PASİF KATILIMLI PARÇACIK SÜRÜSÜ OPTİMİZASYONU YÖNTEMİ İLE OPTİMUM YAPISAL TASARIM YÜKSEK LİSANS TEZİ

Hacer ARIOL

DANIŞMAN
Doç. Dr. Ömer SOYKASAP

MAKİNE MÜHENDİSLİĞİ ANABİLİM DALI

Aralık 2010

AFYON KOCATEPE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

PASİF KATILIMLI PARÇACIK SÜRÜSÜ OPTİMİZASYONU YÖNTEMİ İLE OPTİMUM YAPISAL TASARIM

Hacer ARIOL

DANIŞMAN
Doç. Dr. Ömer SOYKASAP

MAKİNE MÜHENDİSLİĞİ ANABİLİM DALI

Aralık 2010

ONAY SAYFASI

Doç. Dr. Ömer SOYKASAP danışmanlığında
Hacer ARIOL tarafından hazırlanan
"PASİF KATILIMLI PARÇACIK SÜRÜSÜ OPTİMİZASYONU YÖNTEMİ İLE OPTİMUM YAPISAL TASARIM"
başlıklı bu çalışma lisansüstü eğitim ve öğretim yönetmeliğinin ilgili maddeleri uyarınca
31/12/2010
tarihinde aşağıdaki jüri tarafından
Makine Mühendisliği Anabilim Dalında
Yüksek lisans tezi olarak oybirliği/oy çokluğu ile kabul edilmiştir.

Ünvanı, Adı, SOYADI İmza

Başkan : Doç. Dr. Ömer SOYKASAP

Üye : Doç. Dr. Şükrü TAKTAK

Üye : Doç. Dr. Ali ELEREN

Afyonkarahisar Kocatepe Üniversitesi
Fen Bilimleri Enstitüsü Yönetin Kurulu'nun
...../...../..... tarih ve
…… sayılı kararıyla onaylanmıştır.

Doç. Dr. Rıdvan ÜNAL
Enstitü Müdürü

İÇİNDEKİLER

ÖZET

Yüksek Lisans Tezi

PASİF KATILIMLI PARÇACIK SÜRÜSÜ OPTİMİZASYONU YÖNTEMİ İLE OPTİMUM YAPISAL TASARIM

Hacer ARIOL

Afyon Kocatepe Üniversitesi

Fen Bilimleri Enstitüsü

Makine Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Ömer SOYKASAP

Günümüzde mühendislik ve diğer alanlarda daha iyi, kaliteli ürün ve hizmet sunumu hedeflenmektedir. Bu amacın gerçekleştirilmesi için kullanılan malzemelerde, ürünlerin yapısal ve görsel tasarımında optimizasyon tekniklerinin kullanımı kaçınılmazdır.

Pasif Katılımlı Parçacık Sürüsü algoritması hayvanlar arasındaki sosyal etkileşimden esinlenilen bir optimizasyon tekniğidir. PSOPC algoritması içinde modellenen sosyal etkileşim arama uzayının en anlamlı bölgesine doğru hareket eden ve parçacıklar olarak adlandırılan bir bireyler popülasyonuna rehberlik etmek için kullanılır.

Bu çalışmada sezgisel yöntemlerden Pasif Katılımlı Parçacık Sürüsü Optimizasyon Tekniği incelenmiştir. Tekniğin kullanılabilirliğinin testi için literatürde algoritmaların performans değerlendirmelerinde kullanılan üç özel fonksiyon kullanılmıştır. Bu fonksiyonların optimum değerleri, PSOPC'nin ilk hali olan PSO, Genetik Algoritma ve PSOPC algoritmalarında aranmıştır. Yapılan bu çalışmalar doğrultusunda algoritmanın yapısal tasarımda kullanılabileceği tespit edilmiş ve yapısal tasarıma dair bir kiriş bir de yay problemi ele alınarak ve bu problemlere çözüm aranmıştır.

2010, 63 sayfa

Anahtar kelimeler: PSOPC (Pasif Katılımlı Parçacık Sürü Optimizasyonu), PSO, Genetik Algoritma, Sürü Zekası

ABSTRACT

M.Sc. Thesis

OPTIMUM STRUCTURAL DESIGN BY PARTICLE SWARM OPTIMIZATION WITH PASSIVE CONGREGATION

Hacer ARIOL

Afyon Kocatepe University

Graduate School of Natural and Applied Sciences

Department of Mechanical Education

Supervisor: Assoc. Prof. Ömer SOYKASAP

Today it is aimed to supply better and more qualified product and services in engineering and other fields. To realize this aim, it is needed to use optimization techniques in products' structure and visional design in used materials.

In this study, particle swarm optimization with passive congregation is examined using one of the heuristic methods. Particle swarm optimization with passive congregation algorithm is an optimization tecnique that is inspired by the social interaction of animals. Particle swarm optimization with passive congregation is used for guidance to population of algorithm modeled by interaction within search space.

For evaluation of PSOPC, three special functions for are used performans test in literature. These functions' optimum values are searched with, PSOPC and genetic algorithm. By directions of these studies, it is proved that algorithm can be used for structural design. A beam and a spring problem for structural optimization are considered.

2010, 63 pages

Keywords: PSOPC (Particle Swarm Optimization with Passive Congregation), PSO, Genetic Algorithm, Swarm Intelligence

TEŞEKKÜR

Yapılan tez çalışması sırasında danışmanlığımı yürüten hocam Doç. Dr. Ömer SOYKASAP ve tüm hayatım boyunca maddi ve manevi desteklerini hep yanımda hissettiğim aileme sonsuz teşekkür ederim.

Hacer ARIOL

Afyonkarahisar, Aralık 2010

SİMGELER VE KISALTMALAR DİZİNİ

1. Simgeler

ω	Atalet Çarpanı
c_1	Sabit Çarpan
c_2	Sabit Çarpan
r_1	0 ile 1 arasında rastsal sayı
r_2	0 ile 1 arasında rastsal sayı
$Pbest_i^k$	i. parçacığın k. iterasyonuna kadar aldığı en iyi koordinat
$Gbest^k$	Populasyondaki parçacıkların o ana kadar aldığı en iyi koordinat
X_i^k	i. parçacığının k iterasyonundaki koordinatı
X_i^{k+1}	i. parçacığının k+1 iterasyonundaki koordinatı
V_i^k	i. parçacığın k iterasyonu için hesaplanan hız vektörü
V_i^{k+1}	i. parçacığın k+1 iterasyonu için hesaplanan hız vektörü
F_1	Eksenel Kuvvet
Fcr	Burkulma Yükü
H	Kirişin iki ucu arasındaki yükseklik farkı
X	Kirişin duvardaki ucunun yer değiştirmesi
L	Kiriş Uzunluğu
A	Kiriş Kesiti
I	Atalet Moment
E	Elastisite Modülü
P	Uygulanacak Yük
l_1	1.Yayın uzunluğu
l_2	2.Yayın uzunluğu
K_1	Yay katsayısı
K_2	Yay katsayısı
P_1	Yatayda uygulanan kuvvet
P_2	Dikeyde uygulanan kuvvet
X_1	Yataydaki yer değiştirme
X_2	Dikeydeki yer değiştirme

2. Kısaltmalar

PSO	Parçacık Sürüsü Optimizasyonu
PSOPC	Pasif Katılımlı Parçacık Sürüsü Optimizasyonu
GA	Genetik Algoritma

ŞEKİLLER DİZİNİ

RESİMLER DİZİNİ

ÇİZELGELER DİZİNİ

1. GİRİŞ

1.1 Genel Bilgiler

Uzun zaman önce, insanlar doğada bulunan hayvan ve böcek sürülerinin davranışlarını keşfettiler. Kus sürülerinin havada süzülmesi ve farklı sekil alması karıncaların yiyecek araştırması, balık sürülerinin beraberce yüzmesi ve kaçışması bu sürü davranışlarından sadece birkaçıdır. Son yıllarda ise biyologlar ve bilgisayar uzmanları "Yapay Yasam" alanı kapsamı altında bu sürülerin davranışlarının nasıl modellenebileceği aralarındaki iletişimin mantığını üzerinde çalışmalar yapmaktadırlar (İnt. Kyn. 1).

Resim 1.1 Kuş Sürüleri (İnt. Kyn. 3)

Günümüzde biyolojik sistemlerden esinlenilerek ortaya çıkarılmış birçok yöntem hesaplama problemlerinin çözümünde kullanılmaktadır. Örneğin yapay sinir ağları insan beyninin basitleştirilmiş bir modelidir, genetik algoritma ise insan evriminden esinlenilerek ortaya çıkarılmıştır. Biyolojik sistemlerin başka bir çeşidi olan sosyal sistemler, özellikle bireyin çevresiyle ve diğer bireylerle olan etkileşimini ve kolektif davranışlarını incelemektedir. Bu davranışlar sürü zekâsı olarak adlandırılmaktadırlar (Tamer ve Karakuzu 2006).

“Sürü zekası” (Swarm Intelligence) adı verilen bu yaklaşımların optimizasyon problemleri, robotbilim ve askeri uygulamalarda basarılar göstermeleri bu konu üzerindeki çalışmaları arttırmıştır. Sürü zekası, özerk yapıdaki basit bireyler grubunun kolektif bir zeka geliştirmesidir. Bu ise “Stigmergy”, yani "ortam"daki etmenlerin ortama müdahale ederek iletişim kurmaları ve birbirlerinin hareketlerini düzenlemeleri ve “Self-Organization” (Kendinden organizasyon) denen iki mekanizma üzerine kuruludur. Stigmergy vasıtasıyla iletişim, bireyler yaptıkları islerle ortamda değişikliğe sebep olarak, sağlanırken, kendinden organizasyon yardımıyla önceden yapılmış herhangi bir plan olmadan sonuç üretebilmelerini, esnek ve sağlam, merkezi bir yönetim birimi olmadan yapılanmalarını sağlar (İnt. Kyn. 1).

Resim 1.2 Balık Sürüleri (İnt. Kyn. 3)

Sürü sistemlerini ortaya getiren üyeler hem çevresiyle hem diğer üyelerle etkileşimde bulunurlar. Bu üyeler çok basit kuralları takip ederler. Ortada üyelerin davranışlarını denetleyecek bir kural olmamasına rağmen, bu üyelerin davranışları zeki bir sistem ortaya çıkarır (İnt. Kyn. 8).

Doğadaki bir sürü örneği arılar incelenecek olursa, arıların koloni bazlı çalışması fark edilecektir. Bir arı kolonisi kovanlarının sıcaklığını düzenleyebilir. Yemek seçimini besinin kovana uzaklığı ve kalitesiyle kıyaslayarak yapar. Kolonide tamamen bir işbölümü vardır. Yabanarılarında da durum çok farklı değildir. Kâğıt hamurundan

yığınlar yapabilir, yuvalarını koruyucu tabakalarla çevirebilir, yuvalarına ana kapı inşa edebilirler. Beyaz karıncalar aynı şekilde yuvalarına üreme odaları yaparlar. Ayrıca yuvalarını havalandırma kanallarıyla çevirirler. Karıncalar salgılarını besin rotalarına bırakarak besinlere giden yolu öbür karıncalara kolaylık için işaretlerler. Yapraklara uzanmak için vücutlarını birleştirerek zincir görevi görürler. Usta ve çırak arılar arasında işbölümü yaparlar.

Resim 1.3 Karınca Sürüleri (İnt. Kyn. 8)

Canlılar incelendiğinde bireysel bazda düşük kabiliyet, koloni bazında yüksek koordinasyon sayesinde bir bireyin tek başına başarmasının mümkün olmadığı büyük eserler ve yaşamalarını kolaylaştıran çözümler ortaya çıkabilmektedir (İnt. Kyn. 8).

Bir sürü sisteminde, üyelerden biri işini yanlış yapmaya başlasa bile sürü çok geniş olduğundan hatası öbürleri tarafından telafi edilecektir. Bu da sistemin iyiye doğru güncelleneceği anlamına gelmektedir.

Bunun yanı sıra sistemde merkezcil bir koordinasyon olmadığından, sistemin bir kısmının yok edilmesi koordinasyonun yok olmasına sebebiyet vermeyecektir. Sürü sistemlerde koordinasyon tüm sistemin ortak çalışması sonucu beklenmedik bir şekilde ortaya çıkan bir özelliktir (İnt. Kyn. 8).

Sürü zekâsından yola çıkarak Karınca Kolonisi Optimizasyonu ve Parçacık Sürü Optimizasyonu (Particle Swarm Optimization) olmak üzere iki popular metot mevcuttur. Bu metotlardan PSO'ya dair Literatür Araştırması aşağıda bulunmaktadır.

1.2 Literatür Araştırması

Mühendislik bilimi, analiz, dizayn, fabrikasyon, satış, araştırma ve geliştirme gibi birçok faali-yet alanını içermektedir. Hızla gelişmekte olan dünyada, sadece çalışan bir sistem geliştirmek artık tatmin edici olmaktan uzaklaşmaktadır. Önemli olan "en iyi" sistemi geliştirmektir. "En iyi" kavramına, gördüğü işlevlere göre en hafif, en ucuza mal olmuş, en verimli, en hızlı, çok fonksiyonlu, dayanıklı vb. anlamlar verilebilir. Böyle bir sistemin dizaynı, "optimizasyon problemi" olarak formüle edilip çözümlenebilir. En basit tabiri ile optimizasyon problemi, "en iyi"den kastedilen manaları hedef olarak alıp, matematiksel fonksiyonla temsil ettirilerek, belirtilen sınırlar dahilinde bu fonksiyonun minimum bazen maksimum değerinin bulunmasıdır. Böyle bir problemin üç önemli bileşeni vardır. Bunlar; değişkenler, sınırlayıcılar ve amaç fonksiyonudur (Esen ve Ülker 2005).

Ürün geliştirme sürecinde maliyet, malzeme, ağırlık, dayanım,vb. özelliklerin optimize edilmesi gerekmektedir. Yapısal analiz veya stres analizi bu gibi durumlar için büyük önem arz etmektedir. Yapısal analiz aynı zamanda deneysel testlere göre çok daha düşük maliyetlere sahiptir. Tasarım mühendisi, yaptığı tasarımlardaki değişkenlerin tasarıma etkisini en kısa yoldan yapısal analiz yöntemiyle değerlendirebilir. Bu sayede zamandan kazanıldığı gibi, maliyetlerin de minimuma çekilebilme şansına sahip olunacaktır. Buradaki amaç maddeler halinde aşağıda sunulmaktadır (İnt. Kyn. 6).

- Geliştirilen ürünün maliyet ve ürün özellikleri bakımından optimize edilmesi
- Üretim aşamasındaki riskleri minimuma indirilmesi
- İleride doğabilecek sorunları önceden fark edilebilmesi

Parçacık Sürüsü (particle swarm) Optimizasyonu (PSO), 1995 yılında J.Kennedy ve R.C.Eberhart tarafından; kus sürülerinin davranışlarından esinlenilerek geliştirilmiş popülasyon tabanlı stokastik optimizasyon tekniğidir. Doğrusal olmayan problemlerin çözümü için tasarlanmıştır. Çok parametreli ve çok değişkenli optimizasyon problemlerine çözüm bulmak için kullanılmaktadır. PSO, genetik algoritmalar gibi evrimsel hesaplama teknikleriyle bir çok benzerlik gösterir. Sistem rasgele çözümler içeren bir popülasyonla başlatılır ve nesilleri güncelleyerek en optimum çözümü araştırır. PSO da parçacık olarak adlandırılan olası muhtemel çözümler, o andaki optimum parçacığı izleyerek problem uzayında dolaşırlar. PSO'nun klasik optimizasyon tekniklerinden en önemli farklılığı türev bilgisine ihtiyaç duymamasıdır. PSO'yu uygulamak, algoritmasında ayarlanması gereken parametre sayısının az olması sebebiyle oldukça basittir. PSO; fonksiyon optimizasyonu, bulanık sistem kontrolü, yapay sinir ağı eğitimi gibi birçok alanda basarıyla uygulanabilmektedir (Tamer ve Karakuzu 2006).

PSO kus sürülerinin davranışlarının bir benzetimidir. Kuşların uzayda, yerini bilmedikleri yiyeceği aramaları, bir probleme çözüm aramaya benzetilir. Kuşlar yiyecek ararken yiyeceğe en yakın olan kuşu takip ederler. Parçacık olarak adlandırılan her tekil çözüm, arama uzayındaki bir kustur. Parçacık hareket ettiğinde, kendi koordinatlarını bir fonksiyona gönderir ve böylece parçacığın uygunluk değeri ölçülmüş olur. (Yani yiyeceğe ne kadar uzaklıkta olduğu ölçülmüş olur.) Bir parçacık, koordinatlarını, hızını (çözüm uzayındaki her boyutta ne kadar hızla ilerlediği), şimdiye kadar elde ettiği en iyi uygunluk değerini ve bu değeri elde ettiği koordinatları hatırlamalıdır. Çözüm uzayındaki her boyuttaki hızının ve yönünün her seferinde nasıl değişeceği, komsularının en iyi koordinatları ve kendi kişisel en iyi koordinatlarının bir birleşimi olacaktır (Tamer ve Karakuzu 2006).

Parçacık Sürü Optimizasyon algoritması popülasyon temelli bir optimizasyon metodudur. Parçacık sürü optimizasyonu (PSO) zor ve karmaşık problemleri çözümünde etkili bir teknik olarak kullanıla gelmektedir. (2) PSO yapay sinir ağ eğitimi ve fonksiyon minimizasyonu gibi bir çok optimizasyon probleminin çözümünde kolaylıkla kullanılmıştır (Aslantaş vd. 2007).

PSO çok boyutlu bir arama uzayında sürü parçacıklarının arama davranışına dayalı iteratif bir yöntemdir. Her bir iterasyonda, tüm parçacıkların hızları ve pozisyonları güncellenir. PSO'da her parçacık problem için bir çözüm adayı olarak değerlendirilebilir (Karakuzu 2007).

PSO optimum ya da optimuma yakın çözüm bulmak için önce her biri çözüm adayı olan parçacıklar oluşturur. Bu bireyler belli sınırlar içerisinde rasgele seçilir. Bireylerin bir araya gelmesiyle çözüm için gerçekleştirilen popülasyon oluşturulur. Parçacık hareket ettiğinde koordinatlarını bir fonksiyona gönderir ve parçacığın uygunluk değeri (optimum çözüme olan uzaklığı) ölçülmüş olur. Parçacığın konum bilgisi (koordinatlarını), hızı (çözüm uzayında ne kadar hızla ilerlediği) ve güncel en iyi uygunluk değeri ile bu değeri elde ettiği koordinatları hafızada tutulmalıdır. Çözüm uzayındaki her boyutta hızının ve yönünün nasıl değişeceği, komşularının en iyi koordinatları ve kendi kişisel en iyi koordinatlarının bir birleşimi olacaktır (Der vd. 2008).

Algoritma problemin çözümü için aday parçacık popülasyonunun (sürünün) rasgele oluşturulması ile başlar. Sürüdeki her bir parçacığın belirlediği çözüm için aday parametrelerle sistem çözümü yapılır. Elde edilen çözümden problemin yapına göre önceden belirlenen bir ölçüt ile parçacığın uygunluğu sayısal olarak belirlenir. Bu işlem sürüdeki tüm parçacıklar için tekrarlanır ve içlerindeki en iyi uygunluk değerine sahip olan parçacık küresel en iyi olarak etiketlenir. İkinci iterasyon için yeni parçacık değerleri belirlenir. İkinci ve sonraki adımlarda her parçacık için yerel en iyi parçacık, önceki adımlardaki en iyi uygunluk değerini alan aynı indisli parçacık olarak belirlenirken, küresel en iyi parçacık ise o iterasyona kadarki tüm parçacıklar içinden en iyi uygunluk değerini veren parçacık olarak etiketlenir (Karakuzu 2007). Bu öğrenilmiş iki en iyi koordinatın yanı sıra rassal seçilen bir parçacığın koordinatları da algoritmanın güncellenmesine katkı sağlar.

Popülasyonlardaki iki boyutlu davranışlar; çevrelerine adapte olabilme, zengin yiyecek kaynakları bulabilme ve avcılardan kaçabilme gibi 'bilgi paylaşma' yaklaşımı

gerektirmektedir. Bilgi paylaşımı sayesinde sürüler, yiyecek ararken ya da avcıdan kaçarken, hedefe en yakın olan sürü elemanını takip ederler ve kendi hızlarını ve konumlarını en başarılı elemana göre güncellerler (Der vd. 2008).

2.MATERYAL VE METOD

2.1 PSO (Parçacık Sürüsü Optimizasyonu)

Genetik Algoritma gibi popülasyon temelli sezgisel bir algoritmadır. Bireyler arasındaki sosyal bilgi paylaşımını geliştirmeyi amaç edinmiştir. Kuş ve balık sürülerinin iki boyutlu hareketlerinden esinlenerek 1995'te Kennedy ve Eberhart tarafından geliştirilmiştir. Evrimsel algoritmalara benzer, her bir aday çözüm parçacıkla (particle), popülasyon ise sürü ile ifade edilir. Her bir aday çözüm (particle) bir sonraki pozisyonunu, hız (velocity) vektörü, kendi tecrübesi (yerel en iyi - pbest) ve sürü tecrübesine (küresel en iyi - gbest) göre ayarlar (Karahan ve Bingül 2009).

PSO, arama uzayında optimum noktayı ararken, optimize edilecek olan fonksiyonun muhtemel çözümlerini barındıran bir popülasyonu kullanması açısından Genetik Algoritma (GA) gibi gelişime dayalı hesaplama teknikleriyle benzerlik göstermektedir. Bununla birlikte, PSO'da her sürü üyesinin arama uzayındaki hareketini belirleyen ve şartlara göre değişebilen bir hızı (pozisyon değişimi) mevcuttur. Ayrıca her üyenin, daha önce ziyaret edilmiş en iyi noktayı tuttuğu bir hafızası da bulunmaktadır. Böylelikle sürü üyesi bir parçacığın hareketi, kendisinin daha önce ziyaret ettiği en iyi noktaya ve belirli bir topolojik komşuluğundaki en iyi bireye doğru olmaktadır (Aslantaş vd. 2007).

Her parçacık, o ana kadar başardığı en iyi çözümlü koordinatlarını problem uzayında takip eder. Bu değer, pbest olarak adlandırılır. PSO tarafından takip edilen diğer bir en iyi değer, o ana kadar sürü içerisinde bulunan parçacıklar arasındaki en iyi değere sahip parçacığın elde ettiği değerdir. Bu parçacık, gbest olarak adlandırılır. Özetlemek gerekirse, PSO, her iterasyon adımında parçacığın pbest ve gbest konumuna doğru olan hızının değişmesinden ibarettir (Karahan ve Bingül 2009).

