T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

NÖRO-OPTİMAL KONTROL

DOKTORA TEZİ

Elektronik ve Haberleşme Yük. Müh. Yaşar BECERİKLİ

Enstitü Anabilim Dalı : ELEKTRİK VE ELEKTRONİK MÜH.
Enstitü Bilim Dalı : ELEKTRONİK

Bu tez 26 / 10 / 1998 tarihinde aşağıdaki jüri tarafından Oybirliği/~~Oyçokluğu~~ ile kabul edilmiştir.

Prof.Dr. A.Ferit Konar	Prof.Dr. Füsun Selçuk Tunalı	Prof.Dr. Haldun Abdullah
Jüri Başkanı	**Jüri Üyesi**	**Jüri Üyesi**

Doç..Dr. Ramazan Taşaltın	Y.Doç.Dr. A. Coşkun Sönmez
Jüri Üyesi	**Jüri Üyesi**

ÖNSÖZ

Bu tez çalışmamda maddi ve manevi yardımlarını esirgemeyen ve her konuda destek olan değerli hocam Prof. Dr. A.Ferit KONAR'a, aileme ve bana yardımcı olan tüm arkadaşlarıma teşekkür ederim.

Adapazarı, Haziran 1998 Yaşar BECERİKLİ

İÇİNDEKİLER

SİMGELER ve KISALTMALAR

A	İNSDP'deki sistem matrisi, CSTR reaktörü yüzey alanı
a	VDP sistem parametresi
B	İNSDP'deki sistem matrisi, boyutsuz CSTR parametresi
b	Geri-besleme kazanç vektörü, DSA nöron bias vektörü
C	İNSDP'deki adjoint sistem matrisi
C_A	CSTR içindeki A türü reaktant konsantrasyonu
C_{Af}	CSTR'a giren A türü reaktant konsantrasyonu
C_p	CSTR'ı terk eden akının ısı kapasitesi
Da	CSTR Damköhler sayısı
d	ÖKGY'deki yön vektörü, bozucu vektör fonksiyonu, CSTR'da boyutsuz besleme sıcaklık bozması
e	Hata fonksiyonu
F	DSA ağırlık fonksiyonu, CSTR besleme ve çıkan akış hızı
f	Genel amaçlı vektör fonksiyonu
g	Sigmoid fonksiyonu gradyanı, PÖ parametre gradyanı
H	Hamiltonian, yaklaşık Hessian tersi
$\tilde{H}$	Artırılmış Hamiltonian
h	CSTR'da soğutucu kısmın ısı transfer katsayısı, Sigmoid fonksiyonu
J	Performans ölçütü
J_a	Artırılmış performans ölçütü
K	Geri besleme kazanç matrisi
k_0	CSTR sabit reaksiyon hız faktörü
L	Ağırlık fonksiyonu
$\tilde{L}$	Yerel ağırlık fonksiyonu
N	Çapraz durum ağırlık matrisi
nd	Optimal kontrol çözüm yöntemleri için bir iterasyonda çözülecek toplam diferansiyel denklem sayısı

nk	Optimal kontrol problemi için çözüm iterasyon sayısı
P	Riccati dönüşüm matrisi
p	DSA giriş ağırlık matrisi
Q	Durum ağırlık matrisi
q	Sürücü vektör, DSA'da çıkış matrisi
R	Kontrol ağırlık matrisi,, CSTR reaksiyon hız terimi
S	Son durum ağırlık matrisi, ölçekleme matrisi
T	Dinamik zaman sabiti, CSTR reaktör sıcaklığı, Optimal kontrolde problem çözüm zamanı
T_C	CSTR'da soğutma ceket sıcaklığı
T_{c0}	CSTR soğutucu sıcaklığı nominal tasarım değeri
T_f	CSTR'da besleme sıcaklığı
T_{f0}	CSTR besleme sıcaklığı nominal tasarım değeri
T_s	CSTR reaktör sıcaklığı nominal tasarım değeri
t	Zaman
t_0	Başlama zamanı
t_f	Son zaman
u	Kontrol fonksiyonu, sistem giriş fonksiyonu
u^*	Optimal kontrol
v	İNSDP'deki sürücü vektör
W	Kontrol sınırlama ağırlık matrisi
w	İNSDP'deki sürücü vektör, DSA iç bağlantı matrisi
x	Durum vektörü
x_1	1. VDP durum vektörü, normalize CSTR konsantrasyonu
x_2	2. VDP durum vektörü, normalize CSTR sıcaklığı
x_{2s}	CSTR boyutsuz kararlı hal reaktör sıcaklığı
x^*	Optimal yörünge
x_0	Başlangıç durum vektörü
x_d	İstenen durum yörünge vektörü
x_f	Son durum vektörü
y	Sigmoid çıkışı, normalize CSTR sıcaklık hata çıkışı
z	DSA çıkışı

z^d	DSA için istenen çıkış yörüngesi
α	W'nın skaler güncelleme katsayısı
α_T	Göreceli problemi çözme zamanı kazancı (optimal kontrol probleminde)
β	Sigmoid bias parametresi, ÖKGY parametresi, CSTR parametresi
ρ	CSTR'daki sıvının yoğunluğu
γ	Algoritma çıkış test değeri, sigmoid lineersizlik parametresi, CSTR parametresi
γ_w	W için çıkma kriter değeri
λ	Adjoint değişkeni (Lagrange çarpanı)
$\tilde{\lambda}$	Yerel Lagrange çarpanı
$\delta\lambda$	Yerel Lagrange çarpanı
δJ	J'nin birinci varyasyonu
δJ_a	J_a'nın birinci varyasyonu
$\delta^2 J$	J'nin ikinci varyasyonu
ΔH	CSTR reaksiyon ısısı
ΔJ	J için fark
Δp	Parametre farkı
δx	Yerel durum vektörü
δu	Yerel kontrol vektörü
η	ÖKGY parametresi
τ	Adım büyüklüğü, CSTR'da boyutsuz zaman katsayısı
τ_1	Optimal kontrolde bir diferansiyel denklemi çözme zamanı
ϕ	Son durum fonksiyonu
$\tilde{\phi}$	Yerel son durum fonksiyonu

Kısaltmalar

Türkçe		İngilizce	
ADD	Adi diferansiyel denklem	ODE	Ordinary differential equation
AEL	Azaltılmış Euler-Lagrange	REL	Reduced Euler-Langrange

Türkçe		İngilizce	
APÖ	Artırılmış performans indeksi	API	Augmented performance index
BDGY	Birinci dereceden gradyan yöntemi	FOGM	First order gradient method
BFGS	Broyden-Fletcher-Goldfarb-Shanno	BFGS	Broyden-Fletcher-Goldfarb-Shanno
BGBÇ	Bir girişli/Bir çıkışlı	SISO	Single input/single output
Bkz.	Bakınız		
CSTR	Sürekli karıştırmalı tank reaktör	CSTR	Continuous stirred tank reactor
CSA	Cebirsel sinir ağları	ANN	Algebraic neural networks
ÇGÇÇ	Çok girişli/çok çıkışlı	MIMO	Multi input/multi output)
ÇKA	Çok katmanlı algılayıcı	MLP	Multi layer perceptron
ÇKİB	Çok katmanlı ileri-beslemeli	MLFF	Multi layer feed forward
ÇKSA	Çok katmanlı sinir ağı	MLNN	Multi layer neural network
DFP	Davidon-Fletcher-Powell	DFP	Davidon-Fletcher-Powell
DSA	Dinamik sinir ağları	DNN	Dynamic neural networks
DSOK	Dinamik sinirsel optimal kontrol	DNOC	Dynamic neuro-optimal control
EAGY	En hızlı azalan gradyan yöntemi	SDG	Steepest descent gradient
GB	Geri besleme	FB	Feed back
GY	Geriye yayınım	BP	Back-propagation
İB	İleri besleme	FF	Feed forward
İBOK	İleri beslemeli optimal kontrol	FFOC	Feed forward optimal control
İDDÖAY	İkinci dereceden direkt ölçütün azaltılması yöntemi	SODCD	Second order direct cost descent
İDHY	İkinci dereceden Hamiltonian yöntemi	SOG	Second order Hamiltonian
İNSDP	İki noktalı sınır değeri problemi	TPBVP	Two-point-boundary-value-problem
KGY	Konjuge gradyan yöntemi	CG	Conjugate gradient
LKG	Lineer kuadratik Gaussian	LQG	Linear quadratic Gaussian
LKOK	Lineer kuadratik optimal kontrol	LQOC	Linear quadratic optimal control

Türkçe		İngilizce	
LKOR	Lineer kuadratik olmayan regülatör	LNQR	Linear non quadratic regulator
LKR	Lineer kuadratik regülatör	LQR	Linear quadratic regulator
LM	Levenberg-Marquardt	LM	Levenberg-Marquardt
LOKOK	Lineer olmayan kuadratik optimal kontrol	NLQOC	Nonlinear quadratic optimal control
LOOK	Lineer olmayan optimal kontrol	NLOC	Nonlinear optimal control
LOP	Lineer olmayan programlama	NLP	Nonlinear programming
LP	Lineer programlama	LP	Linear programming
OGB	Optimal geri besleme	OFB	Optimal feed-back
ÖKGY	Ölçeklenmiş konjuge gradyan yöntemi	SCG	Scaled conjugate gradient
PID	Oransal-integral-türev	PID	Proportional-integral-derivative
PÖ	Performans ölçütü	PI	Performance index
PSM	Paralel sanal makine	PVM	Parallel virtual machine
RTF	Radyal temel fonksiyon	RBF	Radial basis function
SA	Sinir Ağları	NN	Neural networks
VDP	Van der Pol	VDP	Van der Pol
y.d.t	Yüksek dereceli terimler	h.o.t	High order terms
YPÖ	Yerel performans ölçütü	LPI	Local performance index
YSA	Yapay sinir ağları	ANN	Artificial neural networks
YZ	Yapay zeka	AI	Artificial intelligent
ZDOGB	Zamanla değişen optimal geri besleme	TVOFB	Time varying optimal feed-back
ZOK	Zeki optimal kontrol	IOC	Intelligent optimal control

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Anahtar Kelimeler: Zeki kontrol, nöro-kontrol, optimal kontrol, dinamik sinir ağları, ikinci dereceden lineer olmayan dinamik optimizasyon, adjoint teorisi.

Bu tez çalışmasında, zeki optimal kontrol sistem kavramı ve algoritmalarının geliştirilmesi ve optimal kontrol hesaplamalarının desteğindeki dinamik sinir ağları (DSA)'nın uygulaması sunulmuştur. Zeki kontrol, optimal kontrol teorisi gibi temeli sağlam bilimlerle ve yapay zeka (YZ) ve deneyimler gibi gelişmekte olan bilimlerle ilişkili geniş bir teknoloji alanını kapsar. Bu teknolojilerde, büyük bilgi alanını kullanarak yeni fikirler ortaya çıkar ve hesaplama yönünden etkili yazılım yapıları mümkün olur. Dinamik sistem kontrolüne yapay sinir ağları (YSA)'nın uygulaması, popüler ağ mimarilerinin dinamik olmayan tabiatları tarafından sınırlandırılmıştır. Güçlüklerin çoğu, büyük ağ ölçüsü, uzun eğitim zamanı gibi problemlerdir. Bu problemlerin üstesinden DSA ile gelinebilir.

Bu tez çalışmasında, zeki optimal kontrol problemi, dinamik eşitlik sınırlamalı bir lineer olmayan optimizasyon olarak ve DSA bir kontrol yörünge başlatma sistemi olarak düşünülmüştür. Direkt-azalan-eğrilik veya direkt-ikinci dereceden-azalan algoritma, optimal kontrol hesaplamaları için kullanılmıştır. Bu algoritma, literatürdeki Hamiltonian metotlarıyla karşılaştırılmıştır. Bu algoritma, diğerlerinden konjuge noktalara göre daha gürbüz sonuçlar üretmiştir. Aynı zamanda, yan ürünler olarak yörünge boyunca zamanla-değişen optimal geri-besleme kazançları da üretilmektedir.

Bu çalışmada, zeki optimal kontrol algoritması geliştirilmiştir. Sonuçlandırılan algoritma, DSA için bir otomatik eğitici olarak (kendi kendine öğrenme yapısı) işler ve az bir iterasyon sayısında optimal ileri-beslemeli (İB) kontrol yörüngeleri üretir. Bu yolda, optimal kontrol yörüngeleri DSA tarafından öğrenilir ve genelleştirilir. Hızlandırılmışmış yörünge hesaplamaları, kapalı genel geri-beslemeli gerçek-zamanlı zeki optimal kontrol için geniş yollar açar. Yarı-lineer dinamik bir sistem olarak düşünülen DSA'ın eğitiminde adjoint teorisi kullanılmıştır. Ağırlıkların güncellenmesi (parametre tanıma), tek boyutlu aramalı en hızlı azalan gradyan yöntemi (EAGY), ölçeklenmiş konjuge gradyan yöntemi (ÖKGY), Davidon-Fletcher-Powell (DFP) ve Broyden-Fletcher-Goldfarb-Shanno (BFGS) yöntemleri ile yapılabilir. Bu dört metot, başarılı bir şekilde DSA'na uygulanmıştır.

Benzetim sonuçları, lineer olmayan ikinci derece sistemler olan VDP ve CSTR'ın kontrol edilmesi için verilmiştir.

SUMMARY

NEURO-OPTIMAL CONTROL

Keywords: Intelligent control, neuro-control, optimal control, dynamic neural networks, second order nonlinear dynamic optimization, adjoint theory.

In this dissertation, the improvement of intelligent optimal control algorithms and the application of dynamic neural networks (DNN) in support of optimal control calculations are presented. Intelligent control covers a wide range of technologies related to hard-sciences such as the optimal control theory and soft-sciences such as artificial intelligence (AI) and heuristics. Using the broad knowledge spectrum, hard and soft, in these technologies, new concepts emerge and computationally efficient software structures become feasible. The application of artificial neural networks (ANN) to dynamic system control has been constrained by the non-dynamic nature of popular network architectures. Many of the difficulties are large network sizes, long training times, etc. These problems can be overcome with DNN.

In this dissertation, intelligent optimal control problem is considered as a nonlinear optimization with dynamic equality constraints, and DNN as a control trajectory priming system. Direct-descent-curvature or direct second order descent algorithm has been used for the optimal control computations. This algorithm is compared with the Hamiltonian methods in the literature. The algorithm has generated more robust solutions than the others with respect to conjugate points. The time varying optimal feedback gains are also generated along the trajectory as byproducts.

In this study, intelligent optimal control algorithm has been developed. The resulting algorithm operates as an auto-trainer for DNN (a self-learning structure) and generates optimal feed-forward control trajectories in a significantly smaller number of iterations. In this way, optimal control trajectories are encapsulated and generalized by DNN. Speeding up trajectory calculations opens up avenues for real-time intelligent optimal control with virtual global feedback. The adjoint theory has been used in the training of DNN which is considered as a quasi-linear dynamic system. The updating of weights (identification of parameters) are based on steepest descent gradient (SDG), scaled conjugate gradient (SCG), Davidon-Fletcher-Powell (DFP) and Broyden-Fletcher-Goldfarb-Shanno (BFGS) methods with line search. All methods has been successfully applied to DNN.

Simulation results are given for controlling VDP and CSTR which are nonlinear second order systems.

BÖLÜM 1. GİRİŞ

Bu tez çalışmasında, DSA ve optimal kontrol tekniklerinin bir arada kullanılarak, gerçek zamanlı uygulamalara yönelik hızlandırılmış yörünge üretimi yapısı geliştirilmiştir. DSA, optimal kontrolcü için kontrol başlatıcısı olarak kullanılmıştır ve böylece iyi bir yörüngeden başlayarak bir kaç adımda optimal kontrol ve yörüngeyi bulmak kolayca mümkün olmaktadır. Bir başka deyimle bu çalışmada, optimal kontrol algoritmalarının geliştirilmesi ve optimal kontrol eğitimindeki dinamik sinir ağları (DSA)'nın uygulaması sunulmuştur. Optimal kontrol problemi, dinamik eşitlik sınırlamalı bir lineer olmayan optimizasyon olarak düşünülmüştür. Bu çalışmada, optimal kontrol hesapları için, ikinci dereceden azaltan algoritma olarak direkt ölçüt fonksiyonunu azaltma yaklaşımı kullanılmıştır. Bu algoritma, literatürdeki Hamiltonian metotlarıyla karşılaştırılmıştır. Algoritmada, pozitif tanımlılık şartlarını sağlatan ve parametre olarak eklenen simetrik ağırlık matrisinin otomatik güncelleştirilmesini sağlayan yapı oluşturulmuştur. Bu algoritmanın, konjuge noktalara göre daha gürbüz (robust) sonuçlar ürettiği gösterilmiştir. Ayrıca, aynı zamanda da zamanla değişen optimal geri besleme kazançları, yörünge boyunca üretilmektedir.

Bu çalışmada, DSA, iteratif optimal kontrol hesaplamalarında, kontrol başlatıcı sistemi olarak kullanılmıştır. Sonuçta geliştirilen algoritma, daha az iterasyon ile optimal ileri besleme (İB)'li kontrol fonksiyonu üretmiştir. DSA, tam bağlantılı ağırlıklara sahip yapı modeli ile kullanılmıştır. Dinamik zaman sabiti ve sigmoid fonksiyonuna eklenen iki parametre ile oldukça yüksek kapasiteli hale gelmiştir. Gradyan hesaplamaları için adjoint metodu kullanılmıştır. Bu yöntem ile parametreden bağımsız duyarlılık hesaplamaları yapılmıştır. Bu çalışmanın katkılarından biri de, statik optimizasyon algoritmaları olan ÖKGY, DFP ve BFGS yöntemleri yarı lineer dinamik bir sistem olan DSA eğitimine uygulanmasıdır. DSA' na ait 7 temel parametrelerin tümünün güncellenebilmesi de önemli bir özelliktir. Ayrıca, DSA' nın optimal kontrolde kontrol

başlatıcı sistemi olarak kullanılması ve kendi kendine öğrenme yapısı bu çalışmanın önemli katkılarındandır.

En genel anlamda kontrol, sistemlerin fiziksel davranışlarının etkileriyle ilgilenen bir bilim dalıdır. Kontrol, günlük hayatın her sahasında karşımıza çıkar. Bilerek veya bilmeyerek sürekli olarak kontrol işlemlerini gerçekleştiririz. Mekaniksel, elektriksel, elektro-mekaniksel ve diğer bir çok araçlar insan denetimi olmadan otomatik olarak işleyebilmektedirler. Bu, kontrol (otomatik kontrol) sayesinde olmaktadır. Kontrol işlemlerinin çoğu, geri-beslemeli biçimde gerçeklenmektedir. Geri-beslemeli kontrol, sürekli hatayı gözlediğinden dolayı istenilen sonuca kolayca gidilebilmektedir. İnsan ise kendi kendini biyo-kontrol mekanizmasıyla otomatik olarak kontrol etmektedir.

Kontrol ve otomatik kontrol ifadeleri birbirini tamamlayan işlemlerdir. Kontrol: İncelenmiş davranışların istenilen değerde tutulması veya arzulanan değişimleri yapması için yapılanların hepsidir. Otomatik kontrol: Kontrol işlemini, kurulan bir mekanizma tarafından insan tesiri olmadan arzulanan değer civarında tutulmasıdır.

Son yıllarda, otomatik kontrol teorisinde bir çok gelişmeler olmuştur. Hala gelişmeler sürerken, çok genel bir analiz yöntemi oluşturmak zordur. Bununla beraber, geri beslemeli (GB) kontrol teorisinin gelişimine geriye doğru baktığımızda, bazı temel eğilimleri ve bazı temel gelişimleri ayırmak mümkündür. GB kontrolü, bir mühendislik disiplinidir. GB kontrolü geliştiren etkilerin tarihçesini şöylece özetleyebiliriz[1]:

i) M.Ö. 300 ile M.S. 1200 yılları arasında Arapların ve Yunanlıların zamanı doğru olarak tespit ederek takip edebilmeye yönelik çalışmaları.
ii) Avrupa'daki endüstriyel devrimin genelde 18. yüzyılda üçüncü çeyreğinde başlaması kararlaştırılmış olsa da, kökü 1600'lere dayanır.
iii) 1910 ile 1945 yılları arasında, toplu haberleşmenin başlaması ve Birinci ve İkinci Dünya Savaşları.
iv) Uzay/Bilgisayar çağının 1957'de başlaması.

Endüstriyel devrim ve dünya savaşları arasındaki zamanlarda oldukça önemli gelişmeler olmuştur. 1868'de, J.C. Maxwell bir geri besleme kontrol sisteminin ilk

matematiksel analizini yaptı. Savaş yıllarında yazılım dillerindeki gelişmeler matematiksel analizlerin uygulanmasını sağladı. 1868'den 1900'lü yılların başları arasındaki zaman, otomatik kontrolün ilkel zamanı olarak adlandırılır. 1960'a kadar olan kısma, klasik zaman ve 1960'dan günümüze kadar ki zamana ise modern zaman ismi verilir.

Endüstriyel devrime kadar, geri-beslemeli kontrol sistemlerinin tasarımı, büyük bir mühendislik sezgisiyle beraber deneme-yanılmaya dayalıydı. Bu ise, bilimden çok bir sanattı. 1800'ün ortalarında, matematik ilk kez, geri-beslemeli kontrol sistemlerinin kararlılığını analiz etmek için kullanıldı. 1840'da, Greenwich'te İngiliz kraliyet gök bilimci G.B.Airy, bir teleskopu kontrol etmek için geri-beslemeli bir araç geliştirdi. O araç, dünyanın dönüşünü kompanze etmek için otomatik olarak teleskopu ayarlayan bir hız kontrol sistemiydi. Ne yazık ki, Airy, sisteme kötü osilasyonların girdiğini ve geri-beslemeli kontrol çevriminin uygun olmayan bir yönünü keşfetti. O, kapalı çevrim sistemlerinin kararlılığını tartışan ve analizinde diferansiyel denklemleri kullanan ilk kişiydi. Sonra diferansiyel denklem teorisi iyi bir şekilde geliştirildi. Bu, I. Newton (1642-1727), G.W. Leibniz (1646-1716), J.F. Riccati (1676-1754), Bernoulli kardeşlerin ve diğerleri tarafından sonsuz küçük hesabının keşfi sayesinde oldu. Dinamik sistemlerin hareketinin analiz edilmesinde diferansiyel denklemlerin kullanımı, J.L. Lagrange (1736-1813) ve W.R. Hamilton (1805 1865) tarafından kuruldu.

Kararlılık teorileri de, 1800 yılının ortalarından itibaren gelişmiştir. Bu konuda, ilk olarak J.C Maxwell'in 1868'deki çalışması olmuştur. Bu çalışmada, lineerleştirilmiş diferansiyel denklemi kullanarak sistemin karakteristik denklemini inceledi. Karakteristik denklemin kararlılığa etkisini inceledi. 1877'de E.J. Routh, nümerik bir teknik kullanarak kararlılığı analiz etti. 1877'de Rus I.I. Vyshnegradsky, Maxwell'den bağımsız olarak diferansiyel denklemleri kullanarak regülatörlerin kararlılığını analiz etti. A.M. Lyapunov'un çalışması, kontrol teorisinde büyük bir kaynaktır. 1892'de o genelleştirilmiş enerji fikrini kullanarak lineer olmayan diferansiyel denklemlerin kararlılığını incelemiştir.

Toplu haberleşme sistemlerindeki gelişmeler 20. Yüzyılın başlarından itibaren tüm

alanlarda etkisini göstermiştir. Zaman düzleminde diferansiyel denklemler kullanılarak, kontrol sistemlerinin matematiksel analizleri çıkartıldı. P.S. de Laplace (1749-1827), J. Fourier (7168-1830), A.L. Cauchy (1789-1857) ve diğerleri tarafından geliştirilen frekans düzlemi yaklaşımları, 1920 ve 1930 yılları arasında Bell Telefon Laboratuarlarında keşfedilip haberleşme sistemlerinde kullanıldı. Uzaklara sesi göndermek için kullanılan toplu haberleşme sistemlerindeki ana problem, uzun telefon hattında ses işaretini periyodik olarak kuvvetlendirme gereksinimi idi. Ancak, yapılan uygulamalarda ses ile beraber gürültü de kuvvetlendirilerek gönderiliyordu. Böylece, uygun tekrarlayıcı kuvvetlendirici tasarımı başlıca öneme sahipti. 1927'de, H.S. Black tekrarlayıcı kuvvetlendiricideki bozulmayı azaltmak için negatif geri beslemenin kullanımını gösterdi. 1932'de H.Nyquist, kararlı kuvvetlendirici tasarımı için tekrar elde etme teorisini geliştirdi. 1938'de H.W.Bode, kompleks bir fonksiyonun genlik ve faz frekans cevabı çizimlerini kullandı. Genlik ve faz payı fikrini kullanarak kapalı çevrim kararlılığını araştırdı.

Dünya savaşları sırasında geri beslemeli kontrol teknikleri gelişerek savaş sistemlerinde kullanıldı. O sıralarda önemli askeri problem, gemilerin kontrolü ve seyriydi. 1910'da E.A. Sperry jiraskopu keşfetti ve gemilerin kullanımı ve kararlılığında ve sonraları uçak kontrolünde kullandı. 1922'de N. Minorsky, gemilerin kullanımı için üç-parçalı kontrolcüyü (PID) ilk olarak kullandı. Kapalı çevrimdeki lineer olmayan etkileri düşünmüştü. Dünya savaşları sırasında karşılaşılan önemli problemlerden biri de, gemi ve uçaklardan atılan silah ve bombaların doğru olarak hedeflere atılmasıdır. 1934'de H.L. Házen servo mekanizmaları teorisi konusunda yaptığı araştırma bu şekildeki problemlerde mekanik kontrol teorisinin kullanılması ilkti.

Yeni olarak bulunan radarla birleştirilmiş kontrol ve bilgi işleme problemlerini çalışmak için, 1940'da MIT'de radyasyon laboratuarları kuruldu. 1941'de MIT/Sperry Kurumu'nda katıldığı proje üzerinde çalışıyorken, A.C. Hall, kontrol sistemi tasarımında gürültüyü göz önüne almamanın kötü etkilerini anladı. Transfer fonksiyonu blok diyagramı ve frekans düzlemi metotlarına dayalı tasarım yaklaşımları kullanarak, Radyasyon Laboratuarlarında kontrol tasarımlarında büyük başarılar

sağladı. 1947'de N.B. Nichols, kendisinin Nichols Şeması'nı geri beslemeli sistemlerin tasarımı için geliştirdi. MIT'deki çalışmalarla lineer servo mekanizma teorisi kuruldu. 1948'de Kuzey Amerika Havacılık'ta çalışıyorken W.R. Evans, kök-yerleri tekniğini sundu. Bu teknik, s-düzleminde kapalı çevrim kutup yerleştirmeye imkan vermekteydi. Bu sıralarda aynı zamanda, olasılık teknikleri kontrol ve haberleşmeye girdi. 1942'de MIT'de, N.Wiener, olasılıksal proseslerin modellerini kullanarak bilgi işleme sistemlerini analiz etti. Frekans düzleminde çalışarak, durağan sürekli zamanlı işaretler için istatistiksel olarak bir optimal süzgeç geliştirdi. 1941'de Rus A.N. Kolmogrov, ayrık zamanlı durağan olasılık prosesleri için bir teori geliştirdi.

II. Dünya savaşından sonraki zaman, kontrol teorisinin klasik zamanı olarak isimlendirilebilir. Bu zamanda değişik çözüm takımları geliştirilerek kontrol problemleri çözülmeye çalışılmıştır.

Uzay çağının gelmesiyle ABD'de kontrol tasarımında klâsik kontrol teorisinin frekans düzlemi tekniklerinden uzaklaşıldı ve 1800'lü yılların sonlarının diferansiyel denklem tekniklerine geri dönüldü ki bu teknikler zaman düzlemindeydi. Bunun sebepleri ise aşağıda izah edilmektedir.

Klasik kontrol teorisi, dünya savaşları sırasında ve hemen sonrasında, kontrol tasarım problemleri için çok uygundu. Frekans düzlemindeki yaklaşım lineer zamanla değişmeyen sistemler için uygundu. Bunlar bir girişli/bir çıkışlı (BGBÇ) sistemler için en iyisidir. Ancak çok girişli/çok çıkışlı (ÇGÇÇ) sistemlere uygulamak ise uygun değildir. Klasik kontrol tasarımı lineer olmayan sistemlerde bazı başarılı sonuçlara sahip olmuştur. Frekans düzlemi tekniklerinde gürültü bastırma özelliklerini kullanarak, bir kontrol sistemi sistem parametrelerindeki değişimlere karşı ve ölçme hatalarıyla dış bozuculara karşı gürbüz (robust) olarak tasarımlanabilir. Böylece klasik teknikler, kararsız bir noktada lineerleştirilerek iyi sonuçlar veren lineer olmayan bir sistemin lineerleştirilmiş versiyonlarında kullanılabilir. Frekans düzlemi teknikleri, Nyquist kriterini sağlayan "tanımlayan fonksiyon yaklaşımı" kullanarak basit tipli lineersizlikle sistemlere uygulanabilir.

Ancak, tüm bu klasik teknikler gelişmiş lineer olmayan çok değişkenli sistemler için

kontrol sistemi tasarlamaya uygun değildir. Özellikle uzay sistemleri uygulamaları çok hassasiyet isteyen kontrol yapıları gerektirmektedir.

Rusya'da, lineer olmayan kontrol tasarımlarında büyük çalışmalar olmuştu. Lyapunov'dan sonra çalışmalar zaman düzlemi tekniklerinde yoğunlaşmıştır. 1948'de Ivachenko, ayrık değerler arasında süreksiz olarak anahtarlanan kontrol işaretine dayalı "röle kontrolü" prensibini araştırdı. 1955'de Tsypkin, lineer olmayan kararlılık analizi için kendisinin "dairesel kriterini" geliştirdi. 1957'de Rusya'da ilk uydu olan Sputnik fırlatıldı. Sputnik'in fırlatılması, otomatik kontrol tasarımı konusunda Amerika'da büyük gelişmelere neden oldu. Herhangi bir teoride yaşanan bir başarısızlık sonucunda, tarihsel ve tabi birincil prensiplere geri dönülmesini gerektirdi. Böylece, diferansiyel denklemlere dayalı kontrol teorisinin "ilkel" zamanının zaman düzlemi tekniklerine dönülmesini gerektirdi. Lagrange ve Hamilton'un çalışmaları, bir çok dinamik sistemler için lineer olmayan hareket denklemlerini yazmak için doğrudan kullanılır.

Hemen hemen 1960'da, haberleşme ve kontrol teorisindeki başlıca önemli gelişmeler muhtelif yönlerde bağımsız olarak meydana gelmiştir. 1960'da C.S. Draper, gemi, uçak veya uzay aracı gibi araçların hareketlerinin pozisyonu hakkında doğru olarak bilgi sağlayan jiraskopu kullanarak kendisinin seyir sistemini keşfetti.

1696'da ilk olarak J. Bernoulli, Brachistochrone problemiyle bağlantılı optimallik prensibinden söz etti. Bu problem Bernoulli kardeşler ve I.Newton tarafında çözüldü. 1600'lerde P.De. Fermat'ın optikteki minimum zaman prensibi, 1744'de L.Euler'in çalışması ve Hamilton'un kinetik ve potansiyel enerji farkı üzerine yaptığı sonuçları içeren değişik optimallik prensipleri araştırıldı. Bu optimallik prensiplerinin hepsi minimum prensiplerdir. 1900'lu yılların başlarında Einstein'in görecelik teorisi, 4-boyutlu uzay zaman koordinat sistemi için ilginç sonuçlar vermiştir.

Bir sistemin davranışını optimallik kriteri altında incelemek, insan yapısı kontrol sistemlerini tasarlamak için bir optimal gösterim sunacaktır. Bunun en önemli avantajı bu tasarımlar zaman düzleminde yapılabilir olmasıdır. Modern kontrol tasarımının

içeriğinde, geçiş zamanını veya bazı kontrol sınırlamaları altında genelleştirilmiş bir enerji fonksiyoneli veya performans ölçütünü minimum yapmaktadır. 1957'de R. Bellman, optimal kontrol problemlerini zamanda geriye doğru çözmek için tabi yönü göstererek ayrık zamanlı sistemlerin optimal kontrolüne dinamik programlarını uyguladı. 1958'de L.S. Pontryagin, L.Euler (1707-1783) tarafından geliştirilmiş değişimlerin hesabına dayanarak optimal kontrol problemlerini çözen kendisinin "maksimum prensibini" geliştirmişti. O, optimal kontrol olarak bir açma kapama röle kontrol kanunu çıkararak minimum zaman problemini çözdü. 1950'li yıllarda Amerika'da değişimlerin hesabı, Chicago ve diğer üniversitelerde genel optimal kontrol problemlerine uygulandı.

1960'da R. Kalman yaptığı çalışmaları yayınladı. Bunlardan biri, lineer olmayan sistemlerin zaman düzlemi kontrolünde Lyapunov'un önemli çalışmasını yayınlamıştır [2]. İkinci olarak lineer kuadratik regülatör (LKR) problemi için tasarım denklemlerini sağlayarak sistemin optimal kontrolünü tartıştı [3]. Daha sonra, ayrık Kalman süzgeci için tasarım denklemlerini sağlayarak optimal süzgeçleme ve kestirim teorisini inceledi [4]. Sürekli Kalman süzgeci, Kalman ve Bucy tarafından daha sonra geliştirildi [5]. Modern kontrol teorisinin geliştirilmesiyle klasik kontrol teorisindeki başlıca sınırlamalar ortadan kalkmıştır. Kalman'ın yapmış olduğu çalışmaların çok önemli katkıları olmuştur. Lineer olmayan sistemler gibi zamanla değişen lineer sistemler için de oldukça uygulanabilir olan zaman düzlemi yaklaşımı önemli bir sonuçtur. Kalman, ÇGÇÇ sistemlere uygulanabilen lineer cebir ve matrislere girdi. Kontrol teorisinde Kalman, çok genel bir kuadratik genelleştirilmiş enerji fonksiyonunu minimum yaparak kontrol teorisindeki optimallik kavramını formülize etti. Kestirim teorisinde durağan olmayan zamanla değişen sistemlere uygulanan olasılık kavramlarına girdi. Böylece, C.F.Gauss (1777-1855) tarafından ilk kez gezegenlere ait yörünge kestiriminde kullandığı en küçük kareler yaklaşımı için Kalman süzgeci bir ardışıl çözüm sağlamaktadır. Kalman süzgeci, Wiener süzgecinin durağan olmayan olasılık sistemlerine tabi bir genelleştirilmiş halidir. Kalman'ın teorisi kontrol sisteminde performansı garantileyen optimal bir çözüm sağlar. Lineer olmayan kontrol teorisi de 1960'lardan sonra hızla gelişti [1,6,7]. Bu gelişmeler sonucunda lineer olmayan proses kontrol [8,9], uçak kontrol tasarımı [10-15] için oldukça önemli yer tutar. Lineer

olmayan kontrol teorisi konusunda çok önemli çalışmalar yapılmıştır [6,8,16-20].

Kontrol konusunda daha detaylı bilgiler, son zamanda yayınlanan çalışmalarda bulmak mümkündür. Kontrol, otomatik kontrol, optimal kontrol, adaptif kontrol, geri besleme, kararlılık ve olasılıksal kontrol konularında tarihsel gelişimleriyle beraber daha detaylı bilgiler bulunabilir [21-28]. Ayrıca, optimal kontrol hakkında daha eskiye yönelik bilgilerde edinilebilir [29].

Kontrol, geri besleme ve dinamik sistemlerin birbirleriyle bağlantıları ve kararlılığı çok önemli konulardandır. Lineer zamanla değişmeyen sistemlerde, kontrol edilebilirlik, kutup yerleştirme ve geri besleme konusunda elde edilen sonuçlar lineer olmayan sistemlerin çalışma noktasındaki kararlılığına uygulanmıştır [30]. Ayrıca, lineer olmayan sistemler için kararlı geri besleme kanunun çıkarılması ve performans iyileştirme konusunda [31]'de iteratif bir algoritma verilmiştir.

Optimal kontrol teorisi uygulamada önemli yer tutmaktadır. Bu konuda bir çok çalışma yapılmıştır [1,32,33]. [34]'de ayrık-zamanlı lineer kuadratik Gaussian (LKG) problemi incelenmiş ve Lagrange çarpanı metoduna dayalı yeni bir çözüm geliştirilmiştir.

[35]'de lineer kuadratik regülatör (LKR) probleminin gürbüzlük özelliği incelenmiştir. Genlik ve faz paylarının gürbüz kontrol üzerindeki etkileri araştırılmıştır.

Optimal kontrol, bir çok yapılara sahip problem olarak karşımıza çıkabilir. LKR problemi gibi, lineer olmayan optimal kontrol yapısı daha geneldir. Ayrıca [36]'da lineer kuadratik olmayan regülatör (LKOR) problemi sonlu-ufuk problemini çözmek için dinamik programlamanın bir gerçeklemesi olarak gösterildi.

Birinci dereceden gerekli koşullardan yola çıkılarak, genel son nokta sınırlamaları altında düzgün ve lineer olmayan optimal kontrol problemi ana bir problemdir. Bu problem, değişimlerin hesaplanması konusundaki Euler-Lagrange denkleminin genelleştirilmesi yoluyla çözülmüştür [37].

Lineer veya lineer olmayan optimal kontrol problemlerinin çözümünde Riccati diferansiyel denklemleri veya Riccati fark denklemleriyle karşılaşılır. Çok önemli bir konu olan Riccati denklemleriyle ilgili çok sayıda araştırmalar yapılmıştır [1,4,32,33]. Son zamanlarda da Riccati çözümlemeler ile ilgili yeni yöntemler geliştirilmiştir [38-45].

Optimal kontrol teorisi konusunda literatürde yazılmış ve temel konuları ve uygulamaları içeren önemli kitaplarda mevcuttur [1,6-8,46-49].

[46]'da genel optimizasyon yapılarından başlanıp, değişimlerin hesabı konularına değinilmiştir. Lineer ve lineer olmayan optimal kontrol problemi incelenmiş, birinci ve ikinci dereceden gradyan metotlar Hamiltonian prensibine dayalı olarak anlatılmıştır. Kararlılık ve optimum geri besleme yapısı ele alınmıştır. Ayrıca, uygulamalı örnekler verilerek konu daha anlaşılır şekilde sunulmuştur.

[47]'de temel olarak konjuge gradyan dinamik optimizasyon yöntemi ele alınmıştır. Ayrıca ölçeklenmiş konjuge gradyan konusu incelenmiştir. Sonuçlar, lineer olmayan optimal kontrole uygulanmıştır.

[48], çok temel konulardan başlamıştır. Statik ve dinamik optimizasyon konularını incelemiştir. Lineer programlama (LP) ve lineer olmayan programlama (LOP) konularından optimal kontrole uzanan büyük bir pencereden bakarak bir çok konuyu bir arada vermiştir.

[49]'da temel optimal kontrol konusu kısaca incelenip elektrik güç sistemlerine uygulanması verilmiştir.

Lineer olmayan optimal kontrol (LOOK) teorisinde çözümleme yöntemleri genelde gradyan tabanlı yöntemlerdir. Gerekli koşullar, adjoint teorisi yardımıyla Lagrange çarpanı yöntemi kullanılarak kolayca çıkarılabilmektedir. Daha sonra, birinci derece gradyan, konjuge gradyan ve diğer yöntemler kullanılmaktadır. Burada, Hamiltonian prensibine dayalı yöntemler geliştirilmiştir. Önemli bir yaklaşımda ikinci dereceden

gradyan yöntemlerdir. Literatürde geliştirilen yöntemler Hamiltonian prensibine dayalı yöntemlerdir [6,8,16-19,46,48,50]. Bu yöntemler, bazı şartların yerine getirilmesini gerektirir. Özellikle, konjuge nokta problemi çok önemli bir problemdir. Eğer, çözümde bu noktaya takılırsa, çözümü bulmak mümkün değildir. [16,17]'de bu problem için bir çözüm geliştirilmiştir. Aksi durumda, başka bir noktadan başlamak ve çözüme ulaşmak gereklidir.

Bu tez çalışmasında incelenen ve geliştirilen başka bir yöntem lineer olmayan kuadratik (LOK) problemler için konjuge nokta problemini çözebilmektir. Bu, direkt ölçüt fonksiyonun ikinci derece açılımına dayanır. Ayrıca, lineer olmayan problemler için de eklenen yeteri kadar büyük kontrol sınırlama matrisi ile konjuge nokta problemi çözülmüştür. Bu matrisin, güncellenmesinde oluşturulan bir oran katsayısı kullanılmıştır. Böylece her adımda otomatik güncelleme sağlanmıştır. Sonuçta herhangi bir ek işlem yapmadan, optimal geri besleme kanunu da yörünge boyunca üretilmiştir.

Bütün yapılan bu işlemler, optimal kontrol probleminin daha hızlı çözümlenmesini de amaçlamaktadır. Bir kaç adımda, yüzlerce adımda ulaşılamayacak optimallikte kontrol fonksiyonu bulunabilmektedir. Buda, hızlı uygulama gerektiren sistemler için çok iyi bir yoldur.

1948 yılında J.Von Neumann modern bilgisayarın temellerini atmasından sonra kontrol ve haberleşme alanında çok hızlı gelişmeler olmuştur. 1969'da yarı iletken teknolojisinin bilgisayar teknolojisine mikro işlemci olarak girmesiyle yeni bir dönem başlamıştır. Artık gerçeklenmesi zor olan lineer olmayan kontrol ve özellikle optimal kontrol problemleri kolayca çözülmeye başlanmıştır. Daha sonraları, sayısal kontrol yöntemleri de hızla gelişip uygulamaya girmiştir. Özellikle KONPACT, ADAPS, DIGIKON ve XIMKON programları bu konularda oldukça gelişmiş özelliklere sahiptir [15]. 1983'de kişisel bilgisayarların hayata girmesiyle isteyen her mühendis modern kontrol teorilerini uygulama imkanı bulmuştur. Özellikle, kontrol sistem tasarımına yönelik ticari programların geliştirilmesi daha da kolaylıklar sağlamıştır. Örneğin, ORACLS, MATLAB, MATCAD, Control-C, $MATRIX_x$, SIMNON, Easy-

5, Mathematica gibi ticari programlar.

Kontrol ve kestirim teorisine son 15 yıldır yeni teknikler girmiştir. Bu yeni tekniklerden en önemlisi yapay sinir ağları (YSA) teknolojisidir. Daha önceleri temelleri atılmış olmasına rağmen YSA 1985'den sonra gerçek anlamda uygulama alanlarına girmiştir. Bu konu, insana benzeme veya insana dayalı bir sistem benzetimiyle ortaya çıkmıştır. YSA, beynin hesapsal bir modelidir. Birçok çeşit YSA modeli geliştirilmiştir. Bunların hepsinde bağlantı elemanları olan nöronlar ve bunların ara bağlantı parametreleri olan ağırlıklar mevcuttur. Bir çok amaçlar için kullanılmaktadır. Sınıflama, ayırma, tanıma, kontrol, kestirim, görüntü işleme, ses işleme vb. bir çok konularda uygulama alanı bulmuştur.

Yapısal olarak sinir ağları iki tipe ayrılır. Bunlardan birincisi ileri beslemeli (İB) sinir ağlarıdır. Burada nöronlar gizli katmanlardan oluşur. Bir katmandan bir sonraki katmana ağırlıklarla bağlantılar yapılır. İB sinir ağları da bir çok türe sahiptir. Çok katmanlı algılayıcı (ÇKA) ağı, öğrenme vektörü kuantalama ağı gibi. İkincisi, geri beslemeli (GB) veya dinamik sinir ağları (DSA)'dır . En popüler olanı Hopfield ağıdır. Burada GB ağı bir dinamik hafizaya sahiptir.

YSA modelleri konusundaki çalışmalar uzun bir tarihçeye sahiptir. Detaylı matematik modellerin geliştirilmesi, 50 yıldan önceye, 1943'de Mc Culloch ve Pitts'in çalışmalarına dayanır. Onlar, bir nöron modeline dayalı bir kaç basit elemanların bağlantı kapasiteleri ve potansiyelini araştırdı. 1949'da Hebb, sinir sistemlerinde gerekli olan uyumluluk kanunlarıyla ilgilendi. 1958'de Reosenblatt, algılayıcı (Perceptron) ismini koydu ve sonraları daha ilgi çeken bir yapı tasarladı. Minsky ve Papert 1969'da algılayıcının ciddi bir analizine girdi. Onlar bir çok özellikleri ispatladılar ve bir kaç modelin sınırlarını gösterdiler. 1976'daki Grossberg'in çalışması büyük bir önem kazandı. Bu çalışma, biyolojiksel ve psikolojiksel olaylara dayalıydı ki alışılmamış karakteristiklere sahip lineer olmayan dinamik sistemlerin bir kaç mimarisini ileri sürdü. 1982'de Hopfield, optimizasyon gibi teknik problemleri çözmek için belirli bir lineer olmayan dinamik yapı kullandı. 1986'da paralel dağıtılmış işleme grubundan Rumelhart ve arkadaşları, sonuçlar ve algoritmaların bir serisini bastılar.

Bu çalışma alanda çalışanlara güçlü bir ivme ve katalizör görevi yaptı.

1987'de Lippmann [51] sinir ağlarıyla hesaplamaya yönelik yaptığı çalışma, genel anlamda YSA'nı iyi bir şekilde anlatıyordu. Bu çalışmada, YSA'nın genel özellikleri ve yapıları üzerinde durulmaktadır. Özel olarak sınıflayıcı özelliği üzerinde durulmuştur. Büyük yoğunluklu entegre çiplerinin (VLSI) gelişmesiyle ses ve görüntü tanıma üzerinde yapılabilecek uygulamalar anlatılmaktadır. Ayrıca YSA'nın eğitiminden geriye yayınım (GY) ve momentumlu öğrenme üzerinde durulmaktadır.

Lippmann'ın bu geniş ölçekli makalesinden sonra 1993'de yeni gelişmeleri de içeren bir makale yayınlanmıştır [52]. Bu yayında YSA modellerinin sınırları ve kapasiteleri, ayrıca sinir ağı modelleri detaylı olarak incelenmiştir. Ağ modeli olarak iki yapı incelenmiştir. Birincisi, statik ağlar ki çok çok kullanılan bir yapı olan ÇKA kullanılmıştır. Yani ağ çıkışı, sadece o anki girişe bağlıdır. Diğer bir model olarak da Radyal Temel Fonksiyon (RTF) ağ çeşidi incelenmiştir. İkincisinde dinamik ağlar ele alınmıştır. Düğüm denklemleri, diferansiyel veya fark denklemleriyle ifade edilmektedir. Burada durum geri beslemeli yapı üzerinde özellikle durulmuştur. Eğitim için ise, öğreticili eğitim yapısı incelenmiştir. Yöntem olarak ise standart GY algoritması detaylı bir biçimde ele alınmıştır.

[53]'de dinamik sistemlerin optimizasyonu için gradyan metotları üzerinde durulmuştur. Buradan yola çıkılarak, YSA'nın eğitimi için gerekli gradyan yöntemi olan GY algoritması anlatılmıştır. Burada, ÇKA yapısı özellikle ele alınmıştır. Ayrıca dinamik GY algoritması ÇKA için parametrelerinin optimizasyonunda uygulanmıştır. Kullanılan yapı birinci dereceden gradyan yöntemiyle sınırlıdır. Dinamik yapının bir kontrol problemine de uygulanması incelenmiştir.

[54]'de çok katmanlı ileri beslemeli (ÇKİB) ağlar üzerinde, yığın öğrenme ve patern öğrenme hakkında çalışma yapılmıştır. Lineer olmayan sistem tanıma problemi ÇKİB ve dış geri beslemeli ağ yapısı ile çözülmesi üzerinde çalışılmıştır.

[55]'de ÇKA ile fonksiyon yaklaştırma üzerinde durulmuştur. YSA'nın lineer olmayan

özelliğini veren aktivasyon fonksiyonu üzerinde geniş ölçüde durulmuştur. Sigmoid fonksiyonu kullanılmıştır ki çok genel yapı sağlayacak biçimde lineersizlik parametresi ve eşik biaslama parametresi eklenerek bunun etkileri incelenmiştir.

[56]'da gradyan tabanlı algoritmalar yardımıyla DSA'nın eğitilmesine genel yaklaşım anlatılmıştır. Gürültü olması durumunda modellemenin nasıl yapılacağı üzerinde durulmuştur. Dinamik bir sistemin modellenmesi üzerine örnekler verilmiştir.

[57]'de DSA da eğitim yöntemi için değişimlerin hesabı yapısından yola çıkılarak adjoint teorisi yardımıyla sonuca ulaşılmıştır. Bu yöntemde, zamanda geriye doğru integrasyon yapılması gereklidir. Bessel diferansiyel denkleminin eğitimine uygulanmıştır.

[58]'de standart GY algoritmasında kullanılan öğrenme katsayısı üzerinde bazı çalışmalar yapılmıştır. Genelde sabit alınan bu katsayının her adımda değiştirilmesi eğitim hızını artıracağı için bu konu üzerinde durulmuştur. Momentumlu GY algoritması ve konjuge gradyan algoritması ele alınarak bu iki yöntemin benzerliği gösterilmiştir. Standart GY algoritmasına göre bu yöntemlerin daha hızlı yakınsadığı gösterilmiştir.

DSA üzerinde yapılan çalışmalardan biri [59]'da verildiği gibi dinamik yapıya sahip olan ağ yapısında, parametrelerin değişim etkisi, bağlantı sınırlamalarının ve gecikmenin oluşturacağı etkiler incelenmiştir. Hopfield ağa benzer sinir ağı üzerinde çalışılmıştır. Sigmoid lineersizliğine sahip ağ için gürbüzlük gerek ve yeter koşulları çıkarılmıştır. Zaman gecikmesini sinir ağı üzerindeki etkisi genel ve yerel sonuçlar açısından düşünülmüştür. Zaman gecikmesi için gerekli koşullar genel kararlılık için oluşturulmuştur.

Optimal kontrolde kullanılan değişimlerin hesabı yaklaşımıyla ÇKA probleminin çözümü [60]'da geliştirilmiştir. Burada, önce sürekli zaman formu ele alınıp, Lagrange çarpanı yaklaşımıyla problem çözülüp ayrık yapıya uygulanmıştır. Parametre güncellenmesinde en hızlı azalan gradyan algoritması kullanılmıştır. Yani bir boyutlu

minimizasyon probleminin çözümü yardımıyla her adımda optimum öğrenme katsayısı bulunmuştur.

[61]'de ileri beslemeli sinir ağı yapıları için hızlı öğrenme algoritmaları üzerinde çalışılmıştır. Yaklaşım olarak, her katmanın lineer ve lineer olmayan iki bloğa ayrılabileceğinden yola çıkılmıştır. İkinci dereceden eğitim yöntemi olarak Newton metodunun depolama ve ters alma yükü olduğu düşünülerek bu yaklaşım ile ortalama bir yol tutulmuştur. Yaklaşık Hessian matrisi kullanılarak geliştirilen Levenberg-Marquardt yaklaşımından ve Broyden-Fletcher-Goldfarb-Shanno (BFGS) metodundan bahis edilerek geliştirilen yöntemin daha az hesaplama maliyeti ve kararlılıkla çalıştığı vurgulanmıştır.

[62]'de yaklaşık ikinci dereceden optimizasyon algoritmalarından olan BFGS'in iki tipi ÇKA üzerinde paralel gerçekleştirme ile uygulanmıştır. Sınırlandırılmış ve tam hafizalı BFGS algoritmaları eğitim için kullanılmıştır. Gerçeklemede paralel sanal makine (PSM) ve transputer mimarileri kullanılmıştır. Sonuçda, BFGS algoritmalarının 20 ila 100 kat arasında standart GY algoritmasından daha hızlı olduğu görülmüştür.

Optimizasyon ve YSA birlikte anılmakta olup bu konuda bir çok kitap mevcuttur. YSA'nın eğitilmesi bir optimizasyon veya parametre tanıma problemidir. [63]'de optimizasyon ve işaret işlemeye yönelik YSA'nın kullanımı hakkında detaylı bilgiler verilmektedir. Hem genel optimizasyon problemleri incelenmiş ve hem de bu yöntemlerin YSA'na nasıl uygulanabileceği gösterilmiştir. Bir çok yöntemlerden bahis edilmekte olup lineer olmayan adi diferansiyel denklemler (ADD), fark denklemleri, lineer cebir ve lineer olmayan programlama, YSA'nın analizi ve sentezi için incelenmiştir.

Bu tez çalışmasında DSA kontrol problemine, özel olarak optimal kontrole uygulanması ele alınmıştır. Bu yüzden YSA'nın bir çok yapıda değişik kontrol problemlerine uygulandığı görülmüştür. İç model kontrolcü, modele dayalı kontrol, amaca dayalı kontrol, kontrol amacına dayalı kontrol, benzetim kontrolü, kontrole

dayalı eğitim vb. gibi bir çok kontrol yapısı kullanılmaktadır [64].

[65]'de lineer olmayan dinamik sistemlerin kontrol ve tanımada YSA'nın kullanımı ele alınmıştır. Bu parametrelerin ayarlanmasında statik ve dinamik GY yöntemi tartışılmıştır. Çok katmanlı ve dinamik ağların bağlantıları üzerinde birleşik bir gösterim sunulmuştur. Benzetim sonuçlarında, tanıma ve adaptif kontrol şekillerinin uygulanması verilmiştir.

[66]'da adaptif kontrol çalışmalarından biri uygulanmıştır. Bu çalışmada GY yapısına dayalı ağ, kendinden ayarlı adaptif kontrol olarak lineer olmayan sisteme uygulanmıştır. Aktivasyon fonksiyonu olarak tanh (.) fonksiyonu seçilmiştir. Öğrenme katsayısı başlangıçta sabit bir sayı olarak alınarak, iyi yakınsama durumuna göre artırılmıştır.

[67]'de yine kendinden öğrenmeli lineer olmayan kontrol sistemleri için YSA'nın uygulaması anlatılmaktadır. Lineer olmayan dinamik sistemlerin modellenmesi için çok katmanlı sinir ağı (ÇKSA) emülatör olarak kullanılmıştır. Bunun üzerinden hata geriye doğru yayılarak, diğer bir ÇKSA kontrolcü olarak çalışmaktadır. Sonuçta bir çok lineer olmayan kontrol problemine uygulanabilir bir çözüm geliştirilmiştir.

Sinir ağları, lineer olmayan sistemlerin modellenmesi ve kontrolü için potansiyel olarak genel bir çerçeve sağlar. [68]'de bu potansiyelleri açığa vuran geniş bir bakış açısı getiren araştırma sunulmaktadır. Lineer olmayan sistemlerin modellenmesi, tanıma ve kontrolünün gerçeklenmesinde YSA'nın oynadığı büyük rol gözler önüne serilmektedir. Geniş bir YSA mimari ve uygulamaları kontrol açısından incelenmiştir. Statik ve dinamik sinir ağı konusunda detaylı teorik açıklamalar verilmiştir. Ayrıca DSA için, nöronun dinamikliğini veren 5 çeşit dinamik yapı verilmiştir. Ve eğitimde adjoint teorisinin uygulanması anlatılmıştır. Bu, öğrenmedeki hız problemi açısından önemlidir.

Kararlılık ve kontrol edilebilirlik konuları kontrol teorisinde geniş bir yer tutar. Lineer olmayan kontrol teorisindeki sonuçlara dayalı olarak sinir ağı kullanılarak pratik

kontrolcülerin tasarımlanabileceği görülmüştür [69]. Bu çalışmada dinamik sistemin denge noktası civarındaki kararlılığı problemi ele alınmıştır.

YSA' da kullanılan ve lineersizliği belirleyen aktivasyon fonksiyonları değişik tiplerde seçilebilmektedir. En çok kullanılan sigmoid fonksiyonu, genelde tek parametreli alınmaktadır. [70]'de 3 parametreli sigmoid fonksiyonu incelenmiştir. Parametrelerden biri lineersizliği diğeri de bias ölçüsünü temsil etmektedir. Öğrenme işleminde bunların öğrenme hızını etkilediği görülmüştür.

Adaptif kontrol konusunda [71]'de yapılan çalışmada eğitim aşamasında momentumlu öğrenme kullanılarak yakınsama hızı artırılmaya çalışılmıştır. Ancak, öğrenme ve momentum katsayıları eğitim süresince sabit alınmıştır.

[72]'de ise adaptif kontrol uygulaması için, standart GY algoritması kullanılmıştır. Sabit öğrenme katsayısı kullanılıp değişik değerler için karşılaştırmalar yapılmıştır. Bu çalışmada ayrıca DSA yapısı da kullanılmıştır.

Bu çalışmaların dışında dinamik sistemlerin modellenmesi, kontrolü ve dinamik ve ÇKİB sinir ağlarıyla bunların gerçeklenmesi konularında ve ayrıca YSA' nın öğrenmesi konusunda yapılan çalışmalar mevcuttur [73-76].

Ayrıca YSA ile kontrol, tanıma, kestirim, zeki kontrol vb. konularla ilgili yeterli sayıda kitap bulmak da mümkündür [77-81].

Bu tez çalışmasında ele alınan optimal kontrol ve DSA konusundaki gelişmeler de devam etmektedir. Bu çalışmadaki temel yaklaşım, DSA, optimal kontrol için bir kontrol başlatıcısı olarak kullanılmasıdır.

Yapılan çalışmalardan biri, optimal kontrol teorisinden yararlanarak, sinir ağları için optimal ağırlıkların bulunmasına dayanır [82]. Burada, sürekli ve ayrık zamanlı ağlar için ağırlıkların zamanla değiştiği düşünülmüştür. Ağırlıklar, optimal kontrolde bulunmaya çalışılan kontrol fonksiyonuna karşı gelmektedir.

[83]'de verilen, belli bir ölçüt fonksiyonunu minimum yapacak şekilde istenen herhangi bir yörüngeyi takip edecek dinamik bir sistem için GB/İB kontrol yapısının tasarımlanması probleminin YSA vasıtasıyla çözülmesi çalışılmıştır. Genelde, dinamik programlama ve maksimum prensibe dayalı çözümleme yapılan bu problem, lineer yapıyı koruma prensibi ile isimlendirilen belli bir sinir mimarisiyle nöro-kontrol prensibi belirlenmiştir. Burada kullanacak kontrol fonksiyonun içine sinir ağırlıkları gömülüp, bu ağırlıkların optimize edilmesiyle problem çözülmektedir. Eğitim aşamasında, momentumlu GY algoritması kullanılmış, öğrenme katsayısı gittikçe iterasyon sayısıyla azalan bir fonksiyon yapısında seçilmiş olup momentum katsayısı sabit alınmıştır.

[84]'de ise, YSA tabanlı yakın zamanlı optimal kontrol metodu DC motor için incelenmiştir. ÇKİB nöral yapısı kullanılmıştır. DC motor, değişken darbe genişlik kontrol ve armatür voltaj darbe genişlik yapılarıyla kontrol edilmiştir. Deneysel olarak üretilen yakın zamanlı optimal yörüngelerden, zaman optimal kontrol kanunu YSA' ya öğretilmiştir.

[85]'de yine, zaman optimal kontrol probleminin YSA'ya uygulanışı incelenmiştir. Lineer olmayan optimal kontrol problemlerinde kontrol kanunu sinir ağı vasıtasıyla üretilmesi amaçlanmıştır. Burada sinir ağı iteratif olarak kontrol fonksiyonunu öğrenmektedir. Bu işlem, her an son durum hatası olarak kontrol aksiyonunu tamamlamak için iteratif olarak çözülür. Yapı olarak ÇKA mimarisi kullanılmıştır.

[86]'da GB kontrol yapısı optimal kontrol yardımıyla elde edildikten sonra, zaman içinde GY metoduyla İBSA yapısına uygulanmıştır. Zaman optimal BPTT (TOBPTT) çapraz koşullar altında gradyan tabanlı olarak geliştirilip uygulanmıştır .Ve minimum zamanlı kontrol problemine uygulanarak test edilmiştir .

[87]'de optimal kontrol problemi sonunda elde edilen yörüngeler, İB YSA'ya verilerek kontrolcü yapılmıştır. Bu çalışmada F-18 lineer modelli uçak üzerinde denenmiştir. Hem optimal kontrol problemi çözülmüş ve hem de sonuçlar YSA'nın eğitimi için kullanılmıştır. Optimal kontrolün çözümü ve sinir ağının eğitimi için tek

boyutlu aramalı konjuge gradyan yöntemi kullanılmıştır. Ayrıca optimal nöro-kontrol konusunda yapılan önemli çalışmalarda mevcuttur [88,89].

Genel anlamda YSA'nın kullanımı büyük bir statik ve dinamik optimizasyon problemini çözmeye eşdeğerdir denilebilir. Parametre tanıma veya sistem tanıma konuları da bu problemle ilgilidir. Sistem tanıma problemi gerçek anlamda uygulama için kaçınılmaz bir konudur [90,91]. Bu anlamda YSA'larına bakıldığında bir çok parametreye sahip bir sistem olarak görebiliriz. Bu parametrelerin istenilen giriş çıkış verilerine göre bulunması bir optimizasyon problemidir [92-95]. Sistem tanıma ve parametre bulunması konusunda kullanılan gradyan yöntemler bu tez çalışmasının kapsamı içerisindedir [96].

Bu tez çalışmasında DSA kullanılmış olup ileride avantajlarından bahsedileceği gibi yüksek depolama kapasitesine sahiptir ve dinamik yapıya sahip bir sistemi taklit etmesi kolaydır [97,98]

Hem optimal kontrolde, hem de YSA' da sürekli zamanlı dinamik sistemlerle ilgilenildiği için, sayısal çözümlemelerden yararlanılmıştır. Sayısal diferansiyel denklem çözümü, sayısal integral alma, lineer sistem çözümü, iteratif çözüm gibi konular için sayısal analizden faydalanılmıştır [94,99-106].

Yapılan bu tez çalışmasının organizasyonu şöyledir:

II. Bölümde zeki optimal kontrol kavramı ve yapısı anlatılmıştır. Bu çalışmanın geniş anlamda özetini ve amacını burada bulmak mümkündür.

III. Bölümde önce optimal kontrol problemleri tanıtılıp fonksiyon uzayında LOOK problemi ele alınarak, adjoint teoriden yararlanıp gerekli koşullar çıkarılmıştır. Bu koşullardan, birinci dereceden gradyan yöntemi verilerek iteratif çözüm ile optimal kontrol fonksiyonunun bulunması için gerekli algoritma kısaca verilmiştir. Daha sonra, hızlı yapılar için ikinci dereceden yöntemler incelenmiştir. Birincisi, direkt ölçütün azaltılmasına dayalı yöntem etraflıca çıkarılmıştır. Burada kontrol fonksiyonunu

sınırlayan ve probleme bir sınırlama katan kontrol sınırlama ağırlık matrisinin her adımda otomatik olarak hesaplanmasını sağlayan bir yapı geliştirilmiştir. İkincisi, Hamiltonian prensibine dayalı ikinci derece yöntemdir. Burada gerekli koşulların sağlanılması esası vardır. Literatürde de incelenmiş olan bu yönteme de kontrol sınırlama ağırlık matrisinin otomatik olarak bulunması yapısı eklenmiştir.

IV. Bölümde, ikinci dereceden LOOK yöntemlerinden yararlanarak optimal zamanla değişen geri besleme kanununun çıkarılması incelenmiştir.

V. Bölümde, LOOK ve geri besleme yapılarının denendiği uygulamalar verilmiştir.

VI. Bölümde, DSA konusu ele alınmıştır .YSA üzerine yapılan kısa açıklamadan sonra DSA yapısı ve dinamik denklemlerin çıkarılması ve ayrıca dinamik sistemler için gradyan hesabı adjoint (indirekt) metot kullanılarak verilmiştir. Bu yöntemin DSA'ya uygulanması incelenmiştir. Ayrıca, DSA da kullanılan en hızlı azalan gradyan yöntemi (EAGY) veya geriye yayınım (GY), ölçeklenmiş konjuge gradyan yöntemi (ÖKGY), Davidon-Fletcher-Powell (DFP) ve Broyden-Fletcher-Golfarb-Shanno (BFGS) eğitim yöntemleri anlatılmıştır. DSA'lı bir proses ile bu yöntemlerin denenmesi verilmiştir.

VII. Bölümde, bu tez çalışmasında kullanılan hibrid optimal nöro-kontrol yapısı verilmiştir. DSA ve LOOK probleminin birlikte kullanımı gösterilmiştir. DSA'nın optimal kontrol için., kontrol başlatıcısı olarak kullanılması verilip, GB yapısıyla tamamlanarak gerçek zamanlı uygulamalar için hızlı bir yapı olduğu gösterilmiştir. Uygulama olarak, DSA ve LOOK yapısının, VDP ve CSTR problemleri üzerinde gerçeklenmesi verilmiştir. Yapılan karşılaştırmalar geliştirilen yapı hakkında önemli sonuçlar çıkarmamızı sağlamıştır.

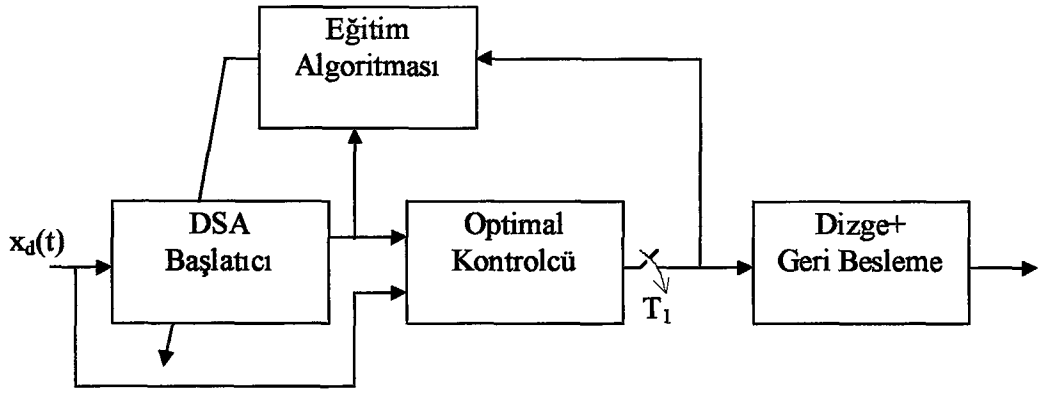
VIII. Bölümde sonuçlar ve IX. Bölümde ileride yapılabilecek çalışmalar hakkında tartışma konuları anlatılmıştır.

BÖLÜM 2. ZEKİ OPTİMAL KONTROL KAVRAMI

Bu çalışmanın temel amacı, direkt ikinci dereceden lineer olmayan optimal kontrol algoritmasının geliştirilmesi ve optimal kontrol eğitimindeki dinamik sinir ağları (DSA)'nın uygulamasının gerçekleştirilmesidir. Bu bağlamda, zeki kontrol yapısı oluşturmak ta çalışmanın hedeflerindendir.

Zeki kontrol, optimal kontrol teorisi gibi katı-bilimlerle ve sinir ağları (SA) gibi yumuşak-bilimlerle ilişkili teknolojilerin geniş bir alanını kapsar. Hibrid teknolojilere dayalı güvenli ve hesaplama yönünden etkin ürün yapıları, yeni fikirler olarak ele alınabilir. Zeki kontrol alanı, temel olarak kontrol teorisi, yapay zeka (YZ) ve deneyim sonuçlarını içerir.

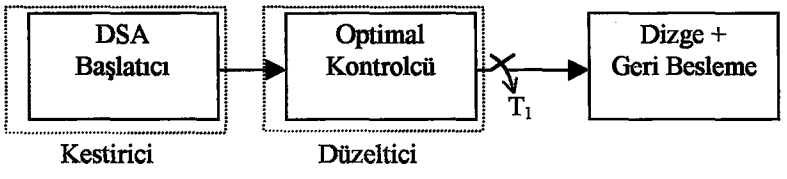
Bu çalışmada, optimal kontrol ve DSA teknolojileri zeki sistem anlamında hibrid bir yapı olarak birleştirilmiştir. Nöron sayısının seçimi, genelde deneysel yaklaşım olmasına rağmen, eğitim yapısı sağlam teorilere dayanır. Bu tez çalışmasında, optimal kontrol amacıyla kullanılan DSA'ın nöron sayısı bir yaklaşımla belirlenmeye çalışılmıştır. Bunlara ek olarak, toplam sistem, DSA tarafından genelleştirilen optimal kontrol fonksiyonu için bir zeki kontrol yapısı olarak düşünülmüştür. Sonuçta oluşturulan algoritma, DSA için bir otomatik-eğitici olarak çalışır ve az iterasyon sayısında ileri-beslemeli-optimal-kontrol (İBOK) fonksiyonu üretir. Yörünge boyunca, zamanla-değişen-optimal-geri-besleme (ZDOGB) kazançları da üretilir. Optimal geri-besleme, iyi bir şekilde sistemi istenilen yörüngede bozuculara rağmen tutabilmektedir. Algoritmanın kendi kendine öğrenme yapısı belirgin biçimde zeki kontrol özelliğini göstermektedir. Bu yapının blok diyagramı Şekil 2. 1'de gösterilmektedir. Burada, T_1 zamanı optimal kontrol iterasyonlarının bittiği zamanı ifade eder.



Şekil 2. 1 Kendi kendine öğrenen zeki optimal kontrol sistemi

Bu tez çalışmasının diğer bir yaklaşımı da, DSA'nın optimal kontrol başlatıcısı olarak kullanılıyor olmasıdır. Optimal kontrol olarak, ikinci dereceden direkt ölçütün azaltılması yöntemi (İDDÖAY) yaklaşımı geliştirilmiştir. Diğer yöntemlere göre daha gürbüz (robust) ve hızlı olduğu gösterilmiştir. DSA kullanımıyla daha da hızlı bir yapı elde edilmiştir. DSA tarafından üretilen yaklaşık optimal kontrol fonksiyonuyla ikinci dereceden optimal kontrol algoritması başlayarak hem doğruluğu ve hem de sistem güvenirliğinin sağlanması hedeflenmiştir. Algoritma, DSA açısından bir kestirici, optimal kontrolcü açısından bir düzeltici olarak düşünülmüştür. Bu yapı da Şekil 2. 2'de gösterildiği gibidir. Bu yapı dinamik-sinirsel-optimal-kontrolcü (DSOK) olarak isimlendirilir.

Geliştirilen zeki-optimal-kontrol (ZOK) algoritmasıyla, gerçek zamanlı uygulamaların gerçekleştirilmesi hedeflenmiştir. Bu yüzden, DSA eğitimi için klasik geriye-yayınım (GY) algoritmasının dışında, statik optimizasyon teorisinde geliştirilmiş ve kullanılan daha hızlı yöntemler uygulanmıştır. Bunun için, dinamik optimizasyon problemine adjoint teorisinden başlanarak iyi çözüm yaklaşımları geliştirilmiştir.



Şekil 2. 2 Kestirici/düzeltici anlamında (DSOK) blok diyagramı

BÖLÜM 3. FONKSİYON UZAYINDA LİNEER OLMAYAN OPTİMAL KONTROL

Lineer olmayan optimal kontrol (LOOK) teorisi [1,3,6,8] çözümü zor olan bir problemdir. Ancak, geliştirilen bir çok yöntem ile LOOK problemleri çözülebilmektedir. Lineer optimal kontrol (LOK) teorisi analitik yönden çözümlenmiş ve bir çok yönden ele alınmış bir konudur [1,6,7,33]. Fakat, LOOK problemine bir adımda getirilmiş analitik çözüm yoktur. Çözümleme için değişik iteratif yöntemler geliştirilmiştir. Bu çalışmada, bu yöntemler ve değişik bir yaklaşımla geliştirilen bir yöntemden bahsedilecektir. Bu kontrol yönteminin dışında, lineer olmayan sistemler ve kontrol konusu da optimal kontrol teorisinden farklı tarzda konuyu ele alır [107,108].

3.1. Problemin İfadesi ve Kullanılacak Notasyonlar

Lineer olmayan bir dinamik sistemin durum denklemi aşağıdaki gibi verilmiştir.

$$\dot{x} = f(x(t), u(t), t), \quad x(t_0) = x_0 \tag{3.1}$$

Burada, $x(t) \in R^n$ durum vektör fonksiyonu, $u(t) \in R^m$ kontrol vektör fonksiyonudur. Bu sistemin kontrol edilebilir olduğu varsayılmaktadır. $x(t_0)$ sistemin başlangıç durum vektörüdür. Eğer u(t) fonksiyonu verilirse, $x(t_0)$ başlangıç koşulu altında x(t) yörüngesi bulunabilir. Eğer sistem t'ye açık olarak bağlı değilse (3.1) eşitliği şöyle olur:

$$\dot{x} = f(x(t), u(t)), \quad x(t_0) = x_0 \tag{3.2}$$

Eğer sistem denge durumunda ise, yani zamanla yörünge değişmiyorsa sistem sürekli durumdadır. Bu durumda $\dot{x}=0$ olur.

Optimal kontrol, verilen bir ölçüt fonksiyonunu (performans ölçütü,PÖ) en az yapacak şekilde kontrol fonksiyonunu bulmak için kullanılır. Burada PÖ 'nün seçimi tasarım problemi ile ilgilidir ki biz bu ölçütün verildiğini varsayıyoruz. Ancak belli yapılardaki PÖ' nü (mesela, kuadratik ölçüt) anlamlandırmak ve üzerinde yorum yapmak mümkündür.

Bu çalışmada incelenecek olan genel yapıdaki PÖ fonksiyonu, BOLZA PROBLEMİ olarak bilinir ve şöyle ifade edilebilir:

$$J = \phi(x(t_f)) + \int_{t_0}^{t_f} L(x(t), u(t))dt \qquad (3.3)$$

Burada J skaler yapıda olup PÖ' nü gösterir. t_0, başlama zamanı, t_f ise ilgilenilen son zaman olup sabit alınmaktadır. ϕ son durum ağırlık fonksiyonu veya terminal sınırlaması, L ise ağırlık fonksiyonudur. ϕ ve L fonksiyonları sistem için belirlenen amaca uygun olarak seçilir. Sistem diferansiyel denklemi $\dot{x}$, PÖ için bir dinamik sınırlamadır.

(3.3) eşitliğinde eğer ϕ=0 alınırsa, o zaman problem LAGRANGE PROBLEMİ olarak bilinir ve,

$$J = \int_{t_0}^{t_f} L(x(t), u(t))dt \qquad (3.4)$$

biçimine dönüşür.

Eğer (3.3) eşitliğinde L=0 yapılırsa, o zaman problem MAYER PROBLEMİ olarak isimlendirilir ve,

$$J = \phi(x(t_f)) \qquad (3.5)$$

şekline dönüşür [6,8,109].

3.2. Lineer Olmayan Optimal Kontrol Probleminin Çözümü

Bu çalışmadaki, optimal kontrolde temel amaç, ikinci dereceden gürbüz (robust) sayısal yöntemleri geliştirmektir. Ancak, çözüm yöntemlerine başlayabilmek ve tamamlılık açısından birinci dereceden yöntem ve gerekli koşulların çıkartılmasıyla problemin çözümüne başlanacaktır.

