T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YAPAY SİNİR AĞLARI YARDIMI İLE SİSTEM KONTROLÜ

Beşir DANDIL

YÜKSEK LİSANS TEZİ

ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ
ANABİLİM DALI

1998
ELAZIĞ

T.C.

FIRAT ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

YAPAY SİNİR AĞLARI YARDIMI İLE SİSTEM KONTROLÜ

Beşir DANDIL

YÜKSEK LİSANS TEZİ

ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ

ANABİLİM DALI

Bu Tez, ... Tarihinde, Aşağıda Belirtilen Jüri Tarafından

Oybirliği/Oyçokluğu ile Başarılı/Başarısız

Olarak Değerlendirilmiştir.

(imza) (imza) (imza)

Danışman

Doç.Dr.Mustafa POYRAZ

ÖZET

YÜKSEK LİSANS TEZİ

YAPAY SİNİR AĞLARI YARDIMI İLE SİSTEM KONTROLÜ

Beşir DANDIL

Fırat Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik-Elektronik Mühendisliği
Anabilim Dalı

1998 Sayfa:107

Yapay sinir ağlarının kontrolör olarak kullanıldığı çeşitli kontrol yöntemleri bulunmaktadır ancak bu çalışmada, yapay sinir ağları ile doğrudan ve uyarlamalı kontrol yöntemleri incelenmiştir. Her iki kontrol yöntemi de çeşitli sistemlere uygulanmış, örneksel ve toplu öğrenme algoritmalarını kullanarak Turbo C++ programlama dilinde yazılan programlarla yapay sinir ağının eğitimi gerçekleştirilmiştir. Sistem çıkış hatası; doğrudan yöntemde sistem üzerinden, dolaylı kontrol yönteminde model yapay sinir ağı üzerinden geriye yayılarak kontrolör olarak kullanılan yapay sinir ağı eğitilmiş ve sistemlerin, seçilen referans işaretlerini izleme performansı incelenmiştir.

Elde edilen simülasyon sonuçlarından, kontrolü yapılan sistemlerin, seçilen referans giriş işaretlerini doğru bir şekilde izlediği ve yapay sinir ağlarının, uyarlamalı kontrol alanında başarı ile uygulanabileceği görülmüştür.

ABSTRACT

Master Thesis

PLANT CONTROL BY AID OF ARTİFİCİAL NEURAL NETWORKS

Beşir DANDIL

University of Fırat
İnstitute of Applied Sciences, Fundemental Branch of Electrical-Electronics Engineering

1998 Page:107

There are various adaptive control methods which the neural networks is used as a controller but, in this study, direct and indirect adaptive control methods is implemented by using neural networks. Bath control methods are applied to the selected systems and, the training of the neural networks, using pattern and batch learning algorithms, is implemented by using Turbo C++ programming language. Neura network controller is trained with the system output error which is backpropagated from the system in the direct method but, from the neural network model of the system in the indirect method and, tracking performance of the control system is determined.

Simulation results show that the controlled system output can accurately track the selcted reference input signals and, neural network controller can be succesfully applied to adaptive control.

TEŞEKKÜR

Bu çalışmanın hazırlanmasında yönetici ve yönlendirici olan tez yöneticim sayın Y.Doç.Dr. Mustafa POYRAZ'a, her türlü şartlarda yardımlarını esirgemeyen sayın Öğr.Gör.Muammer GÖKBULUT'a,bu noktaya gelene kadar beni devamlı olarak destekleyen eşime, çalışmalarımda yardımcı olan arkadaşlarım ve hocalarıma, bana her türlü imkanı sağlayan kurum amirim sayın Prof.Dr. Osman ADIGÜZEL'e Teşekkür ederim.

İÇİNDEKİLER

ŞEKİLLER LİSTESİ

SEMBOLLER LİSTESİ

d	Arzu edilen sistem çıkışı
d_u	Ağın ölçülen çıkışı
e	Sistem çıkış hata vektörü
e_c	Eğitim işaretinin sistem üzerinden yansıtılmış hatası
e_m	Model ağ çıkış hata vektörü
E	Eğitim işareti
net	Ağ giriş aktivite vektörü
u	Ağ çıkış vektörü
v	Saklı katman çıkış vektörü
w^1	Giriş katmanı ağrlık vektörü
w^2	Saklı katman ağrlık vektörü
x	Ağ giriş vektörü
y	Sistem çıkış vektörü
y_m	Model ağ çıkış vektörü
α	Öğrenme katsayısı
β	Momentum katsayısı
Δw^1	Giriş katman ağırlıklarını düzeltme vektörü
Δw^2	Saklı katman ağırlıklarını düzeltme vektörü
$\Phi(.)$	Hareketlendirme fonksiyonu
θ_1	Giriş katmanı bias değeri
θ_2	Saklı katman bias değeri

TABLOLAR LİSTESİ

Bölüm 1.GİRİŞ

Biyolojik sinirlerden esinlenerek elde edilen yapay sinir ağları, biyolojik sinirlerle; bulunduğu ortamın değişmesiyle cevaptaki davranışı değiştirebilme, giriş uyarılarındaki küçük değişimlere karşı tolerans gösterebilme, değişik bazı uyarılar karşısında daha önceki uyarılarılar arasından benzer özellikler keşfederek deneyimi olmadığı halde uyarıyı cevaplayabilme gibi bazı benzer davranışlar gösterebilir. Yapay sinir ağlarının bu özellikleri ve çok büyük genişlikteki bilgi yükünü paralel olarak işleyebilme özelliğinden dolayı gerçek sinir sistemine benzer yapay sinir ağları oluşturabilme fikrini desteklemekte ve yapay zeka araştırmacılarına gelecekte yapay zekanın gerçekleştirilebilmesi yolunda ümit vermektedir. (Harvey, 1994),(Karna ve David, 1989)

Yapay sinir ağlarında hızlı gelişime paralel olarak son yıllarda, yapay sinir ağlarının paralel çalışabilmesi ve öğrenme yeteneklerinden dolayı pek çok uygulamada olduğu gibi kontrol alanında da yaygın olarak uygulanma imkanı bulmuştur. Bugün kontrole duyulan ihtiyaç, belirsiz olan karmaşık sistemlerin çoğalması ile artmış ve bu sebeple klasik kontrol yöntemlerinde yeni gelişmelere ihtiyaç duyulmuştur. Böylece daha genel kontrol kavramına yönelinmiş, yüksek seviyeli karar verme, planlama ve öğrenme önem kazanarak "Akıllı Kontrol"e yönelinmiştir. Akıllı kontrol, otomatik kontrol sistemlerinde kullanım alanını genişletmek ve esnekliği arttırmak için günümüzde oldukça sık uygulama alanı bulmuştur. Yaklaşım tekniği olarak, işlemsel araştırma ve dinamik kontrol alanlarında,

hissetme, sonuca ulaşma, planlama ve hareket etmede akıllı bir tarzda olacak şekilde kullanılır.(Brown ve Harris,1994)

Yapay sinir ağlarıyla kontrol sistemlerinin tasarımındaki temel amaç, sistemin karmaşıklığının nasıl idare edildiğini belirten süreçlerin kontrolü için gerekli olan komutlar ve kontrol yapılarının belirtilmesi kadar istenen davranışın gerçekleştirilmesi için gerekli olan modüllerin tespitidir. (Meistel, 1992)

Kontrol sistemlerinde, genellikle değişen dinamikler ve yük şartları, değişen eylemsizlik, parametre ve çevre şartları ile karşılaşılır. Bu nedenle kontrol edilecek sistemin doğru bir matematiksel modelinin çıkarılmasına gereksinim duyan PID ve optimal kontrol yöntemleri, sistemin doğru bir matematiksel modeli çıkarılsa bile doyum, bozucu girişler, parametre değişimleri ve gürültü gibi bilinmeyen şartlardan dolayı etkin bir şekilde kullanılamazlar. (Weerasooriya ve El-Sharkavi, 1991),(El-Sharkavi v.d., 1994)

Sistemlerin modellenmesi ve kontrolünde, ileri beslemeli ve geri beslemeli çok katmanlı yapay sinir ağları yoğun ilgi görmüş ve bu ağların eğitimi için statik ve dinamik geriye yayınım öğrenme algoritmaları geliştirilmiştir. İleri beslemeli yapay sinir ağları doğrusal olmayan bir dinamik işlevi gerçekleştirir ve dinamik geriye yayınım öğrenme algoritmaları ile eğitilirler. Dinamik geriye yayınım algoritmalarının ardışıl ve zamanda geriye yayınım uygulamalarının yanı sıra,geriye yayınım algoritmalarının toplu ve örneksel öğrenme çeşitleri de kontrol sistemlerine uygulanmaktadır. (Akın ve Gökbulut, 1997),(Koçak v.d., 1996),(Karahan v.d., 1997) Örneksel öğrenme algoritmasında her örnek teker teker işlenerek ağırlık değerleri her örnekte ayarlandığından

gerçek zamanlı öğrenme, örneksel öğrenmede ise örnekler toplu olarak değerlendirilerek ağırlık değerleri bir defada ayarlandığından gerçek zamanlı olmayan öğrenme olarak da adlandırılırlar.

Katlı yapay sinir ağlarının, bilgiyi yerel olarak saklayabilen Radyal tabanlı fonksiyon ağları, her hücrenin bir ağırlık değeri üzerinden öz geri besleme yapıldığı ve ait olduğu hücrenin gecikmiş girişlerinin yine bir ağırlık üzerinden uygulandığı bellek hücreli yapay sinir ağları ve gizli katmandaki hücrelerin öz geri beslemeli olarak kullanıldığı Yerel Geri Beslemeli Küresel İleri Beslemeli ağ yapıları da kontrol sistemlerinde kullanılmaktadır. (Habtom ve Litz, 1997), (Brown ve Harris, 1994)

Yapay sinir ağlarının kontrol alanında uygulanabilmesi için bir çok kontrol yapısı geliştirilmiştir. Bu yapılarda yapay sinir ağı tek başına bir kontrol ağı olarak veya bir sabit kontrolörle yada bulanık mantık uygulamaları ile birlikte kullanılabilirler. Bu yapılardan en fazla kullanılanları; yapay sinir ağının sistemin ters bir modelini gerçekleyecek şekilde oluşturularak doğrudan sistem girişine uygulandığı doğrudan kontrol yapısı, yapay sinir ağının hem sistemin düz modeli hem de ters modelini gerçekleyecek şekilde oluşturularak, düz modelden elde edilen yapay sinir ağının sistemin kendisi gibi, ters modelden elde edilen yapay sinir ağının sistemin kontrolörü gibi davrandığı dolaylı kontrol yapısı, yapay sinir ağının sabit bir kontrolörle birlikte kullanıldığı sabit denetleyicili kontrol yapıları ve bulanık mantık uygulaması ve yapay sinir ağının birlikte kullanıldığı Neural-Fuzzy kontrol yapılarıdır. (Kraft ve Campagna, 1990),(Sbarbaro-Hofer v.d., 1993),(Ho v.d., 1997),(Brown ve Harris, 1994),(Chen ve Chang, 1996)

Yapılan bu tez çalışmasında, yapay sinir ağlarının kontrolör olarak kullanılması incelenerek, yapay sinir ağlarının öğrenme genelleme yeteneklerinin belirlenmesi için bazı uygulamalar yapılmıştır. Bu amaçla;

2. bölümde yapay sinir ağları hakkında genel bir bilgi oluşturulması maksadıyla yapay sinir ağları ve uygulamada sık olarak kullanılan ağ çeşitleri hakkında genel bilgiler verilmiştir.

3. bölümde yapay sinir ağlarının,sistem modellemesinde ve kontrol sistemleri uygulamalarında sık olarak kullanılan kontrol yöntemlerinin yapısı ve işleyişi hakkında genel bilgiler verilmiştir.

4. bölümde ise, ele alınan örnek sistemin yapay sinir ağları ile kontrolü için, kontrol ve model ağlarının oluşturularak eğitimi için gerekli aşamalar anlatılarak kontrol edilmiş sistem çıkışları ve yapay sinir ağlarının öğrenme ve genelleme yeteneklerin belirlenebilmesi için yapılan çalışmalar ve elde edilen sonuçlar verilerek değerlendirilmiştir.

BÖLÜM 2.YAPAY SİNİR AĞLARI

2.1 Giriş

İnsan davranışlarını taklit etmek amacıyla geliştirilen yapay zeka uygulamalarının bir alt dalı olan yapay sinir ağları, ortaya çıkışından günümüze değin bir çok aşamalardan geçmiş ve son yıllardaki teknolojik gelişime paralel olarak gelişmini çok hızlı bir şekilde devam ettirmektedir.

Bu bölümde, yapay sinir ağları hakkında genel bilgiler verilerek uygulamada çok sık kullanılan ağ çeşitleri hakkında bilgiler verilmiş ve bazı avantajlarla beraber uygulamada çıkan sorunlara da değinilmiştir.

2.2 Tarihsel Gelişme

İnsanlar tarih boyunca insan beyninin düşünme ve öğrenme yeteneğini kavrayabilmek için büyük gayret göstermelerine karşın ortaya çıkan zorluklar nedeniyle istenilen sonuçları elde edememişlerdir. Ama anatomi uzmanları ve nörobiyologlar beynin ve sinir sisteminin fonksiyonunun ve yapısının çıkarılması konusunda önemli gelişmeler kaydetmiştir. Bu gelişmeler sonucunda insan zekasına yakın davranışta yapay zeka elde edilmesi fikri

ortaya çıkmış ve bu konuda, günümüzde birçok alanda çalışmalar hızla devam etmektedir.

Yapay sinir ağları, anatomi, psikoloji ve nörobiyoloji alanlarındaki araştırmalarla paralel gelişmektedir. 1943 yılında McCulloh ve Pitts'in yayınladıkları makale yapay sinir ağları konusundaki ilk makale sayılmaktadır. Bu makalede sinirlerin giriş faaliyet seviyesi,belirli bir eşik değerine eriştiğinde aktif hale geçen açık/kapalı anahtar gibi davrandıklarını ve mantık fonksiyonlarını hesaplamak için çeşitli şekillerde bir araya getirilebileceklerini göstermiştir. 1949'da nöropsikolog Hebb yayınladığı makalede beynin öğrenme yeteneğini bir modelle açıkladı. Daha sonra 1954'te M.Minsky yapay sinir ağları konusunda doktora yaptı. 1960'lı yıllarda yapay sinir ağlarına ilgi büyük ölçüde artmış ve bir grup araştırmacı yapay sinir ağlarını üretmek için mevcut teorileri geliştirmeye başlamıştır. Bunlardan F.Rosenblat, McCulloch ve Pitts'in eşik değeri yaklaşımını kullanarak tecrübeye dayanan öğrenebilen eleman ağlarını meydana getirmiştir. Yine bu yıllarda S.Groddberg, verbal öğrenmede deney sonuçlarıyla ifade edilen ağların diferansiyel denklemlerini kurmuş;daha sonra gerçek biyolojik sinir ağlarını ifade eden genel denklem setini çıkarmış ve yaptığı araştırmaların sonunda "Adaptif Rezonans Teorisi"ni geliştirmiştir. Aynı yıllarda B.Widrow ve M.Hoff 'ta adaptif anahtar devreleri için kullanılmak üzere kendi öğrenme kurallarını gerçekleştirmişlerdir.1969 yılında M.Minsky ve S.Papert yapay sinir ağlarındaki sınırlamaları inceledikleri "perceptrons" adlı kitabı yayınlamışlardır. Bu kitapta tek katmanlı ağların ayrıcalıklı veya işlemini gerçeklemek gibi basit sorunları çözemeyecekleri ispatlanmıştır. Çok katmanlı ağlarında bazı problemleri

çözmelerine rağmen uygun öğrenme algoritmalarının bulunmadığından ve mevcut yazılım ve donanım imkanlarındaki sınırlamalardan dolayı yapay sinir ağlarına olan ilgi uzun bir müddet büyük ölçüde azalmıştır . Ayrıca sinirlerin modellenmesi, sinirlerin biyolojik yapılarının iyi bilinmesini gerektirdiğinden dolayı da ,önce tıp,biyoloji ve ilgili bilim dalarındaki gelişmelerinde beklenildiği söylenebilir. Bu yıllardaki yapay sinir ağlarına olan ilginin azalmasına rağmen konuyla ilgilerini kesmeyerek çalışmalarını devam ettirmişlerdir (Özmeteler,1989),(İmrak, 1996).

Yapay sinir ağlarındaki büyük gelişmeler 1980'li yıllarda gerçekleşmiştir. 1980'lerin başında J.Hopfield, gelişmeyle birlikte rasgele bulunan etkenlerin probleme karıştığını ve her çözüm yolunda problemin başlangıç halinde bulunduğuna işaret etmiş, bu nedenle her hesaplamanın doğru çözüm yolu için sürekli yenilene bir mekanizmaya ihtiyaç olduğunu belirterek bu iş için geliştirdiği stratejiyi yapay sinir ağlarında kullanmıştır. 1986 yılında D.Rumelhart, G.Hinton ve R.Williams geriye yayınım öğrenme kuralıyla çok tabakalı ağ sistemlerinin bir çok problemi çözmek için eğitilebileceğini göstermiştir. Son yıllarda bilgisayar teknolojisindeki büyük gelişmelerden dolayı, yapay sinir ağlarının yazılım ve donanım uygulamaları hızla devam etmektedir.

Tüm bu gelişmelere rağmen yapay sinir ağlarının sadece bağlantılar arasındaki ağırlık matrislerinin değiştirilebilmesi ve yapının sabit kalması yapay sinir ağlarının yeteneklerini önemli ölçüde sınırlamaktadır. Yapay sinir ağlarının dinamik ve öğrenme özelliklerinin matematiksel tanımı ve modellenmesi konusundaki ilerlemeler bu konudaki ümit verici gelişmeler olarak gösterilebilir.

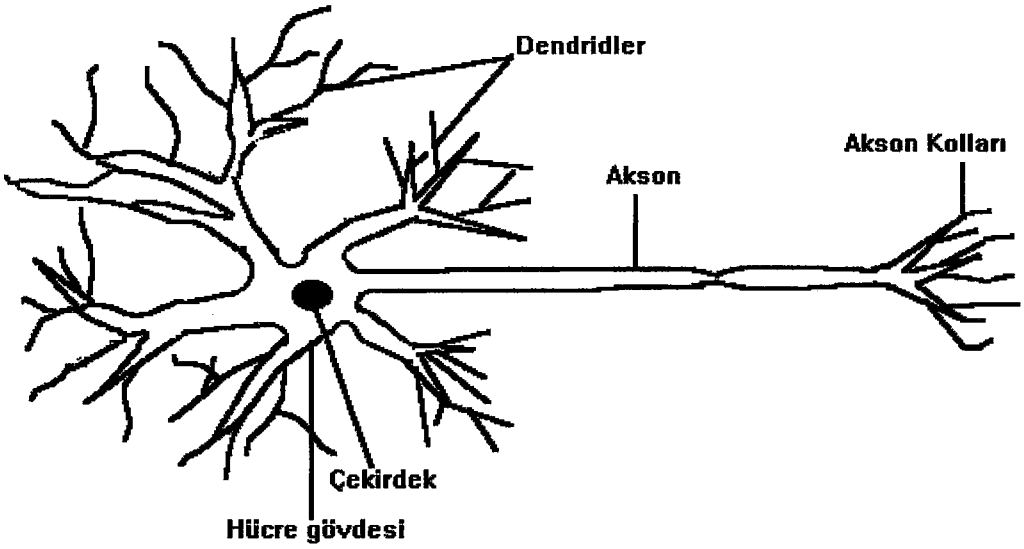
2.3 Sinir Ağlarının Biyolojik Yapısı

Canlıları dış ortamdaki değişiklilere karşı kendi iç ortamını belirli sınırlar içerisinde koruyan sinir sistemi, sinir hücrelerinden meydana gelmiş bir yapıdır. Sinirler vücuttaki diğer hücrelerle birçok ortak özelliklere sahiptir, fakat ilave olarak elektro kimyasal işaretleri algılama, işleme ve iletme özelliklerine sahiptir. Canlı bir sinir sisteminin temelini oluşturan bir sinir hücresi şekil 2.1'de görülmektedir.

Bir sinir hücresi, bir hücre gövdesi ile dendrid ve akson adı verilen uzantılardan meydana gelir. Sinir hücresi, dendrid adı verilen bir kısım uzantılarla diğer sinir hücrelerinden aldığı işaretleri hücre gövdesine taşır ve hücre gövdesinde toplanan bu işaretler değerlendirilerek bir çıkış işareti üretilir ve bu işaretler akson adı verilen uzantılar vasıtasıyla diğer sinir hücresine gönderilir. Bir sinir hücresinde birçok dendrid olmasına karşın tek bir akson vardır. Yani bir sinir hücresinde birçok giriş bulunmasına rağmen tek bir çıkış vardır.

Sinir hücrelerinin en büyük özelliği ise, sinir hücresinin çevreden gelen işaretlere ya cevap vermesi ya da vermemesidir. Eğer sinir hücresine gelen işaret , eşik değerini aşarsa hücre tarafından kabul edilir ve cevaplandırılır, eşik değerini aşamıyorsa yok kabul edilerek hiçbir cevap üretilmez.

İki sinir hücresi arasında birinin dendridi ile diğerinin aksonu arasında bir bağlantı vardır ve bu bağlantı yerine sinaps adı verilir. Bu bağlantı yerinde sinirler birbirine bağlı değildir. İki hücre arasında küçük bir aralık bulunmamasına rağmen iki sinir hücresi darbeleri

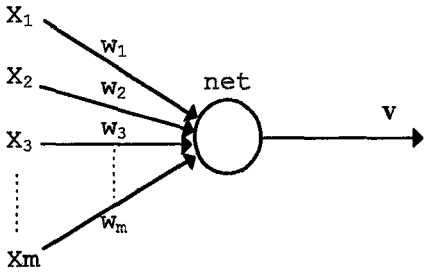


Şekil 2.1 Biyolojik Sinir Ağı

iletmek için birbirine yeterince yakındır. Birinci sinir hücresinin aksonu, sinir hücresinden aldığı işaret sonucunda şişer ve bu şişlik ikinci sinir hücresinin dendridinde bulunan algılayıcılar tarafından kaydedilerek hücre zarının Na^+ ve K^+ iyonları geçirebilme yeteneğini değiştirme yönünde etkileyen kimyasal reaksiyon başlar. Böylece hücre zarı iyonların zarı geçmesine izin verir ve zar potansiyelini geçici olarak değiştirir (Harvey,1994).

2.4 Yapay Sinir Ağlarının Yapısı

Ağlarda kullanılan yapay sinirler, biyolojik sinirlerin benzer karakteristiklerini uygularlar. Sinir



Şekil-2.2 Lineer bir yapay sinir ağı hücresi

sisteminde bir sinir var durumunda iken çıkışta elektriksel darbeler yayar,yok durumunda ise darbelerin yayılması durur. İki sinir hücresi arsındaki bilgi alış verişi bir sinir hücresinin dendridi ile diğer sinir hücresinin aksonu arasındaki sinapslar vasıtasıyla gerçekleştirilir. Sinapsın görevi işareti alan sinire gelen işareti ya durdurucu ya da uyarıcı olacak şekilde bir katsayı ile çarpmaktır.Bu halde, sinapslar sinir girişlerine, diğer tüm sinirlerden gelen işaretleri ağırlık katsayıları ile çarparlar. Sinir hücresinin aldığı ağırlıklı toplam, sinir hücresinin etkinlik seviyesini değiştirir. Gerçek sinirlerin işleyişi sırasında meydana gelen iletişim süresi, yapay sinirlerde yoktur. Şekil 2.2'de bir biyolojik siniri modelleyen, cebrik ve lineer birimli bir yapay sinir hücresi görülmektedir.

x_1 , x_2 , x_3 ...x_m giriş işaretleri, yapay sinire uygulanmaktadır. Her giriş işareti , w_1, w_2 , w_2 , w_3 ...w_m ilgili ağırlık değerleriyle çarpılırken basit sinir modelinde, v çıkış ifadesi aşağıdaki şekilde bir yaklaşıklık yapılır.

$$net=\sum_{i=0}^{m} x_i w_i - \theta \qquad (2.1)$$

$$v=\Phi(net) \qquad (2.2)$$

Burada, girişler **x**, ağırlıklar w vektörüyle toplu olarak gösterilir. Giriş işaretleri biyolojik sinirlerin sinapsislerndeki işaretlere karşı düşen giriş hattındaki elektriksel etkinliği, ağırlık vektörüyse biyolojik sinirlerdeki sinaptik gücü ifade etmektedir.θ, ise eşik değer olarak adlandırılır. $\Phi(.)$ fonksiyonu, bir lineer veya lineer olmayan olmayan fonksiyondur. Şekil 2.3'te $\Phi(.)$ fonksiyonu için kabul görmüş gerçek biyolojik sinirlerin transfer fonksiyonuna yakın lineer olmayan fonksiyonlar görülmektedir. Bu şekilde v çıkış etkinliği VAR ve YOK olarak iki durum arasında sınırlandırılmış olmaktadır. Bu iki durum arasındaki durum değiştirmenin ayrıntıları $\Phi(.)$ fonksiyonunun ayrıntılarıyla belirlenir. Yani v sinir çıkış etkinliği, sinire girişlerden gelen toplam etkinliğe, bir $\Phi(.)$ hareketlendirme fonksiyonu üzerinden bağlıdır.

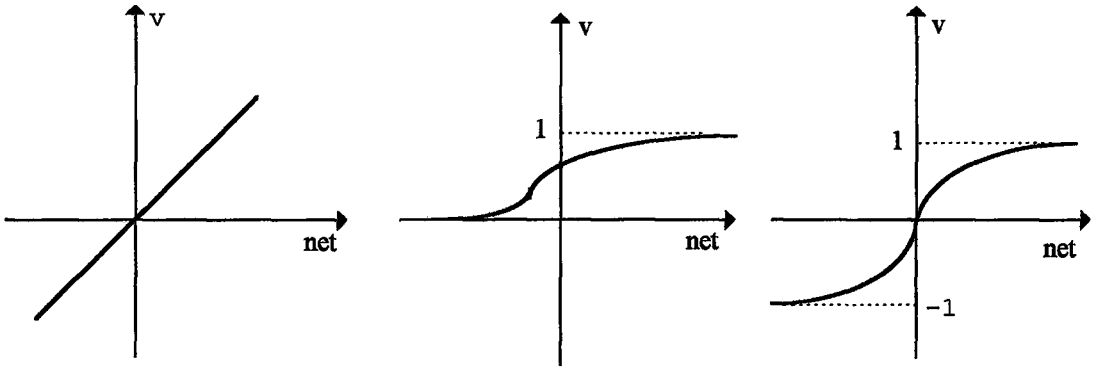
Şekil 2.3(a)'da görüldüğü gibi $\Phi(.)$fonksiyonu lineerdir. 2.3(b) ve 2.3(c)'de ise $\Phi(.)$ fonksiyonu, gerçek biyolojik sinirlerin transfer fonksiyonuna yakın doğrusal olmayan tanh() ve sigmoid fonksiyonlarıdır.

$$\Phi(.)=net \qquad (2.3)$$

$$\Phi(.)=\frac{1}{1+e^{-net}} \qquad (2.4)$$

$$\Phi(.)=tanh() \qquad (2.5)$$

$$\Phi(net)=v \qquad (2.6)$$



Şekil 2.3 Yapay sinir ağlarında sık olarak kullanılan hareketlendirme fonksiyonları

2.5 Yapay Sinir Ağlarında Öğrenme

İnsan beynini modelleyerek elde edilen yapay sinir ağlarının eğitimi, yine insan beyninin eğitiminde olduğu gibi örneklerle eğitilirler. Beyin, birbiriyle bağlantılı binlerce sinir hücresinden meydana gelmiştir ve her bir sinir, diğer sinirlerden bilgi alır ve gönderir. Bu bağlantılardan bazıları güçlü bazıları ise zayıftır. Beyin girişleri alır ve onlara cevap üretir. Bu cevap verme şekli, kısmen beynin genetik olarak programlanmış yapısına göre, fakat daha çok öğrenme ile, yani girişe cevap olarak kendini düzenlemesi ile yapılır. Mühendislikte kullanılan yapay sinir ağları kaba bir şekilde biyoloji üzerine kurulmuştur.

Yapay sinir ağları, programlama yerine örneklerle eğitilirler. Eğitici, sinir ağlarına tanınacak cisimlerin nicel tanımlarını veya söz konusu cisimleri benzer cisimlerden ayırmak için lojik kriter kümeleri sağlamak

zorunda değildir. Bunun yerine sinir ağlarına bazen tanımları ile beraber cisim örnekleri girilir. Ağ, ağırlık matrisindeki değerleri değiştirerek bunları öğrenir ve bir cisim yeniden görüldüğünde uygun cevabı üretir.

Sinir ağlarında, ağırlıklar sabit değildir. Sinir ağlarının öğrenme kuralı, ağ çıktılarının değerlerine cevap olarak üretilen sinirler arasındaki ağırlıkların hepsini veya bazılarını değiştiren bir denklemden ibarettir.

Yapay sinir ağlarının eğitiminde, önemli gelişmeler 1980'li yıllardan sonra olmuştur. Bu yıllarda bazı öğrenme yöntemleri geliştirilmiştir. Bu yöntemlerdeki temel prensip, her tabakanın bağlantı ağılıklarını, yanlışa katkılarına göre bölüştürerek değiştirmektir. Birçok eğitim dönemi sonunda, elde edilen ağırlık grubu doğru sonucu veren bir ağ oluşturmaktır (Hinton, 1992).

Bu eğitim yöntemleri, kuramsal olarak, yerel minimumlar olarak adı verilen yanlış, ancak yokluklarında herhangi küçük bir değişikliğin ağın yanlışlıklarını arttıracağı ağın ağırlık grubuna takılıp kalabilir. Ama gerçekte bir ağ, hemen hemen her zaman en iyi çözümleri öğrenir. Ayrıca kimi öğrenme yöntemleri biyolojik olarak daha akla yatkın olsa da, elde edilen sonuçların kullanılan yönteme bağlılığı yoktur (Hinton v.d., 1993).