Aşağıda PSO'ya dair formulasyon bulunmaktadır (He et al. 2004).

$$V_i^{k+1} = \underbrace{\omega V_i^k}_{\text{Atalet (Inertia)}} + \underbrace{c_1 r_1(Pbest_i^k - X_i^k)}_{\text{Kişisel Etki (Personal Influence)}} + \underbrace{c_2 r2(Gbest^k - X_i^k)}_{\text{Sosyal Etki (Social Influence)}} \quad (2.1)$$

$$X_i^{k+1} = X_i^k + V_i^{k+1} \quad (2.2)$$

ω = Atalet Çarpanı

c_1 = Sabit Çarpan (0,8)

c_2 = Sabit Çarpan (0,8)

r_1 = 0 ile 1 arasında rastsal sayı

r_2 = 0 ile 1 arasında rastsal sayı

$Pbest_i^k$ (Personel Best) = i. parçacığın k. iterasyonuna kadar aldığı en iyi koordinat

$Gbest^k$ (Global Best) = Populasyondaki parçacıkların o ana kadar aldığı en iyi koordinat

X_i^k = i. parçacığının k iterasyonundaki koordinatı

X_i^{k+1} = i. parçacığının k+1 iterasyonundaki koordinatı

V_i^k = i. parçacığın k iterasyonu için hesaplanan hız vektörü

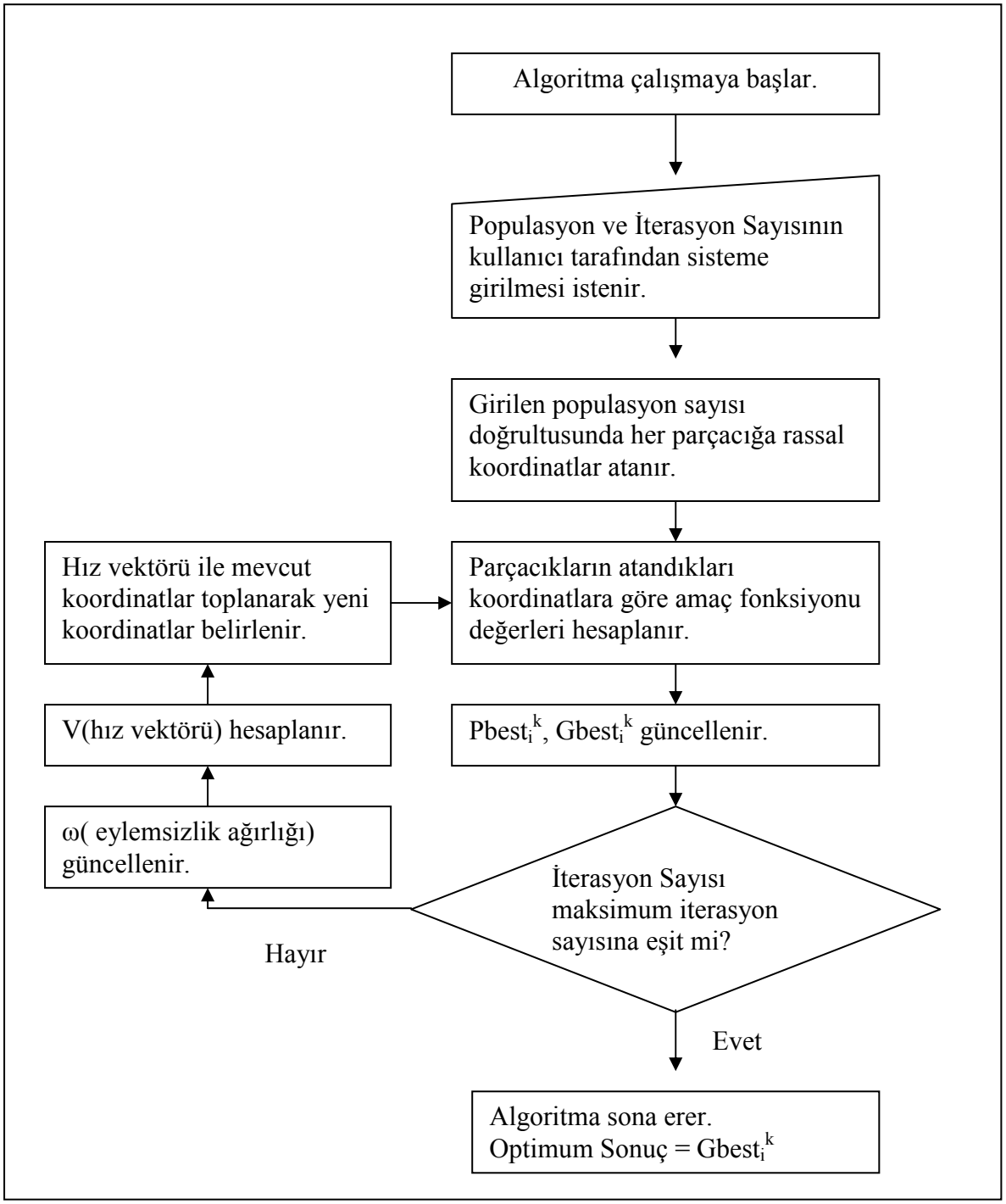
V_i^{k+1} = i. parçacığın k+1 iterasyonu için hesaplanan hız vektörü

V hız vektörüdür. Parçacığın mevcut koordinatına eklenecek ve parçacığa yeni bir koordinat sunacaktır. Bu vektör öğrenmenin gerçekleştiği ve elde edilen bilginin algoritmaya kazandırıldığı yerdir.

ω katsayısı 0,9 ile 0,4 arasında iterasyon sayısına bağlı olarak doğrusal değer almaktadır (Kaveh ve Talatahari 2009). Katsayının 0,9'dan başlayarak iterasyon ilerledikçe azalması, optimum nokta etrafındaki adım genişliğini daraltarak istenen sonuca daha az iterasyonda ulaşılmasını sağlamaktadır.

Pbest (Personal Best Point) i. karıncanın o ana kadar almış olduğu en iyi noktayı ifade eder. Her iterasyon sonrası bu değer güncellenir. Ve daha sonraki iterasyonda hız

vektörü hesaplanmasında yer alır. Gbest (Global Best Point) populasyon içerisinde o ana kadar alınmış en iyi noktayı temsil eder. Pbest gibi her iterasyon sonrası bu nokta da güncellenir ve V vektöründeki yerini alır. c_1 ve c_2 0,8 değerini alacaktır. r_1 ve r_2 0 ile 1 arasında rastsal değer alır.



Şekil 2.1 PSO Algoritması

Şekil 2.1'de Parçacık Sürüsü Optimizasyonuna ait algoritma verilmektedir. Algoritma adımları Formül 2.1 ve Formül 2.2'ye göre hazırlanmıştır.

2.2 PSOPC - Pasif Katılımlı Parçacık Sürüsü Optimizasyonu

PSO ve PSOPC'nin türev bilgisi gerektirmeksizin optimizasyon yapması en büyük üstünlüğü olarak görülebilir. Bu teknik, problemin çözümü için önceden belirlenen parametrelere (parçacık) rasgele atanmış aday çözümlerden oluşan bir popülasyonla (sürü) ile başlar. Her bir PSO algoritması iterasyonunda, sürüdeki her bir parçacık arama uzayındaki kendi pozisyonunu o iterasyona kadar bulunmuş en iyi parçacık pozisyonuna dayalı olarak ayarlar (Karakuzu 2007). Her parçacığın o ana kadar başardığı Pbest ve populasyonun o ana kadar başardığı Gbest noktalarına ilave olarak ayrıca rassal bir parçacık koordinatı da seçilmektedir. Bu seçilim aramaya globallik kazandırmakta ve çözüm aranan bölge dışında kalabilecek muhtemel bir sonucun göz ardı edilmesi riskini azaltmaktadır.

Aşağıda PSOPC'ye dair formulasyon bulunmaktadır.

$$V_i^{k+1} = \underbrace{\omega V_i^k}_{\text{Atalet (Inertia)}} + \underbrace{c_1 r_1(Pbest_i^k - X_i^k)}_{\text{Kişisel Etki (Personal Influence)}} + \underbrace{c_2 r2(Gbest^k - X_i^k)}_{\text{Sosyal Etki (Social Influence)}} + \underbrace{c_3 r_3 (R^k - X_i^k)}_{\text{Pasif Katılım (Passive Congregation)}} \quad (2.3)$$

$$X_i^{k+1} = X_i^k + V_i^{k+1} \quad (2.4)$$

ω = Atalet Çarpanı
c_1 = Sabit Çarpan (0,8)
c_2 = Sabit Çarpan (0,8)
c_3 = Sabit Çarpan (0,6)
r_1 = 0 ile 1 arasında rastsal sayı
r_2 = 0 ile 1 arasında rastsal sayı

r_3 = 0 ile 1 arasında rastsal sayı

$Pbest_i^k$ (Personel Best) = i. parçacığın k. iterasyonuna kadar aldığı en iyi koordinat

$Gbest^k$ (Global Best) = Populasyondaki parçacıkların o ana kadar aldığı en iyi koordinat

R^k (Random Point) = k. iterasyonda seçilen rastsal bir parçacığın koordinatı

X_i^k = i. parçacığının k iterasyonundaki koordinatı

X_i^{k+1} = i. parçacığının k+1 iterasyonundaki koordinatı

V_i^k = i. parçacığın k iterasyonu için hesaplanan hız vektörü

V_i^{k+1} = i. parçacığın k+1 iterasyonu için hesaplanan hız vektörü

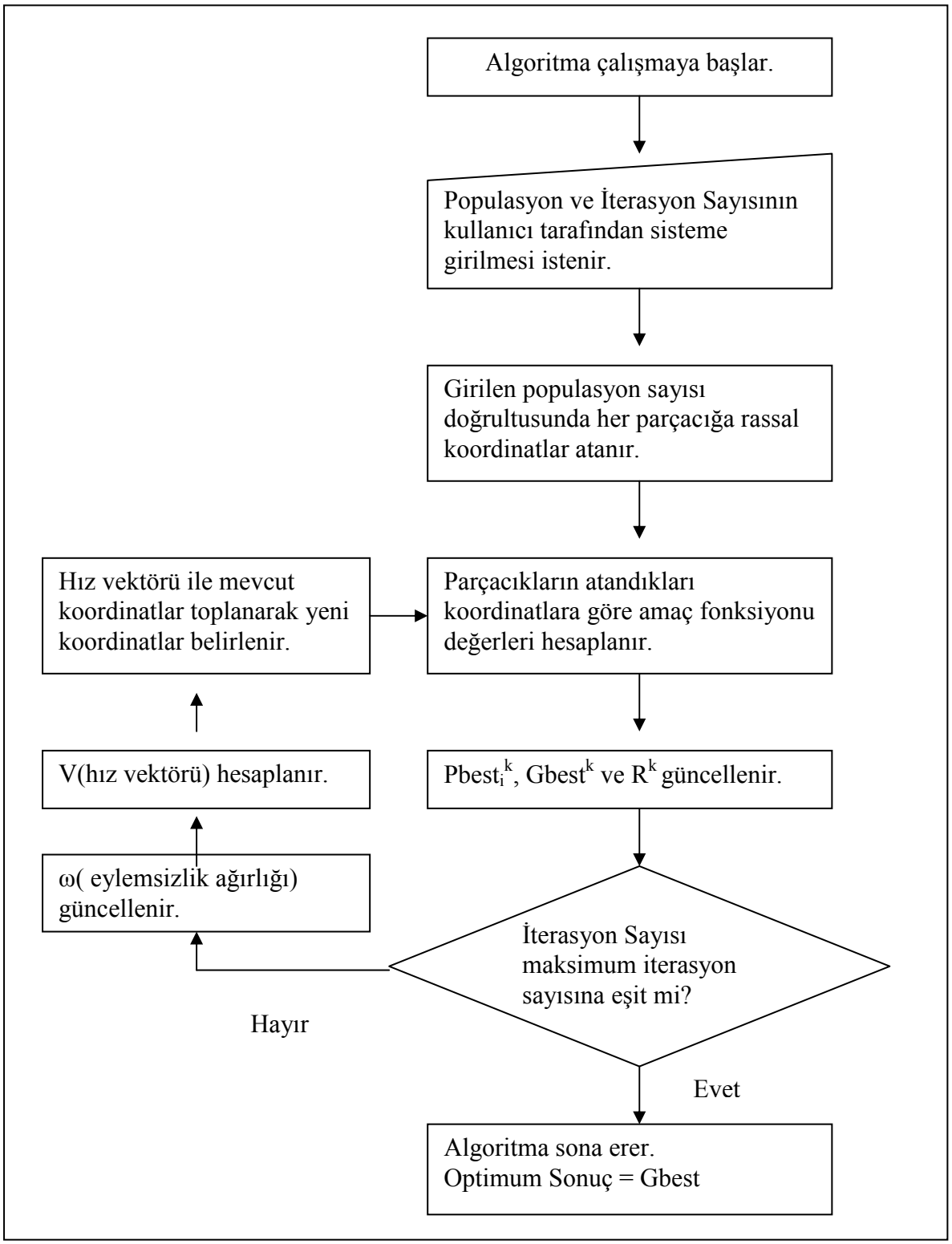
V, PSO' da olduğu gibi bu formülasyonda da hız vektörüdür. Aynen parçacığın mevcut koordinatına eklenecek ve parçacığa yeni bir koordinat sunacaktır. Bu formulasyondaki tek ve en önemli fark vektördeki üçüncü kısım olan populasyon içerisinden rastsal bir parçacığın hız vektörüne dâhil olmasıdır.

Görüldüğü üzere parçacık sürü algoritmasındaki bu güncellemeler basit toplam ve çarpımlardan oluşmakta ve türev bilgisi gerektirmemektedir. Öğrenme faktörleri olan c_1 ve c_2 her parçacığı pbest ve gbest değerlerine doğru çeker. Denklemdeki r_1 ve r_2 ise 0-1 arasında seçilmiş rasgele sayılardır. k iterasyon sayısını göstermektedir. Eylemsizlik ağırlığı olan ω her iterasyonda doğrusal olarak azaltılmalı ve birden küçük seçilmelidir (Der vd. 2008).

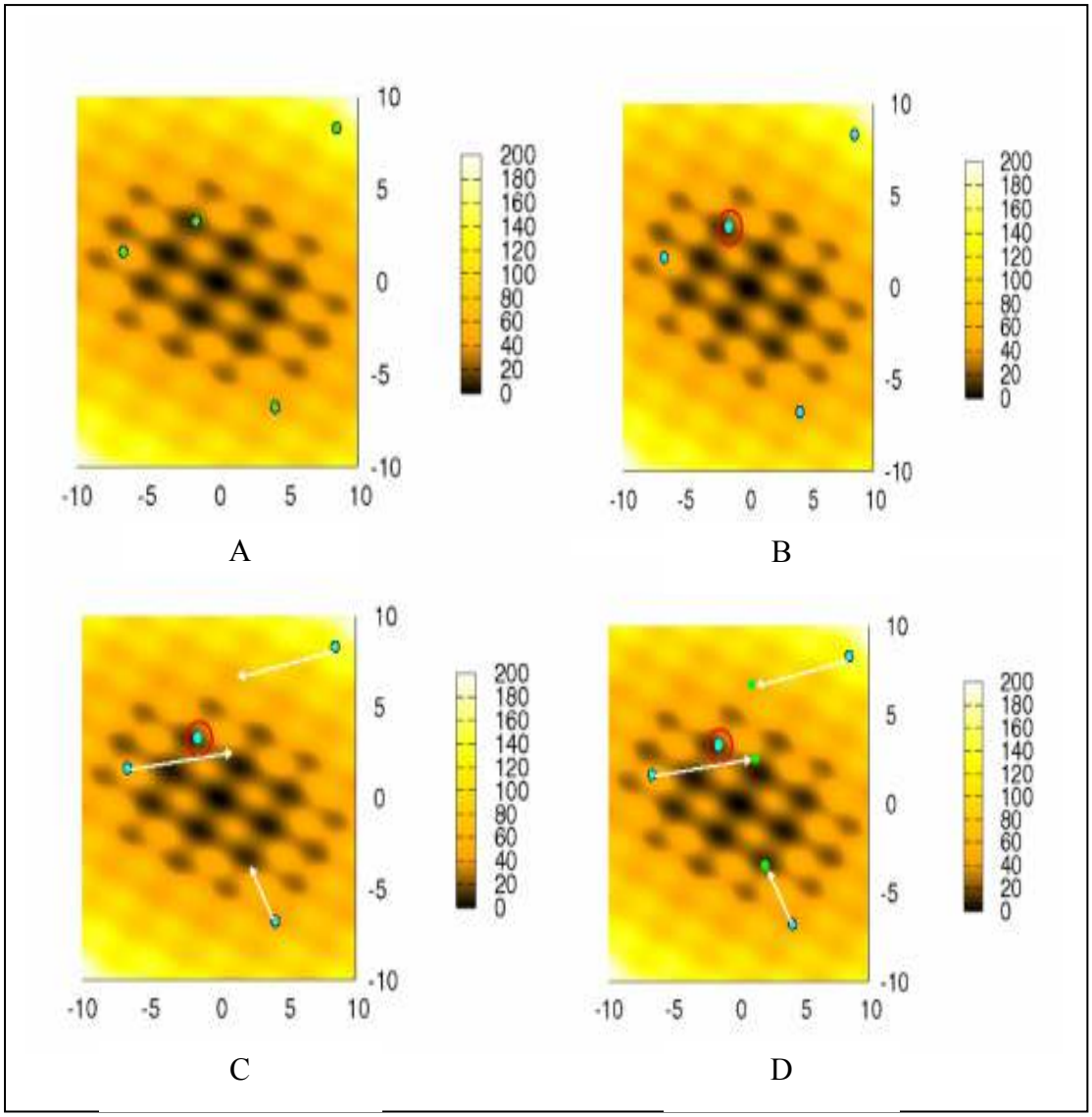
Bu yöntemin en önemli yanı; hem parçacığın hem de sürünün öğrenmesidir. Bu ikisinin entegrasyonuna ayrıca rastsal seçilen bir parçacığın dâhil olması rastsal bir parçacığa şans verilerek geniş alanda arama yapabilme kabiliyetini vermektedir. Bu şekilde herhangi bir parçacığın yanlış bir hareketi populasyon tarafından soğurulmakta ve bu parçacığın populasyonu olumsuz etkilemesi minimuma indirgenmektedir.

ω katsayısı 0,9 ile 0,4 arasında iterasyon sayısına bağlı olarak doğrusal değer almaktadır (Kaveh ve Talatahari 2009). Katsayının 0,9'dan başlayarak iterasyon ilerledikçe azalması, optimum nokta etrafındaki adım genişliğini daraltarak istenen sonuca daha az iterasyonda ulaşılmasını sağlamaktadır. R (Random Point) mevcut populasyon

içerisinden rastsal seçilen bir parçacığın koordinatıdır. Hız vektörüne c_3=0,6 çarpanı ile dâhil olur. Şekil 2.2'de PSOPC ait algoritma verilmektedir.



Şekil 2.2 PSOPC Algoritması



Şekil 2.3 PSOPC algoritmasında parçacıkların koordinatlarının belirlenmesi (İnt. Kyn. 3)

Yapılan bu çalışmaya göre;

A: Rastsal atanan 4 nokta ile popülasyonun oluşturulması,

B: Popülasyon içerisinden Gbest(en iyi parçacık)'in seçilmesi,

C: V hız vektörlerinin hesaplanması ve oklar halinde belirtilmesi

D: V hız vektörü ile ilk koordinatların toplanarak ikinci popülâsyonun bulunmasıdır.

Şekil 2.3'te PSOPC algoritmasının ilk iterasyondaki temel çözüm adımları bulunmaktadır. Grafiklerin sağ tarafında, seçilen fonksiyona dair istenen minimum sonuçların bulunduğu skala verilmektedir. Fonksiyonun birden çok lokal çözümü vardır. Bu çözümler grafiklerde koyu tonlamalı olarak gösterilmektedir.

Algoritmanın daha iyi anlaşılması için başka bir örnek ele alınarak adımlar tekrar incelenecektir. Aşağıda bu adımlar bulunmaktadır.

Basit bir amaç fonksiyonu;

$$F(x) = x_1^2 + x_2^2 - 10 \qquad (2.5)$$

Bu fonksiyona minimum değeri verecek x_1 ve x_2 değişkenleri aranmaktadır. Çözüm 5 parçacık ve 5 iterasyonla aranacak, adımlar daha net görülebilecektir. Fonksiyonun analitik çözümü, $x_1 = x_2 = 0$'da Fmin = -10'dur.

Çizelge 2.1 Rastsal atanan popülasyon

5 Rastsal Başlangıç Noktası			
	x_1	x_2	F
1	-10	0	90
2	2	4	**10**
3	-8	-4	70
4	-3	-8	63
5	-5	7	64

Çizelge 2.2 İlk Popülasyon için hesaplanan V

	X_1'in hız vektörü	X_2'nin hız vektörü
1	9,4392	-0,2527
2	-1,2590	-3,0216
3	-0,3101	1,7780
4	0,8824	2,5149
5	5,2218	-3,7299

Gbest=10

Algoritmanın başlaması için Çizelge 2.1' de gösterilen 5 adet rastsal atanmıştır. Atanan 5 rastsal nokta için amaç fonksiyonu değerleri hesaplanmış ve en iyi değeri veren parçacığın koordinatları Gbest koordinatları olmuştur. Her parçacık için ilk Pbest değerleri ilk koordinatlara göre hesaplanmış değerlerdir, ileriki iterasyonlarda bu değerler alınan daha iyi değerlerle güncellenecektir

Çizelge 2.2'de koordinatlar için hesaplanmış hız vektörleri bulunmaktadır. Bu hız vektörleriyle mevcut koordinatlar toplanarak Çizelge 2.3'de bulunan yeni koordinatlara ulaşılacaktır.

Çizelge 2.3 İkinci Populasyon

2. Populasyon			
	x_1	x_2	F
1	-0,5608	-0,2527	**-9,6216**
2	0,7410	0,9784	-8,4936
3	-8,3101	-2,2220	63,9947
4	-2,1176	-5,4851	24,5700
5	0,2218	3,2701	0,7429

Çizelge 2.4 İkinci Popülasyon için hesaplanan V

	X_1'in hız vektörü	X_2'nin hız vektörü
1	3,6844	-1,1185
2	-3,0659	-6,1920
3	2,8855	3,1010
4	1,6521	5,2033
5	3,6706	-5,0370

Gbest=-9,6216

İkinci popülasyon için güncellenen Gbest değeri Çizelge 2.3'de gösterilmektedir. İkinci popülasyonla Çizelge 2.4'de bulunan hız vektörleri toplanarak üçüncü popülasyonun koordinatları oluşturulacaktır. Yeni koordinatlar Çizelge 2.5'de verilmiştir.

Çizelge 2.5 Üçüncü Popülasyon

3. Populasyon			
	x1	x2	F
1	3,1236	-1,3712	1,6370
2	-2,3249	-5,2135	22,5860
3	-5,4246	0,8790	20,1989
4	-0,4655	-0,2818	**-9,7039**
5	3,8925	-1,7669	8,2731

Çizelge 2.6 Üçüncü Popülasyon için hesaplanan V

	X_1'in hız vektörü	X_2'nin hız vektörü
1	-0,7508	0,1660
2	2,2599	1,6439
3	3,9010	-0,5923
4	0,4515	3,9918
5	-0,4757	0,2276

Gbest=-9,7039

Üçüncü popülasyonda Gbest tekrar güncellenmiştir. Her iterasyonda sonuçlar daha iyiye doğru gitmekte ve güncellemeler yapılmaktadır. Dördüncü populasyon, Çizelge 2.6'da hesaplanan hız vektörleri ve Üüçncü populasyona ait koordinatların toplanmasıyla oluşturulmakta ve Çizelge 2.7'de verilmektedir.