3.2.1. Değişimlerin hesabı ve birinci dereceden gradyan yöntemi (BDGY) ile ileri beslemeli optimal kontrol (İBOK)

Bu çözüm için, Lagrange çarpanı metodundan faydalanarak yola çıkılır. Bu amaçla, Bolza Problemi ele alınacaktır. Önce, artırılmış PÖ (APÖ) diye isimlendirilen ve dinamik sınırlamanın ortadan kaldırıldığı bir yapı ile minimizasyon yapmaya problem indirgenir ki APÖ aşağıda verilmiştir:

$$J_a = \phi(x(t_f) + \int_{t_0}^{t_f} \{L(x,u) + \lambda^T(t)[f(x,u) - \dot{x}]\}dt \tag{3.6}$$

Burada, J_a skaler APÖ, $\lambda(t) \in R^n$ zamanla değişen Lagrange çarpanıdır. Kolaylık alması açısında durum ve kontrol fonksiyonlarında zaman argümanı konulmamıştır. Uygunluk açısından skaler Hamiltonian fonksiyonu tanımlanırsa,

$$H(x,u,\lambda) = L(x,u) + \lambda^T f(x,u) \tag{3.7}$$

Ve APÖ bu şart altında tekrar yazılırsa,

$$J_a = \phi(x(t_f) + \int_{t_0}^{t_f} \{H(x,u,\lambda) - \lambda^T \dot{x}\}dt \tag{3.8}$$

Lagrange çarpanı teorisine göre, (3.3) eşitliği ile verilen J fonksiyonelini en az yapan u(t) kontrol fonksiyonu (3.2) sınırlaması altında bulunması yerine, (3.8) eşitliği ile

verilen J_a fonksiyonelinin sınırlama olmaksızın en az yapılarak bulunmasına dönüştürülür. Bu noktadan bakılırsa, J_a fonksiyoneli değişimleri hesabı (calculus of variation) teorisi kullanılarak küçük artırımlar için hesaplama yapılabilir. Bu teori uygulanırsa, J_a' nın birinci varyasyonu için şu ifade elde edilir:

$$\delta J_a = (\phi_x^T - \lambda^T)\delta x\Big|_{t=t_f} + \int_{t_0}^{t_f} \left\{(H_x^T + \dot{\lambda}^T)\delta x + H_u^T \delta u + (H_\lambda^T - \dot{x}^T)\delta\lambda\right\} dt \qquad (3.9)$$

Burada, H ve ϕ fonksiyonlarının alt indisleri o değişkenlere göre kısmi türevleri olup problemin çözümü Ek A 'da verilmiştir. Buradan gerekli koşullar için,

$$\dot{x} = H_\lambda \quad , \quad x(t_0) = x_0 \qquad (3.10)$$

$$\dot{\lambda} = -H_x \quad , \quad \lambda(t_f) = \phi_x(t_f) \qquad (3.11)$$

$$0 = H_u \qquad (3.12)$$

elde edilir. Burada $\dot{\lambda}$ adjoint diferansiyel denklemidir. Durum ve adjoint sistemi, Hamiltonian sistemi olarak isimlendirilir. H_u ise durağanlık şartı, yani optimal noktadaki durumu belirtir. Eğer H_u'dan u(t) kontrol fonksiyonu çekilebilirse Hamiltonian sistemi iki-noktalı-sınır-değeri problemi (İNSDP) olarak literatürde ele alınmıştır.

(3.12) eşitliği ancak optimal kontrol fonksiyonu bulunduğunda sağlanabilir. Diğer durumlarda eşitlik sağlanmaz. Bu durumda, herhangi bir $u^0(t)$ kontrol fonksiyonundan başlayarak iteratif olarak çözüme gitmek mümkündür. Bu durumda birinci varyasyon aşağıdaki duruma gelir:

$$\delta J_a = \int_{t_0}^{t_f} H_u^T \delta u dt \qquad (3.13)$$

Amaç δJ_a 'yı her adımda azaltmaktır. Bunu sağlayacak durum ise,

$$\delta u = -\tau H_u \quad , \quad \tau > 0 \tag{3.14}$$

olarak seçmektir. Bu durumda birinci varyasyon,

$$\delta J_a = -\tau \int_{t_0}^{t_f} H_u^T H_u dt \tag{3.15}$$

olup azalması garantilenir. Burada τ adım büyüklüğü olup her adımda en iyisi bulunabilir:

$$\min_{\tau} J(u - \tau H_u) \tag{3.16}$$

olarak tek boyutlu minimizasyon problemi çözülür. Veya yakınsamayı sağlayacak bir değerde sabit alınabilir. İterasyon bakımından bakıldığında, başlanılan herhangi bir $u^k(t)$ 'ye $\delta u^k(t)$ kadar bir artırımsal kontrol eklenerek optimal noktaya ulaşılmaya çalışılır.

Gerekli koşullar tekrar düzenlenirse,

$$\dot{x} = H_\lambda = f(x,u) \quad , \quad x(t_0) = x_0 \tag{3.17}$$

$$\dot{\lambda} = -H_x = -L_x - f_x^T \lambda \quad , \quad \lambda(t_f) = \phi_x(t_f) \tag{3.18}$$

$$0 = H_u = L_u + f_u^T \lambda \tag{3.19}$$

Bu anlatılanlara göre birinci dereceden gradyan algoritması aşağıda verilmiştir:

i.) $u^0(t), \tau$ ve γ 'yı seç. k=0.

ii.) $\dot{x} = f(x, u^k)$, $x(t_0)=x_0$, için (t_0,t_f) aralığında x(t) 'yi çöz.

iii.) $\lambda(t_f)$ için, $\phi_x(x(t_f))$ 'i x(t)'den bul.

iv.) Adjoint diferansiyel denklemi, $\dot{\lambda} = -H_x$, $\lambda(t_f) = \phi_x(t_f)$ için zamanda geriye doğru çöz.

v.) H_u 'yu hesapla.

vi.) $\|H_u\| \geq \gamma$ ise $u^{k+1}(t) = u^k(t) - \tau H_u^k$ 'yu hesapla. k=k+1 yap ve ii)'ye dön.

$\|H_u\| < \gamma$ ise dur. $u^{k+1}(t)=u^*(t)$ olarak sakla.

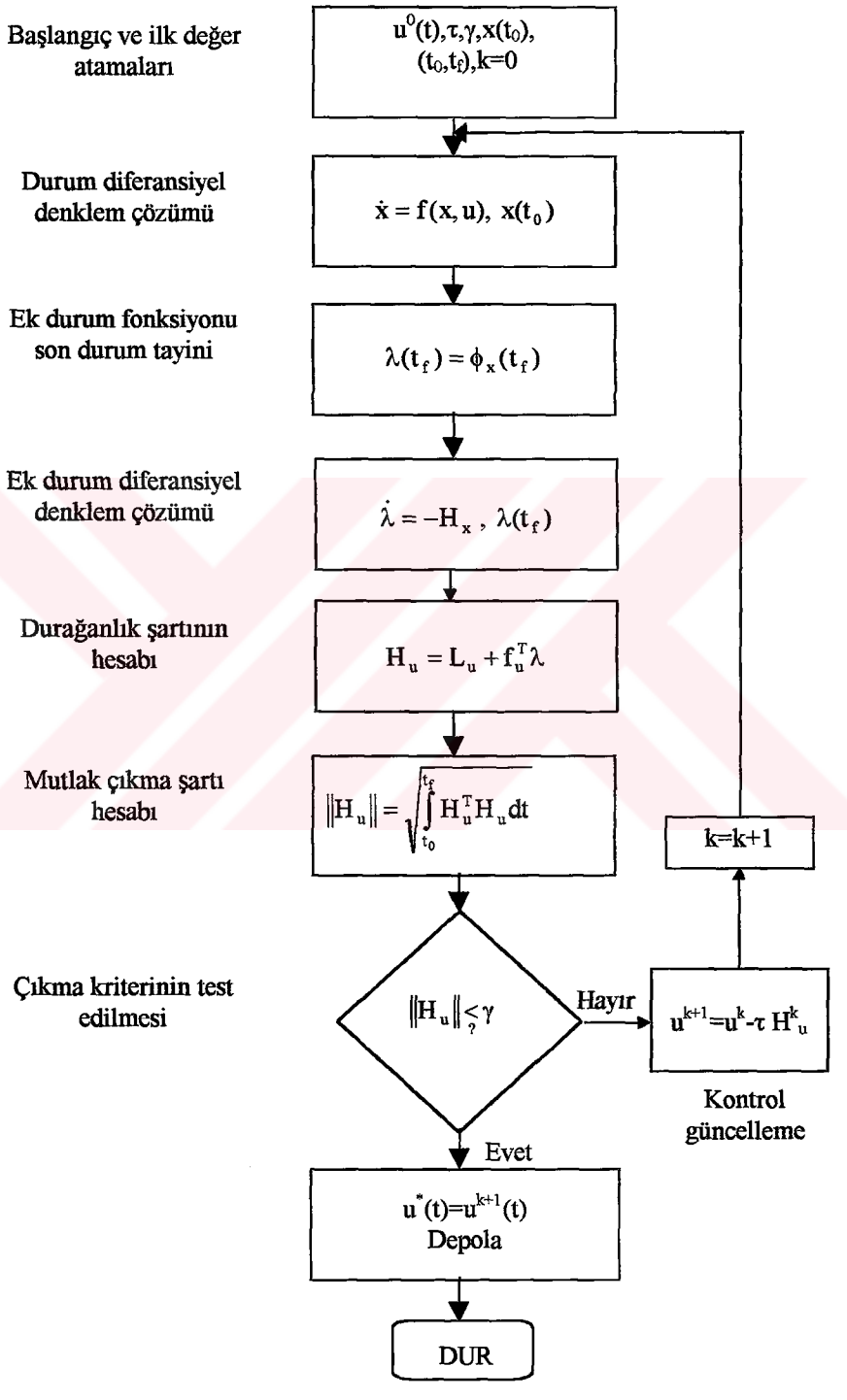
Bu çalışmadaki uygulamalarda, sayısal diferansiyel çözümü için 5. dereceden Runga-Kutta-Butcher metodu kullanılmıştır. Zamanda ileri ve geriye doğru çözümlerde iyi derecede çözümler sağlamaktadır. İntegrasyon zaman aralığının yeteri kadar küçük olması gerekir. Şekil 3. 1 'de birinci dereceden optimal kontrol gradyan algoritması gösterilmektedir.

İBOK bir açık çevrim kontrolüdür. Eğer sistem verilen şartları taşırsa, sistem istenilen yörüngeyi takip edecektir. İBOK için sistem blok diyagramı Şekil 3. 2 'de verilmektedir. Şekil 3. 2 'den görüldüğü gibi, k. iterasyon anında işlemler bitirilmiş olup bulunan kontrol fonksiyonu optimaldir ve sisteme uygulanır.

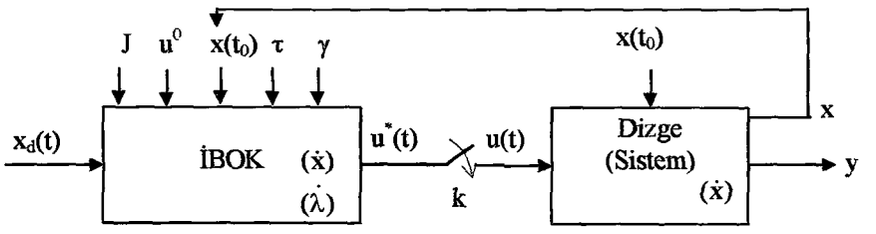
3.2.2. İkinci dereceden gradyan yöntemler

İkinci dereceden gradyan yöntemler, daha hızlı oldukları için tercih edilir. Ancak, daha ağır hesap yüküne sahiptir. Bununla beraber, toplam zaman maliyeti yönünden bakıldığında çok iyi durumdadır. BDGY 'ler optimal noktaya yaklaşıldığında çok yavaş ilerler. Ancak ikinci dereceden yöntemler ise optimal noktaya yaklaşıldıkça çok hızlı çözüme yakınsar. İkinci dereceden terimlerin etkisi daha fazla olduğundan, yakın noktada kuadratik olması yaklaşımı bunu doğrular. Bu çalışmada, ikinci dereceden yöntemlerden iki yaklaşım verilecektir. Birincisi, bu tez çalışmasında üzerinde yoğun bir şekilde çalışılan ve direkt ölçüt fonksiyonunun Taylor serisine açılması esasına dayanan yöntemdir [19,109]. İkincisi ise, literatürde genelde kullanılan ve Hamiltonian

prensibine dayalı olup gerekli koşulları sağlamayı hedefleyen yöntemdir [6,8,16,19,20,50].



Şekil 3. 1 İBOK için BDGY ile LOOK gradyan algoritması akış diyagramı



Şekil 3. 2 İBOK için sistem blok diyagramı

3.2.2.1. İkinci dereceden direkt ölçütün azaltılması yöntemi (İDDÖAY) ve İBOK

Bu metotta PÖ, direkt olarak ikinci dereceye kadar Taylor serisine açılacaktır. Bu yaklaşım, Newton tipine benzemektedir. Bu çözüm, herhangi bir ara sistem kullanmadan direkt olarak artırımsal kontrol fonksiyonunu bulmaya yöneliktir. Şimdi, (3.3) eşitliği ile verilen PÖ , (x,u) yörüngeleri etrafında $u+\delta u$ ve $x+\delta x$ biçiminde ikinci terime kadar Taylor açılımı yazılırsa,

$$J(x+\delta x,u+\delta u)=\phi(x(t_f))+\phi_x^T(t_f)\delta x(t_f)+\tfrac{1}{2}\delta x^T(t_f)\phi_{xx}\delta x(t_f)$$
$$+\int_{t_0}^{t_f}\left\{L(x,u)+\begin{bmatrix}L_x^T & L_u^T\end{bmatrix}\begin{bmatrix}\delta x\\ \delta u\end{bmatrix}+\tfrac{1}{2}\begin{bmatrix}\delta x^T & \delta u^T\end{bmatrix}\begin{bmatrix}L_{xx} & L_{xu}\\ L_{ux} & L_{uu}\end{bmatrix}\begin{bmatrix}\delta x\\ \delta u\end{bmatrix}\right\}dt+(y.d.t) \qquad (3.20)$$

Burada, L ve ϕ ifadelerindeki alt indisler değişkenlere göre birinci veya ikinci dereceden kısmi türevleri göstermektedir. (y.d.t), yüksek dereceli terimleri göstermektedir. İntegral ifadesindeki matris simetriktir. Bundan sonra, yöresel ölçüt için fark ifadesi yazılırsa,

$$\Delta J = J(x+\delta x,u+\delta u) - J(x,u) \qquad (3.21)$$

Ve ifadeler yerlerine yazılırsa aşağıdaki ifadeye ulaşılır:

$$\Delta J=\phi_x^T(t_f)\delta x(t_f)+\tfrac{1}{2}\delta x^T(t_f)\phi_{xx}\delta x(t_f)$$
$$+\int_{t_0}^{t_f}\left\{\begin{bmatrix}L_x^T & L_u^T\end{bmatrix}\begin{bmatrix}\delta x\\ \delta u\end{bmatrix}+\tfrac{1}{2}\begin{bmatrix}\delta x^T & \delta u^T\end{bmatrix}\begin{bmatrix}L_{xx} & L_{xu}\\ L_{ux} & L_{uu}\end{bmatrix}\begin{bmatrix}\delta x\\ \delta u\end{bmatrix}\right\}dt+(y.d.t) \qquad (3.22)$$

(3.22) ifadesine bakıldığında problem, herhangi bir ara işlem yapılmaksızın PÖ üzerinde operasyon yapılarak çözülmeye çalışılmaktadır. Bu ifade, yöresel PÖ (YPÖ) olarak isimlendirilir. (3.22) ifadesine, yöresel veya artırımsal kontrol fonksiyonu δu'yu sınırlandırmak maksadıyla bir integral ceza fonksiyonu eklenir. Eklenen bu ifade aynı zamanda problemin çözümünde pozitif tanımlılık şartlarının sağlanması maksadıyla da kullanılacaktır. Eklenen ifade ise aşağıdaki gibidir:

$$\frac{1}{2}\int_{t_0}^{t_f}(\delta u^T W \delta u)dt \tag{3.23}$$

Bu şart altında YPÖ (y.d.t)'in dışında aşağıda verilmektedir:

$$\Delta J(\delta u) = \phi_x^T(t_f)\delta x(t_f) + \tfrac{1}{2}\delta x^T(t_f)\phi_{xx}\delta x(t_f) + \int_{t_0}^{t_f}\left\{\begin{bmatrix}L_x^T & L_u^T\end{bmatrix}\begin{bmatrix}\delta x\\ \delta u\end{bmatrix} + \tfrac{1}{2}\begin{bmatrix}\delta x^T & \delta u^T\end{bmatrix}\begin{bmatrix}L_{xx} & L_{xu}\\ L_{ux} & L_{uu}+W\end{bmatrix}\begin{bmatrix}\delta x\\ \delta u\end{bmatrix}\right\}dt \tag{3.24}$$

Burada, $\Delta J(\delta u) = \delta J + \delta^2 J$ şeklinde olduğu görülmektedir. δJ birinci, $\delta^2 J$ ise ikinci varyasyondur. $W \in R^{m\times m}$ olup simetrik kontrol sınırlama matrisidir. Buradan görüleceği gibi YPÖ, yöresel kontrol fonksiyonu δu'ya bağlıdır. Sonuçta, problem bir yöresel optimizasyon problemine dönüşmüştür. Burada kullanılacak model ise yöresel model olacaktır ki aşağıda verilmektedir:

$$\delta\dot{x} = f_x\delta x + f_u\delta u \quad , \quad \delta x(t_0) = 0 \tag{3.25}$$

Sonuçta yöresel optimizasyon problemi çözülecektir:

$$\min_{\delta u} \Delta J(\delta u) \tag{3.26}$$

$$\text{sınırlama } f_x\delta x + f_u\delta u - \delta\dot{x} = 0 \quad , \quad \delta x(t_0) = 0 \tag{3.27}$$

Buradaki problem birinci dereceden gradyan yönteminde incelenen yapıya dönüştürülmüştür. Bu benzerliği göstermek için aşağıdaki notasyonlar kullanılacaktır:

$$\Delta J = \tilde{\phi}(\delta x(t_f)) + \int_{t_0}^{t_f} \tilde{L}(\delta x(t), \delta u(t))dt \tag{3.28}$$

Burada,

$$\tilde{\phi}(\delta x(t_f)) = \phi_x^T(t_f)\delta x(t_f) + \tfrac{1}{2}\delta x^T(t_f)\phi_{xx}\delta x(t_f)$$

(3.29)

$$\tilde{L}(\delta x, \delta u) = \begin{bmatrix} L_x^T & L_u^T \end{bmatrix}\begin{bmatrix} \delta x \\ \delta u \end{bmatrix} + \tfrac{1}{2}\begin{bmatrix} \delta x^T & \delta u^T \end{bmatrix}\begin{bmatrix} L_{xx} & L_{xu} \\ L_{ux} & L_{uu} + W \end{bmatrix}\begin{bmatrix} \delta x \\ \delta u \end{bmatrix} \tag{3.30}$$

Burada, $\tilde{L}$ ifadesinde görülen ikinci dereceden kısmi türevlerden oluşan matris simetriktir. Çözüm için önceki bölümde izlenilen yol takip edilecektir. Önce, sınırlı problem, sınırsız probleme dönüştürülecektir. Bunun için artırılış ΔJ şöyle tanımlanır:

$$\Delta J_a = \tilde{\phi}(\delta x(t_f)) + \int_{t_0}^{t_f} \{\tilde{L}(\delta x, \delta u) + \tilde{\lambda}^T(t)(f_x\delta x + f_u\delta u - \delta\dot{x})\}dt \tag{3.31}$$

Burada, $\tilde{\lambda}^T(t) \in R^n$ artırımsal Lagrange çarpanı veya yardımcı adjoint değişkeni olup zamana bağlıdır. Buna bağlı olarak artırımsal Hamiltonian ise aşağıda verilmektedir:

$$\tilde{H}(\delta x, \delta u, \tilde{\lambda}) = \tilde{L}(\delta x, \delta u) + \tilde{\lambda}^T(f_x\delta x + f_u\delta u) \tag{3.32}$$

Bu problem için, önceki bölümdeki gibi gerekli koşulları hemen yazmak mümkündür. Yazılan tüm ifadeler artırımsal değişkenlere bağlıdır. Ve gerekli koşullar aşağıdaki gibidir:

$$\delta\dot{x} = \tilde{H}_{\tilde{\lambda}} = f_x\delta x + f_u\delta u \quad , \quad \delta x(t_0) = 0 \tag{3.33}$$

$$\dot{\tilde{\lambda}} = -\tilde{H}_{\delta x} = -(L_x + L_{xx}\delta x + L_{xu}\delta u + f_x^T\tilde{\lambda}) \quad , \quad \tilde{\lambda}(t_f) = (\phi_x(t_f) + \phi_{xx}\delta x(t_f)) \tag{3.34}$$

$$0 = \tilde{H}_{\delta u} = L_u + (L_{uu} + W)\delta u + L_{ux}\delta x + f_u^T\tilde{\lambda} \tag{3.35}$$

Görüldüğü gibi, ikinci dereceden yöntem için çıkarılan gerekli koşullar birinci dereceden yönteme benzer koşullardır. (3.35) 'ten δu çekilirse,

$$\delta u(t) = -(L_{uu} + W)^{-1}[L_u + L_{ux}\delta x + f_u^T\tilde{\lambda}] \tag{3.36}$$

Bu ifade (3.33) ve (3.34) 'te yerlerine yazılırsa aşağıdaki ifadelere ulaşılır:

$$\delta\dot{x} = f_x\delta x - f_u(L_{uu} + W)^{-1}[L_u + L_{ux}\delta x + f_u^T\tilde{\lambda}] \quad , \quad \delta x(t_0) = 0 \tag{3.37}$$

$$-\dot{\tilde{\lambda}} = L_x + L_{xx}\delta x - L_{xu}(L_{uu} + W)^{-1}[L_u + L_{ux}\delta x + f_u^T\tilde{\lambda}] + f_x^T\tilde{\lambda}, \quad \tilde{\lambda}(t_f) = \phi_x(t_f) + \phi_{xx}\delta x(t_f) \tag{3.38}$$

Bu son iki eşitlik azaltılmış Euler-Langrange (AEL) diferansiyel denklemi veya iki noktalı sınır değeri problemi (İNSDP) adını almaktadır. Ve bu ifadeler düzenlenirse aşağıdaki eşitlikler elde edilir:

$$\delta\dot{x} = [f_x - f_u(L_{uu} + W)^{-1}L_{ux}]\delta x - [f_u(L_{uu} + W)^{-1}f_u^T]\tilde{\lambda} - f_u(L_{uu} + W)^{-1}L_u, \quad \delta x(t_0) = 0 \tag{3.39}$$

$$\dot{\tilde{\lambda}} = -[L_{xx} - L_{xu}(L_{uu} + W)^{-1}L_{ux}]\delta x - [f_x^T - L_{xu}(L_{uu} + W)^{-1}f_u^T]\tilde{\lambda} - L_x + L_{xu}(L_{uu} + W)^{-1}L_u \quad , \quad \tilde{\lambda}(t_f) = \phi_x(t_f) + \phi_{xx}\delta x(t_f) \tag{3.40}$$

Bu ifadeler daha modüler yapıda şöyle yazılabilir:

$$\delta\dot{x} = A(t)\delta x + B(t)\tilde{\lambda} + v(t) \quad , \quad \delta x(t_0) = 0 \tag{3.41}$$

$$\dot{\tilde{\lambda}} = -C(t)\delta x - A^T(t)\tilde{\lambda} - w(t) \quad , \quad \tilde{\lambda}(t_f) = \phi_x(t_f) + \phi_{xx}\delta x(t_f) \tag{3.42}$$

Burada,

$$A(t)\big|_{n\times n} = f_x - f_u(L_{uu} + W)^{-1}L_{ux} \tag{3.43}$$

$$B(t)\big|_{n\times n} = -f_u(L_{uu} + W)^{-1}f_u^T \tag{3.44}$$

$$C(t)\big|_{n\times n} = L_{xx} - L_{xu}(L_{uu} + W)^{-1}L_{ux} \tag{3.45}$$

$$v(t)\big|_{n\times 1} = -f_u(L_{uu} + W)^{-1}L_u \tag{3.46}$$

$$w(t)\big|_{n\times 1} = L_x - L_{xu}(L_{uu} + W)^{-1}L_u \tag{3.47}$$

Euler-Lagrange denklemlerinin çözümü, genel Riccati dönüşümü kullanılarak çözülecektir. Genel Riccati dönüşümü, artırımsal adjoint diferansiyel denkleminin son şart yapısına benzetim yapılarak oluşturulmuştur. Bu dönüşüm aşağıdaki gibidir:

$$\tilde{\lambda} = P\delta x + q \quad , \quad P(t_f) = \phi_{xx}(t_f), \quad q(t_f) = \phi_x(t_f) \tag{3.48}$$

Bu dönüşümde, İNSDP' nin çözümü tam olarak yapılmaktadır. Burada, $P(t) \in R^{nxn}$ Riccati dönüşüm matrisi olup simetriktir. $q(t) \in R^n$ ise sürücü dönüşüm vektörüdür. Problem, P(t) ve q(t) değişkenlerinin diferansiyel denklemlerinin bulunmasıyla çözülebilecektir. (3.48) eşitliğinin her iki tarafının türevi alınırsa şu şekle dönüşür:

$$\dot{\tilde{\lambda}} = \dot{P}\delta x + P\delta\dot{x} + \dot{q} \tag{3.49}$$

Gerekli ifadeler yerine yazılırsa,

$$-C\delta x - A^T\tilde{\lambda} - w = \dot{P}\delta x + P(A\delta x + B\tilde{\lambda} + v) + \dot{q} \tag{3.50}$$

$$(\dot{P} + PA + C)\delta x + (PB + A^T)\tilde{\lambda} + Pv + w + \dot{q} = 0 \tag{3.51}$$

$$(\dot{P} + PA + C)\delta x + (PB + A^T)(P\delta x + q) + Pv + w + \dot{q} = 0 \tag{3.52}$$

$$(\dot{P} + A^T P + PA + PBP + C)\delta x + \dot{q} + A^T q + Pv + PBq + w = 0 \tag{3.53}$$

Son eşitliğin δx'ten bağımsız olabilmesi için katsayısı sıfıra eşitlenir ve aşağıdaki eşitlikler elde edilir:

$$\dot{P} = -(A^T P + PA + PBP + C) \quad , \quad P(t_f) = \phi_{xx}(t_f) \tag{3.54}$$

$$\dot{q} = -(A^T q + Pv + PBq + w) \quad , \quad q(t_f) = \phi_x(t_f) \tag{3.55}$$

Bu eşitliklerde, P simetrik matris olup matris Riccati diferansiyel denklemi, q ise vektör diferansiyel denklemidir. Bu iki denklem çözüldükten sonra, $\tilde{\lambda}$ vektörü bulunabilir ve $\delta u(t)$ için aşağıdaki ifadeler yazılabilir:

$$\delta u(t) = -(L_{uu} + W)^{-1}[L_u + L_{ux}\delta x + f_u^T(P\delta x + q)] \tag{3.56}$$

$$\delta u(t) = -(L_{uu} + W)^{-1}[L_{ux} + f_u^T P]\delta x - (L_{uu} + W)^{-1}(L_u + f_u^T q) \tag{3.57}$$

$$\delta u(t) = K(t)\delta x + b(t) \tag{3.58}$$

Burada,

$$K(t) = -(L_{uu} + W)^{-1}[L_{ux} + f_u^T P] \tag{3.59}$$

$$b(t) = -(L_{uu} + W)^{-1}(L_u + f_u^T q) \tag{3.60}$$

dir. Ve $K(t) \in R^{mxn}$ olup kazanç matrisi, $b(t) \in R^m$ olup sürücü kazanç vektör ifadeleridir.

Görüldüğü gibi, direkt ölçütün azaltılmasıyla elde edilen bu ikinci dereceden yöntemde adjoint değişkenin ayrıca hesabı yoktur. Bu önemli bir sonuçtur. Bu yöntemde toplam, **n** adet zamanda ileri ve **n+n(n+1)/2** adet zamanda geriye doğru olmak üzere toplam **2n+n(n+1)/2** adet diferansiyel denklem çözmek gerekmektedir.

Bu yöntem, iteratif bir yapıya dönüştürülerek İBOK algoritması çıkarılabilir. Artırımsal kontrol fonksiyonu için iteratif yapı şu şekilde ifade edilir:

$$u^{k+1}(t) = u^{k}(t) + \delta u^{k}(t) \tag{3.61}$$

Burada **k** iterasyon sayısıdır. İşlem, artırımların durması veya yeteri kadar yavaşlamasına kadar devam ettirilir. Bütün ifadelerde görülen ve kontrol sınırlama matrisi olarak isimlendirilen **W** matrisi genelde diyagonal seçilir. İterasyona başlarken yeteri kadar büyük seçilerek (L_{uu}+W) 'nın pozitif tanımlı olması sağlanır. Buraya kadar yapılan işlemlerde aşağıdaki varsayımların sağlandığı kabul edilmiştir:

$$\phi_{xx} \geq 0 \quad , \quad \text{pozitif yarı - tanımlı} \tag{3.62}$$

$$(L_{uu}+W)>0 \quad , \quad \text{pozitif tanımlı} \tag{3.63}$$

$$L_{xx} - L_{xu}(L_{uu} + W)^{-1} L_{ux} \geq 0 \quad , \quad \text{pozitif yarı - tanımlı} \tag{3.64}$$

Yukarıdaki varsayımlar, Riccati matris diferansiyel denkleminin yakınsayan çözümü ve azalan ölçüt için sağlanmalıdır. Bir anlamda da, çözümlemede konjuge noktaya takılmamak için gereklidir. Değişik şekillerde konjuge nokta tanımı yapılmaktadır. Burada konjuge nokta kullanımı, P 'nin zamanda değişen köklerinin sol yarı düzleme geçmesi durumundadır. Yani, **P<0** ise (negatif tanımlı) konjuge nokta var demektir. Bunun üstesinden gelmek için yukarıda da bahsedilen W kontrol sınırlama matrisinin seçimi önemlidir. Burada, W 'nın her iterasyonda belirlenebilmesi için bir yapı geliştirilmiştir. Eğer, sistem lineer, PÖ ise kuadratik olup gerekli şartlar sağlansaydı

W=0 olacaktı. Çünkü, bir adımda istenilen optimal noktadaki kontrol fonksiyonu bulunabilir. Yani, W 'nın seçimi kuadratiklik miktarının ölçüsü olarak belirlenmelidir. Bu yapıya benzer olarak statik optimizasyonda yaklaşık-Newton metodunda Fletcher 'in yaklaşımına benzemektedir [92]. Burada bunun için, PÖ 'nün yapısından yararlanılmıştır. (3.22) eşitliğine dikkat edilirse, ifade lineer ve kuadratik açılımların toplamı biçimindedir. Aranılan bilgi, kuadratiklik noktasından ne kadar uzakta olunduğu bilgisi olduğuna göre, lineer ve kuadratik ifadeler bu bilgiyi verecektir. Kuadratik ifadenin, lineer ifadeye oranı bir katsayı ifadesi olarak bu ölçüyü verecektir. Bu katsayı ile W'nın her adımda güncellenmesi yapılmıştır. Bu söylenenler matematiksel olarak ifade edilirse şu yapılar oluşacaktır:

$$W^{k+1} = \alpha^k W^k \tag{3.65}$$

$$\alpha^k = \left| \frac{\frac{1}{2}\delta x^T(t_f)\phi_{xx}\delta x(t_f) + \int_{t_0}^{t_f}\left\{\frac{1}{2}\begin{bmatrix}\delta x^T & \delta u^T\end{bmatrix}\begin{bmatrix}L_{xx} & L_{xu}\\ L_{ux} & L_{uu}\end{bmatrix}\begin{bmatrix}\delta x\\ \delta u\end{bmatrix}\right\}dt}{\phi_x\delta x(t_f) + \int_{t_0}^{t_f}\left\{\begin{bmatrix}L_x^T & L_u^T\end{bmatrix}\begin{bmatrix}\delta x\\ \delta u\end{bmatrix}\right\}dt} \right| \tag{3.66}$$

Böylece, W^{k+1} matrisi her adımda otomatik olarak artırılır veya azaltılır. Bu yaklaşım heuristik olarak düşünülmüştür. Bunun anlamı, $\alpha^k > 1$ veya $0 < \alpha^k \leq 1$ olabilmektedir. Bu konuda değişik analojiler de yapılabilir. [16]'da gösterildiği gibi başarılı adımda α=0.5, başarısız durumda ise α=10 seçilerek W güncellenmiştir. [16]'daki yöntem bir sonraki kısımda anlatılan yöntem için verilmiştir. Ancak, yukarıda çıkartılan yapıda, iterasyon anında herhangi bir analoji yapılmadan direkt olarak güncellenmiştir. Yapılan denemeler göstermiştir ki, bu yaklaşımla hızlı bir şekilde sonuca gidilebilmektedir. Genelde başlangıçta, W>10 seçilmesi uygundur. Optimal nokta civarında $\lim_{k\to\infty}\alpha^k = 0$ ve bunun sonucunda $\lim_{k\to\infty}W^k = 0$ olacaktır. Bu sonucun gösterilmesi ileride Bölüm 5'te nümerik örnekler üzerinde verilecektir. Buradaki temel yaklaşım, limit durumunda ikinci derece terimlerin birinci derece terimlerden daha hızlı olarak sıfıra yaklaşması durumudur.

Şimdi, bu anlatılanlar göz önüne alınarak İBOK için direkt ölçütün azaltılmasına dayalı

ikinci dereceden gradyan algoritması adımları aşağıda verilmektedir:

i) $u^0(t), W^0$, γ, γ_W 'yı başlangıçta seç. k=0.

ii) $\dot{x} = f(x, u^k)$, $x(t_o)=x_0$ için (t_0,t_f) aralığında $x^k(t)$'yi bul.

iii) A(t), B(t) ve C(t) matrislerini ve v(t), w(t) vektör fonksiyonlarını hesapla.

iv) $\phi_x(t_f)$ ve $\phi_{xx}(t_f)$ değerlerini belirle.

v) $\dot{P}$, $P(t_f) = \phi_{xx}(t_f)$ için, $\dot{q}$, $q(t_f) = \phi_x(t_f)$ için zamanda geriye doğru entegre ederek P^k ve q^k ifadelerini bul.

vi) Kazanç matrisi $K^k(t)$'yi ve sürücü kazanç vektörü $b^k(t)$'yi hesapla.

vii) $\dot{x} = f(x, u^k + \delta u^k) = f(x, u^k + K^k(t)[x - x^k] + b^k(t))$, $x(t_0) = x_0$ ve $x^{k+1}(t)$'yi bul.

viii) $\delta x^k = x^{k+1} - x^k$ ve $\delta u^k = K^k \delta x + b^k$ elde edilerek $u^{k+1}(t) = u^k(t) + \delta u^k(t)$ bulunmuş olur.

ix) $\tilde{\lambda} = P^k \delta x^k + q^k$ ve $\tilde{H}_{\delta u}$ ifadelerini bul.

x) $\|\tilde{H}_{\delta u}\| < \gamma$ ve $W < \gamma_W$ ise dur. $u^*(t)=u^{k+1}(t)$, $x^*(t)=x^{k+1}(t)$, $K^*(t)=K^k(t)$ ve $b^*(t)=b^k(t)$ olarak sakla.

$\|\tilde{H}_{\delta u}\| \geq \gamma$ veya $W \geq \gamma_W$ ise, α^k ve $W^{k+1} = \alpha^k W^k$ 'yı hesaplayarak (ii)'ye dön.

Burada, bu algoritmada vii. adımda hesaplanan $x^{k+1}(t)$ için yapısal bir hesaplama yapılmıştır. Çözüm için geri besleme yapısı düşünülerek çözüm yapılmıştır. Her yörünge depolandığı için bu çözüm oldukça iyi ve hızlıdır. Ancak, depolama maliyeti düşünüldüğünde başka bir yöntem olarak, $\delta x(t_0) = 0$ 'dan başlanarak adım adım çözüm yapmakta mümkündür. Böylece depolama problemi çözülmüş olur. Bu algoritma işlerlik açısından hızlı bir şekilde çalışmaktadır. Nümerik integrasyon çözümlemelerde, 5. dereceden Runga-Kutta-Butcher metodu ve integral almada ise Simpson'nun Composite algoritması kullanılmıştır [99-106]. Şekil 3. 3'te, İBOK için ikinci dereceden direkt ölçütün azaltılması metoduna dayalı LOOK gradyan algoritması akış diyagramı gösterilmektedir. İBOK için sistem blok diyagramı Şekil 3. 2'deki gibidir.

Eğer PÖ kuadratik yapıda ise, bu yöntem için önemli bir sonuç ortaya çıkmaktadır. Bu durumda, (3.62)-(3.64) eşitlikleriyle verilen varsayımlar W=0 için otomatik olarak sağlanmaktadır. Bunu görebilmek için kuadratik PÖ yazılısa,

$$J = \tfrac{1}{2}x^T(t_f)Sx(t_f) + \tfrac{1}{2}\int_{t_0}^{t_f}(x^TQx + u^TRu)dt \qquad (3.67)$$

Burada, $Q \geq 0$, $S \geq 0$ olup pozitif yarı tanımlı ve R>0 olup pozitif tanımlı olarak verildiği düşünülürse, W=0 için (3.62)-(3.64) eşitliklerinin hepsinin sağlandığı görülebilir. Bu ise, konjuge nokta problemini otomatik olarak ortadan kaldırmaktadır. Algoritmada, W'nun hesabına gerek olmadan (W=0) çok hızlı bir şekilde LOOK problemi veya özel olarak lineer olmayan kuadratik optimal kontrol (LOKOK) problemi çözülebilmektedir. Bu sonuç çok önemli bir çıkarımdır.

3.2.2.2. Hamiltonian prensibine dayalı ikinci dereceden gradyan yöntem (İDHY) ile İBOK

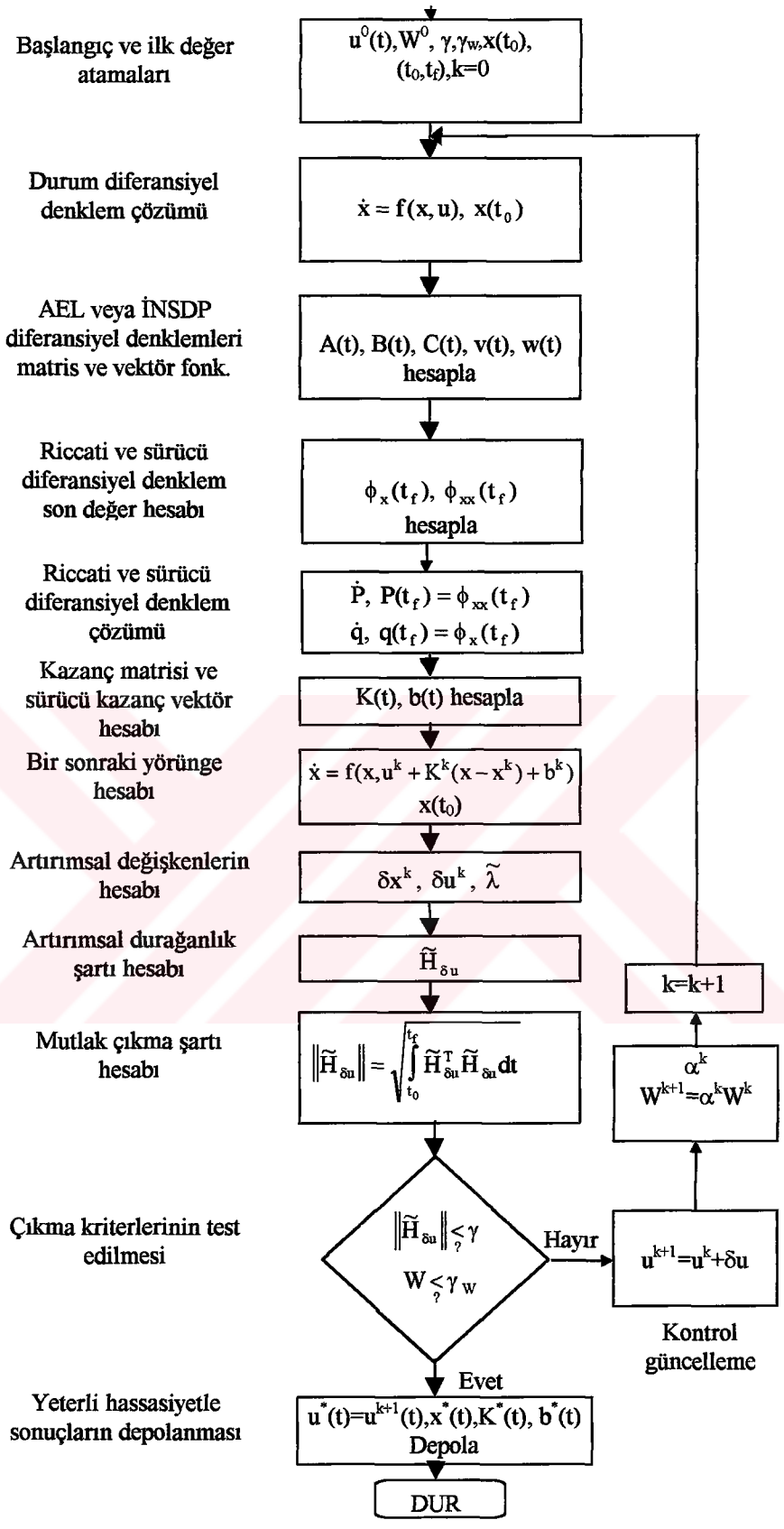
Bu yöntem, temel olarak birinci dereceden yöntemde ele alınan yapıya dayalı olarak geliştirilmiştir. Oradaki gibi, problem önce dinamik sınırlama olmayacak şekilde APÖ oluşturulur. Bu ifade aşağıda verilmiştir:

$$J_a = \phi(x(t_f)) + \int_{t_0}^{t_f}\left\{H(x,u,\lambda) - \lambda^T\dot{x}\right\}dt \qquad (3.68)$$

Buradaki yaklaşım, J_a'yı ikinci dereceden Taylor serisine açarak farkı yazmak ve problemi yerel optimizasyon problemine dönüştürmektir. Dikkat edilirse burada, adjoint değişkeni bulunmakta olup, çözümde ona ait ifadeler aynen birinci dereceden yöntemdeki gibi çıkacaktır. Önceki bölümdeki çözüm gibi problem çözüldüğünde fark ifadesi birinci ve ikinci varyasyondan (y.d.t 'ler dışında) oluşacaktır:

$$\Delta J_a = [\phi_x^T(t_f) - \lambda^T(t)]\delta x + \tfrac{1}{2}\delta x^T(t_f)\phi_{xx}\delta x(t_f) + \int_{t}^{t_f}\left\{H_u^T\delta u + (H_x^T + \dot{\lambda}^T)\delta x\right\}dt$$

$$+\int_{t_0}^{t_f}\left\{(H_\lambda^T - \dot{x}^T) + \delta\lambda^T(H_{\lambda x}\delta x + H_{\lambda u} - \delta\dot{x}) + \tfrac{1}{2}[\delta x^T\ \delta u^T]\begin{bmatrix} H_{xx} & H_{xu} \\ H_{ux} & H_{uu} + W \end{bmatrix}\begin{bmatrix}\delta x \\ \delta u\end{bmatrix}\right\} \qquad (3.69)$$



Şekil 3. 3 İBOK için İDDÖAY ile LOOK gradyan algoritması akış diyagramı

Burada $W \in R^{mxm}$, kontrol sınırlama ağırlık matrisidir. $H_{\lambda x}=f_x$ ve $H_{\lambda u}=f_u$ olduğu kolayca görülebilir. Ayrıca, (3.69) ifadesi birinci varyasyonu ve (3.10) ile (3.11) eşitlileriyle verilen gerekli koşulları verir. (3.12) eşitliği ancak optimal noktada sağlanacağından sıfır değildir. Yerel model ise (3.69)'dan aşağıdaki gibidir:

$$\delta\dot{x} = f_x\delta x + f_u\delta u \quad , \quad \delta x(t_0) = 0 \tag{3.70}$$

Varyasyonel problemin çözümünde geriye kalan fark ifadesi aşağıda verilmiştir:

$$\Delta J_a = \tfrac{1}{2}\delta x^T(t_f)\phi_{xx}\delta x(t_f) + \int_{t_0}^{t_f}\left\{H_u^T\delta u + \tfrac{1}{2}[\delta x^T\ \delta u^T]\begin{bmatrix} H_{xx} & H_{xu} \\ H_{ux} & H_{uu}+W \end{bmatrix}\begin{bmatrix}\delta x \\ \delta u\end{bmatrix}\right\} \tag{3.71}$$

Problem bir önceki bölümdeki gibi, yerel optimizasyon problemi oluşmuştur:

$$\min_{\delta u}\ \Delta J_a(\delta u) \tag{3.72}$$

sınırlama $f_x\delta x + f_u\delta u - \delta\dot{x} = 0 \quad , \quad \delta x(t_0) = 0$ (3.73)

Önceki bölümdeki gibi çözüm yapıldığında ve gerekli dönüşümler yapıldığında,

$$\delta\dot{x} = A(t)\delta x + B(t)\delta\lambda + v(t) \quad , \quad \delta x(t_0) = 0 \tag{3.74}$$

$$\delta\dot{\lambda} = -C(t)\delta x - A^T(t)\delta\lambda - w(t) \quad , \quad \lambda(t_f) = \phi_{xx}\delta x(t_f) \tag{3.75}$$

olarak, İNSDP halini alır. Burada, katsayılar aşağıda verildiği gibidir:

$$A(t)\big|_{n\times n} = f_x - f_u(H_{uu}+W)^{-1}H_{ux} \tag{3.76}$$

$$B(t)\big|_{n\times n} = -f_u(H_{uu}+W)^{-1}f_u^T \tag{3.77}$$

$$C(t)\big|_{n\times n} = H_{xx} - H_{xu}(H_{uu} + W)^{-1}H_{ux} \tag{3.78}$$

$$v(t)\big|_{n\times 1} = -f_u(H_{uu} + W)^{-1}H_u \tag{3.79}$$

$$w(t)\big|_{n\times 1} = -H_{xu}(H_{uu} + W)^{-1}H_u \tag{3.80}$$

Çözüm, genel Riccati dönüşümü yardımıyla yapılır:

$$\delta\lambda = P\delta x + q \tag{3.81}$$

$$\dot{P} = -(A^T P + PA + PBP - C) \quad , \quad P(t_f) = \phi_{xx}(t_f) \tag{3.82}$$

$$\dot{q} = -(A^T q + PBq + Pv + w) \quad , \quad q(t_f) = 0 \tag{3.83}$$

Görüldüğü gibi, sürücü vektör fonksiyonu için son değer sıfır olmaktadır. Diğer yöntemde farklı olduğuna dikkat edilmelidir. Sonuçta, artırımsal kontrol fonksiyonu benzer şekilde aşağıdaki gibidir:

$$\delta u(t) = K(t)\delta x + b(t) \tag{3.84}$$

$$K(t) = -(H_{uu} + W)^{-1}(H_{ux} + f_u^T P) \tag{3.85}$$

$$b(t) = -(H_{uu} + W)^{-1}(H_u + f_u^T q) \tag{3.86}$$

Burada, $K(t) \in R^{mxn}$ olup kazanç matrisi, $b(t) \in R^m$ olup sürücü kazanç vektör fonksiyonlarıdır. Tüm ifadelerin çıkarımı Ek B'de verilmiştir. Burada görüldüğü gibi adjoint değişkeni hesabı gerekmektedir. Bu yüzden diğer yönteme göre, **n** adet fazla diferansiyel denklem çözmek gerekmektedir. Bu yöntemde, **n** adet zamanda ileri, **2n+n(n+1)/2** adet zamanda geriye doğru olmak üzere toplam **3n+n(n+1)/2** adet diferansiyel denklem çözmek gerekmektedir.

Bu yöntemde de İBOK algoritması için,

$$u^{k+1}(t) = u^k(t) + \delta u^k(t) \tag{3.87}$$

yazılabilir. Burada yapılan varsayımlar diğer yönteme benzer:

$$\phi_{xx} \geq 0 \quad , \quad \text{pozitif yarı - tanımlı} \tag{3.88}$$

$$(H_{uu}+W)>0 \quad , \quad \text{pozitif tanımlı} \tag{3.89}$$

$$H_{xx} - H_{xu}(H_{uu}+W)^{-1}H_{ux} \geq 0 \quad , \quad \text{pozitif yarı - tanımlı} \tag{3.90}$$

Bu şartlar sağlandığında (t_0,t_f) aralığında konjuge nokta problemiyle karşılaşılmayacaktır. W'nun eklenmesi zaten bu şartların sağlanmasına da yöneliktir. Burada, W'nun belirlenmesi, İDDÖAY' den biraz farklı olarak aşağıdaki yapıdadır:

$$W^{k+1} = \alpha^k W^k \tag{3.91}$$

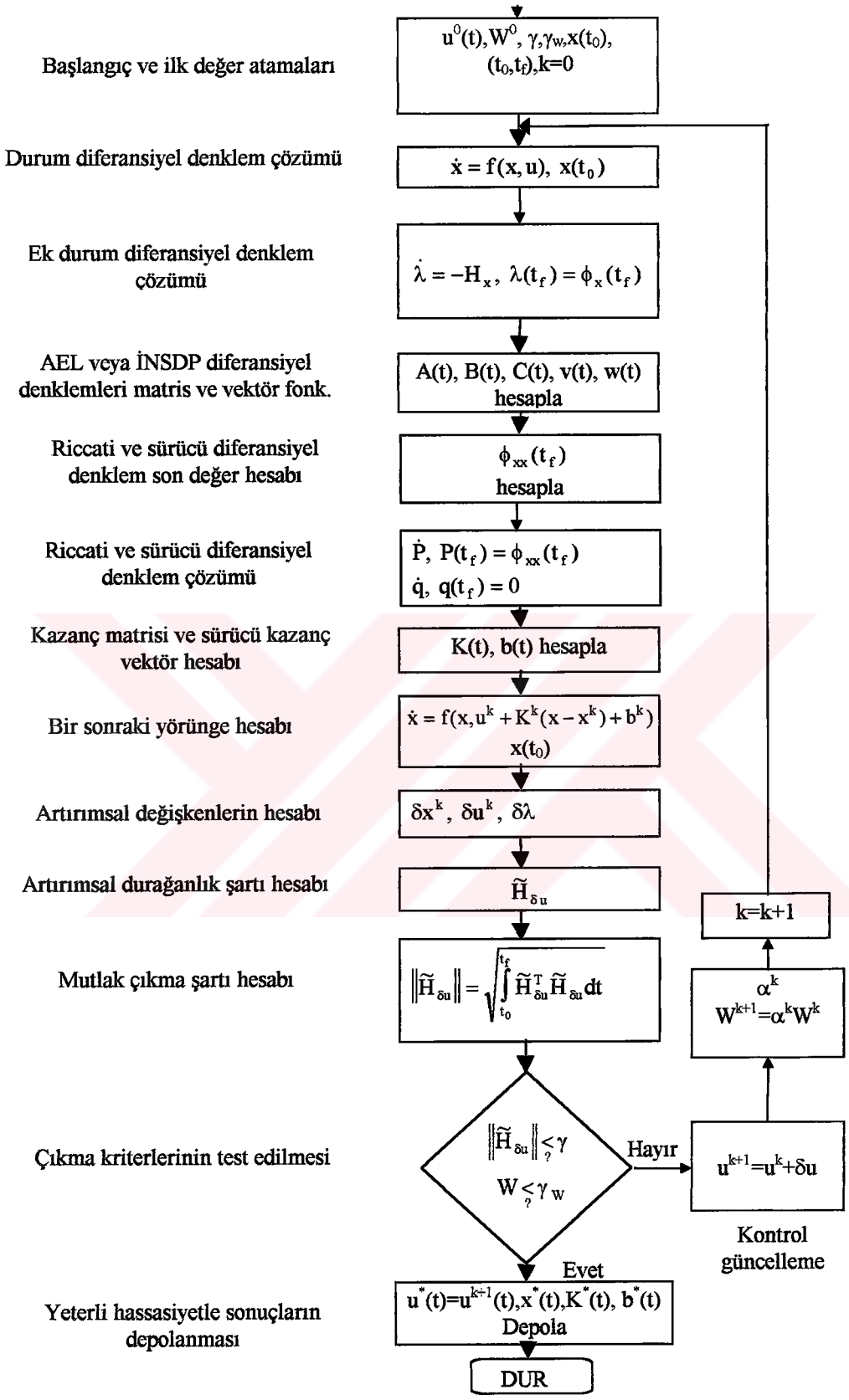
$$\alpha^k = \frac{\Delta J^k}{J^{k-1}} \left| \frac{\int_{t_0}^{t_f} H_u^T \delta u dt}{\frac{1}{2}\delta x^T(t_f)\phi_{xx}\delta x(t_f) + \int_{t_0}^{t_f}\left\{\frac{1}{2}\begin{bmatrix}\delta x^T & \delta u^T\end{bmatrix}\begin{bmatrix}H_{xx} & H_{xu}\\ H_{ux} & (H_{uu}+W)\end{bmatrix}\begin{bmatrix}\delta x\\ \delta u\end{bmatrix}\right\}dt} \right| \tag{3.92}$$

Burada, $\Delta J^k = J^{k-1} - J^k$'dır ve $\alpha^k > 0$ olarak değişik değerler alabilmektedir. Limit durumunda lineer terim sıfırlanacağından ve bağıl fark ifadesi küçüleceğinden $\lim_{k\to\infty}\alpha^k = 0$ olacaktır. [16]'da α için, başarılı adımda α=0.5, başarısız adımda α=10 alınarak güncelleme yapılmıştır. [8,19,46]'da ise değişik bir yaklaşım yapılarak W ağırlık matrisi yerine, artırımsal kontrol fonksiyonunun (δu) küçük bir kısmı alınarak çözüm yapılmıştır. Ancak, bu yaklaşım konjuge nokta açısından bir çözüm getirmemektedir. Yani, eğer başlangıçta veya herhangi bir iterasyon anında konjuge noktaya takılma durumu varsa buradan kurtulmak bu yaklaşımla mümkün değildir.

Sadece, başka bir noktadan başlanarak konjuge noktadan geçmeyip sonuca ulaşmak mümkündür. Bu yüzden, geliştirilen yöntem bu yöntemlere göre oldukça etkili bir çözümdür. Bu yöntemde genelde $W>10^2$ seçilmesi uygundur.

İDDÖAY'nde bahsedilen, LOKOK problemi için (3.88)-(3.90) eşitlikleriyle verilen varsayımların sağlanması garanti edilemez. Ancak yeteri kadar büyük W için ($W>>0$) sağlanması mümkün olabilir. Bu ise, diğer yöntemin üstünlüğünü göstermektedir. Şekil 3. 4'te Hamiltonian prensibine dayalı ikinci dereceden gradyan yöntemi (İDHY) ile İBOK akış diyagramı verilmektedir. Algoritma adımları İDDÖAY'deki gibidir.

Akış diyagramından da görüldüğü gibi, bu yöntemde her adımda adjoint durum denklemi çözümü gerekmektedir.



Şekil 3. 4 İBOK için İDHY ile LOOK gradyan algoritması akış diyagramı

BÖLÜM 4. OPTİMAL GERİ BESLEME YAPISINI BELİRLEME

Burada, geri besleme (GB) kanunu, LOOK probleminin çözümünden yararlanılarak bulunacaktır [1,6,8,16,17,46,47,109]. Lineer kuadratik optimal kontrol (LKOK) problemi [1,6,7,8,33,35,109] için çıkartılmış olan geri besleme yapıları gibi, LOOK problemi yardımıyla zamanla-değişen-optimal-geri-besleme (ZDOGB) ifadeleri elde edilebilmektedir. Bu problem için Bölüm 3'te elde edilen sonuçlar kullanılacaktır. Başka herhangi bir ek işlem ve hesaba gerek yoktur. Yani, ikinci dereceden gradyan yöntem sonucunda elde edilen optimal kontrol ve yörünge fonksiyonları yanında çıkartılan kazanç matris ve vektörü, istenilen yörünge için ZDOGB kanununu vermektedir. Bu sonuç gerçekten çok etkili ve önemli bir sonuçtur. Kullanılan iki ikinci dereceden yöntem için de bu söylenenler geçerlidir.

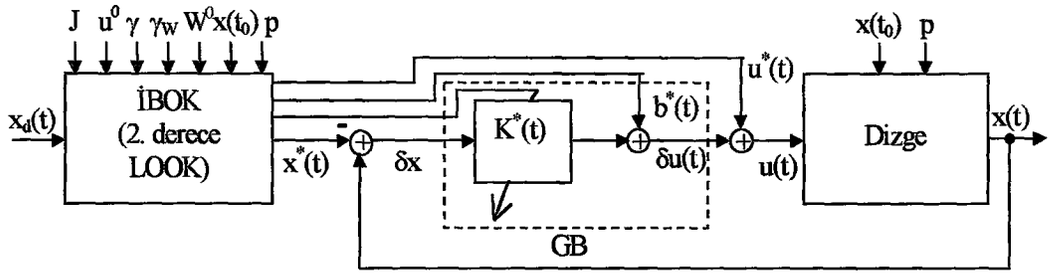
4.1. Geri Besleme ve Yapısal Diyagramın Oluşturulması

Bölüm 3'te elde edilen ve iteratif olarak çözüm yönteminde kullanılan artırımsal kontrol fonksiyonu ifadesi,

$$\delta u(t) = K(t)\delta x + b(t) \tag{4.1}$$

şeklinde idi. Bu yapı, durum GB'si ve sürücü ifadeden meydana gelmektedir. Bu ifade, optimal noktada bulunmuş olan $K^*(t)$ ve $b^*(t)$'ye karşılık $u^*(t)$ ve $x^*(t)$ yörüngelerinin de belirlendiğini gösterir. Bu yörünge civarında sistem tutulmaya çalışılmak istenmektedir. Bu yörüngeden sapma durumda, kazanç ifadeleri devreye girerek sistemi istenilen yörüngeye çekmeye çalışır. Başlangıç durumu bozması, sistem parametrelerinin herhangi bir nedenle değişmesi, herhangi bir nedenle sisteme eklenen gürültü (basamak gürültüsü) gibi bozucu etkilere karşı, LOOK yardımıyla elde edilen (4.1) eşitliği ile verilen GB yapısı oldukça iyi kompanze edici görev görebilmektedir.

Şekil 4. 1'de LOOK probleminin ikinci dereceden yöntemler ile çözülmesi sonucunda elde edilen ZDOGB kanununun sisteme uygulanması gösterilmektedir.



Şekil 4. 1 ZDOGB Yapısı

Şekil 4.1'de verilen geri beslemeli sistem daima kararlıdır. Çözümleme yönteminin sağladığı bu özellik kararsız sistemlere de uygulanabilir [6,8,33,46].

Bir başka anlatımla, ikinci dereceden yöntem ile optimal kontrol problemi çözülürken, aynı zamanda lineer optimal kontrol problemi de çözülmektedir. Yani, herhangi bir adımda bulunan yörüngeler civarında problem lineer kuadratik (LK) problemine dönüştürülebilir ve o noktada geri-besleme (GB) kanunu oluşturulabilir. Buradan yola çıkarak, optimal noktada problem aynı şekilde lineer kuadratik (LK) problemi olarak biçimlendirilerek geri-besleme (GB) yapısı oluşturulur. Bu yapıda (ikinci derece yöntem), lineer kuadratik (LK) probleminin tasarım parametreleri (tasarım matrisleri) otomatik olarak lineer olmayan optimal kontrol (LOOK)'den çıkarılmaktadır. Bu özellik çok önemlidir.

Birinci dereceden yöntem kullanarak LOOK problemi çözüldüğü durumda, GB yapısı oluşturulabilmesi için problem çözülmeli, yani optimal yörüngeler bulunmalıdır. Bulunan bu yörüngeler civarında oluşturulacak yerel model yardımıyla GB kanunu oluşturulabilir. Burada, bunu gerçekleştirmek için (3.71) eşitliği göz önüne alınarak çözüm bulunacaktır. Optimum noktada, $H_u=0$ ve $W=0$ olur. Bu durumda geriye kalan ifade yazılırsa;

$$\Delta J_a = \tfrac{1}{2}\delta x^T(t_f)\phi_{xx}\delta x(t_f) + \int_{t_0}^{t_f}\left(\tfrac{1}{2}\delta x^T\phi_{xx}\delta x + \tfrac{1}{2}\delta u^T H_{uu}\delta u + \delta x^T H_{xu}\delta u\right)dt \qquad (4.2)$$

elde edilir. Bu yapı, standart lineer kuadratik formda (artırımsal değişkenli) olup benzerlik kurulursa ve lineer dinamik eşitlik sınırlaması ile birlikte yazılırsa aşağıdaki genel problem tanımı elde edilir;

$$\min_u J = \tfrac{1}{2}x^T(t_f)S_f x(t_f) + \int_{t_0}^{t_f}\left(\tfrac{1}{2}x^T Q x + \tfrac{1}{2}u^T R u + x^T N u\right)dt \qquad (4.3)$$

sınırlama. $\dot{x} = A(t)x(t) + B(t)u(t), \qquad x(t_0) = x_0$ (4.4)

Burada, $Q \in R^{n\times n} \geq 0$, $R \in R^{m\times m} > 0$, $S_f \in R^{n\times n} \geq 0$, $N \in R^{n\times m}$, $A(t) \in R^{n\times n}$ ve $B(t) \in R^{n\times m}$ olarak belirlenir. Bu problemin çözümü;

$$u(t)=K(t)u(t) \qquad (4.5)$$

$$K(t)=-R^{-1}(N^T+B^T S) \qquad (4.6)$$

olarak durum geri-beslemesi yapısında bulunur. Burada, S(t) Riccati matrisi (simetrik) olup, ona ait olan ilgili diferansiyel denklem sistemi, Riccati dönüşümü yardımıyla kolayca aşağıdaki yapıda bulunarak LK probleminin çözümü tamamlanır;

$$\dot{S} = -\left[(A^T - NR^{-1}B^T)S + S(A - BR^{-1}N^T) - SBR^{-1}B^T S + (Q - NR^{-1}N^T)\right], \quad S(t_f) = S_f \qquad (4.7)$$

Burada, yukarıda verilen şartlara ek olarak, $(Q - NR^{-1}N^T) \geq 0$ olması sağlanır. Buradan yola çıkarak, optimal pertürbasyon geri-besleme probleminin çözümü, yukarıda verilen genel lineer-kuadratik kontrolcüsündeki ağırlık matrislerini ve sistem matrislerini seçerek aşağıdaki gibi elde edilir [6,8,109];

$$Q = H_{xx}\Big|_{(x^*,u^*)},\ R = H_{uu}\Big|_{(x^*,u^*)},\ S_f = \phi_{xx}\Big|_{(x^*(t_f),u^*(t_f))},\ N = H_{xu}\Big|_{(x^*,u^*)} \tag{4.8}$$

$$A(t) = \frac{\partial f}{\partial x}\Big|_{(x^*,u^*)},\ B(t) = \frac{\partial f}{\partial u}\Big|_{(x^*,u^*)} \tag{4.9}$$

Bulunan bu matrisler, ikinci derece yöntemi kullanıldığında otomatik olarak her iterasyon adımında zaten bulunmaktadır. Birinci derece yöntem kullanıldığında ise, problem çözüldükten sonra, matrisler yukarıda verildiği gibi ayrıca hesaplanarak bulunması gerekmektedir.

BÖLÜM 5. LİNEER OLMAYAN OPTİMAL KONTROL ve GERİ BESLEME İÇİN BENZETİM ÇALIŞMALARI

Bu tez çalışmasında incelenen ve geliştirilen teorik çalışmalar, test problemi olarak ta literatürde kullanılan Van-der-Pol (VDP) problemi [1,8,16,17,46] ve kimyasal endüstride çok geniş uygulama olanı olup üzerinde bir çok çalışma yapılan oldukça lineer olamayan bir sistem olan sürekli-karıştırmalı-tank-reaktör (CSTR) [8,97,98,109,110-112] üzerinde uygulanmıştır. Bu problemler bir çok araştırmacının kullandığı önemli sistemlerdir. VDP problemi, konjuge nokta probleminin meydana geldiği ve ayrıca lineer olmama özelliğinin uygulandığı sistem olarak, CSTR ise, lineer olmayan yapılı ve özel çatallanma noktalarına sahip özellikleri olan sistem olarak teorik sonuçların gerçeklenmesi amacıyla seçilerek kullanılmıştır. BDGY, İDDÖAY ve İDHY, İBOK probleminin çözümünde değişik yörüngeler için denenmiştir. Ayrıca, optimal geri besleme (OGB) kanunun oluşturulması ve çeşitli bozucu etkilerin GB yardımıyla kompanze edilmesi gösterilmiştir.

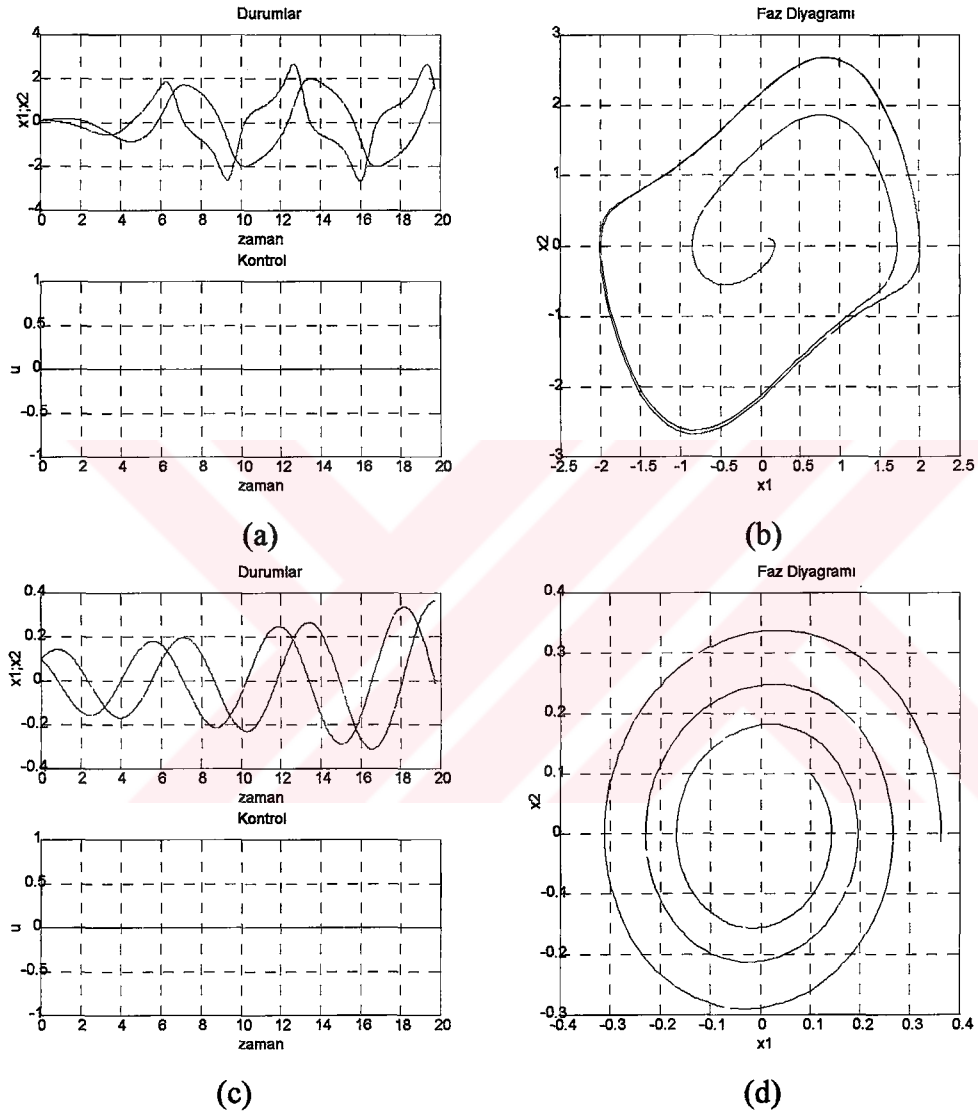
5.1. Van-Der-Pol Osilatör Sistemi

Bu problem, bir parametreli lineer olmayan yapıya sahip ikinci dereceden dinamik bir sistemdir. Durum denklemi şu şekildedir:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -x_1 + a(1-x_1^2)x_2 + u \end{aligned} \qquad (5.1)$$

Burada, sürücü fonksiyon u(t) kontrol fonksiyonu olarak kullanılacaktır. a, sistem parametresidir ve çözümün lineersizlik davranışını belirler. Bu davranışları görmek için bazı sonuçlar verilecektir. $x=[x_1\ x_2]^T$ olmak üzere $x(0)=[0.1\ 0.1]^T$ alınarak (i) a=1.0, ve (ii) a=0.1 durumları için ve kontrol fonksiyonu u(t)=0.0 için benzetim sonuçları

incelenmiştir. Burada, t_0=0 ve t_f=20 sn olarak alınmıştır. Şekil 5. 1 bu durumlar için sistem çıkışlarını ve faz düzlemindeki (x_1-x_2) değişimleri göstermektedir. Sonuçlardan da görüldüğü gibi, osilatör ve sınırlı saykıl durumları bu sistemin kararsız davranışlara sahip olduğunu göstermektedir.



Şekil 5. 1 VDP osilatör problemi için çeşitli parametrelere göre kontrol, durum ve faz-düzlemi gösterimleri. (a),(b) a=1.0, (c),(d) a=0.1

5.1.1. Optimal kontrol yöntemlerinin VDP sistemine uygulanması

Burada, Bölüm 3'te anlatılan çözümleme yöntemleri VDP problemi üzerinde incelenecektir. PÖ olarak kuadratik yapı seçilmiştir. Genel yapısı aşağıdaki şekildedir:

$$J = \tfrac{1}{2}[x(t_f) - x^d(t_f)]^T S[x(t_f) - x^d(t_f)] + \tfrac{1}{2}\int_{t_0}^{t_f} \{[x(t) - x^d(t)]^T Q[x(t) - x^d(t)] + u^T(t)Ru(t)\}dt \quad (5.2)$$

Burada, $Q \geq 0$ ve $S \geq 0$ olup pozitif-yarı tanımlı simetrik ağırlık matrisleri, R>0 olup pozitif-tanımlı simetrik ağırlık matrisleridir. S matrisi, x(t) yörüngesinin $t=t_f$'te istenen $x^d(t_f)$ noktasına ulaştırılmasında etkindir. Q matrisi, kontrol fonksiyonu vasıtasıyla x(t) yörüngesinin istenen $x^d(t)$ yörüngesine uzaklığını etkileyen parametredir. R matrisinin elemanları ise, uygulanan kontrol fonksiyonunu cezalandıran yani, kontrol enerjisini sınırlandıran parametre matrisidir. Q,R,S matrislerinin en uygun bir şekilde seçimi, kullanılacak kontrol fonksiyonunun miktarına ve hassasiyet gibi gereksinimlere dayalı olduğundan büyük oranda mühendislik karar verme problemi olarak değerlendirilebilir.

İlk örnek olarak, $t_0=0$, $t_f=5$, $Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $R = 1$, $S = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$, $x_d(t) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $x(0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ alınarak,

$$J = \tfrac{1}{2}\int_0^5 (x_1^2 + x_2^2 + u^2)dt = \int_0^5 L(x,u)dt \quad (5.3)$$

olur. Ayrıca burada VDP sistem parametresi a=1.0 alınmıştır.

i) Şimdi bu problemi BDGY ile çözümü için Bölüm 3'te çıkartılan ifadeler kullanılırsa,

$$H(x,u,\lambda) = \tfrac{1}{2}(x_1^2 + x_2^2 + u^2) + \lambda_1 x_2 + \lambda_2[-x_1 + a(1 - x_1^2)x_2 + u] \quad (5.4)$$

Gerekli koşullar için,

$$\dot{x} = H_{\lambda} = \begin{bmatrix} H_{\lambda_1} \\ H_{\lambda_2} \end{bmatrix} = \begin{bmatrix} x_2 \\ -x_1 + a(1-x_1^2)x_2 + u \end{bmatrix}, \quad x(0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{5. 5}$$

$$\dot{\lambda} = -H_x = -\begin{bmatrix} H_{x_1} \\ H_{x_2} \end{bmatrix} = \begin{bmatrix} (1+2ax_1x_2)\lambda_2 - x_1 \\ -\lambda_1 + a(x_1^2 - 1)\lambda_2 - x_2 \end{bmatrix}, \quad \lambda(5) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{5. 6}$$

ve optimallik şartı,

$$0 = H_u = u + \lambda_2 \tag{5. 7}$$

olarak bulunur. (5.7) eşitliği ancak optimal $u^*(t)$ için geçerli olacaktır. Adım büyüklüğü $\tau = 0.02$ ve $\gamma = 0.1$ alınarak Şekil 3.1'de verilen birinci dereceden gradyan algoritması uygulanmıştır. Başlangıç kontrol fonksiyonu $u^0(t)=0.0$ alınarak iterasyona başlanmıştır. Şekil 5. 2'de sistemin başlangıç durumu gösterilmektedir. Şekil 5. 3'te ise başlangıçtaki adjoint durum değişkeni, Hamiltonian ve durağanlık şartının (H_u) değişimleri gösterilmektedir. Bu yöntem, optimal noktaya yaklaştığında yavaşlamaktadır. Bu durum PÖ'nün iterasyonla değişimini gösteren Şekil 5. 4'te açıkça görülmektedir. Şekil 5. 5'te ise kontrol fonksiyonunun iterasyonla değişimi gösterilmektedir. Buradan da görülüyor ki, optimal nokta civarına yaklaştıkça kontrol artımı küçük olmaktadır. Şekil 5. 6 ve Şekil 5. 7, optimal nokta civarındaki algoritma değişkenleri, kontrol ve durum değişkenlerini göstermektedir.

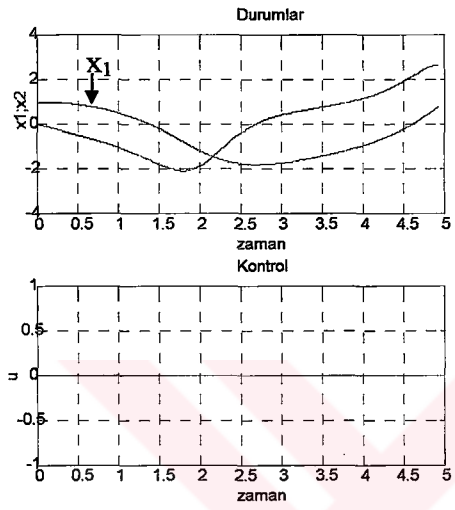
ii) İkinci dereceden direkt ölçütün azaltılması yöntemi (İDDÖAY) ile aynı problemin çözümü aşağıda verilmiştir.