Yapay sinir ağları, denetimli ve denetimsiz olmak üzere iki şekilde eğitilirler. Her ikisinde de, ağ bir seri deneme üzerinden çalışır. Denetimli öğrenmede, ağa hem giriş hem de istenen çıkış bilgisi girilir. Her denemeden sonra, ağ kendi çıkışını doğru cevaplar ile karşılaştırır ve çıkış hatası kabul edilebilecek seviyeye ininceye kadar ağırlıklarını değiştirerek iterasyon yapar. Denetimsiz öğrenmede ise hiçbir hedef vektörü yoktur. Giriş vektörü sisteme uygulanır, ve sitem bu giriş vektörü sisteme

uygulandığında uyumlu bir çıkış üretecek şekilde kendisini organize eder.

Bazı uzmanlar, denetimli öğrenmenin biyolojik olarak geçerli olmayacağını iddia etmişlerdir. Çünkü bir biyolojik sistem içinde direk eğitim için veya cevapları istenen çıkış ile karşılaştırmak için bir eğitici yoktur. Bu eleştirilere rağmen denetimli öğrenme birçok probleme uygulanabilmekte ve tatmin edici sonuçlar elde edilmektedir.

2.6 Yapay Sinir Ağlarının Çeşitleri

Araştırmacılar birçok farklı yapay sinir ağ modelleri yapmışlar ve bunları eğitmek için birçok algoritma geliştirmişlerdir. Bu ağ modellerinin birçoğu biyolojik sinirlerden esinlenerek veya en azından biyolojik olarak geçerli olarak kabul edilmiştir. Bu modellerin her biri belli uygulama alanlarında başarılı olmuşlardır. Kullanımı yaygın olan ve diğer modeller temel oluşturan sinir ağ modelleri tablo 2.1'de gösterilmiştir ve bu ağlar hakkında genel bilgiler verilmiştir.

2.6.1 Hopfield ağı

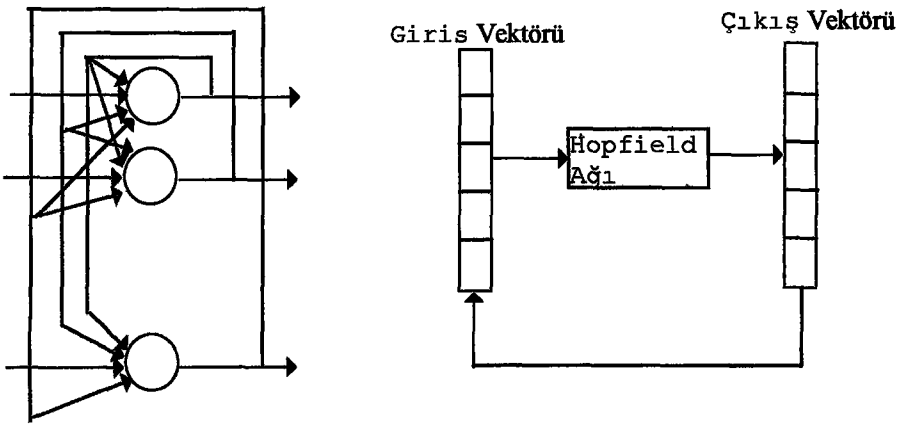
İlk yapay sinir ağ modellerinden biridir ve tipik tekrarlı algoritmadır. Bütün düğüm noktaları birbirine

Ağ Modeli	Çeşidi	Kulanım alanı
Hopfield	Tekrarlı	Optimizasyon
Kohonen	Kendinden Organize	Veri Kodlama
Geçici Farklar	Tahmini	Tahmin
Geriye Yayınım	İleri Beslemeli	Sınıflandırma

Tablo 2.1 Kullanımı yaygın olan yapay sinir ağ modelleri

bağlıdır. Normal olarak ikili girişler kullanır. Bu ağın en kullanışlı olduğu durumlar, girişin tam olarak ikili tabanda gösteriliminin mümkün olduğu durumlardır. Bu ağ, normalde girişlerin sürekli değişkenler olduğu durumlara pek uygulanmamaktadır. Çünkü bu durumda analog büyüklükleri doğru bir şekilde ikili değerlere çevirebilme problemi ortaya çıkmaktadır.

Hopfield kendi ağının değişik düzenlemeleri üzerinde yaptığı çalışmalarla sinir ağlarına olan ilgiyi tazelemiştir. Şekil 2.3'te görülen hopfield ağı, bir içerik adreslenebilir belek olarak veya optimizasyon problemlerinin çözümünde kullanılabilir.

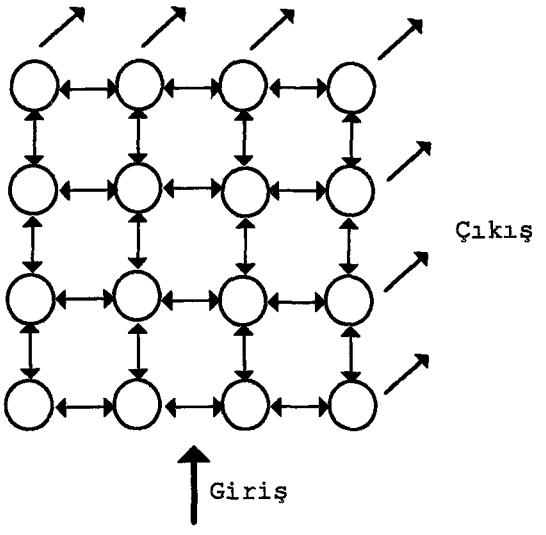


Şekil 2.4 Hopfield ağı

2.6.2 Kohonen ağı

Beyinde alt seviye düzenlemesinin büyük bir kısmı genetik olarak belirlenmiş olmasına rağmen daha büyük seviyedeki düzenlemelerin bir kısmının kendi kendini organize eden algoritmalar tarafından öğrenme esnasında oluşturduğu düşünülmektedir. Bu düşünceden hareketle Kohonen tarafından kendi kendini organize edebilen, denetlenmemiş öğrenme algoritması kullanan, zor olmasına karşın oldukça güçlü ve hızlı olan bir ağdır. Kendinden organize bu ağın temel farkı, düğümlerin bulundukları katman içerisinde de bağlantıda olmalarıdır. Kohonen ağının yapısı şekil 2.5'te görülmektedir.

Şekilden de görüldüğü gibi sinirler birbirine tam olarak bağlıdır. Her bir sinir, çıktısını diğer sinirlere giriş olarak gönderir. Bağlantı kuvvetlerinden birinin sıfır olması mümkündür. İki türlü ağırlık grubu mevcuttur.



Şekil 2.5 Kendinden organize Kohonen ağı

Birinci grup, ağırlıklandırılmış girdilerin toplam aktivasyonunun hesabında, ikinci grup ise sinirler arasında karşılıklı etkileşimi denetlemek amacıyla kullanılırlar. Girdi modellerinin ağırlıkları sinirler arası ağırlıklar sabitleştirilerek adapte edilirler. Her bir girdi sinirlere sunulur ve her bir sinir aktivasyonunu hesaplar. Daha sonra girdi modelleri kaldırılır ve sinirler en büyük çıktılı sinir ile karşılıklı etkilendirilir ve sadece sinir çıktıları göz önünde bulundurulur.

2.6.3 Geçici farklar

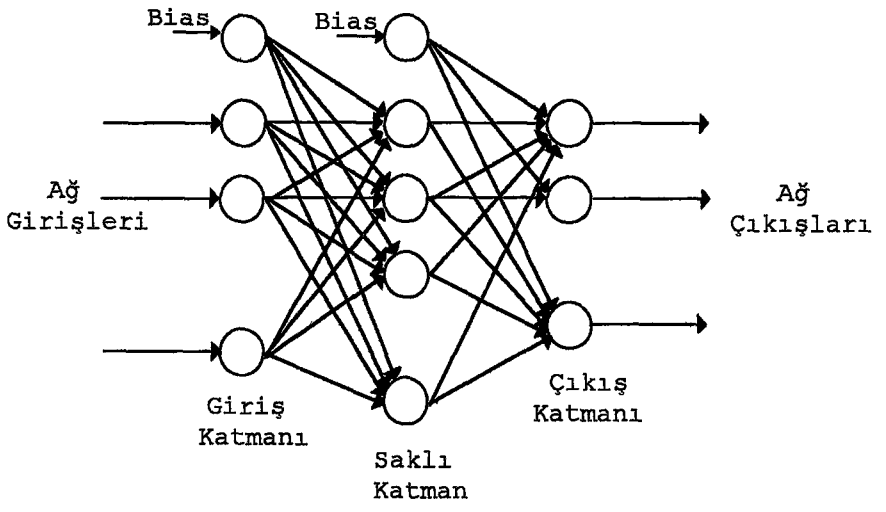
Geçici farklar, zamana bağlı işlemlerin modellenmesinde kullanılan, az sayıdaki yapay sinir ağı

modellerinden biridir. Geriye yayınımdan temel farkı, bir hedefe yönelmiş fonksiyonları öğrenebilmesidir Her hareketin sonucu birkaç adım geçmeden bilinmeyen ağ kullanılarak eğitilir. Bu metodun ilk kullanımı sağa sola hareket eden bir arabadaki sarkaç kontrolüdür. Denetimli öğrenme metodu kullanılır.

2.6.4 Geriye yayınım ağı

Yapay sinir ağlarının en tanınmışı olan geriye yayınım ağı, çok katmanlı ileri beslemeli bir ağdır. Widrow-Hoff öğrenme kuralının bir uzantısı olarak Rumelhart, Hinton ve Williams tarafından 1986 yılında gerçekleştirilmiştir (Rumelhart v.d., 1986). Katmanlardaki sinirler arasında ve bir katmandan önceki katmana geriye doğru bir bağlantı yoktur. Şekil 2.6'da üç katmanlı bir geriye yayınım ağı görülmektedir.

Denetimli öğrenme kullanarak eğitilen geriye yayınım ağında, girişe bir giriş vektörü uygulanır ve çıkıştan bir çıkış vektörü elde edilir. Sonra bu çıkış vektörü istenilen bir çıkış vektörü ile karşılaştırılarak iki vektör arasındaki farkın karesi en küçük olacak şekilde ağ ağırlıklarının ayarlanması sağlanır. Amaç her giriş vektörü için, ağın üreteceği cevabın istenilen çıkış vektörünün aynısı veya yeterince yakın değerlerde olmasını sağlayacak şekilde ağırlık vektörleri oluşturmaktır. Yakın zamana kadar geriye yayınım ağları, eğitim algoritmalarının biyolojik sistemlere uygun olmamasına rağmen son zamanlarda



Şekil 2.6 İleri Beslemeli ağ yapısı

geliştirilen öğrenme algoritmalarıyla yaygın bir şekilde kullanımı sağlanmıştır.

Geriye yayınım ağının eğitimi çok adımlı bir işlemdir. j. düğümündeki giriş ağı net_j, v_i çıkışlarının bir lineer fonksiyonudur ve w_{ij} ağırlıklarıyla birlikte,

$$net_j = \sum_i x_i w_{ij} \qquad i \neq j \tag{2.7}$$

Şeklinde ifade edilir. Giriş ağının bir lineer olmayan fonksiyon olan v_j çıkış düğümü,

$$v_J = \frac{1}{1+e^{-net_J}} \tag{2.8}$$

ile ifade edilir.

Giriş değerlerinin bir lineer fonksiyon ile birleştirilmesi, öğrenmeyi kolaylaştırmaktadır. İteratif bir algoritma olan geriye yayınım öğrenme algoritmasında,

ağ çıkışıyla istenilen çıkış arasındaki hatanın karesini en küçük değere çekmek için aşağıdaki adımlar uygulanır.

1-Başlangıçta tüm ağırlıklara küçük rasgele değerler atanır. Bu ağın simetrisini bozar ve ağırlıkların büyük değerlerle dolmuş hale gelmesini engeller.

2-Giriş tabakasına $x_1, x_2 \ldots x_m$ giriş vektörü uygulanır ve istenilen ağ çıkışları $d_1, d_2 \ldots d_p$ belirtilir. Her denemede girişler yenilenmeli veya ağırlıklar uygun hale gelene kadar eğitim setinden örnekler çevrilerek sunulmalıdır.

3-$y_1, y_2 \ldots y_p$ ağ çıkışları 2.2 ve 2.3 ifadeleri ile, ileriye doğru katmanlardan geçirilerek elde edilir ve tek tek sinirlerin çıkışları hesaplanır. Giriş katmanı sinirlerinin çıkışları, giriş vektörü elemanları ile eşitlenir.(Hareketlendirme fonksiyonu olarak sigmoid fonksiyonu kullanıldığı kabul edilmiştir.)

4-Tekrarlayan bir algoritma kullanılarak ağ ağırlıkları ayarlanır. Bunun için çıkış tabakasından,ağda işlem yapılan her birimden geriye doğru yansıtılır.

$$w_{iJ}(t+1) = w_{iJ}(t) + \alpha v_j E_J \qquad (2.9)$$

burada çıkış katman düğümü için,

$$E_J = -v_j(1-v_j)(y_J - d_J) \qquad (2.10)$$

ve orta katman düğümü için,

$$E_J = -v_J(1-v_J)\sum_k E_k w_{Jk} \qquad (2.11)$$

değerini alır. Öğrenme oranı olarak tanımlanan α, ağırlık değerlerinin salınımını önlemek için kullanılır ve normalde birden küçük bir değerdir.

5-Adım 2'den 4'e kadar olan kısım tekrarlanır.

Geriye yayınım oldukça yavaştır. Karmaşık problemlerde ağırlıkların kararlı hale gelmesi bazen günler ya da

haftalar sürmektedir. Hızı arttırarak zamanı azaltmak için pek çok metot önerilmiştir. Bunlarda biri ağırlık ayarlama ifadelerine momentum teriminin ilave edilmesidir. Bu durumda 2.4 ifadesi,

$$w_{ij}(t+1) = w_{ij}(t) + \alpha y_i E_j + \beta \Delta w_{ij}(t+1) \qquad (2.12)$$

haline gelir ve ifadede yer alan

$$\Delta w_{ij}(t+1) = w_{ij}(t+1) - w_{ij}(t) \qquad (2.13)$$

elde edilir ve burada β momentum sabitidir. Bu metot kullanıldığında ağ, hata alanının bulunduğu dar bandın alt kısmını takip etme eğiliminde olup, eski halinde ise bant arasında salınım yapar.

Hızı artırarak yakınsamayı sağlamanın diğer bir yolu da, her sinire öğrenebilen bias ilave etmektir. Bu özellik her bir sinire +1 ilave ağırlık ekleyerek kolayca sağlanabilir. Bias ağırlığı da diğer ağırlıklar gibi eğitilebilir niteliktedir.

Şekil 2.7' de, yukarıda belirtilen aşamalardan sonra oluşan tipik bir sinirin yeni modeli görülmektedir. Şekilden de görüldüğü gibi ağın öğrenmesi aşağıdaki işlemlerin uygulanması ile gerçekleşmektedir.

1-Saklı katmanındaki sinirlerin aktivasyonunu hesaplama.

2-Çıkış katmanı sinir aktivasyonunu hesaplama.

3-Çıkış katmanındaki hatayı hesaplama.

4-Saklı katmandaki hatayı hesaplama.

5-Sinapsların ikinci saklı katman ile çıkış katmanı arasındaki ağırlıkları hesaplama.

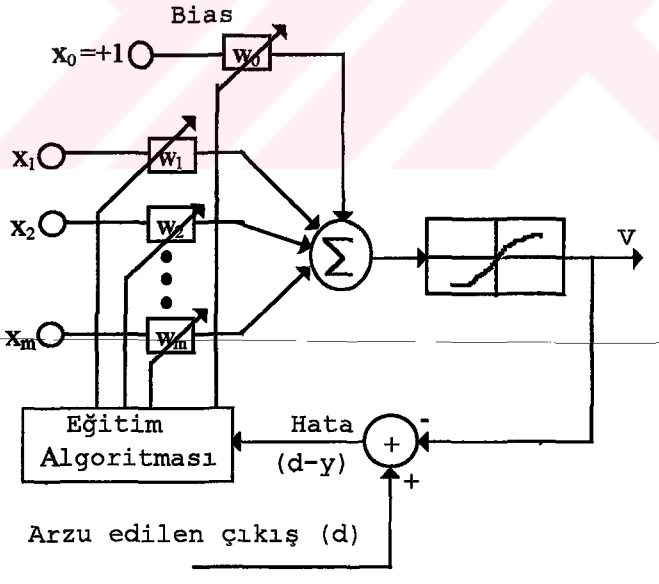
6-İlk katman sinapslarının ağırlıklarını hesaplama.

Yukarıdaki işlemler, çıkış katmanı hatası belirlenen bir değere ulaşıncaya kadar tekrar edilmektedir.

Katmanların boyutu hakkında genelleme yaparak bir değer vermek mümkün değildir.

Giriş ve çıkış katmanlarının boyutu uygulamaya (giriş ve çıkış faktörlerinin adedine) göre tespit edilir. Ancak saklı katmanın boyutu belirsiz olmakla birlikte genellikle giriş katmanının daha büyük olmalıdır.

Çok katmanlı geriye yayınım ağ tasarımı yapılırken karşılaşılan bir diğer sorun da saklı katmanın her birinin kaç tane sinirden oluşacağına karar verme işlemidir. Eğer saklı katmandaki hücre sayısı az seçilirsea, giriş değerleri arasından gerekli özellikleri tanımlamayabilir;gereğinden fazla olması halinde ise, genelleme ve soyutlama hatasına düşebilir. Ortaya çıkan bu sorunun en iyi çözümü ortalama bir değer almaktır.



Şekil 2.7 Sinir hücresinin eğitimi

2.7 Yapay Sinir Ağlarının Özellikleri ve Avantajları

2.7.1 Yapay sinir ağlarının özellikleri

Yapay sinir ağları biyolojik sinirlerden esinlenerek, benzer şekilde basit elamanlardan oluşturulmuştur. Bu özelliklerden bazıları aşağıda belirtilmiştir.

1-Adaptasyon:Yapay sinir ağlarının en önemli özelliklerinden biridir. Bulunduğu ortamın değişmesiyle cevaptaki davranışı değiştirebilme yeteneğidir.

2-Genelleme:Diğer önemli özellik genelleme yeteneğidir.Bu özellik sistemin, giriş uyarılarındaki küçük değişmelere karşı tolerans göstermesini sağlar.

Yapısından ve ağların insan zekası kavramını esas alan mekanizma kullanmasından dolayı, sinir ağları otomatik olarak genelleme yapar.

2.7.2 Yapay sinir ağlarının avantajları

Sinir ağları sahip oldukları avantajlardan dolayı mevcut klasik tek işlemli çözümlerle çalışmanın zor olduğu pek çok konuda uygulanmaktadır. Aşağıda bu avantajlardan bazıları verilmiştir (Blum, 1992).

1-Öncelikle ilgili faktörleri belirlemeye daha az ihtiyaç duyarlar. Genelde ilk yapılacak iş ilişkili faktörlerin istatiksel modelini çıkarmaktır. Sinir ağları

veri sunmada ve ilgili verileri saptamada üstünlüğe sahiptir.

2-Sinir ağları genelde yüzlerce faktöre sahiptir; fakat bütün giriş faktörlerinin kabul edilen etkisi, modelin herhangi bir istatistik modelden daha çok kesin olarak zor problemlere cevap bulmasını sağlar.

3-İstatistik modeller, dolaylı yoldan ilişkileri öğrenme yoluna gider. Halbuki sinir ağları yaklaşımıyla, problemi doğrudan modellemek mümkündür.

4-Giriş faktörlerinin çok olmasından ve verideki bozucu etkiden veya donanımın hatasından kaynaklanan sorunlar azdır.

5-Sinir ağ modellerinde bulunan her sinapsis kendi işlemine sahiptir. Aynı katmanda bulunanlar arasında zaman bağımlılığı olmayıp,tamamen senkron olarak çalışırlar.

BÖLÜM.3 KONTROL SİSTEMLERİNDE YAPAY SİNİR AĞLARI

3.1 Giriş

Yapay sinir ağları ile sistem tanıma ve kontrolü son yıllarda büyük ilgi görmüş ve bu konularla ilgili bir çok çalışma yapılmıştır. Bu çalışmalar sonucunda, yapay sinir ağlarının kontrol alanında uygulanmasına ilişkin çok sayıda yöntem geliştirilmiştir. Bu bölümde uygulamada ilgi gören bazı kontrol yapıları incelenerek, genel bilgiler verilmesi amacı ile bu yöntemlere kısa bir şekilde değinilmiştir.

3.2.Yapay Sinir Ağlarının Kontrol Uygulamaları

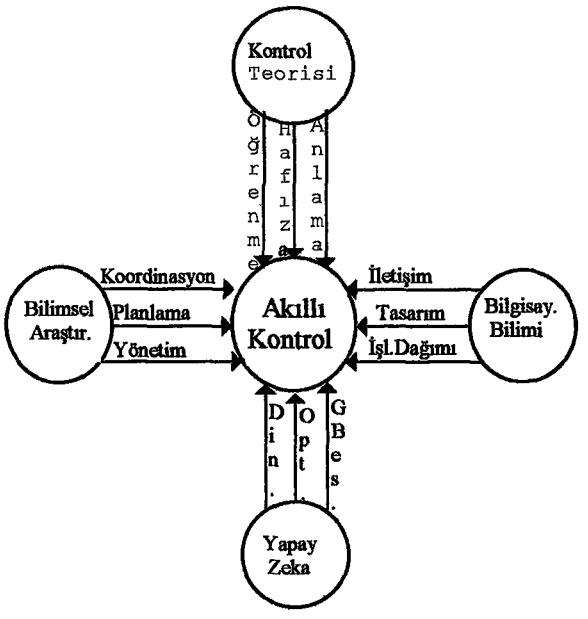
Kontrolün temel amacı, istenilen cevabı verebilecek fiziksel bir işlemciyi elde edebilmektir. Kontrol sistemler teorisinde kontrol edilecek işlemci genelde sistem olarak adlandırılır. Günümüzde kontrole duyulan ihtiyaç, belirsiz olan karmaşık sistemlerin giderek çoğalması ile artmış ve bu sebeple klasik kontrol metotlarında yeni gelişmelere ihtiyaç duyulmuştur. Bu sebeple daha genel kontrol kavramına yönelinmiş, yüksek seviyeli karar verme, planlama ve öğrenme önem kazanmıştır. Yapay sinir ağlarının paralel çalışabilmesi ve öğrenme yeteneklerinden dolayı, pek çok problemi çözmede kullanıldığı gibi son yıllarda kontrol alanında da; işlem kontrolü, robot kontrolü,endüstriyel

üretim, havacılık ve diğer alanlarını içeren kontrol uygulamalarında kullanımı hızla artmaktadır.

Yapay sinir ağları en zor kontrol problemlerini anlama ve çözme yolunda yeni bir yöntem sunmaktadır. Ayrı ayrı çalışan sistemlerin kontrolü, ortam değişimine uyum sağlayacak yüksek dereceli esnekliğe ihtiyaç duymaktadır. Bu değişimler tahmin dışı olabilmekte ve bilinen matematiksel modellerle ifade edilemeyebilir. Bu gibi hallerde, klasik adaptif kontrol teknikleri, kayda değer sonuçlar vermediğinden bu konuda yapay zeka uygulamaları ve özellikle sinir ağları bazı kolaylıklar sunmaktadır (Brown ve Harris, 1994).

İçinde yapısal hatalar bulunan karmaşık dinamik sistemlere uygulanan kontrol, sinir ağlarını içeren akıllı kontrol yöntemidir. Düşük seviyede çalışmayı tanımlama ve kontrol etmede ileri beslemeli sinir ağları, yüksek seviyede ise sistemdeki hatayı bulmak için model tanımlayıcı olarak kullanılırlar.

Kontrol sistemlerinde akıllı kontrol uygulamaları, otomatik kontrol sistemlerinde kulanım alanını genişletmek ve esnekliği arttırmak için, yapay zeka işlemsel araştırma ve dinamik kontrol alanlarında, hissetme, sonuca ulaşma, planlama ve hareket etmede akıllı bir tarzda olacak şekilde Fu(1971) tarafından ortaya konmuştur (Brown ve Harris, 1994). Bu şekilde elde edilecek bir uygulama şekil 3.1'de gösterildiği gibi kontrol yapısını, komutlarını ve bütün sistem karmaşıklığını idare etmek için ortaya konmuş olan ileri kavramlar olarak bilgisayar bilimini içerecek şekilde geliştirilmiştir. Sinirsel kontrol sistemleri özel algoritmalar ile tanımlanmazlar. Esnek ve sağlam bir tarzda komutları işletmek ve çevrelerini hissetme ve sonuca ulaşabilme tekniklerini kullanırlar. Sinirsel kontrol



Şekil3.1 Akıllı Kontrolün İşleyiş Tekniği

sistem tasarımlarının esasında, sistemin karmaşıklığının nasıl idare edildiğini belirten süreçlerin idare edilmesi ve bağlanması için gerekli olan komutlar ve kontrol yapılarının belirtilmesi kadar istenilen davranışın gerçekleştirilmesi için gerekli olan işlemlerin belirlenmesi vardır.

3.3 Kontrol Sistem Yapıları

Herhangi bir sistem yapısında iki önemli bileşen mevcuttur. Bunlar, problemin basit alt başlıklar halinde aşamalı bir şekilde sıralanarak fonksiyonel ayrıştırılması

ve hem yatay hem de dikey olarak komşu olan alt birimlere bilgi yollayan, komut ve kontrol yapılarıdır. Bu yapılar, artan zeka prensibine göre tasarlanırlar. Kontrol aşamalarında en yüksek seviyeler, istenilen bilgilerin anlaşılabilmesi için sonuç çıkaran zeka stratejilerini kullanırlar. Bilgi kontrol yapılarına doğru ilerledikçe, bilgi ve işleme algoritmaları daha az kesin ve daha az akıllı olurlar. Kesin algoritmik ayrışmalar, lokal planlama algoritmaları düşük seviyede koordinasyon ve kontrol teknikleri global sonuç çıkarma rutinlerinden olmalarına rağmen bağımsız problemlerdir.

Kontrol aşamalarında her bir birim, konularını işlemek için sadece yeterli kaynağa (veri ve fonksiyonlara geçiş) sahiptir ve bilgi sistem boyunca dağıtılan bir tarzda depolanır. Kapsamlı sistem kontrol yapısı orijinal olarak Brooks (1986) tarafından önerilmiş ve aşamalı yapıyla bağlantılı olarak geleneksel tarzda his, plan ve eylem devrelerine tercihen tümüyle reaktif davranışı temel alırlar. Birçok basit alt konunun kollektif eylemlerinden ortaya çıkabilen karmaşık ve faydalı davranışı temel alır.

3.4 Kontrol Uygulamalarında Öğrenme Algoritmaları

Öğrenme algoritmalarının, insanın en az seviyede müdahalesiyle zamanla değişen çevrede ve kötü tanımlamalarda çalışması istenir. Bu algoritmalar özellikle, matematiksel analizin mümkün olmadığı bir çok kontrol sisteminde tipik olarak kullanılır.

Birçok farklı algoritma, öğrenen kontrol şemalarında ve uygulamanın tipini genel olarak gösteren bilgi gösterim yapılarında kullanılabilir. Öğrenme algoritmalarının en popüler yapılarından üçü; yapay sinir ağları, bulanık mantık ve uzman sistemlerdir (Brown ve Harris, 1994). Bu algoritmalar, bulanık mantık ve bazı birleşik hafızalı sinir ağlarında ve bazı uzman sistemleri yapay sinir ağları içerisinde kullanılabilmesinden dolayı aralarında güçlü ilişkiler vardır ve birbirinden ayrı değildirler. Öğrenme probleminin temelinde, algoritmanın ara değer hesaplamalarının lokal olarak doğru ve etkili bir şekilde tahmin edilmesi anlamına gelen eğitim örneklerinin sınırlı miktarda algoritmaları genelleştirme kabiliyeti bulunmaktadır. Adaptif olarak yapılandırılan lineer modeller ve kontrolörler, kesinlikle global ve lineer bir ilişkiye sahip olarak düşünülürler. Halbuki önceki bilginin en düşük miktarı istenen fonksiyonun yapısı hakkında olduğu farz edilirse algoritma eğitim örneklerinin birçoğunu genelleştirir ve bilgiyi amaca uygun olarak veri setinden alır.

Bir algoritmanın modelleme kapasitesi, ağlar tarafından yapılan tahminleri üretebilen lineer olmayan fonksiyonların aralıkları tespit eder. Öğrenme kuralı seçilen model yapısını yakınsamanın sınıfını etkilemesine ve kullanılması gereken öğrenme kuralının tipini de tespit edebilmesine rağmen, algoritmanın esas modelleme kapasitesini genel olarak etkilemez.

3.5 Sinirsel Hesaplama

Sinirsel hesaplama (Nöral hesaplama) kalıpları, seri bir makine de gerçekleştirilebilmesi ve bir algoritma içerisinde birleştirilmesine rağmen geleneksel algoritmik hesaplamalardan farklıdır. Yapay sinir ağları problemlere çözümler önerir ve geleneksel algoritmik ayrıştırma teknikleri kullanarak çözülmesi çok zor olan problemlerde kolaylık sağlar. Sinirsel yaklaşımın yararları aşağıdaki şekilde özetlenebilir.

1-Klasik programlamaya tercihen çevreyle etkileşimle öğrenme.

2-Öğrenebilen fonksiyonel ilişkilerin çeşitlerinde kısıtlamaların az oluşu.

3-Genelleştirme özelliği

4-Tabiatında varolan paralel tasarım ve bilgisayara ait yükün birçok basit süreç elemanına dağılabilme özelliği. Bu nedenle ağların işleyici hatalarına ait küçük hata toleranslarına sahip olması

İlk üç özellik herhangi bir öğrenme algoritmasında istenilen özelliklerdir. Dördüncü özellik ise daha geniş gerçek zaman sistemlerine uygulanan bir özelliktir. Bir öğrenme algoritması bu özelliklere sahip ise kontrol sistemi aşağıdaki avantajlara sahip olabilir.

-İnsan müdahalesinin istendiği kadar azaltılması.

-Kontrol sisteminin performansının geliştirilmesi.

-Başlangıçtaki tasarım, zaman ve maliyetteki azalma.

-Kontrol sisteminin esnekliğinin artması.

Yapay sinir ağlarının performansı bu dört özelliğin nasıl yerine getirileceğine bağlıdır. Çevrim içi adaptif

modelleme ve kontrol için algoritmalar aşağıdaki özelliklere sahip olmalıdır.

