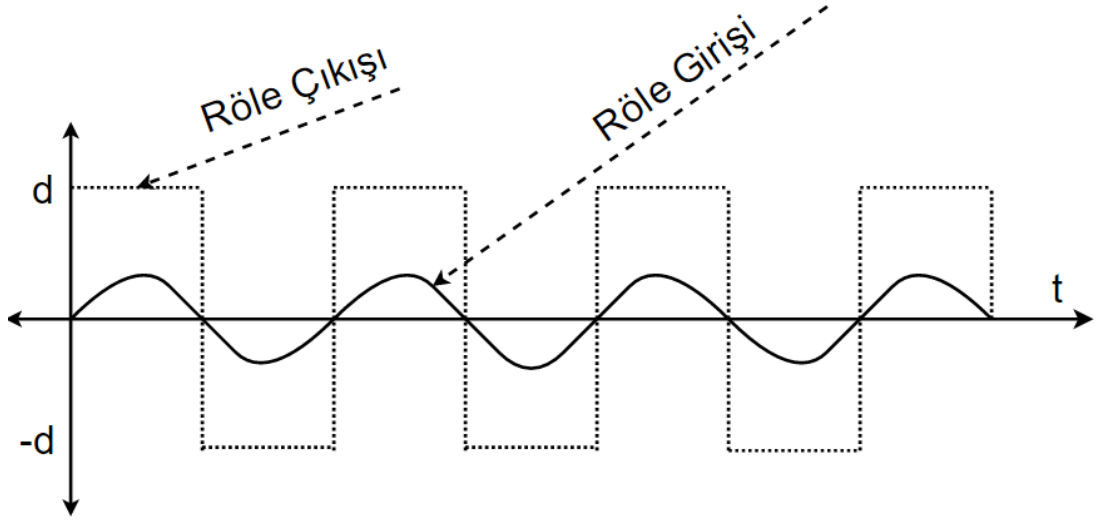
Şekil 3.7.'de rölenin temsili resmi verilmiştir. Eşitlik (3.4)'den rölenin girişine 0'dan büyük bir giriş olduğunda sistem çıkışının d, 0'dan küçük bir giriş uygulandığında ise sistem çıkışının $-d$ olacağı görülmektedir.



Şekil 3.7. Röle yapısı

$$y(t) = \begin{cases} d & \text{eğer } 0 < u \\ -d & \text{eğer } 0 > u \end{cases} \quad (3.4)$$

Eşitlik (3.4)'deki gibi bir rölenin girişine $\sin(\omega t)$ gibi bir sinyal uygulandığında, rölenin çıkışı aynı frekansta fakat d genlikli bir kare dalga sinyali olacaktır. Bu giriş ve çıkış sinyalleri arasındaki bağlantı, Fourier serileri ile ifade edilebilir. Fourier aslında bir sinyalin doğadaki tüm periyodik sinyallerin sinusoidal bileşenlerden oluştuğunu ifade etmektedir. Burada da kare olan çıkış sinyalinin Fourier serisi bulunacaktır.



Şekil 3.8. Sinüs sinyaline karşılık röle çıkışı

Fourier serisinin genel formülü, Eşitlik (3.6)'da verilmiştir. Bu form ortogonal olan sinüs ve cosinüs sinyallerini kullanmaktadır. Bu sinusoidal fonksiyonların ağırlıklandırılmasını sağlayan katsayılar sayesinde, verilen tüm sinyalin frekans domeninde tanımlanabilmesini sağlamaktadır. Dolayısı ile verilen sinyalin hangi frekansları veya harmonikleri içerdiği görülebilmektedir. Bu dönüşüm sayesinde daha fazla veri içeren zaman verisi yerine, onun daha kolay temsilini sağlayan frekans bilgisi kullanılarak temsil edilebilmektedir. Yani Fourier bize, zaman ve frekans domeni arasında kayıplı da olsa bir geçiş sağlamaktadır. Eşitlik (3.6)'da verilen formüle göre, sonsuz sayıda cosinus, sinüs ve bunların katsayıları alındığında, sinyal tam manası ile temsil edilebilmektedir. Lakin bu pratikte çok mümkün olmamaktadır. Formülde verilen a_0 sabit bir sayı olup, bir periyod boyunca sistemin offset değeri olarak düşünülebilir. Eşitlik (3.5)'de işaretin çift olması ve tek olması durumu verilmiştir. Buna göre işaretin tek bileşenlerinin katsayısı a_n , çift bileşenlerinin

katsayısı ise b_n 'dir. Eğer işaret tekil veya çift değilse bu iki katsayıdan işaret oluşturulmaktadır. Fourieri alınan işaret, Şekil 3.8.'deki sürekli periyodik bir kare dalga'dır. Eşitlik (3.7)'den de görüldüğü gibi sinyalin DC bileşeni a_0 ve sinyalin tek bileşeni a_n , 0 olmaktadır. Bunu ifade eden çift katsayı, bileşeni olan b_n değerlerini alacaktır. Eşitlik (3.9)'de b_n değerinin, açılım katsayısı olan n 'ye bağlı olarak sadece tek değerlerinde değer alabildiği gösterilmektedir.

$$\begin{aligned} f(x) &= f(-x) \Rightarrow \text{Çift} \\ f(x) &= -f(-x) \Rightarrow \text{Tek} \end{aligned} \quad (3.5)$$

$$f(t) = a_0 + \sum_{n=1}^{\infty} \left[a_n \cos\left(\frac{2n\pi t}{p}\right) + b_n \sin\left(\frac{2n\pi t}{p}\right) \right] \quad (3.6)$$

$$a_0 = \frac{1}{p} \int_0^p u(t) dt = 0, \quad a_n = \frac{2}{p} \int_0^p u(t) \cos\left(\frac{2n\pi t}{p}\right) dt = 0 \quad (3.7)$$

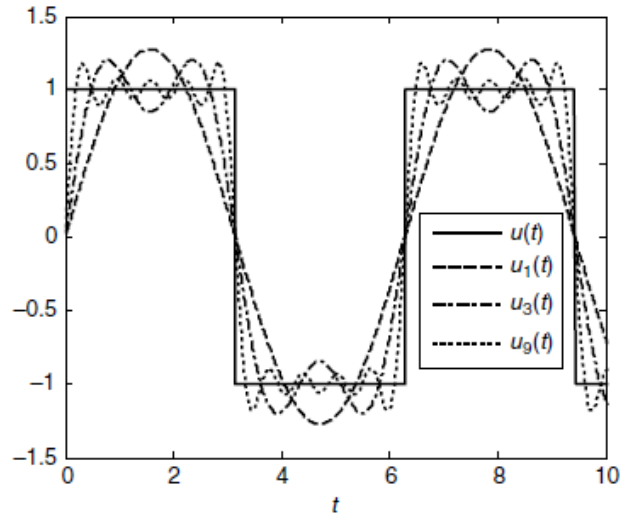
$$b_n = \frac{2}{p} \int_0^p u(t) \sin\left(\frac{2n\pi t}{p}\right) dt = 0 = \frac{2}{p} \int_0^{p/2} d \sin\left(\frac{2n\pi t}{p}\right) dt + \frac{2}{p} \int_{p/2}^p -d \sin\left(\frac{2n\pi t}{p}\right) dt \quad (3.8)$$

$$\begin{aligned} b_n &= \frac{4}{p} \int_{p/2}^p d \sin\left(\frac{2n\pi t}{p}\right) dt \\ b_n &= \begin{cases} \frac{4d}{n\pi} & \text{eğer } n = 1, 3, 5, \dots \\ 0 & \text{eğer } n = 2, 4, 6, \dots \end{cases} \end{aligned} \quad (3.9)$$

Rölenin sinüsoidal işarete vermiş olduğu çıkış sinyali ise, Eşitlik (3.10)'da verilmiştir. Bu eşitlikte açılım katsayısı (n) büyüdükçe, Şekil 3.9.'da görüldüğü gibi $u(t)$ işareti kare dalgaya benzeyecektir.

$$u(t) = \frac{4d}{\pi} \sin(\omega t) + \frac{4d}{3\pi} \sin(3\omega t) + \frac{4d}{5\pi} \sin(5\omega t) + \dots$$

$$u(t) = \sum_{i=1}^n \frac{4d}{(2i-1)\pi} \sin((2i-1)\omega t) \quad (3.10)$$



Şekil 3.9. Fourierin serisinin açılım etkisi

Fakat burada baskın olan işareti alarak, röle çıkışı ilk harmoniğe eşitlenecektir. Dolayısı ile sistem girişi Eşitlik (3.11)'de verildiği gibi $a \sin(\omega t)$ olacaktır. Bu iki sinyal birbirine oranlandığında Eşitlik (3.12)'deki gibi olacaktır.

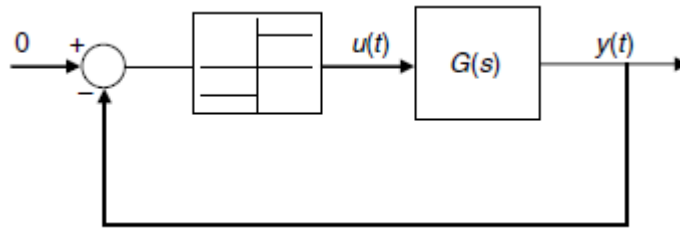
$$u(t) = \frac{4d}{\pi} \sin(\omega t) \quad (3.11)$$

$$N(a) = \frac{u(t)}{y(t)} = \frac{\frac{4d}{\pi} \sin(\omega t)}{a \sin(\omega t)} = \frac{4d}{\pi a} \quad (3.12)$$

Röle içerisinde histeresiz olduğunda, $y(t)$ ve $u(t)$ sinyalleri arasında bir faz kayması olacaktır. Bu da Fourier katsayılarında çiftlik ve tekliği bozacak ve transfer fonksiyonunu değiştirecektir. Sonuç olarak rölenin transfer fonksiyonu, Eşitlik (3.13)'deki gibi olmaktadır.

$$N(a) = \frac{4d}{\pi a^2} \left\{ \sqrt{a^2 - \Delta^2} - j\Delta \right\} \quad (3.13)$$

Bu sistemi kapalı döngüde, Şekil 3.10.'da verilen blok şema olarak gösterebiliriz. Bu sistemin kapalı döngü karakteristik fonksiyonu, Eşitlik (3.14) ile uyuşacaktır. Burada $G(i\omega)$ 'nin sanal kısmı sıfırdır. Ayrıca $\omega_{\text{kritik}} = \frac{2\pi}{P_{\text{kritik}}}$ ve $K_{\text{kritik}} = \frac{4d}{\pi a}$ olarak kabul edilmektedir.



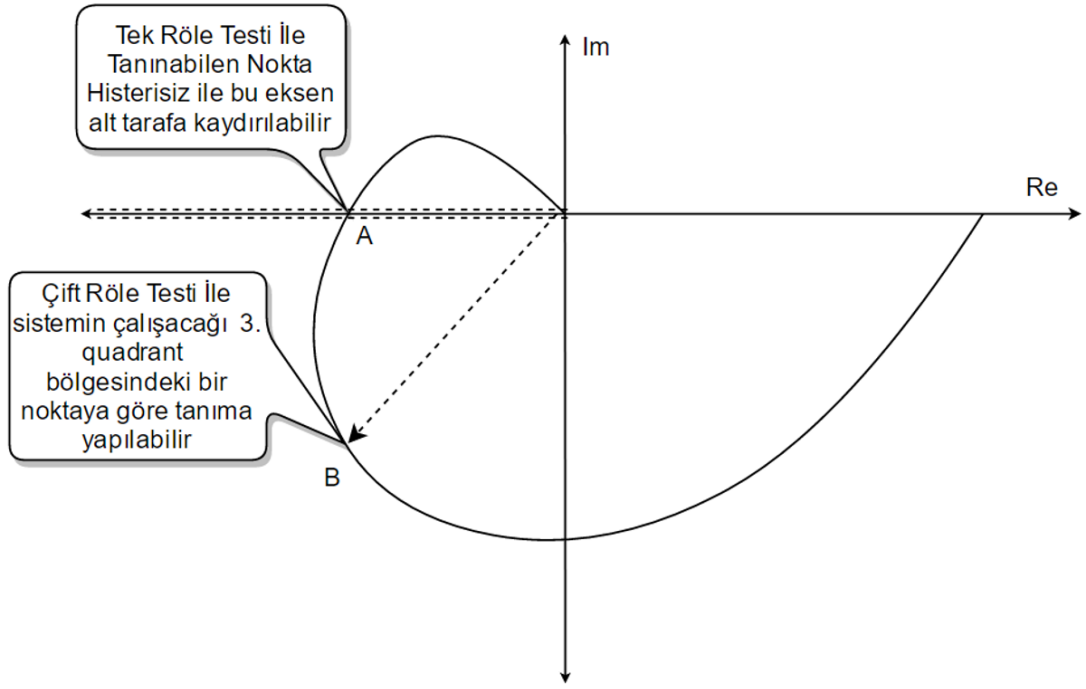
Şekil 3.10. Röle testinin uygulanması

$$N(a)G(i\omega) + 1 = 0 \quad (3.14)$$

$$G(i\omega) = \frac{-1}{N(a)} = -\frac{\pi a}{4d}$$

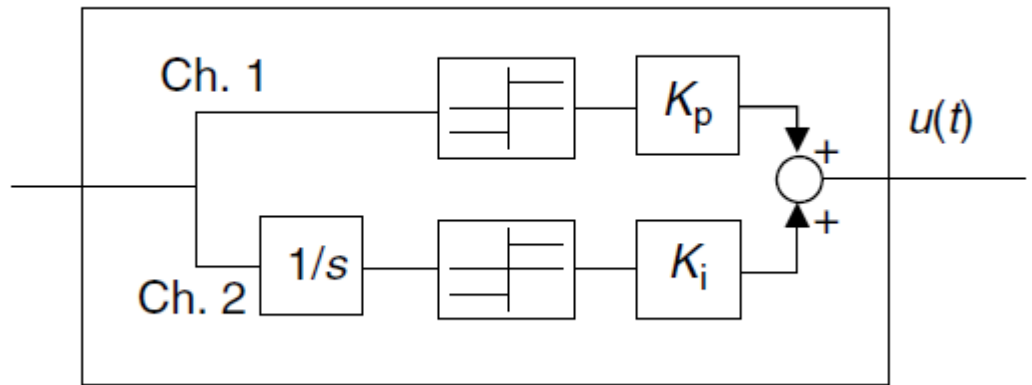
3.1.2. Çift röle testi

Çift röle kullanılarak da sistemin istenilen noktada osilasyona girmesi sağlanabilmektedir. Şekil 3.11.'de görüldüğü gibi tek röle ile A noktasına göre tanıma yapılabilmekte iken, çift röle kullanılan bu teknikle sistemin asıl çalışacağı olan 3. Quadrantta belirlenen B noktasına göre tanıma işlemi yapılabilmektedir.



Şekil 3.11. Tek Röle ve çift rölenin tanıma noktaları

Çift rölenin yapısında iki tane röle olup, çıkış genlikleri ayarlanabilmektedir. Ayrıca bir rölenin girişinde integratör olup, sistem fazının kaydırılması sağlanmaktadır. Bu rölenin Describing Fonksiyonunun bulunması için, yine tek rölenin yapısından faydalanılacaktır.



Şekil 3.12. Çift rölenin bağlantı yapısı

Rölelerin giriş sinyalinin $a \sin(\omega t)$ olduğu durumda, kanal 1'in çıkışı yine kare dalga olacaktır. Dolayısı ile, Kanal 1'de kullanılan rölenin Describing Fonksiyonu Eşitlik (3.15)'de verilmiştir.

$$N(a, K_p) = K_p \frac{4}{\pi a} \quad (3.15)$$

Kanal 2'de kullanılan rölenin girişine yine $a \sin(\omega t)$ sinyali uygulandığında, integratörden dolayı bu sinyal $\frac{-a \cos(\omega t)}{\omega} = \frac{-a \sin(\omega t - \pi/2)}{\omega}$ olacaktır. Yine kare dalgayı bu giriş ile oranlar isek, $\pi/2$ lik bir faz kayması meydana gelecektir. Buda Eşitlik (3.16)'deki gibi ifade edilebilmektedir.

$$N(a, K_i) = K_i \frac{4}{\pi a} \exp(-i\pi/2) \quad (3.16)$$

Lineerlik ifadesinden, girişe karşılık çıkış kanalları toplanabilmektedir. Dolayısı ile tüm Describing Fonksiyonu, Eşitlik (3.17)'de verilmiştir. K_p ve K_i 'nin genliklerine göre sistem fazı kaydırılmakta ve sistemin asıl çalışacağı noktaya göre sistem parametreleri belirlenmektedir.

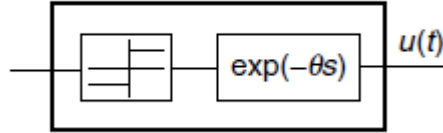
$$\begin{aligned} N(a, K_p, K_i) &= \frac{4}{\pi a} [K_p + K_i \exp(-i\pi/2)] = \frac{4}{\pi a} [K_p + iK_i] \\ &= \frac{4}{\pi a} \sqrt{K_p^2 + K_i^2} \exp \left[-i \arctan \left(\frac{K_i}{K_p} \right) \right] \end{aligned} \quad (3.17)$$

Sistemin karakteristik fonksiyonunu sayesinde parametreleri bulunabilmektedir. Buna göre, Eşitlik (3.18) kullanılarak sistem parametreleri belirlenebilir.

$$\begin{aligned} N(a, K_p, K_i)G(i\omega) + 1 &= 0 \\ G(i\omega) &= \frac{-1}{N(a, K_p, K_i)} \end{aligned} \quad (3.18)$$

3.1.3. Ölü zamanlı röle testi

Röleye ölü zaman eklenerek sistem fazı kaydırılabilmektedir. Bu da yine 3. quadrantta istenilen bir noktaya göre tanıma işlemi yapılabilmesini sağlamaktadır. Şekil 3.13’de görüldüğü gibi rölenin çıkışına, sistem fazını değiştirmeyi sağlayan ölü zaman eklenmiştir.



Şekil 3.13. Ölü zamanlı rölenin yapısı

Rölenin girişine uygulanan $a \sin(\omega t)$ sinyali giriş yaptığında, $u(t)$ sinyali gecikmeli olarak kare dalga çıkışı verecektir. Bu çıkışta yaklaşık olarak, $u(t) = 4 \frac{d \sin(\omega t - \omega \theta)}{\pi}$ olacaktır. Dolayısı ile genlikleri ve fazı arasındaki ilişki Describing Function olarak, Eşitlik (3.19)’de verilmiştir. Bu eşitlik kullanılarak, Karakteristik fonksiyon ile sistem parametreleri arasındaki ilişki Eşitlik (3.19)’da verilmiştir.

$$N(a, \theta) = \frac{4d}{\pi a} \exp(-i\omega\theta) \quad (3.19)$$

Buna göre, Eşitlik (3.20)’de verilen denklem ve belirlenen çalışma noktasının frekansına göre sistem parametreleri belirlenebilir.

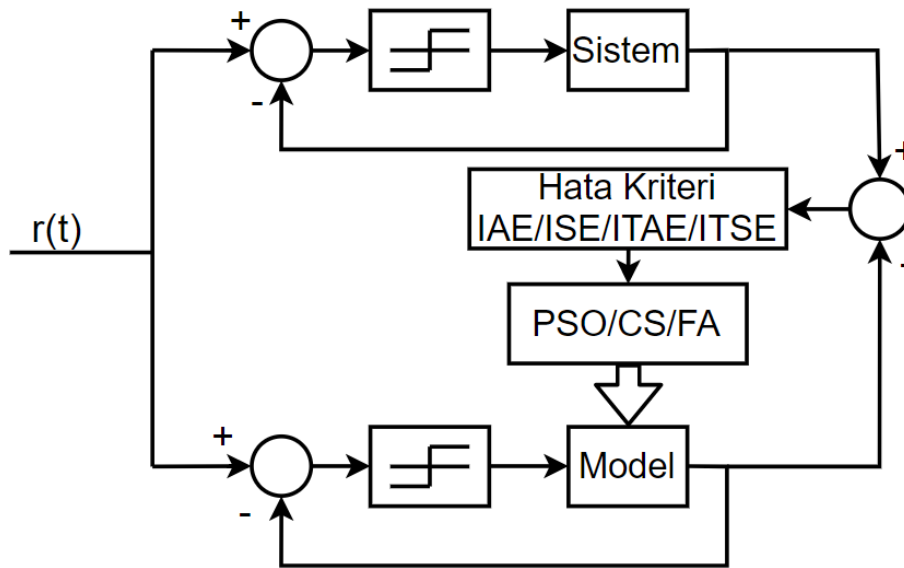
$$N(a, \theta)G(i\omega) + 1 = 0 \quad (3.20)$$

$$G(i\omega) = \frac{-1}{N(a, \theta)}$$

3.2. Optimizasyon Algoritmaları ile Parametre Optimizasyonu

Klasik ve sezgisel optimizasyon teknikleri ile istenilen performans kriterine göre amaç fonksiyonu minimize edilerek, sisteme ait parametreler elde edilebilmektedir. Klasik optimizasyon tekniklerine bakıldığında, kendi içerlerinde prosedürler barındırmaktadır. Kimisi gradyent tabanlı iken, kimisinin ise grid search veya altın oran gibi belirli bir akışı bulunmaktadır. Örnek nokta üzerinden bu prosedürler adım adım yerine getirilmektedir. Fakat örnek nokta lokal bir noktaya yakınsa bu aşamalar bu örnek noktaları lokal noktaya taşıyarak, lokal minimum veren parametreler elde edilmektedir. Bunun için transfer fonksiyon parametrelerini bulunmasında başlangıç değer çalışmaları mevcuttur [33]. Diğer taraftan global arama yapan sezgisel algoritmalar kullanılabilir. Kaya ve Nalbantoğlu'nun yapmış olduğu genetik algoritmalarla geri tepmeli bir sistemin parametreleri, kendi belirledikleri bir performans kriterine göre belirlenmiştir [36]. Farklı bir örnek olarak Yapay Zeka algoritmalarında ağ eğitirken, genellikle çıkışta oluşan hatanın gradyenti alınarak ağın içerisindeki katsayılar güncellenmektedir. Buradaki sorunun aşılması için türevlenebilir aktivasyon fonksiyonları kullanılmaktadır. Buna karşılık Kennedy ve Eberhart PSO algoritması ile ağırlıkların güncellenebileceğini göstermişlerdir. Bununla birlikte başka bir problem olan bulanık modellemede, modelin sistemi iyi bir şekilde temsil edebilmesi için bu tip sezgisel algoritmalar kullanılabilir [71]. Bu tip sezgisel algoritmaların avantajı çoklu nokta içerisinde arama yaparken, hem noktalar kendi etrafına göre değerlerini kontrol ederken hem de en iyi noktaya göre yeni konumları şekillenebilmektedir.

Bu çalışmada da sezgisel algoritmalarından parçacık sürü optimizasyonu, guguk kuşu algoritması ve ateş böceği algoritması kullanılarak röle testine sokulmuş olan sistemin parametreleri belirlenmiştir. Şekil 3.14.'de verilen şekilde sistem ve model röle testine tabii tutulmaktadır. Bu ikisi arasında oluşan hata seçilen amaç fonksiyonuna belirlenmektedir. Elde edilen bu değere göre seçilen optimizasyon algoritmasında offline bir şekilde model parametrelerini belirlemektedir.



Şekil 3.14. Optimizasyon ile model parametrelerinin belirlenmesi

3.3. Farklı Yöntemler

3.3.1. Fourier dönüşümü

Fourier yaklaşımı Bölüm 3.1.1’de söylenildiği gibi, periyodik bir işaretin farklı frekanslardan tekrardan inşa edilmesini sağlamaktadır. Periyodik işaretler Fourier serileri ile ifade edilirken, periyodik olmayan işaretlerde Fourier Dönüşümleri ile ifade edilmektedir.

3.3.1.1. Fourier serisi(limit cycle)

Fourier Serileri Eşitlik (3.6)’da anlaşılır bir şekilde ifade edilebilmektedir. Farklı bir gösterim olan Eşitlik (3.21)’de kullanılarak, yine sinyaller frekans domenine taşınabilmektedir. Sistemin frekans cevabının belirlenmesi için girişe karşılık çıkış sinyalinin genlikleri, sistemin o frekanstaki genlik cevabını verecektedir. Bu form sistemin ilk başlangıçta olan geçici durum cevabını geçip, kalıcı hali olan sürekli haldeki cevabı kullanılmaktadır. Yani sinyal kendini tekrar ettiği zaman kullanılmaktadır. Fakat bakıldığında bu seferde sadece belirli frekansta karşılığı bulunabilmektedir. Dolayısı ile Fourier serisi zamanda kendini tekrar eden periyoda

geçtiği zaman kullanılabilir. Verilen formüllerde t_{ss} işaretin periyodik hale geçiş zamanı iken p işaretin periyodudur [72].

$$\begin{aligned} y(i\omega) &= \int_{t_{ss}}^{t_{ss}+p} y(t)e^{-i\omega t} dt \\ u(i\omega) &= \int_{t_{ss}}^{t_{ss}+p} u(t)e^{-i\omega t} dt \end{aligned} \quad (3.21)$$

$$G(i\omega) = \frac{\int_{t_{ss}}^{t_{ss}+p} y(t)e^{-i\omega t} dt}{\int_{t_{ss}}^{t_{ss}+p} u(t)e^{-i\omega t} dt} \quad (3.22)$$

3.3.1.2. Fourier serisi

Periyodik olan işaretlerin yanında periyodik olmayan işaretlerinde frekans domeninde karşılıkları bulunabilmektedir. Bunun için geliştirilen Fourier transformasyonu, işaretin hangi frekans bileşenlerinin toplamından oluştuğunu ortaya koyan bir yaklaşımdır. Sistemin frekans cevabı için bakıldığında, iki kısma ayrılan bir yapı mevcuttur. İlki geçici hali içine alan kısım, ikincisi ise sürekli periyodik hali içine alan kısımdır. Bu iki kısım Eşitlik (3.23)'da verilmiştir. Verilen formülde t_{ss} işaretin periyodik hale geçiş anı iken p işaretin periyodudur [72].

$$G(i\omega) = \frac{[1 - e^{-i\omega p}] \int_0^{t_{ss}} e^{-i\omega\tau} y(\tau) d\tau + \int_{t_{ss}}^{t_{ss}+p} e^{-i\omega\tau} y(\tau) d\tau}{[1 - e^{-i\omega p}] \int_0^{t_{ss}} e^{-i\omega\tau} u(\tau) d\tau + \int_{t_{ss}}^{t_{ss}+p} e^{-i\omega\tau} u(\tau) d\tau} \quad (3.23)$$

3.3.2. En küçük kareler yöntemi

En küçük kareler yöntemi, bu yöntemi gezegenlerin yörüngesini tahmin etmek için kullanan Gauss'a kadar uzanan bir geçmişe sahiptir. Parametrik model tanıma yöntemlerinden, en kolay yöntem olarak kabul edilebilir. Bu teknikte genellikle sistem modeli olarak ARX, ARMAX gibi farklı model tipleri kullanılmaktadır. Fakat bu çalışmada daha önceden kullanılan ölü zamanlı röle ile sistem parametreleri belirlenecektir. Bu tekniğin yapısı, Eşitlik (3.24)'de verilen formülle gösterilebilir. Bu yapı içerisindeki z ölçülen niceliktir. Eşitlik (3.25)'deki gibi φ M boyutlu bilinen değişkenler, θ ise bilinmeyen değişkenler, dolayısıyla sistem parametreleridir. Genellikle M sistem olduğu için 20-50 arasında alınmaktadır [58].

$$z = \varphi^T \theta \quad (3.24)$$

$$\begin{aligned} \varphi &= [x_1 \quad x_2 \quad \dots \quad x_M] \\ \theta &= [a_1 \quad a_2 \quad \dots \quad a_M] \end{aligned} \quad (3.25)$$

Buradaki problemde, ölçülen büyüklük olan $z_1, \varphi_1, \dots, \varphi_N$ 'yi kullanarak θ olan sistem parametreleri, $\hat{\theta}$ olarak kestirilmektedir. Bu, devamlı olarak çıkış sinyali ölçülerek yapıldığında, Eşitlik (3.26)'daki Z dizisi ve Φ matrisi elde edilmektedir. Bu matrisler kullanılarak, Eşitlik (3.24)'ün daha genişletilmiş hali Eşitlik (3.27)'de elde edilmektedir. Ardından sistem parametrelerinin bulunabilmesi için ise, Eşitlik (3.28) kullanılmaktadır. Normalde direk olarak $\theta = \Phi^{-1}Y$ kullanılabilir, burada matrisin sözde ters olarak kabul edilen $[\varphi^T \varphi]^{-1} \varphi^T$ yapısı kullanılmaktadır. Fakat matris, sistem parametrelerinin çözümlenebilmesi için sistem frekans domenine çevrilecektir.

$$Z = \begin{pmatrix} z_1 \\ \vdots \\ z_N \end{pmatrix} \quad \Phi = \begin{pmatrix} \varphi_1^T \\ \vdots \\ \varphi_N^T \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{m1} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \dots & x_{Nm} \end{pmatrix} \quad (3.26)$$

$$Z = \Phi\theta \quad (3.27)$$

$$\theta = [\Phi^T \Phi]^{-1} \Phi^T Z \quad (3.28)$$

Parametreleri belirlenmek istenen sistemi, en küçük kareler metodunun yapısına çevirmek gerekmektedir. Bunun için temel mantık, sistemleri frekans domenine çevirerek işlemler gerçekleştirmektir. Örnek olarak Eşitlik (3.29)'da birinci dereceden bir sistemin verilmiştir. Frekans domenindeki karşılığı Eşitlik (3.30)'da verilmiştir. Buradan X_i ve Y_i çekilmesi için reel ve sanal kısımlar birbirine eşitlenir. Eşitlik (3.31)'de bu işlemler ifade edilmektedir.

$$G(s) = \frac{Ke^{-Ds}}{\tau s + 1} = \frac{b_0 e^{-Ds}}{a_0 s + 1} \quad (3.29)$$

$$G(j\omega_i) = \frac{b_0 [\cos(\omega_i D) - j \sin(\omega_i D)]}{1 + ja\omega_i} = X_i + jY_i \quad (3.30)$$

$$\begin{aligned} G(j\omega_i) &= M_i e^{jA_i} = M_i (\cos(A_i) + j \sin(A_i)) \\ X_i &= a_0 \omega_i Y_i + b_0 \cos(D\omega_i) \\ Y_i &= -a_0 \omega_i X_i - b_0 \sin(D\omega_i) \end{aligned} \quad (3.31)$$

Yapılan röle testinde, Eşitlik (3.32)'de verilen sistemin kritik kazancı ve kritik frekansı belirlenmektedir. Bu sayede X_i , Y_i ve ölü zaman bulunabilmektedir. Buradan hem normal hemde ölü zamanlı bir test yaparak, Eşitlik (3.33) elde edilip Eşitlik (3.28) kullanılarak sistem parametreleri bulunabilir [10].

$$\begin{aligned} M_i &= 1 / K_U \\ A_i &= -\pi \end{aligned} \quad (3.32)$$

$$\varphi = \begin{bmatrix} w_1 Y_1 & \cos(w_1 D) \\ -w_1 X_1 & -\sin(w_1 D) \\ w_2 Y_2 & \cos(w_2 D) \\ -w_2 X_2 & -\sin(w_2 D) \end{bmatrix} \quad (3.33)$$

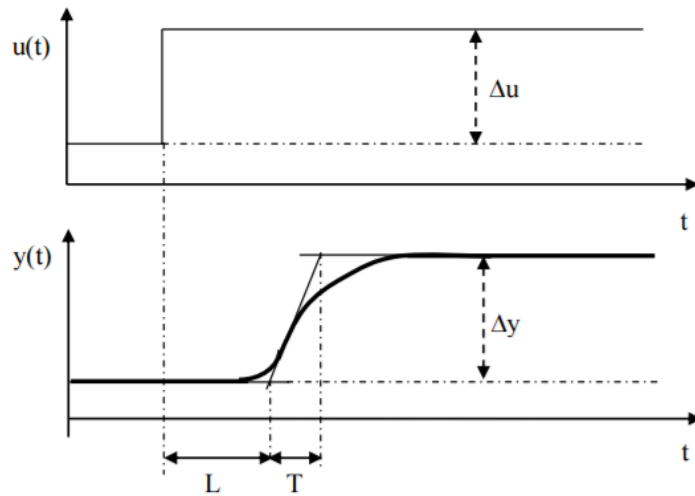
BÖLÜM 4. ÖLÜ ZAMANLI SİSTEMLERİN KONTROLÜ

4.1. Zeigler-Nichols Yöntemi

Bazı sistemlerde transfer fonksiyonunu saptamadaki zorluk, tasarımcıları en uygun denetleyici katsayı değerlerini belirlemenin deneysel yolunu bulmaya itmiştir. En çok kullanılan yöntem Ziegler ve Nichols yöntemidir. John Ziegler ve Nathaniel Nichols isimli iki mühendis 1942 yılında, PID denetleyici parametrelerini belirlemenin iki klasik yöntemini ortaya koydular. Bu iki yöntem, uygulamada özgün yapıda ya da küçük değişikliklerle hala yaygın olarak kullanılmaktadır.