Matris Riccati, (P) ve vektör sürücü (q) diferansiyel denklem için gerekli ve (3.43)-(3.47) eşitlikleriyle verilen matris ve vektör ifadeleri:

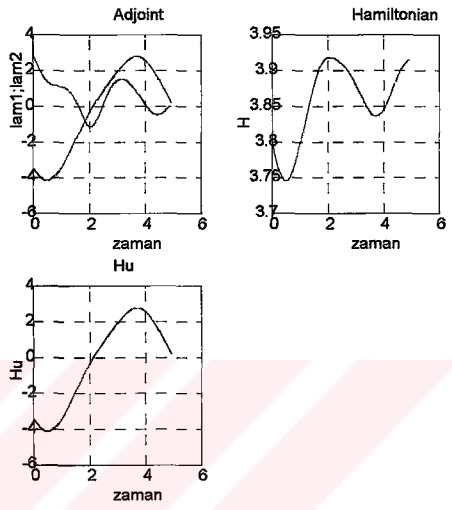
$$f_x = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -(1+2ax_1x_2) & a(1-x_1^2) \end{bmatrix}, \quad f_u = \begin{bmatrix} \frac{\partial f_1}{\partial u} \\ \frac{\partial f_2}{\partial u} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$L_u = \frac{\partial L}{\partial u} = u(t), \quad L_{uu} = \frac{\partial^2 L}{\partial u^2} = 1, \quad L_{ux} = \frac{\partial}{\partial x}\left(\frac{\partial L}{\partial u}\right) = \begin{bmatrix} 0 & 0 \end{bmatrix}, \quad L_{xu} = \frac{\partial}{\partial u}\left(\frac{\partial L}{\partial x}\right) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

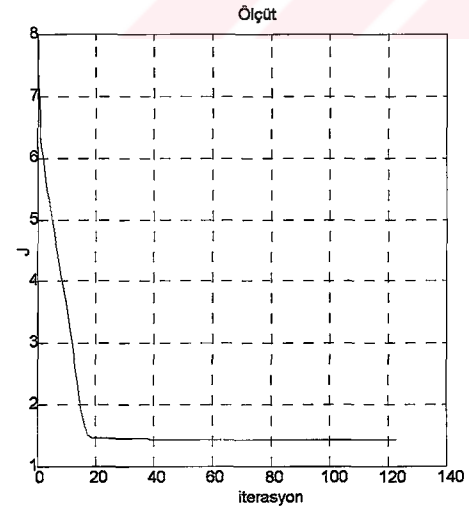
$$L_x = \begin{bmatrix} \frac{\partial L}{\partial x_1} \\ \frac{\partial L}{\partial x_2} \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = x(t), \quad L_{xx} = \begin{bmatrix} \frac{\partial^2 L}{\partial x_1^2} & \frac{\partial^2 L}{\partial x_1 x_2} \\ \frac{\partial^2 L}{\partial x_2 x_1} & \frac{\partial^2 L}{\partial x_2^2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$



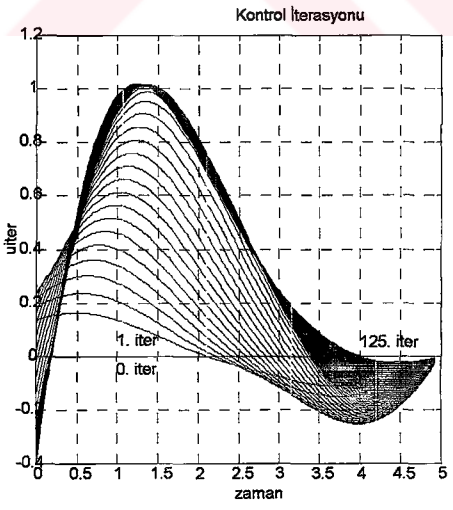
Şekil 5. 2 $u^0(t)$=0.0 için VDP sisteminin cevabı



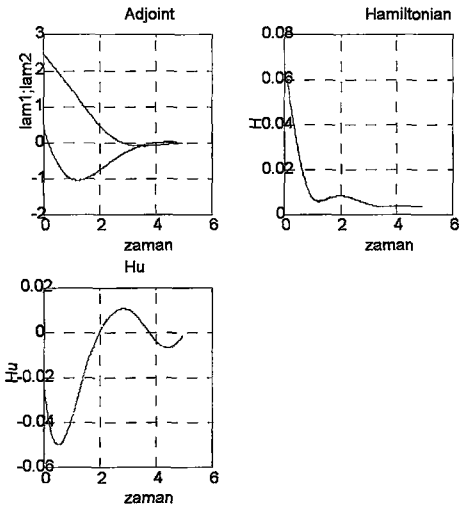
Şekil 5. 3 BDGY için adjoint, Hamiltonian ve H_u'nun başlangıç adımındaki değişimleri



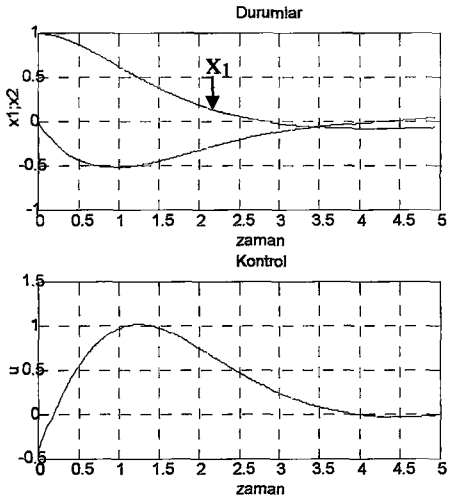
Şekil 5. 4 BDGY için PÖ'nün iterasyonla değişimi



Şekil 5. 5 BDGY için kontrol fonksiyonunun iterasyonla değişimi



Şekil 5. 6 BDGY sonunda optimal noktadaki adjoint, Hamiltonian ve H_u'nun değişimleri



Şekil 5. 7 BDGY sonunda bulunan optimal kontrol ve yörüngelerin değişimleri

$$P(5) = \phi_{xx}(5) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad q(5) = \phi_x(5) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$A(t) = \begin{bmatrix} 0 & 1 \\ -(1+2ax_1x_2) & a(1-x_1^2) \end{bmatrix}, \quad B(t) = (1+W)^{-1}\begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix}, \quad C(t) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$v(t) = (1+W)^{-1}\begin{bmatrix} 0 \\ -u(t) \end{bmatrix}, \quad w(t) = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Bu ifadeler, (3.54) ve (3.55) eşitliklerinde yerlerine yazılırsa, Riccati ve sürücü diferansiyel denklemler sırayla,

$$\dot{P}_{11} = 2(1+2ax_1x_2)P_{12} + (1+W)^{-1}P_{12}^2 - 1, \qquad P_{11}(5) = 0$$

$$\dot{P}_{12} = -P_{11} + a(x_1^2 - 1)P_{12} + (1+2ax_1x_2)P_{22} + (1+W)^{-1}P_{12}P_{22}, \qquad P_{12}(5) = 0$$

$$\dot{P}_{22} = -2P_{12} + 2a(x_1^2 - 1)P_{22} + (1+W)^{-1}P_{22}^2 - 1, \qquad P_{22}(5) = 0$$

$$\dot{q}_1 = [(1+2ax_1x_2) + (1+W)^{-1}P_{12}]q_2 + (1+W)^{-1}P_{12}u - x_1, \qquad q_1(5) = 0$$

$$\dot{q}_2 = -q_1 + [a(x_1^2 - 1) + (1+W)^{-1}P_{22}]q_2 + (1+W)^{-1}P_{22}u - x_2, \quad q_2(5) = 0$$

olarak belirlenir. Kazanç matrisi K(t) ve sürücü kazanç b(t) aşağıdaki gibi bulunur:

$$K(t) = -(1+W)^{-1}\begin{bmatrix} P_{12} & P_{22} \end{bmatrix} = \begin{bmatrix} K_{11}(t) & K_{12}(t) \end{bmatrix}$$

$$b(t) = -(1+W)^{-1}(u + q_2)$$

Artırımsal kontrol, adjoint ve Hamiltonian aşağıdaki gibidir:

$$\delta u(t) = K(t)\delta x + b(t) = -(1+W)^{-1}(P_{12}\delta x_1 + P_{22}\delta x_2 + u + q_2)$$

$$\tilde{\lambda} = \begin{bmatrix} \tilde{\lambda}_1(t) \\ \tilde{\lambda}_2(t) \end{bmatrix} = \begin{bmatrix} P_{11}\delta x_1 + P_{12}\delta x_2 + q_1 \\ P_{12}\delta x_1 + P_{22}\delta x_2 + q_2 \end{bmatrix}$$

$$\tilde{H} = \tilde{L} + \tilde{\lambda}_1\delta x_2 - \tilde{\lambda}_2(1 + 2ax_1x_2)\delta x_1 + \tilde{\lambda}_2 a(1 - x_1^2)\delta x_2 + \tilde{\lambda}_2\delta u$$

Burada,

$\tilde{L} = x_1\delta x_1 + x_2\delta x_2 + u\delta u + \frac{1}{2}[\delta x_1^2 + \delta x_2^2 + (1+W)\delta u^2]'$dir.

Durağanlık şartı ise,

$\tilde{H}_{\delta u} = u + (1+W)^{-1}\delta u + \tilde{\lambda}_2$ olarak bulunur.

Bir sonraki iterasyon için kontrol fonksiyonu,

$$u^{k+1}(t) = u^k(t) + \delta u^k(t) = u^k(t) + K^k(t)\delta x^k + b^k(t) = u^k + K_{11}^k\delta x_1^k + K_{12}^k\delta x_2^k + b^k$$

$$\delta x^k = x^{k+1} - x^k$$

olarak hesaplanır. Kontrol sınırlama ağırlık matrisi güncelleme katsayısı,

$$\alpha = \left| \frac{\int_0^5 \frac{1}{2}(\delta x_1^2 + \delta x_2^2 + \delta u^2)dt}{\int_0^5 (x_1 \delta x_1 + x_2 \delta x_2 + u\delta u)dt} \right| \text{ olup,}$$

$W^{k+1} = \alpha^k W^k$ olarak güncellenir.

Bir adım sonra bulunacak yörüngenin hesabı için, Bölüm 3'te anlatıldığı gibi geri-besleme formunda çözüm yapılmıştır. Yani,

$$\dot{x} = f(x, u^k + K^k(x - x^k) + b^k), \quad x(0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \text{ olarak}$$

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -x_1 + a(1 - x_1^2)x_2 + u^k + K_{11}^k(x_1 - x_1^k) + K_{12}^k(x_2 - x_2^k) + b^k \end{bmatrix} \text{ ifadesi}$$

çözülerek $x^{k+1}(t)$ durum yörüngesi elde edilir ki buradan δx^k ve δu^k ifadeleri kolayca bulunabilir.

Algoritmanın işleyiş şekli Şekil 3.3'te verilmişti. Burada, $u^0(t)=0.0$, $\gamma = 5.10^{-3}$ ve $\gamma_W = 5.10^{-2}$ olarak seçilmiştir. Kontrol sınırlama ağırlık matrisi ise $W^0=100$ olarak alınmıştır. Sistemin başlangıç durum ve kontrol değişkenleri için Şekil 5. 2'de verildiği gibidir. Şekil 5. 8, Şekil 5. 9 ve Şekil 5. 10 sırayla başlangıçtaki artırılmış Hamiltonian ve gradyanını, matris Riccati ve sürücü vektörleri, kazanç matrisi ve sürücü kazanç vektörlerini göstermektedir. Şekillerden de görüldüğü gibi, birinci iterasyon adımı için büyük bir değişim olmuştur. Riccati çözümlemesi için de yakınsak bir çözüm bulunmuştur. Yöntem herhangi bir şekilde konjuge noktaya takılmamıştır. Bu yöntemin çok hızlı bir şekilde optimal noktaya doğru yakınsadığı Şekil 5. 11 'deki PÖ'nün değişiminden görülmektedir. Özellikle optimal noktaya çok hızlı bir şekilde ulaşmıştır. Kontrol sınırlama ağırlık parametresi W ve katsayısı α optimal nokta civarında sıfira doğru $(\alpha \to 0, W \to 0)$ gittiği Şekil 5. 12'de görülmektedir. Zaten teorik olarak optimal noktada W=0 olmalıdır. Optimal noktadaki tüm durumlar ise Şekil 5. 13, Şekil 5. 14, Şekil 5. 15 ve Şekil 5. 16'da

verilmiştir. Şekil 5. 17'de ise optimal yörünge ve kontrol değişimleri verilmiştir. BDGY ile elde edilen optimal kontrol fonksiyonu (Şekil 5. 7) ve İDDÖAY ile elde edilen optimal kontrol fonksiyonu (Şekil 5. 17) karşılaştırıldığında, biraz farklılık görülecektir. Bu fark BDGY için biraz daha iterasyon yapılarak giderilebilir.

Şimdi, bu problem için (3.62)-(3.64) eşitlikleriyle verilen koşulların sağlanıp sağlanmadığına bakılırsa,

$\phi_{xx} = 0 \geq 0$ olup pozitif-yarı tanımlı, $L_{uu}=1>0$ olup pozitif tanımlı ve $L_{xx} - L_{xu}(L_{uu})^{-1}L_{ux} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \geq 0$ olup pozitif yarı tanımlı olarak W=0 için dahi tüm koşullar sağlanmıştır. O halde, konjuge nokta problemi ortadan otomatik olarak kalkmış olmaktadır. Bu durum altında, $W^0=0$ için İDDÖAY işletildiğinde diğerine göre ($W^0=100$) daha hızlı bir şekilde ölçüt minimum yapılamaktadır. Bu durum Şekil 5. 18'de PÖ'nün değişimden görülmektedir. Burada sadece hızlı bir şekilde sonuca ulaşılmış olmakla birlikte optimal noktadaki tüm değişkenler $W^0=100$ için bulunan değişimlerle aynıdır. Bu sonuç, Bölüm 3'te anlatılmıştı. Şekil 5. 19'da ise kontrol fonksiyonunun her adımda nasıl değiştiği gösteriliyor. Şekil 5. 13 ile de karşılaştırıldığında, daha büyük adımlarla ilerlediği açıkça görülebilir.

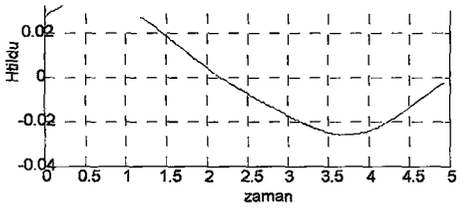
iii) Aynı problemin İDHY ile çözümü aşağıda verilmektedir.
Önce, BDGY'nde verilen Hamiltonian ve adjoint ifadeleri yazılırsa,

$$H(x,u,\lambda) = \tfrac{1}{2}(x_1^2 + x_2^2 + u^2) + \lambda_1 x_2 + \lambda_2[-x_1 + a(1-x_1^2)x_2 + u]$$

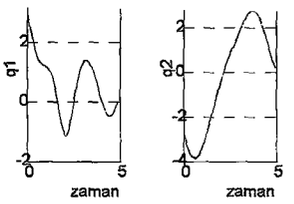
$$\dot{\lambda} = -H_x = -\begin{bmatrix} H_{x_1} \\ H_{x_2} \end{bmatrix} = \begin{bmatrix} (1+2ax_1x_2)\lambda_2 - x_1 \\ -\lambda_1 + a(x_1^2-1)\lambda_2 - x_2 \end{bmatrix}, \quad \lambda(5) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$H_u = u + \lambda_2 = 0$ olup optimallik şartıdır.

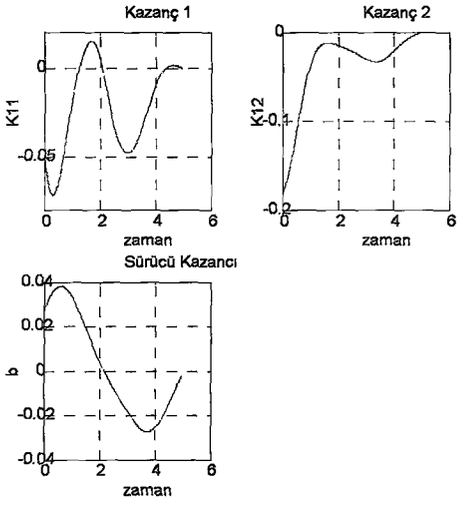
$$f_x = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -(1+2ax_1x_2) & a(1-x_1^2) \end{bmatrix}, \quad f_u = \begin{bmatrix} \frac{\partial f_1}{\partial u} \\ \frac{\partial f_2}{\partial u} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$



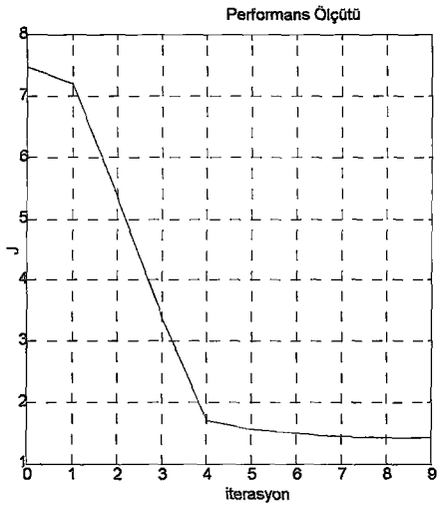
Şekil 5. 8 İDDÖAY için başlangıç durumunda artırılmış Hamiltonian ve $\tilde{H}_{\delta u}$'nun değişimleri (W^0=100)



Şekil 5. 9 İDDÖAY için başlangıç durumunda Riccati matrisi ve sürücü vektör elemanlarının değişimleri (W^0=100)



Şekil 5. 10 İDDÖAY için başlangıç durumunda kazanç ve sürücü kazancın değişimleri (W^0=100)



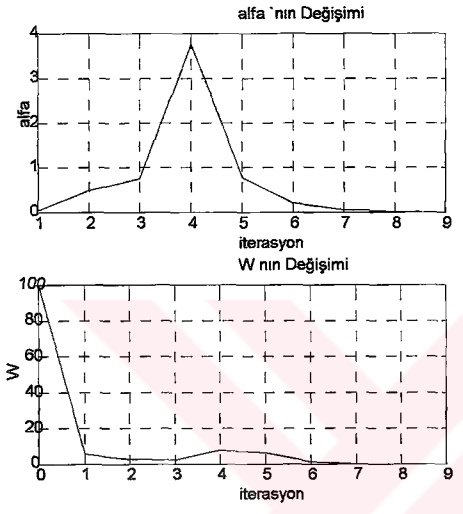
Şekil 5. 11 İDDÖAY için PÖ'nün iterasyonla değişimi (W^0=100)

$$H_x = \begin{bmatrix} \frac{\partial H}{\partial x_1} \\ \frac{\partial H}{\partial x_2} \end{bmatrix} = \begin{bmatrix} -(1+2ax_1x_2)\lambda_2 + x_1 \\ \lambda_1 - a(x_1^2 - 1)\lambda_2 + x_2 \end{bmatrix}, H_{xx} = \begin{bmatrix} \frac{\partial^2 H}{\partial x_1^2} & \frac{\partial^2 H}{\partial x_1 x_2} \\ \frac{\partial^2 H}{\partial x_2 x_1} & \frac{\partial^2 H}{\partial x_2^2} \end{bmatrix} = \begin{bmatrix} -2ax_2\lambda_2 + 1 & -2ax_1\lambda_2 \\ -2ax_1\lambda_2 & 1 \end{bmatrix}$$

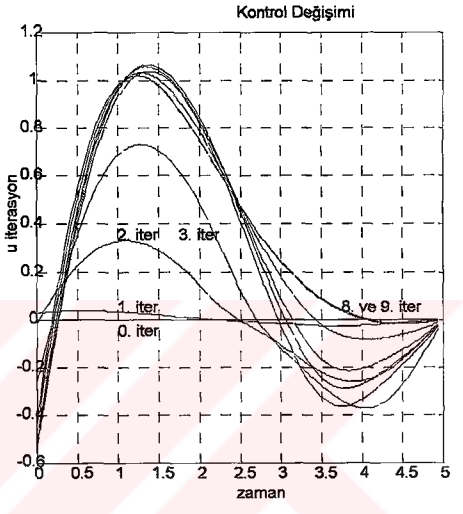
$$P(5) = \phi_{xx}(5) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad q(5) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$A(t) = \begin{bmatrix} 0 & 1 \\ -(1+2ax_1x_2) & a(1-x_1^2) \end{bmatrix}, \quad B(t) = (1+W)^{-1}\begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix}$$

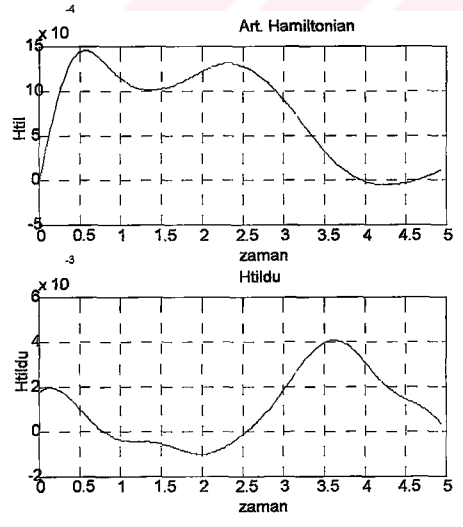
$$C(t) = \begin{bmatrix} -2ax_2\lambda_2 + 1 & -2ax_1\lambda_2 \\ -2ax_1\lambda_2 & 1 \end{bmatrix}$$



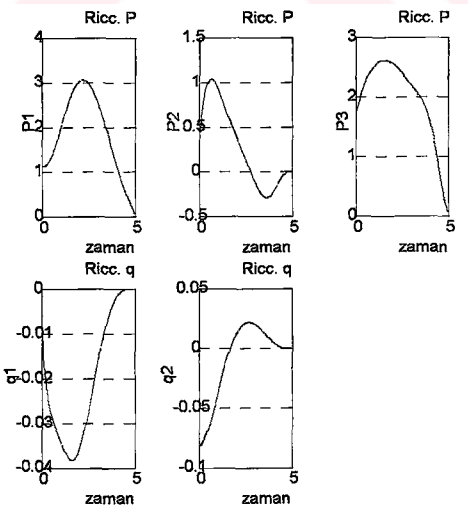
Şekil 5. 12 İDDÖAY için α ve W'nın iterasyonla değişimleri (W^0=100)



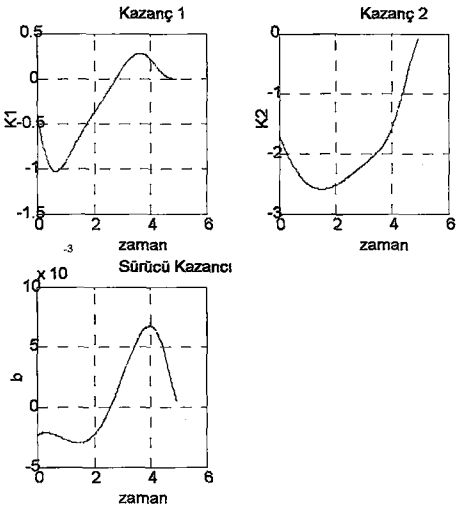
Şekil 5. 13 İDDÖAY ile elde edilen kontrol fonksiyonunun iterasyonla değişimi (W^0=100)



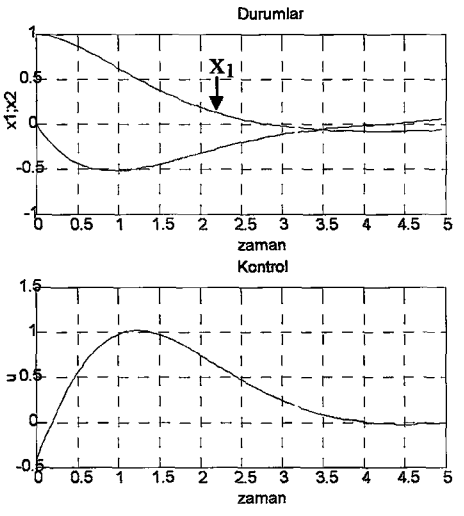
Şekil 5. 14 İDDÖAY için optimal noktadaki artırılmış Hamiltonian ve $\tilde{H}_{\delta u}$'nun değişimi (W^0=100)



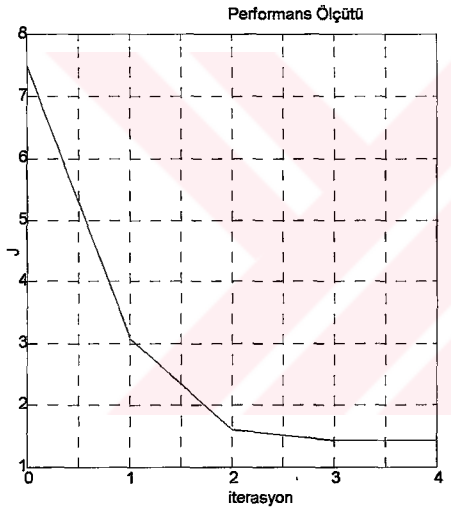
Şekil 5. 15 İDDÖAY için optimal noktadaki Riccati matrisi ve sürücü vektörlerin değişimi (W^0=100)



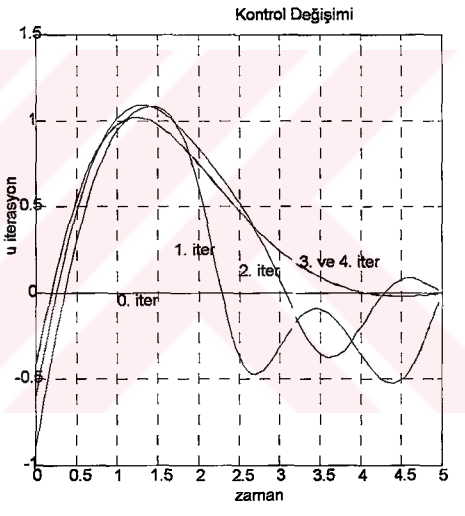
Şekil 5. 16 İDDÖAY için optimal noktadaki kazanç ve sürücü kazancın değişimi (W^0=100)



Şekil 5. 17 İDDÖAY ile bulunan optimal yörünge ve kontrol değişimi (W^0=100)



Şekil 5. 18 İDDÖAY ile PÖ'nün iterasyonla değişimi (W^0=0)



Şekil 5. 19 İDDÖAY ile elde edilen kontrol fonksiyonunun iterasyonla değişimi (W^0=0)

$$v(t) = (1+W)^{-1}(u+\lambda_2)\begin{bmatrix}0\\-1\end{bmatrix}, \quad w(t) = \begin{bmatrix}0\\0\end{bmatrix}$$

Bu ifadeler, (3.82) ve (3.83) eşitliklerinde yerlerine yazılırsa, Riccati ve sürücü diferansiyel denklemler sırayla,

$$\dot{P}_{11} = 2(1+2ax_1x_2)P_{12} + (1+W)^{-1}P_{12}^2 - (1-2ax_2\lambda_2), \qquad P_{11}(5) = 0$$

$$\dot{P}_{12} = -P_{11} + a(x_1^2 - 1)P_{12} + (1 + 2ax_1x_2)P_{22} + (1+W)^{-1}P_{12}P_{22} + 2ax_1\lambda_2, \qquad P_{12}(5) = 0$$

$$\dot{P}_{22} = -2P_{12} + 2a(x_1^2 - 1)P_{22} + (1+W)^{-1}P_{22}^2 - 1, \qquad P_{22}(5) = 0$$

ve

$$\dot{q}_1 = [(1 + 2ax_1x_2) + (1+W)^{-1}P_{12}]q_2 + (1+W)^{-1}P_{12}(u + \lambda_2), \qquad q_1(5) = 0$$

$$\dot{q}_2 = -q_1 + [a(x_1^2 - 1) + (1+W)^{-1}P_{22}]q_2 + (1+W)^{-1}P_{22}(u + \lambda_2), \qquad q_2(5) = 0$$

olarak belirlenir. Kazanç matrisi K(t) ve sürücü kazanç b(t) aşağıdaki gibi bulunur:

$$K(t) = -(1+W)^{-1}\begin{bmatrix} P_{12} & P_{22} \end{bmatrix} = \begin{bmatrix} K_{11}(t) & K_{12}(t) \end{bmatrix}$$

$$b(t) = -(1+W)^{-1}(u + \lambda_2 + q_2)$$

Artırımsal kontrol, adjoint ve Hamiltonian aşağıdaki gibidir:

$$\delta u(t) = K(t)\delta x + b(t) = -(1+W)^{-1}(P_{12}\delta x_1 + P_{22}\delta x_2 + u + \lambda_2 + q_2)$$

$$\delta\lambda = P\delta x + q = \begin{bmatrix} \delta\lambda_1(t) \\ \delta\lambda_2(t) \end{bmatrix} = \begin{bmatrix} P_{11}\delta x_1 + P_{12}\delta x_2 + q_1 \\ P_{12}\delta x_1 + P_{22}\delta x_2 + q_2 \end{bmatrix}$$

$$\tilde{H} = \tilde{L} + \delta\lambda_1\delta x_2 - \delta\lambda_2[(1 + 2ax_1x_2)\delta x_1 + a(1 - x_1^2)\delta x_2 + \delta u]$$

Burada,

$\tilde{L} = (u + \lambda_2)\delta u + \frac{1}{2}[(-2ax_2\lambda_2 + 1)\delta x_1^2 - 4ax_1\lambda_2\delta x_1\delta x_2 + \delta x_2^2 + (1+W)\delta u^2]'$dir.

Durağanlık şartı ise,

$\tilde{H}_{\delta u} = u + \lambda_2 + (1+W)^{-1}\delta u + \delta\lambda_2$ olarak bulunur.

Bir sonraki iterasyon için kontrol fonksiyonu,

$$u^{k+1}(t) = u^k(t) + \delta u^k(t) = u^k(t) + K^k(t)\delta x^k + b^k(t) = u^k + K_{11}^k \delta x_1^k + K_{12}^k \delta x_2^k + b^k$$

$$\delta x^k = x^{k+1} - x^k$$

olarak hesaplanır. Kontrol sınırlama ağırlık matrisi güncelleme katsayısı,

$$\alpha = \frac{\Delta J}{J} \left| \frac{\int_0^5 (u + \lambda_2)\delta u dt}{\int_0^5 \frac{1}{2}[(1 - 2ax_2\lambda_2)\delta x_1^2 - 4ax_1\lambda_2\delta x_1\delta x_2 + \delta x_2^2 + (1+W)\delta u^2)]dt} \right| \text{ olup,}$$

$W^{k+1} = \alpha^k W^k$ olarak güncellenir.

Bir adım sonra bulunacak yörüngenin ((k+1). yörünge) hesabı için, Bölüm 3'te anlatıldığı gibi geri-besleme formunda çözüm yapılmıştır. Yani,

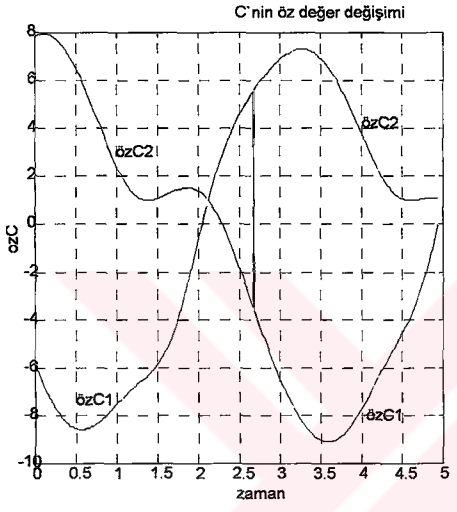
$$\dot{x} = f(x, u^k + K^k(x - x^k) + b^k), \quad x(0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \text{ olarak}$$

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -x_1 + a(1 - x_1^2)x_2 + u^k + K_{11}^k(x_1 - x_1^k) + K_{12}^k(x_2 - x_2^k) + b^k \end{bmatrix} \text{ ifadesi}$$

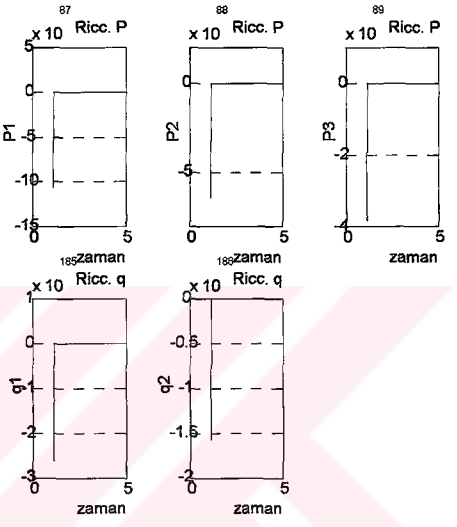
çözülerek $x^{k+1}(t)$ durum yörüngesi elde edilir ki buradan δx^k ve δu^k artırımsal değişken ifadeleri kolayca bulunabilir.

İDHY'nin algoritma yapısı Şekil 3.4'te verildiği gibidir. Bu yöntemde , $u^0(t)$=0.0 ve W^0=0 olarak başlandığında konjuge nokta problemi ile karşı karşıya kalınır. Bunu göstermek için iterasyon başında hesaplanan değişkenlere dikkat edilirse, (3.90) koşulunun sağlanmadığı ve bu sebeple Riccati diferansiyel denklem çözümünün ıraksadığı görülecektir. Şekil 5. 20'de C(t) matrisinin öz değerlerinin negatif olduğu (negatif tanımlı) gösterilmiştir. Şekil 5. 21'de ise, Riccati matris ve sürücü vektör değişkenlerinin değişimi, $t' = 1.16$'da konjuge noktada ıraksadığı açıkça görülmektedir. Bu durum [46]'da detaylı bir şekilde verilmiştir. Bu yaklaşımla çözüme ulaşmak ancak başka bir $u^0(t)$ ile başlamakla mümkün olabilecektir. Ancak, Bölüm 3'te anlatıldığı gibi yeteri kadar büyük W ilavesiyle problemin çözülebileceği

gösterilebilir. Bunun için, W^0=500, $\gamma = 5.10^{-3}$ ve $\gamma_W = 5.10^{-2}$ alınarak, İDDÖAY'ndeki gibi İDHY algoritmasına göre çözüm yapılarak konjuge noktaya takılmadan problem çözülmüştür. Her adımda gerekli koşullar sağlanmaya çalışılmakta ve adjoint değişkeni hesaplanmaktadır. Diğer tüm değişkenler öncekine benzer şekilde çıkmaktadır. Şekil 5. 22'de PÖ'nün ve Şekil 5. 23'te W'nın iterasyonla değişimleri gösterilmektedir. Ayrıca, Şekil 5. 24'te kontrol fonksiyonunun iterasyonla değişimi verilmiştir.



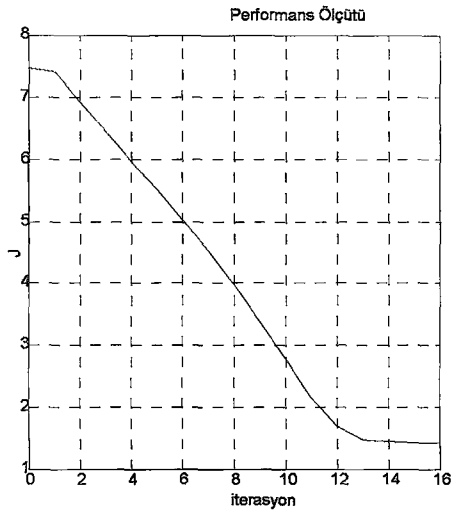
Şekil 5. 20 İDHY'de W=0 ve $u^0(t)$=0.0 için başlangıçta C(t)'nin öz değerlerinin negatif düzlemde olduğunun gösterilmesi



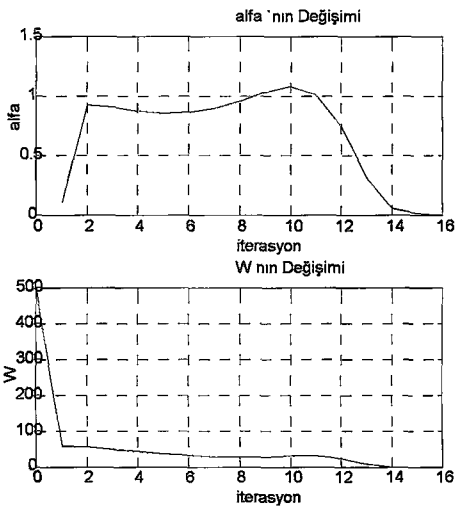
Şekil 5. 21 İDHY'de başlangıçta Riccati matrisinin $t' = 1.16$ sn'de konjuge noktaya takılması (W^0=0, $u^0(t)$=0.0)

5.1.2. Optimal geri besleme (OGB) kanununun VDP sistemi üzerinde değişik bozucular için etkinliğinin gösterilmesi

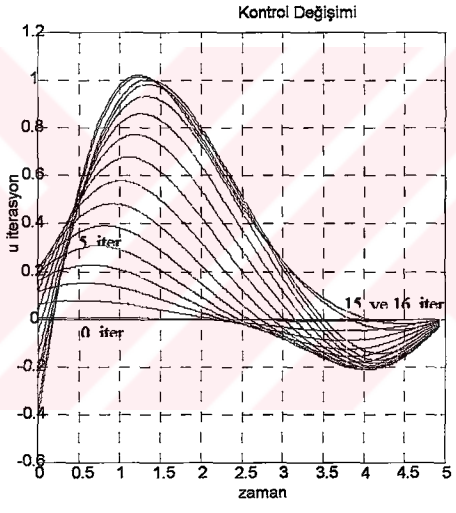
Optimal geri-besleme (OGB) yapısı Bölüm 4'te anlatıldığı gibi, ikinci derece yöntemlerin uygulanması sonunda direkt olarak çıkarılmaktadır. OGB kanununun bozuculara karşı nasıl etkin bir şekilde davrandığı burada gösterilecektir. Burada, üç çeşit bozucunun durumları incelenecektir. Birincisi, sistem parametresinin değişmesi durumu, ikincisi, başlangıç durumunun değişmesi, üçüncüsü ise herhangi bir anda sisteme uygulandığı düşünülen basamak bozması durumlarıdır.



Şekil 5. 22 İDHY'de PÖ'nün iterasyonla değişimi (W^0=500, $u^0(t)$=0.0)



Şekil 5. 23 İDHY'de α ve W'nın iterasyonla değişimleri (W^0=500, $u^0(t)$=0.0)



Şekil 5. 24 İDHY için kontrol fonksiyonunun iterasyonla değişimi (W^0=500, $u^0(t)$=0.0)

OGB denemeleri için aşağıdaki parametreler kullanılacaktır.

t_0=0sn, t_f=5sn, $Q = \begin{bmatrix} 10 & 0 \\ 0 & 1 \end{bmatrix}$, $R = 0.1$, $S = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}$, $xd(t) = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$, $x(0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$,

$u^0(t)$=0.0 ve VDP osilatör parametresi a=1.0 alınmıştır.

İkinci dereceden yöntem olarak Bölüm 3'te anlatılan her iki yöntemde kullanılabilir. Burada verilen sonuçlar İDDÖAY ile elde edilen değerlerdir. Bu yöntemin daha gürbüz ve hızlı olduğu daha önceden anlatılmıştı.

Verilen parametre ve durum yörüngelerine göre elde edilen ileri besleme optimal

kontrol (İBOK) fonksiyonu ve optimal durum yörüngeleri Şekil 5. 25'te verilmiştir. Şekil 5. 26'da ise optimal geri-besleme kazanç matris ve vektör değişimleri verilmektedir. PÖ'nün değişimi Şekil 5. 27'de, kontrol fonksiyonunun iterasyonla değişimi Şekil 5. 28'de verilmektedir. Şimdi de, yukarıda söylenen üç bozucu durum incelenecektir:

i) *Sistem parametresinin değişmesi durumu*: Normal halde VDP sistem parametresi p=1.0 idi. Herhangi bir nedenle parametre değişirse, eğer sistemde sadece İBOK varsa sistem istenilen yörüngeyi takip edemeyecektir. Burada, a=0.1 alınarak %90 oranında değiştirilmiştir. Sadece İBOK durumunda sistem durum yörüngeleri Şekil 5. 29'da gösterilmektedir. Görüldüğü gibi, parametre değişimi sistemi olumsuz biçimde etkileyerek gerçek yörüngeden oldukça saptırmıştır. Eğer, geri-besleme yapısı uygulanırsa, sistem yörüngeleri Şekil 5. 30'da görüldüğü gibi olmaktadır. Sonuçta, OGB kanunu çok etkili bir şekilde sistemi gerçek yörünge üzerinde tutmaktadır. Bu da, sistem parametre değişimlerine duyarsız bir şekilde geri-besleme yapısının gürbüz bir cevap verdiği anlaşılmaktadır.

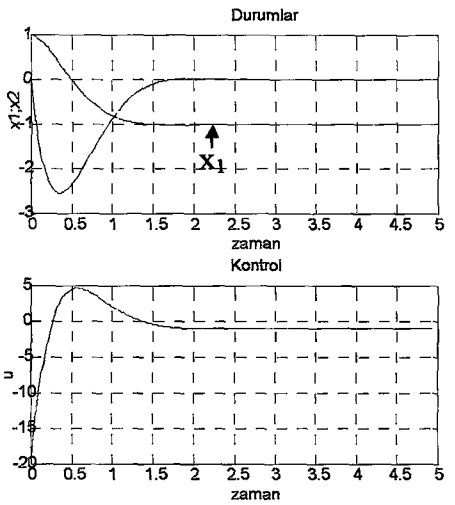
ii) *Başlangıç koşullarının değişmesi durumu*: Normal çözümde, $x(0) = \begin{bmatrix} 1 & 0 \end{bmatrix}^T$ olarak alınmıştır. Şimdi de, başlangıç durumunun herhangi bir sebeple değiştiği, örnek olarak $x(0) = \begin{bmatrix} -2 & 3 \end{bmatrix}^T$ olduğu düşünülsün. Bu durumda sadece İBOK uygulanırsa sistem yörüngeleri Şekil 5. 31'de olduğu gibi farklı olacaktır. Sisteme bulunan optimal kontrol uygulanmasına rağmen, başlangıç koşulları değiştiği için cevap yörüngeleri sapmış şekilde çıkmıştır. Eğer sisteme, bulunan OGB kanunu uygulanırsa sonuç, Şekil 5. 32'de gösterildiği gibi gerçek yörüngeleri çok iyi bir şekilde takip edecektir. Görüldüğü gibi, başlangıç bozması için OGB yapısı iyi bir şekilde kompanze edici görev üslenmektedir.

iii) *Basamak bozması durumu*: Sistem nominal şartların dışında, t=2sn anında $d_2(t)$=5 genliğinde t=5sn'ye kadar basamak bozmasına tabi tutulmaktadır. Bu bozma, $\dot{x}_2(t)$'ye uygulanmıştır. Sadece İBOK uygulanması durumunda Şekil 5. 33'deki gibi $x_2(t)$'de oldukça büyük bir aşım olmaktadır. $x_1(t)$'de fiziksel olarak büyük bir aşımla cevap vermiştir. Bu durum ise bir çok sistem için kesinlikle istenmeyen haldir. Şekil 5. 34'te ise OGB uygulanarak sistem cevapları görülmektedir ve az bir aşımla sistem istenilen yörüngede tutulmaktadır.

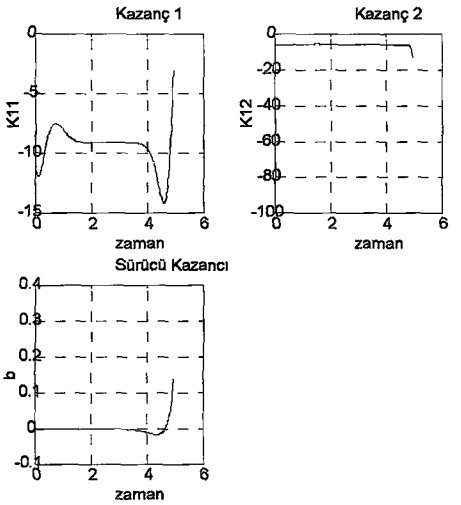
Yukarıda yapılan uygulamada sabit bir yörüngeye gitmek hedeflenmiştir. LOOK algoritmasından çıkarılan sonuçlar, istenilen herhangi bir değişken yörünge için de kullanılabilir. Bunun için de uygulamalar yapılacaktır. Aynı parametreler kullanılarak $x^d_1(t)=\exp(-t)$ olarak birinci durum yörüngesinin üstel olarak azalan bir şekilde bir yörüngeyi takip etmesi istenmiştir. İkinci durum yörüngesi dinamik denklemlere göre birinci durumun türevi olacağı aşikardır. Şekil 5. 35'te algoritma sonunda bulunan optimal kontrol ve x_1 yörünge fonksiyonlarının değişimleri görülmektedir. İyi bir yaklaşımla istenen yörünge takip edilmektedir. Şekil 5. 36'da ise yörüngelerin faz değişimleri görülmektedir.

Bir başka uygulama olarak, sinüzoidal bir yörüngenin takip edilmesi problemi ele alınacaktır. Bunun için, $Q=\begin{bmatrix}100 & 0\\ 0 & 0\end{bmatrix}$, $R=0.001$, $S=\begin{bmatrix}2 & 0\\ 0 & 0\end{bmatrix}$, olup deneyimsel olarak belirlenmiştir. Takip edilmesi istenen yörünge $xd_1(t)=0.5\sin(2\pi t)$ olarak yarım genlikli 1Hz'lik sinüs yörüngesidir. Sonuçta bulunan optimal kontrol ve x_1 yörünge fonksiyonlarının değişimleri Şekil 5. 37'de gösterildiği gibidir. Görüldüğü gibi uygun ölçüt parametreleri seçerek sistem istenilen yörüngeyi iyi bir şekilde takip edebilmektedir. Ancak burada unutulmaması gereken bir nokta vardır ki, takip edilmesi istenen yörüngenin dinamik sistemin fiziki şartlarına uygun olması gerekir. Şekil 5. 38 ise sistemin faz değişimini göstermektedir. Buradan sistemin istenilen yörüngeyi takip ettiği de açık olarak görülmektedir.

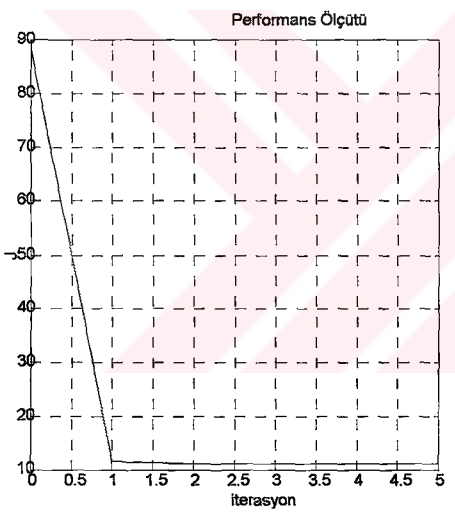
Diğer bir uygulama da, değişik genlikli kare dalga yörüngesinin takip edilmesi problemidir. Burada, $Q=\begin{bmatrix}10 & 0\\ 0 & 1\end{bmatrix}$, $R=0.1$, $S=\begin{bmatrix}10 & 0\\ 0 & 10\end{bmatrix}$ olarak önce kullanılan parametreler kullanılacaktır. Son zaman değeri $t_f=12$ alınmıştır. İstenen yörünge $xd_1(t)$, değişik zamanlar için 1.5, -1, ve 0 olarak alınmıştır. Bulunan optimal kontrol, $x_1(t)$ ve $xd_1(t)$ yörüngeleri Şekil 5. 39'gösterilmiştir. Faz diyagramı ise Şekil 5. 40'ta gösterildiği gibidir. Görüldüğü gibi, istenilen yörüngeye sistem iyi bir şekilde gitmektedir.



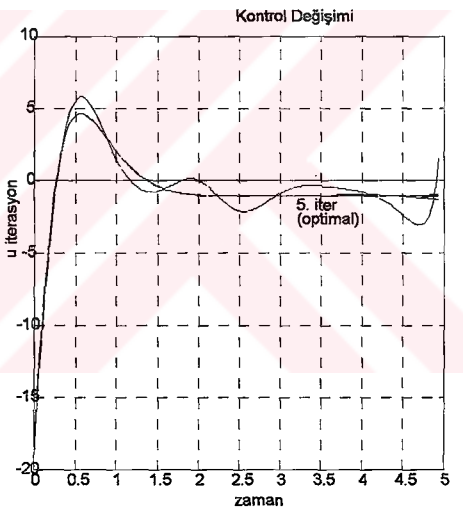
Şekil 5. 25 İDDÖAY ile elde edilen optimal yörünge ve kontrol fonksiyonlarının değişimi (W^0=0, u^0(t)=0.0, a=1.0)



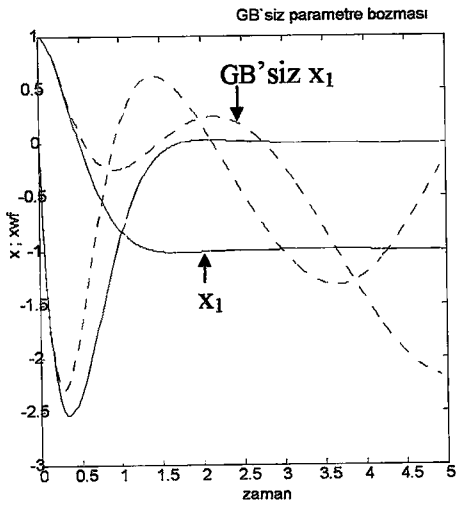
Şekil 5. 26 İDDÖAY sonunda elde edilen optimal geri-besleme (OGB) kazançlarının değişimi (W^0=0, u^0(t)=0.0, a=1.0)



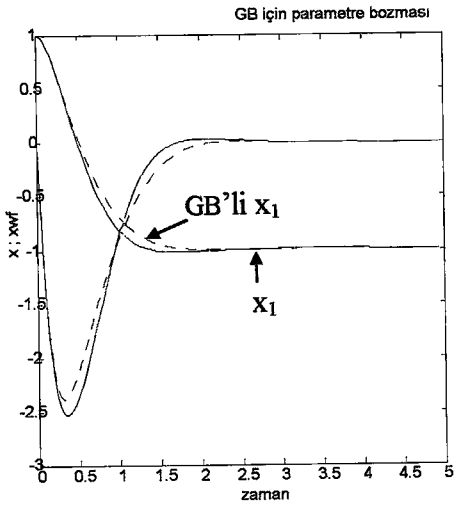
Şekil 5. 27 İDDÖAY için PÖ'nün değişimi (W^0=0, u^0(t)=0.0, a=1.0)



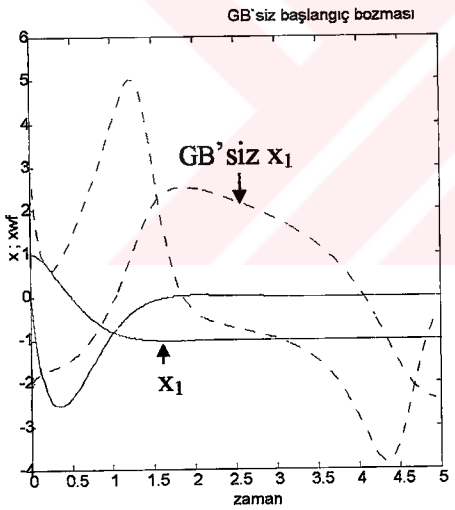
Şekil 5. 28 İDDÖAY ile elde edilen kontrol iterasyonunun değişimi (W^0=0, u^0(t)=0.0, a=1.0)



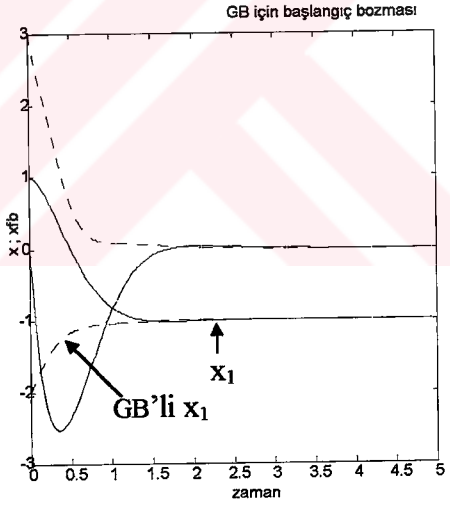
Şekil 5. 29 Sadece İBOK uygulanması sonucunda bozucu olarak a=0.1 için VDP sisteminin yörünge değişimleri (kesikli çizgiler GB'siz durum değişimleri, sürekli çizgiler optimal durumlar)



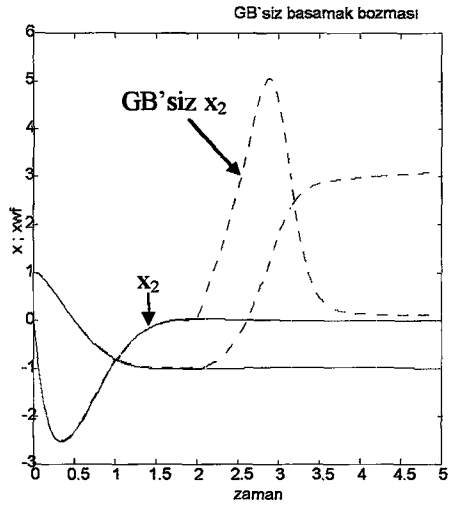
Şekil 5. 30 İBOK ve OGB uygulanması durumunda bozucu olarak a=0.1 için VDP sisteminin yörünge değişimleri (kesikli çizgiler GB'li durum değişimleri, sürekli çizgiler optimal durumlar)



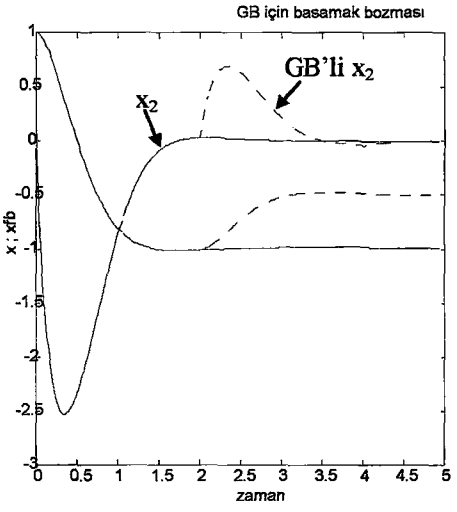
Şekil 5. 31 Sadece İBOK uygulanması sonucunda bozucu olarak $x(0)=[-2\ 3]^T$ için VDP sisteminin yörünge değişimleri (kesikli çizgiler GB'siz durum değişimleri, sürekli çizgiler optimal durumlar)



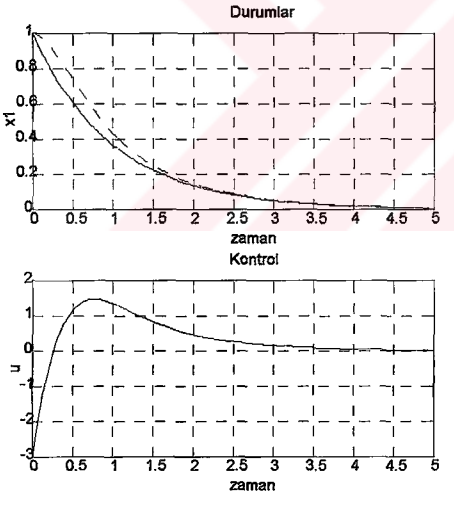
Şekil 5. 32 İBOK ve OGB uygulanması durumunda bozucu olarak $x(0)=[-2\ 3]^T$ için VDP sisteminin yörünge değişimleri (kesikli çizgiler GB'li durum değişimleri, sürekli çizgiler optimal durumlar)



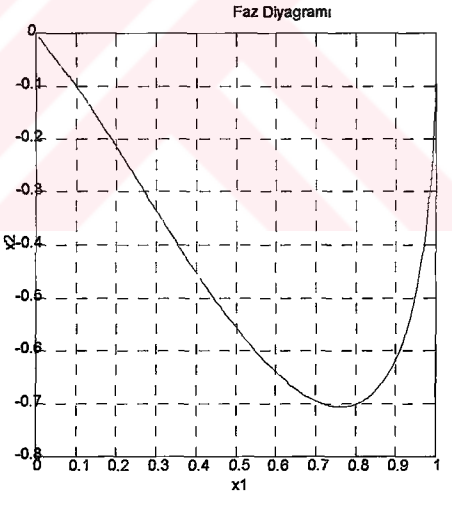
Şekil 5. 33 Sadece İBOK uygulanması sonucunda bozucu olarak t=2sn'de $d_2(t)=5$ büyüklüğünde uygulanan bozucunun yörüngelere etkisi (kesikli çizgiler GB'siz durum değişimleri, sürekli çizgiler optimal durumlar)



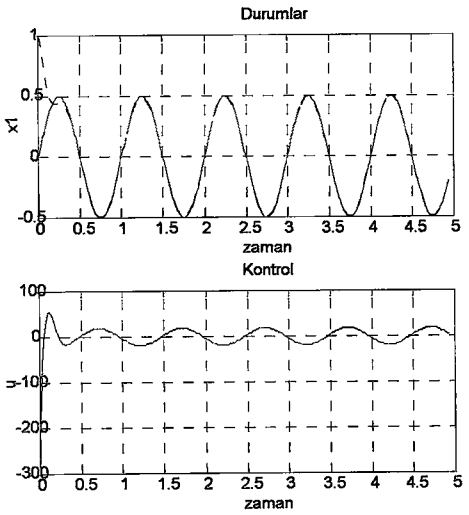
Şekil 5. 34 İBOK ve OGB uygulanması durumunda bozucu olarak olarak t=2sn'de $d_2(t)=5$ büyüklüğünde uygulanan bozucunun kompanze edilmesi (kesikli çizgiler GB'li durum değişimleri, sürekli çizgiler optimal durumlar)



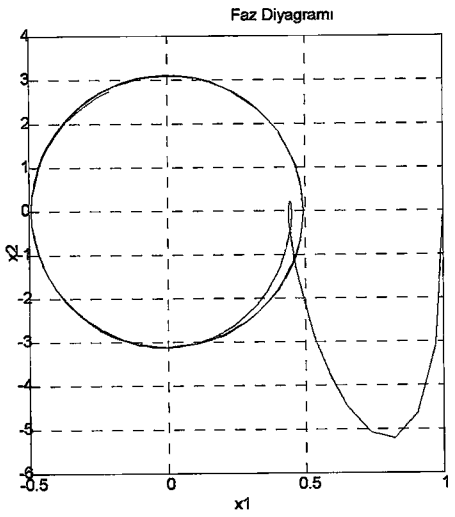
Şekil 5. 35 Değişken bir yörünge ($xd_1=\exp(-t)$) için İDDÖAY sonunda bulunan optimal kontrol ve yörünge fonksiyonları (kesikli çizgi optimal durum yörüngesi, sürekli çizgi istenen durum yörüngesi)



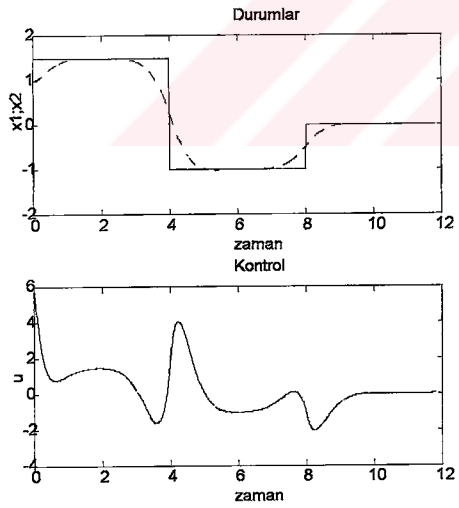
Şekil 5. 36 Değişken bir yörünge ($xd_1=\exp(-t)$) için İDDÖAY sonunda bulunan optimal yörünge fonksiyonlarının faz diyagramı



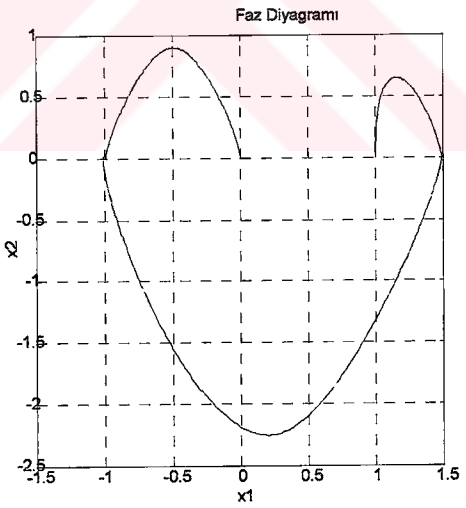
Şekil 5. 37 Zamanla değişken bir yörünge (xd_1=0.5sin(2πt)) için İDDÖAY sonunda bulunan optimal kontrol ve yörünge fonksiyonları (kesikli çizgi optimal durum yörüngesi, sürekli çizgi istenen durum yörüngesi)



Şekil 5. 38 Zamanla değişken bir yörünge (xd_1=0.5sin(2πt)) için İDDÖAY sonunda bulunan optimal yörünge fonksiyonlarının faz diyagramı



Şekil 5. 39 Zamanla değişken bir yörünge için İDDÖAY sonunda bulunan optimal kontrol ve yörünge fonksiyonları (kesikli çizgi optimal durum yörüngesi, sürekli çizgi istenen durum yörüngesi)



Şekil 5. 40 Zamanla değişken bir yörünge için İDDÖAY sonunda bulunan optimal yörünge fonksiyonlarının faz diyagramı

5.2. Ekzotermik CSTR Sistemi

İlk olarak lineer olmayan CSTR (continuous stirred tank reactor) modeli Aris ve Amundson tarafından geliştirilmiş olup izole edilmemiş şartlarda, birinci dereceden terslenemez kimyasal reaksiyonu içerir [8]. Burada kontrol için soğutucu akışının hızından faydalanılmıştır.

Bu çalışmada, Ray [110]'ın modeli kullanılmıştır. CSTR lineer olmayan bir kimyasal prosestir. Bu yüzden kontrol edilmesi çok zordur. CSTR, en basit haliyle $A \rightarrow B$'ye birinci dereceden ekzotermik terslenemez reaksiyonu içerir. Bu sistem, oldukça parametrik duyarlılığa sahiptir ve bazı çalışma şartları için lineer olmayan osilasyonlar gösterir. Aşağıda, Ray'ın kullandığı CSTR'ın matematiksel modeli verilmiştir:

$$V\frac{dC_A}{dt'} = F(C_{Af} - C_A) - Vk_0 \exp(-\frac{E}{RT})C_A \qquad (5.8)$$

$$V\rho C_p \frac{dT}{dt'} = \rho C_p F(T_f - T) + V(-\Delta H)k_0 \exp(-\frac{E}{RT})C_A - hA(T - T_c) \qquad (5.9)$$

Şekil 5. 41'de gösterildiği gibi, V reaktör hacmi, F besleme hızı, C_{Af},T_f sırayla tanka giren reaktant konsantrasyonu ve sıcaklığı ve C_A,T ise reaktördeki sırayla konsantrasyon ve sıcaklık değerleridir. T_c miktarı ceket soğutma sıcaklığıdır. Fiziksel parametreler olan $\rho C_p, h, A, E, \Delta H$'ın hepsi sabit kabul edilir. Burada BGBÇ kontrol problemi düşünülmüştür. Reaktör sıcaklığı T, ceketteki T_c soğutma sıcaklığını ayarlayarak kontrol edilecektir. Reaktant konsantrasyonu C_A, reaktör sıcaklığı uygun bir şekilde kontrol edildiğinde istenen değere yerleştiği kabul edilecektir. Besleme sıcaklığı T_f'teki değişimler bozucu olarak düşünülüp kontrol edilmeye çalışılacaktır. Ayrıca, soğutma sıcaklığının mükemmel bir şekilde kontrol edildiği varsayılmaktadır. Bununla beraber, daha gerçekçi CSTR kontrol sistemlerinde ceket soğutucu hızı ile sıcaklık kontrol edilir. Bu yüzden ceket sıcaklık denklemi de dinamik olarak sisteme dahil edilir [111,112]. Burada ceket sıcaklığının hızlı bir şekilde istenilen değere ulaştığı varsayımı yapılmıştır.

Lineer olmayan CSTR'ın matematiksel denklemleri, aşağıdaki tanımlamalarla boyutsuz değişkenlere dönüştürülür:

$$x_1 = \frac{C_{Af} - C_A}{C_{Af}},\ x_2 = \frac{T - T_{f0}}{T_{f0}}\left[\frac{E}{RT_{f0}}\right],\ \tau = \frac{V}{F},\ \gamma = \frac{E}{RT_{f0}},\ t = \frac{t'}{\tau} = \frac{F}{V}t' \qquad (5.\ 10)$$

$$Da = \frac{k_0 \exp(-\gamma)V}{F},\ B = \frac{(-\Delta H)C_{Af}}{\rho C_p T_{f0}}\left[\frac{E}{RT_{f0}}\right],\ \beta = \frac{hA}{F\rho C_p},\ d = \frac{T_f - T_{f0}}{T_{f0}}\left[\frac{E}{RT_{f0}}\right] \qquad (5.\ 11)$$

$$x_{2c} = \frac{T_{c0} - T_{f0}}{T_{f0}}\left[\frac{E}{RT_{f0}}\right],\ u = x_{2c}(T_c) - x_{2c}(T_{c0}) = \frac{T_c - T_{c0}}{T_{f0}}\left[\frac{E}{RT_{f0}}\right] \qquad (5.\ 12)$$

$$y = x_2 - x_{2s},\ x_{2s} = \frac{T_s - T_{f0}}{T_{f0}}\left[\frac{E}{RT_{f0}}\right] \qquad (5.\ 13)$$

Bu ifadeler kullanılırsa;

$$\frac{dx_1}{dt} = -x_1 + Da(1 - x_1)\exp\left[\frac{x_2}{1 + x_2/\gamma}\right] \qquad (5.\ 14)$$

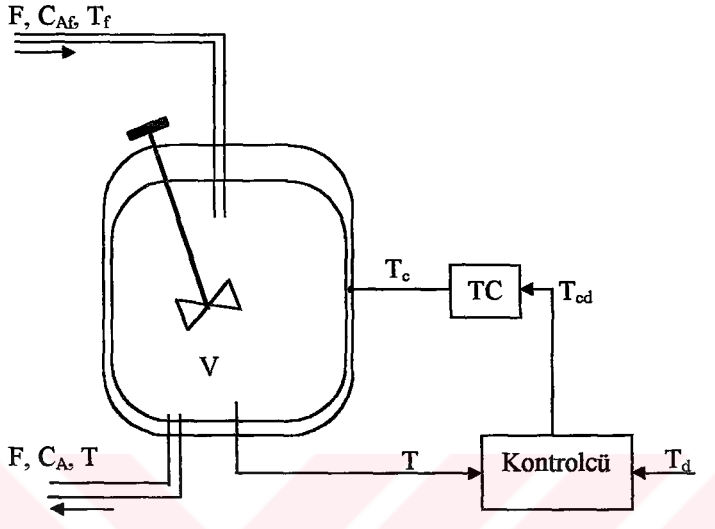
$$\frac{dx_2}{dt} = -x_2 + BDa(1 - x_1)\exp\left[\frac{x_2}{1 + x_2/\gamma}\right] - \beta(x_2 - x_{2c}) + d + \beta u \qquad (5.\ 15)$$

ve gerçek değişkenler ise aşağıdaki gibidir:

$$C_A = (1 - x_1)C_{Af},\ T = (1 + x_2/\gamma)T_{f0},\ T_f = (1 + d/\gamma)T_{f0},\ T_c = T_{c0} + uT_{f0}/\gamma \qquad (5.\ 16)$$

Burada, Da Damköhler sayısı olup reaktör katalisit konsantrasyonu değişmesiyle değişebilir. x_1 ve x_2 sırayla boyutsuz reaktör konsantrasyon ve sıcaklığını temsil eder. $T_{f\,0}$, T_{c0}, T_s sırayla besleme, soğutucu ve reaktör sıcaklığının nominal tasarım

değerleridir. Bununla beraber T_s aynı zamanda, sabit T_{f0}, T_{c0} için Da'ya da bağlıdır. Ayrıca, d, u, y değişkenleri sırayla, besleme sıcaklık bozması (T_f), kontrol (T_c), ve çıkış (T) boyutsuz sapma değişkenleridir.



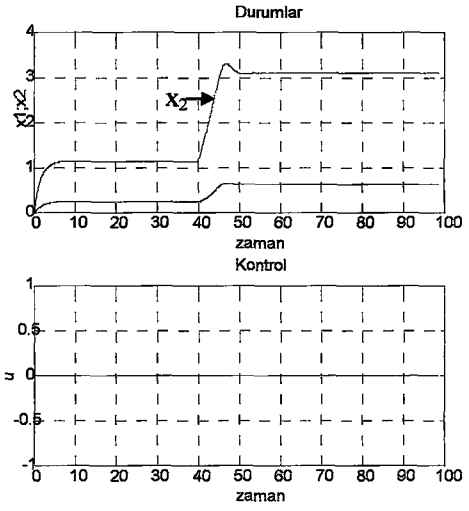
Şekil 5. 41 CSTR sıcaklık kontrol sistemi diyagramı

CSTR'ın lineer olmayan davranışını inceleyerek kontrol yapısına girilecektir. CSTR'ın parametrelerinin bazı bölgelerdeki değişimleriyle besleme sıcaklığındaki küçük değişimler, reaktör sıcaklığında büyük artışlara ve salınımlara neden olur. Bu durumlar aşağıda gösterilmiştir. Bu bilgiler için Ray [110]'ın makalesine başvurulabilir. Bütün parametreler nominal besleme sıcaklığında ($T_{f0}=300^0K$) belirlenmiştir:

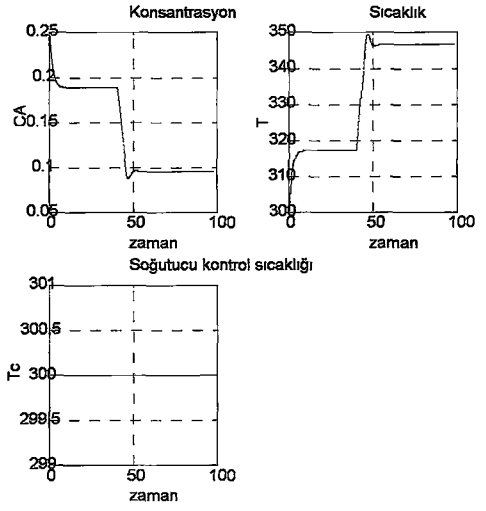
i) B=7.0, β=0.5, γ=20, Da=0.11 için;

Burada reaktör $x(0)=[0\ 0]^T$'dan başlayarak simüle edilecektir. Besleme sıcaklığındaki (T_f) değişimler, reaktör sıcaklığını değiştirmektedir. $T_f=300^0K$ için kararlı durum reaktör sıcaklığı 317^0K civarına oturmakla birlikte, besleme sıcaklığındaki 5^0K'lik artım reaktör sıcaklığında yaklaşık 30^0K'lik artışa sebep olmaktadır. Bu durumlar Şekil 5. 42 ve Şekil 5. 43'de sırayla boyutsuz ve gerçek değişkenler açısından gösterilmektedir. Şekil 5. 44'de ise durumların faz diyagramı gösterilmiştir. Kararlı

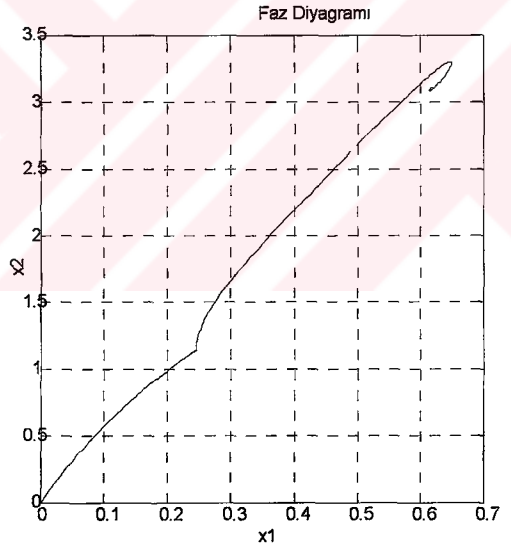
durumların değişmesi burada açıkça görülmektedir.



Şekil 5. 42 CSTR besleme sıcaklığındaki 5^0K'lik artış için açık çevrim durum cevabı (boyutsuz durumlar, B=7.0, β=0.5, γ=20, Da=0.11)



Şekil 5. 43 CSTR besleme sıcaklığındaki 5^0K'lik artış için açık çevrim durum cevabı (gerçek değişken durumları, B=7.0, β=0.5, γ=20, Da=0.11)



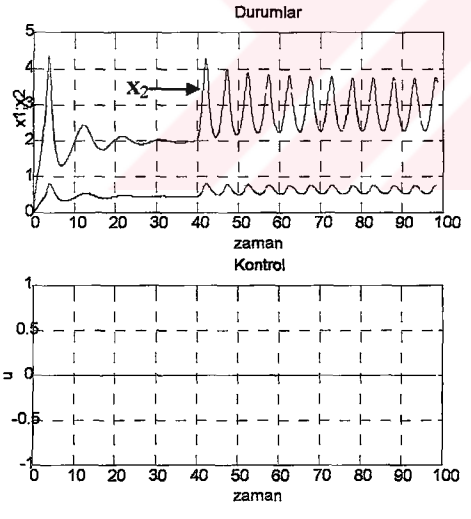
Şekil 5. 44 (i) durumu için boyutsuz durumların faz diyagramı

ii) B=11.0, β=1.5, γ=20, Da=0.135 için;

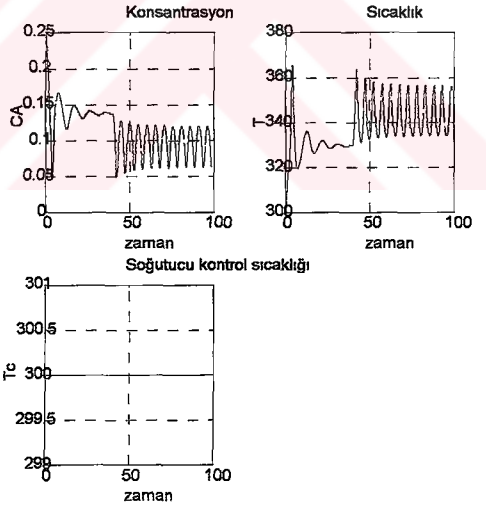
Bu durum, CSTR'ın kendi kendine osilasyona girdiğini göstermektedir. Eğer, besleme sıcaklığı yaklaşık $302^0K<T_f<307^0K$ aralığında ise verilen parametreler altında reaktör

limit döngü osilasyonları yapar. T_f=300^{0}K iken reaktör sıcaklığı bir miktar aşımlara rağmen 330^{0}K'de kararlı duruma oturmakla birlikte, 5^{0}K'lik artma reaktörde limit döngü osilasyonuna neden olur. Bu durum Şekil 5. 45 ve Şekil 5. 46'da gösterilmektedir. Şekil 5. 47'de ise faz diyagramı verilmiştir. Besleme sıcaklığındaki düşme ise ilginç bir sonuç verir. T_f=310^{0}K iken reaktör sıcaklığı büyük sıcaklık aşımlarına rağmen yaklaşık 350^{0}K'de kararlı duruma erişirken, 5^{0}K'lik düşme reaktörde kararsız limit döngü osilasyonuna neden olmaktadır. Durumlar Şekil 5. 48 ve Şekil 5. 49'te gösterilmektedir. Ayrıca, faz diyagramı Şekil 5. 50'te gösterildiği gibi bir hal almaktadır.

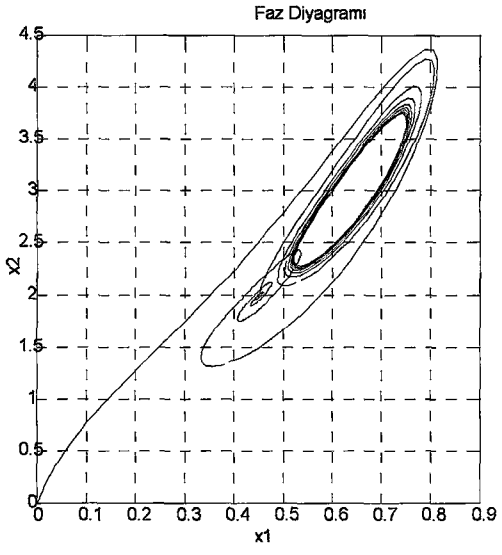
Görüldüğü gibi, parametre değişimleri ve küçük bozucu girişleri reaktör sıcaklığının kararlı durum ve dinamik davranışında büyük ölçüde değişimlere sebep olmaktadır. Bu noktadan bakıldığında, geliştirilecek kontrol sistemlerinin etkinliği ve gürbüzlüğü (robustness) bu özelliklere bakılarak değerlendirilecektir.