Çizelge 2.7 Dördüncü Popülasyon

4. Populasyon			
	x1	x2	F
1	2,3728	-1,2052	-2,9175
2	-0,0650	-3,5697	2,7468
3	-1,5236	0,2867	-7,5965
4	-0,0140	3,7100	3,7641
5	3,4168	-1,5393	4,0438

Çizelge 2.8 Dördüncü Popülasyon için hesaplanan V

	X_1'in hız vektörü	X_2'nin hız vektörü
1	-2,3632	0,5007
2	1,9442	-0,9448
3	3,2128	-0,0624
4	-0,1378	-2,1982
5	-2,9595	2,0645

Dördüncü popülasyon bireylerinden Gbest için daha iyi bir sonuç olmadığı için güncellenme olmamaktadır. Dördüncü popülasyonun Çizelge 2.8'de hesaplanan hız vektörleriyle toplanması sonucu beşinci populasyona ulaşılmaktadır. Beşinci populasyon Çizelge 2.9'da verilmektedir.

Çizelge 2.9 Beşinci Popülasyon

5. Populasyon			
	x1	x2	F
1	0,0095	-0,7045	-9,5036
2	1,8792	-4,5145	13,9124
3	1,6892	0,2243	-7,0963
4	-0,1519	1,5118	-7,6915
5	0,4572	0,5252	-9,5151

Çizelge 2.10 Beşinci Popülasyon için hesaplanan V

	X_1'in hız vektörü	X_2'nin hız vektörü
1	-1,2214	0,0458
2	-1,5269	0,7554
3	-0,7063	-0,3056
4	0,3028	-1,6285
5	-2,0311	-0,3279

Beşinci popülasyon bireylerinden de Gbest için daha iyi bir sonuç olmadığı için güncellenme olmamaktadır. Beşinci popülasyonun Çizelge 2.10'da bulunan hız vektörleriyle toplanması sonucu beşinci populasyona hesaplanmaktadır. Son populasyon Çizelge 2.11'de verilmektedir.

Çizelge 2.11 Son Popülasyonun Koordinatları

Son Populasyon			
	x1	**x2**	**F**
1	-1,2119	-0,6586	-8,0975
2	0,3523	-3,7591	4,2552
3	0,9829	-0,0813	-9,0273
4	0,1509	-0,1167	**-9,9636**
5	-1,5739	0,1973	-7,4840

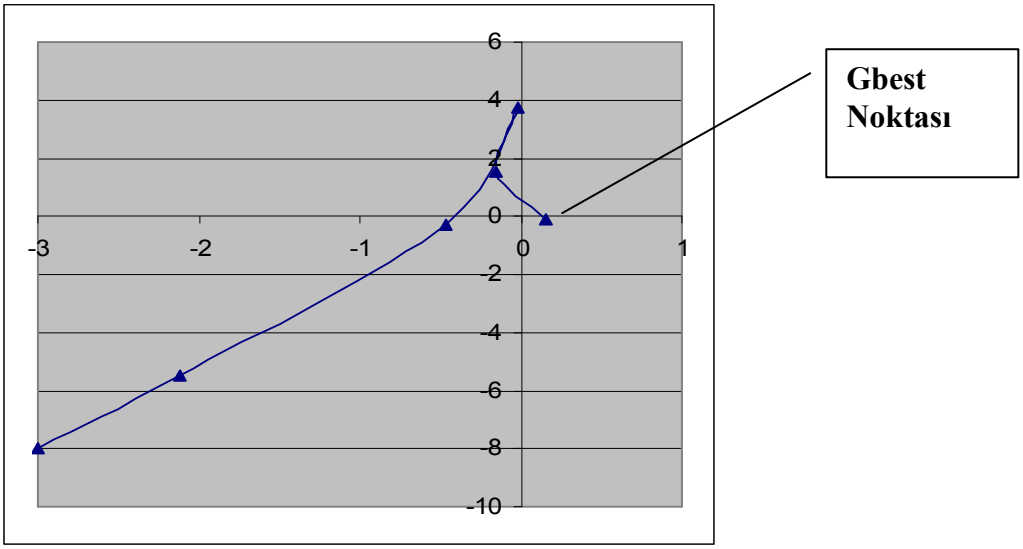
Gbest=-9,7039

En iyi noktayı veren parçacık 4. parçacık oldu. Başlangıç ve 5 iterasyon sonrası aldığı değerler Çizelge 2.11 'da verilmiştir. Daha sonra bu çizelgedeki verilerden bir grafik çizilmiş parçacığın izlediği yol gözlemlenmiştir. Parçacığın bu hareketleri izlemesinde yörüngesini; kendi öğrenmesi, sürünün öğrenmesi ve rastsal bir parçacığın koordinatı belirlemiştir.

Çizelge 2.12 Dördüncü Parçacığın izlediği koordinatlar

4. PARÇACIK		
	x1	**x2**
Başlangıç noktası	-3,0000	-8,0000
1. iterasyon sonrası	-2,1176	-5,4851
2. iterasyon sonrası	-0,4655	-0,2818
3. iterasyon sonrası	-0,0140	3,7100
4. iterasyon sonrası	-0,1519	1,5118
5. iterasyon sonrası	0,1509	-0,1167

Bir parçacın her iterasyondaki yer değiştirmesi hız vektörü sayesinde helezonik bir harekete sebep olmaktadır. Çizelge 2.12'de Gbest değerini veren parçacığın her iterasyondaki koordinatları verilmiştir. Bu koordinatlar, koordinat düzlemine yerleştirildiğinde Şekil 2.4'deki helezonik hareket biçimine ulaşılmaktadır.



Şekil 2.4 Dördüncü Parçacığın izlediği koordinatlar

Bilinen ve beklenen en iyi sonuç olan -10 değeridir. Bu değeri sağlayan X_1 ve X_2 değerleri 0'dır.Yapılan optimizasyon çalışmasında 5 parçacık ve 5 iterasyon kullanılmıştır. Bu kadar az parçacık ve iterasyonla optimum sonuca ciddi anlamda yaklaşılmış ve algoritma X_1 'e 0,1509, X_2'ye -0,1167 değerlerini atamıştır. Atanan değişken değerlerine göre F(x) -9,9636 olmuştur.

Daha karmaşık ve zor modellerde de tekniğin olumlu sonuçlar verdiği bu çalışmanın ilerleyen safhalarında gözlemlenecektir.

2.3 Genetik Algoritma

Olasılık kurallarına göre çalışan genetik algoritmalar, yalnızca amaç fonksiyonuna gereksinim duyarlar. Çözümlerden oluşan populasyonlar eş zamanlı incelemeye tabii olduklarından, yerel en iyi çözümlere takılı kalmazlar. Canlıların yapılarında var olan bir takım özellikler, sanal ortamlarda taklit edilerek modeller geliştirilmeye ve bu modellerle de karşılayan problemlere çözümler bulunmaya çalışılmaktadır. Bu modellerin birisi olan genetik algoritmalar, canlıların çevreye uyum ve genetik özelliklerinin araştırılmasıyla geliştirilmiştir (Karakaya 2007).

Genetik algoritmalar problemlerin çözümü için evrimsel süreci bilgisayar ortamında taklit ederler. Diğer eniyileme yöntemlerinde olduğu gibi çözüm için tek bir yapının geliştirilmesi yerine, böyle yapılardan meydana gelen bir küme oluştururlar. Problem için olası pekçok çözümü temsil eden bu küme genetik algoritma terminolojisinde nüfus adını alır. Nüfuslar vektör, kromozom veya birey adı verilen sayı dizilerinden oluşur. Birey içindeki her bir elemana gen adı verilir. Nüfustaki bireyler evrimsel süreç içinde genetik algoritma işlemcileri tarafından belirlenirler (İnt. Kyn. 4).

Problemin bireyler içindeki gösterimi problemden probleme değişiklik gösterir. Genetik algoritmaların problemin çözümündeki başarısına karar vermedeki en önemli faktör, problemin çözümünü temsil eden bireylerin gösterimidir.

Nüfus içindeki her bireyin problem için çözüm olup olmayacağına karar veren bir uygunluk fonksiyonu vardır. Uygunluk fonksiyonundan dönen değere göre yüksek değere sahip olan bireylere, nüfustaki diğer bireyler ile çoğalmaları için fırsat verilir. Bu bireyler çaprazlama işlemi sonunda çocuk adı verilen yeni bireyler üretirler. Çocuk kendisini meydana getiren ebeveynlerin (anne, baba) özelliklerini taşır.

Yeni bireyler üretilirken düşük uygunluk değerine sahip bireyler daha az seçileceğinden bu bireyler bir süre sonra nüfus dışında bırakılırlar. Yeni nüfus, bir önceki nüfusta yer alan uygunluğu yüksek bireylerin bir araya gelip çoğalmalarıyla oluşur. Aynı zamanda bu nüfus önceki nüfusun uygunluğu yüksek bireylerinin sahip olduğu özelliklerin büyük bir kısmını içerir. Böylelikle, pek çok nesil aracılığıyla iyi özellikler nüfus içersinde yayılırlar ve genetik işlemler aracılığıyla da diğer iyi özelliklerle birleşirler.

Uygunluk değeri yüksek olan ne kadar çok birey bir araya gelip, yeni bireyler oluşturursa arama uzayı içerisinde o kadar iyi bir çalışma alanı elde edilir (İnt. Kyn. 4).

Bir çok alanda uygulama imkanı ve uygulamaları olan genetik algoritmaların işleme adımları şöyle açıklanabilir:

- Arama uzayındaki tüm mümkün çözümler dizi olarak kodlanır.
- Genellikle rastsal bir çözüm kümesi seçilir ve başlangıç populasyonu olarak kabul edilir.
- Her bir dizi için bir uygunluk değeri hesaplanır, bulunan uygunluk değerleri dizilerin çözüm kalitesini gösterir.
- Bir grup dizi belirli bir olasılık değerine göre rastsal olarak seçilip çoğalma işlemi gerçekleştirilir.
- Yeni bireylerin uygunluk değerleri hesaplanarak, çaprazlama ve mutasyon işlemlerine tabi tutulur.
- Önceden belirlenen kuşak sayısı boyunca yukarıdaki işlemler devam ettirilir.
- İterasyon, belirlenen kuşak sayısına ulaşınca işlem sona erdirilir. Amaç fonksiyonuna göre en uygun olan dizi seçilir (Emel ve Taşkın 2002).

Ele alınan optimizasyon problemlerinin çözümleri; Parçacık Sürüsü Optimizasyonu, Pasif Katılımlı Parçacık Sürüsü Optimizasyonu ve Genetik Algoritma teknikleri ile yapılacak ve Pasif Katılımlı Parçacık Sürüsü Optimizasyonunun diğer tekniklere göre avantajlı olup olmadığı incelenecektir. Parçacık Sürüsü Optimizasyonun, Genetik Algoritmayla en önemli benzerliği en başta seçilen popülasyonun rastsal olarak oluşturulması ve sezgisel bir algoritmayla çalışıyor olmasıdır.

3. ALGORİTMA YAZILIMI VE TESTLERİ

PSO ve PSOPC'ye dair kod yazılımı MATLAB 7.7.0.471(R2008b) adlı yazılım paketi ile yapılmıştır. Kıyaslamada kullanılacak olan Genetik Algoritma ise MATLAB 7.7.0.471(R2008b) içerisinde hazır bir optimizasyon modülü olarak yer almaktadır.

3.1 MATLAB Yazılım Paketi

MATLAB, MATrix LABoratory sözcüklerinden gelir ve temelde sayısal ve analitik olarak matematiksel fonksiyonların ifadelerinin kullanıldığı basta mühendislik alanında olmak üzere birçok sayısal analizi kullanan bilimlerde son yıllarda oldukça sik kullanılan bir hazır yazılım paketidir. Özellikle yüksek performans gerektiren algoritma hazırlama ve geliştirme, sayısal analiz, simülasyon, mühendislik problemlerinin sayısal ve grafik çözüm tekniklerinde son derece etkindir (Bekler 2006).

MATLAB yüksek seviyeli bir teknik programlama dili olmasının yanında algoritma geliştirme, verilerin görselleştirilmesi, veri analizi ve sayısal hesaplamalar için etkileşimli bir yazılım paketidir. MATLAB ile teknik hesaplama problemleri, C,C++ ve Fortran gibi geleneksel programlama dillerinden daha hızlı bir şekilde çözülebilmektedir. MATLAB yazılımının birçok alanda uygulamaları vardır. İçerdiği "toolbox" adı verilen paketler aracılığıyla sayısal işaret işleme, kontrol tasarımı, test ve ölçüm, finansal modelleme ve analiz, haberleşme gibi birçok alanda kullanılabilir (İnt. Kyn. 5).

Ana Özellikleri:

• Teknik hesaplamalar için yüksek seviyeli bir dil

• Kodların, dosyaların ve verilerin düzenlenmesi için bir geliştirme ortamı

• İteratif tasarım ve problem çözme yöntemleri için interaktif araçlar

• Lineer cebir, istatistik, Fourier analizi, filtreleme, optimizasyon ve sayısal integrasyon için matematik fonksiyonlar

• Verilerin görselleştirilmesi için 2 ve 3 boyutlu grafik araçları

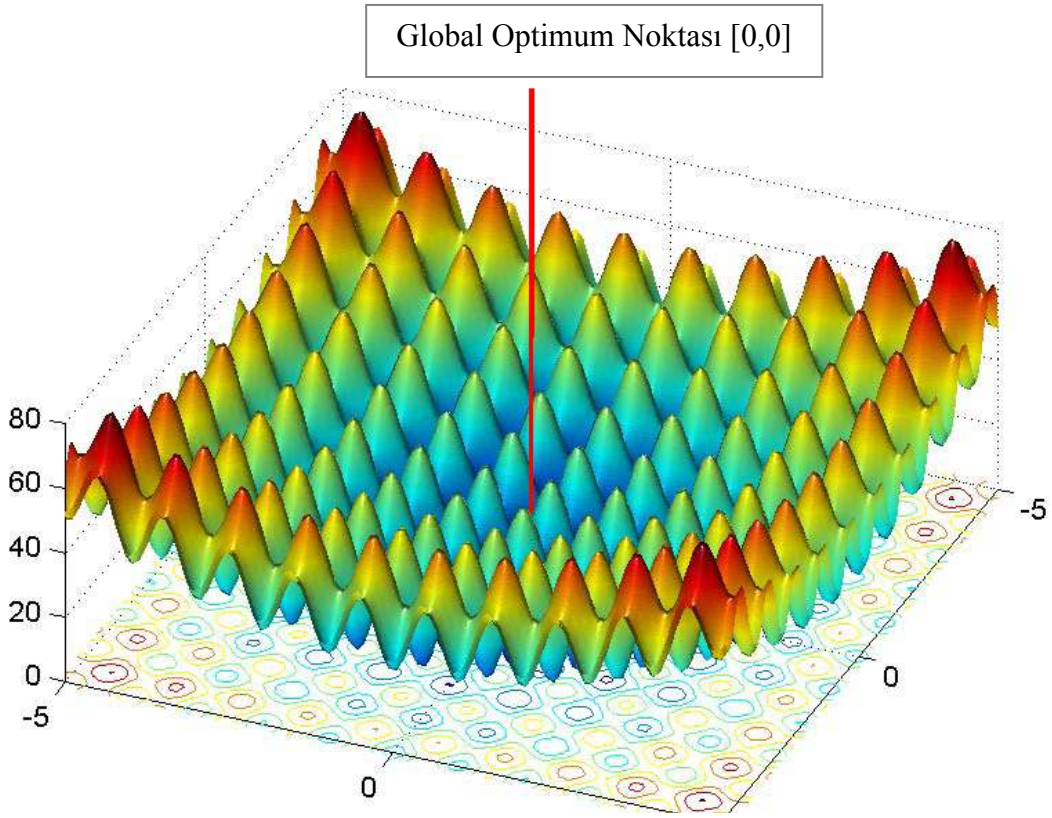
3.2 Rastrigin Fonksiyonunun İncelenmesi

MATLAB modüllerini kullanıcıya anlatmak amacıyla pek çok yardım dosyası oluşturmuştur. Optimizasyon modül(toolbox)'ü içerisinde bulunan Genetik Algoritma için hazırlanmış dosya içerisinden, algoritmanın anlatımında kullanılan fonksiyon ele alınmaktadır. PSO, PSOPC ile GA'nın kıyaslanmasında objektif ve sağlıklı sonuç alınabilmesi için bu fonksiyon tercih edilmiştir.

Aşağıda rastrigin fonksiyonu bulunmaktadır;

$$Ras(x) = 20 + x_1^2 + x_2^2 - 10(\cos 2\pi\, x_1 + \cos 2\pi\, x_2) \qquad (3.1)$$

Rastigin fonksiyonunun bir adet global ve pek çok lokal çözüm bulunmaktadır. Bu çözümlerin bulunduğu grafiksel gösterim Şekil 2.5'de verilmektedir.



Şekil 3.1 Rastrigin Fonksiyonunun global ve lokal çözümlerinin gösterimi

Rastrigin Fonksiyonu iki değişkenli bir fonksiyondur. Fonksiyon için minimum değer aranacaktır. Fonksiyonun global minimumu Şekil 3.1'de de görüldüğü gibi 0'dır. Bu sonuç için x_1 ve x_2 değişkenlerinin alacağı değer de 0'dır. Ayrıca fonksiyonun pek çok lokal optimum sonucu vardır.

PSO ve PSOPC'da fonksiyonun çözümü yapılacak, MATLAB'da yazılan kodların çalışıp çalışmadığı kontrol edilecektir. Fonksiyonu minimum yapan x_1 ve x_2 değerleri bulunacaktır. Bu anlamda PSO, PSOPC ve GA teknikleri ele alınacaktır. Her teknik algoritmalar 20 populasyon ve 100 iterasyon ile çalıştırılacaktır. Değişkenlerin başlangıç değerleri normal dağılıma uyacak şekilde ortalaması 0 ve standart sapması 10 olarak random alınmıştır. Buna göre değişkenler ±30 aralığında %99,7olasılıkla değişecektir. Sırasıyla PSO, PSOPC ve GA sonuçları çizelgelerde verilecektir.

Çizelge 3.1 Rastrigin Fonksiyonunun Parçacık Sürüsü Optimizasyonu ile 20 kez algoritmanın çalıştırılması

PSO					
Çalıştırma Sayısı	**En İyi Değerin Bulunduğu İlk İterasyon**	**Değerlendirilme Sayısı (Popülasyon Sayısı x İterasyon Sayısı)**	**x1 Degeri**	**x2 Degeri**	**Amaç Fonksiyonu Değeri**
1	22	440	0,0002	-0,0002	0,0000
2	89	1780	0,0000	-0,0007	0,0000
3	33	660	0,0001	-0,0002	0,0000
4	97	1940	0,0015	0,0009	0,0006
5	80	1600	0,0001	0,0000	0,0000
6	97	1940	0,0011	-0,0004	0,0002
7	32	640	0,0006	0,0000	0,0000
8	39	780	-0,0008	0,0005	0,0001
9	86	1720	0,0002	-0,0008	0,0001
10	96	1920	0,0005	0,0024	0,0011
11	70	1400	0,0004	-0,0001	0,0000
12	50	1000	0,0002	-0,0001	0,0000
13	39	780	0,0004	0,0000	0,0000
14	48	960	0,0000	-0,0002	0,0000
15	33	660	-0,0003	-0,0001	0,0000
16	46	920	-0,0002	-0,0001	0,0000
17	93	1860	0,0004	-0,0011	0,0002
18	85	1700	0,0005	-0,0002	0,0000
19	99	1980	0,0025	0,0003	0,0012
20	83	1660	0,0008	-0,0003	0,0001
ORTALAMA	**65,85**	**1317**			**0,0002**

Parçacık Sürüsü Optimizasyonu ile algoritma 20 kez çalıştırılmış ve Çizelge 3.1'deki verilere ulaşılmıştır. Bu teknik ile ortalama 66 iterasyonda en iyi sonuçlara ulaşmaktadır. Ayrıca amaç fonksiyonu değerlerinin ortalaması da 0,0002'yi vermektedir.

Çizelge 3.2 Rastrigin Fonksiyonunun Pasif Katılımlı Parçacık Sürüsü Optimizasyonu ile 20 kez algoritmanın çalıştırılması

PSOPC					
Çalıştırma Sayısı	**En İyi Değerin Bulunduğu İlk İterasyon**	**Değerlendirilme Sayısı (Popülasyon Sayısı x İterasyon Sayısı)**	**x1 Degeri**	**x2 Degeri**	**Amaç Fonksiyonu Değeri**
1	38	760	-0,0004	0,0000	0,0000
2	35	700	-0,0003	0,0004	0,0000
3	40	800	0,0000	-0,0003	0,0000
4	43	860	-0,0002	-0,0002	0,0000
5	39	780	0,0000	0,0005	0,0000
6	43	860	0,0003	0,0006	0,0000
7	36	720	0,0003	0,0000	0,0001
8	51	1020	-0,0001	0,0002	0,0000
9	55	1100	0,0004	-0,0001	0,0000
10	60	1200	0,0000	-0,0004	0,0000
11	37	740	-0,0002	0,0003	0,0000
12	42	840	0,0004	0,0001	0,0000
13	37	740	-0,0001	-0,0002	0,0000
14	58	1160	0,0003	-0,0004	0,0000
15	49	980	0,0004	-0,0002	0,0000
16	40	800	0,0004	0,0004	0,0000
17	36	720	0,0005	0,0005	0,0000
18	59	1180	0,0003	0,0001	0,0000
19	36	720	0,0002	-0,0002	0,0000
20	35	700	-0,0003	-0,0003	0,0000
ORTALAMA	**43,45**	**869**			**0,0000**

Bu teknikte PSO gibi 20 kez çalıştırılmış ve Çizelge 3.2 elde edilmiştir. PSOPC için en iyi sonuca ulaşmak için ortalama iterasyon sayısı 43 olmuştur. 0,0000 ortalama amaç fonksiyon değeridir.

Çizelge 3.3 Rastrigin Fonksiyonunun Genetik Algoritma Tekniği ile 20 kez algoritmanın çalıştırılması

GA					
Çalıştırma Sayısı	**En İyi Değerin Bulunduğu İterasyon**	**Değerlendirilme Sayısı (Popülasyon Sayısı x İterasyon Sayısı)**	**x1 Degeri**	**x2 Degeri**	**Amaç Fonksiyonu Değeri**
1	100	2000	0,0269	0,0371	0,4141
2	100	2000	0,0776	-0,9740	2,2531
3	100	2000	0,9945	0,9950	1,9900
4	100	2000	0,0246	-0,9809	1,1534
5	100	2000	-0,0601	1,0113	1,7565
6	100	2000	-0,1159	-0,9745	3,6297
7	100	2000	-0,9130	0,0391	2,5937
8	100	2000	-0,9965	-0,1009	2,9481
9	100	2000	0,9933	-0,8962	3,8496
10	100	2000	0,0472	-0,0745	1,5202
11	100	2000	0,0101	1,0010	1,0222
12	100	2000	0,9379	1,0143	2,6999
13	100	2000	-0,0495	-0,0306	0,6662
14	100	2000	-1,0112	0,0846	2,4335
15	100	2000	-0,0582	1,0309	1,9162
16	100	2000	0,0585	-0,9724	1,7672
17	100	2000	1,0082	0,0382	1,3185
18	100	2000	0,9584	-0,0611	1,9903
19	100	2000	-1,0308	-0,9578	2,5164
20	100	2000	0,9766	-0,9729	2,1527
ORTALAMA	100	**2000**			**2,0296**

PSO ve PSOPC için MATLAB'da yazılan kodların verdiği sonuçlar açıklanmıştır. Genetik Algoritma için ise kod yazılmamış program içerisinde optimizasyon modüllerinde yer alan Genetik Algoritma için oluşturulmuş modül kullanılmıştır.

