T.C.
AYDIN ADNAN MENDERES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
MATEMATİK ANABİLİM DALI
2018-YL-030

İKİ NOKTA SINIR DEĞER PROBLEMLERİNİN PARÇACIK SÜRÜ OPTİMİZASYONU VARYASYONLARI İLE NÜMERİK ÇÖZÜMLERİ

Gülsüm İŞMAN

Tez Danışmanı:
Dr. Öğr. Üyesi Korhan GÜNEL

AYDIN

T.C.
AYDIN ADNAN MENDERES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRLÜĞÜNE
AYDIN

Matematik Anabilim Dalı Yüksek Lisans Programı öğrencisi Gülsüm İşman tarafından hazırlanan İki Nokta Sınır Değer Problemlerinin Parçacık Sürü Optimizasyonu varyasyonları ile Nümerik Çözümleri başlıklı tez, 10.07.2018 tarihinde yapılan savunma sonucunda aşağıda isimleri bulunan jüri üyelerince kabul edilmiştir.

	Ünvanı, Adı Soyadı	Kurumu	İmzası
Başkan :	Dr. Öğr. Üyesi Korhan GÜNEL	ADÜ	
Üye :	Dr. Öğr. Üyesi Refet POLAT	Yaşar Üniversitesi	
Üye :	Dr. Öğr. Üyesi Rıfat AŞLIYAN	ADÜ	

Jüri üyeleri tarafından kabul edilen bu Yüksek Lisans tezi, Enstitü Yönetim Kurulunun ……Sayılı kararıyla …….. tarihinde onaylanmıştır.

Prof. Dr. Aydın ÜNAY

Enstitü Müdürü

T.C.
AYDIN ADNAN MENDERES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRLÜĞÜNE
AYDIN

Bu tezde sunulan tüm bilgi ve sonuçların, bilimsel yöntemlerle yürütülen gerçek deney ve gözlemler çerçevesinde tarafımdan elde edildiğini, çalışmada bana ait olmayan tüm veri, düşünce, sonuç ve bilgilere bilimsel etik kuralların gereği olarak eksiksiz şekilde uygun atıf yaptığımı ve kaynak göstererek belirttiğimi beyan ederim.

10/07/2018

Gülsüm İŞMAN

ÖZET

İKİ NOKTA SINIR DEĞER PROBLEMLERİNİN PARÇACIK SÜRÜ OPTİMİZASYONU VARYASYONLARI İLE NÜMERİK ÇÖZÜMLERİ

Gülsüm İŞMAN

Yüksek Lisans Tezi, Matematik
Tez Danışmanı: Doktor Öğretim Üyesi Korhan GÜNEL
2018, 53 sayfa

Bu çalışmada, ikinci mertebeden lineer veya lineer olmayan adi diferansiyel denklemler için iki nokta sınır koşulları ile verilen sınır değer problemlerinin nümerik çözümlerinin Parçacık Sürü Optimizasyonu ve varyasyonları ile elde edilmesi amaçlanmıştır.

Anahtar Kelimeler: Parçacık Sürü Optimizasyonu, İki Koşullu Sınır Değer Problemi, Adi Diferansiyel Denklem

ABSTRACT

NUMERICAL SOLUTIONS OF TWO POINT BOUNDARY PROBLEMS VIA VARIATIONS OF PARTICAL SWARM OPTIMIZATION

Gülsüm İŞMAN

M.Sc. Thesis, Department of Mathematics
Supervisor: Korhan GÜNEL, Ph.D.
2018, 53 pages

In this work, the numerical solutions of boundary value problems given by two-point boundary conditions for linear or nonlinear second order ordinary differantial equations are aimed to be obtained by some variants of Particle Swarm Optimization.

Key Words: Particle Swarm Optimization, Two-Point Boundary Value Problem, Ordinary Differantial Equation.

ÖNSÖZ

Yüksek lisans eğitimim boyunca yardım, bilgi ve tecrübeleri ile bana sürekli destek olan Doktor Öğretim Üyesi Korhan GÜNEL hocama, maddi manevi destekleriyle hep yanımda olan aileme en içten teşekkürlerimi sunarım.

Gülsüm İŞMAN

İÇİNDEKİLER

KABUL VE ONAY SAYFASI iii
BİLİMSEL ETİK BİLDİRİM SAYFASI v
ÖZET vii
ABSTRACT ix
ÖNSÖZ xi
SİMGELER VE KISALTMALAR DİZİNİ xv
ŞEKİLLER DİZİNİ xvii
ÇİZELGELER DİZİNİ xxiii
1. GİRİŞ 1
2. PARÇACIK SÜRÜ OPTİMİZASYONU 3
2.1. PSO Algoritmasının Kaba Kodu 5
2.2. PSO'nun Parametreleri 6
3. PARÇACIK SÜRÜ OPTİMİZASYONU TÜRLERİ 9
3.1. Temel Parçacık Sürü Optimizasyonu (BPSO) 9
3.2. Azalan Ağırlıklı Parçacık Sürü Optimizasyonu (DWPSO) 9
3.3. Daralma Katsayılı Parçacık Sürü Optimizasyonu (PSO_C) 10
3.4. Zamana Göre Değişen Hız Katsayılı Kendini Örgütleyen Hiyerarşik Parçacık Sürü Optimizasyonu (STVAC_PSO) 10
3.5. Zamana Göre Değişen Parçacık Sürü Optimizasyonu (TVPSO) 11
3.6. Zamana Bağlı Olarak Değişen Atalet Katsayılı Parçacık Sürü Optimizasyonu (TVIW_PSO) 11
3.7. Zamana Bağlı İvmelendirici Katsayılı Parçacık Sürü Optimizasyonu (TVAC-PSO) 12
4. DIRICHLET SINIR KOŞULUNA SAHİP ADİ DİFERANSİYEL DENKLEMLERİN NÜMERİK ÇÖZÜMLERİ 13
5. DENEYSEL ÇALIŞMALAR 17

6. TARTIŞMA VE SONUÇ..... 49
KAYNAKLAR..... 51
ÖZGEÇMİŞ..... 53

SİMGELER VE KISALTMALAR DİZİNİ

ABC	: Yapay Arı Kolonisi
ACO	: Karınca Kolonisi Algoritması
BA	: Yarasa Algoritması
BPSO	: Temel Parçacık Sürü Optimizasyonu
COA	: Guguk Kuşu Algoritması
DWPSO	: Azalan Ağırlıklı Parçacık Sürü Optimizasyonu
FO	: Ateşböceği Algoritması
PSO	: Parçacık Sürü Optimizasyonu
PSO_C	: Daralma Katsayılı Parçacık Sürü Optimizasyonu
STVAC_PSO	: Zamana Göre Değişen Hız Katsayılı Kendini Örgütleyen Hiyerarşik Parçacık Sürü Optimizasyonu
TVAC_PSO	: Zamana Bağlı Olarak Değişen İvmelendirici Katsayılı Parçacık Sürü Optimizasyonu
TVIW_PSO	: Zamana Bağlı Olarak Değişen Atalet Katsayılı Parçacık Sürü Optimizasyonu
TVPSO	: Zamana Göre Değişen Atalet Katsayılı Parçacık Sürü Optimizasyonu

ŞEKİLLER DİZİNİ

Şekil 2.1. PSO Algoritmasının işleyiş şeması 4

Şekil 5.1. (a) Örnek 5.1 için BPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) BPSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 17

Şekil 5.2. (a) Örnek 5.1 için CLPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) CLPSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 18

Şekil 5.3. (a) Örnek 5.1 için DWPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) DWPSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 18

Şekil 5.4. (a) Örnek 5.1 için STVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) STVAC_PSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 19

Şekil 5.5. (a) Örnek 5.1 için PSO_C yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) PSO_C yönteminin yapay sinir ağı çözümü $h = 0.1$ 19

Şekil 5.6. (a) Örnek 5.1 için TVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVAC_PSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 20

Şekil 5.7. (a) Örnek 5.1 için TVIW_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVIW_PSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 20

Şekil 5.8. (a) Örnek 5.1 için TVPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVPSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 21

Şekil 5.9. (a) Örnek 5.1 için BPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) BPSO yönteminin yapay sinir ağı çözümü $h = 0.01$ 22

Şekil 5.10. (a) Örnek 5.1 için CLPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) CLPSO yönteminin yapay sinir ağı çözümü $h = 0.01$ 23

Şekil 5.11. (a) Örnek 5.1 için DWPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) DWPSO yönteminin yapay sinir ağı çözümü $h = 0.01$ 23

Şekil 5.12. (a) Örnek 5.1 için STVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) STVAC_PSO yönteminin yapay sinir ağı çözümü $h = 0.01$ 24

Şekil 5.13. (a) Örnek 5.1 için PSO_C yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) PSO_C yönteminin yapay sinir ağı çözümü $h = 0.01$ 24

Şekil 5.14. (a) Örnek 5.1 için TVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVAC_PSO yönteminin yapay sinir ağı çözümü $h = 0.01$ 25

Şekil 5.15. (a) Örnek 5.1 için TVIW_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVIW_PSO yönteminin yapay sinir ağı çözümü $h = 0.01$ 25

Şekil 5.16. (a) Örnek 5.1 için TVPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVPSO yönteminin yapay sinir ağı çözümü $h = 0.01$ 26

Şekil 5.17. (a) Örnek 5.2 için BPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) BPSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 28

Şekil 5.18. (a) Örnek 5.2 için CLPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) CLPSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 28

Şekil 5.19. (a) Örnek 5.2 için DWPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) DWPSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 29

Şekil 5.20. (a) Örnek 5.2 için STVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) STVAC_PSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 29

Şekil 5.21. (a) Örnek 5.2 için PSO_C yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) PSO_C yönteminin yapay sinir ağı çözümü $h = 0.1$ 30

Şekil 5.22. (a) Örnek 5.2 için TVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVAC_PSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 30

Şekil 5.23. (a) Örnek 5.2 için TVIW_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVIW_PSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 31

Şekil 5.24. (a) Örnek 5.2 için TVPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVPSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 31

Şekil 5.25. (a) Örnek 5.2 için BPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) BPSO yönteminin yapay sinir ağı çözümü $h = 0.01$ 33

Şekil 5.26. (a) Örnek 5.2 için CLPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) CLPSO yönteminin yapay sinir ağı çözümü $h = 0.01$ 33

Şekil 5.27. (a) Örnek 5.2 için DWPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) DWPSO yönteminin yapay sinir ağı çözümü $h = 0.01$ 34

Şekil 5.28. (a) Örnek 5.2 için STVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) STVAC_PSO yönteminin yapay sinir ağı çözümü $h = 0.01$ 34

Şekil 5.29. (a) Örnek 5.2 için PSO_C yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) PSO_C yönteminin yapay sinir ağı çözümü $h = 0.01$ 35

Şekil 5.30. (a) Örnek 5.2 için TVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVAC_PSO yönteminin yapay sinir ağı çözümü $h = 0.01$ 35

Şekil 5.31. (a) Örnek 5.2 için TVIW_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVIW_PSO yönteminin yapay sinir ağı çözümü $h = 0.01$ 36

Şekil 5.32. (a) Örnek 5.2 için TVPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVPSO yönteminin yapay sinir ağı çözümü $h = 0.01$ 36

Şekil 5.33. (a) Örnek 5.3 için BPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) BPSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 38

Şekil 5.34. (a) Örnek 5.3 için CLPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) CLPSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 39

Şekil 5.35. (a) Örnek 5.3 için DWPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) DWPSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 39

Şekil 5.36. (a) Örnek 5.3 için STVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) STVAC_PSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 40

Şekil 5.37. (a) Örnek 5.3 için PSO_C yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) PSO_C yönteminin yapay sinir ağı çözümü $h = 0.1$ 40

Şekil 5.38. (a) Örnek 5.3 için TVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVAC_PSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 41

Şekil 5.39. (a) Örnek 5.3 için TVIW_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVIW_PSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 41

Şekil 5.40. (a) Örnek 5.3 için TVPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVPSO yönteminin yapay sinir ağı çözümü $h = 0.1$ 42

Şekil 5.41. (a) Örnek 5.3 için BPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) BPSO yönteminin yapay sinir ağı çözümü $h = 0.01$.....43

Şekil 5.42. (a) Örnek 5.3 için CLPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) CLPSO yönteminin yapay sinir ağı çözümü $h = 0.01$.....44

Şekil 5.43. (a) Örnek 5.3 için DWPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) DWPSO yönteminin yapay sinir ağı çözümü $h = 0.01$.....44

Şekil 5.44. (a) Örnek 5.3 için STVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) STVAC_PSO yönteminin yapay sinir ağı çözümü $h = 0.01$ 45

Şekil 5.45. (a) Örnek 5.3 için PSO_C yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) PSO_C yönteminin yapay sinir ağı çözümü $h = 0.01$.....45

Şekil 5.46. (a) Örnek 5.3 için TVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVAC_PSO yönteminin yapay sinir ağı çözümü $h = 0.01$.....46

Şekil 5.47. (a) Örnek 5.3 için TVIW_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVIW_PSO yönteminin yapay sinir ağı çözümü $h = 0.01$ 46

Şekil 5.48. (a) Örnek 5.3 için TVPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVPSO yönteminin yapay sinir ağı çözümü $h = 0.01$.....47

ÇİZELGELER DİZİNİ

Çizelge 5.1. Örnek 5.1 için $h = 0.1$ adım aralığıyla oluşturulan maliyet fonksiyonu için eğitim kümesinin hata miktarları 21

Çizelge 5.2. Örnek 5.1 için $h = 0.1$ adım aralığıyla oluşturulan çözüm fonksiyonu için test kümesinin hata miktarları 22

Çizelge 5.3. Örnek 5.1 için $h = 0.01$ adım aralığıyla oluşturulan çözüm fonksiyonu için eğitim kümesinin hata miktarları 26

Çizelge 5.4. Örnek 5.1 için $h = 0.01$ adım aralığıyla oluşturulan çözüm fonksiyonu için test kümesinin hata miktarları 27

Çizelge 5.5. Örnek 5.2 için $h = 0.1$ adım aralığıyla oluşturulan çözüm fonksiyonu için eğitim kümesinin hata miktarları 32

Çizelge 5.6. Örnek 5.2 için $h = 0.1$ adım aralığıyla oluşturulan çözüm fonksiyonu için test kümesinin hata miktarları 32

Çizelge 5.7. Örnek 5.2 için $h = 0.01$ adım aralığıyla oluşturulan çözüm fonksiyonu için eğitim kümesinin hata miktarları 37

Çizelge 5.8. Örnek 5.2 için $h = 0.01$ adım aralığıyla oluşturulan çözüm fonksiyonu için test kümesinin hata miktarları 37

Çizelge 5.9 Örnek 5.3 için $h = 0.1$ adım aralığıyla oluşturulan çözüm fonksiyonu için eğitim kümesinin hata miktarları 42

Çizelge 5.10. Örnek 5.3 için $h = 0.1$ adım aralığıyla oluşturulan çözüm fonksiyonu için test kümesinin hata miktarları 43

Çizelge 5.11. Örnek 5.3 için $h = 0.01$ adım aralığıyla oluşturulan çözüm fonksiyonu için eğitim kümesinin hata miktarları 47

Çizelge 5.12. Örnek 5.3 için $h = 0.01$adım aralığıyla oluşturulan çözüm fonksiyonu için test kümesinin hata miktarları 48

1. GİRİŞ

Günümüzde optimizasyon hemen hemen her alanda kullanılır. Kelime anlamı en iyileme olan optimizasyon, keyfi parametrelere bağlı olan amaç fonksiyonunu birtakım kısıtlayıcı şartlar altında en optimal (en düşük veya en yüksek) yapan değerlerin bulunması işlemidir. Amaç fonksiyonu; minimize (veya maksimize) edilmesi istenen bir diğer deyişle optimum noktaları aranan fonksiyondur. Kısıtlar ise fonksiyonu oluşturan değişkenlerin uyması gereken sınır koşulları olduğunu işaret eder.

Optimizasyon problemlerinin çözümünde klasik yöntemler yetersiz kaldığı için 1970'li yıllarda sezgisel optimizasyon teknikleri çalışılmaya başlanmıştır. Biyolojik sistemlerden etkilenilerek geliştirilmiş birçok algoritma mevcuttur. Zor problemlerin çözümünün geleneksel yöntemlerle çözülmesinin güçlüğü karşısında birçok araştırmacı bu alana yönelmiş olup, çoğu problemde başarılı sonuçlar elde edilmiştir.

Sezgisel optimizasyon yöntemlerinden Genetik Algoritmalar (GA) bir ilk konumundadır. Evrim teorisinden esinlenerek ortaya çıkan bu yöntemi 1976 yılında Holland geliştirmiştir [1]. Genetik Algoritmalar uzay mühendisliği, astronomi, jeofizik, malzeme mühendisliği gibi daha nice alanda uygulanmıştır [2].