4.1.1. Basamak cevabi yöntemi

Sistemin model parametreleri, Şekil 4.1.'deki açık çevrim cevabı ile bulunabilmektedir. Bu cevap kullanılarak, sistemin parametreleri belirlenmeye çalışılmaktadır.



Şekil 4.1. Basamak yanıtı

Yapılan işlemde, sisteme Δu girişi verilir ve sistemden Δy cevabı alınır. Bu veri ile cevaptaki eğimin en büyük olduğu noktaya göre bir teğet çizilir. Ardından sistemin ölü zamanı ve zaman sabiti şekilde gösterildiği gibi belirlenir. Sonra sistem kazancı, $K = \frac{\Delta y}{\Delta u}$ ile bulunur. Burada amaç, sistem derecesi yüksek dahi olsa sistemi düşük mertebeden ifade etmeye çalışmaktır. Bu parametrelere göre sistem için uygun denetleyici tasarımı, Tablo 4.1. kullanılarak yapılabilir [73].

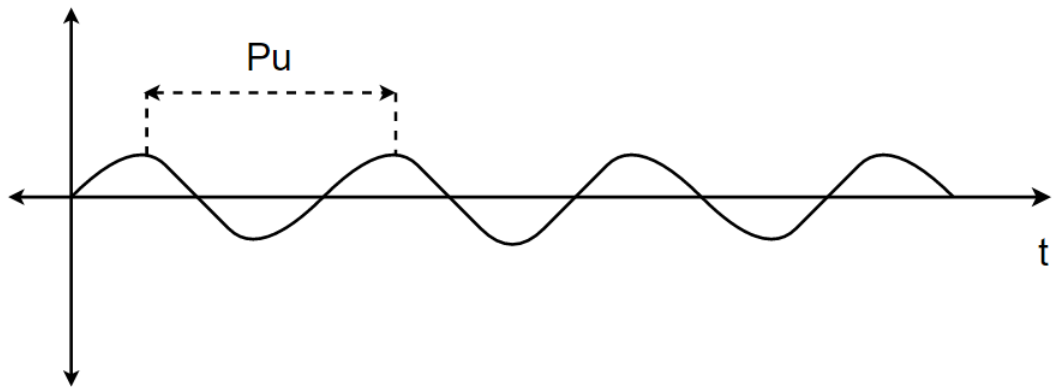
$$\theta = \frac{L}{T} \quad (3.34)$$

Tablo 4.1. Zeigler Nichols Basamak Yanıtına Göre Denetleyici Tasarımı [73]

Denetleyici	K	T_i	T_d
P	$\frac{1}{K\theta}$	-	-
PI	$\frac{0,9}{K\theta}$	3L	-
PID	$\frac{1,2}{K\theta}$	2L	$\frac{L}{2}$

4.1.2. Frekans cevabı yöntemi

Ziegler ve Nichols'un diğer bir yönteminde, denetleyici parametreleri sistemin frekans cevabından faydalanarak elde edilir. Frekans cevabı deneyi, sistemin $-\pi$ fazına kaymasına neden olan periyodun elde edilmesini amaçlar. Basamak yanıtı deneyinde deney, sisteme açık çevrimde uygulanır. Frekans yanıtı yönteminde ise deney, Şekil 3.2.'deki gibi sistemde sadece oransal denetleyici bulunurken gerçekleştirilir. Oransal kazanç, Şekil 4.2.'deki gibi sistem osilasyona girene kadar arttırılarak, sistem kazancı ve sistem periyodu bulunur.



Şekil 4.2. Frekans yanıtı

Yöntemin üstünlüğü, işaretin oldukça kolay oluşturulabilmesidir. Yöntemin sakıncası ise, deneyin kararlılık sınırında yapılmasıdır. Dahası, sistem yanıtının genliği çok büyük olabilir ve deney maliyet ve güvenlik nedenlerinden dolayı yapılamayabilir [4, 27, 72]. Bu işlemler sonucu elde edilen kritik kazanç ve periyot değerleri Tablo 4.2.'de verilen katsayılar ile birlikte kullanılarak, sisteme uygun bir denetleyici tasarımı yapılabilir.

Tablo 4.2. Zeigler Nichols Frekans Yanıtına Göre Denetleyici Tasarımı [73]

Kontrolör	K	T_i	T_d
P	$\frac{K_u}{2}$	-	-
PI	$\frac{K_u}{2,2}$	$\frac{P_u}{1,2}$	-
PID	$\frac{K_u}{1,7}$	$\frac{P_u}{2}$	$\frac{P_u}{8}$

4.2. AMIGO Yöntemi

Literatürde AMIGO tekniğinde ile tasarlanan PI ve PID denetleyicileri, 134 farklı sistem üzerinde denenilerek elde edilmiştir. Buradaki amaç olan denetleyici parametreleri, duyarlılığa (MIGO) ve robustlığa bağlı olarak integral kazancı maksimize edilerek elde edilmiştir. Sistem modeli ve parametreleri basitleştirilerek, denetleyici tasarımı gerçekleştirilir. Bu bağlantılar ile AMIGO ayarlama kuralları

oluşturulmuştur. Temelde iki farklı yöntemi mevcuttur. Birisi basamak cevabı, diğeri frekans cevabı yöntemidir [74, 75].

PID denetleyici tasarımında integral kazancının maksimum olması istenirken, M_s ve M_t duyarlılıkları 1.4'den küçük olacak şekilde denetleyici tasarımı gerçekleştirilmektedir. Bunun için yakınsama yapılarak, sistem parametreleri normalize edilir. Literatürde 134 farklı sistem ile elde edilen sonuçlarda, %15'in altındaki parametrik değişimlerde dahi sistem için uygun denetleyici tasarımının gerçekleştirilebileceği ifade edilmiştir. Basamak cevabı için önerilen PI ve PID denetleyici katsayıları Tablo 4.3.'de verilmiştir.

Tablo 4.3. AMIGO Basamak Cevabına Göre PI PID [73, 76, 77]

	K	T_i	T_d
PI	$\frac{0.15}{K_p} + \left(0.35 - \frac{LT}{(L+T)^2}\right) \frac{T}{K_p L}$	$0.35L + \frac{13LT^2}{T^2 + 12LT + 7L^2}$	-
PID	$\frac{1}{K_p} \left(0.2 + 0.45 \frac{T}{L}\right)$	$\frac{0.4L + 0.8T}{L + 0.1T} L$	$\frac{0.5LT}{0.3L + T}$

Frekans cevabına göre denetleyici tasarımı yapılmak istendiğinde, Zeigler Nichols'deki gibi kritik kazanç ve kritik frekansın bulunması gerekmektedir. Yapılan yakınsamalardan dolayı bu tekniğin uygulanabilmesi için $\kappa = 1 / (K_p K_u) > 0.2$ olması gerekmektedir [77].

Tablo 4.4. AMIGO Frekans Cevabına Göre PI PID Denetleyici [73, 76, 77]

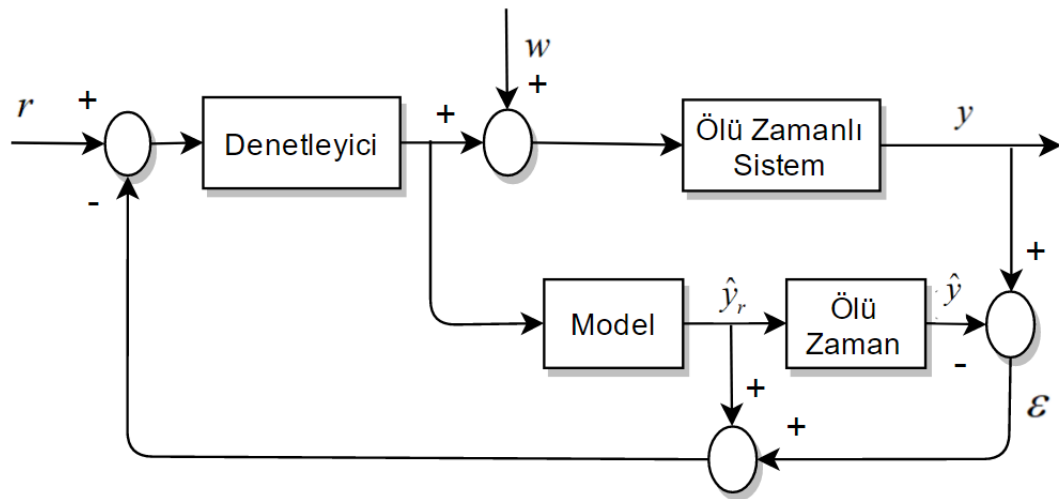
	K	T_i	T_d
PI	$0.16K_u$	$\frac{1}{1 + 4.5\kappa} T_u$	-
PID	$(0.3 - 0.1\kappa^4)K_u$	$\frac{0.6}{1 + 2\kappa} T_u$	$\frac{0.15(1 - \kappa)}{1 - 0.95\kappa} T_u$

4.3. Smith Öngörücüsü

Ölü Zamanlı Sistemlerin denetimi için önerilmiş bir öngörücüdür. Zaman gecikmeli sistemlerde, referans girişi ile sistem çıkışı arasındaki gecikme süresince bir hata

meydana gelmektedir. Ölü zamanı yüksek sistemlere doğrudan klasik PI PID denetleyici tasarımı yapıldığında, sistem genellikle yavaş cevap vermektedir. Aslında PID denetleyicisinde türev operatörü öngörü içermekte fakat denetleyici işareti uygun olmayabilmektedir. Bu nedenle genellikle türev operatörü düşük tutulmaya çalışılmaktadır. Ayrıca zaman sabiti düşük sistemlerde, ani değişimlere karşı veya yüksek frekanslı referans işaretine karşılık kontrol işaretinin çok büyük olmasına neden olmaktadır. Bu nedenden ötürü D operatörü pek tercih edilmemektedir [78].

Ölü Zaman içeren sistemlerde öngörü yapabilmek için kullanılan bir yöntem, Smith Öngörücüsüdür. Bu yöntem sistemin ölü zamanını kompanse etmek için, Smith tarafından öngörülmuş bir tekniktir. Şekil 4.3.'de literatürde önerilen blok diyagram yapısı gösterilmektedir. Bu teknik sayesinde, ölü zaman ortadan kaldırılarak denetleyici tasarımı yapılmasına olanak sağlanmaktadır. Burada önemli olan, sistemin kendisi ile sistem modelinin uyuşmasıdır. Modelin yapısı ve parametreleri sisteme ne kadar yakın olursa, o kadar doğru bir öngörü yapılabilir.



Şekil 4.3. Smith öngörücüsü

Smith Öngörücüsünde denetleyici çıkışından gelen sinyal, sisteme ve modele uygulanır. İdeal durumda bu denetleyici işareti, sistem çıkışına (y) ve ölü zamanlı model çıkışını (\hat{y}) neden olur. Bu ideal durumda ölü zamanlı model ile sistem aynı olduğu için, iki sinyal arasındaki farkın (ε) sıfır olması beklenmektedir. Bu hata sıfır

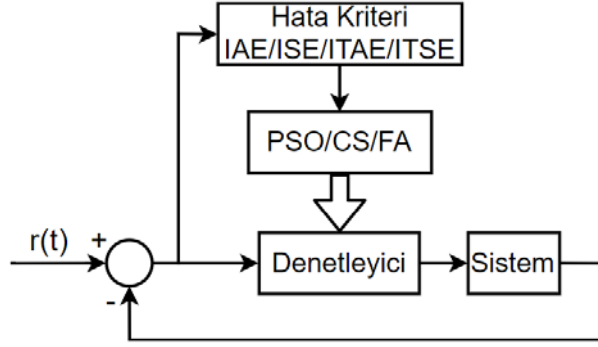
olduğunda ölü zaman içermeyen modelin çıkışı (\hat{y}_r), geri besleme olarak alınır. Referans ile geri beslemenin farkı alınarak, hata sinyali oluşturulur. Bu sinyal denetleyiciye gelerek, denetim işareti oluşturulur. Ayrıca ideal durumda, sistem girişi ve çıkışında bozucu olmadığı varsayımı yapılır. Eğer bozucu olursa, bu durumda model ile sistem uyumsuzluğu meydana gelebilir. Fakat ideal durum düşüldüğünde, model üzerinde istenilen bir teknikle denetleyici tasarımı yapılabilir. Bunun için modelin asimtotik kararlı olması gerekmektedir.

Sistemin denetlenmesi model üzerinden olan Smith öngörücüsü'nde sistem üzerindeki ölü zamanı dğerildiği için ölü zamansız modele herhangi bir kontrol tekniği ile Zeigler Nichols, Choen-Coon, Wang-Juang-Chen, MIGO, AMIGO... vb denetleyici tasarımı yapılabilmektedir. Bu çalışmada da Smith öngörücüsü için Zeighler-Nichols tekniği seçilere ölü zamansız modele denetleyici tasarımı yapılmıştır.

4.4. Optimizasyon Algoritmaları ile Denetleyici Katsayılarının Belirlenmesi

Optimizasyon teknikleri ile yapılan işlemlerde, genellikle belirlenen amaç fonksiyonuna göre denetleyici parametreleri belirlenmeye çalışılmaktadır. Klasik gradyent tabanlı optimizasyon teknikleri de bu tip sistemler için belirlenen amaç fonksiyonunu minimize etmek için kullanılabilirdiği gibi, sezgisel algoritmalarda kullanılabilir. Klasik optimizasyon teknikleri sezgisel metotlara göre hızlı olmasına karşılık, başlangıç değerlerine ve lokal bölgeler konusunda daha duyarlıdır. Uygun belirlenmeyen başlangıç noktası, amaç fonksiyonunu birçok noktada minimum yapan yerlerden sadece birini üretebilmektedir. Bu sorun optimizasyon tekniklerinde istenmeyen bir durum olup, sezgisel algoritmalarda sürekli olarak iyileştirmeler yapılmaktadır. Bu çalışmalar sezgisel olarak genetik algoritmalar ile başlamış olup, yeni geliştirilen algoritmalar ile performansları denenmeye devam edilmektedir. Çünkü sezgisel algoritmalarda en büyük sıkıntı, bir problem için geliştirilmiş olan her algoritmanın iyi sonuç veremeyebilmesidir. Bu çalışmada, geliştirilmiş olan bu tip algoritmaların başarıları değerlendirilmiştir.

Bu çalışmada da Şekil 4.4.'de verilen diyagramda referans işareti ile sistem çıkış arasındaki fark seçilen hata kriterine göre hesaplanmaktadır. Ardından bu değer seçilen optimizasyon algoritmasına ve denetleyici tipine göre minimum yapılmaya çalışılmaktadır.



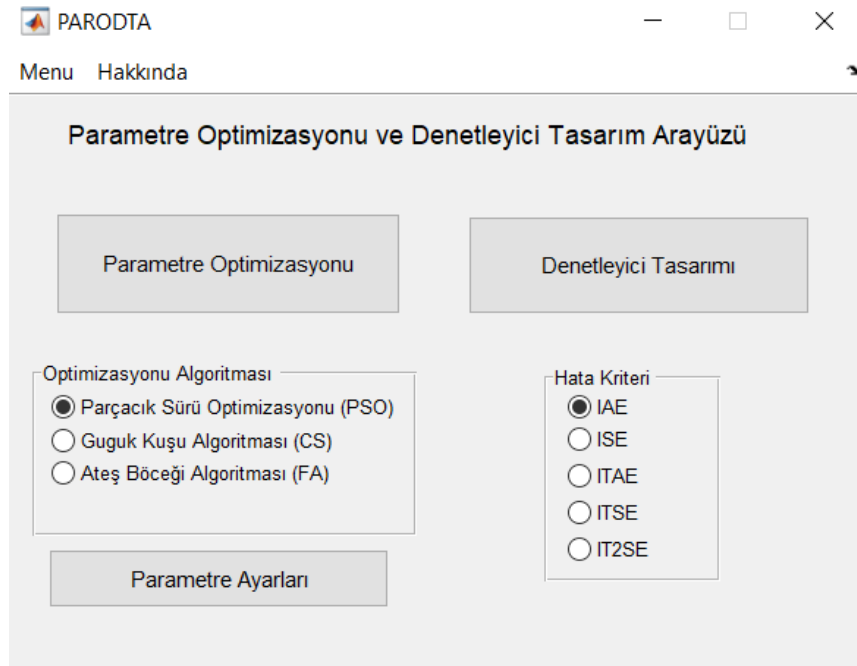
Şekil 4.4. Sezgisel algoritmalar ile denetleyici tasarımı bloğu

BÖLÜM 5. ÖLÜ ZAMANLI SİSTEMLERİN MODELLENMESİ VE DENETİMİ İÇİN TOOLBOX GELİŞTİRME

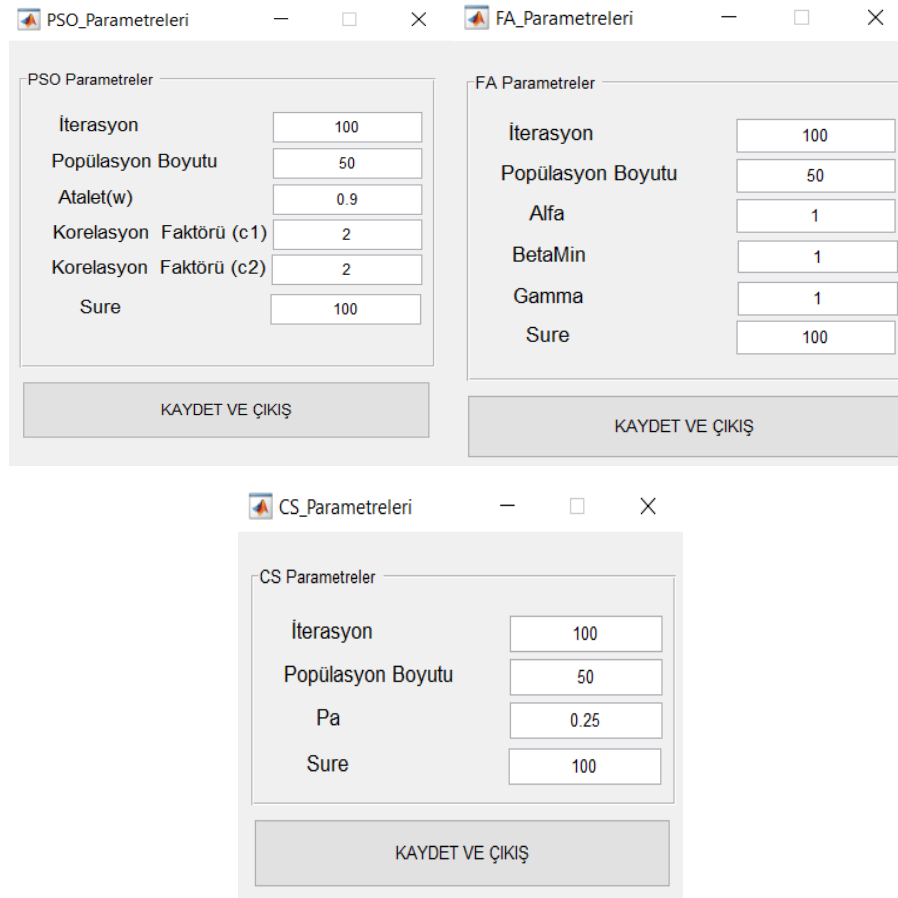
Bu çalışmada, ölü zamanlı sistemlerin modellenmesinde ve denetlenmesinde Ateş Böceği Algoritması, Guguk Kuşu Algoritması ve Parçacık Sürü Optimizasyonu algoritmalarının kullanılması için bir arayüz gerçekleştirilmiştir. Bu arayüz sayesinde, seçilen algoritma ile sistem parametreleri, belirlenen sistem tipine göre optimize edilebilmekte, ayrıca seçilen denetleyici tipine göre sistem kontrol edilebilmektedir.

5.1. Parodta Arayüzü

Oluşturulan Parametre Optimizasyonu ve Denetleyici Tasarım (Parodta) Arayüzü Şekil 5.1.'de verilmiştir. Parodta üzerinde optimizasyonu gerçekleştirecek olan algoritma, Optimizasyon Algoritması kısmından seçilmektedir. Ayrıca minimize edilmek istenen performans indeksi Hata Kriteri bölümünden seçilmelidir. Seçilen algoritmanın parametresi değiştirilmek istendiği takdirde, Parametre Ayarları butonuna tıklanır ve o algoritmaya ait parametre ekranı Şekil 5.2.'deki gibi açılır. Buradan değiştirilen parametreler, Kaydet ve Çık butonuna basılarak değişiklikler kayıt edilir. Sistem için istenen modele uygun parametreler belirlenmek istendiğinde, Parametre Optimizasyonu butonuna tıklanır ve ona ait arayüz açılır. Eğer denetleyici tasarımı yapılmak isteniyorsa, Denetleyici Tasarımı butonuna tıklanır ve parametreler belirlenir.



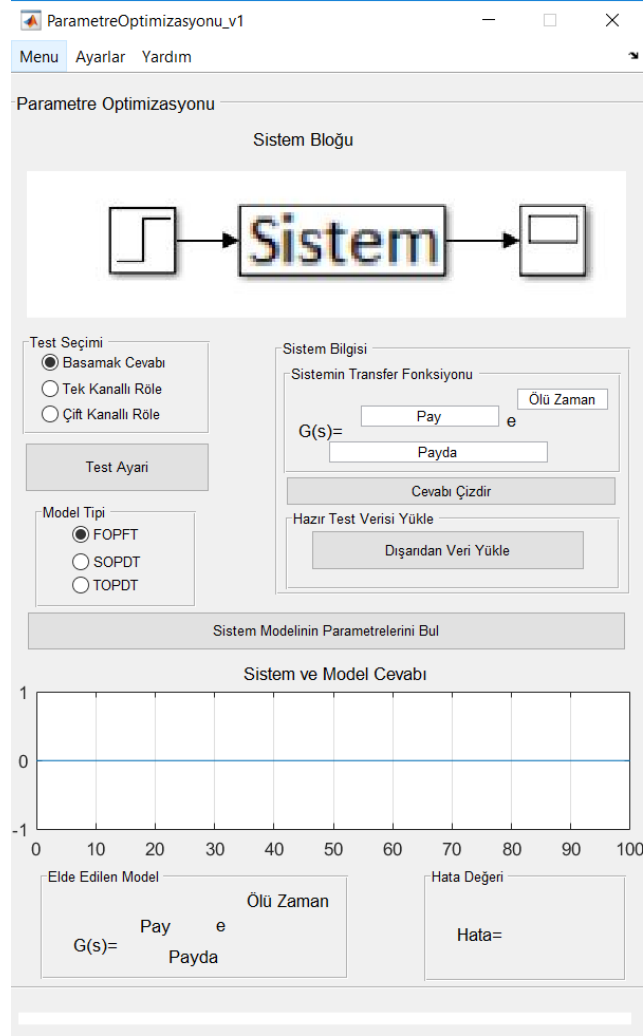
Şekil 5.1. PARODTA arayüzü



Şekil 5.2. PSO CS ve FA parametrelerini değiştirme arayüzü

5.1.1. Parametre optimizasyon arayüzü

Parodta arazüyünde Parametre Optimizasyonu butonuna tıklanıldığında, Şekil 5.3.'deki arayüz açılmaktadır. Bu arayüz üzerinde teste ait ayarlamalar yapılarak, seçili algoritmaya göre parametreler optimize edilmektedir. Dolayısı ile yapılmak istenen Test seçimi kısmında, hangi testin uygulanacağı belirlenmektedir. Test Ayarı butonuna tıklanarak, yapılmak istenen testin parametreleri ayarlanabilir. Tek Kanallı ve Çift Kanallı rölenin ayarları, Şekil 5.4.'deki arayüz ile yapılabilmektedir. Sistem bilgisi kısmından sistemin transfer fonksiyonu payı, paydası ve ölü zaman bilgileri girilebildiği gibi, Dışarıdan Veri Yükle butonu ile sisteme ait dışarıdan veri girişi de yapılabilmektedir. Model Tipi bölümünden de model derecesi indirgemeye benzer bir yapı olan ve Describing Function yapısı kullanılarak sistemin model tipi seçilebilmektedir. Bu ayarlar yapıldıktan sonra Sistem Modelinin Parametrelerini Bul butonuna tıklanarak, seçilmiş olan algoritma ve hata fonksiyonuna göre sistemin parametreleri bulunmaktadır.



Şekil 5.3. Parametre optimizasyonu arayüzü

The image shows two side-by-side screenshots of software windows for configuring role settings. The left window is titled 'TekKanalliRoleAya...' and the right is 'CiftKanalliRoleAya...'. Both windows have a similar layout with three main sections: 'Sistem Ayarlari', 'Röle Ayarlari', and 'Integratörlü Röle Ayarlari'. Each section contains several input fields with numerical values. At the bottom of each window is a 'KAYDET VE ÇIKIŞ' button.

Section	Parameter	Value
Sistem Ayarlari	Başlangıç	0
	Referans	0
Röle Ayarlari	Rölenin Üst Çıkış Değeri	1
	Rölenin Alt Çıkış Değeri	-1
	Histerisiz	0
Integratörlü Röle Ayarlari	Rölenin Üst Çıkış Değeri	0.1
	Rölenin Alt Çıkış Değeri	-0.1
	Histerisiz	0

Şekil 5.4. Tek-Çift kanallı röle ayarı

Şekil 5.5.'de Menu butonundan elde edilen sistemin Bode ve Nyquist eğrileri çizdirilebilir. Sayfadan çıkmak istenildiğinde, Çıkış butonuna basılarak bu ekrandan çıkılabilir. Ayrıca optimizasyon ayarları, Şekil 5.6.'de verilen Ayarlar kısmından değiştirilebilir.

The image shows a menu bar with three tabs: 'Menu', 'Ayarlar', and 'Yardım'. The 'Ayarlar' tab is selected and highlighted. Below it, a dropdown menu is open, showing 'Diyagramlar >' and 'Çıkış'. The 'Diyagramlar >' option is expanded to show two sub-options: 'Bode Diyagramı' and 'Nyquist Eğrisi'.

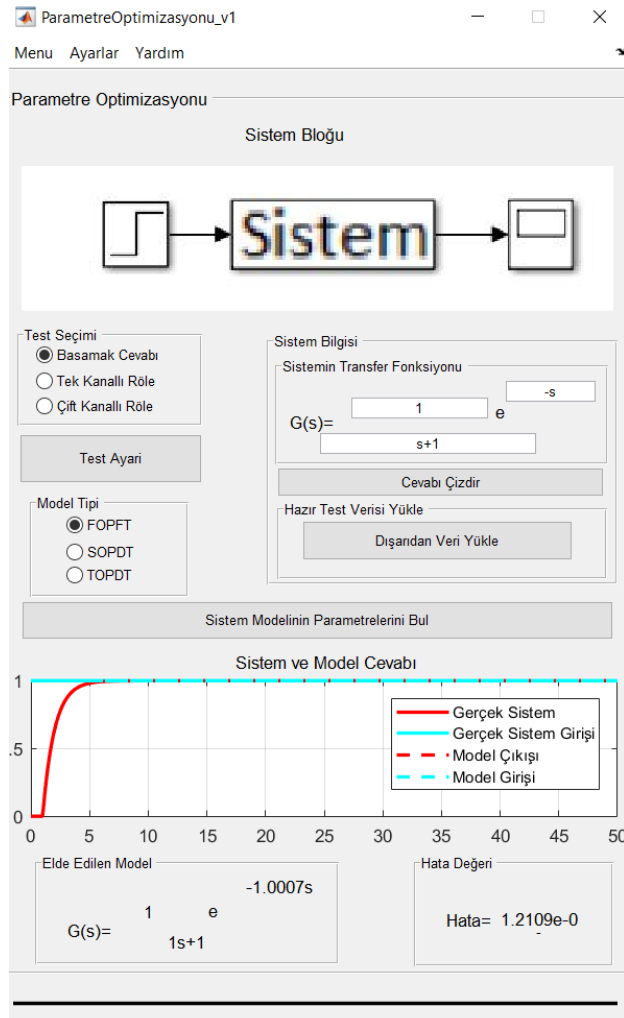
Şekil 5.5. Menu sekmesi

The image shows a menu bar with three tabs: 'Menu', 'Ayarlar', and 'Yardım'. The 'Ayarlar' tab is selected and highlighted. Below it, a dropdown menu is open, showing 'Param' and three sub-options: 'PSO Ayarları', 'CS Ayarları', and 'FA Ayarları'.