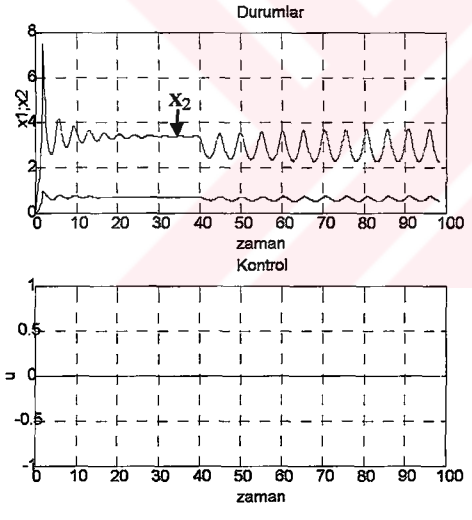
Şekil 5. 45 CSTR besleme sıcaklığındaki 5^{0}K'lik artış için açık çevrim durum cevabı (boyutsuz durumları, B=11.0, β=1.5, γ=20, Da=0.135)



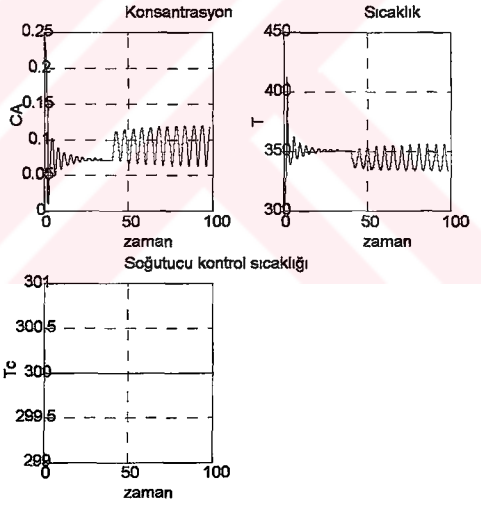
Şekil 5. 46 CSTR besleme sıcaklığındaki 5^{0}K'lik artış için açık çevrim durum cevabı (gerçek değişkenler, B=11.0, β=1.5, γ=20, Da=0.135)



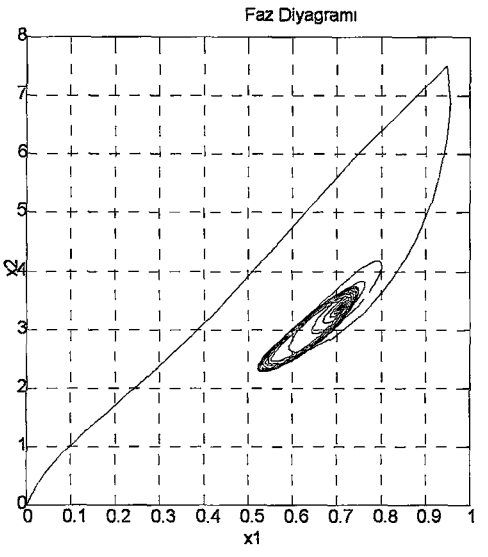
Şekil 5. 47 (ii) durumu için 5^0K'lik bozucu artımına göre boyutsuz durumların faz diyagramı



Şekil 5. 48 CSTR besleme sıcaklığındaki (T_f=310) 5^0K'lik düşme için açık çevrim durum cevabı (boyutsuz durumları, B=11.0, β=1.5, γ=20, Da=0.135)



Şekil 5. 49 CSTR besleme sıcaklığındaki (T_f=310) 5^0K'lik düşme için açık çevrim durum cevabı (gerçek değişkenler, B=11.0, β=1.5, γ=20, Da=0.135)



Şekil 5. 50 (ii) durumu için 5^0K'lik bozucu düşmesine (T_f=305) göre boyutsuz durumların faz diyagramı

5.2.1. Optimal kontrol yöntemlerinin CSTR sistemine uygulanması

Burada, CSTR'ın nominal parametrelerinin (i) durumunda verilen değerlerin (B=7.0, β=0.5, γ=20, Da=0.11) olduğu kabul edilecektir. CSTR bu bölgede, çok küçük değişimlere büyük artışlarla cevap vermektedir. Bu yüzden bu bölgede çalışılacaktır. İleride (Bölüm 8) diğer bölgede çalışılacaktır. (5.2) eşitliği ile verilen PÖ yapısı kullanılacaktır. BDGY için sonuçlar yerine İDDÖAY ve İDHY sonuçları karşılaştırılacaktır. Zaten BDGY yavaş işlemektedir. Ağırlıklı olarak İDDÖAY'nin etkinliği gösterilecektir. Lineer olmayan yapıda olan CSTR sistemi için optimal kontrol yapısının çok iyi sonuç ürettiği gösterilecektir.

Kullanılacak parametreler;

$$t_0=0, \qquad t_f=10,\ Q=\begin{bmatrix}0 & 0\\ 0 & 30\end{bmatrix},\ R=0.05,\ S=\begin{bmatrix}0 & 0\\ 0 & 20\end{bmatrix},\ xd(t)=\begin{bmatrix}0\\ xd_2(t)\end{bmatrix},\ x(0)=\begin{bmatrix}0\\ 0\end{bmatrix}$$

olarak belirlenmiştir. Burada, T^d=317^0K olması istenmektedir. Bunun için belirli bir yörüngenin verildiği düşünülecektir. Yani, eğer verilen bu yörünge takip edilerek sıcaklık değiştirilirse ekonomik çözüme ulaşıldığı varsayılmaktadır. Bu yörüngenin de T^d(t)=317-17*exp(-t) olduğu kabul edilmektedir. Buradan, boyutsuz sıcaklık

değişkeni $xd_2(t)=(T^d(t)-300)*20/300$ olur. $xd_2(10)\cong 1.13$ olarak kolayca belirlenebilir. Boyutsuz başlangıç sıcaklığı $x_2(0)=0$ verildiğinden $T(0)=300^0K$ olduğu bulunabilir. Besleme konsantrasyonu $C_{Af}=0.25$ olup reaktör başlangıç konsantrasyonunun $C_A(0)=0.25$ olduğu düşünülmektedir. Ve PÖ,

$$J=\tfrac{1}{2}20(x_2(10)-1.13)^2+\tfrac{1}{2}\int_0^{10}(30(x_2(t)-xd_2(t))^2+Ru^2(t))dt$$ olarak tespit edilmiş olur.

i) İDDÖAY ile CSTR probleminin çözümü;

Matris Riccati ve vektör sürücü diferansiyel denklemlerinin çözümleri için (Bkz. (3.43)-(3.47) eşitlikleri) aşağıdaki ifadeler çıkarılmıştır:

$$f_x=\begin{bmatrix}\frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2}\\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2}\end{bmatrix}=\begin{bmatrix}-(1+Da\exp(\frac{x_2}{1+x_2/\gamma})) & Da\frac{(1-x_1)}{(1+x_2/\gamma)^2}\exp(\frac{x_2}{1+x_2/\gamma})\\ -BDa\exp(\frac{x_2}{1+x_2/\gamma}) & -(1+\beta-BDa\frac{(1-x_1)}{(1+x_2/\gamma)^2}\exp(\frac{x_2}{1+x_2/\gamma}))\end{bmatrix}$$

$$f_u=\begin{bmatrix}\frac{\partial f_1}{\partial u}\\ \frac{\partial f_2}{\partial u}\end{bmatrix}=\begin{bmatrix}0\\ \beta\end{bmatrix},L_x=\begin{bmatrix}\frac{\partial L}{\partial x_1}\\ \frac{\partial L}{\partial x_2}\end{bmatrix}=\begin{bmatrix}0\\ 30(x_2-1.13)\end{bmatrix},\quad L_{xx}=\begin{bmatrix}\frac{\partial^2 L}{\partial x_1^2} & \frac{\partial^2 L}{\partial x_1x_2}\\ \frac{\partial^2 L}{\partial x_2x_1} & \frac{\partial^2 L}{\partial x_2^2}\end{bmatrix}=\begin{bmatrix}0 & 0\\ 0 & 30\end{bmatrix}$$

$$L_u=\tfrac{\partial L}{\partial u}=Ru(t),\quad L_{uu}=\tfrac{\partial^2 L}{\partial u^2}=R,\quad L_{ux}=\tfrac{\partial}{\partial x}\left(\tfrac{\partial L}{\partial u}\right)=\begin{bmatrix}0 & 0\end{bmatrix},\quad L_{xu}=\tfrac{\partial}{\partial u}\left(\tfrac{\partial L}{\partial x}\right)=\begin{bmatrix}0\\ 0\end{bmatrix}$$

$$P(10)=\phi_{xx}(10)=\begin{bmatrix}0 & 0\\ 0 & 20\end{bmatrix},\quad q(10)=\phi_x(10)=\begin{bmatrix}0\\ 20(x_2(10)-1.13)\end{bmatrix}$$

$$A(t)=f_x,\quad B(t)=(R+W)^{-1}\begin{bmatrix}0 & 0\\ 0 & -\beta^2\end{bmatrix},\quad C(t)=\begin{bmatrix}0 & 0\\ 0 & 30\end{bmatrix}$$

$$v(t)=(R+W)^{-1}\begin{bmatrix}0\\ -R\beta u(t)\end{bmatrix},\quad w(t)=\begin{bmatrix}0\\ 30(x_2-xd_2(t))\end{bmatrix}$$

Bu ifadeler, (3.54) ve (3.55) eşitliklerinde yerlerine yazılırsa, Riccati ve sürücü diferansiyel denklemler sırayla,

$$\dot{P}_{11} = 2(1 + Da\exp(\tfrac{x_2}{1+x_2/\gamma}))P_{11} + 2BDa\exp(\tfrac{x_2}{1+x_2/\gamma})P_{12} + (R+W)^{-1}\beta^2 P_{12}^2,\ P_{11}(10) = 0$$

$$\dot{P}_{12} = -Da\tfrac{(1-x_1)}{(1+x_2/\gamma)^2}\exp(\tfrac{x_2}{1+x_2/\gamma})P_{11} + (2 + \beta + Da(1 - B\tfrac{(1-x_1)}{(1+x_2/\gamma)^2})\exp(\tfrac{x_2}{1+x_2/\gamma}))P_{12}$$
$$+ BDa\exp(\tfrac{x_2}{1+x_2/\gamma})P_{22} + (R+W)^{-1}\beta^2 P_{12}P_{22},\ P_{12}(10) = 0$$

$$\dot{P}_{12} = -2Da\tfrac{(1-x_1)}{(1+x_2/\gamma)^2}\exp(\tfrac{x_2}{1+x_2/\gamma})P_{12} + 2(1 + \beta - BDa\tfrac{(1-x_1)}{(1+x_2/\gamma)^2})\exp(\tfrac{x_2}{1+x_2/\gamma}))P_{22}$$
$$+ (R+W)^{-1}\beta^2 P_{22}^2 - Q_{22},\ P_{22}(10) = S_{22}$$

ve

$$\dot{q}_1 = (1 + Da\exp(\tfrac{x_2}{1+x_2/\gamma}))q_1 + (BDa\exp(\tfrac{x_2}{1+x_2/\gamma}) + (R+W)^{-1}\beta^2 P_{12})q_2$$
$$+ (R+W)^{-1}R\beta u P_{12},\ q_1(10) = 0$$

$$\dot{q}_2 = -Da\tfrac{(1-x_1)}{(1+x_2/\gamma)^2}\exp(\tfrac{x_2}{1+x_2/\gamma})q_1 + (1 + \beta - BDa\tfrac{(1-x_1)}{(1+x_2/\gamma)^2})\exp(\tfrac{x_2}{1+x_2/\gamma}) + (R+W)^{-1}\beta^2 P_{22})q_2$$
$$+ (R+W)^{-1}R\beta u P_{22} - Q_{22}(x_2 - xd_2(t)),\ q_2(10) = S_{22}(x_2(10) - 1.13)$$

olarak belirlenir. Kazanç matrisi K(t) ve sürücü kazanç b(t) aşağıdaki gibi bulunur:

$$K(t) = -(R+W)^{-1}\beta\begin{bmatrix}P_{12} & P_{22}\end{bmatrix} = \begin{bmatrix}K_{11}(t) & K_{12}(t)\end{bmatrix}$$

$$b(t) = -(R+W)^{-1}(Ru + \beta q_2)$$

Artırımsal kontrol, adjoint ve Hamiltonian aşağıdaki gibidir:

$$\delta u(t) = K(t)\delta x + b(t) = -(R+W)^{-1}(\beta(P_{12}\delta x_1 + P_{22}\delta x_2) + Ru + \beta q_2)$$

$$\tilde{\lambda} = P\delta x + q = \begin{bmatrix}\tilde{\lambda}_1(t)\\ \tilde{\lambda}_2(t)\end{bmatrix} = \begin{bmatrix}P_{11}\delta x_1 + P_{12}\delta x_2 + q_1\\ P_{12}\delta x_1 + P_{22}\delta x_2 + q_2\end{bmatrix}$$

$$\tilde{H} = \tilde{L} + \tilde{\lambda}^T \begin{bmatrix} -(1 + Da\exp(\frac{x_2}{1+x_2/\gamma}))\delta x_1 + Da\frac{(1-x_1)}{(1+x_2/\gamma)}\exp(\frac{x_2}{1+x_2/\gamma})\delta x_2 \\ -BDa\exp(\frac{x_2}{1+x_2/\gamma})\delta x_1 - (1+\beta - BDa\frac{(1-x_1)}{(1+x_2/\gamma)}\exp(\frac{x_2}{1+x_2/\gamma}))\delta x_2 + \beta\delta u \end{bmatrix}$$

Burada,

$$\tilde{L}(\delta x, \delta u) = Q_{22}(x_2 - 1.33)\delta x_2 + R u \delta u + \tfrac{1}{2}[Q_{22}\delta x_2^2 + (R+W)\delta u^2]' \text{dir.}$$

Durağanlık şartı ise,

$\tilde{H}_{\delta u} = Ru + (R+W)^{-1}\delta u + \beta\tilde{\lambda}_2$ olarak bulunur.

Bir sonraki iterasyon için kontrol fonksiyonu,

$$u^{k+1}(t) = u^k(t) + \delta u^k(t) = u^k(t) + K^k(t)\delta x^k + b^k = u^k + K_{11}^k \delta x_1^k + K_{12}^k \delta x_2^k + b^k(t)$$

$$\delta x^k(t) = x^{k+1}(t) - x^k(t)$$

olarak hesaplanır. Kontrol sınırlama ağırlık matrisi güncelleme katsayısı,

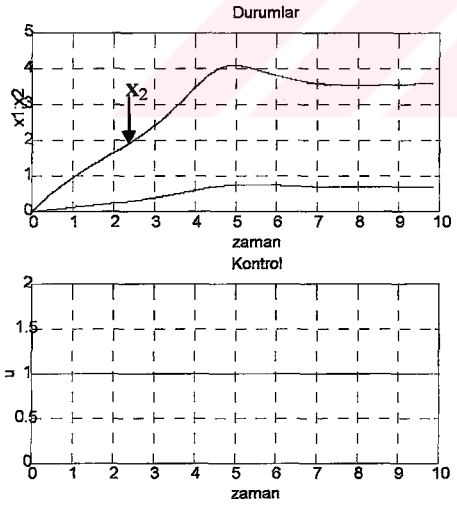
$$\alpha = \left| \frac{\frac{1}{2}S_{22}\delta x_2^2(10) + \int_0^5 \frac{1}{2}(Q_{22}\delta x_2^2 + R\delta u^2)dt}{Q_{22}(x_2(10) - 1.13)\delta x_2(10) + \int_0^5 (Q_{22}(x_2(t) - xd_2(t))\delta x_2 + Ru\delta u)dt} \right| \text{ olup,}$$

$W^{k+1} = \alpha^k W^k$ olarak güncellenir.

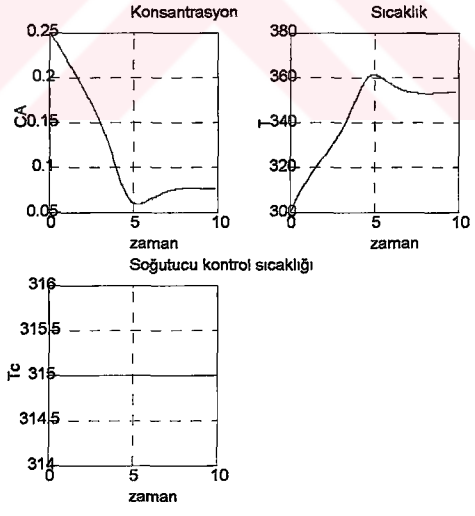
Bir adım sonra bulunacak yörüngenin hesabı için, Bölüm 3'te anlatıldığı gibi geri-besleme formunda çözüm yapılmıştır. Yani,

$\dot{x} = f(x, u^k + K^k(x - x^k))$, $x(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ ifadesi çözülerek $x^{k+1}(t)$ durum yörüngesi elde edilir ki buradan δx^k ve δu^k ifadeleri kolayca bulunabilir.

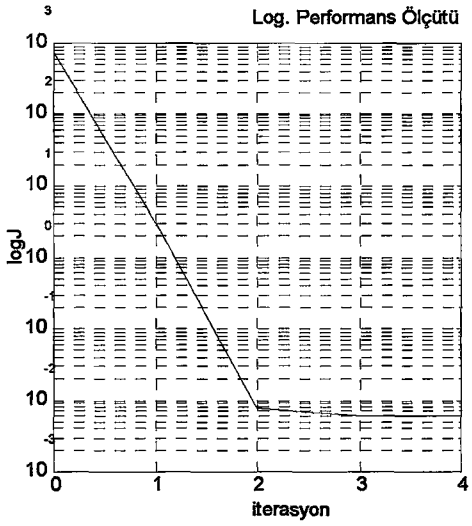
Algoritmanın işleyiş şekli bir önceki örnekteki gibidir. Burada, $u^0(t)=1.0$, $\gamma = 5.10^{-2}$ ve $\gamma_w = 5.10^{-2}$ alınmıştır. Konjuge nokta problemine İDDÖAY'de, PÖ'nün Q,R,S matrislerinin uygun seçilmesi nedeniyle karşılaşılmayacağından W=0 alınarak algoritma işletilmiştir. Sistemin başlangıç durumu, kontrol ve durum değişkenlerinin boyutsuz ve gerçek durumları Şekil 5. 51 ve Şekil 5. 52'da verilmiştir. Bu algoritma hızlı bir şekilde işlediğinden değişim adımları başlangıçta belirgindir. Şekil 5. 53'de PÖ'nün (yarı logaritmik) iterasyon adımıyla değişiminden bu sonuç açık bir şekilde görülmektedir. Şekil 5. 54'de ise kontrol fonksiyonunun iterasyona göre değişimi verilmiştir. Şekil 5. 55'da artırılmış Hamiltonian, Şekil 5. 56'de Riccati değişkenleri ve Şekil 5. 57'de ise kazanç matris ve vektör değişkenlerinin optimal noktadaki değişimleri görülmektedir. Şekil 5. 58 ve Şekil 5. 59'te sırayla, optimal noktadaki boyutsuz ve gerçek durum ve kontrol yörüngeleri görülmektedir. Şekil 5. 58'da istenen ve optimal sıcaklık yörüngesi arasındaki fark edilemeyecek kadar küçüktür. Görüldüğü gibi reaktör sıcaklığı kısa zamanda herhangi bir salınım ve aşma göstermeden istenilen değere oturmaktadır.



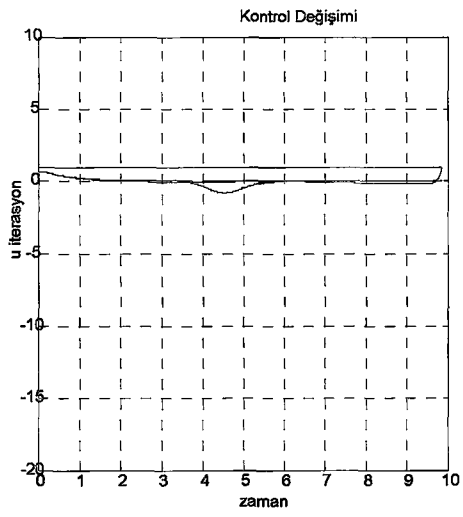
Şekil 5. 51 İDDÖAY'de $u^0(t)=1.0$ için CSTR sisteminin cevabı (boyutsuz)



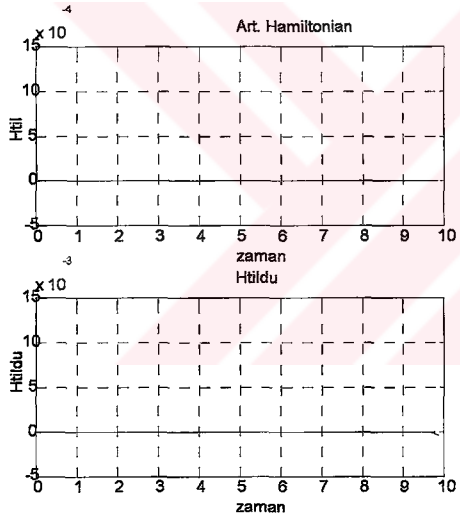
Şekil 5. 52 İDDÖAY'de $T_c=315^0K$ için CSTR sisteminin cevabı (gerçek değişkenler)



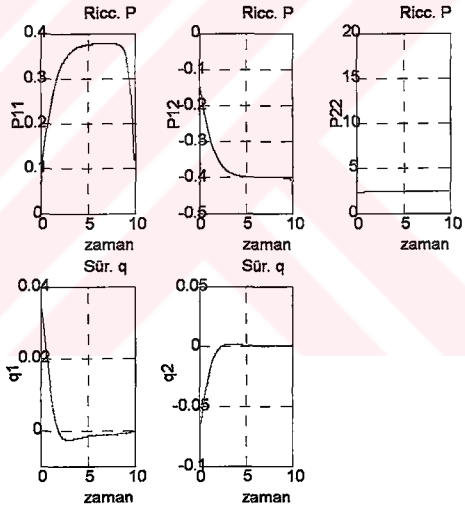
Şekil 5. 53 İDDÖAY için PÖ'nün iterasyonla değişimi (Yarı logaritmik)



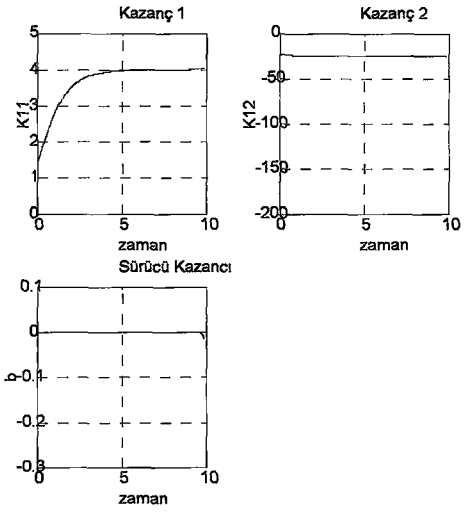
Şekil 5. 54 İDDÖAY ile elde edilen kontrol fonksiyonunun iterasyonla değişimi



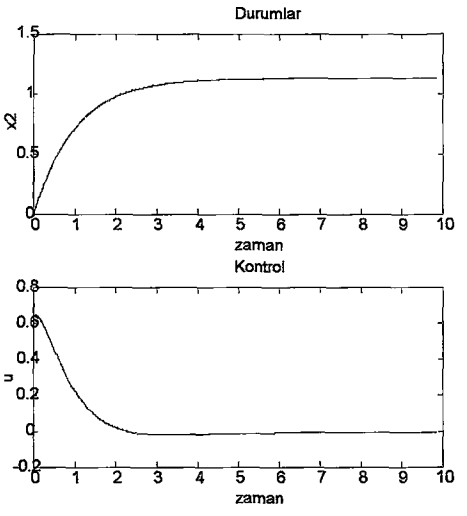
Şekil 5. 55 İDDÖAY için optimal noktadaki artırılmış Hamiltonian ve $\widetilde{H}_{\delta u}$ 'nun değişimi



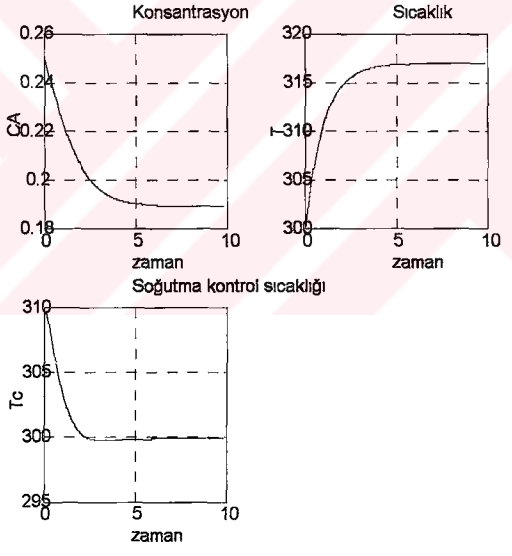
Şekil 5. 56 İDDÖAY için optimal noktadaki Riccati matrisi ve sürücü vektörün değişimleri



Şekil 5. 57 İDDÖAY için optimal noktadaki kazanç ve sürücü kazancın değişimleri



Şekil 5. 58 İDDÖAY sonucunda bulunan optimal sıcaklık durum ve kontrol yörüngelerinin değişimleri (boyutsuz, sürekli çizgi istenen yörünge, kesikli çizgi optimal yörünge)

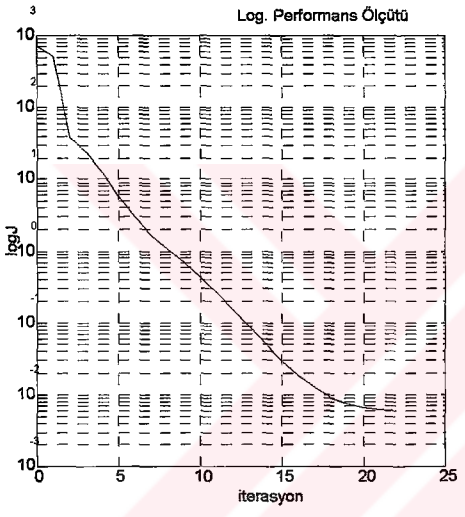


Şekil 5. 59 İDDÖAY sonucunda bulunan optimal durum ve kontrol yörüngelerinin değişimleri (gerçek değişkenler)

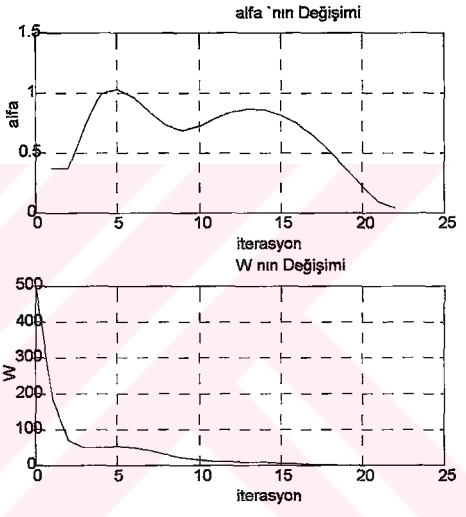
ii) İDHY ile CSTR probleminin çözümü;

Bu yöntem ile çözüm de bir önceki örnekte (VDP) olduğu gibi çözümlenebilir. Burada detaylı formüllere girilmeyecek, sadece sonuçları verilecektir. CSTR sistemi lineer

olmayan bir yapıda olduğundan kararsız davranışlar gösterdiği incelenmişti. Bu yöntemde, kontrol sınırlama ağırlık matrisi kullanmaksızın algoritmayı işletmek mümkün olmamaktadır. Çünkü konjuge nokta problemiyle burada da karşılaşılmaktadır. Bu yüzden, $W=5.10^2$ alınarak algoritma işletilmiştir. Şekil 5. 60'ta PÖ'nün (yarı logaritmik) değişimi verilmiştir. Bu, Şekil 5. 53 ile karşılaştırıldığında daha yavaş olduğu kolayca görülebilir. Özellikle başlangıca yakın yerlerde ilerleme yavaştır. Şekil 5. 61' de, α ve W'nın değişimleri verilmiştir. Görüldüğü gibi W'nın arttığı ve azaldığı değerler olup sonuçta sıfıra $(W \rightarrow 0)$ gitmektedir. Optimal noktadaki sonuçlar ise diğer yöntemde verildiği gibidir.



Şekil 5. 60 İDHY için PÖ'nün değişimi (yarı logaritmik)



Şekil 5. 61 İDHY için α ve W'nın iterasyonla değişimleri $(W^0=5.10^2)$

5.2.2. Optimal geri besleme (OGB) kanununun CSTR sistemi üzerinde değişik bozucular için etkinliğinin gösterilmesi

Burada da üç çeşit bozucu durum düşünülmüştür. Birincisi, sistem parametrelerinin değişmesi durumu, ikincisi, başlangıç durumunun değişmesi, üçüncüsü ise hem sistem parametrelerinin değişmesi hem de sisteme bir anda uygulandığı düşünülen basamak bozması durumlarıdır. OBG denemeleri için kullanılacak tüm parametre ve durumlar başlangıçta verildiği gibidir.

i) Sistem parametrelerinin değişmesi durumu: CSTR için kabul ettiğimiz nominal

parametre değerlerinin değiştiğini (B=11.0, β=1.5, γ=20, Da=0.135) düşünerek sadece İBOK ve OGB durumları gösterilecektir. Sadece İBOK sisteme uygulandığında sistem durum yörüngeleri ve faz diyagramı ve Şekil 5. 62 ve Şekil 5. 63'de verildiği gibidir. Görüldüğü gibi sistem sıcaklık yörüngesi oldukça gerçek yörüngesinden saparak farklı bir kararlı durum sıcaklığına oturmuştur. Bu durum faz diyagramından da görülebilir. Sadece İBOK ve OGB yapısı uygulanırsa sistem sıcaklık yörüngeleri Şekil 5. 64'deki gibi, ve OGB durumu için faz diyagramı Şekil 5. 65'te görüldüğü gibidir. Bu örnekte t_f=40 alınarak simülasyon sonuçlarının daha iyi görünmesi sağlanmıştır.

ii) Başlangıç koşullarının değişmesi durumu: Burada $x(0)=[0.6\ -4]^T$ alınarak yani $[C_A(0)\ T(0)]^T=[0.1\ 240]^T$ olarak sistem cevabı gözlemlenmiştir. Gerçekte $x(0)=[0\ 0]^T$ idi. Bu durumda sadece İBOK, CSTR sistemine uygulandığında durum yörüngeleri ve faz değişimi Şekil 5. 66 ve Şekil 5. 67'deki sonuçlar elde edilir. Buradan reaktör sıcaklığının istenilen sıcaklığa çok uzun zaman sonra ulaşacağı görülmektedir. Sisteme sadece İBOK ve OGB kanunu uygulanırsa sıcaklık değişimleri Şekil 5. 68'daki gibi, ve OGB durumu için faz değişimi ise Şekil 5. 69 'te görüldüğü gibi reaktör sıcaklığı çok kısa zamanda istenilen sıcaklığa ulaştırılıp orada tutulmaktadır. Bu örnekte ise t_f=5 alınmıştır.

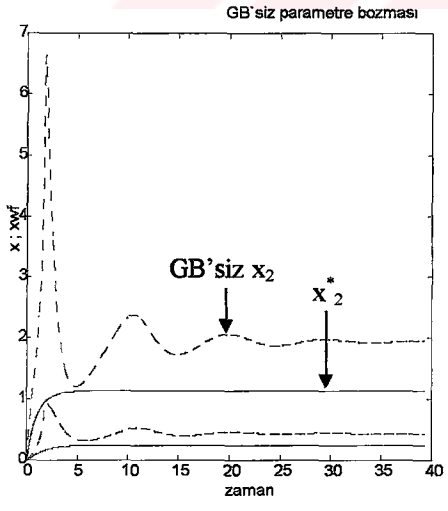
iii) Hem parametrelerin değişmesi, hem de basamak bozması durumu: Bu uygulamada iki basamak bozması durumu incelenecektir. Parametrelerin (i)'de verildiği gibi değiştiği düşünülecektir. Reaktörü besleyen giriş besleme sıcaklığı normalde 300^0K'de gelmektedir. Ancak t_1=40'da bu sıcaklığın 5^0K arttığı düşünülecektir. Burada t_f=100 olarak alınmıştır. Bu durumda sadece İBOK sisteme uygulanırsa yörünge ve faz değişimleri Şekil 5. 70 ve Şekil 5. 71'deki gibi bir değişime uğrar. Görüldüğü gibi t_1=40'dan itibaren sistem reaktör sıcaklığı salınım yapmaya ve büyük sapmalar göstermeye başlamıştır. Eğer sisteme sadece İBOK ve OGB kanunu uygulanırsa, reaktör sıcaklık değişimi Şekil 5. 72'deki gibi ve OGB durumu için faz değişimi Şekil 5. 73'deki gibi oldukça iyi bir şekilde salınımsız olarak görülür. Buradan parametre ve basamak bozucu duyarlılığı çok yüksek olan reaktörün, başarılı bir şekilde OGB ile kararlı halde tutulduğu gösterilmiştir. İkinci olarak, besleme sıcaklığı T_f'in başlangıçta 310^0K'de olduğu düşünülerek t_1=40'da 5^0K'lik düşüş yaptığı varsayılmıştır. Bu durumda sadece İBOK uygulanması sonucunda durum ve

faz değişimleri Şekil 5. 74 ve Şekil 5. 75'de verilmektedir. 5^0K'lik düşüşün olduğu andan itibaren sistem reaktör sıcaklığı kararsız limit döngü içine girmiştir. Bu ise hiç istenmeyen bir durumdur. Eğer OGB yapısı ve sadece İBOK durumları için sistem sıcaklık durum ve OGB'li faz cevapları Şekil 5. 76 ve Şekil 5. 77'deki gibi olur. Açıkça görüldüğü gibi reaktör sıcaklığı iyi bir şekilde istenilen değer civarında salınımsız olarak tutulmaktadır.

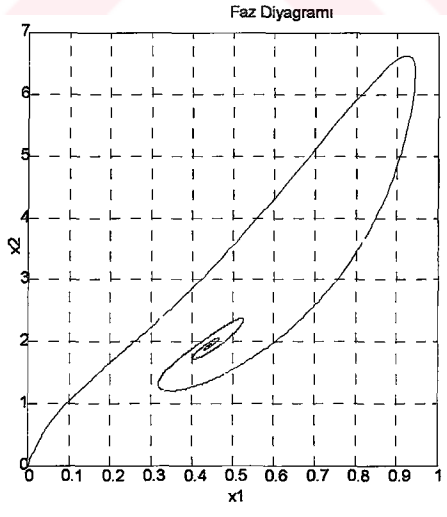
Bu uygulamaların dışında, değişik bir uygulama yapılacaktır. Yine parametrelerin (i)'deki gibi değiştiği kabul edilmiştir. Besleme sıcaklığında meydana gelebilecek büyük değişimlerin etkisi göz önüne alınacaktır. Besleme sıcaklığının, başlangıçtan t=10'a kadar 330^0K, t=20'ye kadar 280^0K ve t=40'a kadar 305^0K olarak değiştiği kabul edilerek GB'nin etkisi incelenecektir. Bu uygulamada t_f=40 seçilmiştir. Yani uygulanan sıcaklık bozması aşağıdaki şekildedir:

$$d(t) = \begin{cases} 2 & 0 < t \le 10 \\ -4/3 & 10 < t \le 20 \\ 1/3 & t > 20 \end{cases}$$

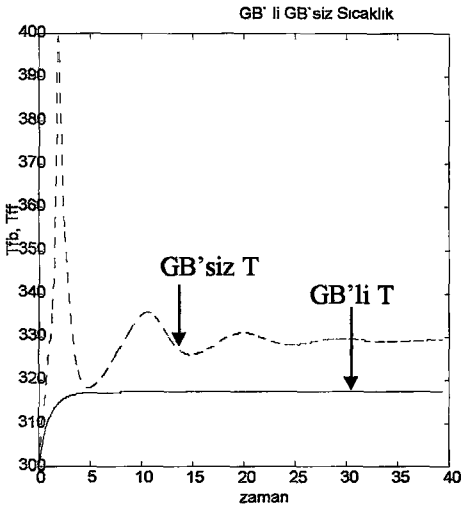
Bu durumda, Şekil 5. 78'de GB'li ve GB'siz sıcaklık yörüngeleri gösterilmektedir. Şekil 5. 79 GB'li ve Şekil 5. 80'de ise GB'siz durumlara ilişkin faz diyagramı verilmektedir.



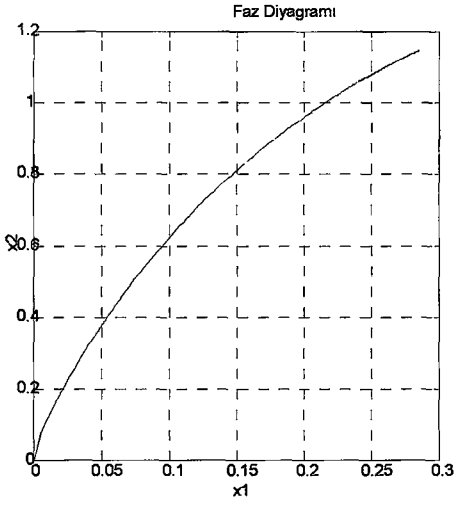
Şekil 5. 62 Sadece İBOK uygulanması sonucunda ve parametre bozması durumunda CSTR sisteminin yörünge değişimleri (boyutsuz, kesikli çizgiler GB'siz durum, B=11.0, β=1.5, γ=20, Da=0.135)



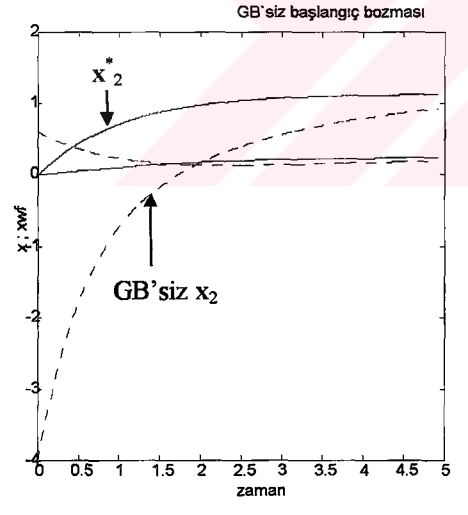
Şekil 5. 63 Sadece İBOK uygulanması sonucunda ve parametre bozması durumunda CSTR sisteminin faz değişimi (B=11.0, β=1.5, γ=20, Da=0.135)



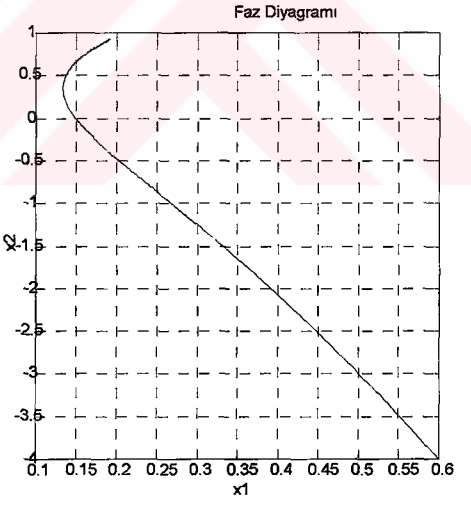
Şekil 5. 64 Sadece İBOK ve OGB uygulanması sonucunda ve parametre bozması durumunda CSTR sisteminin sıcaklık yörünge değişimleri (gerçek değişkenler, kesikli çizgiler GB'siz durum, B=11.0, β=1.5, γ=20, Da=0.135)



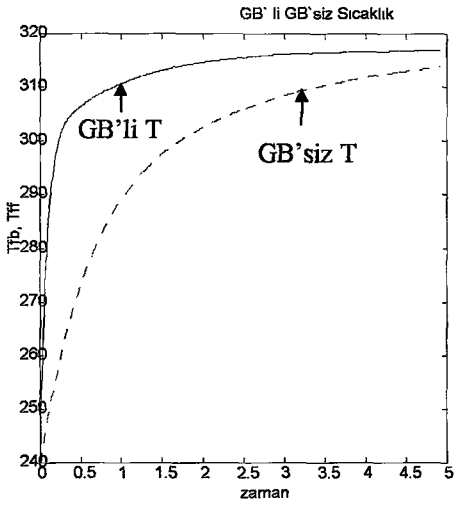
Şekil 5. 65 İBOK ve OGB uygulanması durumunda ve parametre bozması durumunda CSTR sisteminin faz değişimi (boyutsuz, B=11.0, β=1.5, γ=20, Da=0.135)



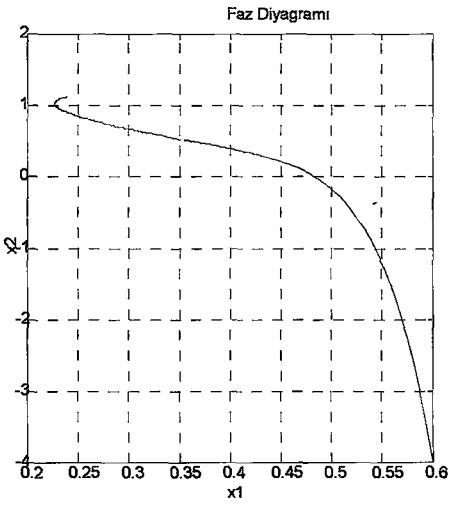
Şekil 5. 66 Sadece İBOK uygulanması sonucunda ve bozucu olarak $x(0)=[0.6\ -4]^T$ için CSTR sisteminin yörünge değişimleri (boyutsuz, kesikli çizgiler GB'siz durum, B=7.0, β=0.5, γ=20, Da=0.11)



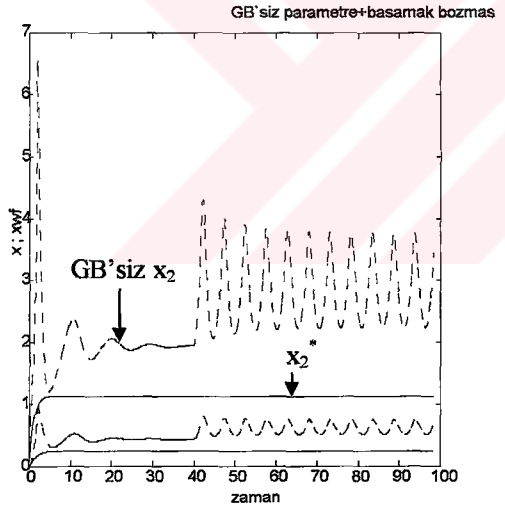
Şekil 5. 67 Sadece İBOK uygulanması sonucunda ve bozucu olarak $x(0)=[0.6\ -4]^T$ için CSTR sisteminin faz değişimi (boyutsuz, B=7.0, β=0.5, γ=20, Da=0.11)



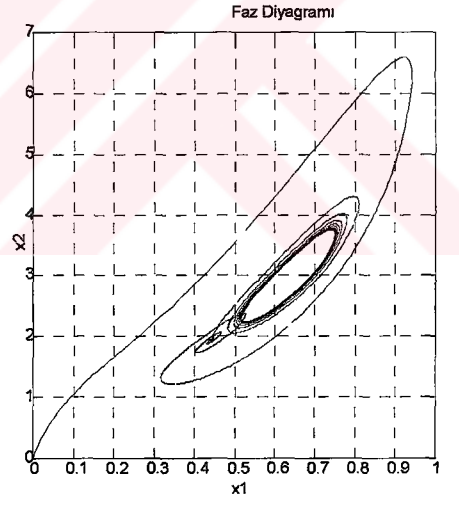
Şekil 5. 68 Sadece İBOK uygulanması sonucunda ve bozucu olarak $[C_A(0)\ T(0)]^T=[0.1\ 240]^T$ için CSTR sisteminin sıcaklık yörünge değişimleri (gerçek değişkenler, kesikli çizgiler GB'siz durum, B=7.0, β=0.5, γ=20, Da=0.11)



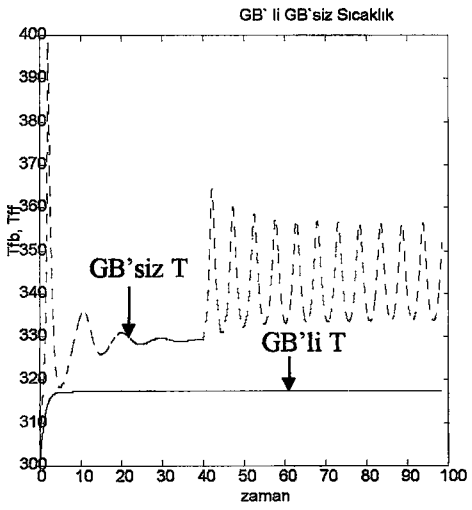
Şekil 5. 69 İBOK ve OGB uygulanması durumunda ve bozucu olarak $x(0)=[0.6\ -4]^T$ için CSTR sisteminin faz değişimi (boyutsuz, B=7.0, β=0.5, γ=20, Da=0.11)



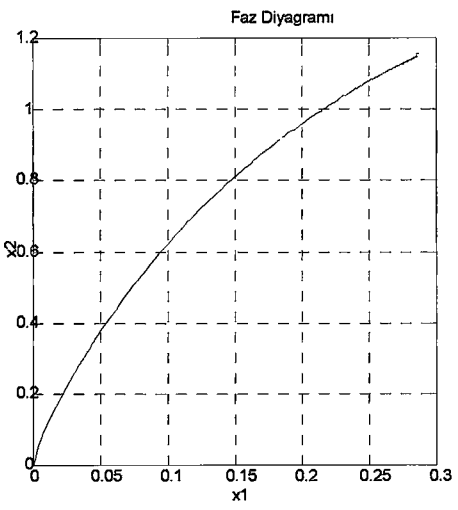
Şekil 5. 70 Sadece İBOK uygulanması durumunda hem parametre bozması ve hem de t_1=40'da T_f=305^{0}K olarak yapılan 5^{0}K basamak bozması durumunda CSTR sisteminin yörünge değişimleri (boyutsuz, kesikli çizgiler GB'siz durum, B=11.0, β=1.5, γ=20, Da=0.135)



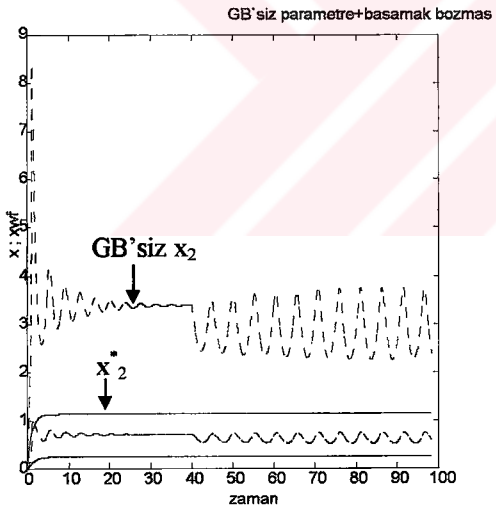
Şekil 5. 71 Sadece İBOK uygulanması durumunda hem parametre bozması ve hem de t_1=40'da T_f=305^{0}K olarak yapılan 5^{0}K basamak bozması durumunda CSTR sisteminin faz değişimi (boyutsuz, B=11.0, β=1.5, γ=20, Da=0.135)



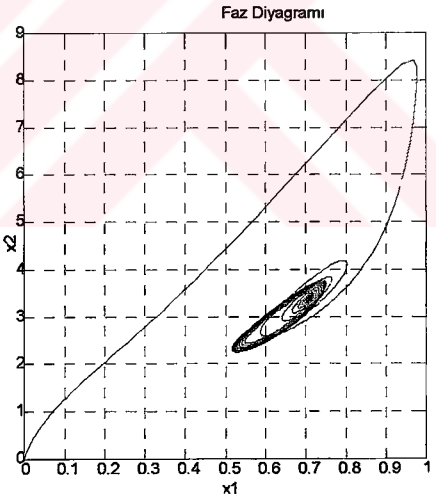
Şekil 5. 72 Sadece İBOK uygulanması durumunda hem parametre bozması ve hem de t_1=40'da T_f=305^{0}K olarak yapılan 5^{0}K basamak bozması durumunda CSTR sisteminin sıcaklık yörünge değişimleri (gerçek değişkenler, kesikli çizgiler GB'siz durum, B=11.0, β=1.5, γ=20, Da=0.135)



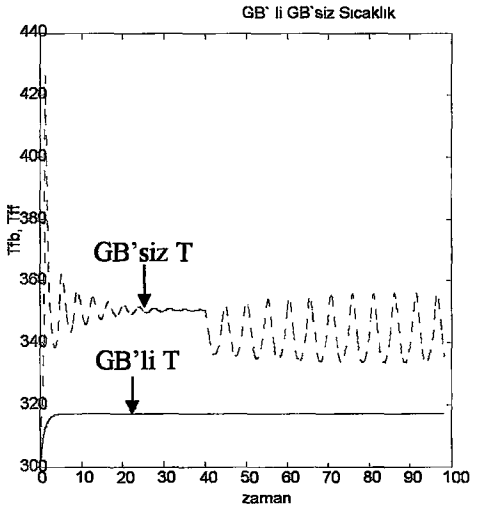
Şekil 5. 73 İBOK ve OGB uygulanması durumunda hem parametre bozması ve hem de t_1=40'da T_f=305^{0}K olarak yapılan 5^{0}K basamak bozması durumunda CSTR sisteminin faz değişimi (boyutsuz, B=11.0, β=1.5, γ=20, Da=0.135)



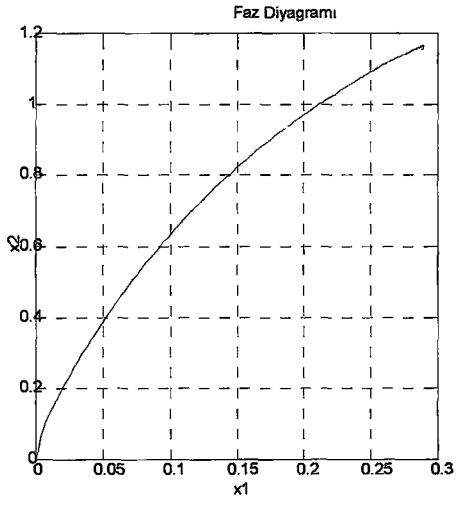
Şekil 5. 74 Sadece İBOK uygulanması durumunda hem parametre bozması ve hem de başlangıçta T_f=310^{0}K olarak bozularak t_1=40'da T_f=305^{0}K olarak 5^{0}K'lik düşmeyle basamak bozması durumunda CSTR sisteminin yörünge değişimleri (boyutsuz, kesikli çizgiler GB'siz durum, B=11.0, β=1.5, γ=20, Da=0.135)



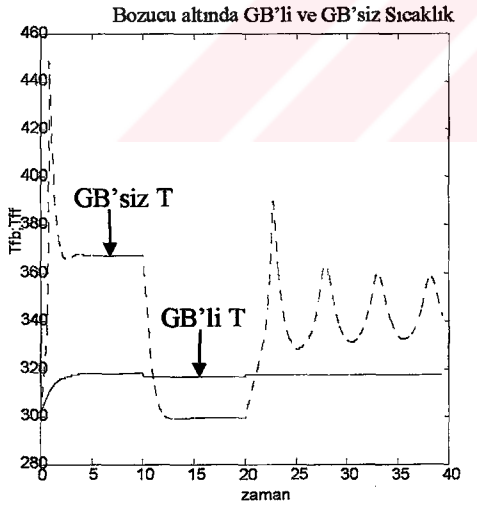
Şekil 5. 75 Sadece İBOK uygulanması durumunda hem parametre bozması ve hem de başlangıçta T_f=310^{0}K olarak bozularak t_1=40'da T_f=305^{0}K olarak 5^{0}K'lik düşmeyle basamak bozması durumunda CSTR sisteminin faz değişimi (boyutsuz, B=11.0, β=1.5, γ=20, Da=0.135)



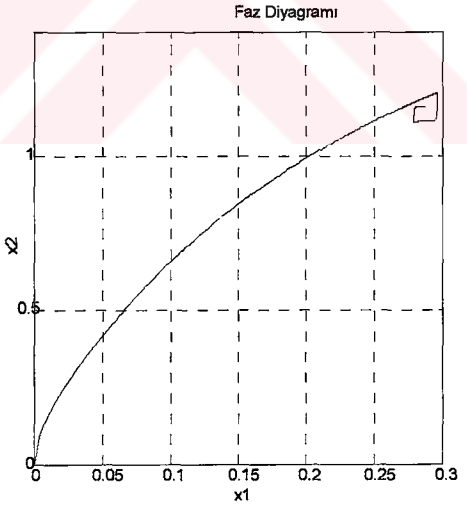
Şekil 5. 76 Sadece İBOK uygulanması durumunda hem parametre bozması ve hem de başlangıçta T_f=310^{0}K olarak bozularak t_1=40'da T_f=305^{0}K olarak 5^{0}K'lik düşmeyle basamak bozması durumunda CSTR sisteminin sıcaklık yörünge değişimleri (gerçek değişkenler, kesikli çizgiler GB'siz durum, B=11.0, β=1.5, γ=20, Da=0.135)



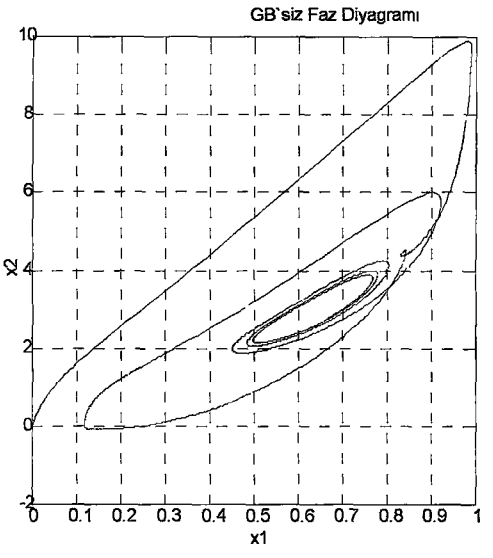
Şekil 5. 77 İBOK ve OGB uygulanması durumunda hem parametre bozması ve hem de başlangıçta T_f=310^{0}K olarak bozularak t_1=40'da T_f=305^{0}K olarak 5^{0}K'lik düşmeyle basamak bozması durumunda CSTR sisteminin faz değişimi (boyutsuz, B=11.0, β=1.5, γ=20, Da=0.135)



Şekil 5. 78 Sadece İBOK uygulanması durumunda hem parametre bozması ve hem de T_f'in değişik değerlerdeki bozma durumunda CSTR sisteminin sıcaklık yörünge değişimleri (gerçek değişkenler, kesikli çizgiler GB'siz durum, B=11.0, β=1.5, γ=20, Da=0.135)



Şekil 5. 79 İBOK ve OGB uygulanması durumunda hem parametre bozması ve hem de T_f'in değişik değerlerdeki bozma durumunda CSTR sisteminin faz değişimi (boyutsuz, B=11.0, β=1.5, γ=20, Da=0.135)



Şekil 5. 80 Sadece İBOK uygulanması durumunda hem parametre bozması ve hem de T_f'in değişik değerlerdeki bozma durumunda CSTR sisteminin faz değişimi (boyutsuz, B=11.0, β=1.5, γ=20, Da=0.135)

5.3. Sonuçlar

Önceki kısımda örnekler üzerinde gösterildiği gibi, OGB, bozucuların olumsuz etkilerini sistem üzerinden izole edip istenilen yörünge üzerinde tutmayı iyi bir şekilde yerine getirmektedir. Bu yüzden ikinci dereceden yöntemler hem optimal yörünge bulmak ve hem de OGB kanununun çıkarılması açısından çok önemli özelliklere sahiptir. Ayrıca, geri-beslemeli sistem her zaman kararlıdır.

İDDÖAY, İDHY'ne göre daha gürbüz (robust) bir yapıya sahiptir. İDDÖAY her adımda, ölçütün azaltılmasına çalışmaktadır. İDHY ise gerekli koşulları sağlamaya yönelik bir yapıdadır. Bu yüzden, İDHY örneklerde de gösterildiği gibi konjuge nokta problemi ile karşılaşmaktadır. Ancak, bu problem Bölüm 3'te verildiği gibi her iki yöntem için W'nın etkisiyle ortadan kaldırılabilmektedir. Özel olarak, lineer olmayan kuadratik (LOK) problem için İDDÖAY, tüm şartları sağladığı için konjuge nokta problemi ile karşılaşmaz. Ancak İDHY'nde bu garanti edilemez.

Birinci derece yöntem, ikinci derece yöntemlere göre problemi çözme zamanı

açısından çok yavaştır. Bu yöntemlerin yakınsama, konjuge noktaya takılma, hesap karmaşıklığı ve göreceli olarak problemi çözme zamanı kazancı açısından karşılaştırılması Tablo 5. 1'de verilmiştir. Tabloda gösterilen problemi çözme zamanı kazancı şu şekilde belirlenmiştir. Yöntemlerin problemi çözme iterasyon sayısı **nk** ve yöntemler için bir iterasyondaki çözülmesi gereken diferansiyel denklem sayısı ise **nd** olsun. Bu durumda, problemi çözme zamanı T, **(nk.nd)** ile orantılı olacaktır. Yani, $\mathbf{T \propto (nk.nd)}$ olarak gösterilebilir. Bir diferansiyel denklemin çözüm zamanı τ_1 olarak alınıp tüm yöntemler için yaklaşık aynı olduğu varsayılmaktadır. Bu durumda, problemin çözme zamanı, $\mathbf{T = \tau_1.(nk.nd)}$ olur. Yöntemlerin performans karşılaştırması için BDGY temel alınarak, göreceli problemi çözme zamanı kazancı (α_T) belirlenecektir ki $\alpha_T = \dfrac{\tau_1.(\mathbf{nk.nd})_{(\text{diğer yönt.})}}{\tau_1.(\mathbf{nk.nd})_{(\text{BDGY})}} = \dfrac{(\mathbf{nk.nd})_{(\text{diğer yönt.})}}{(\mathbf{nk.nd})_{(\text{BDGY})}}$ olarak bulunur. Tablodan da görüldüğü gibi bu oran, BDGY için 1 olup, diğer yöntemler için 1'de küçüktür. En iyi yöntem olarak, bu oranın sıfıra en yakını olan İDDÖAY'dir. Tablo 5. 1'de verilen sonuçlar, (5.1.1)'de verilen problemin çözümünden elde edilmiştir. Bu problemde, kontrol fonksiyonu için bağıl kontrol hatasının 5.10^{-3}'ten küçük olması tüm yöntemler için kullanılmıştır.

Tablo 5. 1 Optimal kontrol yöntemlerinin karşılaştırılması

	Gradyan Yöntemi (BDGY)	**Geliştirilen Doğrudan Azaltma Yöntemi (İDDÖAY)**	**Geliştirilen Hamiltonian Yöntemi (W'lı İDHY)**	**Klasik İkinci Derece Hamiltonian Yöntemi (İDHY, W=0)**
Yakınsama	Çok yavaş	Çok hızlı	Hızlı	Hızlı
Konjuge Noktaya Takılma	Yok	Yok	Yok	Olabilir
Hesap Karmaşıklığı (1 iterasyonda)	n zamanda ileri n zamanda geri Toplam 2n	n zamanda ileri n+n(n+1)/2 geri Top. 2n+n(n+1)/2	n zamanda ileri 2n+n(n+1)/2 geri Top. 3n+n(n+1)/2	n zamanda ileri 2n+n(n+1)/2 geri Top. 3n+n(n+1)/2
Göreceli Problemi Çözme Zamanı Kazancı (α_T)	1	0.112	0.16	0.16 (Konjuge nokta yoksa)

BÖLÜM 6. DİNAMİK SİNİR AĞLARI (DSA) ve YAPISI

Yapay sinir ağlarının (YSA) dinamik bir türü olan dinamik sinir ağları (DSA) geniş kapasiteli bir yapıya sahip dinamik bir sistemdir. Cebirsel sinir ağları (CSA) da çok geniş uygulama alanına sahiptir. Bu konuda literatürde bir çok çalışma bulmak mümkündür [51-64]. Bu çalışmada, sürekli zamanlı DSA yapısı ve optimal kontrol uygulaması incelenmiştir. Kullanılan DSA modeli Hopfield [51-53,97,98,113] tipine benzer yapıdadır.

YSA, modelleme ve kontrolde son zamanlarda yaygın bir şekilde kullanılmaktadır. CSA'nın iç dinamik eksikliği yüzünden ve sistemlerin dinamik yapılara sahip olmasından dolayı, modeli direkt olarak yakalaması mümkün olamaz. Bu problem için, geri-beslemeli CSA yapıları geliştirilmiştir [114-116]. Ancak burada da, ağın büyük olması, parametre (ağırlık) sayısının çok fazla olması ve eğitim zamanının uzun olması problemleriyle karşılaşılır.

DSA ise, dinamik elemanlara sahip olduğu için, bu problemlerin üstesinden kolayca gelmektedir. DSA, sistem dinamiklerini büyük oranda yakalamakta ve yüksek depolama kapasitesine sahip olmaktadır.

Eğitim yapısı olarak, öğreticili eğitim yapısı kullanılmıştır. Burada, giriş-çıkış verileri kullanılarak, ağ parametrelerinin ayarlanması yapılmaktadır. İleri beslemeli cebirsel ağlarda, ağ çıkış hatasının ağırlıklara göre gradyanlarını hesaplamak kolaydır. Ancak, DSA'nda gradyan hesabı daha karmaşıktır. Fakat, bu çalışmada gradyan hesabı, sistem teorisinden bilinen dinamik sistem analizi, kontrol ve tanıma uygulamaları kullanılarak kolayca yapılmıştır. 1980'li yılların başında, Hoppfield'in yapmış olduğu birleşik hafıza çalışmaları, sürekli-zamanlı DSA konusunda temel olmuş ve değişik yapıları kullanılmıştır.

6.1. Dinamik Sinir Ağları ve Yapısı

Dinamik sinir ağı (DSA), sınırlandırılmamış bağlantılı dinamik elemanlara sahip bir işleme birimleri olarak düşünülür. Bu yapıda DSA, ileri ve geri-beslemeli bağlantılardan ve lineer olmayan çıkış fonksiyonundan meydana gelmektedir. Her nöron, bir dinamik hücreye sahiptir. Dinamik hücre, birinci dereceden zaman sabiti olan bir yapıdadır. Bir DSA yapısı Şekil 6. 1'de gösterilmektedir. Orada, zamanla değişen l adet giriş işareti ve n adet ise nöron birimi bulunmaktadır. Her nöron birimi, kendilerinden ve diğer tüm birimlerden giriş almaktadırlar. Birimin çıkışı y_i, durum değişkeni x_i'nin bir sigmoidsel fonksiyonu olarak $h(x_i)$'dir. Ağ çıkışı z_i, m adet olup, birim çıkışlarının lineer ağırlıklı toplamlarıdır. p_{ij} ağırlıkları, i. nörona j. girişten gelen bağlantıları, w_{ij} iç nöron bağlantıları olup i. nörona j. nörondan gelen bağlantıları, q_{ij} ise i. ağ çıkışına j. nörondan gelen bağlantıları gösterir. T_i, i. nörona ait dinamiğin zaman sabitini, b_i ise i. nörona ait kutuplama parametrelerini tanımlar.

Hesaplama modeli aşağıdaki denklemlerle özetlenebilir:

$$z_i = \sum_{j=1}^{n} q_{ij} y_j \tag{6.1}$$

$$y_i = h(x_i) = \frac{1}{1+\exp(-x_i)} \tag{6.2}$$

$$T_i \dot{x}_i = -x_i + \sum_{j=1}^{n} w_{ij} y_j + \sum_{j=1}^{l} p_{ij} u_j + b_i \quad , \quad x_i(0) = x_{i0} \tag{6.3}$$

Dinamik yapıdan dolayı başlangıç koşulları durum değişkenleri için belirlenmelidir. Yukarıdaki eşitlikler matris-vektör notasyonunda aşağıdaki gibi yazılabilir:

$$z_{(mx1)} = q_{(mxn)} y_{(nx1)} \qquad \text{,ağ çıkışı} \tag{6.4}$$

$$y_{(nx1)} = s(\tilde{x}_{(nxn)}) = \mathrm{diag}[(I+\exp(-\tilde{x}_{(nxn)}))^{-1}] \qquad \text{,aktivasyon çıkışı} \tag{6.5}$$

$$T_{(nxn)} \dot{x} = -x_{(nx1)} + w_{(nxn)} y_{(nx1)} + p_{(nx1)} u_{(lx1)} + b_{(nx1)}, \quad x(0) = x_0 \text{ ,dinamik yapı} \tag{6.6}$$

Burada $\tilde{x}$, x vektörünü diyagonali üzerinde bulunduran matristir. Parantez içindeki indisler, değişkenlerin boyutlarını göstermektedir. (6.5) ifadesindeki diag(.) diyagonali ifade eder. T'nin diyagonal elemanları ise dinamiğin zaman sabitlerini içerir. Aktivasyon fonksiyonu sigmoid olarak seçilmiştir. Başka çeşit aktivasyon fonksiyonları da kullanılmaktadır. Bu çalışmada, sigmoig fonksiyonunun daha genel hali olan aşağıdaki yapı kullanılmıştır:

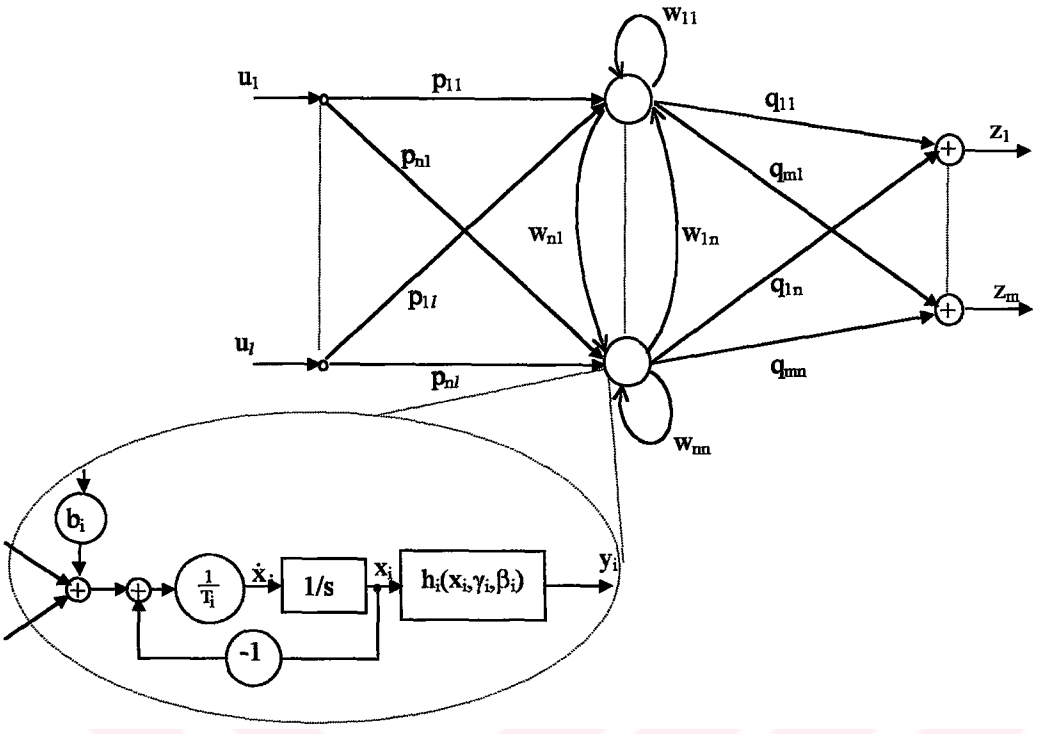
$$y_i = h_i(x_i, \gamma_i, \beta_i) = \frac{1}{1+\exp[-(\gamma_i x_i + \beta_i)]} \tag{6.7}$$

Burada γ_i, sigmoid fonksiyonunun lineersizliğini belirleyen parametre, β_i ise orijinden olan kaymayı veya kutuplama değerini belirleyen parametredir. Şekil 6. 2'de değişik γ_i ve β_i değerleri için genel sigmoid fonksiyonunun değişimlerini göstermektedir.

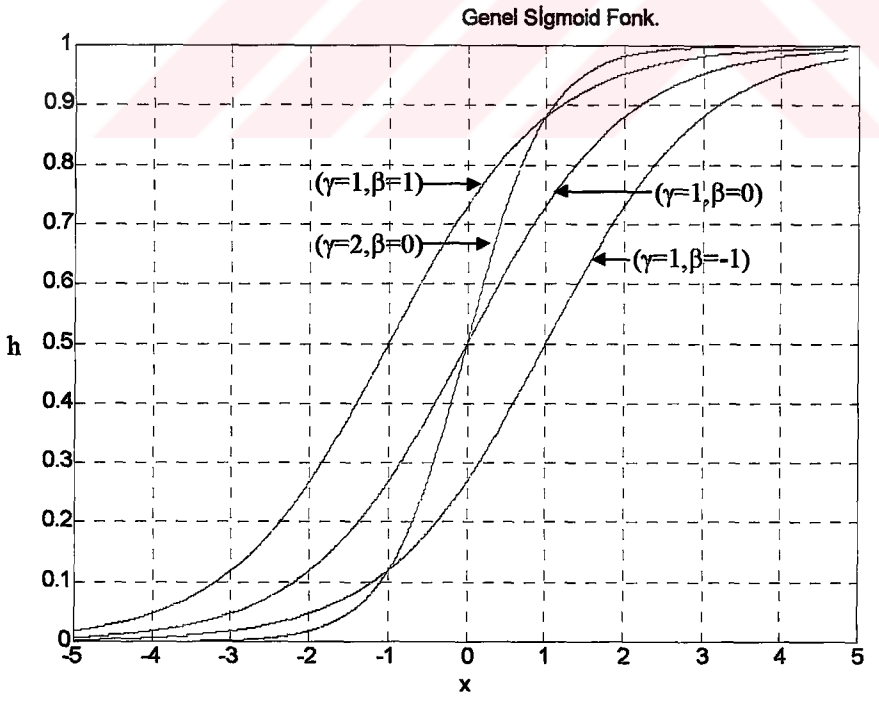
Bu çalışmada, sigmoid fonksiyonuna eklenen iki parametreyle beraber, toplam 7 tane ana parametre DSA için parametre (ağırlık) olarak kullanılıp güncelleme yapılacaktır. Bütün parametrelerin aktif hale getirilip eğitim esnasında güncellenebilmesi, sinir ağı için çok büyük bir kapasite ve genelleştirme imkanı vermektedir. Bu durumda toplam parametre sayısı $\mathbf{n^2+4n+nl+mn}$ ile bulunabilir.

Şekil 6. 1'de verilen DSA modelinin çıkışı, parametreleri, başlangıç koşulları ve giriş yörüngesi verildiğinde nümerik yöntemler kullanılarak t=0'dan t_f'ye kadar belirlenebilir. Durum denklemlerinin çözümünde Bölüm 3'te de bahsedilen 5. dereceden Runga-Kutta-Butcher metodu kullanılmıştır. Bu yüzden, DSA'ın zaman sabiti için bir alt sınır konulmuştur.

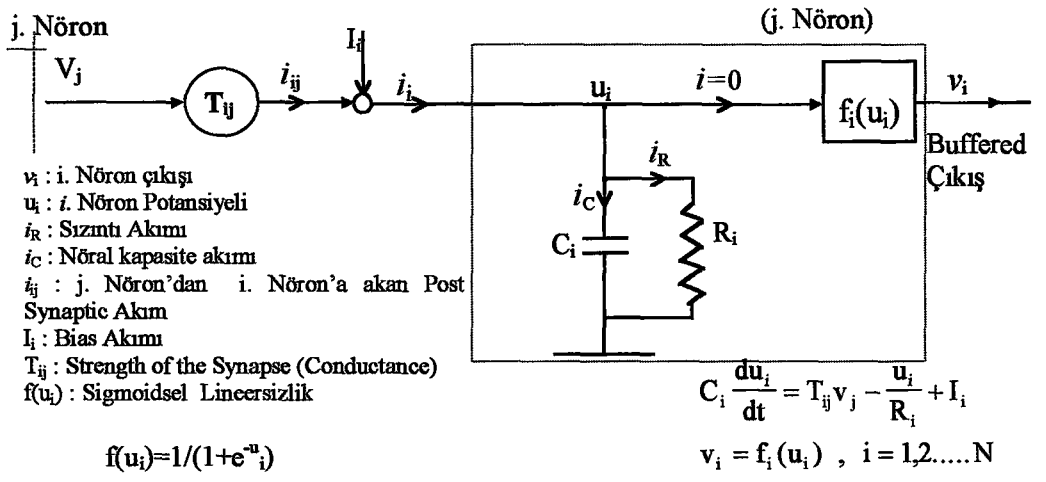
Kullanılan bu DSA modeli, Hoppfield'in modeline benzer. Hoppfield, biyolojik nöron yaklaşımından bu modeli oluşturmuştur. Hoppfield modeli Şekil 6. 3'te verilmiştir.



Şekil 6. 1 DSA için kullanılan model yapısı



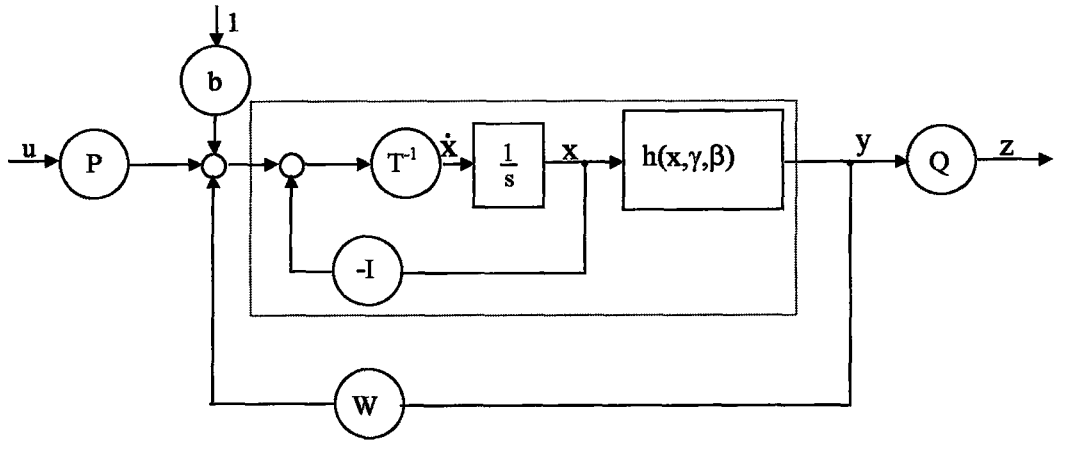
Şekil 6. 2 Genel sigmoid fonksiyonunun parametrelerine göre değişimi



Şekil 6. 3 Hoppfield'in biyolojik nöron yaklaşımından dinamik nöron modeli

Görüldüğü gibi Hoppfield modeli ile Şekil 6. 1'de verilen DSA modeli, notasyon farkıyla birbirine benzer. Bir çok araştırmacı, Hoppfield modeline dayalı modeller geliştirmişlerdir.

DSA modelini daha genel bir yapıda, durum-uzayı gösterimiyle ifade etmek mümkündür [113]. Şekil 6. 4'te genelleştirilmiş nöral-dinamik yapısı verilmiştir. Bu yapı, (6.4)-(6.7) eşitliklerindeki matris-vektör ifadelerinin grafik olarak çizilmesidir. Görüldüğü gibi nöron dinamiği yarı-lineer dinamik bir sistemdir. Lineersizliği veren ifade ise cebirseldir.



Şekil 6. 4 Genelleştirilmiş DSA model yapısı

Bu konuda kullanılan W,P,Q,b parametreleri bir önceki bölümlerde optimal kontrol için kullanılan parametrelerden farklı olup karıştırılmamalıdır. Burada, dinamik yapı için genel ifade olarak,

$$\dot{x} = f(x,p,u) \quad , x(0) = x_0 \tag{6.8}$$

olarak yazılabilir. Buradaki **p**, genel anlamda tüm parametre vektörüdür.

Bu tez çalışmasında, DSA için yörünge takip edici problemi ele alınmıştır. Böylece, zeki bir sistem oluşturulmaya çalışılmıştır. Eğer istenilen yörünge veya yörüngeler $z^d(t)$ verilirse, DSA bu yörüngeleri izleyebilecek şekilde bir yapılanma içine girmesi, yani parametrelerini uygun bir şekilde bulunması problemi çözülmelidir. Burada, başlangıç koşulları ve giriş yörüngesinin verildiği düşünülmektedir. Bu problem için performans ölçütü (PÖ) kuadratik olarak seçilmiştir:

$$J = \frac{1}{2}\int_0^{t_f} [z(t) - z^d(t)]^T [z(t) - z^d(t)]dt = \int_0^{t_f} F(x,t)dt \tag{6.9}$$

Burada, $e(t)=z(t)-z^d(t)$ olarak hata fonksiyonudur. $z(t)$, DSA model cevabı (çıkışı), $z^d(t)$ istenilen sistem cevabıdır. (6.9) eşitliği ile verilen ölçüt fonksiyonuna göre, bu fonksiyonu parametrelere göre minimum yapmak, DSA eğitimini yapmak anlamına gelir. Basit olarak problem, parametre kestirimi problemiyle eşdeğerdir. F(.) fonksiyonu değişik yapılarda seçilebilir. Burada, iyi bir yapı olan kuadratik yapı seçilmiştir.

Bundan sonraki kısımlarda, gradyan tabanlı parametre optimizasyonu problemi dinamik sistemler için incelenecektir.

6.2. Dinamik Sinir Ağlarında PÖ Minimizasyonu

Bu çalışmada, problem lineer olmayan optimizasyon problemi olarak ele alınmıştır. Bunun için, gradyan tabanlı algoritmalar çözüm için kullanılmıştır. Parametre

optimizasyonu, YSA için literatürde kullanılan eğitim anlamında kullanılmaktadır. Kısaca, geliştirilecek algoritmada aşağıdaki adımlar kullanılacaktır:

i) Belirlenen başlangıç koşulları, parametre ve giriş yörüngesine göre, DSA çıkışı için gerekli simülasyonu yap. Bunun için **n** tane diferansiyel denklem çözülecektir.
ii) Verilen yörüngeye göre, PÖ gradyanını hesapla.
iii) Kullanılan optimizasyon yöntemine göre yeni parametreleri hesapla.