-Belirli bir bilgiyi kararlı bir şekilde ve gerçek zamanda öğrenme.

-Kararlılık özelliği ve öğrenme yakınsamasının ispatının mümkün olması.

3.6 Kontrol ve Modellemede Sinir Ağları

Son yıllarda adaptif modelleme ve kontrolde, öğrenmeli algoritma ve biyolojik temelli modeller kullanımına büyük bir ilgi oluşmuştur. Akıllı kontrol uygulamaları; zamanla değişen ortamlarda ve kötü tanımlanma durumlarında işleyebilme, çevreden gelen etkiler gibi sistem dinamiğindeki değişikliklere uyarlanabilme, kararlı bir tarzda belirli bilgilerin öğrenilebilmesi, sistem dinamiğindeki bazı kısıtlamaların yerleştirilebilmesi gibi özelliklere sahip algoritmalar ister.

İnsan öğrenimi, bütün bu özelliklerin elemanları somutlaştırmak için ortaya çıkar ve genellikle araştırmacılar bu gibi özellikleri makinelere uyarlamaya gayret etmektedirler.

Farklı bütün kontrol problemleri için araştırma gerçek olmayan bir çabadır. Yapay sinir ağlarında kullanılan öğrenme algoritmaları, insan bilgi işleme sistemlerinde en iyi başlangıç yaklaşımlarıdır. Yapay sinir ağları öğrenme kabiliyetleri ve modellemeleri analiz edebildiğinden insan davranışına uyarlanabilir. Lau ve Widrow, (1990) "Beynin işleyişi hakkında mikroskobik, makroskobik bütün ve kesin

bir görüş için yaklaşık 50 yıl beklenebilir. Mühendisler bu kadar uzun bir süre bekleyemezler." şeklinde bir tez öne sürmüşlerdir. Şu an yapılan araştırmaları iki kategoriye ayırmak mümkündür.

-Beynin fonksiyonelliği hakkında yeni teoriler üretmek

-Gerçek dünyadaki problemlere uygulamak.

Bu iki uygulama alanı arasında, birbirini teşvik de vardır fakat ikinci alan, sinirsel olmayan yani klasik çözümler yerine sinirsel çözümler öne sürmek için sinir ağ teorileri üretilmesi üzerindedir. Geçmişte yapay sinir ağ algoritmaları, uygunluğu düşünülmeksizin ve diğer algoritmaların daha fazla uygun olmasına bakılmaksızın kontrol problemlerine ve modellemelere uygulanmıştır.

Öğrenme algoritmaları, kontrol mühendislerine daha çok önerilebilir. Bu algoritmalardan beklenen sonuç, adaptasyon gerçeklemeleri idealden çok uzak olmalarına rağmen artan çözüm kalitesiyle azalan tasarım ve işlem maliyetiyle sistem performansının gelişmesidir. Genel olarak adaptif algoritmalar, kontrol ediciler ve lineer sistem modellerine uygulanır ve adaptasyon şemalarının esnekliğini tespit eden birçok parametre seçilmelidir. Sinir ağları, sadece yaklaşım olmalarına ve bazı sinirsel öğrenme algoritmaları adaptif kontrol/işaret işleme alanlarında geliştirilen tekniklerle karşılaştırıldığında daha ilkel görünmesine rağmen lineer olmayan sistemlere uygulanabilen algoritmalar olmaları sebebiyle büyük avantaj sağlarlar (Sanner ve Slotine, 1992).

Yapay sinir ağlarının çoğunun bağımsız kontrol edicileri olduğu iddia edilmekle birlikte, sık sık kullanılan modellerin birçoğu sabit bir ağ yapısına sahip olduğundan parametreleri uydurmak için adım adım ilerleyen iniş kuralları kullanıldığından bu yorumların genel olarak

yanlış olduğu otoritelerin görüşüdür. Ağın yapısı lineer olmayan adaptasyondan dolayı çok esnek olabilir, fakat model tabanlıdır ve bu ağlar geleneksel lineer adaptif algoritmalardan ayrılmak için zayıf veya yumuşak modelleme şemaları olarak adlandırılırlar. Bu durumda ne gibi fonksiyonel, gösterimsel ve genelleştirme özelliklerine sahip olduğunun tespit edilmesi için farklı yapay sinir ağlarının modelleme kapasitelerini araştırmak önemlidir.

3.7 Sinirsel Modelleme ve Kontrol Yapısı

Sinirsel modelleme ve kontrol algoritmaları tarif edilmeden önce, temel öğrenme modüllerinin nasıl uygulanabileceğinin anlaşılması yararlı olacaktır. Bu modelleme ve kontrol yapısı genel olarak bağımsız ağlardır. Birçok öğrenme algoritması, bazıları diğerlerinden daha uygun olabilmesine rağmen kullanılabilir.

Birçok sinirsel modelleme ve kontrol algoritmaları sürekli zaman domeninde sistemin durumunu değerlendiren değişkenlerin ölçülerek kullanılmasıyla sürekli zaman domeninde ifade edilir. Sistemin durumundaki (model) değişiklikleri önceden haber vermek veya kontrol işaretinde istenen değişiklikleri hesaplamak için kontrol işareti ve istenen sistem tepkisi ve kontrolör davranışını yerine getirmek için gereklidir. Bununla birlikte kontrol uygulamaları ve sinirsel modellemenin büyük çoğunluğu örneklenmiş sistemler olarak yerine getirilmişlerdir ve tek girişli tek çıkışlı lineer olmayan sistemler için durum denkleminin seçimi;

$$x'(t)=f(x(t),u(t)) \quad (3.1)$$

$$y(t)=g(x(t)) \quad (3.2)$$

Burada x(t) t anındaki sistem durum vektörü, u(t) mevcut kontrol işareti ve y(t) ise gözlenebilir sistem çıkışıdır. Karşılık gelen ayrık zaman durum denklemleri;

$$x(t+1)=f(x(t),u(t)) \quad (3.3)$$

$$y(t)=g(x(t)) \quad (3.4)$$

olarak ifade edilirler.

Sistem modellerinin çoğunluğu çıkış işaretinin bütün sistem durumları hakkında etkili olarak geniş bilgi içerdiği farz edilir ve bu durumda yukarıdaki ayrık zaman durum denklemi tekrar aşağıdaki şekilde düzenlenebilir.

$$Y(t)=h(y(t-1),u(t)) \quad (3.5)$$

Burada y(t-1), çıkışın geçmişinden oluşturulmuş n_y süresinin vektörü $y(t-1),y(t-2)...y(t-n_y)$ ve u(t) ise geçmişten ve geçerli kontrol hareketinden oluşturulmuş n_u süresinin vektörüdür $u(t),u(t-1)...u(t-n_u)$.

Önce ayrık veya sürekli gösterimin hangisinin olacağına karar verilir, sistemin derecesi ($n_y$ve n_u gecikmeleri) tespit edilmelidir. Herhangi bir öğrenmeli kontrol sisteminde çok küçük bir değer seçimi modellenmemiş dinamiğin öğrenmeli kontrol sisteminde kararlılığı etkileyebilmesine rağmen model için genelleştirme zayıf yakınsama oranında tahmin edilen sonuçlar parametreleştirme üzerindedir. Sistemin derecesi tahmin edilemediği zamanlarda da adaptif nöro-fuzzy sistemlerin kullanılabilmesi istenir ve bu belli kontroller için olabilmesine rağmen, nöro-fuzzy haritalar basit lineer olmayan fonksiyonlardır ve kararlılığı garanti eden bilgi giriş vektöründe mevcut değilse kontrol döngüsü kararsız olabilir. Bunlar lineer haritalar olduğunda bu niceliklerin seçimi ile teorinin büyük bölümü geliştirilir.

80'li yıllarda Billings ve öğrencileri Narmax olarak adlandırılan genel bir lineer olmayan modelleme yapıları geliştirmişlerdir. Narmax modeli şöyle tanımlanabilir;

$$y(t)=h(y(t-1)...y(t-n_y,u(t-1)...u(t-n_u))+e(t) \quad (3.6)$$

y,u ve e sırayla sistem çıkışı, girişi ve ek bozulma vektörleridir. Bu çok genel bir ilişkidir ve birçok yapay sinir ağı bu formda olabilir (Billings v.d., 1992).

3.7.1 Modellendirme stratejileri

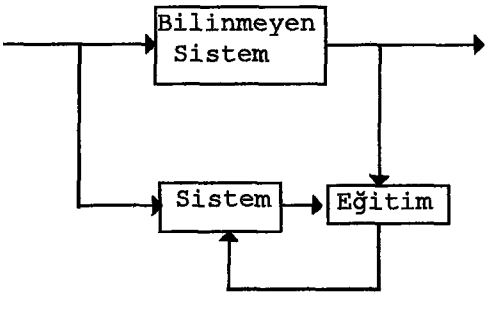
Bir öğrenme modülünde, modellendirme kabiliyetini ortaya çıkarabilen dört temel yapı mevcuttur;temel sistem modeli,ters sistem modeli,uzman ters sistem modeli ve operatör modeli. Bu dört durumdan üçü için, istenen ağ çıkış değeri doğrudan doğruya kullanılabilir ve uzmanlaşmış öğrenme kuralı ağırlık setini eğitmek amacı ile kullanılabilir. Uzmanlaşmış ters sistem modelleri algoritmasındaki hata, sistem çıkışında oluşmakta, bunun yanında ağ çıkışı ise sisteme giriş oluşturmaktadır. Bundan dolayı ters modeli eğitmek için, sistem çıkış hatasını geri beslemeyi sağlayacak bir yöntem gereklidir. Tüm ağ giriş verilerinin sağlanması, ağın yeteneklerinin yakınlaştırılması, ölçüm ve modelleme hata/gürültüleri filtre edecek eğitim kuralının kabiliyeti için, tüm bu şemaların başarısında giriş işaretinin yeterli derecede uyarılmış olması gereklidir.

3.7.1.1 Sistem modellendirme

Birçok değişik sebepten dolayı bir sistem modeline ihtiyaç vardır: Sistem çıkışının bir tahminini gerektiren daha büyük bir geri besleme kontrol çevrimi içerisinde kullanmak, gerçek çıkışın mevcut olmadığı durumlarda (zaman gecikmesi nedeniyle) sistem performansının tahmin edilmesi, hata belirlenmesi ve çevrim dışı kontrolör tasarım stratejisinin de kullanılması için. Temel yapı çok basittir. kontrol işareti ve sistem konumlarının (uygun bir şekilde geciktirilmiş) örnekleri alınır ve bu ağ, giriş ağını oluşturur. Ağ çıkışı daha sonra hesaplanır ve ölçülmüş olan sistem çıkışı ile karşılaştırılarak bu iki değer arsındaki fark ağ çıkış hatasını azaltmak için ağırlık vektörünü ayarlamada kullanılır. İleri besleme sistem modelleme şeması şekil 3.2'de gösterilmiştir.

$$e_y(t)=d(t)-y(t) \qquad (3.7)$$

$e_y(t)$ sistem çıkış hatası, $d(t)$ ölçülmüş olan sistem çıkışı, $y(t)$ t zamanındaki ağ vektörüdür.



Şekil 3.2 İleri Besleme sistem modelleme blok şeması.

Öğrenme kuralları, çıkış hatalarının karelerinin toplamını en aza indirgeyecek şekilde formüle edilir ve eğitim işareti;

$$E = f(e_y^2(t)) \tag{3.8}$$

olarak ifade edilir. Burada beklenti operatörü, $\Phi(.)$ ağ giriş muhtemel hareketlendirme fonksiyonu kullanılarak hesaplanır. Bu genellikle soyut bir muhtemel hareketlendirme fonksiyonu olup, yukarıdaki ifadenin sağ tarafı bir anlık çıkış hatalarının karesinin sınırlı bir toplamı haline gelir.

Çıkış hatalarının karelerinin toplamı, ağın istenen fonksiyonu ne derece yeniden çoğaltılabileceği hakkında bir ipucu verir ancak, eğitim setinin veri noktalarının setini temsil eden bir seti içermesi gerekir.

3.7.1.2 Doğrudan ters sistem modellemesi

Ters sistem modellemeden amaç, tüm kontrolör/sistem yapısının birim transfer fonksiyonuna sahip olacak şekilde bir kontrolörün formüle edilmesidir. Modelleme hatalarının transfer fonksiyonunu birim değerden uzaklaştıracağı kaçınılmazdır, ancak böyle bir ters modelin bir standarda ilaveten ileri besleme ön düzenleyici olarak kullanımında, lineer geri besleme kontrolörünü genellikle geniş bir grup lineer olmayan sistem için iyi performans sağlar. Doğrudan ters sistem modellemesinin blok şeması şekil 3.3'te gösterilmiştir.

Ters modelin iyi bir şekilde tanımlanması için, eğitim örneklerinin mutlaka benzersiz olması gerekir. Sistem ters çevrilebildiğinde bu durum sağlanır veya ters çevrilemeyen sistem eğitim verileri, giriş uzayının sınırlı bir alanında bulunursa, sistem kısmen ters çevrilebilir. Bununla birlikte, jakobyenik fazla değişen sistemler için, bu yaklaşımın kullanılması ve modelleme hatasının sıfıra yaklaşmadığı durumlarda özen gösterilmelidir. Bu durum, eğitim işaretini kullanarak ağın sistem çıkış uzayından ziyade kontrol uzayındaki çıkıştaki hataların karelerinin toplamını asgariye indirdiğinden dolayıdır.

$$E_u=f(e_u^2(t)) \tag{3.9}$$

burada,

$$e_u(t)=d_u(t)-u(t) \tag{3.10}$$

olup,

u(t)=ağın kontrol çıkışı

$d_u(t)$=ağın ölçülen çıkışıdır.

Kontrol işaretindeki hata (tek girişli ve tek çıkışlı bir sistem için) kontrol sistemindeki bir hataya aşağıdaki formülle bağlıdır.

$$e_y(t)\cong \frac{dy(t)}{du(t)}e_u(t) \tag{3.11}$$

Burada dy/du sistemin türevi veya onun jakobyenidir. Böylece 3.8 ve 3.9 ifadeleriyle verilen eğitim işaretleri yaklaşık olarak aşağıdaki formülle birbirine bağıntılıdır.

$$E_y\cong f\left(\left(\frac{dy(t)}{du(t)}\right)^2 e_u^2(t)\right) \tag{3.12}$$

Sistemin lineer olmadığı durumda jakobyenin değeri değişmekte ve kontrol hatalarına farklı ağırlıklar

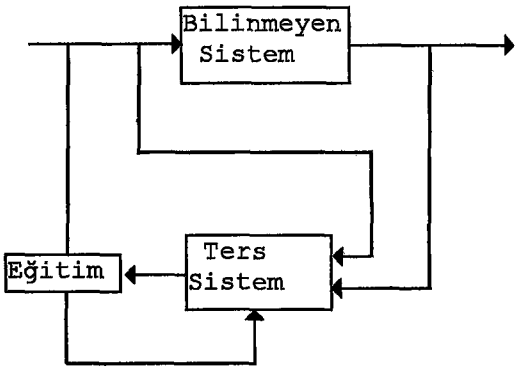
uygulanmaktadır. eğitim işaretleri birimin basitçe lineer olarak değerinin ölçülmüş hali gibi olmadığı bir eşdeğerliliktir ve tasarımcının bir performans ölçümünün minimize edilmesinin, diğerinin de minimize olacağı anlamına gelmediğinin farkında olması gerekir. Şayet çıkış hatalarının sistem jakobyen denklemi ile bağlantısı yoksa bu ifade aşağıdaki şekilde sadeleşir.

$$E_y \cong f\left(\left(\frac{dy(t)}{du(t)}\right)^2 e_u^2(t)\right)E_u \qquad (3.13)$$

ve jakobyenin etkisi öğrenme hızı ile bağdaştırılabilir.

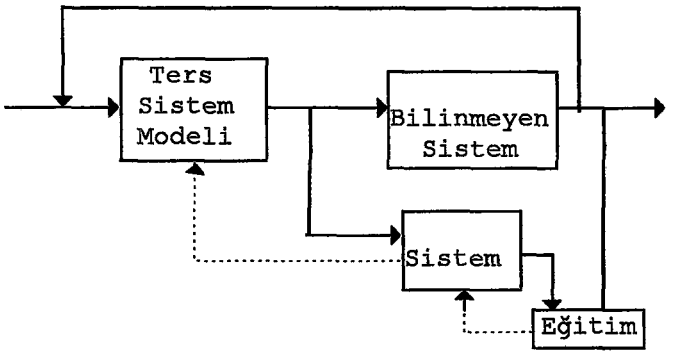
3.7.1.3 Uzmanlaşmış ters sistem modelleme

Bu yaklaşımın amacı birim transfer fonksiyonuna sahip olan bir ters model sistem yapısı sağlamaktır. İlk önce bir ileri sistem modeli oluşturulur. Ters modelin parametrelerini düzenlemek için ileri sistem modelinden geriye doğru geçirilen bir hata işaretini sağlamak için sistem cevabıyla istenen çıkışın arasındaki fark kullanılır (Brown ve Harris, 1994), (Jordan ve Rumelhart, 1991). Bu yaklaşımın daha önceki algoritmaya göre önemli avantajı, bu modelde bir hedef sürücü olmasıdır. Açık çevrim uygulamalarında, sistem çıkış hatası ters modelin giriş uzay bölgelerinde belirlenmemiş bölgelere etki etmesine sebep olur. Bunun yanında doğrudan ters modelleme yaklaşımı ise kontrol işaretinin yeterli derecede uyarıp uyarmadığını öğrenebilir. Bir uzmanlaşmış ters sistem modellemesi blok şeması şekil 3.4'te görülmektedir.



Şekil 3.3 Doğrudan ters sistem modelleme blok şeması.

Öğrenme algoritması, sistem çıkış hatalarının karelerinin toplamını minimize etmeye gayret eder. Bunun yanında doğrudan ters sistem modeli yaklaşımında ise çıkış hatalarının karelerini kontrol çabasını minimize etmektedir. Bu iki değer 3.7 ve 3.8 deki ifadelerle birbirine bağlantılıdır. Şayet sistem lineer değilse gerçek ters sistem ve adaptif model arasında bir uyumsuzluk mevcutsa, en uygun parametre değerleri genellikle farklı olduğundan doğrudan ve uzmanlaşmış ters modelleme yaklaşımları birbiriyle uyum sağlamazlar.Psaltis tarafından belirtildiği gibi doğrudan eğitimin uzmanlaşmış eğitimden önce gerçekleştirilmesinin bazı yararlarının olmasının yanında simülasyonların bu şekilde yapılması görünür bir avantaj sağlamamaktadır. Sistem kazançlarının birim değere yakın olmadığı ve modellemenin uyumsuzluk içinde olduğu en uygun ağırlıklar farklı ve bu iki model algoritmalarının yakınsamasının oldukça farklı oranlarıyla ilgili iki örnek Brown ve Harris (1993) tarafından verilmiştir. Ters modelleme yapıları kontrolörlerin sentezinde kullanılabilse

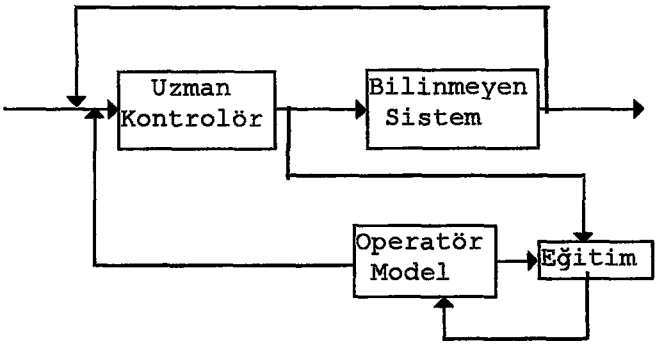


Şekil3.4 Uzmanlaşmış ters sistem modelleme blok şeması.

bile geri besleme bilgi eksikliğinden dolayı alternatif öğrenme kontrolleri kadar kullanışlı değildir. Hunt (1992)Bu durum kısmen açık çevrim ters model uygulamasıyla giderilebilir ancak yukarıdaki yorumların da dikkate alınması gereklidir (Brown ve Harris, 1994).

3.7.1.4 Operatör modelleme

Öğrenen algoritma ile tecrübeli sistem arasında bir paralellik vardır. Bunların cevabı ağı eğitmede kullanılan, istenilen ağ çıkışını oluşturur. Bu eğitim işareti operatörün benzer girişleri için farklı tepkilerinden dolayı büyük miktarda gürültü içerir. Bu sebeple klasik ağ öğrenmeleri kullanılmadan önce bu işaretin filtrelenmesi gereklidir. Bütün modelleme stratejilerinde olduğu gibi eğitim kümesinin ilgili işlem alanından yeterli zengin



Şekil3.5 Operatör Modelleme Blok Şeması

örneklere sahip olmasına dikkat edilmelidir ve operatörde mevcut olan tüm bilgileri ağ giriş vektörü içermelidir. Operatör modellemenin blok şeması şekil 3.5'te görülmektedir.

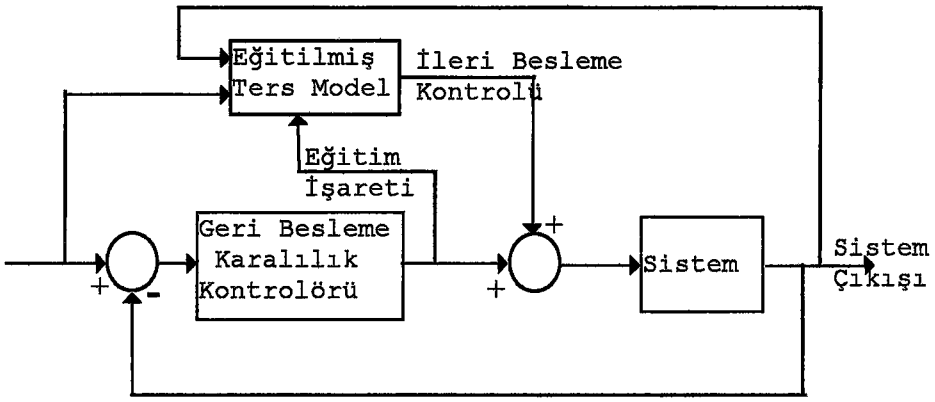
3.7.2 Denetleyicili Kontrol Yapısı

Düşük seviyeli kontrol algoritmalarının, belirli modelleme ve kontrol yapılarının içerisine yerleştirilmesi gereklidir. Açık çevrim öğrenme kontrolörünün formüle edilmesindeki problemlerden biri istenen kontrol işaretinin çoğu zaman mevcut olmamasıdır ve genellikle kontrolörün eğitiminde sadece istenilen çıkış işareti kullanılabilir. Adaptif kontrol alanında formüle edilmiş iki farklı yaklaşım vardır. Bunlar, doğrudan ve dolaylı tasarım yaklaşımlarıdır (Aström ve Wittenmark, 1989).

Doğrudan adaptif kontrol tasarısı arzu edilen kontrolörün açık bir modelini oluşturur. Bunun yanında dolaylı tasarı ise sistemin bir modelini üretir ve önceden tanımlanan optimizasyon/ters çevirme işlemini kullanarak kontrol kuralının sentezini yapar. Örneğin fuzzy kural esaslı bir kontrolör kullanıldığında kendi kendini organize eden fuzzy kontrolörlerinin büyük çoğunluğu doğrudan tasarıdır. Kural esasını güncelleştirecek sistem çıkışındaki hataları kontrol işaretindeki hatalarla irtibatlandıran bir performans endeksi mevcuttur. Bunun aksine tahmin edilen kontrol algoritmalarının kullanıldığı açık bir sistem modelinin genellikle adaptif sinirsel kontrol tasarılarının büyük çoğunluğu dolaylı tasarılardır.

3.7.2.1 Kontrolörlerin kararlılığı

Öğrenen bir ağı eğitmek için kullanılan sabit, dengeleyici lineer geri beslemeli bir kontrolörün en basit doğrudan öğrenen şemalarından biri şekil 3.6'da gösterilmiştir (Kraft ve Campagna, 1990). Lineer kontrolör, kapalı çevrim sisteminin her çalışma bölgesinde kararlı olacak şekilde tasarımı yapılmaktadır ve kontrol işareti öğrenme modülüne bir eğitim işareti sağlamaktadır. Kapalı çevrim sisteminin performansı güncel çalışma noktası değiştikçe ağın iteratif eğilimi çevrimdeki kendi performansını aşamalı olarak iyileştirir. Çalışma noktası değiştikçe öğrenen kontrolör, sistem başlangıç şartlarına geri döndüğünde, öğrenilmiş cevabı unutmayacak şekilde istenilen kontrol düzleminin lineer olmayan bir modelini



Şekil 3.6 Doğrudan Öğrenen KontrolŞeması

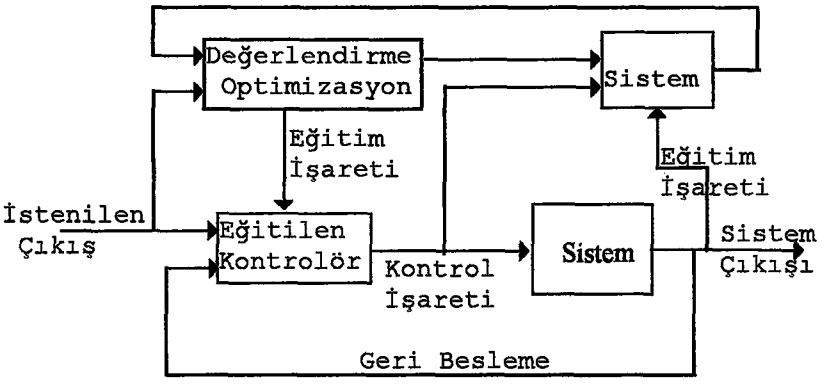
geliştirir ve bu model iyileştirilebilir. Bu durum geçici olarak kararlı olan bir modelini gerektirir; giriş uzayındaki bir bölge hakkında öğrenme diğer bölgedeki saklı bilgiyi en az düzeyde etkiler.

Algoritmanın basitliğinin yanında bu yaklaşımın önemli bir sakıncası vardır; sabit lineer kontrolör tasarımı. Bu kontrolörlerin tasarımıyla kıyaslandığında, algoritmanın daha sağlam olduğu iddia edilmiştir ancak öğrenme modülünün yakınsama oranı eğitim işaretinin kalitesine bağlıdır. Lineer kontrolörün performansı zayıfsa öğrenme modülünün kavraması yavaş olur.

3.7.2.2 Tahmini öğrenme kontrol şemaları

Dolaylı öğrenmeli kontrol şemaları, gelecekteki birçok zaman adımları için kendi hareket tarzının etkilerini değerlendirerek ve en uygun güncel kontrol hareketini seçerek bir kontrol stratejisini belirlemeye çalışır ve bunu bilahare sisteme uygular. Yapı ise bir, sistem modelinin gelişimini bir kontrol değerlendiren bir performans fonksiyonunu ve en iyi kontrol hareketini belirleyen bir optimizasyon tekniğine ihtiyaç gösterir. Öğrenme kontrol elemanının dahil edildiği bir şema şekil 3.7'de gösterilmiştir ve burada yeterli eğitimden sonra tam optimizasyon hesaplarının yapılmasına gerek olmadığı gibi hesaplayıcı kaynaklar diğer görevlere tahsis edilebilir. Eğer sistem zamanla değişiyorsa, model genellikle adaptiftir. Ancak işlem modeli zayıfsa, başlangıç optimizasyon hesapları çok zayıf kapalı döngü kontrolü sağlayabilir.

Şayet, sistem iyi ve performans fonksiyonu ile arama stratejisi iyi seçilmişse bu kontrol şeması mükemmel kapalı

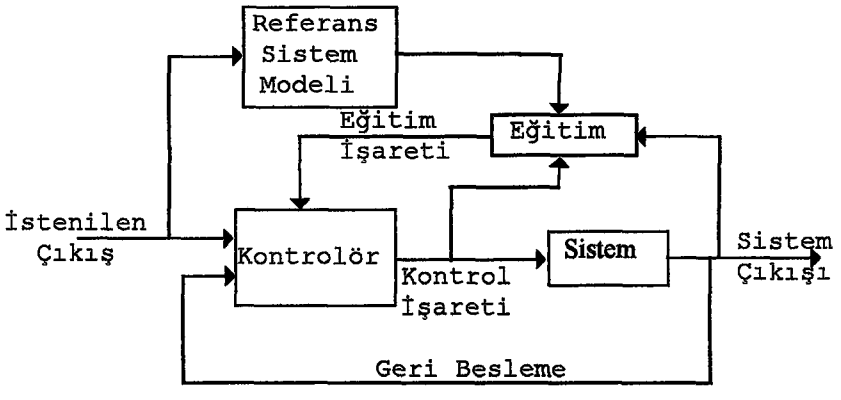


Şekil 3.7 Tahmini Öğrenme Kontrol Blok Diyagramı

döngü kontrolü sağlayabilir. Bununla beraber çok basamaklı ileri optimizasyon hesapları genellikle çok pahalıdır ve sadece zaman yönünden kritik olmayan sistemlere uygulanabilir. Bir kısmı Tolle ve Ersü (1992) tarafından açıklanan bu yapıyla ilgili birçok basitleştirmeler sağlanabilir ve bu tekniği gerçek zaman kontrolleri için daha uygun kılar (Brown ve Harris, 1994).

