T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ELEKTRİK DEVRELERİNİN YAPAY SİNİR AĞLARI İLE TANINMASI VE KONTROLÜ

Mehmet SAMAN

YÜKSEK LİSANS TEZİ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ
ANA BİLİM DALI

Bu Tez ,07/02/2003 Tarihinde Aşağıda Belirtilen Jüri Tarafından Oybirliği /Oyçokluğu ile Başarılı/Başarısız Olarak Değerlendirilmiştir.

(İmza)	(İmza)	(İmza)
Danışman	Üye	Üye
Doç. Dr. Yakup Demir		Yrd.Doç.Dr. Selçuk YILDIRIM

Bu tezin kabulü, Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 19/02/2003 tarih ve.....7.12.....sayılı kararıyla onaylanmıştır.

TEŞEKKÜR

Bu çalışmam boyunca beni daima destekleyen ve teşvik eden, ayrıca da büyük sabır gösteren çok değerli hocam Doç. Dr. Yakup DEMİR'e, her türlü yardımı esirgemeyen arkadaşım Arş. Gör. Ayşegül UÇAR'a ve desteklerinden dolayı değerli mesai arkadaşlarıma sonsuz teşekkürlerimi sunarım.

İÇİNDEKİLER

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

SİMGELER LİSTESİ

T	Eşik değeri
x	Giriş vektörü
y	Çıkış vektörü
d	İstenilen çıkış değeri
w	Ağırlık matrisi
f(.)	Aktivasyon fonksiyonu
Γ[.]	Doğrusal olmayan matris oparetörü
Δ	Gecikme elemanı
u	Sistem giriş vektörü
ℓ	Mesafe fonksiyonu
c,η	Öğrenme oranı
r	Referans giriş sinyali
e	Karesel hata
α,β	Öğrenme sabiti
δ	Hata terimi
p	Giriş datalarının sayısı
W2, w	Çıkış ile gizli katman arasındaki ağırlık matrisi
W1, v	Giriş ile gizli katman arasındaki ağırlık matrisi
N	Bileşik doğrusal olmayan matris oparetörü

ÖZET

Yüksek Lisans Tezi

ELEKTRİK DEVRELERİNİN YAPAY SİNİR AĞLARI İLE TANINMASI VE KONTROLÜ

Mehmet SAMAN

Fırat Üniversitesi

Fen Bilimleri Enstitüsü

Elektrik-Elektronik Mühendisliği

Ana Bilim Dalı

2003, Sayfa: 84

Klasik ve modern kontrol teorisi lineer modeller temeline dayanmaktadır. Fakat pratikte sistemlerin genellikle lineer olmaması, lineer kontrol teorisinin başarılı uygulamalarını yapmak için büyük engel teşkil etmektedir. Doğal olarak lineer olmayan dönüşüm kabiliyetlerine sahip olan yapay sinir ağları (YSA), lineer olmayan tanıma ve kontrol problemlerine güçlü ve alternatif bir çözüm getirmiştir.

Literatürde YSA'ların farklı yapıları mevcuttur. Bu tezde ileri beslemeli yapay sinir ağları (İYSA), radyal tabanlı yapay sinir ağları (RTFA) ve modüler yapay sinir ağları (MYSA) ele alınmıştır. İYSA'lar sistem tanımada oldukça iyi performans gösterirler fakat eğitim süreleri uzundur. İYSA'ların tanıma performansı farklı eğitim algoritmaları kullanılarak düzeltilebilir. Bu çalışmada İYSA'ların eğitimi için geriye yayılım, Levenberg Marquard, Gauss Newton ve eşlenik iniş algoritmaları ele alınmış ve simülasyon sonuçları ile birlikte tartışılmıştır.

MYSA'lar her parçayı tanımak amacıyla farklı yapıları ve öğrenme algoritmalarını birlikte kullanabildikleri için, İYSA'lara göre daha az sürede eğitilmelerine rağmen, sadece parça parça doğrusal karakteristiğe sahip olan sistemlerin tanınmasında İYSA'lardan daha iyi performans gösterirler. Eğitimi diğer ağlara göre oldukça hızlı olan RTFA'ların tanıma performansı ise radyal taban fonksiyonlarının parametrelerine ve lineer ağırlıklara göre değişmektedir.

Ele alınan eğitim yöntemlerine ve YSA'ların lineer olmayan sistemleri güçlü bir şekilde tanıma yeteneğine dayanarak sistem tersini elde etme ile kontrol, içten model tabanlı kontrol, optimal

kontrol ve tahmini kontrol yaklaşımları ile kontrolör tasarımı yöntemleri örnek elektrik devrelerine uygulanmış ve simülasyon sonuçları ile ele alınmıştır.

Anahtar Kelimeler: İYSA, MYSA, RTFA, Nöral Kontrolör, Guasi-Nevton, Geriye Yayılım, Levenberg-Marquardt, Eşlenik Eğim.

ABSTRACT

Master Thesis

IDENTIFICATION AND CONTROL OF ELECTRICAL CIRCUITS USING NEURAL NETWORKS

Mehmet SAMAN

Fırat University

Institute of Scientific

Department of Electrical-Electronic

2003, Page:84

A classical and modern control theory are based upon linear models. However, the fact that most practical systems are nonlinear constitutes an enormous difficulty for doing successfully many applications. Using artificial neural networks (ANNs) that have naturally nonlinear mapping capability is an alternative and powerful solution to nonlinear control and identification problems.

Different architectures of ANNs have been used in the literature. In this thesis, three different ANNs architectures are covered: feedforward neural networks (FNNs), radial basis function neural network (RBFNN), and modular neural networks (MNNs). Although FNN gives good performances in system identification, long training time for its learning is required. The learning algorithms are used to improve their identification performances. Levenberg Marquard, Gauss-Newton, and Conjugate Gradient here are taking into consideration and discussed by the simulation results.

When MNNs are compared with FNNs, they provides more performances than FNNs for only piece-wise linear system in term of less training time. Because MNNs combines different the learning algorithms and ANNs architectures. On the other hand, RBFNN trains in very small time and their performances changes according to their linear weights and radial basis functions parameters.

Based on these learning methods and identification capability nonlinear system of ANNs, inverse control, internal model based control, optimal control and predictive control methodologies are applied on the example electrical circuits and excessive simulation results are discussed in this thesis.

Keywords: ANN, FNN, MNN, RBFNN, Neuro-Control, Guasi-Nevton, Backpropagation, Levenberg-Marquardt, Conjugate-Gradient.

BÖLÜM 1. GİRİŞ

İnsanoğlu tarih boyunca makinanın rolünü arttıran ve insanın rolünün daha da azalmasını gerektiren fiziksel uygulamaların arayışına girmiştir. Bunun sonucunda yeni teknolojiler, endüstride yenilikler ve ihtiyaçlar ortaya çıkmıştır. Bu gelişmeler problem alanının çok boyutluluğunu ortaya çıkarmıştır. Yapay Sinir Ağları (YSA), fuzzy lojik ve genetik algoritmalar gibi zeki sistemler bu hızlı büyümenin doğal sonucudur.

YSA paralel veri işleme özellikleri, uyarlanabilirlik ve güçlü dönüştürme özelliklerinden dolayı modelleme, kontrol gibi bir çok araştırma alanına yeni bir düşünüş tarzı getirmiştir. Özellikle öğrenebilme özelliği bu ağları daha çekici kılmıştır [1, 2, 3, 4, 5]. Sistemler doğrusal olarak davrandığında, sistemlerin analiz ve kontrolü için lineer cebrin güçlü özellikleri kullanılabilir. Gerçekte sistemlerin çoğu doğrusal olmayıp eşdeğer bir doğrusal gösterimle modellenebilirler [6]. YSA'ın özellikle doğrusal olmayan sistemleri modelleme kabiliyeti, bu tip sistemlerin analizinde yaygın olarak kullanılmasını sağlamıştır [7].

Günümüz teknolojisi çok kompleks sistemlerin kontrolünü gerektirmektedir. Sistemin kompleksliğinin artması çok daha özellikli kontrolörlere ihtiyaç göstermiştir. İşlem hakkında kesin bir bilginin yokluğu, önemli parametrelerin zamana bağlı değişimi, doyum sorunları ve zaman gecikmesinin uzunluğu klasik bir kontrolör dizaynını oldukça güçleştirir. YSA'nın kullanımı kontrolör tasarımını kolaylaştıran bir yaklaşımdır [8].

YSA'lara ilişkin farklı görüşler vardır. Bu görüşlerden birincisi, YSA'ların matematiksel algoritmaların sadece bir sınıfı olduğu şeklindedir. İkinci görüşte ise, YSA' ların canlı organizmalarda bulunan biyolojik sinir ağlarını taklit eden suni ağlar olduğu tanımlanmıştır [9]. Ancak birinci görüş daha akla uygundur. Çünkü YSA'lar biyolojik sinir ağlarından etkilenmiş olmasına rağmen gerçeğini taklit etmekten oldukça uzaktır [10]. Literatürde, YSA'lar için; neurocomputing, network computation, katmanlı adaptif sistemler, connectionism, kendi kendini düzenleyen devreler veya neuromorphic sistem veya devreler gibi birçok isimlendirmeler mevcuttur. Bunlar değişik bakış açıları veya farklı inceleme konularından kaynaklanmaktadır. Bir YSA iç içe bağlanmış bir çok otonom temel elemandan oluşur. Bu elemanların her biri basit bir fonksiyona sahiptir. Kompleks fonksiyonlar, bu basit elemanların dinamik iç bağlantılarının gelişen davranışları olarak elde edilirler [7, 11].

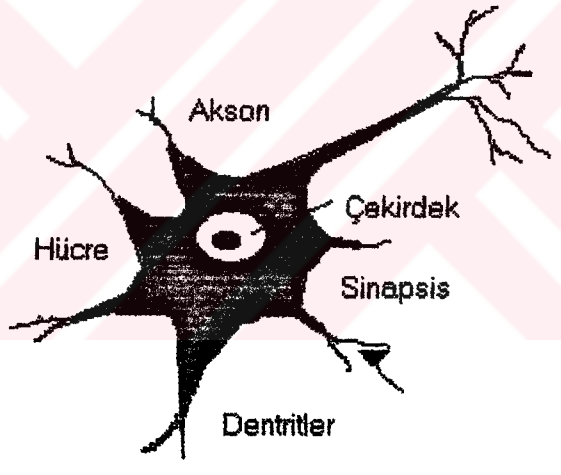
Bu çalışmada; YSA'lar farklı ağ yapılarına göre ileri ve geri beslemeli YSA'lar, radyal tabanlı fonksiyonlu ağlar (RTFA) ve modüler yapay sinir ağları (MYSA) ana başlıkları altında incelenmiştir. Bu YSA yapılarının modelleme kabiliyetleri karşılaştırılmıştır ve çeşitli YSA kontrol yapıları incelenmiştir. İleri beslemeli yapay sinir ağlarının (İYSA) eğitiminde geriye yayılım metodu, Levenberg-Marqurdt metodu, eşlenik-eğim metodu, Guasi-Newton metodu gibi bir çok yöntem kullanılmaktadır. RFTA'ların eğitimde rasgele seçilmiş sabit merkezler, merkezlerin kendi kendini organize ederek secimi ve merkezlerin denetimli seçimi olmak üzere üç metot sunulmuştur. Modüler ağların eğitimi için tahmini gradient öğrenme algoritması kullanılmıştır.

Bu çalışmanın amacı YSA'ların yapısı hakkında genel bilgi verip elektrik devrelerinin modellenmesi ve kontrolünde nasıl kullanıldıklarını göstermektir. Çalışma 7 bölümden oluşmaktadır. Bölüm 1 YSA'lar hakkında kısa bir giriş içermektedir. Bölüm 2'de ileri ve geri beslemeli YSA'lardan kısaca bahsedilerek, İYSA, RTFA ve MYSA incelenmiş ve genel özellikleri verilmiştir. Bölüm 3'de İYSA'lar için öğrenme ve adaptasyon algoritmaları tanıtılmıştır. Bölüm 4'de YSA'lar ile sistem tanımlama için yöntem verilmiştir. Bölüm 5'de ise İYSA kullanılarak yapılan kontrol metotları tanıtılmıştır. Bölüm 6'da bazı elektrik devrelerinin YSA ile modellenmesi ve kontrolü yapılmış ve simülasyon sonuçları verilmiştir. Bölüm 7'de ise çalışma hakkında bir değerlendirme yer almaktadır.

BÖLÜM 2. YAPAY SİNİR AĞLARININ FARKLI YAPILARI

2.1. Giriş

Bir insan beyni, nöron denilen yaklaşık 10^{11} hesaplama elamanından ibarettir. Nöron adı verdiğimiz temel sinir hücresi, biyolojik sinir sistemin temel yapı bloğudur. Şematik olarak Şekil 2.1'deki gibidir. Tipik bir hücre 3 temel bölgeye sahiptir; bunlar soma adı verilen hücre gövdesi, akson ve dentritler. Dentritler gövde etrafını saran bir dentrit ağacı biçiminde olup bilgiyi, iletim hattı işlevi gören aksonlar aracılığıyla nöronlardan alır. Bir aksonun uç parçası dallanır ve bu dallardan her biri komşu nöronların dentritleriyle hemen hemen temas halindeki bir kökte son bulur. Bu akson-dentrit temas noktasına snapsis denir. Snapsiste temasın oluşması bir eşik değerinin aşılmasına bağlıdır. Nöronların birbirleriyle elektriksel darbeler vasıtasıyla iletişim kurduğu yaygın kabul gören bir varsayımdır [1].



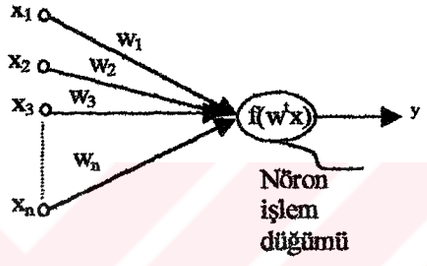
Şekil 2.1 Biyolojik nöron yapısı

Biyolojik modelin oldukça basitleştirilmiş halinin göz önüne alındığı şematik bir nöron modeli ilk olarak 1943'de Mc Culloch ve Pitts tarafından formülize edilmiştir. Biyolojik nöronla ilgili ilk önemli yaklaşımda 1958'de Frank Rosenblatt tarafından keşfedilen algılayıcı olmuştur [10].

2.2. İleri ve Geri Beslemeli YSA'lar

Mc Culloch-Pitts nöron modeli sadece 0-1 binary durumuna izin verir ve devredeki bütün nöronların çalışmasını senkron kabul eder. Bu modelde ağırlıklar ve eşik değerleri yerleştirilir ve sinyal akışı dışında nöronlar arasında iç etkileşim meydana gelmez. YSA'ları ve hesaplama algoritmaları bu modelden çok daha farklı özelliklere sahip olan nöron modellerin bir karışımını çalıştırır [10].

Standart bir nöron, tek bir çıkış ve ağırlıklı giriş bağlantılarından ibaret bir işlem elamanıdır. Ayrıca nöron bir bias değerine sahiptir. Bu değer ağırlıklı ekstra bir giriş olarak göz önüne alınarak işleme dahil edilebilir. n girişli bir yapay nöron Şekil 2.2'de görülmektedir.



Şekil 2.2 Yapay Nöron

Bu sembolik gösterim ağırlıkların bir dizisini ve nöron işlem birimini gösterir, nöron çıkış sinyali aşağıdaki ifadeyle elde edilir.

$$y = f(w^t x) \quad \text{veya} \tag{2.1a}$$

$$y = f(\sum_{i=1}^{n} w_i x_i) \tag{2.1b}$$

burada w ağırlık vektörü, x ise giriş vektörü olup aşağıdaki gibi tanımlanır.

$$w = [w_1 \quad w_2 \quad \cdots \quad w_n]^t$$
$$x = [x_1 \quad x_2 \quad \cdots \quad x_n]^t$$

Buradaki f(.) aktivasyon fonksiyonudur. Fonksiyonun değişkeni nöronun kazancıdır, bu nedenle genellikle bir net değişkeni tanımlanır ve aktivasyon fonksiyonu f(net) biçiminde kullanılır. Buradaki net değişkeni ağırlık ve giriş vektörlerinin skaler bir ürünü olarak tanımlanır.

$$Net = w^t x \tag{2.2}$$

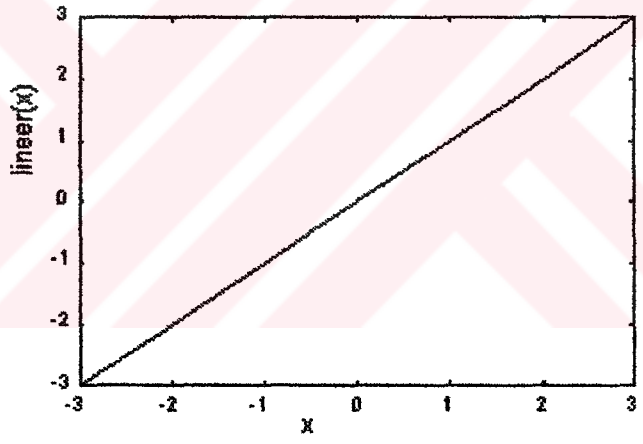
Bu net değeri biyolojik nöronların zar potansiyelinin bir benzeri olarak düşünülebilir. Bu tanımlamalarda nöronun bias değeri ayrı olarak değil, giriş vektöründe ilave bir değer olarak yer alır. Yani x vektörünün (n-1) elemanı normal girişlerdir, n. değer ise bu bias değeridir.

Şekil 2.2'de ağırlıklandırılmış girişlerin toplamına bir aktivasyon fonksiyonu uygulayan temel bir nöron yapısı görülmektedir. Ancak bu yapıdaki net tanımı tek değildir, farklı YSA yapıları için farklı net tanımları göz önüne alınabilir. Bu düğümde aşağıda belirtilen aktivasyon fonksiyonlarından biri kullanılabilir [9].

1- Lineer aktivasyon fonksiyonları
2- Non–lineer aktivasyon fonksiyonları
   i. Lojistik fonksiyonu
   ii. Hiperbolik tanjant fonksiyonu
   iii. Eşik fonksiyonu
   iv. Signum fonksiyonu vb.

Lineer aktivasyon çıkışı girişine eşittir. Sürekli çıkışlar gerektiği zaman çıkış katındaki aktivasyon fonksiyonu lineer aktivasyon fonksiyonu seçilebilir. Denklem (2.3)'deki lineer aktivasyon fonksiyonu x=[-3;3] aralığı için Şekil 2.3'de görüldüğü gibidir.

$$f(x) = x \tag{2.3}$$

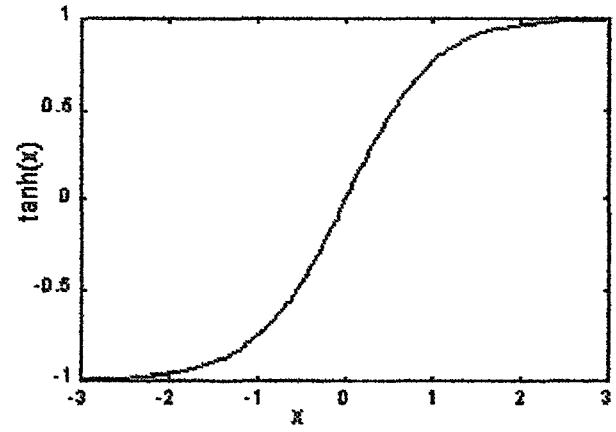


Şekil 2.3 x=[-3;3] aralığında lineer aktivasyon fonksiyonu

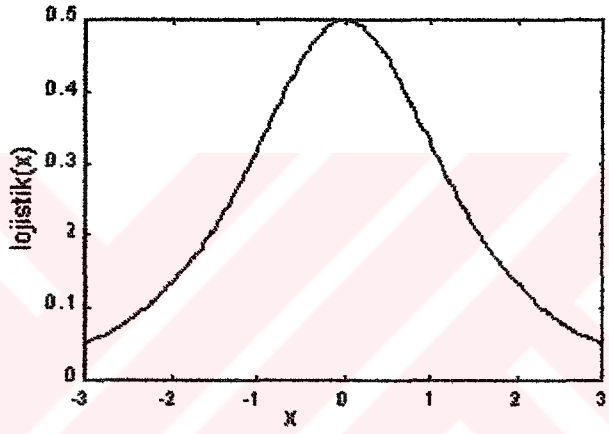
Doğrusal olmayan aktivasyon fonksiyonlarının birçok çeşidi vardır [12]. Türevi alınabilir doğrusal olmayan aktivasyon fonksiyonları geriye yayınımlı ağ çalışmalarında kullanılabilir. En çok kullanılanlar hiperbolik tanjant ve lojistik fonksiyonlarıdır. Bu fonksiyonların matematiksel ifadeleri Denklem (2.4) ve (2.5)'deki gibi olup x=[-3;3] aralığı için Şekil 2.4'de görüldüğü gibidir.

$$f(x) = \tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.4}$$

$$f(x) = lojistik(x) = \frac{1}{1 + e^{-\beta x}} \tag{2.5}$$



a)



b)

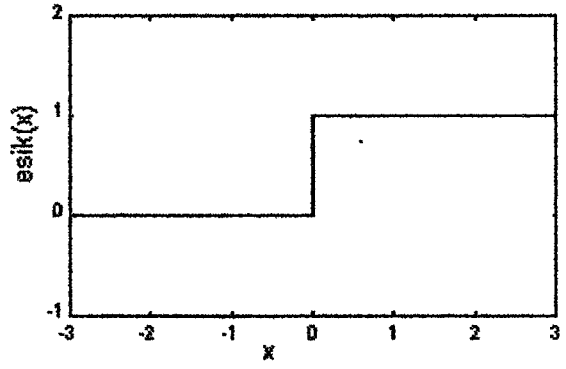
Şekil 2.4 x=[-3;3] aralığında aktivasyon fonksiyonları **a)** Hiperbolik tanjant ve **b)** lojistik (β=1)

Denklem (2.5)'deki gibi tanımlanan lojistik aktivasyon fonksiyonunda β eğim sabiti olup örneklerde daima 1 olarak düşünülür, fakat bu farklı bir değerde alabilir. Lojistik fonksiyonun çıkışı 0 ve 1 arasındadır.

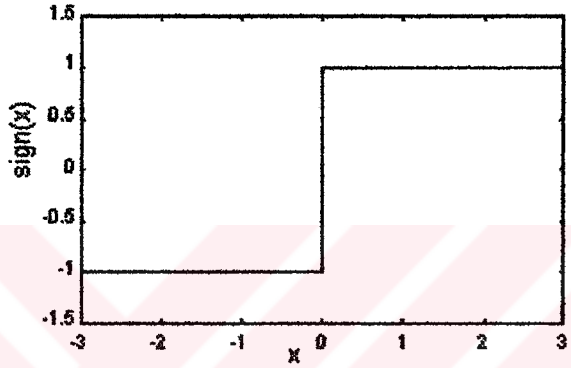
Türevi alınamayan doğrusal olmayan aktivasyon fonksiyonları genellikle algılayıcılar ve yarışmacı ağların çıkışları olarak kullanılır. Eşik ve signum olmak üzere iki tipi vardır; eşik fonksiyonunun çıkışı 0 yada 1, signumun çıkışı ise –1 yada 1 dir. Bu fonksiyonların matematiksel ifadeleri Denklem (2.6) ve (2.7)'de, grafikleri de Şekil 2.5'de görülmektedir.

eşik fonksiyonu $$f(x)=\begin{cases}1 & x\geq 0\\ 0 & x<0\end{cases} \tag{2.6}$$

signum fonksiyonu $$f(x)=\begin{cases}1 & x>0\\ 0 & x=0\\ -1 & x<0\end{cases} \tag{2.7}$$



a)

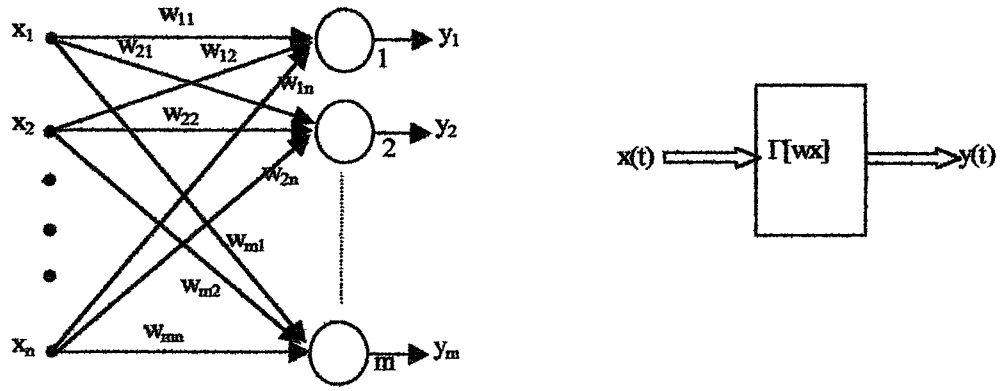


b)

Şekil 2.5 x=[-3;3] aralığında aktivasyon fonksiyonları **a)** eşik ve **b)** signum

2.2.1. İleri Beslemeli YSA

Şekil 2.6'da görülen n girişli m nöronlu ileri beslemeli YSA yapısını göz önüne alalım. Bu yapının giriş ve çıkış vektörleri sırasıyla Denklem (2.8)'deki gibidir.



a) b)

Şekil 2.6 İleri beslemeli YSA'nın **a)** yapısı ve **b)** blok diyagramı

$$x = [x_1 \quad x_2 \quad \cdots \quad x_n]^t$$

$$y = [y_1 \quad y_2 \quad \cdots \quad y_m]^t \tag{2.8}$$

w_{ij} ağırlığı i. nöron ile j. girişi bağlar. Burada (i) indisi sinyalin varış yerini (düğümünü), (j) indisi kaynak düğümünü (sinyalin geldiği yeri) gösterir. Böylece i. nöronun aktivasyon değeri Denklem (2.9)'daki gibi yazılır.

$$net_i = \sum_{j=1}^{n} w_{ij} \cdot x_j \ , \ i=1,2, ...,m \tag{2.9}$$

i=1,2 ....m değerleri için f(net$_i$) aktivasyon fonksiyonuna ilişkin doğrusal olmayan dönüşüm çıkışı Denklem (2.10)'da tanımlanır.

$$y_i = f(w_i^t x) \ , \quad i=1,2, ...,m \tag{2.10}$$

Buradaki ağırlık vektörü w_i i. çıkış düğümüne doğru yönelen ağırlıkları içerir ve Denklem (2.11)' deki gibi tanımlanır.

$$w_i = [w_{i1} \quad w_{i2} \quad \cdots \quad w_{in}] \tag{2.11}$$

Doğrusal olmayan matris operatörü Γ, giriş uzayı x'in bu devreyle çıkış uzayı y'ye dönüşümünü ifade eder.

$$y = \Gamma[w.x] \tag{2.12a}$$

Buradaki w ağırlık matrisi yada bağlantı matrisi olarak isimlendirilir.

$$w = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ w_{21} & w_{22} & \cdots & w_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m1} & w_{m2} & \cdots & w_{mn} \end{bmatrix} \tag{2.12b}$$

$$\Gamma[.] = \begin{bmatrix} f(.) & 0 & \cdots & 0 \\ 0 & f(.) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & f(.) \end{bmatrix} \tag{2.12c}$$

Γ[.] matris operatörünün diyagonalındaki f(.) doğrusal olmayan aktivasyon fonksiyonları, her nöronun *net* aktivasyon değerindeki bileşenlerini değişken kabul ederek çalışır. Her aktivasyon değeri yalnız kendi ağırlık vektörlü bir girişin skaler ürünüdür.

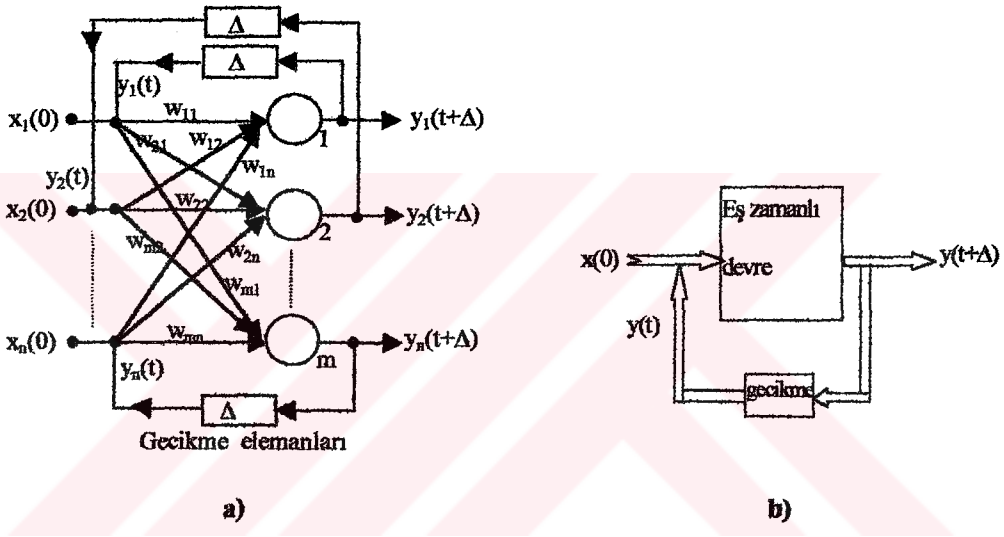
İleri beslemeli YSA'da bir giriş örneğinin denklem (2.12a)'daki gibi bir çıkış örneğine dönüşümü anlık olarak gerçekleşir, yani zaman gecikmesi yoktur. Bu nedenle Denklem (2.13)'deki gibi yazılabilir.

$$y(t) = \Gamma[w.x(t)] \tag{2.13}$$

Şekil 2.6b'de ileri beslemeli YSA'nın blok diyagramı görülmektedir. Çok katmanlı bir devre oluşturmak için ağın bu modeli kaskat bağlanabilir. İleri beslemeli ağ, x(t)'yi y(t)'ye dönüştürürken açıkça geri besleme bağlantısına sahip olmasa bile çıkış değerleri genellikle istenilen çıkış değerini sağlayan öğretici bilgiyle karşılaştırılır ve bir hata sinyali devrenin ağırlıklarını uygun hale getirmek için kullanılır.

2.2.2. Geri Beslemeli YSA

Bir geri beslemeli devre Şekil 2.6a'da görülen ileri beslemeli ağ yapısından nöronların çıkışlarını, girişlerine bağlayarak elde edilebilir ve Şekil 2.7'deki gibi oluşturulur [10].



Şekil 2.7 Geri beslemeli YSA'nın **a)** yapısı ve **b)** blok diyagramı

Geri beslemeli YSA yapısında geçmiş çıkışlar kullanılarak o anki çıkış elde edilir. Eğer bir önceki çıkış y(t) ise, bir sonraki çıkış y(t+Δ) ile gösterilebilir. Burada Δ gecikme olarak tanımlanır. İleri beslemeli ağlardakine benzer notasyonlar kullanılarak y(t +Δ) çıkışı Denklem (2.14)'deki gibi yazılır.

$$y(t+\Delta) = \Gamma[w.y(t)] \tag{2.14}$$

Bu denklem Şekil 2.7b'deki blok diyagramıyla gösterilebilir. x(t) girişine sadece bu ağı başlatmak için ihtiyaç vardır. Yani x(0)=y(0) olur. Giriş daha sonra kaldırılabilir ve sistem t>0 için bağımsız kalabilir.