GA toolbox'ı içerisinde default değeri değiştirilen çalıştırma kriterleri;

Population Size: 20

Initial Range: [-30;30]

Mutation Function: Gaussian

Stopping Criteria;

Generation: 100

Stall Generation: 100

Genetik Algoritma için ortalama iterasyon sayısı 100'dür. Fakat verdiği en iyi amaç fonksiyonu değerlerinin ortalaması incelendiğinde bu değerin 2,0296 olduğu gözlemlenmektedir.

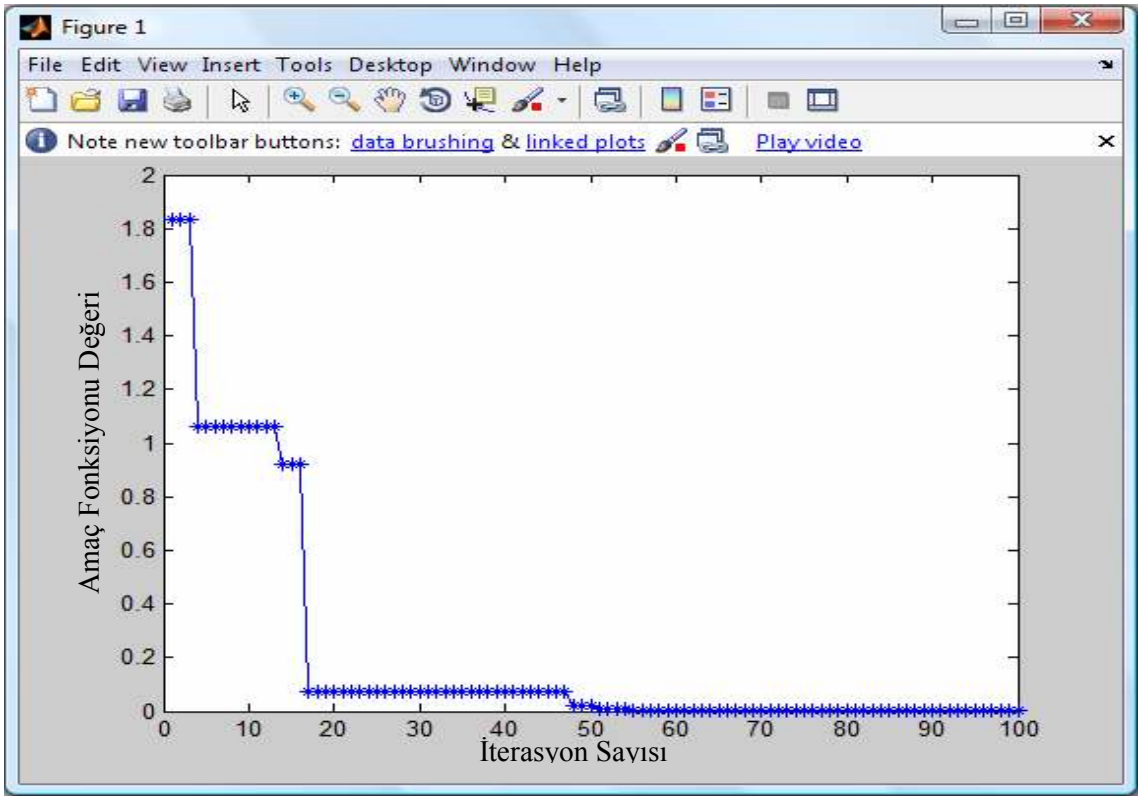
İterasyon sayısının ortalama 100 çıkmasının sebebi; durma kriterlerinde stall generation değerinin 100 seçilmesidir. Bu değer son 100 iterasyondaki en iyi değer bulunana kadar programı durdurma anlamına gelmektedir.

Çizelge 3.4 Rastrigin Fonksiyonu için PSO, PSOPC ve GA karşılaştırılması

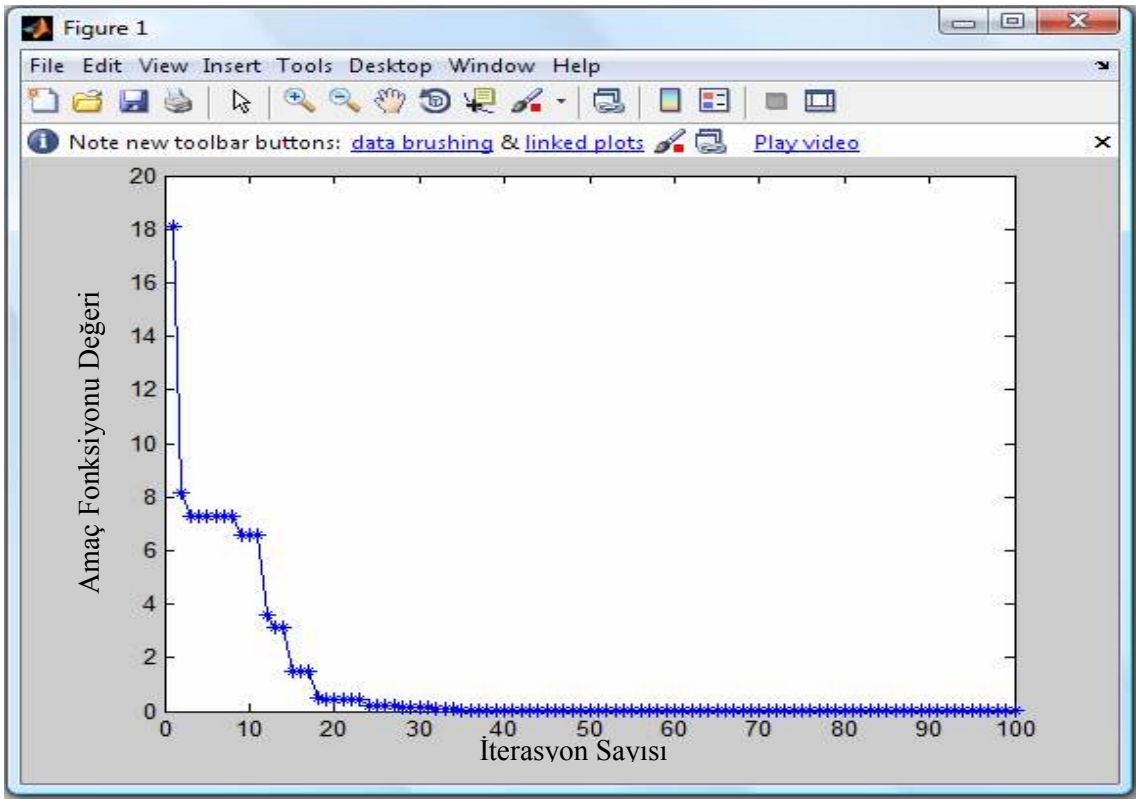
	PSO		**PSOPC**		**GA**	
RASTRIGIN FONKSİYONU	**Amaç Fonksiyon Değeri Ortalaması**	**Amaç Fonksiyonunun Değerlendiril-me Ortalaması**	**Amaç Fonksiyon Değeri Ortalaması**	**Amaç Fonksiyonunun Değerlendiril-me Ortalaması**	**Amaç Fonksiyon Değeri Ortalaması**	**Amaç Fonksiyonunun Değerlendiril-me Ortalaması**
	0,0002	1317	0,0000	869	2,0296	2000
	PSOPC'ye kıyasla				**PSOPC'ye kıyasla**	
	PSOPC'nin yakalamış olduğu amaç fonksiyon değerine çok yaklaşmıştır.	Ortalama Fonksiyonun Değerlendirme Sayılarına göre PSOPC %34 daha verimlidir.	---	---	GA'nın Amaç Fonksiyon Değeri Ortalaması PSOPC'ye göre%99,9 verimsizdir.	Ortalama Fonksiyonun Değerlendirme Sayılarına göre PSOPC %57 daha verimlidir.

PSO ve PSOPC'nin bu örnekte GA'ya göre göstermiş olduğu avantaj oldukça önemlidir. Bunun sebebi daha önce de bahsedildiği gibi bu örneğin MATLAB'a ait Genetik Algoritma'nın anlatımı için hazırlanmış olan kaynaktan alınmasıdır.

GA zaman zaman lokal çözümleri bulmakta, global çözümü daha az yakalayabilmektedir. PSO ve PSOPC ise global çözüm yada global çözüme çok yakın değerleri vermektedir.



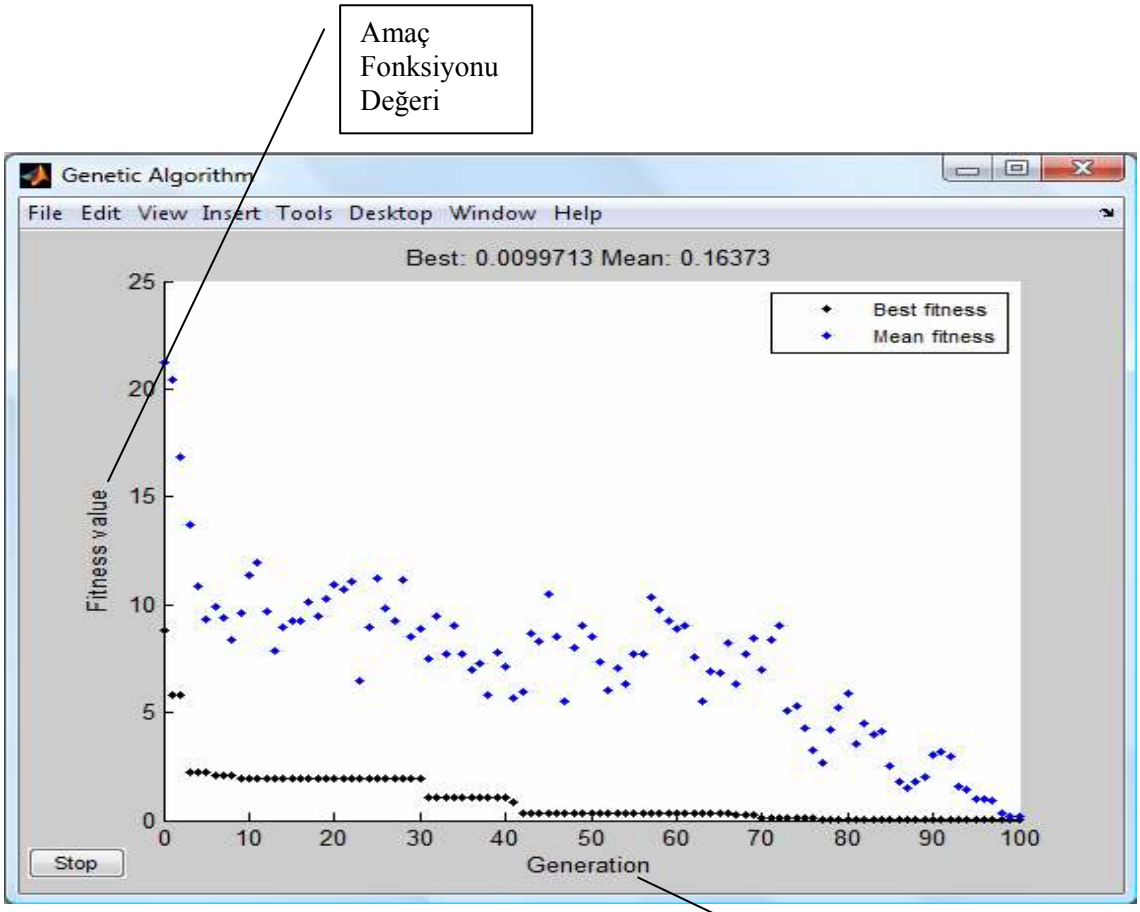
Şekil 3.2 Rastrigin Fonksiyonu, PSO amaç fonksiyonu değeri – iterasyon sayısı grafiği



Şekil 3.3 Rastrigin Fonksiyonu, PSOPC amaç fonksiyonu değeri – iterasyon sayısı grafiği

Şekil 3.2'da Parçacık Sürüsü Optimizasyon grafik çıktısı bulunmaktadır. Şekil 3.3'de ise Pasif Katılımlı Parçacık Sürüsü Optimizasyon grafik çıktısı gösterilmektedir. Bu iki grafik incelendiğinde PSO'ye ilave edilen pasif katılımın algoritmayı daha az iterasyonda daha iyi sonuca götürdüğü görülmektedir. Bahsedilen avantaj Çizelge 3.4'de sayısal verilere dayalı olarak verilmektedir. İki gösterim sonucunda PSOPC'nin PSO'dan daha güvenilir sonuçlar verdiği tespit edilmektedir.

Şekil 3.4'de aynı fonksiyonun çözümü sonucu programın verdiği grafik çıktısı bulunmaktadır.



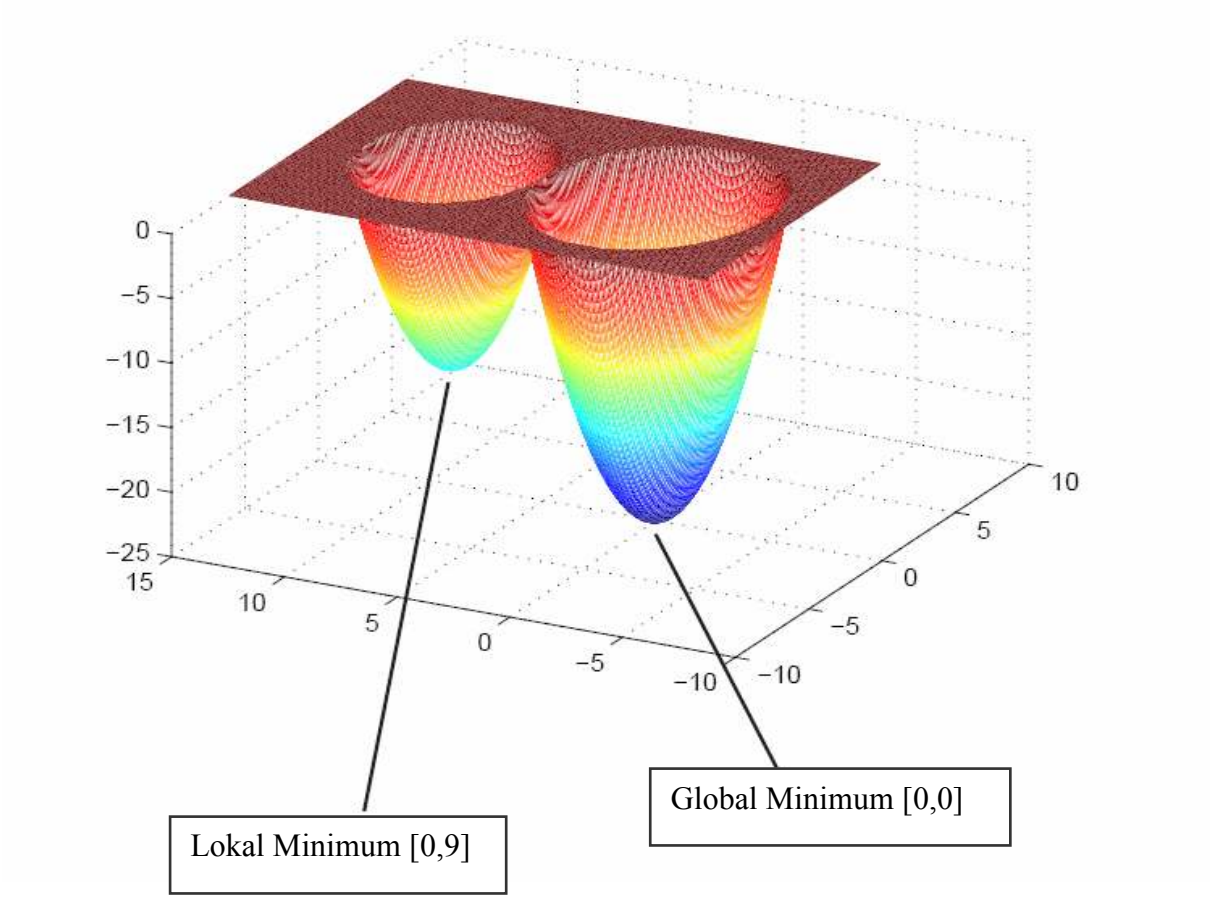
Şekil 3.4 Rastrigin Fonksiyonu, GA amaç fonksiyonu değeri – iterasyon sayısı grafiği

3.3 Bir Lokal Bir Global Optimumu olan Fonksiyonun İncelenmesi

MATLAB'ın optimizasyon modülünde bulunan tekniklerden biri de Model Arama tekniğidir. Model Arama tekniği için bir yardım dosyası oluşturmuştur. Bu dosyada bulunan fonksiyon ele alınacaktır. Yazılımın kontrolü ve kıyaslamanın gerçekçi olabilmesi için bu fonksiyon tercih edilmiştir.

Fonksiyonun formülasyonu;

$$f(x_1,x_2) = \begin{cases} x_1^2 + x_2^2 - 25 & \text{ise} \quad x_1^2 + x_2^2 \leq 25 \\ x_1^2 + (x_2 - 9)^2 - 16 & \text{ise} \quad x_1^2 + (x_2 - 9)^2 \leq 16 \\ 0 \end{cases} \tag{3.2}$$



Şekil 3.5 Fonksiyonunun global ve lokal çözümlerinin gösterimi

Bu fonksiyonun iki değişkenli bir fonksiyondur. Fonksiyonun bir tane global optimumu bir tane de lokal optimumu vardır. Lokal minimum için $x_1 = 0$ ve $x_2 = 9$ değerlerini almaktadır. Amaç Fonksiyon değeri ise -16'dır. Global optimumda ise $x_1 = 0$ ve $x_2 = 0$ değerleri amaç fonksiyon değerini -25 vermektedir. Bu değerler Şekil 3.5'de gösterilmektedir.

Bu fonksiyon PSO ve PSOPC için yazılan kodlarla birlikte MATLAB'da çözülecektir. GA için ise program içerisinde bulunan hazır modül kullanılacaktır. Yine kıyaslamada eşitliğin sağlanması için teknikler 20 populasyon ve 100 iterasyonla çalıştırılacaktır. Değişkenlerin başlangıç değerleri normal dağılıma uyacak şekilde ortalaması 0 ve standart sapması 10 olarak random alınmıştır. Buna göre değişkenler ±30 aralığında %99,7olasılıkla değişecektir. Sırasıyla PSO, PSOPC ve GA sonuçları verilecektir.

Çizelge 3.5 Parçacık Sürüsü Optimizasyonu ile 20 kez algoritmanın çalıştırılması

PSO					
Çalıştırma Sayısı	**En İyi Değerin Bulunduğu İterasyon**	**Değerlendirilme Sayısı (Popülasyon Sayısı x İterasyon Sayısı)**	**x1 Degeri**	**x2 Degeri**	**Amaç Fonksiyonu Değeri**
1	96	1920	0,0000	0,0000	-25,0000
2	69	1380	0,0000	0,0000	-25,0000
3	81	1620	-0,0000	0,0000	-25,0000
4	97	1940	0,0128	0,0013	-24,9998
5	99	1980	-0,0001	0,0001	-25,0000
6	76	1520	0,0000	0,0000	-25,0000
7	100	2000	-0,0001	0,0000	-25,0000
8	62	1240	0,0000	0,0000	-25,0000
9	89	1780	0,0004	0,0022	-25,0000
10	99	1980	0,0000	0,0000	-25,0000
11	98	1960	-0,0414	0,0195	-24,9979
12	86	1720	-0,0006	0,0000	-25,0000
13	62	1240	0,0000	0,0000	-25,0000
14	95	1900	0,0000	0,0000	-25,0000
15	98	1960	-0,0008	0,0000	-25,0000
16	68	1360	0,0000	0,0000	-25,0000
17	82	1640	0,0000	0,0002	-25,0000
18	99	1980	-0,0036	-0,0007	-25,0000
19	97	1940	0,0000	0,0000	-25,0000
20	93	1860	0,0008	-0,0004	-25,0000
ORTALAMA	**87,30**	**1746**			**-24,9999**

Parçacık Sürüsü Optimizasyonu ile algoritma 20 kez çalıştırılmış ve Çizelge 3.5'deki verilere ulaşılmıştır. Ortalama 87 iterasyonda en iyi sonuçlara ulaşmaktadır. Amaç fonksiyonu değerlerinin ortalaması -25 beklenirken bu algoritma -24,9999 sonucunu vermiştir.

Çizelge 3.6 Pasif Katılımlı Parçacık Sürüsü Optimizasyonu ile 20 kez algoritmanın çalıştırılması

PSOPC					
Çalıştırma Sayısı	**En İyi Değerin Bulunduğu ilk İterasyon**	**Değerlendirilme Sayısı (Popülasyon Sayısı x İterasyon Sayısı)**	**x1 Degeri**	**x2 Degeri**	**Amaç Fonksiyonu Değeri**
1	59	1180	0,0000	0,0000	-25,0000
2	47	940	0,0000	0,0000	-25,0000
3	62	1240	0,0000	0,0000	-25,0000
4	48	960	0,0003	0,0000	-25,0000
5	65	1300	0,0000	0,0000	-25,0000
6	50	1000	0,0000	0,0000	-25,0000
7	56	1120	0,0000	0,0000	-25,0000
8	60	1200	0,0000	0,0000	-25,0000
9	72	1440	0,0000	0,0000	-25,0000
10	96	1920	-0,0159	-0,0243	-24,9992
11	49	980	0,0000	0,0000	-25,0000
12	84	1680	0,0000	0,0000	-25,0000
13	89	1780	0,0000	0,0000	-25,0000
14	52	1040	0,0000	0,0000	-25,0000
15	51	1020	0,0000	0,0000	-25,0000
16	92	1840	0,0000	0,0000	-25,0000
17	48	960	0,0000	0,0000	-25,0000
18	69	1380	0,0000	0,0000	-25,0000
19	59	1180	0,0000	0,0000	-25,0000
20	55	1100	0,0000	0,0000	-25,0000
ORTALAMA	**63,15**	**1263**			**-25,0000**

Bu teknikte PSO gibi 20 kez çalıştırılmış ve Çizelge 3.6 elde edilmiştir. PSOPC için en iyi sonuca ulaşmak için ortalama iterasyon sayısı 63 olmuştur. Amaç fonksiyonu değerlerinin ortalaması için -25 değerini vermiştir.