3.7.2.3 Model referanslı adaptif kontrol

Model referanslı öğrenme kontrol yapısı, lineer adaptif kontrol alanında yaygın bir şekilde kullanılmıştır.Model referanslı öğrenme kontrol şeması şekil 3.8'de gösterilmiştir. Kontrolün amacı, sistem çıkışı y(t)'nin, referans sistem çıkışı d(t)'yi asimtotik olarak takip ederek, kontrol işaretini kararlı kılacak şekilde ayarlamaktır ve burada;



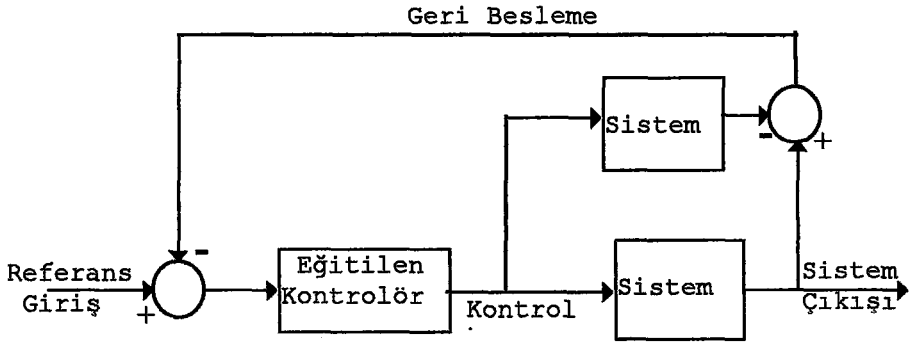
Şekil 3.8 Model Referanslı Kontrol Yapısı Blok Şeması

$\lim_{t\to\infty} \|y(t)-d(t)\| \leq \varepsilon$ (ε küçük bir pozitif sabit) (3.14)

Bu algoritmanın performansı, uygun bir referans modelin seçimi ve uygun bir öğrenme mekanizmasının türetilmesine bağlıdır. 1960'lardaki araştırmacılar basit gradyant tabanlı öğrenme kurallarının bazen yetersiz kaldığını fark ettiler ve durumun daha genel lineer olmayan sistem modelleri ve kontrolörleri için de bu şekilde olmaması için hiçbir sebep yoktur.

3.7.2.4 Dahili model kontrolü

Dahili model kontrolü, tahmini öğrenme kontrol şemalarına benzer bir yapıyı kullanmaktadır. Bir öğrenme modülü, model referanslı adaptif kontrol şemasında kullanılan referans işaretinden ziyade uygulanmış olan kontrol işaretini alarak işlemi doğrudan modellemede kullanılır. Model ile ölçülmüş sistem çıkışı arasındaki hata,bir geri besleme işareti olarak kullanılır ve bilahare



Şekil 3.9 Dahili Model Kontrol Yapısı Blok Şeması

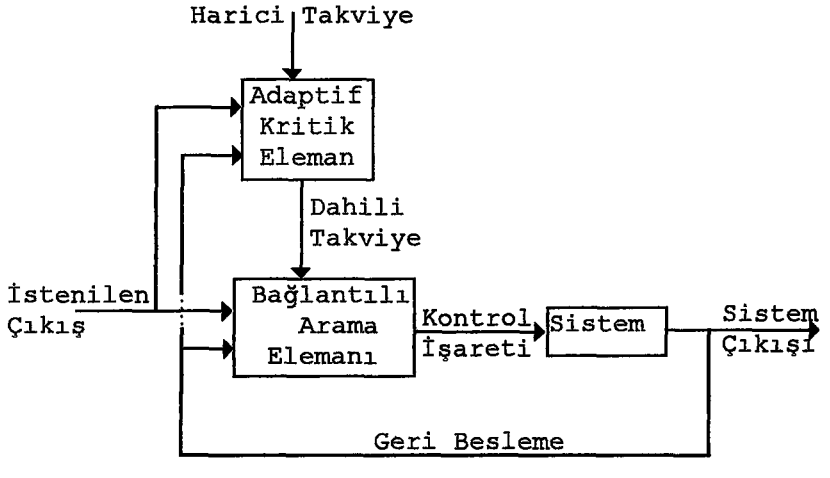
kontrolöre aktarılır. Dahili sistem modeli genellikle bir ters sistem modeli olarak tasarlanır(ters sistem modeli mevcutsa) ve daha önceki bölümde açıklanan ters modelleme şemalarından herhangi biri uygun bir kontrolörü sentez etmede kullanılır. Dahili model kontrolünün blok şeması şekil 3.9'da görülmektedir (Sbarbaro-Hofer v.d.,1993).

Dahili model kontrol çevrimleriyle ilgili birçok teorik kararlılık sonuçları mevcuttur. Ancak bu sonuçlar sistemin açık çevrim kararlılığıyla birlikte kesin ya da ters modellemenin olduğunu kabullenir. Bu kabullerin yapılmasına rağmen bu yaklaşımın, lineer olmayan sistemleri kapsadığı ve kararlı analiz sonuçları verdiği iddia edilmektedir.

3.7.3 Takviye edilmiş öğrenme sistemleri

Barto'nun 1983'de yayınladığı makale ile takviyeli öğrenme şemalarının, yapay sinir ağlarıyla yakın ilişkili oldukları ortaya çıkmıştır. Takviyeli kontrol şemaları en düşük denetleyicili öğrenme algoritmalarıdır; kullanılabilir tek bilgi, kontrol hareketlerinin belirli bir kümesinin başarılı olup olmadığıdır. İlk uygulama, ters bir pendulumu dengelemede kullanılarak platformun başlangıç noktasından belirli bir mesafeye hareket etmeyecek ve ters pendulumun dikey durumda kalacak şekilde sınırlanarak kontrolünde kullanılmıştır. Bu sınırlayıcılardan herhangi biri ihlal edildiğinde öğrenme algoritmasına bir hata işareti gönderilmektedir.

Kontrolörün ters pendulumun dengesini iyi bir şekilde sağladıktan sonra, tanımdan da anlaşılacağı gibi çok seyrek olarak oluşan bozulmalar için daha az bir eğitimin gerekli olduğu açıkça görülmektedir. Barto tarafından önerilen çözüm, bir kontrol şemasının yapılmasından ibarettir ve iki adet adaptif elemandan oluşur; bağlantılı arama elemanı ve adaptif kritik eleman. Şekil 3.10'da görüldüğü gibi bağlantılı arama elemanı verilen performans hedeflerini sağlayacak olan en uygun kontrol işaretlerini yeniden üretmeye çalışırken, adaptif kritik eleman da kontrolörün performansını içten denetlemeye ve bağlantılı arama elemanını eğitmek için dahili bir güçlendirilmiş işaret sağlar. Adaptif kritik eleman harici işaret kullanılarak eğitilir. Kontrol elemanının sürekli iç eğitimi, sistemin tüm performansını büyük ölçüde geliştirdiği belirlenmiştir. Geçmiş yıllarda içinde tüm sistemin teorik olarak daha iyi anlaşıldığı ve bu tekniğin değişik biçimini kullanan birçok simülasyon ve uygulamaları yapılmıştır.



Şekil 3.10 Takviyeli Öğrenme Kontrol Yapısı Blok Şeması

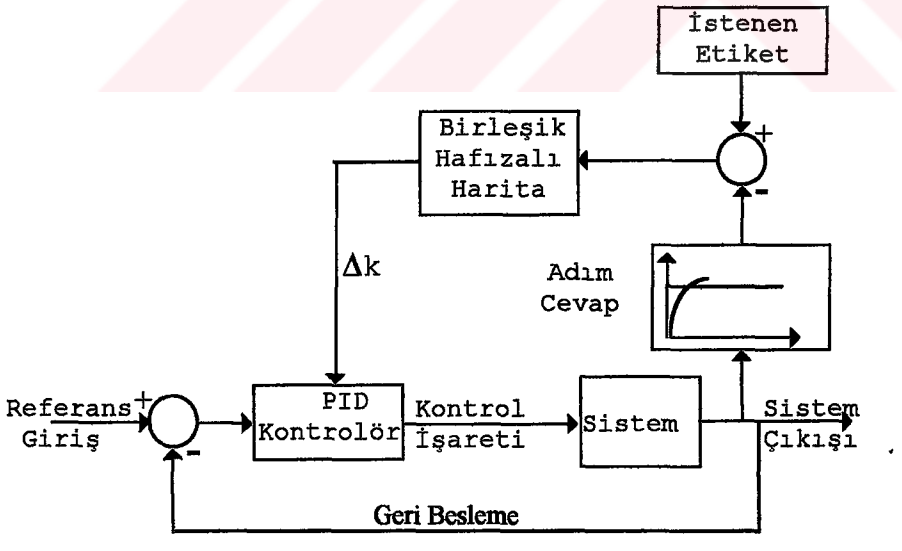
3.7.4.Lineer kontrolörleri parametreleme

Son yıllarda çok sayıda farklı sinirsel modelleme ve kontrol yapıları önerilmektedir. Klasik kontrol şemalarıyla bağlantılı olan bu önerilerin birçoğu önceki bölümler de açıklanmıştır. Sinirsel kontrol yapılarını başlatan bu orijinal parametreleme yapay sinir ağlarına duyulan yakın ilginin sonucu olarak ortaya çıkmıştır. Bu model herhangi bir fonksiyonel bağıntıyı öğrenmek için adaptif sistemlerin kabiliyetini ortaya çıkarmaya çalışır. Akıllı bir kazanç-proglamlama yaklaşım tipinin kullanılması için bir çok sebep vardır; Lineer geri beslemeli kontrolörlerin sanayi ve endüstri de yaygın olarak kabul görmesi, sağlamlık ve kapalı çevrim kararlılığı ile ilgili çok sayıda teorik ve uygulamalı sonuçların mevcut oluşu ve bunların yerine getirilmesindeki düşük maliyettir. Başarılı sistemler maliyetlerin düşmesine sebep olur ve zaman değişimli sistem dinamiklerine uyum sağlama özelliğini taşırlar.

Tüm bu yaklaşımlar, problemi basitleştirdiğinden sistemin yapısı ile ilgili bazı geçmiş bilgileri kabullenmektedir. Kumar ve Guez (1991) dolaylı bir kontrol tasarım yapısını benimsemişlerdir. Sistemin yavaş şekilde değişen ikinci derece lineer bir sistem kabul edilmekte ve kapalı çevrim cevabını açıklayan bir dizi özellikler çıkarılmaktadır. Bu özellikler gecikme zamanı, yükselme zamanı, maksimum taşma,durulma zamanı v.s. içerebilir ve bu özellikler sistem parametrelerini tahmin eden bir adaptif rezonans teori esaslı sınıflandırıcıya iletilir. Bu çıkış arzulanan bir dizi kapalı çevrim cevap karakteristiğiyle birlikte, klasik kutup değişim tekniklerini benimseyerek bir dizi lineer kontrol kazançları üretmek için kullanılır.

Kapalı çevrim sistem cevabını tanımlayan özellik etiketleri çıkarma fikri Lawrence ve Harris (1992) tarafından ayrıca önerilmiştir. İstenen kapalı çevrim cevabı sistemin kapalı çevrim basamak cevabını tanımlayan etiketlerle kıyaslanan bir dizi özellik etiketi ifade edilir. Şekil 3.10'da gösterildiği gibi, bu hata vektörü sistem cevabının istenen çıkışa daha yakın olması için PID parametrelerindeki değişikliği tahmin eden yapay sinir ağına iletilir (Brown ve Harris, 1994) .

Sadece giriş-çıkış verilerini kullanarak bir dizi açık çevrim PID parametrelerini sentez etme yeteneği uzun yıllardan beri araştırılmaktadır. Genellikle PID kontrolörlerinin dayanaklılığını arttırabilen böyle bir sistemin geri ödeme potansiyeli oldukça büyüktür ve bu araştırma sahası hala gelişmeye oldukça elverişli bir durumdadır.



Şekil 3.11 Direk PID Kontrol Tabanlı TahminiKontrolYapısı Blok Şeması

3.8 Lineer Birleşimler

Adaptif lineer birleşimler yapay sinir ağlarının temelini oluşturlar. Adaline, perseptron ve daha birçok sinir ağında sıkça kullanılımaktadır. Adaptif lineer birleşimler basit olarak girişlerin ağırlıklı toplamını oluşturur. Bu değer sinir ağının görüntü tanıma işinde kullanılırsa, ikili çıkış üretimi için başlangıç olarak kullanılabilir. p boyutlu ağ giriş vektörü x ve ağırlık vektörü w olsun. Sürekli lineer birleşim çıkışı y;

$$v = \sum_{i=1}^{p} net_i w_i = net^T w \qquad (3.15)$$

Burada net ≡ x'tir ve adaptif lineer birleşim şekil 3.11'de gösterilmiştir. Genellikle çoğaltılmış a_0 terimi bire eşitlenir ve bias olarak adlandırılır. Buradaki ağın çıkış sınır değeri ise;

$$\text{eğer } a^T w > 0 \quad \text{diğer durumlarda} \quad \begin{cases} 1 \\ 0 \end{cases}$$

Sinir ağının girişi ön işlemden geçirilmişse ($x \rightarrow a$) perseptron da ilave bir katmana sahip olur ve değiştirilmiş bu girişlerin ağırlıklı kombinezonu alınır.

Hata düzeltme eğitim kuralı kullanılarak, ağırlık vektörü w ayarlanabilir ve ağın genel davranışı kabul edilecek durumdaysa eğitim sona erer. Eğitim giriş kümesi örneğindeki $\{a(t)\}_{t=1}^{L}$ lineerleştirilebiliyorsa o zaman ağ doğru sonucu gerçekleştirir. Bu durumda öğrenme algoritması az sayıdaki iterasyonla son bulur. P boyutlu giriş uzayındaki eğitim girişlerini a(p-1) boyutlu hiper düzlemle ayrılabilirse eğitim küme örnekleri lineerleştirilebilir.

Bu hiperdüzlem $0=a^Tw$ sağlayacak ağırlıklar kümesinden oluşur. Bu durum adaptif lineer birleşimler için tanımlayıcı sınırı belirler.

3.8.1 Lineer modeller

t zamanında lineer model çıkışı;

$$y(t)=\sum_{i=1}^{p} a_i(t)w_i(t-1)=a^Tw(t-1) \qquad (3.16)$$

olarak ifade edilir. Burada y(t), modelin skaler çıkışı a(t) ise p boyutlu giriş vektörü, w(t-1) ise p-boyutlu parametre vektörüdür. Ağırlık vektörü her çıkış hesaplamasından sonra alınırsa model giriş vektörüne göre tam olarak lineer olmaz, çünkü farklı zamanlarda aynı girişin ağa verilmesiyle farklı model çıkışları elde edilir. Ortaya çıkan bu uyumsuzluğu çözmek için, bir adaptif lineer sistem veya bir adaptif lineer modelin adaptif model olarak tanımlanması gereklidir ki burada ayarlanabilir parametrelerin sabit tutulması durumunda sistem lineer bir model haline gelir.

Denklem 3.16'da verilen lineer modeli iki önemli açıklaması vardır; yanal model ve çok girişli model. Çok girişli model girişlerini aynı anda birçok farklı kaynaktan sağlar. Modelin p sayıdaki işareti t zamanında algıladığını $\{x_i(t)\}_{i=1}^{p}$ ile gösterelim. O zaman girişler;

$$a_i(t)=x_i(t) \qquad i=1,2,3....p \qquad (3.17)$$

Yanal girişe sadece bir giriş ulaşır. Modele gelen her giriş bu orijinal işaretin gecikmeli şeklidir. Ölçülen işareti x(t) ile gösterdiğimizde, girişler;

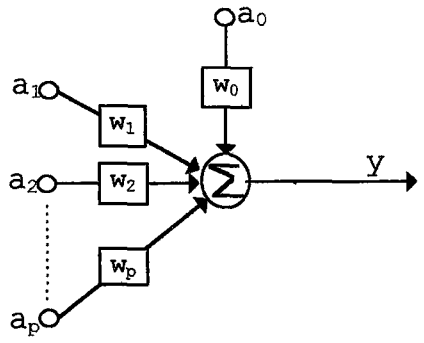
$a_i(t)=x(t-i+1)$ $\quad i=1,2,3...p$ (3.18)

şeklinde olur. Bu farklılıklar büyük çapta uygulamaya bağlıdır ve öğrenme kuralı kullanılan model göz önüne alınmadan türetilebilir. Bununla beraber, göz önüne alınması gereken hususlar,giriş işaretlerinin bağımsız oluşları ve giriş vektörlerinin birbirleriyle olan bağlantılarıdır.

Şayet, sabit fonksiyonların modellenmesi veya sabit ve lineer elemanlara sahip olması istenirse yukarıdaki denklemlere bir bias teriminin ilave edilmesi gerekir. Aksi halde girişin sıfır olması durumunda çıkışta her zaman sıfır olur. Bias terimi girişlerden birinin sabit bir değere eşitlenmesiyle sağlanır (genellikle 1) ve fazladan $a_0(t)$ girişi bias terimini gösterecek şekilde belirtilir.

3.8.2 Model performansı

Bir ağırlık vektörünün uyum sağlaması için uygun bir öğrenme kuralının benimsenmesi gerekir. Denetimli eğitim için, istenen d(t) çıkışının alınması gerekir. Böylece ağ çıkış hatası e(t)=d(t)-y(t) olur. Bu da bize mevcut modelin performansı hakkında bilgi verir. Ağırlık vektörü mevcut modelin performansını iyileştirmek için uğraşacak şekilde ayarlanır ve bu durum çıkış hatasının ağırlık vektörüne geri beslenmesini gerektirir. Anlık çıkış hatasını kullanan, anlık öğrenme algoritmaları genellikle ağırlık vektörünü benimseyen mevcut tüm eğitim verilerini kullanan tüm eğitim kurallarının bir yaklaşımıdır. Bu öğrenme kuralları, önceden belirlenmiş bir eğitim fonksiyonunu



Şekil 3.12 Temel Bir Adaptif Lineer Birleşim

minimize etmeye çalışır.

$$E = \begin{cases} f\left(\left|e_y(t)\right|\right) \\ f\left(e_y^2(t)\right) \\ \max_t \left|e_y(t)\right| \end{cases}$$

Burada, beklenen operatör t için eğitim kümesi;

$$\{x(t), d(t)\}_{t=1}^{L}$$

olarak alınır. İstenen görevin modelini çıkaracak ağı mümkün kılacak bir ağırlık vektörü yoksa, önemsenen her bir anlık hata değişken olduğundan üç performans fonksiyonunun optimal ağırlık vektörleri farklıdır. Performans fonksiyonu aynı zamanda öğrenme kuralının karmaşıklığını belirler. Örneğin $\max_t \left|e_y(t)\right|$ ile verilen bölünmez eğitim işaretini minimize edebilmek için ve çok karmaşık öğrenme kurallarının bulunması gereklidir.

Ortalama karesel hata performans denkleminin doğrudan veya kapalı bir şekilde bir çözümü olmasına rağmen adaptif ağırlık vektörünün eğitimi için iteratif eğitim metotları önerilmektedir. Optimizasyon probleminin boyutu, gerçek

zaman sınırlayıcısıyla birlikte göz önüne alındığında bu yöntem uygun kabul edilebilir.

Ayrıca iteratif yöntemler, en uygun ağırlık vektörü tahminlerinin saflaştırılması esasına dayanır ve genellikle matris dönüşüm işlemi gerçekleştirilmez. Bu durum singuler matrisin tersini gerektirmez ve bu algoritmalar birçok avantajlar sunmaktadır.

-Gerçek zamanda çalışabilen basit dağınık iteratif algoritmalar gerçekleştirilebilir.

-Yararlı düşük-dopruluk dereceli çözümler kolayca elde edilebilir.

Statik bir eğitim veri kümesinden kapalı çevrim ağırlık vektörünün elde edilmesi gerekiyorsa matris dönüşüm kuralları, hatta iteratif algoritma ile yeterli olabilir. İteratif algoritmaları sadece gerçek zaman işlemi kritik bir faktör olduğu zaman kullanılabilir.

BÖLÜM.4 BU TEZDE YAPILAN ÇALIŞMALAR

4.1.Giriş

Klasik kontrol yöntemlerinde, sistem kontrolünde kullanılacak olan kontrolün seçimi, seçilen bu kontrolörün nasıl bağlanacağı ve parametrelerinin tespiti kontrol mühendisleri için önemli bir sorundur. Kontrolör parametreleri, sistem parametrelerine göre seçildiğinden, sistemdeki modelleme hataları, sistemdeki dinamik parametrelerin değişimi ve bozucu etkiler gibi dış etkenlerden dolayı, sistem kararlılığını istenilen ölçüde sağlayamaz. Klasik kontrol yöntemlerindeki tüm bu sorunlara karşın, yapay sinir ağlarının öğrenme ve genelleme yapabilme yeteneklerinin olmasından ve matematiksel denklem kurma zorunluluğu olmamasından dolayı, kontrol alanın da çok geniş olarak kullanılmaya başlanmıştır.

Sistem kontrolünde, herhangi bir fonksiyonu yaklaştırabilme yeteneği diğer ağlara göre daha iyi olan çok katlı ileri veya geri beslemeli yapay sinir ağları ilgi görmüştür. Eğitim algoritması olarak da, belirlenen bir performans kriterinin parametrelere göre kısmi türevlerinin alınarak geriye yayındığı, geriye yayınım algoritması yoğun ilgi görmüştür.

Yapay sinir ağının kontrolör olarak kullanılması genel anlamda iki yöntemle yapılır. Bunlar, doğrudan ve dolaylı kontrol yöntemleridir. Doğrudan kontrol yönteminde yapay sinir ağı, sistemin tersini öğrenecek şekilde eğitilerek yapay sinir ağının bir kontrolör olarak çalışması sağlanır.

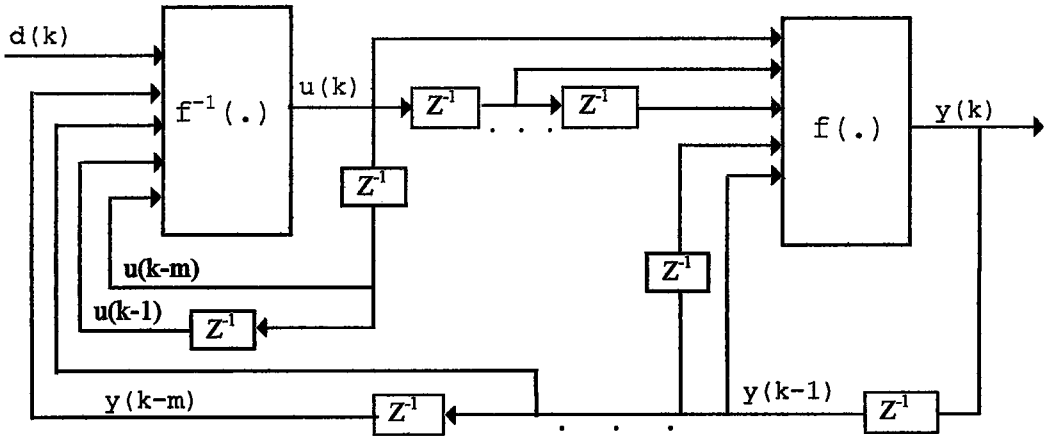
Dolaylı kontrol yönteminde ise yapay sinir ağı, sistemin hem ters hem de düz modelini öğrenecek şekilde eğitilerek; düz modelin sistemin kendisi, ters modelin ise sistemin kontrolörü olarak kullanılması sağlanır.

Bu bölümde, örnek olarak seçilen sistemlerin, yapay sinir ağlarının kontrolör olarak kullanıldığı doğrudan ve dolaylı kontrol yöntemleri ile modellenmesi ve kontrolünün nasıl yapılacağı, simülasyon için hazırlanan program akış şeması ve programlardan elde edilen benzetim sonuçları ayrıntılı olarak incelenmiştir.

4.2.Doğrudan Kontrol Yöntemi İle Sistem Kontrolü

Doğrudan kontrol yapısında yapay sinir ağı sistemin ters modeliyle oluşturulur. Kontrol ağının çıkışı ve sistem ağının çıkışı zaman gecikme operatörü ile birlikte geri besleme yapılarak ağ girişine uygulanır. Ağ çıkışı, girişine geri besleme yapıldığından kontrol ağı dinamik yapıdadır. Şekil 4.1'de doğrudan kontrol yapısının blok diyagramı, şekil 4.2'de ise kontrol yapısına yapay sinir ağından oluşan dinamik bir kontrol ağının yerleştirildiği uyarlamalı doğrudan kontrol yapısının blok şeması görülmektedir.

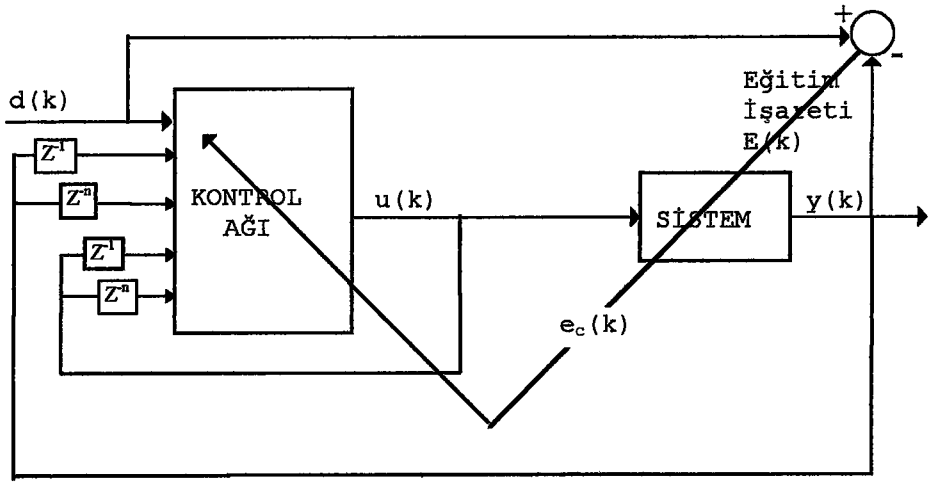
Kontrol ağının girişi, arzu edilen giriş, kendi çıkışının ve sistem girişinin gecikmiş zamanlarından, sistem girişi ise kontrol ağının çıkışından beslenmektedir. Şekil 4.3'de kontrol ağının açık şeması görülmekte ve burada; d,x,u ve y kolon vektörler olmak üzere; x kontrol ağının girişi, d arzu edilen çıkış veya referans giriş, u



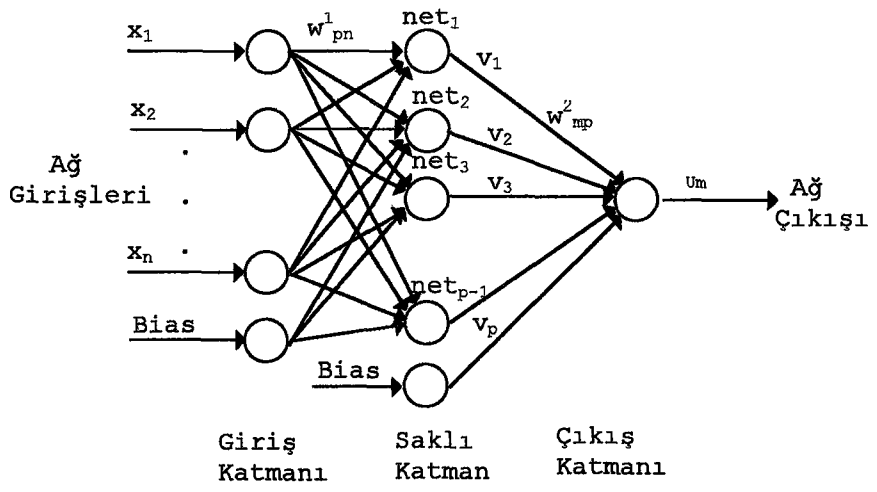
Şekil 4.1 Doğrudan sistem modellemede ters sistemin oluşturulması

kontrol ağının çıkışı ve aynı zamanda sistem girişi, y ise sistem çıkışıdır.

Kontrol ağı, girişi x boyutlu ve çıkışı tek boyutlu olan üç katmanlı bir ağdır. Ağ girişi sistemin derecesine



Şekil 4.2 Doğrudan Kontrol yöntemi Blok Şeması



Şekil 4.3 Kontrolör olarak kullanılan yapay sinir ağı

göre seçilir. Bunun için sistemin fark denklemi elde edilerek sistemin giriş çıkış arasındaki bağıntı;

$$y(k)=f(u(k),u(k-1)...,y(k-1),y(k-2)...) \qquad (4.1)$$

şeklinde elde edilir. Buradan, kontrolün amacına uygun olarak sistemin tersi elde edilerek, kontrol ağının girişleri;

$$u(k)=g(y(k),y(k-1)...,u(k-1),u(k-2)...) \qquad (4.2)$$

şeklinde elde edilir. Ağ girişinde y(k) yerine d(k) arzu edilen çıkış işareti kullanılır ve yakınsamayı artırmak için bir de bias kullanılmıştır.

Ağın eğitiminde, eğitim işaretinin ağırlıklara göre gradyanının geriye yayınımı esasına dayanan geriye yayınım algoritması kullanılmıştır.

Eğitim işlemi yapay sinir ağının giriş örneklerinin her birinin uygulanışında ağırlıklar ayarlanırsa örneksel öğrenme algoritması(çevrim içi=on line öğrenme) bütün örnekler uygulandıktan sonra bir defa ayarlanırsa toplu

öğrenme (çevrim dışı=of line öğrenme) algoritması elde edilir. Eğitim işaretinin ağırlıklara göre gerçek gradyanı toplu öğrenmeyle elde edilir. Örneksel öğrenme gerçekte toplu öğrenmenin yani gerçek gradyanın bir yaklaşığıdır ve toplu öğrenmeye göre başarı şansı daha düşük olabilmektedir(Gökbulut, 1996)

Örneksel öğrenme algoritması, örnekleri tek tek işlediğinden sistem çalışırken, sistem çıkışından alınan örneklerle kontrol ağının eğitimini sağlama imkanı vermesinden dolayı gerçek zamanlı öğrenme olarak ta anılır. Toplu öğrenmede ise belirlenen bir grup örneğin toplu olarak işlenerek ağ ağırlıklarının bir defada değiştirilmesinden dolayı gerçek zamanlı öğrenmede kullanılma şansı yoktur. Fakat toplu öğrenmeyle eğitilen ağın ağırlıkları, örneksel öğrenmede kullanılabilir.(Koçak, 1996)

4.2.1 Kontrolör yapay sinir ağının eğitimi ve eğitim algoritması

Yapay sinir ağının doğrudan kontrol yapısında kontrolör olarak kullanılması için kontrol edilecek olan $f(.)$ fonksiyonunun $h(.)=f^{-1}(.)$ olarak tersinin elde edilmesi ve bulunan bu ters sistemin çok katlı bir yapay sinir ağı ile oluşturulması gereklidir. Bu şekilde ters sistemden elde edilerek oluşturulan yapay sinir ağının bir kontrolör gibi davranması sağlanarak, sistemin uygulanan referans giriş işaretini başarılı bir şekilde izlemesini

sağlayarak, sistem çıkışından uygun bir sistem çıkışı elde edilmeye çalışılır.

Eğitimin amacı, kontrol ağının ters sistemi taklit edecek şekilde öğretilmesini sağlamaktır. Bu sebeple yapay sinir ağının sistemin tersini temsil edebilecek şekilde doğru olarak oluşturulması gereklidir.