Tek katmanlı geri beslemeli ağların iki temel kategorisi vardır [10].

a- Ayrık zamanlı ağlar

b- Sürekli zamanlı ağlar

Ayrık zamanlı devrelerde kolaylık için gecikme elemanı (örnekleme periyodu), Δ=1 seçilirse, bu durumda birim gecikme kabul edilir ve örnekleme zamanıyla doğal sayılar olarak seçilmiş olan indisler eşitlenmiş olur. Ayrık zamanlı bir YSA için Denklem (2.14), Denklem (2.15a)'daki gibi düzenlenir.

$$y^{k+1} = \Gamma[w.y^k] \quad k=1,2,...,K \tag{2.15a}$$

Burada; k, ayrık zaman anlarını gösterir. Şekil 2.7'deki ağa k+1 anındaki cevap k=0'dan başlayarak devrenin tüm geçmişine bağlı olduğundan tekrarlı denir. k+1 anındaki cevap açık olarak yazılırsa

$$y^{k+1} = \Gamma[w\Gamma[w\Gamma[.....\Gamma[w.x^0].....] \tag{2.15b}$$

olur. Burada çıkış, geçmişteki cevapların tekrarlı bir serisi biçimindedir. Bu tekrarlı sistemler, tipik olarak datanın ayrık bir gösterimiyle çalışırlar ve bünyelerinde eşik aktivasyon fonksiyonlu nöronlar içerirler. Ayrık zaman girişli ve bir ayrık zaman gösterimli bir sisteme otomatik, yani kendinden hareket eden sistem denir. Denklem (2.15), k=1,2,...,K anlarında devrenin durumlarını ve durum geçiş dizisini elde edilmesini tanımlarlar. Devre 0 anında x^0'da başlatılır ve muhtemel bir denge noktası buluncaya kadar k=1,2,...,K için y^k durum geçişlerine gider. Bu denge noktası tek veya sınırlı sayıda olabilir.

Şekil 2.7'deki gecikme elemanları yerine uygun sürekli zaman gecikme elemanları konularak sürekli zamanlı YSA devresi elde edilebilir. Böyle gecikme elemanları enerji depolayan elemanlarla gerçekleştirilebilir. Sürekli zamanlı YSA'nın giriş ve çıkışları arasındaki bağıntı Denklem (2.14)'deki gibi olacaktır. Δ yerine enerji depolayan elemanların oluşturduğu gecikme zamanı (faz farkı veya zaman sabiti) alacaktır.

2.3. Radyal Tabanlı Fonksiyon Ağları

2.3.1. Giriş

Denetimli bir YSA'nın tasarımı, farklı metotların karışımında aranabilir. Yüksek boyutlu bir uzayda YSA tasarımı, bir eğri uydurma problemi olarak ele alınabilir. Bu yaklaşıma göre öğrenme, belli bir uyum ölçütüne göre eğitim bilgisiyle en iyi uyumu gösteren yüzeyi bulmaya eşdeğerdir. Ara değer bulma işleminin düzenlenmesi anlamında, bu bakış açısı RTFA'nın oluşturulması fikrinin temelidir [9, 13].

En temel biçimdeki bir RTFA üç katmanlıdır. İkinci katman çok katmanlı YSA'lardakinden farklı olan, yeterli boyutta bir gizli katmandır. Üçüncü katman, girişe uygulanmış uyartın örneklerine ağın cevabını sağlayan çıkış katmanıdır. Giriş uzayından gizli birim uzayına doğrusal olmayan bir dönüşüm vardır. Doğrusal olmayan yüksek boyutlu bir uzaydaki sınıflandırma probleminin, düşük boyutlu uzaydakine göre doğrusal olarak ayrıştırılabilmesi daha muhtemeldir. Bu, RTFA'larda gizli birim uzayının yüksek boyutlu olmasına yol açar. Gizli birimlerin merkezlerinin ayarlanabilir yapılmasıyla bu problem giderilebilir [9].

RTFA'lar doğrusal olmayan fonksiyonların tahmini için uygundur. Gizli katman birimleri genellikle gaussian olan radyal tabanlı fonksiyonlara sahiptir. RTFA'ların gerçekleştirilmesinde klasik yaklaşım, giriş bilgisinin bazı özellikleri temelinde gizli birimlerin sayısını belirlemek (yani, gaussian fonksiyonların merkez ve genişliklerini) ve doğrusal en küçük kareler metodu kullanarak, çıkış ve gizli birimlerin bağlantı ağırlıklarının değerlerini bulmaktır [14, 15].

2.3.2. Genelleştirilmiş RTFA'lar

Klasik yaklaşımda, RTFA'nın birinci katmanında, giriş vektörünün boyutu p kadar giriş düğümü vardır. Gizli katman olan ikinci katmanda, doğrudan tüm giriş düğümlerine bağlı olan, doğrusal olmayan gizli birimler mevcuttur. Bu birimlerin sayısı data sayısına eşittir ve aktivasyon fonksiyonları Green fonksiyonlarıyla tanımlanır. Bundan dolayı i. gizli birim çıkışı Green fonksiyonunun değeridir. Üçüncü katman ise, çıkış katmanıdır ve gizli katmana bağlı, giriş bilgilerinin lineer ağırlıklandırılmış toplamını çıkış olarak sağlayan tek bir lineer birimden oluşur. Şekil 2.8'de klasik yaklaşıma göre bir RTFA'nın yapısı görülmektedir.

Green fonksiyonları matrisel denklemler için ters matrisin işlevine benzer olarak, lineer bir differansiyel denklem için aynı işlevi görürler. $G(x;x_i)$, gösterimi, x_i merkezli bir Green fonksiyonunu ifade eder.

$$P^*P.G(x;x_i)=0 \tag{2.16}$$

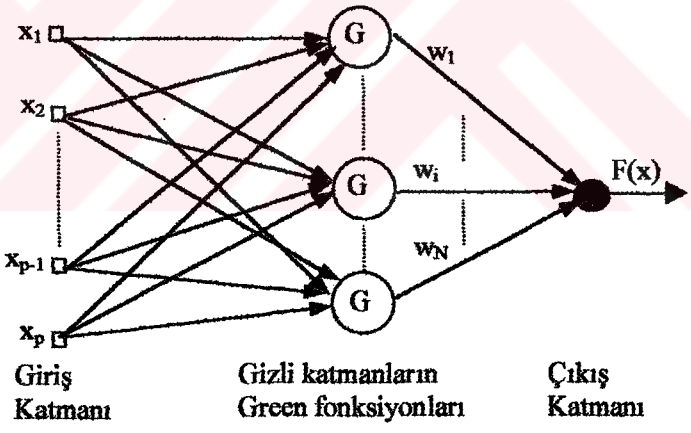
Denklem (2.16)'daki eşitliği, x_i noktası hariç her x noktası için sağlayan fonksiyonlar Green fonksiyonu olarak bilinir. Burada; P: bir lineer differansiyel operatörü, P^*: P'nin adjointini ve P^*P: self-adjoint differansiyel operatörü göstermektedir. Eğer P operatörü hem doğrusal hem de eksenel olarak sabit ise $G(x;x_i)$, sadece Euclean norma bağlıdır. Bu durumda $G(x;x_i)$, radyal tabanı bir fonksiyondur.

$$G(x;x_i)=G(\|x-x_i\|) \tag{2.17}$$

Şekil 2.8'de görülen klasik RTFA'nın çıkışı Denklem (2.18)'deki gibi bulunur.

$$F(x)=\sum_{i=1}^{N} w_i.G(\|x-x_i\|) \tag{2.18}$$

Şekil 2.8'deki RTFA'nın gizli birim sayısı giriş data sayısı N'e eşittir. N'in büyük olması hesapsal güçlüklere neden olur. Özellikle, ağın lineer ağırlıklarının hesabı, NxN boyutlu bir matrisin tersinin alınmasını gerektirir.



Şekil 2.8 Klasik RTFA

Bu hesapsal güçlüklerin üstesinden gelmek için, ağın karmaşıklığını düzenlenmiş çözüme indirgeyecek olan bir yaklaşım kullanılır. Faydalanılan yaklaşım, daha düşük boyutlu bir uzaydaki optimal alt çözümü araştırmaktır. Bu, değişimli problemlerde, Galerkin metodu olarak bilinen standart bir teknik kullanılarak yapılır. Bu tekniğe göre, yakınsanmış $F^*(x)$ çözümü sonlu bir tabanda genişletilir,

$$F^*(x)=\sum_{i=1}^{M} w_i.\varphi_i(x). \tag{2.19}$$

Burada; $\{\varphi_i(x)\,|_{i=1,2,...M}\}$, taban fonksiyonlarının yeni bir dizisidir. (Bu fonksiyonların lineer bağımsız oldukları kabul edilir). Genellikle, taban fonksiyonlarının sayısı data noktalarının sayısından daha azdır M≤N ve w_i ağırlıkların yeni bir dizisi ile bilinen radyal tabanlı fonksiyonlar Denklem(2.20)'deki gibi yazılabilir.

$$\varphi_i(x) = G(\|x - t_i\|) \quad i=1,2,...,M \tag{2.20}$$

Burada hesaplanmış olan merkezlerin dizisi $\{t_i\,|_{i=1,2,...M}\}$ formundadır. Taban fonksiyonlarının bu özel seçimi, sadece M=N ve $t_i = x_i$, i=1,2,...N durumunu garanti eder. Bu nedenle, Denklem (2.20), (2.19)'da yerine konursa

$$F^*(x) = \sum_{i=1}^{M} w_i.G(\|x - t_i\|). \tag{2.21}$$

elde edilir. Bu durumda, Denklem (2.22)'deki yeni amaç fonksiyonu minimize edilir.

$$\wp(F^*) = \sum_{i=1}^{N}\left(d_i - \sum_{j=1}^{M} w_j.G(\|x_i - t_j\|)\right)^2 + \lambda\|PF^*\|^2 \tag{2.22}$$

Denklem (2.22)'nin sağ tarafındaki ilk terim, Euclidian normunun karesi şeklinde ifade edilebilir $\|d\text{-}G.w\|^2$. Burada

$$d = [d_1, d_2, \ldots, d_N]^T \tag{2.23}$$

$$G = \begin{bmatrix} G(x_1;t_1) & G(x_1;t_2) & \cdots & G(x_1;t_M) \\ G(x_2;t_1) & G(x_2;t_2) & \cdots & G(x_2;t_M) \\ \vdots & \vdots & & \vdots \\ G(x_N;t_1) & G(x_N;t_2) & \cdots & G(x_N;t_M) \end{bmatrix} \tag{2.24}$$

$$w = [w_1, w_2, \ldots, w_M]^T \tag{2.25}$$

olup, d: istenilen cevap vektörünü, G: Green fonksiyonları matrisini ve w ağırlık vektörünü göstermektedir. Denklem (2.21)'deki F^* fonksiyonu, P operatörü için Green fonksiyonlarının lineer kombinezonudur [9]. Bu nedenle, Denklem (2.22)'nin sağ tarafındaki ikinci terim

$$\begin{aligned} \|PF^*\|^2 &= (PF^*, PF^*)_H \\ &= \left[\sum_{i=1}^{M} w_i.G(x;t_i), P^*P\sum_{i=1}^{M} w_i.G(x;t_i)\right]_H = \left[\sum_{i=1}^{M} w_i.G(x;t_i), \sum_{i=1}^{M} w_i.\delta_{t_i}\right]_H \\ &= \sum_{j=1}^{M}\sum_{i=1}^{M} w_j.w_i.G(t_j;t_i) \\ &= w^T.G_0.w \end{aligned} \tag{2.26}$$

olur. Burada G_0, MxM boyutlu simetrik bir matristir ve

$$G_0 = \begin{bmatrix} G(t_1;t_1) & G(t_1;t_2) & \cdots & G(t_1;t_M) \\ G(t_2;t_1) & G(t_2;t_2) & \cdots & G(t_2;t_M) \\ \vdots & \vdots & & \vdots \\ G(t_M;t_1) & G(t_M;t_2) & \cdots & G(t_M;t_M) \end{bmatrix} \tag{2.27}$$

olarak tanımlanır. Böylece, w ağırlık vektörüne göre Denklem (2.22)'nin minimum yapılması için, $M \approx \frac{N}{2} + \frac{\alpha}{2}\sqrt{N}$ boyutlu bir uzaydaki ayrışabilir N datanın asimtotik olasılığından, Denklem (2.28)'deki ifade elde edilir.

$$(G^T.G + \lambda.G_0).w = G^T.d \tag{2.28}$$

λ düzenleme parametresinin sıfıra yaklaşmasıyla, w ağırlık vektörü

$$w = G^+.d, \quad \lambda=0 \tag{2.29}$$

olup,

$$G^+ = \left(G^T.G\right)^{-1}.G^T \tag{2.30}$$

olarak tanımlanır.

Ağırlıklandırılmış norm

Denklem (2.22)'nin tahmini çözümündeki norma, genellikle Euclidean norm denir. Ancak x vektörünün belirli elemanları farklı durumlara ait olduğunda, genel bir ağırlıklandırılmış norm düşünmek daha uygundur.

$$\|x\|_c^2 = (Cx)^T.(Cx) = x^T.C^T.C.x \tag{2.31}$$

Burada; C: pxp boyutlu norm ağırlıklandırma matrisi, p: giriş vektörünün boyutudur. C ağırlıklandırma matrisine ilişkin, 3 özel durum aşağıdaki gibi tanımlanır:

1-*C* matrisi, standart Euclidean norm elde edilmesi için birim matrise eşittir,

$$\|x\|_c^2 = \|x\|^2 \qquad C=I. \tag{2.32}$$

2-*C* matrisi diyagonal bir matristir. Bu durumda; diyagonal elemanlar her giriş koordinatına özel bir ağırlık atar,

$$\|x\|_c^2 = \sum_{k=1}^{p} C_k^2.x_k^2. \tag{2.33}$$

Burada; x_k: x giriş vektörünün k. elemanıdır ve C_k: C matrisinin k. diagonal elemanıdır.

3-C matrisi diyagonal olmayan bir matristir. Bu durumda ağırlıklandırılmış norm ikinci derece formdadır.

$$\|x\|_c^2 = \sum_{k=1}^{p} \sum_{l=1}^{p} a_{kl} . x_k . x_l . \tag{2.34}$$

Burada a_{kl}: $C^T.C$ matrisinin kl. elemanıdır.

Ağırlıklandırılmış norm tanımı kullanılarak, Denklem (2.22)'de verilen ifade daha da genelleştirilerek tekrar yazılabilir.

$$F^*(x) = \sum_{i=1}^{M} w_i . G\left(\|x - t_i\|_{c_i}\right) \tag{2.35}$$

Ağırlıklandırılmış normun kullanımı, aşağıdaki Gaussian Radyal tabanlı fonksiyonlar bağlamında da doğrulanabilir.

$$G\left(\|x - t_i\|_{c_i}\right) = \exp\left[-(x - t_i)^T . C_i^T . C_i . (x - t_i)\right]$$

$$= \exp\left[-\frac{1}{2} . (x - t_i)^T . \Sigma_i^{-1} . (x - t_i)\right]. \tag{2.36}$$

Burada, C_i: norm ağırlıklama matrisini ve $G(\|x\text{-}t_i\|_{ci})$, t_i'de merkezli radyal tabanlı bir fonksiyonu gösterir. Buradaki Σ_i^{-1} ters matrisi Denklem (2.37)'deki gibi tanımlanır.

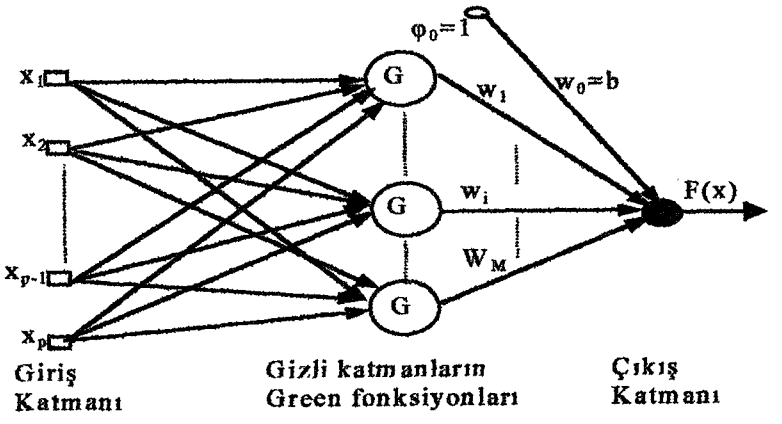
$$\frac{1}{2} \Sigma_i^{-1} = C_i^T . C_i \tag{2.37}$$

Denklem (2.36), Σ_i kovaryans matrisli ve t_i merkez vektörlü çok değişimli bir gaussian dağılımı ifade eder [14, 15]. Denklem (2.22)'de verilen tahmin problemi için bu çözüm, Şekil 2.9'da görülen yapıya sahip genelleştirilmiş RTFA için bir çatı sağlar. Bu ağ, çıkış birimine uygulanan +1 değerli bias gerektirir.

Yapısal terimlerde Şekil 2.9'daki genelleştirilmiş RTFA Şekil 2.8'deki klasik RTFA'ya benzer. Ancak bu ağlar arasında iki önemli fark vardır [9].

1- Şekil 2.9'daki genelleştirilmiş RTFA'nın gizli katmanlarındaki birim sayısı M dir. M, eğitim için var olan örneklerin sayısı N'den daha küçük sıradan bir sayıdır. Şekil 2.8'deki klasik RTFA'daki birimlerin sayısı tam olarak N'e eşittir.

2- Şekil 2.9'daki genelleştirilmiş RTFA'nın bilinmeyen parametreleri çıkış katmanıyla ilişkili lineer ağırlıklar, radyal taban fonksiyonlarının merkezlerinin yerleri ve gizli katmana ilişkin norm ağırlıklandırma matrisidir. Şekil 2.8'deki klasik RTFA'da merkezleri eğitim data noktaları olan Green fonksiyonlarının bir dizisiyle tanımlanan gizli katman aktivasyon fonksiyonları bilinir, ağın bilinmeyen parametreleri sadece, çıkış katmanının lineer ağırlıklarıdır.



Şekil 2.9 Genelleştirilmiş RTFA yapısı

2.3.3.RTFA ile Çok Katmanlı Algılayıcıların Karşılaştırılması

RTFA ve Çok Katmanlı Algılayıcılar (ÇKA), doğrusal olmayan çok katmanlı İYSA olup, her ikisi de genel hesaplayıcılardır. ÇKA'nın taklit etme kabiliyetleri daima bir RTFA'da vardır [9]. Bu iki ağı bir birinden ayıran önemli özellikler şunlardır:

1- Bir RTFA tek bir gizli katmana sahiptir. Bir ÇKA ise bir veya daha fazla gizli katmana sahip olabilir.

2- ÇKA'ın gizli veya çıkış katmanında yerleşmiş olan hesaplama düğümleri genel bir nöron modeli kullanırlar. RTFA'ın gizli katmanındaki hesaplama düğümleri ise, ağın çıkış katmanındakilerden tamamen farklıdır ve farklı bir amaca hizmet ederler.

3- RTFA'nın gizli katmanı doğrusal değildir oysa çıkış katmanı doğrusaldır. Sınıflandırıcı olarak kullanılan bir ÇKA'nın çıkış ve gizli katmanları genelde tümüyle doğrusal olmayan birimlerden oluşur. Ancak ÇKA doğrusal olmayan bir tanımlama problemini çözmek için kullanıldığında, çıkış için doğrusal bir katman genellikle tercih edilir.

4- RTFA'da her gizli birimin aktivasyon fonksiyonu, o birimin merkezi ve giriş vektörü arasındaki euclidean normu argüman olarak alır. ÇKA'da ise her gizli birim aktivasyon fonksiyonu, o birimin ağırlık vektörü ve giriş vektörünün çarpımını argüman kabul eder.

5- ÇKA'lar doğrusal olmayan giriş–çıkış dönüşümüne global tahminler oluştururlar. Dolayısıyla, eğitim datasının küçük veya var olmadığı giriş uzayı bölgelerinde genelleme yeteneklerine sahiptirler. Buna karşın, yerel doğrusal olmamanın üstel olarak bozulmasını kullanan RTFA (yani gaussian fonksiyonlar), doğrusal olmayan giriş-çıkış dönüşümüne lokal tahminler oluştururlar. Bu nedenle, hızlı öğrenme

yeteneklidirler ve eğitim datasının sunumu oranında duyarlılık azaltılır. Bir çok durumda, bir dönüşüm istenilen doğrulukta yaklaştırmak için, uygun giriş uzayını kapsayacak radyal tabanlı fonksiyonların sayısı oldukça fazla olur.

2.3.4. Karma Modeller

Gaussian dağılımların karışımı olan Denklem (2.35)'deki ifade karma modellerle yakın ilişkilidir. Olasılık dağılımlarının karışımları, oldukça değişik uygulamalarda model olarak yaygınca kullanılmaktadır. Karma modele göre, sonlu sayıda dağılımların karması olan, bir S üst dağılım vardır. Bu dağılımlar S_1, S_2,.... S_n ve bu karma π_1, π_2,...., π_n oranlarıyla belirlenir;

$$\sum_{i=1}^{n} \pi_i = 1 \text{ ve } \pi_i \geq 0 \quad \text{tüm i'ler için} \tag{2.38}$$

S'deki gelişi güzel bir x vektörünün olasılık yoğunluk fonksiyonu Denklem (2.39)'daki gibi ifade edilir;

$$f(x;\varphi) = \sum_{i=1}^{n} \pi_i . f_i(x;\theta) . \tag{2.39}$$

Burada; $f_i(x;\theta)$: S_i'ye ilişkin olasılık yoğunluk fonksiyonudur ve θ: bilinmeyen tüm ilgili parametrelerin vektörünü gösterir. φ vektörü, θ ve karma oranlar π_1, π_2,...., π_n'den oluşur. Problem, gözlem vektörlerinin α_1, α_2,....., α_N bir dizisini vererek ve bilinmeyen parametreleri tahmin etmektir. Böylece karma model probleminin aslında, Denklem (2.35) ifadesindeki Σ_i, t_i ve w_i 'yi bulma problemine benzediği görülür.

2.3.5. Öğrenme Yöntemi

Bir RTFA'nın öğrenme işlemi, aşağıdaki gibi ifade edilebilir. Çıkış katmanındaki ağırlıklar, gizli birimlerin doğrusal olmayan aktivasyon fonksiyonlarına göre farklı bir zaman ölçeğinde değişmek eğilimindedir. Bu nedenle; gizli birimlerin aktivasyon fonksiyonları doğrusal olmayan bir optimizasyon stratejisine göre yavaş olarak gelişirken, çıkış katmanının ağırlıkları doğrusal bir optimizasyon stratejisi ile kendi kendilerini hızla ayarlarlar. Bu nedenle, RTFA'nın gizli ve çıkış katmanlarının eğitimi için farklı optimizasyon metotları kullanmak ve hatta bu katmanlar için farklı zaman ölçütleriyle çalışmak mantıklı olur.

Bir RTFA'ın tasarımında takip edilebilecek farklı öğrenme stratejileri vardır. Ağın radyal tabanlı fonksiyonlarının merkezlerinin nasıl belirlendiğine bağlı olarak, üç farklı yaklaşım tanımlanabilir.

1. Merkezlerin Rasgele Seçimi

En basit yaklaşım, gizli birimlerin aktivasyon fonksiyonlarını tanımlayan radyal tabanlı fonksiyonları sabit kabul etmektir. Merkezlerin yerleri eğitim data dizisinden rasgele seçilebilir. Bu eğitim datası, temsilci tarzında dağılmasını sağlayan akla uygun bir yaklaşım olarak düşünülebilir. Radyal tabanlı fonksiyonlar için, standart sapması merkezlerin dağılımına göre sabit olan isotropik bir gaussian fonksiyon kullanılabilir. t_i'de merkezlenmiş radyal tabanlı bir fonksiyon Denklem (2.40)'daki gibi tanımlanır,

$$G\left(\|x-t_i\|^2\right)=\exp\left(-\frac{M}{d^2}\|x-t_i\|^2\right), \quad i=1,2,..M. \tag{2.40}$$

Burada; M merkezlerin sayısını ve d seçilen merkezler arasındaki maksimum uzaklığı göstermektedir. Sonuçta, tüm gaussian radyal tabanlı fonksiyonların standart sapması σ'da sabitlenir.

$$\sigma=\frac{d}{\sqrt{2M}} \tag{2.41}$$

Bu seçim gaussian fonksiyonların aşırı sivri ve yassı olmamasını sağlar. Bu yaklaşımda ayarlanacak parametreler sadece ağın çıkış katmanındaki lineer ağırlıklardır. Burada lineer ağırlıklar Pseudoinvers metodu kullanılarak aşağıdaki gibi elde edilir.

$$w=G^{+}.d \tag{2.42}$$

$$G=\{g_{ji}\} \tag{2.43}$$

$$g_{ji}=\exp\left(-\frac{M}{d^2}\|x_j-t_i\|^2\right), \quad j=1,2,..N,\ i=1,2,..M \tag{2.44}$$

Burada x_j, eğitim dizisinin j. giriş vektörüdür.

NxM boyutlu gerçel bir G matrisinin Pseudoinversinin hesaplanması için aşağıdaki tekil değer ayrışımı kullanılır.

$$U=\{U_1,U_2,...U_N\} \text{ ve } V=\{V_1,V_2,...V_M\}$$

$$U^TGV=diag(\sigma_1,\sigma_2,...\sigma_K), \quad K=min(N,M) \tag{2.45}$$

$$\sigma_1\geq\sigma_2\geq...\geq\sigma_K>0 \text{ ve } U, V \text{ ortagonal matrislerdir.}$$

U matrisinin kolon vektörlerine, G'nin sol tekil vektörleri ve V matrisinin kolon vektörlerine G'nin sağ tekil vektörleri denir. $\sigma_1,\sigma_2,...\sigma_K$'ya G matrisinin tekil değerleri denir. Tekil değer ayrışımı teoremine göre, G'nin, MxN'lik Pseudoinversi Denklem (2.46)'daki gibi tanımlanır.

$$G^{+}=V.\Sigma^{+}U^{T}. \tag{2.46}$$

Burada; Σ^+, G'nin tekil değerleri terimleriyle tanımlanmış NxN boyutlu bir matristir.

$$\Sigma^+ = diag(\frac{1}{\sigma_1},\frac{1}{\sigma_2},\ldots,\frac{1}{\sigma_K},0,\ldots 0) \tag{2.47}$$

2. Merkezlerin Kendi Kendini Organize Ederek Seçimi

İkinci yaklaşımda radyal tabanlı fonksiyonların merkezleri, kendi kendilerini organize etme metodu ile lineer ağırlıklar ise denetimli bir öğrenme kuralı kullanılarak hesaplanır [15]. Gizli birimlerin merkezlerinin kendi kendini organize eden seçimi için, standart k en yakın komşuluk kuralı uygulanır. Bu kural, k en yakın örnek içinden en sık ifade edilmiş nitelemeyi ona atayarak bir x giriş vektörünü sınıflandırır. Çıkış katmanındaki lineer ağırlıkların hesabı ise karelerin en küçük ortalaması (LMS) algoritması gibi bir öğrenme kuralı ile bulunur, RTFA'daki gizli birimlerin çıkışları LMS algoritmasına giriş olarak alınır [13].

3. Merkezlerin Denetimli Seçimi

Üçüncü yaklaşımda radyal tabanlı fonksiyonların merkezleri ve ağın diğer tüm bağımsız parametreleri denetimli bir öğrenme işlemine tabii tutulur. Bu genelleştirilmiş RTFA olarak adlandırılır ve eğitimi için genelleştirilmiş LMS algoritması olan eğim iniş işlemi uygulanır. Böyle bir öğrenme prosedürünün gelişiminde birinci adım, amaç fonksiyonunun anlık değerini tanımlamaktır.

$$\wp = \frac{1}{2}\sum_{j=1}^{N} e_j^2. \tag{2.48}$$

Burada; N: öğrenme işlemini gerçekleştirmek için kullanılan eğitim örneklerinin sayısını ve e_j: hata sinyalini göstermektedir.

$$e_j = d_j - F^*(x_j) = d_j - \sum_{j=1}^{M} w_i . G\left(\left\| x_j - t_i \right\|_{c_i}\right) \tag{2.49}$$

$\wp$'yi minimum yapan w_i,t_i ve Σ_i^{-1} parametrelerini bulmak gerekir. Öğrenme işlemineilişkin bilgiler Tablo 2.1'de özetlenir. Bu işlemle ile ilgili önemli bilgiler aşağıda sıralanmıştır.

- $\wp$ amaç fonksiyonu, doğrusal w_i parametrelerine göre iç bükeydir. Buna karşın t_i merkezleri ve Σ_i^{-1} matrisine göre dış bükey dir.
- w_i,t_i ve Σ_i^{-1} için güncelleme denklemlerine sırasıyla η_1, η_2, η_3 öğrenme parametreleri atanır.

- Bir RTFA için Tablo 2.1'de tanımlanmış eğim iniş prosedürü, geriye yayılım algoritmasından farklı olarak hata geriye yayılımını içermez.
- $\frac{\partial \wp}{\partial t_i}$ gradyant vektörü, kümelendirme etkisine benzer bir etkiye sahiptir.

Tablo 2.1 Genelleştirilmiş RTFA için güncelleme ifadeleri

1. Doğrusal ağırlıklar (çıkış katmanı) $\frac{\partial \wp(n)}{\partial w_i(n)} = \sum_{j=1}^{N} e_j(n).G(\|x_j - t_i(n)\|_{c_i})$ $w_i(n+1) = w_i(n) - \eta_1 . \frac{\partial \wp(n)}{\partial w_i(n)} \qquad i=1,2,...M$
2. Merkezlerin yerleri (gizli katman) $\frac{\partial \wp(n)}{\partial t_i(n)} = 2w_i(n).\sum_{j=1}^{N} e_j(n).G'(\|x_j - t_i(n)\|_{c_i}).\Sigma_i^{-1}.[x_j - t_i(n)]$ $t_i(n+1) = t_i(n) - \eta_2 . \frac{\partial \wp(n)}{\partial t_i(n)} \qquad i=1,2,...M$
3. Merkezlerin genişliği (gizli katman) $\frac{\partial \wp(n)}{\partial \Sigma_i^{-1}(n)} = -w_i(n).\sum_{j=1}^{N} e_j(n).G'(\|x_j - t_i(n)\|_{c_i}).Q_{ji}(n)$ $Q_{ji}(n) = [x_j - t_i(n)].[x_j - t_i(n)]^T$ $\Sigma_i^{-1}(n+1) = \Sigma_i^{-1}(n) - \eta_3 . \frac{\partial \wp(n)}{\partial \Sigma_i^{-1}(n)}$

2.4. Modüler Yapay Sinir Ağları

2.4.1. Giriş

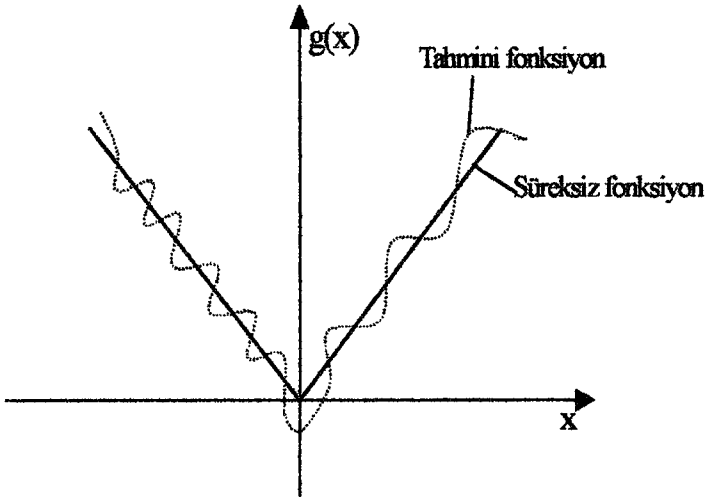
YSA'lar da organizasyonun hiyerarşi aşamaları aşağıdaki gibi sınıflandırılabilir. En temel aşamada snapsisler yer alır ve bunları nöronlar, nöron katmanları ile ağın kendisi takip eder. Bu açıdan bakıldığında, ağların yapısı, nöronlar veya nöron katmanları halinde modüler bir özelliğe sahiptir [16]. Modüler yapay sinir ağlarının (MYSA) dizaynı, farklı yapıya sahip ağların bir araya getirilmesi ve her bir yapının avantajını ortaya çıkaracak bir öğrenme algoritmasının geliştirilmesinden oluşur [17]. Bu bölümde, MYSA'ların, denetimli ve denetimsiz öğrenme algoritmalarının ortak kullanımına dayalı bir sınıfı tanıtılacaktır.