Şekil 5.6. Ayarlar sekmesi

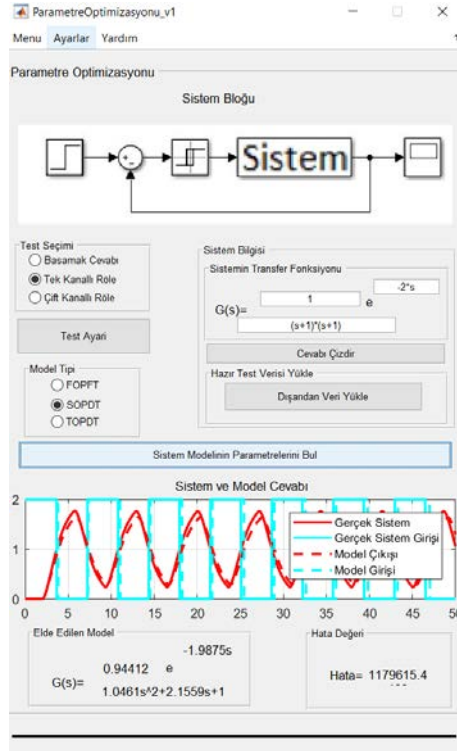
Örnek uygulama olarak Şekil 5.7.'de verilen birinci dereceden bir sistemin parametreleri, basamak cevabına bağlı olarak optimize edilmeye çalışılmıştır. Elde

edilen Model kısmında, sistemin transfer fonksiyonu verilmiş ve Hata Değeri bölümünde de daha önceden belirlenmiş olan amaç fonksiyonuna bağlı olarak elde edilen hata değeri verilmiştir.



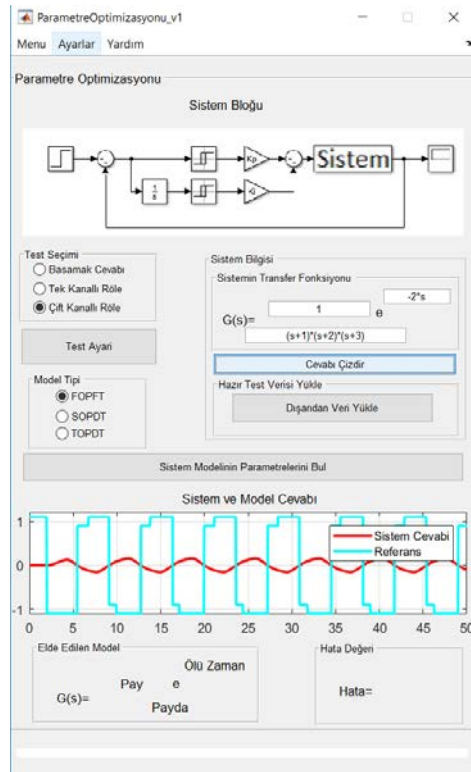
Şekil 5.7. Basamak cevabı örnek uygulama

İkinci örnek uygulama olarak, ikinci dereceden bir sistem seçilmiştir. Bu sistemin parametrelerini bulmak için tek kanallı röle kullanılmıştır. Elde edilen parametreler Şekil 5.8.'deki arayüzde görüntülenmektedir.



Şekil 5.8. Tek kanallı röle testi

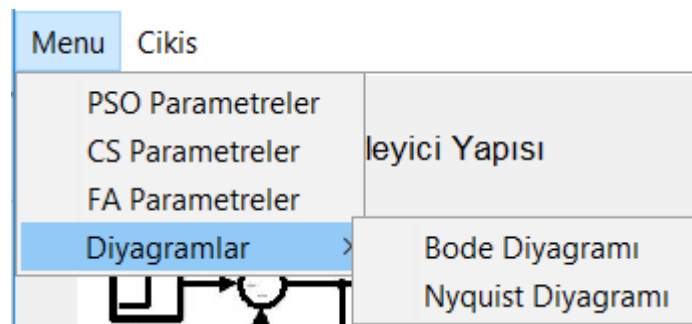
Üçüncü örnek uygulama olarak, üçüncü dereceden ölü zamanlı bir sistem seçilerek, parametrelerini belirlemek için çift kanallı röle kullanılmış ve elde edilen sonuçlar Şekil 5.9.'da verilmiştir.



Şekil 5.9. Çift kanallı röle testi

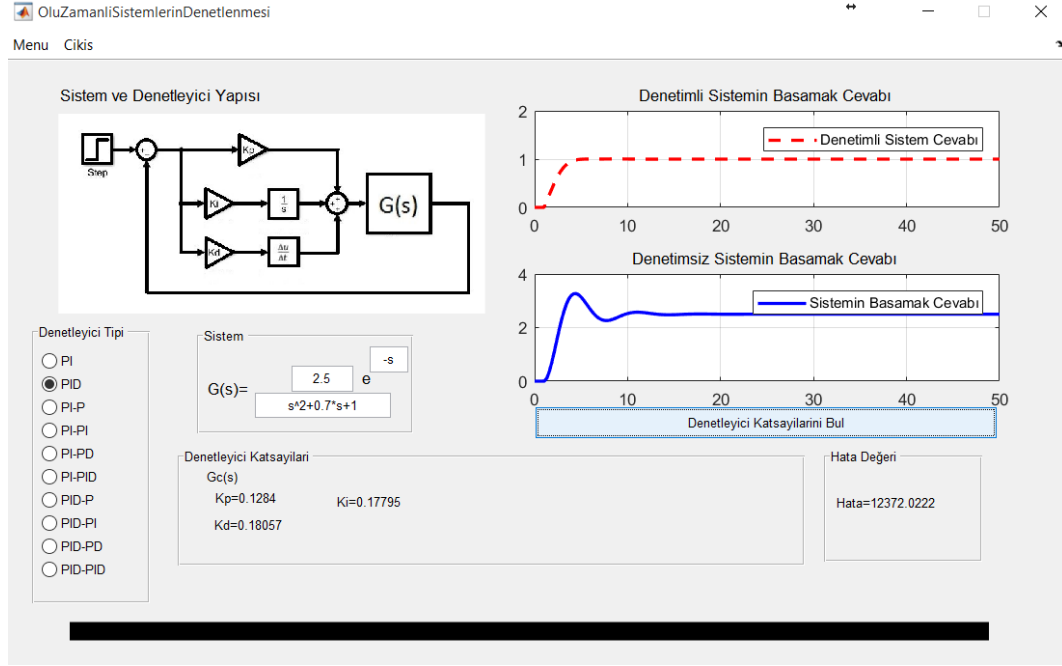
5.1.2. Denetleyici tasarım arayüzü

Denetleyici Tasarım Arayüzü daha önceden belirlenen optimizasyon algoritmasına, performans indeksine ve bu arayüz üzerinde seçilen denetleyici tipine göre sisteme denetleyici tasarımı yapmaktadır. Bu arayüzün Şekil 5.10.'de verilen Menu butonu, algoritma parametreleri değiştirilmek istendiğinde veya elde edilen sistemin Bode diyagramı veya Nyquist Eğrisi çizilmek istendiğinde kullanılabilir.



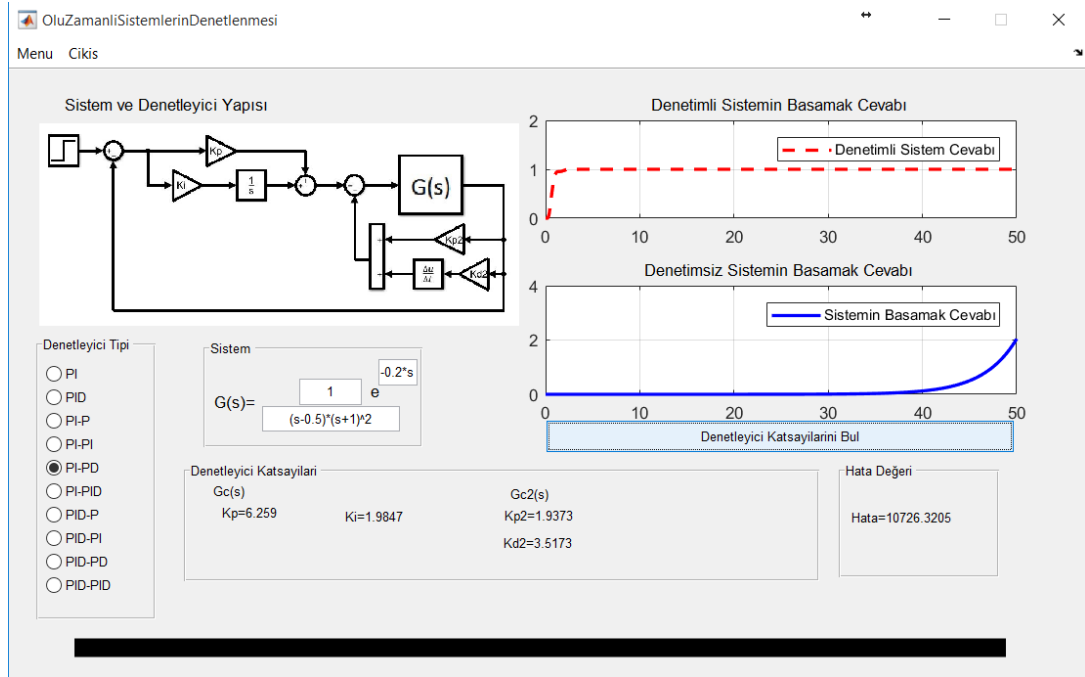
Şekil 5.10. Diyagramlar alt menüsü

Şekil 5.11.'de verilen arayüzün sistem kısmına, sisteme ait transfer fonksiyonun payı, paydası ve ölü zaman bilgisi girilir. Ardından tasarlanmak istenen denetleyici tipi PI, PID, PI-P, PI-PD vb. seçilir. Bu seçim yapıldıktan sonra, Denetleyici Katsayıları butonuna tıklanır. Elde edilen sonuçlar, Denetleyici Katsayıları ve Hata Değeri bölümünde görüntülenmektedir.



Şekil 5.11. Denetleyici tasarım uygulaması

Aynı arayüz üzerinde farklı bir uygulama olarak Şekil 5.12.'de görülen denetim yapısı verilmiştir. Bu uygulamada, Kararsız ölü zamanlı bir sistem için denetleyici tasarımı gerçekleştirilmiş ardından sistem cevabı ve sistem için optimize edilmiş denetleyici ile sistem cevabı verilmiştir. Bu denetleyici, kararsız olan bu sistemi kararlı hale getirmiş, aynı zamanda kararlı durum hatasını sıfır yapmıştır.



Şekil 5.12. Denetleyici tasarım uygulaması

BÖLÜM 6. SONUÇLAR

Bu çalışmada sistem modeline ait parametrelerinin bulunması ve denetleyici tasarımı için PSO, FA ve CS algoritmaları kullanılmıştır. Parametre optimizasyonu yönteminde sistem modeline uygun parametreler, seçilen test düzeneğine, amaç fonksiyonuna ve algoritmaya göre elde edilmiştir. Denetleyici tasarımında da ölü zamanlı sisteme uygun denetleyici parametreleri, seçilen hata kriterine, algoritmaya ve denetleyici tipine göre elde edilmiştir. Ayrıca yapılan modelleme ve denetleyici tasarımlarında kolay bir kullanım sağlaması için bir arayüz oluşturulmuş ve bu arayüz Bölüm 5’de verilmiştir. Sistem model parametrelerinin belirlenmesinde her bir durum için beşer deneme yapılarak sonuçlar elde edilmiştir. Daha sonra elde edilen bu sonuçlar ile gerçek sistem karşılaştırılmış ve yüzde hatalarına bakılmıştır. Ayrıca diğer algoritmalarla kıyaslama yapılmış ve bu problem için PSO ve FA algoritmalarının daha başarılı sonuçlar ürettiği görülmüştür. Bu sonuçlar Intel(R) Core(TM) i7-6700 HQ CPU @ 2.60 Ghz, 64 Bit, 8GB RAM’e sahip bir bilgisayardan elde edilmiştir. Yapılan çalışma Matlab 2017a programı kullanılarak gerçekleştirilmiştir.

6.1. Analiz Sonuçları

Yapılan çalışmalarda hem sistemin model parametreleri belirlenmiş, hemde denetleyici tasarımı yapılmıştır. Sonuçları ise iki farklı alt başlıkta verilmiştir. Bu çalışmada kullanılan algoritmaların parametreleri, literatürdeki yayınlarda sıklıkla kullanılan parametreler olarak seçilmiş ve bu parametreler Tablo 6.1.’de verilmiştir.

Tablo 6.1. Algoritmaların parametreleri

PSO		CS		FA	
Sürüdeki Parçacık Sayısı	40	Sürüdeki Ateşböceği Sayısı	40	Sürüdeki Ateşböceği Sayısı	40
Maximum İterasyon	100	Maximum İterasyon	100	Maximum İterasyon	100
c_1	2	p_a	0,2	β_0	0,8
c_1	2	-	-	γ	0,2
w	0,9	-	-	α	0,2

6.1.1. Parametre optimizasyon sonuçları

Parametre optimizasyonu için gri kutu olarak ifade edilen, model tipi belli fakat model parametreleri tam olarak bilinmeyen yapı kullanılmış ve belli olmayan parametre değerleri bulunmuştur. Bu amaçla beş farklı sistem için PSO, CS ve FA algoritmaları kullanılarak model parametreleri belirlenmeye çalışılmıştır. Elde edilen sonuçlar ile parametrelerin ortalama yüzde hatası (% KH), amaç fonksiyonunun ortalama değeri (ORT) ve amaç fonksiyon değerlerinin standart sapma (STD) değerleri hesaplanmıştır. Her adım için beş uygulama yapılarak bu veriler elde edilmiştir.

Örnek Uygulama 1:

İlk uygulama olarak yapısı Eşitlik (5.1)'de verilen birinci dereceden ölü zamanlı bir transfer fonksiyonu ele alınmıştır. Öncelikle sistem basamak cevabı, tek ve çift röle ile test edilmiş ve ardından elde edilen veriler amaç fonksiyonuna göre PSO, CS ve FA algoritmaları ile optimize edilerek, uygun parametre değerleri bulunmuştur. $G_1(s)$ sistemi için Basamak cevabı, Tek röle ve Çift röle cevaplarına göre elde edilen sonuçlar, sırası ile Tablo 6.2., Tablo 6.3. ve Tablo 6.4.'de verilmiştir. Elde edilen sonuçlara göre, Basamak cevabı yönteminde PSO algoritması daha başarılı iken, tek röle ve çift röle yöntemlerinde FA algoritması daha başarılı sonuçlar üretmektedir.

$$G_1(s) = \frac{1}{s+1} e^{-s} \quad (5.1)$$

Tablo 6.2. $G_1(s)$ için gerçek sistem ile algoritmaların basamak cevabına göre bulduğu sonuçların benzerlikleri

	IAE			ISE			ITAE		
	%KH	ORT	STD	%KH	ORT	STD	%KH	ORT	STD
PSO	2,67e-7	5,84e-4	1,39e-4	8,31e-4	6,75e-4	1,98e-4	1,24e-7	6,96e-4	2,22e-4
CS	0,011	16,56	4,82	9,99e-3	0,07	0,03	1,18e-2	89,55	88,08
FA	3e-3	2,02	1,23	6,60e-3	2,56e-3	0,52	1,22e-1	111,87	82,93

Tablo 6.3. $G_1(s)$ için gerçek sistem ile algoritmaların tek röle cevabına göre bulduğu sonuçların benzerlikleri

	IAE			ISE			ITAE		
	%KH	ORT	STD	%KH	ORT	STD	%KH	ORT	STD
PSO	1,11	428,07	445,35	0,61	87,99	169,18	1,41	9225,92	6914,11
CS	0,087	125,49	113,06	0,092	4,53	3,35	0,120	3776,49	4158,18
FA	0,018	104,87	50,46	0,0231	7,855e-1	3,22e-1	0,024	9,72	5,80

Tablo 6.4. $G_1(s)$ için gerçek sistem ile algoritmaların çift röle cevabına göre bulduğu sonuçların benzerlikleri

	IAE			ISE			ITAE		
	%KH	ORT	STD	%KH	ORT	STD	%KH	ORT	STD
PSO	0,11	60,08	75,18	0,51	94,22	178,61	0,11	3739,44	5993,74
CS	0,16	49,55	8,98e-1	7,4e-2	4,06	3,02	0,12	3854,44	4012,74
FA	0,020	11,76	5,48	2,9e-2	0,44	0,44666	0,041	426,38	21,65

Ayrıca FOPDT sistemi için bulunan parametreler ile literatürde yapılmış olan bazı çalışmaların sonuçları Tablo 6.5.'de verilmiştir. Optimizasyonla elde edilen en iyi sonuçlar literatürdeki sonuçlardan daha iyi iken en kötü sonucu kötü olduğu görülmektedir. Lakin iterasyon sayısı ve birey sayısı arttıkça kestirim hatası azalması beklenmektedir.

Tablo 6.5. FOPDT Sisteminin en iyi ve en kötü parametreleri ve literatürdeki sonuçlar

	K	τ	L
En iyi Optimizasyon (PSO-Basamak Yanıtı-IAE, ITAE kriterleri)	1	1	1
En Kötü Optimizasyon (PSO-Tek Röle-ITAE Kriteri)	2,0271	7,48478	0,9009
Mamat ve Flemming [79]	0,99	0,99	1,04
Kealy ve O'Dwyer [25]	0,9999	1,0964	1,0415

Örnek Uygulama 2:

İkinci uygulama olarak Eşitlik (5.2)'de verilen ikinci dereceden ölü zamanlı bir sistemin transfer fonksiyonu ele alınmıştır. Öncelikle sistem basamak cevabı, tek ve çift röle ile test edilmiş ve ardından elde edilen veriler amaç fonksiyonlarına göre PSO, CS ve FA algoritmaları ile optimize edilerek, uygun parametre değerleri bulunmuştur. $G_2(s)$ sistemi için elde edilen sonuçlar sırası ile Tablo 6.6., Tablo 6.7. ve Tablo 6.8.'da verilmiştir. Basamak cevabında FA daha başarılı sonuçlar gösterirken tek röle testinde farklı amaç fonksiyonları için FA ve PSO başarılı sonuçlar elde edilmiştir. Çift röle testinde PSO daha başarılı sonuçlar elde etmiştir.

$$G_2(s) = \frac{1}{(s+1)^2} e^{-2s} \quad (5.2)$$

Tablo 6.6. $G_2(s)$ için gerçek sistem ile algoritmaların basamak cevabına göre bulunduğu sonuçların benzerlikleri

	IAE			ISE			ITAE		
	%KH	ORT	STD	%KH	ORT	STD	%KH	ORT	STD
PSO	0,043	16,894	29,24	0,045	0,720	0,968	0,0079	72,578	33,751
CS	0,065	95,535	118,25	0,033	0,213	0,368	0,0291	122,655	1,055
FA	0,006	6,413	0,422	0,008	0,036	0,033	0,0072	11,204	8,968

Tablo 6.7. $G_2(s)$ için gerçek sistem ile algoritmaların tek röle cevabına göre bulunduğu sonuçların benzerlikleri

	IAE			ISE			ITAE		
	%KH	ORT	STD	%KH	ORT	STD	%KH	ORT	STD
PSO	0,0886	110,20	103,67	0,0898	3,2267	3,2988	0,099	660,304	90,679
CS	0,0233	45,712	29,123	0,3207	22,316	16,520	0,033	452,499	59,314
FA	0,0008	0,1018	0,084	0,1402	0,8647	1,4960	0,007	3,964	3,0562

Tablo 6.8. $G_2(s)$ için gerçek sistem ile algoritmaların çift röle cevabına göre bulunduğu sonuçların benzerlikleri

	IAE			ISE			ITAE		
	%KH	ORT	STD	%KH	ORT	STD	%KH	ORT	STD
PSO	0,0008	0,0742	0,0295	0,0055	0,0007	0,0001	0,0086	4,351726	2,6236
CS	0,0115	23,691	22,028	0,0418	1,4422	1,9203	0,1320	876,8889	687,73
FA	0,0412	56,519	34,921	0,15837	7,4298	5,7275	0,1133	742,6756	759,45

Örnek Uygulama 3:

Bu uygulamada da Eşitlik (5.3)'de verilen üçüncü dereceden ölü zamanlı bir sistemin transfer fonksiyonu ele alınmıştır. Öncelikle sistem basamak cevabı, tek ve çift röle ile test edilmiş ve ardından elde edilen veriler amaç fonksiyonuna göre PSO, CS ve FA algoritmaları ile optimize edilerek, uygun parametre değerleri bulunmuştur. $G_3(s)$

sistemi için Basamak cevabı, Tek röle ve Çift röle cevaplarına göre elde edilen sonuçlar, sırası ile Tablo 6.9., Tablo 6.10. ve Tablo 6.11.'de verilmiştir. Elde edilen sonuçlara göre, Basamak cevabı yönteminde CS ve FA farklı performans kriterlerine göre daha başarılı olduğu görülmektedir. PSO algoritması daha başarılı iken, tek röle ve çift röle yöntemlerinde FA algoritması daha başarılı sonuçlar üretmektedir.

$$G_3(s) = \frac{2}{(s+1)(s+2)(s+3)} e^{-0.5s} \quad (5.3)$$

Tablo 6.9. $G_3(s)$ için gerçek sistem ile algoritmaların basamak cevabına göre bulunduğu sonuçların benzerlikleri

	IAE			ISE			ITAE		
	%KH	ORT	STD	%KH	ORT	STD	%KH	ORT	STD
PSO	0.3103	1,66261	1,50519	0.2607	0,0919	0,01432	0.3635	26,48824	35,9295
CS	0,1042	21,897	19,263	0,099004	0,0400	0,0077	0,09813	55,90836	13,408
FA	0,0904	9,1319	0,0202	0,134834	0,6043	0,8278	0,28317	566,4273	631,7873

Tablo 6.10. $G_3(s)$ için gerçek sistem ile algoritmaların tek röle cevabına göre bulunduğu sonuçların Benzerlikleri

	IAE			ISE			ITAE		
	%KH	ORT	STD	%KH	ORT	STD	%KH	ORT	STD
PSO	0.2266	5,35317	4,84913	0.1672	0,02592	0,03106	0.2934	1324,67389	2142,3
CS	0,1719	156,87	105,81	0,157732	0,3963	0,1239	0,12706029	886,688071	1137,9
FA	0,1438	123,51	104,69	0,118112	0,2386	0,1134	0,13371816	955,142222	707,80

Tablo 6.11. $G_3(s)$ için gerçek sistem ile algoritmaların çift röle cevabına göre bulunduğu sonuçların benzerlikleri

	IAE			ISE			ITAE		
	%KH	ORT	STD	%KH	ORT	STD	%KH	ORT	STD
PSO	0,3764	28,236	27,8462	0,137405	0,0207	0,01116	0,33587020	129,581872	38,7084
CS	0,1042	86,032	4,87729	0,122973	0,3763	0,0207	0,13401669	676,157326	425,789
FA	0,187	90,210	14,833	0,1064	2,0968	2,03807	0,2427	1019,3128	394,7

Örnek Uygulama 4:

Bu uygulamada olarak yapısı Eşitlik (5.4)'de verilen integratör içeren birinci dereceden ölü zamanlı bir transfer fonksiyonu ele alınmıştır. Bu sistemler genellikle kimyasal süreçlere örnek olarak verilebilmektedir. Öncelikle sistem basamak cevabı, tek ve çift röle ile test edilmiş ve ardından elde edilen veriler amaç fonksiyonlarına göre PSO, CS ve FA algoritmaları ile optimize edilerek, uygun parametre değerleri bulunmuştur. $G_4(s)$ sistemi için Basamak cevabı, Tek röle ve Çift röle cevaplarına göre elde edilen sonuçlar, sırası ile Tablo 6.12., Tablo 6.13. ve Tablo 6.14.'de verilmiştir. Elde edilen sonuçlara göre, Basamak cevabı yönteminde CS algoritması FA ve PSO'ya göre daha iyi sonuçlar elde edilmiştir. Tek röle testinde PSO ve CS farklı amaç

fonksiyonları için başarı gösterirken çift röleli testinde farklı amaç fonksiyonlarına göre tüm algoritmaların performans değerleri farklılık göstermiştir. Daha kesin yargıya varılabilmesi için iterasyon sayısı ve sürüdeki birey sayıları arttırılması gerektiğini göstermektedir.

$$G_4(s) = \frac{1}{s(s+1)} e^{-0.5s} \quad (5.4)$$

Tablo 6.12. $G_4(s)$ için gerçek sistem ile algoritmaların basamak cevabına göre bulduğu sonuçların benzerlikleri

	IAE			ISE			ITAE		
	%KH	ORT	STD	%KH	ORT	STD	%KH	ORT	STD
PSO	1,0021	3632,8	3252,4	0,873783	1245,8	1621,9	0,33100	26675,8703	1791,6
CS	0,0525	867,18	1123,2	0,009985	0,6723	0,9153	0,01120	1644,69891	1285,9
FA	0,2146	1977,7	2724,6	0,403941	900,86	1058,9	0,05756	8402,09326	166,99

Tablo 6.13. $G_4(s)$ için gerçek sistem ile algoritmaların tek röle cevabına göre bulduğu sonuçların benzerlikleri

	IAE			ISE			ITAE		
	%KH	ORT	STD	%KH	ORT	STD	%KH	ORT	STD
PSO	3,46e-5	0,0059	0,0052	0,001894	0,0070	0,0088	0,01739	255,202789	271,25
CS	0,0136	257,53	100,33	0,020057	11,875	11,070	0,00865	1357,07338	669,38
FA	0,0034	69,401	3,5556	0,506450	1638,5	2316,4	0,47497	38459,5761	53577,2

Tablo 6.14. $G_4(s)$ için gerçek sistem ile algoritmaların çift röle cevabına göre bulduğu sonuçların benzerlikleri

	IAE			ISE			ITAE		
	%KH	ORT	STD	%KH	ORT	STD	%KH	ORT	STD
PSO	1,1822	8219,6	0,2703	0,00766	0,2836	0,3995	1,05658	93239,94	1,8432
CS	0,1317	4154,4	3524,9	0,02195	25,804	28,496	0,01167	3001,371	1366,4
FA	0,0008	1,0844	0,0014	0,00116	0,0742	0,0424	0,58533	4923,594	6227,0

Örnek Uygulama 5:

Son uygulama olarak genellikle kimyasal süreçlerde kullanılan Eşitlik (5.5)'de verilen non-minimum fazlı birinci dereceden ölü zamanlı sistemin transfer fonksiyonu ele alınmıştır. Öncelikle sistem basamak cevabı, tek ve çift röle ile test edilmiş ve ardından elde edilen veriler amaç fonksiyonuna göre PSO, CS ve FA algoritmaları ile optimize edilerek, uygun parametre değerleri bulunmuştur. $G_5(s)$ sistemi için Basamak cevabı, Tek röle ve Çift röle cevaplarına göre elde edilen sonuçlar, sırası ile Tablo 6.15., Tablo 6.16. ve Tablo 6.17.'de verilmiştir. Elde edilen sonuçlara göre, Basamak ve çift röle

testinde PSO daha iyi sonuç verirken tek röleli testine göre CS ve FA daha başarılı sonuçlar vermiştir.

$$G_5(s) = \frac{-0.2s + 1}{s + 1} e^{-0.2s} \quad (5.5)$$

Tablo 6.15. $G_5(s)$ için gerçek sistem ile algoritmaların basamak cevabına göre bulduğu sonuçların benzerlikleri

	IAE			ISE			ITAE		
	%KH	ORT	STD	%KH	ORT	STD	%KH	ORT	STD
PSO	0,0005	0,1004	0,1407	0,005506	0,0394	0,0550	0,00209825	0,04053959	0,0562
CS	0,0141	12,206	7,5815	0,010575	0,2687	0,1403	0,01275215	25,8865297	30,842
FA	0,4231	249,70	322,76	0,251968	20,799	28,626	0,46161765	938,438619	637,59

Tablo 6.16. $G_5(s)$ için gerçek sistem ile algoritmaların tek röle cevabına göre bulduğu sonuçların benzerlikleri

	IAE			ISE			ITAE		
	%KH	ORT	STD	%KH	ORT	STD	%KH	ORT	STD
PSO	2,0807	68,163	3,4791	0,732	38,937	44,293	1,975	597,906	96,765
CS	0,8497	522,00	183,61	0,716	73,252	4,7321	1,580	13731,21	11141,8
FA	0,905	967,84	34,289	1,663	200,86	171,70	0,617	10415,69	7535,77

Tablo 6.17. $G_5(s)$ için gerçek sistem ile algoritmaların çift röle cevabına göre bulduğu sonuçların benzerlikleri

	IAE			ISE			ITAE		
	%KH	ORT	STD	%KH	ORT	STD	%KH	ORT	STD
PSO	0,5821	41,431	12,879	1,176902	4,5973	4,9839	1,04123383	8874,545	5581,030
CS	0,6049	1638,7	1922,4	1,991400	424,30	115,56	1,75189637	18347,33	14157,28
FA	1,0676	2578,2	848,37	1,976904	111,58	56,542	1,09815657	2510,166	2903,613

6.1.2. Denetleyici tasarım sonuçları

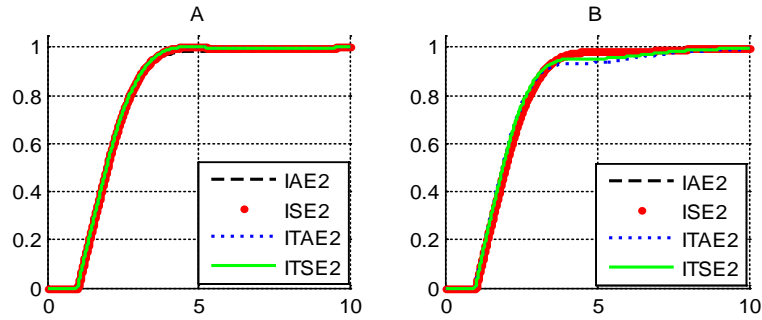
Bu bölümde yukarıda modellenmesi yapılan sistemler için denetleyici tasarımı yapılmıştır. Bu amaçla seçilen algoritma ve seçilen denetleyici tipine göre her bir sistem için denetleyici katsayıları bulunmuştur. Yine elde edilen sonuçlar karşılaştırılmıştır.

Yukarıda modellenmesi yapılan $G_1(s)$ sistemi için, PI denetleyici tasarım sonuçlarının analizleri Tablo 6.18.'de verilirken, PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.1.'de, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.2.'de amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.3.'de denetlenmiş sistem cevapları verilmiştir. PI denetleyici tasarım sonuçlarında

çok farklılık görülmemesine rağmen, lokal arama yapan PSO algoritmasının global arama yapan algoritmalara (CS, FA) göre nisbeten daha iyi sonuç verdiği görülmüştür.