Popülasyon tabanlı sezgisel algoritmalardan bir diğeri olan Karınca Kolonisi Optimizasyonu (ACO), 1990'lı yılların başında M. Dorigo ve çalışma arkadaşları tarafından geliştirilmiştir [3]. ACO algoritması, karıncaların sürü halindeki davranışlarını baz alarak, ayrık optimizasyon problemlerinin çözümü için geliştirilmiş bir algoritmadır. Sürü zekasına dayalı olarak geliştirilen bir başka algoritma ise Yapay Arı Kolonisi (ABC) optimizasyon algoritmasıdır. 2005 yılında Karaboğa tarafından tanıtılmış olan bu optimizasyon yönteminde bal arılarının yiyecek arama davranışlarından esinlenilmiştir [4]. Bu yöntem özellikle çizelgeleme alanında birçok problemi başarıyla çözmüştür.

Sürü zekasına dayalı pek çok sezgisel yaklaşım önerilmiştir. Guguk Kuşu Algoritması (COA) guguk kuşlarının kuluçka paraziti türlerini temel alarak 2001 yılında geliştirilmiştir [6]. Ateşböceği Algoritması (FO) 2009 yılında ve Yarasa

Algoritması (BA) 2010 yılında geliştirilmiş sezgisel optimizasyon algoritmalarına birer örnektir [7].

Bu çalışmada, x bağımsız bir değişken olmak üzere, $x \in [a, b]$ için $y''(x) = \big(x, y(x), y'(x)\big)$ formundaki ikinci mertebeden lineer veya lineer olmayan adi diferansiyel denklemler için $y(a) = A$, $y(b) = B$ koşulları altındaki sınır değer problemlerinin nümerik çözümlerinin Parçacık Sürü Optimizasyonu yöntemiyle elde edilmesi amaçlanmaktadır.

Bu tezin ikinci bölümünde, Parçacık Sürü Optimizasyonu hakkında genel bir bilgi verilmiş olup üçüncü bölümde tezde kullanılan Parçacık Sürü Optimizasyonu türleri verilmiştir. Bölüm dörtte Dirichlet sınır değer koşuluna sahip adi diferansiyel denklemlerin ileri beslemeli yapay sinir ağları ile sayısal çözümlerinin nasıl elde edildiği hakkında bilgiler verilmiştir. Önerilen yaklaşım beşinci bölümde deneysel çalışmalar yapılarak örnekler üzerinde test edilmiştir. Son bölümünde ise elde edilen sonuçlar karşılaştırılarak öneriler sunulmuştur.

2. PARÇACIK SÜRÜ OPTİMİZASYONU

Parçacık Sürü Optimizasyonu (PSO) 1995 yılında Dr. Eberhart ve Dr. Kennedy tarafından geliştirilmiş popülasyon temelli sezgisel bir optimizasyon tekniğidir. PSO'nun temelinin sosyolojik esinlenmeli olduğu söylenebilir. Çünkü algoritmanın orijinal fikri, kuşların sürü halinde toplanmasıyla ilişkilendirilmiş sosyolojik davranışlara dayanır. Kuş, balık ve hayvan sürülerinin bir "bilgi paylaşma" yaklaşımı uygulayarak çevrelerine adapte olabilme, zengin yiyecek kaynağı bulabilme ve avcılardan kaçabilme yeteneklerinden yararlanılmıştır.

PSO, Genetik Algoritmalar gibi evrimsel hesaplama teknikleri ile birçok benzerlik gösterir. Sistem rastgele çözümlerden oluşan bir popülasyon ile başlatılır ve en iyi çözüm için nesilleri güncelleyerek arama yapar. Buna karşın Genetik Algoritmaların tersine, PSO'da çaprazlama (crossover) ve mutasyon gibi evrimsel operatörler yoktur. PSO'da parçacık (particle) denilen potansiyel çözümler, mevcut en iyi çözümleri takip ederek problem uzayında gezinirler.

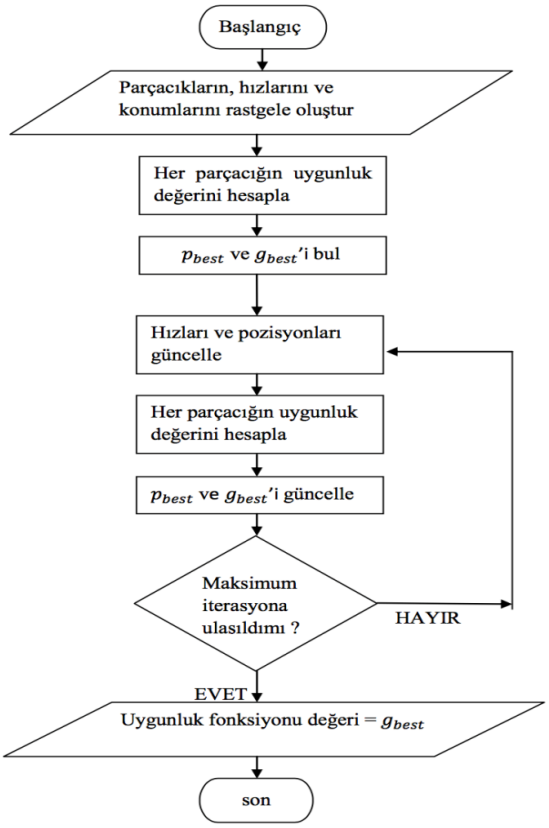
PSO, optimum ya da optimuma yakın çözüm bulmak için önce her biri aday çözümü sunan bireyler oluşturur. Bu bireylerin oluşturulması gelişigüzel, düzenli ya da her ikisi şeklinde yapılabilir. Bireylerin bir araya gelmesinden çözüm için gerçekleştirilen popülasyon (sürü) meydana gelir. Bu model, bireyler arasındaki bilginin paylaşımını esas alır. Her bir parçacık kendi pozisyonunu sürüdeki en iyi parçacığa doğru ayarlarken, bir önceki tecrübesinden yararlanır.

Günümüzde PSO, pek çok alanda başarılı bir şekilde uygulanmaktadır. Bunlardan bazıları; fonksiyon optimizasyonu, çizelgeleme, yapay sinir ağlarının eğitimi, bulanık sistem kontrolü, görüntü işleme ve Genetik Algoritmaların uygulanabildiği diğer alanlardır.

Daha önce de açıklandığı gibi, PSO kuş veya balık sürülerinin davranışlarını benzetir. Bir alanda rastgele yiyecek arayan bir kuş grubunun olduğunu varsayalım. Kuşların hiçbiri yiyeceğin nerede olduğunu bilmesin. Fakat her iterasyon sonunda yiyeceğin ne kadar uzaklıkta olduğunu bilsinler. Bu durumda en iyi strateji nedir? En etkili olanı yiyeceğe yakın olan kuşu takip etmektir. İşte PSO bu stratejiye göre çalışır ve optimizasyon problemlerini çözmek için kullanılır. Bu modelde her bir çözüm arama uzayındaki bir kuştur ve bunlar birer "parçacık" olarak isimlendirilir.

Tüm parçacıkların, optimize edilecek uygunluk fonksiyonu tarafından değerlendirilen bir uygunluk değeri ve uçuşlarını yönlendiren hız bilgileri vardır. Parçacıklar problem uzayında mevcut optimum parçacıkları takip ederek uçarlar.

PSO bir grup rastgele üretilmiş çözüm ile (parçacık ile) başlatılır ve nesiller güncellenerek en uygun değerler araştırılır. Her iterasyonda, her bir parçacık en iyi uygunluk değeridir. Ayrıca bu değer daha sonra kullanılmak üzere hafızada tutulur ve $pbest$, yani parçacığın en iyi değeri, olarak isimlendirilir. Diğer en iyi değer ise popülasyondaki herhangi bir parçacık tarafından o ana kadar elde edilmiş en iyi uygunluk değerine sahip çözümdür. Bu değer popülasyon için en iyi değerdir ve $gbest$ olarak isimlendirilir.



Şekil.1. PSO Algoritmasının işleyiş şeması

Şimdi, D boyutlu bir arama uzayında N tane parçacık verilsin. Matrise göre i. parçacık

$\boldsymbol{x}_i = [x_{i1}, x_{i2}, \dots x_{iD}]$ ve bu parçacığın bilinen en iyi konumu;

$\boldsymbol{pbest}_i = [p_{i1}, p_{i2}, \dots, p_{iD}]$ olsun.

$\boldsymbol{gbest} = [p_1, p_2, \dots, p_D]$ ise tüm parçacıklar arasında elde edilen en iyi konum olsun.

i. parçacığın her t iterasyondaki değişim miktarını gösteren hız vektörü;

$\boldsymbol{v}_i(t) = [v_{i1}, v_{i2}, \dots, v_{iD}]$

olarak ifade edilir. Bu iki en iyi değer bulunduktan sonra parçacık, hızını ve konumunu sırasıyla aşağıdaki (1) ve (2) denklemlerine göre günceller.

$$\boldsymbol{v}_i(t+1) = \boldsymbol{v}_i(t) + c_1 r_1\big(\mathbf{pbest}_i(t) - \boldsymbol{x}_i(t)\big) + c_2 r_2(\mathbf{gbest}(t) - \boldsymbol{x}_i(t)) \quad (2.1)$$

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+1) \quad (2.2)$$

Burada r katsayısı (0,1) arasında üretilen rastgele bir değeri, i parçacık numarasını, t ise iterasyon sayısını gösterir. c_1 ve c_2 öğrenme faktörleridir ve hızlanma katsayıları olarak da adlandırılır. Bunlar parçacıkları **pbest** ve **gbest** konumlarına doğru yönlendiren sabitlerdir. c_1 parçacığın kendi tecrübesine göre, c_2 ise sürüdeki diğer parçacıkların tecrübesine göre hareketi yönlendirir. Düşük değerler seçilmesi parçacıkların hedef bölgeye doğru çekilmeden önce, bu bölgeden uzak yerlerde dolaşmalarına imkan verir. Ancak hedefe ulaşma süresi uzayabilir. Diğer yandan yüksek değerler seçilmesi hedefe ulaşmayı hızlandırırken, beklenmedik hareketlerin oluşmasına ve hedef bölgenin es geçilmesine sebep olabilir.

2.1. PSO Algoritmasının Kaba Kodu

for her parçacık için

parçacığı başlangıç konumuna getir

end

do

for her parçacık için

uygunluk değerini hesapla

eğer uygunluk değeri **pbest**'ten daha iyi ise

şimdiki değeri yeni **pbest** olarak ayarla

end

tüm parçacıkların bulduğu **pbest** değerlerinin en iyisini, tüm parçacıkların **gbest**'i olarak ayarla

for her parçacık için

denklem(**1**)'e göre parçacık hızını ayarla

denklem(**2**)'ye göre parçacık konumunu güncelle

end

while maksimum iterasyon sayısına veya minimum hata koşulu sağlanana kadar devam et

2.2. PSO'nun Parametreleri

PSO'nun avantajlarından birisi reel sayılarla çalışıyor olmasıdır. Genetik Algoritmalardaki gibi hesaplama yapabilmek için ikili kodlamadan dönüştürme yapılması ya da bazı özel kullanılması zorunlu operatörlere ihtiyaç duyulmaz. Örneğin;

$f(x) = x_i^2 + x_2^2 + x_3^2$

fonksiyonu için çözüm bulmayı deneyelim. Burada 3 bilinmeyen olduğundan $D = 3$ boyutludur ve parçacık (x_1, x_2, x_3) şeklinde ayarlanır ve fonksiyonumuz da uygunluk fonksiyonu olarak kullanılır. Optimumu bulmak için belli bir prosedür uygulanır. Sonlanma kriteri olarak maksimum iterasyon veya minimum hata koşulu

sağlanması kullanılır. Görüldüğü gibi PSO'da ihtiyaç duyulan çok az sayıda parametre vardır. Bu parametrelerin listesi şu şekildedir:

- **Parçacık Sayısı:** Genellikle 20 ile 40 arasında alınır. Aslında çoğu problem için sayıyı 10 almak iyi çözümler elde etmek için yeterlidir. Bazı zor veya özel problemler için 100 veya 200 parçacık kullanılması gerekebilir.
- **Parçacık Boyutu:** Optimize edilecek probleme göre değişmektedir.
- **Parçacık Aralığı:** Optimize edilecek probleme göre değişmekle birlikte farklı boyutlarda ve aralıklarda parçacıklar tanımlanabilir.
- **V_{max}:**Bir iterasyonda bir parçacıkta meydana gelecek maksimum değişikliği (hız) belirler. Genellikle parçacık aralığına göre belirlenir. Örneğin, x_1 parçacığı (-10,10) aralığında ise $V_{max} = 20$ ile sınırlandırılabilir.
- **Öğrenme Faktörleri:** c_1 ve c_2 genellikle 2 olarak seçilir. Fakat farklı da seçilebilir. Genellikle bu katsayılar [0,4] aralığından alınır.
- **Durma Koşulu:** Maksimum iterasyon sayısına ulaşıldığında veya değer fonksiyonu istenilen seviyeye ulaştığında algoritma durdurulabilir.

3. PARÇACIK SÜRÜ OPTİMİZASYONU TÜRLERİ

3.1. Temel Parçacık Sürü Optimizasyonu (BPSO)

Bir parçacık sürüsündeki bireyler, komşu bireylerin başarısını ve kendi başarılarını taklit etmek için çok basit bir davranış izlemektedir. Bu basit davranıştan meydana gelen ortak davranış, yüksek boyutlu bir arama alanının en uygun bölgelerinin keşfedilmesidir [8].

Bir PSO algoritması, her parçacığın olası bir çözümü temsil ettiği bir parçacık yığınını içerir. Evrimsel hesaplama paradigmalarına benzer şekilde bir sürü bir popülasyona benzerken bir parçacık da bir bireye benzerdir. Basit terimlerle, parçacıklar çok boyutlu bir arama alanıyla "akıtılır", buradaki her parçacığın konumu, kendi deneyimlerine ve komşularının konumuna göre ayarlanır. $\mathbf{x}_i(t)$: t zaman adımındaki arama uzayında $i.$ parçacığın konumunu belirtsin. (aksi belirtilmedikçe t ayrık zamanı belirtir.) Parçacığın konumu bir önceki konumuna yeni hızın eklenmesiyle bulunur. Diğer bir deyişle,

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+1)\,, \quad x_i(0) \sim U(x_{min}, x_{max}) \tag{3.1}$$

Optimizasyon sürecini yönlendiren hız vektörüdür ve parçacığın deneyimsel bilgisini ve parçacıklardan toplumsal olarak değiş tokuş edilen bilgiyi yansıtmaktadır. Bir parçacığı deneyimsel bilgisi genellikle "bilişsel bileşen" olarak adlandırılır. Bu ilk zaman adımından beri bulunan kendi en iyi konumundan parçacığın uzaklığına (parçacığın kişisel en iyi konumu) orantılıdır. Sosyal olarak değiş tokuş edilen bilgilere, hız denkleminin "sosyal bileşeni" denir.

3.2. Azalan Ağırlıklı Parçacık Sürü Optimizasyonu (DWPSO)

Shi ve Eberhart, orijinal PSO'dan güncellenmiş denklemin hızına lineer olarak azalan bir atalet faktörü getirilen, doğrusal düşen bir ağırlık parçacık optimizasyonunu önermişlerdir [9]. Bu yaklaşım PSO'ya kıyasla önemli ölçüde gelişmiştir. Çünkü bu yaklaşımda kümenin küresel ve yerel arama yeteneklerini etkin bir şekilde dengeler. Burada w_{ldw}: atalet ağırlığı olup, arama işlemi boyunca 0.9'dan 0.4'e doğrusal olarak azalır.

$$\boldsymbol{v}_i(t+1) = w_{ldw}\boldsymbol{v}_i(t) + c_1 r_1\big(\mathbf{pbest}_i(t) - \boldsymbol{x}_i(t)\big) + c_2 r_2(\mathbf{gbest}(t) - \boldsymbol{x}_i(t)) \quad (3.2)$$

$$w_{ldw} = w_{start} - (w_{start} - w_{end}) \times \frac{t}{t_{max}} \quad (3.3)$$

3.3. Daralma Katsayılı Parçacık Sürü Optimizasyonu (PSO_C)

Her parçacığın konumu ve hızı kendi sınırlarına sahiptir. Konum sınırları için alt ve üst sınırlar, parçacığın konumu tarafından temsil edilen değişkenlerin sınırlarından gelir. Bununla birlikte parçacıklar için hız sınırları kullanıcılar tarafından tanımlanabilir. Genel olarak optimizasyon problemleri için PSO yönteminin çözüm kalitesi, bilişsel ve sosyal parametrelere ve parçacıkların hız limitine duyarlıdır. Bu nedenle, bilişsel ve sosyal faktörleri ayarlayarak veya $[-V_{idmax}, V_{idmax}]$ aralığında hız aralığını sınırlandırarak PSO algoritmasının arama ve kullanma özelliklerini kontrol etmek için çeşitli girişimler yapılmıştır. Bu yaklaşımda, bir sınırlama faktörünün PSO algoritmasının istikrarlı bir şekilde yakınsamasını sağlamak için gerekli olabileceği savunulmuştur [10].