Yaklaşım problemi göz önüne alınarak MYSA'ların dizaynı için temel oluşturulabilir. İstenilen giriş çıkış dönüşümü lokal bir metot kullanılarak yaklaştırılabilir. Böyle bir modele örnek olarak RTFA gösterilebilir. Tek bir işi öğrenmek, genellikle az eğitim örneği gerektirdiği için, lokal metotların kullanımı, hızlı öğrenme ve bundan dolayı gerçek zamanlı çalışma kabiliyeti avantajı sağlar [9]. Buna karşın lokal metotların dezavantajı hafızayı yoğun olarak kullanmalarıdır. Alternatif olarak yaklaşım problemi, global bir metot kullanılarak gerçekleştirilebilir. Çok katmanlı ağların geriye yayılım algoritmasıyla eğitilmesi bu modele bir örnektir. Global metotlarda, daha az bilgi depolanır ve daha iyi genelleme performansı sağlanır. Ancak bu metotlarda, öğrenme işlemi oldukça yavaş gerçekleşir [16]. Lokal ve global metotların avantajları, ilk defa 1980'li yılların ortasında Hinton ve Jacobs [18] tarafından, giriş çıkış dönüşümünün temel yapısına yerleşen modüler bir yapı kullanılarak birleştirilmiştir. Bu bölümde 1991'de ilk defa Jacops ve Jordan tarafından önerilen MYSA sınıfı ele alınmıştır [9, 19].

Modüler yaklaşımın en önemli özelliklerinden bir tanesi süreksiz fonksiyonlara büyük bir kararlılıkla yaklaşmasıdır. Örnek olarak Denklem (2.50)'deki bir boyutlu süreksiz g(x) fonksiyonunu göz önüne alalım.

$$g(x) = \begin{cases} x & x > 0 \\ -x & x \leq 0 \end{cases} \tag{2.50}$$

Eğer bu fonksiyonun yaklaşımı için bir tam bağlantılı ağ kullanılırsa; yaklaştırılan sonuç Şekil 2.10'da kesikli çizgilerle görüldüğü gibi süreksiz fonksiyon civarında kararsız bir davranış gösterebilir. Böyle durumlarda yaklaştırılacak fonksiyonun bölünmesi tercih edilir ve her ayrı parçayı yaklaştırmak için bir modül ağ kullanılır [9, 20].



Şekil 2.10 Süreksiz fonksiyon

Modüler yaklaşımın kullanımı, nörobiyolojik temelde de doğrulanabilir. Modülerite omurgalı hayvanların sinir sisteminde var olan bir özelliktir. Görüntü algılaması bunun en belirgin örneğidir. Görme, yüksek karmaşık hesaplamalar gerektirir. Sinir sistemi, tıpkı bir mühendisin karmaşık bir sistemi dizayn ederken yapacağı gibi, bu hesapları aşağıdaki biçimde parçalara ayırır.

1- Farklı alt işler için görsel sistemde ayrı birimler oluşturulur. Bu birimler, hesaplamanın belli tipleri için optimize edilmiş olan nöron yapılarından oluşur.
2- Ayrı birimler, visual korteksin altıncı alanının iç yapısıyla örneklendiği gibi birkaç kez kopya edilir.
3- Birimler arasındaki bilginin etkin dolaştırılması ve düzenlenmesi sağlanır.

İnsan beyni, karmaşık hesapları az bir uğraşla yapabildiği basit hesapların bileşkesi olarak gerçekleştirebilir. Bu hesap tekniğinde modülerite ilave bir değişken olarak görülebilir. Bu değişken, karmaşık işlemleri yapabilen daha yüksek dereceli hesaplama birimlerinin oluşumunu mümkün kılar.

2.4.2. Modüleritenin Temel Fikirleri

Eğer ağla yapılacak hesaplama, birbiriyle iletişimsiz ayrık biçimde çalışan iki veya daha fazla parçaya ayrıştırılabiliyorsa ağın modüler olduğu söylenir. Modüllerin çıkışlarına, bilginin modüllere geri beslenmesine müsaade etmeyen bir birleştirme birimi aracılık eder. Birleştirme birimi, modül çıkışlarının sistemin son çıkışına nasıl ekleneceğine ve hangi modülün hangi eğitim örneklerini öğreneceğine karar verir. Bu nedenle, modülerite karmaşık bir hesaplama işini daha basit parçalara bölen ve sonra bu parçalara ait çözümleri birleştirerek gerçekleştiren

bölme ve alma prensibi olarak görülebilir. MYSA, bağlantılı bir yöntem ile denetimli ve denetimsiz tarzdaki öğrenme algoritmalarını birleştirir.

Denetimli öğrenme, ağın farklı modüllerini eğitmek için, istenilen çıkışları sağlayan dış bir eğiticiye ihtiyaç duyar. Bununla birlikte, eğitmen ağın hangi modülünün hangi istenilen cevabı üreteceğini belirlemez, bu atama denetimsiz öğrenme algoritmasının görevidir [21].

Denetimsiz öğrenme için istenilen cevabı doğru olarak üretme amacıyla birbirleriyle yarışan modüller kullanılır. Birleştirme birimi, doğruluğun sağlanması için farklı modüller arasında kılavuzluk rolüne sahiptir. Dolayısıyla, ağın modülleri giriş uzayının farklı bölgelerini öğrenerek uzmanlaşır. Yarışmacı öğrenme algoritmasında, her farklı bölge için istenilen cevaba en yakın cevabı üreten modül kazanan ve diğerleriyse kaybedenlerdir. Ayrıca her bir modül öğrenme kabiliyetiyle orantılı olarak eğitim bilgisinin bir miktarını alır. Bu kazanan modülün kaybedenden daha fazla eğitim bilgisi aldığını ifade eder.

Denetimsiz öğrenme eğitim parametrelerini doğru eğitmek için pozitif geri besleme gerektirir. Bu nedenle MYSA'larda yarış pozitif geri besleme etkisiyle yapılır. Belli bir modül eğitim bilgisinin bir bölümünü iyi öğrenirse, daha sonra bu bölümle karşılaştığında daha iyi sonuç verir ve bu bölüm hakkında öğrenmesini arttırarak uzmanlaşır. Diğer bölümlerle karşılaştığında kötü sonuç verir ve öğrenme kötüleşir. Her iki durumda da, pozitif geri besleme etkisi, kendi kendini kuvvetlendirerek ortaya çıkacaktır. Yani, iyi daha iyi olyurken, kötü daha kötü olacaktır.

2.4.3. Modüler Ağların Avantajları

MYSA'nın öğrenme hızı, ifade yeteneği ve donanım sınırlamaları bakımından tek bir YSA'ya göre birkaç avantajı vardır [20].

Öğrenme Hızı: Eğer karmaşık bir fonksiyon, doğal olarak daha basit fonksiyonlar dizisine ayrıştırılabiliyorsa MYSA kullanılabilir. MYSA, çok katmanlı YSA'nın ayrıştırılmamış karmaşık fonksiyonu öğrenmesine göre daha basit fonksiyonlar dizisini daha hızlı öğrenebilir [22]. Örneğin, Denklem (2.50)'de tanımlanan parça-parça lineer fonksiyonu göz önüne alalım. Bu fonksiyon, basitçe iki lineer nöron ve bir birleştirme biriminden ibaret bir MYSA ile öğrenilebilir. Birinci nöron $x>0$ için $g(x)=x$ fonksiyonunu öğrenir, diğer nöron $x\leq 0$ için $g(x)=-x$ fonksiyonunu öğrenir. MYSA iki sebepten dolayı, $g(x)$ fonksiyonunu çok katmanlı YSA'dan daha hızlı öğrenebilmektedir.

1- Gizli nörona sahip değildir.

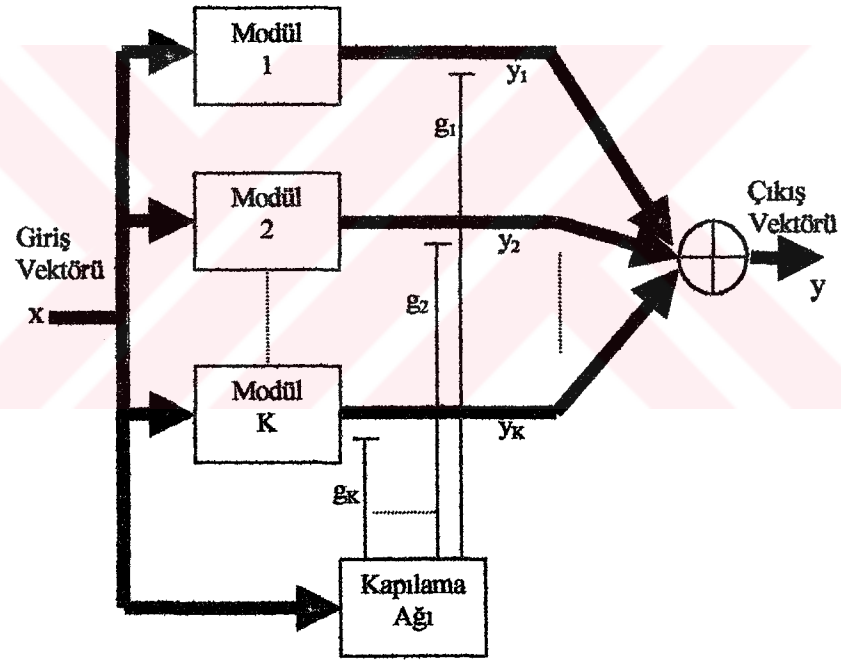
2- Her bir modül ayrı lineer fonksiyonu öğrenir.

2.4.4. Birleşik Gaussian Karma Model

Şekil 2.11'de gösrülen MYSA'nın özel yapısını göz önüne alalım. Bu yapı, uzman ağlar olarak bilinen K tane modül ve uzman ağlar arasındaki aracı fonksiyonu yürüten kapılama ağı isimli bir birleştirme biriminden ibarettir [22]. Uzman ağlara ve kapılama ağına aynı eğitim örnekleri eş zamanlı uygulanarak MYSA eğitilir. Eğitim örnekleri p boyutlu x giriş vektörü ve q boyutlu d istenen çıkış vektörüyle belirlenir. MYSA'nın çıkışı Denklem (2.51)'deki gibi tanımlanır

$$y = \sum_{i}^{K} g_i . y_i . \tag{2.51}$$

Burada; y_i: i. uzman ağın (qx1) boyutlu çıkış vektörünü, g_i: kapılama ağının çıkışını, K: uzman ağ sayısını gösterir.



Şekil 2.11 MYSA'nın blok diyagramı

Bu MYSA'nın eğitiminde kullanılan örnekler, {x,d}, farklı sayıdaki gerileme işlemiyle oluşturulur ve aşağıdaki gibi ilerleyen bir öğrenme algoritması kullanılır [9].

1- x giriş vektörü daha önce elde edilmiş bilgilerden rasgele seçilir.

2- $P(i|x)$ dağılımından bir kural veya uzman ağ seçilir, bu dağılım verilen x giriş vektörü için i. kuralın olasılığıdır.

3- İstenilen cevap vektörü d, seçilmiş i kural ile gerileme işlemine göre elde edilir.

$$d = F_i(x) + \varepsilon_i \qquad i=1,2,...,K \tag{2.52}$$

Burada $F_i(x)$, bir fonksiyon vektörü ve ε_i rasgele bir vektördür. Kolaylık için, ε_i rasgele vektörünün kovaryans matrisi $\sigma^2 I$ ve sıfır ortalama ile dağıtılmış gaussian olduğu kabul edilebilir. Burada; I birim matris ve σ^2 genel varyanstır. Problemleri daha da basitleştirmek için σ^2 =1'e eşitlenir.

Her uzmanın çıkış vektörü, çok değişkenli gaussian dağılımın şartlı ortalaması olarak görülebilir. Bundan dolayı, i. uzman ağın çıkış vektörü y_i Denklem (2.53)'deki gibi tanımlanır,

$$y_i = \mu_i, \qquad i=1,2,.....,K. \tag{2.53}$$

Burada; μ_i vektörü, x giriş vektörüne karşılık gelen d istenilen cevabın şartlı ortalamasıdır ve Denklem (2.52) göz önüne alınarak Denklem (2.54)'deki gibi seçilir,

$$\mu_i = E[d|x,i] = F_i(x), \; i=1,2,...,K. \tag{2.54}$$

Burada; basitlik için $\varepsilon_1 = \varepsilon_2 = \varepsilon_3 = ... = \varepsilon_K$ ve ε_i'nin kovaryans matrisi birim matris kabul edilir,

$$A_i=I, \quad i=1,2,.....,K. \tag{2.55}$$

i. uzman ağ için istenilen cevap d'nin çok değişkenli gaussian dağılımı, Denklem (2.56)'daki gibi ifade edilir,

$$\begin{aligned} f(d \mid x,i) &= \frac{1}{(2\pi.\det A_i)^{q/2}}.\exp(-\frac{1}{2}.(d-y_i)^T A_i^{-1}.(d-y_i)) \\ &= \frac{1}{(2\pi)^{q/2}}.\exp(-\frac{1}{2}(d-y_i)^T.(d-y_i)) \\ &= \frac{1}{(2\pi)^{q/2}}.\exp(-\frac{1}{2}\|d-y_i\|^2), i=1,2,...,K \end{aligned} \tag{2.56}$$

Burada; $\|.\|$, kapsanan vektörün Euclidean normunu gösterir.

Bu temelde, Denklem (2.57)'deki d'nin olasılık dağılımını (K farklı çok değişkenli gaussian dağılımın lineer bir kombinezonu olan) karma model olarak ele alınabilir.

$$f(d|x) = \sum_{i=1}^{K} g_i.f(d|x,i) = \frac{1}{(2\pi)^{q/2}}.\sum_{i=1}^{K} g_i.\exp(-\frac{1}{2}\|d-y_i\|^2) \tag{2.57}$$

Denklem (2.57)'nin olasılık dağılımına birleşik gaussian karma model denir.

Öğrenme algoritmasının amacı, verilen giriş-çıkış data dağılımını modellemektir. Bunu yapmak için, uygun boyutlu ağırlık vektörü w ve kapılama devresindeki tüm çıkışları içeren g vektörü, sırasıyla Denklem (2.58) ve Denklem (2.59)'daki gibi düzenlenir.

$$w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_k \end{bmatrix} \tag{2.58}$$

$$g = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_k \end{bmatrix} \tag{2.59}$$

Denklem (2.57) ile tanımlanan durumlarda logaritma fonksiyonu değişkene göre monotonik artan bir fonksiyon olduğu için f(d | x)'den ziyade, f(d | x)'in doğal algoritmasıyla çalışmak tercih edilir.

$$l(w,g) = \ln f(d|x) \tag{2.60}$$

Denklem (2.60)'da, Denklem (2.57) yerine yazılırsa ve $-\ln(2\pi)^{q/2}$ sabit terimi ihmal edilirse, logaritmik olasılık fonksiyonu l(w,g) Denklem (2.61)'deki gibi elde edilir.

$$l(w,g) = \ln \sum_{i=1}^{K} g_i . \exp(-\frac{1}{2}\|d - y_i\|^2) \tag{2.61}$$

Burada; l(w,g) amaç fonksiyonu olarak alınır ve MYSA'nın tüm bağımsız parametrelerine göre maksimize edilir.

MYSA'nın bilinmeyen büyüklüklerinin birkaçı için aşağıdaki açıklamalar yapılabilir.

1- Uzman ağların optimize edilmiş modülünün çıkış vektörleri y_1, y_2, ...y_K bilinmeyen şartlı ortalama vektörlerdir.

2- Optimize edilmiş kapılama ağının çıkışları g_1, g_2, ..g_K şartlı öncelikli olasılıklardır.

Şekil (2.11)'deki modüler yapının farklı uzman ağları keyfi bir bağlantıya müsaade ederken, kapılama ağının çıkış nöronlarının aktivasyon fonksiyonları iki şartı sağlamak için sınırlandırılır.

$0 \leq g_i \leq 1$ tüm i'ler için (2.62)

ve

$$\sum_{i=1}^{K} g_i = 1 \tag{2.63}$$

bu iki sınır, eğer g_i aktivasyonları bir öncelikli olasılık olarak yorumlanırsa geçerlidir.

Sınırlandırılmamış değişkenlerin bir dizisi $\{u_j | j = 1,2,...,K\}$ için kapılama ağının i. çıkış nöronunun aktivasyonu g_i, Denklem (2.64)'deki gibi tanımlanarak Denklem (2.62) ve (2.63)'deki iki koşul sağlanır.

$$g_i = \frac{\exp(u_i)}{\sum_{j=1}^{K} \exp(u_j)} \tag{2.64}$$

Burada; u_i, kapılama ağının i. çıkış nöronuna uygulanmış girişlerin ağırlıklanmış toplamıdır. Bu normalize edilmiş üstel dönüşüm, lojistik fonksiyonun çok girişli bir genellemesi olarak görülebilir. Bu dönüşüm softmax olarak adlandırılır.

2.4.5 Tahmini Eğim Öğrenme Algoritması

MYSA'nın eğitimi için kulllanılan tahmini eğim öğrenme algoritmasında i.uzman ağın çıkışı ile ilişkilendirilen ardıl olasılık Denklem (2.65)'deki gibi tanımlanır.

$$h_i = \frac{g_i . \exp(-\frac{1}{2}\|d - y_i\|^2)}{\sum_{j=1}^{K} g_i . \exp(-\frac{1}{2}\|d - y_i\|^2)}, \quad i=1,2,...,K \tag{2.62}$$

Bu olasılık, giriş vektörü x ve istenilen cevap vektörü d'nin her ikisine bağlıdır. Ardıl olasılıklar iki koşulu sağlar,

$0 \leq h_i \leq 1$ tüm i'ler için (2.66)

$$\sum_{i\ 1}^{K} h_i = 1 \tag{2.67}$$

MYSA'da iki farklı parametre ayarlaması yapılır.

1- Farklı uzman ağlardaki bağlantı ağırlıklarının değişimi.

2- Kapılama ağındaki bağlantı ağırlıklarının değişimi.

Tüm parametre ayarlamaları eş zamanlı yürütülür.

Uzman Ağların Ayarlanması

Şekil (2.11)'deki MYSA için logaritmik olasılık fonksiyonu Denklem (2.61) ile tanımlanır. Bu denklemin, i. uzman ağın y_i çıkış vektörüne göre türevi alınarak basitleştirildikten sonra Denklem (2.68)'deki (qx1)'lik kısmi türev elde edilir.

$$\frac{\partial l}{\partial y_i} = h_i . (d - y_i) \quad i=1,2,...,K \tag{2.68}$$

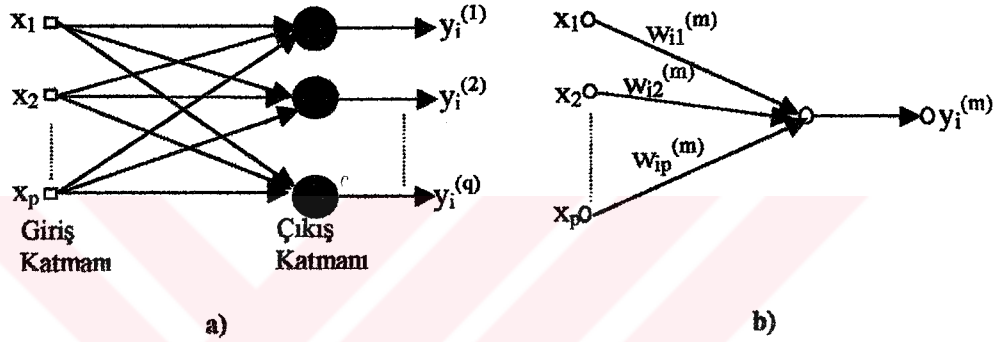
Denklem (2.68); eğitim işlemi boyunca Şekil (2.11)'deki i. uzman ağın bağlantı ağırlıklarının, y_i ve d vektörleri arasındaki hatayı düzeltmek için ayarlandığını ifade eder. Şekil 2.12a'da tek

katmandan oluşan uzman ağ ve sinyal akış şeması görülmektedir. Burada nöronların seçimi sınıflandırma veya gerileme problemine göre yapılır.

* Doğrusal olmayan gerileme probleminde, kalıntıların genel olarak çok değişkenli gaussian dağılıma sahip oldukları kabul edilir. Uygun bir yöntemle uzman ağların çıkış nöronları doğrusal olarak modellenir.

* Sınıflandırma probleminde, çıkış nöronlarının genellikle sigmoidal olduğu kabul edilir. Bu durumda çok değişkenli gaussian bir dağılımdan ziyade Bernoulli dağılımlarının bir karışımı kullanılır.

Her iki durumda da, kapılama ağlarında doğrusal olmayan softmax kullanılır.



Şekil 2.12 a) Doğrusal nöronların tek katmanından oluşmuş uzman ağ
b) Bir doğrusal nöronun sinyal akış şeması

Doğrusal olmayan gerileme problemi ele alındığı zaman, giriş vektörü x ve ağırlık vektörü $w_i^{(m)}$'e karşılık bir iç çarpım olarak i. uzman ağın y_i çıkış vektörünün m. elemanı Denklem (2.69)'daki gibi tanımlanır.

$$y_i^{(m)} = x^T . w_i^{(m)} \qquad \begin{cases} i = 1,2,.....,K \\ m = 1,2,....,q \end{cases} \tag{2.69}$$

Burada T transpozu ve $w_i^{(m)}$ ağırlık vektörü, i. uzman ağdaki m. nöronun ağırlık vektörünü gösterir. Bu w_i vektörü $w_{i1}^{(m)}, w_{i2}^{(m)}, ..., w_{ip}^{(m)}$ elemanlarından oluşur. Bu nedenle $y_i^{(m)}$'in $w_i^{(m)}$ ağırlık vektörüne göre türevinin alınmasıyla, Denklem (2.70) elde edilir.

$$\frac{\partial y_i^{(m)}}{\partial w_i^{(m)}} = x \tag{2.70}$$

$w_i^{(m)}$ ağırlık vektörüne göre l logaritmik olasılık fonksiyonunun duyarlılık vektörü, $\partial l / \partial w_i^{(m)}$ fonksiyonel türeviyle tanımlanır. Zincir kuralı kullanılarak duyarlılık vektörü Denklem (2.71)'deki gibi ifade edilir.

$$\frac{\partial l}{\partial w_i^{(m)}} = \frac{\partial l}{\partial y_i^{(m)}} \cdot \frac{\partial y_i^{(m)}}{\partial w_i^{(m)}} . \tag{2.71}$$

Buradaki $\frac{\partial l}{\partial y_i^{(m)}}$ kısmi türevi, Denklem (2.68)'de tanımlanmış olan $\frac{\partial l}{\partial y_i}$ fonksiyonel türevin m. elemanıdır ve Denklem (2.72)'deki gibi tanımlanır.

$$\frac{\partial l}{\partial y_i^{(m)}} = h_i.(d^{(m)} - y_i^{(m)}) = h_i.e_i^{(m)} \tag{2.72}$$

Burada $e_i^{(m)}$, i. uzman ağdaki m. nöronun çıkışında oluşan hata sinyalini göstermektedir.

$$e_i^{(m)} = d^{(m)} - y_i^{(m)} \tag{2.73}$$

Denklem (2.70) ve (2.72), (2.71)'de yerine yazılırsa,

$$\frac{\partial l}{\partial w_i^{(m)}} = h_i.e_i^{(m)}.x \quad \begin{cases} i = 1,2,...,K \\ m = 1,2,...,q \end{cases} \tag{2.74}$$

olur.Farklı uzman ağların synaptik ağırlıklarına göre logaritmik olasılık fonksiyonunu maksimum yapmak için, ağırlık uzayında eğim yükseltme kullanılır ve Denklem (2.75)'deki $\Delta w_i^{(m)}$ ile ağırlık vektörü yenilenir.

$$\Delta w_i^{(m)} = \eta.\frac{\partial l}{\partial w_i^{(m)}} \quad \begin{cases} i = 1,2,...,K \\ m = 1,2,...,q \end{cases} \tag{2.75}$$

Burada; η küçük bir öğrenme oran parametresini göstermektedir. Öğrenme algoritmasının n. döngüsündeki (adımındaki) ağırlık vektörünün $w_i^{(m)}$ değerini göstermek için $w_i^{(m)}(n)$ gösterimini kullanarak n+1. adımdaki ağırlık vektörünün yenilenmiş değeri kendinden yenilemeli olarak Denklem (2.76)'daki gibi hesaplanır.

$$\begin{aligned} w_i^{(m)}(n+1) &= w_i^{(m)}(n) + \Delta w_i^{(m)}(n) \\ &= w_i^{(m)}(n) + \eta.h_i.e_i^{(m)}.x \end{aligned} \quad \begin{cases} i = 1,2,...,K \\ m = 1,2,...,q \end{cases} \tag{2.76}$$

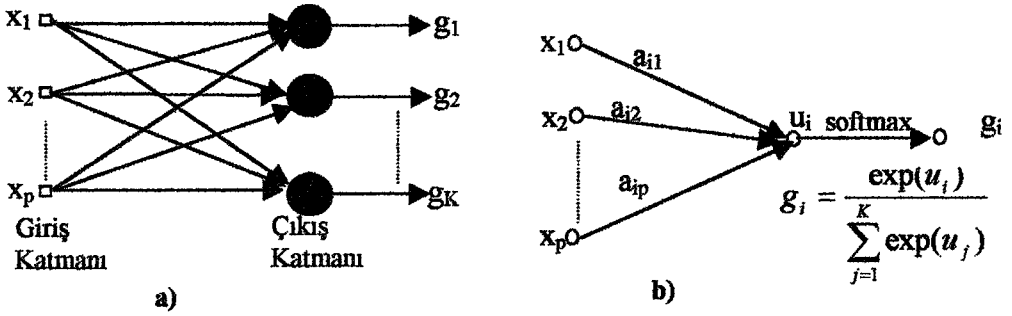
Kapılama Ağının Adaptasyonu

Şekil 2.13a'da görüldüğü gibi kapılama ağı, çıkış nöronlarının tek bir katmanından oluşur. Bu yapı, uzman ağların yapısını gösteren Şekil 2.12a'dan iki bakımdan farklıdır. Birincisi, uzman ağ q çıkış nöronuna sahiptir. İkincisi, uzman ağlarda lineer çıkış nöronları kullanılırken, kapılama ağında çıkış nöronları softmax aktivasyon fonksiyonuna sahiptir.

Kapılama ağının i. çıkış nöronundaki sınırlandırılmamış değişkenlerin o nörona uygulanan girişlerin ağırlıklı toplamı u_i ile gösterilirse i. çıkış nöronunun g_i aktivasyonu,

Denklem (2.64)'ün softmaxıyla u_i ilişkilendirilir. Bundan dolayı Denklem (2.64), Denklem (2.71)'de verilen logaritmik olasılık fonksiyonunun tanımında yerine konursa ve Denklem (2.64)'ün paydasındaki toplam teriminin tüm i'ler için aynı olduğu hatırlanırsa, logaritmik olasılık fonksiyonunun ifadesi Denklem (2.77)'deki gibi tekrar yazılabilir.

$$l = \ln \sum_{i=1}^{K} \exp(u_i).\exp(-\frac{1}{2}\|d - y_i\|^2) - \ln \sum_{j=1}^{K} \exp(u_j) \tag{2.77}$$



Şekil 2.13 a) Kapılama ağı için softmax nöronların tek katmanı **b)** Bir softmax nöronun sinyal akış şeması.

Logaritmik olasılık fonksiyonu l'nin, kapılama ağının i. nöronuna uygulanmış girişlerin ağırlıklı toplamı u_i'ye göre kısmi türevi, Denklem (2.78)'deki gibi bulunur.

$$\frac{\partial l}{\partial u_i} = h_i - g_i \tag{2.78}$$

Bu denklem, önceki olasılıklar, g_i'nin sonraki olasılıklar, h_i'ye doğru hareket etmesi için kapılama ağının i. çıkış nöronun ayarlanması gerektiğini ifade eder. Şekil 2.13b'deki sinyal akış grafiğinden, kapılama ağının i. çıkış nöronunun ağırlıklı toplamının, giriş vektörü ve ilgili ağırlık vektörü a_i'nin bir iç çarpımı olduğu görülmektedir.

$$u_i = x^t a_i \tag{2.79}$$

Burada; a_i vektörü, kapılama devresindeki i. nöronun a_{i1}, a_{i2},..... a_{ip} ağırlıklarından oluşur. Be nedenle, ağırlıklı toplamı, u_i'nin ağırlık vektörü a_i'ye göre kısmi türevi (px1)'ik bir vektördür.

$$\frac{\partial u_i}{\partial a_i} = x \quad i=1,2,...,K \tag{2.80}$$

Ağırlık vektörüne göre logaritmik olasılık fonksiyonu l'nin duyarlılık vektörü $\partial l/\partial a_i$ kısmi türeviyle tanımlanır. Zincir kuralı uygulanarak, duyarlılık vektörü Denklem (2.81)'deki gibi elde edilir.

$$\frac{\partial l}{\partial a_i} = \frac{\partial l}{\partial u_i}.\frac{\partial u_i}{\partial a_i} \quad i=1,2,...,K \tag{2.81}$$

Denklem (2.78) ve (2.80) (2.81)'de yerine yazılırsa,

$$\frac{\partial l}{\partial a_i} = (h_i - g_i).x \tag{2.82}$$

olur.Denklem (2.83)'deki Δa_i ile ağırlık vektörü yenilenir.

$$\Delta a_i = \eta.\frac{\partial l}{\partial a_i} = \eta.(h_i - g_i).x \tag{2.83}$$

Burada; uzman ağların adaptasyonu için kullanılan öğrenme oranı parametresini göstermektedir. Öğrenme algoritmasının n .döngüsünde kapılama ağının i. çıkış nöronunun ağırlık vektörü $a_i(n)$ kendinden yenileme kullanılarak Denklem (2.84)'deki gibi güncellenir.

$$a_i(n+1) = a_i(n) + \eta.(h_i(n) - g_i(n)).x \tag{2.84}$$

Şekil 2.11 için Öğrenme Algoritmasının Özeti

1- Farklı uzman ağların ve kapılama ağının başlangıç ağırlıkları aynı metotla küçük değerler kullanılarak atanır.

