
GAZİ ÜNİVERSİTESİ
Gazi
1926

İNSAN SANTRİFÜJ SİSTEMLER İÇİN YAPAY SİNİR AĞLARI TABANLI AÇISAL HIZIN HESAPLANMASI

Yakup CENGİZ

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

EKİM 2017

Yakup CENGİZ tarafından hazırlanan "İNSAN SANTRİFÜJ SİSTEMLER İÇİN YAPAY SİNİR AĞLARI TABANLI AÇISAL HIZIN HESAPLANMASI" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ ile Gazi Üniversitesi Bilgisayar Mühendisliği Anabilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Danışman: Prof. Dr. Şeref SAĞIROĞLU

Bilgisayar Mühendisliği Anabilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ...……

Başkan: Doç. Dr. Refik SAMET

Bilgisayar Mühendisliği Anabilim Dalı, Ankara Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ……...

Üye: Doç. Dr. Suat ÖZDEMİR

Bilgisayar Mühendisliği Anabilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ……...

Tez Savunma Tarihi: 31/10/2017

Jüri tarafından kabul edilen bu tezin Yüksek Lisans Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

……..…….

Prof. Dr. Hadi GÖKÇEN

Fen Bilimleri Enstitüsü Müdürü

ETİK BEYAN

Gazi Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Yakup CENGİZ

31/10/2017

İNSAN SANTRİFÜJ SİSTEMLER İÇİN YAPAY SİNİR AĞLARI TABANLI AÇISAL HIZIN HESAPLANMASI

(Yüksek Lisans Tezi)

Yakup CENGİZ

GAZİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Ekim 2017

ÖZET

Yüksek performanslı askeri jetlerde, ani ve keskin dönüşler, acil durum manevraları, ani yükseliş ve alçalış neticesinde maruz kalınan yüksek merkezkaç kuvvetler, pilotun fizyolojik durumunu ve performansını önemli derecede etkilemektedir. Yüksek G kuvvetlerine maruz kalan pilotların hayatlarını korumak ve olası kaza sonucunda oluşabilecek maddi zararları engellemek için İnsan Santrifüj Sistemler geliştirilmiştir. Pilot eğitimlerinde kullanılan gerçek bir uçuş esnasında karşılaşılan kuvvetleri simüle etmek için hazırlanan G profillerinin başarılı bir şekilde çalıştırılması için açısal hızın hesaplanması gerekir. İnsan Santrifüj Sistem hareketi için gerekli olan açısal hız, azalan G komutları için hesaplanamamaktadır. Bu çalışmada, açısal hızın hesaplanması için kullanılan, dairesel hareket denklemlerinden elde edilen diferansiyel denklem, varlık ve teklik açısından incelenmiştir. Daha önce yapılan çalışmalarda kullanılan Runge-Kutta tabanlı nümerik yöntemler yerine, açısal hız dördüncü dereceden bir polinom denklem kurularak çözülmeye çalışılmıştır. Bu çalışmada, açık çevrim G profilleri için Yapay Sinir ağları tabanlı bir matematiksel model önerilmiştir. Daha önce yapılan çalışmalarda azalan G komutlarına karşılık açısal hızın hesaplanması için önerilen algoritmanın eksiklikleri belirtilmiş ve eksiklerin giderilmesi için Radyal Tabanlı Fonksiyon Ağ modeli önerilmiştir. Radyal Tabanlı Fonksiyon Ağ modelini eğitmek ve test etmek amacıyla python dili kullanılarak basit bir G profil editör yazılmıştır. Çok Katmanlı Yapay Sinir Ağın eğitilmesi, test edilmesi ve açısal hızın hesaplanmasında kullanılan diğer yöntemlerle karşılaştırılması amacıyla C++ dili kullanılarak bir uygulama yazılmıştır. Elde edilen sonuçlar, açısal hız değerinin Yapay Sinir Ağ modeli kullanılarak açık çevrim G profilleri için çözülebileceğini göstermiştir.

Bilim Kodu	:	92431
Anahtar kelimeler	:	İnsan Santrifüj Sistemler, Diferansiyel Denklemler, Yapay Sinir Ağlar, Euler, Runge Kutta, Radyal Tabanlı Fonksiyon Ağlar
Sayfa Adedi	:	95
Danışman	:	Prof. Dr. Şeref SAĞIROĞLU

CALCULATING ANGULAR VELOCITY FOR HUMAN CENTRIFUGE SYSTEMS BASED ON ARTIFICIAL NEURAL NETWORKS

(M. Sc. Thesis)

Yakup CENGİZ

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

October 2017

ABSTRACT

In high-performance military aircraft, high centrifugal forces result in sharp turns and such emergency maneuvers significantly affect the pilot's physiological condition and performance. Human Centrifuge Systems have been developed to protect pilots exposed to high G forces and prevent material loss that result from possible accidents. To achieve G profiles prepared to generate artificial forces experienced during a real flight, it is necessary to calculate angular velocity. However, angular velocity, which is necessary for movement of Human Centrifuge System, can not be calculated for decreasing G commands. In this study, the differential equation obtained from the circular motion equations to calculate angular velocity is examined in terms of existence and uniqueness. Instead of using the Runge-Kutta based numerical methods used in the previous studies, a forth order polynomial equation established to solve angular velocity. In this study, a mathematical model based on Artificial Neural Networks for open loop G profiles is proposed. Lack of the algorithm used in the previous studies for the calculation of angular velocity is pointed out and Radial Based Function Network model is proposed to eliminate the deficiencies of the algorithm. A simple G profile editor was written using the python language to train and test Radial Basis Function Network used in this study. An application has been written using the C ++ language for the purpose of training, testing the Multi Layer Artificial Neural Network and comparing the model with other methods used in the calculation of angular velocity. The results obtained shows that angular velocity can be calculated by using an Artificial Neural Network model for open loop G profiles.

Science Code	:	92431
Key Words	:	Human Centrifuge, Differential Equation, Euler, classic Runge Kutta, Artificial Neural Network, Curve Fitting, Radial Basis Function Network
Page Number	:	95
Supervisor	:	Prof. Dr. Şeref SAĞIROĞLU

TEŞEKKÜR

Yüksek lisans eğitimi boyunca bilgi ve tecrübelerini bana aktararak beni yönlendiren değerli hocam Sayın Prof. Dr. Şeref SAĞIROĞLU hocama destek ve yardımları için teşekkürlerimi sunarım.

Tüm öğrenim hayatım boyunca her türlü desteklerini hiçbir zaman esirgemeyerek bugünlere gelmemi sağlayan ve her zaman yanımda olan ailem ile çalışmalarım boyunca her zaman yanımda olan, beni her koşulda destekleyen ve motive eden hayat arkadaşım Melike CENGİZ'e, oğlum Ömer Kaan'a ve kızım Doğa'ya teşekkür ederim.

İÇİNDEKİLER

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış olan simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklamalar**
$\boldsymbol{\omega}$	Açısal Hız
$\boldsymbol{\alpha}$	Açısal ivme
$\boldsymbol{g}$	Yerçekimi ivmesi
$\boldsymbol{G}$	Bileşke G kuvveti
$+\boldsymbol{G_x}$	Göğüsten sırta doğru etki eden *G* kuvveti
$-\boldsymbol{G_x}$	Sırttan göğüse doğru etki eden *G* kuvveti
$\boldsymbol{G_y}$	Omuzdan omuza etkiyen *G* kuvveti
$\boldsymbol{G_z}$	Vücudun dikey eksende maruz kaldığı yer çekimi kuvveti
$+\boldsymbol{G_z}$	Baştan ayaklara doğru etkiyen *G* kuvveti
$-\boldsymbol{G_z}$	Ayaklardan başa doğru etkiyen *G* kuvveti
$\boldsymbol{\alpha}$	Öğrenme katsayısı
$\boldsymbol{\mu}$	Momentum katsayısı
$\boldsymbol{sn}$	Saniye
$\boldsymbol{rad}$	Radyan
$\boldsymbol{m}$	Metre
$\boldsymbol{\varphi}$	Radyal Tabanlı Fonksiyon

Kısaltmalar	**Açıklamalar**
ADD	Adi Diferansiyel Denklem
AGSM	Anti-G Straining Manevrası
cRK	4. dereceden klasik Runge-Kutta
Closed Loop	Kapalı Döngü
ÇKA	Çok Katlı Algılayıcılar
ÇKYSA	Çok Katmanlı Yapay Sinir Ağı
İBYSA	İleri Beslemeli Yapay Sinir Ağı

Kısaltmalar	Açıklamalar
İSS	İnsan Santrifüj Sistemler
EKG	Elektrokardiogram
FF	İleri Beslemeli
GBYSA	Geri Beslemeli Yapay Sinir Ağı
GOR	Gradual Onset Run - Kademeli Artan *G* Eğitimi
GYA	Geri Yayılım Algoritması
G-LOC	G Kuvvetlerine Bağlı Bilinç Kaybı
GD	Gradyan İniş
MLP	Multi Layer Perseptron
Offset	1 saniyedeki *G* azaltma miktarı
Onset	1 saniyedeki *G* artış miktarı
Open Loop	Açık Döngü
Pitch eksen	Uçağın burun aşağı yukarı hareket ekseni, Yunuslama
Planetary eksen	Sistemin dönme yörüngesi
ROR	Ani Artan *G* Eğitimi (Rapid Onset Run)
Roll eksen	Uçağın sağa sola yatış hareket ekseni, Yuvarlanma
RTFA	Radyal Tabanlı Fonksiyon Ağı
RTF	Radyal Tabanlı Fonksiyon
SACM	Benzetilmiş Hava Muharebesi
YSA	Yapay Sinir Ağı
YZ	Yapay Zeka

1 . GİRİŞ

Yüksek performanslı askeri jetlerde, ani ve keskin dönüşler, acil durum manevraları, ani yükseliş ve alçalış neticesinde maruz kalınan yüksek merkezkaç kuvvetler, pilotun fizyolojik durumunu ve performansını önemli derecede etkiler. Yüksek *G* kuvvetlerine maruz kalan pilotların hayatlarını korumak ve olası kaza sonucunda oluşabilecek maddi zararları engellemek için İnsan Santrifüj Sistemler (İSS) geliştirilmiştir [6]. Bu sistemler yardımıyla, pilotlar çeşitli eğitimlerden geçerek zorlu uçuşlara hazır hale gelirler. Uygulanan eğitimler sonucunda, pilotların *G* toleransı artmakta ve G-LOC (G Kuvvetlerine Bağlı Bilinç Kaybı) olma riskleri azalmaktadır [4].

İSS'ler gerçek bir uçuş esnasında karşılaşılan *G* kuvvetlerini pilotlara en gerçekçi şekilde uygulanmasını sağlayan dinamik uçuş simülatörleridir [1, 2]. Sistem yunuslama (pitch), yuvarlanma (roll) ve planetary olmak üzere üç eksene sahiptir. Ana hareketi sağlayan eksen planetary eksendir. İSS'lerde verilen eğitimin temel amacı G_z (vücudun dikey eksende maruz kaldığı yer çekimi kuvveti) kuvvetini eğitim gören pilota uygulamaktır. Gerçek uçuş esnasında pilotların karşılaştıkları en büyük problem maruz kaldıkları pozitif G_z (baştan ayaklara doğru etkiyen *G* kuvveti) kuvvetidir. Pilotlara yaptıkları manevralar sonucunda sadece G_z kuvveti uygulayabilmek için yunuslama (pitch) ve yuvarlanma (roll) eksenleri uygun pozisyonlara hareket ettirilerek diğer kuvvetlerden dolayı oluşabilecek olumsuz etkiler olabildiğince yok edilmeye çalışılır [5]. İSS'lerde maruz kalınan *G* kuvveti yer çekimi ivmesinin ($g = 9,8 m/sn^2$) katları şekilde ifade edilir. 2 *G*'lik bir ivmeden bahsederken aslında $g = 19,6 m/sn^2$ değerinde bir ivmelenmeden söz edilir. Birim zamanda değişen *G* kuvveti, negatif yönlü ise sistemde uygulanmak istenen kuvvet, azalan *G* komutu; pozitif yönde olan kuvvet, artan *G* komutu olarak adlandırılır. Bu sistemlerde kullanılan onset, offset kavramları sırasıyla artan ve azalan *G* komutları için birim zamandaki değişimi ifade etmektedir. Uygulanmak istenen *G* kuvveti daha önce hazırlanan bir veriden (açık çevrim *G* profil) ya da çalışma zamanında hesaplanarak (kapalı çevrim *G* profil) belirlenebilir.

İSS yardımıyla uygulanmak istenen *G* kuvvetini yerine getirebilmek için hesaplanan açısal hız değeri planetary eksenin ne kadar hızlı döndürüleceğini belirlemekte kullanılmaktadır. Açısal hız bir objenin birim zamanda dairesel bir yörüngede açısal olarak yer değiştirme miktarına

verilen isimdir. Açısal hız değeri doğrusal olmayan birinci mertebeden bir diferansiyel denklem yardımıyla hesaplanır; ancak diferansiyel denklem azalan *G* komutları için çözüm sunamamaktadır. Vidakovic ve diğerleri tarafından yapılan iki ayrı çalışmada bu durum ele alınmış; ancak iyi bir sonuç elde edilememiştir. Vidakovic ve diğerleri yaptıkları çalışmada klasik Runge Kutta (cRK) yönteminin onset komutlarda en iyi sonuçları verdiğini önermişlerdir [1, 2]. Azalan *G* komutları için ihtiyaç duyulan açısal hızın hesaplanması için bir algoritma önermişlerdir. Önerilen algoritma, ancak iki komut arasındaki zamanın sabit olması garanti edildiği durumlarda ve uygulanacak komutların önceden bilinebilmesi yani açık çevrim profiller için doğru çalışmaktadır. Gerçek zamanlı bir çözüm sunamaması, ihtiyaç duyulan hesaplama süresi ve hafıza miktarı açısından değerlendirildiğinde, bu yöntem dezavantajları olan bir yöntemdir. Bu sistemler hakkında yayınlanan bir patentte, Crosbie ve Colombo, açısal hız değerinin çözülememesinden dolayı, teğetsel ivmenin ihmal edilmesiyle elde edilen açısal hız denklemini kullanmıştır [15]. Teğetsel ivmenin ihmal edilmesi ile elde edilen çözüm, yunuslama (pitch) eksen için belirlenen ivme değerinde artışa neden olmakla birlikte sistem ve eğitim gören pilot üzerinde olumsuz etkilere neden olmaktadır.

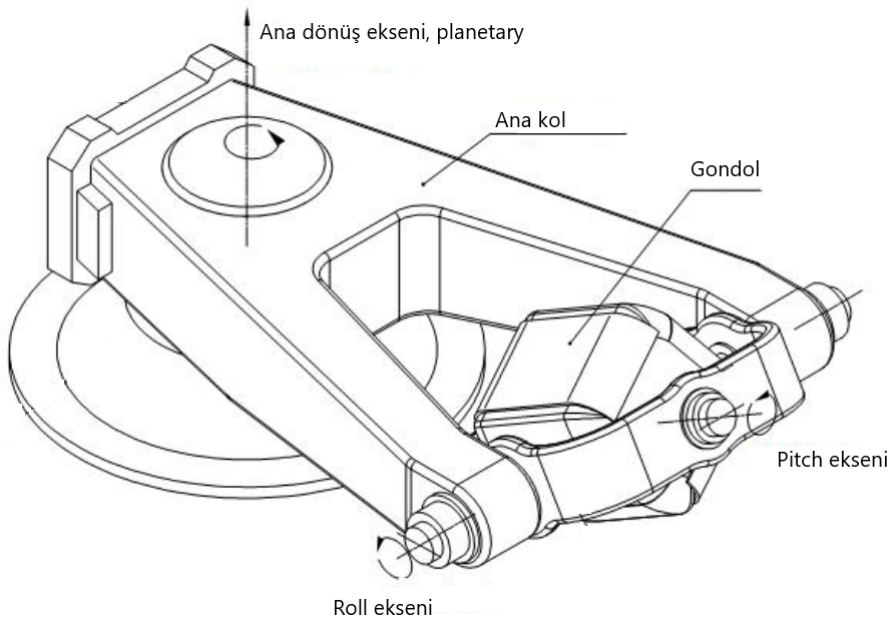
İSS'lerde verilen eğitimin başarısını belirleyen en önemli parametre uygulanmak istenen kuvvetin hangi oranda başarılı bir şekilde uygulandığıdır. Açısal hızın hesaplanması bu bakımdan bu sistemler açısından çok önemlidir. Bu çalışma açısal hız komutunun hesaplanabilmesi için yapılan araştırmalar ve sonuçları içermektedir. Artan *G* komutlarının cRK yöntemi ile yeterince iyi bir şekilde çözüldüğü daha önce yapılan çalışmalarda belirtildiği gibi bu çalışmada da test edilerek görülmüştür. Bu çalışmada ayrıca *G* komutlarına karşılık açısal hızın çözümü için her bir *G* komutuna karşılık katsayıları farklı olan polinom denklemler oluşturularak çözüm aranmıştır. Açısal hız, polinom denkemin köklerin bulunması yardımıyla hesaplanmıştır. Her iki yöntemin de azalan *G* komutları için açısal hızı hesaplayamadığı görülmüş, bundan dolayı bu çalışma azalan *G* komutları için açısal hızın hesaplanması konusuna odaklanmıştır. Bu çalışmanın temel amacı ve kapsamı uzun kollu İSS'lerde kullanılmak üzere açık çevrim *G* profil komutlarına karşılık açısal hız hesaplaması için bir matematiksel model bulmaktır. Önceden bu konuda yapılan çalışmalar incelenmiş; bu yöntemlerin avantaj ve dezavantajları belirlendikten sonra YSA tabanlı modeller açısal hızın hesaplanması amacı ile oluşturulmuştur.

Bölüm 2'de İSS'lerin yapısı ve çalışma prensibi ile alakalı bilgiler verilmiştir. Bölüm

3’te açısal hızın hesaplanmasında kullanılan YSA tanımı, tarihçesi, yapısı, eğitilmesi için kullanılan algoritmalar konusunda bilgiler verilmiştir. Açısal hız hesaplamasında kullanılan yöntemler Bölüm 4’de anlatılmıştır. Bölüm 5’de bu çalışmada bahsedilen yöntemleri test edebilmek için hazırlanan uygulama ve yapılan çalışmalar sonucu elde edilen deneysel sonuçlar gösterilmiştir. Bölüm 6’da bu çalışmada elde edilen sonuçlar ve ileride yapılabilecek çalışmalar için öneriler anlatılmaktadır.

2 . İNSAN SANTRİFÜJ SİSTEMLER (İSS)

Yüksek manevra kabiliyetine sahip uçakların kullanılmaya başlanması ile birlikte pilotlar üzerinde etki eden kuvvetleri incelemek üzere bir araştırma alanı açılmıştır. Bu araştırmaları yapabilmek için geliştirilen Şekil 2.1'de gösterilen üç serbestlik derecesine sahip İnsan Santrifüj Sistem(İSS) adı verilen sistemler kullanılmaktadır [1, 2].



Şekil 2.1. İnsan santrüj sistemi

2.1. Kullanım Alanları ve Tarihçe

G eğitimlerinde kullanılan profillerin çeşitleri ve ne tür eğitimlerde kullanıldığını Dan ve diğerleri yaptıkları çalışmada [6] profillerin amaçları ve önemini vurgulayarak anlatmışlardır. Bu çalışmada kullanılan profillerin NATO STANAG 3827 AMD standardına uygun olduğu ifade edilmiştir. İSS'ler bir uçuş senaryosunu olabildiğince en gerçekçi şekilde uygulanabilmesini sağlamaktadırlar. Bu tür eğitimler pilotlara uçuş esnasında karşılaşabilecekleri zorlukları önceden gösterebilmesi açısından çok önemlidir [35]. Cammarota yaptığı çalışmada, gerçek bir uçuşun İSS'de nasıl gerçekleştirildiğini anlatmıştır [12]. Lin ve diğerleri yaptıkları çalışmada İSS'de uygulanan eğitimlerden ve eğitimin pilotlar üzerindeki olumlu etkilerinden bahsetmektedir [34].

İSS'lerin yapısı ve hareketini sağlayan bir kontrol algoritması Vladimir M. Kvrgic tarafından hazırlanan bir çalışma ile sunulmuştur [5]. Tsai ve Shih yaptıkları çalışmada, G_z eğitimleri konusunda bilgiler verip sistemin nasıl çalıştığını gösteren bir blok diyagram sunmuşlardır [43].

İSS'ler, araştırma, tıbbi tedavi ve eğitim amaçlı olarak kullanılmaktadır [4]. Li ve diğerleri yaptıkları çalışmalarda, G_z kuvveti altında EEG sinyallerini incelemişlerdir [32, 33]. Wojtkowiak yaşları 23 ile 26 yaşları arasında değişen 68 pilotun kanın akış hızını G_z kuvveti altında incelemiştir [45]. Iwasaki, ve diğerleri 9 sağlıklı kişiyi bir haftalık sürekli 2 G kuvveti uygulamışlardır, eğitim sonunda G kuvvetinin kardiyovasküler sistemine olan etkilerini incelemişlerdir [24]. Bir başka çalışmada, yüksek G_z kuvvetinin fizyolojik etkileri incelenmiştir [30]. Ercan ve diğerleri yaptıkları çalışmada pilotların sağlık kontyuvarlanma (roll)erinde insan santrifüj sistemlerin önemini anlatmışlardır [18]. İSS'ler uçuş eğitimleri konusunda kullanılan en önemli sistemlerdir. Bu çalışmada, pilot eğitimleri için son derece önemli bu sistemler hakkında bilgiler toplanmış ve İSS'de karşılaşılan bir problemin çözümü için öneriler sunulmuştur. Bu sistemlerde kullanılan ve sistemin hareketi için son derece gerekli olan açısal hız değerinin hesaplamasında kullanılan ancak her durum için çözüm sunamayan doğrusal olmayan diferansiyel denklem incelenmiştir, çözüm bulamadığı noktalar için neden çözüm bulunamadığı araştırılmıştır. Açısal hızın hesaplanması konusunda literatürde iki çalışma yapılmıştır [1, 2]. Çalışmanın ilerleyen bölümlerinde bu çalışmalarda kullanılan yöntemler anlatılacaktır. Bu çalışmada yer alan açısal hız çözümü ile alakalı çalışmamız *ISDFS 2017* de yayınlanmıştır [14]. Bu sistemler hakkında yayınlanan bir patentte, Crosbie ve Colombo, açısal hız değerinin çözülememesinden dolayı, teğetsel ivmenin ihmal edilmesiyle elde edilen açısal hız denklemini kullanmıştır [15]. Krishnaswamy, 1965 yılında yaptığı çalışmada, İSS'lerin istenmeyen etkilerini azaltma konusunda bir çalışma yapmıştır [27] ve açısal hızın neden çözülemediğini açıklamıştır. 1965, 2012 ve 2013 yıllarında yapılan çalışmalara bakıldığında açısal hız değerinin istenilen düzeyde çözülemediği görülmektedir.

İSS'lerin tarihi incelendiğinde, çok uzun yıllardır farklı amaçlar için kullanıldığını görmekteyiz [3, 39, 42]. İlk yapılan çalışmalarda zihinsel hastaları iyileştirmek amacıyla kullanılmıştır [39]. İnsan üzerinde ilk çalışma Erasmus Darwin tarafından yapılmıştır. Darwin yaptığı çalışmada, yer çekiminden dolayı oluşan G kuvveti, kanının beyinde ya da ayakta

toplanması amacıyla kullanılabileceğini önermiştir [3]. Yüksek *G* kuvveti neticesinde oluşan potansiyel problemler, ilk olarak 1. Dünya savaşındaki uçak kazalarında görülmüştür, ilk olarak havada bayılma olarak tanımlanan *G* LOC kavramı araştırılmaya başlanmıştır. Bu yıllarda ayrıca G'ye karşı tolerans artırmak için pilotun hangi tür dayanıklılık manevraları yapabileceği araştırılmıştır. İkinci dünya savaşında uçakların manevra kabiliyetlerinin gelişimi ile birlikte *G* kuvvetlerinin zararlı etkileri konusunda yapılan çalışmalar ivme kazanmıştır [4]. Bu yıllar ve sonrasında havacılık açısından bu konunun öneminin farkına varılmış, uçuş mürettebatında *G* kuvvetinden kaynaklı sorunları azaltabilmek için çalışmalar daha da artmıştır. Bu çalışmalar sonucunda günümüzde modern askeri uçaklarda da kullanılan G-suit adı verilen ve pilotu yüksek *G* kuvvetinden koruyan kıyafet tasarlanmıştır. Pilotun uçuş esnasında oturduğu koltuğun yatış pozisyonu ve uçuş öncesi pilotun genel sağlık durumunun, *G* toleransına katkıları üzerinde çalışmalar yapılmıştır. Ölümcül uçak kazalarının önemli bir nedeninin G-LOC olduğu yine bu dönemlerde yapılan çalışmalarla anlaşılmıştır. Bilimsel araştırmalar ve uçuş mürettebatının eğitimleri için İSS'ler kullanılmaya başlanmıştır. Havacılıkta birim zamanda değişen *G* kuvvetleri onset olarak adlandırılır. 1935-1964 yılları arasında üretilen İSS'ler tek eksenli ve düşük *G* onset profilleri çalıştırmak için kullanılmaktaydı. 1964 ve 1984 yılları arasında iki eksenli sistemler ortaya çıkması ile beraber zamanla 6 G/s ve daha sonra 10 G/s onset yapabilme yeteneklerine sahip sistemler geliştirilmiştir. İSS'ler halen bir çok gelişmiş ülkede pilot eğitimleri, araştırma ve geliştirme konularında başarılı bir şekilde kullanılmaktadır [37].

İSS'in dönüş yarı çapı gerçekte olan bir uçuş sırasındaki dönüş yarı çapından oldukça küçüktür. Dönüş yarı çapının küçük olması Coriolis kuvveti adı verilen ve pilot üzerinde hoş olmayan bir etkiye neden olmaktadır. Bu kuvvet dönüş sırasında kafanın sağa sola döndürülmesi sonucu orta çıkmaktadır. Gerçek bir uçuşta Coriolis olmaması için dönüş yarı çapının 60 m üzerinde olması gerekir; ancak İSS'de böylesine büyük bir yarı çapın olabilmesi motora bağlı olan ana kolunun döndürülmesi için gerekli olan tork miktarından dolayı mümkün değildir. Yapılan çalışmalarda İSS'lerin yarı çapının 8 ile 9 metre arasında olduğu görülmüştür [1, 2, 5].

Zorana ve diğerleri tarafından yapılan çalışmada yüksek *G* kuvveti eğitimlerinde kullanılan profiller ile alakalı bilgiler verilmiştir [6].

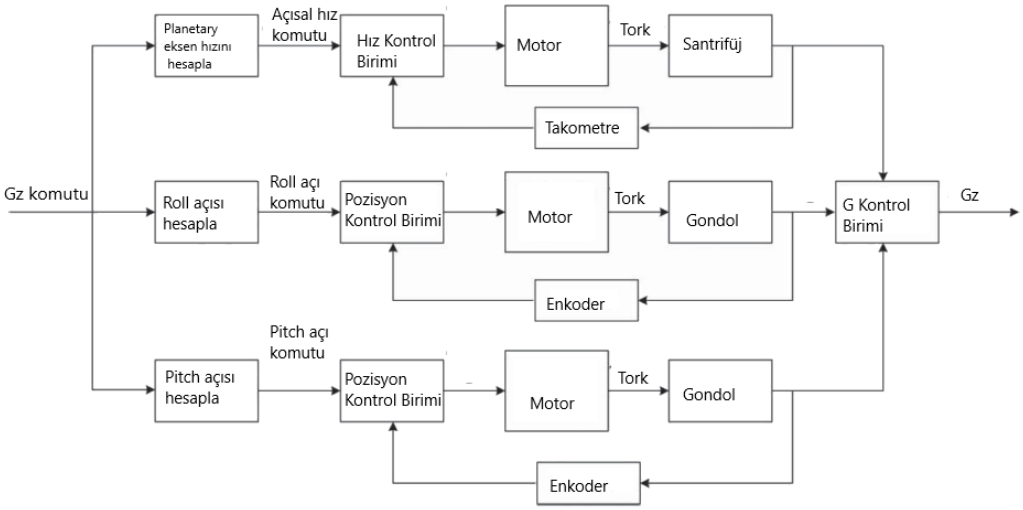
Günümüzde İSS'ler pilot eğitimleri konusunda var olan en değerli araçlar olmalarına rağmen,

bu sistemlerin nasıl çalıştıkları ile alakalı çalışmalar literatürde oldukça kısıtlıdır.

Bu çalışmada önerilen yöntemler, azalan G komutları için önerilen algoritmanın dezavantajlarını yok etmeye dayalı araştırma ve sonuçları içermektedir.

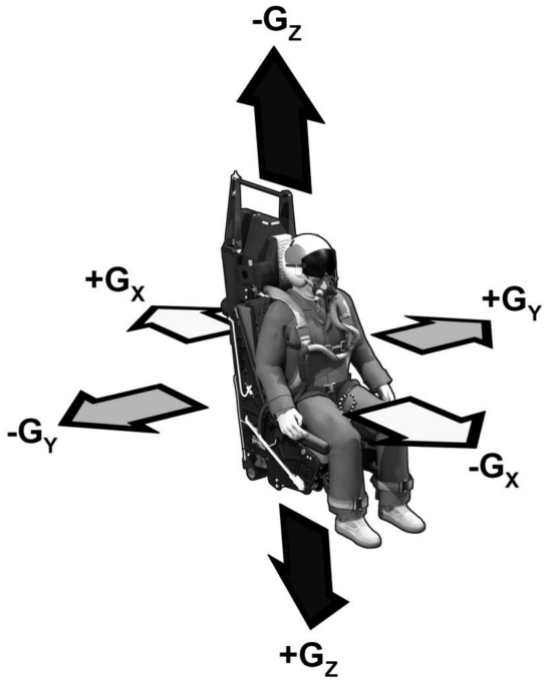
2.2. Genel Yapısı ve Çalışma Prensibi

Tsai ve Shih yaptıkları çalışmada, G_z eğitimleri konusunda faydalı bilgiler vermişler; Şekil 2.2'de gösterilen bir sistemi blok diyagram olarak sunmuşlardır [43].

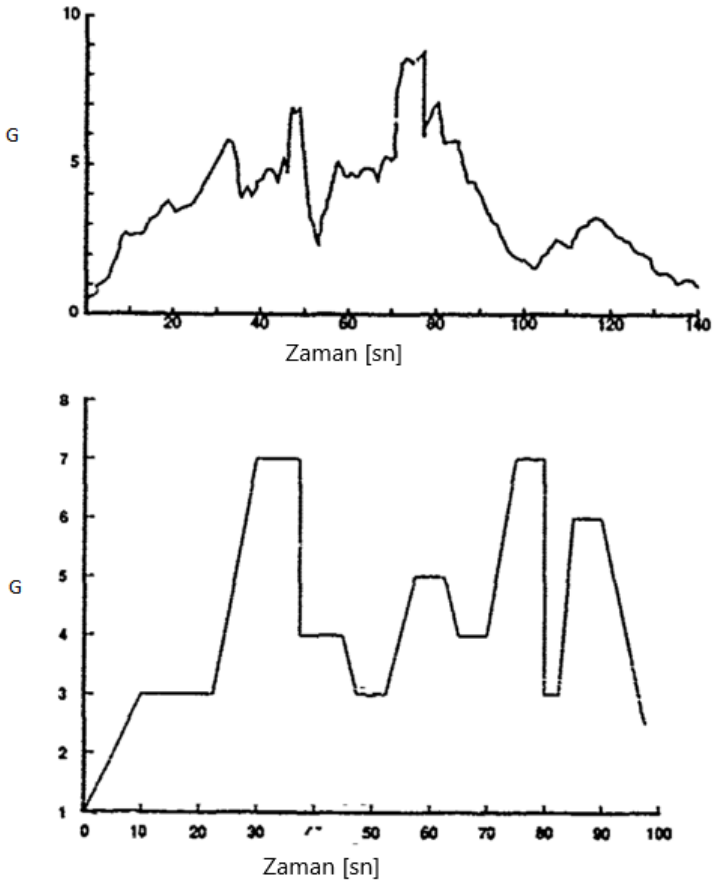


Şekil 2.2. İnsan santrifüj sistem için kontrol blok diyagram

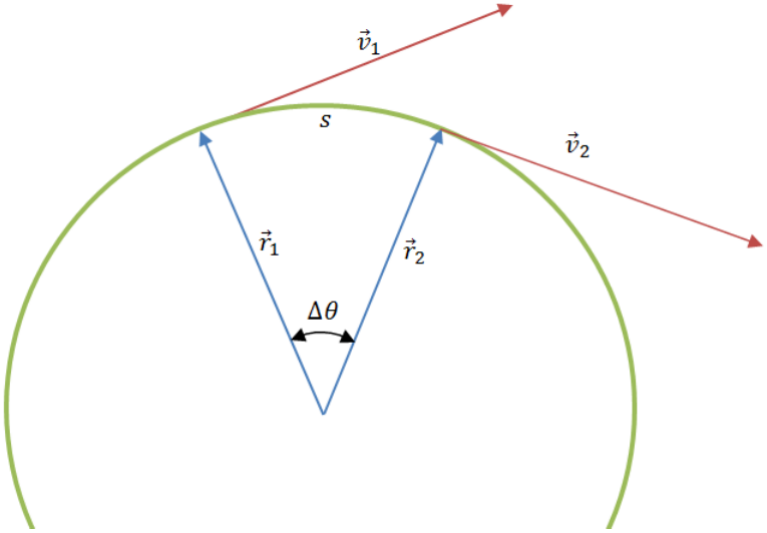
Şekil 2.3'de pilot üzerine etki eden kuvvetler gösterilmiştir. Yukarıdan aşağı etki eden G_z kuvveti eğitim sırasında karşılaşılan en önemli kuvvettir ve olabildiğince eğitimler esnasında bu kuvvet tek başına uygulanmaya çalışılır. Baş bölgesinden ayaklara doğru olan bu kuvvet kanın ayak bölgesine doğru akmasına neden olur. Aşağı doğru olan yön pozitif G_z kuvvet olarak değerlendirilir [17]. Yapılan eğitimlerin büyük çoğunluğu yalnız pozitif G_z kuvvetlerini uygulamayı amaçlar. Ayak bölgesinden baş bölgesine doğru olan kuvvet negatif G_z kuvveti olarak adlandırılır. Negatif G_z kuvveti, kanın beyne doğru ilerlemesine neden olur ve ölümcül olabilmektedir. Diğer iki kuvvet G_x, G_y kuvvetleridir, eğitim sırasında bu kuvvetlerin gerçek uçuşa yakın olarak simüle edilebilmesi için minimum olması beklenir. İSS'ler Şekil 2.4'de gösterilen bir uçuş senaryosunu, gerçek bir uçuş yapmadan, olabildiğince en gerçekçi şekilde uygulanabilmesini sağlamaktadırlar.



Şekil 2.3. Yerçekimsel eksenler



Şekil 2.4. F-4 uçağına ait gerçek ivme ölçerden toplanan veriler (Üst). İSS üzerinde çalıştırılan *G* profil verisi grafiği (Alt)



Şekil 2.5. Düzgün dairesel hareket

İSS'lerin çalışması dairesel harekete dayanmaktadır. Düzgün dairesel hareket, bir yörüngeyi sabit hızla geçen bir cismin hareketini tanımlar [41]. Cisminin dönme eksenine olan uzaklığı hareket boyunca her zaman sabit kalır. Cismin sürati (speed) sabit olmasına rağmen, hızı (velocity) sabit değildir. Sürat skaler bir büyüklük iken hız vektörel bir büyüklüktür, diğer bir ifade ile hızdan bahsedilirken bir yönü ve büyüklüğünün olduğu anlaşılır. Şekil 2.5'te dairesel harekette süratin sabit ama hızın yönünün nasıl değiştiği gösterilmektedir. Bu değişken hız bir ivmenin varlığını gösterir, ivme hız vektörlerin farkının geçen zamana bölünmesi ile elde edilir. Dairesel hareket sonucunda oluşan bu ivme İSS'lerin de çalışma temelini oluşturmakta ve uçuş esnasında pilotlar üzerinde *G* kuvvetlerinin simüle edilmesini sağlanmaktadır.