Daralma faktörü olan parçacıklar için değiştirilmiş hız denklemi aşağıdaki gibi ifade edilmiştir:

$$\boldsymbol{v}_i(t+1) = K \times [\boldsymbol{v}_i(t) + c_1 r_1\big(\mathbf{pbest}_i(t) - \boldsymbol{x}_i(t)\big) + c_2 r_2\big(\mathbf{gbest}(t) - \boldsymbol{x}_i(t)\big)] \quad (3.4)$$

$$K = \frac{2}{\left|2 - \emptyset - \sqrt{\emptyset^2 - 4\emptyset}\right|}, \ \emptyset = c_1 + c_2, \emptyset > 4 \quad (3.5)$$

Daralma katsayılı parçacık sürü optimizasyonunda $\emptyset$ faktörü, sistemin yakınsama özelliği üzerinde bir etkiye sahiptir ve kararlılığı garanti etmek için 4.0'dan büyük olmalıdır. Bununla birlikte $\emptyset$ değeri arttıkça, daralma K üretimin çeşitliliğini azaltır ve bu da daha yavaş tepki alınmasına sebep olur. $\emptyset$ değeri genelde $\emptyset = 4.1$ ve $c_1 = c_2 = 2.05$ seçilir. PSO'da uygulanan daralma faktörü olduğunda arama prosedürü matematik teorisine dayanan yöntemin yakınsamasını sağlar. Sonuç olarak daralma katsayılı parçacık sürü optimizasyonunda temel parçacık sürü optimizasyonu yaklaşımından daha kaliteli çözümler elde edilebilir.

3.4. Zamana Göre Değişen Hız Katsayılı Kendini Örgütleyen Hiyerarşik Parçacık Sürü Optimizasyonu (STVAC_PSO)

Bu yaklaşımda, hız sıfıra ulaşırsa parçacıkların durgunlaşmasının sebebi olarak momentum azlığı sebep gösterilir. Bu sınırlamanın üstesinden gelmek için, bir parçacığın hız vektörü, arama sonrasında durgun hale geldiğinde rastgele bir hızla başlatılır [11].

$$V_{id} = sign(V_{id}).\min(abs(V_{id}, v_{idmax})) \tag{3.6}$$

$$(V_{idmax} = 100) \tag{3.7}$$

3.5. Zamana Göre Değişen Parçacık Sürü Optimizasyonu (TVPSO)

Başlangıçta büyük c_1 ve küçük c_2 katsayısı seçilir ki bu da parçacıkların tasarım alanının etrafında hareket etmesine izin verilir, bu da daha yüksek keşif kabiliyetini gösterir. Öte yandan küçük c_1 ve büyük c_2 seçilmesi global en iyi çözüme (gbest) yakınsamak içindir. Bu nedenle, keşif sürecinde c_2'nin değeri 2.5'ten 0.5'e ve c_1'nin değeri 0.5'ten 2.5'e çıkar [12].

$$c_1 = \left(c_{1f} - c_{1i}\right)\frac{t}{T} + c_{1i} \tag{3.8}$$

$$c_2 = (c_2 f - c_{2i})\frac{t}{T} + c_{2i} \tag{3.9}$$

3.6. Zamana Bağlı Olarak Değişen Atalet Katsayılı Parçacık Sürü Optimizasyonu (TVIW_PSO)

Bu yaklaşımda atalet sayısının lineer olarak azalması temel alınmıştır [11]. Uygun atalet katsayısının seçimi global ve lokal keşif yeteneği arasında denge sağlıyor ve böylelikle en az iterasyonda optimum sonucu bulmuş oluyor.

$$w = (w_{max} - w_{min}) \times \frac{k_{max}-k}{k_{max}} + w_{min} \tag{3.10}$$

Yakınsama oranını arttırmak için, daralma faktörü (C) aşağıdaki denklemde ifade edilen öz değer analizi ile analiz edilir.

$$C = \frac{2}{\left|2-\emptyset-\sqrt{\emptyset^2-4\emptyset}\right|}, \quad 4.1 \leq \emptyset \leq 4.2 \tag{3.11}$$

Sistemin yakınsama özelliği $\emptyset$ tarafından kontrol edilir. $\emptyset$ faktörünün artması, yakınsama oranının yavaşlamasına yol açar, çünkü nüfus çeşitliliği azalır.

Her bir parçacığın değişen hızı aşağıdaki denklem ile ifade edilir:

$$\boldsymbol{v}_i(t+1) = C \times [\boldsymbol{v}_i(t) + c_1 r_1 \big(\mathbf{pbest}_i(t) - \boldsymbol{x}_i(t)\big) + \ c_2 r_2 \big(\mathbf{gbest}(t) - \boldsymbol{x}_i(t)\big)]$$

3.7. Zamana Bağlı İvmelendirici Katsayılı Parçacık Sürü Optimizasyonu (TVAC-PSO)

Zamana bağlı olarak değişen atalet katsayılı PSO hızlı bir yakınsama sağladığından daha iyi bir çözüm belirleyebiliyor olsa da, arama sonunda çeşitlilik nedeniyle en uygun çözüme ince ayar yapma kabiliyeti eksiktir. Zamana bağlı değişen ivmelendirici katsayılı PSO da global aramayı erken aşamada geliştirir ve aramaları sonucunda küresel optimuma doğru yakınsama sağlar [13]. c_1 ve c_2 hızlandırma katsayılarını değiştirerek bilişsel bileşen azalır arama devam ederken sosyal bileşen artar. Büyük bir bilişsel bileşen ve küçük bir sosyal bileşen, parçacıkların erken aşamalarda nüfusa en iyi hareket ettirmek yerine arama alanının etrafında hareket etmesine izin verir. Matematiksel olarak ivmelendirici katsayılar aşağıdaki şekilde hesaplanır:

$$c_1 = \left(c_{1f} - c_{1i}\left(\frac{k}{k_{max}}\right) + c_{1i}\right) \tag{3.13}$$

$$c_2 = \left(c_{1f} - c_{2i}\left(\frac{k}{k_{max}}\right) + c_{2i}\right) \tag{3.14}$$

4. DIRICHLET SINIR KOŞULUNA SAHİP ADİ DİFERANSİYEL DENKLEMLERİN NÜMERİK ÇÖZÜMLERİ

$$\begin{cases} y'' = f(x, y(x), y'(x)), \ x \in [a,b] \\ y(a) = A \\ y(b) = B \end{cases} \tag{4.1}$$

formundaki diferansiyel denklemler Dirichlet sınır değer problemi olarak bilinir. x bağımsız bir değişken olup, $[a,b]$ kapalı aralığından seçilir. y hesaplanacak olan bilinmeyen bir bağımlı değişkendir.

$y_T = (x, \mathbf{p})$ fonksiyonunun (4.1) denklemi için sınır koşullarını sağlayan ve yapay sinir ağı çözümüne bağlı bir yaklaşık çözüm olduğunu varsayalım. $\mathbf{p}$ yapay sinir ağının nöronları arasındaki bağlantıların ağırlıklarını ve eşik değerleri içeren ayarlanabilir parametre vektörüdür. $\mathbf{p}$ vektörünün boyutu yapay sinir ağının ara katmanında yer alan nöron sayısına bağlı olarak değişir. Eğer yapay sinir ağı m adet nöron içeriyorsa $\mathbf{p} \in \mathbb{R}^{3m}$ olur. Deneme fonksiyonu olarak adlandırılan ve a ve b noktalarındaki sınır koşullarını sağlayan y_T fonksiyonu (4.2) eşitliğinde verildiği gibi tanımlıdır.

$$y_T = \frac{bA-aB}{b-a} + \frac{B-A}{b-a}x + (x-a)(x-b)Net(x, \mathbf{p}) \tag{4.2}$$

(4.2) eşitliğinde verilen $Net(x, \mathbf{p})$ ileri beslemeli yapay sinir ağının $j = 1, 2, \dots, n$ için x_j girdisine göre yapay sinir ağının çıktısı (4.3) eşitliğinde verilmiştir.

$$Net(x_j, \boldsymbol{p}) = \sum_{i=1}^{m} \alpha_i \sigma(z_i) \tag{4.3}$$

(4.3) eşitliğinde $z_i = \omega_i x_j + \beta_i$ yapay sinir ağının nöronlarını temsil eder ve ω_i yapay sinir ağının girdi katmanından ara katmandaki i. nörona giden bağlantının ağırlığıdır. α_i değeri ise ara katmandaki i. nörondan çıktı katmanındaki nörona giden ağırlık parametresidir. β_i değeri z_i nöronunun eşik değeridir. Yapay sinir ağında kullanılan aktivasyon fonksiyonu sigmoid fonksiyonudur.

(4.2) denkleminde verilen ifadenin türevi alındığında;

$$\frac{\partial y_T}{\partial x} = \frac{B-A}{b-a} + (2x-a-b)Net(x, \mathbf{p}) + (x-a)(x-b)\frac{\partial Net(x, \mathbf{p})}{\partial x} \tag{4.4}$$

denklemi elde edilir.

(4.4) ile ifade edilmiş olan denklemin türevi alınırsa;

$$\frac{\partial^2 y_T}{\partial x^2} = 2Net(x,\mathbf{p}) + (2x-a-b)\frac{\partial Net(x,\mathbf{p})}{\partial x} + (x-b)\frac{\partial Net(x,\mathbf{p})}{\partial x} + (x-a)\frac{\partial Net(x,\mathbf{p})}{\partial x} + (x-a)(x-b)\frac{\partial^2 Net(x,\mathbf{p})}{\partial x^2} \quad (4.5)$$

ve (4.5) denklemini düzenlendiğinde, (4.6) denklemi elde edilmiş olur.

$$\frac{\partial^2 y_T}{\partial x^2} = 2Net(x,\mathbf{p}) + (4x-2a-2b)\frac{\partial Net(x,\mathbf{p})}{\partial x} + (x-a)(x-b)\frac{\partial^2 Net(x,\mathbf{p})}{\partial x^2} \quad (4.6)$$

E mutlak hata olmak üzere (4.7) denkleminde verildiği şekildedir.

$$E = \frac{1}{2}\sum_{j=1}^{n}\left[\frac{\partial^2 y_T}{\partial x_j^2} - f\left(x_j, y_T, \frac{\partial y_T}{\partial x_j}\right)\right]^2 \quad (4.7)$$

Burada amaç hatanın mümkün olduğunca sıfıra yakın bulunmasıdır. Hata fonksiyonu sırasıyla α_i, ω_i ve β_i değişkenlerine göre türev alındığında (4.8), (4.9) ve (4.10) denklemleri elde edilmiş olur.

$$\frac{\partial E}{\partial \alpha_i} = \sum_{j=1}^{n}\left[\frac{\partial^2 y_T}{\partial x_j^2} - f\left(x_j, y_T, \frac{\partial y_T}{\partial x_j}\right)\right]\left[\frac{\partial^3 y_T}{\partial \alpha_i \partial x_j^2} - \frac{\partial f\left(x_j, y_T, \frac{\partial y_T}{\partial x_j}\right)}{\partial \alpha_i}\right] \quad (4.8)$$

$$\frac{\partial E}{\partial \omega_i} = \sum_{j=1}^{n}\left[\frac{\partial^2 y_T}{\partial x_j^2} - f\left(x_j, y_T, \frac{\partial y_T}{\partial x_j}\right)\right]\left[\frac{\partial^3 y_T}{\partial \omega_i \partial x_j^2} - \frac{\partial f\left(x_j, y_T, \frac{\partial y_T}{\partial x_j}\right)}{\partial \omega_i}\right] \quad (4.9)$$

$$\frac{\partial E}{\partial \beta_i} = \sum_{j=1}^{n}\left[\frac{\partial^2 y_T}{\partial x_j^2} - f\left(x_j, y_T, \frac{\partial y_T}{\partial x_j}\right)\right]\left[\frac{\partial^3 y_T}{\partial \beta_i \partial x_j^2} - \frac{\partial f\left(x_j, y_T, \frac{\partial y_T}{\partial x_j}\right)}{\partial \beta_i}\right] \quad (4.10)$$

$$\frac{\partial Net(x,\mathbf{p})}{\partial x} = \sum_{i=1}^{m}\alpha_i \omega_i \sigma(z_i)\big(1-\sigma(z_i)\big) \quad (4.11)$$

$$\frac{\partial^2 Net(x,\mathbf{p})}{\partial x^2} = \sum_{i=1}^{m}\alpha_i {\omega_i}^2 \sigma(z_i)\big(1-\sigma(z_i)\big)\big(1-2\sigma(z_i)\big) \quad (4.12)$$

(4.8) denkleminde verilen türevlerin bulunması için (4.13), (4.14), (4.15), (4.16) eşitlikleri hesaplanmak için verilmiştir.

$$\frac{\partial Net(x,\mathbf{p})}{\partial \alpha_i} = \sigma(z_i) \tag{4.13}$$

$$\frac{\partial^2 Net(x,\mathbf{p})}{\partial {\alpha_i}^2} = 0 \tag{4.14}$$

$$\frac{\partial^2 Net(x,\mathbf{p})}{\partial \alpha_i \partial x_j} = \omega_i\, \sigma(z_i)\big(1-\sigma(z_i)\big) \tag{4.15}$$

$$\frac{\partial^3 Net(x,\mathbf{p})}{\partial \alpha_i \partial {x_j}^2} = {\omega_i}^2\, \sigma(z_i)\big(1-\sigma(z_i)\big)\big(1-2\sigma(z_i)\big) \tag{4.16}$$

(4.9) denkleminde verilen türevlerin hesaplanması için (4.17), (4.18), (4.19), (4.20) eşitliklerine ihtiyaç duyulmaktadır.

$$\frac{\partial Net(x,\mathbf{p})}{\partial \omega_i} = \alpha_i x_j\, \sigma(z_i)\big(1-\sigma(z_i)\big) \tag{4.17}$$

$$\frac{\partial^2 Net(x,\mathbf{p})}{\partial {\omega_i}^2} = \alpha_i {x_j}^2\, \sigma(z_i)\big(1-\sigma(z_i)\big)\big(1-2\sigma(z_i)\big) \tag{4.18}$$

$$\frac{\partial^2 Net(x,\mathbf{p})}{\partial \omega_i \partial x_j} = \alpha_i\, \sigma(z_i)\big(1-\sigma(z_i)\big)\big\{1+\omega_i x_j\big(1-\sigma(z_i)\big)+x_j\sigma(z_i)\big\} \tag{4.19}$$

$$\begin{aligned}\frac{\partial^3 Net(x,\mathbf{p})}{\partial \omega_i \partial {x_j}^2} &= 2\omega_i\sigma(z_i)\big(1-\sigma(z_i)\big)\big(1-2\sigma(z_i)\big)\\ &+{\omega_i}^2 x_j\sigma(z_i)\big(1-\sigma(z_i)\big)^2\big(1-2\sigma(z_i)\big)\\ &-{\omega_i}^2 x_j\sigma(z_i)^2\big(1-\sigma(z_i)\big)\big(1-2\sigma(z_i)\big)\end{aligned} \tag{4.20}$$

(4.10) denkleminde verilen türevlerin hesaplanması için (4.21), (4.22), (4.23), (4.24) eşitlikleri verilmiştir.

$$\frac{\partial Net(x,\mathbf{p})}{\partial \beta_i} = \alpha_i\, \sigma(z_i)\big(1-\sigma(z_i)\big) \tag{4.21}$$

$$\frac{\partial^2 Net(x,\mathbf{p})}{\partial {\beta_i}^2} = \alpha_i\, \sigma(z_i)\big(1-\sigma(z_i)\big)\big(1-2\sigma(z_i)\big) \tag{4.22}$$

$$\frac{\partial^2 Net(x,\mathbf{p})}{\partial \beta_i \partial x_j} = \alpha_i\omega_i\sigma(z_i)\big(1-\sigma(z_i)\big)\big(1-2\sigma(z_i)\big) \tag{4.23}$$

$$\begin{aligned}\frac{\partial^3 Net(x,\mathbf{p})}{\partial \beta_i \partial {x_j}^2} = \alpha_i {\omega_i}^2 \Big(&\sigma(z_i)\big(1-\sigma(z_i)\big)^2\big(1-2\sigma(z_i)\big)-\sigma(z_i)^2\big(1-\sigma(z_i)\big)\big(1-\\ &2\sigma(z_i)\big)-2\sigma(z_i)^2\big(1-2\sigma(z_i)\big)^2\Big)\end{aligned} \tag{4.24}$$

Bir sonraki bölümde, ikinci bölümde bahsedilen PSO türevleri kullanılarak Dirichlet sınır değer problemlerinin çözümlerine ait deneysel çalışmalar sunulmuştur.

5. DENEYSEL ÇALIŞMALAR

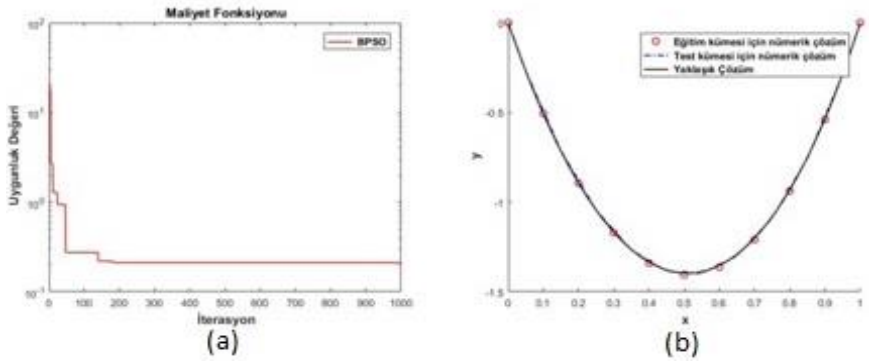
Bu bölümde lineer ve lineer olmayan ikinci mertebeden farklı sınır koşullarına sahip Dirichlet Sınır Değer Problemleri için örnekler verilmiş ve tezde bahsi geçen yöntemlerle çözümleri elde edilmiştir.