Ağın oluşturulması, sistem giriş çıkışı ve sistem derecesine göre seçilir. Ağ girişleri, sistemin ayrık zaman denklemindeki gecikmelerden ve sisteme uygulanan giriş fonksiyonundan oluştuğu için, ağ girişindeki hücre sayısı sistemin derecesine göre seçilmelidir. Fakat, fark denklemlerindeki bazı zaman gecikmelerinin girişe uygulanmaması ile ağ girişlerinin sayısı düşürülebilir. Dolayısı ile sistemin derecesi düşürülmüş bir modeli elde edilebilir ve böylece eğitim süresi düşürülmüş olur.(Bates v.d., 1993) Bu şekilde bir eğitimle, eğitim süresi düşürülmesine karşın ağın genelleştirme özelliği de önemli ölçüde düşmektedir. Bu sebeple, ağ oluşturulurken mümkün olduğunca, ağ girişindeki hücre sayısının sistem derecesine göre seçilmesine gayret edilmelidir.

Kontrol ağının, saklı katmanındaki hücre sayısının seçimi ise tamamen tecrübeyle belirlenmelidir. Genel olarak, M giriş hücrelerinin sayısı, N çıkış hücrelerinin sayısını göstermek üzere, saklı katmandaki hücre sayısı N(M+1) olarak belirlenebilir. (ZHANG v.d., 1995) Ama bu seçimde bir kesinlik yoktur. Saklı katmandaki hücre sayısı bu değerin üzerinde veya altında da seçilebilir, fakat saklı katmandaki hücre sayısının giriş katmanındaki hücre sayısından çok yukarıda veya çok altında seçilmemesine özen gösterilmelidir. Zira saklı katmandaki hücre sayısının fazla seçilmesi halinde ağırlık taşması denilen, ağın

öğrenmiş gibi göründüğü halde öğrenememe durumu ortaya çıkabilmektedir.

Çıkış katmanındaki hücre sayısı ise tamamen sistem girişine göre seçilmelidir. Sistem tek girişli ise kontrol ağının da çıkış katmanında tek hücre bulunmalıdır.

Kontrolör olarak kullanılacak olan yapay sinir ağı bu şekilde oluşturulduktan sonra yeni kontrol yapısı şekil 4.2'deki hali alır. Ağ eğitimi hatanın geriye yayınımı kullanılarak yapılır. Bu amaçla eğitim işareti E, çıkıştaki hataların karelerinin toplamı, d_j arzu edilen çıkış ve y sistem çıkışı olmak üzere k. örnekteki çıkış hatası ve eğitim işareti,

$$e_j(k)=d_j(k)-y_j(k) \quad j=1,2...m \tag{4.3}$$

$$E(k)=\frac{1}{2}\sum_{j=1}^{m} e_j^2(k) \quad k=1,2,N \tag{4.4}$$

olarak hesaplanır.

Ağın giriş katmanını, saklı katmana bağlayan ağırlıklar w_{il}^1, saklı katmanı çıkış katmanına bağlayan ağırlıklar w_{ji}^2, x=nx1 giriş vektörü, v=px1 saklı katman çıkış vektörü θ_1 ve θ_2 , biaslar, $\Phi(.)$ saklı katmanda kullanılan hareketlendirme fonksiyonu olmak üzere üç katmanlı doğrusal giriş ve çıkışa sahip çok katlı yapay sinir ağında, katmanlar arasındaki bağıntılar aşağıdaki şekilde bulunabilir.

$$net_i=\sum_{l=0}^{n} w_{il}^1 x_l \qquad l=1,...n \tag{4.5}$$

$$v_i=\Phi_i(net_i) \qquad i=1,...p \tag{4.6}$$

$$y_j=\sum_{i=0}^{p} w_{ji}^2 v_i \qquad j=1,...m \tag{4.7}$$

Eğitim işaretinin, ağırlıklara göre minimize edilebilmesi için eğiminin belirlenebilmesi gereklidir. Bu

sebeple her bir k. örnekte kontrol ağının saklı katmanındaki ağırlıklarının uyarlanmasında,

$$w_{ji}^{2}(k)=w_{ji}^{2}(k-1)+\Delta w_{ji}^{2}(k) \tag{4.8}$$

$$\Delta w_{ji}^{2}(k)= -\alpha\frac{\partial E_j(k)}{\partial w_{ji}^{2}(k)}+\beta\Delta w_{ji}^{2}(k-1) \tag{4.9}$$

giriş katman ağırlıklarının uyarlamasında,

$$w_{il}^{1}(k)= w_{il}^{1}(k-1)+\Delta w_{il}^{1}(k) \tag{4.10}$$

$$\Delta w_{il}^{1}(k)= -\alpha\frac{\partial E_j(k)}{\partial w_{il}^{1}(k)}+\beta\Delta w_{il}^{1}(k-1) \tag{4.11}$$

denklemleri kullanılır.

S-ağırlıkları uyarlama iterasyonu olmak üzere, N adet örneğin uyarlanmasından sonra ağırlıkların toplu olarak uyarlanmasında ise,

$$\Delta w(S) = -\alpha\frac{\partial E}{\partial w(S)}+\beta\Delta w(S-1) \tag{4.12}$$

$$w(S)= w(s-1)+\Delta w(S) \tag{4.13}$$

bağıntıları kullanılır.

Doğrudan kontrol yönteminde, ağ çıkışı sistem girişine verildiğinde ve öğrenme algoritması olarak hatanın kısmi türevleri bulunarak geri yansıtmasını hedefleyen geriye yayınım algoritması kullanıldığından hatanın sistem üzerinden geri yansıtılması gerekmektedir.

Bu nedenle sistemin $\frac{\partial y}{\partial u}$ kısmi türevinin veya en azından sistem türevinin işaretinin bilinmesi gerekmektedir (Zhang v.d., 1995).

Şekil 4.2'de görülen,hatanın sistem üzerinden yansıtılarak elde edilen $e_c(k)$ hatası,

$$e_c(k) = e_j(k)\frac{\partial y(k)}{\partial u(k)} \tag{4.14}$$

bağıntısından hesaplanabilir. $\frac{\partial y(k)}{\partial u(k)}$ kısmi türevi sayısal türev yöntemleri ile ,küçük bir hata yaklaşıklığı ile ,

$$\frac{\partial y(k)}{\partial u(k)} = \frac{\partial y(k) - \partial y(k-1)}{\partial u(k) - \partial u(k-1)} \tag{4.15}$$

bağıntısından bulunabilir.

O halde, geriye yayınım algoritmasında, denklem 4.9 ve 4.11 deki ifadelerindeki $\frac{\partial E_j(k)}{\partial w_{ij}^2(k)}$ ve $\frac{\partial E_j(k)}{\partial w_{il}^1(k)}$ ifadeleri;

$$\frac{\partial E(k)}{\partial w_{ij}^2(k)} = \frac{\partial E(k)}{\partial e(k)}\frac{\partial e(k)}{\partial y(k)}\frac{\partial y(k)}{\partial u(k)}\frac{\partial u(k)}{\partial w_{ij}^2(k)} \tag{4.16}$$

$$\frac{\partial E(k)}{\partial e(k)} = e(k) \tag{4.17}$$

$$\frac{\partial e(k)}{\partial y(k)} = -1 \tag{4.18}$$

$$\frac{\partial y(k)}{\partial u(k)} = \frac{\partial y(k) - \partial y(k-1)}{\partial u(k) - \partial u(k-1)} \tag{4.19}$$

$$\frac{\partial u(k)}{\partial w_{ij}^2(k)} = v_i(k) \tag{4.20}$$

$e_c(k) = e_j(k)\frac{\partial y(k)}{\partial u(k)}$ olduğundan

$$\frac{\partial E(k)}{\partial u(k)} = -e_c(k) \tag{4.21}$$

dır. Bu durumda,

$$\frac{\partial E(k)}{\partial w_{ij}^2(k)} = -e_c(k)v_i(k) \tag{4.22}$$

$$\frac{\partial E(k)}{\partial w_{il}^{1}(k)} = \frac{\partial E(k)}{\partial e(k)} \frac{\partial e(k)}{\partial y(k)} \frac{\partial y(k)}{\partial u(k)} \frac{\partial u(k)}{\partial v_i(k)} \frac{\partial v_i(k)}{\partial net_i(k)} \frac{\partial net_i}{\partial w_{il}^{1}} \qquad (4.23)$$

$\frac{\partial E(k)}{\partial e(k)} \frac{\partial e(k)}{\partial y(k)} \frac{\partial y(k)}{\partial u(k)} = -e_c(k)$ olarak hesaplanmıştı,

$$\frac{\partial u(k)}{\partial v(k)} = w_{ij}^{2}(k) \qquad (4.24)$$

$$\frac{\partial v_i(k)}{\partial net_i(k)} = \Phi_i'(net) \qquad (4.25)$$

$$\frac{\partial (net)}{\partial w_{il}^{1}} = x_1 \qquad (4.26)$$

$$\frac{\partial E(k)}{\partial w_{il}^{1}(k)} = -e_c(k)\Phi_i'(net)w_{ij}^{2}(k)x_1(k) \qquad (4.27)$$

olarak hesaplanabilir ve bu bağıntılarla kontrol ağındaki ağırlık değerleri ayarlanmış olur.

Eğitime başlamadan önce, kontrol ağındaki ağırlıklar ilk halde rasgele seçilerek ağın simetrisi bozulur. Bu durumda ağ, sisteme hatalı bir u(k) girişi uygular ve sistemden hatalı bir y(k) değeri elde edilir. Elde edilen bu hatalı y(k) değeri referans giriş işareti d(k) ile karşılaştırılır.Ortaya çıkan hata işareti, geriye yayınım algoritması kullanılarak yukarıdaki bağıntılara bağlı olarak kontrol ağındaki ağırlık değerlerinin ayarlanması sağlanır. Bu şekilde tekrarlanan eğitimle ağ, referans giriş işaretine nasıl doğru cevap vereceğini öğrenir. Eğitim sayısı artırıldıkça ağ daha da iyi öğrenerek çıkış hatasını azaltır. Hatanın istenilen değerin altına düşmesiyle eğitim kesilir ve kontrol ağının giriş ve saklı katmanındaki ağırlıklar saklanarak eğitim işlemi kesilir. Eğitimin kesilmesine, toplu öğrenmede hataların karelerinin toplamının istenilen değere düşmesi ile, örneksel öğrenmede

ise program zaman zaman durdurularak sistem çıkışındaki düzelme kontrol edilerek karar verilir.

4.2.2 Seçilen bir sistemin ters yön modeli için yapay sinir ağı seçimi ve simülasyon sonuçları

Bir sistemin yapay sinir ağları ile modellenebilmesi için sistemin tam bir matematiksel modeline ihtiyaç duyulmamakla birlikte kontrol ağının, giriş ve çıkış katmanlarındaki hücre sayısının belirlenebilmesi için sistemin derecesini belirleyen yaklaşık bir modeline ve giriş çıkış davranışının iyi bilinmesine ihtiyaç vardır. Ayrıca doğrudan kontrol yönteminde çıkış hatası sistem üzerinden yansıtıldığı için sistemin türevinin veya türev işaretinin bilinmesi gereklidir.

Daha önceden de belirtildiği gibi hem doğrudan hem de dolaylı kontrol yapılarında eğitim, toplu ve örneksel olarak yapılabilir. Toplu öğrenmede belirli miktarda sistemin giriş değerlerine karşılık sistemin çıkış değerlerinin ölçüm yoluyla veya deneysel olarak önceden belirlenmesi gereklidir. Örneksel eğitimde ise örnekler tek tek eğitildiğinden ağ eğitiminde kullanılacak sistemin giriş çıkış değerlerine önceden sahip olmaya gerek yoktur. Sistem çalışırken sistem giriş ve çıkışından örnekler alınarak ağ eğitimi yapılabilir. Bu nedenle, örneksel eğitime gerçek zamanlı eğitim adı da verilir. Bu çalışmada kullanılan sistemler her iki eğitim metoduna göre kontrol edilecektir.

Bu çalışmada pratikte çok karşılaşılan, ikinci dereceden sistemler kullanılacaktır. Yapılan çalışmada, pratik kısım olmayıp, benzetim yapıldığından matematiksel modeli bilinmeyen sistemler kullanılmayacaktır. Ele alınan örnek sistemlerin matematiksel modellerinin olduğu kabul edilecektir. Ele aldığımız sistemin doğrudan kontrol yapısına yerleştirilmesi için, sistemin fark denklemine ihtiyaç vardır. Sürekli zamandan geçişte, örnekleme periyodunun, sistemin zaman sabitinin en az onda biri kadar olmasına dikkat edilmelidir.

Kontrol edilmek üzere seçilen ikinci dereceden sistemin transfer fonksiyonu,

$$G_{(s)}=\frac{2}{S^2+2S+2} \qquad (4.28)$$

şeklindedir. Transfer fonksiyonunun seçiminde, uygulanan girişe karşı, bozuk bir çıkış vermesine dikkat edilmiştir. Eğitim sırasında uygulanan giriş işareti ise, sistemin dinamik davranışlarını tam olarak ortaya çıkarabilmesi için sinüzoidal giriş seçilmiştir. Transfer fonksiyonun ayrık zamanda elde edilen giriş çıkış bağıntısı aşağıdaki şekilde elde edilmiştir.

$$y(k)=f(u(k),u(k-1),u(k-2),y(k-1),y(k-2)) \qquad (4.29)$$

Kontrol ağının oluşturulması için gerekli olan ters sistem modeli için 4.27 denkleminden,

$$u(k)=f(y(k),y(k-1),y(k-2),u(k-1),u(k-2)) \qquad (4.30)$$

olarak ters sistem modeli elde edilir. 4.30 denkleminde y(k) yerine arzu edilen çıkış değeri olan referans giriş işareti kullanılarak 4.30 denklemi aşağıdaki şekilde düzenlenir.

$$u(k)=f(d(k),y(k-1),y(k-2),u(k-1),u(k-2)) \qquad (4.31)$$

Böylece kontrol ağının giriş katmanında bulunması gereken hücre sayısı belirlenmiş olur. Bu girişlere birde,

değeri 1 olan bias terimi eklenerek giriş katmanının sayısı sekiz olarak hesaplanmış olur. Sistem tek giriş tek çıkışlı olduğundan ağın çıkış katmanındaki hücre sayısı da bir olmak zorundadır. Saklı katmandaki hücre sayısının belirlenmesinde daha öncede belirtildiği gibi kesin bir yöntem olmadığından burada saklı katmandaki hücre sayısı tecrübeyle yedi olarak seçilmiştir.

Yapay sinir ağının eğitiminde kullanılan, momentum katsayısı ve öğrenme oranının seçiminde de kesin bir yöntem yoktur. Fakat ağın kendi içerisinde momentum katsayısını ve öğrenme oranını adaptif olarak ayarladığı bazı yöntemler bulunmaktadır (Drago v.d., 1995). Ancak bu yöntemler toplu öğrenmede kullanılabildiğinden, toplu öğrenme algoritmasının kullanıldığı programlarda momentum katsayısı ve öğrenme oranı uyarlanarak bulunmuştur. Örneksel öğrenme algoritmasının kullanıldığı programlarda ise momentum katsayısı ve öğrenme oranı deneme yolu ile en uygun değerleri bulunarak dışarıdan verilmiştir.

Kontrol ağının saklı katmanında hareketlendirme fonksiyonu olarak, pozitif ve negatif bölgede başlangıç noktasına göre simetrik olması bakımından tanh() fonksiyonu seçilmiş, giriş katmanında doğrusal (f(x)=x) fonksiyonu kullanılmıştır. Çıkış katmanında ise sistem girişinin sınırlanması gerekli olduğu durumlarda tanh(.) fonksiyonu, sistem girişinin bir sınırlamaya ihtiyaç olmadığı durumlarda ise f(x)=x doğrusal fonksiyonu kullanılmıştır.

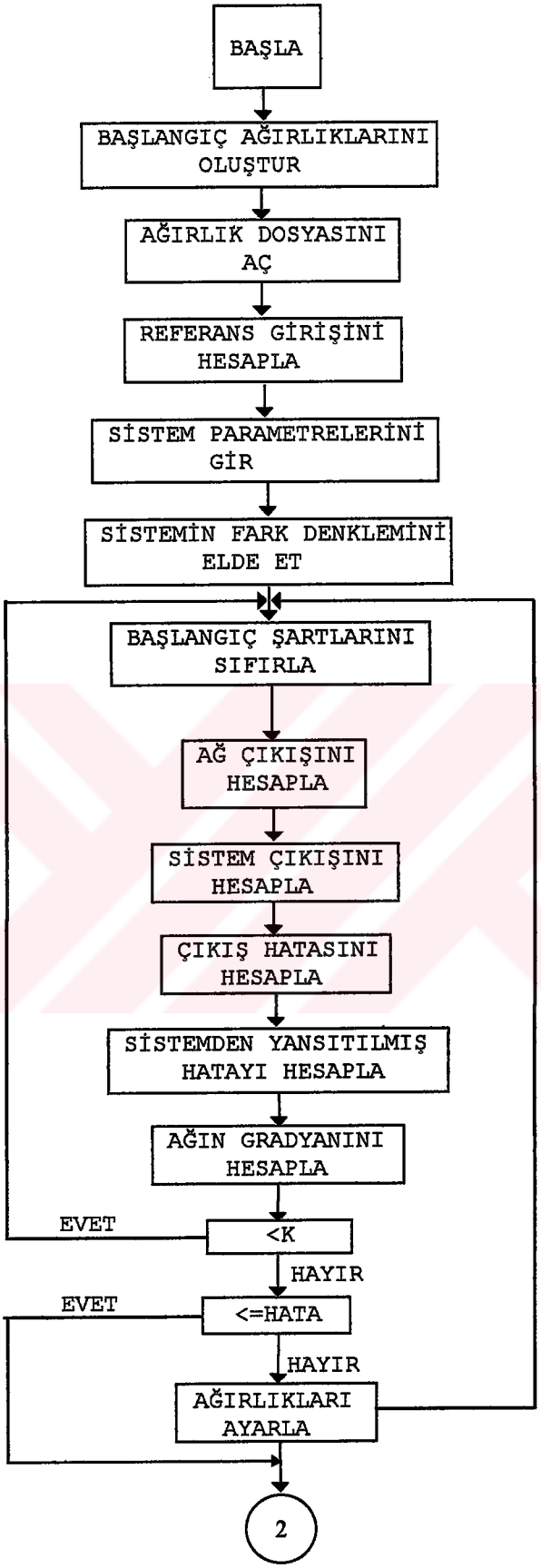
Kontrolör olarak kullanılacak olan yapay sinir ağı yukarıda anlatıldığı şekilde oluşturularak simülasyon için, Tc++ dilinde toplu öğrenme için TTK1, örneksel öğrenme için TTK2 adlı iki bilgisayar programı hazırlanmış ve hazırlanan bu programlarla kontrol ağı eğitilmiştir. Toplu öğrenmede hata minimum değere geldiğinde, örneksel öğrenmede ise

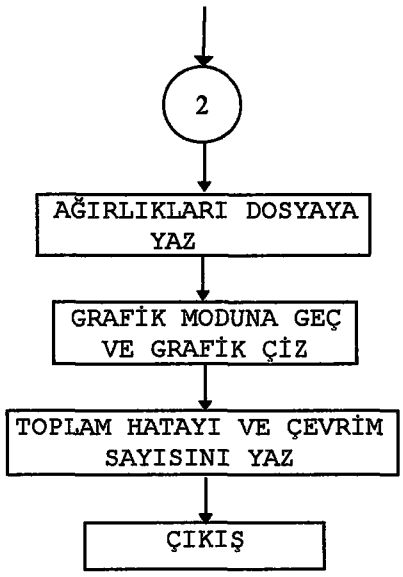
sistem çıkışı kontrol edilerek kontrol ağının, sistemi yeterince kontrol ettiğine emin olunduktan sonra eğitim kesilmiştir.

Şekil 4.4 de toplu öğrenme ile doğrudan kontrol yöntemi için hazırlanan programın (TTK1) akış şeması görülmektedir. Bu programda kullanılan ana program EK1'de verilmiştir.

Şekil 4.5'te toplu öğrenme algoritması kullanılarak 4.28 denklemindeki sisteme göre sinüzoidal referans giriş işaretine göre 0.09 hata ile eğitildikten sonra elde edilen kontrol edilmiş çıkış işareti, sistemin kontrol edilmeden önceki sinüzoidal giriş işaretine verdiği cevap ve referans giriş işareti görülmektedir. Şekil 4.5'den de görüldüğü gibi sistem çıkışı arzu edilen çıkış işareti olan sinüzoidal çıkış işaretini yeterince izlediğinden, 0.09 hata yeterli görülerek eğitim kesilmiştir. Eğitim kesildikten sonra kontrolör olarak kullanılan yapay sinir ağının giriş ve çıkış katmanındaki ağırlıklar bir dosyaya kaydedilerek saklanmıştır. Bundan sonraki işlemlerde artık kontrol ağı eğitilmeyecek ve ağırlıkları bu en son eğitimden elde edilen ağırlık değerleri kullanılacaktır.

Kontrolör olarak kullanılan yapay sinir ağı yukarıda belirtilen hata ile eğitildikten sonra, yapay sinir ağlarının en önemli özellikleri olan öğrenme ve genelleştirme yetenekleri incelenecektir. Öğrenme özelliğini belirlemek amacıyla, eğitimde kullanılan sistemin parametrelerinde rasgele değişiklikler yapılarak kontrolör olarak kullanılan yapay sinir ağının, değişen bu yeni sistemleri de kontrol edebilme yeteneği incelenecektir. Genelleme yeteneğini incelemek amacıyla da, sinüzoidal giriş işaretine göre eğitilen kontrol ağının daha değişik giriş işaretlerine karşılık sistemi kontrol


BAŞLA
BAŞLANGIÇ AĞIRLIKLARINI OLUŞTUR
AĞIRLIK DOSYASINI AÇ
REFERANS GİRİŞİNİ HESAPLA
SİSTEM PARAMETRELERİNİ GİR
SİSTEMİN FARK DENKLEMİNİ ELDE ET
BAŞLANGIÇ ŞARTLARINI SIFIRLA
AĞ ÇIKIŞINI HESAPLA
SİSTEM ÇIKIŞINI HESAPLA
ÇIKIŞ HATASINI HESAPLA
SİSTEMDEN YANSITILMIŞ HATAYI HESAPLA
AĞIN GRADYANINI HESAPLA
EVET
<K
HAYIR
EVET
<=HATA
HAYIR
AĞIRLIKLARI AYARLA
2



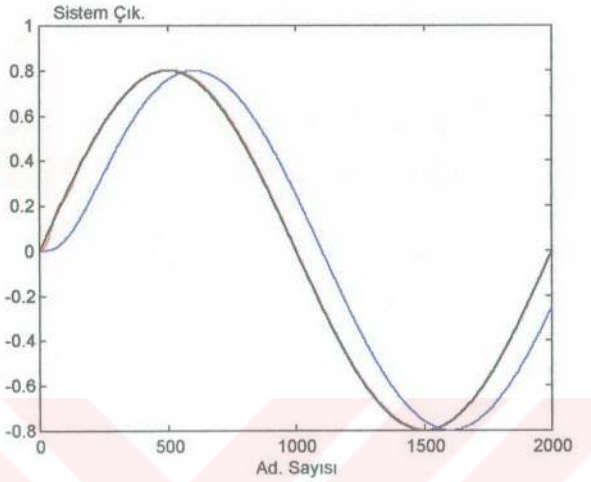
Şekil 4.4 Toplu öğrenme ile doğrudan kontrol yapısı programının (TTK1) blok şeması

edebilme yeteneği incelenecek ve bu amaçla da referans giriş işareti olarak basamak giriş işareti kullanılacaktır.

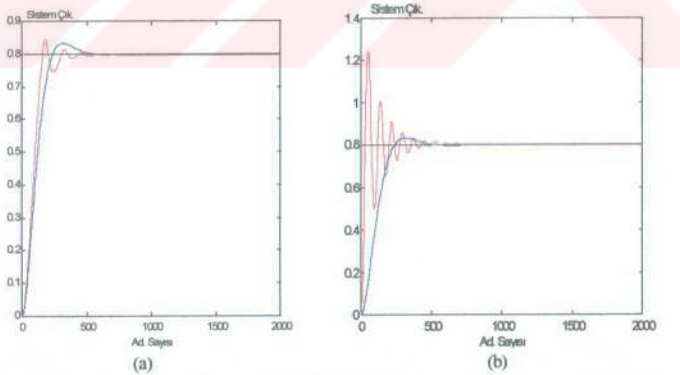
Bu çalışmada, basamak giriş işaretine kontrol edilmiş sistemin verdiği cevap, kontrol ağının çıkışında hareketlendirme fonksiyonu olarak tanh(.) ve doğrusal f(x)=x fonksiyonu olmak üzere iki durum için incelenmiştir. eğitilen sistemin eğitim sonunda basamak referans giriş işaretine verdiği cevap görülmektedir. 4.6(a)'da ki şekilde kontrol ağının çıkışında hareketlendirme fonksiyonu olarak doğrusal f(x)=x fonksiyonu 4.6(b)'deki şekilde ise kontrol ağının çıkışında tanh(.) fonksiyonu kullanılmıştır.

Kontrol edilen sistemin parametreleri değiştirilerek, yeni sistem,

$$G_{(s)} = \frac{6}{S^2+2S+6} \qquad (4.32)$$

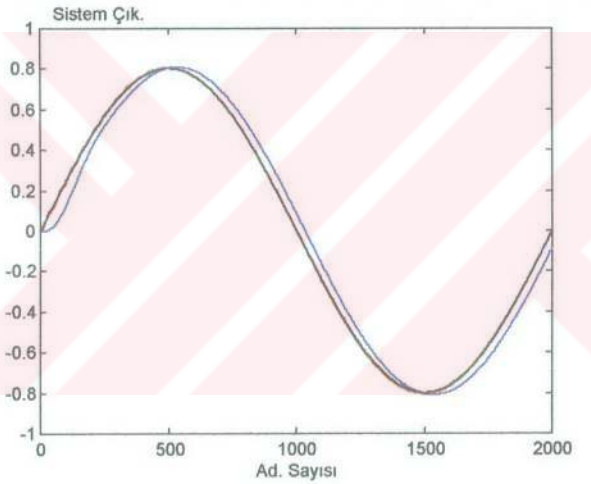


Şekil 4.5 Denklem 4.28'deki sistemin sinüzoidal referans giriş işaretine verdiği cevap. referans giriş, kontrol edilmemis sistem cıkısı. kontrol edilmis sistem cıkısı

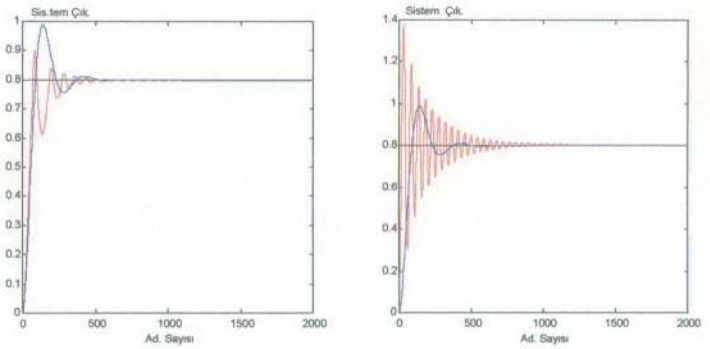


Şekil 4.6 Yapay sinir ağının 4:28'deki denklemi genelleme yeteneği (a)Çıkış katmanındaki hareketlendirme fonksiyonu tanh(),(b)Çıkış katmanında hareketlendirme fonksiyonu f(x)=x iken basamak giriş işaretine sistemin verdiği cevap. referans giriş, kontrol edilmemiş sistem çıkışı, kontrol edilmiş sistem çıkışı

olarak alındığında, sistemin sinüzoidal referans giriş işaretine vermiş olduğu cevap şekil 4.7'de görülmektedir. Bu şekilden de görüldüğü gibi sistemin parametrelerindeki değişikliğe rağmen, kontrol edilmiş sistem çıkışı referans giriş işaretini başarılı bir şekilde izlemektedir. Şekil 4.8'de ise sistemin basamak giriş işaretine vermiş olduğu cevap görülmektedir.