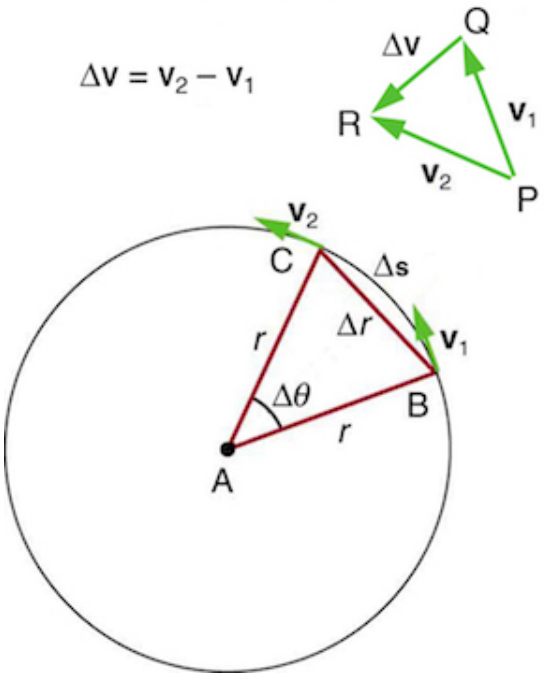
Bu sistemlerdeki en önemli parametre sistemin yapabildiği maksimum onset (G kuvvetinin birim zamanda değişimi) oranıdır. Bu çalışmada Lola Enstitüsü tarafından geliştirilmekte olan İSS'nin yetenekleri incelenmiştir [1]. İSS uygulayabildikleri maksimum *G* kuvveti, kullanılan motor, kol uzunluğu gibi faktörlere göre farklılık gösterebilirler; ancak kontrol açısından bakıldığında benzer uygulamalara sahiptirler. Sistem kontrolünü sağlayan yazılımlarda başlıca iki operasyonel mod bulunur:

- Açık çevrim mod (Önceden tanımlı *G* profillerinin yürütülmesi için kullanılır)
- Kapalı çevrim mod (Pilot manevra kolu hareket ettirerek uçar, *G* kuvveti aerodinamik hesaplamalar sonucunda hesaplanır)

2.3. Kinematik

İSS'in çalışma prensibi dairesel harekete dayanır. İSS'de yer alan ana kolun ucundaki kabininin sabit bir merkez etrafında, yarıçapın sabit olduğu dairesel bir yörüngede, döndürülmesi sonucu istenen G kuvvetler elde edilir. Bir nesne dairesel bir yörüngede sabit hızla hareket etmesine rağmen ivmelenebilir. İvmenin yönü hareketin gerçekleştiği yörüngenin merkezine doğrudur. Sabit hız olmasına karşı oluşan ivmenin nedeni gerçekleşen hareketin yönündeki değişiklikten kaynaklanır. Bu ivmeyi, günlük yaşantımızda da deneyimleriz. Arabayla bir köşeyi dönerken direksiyonu sabit tutar ve sabit hızla hareket edersek, bir yanal ivmelenme (lateral acceleration) deneyimleriz. Dönüş hızınız ne kadar yüksekse, bu ivme o kadar etkili bir şekilde hissedilir [41, 44]. Şekil 2.6'da bir nesnenin iki farklı noktadaki hızları ve yönleri gösterilmiştir. Sürat değişimin Δv merkeze doğru olduğu gösterilmiştir. G_c merkezcil ivme olarak adlandırılır ve Eşitlik 2.1'de gösterildiği gibi birim zamandaki hız vektöründeki değişime eşittir.

$$G_c = \frac{\Delta v}{\Delta t} \tag{2.1}$$



Şekil 2.6. Dairesel hareket

Şekil 2.6'da düzgün dairesel harekette ($v_1 = v_2$) ABC ve PQR üçgenlerinin benzer olduğu görülmektedir. Benzerlik kuralları uygulandığında ise Eşitlik 2.2 elde edilir.

$$\frac{\Delta v}{v} = \frac{\Delta s}{r} \tag{2.2}$$

Eşitlik 2.2'de Δv yalnız bırakılırsa Eşitlik 2.3 elde edilir.

$$\Delta v = \frac{v}{r}\Delta s \tag{2.3}$$

Eşitlik 2.3'te her iki taraf zamandaki değişime, Δt, bölünürse, Eşitlik 2.4 elde edilir.

$$\frac{\Delta v}{\Delta t} = \frac{v}{r}\frac{\Delta s}{\Delta t} \tag{2.4}$$

Eşitlik 2.4'te $\frac{\Delta s}{\Delta t} = v$ bilindiğinden ve Eşitlik 2.1'de gösterilen merkezcil ivme yerine yazılırsa Eşitlik 2.5 elde edilir.

$$G_c = \frac{v^2}{r} \tag{2.5}$$

Burada dikkat edilirse merkezcil ivme, süratin karesi ile orantılıdır. Bir örnek verirsek, bir araçta 100 km/s ile bir virajı dönmek, 50 km/s ile virajı dönmekten dört kat daha zordur. Merkezcil ivmeyi Eşitlik 2.7'de gösterildiği gibi açısal hız ile ifade edilebilir. Bu denkleme şu şekilde ulaşılabilir. Dairesel yörüngede $\Delta\theta$ kadar açı yapılırsa Δs kadar yol alınır. Birim zamandaki yer değiştirme doğrusal hız, v, olarak ifade edersek, $v = \frac{\Delta s}{\Delta t}$ denklemi yazılabilir. $\Delta\theta = \frac{\Delta s}{r}$ denkleminden $\Delta s = r\Delta\theta$ olduğu görülmektedir. Son denklemler kullanılarak, doğrusal hız Eşitlik 2.6'de gösterildiği gibi yazılabilir.

$$v = \frac{r\Delta\theta}{\Delta t} = \omega r \tag{2.6}$$

$$G_c = \omega^2 r \tag{2.7}$$

Nesnenin hızının hem büyüklüğünün hem de doğrultusunun değiştiği durumda merkezcil ivmeye G_c ek olarak cismin teğetsel hızının v değişiminden dolayı bir ivmeden söz edilir. Bu ivmeye teğetsel ivme denir, G_t ile gösterilir. Bileşke ivme, G, Eşitlik 2.10 ve daha açık bir ifadeyle Eşitlik 2.11'de gösterildiği gibi hesaplanır.

$$\omega' = \alpha = \frac{\omega_{(t)} - \omega_{(t-1)}}{\triangle t} \tag{2.8}$$

$\omega_{(t-1)}$ bir önceki zaman çerçevesindeki açısal hızı, $\triangle t$ iki zaman çerçevesi arasında geçen süreyi, $\omega_{(t)}$ ve ω' sırasıyla o andaki açısal hızı ve açısal hızın birim zamandaki değişimi olan açısal ivmeyi gösterir.

$$G_t = \frac{dv}{d\triangle t} = \frac{dwr}{d\triangle t} = \alpha r \tag{2.9}$$

$$G^2 = {G_c}^2 + {G_t}^2 + g^2 \tag{2.10}$$

$$G^2 = \left(\frac{\omega^2 r}{g}\right)^2 + \left(\frac{\alpha r}{g}\right)^2 + 1 \tag{2.11}$$

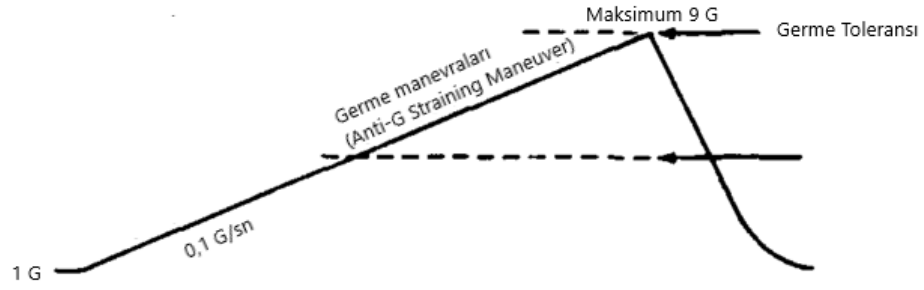
Eşitlikde yer alan G_c merkezcil ivmeyi, G_t teğetsel ivmeyi, g yer çekiminden dolayı oluşan ivmeyi, r hareketin gerçekleştiği dairesel yörüngenin yarıçapını temsil etmektedir. Teğetsel ivme hızın büyüklüğünün zamana göre değişimini ifade eder (Eşitlik 2.9).

2.4. Yüksek G Eğitimi

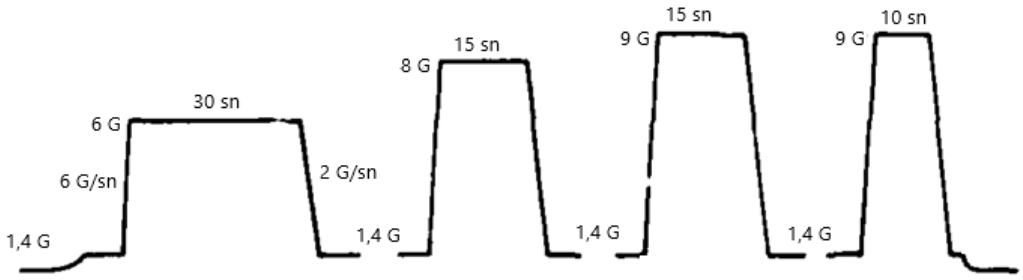
Yüksek G eğitimlerinin amacı pilotların yaptıkları manevralar sonucu maruz kaldıkları G kuvvetlerine karşı tolerans değerini yükselmektir. Bu eğitimler, G stres ve G tolerans mekanizmasının fiziksel anlamlarını daha iyi kavratabilmek ve pilotların bu gibi durumlar karsısında yapabileceği hamleleri (Anti G Straining Manevrası) önceden anlatabilmeye olanak vermektedir.

Yeryüzünde 80 kg ağırlığındaki bir kişi pozitif 6 G_z kuvveti altında 6 x 80 = 540 kg olur. Ağırlığın bu derece yükselmesi vücut hareketlerini güçleştirir. Maruz kalınan ivme bazen uçaktan atlanılmasına bile imkan vermez. Bu tip durumları aşmak için bazı uçaklarda atlama sandalye sistemleri kullanılır.

Yüksek G profilleri ile alakalı yapılan bir çalışmada uygulanan eğitimler ve amaçları anlatılmıştır [26]. Şekil 2.7'de düşük onset değerine sahip profil, Şekil 2.8'de yüksek onsetli ROR eğitimi için kullanılan grafikler gösterilmiştir. ROR profillerinin amacı, verilen bir onset değeri ile hedef G seviyesine ulaşmak ve ulaşılan G seviyesinde bir müddet bekletilerek eğitim vermektir.

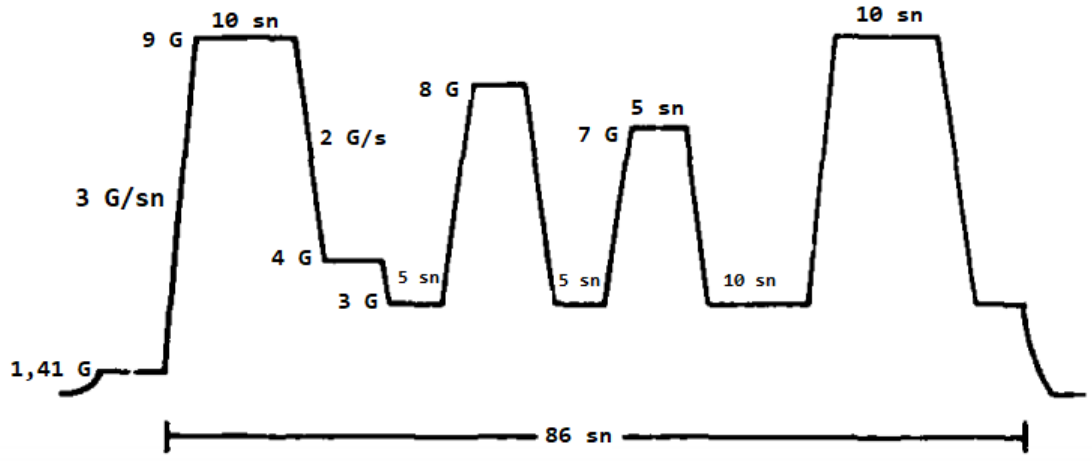


Şekil 2.7. GOR



Şekil 2.8. ROR

ROR eğitimlerinden sonra Benzetilmiş Hava Muharebesi (SACM, Simulated Air Combat Maneuvering) adı verilen eğitim uygulanır. Şekil 2.9'da bu eğitimde kullanılan 86 saniyeden oluşan *G* profili gösterilmektedir. Belli *G* seviyesine çıktıktan sonra en düşük 3 *G* seviyesine inmesi daha sonra başka bir *G* seviyesine çıkarılması şeklinde bir dizi *G* profilinin birleştirilmesi ile oluşturulmuştur. Eğitim gören pilotun manevra kolu yardımıyla istenen *G* profilini simüle eden başka bir uçağı takip etmesi amaçlanmaktadır.



Şekil 2.9. Benzetilmiş hava muharebesi

Eğitimler genellikle düşük onset içeren profiller ile başlar, eğitim başarısına göre daha yüksek *G* kuvvetleri uygulanarak pilotların yetenekleri ve dayanıklılıkları test edilir [6]. İSS'lerde kullanılan bütün pilot eğitimlerinin temel amacı gerçek uçuş esnasında karşılaşabilecek kuvvetleri en gerçekçi şekilde simüle etmektir [6]. Sistemin hareketi ve uygulanan kuvvetlerin hesaplaması ile alakalı Kvrgic ve diğerleri tarafından yapılan bir çalışma mevcuttur [5]. Bu sistem açısından bazı önemli denklemler Bölüm 2.3'te anlatılmıştır. Yüksek *G* eğitimi konusunda bilgiler Bölüm 2.4'de anlatılmıştır.

2.5. G-LOC (G Kuvvetlerine Bağlı Bilinç Kaybı)

G kuvvetlerine bağlı olarak oluşan bilinç kaybı G-LOC olarak ifade edilir. Bu terimdeki *G* harfi maruz kalınan *G* kuvvetini ifade etmektedir. Beyin ve gözler kendi fonksiyonlarını devam ettirebilmeleri için oksijen ve glikoza ihtiyaç duyarlar. Bu organlarda yeterli miktarda oksijen ve glikozun ulaşamadığı durumda G-LOC meydana gelir. Beyinin ihtiyaç duyduğu besinler kan yoluyla iletilir, bu iletim yer çekimi kuvvetine karşı sürekli devam eder. Uzun süre $+Gz$ kuvvetine maruz kalındığında bu organlara giden glikoz ve oksijen miktarı azalır ve organların fonksiyonları yavaşlar ve durur.

Günümüzde çok yüksek hızlara ulaşan ve +12 G_z gibi kuvvetlere ulaşan manevralar yapabilen modern savaş uçakları ile birlikte G-LOC sorunu ile daha çok karşılaşılmaktadır. *G* kuvvetinin şiddetine bağlı olarak görüş keskinliğinin azaldığı; İSS'ler kullanılarak gerçekleştirilen deneylerde kanıtlanmıştır. Havacılıkta G-LOC dışında kullanılan görüş ile alakalı bazı kavramlar ve durumlar aşağıda listelenmiştir.

- Tünel Görüş : Çevresel görüşün kaybolduğu durumlar
- Grayout : Renkli görüşün kaybolduğu durumlar
- Blackout : Görmenin tamamen kaybolduğu durumlar

Pilot *G* kuvvetlerine maruz kaldığında üzerinde hissettiği ağırlık artışından dolayı baş ve kol hareketlerini yapmakta zorlanmaya başlar. Kademeli olarak artan *G* kuvvetleri neticesinde göze ulaşan kan miktarındaki azalmaya bağlı olarak Grayout olarak tanımlanan görüş alanın kaybolması olayı meydana gelir. Bu olayın devamında Tünel Görüş olarak ifade edilen çevresel görüşün azalması yaşanır. Uygulanan *G* kuvvetinin yükselmeye devam etmesi

durumunda göze gelen kan akımı durur ve görüşün tamamen kaybolduğu Blackout olarak ifade edilen durum meydana gelir. Blackout olayından sonra da G kuvveti artmaya ya da aynı seviyede kalmaya devam ederse, bilinç kaybı olarak ifade edilen G-LOC olayı meydana gelir. G-LOC gerçekleştikten sonra, G kuvvetinin uygulanmaya devam edilmesi durumunda kişinin beyin ölümü gerçekleşebilir. G-LOC sonrası, normal G_z düzeylerine ulaşıldığında pilot bir süre bilinçsiz kalır. Uçuşta G-LOC meydana geldiğinde uçağın kontrolü belli bir süre kaybedileceğinden dolayı sonuçları da ölümcül olabilmektedir.

3 . YAPAY SİNİR AĞLARI

Yapay zeka, bir bilgisayar ya da robotun, insana ait olan fikir yürütme, genelleme, problem çözebilme, geçmiş tecrübelerden öğrenme gibi nitelikleri yerine getirme yeteneği olarak tanımlanmaktadır [38]. Yapay zekaya sahip bir sistemin oluşturulması için psikoloji, dil bilimi, bilgisayar bilimleri ve sinir bilim gibi bilim dallarının içinde yer aldığı disiplinler arası bir çalışma yürütmek gerekir. Yapay zeka araştırmaları, insan beyninin nasıl çalıştığı, beyin tarafından yerine getirilen görevlerin incelenmesi, makinelerin benzer görevleri nasıl taklit edecekleri konuları ile yakından ilgilenir. İnsan beyni yeryüzünde bilinen en karmaşık yapılardan biridir. Bu karmaşık yapıyı anlamak, insan gibi düşünen ve davranan bir sistem üretmek, insanoğlunun uzun yıllardır üzerinde çalıştığı en önemli konulardan biridir. Beynimizle alakalı bildiğimiz bilgiler daha çok giriş ve çıkış süreçleri ile alakalıdır. Beynin anlaşılması gereken en önemli aşaması olan merkezi değerlendirme kısmı halen iyi anlaşılamamıştır.

Bu bölümde Yapay Zeka teknikleri içerisinde yer alan Yapay Sinir Ağları (YSA) hakkında genel bilgiler verilecektir. Bölüm 3.1'de YSA için yapılan tanımlar ve tarihçesi anlatılmıştır. Sırasıyla Bölüm 3.2 ve Bölüm 3.3'te beyin yapısı ve sinir sistemi için önemli olan nöron ve sayısal ortamda taklit edilmeye çalışılan nöron diğer adıyla işlem birimi yapıları anlatılmıştır. Bölüm 3.4'te YSA türleri anlatılmıştır. Bölüm 3.5'te YSA'yı eğitmek için kullanılan algoritmalara yer verilmiştir. Bir çok problemin çözümünde kullanılan YSA modellerinin kurulumu parametre sayısının fazlalığından dolayı kolay değildir. Parametrelerin seçimi problemin türüne göre değişkenlik gösterir. Başlıca belirlenmesi gereken parametreler : kullanılacak ağın türü, öğrenme algoritması, ağda yer alacak nöron ve katman sayısı, kullanılacak olan aktivasyon fonksiyonu, ağırlıkların başlangıç değerlerinin ne olacağı gibi parametreler ağın performansını belirler [40]. Bu çalışmada yapılan denemeler neticesinde en iyi sonuç veren YSA modeli ve RTFA modeline Bölüm 4.6 ve Bölüm 4.7'de yer verilmiştir.

3.1. YSA'nın Tanımı ve Tarihçesi

YSA canlı organizmalarda bulunan sinir sisteminden esinlenerek oluşturulan bir modeldir [38]. Biyolojik öğrenmeyi temel alan sinir sistemine benzer bir yapıya sahiptir. İnsan, beynin öğrenme yeteneği sayesinde yeni bilgiler üretebilir, düşünebilir ve bir olayı gözlemleyebilir.

YSA bu yeteneklerini taklit edebilen sistemler geliştirmek için tasarlanmıştır. Nielsen YSA'nın tanımı "Dışarıdan gelen girdilere dinamik olarak yanıt oluşturma yoluyla bilgi işleyen, birbiriyle bağlantılı basit elemanlardan oluşan bilgi işlem sistemidir" olarak yapmıştır [13]. Haykin ise "Bilgiyi depolamak için doğal eğilimi olan basit birimlerden oluşan paralel dağıtılmış bir işlemci" olarak tanımlamıştır [21]. YSA'nın en temel ortaya çıkış amacı; insan beyninden yola çıkarak, öğrenme sürecinin matematiksel modellenmesidir.

YSA başlıca sınıflandırma, modelleme ve tahmin uygulamaları olmak üzere pek çok alanda kullanılmaktadır [38]. YSA ile matematiksel modelleri oluşturulamayan çok zor olarak tanımlanan problemleri çözmek için kullanılmaktadır [21]. YSA başlıca kullanım alanları aşağıda listelenmiştir.

- Doğrusal olmayan sistem modelleme
- Akıllı kontrol
- Optimizasyon
- Sınıflandırma
- Örüntü tanıma, ilişkilendirme ve eşleştirme

YSA temellerinin atıldığı en önemli olaylar 1940'lı yıllarda McCulloch ve Pitts ve Hebb tarafından yapılan çalışmalara dayanmaktadır. 1943 yılında McCulloch ve Pitts, matematik ve algoritmalara dayalı elektrik devreleriyle çok basit bir sinir hücresi modellemişlerdir [36]. 1949'de psikolog Donald Hebb tarafından şimdi Hebbian öğrenme olarak bilinen YSA'nın ilk şekli sunulmuştur [23]. Bu ağ yapısı girdi ve çıktı katmanından oluşmakta, ağırlıklar elde edilen çıktı be girdilere göre tekrar hesaplanması temeline dayanıyor. 1958 yılında Frank Rosenblatt, iki katmanlı bir öğrenme modeli olan perceptronu oluşturdu [22]. Sırasıyla gerçekleşen bu olaylar bugünkü YSA'nın temellerini oluşturmaktadır. 1969 yılında yayımlanan Misnky ve Pappert tarafından yazılan Algılayıcılar isimli kitapta yer alan perceptron doğrusal olmayan problemleri çözemediğinden bilimsel değer taşımadığını iddia etmesiyle bu konuda yapılan araştırmalarda bir duraklama görülmüştür. Misnky ve Pappert tezlerinin haklılığını anlatabilmek için XOR probleminin algılayıcılar tarafından çözülemediğini örnek olarak göstermişlerdir. 1975 yılında ise Paul Webor geriye yayılım algoritmasını oluşturuldu. Rumelhart ve diğerleri 1982 yılında YSA tarihsel gelişimi bakımından çok önemli olan çok katmanlı algılayıcılar (multilayer perceptron) konusunu orta

atmışlardır. Daha önce Misnky ve Pappert tezlerinde yayınlanan tek katmanlı algılayıcıların çözemediği XOR problemi çok katmanlı algılayıcıların ortaya çıkması ile çözüme ulaşmıştır. 1988 yılında Broomhead ve Lowe, yaptıkarı çalışmada Çok katmanlı algılayıcılar modeline alternatif olarak geliştirilen Radyal Tabanlı Fonksiyonlar (RTF) modelini oluşturdular.

YSA'da bilgiler ağın tamamında saklanır. Geleneksel programlamada olduğu gibi bilgiler veri tabanları ya da dosyalarda tutulmaz, oluşturulan ağın tamamına yayılarak saklanmaktadır. Ağda yer alan hücrelerden (nöron ya da işlem birimi) bazılarının işlevini yitirmesi, anlamlı bilginin kaybolmasına neden olmaz. YSA'nın öğrenmenin gerçekleşmesi için ilk önce örneklerin belirlenmesi, daha sonra eldeki örneklerin ağda işleme sokularak istenen çıktılara göre ağın eğitilmesi gerekir. Ağın başarısı, seçilen örneklerin ne kadar doğru seçilmesi ile yakından ilgilidir. Oluşturulan ağda problem bütün yönleri ile ele alınmazsa, ağ yanlış çıktılar üretebilir. YSA en dikkat çekici bir özelliği de daha önce görülmemiş örnekler hakkında bilgi üretebilmesidir [20, 25]. YSA'lar eğitimleri sırasında kendilerine verilen örneklerden genellemeler çıkarırlar ve bu genellemeler kullanılarak yeni örnekler hakkında bilgi üretebilirler. Algılamaya yönelik olaylarda kullanılabilirler [40].

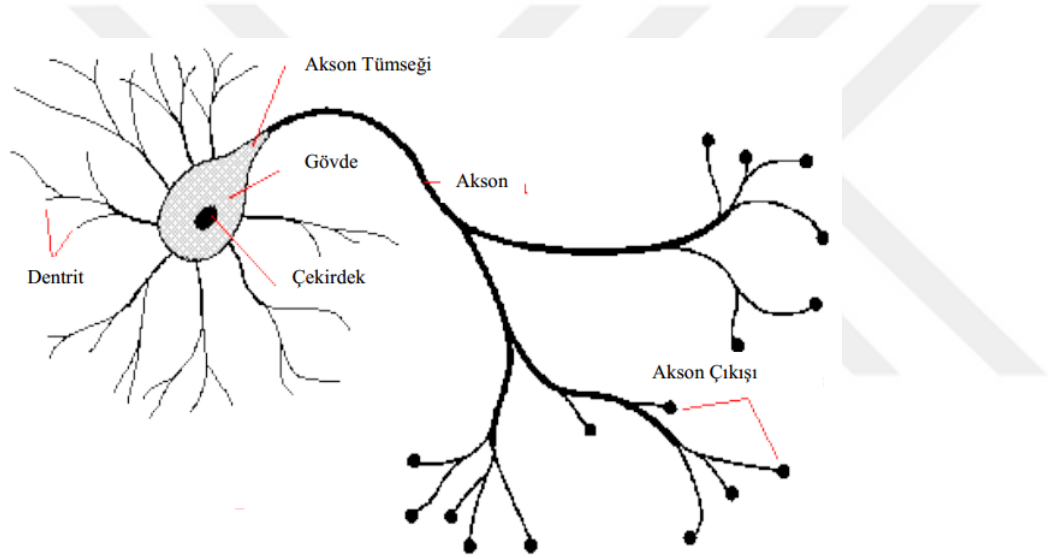
3.2. Biyolojik Nöron

Beynin ana görevleri arasında olan, düşünme ve karar verme işlevlerinin nasıl gerçekleştirildiği konusuna; yapılan araştırmalar halen tam olarak bir cevap sunabilmiş değillerdir. Beyin milyarlarca sinir hücresinin bir araya gelmesi ile oluşur. Beyin hücrelerine nöron adı verilir ve beyinde yaklaşık 1,5 10^{10} adet farklı tipte nöron olduğu düşünülmektedir. Bir tek nöronun tepki verme süresinin sayısal bir lojik kapı devresinin ortalama cevap verme hızından çok daha yavaştır; ancak diğer nöronlarla beraber paralel olarak çalışırken son derecede hızlı yanıt üretebilir.

Biyolojik sinir sisteminin temel yapı taşı olan nöronlar temelde üç bölgeye ayrılırlar: Bunlar,

- Soma
- Akson (axon)
- Dendrit'lerdir (dentrite).

Bu bölgeler bilgilerin girişi ve iletiminde çeşitli görevler almaktadırlar. Şekil 3.1'de biyolojik nöron genel görünümü ve nöronu oluşturan birimler gösterilmiştir. Soma olarak adlandırılan içerisinde nucleus adı verilen hücre çekirdeği bulunduran bu yapı hücreyi denetler ve hücre etkinliklerin tümünü yönetmekle sorumludur. Hücre gövdesinden dentritler ve akson olarak adlandırılan iki çeşit uzantı çıkmaktadır. Dentritler bilgiyi, iletim hatları olarak düşünülen aksonlar vasıtasıyla almak ve hücre gövdesine taşımakla sorumludur. Aksonlar gövdedeki bilgiyi diğer nöronların dentritlerine ulaştırmakla görevlidir. Akson ile dentritler arasında sinaps adı verilen küçük boşluklar bulunur. Sinyal akson ucuna ulaştığında kimyasal maddeler bu boşluğa yayılır. Sinapslar, bilgilerin uzun süre depolandığı bilgi saklama yerleri olarak değerlendirilmektedir.

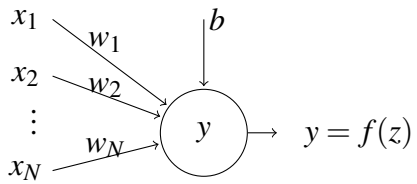


Şekil 3.1. Biyolojik nöron genel görünümü ve nöronu oluşturan birimler

Nörona bağlı tüm dentritlerden gelen sinyaller hücre gövdesinde toplanır. Bu toplam değer bir eşik değerini aştığı zaman nöron ısınmaya başlar ve aksonlar aracılığı ile diğer nöronlara sinyal gönderilir.

3.3. Yapay Nöron

YSA'da en temel unsur yapay sinir hücresidir. Şekil 3.2'de bir yapay sinir hücresi ve onun bileşenleri gösterilmiştir. Yapay nöronun bileşenlerinin biyolojik nöron yapısı içindeki karşılıkları Çizelge 3.1'de gösterilmiştir.



Şekil 3.2. Yapay sinir hücresi ve bileşenleri

Çizelge 3.1. Biyolojik nöron ile yapay nöron karşılaştırması

Biyolojik Nöron	Yapay Nöron
Dentrit	Birleştirme Fonksiyonu
Hücre Gövdesi	Aktivasyon Fonksiyonu
Akson	Nöron Çıkışı
Sinapslar	Ağırlıklar

Bir yapay sinir hücresi aşağıda listelenen beş bileşenden oluşmaktadır.

- Girdiler
- Ağırlıklar
- Birleştirme (Toplama) Fonksiyonu
- Aktivasyon Fonksiyonu
- Çıktı

3.3.1. Girdiler

Diğer nöronlardan ya da dış ortamlardan gelen bilgilerdir. Şekil 3.2'de $x_1, x_2, \ldots, x_N$ olarak gösterilmiştir.

3.3.2. Ağırlıklar

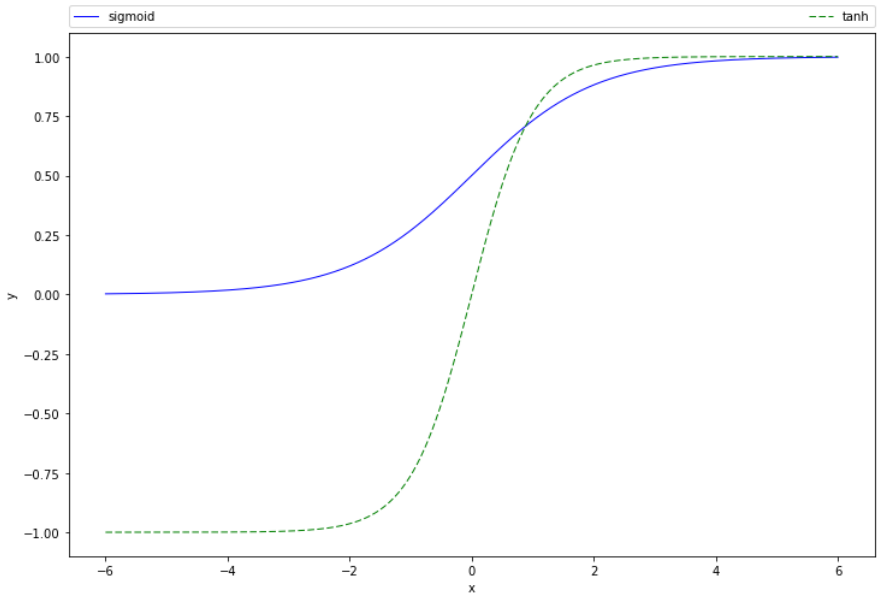
Ağırlıklar bağlı bulundukları girdinin hücre üzerindeki etkisini belirler. Girdi olarak kullanılacak değerlerin matematiksel katsayısını gösterir. Girdilerin nöron hücresi arasında iletimini sağlayan tüm bağlantıların farklı ağırlık değerleri bulunmaktadır. Ağırlıklar öğrenme esnasında doğru çıktı değerini üretebilmek için en uygun hale getirilirler, diğer bir değişle optimize edilirler. Şekil 3.2'de $w_1, w_2, \ldots, w_N$ olarak gösterilmiştir.

3.3.3. Birleştirme fonksiyonu

Bir nöron hücresine gelen net girdiyi hesaplayan fonksiyondur. Genellikle girişlerin ilgili ağırlıkla çarpımlarının toplanmasıyla hesaplanır. Şekil 3.2'de $z = w_1x_1 + \ldots + w_nx_n$ ile ifade edilmiştir.

3.3.4. Aktivasyon fonksiyonu

Transfer fonksiyonu olarak da adlandırılan aktivasyon fonksiyonu, nörona (işlem birimi) gelen net girdiyi (z) işleyerek, nöronun bu girdiye karşılık üreteceği çıktıyı (y) belirler. Aktivasyon fonksiyonu problemin türüne göre tasarımcı tarafından seçilir. En çok kullanılan aktivasyon fonksiyonları sigmoid ve hiperbolik tanjant fonksiyonlarıdır. Şekil 3.3'te sigmoid, tanjant hiperbolik aktivasyon fonksiyonları grafik üzerinde gösterilmiştir. Bu aktivasyon fonksiyonlarının türevleri Şekil 3.4'de grafik üzerinde gösterilmiştir. Aktivasyon fonksiyonların türevlenebilir ve sürekli olması gerekir [29]. Aktivasyon fonksiyonu Şekil 3.2'de f olarak gösterilmiştir.



Şekil 3.3. Aktivasyon fonksiyonları

Aktivasyon fonksiyonlarının matematiksel denklemleri Çizelge 3.2'de gösterilmiştir. Çizelge 3.2'de gösterilen doğrusal aktivasyon fonksiyonu YSA'da genellikle çıkış katmanında kullanılır ve hücrenin net girdisini hücre çıkışı olarak verir. Şekil 3.5'te doğrusal aktivasyon fonksiyonu grafiksel olarak türevi ile beraber gösterilmiştir.