Çizelge 3.7 Genetik Algoritma Tekniği ile 20 kez algoritmanın çalıştırılması

GA					
Çalıştırma Sayısı	**En İyi Değerin Bulunduğu İterasyon**	**Değerlendirilme Sayısı (Popülasyon Sayısı x İterasyon Sayısı)**	**x1 Degeri**	**x2 Degeri**	**Amaç Fonksiyonu Değeri**
1	100	2000	0,1439	0,3596	-24,8500
2	100	2000	-0,0141	0,4170	-24,8259
3	100	2000	-0,2325	0,0340	-24,9448
4	100	2000	0,1813	-0,3022	-24,8758
5	100	2000	0,2506	0,0049	-24,9372
6	100	2000	0,0667	-0,0665	-24,9911
7	100	2000	0,2869	-0,0316	-24,9167
8	100	2000	0,1281	0,1133	-24,9708
9	100	2000	-0,1559	-0,0098	-24,9756
10	100	2000	0,0137	0,1294	-24,9831
11	100	2000	-0,1990	-0,0113	-24,9603
12	100	2000	0,0041	-0,0885	-24,9922
13	100	2000	0,0486	0,0385	-24,9962
14	100	2000	0,0382	-0,0852	-24,9913
15	100	2000	0,0537	0,6450	-24,5810
16	100	2000	0,0419	0,4675	-24,7797
17	100	2000	0,0545	-0,1310	-24,9799
18	100	2000	-0,3265	-0,0198	-24,8930
19	100	2000	0,1421	-0,0178	-24,9795
20	100	2000	-0,1640	-0,0162	-24,9728
ORTALAMA	**100,00**	**2000**			**-24,9198**

Genetik Algoritma için ortalama iterasyon sayısı 100 olmuştur. Amaç fonksiyonu değerlerinin ortalaması -25 beklenirken bu algoritma -24,9198 sonucunu vermiştir.

GA toolbox'ı içerisinde default değeri değiştirilen çalıştırma kriterleri;

Population Size: 20

Initial Range: [-30;30]

Mutation Function: Gaussian

Stopping Criteria;

Generation: 100

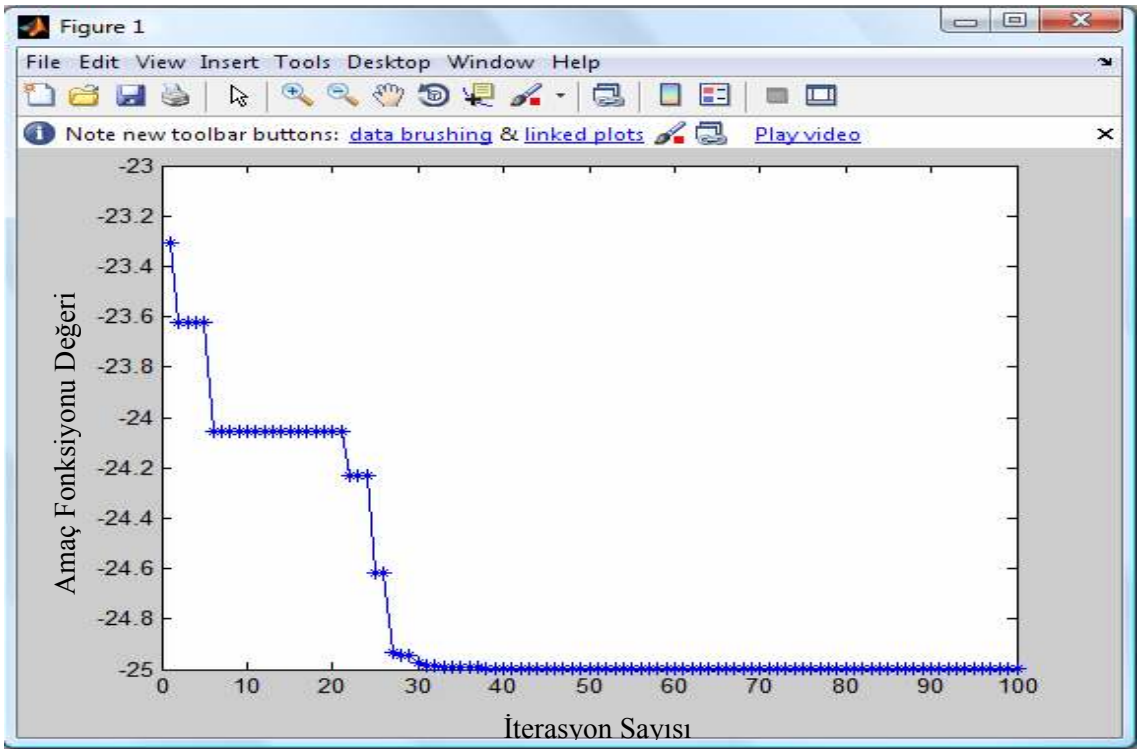
Stall Generation: 100

İterasyon sayısının ortalama 100 çıkmasının sebebi; durma kriterlerinde stall generation değerinin 100 seçilmesidir. Bu değer son 100 iterasyondaki en iyi değer bulunana kadar programı durdurma anlamına gelmektedir.

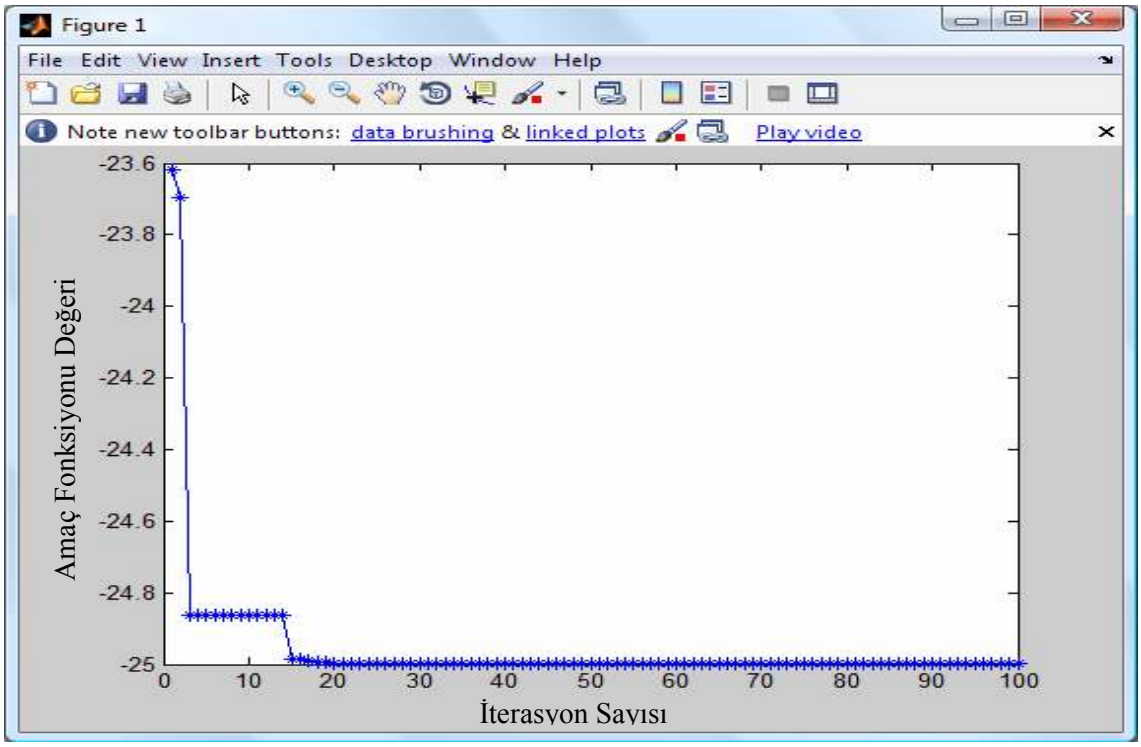
Çizelge 3.8 PSO, PSOPC ve GA karşılaştırılması

	PSO		**PSOPC**		**GA**	
BİR LOKAL BİR GLOBALOPTİMUMU OLAN FONKS.	**Amaç Fonksiyon Değeri Ortalaması**	**Amaç Fonksiyonunun Değerlendirilme Ortalaması**	**Amaç Fonksiyon Değeri Ortalaması**	**Amaç Fonksiyonunun Değerlendirilme Ortalaması**	**Amaç Fonksiyon Değeri Ortalaması**	**Amaç Fonksiyonunun Değerlendirilme Ortalaması**
	-24,9999	1746	-25,0000	1263	-24,9198	2000
	PSOPC'ye kıyasla				**PSOPC'ye kıyasla**	
	PSOPC, Amaç Fonksiyon Değeri Ortalamalarında %0,0003 daha verimli	PSOPC, Ortalama Fonksiyonun Değerlendirme Sayılarına göre %27,7 daha verimli	---	---	PSOPC, Amaç Fonksiyon Değeri Ortalamalarında %0,3205 daha verimli	PSOPC, Ortalama Fonksiyonun Değerlendirme Sayılarına göre %36,85 daha verimli

Çizelge 3.8'de de görüldüğü gibi bulunan amaç fonksiyonu değerleri üç algoritma da da global optimuma eşit yada çok yakın sonuçlar vermiştir. Algoritmalar iterasyon sayılarına göre ele alındığında PSOPC'nin gözle görünür derecede diğer yöntemlere fark attığı tespit edilmiştir.



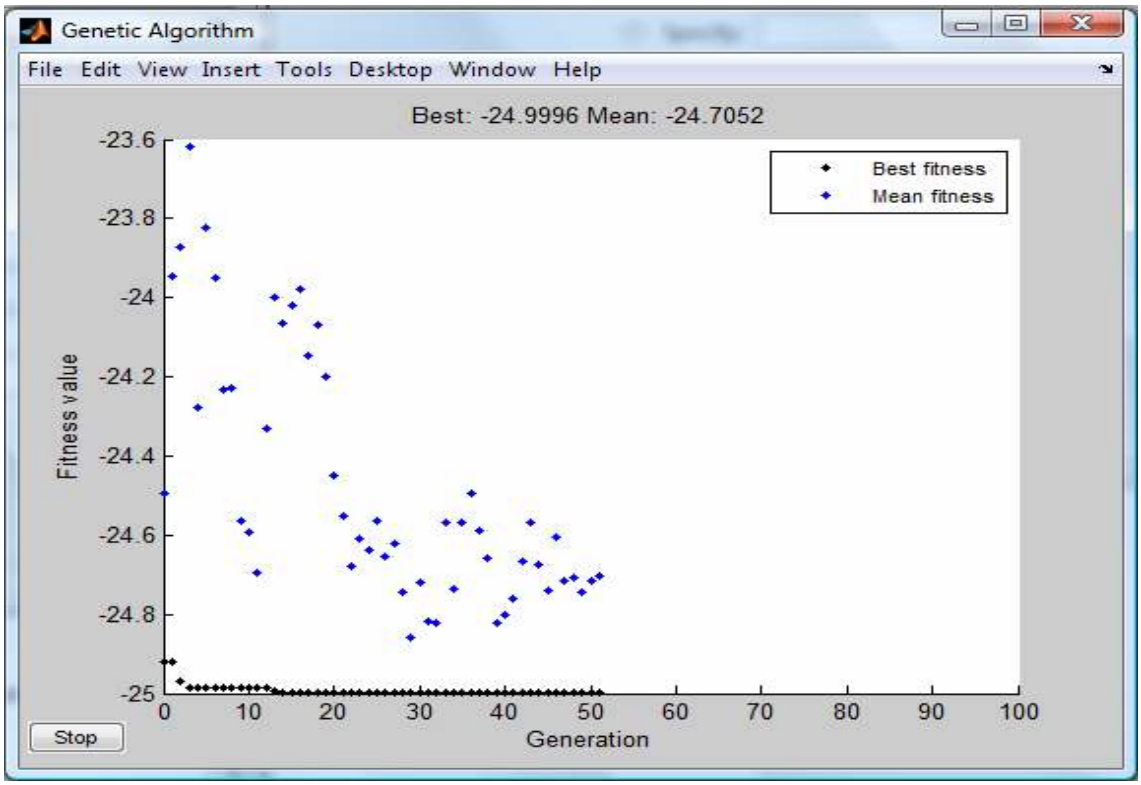
Şekil 3.6 PSO amaç fonksiyonu değeri – iterasyon sayısı grafiği



Şekil 3.7 PSOPC amaç fonksiyonu değeri – iterasyon sayısı grafiği

Şekil 3.6'da Parçacık Sürüsü Optimizasyon grafik çıktısı bulunmaktadır. Şekil 3.7'de ise Pasif Katılımlı Parçacık Sürüsü Optimizasyon grafik çıktısı gösterilmektedir. Bu iki grafik incelendiğinde PSO'ye ilave edilen pasif katılımın algoritmayı daha az iterasyonda daha iyi sonuca götürdüğü görülmektedir. Bahsedilen avantaj Çizelge 3.10'da sayısal verilere dayalı olarak verilmektedir. İki gösterim sonucunda PSOPC'nin PSO'dan daha güvenilir sonuçlar verdiği tespit edilmektedir.

Şekil 3.8'de aynı fonksiyonun çözümü sonucu programın verdiği grafik çıktısı bulunmaktadır.



Şekil 3.8 GA amaç fonksiyonu değeri – iterasyon sayısı grafiği

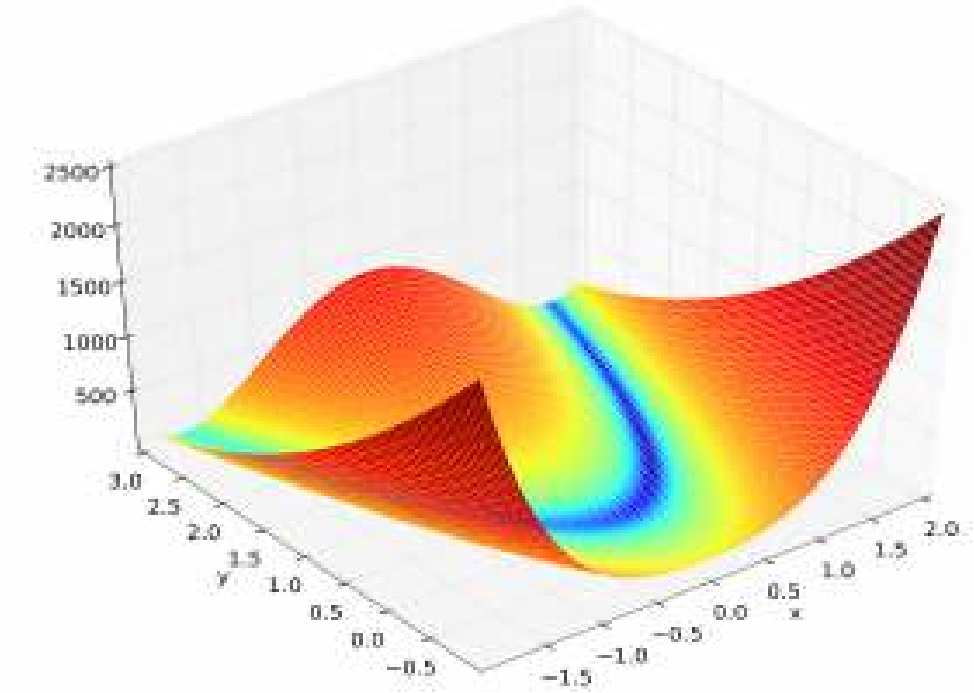
3.4 Rosenbrock Fonksiyonu İncelemesi

Rosenbrock Fonksiyonu; Matematik Optimizasyonunda konveks olmayan Optimizasyon Algoritmalarının performanslarını test etmek amacıyla 1960'ta Rosenbrock tarafından geliştirilmiş olan bir fonksiyondur. Rosenbrock Vadisi(Rosenbrock's valley) ya da Rosenbrock Muz Fonksiyonu(Rosenbrock's banana function) şeklinde de bilinmektedir (İnt. Kyn. 2).

Şu şekilde tanımlanır:

$$f(x_1, x_2) = (1 - x_1)^2 + 100\,(x_2 - x_1^2)^2 \qquad (3,4)$$

Koordinat düzleminde fonksiyona minimum değer veren alan; uzun, dar, parabolik şekilli bir düz vadiye benzemektedir. Fonksiyonun global minimumunu veren noktaları (x,y) = (1,1)'dir. Ayrıca bu koordinatlarda fonksiyon 0 değerini alır. Fonksiyonun grafiksel gösterimi Şekil 3.9'da verilmektedir.



Şekil 3.9 Rosenbrock Fonksiyonunun global ve lokal çözümlerinin gösterimi

Bu fonksiyon ele alınan diğer fonksiyonlara göre oldukça zor bir fonksiyon olduğu gözlemlenmiştir. Kod yazılımı sonra PSO, PSOPC ve GA'nın verdiği sonuçlar bu gözlemin sağlanmasında etken olmuştur.

Rosenbrock fonksiyonun zor bir fonksiyon olması sebebiyle diğer fonksiyonlarda kullanılan 20 populasyon ve 100 iterasyon bu fonksiyon için yeterli gelmemektedir. Sonuç olarak bu fonksiyonun değerlendirilmesinde algoritmalar 50 populasyon ve 200 iterasyonla çalıştırılmıştır. Değişkenlerin başlangıç değerleri normal dağılıma uyacak şekilde ortalaması 0 ve standart sapması 10 olarak random alınmıştır. Buna göre değişkenler ±30 aralığında %99,7olasılıkla değişecektir. Sırasıyla PSO, PSOPC ve GA sonuçları çizelgelerde verilecektir.

Çizelge 3.9 Rosenbrock Fonksiyonunun PSO ile 20 kez algoritmanın çalıştırılması

PSO					
Çalıştırma Sayısı	**En İyi Değerin Bulunduğu ilk İterasyon**	**Değerlendirilme Sayısı (Popülasyon Sayısı x İterasyon Sayısı)**	**x1 Degeri**	**x2 Degeri**	**Amaç Fonksiyonu Değeri**
1	184	3680	1,0229	1,0514	0,0031
2	63	1260	1,0376	1,0775	0,0015
3	180	3600	1,4077	1,9841	0,1669
4	119	2380	0,9698	0,9407	0,0009
5	55	1100	0,8886	0,7882	0,0126
6	90	1800	0,9398	0,8830	0,0036
7	50	1000	0,9813	0,9625	0,0004
8	104	2080	1,0279	1,0570	0,0008
9	181	3620	0,9351	0,8746	0,0042
10	189	3780	0,6642	0,4304	0,1244
11	179	3580	1,1423	1,3058	0,0203
12	61	1220	1,7995	3,2365	0,6394
13	109	2180	0,9842	0,9674	0,0004
14	179	3580	0,8944	0,8020	0,0115
15	71	1420	1,0122	1,0263	0,0004
16	172	3440	1,0030	1,0064	0,0000
17	52	1040	0,9911	0,9808	0,0003
18	128	2560	1,1201	1,2545	0,0144
19	17	340	1,0836	1,1745	0,0070
20	54	1080	1,0193	1,0380	0,0005
ORTALAMA	**111,85**	**2237**			**0,0506**

Parçacık Sürüsü Optimizasyonu ile algoritma 20 kez çalıştırılmış ve Çizelge 3.9'daki verilere ulaşılmıştır. Ortalama 112 iterasyonda en iyi sonuçlara ulaşmaktadır. Amaç fonksiyonu değerlerinin ortalaması 0 beklenirken bu algoritma 0,0506 sonucunu vermiştir.

Çizelge 3.10 Rosenbrock Fonksiyonunun PSOPC ile 20 kez algoritmanın çalıştırılması

PSOPC					
Çalıştırma Sayısı	**En İyi Değerin Bulunduğu ilk İterasyon**	**Değerlendirilme Sayısı (Popülasyon Sayısı x İterasyon Sayısı)**	**x1 Degeri**	**x2 Degeri**	**Amaç Fonksiyonu Değeri**
1	192	3840	1,0218	1,0360	0,0071
2	120	2400	1,0086	1,0124	0,0025
3	139	2780	0,9690	0,9412	0,0014
4	127	2540	1,0198	1,0357	0,0022
5	148	2960	0,9893	0,9748	0,0016
6	200	4000	1,0143	1,0261	0,0009
7	123	2460	0,9858	0,9745	0,0009
8	186	3720	0,9219	0,8451	0,0083
9	200	4000	1,0307	1,0630	0,0010
10	164	3280	1,0386	1,0783	0,0015
11	191	3820	1,0412	1,0835	0,0017
12	109	2180	1,0024	1,0039	0,0001
13	112	2240	1,0337	1,0622	0,0051
14	194	3880	0,9855	0,9697	0,0004
15	175	3500	0,9827	0,9625	0,0013
16	38	760	1,0156	1,0305	0,0003
17	175	3500	1,0231	1,0466	0,0005
18	46	920	1,0038	1,0083	0,0001
19	152	3040	1,0332	1,0602	0,0063
20	151	3020	0,9929	0,9854	0,0001
ORTALAMA	**147,10**	**2942**			**0,0022**

Bu teknikte PSO gibi 20 kez çalıştırılmış ve Çizelge 3.10 elde edilmiştir. PSOPC için en iyi sonuca ulaşmak için ortalama iterasyon sayısı 147 olmuştur. Amaç fonksiyonu değerlerinin ortalaması için 0,0022 değerini vermiştir.

Çizelge 3.11 Rosenbrock Fonksiyonunun GA ile 20 kez algoritmanın çalıştırılması

GA					
Çalıştırma Sayısı	**En İyi Değerin Bulunduğu İterasyon**	**Değerlendirilme Sayısı (Popülasyon Sayısı x İterasyon Sayısı)**	**x1 Degeri**	**x2 Degeri**	**Amaç Fonksiyonu Değeri**
1	151	3020	2,1507	4,6033	1,3735
2	151	3020	0,5969	0,4262	0,6501
3	151	3020	1,0125	1,0584	0,1101
4	151	3020	-1,5500	2,3684	6,6194
5	151	3020	-1,1645	1,3426	4,7028
6	151	3020	-0,3787	0,1653	1,9487
7	151	3020	0,0994	0,0010	0,8195
8	151	3020	1,1173	1,3013	0,2931
9	151	3020	0,9813	0,8237	1,9399
10	151	3020	1,0570	1,1760	0,3476
11	151	3020	0,9698	0,9531	0,0165
12	151	3020	0,4292	0,1594	0,3874
13	151	3020	1,9263	3,7666	1,1723
14	151	3020	2,5488	6,5925	3,3190
15	151	3020	2,5147	6,4470	3,8098
16	151	3020	2,0826	4,4474	2,3832
17	151	3020	2,1256	4,5423	1,3248
18	151	3020	0,3973	0,2256	0,8227
19	151	3020	0,1012	0,0090	0,8079
20	151	3020	0,7240	0,5772	0,3574
ORTALAMA	**151,00**	**3020**			**1,6603**

Genetik Algoritma için ortalama iterasyon sayısı 151 olmuştur. Amaç fonksiyonu değerlerinin ortalaması 0 beklenirken bu algoritma 1,6603 sonucunu vermiştir. Çizelge 3.11' çalıştırma sonuçları verilmektedir.