Bu adımlar, kullanılan sonlandırma kriterlerine (ölçütün yeterince küçültülmesi, parametre değişimindeki azalma vb.) göre tekrarlanır. Başlangıç parametreleri bu çalışmada rasgele üretilerek seçilmiştir.

6.2.1. DSA'da gradyan hesabı

Ölçüt fonksiyonunun tüm ağ parametrelerine göre gradyanının veya duyarlılığının hesabı, parametre güncellemek veya eğitim için gereklidir. DSA'ına ait 7 temel parametreye göre PÖ'nün gradyanları şöyle verilebilir:

$$\frac{\partial J}{\partial p_{ij}}, \frac{\partial J}{\partial w_{ij}}, \frac{\partial J}{\partial q_{ij}}, \frac{\partial J}{\partial T_i}, \frac{\partial J}{\partial b_i}, \frac{\partial J}{\partial \gamma_i}, \frac{\partial J}{\partial \beta_i} \tag{6.10}$$

Çıkış parametresi **q**'ya göre gradyan kolayca şöyle yazılabilir:

$$\frac{\partial J}{\partial q_{ij}} = \int_0^{t_f} [z_i(t) - z_i^d(t)] \frac{\partial z_i}{\partial q_{ij}} dt = \int_0^{t_f} e_i(t) y_j dt \tag{6.11}$$

Yukarıdaki ifadenin değerini bulmak için, DSA'ının simüle edilip z(t) ve $y_j(t)$ yörüngeleri bulunarak işlem tamamlanır. Diğer gradyanları hesaplayabilmek için, parametrelere bağlı lineer diferansiyel denklem setinin çözülmesi gerekmektedir. Bu hesaplama için, ileri duyarlılık analizi [53,97,98,113] (direkt yöntem) ve adjoint duyarlılık analizi [97,98,113] (endirekt yöntem) olmak üzere iki yöntem kullanılabilir. Bu tez çalışmasında, adjoint duyarlılık analizi üzerinde çalışılmış ve uygulanmıştır.

Adjoint yöntemi etkin bir yöntemdir. İleri duyarlılık hesabında $\mathbf{n^3+4n^2+n^2l}$ adet lineer diferansiyel denklem çözmek gerekir. Görüldüğü gibi, nöron sayısı arttıkça parametre sayısı da kübik bir şekilde artacağından, hesap maliyeti ileri duyarlılık analizinde yüksektir. Adjoint yönteminde yalnız **n** sayıda diferansiyel denklem çözmek yeterlidir.

6.2.1.1. Adjoint duyarlılık analizi ile gradyan hesabı

Optimal kontrol teorisinde kullanılan değişimlerin hesabı yardımıyla, parametre uzayında adjoint duyarlılık analizi kullanılacaktır. Bu metot, Lagrange çarpanı yardımıyla dinamik sınırlama ortadan kaldırılarak çözülebilir. Adjoint duyarlılık analizi için gerekli teorik çıkarımlar Ek C'de detaylı olarak verilmiştir. (6.9) ile verilen PÖ, (6.3) veya (6.6) eşitlikleriyle verilen dinamik sınırlama altında adjoint analiziyle aşağıdaki gibi yapılabilir:

Adjoint diferansiyel denklemi,

$$-\dot{\lambda}_i = -\frac{1}{T_i}\lambda_i + \frac{1}{T_i}\sum_j w_{ij} y_j' \lambda_j + e_i(t)\sum_j q_{ij} y_j' \quad , \quad \lambda_i(t_f) = 0 \tag{6.12}$$

ile bulunur. Burada,

$$y_j' = \gamma_j h_j (1 - h_j) \tag{6.13}$$

olarak verilir. Burada, adjoint sistemi dinamik sistemle birleştirilmiştir. Adjoint vektörünün büyüklüğü **n**'dir. Yani, DSA'daki nöron sayısı veya derecesi ile adjoint sistem derecesi aynıdır. Zamanda geriye doğru çözülerek bulunur. DSA'daki parametre sayısından bağımsızdır. Bu önemli bir sonuçtur. Duyarlılık hesabı için **n** adet diferansiyel denklem zamanda geriye doğru çözülerek bulunur.

Adjoint metot yardımıyla DSA iç parametre gradyanları aşağıdaki gibi hesaplanır:

$$\frac{\partial J}{\partial w_{ij}} = \int_0^{t_f} \frac{\lambda_i y_j}{T_i} dt \tag{6.14}$$

$$\frac{\partial J}{\partial p_{ij}} = \int_0^{t_f} \frac{\lambda_i u_j}{T_i} dt \tag{6.15}$$

$$\frac{\partial J}{\partial b_i} = \int_0^{t_f} \frac{\lambda_i}{T_i} dt \tag{6.16}$$

$$\frac{\partial J}{\partial T_i} = -\int_0^{t_f} \frac{\lambda_i}{T_i^2} [-x_i + \sum_{j=1}^{n} w_{ij} y_j + \sum_{j=1}^{1} p_{ij} u_j + b_i] dt \tag{6.17}$$

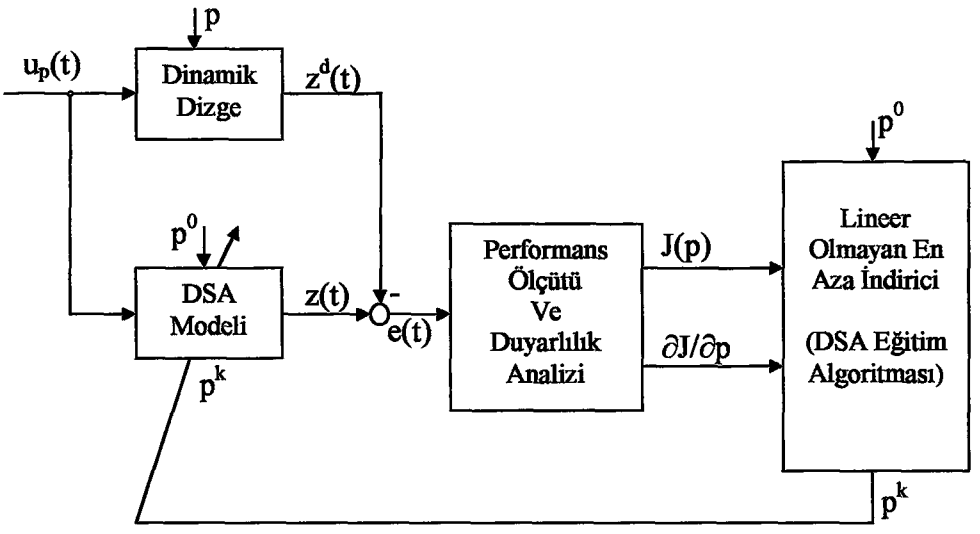
$$\frac{\partial J}{\partial \gamma_j} = \int_0^{t_f} (\sum_i \frac{\lambda_i}{T_i} w_{ij}) x_j h_j (1 - h_j) dt \tag{6.18}$$

$$\frac{\partial J}{\partial \beta_j} = \int_0^{t_f} (\sum_i \frac{\lambda_i}{T_i} w_{ij}) h_j (1 - h_j) dt \tag{6.19}$$

Görüldüğü gibi, parametre gradyanları adjoint değişkeni yardımıyla bulunabilmektedir. Bu yüzden, hesaplama yönünden az maliyet getirmektedir.

6.3. Gradyan Yöntemiyle DSA Eğitim Algoritmaları

YSA'nda eğitim, sistem teorisinde parametre kestirimi veya optimizasyonu anlamındadır. DSA'nda yukarıda anlatılan yapı ışığında, toplam parametre (p) vektörü sayısının **m** olduğu düşünülsün. İstenilen, verilen PÖ için DSA parametrelerini istenilen yörüngeyi tanıyacak şekilde bulmaktır. Bu yapıldığında PÖ en aza indirilmiş olacaktır. Yukarıda anlatıldığı gibi, PÖ olan J'nin DSA parametrelerine göre gradyanlarının hesaplandığı düşünülsün. Bundan sonra, aşağıda verilen optimizasyon algoritmalarından biri DSA'na uygulanabilir. Bir DSA modelinin, bir dinamik dizge için eğitilmesi Şekil 6. 5'te gösterilmektedir.



Şekil 6. 5 DSA sisteminin eğitim blok diyagramı

6.3.1. En hızlı azalan gradyan yöntemi (EAGY)

Bu yöntem basit, ancak etkili bir yöntemdir. Literatürde geriye-yayınım ismiyle kullanılır. İfadesi ise şu şekildedir:

$$p^{k+1} = p^k + \Delta p^k \tag{6.20}$$

$$\Delta p^k = -\tau^k \left(\frac{\partial J}{\partial p}\right)^k = -\tau^k \nabla J_p^k = -\tau^k g_p^k \tag{6.21}$$

Burada k,iterasyon adımını, τ^k ise k. adımdaki en iyi adım büyüklüğünü ifade eder. Adım büyüklüğü,

$$\min_{\tau} J(p - \tau \frac{\partial J}{\partial p}) \tag{6.22}$$

ifadesinin tek boyutlu optimizasyonu ile bulunur. Tek boyutlu arama için çeşitli yöntemler kullanılabilir [92-95,114,116,117]. Bu çalışmada kübik interpolasyon yöntemi kullanılmıştır. Sınırlama için ise, 2^k, 2^{-k} ile çalışan bir yerine koyma yöntemi uygulanmıştır. Bu konuda literatürde geniş yayın bulmak mümkündür. Bulunan τ

yaklaşık bir değer olduğu denemelerde görülmüştür. Genelde literatürde, YSA uygulamalarının yapıldığı çalışmalarda sabit τ kullanılmaktadır. Bunun (τ) optimizasyonu bile eğitimi hızlandırmaktadır.

6.3.2. Ölçeklenmiş konjuge gradyan yöntemi (ÖKGY)

Konjuge gradyan yöntemi (KGY), optimizasyon teorisinde yapısal basitliği ve hızlı yapısıyla fazla uygulama alanı bulmuştur. Temeli, sadece gradyan bilgisine dayalı olarak geçmiş bilgileri de kullanarak güncelleme yapmaya dayanır [47,92-95,114,116-119]. Konjuge yön belirlenerek, gradyan yönünden farklı yönde gidilerek ilerlenir. Bu çalışmada, standart KGY'nden farklı olarak ölçeklenmiş KGY (ÖKGY) gerçeklenmiştir. ÖKGY, parametrelerin birbirlerinden uzak veya biri diğerlerinden çok küçük veya büyük olan uygulamalarda etkilidir [47]. Gerçekte parametrelerin hangi değerleri alacağı baştan bilinmediği için, bu yöntemi kullanmak çok avantajlıdır.

Bu çalışmada, statik optimizasyon teorisinde geliştirilen ÖKGY, dinamik bir yapıya sahip olan DSA üzerinde geliştirilerek gerçeklenmiştir. Aşağıda, Polak ve Ribiére'nin ÖKGY adımları verilmiştir:

$$p^{k+1} = p^k + \tau^k d^k \tag{6.23}$$

$$\tau^k = \min_{\tau} J(p^k + \tau d^k) > 0 \tag{6.24}$$

$$d^{k+1} = -S_i g_p^{k+1} + \beta^k d^k \quad , \quad d^0 = -S_i g_p^0 \quad , \quad S_0 = I \tag{6.25}$$

$$\beta^k = \frac{(\Delta g_p^{k-1})^T S_i g_p^k}{(g_p^{k-1})^T S_i g_p^{k-1}} \quad , \quad \Delta g_p^{k-1} = g_p^k - g_p^{k-1} \tag{6.26}$$

$$S_{i+1}^{k+1} = S_{i+1}^k + \eta^k d^k (d^k)^T \quad , \quad S_{i+1}^0 = 0 \tag{6.27}$$

$$\eta^k = \frac{\tau^k}{(g_p^k)^T S_i g_p^k} \tag{6.28}$$

$$S_{i+1} = S_{i+1}^m \quad , \quad p_{i+1}^0 = p_i^m \tag{6.29}$$

Burada, **i** indisi en çok parametre sayısı olan **m**'ye kadar artabilen iç döngü sayısıdır. S_i matrisi **mxm**'lık simetrik matristir. S_i matrisi i=m olduğunda güncellenir. (6.24) eşitliği tek boyutlu arama işlemidir. (6.26) eşitliği Polak ve Ribiére'nin KGY için önerdiği ve etkili ve hızlı işleyen parametredir. d^k ise, gidilen yön vektörüdür. Gradyan vektörü g_p, DSA için adjoint yöntem ile elde edilmiş durumda olduğundan, geriye sadece dinamik sistem için tek boyutlu arama işlemi kalır ki, yukarıda bu konu hakkında temel bilgi verilmiştir.

Bu yöntem, DSA eğitiminde kullanılırken ek olarak, ölçekleme matrisi S_i üzerinde belirli durumlarda birim matrise indirgeme işlemi yapılmıştır. Algoritma işlerken, bazı durumlarda ilerleme durma noktasına gelir ki, bu yavaşlama belirli adımdan fazla olduğunda S_i matrisi birim matrise indirgenerek başlangıç durumundaki gibi gradyanın ters yönünde iterasyona devam edilir. Bu yaklaşım ile algoritmanın iterasyon adımlarında azalma olduğu yapılan uygulamalarda görülmüştür. Yöntemde ölçekleme matrisinin depolanması gerekmektedir. KGY'ne göre depolama yönünden fazla bellek gerektirmektedir. Bellek problemi olmayan uygulamalarda çok etkili bir yöntemdir.

6.3.3. Davidon-Fletcher-Powel (DFP) yöntemi

Bu yöntem, optimizasyon teorisinde yaklaşık-Newton metotlarından olup, değişken-metrik-algoritması olarak ta bilinir. Bu algoritma, Newton metodunda [92-95,114,116-119] gerekli olan hessian matrisinin tersinin iteratif olarak yaklaşık hesabına dayalı olarak işler. Algoritma adımları aşağıda verilmiştir:

$$p^{k+1} = p^k + \tau^k d^k \quad , \tau^k = \min_{\tau} J(p^k + \tau d^k) > 0 \tag{6.30}$$

$$d^k = -H^k g_p^k \tag{6.31}$$

$$H^{k+1} = H^k + \left[\frac{\Delta p^k (\Delta p^k)^T}{(\Delta p^k)^T \Delta g_p^k} - \frac{H^k \Delta g_p^k (\Delta g_p^k)^T (H^k)^T}{(\Delta g_p^k)^T H^k \Delta g_p^k} \right] \quad , \quad H^0 = I \tag{6.32}$$

$$\Delta p^k = p^{k+1} - p^k \quad , \quad \Delta g_p^k = g_p^{k+1} - g_p^k \tag{6.33}$$

Burada H^k, **mxm**'lik simetrik hessian matrisinin yaklaşık tersidir. Burada da, yapılan yaklaşıklıklardan ve yuvarlama hatalarından dolayı iyileşme yavaşlaması durumunda H^k matrisi birim matrise indirgenerek gradyanın tersi yönünde tekrar başlanıp ilerlemeye devam edilir. Bu yöntem, ÖKGY'den daha hızlı işleyen bir yapıya sahiptir. Yöntemde, H^k matrisinin depolanması gerekmektedir. Buradaki H^k matrisi optimal kontrol konusunda gördüğümüz Hamiltonian ifadesiyle karıştırılmamalıdır. Δp ve Δg sırayla, parametre ve ölçüt gradyan farklarıdır.

6.3.4. Broyden-Fletcher-Goldfarb-Shanno (BFGS) yöntemi

Bu yöntemde yaklaşık-Newton yöntemlerinden olup, DFP'ye göre çoğu zaman daha etkin bir yöntemdir. Yine bu yöntemde, hessian matrisinin yaklaşık tersinin hesaplanması için değişik bir yapı geliştirilmiştir [92-95,114,116-119]. Algoritma adımları aşağıdaki gibidir:

$$p^{k+1} = p^k + \tau^k d^k \quad , \quad \tau^k = \min_{\tau} J(p^k + \tau d^k) > 0 \tag{6.34}$$

$$d^k = -H^k g_p^k \tag{6.35}$$

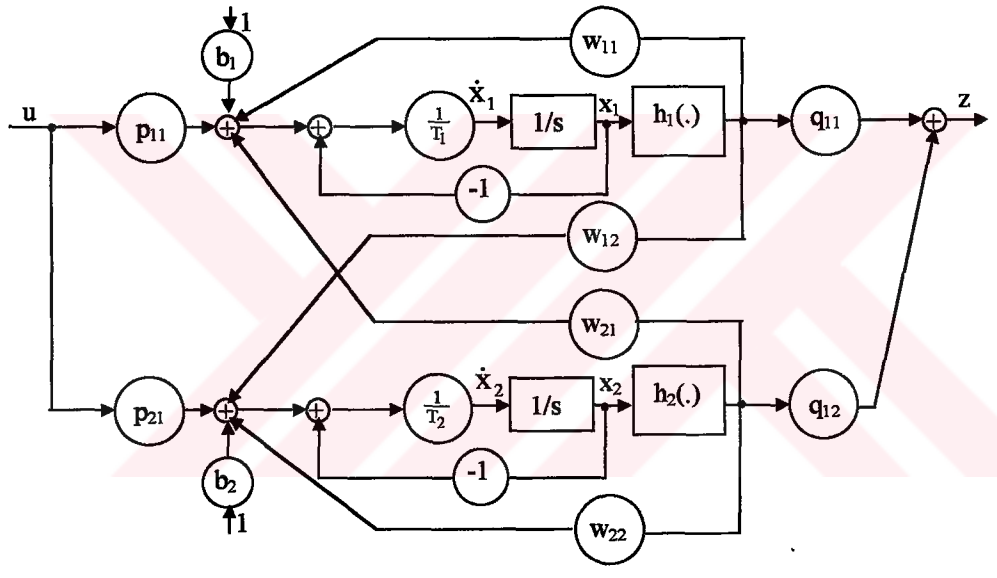
$$H^{k+1} = \left[I - \frac{\Delta p^k (\Delta g_p^k)^T}{(\Delta p^k)^T \Delta g_p^k} \right] H^k \left[I - \frac{\Delta p^k (\Delta g_p^k)^T}{(\Delta p^k)^T \Delta g_p^k} \right]^T + \frac{\Delta p^k (\Delta p^k)^T}{(\Delta p^k)^T \Delta g_p^k} \quad , \quad H^0 = I \tag{6.36}$$

DFP yöntemi için söylenen diğere özellikler BFGS algoritması için de geçerlidir.

6.4. İki Dinamik Nöronlu Sistemle Diğer DSA Modelinin Benzetim Çalışması

Uygulama olarak, ikinci dereceden (iki nöronlu) parametreleri belli olan DSA istenen sistem olarak düşünülmektedir. Bunu modelleyecek ağ yapısı da ikinci dereceden olacaktır. Böylece, geliştirilen yöntemlerin etkinlikleri karşılaştırılacaktır.

Önce ikinci dereceden DSA modelinin bütün matematiksel ifadelerini çıkararak, anlaşılması kolay hale getirilecektir. Şekil 6. 6'da dizge ve model olarak kullanılacak iki nöronlu DSA'ın açık yapısı görülmektedir. Burada, bir girişli/bir çıkışlı (BGBÇ) durum ele alınacaktır.



Şekil 6. 6 İki dinamik nöronlu DSA sistemi (BGBÇ sistem)

Şekil 6. 6'ya göre aşağıdaki ifadeler yazılır:

$$J = \tfrac{1}{2}\int_0^{t_f} [z(t) - z^d(t)]^2 \, dt = \tfrac{1}{2}\int_0^{t_f} [qy - z^d(t)]^2 \, dt \qquad (6.37)$$

Dinamik denklemler:

$$\dot{x}_1 = \frac{1}{T_1}(-x_1 + w_{11}y_1 + w_{12}y_2 + p_{11}u + b_1) = f_1 \quad , \quad x_1(0) = x_{10} \tag{6.38}$$

$$\dot{x}_2 = \frac{1}{T_2}(-x_2 + w_{21}y_1 + w_{22}y_2 + p_{21}u + b_2) = f_2 \quad , \quad x_2(0) = x_{20} \tag{6.39}$$

Sigmoid çıkışları:

$$y_1 = h_1(x_1, \gamma_1, \beta_1) = \frac{1}{1 + \exp[-(\gamma_1 x_1 + \beta_1)]} = h_1 \tag{6.40}$$

$$y_2 = h_2(x_2, \gamma_2, \beta_2) = \frac{1}{1 + \exp[-(\gamma_2 x_2 + \beta_2)]} = h_2 \tag{6.41}$$

Ağ çıkışı:

$$z(t) = q_{11}y_1 + q_{12}y_2 = \begin{bmatrix} q_{11} & q_{12} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = qy \tag{6.42}$$

Hata fonksiyonu:

$$e(t)=z(t)-z^d(t) \tag{6.43}$$

Ölçüt:

$$J = \tfrac{1}{2}\int_0^{t_f} e(t)^2\,dt = \int_0^{t_f} F(x,t)dt \tag{6.44}$$

$$\frac{\partial F}{\partial x} = e(t)\frac{\partial z}{\partial x} = e(t)\left(\frac{\partial y}{\partial x}\right)^T q^T \tag{6.45}$$

$$\frac{\partial y_i}{\partial x} = y_i' = \gamma_i h_i (1 - h_i) = g_i \tag{6.46}$$

Bu son ifadeler açık yazılırsa:

$$\frac{\partial F}{\partial x_1} = e(t)\frac{\partial z}{\partial x_1} = e(t) q_{11} y_1' = e(t) q_{11} \gamma_1 h_1 (1 - h_1) = e(t) q_{11} g_1 \tag{6.47}$$

$$\frac{\partial F}{\partial x_2} = e(t) q_{12} g_2 \tag{6.48}$$

Adjoint diferansiyel denklemi:

$$-\dot{\lambda} = \left(\frac{\partial f}{\partial x}\right)^T \lambda + \frac{\partial F}{\partial x} \quad , \quad \lambda(t_f) = 0 \tag{6.49}$$

$$-\dot{\lambda} = A^T \lambda + \frac{\partial F}{\partial x} \quad , \quad \lambda(t_f) = 0 \tag{6.50}$$

Gerekli hesaplamalar sonunda:

$$\begin{bmatrix} -\dot{\lambda}_1 \\ -\dot{\lambda}_2 \end{bmatrix} = \begin{bmatrix} (-1 + w_{11} g_1) & w_{21} g_1 \\ w_{12} g_2 & (-1 + w_{22} g_2) \end{bmatrix} \begin{bmatrix} \frac{1}{T_1} & 0 \\ 0 & \frac{1}{T_2} \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} + e(t) \begin{bmatrix} g_1 & 0 \\ 0 & g_2 \end{bmatrix} \begin{bmatrix} q_{11} \\ q_{12} \end{bmatrix} \tag{6.51}$$

Ve adjoint duyarlılık hesaplamaları:

$$\frac{\partial J}{\partial p} = \int_0^{t_f} \lambda^T \frac{\partial f}{\partial p} dt \tag{6.52}$$

Tüm parametreler için (w,p,q,b,T,γ,β) duyarlılıkları:

$$\frac{\partial J}{\partial w_{11}} = \int_0^{t_f} \frac{\lambda_1 y_1}{T_1} dt, \quad \frac{\partial J}{\partial w_{12}} = \int_0^{t_f} \frac{\lambda_1 y_2}{T_1} dt, \quad \frac{\partial J}{\partial w_{21}} = \int_0^{t_f} \frac{\lambda_2 y_1}{T_2} dt, \quad \frac{\partial J}{\partial w_{22}} = \int_0^{t_f} \frac{\lambda_2 y_2}{T_2} dt, \tag{6.53}$$

$$\frac{\partial J}{\partial p_{11}} = \int_0^{t_f} \frac{\lambda_1 u}{T_1} dt, \quad \frac{\partial J}{\partial p_{21}} = \int_0^{t_f} \frac{\lambda_2 u}{T_2} dt \tag{6.54}$$

$$\frac{\partial J}{\partial q_{11}} = \int_0^{t_f} e(t) y_1 dt, \quad \frac{\partial J}{\partial q_{12}} = \int_0^{t_f} e(t) y_2 dt \tag{6.55}$$

$$\frac{\partial J}{\partial b_1} = \int_0^{t_f} \frac{\lambda_1}{T_1} dt, \quad \frac{\partial J}{\partial b_2} = \int_0^{t_f} \frac{\lambda_2}{T_2} dt \tag{6.56}$$

$$\frac{\partial J}{\partial T_1} = -\int_0^{t_f} \frac{\lambda_1}{T_1} f_1 dt, \quad \frac{\partial J}{\partial T_2} = -\int_0^{t_f} \frac{\lambda_2}{T_2} f_2 dt \tag{6.57}$$

$$\frac{\partial J}{\partial \gamma_1} = \int_0^{t_f} (w_{11} \frac{\lambda_1}{T_1} + w_{21} \frac{\lambda_2}{T_2}) x_1 h_1 (1 - h_1) dt, \quad \frac{\partial J}{\partial \gamma_2} = \int_0^{t_f} (w_{12} \frac{\lambda_1}{T_1} + w_{22} \frac{\lambda_2}{T_2}) x_2 h_2 (1 - h_2) dt \tag{6.58}$$

$$\frac{\partial J}{\partial \beta_1} = \int_0^{t_f} (w_{11} \frac{\lambda_1}{T_1} + w_{21} \frac{\lambda_2}{T_2}) h_1 (1 - h_1) dt, \quad \frac{\partial J}{\partial \beta_2} = \int_0^{t_f} (w_{12} \frac{\lambda_1}{T_1} + w_{22} \frac{\lambda_2}{T_2}) h_2 (1 - h_2) dt \tag{6.59}$$

Böylece, bütün parametrelere göre gradyan hesaplanabilmektedir. Bunun için önce, (6.51) ile verilen adjoint sistemini zamanda geriye doğru çözmek gerekir. Sistemde toplam 16 tane parametre vardır. İki tanesi çıkışta (q) olduğundan, gradyanı direkt olarak hesaplanır. Geriye kalan 14 parametre için sadece iki tane zamanda geriye doğru adjoint diferansiyel denklemi çözmekle bütün gradyanlar hesaplanabilmektedir. Bu, direkt yönteme göre büyük oranda hesap maliyeti kazancı sağladığı açıkça görülmektedir [97,98].

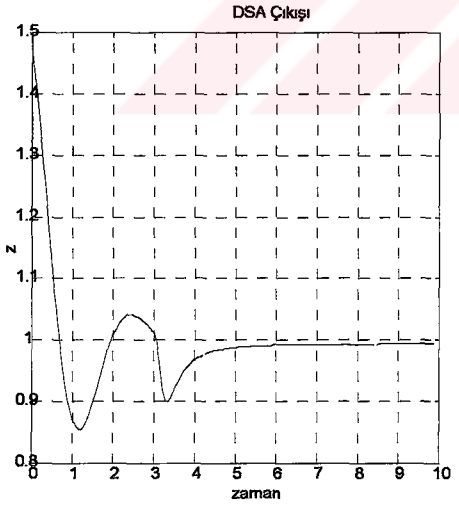
Örnek olarak, hedef dizge için aşağıdaki parametreler kullanılacaktır [97,98]:

$$\mathbf{w}=\begin{bmatrix}0 & -10\\ 10 & 0\end{bmatrix},\mathbf{p}=\begin{bmatrix}5\\ -5\end{bmatrix},\mathbf{q}^{T}=\begin{bmatrix}1\\ 1\end{bmatrix},\mathbf{b}=\begin{bmatrix}0\\ 0\end{bmatrix},\mathbf{T}=\begin{bmatrix}1\\ 1\end{bmatrix},\gamma=\begin{bmatrix}1\\ 1\end{bmatrix},\beta=\begin{bmatrix}0\\ 0\end{bmatrix} \quad (6.60)$$

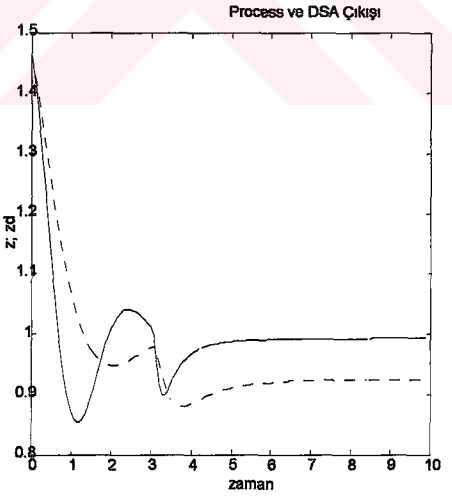
ve t_0=0, t_f=10sn olarak, başlangıç durumu ise, $x(0)=[1\ 1]^T$ olarak alınıyor. İntegrasyon aralığı 0.1, giriş işareti $t\in[0,3]$ aralığında 1.0, $t\in(3,10]$ aralığında ise -1.0 olarak DSA'na uygulanıyor. Cevap çıkışı, Şekil 6. 7'de gösterildiği gibi olup, eğitimi yapılacak diğer DSA için hedef yörünge olacaktır. Eğitimi yapılacak diğer DSA için sadece **w** ve **p** parametrelerinin bulunması problemi çözülecektir. Diğer parametreler aynen alınıp sadece **w** ve **p** için güncelleme yapılacaktır. Model DSA'nın başlangıç parametreleri aşağıdaki gibi alınmıştır:

$$\mathbf{w}=\begin{bmatrix}0 & -5\\ 5 & 0\end{bmatrix},\mathbf{p}=\begin{bmatrix}2.5\\ -2.5\end{bmatrix},\mathbf{q}^{T}=\begin{bmatrix}1\\ 1\end{bmatrix},\mathbf{b}=\begin{bmatrix}0\\ 0\end{bmatrix},\mathbf{T}=\begin{bmatrix}1\\ 1\end{bmatrix},\gamma=\begin{bmatrix}1\\ 1\end{bmatrix},\beta=\begin{bmatrix}0\\ 0\end{bmatrix},x(0)=\begin{bmatrix}1\\ 1\end{bmatrix} \quad (6.61)$$

Bu durumda, iterasyon başlangıç durumu için hedef ve model DSA cevapları birlikte Şekil 6. 8'de gösterilmektedir.



Şekil 6. 7 DSA dizgesinin çıkış yörüngesi

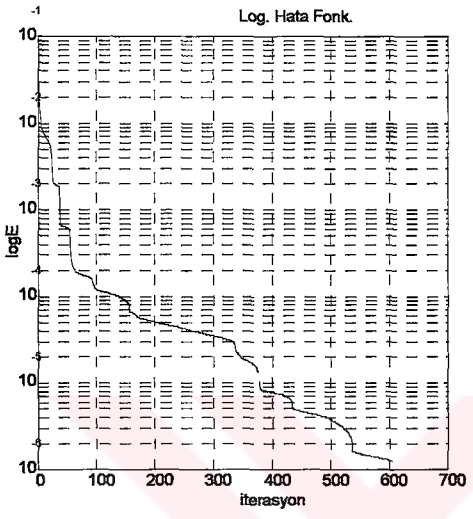


Şekil 6. 8 DSA dizge ve başlangıç durumunda model çıkışları (kesikli çizgi model çıkışı)

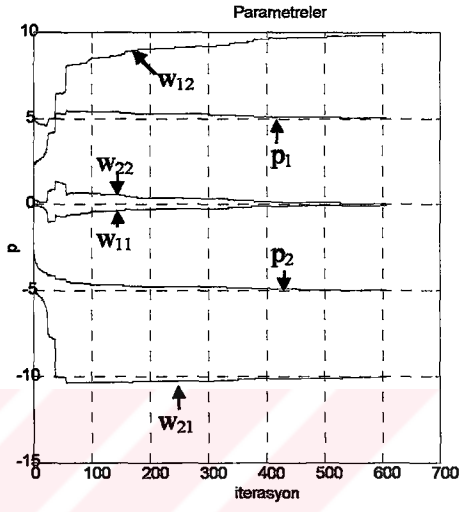
Eğitim algoritmaları olarak (6.3)'te anlatılan 4 yöntem kullanılarak karşılaştırılacaktır.

i) EAGY için yapılan denemede PÖ'nün değişimi Şekil 6. 9'da verilmiştir. Görüldüğü gibi ölçüt, optimal parametrelere yaklaştıkça veya hata en aza yaklaştıkça

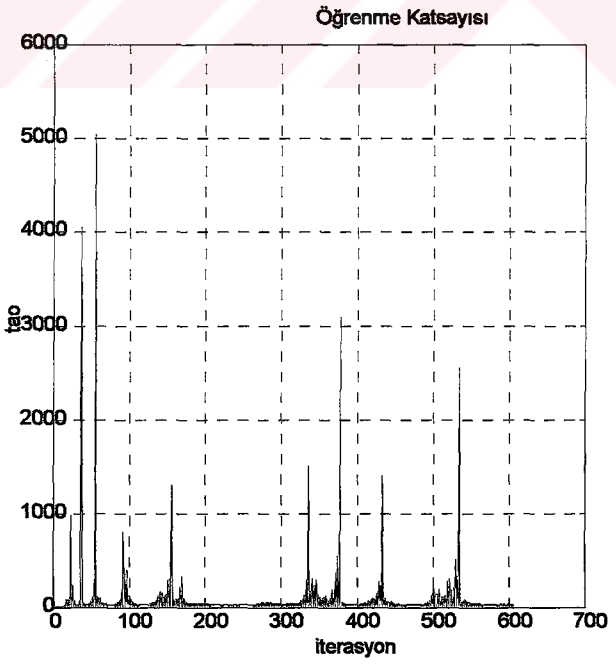
yavaşlamaktadır. **w** ve **p** parametrelerin değişimi ise Şekil 6. 10'da verilmektedir. Adım büyüklüğünün veya öğrenme katsayısının (τ) değişimi Şekil 6. 11'de veriliyor. Bu katsayı iterasyonla değişmektedir. Bu ise, sabit öğrenme katsayılı eğitime göre hızlı sonuç verir. Şekil 6. 12'de ise DSA dizge ve model çıkışlarının değişik iterasyonlar sonucundaki değişimlerini göstermektedir.



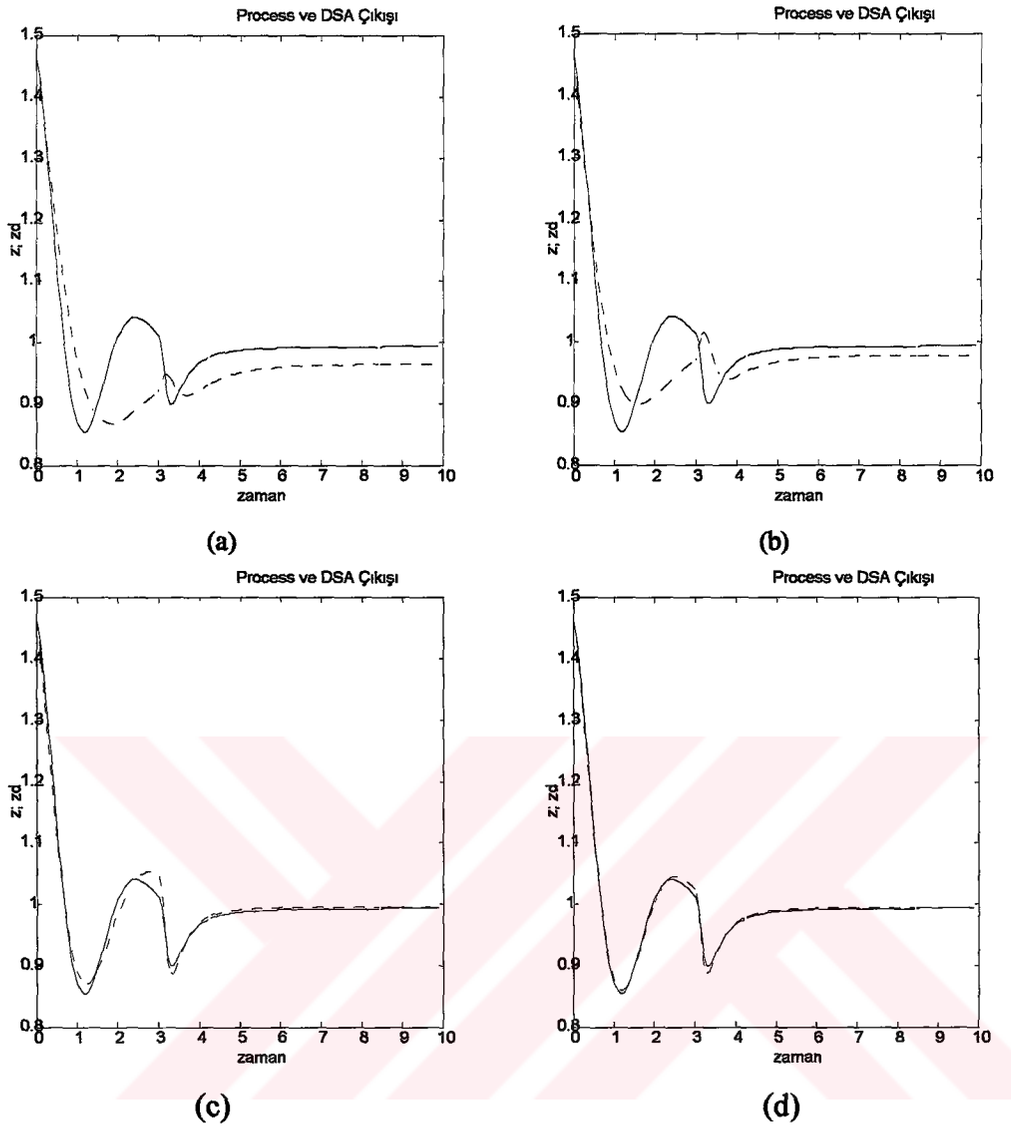
Şekil 6. 9 EAGY için PÖ'nün değişimi (yarı logaritmik)



Şekil 6. 10 EAGY'de w ve p parametrelerinin iterasyonla değişimleri

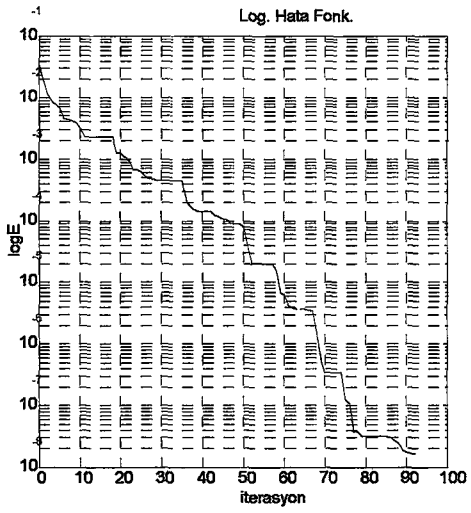


Şekil 6. 11 EAGY'de öğrenme katsayısının (τ) değişimi

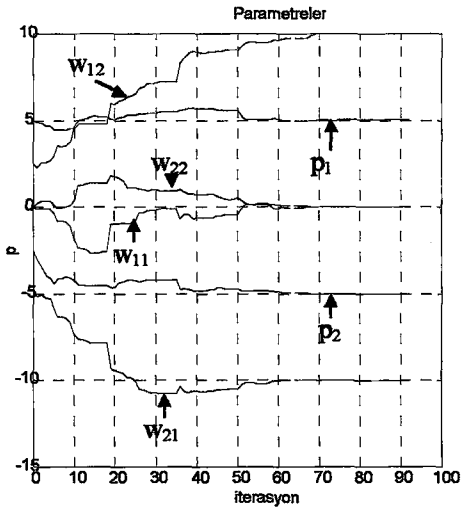


Şekil 6. 12 EAGY'de DSA dizge ve model çıkışları (kesikli çizgi model çıkışları) (a) 1.adım (b) 5. adım (c) 50. adım (d) 150. adım

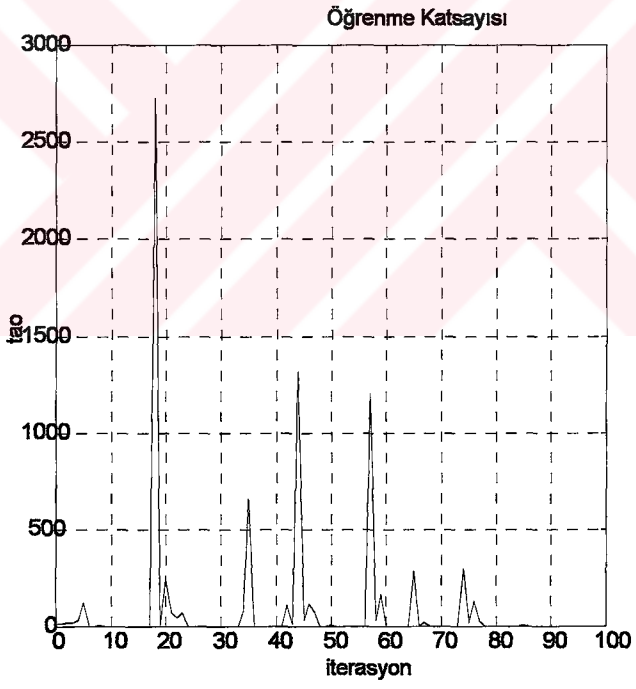
ii) ÖKGY için de aynı işlemler yapılmıştır. Bu yöntemin, EAGY'ne göre daha hızlı yakınsadığı Şekil 6. 13'ten görülebilir. Şekil 6. 14'te **w** ve **p** parametrelerinin değişimi görülmektedir. Buradan, bu yöntemin parametreleri hızlı bir şekilde sürekli hale getirdiği anlaşılabilir. Şekil 6. 15'te ise öğrenme katsayısının iterasyonla değişimi görülmektedir. Şekil 6. 16'da, DSA dizge ve model çıkışlarının çeşitli iterasyon adımları için değişimleri verilmektedir.



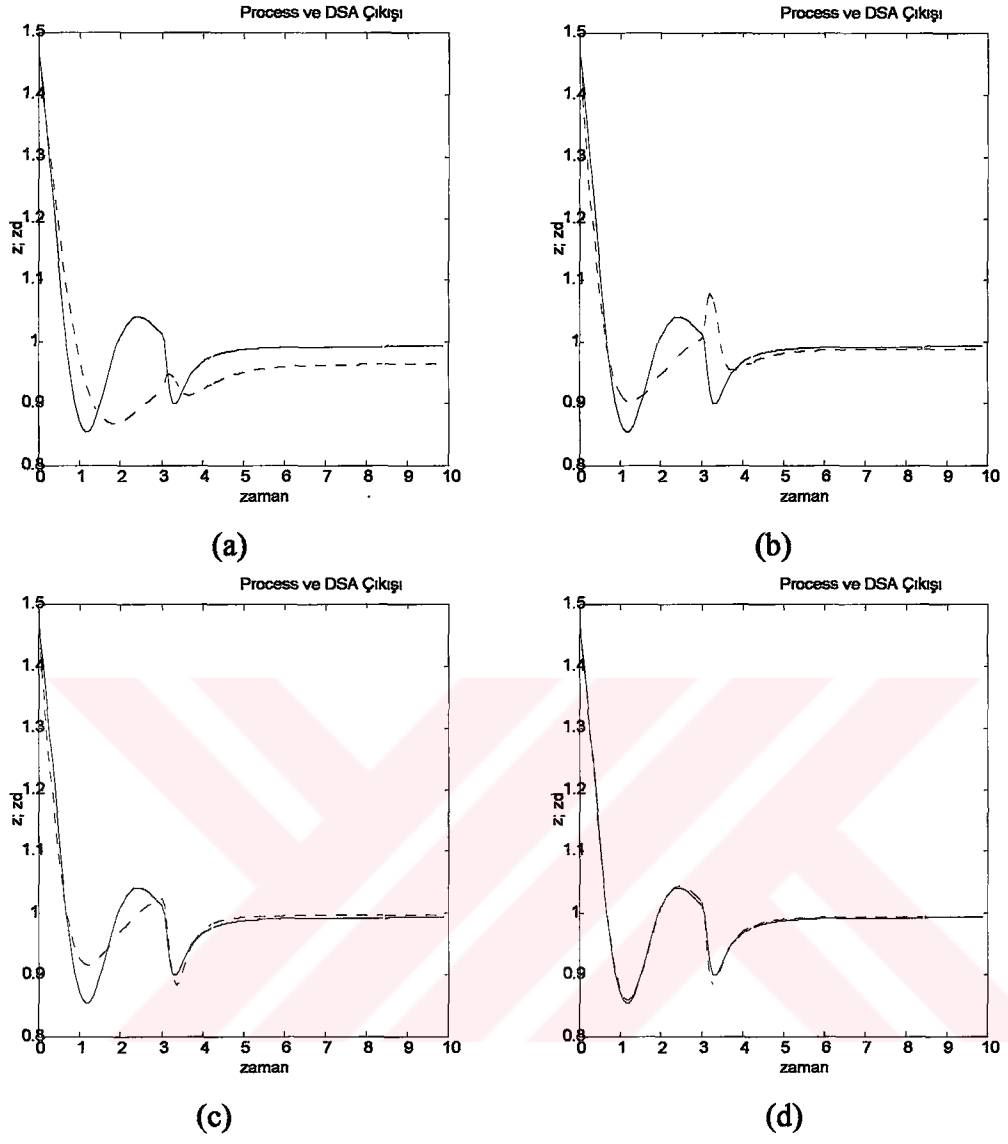
Şekil 6. 13 ÖKGY için PÖ'nün değişimi (yarı logaritmik)



Şekil 6. 14 ÖKGY w ve p parametrelerinin iterasyonla değişimleri

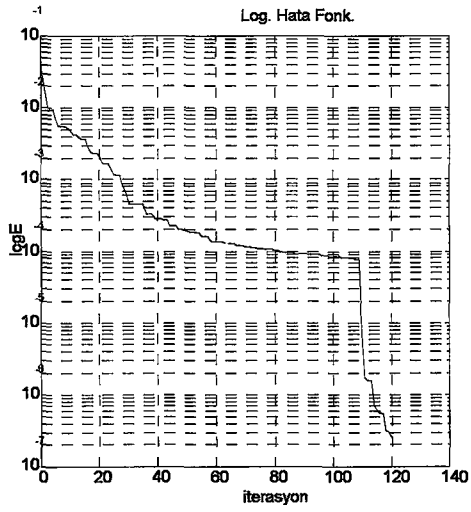


Şekil 6. 15 ÖKGY'de öğrenme katsayısının (τ) değişimi

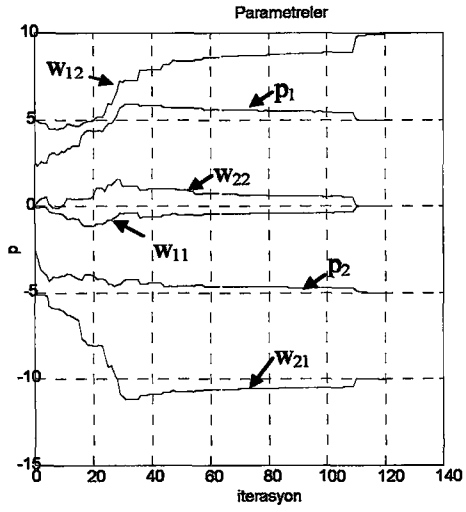


Şekil 6. 16 ÖKGY'de DSA dizge ve model çıkışları (kesikli çizgi model çıkışları) (a) 1.adım (b) 5. adım (c) 15. adım (d) 50. adım

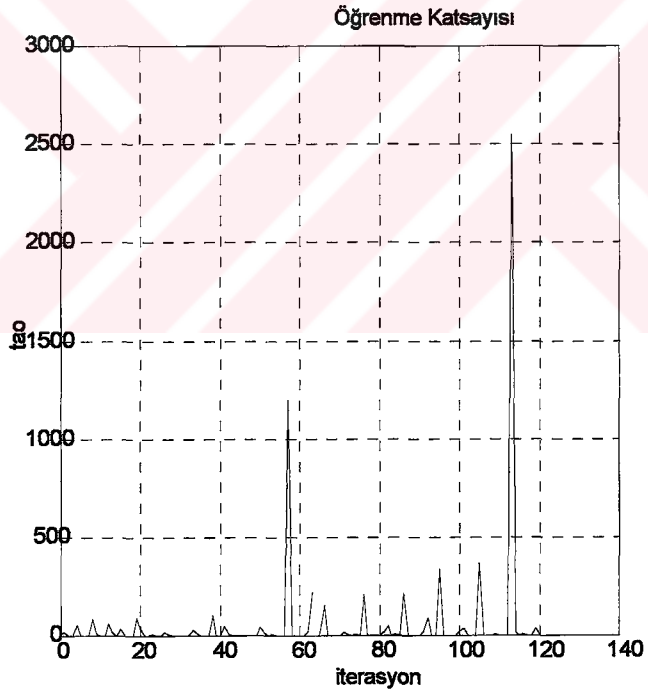
iii) DFP yönteminin performans denemesi sonunda, ÖKGY'nden biraz daha yavaş bir yakınsama ve ilerleme sağladığı görülmüştür. Şekil 6. 17'de PÖ, Şekil 6. 18'de ise parametrelerin değişimi verilmiştir. Öğrenme katsayısının değişimi Şekil 6. 19'da gösterilmektedir. Şekil 6. 20'de ise değişik iterasyon adımlarındaki çıkışlar verilmiştir.



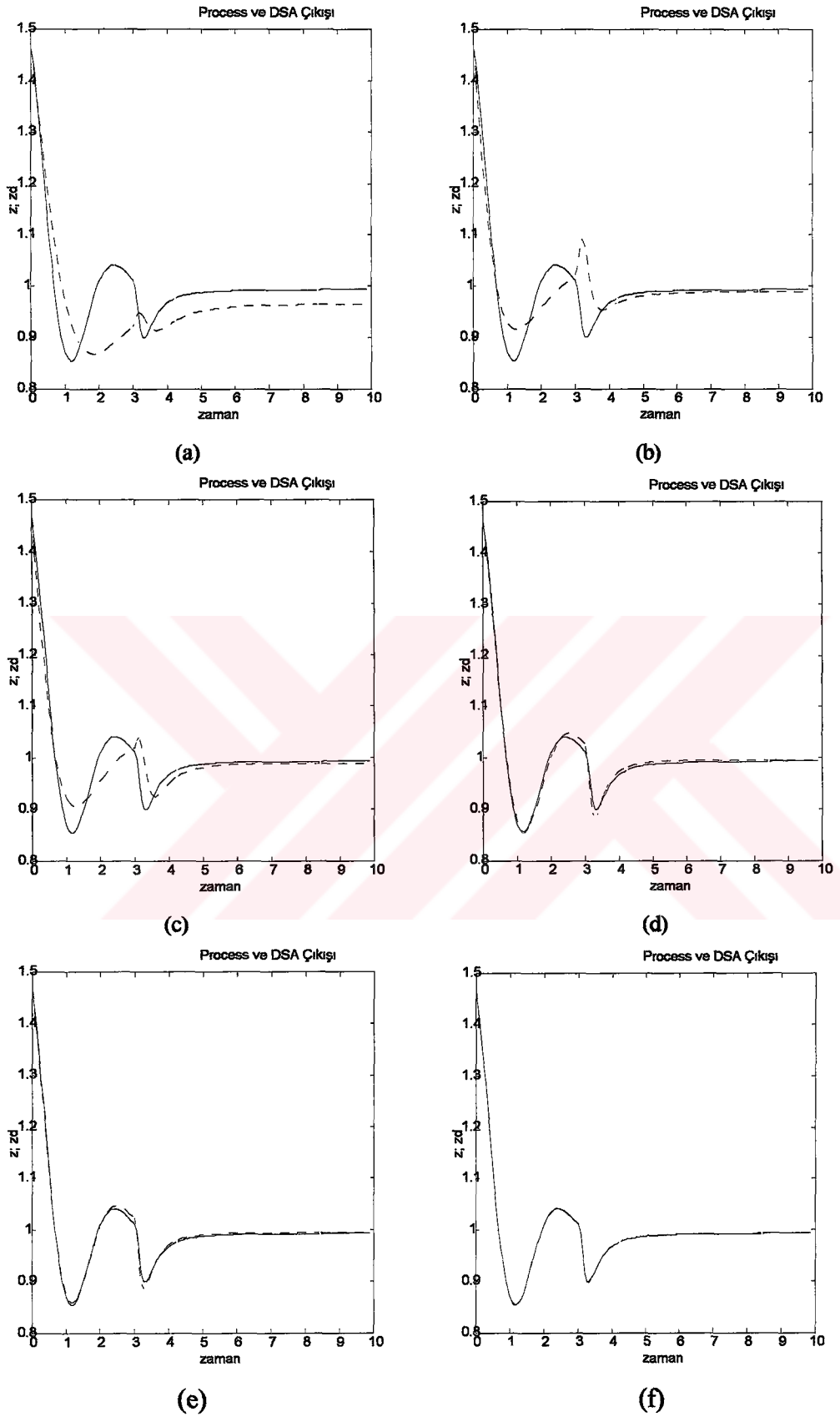
Şekil 6. 17 DFP yöntemi için PÖ'nün değişimi (yarı logaritmik)



Şekil 6. 18 DFP yönteminde w ve p parametrelerinin iterasyonla değişimleri

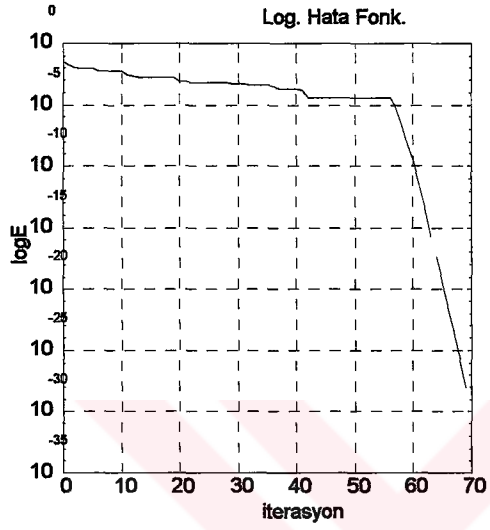


Şekil 6. 19 DFP yönteminde öğrenme katsayısının (τ) değişimi

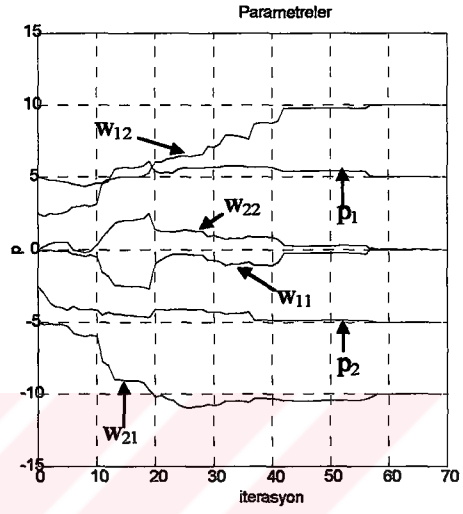


Şekil 6. 20 DFP yönteminde DSA dizge ve model çıkışları (kesikli çizgi model çıkışları) (a) 1.adım (b) 5. adım (c) 15. adım (d) 50. adım (e) 80. adım (f) 120. adım

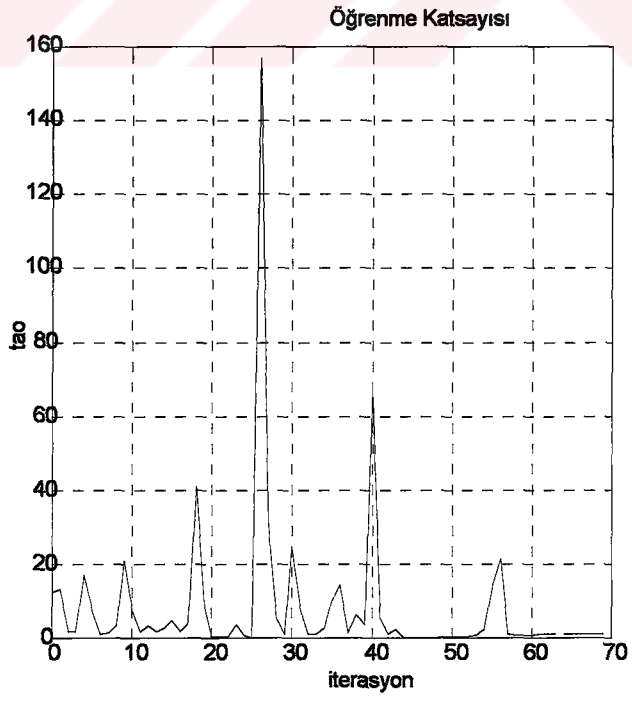
iv) BFGS yöntemi, bu çalışmada kullanılan son eğitim yöntemidir. BFGS, DFP ve ÖKGY'ne göre daha iyi performans göstermekle beraber, genelde daha hızlı yakınsar. BFGS ile ilgili benzer benzetim çıkışları Şekil 6. 21, Şekil 6. 22, Şekil 6. 23 ve Şekil 6. 24'te verilmektedir. Öğrenme katsayısının optimum noktaya yaklaşıldığında 1'e doğru gittiğine dikkat edilmelidir.



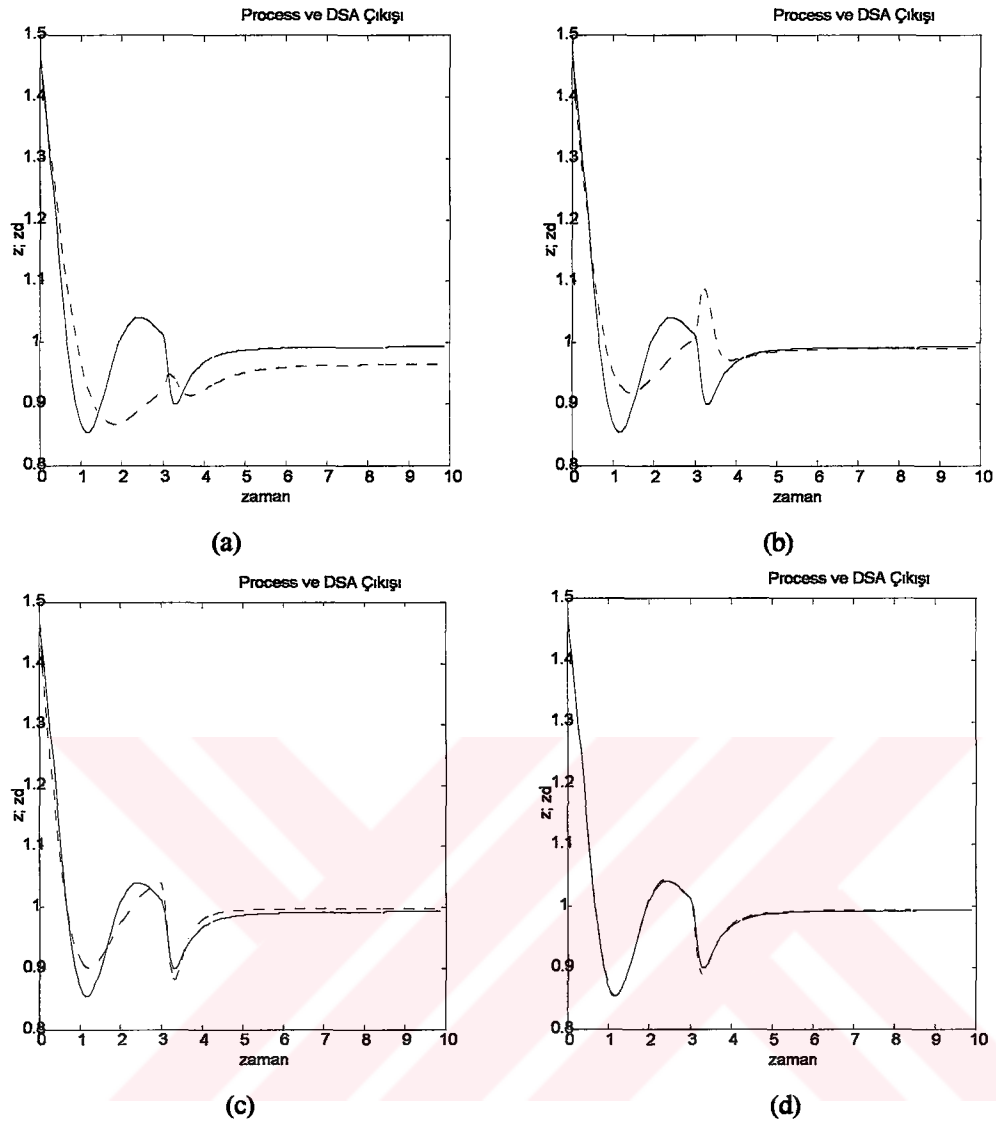
Şekil 6. 21 BFGS yöntemi için PÖ'nün değişimi (yarı logaritmik)



Şekil 6. 22 BFGS yönteminde w ve p parametrelerinin iterasyonla değişimleri



Şekil 6. 23 BFGS yönteminde öğrenme katsayısının (τ) değişimi

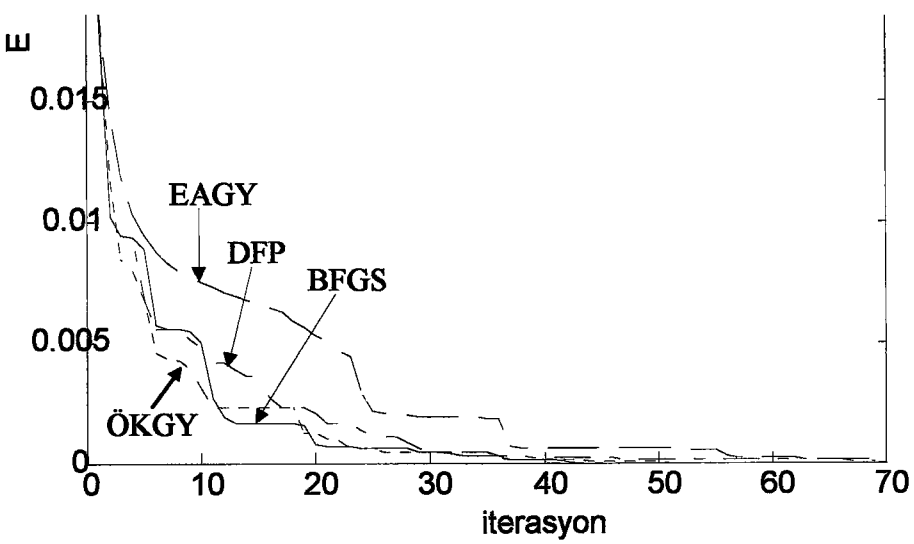


Şekil 6. 24 BFGS yönteminde DSA dizge ve model çıkışları (kesikli çizgi model çıkışları) (a) 1.adım (b) 5. adım (c) 15. adım (d) 50. adım

Dinamik bir sistem olan DSA'na başarılı bir şekilde 4 optimizasyon yöntemi uygulanmıştır. Bu yöntemlerin hepsinin bir arada PÖ'lerinin değişimleri Tablo 6. 1'de verilmektedir. Sonuçlara göre ÖKGY ve BFGS yöntemleri birbirlerine yakın, diğer ikisine göre iyi performans göstermişlerdir. BFGS yöntemi çözüm noktasına yaklaştıkça çok hızlı yakınsadığı görülmektedir. ÖKGY'nin DFP'ye göre daha etkili çıkması da dikkat çekici bir sonuç olarak görülebilir. EAGY ise yavaş kalmaktadır. Ancak, kolay anlaşılabilir ve basit yapısı nedeniyle çok yaygın olarak kullanılmaktadır. Şekil 6. 25'te ise dört yöntemin bir arada PÖ tabanlı olarak karşılaştırılması gösterilmektedir. Şekilde 70 iterasyona kadar olan değişimler verilmiştir.

Tablo 6. 1 Dört yöntem için PÖ'nün karşılaştırılması

İterasyon (k)	EAGY	ÖKGY	DFP	BFGS
0	3.4849e-2	3.4849e-2	3.4849e-2	3.4849e-2
1	1.8877e-2	1.8877e-2	1.8877e-2	1.8877e-2
2	1.4508e-2	1.1737e-2	1.0298e-2	1.0246e-2
3	1.1956e-2	8.5097e-3	9.4833e-3	9.4801e-3
4	1.0319e-2	8.0409e-3	9.3594e-3	9.3629e-3
5	9.4624e-3	6.7832e-3	6.8213e-3	8.8551e-3
6	8.7929e-3	4.5683e-3	5.6023e-3	5.7743e-3
7	8.3596e-3	4.3527e-3	5.5669e-3	5.5809e-3
8	7.9839e-3	4.2729e-3	5.5669e-3	5.5493e-3
9	7.7414e-3	3.9861e-3	5.1782e-3	5.5288e-3
10	7.5014e-3	3.1724e-3	4.8142e-3	5.0475e-3
.				
15	6.6564e-3	2.3360e-3	3.6352e-3	1.6793e-3
.				
20	5.3447e-3	1.2653e-3	2.0895e-3	8.2977e-4
.				
30	1.9569e-3	4.5247e-4	4.7129e-4	5.0053e-4
.				
40	6.6253e-4	1.4076e-4	2.7861e-4	1.9769e-4
.				
50	6.1642e-4	8.2466e-5	1.9219e-4	3.9762e-5
.				
57	2.5660e-4	2.0129e-5	1.5773e-4	7.7875e-6
.				
60	2.1712e-4	6.2037e-6	1.3453e-4	3.6409e-10
.				
70	1.8358e-4	3.4641e-7	1.1542e-4	6.2615e-31
.				
80	1.7274e-4	3.1603e-8	1.0137e-4	
.				
93	1.4661e-4	1.6796e-8	8.9829e-5	
.				
121	1.0875e-4		1.5656e-7	
.				
200	5.0556e-5			
.				
300	3.4818e-5			
.				
400	7.7709e-6			
.				
600	1.2761e-6			
.				
615	1.2055e-6			



Şekil 6. 25 Dört yöntemin PÖ tabanlı karşılaştırılması

BÖLÜM 7. HİBRİD DİNAMİK YÖRÜNGELEYİCİ TASARIMI

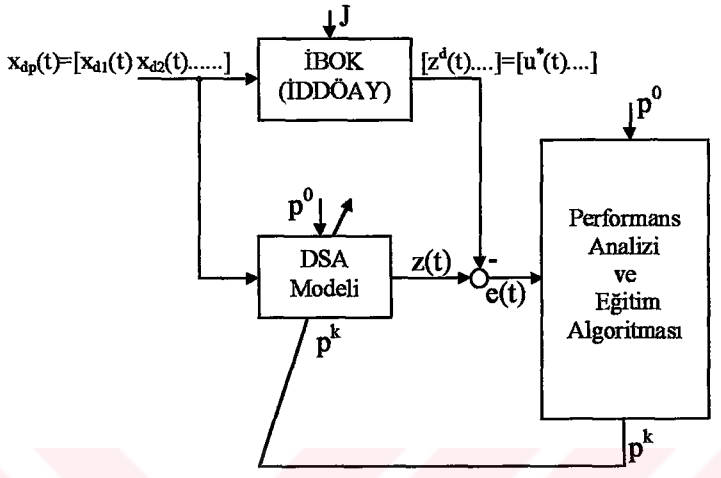
Bu bölümde, DSA ve optimal kontrol tekniklerinin bir arada kullanılarak, gerçek zamanlı uygulamalara yönelik hızlandırılmış yörünge üretimi yapısı geliştirilmiştir. Optimal kontrolün YSA ile kullanımı konusunda geliştirilen yöntemler genelde, optimal kontrol probleminin çözümüne yöneliktir [84-89]. Bu tez çalışmasında, DSA yapısı kullanılmıştır ve DSA, optimal kontrolcü için kontrol başlatıcısı olarak kullanılmıştır. Böylece, iyi bir yörüngeden başlayarak bir kaç adımda optimal kontrol ve yörüngeyi bulmak kolayca mümkün olmaktadır. Ve geri-besleme yapısı da otomatik olarak çıkarıldığı önceki bölümlerde anlatılmıştı. Ayrıca, oluşturulan kendi kendine öğrenme yapısıyla sistemin çalışma aşamasında da eğitime devam etmesi sağlanmıştır. Böylece, eğitimde kullanılmayan yörüngeler de bu sayede kullanılmakta ve algoritma gittikçe daha iyi performansta çalışmaktadır.

7.1. DSA'lı Yörünge Başlatıcı ve LOOK Yörüngeleyici Entegrasyonu

Optimal kontrol problemi Bölüm 3'te detaylı olarak incelenmişti. Burada, optimal kontrol yöntemi ile elde edilen değişik amaçlı yörüngelerin DSA'a aktarılması veya öğretilmesi amaçlanmaktadır. Bir başka anlatımla, optimal kontrolden elde edilen optimal ileri besleme (İB) yörünge bilgileri, DSA'ın kapasitesi oranında depolanmakta ve böylece farklı yörüngeleri genelleyerek sonraki kullanımda büyük oranda zaman tasarrufu sağlamaktadır. Burada, LOOK olarak İDDÖAY kullanılmıştır. Diğerinin de kullanılması mümkündür. DSA'ın eğitilmesi İB'li olarak İDDÖAY yardımıyla Şekil 7. 1'de gösterilmektedir. Bir çok değişik giriş yörüngeleri için optimal kontrol çıkışlarına göre DSA eğitilmektedir. Bölüm 6'da anlatılan eğitim algoritmaları burada kullanılmıştır.

Optimal kontrol işlemleri, bir dinamik programlama veya bir dinamik sistem

optimizasyonu olarak görülebilir. Bu nedenle yapılan işlem, bir dinamik sistemin modellenmesi anlamına da gelir. Modelleyici olarak DSA kullanıldığı için, dinamik nöron sayısı konusunda bazı analojiler yapılabilir. Lineer olmayan bir sistemin modellenmesinde, en az derecesi kadar DSA'da nöron kullanmanın yeterli olabileceği gösterilebilir [97,98].



Şekil 7. 1 DSA'ın optimal kontrol yörüngeleriyle eğitilmesi

Bu yaklaşımdan yola çıkarak, optimal kontrol probleminin çözümü sonucunda, kazanç matrisi K(t) ve sürücü kazanç vektörü b(t)'nin DSA tarafından öğrenilmesi gerektiği düşünülürse, bunların dereceleri toplamı kadar (en azından) yaklaşık olarak DSA nöron sayısı tespit edilebilir. Böylece, **n** sistem derecesi ve **m** kontrol derecesi olmak üzere nöron sayısı için,

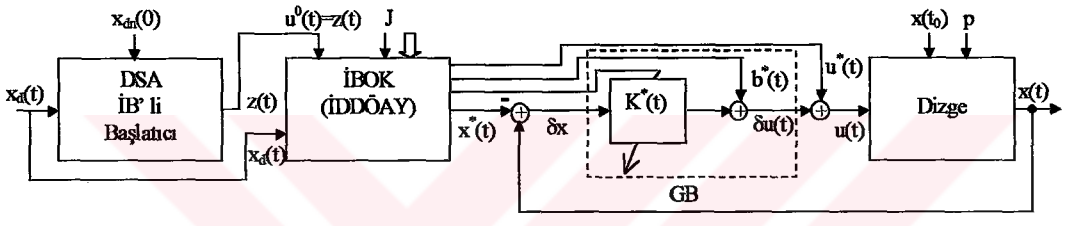
$$dns=m(n+1) \tag{7.1}$$

yazılabilir. Bu sayıyı heuristik bir değer olarak önermekteyiz. Sistemin lineersizlik durumuna göre bu sayı artırılabilir. Ancak buna gerek duyulacak problemlerin tanımı tez dışında bırakılmıştır. Zaten, DSA modeli, dinamik zaman sabiti ve lineersizlik parametrelerini de içerdiği için bu duruma gerek duyulmayacaktır. DSA başlangıç koşulları ve başlangıç parametreleri rasgele atanmaktadır.

Eğitim aşaması tamamlandıktan sonra, DSA için performans testleri yapılabilir. Eğitim

seti içinde olmayan yörüngeler için kontrol çıkışları, optimal kontrol çıkışlarıyla karşılaştırılır. Yapılan denemelerdeki genellemelerde oldukça iyi sonuçlar alınmıştır. Bu sonuçlar bir sonraki bölümde verilecektir.

Bu aşamadan sonra DSA ve optimal kontrol yapısı, İB ve GB biçiminde tasarımlanarak yörünge üreticisi tamamlanır. Burada, DSA'ın ürettiği yaklaşık optimal kontrol fonksiyonu, optimal kontrolcü için başlangıç kontrol fonksiyonu olacaktır. Bir kaç adım içinde hem optimal kontrol ve hem de GB'li sistem oluşturulacaktır. Dolayısıyla, tam bir yörünge üreticisi olarak dinamik-sinirsel-optimal-kontrol (DSOK) yapısı tamamlanmış olur. Bu durum Şekil 7. 2'de açık bir şekilde gösterilmektedir.



Şekil 7. 2 DSOK ve GB yapısı ile yörüngeleyici entegrasyonu

7.2. Kendi Kendine Öğrenme Yapısı ile Oluşturulan Zeki Hibrid Yapı

Bu yapıda, başlangıçtan itibaren sistem gerçeklenme aşamasında olup DSA, optimal kontrolcüyü takip etmeye çalışmaktadır. Ayrıca, yukarıda da anlatılığı gibi başlangıçta bir miktar eğitim yapıldıktan sonra da otomatik öğrenmeye başlanabilir. Ancak buna gerek duyulmadan da yapılabilir. Bu durumda başlangıçta DSA uygun sonuçlar vermemesine karşılık, gittikçe öğrenerek daha iyi sonuçlar verecektir. Oluşturulan bu zeki hibrid sistemi Şekil 2.1'de verildiği gibidir.

7.3. DSA ve LOOK Yapılarıyla Hibrid Dinamik Yörüngeleyici Benzetim Çalışması

Burada, Bölüm 5'te incelenen VDP ve CSTR sistemleri hibrid dinamik yörüngeleyici ile kontrol edilecektir. Oradaki tüm parametre ve değişkenler burada da kullanılacaktır

7.3.1. DSOK olarak VDP sisteminin kullanılması

Optimal kontrol PÖ'nün parametreleri Bölüm 5'deki gibi alınacaktır:

$Q=\begin{bmatrix}10 & 0\\ 0 & 1\end{bmatrix}$, $R=0.1$, $S=\begin{bmatrix}10 & 0\\ 0 & 10\end{bmatrix}$, $t_0=0$, $t_f=5sn$, $x(0)=\begin{bmatrix}1\\ 0\end{bmatrix}$ ve VDP sistem parametresi a'nın ölçülebildiği veya verildiği varsayılmaktadır. Eğitimde bu parametre de DSA girişi olarak kullanılacaktır.

DSA eğitimi için bir eğitim seti hazırlanmıştır. İstenilen yörüngeler ve sistem parametreleri için belirlenen bu set aşağıdaki gibidir:

$[x^d(t)]=\begin{bmatrix}-2 & -1 & 0 & 1 & 2\\ 0 & 0 & 0 & 0 & 0\end{bmatrix}r(t)$ ve $a=\begin{bmatrix}1 & .5 & .1\end{bmatrix}$ olarak t=0'dan t_f=5sn'ye kadar sabit (regülatör) bir şekilde giriş yörünge seti (istenen yörüngeler) alınmıştır. Buradaki r(t) birim basamak fonksiyonudur. Buna karşı olarak, $[z^d(t)]=[u(t)]$ kontrol fonksiyonları her yörünge için optimal kontrol yardımıyla bulunarak DSA için hedef yörüngeler adım adım belirlenecektir. Görüldüğü gibi, her hedef yörünge için VDP parametreleri de değişmekte olup parametre uzayı da taranmaktadır.

Bu eğitim seti, yukarıda anlatıldığı gibi, Bölüm 6'da verilen optimizasyon yöntemleriyle DSA'nın eğitimi yapılmıştır. Burada, eğitim setinden bir yörünge DSA girişine verilir. Sonuçta DSA çıkışı İDDÖAY algoritmasına uygulanır. Sonra T_1 zaman sonra optimal kontrol algoritması, optimal kontrol yörüngesini üretir. Bu yörünge DSA ağırlıklarının güncellenmesinde kullanılır. Bu yaklaşıma sürekli olarak, DSA'dan daha iyi sonuç elde edilene kadar devam edilir. DSA başlangıç koşulları rasgele seçilmiştir.