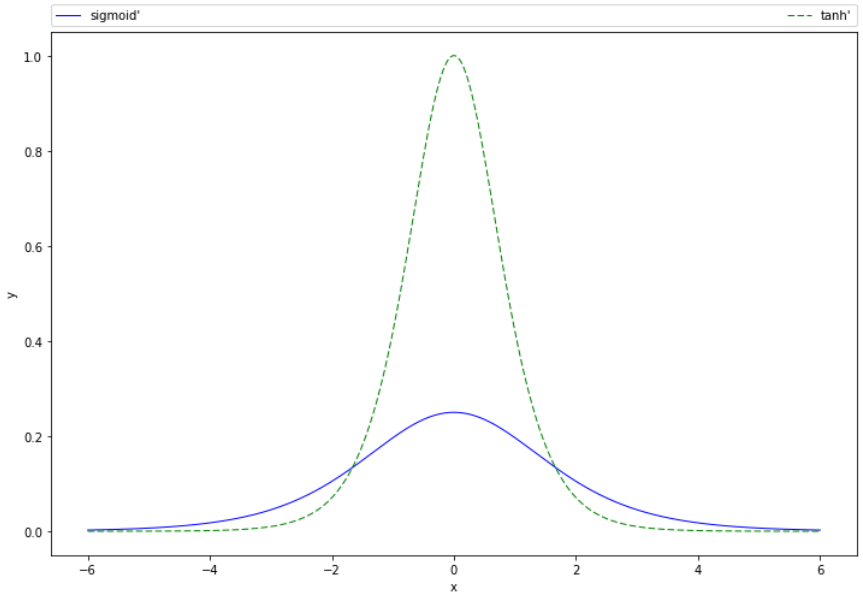
Çizelge 3.2. Aktivasyon fonksiyonları

Fonksiyon Adı	Formül	Aralık
Sigmoid	$f(x) = \frac{1}{1+e^{-x}}$	$f(\cdot) \in [-1, 1]$
Tanh	$f(\alpha) = \frac{e^{\alpha}-e^{-\alpha}}{e^{\alpha}+e^{-\alpha}}$	$f(\cdot) \in [-1, 1]$
Doğrusal	$f(x) = x$	$f(\cdot) \in \mathbb{R}$

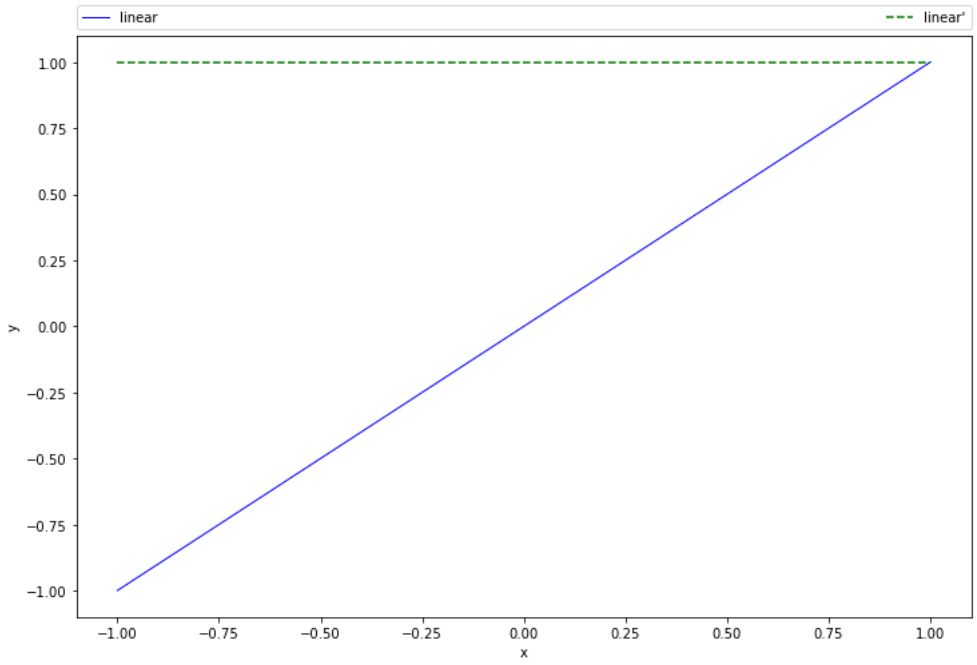
sigmoid fonksiyonu, türevi alınabilir, sürekli ve doğrusal olmayan bir fonksiyondur. Fonksiyon her bir girdi değeri için 0 ile 1 arasında değer üretir. Tanjant hiperbolik fonksiyon sigmoid fonksiyondan farklı olarak girdi aralığı daha geniş olmakla birlikte -1 ile 1 arasında değer üretir. Aktivasyon fonksiyonları elde etmek için kullanılan fonksiyonlar EK-3'de gösterilmiştir.

3.3.5. Çıktı

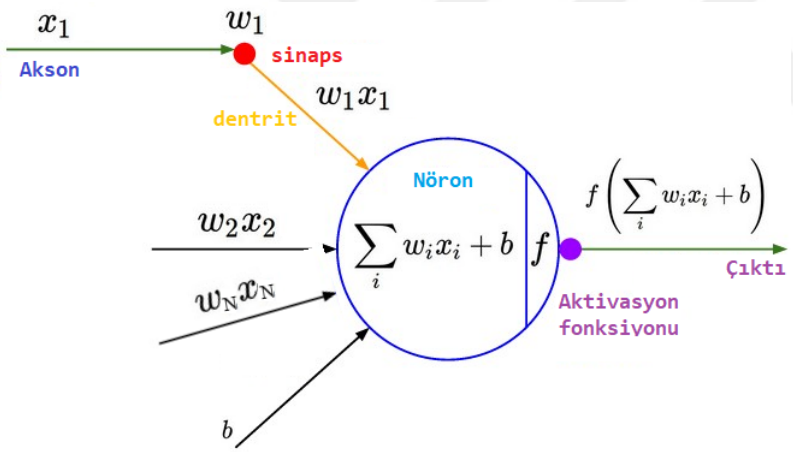
Nörona gelen net girdinin aktivasyon fonksiyonundan geçirildikten sonra elde edilen değeridir. Bu değer başka bir nöron için girdi olarak kullanılabilir. Şekil 3.2'de y olarak gösterilmiştir. Çıktının matematiksel denklemi girdiler, ağırlıklar ve aktivasyon fonksiyonu kullanarak $y(z) = f(w_1x_1 + \ldots + w_nx_n + b)$ olarak ifade edilebilir. Şekil 3.6'da biyolojik nörona benzetilen bir yapay nöron modeli karşılıkları ile birlikte gösterilmiştir.



Şekil 3.4. Aktivasyon fonksiyonların türevleri



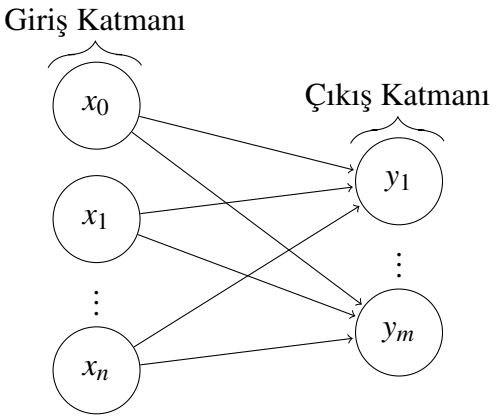
Şekil 3.5. Doğrusal aktivasyon fonksiyon grafiği



Şekil 3.6. Nöron Modeli

3.3.6. Algılayıcı (Perceptron)

Bu bölümde ilk defa Rosenblatt tarafından 1958 yılında tanıtılan perceptron [22] anlatılacaktır. Bütün işlem birimleri kendi çıktıları ile etiketlenmiştir. $y_i = f(z_i)$ değerleri çıkış işlem birimleri için hesaplanan çıktı değerler; x_i ile ağa giriş değerlerini göstermektedir. x_i değerleri ağırlıklandırılmış toplam kuralına göre ağa yayılır. Ağda yer alan $x_0 := 1$ değeri bias temsil eder. Her giriş birimi bütün çıkış birimlerine bağlıdır (Şekil 3.7). Her bir çıkış birimi y_i $(1 \leq i \leq m)$ aşağıda gösterildiği şekilde hesaplanır.



Şekil 3.7. *n* tane giriş ve *m* tane çıkışa sahip bir perceptron ağı

$$y_i = f(z_i) \tag{3.1}$$

$$z_i = \sum_{k=1}^{n} w_{ik} x_k + w_{i0} x_0 \tag{3.2}$$

Şekil 3.7'de bias ya da eşik değerini gösteren işlem biriminin değeri 0 olarak seçilmesi durumunda her bir çıkış biriminin toplam ağırlıklandırılmış net giriş değeri aşağıda gösterildiği şekilde hesaplanır.

$$z_i = \sum_{k=1}^{n} w_{ik} x_k \tag{3.3}$$

3.4. Yapay Sinir Ağların Sınıflandırılması

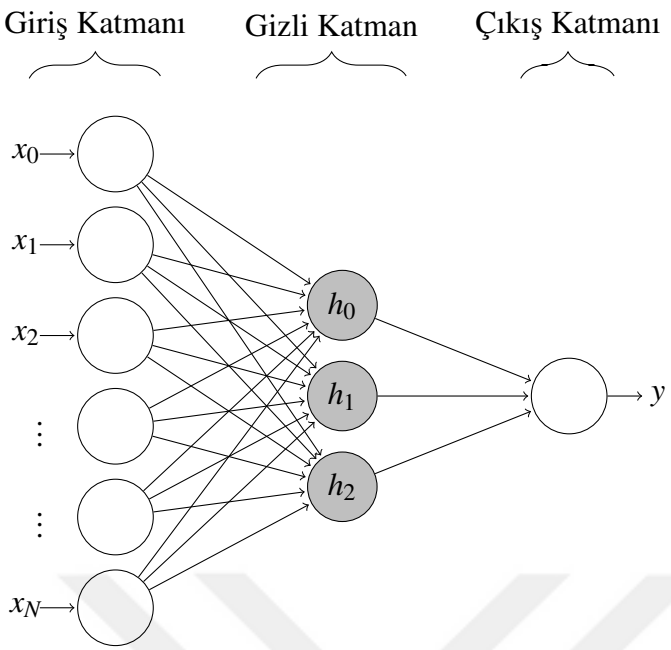
YSA birbirine bağlantılı işlem birimleri gibi düşünebiliriz. İşlem birimleri arasındaki bağlantıların yapısı ağın yapısını belirler. YSA'nın işaretinin akış yönüne göre ileri beslemeli (feedforward) ve geri beslemeli ağ yapıları bulunmaktadır. İstenilen çıktıyı elde etmek için bağlantıların nasıl güncellenmesi gerektiği, öğrenme algoritması ile belirlenir. YSA'nın görevi; bir girdi örüntüsüne karşılık bir çıktı sinyali oluşturmak ve bu sinyali diğer hücrelere iletmektir. Her girdi ile çıktı arasındaki ilişkiyi temsil eden ağırlıklar, her yeni girdi örüntüsü ve çıktı sinyaline göre tekrar ayarlanır. Bu ayarlama süreci öğrenme olarak adlandırılır, girdi örüntüleri, ağırlıklarındaki değişim stabilize (durağan) olana kadar devam eder. Öğrenme biçimine göre YSA'lar Danışmalı (supervised) ve Danışmansız (unsupervised) olarak iki grupta incelenir.

Bir YSA modeli, genel olarak, girdi katmanı, gizli katman ve çıktı katmanından oluşmaktadır. Bu katmanların her biri, bilgi işleyişini sağlayacak şekilde farklı görevlere sahip nöronlar içermektedir. Şekil 3.8'de ileri beslemeli ağ yapısı bu katmanları içerecek şekilde gösterilmiştir. Ağ yapılarına geçmeden önce bu katmanlar anlatılacaktır.

Giriş Katmanı: Yapay sinir ağı modelinin dış dünya ile bağlantısının olduğu katmanıdır. Bu katmanda bulunacak nöron sayısı bağımsız değişken sayısı kadardır. *Gizli Katman:* YSA'da kara kutu olarak da bilinen gizli katmanın görevi giriş katmanından kendisine doğru iletilen bilgiyi işlemektedir. Giriş ve çıkış katmanındaki nöron sayısı eğitim verisindeki değişkenlerinin sayısı ile belirlenmekte iken, gizli katman sayısı ve her bir gizli katmanda yer alacak nöron sayıları kullanıcı tarafından belirlenmektedir [7]. Gizli katman nöronları bilgiyi genellikle doğrusal olmayan aktivasyon fonksiyonları aracılığı ile işlerler. Aktivasyon fonksiyonunun seçimi oluşturulan ağın başarısını etkileyen önemli parametrelerden birisidir. Gizli katman, birden fazla katmandan oluşabileceği gibi tek bir katmandan da oluşabilir. Her bir gizli katman sadece nöronlardan meydana gelir. Basit YSA modellerinde girdi katmanı direk çıkış katmanına bağlanır; ancak bu tip ağlar bağımlı ve bağımsız değişkenler arasındaki doğrusal olmayan ilişkiyi bulamamaktadır. Gizli katman, YSA'ya girdi ile çıktı arasındaki doğrusal olmayan ilişkiyi modelleme gücünü kazandıran katmandır. *Çıkış Katmanı:* YSA modelinde gizli katmanda işlenen verinin dış dünyaya iletildiği katmandır. Gizli katmandan kendisine gelen bilgi genellikle doğrusal aktivasyon fonksiyonu kullanılarak doğrudan ağın çıktısı olarak dış dünyaya iletilir. Gizli katman bulunmayan basit YSA modellerinde ise çıktı katmanı nöronları, gelen bilgiyi bir eşik değer fonksiyonundan geçirerek dış dünyaya iletir.

3.4.1. İleri beslemeli ağlar

Şekil 3.8'de gösterilen girdi katmanı, gizli katman (veya ara katman) ve sonuç katmanından oluşan bu ağ yapısında nöronlar arası iletim yönü girdi katmanından çıkış katmanına doğru tek yönlüdür. Bu yapısında bulunan işlem birimleri (nöronlar) kendi katmanında bulunan işlem birimlerine bağlanamazlar, yalnızca bir sonraki katmandaki işlem birimlerine bağlanabilirler. Giriş katmanı, dış ortamdan aldığı sinyali ya da bilgiyi herhangi bir değişikliğe uğratmadan bir sonraki gizli katmandaki işlem birimlerine iletir. Sinyal, gizli ve çıktı katmanında işlenerek ağ çıkışı belirlenir.



Şekil 3.8. Yapay Sinir Ağ'ın genel görünümü

İki nöron arasında iletilen bilgi, nöronlar arası bağlantı üzerinde tanımlı olan ağırlık değeri ile çarpılarak iletilir. Herhangi bir nörona iletilen net değer, kendisine bağlı olan bir önceki katmandaki nöronların çıktılarının doğrusal bir kombinasyonudur. Eşitlik 3.4'te doğrusal ilişki matematiksel olarak gösterilmiştir. Bu denklemde w ağırlıkları, x bir önceki katmandaki nöron çıktılarını, y ise katmana gelen işlenmiş bilgiyi temsil etmektedir.

$$y(x_1,\ldots,x_n) = f(w_1x_1 + \ldots + w_nx_n) \tag{3.4}$$

Geriye yayılım öğrenme algoritması olarak bilinen ve ileri beslemeli YSA'ların eğitiminde etkin olarak kullanıldığından bu türdeki ağlara geriye yayılım ağları da denilmektedir.

3.4.2. Geri beslemeli ağlar

Geri Beslemeli Yapay Sinir Ağlar (GBYSA) ileri beslemeli ağ yapısından farklı olarak ağda yer alan bir nöronun çıktısı, kendisinden sonra gelen nöron katmanına girdi olarak verilebileceği gibi; kendinden önceki katmanda veya kendisi ile aynı katmanında bulunan bir nörona da girdi olarak bağlanabilir.

3.5. Yapay Sinir Ağların Eğitilmesi

İnsanoğlu doğumundan itibaren dış ortamdan algıladığı sinyalleri yorumlar ve bazı çıkarımlarda bulunur. Bu çıkarımlar insanların davranışlarını belirler. İnsanın yaşamı boyunca karşılaştığı olaylar karşısında elde ettiği tecrübeleri artmakta ve bu olaylar karşısında nasıl tepki vermesi gerektiğine daha doğru bir şekilde karar vermektedir. Hiç karşılaşmadığı bir olay karşısında tecrübesiz kalabilir; ama daha önce elde ettiği tecrübelerden bir sonuç çıkarabilir. YSA öğrenme süreci tıpkı insanın öğrenme sürecine benzer: Dış dünyadan verinin alınması, işlenmesi ve tepki ya da çıkış üretmesi gibi. Bu çıkış yine tecrübeyle elde edilen bir çıkışla karşılaştırılarak hata bulunur. Öğrenme algoritmaları hesaplanan hata değerinin yani daha önce elde edilen tecrübe ile elde edilen çıktı arasındaki farkın azaltılmasını amaçlar. YSA'da hata değeri nöronlar arasındaki ağırlıkların ayarlanması ile sağlanmaktadır. Ağırlıkların sürekli yenilenip istenilen sonuca ulaşılana kadar geçen zamana öğrenme adı verilir. YSA öğrenme işlemi tamamlandıktan sonra eğitim aşamasında kullanılmayan başka girişler verilmek suretiyle ağın oluşturduğu çıktı ile istenen çıktı arasındaki farka bakılır. Eğer elde edilen çıktı istenen değere istenilen kadar yakın bir değere sahipse YSA elde edilmek istenen matematiksel modeli bulabilmiş demektir. YSA modeli oluşturma süresince veriler eğitim ve test verileri olarak ikiye ayrılır; eğitim verisi ağın eğitilmesi için kullanılırken, test verisi ağın eğitim verileri dışındaki performansını ölçmede kullanılır.

3.5.1. Geriye yayılım algoritması

Geri Yayılımlı Yapay Sinir Ağları(GYYSA) ile ağ içerisinde ileri geri gezilerek en iyi ağırlıkların bulunması hedeflenir. Çalışması çıkış katmanında elde edilen hataların ağa tekrar geri uygulanarak, ağda yer alan bütün ağırlıkların güncellenmesi esasına dayanır. Yapılan iterasyonlarla elde edilen hata değeri daha önce belirlenen hata seviyesinin altına indiğinde ağ öğrenme işlemini tamamlar. Algoritma işlem adımları şu şekilde sıralanabilir.

1. Girişler ve gizli katmanlar için ağırlıklar belirlenir. Genellikle rastgele değerlerden oluşurlar
2. Öğretilecek veriler girişe uygulanır
3. İleri besleme : Girdi katmanına verilen veriler çıkış katmanına doğru iletilir.
4. İstenilen çıkış ile elde edilen çıktı karşılaştırılır ve hata değeri hesaplanır.

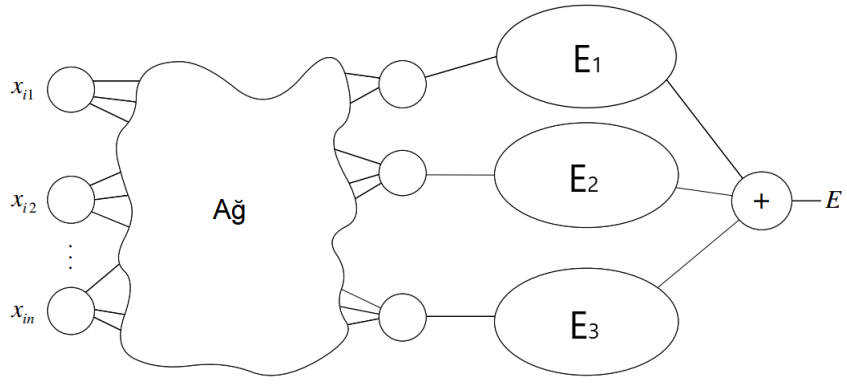
5. Hata değeri çıkış katmanından gizli katmana, gizli katmandan giriş katmanına doğru yayılır.
6. Ağırlıklar hata değerine göre yeniden düzenlenir ve 2. adıma geçilir.

Geri yayılım algoritmasının temelini oluşturulan gradyan iniş metodu konunun daha anlaşılması için önemlidir.

n tane giriş, m tane çıkış işlem birimi olan ileri beslemeli bir ağ olduğunu varsayalım. Ara katman sayısı değişkendir. Eğitim verisi $\{(x_1,t_1),(x_2,t_2),\ldots,(x_p,t_p)\}$ $x_i = [x_{i1},x_{i2},\ldots,x_{in}]$ve $t_i = [t_{i1},t_{i2},\ldots,t_{im}]$ vektörlerinden oluşmaktadır. Eğitim verisi p tane örnekten oluşmaktadır.

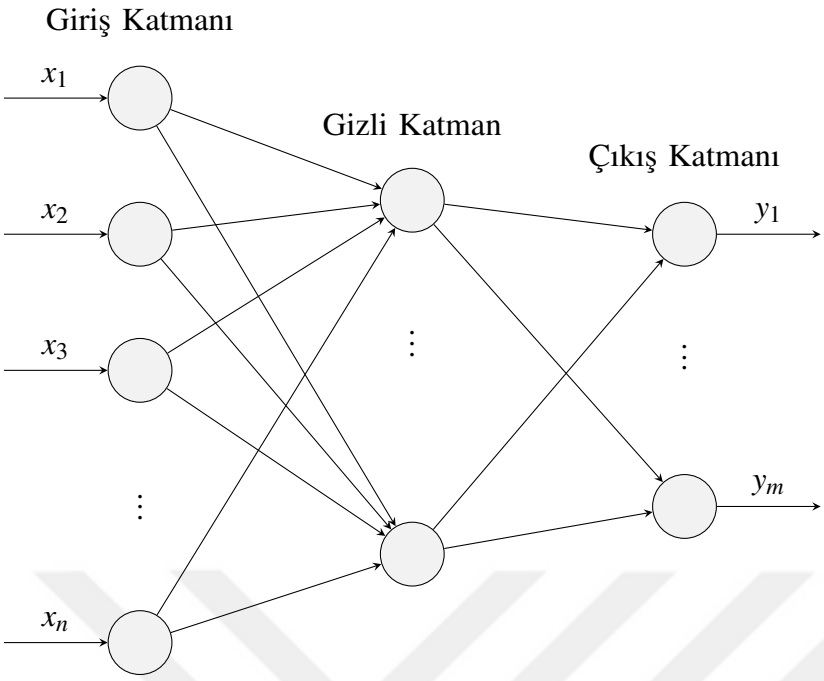
YSA'dan elde edilen çıkışların, beklenen değerlere ne kadar uzak olduklarının, gradyan iniş metodu ile minimize edilebilecek bir fonksiyon yardımıyla ölçülmesi gerekir. Geri yayılım algoritmasında, en küçük kareler ortalaması hata fonksiyonu olarak kullanılır. İleri beslemeli ağın çıktısını y, beklenen çıkış değerini t (target) olarak ifade edersek ağdaki hata değeri E, Eşitlik 3.5'de gösterildiği şekilde hesaplanır. Şekil 3.9'da bu işlemler grafiksel olarak ağ üzerinde gösterilmiştir.

$$E = \frac{1}{2}\sum_{i\in m}(t_i - y_i)^2 \tag{3.5}$$



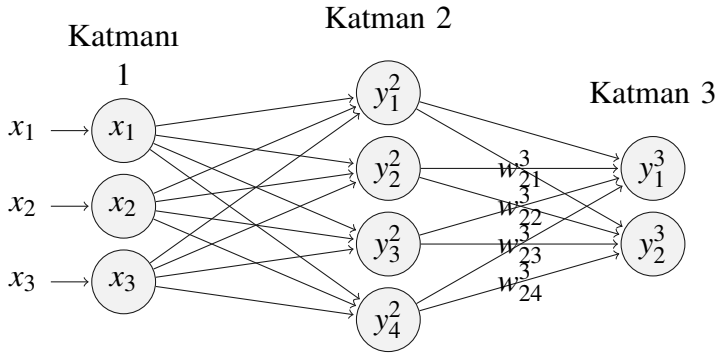
Şekil 3.9. Geriye yayılım algoritmasında hata hesaplama

Geriye yayılım algoritmasında kullanılan matematiksel denklemlerde kullanılan ve ağ üzerinde yer alan ağırlıklar, nöron çıkışı gibi parametrelerin gösterimi Şekil 3.11'de gösterilmiştir. Tek katmanlı ağlardaki gösterimden farklı olarak ağırlıkları ifade eden sembolün üzerinde katman numarasını ifade eden l belirteci kullanılmıştır. Tek katmanlı



Şekil 3.10. Çok katmanlı yapay sinir ağ yapısı

ağlarda her bir çıkış değer y_i ile gösterilirken çok katmanlı ağlarda aktivasyon fonksiyonunun uygulanması sonrasında elde edilen çıkış değeri katman numarasını da içerecek şekilde kullanılmıştır. Gösterimde yapılan bir diğer değişim ise aktivasyon fonksiyonu uygulandıktan sonra elde edilen çıkış değerini gösteren ifadenin nöron şeklinin üzerinde etiketlenmiş olmasıdır.



Şekil 3.11. Çok katmanlı yapay sinir ağı içindeki parametrelerin gösterimi

Bias ve aktivasyon fonksiyonu için aynı gösterim kullanılmıştır. Eşitliklerde bias için l'inci katmandaki j'inci nöron ifade edecek şekilde b_j^l gösterilmiştir. Aktivasyon çıkışı elde edilen değer y_j^l ile gösterilmiştir. Aynı şekilde bu ifade, l'inci katmandaki j'inci nöron için aktivasyon çıkış değerini temsil etmektedir. sigmoid (σ) fonksiyon kullanan bir nöron çıkış değeri

Eşitlik 3.6'de gösterilmiştir. Görüleceği üzere l'inci katmanda yer alan bir nöron aktivasyon fonksiyon girişi bir önceki katmanda $(l-1)$ yer alan çıkışlar ve l'inci katmana bağlı olan ağırlık değerlerini kullanarak hesaplanmaktadır.

$$y_j^l = \sigma\left(\sum_k w_{jk}^l y_k^{l-1} + b_j^l\right), \tag{3.6}$$

Eşitlik 3.6'de y^l, w^l, b^l matris şeklinde ifade edilirse Eşitlik 3.8'de gösterildiği gibi daha sade bir şekilde de ifade edilebilir.

$$y^l = \begin{bmatrix} y_1^l \\ y_2^l \\ \cdots \\ y_m^l \end{bmatrix}, w^l = \begin{bmatrix} w_1^l \\ w_2^l \\ \cdots \\ w_m^l \end{bmatrix}, b^l = \begin{bmatrix} b_1^l \\ b_2^l \\ \cdots \\ b_m^l \end{bmatrix} \tag{3.7}$$

$$y^l = \sigma((w^l)^T y^{l-1} + b^l) \tag{3.8}$$

Toplam net girdi z olarak ifade etmişkin önceki bölümlerde, aynı şekilde çok katmanlı ağlarda kat numarasını içerecek şekilde toplam net girdi değeri $z^l = w^l y^{l-1} + b^l$ ile ifade edilir. Böylece nöron çıktısı daha basit forma indirgenebilir (Eşitlik 3.9).

$$y^l = \sigma(z^l) \tag{3.9}$$

Ağırlıklara başlangıç değerleri Gauss dağılım gösteren rastgele sayılar üzerinden atanır [21]. GYYSA'da ilk adım ileriye doğru hareket ederek çıktı ya da birden fazla çıktı olması durumunda çıktılar elde etmektir. Eşitlik 3.6 yardımıyla her bir nöron için çıktı hesaplanır. Çıkış katmanında yer alan nöronlar için hesaplanan çıktı (y_j^l) ile eğitim başlamadan önce belirlenen ve olmasının beklendiği değerler (t_j) arasındaki fark hata olarak ifade edilir. Çıkış katmanında yer alan bütün nöronlar için hata değeri belirlenir. Bütün hata değerlerinin toplamı, toplam hatayı (E) ifade eder, Eşitlik 3.10 ile gösterildiği şekilde hesaplanır.

İstenen çıkış değeri t_j ile çıktı katmanında elde edilen y_j^l arasındaki farklarının toplamının l'inci katmandaki j'inci nöron için hesaplanan net girdiye göre olan kısmi türevi δ_j^l o nöron için hesaplanan hatayı ifade etmektedir ve Eşitlik 3.14 ile gösterildiği şekilde hesaplanır.

$$E = \frac{1}{2}\sum_j (t_j - y_j^l)^2 \tag{3.10}$$

$$\delta_j^l \equiv \frac{\partial E}{\partial z_j^l} \tag{3.11}$$

Eşitlik 3.11'de gösterilen ifade Eşitlik 3.12'de gösterildiği şekilde zincir kuralı uygulanıp yazılırsa hata değeri Eşitlik 3.14'de gösterildiği şekilde hesaplanır.

$$\delta_j^l = \frac{\partial E}{\partial y_j^l}\frac{\partial y_j^l}{\partial z_j^l} \tag{3.12}$$

$$\sigma'(z_j^l) = \frac{\partial y_k^l}{\partial z_j^l} \tag{3.13}$$

Çıkış katmanında doğrusal aktivasyon kullanılması durumunda kısmi türev hesaplamasından hata değeri çıkış katmandaki nöronlar için $y_j - t_j$ olarak hesaplanır.

$$\delta_j^l = \frac{\partial E}{\partial y_j^l}\sigma'(z_j^l) \tag{3.14}$$

Algoritmanın bir sonraki adımında çıkış katmanında hesaplanan hata değeri geriye doğru yayılarak ilerler. Bu durumda δ_j^l değerini bir sonraki katmanda hesaplanan hata değerleri δ_j^{l+1} içerecek şekilde yeni bir denklem yazmamız gerekiyor. Eşitlik 3.11'e zincir kuralı uygulanırsa δ_j^l değeri Eşitlik 3.15 ile gösterildiği şekilde bir sonraki katmandaki hatayı içerecek şekilde yeniden yazılır.

$$\begin{aligned} \delta_j^l &= \tfrac{\partial E}{\partial z_j^l} \\ &= \textstyle\sum_k \frac{\partial E}{\partial z_k^{l+1}}\frac{\partial z_k^{l+1}}{\partial z_j^l} \\ &= \textstyle\sum_k \frac{\partial z_k^{l+1}}{\partial z_j^l}\delta_k^{l+1} \end{aligned} \tag{3.15}$$

Eşitlik 3.15'in son teriminde yer alan z_k^{l+1} ifadesi daha açık bir formda Eşitlik 3.16'de gösterildiği şekilde hesaplanır.

$$z_k^{l+1} = \sum_j w_{kj}^{l+1} y_j^l + b_k^{l+1} = \sum_j w_{kj}^{l+1}\sigma(z_j^l) + b_k^{l+1}. \tag{3.16}$$

z_k^{l+1} değerinin z_j^l 'ye göre türevi Eşitlik 3.17 ile gösterildiği şekilde hesaplanır.

$$\frac{\partial z_k^{l+1}}{\partial z_j^l} = w_{kj}^{l+1}\sigma'(z_j^l) \tag{3.17}$$

Eşitlik 3.17 ile ifade edilen bölüm, Eşitlik 3.15'te yerine yazılırsa, en genel ifade ile bir nöron için hata değeri Eşitlik 3.18 ile gösterildiği şekilde hesaplanır.

$$\delta_j^l = \sum_k w_{kj}^{l+1} \delta_k^{l+1} \sigma'(z_j^l) \tag{3.18}$$

Her düğüm için hata değerleri hesaplandıktan sonra ağırlıklar Eşitlik 3.21 ile gösterildiği şekilde güncellenir.

$$\frac{\partial E}{\partial w_{kj}^l} = y_k^{l-1} \delta_j^l \tag{3.19}$$

$$\frac{\partial E}{\partial b_j^l} = \delta_j^l \tag{3.20}$$

$$w'_{kj} = w_{kj} - \alpha \frac{\partial E}{\partial w_{kj}} \tag{3.21}$$

$$\Delta w_{kj}(i) = -\alpha \frac{\partial E}{\partial w_{kj}} + \mu \Delta w_{kj}(i-1) \tag{3.22}$$

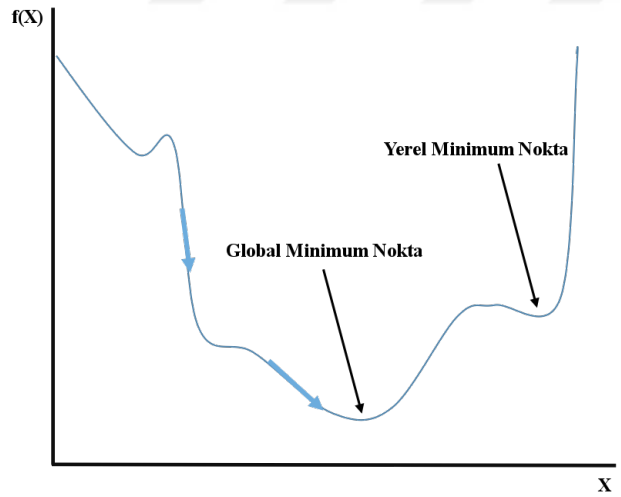
Eşitlik 3.22'de α öğrenme katsayısı, μ momentum katsayısı ve $\frac{\partial E}{\partial w_{kj}}$ katmanlar arasındaki ağırlığın değişiminin E hata üzerindeki değişimini ifade etmektedir.

Eşitlik 3.22'de gösterilen öğrenme katsayısı (α) istenilen sonuca ulaşmak için gerekli olan iterasyon sayısını değiştirmektedir. α değeri $[0-1]$ aralığında seçilir. Sıfıra yakın seçilmesi durumunda minimum hata seviyesine ulaşmak için gerekli olan iterasyon sayısı fazla olacaktır. Yakınsamanın yavaş olması durumunda bu katsayı artırılır. Katsayının büyük olarak seçilmesi durumunda ağın öğrenmesi gerçekleşmeyebilir. Eğitim verileri öğretimi sonunda iyi sonuçların alınması yanıltıcı olabilir. Test verilerinde elde edilen kötü sonuç ağın öğretilmediği ezberletildiği sonucunu çıkarır. Böyle bir durumda öğrenme katsayısının küçültülmesi ve tekrar denenmesi gerekir. Bu tür sorunu aşmak için adaptif yöntemler önerilmiştir. Çözümden uzak bölgelerde değer büyük, çözüm noktası civarında ise katsayı küçük seçilerek gerekli olan iterasyon sayısı optimum seviye olması sağlanır.

Yerel minimum noktalarda takılmayı önlemek için momentum katsayısı(μ) kullanılır. Momentum katsayısı genellikle $[0-1]$ aralığında seçilir. Bu katsayı, yerel minimum noktalarında takılmayı önlemektedir. Ağırlıklarda gerçekleşen önceki değişim momentum

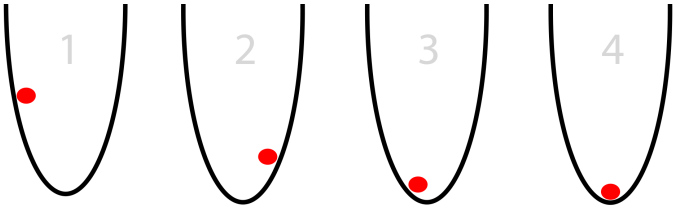
katsayısı ile çarpılır ve yeni ağırlığa etki olarak eklenir. Momentum değeri büyük seçildiğinde daha iyi sonuçlar alındığı gözlenmiştir.

Öğrenme katsayısı nasıl belirlenmesi gerektiği konusunun iyi anlaşılması için gradyan iniş yöntemini örnekler üzerinde inceleyelim. Türev, bir fonksiyonun bir değişkeninde meydana gelen sonsuz küçüklükteki bir değişimin fonksiyonun değerinde meydana getireceği artış ya da azalışın o değişkendeki değişime olan oranıdır. Tek değişkenli bir fonksiyonun bir noktadaki türevi, fonksiyon grafiğinde o noktaya teğet olan doğrunun eğimi olarak ifade edilir. Bir fonksiyonun ne yönde arttığını anlamak için eğim bilgisi kullanılır. Eğer eğim 0 ise fonksiyon grafiğinin yönünün değişmesi söz konusu olabilir. Örneğin $y = f(x)$ eğrisinin rastgele bir noktası üzerinde durduğumuzu ve fonksiyon grafiğindeki en dipte bulunan noktaya gitmeye çalıştığımızı hayal edelim. Bulunduğunuz noktada fonksiyonun türevinin tersi yönde hareket edersek dibe, türev yönünde hareket edersek tepeye çıkabilirsiniz. Daha genel ifadeyle, gradyan iniş optimizasyon yöntemi, bir fonksiyon üzerindeki bir noktadan başlayarak türevinin tersi yönünde bu noktayı hareket ettirerek global minimum noktaya (Şekil 3.12) yaklaşma amaçlanır.