Örnek 5.1. $\begin{cases} y'' = -y' + 2y + 8e^x,\ x \in [0,1] \\ y(0) = 0, \\ y(1) = 0 \end{cases}$ (5.1)

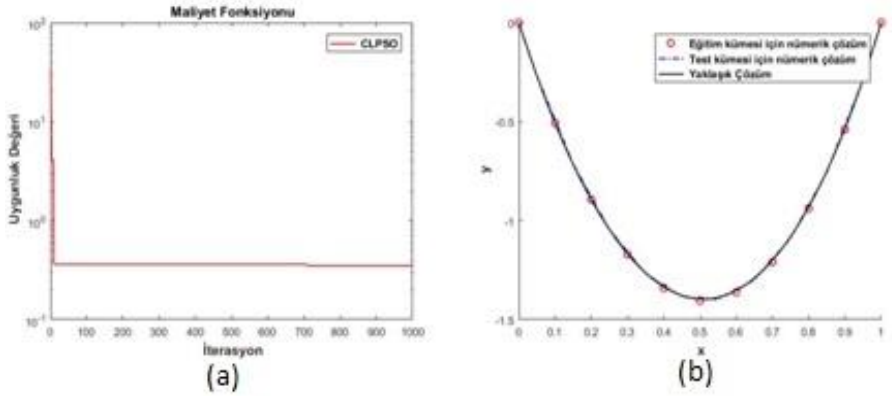
(5.1) denklemi ile verilen 2. mertebeden lineer Dirichlet sınır değer problemlerinin gerçek çözümü

$$y_e = \frac{8}{3}\frac{e^3}{e^3-1}(e^{-2x} - e^x) + xe^x \quad (5.2)$$

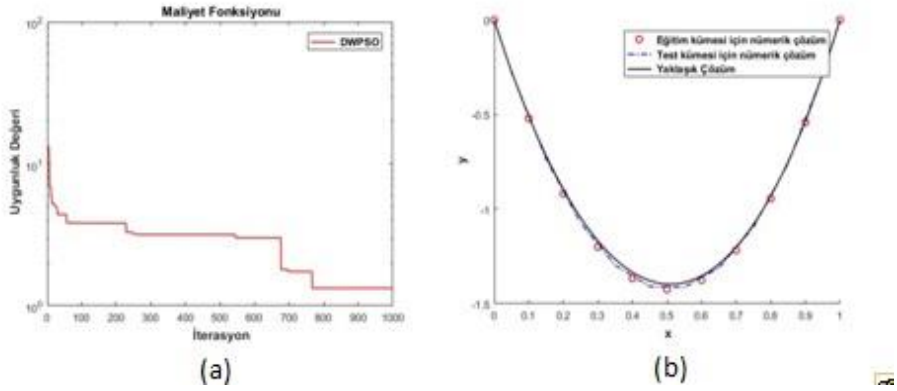
(5.2) denklemi ile verilmiştir. (5.1) denkleminin $h = 0.1$ için tezde geçen PSO varyasyonlarından elde edilen nümerik çözümler eğitim ve test kümeleri için Çizelge 5.1 ve Çizelge 5.2'de sunulmuştur. Şekil 5.1 (5.1) denkleminin BPSO yöntemiyle elde edilen maliyet değerindeki iterasyona bağlı değişimini gösterir ve denklemin çözümünü verir. BPSO, CLPSO, DWPSO, STVAC_PSO, PSO_C, TVAC_PSO, TVIW_PSO, TVPSO yöntemlerinden elde edilen çözümler sırasıyla Şekil 5.2-Şekil 5.18 ile gösterilmiştir.



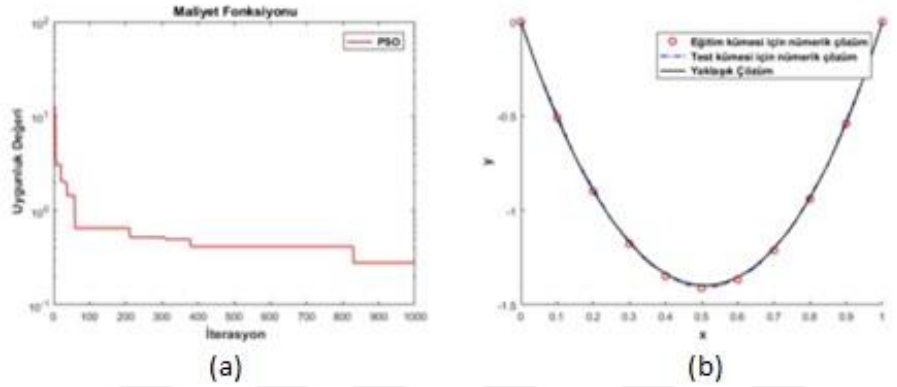
Şekil 5.1. (a) Örnek 5.1 için BPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) BPSO yönteminin yapay sinir ağı çözümü $(h = 0.1)$



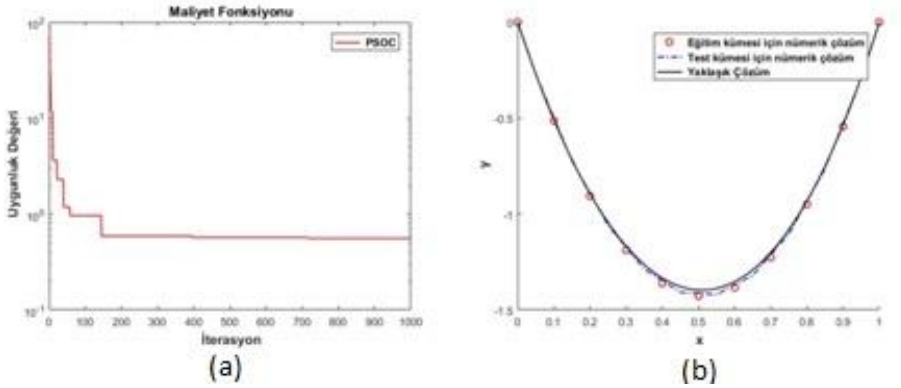
Şekil 5.2. (a) Örnek 5.1 için CLPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) CLPSO yönteminin yapay sinir ağı çözümü $(h = 0.1)$



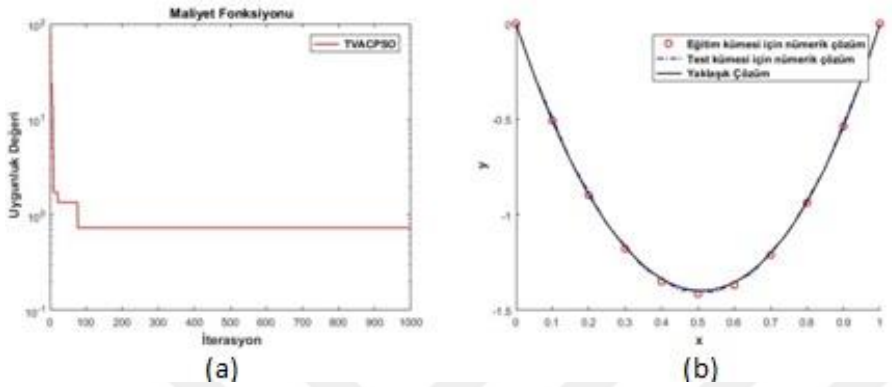
Şekil 5.3. (a) Örnek 5.1 için DWPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) DWPSO yönteminin yapay sinir ağı çözümü $(h = 0.1)$



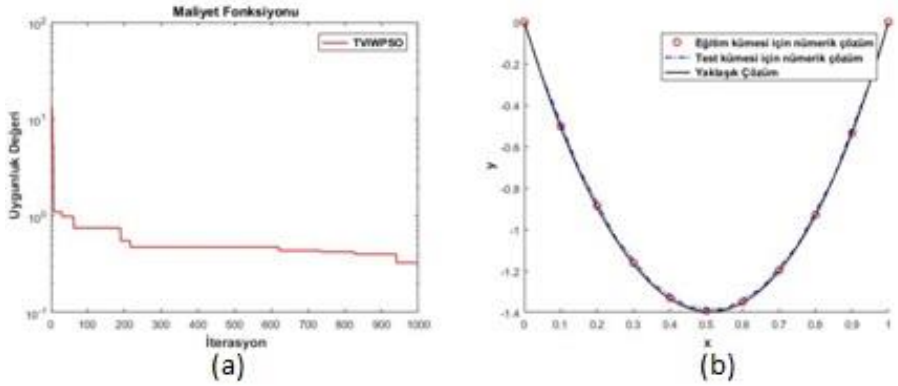
Şekil 5.4. (a) Örnek 5.1 için STVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) STVAC_PSO yönteminin yapay sinir ağı çözümü $(h = 0.1)$



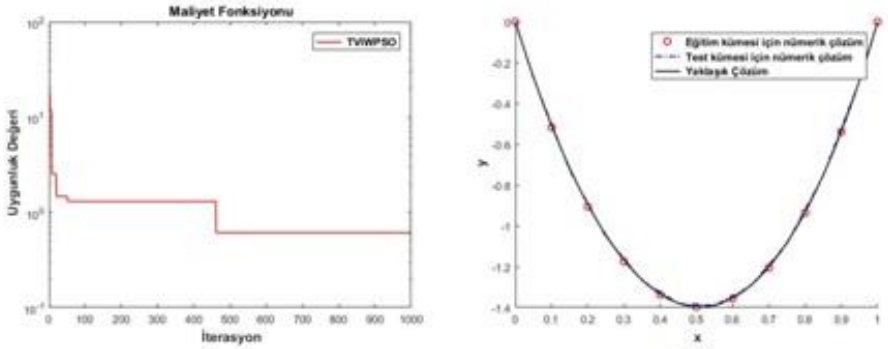
Şekil 5.5. (a) Örnek 5.1 için PSO_C yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) PSO_C yönteminin yapay sinir ağı çözümü $(h = 0.1)$



Şekil 5.6. (a) Örnek 5.1 için TVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVAC_PSO yönteminin yapay sinir ağı çözümü ($h = 0.1$)



Şekil 5.7. (a) Örnek 5.1 için TVIW_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVIW_PSO yönteminin yapay sinir ağı çözümü ($h = 0.1$)



Şekil 5.8. (a) Örnek 5.1 için TVPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVPSO yönteminin yapay sinir ağı çözümü $(h = 0.1)$

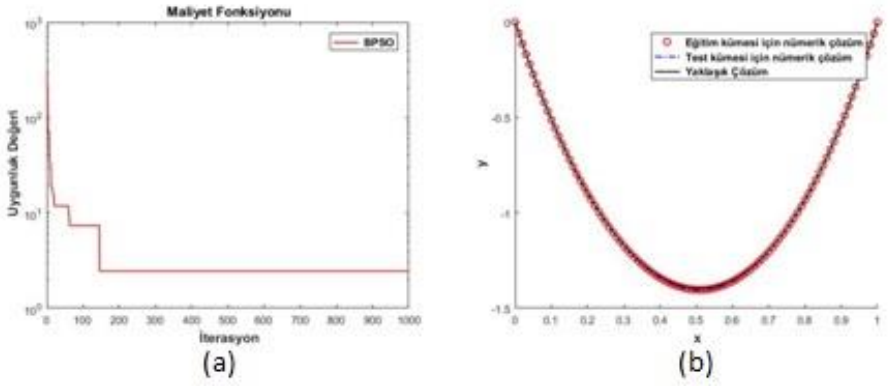
Çizelge 5.1. Örnek 5.1 için $h = 0.1$ adım aralığıyla oluşturulan maliyet fonksiyonu için eğitim kümesinin hata miktarları

		Mutlak Hata $(\times 10^{-3})$							
k	x_k	**BPSO**	**CLPSO**	**DWPSO**	**STVAC_ PSO**	**PSO_ C**	**TVAC_ PSO**	**TVIW _PSO**	**TVPSO**
1	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.1	1.43	7.67	1.21	1.40	0.99	1.86	7.57	8.29
3	0.2	0.55	14.12	0.54	0.66	4.28	2.42	9.11	10.15
4	0.3	4.15	18.08	1.05	1.50	7.91	3.18	7.02	6.47
5	0.4	7.45	18.90	2.30	4.15	10.18	5.64	4.55	2.71
6	0.5	9.32	16.78	2.39	6.20	10.37	9.77	3.42	2.07
7	0.6	9.55	12.67	1.31	6.79	8.78	13.95	3.53	3.53
8	0.7	8.58	8.03	0.05	5.59	6.33	15.83	3.70	4.75
9	0.8	6.95	4.24	0.57	2.99	3.97	13.57	2.87	4.46
10	0.9	4.58	1.94	0.10	0.27	2.14	7.241	1.19	2.77
11	1.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

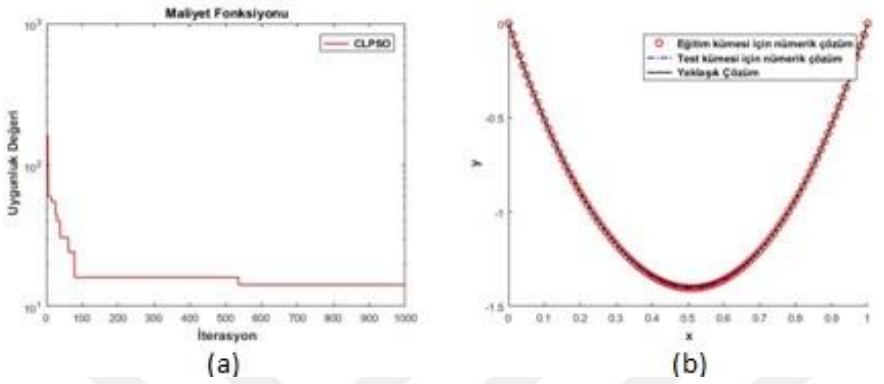
Çizelge 5.2. Örnek 5.1 için $h = 0.1$ adım aralığıyla oluşturulan çözüm fonksiyonu için test kümesinin hata miktarları

		Mutlak Hata ($\times 10^{-3}$)							
k	x_k	**BPSO**	**CLPSO**	**DWPSO**	**STVAC _PSO**	**PSO _C**	**TVAC_ PSO**	**TVIW _PSO**	**TVPSO**
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.05	1.19	3.89	0.86	1.02	0.12	1.11	4.52	4.81
3	0.15	0.77	11.14	1.07	1.26	2.46	2.24	9.02	10.13
4	0.25	2.28	16.47	0.22	0.29	6.18	2.66	8.29	8.69
5	0.35	5.93	18.89	1.79	2.84	9.29	4.16	5.68	4.29
6	0.45	8.59	18.16	2.51	5.31	10.53	7.57	3.79	2.00
7	0.55	9.62	14.89	1.9	6.72	9.75	11.99	3.38	2.69
8	0.65	9.18	10.32	0.59	6.41	7.60	15.32	3.69	4.28
9	0.75	7.83	5.9	0.48	4.41	5.09	15.27	3.43	4.81
10	0.85	5.90	2.92	0.33	1.52	3.00	10.80	2.06	3.75
11	0.95	2.73	1.10	0.41	0.39	1.22	3.38	0.44	1.54
12	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

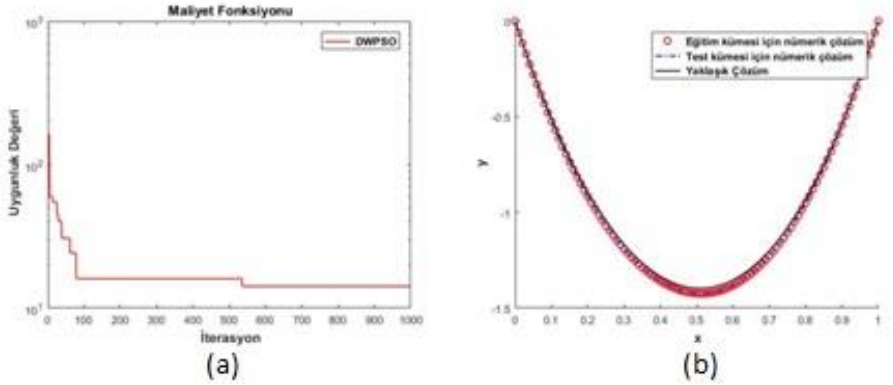
(5.1) denkleminin $h = 0.01$ için tezde geçen PSO varyasyonlarından elde edilen nümerik çözümler Çizelge 5.3 ve Çizelge 5.4'te verilmiş olup yöntemlerden elde edilen çözümler sırasıyla Şekil 5.9-Şekil 5.16 ile gösterilmiştir.