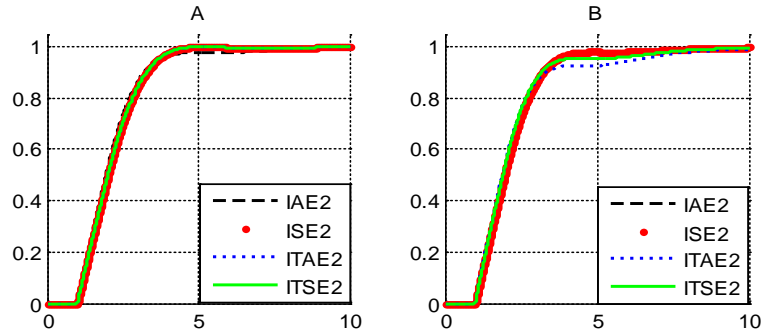
Tablo 6.18. G₁ Sisteminde PI için PSO CS ve FA sonuçları

	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	4059,0	4059,0	4,2e-6	4059,33	4059,54	0,346	4059,15	4059,17	0,02129
ISE2	4256,9	4256,96	4,98e-5	4257,78	4262,00	5,234	4256,97	4257,16	0,209
ITAE2	3875,6	3875,63	1e-4	3875,78	3875,98	0,317	3875,75	3875,81	0,05484
ITSE2	3827,0	3827,02	8,5e-6	3827,13	3827,74	0,943	3827,07	3827,11	0,0336



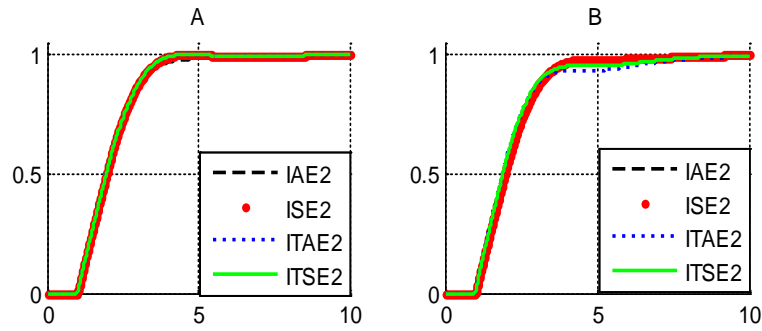
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.1. PSO için PI denetleyici sonuçları



a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.2. Cuckoo Search için PI denetleyici sonuçları



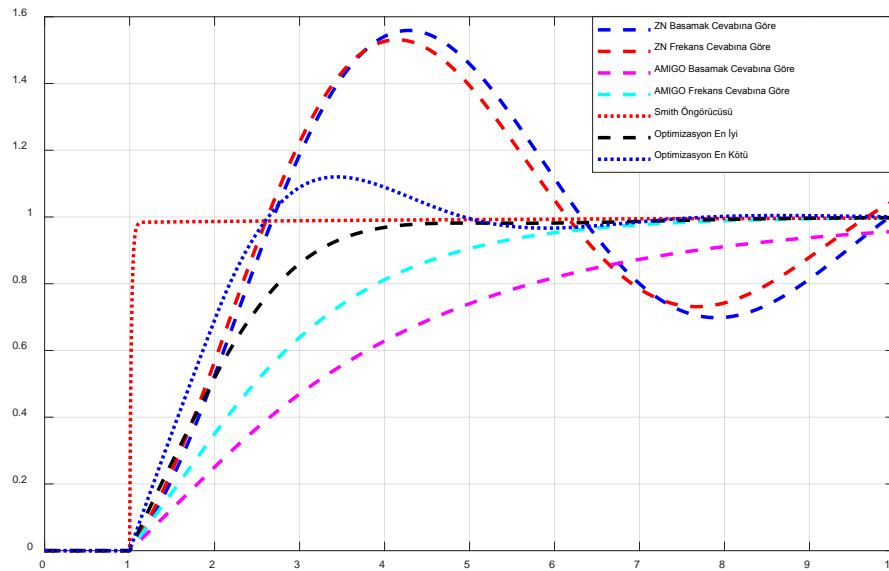
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.3. Ateş Böceği Algoritması için PI denetleyici sonuçları

$G_1(s)$ sistemi için tasarlanan PI denetleyici Zeigler Nichols, AMIGO veya Smith gibi farklı teknikler ile denetlendiğinde kullanılması gereken denetleyici katsayıları Tablo 6.19.'da verilmiştir. Elde edilen denetleyici katsayılarına göre sistem cevapları Şekil 6.4.'de verilmiştir. Elde edilen sonuçlara göre PI için en iyi yapının Smith öngörücüsü olduğu görülmüştür. Lakin bu özel yapının aksine diğer PI denetleyicilerle kıyaslandığında, optimizasyonla elde edilen en iyi ve en kötü denetleyici katsayılarının Zeigler Nichols ve AMIGO tekniklerinden daha iyi sonuçlar verdiği görülmektedir.

Tablo 6.19. $G_1(s)$ Sistemi için farklı PI denetleyici katsayıları

	K_p	K_i
Zeigler Nichols Basamak Cevabına Göre	0,30219	0,90657
Zeigler Nichols Frekans Cevabına Göre	0,36515	0,912875
AMIGO Basamak Cevabına Göre	0,25036	0,25036
AMIGO Frekans Cevabına Göre	0,34999	0,34999
Smith Öngörücüsü	50,521	9,09378
Optimizasyona Göre En İyi	0,55106	0,46023
Optimizasyona Göre En Kötü	0,7336557	0,617515041



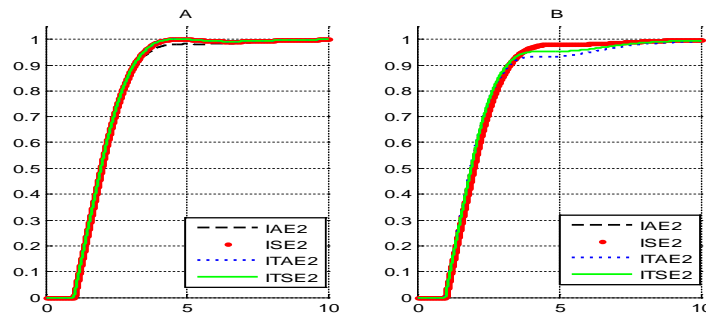
Şekil 6.4. $G_1(s)$ Sistemi için farklı PI denetleyici sonuçları

$G_1(s)$ sistemi için PID denetleyici tasarım sonuçlarının analizleri Tablo 6.20.'de verilirken, denetlenmiş sistem cevapları olarak PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.5.'de, CS'nin amaç fonksiyon değerine göre

en iyi ve en kötü sonuçları Şekil 6.6.'da, FA'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.7.'de verilmiştir. PID denetleyici tasarım sonuçlarında çok farklılık görülmemesine rağmen, lokal arama (PSO) yapan algoritmanın, global arama yapan algoritmalara (CS, FA) göre nisbeten daha iyi sonuç verdiği görülmüştür.

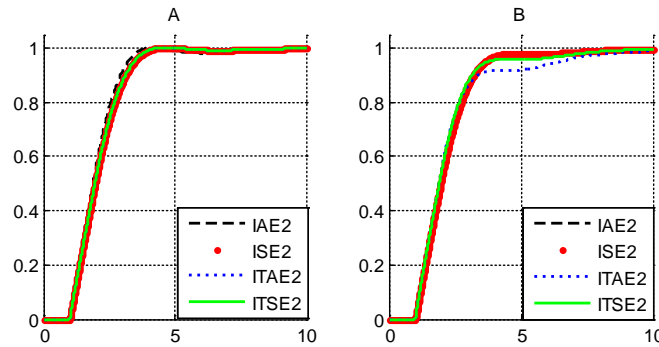
Tablo 6.20. G_1 Sisteminde PID için PSO CS ve FA sonuçları

	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	4059,15	4059,17	0,0212	4059,669	4085,213	43,55	4059,158	4059,175	0,0212
ISE2	4256,97	4257,16	0,2098	4257,134	4257,605	0,573	4256,979	4257,163	0,2098
ITAE2	3875,75	3875,81	0,0548	3876,806	3884,875	7,287	3875,75	3875,813	0,0548
ITSE2	3827,07	3827,11	0,0336	3829,018	3856,223	41,620	3827,074	3827,11	0,03368



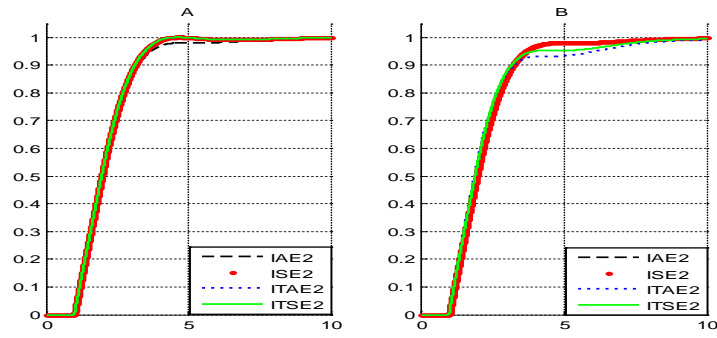
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.5. PSO için PID denetleyici sonuçları



a) En İyi Amaç Fonksiyon Değerine Göre b) En Kötü Amaç Fonksiyon Değerine Göre

Şekil 6.6. CS için PID denetleyici sonuçları



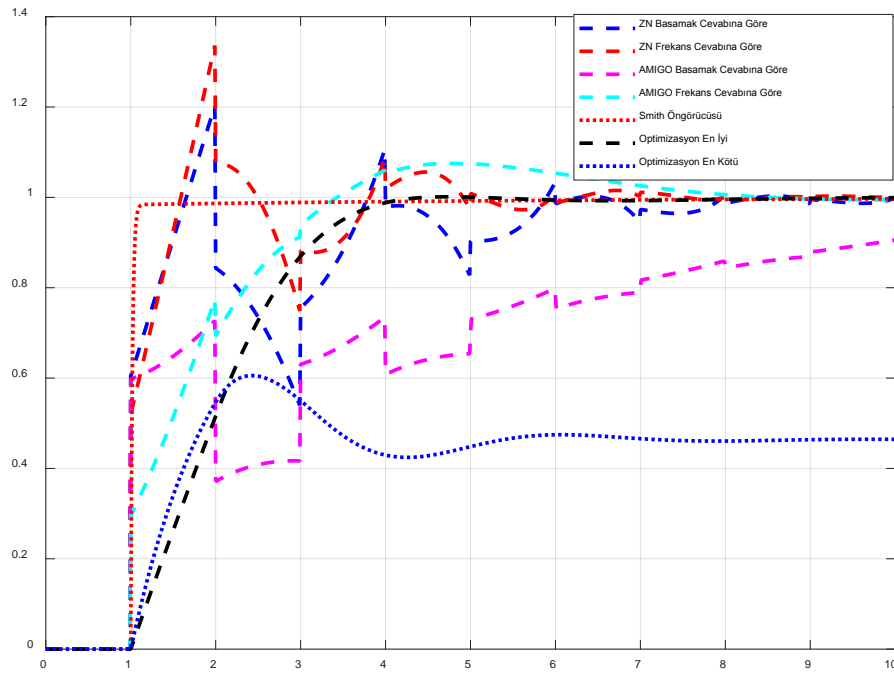
a) En iyi amaç fonksiyon değerine göre b) En Kötü amaç fonksiyon değerine göre

Şekil 6.7. FA için PID denetleyici sonuçları

$G_1(s)$ sistemi için tasarlanan PID denetleyici Zeigler Nichols, AMIGO veya Smith gibi farklı teknikler ile denetlendiğinde kullanılması gereken denetleyici katsayıları Tablo 6.21.'de verilmiştir. Elde edilen denetleyici katsayılarına göre sistem cevapları Şekil 6.8.'de verilmiştir. Elde edilen sonuçlara göre PID için en iyi yapının Smith öngörücüsü olduğu görülmüştür. Lakin bu özel yapının aksine diğer PID denetleyicilere bakıldığında, optimizasyonla elde edilen en iyi katsayıların Zeigler Nichols ve AMIGO tekniklerinden çok daha iyi sonuç verdiği görülmektedir. Diğer taraftan, türev katsayısından ve sistemin zaman sabitinin düşük olmasından dolayı, sistem cevabında kabul edilemeyecek ani geçişlerin olduğu görülmektedir.

Tablo 6.21. $G_1(s)$ Sistemi için farklı PID denetleyici katsayıları

	K_p	K_i	K_d
Zeigler Nichols Basamak Cevabına Göre	1,20876	0,60438	0,60438
Zeigler Nichols Frekans Cevabına Göre	1,31454	0,87636	0,5195
AMIGO Basamak Cevabına Göre	0,657855	0,2526	0,59805
AMIGO Frekans Cevabına Göre	0,6664	0,68008	0,2962
Smith Öngörücüsü	50,521	9,09378	0
Optimizasyona Göre En iyi	0,53913	0,47564	0
Optimizasyona Göre En Kötü	0,86453	0	0

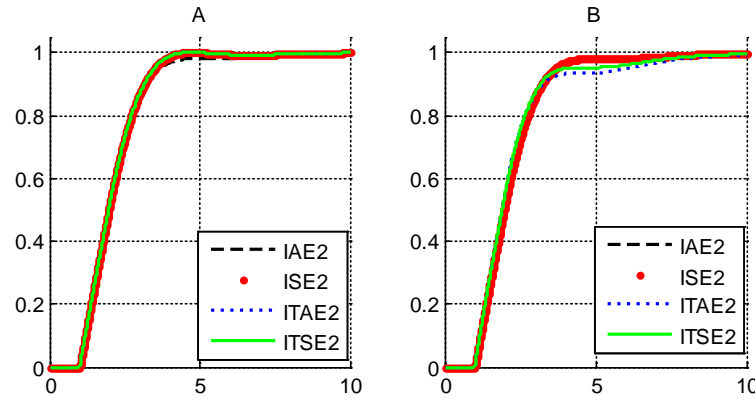


Şekil 6.8. $G_1(s)$ Sistemi için farklı PID denetleyici sonuçları

$G_1(s)$ sistemi için PI-P denetleyici tasarım sonuçlarının analizleri Tablo 6.18.'de verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.9.'da, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.10.'de, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.11.'de verilmiştir. PI-P denetleyici tasarım sonuçlarında çok farklılık görülmemesine rağmen, lokal arama (PSO) yapan algoritmanın, global arama yapan algoritmalara (CS, FA) göre nisbeten daha iyi sonuç verdiği görülmüştür.

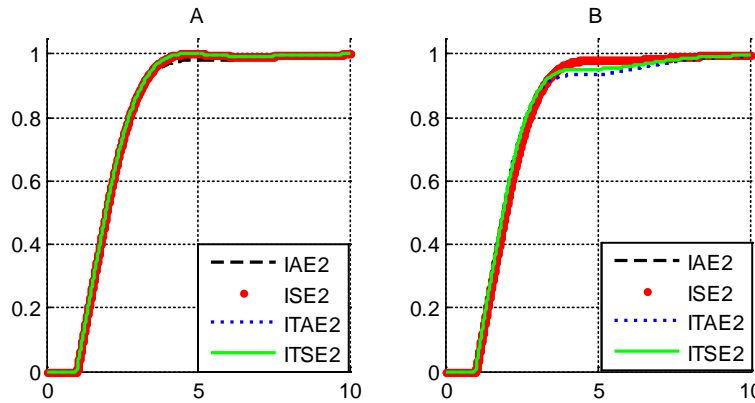
Tablo 6.22. G_1 sisteminde PI-P için PSO CS ve FA sonuçları

	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	4059,07	4059,30	0,40658	4166,749	4480,580	536,67	4059,293	4059,718	0,50655
ISE2	4256,96	4256,96	0,00023	4266,520	4829,620	789,22	4257,254	4258,137	0,77747
ITAE2	3875,63	3875,63	3,85385	4030,542	4473,084	385,08	3875,652	3875,729	0,11837
ITSE2	3827,02	3827,02	6,88308	3829,123	4158,526	565,26	3827,035	3827,692	0,79428



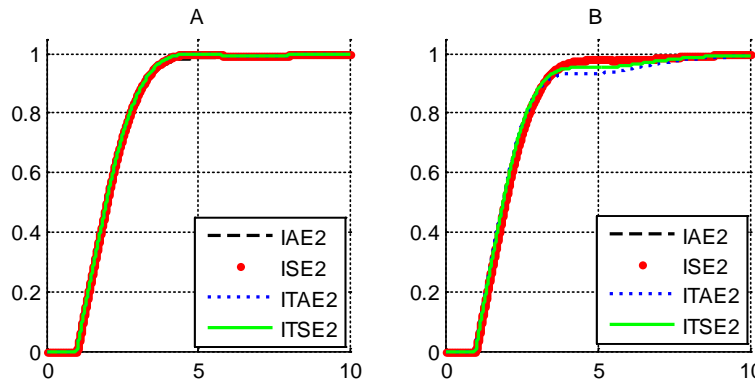
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.9. PSO için PI-P denetleyici sonuçları



a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.10. CS için PI-P denetleyici sonuçları



a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

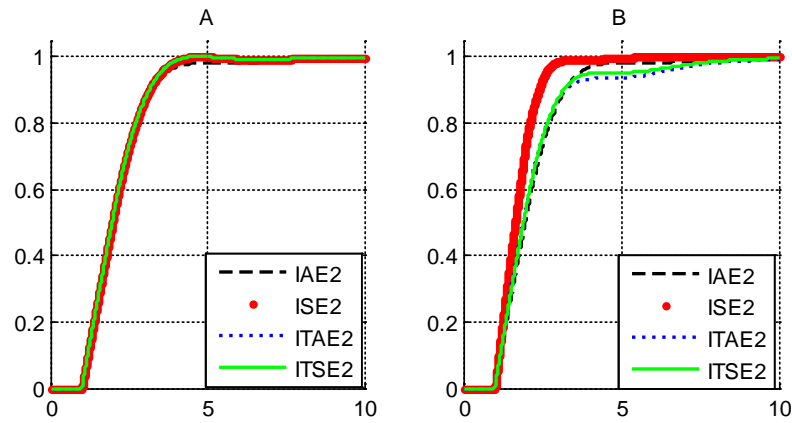
Şekil 6.11. FA için PI-P denetleyici sonuçları

$G_1(s)$ sistemi için PI-PD denetleyici tasarım sonuçlarının analizleri Tablo 6.23.'de verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.12.'de, CS'nin amaç fonksiyon değerine göre en iyi

ve en kötü sonuçları Şekil 6.13.'de, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.14.'de verilmiştir. PI-PD denetleyici tasarım sonuçlarında çok farklılık görülmemesine rağmen, FA algoritması ile genel olarak daha iyi sonuçlar elde edilmiştir.

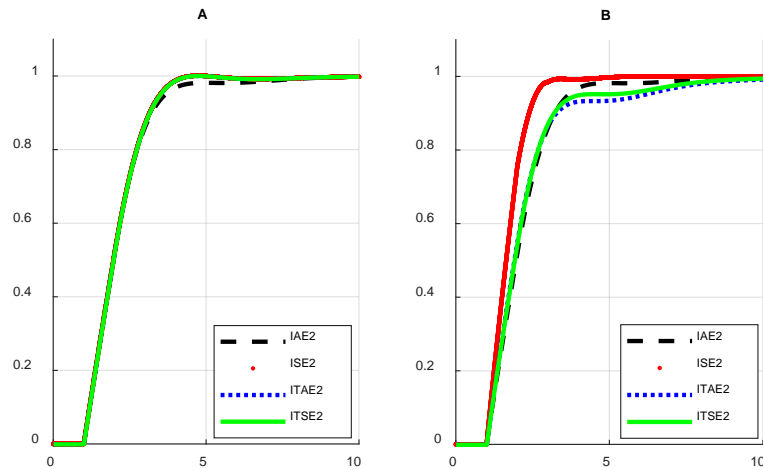
Tablo 6.23. G_1 sisteminde PI-PD için PSO CS ve FA sonuçları

	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	4059,07	4059,07	0,00333	3935,704	132238,5	222173	3910,123	3913,256	3,76254
ISE2	3882,11	4132,01	216,417	4442,587	823160,5	14,1e5	3909,687	3958,953	42,7143
ITAE2	3875,63	3875,63	0,00495	4765,271	7793,383	5244,8	3809,806	3815,791	9,63851
ITSE2	3827,02	3827,02	1e-05	3888,449	24255,69	35277	3702,366	3718,603	14,1423



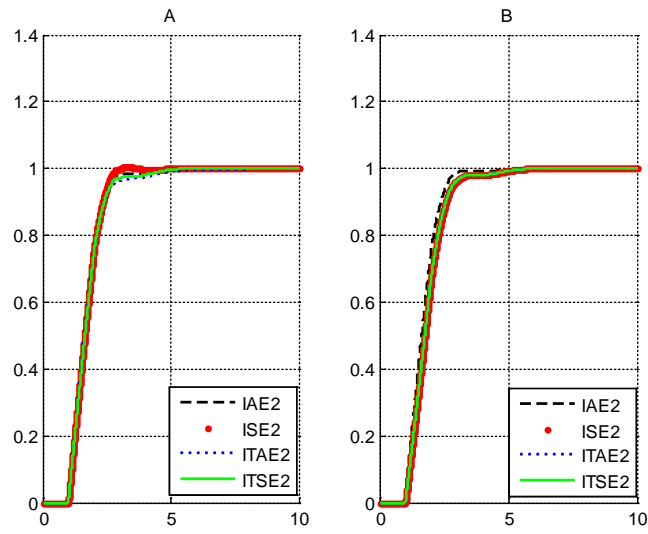
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.12. PSO için PI-PD denetleyici sonuçları



a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.13. CS için PI-PD denetleyici sonuçları



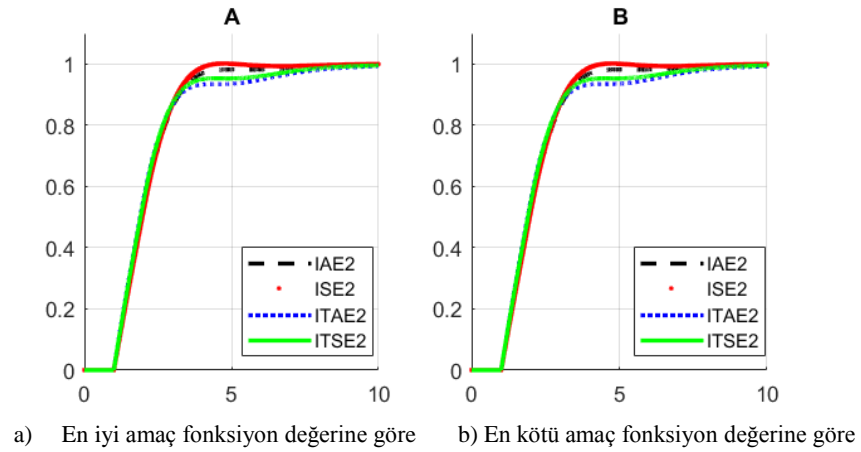
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.14. FA için PI-PD denetleyici sonuçları

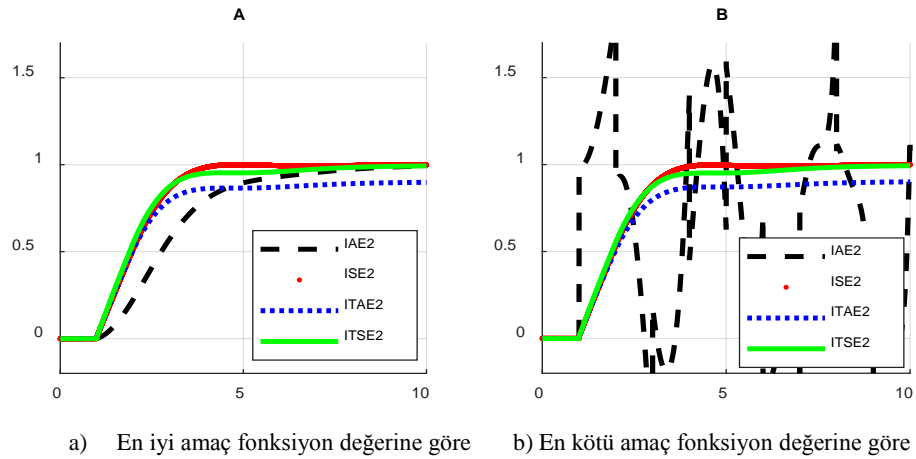
$G_1(s)$ sistemi için PID-PD denetleyici tasarım sonuçlarının analizleri Tablo 6.24.'de verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.15.'de, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.16.'da, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.17.'de verilmiştir. Sonuçlara bakıldığında, en iyi sonucun PSO algoritması tarafından verildiği görülmektedir.

Tablo 6.24. G_1 sisteminde PID-PD için PSO CS ve FA sonuçları

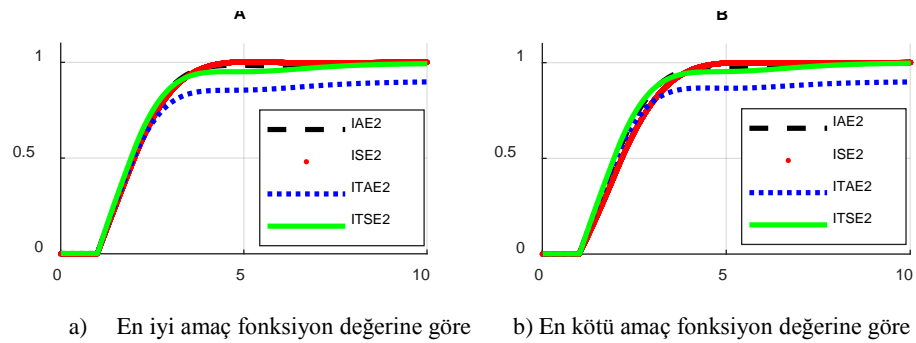
	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	4059,07	4059,07	1,75e-05	4387,518	131030,7	21,5e4	4062,197	4068,920	8,86825
ISE2	4256,96	4256,96	0,00042	4302,391	4304,330	3,0067	4268,556	4312,268	73,5411
ITAE2	3875,63	3875,63	0,00030	3792,257	3792,275	0,0291	3736,192	3737,608	1,43511
ITSE2	3827,02	3827,02	4,02e-05	3878,660	3878,842	0,1673	3825,358	3831,165	7,25161



Şekil 6.15. PSO için PID-PD denetleyici sonuçları



Şekil 6.16. CS için PID-PD denetleyici sonuçları



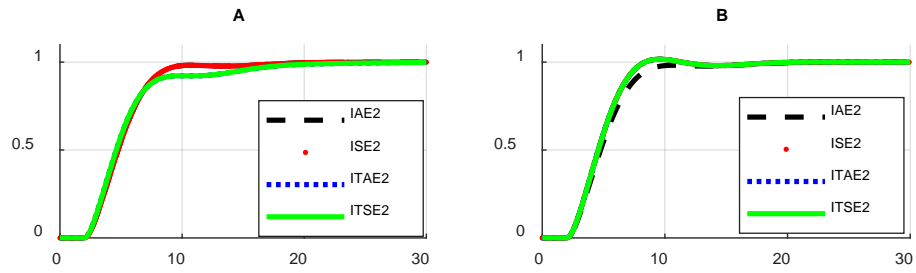
Şekil 6.17. FA için PID-PD denetleyici sonuçları

$G_2(s)$ sistemi için PI denetleyici tasarım sonuçlarının analizleri Tablo 6.25.'de verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.18.'de, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.19.'da, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.20.'de verilmiştir. PI denetleyici tasarım sonuçlarında çok

farklılık görülmemesine rağmen, FA ve PSO algoritmaları ile genel olarak daha iyi sonuçlar elde edilmiştir.

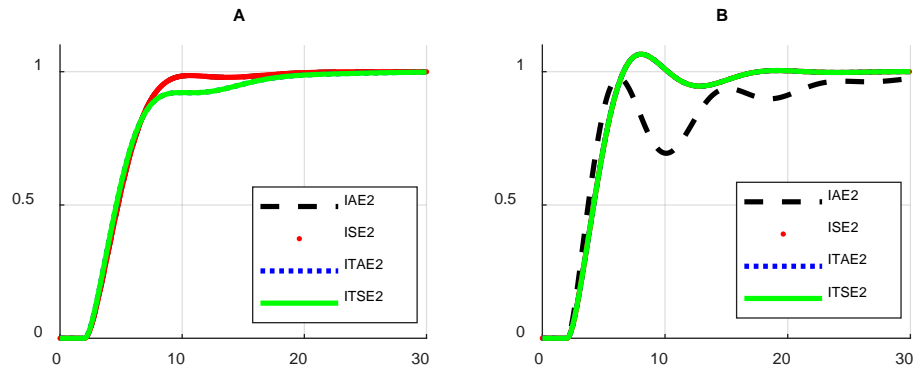
Tablo 6.25. G₂ Sisteminde PI için PSO CS ve FA sonuçları

	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	5097,08	5097,08	1,03-05	5097,733	5952,013	1452,1	5097,092	5097,629	0,89862
ISE2	7919,07	11864,5	6508,00	8315,389	8759,258	385,16	8298,570	8299,588	1,11353
ITAE2	4655,65	4655,65	2,9e-05	4655,655	4656,914	1,0899	4655,694	4655,875	0,18742
ITSE2	4765,67	5692,27	802,458	6157,018	6660,638	864,39	6155,644	6155,677	0,03319



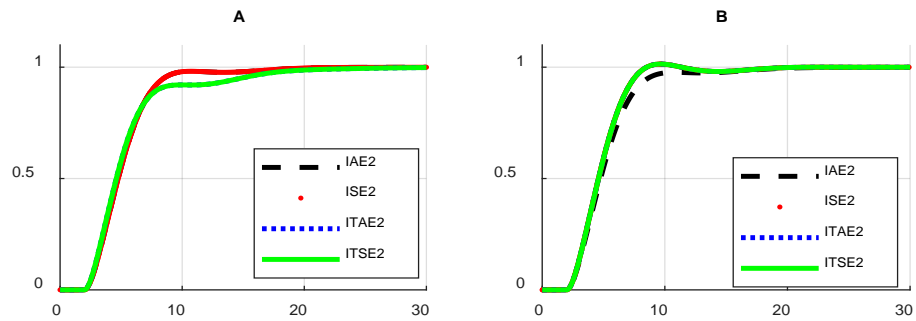
a) En iyi Amaç Fonksiyon Değerine Göre b) En Kötü Amaç Fonksiyon Değerine Göre

Şekil 6.18. PSO için PI Denetleyici Sonuçları



a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.19. CS için PI denetleyici sonuçları



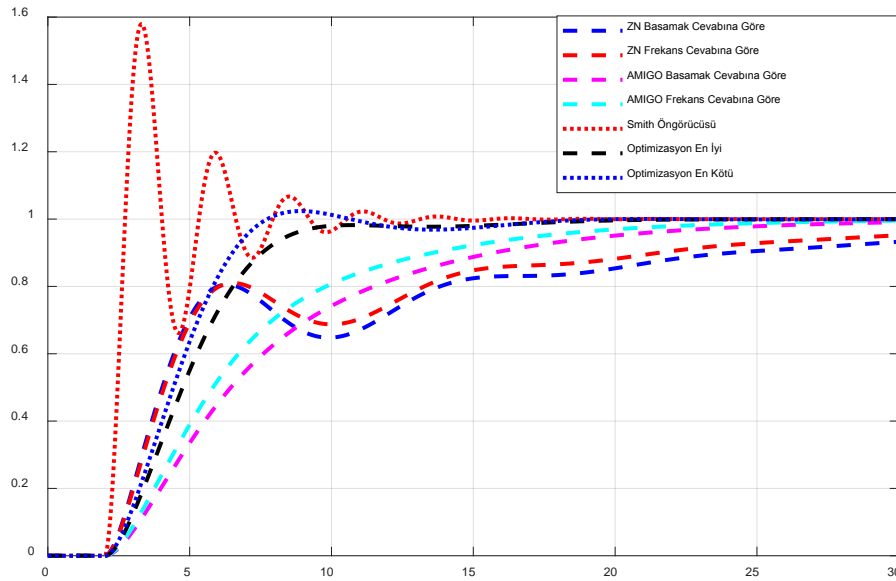
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.20 FA için PI denetleyici sonuçları

$G_2(s)$ sistemi için tasarlanan PI denetleyici Zeigler Nichols, AMIGO veya Smith gibi farklı teknikler ile denetlendiğinde kullanılması gereken denetleyici katsayıları Tablo 6.26.'da verilmiştir. Elde edilen denetleyici katsayılarına göre sistem cevapları Şekil 6.21.'de verilmiştir. Elde edilen sonuçlara göre PI için en iyi yapının Smith öngörücüsü olduğu görülmüştür. Diğer taraftan, bu özel yapının aksine diğer PI denetleyicilere bakıldığında, optimizasyonla elde edilen en iyi ve en kötü denetleyici katsayılarının Zeigler Nichols ve AMIGO tekniklerinden daha iyi sonuçlar verdiği görülmektedir.