GA toolbox'ı içerisinde default değeri değiştirilen çalıştırma kriterleri;

Population Size: 50

Initial Range: [-30;30]

Mutation Function: Gaussian

Stopping Criteria;

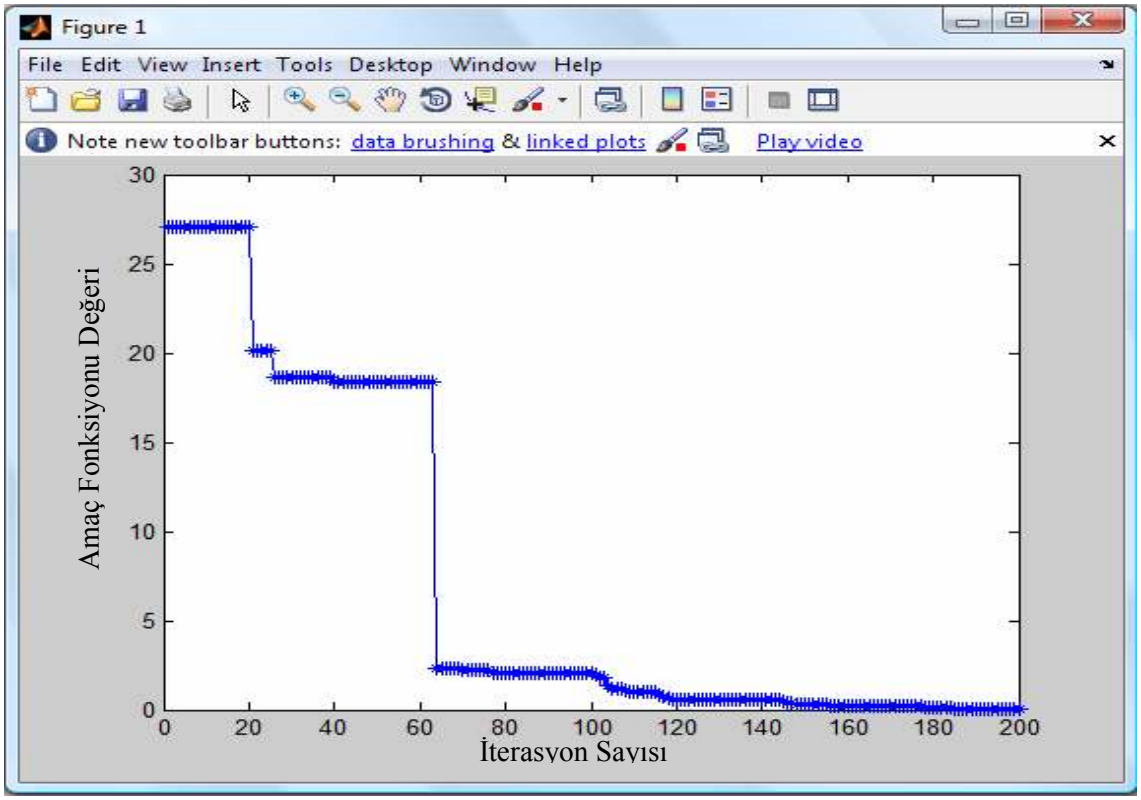
Generation: 200

Stall Generation: 150

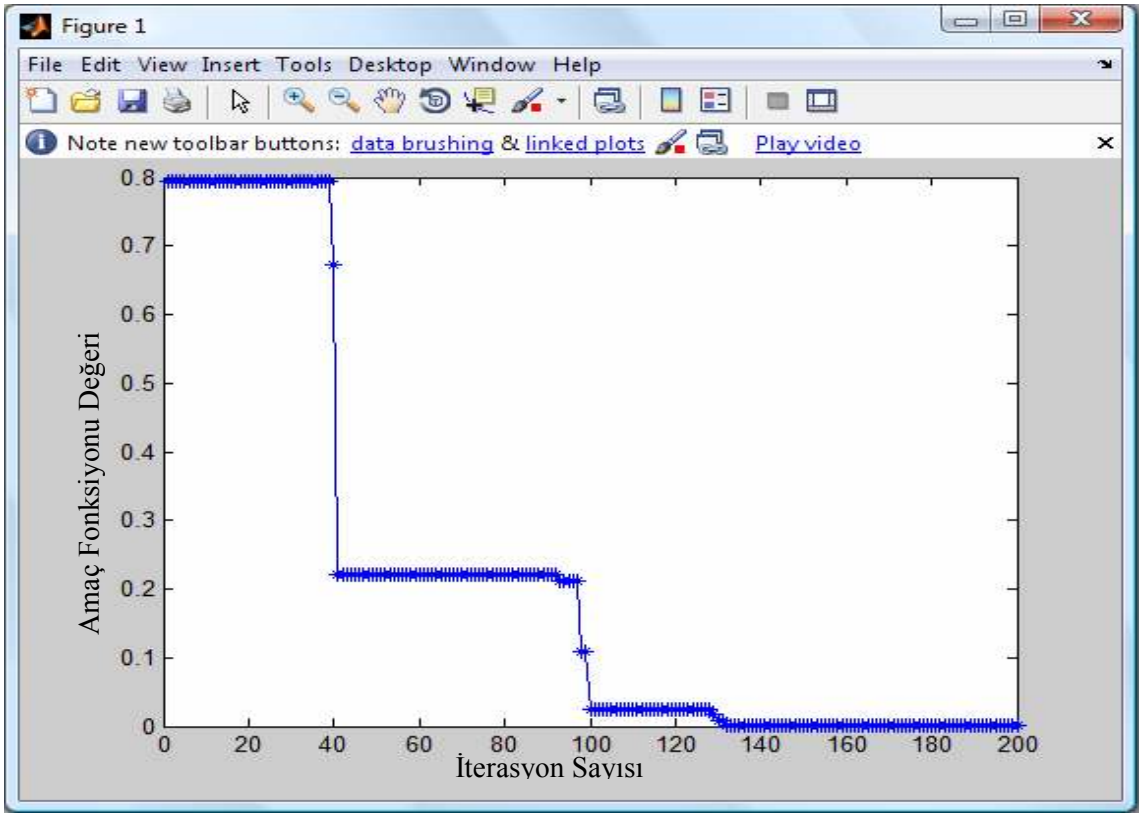
Çizelge 3.12 Rosenbrock Fonksiyonunun PSO, PSOPC ve GA ile karşılaştırılması

	PSO		PSOPC		GA	
ROSENBROCK FONKSİYONU	**Amaç Fonksiyon Değeri Ortalaması**	**Amaç Fonksiyonun un Değerlendiril-me Ortalaması**	**Amaç Fonksiyon Değeri Ortalaması**	**Amaç Fonksiyonun un Değerlendiril-me Ortalaması**	**Amaç Fonksiyon Değeri Ortalaması**	**Amaç Fonksiyonun un Değerlendiril-me Ortalaması**
	0,0506	2237	0,0022	2942	1,6603	3020
	PSOPC'ye kıyasla				**PSOPC'ye kıyasla**	
	PSOPC, Amaç Fonksiyon Değeri Ortalamaları nda %95,65 daha verimli	PSO, Ortalama Fonksiyonun Değerlendir-me Sayılarına göre PSOPC'den %23,96 daha verimli	---	---	PSOPC, Amaç Fonksiyon Değeri Ortalamaların da %99,86 daha verimli	PSOPC Amaç Fonksiyonu Değerlendiril-me Ortalamasında GA'ya göre %2,58 daha verimli

Bu fonksiyon kullanılarak yapılan algoritmaların testi sonucu PSOPC ve GA birbirlerine çok yakın değerler vermiştir. PSO daha az iterasyonda minimumunu hesaplamış fakat bu algoritmadan PSOPC ve GA'nin verdiği değerler kadar iyi değerler alınamamıştır. Kıyaslamaya dair sayısal veriler Çizelge 3.12'de verilmektedir.



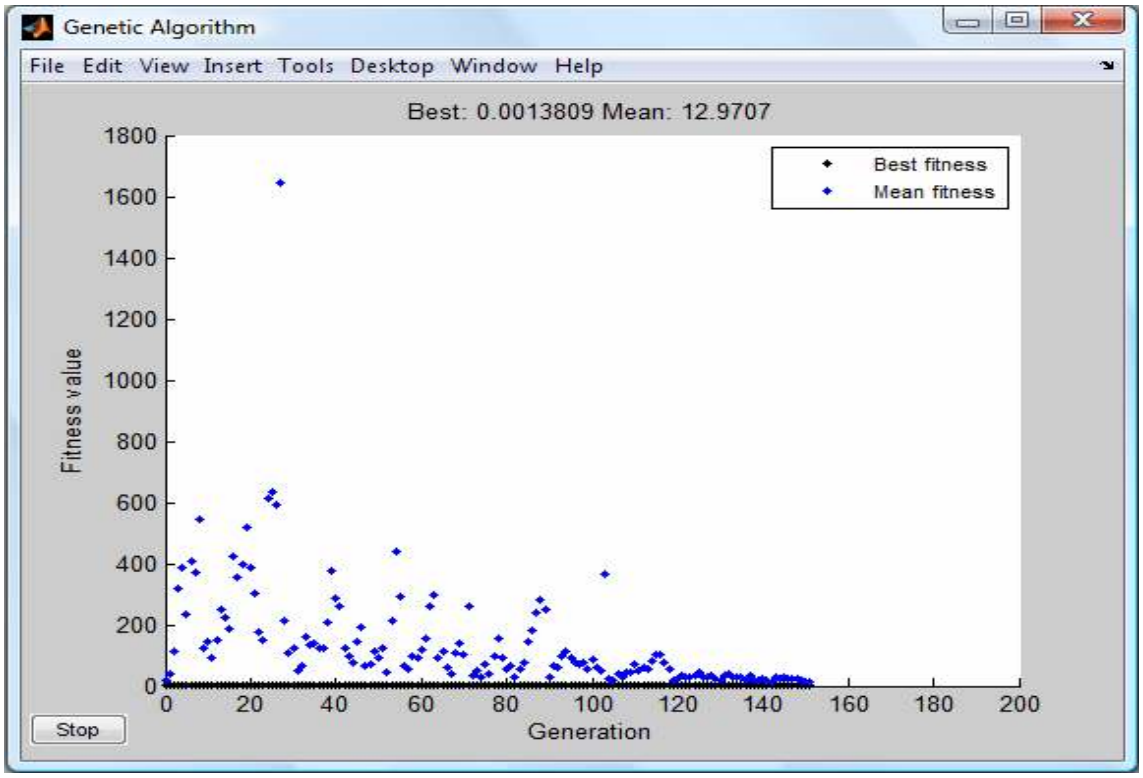
Şekil 3.10 Rosenbrock Fonks. PSO amaç fonksiyonu değeri–iterasyon sayısı grafiği



Şekil 3.11 Rosenbrock Fonks. PSOPC amaç fonksiyonu değeri – iterasyon sayısı grafiği

Şekil 3.10'da Parçacık Sürüsü Optimizasyon grafik çıktısı bulunmaktadır. Şekil 3.11'de ise Pasif Katılımlı Parçacık Sürüsü Optimizasyon grafik çıktısı gösterilmektedir. Bu iki grafik incelendiğinde PSO'ye ilave edilen pasif katılımın algoritmayı daha az iterasyonda daha iyi sonuca götürdüğü görülmektedir. Bahsedilen avantaj Çizelge 3.12'de sayısal verilere dayalı olarak verilmektedir. İki gösterim sonucunda PSOPC'nin PSO'dan daha güvenilir sonuçlar verdiği tespit edilmektedir.

Şekil 3.12'de aynı fonksiyonun çözümü sonucu programın verdiği grafik çıktısı bulunmaktadır.



Şekil 3.12 Rosenbrock Fonks. GA amaç fonksiyonu değeri – iterasyon sayısı grafiği

3.5 Algoritma Yazılımının Değerlendirilmesi

Yazılımın testi için üç ayrı özel fonksiyonda çalışmalar yapılmıştır. Literatürde bu fonksiyonlar optimizasyon algoritmalarının performans testleri için kullanılmaktadırlar.

Çizelge 3.13'de bulunan verilerden yola çıkarak Pasif Katılımlı Parçacık Sürü Optimizasyonunun, Genetik Algoritma ve Temel Parçacık Sürü Optimizasyonuna göre daha iyi sonuçlar verdiği görülmektedir.

Bu değerlendirmeler doğrultusunda yazılıma ve algoritmaya dair test işlemi olumlu sonuçlandığı ve yazılım ve algoritmanın Yapısal Tasarım alanında incelenecek olan problemlerin çözümünde rahatlıkla kullanılabileceği sonucuna ulaşılmaktadır.

Çizelge 3.13 Üç Ayrı Fonksiyonda Algoritmaları Kıyaslanması

	Bilinen Global Optimum Amaç Fonksiyon Değeri	**PSO**		**PSOPC**		**GA**	
		Amaç Fonksiyon Değeri Ortalaması	**İterasyon Sayısı Ortalaması**	**Amaç Fonksiyon Değeri Ortalaması**	**İterasyon Sayısı Ortalaması**	**Amaç Fonksiyon Değeri Ortalaması**	**İterasyon Sayısı Ortalaması**
RASTRIGIN FONKSİYONU	**0,0000**	0,0002	65,85	0,0000	43,45	2,0295	100
BİR LOKAL BİR GLOBAL OPTİMUMU OLAN FONKSİYON	**-25,0000**	-24,9999	87,3	-25,0000	63,15	-24,9198	100
ROSENBROCK FONKSİYONU	**0,0000**	0,0506	111,85	0,0022	147,1	1,6603	151

5. YAPISAL TASARIM

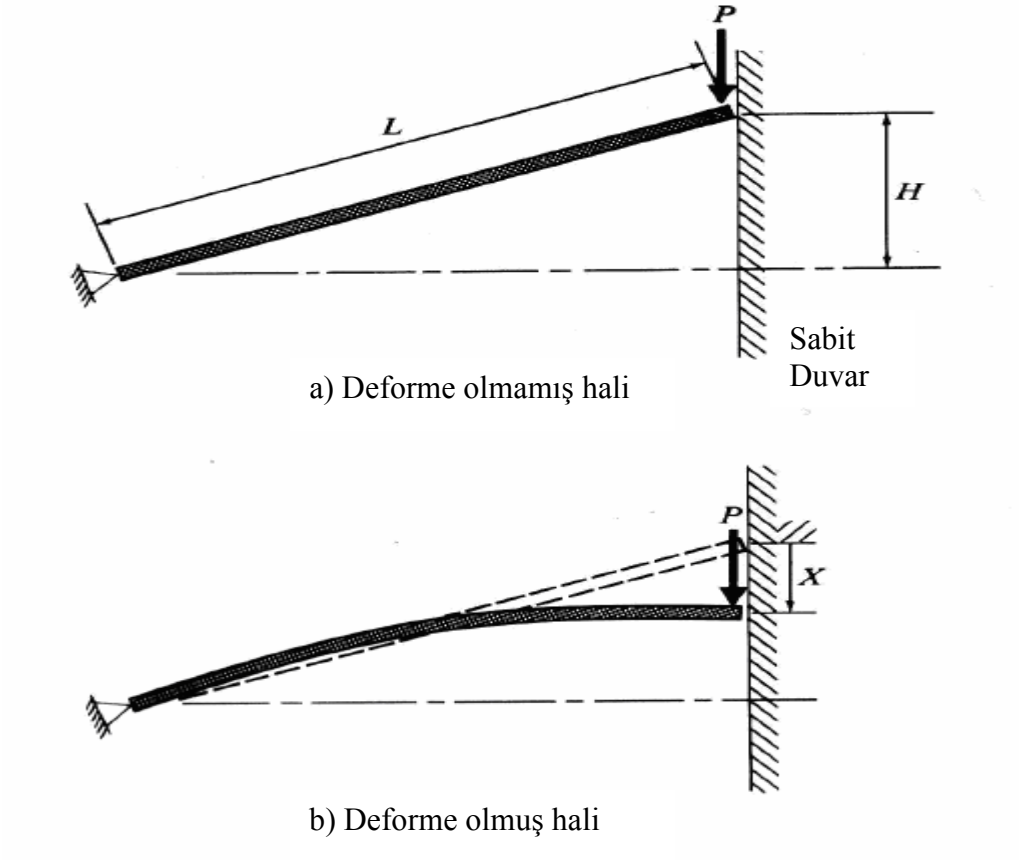
Endüstriyel isletmelerde yüksek rekabet gücüne sahip ürünleri üretebilmek için tasarım ve imalat teknolojilerinde yapısal değişimlere gereksinim vardır. Üretim yeteneği önemlidir ve geliştirilmelidir, ancak tasarım ve tasarım doğrulama, teknoloji edinim ve teknoloji geliştirme yeteneklerine sahip olunduğu oranda üretme yeteneği değerli olmaktadır. Üretim yeteneğinin yanında tasarım, tasarım doğrulama yeteneği ve temel teknolojilerin edinilmesi ve geliştirilmesi gerekmektedir. Bu nedenle, yeni teknolojilerin ve yöntemlerin tasarım ve tasarım doğrulama aşamalarında klasik yöntemlerin yerine kullanılması, teknolojiye dayalı rekabet ortamının oluşturulabilmesi için önem taşımaktadır (İnt. Kyn. 7) .

Algoritma Yazılımı ve Testleri başlığı adı altında Genetik Algoritma, Parçacık Sürü Optimizasyonu ve Pasif Katılımlı Parçacık Sürü Optimizasyonu teknikleri kıyaslanmış olup Pasif Katılımlı Parçacık Sürü Optimizasyonu için yazılan kodların ve bu tekniğin diğerlerine göre avantajları incelenmiştir.

Bu tespit doğrultusunda Yapısal Tasarım için incelenecek olan 2 problem tekrar Genetik Algoritma ve Parçacık Sürü Optimizasyonu ile ele alınmayacak olup Parçacık Sürü Optimizasyonu ile çözümleri yapılacak ve çözümler literatürde bulunan en iyi sonuçlarla kıyaslanacaktır.

5.1 Kiriş Problemi

Bu kiriş probleminde kirişin bir ucu sabitlenmiş, diğer ucu ise sabit duran bir duvara dayanmıştır. Kirişin duvara dayalı olan ucuna bir P yükü uygulanacak ve kırılmadan taşıyabileceği maksimum P yükü araştırılacaktır. Kiriş Şekil 4.1 de verilmektedir.



Şekil 4.1 Kirişin deforme olmuş ve olmamış halinin gösterimi

Şekil 4.1a'da kirişe uygulanan P yükü sıfırdır. Bu durumda kiriş herhangi bir deformasyona uğramamıştır. Şekil 4.1b'de ise kirişe sıfırdan büyük bir P yükü uygulanmaktadır. Uygulanan P yükü sonucunda kirişte şekil değiştirmeler meydana gelmektedir. Bu problemin ele alınmasındaki amaç bu şekil değiştirme sonucunda kirişin kırılmadan kaldırabileceği maksimum P yükünün bulunmasıdır. Aynı zamanda maksimum P yükü hesaplamasında kirişin duvarda bulunduğu ilk nokta ile ikinci nokta arasındaki uzaklık olan x de belirlenecektir. Bu x değeri maksimum P yükü ile kirişin duvarda hangi noktaya kadar kırılmadan ilerleyebileceğini gösterecektir.

Kiriş problemine dair oluşturmuş model (Vanderplaats 1984);

Eksenel yük F ve kritik burkulma yükü Fcr aşağıda verilmiştir.

$$F_1 = AE\,[1 - (1 + (X^2 - 2HX)/L^2)^{(0,5)}\,]^2 \qquad (4.1)$$

$$Fcr = \frac{\pi^2 E I}{L^2} \qquad (4.2)$$

Probleme dair amaç fonksiyonu şu şekildedir;

$$P = \frac{(H - X)\,F}{(L2 + X2 - 2HX\ )^{(0,5)}} \qquad (4.3)$$

$$F = \min (F_1, F_{cr})$$

Ayrıca çözümde $0 \leq X \leq H$ kısıtı dikkate alınacaktır.

Formülasyonlarda kullanılan ifadeler;

F_1= Eksenel Kuvvet

Fcr = Burkulma Yükü

H = Kirişin iki ucu arasındaki yükseklik farkı

X = Kirişin duvardaki ucunun yer değiştirmesi

L = Kiriş Uzunluğu

A= Kiriş Kesiti

I = Atalet Moment

E = Elastisite Modülü

Kirişe uygulanabilir maksimum P yükünün hesaplanmasında optimizasyon modelini direkt olarak etkileyecek tek bağımsız değişken X değişkenidir.

Probleme ait sayısal veriler aşağıdaki gibi alınmıştır.

H = 25.0 cm = 0.25 m

L = 250.0 cm = 2.5 m

A = 25.0 cm^2 = 25 x 10^{-4} m^2

I = 750.0 cm^4 = 750 x 10^{-8} m^2

E = 70 GPa = 70 x 10^9 N/m^2

Çizelge 4.1 Kiriş Probleminin PSOPC ile çalıştırılması

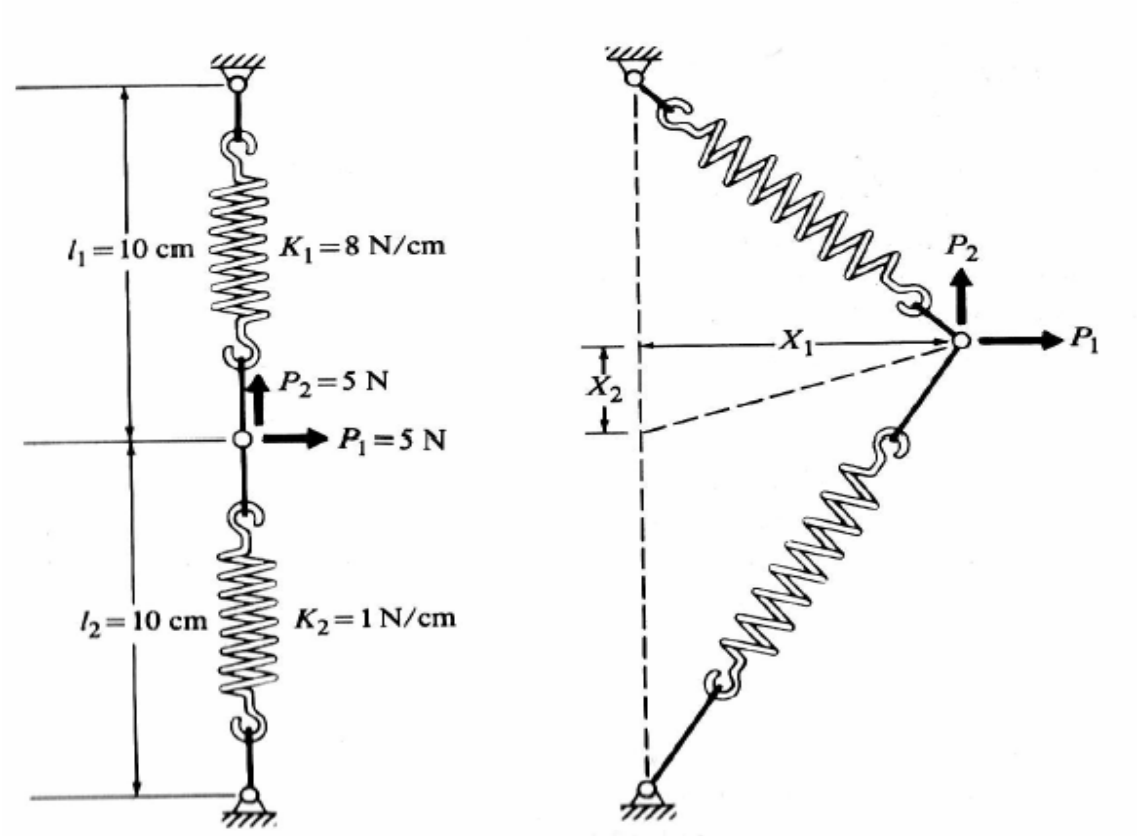
PASİF KATILIMLI PARÇACIK SÜRÜ OPTİMİZASYONU				
Çalıştırma Sayısı	**En İyi Değerin Bulunduğu İlk İterasyon**	**Değerlendirilme Sayısı (Popülasyon Sayısı x İterasyon Sayısı)**	**x Degeri**	**Amaç Fonksiyonu Değeri**
1	50	1000	0,10550538	33847,85502164
2	2	40	0,10504343	33846,43362086
3	2	40	0,10551479	33847,87312615
4	65	1300	0,10542119	33847,67379679
5	34	680	0,10497931	33846,15372555
6	2	40	0,10648262	33847,42906409
7	2	40	0,10676938	33846,42280924
8	84	1680	0,10651357	33847,33965573
9	61	1220	0,10540686	33847,63950426
10	65	1300	0,10613903	33848,10891585
11	5	100	0,10618958	33848,04490529
12	75	1500	0,10674255	33846,53387290
13	55	1100	0,10648180	33847,43136422
14	2	40	0,10590507	33848,24315696
15	58	1160	0,10605934	33848,18457529
16	73	1460	0,10562035	33848,04657497
17	71	1420	0,10552781	33847,89747337
18	2	40	0,10682117	33846,19859928
19	85	1700	0,10629415	33847,87304002
20	62	1240	0,10587919	33848,24162532
ORTALAMA	**42,75**	**855**	**0,10596483**	**33847,48122139**

Algoritma 20 parçacık ve 100 iterasyonla çalıştırılmıştır. 20 kez çalıştırılan algoritmanın sonuçları Çizelge 4.1'de verilmiştir. Amaç Fonksiyonu Değeri için hesaplanan değerlerin ortalaması 33847,4812'dir. Bu değere ortalama 42,75 iterasyonla ulaşılmıştır. Buna bağlı olarak amaç fonksiyonu değerlendirmesi 855 kez olmaktadır.