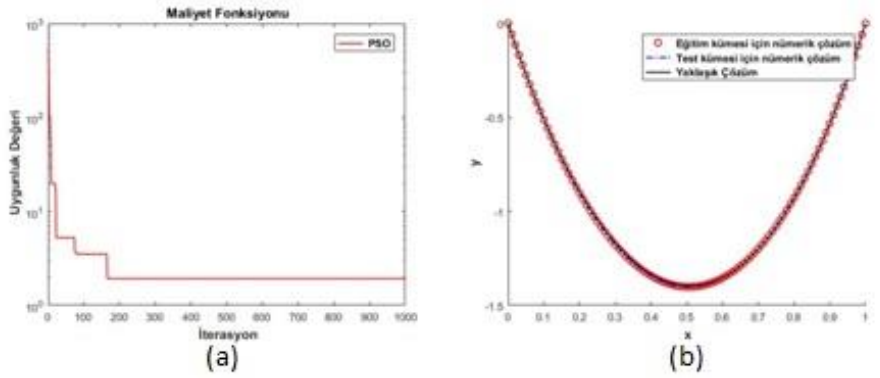
Şekil 5.9. (a) Örnek 5.1 için BPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) BPSO yönteminin yapay sinir ağı çözümü $(h = 0.01)$



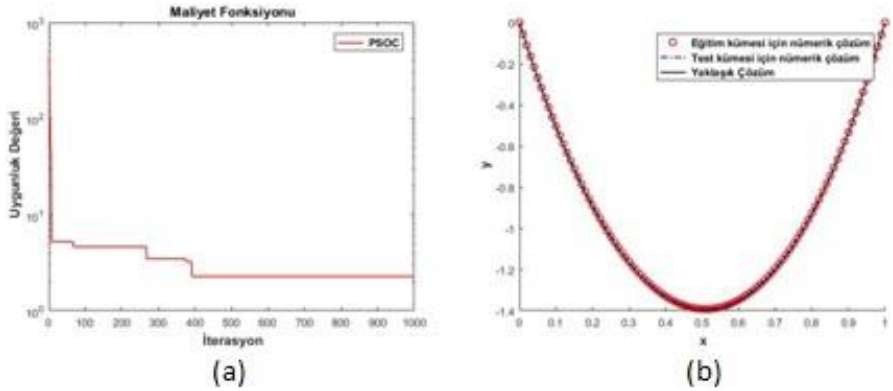
Şekil 5.10. (a) Örnek 5.1 için CLPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) CLPSO yönteminin yapay sinir ağı çözümü ($h = 0.01$)



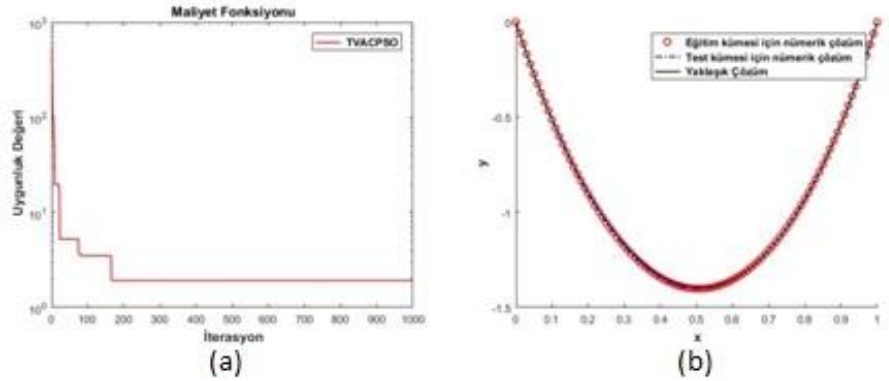
Şekil 5.11. (a) Örnek 5.1 için DWPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) DWPSO yönteminin yapay sinir ağı çözümü ($h = 0.01$)



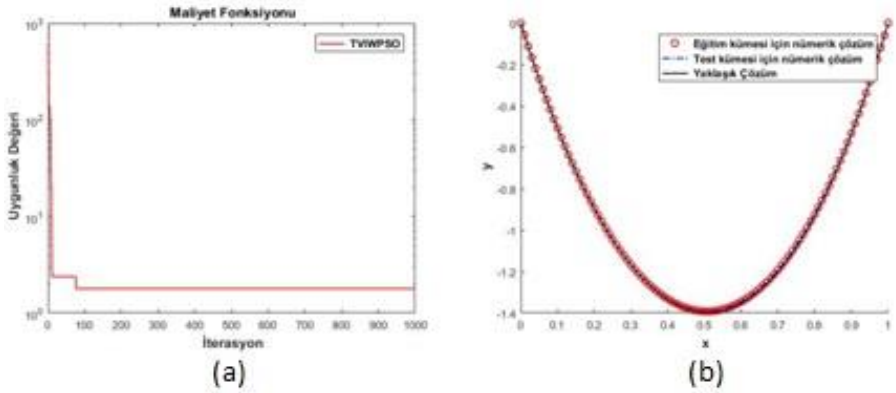
Şekil 5.12. (a) Örnek 5.1 için STVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) STVAC_PSO yönteminin yapay sinir ağı çözümü $(h = 0.01)$



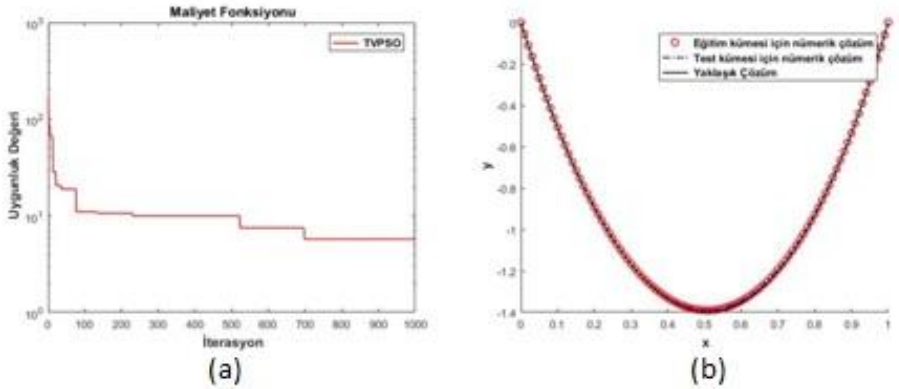
Şekil 5.13. (a) Örnek 5.1 için PSO_C yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) PSO_C yönteminin yapay sinir ağı çözümü $(h = 0.01)$



Şekil 5.14. (a) Örnek 5.1 için TVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVAC_PSO yönteminin yapay sinir ağı çözümü $(h = 0.01)$



Şekil 5.15. (a) Örnek 5.1 için TVIW_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVIW_PSO yönteminin yapay sinir ağı çözümü $(h = 0.01)$



Şekil 5.16. (a) Örnek 5.1 için TVPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVPSO yönteminin yapay sinir ağı çözümü $(h = 0.01)$

Çizelge 5.3. Örnek 5.1 için $h = 0.01$ adım aralığıyla oluşturulan çözüm fonksiyonu için eğitim kümesinin hata miktarları

		Mutlak Hata $(\times 10^{-3})$							
k	x_k	**BPSO**	**CLPSO**	**DWPSO**	**STVAC_PSO**	**PSO_C**	**TVAC_PSO**	**TVIW_PSO**	**TVPSO**
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.01	0.58	1.62	0.76	0.31	0.34	0.43	0.29	0.78
3	0.02	1.07	3.25	1.48	0.63	0.69	0.88	0.55	1.44
4	0.03	1.48	4.87	2.15	0.95	1.05	1.36	0.78	1.98
5	0.04	1.82	6.46	2.76	1.28	1.42	1.86	0.98	2.41
11	0.10	2.80	14.76	5.40	3.43	3.59	4.99	1.64	3.35
21	0.20	3.36	22.06	6.26	6.96	4.84	9.82	1.42	2.16
31	0.30	4.73	23.93	4.92	8.88	1.64	13.39	1.13	1.64
41	0.40	6.32	25.46	3.82	8.16	3.33	15.73	1.97	3.14
51	0.50	6.81	28.20	3.94	5.18	6.79	16.99	4.14	5.67
61	0.60	5.67	29.89	4.86	1.38	7.33	16.88	6.79	7.32
71	0.70	3.53	27.31	5.48	1.68	5.51	14.84	8.44	6.81
81	0.80	1.52	19.11	4.87	3.04	2.92	10.50	7.70	4.32
91	0.90	0.42	7.69	2.87	2.54	1.04	4.59	4.27	1.50
101	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Çizelge 5.4. Örnek 5.1 için $h = 0.01$ adım aralığıyla oluşturulan çözüm fonksiyonu için test kümesinin hata miktarları

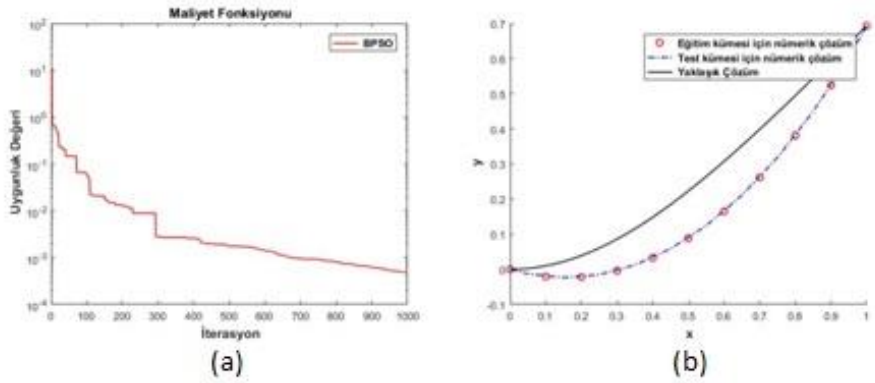
		Mutlak Hata ($\times 10^{-3}$)							
k	x_k	**BPSO**	**CLPSO**	**DWPSO**	**STVAC_PSO**	**PSO_C**	**TVAC_PSO**	**TVIW_PSO**	**TVPSO**
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.01	3.03	3.03	3.03	3.03	3.03	3.04	3.03	3.03
3	0.02	6.07	6.08	6.07	6.08	6.06	6.08	6.07	6.07
4	0.03	9.11	9.12	9.11	9.12	9.10	9.13	9.11	9.11
5	0.04	12.15	12.17	12.15	12.17	12.14	12.18	12.15	12.16
11	0.10	30.43	30.52	30.42	30.48	30.41	30.55	30.45	30.47
21	0.20	60.87	60.97	60.83	60.85	60.86	61.07	60.86	60.87
31	0.30	90.05	90.01	90.02	89.88	90.06	90.20	90.03	89.94
41	0.40	115.53	115.39	115.47	115.34	115.56	115.55	115.56	115.36
51	0.50	134.14	134.06	134.06	134.17	134.21	134.09	134.25	134.06
61	0.60	142.41	142.50	142.47	142.73	142.47	142.37	142.52	142.50
71	0.70	136.73	136.98	137.03	137.24	136.74	136.76	136.75	136.95
81	0.80	113.51	113.75	113.84	113.94	113.44	113.57	113.40	113.71
91	0.90	69.11	69.21	69.23	69.28	69.05	69.13	69.01	69.18
101	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Örnek 5.2
$$\begin{cases} y'' = \dfrac{-2x}{1+x^2}\,y' + y + \dfrac{2}{1+x^2} - \log\left(1+x^2\right),\ x \in [0,1] \\ y(0) = 0, \\ y(1) = \log(2) \end{cases} \tag{5.3}$$

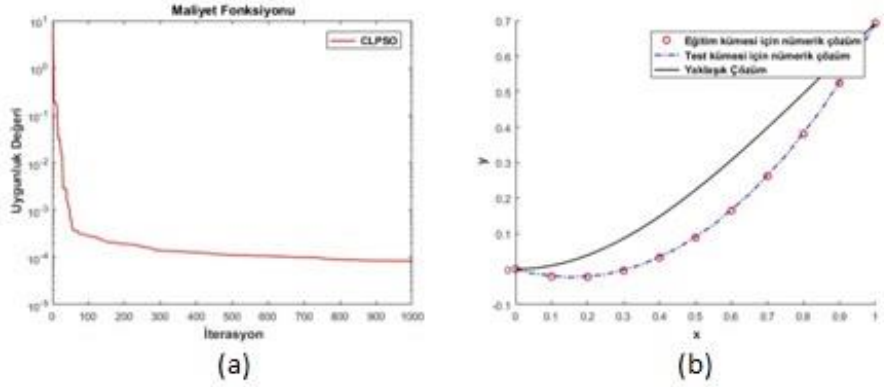
(5.3) denklemi ile verilen 2. mertebeden lineer Dirichlet sınır değer problemlerinin gerçek çözümü

$$y_e = \log(1 + x^2) \tag{5.4}$$

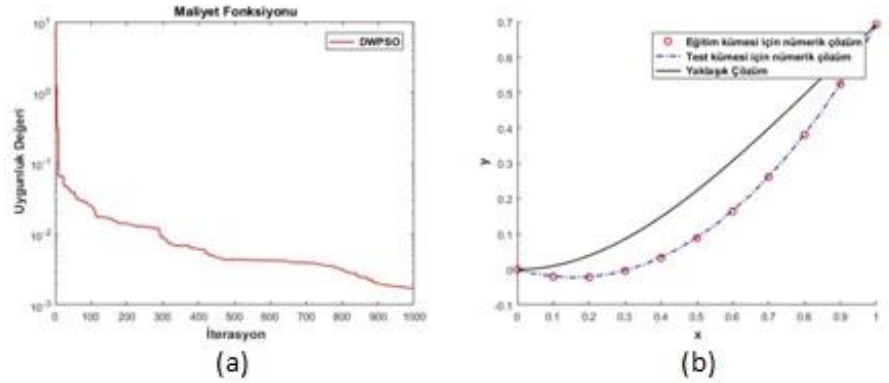
(5.4) denklemi ile verilmiştir. (5.3) denkleminin $h = 0.1$ için tezde geçen PSO varyasyonlarından elde edilen nümerik çözümler eğitim ve test kümeleri için Çizelge 5.5 ve Çizelge 5.6'da sunulmuştur. Şekil 5.17-Şekil 5.24 (5.3) denkleminin PSO yöntemleriyle elde edilen maliyet değerindeki iterasyona bağlı değişimini gösterir ve denklemin çözümünü verir.



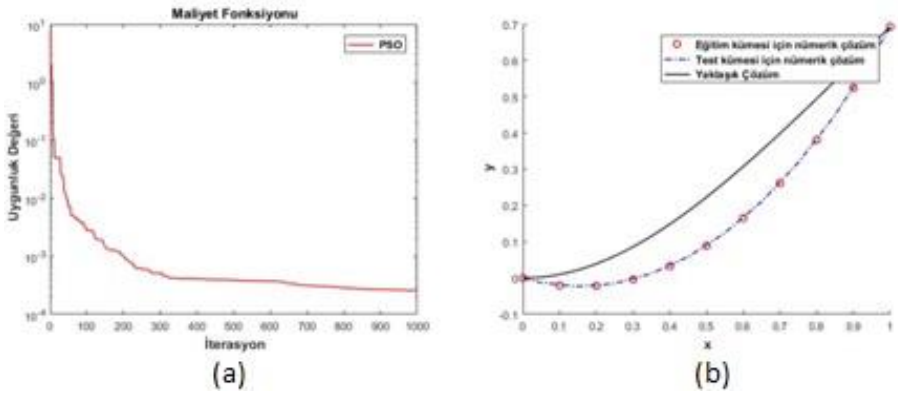
Şekil 5.17. (a) Örnek 5.2 için BPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) BPSO yönteminin yapay sinir ağı çözümü $(h = 0.1)$



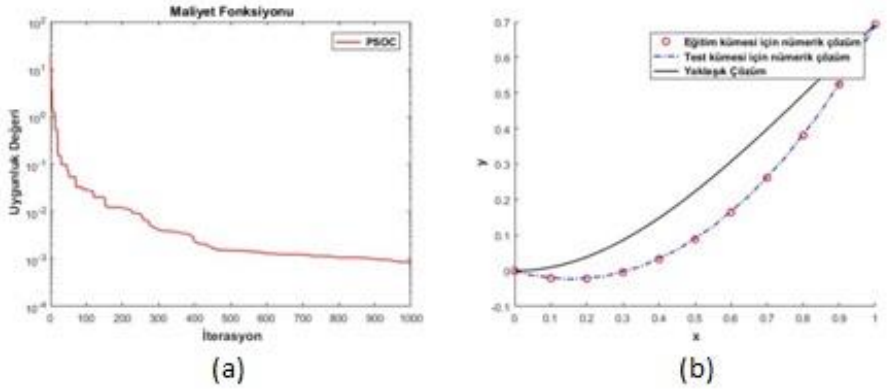
Şekil 5.18. (a) Örnek 5.2 için CLPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) CLPSO yönteminin yapay sinir ağı çözümü $(h = 0.1)$



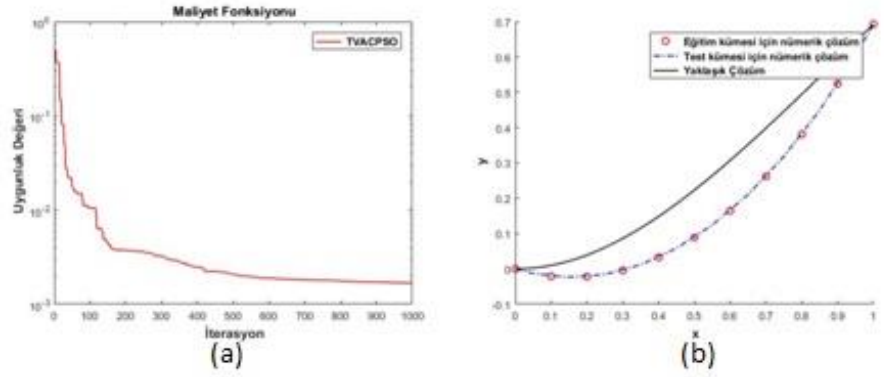
Şekil 5.19. (a) Örnek 5.2 için DWPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) DWPSO yönteminin yapay sinir ağı çözümü $(h = 0.1)$