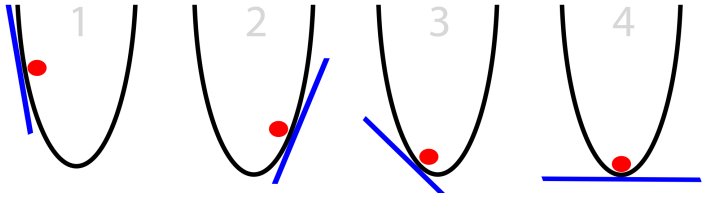


Şekil 3.12. Yerel ve global minimum noktalar

Bir hata fonksiyonunda noktaların bulunabileceği konumları Şekil 3.13 ile gösterildiği şekilde bir eğri üzerende hareket eden bir topa benzeterek anlatalım. Eğri üzerindeki top ancak eğimi sıfır olunca duracaktır, sıfıra yaklaşmak için topun bulunduğu yerin eğim bilgisi kullanılarak yaklaşma sağlanır (Şekil 3.13). Şekil 3.14'de topun eğimi çizgiler ile gösterilmiştir.



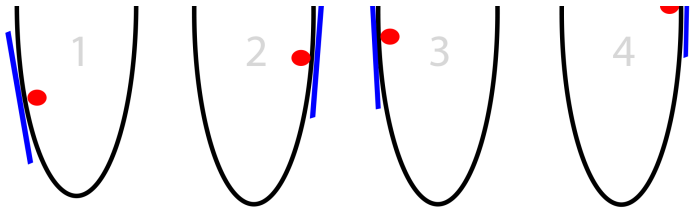
Şekil 3.13. Gradyan iniş için örnek gösterim



Şekil 3.14. Gradyan iniş yönteminde optimum sonucun bulunması

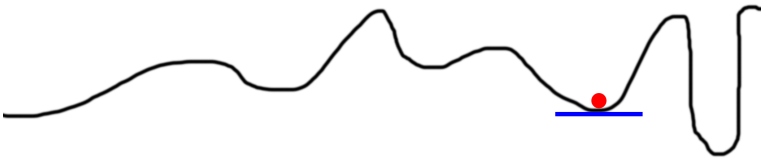
Algoritma çalıştırılırken bazı sorunlar ile karşılaşılabilir: Hesaplanan eğim çok büyük olabilir (Şekil 3.15). Eğim büyük olması durumunda, ilerleme adımlarını küçültmek gerekir. Adımların küçük olabilmesini sağlamak için, 0 ile 1 değeri arasında bir katsayı ile eğim çarpılır; böylece eğim çok büyük olsa bile adım küçük olacaktır. Bu katsayıya öğrenme katsayısı denmektedir, α ile gösterilir. Algoritma şu şekilde iyileştirilir:

- x'in eğimini hesapla
- x değerini eğimin tersi yönde değiştir. $(x = x - \alpha * f'(x))$
- Eğim 0 olana kadar yukarıdaki adımları tekrarla

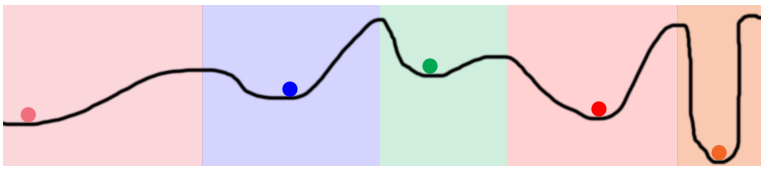


Şekil 3.15. Gradyan iniş yönteminde bulunulan noktanın türevinin yüksek olması durumu

Karşılaşılabilecek bir diğer sorun eğrinin şeklinden kaynaklıdır. Burada global minimum bulamama ile karşı karşıya kalınabilir (Şekil 3.16). Bu konudaki çözüm rastgele noktalar seçerek farklı alanları tarama yöntemi ile aşılabilir (Şekil 3.17). Eğimin çok küçük olması durumunda ise α değerinin artırılması gerekir (Şekil 3.18). Gradyan iniş algoritmasının nasıl çalıştığını ve etkilerini görebilmek hazırlanan basit bir kod örneği EK-6 ile sunulmuştur.



Şekil 3.16. Gradyan iniş yönteminde global minimumu bulma sorunu



Şekil 3.17. Gradyan iniş yönteminde rastgele noktaları tarayarak arama

Şekil 3.18. Gradyan iniş yönteminde bulunulan noktanın türevinin düşük olma durumu

4 . YÖNTEMLER

4.1. Açısal Hız Hesaplanmasında Kullanılan Diferansiyel Denklem

İSS'de kullanılan Eşitlik 4.2 ile gösterilen açısal hızın hesaplanmasında kullanılan birinci mertebeden lineer olmayan diferansiyel denklem artan *G* komutları için iyi bir açısal hız çözümü sunarken; azalan *G* komutları için çözüm sağlayamamaktadır. Kullanılan diferansiyel denklem azalan *G* komutları için kullanılamadığından dolayı farklı yöntemler araştırılmıştır. Bu çalışmada diferansiyel denklemin neden çözümsüz olduğu varlık teklik konusu üzerinden açıklanmaya çalışılmıştır.

Kullanılan denklem birinci mertebeden doğrusal olmayan Adi Diferansiyel Denklem (ADD) türündendir. Doğrusal olmayan denklem sınıfından diferansiyel denklemlerin çözümü doğrusal olan denklemlere göre daha zordur. Bazı doğrusal olmayan denklemleri çözebilmek için doğrusal olan yapıdaki diferansiyel denklemlere indirgenir (Bernoulli Diferansiyel Denklemi, Riccati Diferansiyel Denklemi); ancak bu açısal hız hesaplanmasında kullanılan denklem bu türlere dönüşememektedir.

Farklı değerler alabilen niceliklere değişken denir. Denklemlerde bağımlı ve bağımsız değişkenler olmak üzere iki tür değişkenden bahsedilir. Bağımlı değişkenler, bağımsız değişkenlerin değişimden dolayı meydana gelen değişimi ifade eder. Bir değişkenin başka bir değişkene göre değişme hızına türev; bağımlı ve bağımsız değişkenler ve bunların türevleri arasındaki bir bağıntıya da diferansiyel denklem (DD) denir. DD'de bir tek bağımsız değişken varsa denkleme Adi Diferansiyel Denklem (ADD) denir.

Bir DD içinde bulunan en yüksek mertebeli türevin mertebesine diferansiyel denklemin mertebesi; en yüksek mertebeli türevinin derecesine diferansiyel denklemin derecesi denir. Bir diferansiyel denklemin çözümü; genel çözüm, özel çözüm ve tekil çözüm olmak üzere üçe ayrılır. *n*'inci mertebeden bir ADD'nin *n* tane keyfi sabit içeren çözümüne genel çözüm denir. Elde edilen genel çözümdeki n keyfi sabite belli değerler verilmek suretiyle elde edilen çözüme özel çözüm denir. Genel çözümdeki *n* keyfi sabitin herhangi bir şekilde seçimi ile elde edilemeyen çözümüne ise tekil çözüm denir.

Bir ADD eğer koşullar bağımlı değişken ve türevlerine göre tek bir noktada verilmişse başlangıç değer problemi; eğer koşullar en az farklı iki noktada tanımlanmışsa sınır değer problemi olarak çözülür. Bu bilgiler ışığında bu çalışmaya konu olan denklem (Eşitlik 4.2) incelenirse: Eşitliğin sadece $t = 0$ anındaki durumu bilindiğinden bu denklem başlangıç değer problemidir. Diferansiyel denklemin bir çözümü olup olmadığı konusunda ilk yapılan inceleme varlık ve teklik durumlarını incelemektir. t zaman değerini göstermekte olup ve bağımsız bir değişkendir. Eşitlik 2.11 kullanılarak açısal hız hesaplaması için Eşitlik 4.2 ile gösterilen diferansiyel denklem oluşturulmuştur.

$$H(t) = (G(t)^2 - 1)(\frac{g}{R})^2 \tag{4.1}$$

$$\omega(t)' = f(t,\omega) = \pm\sqrt{|H(t) - \omega(t)^4|} \tag{4.2}$$

Eşitliklerde g yer çekimi ivmesini, R sistemin ana kol yarı çapını, $G(t)$ t zamanında uygulanmak istenen G kuvvetini, ω açısal hızı, $omega'$ birim zamanda açısal hız değişimini temsil etmektedir. Eşitlik 4.2'de g ve R değerleri sabittir. Bu çalışmada sabit olan bu değerler sırasıyla $g = 9.81m/sn^2$ ve $R = 7.62m$ olarak alınmıştır. Picard varlık ve teklik teoremi, ω hesaplamasında neden diferansiyel denklemin başarısız olduğunu göstermek için kullanılabilir [19]. Teorem bize aşağıdaki soruların cevaplarını verir.

- Hangi koşullar altında Eşitlik 4.3'ün bir çözümü vardır.
- Hangi koşullar altında Eşitlik 4.3'ün tekil çözümü vardır.

Eşitlik 4.2 aşağıda gösterilen Eşitlik 4.3 formunda yazalım.

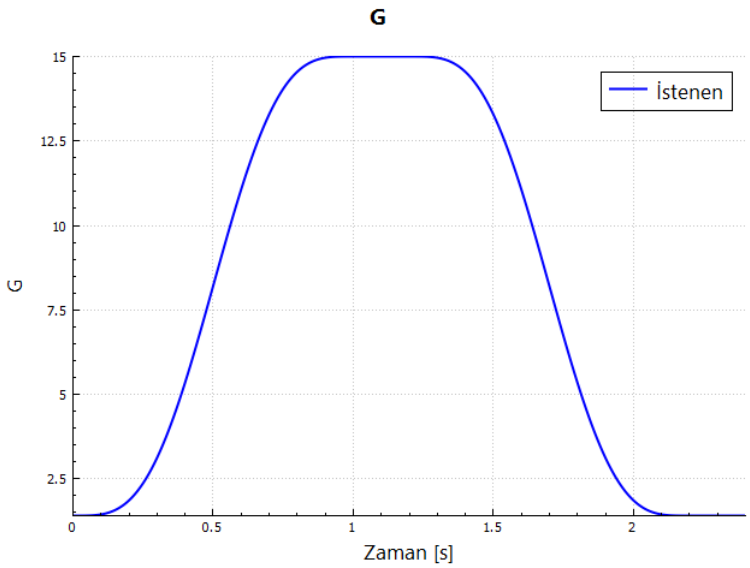
$$\omega_0 = \omega(t = 0) = \sqrt[4]{H(t)} \tag{4.3}$$

Eğer $f(t,\omega)$ ve $\frac{dF}{dw}$, $R = (t,\omega) : t_0 - a < t < t_0 + a, w_0 - b < w < w_0 + b$ dikdörtgensel bir bölgesinde sürekliyse ve aynı bölgedeki bir $(t = 0, w_0)$ noktasından geçiyorsa, diğer bir ifadeyle $w(t = 0) = w_0$ mevcut ise, bu durumda diferansiyel denklem $(t = 0, w_0)$ noktasını içerisinde bulunduran R'nin alt bölgesinde en az bir çözüme sahiptir. Ayrıca $\frac{df}{dw}$ türevi de bu R bölgesinde sürekli bir fonksiyon ise elde edilecek çözüm tektir. Bu ifade Eşitlik 4.4'da verilmiştir.

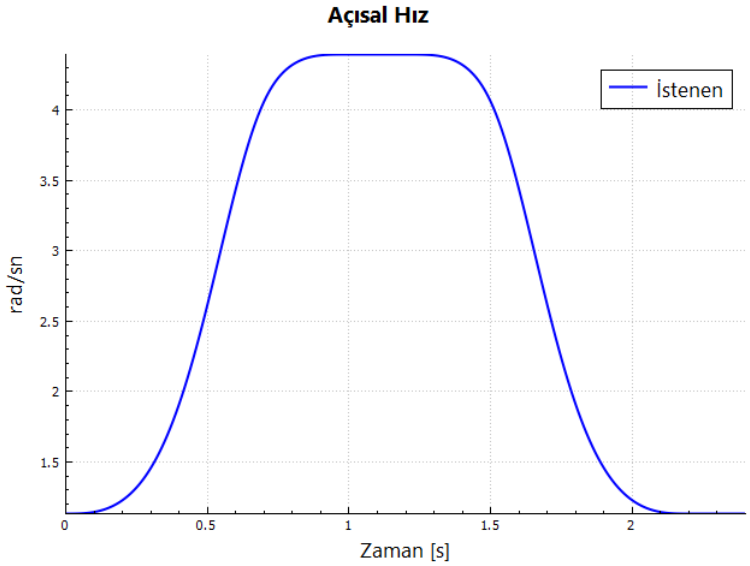
$$\frac{f(t,w)}{\partial w} = \frac{\partial\sqrt{H-w^4}}{\partial w} = \frac{-2w^3}{\sqrt{H-w^4}} \tag{4.4}$$

Eşitlik 4.4'da açıkça görüldüğü gibi $H \neq w^4$ olduğu durumlarda süreklidir. $w' = 0$ olduğu durumda süreklilik bozulur. Süreklik olmadığından teklik yoktur.

Problemi daha iyi anlatabilmek için bazı grafikler aşağıda gösterilmiştir. Şekil 4.1'de uygulanmak istenen G komutu grafiği gösterilmiştir. Bu grafikteki G komutunun elde edilebilmesi için sistemde uygulanması gereken açısal hız grafiği Şekil 4.2'de gösterilmiştir.



Şekil 4.1. Artan ve azalan G komut dizisi grafiği



Şekil 4.2. Açısal hız grafiği

Açısal hız verisi artan *G* komutları için polinom denklem köklerinin bulunması yöntemi elde edilebiliyor; ancak azalan *G* komutları için kökler bulunamadığından kesin bir çözüm elde edilemiyor. Şu varsayımdan yola çıkarak azalan *G* grafiğine en yakın grafik bulunabilir. Azalan *G* komutları ters çevrilerek artan *G* komutları dizisi elde edilir ve işlem bu artan değerlere göre yapılarak açısal hız dizisi oluşturulur. Elde edilen dizi tekrar ters çevrilerek istenen açısal hız komutları azalan *G* komutları için elde edilir. Şekil 4.3 ile bu algoritmanın akış diyagramı gösterilmiştir.

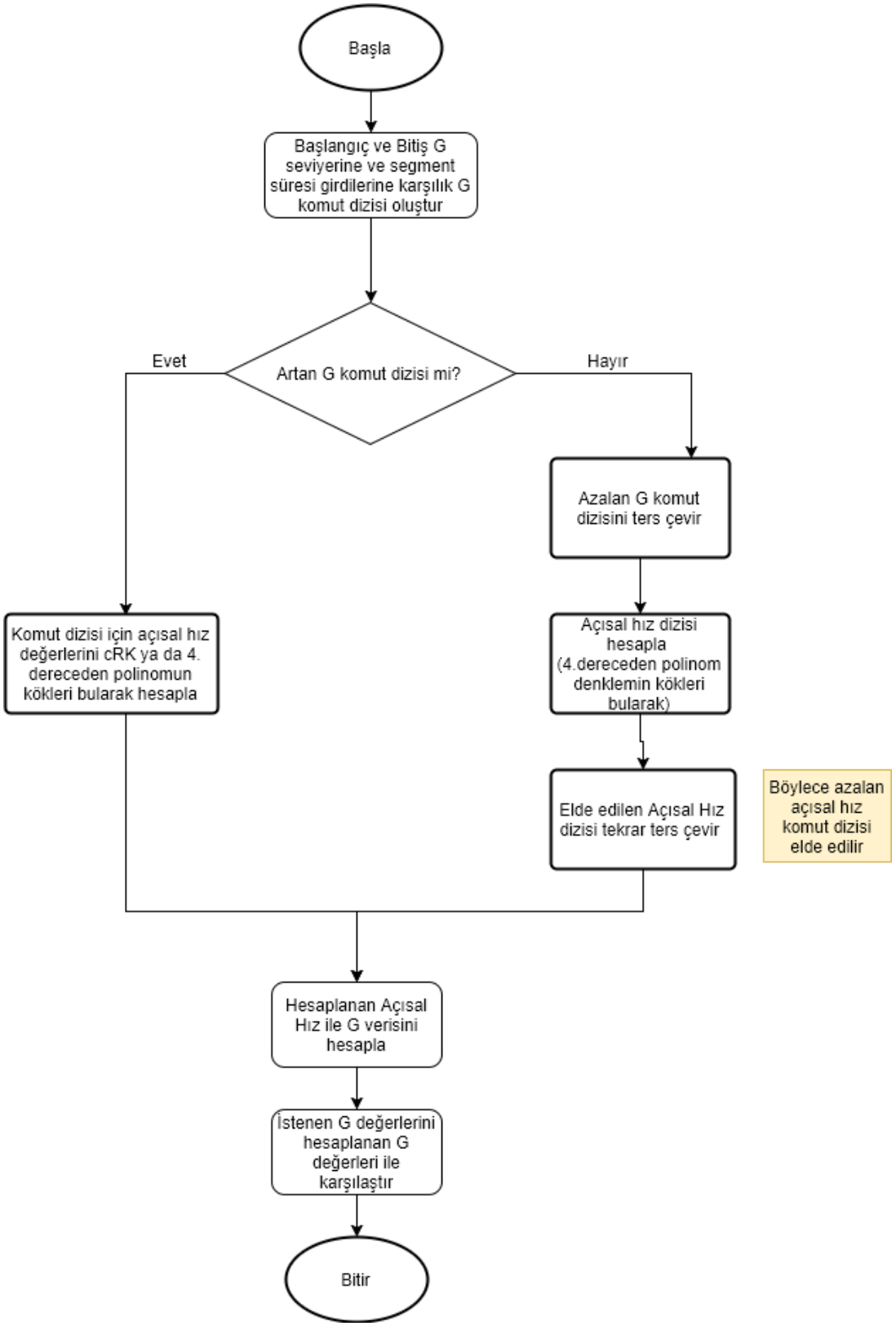
Polinom denklemin bu komutlar için nerede çözümsüz olduğu Şekil 4.4 ve Şekil 4.5'te gösterilmiştir. Şekillerde taralı bölge en az bir kökün istenen açısal hız için çözüm olduğu noktaları; taralı olmayan bölge ise polinom denklemin bütün köklerinin karmaşık sayılardan oluştuğu noktaları göstermektedir. Şekillerden rahatlıkla görüleceği gibi, artan *G* komutları için düzgün sonuçlar vermekte ancak azalan *G* komutları için çözüm bulamamaktadır.

Literatürde bu konuda yapılan çalışmalarda [1, 2], açısal hız hesaplamasında cRK yöntemi önerilmiştir. Bu yöntemin artan *G* komutları için en iyi yöntem olduğu belirtilmiştir. Bu çalışmada 4. dereceden polinom denklemden kök bulma yönteminin de cRK'ya alternatif olarak kullanılabileceği önerilmiştir.

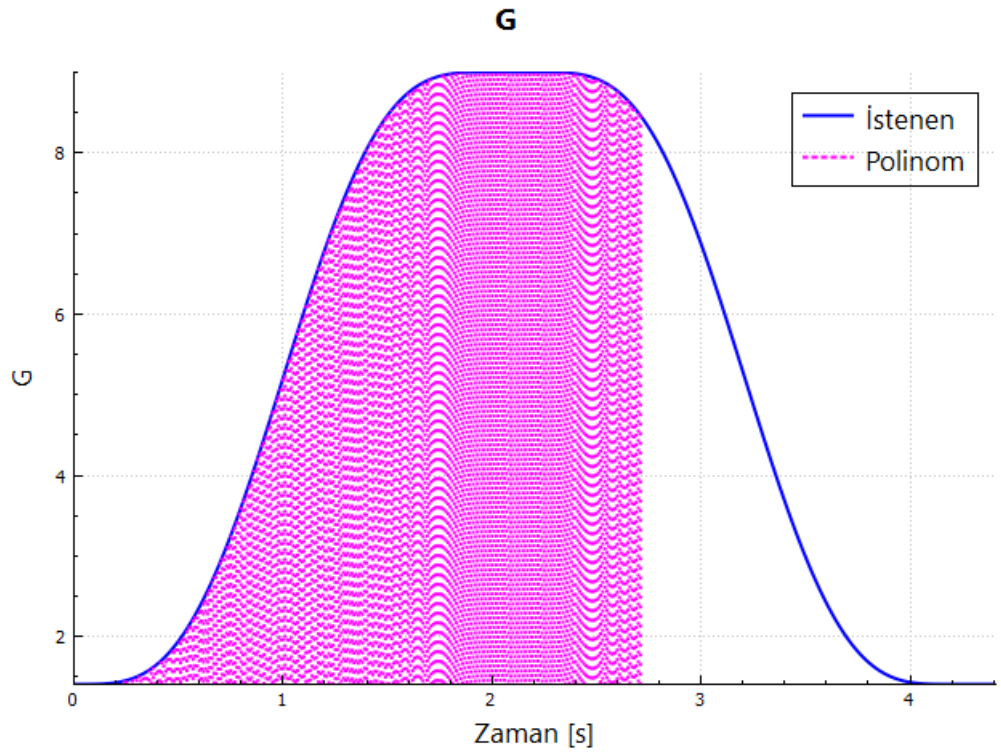
Şekil 4.6 ve Şekil 4.7'de cRK yönteminin artan ve azalan *G* komutları ne oranda çözebildiği gösterilmiştir. Şekiller artan *G* komutları için kabul edilebilir sonuçlar elde edebildiğimizi göstermekle birlikte azalan *G* komutları için sonuçların istenilen düzeyde olmadığı sonucu çıkarılabilir.

Bu sistemlerde *G* komutunu uygulayabilmek için gerekli olan açısal hız bilgisi Eşitlik 4.2 ile gösterilen diferansiyel denkleminden elde edilir. Diferansiyel denklemin artan *G* komutları için bir çözümü olmasına rağmen azalan *G* komutları için çözümü yoktur. Bu çalışmanın amacı açık çevrim profillerin yanında kapalı çevrim profiller için de genel bir çözüm bulmak ya da var olan çözümleri iyileştirmektir.

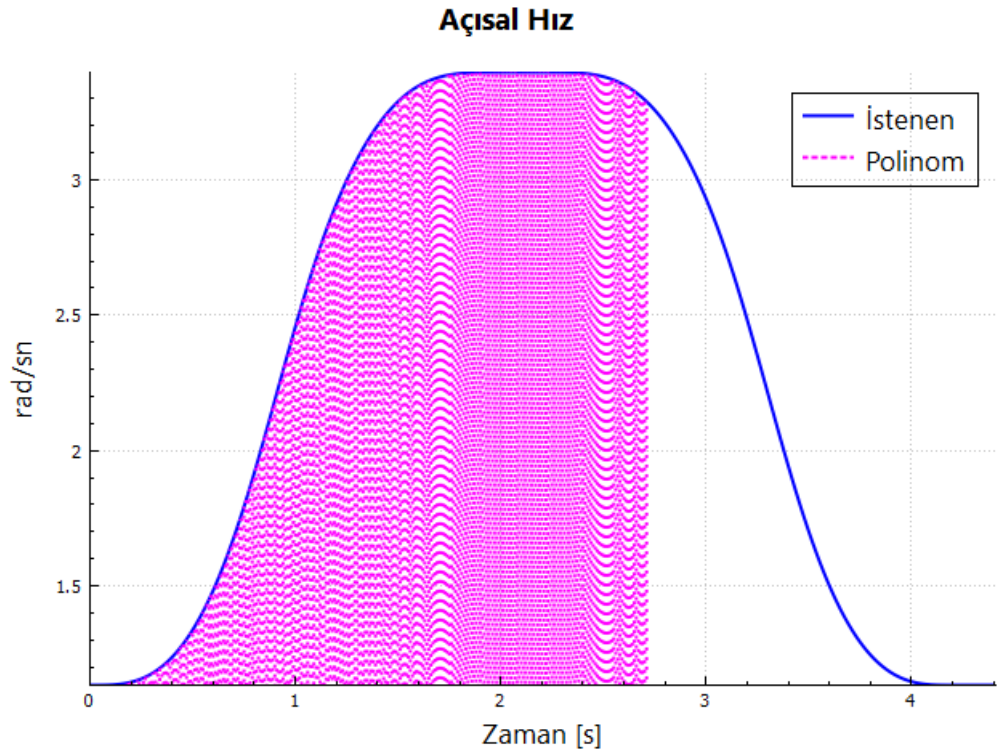
Eşitlik 4.5 ile gösterilen 4. dereceden polinom denklemin, iki gerçel ve iki kompleks kök elde edilmektedir (Şekil 4.8). İki kompleks kök ihmal edildiğinde ise geriye kalan gerçel köklerden biri açısal hız olarak kullanılabilir. Bu denklem çözümü artan *G* komutları için iyi



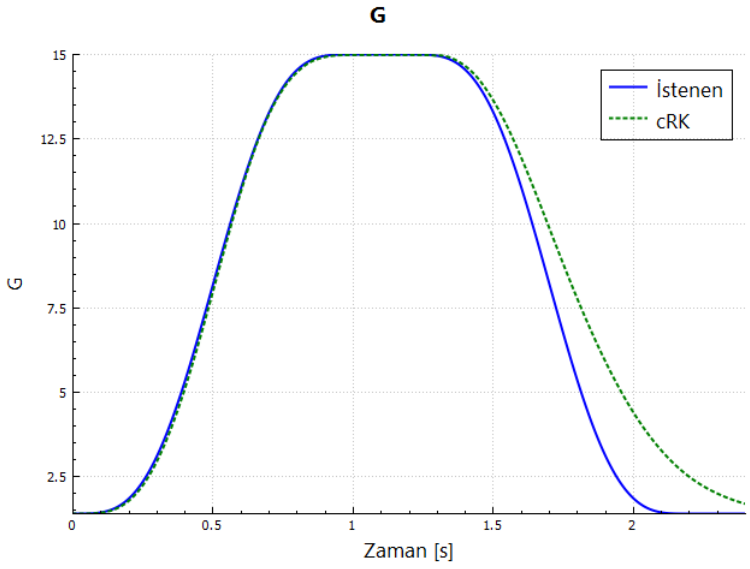
Şekil 4.3. Azalan *G* komutlarının ters çevrilerek azalan açısal hız dizisi hesaplanmasında kullanılan algoritmanın akış diyagramı



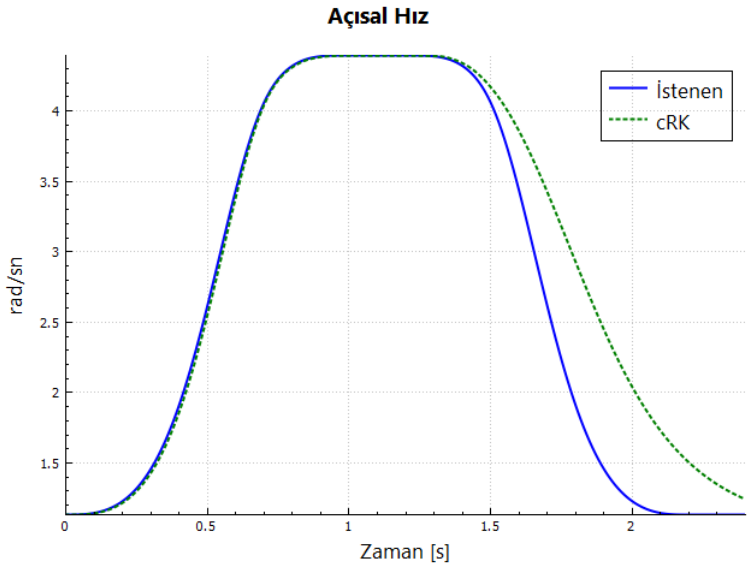
Şekil 4.4. Polinom kök bulma yöntemi ile elde edilen açısal hız verisinden *G* komut dizisinin hesaplanması



Şekil 4.5. Polinom kök bulma yöntemi ile elde edilen açısal hız



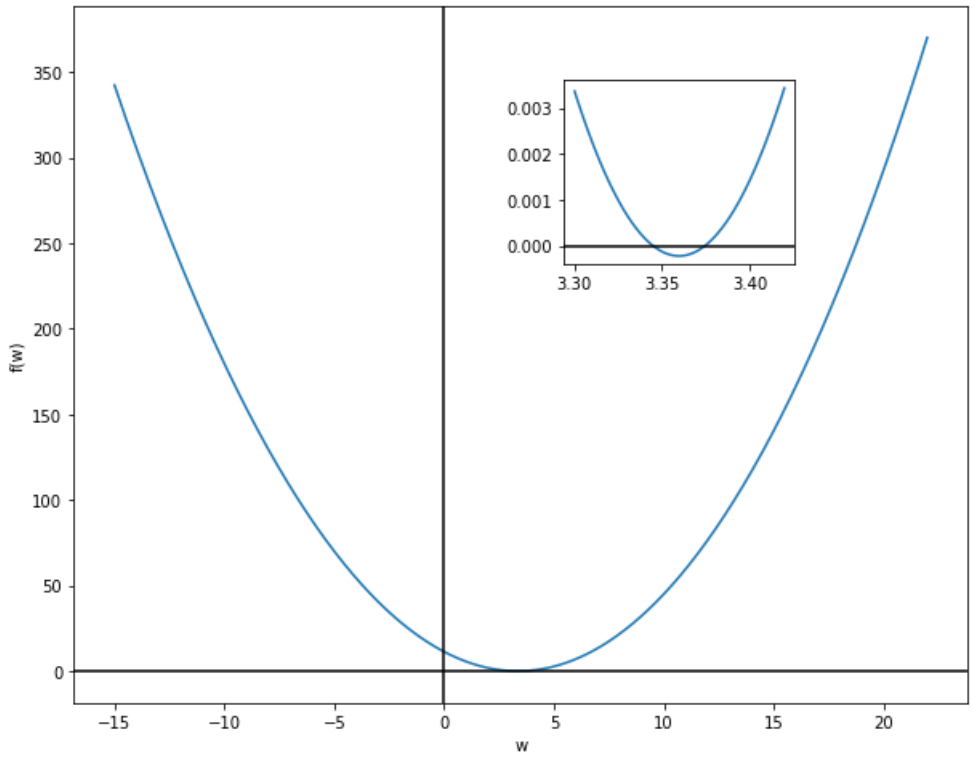
Şekil 4.6. cRK yöntemi ile elde edilen açısal hız verisinden hesaplanan *G* verisi



Şekil 4.7. cRK yöntemi ile elde edilen açısal hız

sonuçlar üretmektedir; ancak azalan *G* komutları için köklerin dördünün de kompleks kök olma durumu oluşmaktadır. Bundan dolayı bu yöntem azalan *G* komutları için kullanılamaz. Şekil 4.9 ile azalan *G* komutu için oluşturulan bir polinom denklem gösterilmektedir. Şekle dikkat edilirse *w* ekseninin hiç bir nokta kesilmediği rahatlıkla görülebilir. Ekseni herhangi bir noktada kesmemesinin anlamı bu denklemin yalnızca karmaşık kökleri vardır, reel kökü yoktur. Unutulmaması gereken en önemli ayrıntı Eşitlik 4.5 ile ifade edilen polinom denklem *G* komutu değişmesine bağlı olarak katsayıları her zaman çerçevesi için farklı olur. Bir segment 100 zaman diliminden oluşuyorsa açısal hızı çözmek için 100 polinom denklem

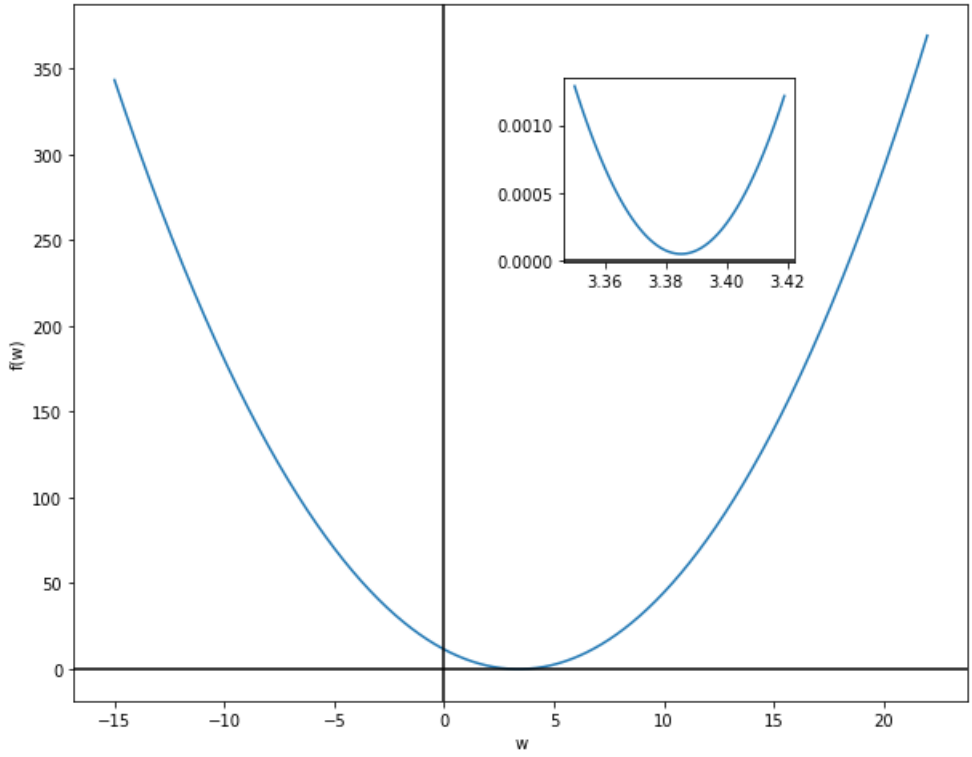
çözülmesi gerekir. Bu çalışmada yapılan deneylerde artan G komutları için oluşturulan polinom denklemlerinin her zaman w eksenini (Şekil 4.8) en az iki noktada kestiği görülmüştür. Azalan G komutları için polinom grafiği çizildiğinde w eksenin hiç kesilmediği durumlar gözlenmiştir.



Şekil 4.8. Artan G komutu için oluşturulan polinom grafiği

$$\triangle t^2\omega^4 + \omega^2 - 2\omega_{t-1}\omega + \omega_{t-1}^2 - H\triangle t^2 = 0 \tag{4.5}$$

Açısal hızın çözümü bu sistemde uygulanan eğitimin amaçlarının yerine getirilmesi için büyük öneme sahiptir. Uygulanan G_z kuvveti ne kadar doğru şekilde hesaplanırsa, G_x ve G_y kuvvetleri de o kadar küçük olur. G_x ve G_y kuvvetleri eğitim sırasında uygulanmak istenmeyen kuvvetlerdir. Bu çalışma açısal hızın çözümü konusunda daha önce yapılan çalışmaları tekrar edip, bu çalışmada yeni önerilen yöntemlerin karşılaştırılmasına olanak vermektedir. Açısal hız çözümünde kullanılan yöntemler Bölüm 4'te anlatılmıştır.



Şekil 4.9. Azalan G komutu için oluşturulan polinom grafiği

4.2. Fourier dönüşümü

Fourier dönüşümü, işaret işleme alanında ilk ortaya atılan dönüşümdür. Bir sinyal, farklı genlik, frekans ve fazlarda kosinüs ve sinüs temel bileşenlerinin toplamı olarak ifade edilir. Bu dönüşüm sayesinde sinyallerin spektrumları analiz edilebilmekte ve frekans alanında sistemlerin ve özellikleri belilenebilmektedir. Sürekli Fourier Dönüşümü ve Ayrık Fourier Dönüşümü sırasıyla Denklem 4.6 ve Denklem 4.7 ile elde edilir [10].

$$X(w) = F(x(t)) = \int_{-\infty}^{\infty} x(t)\mathrm{e}^{-jwt}\,\mathrm{d}t \tag{4.6}$$

$$X(e^{jn}) == \sum_{-\infty}^{\infty} x[n]\mathrm{e}^{-jwn}\,\mathrm{d}t \tag{4.7}$$

Denklem 4.7'de $x[n]$ gerçek bir ayrık-zamanlı işareti ifade eder. Sürekli Fourier Dönüşümü denkleminde $x(t)$ işareti, tüm zaman aralığında kompleks çarpan ile çarpılıp toplanmaktadır. Sonuç olarak dönüşüm, $X(w)$ Fourier katsayılarını vermektedir.