DSA için nöron sayısı 3 olarak alınmıştır. Yukarıda anlatıldığı gibi, VDP sisteminde kontrol sayısı 1 ve sistem derecesi 2 olduğu için (7.1) eşitliğinden dns=3 bulunur. DSA giriş sayısı 3, çıkış sayısı 1 adet olup toplam parametre sayısı 33'dur. Tüm parametrelere göre güncelleme yapılabilir.

Bu deneme çalışmasında, nöron bias parametreleri ve sigmoid bias parametreleri

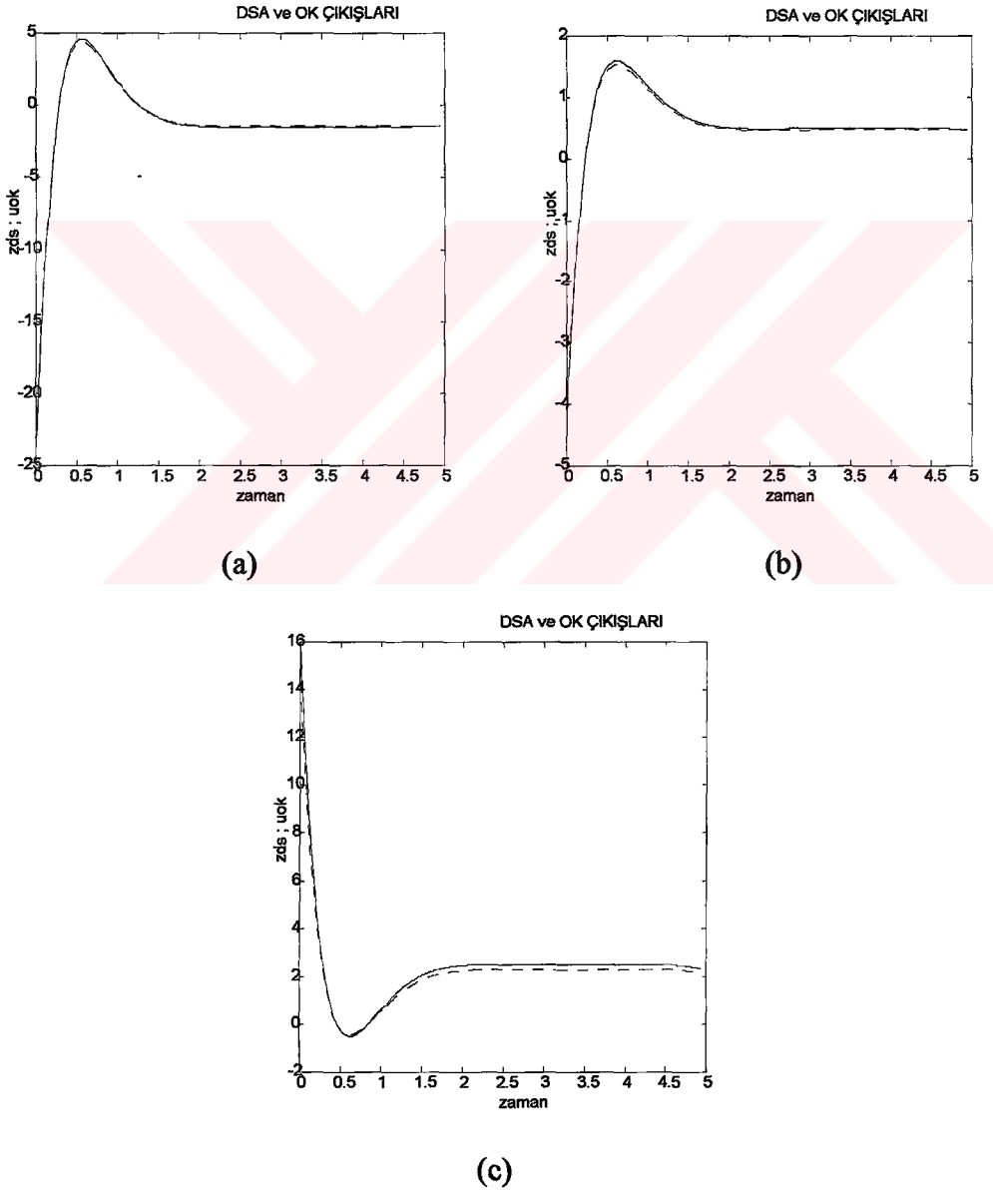
dışındaki tüm parametreler serbest bırakılmıştır (Bu duruma birkaç denemeden sonra karar verilmiştir). Bu durumda 27 parametre üzerinde optimizasyon, yani eğitim yapılmıştır. Gradyan hesaplamalarında adjoint duyarlılık yöntemi kullanılmıştır. Dinamik zaman sabiti T'ler için bir alt sınır olarak 0.1 kullanılmıştır. Sigmoid fonksiyonu lineersizlik parametresi γ için de 10^{-3} alt sınırlama olarak kullanılmıştır. Eğitimde kullanılan eğitim seti için, bir iterasyon bu sete karşılık gelmektedir. Başlangıçta tüm parametreler ve başlangıç durumları rasgele olarak atanmıştır. Burada, 4 eğitim yöntemi de kullanılabilmektedir. Sonuçlar BFGS ile elde edilmiştir.

Bu aşamadan sonra, değişik yörüngeler için testler yapılacaktır. DSA parametreleri, verilen şartlar altında optimal kontrol problemini iyi yaklaşıklıkla çözebilecek şekilde ayarlanmış olmaktadır. Eğitim aşamasında kullanılmayan yörüngeler için test edilecektir. Ayrıca, eğitim seti dışındaki yörünge için de performansı incelenecektir. Burada 3 test yörüngesi denenecektir:

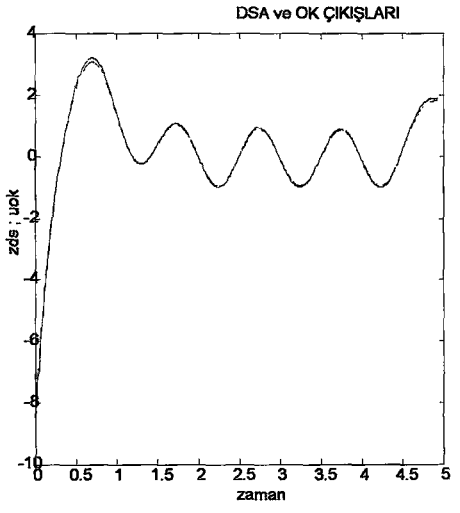
$x_t^d(t) = \begin{bmatrix} -1.5 & 0.5 & 2.5 \\ 0.0 & 0.0 & 0.0 \end{bmatrix} r(t)$ ve VDP test parametreleri $a_t = \begin{bmatrix} .7 & .3 & .01 \end{bmatrix}$ olmak üzere Şekil 7. 3'te sırayla bu yörüngeler için DSA çıkışları ve optimal yörüngeler verilmektedir. Görüldüğü gibi yüksek yaklaşıklı sonuçlar elde edilmiştir. Eğitim seti dışındaki yörünge yaklaşımı da kullanılabilir özelliktedir. Bu adımdan sonra, Şekil 7. 2'de gösterildiği gibi DSA çıkışı, optimal kontrolcüye başlangıç kontrolü olarak atanarak birkaç adımda optimal kontrol fonksiyonu bulunur. Ve GB yapısı da oluşturulur. GB durumları ve bozucu etkileri Bölüm 5'te gösterildiği gibidir.

Ayrıca, değişken bir yörünge için yapılan test başarılı olmuştur. İstenen yörüngenin sinüzoidal olması durumunda DSA cevabı çok iyi bir yaklaşım yapabilmiştir. 0.1 genlikli 1Hz'lik bir sinüs yörüngesi VDP sisteminde birinci durum olup istenilen yörünge $x^d(t) = \begin{bmatrix} 0.1\sin(2\pi t) & 0.2\pi\cos(2\pi t) \end{bmatrix}^T$ şeklinde DSA girişlerine uygulandığında, DSA çıkışı ve optimal sonuç Şekil 7. 4'te verilmiştir. Şekil 7. 5'te ise optimal VDP durum yörüngeleri verilmektedir. Bu uygulamada VDP sistem parametresi a=.5 olarak alınmıştır.

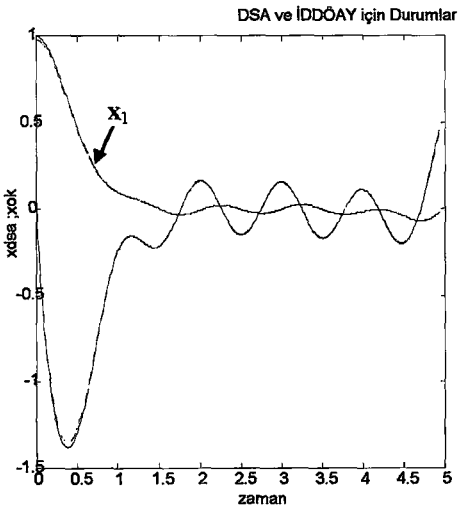
Bir başka uygulama olarak, değişik genlikli dikdörtgen darbeli hedef yörünge (1. durum değişkeni için) üzerinde çalışılacaktır. Bu uygulamada VDP parametresi a=.1 ve son zaman değeri t_f=15 olarak kullanılmıştır. Bu probleme ait DSA ve optimal kontrol çıkışları Şekil 7. 6'da gösterilmektedir. Aynı zamanda, hedef yörünge ve DSA'nın direkt olarak sisteme uygulanması sonucu meydana gelen cevap yörüngesi ise Şekil 7. 7'de gösterildiği gibidir. Görülmektedir ki, sabit yörüngeler için eğitilen DSA, değişken yörüngeler için de iyi sonuçlar vermiştir. Bu da, DSA kapasitesinin çok yüksek ve genelleştirme özelliğinin iyi olduğunu göstermektedir.



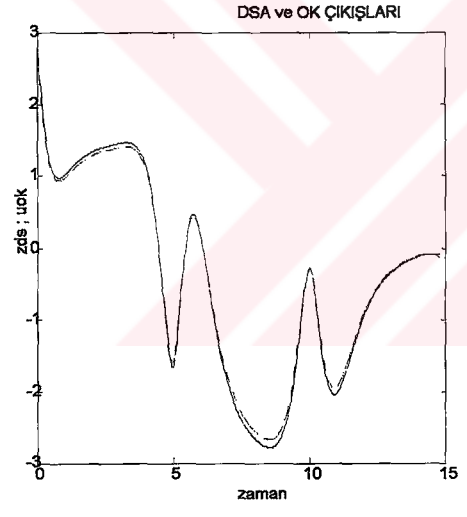
Şekil 7. 3 VDP sistemi için İBOK için DSA performansının test edilmesi (regülatör problemi, kesikli çizgi DSA çıkışı) (a) $x^d(t)=[-1.5\ 0]^T r(t)$;a=.7, (b) $x^d(t)=[0.5\ 0]^T r(t)$;a=.3, (c) $x^d(t)=[2.5\ 0]^T r(t)$;a=.01



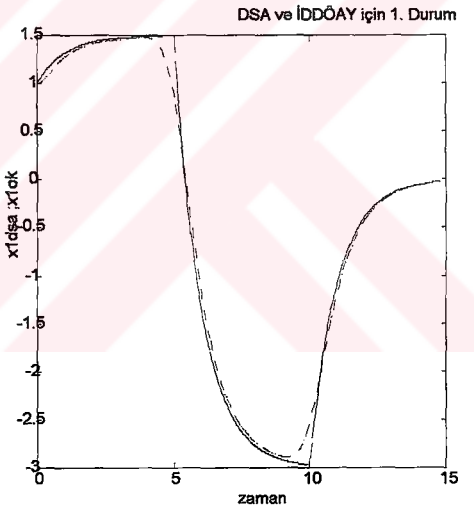
Şekil 7. 4 VDP sisteminde sinüzoidal yörünge için DSA performansının test edilmesi (kesikli çizgi DSA çıkışı)



Şekil 7. 5 Sinüzoidal yörünge için optimal VDP durum yörüngeleri (sürekli çizgi), DSA çıkışı doğrudan sisteme uygulandığında cevap yörüngeleri (kesikli çizgi)



Şekil 7. 6 VDP probleminde dikdörtgen test yörüngesi için DSA ve optimal kontrol çıkışları (kesikli çizgi DSA çıkışı)



Şekil 7. 7 VDP probleminde dikdörtgen (1.durum değişkeni için) test yörüngesi (sürekli çizgi), DSA direkt olarak sisteme uygulandığında cevap yörüngesi (kesikli çizgi)

7.3.2. DSOK olarak CSTR sisteminin kullanılması

DSA ile optimal kontrol yapısı bu bölümde CSTR sistemi üzerinde test edilecektir. Yine burada da, CSTR reaktör sıcaklığı için yörünge seti kullanılacaktır. Burada kullanılan parametre ve değişkenler aşağıda verilmiştir:

$Q = \begin{bmatrix} 0 & 0 \\ 0 & 30 \end{bmatrix}$, $R = 0.3$, $S = \begin{bmatrix} 0 & 0 \\ 0 & 30 \end{bmatrix}$, $t_0 = 0$, $t_f = 10$, $x(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ ve nominal CSTR sapma değişkenleri $B = 11$, $\beta = 1.5$, $\gamma = 20$, $Da = 0.135$ olarak alınmıştır. CSTR için bu çalışma bölge bölgesinin kritik kararsız bir bölge olduğu daha önceden belirtilmiştir. Kullanılan yörünge seti ise 250^0K-350^0K arasında olmak üzere:

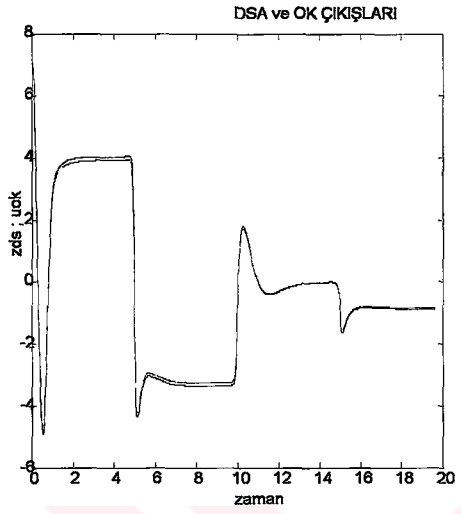
$[T^d(t)] = \begin{bmatrix} 250 & 270 & 290 & 310 & 330 & 350 \end{bmatrix} r(t)$ olarak giriş yörünge seti alınmıştır. İlk olarak, bu setlerden biri DSA girişine uygulanır. Ve ardından, DSA çıkışı İDDÖAY algoritmasına uygulanır. Sonra, iterasyon adımları sonunda algoritma optimal kontrol yörüngesini üretir. Bu yörünge DSA parametrelerinin güncellenmesinde kullanılır. Bu yaklaşım, DSA'dan gittikçe daha iyi sonuçlar alınıncaya kadar diğer yörüngeler için de tekrarlanır.

DSA için 3 nöron kullanılmıştır. Yukarıda verilen bilgilere göre dns=3 olarak belirlenir. DSA giriş sayısı 1, çıkış sayısı 1 olup toplam parametre sayısı 27 adettir. Bu uygulamada nöron bias parametreleri (b)'nin dışındaki tüm parametreler serbest bırakılarak 24 parametre ile optimizasyon algoritmaları çalıştırılmıştır.

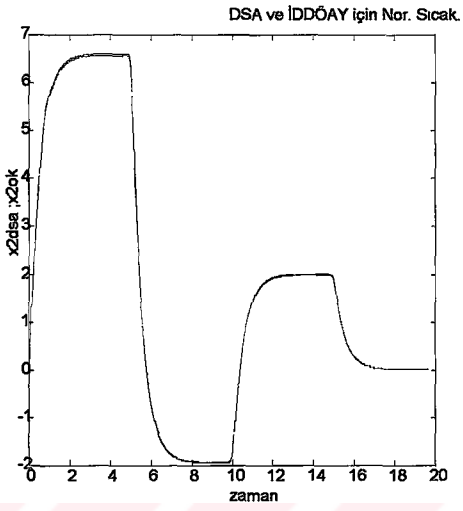
Burada da, gradyan hesaplamalarında adjoint yöntemi kullanılmıştır. Dinamik zaman sabiti T'ler için 0.1, sigmoid lineersizlik parametresi γ'lar için ise de 10^{-3} alt sınırlamaları yapılmıştır. Önceki örnekte olduğu gibi, DSA anlatılan tüm optimizasyon yöntemleriyle eğitilebilmektedir. Verilecek sonuçlar BFGS'e göre olacaktır.

Uygulama olarak, değişik genlikli dikdörtgen darbeli hedef yörünge (sıcaklık durum değişkeni için) üzerinde çalışılacaktır. Bu uygulamada son zaman değeri t_f=20 olarak kullanılmıştır. Bu probleme ait DSA ve optimal kontrol çıkışları Şekil 7. 8'de gösterilmektedir. Aynı zamanda, optimal sıcaklık ve DSA'nın direkt olarak sisteme uygulanması sonucu meydana gelen cevap yörüngeleri (normalize sıcaklık yörüngeleri) ise Şekil 7. 9'da gösterildiği gibidir. Ve son olarak, hedef yörünge ve DSA'nın direkt olarak sisteme uygulanması sonucu meydana gelen cevap yörüngeleri (sıcaklık yörüngeleri) ise Şekil 7. 10'da gösterildiği gibidir. Bu adımdan sonra OGB yapısı oluşturularak Bölüm 5'te incelenen durumlar gibi toplam zeki kontrol sistemi

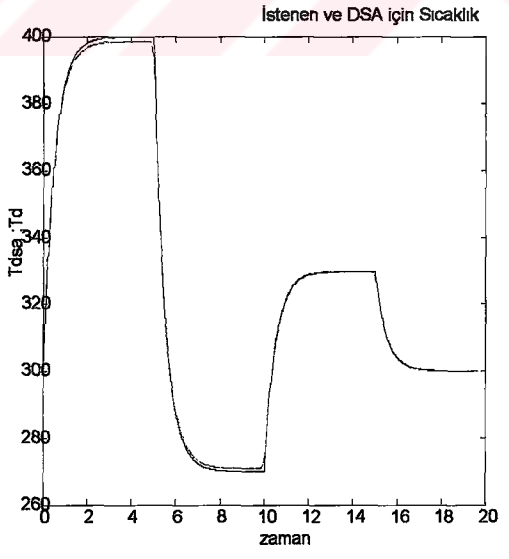
tamamlanır. DSA'nın iyi bir yaklaşım ve genelleme yaptığı buradan da gözlemlenmektedir.



Şekil 7. 8 CSTR sisteminde dikdörtgen test yörüngesi için DSA ve optimal kontrol çıkışları (kesikli çizgi DSA çıkışı)



Şekil 7. 9CSTR sisteminde dikdörtgen test yörüngesi (normalize sıcaklık durum değişkeni için) probleminde DSA çıkışı direkt olarak sisteme uygulandığında cevap yörüngesi (kesikli çizgi) ve İDDÖAY ile bulunan optimal yörünge (sürekli çizgi)



Şekil 7. 10 CSTR sisteminde dikdörtgen (sıcaklık durum değişkeni için) test yörüngesi (sürekli çizgi), DSA çıkışı direkt olarak sisteme uygulandığında cevap yörüngesi (kesikli çizgi)

BÖLÜM 8. SONUÇLAR

Bu tez çalışmasında, optimal kontrol ve DSA teknolojilerinin bir arada kullanılması ile oluşturulan zeki optimal kontrol sisteminin geliştirilmesi amaçlanmıştır. Optimal kontrol, lineer olmayan yapılar üzerinde 2. dereceden yöntemler için incelenmiştir. İDDÖAY, bu çalışmada üzerinde yoğun olarak çalışılan yöntemdir. Bu yöntemin oldukça gürbüz bir yapıda olduğu gösterilmiştir. Konjuge nokta problemi açısından iyi bir şekilde işlediği gösterilmiştir. Ayrıca, kontrol sınırlama ağırlık matrisi (W) yardımıyla bu problemin giderilebileceği İDHY için de gösterilmiştir. Bu matrisin iterasyon süresince otomatik olarak güncellenebilmesini sağlayan ve kuadratiklik özelliğinden faydalanılan bir yapı geliştirilmiştir. Bu özellik her iki yöntemde başarılı bir şekilde kullanılmıştır. İDDÖAY'nin İDHY'ne göre daha gürbüz ve hızlı olduğu işlediği gösterilmiştir. Bütün teorik sonuçlar VDP ve CSTR sistemleri üzerinde denenmiştir. 2. derece yöntemden GB yapısının direkt olarak elde edilmesi uygulama açısından da çok önemli bir çıkarımdır. Çıkarılan GB kanunu verilen şartlar altında optimal özelliğe sahiptir. GB'nin, değişik bozucular için yapılan denemelerde çok iyi bir şekilde sistemi istenilen yörüngede tuttuğu gösterilmiştir. Yapılan optimal kontrol algoritmaları fonksiyon uzayında çıkarılmıştır.

Son yıllarda kullanılan modern tekniklerden olan YSA, optimal kontrol için incelenmiştir. Bu çalışmada DSA yapısı kullanılmıştır. DSA yapısı ve ifadeleri detaylı olarak çıkarılmıştır. Aktivasyon fonksiyonu olarak kullanılan sigmoid fonksiyonu genel amaçlı olarak, lineersizlik ve bias parametreleriyle daha etkin hale getirilmiştir. Ve bu parametreler DSA parametreleri olarak düşünülüp güncellenebilmektedir. Dinamik zaman sabiti de bu amaçla kullanılmıştır. DSA, bir dinamik sistem olarak düşünüldüğünden problem parametre optimizasyonu olarak ele alınmıştır. PÖ'nün parametreye göre gradyanları adjoint metoduyla hesaplanmıştır. Bu yöntem vasıtasıyla, DSA derecesi kadar diferansiyel denklem çözerek bütün gradyanlar

bulunmuştur. Büyük bir hesaplama maliyet kazancı sağlamıştır. Eğitim algoritmaları olarak, EAGY, ÖKGY, DFP ve BFGS yöntemleri yarı-lineer dinamik bir sistem olan DSA eğitimine adapte edilerek geliştirilmiştir. ÖKGY, DFP ve BFGS yöntemlerinin oldukça iyi sonuçlar verdiği örnekler üzerinde gösterilmiştir.

Bu çalışmada, DSA'nın, optimal kontrol başlatıcısı olarak kullanılması hedeflenmiştir. Bu yapı için, İDDÖAY'inden elde edilen sonuçlarla DSA yukarıda söylenen yöntemlerle eğitilmiştir. Eğitimde kullanılmayan ve eğitim aralığı dışındaki denemeler için de (genelleme) oldukça iyi sonuçlar elde edilmiştir. Eğitim için sabit yörüngeler yani regülatör durumu kullanılmasına rağmen, sinüzoidal, dikdörtgen vb. gibi değişken yörüngeler için de çok iyi bir yaklaşımla DSA kontrol fonksiyonu üretmiştir. Daha sonra DSA'nın ürettiği kontrol fonksiyonu optimal kontrolcünün başlangıç kontrolü olarak bir kaç adımda optimal kontrol, durum yörüngesi ve GB yapısı oluşturulmuştur.

Ayrıca, sistem yapısı kendi kendine öğrenme biçiminde de kullanılabilmektedir. Bu özellikle beraber toplam sistem, bir zeki-hibrid yapıda çalışabilmektedir. Sonuçta oluşturulan yapı, gerçek zaman uygulamalarında kullanılmaya yöneliktir. Anlatılan bu özellikler, bilimsel değerlendirme açısından Tablo 8.1'de özetlenmiştir.

Bu tez çalışmasında anlatılan ve geliştirilen tüm teorik çalışmalar ve uygulama sonuçları MATLAB [120-123] programı yardımıyla simüle edilmiştir. Bu çalışmada geliştirilen simülasyon programı kaynak kodları, genel özellikleriyle beraber ilgili konu bağlantıları ve genel blok yapısı Ek D'de verilmiştir.

Tablo 8. 1 Tez nicelikleri ve bulguları

GENEL TEZ NİTELİKLERİ	İLGİLİ TEZ NİCELİKLERİ (BULGULARI)
Bilime Yenilik Getirme	• LOOK probleminin çözümünde, konjuge nokta problemini otomatik olarak ortadan kaldıran direkt 2. dereceden yöntem iyileştirilerek kullanılmıştır [Böl. 3]. Aynı zamanda, zamanla değişen optimal GB kazancı da yan ürün olarak üretilmektedir [Böl. 4]. • YSA olarak genelleştirilmiş aktivasyon fonksiyonuna sahip DSA modeli oluşturulup, adjoint metodu ile gradyan hesabı yapılarak hızlı optimizasyon yöntemleri eğitim yapısına adapte edilmiştir [Böl. 6]. • DSA, optimal kontrolcü için bir yörünge başlatıcısı olarak kullanılıp, sonuçta oluşturulan algoritma daha az iterasyon ile optimal kontrol fonksiyonu üretmektedir [Böl. 7]. • Hızlandırılmış yörünge hesaplamaları sayesinde, sanal global geri beslemeli gerçek zamandaki nöro-optimal kontrol uygulamalarına kapı açılmış olmaktadır
Yeni Bir Yöntem Getirme	• Direkt 2. dereceden optimal kontrol probleminin çözümünde kullanılan kontrol sınırlama matrisinin otomatik olarak güncellenmesini sağlayan bir yapı geliştirilmiş olup yapı 2. dereceden Hamiltonian metoduna da uygulanabilmektedir [Böl. 3]. Bu özellik algoritmayı dayanıklı (robust) ve hızlı bir hale getirmiştir. • DSA yapısında kullanılan tüm parametreler (lineer ve lineer olmayan) adjoint metodu kullanılarak tüm optimizasyon yöntemleriyle güncellenebilecegi gösterilmiştir [Böl. 6]. • Nöro-optimal kontrol otomatik öğrenmeli yapıda gerçekleştirilerek algoritma farklı dizgelere uygulanabilen bir ürün olarak kullanılabilmektedir [Böl. 7].
Bilinen Bir Yöntemi Yeni Bir Alana Uygulama	• DSA ve optimal kontrol teknolojileri bir arada kullanılarak, gerçek zamanlı uygulamalara yönelik hızlandırılmış yörünge üretimi yapısı geliştirilmiştir [Böl. 7]. • DSA gradyan hesabı için, adjoint metodu kullanılarak değişik optimizasyon yöntemleri eğitim aşamasında kullanılmıştır [Böl. 6].

BÖLÜM 9. TARTIŞMA VE ÖNERİLER

Bu tez çalışmasında sunulan bütün teorik sonuçlar, uygulamada açık sayısal yöntem ve yazılımlarla gerçekleştirilebilir. Özellikle, zamanda geriye doğru entegrasyonda simetriklik özelliğini çok iyi sağlayacak daha etkin sayısal yöntemler araştırılabilir. İkinci dereceden optimal kontrol yöntemleri hem İB kontrolü ve hem de GB kontrol kanununu oluşturduğu için çok iyi yapıya sahiptir. İDDÖAY ise diğerinden özellikle konjuge nokta yönünden daha gürbüzdür. Kontrol sınırlama ağırlık matrisinin (W) otomatik olarak güncellenmesi yapısı da etkili bir çözüm sunar. Ayrıca, konjuge nokta problemini de uygun seçim şartıyla ortadan kaldırabilmektedir. Burada, farklı bir güncelleme yapısı belirleyerek algoritma hızlandırılabilir. Önerilen bu yapının matematiksel çıkarımı da açık bir konudur.

DSA, yapısından gelen özelliklerle yüksek depolama kapasitesine sahiptir. Zaman sabiti ve artırılmış parametreli sigmoid fonksiyonuyla daha da zengin bir yapıya dönüştürülmüştür. Başka çeşit aktivasyon fonksiyonlarının performansları da incelenebilecek bir konudur. Gradyan hesabında adjoint metodu etkili bir şekilde kullanılmıştır. Bu çalışmada kullanılan eğitim algoritmaları geriye yayınım veya basit gradyan yöntemine göre hızlı işleyen yapılardadır. Bunların dışında geliştirilmiş optimizasyon yöntemleri de DSA eğitimine adapte edilebilir. Özellikle, tam ikinci derece yöntemi (Newton durumu) geliştirilip uygulanmaya değer yöntemdir.

DSA ve optimal kontrol bir arada, optimal kontrol başlatıcısı olarak ve kendi kendine öğrenebilen yapıda kullanılarak gerçek zamanlı uygulamalar için zeki-optimal kontrol yapısı oluşturulmuştur. Hızlandırılmış yörünge algoritmaları, sanal global geri beslemeli gerçek zamanlı zeki optimal kontrol için yol açacaktır. Sonuçta çok iyi yaklaşımlarla kontrol fonksiyonu elde edilebilmektedir. Eğitim esnasında, optimal kontrol algoritmasının iterasyonları sonucunda elde edilen yörüngelerle güncellemeler

yapılmıştır. Bunun dışında, her adımda elde edilen artırımsal veya yöresel yörüngeler DSA için kullanılabilecek bir bilgi olarak düşünülebilir. Bu, araştırılması gereken önemli bir yaklaşımdır.

KAYNAKLAR

[1] LEWIS, F.L.,: "Applied Optimal Control and Estimation", Prentice-Hall, Englewood Cliffs, New Jersey, 1992.

[2] KALMAN, R.E., and BERTRAM, J.E.,: "Control System Analysis and Design via the Second Method of Lyapunov I. Continuous-Time Systems", Trans. ASME J. Basic Eng., pp. 371-393, June 1960.

[3] KALMAN, R.E.,: "Contributions to the Theory of Optimal Control", Bol. Soc. Mat. Mexicana, vol.5, pp.102-119, 1960.

[4] KALMAN, R.E.,: "A New Approach to Linear Filtering and Prediction Problems", ASME J. Basic Eng., vol.82, pp.34-35, 1960.

[5] KALMAN, R.E., and BUCHY, R.S.,: "New Results in Linear Filtering and Prediction Theory", ASME J. Basic Eng., vol.80, pp.193-196, 1961.

[6] BRYSON, A.E. and HO, Y.C.,: "Applied Optimal Control", Hemisphere Publishing Corporation, 1975.

[7] BROGAN, W.L.,: "Modern Control Theory", Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1991.

[8] LAPIDUS, L., and LUUS, R.,: "Optimal Control of Engineering Processes", Blaisdell Publishing Company, 1966.

[9] KONAR, A.F., and MAHESH, J.K.,: Computer-Aided Engineering of Large Scale Process Control Systems", Proceedings of the Eleventh Annual Advanced Control Conference, West Lafayette, Indiana, 1985.

[10] MCLEAN, D.,: "Automatic Flight Control Systems", Prentice-Hall Int. Ltd., 1990.

[11] BLAKELOCK, J.H.,: "Automatic Control of Aircraft and Missiles", John Wiley & Sons, Inc., 1991.

[12] STEVENS, B.L., and LEWIS, F.L.,: "Aircraft Control and Simulation", John Wiley & Sons, Inc., 1992.

[13] KONAR, A.F., and WARD, M.D.,: "Development of Weapon Delivery Models and Analysis Programs", AFFDL-TR-71-123, vol.I,II,III, April 1972.

[14] KONAR, A.F., MAHESH, J.K., and KIZILOS, B.,: "Digital Flight Control Systems for Tactical Fighters", AFFDL-TR-73-119, June 1974.

[15] KONAR, A.F., ve BECERİKLİ, Y.,: "Bilgisayar Destekli Uçuş Kontrol Tasarımı ve XIMKON Yazılımı", Savunma Sanayiindeki Teknolojik Gelişmeler Sempozyumu, Ankara 1997.

[16] BULLOCK, T.E.,: "Computation of Optimal Controls by a Method Based on Second Variations", Department of Aeronautics and Astronautics, Stanford University, SUDAAR 297, December 1966.

[17] BULLOCK, T.E., and FRANKLIN, G.F.,: "A Second Order Feedback Method for Optimal Control Computations", IEEE, Trans. on Automatic Control, vol. AC-12, no.6, pp.666-673, December 1967.

[18] FRANKLIN, G:F.,: "Note on a Problem in Second Order Gradient Methods", IEEE, Trans. on Automatic Control (Correspondence), vol. AC-11, p.623, December 1966.

[19] MITTER, S.K.,: "Successive Approximation Methods for the Solution of Optimal Control Problems", Automatica, vol.3, pp.135-149, Pergamon Press, 1966.

[20] KOPP, R.E., and MOYER, H.G.,: "Trajectory Optimization Techniques", in Advances in Control Systems (C.T. Leondes, ed.), vol.4, pp.103-155, Academic Press, New York, 1966.

[21] BUSHNELL, L.G.,: "On the History on Control", IEEE, Control Systems Magazine, vol.16, no.3, pp.14-16, June 1996.

[22] BENNETT, S.,: "A Brief History of Automatic Control", IEEE, Control Systems Magazine, vol.16, no.3, pp.17-25, June 1996.

[23] BRYSON, A.E.,: "Optimal Control- 1950 to 1985", IEEE, Control Systems Magazine, vol.16, no.3, pp.26-33, June 1996.

[24] ATHERTON, D.P.,: "Early Development in Nonlinear Control", IEEE, Control Systems Magazine, vol.16, no.3, pp.34-43, June 1996.

[25] ÅSTRÖM, K.J.,: "Adaptive Control Around 1960", IEEE, Control Systems Magazine, vol.16, no.3, pp.44-49, June 1996.

[26] MICHEL, A.N.,: "Stability: The Common Thread in the Evolution of Feedback Control", IEEE, Control Systems Magazine, vol.16, no.3, pp.50-60, June 1996.

[27] ZAMES, G.,: "Input-Output Feedback Stability and Robustness, 1959-85", IEEE, Control Systems Magazine, vol.16, no.3, pp.61-66, June 1996.

[28] MITTER, S.K.,: "Filtering and Stochastic Control: A Historical Perspective",

IEEE, Control Systems Magazine, vol.16, no.3, pp.67-76, June 1996.

[29] SUSSMANN, H.J., and WILLEMS, J.C.,: "300 Years of Optimal Control: From the Brachystochrone to the Maximum Principle", IEEE, Control Systems Magazine, vol.17, no.3, pp.32-44, June 1997.

[30] WILLEMS, J.C.,: "On Interconnections, Control and Feedback", IEEE, Trans. on Automatic Control, vol.42, no.3, pp.326-339, March 1997.

[31] BEARD, R., SARIDIS, G., and WEN, J.,: "Improving the Performance of Stabilizing Controls for Nonlinear Systems", IEEE, Control Systems Magazine, vol.16, no.5, pp.27-35, October 1996.

[32] LEWIS, F.L.,: "Optimal Estimation with an Introduction to Stochastic Control Theory", John Wiley & Sons. Inc., 1986.

[33] ANDERSON, B.D.O., and MOORE, J.B.,: "Optimal Control Linear Quadratic Methods", Prentice-Hall Inter. Inc., Englewood Cliffs, NJ, 1989.

[34] DAVIS, M.H.A., and ZERVOS, M.,: "A New Proof of the Discrete Time LQG Optimal Control Theorems", IEEE, Trans. on Automatic Control, vol.40, no.8, pp.1450-1453, August 1995.

[35] ZHANG, C., and FU, M.,: "A Revisit to the Gain and Phase Margins of Linear Quadratic Regulators", IEEE, Trans. on Automatic Control, vol.41, no.10, pp.1527-1530, October 1996.

[36] SHAMMA, J.S., and XIONG, D.,: "Linear Nonquadratic Optimal Control", IEEE, Trans. on Automatic Control, vol.42, no.6, pp.875-879, June 1997.

[37] PINHO, M.do.R. de, and VINTER, R.B.,: "An Euler-Lagrange Inclusion for Optimal Control Problems", IEEE, Trans. on Automatic Control, vol.40, no.7, pp.1191-1198, July 1995.

[38] CALLIER, F.M., and WINKIN, J.J.,: "Asymptotic Behavior of the Solution of the Projection Riccati Differential Equation", IEEE, Trans. on Automatic Control, vol.41, no.5, pp.646-659, May 1996.

[39] WIMMER, H.K.,: "The Set of Positive Semidefinite Solutions of the Algebraic Riccati Equation of Discrete Time Optimal Control", IEEE, Trans. on Automatic Control, vol.41, no.5, pp.660-671, May 1996.

[40] LEE, C.H.,: "New Results for the Bounds of the Solution for the Continuous Riccati and Lyapunov Equations", IEEE, Trans. on Automatic Control, vol.42, no.1, pp.118-123, January 1997.

[41] GERSTNER, A.B., and FABBENDER, H.,: "A Jacobi-Like Method for Solving Algebraic Riccati Equation on Parallel Computers", IEEE, Trans. on Automatic

Control, vol.42, no.8, pp.1071-1084, August 1997.

[42] IONESCU, V., OARĂ, C., and WEISS, M.,: "General Matrix Pencil Techniques for the Solution of Algebraic Riccati Equations: A Unified Approach", IEEE, Trans. on Automatic Control, vol.42, no.8, pp.1085-1097, August 1997.

[43] LEE, C.H.,: "Eigenvalue Upper Bounds of the Solution of the Continuous Riccati Equation", IEEE, Trans. on Automatic Control, vol.42, no.9, pp.1268-1271, September 1997.

[44] LAUB, A.J., and GAHINET, P.,: "Numerical Improvements for Solving Riccati Equations", IEEE, Trans. on Automatic Control, vol.42, no.9, pp.1303-1308, September 1997.

[45] BENNER, P., LAUB, A.J., and MEHRMANN, V.,: "Benchmarks for the Numerical Solution of Algebraic Riccati Equations", IEEE, Control Systems Magazine, vol.17, no.5, pp.18-28, October 1997.

[46] MERRIAM III, C.W.,: "Optimization Theory and the Design of Feedback Control Systems", Mc Graw-Hill, Inc., 1964.

[47] HASDORFF, L.,: "Gradient Optimization and Nonlinear Control", John Wiley & Sons., Inc., 1976.

[48] PIERRE, D.A.,: "Optimization Theory with Applications", Dover Publication, Inc., 1986.

[49] CHRISTENSEN, G.S., EL-HAWARY, M.E., and SOLIMAN, S.A.,: "Optimal Control Applications in Electric Power System", Plenum Press, 1987.

[50] WANG, Q., SPEYER, J.L., and DEWELL, L.O.,: "Regulators for Optimal Periodic Processes", IEEE, Trans. on Automatic Control, vol.40, no.10, pp.1767-1778, October 1995.

[51] LIPPMANN, R.P.,: "An Introduction to Computing with Neural Nets", IEEE, ASSP Magazine, pp.4-22, April 1987.

[52] HUSH, D.R., and HORNE, B.,: "Progress in Supervised Neural Networks, What's New Since Lippmann", IEEE, Signal Processing Magazine, pp.8-39, January 1993.

[53] NARENDA, K.S., and PARTHASARATHY, K.,: "Gradient Methods for the Optimization of Dynamical Systems Containing Neural Networks", IEEE, Trans. on Neural Networks, vol.2, no.2, pp.252-262, March 1991.

[54] QIN, S.Z., SU, H.T., and MCAVOY, T.J.,: "Comparison of Four Neural Net Learning Methods for Dynamic System Identification", IEEE, Trans. on Neural Networks, vol.3, no.1, January 1992.

[55] GEVA, S., and SITTE, J.,: "A Constructive Method for Multivariate Function Approximation by Multilayer Perceptrons", IEEE, Trans. on Neural Networks, vol.3, no.4, pp.621-624, July 1992.

[56] NERRAND, D., RAUSSEL-RAGOT, P., URBANI, D., PERSONNAZ, L., and DREYFUS, G.,: "Training Recurrent Neural Networks: Why and How? An Illustration in Dynamical Process Modelling", IEEE, Trans. on Neural Networks, vol.5, no.2, March 1994.

[57] OLUROTIMI, O.,: "Recurrent Neural Network Training with Feedforward Complexity", IEEE, Trans. on Neural Networks, vol.5, no.2, pp.185-197, March 1994.

[58] YU, X.H., CHEN, G.A., and CHENG, S.X.,: "Dynamic Learning Rate Optimization of the Backpropagation Algorithm", IEEE, Trans. on Neural Networks, vol.6, no.3, May 1995.

[59] MICHEL, A.N., WANG, K., LIU, D., and YE, H.,: "Qualitative Limitations Incurred in Implementations of Recurrent Neural Networks", IEEE, Control Systems Magazine, vol.15, no.3, pp.52-65, June 1995.

[60] KUZUOĞLU, M., and LEBLEBİCİOLU, K.,: "Infinite-Dimensional Multilayer Perceptrons", IEEE, Trans. on Neural Networks, vol.7, no.4, pp.889-896, July 1996.

[61] PARISI, R., DICLAUDID, E.D., ORLANDI, G., and RAO, B.D.,: "A Generalized Learning Paradigm Exploiting the Structure of Feedforward Neural Networks", IEEE, Trans. on Neural Networks, vol.7, no.6, pp.1450-1460, November 1996.

[62] MCLOONE, S., and IRWIN, G.W.,: "Fast Parallel Off-Line Training of Multilayer Perceptrons", IEEE, Trans. on Neural Networks, vol.8, no.3, pp.646-653, May 1997.

[63] CICHOCKI, A., and UNBEHAUEN, R.,: "Neural Networks for Optimization and Signal Processing", John Wiley & Sons. Ltd. & B.G. Teubner, 1993.

[64] AGARWAL, M.,: "A Systematic Classification of Neural Network-Based Control", IEEE, Control Systems Magazine, vol.17, no.2, pp.75-93, April 1997.

[65] NARENDA, K.S., and PARTHASARATHY, K.,: "Identification and Control of Dynamical Systems Using Neural Networks", IEEE, Trans. on Neural Networks, vol.1, no.1, pp.4-27, March 1990.

[66] CHEN, F.C.,: "Back-Propagation Neural Networks for Nonlinear Self-Tuning Adaptive Control", IEEE, Control Systems Magazine, pp.44-48, April 1990.

[67] NGUYEN, D.H., and WIDROW, B.,: "Neural Networks for Self-Learning

Control Systems", IEEE, Control Systems Magazine, pp.18-28, April 1990.

[68] HUNT, K.J., SBARBARO, D., ŻBIKOWSKI, R., and GAWTHROP, P.J.,: "Neural Networks for Control Systems- A Survey", Automatica, vol.28, no.6., pp.1083-1112, 1992.

[69] LEVIN, A.U., and NARENDA, K.S.,: "Control of Nonlinear Dynamical Systems Using Neural Networks: Controllability and Stabilization", IEEE, Trans. on Neural Networks, vol.4, no.2, pp.192-206, March 1993.

[70] SARTORI, M.A., and ANTSAKLIS, P.J.,: "Implementation of Learning Control Systems Using Neural Networks", IEEE, Control Systems Magazine, pp.49-57, April 1992.

[71] HOSKINS, D.A., HWANG, J.N., and VAGNERS, J.,: "Iterative Inversion of Neural Networks and Its Application to Adaptive Control", IEEE, Trans. on Neural Networks, vol.3, no.2, pp.292-301, March 1992.

[72] KARAKAŞOĞLU, A., SUDHARSANAN, S.I. and SUNDARESHAN, M.K.,: "Identification and Decentralized Adaptive Control Using Dynamical Neural Networks with Application to Robotic Manipulators", IEEE, Trans. on Neural Networks, vol.4, no.6, pp.919-930, November 1993.

[73] TESHNEHLAB, M., and WATANABE, K.,: "Self Tuning of Computed Torque Gains by Using Neural Networks with Flexible Structures", IEE, Proc. Control Theory Appl., vol.141, no.4, pp.235-242, July 1994.

[74] YUH, J.,: "Learning Control for Underwater Robotic Vehicles", IEEE, Control Systems Magazine, pp.39-46, April 1994.

[75] KU, C.C., and LEE, K.Y.,: "Diagonal Recurrent Neural Networks for Dynamic Systems Control", IEEE, Trans. on Neural Networks, vol.6, no.1, pp.144-156, January 1995.

[76] LIGHTBODY, G., and IRWIN, G.W.,: "Direct Neural Model Reference Adaptive Control", IEE, Proc. Control Theory Appl., vol.142, no.1, pp.31-43, January 1995.

[77] MILLER, W.T., SUTTON, R.S., and WERBOS, P.J.,: "Neural Networks for Control", The MIT Press, 1992.

[78] HARRIS, C.J.,: "Advances in Intelligent Control", Taylor & Francis Ltd., 1994.

[79] HUNT, K., IRWIN, G., and WARWICK, K.,: "Neural Network Engineering in Dynamic Control Systems", Springer-Verlag London Ltd., 1996.

[80] OMATU, S., KHALID, M., and YUSOF, R.,: "Neuro-Control and Its Applications", Springer-Verlag London Ltd., 1996.

[81] PHAM, D.T., and XING, L.,: "Neural Networks for Identification, Prediction and Control", Springer-Verlag London Ltd., 1997.

[82] FAROTIMI, O., DEMBO, A., and KAILATH, :"A General Weight Matrix Formulation Using Optimal Control", IEEE, Trans. on Neural Networks, vol.2, no.3, pp.378-394, May 1991.

[83] PARISINI, T., and ZOPPOLI, R.,: "Neural Networks for Feedback Feedforward Nonlinear Control Systems", IEEE, Trans. on Neural Networks, vol.5, no.3, pp.436-449, May 1994.

[84] YEN, V., LIU, T.Z., and LIU, D.Y.,: "Neural-Network-Based Near-Time-Optimal Position Control Method for DC Motor Servo Systems", IEE, Proc. Control Theory Appl., vol.142, no.5, pp.493-500, September 1995.

[85] NIESLER, T.R., and DU PLESIS, J.J.,: "Time-Optimal Control by Means of Neural Networks", IEEE, Control Systems Magazine, vol.15, no.5, pp.23-33, October 1995.

[86] PLUMER, E.S.,: "Optimal Control of Terminal Processes Using Neural Networks", IEEE, Trans. on Neural Networks, vol.7, no.2, pp.408-418, March 1996.

[87] MOHLER, R.R., and ZAKRZEWSKI, R.R.,: "Suboptimal Intelligent Control: Agile Aircraft Application", IEEE, Control Systems Technology, vol.4, no.4, pp.363-368, July 1996.

[88] PARK, Y.M., CHOI, M.S., and LEE, K.Y.,: "An Optimal Tracking Neuro-Controllers for Nonlinear Dynamic Systems", IEEE, Trans. on Neural Networks, vol.7, no.5, pp.1099-1110, September 1996.

[89] BERSINI, H., and GORRINI, V.,: "A Simplification of the Backpropagation-Through-Time Algorithm for Optimal Neuro Control", IEEE, Trans. on Neural Networks, vol.8, no.2, pp.437-441, March 1997.

[90] LJUNG, L.,: "System Identification", Prentice-Hall, Inc., 1987.

[91] CLAUSEN, S.T.,: "System Identification and Robust Control, A Case Study Approach", Springer-Verlag London Ltd., 1996.

[92] SCALES, L.E.,: "Introduction to Nonlinear Optimization", Springer-Verlag New York Inc., 1985.

[93] ARORA, J.S.,: "Introduction to Optimum Design", McGraw-Hill, Inc., 1989.

[94] PRESS, W.H., TEUKOLSKY, S.A., VETTERLING, W.T., and FLANNERY, B.P.,: "Numerical Recipes in C", Cambridge University Press, 1995.

[95] CHONG, E.K.P., and ŻAK, S.H.,: "An Introduction to Optimization", John Wiley & Sons., Inc., 1996.

[96] KONAR, A.F.,: "Gradient and Curvature in Nonlinear Identification", Proc. Honeywell Advanced Control Workshop, January 1991.

[97] KONAR, A.F., and SAMAD, T.,: "Dynamic Neural Networks", Technical Report SSDC-92-I 4152-2, Honeywell Technology Center, 3660 Technology Drive, Minneapolis, MN 55418, 1992.

[98] KONAR, A.F., and BECERİKLİ, Y., and SAMAD, T.,: "Trajectory Tracking with Dynamic Neural Networks", IEEE, Int. Sym. On Intelligent Control (ISIC'97), İstanbul 1997.

[99] BURDEN, R.L., and FAIRES, J.D.,: "Numerical Analysis", PWS Publishers, 1985.

[100] ATKINSON, K.E.,: "An Introduction to Numerical Analysis", John Wiley & Sons, Inc., 1989.

[101] CONSTANTINIDES, A.,: "Applied Numerical Methods with Personal Computers", McGraw-Hill, 1988.

[102] CHAPRA, S.C., and CANALE, R.P.,: "Numerical Methods for Engineers", McGraw-Hill, Inc., 1989.

[103] TUMA, J.J.,: "Handbook of Numerical Calculations in Engineering", McGraw-Hill, Inc., 1989.

[104] NAKAMURA, S.,: "Applied Numerical Methods with Software", Englewood Cliffs, N.J., Prentice-Hall, Inc., 1991.

[105] GLASSEY, R.,: "Numerical Computation Using C", Academic Press, Inc., 1993.

[106] VLACH, J., and SINGHAL, K.,: "Computer Methods for Circuit Analysis and Design", Van Nostrand Reinhold, International Thomson Publishing, Inc., 1994.

[107] ISIDORI, A.,: "Nonlinear Control Systems", Springer-Verlag London Ltd., 1995.

[108] KHALIL, H.K.,: "Nonlinear Systems", Prentice-Hall, Inc., 1996.

[109] KONAR, A.F.,: "Optimal Kontrol Teorisi", Ders Notları, Sakarya Üniversitesi, Sakarya, 1995.

[110] RAY, W.H.,: "New Approaches to the Dynamics of Nonlinear Systems with Implications for Process and Control System Design", in Chemical Process Control II,

Edgar Seabor Edit., pp.245-267, 1981.

[111] LUYBEN, W.L.,: "Process Modelling, Simulation and Control for Chemical Engineers", McGraw-Hill, Inc., 1990.

[112] KONAR, A.F., et al,: "Dynamic Process Simulation and Control", Honeywell Inc., CSDD, Minnesota, 1987.

[113] KONAR, A.F.,: "Nöral Kontrol Teorisi ve Uygulamaları", Ders Notları, Sakarya Üniversitesi, Sakarya, 1996.

[114] KOSKO, B.,: "Neural Networks and Fuzzy Systems", Prentice-Hall, Inc., Englewood Cliffs, N.J., 1992.

[115] LI, H., and GUPTA, M.,: "Fuzzy Logic and Intelligent Systems", Kluwer Academic Publishers, 1995.

[116] JANG, J-S.R., TSAI, C., and MIZUTANI, E.: "Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence", Prentice-Hall, Inc., N.J., 1997.

[117] KONAR, A.F.,: "Optimizasyon Teorisi ve Uygulamaları", Ders Notları, Sakarya Üniversitesi, Sakarya, 1997.

[118] SHEPHERD, A.J.,: "Second-Order Methods for Neural Networks", Springer-Verlag London Ltd., 1997.

[119] GILL, P.E., MURRAY, W., and WRIGHT, M.H.,: "Practical Optimization", Academic Press Ltd., 1993.

[120] Matlab, High Performance Numeric Computation and Visualization Software, Reference Guide, The MathWorks, Inc., 1992.

[121] Matlab, High Performance Numeric Computation and Visualization Software, User's Guide, The MathWorks, Inc., 1993.

[122] Simulink, Dynamic Simulation Software, The MathWorks, Inc., 1993.

[123] FREDERICK, D.K. and CHOW, J.H.,: "Feedback Control Problems Using Matlab and The Control System Toolbox", PWS Publishing Company, 1995.

EK A

DEĞİŞİMLERİN HESABI İLE LOOK PROBLEMİ İÇİN GEREKLİ KOŞULLARIN ÇIKARTILMASI

(3.8) eşitliği göz önüne alınırsa fark ifadesi yazılabilir:

$$\Delta J_a = J_a(x+\delta x, u+\delta u, \lambda+\delta\lambda) - J_a(x,u,\lambda) \qquad \text{(Ek A1)}$$

Burada ΔJ_a, küçük artırımlı J_a ile J_a arasındaki farktır. Gerekli ifadeler yerine yazılırsa aşağıdaki denklem elde edilir:

$$\Delta J_a = \phi[x(t_f)+\delta x(t_f)] + \int_{t_0}^{t_f}[H(x+\delta x, u+\delta u, \lambda+\delta\lambda) - (\lambda+\delta\lambda)^T(\dot{x}+\delta\dot{x})]dt - \phi[x(t_f)] - \int_{t_0}^{t_f}[H(x,u,\lambda) - \lambda^T\dot{x}]dt \qquad \text{(Ek A2)}$$

Son ifade birinci dereceye kadar Taylor serisine açılırsa aşağıdaki ifade elde edilir:

$$\Delta J_a = \phi[x(t_f)] + \phi_x^T(t_f)\delta x(t_f) + \int_{t_0}^{t_f}[H(x,u,\lambda) + H_x^T\delta x + H_u^T\delta u + H_\lambda^T\delta\lambda]dt - \int_{t_0}^{t}[\lambda^T\dot{x} + \lambda^T\delta\dot{x} + \delta\lambda^T\dot{x}]dt + (y.d.t) - \phi[x(t_f)] - \int_{t_0}^{t_f}[H(x,u,\lambda) - \lambda^T\dot{x}]dt \qquad \text{(Ek A3)}$$

Burada y.d.t., yüksek dereceli terimleri ifade eder. Gerekli sadeleştirmeler yapıldıktan sonra ifade aşağıdaki gibi yazılır:

$$\Delta J_a = \phi_x^T(t_f)\delta x(t_f) + \int_{t_0}^{t_f}[H(x,u,\lambda) + H_x^T\delta x + H_u^T\delta u + H_\lambda^T\delta\lambda]dt - \int_{t_0}^{t}[\lambda^T\dot{x} + \lambda^T\delta\dot{x} + \delta\lambda^T\dot{x}]dt + (y.d.t.) \quad \text{(Ek A4)}$$

(Ek A4) ifadesinde sadece birinci terimler alınırsa, J_a'nın birinci varyasyonu bulunmuş olur:

$$\delta J_a = \phi_x^T(t_f)\delta x(t_f) + \int_{t_0}^{t_f}[H(x,u,\lambda) + H_x^T\delta x + H_u^T\delta u + H_\lambda^T\delta\lambda]dt - \int_{t_0}^{t}[\lambda^T\dot{x} + \lambda^T\delta\dot{x} + \delta\lambda^T\dot{x}]dt \quad \text{(Ek A5)}$$

Bu ifadede $\int \lambda^T\delta\dot{x}dt$, kısmi integrasyon kuralı ile çözülürse şu ifadeye ulaşılır:

$$\int_{t_0}^{t_f}\lambda^T\delta\dot{x}dt = \lambda^T(t_f)\delta x(t_f) - \lambda^T(t_0)\delta x(t_0) - \int_{t_0}^{t_f}\dot{\lambda}^T\delta x dt \quad \text{(Ek A6)}$$

Burada, başlangıç zamanı t_0 sabit kabul edilirse $\delta x(t_0) = 0$ olur. Bu durumda,

$$\int_{t_0}^{t_f}\lambda^T\delta\dot{x}dt = \lambda^T(t_f)\delta x(t_f) - \int_{t_0}^{t_f}\dot{\lambda}^T\delta x dt \quad \text{(Ek A7)}$$

bulunur. (Ek A7), (Ek A5)'te yerine yazılırsa,

$$\delta J_a = (\phi_x^T(t_f) - \lambda^T(t_f))\delta x(t_f) + \int_{t_0}^{t_f}[(H_x^T + \dot{\lambda}^T)\delta x + H_u^T\delta u + (H_\lambda^T - \dot{x}^T)\delta\lambda]dt \quad \text{(Ek A8)}$$

eşitliği bulunmuş olur. İstenilen çözümün PÖ'deki artımın bitmesi yani $\delta J_a = 0$ olması demektir. Bunun için, küçük değişimler olan bağımsız değişkenlere bağlı olmadan, şart

ancak $(\delta x, \delta u, \delta \lambda)$'nın katsayı terimlerinin sıfırlanmasıyla elde edilir. Aynı zamanda bu, gerekli koşulların çıkartılması anlamına gelir:

$$\dot{x} = H_{\lambda} \quad , \quad x(t_0) = x_0 \qquad \text{(Ek A9)}$$

$$\dot{\lambda} = -H_x \quad , \quad \lambda(t_f) = \phi_x(t_f) \qquad \text{(Ek A10)}$$

$$0 = H_u \qquad \text{(Ek A11)}$$

Burada, (Ek A9) dinamik durum şartı, (Ek A10) ek durum şartı ve (Ek A11) ise durağanlık veya optimallik şartlarıdır.

EK B

HAMILTONIAN PRENSİBİNE DAYALI İKİNCİ DERECEDEN LOOK PROBLEMİNİN ÇÖZÜMÜ

Bolza problemi için önce Hamiltonian yazılırsa,

$$H(x,u,\lambda) = L(x,u) + \lambda^T(t) f(x,u) \qquad \text{(Ek B1)}$$

APÖ yazılırsa,

$$J_a = \phi(x(t_f)) + \int_{t_0}^{t_f} [H(x,u,\lambda) - \lambda^T \dot{x}]dt \qquad \text{(Ek B2)}$$

J_a, (x,u,λ) yörüngeleri civarında ikinci dereceye kadar Taylor serisine açılırsa,

$$\begin{aligned} J_a(x+\delta x, u+\delta u, \lambda+\delta\lambda) &= \phi(x(t_f)) + \phi_x^T(t_f)\delta x(t_f) + \tfrac{1}{2}\delta x^T(t_f)\phi_{xx}\delta x(t_f) \\ &+ \int_{t_0}^{t_f} [H + H_x^T\delta x + H_u^T\delta u + H_\lambda^T\delta\lambda + \tfrac{1}{2}(\delta x^T H_{xx}\delta x + \delta x^T H_{xu}\delta u + \delta u^T H_{ux}\delta x \\ &+ \delta u^T H_{uu}\delta u + \delta x^T H_{x\lambda}\delta\lambda + \delta\lambda^T H_{\lambda x}\delta x + \delta u^T H_{u\lambda}\delta\lambda + \delta\lambda^T H_{\lambda u}\delta u) \\ &- \lambda^T\dot{x} - \lambda^T\delta\dot{x} - \delta\lambda^T\dot{x} - \delta\lambda^T\delta\dot{x}]dt + (y.d.t.) \end{aligned} \qquad \text{(Ek B3)}$$

Burada, $H_{\lambda x}=H_{x\lambda}$ ve $H_{u\lambda}=H_{\lambda u}$ eşitlikleri vardır. Şimdi de, fark ifadesi yazılırsa,

$$\Delta J_a = J_a(x+\delta x, u+\delta u, \lambda+\delta\lambda) - J_a(x,u,\lambda) \qquad \text{(Ek B4)}$$

$$\Delta J_a = \delta J + \delta^2 J = \phi_x^T(t_f)\delta x(t_f) + \tfrac{1}{2}\delta x^T(t_f)\phi_{xx}\delta x(t_f) + \int_{t_0}^{t_f}[H_x^T\delta x + H_u^T\delta u$$

$$+ H_\lambda^T\delta\lambda + \tfrac{1}{2}\begin{bmatrix}\delta x^T & \delta u^T\end{bmatrix}\begin{bmatrix}H_{xx} & H_{xu}\\ H_{ux} & H_{uu}\end{bmatrix}\begin{bmatrix}\delta x\\ \delta u\end{bmatrix} + \delta\lambda^T H_{\lambda x}\delta x + \delta\lambda^T H_{\lambda u}\delta u \qquad \text{(Ek B5)}$$

$$- \lambda^T\delta\dot{x} - \delta\lambda^T\dot{x} - \delta\lambda^T\delta\dot{x}]dt + (y.d.t.)$$

Burada, $\int_{t_0}^{t_f}\lambda^T\delta\dot{x}dt$ ifadesi, kısmi integrasyonla bulunursa,

$$\int_{t_0}^{t_f}\lambda^T\delta\dot{x}dt = \lambda^T(t_f)\delta x(t_f) - \int_{t_0}^{t_f}\dot{\lambda}^T\delta x dt \qquad \text{(Ek B6)}$$

Burada, t_0 sabit olduğundan $\delta x(t_0) = 0$ alınmıştır. Fark tekrar düzenlenirse,

$$\Delta J_a = (\phi_x^T(t_f) - \lambda^T(t_f))\delta x(t_f) + \tfrac{1}{2}\delta x^T(t_f)\phi_{xx}\delta x(t_f) + \int_{t_0}^{t_f}[(H_x^T + \dot{\lambda}^T)\delta x$$

$$+ (H_\lambda^T - \dot{x}^T)\delta\lambda + H_u^T\delta u + \delta\lambda^T(H_{\lambda x}\delta x + H_{\lambda u}\delta u - \delta\dot{x}) \qquad \text{(Ek B7)}$$

$$+ \tfrac{1}{2}\begin{bmatrix}\delta x^T & \delta u^T\end{bmatrix}\begin{bmatrix}H_{xx} & H_{xu}\\ H_{ux} & H_{uu}\end{bmatrix}\begin{bmatrix}\delta x\\ \delta u\end{bmatrix}]dt + (y.d.t.)$$

Burada, $H_{\lambda x}=f_x$ ve $H_{\lambda u}$'dur. Son ifadeden birinci varyasyon yazılırsa,

$$\delta J = (\phi_x^T(t_f) - \lambda^T(t_f))\delta x(t_f) + \int_{t_0}^{t_f}[(H_x^T + \dot{\lambda}^T)\delta x + (H_\lambda^T - \dot{x}^T)\delta\lambda + H_u^T\delta u \qquad \text{(Ek B8)}$$

$$+ \delta\lambda^T(f_x\delta x + f_u\delta u - \delta\dot{x})]dt$$

Burada, $x(t_f)$ serbest olduğundan $\delta x(t_f) \neq 0$ olup $\lambda(t_f) = \phi_x(t_f)$'tir. İfade, diğer

artırımsal değişkenlere bağlı olmaması için gerekli koşullar çıkarılırsa,

$$\dot{x} = H_{\lambda} \quad , \quad x(t_0) = x_0 \tag{Ek B9}$$

$$\dot{\lambda} = -H_x \quad , \quad \lambda(t_f) = \phi_x(t_f) \tag{Ek B10}$$

Geriye kalan ifade olan H_u, ancak optimal noktada sağlanacağından,

$$\delta J = \int_{t_0}^{t_f} H_u^T \delta u dt \tag{Ek B11}$$

olur. Ayrıca yerel model için,

$$\delta\dot{x} = f_x \delta x + f_u \delta u \quad , \quad \delta x(t_0) = 0 \tag{Ek B12}$$

kolayca yazılabilir. İkinci varyasyon yazılırsa,

$$\delta^2 J = \tfrac{1}{2}\delta x^T(t_f)\phi_{xx}\delta x(t_f) + \tfrac{1}{2}\int_{t_0}^{t_f}\begin{bmatrix}\delta x^T & \delta u^T\end{bmatrix}\begin{bmatrix}H_{xx} & H_{xu}\\ H_{ux} & H_{uu}\end{bmatrix}\begin{bmatrix}\delta x\\ \delta u\end{bmatrix}dt \tag{Ek B13}$$

bulunur. Fark ifadesinin son hali tekrar yazılırsa ve $\frac{1}{2}\|\delta u\|_W$ integral sınırlaması eklenerek aşağıdaki ifadeye ulaşılır:

$$\begin{aligned}\Delta J_a = {} & \tfrac{1}{2}\delta x^T(t_f)\phi_{xx}\delta x(t_f) + \int_{t_0}^{t_f}[H_u^T\delta u \\ & + \tfrac{1}{2}\begin{bmatrix}\delta x^T & \delta u^T\end{bmatrix}\begin{bmatrix}H_{xx} & H_{xu}\\ H_{ux} & H_{uu}+W\end{bmatrix}\begin{bmatrix}\delta x\\ \delta u\end{bmatrix}]dt + (\text{y.d.t.})\end{aligned} \tag{Ek B14}$$

Şimdi (y.d.t.)'ler ihmal edilerek problem yerel optimizasyon problemine dönüşmüştür:

$$\min_{\delta u} \Delta J_a(\delta u) = \tfrac{1}{2}\delta x^T(t_f)\phi_{xx}\delta x(t_f) + \int_{t_0}^{t_f} \tilde{L}(\delta x, \delta u)dt \qquad \text{(Ek B15)}$$

sınırlama. $f_x\delta x + f_u\delta u - \delta\dot{x} = 0 \quad , \quad \delta x(t_0) = 0$ (Ek B16)

Bu ikinci derece (accessory) veya yerel optimizasyon problemini çözmek için artırımsal Hamiltonian tanımlanırsa,

$$\tilde{H}(\delta x, \delta u, \delta\lambda) = \tilde{L}(\delta x, \delta u) + \delta\lambda^T(f_x\delta x + f_u\delta u) \qquad \text{(Ek B17)}$$

$$\tilde{H}(\delta x, \delta u, \delta\lambda) = H_u^T\delta u + \tfrac{1}{2}\left[\delta x^T \quad \delta u^T\right]\begin{bmatrix} H_{xx} & H_{xu} \\ H_{ux} & H_{uu} + W \end{bmatrix}\begin{bmatrix}\delta x \\ \delta u\end{bmatrix} + \delta\lambda^T(f_x\delta x + f_u\delta u) \qquad \text{(Ek B18)}$$

bulunabilir. Bu problem için gerekli koşullar yazılırsa,

$$\delta\dot{x} = \tilde{H}_{\delta\lambda} = f_x\delta x + f_u\delta u \quad , \quad \delta x(t_0) = 0 \qquad \text{(Ek B19)}$$

$$\delta\dot{\lambda} = -\tilde{H}_{\delta x} = -(H_{xx}\delta x + H_{xu}\delta u + f_x^T\delta\lambda) \quad , \quad \delta\lambda(t_f) = \phi_{xx}\delta x(t_f) \qquad \text{(Ek B20)}$$

$$0 = -\tilde{H}_{\delta u} = H_u + H_{ux}\delta x + (H_{uu} + W)\delta u + f_u^T\delta\lambda \qquad \text{(Ek B21)}$$

elde edilir. Son eşitlikten δu bulunabilir:

$$\delta u(t) = -(H_{uu} + W)^{-1}(H_u + H_{ux}\delta x + f_u^T\delta\lambda) \qquad \text{(Ek B22)}$$

Bu ifade yerine yazılırsa,

$$\delta\dot{x} = [f_x - f_u(H_{uu} + W)^{-1}H_{ux}]\delta x - [f_u(H_{uu} + W)^{-1}f_u^T]\delta\lambda$$
$$- f_u(H_{uu} + W)^{-1}H_u \quad , \quad \delta x(t_0) = 0 \qquad \text{(Ek B23)}$$

$$\delta\dot{\lambda} = -[H_{xx} - H_{xu}(H_{uu} + W)^{-1}H_{ux}]\delta x - [f_x^T - H_{xu}(H_{uu} + W)^{-1}f_u^T]\delta\lambda$$
$$- H_{xu}(H_{uu} + W)^{-1}H_u \quad , \quad \delta\lambda(t_f) = \phi_{xx}\delta x(t_f) \qquad \text{(Ek B24)}$$

elde edilir. Bu eşitlikler İNSDP olup çözüm için genel Riccati dönüşümü kullanılır:

$$\delta\lambda = P\delta x + q \qquad \text{(Ek B25)}$$

$$\delta\dot{x} = A(t)\delta x + B(t)\delta\lambda + v(t) \quad , \quad \delta x(t_0) = 0 \qquad \text{(Ek B26)}$$

$$\delta\dot{\lambda} = -C(t)\delta x - A^T(t)\delta\lambda - w(t) \quad , \quad \delta\lambda(t_f) = \phi_{xx}\delta x(t_f) \qquad \text{(Ek B27)}$$

Burada, katsayı matris ve vektörleri (3.76)-(3.80) eşitlikleriyle verildiği gibidir. P ve q'nun diferansiyel denklemleri için (Ek B25)'in türevi alınarak aşağıdaki ifadeler bulunur:

$$\dot{P} = -(A^TP + PA + PBP + C) \quad , \quad P(t_f) = \phi_{xx}(t_f) \qquad \text{(Ek B28)}$$

$$\dot{q} = -(A^Tq + PBq + Pv + w) \quad , \quad q(t_f) = 0 \qquad \text{(Ek B29)}$$

Artırımsal kontrol fonksiyonu düzenlenirse aşağıdaki ifadeler elde edilir:

$$\delta u(t) = -(H_{uu} + W)^{-1}[(H_{ux} + f_u^TP)\delta x + (H_u + f_u^Tq)] \qquad \text{(Ek B30)}$$

$$\delta u(t) = K(t)\delta x + b(t) \qquad \text{(Ek B31)}$$

Burada,

$$K(t) = -(H_{uu} + W)^{-1}(H_{ux} + f_u^T P) \qquad \text{(Ek B32)}$$

$$b(t) = -(H_{uu} + W)^{-1}(H_u + f_u^T q) \qquad \text{(Ek B33)}$$

Sonuçta elde edilen yapı, geri-besleme yapısında olup bu yapı, hem çözümlemede hem de geri-besleme için kullanılacaktır.

EK C

DİNAMİK SİSTEMLERDE ADJOINT DUYARLILIK ANALİZİ

Burada, varyasyonel hesap yardımıyla [68,96,109] adjoint duyarlılık analizi geliştirilecektir. Dinamik, lineer olmayan bir sistemin, parametrelerinin varyasyonunun verilen bir PÖ üzerindeki etkisini bulmak problemin çözümüne ulaşmak için yapılacak çalışmadır. PÖ, şu şekilde verilsin:

$$J = \int_0^{t_f} F(x,t)dt \qquad \text{(Ek C1)}$$

ve dinamik sınırlama,

$$\dot{x} = f(x,p) \quad , \quad x(t_0) = x_0 \qquad \text{(Ek C2)}$$

Burada, $p \in R^m$ m boyutlu vektör, $x \in R^n$ n boyutlu vektör fonksiyonudur. Parametrelerdeki değişim arandığından, p'ye göre varyasyon bulunacaktır ve dinamik sisteme kontrol fonksiyonunun uygulandığı varsayılmaktadır. PÖ ve dinamik sistem üzerindeki varyasyonlar şu şekilde yazılabilir:

$$\delta J = \int_0^{t_f} \left(\frac{\partial F}{\partial x}\right)^T \delta x dt \qquad \text{(Ek C3)}$$

$$\delta\dot{x} = f_x \delta x + f_p \delta p \quad , \quad \delta x(0) = 0 \qquad \text{(Ek C4)}$$

Burada x(0)'ın bilindiği kabul edilmektedir.

Şimdi de artırılmış performans ölçütü (APÖ) tanımlanırsa,

$$J_a = \int_0^{t_f} [F(x,t) + \lambda^T (f(x,p) - \dot{x})]dt \qquad \text{(Ek C5)}$$

Burada $\lambda \in R^n$ olup Lagrange çarpanıdır. APÖ için birinci varyasyon yazılırsa aşağıdaki ifadeler elde edilir:

$$\delta J_a = \int_0^{t_f} \left[\left(\frac{\partial F}{\partial x} \right)^T \delta x + \lambda^T \left(\frac{\partial f}{\partial x} \delta x + \frac{\partial f}{\partial p} \delta p - \delta \dot{x} \right) \right] dt \qquad \text{(Ek C6)}$$

$$\delta J_a = \int_0^{t_f} \left[\left(\left(\frac{\partial F}{\partial x} \right)^T + \lambda^T \frac{\partial f}{\partial x} \right) \delta x + \lambda^T \frac{\partial f}{\partial p} \delta p \right] dt - \int_0^{t_f} \lambda^T \delta \dot{x} dt \qquad \text{(Ek C7)}$$

Burada, ikinci integral kısmi integrasyonla bulunursa,

$$\int_0^{t_f} \lambda^T \delta \dot{x} dt = \lambda^T(t_f)\delta x(t_f) - \lambda^T(t_0)\delta x(t_0) - \int_0^{t_f} \dot{\lambda}^T \delta x dt \qquad \text{(Ek C8)}$$

Bu eşitlik yerine yazılırsa ve $\delta x(0)=0$ alınırsa,

$$\delta J_a = \int_0^{t_f} \left[\left(\left(\frac{\partial F}{\partial x} \right)^T + \lambda^T \frac{\partial f}{\partial x} + \dot{\lambda}^T \right) \delta x + \lambda^T \frac{\partial f}{\partial p} \delta p \right] dt - \lambda^T(t_f)\delta x(t_f) \qquad \text{(Ek C9)}$$

Varyasyonun bağımsız olabilmesi için yukarıdaki eşitlikten,

$$-\dot{\lambda} = \left(\frac{\partial f}{\partial x} \right)^T \lambda + \left(\frac{\partial F}{\partial x} \right) \quad , \quad \lambda(t_f) = 0 \qquad \text{(Ek C10)}$$

olarak adjoint (Lagrange) diferansiyel denklemi bulunur. Geriye kalan varyasyon ise,

$$\delta J_a = \int_0^{t_f} \left(\lambda^T \frac{\partial f}{\partial p} \delta p \right) dt \qquad \text{(Ek C11)}$$

olur. Ve parametre duyarlılığı son ifadeden şu şekilde yazılabilir:

$$\frac{\partial J_a}{\partial p} = \int_0^{t_f} \left(\lambda^T \frac{\partial f}{\partial p} \right) dt \qquad \text{(Ek C12)}$$

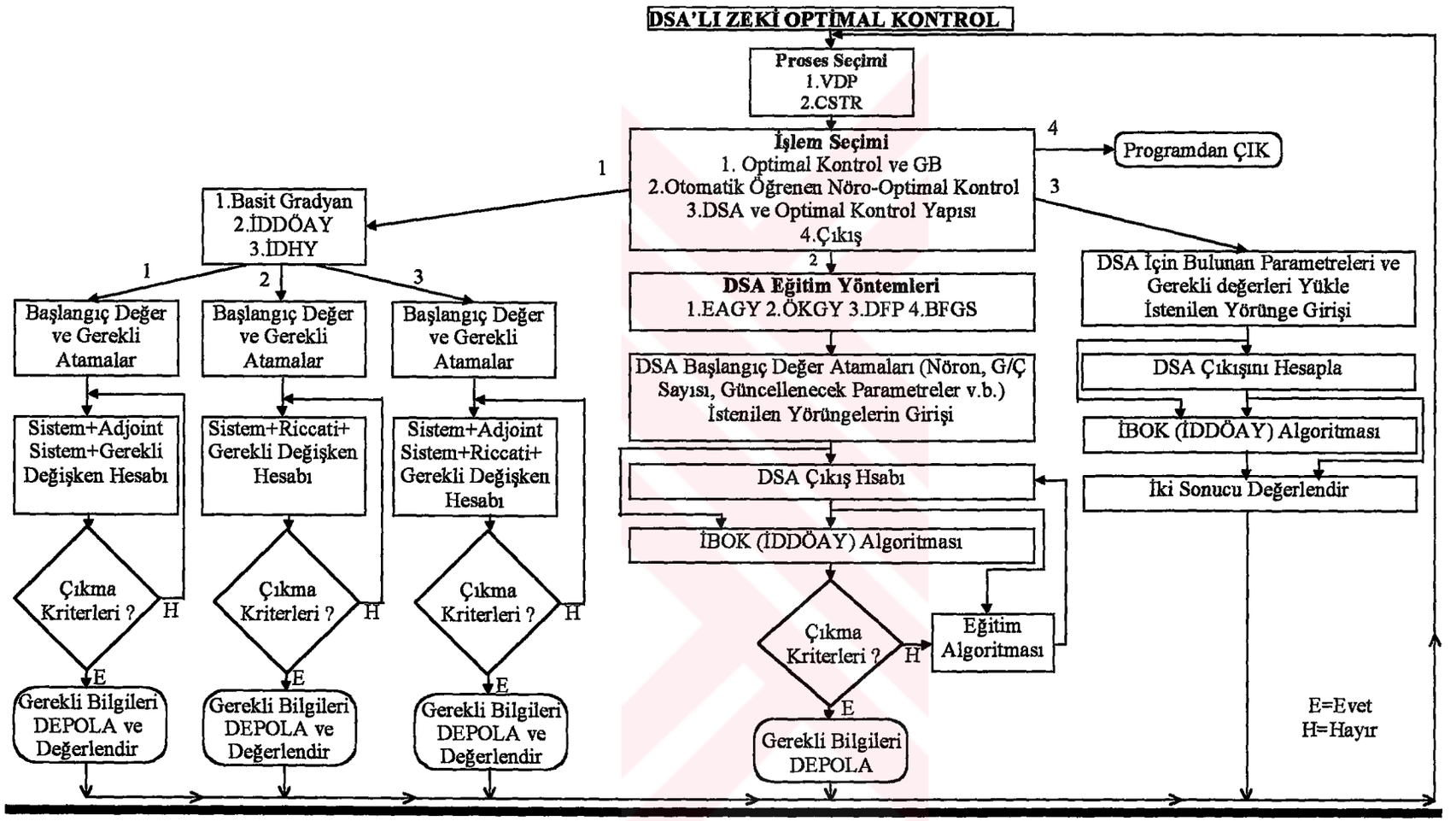
Görüldüğü gibi, gradyan hesabı, parametre sayısından bağımsız olup sadece sistem derecesine bağlıdır.