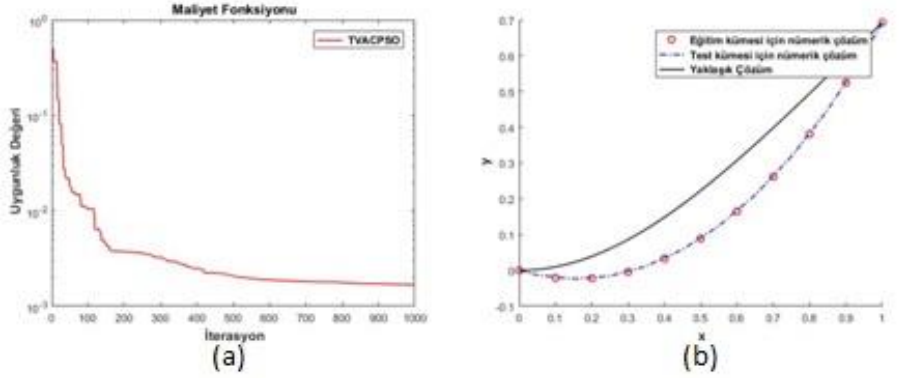
Şekil 5.20. (a) Örnek 5.2 için STVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) STVAC_PSO yönteminin yapay sinir ağı çözümü $(h = 0.1)$



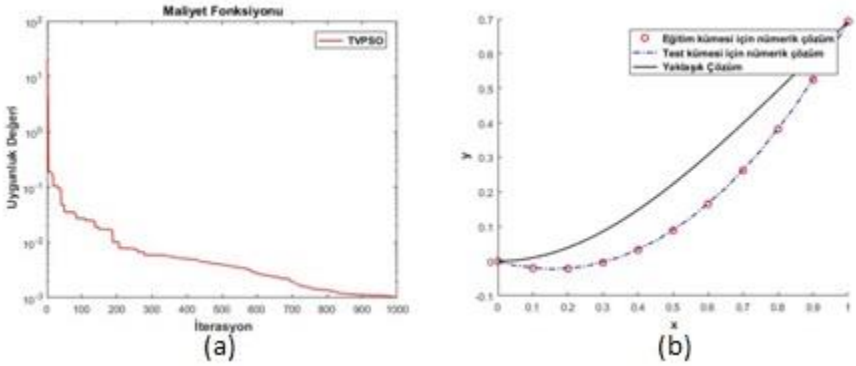
Şekil 5.21. (a) Örnek 5.2 için PSO_C yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) PSO_C yönteminin yapay sinir ağı çözümü $(h = 0.1)$



Şekil 5.22. (a) Örnek 5.2 için TVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVAC_PSO yönteminin yapay sinir ağı çözümü $(h = 0.1)$



Şekil 5.23. (a) Örnek 5.2 için TVIW_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVIW_PSO yönteminin yapay sinir ağı çözümü $(h = 0.1)$



Şekil 5.24. (a) Örnek 5.2 için TVPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVPSO yönteminin yapay sinir ağı çözümü $(h = 0.1)$

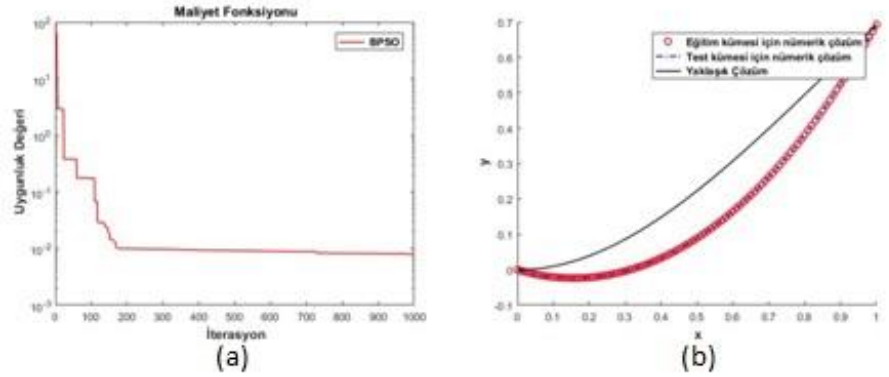
Çizelge 5.5. Örnek 5.2 için $h = 0.1$ adım aralığıyla oluşturulan çözüm fonksiyonu için eğitim kümesinin hata miktarları

		Mutlak Hata							
k	x_k	**BPSO**	**CLPSO**	**DWPSO**	**STVAC_PSO**	**PSO_C**	**TVAC_PSO**	**TVIW_PSO**	**TVPSO**
1	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.1	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
3	0.2	0.06	0.06	0.06	0.06	0.06	0.06	0.06	0.06
4	0.3	0.08	0.09	0.08	0.08	0.08	0.09	0.08	0.09
5	0.4	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11
6	0.5	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13
7	0.6	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14
8	0.7	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13
9	0.8	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11
10	0.9	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
11	1.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

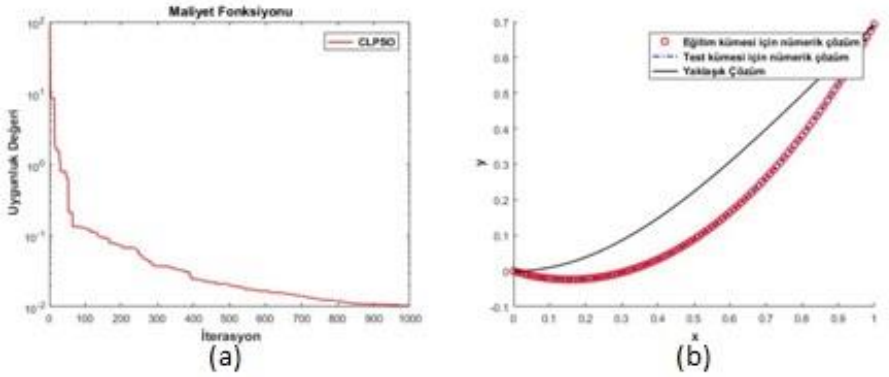
Çizelge 5.6. Örnek 5.2 için $h = 0.1$ adım aralığıyla oluşturulan çözüm fonksiyonu için test kümesinin hata miktarları

		Mutlak Hata							
k	x_k	**BPSO**	**CLPSO**	**DWPSO**	**STVAC_PSO**	**PSO_C**	**TVAC_PSO**	**TVIW_PSO**	**TVPSO**
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.05	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
3	0.15	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04
4	0.25	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07
5	0.35	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10
6	0.45	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12
7	0.55	0.13	0.11	0.13	0.13	0.13	0.13	0.14	0.13
8	0.65	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14
9	0.75	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12
10	0.85	0.09	0.09	0.09	0.09	0.09	0.09	0.09	0.09
11	0.95	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
12	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

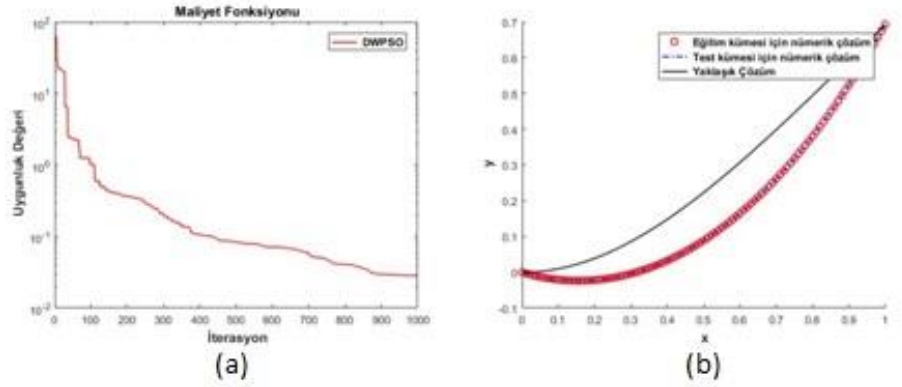
(5.3) denkleminin $h = 0.01$ için tezde geçen PSO varyasyonlarından elde edilen nümerik çözümler Çizelge 5.7 ve Çizelge 5.8'de verilmiş olup yöntemlerden elde edilen çözümler sırasıyla Şekil 5.25-Şekil 5.32 ile gösterilmiştir.



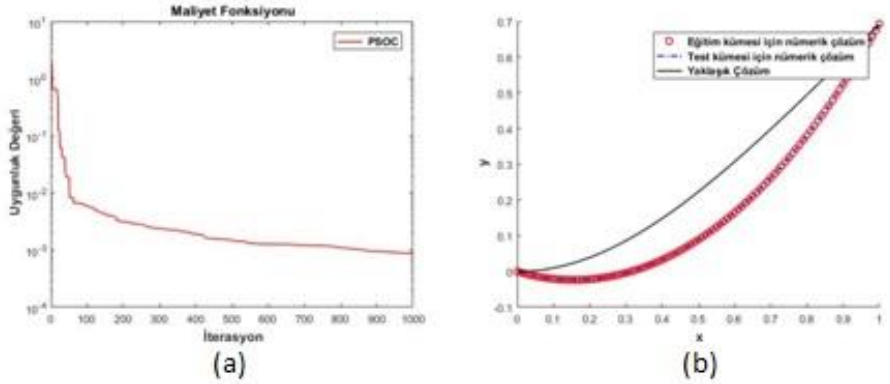
Şekil 5.25. (a) Örnek 5.2 için BPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) BPSO yönteminin yapay sinir ağı çözümü $(h = 0.01)$



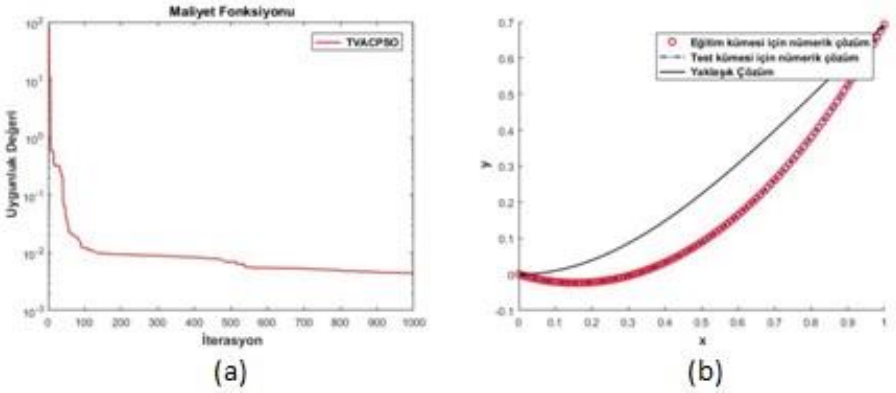
Şekil 5.26. (a) Örnek 5.2 için CLPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) CLPSO yönteminin yapay sinir ağı çözümü $(h = 0.01)$



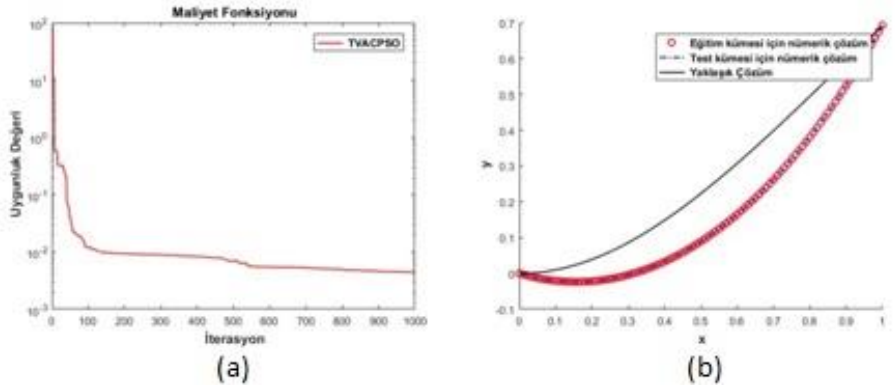
Şekil 5.27. (a) Örnek 5.2 için DWPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) DWPSO yönteminin yapay sinir ağı çözümü ($h = 0.01$)



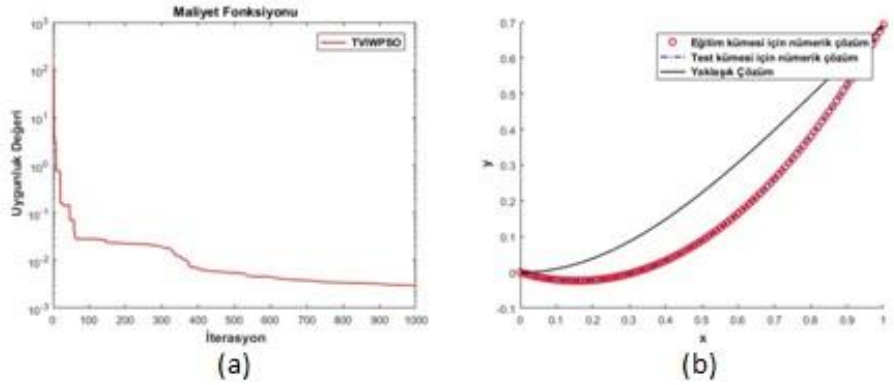
Şekil 5.28. (a) Örnek 5.2 için STVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) STVAC_PSO yönteminin yapay sinir ağı çözümü ($h = 0.01$)



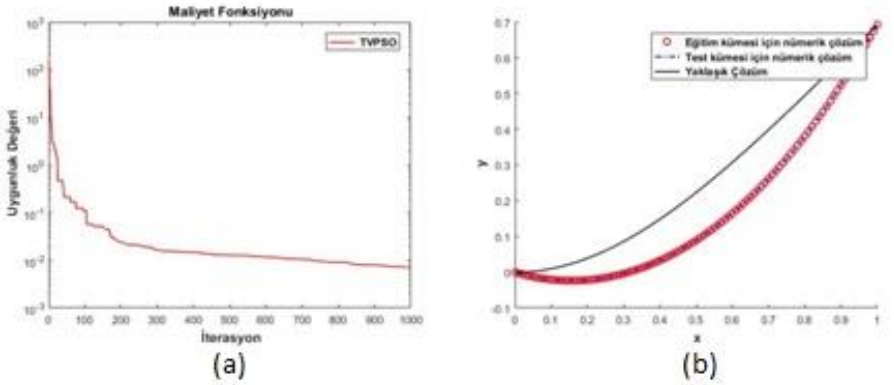
Şekil 5.29. (a) Örnek 5.2 için PSO_C yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) PSO_C yönteminin yapay sinir ağı çözümü $(h = 0.01)$



Şekil 5.30. (a) Örnek 5.2 için TVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVAC_PSO yönteminin yapay sinir ağı çözümü $(h = 0.01)$



Şekil 5.31. (a) Örnek 5.2 için TVIW_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVIW_PSO yönteminin yapay sinir ağı çözümü ($h = 0.01$)



Şekil 5.32. (a) Örnek 5.2 için TVPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVPSO yönteminin yapay sinir ağı çözümü ($h = 0.01$)

Çizelge 5.7. Örnek 5.2 için $h = 0.01$ adım aralığıyla oluşturulan çözüm fonksiyonu için eğitim kümesinin hata miktarları

		Mutlak Hata ($\times 10^{-3}$)							
k	x_k	**BPSO**	**CLPSO**	**DWPSO**	**STVAC_PSO**	**PSO_C**	**TVAC_PSO**	**TVIW_PSO**	**TVPSO**
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.01	3.03	3.03	3.03	3.03	3.03	3.04	3.03	3.03
3	0.02	6.07	6.08	6.07	6.08	6.06	6.08	6.07	6.07
4	0.03	9.11	9.12	9.11	9.12	9.10	9.13	9.11	9.11
5	0.04	12.15	12.17	12.15	12.17	12.14	12.18	12.15	12.16
11	0.10	30.43	30.52	30.42	30.48	30.41	30.55	30.45	30.47
21	0.20	60.87	60.97	60.83	60.85	60.86	61.07	60.86	60.87
31	0.30	90.05	90.01	90.02	89.88	90.06	90.20	90.03	89.94
41	0.40	115.53	115.39	115.47	115.34	115.56	115.55	115.56	115.36
51	0.50	134.14	134.06	134.06	134.17	134.21	134.09	134.25	134.06
61	0.60	142.41	142.50	142.47	142.73	142.47	142.37	142.52	142.50
71	0.70	136.73	136.98	137.03	137.24	136.74	136.76	136.75	136.95
81	0.80	113.51	113.75	113.84	113.94	113.44	113.57	113.40	113.71
91	0.90	69.11	69.21	69.23	69.28	69.05	69.13	69.01	69.18
101	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Çizelge 5.8. Örnek 5.2 için $h = 0.01$ adım aralığıyla oluşturulan çözüm fonksiyonu için test kümesinin hata miktarları