Tablo 6.26. $G_2(s)$ sistemi için Farklı PI denetleyici katsayıları

	K_P	K_I
Zeigler Nichols Basamak Cevabına Göre	0,728688	0,10716
Zeigler Nichols Frekans Cevabına Göre	0,691012	0,11914
AMIGO Basamak Cevabına Göre	0,230413	0,12127
AMIGO Frekans Cevabına Göre	0,27532	0,13676
Smith Öngörücüsü	5,96785	7,021
Optimizasyona Göre En İyi	0,3919	0,1925
Optimizasyona Göre En Kötü	0,4658	0,2148



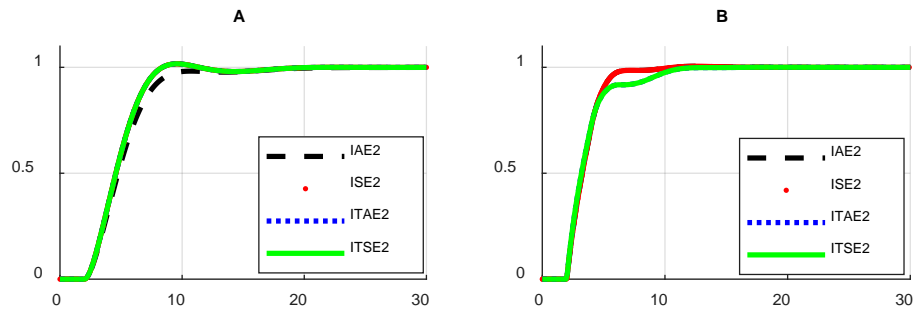
Şekil 6.21. $G_2(s)$ Sistemi için Farklı PI Denetleyici Sonuçları

$G_2(s)$ sistemi için PID denetleyici tasarım sonuçlarının analizleri Tablo 6.27.'de verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.22.'de, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.23.'de, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.24.'de verilmiştir. PID denetleyici tasarım sonuçlarında

performans kriterlerine göre çok fazla farklılık görülmemesine rağmen, FA ve PSO algoritmaları ile genel olarak daha iyi sonuçlar elde edilmiştir.

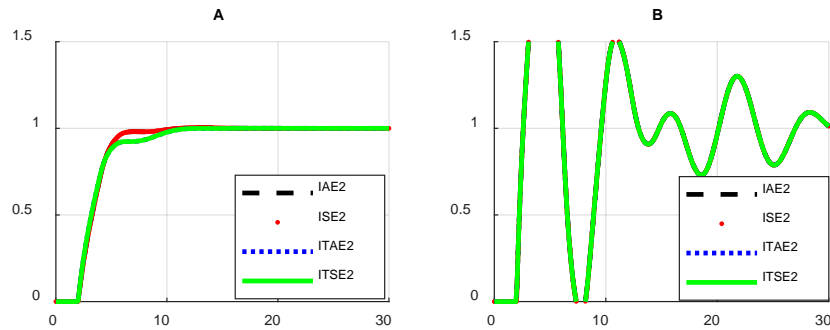
Tablo 6.27. G₂ Sisteminde PID için PSO CS ve FA sonuçları

	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	4538,8	4910,9	322,31	6748,76	37948,2	32106	4540,33	4542,80	2,9737
ISE2	8298,3	8298,3	3915,54	6037,53	7695,71	1469,	5625,53	5638,79	11,518
ITAE2	4281,4	4406,1	216,06	6197,68	12894,5	9907,	4282,38	4283,81	2,0579
ITSE2	4765,6	5692,2	802,45	4799,39	6571,15	2097,	4766,17	4766,77	0,6184



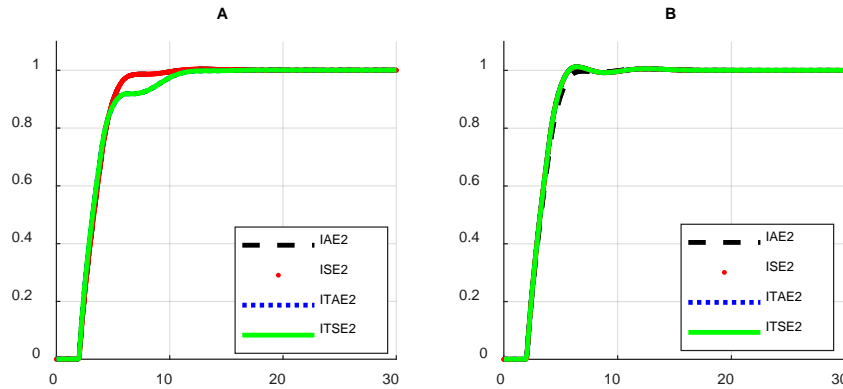
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.22. PSO için PID denetleyici sonuçları



a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.23. CS için PID denetleyici sonuçları



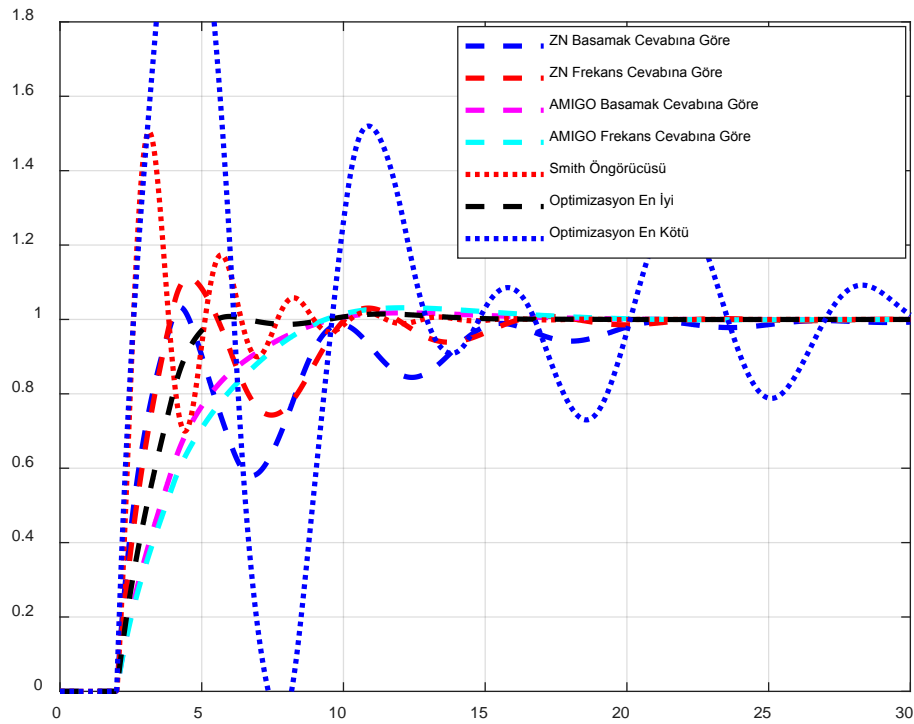
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.24. FA için PID denetleyici sonuçları

$G_2(s)$ sistemi için tasarlanan PID denetleyici Zeigler Nichols, AMIGO veya Smith gibi farklı teknikler ile denetlendiğinde kullanılması gereken denetleyici katsayıları Tablo 6.28.'de verilmiştir. Elde edilen denetleyici katsayılarına göre sistem cevaplarına göre sistem cevapları Şekil 6.25.'de verilmiştir. Elde edilen sonuçlara göre PID için en iyi sonucu optimizasyonla elde edilen en iyi değeri ve yine en kötü sonucu optimizasyonun en kötü denetleyici katsayıları ile elde edilmiştir.

Tablo 6.28. $G_2(s)$ Sistemi için Farklı PID Denetleyici Katsayıları

	K_P	K_i	K_d
Zeigler Nichols Basamak Cevabına Göre	0,985918	0,21433	1,1338
Zeigler Nichols Frekans Cevabına Göre	1,0579	0,28593	0,92641
AMIGO Basamak Cevabına Göre	0,56207	0,25549	0,5007
AMIGO Frekans Cevabına Göre	0,4936	0,2468	0,4837
Smith Öngörücüsü	7,086352	14,042	1,1008
Optimizasyona Göre En İyi	0,7651	0,3163	0,6563
Optimizasyona Göre En Kötü	1,4119	1,4503	2,3156

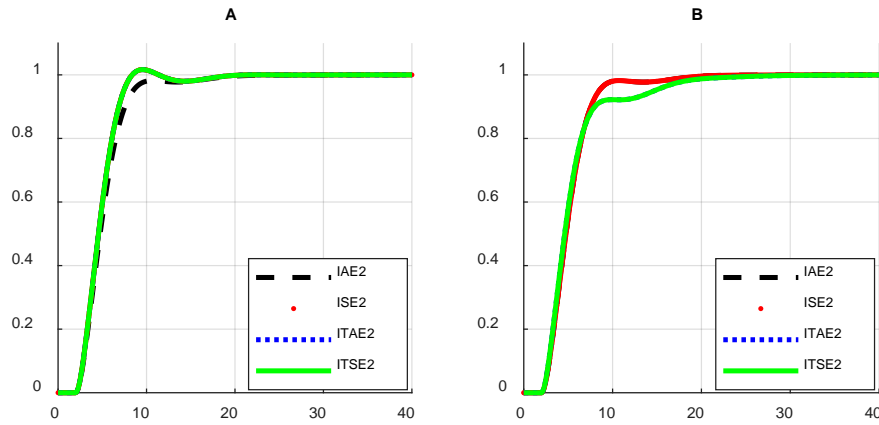


Şekil 6.25. $G_2(s)$ Sistemi için Farklı PID Denetleyici Sonuçları

$G_2(s)$ sistemi için PI-P denetleyici tasarım sonuçlarının analizleri Tablo 6.29.'da verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.26.'da, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.27.'de, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.28.'de verilmiştir. PI-P denetleyici tasarım sonuçlarında performans kriterlerine göre farklılık görülmemesine rağmen, FA ve PSO algoritmaları ile genel olarak daha iyi sonuçlar elde edilmiştir.

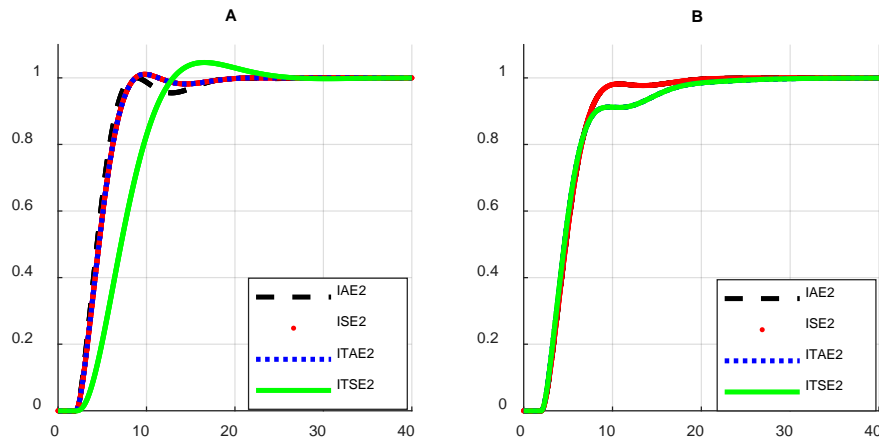
Tablo 6.29. G_2 Sisteminde PI-P için PSO CS ve FA sonuçları

	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	5097,08	5097,08	2,7e-05	5097,092	5148,587	89,178	5097,130	5099,585	2,7632
ISE2	8298,35	8298,35	0,00011	8299,230	8306,115	6,9538	8298,614	8301,807	4,5861
ITAE2	4655,65	4655,65	5,9e-05	4657,058	4661,245	3,7373	4655,891	4687,893	52,299
ITSE2	6155,57	6155,57	8,77537	6178,300	7601,907	2161,9	6156,012	6156,475	0,5496



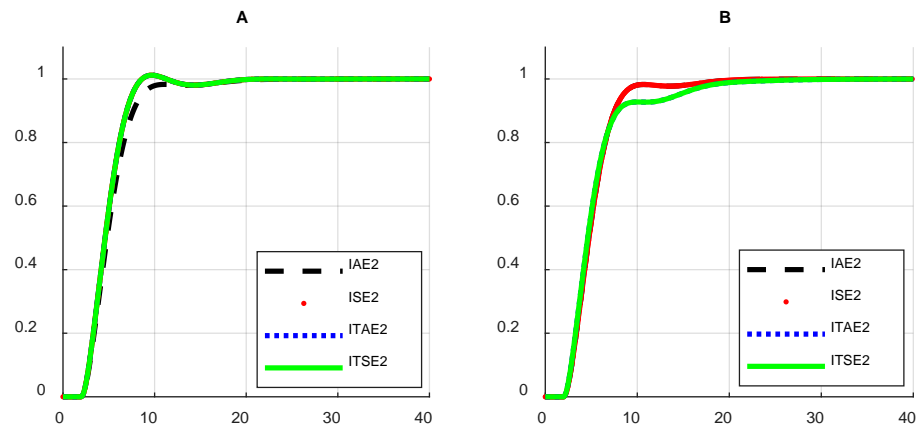
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.26. PSO için PI-P denetleyici sonuçları



a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.27. CS için PI-P denetleyici sonuçları



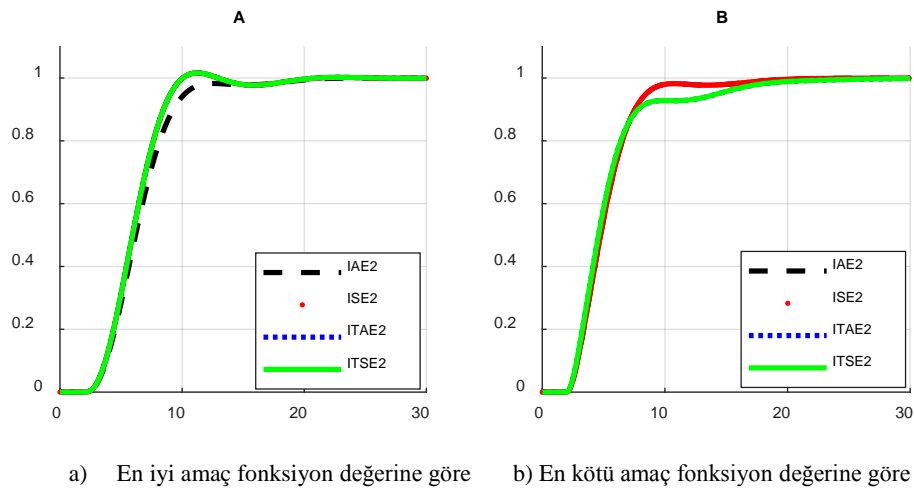
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.28. FA için PI-P denetleyici sonuçları

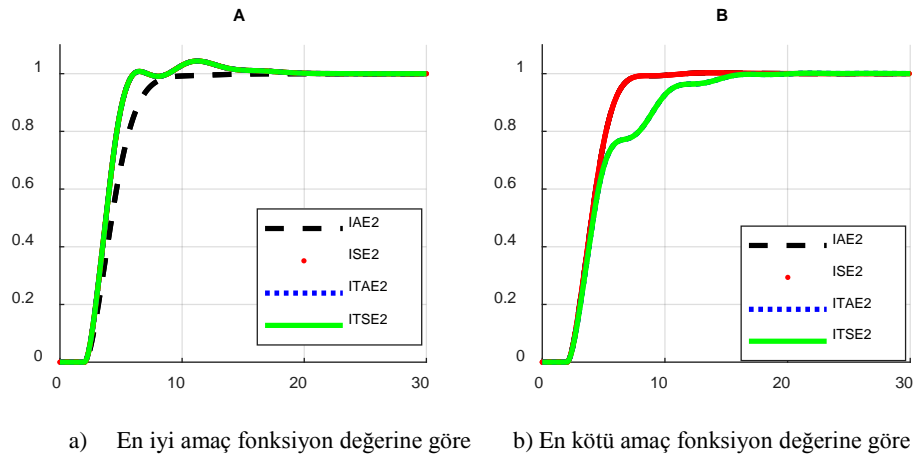
$G_2(s)$ sistemi için PI-PD denetleyici tasarım sonuçlarının analizleri Tablo 6.30.'da verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.29.'da, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.30.'da, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.31.'de verilmiştir. PI-PD denetleyici tasarım sonuçlarında performans kriterlerine göre farklılık görülmemesine rağmen, FA algoritması ile genel olarak daha iyi sonuçlar elde edilmiştir.

Tablo 6.30. G_2 Sisteminde PI-PD için PSO CS ve FA sonuçları

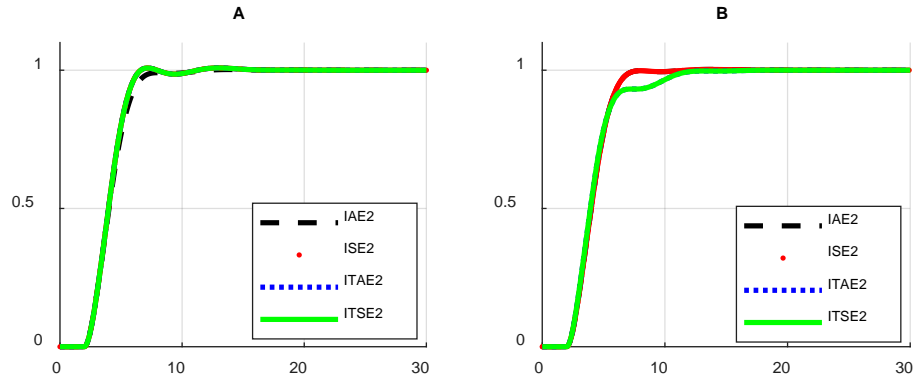
	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	5097,08	5248,22	261,778	4755,736	4789,729	40,804	4755,890	4758,512	3,72412
ISE2	8298,35	9179,22	1525,71	6467,159	6766,407	394,13	6458,631	6548,619	115,576
ITAE2	4648,92	4653,41	3,88255	4602,344	4880,532	308,82	4481,195	4487,354	5,98869
ITSE2	6155,57	6155,57	6,08710	5538,289	5579,068	70,629	5417,584	5420,015	2,93287



Şekil 6.29. PSO için PI-PD denetleyici sonuçları



Şekil 6.30. CS için PI-PD denetleyici sonuçları

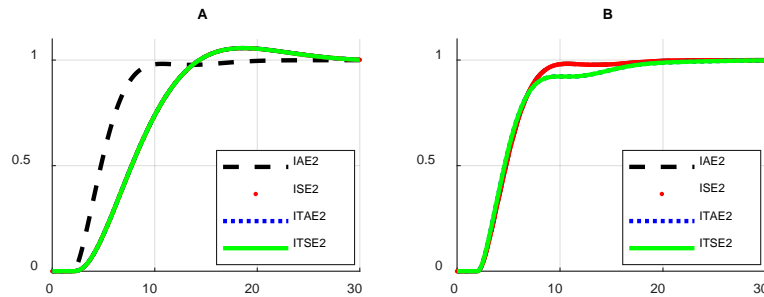


a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre
Şekil 6.31. FA için PI-PD denetleyici sonuçları

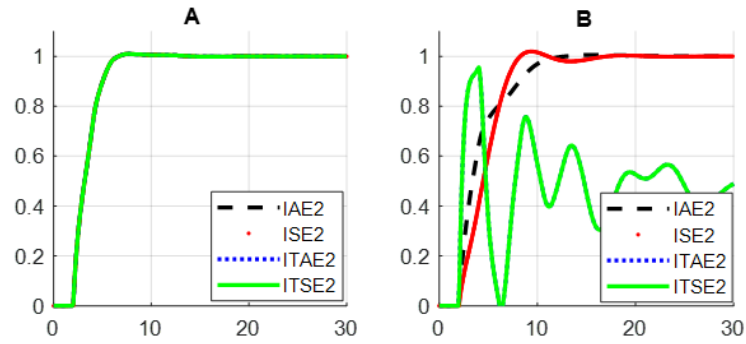
$G_2(s)$ sistemi için PID-PD denetleyici tasarım sonuçlarının analizleri Tablo 6.31.'de verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.32.'de, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.33.'de, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.34.'de verilmiştir. PID-PD denetleyici tasarımında algoritmaların başarıları performans kriterine göre farklılık göstermiştir.

Tablo 6.31. G_2 Sisteminde PID-PD için PSO CS ve FA sonuçları

	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	5097,08	5097,08	1,08e-05	4559,142	4686,190	119,78	4548,501	4552,301	3,73065
ISE2	7919,07	11864,5	6508,00	6233,544	7351,319	983,07	5617,744	5659,557	41,8579
ITAE2	4655,65	4655,65	2,97e-05	10947,23	14121,76	5498,4	4190,412	4193,603	2,9643
ITSE2	4765,67	5692,27	802,458	4728,847	5036,713	518,37	4769,680	4791,125	20,5671

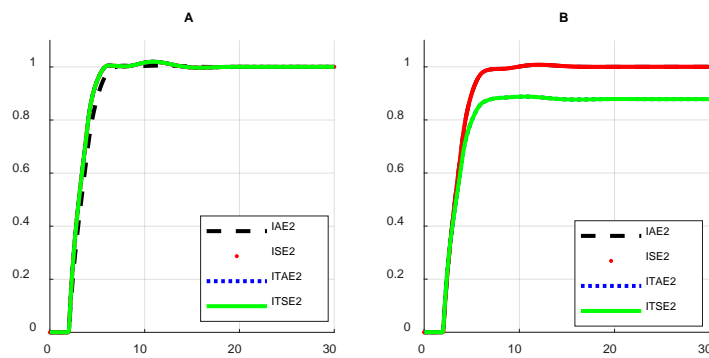


a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre
Şekil 6.32. PSO için PID-PD denetleyici sonuçları



a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.33. CS için PID-PD denetleyici sonuçları



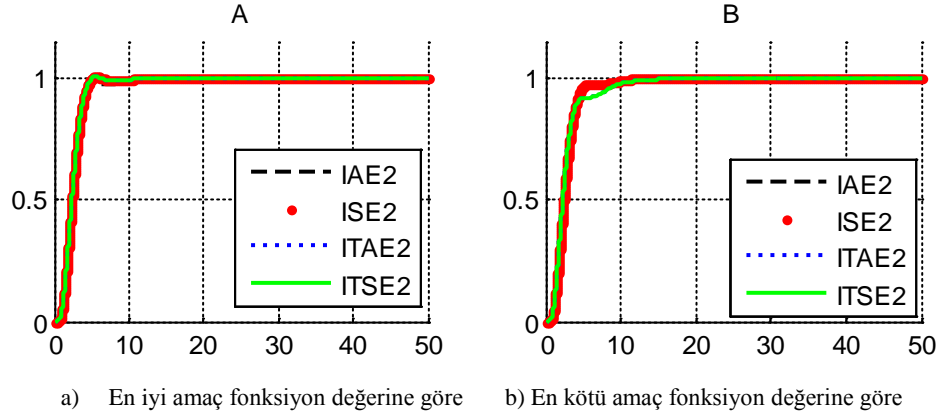
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.34. FA için PID-PD denetleyici sonuçları

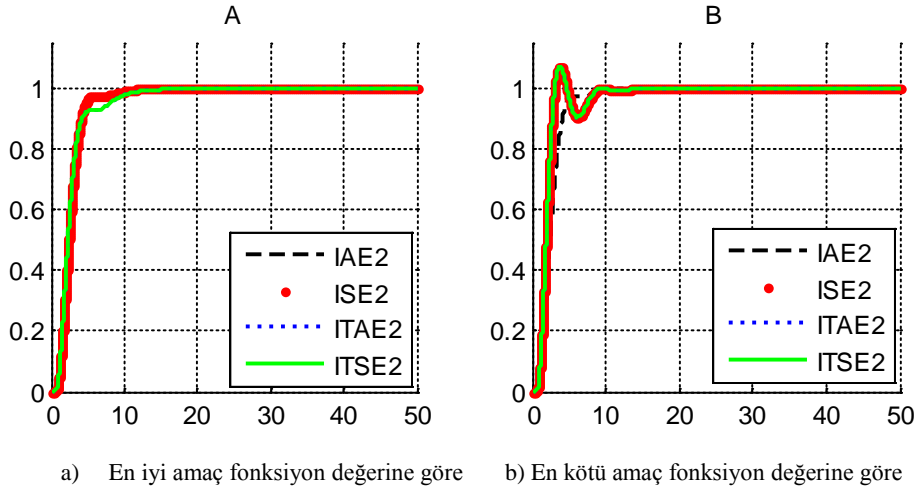
$G_3(s)$ sistemi için PI denetleyici tasarım sonuçlarının analizleri Tablo 6.32.'de verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.35.'de, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.36.'da, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.37.'de verilmiştir. PI denetleyici tasarımı için PSO ve FA algoritmaları ile daha iyi sonuçlar elde edilmiştir.

Tablo 6.32. G_3 Sisteminde PI için PSO CS ve FA sonuçları

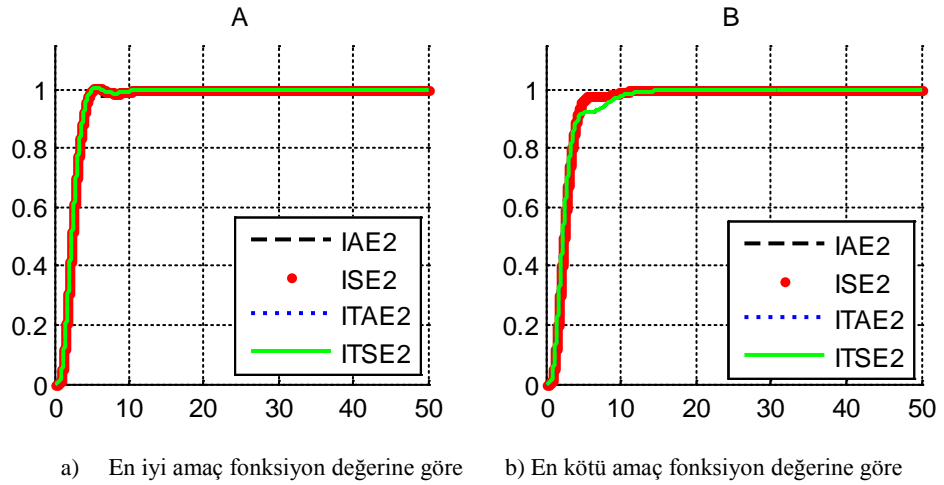
	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	4235,59	4235,59	5,7e-06	4235,617	4235,791	0,2012	4235,603	4235,605	0,00210
ISE2	4768,69	4768,69	2,30e-05	4769,041	5152,681	661,04	4768,703	4768,712	0,00784
ITAE2	3969,62	3969,62	4,07975	3969,735	4026,805	66,442	3969,622	3969,631	0,01407
ITSE2	4048,71	4048,71	1,94e-06	4048,720	4048,739	0,0305	4048,719	4048,724	0,00496



Şekil 6.35. PSO için PI denetleyici sonuçları



Şekil 6.36. CS için PI denetleyici sonuçları

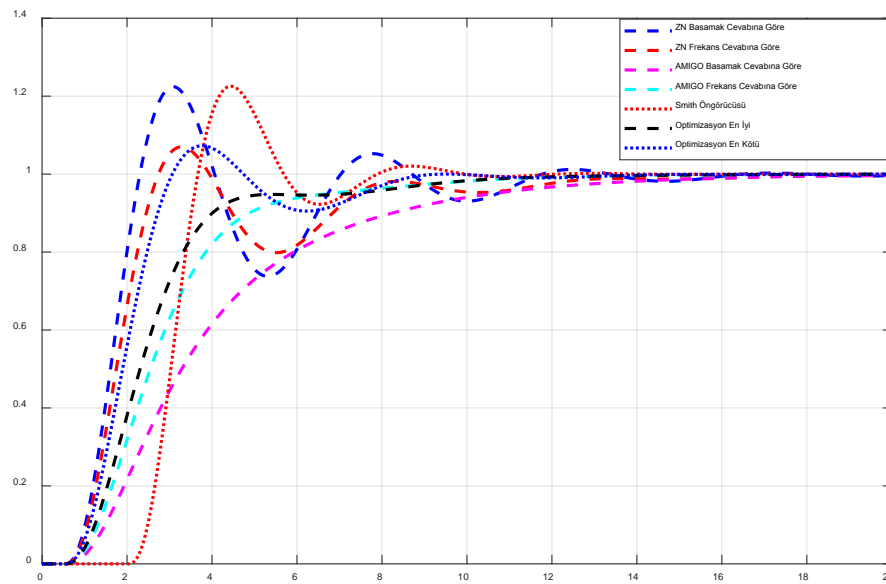


Şekil 6.37. FA için PI denetleyici sonuçları

$G_3(s)$ sistemi için tasarlanan PI denetleyici Zeigler Nichols, AMIGO veya Smith gibi farklı teknikler ile denetlendiğinde kullanılması gereken denetleyici katsayıları Tablo 6.33.'de verilmiştir. Elde edilen denetleyici katsayılarına göre sistem cevapları Şekil 6.38.'de verilmiştir. Elde edilen sonuçlara göre PID için en iyi sonuç, optimizasyonla elde edilen en iyi değer ile ve yine en kötü sonuç optimizasyonun en kötü denetleyici katsayıları ile elde edilmiştir.