Fourier dönüşüm diferansiyel denklemlerin çözümünde kullanılmaktadır. Bu dönüşüm

sayesinde zaman uzayındaki diferansiyel denklemler frekans uzayında lineer denklemler olarak ifade edilirler. Frekans uzayında bir denklemin çözümü bulunduktan sonra ters dönüşümle zaman uzayındaki karşılığı elde edilir ki bu diferansiyel denklemin çözümüdür.

Bu yöntemin uygulanması farklı bir araştırma konusu olduğundan dolayı bu bölümde yalnızca ileride yapılabilecek açısal hızın hesaplanması için kullanılan diferansiyel denklem konusunda yardımcı bilgiler verilmiştir. Öncelikle diferansiyel denklem heterojen türde bir denklem olduğundan karakteristik denklemi yazılarak çözüm bulunamaz. Bölüm 4.1'de diferansiyel denklem incelenmiş ve varlık teklik açısından bakıldığında bir çözümün varlığı mevcut olduğu ancak tekliği sağlamadığı görülmüştür.

4.3. Geri Euler Yöntemi (Backward Euler)

Euler yöntemi Taylor-serisinin özel bir halidir. Taylor-serisi, bir fonksiyonun, o fonksiyona ait terimlerin tek bir noktadaki türevlerinden hesaplanan sonsuz toplamı şeklinde yazılması ile elde edilir. Eğer Taylor-serisi sıfır merkezli ise daha basit bir formda yazılır ve bu özel seriye Maclaurin serisi denir.

Her dereceden türevli bir $f(x)$ fonksiyonunun a gerçel ya da karmaşık bir sayı olmak üzere a noktası etrafındaki yaklaşık değerini veren polinom fonksiyon için Taylor-serisi şu şekilde oluşturulur:

$$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \ldots + \frac{f^{(n)}(a)}{n!}(x-a)^n + \ldots \tag{4.8}$$

Yukarıdaki denklem daha genel bir ifade ile aşağıdaki gibi yazılabilir.

$$\sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n \tag{4.9}$$

Burada $n!$, n faktöriyeli; $f^{(n)}(a)$ ise f fonksiyonunun n. dereceden türevinin a noktasındaki değerini belirtmektedir. f fonksiyonunun sıfırıncı dereceden türevi f'nin kendisiyle tanımlanmıştır.

$a = 0$ olarak belirlendiğinde elde edilen Maclaurin serisi aşağıdaki şekilde ifade edilir:

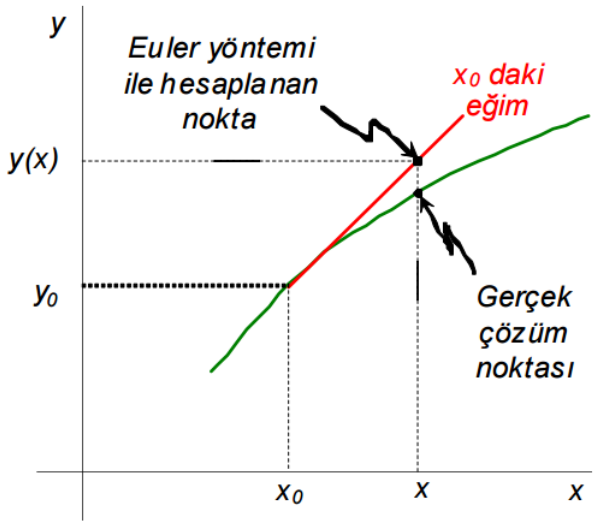
$$f(0)+f'(0)x+\frac{f''(0)}{2!}x^2+\frac{f^{(3)}(0)}{3!}x^3+\cdots \tag{4.10}$$

Euler yöntemi, Taylor-serisinin sadece birinci dereceden terimini kullanan bir yöntemdir.

$$f(x)=f(a)+\frac{f'(a)}{1!}(x-a) \tag{4.11}$$

$$f'(a)=\frac{f(x)-f(a)}{x-a} \tag{4.12}$$

Şekil 4.10'da Euler yönteminin uygulanışı ve çözümde olası hata olma durumu gösterilmektedir.



Şekil 4.10. Euler Yöntemi

4.4. Klasik Runge-Kutta Yöntemi

Adi diferansiyel denklemler'in çözümünde en çok bilinen yöntemlerden biri olan Runge-Kutta yöntemi 1900'lü yllarda Alman matematikçiler C. Runge ve M.W. Kutta tarafından geliştirilmiştir [9, 11, 51].

Aşağıdaki gibi tanımlanan bir başlangıç değer problemini ele alalım.

$$y'=f(t,y), \qquad y(t_0)=y_0 \tag{4.13}$$

Runge-Kutta metodu aşağıdaki formda yazılabilir-.

$$y_{n+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{4.14}$$

$\frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$ artım fonksiyonu, diğer bir deyişle, o zaman aralığındaki eğimi ifade eder. Eşitlikdeki k_1, k_2, k_3, k_4 değerleri aşağıda gösterildiği gibi hesaplanır.

$$k_1 = hf(t_n, y_n) \tag{4.15}$$

$$k_2 = hf(t_n + \frac{h}{2}, y_n + \frac{k_1}{2}) \tag{4.16}$$

$$k_3 = hf(t_n + \frac{h}{2}, y_n + \frac{k_2}{2}) \tag{4.17}$$

$$k_4 = hf(t_n + h, y_n + k_3) \tag{4.18}$$

- k_1 : h aralığının başlangıcındaki eğimdir.
- k_2 : h aralığının orta noktasındaki eğimdir. y_n değerine, Euler Yöntemi kullanılarak elde edilen eğim k_1 eklenir.
- k_3 : h aralığın orta noktası için hesaplanan başka bir eğimdir. Fonksiyona parametre olarak verilen y_n değerine k_2 eğimi eklenir.
- k_4 : h aralığının sonundaki eğimdir. y_n değerine k_3 eğimi eklenir.

y_{n+1} değeri o anki y_n değerine h aralığının büyüklüğüyle tahmini eğimin çarpımının eklenmesiyle elde edilir. Artım fonksiyonu ile elde edilen eğim, k_1, k_2, k_3, k_4 eğimlerin ağırlıklı ortalamasıdır. Açısal hızın bulunmasında aşağıdaki denklemler cRK'ya uygun olarak düzenlenmiştir. Başlangıç değer problemi denklemleri Eşitlik 4.19'de gösterilmiştir.

$$\omega' = f(t, \omega), \qquad \omega(t_0) = \omega_0 \tag{4.19}$$

Bu denklemlerde; sistemin ana kolunun açısal hızı ω, zaman t, açısal hızın zamana göre değişimi ω' ile gösterilmiştir. Aşağıda artım fonksiyonu ve her bir eğimi ifade eden denklemler gösterilmiştir.

$$\omega_{n+1} = \omega_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{4.20}$$

Eşitliklerde ω_{n+1} hesaplanmak istenen açısal hızı, ω_n bir önceki açısal hızı temsil etmektedir.

$$k_1 = hf(t_n, \omega_n) \tag{4.21}$$

$$k_2 = hf(t_n + \frac{h}{2}, \omega_n + \frac{k_1}{2}) \tag{4.22}$$

$$k_3 = hf(t_n + \frac{h}{2}, \omega_n + \frac{k_2}{2}) \tag{4.23}$$

$$k_4 = hf(t_n + h, \omega_n + k_3) \tag{4.24}$$

Bu çalışmada açısal hız çözümünde Euler, Runge-Kutta 4, Cash-Karp, Dormand-Prince 5, Fehlberg 78 yöntemleri incelenmiş ve test edilmiştir. Bu yöntemlerin hepsi benzer sonuçlar ürettiğinden dolayı Bölüm 5'te açısal hız çözümü için yalnızca cRK yönteminin sonuçları gösterilmiştir.

4.5. Eğri Uydurma (Curve Fitting)

Şekil 4.11'de Eşitlik 2.10 ile elde edilen G eğrisi ile teğetsel ivmelenmenin ihmal edilmesi ile elde edilen G eğrisinin (Eşitlik 4.35) birbirine benzer olduğu görülmektedir. Aradaki farkı tahmin eden bir eğri bulabilmek amacıyla WolframAlpha [54] aracı kullanılarak denemeler yapılmıştır. Bu çalışmada elde edilen eğri denklemi gösterilmiştir.

$$\omega_{wt} = \sqrt[4]{(G^2 - 1)g/R^2} \tag{4.25}$$

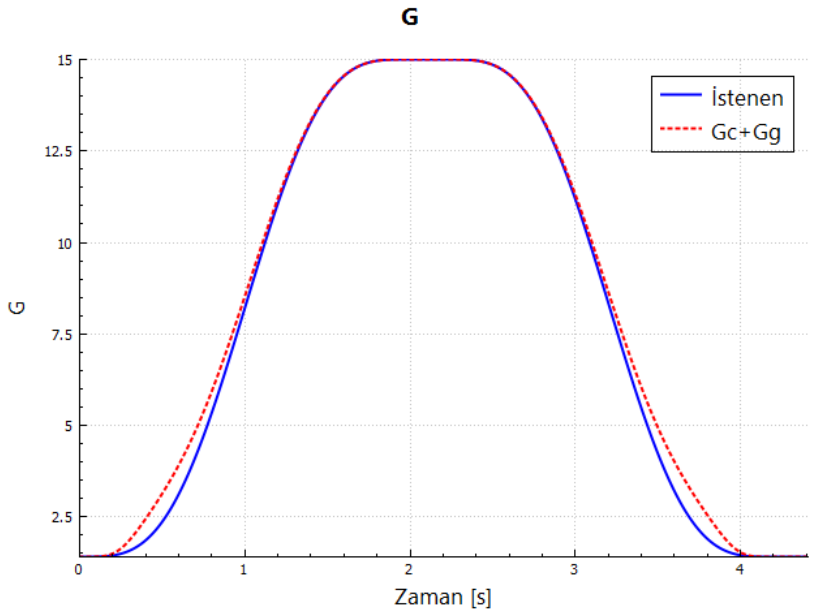
ω_{wt}, teğetsel ivme ihmal edildiğinde hesaplanan açısal hızı ifade eder.

$$\omega_{fark} = \frac{\dot{H}^2}{256\sqrt[2]{H^3} * \omega_{wt}} \tag{4.26}$$

$$\omega_{tahmini} = \omega_{wt} - \omega_{fark} \tag{4.27}$$

Tahmini açısal hız farkı ω_{fark} ile gösterilmiştir.

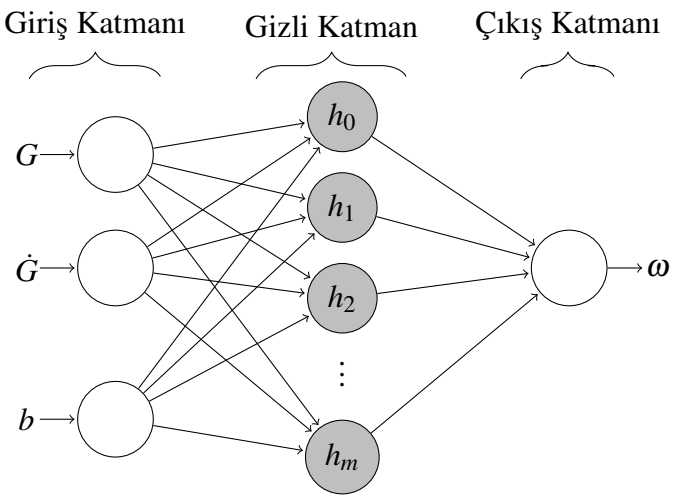
Bölüm 5'te, "Eğri Uydurma" ile gösterilen ω ve G eğrileri, Eşitlik 4.27'den elde edilmiştir.



Şekil 4.11. İstenen G ve teğetsel ivme ihmal edildiğinde elde edilen G eğrisi (ω_{wt})

4.6. Yapay Sinir Ağları

Açısal hızın çözümünde kullanılan YSA modeli Şekil 4.12'de gösterilmiştir. Şekilde görüldüğü gibi girdi katmanı istenen G kuvveti ve birim zamandaki G değişimi verilerinden oluşmaktadır. Gizli katman sayısı sonuca önemli oranda iyileşme sağlamadığından hesaplama maliyeti göz önüne alıp modelde tek gizli katman kullanılmıştır.



Şekil 4.12. Açısal hız çözümü için YSA yapısı

Eğitim verisi ağa tek tek vermek yerine küçük yığınlar (mini batch) halinde verilmiştir.

Ağ eğitilirken olabilecek bütün olasılıkları içeren büyük bir eğitim kümesi hazırlanmıştır. Literatürde büyük veri kümeleri üzerinde çalışan YSA'lar için yığın eğitim kümesi önerilmiştir [8]. Eğitim kümesini yığın şeklinde hazırlamanın avantajı yerel minimuma takılma sorununu çözmektedir. Öğrenme katsayısı ve momentum parametrelerine ek olarak yığın büyüklüğü parametresi eklenir. Yığın büyüklüğü 1 olarak seçilirse yöntemin adı gradyan iniş algoritması olur. Eğitim kümesindeki eleman sayısı kadar seçilirse kullanılan algoritmanın adı Stokastik İniş Algoritması olur. Yığın büyüklüğü 1 ile eğitim kümesi eleman sayısı arasında bir değer seçilmesi durumunda yöntemin adı Mini Stokastik İniş Algoritması olur.

Eşitlik 4.28 ve Eşitlik 4.29 ile ağırlıkların güncellenmesi için kullanılan formüller verilmiştir.

$$\Delta w^t = \eta \Delta w^{t-1} - \frac{\alpha}{|B_t|} \sum_{n \in B_t} \nabla E_n(w) \tag{4.28}$$

$$w^t = w^{t-1} + \Delta w^t \tag{4.29}$$

Denklemde, B_t rastgele seçilen t'inci mini yığında bulunan eğitim kümesini, $|B_t|$ yığın büyüklüğünü temsil etmektedir. Öğrenme katsayısı değeri [1e-5, 0,1] aralığında denenmiştir. Momentum değeri [0, 1) aralığında denenmiştir.

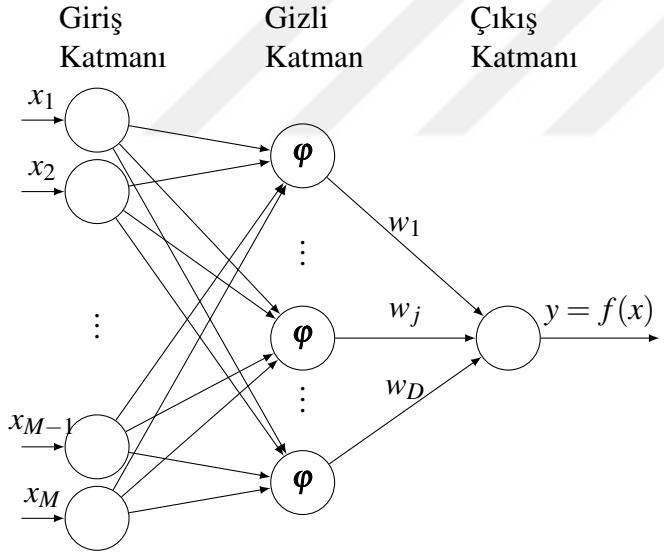
Bölüm 5'te, "YSA" ile gösterilen açısal hız değerleri Şekil 4.12'de gösterilen YSA modeli kullanılarak elde edilmiştir.

4.7. Radyal Tabanlı Fonksiyon Ağlar

Radyal Tabanlı Fonksiyon Ağlar (RTFA), çok katmanlı ileri beslemeli YSA'nın özel bir halidir. İlk defa 1988 yılında Broomhead ve Lowe tarafından yapılan bir çalışmada tanıtılmıştır [16]. Giriş katmanı, ara katman ve çıkış katmanı olmak üzere üç katmandan oluşur. İki önemli özelliği vardır: Birincisi, ağda tek gizli katman bulunması, ikincisi ise gizli katman nöronlarında aynı tür aktivasyon fonksiyon kullanılmasının yerine çekirdek fonksiyon olarak adlandırılan Radyal Tabanlı Fonksiyonların (RTF) kullanılmasıdır. Şekil 4.13'de ağın genel görünümü gösterilmektedir. Şekilde gösterilen ağ M tane girdi D tane RBF kullanarak tek bir çıktı değerini elde etmek için kurulmuştur. RTFA'da öğrenme danışmanlı öğrenme olarak gerçekleşir. Öğrenmede temel yaklaşım, bir grup RTF'nin ağırlıklandırması ile istenen f

fonksiyonuna yaklaşmaktır. Ağ mimarisinin basitliği nedeniyle ÇKYSA modellerine göre birçok avantajı vardır. RTFA'da ağırlıkların yalnızca gizli ve çıkış katmanı arasında yer almasından dolayı geri yayılım algoritmalarına göre bu ağlar daha hızlı eğitilebilir [28]. Gizli katmandaki nöronların aktivasyon fonksiyonları bir c merkezi ve β bant genişliği ile belirlenir.

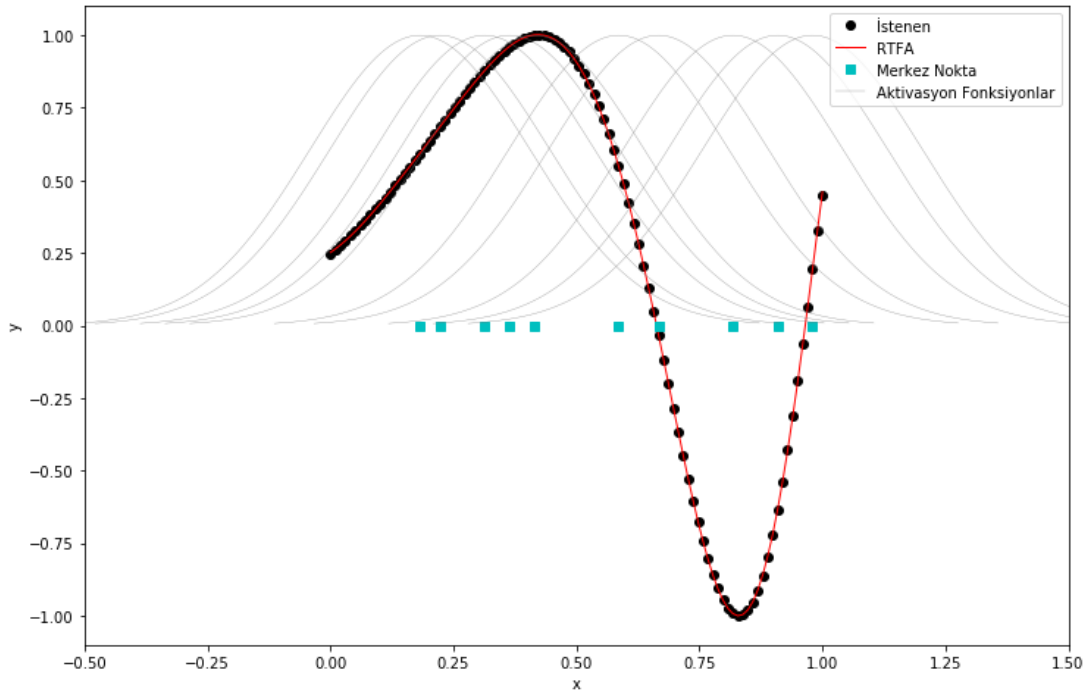
RTFA'da her nöronda farklı aktivasyon fonksiyonların kullanılması ile farklı bölgelerinde farklı davranışlar gösteren girdi uzayları üzerinde iyi sonuçlar elde edilir. Girdi hiç bir şekilde işlenmeden doğrudan girdi katmanı aracılığı ile gizli katman nöronlarına iletilir. Girdi katmanını gizli katmana bağlayan tüm ağırlık değerlerinin 1 olduğu ve çözüm süresince değişmediği varsayılır [46]. RTFA değiştirilecek parametre sayısı Çok Katmanlı YSA'ya göre daha azdır, bundan dolayı öğrenme süreci daha kısadır. Ağ oluşturulurken en önemli adımlardan biri gizli katmandaki nöron sayısının belirlenmesidir. En uygun nöron sayısı yapılan denemelerle belirlenir. Bu çalışmada RTFA kullanabilmek için SciPy [52]



Şekil 4.13. Radyal tabanlı yapay sinir ağının genel görünümü

kütüphanesi kullanılmıştır. Şekil 4.13'de $y = sin(2(x+0.5)^3)$ eğrisinin RTFA kullanılarak çözümü gösterilmiştir. Gizli katmanda 10 tane nöron kullanılmıştır. Şekildeki grafiği elde etmek için kullanılan kod EK-5 ile sunulmuştur.

RBFN kullanarak x girişine karşılık n'inci çıkış değeri aşağıda gösterildiği şekilde hesaplanır. Denklemde en fazla N tane çıkış için hesaplama yapılır.



Şekil 4.14. Radyal tabanlı yapay sinir ağı kullanarak doğrusal olmayan bir eğrinin çözümü

$$y_n(\mathbf{x}) = \sum_{d=1}^{D} w_{dn}\varphi_d(\|\mathbf{x}-\mathbf{c}_d\|), \quad n = 1,\ldots,N, \tag{4.30}$$

Denklemde d gizli katmandaki nöron sırasını, c_d merkez noktayı, φ_d RTF ve w_{dn} gizli katman ile çıkış katmanı arasındaki ağırlığı gösterir. K tane örneğimiz $((\hat{\mathbf{x}}_k, \hat{\mathbf{y}}_k))$ olduğunu düşünelim. Bu durumda denklemler aşağıda gösterilen matrisler şeklinde daha genel bir ifade olarak yazılabilir. Bu denklemlerde kullanılan aktivasyon fonksiyonu aşağıda gösterilmiştir.

$$\varphi\Big(\|\mathbf{x}-\mathbf{c}_d\|\Big) = \exp\left[-\beta\,\|\mathbf{x}-\mathbf{c}_d\|^2\right] \tag{4.31}$$

$$\underbrace{\begin{bmatrix} \hat{y}_{11} & \cdots & \hat{y}_{1N} \\ \vdots & \ddots & \vdots \\ \hat{y}_{K1} & \cdots & \hat{y}_{KN} \end{bmatrix}}_{Y} = \underbrace{\begin{bmatrix} \varphi_1(\|\hat{\mathbf{x}}_1-\mathbf{c}_1\|) & \cdots & \varphi_D(\|\hat{\mathbf{x}}_1-\mathbf{c}_D\|) \\ \vdots & \ddots & \vdots \\ \varphi_1(\|\hat{\mathbf{x}}_K-\mathbf{c}_1\|) & \cdots & \varphi_D(\|\hat{\mathbf{x}}_K-\mathbf{c}_D\|) \end{bmatrix}}_{\Phi} \underbrace{\begin{bmatrix} w_{11} & \cdots & w_{1N} \\ \vdots & \ddots & \vdots \\ w_{D1} & \cdots & w_{DN} \end{bmatrix}}_{W} \tag{4.32}$$

Burada Y, W ve Φ matrisleri sırası ile $K \times N$, $K \times D$ and $D \times N$ boyutlu matrislerdir. K nokta sayısını, N çıkış katmanında yer alan çıktı sayısını, D gizli katmandaki nöron sayısını temsil etmektedir.

Literatürde W ağırlıklarını bulabilmek için kullanılan en yaygın yöntem sözde-ters (pseudo-inverse) yöntemidir. Bu yönteme göre ağırlıklar, gizli katmanda yer alan nöron

çıkışlarının (aktivasyon fonksiyonu ile hesaplanan değer) tersinin istenen çıkış değeri ile çarpılması ile elde edilir.

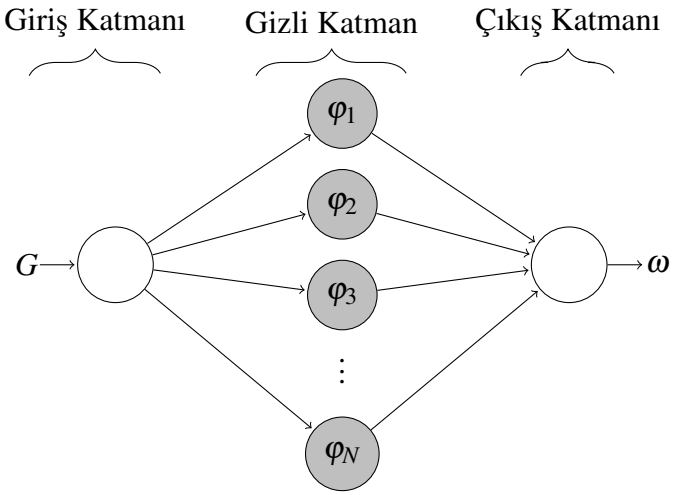
$$W = (\Phi^T \Phi)^{-1} \Phi^T \tag{4.33}$$

$$Y = \Phi^{\dagger} Y. \tag{4.34}$$

Denklemde $\Phi^{\dagger}$ sözde-tersini ifade etmektedir.

Açık çevrim G profiller için azalan G komut dizisi ters çevrilip açısal hız dizisi bulunarak çözüm elde edilmeye çalışılmaktadır; ancak bu yöntemin bazı dezavantajları bulunmaktadır. Azalan G komut dizisindeki her bir komut zaman aralığının sabit olduğu varsayımı yapılmaktadır. Gerçek zamanlı olmayan bir işletim sisteminde zamanın sabit olmasını garanti etmek zordur. Ayrıca bu yöntem için hafızada G komut sayısı kadar yer almak zorunluluğu vardır. Bu çalışmada açısal hızın ters çevrilerek çözümü için önerilen algoritmanın eksiklikleri giderilmeye ve bir başka alternatif bulunmaya çalışılmıştır.

Artan G eğrisi kullanılarak ağ üzerinde danışmanlı eğitim yapıldı. Eğitilmiş ağ üzerinde artan ve azalan G eğrisi verisi kullanarak ağ test edilerek başarısı gözlemlenmiştir.



Şekil 4.15. Açısal hız hesaplamak için kullanılan RTFA modeli

4.8. Teğetsel İvmelenmeyi İhmal Ederek Hesaplama

Açısal hızın çözümünde kullanılan bir diğer yöntem ise Eşitlik 2.10'de türev içeren teğetsel ivmeyi ihmal ederek elde edilen Eşitlik 4.35'i kullanmaktır. Bu yöntemde açısal hız, istenen bütün G komutları için düzgün doğrusal hareket kurallarına göre hesaplanır. Bu denklemde türev olmadığı için çözüm oldukça kolaylaşmaktadır. Bu yöntem "Motion Base Control Process And Pilot Perceptual Simulator" patent dokümanından alınmıştır [15].

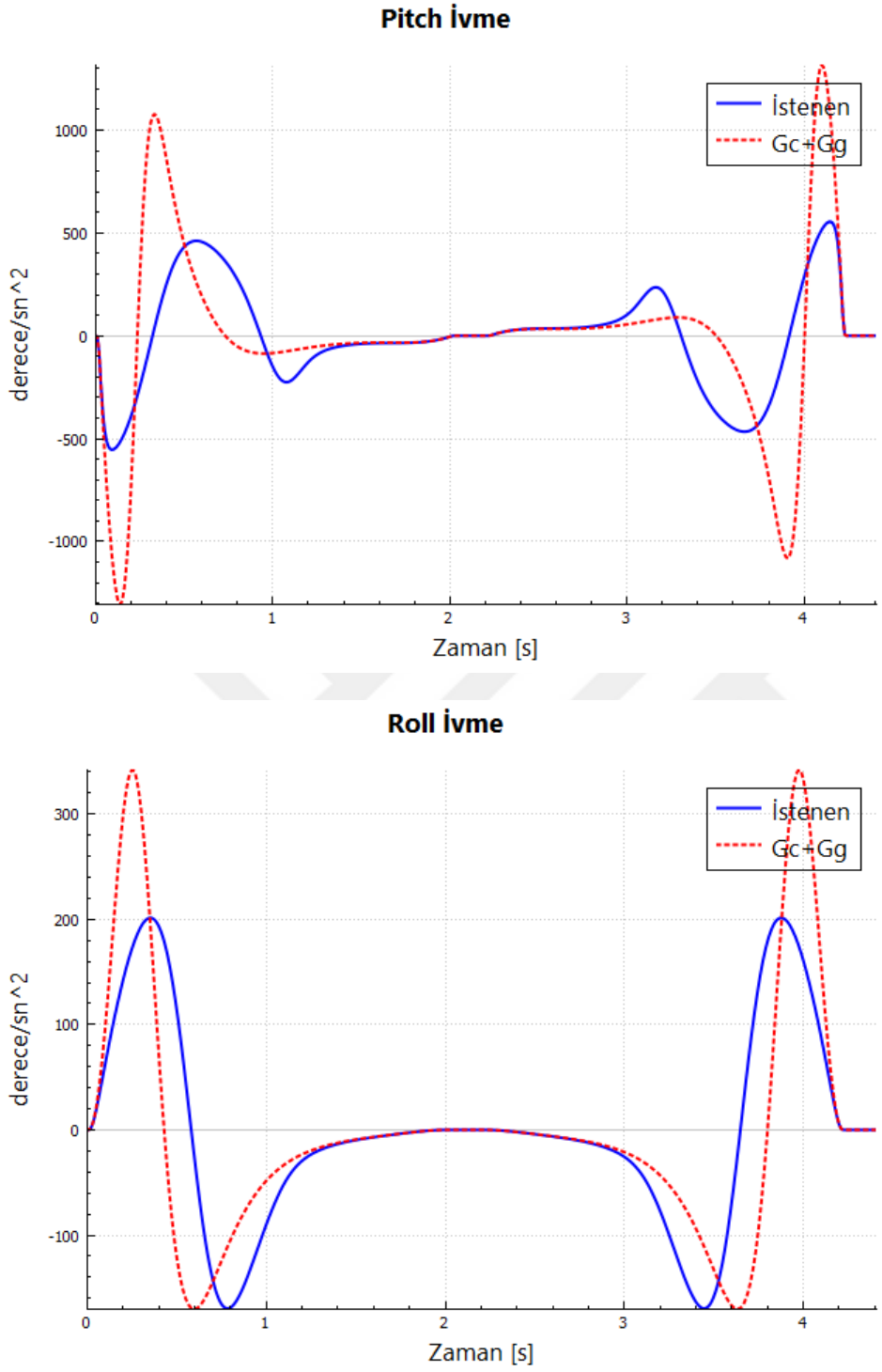
$$G^2 = {G_c}^2 + g^2 \tag{4.35}$$

Bölüm 5'te bu yöntem ile elde edilen sonuçlar "Gc+Gg" olarak gösterilmiştir.

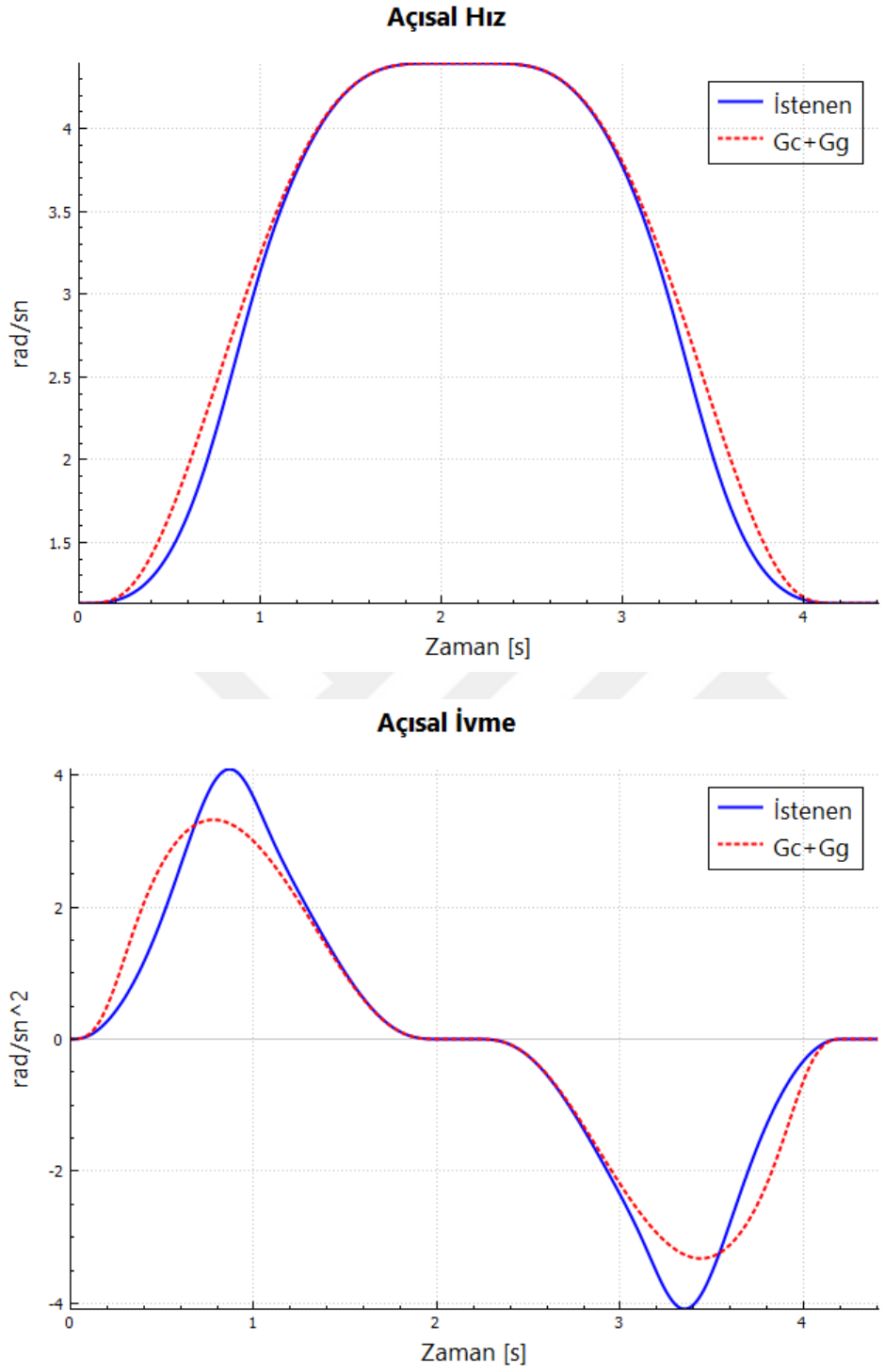
$$w = \sqrt[4]{(G^2 - 1)g/R^2} \tag{4.36}$$

Eşitlik 4.36'da gösterilen denklemin en büyük dezavantajı istenen G komutu için daha yüksek açısal hız oluşturmasıdır (Şekil 4.17). Elde edilen yüksek açısal hız değeri yunuslama (pitch) ve yuvarlanma (roll) eksenlerine uygulanan komutlarda oluşan istenmeyen G_x ve G_y etkilerine neden olmaktadır (Şekil 4.16).