Şekil 4.7 Denklem 4.32'deki sistemin sinüzoidal referans giriş işaretine verdiği cevap._____referans giriş,_____ kontrol edilmemiş sistem çıkışı,_____kontrol edilmiş sistem çıkışı

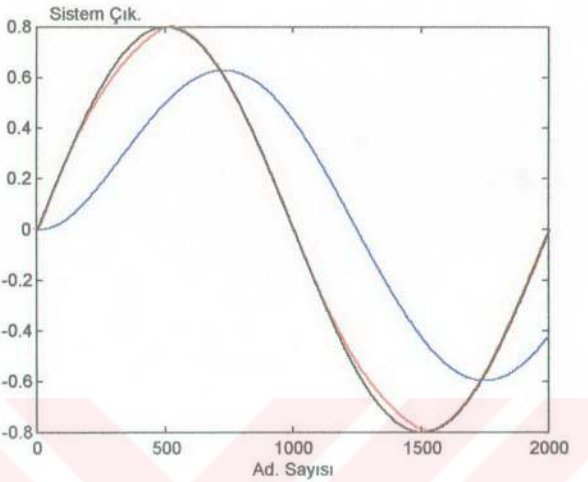


Şekil 4.8 Yapay sinir ağının 4.32'deki denklemi genelleme yeteneği (a)Çıkış katmanındaki hareketlendirme fonksiyonu tanh(),(b)Çıkış katmanında hareketlendirme fonksiyonu f(x)=x iken basamak giriş işaretine sistemin verdiği cevap. ——— referans giriş, ——— kontrol edilmemiş sistem çıkışı, ——— kontrol edilmiş sistem çıkışı

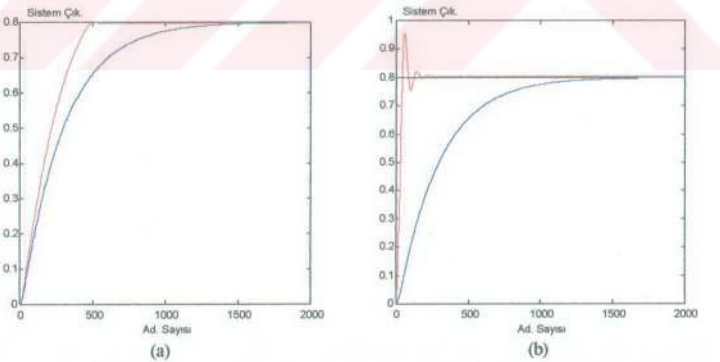
Kontrol edilecek sistemin parametreleri yine değiştirilerek yeni sistem,

$$G_{(s)} = \frac{10}{5S^2+6S+10} \qquad (4.33)$$

olarak seçildiğinde, kontrol edilmiş sistemin sinüzoidal referans giriş işaretine verdiği cevap şekil 4.9'da, basamak giriş işaretine verdiği cevap ise şekil 4.10'da görülmektedir. Bu şekillerden de görüldüğü gibi, doğrudan kontrol yapısında toplu öğrenme algoritmasıyla eğitilen yapay sinir ağı, sistemdeki değişiklikleri başarı ile tanıyabilmekte ve eğitim haricindeki giriş işaretlerine karşı vermiş olduğu cevaplarda da belirli bir iyileşme olduğundan genelleme yeteneği de belirli ölçülerde sağlanabilmiştir. Kontrol ağının genellemede en önemli problemi, çıkış katmanındaki hareketlendirme fonksiyonunun



Şekil 4.9 Denklem 4.33'deki sistemin sinüzoidal referans giriş işaretine verdiği cevap.——referans giriş,——kontrol edilmemiş sistem çıkışı,——kontrol edilmiş sistem çıkışı



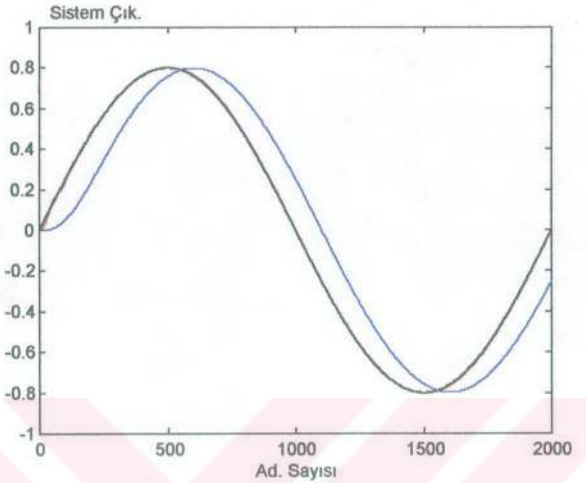
Şekil 4.10 Yapay sinir ağının 4.33'deki denklemi genelleme yeteneği (a)Çıkış katmanındaki hareketlendirme fonksiyonu tanh(),(b)Çıkış katmanında hareketlendirme fonksiyonu f(x)=x iken basamak giriş işaretine sistemin verdiği cevap.——referans giriş,——kontrol edilmemiş sistem çıkışı,——kontrol edilmiş sistem çıkışı

doğrusal olduğu durumlarda salınımları yeterince engelleyememesidir.

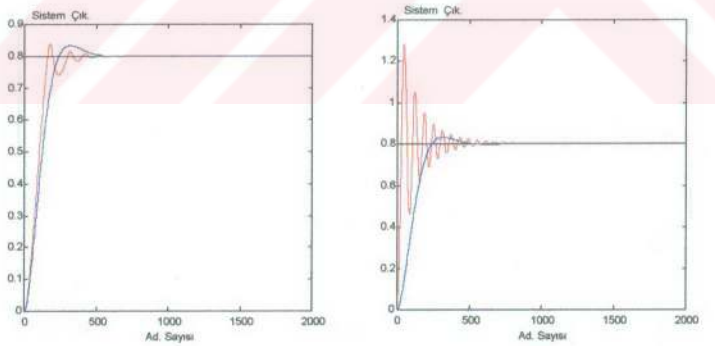
Toplu öğrenmede, eldeki belirli miktardaki veri toplu olarak eğitildiğinden gerçek zamanlı kontrol yapıları için uygun değildir. Gerçek zamanlı kontrol için; geriye yayınım algoritmasında her bir örneğin teker teker işlendiği örneksel öğrenme kullanılarak TTK2 adlı bir program hazırlanmıştır. Bu programda, yine doğrudan kontrol yapısı içerisinde, ağ ağırlıklarının uyarlanmasında geriye yayınım algoritması kullanılmıştır. Toplu öğrenmeden tek farkı örneklerin tek tek işlenerek ağırlıkların her bir örnekte yeniden uyarlanmasıdır.

Hazırlanan programda kontrol edilecek sistem olarak yine denklem 4.28'deki sistem seçilmiştir. Bu sistemi kontrol etmek amacıyla kontrolör olarak kullanılan yapay sinir ağı 50 dakikalık bir süre ile eğitilmiş ve bu süre sonunda sistem çıkışı kontrol edilerek, sistem çıkışının referans giriş işaretini yeterince izlediği kabul edilip eğitim kesilmiş ve elde edilen bu ağırlık değerleri bir dosyada saklanarak programdaki eğitim bölümleri kardırılmış ve kontrol ağı bu ağırlık değerleriyle sabitlenmiştir. Eğitim kesildikten sonra sistemin sinüzoidal referans giriş işaretine verdiği cevap şekil 4.11'de, basamak giriş işaretine vermiş olduğu cevap ise şekil 4.12' de verilmiştir. Bu aşamadan sonra, yine yapay sinir ağının öğrenme ve genelleme yeteneklerinin incelenmesi amacıyla sistem parametrelerinde rasgele değişiklikler yapılacak ve sistem girişine değişik giriş işaretleri uygulanacaktır. Bu amaçla sistem parametreleri,

$$G_{(s)} = \frac{5}{S^2 + 3S + 5} \qquad (4.34)$$



Şekil 4.11 Denklem 4.28'deki sistemin sinüzoidal referans giriş işaretine verdiği cevap.——referans giriş,——kontrol edilmemis sistem cıkısı.——kontrol edilmis sistem cıkısı



(a) (b)

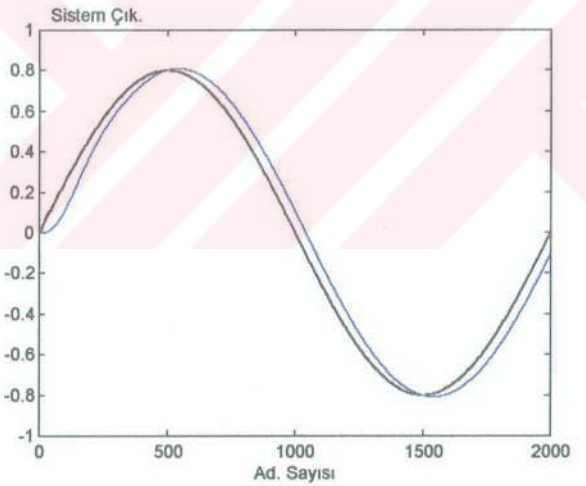
Şekil 4.12 Yapay sinir ağının 4.28'deki denklemi genelleme yeteneği (a)Çıkış katmanındaki hareketlendirme fonksiyonu tanh(),(b)Çıkış katmanında hareketlendirme fonksiyonu f(x)=x iken basamak giriş işaretine sistemin verdiği cevap.——referans giriş,——kontrol edilmemiş sistem çıkışı,——kontrol edilmiş sistem çıkışı

olarak değiştirildiğinde yeni sistemin sinüzoidal giriş işaretine verdiği cevap şekil 4.13'te, basamak giriş işaretine vermiş olduğu cevap ise şekil 4.14'te görülmektedir.

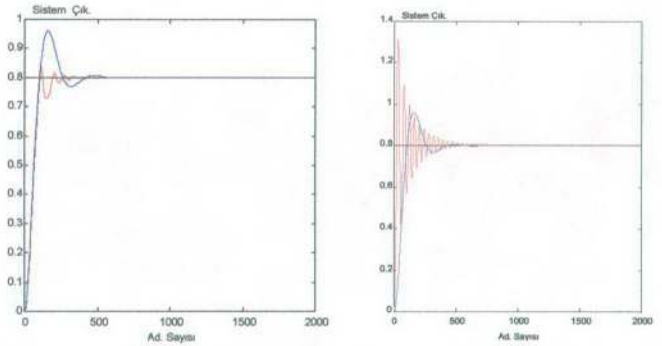
Kontrol edilecek olan sistemin parametreleri,

$$G_{(s)} = \frac{10}{S^2+8S+10} \qquad (4.35)$$

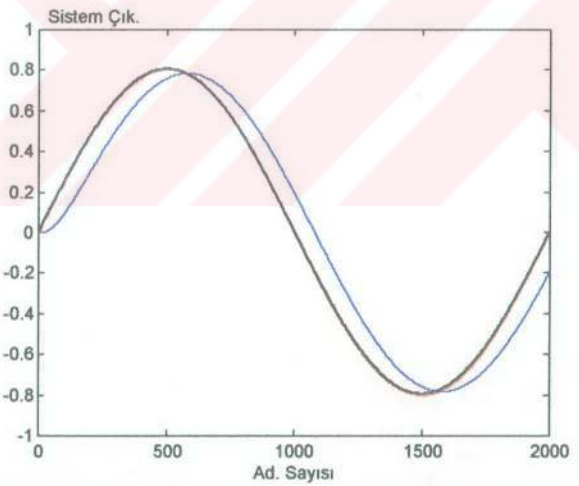
olarak değiştirildiğinde kontrol edilmiş sistemin,referans giriş işareti olarak sinüzoidal işarete verdiği cevap şekil 4.15'te, basamak giriş işaretine vermiş olduğu cevap ise şekil 4.16'da görülmektedir.



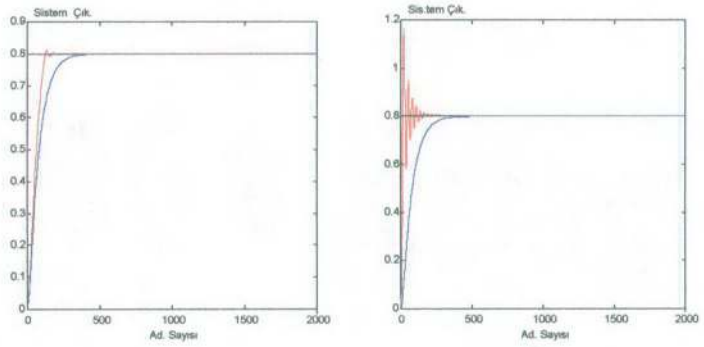
Şekil 4.13 Denklem 4.34'deki sistemin sinüzoidal referans giriş işaretine verdiği cevap.——referans giriş,——kontrol edilmemiş sistem çıkışı,——kontrol edilmiş sistem çıkışı



Şekil 4.14 Yapay sinir ağının 4.34'teki denklemi genelleme yeteneği (a)Çıkış katmanındaki hareketlendirme fonksiyonu tanh(),(b)Çıkış katmanında hareketlendirme fonksiyonu f(x)=x iken basamak giriş işaretine sistemin verdiği cevap. referans giriş, kontrol edilmemiş sistem çıkışı, kontrol edilmiş sistem çıkışı



Şekil 4.15 Denklem 4.35'deki sistemin sinüzoidal referans giriş işaretine verdiği cevap. referans giriş, kontrol edilmemis sistem cıkısı. kontrol edilmis sistem cıkısı



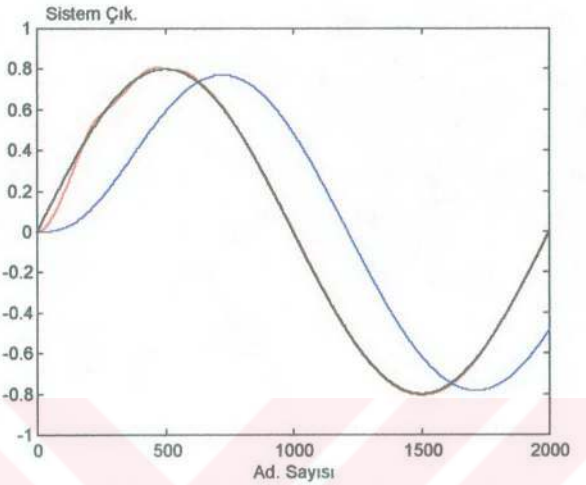
(a) (b)

Şekil 4.16 Yapay sinir ağının 4.35'deki denklemi genelleme yeteneği (a)Çıkış katmanındaki hareketlendirme fonksiyonu tanh(),(b)Çıkış katmanında hareketlendirme fonksiyonu f(x)=x iken basamak giriş işaretine sistemin verdiği cevap. ——— referans giriş, ——— kontrol edilmemiş sistem çıkışı, ——— kontrol edilmiş sistem çıkışı

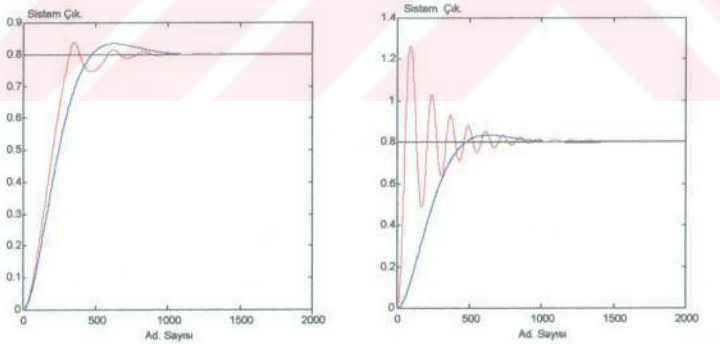
Sistem parametreleri tekrar değiştirilerek,

$$G_{(s)} = \frac{5}{10S^2+10S+5} \qquad (4.36)$$

olarak sistem tekrar değiştirildiğinde yeni sistemin sinüzoidal referans giriş işaretine vermiş olduğu cevap şekil 4.17'de, basamak giriş işaretine vermiş olduğu cevap ise şekil 4.18'de görülmektedir.



Şekil 4.17 Denklem 4.36'daki sistemin sinüzoidal referans giriş işaretine verdiği cevap. ——— referans giriş, ——— kontrol edilmemiş sistem çıkışı, ——— kontrol edilmiş sistem çıkışı

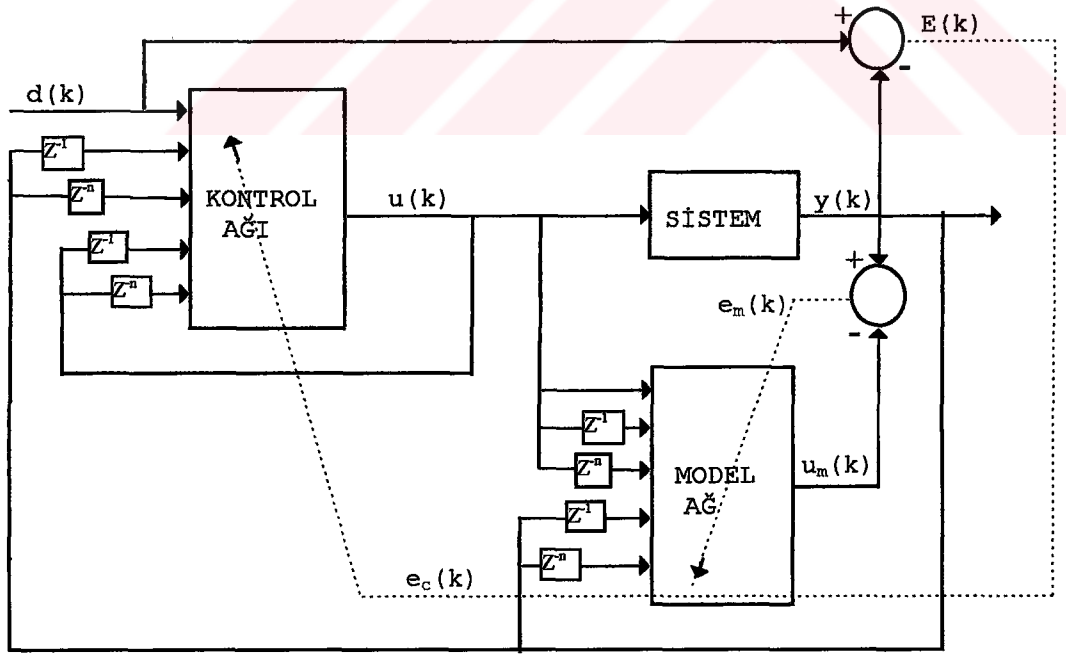


(a) (b)

Şekil 4.18 Yapay sinir ağının 4.36'daki denklemi genelleme yeteneği (a)Çıkış katmanındaki hareketlendirme fonksiyonu tanh(), (b)Çıkış katmanında hareketlendirme fonksiyonu f(x)=x iken basamak giriş işaretine sistemin verdiği cevap. ——— referans giriş, ——— kontrol edilmemiş sistem çıkışı, ——— kontrol edilmiş sistem çıkışı

4.3 Dolaylı Kontrol Yöntemi İle Sistem Kontrolü

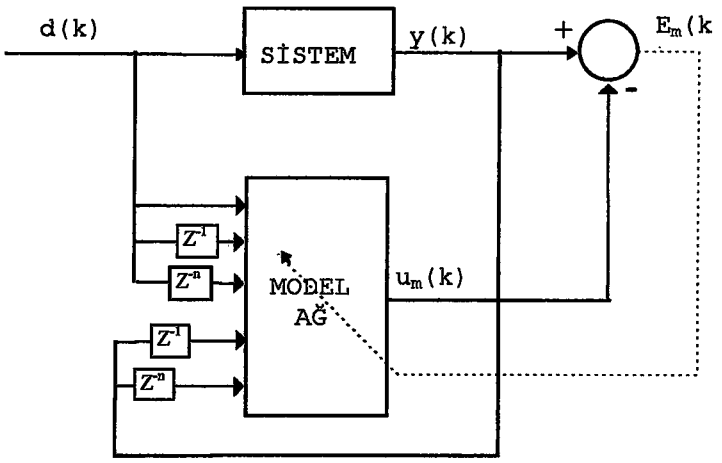
Dolaylı kontrol yapısı prensip olarak doğrudan kontrol yapısı ile benzer yapıdadır. Dolaylı kontrol yapısında, doğrudan kontrol yapısına ek olarak sistemin kendisi gibi davranacak şekilde eğitilmiş bir model yapay sinir ağı kullanılmıştır. Uygulamadaki temel fark; doğrudan kontrol yönteminde kontrol ağının çıkışı, hem sistem girişine hem de zaman gecikmeleriyle birlikte sistemi tanıması amacıyla yerleştirilen model ağa sunulmaktadır. Böylece dolaylı kontrol yönteminde, biri kontrolör gibi davranacak, diğeri ise sistem gibi davranacak şekilde iki adet yapay sinir ağı kullanılmaktadır. Şekil 4.19'da dolaylı kontrol yapısının blok diyagramı görülmektedir.



Şekil 4.19 Dolaylı Kontrol yöntemi Blok Şeması

Kontrolör olarak kullanılan yapay sinir ağının oluşturulması, doğrudan kontrol yönteminde olduğu gibi sistemin derecesi göz önüne alınarak, ağ girişleri olarak sistem çıkışlarının ve kontrol ağının gecikmiş zamanları seçilir. Sistem gibi davranması istenen model ağının girişleri ise, kontrol ağının çıkışları ve gecikmiş zamanlarıyla birlikte sistem çıkışlarının gecikmiş zamanlarının uygulanmasıyla oluşturulur. Her iki ağında saklı katmanlarının oluşturulması için izlenmesi gereken yol, bölüm 4.2'de açıklandığı şekildedir.

Dolaylı kontrol yönteminin uygulanmasında öncelikle sistemin düz yön modeli elde edilerek yapay sinir ağıyla oluşturulur ve kontrol ağından bağımsız olarak eğitilir. Bu şekilde bir eğitim için gerekli olan blok şeması şekil 4.20'de görülmektedir. Bu şekilden de görüldüğü gibi model ağın oluşturulması için, sistem giriş ve çıkışlarının gecikmiş zamanlarına ihtiyaç vardır. Eğitim işlemi kontrol ağından bağımsız olacağı için, model ağının ve sistemin girişi olarak kontrol ağının çıkışı yerine arzu edilen



Şekil 4.20 Model ağı blok şeması

çıkış işareti uygulanarak yapılır. Bu şekilde bir eğitimde sistem, arzu edilen giriş değerine karşılık bir çıkış verir ve model ağı kendi çıkışıyla, sistem çıkışı arasındaki hatadan faydalanılarak model ağının, sistemi tanıması sağlanır. Daha sonra sistemi tanıyan ağ, dolaylı kontrol yapısına yerleştirilerek sistem çıkış değerleri istenilen seviyeye gelinceye kadar kontrol ağı eğitilir.

4.3.1 Kontrolör ve model yapay sinir ağlarının eğitimi ve eğitim algoritması

Dolaylı kontrol yöntemi, kontrol edilmek istenen sistemin ters ve düz modellerinden oluşturulan iki tane çok katlı yapay sinir ağından meydana gelir. Sistemin düz yön modelinden elde edilen yapay sinir ağı model ağ, ters yön modelinden elde edilen yapay sinir ağı ise kontrol ağı olarak adlandırılmıştır. Sistemin ters ve düz modellerini elde etmek için, eğer sistem denklemi mevcut ise sistem denkleminden elde edilir. Şayet bilinmeyen bir sistem ise sistemin derecesinin bilinmesi veya tahmin edilerek ağ girişlerinin, tahmin edilen bu dereceye göre oluşturulması gereklidir.

Eğitim şekli olarak doğrudan kontrol yöntemi ile oldukça benzer yapıda olan dolaylı kontrol yönteminin eğitim şekli; model ağın sistemi tanıyarak sistem gibi davranmasını sağlamak ve kontrol ağının eğitimi için gerekli olan , sistem üzerinden geriye yansıtılmış hata işaretinin sistem gibi davranan model ağ üzerinden yansıtılarak elde edilmesi temeline dayanır.

Kontrolör ve model yapay sinir ağının eğitimde geriye yayınım öğrenme algoritması kullanılmaktadır. Model eğitimi için, sistem çıkışı ile model çıkışı arasındaki hata işareti, $e_m=y-y_m$, kontrol ağının eğitimi içinse şekil 4.20'de belirlenen arzu edilen çıkış ile sistem çıkışı arasında oluşan hata işareti e=d-y kullanılır.

Model üzerinden yansıtılarak kontrol ağının girişine uygulanan e_c hatası dolaylı kontrol yönteminde,

$$e_C=e\frac{\partial y_m}{\partial u} \qquad 4.30$$

şeklinde olur. Burada $\frac{\partial y_m}{\partial u}$ kısmi türevi;

$$\frac{\partial y_m}{\partial u}=\frac{\partial y_m}{\partial v}\frac{\partial v}{\partial net}\frac{\partial net}{\partial u} \qquad (4.31)$$

$$\frac{\partial y_m}{\partial v}=w_{ij}^2 \qquad (4.32)$$

$$\frac{\partial v}{\partial net}=\Phi'(net) \qquad (4.33)$$

$$\frac{\partial net}{\partial u}=w_{iv}^1 \qquad (4.34)$$

olmak üzere

$$\frac{\partial y_m}{\partial u}=w_{ij}^2 \tanh'(net) w_{iv}^1 \qquad (4.35)$$

olarak bulunabilir. Burada w_{iv}^1, u(k) girişini saklı katmandaki hücrelere bağlayan ağırlık değerleridir.(Chan 1993) Kontrol ağının eğitiminde kullanılacak olan, model sistem üzerinden geriye yansıtılmış hata işareti $e_C(k)$ ise,

$$e_C(k)=e_m(k)w_{ij}^2\Phi'(net)w_{iv}^1 \qquad (4.36)$$

olarak hesaplanır.

4.3.2 Seçilen bir sistem için düz yön ve ters yön yapay sinir ağının seçimi ve simülasyon sonuçları

Dolaylı kontrol yöntemiyle bir sistemin kontrol edilebilmesi için öncelikle sistem denkleminden faydalanılarak, sistemin ters ve düz yön modelinin belirlenmesi gereklidir. Bu amaçla sistemin düz yön modeli

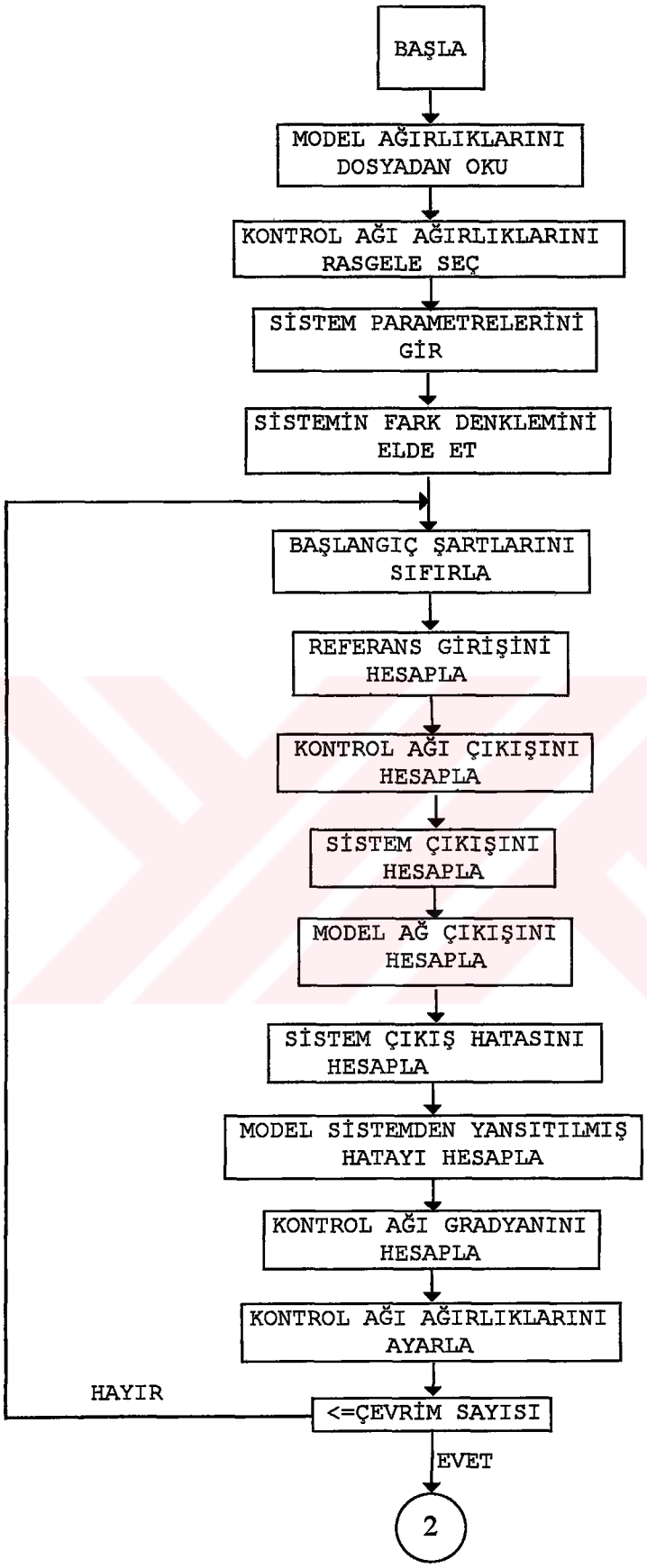
$$y(k)=f(u(k-1),u(k-2)...,y(k),Y(k-1),...) \qquad (4.37)$$

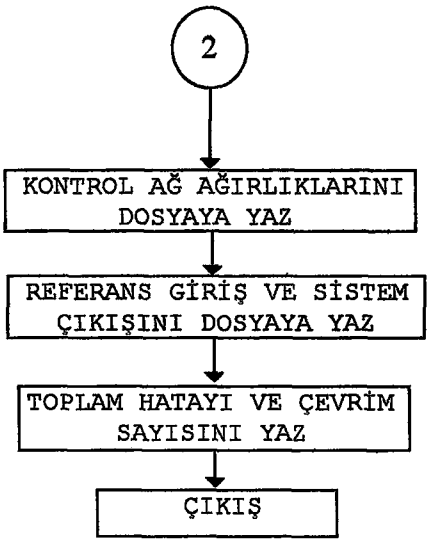
ve ters yön modeli

$$u(k)=F(u(k),U(k-1)...,y(k-1),y(k-2)...) \qquad (4.38)$$

şeklinde elde edilir. Elde edilen düz yön modeli ile model yapay sinir ağı, ters yön modeli ile de kontrolör yapay sinir ağı oluşturulur. Oluşturulan her iki ağında; giriş katmanı, saklı katmanı ve çıkış katmanındaki hücre sayısı, bölüm 4.2.2 de anlatıldığı şekilde belirlenir.

Bölüm 4.2.2 de kontrol edilmek üzere seçilerek kontrol ağının eğitildiği 4.28 denklemindeki sistem, doğrudan ve dolaylı kontrol yapılarını kıyaslama açısından dolaylı kontrol yapısında da, model ağın ve kontrol ağının eğitiminde kullanılacaktır. Bu sisteme göre yapay sinir ağıyla, kontrol ağı ve model ağ oluşturularak; sistemi tanıyacak şekilde oluşturulan model ağ için Tc++ programlama dilinde simülasyon için, toplu öğrenmede MODEL1 örneksel öğrenmede MODEL2 programları, sistemi kontrol etmek için ise, toplu öğrenmede NMTD1 ve örneksel öğrenmede NMTD2 adlı programlar hazırlanmıştır.