Tablo 6.33. $G_3(s)$ Sistemi için Farklı PI Denetleyici Katsayıları

	K_P	K_I
Zeigler Nichols Basamak Cevabına Göre	4,39264	1,5688
Zeigler Nichols Frekans Cevabına Göre	3,62587	1,2503
AMIGO Basamak Cevabına Göre	0,9929	0,70926
AMIGO Frekans Cevabına Göre	1,492215	0,99481
Smith Öngörücüsü	9,54577	7,3429
Optimizasyona Göre En iyi	1,8416	1,4384
Optimizasyona Göre En Kötü	2,8288	1,4384



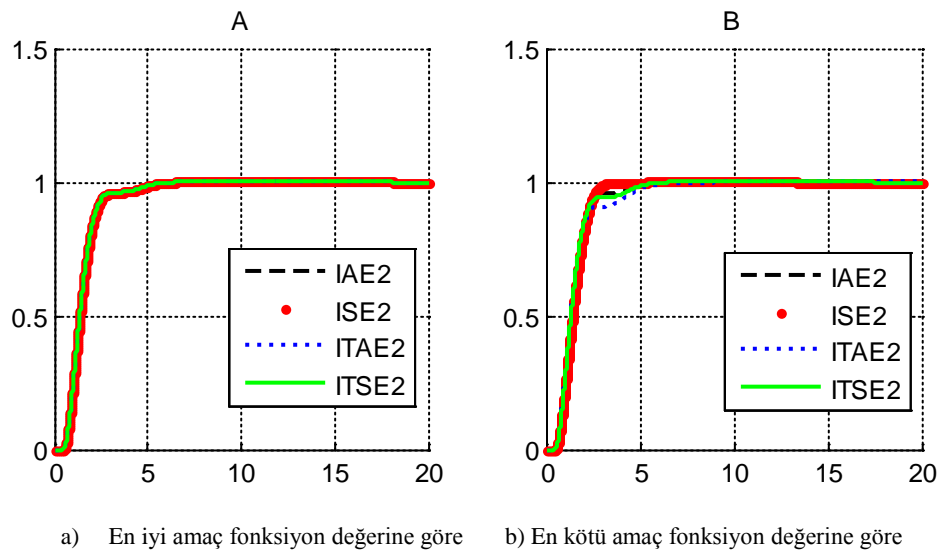
Şekil 6.38. $G_3(s)$ Sistemi için Farklı PI Denetleyici Sonuçları

$G_3(s)$ sistemi için PID denetleyici tasarım sonuçlarının analizleri Tablo 6.34.'de verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre

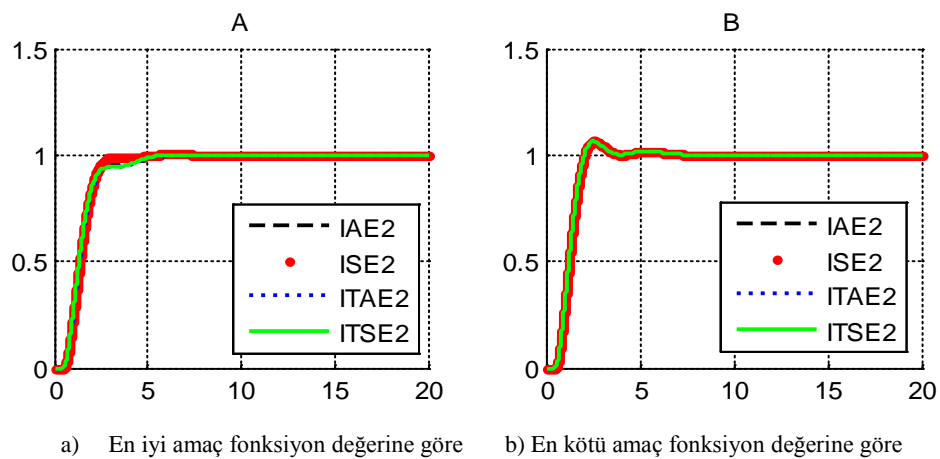
en iyi ve en kötü sonuçları Şekil 6.39.'da, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.40.'da, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.41.'de verilmiştir. PID denetleyici için, CS algoritması ile denetleyici tasarımında aşım meydana gelmiş, fakat diğer algoritmalarda bu durum gözlemlenmemiştir. Ayrıca amaç fonksiyonunun minimum değeri bakımından da PSO ve FA algoritmaları ile daha iyi sonuçlar elde edilmiştir.

Tablo 6.34. G₃ Sisteminde PID için PSO CS ve FA sonuçları

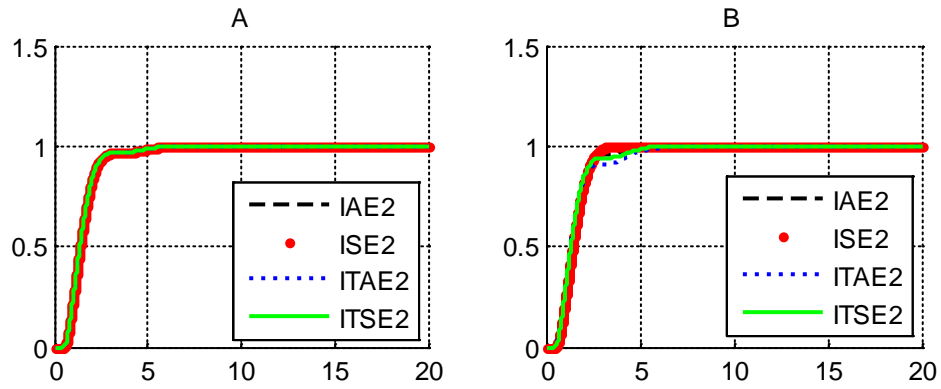
	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	3850,28	3850,28	0,00058	3850,349	4012,345	270,11	3850,385	3850,689	0,40054
ISE2	3818,40	3818,40	0,00474	3831,784	4016,496	243,18	3818,556	3818,688	0,18399
ITAE2	3711,67	3711,67	0,00036	3717,007	3839,180	171,04	3711,692	3711,823	0,11344
ITSE2	3578,16	3578,16	0,00052	3578,674	3717,742	123,91	3578,209	3578,230	0,02163



Şekil 6.39. PSO için PID denetleyici sonuçları



Şekil 6.40. CS için PID denetleyici sonuçları



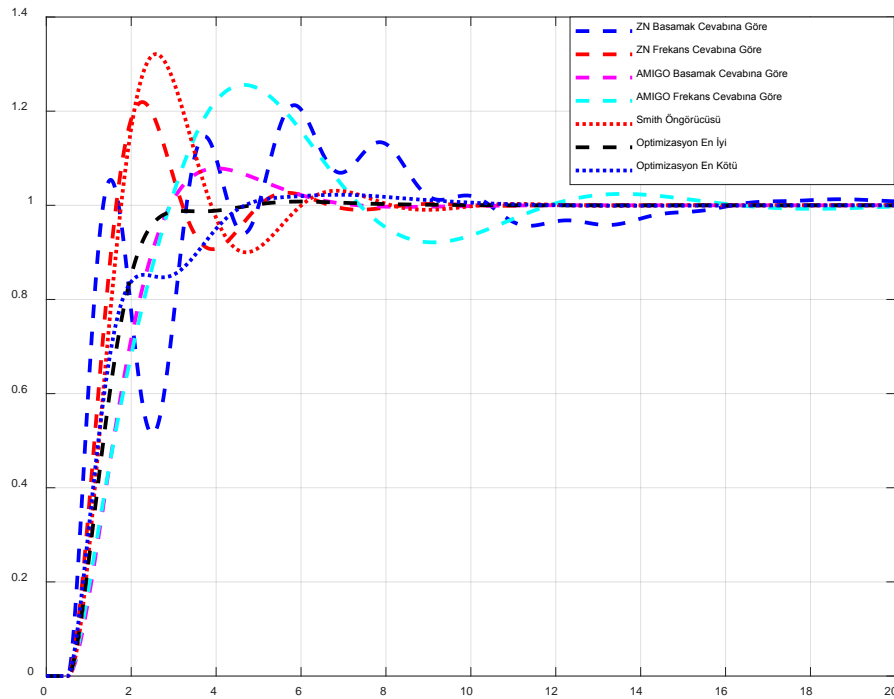
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.41 FA için PID denetleyici sonuçları

$G_3(s)$ sistemi için tasarlanan PID denetleyici Zeigler Nichols, AMIGO veya Smith gibi farklı teknikler ile denetlendiğinde kullanılması gereken denetleyici katsayıları Tablo 6.35.'de verilmiştir. Elde edilen denetleyici katsayılarına göre sistem cevapları Şekil 6.42.'de verilmiştir. Elde edilen sonuçlara göre PID için en iyi sonucu optimizasyonla elde edilen denetleyici katsayıları vermiştir.

Tablo 6.35. $G_3(s)$ Sistemi için Farklı PID Denetleyici Katsayıları

	K_P	K_i	K_d
Zeigler Nichols Basamak Cevabına Göre	2,5413	3,1375	5,96125
Zeigler Nichols Frekans Cevabına Göre	5,47456	3,0008	2,4909
AMIGO Basamak Cevabına Göre	2,87742	2,0553	1,09526
AMIGO Frekans Cevabına Göre	2,0681	2,68853	1,4592
Smith Öngörücüsü	12,62996	14,686	2,7544
Optimizasyona Göre En iyi	3,5504	2,1147	1,7866
Optimizasyona Göre En Kötü	3,2831	2,0302	2,5636

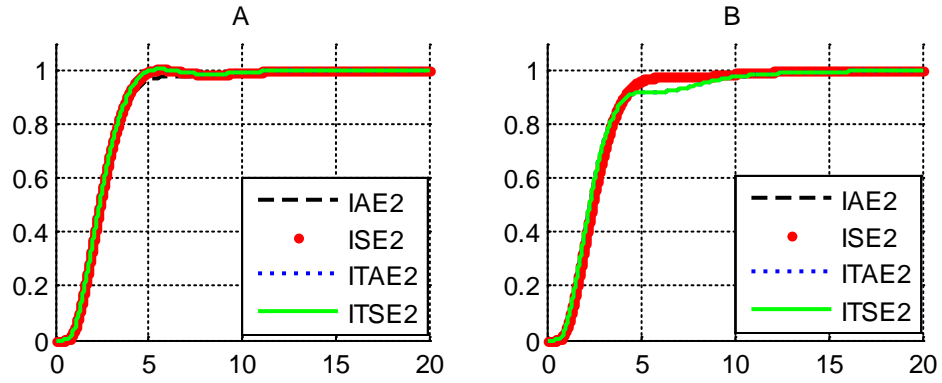


Şekil 6.42. $G_3(s)$ Sistemi için Farklı PID Denetleyici Sonuçları

$G_3(s)$ sistemi için PI-P denetleyici tasarım sonuçlarının analizleri Tablo 6.36.'da verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.43.'de, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.44.'de, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.45.'de verilmiştir. PI-P denetleyici tasarımında, PSO ve FA algoritmaları ile daha iyi sonuçlar elde edilmiştir.

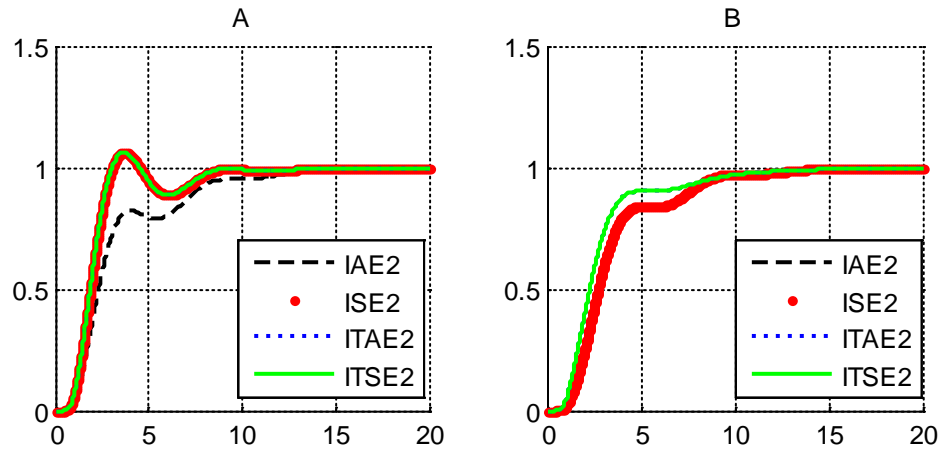
Tablo 6.36. $G_3(s)$ Sisteminde PI-P için PSO CS ve FA sonuçları

	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	4235,59	4768,69	3969,62	4048,719	4235,599	4768,6	3969,621	4048,719	4235,59
ISE2	4235,59	4768,69	3969,62	4048,719	4235,599	4768,6	3969,621	4048,719	4235,59
ITAE2	3,e-05	2e-05	3,31013	3,61e-06	3,0e-05	2e-05	3,3e-05	3,6-06	3e-05
ITSE2	4533,08	4976,06	3971,23	4073,751	4533,086	4976,0	3971,233	4073,751	4533,08



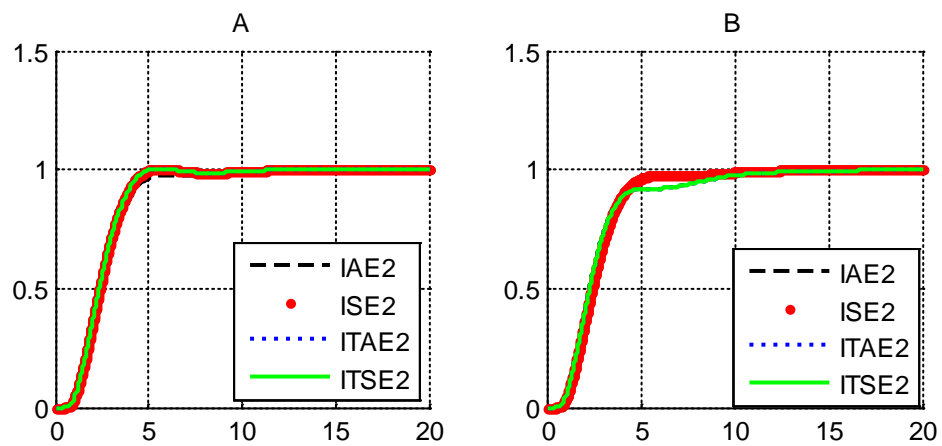
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.43. PSO için PI-P denetleyici sonuçları



a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.44. CS için PI-P denetleyici sonuçları



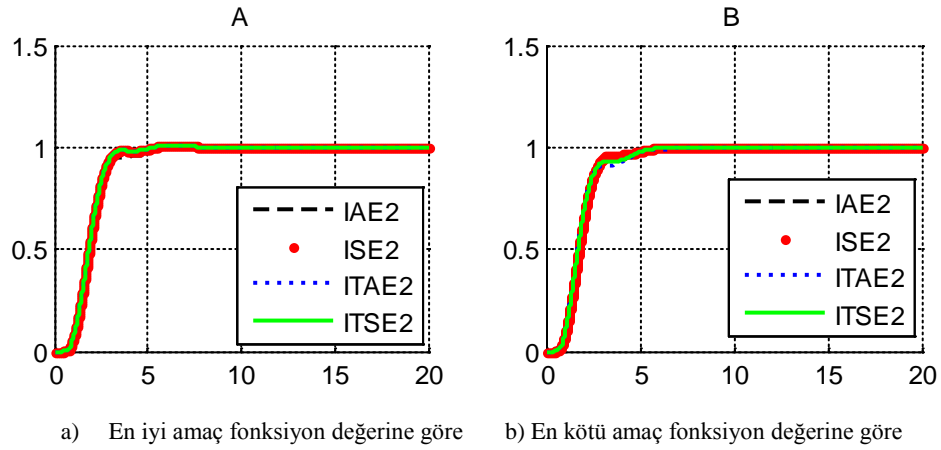
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.45. FA için PI-P denetleyici sonuçları

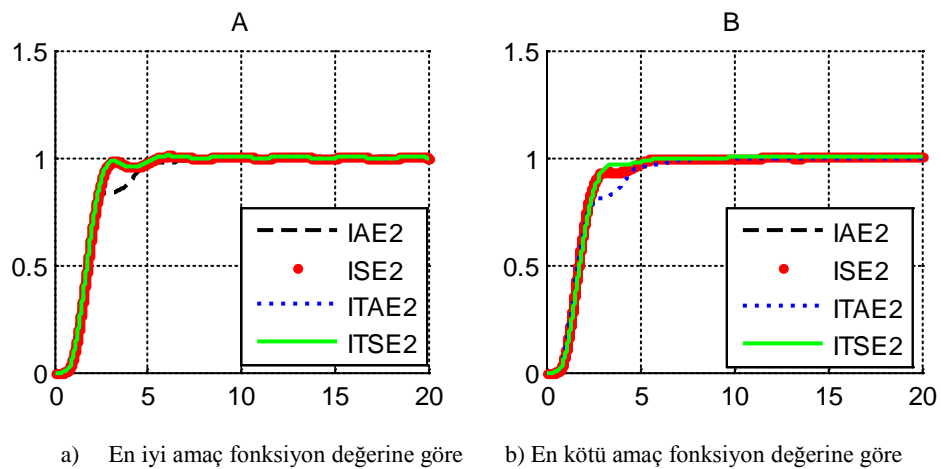
$G_3(s)$ sistemi için PI-PD denetleyici tasarım sonuçlarının analizleri Tablo 6.37.'de verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.46.'da, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.47.'de, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.48.'de verilmiştir. PI-PD denetleyici tasarımı için PSO ve FA algoritmaları ile daha iyi sonuçlar elde edilmiştir.

Tablo 6.37. G_3 Sisteminde PI-PD için PSO CS ve FA sonuçları

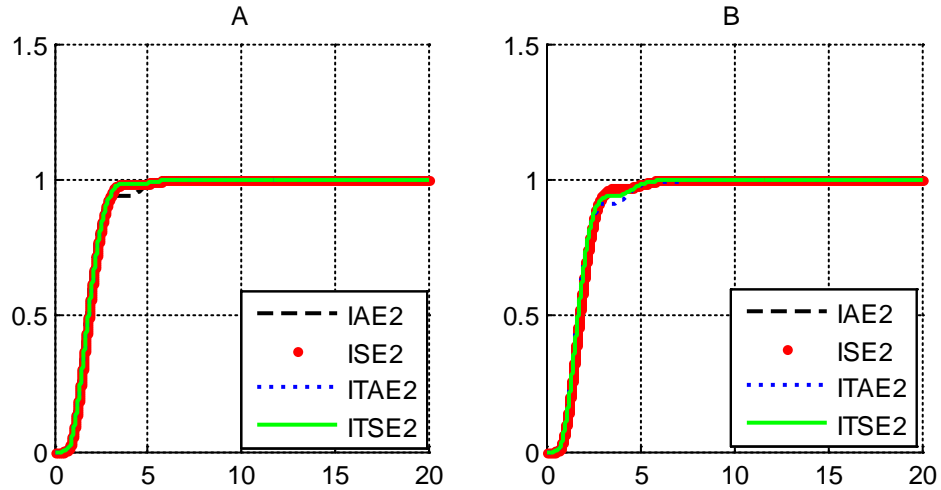
	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	3964,24	3984,59	23,6892	4001,314	4013,642	21,189	3970,972	3979,597	7,84453
ISE2	4102,00	4136,96	37,6825	4170,181	4238,000	78,455	4033,259	4053,931	22,5479
ITAE2	3815,96	3815,97	0,01351	3833,491	3845,768	10,632	3816,151	3816,246	0,08296
ITSE2	3720,79	3729,92	15,7923	3754,316	3833,010	69,921	3721,112	3722,339	1,57978



Şekil 6.46. PSO için PI-PD denetleyici sonuçları



Şekil 6.47. CS için PI-PD denetleyici sonuçları



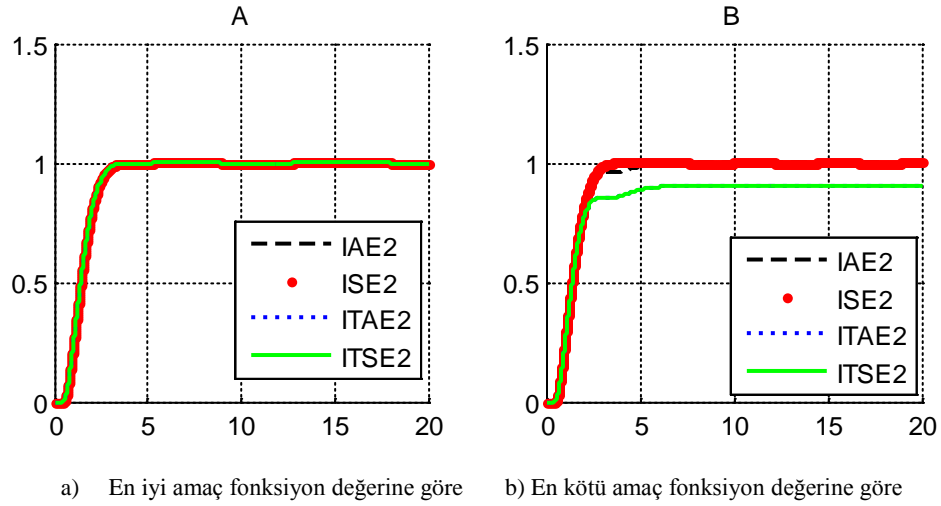
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.48. FA için PI-PD denetleyici sonuçları

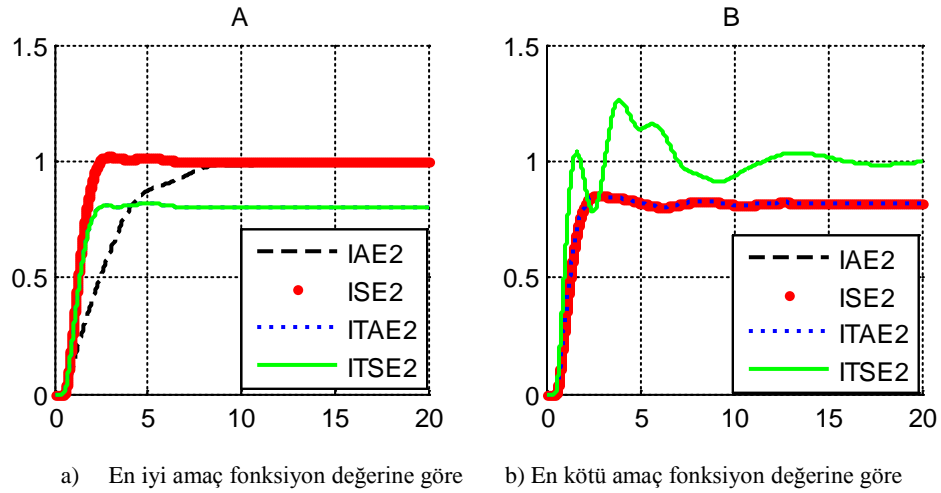
$G_3(s)$ sistemi için PID-PD denetleyici tasarım sonuçlarının analizleri Tablo 6.38.'de verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.49.'da, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.50.'de, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.51.'de verilmiştir. PID-PD denetleyici tasarımında sonuçlar performan kriterine göre değişse de, genel olarak PSO ve FA algoritmaları ile daha iyi sonuçlar elde edilmiştir.

Tablo 6.38. G_3 Sisteminde PID-PD için PSO CS ve FA sonuçları

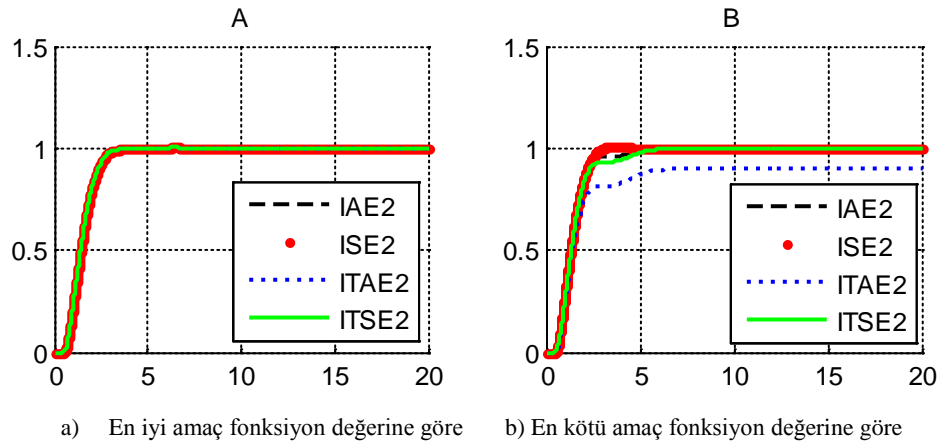
	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	3850,29	3859,51	7,98745	4354,498	5314,854	1145,1	3851,219	3859,676	7,34924
ISE2	3806,47	3812,09	5,81091	3960,158	4553,151	572,27	3805,303	3818,534	15,5322
ITAE2	3563,21	3636,50	95,6576	3824,844	4187,620	369,15	3579,087	3591,095	15,7415
ITSE2	3578,13	3579,15	1,74087	4221,759	6228,841	2034,0	3578,410	3587,971	8,28022



Şekil 6.49. PSO için PID-PD denetleyici sonuçları



Şekil 6.50. CS için PID-PD denetleyici sonuçları

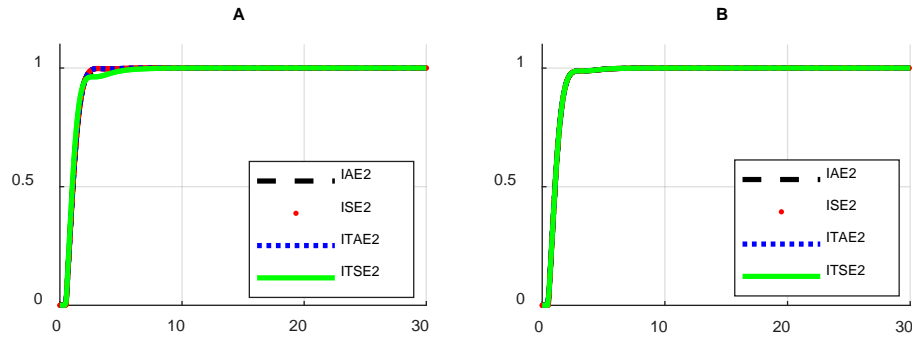


Şekil 6.51. FA için PI-PD denetleyici sonuçları

$G_4(s)$ sistemi için PID denetleyici tasarım sonuçlarının analizleri Tablo 6.39.'da verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.52.'de, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.53.'de, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.54.'de verilmiştir. PID denetleyici tasarımı için PSO ve FA algoritmaları ile daha iyi sonuçlar elde edilmiştir.

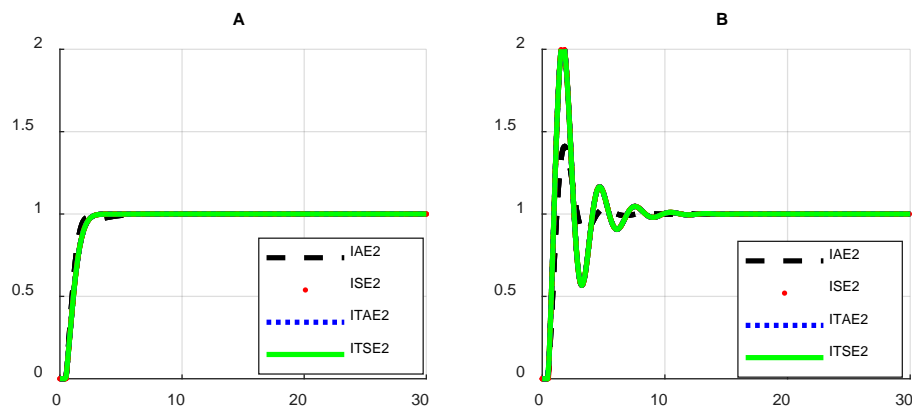
Tablo 6.39. G_4 Sisteminde PID için PSO CS ve FA sonuçları

	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	3724,16	3724,16	1,12e-05	3766,343	5555,243	1932,0	3724,187	3724,307	0,108160
ISE2	3616,63	3616,63	5,51e-06	3628,107	9987,875	6851,7	3616,693	3616,731	0,033567
ITAE2	3621,27	3621,27	5,34e-05	3760,358	9259,481	5219,9	3621,281	3621,335	0,051136
ITSE2	3473,78	3473,78	3,14e-06	4158,204	5494,147	1835,5	3473,822	3473,862	0,046944



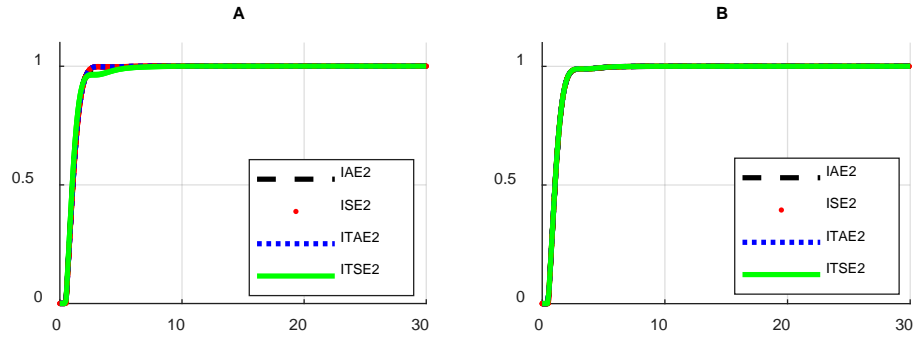
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.52. PSO için PID denetleyici sonuçları



a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.53. CS için PID denetleyici sonuçları



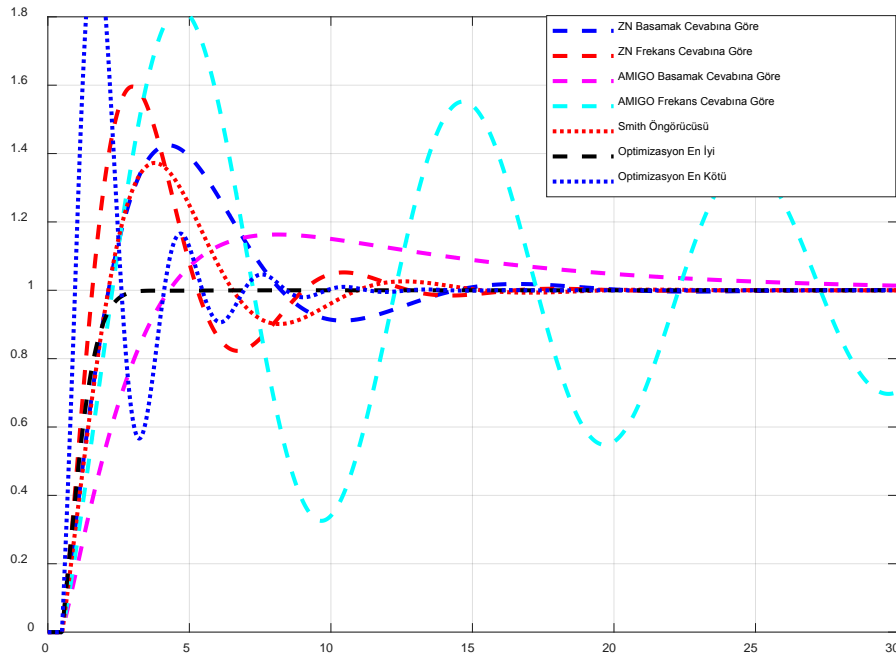
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.54. FA için PID denetleyici sonuçları

$G_4(s)$ sistemi için tasarlanan PID denetleyici Zeigler Nichols, AMIGO veya Smith gibi farklı teknikler ile denetlendiğinde kullanılması gereken denetleyici katsayıları Tablo 6.46.'da verilmiştir. Elde edilen denetleyici katsayılarına göre sistem cevapları Şekil 6.55.'de verilmiştir. Elde edilen sonuçlara göre PID için en iyi sonucu optimizasyonun en iyi değeri verirken, bunun yanısıra kabul edilebilir basamak cevabı olarak ZN ve AMIGO teknikleri ile elde edilen denetleyici katsayılarının olduğu görülmüştür.