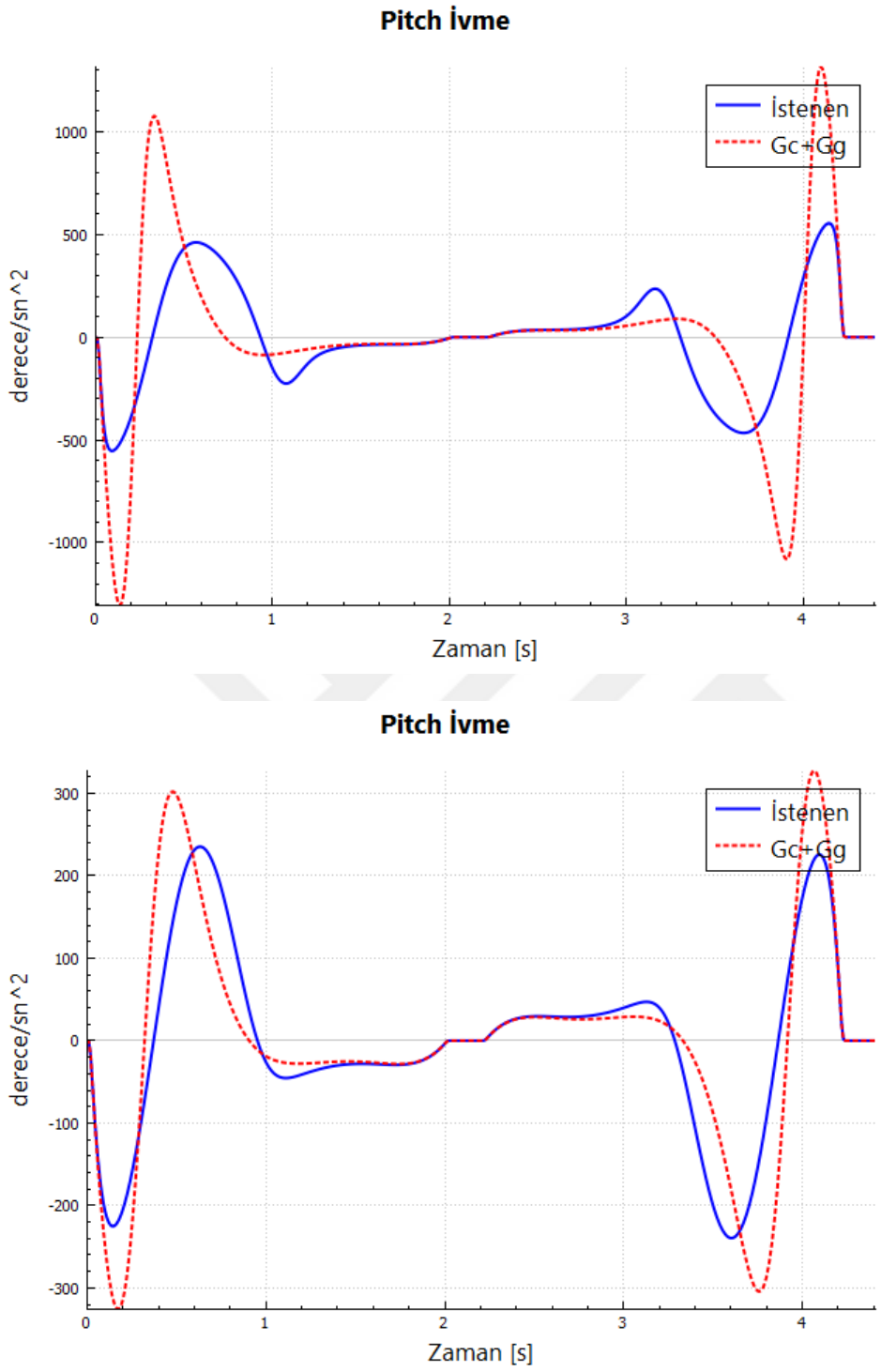
Kısa kollu sistemlerde oluşan hata etkisi daha az olacağı düşünülürse karmaşık nümerik yöntemler yerine bu yöntem kullanılabilir. Şekil 4.18 kısa kollu (3 metre) bir sistem ile uzun kollu sistemde sistemin en yüksek uygulayabileceği G değerlerine karşılık yunuslama (pitch) eksenin ivme değerleri gösterilmiştir. Uzun kollu sistemde elde edilen ivme değerinin kısa kollu olan sistemdekine göre çok yüksek olduğu görülmektedir.



Şekil 4.16. Teğetsel ivmenin ihmal edilmesinin yunuslama (pitch) ve yuvarlanma (roll) eksenleri üzerindeki etkileri



Şekil 4.17. Teğetsel ivmenin ihmal edilmesinin hesaplanan açısal hız ve açısal ivme üzerindeki etkileri

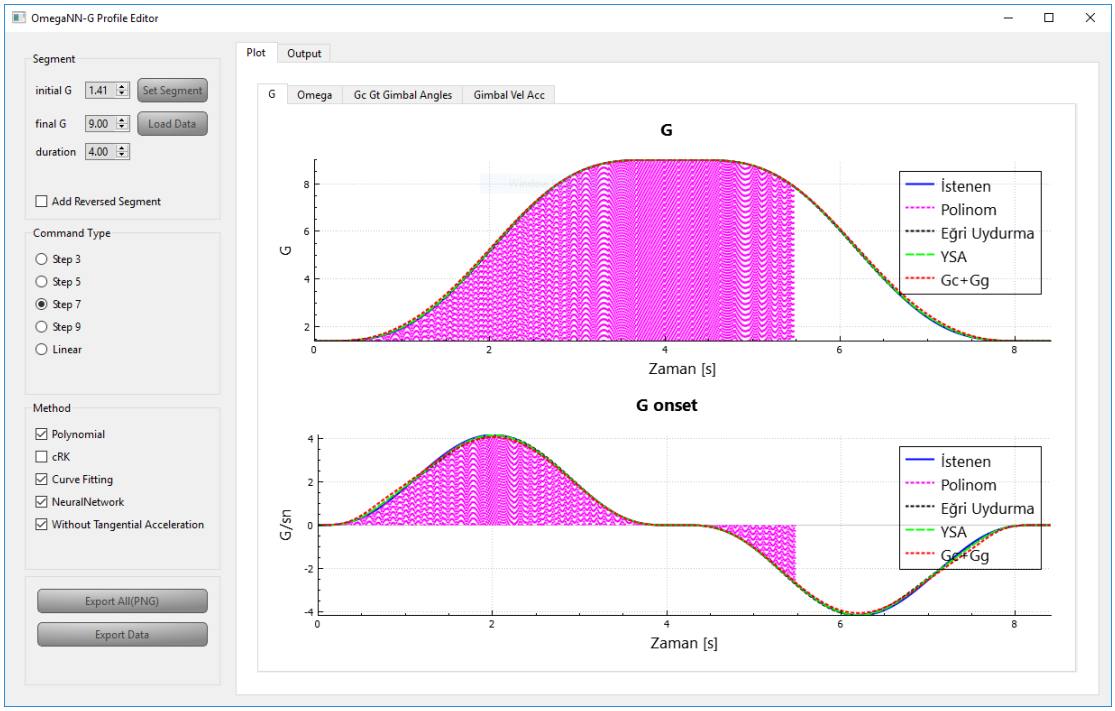


Şekil 4.18. Teğetsel ivmenin ihmal edilmesinin uzun kollu (üst) ve kısa kollu (alt) sistemlerdeki etkisinin karşılaştırılması

5 . UYGULAMA VE DENEYSEL SONUÇLAR

Bu çalışmada kullanılan yöntemleri test edebilmeyi sağlayan uygulamalar python, Octave, wolframAlpha, Qt/C++ ve boost kütüphaneleri kullanılarak geliştirilmiştir ([47–50, 54]). Profil editör olarak tasarlanan uygulamanın temel amacı istenilen G komutlarına karşılık sistem açısından önemli olan verileri oluşturmak ve grafiksel olarak karşılaştırmalı olarak göstermektir.

Geliştirilen uygulamadan bir görünüm Şekil 5.1'de gösterilmiştir.

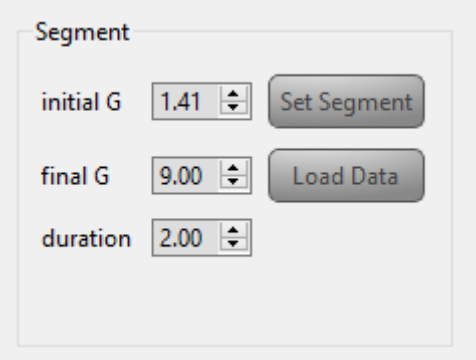


Şekil 5.1. G profil hazılayıcı uygulama

Profil editör veri üretebilmek için aşağıdaki girdilere ihtiyaç duyar.

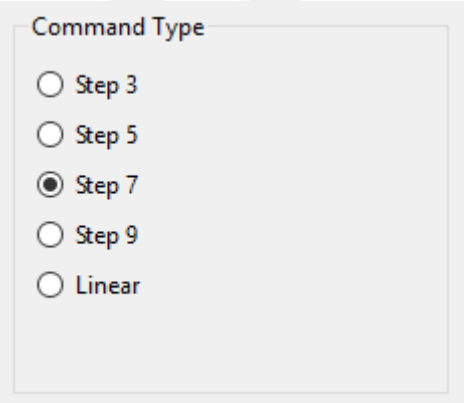
- Başlangıç G seviyesi
- Bitiş G seviyesi
- Segment süresi

Şekil 5.2'de G segmenti tanımlamak için gerekli olan komutların girileceği ekranı göstermektedir.



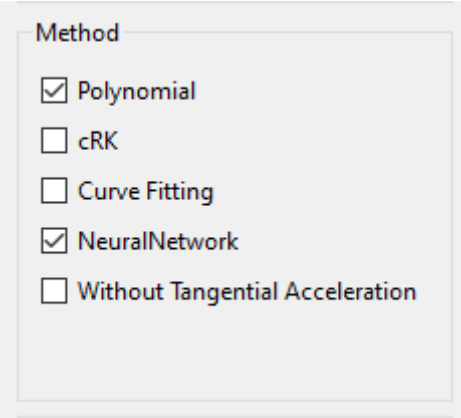
Şekil 5.2. G profil hazılayıcı - segment

Şekil 5.3'de oluşturulacak *G* segmentin hangi algoritma ile oluşturulacağına karar vermek için gerekli olan seçenekleri içeren ekran gösterilmektedir.



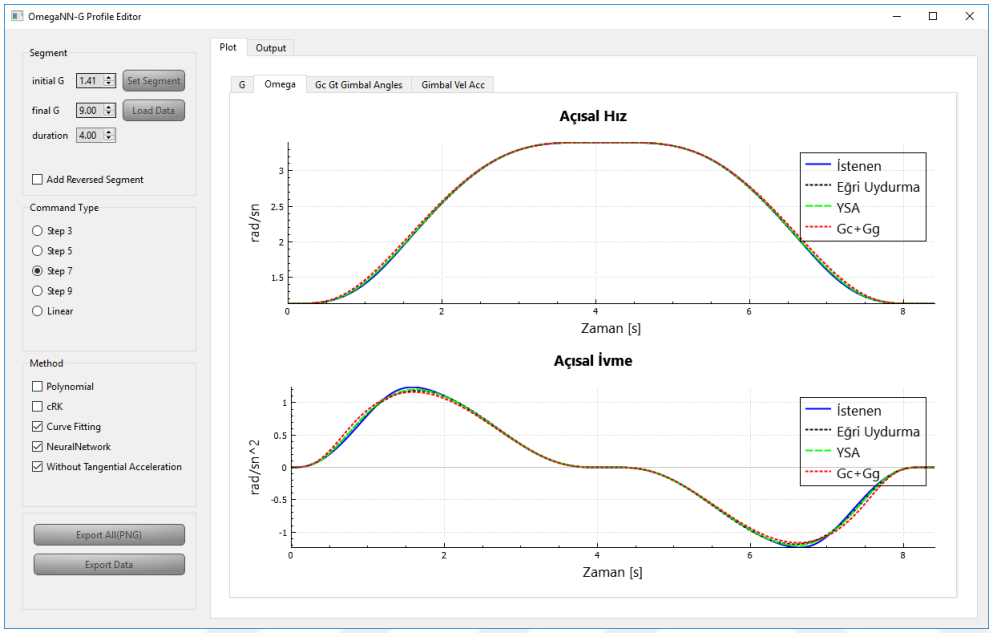
Şekil 5.3. G profil hazılayıcı - *G* komut tipi

Şekil 5.4'te oluşturulan *G* segmente karşılık açısal hızın hangi algoritma ile hesaplanacağına karar vermek için gerekli olan seçenekleri içeren ekran gösterilmektedir.

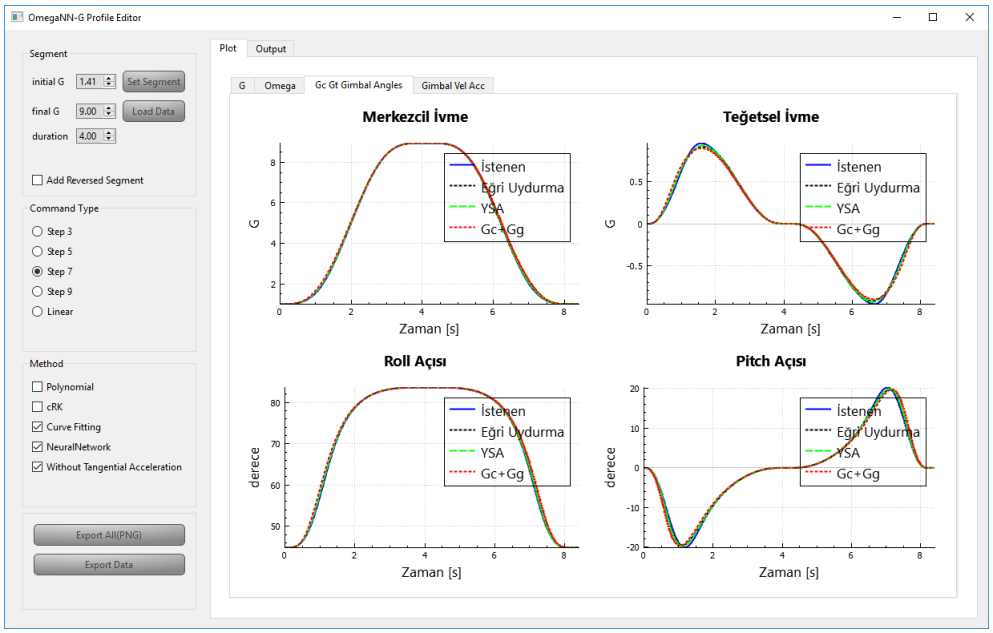


Şekil 5.4. G profil hazılayıcı - uygulanacak yöntemler

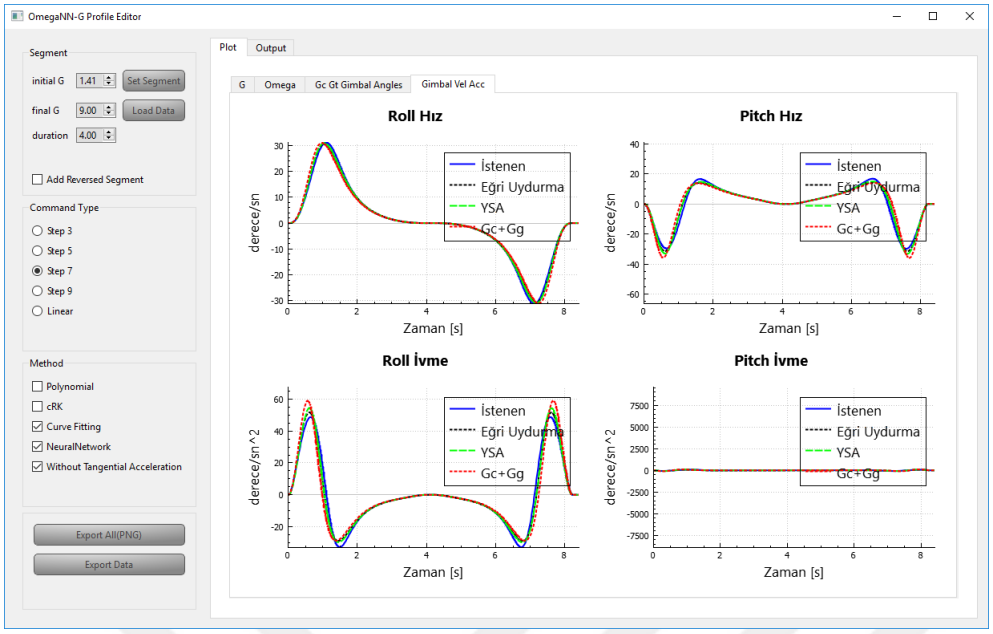
Şekil 5.5'te oluşturulan *G* segmente karşılık hesaplanan açısal hız ve ivme değerlerinin grafiksel ekranı gösterilmektedir. Şekil 5.6 ve Şekil 5.7'de oluşturulan *G* segmente karşılık hesaplana yunuslama (pitch) yuvarlanma (roll) eksenleri ile alakalı detaylı veriler içeren grafikler gösterilmektedir.



Şekil 5.5. G profil hazılayıcı uygulama - planetary eksen hız ve ivme grafikleri



Şekil 5.6. G profil hazılayıcı uygulama - yunuslama (pitch), yuvarlanma (roll) eksen açı grafikleri



Şekil 5.7. G profil hazılayıcı uygulama - yunuslama (pitch), yuvarlanma (roll) eksen hız ve ivme grafikleri

Bu bölümde Bölüm 4'de anlatılan yöntemler kullanılarak açısal hız çözümünde elde edilen sonuçlar karşılaştırmalı olarak verilmiştir. Hatırlanacağı üzere Bölüm 4.1'de artan *G* komutları için açısal hızın bulunabildiği; ancak azalan *G* komutları için açısal hızın bulanamadığı anlatılmıştır. Artan *G* komutları için açısal hız cRK ya da 4. dereceden polinom denklemin kökleri bulunması yoluyla en iyi şekilde hesaplanabildiği için bu bölümde yer alan testler ve sonuçlar azalan *G* komutlarını kapsamaktadır.

Teorik olarak yöntemlerin ne kadar başarılı olduğunu görebilmek için; yapılan testlerde sistem açısından uygulanması zor olan profillerden biri tercih edilmiştir. Azalan profil segment 15 *G* ile başlar 1,41 *G* ile biter, artan *G* segment tam tersi değerleri alır. Bir diğer önemli parametre profil segment süresidir. Yapılan çalışmada segment süresi ne kadar küçük seçilirse, ortaya çıkan hata da bir o kadar büyük olduğu görülmüştür. Aynı yöntemlerin farklı profil segment süreleri işletilmesi sonucu gösterilen grafiklerde bu durum rahatlıkla görülebilir.

Bu bölümde yer alan denemelerde kullanılan *G* komutları smooth step algoritması [53] yardımıyla elde edilmiştir. Bu algoritma farklı derecelerde S şeklinde komutlar hazırlamak için kullanılır. Eşitlik 5.3'de beşinci dereceden smooth-step algoritması ile *G* komutların oluşturulması için kullanılan denklem gösterilmiştir. Yüksek dereceli S fonksiyonlar daha yumuşak komut geçişleri sağlarlar. Aşağıdaki denklemlerde S_n *n*' inci dereceden bir S

fonksiyonunu temsil eder.

$$S_1(x) = x \tag{5.1}$$

$$S_3(x) = -2x^3 + 3x^2 \tag{5.2}$$

$$S_5(x) = 6x^5 - 15x^4 + 10x^3 \tag{5.3}$$

$$S_7(x) = -20x^7 + 70x^6 - 84x^5 + 35x^4 \tag{5.4}$$

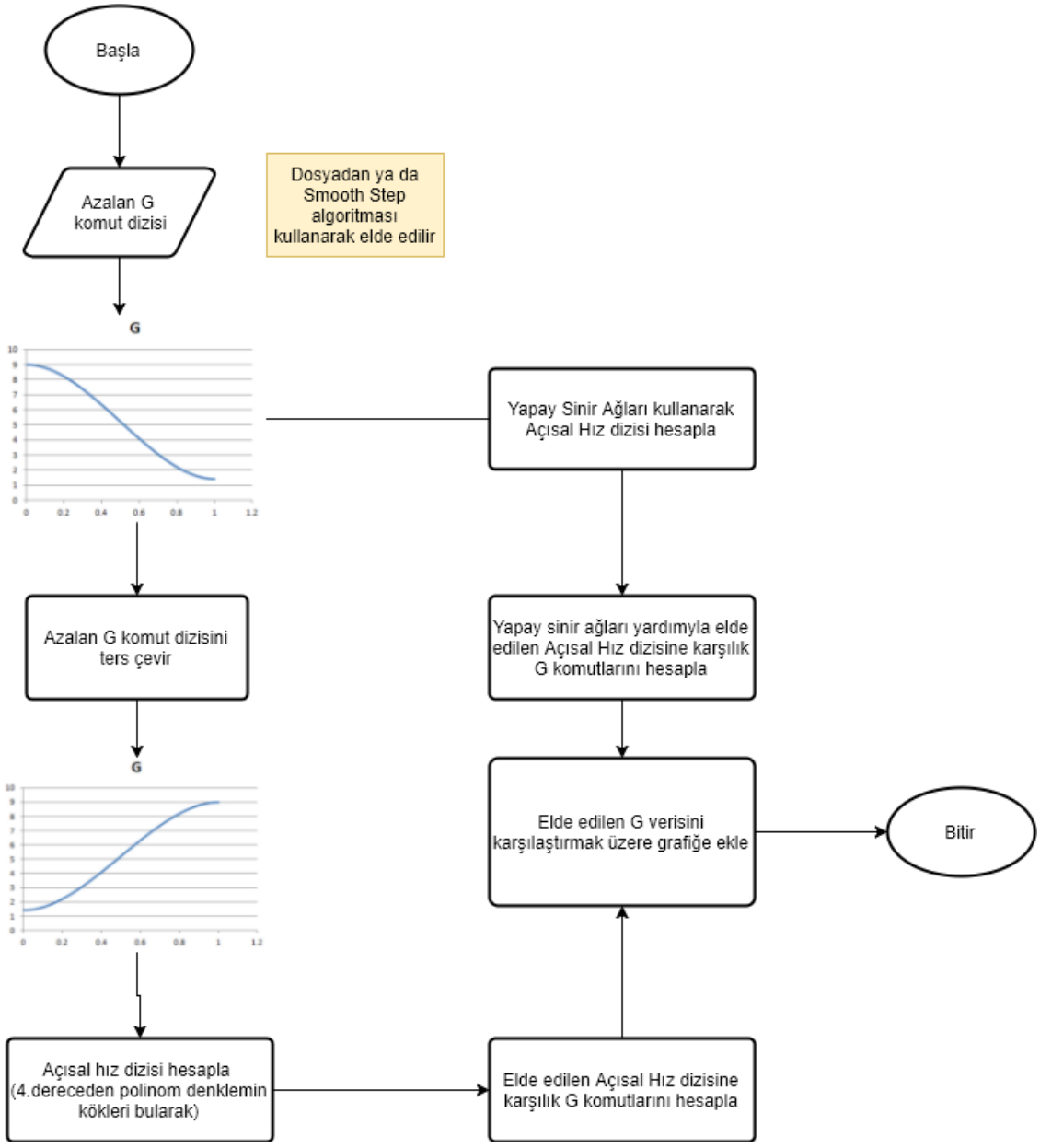
$$S_9(x) = 70x^9 - 315x^8 + 540x^7 - 420x^6 + 126x^5 \tag{5.5}$$

Aşağıda listelenen adımlar takip edilerek uygulanmak istenen G komutları ile Bölüm 4'te anlatılan yöntemi kullanarak elde edilen G komutları karşılaştırılabilir. Şekil 5.8'de yapay sinir ağları yöntemi başarısının karşılaştırılması için gerekli olan adımlar akış diyagramında gösterilmiştir. YSA dışındaki yöntemler için benzer adımlar uygulanır.

1. G komut dizisini dosyadan oku ya da SmoothStep algoritmasını kullanarak oluştur.
2. G komut dizisini ters çevir, böylece artan G komut dizisi elde edilir. Ters çevrilen G dizisini kullanarak ω dizisi oluştur.
3. Bölüm 4'de anlatılan yöntemi kullanarak G komut dizisi girdi olarak alıp ω dizisi hesapla.
4. Adım 2'den elde edilen ω dizisi kullanarak istenen G komutunu hesapla.
5. Adım 3'den elde edilen ω dizisi kullanarak istenen G komutunu hesapla.

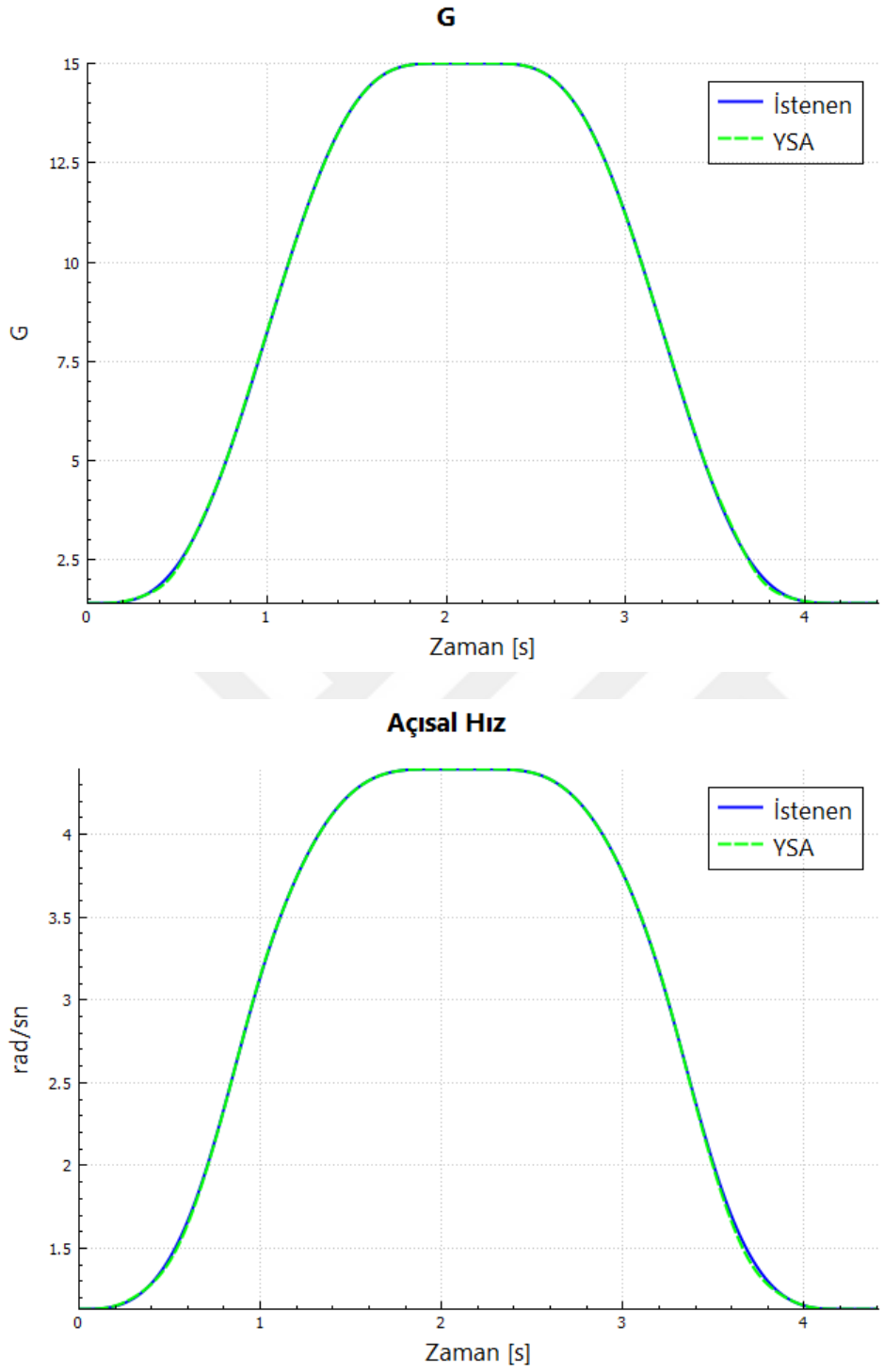
İlk iki adım istenilen G dizisine en yakın sonucu elde etmek için uygulanmıştır. azalan G dizisi ters çevrildiğinde artan G dizisi elde edilir. Daha önce hatırlanacağı üzere artan G komutları polinom denklemin köklerinin bulunması yöntemi ile başarılı bir şekilde bulunabiliyor. Ters çevrilen G komutlarından elde edilen açısal hız dizisi tekrar ters çevrilerek azalan G komutlarına karşılık azalan açısal hız dizisi elde edilir.

Bu bölümde YSA ile elde edilen çözüm diğer yöntemlerle elde edilenlerle karşılaştırılacaktır. YSA'nın diğer yöntemlere göre avantaj ve dezavantajları incelenecektir.



Şekil 5.8. Uygulanmak istenen *G* komutları ve YSA kullanılarak elde edilen *G* komutlarını karşılaştırmak için uygulanan akış diyagramı

Şekil 5.9'da YSA temelli geliştirilen modelin teorik olarak başarılı sonuçlar verdiğini göstermektedir. Profil segment süresi 2 saniyenin üzerinde olduğu sürece problem çözümünde kullanılan en iyi yöntem olduğu görülmüştür. Profil segment süresi 1,5 saniyeye kadar kararlı sonuçlar üretmesine rağmen; daha kısa segmenler için kararsız olabildiği gözlenmiştir. Daha kısa segmentler için ayrı bir YSA modeli kurulması gerektiği için bu çalışmada YSA çözümleri 2 saniye ile sınırlandırılmıştır. Şekil 5.9'da *G* komutlarına karşılık açısal hız grafiği verileri sunulmuştur. Şekil 5.15'te daha uzun profil segment kullanıldığında YSA'nın daha kısa süreli segmentlere göre başarısının arttığı görülmektedir.



Şekil 5.9. YSA modeli kullanarak elde edilen açısal hız çözümü

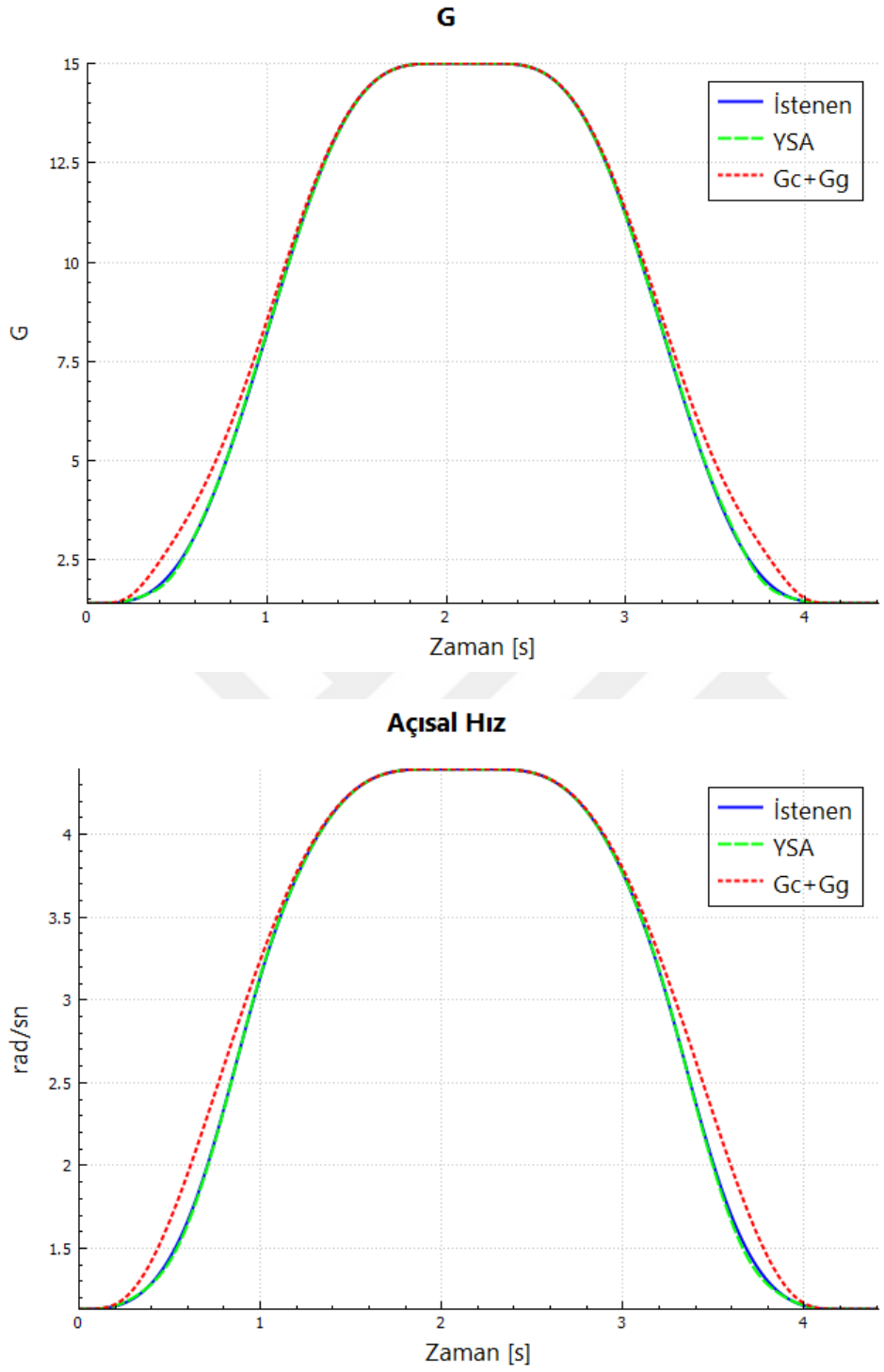
Şekil 5.10'da YSA ile teğetsel ivmelenmenin ihmal edildiği çözüm aynı tür *G* profil segment çalıştırıldıktan sonra elde edilen sonuçlar grafiksel olarak gösterilmiştir. İki saniye ve üzeri profil segmentler için YSA'nın daha iyi sonuçlar ürettiği görülmüştür. YSA'nın yunuslama (pitch) ve yuvarlanma (roll) eksen açılarının hesaplanmasında da iyi sonuçlar ürettiği Şekil 5.11'de rahatlıkla görülmektedir.

Şekil 5.12'de YSA ile cRK sonuçlarının karşılaştırması grafiksel olarak sunulmuştur. Azalan *G* komutları uygulanması açısından YSA nın cRK dan daha iyi bir yöntem olduğu görülmektedir. Artan *G* komutları için cRK daha iyi sonuçlar üretmiştir.