BAŞLA
MODEL AĞIRLIKLARINI DOSYADAN OKU
KONTROL AĞI AĞIRLIKLARINI RASGELE SEÇ
SİSTEM PARAMETRELERİNİ GİR
SİSTEMİN FARK DENKLEMİNİ ELDE ET
BAŞLANGIÇ ŞARTLARINI SIFIRLA
REFERANS GİRİŞİNİ HESAPLA
KONTROL AĞI ÇIKIŞINI HESAPLA
SİSTEM ÇIKIŞINI HESAPLA
MODEL AĞ ÇIKIŞINI HESAPLA
SİSTEM ÇIKIŞ HATASINI HESAPLA
MODEL SİSTEMDEN YANSITILMIŞ HATAYI HESAPLA
KONTROL AĞI GRADYANINI HESAPLA
KONTROL AĞI AĞIRLIKLARINI AYARLA
HAYIR
<=ÇEVRİM SAYISI
EVET
2



Şekil 4.21 Dolaylı kontrol yöntemi ile örneksel öğrenme programının (NMTD2) akış şeması

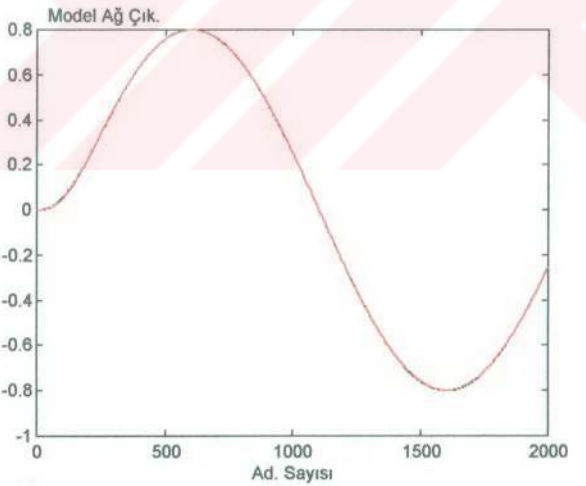
Şekil 4.21'de dolaylı kontrol yönteminde örneksel öğrenme için hazırlanan NMTD2 programının akış şeması görülmektedir. Bu programın ana programı ve ana programda kullanılan alt programları EK2'de verilmiştir.

Toplu öğrenme ile hazırlanan MODEL1 programı sinüsoidal giriş işaretine göre 0.01 hata ile eğitilerek model ağın, giriş katmanındaki ve çıkış katmanındaki ağırlık değerleri bir dosyada saklanmıştır. Bu ağırlık değerleriyle model ağın çıkışı şekil 4.22'de görülmektedir. Bu şekilden de görüldüğü gibi model ağ, sistemi başarılı bir şekilde tanımaktadır.

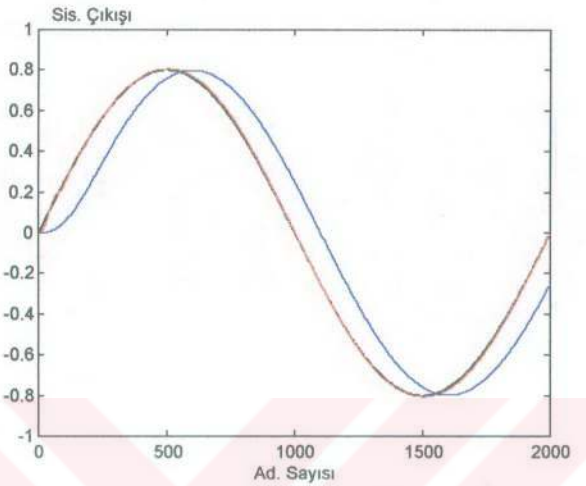
Kontrol ağı da, önceden eğitilerek saklanan model ağ ağırlıkları kullanılarak,sistem modeli üzerinden geriye yansıtılan hata işareti ile sinüzoidal giriş işaretine göre 0.21 hata ile eğitilerek giriş katmanı ve saklı katman ağırlıkları bir dosyada saklanmıştır. Bu ağırlık değerleri

için denklem 4.28'deki sistemin kontrol edilmiş çıkışı şekil 4.23 'te görülmektedir. Şekil 4.24'de ise kontrol ağının genelleme yeteneğini belirlemek üzere sinüzoidal giriş işaretine göre kontrol edilen sistemin basamak cevabı görülmektedir. Bu bölümde de, önceki bölümde olduğu gibi birbirine alternatif olması bakımından, kontrol ağının çıkışında hareketlendirme fonksiyonu olarak tanh(.) ve doğrusal f(x)=x fonksiyonları varken ki durumlar ayrı ayrı verilecektir.

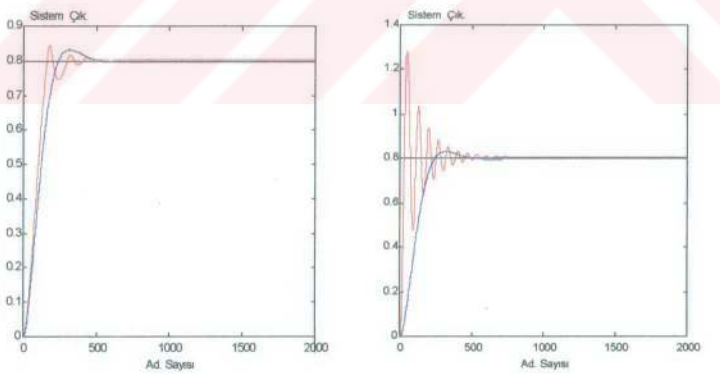
Öğrenme yeteneğini belirlemek amacıyla sistem parametrelerinde rasgele değişiklikler yapılacak, genelleme yeteneğini belirlemek amacıyla da sinüzoidal giriş işareti yerine basamak işareti uygulanacaktır.



Şekil 4.22 Toplu öğrenme ile model ağın sistemi tanıması
—— Sistem çıkışı, —— model ağ çıkışı



Şekil 4.23 Denklem 4.28'deki sistemin sinüzoidal referans giriş işaretine verdiği cevap.——referans giriş,—— kontrol edilmemis sistem cıkısı.——kontrol edilmis sistem cıkısı



Şekil 4.24 Yapay sinir ağının 4.28'teki denklemi genelleme yeteneği (a)Çıkış katmanındaki hareketlendirme fonksiyonu tanh(),(b)Çıkış katmanında hareketlendirme fonksiyonu f(x)=x iken basamak giriş işaretine sistemin verdiği cevap.——referans giriş,—— kontrol edilmemiş sistem çıkışı,——kontrol edilmiş sistem çıkışı

Bu amaçla 4.28 denklemi,

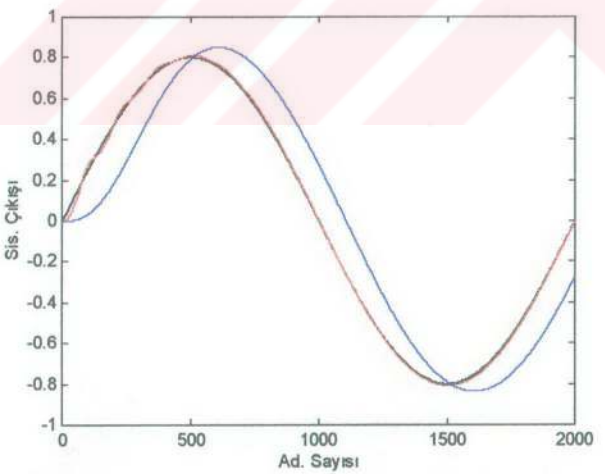
$$G_{(s)} = \frac{1}{S^2+S+1} \qquad (4.39)$$

şeklinde değiştirildiğinde, kontrol edilmiş sistemin sinüzoidal referans giriş işaretine verdiği cevap şekil 4.25'de basamak giriş işaretine verdiği cevap ise şekil 4.26'da görülmektedir.

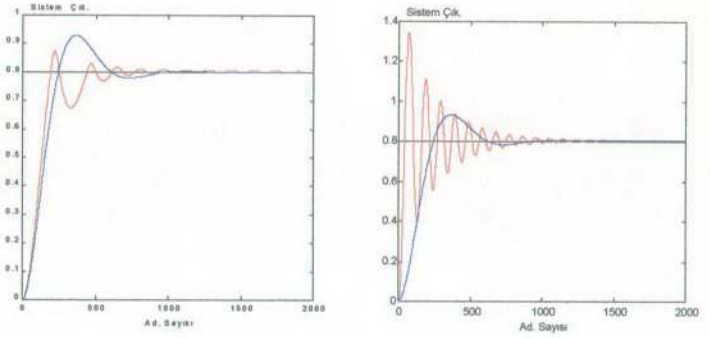
Sistem parametreleri tekrar,

$$G_{(s)} = \frac{4}{S^2+8S+4} \qquad (4.40)$$

olarak değiştirildiğinde, kontrol edilmiş sistemin sinüzoidal referans giriş işaretine verdiği cevap şekil 4.27'de basamak giriş işaretine verdiği cevap ise şekil 4.28'da görülmektedir.

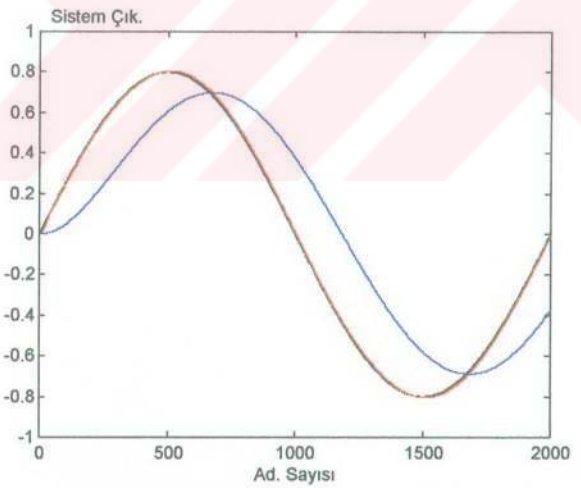


Şekil 4.25 Denklem 4.39'daki sistemin sinüzoidal referans giriş işaretine verdiği cevap._____referans giriş,_____ kontrol edilmemiş sistem çıkışı,_____kontrol edilmiş sistem çıkışı

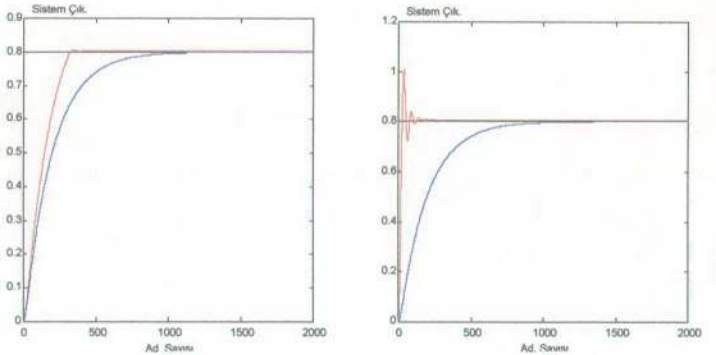


(a) (b)

Şekil 4.26 Yapay sinir ağının 4.39'teki denklemi genelleme yeteneği (a)Çıkış katmanındaki hareketlendirme fonksiyonu tanh(),(b)Çıkış katmanında hareketlendirme fonksiyonu f(x)=x iken basamak giriş işaretine sistemin verdiği cevap. referans giriş, kontrol edilmemiş sistem çıkışı, kontrol edilmiş sistem çıkışı



Şekil 4.27 Denklem 4.40'daki sistemin sinüzoidal referans giriş işaretine verdiği cevap. referans giriş, kontrol edilmemiş sistem çıkışı, kontrol edilmiş sistem çıkışı

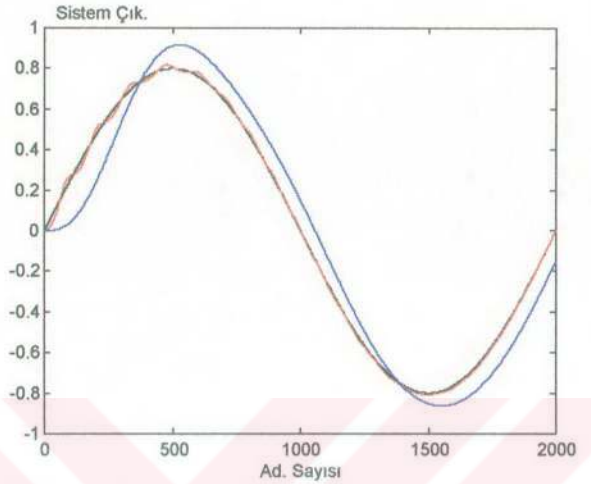


Şekil 4.28 Yapay sinir ağının 4.40'daki denklemi genelleme yeteneği (a)Çıkış katmanındaki hareketlendirme fonksiyonu tanh(),(b)Çıkış katmanında hareketlendirme fonksiyonu f(x)=x iken basamak giriş işaretine sistemin verdiği cevap. ——— referans giriş, ——— kontrol edilmemiş sistem çıkışı, ——— kontrol edilmiş sistem çıkışı

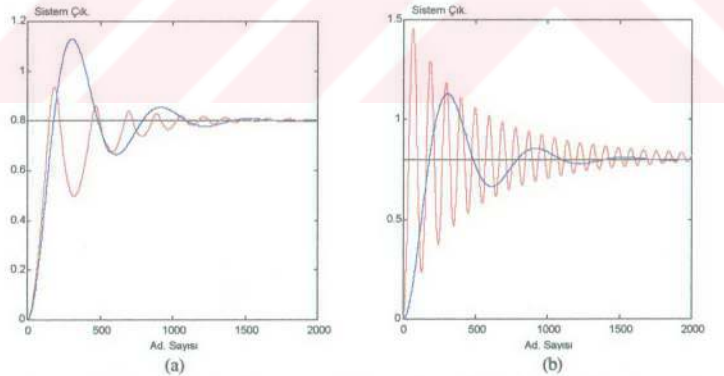
Sistem parametreleri yeniden,

$$G_{(s)} = \frac{8}{7s^2 + 4s + 8} \qquad (4.41)$$

olarak değiştirildiğinde, kontrol edilmiş sistemin sinüzoidal referans giriş işaretine verdiği cevap şekil 4.29'da basamak giriş işaretine verdiği cevap ise şekil 4.30'da görülmektedir.



Şekil 4.29 Denklem 4.41'daki sistemin sinüzoidal referans giriş işaretine verdiği cevap._____referans giriş,_____ kontrol edilmemiş sistem çıkışı,_____kontrol edilmiş sistem çıkışı

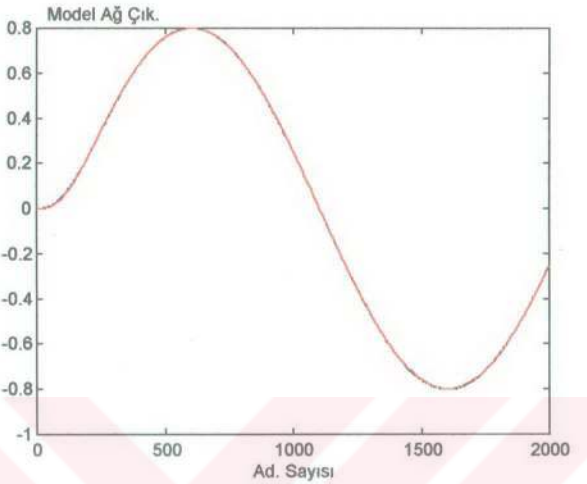


Şekil 4.30 Yapay sinir ağının 4.41'daki denklemi genelleme yeteneği (a)Çıkış katmanındaki hareketlendirme fonksiyonu tanh(),(b)Çıkış katmanında hareketlendirme fonksiyonu f(x)=x iken basamak giriş işaretine sistemin verdiği cevap._____referans giriş,_____kontrol edilmemiş sistem çıkışı,_____kontrol edilmiş sistem çıkışı

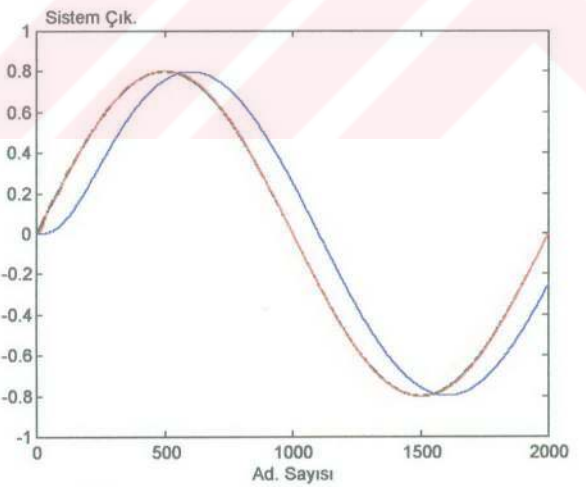
Örneksel öğrenme ile sistemi tanıması için hazırlanan MODEL2 programı sinüzoidal giriş işaretine göre eğitilerek, model ağın denklem 4.28'deki sistemi 10 dakikalık eğitimden sonra yeterince tanıdığı kabul edilerek 0.02 hata ile eğitim kesilmiş ve bu hata değerindeki model ağının giriş ve saklı katmanındaki ağırlık değerleri bir dosyada saklanmıştır.Elde edilen bu model ağırlıkları ile model ağın çıkışı şekil 4.31'de görülmektedir.

Model ağ, bu şekilde kontrol edilecek sistemi tanıması için eğitildikten sonra, 4.28 denklemindeki sistemi kontrol edebilmek için hazırlanan NMTD2 programında, model sistem üzerinden yansıtılarak kontrol ağının eğitiminde kullanılacak olan eğitim işaretinin elde edilebilmesinde model ağın bu ağırlık değerleri kullanılarak, kontrol ağı sinüzoidal referans giriş işaretine göre eğitilmiş ve 50 dakikalık bir eğitimden sonra 0.24 hata ile eğitim kesilmiş ve bu hata değerindeki kontrol ağının giriş ve saklı katman ağırlıkları bir dosyada saklanmıştır. Bu ağırlık değerlerinden elde edilen sinüsoidal referans giriş işaretine göre kontrol edilmiş sistem çıkışı şekil 4.32'de görülmektedir.

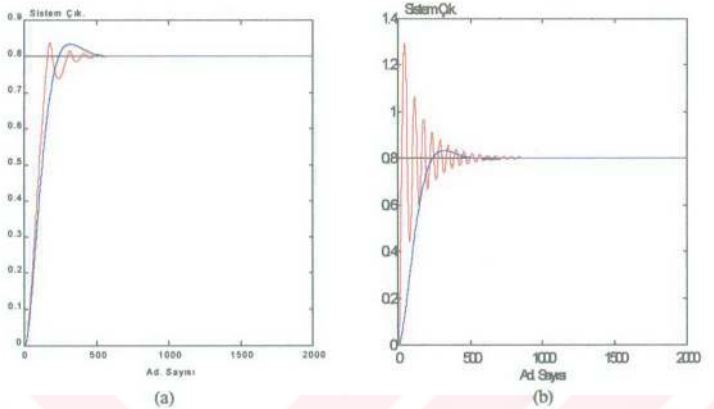
Bu aşamadan sonra, daha önce yapıldığı şekilde eğitilen kontrol ağının öğrenme yeteneğinin belirlenmesi amacı ile sistem parametrelerinde rasgele değişiklikler yapılacak ve kontrol ağının genelleme yeteneğinin belirlenmesi için referans giriş işareti basamak giriş olarak değiştirilecektir. Sistemin basamak cevabı yine, kontrol ağı çıkışında hareketlendirme fonksiyonu olarak hem tanh(.) fonksiyonu hemde doğrusal f(x)=x fonksiyonunun olduğu durumda alınan sistem çıkışları şekil 4.33'te ayrı ayrı verilmiştir.



Şekil 4.31 Örneksel öğrenme ile model ağın sistemi tanıması ——Sistem çıkışı, ——model ağ çıkışı



Şekil 4.32 Denklem 4.28'deki sistemin sinüzoidal referans giriş işaretine verdiği cevap._____referans giriş,_____ kontrol edilmemiş sistem çıkışı,_____kontrol edilmiş sistem çıkışı



(a) (b)

Şekil 4.33 Yapay sinir ağının 4.28'deki denklemi genelleme yeteneği (a)Çıkış katmanındaki hareketlendirme fonksiyonu tanh(),(b)Çıkış katmanında hareketlendirme fonksiyonu f(x)=x iken basamak giriş işaretine sistemin verdiği cevap. ——— referans giriş, ——— kontrol edilmemiş sistem çıkışı, ——— kontrol edilmiş sistem çıkışı

Denklem 4.28'deki sistem,

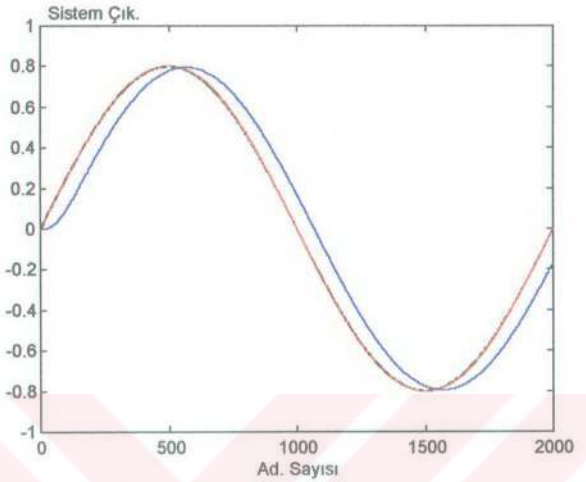
$$G_{(s)} = \frac{6}{S^2+4S+6} \qquad (4.42)$$

olarak değiştirilmiş ve ağırlık değerleri daha önceden saklanan kontrol ağının, yeni sistemi sinüzoidal referans giriş işaretine göre kontrolü şekil 4.34'de, basamak giriş işaretini kontrolü ise şekil 4.35'de verilmiştir.

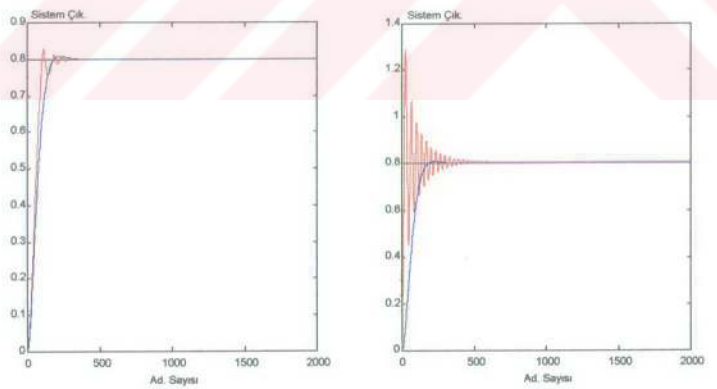
Sistem parametreleri tekrar değiştirilerek yeni sistem

$$G_{(s)} = \frac{3}{S^2+5S+3} \qquad (4.43)$$

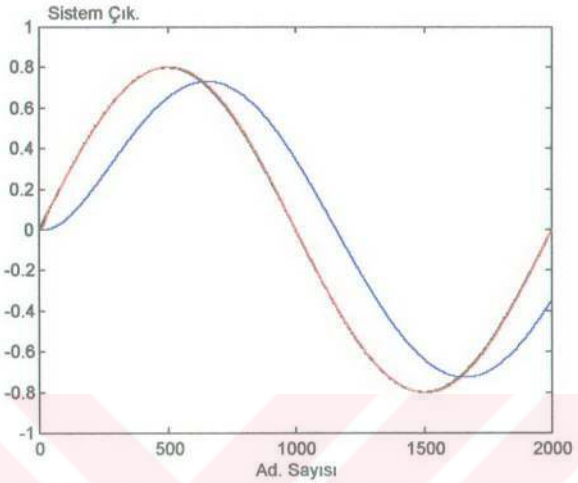
olarak alındığında, yeni sistemin sinüzoidal referans giriş işaretine vermiş olduğu cevap şekil 4.36'da, basamak giriş işaretine vermiş olduğu cevap ise şekil 4.37'de verilmiştir.



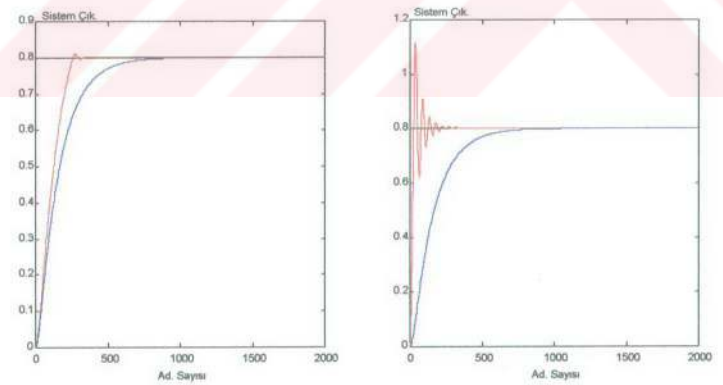
Şekil 4.34 Denklem 4.42'deki sistemin sinüzoidal referans giriş işaretine verdiği cevap._____ referans giriş,_____ kontrol edilmemiş sistem çıkışı,_____kontrol edilmiş sistem çıkışı



Şekil 4.35 Yapay sinir ağının 4.42'deki denklemi genelleme yeteneği (a)Çıkış katmanındaki hareketlendirme fonksiyonu tanh(), (b)Çıkış katmanında hareketlendirme fonksiyonu f(x)=x iken basamak giriş işaretine sistemin verdiği cevap._____ referans giriş,_____ kontrol edilmemiş sistem çıkışı,_____ kontrol edilmiş sistem çıkışı



Şekil 4.36 Denklem 4.43'deki sistemin sinüzoidal referans giriş işaretine verdiği cevap.——referans giriş,——kontrol edilmemis sistem cıkısı.——kontrol edilmis sistem cıkısı

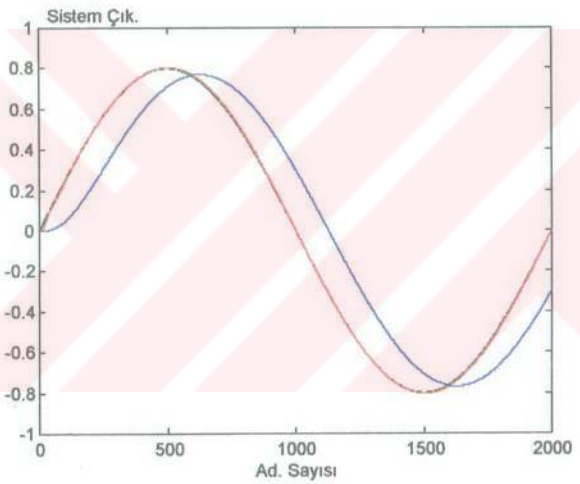


Şekil 4.37 Yapay sinir ağının 4.43'deki denklemi genelleme yeteneği (a)Çıkış katmanındaki hareketlendirme fonksiyonu tanh(),(b)Çıkış katmanında hareketlendirme fonksiyonu f(x)=x iken basamak giriş işaretine sistemin verdiği cevap.——referans giriş,——kontrol edilmemiş sistem çıkışı,——kontrol edilmiş sistem çıkışı

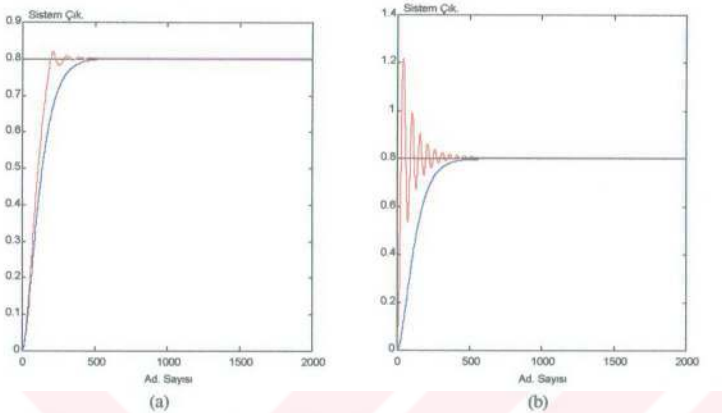
Sistem parametreleri yeniden,

$$G_{(s)} = \frac{7}{3S^2+9S+7} \qquad (4.44)$$

olarak değiştirlildiğinde, kontrol edilmiş sistemin sinüsoidal referans giriş işaretine verdiği cevap şekil 4.38'de, basamak giriş işaretine verdiği cevap ise şekil 4.39'da görülmektedir.



Şekil 4.38 Denklem 4.44'deki sistemin sinüzoidal referans giriş işaretine verdiği cevap._____referans giriş,_____ kontrol edilmemiş sistem çıkışı,_____kontrol edilmiş sistem çıkışı



(a) (b)

Şekil 4.39 Yapay sinir ağının 4.44'deki denklemi genelleme yeteneği (a)Çıkış katmanındaki hareketlendirme fonksiyonu tanh(),(b)Çıkış katmanında hareketlendirme fonksiyonu f(x)=x iken basamak giriş işaretine sistemin verdiği cevap. ——— referans giriş, ——— kontrol edilmemiş sistem çıkışı, ——— kontrol edilmiş sistem çıkışı

SONUÇLAR

Yapay sinir ağları, geçmişte öğrenmiş olduğu bilgilerden yola çıkarak karşılaşmış olduğu yeni durumlara uyum sağlayabilme özelliğinden dolayı,küçük parametre değişimlerini tölere edebilmekte ve genelleme yapabilmektedir. Bu özellikleriyle, yapay sinir ağları kontrol alanına da uygulanmakta ve klasik kontrol yöntemlerinde karşılaşılan birçok sorunu çözebilmektedir. Bunun yanında kontrol uygulamalarında, eğitim işareti sistem üzerinden veya sistem modeli üzerinden geriye yansıtıldığından ve kontrolör olarak kullanılan yapay sinir ağının girişleri, kendi çıkışlarını da içerdiğinden eğitim süresinin uzun sürmesi ve hatanın yerel minimumda kalabilme olasılığı gibi bazı sorunlarla karşılaşılabilmektedir. Bu sorunların aşılabilmesi içinde daha dinamik öğrenme algoritmalarına ve daha etkin öğrenme algoritmalarına ihtiyaç duyulmaktadır.

Bu çalışmada, yapay sinir ağları ile kontrol uygulamalarında temel olarak kullanılan doğrudan ve dolaylı kontrol yapıları ile hem verilerin teker teker işlendiği örneksel öğrenme hem de verilerin toplu olarak işlendiği toplu öğrenme uygulamalarında, örnek alınan sistemlerin hazırlanan bilgisayar programlarında simülasyonu yapılmıştır. Yapılan çalışmalarda, toplu öğrenmelerde momentum ve öğrenme katsayılarının uyarlandığı bazı algoritmaların olmasından dolayı örneksel öğrenmeye göre daha kolay ve kısa zamanda kontrol ağı eğitilebilmektedir. Kontrol yöntemi olarak ise, doğrudan kontrol yöntemi ve dolaylı kontrol yöntemlerinin her ikisinin de eğitim

süreleri ve hata değerlerinin birbirine yakın değerlerde olduğu görülmüştür.