Tablo 6.40. $G_4(s)$ Sistemi için Farklı PID Denetleyici Katsayıları

	K_P	K_i	K_d
Zeigler Nichols Basamak Cevabına Göre	0,8	0,26667	0.6
Zeigler Nichols Frekans Cevabına Göre	1,28616	0,5359	0,7716
AMIGO Basamak Cevabına Göre	0,4425	0,0375	0,33
AMIGO Frekans Cevabına Göre	0,6366	0,42441	0,42441
Smith Öngörücüsü Optimizasyona Göre	1,2	0,6	0,6
Optimizasyona Göre	0.8263	0	0.8414
En iyi Optimizasyona Göre	2.9266	1.7381	1.9176
En Kötü			

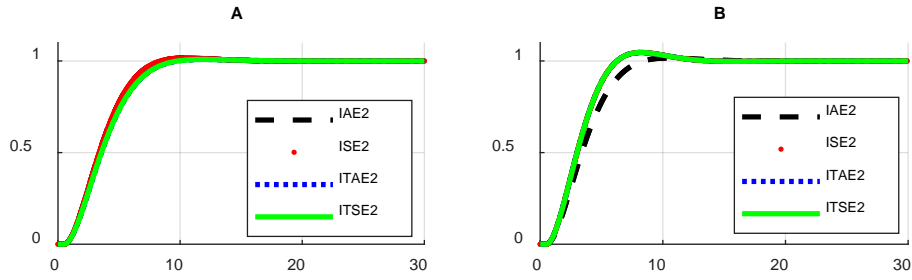


Şekil 6.55. $G_4(s)$ Sistemi için farklı PID denetleyici sonuçları

$G_4(s)$ sistemi için PI-P denetleyici tasarım sonuçlarının analizleri Tablo 6.41.'de verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.56.'da, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.57.'de, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.58.'de verilmiştir. PI-P denetleyici tasarımı için PSO ile daha iyi sonuçlar elde edilmiştir.

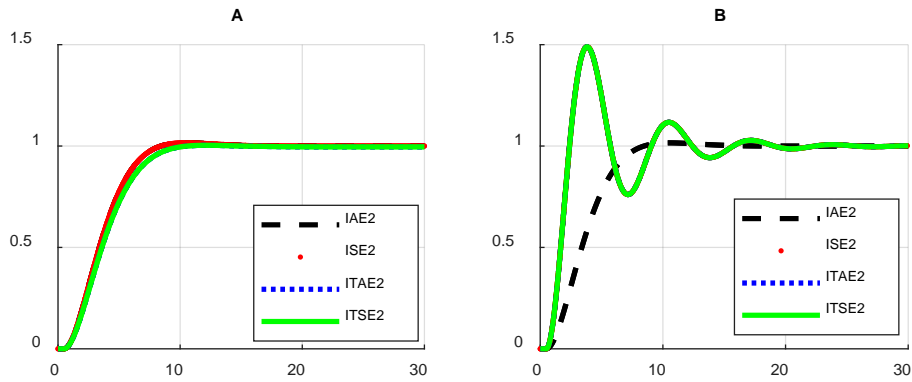
Tablo 6.41. G_4 Sisteminde PI-P için PSO CS ve FA sonuçları

	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	4677,23	4677,23	8,07e-08	4677,231	4677,239	0,0118	4677,233	4677,241	0,012933
ISE2	6217,81	6217,81	1,55e-07	6217,817	9134,974	5052,2	6217,812	6217,813	0,000798
ITAE2	4308,50	4308,50	1,39e-06	4299,289	4307,322	7,5101	4186,005	4186,427	0,577714
ITSE2	4764,66	4767,91	2,81845	4769,859	5168,861	681,47	4765,471	4767,138	2,133852



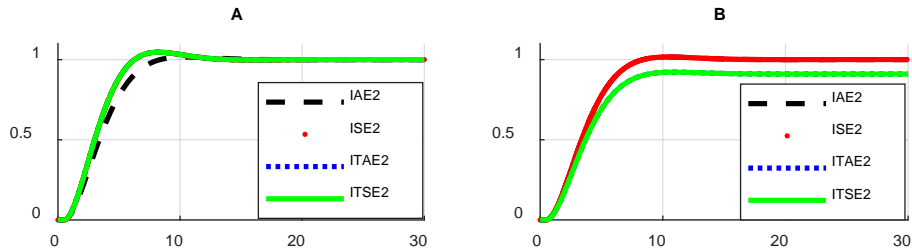
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.56. PSO için PI-P denetleyici sonuçları



a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.57. CS için PI-P denetleyici sonuçları



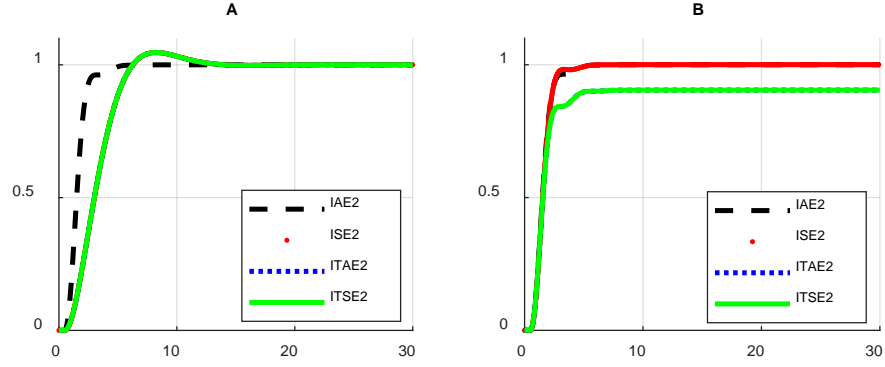
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.58. FA için PI-P denetleyici sonuçları

$G_4(s)$ sistemi için PI-PD denetleyici tasarım sonuçlarının analizleri Tablo 6.42.'de verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.59.'da, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.60.'da, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.61.'de verilmiştir. PI-PD denetleyici tasarımı için performans değerleri bakımından PSO ve FA iyi sonuçlar verirken, sistem yanıtlarına bakıldığında CS ile FA'nın daha iyi sonuçlar verdiği görülmüştür.

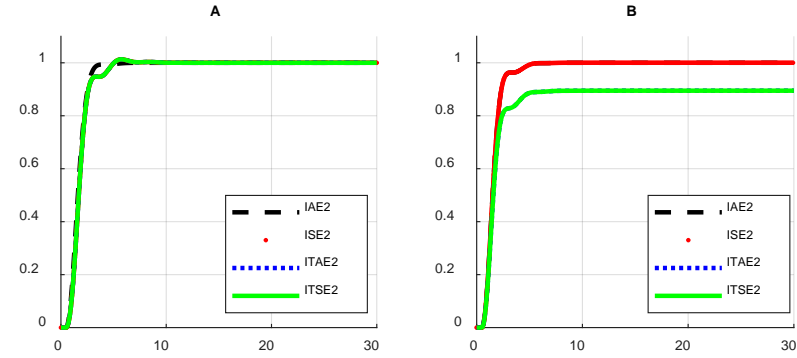
Tablo 6.42. G₄ Sisteminde PI-PD için PSO CS ve FA sonuçları

	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	3878,97	3879,01	0,06503	3879,009	3891,419	10,955	3879,114	3881,075	3,270834
ISE2	3865,84	4649,83	1357,91	3904,970	3995,359	137,97	3890,117	3910,306	34,54448
ITAE2	3602,31	3699,45	84,1184	3608,303	3731,461	107,60	3602,489	3603,120	0,918173
ITSE2	3620,55	3620,55	0,00013	3626,223	3699,822	69,989	3616,693	3617,460	0,723694



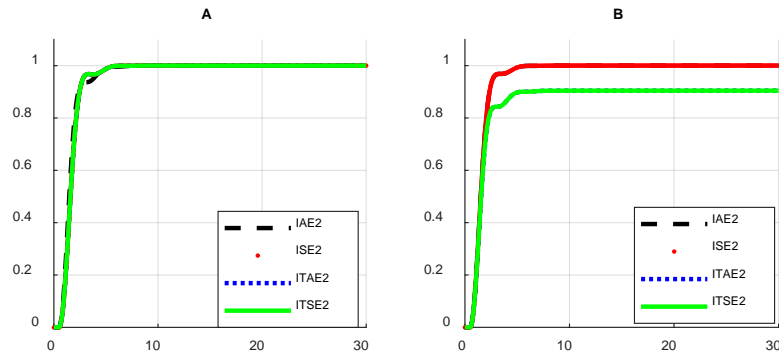
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.59. PSO için PI-PD denetleyici sonuçları



a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.60. CS için PI-PD denetleyici sonuçları



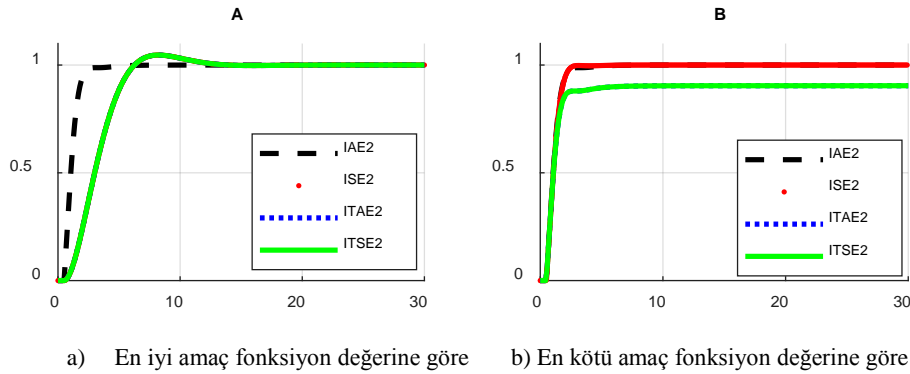
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.61. FA için PI-PD denetleyici sonuçları

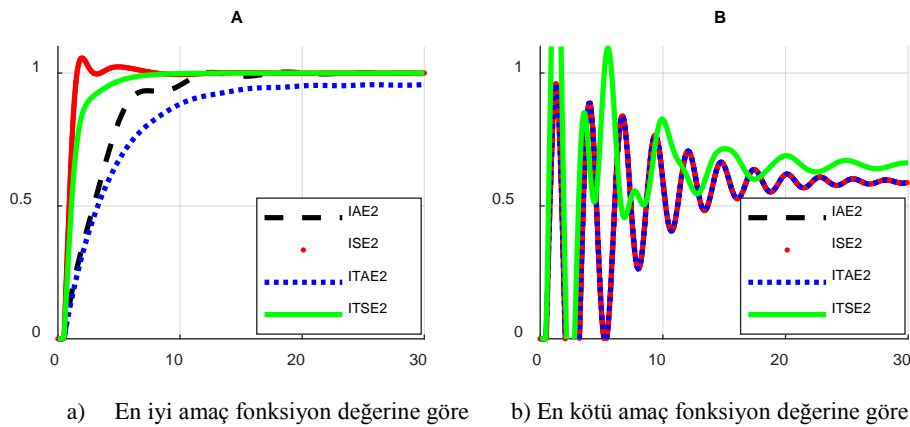
$G_4(s)$ sistemi için PID-PD denetleyici tasarım sonuçlarının analizleri Tablo 6.43.'de verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.62.'de, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.63.'de, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.64.'de verilmiştir. PID-PD denetleyici tasarımında sistem yanıtları bakımından FA algoritması ile elde edilen sonuçlar diğer algoritmaların sonuçlarına göre daha başarılıdır.

Tablo 6.43. G_4 Sisteminde PID-PD için PSO CS ve FA sonuçları

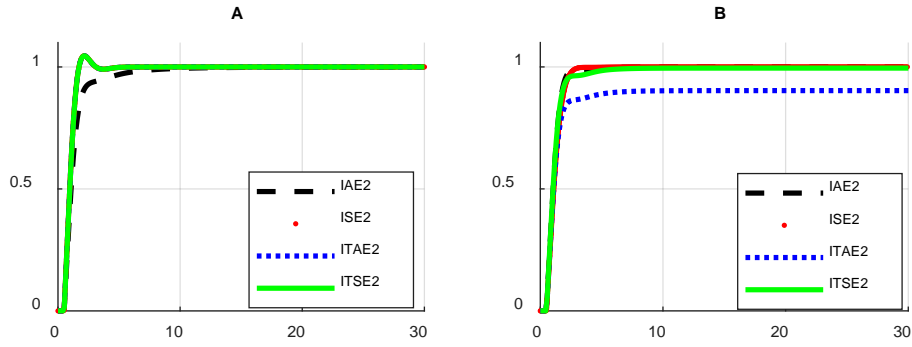
	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	3724,16	3724,16	2,13e-05	4661,108	15273,86	17,8e3	3725,555	3754,082	47,57042
ISE2	3616,63	4483,69	1501,79	4337,024	4672,480	431,80	3626,646	3823,901	170,8810
ITAE2	3468,80	3570,45	88,0288	4129,239	7134,393	4182,8	3474,862	3477,113	2,130762
ITSE2	3473,78	3473,78	3,93e-05	3516,614	38273,47	36,9e3	3471,704	3483,031	10,60898



Şekil 6.62 PSO için PID-PD denetleyici sonuçları



Şekil 6.63. CS için PID-PD denetleyici sonuçları



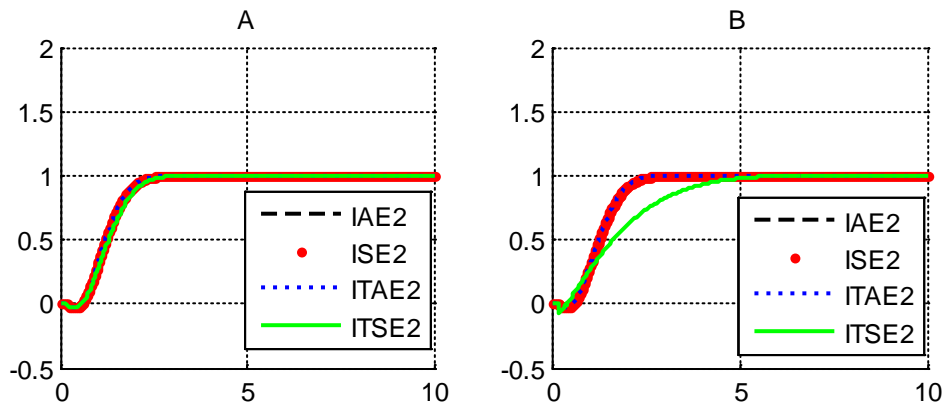
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.64. FA için PID-PD denetleyici sonuçları

$G_5(s)$ sistemi için PI denetleyici tasarım sonuçlarının analizleri Tablo 6.44.'de verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.65.'de, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.66.'da, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.67.'de verilmiştir. PI denetleyici tasarımında sistem yanıtları bakımından PSO algoritması FA ve CS algoritmasına göre sonuçlarına göre daha başarılıdır. Ayrıca Tablo 6.44.'de PSO'nun bu tasarım için standart sapmasının düşük olduğu görülmektedir.

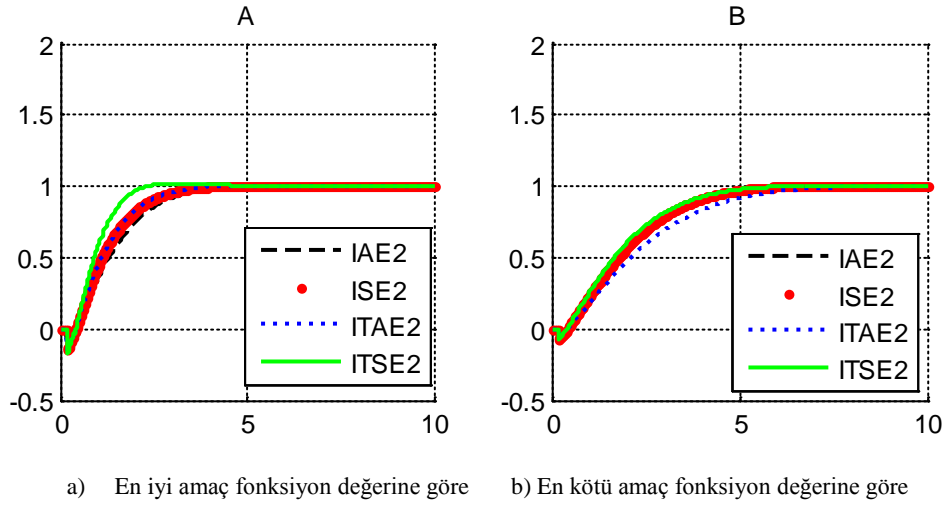
Tablo 6.44. G_5 Sisteminde PI için PSO CS ve FA sonuçları

	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	4344,39	4344,39	1,81e-05	4344,398	4369,510	38,857	4344,396	4344,405	0,009635
ISE2	4448,04	4448,04	2,13e-05	4449,380	4456,987	11,334	4448,045	4448,057	0,016824
ITAE2	4089,20	4089,20	8,59e-06	4089,258	4188,165	169,47	4089,205	4089,213	0,010965
ITSE2	4061,08	4061,08	1,3e-06	4061,344	4242,115	313,02	4061,093	4061,100	0,007368

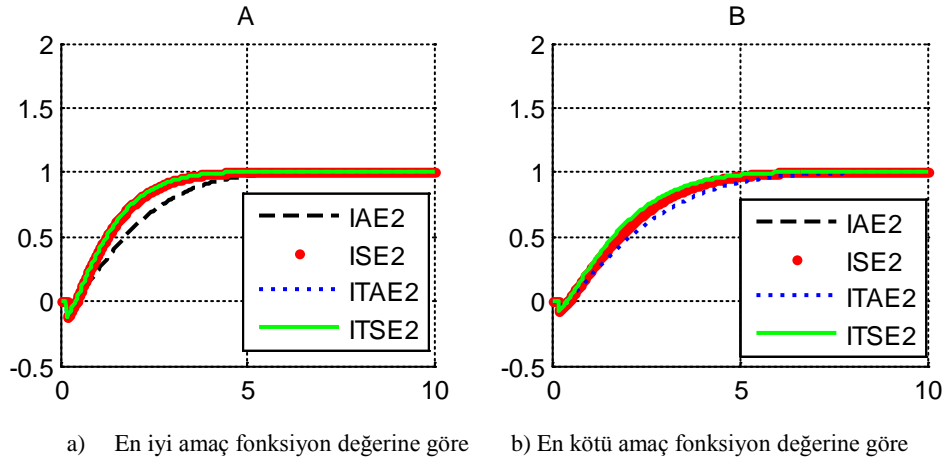


a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.65. PSO için PI Denetleyici Sonuçları



Şekil 6.66. CS için PI denetleyici sonuçları

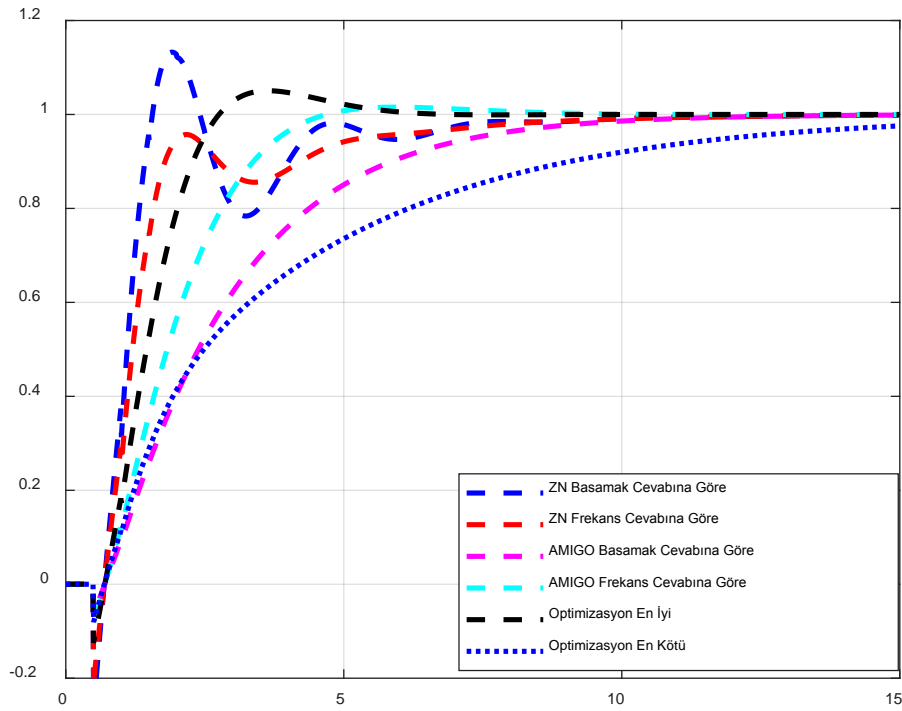


Şekil 6.67. FA için PI denetleyici sonuçları

$G_5(s)$ sistemi için tasarlanan PI denetleyici Zeigler Nichols, AMIGO gibi farklı teknikler ile denetlendiğinde kullanılması gereken denetleyici katsayıları Tablo 6.45.'de verilmiştir. Elde edilen denetleyici katsayılarına göre sistem cevapları Şekil 6.68.'de verilmiştir. Elde edilen sonuçlara göre PID için ZN ve AMIGO teknikleri optimizasyonun en kötü sonucundan iyi sonuç vermiştir. Lakin optimizasyonun iyi sonucu, diğer cevaplara göre daha hızlı bir şekilde kararlı duruma geçmektedir.

Tablo 6.45. $G_4(s)$ Sistemi için Farklı PI Denetleyici Katsayıları

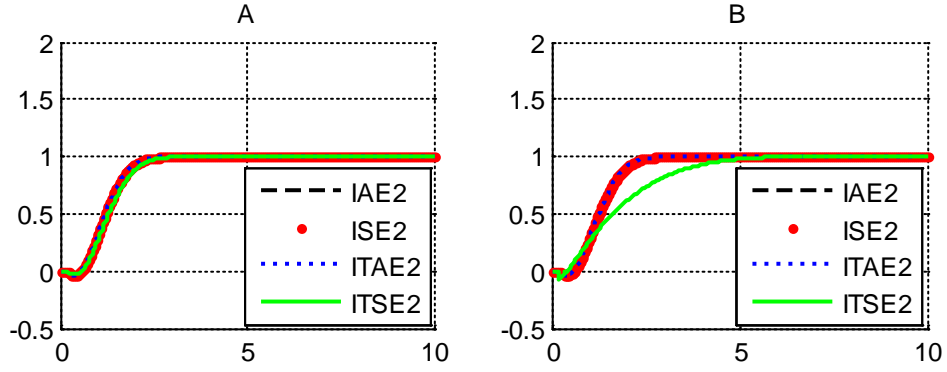
	K_P	K_I
Zeigler Nichols Basamak Cevabına Göre	1,34786	0,67393
Zeigler Nichols Frekans Cevabına Göre	1,0426	0,57924
AMIGO Basamak Cevabına Göre	0,3173	0,33061
AMIGO Frekans Cevabına Göre	0,4163	0,50773
Optimizasyona Göre En iyi	0.6242	0.6965
Optimizasyona Göre En Kötü	0.4030	0.2502

Şekil 6.68. $G_4(s)$ sistemi için Farklı PI denetleyici sonuçları

$G_5(s)$ sistemi için PI-P denetleyici tasarım sonuçlarının analizleri Tablo 6.46.'da verilirken, denetlenmiş sistem cevapları için PSO'nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.69.'da, CS'nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.70.'de, FA'nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.71.'de verilmiştir. PI-P denetleyici tasarımında sistem yanıtları bakımından FA algoritması ile elde edilen sonuçlar diğer algoritmaların sonuçlarına göre daha başarılıdır.

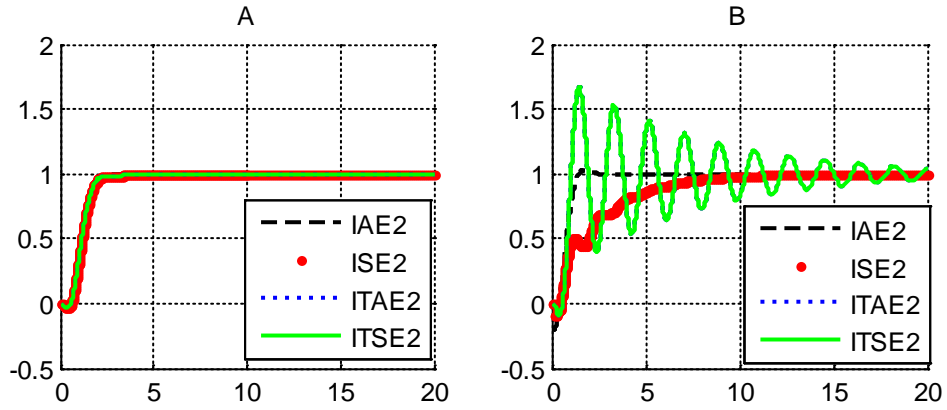
Tablo 6.46. G₅ Sisteminde PI-P için PSO CS ve FA sonuçları

	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	3989,4	3991,12	1,48089	3991,921	4473,030	702,96	3989,46	3990,339	1,28461
ISE2	3873,5	3875,25	1,65378	3887,411	5487,245	1632,4	3873,58	3873,839	0,21774
ITAE2	3893,2	3895,32	2,60799	3903,919	16378,40	21588	3893,21	3893,247	0,02711
ITSE2	3752,6	3958,26	178,096	3917,776	8426,380	6935,1	3749,18	3749,202	0,01386



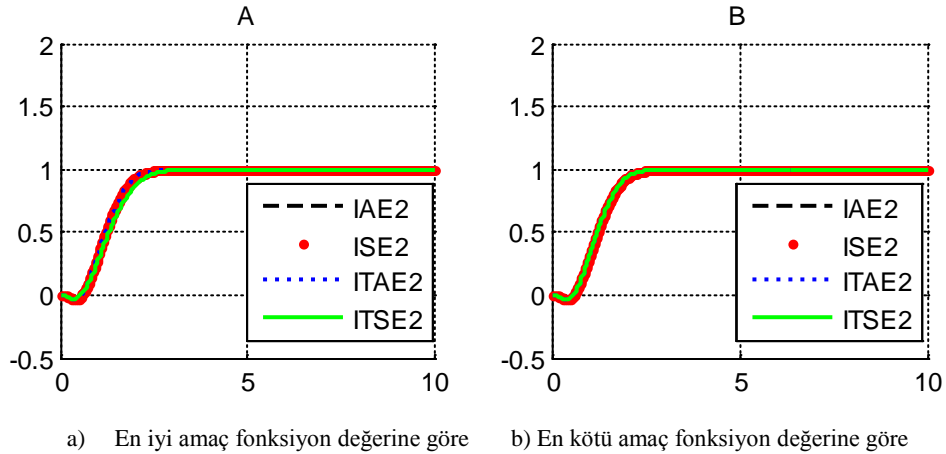
a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.69. PSO için PI-P denetleyici sonuçları



a) En iyi amaç fonksiyon değerine göre b) En kötü amaç fonksiyon değerine göre

Şekil 6.70. CS için PI-P denetleyici sonuçları

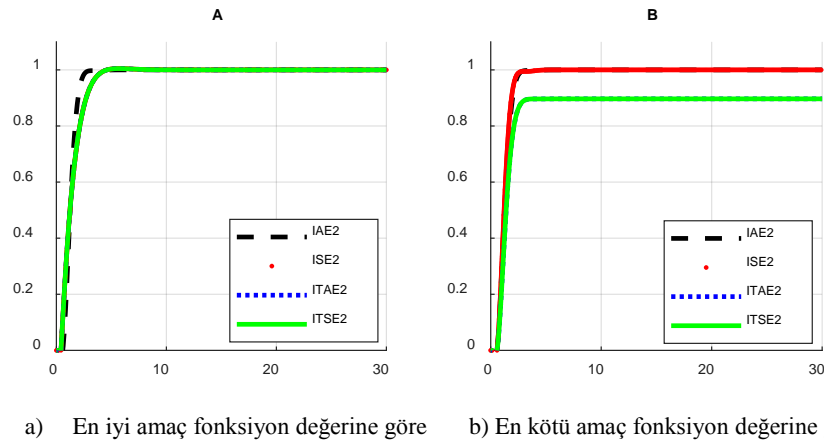


Şekil 6.71. FA için PI-P denetleyici sonuçları

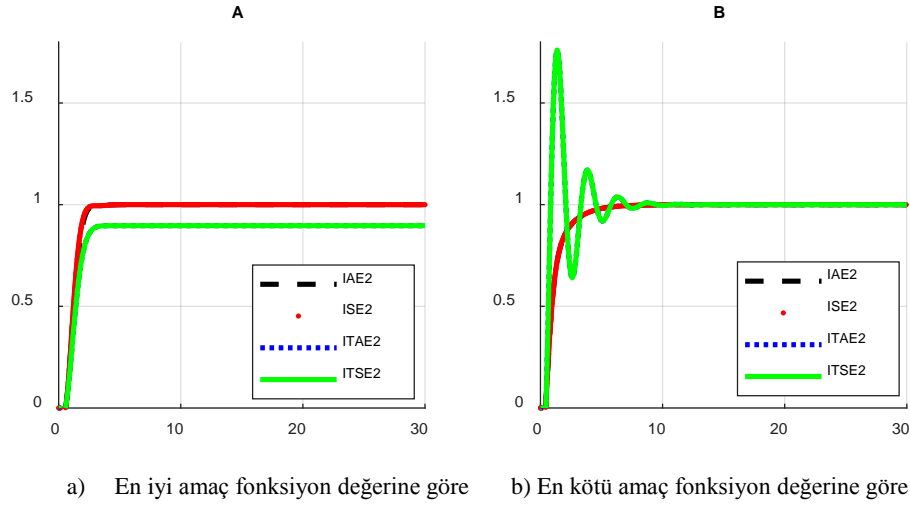
$G_5(s)$ sistemi için PI-PI denetleyici tasarım sonuçlarının analizleri Tablo 6.47’de verilirken, denetlenmiş sistem cevapları için PSO’nun amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.72.’de, CS’nin amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.73.’de, FA’nın amaç fonksiyon değerine göre en iyi ve en kötü sonuçları Şekil 6.74.’de verilmiştir. PI-PI denetleyici tasarımında amaç fonksiyonunun minimum değeri seçilen performans kriterine göre farklılık göstermesine rağmen, sistem yanıtları bakımından FA algoritması ile elde edilen sonuçlar diğer algoritmaların sonuçlarına göre daha başarılıdır.