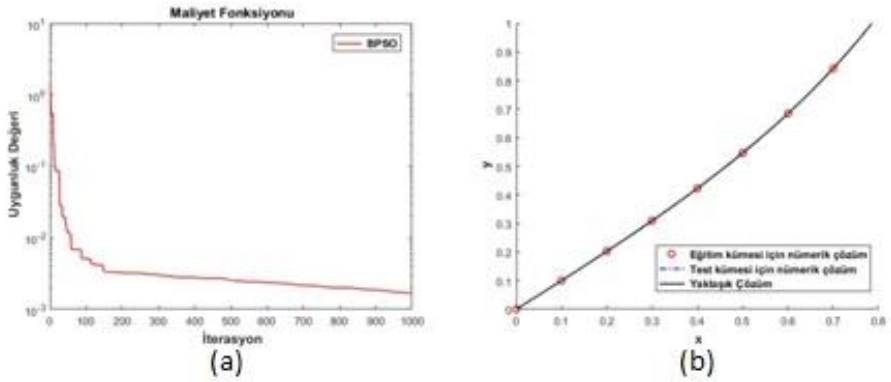
		Mutlak Hata ($\times 10^{-1}$)							
k	x_k	**BPSO**	**CLPSO**	**DWPSO**	**STVAC_PSO**	**PSO_C**	**TVAC_PSO**	**TVIW_PSO**	**TVPSO**
1	0.000	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.005	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
3	0.015	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04
4	0.025	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07
5	0.035	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10
10	0.085	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25
20	0.185	0.56	0.56	0.56	0.56	0.56	0.56	0.56	0.56
30	0.285	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85
40	0.385	1.12	1.11	1.12	1.11	1.12	1.10	1.12	1.11
50	0.485	1.31	1.31	1.31	1.31	1.31	1.31	1.32	1.31
60	0.585	1.41	1.42	1.42	1.42	1.42	1.41	1.42	1.42
70	0.685	1.38	1.38	1.38	1.39	1.38	1.38	1.38	1.38
80	0.785	1.18	1.18	1.18	1.18	1.18	1.18	1.18	1.18
90	0.885	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77
100	0.985	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12
102	1.000	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Örnek 5.3
$$\begin{cases} y'' = 2yy',\ x \in \left[0, \dfrac{\pi}{4}\right] \\ y(0) = 0, \\ y\left(\dfrac{\pi}{4}\right) = 1 \end{cases} \tag{5.5}$$

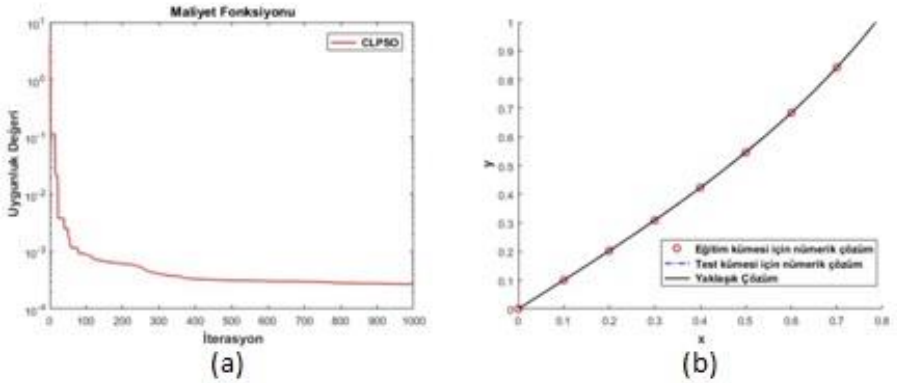
(5.3) denklemi ile verilen 2. mertebeden lineer olmayan Dirichlet sınır değer problemlerinin gerçek çözümü

$$y_e = \tan(x) \tag{5.6}$$

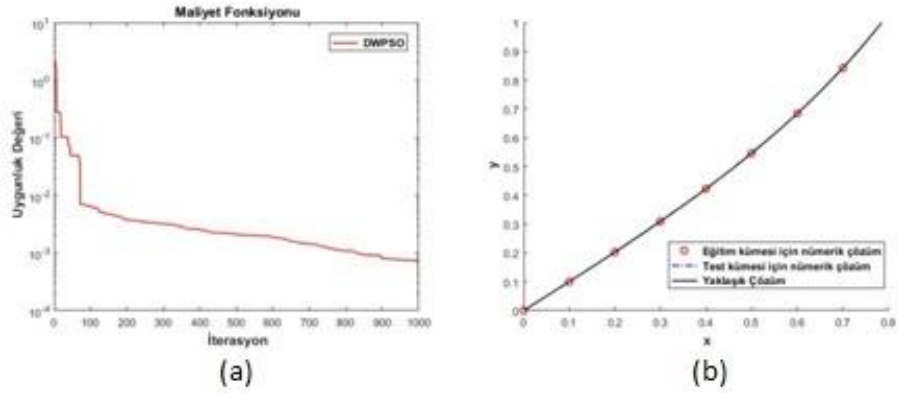
(5.6) denklemi ile verilmiştir. (5.5) denkleminin $h = 0.1$ için tezde geçen PSO varyasyonlarından elde edilen nümerik çözümler eğitim ve test kümeleri için Çizelge 5.9 ve Çizelge 5.10'da sunulmuştur. Şekil 5.33-Şekil 5.40 (5.5) denkleminin PSO yöntemleriyle elde edilen maliyet değerindeki iterasyona bağlı değişimini gösterir ve denklemin çözümünü verir.



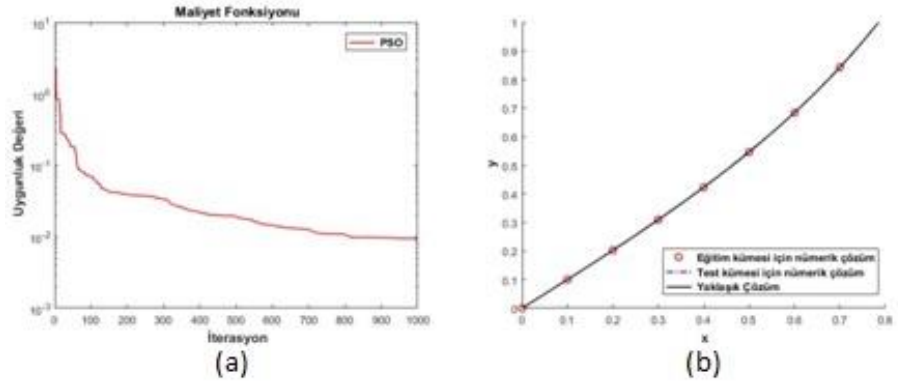
Şekil 5.33. (a) Örnek 5.3 için BPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) BPSO yönteminin yapay sinir ağı çözümü $(h = 0.1)$



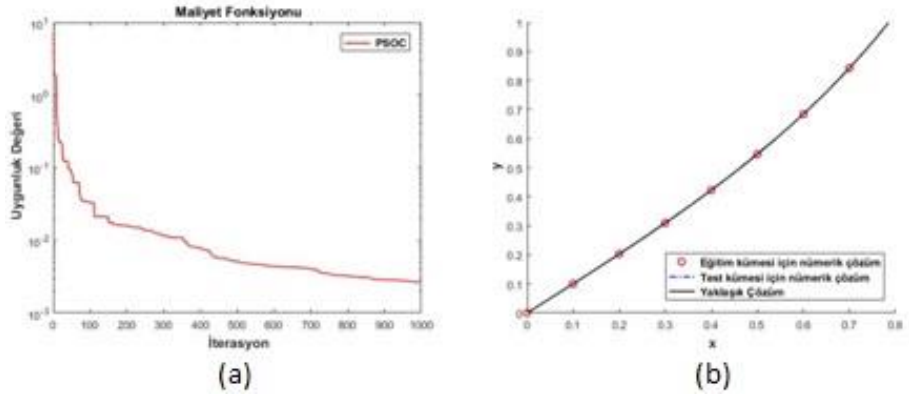
Şekil 5.34. (a) Örnek 5.3 için CLPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) CLPSO yönteminin yapay sinir ağı çözümü ($h = 0.1$)



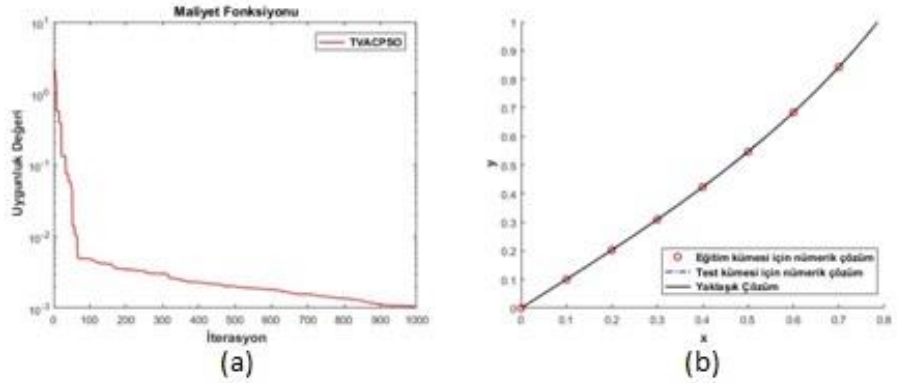
Şekil 5.35. (a) Örnek 5.3 için DWPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) DWPSO yönteminin yapay sinir ağı çözümü ($h = 0.1$)



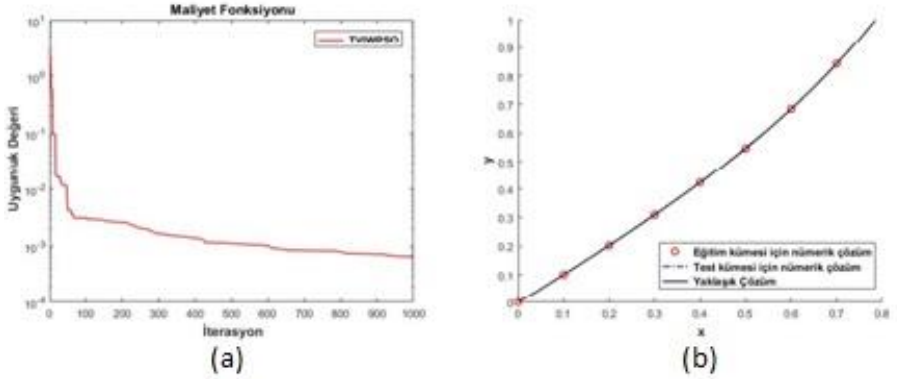
Şekil 5.36. (a) Örnek 5.3 için STVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) STVAC_PSO yönteminin yapay sinir ağı çözümü ($h = 0.1$)



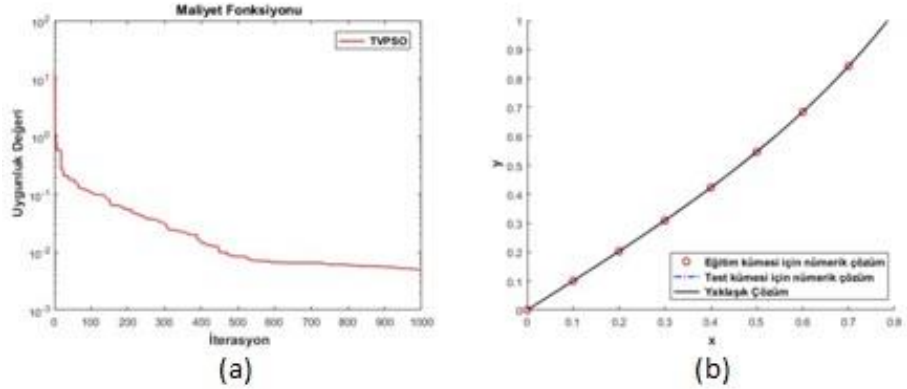
Şekil 5.37. (a) Örnek 5.3 için PSO_C yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) PSO_C yönteminin yapay sinir ağı çözümü ($h = 0.1$)



Şekil 5.38. (a) Örnek 5.3 için TVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVAC_PSO yönteminin yapay sinir ağı çözümü $(h = 0.1)$



Şekil 5.39. (a) Örnek 5.3 için TVIW_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVIW_PSO yönteminin yapay sinir ağı çözümü $(h = 0.1)$



Şekil 5.40. (a) Örnek 5.3 için TVPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVPSO yönteminin yapay sinir ağı çözümü ($h = 0.1$)

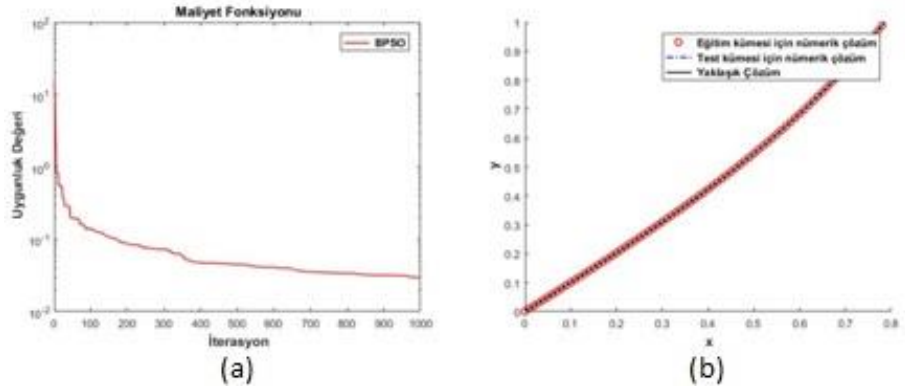
Çizelge 5.9 Örnek 5.3 için $h = 0.1$ adım aralığıyla oluşturulan çözüm fonksiyonu için eğitim kümesinin hata miktarları

		Mutlak Hata ($\times 10^{-4}$)							
k	x_k	**BPSO**	**CLPSO**	**DWPSO**	**STVAC_PSO**	**PSO_C**	**TVAC_PSO**	**TVIW_PSO**	**TVPSO**
1	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.1	0.48	0.11	0.66	0.51	0.17	0.43	0.71	0.76
3	0.2	0.65	0.13	2.01	0.002	1.50	0.11	0.11	0.60
4	0.3	2.08	0.47	3.57	0.96	3.04	0.10	1.24	1.45
5	0.4	1.87	0.88	0.96	1.14	2.26	0.67	1.82	1.55
6	0.5	0.05	0.39	3.37	0.22	0.41	0.74	1.04	0.71
7	0.6	0.87	0.22	3.79	0.50	0.67	0.12	0.14	1.68
8	0.7	1.13	0.50	1.45	0.45	0.95	1.49	0.81	2.47

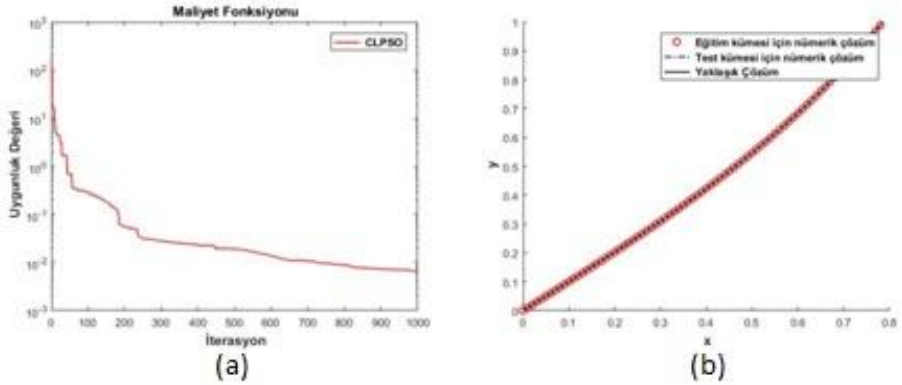
Çizelge 5.10. Örnek 5.3 için $h = 0.1$ adım aralığıyla oluşturulan çözüm fonksiyonu için test kümesinin hata miktarları

		Mutlak Hata ($\times 10^{-4}$)							
k	x_k	**BPSO**	**CLPSO**	**DWPSO**	**STVAC_PSO**	**PSO_C**	**TVAC_PSO**	**TVIW_PSO**	**TVPSO**
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.05	0.40	0.16	0.79	0.34	0.31	0.27	0.44	0.63
3	0.15	0.08	0.008	0.46	0.38	0.49	0.36	0.59	0.61
4	0.25	1.47	0.16	3.24	0.51	2.46	0.11	0.56	0.93
5	0.35	2.24	0.76	2.76	1.20	3.00	0.22	1.76	1.79
6	0.40	1.87	0.88	0.96	1.14	2.26	0.67	1.82	1.55
7	0.45	1.05	0.74	1.32	0.77	1.00	0.92	1.57	0.61
8	0.55	0.73	0.006	4.39	0.28	1.47	0.23	0.47	1.75
9	0.60	0.87	0.22	3.79	0.50	1.67	0.12	0.14	1.68
10	0.65	0.16	0.04	1.54	0.23	0.73	0.27	0.28	0.06
11	0.70	1.13	0.50	1.45	0.45	0.95	1.49	0.81	2.47
12	0.75	1.67	0.81	2.70	0.85	1.80	2.08	1.03	3.32
13	0.78	0.58	0.67	0.93	0.59	0.85	0.16	0.46	0.93

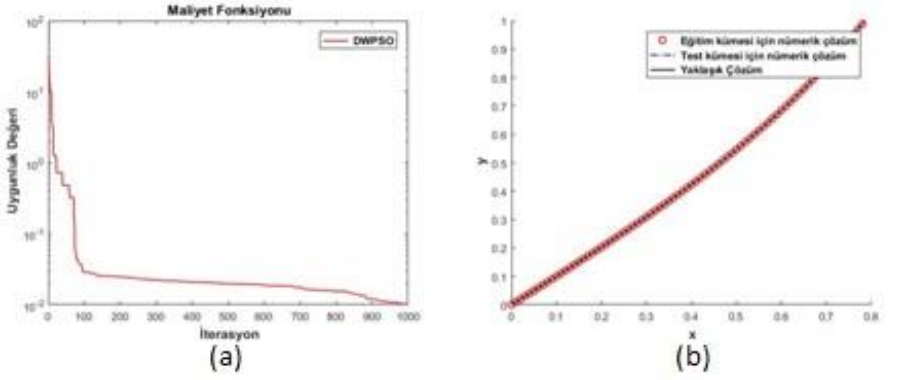
(5.5) denkleminin $h = 0.01$ için tezde geçen PSO varyasyonlarından elde edilen nümerik çözümler Çizelge 5.11 ve Çizelge 5.12'de verilmiş olup yöntemlerden elde edilen çözümler sırasıyla Şekil 5.41-Şekil 5.48 ile gösterilmiştir.