Bu problem için literatürde bulunan en iyi sonuç P=33848 N'dur. PSOPC'de ise bu değer ortalama 33847'dir. 20 çalıştırma içerisinde 33848,2431 değerini almış bir parçacık da vardır. Bu veriler sonucunda algoritmanın başarısı tekrar gözlemlenmektedir.

5.2 Yay Problemi

Birer uçlarından sabit diğer uçları hareketli mekanizmayla birbirine bağlı olan iki yayın bulunduğu düzeneğe ait problem ele alınacaktır. Hareketli mekanizmaya bir yatay bir dikey kuvvet uygulanacak, denge hali mekanizmanın potansiyel enerjisini en küçük yaparak bulunacaktır.



Şekil 4.2 Yay Problemi

Yay problemine ait potansiyel enerji formülü aşağıdadır (Vanderplaats 1984).

$$PE = \tfrac{1}{2} K_1 (\Delta l_1)^2 + \tfrac{1}{2} K_2 (\Delta l_2)^2 - P_1X_1 - P_2X_2 \quad (4.4)$$

veya şu şekilde yeniden düzenlenebilir;

$$PE = \tfrac{1}{2} K_1 ((X_1^2 + (l_1 - X_2)^2)^{0,5} - l_1)^2 + \tfrac{1}{2} K_2 ((X_1^2 + (l_2 - X_2)^2)^{0,5} - l_1)^2 - P_1X_1 - P_2X_2 \quad (4.5)$$

Formülasyonlarda kullanılan ifadeler;

l_1 = 1.Yayın uzunluğu
l_2 = 2.Yayın uzunluğu
K_1 = Yay katsayısı
K_2 = Yay katsayısı
P_1 = Yatayda uygulanan kuvvet
P_2 = Dikeyde uygulanan kuvvet
X_1 = Yataydaki yer değiştirme
X_2 = Dikeydeki yer değiştirme

Mekanizmanın minimum potansiyel enerjisinin hesaplanmasında X_1 ve X_2 değişkenleri probleme ait iki bağımsız değişkendir. Bu iki değişken için uygun değerler bulunması sonucu optimum potansiyel enerji hesaplanabilecektir.

Formülasyonlarda kullanılan sayısal veriler;

l_1 = 10 cm
l_2 = 10 cm
K_1 = 8 N/m^2
K_2 = 1 N/m^2
P_1 = 5 N
P_2 = 5 N

Çizelge 4.2 Yay Probleminin PSOPC ile çalıştırılması

PASİF KATILIMLI PARÇACIK SÜRÜ OPTİMİZASYONU					
Çalıştırma Sayısı	**En İyi Değerin Bulunduğu İlk İterasyon**	**Değerlendirilme Sayısı (Popülasyon Sayısı x İterasyon Sayısı)**	**x1 Degeri**	**x2 Degeri**	**Amaç Fonksiyonu Değeri**
1	23	460	8,60514610	4,56722322	-41,80079301
2	190	3800	8,60347983	4,53316552	-41,80552887
3	89	1780	8,56480022	4,46670470	-41,80145730
4	89	1780	8,58540722	4,49426646	-41,80482512
5	196	3920	8,62043770	4,58310208	-41,80159872
6	47	940	8,51861525	4,47324926	-41,78408042
7	91	1820	8,68722865	4,52181910	-41,79661295
8	2	40	8,63209109	4,45812415	-41,79929726
9	172	3440	8,68036471	4,79002287	-41,73536347
10	26	520	8,70600191	4,59952994	-41,79980398
11	96	1920	8,63320030	4,48524078	-41,80448970
12	3	60	8,63944193	4,60163669	-41,80186881
13	59	1180	8,57010903	4,52279369	-41,79784681
14	185	3700	8,60715924	4,43846565	-41,79958254
15	152	3040	8,65415300	4,64135136	-41,79526330
16	162	3240	8,61776137	4,55108438	-41,80608492
17	27	540	8,64164433	4,57157526	-41,80662964
18	43	860	8,70562958	4,65037152	-41,79684774
19	3	60	8,55578869	4,44209369	-41,79925127
20	164	3280	8,71557534	4,67339435	-41,79236966
ORTALAMA	**90,95**	**1819**	**8,62720177**	**4,55326073**	**-41,79647978**

Bu problem bir önceki kiriş problemine göre daha zor olduğu için 100 parçacık ve 200 iterasyonla çalıştırılmıştır. Çizelge 4.2'de de görüldüğü gibi algoritmanın 20 ayrı çalıştırılması sonucu hesaplanan amaç fonksiyonlarının ortalamaları olan -41,80 değerine yaklaşık 91 iterasyonla ulaşılmaktadır. Literatürde bulunan en iyi sonuç ise -41,81'dir.

5. SONUÇLAR

Bireylerin sosyal bilgi paylaşımı üzerine geliştirilen, populasyon tabanlı bir sezgisel yöntem olan Pasif Katılımlı Parçaçık Sürü Optimizasyonu ayrıntılı olarak incelenmiştir. Bunun için mevcut algoritmanın yanı sıra pasif katılımın geliştirilmesinden önceki hali olan Parçacık Sürüsü Optimizasyonu da ele alınmıştır. Ayrıca yine populasyon tabanlı ve sezgisel bir yöntem olan Genetik Algoritma da bu incelemeye dâhil olmuştur.

Pasif Katılımlı Parçacık Sürüsü Optimizasyonunun, Genetik Algoritma ve Parçacık Sürüsü Optimizasyonu ile kıyaslanmasının temel nedeni diğer tekniklere göre avantajının belirlenmesi ve kullanılabilinirliğinin derecesinin tespit edilmesidir. Algoritma yazılımı ve testi başlığı adı altında Pasif Katılımlı Parçacık Sürüsü Optimizasyonu algoritması ve MATLAB'da yazılan yazılımın güvenilirliği kontrol edilmiştir. Literatürde algoritmaların testi için kullanılan özel fonksiyonlar vardır. Bu fonksiyonlardan Rastrigin Fonksiyonu, bir lokal bir global optimumu olan fonksiyon ve Rosenbrock fonksiyonu seçilmiştir. Üç ayrı teknik için bahsedilen fonksiyonlar incelenmiş ve incelemeye dair sayısal veriler çizelgelerde verilmiştir.

Pasif Katılımlı Parçacık Sürüsü Optimizasyonunun özel fonksiyonlarda değerlendirilmesi sonucunda optimizasyon problemlerinin çözümünde olumlu sonuçlar verdiği görülmüştür. Bu başarıdan yola çıkarak algoritma ve yazılımın yapı tasarımına ait iki temel problem çeşidi üzerinde uygulanmasına karar verilmiştir. Bir kiriş ve bir yay probleminin çözümü yapılmış ve çıkan sonuçlar literatürde bulunan optimum değerlerle karşılaştırılmıştır. Bu karşılaştırmadan olumlu sonuçlar elde edilmiştir.

Bu veriler sonucunda bu algoritmanın pek çok alanda karşılaşılabilecek olan optimizasyon problemlerinin çözümünde kullanılabileceği ve istenen sonuçlara rahatça ulaşılabileceği öngörülmektedir.

6. KAYNAKLAR

Aslantaş V., Doğan A., Kurban R., 2007, "Parçacık Sürü Optimizasyonu ile DWT-SVD Tabanlı Resim Damgalama", Uluslar arası Katılımlı Bilgi Güvenliği ve Kriptoloji Konferansı

Bekler T., 2006 ,"Matlab Ders Notları" Canakkale Onsekiz Mart Universitesi Jeofizik Mühendisligi Bölümü

Der V., Vural R. A. , Yıldırım T., 2008, "Parçacık Sürü Optimizasyonu Tabanlı Evirici Tasarımı", Eleco'2008 Elektrik - Elektronik - Bilgisayar Mühendisliği Sempozyumu Ve Fuarı Bildirileri

Emel G. G., Taşkın Ç., 2002, "Genetik Algoritmalar Ve Uygulama Alanları", Uludağ Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi Cilt XXI, Sayı 1, s. 129-152

Esen Y., Ülker M., 2005, "Malzeme Bakımından Lineer Olmayan Çok Katlı Uzay Çerçevelerin Ansys İle Optimizasyonu", Gazi Üniv. Müh. Mim. Fak. Der. , Cilt 23, No 2, 485-494

He S., Wu Q. H., Wen J. Y., Saunders J.R., Paton R.C., 2004, "A particle swarm optimizer with passive congregation", BioSystems, Vol.87, pp. 135-147

Karahan O., Bingül Z., 2009, "Stewart Platformunun Parçacık Sürü Optimizasyonlu Bulanık Mantık Kontrolü", Tok'09 Otomatik Kontrol Ulusal Toplantısı Programı

Karakaya Ş., 2007, "Tabakalı Kompozit Plakların Gelişmiş Global Optimizasyon Teknikleriyle Yapısal Optimizasyonu", Yüksek Lisans Tezi, Afyon Kocatepe Üniversitesi, Fen Bilimleri Enstitüsü, Afyonkarahisar

Karakuzu C., 2007, "Parçacık Sürü Optimizasyonu İle Bulanık-Nöral Kontrolör Eğitimi Ve Benzetim Örnekleri", Elektrik Mühendisleri Odası IV.Otomasyon Sempozyumu

Kaveh A., Talatahari S., 2009, "A particle swarm ant colony optimization for truss structures with discrete variables", Journal of Constructional Steel Research, Vol. 65, pp. 1558-1568

Li L., Huang Z., Liu F.,2006," An Improved Particle Swarm Optimizer for Truss Structure Optimization", Computational Intelligence and Security, 2006 International Conference, Vol. 1, pp. 924 - 928

Tamer S., Karakuzu C., 2006, "Parçacık Sürüsü Optimizasyon Algoritması ve Benzetim Örnekleri", Eleco'2006 Elektrik - Elektronik - Bilgisayar Mühendisliği Sempozyumu Ve Fuarı Bildirileri

Vanderplaats G. N.,1984, "Numerical Optimization Techniques for Engineering Design: With Applications", McGraw-Hill , USA .

6.1 İnternet Kaynakları **Erişim Tarihleri**

1. http://www.elektrik.gen.tr/icerik/ant-system-algoritmas%C4%B1n%C4%B1n-java-ile-g%C3%B6rselle%C5%9Ftirilmesi 18.12.2010
2. http://en.wikipedia.org/wiki/Rosenbrock_function 19.11.2010
3. http://iridia.ulb.ac.be/~mmontes/slidesCIL/slides.pdf 22.10.2010
4. http://tr.wikipedia.org/wiki/Genetik_algoritma 25.11.2010
5. http://www.webhocam.net/Konuizle.asp?t=6741 25.11.2010
6. http://www.mechsoft.com.tr/Mechsoft/mechanic/tr/yapisalanaliz.html 18.12.2010
7. http://www.mmo.org.tr/resimler/dosya_ekler/84b6fbb10729ed4_ek.pdf?dergi=46 18.12.2010
8. http://taygunkekec.com/files/publications/SwarmRoboticsPaper.pdf 18.12.2010

ÖZGEÇMİŞ

Adı Soyadı	HACER ARIOL
Doğum Yeri	AFYONKARAHİSAR
Doğum Tarihi	21/01/1985
Medeni Hali	BEKAR
Yabancı Dili	İNGİLİZCE

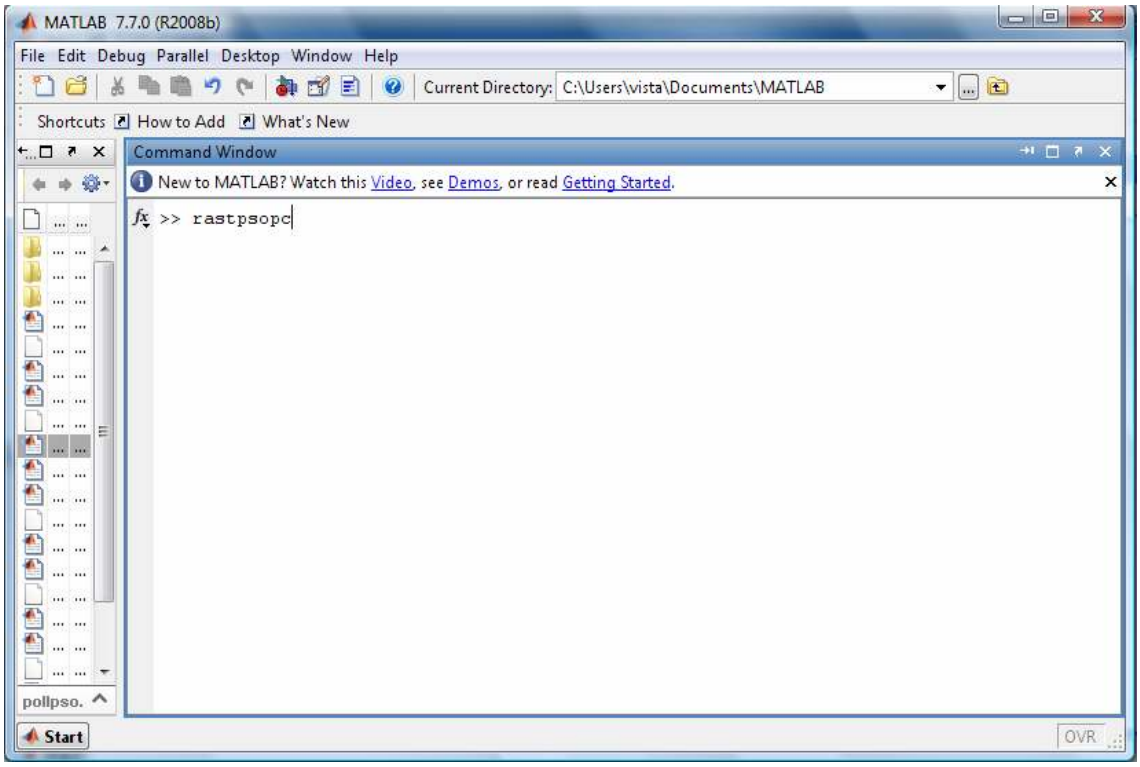
Eğitim Durumu (Kurum ve Yıl)

Lise	SANDIKLI ANADOLU LİSESİ(2000-2003)
Lisans	SELÇUK ÜNİVERSİTESİ ENDÜSTRİ MÜHENDİSLİĞİ BÖLÜMÜ (2003-2007)
Yüksek Lisans	AFYON KOCATEPE ÜNİVERSİTESİ, FEN BİLİMLERİ ENSTİTÜSÜ, MAKİNE MÜHENDİSLİĞİ ANABİLİM DALI (2009-2011)

EKLER

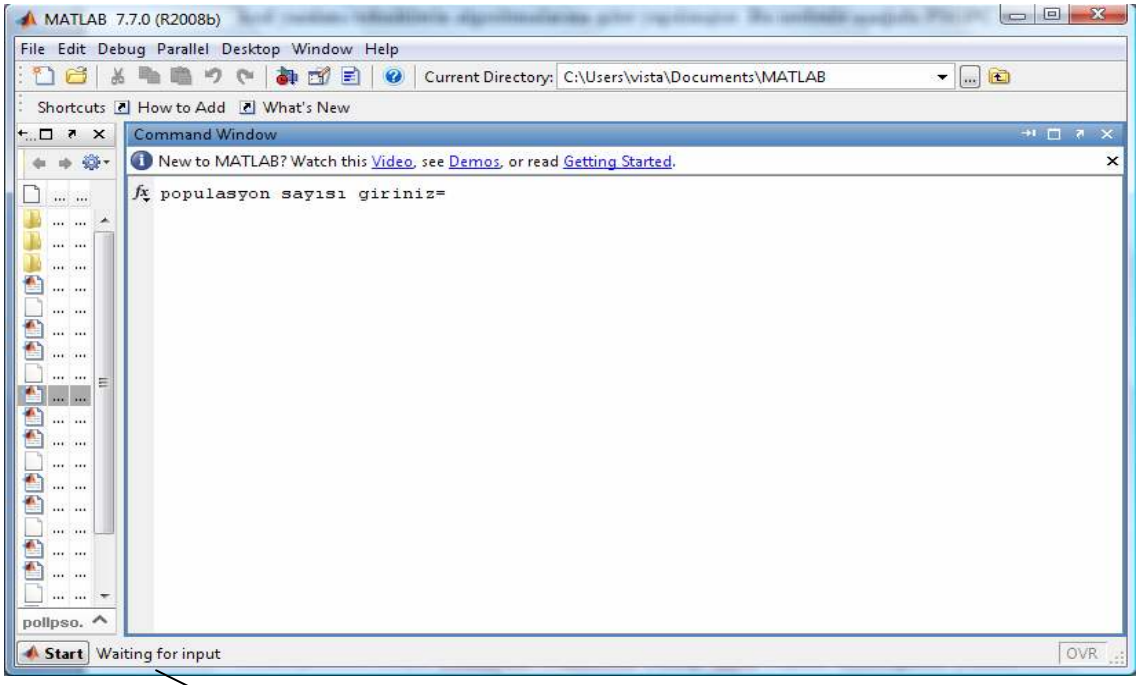
Ek.1 PSOPC Algoritmalarının Arayüz ve Program Çıktısı

Matlab'da PSO ve PSOPC için yazılan kodlamanın çalıştırılmasındaki arayüz ve program çıktıları verilecektir. İki teknik için de arayüz ve çıktıların formatı aynıdır fakat kod yazılımı tekniklerin algoritmalarına göre yapılmıştır. Bu nedenle yalnızca PSOPC ile çalıştırılan Rastrigin Fonksiyonunun çözümü gösterimi incelenecektir.

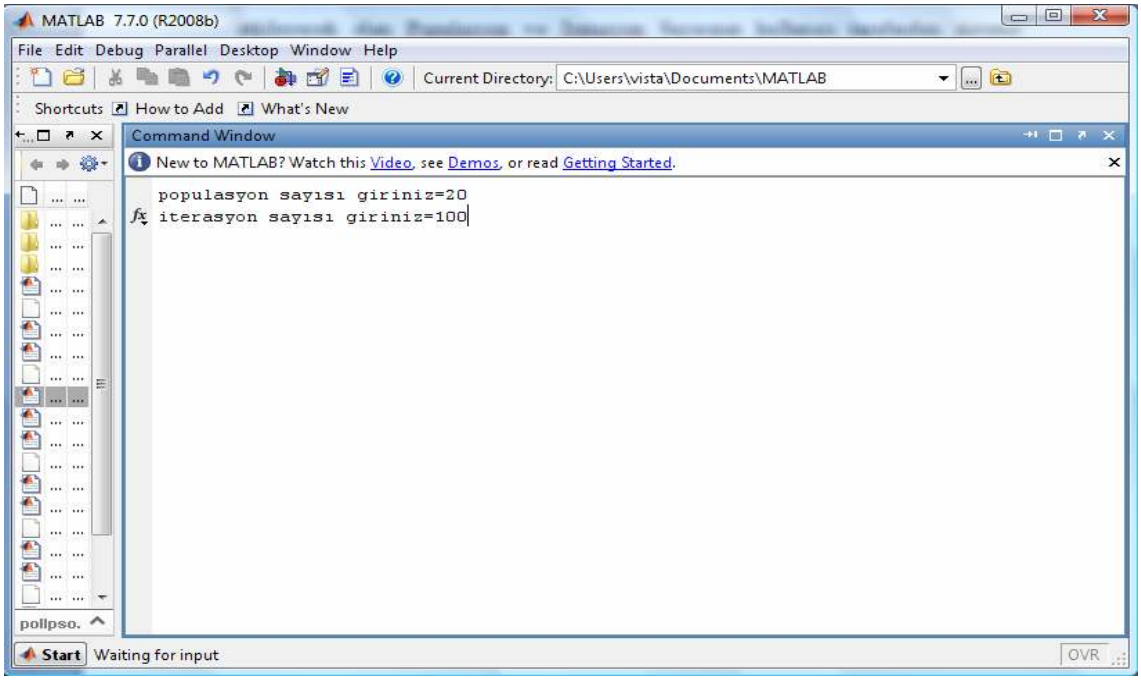


Rastrigin Fonkisyonun çözümü için oluşturulmuş M-File dosyasının adına rastpsopc konmuştur.Bu dosya Command Window'a yazılıp enter tuşuna basıldığında algoritma çalışmaya başlamaktadır.