EK D

ZEKİ OPTİMAL KONTROL PROGRAMI

Bu tez çalışmasındaki teorik sonmuçlar bilgisayar programlama ile gerçekleştirilmiş ve basit bir benzetim paketi bu ekte sunulmuştur. Şekil D.1 programın akış diyagramını göstermektedir. İlgili program modüllerinin protitipi MATLAB [120-123] sisteminde geliştirilmiştir. Kullanılan kaynak kodları aşağıda listelenmiştir. Tablo D. 1'de yazılan fonksiyonların genel özellikleri ve konuya bağlantı yapılması açısından ilgili sayfa numaraları verilmiştir.

Başka platformlarda ve daha da genelleştirilmiş zeki-optimal kontrol programının geliştirilmesi bu tez kapsamının dışında kalmıştır. Bu bağlamda, Intelligent Control-IKON program paketinin gerçekleştirilmesi için çalışmalar ARGE porojesi olarak devam etmektedir.



Şekil D.1 Zeki optimal kontrol yazılımı blok diyagramı

Tablo D. 1 Yazılımda kullanılan fonksiyonların özellikleri ve ilgili konuyla bağlantıları

Fonksiyon adı	Genel Özellikleri	Sayfa	
		Teori/ Uygulama	Program
ydtmain.m	Yapılan çalışmanın ana program parçası. Üç ana bölümden oluşur: • Optimal kontrol ve geri-besleme • DSA eğtimi ve nöro-optimal kontrol • Optimal kontrol ve DSA'ların birlikte kullanımı	22,45,93, 121	160
ydtoc.m	Sürekli lineer olmayan optimal kontrol algoritmaları ve geri-besleme yapısının oluşturulması. Bu fonksiyon verilen ölçüte göre 3 optimal kontrol yöntemiyle problemi çözer: • Basit gradyan yöntemi (BDGY) • Doğrudan 2. derece ölçüt azaltma yöntemi (İDDÖAY) • 2. derece Hamiltonian yöntemi (İDHY)	24,29,38	163
ysmplgr.m	Basit gradyan (BDGY) için Adjoint, Hamiltonian ve kontrol fonksiyonunun hesabi	24,25,26	171
yxlmvp.m	Van der Pol (VDP) sistemine ait adjoint diferansiyel denklemi	52	172
yxlmcs.m	CSTR sistemine ait adjoint diferansiyel denklemi	84	172
yxvp.m	VDP sistemi diferansiyel denklemi	49	173
yxcs.m	CSTR sistemi diferansiyel denklemi	72	173
yxfbvp.m	Geri-besleme yapısındaki VDP diferansiyel denklemi	56	174
yxfbcs.m	Geri-besleme yapısındaki CSTR diferansiyel denklemi	81	174
ypricvp.m	VDP sistemine ait İDDÖAY'deki Riccati matris diferansiyel denklemi	54	175
ypriccs.m	CSTR sistemine ait İDDÖAY'deki Riccati matris diferansiyel denklemi	79	175
yqricvp.m	VDP sistemine ait İDDÖAY'deki vektör diferansiyel denklemi	54,55	175
yqriccs.m	CSTR sistemine ait İDDÖAY'deki vektör diferansiyel denklemi	79	176
ypricvph.m	VDP sistemine ait İDHY'deki Riccati matris diferansiyel denklemi	60,61	176
ypriccsh.m	CSTR sistemine ait İDHY'deki Riccati matris diferansiyel denklemi	84	177
yqricvph.m	VDP sistemine ait İDHY'deki vektör diferansiyel denklemi	61	177
yqriccsh.m	CSTR sistemine ait İDHY'deki vektör diferansiyel denklemi	84	178
ygerdnvp.m	VDP sistemi için 2. Derece yöntemlerde geri besleme denemeleri	63	178
ygerdncs.m	CSTR sistemi için 2. Derece yöntemlerde geri besleme denemeleri	84	179
ygraf.m	VDP ve CSTR sistemleri için İDDÖAY ile elde edilen sonuçların grafiklerinin çizimleri	58-60 81-83	184
ygrafh.m	VDP ve CSTR sistemleri için İDHY ile elde edilen sonuçların grafiklerinin çizimleri	63,64 84	184
ycsod.m	VDP ve CSTR sistemleri için 2. Derece yöntemlerdeki kontrol, durum ve ölçütün ityerasyonla değişimlerinin grafiklerinin çizimleri	58-60 63-64 81-84	185
ydtdnn.m	Bu fonksiyon "yoptim.m" optimizasyon fonksiyonunu kullanarak 4 optimizasyon yöntemi ile DSA modelini eğitir	93	186
yoctrain.m	DSA parametrelerinin başlangıç değerlerinin rasgele atanması	99	188

Fonksiyon adı	Genel Özellikleri	Sayfa	
		Teori/ Uygulama	Program
y2dnden.m	2-dinamik nöronlu bir sistemle diğer 2-dinamik nöronlu sistemin modellenmesi uygulaması için model ve hedef DSA parametre atanması	110	188
ygenden.m	N-dinamik nöronlu bir sistemle diğer N-dinamik nöronlu modellenmesi uygulaması için (genel bir örnek için) model ve hedef DSA parametrelerinin rasgele atanması	99	189
yxdn.m	DSA için durum diferansiyel denklemi	94-95	190
yoptim.m	Bu fonksiyon 4 optimizasyo yöntemini icra eder: i) EAGY ii) ÖKGY iii) DFP iv)BFGS	102-105	190
ylisrch.m	Tek boyutlu arama algoritması. Kübik yaklaşımlı interpolasyon kullanılıyor	102	194
yciz_yuk.m	Optimizasyon algoritmaları için ilgili sistemin çıkışlarının çizimi ve parametrelerin depolanmasını yapar	110-118	196
ydncost.m	DSA'ları için ölçüt, durum ve çıkışların hesabı	94,98	197
ydngrad.m	DSA'ların adjoint yöntemine dayalı ölçüt gradyan hesabı	101	198
yxdnlam.m	DSA için adjoint diferansiyel denklem sistemi	100	199
ytrsh_p.m	DSA'da zaman sabiti ve sigmoid lineer olmama parametresi için alt sınırlama	95	199
yrk5.m	Butcher'in 5. dereceden sabit adımlı Runga-Kutta metodu ile nümerik zamanda ileri ve geri lineer olmayan adi diferansiyel denklem çözümü	27,95	200
yinteg.m	Simpson'un (1/3) composite nümerik integral alma algoritması	37	201
ysigmoid.m	2 parametreli genel sigmoid fonksiyonu	95	201

```
% ***OPTİMAL KONTROLDA DİNAMİK SİNİR AĞLARI (DSA) İLE***
%                    ***YÖRÜNGE KESTİRİMİ***

% Program 3 ana bölümü içerir.
% (i) Optimal Kontrol ve geri-besleme
% (ii) DSA için eğitimi (parametre tahmini) ve nöro-optimal kontrol
% (iii) Optimal Kontrol ve DSA'larının birlikte kullanılması

% Yaşar BECERİKLİ- 14:11 05.11.1997

clear all; clc;

while 1, clear;
global ap;%VDP parametresi
global Gama; global Beta; global Da;%CSTR parametreleri
global BB; global dd; global x2c; global Tf;  %

format long e; format compact; global sil; global plant; clc;

disp('OPTİMAL KONTROL VE DİNAMİK SİNİR AĞLARI'); pause; clc;
disp('UYGULANACAK PROCESS');
disp('1=VDP ; 2=CSTR');
plant=input(':'); plantt=plant; planttt=plant; plants=plant; plantss=plant;

if plant==1, plant
disp('VDP Bünye Parametresi ?');
ap=input(':');

ubas=0.0; tnot=0; tfinal=5; tf=tfinal; ndata=501; nd=ndata; xnot=[1;0];
%xdd=[-2 -1 0 1 2]';  xcarp=[1 0]';
app=ap;
%app=[1 .5 .1]';
xdd=[0]'; xcarp=[1 0]';
end;%END OF plant=1

if plant==2, plant
disp('CSTR Parametreleri ?');
disp('B (7-8-11) ?'); ap1=input(':');
disp('Beta (.5-.3-1.5) ?'); ap2=input(':');
disp('Da (.11-.072-.135) ?'); ap3=input(':'); ap=[ap1;ap2;ap3]; aap=ap; app=ap;
ubas=1.0; tnot=0; tfinal=20; tf=tfinal; ndata=2001;
xnot=[.25;300];
%xdd=[317 320 330 340 350 380]'; xcarp=[0 1]';
app=ap;
%          B       Beta        Da
%app=[7 8 11;.5 .3 1.5;.11 .072 .135];
xdd=[280]'; xcarp=[0 1]';
end;%END OF plant=2
hdel=(tfinal-tnot)/(ndata-1); tbas=tnot; tson=tfinal;

disp('BİR İŞLEMİ SEÇİNİZ:');
disp('1=1. ve 2. DERECEDEN OPTİMAL KONTROL VE GERİ BESLEME');
disp('2=DİNAMİK SİNİR AĞI (DSA) EĞİTİMİ VE NÖRO-OPTİMAL KONTROL');
disp('3=DİNAMİK SİNİR AĞI (DSA) VE OPTİMAL KONTROL YAPISI');
disp('0=ÇIKIŞ');
islem=input(':');
if islem==0, clc; break; end;
```

```
%%OPTİMAL KONTROL
if islem==1, clc; sil=1;
disp(''); disp('OPTİMAL KONTROL YÖNTEMİ');
disp('1=BASİT GRADYAN'); disp('2=DİRECT 2nd ORDER COST DECENT (İDDÖAY)');
disp('3=2nd ORDER HAMILTANIAN (İDHY)');
yont=input(':');
xdpattern=[]; upattern=[]; xpattern=[];
[mxa,mxb]=size(xdd); [mxpa,mxpb]=size(app);
if plant==2, mxpa=mxpb; end;
for asay=1:mxpa,
for ksay=1:mxa, if plant==1, ap=app(asay), xddk=xdd(ksay), end;
   if plant==2, ap=app(:,asay), xddk=xdd(ksay), end;
 for tu=tbas:hdel:tson,
  itt=round(tu/hdel+1);
if plant==1,
xdt(:,itt)=xdd(ksay)*xcarp;
apd(:,itt)=app(asay);
%if tu<=4,  xdt(:,itt)=1.5*xcarp; end;
%if tu>4 & tu<=8,  xdt(:,itt)=-1*xcarp; end;
%if tu>8 & tu<=12,  xdt(:,itt)=0*xcarp; end;
%  xdt(:,itt)=.5*[sin(2*pi*tu);0];
end;
%CSTR
if plant==2,
apd(:,itt)=app(:,asay);
%if tu<=5, xdt(:,itt)=(320-(320-xnot(2))*exp(-2*tu))*xcarp; end; if tu==5, xdte=xdt(2,itt); end;
%if tu>5 & tu<=10, xdt(:,itt)=(280-(280-xdte)*exp(2*5-2*tu))*xcarp; end; if tu==10,
xdte=xdt(2,itt); end;
%if tu>10 & tu<=15, xdt(:,itt)=(340-(340-xdte)*exp(2*10-2*tu))*xcarp; end; if tu==15,
xdte=xdt(2,itt); end;
%if tu>15 xdt(:,itt)=(300-(300-xdte)*exp(2*15-2*tu))*xcarp; end;

xdt(:,itt)=(xdd(ksay)-(xdd(ksay)-xnot(2))*exp(-tu))*xcarp;
%xdt(:,itt)=xdd(ksay)*xcarp;
  end;
 end;
 xdesired=xdt;

for kk=1:ndata, ut(1,kk)=ubas; end; unot=ut;

[t,uoc,xoc]=ydtoc(unot,xnot,tnot,tfinal,ndata,xdesired,yont); clc;
if plant==2, xdesired=[xdesired(1,:);(xdesired(2,:)-300)*20/300]; end;
xdpattern=[xdpattern;xdesired;apd]; if plant==2, xdpattern=xdpattern([2 5],:); end;
upattern=[upattern;uoc]; xpattern=[xpattern;xoc];
end; end;
save xdpattern; save upattern; save xpattern;
end;%%END OF İŞLEM=1

%%DNN EĞİTİMİ
if islem==2, clc; disp('');
disp('EĞİTİM YÖNTEMİNİ SEÇİNİZ:');
disp('1=STEEPEST DESCENT GRADYAN (SDG) (BACK-PROPAGATION-BP)');
disp('2=SCALED CONJUGATE GRADYAN (SCG)');
disp('3=DAVIDON-FLETCHER-POWEL (DFP)');
disp('4=BROYDEN-FLETCHER-GOLFARB-SHANNO (BFGS)');
yont=input(':');
clc; disp(''); disp('SEÇİM');
```

```
disp('1=OPTİMAL KONTROL EĞİTİMİ'); disp('2=DENEMELER'); dene=input(':');
clc; disp('');
disp('1=GRAFİKLERDE DUR');
dur=input(':');
if plant==1, pattern=15; dneu=3; gsay=3; csay=1; end;
if plant==2, pattern=6; dneu=3; gsay=1; csay=1; end;

%BAŞLANGIÇ ATAMASI
xtdn0=[];
for tu=1:dneu,
if dneu==1, xtdn0(:,tu)=1; elseif tu==1, xtdn0(:,tu)=1.0; elseif tu==2, xtdn0(:,tu)=0.5;
elseif tu==3, xtdn0(:,tu)=0.1; end; end;

if dene==2, pattern=1; xdnnot=[1 1]'; else xdnnot=xtdn0'; end;

save xdnnot;
[prdn,prrdn,Edn,taoiterdn,iterdn]=ydtdnn(dneu,gsay,csay,xdnnot,tnot,tfinal,ndata,dene,yont,dur,
pattern); save prdn; save prrdn; save Edn; save taoiterdn; save iterdn;

end;%%END OF İŞLEM=2

%%DNN VE OPTİMAL KONTROL
if islem==3, clc;
if plant==1, pattern=1; dneu=3; gsay=3; csay=1; end;
if plant==2, pattern=1; dneu=3; gsay=1; csay=1; end;
pattern=1; dene=3; deneme=3; yont=1;
t=tnot:hdel:tfinal;
disp('');
if plant==1, disp('İSTENİLEN SİSTEM ÇIKIŞI (VDP)'); end;
if plant==2, disp('İSTENİLEN SİSTEM ÇIKIŞI (CSTR)'); end;
isc=input(':');

 for tu=tbas:hdel:tson,
  itt=round(tu/hdel+1);
if plant==1,
%  utt(:,itt)=isc*xcarp;
utt(:,itt)=[.1*sin(2*pi*tu);.2*pi*cos(2*pi*tu)];

%if tu<=5, xdt(:,itt)=(1.5-(1.5-xnot(1))*exp(-tu))*xcarp; end; if tu==5, xdte=xdt(1,itt); end;
%if tu>5 & tu<=10, xdt(:,itt)=(-3-(-3-xdte)*exp(5-tu))*xcarp; end; if tu==10, xdte=xdt(1,itt); end;
%if tu>10, xdt(:,itt)=(0-(0-xdte)*exp(10-tu))*xcarp; end;
%utt=xdt;
end;

%CSTR
if plant==2,
if tu<=5, xdt(:,itt)=(400-(400-xnot(2))*exp(-2*tu))*xcarp; end; if tu==5, xdte=xdt(2,itt); end;
if tu>5 & tu<=10, xdt(:,itt)=(270-(270-xdte)*exp(2*5-2*tu))*xcarp; end; if tu==10, xdte=xdt(2,itt);
end;
if tu>10 & tu<=15, xdt(:,itt)=(330-(330-xdte)*exp(2*10-2*tu))*xcarp; end; if tu==15,
xdte=xdt(2,itt); end;
if tu>15, xdt(:,itt)=(300-(300-xdte)*exp(2*15-2*tu))*xcarp; end;
utt=xdt;
% utt(:,itt)=(isc-(isc-xnot(2))*exp(-2*tu))*xcarp;
end; end;
udn=utt; save udn; load xdnnot;
```

```
dur=0;
%DNN
dene=deneme
[prdn,prrdn,Edn,taoiterdn,iterdn]=ydtdnn(dneu,gsay,csay,xdnnot,tnot,tfinal,ndata,dene,yont,dur,
pattern);

hold on;clf;
plot(t,prdn,'k'); grid;
title('DSA KONTROL KESTİRİMİ');
xlabel('zaman'); ylabel('zdn=u0');
pause2(0.1);
hold off;

%%OPTİMAL KONTROL
sil=1; tbas=tnot; tson=tfinal; xdesired=utt; unot=prdn; clc; disp('');
disp('OPTİMAL KONTROL YÖNTEMİ');
disp('1=SİMPLE GRADYAN');
disp('2=DİRECT 2nd ORDER COST DECENT');
disp('3=2nd ORDER HAMILTANIAN ');
yont=input(':');
[t,uoc,xoc]=ydtoc(unot,xnot,tnot,tfinal,ndata,xdesired,yont);

hold on; clf; plot(t,uoc,'k',t,prdn,':k'); title('DSA ve OK ÇIKIŞLARI');
xlabel('zaman'); ylabel('zds ; uok'); pause2(0.1); hold off; pause;

if plant==1, hold on; clf; plot(t,xoc,'k',t,xprdn,':k'); title('DSA ve İDDÖAY için Durumlar');
xlabel('zaman'); ylabel('xdsa ;xok'); pause2(0.1); hold off; pause;
end;

if plant==2, hold on; clf; plot(t,xoc(2,:),'k',t,xprdn(2,:),':k'); title('DSA ve İDDÖAY için Sıcaklık');
xlabel('zaman'); ylabel('x2dsa ;x2ok'); pause2(0.1); hold off; pause;

hold on; clf; plot(t,xdesired(2,:),'k',t,Trdn,':k'); title('İstenen ve DSA için Sıcaklık');
xlabel('zaman'); ylabel('Tdsa ;Td'); pause2(0.1); hold off; pause;
end;

end;%%END OF İŞLEM=3

end;%%END OF while

.....
% ***SÜREKLİ LİNEER OLMAYAN OPTİMAL KONTROL (SLOOK) ALGORİTMALARI***
%              ***VE GERİ-BESLEME***
function [t,u,x]=ydtoc(unot,xnot,tnot,tfinal,ndata,xdesired,yontem)

% Bu fonksiyon verilen şartlar (ÖLÇÜT fonksiyoneli) altında 3 tane Lineer
% olmayan OPTİMAL KONTROL yöntemleriyle OPTİMAL kontrol ve yörüngeyi hesaplar
% Fonsiyona girilen değişkenler:
% unot: İterasyona başlamak için başlangıç kontrol fonksiyonu
% xnot: İlgili sistemin başlangıç koşulları
% tnot, tfinal: Sırayla, başlangıç ve son zaman
% ndata: Problem çözümünde  kullanılacak veri sayısı. (Bu sayıdan diferansiyel denklem
%çözümündeki 'delta' integrasyon adımı bulunur)
% xdesired: İlgili sistemin gitmesi istenen yörünge
% yontem: SLOOK çözüm yöntemleri: 1= Basit Gradyan
%                                 2= Direk 2.derece Ölçüt Azaltma
%                                 3= 2.derece Hamiltonian
```

```
% Fonsiyondan dönen değişkenler:
% t: Sistemin zaman değerleri
% u: Verilen ölçütü en küçük yapan OPTİMAL KONTROL vektör fonksiyonu
% x: Verilen ölçütü en küçük yapan OPTİMAL YÖRÜNGE vektör fonksiyonu

% Yaşar BECERİKLİ- 21:29 17.03.1998

global sil; global ozA; global ozC; global ozP;
global t0; global tf; global nd; global u; global x; global xpr; global xqr; global Q; global R; global S;
global xd; global ww; global lm; global K; global b; global xxs; global dis; global tdis; global plant;

global ap;%VDP parametresi
global Gama; global Beta; global Da;%CSTR parametreleri
global BB; global dd; global ds; global x2c; global Tf;  %

u=unot; x0=xnot; t0=tnot; tf=tfinal; nd=ndata; xd=xdesired; hp=(tf-t0)/(nd-1); hn=-hp;
secy=yontem;
if plant==1,
Q=1*[10 0;0 1]; R=.1; S=1*[10 0;0 10]; end;%END OF plant=1
if plant==2,
 Cs0=0.25; Ts=300; Tc=300; CAf=0.25; Tf=300;
 V=48; F=40; To=V/F;
 Tf0=300; Tc0=300;
x2s=(Ts-Tf0)*Gama/Tf0; x2c=(Tc0-Tf0)*Gama/Tf0;
Da=20; BB=ap(1), Beta=ap(2), Da=ap(3)
for tu=t0:hp:tf,
  if tu>=40, Tff=Tf+0; else Tff=Tf; end;
  if tu<=10, Tffs=Tf+30; end;
  if tu>10 & tu<=20, Tffs=Tf-20; end;
  if tu>20, Tffs=Tf+5; end;
    it=round(tu/hp+1); ds(it)=(Tffs-Tf0)*Gama/Tf0; dd(it)=(Tff-Tf0)*Gama/Tf0;
 end;

x0=[(CAf-x0(1))/CAf (x0(2)-Tf0)*Gama/Tf0]'; xd=[xd(1,:);(xd(2,:)-Tf0)*Gama/Tf0];

 Q=30*[0 0;0 1]; R=.3; S=30*[0 0;0 1];
end;%END OF plant=2

if sil==1,
disp('1=Grafikler için Dur');
dur=input(':');
disp('1=KONTROL DOSYADAN OKUNSUN');
secu=input(':');
if secu==1, load inputu; end;
if ((secy==2) | (secy==3)),
disp('W nın yontemi; W Sıfır=0 ; W Adaptif=1');
wyon=input(':'); end;
elseif sil==0, dur=0; secu=0; wyon=0; end;

if secy==1, tao=.02; end; epsilon=.1; xdf=xd(:,nd);
if sil==1, iterm=10000; elseif sil==0, iterm=0; end;

%ITERATION
for iter=0:iterm,
if sil==1, iter=iter, iteration(iter+1)=iter; elseif sil==0, iteration(iter+1)=iter; end;
if iter==0,
```

```
%VDP
if plant==1, [t,x]=yrk5('yxvp',t0,tf,hp,x0); end;
%CSTR
if plant==2, [t,x]=yrk5('yxcs',t0,tf,hp,x0);
 CA=(1-x(1,:))*CAf; Tr=(1+x(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama; end;
xf=x(:,nd);
J(iter+1)=yinteg(hp,sum(Q*(x-xd).^2/2)+R*u.^2/2)+sum(S*(xf-xdf).^2/2);
if sil==1, JJ=J(iter+1), end;
if (secy==1), tao, save Jsimple J; end; if (secy==2), save Jsecond J; end;
if (secy==3), save Jsecondh J; end;
x1iter(iter+1,:)=x(1,:); x2iter(iter+1,:)=x(2,:); uiter(iter+1,:)=u;
end;%END OF iter==0

if sil==1,
hold on; clf; subplot(2,1,1); plot(t,x,'k'); grid; title('Durumlar'); xlabel('zaman'); ylabel('x1;x2');
subplot(2,1,2); plot(t,u,'k'); grid;  title('Kontrol'); xlabel('zaman'); ylabel('u');
pause2(1); hold off; if iter==0, pause; end; if dur==1, pause; end;

hold on; clf; plot(x(1,:),x(2,:),'k'); grid; title('Faz Diyagramı'); xlabel('x1'); ylabel('x2');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%CSTR: Gerçek değerler
if plant==2,
hold on; clf; subplot(2,2,1); plot(t,CA,'k'); grid; title('Konsantrasyon'); xlabel('zaman'); ylabel('CA');
subplot(2,2,2); plot(t,Tr,'k'); grid; title('Sıcaklık'); xlabel('zaman'); ylabel('T');
subplot(2,2,3); plot(t,Tc,'k'); grid;  title('Soğutucu kontrol sıcaklığı'); xlabel('zaman'); ylabel('Tc');
pause2(1); hold off; if iter==0, pause; end; if dur==1, pause; end;
end;%END OF plant=2
end;

%SIMPLE GRADYAN
if (secy==1),
lmtf=S*(xf-xdf); [t,lm,Ham,Hu,u,Hu2,dJa]=ysmplgr(lmtf,tao,iter); Hu2(iter+1)
%CSTR
if plant==2,
CA=(1-x(1,:))*CAf; Tr=(1+x(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
end;%END OF plant=2

if iter>0,
if plant==1, [t,x]=yrk5('yxvp',t0,tf,hp,x0); end; if plant==2, [t,x]=yrk5('yxcs',t0,tf,hp,x0); end;
xf=x(:,nd); J(iter+1)=yinteg(hp,sum(Q*(x-xd).^2/2)+R*u.^2/2)+sum(S*(xf-xdf).^2/2);
JJ=J(iter+1), if secy==1, save Jsimple J; end;

x1iter(iter+1,:)=x(1,:); x2iter(iter+1,:)=x(2,:); uiter(iter+1,:)=u;
end;

hold on; clf; subplot(2,2,1); plot(t,lm,'k'); grid; title('Adjoint'); xlabel('zaman'); ylabel('lam1;lam2');

subplot(2,2,2); plot(t,Ham(iter+1,:),'k'); grid; title('Hamiltonian'); xlabel('zaman'); ylabel('H');
subplot(2,2,3); plot(t,Hu(iter+1,:),'k'); grid; title('Hu'); xlabel('zaman'); ylabel('Hu');
pause2(0.1); hold off; if dur==1, pause; end;
%
if sqrt(Hu2(iter+1))<5e-2, disp('stop'); dur=1, save inputu u;
hold on; clf; subplot(2,1,1); plot(t,x,'k'); grid; title('Durumlar'); xlabel('zaman'); ylabel('x1;x2');
subplot(2,1,2); plot(t,u,'k'); grid; title('Kontrol'); xlabel('zaman'); ylabel('u');
pause2(1); hold off; if iter==0, pause; end; if dur==1, pause; end;
```

```
%CSTR: Gerçek değerler
if plant==2,
hold on; clf; subplot(2,2,1); plot(t,CA,'k'); grid; title('Konsantrasyon'); xlabel('zaman'); ylabel('CA');
subplot(2,2,2); plot(t,Tr,'k'); grid; title('Sıcaklık'); xlabel('zaman'); ylabel('T');
subplot(2,2,3); plot(t,Tc,'k'); grid; title('Soğutma kontrol sıcaklığı'); xlabel('zaman'); ylabel('Tc');
pause2(1); hold off; if iter==0, pause; end; if dur==1, pause; end;
end;%END OF plant=2

hold on; clf; subplot(2,2,1); plot(t,lm,'k'); grid; title('Adjoint'); xlabel('zaman'); ylabel('lam1;lam2');
subplot(2,2,2); plot(t,Ham(iter+1,:),'k'); grid;  title('Hamiltonian'); xlabel('zaman'); ylabel('H');
subplot(2,2,3); plot(t,Hu(iter+1,:),'k'); grid;  title('Hu'); xlabel('zaman'); ylabel('Hu');
pause2(0.1); hold off; if dur==1, pause; end;

hold on;clf; plot(iteration,J,'k'); grid; title('Ölçüt'); xlabel('iterasyon'); ylabel('J');
pause2(0.1); hold off; if dur==1, pause; end;

hold on; clf; plot(t,uiter,'k'); grid; title('Kontrol İterasyonu'); xlabel('zaman'); ylabel('uiter');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

break; end;
end;%END OF secy=1

%SECOND ORDER COST AZALTMASI
if secy==2,
J2=1e99; if iter==0, if wyon==0, ww=0.0; else ww=1.0e2; end; witer(1)=ww; ww0=ww; end;

ub=u;
while J2>J(iter+1),
xpr0=[S(1,1) S(1,2) S(2,2)]'; xqr0=S*(xf-xdf);
%VDP
if plant==1,
 [t,xpr]=yrk5('ypricvp',t0,tf,hn,xpr0); [t,xqr]=yrk5('yqricvp',t0,tf,hn,xqr0); end;
%CSTR
if plant==2,
 [t,xpr]=yrk5('ypriccs',t0,tf,hn,xpr0); [t,xqr]=yrk5('yqriccs',t0,tf,hn,xqr0); end;

for kk=1:nd, ozP(:,kk)=eig([xpr(1,kk) xpr(2,kk);xpr(2,kk) xpr(3,kk)]);
end; %plot(t,ozP,'k'); grid; pause;

if plant==1, Beta=1.0; end;
K=-(R+ww)\(Beta*[xpr(2,:)' xpr(3,:)']); K11(:,iter+1)=K(:,1); K22(:,iter+1)=K(:,2);
bb(iter+1,:)=-(R+ww)\(Beta*xqr(2,:)+R*u); b=bb(iter+1,:);

xxs=x; tdis=tf; dis=0;
if plant==1, [t,xfb]=yrk5('yxfbvp',t0,tf,hp,x0); end;%VDP
if plant==2, [t,xfb]=yrk5('yxfbcs',t0,tf,hp,x0); end;%CSTR
x=xfb; xf=xfb(:,nd); dx=xfb-xxs; dxf=dx(:,nd); du(iter+1,:)=sum(K'.*dx)+b; duu=du(iter+1,:);
u=uiter(iter+1,:)+du(iter+1,:);

J(iter+2)=yinteg(hp,sum(Q*(x-xd).^2/2)+R*u.^2/2)+sum(S*(xf-xdf).^2/2);
save Jdirekt J; save inputu u;

%%%CSTR
if plant==2, CA=(1-x(1,:))*CAf; Tr=(1+x(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama; end;

Latil1=xpr(1,:).*dx(1,:)+xpr(2,:).*dx(2,:)+xqr(1,:);
Latil2=xpr(2,:).*dx(1,:)+xpr(3,:).*dx(2,:)+xqr(2,:); Latil=[Latil1;Latil2];
```

```
if plant==1, Beta=1.0; end; fLatil=[0 Beta]*Latil;
Hdu(iter+1,:)=R*u+(R+ww)*du(iter+1,:)+fLatil; Hdu2(iter+1)=yinteg(hp,Hdu(iter+1,:).^2);

%ARTIRIMIŞ HAMILTANIAN
Ltil=sum(Q*((x-xd).*dx))+R*u.*du(iter+1,:)+(sum(Q*dx.^2)+R*du(iter+1,:).^2)/2;
Ltilw=sum(Q*((x-xd).*dx))+R*u.*du(iter+1,:)+(sum(Q*dx.^2)+(R+ww)*du(iter+1,:).^2)/2;
dL=sum(Q*((x-xd).*dx))+R*u.*du(iter+1,:); dLqu=(sum(Q*dx.^2)+R*du(iter+1,:).^2)/2;
dLquw=(sum(Q*dx.^2)+(R+ww)*du(iter+1,:).^2)/2;
idL=yinteg(hp,dL)+sum(S'*dxf); idLqu=yinteg(hp,dLqu)+sum(dxf'*S*dxf)/2;
idLquw=yinteg(hp,dLquw)+sum(dxf'*S*dxf)/2; iLtil=yinteg(hp,Ltil); iLtilw=yinteg(hp,Ltilw);
alpha= idLqu/idL;
alphaiter(iter+1)=alpha;

if plant==1,
Htil(iter+1,:)=Ltilw+sum(Latil.*[dx(2,:);-(1+2*ap*x(1,:).*x(2,:)).*dx(1,:)+ap*(1-x(1,:).^2).*dx(2,:)+
du(iter+1,:)]); end;
if plant==2, expx2=exp(x(2,:)./(1+x(2,:)/Gama)); x12g=(1-x(1,:))./(1+x(2,:)/Gama).^2;
 Htil(iter+1,:)=Ltilw+sum(Latil.*[-(1+Da*expx2).*dx(1,:)+Da*(x12g.*expx2).*dx(2,:);-BB*Da
*expx2.*dx(1,:)-(1+Beta-BB*Da*(x12g.*expx2)).*dx(2,:)+Beta*du(iter+1,:)]); end;

J2=J(iter+2);
if J(iter+2)<J(iter+1),
if sil==1, alpha, ww, JJ=J2, end; end;

if sil==1, bkhata=yinteg(hp,(u-uiter(iter+1,:)).^2), elseif sil==0,
bkhata=yinteg(hp,(u-uiter(iter+1,:)).^2); end;

if sil==1,
if (sqrt(Hdu2(iter+1))<5e-3 & (sqrt(bkhata)<5e-3 & ww<5e-2)),
   disp('stop'); break; end; end;

if J(iter+2)>J(iter+1), u=ub; x=xxs;
 ww=5*ww, if wyon==0, if ww==0, ww=1; end; end; elseif wyon==0, ww=0; end;
 witer(iter+1)=ww;

end;%END OF WHILE

if sil==1,
if (sqrt(Hdu2(iter+1))<5e-3 & (sqrt(bkhata)<5e-3 & ww<5e-2)), dur=1
 save inputu u; save dinputu duu; save wcont ww; save states x; save dstates dx; save Kgain K;
 save bgain b; save xpr; save xqr; disp('stop');
 x1iter(iter+2,:)=x(1,:); x2iter(iter+2,:)=x(2,:); uiter(iter+2,:)=u; iteration(iter+2)=iter+1;

if wyon==1, witer(iter+2)=abs(alpha)*ww; end;

%ÇIKIŢ GRAFÝKLER
ygraf;

hold on; clf;
%subplot(2,1,1); plot(t,x,'k');
subplot(2,1,1); plot(t,x(1,:),':k',t,xd(1,:),'k');
%subplot(2,1,1); plot(t,x(2,:),':k',t,xd(2,:),'k');
title('Durumlar'); xlabel('zaman'); ylabel('x1;x2');
%ylabel('x1');
%ylabel('x2');

subplot(2,1,2); plot(t,u,'k'); title('Kontrol'); xlabel('zaman'); ylabel('u');
```

```
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

hold on; clf; plot(x(1,:),x(2,:),'k'); grid; title('Faz Diyagramı'); xlabel('x1'); ylabel('x2');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%%%CSTR: Gerçek değerler
if plant==2,
hold on; clf; subplot(2,2,1); plot(t,CA,'k'); grid; title('Konsantrasyon'); xlabel('zaman'); ylabel('CA');
subplot(2,2,2); plot(t,Tr,'k'); grid; title('Sıcaklık'); xlabel('zaman'); ylabel('T');
subplot(2,2,3); plot(t,Tc,'k'); grid; title('Soğutma kontrol sıcaklığı'); xlabel('zaman'); ylabel('Tc');
pause2(1); hold off; if iter==0, pause; end; if dur==1, pause; end; end;%END OF plant=2

%İterasyonla Değişimler
ycsodeg;

%GERİ BESLEME DENEMELERİ
if plant==1, ygerdnvp; end; if plant==2, ygerdncs; end;
break; end;

end;

if wyon==1, ww=abs(alpha)*ww; end;
witer(iter+2)=ww;

save Jsecond J; x1iter(iter+2,:)=x(1,:); x2iter(iter+2,:)=x(2,:); uiter(iter+2,:)=u; u2=u; save u2 u2;

if sil==1,
%GRAFİKLER
ygraf; end;

end;%END OF secy


if sil==1,
%SECOND ORDER HAMILTANIAN
if secy==3,
J2=1e99; if iter==0, if wyon==0, ww=0.0; epsilon=.1; else ww=5.0e2; epsilon=1; end;
  witer(1)=ww; ww0=ww; end;

ub=u;
while J2>J(iter+1),
%ADJOINT
lmtf=S*(xf-xdf); xpr0=[S(1,1) S(1,2) S(2,2)]'; xqr0=[0 0]';
if plant==1,
 [t,lm]=yrk5('yxlmvp',t0,tf,hn,lmtf);  [t,xpr]=yrk5('ypricvph',t0,tf,hn,xpr0);
 [t,xqr]=yrk5('yqricvph',t0,tf,hn,xqr0);
%A ve C Matrisi
A=[zeros(1,nd);ones(1,nd);-(1+2*x(1,:).*x(2,:));1-x(1,:).^2];
C=[-2*x(2,:).*lm(2,:)+Q(1,1);-2*x(1,:).*lm(2,:);ones(1,nd)*Q(2,2)];
end;%END OF plant=1

if plant==2,
 [t,lm]=yrk5('yxlmcs',t0,tf,hn,lmtf); [t,xpr]=yrk5('ypriccsh',t0,tf,hn,xpr0);
 [t,xqr]=yrk5('yqriccsh',t0,tf,hn,xqr0);
%A ve C Matrisi
expx2=exp(x(2,:)./(1+x(2,:)/Gama)); x12g=(1-x(1,:))./(1+x(2,:)/Gama).^2;
A=[-(1+Da*expx2);Da*(x12g.*expx2);-BB*Da*expx2;-(1+Beta-BB*Da*(x12g.*expx2))];
```

```
C=[ones(1,nd)*Q(1,1);-
Da*(lm(1,:)+BB*lm(2,:)).*expx2./(1+x(2,:)/Gama).^2;Da*(lm(1,:)+BB*(lm(2,:)).*(1-x(1,:))).*(1-
2*(1+x(2,:)/Gama)/Gama).*expx2./(1+x(2,:)/Gama).^4+Q(2,2)];
end;%END OF plant=2

for kk=1:nd,
 ozA(:,kk)=eig([A(1,kk) A(2,kk);A(3,kk) A(4,kk)]);
 ozC(:,kk)=eig([C(1,kk) C(2,kk);C(2,kk) C(3,kk)]);
 ozP(:,kk)=eig([xpr(1,kk) xpr(2,kk);xpr(2,kk) xpr(3,kk)]);
end;  %clf; plot(t,real(ozA),'k'); grid; pause;
  clf; plot(t,ozC,'k'); grid; xlabel('zaman'); ylabel('ozC'); title('C`nin öz değer değişimi'); pause;
  clf; plot(t,ozP,'k'); grid; pause;

if iter>2, epsilon=1; end;
if plant==1, Beta=1; end;
K=-(R+ww)\(Beta*[xpr(2,:)' xpr(3,:)']); K11(:,iter+1)=K(:,1); K22(:,iter+1)=K(:,2);
bb(iter+1,:)=epsilon*(-(R+ww)\(Beta*(xqr(2,:)+lm(2,:))+R*u)); b=bb(iter+1,:);

xxs=x;  tdis=tf; dis=0;
if plant==1, [t,xfb]=yrk5('yxfbvp',t0,tf,hp,x0); end;
if plant==2, [t,xfb]=yrk5('yxfbcs',t0,tf,hp,x0); end;
x=xfb; xf=xfb(:,nd);
dx=xfb-xxs; dxf=dx(:,nd); du(iter+1,:)=sum(K'.*dx)+b; duu=du(iter+1,:);
u=uiter(iter+1,:)+du(iter+1,:);

J(iter+2)=yinteg(hp,sum(Q*(x-xd).^2/2)+R*u.^2/2)+sum(S*(xf-xdf).^2/2);
save Jdirekt J; save inputu u;

%CSTR
if plant==2,
CA=(1-x(1,:))*CAf; Tr=(1+x(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama; end;

dLatil1=xpr(1,:).*dx(1,:)+xpr(2,:).*dx(2,:)+xqr(1,:);
dLatil2=xpr(2,:).*dx(1,:)+xpr(3,:).*dx(2,:)+xqr(2,:);
dLatil=[dLatil1;dLatil2];
if plant==1, Beta=1.0; end; fLatil=[0 Beta]*dLatil;
Hdu(iter+1,:)=R*u+Beta*lm(2,:)+(R+ww)*du(iter+1,:)+fLatil;
Hdu2(iter+1)=yinteg(hp,Hdu(iter+1,:).^2);

%HAMILTONIAN
if plant==1,
Ham(iter+1,:)=(sum(Q*(x-xd).^2)+R*u.^2)/2+lm(1,:).*x(2,:)+lm(2,:).*(-x(1,:)+ap*(1-x(1,:).^2).*
x(2,:)+u);  Hu(iter+1,:)=R*u+lm(2,:); end;
if plant==2,
 fdot=yxcs(-9999,x);
 Ham(iter+1,:)=(sum(Q*(x-xd).^2)+R*u.^2)/2+sum(lm.*fdot);  Hu(iter+1,:)=R*u+Beta*lm(2,:);
end;

%ARTIRILMIŞ HAMILTONIAN
if plant==1, delJ=J(iter+1)-J(iter+2); bagJ=delJ/J(iter+1);
 Ltil=(R*u+lm(2,:)).*du(iter+1,:)+(sum(Q*dx.^2)-2*ap*x(2,:).*lm(2,:).*dx(1,:).^2-4*ap*x(1,:).*
lm(2,:).*dx(1,:).*dx(2,:)+R*du(iter+1,:).^2)/2;
 Ltilw=(R*u+lm(2,:)).*du(iter+1,:)+(sum(Q*dx.^2)-2*ap*x(2,:).*lm(2,:).*dx(1,:).^2-4*ap*x(1,:).*
lm(2,:).*dx(1,:).*dx(2,:)+(R+ww)*du(iter+1,:).^2)/2;
dL=(R*u+lm(2,:)).*du(iter+1,:);
dLqu=(sum(Q*dx.^2)-2*ap*x(2,:).*lm(2,:).*dx(1,:).^2-4*ap*x(1,:).*lm(2,:).*dx(1,:).*dx(2,:)+R*
du(iter+1,:).^2)/2;
```

```
dLquw=(sum(Q*dx.^2)-2*ap*x(2,:).*lm(2,:).*dx(1,:).^2-4*ap*x(1,:).*lm(2,:).*dx(1,:).*dx(2,:)+
(R+ww)*du(iter+1,:).^2)/2;
end;
if plant==2, delJ=J(iter+1)-J(iter+2); bagJ=delJ/J(iter+1);
 expx2=exp(x(2,:)./(1+x(2,:)/Gama));
 Ltil=(R*u+Beta*lm(2,:)).*du(iter+1,:)+(sum(Q*dx.^2)-
2*Da*(lm(1,:)+BB*lm(2,:)).*expx2.*(dx(1,:).*dx(2,:))./(1+x(2,:)/Gama).^2+Da*(lm(1,:)+BB*lm(2,:
)).*(1-x(1,:)).*(1-
2*(1+x(2,:)/Gama)/Gama).*expx2.*dx(2,:).^2./(1+x(2,:)/Gama).^4+R*du(iter+1,:).^2)/2;
 Ltilw=(R*u+Beta*lm(2,:)).*du(iter+1,:)+(sum(Q*dx.^2)-2*Da*(lm(1,:)+BB*lm(2,:)).*expx2.*
(dx(1,:).*dx(2,:))./(1+x(2,:)/Gama).^2+Da*(lm(1,:)+BB*lm(2,:)).*(1-x(1,:)).*(1-2*(1+x(2,:)/Gama)
/Gama).*expx2.*dx(2,:).^2./(1+x(2,:)/Gama).^4+(R+ww)*du(iter+1,:).^2)/2;
 dL=(R*u+Beta*lm(2,:)).*du(iter+1,:);
 dLqu=(sum(Q*dx.^2)-2*Da*(lm(1,:)+BB*lm(2,:)).*expx2.*(dx(1,:).*dx(2,:))./(1+x(2,:)/Gama).^2+
Da*(lm(1,:)+BB*lm(2,:)).*(1-x(1,:)).*(1-2*(1+x(2,:)/Gama)/Gama).*expx2.*dx(2,:).^2./    (1+x(2,:)/
Gama).^4+R*du(iter+1,:).^2)/2;
 dLquw=(sum(Q*dx.^2)-2*Da*(lm(1,:)+BB*lm(2,:)).*expx2.*(dx(1,:).*dx(2,:))./(1+x(2,:)/Gama).^2
+Da*(lm(1,:)+BB*lm(2,:)).*(1-x(1,:)).*(1-2*(1+x(2,:)/Gama)/Gama).*expx2.*dx(2,:).^2./
(1+x(2,:)/Gama).^4+(R+ww)*du(iter+1,:).^2)/2;
 end;
 idL=yinteg(hp,dL);idLqu=yinteg(hp,dLqu)+sum(dxf'*S*dxf)/2;
 idLquw=yinteg(hp,dLquw)+sum(dxf'*S*dxf)/2; iLtil=yinteg(hp,Ltil); iLtilw=yinteg(hp,Ltilw);
 alpha=bagJ*idL/idLqu
 alphaiter(iter+1)=alpha;

 if plant==1,
 Htil(iter+1,:)=Ltilw+sum(dLatil.*[dx(2,:);-(1+2*ap*x(1,:).*x(2,:)).*dx(1,:)+ap*(1-x(1,:).^2).*
 dx(2,:)+du(iter+1,:)]); end;
 if plant==2,
 x12g=(1-x(1,:))./(1+x(2,:)/Gama).^2;
 Htil(iter+1,:)=Ltilw+sum(dLatil.*[-(1+Da*expx2).*dx(1,:)+Da*(x12g.*expx2).*dx(2,:);...
   -BB*Da*expx2.*dx(1,:)-(1+Beta-BB*Da*(x12g.*expx2)).*dx(2,:)+Beta*du(iter+1,:)]); end;

 J(iter+2)=yinteg(hp,sum(Q*(x-xd).^2/2)+R*u.^2/2)+sum(S*(xf-xdf).^2/2); J2=J(iter+2);
 if J(iter+2)<J(iter+1), alpha
 ww, JJ=J2,
 end;

 bkhata=yinteg(hp,(u-uiter(iter+1,:)).^2)
 if (sqrt(Hdu2(iter+1))<5e-2 & (sqrt(bkhata)<5e-2 & ww<5e-2)),
   disp('stop'); break; end;

 if J(iter+2)>J(iter+1), u=ub; x=xxs;
 ww=5*ww; if wyon==0, if ww==0, ww=1; end; end; elseif wyon==0, ww=0; end; witer(iter+1)=ww;

 end;%END OF WHILE

 if (sqrt(Hdu2(iter+1))<5e-2 & (sqrt(bkhata)<5e-2 & ww<5e-2)), dur=1
  save inputu u; save dinputu duu; save wcont ww; save states x; save dstates dx; save Kgain K;
  save bgain b; save xpr; save xqr; disp('stop');
 x1iter(iter+2,:)=x(1,:); x2iter(iter+2,:)=x(2,:); uiter(iter+2,:)=u; iteration(iter+2)=iter+1;
 if wyon==1, witer(iter+2)=abs(alpha)*ww; end;

 %%%ÇIKIŢ GRAFÝKLER
 ygrafh;

 hold on; clf; subplot(2,1,1); plot(t,x,'k'); grid; title('Durumlar'); xlabel('zaman'); ylabel('x1;x2');
```

```
subplot(2,1,2); plot(t,u,'k'); grid; title('Kontrol'); xlabel('zaman'); ylabel('u');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

hold on; clf; plot(x(1,:),x(2,:),'k'); grid; title('Faz Diyagramı'); xlabel('x1'); ylabel('x2');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%CSTR: Gerçek değerler
if plant==2,
hold on; clf; subplot(2,2,1); plot(t,CA,'k'); grid; title('Konsantrasyon'); xlabel('zaman'); ylabel('CA');
subplot(2,2,2); plot(t,Tr,'k'); grid; title('Sıcaklık'); xlabel('zaman'); ylabel('T');
subplot(2,2,3); plot(t,Tc,'k'); grid; title('Soğutma kontrol sıcaklığı'); xlabel('zaman'); ylabel('Tc');
pause2(1); hold off; if iter==0, pause; end; if dur==1, pause; end;
end;%END OF plant=2

%İterasyonla Değişimler
ycsodeg;

%GERİ BESLEME DENEMELERİ
if plant==1, ygerdnvp; end;
if plant==2, ygerdncs; end;
break; end;

if wyon==1, ww=abs(alpha)*ww; end;
witer(iter+2)=ww;

save Jsecond J; x1iter(iter+2,:)=x(1,:); x2iter(iter+2,:)=x(2,:); uiter(iter+2,:)=u;

%GRAFİKLER
ygrafh;

end;%END OF secy
end;

end;%END OF ITERATION

.....
%    ***BASİT GRADYAN YÖNTEMİ İÇİN ADJOİNT, HAMİLTANIAN ***
%          *** VE KONTROL FONKSİYONU HESABI ***
function [t,lm,Ham,Hu,u,Hu2,dJa]=ysmplgr(lmtf,tao,iter)

% Basit gradyan algoritması için gerekli hesaplamalar yapılıyor
% Fonsiyona girilen değişkenler:
% lmtf: Adjoint diff. denklem için son değer vektörü
% tao: iterasyon için adım büyüklüğü
% iter: iterasyon değeri
% Fonsiyondan dönen değişkenler:
% t: Sistemin zaman değerleri
% lm: Adjoint durum değişkeni vektör fonksiyonu
% Ham: Gerekli koţullarýn uygulandığı Hamiltonian
% Hu: Ham'ın kontrol fonksiyonuna göre gradyanı
% u: k. iterasyon sonunda elde edilen kontrol fonksiyonu
% Hu2: Çıkış kriteri olarak kullanılan Hu' nun modülü (||Hu||)
% dJa: k. iterasyon sonunda elde edilen artırımsal ölçüt

% Yaşar BECERİKLİ- 21:29 17.03.1998
```

```
global t0; global tf; global nd; global u; global x; global Q; global R; global xd; global lm; global
plant; global ap;

global Gama; global Beta; global Da; global BB; global dd; global x2c;
hp=(tf-t0)/(nd-1); hn=-hp;

%ADJOINT OF FIRST ORDER

%VDP
if plant==1,
[t,lm]=yrk5('yxlmvp',t0,tf,hn,lmtf);
%1st ORDER GRADYAN (SIMPLE GRADYAN)
Hx=[x(1,:)-(1+2*ap*x(1,:).*x(2,:)).*lm(2,:);x(2,:)-ap*(x(1,:).^2-1).*lm(2,:)+lm(1,:)];
Hu(iter+1,:)=R*u+lm(2,:); u=u-tao*Hu(iter+1,:); Hu2(iter+1)=sum(yinteg(hp,Hu(iter+1,:).^2));
dJa(iter+1)=-tao*Hu2(iter+1); fdot=yxvp(-9999,x);
Ham(iter+1,:)=(sum(Q*(x-xd).^2)+R*u.^2)/2+sum(lm.*fdot);
end;%END OF plant=1

%CSTR
if plant==2,
[t,lm]=yrk5('yxlmcs',t0,tf,hn,lmtf);
%1st ORDER GRADYAN (SIMPLE GRADYAN)
expx2=exp(x(2,:)./(1+x(2,:)/Gama)); x12g=(1-x(1,:))./(1+x(2,:)/Gama).^2;
Hx=[-(1+Da*expx2).*lm(1,:)-BB*Da*(expx2.*lm(2,:))+Q(1,1)*(x(1,:)-xd(1,:));...
  Da*(x12g.*expx2).*lm(1,:)-(1+Beta-BB*Da*(x12g.*expx2)).*lm(2,:)+Q(2,2)*(x(2,:)-xd(2,:))];
Hu(iter+1,:)=R*u+Beta*lm(2,:); u=u-tao*Hu(iter+1,:); Hu2(iter+1)=sum(yinteg(hp,Hu(iter+1,:).^2));
dJa(iter+1)=-tao*Hu2(iter+1);
fdot=yxcs(-9999,x); Ham(iter+1,:)=(sum(Q*(x-xd).^2)+R*u.^2)/2+sum(lm.*fdot);
end;%END OF plant=2

.....
% *** VAN-DER POL ADJOINT DİFF. DENKLEMİ ***
function yx=yxlmvp(t,lam)

% Adjoint Diferansiyel Denklemleri (Van-Der Pol Denklemlerinin)
% t: zaman ; lm: Adjoint Durum Vektör Fonksiyonu
% yx: Dönen Diferansiyel Denklem Vektörü

% Yaşar BECERİKLİ-16:11 06.07.1997

global x; global t0; global tf; global nd; global u; global Q; global R; global xd; global ap;

if t~=-9999,
 it=round(t*(nd-1)/(tf-t0)+1);
 yx1=(1+2*ap*x(1,it).*x(2,it)).*lam(2)-Q(1,1)*(x(1,it)-xd(1,it));
 yx2=-lam(1)-ap*(x(1,it).^2-1).*lam(2)-Q(2,2)*(x(2,it)-xd(2,it));
 yx=[yx1;yx2];
else
 yx1=(1+2*ap*x(1,:).*x(2,:)).*lam(2,:)-Q(1,1)*(x(1,:)-xd(1,:));
 yx2=-lam(1,:)-ap*(x(1,:).^2-1).*lam(2,:)-Q(2,2)*(x(2,:)-xd(2,:));
 yx=[yx1;yx2]; end;

.....
% *** CSTR ADJOINT DİFF. DENKLEMİ ***
function yx=yxlmcs(t,lam)

% Adjoint Diferansiyel Denklemleri (CSTR Denklemlerinin)
```

```
% t: zaman ; lam: Adjoint Durum Vektör Fonksiyonu
% yx: Dönen Diferansiyel Denklem Vektörü

% Yaşar BECERİKLİ-00:19 07.03.1998

global x; global t0; global tf; global nd; global u; global Q; global R; global xd;

global Gama; global Beta; global Da; global BB; global dd; global x2c;

if t~=-9999,
 it=round(t*(nd-1)/(tf-t0)+1);
 expx2=exp(x(2,it)/(1+x(2,it)/Gama)); x12g=(1-x(1,it))/(1+x(2,it)/Gama).^2;
 yx1=(1+Da*expx2)*lam(1)-BB*Da*expx2*lam(2)-Q(1,1)*(x(1,it)-xd(1,it));
 yx2=-Da*(x12g*expx2)*lam(1)+(1+Beta-BB*Da*(x12g*expx2))*lam(2)-Q(2,2)*(x(2,it)-xd(2,it));
 yx=[yx1;yx2];
else
 expx2=exp(x(2,:)./(1+x(2,:)/Gama)); x12g=(1-x(1,:))./(1+x(2,:)/Gama).^2;
 yx1=(1+Da*expx2).*lam(1,:)-BB*Da*(expx2.*lam(2,:))-Q(1,1)*(x(1,:)-xd(1,:));
 yx2=-Da*(x12g.*expx2).*lam(1,:)+(1+Beta-BB*Da*(x12g.*expx2)).*lam(2,:)-Q(2,2)*(x(2,:)-
xd(2,:));
 yx=[yx1;yx2]; end;

.....
% *** VAN-DER POL DİFF. DENKLEMLERİ ***
function xdot=yxvp(t,xx)

% Van-Der Pol Diferansiyel Denklemleri
% t: zaman ; x: durum vektörü ;
% xdot: Dönen Diferansiyel Denklem Vektörü

% Yaşar BECERİKLİ- 16:15 06.07.1997

global t0; global tf; global nd; global u; global ap;

if t~=-9999,
 it=round(t*(nd-1)/(tf-t0)+1);
 xdot1=xx(2); xdot2=-xx(1)+ap*(1-xx(1).^2)*xx(2)+u(it); xdot=[xdot1;xdot2];
else
  xdot1=xx(2,:); xdot2=-xx(1,:)+ap*(1-xx(1,:).^2).*xx(2,:)+u;
xdot=[xdot1;xdot2]; end;

.....
% *** CSTR DİFF. DENKLEMLERİ ***
function xdot=yxcs(t,x)

% CSTR Diferansiyel Denklemleri
% t: zaman ; x: durum vektörü ;
% xdot: Dönen Diferansiyel Denklem Vektörü

% Yaşar BECERİKLİ- 16:29 01.03.1998

global t0; global tf; global nd; global u;
global Gama; global Beta; global Da; global BB; global dd; global x2c;

if t~=-9999,
 it=round(t*(nd-1)/(tf-t0)+1);
 expx2=exp(x(2)/(1+x(2)/Gama));
```

```
 xdot1=-x(1)+Da*(1-x(1))*expx2;
 xdot2=-x(2)+BB*Da*(1-x(1))*expx2-Beta*(x(2)-x2c)+dd(it)+Beta*u(it);
 xdot=[xdot1;xdot2];
else expx2=exp(x(2,:)./(1+x(2,:)/Gama));
 xdot1=-x(1,:)+Da*(1-x(1,:)).*expx2;
 xdot2=-x(2,:)+BB*Da*(1-x(1,:)).*expx2-Beta*(x(2,:)-x2c)+dd+Beta*u;
xdot=[xdot1;xdot2]; end;
```

.....

```
% *** GERİ BESLEME YAPISINDAKİ VAN-DER.POL DİFF. DENK. ***
function xdot=yxfbvp(t,xx)

% Van-Der Pol Geri Besleme Diferansiyel Denklemleri
% t: zaman ; xx: durum vektörü ; xdot: Dönen Diferansiyel Denklem Vektörü

% Yaşar BECERİKLİ- 17:12 19.09.1997

global t0; global tf; global nd; global u; global K; global b; global xxs;
global dis; global tdis; global ap;

if t~=-9999,
it=round(t*(nd-1)/(tf-t0)+1);
if t>=tdis, d=dis; else d=0; end;
 xdot1=xx(2); xdot2=-xx(1)+ap*(1-xx(1).^2)*xx(2)+u(it)-K(it,:)*(xx-xxs(:,it))+b(:,it)+d;
 xdot=[xdot1;xdot2];
else
 xdot1=xx(2,:); xdot2=-xx(1,:)+ap*(1-xx(1,:).^2).*xx(2,:)+u-sum(K'.*(xx-xxs))+b;
 xdot=[xdot1;xdot2]; end;
```

.....

```
% *** GERİ BESLEME YAPISINDAKİ CSTR DİFF. DENK. ***
function xdot=yxfbcs(t,xx)

% CSTR Geri Besleme Diferansiyel Denklemleri
% t: zaman ; xx: durum vektörü ; xdot: Dönen Diferansiyel
% Denklem Vektörü

% Yaşar BECERİKLİ- 00:11 10.03.1998

global t0; global tf; global nd; global u; global K; global b; global xxs; global dis; global tdis;
global Gama; global Beta; global Da; global BB; global dd; global x2c;

if t~=-9999,
it=round(t*(nd-1)/(tf-t0)+1);
if t>=tdis, d=dis; else d=0; end;
 expx2=exp(xx(2)/(1+xx(2)/Gama));
 xdot1=-xx(1)+Da*(1-xx(1))*expx2;
 xdot2=-xx(2)+BB*Da*(1-xx(1))*expx2-Beta*(xx(2)-x2c)+dd(it)+d+Beta*(u(it)-K(it,:)*(xx-
xxs(:,it))+b(:,it));
 xdot=[xdot1;xdot2];
else
 expx2=exp(xx(2,:)./(1+xx(2,:)/Gama));
 xdot1=-xx(1,:)+Da*(1-xx(1,:)).*expx2;
 xdot2=-xx(2,:)+BB*Da*(1-xx(1,:)).*expx2-Beta*(xx(2,:)-x2c)+dd+Beta*(u-sum(K'.*(xx-xxs))+b);
 xdot=[xdot1;xdot2]; end;
```