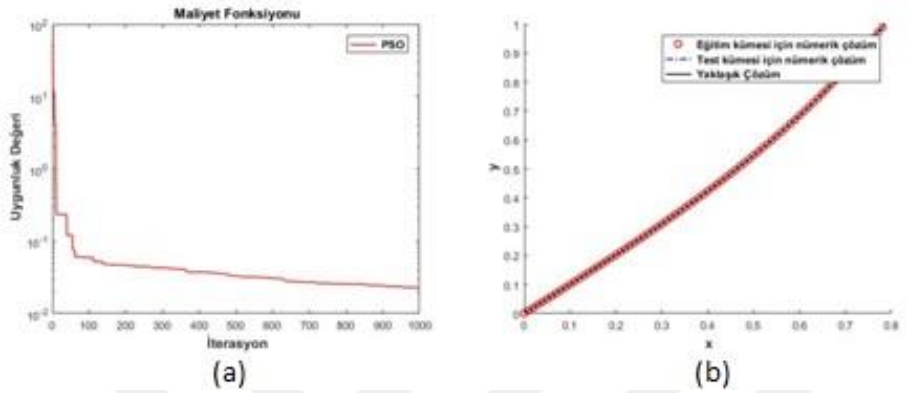
Şekil 5.41. (a) Örnek 5.3 için BPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) BPSO yönteminin yapay sinir ağı çözümü $(h = 0.01)$



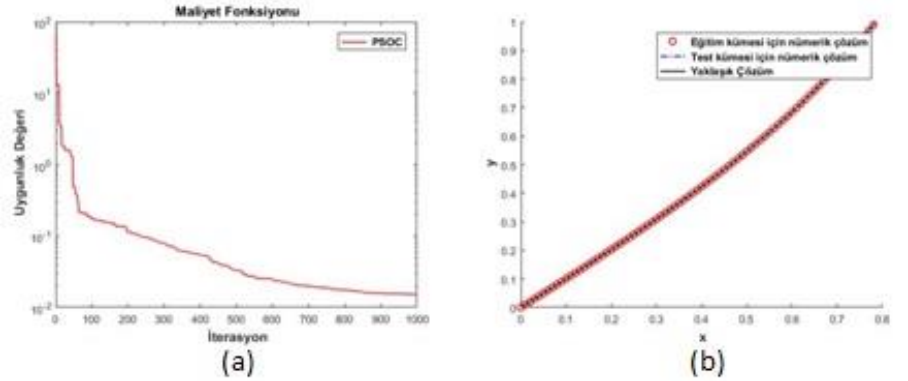
Şekil 5.42. (a) Örnek 5.3 için CLPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) CLPSO yönteminin yapay sinir ağı çözümü $(h = 0.01)$



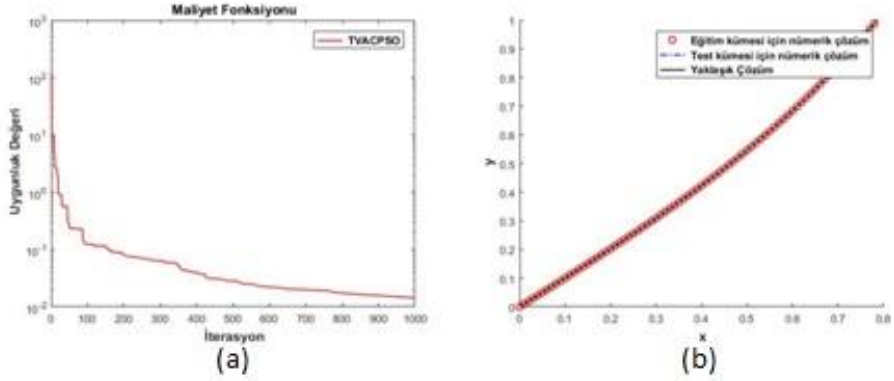
Şekil 5.43. (a) Örnek 5.3 için DWPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) DWPSO yönteminin yapay sinir ağı çözümü $(h = 0.01)$



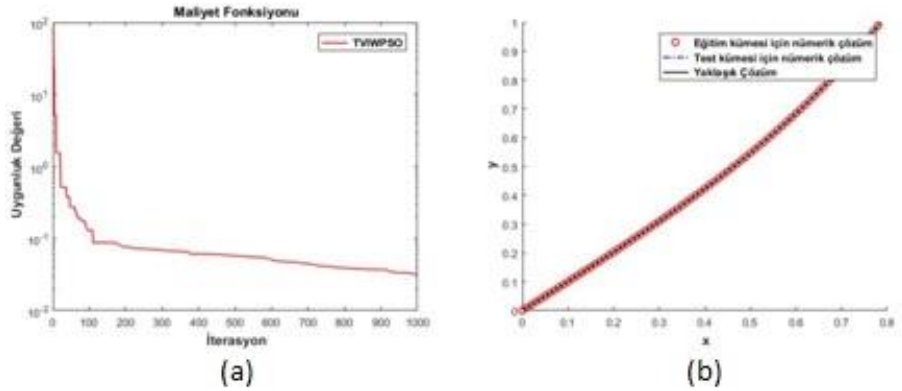
Şekil 5.44. (a) Örnek 5.3 için STVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) STVAC_PSO yönteminin yapay sinir ağı çözümü $(h = 0.01)$



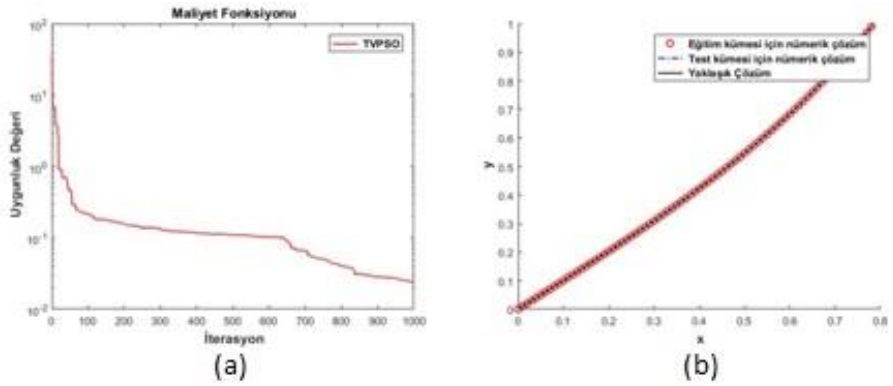
Şekil 5.45. (a) Örnek 5.3 için PSO_C yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) PSO_C yönteminin yapay sinir ağı çözümü $(h = 0.01)$



Şekil 5.46. (a) Örnek 5.3 için TVAC_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVAC_PSO yönteminin yapay sinir ağı çözümü ($h = 0.01$)



Şekil 5.47. (a) Örnek 5.3 için TVIW_PSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVIW_PSO yönteminin yapay sinir ağı çözümü ($h = 0.01$)



Şekil 5.48. (a) Örnek 5.3 için TVPSO yöntemiyle oluşturulan maliyet fonksiyonunun iterasyona bağlı uygunluk değeri grafiği (b) TVPSO yönteminin yapay sinir ağı çözümü ($h = 0.01$)

Çizelge 5.11. Örnek 5.3 için $h = 0.01$ adım aralığıyla oluşturulan çözüm fonksiyonu için eğitim kümesinin hata miktarları

		Mutlak Hata ($\times 10^{-4}$)							
k	x_k	**BPSO**	**CLPSO**	**DWPSO**	**STVAC_PSO**	**PSO_C**	**TVAC_PSO**	**TVIW_PSO**	**TVPSO**
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.01	0.12	0.05	0.01	0.06	0.01	0.08	0.04	0.06
3	0.02	0.26	0.13	0.03	0.16	0.03	0.19	0.10	0.17
4	0.03	0.40	0.23	0.04	0.30	0.05	0.32	0.18	0.29
5	0.04	0.52	0.34	0.05	0.47	0.06	0.45	0.27	0.40
11	0.10	0.86	1.00	0.20	1.55	0.04	1.13	0.71	0.30
21	0.20	1.27	0.75	0.17	1.80	0.18	0.98	0.20	1.41
31	0.30	3.18	0.82	0.33	0.14	0.13	0.39	1.69	0.82
41	0.40	4.67	1.59	0.21	2.07	0.02	1.17	2.70	1.28
51	0.50	3.32	0.24	0.23	1.86	0.55	0.24	1.13	1.67
61	0.60	0.13	1.89	0.45	0.09	0.20	1.47	1.50	0.13
71	0.70	0.89	1.68	0.94	0.85	0.66	1.45	1.49	0.64
79	0.78	0.03	0.01	0.001	0.004	0.03	0.03	0.02	0.01

Çizelge 5.12. Örnek 5.3 için $h = 0.01$adım aralığıyla oluşturulan çözüm fonksiyonu için test kümesinin hata miktarları

		Mutlak Hata ($\times 10^{-4}$)							
k	x_k	**BPSO**	**CLPSO**	**DWPSO**	**STVAC_PSO**	**PSO_C**	**TVAC_PSO**	**TVIW_PSO**	**TVPSO**
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.005	0.62	0.02	0.009	0.02	0.006	0.04	0.01	0.02
3	0.015	0.19	0.08	0.02	0.11	0.02	0.14	0.72	0.11
4	0.025	0.33	0.18	0.03	0.23	0.04	0.26	0.14	0.23
5	0.035	0.46	0.28	0.05	0.38	0.06	0.38	0.22	0.34
10	0.085	0.83	0.86	0.16	1.38	0.01	1.00	0.63	0.46
20	0.185	1.12	0.92	0.22	1.92	0.21	1.11	0.40	1.23
30	0.285	2.84	0.58	0.27	0.21	0.11	0.17	1.40	1.09
40	0.385	4.60	1.60	0.28	1.88	0.02	1.16	2.70	1.02
50	0.475	3.94	0.76	0.19	2.11	0.38	0.63	1.74	1.82
60	0.575	0.89	1.45	0.19	0.42	0.47	1.10	0.95	0.56
70	0.675	1.01	2.07	1.03	0.93	0.64	1.73	1.87	0.66
80	0.765	0.002	0.19	0.11	0.08	0.005	0.23	0.07	0.11
83	0.785	0.58	0.67	0.93	0.85	0.46	0.46	0.36	0.33

6. TARTIŞMA VE SONUÇ

Bu çalışmada ikinci mertebeden adi diferansiyel denklemler için Dirichlet Sınır Değer Problemlerinin nümerik çözümleri ileri beslemeli yapay sinir ağı modelleriyle elde edilmiştir. Yapay sinir ağı, Parçacık Sürü Optimizasyonunun sekiz farklı varyasyonu ile eğitilmiş ve lineer ile lineer olmayan örnekler üzerinde elde edilen sonuçlar karşılaştırılmıştır.

Nümerik çözümler elde edilirken çözümün arandığı kapalı aralığın $h = 0.1$ ve $h = 0.01$ sabit adım uzunlukları kullanılarak parçalanışları oluşturulmuştur ve bu parçalanışlar yapay sinir ağının birer girdisi olarak eğitim kümelerini oluşturmuştur. Ek olarak yapay sinir ağının eğitiminde maksimum devir sayısı 1000 olarak belirlenmiştir. Elde edilen nümerik çözümler denklemin analitik çözümleri ile karşılaştırılmıştır. Sonuç olarak, üç farklı örnek üzerinde iki farklı eğitim kümesi için oluşturulan yapay sinir ağları sekiz farklı metotla onar kez eğitilmiştir. Eğitim sonucunda elde edilen parametre değerleri kullanılarak yapay sinir ağının test işlemi gerçekleştirilmiş, sonuçlar mutlak hata, ortalama karesel hata ve çözümlerin standart sapmaları açısından değerlendirilmiştir.

Toplamda gerçekleştirilen 480 farklı deneysel çalışma sonucunda elde edilen bulgulara göre PSO'nun nümerik çözümleri elde etmede yetkin olduğu görülmüştür. Sonuçlar göstermektedir ki kullanılan PSO türünün çözümü elde etmede ayırt edici olmadığı, her bir varyasyonun çözüme yakınsadığı görülmüştür. Farklı noktalarda farklı PSO türlerinin daha iyi sonuç verdiği, ancak tüm noktalarda en iyi çözümün elde edildiği tek bir yöntemin belirlenemediği ve bu hususta bir genelleme yapılamadığı fark edilmiştir.

Ulaşılan çözümlerin tutarlılığının incelenebilmesi için klasik nümerik metotlarla elde edilen nümerik çözümlerle karşılaştırılması gereklidir. Bu tezde nümerik çözümler sadece analitik çözümlerle karşılaştırılmıştır.

KAYNAKLAR

[1] Goldberg, D. E., (1989). Genetic Algorithms in Search Optimization and Machine Learning, Addison Wesley, ISBN 0201157675.

[2] Lim, T. Y., (2014). Structured population genetic algorithms: a literature survey, **Artif Intell Rev**, 41, 385-399.

[3] Erdogmus, P., (2010). Particle Swarm Optimization Performance On Special Linear Programming Problems, **Scientific Research and Essays**, 5(12), 1506-1518.

[4] Mehrabian, R., Lucas, C., (2006). A novel numerical optimization algorithm inspired from weed colonization. **Ecological Informatics**, 1(4), 355-366.

[5] Rajabioun, R., (2011). Cuckoo Optimization Algorithm, **Applied Soft Computing**, 11(8), 5508-5518.

[6] Mishra, S., Shaw, K., Mishra, D., (2012). A New Meta-heuristic Bat Inspired Classification Approach for Microarray Data, **Procedia Technology**, 4, 802-806.

[7] Yang, X. S, Hosseini,S. S. S, Gandomi, A. H., (2012). Firefly Algorithm for solving nonconvex economic dispatch problems with valve loading effect, **Applied Soft Computing**, 12(3), 1180-1186.

[8] Engelbrecht A. P, Computational Intelligence: An Introduction, Chichester, England: John Wiley & Sons, 2007.

[9] Yang, C. -H, C.-J. Hsiao and L.-Y. Chuang, (2010). Linearly Decreasing Weight Particle Swarm Optimization with Accelerated Strategy for Data Clustering. **IAENG International Journal of Computer Science**, 37(3), 1-8.

[10] Eberhart R. C, and Y. Shi, Particle Swarm Optimization: Developments, Applications and Resources, Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546), Seoul, 2001, 81-86 vol. 1. doi: 10.1109/CEC.2001.934374

[11] J. Polprasert and W. Ongsakul, Chaotic based PSO with Time-Varying Acceleration Coefficients for Security Constrained Optimal Power Flow Problem, International Conference and Utility Exhibition 2014 on Green Energy for Sustainable Development, 2014.

[12] A. Sengupta and V. K. Mishra, Time Varying vs. Fixed Acceleration Coefficient PSO Driven Exploration during High Level Synthesis: Performance and Quality Assessment, 2014 International Conference on Information Technology, Bhubaneswar, 2014, 281-286. doi: 10.1109/ICIT.2014.16

[13] A. Ratnaweera, S. K. Halgamuge and H. C. Watson, Self-Organizing Hierarchical Particle Swarm Optimizer With Time-Varying Acceleration Coefficients, **IEEE Transactions on Evolutionary Computation**, 8(3), 240-255, June 2004. doi: 10.1109/TEVC.2004.826071

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Gülsüm İşman

Doğum Yeri Ve Tarihi : Söke, 29.07.1993

EĞİTİM DURUMU

Lisans Öğrenimi : Yıldız Teknik Üniversitesi

Yüksek Lisans Öğrenimi : Adnan Menderes Üniversitesi

Yabancı Diller : İngilizce

BİLİMSEL FAALİYETLERİ

A) Bildiriler

Günel, K., İşman, G. and Kocakula, M., (2018). Numerical Solutions of ODEs with Dirichlet Boundary Conditions via Recurrent Neural Networks, International Conference on Applied Mathematics in Engineering, June 27-29,2018, Balıkesir, Turkey.

İLETİŞİM

E-Posta Adresi : gulsumisman@gmail.com

Tarih : 10/07/2018