Tablo 6.47. G_5 Sisteminde PI-PI için PSO CS ve FA sonuçları

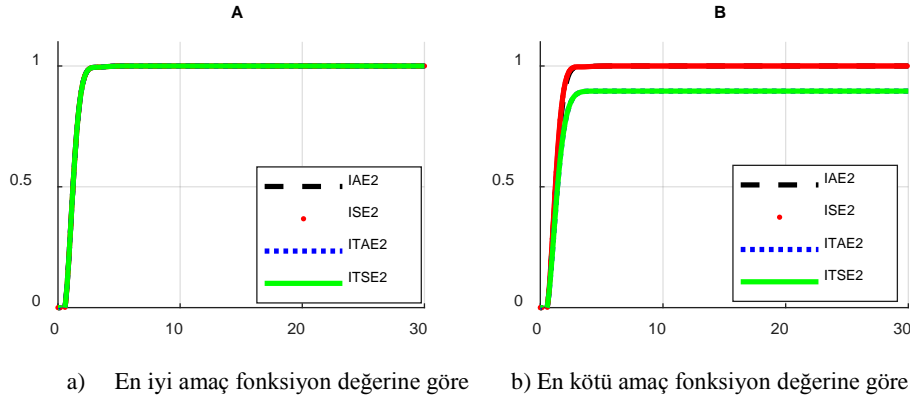
	PSO			CS			FA		
	Eİ	ORT	STD	Eİ	ORT	STD	Eİ	ORT	STD
IAE2	3989,37	3990,92	1,37364	3989,363	4325,110	457,40	3989,488	3990,040	0,886035
ISE2	3873,65	4066,19	330,698	3873,566	3884,983	13,183	3874,186	3874,550	0,522294
ITAE2	3727,74	3903,41	180,938	3724,323	7821,249	7010,8	3724,630	3726,138	1,939920
ITSE2	3749,01	3750,09	1,34941	3742,620	3768,600	41,990	3743,610	3744,784	1,172706



Şekil 6.72. PSO için PI-PI denetleyici sonuçları



Şekil 6.73. CS için PI-PI denetleyici sonuçları



Şekil 6.74. FA için PI-PI denetleyici sonuçları

6.2. Tartışma ve Gelecek Çalışmalar

Bu tez çalışmasında, ölü zamanlı sistemlerin model parametrelerinin tanınması ve elde edilen sistem modelinin denetlenmesi için literatürde kullanılan tekniklerden bazıları tanıtılmıştır. Ardından, bilgisayar teknolojisinin gelişmesi ile ilerlemiş olan, zor ve karmaşık problemlerin çözümlerinde kullanılan sezgisel algoritmalar tanıtılmıştır. Ayrıca, bu algoritmaların modelleme ve denetim problemlerine uyarlanması için kullanılan amaç fonksiyonlarından bahsedilmiştir. Akabinde Parçacık Sürü Optimizasyonu, Gugukkuşu Algoritması ve Ateşböceği Algoritmaları tanıtılmıştır. Bu algoritmaların, Basamak Cevabı yöntemi, Auto-tuning tekniğinde kullanılan tek kanallı ve çift kanallı röle testleri ile birlikte kullanılması sonucu, sistem parametrelerinin belirlenmesi sağlanmıştır. Ayrıca modeli bilinen sistemler için,

Parçacık Sürü Optimizasyonu, Gugukkuşu Algoritması ve Ateşböceği Algoritmaları kullanılarak denetleyici tasarımları gerçekleştirilmiştir. Ardından bu işlemi kolay hale getirilebilmesi için, Matlab 2017a da bir arayüz programı hazırlanmıştır. Bu arayüz kullanılarak beş farklı sistem için hem parametre kestirimi, hemde denetleyici tasarımları gerçekleştirilmiştir.

Bu tez çalışmasında elde edilen sonuçlara göre parametre kesitiriminde PSO ve FA alogoritmaları daha başarılı olduğu görülmüştür. Elde edilen sonuçlardan bazıları literatürde bulunan çalışmalar ile kıyaslanmış ve optimizasyon tekniğinin daha başarılı olduğu görülmüştür. Ardından elde edilen sisteme çeşitli denetleyici yapılmıştır. Optimizasyonla elde edilen denetleyici ile basamak ve frekans cevabına göre Zeigler Nichols tekniği, revize edilmiş olan AMIGO teknikleri ve kimi uygulama için Smith öngörücüsü ile denetleyici tasarımı yapılmıştır. Elde edilen sonuçlara bakıldığında genel olarak PSO ve FA algoritmaları ve Smith öngörücüsünün bazı uygulamaları ile daha başarılı sonuçlar elde edilmiştir.

Algoritmalar ile yapılan çalışmalarda, hepsinin aynı özellikler ile kıyas edilebilmesi için iterasyon sayısı ve sürü sayısı kısıtlı tutulmuştur. Bunun haricinde algoritmalara ait parametreler, literatürde yaygın olarak kullanılan parametreler olarak seçilmiştir. Bu kısıtlar ile elde edilen sonuçlarda, sürünün lokal bölgede takılmış olması yada yakınsama yapamadan işlemi sonlandırmış olması ihtimali vardır. Ancak algoritmalara ait bu parametreler probleme göre değiştirildiğinde, bazı sonuçlarda daha iyi başarı elde edildiği görülmüştür.

Bundan sonraki çalışmalarda algoritmalara ait parametreler değiştirilerek, bu problemler için başarısı araştırılabilir.

Bu çalışmada parametre optimizasyonu yapılırken, giriş ve çıkışta gürültü olmadığı varsayımı yapılmıştır. Diğer taraftan, sistem girişinde ve çıkışında gürültü olmuş olsa idi, yapısı itibari ile rölenin bu gürültüyü bastıracağı düşünülebilir. Dolayısı ile çalışmada bu durum göz ardı edilerek, gerçek sistem çıkışı ile model çıkışı arasındaki işaretlerin farkı kullanılarak sistem parametreleri optimize edilmiştir.

Yine gelecek çalışma olarak, iki kanallı bir röle geliştirilerek, rölelerin genlikleri ve histeresiz değerlerini sisteme adapte edecek bir algoritma ile sistemin osilasyona girmesinin sağlanabileceği düşünülmektedir. Bu röle, bilinen bir çok sisteme uygulanarak, bir tablo oluşturulabilir. Çıkarılan tablo ile verileri sisteme adapte olan rölenin parametreleri arasında ilişki kurularak normalize edilebilir. Bu sayede, her sistem bu tabloya göre sınıflandırılabilir ve parametreleri belirlenebilir.

Rölenin sadece kare histeresiz yapısının yanında, farklı harmonikleri de içeren histeresiz yapıları ve asimetrik özelliği katılarak bu alan üzerinde araştırmalar yapılabilir.

Uygulanan performans indekslerinin yanısıra, yüzde aşım, oturma zamanı, yükselme zamanı ve kararlı durum hatası gibi farklı performans kriterlerini içeren indekslerin performansları da incelenebilir.

Bu tip algoritmalar bu sistemlerin yanısıra, nonlinear zamanla değişen sistemlerin model parametrelerinin belirlenmesinde ve denetiminde, ölü zaman içeren ve ölü zamanı zamanla değişen sistemlerin de tanınması ve denetiminde de kullanılabilir.

Algoritmalar geliştirilerek, melez veya modifiye edilmiş yeni algoritmalar oluşturulabilir. Oluşturulan bu algoritmalarda, parametrik incelemeler ile algoritmanın başarısı incelenebilir. Ayrıca oluşturulan melez algoritmaların bu tip problemler üzerindeki başarısı incelenebilir.

KAYNAKLAR

- [1] J. C. Maxwell, "I. On governors," Proceedings of the Royal Society of London, vol. 16, pp. 270-283, 1868.
- [2] E. J. Routh, A treatise on the stability of a given state of motion: particularly steady motion: Macmillan and Company, 1877.
- [3] J. G. Ziegler, N. B. Nichols, and N. Y. Rochester, "Optimum Settings for Automatic Controllers," 1942.
- [4] K. J. Astrom and T. Hagglund, Advanced PID Control-ISA Instrumentation, Systems and Automation Society Lund, Sweden: The Instruments Systems and Automation Society, 2006.
- [5] A. Bir, "Otomatik Kontrol Sistemlerinin Öncüleri," Elektrik Mühendisleri Odası Aylık Yayın Organı, vol. 17, 1983.
- [6] Y. Z. Tsytkin, "Relay Control Systems," Cambridge University Press, 1984.
- [7] Y. Z. Tsytkin, "Frequency responses of relay systems," Automat. i Telemekh., vol. 20, pp. 1603-1610, 1959.
- [8] K. J. Åström and T. Hägglund, "Automatic Tuning of Simple Regulators With Specifications on Phase and Amplitude Margins," Automatica, vol. 20, pp. 645-651, 1984.
- [9] W. L. Luyben, "Derivation of Transfer Functions for Highly Nonlinear Distillation Columns," Ind. Eng. Chem. Res. 1987,26, 2490-2495, 1987.
- [10] W. Li, E. Eskinat, and W. L. Luyben, "An improved autotune identification method," Industrial & engineering chemistry research, vol. 30, pp. 1530-1541, 1991.
- [11] E. Eskinat, S. H. Johnson, and W. L. Luyben, "Use of Auxiliary Information in System Identification," Ind. Eng. Chem. Res. 1993, 1993.
- [12] J. Kennedy and R. Eberhart, "Particle swarm optimization," in Neural Networks, 1995. Proceedings., IEEE International Conference on, 1995, pp. 1942-1948.

- [13] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Micro Machine and Human Science*, 1995. MHS'95., Proceedings of the Sixth International Symposium on, 1995, pp. 39-43.
- [14] S. W. Sung, J. H. Park, and I.-B. Lee, "Modified Relay Feedback Method," 1995.
- [15] W. K. Ho, E. B. Feng, and O. P. Gan, "A novel relay auto-tuning technique for processes with integration," *Control Engineering Practice* 4.7, vol. 7, pp. 923-928, 1996.
- [16] K. K. Tan, T. H. Lee, and Q. G. Wang, "Enhanced Automatic Tuning Procedure for Process Control of PI/ PID Controllers," *AIChE Journal*, vol. Vol. 42, No. 9, 1996.
- [17] S. H. Shen, J. S. Wu, and C. C. Yu, "Use of biased-relay feedback for system identification," *AIChE Journal*, vol. 42, pp. 1174-1180, 1996.
- [18] Q.-G. Wang, C.-C. Hang, and Q. Bi, "Process frequency response estimation from relay feedback," *Control Eng. Practice*, Vol. 5, No.9, pp. 1293-1302, 1997.
- [19] Q. Bit, Q.-G. Wang, and C.-C. Hang, "Relay-based Estimation of Multiple Points on Process Frequency Response," *Automatica*, Pergamon, vol. 33, No. 9, pp. 1753-1757, 1997.
- [20] Q.-G. Wang, C.-C. Hang, and B. Zou, "Low-Order Modeling from Relay Feedback," *Ind. Eng. Chem. Res.* 1997, 36, 375-381, 1997.
- [21] M. Friman, "Automatic Re-Tuning of PI Controllers in Oscillating Control Loops," in *Proceedings of the 36th Conference on Decision & Control*, San Diego, California USA 1997, pp. 3643-3647.
- [22] M. Friman and K. V. Waller, "A two-channel relay for autotuning," *Industrial & engineering chemistry research*, vol. 36, pp. 2662-2671, 1997.
- [23] Y. Mitsukura, T. Yamamoto, and M. Kaneda, "A Design of Self-Tuning PID Controllers Using a Genetic Algorithm," in *Proceedings of the American Control Conference June 1999*, San Diego. Californi, 1999.
- [24] W. L. Luyben, "Getting More Information from Relay-Feedback Tests," *Ind. Eng. Chem. Res.* 2001, 40, 4391-4402, 2001.
- [25] T. Kealy and A. O'Dwyer, "Closed loop identification of a first order plus dead-time process model under PI control," 2002.

- [26] Z. L. Gaing, "A Particle Swarm Optimization Approach for Optimum Design of PID Controller in AVR System," *IEEE Transactions on Energy Conversion*, vol. 19, pp. 384-391, 2004.
- [27] D. I. Wilson, "Relay-based PID Tuning," 2005.
- [28] I. Kaya, "Parameter Estimation for Integrating Processes Using Relay Feedback Control under Static Load Disturbances," *Ind. Eng. Chem. Res.*, 2006.
- [29] A. F. Boz and S. Yavuz, "Sıfırsız 3 Kutuplu Sistemler İçin Optimal PI-PD Denetleyici Tasarım Yöntemi," *Politeknik Dergisi*, vol. 11, 2008.
- [30] A. F. Boz and Y. SARI, "İki sıfırlı standart formlar ve optimal PID-PD denetleyici tasarımı," presented at the 5. Uluslararası İleri Teknolojiler Sempozyumu(IATS'09), Karabük, Türkiye, 2009.
- [31] X.-S. Yang and S. Deb, "Cuckoo Search via Lévy Flights," *Nature & Biologically Inspired Computing*, pp. 210 - 214, 9-11 Dec. 2009.
- [32] X.-S. Yang, "Firefly Algorithms for Multimodal Optimization," 7 Mar 2010.
- [33] K. Soltesz, T. Hägglund, and K. J. Åström, "Transfer function parameter identification by modified relay feedback," in *American Control Conference (ACC)*, 2010, 2010, pp. 2164-2169.
- [34] P. Civicioglu and E. Besdok, "A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms," *Artificial Intelligence Review*, vol. 39, pp. 315-346, 2011.
- [35] V. Bobál, P. Chalupa, M. Kubalčík, and P. Dostál, "Identification and Self-tuning Control of Time-delay Systems," 2012.
- [36] İ. Kaya and M. Nalbantoğlu, "Röle geri-beslemeli sistemlerde genetik algoritma ile modelleme," *Dicle Üniversitesi Mühendislik Fakültesi*, vol. 3, pp. 31-39, 2012.
- [37] A. Mirzal, S. Yoshii, and M. Furukawa, "PID parameters optimization by using genetic algorithm," *arXiv preprint arXiv:1204.0885*, 2012.
- [38] S. A. Misal, R.W.Gaikwad, and a. Dhirendra, "Model Identification Using Identification Tool and Estimation of Optimum Control Parameters Using Relay Tuning Method for Bioreacto," 2012.
- [39] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Cuckoo Search algorithm: a metaheuristic approach to solve structural optimization problems," *Springer-Verlag*, vol. 29, p. 18, 2013.

- [40] J. Berner, K. J. Åström, and T. Hägglund, "Towards a new generation of relay autotuners," *IFAC Proceedings Volumes*, vol. 47, pp. 11288-11293, 2014.
- [41] G. A. Hassaan, "Tuning Of A PD-PI Controller Used With A Highly Oscillating Second –Order Process," 2014.
- [42] R. Prokop, J. Korbel, and R. Matusu, "Autotuning for Delay Systems An Algebraic Approach," *15th International Carpathian Control Conference (ICCC)*, 2014.
- [43] K. Karagül, "Guguk Kuşu Algoritması: Bir Plastik Atık Toplama Uygulaması," *15th International Symposium on Econometrics, Operations Research and Statistic, Isparta, Turkey*, vol. 15, pp. 775-784, 22-25 May 2014 2014.
- [44] O. Roeva and T. Slavov, "Firefly algorithm tuning of PID controller for glucose concentration control during E. coli fed-batch cultivation process," *Proceedings of the Federated Conference on Computer Science and Information Systems* pp. 455–462, 2014.
- [45] Z. Batık, S. Kaçar, Ü. Çavuşoğlu, A. Akgül, and A. Sevin, "Kontrolör Tasarımı için GA Kullanıldığı MATLAB ve. NET Tabanlı Bir Windows Uygulaması," *Academic Platform Journal of Engineering and Science*, vol. 2, pp. 24-34, 2014.
- [46] J. Berner, "Automatic tuning of PID controllers based on asymmetric relay feedback," in *Department of Automatic Control*, ed. Sweden: Lund University, 2015.
- [47] J. Berner, T. Hägglund, and Karl J. Åström, "Asymmetric relay autotuning – Practical features for industrial use," *Control Engineering Practice*, vol. 54, pp. 231-245, 2016.
- [48] Q. Jin, L. Qi, B. Jiang, and Q. Wang, "Novel improved cuckoo search for PID controller design," *Transactions of the Institute of Measurement and Control*, vol. 37, pp. 721-731, 2015.
- [49] S. Meena and K. Chitra, "Modified Approach of Firefly Algorithm for Non-Minimum Phase Systems," *Indian Journal of Science and Technology*, vol. 8, 2015.
- [50] S. Nema and P. Kumar Padhy, "Identification and cuckoo PI-PD controller design for stable and unstable processes," *Transactions of the Institute of Measurement and Control*, vol. 37, pp. 708-720, 2015.
- [51] S. Kavirayani and N. Gundavarapu, "Naturally Inspired Firefly Controller For Stabilization Of Double Inverted Pendulum," *Technological Engineering*, vol. 12, 2015.

- [52] L. Luo and L. Lv, "Cuckoo Search Algorithm Based On Local Optimization In The PID Parameter Optimization," presented at the 2nd Workshop on Advanced Research and Technology in Industry Applications (WARTIA 2016), 2016.
- [53] E. Karaarslan and K. Zengin, "Ateş Böceği Algoritması ile Haftalık Ders Programı Hazırlama," EEB 2016 Elektrik-Elektronik ve Bilgisayar Sempozyumu, 11-13 Mayıs 2016.
- [54] A. Gupta and P. K. Padhy, "Modified Firefly Algorithm based controller design for integrating and unstable delay processes," Engineering Science and Technology, an International Journal, vol. 19, pp. 548-558, 2016.
- [55] X.-S. Yang, Nature-inspired optimization algorithms, 1st ed., 2014.
- [56] A. Tozan, F. E. Sevilgen, and O. İnce, "Sensör Yerleştirme Probleminin Parçacık Sürü Optimizasyonu ile Çözümü."
- [57] M. A. Belen, M. Alıcı, A. Çor, and F. Güneş, "Ateşböceği Algoritması ile Mikrodalga Transistör Performansının Karakterizasyonu," ELECO-2014 Elektrik-Elektronik-Bilgisayar ve Biyomedikal Mühendisliği Sempozyumu, pp. 491-494, 2014.
- [58] T. Soderstrom and P. Stoica, "System Identification," 1989.
- [59] İ. Tunç, "Poisson moment fonksiyonu yaklaşımıyla sürekli zaman modeli kestirimi," Yüksek Lisans, Mekatronik Mühendisliği, Bursa Teknik Üniversitesi FBE, 2016.
- [60] K. Soltez, "On Automation of the PID Tuning Procedure," 2012.
- [61] E. Öztemel, Yapay Sinir Ağları: PapatyaYayincilik, Istanbul, 2003.
- [62] K. Kristinsson and G. A. Dumont, "System identification and control using genetic algorithms," IEEE Transactions on Systems, Man, and Cybernetics, vol. 22, pp. 1033-1046, 1992.
- [63] G. Panda, P. M. Pradhan, and B. Majhi, "IIR system identification using cat swarm optimization," Expert Systems with Applications, vol. 38, pp. 12671-12683, 2011.
- [64] B. Luitel and G. K. Venayagamoorthy, "Particle swarm optimization with quantum infusion for system identification," Engineering Applications of Artificial Intelligence, vol. 23, pp. 635-649, 2010.
- [65] H. Tang, S. Xue, and C. Fan, "Differential evolution strategy for structural system identification," Computers & Structures, vol. 86, pp. 2004-2012, 2008.

- [66] M. S. Voss and X. Feng, "ARMA model selection using particle swarm optimization and AIC criteria," *IFAC Proceedings Volumes*, vol. 35, pp. 349-354, 2002.
- [67] S. Chen, T. Mei, M. Luo, and X. Yang, "Identification of nonlinear system based on a new hybrid gradient-based PSO algorithm," in *Information Acquisition, 2007. ICIA'07. International Conference on*, 2007, pp. 265-268.
- [68] M. Shafaati and H. Mojallali, "Modified firefly optimization for IIR system identification," *Journal of Control Engineering and Applied Informatics*, vol. 14, pp. 59-69, 2012.
- [69] A. Gotmare, R. Patidar, and N. V. George, "Nonlinear system identification using a cuckoo search optimized adaptive Hammerstein model," *Expert systems with applications*, vol. 42, pp. 2538-2546, 2015.
- [70] J. Ahmed and Z. Salam, "A Maximum Power Point Tracking (MPPT) for PV system using Cuckoo Search with partial shading capability," *Applied Energy*, vol. 119, pp. 118-130, 2014.
- [71] T. Kumbasar, İ. Eksin, Müjde Güzelkaya, and E. Yeşil, "Big Bang Big Crunch Optimization Method Based Fuzzy Model Inversion," *MICAI 2008: Advances in Artificial Intelligence*, pp. 732-740, 2008.
- [72] S. W. Sung; and J. Lee, *Process identification and pid control*, John Wiley & Sons, 2009
- [73] H. Develi, "Ziegler Nichols Yöntemİ ve Migo Yaklaşımı," *Yüksek Lisans, Kontrol ve Otomasyon, İTÜ, İstanbul Teknik Üniversitesi Fen Bilimleri*, 2004.
- [74] J. Bennar, "Automatic Tuning of PID Controllers based on Asymmetric Relay Feedback," 2015.
- [75] W. S. Levine, "The Control Systems Handbook", Crc Press, 1996
- [76] T. Haggland; and K. J. Astrom, "Revisiting the Ziegler–Nichols rules for PI control," *Asian Journal of Control*, vol. 4, pp. 364-380, 2002.
- [77] K. J. Åström and T. Hägglund, "Revisiting the Ziegler–Nichols step response method for PID control," *Journal of Process Control*, vol. 14, pp. 635-650, 2004.
- [78] T. Yücelen. "Uzun ölü zamanlı sistemler için Smith Öngörücüsü Yöntemi ile PI-P Kontrolör Tasarımı".
- [79] R. Mamat and P. Fleming, "Method for on-line identification of a first order plus dead-time process model," *Electronics letters*, vol. 31, pp. 1297-1298, 1995.

EKLER

EK 1: PSO Matlab Kodu

```
function PSO()
Ub=[10 10];
Lb=[0 0];
boyut=2;
swarm_size =genelSuruBoyutu;
maxIter = maximumIterasyon;
inertia = 0.9;
correction_factor1 = 2;
correction_factor2 = 2;
    swarm=zeros(swarm_size,4,boyut);
swarm(1:swarm_size,1,1:boyut) =Initialize_PSO_denetleyici(swarm_size,Lb,Ub);
swarm(:,2,:) = 0;
swarm(:,4,1) = 10e5;
    [swarm]=SINIRLANDIRMA(swarm,Lb,Ub);
fval = PSO_Olcme(swarm,KontrolorTipi,PIndex,sure,pay,payda,L);
[hata konum]=min(fval);
iterasyon=1;
hata;
Hatalar(iterasyon)=hata;
Eniyiler=[swarm(konum, 1, :)];
while iterasyon<maxIter
iterasyon=iterasyon+1;
for i=1:boyut
swarm(:, 1, i) = swarm(:, 1, i) + swarm(:, 2, i)/1.3
```

```

end
[swarm]=SINIRLANDIRMA(swarm,Lb,Ub);
fval = PSO_Olcme(swarm,);
    for ii = 1:swarm_size
        if fval(ii) < swarm(ii,4,1)
            swarm(ii, 3, 1:boyut) = swarm(ii, 1, 1:boyut);
            swarm(ii, 4, 1) = fval(ii);
        end
    end
    end
    [~, gbest] = min(swarm(:, 4, 1));
    for i=1:boyut
        swarm(:, 2, i) = inertia*(rand(swarm_size,1).*swarm(:, 2, i)) +
correction_factor1*(rand(swarm_size,1).*(swarm(:, 3, i) ...
- swarm(:, 1, i))) + correction_factor2*(rand(swarm_size,1).*(swarm(gbest, 3, i)
- swarm(:, 1, i)));
    end
    [hata konum]=min(fval);
end
end
function donus=PSO_Olcme(swarm)
[sat sut]=size(swarm(:,1,:));
for i=1:sat
for j=1:sut
    u(j)=swarm(i,1,j);
end
donus(i)=sum(u);
end
end
function [swarm]=SINIRLANDIRMA(swarm,Lb,Ub)
[sat sut]=size(swarm(:,1,:));
[sat2 sut2 r]=size(swarm(:,1,:));
Lb;

```

```

for i=1:sat
for ii=1:r
    if swarm(i,1,ii)>Ub(ii)
        swarm(i,1,ii)=Ub(ii);
    end
    if swarm(i,1,ii)<Lb(ii)
        swarm(i,1,ii)=Lb(ii);
    end
end
end
end
end

```

EK 2: CS Matlab Kodu

```

function CS()
iterasyon=1;
n=40;
maxIter=100;

pa=CS_pa;
Ub=[10 10];
Lb=[0 0];

Ub=Ub(1:boyut);
Lb=Lb(1:boyut);
nest=zeros(n,boyut);
for i=1:boyut
    nest(:,i)= rand(n,1).*(Ub(i));
end
for j=1:size(nest,1)
    s=nest(j,:);
    nest(j,:)=simplebounds(s,Lb,Ub);
end

```

```

end
fitness=10e15*ones(n,1);
[fmin,bestnest,nest,fitness]=get_best_nest(nest,nest,fitness);
best_eski=bestnest;
[sat sut]=size(bestnest);
a=fobj(bestnest);
beta=1.5;
iterasyon=1;
while iterasyon<maxIter
    new_nest=get_cuckoos(nest,bestnest,Lb,Ub,beta);
    [fnew,best,nest,fitness]=get_best_nest(nest,new_nest,fitness);
    new_nest=emptyt_nests(nest,Lb,Ub,pa);
    [fnew,best,nest,fitness]=get_best_nest(nest,new_nest,fitness);
    if fnew<fmin
        fmin=fnew;
        bestnest=best;
    end
    iterasyon=iterasyon+1;
end
end
function donus=fobj(u)
donus=sum(u);
end
function [fmin,bestnest,nest,fitness]=get_best_nest(nest,newnest,fitness)
for j=1:size(nest,1)
    fnew=fobj(newnest(j,:));
    if fnew<=fitness(j)
        fitness(j)=fnew;
        nest(j,:)=newnest(j,:);
    end
end
end
[fmin,k]=min(fitness);

```

```

bestnest=nest(k,:);
end
function new_nest=empty_nests(nest,Lb,Ub,pa)
global iterasyon
n=size(nest,1);
K=rand(size(nest))>pa;
stepsize=rand*(nest(randperm(n),:)-nest(randperm(n),:));
new_nest=nest+0.025*stepsize.*K;
for j=1:size(new_nest,1)
    s=new_nest(j,:);
    new_nest(j,:)=simplebounds(s,Lb,Ub);
end
end
function [fmin,bestnest,nest,fitness]=get_best_nest(nest,newnest,fitness)
for j=1:size(nest,1)
    fnew=fobj_Denetleyici(newnest(j,:));
    if fnew<=fitness(j)
        fitness(j)=fnew;
        nest(j,:)=newnest(j,:);
    end
end
end
[fmin,k]=min(fitness);
bestnest=nest(k,:);
end
function nest=get_cuckoos(nest,best,Lb,Ub,beta)
    sigma=(gamma(1+beta)*sin(pi*beta/2)/(gamma((1+beta)/2)*beta*2^((beta-
1)/2)))^(1/beta);
    n=size(nest,1);
    for j=1:n
        s=nest(j,:);
        u=randn(size(s))*sigma;
        v=randn(size(s));

```



```

    step=u./abs(v).^(1/beta);
    stepsize=0.025*step.*(s-best);
    s=s+stepsize.*randn(size(s));
    nest(j,:)=simplebounds(s,Lb,Ub);
    end
end
function donus=fobj(u)
donus=sum(u);
end

```

EK 3: FA Matlab Kodu

```

function FA()
Ub=[10 ;10]
Lb=[0; 0; ];
boyut;
iterasyon=1;
d=boyut;
u0=Lb+(Ub-Lb).*rand(1,d);
iterasyon=0;
Hatalar=[];
Eniyiler=[];
nbest=0;
fbest=0;
NumEval=0;n=40;
MaxIteration=100;
alpha=1;
betamin=0.8;
gamma=0.2;
d=2;
zn=ones(n,1)*inf;
[ns,Lightn]=init_ffa(n,d,Lb,Ub,u0);

```

```

fbest=inf;
ns2=ns;
Lightbest=cost_2(ns(1,:));
nbest=ns(1,:);
while iterasyon<=MaxIteration
    iterasyon=iterasyon+1;
    for i=1:n
        zn(i)=cost_2(ns2(i,:));
        Lightn(i)=zn(i);
    end
    [Lightn,Index]=sort(zn);
    ns_tmp=ns;
    for i=1:n
        if cost_2(ns2(i,:))<cost_2(ns(i,:))
            ns(i,:)=ns2(i,:);
        end
    end
    for i=1:n
        if cost_2(ns(i,:))<Lightbest
            Lightbest=cost_2(ns(i,:));
            nbest=ns(i,:);
            fbest=Lightbest;
        end
        Lightn(i)=cost_2(ns(i,:));
    end
    nso=ns;
    Lighto=Lightn;
[ns2]=ffa_move(n,d,ns,Lightn,nso,Lighto,nbest,Lightbest,alpha,betamin,gamma,Lb,
Ub);
end
end
function [ns,Lightn]=init_ffa(n,d,Lb,Ub,u0)

```

```

if length(Lb)>0
    for i=1:n
        ns(i,:)=Lb+(Ub-Lb).*rand(1,d);
    end
else
    ns(i,:)=u0+randn(1,d);
end
Lightn=ones(n,1)*10^100;
end
function
[ns]=ffa_move(n,d,ns,Lightn,nso,Lightn,nbest,Lightbest,alpha,betamin,gamma,Lb,U
b)
scale=abs(Ub-Lb);
for i=1:n
    for j=1:n
        r=sqrt(sum(ns(i,:)-ns(j,:)).^2);
        if Lightn(i)>Lightn(j)
            beta0=1;
            beta=(beta0-betamin)*exp(-gamma*r.^2)+betamin;
            tmpf=alpha.*(rand(1,d)-0.5).*scale;
            ns(i,:)=ns(i,:).*(1-beta)+nso(j,:).*beta+tmpf;
        end
    end
end
[ns]=findlimits(n,ns,Lb,Ub);
end
function donus= cost_2 (u)
donus=sum(u);
end

```

ÖZGEÇMİŞ

Murat Erhan Çimen, 04.06.1992'da İstanbul Bakırköy'de doğdu. İlk, ortaokulu İstanbul'da tamamladı. 2006 yılında Bağcılar Teknik Lisesi'ne başlayıp 2010 yılında okul birincisi olarak mezun oldu. Aynı sene Sakarya Üniversitesi Mekatronik Mühendisliği Bölümü'nü kazandı. 2014 yılında bölüm birincisi, fakülte birincisi ve üniversite ikincisi olarak mezun oldu. Ardında Elektrik-Elektronik Mühendisliği'nden de 2015 yılında mezun oldu. 2015 yılında Ömer Halisdemir Üniversitesi'nde araştırma görevlisi olarak 6 ay çalıştıktan sonra istediği danışman ve alanla ilgili çalışmasına müsaade edilmediğinden ötürü görevinden istifa etti. Aynı yıl Sakarya Üniversitesi Elektrik Elektronik Mühendisliği'nde yine araştırma görevlisi olarak atandı. Elektronik, kontrol ve yazılım alanlarına ilgi duyan Murat Erhan Sakarya ve İTÜ'de yüksek lisans eğitimleri devam etmektedir. Ayrıca Sakarya Üniversitesi Elektrik Elektronik Mühendisliği'nde araştırma görevlisi olarak görevine halen devam etmektedir.