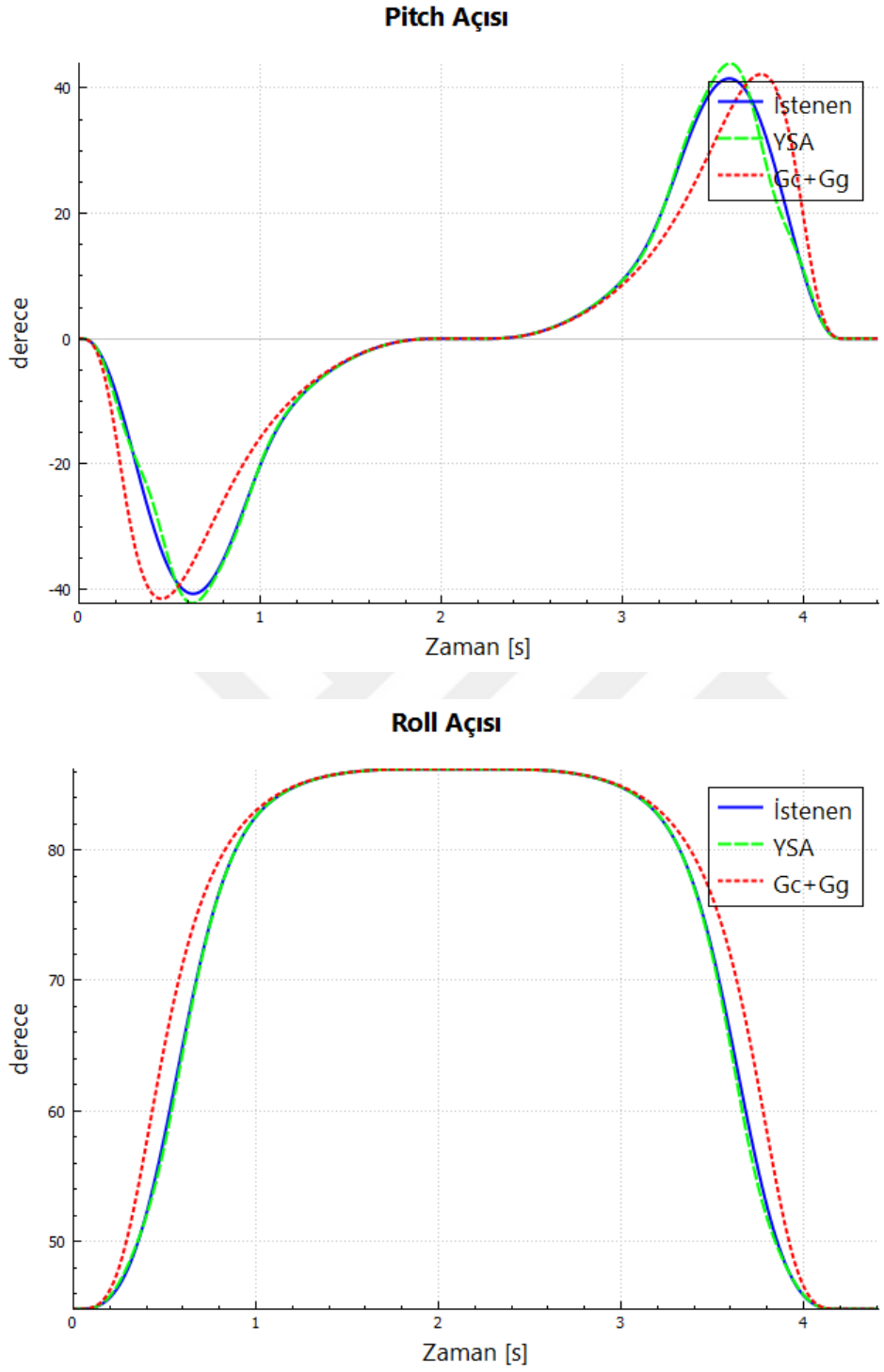
Şekil 5.13'de eğri uydurma yönteminin YSA karşılaştırması grafiksel olarak verilmiştir. Eğri uydurma yöntemi diğer yöntemlere göre YSA'ya en yakın sonuçları vermesine rağmen, oluşturulan denklemin hazırlanma maliyeti göz önüne alındığında YSA'dan daha iyi bir çözüm olmadığı görülmüştür. Ayrıca Şekil 5.14'te birim zamanda *G* değişimi ve yunuslama (pitch) eksen açıları, YSA'nın daha iyi bir çözüm olduğunu açıkça göstermektedir.

Şekil 5.17 ile doğrusal artan *G* komut dizisinin eğitim verisi gösterilmiştir. Şekil 5.17 ise doğrusal artan *G* komut dizisinin eğitimi sonrası elde edilen RTFA modeli ile hesaplanan açısal hız dizisi gösterilmiştir. Eğitim ve test verileri çok farklı olmalarına rağmen RTFA nın açısal hız değerini teğetsel ivmenin ihmal edildiği yöntemden daha iyi çözdüğü görülebilir.

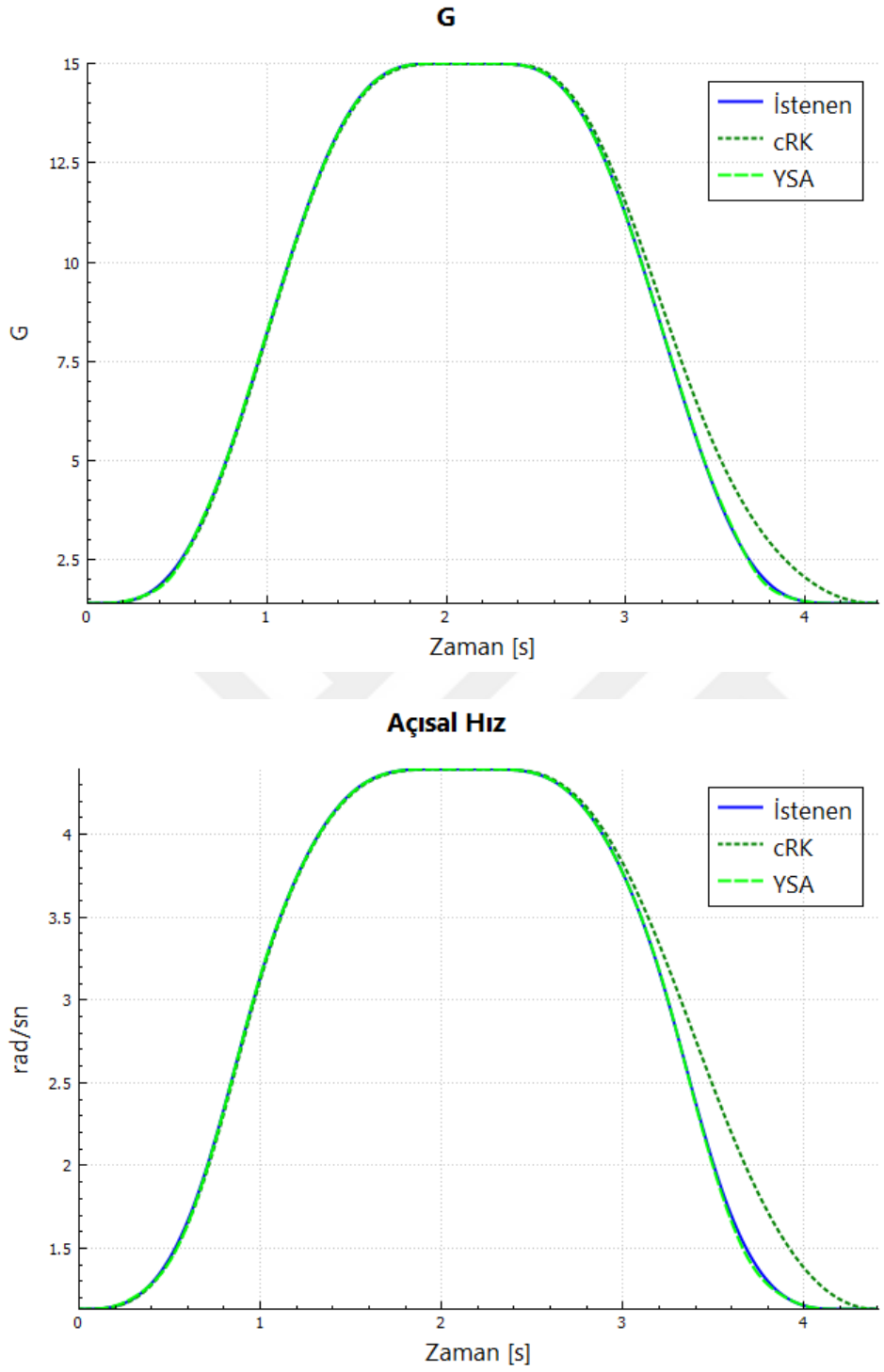
Şekil 5.20'de görüleceği gibi RTFA eğitim ve test verisi aynı tür *G* komutu dizisi (Smooth Step 9) için açısal hız değeri eksiksiz hesaplanabilmektedir. Eğitim ve test verilerinde başlangıç ve bitiş *G* seviyelerinin aynı olduğu durumlarda RTFA problemi teğetsel ivmenin ihmal edildiği yöntemden daha iyi sonuçlar vermektedir. RTFA modeli azalan *G* komutlarının ters çevrilip açısal hızı hesaplayan algoritmanın eksiklerini gidermek için alternatif olarak oluşturulmuştur. Çalışma zamanında iki komut arasındaki zamanın sabit olması gibi bir ön şart gerek kalmamaktadır. Her istenen *G* komutu için açısal hız değerini hafızada tutma gerekliliği ortadan kalkmıştır. Şekil 5.21 ile RTFA modelinin diğer yöntemlerle nasıl karşılaştırıldığını gösteren akış diyagramında gösterilmiştir.



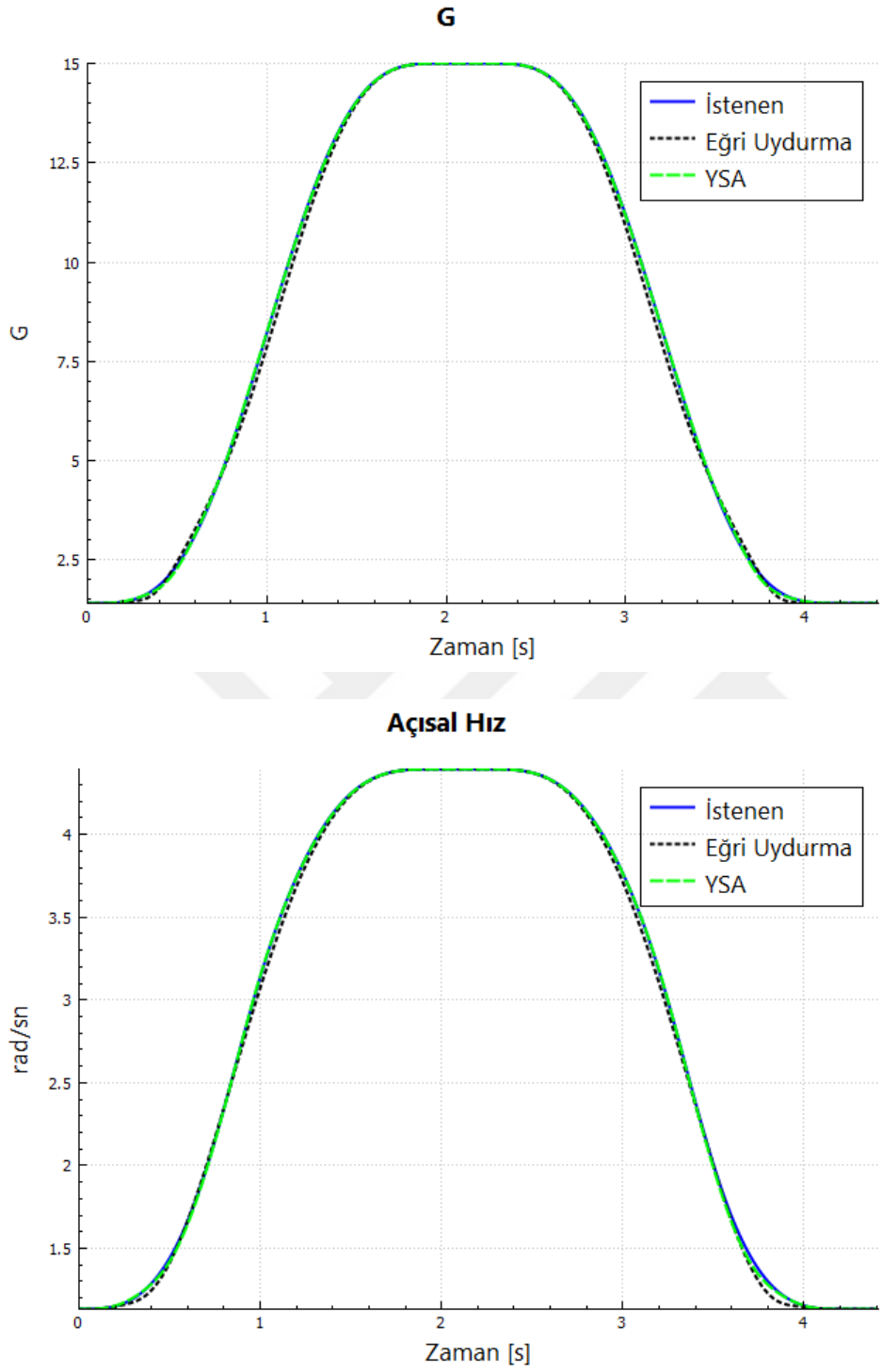
Şekil 5.10. YSA modeli ile teğetsel ivmenin ihmal edilmesi ile açısal hız hesaplama karşılaştırması



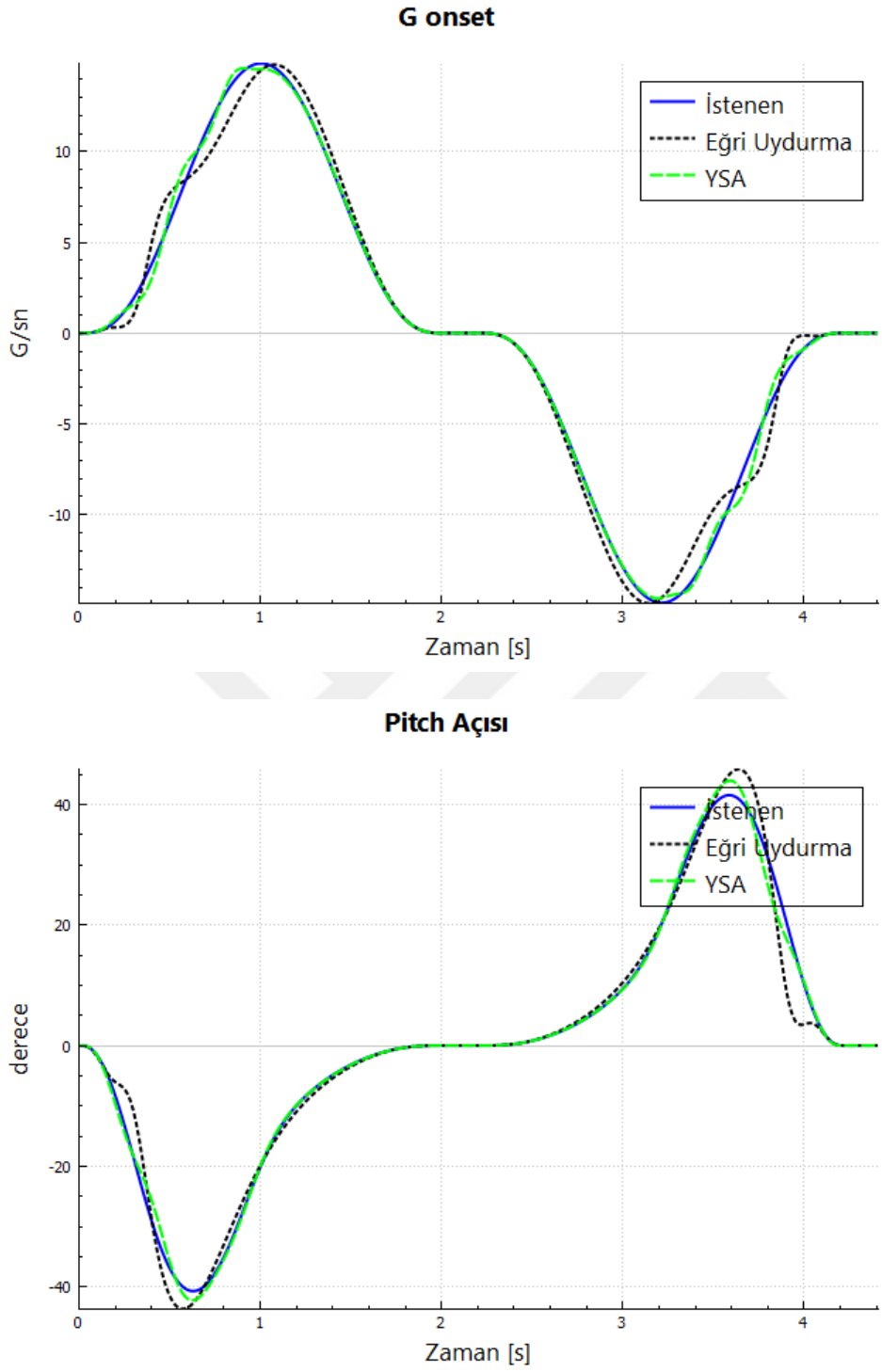
Şekil 5.11. YSA modeli ile teğetsel ivmenin ihmal edilmesi ile elde edilen yunuslama (pitch) yuvarlanma (roll) eksen açılarının karşılaştırması



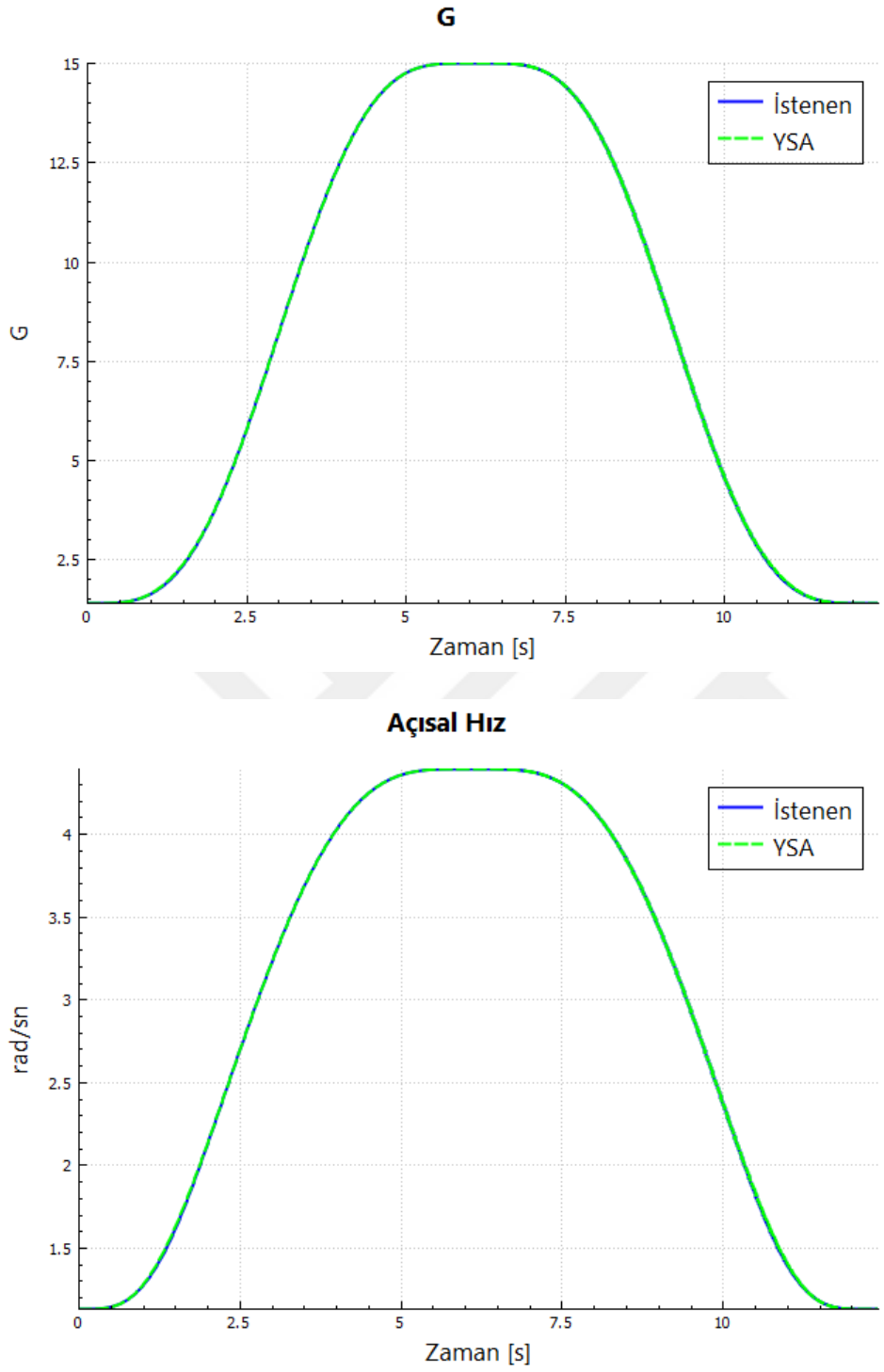
Şekil 5.12. Açısal hız çözümü için YSA modeli ve cRK yöntemlerinin sonuçlarının karşılaştırılması



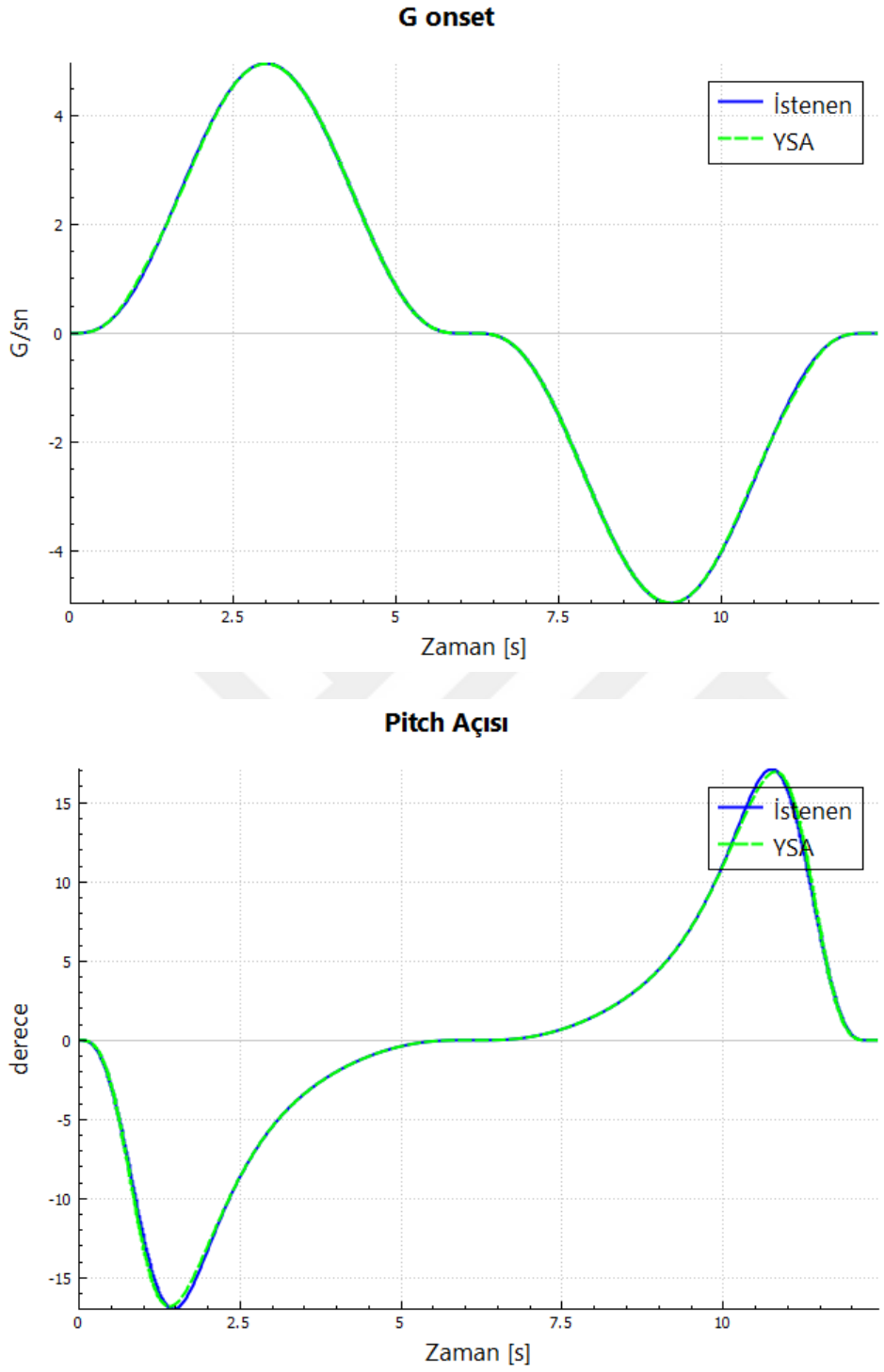
Şekil 5.13. Açısal hız çözümü için YSA ve eğri uydurma yöntemlerinin karşılaştırılması



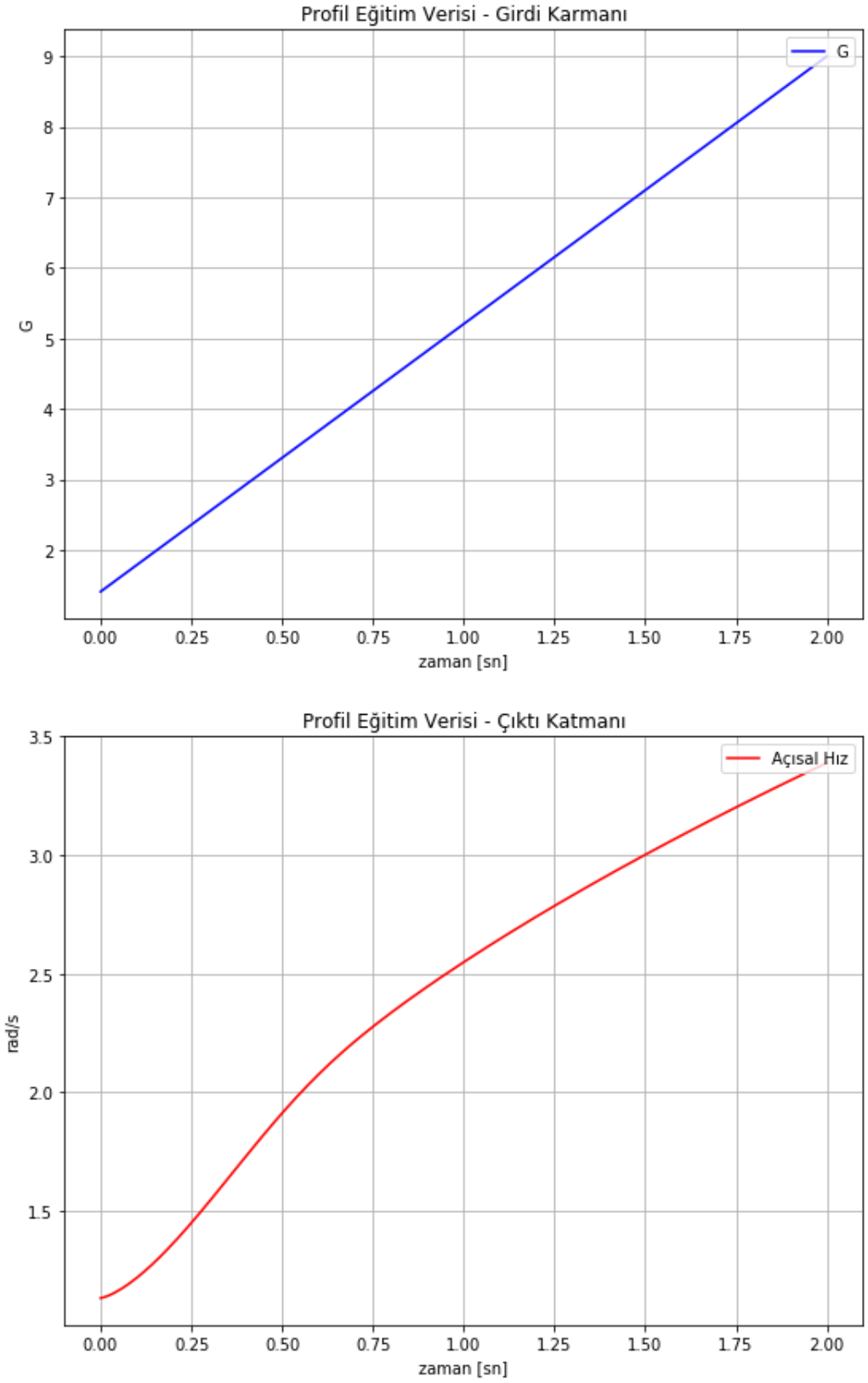
Şekil 5.14. Açısal hız çözümü için YSA ve eğri uydurma yöntemlerinin *G* değişimi ve yunuslama (pitch) eksen açı verileri ile karşılaştırılması



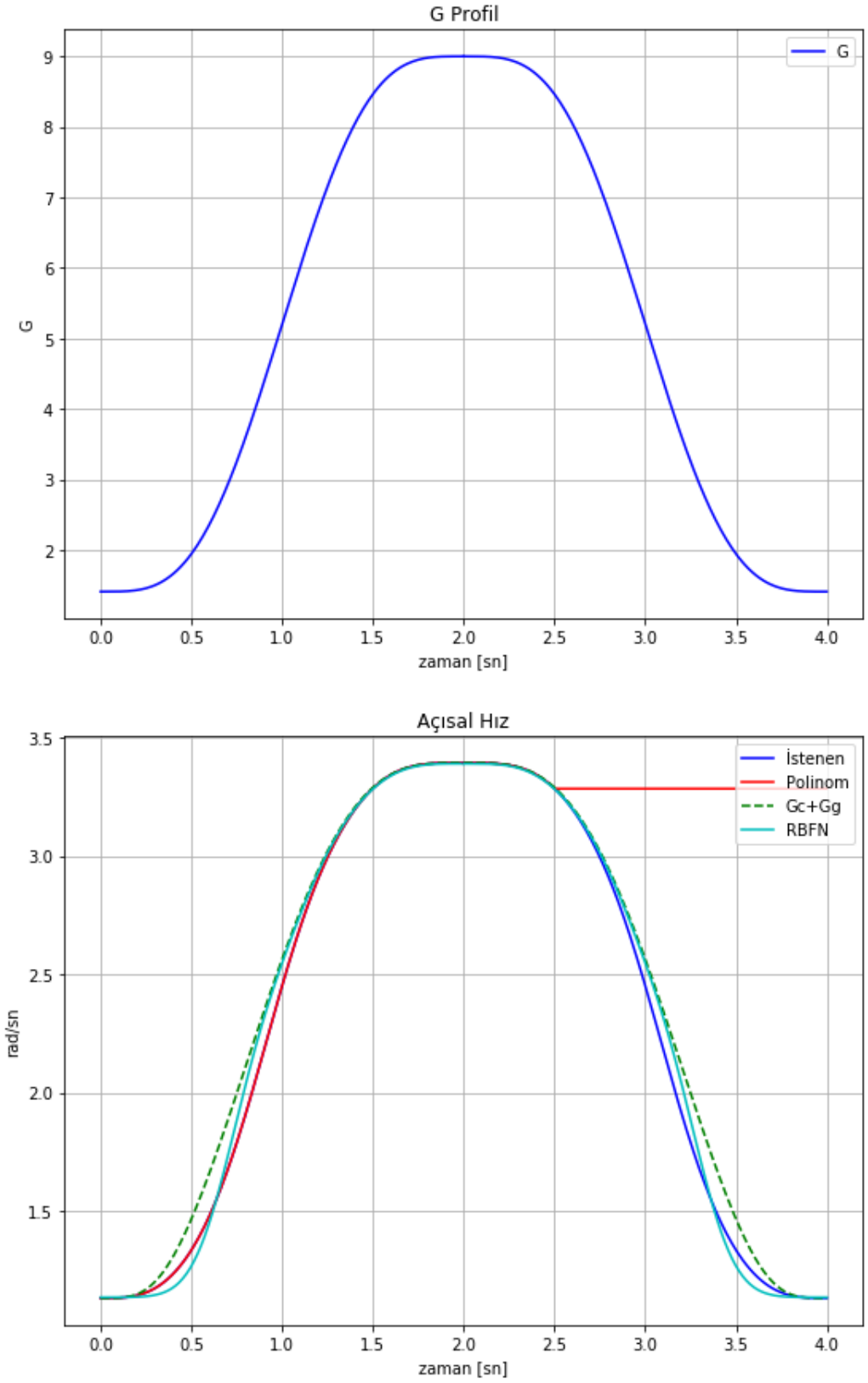
Şekil 5.15. YSA modeli kullanılarak açısal hız çözümü (uzun profil segment)



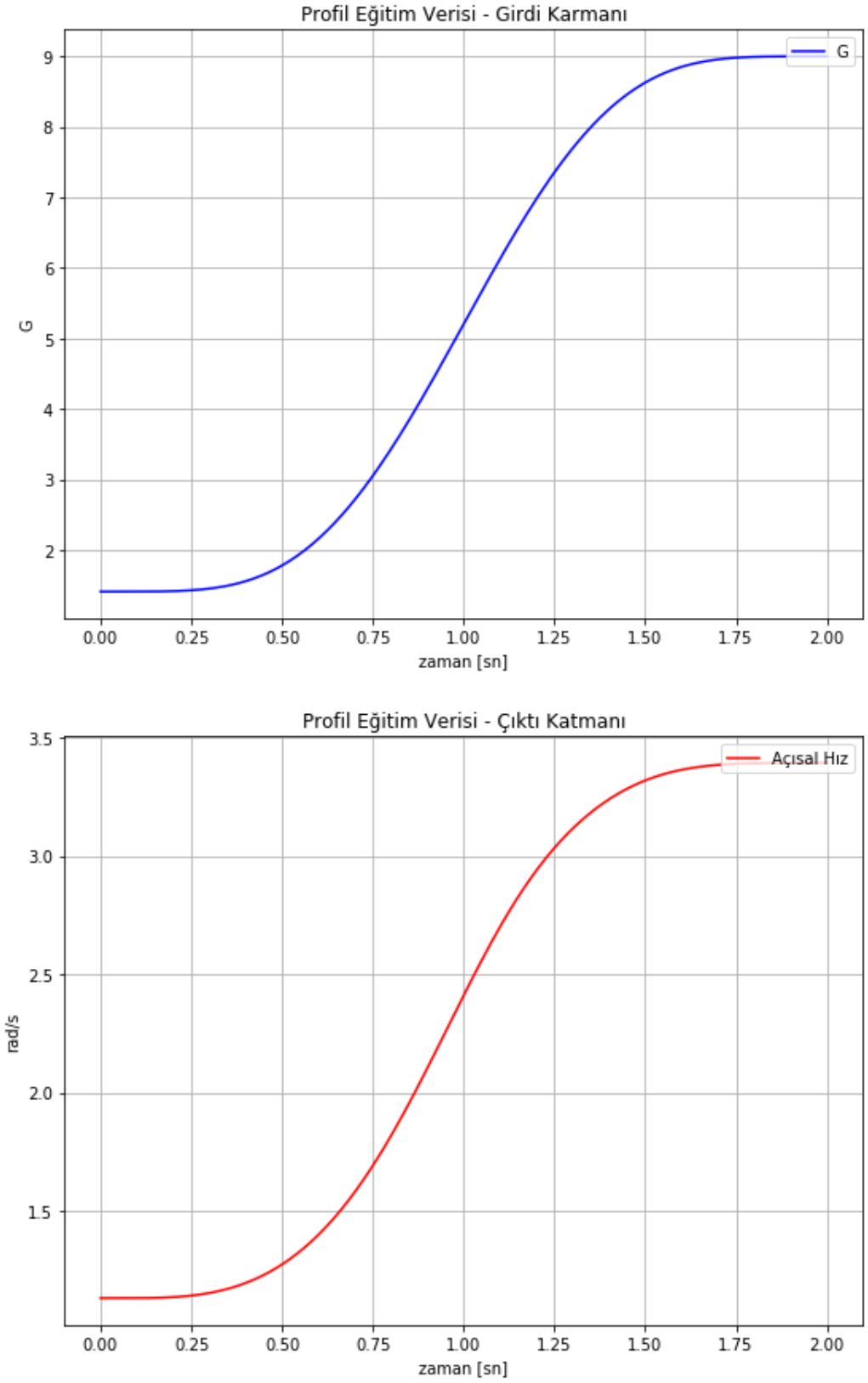
Şekil 5.16. YSA modeli kullanılarak uzun profil segment için açısal hız çözümü (G değişimi ve yunuslama (pitch) eksen açı verileri ile)