2- Uzman ve kapılama ağlarının adaptasyonu

$$u_i(n) = x^T.a_i(n)$$

$$g_i(n) = \frac{\exp(u_i(n))}{\sum_{j=1}^{k}\exp(u_j(n))}$$

$$y_i^{(m)}(n) = x^T.w_i^{(m)}(n)$$

$$y_i(n) = \left[y_i^{(1)}(n), y_i^{(2)}(n), \ldots y_i^{(q)}(n)\right]^T$$

$$h_i(n) = \frac{g_i(n).\exp(-\frac{1}{2}.\|d - y_i(n)\|^2)}{\sum_{j=1}^{K} g_i(n).\exp(-\frac{1}{2}.\|d - y_i(n)\|^2)}$$

$$e_i^{(m)}(n) = d^{(m)} - y_i^{(m)}(n)$$

$$w_i^{(m)}(n+1) = w_i^{(m)}(n) + \eta.h_i(n).e_i^{(m)}(n).x$$

$$a_i(n+1) = a_i(n) + \eta.\left[h_i(n) - g_i(n)\right].x$$

3- 2. adım varolan tüm eğitim örnekleri için tekrarlanır.

4- 2. ve 3. adımlardaki hesaplamalar ağ kararlı hale gelinceye kadar tekrarlanır.

BÖLÜM 3. ÖĞRENME VE ADAPTASYON

3.1. Giriş

Yaklaşım teorisi, sürekli çok değişkenli h(x) fonksiyonuna başka bir H(w,x) fonksiyonuyla yaklaşmayla ilgilenir. Burada $x=[x_1\ x_2\ .....x_n]$ giriş vektörü ve $w=[w_1\ w_2\ .....w_n]$ ağırlık vektörüdür. Öğrenme işlemi, {x} eğitim örneklerinin bir dizisi temelinde h(x)'in mümkün en iyi yaklaşımını sağlayan w'yı bulmaktır. Kullanılacak H(w,x)'in seçimi önemlidir. Kötü seçilen bir fonksiyon, parametreler iyi seçilse bile tahminde kusurlara sebep olur. Seçilmiş olan H(w,x)'in optimal w^* parametrelerini bulmak için öğrenme algoritması uygulanır. Öğrenme algoritması Denklem (3.1)'de tanımlanan mesafe fonksiyonu ℓ'yi minimize eder [10].

$$\ell[H(w^*,x),h(x)] \le \ell[H(w,x),h(x)] \tag{3.1}$$

İleri yönlü tek katmanlı veya çok katmanlı YSA, tanımlanmış olan istenilen dönüşümü elde etmek için öğrenmeye tabii tutulabilir. Geri beslemeli YSA'larda ise öğrenme denge noktalarının kodlanmasına eşdeğerdir.

YSA'larda öğrenme temelde üç gruba ayrılır. Bu gruplandırma ağırlıkların uyarlanabilmesi için gereksinim duyulan uyarı sinyallerine göre yapılır [9, 10].

Denetimli Öğrenme: Bu öğrenmede, zamanın her anında giriş uygulandığında, sistemin istenilen cevabının, bir eğitici tarafından sağlandığı kabul edilir. İstenilen ve gerçek cevap arasındaki fark, bir hata ölçütü olarak düşünülür ve dıştan ağ parametrelerini ayarlamak için kullanılır. Ağırlıkların ayarlanabilir olduğu kabul edildiğinde, eğitici ağın ağırlık matrisini uygun hale getirmek için bir azaltma ve artırma işlemi yapar. Örneğin cevabı bilinen giriş örnekleri veya durumları öğretmede, bilinen cevapla ağ cevabı arasındaki hata ağırlıkların ayarlanmasında kullanılabilir. Böylece hata azaltılır ve öğrenmenin bu grubu oldukça yaygındır. YSA'daki öğrenmenin bir çok safhasında kullanılır. Eğitim dizisi denilen giriş-çıkış örneklerinin bir dizisi bu öğrenme biçimi için gereklidir.

Denetimli öğrenme, doğru dönüşüme pozitif yanlış dönüşüme ise negatif katkıda bulunur. Eğitici hatanın eğiminin negatif yönünü tahmin eder ve böylece hatayı azaltır. Eğiticili öğrenme; fonksiyon yaklaştırma, nesne tanıma ve sistem tanımlama gibi alanlarda kullanılır.

Denetimsiz Öğrenme: Bu öğrenme de istenilen çıkış bilinmez. Bu nedenle, YSA'nın davranışını geliştirmek için kesin hata bilgisi kullanılamaz. Cevapların uygunluğu veya

uygunsuzluğu gibi bir bilgi varolmadığından, öğrenme sınır değerlerde veya rastgele girişlerin cevaplarının gözlenmesi sonucu istenilen amaca göre YSA ağırlıkları ayarlanır ve öğrenme gerçekleştirilir.

Denetimsiz öğrenme video ile bir dersin öğretilmesine benzetilebilir. Tam bir öğrenme söz konusu değildir. Çoğu durumlarda öğrenme denetimsiz bir ortamda mümkün olamaz. Eğiticisiz öğrenme, ağ giriş verilerini gruplandırmak, sürekli zaman girişlerini nitelemek, ağ giriş verilerini daha farklı boyutlu uzayda göstermek, ağ giriş sinyalini temsil eden özellikleri belirlemek gibi amaçlar için kullanılır.

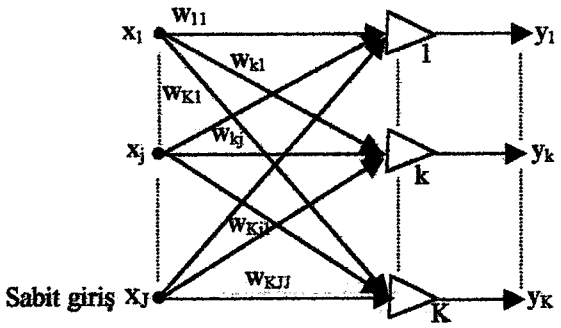
Takviyeli Öğrenme: Takviyeli öğrenme yönteminde, YSA'nın öğrenmesi için bir eğitici sinyal bulunmaz. Ancak, ağın davranışının uygunluğunu belirten bir öz yetenek bilgisine göre ağırlıklar ayarlanır. Bundan dolayı genellikle gerçek zamanlı öğrenme yöntemidir ve deneme yanılma esasına göre YSA öğrenir.

3.2. Geriye Yayılım Öğrenme Algoritması

3.2.1. Çok Katmanlı YSA için Delta Öğrenme Kuralı

Delta öğrenme kuralı, çok katmanlı İYSA'lara uygulanan bir eğitim algoritmasıdır. Hata geriye yayılımı eğitim algoritması veya kısaca geriye yayılım eğitimi de denir [10, 23, 24]. Geriye yayılım eğitimi, çok katmanlı ağlarda tecrübeyle giriş-çıkış dönüşüm bilgisinin kazandırılmasını sağlar. Bu eğitim süresince giriş örnekleri ağa uygulanır. Eğer uygulanan bir örnek neticesinde hatalı bir çıkış oluşursa, hatanın en küçük karesi kuralına göre ağırlıklar veya biaslar ayarlanır. Bu ayarlama veya öğrenme kabul edilebilir bir hataya ulaşıncaya kadar sürer. Genellikle dönüşüm hatası kümülatiftir ve tüm eğitim dizisi üzerinden hesaplanır.

Sınıflandırma veya gruplandırma işlemleri boyunca, kendi kendine eğitilmiş YSA ileri beslemeli biçimde çalışır. Ağırlıkların geriye yayılım kuralıyla uyarlanması, gizli katmanlar üzerinden çıkış katmanından giriş katmanına doğru tamamen geriye doğrudur [25, 26]. Bu öğrenme algoritmasını formüle etmek için sürekli algılayıcılı (sürekli aktivasyon fonksiyonlu) Şekil 3.1'deki tek katmanlı yapıyı göz önüne alalım.



Şekil 3.1 Tek katmanlı YSA

Ağın girişi x_j ve çıkışı y_k dır. Burada j=1,2,...,J, k=1,2,...,K dır. w_{kj}, j. nöronun çıkışı ile k'nın girişini bağlar. Bu durumda çıkış Denklem (3.2a)'daki gibi olur.

$$y = \Gamma[w.x] \tag{3.2a}$$

Burada; giriş, çıkış ve ağırlık vektörleri sırasıyla aşağıdaki gibidir.

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_J \end{bmatrix}, \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_K \end{bmatrix}, \quad w = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1J} \\ w_{21} & w_{22} & \cdots & w_{2J} \\ \vdots & \vdots & \vdots & \vdots \\ w_{K1} & w_{K2} & \cdots & w_{KJ} \end{bmatrix}$$

Non-lineer diyagonal operatör

$$\Gamma(.) = \begin{bmatrix} f(.) & 0 & \cdots & 0 \\ 0 & f(.) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & f(.) \end{bmatrix} \quad \text{ve amaçlanan hedef çıkış vektörü} \quad d = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_K \end{bmatrix}$$

biçimindedir. Denklem (3.2a)'ya göre k. katmanın aktivasyon vektörü

$$net_k = w.x \tag{3.2b}$$

biçimindedir. Belli bir giriş için k=1,2,....K çıkışlarındaki tüm hataların karesini kapsaması için Denklem (3.3)' deki gibi genelleştirilir.

$$E_p = \frac{1}{2}\sum_{k=1}^{K}(d_{pk} - y_{pk})^2 = \frac{1}{2}\left\|d_{pk} - y_{pk}\right\|^2 \tag{3.3}$$

Burada; p girişteki belli bir örneği gösterir. Ağırlıkların ayarlanmasında eğim azaltma ile hatanın küçüldüğü kabul edilir ve basitleştirmek için bias değerleri T_k, (k=1,2,...,K), diğer ağırlıklarla beraber ayarlanır. y_j=-1 için biaslar $w_{kj}=T_k$, k=1,2,...,K ile formülize edilir. Her aşamada ağırlıklardaki düzeltme Denklem (3.4a)'daki gibi hesaplanır.

$$\Delta w_{kj} = -\eta.\frac{\partial E}{\partial w_{kj}}. \tag{3.4a}$$

Burada; E, Denklem (3.3)'de tanımlanan hatayı göstermektedir. k katmandaki her bir düğüm için bu ifade Denklem (3.4b)'deki gibi tekrar yazılırsa,

$$net_k = \sum_{j=1}^{J} w_{kj}.x_j \tag{3.4b}$$

nöronların çıkışları kullanılarak $y_k=f(net_k)$ k. nöronun ürettiği çıkışa göre hatanın değişimi Denklem (3.5)'daki gibi olur.

$$\delta_{yk} = -\frac{\partial E}{\partial(net_k)} \tag{3.5}$$

Türev ifadesine zincir kuralı uygulanırsa, Denklem (3.6) ve (3.7) elde edilir.

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial(net_k)}.\frac{\partial E}{\partial w_{kj}} \tag{3.6}$$

$$\frac{\partial(net_k)}{\partial w_{kj}} = x_j \tag{3.7}$$

Denklem (3.5) ve (3.7) birleştirilirse Denklem (3.8) elde edilir.

$$\frac{\partial E}{\partial w_{kj}} = -\delta_{yk}.x_j \tag{3.8}$$

Bu durumda ağırlıklardaki düzeltme Denklem (3.9)'daki gibi olur.

$$\Delta w_{kj} = \eta.\delta_{yk}.x_j \quad j=1,2,...,J \tag{3.9}$$

Denklem (3.9)'daki ifade tek katmanlı ağlar içindir ve aktivasyon fonksiyonunu biçimine bağlı değildir. Hata fonksiyonu net_k'ya bağlı bir fonksiyon olduğu gibi y çıkış fonksiyonu da net_k'nın bir fonksiyonudur. Bu ifadeler birleştirirse Denklem (3.10) elde edilir.

$E(net_k)=E[y_k(net_k)]$ ve

$$\delta_{yk} = -\frac{\partial E}{\partial y_k}.\frac{\partial y_k}{\partial(net_k)} \tag{3.10}$$

Amaç fonksiyonunun çıkışa göre, aktivasyon fonksiyonunun girişe göre türevi sırasıyla Denklem (3.11a) ve (3.11b)'deki gibidir.

$$\frac{\partial y_k}{\partial(net_k)} = f_k'(net) \tag{3.11a}$$

$$\frac{\partial E}{\partial y_k} = -(d_k - y_k) \tag{3.11b}$$

Bu denklemler birleştirilirse, Denklem (3.12) ve (3.13) elde edilir.

$$\delta_{yk} = (d_k - y_k).f_k'(net_k) \tag{3.12}$$

$$\Delta w_{kj} = \eta.(d_k - y_k).f_k'(net_k).y_j \tag{3.13}$$

Burada; k=1,2,...,K, j=1,2,...,J dır ve J giriş vektörünün, K ise çıkış vektörünün böyutunu gösterir.

Ağırlıkların güncellenmesi Denklem (3.14)'deki ifadeyle gerçekleştirilir.

$$w' = w + \eta \delta_y X^t \tag{3.14}$$

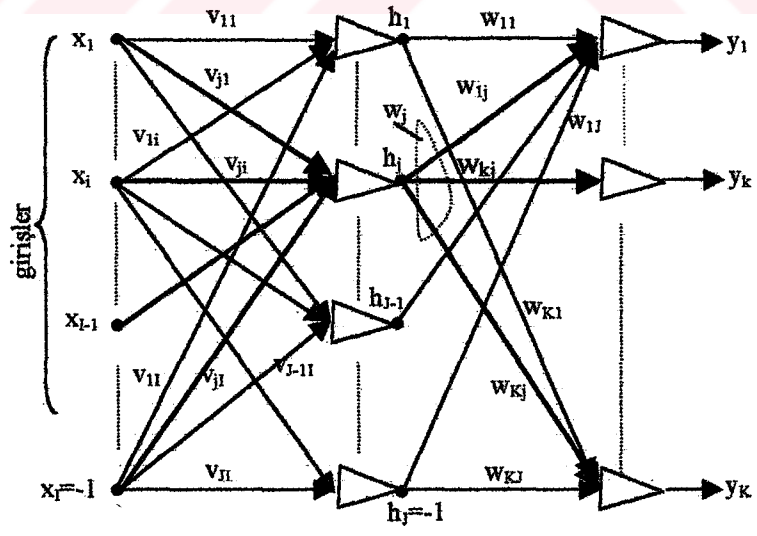
Burada

$$\delta_y = \begin{bmatrix} \delta_{y1} \\ \delta_{y2} \\ \vdots \\ \delta_{yk} \end{bmatrix}$$ ve X vektörü J-1 adet giriş ve (-1) değerli bir eşik değerinden oluşur.

3.2.2. Genelleştirilmiş Delta Öğrenme Kuralı

Şekil 3.2'deki iki katmanlı YSA tam manasıyla işlemsel nöronlu iki katmana sahiptir. Ancak, eğer düğümler katman sayılırsa üç katmanlı bir devre olarak ta kabul edilebilir. Sinyallerin i. kolonu gerçekte bir işlem katmanı olmayan bir (giriş) nöron katmanın bir çıkış cevabı olarak anlaşılabilir. Bu devre iki nöron katmanlı veya üç düğüm katmanlı olarak göz önüne alınabilir.

Çıkışları doğrudan hesaplanamayan nöron katmanlarına gizli veya iç katmanlar denir. Bu nedenle tüm katmanlar, çıkış katı hariç gizli katmanlardır. Nöronların j. katmanının çıkışı giriş ve çıkışlardan doğrudan hesaplanamadığı için Şekil 2.3'deki yapının tek gizli katmanlı bir



Şekil 3.2 İki katmanlı YSA

ağ olduğu söylenebilir. Sonuç olarak, alt indislerin değişimine dikkat ederek, çıkış katmanı olmayan nöronların herhangi bir katmanı için ΔV_{ji} ağırlığının artışı hususunda genel bir ifade elde edilirse, hesaplanmış ağırlıklar Şekil 2.3'de görüldüğü gibi i. düğümden j. düğüme doğru elde edilir. Bu durumda gizli katmanlar için negatif eğim düşme formülü Denklem (3.15a)'daki gibi olur.

$$\Delta v_{ji} = -\eta . \frac{\partial E}{\partial v_{ji}}, \quad j=1,2,....J, \quad i=1,2,...,I \tag{3.15a}$$

$$\frac{\partial E}{\partial v_{ji}} = \frac{\partial E}{\partial (net_j)} . \frac{\partial net_j}{\partial v_{ji}} \tag{3.15b}$$

Katmanlara girişler x_i olup i=1,2,...,I dir. Denklem (3.17)'ye göre Denklem (3.15b)'nin ikinci terimi x_i'ye eşittir ve ağırlıkların ayarlanması Denklem (3.15c)'deki gibi olur

$$\Delta v_{ji} = \eta . \delta_{hj} . x_i . \tag{3.15c}$$

Burada δ_{hj}, h çıkışına sahip gizli katmanın hata sinyali terimidir. Bu hata sinyali gizli katmanın j. nöronu tarafından üretilir. Burada j=1,2,...,J ve hata sinyali Denklem (3.16)' deki gibi olur.

$$\delta_{hj} = -\frac{\partial E}{\partial (net_j)} \tag{3.16}$$

j. düğümdeki hata sinyalini Denklem (3.17a)'daki gibi hesaplanabilir.

$$\delta_{hj} = -\frac{\partial E}{\partial h_j} . \frac{\partial h_j}{\partial (net_j)} \tag{3.17a}$$

Burada;

$$\frac{\partial E}{\partial h_j} = \frac{\partial}{\partial h_j} \left(\frac{1}{2} \sum_{k=1}^{K} \{ d_k - f(net_k(h)) \}^2 \right) \tag{3.17b}$$

$$\frac{\partial h_j}{\partial (net_j)} = f_j'(net_j) \tag{3.17c}$$

dir. Denklem (3.17b)'deki diferansiyel ifade açılırsa ve Denklem (3.12) ifadede kullanılarak basitleştirilirse Denklem (3.18) elde edilir.

$$\frac{\partial E}{\partial h_j} = -\sum_{k=1}^{K} \delta_{hk} . w_{kj} \tag{3.18}$$

Denklem (3.17c) ve (3.18) ifadeleri birleştirilirse ve Denklem (3.17a) tekrar düzenlenirse

$$\delta_{hj} = f_j'(net_j) . \sum_{k=1}^{K} \delta_{yk} . w_{kj} \quad j=1,2,...,J \tag{3.19}$$

olur, bu gizli katmandaki ağırlık ayarlaması Denklem (3.20a)'daki gibidir.

$$\Delta v_{ji} = \eta . f_j'(net_j) . z_i . \sum_{k=1}^{K} \delta_{ok} . w_{kj} \quad j=1,2,...,J, \; i=1,2,...,I \tag{3.20a}$$

Burada $f'_j(net_j)$ basit delta öğrenme kuralında olduğu gibi hesaplanır ve Denklem (3.20a) ifadesi genelleştirilmiş delta öğrenme kuralını ifade eder. Gizli katmandaki j. nöronun önündeki ağırlıkların uyarlanması, çıkış ile j. nörona bağlı sonraki komşu düğüm katmanlarındaki tüm δ değerlerinin ağırlıklandırılmış toplamıyla orantılıdır. j. düğümden dışa doğru yayılan bu ağırlıklar kendi kendilerinin ağırlık faktörüdür. j. gizli neronun δ_{hj} etkisindeki ağırlıklar Şekil 3.2'deki çıkış katında dikkat çekecek biçimde koyulaştırılmıştır. Gizli katmanın ayarlanmış ağırlıkları Denklem (3.20b)'deki gibi ifade edilebilir,

$$v_{ji}^t = v_{ji} + \eta . f_j'(net_j) . x_i . \sum_{k=1}^{K} \delta_{yk} . w_{kj} \quad j=1,2...,J, \; i=1,2,...,I \tag{3.20b}$$

bu ifade vektörel olarak yazılırsa Denklem (3.21) elde edilir.

$$v^t = v + \eta . \delta_h . x^t \tag{3.21}$$

Burada

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_I \end{bmatrix}, \quad v = \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1I} \\ v_{21} & v_{22} & \cdots & v_{2I} \\ \vdots & \vdots & \vdots & \vdots \\ v_{J1} & v_{J2} & \cdots & v_{JI} \end{bmatrix}$$

δ_h' de Denklem (3.19)'da verilmiş olan δ_{hj}' lerden oluşan bir kolon vektörüdür.

$$\delta_h = w_j^t . \delta_y . f_h' \tag{3.22}$$

f'_h, f'_j'lerden (aktivasyon fonksiyonlarının türevlerinden) oluşan kolon vektörüdür. δ_y ise Denklem (3.14)'de tanımlandığı gibidir. Çıkış katmanındaki ağırlıkların ayarlanması için delta eğitim kuralı Denklem (3.14) ve gizli katman ağırlıklarının ayarlanması için genelleştirilmiş delta eğitim kuralı Denklem (3.21) karşılaştırılırsa her iki formülün de anlaşılır ve kalıpsal olduğu görülür. Giriş sinyalleri ve ağırlıkların yerini gösteren alt indislerde ve hata sinyal vektörü δ'nın hesaplanmasında farklılık vardır. δ_y vektörü skalar girişler içerir. δ_y vektörünün her elamanı aktivasyon fonksiyonun türeviyle gerçek ve istenilen çıkışların farkının çarpımıdır. Ancak δ_h vektörü takip eden katmanlarca üretilen δ_y hata sinyallerinin eklenerek ağırlıklandırılmış toplamını ifade eden $w_j^t . \delta_y . f_h^t$ skaler sonucu olan girişleri içerir. Genelleştirilmiş δ eğitim kuralı bir katmanla hatayı geriye doğru yayar, aynı işlemin tüm katmanlar için tekrarlanmasına müsaade eder.

3.2.3. Geriye Yayılım Eğitimi

Şekil 3.2'deki YSA eğitime ihtiyaç duyar. Eğitim örnekleri vektörü x, eğitici tarafından sağlanan istenilen cevap vektörü d ile çiftler halinde düzenlenmiş olmalıdır.

Genelde katmanlı olarak düzenlenmiş ağlar x giriş vektörünü, y çıkış vektörüne Denklem (3.23)'deki gibi dönüştürür.

$$y = N[x] \tag{3.23}$$

Burada; N bileşik doğrusal olmayan bir matris oparetörünü göstermektedir. İki katmanlı bir ağ için Denklem (3.23)'deki gibi x→y dönüşümü, dönüşüm içinde dönüşüm olarak gösterilebilir veya

$$y = \Gamma[w\Gamma[vx]] \tag{3.24a}$$

olur, burada iç dönüşüm

$$\Gamma[vx] = h \tag{3.24b}$$

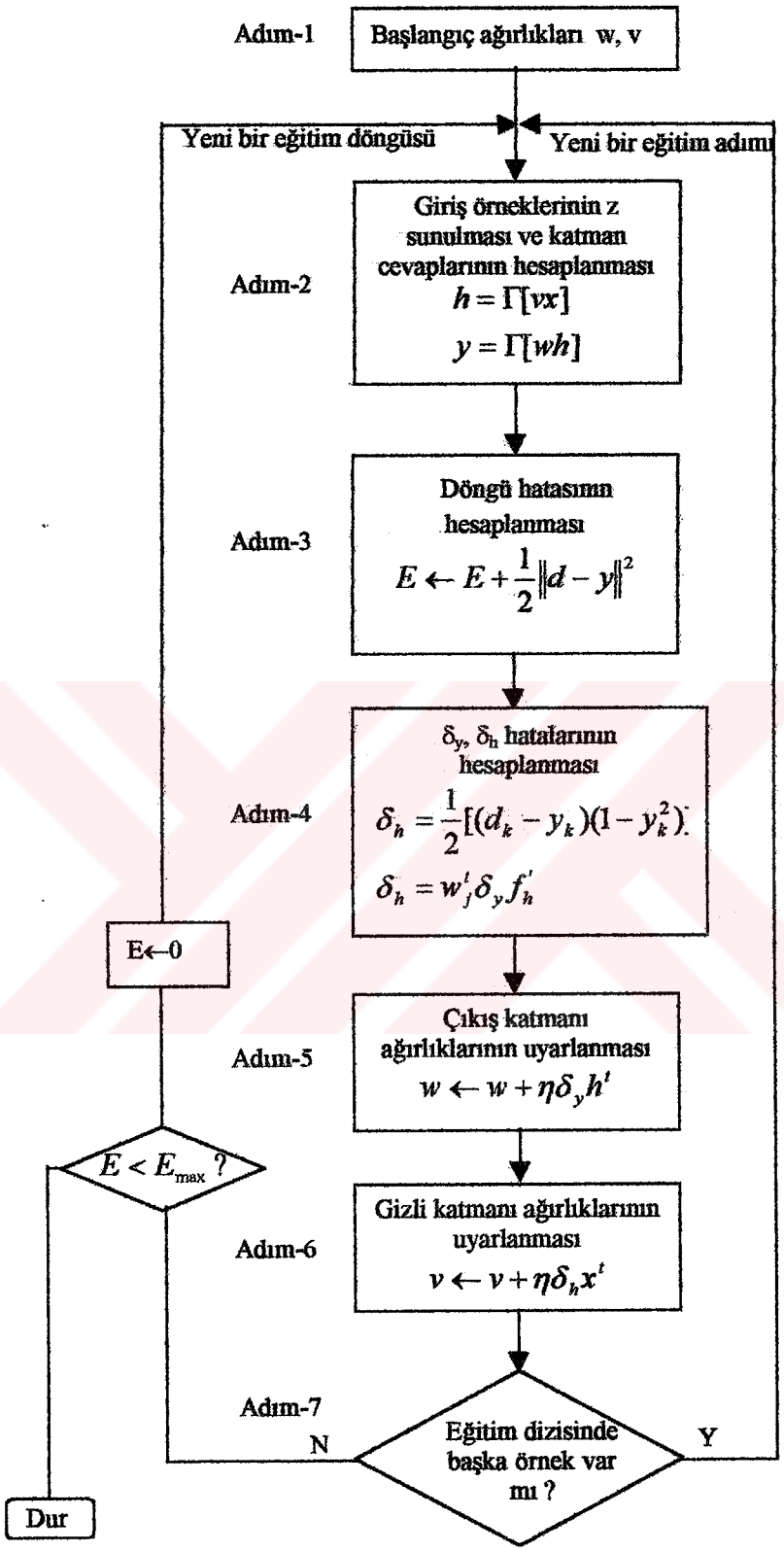
ve bu iç dönüşüm gizli katmandaki x→h dönüşümüyle ilgilidir. Okun yönü bir uzaydan bir diğerine dönüşümüyle ifade eder. Dönüşümün her biri, katmanlı YSA 'nın tek bir katmanıyla elde edilebilir. Γ operatörü her bir elemanı aktivasyon fonksiyonlarına özdeş doğrusal olmayan diyagonal bir operatördür. Bu operatörün diyagonal elemanların her bir nöronun girişinde üretilen net değerlerini kullanır. f(.) fonksiyonun değişkeni sırasıyla gizli ve çıkış katmanları için net_j ve net_k vektörlerinin elamanlarıdır. Denklem (3.24b)'den görülebileceği gibi özdeş ve sabit aktivasyon fonksiyonlarının kabulü x→y dönüşümü için y, d'ye yaklaşır. Sadece parametreler olan ağırlıklar bu sonuca götürür. $||\ d\text{-}y\ ||^2$'ye bağlı minimize edeceğimiz hata değeri ile ayarlanan iki matrise V ve W sahip olunur. Böylece parametreler olarak sunulan ağırlıklar ile çok yönlü doğrusal olmayan dönüşüm sistemleri olarak katmanlı sinir ağlarına bakabiliriz.

Şekil 3.3a'da basit iki katmanlı bir YSA için hata geriye yayılım eğitim algoritmasının akış diyagramı görülmektedir. Öğrenme ileri beslemeli hatırlama işlemi ile başlar (adım-2). Tek bir örnekleme vektörü x giriş de sunulduktan sonra katmanların cevapları h ve y hesaplanır. Sonra hata sinyalini takip eden fazla hata sinyali hesaplanır (adım-4). Hata sinyal vektörü birinci çıkış katmanında hesaplanmalı ve giriş düğümüne doğru yayılmalıdır. K×J boyutlu ağırlıklar w matrisi 5. adımda ayarlanır. Sonra J×I ağırlıktan oluşan v matrisi de 6. adımda ayarlanmalı.

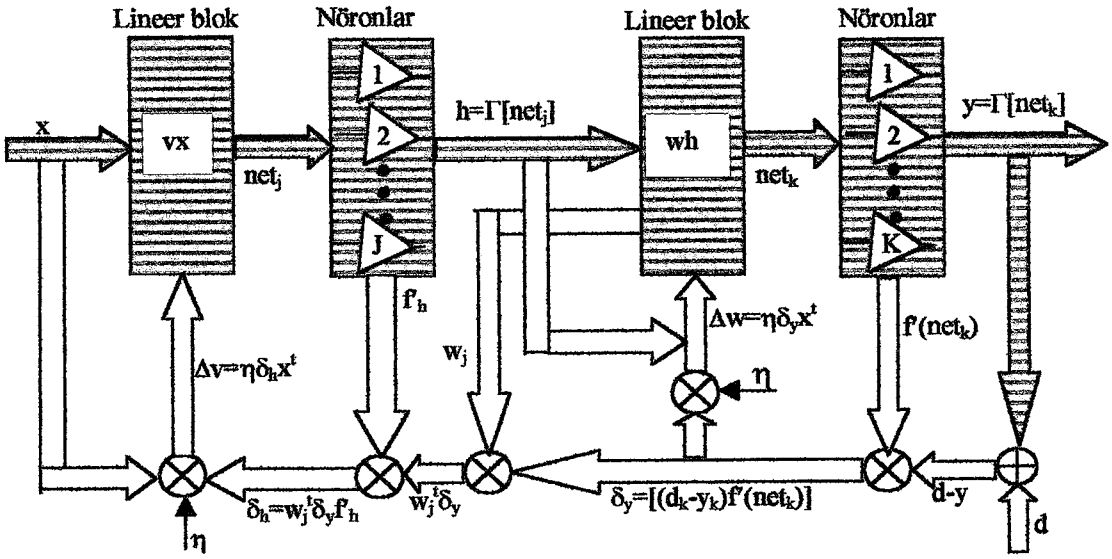
Girişin çıkışa dönüşümünün kümülatif döngü hatası, giriş eğitim dizisindeki tüm sürekli çıkış hataları üzerine bir toplam olarak 3. adımda hesaplanır. Giriş eğitim döngüsü için

son hata değeri eğitim dizisin de $\{x_1\ x_2\ ....\ x_p\}$ tanımlanmış her girişten sonra hesaplanır. Öğrenme işlemi 8. adımda görüldüğü gibi hata bir sınırın altına indiğinde durur.

Hata geriye yayılımıyla eğitilen YSA'nın çalışmasının blok diyagramı Şekil 3.3b'de görülmektedir ve YSA'daki hatanın ve sinyalin akışını açıklar. Diyagramın renkli kısımları ileri beslemeli hatırlamayı gösterir. Diyagramın boş kısımları YSA'nın eğitim fazını gösterir. k=1,2,...,K için her çıkıştan d-y hatasının geriye yayılımı, negatif eğim azaltma tekniği kullanılarak, hata sinyal vektörü δ_h'nün hesaplanması ve çıkış katmanını ağırlık matrisinin uyarlanması Δw'nın hesabı gibi fonksiyonel adımlara bölünür. Bu diyagram δ_h ve giriş katmanın ağırlık uyarlaması Δv sonucunun hesabını da gösterir.