Yapılan simülasyonlarda, kullanılan her iki yöntemden de yapay sinir ağlarının kontrolör olarak kullanımında oldukça iyi sonuçlar elde edilmiştir. Kontrol edilen sistemin parametrelerindeki değişimleri tölere edebilmekte ve değişik giriş işaretlerini de genelleyebilmektedir. Eğitimde kullanılan sinüzoidal giriş işaretlerinde sorun olmamakla beraber, basamak giriş işaretlerinde kontrol edilmiş sistem çıkışında belirli bir iyileşme olmasına karşın, özellikle kontrol ağının çıkışında doğrusal fonksiyon kullanıldığında yükselme zamanı oldukça düşmesine rağmen salınımlar önlenememektedir. Fakat bu soruna karşın, yapılan çalışmadaki sonuçlar bize yapay sinir ağlarının kontrol alanında başarı ile kullanılabileceğini ve öğrenme algoritmalarında gerçekleşecek gelişmelerle gelecekte kontrol alanında daha geniş bir uygulama alnı bulacağını göstermektedir.

1. AKIN, E.,GÖKBULUT,M., (1997). Fırçasız Doğru Akım Motorlarının YSA ile Tanılanması Ve Öğrenme Algoritmaları. **F.Ü. Fen Bil. Ens. Araştırma Raporu**

2. BATES, J.,ELBULUK, M.E.,ZİNGER, D.S.,(1993). Neural NetworkControl of a Chopper_Fed DC Motor. **IEEE 8th APEC Conference.** pp:887-891

3. BİLLİNGS, S.A.,JAMALUDDİN, H.B., CHEN, S., (1992). Properties of neural Networks With Application to Modelling Nonlineer Systems. **İnt.j Control.** vol.55,No.1,pp:193-224.

4. BILLING, S.A., JAMALUDDİN, H.B., CHEN, S.,(1992). Properties of Neural Networks With Application to Modelling Nonlineer Systems. **Int j.Control.** pp:193-224

5. BROWN, M.,HARRİS, C., (1994). Neuro-Fuzzy Adaptive Modelling and Control. **Prentice Hall int.**

6. CHAN, H.C.,CHAU, K.T.,CHAN, C.C,(1993).A Neural NetworkController for Switching Power Converters. **IEEE 8th APEC Conference.** pp:887-891

7. CHEN, C.L.,CHANG, F.Y., (1996). Design And Analysis And Neural/Fuzzy Variable Structural PID Control Systems. **IEEE Proc. Control Theory Appl.** Vol:143, No:2, pp:200-208.

8. DRAGO, G.P., MORANOO, M., RİDELLA, S.,(1995). An Adaptive Momentum Back Probagation Neural Computing & Application & pringer-Verlog. **London Lmt.** pp:213-221.

9. EL-SHARKAWİ,M.A., EL-SAMAHY,A.A., EL-SAYED, M.L., (1994). High Performance Drive Of DC Brushless Motor Using Neural Networks. **IEEE Transaction On Energy Conversion.** Vol:9, No:2, pp:317-322

10. GÖKBULUT, M.,ALBOSTAN, A., (1996). Sabit Mıknatıslı Senkron Motorların Yapay Sinir Ağları ile Modellenmesi ve Uyarlamalı Denetimi. **Gazi Ünv. Müh. Mim Fak. Dergisi.** vol:11, No:2 sayfa:22-28.

11. HABTOM, R., LİTZ,L., (1997. Neural Network-Based Control Of Nonlinear Dynamic Systems Wıth Unmeasured Input. **Proc. 12th Conf. On Systems Engineering ICSE'97.** pp:296-300.

12. HARVEY, R. L., (1994). Neural Network Principles. **Prentice-Hall, Inc. New Jersey.**

13. HİNTON, G.E.,(1992). How Neural Networks Learn from Exprience. **Scientific American Sept.**

14. HİNTON,G.E., PLAUT, D.C., SHALLİCE, T.,(1993). Beyin Hasarlarının Simülasyonu. **Bilim Scientific American Türkçe Baskı.pp:58-64.**

15. HO, K.C., PONNAPALLI, P.V.S, THOMSON, M., (1997). Novel Pruning And Extrapolation Techniques For The Optimization Of Multi-Type Fanns Architectures: Application To CSTR Modelling. **Proc. 12th Int. Conf. On Systems Engineering ICSE'97.** pp:313-318.

16. İMRAK, C.E.,(1996). Asansör Sistemlerinin Trafik Analizi Dizaynı ve Simülasyonu. **İTÜ Fen Bil. Ens. Doktora Tezi.**

17. KARAHAN, Ö., HALICI, U., LEBLEBİCİOĞLU, K., (1997). Yapay Sinir Ağı İle Bütünleştirilmiş Model Öngörümlü Denetim.**Otomasyon Teknolojileri Sempozyumu TOK'97.** pp:29-34.

18. KARNA, K.N., DAVİD, M.B., (1989). An Artificial Neural Networks Tutorial Part:1-Basics. **Neural Networks.** Vol:1, No:1, pp:5-23.

19.KRAFT, L.G., CAMPAGNA, D.P., (1990). A Comparison Between CMAC Neural Network Control and Two traditional Control Systems. **IEEE Control Systems Magazine.** pp:36-43.11

20. ÖZMETELER, E.,(1989). Yapay Nöral Ağlar. **İTÜ Fen Bil. Ens. Yüksek Lisans Tezi.**

21. RUMELHART, D.,HINTON, G.E., WILLIAMS, R.J.,(1986). Parallel Distributed Processing; Explaration in the Microstructure of Congition. **The MIT Press Cambridge.**

22. SANNER, R.M.,SLOTİNE, J.J.E., (1992). Gaussian Networks for Direct Adaptive Control. **IEEE Trans. Neural Networks.** pp:887-863.

23. SBARBARO-HOFER, D.,NEUMERKEL, D.,HUNT, K.,(1993).Neural Control of a Steel Rolling Mill. **IEEE Control System Magazine.** pp:70-75.

24. KOÇAK, T.,CILIZ, K., KAYNAK, O.,(1996). Yapay Sinir Ağları ile Gerçek Zamanda Denetim. **Otomasyon Teknolojileri Sempozyumu TOK'96.** pp:23-30.

25. WEERASOORİYA, S., EL-SHARKAWİ, M.A.,(1991). Identification And Control Of DC Motor Using Back-Propagation Neural Networks. **IEEE Transaction On Energy Conversion**. Vol:6, No:4, pp:663-669.

26. ZHANG, Y., SEN, P., HEARN, G.E., (1995). An On-Line Trained Adaptive Neural Controller. **IEEE Control Systems**. pp:67-75.

EK-1

```
#include<stdio.h>
#include<graphics.h>
#include<stdlib.h>
#include<conio.h>
#include<math.h>
#define pi 3.141592654
#define ADIM 2000
#define GRS 6
#define SKL 7

void ref_gir(double *ur)
     {
        register int i;
        double x;
          for(i=0;i<ADIM;i++)
            {
            x=pi*i/1000;
            *(ur+i)=sin(x);
            }
            return;
     }

void kts_oku(double py0,double py1,double py2,double pd0,double
              pd1,double pd2)
       {
        double x,t,k,a,b,c;
        printf("Örnekleme peryodunu giriniz\n T=");
        scanf("%lf",&t);
        printf("Sistem parametrelerini;\n k/a*S^2+b*S+c\nOlacak
               şekilde giriniz);
        printf("k=");
        scanf("%lf",&k);
        printf("a=");
        scanf("%lf",&a);
        printf("b=");
        scanf("%lf",&b);
        printf("c=");
        scanf("%lf",&c);
        x=2/t;
        py0=k
        py1=2*k;
        py2=k;
        pd0=a*x*x+b*x+c;
        pd1=-2*a*x*x+2*c;
        pd2=a*x*x-b*x+c;
```

```
        }

void carp1(double x[GRS],double w_g[GRS][SKL-1],double n[SKL-1])
     {
     register int i,j;
     double top=0.0;

         for(j=0;j<SKL-1;j++)
            {
            top=0.0;
            for(i=0;i<GRS;i++)
            top=top+x[i]*w_g[i][j];
            n[j]=top;
            }
     }

void sak_cikis(double net[SKL],double v[SKL])
     {
     register int i;
        for(i=0;i<SKL;i++)
           {
           v[i]=tanh(net[i]);
           }
     }

void turev(double n[SKL],double tur[SKL])
     {
     register int i;
        for(i=0;i<SKL;i++)
           {
           tur[i]=1/((cosh(n[i])*(cosh(n[i]))));
           }
     }

void topla_s(double a[SKL],double b[SKL],double topl[SKL])
     {
     register int i;
     for(i=0;i<SKL;i++)
        topl[i]=a[i]+b[i];
     }

void topla_g(double a[GRS][SKL-1],double b[GRS][SKL-1],double
              topl[GRS][SKL-1])
     {
     register int i,j;
       for(i=0;i<GRS;i++)
         {
         for(j=0;j<SKL-1;j++)
            {
```

```
            topl[i][j]=a[i][j]+b[i][j];
            }
         }
      return;
      }

double mutlak(double hata)
      {
      if(hata<0)return(-1*hata);
      return(hata);
      }

void carp_k_s(double dw[SKL],double x,double dw_e[SKL])
      {
      register int i;
         for(i=0;i<SKL;i++)
            {
            dw_e[i]=x*dw[i];
            }
      }

void carp_k_g(double dw[GRS][SKL-1],double x,double
               dw_e[GRS][SKL-1])
      {
      register int i,j;
         for(i=0;i<GRS;i++)
            {
            for(j=0;j<SKL-1;j++)
            {
            dw_e[i][j]=x*dw[i][j];
            }
            }
      }

void eski_dw_sk(double dw[SKL],double dw_eski[SKL])
      {
      register int i;
         for(i=0;i<SKL;i++)
            {
            dw_eski[i]=dw[i];
            }
      }

void eski_dw_gr(double dw[GRS][SKL-1],double dw_eski[GRS][SKL-1])
      {
      register int i,j;
         for(i=0;i<GRS;i++)
            {
            for(j=0;j<SKL-1;j++)
```

```
            {
            dw_eski[i][j]=dw[i][j];
            }
            }
      }

void agr_kyd(double w_gr[GRS][SKL-1],double [SKL])
      {
      FILE *f1g,*f1s;
      register unsigned int i,j;
      if((f1g=fopen("wdkg.dat","w"))==NULL)
        {
        printf("Dosya Açamìyorum");
        exit(0);
        }
        else{
            for(i=0;i<GRS;i++)
               {
               for(j=0;j<SKL-1;j++)
                  fprintf(f1g,"\n%.20lf",w_g[i][j]);
               }
            }
             fclose(f1g);
                 if((f1s=fopen("wdks.dat","w"))==NULL)
                   {
                   printf("Dosya Açamìyorum");
                   exit(0);
                   }
                   else{
                       for(i=0;i<SKL;i++)
                          fprintf(f1s,"\n%.20lf",w_s[i]);
                        }
                 fclose(f1s);
     }

void cks_kyd(double *y,double *uref)
      {
      FILE *fcks,*fur;
      register unsigned int i,j;

      if((fcks=fopen("cks.dat","w"))==NULL)
        {
        printf("Dosya Açamìyorum");
        exit(0);
        }
        else{
            for(i=0;i<ADIM;i++)
            fprintf(fcks,"\n%.20lf",*(y+i));
            }
```

```
      fclose(fcks);
             if((fur=fopen("grs.dat","w"))==NULL)
               {
               printf("Dosya Açamıyorum");
               exit(0);
               }
               else{
                for(i=0;i<ADIM;i++)
                fprintf(fur,"\n%.20lf",*(uref+i));
                }
             fclose(fur);
      }
/*
void bas_w_gr(double w_gr[GRS][SKL-1])
      {
      register unsigned int i,j;
      FILE *f1;
      if((f1=fopen("wdkg.dat","r"))==NULL)
        {
        printf("Dosyayı Aaçamıyorum");
        exit(0);
        }
           for(i=0;i<GRS;i++)
           {
           for(j=0;j<SKL-1;j++)
              fscanf(f1,"%lf",&w_gr[i][j]);
           }
      fclose(f1);
      }

void bas_w_sak(double w_s[SKL])
      {
      register unsigned int i;
      FILE *f2;
      if((f2=fopen("wdks.dat","r"))==NULL)
        {
        printf("Dosyayı Aaçamıyorum");
        exit(0);
        }
        for(i=0;i<SKL;i++)
           fscanf(f2,"%lf",&w_s[i]);
      fclose(f2);
      }
*/
void bas_w_gr(double w_gr[GRS][SKL-1])
      {
      register int i,j;
      double a;
         for(i=0;i<GRS;i++)
```

```
            {
            for(j=0;j<SKL-1;j++)
               {
               a=rand();
               w_gr[i][j]=a/RAND_MAX;
               }
            }
       }
void bas_w_sak(double w_s[SKL])
     {
     register int i;
     double a;
        for(i=0;i<SKL;i++)
            {
            a=rand();
            w_s[i]=a/RAND_MAX;
            }
     }

void main()
     {
     register int i,j,k;
     double alf,beta,py0,py1,py2,pd0,pd1,pd2;
     double *u,*y,x[GRS],*uref,net[SKL],tur[SKL-1],w_g[GRS][SKL-
1],w_s[SKL],v[SKL],dw_g[GRS][SKL-1],dw_s[SKL],dw_g_es[GRS][SKL-
1],dw_s_es[SKL];
     double e,ec1,ec2,ec3,ec,turc,alfy,top1,Ey,Ees,iter=0.0;
     uref=(double *)malloc((ADIM)*sizeof(double));
     u=(double *)malloc((ADIM+3)*sizeof(double));
     y=(double *)malloc((ADIM+3)*sizeof(double));

       if(!u&!y&!uref)
         {
         printf("Hafìza Ayìramìyorum\n");
         exit(1);
         }
         gotoxy(1,6);
         *uref=0.0;uref=uref;
         for(i=0;i<GRS;i++)
            {
            for(j=0;j<SKL-1;j++)
            dw_g[i][j]=0.0;
            }
        for(i=0;i<SKL;i++)
           dw_s[i]=0.0;

         ref_gir(uref);
         kts_oku(py0,py1,py2,pd0,pd1,pd2);
         bas_w_gr(w_g);
```

```
        bas_w_sak(w_s);
        alfy=0.0;Ees=5000;alf=0.00001;beta=0.9;

  while(1)
       {
       top1=0.0;
       Ey=0.0;
       *y=0.0;*(y+1)=0.0;*(y+2)=0.0;y=y+2;
       *u=0.0;*(u+1)=0.0;*(u+2)=0.0;u=u+2;
       iter++;
       eski_dw_sk(dw_s,dw_s_es);
       eski_dw_gr(dw_g,dw_g_es);
     for(i=0;i<GRS;i++)
        {
        for(j=0;j<SKL-1;j++)
           dw_g[i][j]=0.0;
        }
       for(i=0;i<SKL;i++)
       dw_s[i]=0.0;

       for(k=0;k<ADIM;k++)
          {
          e=0.0;
          ec=0.0;
          ec1=0.0;ec2=0.0;ec3=0.0;
          top1=0.0;
          turc=0.0;
            x[0]=(*(uref+k));
            x[1]=(*(u+(k-1));
            x[2]=(*(u+(k-2));
            x[3]=(*(y+(k-1)));
            x[4]=(*(y+(k-2)));
            x[5]=1.0;
                 carp1(x,w_g,net);
                   net[SKL-1]=1.0;
                 sak_cikis(net,v);
                 turev(net,tur);
                    for(i=0;i<SKL;i++)
                    top1+=v[i]*w_s[i];
                    *(u+k)=tanh(top1);
                    turc=1/((cosh(*(u+k))*cosh(*(u+k))));
*(y+k)=(py0*(*(u+k))+py1*(*(u+(k-1)))+py2*(u+(k-2)))-pd1*(*(y+(k-
1)))-pd2*(*(y+(k-2))))/pd0;
                e=(*(uref+k))-(*(y+k));
                ec1=((*(y+k))-(*(y+(k-1))));
                ec2=((*(u+k))-(*(u+(k-1))));
                if(ec1*ec2>0){ec3=1.0;}
                else if(ec1*ec2<0){ec3=-1.0;}
                else {ec3=0.0;}
```

```
        ec=e*ec3;

        for(i=0;i<SKL;i++)
        dw_s[i]+=(ec*turc*v[i]);
              for(j=0;j<SKL-1;j++)
             {
             for(i=0;i<GRS;i++)
                dw_g[i][j]+=(x[i]*ec*turc*tur[j]*w_s[j]);
             }

Ey=Ey+e*e;
}
printf("%.0lf  %.10lf  %.10lf\n",iter,alf,Ey);
if(Ey<(0.1))
  {
    cks_kyd(y,uref);
    agr_kyd(w_g,w_s);

  exit(1);
  }

   else if(Ey<Ees)
       {
       alf=1.05*alf;
       carp_k_s(dw_s,alf,dw_s);
       carp_k_s(dw_s_es,beta,dw_s_es);
       topla_s(dw_s,dw_s_es,dw_s);
       topla_s(w_s,dw_s,w_s);
       carp_k_g(dw_g,alf,dw_g);
       carp_k_g(dw_g_es,beta,dw_g_es);
       topla_g(dw_g,dw_g_es,dw_g);
       topla_g(w_g,dw_g,w_g);
       }
       else if(Ees<=Ey&Ey<1.05*Ees)
              {
              alf=0.7*alf;
              carp_k_s(dw_s,alf,dw_s);
              carp_k_s(dw_s_es,0.0,dw_s_es);
              topla_s(dw_s,dw_s_es,dw_s);
              topla_s(w_s,dw_s,w_s);
              carp_k_g(dw_g,alf,dw_g);
              carp_k_g(dw_g_es,0.0,dw_g_es);
              topla_g(dw_g,dw_g_es,dw_g);
              topla_g(w_g,dw_g,w_g);
              }
              else
              {
              alf=0.1*alf;
              alfy=-1.0*alf;
```

```
                    carp_k_s(dw_s,alfy,dw_s);
                    carp_k_s(dw_s_es,0.0,dw_s_es);
                    topla_s(dw_s,dw_s_es,dw_s);
                    topla_s(w_s,dw_s,w_s);
                    carp_k_g(dw_g,alfy,dw_g);
                    carp_k_g(dw_g_es,0.0,dw_g_es);
                    topla_g(dw_g,dw_g_es,dw_g);
                    topla_g(w_g,dw_g,w_g);
                    }

        Ees=Ey;
        }
}
```

EK-2

```
#include<conio.h>
#include<graphics.h>
#include<math.h>
#define BIAS 1
#define pi 3.141592654
#define ZMN 500000
#define ADIM 2000
#define GRS 6
#define SKL 7
#define betak 0.4
#define alfk 0.001

void kts_oku(double py0,double py1,double py2,double pd0,double
      pd1,double pd2)
     {
      double x,t,k,a,b,c;
      printf("Örnekleme peryodunu giriniz\n T=");
      scanf("%lf",&t);
      printf("Sistem parametrelerini;\n k/a*S^2+b*S+c\nOlacak
            şekilde giriniz);
      printf("k=");
      scanf("%lf",&k);
      printf("a=");
      scanf("%lf",&a);
      printf("b=");
      scanf("%lf",&b);
      printf("c=");
      scanf("%lf",&c);
      x=2/t;
      py0=k
      py1=2*k;
      py2=k;
      pd0=a*x*x+b*x+c;
      pd1=-2*a*x*x+2*c;
      pd2=a*x*x-b*x+c;
     }

void carp1(double a[GRS],double b[GRS][SKL-1],double c[SKL-1])
     {
     register unsigned int i,j;
     double top=0.0;
     for(i=0;i<SKL-1;i++)
        {
        top=0.0;
        for(j=0;j<GRS;j++)
```

```
            top=top+a[j]*b[j][i];
            c[i]=top;
         }
      }

void sak_kat(double net[SKL],double act[SKL])
      {
      register unsigned int i;
      for(i=0;i<SKL;i++)
         act[i]=tanh(net[i]);
      }

void turev(double n[SKL],double t[SKL])
      {
      register unsigned int i;
      for(i=0;i<SKL;i++)
         t[i]=1/((cosh(n[i]))*(cosh(n[i])));
      }

void carp2(double a[SKL-1],double b[GRS],double c[GRS][SKL-1])
      {
      register unsigned int i,j;
      for(i=0;i<SKL-1;i++)
         {
         for(j=0;j<GRS;j++)
            c[j][i]=a[i]*b[j];
         }
      }

void topla1(double a[SKL],double b[SKL],double c[SKL])
      {
      register unsigned int i;
      for(i=0;i<SKL;i++)
         c[i]=a[i]+b[i];
      }

void topla2(double a[GRS][SKL-1],double b[GRS][SKL-1],double
c[GRS][SKL-1])
      {
      register unsigned int i,j;
      for(i=0;i<GRS;i++)
         {
         for(j=0;j<SKL-1;j++)
            c[i][j]=a[i][j]+b[i][j];
         }
      }

double mutlak(double hata)
      {
```

```
      if(hata<0)return(-1*hata);
      return(hata);
      }
 /*
void bas_w_gr(double w_gr[GRS][SKL-1])
      {
      register unsigned int i,j;
      FILE *f1;
      if((f1=fopen("wkg.dat","r"))==NULL)
        {
        printf("Dosyayì Aaçamìyorum");
        exit(0);
        }
           for(i=0;i<GRS;i++)
           {
           for(j=0;j<SKL-1;j++)
              {
              fscanf(f1,"%lf",&w_gr[i][j]);
              }
           }
      fclose(f1);
      }

void bas_w_sak(double w_s[SKL])
      {
      register unsigned int i;
      FILE *f2;
      if((f2=fopen("wks.dat","r"))==NULL)
        {
        printf("Dosyayì Aaçamìyorum");
        exit(0);
        }
        for(i=0;i<SKL;i++)
        {
        fscanf(f2,"%lf",&w_s[i]);
        }
        fclose(f2);
      }
*/
void bas_w_gr(double w_gr[GRS][SKL-1])
      {
      register int i,j;
      double a;
         for(i=0;i<GRS;i++)
             {
             for(j=0;j<SKL-1;j++)
             {
             a=rand();
             w_gr[i][j]=a/RAND_MAX;
```

```
            }
            }
      }

void bas_w_sak(double w_s[SKL])
     {
     register int i;
     double a;
        for(i=0;i<SKL;i++)
           {
           a=rand();
           w_s[i]=a/RAND_MAX;
           }
     }
void bas_wm_gr(double w_gr[GRS][SKL-1])
     {
     register unsigned int i,j;
     FILE *f1;
     if((f1=fopen("wg.dat","r"))==NULL)
       {
       printf("Dosyayì Aaçamìyorum");
       exit(0);
       }
          for(i=0;i<GRS;i++)
          {
          for(j=0;j<SKL-1;j++)
             {
             fscanf(f1,"%lf",&w_gr[i][j]);
             }
          }
     fclose(f1);
     }

void bas_wm_sak(double w_s[SKL])
     {
     register unsigned int i;
     FILE *f2;
     if((f2=fopen("ws.dat","r"))==NULL)
       {
       printf("Dosyayì Aaçamìyorum");
       exit(0);
       }
       for(i=0;i<SKL;i++)
       {
       fscanf(f2,"%lf",&w_s[i]);
       }
       fclose(f2);
     }
void eski_dw_sak(double dw[SKL],double x,double dw_eski[SKL])
```

```
      {
      register unsigned int i;
      for(i=0;i<SKL;i++)
         dw_eski[i]=x*dw[i];
      }

void eski_dw_gr(double dw[GRS][SKL-1],double x,double
                dw_eski[GRS][SKL-1])
      {
      register unsigned int i,j;
      for(i=0;i<GRS;i++)
         {
         for(j=0;j<SKL-1;j++)
            dw_eski[i][j]=x*dw[i][j];
         }
      }
void cks_kyd(double *y,double *uref)
      {
      FILE *fcks,*fur;
      register unsigned int i,j;

      if((fcks=fopen("cks.dat","w"))==NULL)
        {
        printf("Dosya Açamìyorum");
        exit(0);
        }
        else{
            for(i=0;i<ADIM;i++)
            fprintf(fcks,"\n%.20lf",*(y+i));
            }
      fclose(fcks);
             if((fur=fopen("grs.dat","w"))==NULL)
               {
               printf("Dosya Açamìyorum");
               exit(0);
               }
               else{
                for(i=0;i<ADIM;i++)
                fprintf(fur,"\n%.20lf",*(uref+i));
                }
             fclose(fur);
      }

void main()
      {
      double k=0.0,rad;
      unsigned int i,j,l=0.0;
      FILE *fkg,*fks,*fmg,*fms,*fwg,*fws;
      double *uref,*y1,*u1,u2,*y,e1,ec,ec1[SKL-
```

```
1],x1[GRS],x2[GRS],n1[SKL],n2[SKL],tur1[SKL],tur2[SKL],v1[SKL],v2[
SKL],w_k_g[GRS][SKL-
1],w_k_s[SKL],dw_k_g[GRS][SKL-1],dw_k_s[SKL],dw_k_g_es[GRS][SKL-
1],dw_k_s_es[SKL],w_m_g[GRS][SKL-1],w_m_s[SKL],sgm_k_g[SKL-1];
     double top1,turm,turc,top2,tpec,E1=0.0;
     double py0,py1,py2,pd0,pd1,pd2;
     uref=(double *)malloc((2000)*sizeof(double));
     y1=(double *)malloc((2000)*sizeof(double));
     u1=(double *)malloc((3)*sizeof(double));
     y=(double *)malloc((3)*sizeof(double));

       if(!u1&!y&!uref)
       {
       printf("hafiza ayiramiyorum");
       exit(1);
       }
         gotoxy(1,6);
            *uref=0.0;uref=uref;
            *u1=0.0;*(u1+1)=0.0;*(u1+2)=0.0;u1=u1+2;
            *y=0.0;*(y+1)=0.0;*(y+2)=0.0;y=y+2;
            *y1=0.0;y1=y1;

           for(i=0;i<SKL;i++)
              dw_k_s[i]=0.0;
               for(i=0;i<GRS;i++)
                  {
                  for(j=0;j<SKL-1;j++)
                     dw_k_g[i][j]=0.0;
                  }
             kts_oku(py0,py1,py2,pd0,pd1,pd2);
             bas_w_gr(w_k_g);
             bas_w_sak(w_k_s);
             bas_wm_gr(w_m_g);
             bas_wm_sak(w_m_s);

  while(1)
       {
       k++;
       ec=0.0;
       e1=0.0;
       u2=0.0;
       rad=0.0;
       top1=0.0;
       top2=0.0;
       tpec=0.0;
       turc=0.0;
       turm=0.0;
       rad=pi*k/1000;
```

```
            x1[0]=sin(rad);
            x1[1]=(*(u1-1));
            x1[2]=(*(u1-2));
            x1[3]=(*(y-1));
            x1[4]=(*(y-2));
            x1[5]=BIAS;
                  carp1(x1,w_k_g,n1);
                    n1[SKL-1]=BIAS;
                  sak_kat(n1,v1);
                  turev(n1,tur1);
                    for(i=0;i<SKL;i++)
                       top1=top1+v1[i]*w_k_s[SKL-1];
                       *(u1+0)=tanh(top1);
                       turc=1/((cosh(*(u1+0)))*(cosh(*(u1+0))));
        *(y+0)=(py0*(*(u1+0))+ py1*(*(u1-1))+ py2*(*(u1-2))-
pd1*(*(y-1))-pd2*(*(y-2)))/pd0;
        e1=x1[0]-(*(y+0));
        printf("y=%lf   uref(%.0lf)=%.4lf\n",*(y+0),k,x1[0]);
            x2[0]=(*(u1+0));
            x2[1]=(*(u1-1));
            x2[2]=(*(u1-2));
            x2[3]=(*(y-1));
            x2[4]=(*(y-2));
            x2[5]=BIAS;
                  carp1(x2,w_m_g,n2);
                    n2[SKL-1]=BIAS;
                  sak_kat(n2,v2);
                  turev(n2,tur2);
                    for(i=0;i<SKL;i++)
                       top2=top2+v2[i]*w_m_s[i];
                       u2=(top2);
       eski_dw_gr(dw_k_g,betak,dw_k_g_es);
       eski_dw_sak(dw_k_s,betak,dw_k_s_es);
       for(i=0;i<SKL-1;i++)
       ec1[i]=w_m_s[i]*tur2[i];
          for(i=0;i<SKL-1;i++)
             tpec=tpec+ec1[i]*w_m_g[0][i];
               if(tpec<0){tpec=-1;}else
if(tpec>0){tpec=1;}else{tpec=0.0;}
                  ec=e1*tpec;
                for(i=0;i<SKL;i++)
                   dw_k_s[i]=alfk*ec*turc*v1[i];
                      for(i=0;i<SKL-1;i++)
                         sgm_k_g[i]=alfk*ec*turc*tur1[i]*w_k_s[i];

                      carp2(sgm_k_g,x1,dw_k_g);
                      topla2(dw_k_g,dw_k_g_es,dw_k_g);
                      topla2(w_k_g,dw_k_g,w_k_g);
                      topla1(dw_k_s,dw_k_s_es,dw_k_s);
```

```
                topla1(w_k_s,dw_k_s,w_k_s);
       if(k>=ZMN)
      {
      *(y1+1)=*(y+0);
      *(uref+1)=x1[0];
      l++;
      E1+=e1*e1;
      }
       if(k>(ZMN+ADIM))
         {
         clrscr();
             cks_kyd(y1,uref);
              if((fkg=fopen("wkg.dat","w"))==NULL)
                {
                printf("Dosya Açamìyorum");
                exit(0);
                }
                  for(i=0;i<GRS;i++)
                  {
                  for(j=0;j<SKL-1;j++)
                  fprintf(fkg,"\n%.20lf",w_k_g[i][j]);
                  }
                fclose(fkg);

                    if((fks=fopen("wks.dat","w"))==NULL)
                      {
                      printf("Dosya Açamìyorum");
                      exit(0);
                      }
                    for(i=0;i<SKL;i++)
                       fprintf(fks,"\n%.20lf",w_k_s[i]);
                    fclose(fks);
         printf("E1=%lf  \n",E1);
         getch();
         exit(1);
         }

 *(y-2)=(*(y-1));
 *(y-1)=(*(y+0));
 *(y+0)=0.0;
 *(u1-2)=*(u1-1);
 *(u1+0)=*(u1+0);
 *(u1+0)=0.0;

}
}
```