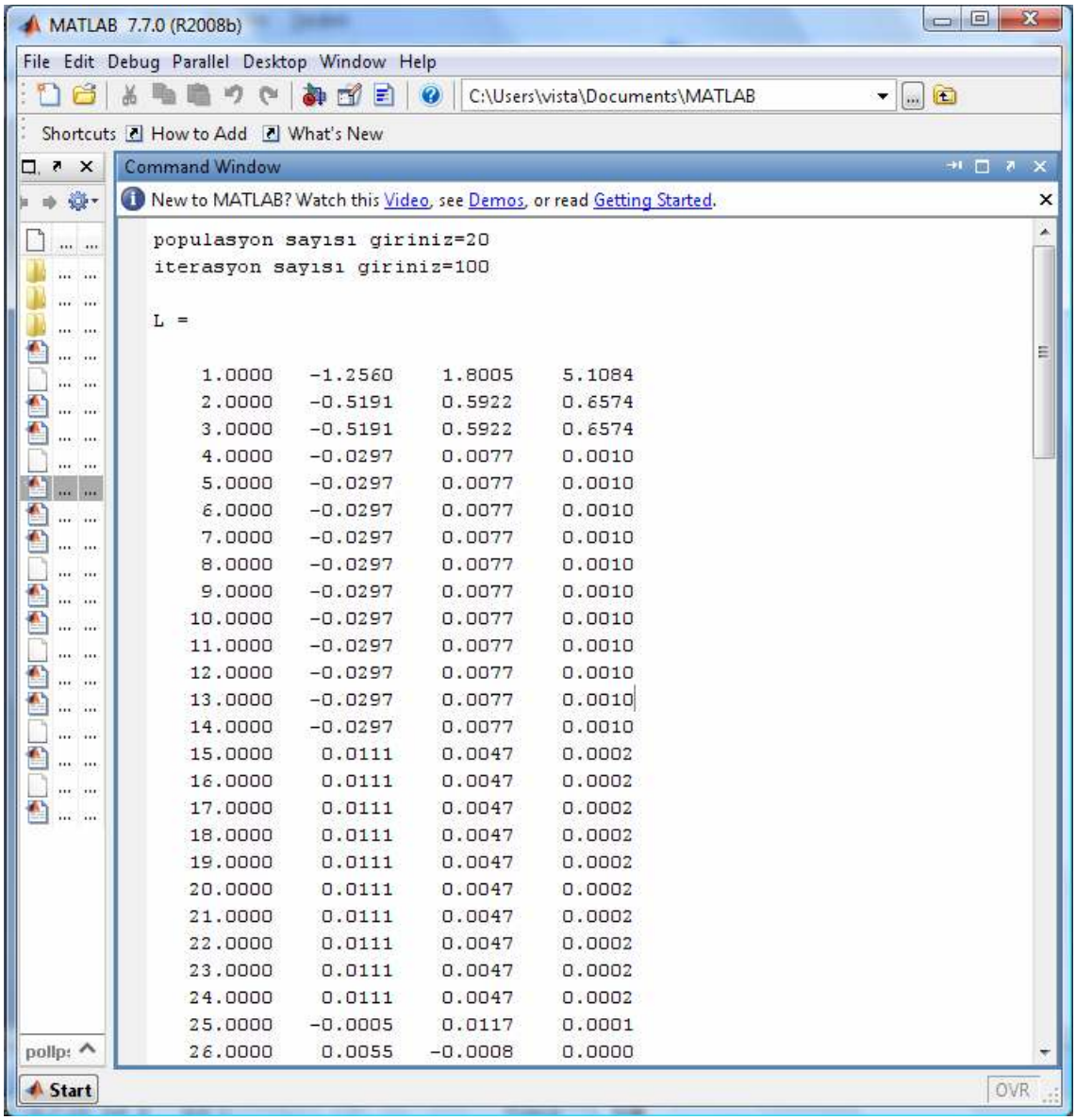
Kodlar yazılırken kullanıcının insiyatif kullanabilmesi istenmiştir. Sonucu direkt olarak etkileyecek olan Populasyon ve İterasyon Sayısının kullanıcı tarafından sisteme girilmesi istenir. Bu fonksiyonun çözümü için 20 populasyon sayısı ve 100 iterasyon sayısı kullanılmıştır.



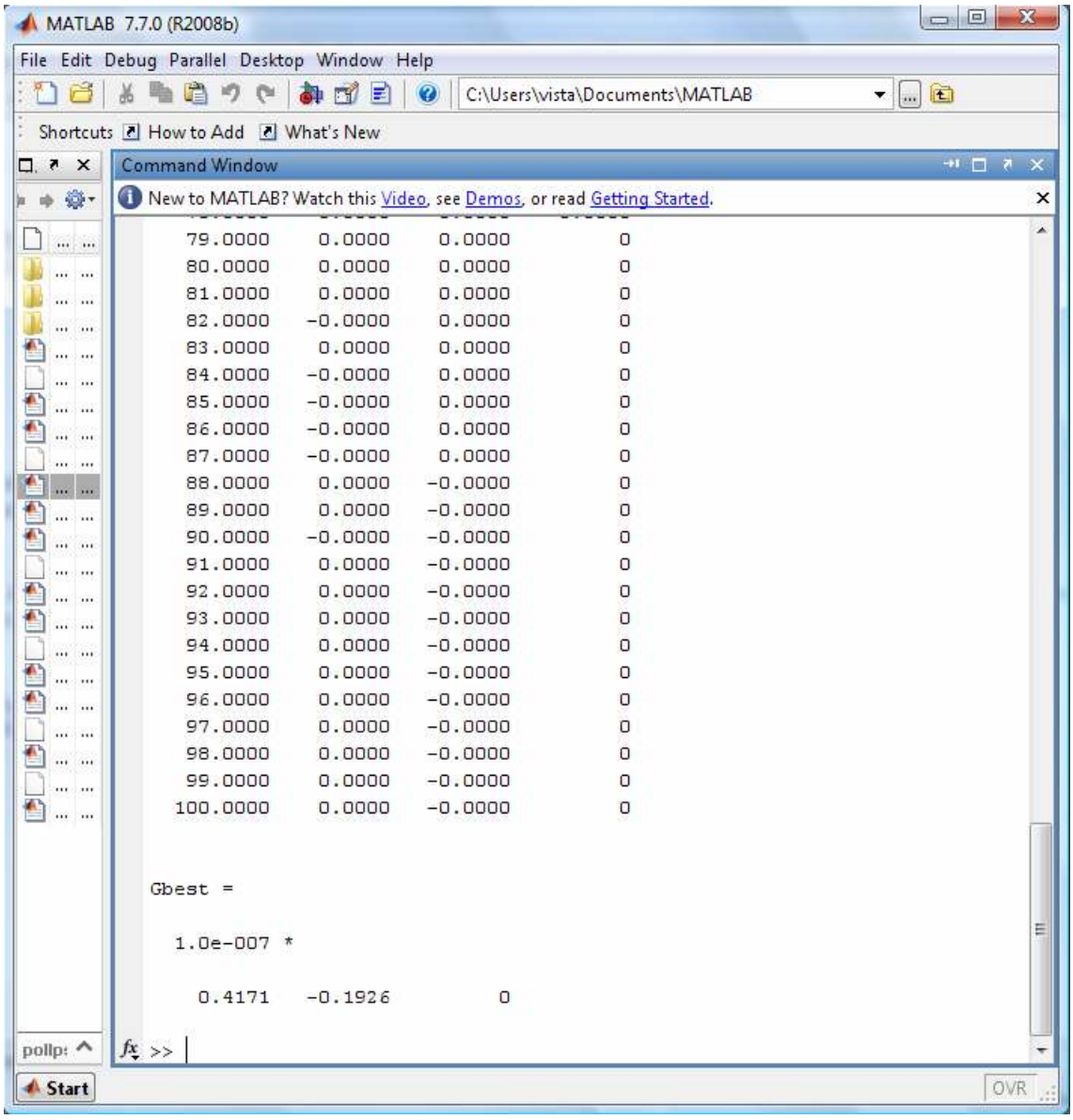
Girdi için bekleniyor



Populasyon sayısı 20 olarak yazıldıktan sonra enter tuşuna tekrar basılır. Sistem bu defa da iterasyon sayısının girilmesini ister. İterasyon sayısının girilmesinden sonra enter'a basıldığında artık program kullanıcının sisteme veri girmesine ihtiyaç duymadan devam eder.

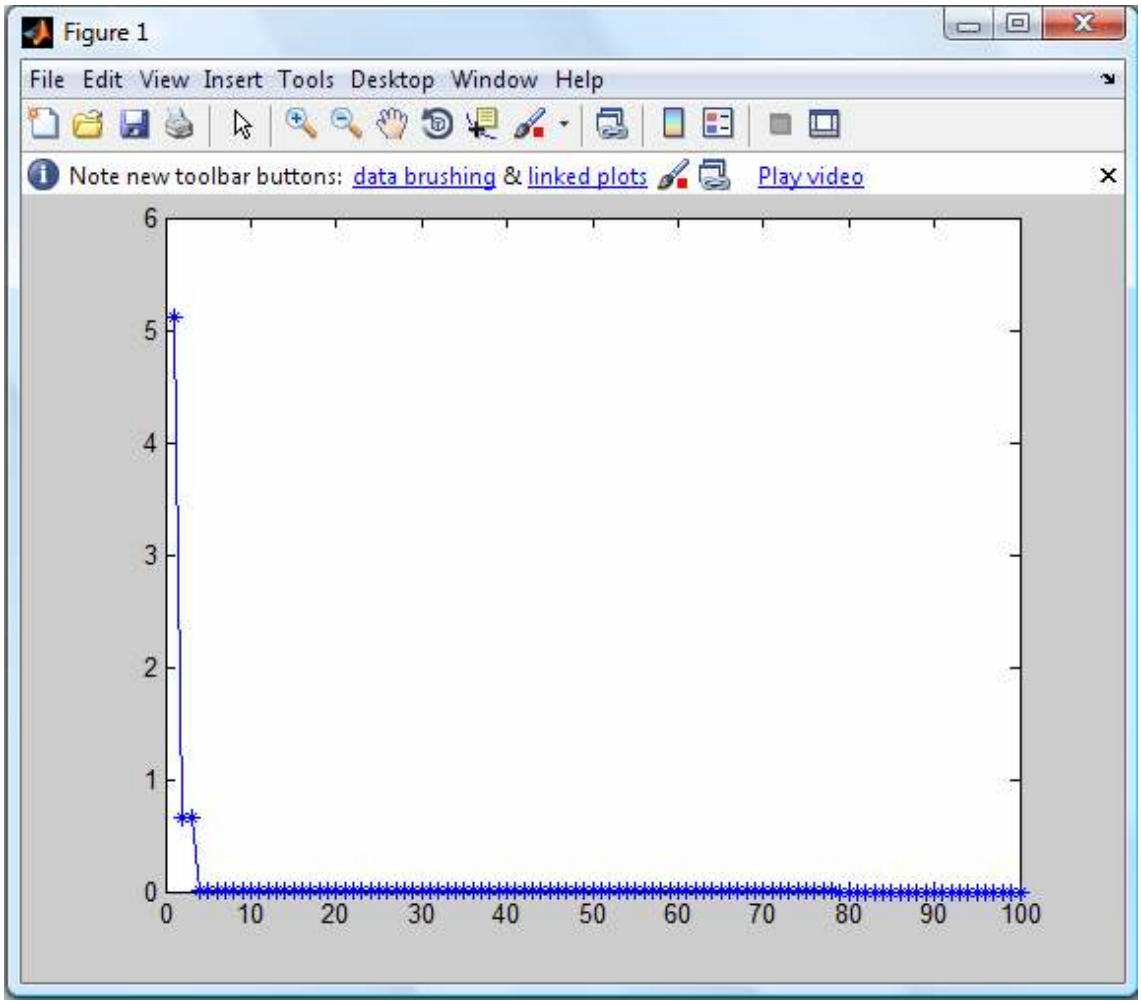


L matrisi olarak adlandırılan matris her iterasyonda x1, x2 ve amaç fonksiyonu değerlerini vermektedir. Görüldüğü gibi 26. iterasyonda virgülden sonra 4 hanede istenen optimum sonucu vermektedir.

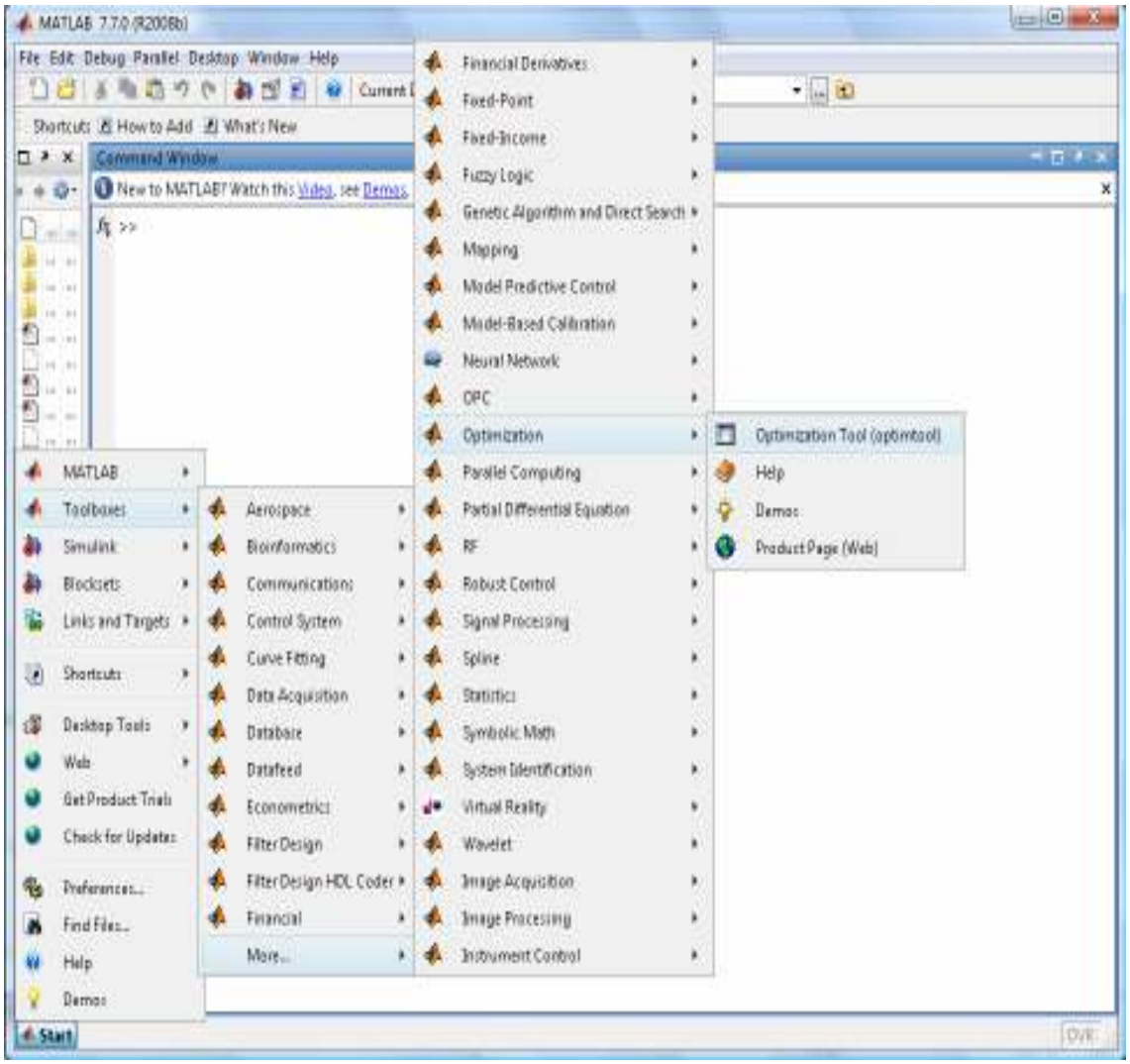


Gbest matrisi 100 iterasyon çalıştırılması sonucu alınan en iyi değeri vermektedir. Optimizasyon çalışmalarında her zaman en iyi sonuca ulaşmak zaman ve maliyet açısından mümkün olmayabilir. Bu tarz durumlarda virgülden sonraki basamak hassasiyetinin ne kadar istendiği iyi tespit edilmelidir.

Ayrıca her iterasyonda x1, x2 ve amaç fonksiyonu değerlerini veren L matrisindeki veriler kullanılarak iterasyon sayısı - amaç fonksiyonu grafiği verilmektedir.



Ek.2 Genetik Algoritmasının Arayüzü ve Program Çıktısı



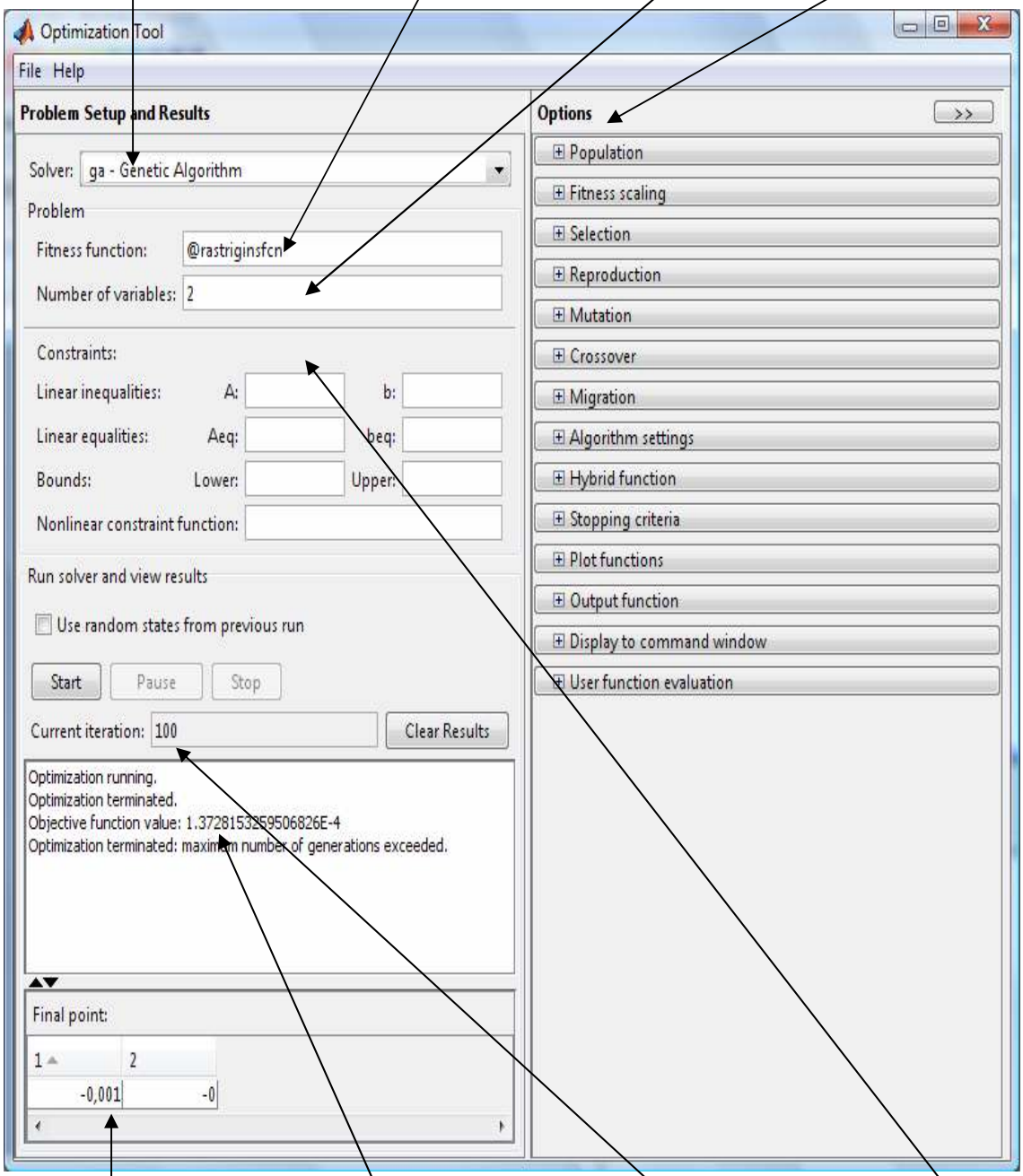
Matlab optimizasyon toolbox'ının kullanımı yukarıdaki resimde verilmiştir. Matlab'ın genel arayüsünden start tuşuna basılır. Optimization'da optimization tool seçilir. Bu seçimden sonra aşağıdaki ekran gelir. Solver'dan Genetik Algoritma seçilir ve GA için oluşturulmuş arayüz kullanıcının karşına gelir.

Kullanılacak olan optimizasyon tekniğinin seçildiği bölüm

Problem için oluşturulmuş M-File dosyasının @ ile adresinin gösterildiği bölüm

Değişken sayısının girildiği bölüm

Çalıştırma kriterlerinin belirlendiği bölüm

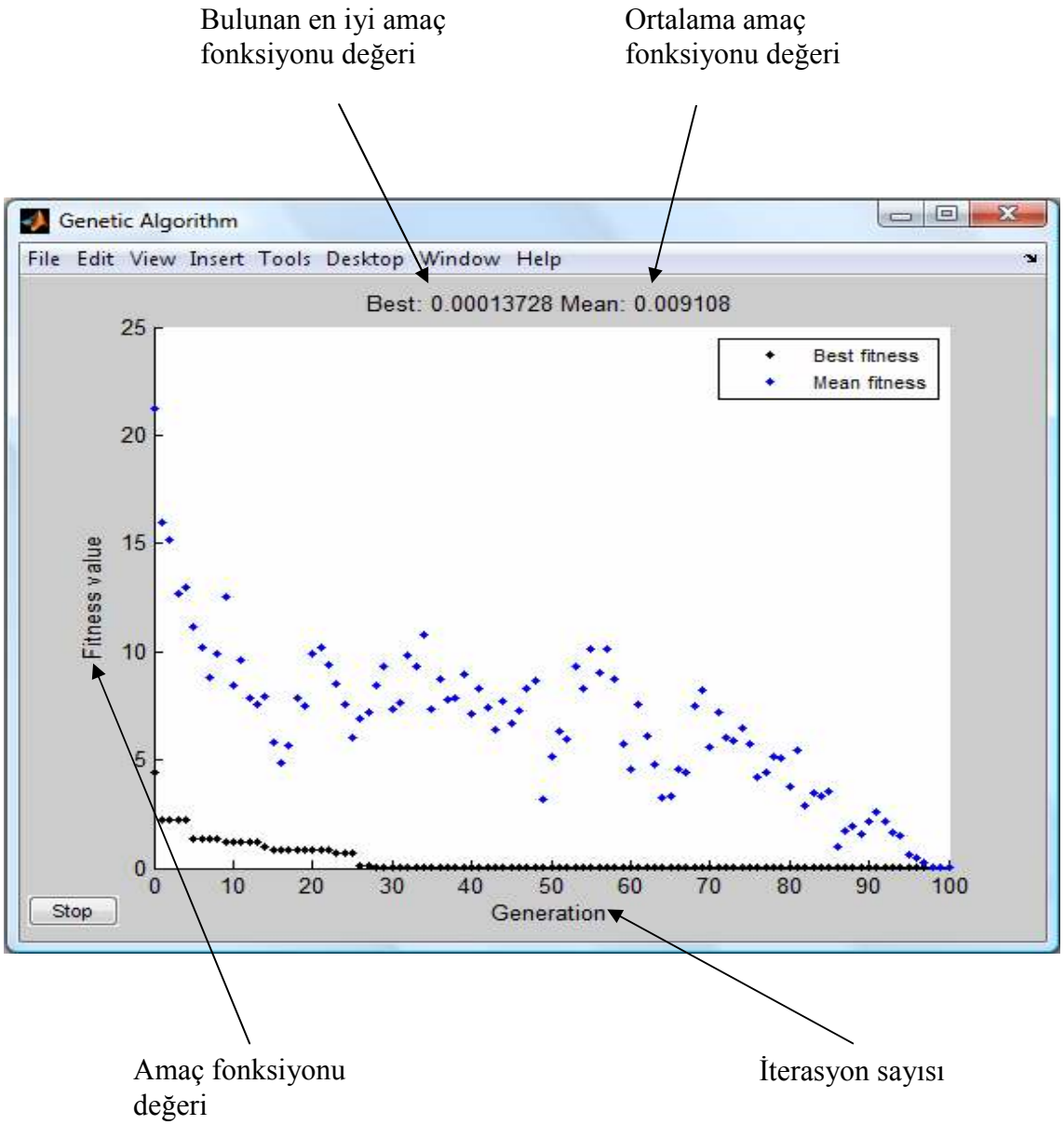


x1 ve x2 değerlerini

Amaç fonksiyonu değeri

En iyi değerin bulunduğu iterasyon sayısı

Kısıt bildirme alanı



Genetik algoritmanın iterasyon sayısı – amaç fonksiyonu grafiği yukarıda verilmiştir.