Adım-1
Başlangıç ağırlıkları w, v
Yeni bir eğitim döngüsü
Yeni bir eğitim adımı
Giriş örneklerinin z sunulması ve katman cevaplarının hesaplanması
$h = \Gamma[vx]$
$y = \Gamma[wh]$
Adım-2
Döngü hatasının hesaplanması
$E \leftarrow E + \frac{1}{2}\|d - y\|^2$
Adım-3
δ_y, δ_h hatalarının hesaplanması
$\delta_h = \frac{1}{2}[(d_k - y_k)(1 - y_k^2)$
$\delta_h = w_j^t \delta_y f_h'$
Adım-4
E←0
Çıkış katmanı ağırlıklarının uyarlanması
$w \leftarrow w + \eta \delta_y h^t$
Adım-5
$E < E_{max}$?
Gizli katmanı ağırlıklarının uyarlanması
$v \leftarrow v + \eta \delta_h x^t$
Adım-6
Adım-7
N
Eğitim dizisinde başka örnek var mı ?
Y
Dur

a)



b)

Şekil 3.3 a) Geriye yayılım eğitim algoritmasının akış diyagramı **b)** Geriye yayılım eğitilen YSA' nın blok diyagramı

3.2.4. Öğrenme Oranı Uyum Etkisi

Geriye yayılım eğitim algoritmasının en önemli dezavantajı hedef hataya yavaş yakınsamasıdır. Hatanın birinci türevi kullanıldığından amaç fonksiyonu minimum yapılırken, ağırlık uzayında takip edilen yörünge, negatif eğim yönüyle tespit edilir. Geriye yayılım hata yüzeyi, bir ağırlık yüzeyi boyunca düz olduğunda, ani eğim bilgisi küçük genlikli olacaktır. Bunun bir sonucu olarak, istenilen performansı yakalamak için döngü sayısı artacak ve uzun zaman alacaktır. Global minimuma yakın noktalarda, ağırlık vektöründeki değişim küçük değerlere sahip olacaktır. Bu nedenle, işlemin zaman içindeki değişim hızı logaritmik olarak azalacaktır. Bu dezavantajdan kurtulmak için en faydalı metot, adaptif bir öğrenme oranı kullanmaktır. Öğrenme oranının adaptasyonu Denklem (3.25)'deki gibi yapılır [8, 10].

$$\Delta\eta = \begin{cases} +\gamma & \Delta E < 0 \\ -\beta\eta & \Delta E > 0 \\ 0 & diğer \end{cases} \tag{3.25}$$

Böylece, her döngü için amaç fonksiyonunun aldığı değer hesaplanır ve son değerle bir önceki karşılaştırılır. Bu karşılaştırma sonucuna göre öğrenme oranına bir değişim verilerek ağırlıkların adaptasyonu sağlanır. Amaç fonksiyonunun değerindeki azalma, ağırlık vektöründeki değişimin yavaşladığını gösterir. Çalışma hızını artırmak için basit bir γ sabiti eklenmesiyle öğrenme oranı

arttırılır. Eğer geçmiş birkaç döngüde, ağırlıkların yenilemesiyle amaç fonksiyonunun değeri artıyorsa, yörüngeyi belirleyen öğrenme oranı çok büyük demektir. Daha küçük adımlara ihtiyaç vardır ve öğrenme oranı (1-β) ile azaltılmalıdır.

3.2.5. Ardışıl Geriye Yayılım Öğrenme Algoritması

Daha önce ifade edildiği gibi, tek bir örneğe ait hatanın ($(d_{pk} - y_{pk})^2$) azaltılması temelindeki hata geriye yayılım öğrenmesi, küçük ağırlık ayarlarını gerektirir. Ağırlık ayarlarının, eğitim örneklerinin her sunumunu takiben yapılmasına ardışıl yenileme denir. McClelland ve Rumelhart'ın [10] göstermiş olduğu gibi hata geriye yayılım öğrenme algoritması P, sunumun tüm döngüsü üzerinden hesaplanan Denklem (3.3)'deki tüm hata fonksiyonunun minimumum yapılmasında da kullanılabilir, yeter ki öğrenme oranı yeterince küçük olsun.

Şekil (3.3a)'daki akış şemasıyla ve Bölüm 3.2.3'deki geriye yayılım öğrenme algoritmasıyla özetlenmiş olduğu gibi anlık ve tek bir öğrenme hatasının minimumum yapılmasının avantajı, algoritmanın hata yüzeyinin doğru bir eğim inişi yerine getirmesidir. Bundan başka, bilgisayar benzetimi boyunca, algoritma tarafından tespit edilen ağırlık ayarlarının saklanması gerekmez ve P bileşenden oluşan hata sinyalinin ve ağırlık ayarlama hesap adımlarının yavaş yavaş kompanzesine ihtiyaç duyulmaz. Ancak bu yolla eğitilen ağ, döngüde en yeni örneklere doğru eğilebilir. Bu özel problemi önlemek için, ya küçük bir öğrenme oranı kullanılmalı veya kümülatif ağırlıklar, çıkış ve gizli katmanların her ikisi tüm öğrenme döngüsü sonunda Denklem (3.26)'daki gibi güncellenmelidir [10, 27].

$$\Delta w = \sum_{p=1}^{P} \Delta w_p \tag{3.26}$$

Bu eğitimde ortalama bir etkiye de sahip olabilir. Hesaplanmış her eğitim döngüsünden sonra hem kümülatif ağırlık ayarı hem de her örneğin sunumundan sonraki artışlı ağırlık ayarı tatmin edici çözümler oluşturabilmekle beraber, eğitimin en iyi rasgele koşullar altında gerçekleştiği duruma dikkat edilmelidir. Her örnek sunumundan sonra artımlı ağırlık yenilemesini kullanmak bu nedenle uygun görülür. Ağırlıkların, sunulan en son eğitim örneğine yönlenmesi bu algoritmada karşılaşılacak bir problemdir. Eğitim örneklerine gürültü eklemek problemini hafifletir [28].

3.3. Levenberg-Marquardt Öğrenme Algoritması

Levenberg-Marquardt, Newton metoduna bir yaklaşımdır. Newton'un orijinal yaklaşımı bir F(w) fonksiyonunun, w parametre vektörünün Denklem (3.27)'deki yenilenme ile minimum yapıldığını kabul eder [8, 27].

$$\Delta w = -(\nabla^2 E(w))^{-1}.\nabla E(w) \tag{3.27}$$

Performans fonksiyonunun Denklem (3.28)'deki gibi tanımlandığı kabul edilir.

$$E(w) = \frac{1}{2}\sum_{k=1}^{K}(d_k - y_k)^T.(d_k - y_k) \tag{3.28}$$

Burada; k: eğitim çiftlerini, K: eğitim çiftlerinin sayısını, d: hedef çıkışı ve y: ağ çıkışını gösterir. Eğer Denklem (3.28) ağırlık vektörüne bağlı çıkış hatasına göre tekrar yazılırsa, Denklem (3.29) elde edilir.

$$E(w) = \frac{1}{2}\sum_{k=1}^{K}e_k(w)^T.e_k(w) \tag{3.29}$$

Amaç, Denklem (3.29)'daki her çarpımı minimum yapmaktır. Eğer, bir w_0 civarında Taylor seri açılımı $e_k(w)$'a uygulanırsa Denklem (3.30) elde edilir.

$$e_k(w) \approx \hat{e}_k(w) = e_k(w_0) + J^T.(w - w_0) \tag{3.30}$$

Burada; J Jakobian matristir, w_0'da Denklem (3.31)'deki ifade ile hesaplanır. Bu matrisin tanımı parametre vektörünün j. parametresine göre i. çıkışta hesaplanan hatanın türevini gösterir. bu ifade açıkça jakobian matrisinin satır sayısının, eğitim çiftleri sayısı ve ağ çıkışlarının sayısıyla çarpımına eşit olduğunu işaret eder. Benzer olarak, parametre vektörü ağın tüm ağırlıkları ve biasları için bir girişe sahip olacaktır.

$$J_{ij} = \frac{\partial e_{ki}(w)}{\partial w_j} \tag{3.31}$$

Denklem (3.32)'de tanımlandığı gibi yaklaşık bir hata bileşeni tanımlarız. Bu denklemin w parametre vektörüne göre differansiyeli ve Denklem (3.33)'deki eşitlik, lineer en küçük kareler probleminin olağan denklemini verir.

$$\Phi(w) = \hat{e}_k^T(w).\hat{e}_k(w) \tag{3.32}$$

$$J^T.J.(w - w_0) + J^T.e_k(w) = 0 \tag{3.33}$$

Denklem (3.33)'den, parametre vektöründeki değişim çekilirse

$$\Delta w = -(J^T.J)^{-1}.J^T.e_k(w) \tag{3.34}$$

olur. Denklem (3.34)'de görülen terimler aşağıdaki gibi açıklanır.

$$\nabla E(w) = J^T . e_k(w) \tag{3.35}$$

$$\nabla^2 E(w) = J^T . J \tag{3.36}$$

Denklem (3.35) performans fonksiyonunun birinci türevini, Denklem (3.36) ise performans fonksiyonunun ikinci türevini gösterir ve elde edilen matrise Hessian matrisi denir. Bu matrise Denklem (3.37)'deki gibi ilave bir terim eklenerek metot güçlendirilir [8].

$$\Delta w = -(J^T . J + \mu . I)^{-1} . J^T . e_k(w) \tag{3.37}$$

Bu metodun, Denklem (3.32)'deki Φ'i minimum yapması mümkündür ama genellikle E(w)'ı minimum yapmayabilir. Bu nedenle, bu metodun gelişen parçası olarak Hessian matrisine ölçeklendirme tanımlanır. Eğer bir döngü adımı E(w)'ı azaltıyorsa, μ azaltılır, azaltmıyorsa birden daha büyük bir faktörle μ arttırılır. μ büyükse, bu metotta aşırı azalma olur, çünkü bu düzeltme J^TJ terimini baskın yapar ve sadece birinci dereceden türev bilgisi olduğu gibi kalır, diğer durumda ise, eğer μ çok küçükse bu metot tamamen Gauss-Newton metodu olur. Algoritmada anahtar adım, jakobian matrisin gelişimidir. Aşağıda çok katmanlı algılayıcı durumunda bu matrisin gelişimi açıklanır.

1- Jakobianın satır sayısı KxN'e eşittir. Burada, K eğitim çiftlerinin sayısı ve N ağ çıkışlarının sayısıdır. Bu nedenle, ilk N satır ilk eğitim örneği için türevlerin değişimine karşılıktır. Bir alt matris gibi her eğitim örneği için düzenlenir.

2- Jakobianın kolon sayısı, devrenin parametrelerinin sayısına eşittir. Bunlar ağırlıklar ve biaslardır.

3- Bir w_{hi}^l ağırlığına karşılık olan girişler için Denklem (3.38) bu girişlerin gelişimini tanımlar. l. katmanın h. nöron biasına karşılık olan girişler, Denklem (3.39) kullanılarak gelişir.

$$J_{ij} = \delta_h^l . d^{l-1} \tag{3.38}$$

$$J_{ij} = \delta_h^l \tag{3.39}$$

4- Denklem (3.38) ve (3.39) klasik hata geriye yayılımının kullanıldığını belirtir. Her çıkışın türevi jakobianın farklı satırlarında görüleceğinden, hata vektörünün geriye yayılımında, sadece ilgilenilen delta değeri geriye yayılır, diğerleri geçici olarak sıfırlanır ve daha sonra geriye yayılır.

Aşağıda çok katmanlı algılayıcılara bu algoritmanın uygulanma prosödürü sıralanmıştır.

1- Tüm ağırlıklar ve biaslar küçük rasgele değerlere ayarlanır.

2- μ ve β girilir (örn: μ=.01, β=10).

3- Ağa bir giriş örneklemesi girilir.

4- Delta değerleri hesaplanır ve jakobianın N satırı hesaplanır.
5- Eğer bütün girişler girildiyse devam edilir aksi halde 3. adıma geri dönülür.
6- Denklem (3.37) çözülerek parametre vektöründeki değişim elde edilir.
7- Yeni parametre vektörü kullanılarak bir döngüye geçilir.
8- Eğer yeni karesel hata toplamı azaldıysa, β ile μ azaltılır, aksi halde son güncelleme iptal edilir ve β ile μ değeri arttırılarak 6. adıma geri dönülür.
9- Eğer algoritma yakınsadıysa durulur aksi halde 3. adıma geri dönülür.

3.4. Eşlenik Eğim Metodu

Eğim inişin birinci derece metodunda, ağırlıkların ayarlanması için kullanılan vektörün yönü basitçe eğim vektörünün negatifidir. Dolayısıyla, çözümle elde edilmiş bir minimuma yaklaşım zikzak bir yol izleyebilir. Eşlenik eğim metodu, eğim vektörleri ve yön arasındaki çapraz ilişkiyi kapsayarak bu problemi elimine eder [8, 27]. Bu metot en fazla N adımda, N değişkenli ikinci derece bir fonksiyonun lokal minimumunu garantiler. Çok katmanlı bir algılayıcının eğitminde kullanılan amaç fonksiyonu gibi ikinci derece olmayan fonksiyonlar için, yöntem N adımdan ziyade kendinden yenilemelidir ve yakınsamanın olması için bir kriter gerektirir.

Ağırlık vektörü Denklem (3.40)'daki kurala göre göncellenir.

$$w(n+1) = w(n) + \eta(n).p(n) \tag{3.40}$$

Burada; η(n) öğrenme oran parametresi, n döngü adımını ve p(n), algoritmanın n. adımdaki yön vektörünü göstermektedir. p(0) başlangıç yön vektörü n=0 başlangıç noktasındaki negatif eğim vektörüne eşitlenir.

$$p(0) = -g(0) \tag{3.41}$$

Daha sonra bir sonraki yön vektörü, bir önceki yön vektörü ve o anki eğim vektörünün lineer bir kombinezonu olarak hesaplanır.

$$p(n+1) = -g(n+1) + \beta(n).p(n) \tag{3.42}$$

Buradaki β(n), bir zaman değişim sabitini göstermektedir. g(n) ve g(n+1) eğim vektörlerine bağlı olarak β(n)'nin hesabı için değişik kurallar vardır, bunlardan ikisi aşağıdaki gibidir [10].

1- Fletcher– Reeves formülü

$$\beta(n) = \frac{g^T(n+1).g(n)}{g^T(n).g(n)} \tag{3.43}$$

2- Polak-Ribière formülü

$$\beta(n) = \frac{g^T(n+1).[g(n+1) - g(n)]}{g^T(n).g(n)} \tag{3.44}$$

Denklem (3.40)'daki güncelleme formülündeki öğrenme oran parametresi η(n), w(n) ve p(n) sabit değerleriyle $\wp_{AV}(w(n) + \eta.p(n))$ amaç fonksiyonunu minimize edecek şekilde Denklem (3.45)deki gibi hesaplanır.

$$\eta(n) = \arg\min_{\eta} \{\wp_{AV}(w(n) + \eta.p(n))\} \tag{3.45}$$

Çok katmanlı algılayıcıların denetimli eğitimi için kullanılan, eşlenik eğim tabanlı geriye yayılım öğrenme algoritmasının, standart geriye yayılım algoritmasına göre daha az döngü gerektirdiği bir çok çalışmada gösterilmiştir. Bunun yanında hesapsal olarak daha karışıktır.

3.5. Gauss-Newton Metodu

Gauss- Newton eğitim algoritması ağırlıkların ayarlanmasını amaç fonksiyonunun gerçek azalması ve tahmini azalması arasındaki oranın boyutuna göre yapar [25, 27]. Amaç fonksiyonu olarak Denklem (3.28)'deki standart en küçük toplam karesel hatalar fonksiyonunu kullanır. Taylor serisi açılımı kullanılarak, E(w) amaç fonksiyonundaki ΔE(w) değişimi Δw'nin ikinci derece bir fonksiyonu Denklem (3.47)'deki gibi elde edebilir.

$$\Delta E(w) = E(w + \Delta w) - E(w)$$

$$\Delta E(w) \cong g.\Delta w + \frac{1}{2}.\Delta w^T.H.\Delta w \tag{3.46}$$

Burada; g eğim vektörünü, H ise Hessian matrisi gösterir ve sırasıyla Denklem (3.47) ve (3.48)'da görüldüğü gibi hesaplanırlar.

$$g = \frac{\partial E(w)}{\partial w} = J.e_k(w) \tag{3.47}$$

$$H = \frac{\partial^2 E(w)}{\partial w^2} = J^T.J \tag{3.48}$$

Burada; J Denklem (3.31)'deki jakobyan vektörünü gösterir. Denklem (3.46)'nın Δw'a göre türevi alınırsa, ΔE(w) değişiminin g+H.Δw=0 sağlandığında minimum olacağı görülür. Bu ifadeden ağırlık matrisindeki değişim miktarı Denklem (3.49)'daki gibi bulunur.

$$\Delta w = -H^{-1}.g \tag{3.49}$$

Burada; H^{-1} Hessian matrisin tersidir. Denklem (3.49)'deki ifade Newton metodunun temel yaklaşımıdır [10].

Hessian matrisi ve tersini kullanması, Newton metodunun çok katmanlı algılayıcıların eğitiminde uygulanmasına engeldir. Hessian matrisin hesabının güç olması yanında, tersi alınabilmesi tekil olması durumunda daha da karmaşıktır. Çok katmanlı algılayıcıların denetimli eğitilmesi durumunda Hessian matrisin daima tekil olmamasının garantisi yoktur. Newton metodunun, çok katmanlı algılayıcılarda kullanımı için Newton metodundan türetilmiş algoritmalar kullanılır. Gauss-Newton bulardan birisidir. Bu metotta ağırlıkların yenilenmesi, hessian matrisin tersi yerine hessian matrisin tersine bir yaklaşım sağlayan karşılıklı ilişki (kovaryans) matrisi ve jakobyan vektör yardımıyla belirlenen bir öğrenme oranıyla yapılır. Yenileme işlemi için değişik metotlar kullanılır [10, 27].

1- Unutma Faktörlü Yenileme Metodu

$$\begin{aligned} K(k) &= P(k-1).J(k).(\lambda.I + J^T(k).P(k-1).J(k))^{-1} \\ w(k) &= w(k-1) + K(k).(d(k) - y(k)) \\ P(k) &= (P(k-1) - K(k).J^T.P(k-1))/\lambda \end{aligned} \tag{3.50}$$

Burada, k döngü adımını, P kovaryans matrisi, λ unutma faktörünü, I birim matrisi, J jakobyan matrisi, w ağırlık matrisini, K öğrenme oranını, d gerçek çıkış ve y model çıkışını gösterir.

2- Kararlı Yörünge Yenileme Metodu

$$\begin{aligned} K(k) &= P(k-1).J(k).(1 + J^T(k).P(k-1).J(k))^{-1} \\ w(k) &= w(k-1) + K(k).(d(k) - y(k)) \\ \overline{P}(k) &= P(k-1) - K(k).J^T.P(k-1) \\ P(k) &= \frac{\alpha_{max} - \alpha_{min}}{tr(\overline{P}(k))}.\overline{P}(k) - \alpha_{min}.I \end{aligned} \tag{3.51}$$

Burada; α_{min} ve α_{max} kovaryans matrisin alt ve üst sınırlarını, tr ise matrisin diyagonal elemanlarının toplamını alan bir operatörü gösterir.

3. Unutma Faktörlü Ve Ve Sıfırlayan Algoritma Yenileme Metodu

$$\begin{aligned} K(k) &= \alpha.P(k-1).J(k).(1 + J^T(k).P(k-1).J(k))^{-1} \\ w(k) &= w(k-1) + K(k).(d(k) - y(k)) \\ P(k) &= \frac{1}{\lambda}.P(k-1) - K(k).J^T.P(k-1) + \beta.I - \delta.P^2(k-1) \end{aligned} \tag{3.52}$$

Burada; α öğrenmede uygun azalmayı sağlayan sabiti, β yeterince fazla adım boyutunu, δ başlangıç adım boyutunu gösterir.

Algoritma aşağıdaki gibi özetlenebilir,

1) Başlangıç ağırlık vektörlerinin ve Gauss-Newton metoduyla ilgili parametrelerin değerleri seçilir.
2) Ağ çıkışı, jakobyan vektör hesaplanır.
3) Yenileme metotlarından birisi kullanılarak ağırlıklar yenilenir.
4) İstenilen kritere ulaşılıncaya kadar 2. ve 3. adımlar tekrarlanır.

YSA eğitimi için üstel unutma faktörlü, tipik olarak en hızlı yakınsamayı veren metottur. Ancak, bir YSA'da genellikle var olan çok sayıdaki ağırlıklar nedeniyle, ağırlık uzayının tüm doğrultularının uygun biçimde harekete geçmesini sağlamak zor olduğundan unutma faktörünü seçerken dikkat edilmelidir. Eğer λ çok küçükse, kovaryans matrisinin belli öz değerleri kontrol edilemez biçimde artacaktır. Sabit yörünge metodu P'nin anormal büyümesini önlemek amacıyla, P'nin öz değerleri için alt ve üst değerlerle (α_{min} ve α_{max}) sınırlarlar.

Unutma faktörlü metot için, bir başlangıç P(0) matrisi seçilmelidir. Bunun için genellikle $P(0)=c \times I$ seçilir. Burada c, $10 - 10^4$ aralığında büyük bir sayıdır. Diğer iki metot için P(0) başlangıç değeri, diyagonal elemanları en yüksek öz değerler olan diyagonal bir matristir [27, 29].

BÖLÜM 4. YAPAY SİNİR AĞLARINI KULLANARAK SİSTEM TANIMA

4.1. Sistem Tanıma

Geleneksel ve YSA temelli sistem tanıma işlemi deneysel veri temelli dinamik bir sistemin matematiksel modelini oluşturmaktır. YSA'nın doğrusal olmayan fonksiyonları yeterli doğrulukta yaklaştırabilme yeteneği, doğrusal olmayan sistemlerin tanınmasında kullanılmasını mümkün kılar. YSA temelli sistemi tanımada, model çıkışlarını ölçülen gerçek çıkışlara benzer yapmak için YSA'nın eşik ve iç ağırlıkları ayarlanır. Bu işlem, sistemin giriş-çıkış verilerinin kullanılarak YSA'nın eğitilmesidir. Geleneksel metotlar diferansiyel denklemin katsayılarını veya durum uzay gösterimlerini hesaplamak için gerileme tekniklerini ve deneysel verileri kullanır.

Sistem tanıma problemi matematiksel olarak, k=1,2,...,K adet örnek için sistemden ölçülen giriş u(k) ve çıkış d(k) örneklerinden yararlanarak sistemin gelecekteki çıkışını veren doğrusal olmayan çıkışını belirlemek şeklinde tanımlanabilir. Sistem tanıma problemi model giriş vektörünün seçimi ve doğrusal olamayan ilişkinin belirlenmesi şeklinde alt problemlere ayrıştırılabilir.

4.1.1. Sistem Tanıma Modelleri

Sistem tanıma modellerini doğrusal ve doğrusal olmayan modeller olarak ayırmak mümkündür. Ancak her ikisinde de modelleme mantığı aynıdır, çünkü doğrusal olmayan sistemler genellikle doğrusallaştırılarak modellenir. En yaygın doğrusal modellemelerden birisi ARX modelidir. Çıkışı, sistemin geçmiş durumlarına bağlı olan dinamik bir modeldir. Belli bir anda modelin sabit çıkışı sadece o andaki girişe bağlıdır. Diferansiyel denklem formundaki tanımlamalar ARX modelidir

Doğrusal olmayan modeller giriş vektörlerine göre doğrusal olmayan dinamik modeller ve işleve göre doğrusal olmayan modeller olarak iki sınıfa ayrılabilir [23].

4.1.1.1. Giriş Vektörüne Göre Doğrusal Olmayan Dinamik Modeller

Tanımalarda kullanılan x durum uzay vektörünü, y model çıkış vektörünü, i=1,2,.....n sistem çıkışındaki gecikmeyi ve j=1,2,....m ise sistem girişindeki gecikmeyi göstermektedir.

NFIR Modeller: Model giriş vektörü olarak sadece sistem girişlerini kullanan modellerdir.

$$x(k) = \left(u(k-j)\right) \tag{4.1}$$

NARX Modeller: Model giriş vektörü olarak sistem girişi ile birlikte çıkışlarını da kullanan modellerdir. NARX modeller, seri-paralel model olarak ta bilinir ve YSA ile gerçekleştirilmek istenirse ileri beslemeli YSA gerektirir.

$$x(k) = \left(d(k-i), u(k-j)\right) \tag{4.2}$$

NARMAX Modeller: Model giriş vektörü olarak sistem giriş ve çıkışlarıyla birlikte hata girişini de kullanan modellerdir. NARMAX modeller gerçekte sistemin genel bir modelini gösterir ve ileri beslemeli YSA yapılarıyla gerçekleştirilebilir.

$$x(k) = \left(d(k-i), u(k-j), e(k-i)\right) \tag{4.3}$$

Burada; e hata vektörünü göstermektedir.

NOE Modeller: Model giriş vektörü olarak sistem girişi ile birlikte model çıkışlarını da kullanan modellerdir. NOE modeller, paralel modeller olarak bilinir ve geri beslemeli YSA yapısıyla gerçekleştirilebilir.

$$x(k) = \left(y(k-1), u(k-j)\right) \tag{4.4}$$

NBJ Modeller: Model giriş vektörü olarak sistem girişi ve model çıkışı ile birlikte hata girişlerini de kullanan modellerdir. NBJ modeller, geri beslemeli YSA yapısıyla gerçekleştirilebilir.

$$x(k) = \left(y(k-i), u(k-j), e(k-i)\right) \tag{4.5}$$

Bu modeller dışında, durum değişkenleri ile birlikte sistem girişini model giriş vektörü olarak kullanan doğrusal olamayan durum uzay modelleri de sayılabilir. Doğrusal olmayan durum uzay modelleri , YSA ile gerçekleştirilmeleri durumunda orta katman çıkışlarından geri besleme yapılan katmanlı YSA ile modellenebilir.

4.1.1.2. Doğrusal Olmayan İşleve Göre Model Yapıları

Sistemin giriş ve çıkışlarının doğrusal olup olmamasına göre 4 çeşit model yapısı tanımlanabilir [30]. f(.) ve g(.) türevi alınabilir doğrusal olmayan fonksiyonlar olmak üzere bu 4 model, model giriş vektörlerine göre NARX yada NOE model yapısındadır.

Model 1: Sistem çıkışlarına göre doğrusal, sistem girişlerine göre doğrusal olmayan model yapıları:

$$d(k) = a_i.d(k-i) + g\left(u(k-j)\right) \tag{4.6}$$

Model 2: Sistem girişlerine göre doğrusal, sistem çıkışlarına göre doğrusal olmayan model yapısıdır:

$$d(k+1) = f\big(d(k-i)\big) + b_j u(k-j) \tag{4.7}$$

Model 3: Sistem giriş ve çıkışlarına göre doğrusal olmayan ancak, sistem giriş ve çıkışlarının farklı doğrusal olmayan fonksiyonlarla gösterilebileceği model yapısıdır.

$$d(k+1) = f\big(d(k-i)\big) + g\big(u(k-j)\big) \tag{4.8}$$

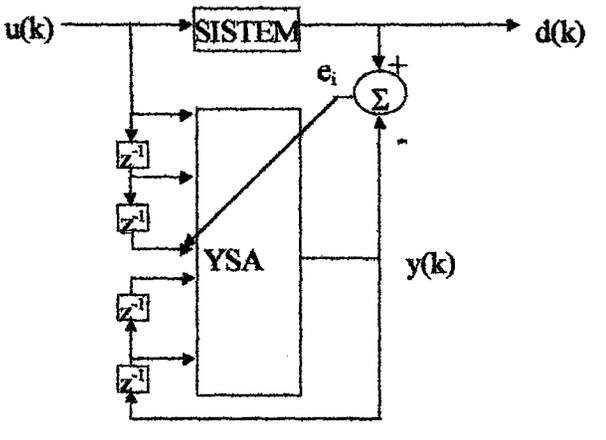
Model 4: Sistem giriş ve çıkışlarına göre doğrusal olmayan ancak, sistem giriş ve çıkışlarının tek bir doğrusal olmayan fonksiyonla gösterilebileceği model yapısıdır. Genel olarak doğrusal olmayan sistemler bu model yapısıyla tanımlanır.

$$d(k+1) = f\big(d(k-i), u(k-j)\big) \tag{4.9}$$

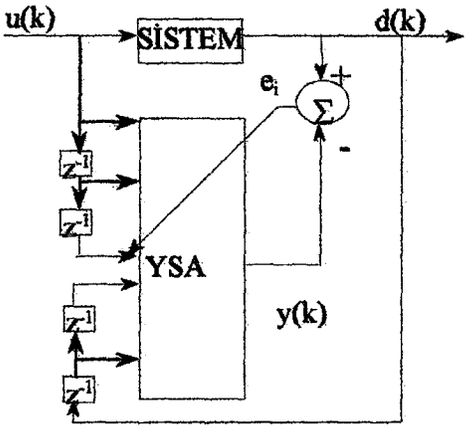
4.2. YSA ile Sistem Tanımlama

Bir fonksiyon yaklaştırma probleminde kullanılan YSA'nın yapısına göre, gizli katmanlardaki hücre sayısı ve kararlılık analizleri gibi kesin ispat edilmemiş bilgilere rağmen YSA, sistem tanımada yaygın olarak kullanılmaktadır. Düz modelleme ve ters modelleme olarak iki modelleme yöntemi mevcuttur.

Bir sistemin düz yön dinamiklerini gösterecek şekilde, YSA modelinin seçilmesi ve eğitilmesine düz modelleme denir [23]. Bir sistemin düz yön dinamikleri matematiksel olarak, d(k)=f(d(k-i), u(k-j)) ile ifade edilebilir. Düz modellemede istenilen çıkış sistem çıkışıdır. YSA sistem çıkışı ile model çıkışı arasındaki hata sinyali kullanılarak eğitilir. Şekil 4.1'de görülen paralel model, YSA'ya sistem girişi ve model çıkışından oluşan bir giriş uygulanarak elde edilir.

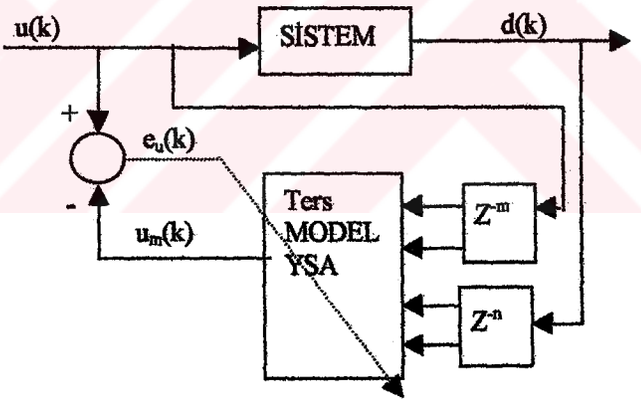


Şekil 4.1 YSA ile düz modelleme, paralel (NOE) model



Şekil 4.2 YSA ile düz tanımlama, seri-paralel (NARX) model

Şekil 4.2'de görülen seri-paralel modelde, YSA'ya sistem girişi ve sistem çıkışından oluşan bir giriş uygulanır. Bir sistemin ters yön dinamiklerini gösterecek şekilde YSA modelinin seçilmesi ve eğitilmesine ters modelleme denir. Bir sistemin ters yön dinamikleri matematiksel olarak, $u_m(k)=f(d(k-i), u(k-j))$ ile ifade edilebilir, burada u_m model çıkışını ifade eder. Bu modele ilişkin blok diyagramı Şekil 4.3'de görüldüğü gibidir.



Şekil 4.3 Bir sistemin ters YSA modeli

Ters YSA modelin arzu edilen çıkışı, sistem girişidir ve sistem girişi ile model çıkışı arasındaki hata sinyalinden yararlanılarak YSA eğitilir.