.....

```
%  *** VAN-DER POL SİSTEMİNE AİT DİREK ÖLÇÜT AZALTILMASI ***
%    *** YÖNTEMİNDEKİ RİKKATİ MATRİS DİFF. DENKLEMLERİ ***
function pp=ypricvp(t,p)

% Riccati Matris DifF. Denklemeleri (Van-Der Pol Denklemlerinin)
% Lamtil=Pdx+q Dönüşüm Formundaki P Matrisi
% t: zaman ; p: Riccati Matrisinin (Simetrik Matris) Durum Vektörü
% pp: Dönen Diferansiyel Denklem Vektörü
% NOT: Riccati Matrisi Simetrik Olduğundan, Elemanları Vektörel Hale
% Getirilip Çözüldükten Sonra Tekrar Simetrik Matrise Dönüştürülür

%  Yaşar BECERİKLİ- 17:03 06.07.1997

global t0; global tf; global nd; global x; global b; global Q; global R; global ww; global ap;

hp=(tf-t0)/(nd-1); it=round(t*(nd-1)/(tf-t0)+1);

xp11=2*(1+2*ap*x(1,it)*x(2,it))*p(2)+p(2)^2/(R+ww)-Q(1,1);
xp12=-p(1)+ap*(x(1,it)^2-1)*p(2)+(1+2*ap*x(1,it)*x(2,it))*p(3)+p(2)*p(3)/(R+ww);
xp22=-2*p(2)+2*ap*(x(1,it)^2-1)*p(3)+p(3)^2/(R+ww)-Q(2,2);
pp=[xp11;xp12;xp22];

.....
%  *** CSTR SİSTEMİNE AİT DİREK ÖLÇÜT AZALTILMASI ***
%    *** YÖNTEMİNDEKİ RİKKATİ MATRİS DİFF. DENKLEMLERİ ***
function pp=ypriccs(t,p)

% Riccati Matris DifF. Denklemeleri (CSTR Denklemlerinin)
% Lamtil=Pdx+q Dönüşüm Formundaki P Matrisi
% t: zaman ; p: Riccati Matrisinin (Simetrik Matris) Durum Vektörü
% pp: Dönen Diferansiyel Denklem Vektörü
% NOT: Riccati Matrisi Simetrik Olduğundan, Elemanları Vektörel Hale
% Getirilip Çözüldükten Sonra Tekrar Simetrik Matrise Dönüştürülür

%  Yaşar BECERİKLİ- 12:16 07.03.1998

global t0; global tf; global nd; global x; global b; global Q; global R; global ww;
global Gama; global Beta; global Da; global BB; global dd; global x2c;

hp=(tf-t0)/(nd-1); it=round(t*(nd-1)/(tf-t0)+1);

expx2=exp(x(2,it)/(1+x(2,it)/Gama)); x12g=(1-x(1,it))/(1+x(2,it)/Gama)^2;
xp11=2*(1+Da*expx2)*p(1)+2*BB*Da*expx2*p(2)+(Beta*p(2))^2/(R+ww)-Q(1,1);
xp12=-Da*x12g*expx2*p(1)+(2+Beta+Da*(1-BB*x12g)*expx2)*p(2)+ BB*Da* expx2* p(3)+
Beta^2*p(2)*p(3)/(R+ww);
xp22=-2*Da*x12g*expx2*p(2)+2*(1+Beta-BB*Da*x12g*expx2)*p(3)+(Beta*p(3))^2/(R+ww)-
Q(2,2);
pp=[xp11;xp12;xp22];

.....
%  *** VAN-DER POL SİSTEMİNE AİT DİREK ÖLÇÜT AZALTILMASI ***
%    *** YÖNTEMİNDEKİ VEKTÖR DİFF. DENKLEMLERİ ***
function qq=yqricvp(t,q)

% Riccati Vektör DifF. Denklemeleri (Van-Der Pol Denklemlerinin)
% Lamtil=Pdx+q Dönüşüm Formundaki q Vektörü
% t: zaman ; q: Riccati Vektörünün Durum Uzayı Vektörü
```

```
% qq: Dönen Diferansiyel Denklem Vektörü

% Yaşar BECERİKLİ- 17:10 06.07.1997

global t0; global tf; global nd; global x; global b; global u; global xpr;
global Q; global R; global xd; global ww; global ap;

hp=(tf-t0)/(nd-1); it=round(t*(nd-1)/(tf-t0)+1);

xq1=((1+2*ap*x(1,it)*x(2,it))+xpr(2,it)/(R+ww))*q(2)+xpr(2,it)*R*u(it)/(R+ww)-Q(1,1)*(x(1,it)-
xd(1,it));
xq2=-q(1)+(ap*(x(1,it)^2-1)+xpr(3,it)/(R+ww))*q(2)+xpr(3,it)*R*u(it)/(R+ww)-Q(2,2)*(x(2,it)-
xd(2,it));
qq=[xq1;xq2];

.....
%  *** CSTR SİSTEMİNE AİT DİREK ÖLÇÜT AZALTILMASI ***
%    *** YÖNTEMİNDEKİ VEKTÖR DİFF. DENKLEMLERİ ***
function qq=yqriccs(t,q)

% Riccati Vektör DifF. Denklemeleri (CSTR Denklemlerinin)
% Lamtil=Pdx+q Dönüşüm Formundaki q Vektörü
% t: zaman ; q: Riccati Vektörünün Durum Uzayı Vektörü
% qq: Dönen Diferansiyel Denklem Vektörü

% Yaşar BECERİKLİ- 13:33 07.03.1998

global t0; global tf; global nd; global x; global b; global u; global xpr;
global Q; global R; global xd; global ww;
global Gama; global Beta; global Da; global BB; global dd; global x2c;

hp=(tf-t0)/(nd-1); it=round(t*(nd-1)/(tf-t0)+1);

expx2=exp(x(2,it)/(1+x(2,it)/Gama)); x12g=(1-x(1,it))/(1+x(2,it)/Gama)^2;
xq1=(1+Da*expx2)*q(1)+(BB*Da*expx2+Beta^2*xpr(2,it)/(R+ww))*q(2)+xpr(2,it)*Beta*R*u(it)/(
R+ww)-Q(1,1)*(x(1,it)-xd(1,it));
xq2=-Da*x12g*expx2*q(1)+(1+Beta-BB*Da*x12g*expx2+Beta^2*xpr(3,it)/(R+ww))*q(2)+
xpr(3,it)*Beta*R*u(it)/(R+ww)-Q(2,2)*(x(2,it)-xd(2,it));
qq=[xq1;xq2];

.....
%  *** VAN-DER POL SİSTEMİNE AİT 2. DERECE HAMILTONIAN ***
%    *** YÖNTEMİNDEKİ RİKKATİ MATRİS DİFF. DENKLEMLERİ ***
function pp=ypricvph(t,p)

% 2.Derece Hamiltonian Çözümüyle Riccati Matris
% Diferansiyel Denklemeleri (Van der Pol Denklemlerinin)
% Lamtil=Pdx+q Dönüşüm Formundaki P Matrisi
% t: zaman ; p: Riccati Matrisinin (Simetrik Matris) Durum
% Uzayı Vektörü
% pp: Dönen Diferansiyel Denklem Vektörü
% NOT: Riccati Matrisi Simetrik Olduğundan, Elemanları Vektörel
% Hale Getirilip Çözüldükten Sonra Tekrar Simetrik Matrise Dönüştürülür

% Yaşar BECERİKLİ- 18:54 13.07.1997

global t0; global tf; global nd; global x; global b; global Q; global R; global ww; global lm; global ap;
```

```
hp=(tf-t0)/(nd-1); it=round(t*(nd-1)/(tf-t0)+1);

xp11=2*(1+2*ap*x(1,it)*x(2,it))*p(2)+p(2)^2/(R+ww)-(Q(1,1)-2*ap*x(2,it)*lm(2,it));
xp12=-p(1)+ap*(x(1,it)^2-
1)*p(2)+(1+2*ap*x(1,it)*x(2,it))*p(3)+p(2)*p(3)/(R+ww)+2*ap*x(1,it)*lm(2,it);
xp22=-2*p(2)+2*ap*(x(1,it)^2-1)*p(3)+p(3)^2/(R+ww)-Q(2,2);
pp=[xp11;xp12;xp22];

.....
%  *** CSTR SİSTEMİNE AİT 2. DERECE HAMILTONIAN ***
%    *** YÖNTEMİNDEKİ RİKKATİ MATRİS DİFF. DENKLEMLERİ ***
function pp=ypriccsh(t,p)

% 2.Derece Hamiltonian Çözümüyle Riccati Matris
% Diferansiyel Denklemeleri (CSTR Denklemlerinin)
% Lamtil=Pdx+q Dönüşüm Formundaki P Matrisi
% t: zaman ; p: Riccati Matrisinin (Simetrik Matris) Durum
% Uzayı Vektörü
% pp: Dönen Diferansiyel Denklem Vektörü
% NOT: Riccati Matrisi Simetrik Olduğundan, Elemanları Vektörel
% Hale Getirilip Çözüldükten Sonra Tekrar Simetrik Matrise Dönüştürülür

% Yaşar BECERİKLİ- 12:13 09.03.1998

global t0; global tf; global nd; global x; global b; global Q; global R; global ww; global lm;
global Gama; global Beta; global Da; global BB; global dd; global x2c;

hp=(tf-t0)/(nd-1); it=round(t*(nd-1)/(tf-t0)+1);

expx2=exp(x(2,it)/(1+x(2,it)/Gama)); x12g=(1-x(1,it))/(1+x(2,it)/Gama)^2;
xp11=2*(1+Da*expx2)*p(1)+2*BB*Da*expx2*p(2)+(Beta*p(2))^2/(R+ww)-Q(1,1);
xp12=-Da*x12g*expx2*p(1)+(2+Beta+Da*(1-BB*x12g)*expx2)*p(2)+BB*Da*expx2*p(3)+
Beta^2*p(2)*p(3)/(R+ww)+Da*(lm(1,it)+BB*lm(2,it))*expx2/(1+x(2,it)/Gama)^2;
xp22=-2*Da*x12g*expx2*p(2)+2*(1+Beta-BB*Da*x12g*expx2)*p(3)+(Beta*p(3))^2/(R+ww)-
Da*(lm(1,it)+BB*lm(2,it))*(1-x(1,it))*(1-2*(1+x(2,it)/Gama)/Gama)*expx2/(1+x(2,it)/Gama)^4-
Q(2,2);
pp=[xp11;xp12;xp22];

.....
%  *** VAN-DER POL SİSTEMİNE AİT 2. DERECE HAMILTONIAN ***
%    *** YÖNTEMİNDEKİ VEKTÖR DİFF. DENKLEMLERİ ***
function qq=yqricvph(t,q)

% 2.Derece Hamiltonian Çözümüyle Riccati Vektör
% Diferansiyel Denklemleri (Van der Pol Denklemlerinin)
% Lamtil=Pdx+q Dönüşüm Formundaki q Vektörü
% t: zaman ; q: Riccati Vektörünün Durum Uzayı Vektörü
% qq: Dönen Diferansiyel Denklem Vektörü

% Yaşar BECERİKLİ- 19:00 13.07.1997

global t0; global tf; global nd; global x; global b; global u; global xpr;
global Q; global R; global xd; global ww; global lm; global ap;

hp=(tf-t0)/(nd-1); it=round(t*(nd-1)/(tf-t0)+1);

xq1=((1+2*ap*x(1,it)*x(2,it))+xpr(2,it)/(R+ww))*q(2)+xpr(2,it)*(R*u(it)+lm(2,it))/(R+ww);
```

```
xq2=-q(1)+(ap*(x(1,it)^2-1)+xpr(3,it)/(R+ww))*q(2)+xpr(3,it)*(R*u(it)+lm(2,it))/(R+ww);
qq=[xq1;xq2];

.....
%  *** CSTR SİSTEMİNE AİT 2. DERECE HAMILTONIAN ***
%    *** YÖNTEMİNDEKİ VEKTÖR DİFF. DENKLEMLERİ ***
function qq=yqriccsh(t,q)

% 2.Derece Hamiltonian Çözümüyle Riccati Vektör
% Diferansiyel Denklemleri (CSTR Denklemlerinin)
% Lamtil=Pdx+q Dönüşüm Formundaki q Vektörü
% t: zaman ; q: Riccati Vektörünün Durum Uzayı Vektörü
% qq: Dönen Diferansiyel Denklem Vektörü

% Yaşar BECERİKLİ- 13:32 09.03.1998

global t0; global tf; global nd; global x; global b; global u; global xpr;
global Q; global R; global xd; global ww; global lm;
global Gama; global Beta; global Da; global BB; global dd; global x2c;

hp=(tf-t0)/(nd-1); it=round(t*(nd-1)/(tf-t0)+1);

expx2=exp(x(2,it)/(1+x(2,it)/Gama)); x12g=(1-x(1,it))/(1+x(2,it)/Gama)^2;
xq1=(1+Da*expx2)*q(1)+(BB*Da*expx2+Beta^2*xpr(2,it)/(R+ww))*q(2)+xpr(2,it)*Beta*(R*u(it)+
Beta*lm(2,it))/(R+ww);
xq2=-Da*x12g*expx2*q(1)+(1+Beta-BB*Da*x12g*expx2+Beta^2*xpr(3,it)/(R+ww))*q(2)+
xpr(3,it)*Beta*(R*u(it)+Beta*lm(2,it))/(R+ww);
qq=[xq1;xq2];

.....
% *** 2. DERECE YÖNTEMLERDE GERİ BESLEME DENEMELERİ ***
%            *** VDP SİSTEMİ İÇİN ***
% ygerdnvp.m

% Burada üç bozucu duruma göre deneme yapılıyor
% i) Sistem parametresi değişmesi durumu
% ii) Başlangıç şartı değiştirilerek yapılan bozulma
% iii) Bir anda sisteme uygulanan birim basamak bozulması

% Yaşar BECERİKLİ- 17:46 05.11.1997

x0fb=x0; tdis=2; dis=0.0; ap=.1;
[tt,xff]=yrk5('yxfbvp',t0,tf,hp,x0fb);
hold on; clf; plot(tt,xff,':k',tt,x,'k'); title('GB için parametre bozması'); xlabel('zaman');
ylabel('x ; xwf');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

x0fb=[-2;3]; tdis=2; dis=0.0; ap=1.0; xxs=x;
[tt,xff]=yrk5('yxfbvp',t0,tf,hp,x0fb);
hold on; clf; plot(tt,xff,':k',tt,x,'k'); title('GB için batlangýç bozmasý'); xlabel('zaman');ylabel('x ;
xfb');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%
x0fb=x0; tdis=2; dis=5; ap=1.0;
[tt,xff]=yrk5('yxfbvp',t0,tf,hp,x0fb);
hold on; clf; plot(tt,xff,':k',tt,x,'k');title('GB için basamak bozması'); xlabel('zaman'); ylabel('x ; xfb');
```

```
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%GERİ BESLEMESİZ DURUMLAR
Kgec=K; bgec=b; K=0*K; b=0*b;
% PARAMETRE BOZMASI
x0fb=x0; tdis=2; dis=0.0; ap=.1;
[tt,xff]=yrk5('yxfbvp',t0,tf,hp,x0fb);
hold on; clf; plot(tt,xff,':k',tt,x,'k'); title('GB`siz parametre bozması'); xlabel('zaman');
ylabel('x ; xwf');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

% BAŞLANGIÇ BOZMASI
x0fb=[-2;3]; tdis=2; dis=0.0; ap=1.0;
[tt,xff]=yrk5('yxfbvp',t0,tf,hp,x0fb);
hold on; clf; plot(tt,xff,':k',tt,x,'k'); title('GB`siz baţlangýç bozmasý'); xlabel('zaman');ylabel('x ;
xwf');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

K=0*K; b=0*b;
% BASAMAK BOZMASI
x0fb=x0; tdis=2; dis=5; ap=1.0;
[tt,xff]=yrk5('yxfbvp',t0,tf,hp,x0fb);
hold on; clf; plot(tt,xff,':k',tt,x,'k'); title('GB`siz basamak bozması'); xlabel('zaman'); ylabel('x ; xwf');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%
K=Kgec; b=bgec;

.....
% *** 2. DERECE YÖNTEMLERDE GERİ BESLEME DENEMELERİ ***
%          *** CSTR SİSTEMİ İÇİN ***
% ygerdncs.m

% Burada iki bozucu duruma göre deneme yapılıyor
% i) Sistem parametresi değişmesi durumu
% ii) Başlangıç şartı değiştirilerek yapılan bozulma
% ii) Hem sistem parametresi değişmesi, hem de bir anda
% sisteme uygulanan birim basamak bozulması

% Yaşar BECERİKLİ- 00:06 08.03.1998

feed=1;
if feed==0,

x0fb=x0; tdis=40.0; dis=0.0;
%BB=8; Beta=.3; Gama=20; Da=0.072;
BB=11; Beta=1.5; Gama=20; Da=0.135;
[tt,xff]=yrk5('yxfbcs',t0,tf,hp,x0fb);

CA=(1-x(1,:))*CAf; Tr=(1+x(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
CAff=(1-xff(1,:))*CAf; Trff=(1+xff(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
Trffbp=Trff;
hold on; clf; plot(tt,xff,':k',tt,x,'k'); title('GB için parametre bozması'); xlabel('zaman');
ylabel('x ; xwf');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

hold on; clf;plot(xff(1,:),xff(2,:),'k'); grid; title('Faz Diyagramı'); xlabel('x1'); label('x2');
```

```
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%CSTR: Gerçek değerler
hold on; clf; subplot(2,1,1); plot(t,CA,'k',t,CAff,':k'); title('GB için Konsantrasyon'); xlabel('zaman');
ylabel('CA');
subplot(2,1,2); plot(t,Tr,'k',t,Trff,':k'); title('GB için Sıcaklık'); xlabel('zaman'); ylabel('T');
pause2(1); hold off; if iter==0, pause; end;if dur==1, pause; end;

%
x0fb=[.6;-4]; dis=0.0;
BB=7; Beta=.5; Gama=20; Da=0.11;xxs=x;
[tt,xff]=yrk5('yxfbcs',t0,tf,hp,x0fb);

CA=(1-x(1,:))*CAf; Tr=(1+x(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
CAff=(1-xff(1,:))*CAf; Trff=(1+xff(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
Trffbbs=Trff;
hold on; clf; plot(tt,xff,':k',tt,x,'k'); title('GB için batlangýç bozmasý'); xlabel('zaman');ylabel('x ;
xfb');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

hold on; clf; plot(xff(1,:),xff(2,:),'k'); grid; title('Faz Diyagramı'); xlabel('x1'); ylabel('x2');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%CSTR: Gerçek değerler
hold on; clf; subplot(2,1,1); plot(t,CA,'k',t,CAff,':k'); title('GB için Konsantrasyon'); xlabel('zaman');
ylabel('CA');
subplot(2,1,2); plot(t,Tr,'k',t,Trff,':k'); title('GB için Sıcaklık'); xlabel('zaman'); ylabel('T');
pause2(1); hold off; if iter==0, pause; end; if dur==1, pause; end;

% PARAMETRE+BASAMAK BOZMASI
x0fb=x0; dis=5*Gama/300;
%BB=7; Beta=.5; Gama=20; Da=0.11;
%BB=8; Beta=.3; Gama=20; Da=0.072;
BB=11; Beta=1.5; Gama=20; Da=0.135;
[tt,xff]=yrk5('yxfbcs',t0,tf,hp,x0fb);

CA=(1-x(1,:))*CAf; Tr=(1+x(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
CAff=(1-xff(1,:))*CAf; Trff=(1+xff(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
Trffbpb=Trff;
hold on; clf; plot(tt,xff,':k',tt,x,'k'); title('GB için parametre+basamak bozması'); xlabel('zaman');
ylabel('x ; xfb');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

hold on; clf; plot(xff(1,:),xff(2,:),'k'); grid; title('Faz Diyagramı'); xlabel('x1'); ylabel('x2');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%CSTR: Gerçek değerler
hold on; clf; subplot(2,1,1); plot(t,CA,'k',t,CAff,':k'); title('GB için Konsantrasyon'); xlabel('zaman');
ylabel('CA');
subplot(2,1,2); plot(t,Tr,'k',t,Trff,':k'); title('GB için Sıcaklık'); xlabel('zaman'); ylabel('T');
pause2(1); hold off; if iter==0, pause; end; if dur==1,
pause; end; dds=dd;

% PARAMETRE+BASAMAK BOZMASI
%Burada batta Tf=310 (dd=2/3) (dd=0.0 idi)yapýlarak T=40'da 5 derece düţürülüyor
x0fb=x0; dd=dds+2/3; dis=-5*Gama/300;
%BB=7; Beta=.5; Gama=20; Da=0.11;
```

```
%BB=8; Beta=.3; Gama=20; Da=0.072;
BB=11; Beta=1.5; Gama=20; Da=0.135;
[tt,xff]=yrk5('yxfbcs',t0,tf,hp,x0fb);

CA=(1-x(1,:))*CAf; Tr=(1+x(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
CAff=(1-xff(1,:))*CAf; Trff=(1+xff(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
Trffbpb2=Trff;
hold on; clf; plot(tt,xff,':k',tt,x,'k'); title('GB için parametre+basamak bozması'); xlabel('zaman');
ylabel('x ; xfb');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

hold on; clf; plot(xff(1,:),xff(2,:),'k'); grid; title('Faz Diyagramı'); xlabel('x1'); ylabel('x2');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%CSTR: Gerçek değerler
hold on; clf; subplot(2,1,1); plot(t,CA,'k',t,CAff,':k'); title('GB için Konsantrasyon'); xlabel('zaman');
ylabel('CA');
subplot(2,1,2); plot(t,Tr,'k',t,Trff,':k'); title('GB için Sıcaklık'); xlabel('zaman'); ylabel('T');
pause2(1); hold off; if iter==0, pause; end; if dur==1, pause; end;
dd=dds;

%GERİ BESLEMESİZ DURUMLAR
Kgec=K; bgec=b; K=0*K; b=0*b;
%% PARAMETRE BOZMASI
x0fb=x0; dis=0.0;
%BB=8; Beta=.3; Gama=20; Da=0.072;
BB=11; Beta=1.5; Gama=20; Da=0.135;
[tt,xff]=yrk5('yxfbcs',t0,tf,hp,x0fb);

CA=(1-x(1,:))*CAf; Tr=(1+x(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
CAff=(1-xff(1,:))*CAf; Trff=(1+xff(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
hold on; clf; plot(tt,xff,':k',tt,x,'k'); title('GB`siz parametre bozması'); xlabel('zaman');
ylabel('x ; xwf');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%
hold on; clf; plot(xff(1,:),xff(2,:),'k'); grid; title('Faz Diyagramı'); xlabel('x1'); ylabel('x2');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%CSTR: Gerçek değerler
hold on; clf; subplot(2,1,1); plot(t,CA,'k',t,CAff,':k'); title('GB`siz Konsantrasyon'); xlabel('zaman');
ylabel('CA');
subplot(2,1,2); plot(t,Tr,'k',t,Trff,':k'); title('GB`siz Sıcaklık'); xlabel('zaman'); ylabel('T');
pause2(1); hold off; if iter==0, pause; end;if dur==1, pause; end;

%CSTR: Gerçek değerler
hold on; clf; plot(t,Trffbp,'k',t,Trff,':k'); title('GB` li GB`siz Sıcaklık'); xlabel('zaman');
ylabel('Tfb, Tff');
pause2(1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%% BAŞLANGIÇ BOZMASI
x0fb=[.6;-4]; dis=0.0;
BB=7; Beta=.5; Gama=20; Da=0.11;
[tt,xff]=yrk5('yxfbcs',t0,tf,hp,x0fb);

CA=(1-x(1,:))*CAf; Tr=(1+x(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
CAff=(1-xff(1,:))*CAf; Trff=(1+xff(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
```

```
hold on; clf; plot(tt,xff,':k',tt,x,'k'); title('GB`siz baţlangýç bozmasý'); xlabel('zaman');ylabel('x ; xwf');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

hold on; clf; plot(xff(1,:),xff(2,:),'k'); grid; title('Faz Diyagramı'); xlabel('x1'); ylabel('x2');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%CSTR: Gerçek değerler
hold on; clf; subplot(2,1,1); plot(t,CA,'k',t,CAff,':k'); title('GB`siz Konsantrasyon'); xlabel('zaman');
ylabel('CA');
subplot(2,1,2); plot(t,Tr,'k',t,Trff,':k'); title('GB`siz Sıcaklık'); xlabel('zaman'); ylabel('T');
pause2(1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%CSTR: Gerçek değerler
hold on; clf; plot(t,Trffbbs,'k',t,Trff,':k'); title('GB` li GB`siz Sıcaklık'); xlabel('zaman');
ylabel('Tfb, Tff');
pause2(1); hold off; if iter==0, pause; end; if dur==1, pause; end;

K=0*K; b=0*b;
%% PARAMETRE+BASAMAK BOZMASI
x0fb=x0; dis=5*Gama/300;
%BB=7; Beta=.5; Gama=20; Da=0.11;
%BB=8; Beta=.3; Gama=20; Da=0.072;
BB=11; Beta=1.5; Gama=20; Da=0.135;
[tt,xff]=yrk5('yxfbcs',t0,tf,hp,x0fb);

CA=(1-x(1,:))*CAf; Tr=(1+x(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
CAff=(1-xff(1,:))*CAf; Trff=(1+xff(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
hold on; clf; plot(tt,xff,':k',tt,x,'k'); title('GB`siz parametre+basamak bozması'); xlabel('zaman');
ylabel('x ; xwf');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

hold on; clf; plot(xff(1,:),xff(2,:),'k'); grid; title('Faz Diyagramı'); xlabel('x1'); ylabel('x2');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%CSTR: Gerçek değerler
hold on; clf; subplot(2,1,1); plot(t,CA,'k',t,CAff,':k'); title('GB`siz Konsantrasyon'); xlabel('zaman');
ylabel('CA');
subplot(2,1,2); plot(t,Tr,'k',t,Trff,':k'); title('GB`siz Sıcaklık'); xlabel('zaman'); ylabel('T');
pause2(1); hold off; if iter==0, pause; end;if dur==1, pause; end;

%CSTR: Gerçek değerler
hold on; clf; plot(t,Trffbpb,'k',t,Trff,':k'); title('GB` li GB`siz Sıcaklık'); xlabel('zaman');
ylabel('Tfb, Tff');
pause2(1); hold off; if iter==0, pause; end;if dur==1, pause; end;

K=0*K; b=0*b;
%% PARAMETRE+BASAMAK BOZMASI
%Burada baţta Tf=310 (dd=2/3)(dd=0 idi) yapýlarak T=40'da 5 derece düţürülüyor
x0fb=x0; dd=dds+2/3; dis=-5*Gama/300;
%BB=7; Beta=.5; Gama=20; Da=0.11;
%BB=8; Beta=.3; Gama=20; Da=0.072;
BB=11; Beta=1.5; Gama=20; Da=0.135;
[tt,xff]=yrk5('yxfbcs',t0,tf,hp,x0fb);

CA=(1-x(1,:))*CAf; Tr=(1+x(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
CAff=(1-xff(1,:))*CAf; Trff=(1+xff(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
```

```
hold on; clf; plot(tt,xff,':k',tt,x,'k'); title('GB`siz parametre+basamak bozması'); xlabel('zaman');
ylabel('x ; xwf');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

hold on; clf; plot(xff(1,:),xff(2,:),'k'); grid; title('Faz Diyagramı'); xlabel('x1');ylabel('x2');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%CSTR: Gerçek değerler
hold on; clf; subplot(2,1,1); plot(t,CA,'k',t,CAff,':k'); title('GB`siz Konsantrasyon'); xlabel('zaman');
ylabel('CA');
subplot(2,1,2); plot(t,Tr,'k',t,Trff,':k'); title('GB`siz Sıcaklık'); xlabel('zaman'); ylabel('T');
pause2(1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%CSTR: Gerçek değerler
hold on; clf; plot(t,Trffbpb2,'k',t,Trff,':k'); title('GB` li GB`siz Sıcaklık'); xlabel('zaman');
ylabel('Tfb, Tff');
pause2(1); hold off; if iter==0, pause; end; if dur==1, pause; end;

dd=dds;

%%%
K=Kgec; b=bgec;
end;
dd=ds; x0fb=x0; tdis=40.0; dis=0.0;
%BB=8; Beta=.3; Gama=20; Da=0.072;
BB=11; Beta=1.5; Gama=20; Da=0.135;
[tt,xff]=yrk5('yxfbcs',t0,tf,hp,x0fb); Kgec=K; bgec=b; K=0*K; b=0*b;
[tt,xffs]=yrk5('yxfbcs',t0,tf,hp,x0fb); save xff; save xffs;

CA=(1-x(1,:))*CAf; Tr=(1+x(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
CAff=(1-xff(1,:))*CAf; Trff=(1+xff(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
CAffs=(1-xffs(1,:))*CAf; Trffs=(1+xffs(2,:)/Gama)*Tf0; Tc=Tc0+Tf0*u/Gama;
Trffbp=Trff; save Trff; save Trffs;

hold on; clf; plot(tt,xff,':k',tt,x,'k'); title('GB için parametre bozması'); xlabel('zaman');
ylabel('x ; xwf');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

hold on; clf; plot(xff(1,:),xff(2,:),'k'); grid; title('Faz Diyagramı'); xlabel('x1'); ylabel('x2');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

hold on; clf; plot(xffs(1,:),xffs(2,:),'k'); grid; title('GB`siz Faz Diyagramı'); xlabel('x1'); ylabel('x2');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%CSTR: Gerçek değerler
hold on; clf; subplot(2,1,1); plot(t,CA,'k',t,CAff,':k'); title('GB için Konsantrasyon'); xlabel('zaman');
ylabel('CA');
subplot(2,1,2); plot(t,Tr,'k',t,Trff,':k'); title('GB için Sıcaklık'); xlabel('zaman'); ylabel('T');
pause2(1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%%
hold on; clf; plot(tt,x(2,:),'k',tt,xff(2,:),':k',tt,xffs(2,:),':k'); title('Bozucu altında GB`li ve GB`siz OK');
xlabel('zaman'); ylabel('x2;x2fb;x2ff');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;
%
hold on; clf; plot(tt,xff(2,:),'k',tt,xffs(2,:),':k'); title('Bozucu altında GB`li ve GB`siz OK');
xlabel('zaman'); ylabel('x2fb;x2ff');
```

```
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%
hold on; clf; plot(tt,Trff,'k',tt,Trffs,':k'); title('Bozucu altında GB`li ve GB`siz OK');
xlabel('zaman'); ylabel('Tfb;Tff');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

.....
% *** DİREK ÖLÇÜT AZALTILMASI YÖNTEMİNDE ELDE EDİLEN ***
%           *** SONUÇLARIN GRAFİKLERİ ***
%         *** VDP VE CSTR SİSTEMLERİ İÇİN ***
% ygraf.m

% Çizilen Grafikler:
% Artırılmış Hamiltonian (Htil), Htil 'in artırımsal kontrol
% fonksiyonuna göre gradyanı (Hdu), Matris Rikkati
% Diff. Denk. Çözümü (xpr)
% Vektör Rikkati Diff. Denk. Çözümü (xqr), Kazanç Matrisi (K),
% Sürücü Kazanç Vektörü (b)

% Yaşar BECERİKLİ- 17:43 05.11.1997

hold on; clf; subplot(2,1,1); plot(t,Htil(iter+1,:),'k'); grid;  title('Art. Hamiltonian'); xlabel('zaman');
ylabel('Htil');
subplot(2,1,2); plot(t,Hdu(iter+1,:),'k'); grid; title('Htildu'); xlabel('zaman'); ylabel('Htildu');
pause2(0.1); hold off; if dur==1, pause; end;

hold on; clf; subplot(2,3,1); plot(t,xpr(1,:),'k'); grid; title('Ricc. P'); xlabel('zaman'); ylabel('P11');
subplot(2,3,2); plot(t,xpr(2,:),'k'); grid; title('Ricc. P'); xlabel('zaman'); ylabel('P12');
subplot(2,3,3); plot(t,xpr(3,:),'k');grid; title('Ricc. P'); xlabel('zaman'); ylabel('P22');
subplot(2,3,4); plot(t,xqr(1,:),'k'); grid; title('Sür. q'); xlabel('zaman'); ylabel('q1');
subplot(2,3,5); plot(t,xqr(2,:),'k'); grid; title('Sür. q'); xlabel('zaman');ylabel('q2');
pause2(0.1); hold off; if dur==1, pause; end;

hold on; clf; subplot(2,2,1); plot(t,K(:,1),'k'); grid; title('Kazanç 1'); xlabel('zaman'); ylabel('K11');
subplot(2,2,2); plot(t,K(:,2),'k'); grid; title('Kazanç 2'); xlabel('zaman'); ylabel('K12');
subplot(2,2,3); plot(t,bb(iter+1,:),'k'); grid; title('Sürücü Kazancı'); xlabel('zaman'); ylabel('b');
pause2(0.1); hold off; if dur==1, pause; end;

.....
% *** 2. DERECE HAMILTONIAN YÖNTEMİNDE ELDE EDİLEN ***
%           *** SONUÇLARIN GRAFİKLERİ ***
%         *** VDP VE CSTR SİSTEMLERİ İÇİN ***
% ygrafh.m

% Çizilen Grafikler:
% Adjoint durum değişkeni (lm),
% Gerekli koţullarýn sađlandýđý Hamiltonian (Ham),
% Ham 'ın kontrol fonksiyonuna göre gradyanı (Hu),
% Artırılmış Hamiltonian (Htil),
% Htil 'in artırımsal kontrol fonksiyonuna göre gradyanı (Hdu),
% Matris Rikkati Diff. Denk. Çözümü (xpr),
% Vektör Rikkati Diff. Denk. Çözümü (xqr), Kazanç Matrisi (K),
% Sürücü Kazanç Vektörü (b)

% Yaşar BECERİKLİ- 19:02 05.11.1997
```

```
hold on; clf; subplot(2,3,1); plot(t,lm(1,:),'k'); grid; title('ADJOINT'); xlabel('zaman'); ylabel('lam1');
subplot(2,3,2); plot(t,lm(2,:),'k'); grid; title('ADJOINT'); xlabel('zaman'); ylabel('lam2');
subplot(2,3,3); plot(t,Ham(iter+1,:),'k'); grid; title('HAMILTONIAN'); xlabel('zaman'); ylabel('H');
subplot(2,3,4); plot(t,Hu(iter+1,:),'k'); grid; title('Hu'); xlabel('zaman'); ylabel('Hu');
subplot(2,3,5); plot(t,Htil(iter+1,:),'k'); grid; title('Artırılmış HAMILTONIAN'); xlabel('zaman');
ylabel('Htil');
subplot(2,3,6); plot(t,Hdu(iter+1,:),'k'); grid; title('Htildudot'); xlabel('zaman'); ylabel('Hdu');
pause2(0.1); hold off; if dur==1, pause; end;

hold on; clf; subplot(2,3,1); plot(t,xpr(1,:),'k'); grid; title('Ricc. P'); xlabel('zaman'); ylabel('P1');
subplot(2,3,2); plot(t,xpr(2,:),'k'); grid; title('Ricc. P'); xlabel('zaman'); ylabel('P2');
subplot(2,3,3); plot(t,xpr(3,:),'k'); grid; title('Ricc. P'); xlabel('zaman'); ylabel('P3');
subplot(2,3,4); plot(t,xqr(1,:),'k'); grid; title('Ricc. q'); xlabel('zaman'); ylabel('q1');
subplot(2,3,5); plot(t,xqr(2,:),'k'); grid; title('Ricc. q'); xlabel('zaman'); ylabel('q2');
pause2(0.1); hold off; if dur==1, pause; end;

hold on; clf; subplot(2,2,1); plot(t,K(:,1),'k'); grid; title('Kazanç 1'); xlabel('zaman'); ylabel('K1');
subplot(2,2,2); plot(t,K(:,2),'k'); grid; title('Kazanç 2'); xlabel('zaman'); ylabel('K2');
subplot(2,2,3); plot(t,bb(iter+1,:),'k'); grid; title('Sürücü Kazancı');xlabel('zaman'); ylabel('b');
pause2(0.1); hold off; if dur==1, pause; end;


.....
% *** 2. DERECE YÖNTEMLERDEKİ KONTROL,DURUM VE ÖLÇÜTÜN ***
%     *** İTERASYONLA DEĞİŞİMLERİNİN GRAFİKLERİ ***
%         *** VDP VE CSTR SİSTEMLERİ İÇİN ***
% ycsod.m

% Çizilen Grafikler:
% Kontrol değişimi, Durum değişimi, Ölçüt değişimi
% Eğer Kontrol sınırlaması için kullanılan W seçilmişse
% W 'nın ve alfa'nın değişimleri de çizilir

% Yaşar BECERİKLİ- 17:45 05.11.1997

%KONTROL
hold on; clf; plot(t,uiter,'k',t,u,':k') grid; title('Kontrol Değişimi'); xlabel('zaman');
ylabel('u iterasyon');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%STATES
hold on; clf; subplot(2,1,1); plot(t,x1iter,'k',t,x(1,:),':k') grid; title('1.Durum'); xlabel(' zaman');
ylabel('x1');
subplot(2,1,2); plot(t,x2iter,'k',t,x(2,:),':k') grid; title('2.Durum'); xlabel('zaman'); ylabel('x2');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%ÖLÇÜT
hold on; clf; plot(iteration,J,'k') grid; title('Performans Ölçütü'); xlabel('iterasyon'); ylabel('J');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%Yarı Logaritmik Ölçüt
hold on; clf; semilogy(iteration,J,'k') grid;  title('Log. Performans Ölçütü');
xlabel('iterasyon'); ylabel('logJ');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end;

%W ve alfa
if wyon==1, [ait,bit]=size(iteration); itert=1:bit-1;
hold on; clf; subplot(2,1,1); plot(itert,abs(alphaiter),'k'); grid; title('alfa `nın Değişimi');
```

```
xlabel('iterasyon'); ylabel('alfa');
subplot(2,1,2); plot(iteration,witer,'k'); grid;  title('W nın Değişimi');
xlabel('iterasyon'); ylabel('W');
pause2(0.1); hold off; if iter==0, pause; end; if dur==1, pause; end; end;

.....
% ***DİNAMİK SİNİR AĞLARI (DSA) VE EĞİTİM ALGORİTMALARI ***
function [p,pp,E,taoiter,iter]=ydtdnn(dneu,gsay,csay,xnot,tnot,tfinal,ndata,dene,yont,dur,pattern)

% Bu fonksiyon "yoptim" optimizasyon fonksiyonunu kullanarak
% 4 optimizasyon yöntemi ile DSA modelini eğitir
% Fonsiyona girilen değişkenler:
% dneu: Dinamik nöron sayısı
% gsay: Giriţ sayýsý
% csay: Çıkış sayısı
% xnot: DSA sisteminin baţlangýç koţullarý
% tnot, tfinal: Sırayla, başlangıç ve son zaman
% ndata: Problem çözümünde kullanılacak veri sayısı. (Bu sayıdan dife-
%       ransiyel denklem çözümündeki 'delta' integrasyon adımı bulunur)
% dene: Optimal kontrol veya denemeler için kullanılır:
%       dene=1 ise, OPTİMAL KONTROL eğitimi
%       dene=2 ise, değişik denemeler
%       dene=3 ise, DSA sadece yüklenen parametrelerine göre çıkış verir
% yont: DSA eğitim yöntemleri: 1= En Hızlı Azalan Gradyan (SDG)
%                              2= Ölçeklenmiş Konjuge Gradyan (SCG)
%                              3= DAVIDON-FLETCHER-POWEL (DFP)')
%                              4= BROYDEN-FLETCHER-GOLFARB-SHANNO (BFGS)
% dur: DSA çıkışlarının 1 ise her iterasyonda bekler, 0 ise sürer
% pattern: Eğitilecek giriş-çıkış dünyasına ait fonksiyonlar seti sayısı
% Fonsiyondan dönen değişkenler:
% p: Eğitim sonunda DSA ait parametre vektörü,
% pp: Eğitim süresince DSA ait parametre vektörlerinin değişimi
% E: Ölçüt'ün eğitim süresince (iterasyon) değişim
% taoiter: Tek boyutlu arama parametresinin (adım büyüklüğü)
%         iterasyonla değişimi
% iter: iterasyon vektörü
% NOT: Eğer dene=3 ise, p=zdn olup DSA çıkışını, pp=1, E=1, taoiter=1,
%      iter=1 değerlerini alır

% Yaşar BECERİKLİ- 20:16 04.11.1997

global err; global udn; global wdn; global pdn; global qdn; global Tdn;
global bdn; global gamadn; global betadn;
global Adn; global xdn;
global dns; global gs; global cs;
global t0; global tf; global hp; global xdn0;
global xdn; global ysig; global zdnd; global crp;

xdn0=xnot; t0=tnot; tf=tfinal; nd=ndata; train=dene; yontem=yont;
dns=dneu; gs=gsay; gss=gs; cs=csay; hp=(tf-t0)/(nd-1); hn=-hp;
sw=dns^2; sp=dns*gs; sq=cs*dns; sb=dns; sT=dns; sgama=dns; sbeta=dns;

%OPTİMAL KONTROL EĞİTİMİ
%                       w p q b Tg be
if train==1, crpwg=[1 1 1 1 1 1 1]';
 load xdpattern; load upattern; [ag,bg]=size(xdpattern); [ac,bc]=size(upattern);
 gpat=ag/pattern; cpat=ac/pattern; udn=xdpattern([1:gpat],:); zdnd=upattern([1:cpat],:);
```

```
gs=gss; yoctrain;
end;%%END OF train=1

if train==2, clc;
 denemeler=input('DENEMELER; EVET=1:');
if denemeler==1, clc;
 deneme=input('1=2-DNN lu DENEME; 2=GENEL DENEME:');
 crpwg=[1 1 0 0 0 0 0]';
if deneme==1,  y2dnden;
end;%%END OF DENEME=1
if deneme==2, ygenden;
end;%%END OF DENEME=2
end;%%END OF DENEMELER
end;%%END OF train=2

if train==3, load udn; end;%%END OF train=3

agir=input('Ağırlıklar Yüklensin mi?: Evet=1:');
if agir==1, load wdn; load pdn; load pdn; load qdn;
  load bdn; load Tdn; load gamadn; load betadn; end;

%   w p q b T g be
crp=[1 1 1 1 1 0 0]';

%ITERATION
wdn=wdn'; pdn=pdn'; qdn=qdn';
wdnit=wdn(:); wdn=wdn';
pdnit=pdn(:); pdn=pdn';
qdnit=qdn(:); qdn=qdn';
bdnit=bdn; Tdnit=diag(Tdn); gamadnit=gamadn; betadnit=betadn;

pardn0=[crp(1)*wdnit;crp(2)*pdnit;crp(3)*qdnit;crp(4)*bdnit;...
    crp(5)*Tdnit;crp(6)*gamadnit;crp(7)*betadnit];

pardn=[];
if crp(1)==1, pardn=pardn0([1:sw]); end; ss=sw;
if crp(2)==1, pardn=[pardn;pardn0([ss+1:ss+sp])]; end; ss=ss+sp;
if crp(3)==1, pardn=[pardn;pardn0([ss+1:ss+sq])]; end; ss=ss+sq;
if crp(4)==1, pardn=[pardn;pardn0([ss+1:ss+sb])]; end; ss=ss+sb;
if crp(5)==1, pardn=[pardn;pardn0([ss+1:ss+sT])]; end; ss=ss+sT;
if crp(6)==1, pardn=[pardn;pardn0([ss+1:ss+sgama])]; end; ss=ss+sgama;
if crp(7)==1, pardn=[pardn;pardn0([ss+1:ss+sbeta])]; end;

size(pardn)
[t,xdn]=yrk5('yxdn',t0,tf,hp,xdn0);
ysig=ysigmoid(xdn,gamadn,betadn); zdn=qdn*ysig; if train~=3, err=zdn-zdnd; end;

hold on; clf; if train ~=3, plot(t,zdnd,'k',t,zdn,':k'); title('Process ve DSA Çıkışı'); xlabel('zaman');
ylabel('z; zd');
elseif train==3, plot(t,zdn,'k'); title('DSA Çıkışı'); xlabel('zaman'); ylabel('z'); end; pause; hold off;

save wdn wdn; save pdn pdn; save qdn qdn; save bdn bdn; save Tdn Tdn; save gamadn gamadn;
save betadn betadn;

if train==3, p=zdn; pp=1; E=1; taoiter=1; iter=1; end;%%END OF train=3

if train~=3,
```

```
% pdn: Giriş ile dinamik nöronlar bağlayan parametre matrisi
% qdn: Dinamik nöronlar ile çıkışları bağlayan parametre matrisi
% bdn: Girişlerden bağlanan bias parametre vektörü
% Tdn: Dinamik nöronların zaman sabiti parametre matrisi (diagonal)
% gamadn: Aktivasyon fonksiyonunun lineer olmama parametre vektörü (>0)
% betadn: Aktivasyon fonksiyonunun bias parametre vektörü

% Yaşar BECERİKLİ- 20:38 12.10.1997

hp=(tf-t0)/(nd-1); hn=-hp;

%INITIALIZE
wdn=randn(dns); if crpwg(1)==0, wdn=ones(dns); end;
pdn=randn(dns,gs); if crpwg(2)==0, pdn=ones(dns,gs); end;
qdn=randn(cs,dns); if crpwg(3)==0, qdn=ones(cs,dns); end;
bdn=randn(dns,1); if crpwg(4)==0, bdn=zeros(dns,1); end;
Tdn=abs(diag(randn(1,dns),0))+.2*eye(dns); if crpwg(5)==0, Tdn=eye(dns); end;
gamadn=abs(randn(dns,1)); if crpwg(6)==0, gamadn=ones(dns,1); end;
betadn=randn(dns,1); if crpwg(7)==0, betadn=zeros(dns,1); end;

.....
% *** 2-DİNAMİK NÖRONLU BİR SİSTEMİ DİĞER 2-DİNAMİK NÖRONLU ***
%           *** SİSTEMLE MODELLEME ÖRNEĞİ ***
% y2dnden.m

% Burada 2-Dinamik nöronlu sistem parametrelerine göre çıkış
% belirleniyor ve diğer 2-Dinamik nöronlu sistem başlangıç
% parametreleri atanıyor

% Yaşar BECERİKLİ- 21:40 02.11.1997
```

```
nd=151; t0=0; tf=10; hp=(tf-t0)/(nd-1); hn=-hp; dns=2; gs=1; cs=1;
sw=dns^2; sp=dns*gs; sq=cs*dns; sb=dns; sT=dns; sgama=dns; sbeta=dns;

for tu=t0:hp:tf,
it=round(tu/hp+1);
if tu<=3.0, utt(:,it)=1.0; end;
if tu>3, utt(:,it)=-1.0; end;
end; udn=utt; udnp=udn;

wdn=[0 -10;10 0]; pdn=[5 -5]'; Tdn=[1 0;0 1]; qdn=[1 1]; gamadn=[1.0 1.0]'; betadn=[0.0 0.0]';
bdn=[0 0]';
wdnd=wdn; pdnd=pdn; qdnd=qdn; bdnd=bdn; Tdnd=Tdn; gamadnd=gamadn; betadnd=betadn;
xdn0=[1.0 1.0]';

[t,xdnd]=yrk5('yxdn',t0,tf,hp,xdn0);
yxdnd=ysigmoid(xdnd,gamadn,betadn); %PROCESS ÇIKIŞI
zdnd=qdn*yxdnd; zdndp=zdnd;

%GERİYE DOĞRU INTEGRATION DENEMESİ
xf=xdnd(:,nd);
[t,xdng]=yrk5('yxdn',t0,tf,hn,xf);
yxdng=ysigmoid(xdng,gamadn,betadn); zdng=qdn*yxdng;

plot(t,zdnd,'k'); grid; title('DSA Çıkışı'); xlabel('zaman'); ylabel('z'); pause;
plot(t,zdnd,'k',t,zdng,':k'); grid; pause;

%INITIALIZE
wdn=[0 -5;5 0]; pdn=[2.5 -2.5]'; qdn=[1 1]; bdn=[0 0]'; Tdn=[1 0;0 1]; gamadn=[1.0 1.0]';
betadn=[0.0 0.0]'; xdn0=[1.0 1.0]';

.....
% *** N-DİNAMİK NÖRONLU BİR SİSTEMİ DİĞER N-DİNAMİK NÖRONLU ***
%          *** SİSTEMLE MODELLEME ÖRNEĞİ ***
% ygenden.m

% Burada N-Dinamik nöronlu sistem parametrelerine göre çıkış
% belirleniyor ve diğer N-Dinamik nöronlu sistem başlangıç
% parametreleri atanıyor

% Yaşar BECERİKLİ- 22:16 02.11.1997

nd=501; t0=0; tf=10; hp=(tf-t0)/(nd-1); hn=-hp; dns=3; gs=1; cs=1;
sw=dns^2; sp=dns*gs; sq=cs*dns; sb=dns; sT=dns; sgama=dns; sbeta=dns;

ugec1=ones(gs,1); ugec2=zeros(gs,1);
for tu=t0:hp:tf,
it=round(tu/hp+1);
if tu<=3.0, ut2(:,it)=ugec1; end;
if (tu>2.0 & tu<=3), ut2(:,it)=ugec2; end;
if tu>3, ut2(:,it)=ugec2; end;
end;
udn=ut2; udnp=udnd;

%BAŞLANGIÇ ATAMASI
xdn0=[];
for tu=1:dns,
if dns==1, xdn0(:,tu)=1; elseif tu<=fix(dns/4), xdn0(:,tu)=1; else xdn0(:,tu)=0; end;
```

```
end; xdn0=xdn0';

wdn=randn(dns); wdnd=wdn;
pdn=randn(dns,gs); pdnd=pdn;
qdn=randn(cs,dns); qdnd=qdn;
bdn=randn(dns,1); bdnd=bdn;
Tdn=abs(diag(randn(1,dns),0))+.1*eye(dns); Tdnd=Tdn;
gamadn=abs(randn(dns,1)); gamadnd=gamadn;
betadn=randn(dns,1); betadnd=betadn;

[t,xdnd]=yrk5('yxdn',t0,tf,hp,xdn0);
yxdnd=ysigmoid(xdnd,gamadn,betadn);  %PROCESS ÇIKIŞI
zdnd=qdn*yxdnd; zdndp=zdnd;

%GERİYE DOĞRU INTEGRATION DENEMESİ
xf=xdnd(:,nd);
[t,xdng]=yrk5('yxdn',t0,tf,hn,xf);
yxdng=ysigmoid(xdng,gamadn,betadn); zdng=qdn*yxdng; err=zdng-zdnd;

plot(t,zdnd,'k'); grid; pause;
plot(t,zdnd,'k',t,zdng,':k'); grid; pause;

%INITIALIZE
wdn=randn(dns); if crpwg(1)==0, wdn=ones(dns); end;
pdn=randn(dns,gs); if crpwg(2)==0, pdn=ones(dns,gs); end;
qdn=randn(cs,dns); if crpwg(3)==0, qdn=ones(cs,dns); end;
bdn=randn(dns,1); if crpwg(4)==0, bdn=zeros(dns,1); end;
Tdn=abs(diag(randn(1,dns),0))+.2*eye(dns); if crpwg(5)==0, Tdn=eye(dns); end;
gamadn=abs(randn(dns,1)); if crpwg(6)==0, gamadn=ones(dns,1); end;
betadn=randn(dns,1); if crpwg(7)==0, betadn=zeros(dns,1); end;

.....
% *** DSA İÇİN DURUM DİFF DENKLEMLERİ ***
function xdot=yxdn(t,x)

% DSA 'da dinamik nöronlar için çözüm yapılıyor
% t: zaman, x: nöron durum vektörü
% xdot: Dönen Dinamik nöron Diff. Denklem Vektörü

% Yaşar BECERİKLİ- 21:01 05.11.1997

global g; global wdn; global pdn; global qdn; global Tdn; global bdn; global gamadn; global betadn;
global udn; global hp;

if t~=-9999,
 it=round(t/hp+1);  sig=ysigmoid(x,gamadn,betadn); xdot=Tdn\(-x+wdn*sig+pdn*udn+bdn);
else
 [a,b]=size(x);
 sig=ysigmoid(x,gamadn,betadn);  bdd=bdn*ones(1,b); xdot=Tdn\(-x+wdn*sig+pdn*udn+bdd);
end;

.....
%   *** STEEPEST DESCENT GRADYAN (SDG, S. BACK-PROPAGATION-BP) ***
% *** SCALED CONJUGATE GRADYAN (SCG), DAVIDON-FLETCHER-POWEL (DFP) ***
%      *** VE BROYDEN-FLETCHER-GOLFARB-SHANNO (BFGS)') ***
function [x,xx,J,alff,iter]=yoptim(cost,p0,yontem,pattern,udnp,zdndp,dur,grad)
```

```
% Bu fonksiyon 4 optimizasyon yöntemini icra eder
% Fonsiyona girilen değişkenler:
% cost: 'cost' bir .m dosyası olup en küçük yapılacak ölçütttür
% p0: 'cost' fonksiyonunun başlangıç parametre vektörü
% yontem: Optimizasyon yöntemleri:
%       1= En Hızlı Azalan Gradyan (SDG)
%       2= Ölçeklenmiş Konjuge Gradyan (SCG)
%       3= DAVIDON-FLETCHER-POWEL (DFP)')
%       4= BROYDEN-FLETCHER-GOLFARB-SHANNO (BFGS)
% pattern: Eğitilecek giriş-çıkış dünyasına ait fonksiyonlar seti sayısı
% udnp: Giriş seti
% zdndp: İstenen çıkış seti
% dur: Çizilecek grafikler için 1 ise bekler, 0 ise sürekli devam eder
% grad: 'grad' bir .m dosyası olup ölçütün parametrelere göre gradyanı
% Fonsiyondan dönen değişkenler:
% x: Optimizasyon sonunda sisteme ait parametre vektörü,
% xx: Optimizasyon süresince sisteme ait parametre vektörlerinin değişimi
% J: Ölçütün optimizasyon süresince (iterasyon) değişimi
% alff: Tek boyutlu arama parametresinin (adım büyüklüğü) iterasyonla değişimi
% iter: iterasyon vektörü

% Yaşar BECERİKLİ- 15:47 27.10.1997

if pattern>1, [ag,bg]=size(udnp); [ac,bc]=size(zdndp);
 gpat=ag/pattern; cpat=ac/pattern;
 udn=udnp([1:gpat],:);
 zdnd=zdndp([1:cpat],:);
 patgir=gpat; patcik=cpat; pats=1;
else udn=udnp; zdnd=zdndp; end;

opt_yon=yontem; dfp_bfgs=opt_yon;

x=p0; flaq=0; flaq1=0; flaq2=0; alfixs=1e1;
tol_grad=1e-5; tol_x=1e-3;

ugir=udn; zcikd=zdnd;
[J(1),t,xdn,ysig,zcik]=feval(cost,x,ugir,zcikd);

%SDG
if (opt_yon==1), ugir=udn; zcikd=zdnd;
xx=x; [a,b]=size(x); k=0; dx=ones(a,1); gr=feval(grad,x,ugir);
while 1,
k=k+1, iter(k)=k-1; lop=lop+1;
gr=feval(grad,x,ugir);

if k~=1, dx=x-xs; dgr=gr-grs; dxbag=dx./xs;
 mdx=max(dx==0); mdgr=max(dgr==0);
 if ((sqrt(gr'*gr/2)<tol_grad) | (sqrt(dxbag'*dxbag/2)<tol_x/2)),
  tao=ylisrch(cost,x0,d,JJ,alfix,ugir,zcikd), alff(k)=tao; disp('Yeterli Hassasiyet'); %clear iter(k);
  gr, dx, yciz_yuk(1,k,t,zcik,zcikd,dur); pause; break; end;
end;

d=-gr; x0=x;

if sqrt(gr'*gr/2)>tol_grad/1e15,
 if k==1, alfix=alfixs; else alfix=tao; end;
```

```
JJ=J(k)
tao=ylisrch(cost,x0,d,JJ,alfix,ugir,zcikd), alff(k)=tao;

else tao=1; end;%END OF if (sqrt(gr'*gr/2)>tol_grad)

xs=x; grs=gr;
x=x+tao*d; x=ytrsh_p(x);
[J(k+1),t,xdn,ysig,zcik]=feval(cost,x,xs,zcikd); xx(:,k+1)=x;

if k>1, dx=x-xs; dgr=gr-grs; dxbag=dx./xs; end;
if flaq1>2, flaq1=0;
 if flaq2>2, flaq2=0;
  if ((sqrt(gr'*gr/2)<tol_grad) & (sqrt(dxbag'*dxbag/2)<tol_x/2)),
  tao=ylisrch(cost,x0,d,JJ,alfix,ugir,zcikd),  alff(k)=tao; disp('Yeterli Hassasiyet');
  gr, dx, yciz_yuk(1,k,t,zcik,zcikd,dur); pause; break; end;
 end; end;
flaq1=flaq1+1; flaq2=flaq2+1;

%Çıkışların çizimi ve Parametrelerin kaydı
yciz_yuk(1,k,t,zcik,zcikd,dur);

if pattern>1, pats=pats+1;
 udn=udnp([patgir+1:patgir+gpat],:);
 zdnd=zdndp([patcik+1:patcik+cpat],:);
 patgir=patgir+gpat; patcik=patcik+cpat;
 if pats==pattern, patgir=0; patcik=0; pats=0; end;
  ugir=udn; zcikd=zdnd;
[J(k+1),t,xdn,ysig,zcik]=feval(cost,x,xs,zcikd);
end;

end;%%END OF while (ANA DÖNGÜ)
k
end;%%END OF SDG=1

%SCG
if (opt_yon==2), ugir=udn; zcikd=zdnd;
 xx=x; [a,b]=size(x); k=0; dx=ones(a,1); S=eye(a); SS=zeros(a); lop=0; gr=feval(grad,x,ugir);

while 1,
 k=k+1, iter(k)=k-1; lop=lop+1;
 gr=feval(grad,x,ugir);

if k~=1, dx=x-xs; dgr=gr-grs; dxbag=dx./xs;
  beta=((gr-grs)'*S*gr)/(grs'*S*grs);
  mdx=max(dx==0); mdgr=max(dgr==0);
 if ((sqrt(gr'*gr/2)<tol_grad) & (sqrt(dxbag'*dxbag/2)<tol_x/2)),
  tao=ylisrch(cost,x0,d,JJ,alfix,ugir,zcikd),  alff(k)=tao; disp('Yeterli Hassasiyet'); %clear iter(k);
   gr, dx, yciz_yuk(1,k,t,zcik,zcikd,dur); pause; break; end;
else beta=0; d=zeros(a,1); end;

if flaq>2,
if (sqrt(dxbag'*dxbag/2)<=tol_x/2), flaq=0; beta=0; S=eye(a), end; end; flaq=flaq+1;

d=-S*gr+beta*d; x0=x;

if sqrt(gr'*gr/2)>tol_grad/1e15,
 if k==1, alfix=alfixs; else alfix=tao; end;
```

```
if flaq==1,
if (sqrt(dxbag'*dxbag/2)<=tol_x/2), alfix=alfixs, end; end;

JJ=J(k)
tao=ylisrch(cost,x0,d,JJ,alfix,ugir,zcikd), alff(k)=tao;

else tao=1; end;%% END OF if (sqrt(gr'*gr/2)>tol_grad)

mu=tao/(gr'*S*gr); SS=SS+d*d'; %S=eye(a); %S=SS;

xs=x; grs=gr;
x=x+tao*d; x=ytrsh_p(x);
[J(k+1),t,xdn,ysig,zcik]=feval(cost,x,xs,zcikd); xx(:,k+1)=x;
if lop==a, S=SS; lop=0; beta=0; SS=zeros(a); end;%%END OF if

if flaq1>2, flaq1=0;
if flaq2>2, flaq2=0;
 if ((sqrt(gr'*gr/2)<tol_grad) & (sqrt(dxbag'*dxbag/2)<tol_x/2)),
  tao=ylisrch(cost,x0,d,JJ,alfix,ugir,zcikd),  alff(k)=tao; disp('Yeterli Hassasiyet');
  gr, dx, yciz_yuk(1,k,t,zcik,zcikd,dur); pause; break; end;
end; end;
flaq1=flaq1+1; flaq2=flaq2+1;

%Çıkışların çizimi ve Parametrelerin kaydı
yciz_yuk(1,k,t,zcik,zcikd,dur);

if pattern>1, pats=pats+1;
 udn=udnp([patgir+1:patgir+gpat],:);
 zdnd=zdndp([patcik+1:patcik+cpat],:);
 patgir=patgir+gpat;  patcik=patcik+cpat;
 if pats==pattern, patgir=0; patcik=0; pats=0; end;
 ugir=udn; zcikd=zdnd;
[J(k+1),t,xdn,ysig,zcik]=feval(cost,x,xs,zcikd);
end;
end;%%END OF while (ANA DÖNGÜ)
k
end;%%END OF opt=yon=2 (SCG)

% DFP ve BFGS
if ((opt_yon==3) | (opt_yon==4)), ugir=udn; zcikd=zdnd;
xx=x; [a,b]=size(x); k=0; gr=feval(grad,x,ugir); dx=ones(a,1);

while 1,
k=k+1, iter(k)=k-1;
 gr=feval(grad,x,ugir); %gradyan(:,k)=gr;

if k~=1, dx=x-xs; dgr=gr-grs; dxbag=dx./xs;
  mdx=max(dx==0); mdgr=max(dgr==0);
 if k<=3, if ((sqrt(gr'*gr/2)<tol_grad) | (sqrt(dxbag'*dxbag/2)<tol_x/2)),
  tao=ylisrch(cost,x0,d,JJ,alfix,ugir,zcikd), alff(k)=tao; disp('Yeterli Hassasiyet'); %clear iter(k);
  gr, dx, yciz_yuk(1,k,t,zcik,zcikd,dur); pause; break; end; end;

 if ((sqrt(gr'*gr/2)<tol_grad) & (sqrt(dxbag'*dxbag/2)<tol_x/2)),
  tao=ylisrch(cost,x0,d,JJ,alfix,ugir,zcikd), alff(k)=tao; disp('Yeterli Hassasiyet'); %clear iter(k);
  gr, dx, yciz_yuk(1,k,t,zcik,zcikd,dur); pause; break; end;

%DFP
```

```
 if dfp_bfgs==3, inH=inH+(dx*dx')/(dx'*dgr)-((inH*dgr)*(inH*dgr)')/(dx'*inH*dgr);
%BFGS
elseif dfp_bfgs==4, inH=(eye(a)-(dx*dgr')/(dx'*dgr))*inH*(eye(a)-(dx*dx')/(dx'*dgr))'...
                          +(dx*dx')/(dx'*dgr); end;
else inH=eye(a); end;

if flaq>2, sqrt(dxbag'*dxbag/2)
if (sqrt(dxbag'*dxbag/2)<=tol_x/2), flaq=0; inH=eye(a), end; end; flaq=flaq+1;

d=-inH*gr; x0=x;

if sqrt(gr'*gr/2)>tol_grad/1e15,
if k==1, alfix=alfixs; else alfix=tao; end;
if flaq==1,
if (sqrt(dxbag'*dxbag/2)<=tol_x/2), alfix=alfixs, end; end;

JJ=J(k)
tao=ylisrch(cost,x0,d,JJ,alfix,ugir,zcikd), alff(k)=tao;
else tao=1; end;

xs=x; grs=gr;
x=x+tao*d; x=ytrsh_p(x);

[J(k+1),t,xdn,ysig,zcik]=feval(cost,x,xs,zcikd); xx(:,k+1)=x;

if flaq1>2, flaq1=0;
if flaq2>2, flaq2=0;
 if ((sqrt(gr'*gr/2)<tol_grad) & (sqrt(dxbag'*dxbag/2)<tol_x/2)),
  tao=ylisrch(cost,x0,d,JJ,alfix,ugir,zcikd),  alff(k)=tao;  disp('Yeterli Hassasiyet');
  gr, dx, yciz_yuk(1,k,t,zcik,zcikd,dur); pause; break; end;
end; end;
flaq1=flaq1+1; flaq2=flaq2+1;

%Çıkışların çizimi ve Parametrelerin kaydı
yciz_yuk(1,k,t,zcik,zcikd,dur);

if pattern>1, pats=pats+1;
udn=udnp([patgir+1:patgir+gpat],:);
zdnd=zdndp([patcik+1:patcik+cpat],:);
patgir=patgir+gpat; patcik=patcik+cpat;
 if pats==pattern, patgir=0; patcik=0; pats=0; end;
ugir=udn; zcikd=zdnd;
[J(k+1),t,xdn,ysig,zcik]=feval(cost,x,xs,zcikd);
end;
end;%%END OF WHILE ||gr||>1e-15
k
end;%%END OF opt_yon=3,4 (DFP,BFGS)

.....
% *** KÜBİK YAKLAŞIMLI TEK BOYUTLU ARAMA ALGORİTMASI ***
function alfa=ylisrch(cost,x0,d,J,alfix,ugirdn,zcikdn)

% Bu fonksiyon tek boyutlu arama yapar
% Yöntem olarak kübik yaklaşım kullanılmıştır
% Tek boyut parametresi sınırlaması 2'nin pozitif veya negatif üssü
% olarak değiştirilerek bulunuyor ve sonra kübik yaklaşım yapılıyor
% Fonsiyona girilen değişkenler:
```

```
% cost: 'cost' bir .m dosyası olup en küçük yapılacak ölçütttür
% x0: 'cost' fonksiyonunun başlangıç parametre vektörü
% d: O anki arama yönü vektörü
% J: O anki ölçüt değeri
% alfix: Tek boyut parametresinin başlangıç değeri
% ugirdn: Dinamik sistemlerde giriş fonksiyonu
% zcikdn: Dinamik sistemlerde istenen çıkış fonksiyonu
% Fonsiyondan dönen değişkenler:
% alfa: O durumdaki en iyi yaklaşık tek boyut parametre değeri

% Yaşar BECERİKLİ- 22:50 27.10.1997

udn=ugirdn; zdnd=zcikdn; k=1;
 Jpol(1)=J(k); xg=x0+alfix*d; Jpol(2)=feval(cost,xg,udn,zdnd);
 alf(2)=alfix; loop=2;
if Jpol(loop)<=Jpol(loop-1),
%Yön Belirleme
 loop=loop+1; alfv=[alfix*2^(loop-2);alfix*2^(1-loop)];
 xg1=x0+alfv(1)*d; xg2=x0+alfv(2)*d;
 Jpol1=feval(cost,xg1,udn,zdnd); Jpol2=feval(cost,xg2,udn,zdnd);

if ((Jpol2>=Jpol(2)) & (Jpol1>Jpol(2)) & (Jpol1<=J(k))), loop=loop+1;
 alf(loop)=alfv(1); alf(loop-1)=alf(loop-2);  alf(loop-2)=alfv(1);
 Jpol(loop)=Jpol1; Jpol(loop-1)=Jpol(loop-2); Jpol(loop-2)=Jpol2;
elseif ((Jpol2>=Jpol(2)) & (Jpol1>Jpol(2)) & (Jpol1>J(k))), kat=1;
 alfsb0=alf(loop-1); alfsb1=alfv(1); fark10=alfsb1-alfsb0;
 alf(loop)=alf(loop-1); alf(loop-1)=alfv(2); Jpol(loop)=Jpol(loop-1);
 Jpol(loop-1)=Jpol2; loop=loop+1;
 while Jpol1>J(k), alf(loop)=alfsb0+fark10*2^(-kat);
  xg=x0+alf(loop)*d; Jpol(loop)=feval(cost,xg,udn,zdnd); Jpol1=Jpol(loop); kat=kat+1;
 end;%%END OF while
elseif ((Jpol2>=Jpol(2)) & (Jpol1<Jpol(2))), yon=1; loop=loop+1;
 alf(loop)=alfv(1); alf(loop-1)=alf(loop-2);  alf(loop-2)=alfv(2);
 Jpol(loop)=Jpol1; Jpol(loop-1)=Jpol(loop-2); Jpol(loop-2)=Jpol2;
 while Jpol(loop)<=Jpol(loop-1), loop=loop+1;
  alf(loop)=alfix*2^(loop-3); xg=x0+alf(loop)*d; Jpol(loop)=feval(cost,xg,udn,zdnd);
  if ((Jpol(loop)>J(k)) & (Jpol(loop-1)>Jpol(loop-2))), alfs=alf; Jpols=Jpol;
   alf=[]; Jpol=[]; alf=alfs([1:loop-1]); Jpol=Jpol([1:loop-1]);
  elseif ((Jpol(loop)>J(k)) & (Jpol(loop-1)<=Jpol(loop-2))), kat=1;
   alfsb0=alf(loop-1); alfsb1=alf(loop); fark10=alfsb1-alfsb0;
   while Jpol(loop)>J(k), alf(loop)=alfsb0+fark10*2^(-kat);
   xg=x0+alf(loop)*d; Jpol(loop)=feval(cost,xg,udn,zdnd); kat=kat+1; end;%%END OF while
  end;%%END OF if-elseif
 end;%%END OF while
elseif ((Jpol1>Jpol(2)) & (Jpol2<=Jpol(2)) & (Jpol1<=J(k))), loop=loop+1; yon=-1;
 alf(loop)=alfv(1); alf(loop-1)=alf(loop-2);  alf(loop-2)=alfv(2);
 Jpol(loop)=Jpol1; Jpol(loop-1)=Jpol(loop-2); Jpol(loop-2)=Jpol2;
elseif ((Jpol1>Jpol(2)) & (Jpol2<=Jpol(2)) & (Jpol1>J(k))),
 alf(loop)=alf(loop-1); alf(loop-1)=alfv(2);
 Jpol(loop)=Jpol(loop-1); Jpol(loop-1)=Jpol2;
end;%%END OF if-elseif-elseif
%
else
 loop2=loop+1; alf(loop)=alfix*2^(2-loop2); xg=x0+alf(loop)*d;
 Jpol(loop)=feval(cost,xg,udn,zdnd);
 while Jpol(loop)>J(k), loop2=loop2+1; alf(loop)=alfix*2^(2-loop2);
  xg=x0+alf(loop)*d; Jpol(loop)=feval(cost,xg,udn,zdnd);
```

```
 end;%%END OF while
alfix=alf(loop);
%Yön Belirleme
 loop=loop+1; alfv=[alfix*2^(loop-2);alfix*2^(1-loop)];
 xg1=x0+alfv(1)*d; xg2=x0+alfv(2)*d;
 Jpol1=feval(cost,xg1,udn,zdnd); Jpol2=feval(cost,xg2,udn,zdnd);

if ((Jpol2>=Jpol(2)) & (Jpol1>Jpol(2)) & (Jpol1<=J(k))), loop=loop+1;
 alf(loop)=alfv(1); alf(loop-1)=alf(loop-2);  alf(loop-2)=alfv(2);
 Jpol(loop)=Jpol1; Jpol(loop-1)=Jpol(loop-2); Jpol(loop-2)=Jpol2;
elseif ((Jpol2>=Jpol(2)) & (Jpol1>Jpol(2)) & (Jpol1>J(k))), kat=1;
 alfsb0=alf(loop-1); alfsb1=alfv(1); fark10=alfsb1-alfsb0;
 alf(loop)=alf(loop-1); alf(loop-1)=alfv(2); Jpol(loop)=Jpol(loop-1);
 Jpol(loop-1)=Jpol2; loop=loop+1;
 while Jpol1>J(k), alf(loop)=alfsb0+fark10*2^(-kat);
  xg=x0+alf(loop)*d; Jpol(loop)=feval(cost,xg,udn,zdnd); Jpol1=Jpol(loop); kat=kat+1;
 end;%%END OF while
elseif ((Jpol2>=Jpol(2)) & (Jpol1<Jpol(2))), yon=1; loop=loop+1;
 alf(loop)=alfv(1); alf(loop-1)=alf(loop-2);  alf(loop-2)=alfv(2);
 Jpol(loop)=Jpol1; Jpol(loop-1)=Jpol(loop-2); Jpol(loop-2)=Jpol2;
 while Jpol(loop)<=Jpol(loop-1), loop=loop+1;
  alf(loop)=alfix*2^(loop-3); xg=x0+alf(loop)*d; Jpol(loop)=feval(cost,xg,udn,zdnd);
  if ((Jpol(loop)>J(k)) & (Jpol(loop-1)>Jpol(loop-2))), alfs=alf; Jpols=Jpol;
   alf=[]; Jpol=[]; alf=alfs([1:loop-1]); Jpol=Jpol([1:loop-1]);
  elseif ((Jpol(loop)>J(k)) & (Jpol(loop-1)<=Jpol(loop-2))), kat=1;
   alfsb0=alf(loop-1); alfsb1=alf(loop); fark10=alfsb1-alfsb0;
   while Jpol(loop)>J(k), alf(loop)=alfsb0+fark10*2^(-kat);
   xg=x0+alf(loop)*d; Jpol(loop)=feval(cost,xg,udn,zdnd); kat=kat+1; end;%%END OF while
  end;%%END OF if-elseif
 end;%%END OF while
elseif ((Jpol1>Jpol(2)) & (Jpol2<=Jpol(2)) & (Jpol1<=J(k))), loop=loop+1; yon=-1;
 alf(loop)=alfv(1); alf(loop-1)=alf(loop-2);  alf(loop-2)=alfv(2);
 Jpol(loop)=Jpol1; Jpol(loop-1)=Jpol(loop-2); Jpol(loop-2)=Jpol2;
elseif ((Jpol1>Jpol(2)) & (Jpol2<=Jpol(2)) & (Jpol1>J(k))),
 alf(loop)=alf(loop-1); alf(loop-1)=alfv(2);
 Jpol(loop)=Jpol(loop-1); Jpol(loop-1)=Jpol2;
end;%%END OF if-elseif-elseif

end;%%END OF if else (Jpol<=Jpol)

if alf(loop)/alf(2)>20000, [ao,bo]=min(Jpol); Jpolg=Jpol; alfg=alf;
 alf=alf([bo-1:bo+1]); Jpol=Jpol([bo-1:bo+1]); alfb=alf(1); [as,bs]=size(alf);
 sloop=bs;
 else alfb=0; sloop=loop; end;
alfinc=alfb:alf(2):alf(sloop);
Jinc=interp1(alf,Jpol,alfinc,'spline');
[aj,bj]=min(Jinc); alfa=alfinc(bj);

.....
% *** OPTİMİZASYON ALGORİTMASI İÇİN İLGİLİ SİSTEMİN ***
%  *** ÇIKIŢLARININ ÇÝZÝMÝ VE PARAMETRELERÝN KAYDI ***
function yciz_yuk(nu_ciz,iteras,zaman,cikis,cikisd,bekle)

% Bu fonksiyon sadece çizim ve parametre kaydeder
% nu_ciz: Kaç iterasyonda bir çizim ve parametre kaydı yapılacağını verir
% iteras: O anki iterasyon sayısı
% zaman: Dinamik sistemlerdeki zaman vektörü
```

```
% cikis: Dinamik sistemlerin çıkışı
% cikisd: Dinamik sistemler için istenen çıkış
% bekle: Çizim sonunda 1 ise bekler

%Yaşar BECERİKLİ- 20:10 02.11.1997

global wdn; global pdn; global qdn; global Tdn; global bdn; global gamadn; global betadn;

if floor(iteras/nu_ciz)*nu_ciz==iteras,
hold on; clf; plot(zaman,cikisd,'k',zaman,cikis,':k'); title('Process ve DSA Çıkışı'); xlabel('zaman');
ylabel('z; zd');
pause2(0.1); if bekle==1, pause; end; hold off; end;

if floor(iteras/nu_ciz)*nu_ciz==iteras,
save wdn wdn; save pdn pdn; save qdn qdn; save bdn bdn; save Tdn Tdn; save gamadn gamadn;
save betadn betadn; end;

.....
% *** DSA' LARI İÇİN ÖLÇÜT VE DİĞER ÇIKIŞLARIN BULUNMASI ***
function [E,t,xdn,ysig,zdn]=ydncost(pardn,ugir,zcikd)

% DSA için verilen değerlere göre ölçüt, durum ve çıkışları hesaplar
% pardn: DSA parametre vektörü
% ugir: DSA giriş fonksiyonu
% zcikd: İstenen çıkış değeri
% E: Ölçüt
% t: Zaman vektörü
% xdn: Dinamik nöron durum vektörü
% ysig: DSA aktivasyon fonksiyonu çıkışı
% zdn: DSA çıkışı

%Yaşar BECERİKLİ- 21:40 02.11.1997

global udn; global wdn; global pdn; global qdn; global Tdn; global bdn;
global gamadn; global betadn;
global dns; global gs; global cs;
global t0; global tf; global hp; global xdn0;
global err; global xdn; global ysig; global zdnd; global crp;

udn=ugir; zdnd=zcikd;
sw=dns^2; sp=dns*gs; sq=cs*dns; sb=dns; sT=dns; sgama=dns; sbeta=dns;

if crp(1)==1, tg=1:dns:sw;
 for k=1:dns, wdn(k,:)=pardn([tg(k):tg(k)+dns-1])'; end; ss=sw; end;
if crp(2)==1, tg=ss:gs:ss+sp-1;
 for k=1:dns, pdn(k,:)=pardn([tg(k)+1:tg(k)+gs])'; end; ss=ss+sp; end;
if crp(3)==1, tg=ss:dns:ss+sq-1;
 for k=1:cs,  qdn(k,:)=pardn([tg(k)+1:tg(k)+dns])'; end; ss=ss+sq; end;
if crp(4)==1, tg=ss:sb:ss+sb;
        bdn=pardn([tg(1)+1:tg(2)]); ss=ss+sb; end;
if crp(5)==1, tg=ss:sT:ss+sT;
        Tdn=diag(pardn([tg(1)+1:tg(2)]),0); ss=ss+sT; end;
if crp(6)==1, tg=ss:sgama:ss+sgama;
        gamadn=pardn([tg(1)+1:tg(2)]); ss=ss+sgama; end;
if crp(7)==1, tg=ss:sbeta:ss+sbeta;
        betadn=pardn([tg(1)+1:tg(2)]); end;
Tt=diag(Tdn)>=.1; Ttn=diag(Tdn)<.1;
```

```
Tdn=diag(Tt.*diag(Tdn)+Ttn.*(.1*ones(dns,1)),0);
neggama=gamadn>0; zerogama=neggama==0;
gamadn=gamadn.*neggama+zerogama*1e-3;
[t,xdn]=yrk5('yxdn',t0,tf,hp,xdn0);
ysig=ysigmoid(xdn,gamadn,betadn); zdn=qdn*ysig; err=zdn-zdnd;
[aa,bb]=size(zdn);
if aa~=1, F=sum(err.*err)/2; else F=(err.*err)/2; end;
E=yinteg(hp,F);

.....
% *** DSA' LARI ADJOINT YÖNTEMİNE DAYALI GRADYAN HESABI ***
function dEpar=ydngrad(pardn,ugir)

% DSA için verilen değerlere göre Adjoint yöntemine göre ÖLÇÜT gradyan
% pardn: DSA parametre vektörü
% ugir: DSA giriş fonksiyonu
% dEpar: Ölçütün parametrelerine göre gradyanı

%Yaşar BECERİKLİ- 22:26 02.11.1997

global udn; global wdn; global pdn; global qdn; global Tdn; global bdn;
global gamadn; global betadn;
global dns; global gs; global cs;
global t0; global tf; global hp; global xdn0;
global err; global xdn; global ysig; global crp;

udn=ugir;
sw=dns^2; sp=dns*gs; sq=cs*dns; sb=dns; sT=dns; sgama=dns; sbeta=dns;

if crp(1)==1, tg=1:dns:sw;
 for k=1:dns, wdn(k,:)=pardn([tg(k):tg(k)+dns-1])'; end; ss=sw; end;
if crp(2)==1, tg=ss:gs:ss+sp-1;
 for k=1:dns, pdn(k,:)=pardn([tg(k)+1:tg(k)+gs])'; end; ss=ss+sp; end;
if crp(3)==1, tg=ss:dns:ss+sq-1;
 for k=1:cs,  qdn(k,:)=pardn([tg(k)+1:tg(k)+dns])'; end; ss=ss+sq; end;
if crp(4)==1, tg=ss:sb:ss+sb;
        bdn=pardn([tg(1)+1:tg(2)]); ss=ss+sb; end;
if crp(5)==1, tg=ss:sT:ss+sT;
        Tdn=diag(pardn([tg(1)+1:tg(2)]),0); ss=ss+sT; end;
if crp(6)==1, tg=ss:sgama:ss+sgama;
        gamadn=pardn([tg(1)+1:tg(2)]); ss=ss+sgama; end;
if crp(7)==1, tg=ss:sbeta:ss+sbeta;
        betadn=pardn([tg(1)+1:tg(2)]); end;

Tt=diag(Tdn)>=.1; Ttn=diag(Tdn)<.1;
Tdn=diag(Tt.*diag(Tdn)+Ttn.*(.2*ones(dns,1)),0);
neggama=gamadn>0; zerogama=neggama==0;
gamadn=gamadn.*neggama+zerogama*1e-3;


lmdnf=zeros(dns,1);
%ADJOINT (LAMDA) DIFF.DENK.ÇÖZ.
[t,lmdn]=yrk5('yxdnlam',t0,tf,-hp,lmdnf);
[a,b]=size(lmdn);

%PARAMETRE GRADYANLARI
for m=1:dns,
```

```
 for n=1:dns,
dEwdn(m,n)=yinteg(hp,lmdn(m,:).*ysig(n,:))/Tdn(m,m);
if n==1,
for np=1:gs, dEpdn(m,np)=yinteg(hp,lmdn(m,:).*udn(np,:))/Tdn(m,m);
 end; end;
if m==1;
for nq=1:cs, dEqdn(nq,n)=yinteg(hp,err(nq,:).*ysig(n,:));
 end; end;
end; end;

Ibir=ones(a,b);
dEbdn=yinteg(hp,lmdn)./diag(Tdn);
dETdn=diag(-yinteg(hp,lmdn.*yxdn(-9999,xdn))./diag(Tdn.^2),0);
dEgamadn=yinteg(hp,((Tdn\wdn)'*lmdn).*(xdn.*(ysig.*(Ibir-ysig))));
dEbetadn=yinteg(hp,((Tdn\wdn)'*lmdn).*(ysig.*(Ibir-ysig)));

%TUM PARAMETRE GRADYAN VEKTORU
dEwdn=dEwdn'; dEpdn=dEpdn'; dEqdn=dEqdn';
dEpardn0=[crp(1)*dEwdn(:);crp(2)*dEpdn(:);crp(3)*dEqdn(:);crp(4)*dEbdn;crp(5)*diag(dETdn);cr
p(6)*dEgamadn;crp(7)*dEbetadn];
dEwdn=dEwdn'; dEpdn=dEpdn'; dEqdn=dEqdn';

dEpardn=[];
if crp(1)==1, dEpardn=dEpardn0([1:sw]); end; ss=sw;
if crp(2)==1, dEpardn=[dEpardn;dEpardn0([ss+1:ss+sp])]; end; ss=ss+sp;
if crp(3)==1, dEpardn=[dEpardn;dEpardn0([ss+1:ss+sq])]; end; ss=ss+sq;
if crp(4)==1, dEpardn=[dEpardn;dEpardn0([ss+1:ss+sb])]; end; ss=ss+sb;
if crp(5)==1, dEpardn=[dEpardn;dEpardn0([ss+1:ss+sT])]; end; ss=ss+sT;
if crp(6)==1, dEpardn=[dEpardn;dEpardn0([ss+1:ss+sgama])]; end; ss=ss+sgama;
if crp(7)==1, dEpardn=[dEpardn;dEpardn0([ss+1:ss+sbeta])]; end;
dEpar=dEpardn;

.....
% *** DSA İÇİN ADJOINT DİFF. DENKLEM SİSTEMİ ***
function xdot=yxdnlam(t,xdnl)

% DSA 'da GRADYAN hesabı için Adjoint sistemi
% t: zaman, xdnl: DSA için Adjoint durum değişkeni
% xdot: Dönen Adjoint Diff. Denklem Vektörü

% Yaşar BECERİKLİ- 22:36 05.11.1997

global err; global wdn; global pdn; global qdn; global Tdn; global bdn
global gamadn; global betadn;
global udn; global hp;
global Adn; global xdn;

[a,b]=size(xdnl); it=t/hp+1;
gss=ysigmoid(xdn(:,it),gamadn,betadn);
gr=gamadn*(gss*(1-gss)); Adr=wdn*diag(gr,0)-eye(a); Adn=Tdn\Adr;
xdot=-Adn'*xdnl-(qdn'*err(:,it)).*gr;

.....
% *** DSA 'DA ZAMAN SABİTİ VE SİGMOİD LİNEER OLMAMA PARAMETRESİ İÇİN ***
%                                   *** SINIRLAMA ***
function x=ytrsh_p(pardn)
```

```
% Zaman sabiti .1 'nin altına diğer parametre 1e-3'ün altına indirilmiyor
% pardn: DSA 'a ait parametre vektörü
% x: Sınırlama yapılmış parametre vektörü

% Yaşar BECERİKLİ- 01:45 03.11.1997

global wdn; global pdn; global qdn; global Tdn; global bdn; global gamadn; global betadn;
global dns; global gs; global cs; global crp;

sw=dns^2; sp=dns*gs; sq=cs*dns; sb=dns; sT=dns; sgama=dns; sbeta=dns;

if crp(1)==1, tg=1:dns:sw;
 for k=1:dns, wdn(k,:)=pardn([tg(k):tg(k)+dns-1])'; end; ss=sw; end;
if crp(2)==1, tg=ss:gs:ss+sp-1;
 for k=1:dns, pdn(k,:)=pardn([tg(k)+1:tg(k)+gs])'; end; ss=ss+sp; end;
if crp(3)==1, tg=ss:dns:ss+sq-1;
 for k=1:cs,  qdn(k,:)=pardn([tg(k)+1:tg(k)+dns])'; end; ss=ss+sq; end;
if crp(4)==1, tg=ss:sb:ss+sb;
        bdn=pardn([tg(1)+1:tg(2)]); ss=ss+sb; end;
if crp(5)==1, tg=ss:sT:ss+sT;
        Tdn=diag(pardn([tg(1)+1:tg(2)]),0); ss=ss+sT; end;
if crp(6)==1, tg=ss:sgama:ss+sgama;
        gamadn=pardn([tg(1)+1:tg(2)]); ss=ss+sgama; end;
if crp(7)==1, tg=ss:sbeta:ss+sbeta;
        betadn=pardn([tg(1)+1:tg(2)]); end;

Tt=diag(Tdn)>=.1; Ttn=diag(Tdn)<.1;
Tdn=diag(Tt.*diag(Tdn)+Ttn.*(.1*ones(dns,1)),0);
neggama=gamadn>0; posgama=gamadn<0; posgama=1e-3*posgama;
gamadn=gamadn.*neggama; gamadn=gamadn+posgama;

wdn=wdn'; pdn=pdn'; qdn=qdn';
wdnit=wdn(:); wdn=wdn';
pdnit=pdn(:); pdn=pdn';
qdnit=qdn(:); qdn=qdn';
bdnit=bdn;
Tdnit=diag(Tdn);
gamadnit=gamadn;
betadnit=betadn;

pardn0=[crp(1)*wdnit;crp(2)*pdnit;crp(3)*qdnit;crp(4)*bdnit;...
   crp(5)*Tdnit;crp(6)*gamadnit;crp(7)*betadnit];
pardn=[];
if crp(1)==1, pardn=pardn0([1:sw]); end; ss=sw;
if crp(2)==1, pardn=[pardn;pardn0([ss+1:ss+sp])]; end; ss=ss+sp;
if crp(3)==1, pardn=[pardn;pardn0([ss+1:ss+sq])]; end; ss=ss+sq;
if crp(4)==1, pardn=[pardn;pardn0([ss+1:ss+sb])]; end; ss=ss+sb;
if crp(5)==1, pardn=[pardn;pardn0([ss+1:ss+sT])]; end; ss=ss+sT;
if crp(6)==1, pardn=[pardn;pardn0([ss+1:ss+sgama])]; end; ss=ss+sgama;
if crp(7)==1, pardn=[pardn;pardn0([ss+1:ss+sbeta])]; end;
x=pardn;

.....
% *** ZAMANDA İLERİ VE GERİ LİNEER OLMAYAN DİFF. DENKLEM ÇÖZÜMÜ ***
function [t,x]=yrk5(ypfun,t0,tf,h,x0)

% Butcher'ın 5.Dereceden Sabit Adımlı Runga-Kutta Metodu ile Nümerik,
```

```
% Zamanda İleri ve Geri Lineer Olmayan Adi Diferansiyel Denklem Çözümü
% ypfun: Diferansiyel Denklem Fonksiyonu (bir .m dosyası)
% t0: Başlangıç Zamanı; tf: Son Zaman
% h: İntegrasyon Aralığı, Pozitif veya Negatif Olabilir
% x0: Durumların Başlangıç veya Son Şartları
% t: Fonksiyondan Dönen Zaman Vektörü
% x: Fonksiyondan Dönen İntegre Edilen Durum Vektörü

%  Yaşar BECERİKLİ- 22:49 06.07.1997

[a,b]=size(x0);
if h>0, xx=x0; indx=1; t(indx)=t0;
for tt=t0:h:tf-h,
   k1=h*feval(ypfun,tt,xx);  k2=h*feval(ypfun,tt+h/4,xx+k1/4);
   k3=h*feval(ypfun,tt+h/4,xx+k1/8+k2/8);  k4=h*feval(ypfun,tt+h/2,xx-k2/2+k3);
   k5=h*feval(ypfun,tt+3*h/4,xx+3*k1/16+9*k4/16);
   k6=h*feval(ypfun,tt+h,xx-3*k1/7+2*k2/7+12*k3/7-12*k4/7+8*k5/7);
   xx=xx+(7*k1+32*k3+12*k4+32*k5+7*k6)/90; indx=indx+1; t(indx)=tt+h;
   for m=1:a, xn(m,indx)=xx(m); end; end; x=xn; t=t'; end;
if h<0, xx=x0; indx=((tf-t0)/(-h))+1; t(indx)=tf;
for tt=tf:h:t0-h,  indx=indx-1;
   k1=h*feval(ypfun,tt,xx);  k2=h*feval(ypfun,tt+h/4,xx+k1/4);
   k3=h*feval(ypfun,tt+h/4,xx+k1/8+k2/8);  k4=h*feval(ypfun,tt+h/2,xx-k2/2+k3);
   k5=h*feval(ypfun,tt+3*h/4,xx+3*k1/16+9*k4/16);
   k6=h*feval(ypfun,tt+h,xx-3*k1/7+2*k2/7+12*k3/7-12*k4/7+8*k5/7);
   xx=xx+(7*k1+32*k3+12*k4+32*k5+7*k6)/90;  t(indx)=tt+h;
   for m=1:a, xn(m,indx)=xx(m); end; end; x=xn; t=t'; end;

.....
%  *** SİMPSON 1/3 COMPOSITE NÜMERİK İNTEGRASYON'U ***
function xi=yinteg(dt,x)

%  Nümerik İntegrasyon Algoritması (Simpson'nun(1/3) Composite Algoritması)
%  Burada dt integrasyon aralığı, x ise integrali alınacak
%  fonksiyonun vektör ifadesidir
%  xi: İntegral Sonucu

%  Yaşar BECERİKLİ- 16:05 06.07.1997

[a,b]=size(x);
vek(1:2:b)=2*ones(1,(b)/2); vek(2:2:b-1)=4*ones(1,(b-1)/2);
vek(1)=1; vek(b)=1; xi=dt*vek*x'/3; xi=xi';

.....
% *** 2 PARAMETRELİ SIGMOID FONKSİYONU ***
function sig=ysigmoid(xsig,gama,beta)

% xsig: Sigmoid fonksiyonunun giriş değişkeni
% gama:      "        "       lineer olmama parametresi
% beta:      "        "       bias parametresi
% sig: Sigmoidin giriş ve parametrelerine göre çıkışı

% Yaşar BECERİKLİ- 21:09 05.01.1997

[a,b]=size(xsig);
gamaa=diag(gama); betaa=beta*ones(1,b); sig=1/(ones(a,b)+exp(-(gamaa*xsig+betaa)));
```

ÖZGEÇMİŞ

Yaşar Becerikli, 19 Aralık 1969'da Gemlik'te doğdu. İlk ve orta öğrenimini Gemlik'te tamamladı. 1991 yılında Yıldız Üniversitesi Kocaeli Mühendislik Fakültesi Elektronik ve Haberleşme Mühendisliği Bölümü'nden mezun oldu. 1994 yılında İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü Elektronik-Haberleşme Anabilim dalında Yüksek Lisans'ını tamamladı. Halen Sakarya Üniversitesi Elek-Elektronik Müh. Bölümü Haberleşme Anabilim dalında Araş. Görevlisi olarak çalışmaktadır. Yazar evli ve bir çocuk babasıdır.