4.3 YSA ile Sistem Tanımanın Temel Adımları

A. Deneysel giriş/çıkış verilerini birleştirmek.

B. Aday model yapılarını tahmin etmek ve seçmek.

C. Modelleri gözden geçirmek ve en iyi modeli seçmek.

Bu üç adım aşağıdaki genel prosedürü kullanarak başarıyla sonuçlandırılabilir [30]:

1. Bir deney dizayn etmek ve verileri seçmek. Bu verilerin çalışma kümesi içinde, çalışmadan doğru bir sonuç alabilmek için sistemi uyaran girişlerin bütün değerlerinin bulunması gerekir.
2. Verileri işlemek.
3. Bir model yapısı seçmek.
4. Belirlenen yapı için en iyi parametreleri hesaplamak.
5. Modelin özelliklerini belirlemek.
6. Eğer bu özellikler kabul edilebilirse çıkılır kabul edilemezse 3'e geri dönülür.

Veri kümesinin elde edildiğini varsayarsak, bir sonraki adım bir model yapısı seçmektir. Bu seçim lineer olmayan durumda, lineer durumdakinden daha zordur. Sadece gerileyicilerle kümesini seçmek gerekmez aynı zamanda ağ yapısı da gerekir. Burada kullanılan yaklaşım lineer sistem tanımadan esinlenme temelli gerileyicilerle seçmektir ve daha sonra giriş olarak verilen gerileyicilerle ağ yapısını seçmektir [30].

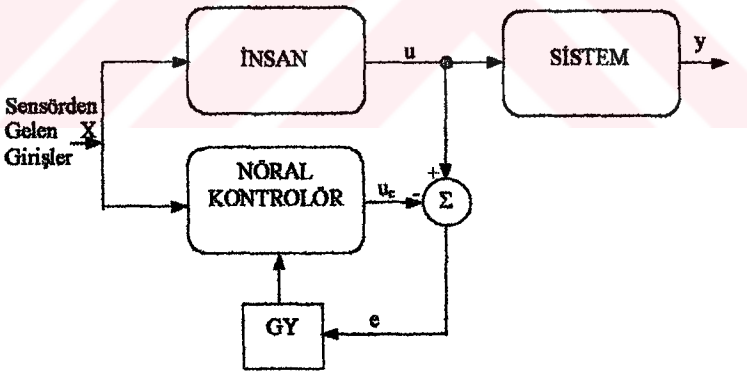
Genel model yapısı seçildiği zaman, model derecesi seçilmelidir. Model derecesi gerileyici olarak kullanılan geciktirilmiş sinyalin sayısıdır. Bu sistem bilgisi veya deneyler sayesinde tahmin edilir. Ağ sistem tanıma modeli çalıştırılır ve test edilir. Çıkışların geciktirilmiş girişler ve hata terimleri arasındaki aşırı ilişkisi, geciktirilmiş giriş ve çıkışların hesaplanmış olan hata üzerindeki etkisi ile görülür. Eğer çıkış hata terimlerinin geciktirilmiş giriş sinyalleriyle aşırı derecede ilişkisi varsa, bu geciktirilmiş giriş sinyalinin bir giriş olarak içine alınması gerektiğine inanmak için bir sebeptir. Bu, modele ait geciktirilmiş girişlerin sayısını belirlemek için tecrübeye dayanan bir metottur [30].

BÖLÜM 5. NÖRAL-KONTROL YAPILARI

Bu bölümde sadece nöral kontrolör geliştirmek için kullanılmakta olan farklı metotlara değinilmiştir. Bu metotların çoğu ters kontrol tabanlıdır. Öncelikle bu metotlardan bahsedilir ve sonra belki de nöral-kontrol yaklaşımların en ümit vereni olan "YSA model tabanlı tahmini kontrol yapısı" ele alınır [31].

5.1. Ters Sistem Tabanlı Nöral-Kontrol Yapısı

Taklit ederek öğrenme biyolojik sistemlerin çoğunun temel özelliklerinden birisidir. İYSA'da görüldüğü gibi denetimli bir sinir ağı, başka bir sistemin davranışını taklit etmek üzere doğal olarak düzenlenir. Nöral kontrolörü geliştirmekle ilgili ilk metot, bir insan kontrolörünün yerini almakla ilgilidir. Bu teknik, ilk olarak 1964'de ters sarkacı kontrol etmek için kullanılmıştır [7]. YSA eğitimi basitçe, insan kontrolör tarafından alınan sensör bilgileri ve kontrol girişi arasındaki dönüşümü öğrenmekten ibarettir [29]. Şekil 5.1'de görüldüğü gibi bu metot da en önemli problem, insani tecrübenin eğitim aşamasında kullanılabilir bilgi haline getirilmesidir.



Şekil 5.1 İnsanı taklit eden nöral-kontrolörün öğrenmesine ilişkin blok diyagramı

Benzer bir yaklaşım klasik metotla tasarlanmış başka bir kontrolörün davranışını taklit eden nöral kontrolörü eğitmek için kullanılabilir. Bu kontrolörler çok fazla hesaplama gerektiren klasik kontrol yaklaşımlarında yararlı olabilir. Bir çok uygulamada nöral kontrolörler bir sistemin tersi olarak eğitilirler [10, 32, 33]. Şekil 5.2'de gösterilen bu yaklaşım özellikle hızlı dinamiklere sahip sistemler için kullanılabilir [7]. Bu kontrol yapısında kullanılan nöral kontrolörler farklı öğrenme metotlarıyla eğitilirler.

5.1.1. Direkt Ters Öğrenme Metodu

Direkt ters kontrol, bir sistemi kontrol etmek için o sistemin ters dinamiğini kullanmayı amaçlar [7]. Kontrol edilecek sistemin Denklem (5.1)'deki gibi tanımlandığı kabul edilirse,

$$y(t) = f(y(t-1),..,y(t-n),u(t-1),....,u(t-m) \tag{5.1}$$

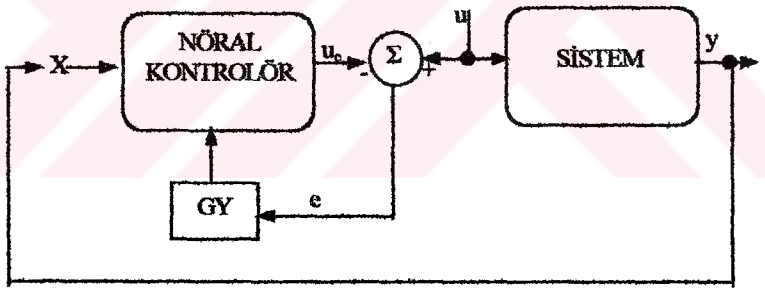
sistemin ters dinamiği Denklem (5.2)'deki gibi elde edilir.

$$u(t) = f^{-1}(y(t+1),y(t),...,y(t-n+1),u(t-1),...,u(t-m+1)) \tag{5.2}$$

Kontrol sisteminin, referans girişi kusursuz bir şekilde izleyebilmesi için Denklem (5.2)'deki y(t+1) ifadesi, referans giriş r(t+1) ile değiştirilir ve elde edilen gerileme matrisine (X) göre sistemin ters YSA modeli eğitilir ve nöral kontrolör olarak kullanılır. Burada; m, bir sonraki çıkışı hesaplarken kullanılan geçmiş girişlerin sayısını, n ise geçmiş çıkışların sayısını gösterir. Eğitim işlemi toplu yapıldığında bu işleme direkt ters öğrenme metodu adı verilir. Eğitilmiş nöral kontrolör, Denklem (5.3)'deki kontrol sinyalini üretir [29].

$$u_c(t) = f^{-1}(r(t+1),y(t),...,y(t-n+1),u(t-1),....,u(t-m+1)) \tag{5.3}$$

Şekil 5.2'de direkt ters öğrenme metoduna ilişkin blok diyagramı görülmektedir. u eğitim için seçilen kontrol sinyalini, u_c ise nöral kontrolörün ürettiği kontrol sinyalini göstermektedir.



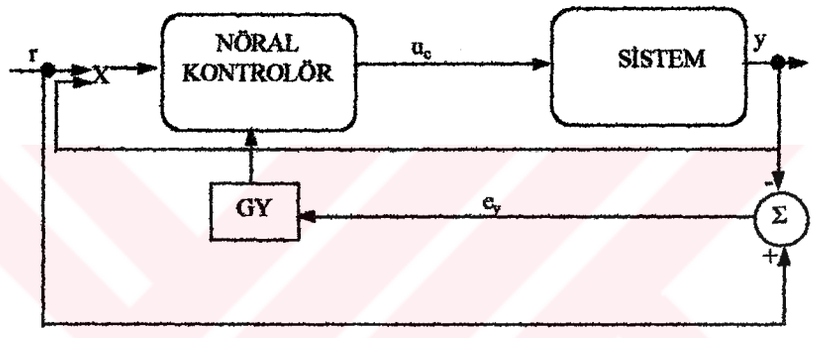
Şekil 5.2 Direkt ters öğrenme metoduna ilişkin blok diyagramı

Nöral kontrolör, u sinyallerinin bir dizisiyle eğitilir. Sistem çıkışı ve giriş vektörleri arasında bir geri besleme yolu olmasına rağmen, bu kontrol sistemi, uygun bir kontrol sinyaline karar vermek için sistem çıkışı ve referans giriş arasındaki hata e_y=r-y'yi göz önüne almadığından açık çevrimdir. Açıkça sistem çıkışı ve istenilen çıkış arasındaki herhangi bir sapmayı gidermek için YSA tarafından alınmış herhangi bir önlem olmayacaktır. Bu bakımdan, bu işlem geri beslemeli bir kontrol sistemi değil açık çevrim bir kontrol sistemidir.

Bu öğrenme metodunun uygulanmasındaki ana zorluk, eğitim sinyali u'un seçimidir. Kontrolörün eğitimi için, seçilen kontrol sinyali, sistemi istenilen çalışma bölgesine taşıyabilmelidir. Böyle bir sinyalin tespiti, sistem hakkında güçlü bilgiler olmaksızın oldukça zordur.

5.1.2. Özelleştirilmiş Ters Öğrenme Metodu

Bu öğrenme metodunun amacı, direkt ters öğrenme metoduyla eğitilmiş nöral kontrolörün performansını iyileştirmektir. Bu metotta nöral kontrolör, kontrol performansı e_y=r-y'yi minimum yapmak için örneksel olarak eğitilir. Bu özellik, direkt ters öğrenme ve özelleştirilmiş ters öğrenme arasındaki temel farktır.



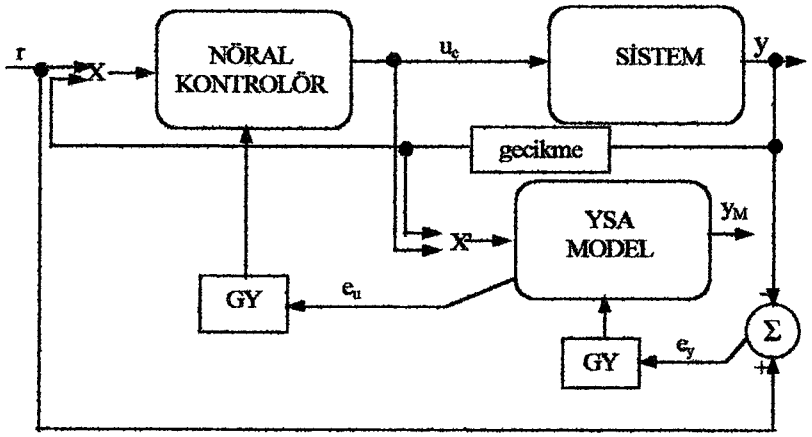
Şekil 5.3 Özelleştirilmiş ter öğrenme metoduna ilişkin blok diyagramı

Gerçekte nöral kontrolörün e_u=u-u_c öğrenme performansıyla eğitilmesi uygundur. u, istenilen çıkışı sağlayacak kontrol sinyalini ifade eder. Bu metot eğer kontrol performansı güvenilir ise, nöral kontrolörü optimal ters çözüme götürür. Çünkü, e_y olması gerektiği gibi model çıkışıyla direkt ilişkili değildir. Bu metodu uygulamak için işlemin jakobianının gerekli olduğu ispat edilmiştir. Bu nedenle bir çok sisteme uygulanmaz. Ancak, hatanın e_u ham bir tahmininin dahi, öğrenmenin gerçekleşmesini sağlayabilir. Örneğin, sistem türevlerinin işaretinden, iyi bir öğrenme elde edilebilir [7].

5.1.3 Zamanla Geri Yayılan Öğrenme Metodu

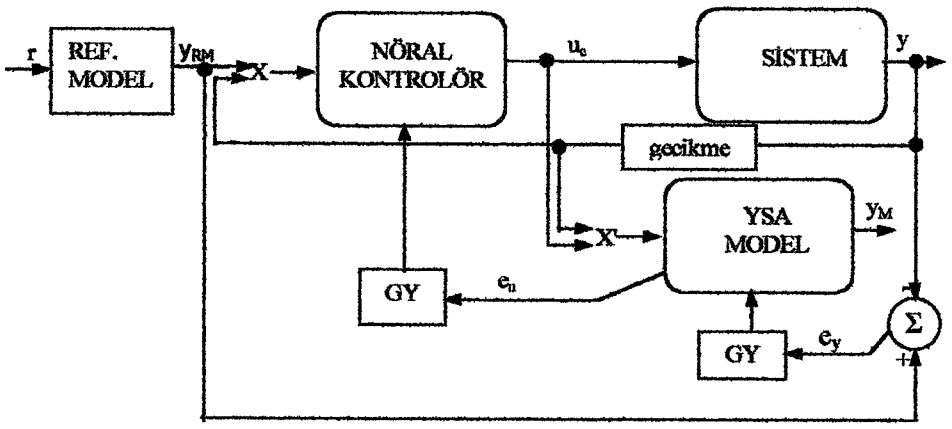
Özelleştirilmiş ters öğrenme metodunun problemi, hata performansı e_y=r-y'nin güvenilir olmamasıdır, çünkü bu performans, nöral kontrolörün çıkışıyla, direkt ilişkili değildir. Bu problemin üstesinden gelebilmek için zamanla geri yayılan öğrenme metodu geliştirilmiştir. Bu metotta, sistemin ileri yönlü modeli kullanılarak geriye yayılmış bir performans e_y vasıtasıyla

e_u'nun izlenmesi amaçlanır. Şekil 5.4'de görüldüğü gibi, ileri beslemeli bir modelin giriş katmanına doğru geriye yayılmış hatası e_y, e_u=u-u_c hatasını karşılamalıdır.



Şekil 5.4 Zamanla geri yayılan öğrenme metoduna ilişkin blok diyagramı

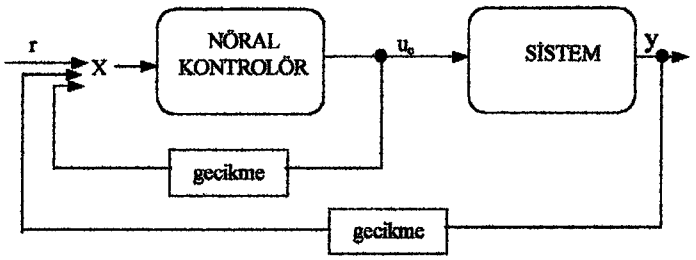
İlk olarak sistemin YSA modeli bilinen metotlardan birisiyle elde edilir. YSA model, nöral kontrolörün eğitimi esnasında e_y hatasının geriye yayılması amacıyla kullanılır, tekrardan adaptasyona tabii olmaz. e_y hatası sistem çıkışı yerine YSA model çıkışı y_M kullanılarak tespit edilir. Eğitim örneksel yapılır ve nöral kontrolörün parametreleri ayarlanır. Eğitimin temelini oluşturan ifade Denklem (5.2)'deki gibidir. Bu ayarlama esnasında YSA modelin girişine indirgenmiş hata uygulanır. Eğitim için ardışıl geriye yayılım, Levenberg-Marquart, Gauss-Newton gibi algoritmalar kullanılabilir [29]. Bu metot, nöral kontrolörün eğitiminde sistemi kullanmanın uygun olmadığı durumlarda da faydalı olabilir. Doğal olarak sistem çıkışının kullanılması daha iyi sonuç verir [7].



Şekil 5.5 Zamana göre geri yayılan ve model referanslı öğrenme metoduna ilişkin blok diyagramı

Şekil 5.5'de zamanla değişmeyen basit bir istenilen çıkış yerine istenilen geçiçi sistem çıkışı y_{RM}'i verecek bir referans modelin yapıya eklendiği model referanslı öğrenme metoduna ilişkin blok diyagramı görülmektedir [7].

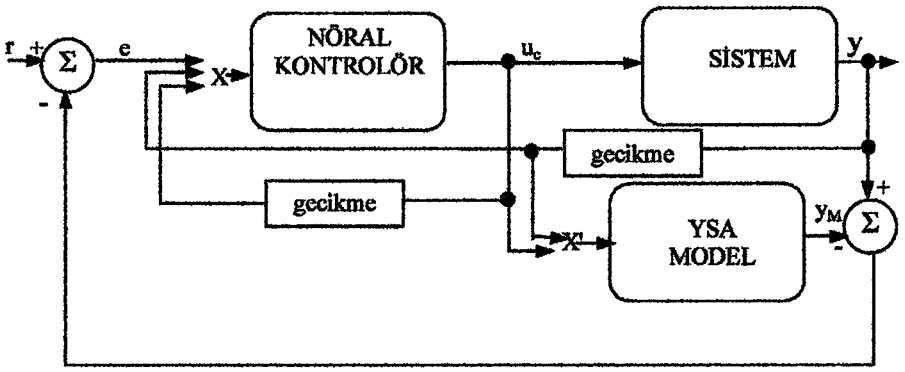
Yukarıda bahsedilen öğrenme metotlarıyla eğitilen nöral kontrolörler Şekil 5.6'daki kontrol yapısında kullanılır. Bu yapıya direkt ters nöral-kontrol yapısı denir. Kontrol sinyali Denklem (5.1)'deki gibi hesaplanır [29].



Şekil 5.6 Direkt ters nöral-kontrol yapısı

5.2. İçten YSA Model Tabanlı Kontrol Yapısı

İçten YSA model tabanlı kontrol yapısı da aslında ters sistem temelli bir kontrol yapısıdır. Ancak direkt ters nöral-kontrol yapısından farklı olarak sistemin kontrolü aşamasında sadece sistemin ters modelini değil iyi eğitiliş bir YSA modelinide kullanır. Direkt ters nöral-kontrol yapısında kontrolör geçmiş giriş çıkış değerleri dışında referans değeri de giriş olarak kullanır. İçten YSA model tabanlı kontrol yapısında ise giriş olarak kullanılan referans değer yerine bir hata ifadesi kullanılır [29].



Şekil 5.7 İçten YSA model tabanlı kontrol yapısı

Bu hata ifadesi Denklem (5.4)'deki gibi hesaplanır,

$$e = r - y + y_M \tag{5.4}$$

Burada; y sistemin gerçek çıkışını, y_M YSA modelinin çıkışını, r ise referans girişi göstermektedir. Kontrol sinyali denklem (5.5)'deki gibi ters YSA model tarafından tespit edilir.

$$u_c(t) = f_T^{-1}(e(t), y(t),..., y(t-n+1), u_c(t-1),..., u_c(t-m+1)) \quad (5.6)$$

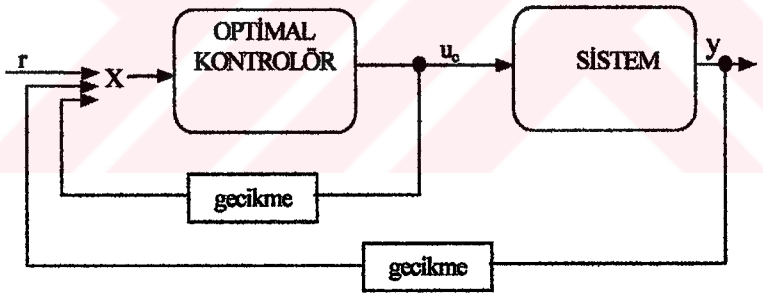
İçten YSA model tabanlı kontrol yapısında kullanılan, sistemin YSA modeli sistem tanımada kullanılan metotlarla, kontrolör ise direkt ters nöral-kontrol yapısında kullanılan öğrenme metotlarıyla eğitilir. Bu iki model Şekil 5.7.'de görülen kontrol yapısında kullanılır.

5.3. YSA Tabanlı Optimal Kontrol Yapısı

YSA tabanlı optimal kontrol yapısında sistemin ters YSA modeli bir optimal kontrolör olarak eğitilir. İlk olarak sistemin YSA modeli sistem tanımada kullanılan metotlardan birisiyle ve ters YSA model ise direkt ters öğrenme metoduyla toplu eğitimle elde edilir. Daha sonra bu ters YSA model, YSA modelde kullanılarak Denklem (5.6)'daki amaç fonksiyonunu minimize edecek şekilde örneksel eğitilip optimal kontrolör elde edilir.

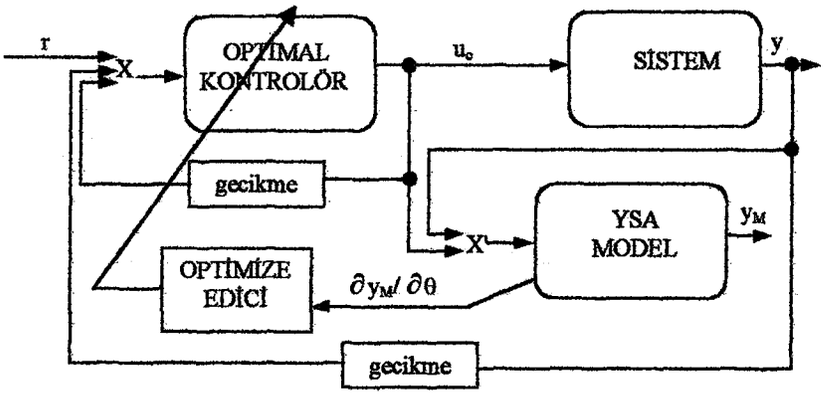
$$E(\theta) = \sum_t \left[(r(t+1) - y(t+1))^2 + rho * (u(t|\theta))^2 \right] \quad (5.6)$$

YSA tabanlı optimal kontrol yapısı Şekil 5.8'de görülmektedir [29].



Şekil 5.8 YSA tabanlı optimal kontrol yapısı

YSA tabanlı optimal kontrol yapısının direkt ters nöral-kontrol yapısından farkı, kontrolörlerin farklı yöntemlerle eğitilmesidir. Direkt ters nöral-kontrol yapısında eğitimin amacı sadece karesel hatayı minimum yapmak iken, YSA tabanlı optimal kontrol tekniğinde hatanın yanı sıra kontrol sinyalinin değişimide hesaba katan bir amaç fonksiyonu minimum yapılmaya çalışılır. Eğitim için genellikle Gauss-Newton optimizasyon tekniği kullanılır. Sistemin YSA modeli ise sadece eğitim aşamasında kullanılmaktadır. Sistemin jacobyeni yerine YSA modelinin Jakobyeni kullanılmaktadır [34]. Optimal kontrolörün eğitimine ilişkin bir blok diyagramı Şekil 5.9'da görülmektedir.



Şekil 5.9 YSA tabanlı optimal kontrolörün eğitimine ilişkin blok diyagramı

5.4. YSA Model Tabanlı Tahmini Kontrol Yapısı

Tahmini kontrol yapısı, gerçek dünyadaki sistemlerin çoğunu etkileyen ölü zaman (yani gecikme) probleminin üstesinden gelmek için geliştirilmiştir. Yani sistemler bir kontrol girişine asla hemen tepki göstermeyeceklerdir. Sistemin girişe reaksiyon göstermesi için geçen süre gecikme zamanıdır.

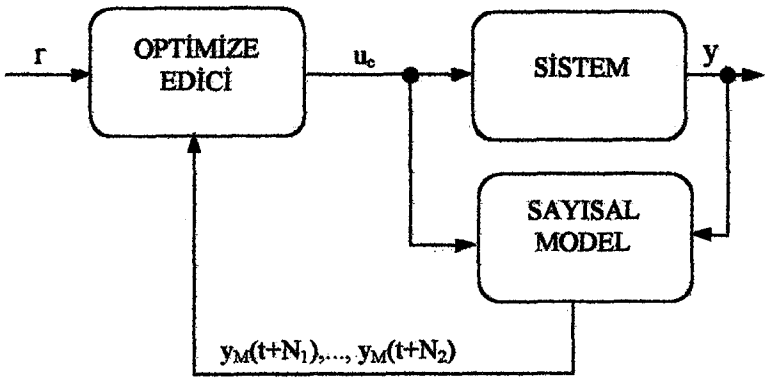
Klasik geri beslemeli kontrolde amaç, sistem çıkışını y, referans giriş r'ye doğru hareket ettirmek için gerçek kontrol hatasına göre sistemde etkin olmaktır.

$$e = r - y \tag{5.6}$$

Eğer sistemde 1 s'lik bir zaman gecikmesi olduğu kabul edilirse, kontrol etkisinin birinci saniyesi boyunca, kontrol hatası e'nin etkilenmeyeceği açıktır. Böylece, birinci saniye boyunca kontrolör aynı tarzda rol oynayacaktır. Bu istenmeyen davranışı önlemek için, birinci saniye boyunca beklemek ve sonra bir kontrol sinyalini etkinleştirmek bir çözümdür. Hızlı kontrol gerektiren bir durumda bu başarılamayacaktır. Bu yüzden, başka bir çözüm, bir saniye sonra sistem durumunun ne olabileceğini tahmin edebilecek bir sistem modeli kullanmak ve bu model çıkışına göre kontrol hatasını tespit etmektir. Bu tahmini kontrolün temelidir.

Bu fikir karmaşık dinamikli sistemlere ve benzersiz sistem modellerine değinilerek genelleştirilmiştir ve bu kontrolörler "genelleştirilmiş tahmini kontrol"(GTK) olarak adlandırılmıştır. GTK, aslında model tabanlı tahmini bir kontrolörün dizaynıdır.

YSA tabanlı tahmini kontrol, üç temel bileşenden oluşur: Sistem, sistemin YSA modeli ve bir fonksiyon optimize edici. Bu bileşenler Şekil 5.10.'da görüldüğü gibi birleştirilir.



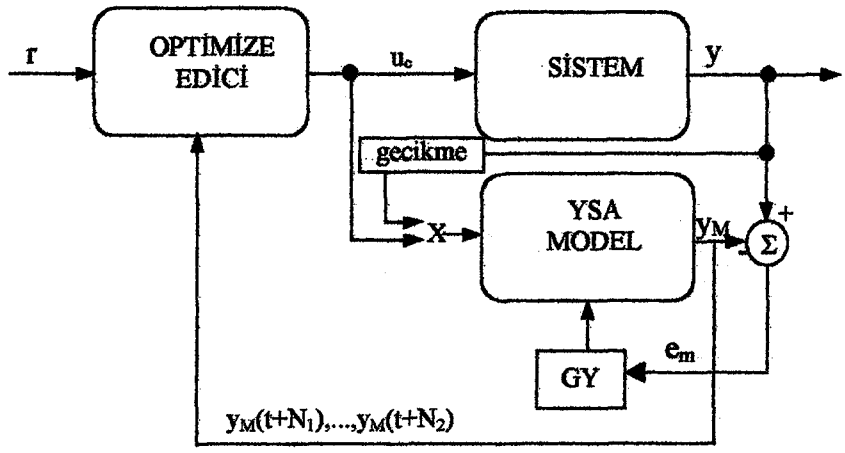
Şekil 5.10 Model tabanlı tahmini kontrol yapısı

Sistemin YSA modeli gelecekteki davranışları tahmin etmek için kullanılır. Sistemin tahmin edilmiş davranışlarına göre, optimize edici, sistem çıkışını referans girişe yaklaştıracak u_c sinyalinin gereken dizisini belirler. Bu optimize edici Denklem (5.7)'deki ikinci derece bir amaç fonksiyonu kullanır.

$$J(N_1,N_2,N_u,t)=\sum_{i=t+N_1}^{t+N_2}\mu(i)\left[r(i)-y_M(i)\right]^2+\sum_{i=t}^{t+N_u}\lambda(i)\left[\Delta u_c(i-1)\right]^2 \qquad (5.7)$$

Burada; r: zamana göre değişen veya sabit olan referans girişi, y_M: YSA modelinin çıkışı, $\Delta u_c(i)=u_c(i)-u_c(i-1)$: kontrol sinyalindeki artışı, N_1 ve N_2: tahmin ufkunu tanımlar, N_u ise kontrol ufkunu tanımlar. Genellikle [0,1] aralığında olan μ ve λ sabitleri gelecekteki davranışın ağırlıklandırmasını göstermektedirler. N_1'yi sistemin zaman gecikmesine eşit almak uygundur.

Bu kontrol işleminin, sistem çıkışı gerektirmediğinden açık çevrim olduğu düşünülebilir. Ancak, sistem çıkışı sayısal modelde kullanılır. Sistem ve sayısal model arasındaki uyuşmazlık nedeniyle, sistem davranışının, bir süre sonra modeldeki davranıştan sapacağından dolayı bu önemlidir. Bu işlemin önemli bir karakteristiği, bir kontrolör dizaynı gerektirmemesidir, sadece, sistemin ileri beslemeli modeli, YSA tabanlı tahmini kontrolün verimliliğini etkileyecektir. Bu nedenle, bu düşüncede YSA kullanmak doğru olur. Aslında, modelin açık olmama problemi, bu tasarımında azaltılmış bir etkiye sahiptir, çünkü burada en önemlisi modelin dayanıklılığı ve doğruluğudur. Bu kontrol yapısında, YSA çoğunlukla sistemi modellemek için kullanılır. Bu tip bir YSA model tabanlı bir tahmini kontrol yapısı Şekil 5.11'de görülmektedir

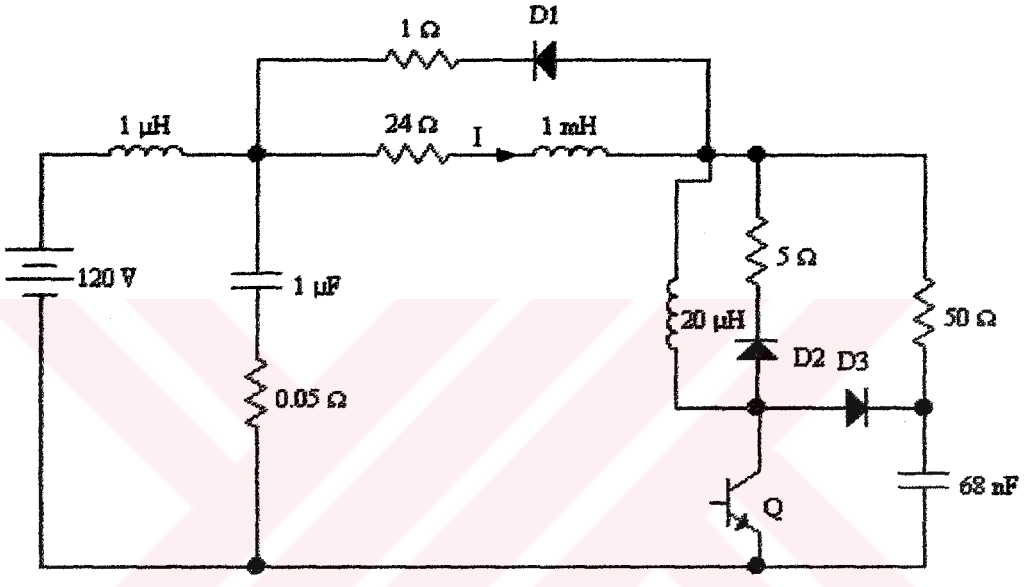


Şekil 5.11 YSA model tabanlı tahmini kontrolör yapısı

BÖLÜM 6. ÖRNEKLER

Örnek 1.

Bu örnekte Şekil 6.1'de görülen koruma elemanlı bipolar tranzistör devresi çok katmanlı İYSA, RTFA ve MYSA yapılarıyla modellenmiştir. Her bir ağın performansına ilişkin bilgiler Tablo 6.1'de verilmiştir. Ayrıca benzetim sonuçlarına ait şekiller eklenmiştir.



Şekil 6.1 Koruma elemanlı bipolar tranzistör devresi

Şekil 6.1'deki devrede kullanılmak üzere giriş olarak, tranzistörün bazına 0-1 V genlikli ve değişken frekanslı kare dalga uygulandı. Çıkış olarak ise 24 ohm'luk dirençten gecen akım seçildi. Örnekleme adımı 0.0002 ms alınarak 1000 data elde edildi ve elde edilen dataların 501 tanesi ağların eğitiminde eğitim örnekleri olarak kullanıldı. Test örneklerini elde etmek için tranzistörün bazına genliği 0-1 V arasında olan 20 kHz frekanslı kare dalga sinyali uygulandı [35]. Elde edilen değerler ağların performansını gözlemlemek için kullanıldı.

Tablo 6.1 Ağlara ilişkin test ve eğitim hataları

Ağ Yapısı	Hata (LNTKH)		
	Test Hatası	Eğitim Hatası	Döngü Sayısı
MYSA	-10.0308	-9.4828	35
İYSA	-14.7948	-6.4475	500
RTFA	-11.3231	-5.7754	--

Bu örnekte; tranzistörün baz sinyali giriş, 24 ohm'luk direnç ve 1 mH'lik endüktansdan geçen akım ise çıkış kabul edilmiştir. Devre beş durum değişkenine sahip olduğundan gelecek çıkışları hesaplamak için giriş ve çıkışın beş gecikmişi kullanılmıştır. Yani gerileme matrisinde beş geçmiş giriş, beş de geçmiş çıkış mevcuttur. Ağların her üçü de toplu öğrenmeyle eğitilip toplu teste tabii tutulmuştur. Ağların performansları ağ çıkışlarıyla gerçek çıkış arasındaki logaritmik olarak normalize edilmiş toplam karesel hata (LNTKH) alınarak gözlemlenmiştir. Bu değerler Tablo 6.1'de görülmektedir.

Kullanılan MYSA yapısı her birinde lineer aktivasyon fonksiyonuna sahip birer nöron bulunan iki uzman ağ ve bünyesinde softmax aktivasyon fonksiyonuna sahip iki nöron bulunduran bir kapılama ağından ibarettir. MYSA tahmini-eğim öğrenme algoritması kullanılarak eğitilmiştir. Tablo 6.1'den de görüldüğü gibi ağın eğitim performansı ile test performansı diğer ağlara göre birbirine daha yakındır. Şekil 6.2a'dan görüldüğü gibi ağın test performansı, çıkış bilgisi doğrular halinde olduğunda ağın tahmin kabiliyetini artırmaktadır.

İYSA yapısı giriş, çıkış ve bir gizli katmandan ibarettir. Gizli katmanda hiperbolik tanjant aktivasyon fonksiyonuna sahip on bir nöron, çıkış katmanında ise lineer aktivasyon fonksiyonuna sahip bir nöron bulunmaktadır. Ağ Levenberg- Marquardt metoduyla eğitilmiştir. Tablo 6.1'den de görüleceği gibi en iyi performansı sağlayan ağdır. Ancak eğitim süresi modüler ağa göre oldukça uzundur. Şekil 6.2b'den görüldüğü gibi İYSA giriş çıkış dönüşümünü her durumda tatmin edici derecede gerçekleştirebilmektedir.

RTFA yapısı 10 girişlidir, gizli katmanında 137 radyal taban fonksiyonlu nörona ve bir lineer çıkış nöronuna sahiptir. Gizli katmandaki nöron sayısı denetimli bir metot ile seçilmiştir. Bu metot da nöron sayısı makul hata seviyesine ulaşılıncaya kadar artırılır. Radyal tabanlı fonksiyonların merkez mesafeleri σ =.001 sabitlenmiştir. Tablo 6.1'den görüldüğü gibi kabul edilebilecek kadar iyi bir performansa sahiptir. Buna karşın ağın eğitim için kullanılan bilgi sınırları dışına çıkıldığında tahmin kabiliyeti zayıflamaktadır. Şekil 6.2c'den görüldüğü gibi ilk 500 değerden sonra ağ çıkışıyla gerçek çıkış arasındaki hata diğer ağların aksine artmaktadır.

Ağ yapılarını özetlenecek olursa:

1) MYSA

   Başlangıç öğrenme oranı: 0.01

   Öğrenme kuralı: Adaptif öğrenme oranı kullanan tahmini-eğim öğrenme kuralı

   Ağ yapısı: 2 uzman ağ, 1 kapılama ağı

   Uzman ağların yapısı : 10-1

   Uzman ağ nöronlarının aktivasyon fonksiyonu: Lineer

   Kapılama ağının yapısı: 10-2

   Kapılama ağ nöronlarının aktivasyon fonksiyonu: Softmax

2) İYSA

Başlangıç öğrenme oranı: 0.1

Öğrenme kuralı: Levenberg-Marquart

Gizli katman nöronlarının aktivasyonu fonksiyonu: Hiperbolik tanjant

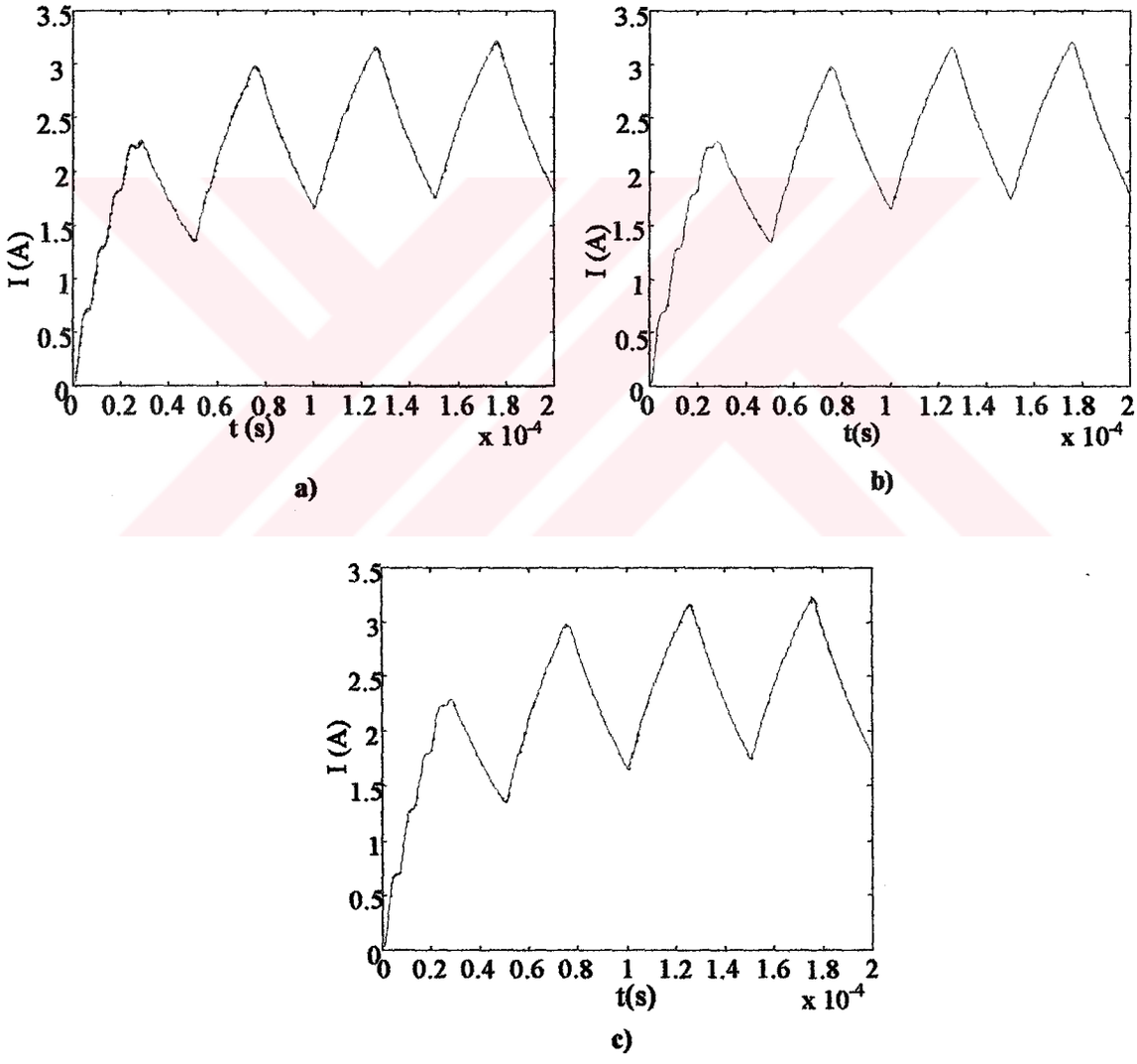
Çıkış katmanı nöronlarının aktivasyon fonksiyonu: Lineer

Ağ yapısı: 10-11-1

3) RTFA

σ parametresi: 0.001

ağ yapısı: 10-137-1



Şekil 6.2 a) MYSA çıkışı ve gerçek çıkış **b)** İYSA çıkışı ve gerçek çıkış **c)** RTFA çıkışı ve gerçek çıkış (siyah çizgiler ağ çıkışları, mavi çizgiler gerçek çıkışlar)

Örnek 2.

Bu örnekte Şekil 6.3'deki otonom Chua devresinin çift sipiralli çekicileri modellenmiştir. Bu devreye ilişkin Durum Denklemleri Denklem (6.1)'deki gibidir.

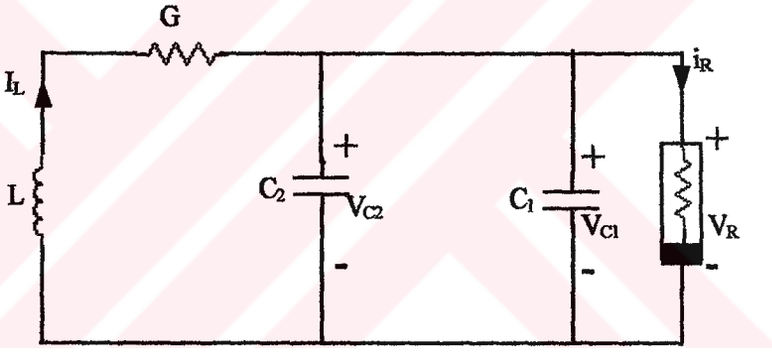
$$C_1.\frac{dV_{c_1}}{dt} = G.(V_{c_2} - V_{c_1}) - g(V_{c_1})$$

$$C_2.\frac{dV_{c_2}}{dt} = G.(V_{c_1} - V_{c_2}) + I_L \qquad (6.1)$$

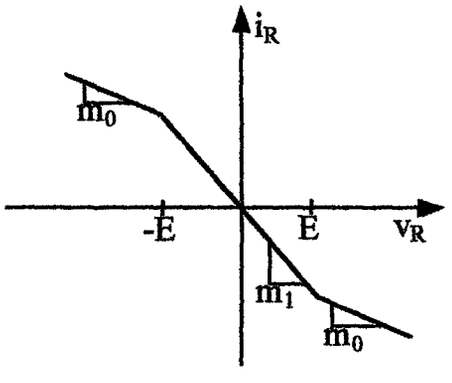
$$L.\frac{dI_L}{dt} = -V_{c_2}$$

Doğrusal olmayan direncin parça-parça lineerleştirilmiş karakteristiği Şekil 6.3a'da görülmektedir. Bu akımı iki kırılma noktasına sahip parça-parça lineer bir fonksiyon olan $g(V_{c1})$ ile Denklem (6.2)'deki gibi ifade edilir.

$$g(V_{c_1}) = m_0.V_{c_1} + 0.5.(m_1 - m_0).(|V_{c_1} + E| - |V_{c_1} - E|) \qquad (6.2)$$



a)



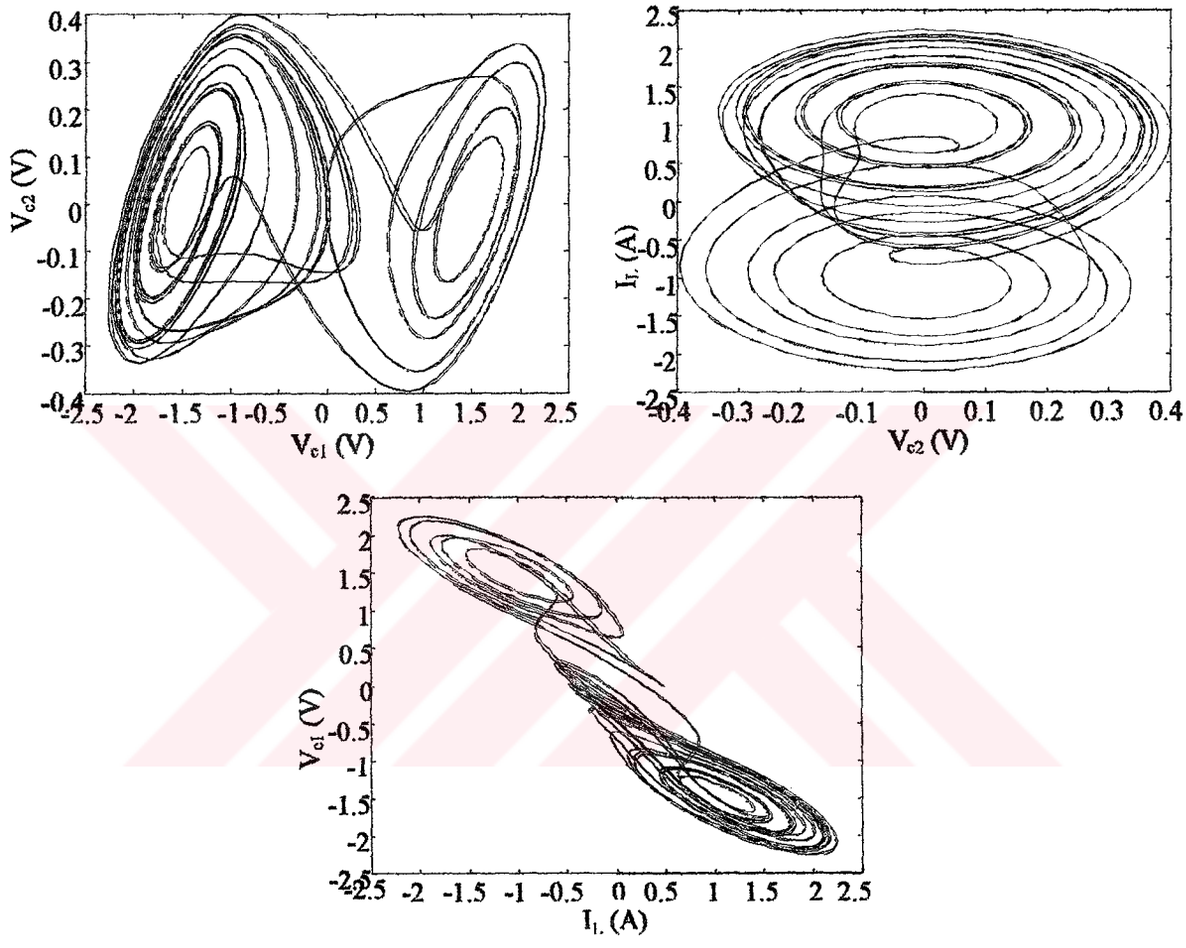
b)

Şekil 6.3. a) Chua Devresi **b)** Lineer olmayan direncin parça-parça akım-gerilim karakteristiği

Direncin akım-gerilim karakteristiği iki lineer parçadan oluşur. Birinci lineer parça [-E,E] değerleri arasındaki gerilim bölgesinde tanımlıdır ve eğimi m_1 ile, ikinci lineer parça ise [-E,E]

değer aralığı dışındaki gerilim bölgesinde tanımlıdır ve eğimi m_0 ile gösterilir. Şekil 6.3a'daki Chua devresine ilişkin büyüklükler şöyledir; m_1=-0.5 1/Ω, m_0=-0.8 1/Ω, E=1 V, G=0.7 1/Ω, L=7 H, C_1=9 F, C_2=1 F.

Chua devresinin modellenmesi için yapılmış farklı çalışmalar mevcuttur. Bu örnekte ağ I_L, V_{c1}, V_{c2}'den oluşan üç çıkış ve bunların geciktirilmişlerinden oluşan 12 girişe sahiptir [36].



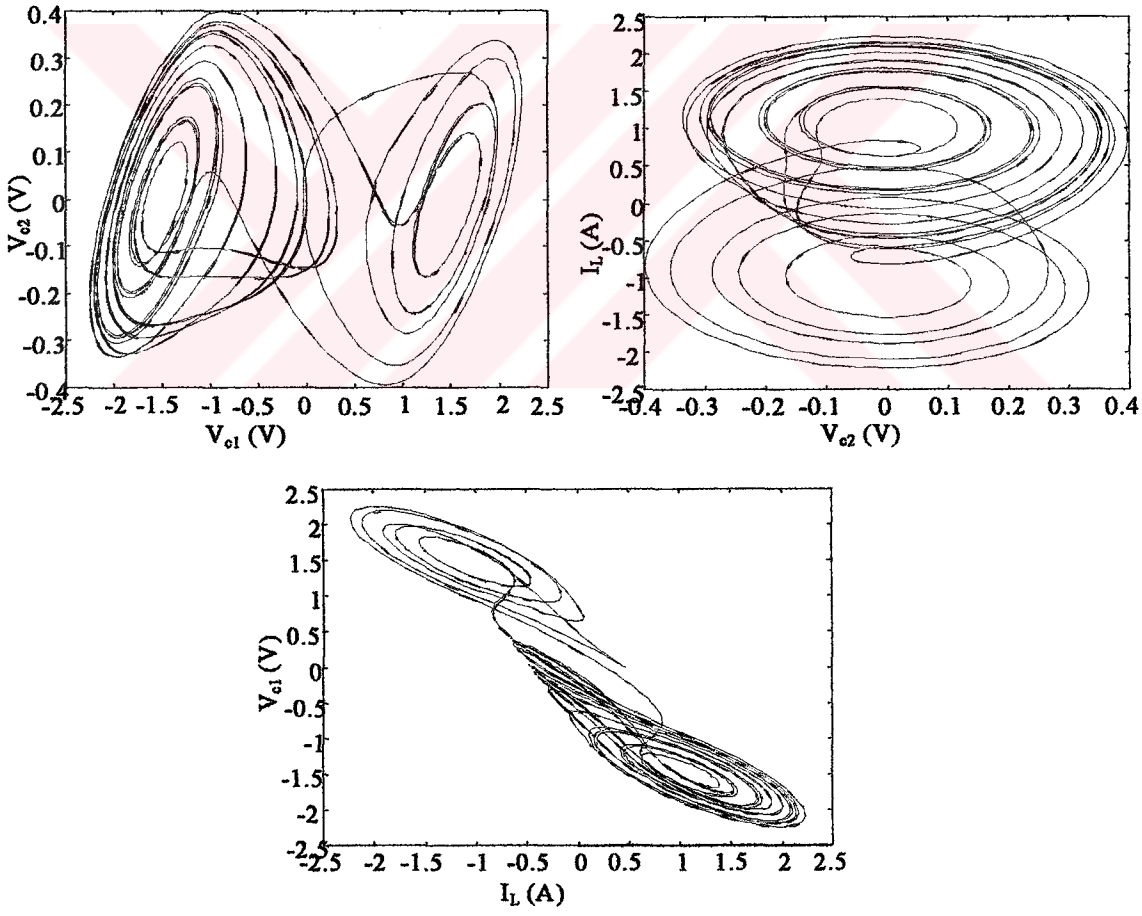
Şekil 6.4. MYSA modelin ve chua devresinin çift sipiralli çekicileri
(siyah ağ çıkışı, mavi gerçek çıkış)

Denklem (6.1)'deki Durum Denklemleri $V_{C1}(0)$=0.9365V, $V_{C2}(0)$=-0.061V $I_L(0)$=-0.1883A başlangıç şartları ve Runge-Kutta Metodu ile çözülmüştür. 0-70 s zaman aralığında 1750 data elde edilmiştir. Bu dataların ilk 500 tanesi ağların eğitiminde kullanılmıştır. Daha sonra başlangıç şartları $V_{C1}(0)$=-0.9365V, $V_{C2}(0)$=0.061V, $I_L(0)$=0.1883A alınarak çözüm tekrarlanmış ve elde edilen datalar test için kullanılmıştır. Ağların eğitimi ve testi toplu öğrenmeyle yapılmıştır.

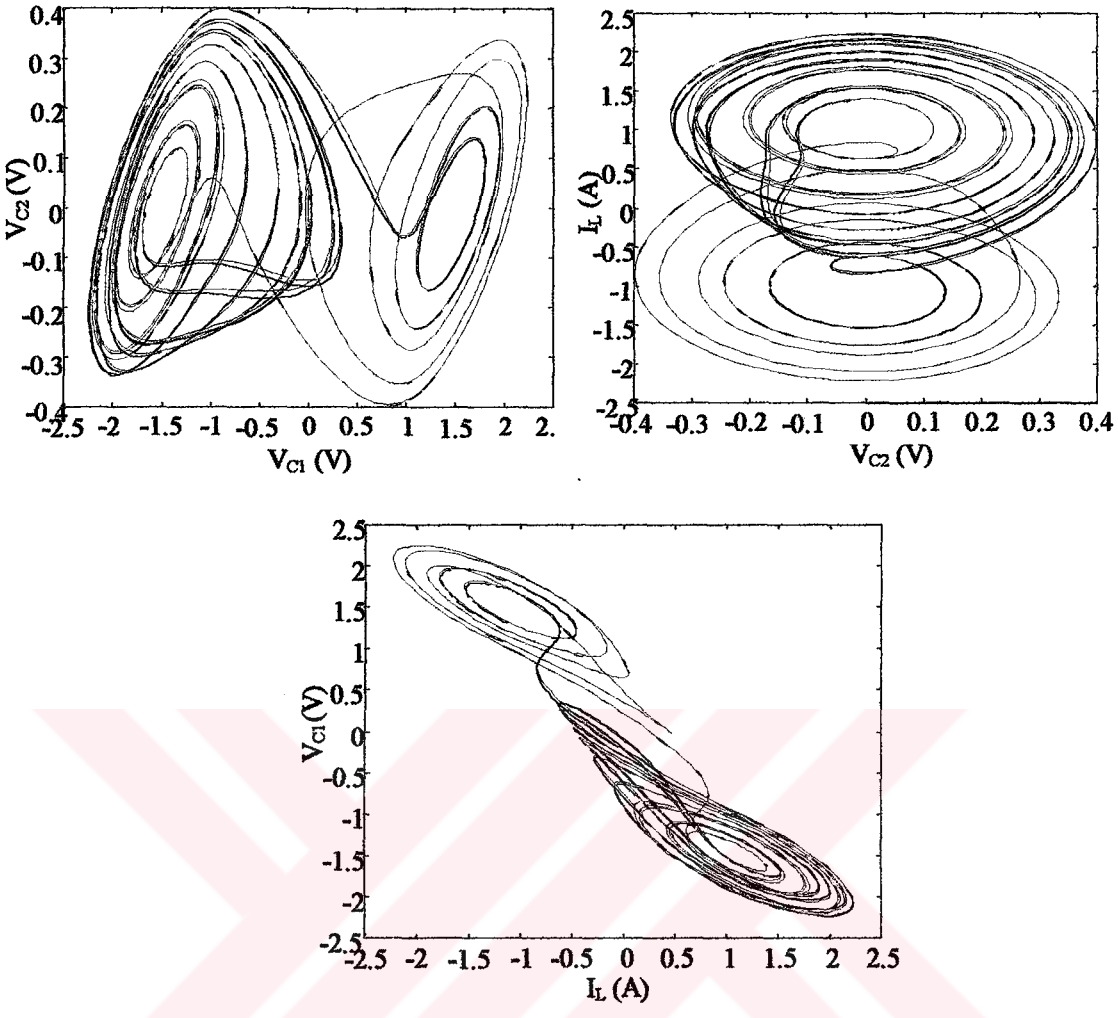
Tablo 6.2. Ağlara ait test ve eğitim hataları

Ağ	Test Hatası (LNTKH)			
	V_{c1}	V_{c2}	I_L	Döngü Sayısı
MYSA	-3.6204	-6.8418	-6.7169	55
İYSA	-4.5788	-7.8906	-8.0557	500
RTFA	-3.9763	-5.1284	-4.6443	--

Bu örnekte Levenberg-Marquardt metoduyla eğitilmiş olan çok katmanlı İYSA en iyi sonucu vermektedir. Gizli katmandaki nöron sayısı denetimli bir metot ile seçilmiştir. Bu metot da nöron sayısı makul hata seviyesine ulaşılıncaya kadar artırılır. RTFA için en iyi sonuc merkez uzaklıkları 0.0001 alındığında elde edilmiştir.



Şekil 6.5 İYSA modelin ve chua devresinin çift sipiralli çekicileri
(siyah ağ çıkışı, mavi gerçek çıkış)



Şekil 6.6 RTFA modelin ve chua devresinin çift sipiralli çekicileri (siyah ağ çıkışı, mavi gerçek çıkış)

Ağ yapılarını özetleyecek olursak:

1) MYSA

Başlangıç öğrenme oranı: 0.001

Öğrenme kuralı: Adaptif öğrenme oranı kullanan tahmini-eğim öğrenme kuralı

Ağ yapısı: 3 uzman ağ, 1 kapılama ağı

Uzman ağların yapısı: 12-3

Uzman ağ nöronlarının aktivasyon fonksiyonu: Lineer

Kapılama ağının yapısı: 12-3

Kapılama ağ nöronlarının aktivasyon fonksiyonu: Softmax

Kapılama ağının yapısı: 12-3

2) İYSA

Başlangıç öğrenme oranı: 0.1

Öğrenme kuralı: Levenberg-Marquart

Gizli katman nöronlarının aktivasyonu fonksiyonu: Hiperbolik tanjant

Çıkış katmanı nöronlarının aktivasyon fonksiyonu: Lineer

Ağ yapısı: 12-13-3

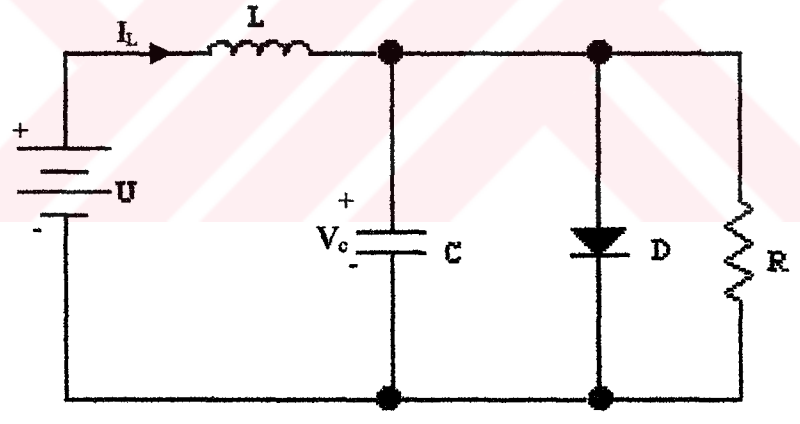
3) RTFA

σ parametresi: 0.0001

Ağ yapısı: 12-84-3

Örnek 3.

Bu örnekte öğrenme algoritmalarının performansını ölçmek için Şekil 6.7'de görülen tünel diyot osilatör devresi ele alınmıştır. Devreye ilişkin durum denklemleri Denklem (6.3)'de verilmiştir. Devreye giriş olarak genliği 0-1 V arasında rasgele değişen bir sinyal uygulanıp, çıkış olarak kondansatör üzerindeki gerilim, Vc, seçilmiştir.



Şekil 6.7 Tünel diyotlu RLC devresi

$$\frac{dV_c}{dt} = I_L$$
$$\frac{dI_L}{dt} = -V_c + I_L.(1.4 - 0.14.I_L^2) + 4.U \qquad (6.3)$$
$$V_c(0) = I_L(0) = -5$$

Bu örnekte Şekil 6.7'deki devreye ait olan Denklem (6.3)'deki durum değişkenleri 0-10 s zaman aralığında dört adımlı Runge-Kutta metodu ile çözülerek 1000 data elde edilmiştir.

Eğitim için bu dataların ilk 500 tanesi, test için ise tüm datalar kullanılmıştır. Devrenin İYSA modelinin elde edilmesi ve bu modelin farklı eğitim algoritmalarıyla eğitilerek performansların karşılaştırılması amaçlanmıştır. Eğitim için Ardışıl Geriye Yayılım, Levenberg-Marquardt, eşlenik iniş ve Gausi-Newton eğitim algoritmaları kullanılmıştır. Tablo 6.3'de her bir algoritmanın performansı LNTKH olarak verilmiştir.

Tablo 6.3 Eğitim algoritmalarına ilişkin eğitim hataları

Algoritma	Hata (LNTKH)	
	Hata	Döngü sayısı
Ardışıl Geriye Yayılım	-1.9896	2000
Levevberg-Marquardt	-6.8721	500
Eşlenik iniş	-3.3064	500
Guasi-Newton	-5.2372	500

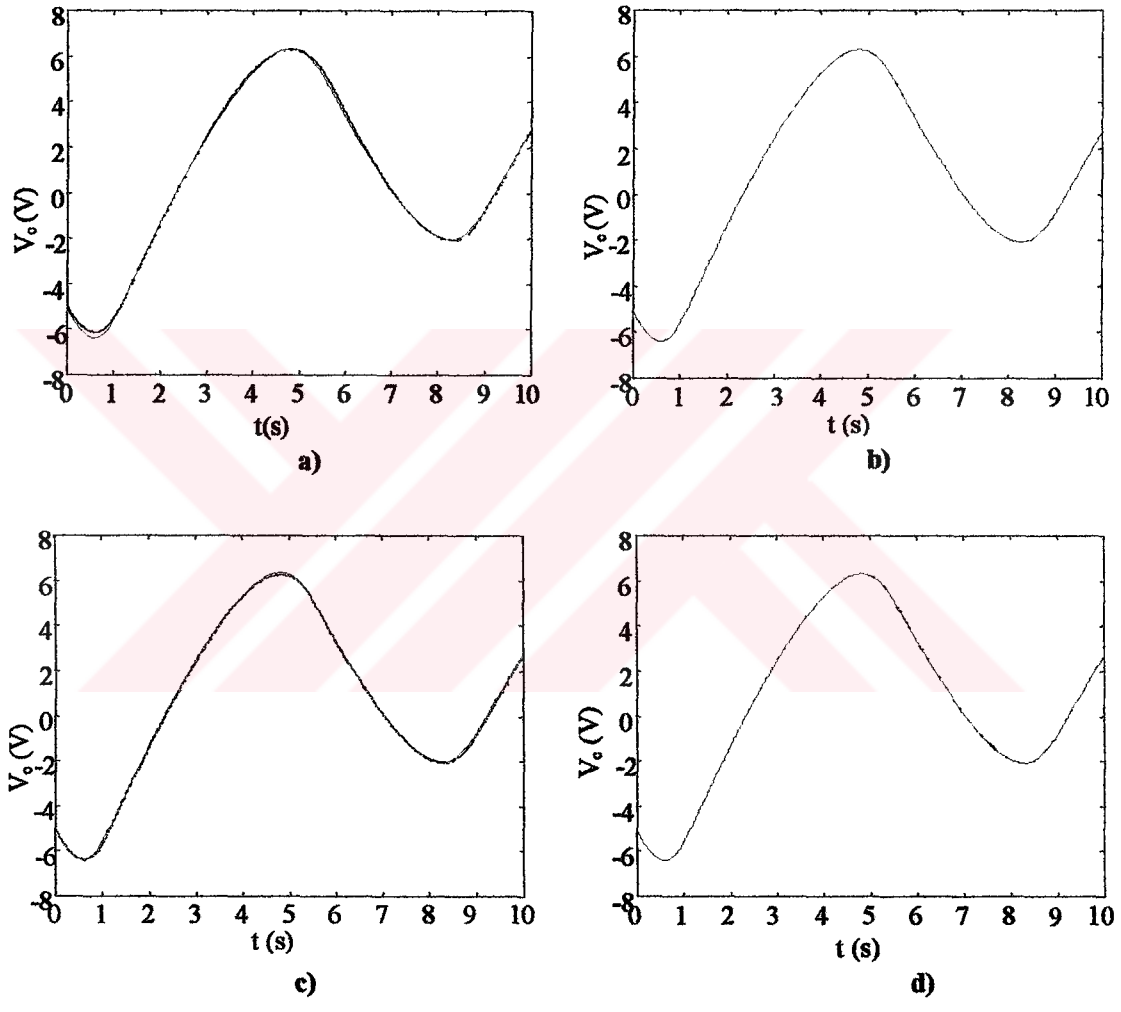
Ağ yapısını belirlemek için alternatif bir çözüm sistemin fark denklemlerine bakmaktır [29]. Burada ağ yapısı olarak devre ikinci derece bir sistemi temsil ettiğinden, dolayı, durum uzayı modeli elde edilirken iki geciktirilmiş çıkış, bir giriş birde geciktirilmiş giriş kullanılmıştır. Bundan dolayı YSA modelin giriş vektörü (1x4), çıkış vektörü ise (1x1) boyutlu oluşturulmuştur. Yani, giriş sayısı 4, gizli katmandaki nöron sayısı 5, çıkış katmanındaki nöron sayısı 1 şeklindedir. Gizli katmanda hiperbolik tanjant aktivasyon fonksiyonu, çıkış katmanında lineer aktivasyon fonksiyonu kullanılmıştır.

İYSA'nın ardışıl geriye yayılım algoritmasıyla eğitiminde sabit öğrenme oranı 0.0001 olarak seçilmiş. Ardışıl Geriye Yayılım algoritmasıyla eğitilmiş İYSA modelin ve devrenin çıkışı Şekil 6.8a'da gösterilmiştir. Öğrenme oranının daha fazla küçülmesiyle yakınsamanın geciktiği, gereğinden fazla büyümesi durumunda İYSA'nın yakınsıyamadığı gözlenmiştir.

Levenberg–Marquardt algoritmasında ise başlangıç öğrenme oranı 0.01 olarak seçilerek sonuca gidilmiştir. Şekil 6.8b'de Levenberg–Marquardt algoritmasıyla eğitilmiş İYSA modelin ve devrenin çıkışı görülmektedir. Başlangıç öğrenme oranı gereğinden daha büyük seçildiğinde öğrenmenin gerçekleşemediği, eğitimin daha başında ıraksamanın oluştuğu gözlenmiştir.

Eşlenik iniş algoritmasında ise yön vektörünün zaman değişim sabiti tespiti Fletcher-Reeves formülüne göre yapılıp, öğrenme oranı 0.001 seçilmiştir. Öğrenme oranının büyük seçilmesiyle hata yüzeyindeki zikzakların büyüdüğü ve ağın yakınsamasının güçleştiği gözlenmiştir. Bu algoritmayla eğitilmiş İYSA modelin ve devrenin çıkışı Şekil 6.8c'deki gibidir.

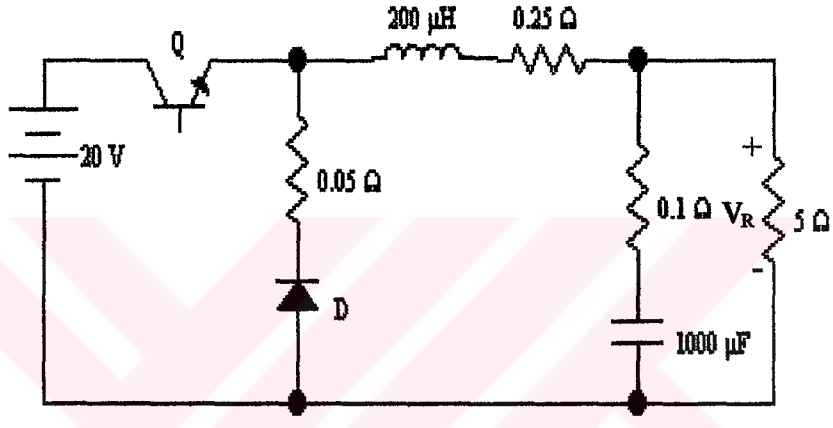
Gauss-Newton algoritmasında unutma faktörlü yenileme metodu kullanılmış ve bu metoda ilişkin parametreler şöyle seçilmiş. Kovaryans matrisinin P'nin başlangıç değeri 10, unutma faktörü ise 0.995 olarak alınmıştır. Bu algoritmayla eğitilmiş İYSA modelin ve devrenin çıkışı Şekil 6.8d'de görülmektedir. Diagonal elemanların değerlerinin küçük seçilmesi durumunda P'nin öz değerlerinin sıfıra ulaştığı ve eğitimin gerçekleşmediği, 0 ila 1 arasında olması gereken unutma faktörünün küçük seçilmesi durumun da ise eğitimin yavaşladığı gözlenmiştir.



Şekil 6.8 İYSA modelin çıkışları ve hedef çıkışlar **a)** Ardışıl geriye yayılım **b)** Levenberg-marquardt **c)** Conjugate-gradient **d)** Guasi-newton (siyah ağ çıkışı, mavi gerçek çıkış)

Örnek 4.

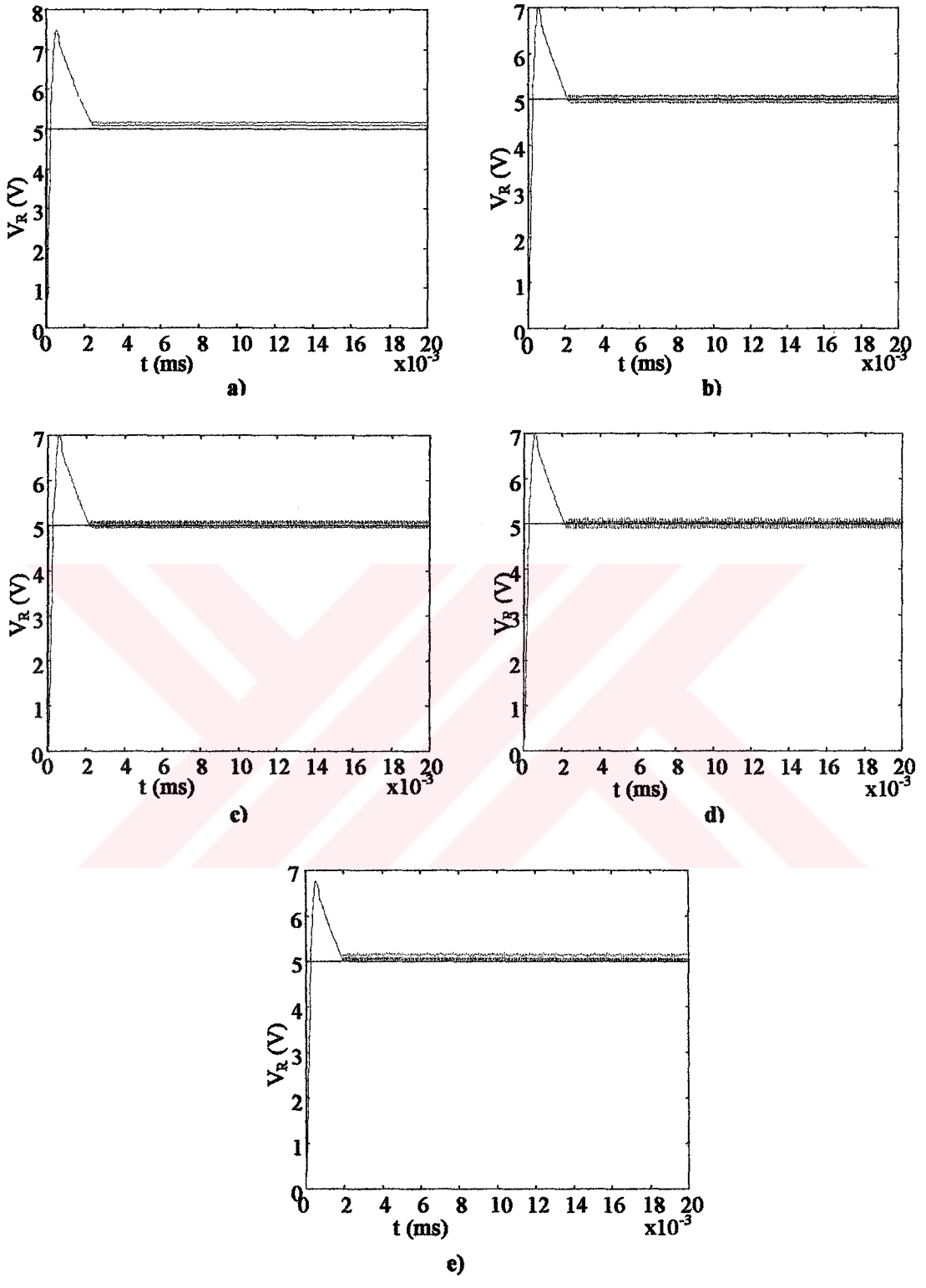
Bu örnekte kontrol yapılarının performanslarını karşılaştırmak için Şekil 6.9'da görülen konvertör devresi ele alınmıştır. Burada 5 ohm'luk direncin gerilimi V_R kontrol edilmiştir. Giriş olarak tranzistörün baz uçuna uygulanan sinyal alınmıştır. Bu örnekle direkt ters nöral-kontrol, içten YSA model tabanlı kontrol, YSA tabanlı optimal kontrol ve YSA tabanlı tahmini kontrol yapılarına ilişkin performansların karşılaştırılması amaçlanmıştır. Devrenin Matlab benzetim modeli oluşturulmuş ve giriş olarak 0-1V genlikli rasgele dağılımlı bir sinyal kullanılarak eğitim örnekleri elde edilmiştir. Bu örnekler nöral-kontrol yapılarında kullanılan model ve kontrolör ağlarının modelinin eğitiminde kullanılmıştır.



Şekil 6.9. Titreşimli konvertör devresi

Tüm kontrol yapılarında model ve kontrolör ağlar İYSA kullanılarak gerçekleştirilmiştir. Ele alınan devre 2. dereceden sistem davranışı gösterdiği için hem nöral kontrolör hem de devrenin YSA modeli 4-5-1 yapısında oluşturulmuştur.

İlk olarak devrenin ters davranışını modelleyen nöral kontrolör, Levenberg-Marquardt algoritması kullanılarak direkt ters öğrenme metoduyla toplu olarak eğitilir. Elde edilen sonuç Şekil 6.10a'da görülmektedir. Daha iyi bir sonuca ulaşabilmek için eldeki nöral kontrolör, özelleştirilmiş ters öğrenme metoduyla tekrar eğitilir. Bu öğrenme metodunda devrenin YSA modeline de ihtiyaç vardır. Devrenin YSA modeli olarak, Levenberg-Marquardt algoritmasıyla toplu eğitilmiş bir İYSA alınır. Bu eğitim Gauss-Newton aloritmasıyla örneksel olarak gerçekleştirilir. İkinci kez eğitilmiş olan nöral kontrolörün kullanıldığı direkt ters nöral-kontrol yapısıyla elde edilen sonuç Şekil 6.10b'de görülmektedir.



Şekil 6.10 Nöral kontrol yapılarına ilişkin sonuçlar **a)** Direkt ters öğrenmeyle eğitilmiş direkt ters nöral-kontrol **b)** Özelleştirilmiş ters öğrenmeyle eğitilmiş direkt ters nöral-kontrol **c)** İçten YSA model tabanlı kontrol **d)** YSA tabanlı optimal kontrol **e)** YSA model tabanlı tahmini kontrol (siyah hedef çıkış, mavi gerçek çıkış)

Aynı nöral kontrolör, devrenin YSA modeliyle beraber, içten YSA model tabanlı kontrol yapısında kullanılmıştır. Elde edilen çıkış Şekil 6.10c'de görülmektedir. YSA tabanlı optimal kontrol yapısı kullanılarak elde edilen çıkış ise Şekil 6.10d'de verilmiştir. Şekil 6.10e'de ise YSA model tabanlı tahmini kontrol yapısıyla elde edilen çıkış görülmektedir.

Örnek 5.

Bu örnekte YSA kontrol yapılarıyla klasik kontrol yapılarını karşılaştırmak için Durum Denklemleri (6.4)'de verilen bir DC motor ele alınmış ve pozisyon kontrolü hedeflenmiştir (Messner ve Tilbury,1998).

$$\frac{di}{dt} = \frac{1}{L}(-R.i + U - K.\frac{d\theta}{dt})$$
$$\frac{d^2\theta}{dt^2} = \frac{1}{j}(K.i - b.\frac{d\theta}{dt}) \qquad (6.4)$$

DC motora ilişkin parametreler aşağıdaki gibidir.

Rotorun atalet momenti $J=3.2284\text{x}10^{-6}$ kg.m^2/s^2

Mekanik sistemin aşma oranı $b=3.5077\text{x}10^{-6}$ Nms

Elektromotor kuvvet sabiti K=0.0274 Nm/Amp

Elektriksel direnç R=4Ω

Elektriksel indüktans $L=2.75\text{x}10^{-6}$H

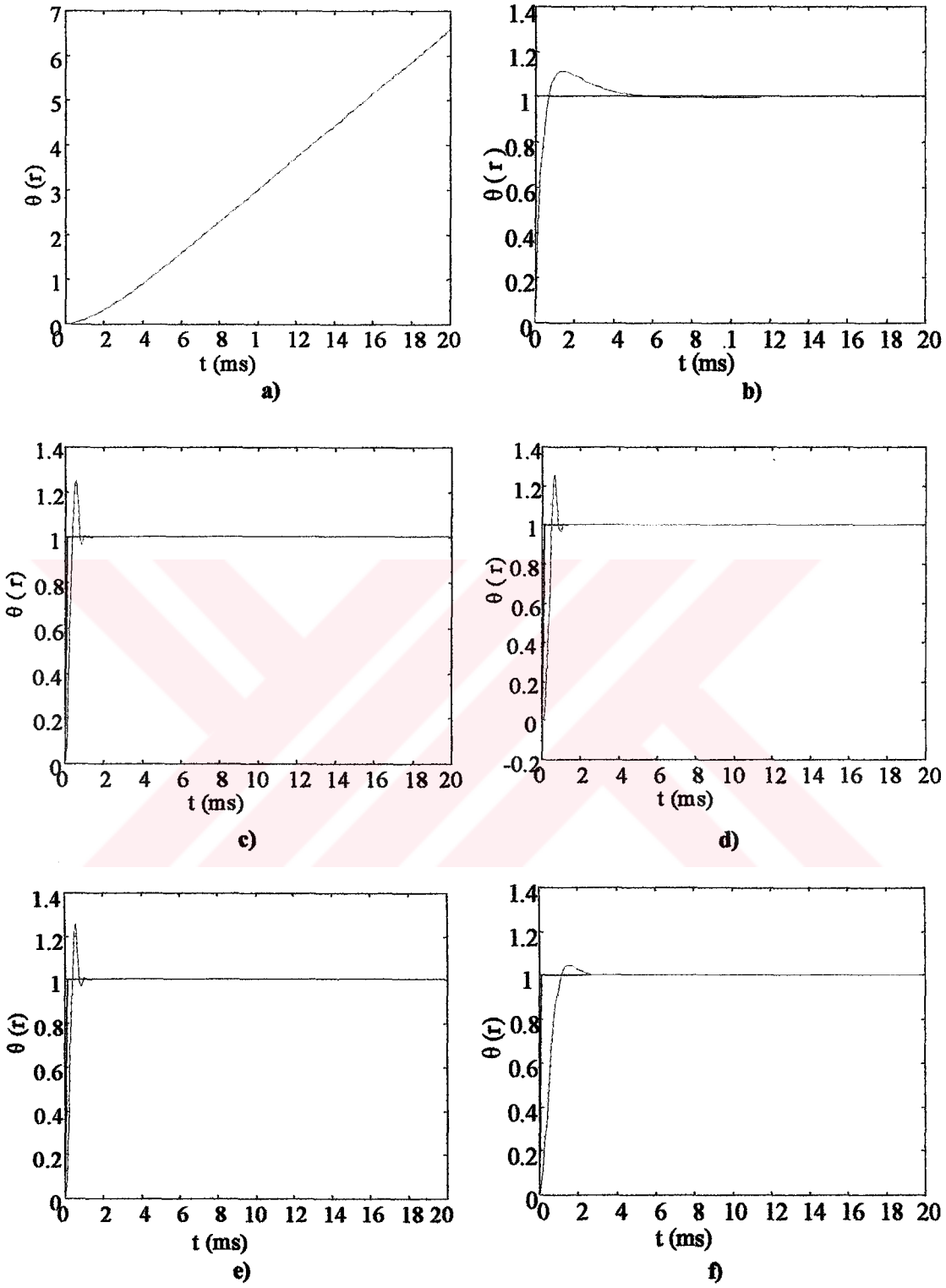
Giriş U Kaynak gerilimi (V)

Çıkış θ= Pozisyon açısı (radyan)

Durum Denklemlerinden anlaşılacağı gibi DC motor ikinci derece bir sistem gibi davranışa sahip olduğu için kontrolör ve model İYSA'ların ağ yapıları 4-5-1 biçiminde oluşturulmuştur. DC motorun YSA modeli Levenberg-Marquardt algoritmasıyla toplu olarak eğitilmiştir. Direkt ters nöral-kontrol ve içten YSA model tabanlı kontrol yapılarında kullanılan nöral kontrolör ise önce Levenberg-Marquardt algoritmasıyla toplu, daha sonra performansını iyileştirmek için Gauss-Newton algoritması ile örneksel olarak eğitilmiştir. YSA tabanlı optimal kontrol yapısında kullanılan ağlar ise Bölüm 5.3'de açıklandığı gibi eğitilmiştir. YSA model tabanlı tahmini kontrol yapısında kullanılan sistemin YSA modeli yukarıda verilen ağ yapısında olup, Levenberg-Marquardt metoduyla toplu olarak eğitilmiştir.

Klasik kontrol tekniklerinden PID kontrolörün kazanç katsayıları şu kriterlere göre seçilmiştir; yerleşme zamanı 0.04 s'den az, maksimum aşma %16'dan az, sürekli hal hatası 0. Bu kriterleri sağlayan PID kazanç katsayıları K_p=17, K_i=600, K_d=0.15 olarak saptanmıştır [37].

Şekil 6.11'de DC motorun çeşitli durumlar için pozisyon cevapları verilmiştir. Şekil 6.11a'da birim basamak giriş için DC motorun açık çevrim cevabı görülmektedir. Şekil 6.11b'de PID kontrolör kullanılarak birim basamak giriş için motorun kapalı çevrim cevabı görülmektedir. Şekil 6.11b'den görüldüğü gibi elde edilen cevap kontrolör parametreleri seçilirken temel alınan kontrol esaslarını karşılamaktadır. Ancak kontrolör parametreleri tespit edilirken sistemin bilinen dinamiğinden faydalanıldığı unutulmamalıdır. Şekil 6.11c'de ise özelleştirilmiş öğrenmeyle elde edilmiş nöral kontrolör kullanan direkt ters nöral-kontrol yapısıyla elde edilen cevap görülmektedir. İçten YSA model tabanlı kontrol yapısı kullanılarak elde edilen cevap Şekil 6.11d'de verilmiştir. Şekil 6.11e'de ise YSA tabanlı optimal kontrol yapısıyla elde edilen cevap görülmektedir. Bu yapılarla elde edilen cevaplarda maksimum aşma şartının sağlanamadığı görülür. Şekil 6.11f'de görülen YSA model tabanlı tahmini kontrol yapısıyla elde edilen sonucun ise en iyi sonuç olduğu açıktır. Bu yapının başarısı kullanılan YSA sistem modelinin başarımıyla orantılıdır.



Şekil 6.11 a) DC motorun açık çevrim birim basamak cevabı
b) PID kontrolör ile elde edilen cevap
c) Direkt ters nöral-kontrol yapısıyla elde edilen cevap
d) İçten YSA model tabanlı kontrol yapısıyla elde edilen cevap
e) YSAtabanlı optimal kontrol yapısıyla elde edilen cevap
f) YSA model tabanlı tahmini kontrol yapısıyla elde edilen cevap

BÖLÜM 7. SONUÇLAR

Bu tezde değişik YSA yapıları ve farklı öğrenme algoritmaları incelenmiştir. Ağ yapılarının değişik performanslarda doğrusal olmayan dinamiğe sahip devreleri modellediği ve bu modellerin kontrol amacıyla kullanılabildiği gösterilmiştir. YSA'lar ile yapılan modellemelerde devre veya sistemlerin sadece giriş-çıkış ilişkisini veren bilgi kullanılarak modelleme işlemi başarıyla gerçekleştirilmiştir. Buradan sistemlerin iç dinamikleri bilinmeden sayısal datalar kullanılarak modellenebileceği sonucuna ulaşılmıştır. Ancak bu devreler için uygun bir örnekleme periyodu seçimi modellemenin başarısı için önemlidir. Örnekleme periyodunun büyük alınması devrenin davranışını tanımlayacak yeterli bilginin alınamamasına, çok küçük seçilmesi ise eğitim için fazla data kullanılacağından ilave hesapsal yüke neden olmaktadır. Devrelerin modellenmesi için üç çeşit YSA yapısı kullanılmıştır. Bunlardan İYSA'ların oldukça iyi performans gösterdikleri buna karşın eğitimleri için daha fazla zaman gerektiği gözlenmiştir. MYSA'lar ise İYSA'lar kadar olmasa da iyi bir performans sergilemektedirler ve eğitimleri için harcanması gereken zaman daha az olmaktadır. Ayrıca doğrular halinde süreksiz davranışlar gösteren sistemlerin modellenmesinde performansları artmaktadır. Bunun yanında uygun eğitim yapılarak sistemin kompleks olan dinamiğinin her bir parçasının ayrı bir uzman ağ tarafından öğrenilmesi sağlanabilir. Bu da analiz imkanı vermesi bakımından önemlidir. RTFA'nın yüksek eğitim başarımına rağmen giriş uzayının değişiminde yüksek hatalar ürettiği görülmektedir. İYSA'ların eğitimi için Levenberg-Marquardt, ardışıl geriye yayılım, Eşlenik-eğim ve Gauss-Newton eğitim algoritmaları kullanılmış en iyi sonuç Levenberg-Marquardt metoduyla elde edilmiştir.

Bu çalışmada nöral-kontrol yapıları da incelenmiş ve sistemlerin dinamik yapılarına ilişkin sistemin jacobiani gibi güçlü matematiksel bilgiler olmadan kontrollerinin mümkün olduğu görülmüştür. Nöral-kontrol yapılarından direkt ters nöral-kontrol yapısı, içten YSA model tabanlı kontrol yapısı, YSA tabanlı optimal kontrol yapısı ve YSA model tabanlı tahmini kontrol yapısı bu çalışmada incelenmiştir. Tüm kontrol yapılarında İYSA'lar kullanılmıştır. Nöral- kontrol yapılarının incelenmesi sonucu bu kontrol yapılarıyla elde edilen performansın kullanılan YSA'ların modelleme yetenekleriyle doğru orantılı olduğu görülmüştür. Nöral-kontrol yapıları karmaşık hesaplamalara gerek kalmaksızın kontrol işlemi gerçekleştirebilmektedir. Ancak bu yapılar kullanılarak gerçekleştirilen kontrol dizaynında geçici rejime ilişkin sınırlamaları gerçekleştirmek her zaman mümkün olmamaktadır. Sürekli rejim kriterleri çoğunlukla başarıldığı halde geciçi rejime ilişkin kriterlerin başarımına tasarımcı farklı YSA'lar veya öğrenme algoritmaları kullanılarak müdahale edebilmektedir. Bunun

başlıca sebebi nöral-kontrolörlerin basit bir optimizasyon algoritması kullanılarak eğitilmesidir. Bu çalışmada en iyi başarımın YSA model tabanlı tahmini kontrol tapısıyla gerçekleştirildiği görülmektedir, ancak bu yapıda da her örnekleme adımında bir optimizasyon gerçekleştirildiğinden hesaplama hızı ile ilgili problemler vardır. Diğer yapılarda böyle bir sorun yoktur ve dinamiği hızlı değişen sistemlere başarıyla uygulanabilirler.

KAYNAKLAR

1. Karna, K. ve Breen D., 1989. An artificial neural networks tutorial :part1-basics, Neural Netwoks, v 1 n 1 p 4-23.
2. Sqrensen O., 1998. Systems Identification, Prediction Simulation and Control with Neural Networks, Proceedings of Neural Networks and Their Applications, Marseille, 11-13, March, p 53-60.
3. Jie Z., Julian M., 1999. Recurrent Neuro-Fuzzy Networks for Nonlinear Process Modeling. IEEE Trans. on Neural Networks, v 10 n 2 p 313-326.
4. Kalogirov, S.,2000, Applications of artifical neural networks for energy systems, Applied energy 67, 17-35.
5. Lizarraga,I.and Etxebarria,U.,1998, Real-time adaptive neural control of a class of nonlinear systems, Engineering applications of artifical intelligence 12, 3-15.
6. Leenaerts, D.M.W.and Bokhoven,W.M.G.,1998, *Piecewise linear modelling and analysis.* Eindoven university of technology.
7. Ronco,E. and Gawthrop,P.J., 1997, Neural networks for modelling and control, University of Glasgow, Technical Report.
8. Efe, M. Ö, 1996. Identification and Control of Nonlinear Dynamical Systems Using Neural Networks, B.Ü. Systems and Control Engineering, M.S. Thesis.
9. Haykin, Simon, 1994, *Neural Networks*. Macmillan College Publishing Company, Newyork.
10. Zurada, Jacek M., 1992, *Introduction to Artificial Neural Systems*. Princes Hall.
11. Reich,Y.and Barai,S.,2000, A methodolgy for building neural networks models for empirical engineering data, Engineering applications of artifical intelligence 13, 685-694.
12. Gökbulut, M., 1998. Fırçasız Doğru Akım Motorlarının Yapay Sinir Ağları ile Uyarmalı Denetimi, Doktora tezi, Erciyes Univesitesi Fen Bilimleri Enstitüsü.
13. Zheng,G.L.and Billings,S.A.,1995, Radial basis function network configuration using mutual information and orthogonal least squarse algorithm, Neural Networks vol.9, no.9.
14. Yingwei,L., Sundararajan,N. and Saratchandran,P.,1998, Adaptive nonlinear system identification using minimal radial basis function neural networks, Nanyang Technological University.
15. Fung,C., Billings,S.A.and Luo,W.,1996, On-line supervised adaptive training using radial basis function networks, Neural Networks vol.9, no.9, pp.1597-1617.

16. Kecman,V.,1996, System identification using modular neural network with improved learning, IEEE Computer Society Press, Venice, pp-40-48.
17. Jordan,M.I.and Jacobs,R.A., 1994, Hierarchical mixtures of experts and EM algorithm, Neural Computation, 6, 181-214.
18. Neal,R.M.and Hinton,G.E.,1997, A view of the EM algorithm that justifies incremental, sparse and other variants.
19. Meila,M.and Jordan,M.I., 2000, learning with mixtures of trees, Journal of machine learning research , oct 11.
20. Ronco,E. and Gawthrop,P.J., 1995, Modular neural networks:a state of the art, University of Glasgow, Technical Report .
21. Ronco,E., Gollee,H.and Gawthrop,P., 1997,Modular neural networks and self decomposition, University of Glasgow, Technical Report.
22. Ronco,E.,Gawthrop,P.J. and Hill,D.J.,1998, Gated modular neural networks for control oriented modelling, University of Glasgow, Technical Report.
23. Hines, J. W., 1997, *Fuzzy and Neural Approaches in Engineering MATLAB Supplement*. Prentice Hall.
24. Stroeve,S., 1998, An analysis of learning control by backpropagation through time, neural networks 11, 709-721.
25. Krose, B., Smagt, P. V., 1996, *An Introduction to Neural Network.* Book of Amsterdam of University.
26. Yao,X., 1999, Evolving artifical neural networks, proceedings of the IEEE, vol.87, no.9,sep.
27. Nϕrgaard, M., 2000. Neural Network Based Systems Identification for Use with Matlab, Technical Report 00-E-891, Department of Automation Technical University of Denmark.
28. Fu,L.,Hsu,H.H.and Principe,1996,İncremental Backpropagation learning networks, İEEE Transactions on Neural Networks, Vol.7, No.7, May.
29. Nϕrgaard, M., 2000. Neural Network Based Control systems Design Toolkit for Use with Matlab, Technical Report 00-E-891, Department of Automation Technical University of Denmark.
30. Uçar, A., 1999. Fuzzy-Nöral ve Yapay Sinir Ağları ile Lineer Olmayan Sistemlerin Modellenmesi ve Kontrolü, Yüksek Lisans Tezi, Fırat Üniversitesi Fen Bilimleri Enstitüsü.
31. Zamarreno,J.M.and Vega,P.,1998, Neural predictive control; application to a highly non-linear system, Engineering applications of artifical intelligence 12, 149-158.
32. Lee,T.H., Tan,K.K., Huang,S.N.and Dou,H.F.,2000, İntelligent control of precision linear actuators, Engineering applications of artifical intelligence 13, 671-684.

33. Xu,X.and Hines,J.W.,1996, Sensor calibration monitoring and fault detection using neural networks based techiques.
34. Saman,M.,Uçar,A.and Demir,Y.,2001, Yapay sinir ağları ile optimal kontrol, Elektrik-Elektronik-Bilgisayar Mühendisliği 9.ulusal kongresi, Kocaeli, s.468-471.
35. Bedrosian,D.and Vlach,J.,1992, Time-domain analysis of networks with internally controlled switches, İEEE Trans.Circuits Syst., Vol.39, No.3, 199-212.
36. Uçar,A.and Demir, Y., 2001, Chua devresinin yapay zeka teknikleri ile modellenmesi, Elektrik-Elektronik-Bilgisayar Mühendisliği 9.ulusal kongresi, Kocaeli, s.505-508.
37. Messner, B. and Tilbury,D., 1998, *Control tutorials for matlab and simulink.* Addison-wesley Publishing copany.inc.

ÖZGEÇMİŞ

1970'de Elazığ'da doğdum. İlk ve orta öğrenimimi bu ilde tamamladıktan sonra, 1992 yılında İ.T.Ü. Elektrik Mühendisliği bölümünden mezun oldum. 1993 yılında Fırat Üniversitesi Teknik Bilimler Meslek Yüksek Okulunda öğretim görevlisi olarak çalışmaya başladım ve halen aynı görevimi sürdürmekteyim. 1999 yılında Fırat Üniversitesi Mühendislik Fakültesi Elektrik-Elektronik Ana Bilim dalında Yüksek lisans Çalışması yapmaktayım.