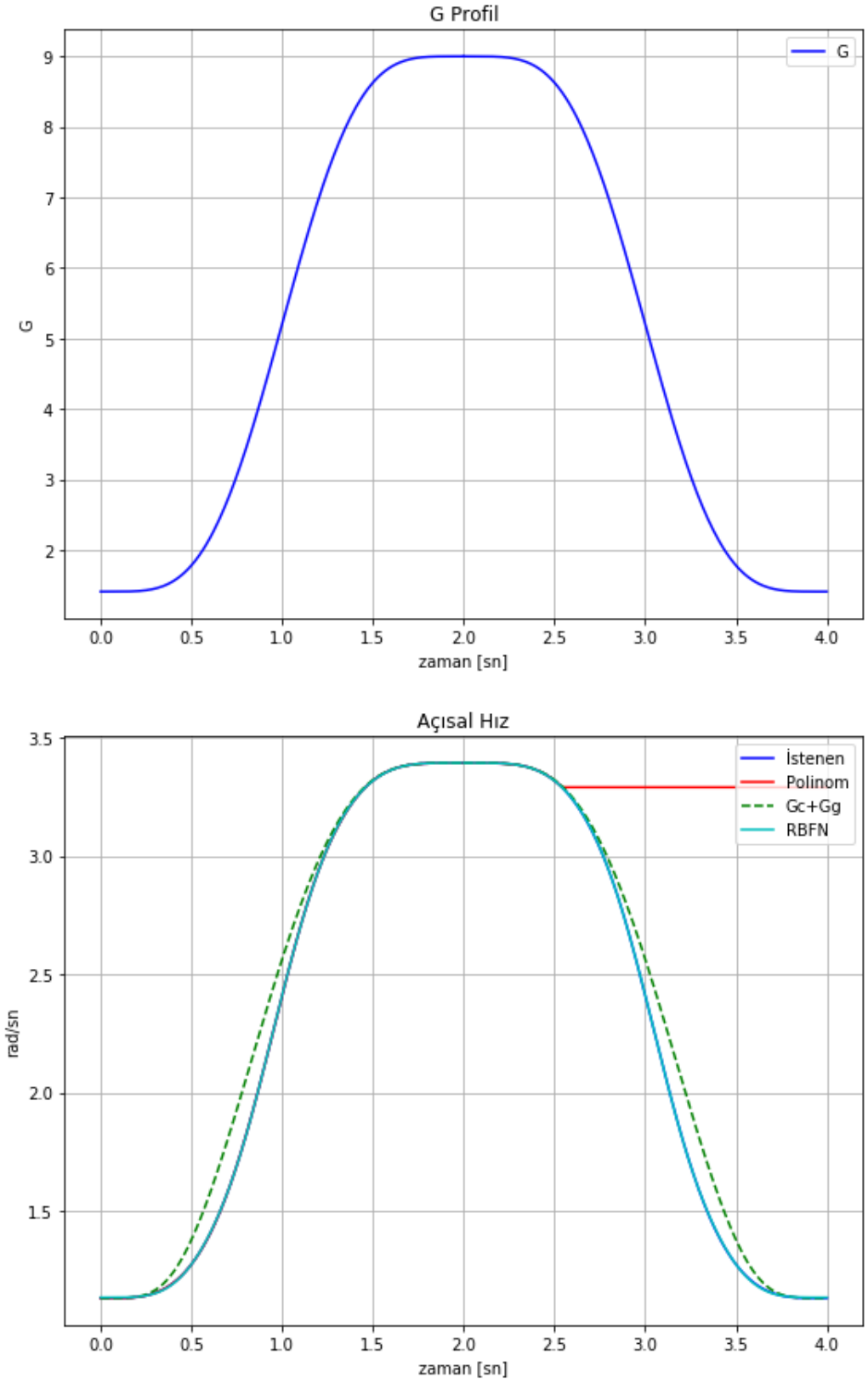
Şekil 5.17. RTFA eğitim verisi (doğrusal artan *G* komut dizisi)



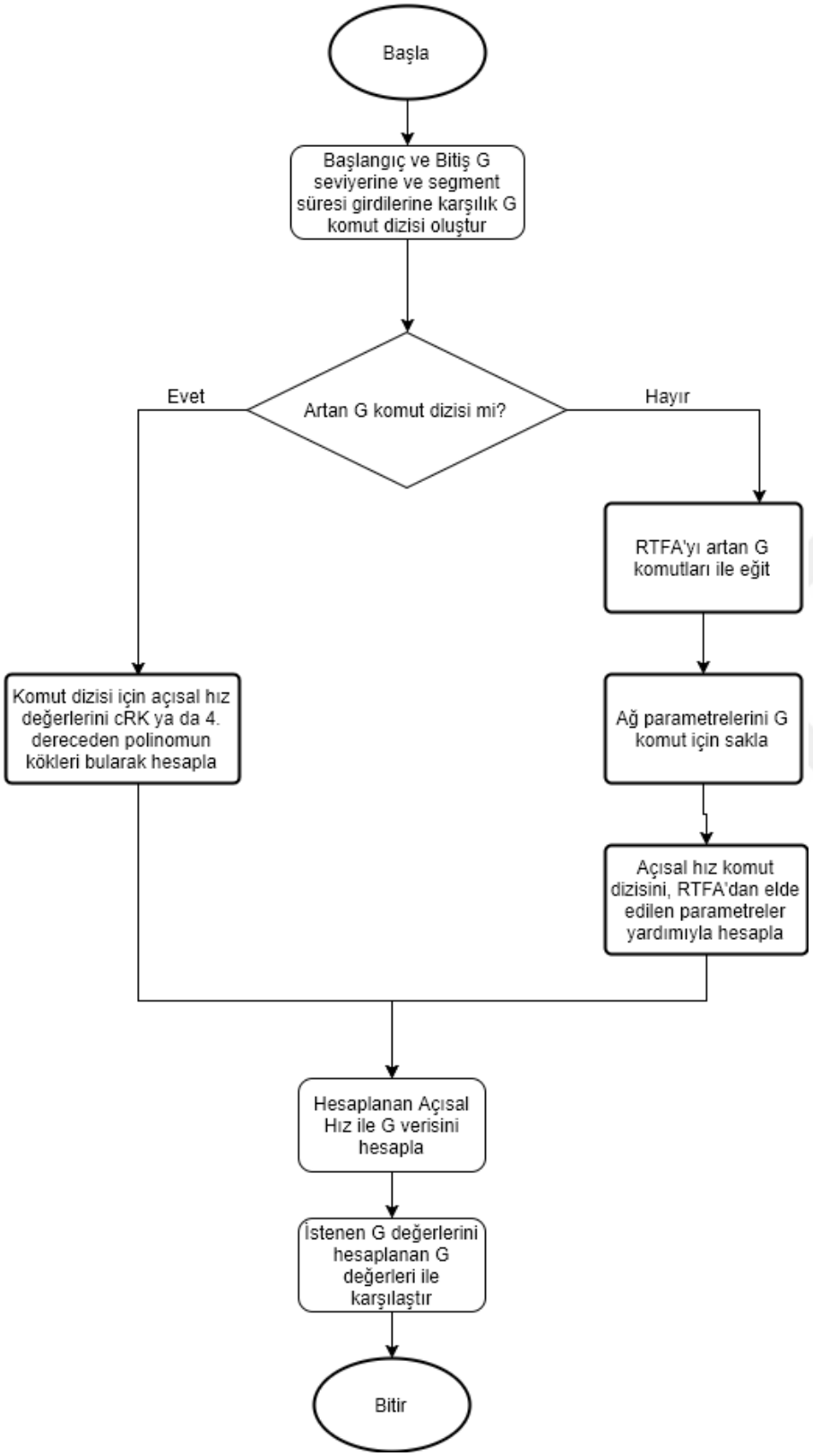
Şekil 5.18. Doğrusal artan *G* komutu ile eğitilen RBFA'nın S şeklinde *G* komutları için hesapladığı açısal hız grafiği



Şekil 5.19. RTFA eğitim verisi, S şeklinde artan *G* komut grafiği



Şekil 5.20. S şeklinde artan *G* komutu ile eğitilen ağın aynı tür *G* komutları için hesapladığı açısal hız grafiği



Şekil 5.21. İstenen *G* komutlarının RTFA modeli ve diğer yöntemler kullanılarak hesaplanan *G* komutları ile karşılaştırılması için kullanılan akış diyagramı

6 . SONUÇ VE ÖNERİLER

Pilot eğitimlerinde kullanılan İSS'nin gerçek uçuşlarda karşılaşılan kuvvetleri en gerçekçi şekilde simüle etmesi beklenir. Uçuş eğitiminde kullanılan profilin başarılı bir şekilde uygulanabilmesi için gerekli en önemli adım *G* verilerinden sistemin anlayacağı açısal hız verisinin hesaplanmasıdır. Artan *G* komutları için açısal hız değeri cRK ile istenen düzeyde hesaplanabildiği Vidakovic ve arkadaşlarının yaptıkları çalışmalarda gösterilmiştir [1, 2]. Bu çalışmada cRK'ya alternatif bir yöntem önerilmiştir. Bu yöntemde, her bir *G* komutu için dördüncü dereceden bir polinom denklem oluşturularak açısal hız hesaplanır (Eşitlik 4.5). Polinom denklem çözümü açısal hız değerini artan *G* komutları için tıpkı cRK gibi çok iyi çözmesine rağmen; azalan *G* komutları için köklerin hepsinin karmaşık kök olduğu çözümler sunması nedeniyle azalan *G* komutları için kullanılamaz. Polinom denklem kurularak kök bulma ve cRK yöntemleri açısal hızın azalan *G* komutlarına karşılık hesaplanmasında kullanılamaz. Her iki yöntemin de artan *G* komutları için kullanılabileceği yapılan testler ile gösterilmiştir.

Açısal hız değerinin azalan *G* komutları için neden hesaplanamadığı konusunda incelelemeler yapılmıştır. Kullanılan diferansiyel denklemin varlık teklik açısından incelenmiştir. Yapılan incelemeler sonucunda; diferansiyel denklemin teklik sağlayamağı için azalan *G* komutları için kullanılamayacağı gösterilmiştir (Bölüm 4.1). Düzgün dairesel hareket esnasında açısal hızın bütün durumlar için hesaplandığı görülmüştür. Düzgün olmayan dairesel hareket olması durumunda açısal hızın artan *G* komutlar için her zaman çözülebildiği; ancak azalan *G* komutları için çözümün reel sayılar kümesinde çözümü olmadığı yapılan testler ile gösterilmiştir. Problemi çözmek için nümerik yöntemler incelenmiş, açısal hız değerini hesaplayabilecek bir matematiksel modelin YSA kullanarak elde edilebileceği gösterilmiştir.

Teğetsel ivmenin ihmal edilmesi ile hesaplanan açısal hız değerinin kısa kollu sistemler için teğetsel ivmenin çok büyük değerler olmayacağı ve ulaşılan en yüksek G seviyelerinin düşük olacağı düşünülerek kullanıbileceği gösterilmiştir (Bölüm 4.8). Ancak uzun kollu sistemlerde yüksek yunuslama (pitch) ivme değerleri (Şekil 4.16) ve eğitim alan pilot üzerindeki olumsuz etkilerinden dolayı kullanılmaması gerekir. Bu eksenlerdeki yüksek ivme pilot eğitimlerinde istenmeyen G_x ve G_y kuvvetlerinin oluşmasına neden olmaktadır.

G eğitimlerinin öncelikli amacının yalnızca kafadan ayaklara doğru olan $+G_z$ kuvvetini uygulanması olarak düşünülürse teğetsel ivme uzun kollu sistemlerde ihmal edilmemelidir.

YSA ile oluşturulan ağ farklı parametreler ile test edilmiştir. İstenen G değeri ile birlikte istenen G değerinin birim zamanındaki değişimi ifade eden onset değeri girdi olarak kullanıldığında en iyi sonuçlar alınmıştır. Yalnızca G değerinin girdi olduğu ağlarda sonucun cRK ve teğetsel ivme ihmal edildiği yöntemlerden önemli derece bir iyileşme görülmemiştir. Eğitim verisi ağa tek tek vermek yerine küçük yığınlar halinde verildiğinde daha iyi sonuçlar alınmıştır (Bölüm 4.6). İki saniye ve üzerindeki *G* segmentler için YSA'nın diğer bütün yöntemlerden daha iyi sonuçlar elde ettiği yapılan testlerde görülmüştür. YSA modeli ile elde edilen açısal hız değerleri gerçek bir sistem üzerinde açık çevrim G profilleri için kullanılabileceği hazırlanan iki ayrı G profil editör uygulamasıyla gösterilmiştir. Oluşturulan YSA modeli ile hem artan hem azalan *G* komutları için çözüm üretir.

Bu çalışmada altının önemle çizilmesi gereken bazı konular mevcuttur. Oluşturulan YSA modeli, matematiksel formüller kullanarak elde edilen veriler ile oluşturulmuştur. Açısal hızın hesaplanmasında kullanılan matematiksel denklemlerde üç sabit değer bulunmaktadır. Bunlar; yer çekimi ivmesi, iki komut arasında geçen zaman ve sistem ana kol uzunluğu. Bu parametrelerden en önemlisi ana kol uzunluğudur. Ana kol uzunluğu parametresi her sistem için farklı olabileceği düşünülürse her sistem için oluşturulan YSA modelindeki ağırlıkların farklı olacağı bilinmelidir. İki komut arasındaki zaman dilimi eğitim verilerinde 0,01 sn olarak belirlenmiş, test verilerinde 0,1 sn, 0,5 sn ve 0,001 sn gibi farklı zaman aralıkları ile testler yapılmış ve YSA modelinin çözümüne bir etkisi olmadığı görülmüştür. Son parametre olan yer çekimi ivmesi her yerde aynı olduğu kabul edildiğinden girdi ve çıktı değerlere etkisi yoktur. Bu çalışmada oluşturulan model açık çevrim profil için kullanılabiliyorken, kapalı çevrim profil için her zaman çalışmayabilir. Açık çevrim profiller uygulanacak G komutlar önceden bilindiğinden YSA ile oluşturulan modelin performansı ölçülebilmektedir. Kapalı çevrim profillerde ise anlık olarak istenen G değerine karşılık hesaplanan açısal hız değerinin ne olacağı bilinemediğinden güvenlik açısından sakıncalar oluşturacaktır.

Yapılan araştırmalar sırasında karşılaşılan önemli problemler şunlardır: Açısal hız hesaplanmasında kullanılan diferansiyel denklem, azalan G komutları için çözüm üretemediğinden açısal hız hesaplamasında kullanılamaması, istenilen G değerini ile açısal

hız değeri arasındaki doğrusal olmayan ilişki, Oluşturulan 4. dereceden polinom denklemlerin azalan G değerleri için reel kökün bulunamaması, bütün köklerin kompleks köklerden oluşması, Euler, cRK gibi yöntemlerin azalan G komutları için açısal hız değerini çözememesi, Teğetsel ivme ihmal edildiğinde elde edilen açısal hız değerinin yunuslama (pitch) ekseni üzerindeki olumsuz etkileri, eğitimlerde istenilmeyen Gx ve Gy kuvvetlerine neden olması. YSA modeli, açık çevrim profiller için iki saniye ve üzerindeki segmentlerde iyi sonuçlar üretiyor. Uygulanacak olan G onset değerinin düşük olacağı garanti edilirse kapalı çevrim profiller için de kullanılabilir. Yapılan bu çalışmada açık çevrim profiller için YSA modeli oluşturulmuştur; ancak ileride yapılacak çalışmalarda kapalı çevrim profiller için farklı YSA modeller araştırılabilir. Pilot eğitimlerinin büyük kısmı açık çevrim profillerden oluştuğu düşünüldüğünde bu çalışma bu sistemlerde yer alan bu sorunun büyük bir kısmını çözebilir. Kapalı çevrim profillerde genellikle düşük onsetli komutlar uygulanmaktadır, düşük onsetli komutlar için teğetsel ivmenin ihmal edildiği denklem kullanılabilir. Diferansiyel denklemlerin çözümünde Fourier dönüşümü kullanılarak çözümler bulunabilmektedir. Bu çalışmada yer alan açısal hızın çözümünde kullanılan differansiyel denklem bu yöntem kullanılarak çözümü ileriki çalışmalarda aranabilir.

Bu çalışmada İSS'nin pilot eğitimleri açısından ne kadar önemli bir araç olduğu bir kez daha vurgulanmıştır. Gerçek bir uçuşun maliyeti ve olası kaza senaryoları göz önüne alındığında bu sistemlerin pilot eğitimlerinde vazgeçilmez olduğu görülmektedir. İSS'ler eğitim maliyetleri azaltmanın yanı sıra pilotların gerçek uçuşa daha hızlı bir şekilde hazır hale gelmeleri için kullanılmaktadır. Bu sistemlerin hayati öneme sahip olmalarına rağmen literatürde bu sistemler ile alakalı yapılan çalışmaların oldukça kısıtlı olduğu görülmüştür. İSS'ler hakkında bilgiler içermesi ve ileride bu sistemler konusunda yapılacak araştırmalara yardımcı olması açısından bu çalışma önemlidir. Ayrıca İSS'ler için önemli olan; ancak hesaplanamayan açısal hız değerinin hesaplanması amacıyla YSA tabanlı modellerin kullanılması, bu tip problemlerde YSA'nın kullanabileceği göstermektedir ve ileride yapılacak çalışmalara zemin hazırlamıştır.

KAYNAKLAR

1. Vidakovic, J., Ferenc, G., Lutovac, M., and Kvrgic, V. (2012). *Development and implementation of an algorithm for calculating angular velocity of main arm of human centrifuge*. 15th International Power Electronics and Motion Control Conference and Exposition, Novi Sad, Serbia, 1-6.

2. Vidakovic, J., Lazarevic, M., Kvrgic, V. M., Dancuo, Z., and Lutovac, M. (2013). Comparison of Numerical Simulation Models for Open Loop Flight Simulations in the Human Centrifuge. *Proceedings in Applied Mathematics and Mechanics*, 13(1), 485-486.

3. Landolt, J. P. (1990). High G Physiological Protection Training. *Aerospace Medical Panel, Advisory Group For Aerospace Research and Development*, 322, 1-74.

4. Secretary of the Air Force. (2014). *G Awareness for Aircrew*. USA: Department of the air force, AFPAM11-419, 4-23.

5. Kvrgic, V. M., Vidakovic, J. Z., Lutovac, M. M., Ferenc, G. Z., and Cvijanovic, V. B. (2014). A control algorithm for a centrifuge motion simulator. *Robotics and Computer-Integrated Manufacturing*, 30(4), 399-412.

6. Dan, Z. and Zeljkovi, V. (2012). High G Training Profiles in a High Performance Human Centrifuge. *Scientific Technical Review*, 62(1), 64-69.

7. Bishop, C.M. (2006). *Pattern recognition and machine learning*. USA: Springer, 225-291.

8. Bottou, L. (2010). *Large-Scale Machine Learning with Stochastic Gradient Descent*. Proceedings of Computational Statistics (COMPSTAT) 2010, Paris, France, 177-186.

9. Butcher, J. C. (1996). A history of Runge-Kutta methods. *Applied Numerical Mathematics*, 20, 247-260.

10. Oppenheim, A. V., Willsky, A. S., Nawab, S. H. (1996) *Signals And Systems (2nd Ed.).* USA: Prentice Halls, 361-427.

11. Butcher, J. C. (2008). *Numerical methods for ordinary differential equations, second edition*. New York: Wiley, 1-137.

12. Cammarota, J. (1990). *Evaluation of full-sortie closed-loop simulated aerial combat maneuvering on the human centrifuge*. IEEE Conference on Aerospace and Electronics, Army Research Laboratory, New York, USA, 838-842.

13. Caudill, M. (1987). Neural Networks Primer, Part I. *AI Expert*, 2(12), 46-52.

14. Cengiz, Y. and Sagiroglu, S. (2017). *System safety of human centrifuge and solving angular velocity of main arm with artificial neural network*. 2017 5th International Symposium on Digital Forensic and Security, Tirgu Mures, Romania, 1-6.

15. Crosbie, R.J., Colombo, J., and Black, P.E. (1991). *Motion base control process and pilot perceptual simulator. U.S. Patent No. US5021982A*. Washington, DC: U.S. Patent and Trademark Office.

16. Broomhead, D. and Lowe, D. (1988). Multivariable functional interpolation and adaptive networks. *Complex Systems*, (2), 321-355.

17. Durukan, M. (2008). *Jet Pilotlarına Uygulanan Anaerobik Antrenman Programlarının Bazı Bedensel ve Fizyolojik Parametreler ile G-toleransına olan Etkilerinin İncelenmesi*. Doktora Tezi, Gazi Üniversitesi Sağlık Bilimleri Enstitüsü, Ankara, 48-82.

18. Ercan, E., Ata, N., Dulkadir, Z., Ersal, Y., and Akin, A. (2010). *Turkish Aeromedical Center*. 2010 5th International Symposium on Health Informatics and Bioinformatics, Antalya, Turkey, 56-58.

19. Peterson, A. C. (1978). Existence-uniqueness for ordinary differential equations *Journal of Mathematical Analysis and Applications*, 64(1), 166-172.

20. Gultekin, S. S., Guney, K., and Sagiroglu, S. (2003). Neural Networks for the Calculation of Bandwidth of Rectangular Microstrip Antennas. *Applied Computational Electromagnetics Society Journal*, 18(2), 110-118.

21. Haykin, S. S. (1999). *Neural networks : a comprehensive foundation second edition*. USA: Prentice Hall, 139-340.

22. Heaton, J. (2008). *Introduction to Neural Networks with C#*. USA: Heaton Research, 99-166.

23. Hebb, D. (1949). *The Organization of Behavior. A neuropsychological theory*. New York: John Wiley, 17-37.

24. Iwasaki, K., Hirayanagi, K., Sasaki, T., Kinoue, T., Ito, M., Miyamoto, A.,Igarashi, M., and Yajima, K. (1998). Effects of repeated long duration +2Gz load on man's cardiovascular function. *Acta Astronautica*, 42(1-8), 175-183.

25. Karaboga, D., Guney, K., Sagiroglu, S., and Erler, M. (1999). Neural computation of resonant frequency of electrically thin and thick rectangular microstrip antennas. *IEE Proceedings - Microwaves, Antennas and Propagation*, 146(2), 155-159.

26. Gillingham, K., Fosdick J. (1988). High-G Training for Fighter Aircrew. *Science and Technology Committee Panel*, 59, 12-19.

27. Krishnaswamy, P.B. and Fegley, K.A. (1965). Reducing Undesirable Effects in the Human Centrifuge. *IEEE Transactions on Aerospace and Electronic Systems*, AES-1(1), 29-38.

28. Kumar, M. and Yadav, N. (2011). Multilayer perceptrons and radial basis function neural network methods for the solution of differential equations: A survey. *Computers and Mathematics with Applications*, 62(10), 3796-3811.

29. LeCun, Y.A., Bottou, L., Orr, G. B., and Muller, K.R. (2012). *Neural Networks: Tricks of the Trade, Efficient backprop)*, Berlin: Springer Berlin Heidelberg, 9-50.

30. Leverett, S.D. and Burton, R.R. (1979). *Life Sciences and Space Research*, Oxford: Pergamon Press, 171-185.

31. Levin, B. and Kiefer, D. (2002). Dynamic Flight Simulator for Enhanced Pilot Training. *SAFE Europe*, (March), 1-5.

32. Li, Y., Zhang, T., Wang, B., and Nakamura, M. (2012). *EEG and related physiological signals investigation under the condition of +Gz accelerations*. 2012 ICME International Conference on Complex Medical Engineering, Kobe, Japan, 60-65.

33. Li, Y.F., Yang, J.J., Zhang, L.H., Zhang, T., Deng, L., and Wang, B. (2015). Research of EEG change feature under +Gz acceleration. *Computers in Industry*, 70(2015), 144-152.

34. Lin, P.-C., Li, S.-C., Wang, J., and Chen, C.-C. (2010). *Measurement of military aircrews' ability to adapt to human-use centrifuge*. 40th International Conference on Computers and Industrial Engineering, Awaji City, Japan, 1-6.

35. Lin, P.C., Wang, J., and Li, S.C. (2012). Subjective stress factors in centrifuge training for military aircrews. *Applied Ergonomics*, 43(4), 658-663.

36. McCulloch, W.S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4), 115-133.

37. Modak, S., Singh, A., and Gomez, G. (2001). Human centrifuge: A tool for research and training. *Methods and Medicine*, 45(2), 101-105.

38. Nabiyev, V.V. (2012). *Yapay Zeka*. Ankara: Seçkin Yayıncılık, 565–599

39. Pain, S. (2006). The human centrifuge. *New Scientist*, 192(2576), 54-55.

40. Sagiroglu, S. (1998). Artificial Neural Networks in Robotic Applications. *Mathematical and Computational Applications*, 3(3), 67-81.

41. Serway, R., Kirkpatrick, L., and Jewett, J. (2008). *Physics for Scientists and Engineers with Modern Physics*. USA: Thomson Learning, 137-162

42. Swift, N. (2010). Centrifuging Mental Patients. *Annals of improbable research*, 20(3), 14-15.

43. Tsai, M.H. and Shih, M.C. (2009). Load tracking control of a centrifuge driven by servo hydraulic systems. *Proceedings of the Institution of Mechanical Engineers, Part G : Journal of Aerospace Engineering*, 223(6), 669-682.

44. Urone, P.P. and Hinrichs, R. (2012). *College Physics*. Texas: OpenStax, 199-239.

45. Wojtkowiak, M. (1981). The Application of Blood Flow Velocity Measurement under the Influence of + Gz acceleration. *Journal of Gravitational Physiology*, 24(6), 273-277.

46. Wu, Y., Wang, H., Zhang, B., and Du, K.L. (2012). Using Radial Basis Function Networks for Function Approximation and Classification. *International Scholarly Research Notices Applied Mathematics*, 2012, 1-34.

47. İnternet: odeint, başlangıç değer problemi çözmek için kullanılan C++ programlama dili kullanılarak hazırlanan kütüphane, URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.boost.org%2Fdoc%2Flibs%2F1_65_0%2Flibs%2Fnumeric%2Fodeint%2Fdoc%2Fhtml%2Findex.html&date=2017-11-30Son Erişim Tarihi: 30/11/2017.

48. İnternet: Octave, sayısal hesaplamalar için tasarlanmış yüksek seviyeli dil, URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.gnu.org%2Fsoftware%2Foctave%2F&date=2017-11-30, Son Erişim Tarihi: 30/11/2017.

49. İnternet: Python, URL: http://www.webcitation.org/query?url=https%3A%2F%2Fwww.python.org%2F&date=2017-11-30, Son Erişim Tarihi: 30/11/2017.

50. İnternet: C++ programlama dili tabanlı kullanıcı arayüzü geliştirme aracı, URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.qt.io%2F&date=2017-11-30, Son Erişim Tarihi: 30/11/2017.

51. İnternet: Runge-Kutta yöntemi, URL: http://www.webcitation.org/query?url=http%3A%2F%2Fen.wikipedia.org%2Fwiki%2FRunge%E2%80%93Kutta_methods&date=2017-11-30, Son Erişim Tarihi: 30/11/2017.

52. İnternet: SciPy, python tabanlı açık kaynak matematik kütüphanesi, URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.scipy.org&date=2017-11-30, Son Erişim Tarihi: 30/11/2017.

53. İnternet: Smoothstep fonksiyonları, S şeklinde fonksiyonlar, URL: http://www.webcitation.org/query?url=http%3A%2F%2Fen.wikipedia.org%2Fwiki%2FSmoothstep&date=2017-11-30, Son Erişim Tarihi: 30/11/2017.

54. İnternet: Wolfram Alpha tarafından geliştirilen hesaplama motoru,, URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.wolframalpha.com&date=2017-11-30, Son Erişim Tarihi: 30/11/2017.

EKLER

EK-1. G profil segment hazırlama

İSS'lerde kullanılan *G* profiller genellikle doğrusal komutlar olarak hazırlanmaktadır; ancak bu çalışmada farklı türde eğriler incelenerek denklemin çözümünün farklı eğrilere karşı nasıl sonuçlar verdiği de incelenmiştir. Aşağıdaki Kaynak Kod 6.1'de *G* komutları oluşturmak için kullanılmıştır.

Kaynak Kod 6.1. G komut oluşturmak için kullanılan kaynak kodlar

```
import numpy as np;

class SignalGenerator:
    def __init__(self):
        pass;
    def sin_wave(self, t, initial, final, duration):
        amplitude = final - initial;
        return initial + (amplitude * np.sin(np.pi / 2.0 * (t / duration)));
    def smooth_step_3(self, t, initial, final, duration):
        x = t / duration;
        np.clip(x, 0.0, 1.0, out = x);
        step = 3 * pow(x, 2) - 2 * pow(x, 3);
        return initial + step * (final - initial);
    def smooth_step_5(self, t, initial, final, duration):
        x = t / duration;
        np.clip(x, 0.0, 1.0, out = x);
        step = 6 * pow(x, 5) - 15 * pow(x, 4) + 10 * pow(x, 3);
        return initial + step * (final - initial);
    def smooth_step_7(self, t, initial, final, duration):
        x = t / duration;
        np.clip(x, 0.0, 1.0, out = x);
        step = -20 * pow(x, 7) + 70 * pow(x, 6) - 84 * pow(x, 5) + 35*pow(x,4);
        return initial + step * (final - initial);
    def smooth_step_9(self, t, initial, final, duration):
        x = t / duration;
        np.clip(x, 0.0, 1.0, out = x);
        step = 70 * pow(x, 9) - 315 * pow(x, 8) + 540 * pow(x, 7)
              - 420*pow(x,6) + 126 *pow(x,5);
        return initial + step * (final - initial);
    def sine_step(self, t, initial, final, duration):
        if ( t > duration).any() :
            return final;
        elif ( duration < 0 ):
            return initial;
        step = 0.5 + (np.sin((2 * (t / duration) - 1.0) * (np.pi / 2.0)) / 2.0);
        return initial + step * (final - initial);
    def linear(self, t, initial, final, duration):
        if ( t > duration).any():
            return final;
        elif ( duration < 0 ):
            return initial;
        else:
            deltaY = final - initial;
            return initial + (deltaY * t / duration);
```

EK-1. (devam) G profil segment hazırlama

Kullanılacak komutların doğrusal olması, hareketin başlangıcı ve sonunda çok sert hareketlere neden olabilir. Komutlar arası yumuşak bir geçiş olmasını sağlayacak yazılım kodu Kaynak Kod 6.2'de gösterilmiştir.

Kaynak Kod 6.2. S şeklinde *G* komut oluşturmak için kullanılan kaynak kodlar

```
from scipy import *
import math;
import numpy as np
from matplotlib import pyplot as plt

class SCurveProfile:
    def __init__(self, dt, Vi, Vf, maxA, AA):
        self.dt      = dt;
        self.Vi      = Vi;
        self.Vf      = Vf;
        self.Vsign   = np.sign(Vf-Vi);
        self.maxA    = maxA
        self.AA      = AA
        self.JA      = math.pow(maxA,2) * AA / ((Vf-Vi)*(maxA-AA))
        self.t1      = self.Vsign * self.maxA / self.JA;
        self.t2      = self.Vsign * ((Vf-Vi)/self.AA) - self.t1;
        self.t3      = self.Vsign * (Vf-Vi)/self.AA;

        self.currentJ = 0.0;
        self.currentA = 0.0;
        self.currentV = 0;

    def process(self, t):
        self.currentJ = 0;

        if ( t <= self.t1 ):
            self.currentA = self.JA * t;
            self.currentV = self.Vi + 0.5 * self.JA * pow(t,2);
        elif ( t <= self.t2) :
            self.currentV = self.Vi + math.pow(self.maxA,2)/(2*self.JA)
                          + self.Vsign * self.maxA * (t-self.t1);
            self.currentA = self.Vsign * self.maxA;
        elif ( t <= self.t3 ) :
            self.currentA = self.Vsign * self.maxA - self.JA *(t-self.t2);
            self.currentV = self.Vi + (self.Vf-self.Vi)
                          - 0.5 * self.JA * pow(self.t3-t,2);
        else:
            self.currentA = 0;
```

EK-2. Açısal hız çözümü için RTFA

Kaynak Kod 6.3. RBFA Kaynak Kod

```
import csv
import random as rand
import math
import matplotlib.pyplot as plt
import numpy as np
import matplotlib.mlab as mlab
import operator

from scipy.interpolate import Rbf
from scipy import *


def LoadCSV(filename):
    lines = csv.reader(open(filename, "r"))
    dataset = list(lines);

    for i in range(len(dataset)):
        dataset[i] = [float(x) for x in dataset[i]]

    return dataset

if __name__ == '__main__':
    filename  = '.\\DATA15G_9Gs_training.csv'
    trainingSet = LoadCSV(filename);

    x = np.matrix(trainingSet)[:,0]
    y = np.matrix(trainingSet)[:,1]

    rbfi      = Rbf(x,y, function='multiquadric')

    filename   = '.\\DATA15G_9Gs_training.csv'
    testingSet = LoadCSV(filename);

    x_test = np.matrix(testingSet)[:,0];
    y_test = np.matrix(testingSet)[:,1];

    #zPredicted = rbfi(x_test, y_test)
    yPredicted = rbfi(x_test)
    error = y_test - yPredicted;
    plt.figure(figsize=(12, 8))
    plt.plot(y_test, 'g-', linewidth=2)
    plt.plot(yPredicted, 'r-', linewidth=2)
    plt.plot(error, 'b-', linewidth=2)
    #plt.plot(t, y, 'k-')
```

Kaynak Kod 6.4. Aktivasyon fonksiyonlarını elde etmek için kullanılan kaynak kodlar

```
import numpy as np;

class ActivationFunction:
    def __init__(self):
        pass;
    import numpy as np

    def sigmoid(self, x, derivative=False):
        if (derivative == True):
            z = 1 / (1 + np.exp(-x))
            return z * (1 - z)
        return 1 / (1 + np.exp(-x))

    def tanh(self, x, derivative=False):
        if (derivative == True):
            z= np.tanh(x);
            return (1 - (z ** 2))
        return np.tanh(x)

    def linear(self, x, derivative=False):
        out = np.zeros(len(x));
        if (derivative == True):
            for i in range(0, len(x)):
                out[i] = 1
            return out

        for i in range(0, len(x)):
            out[i] = x[i]
        return out

    def step(self, x, derivative=False):
        out = np.zeros(len(x));
        if (derivative == True):
            for i in range(0, len(x)):
                if x[i] > 0:
                    out[i] = 0
            return out
        for i in range(0, len(x)):
            if x[i] > 0:
                out[i] = 1
            else:
                out[i] = 0
        return out
```

Kaynak Kod 6.5. Profil Editör

```
import globals;
from profileparser import *;
from signalgenerator import SignalGenerator;
import numpy as np;
from matplotlib import pyplot as plt
from plotmanager import PlotManager
from omegasolver import OmegaSolver;
from rbfn import RBFN;

solverList = ['actual', 'polynomial', 'omega2d']
curveTypeList = ['b-', 'r-', 'g-']

class ProfileEditor:
    def __init__(self):
        self.profileParser = ProfileParser();
        self.rbfn = RBFN(1,8,1);
        self.initialize_rbfn();
    def load(self, profile_file_name):
        segmentList = self.profileParser.parse(profile_file_name);
        sg = SignalGenerator();
        profileTime = 0.0;
        initialG   = globals.MIN_G;
        plotmanager = PlotManager()
        omegaSolver = OmegaSolver();
        for i in range(len(segmentList)):
            gSegment   = segmentList[i];
            t          = np.linspace(0.0, gSegment.duration,
                                     gSegment.duration * (1.0/globals.dt))
            segmentTime = [x + profileTime for x in t]
            desiredG   = sg.linear(t,initialG, gSegment.finalG,
                                   gSegment.duration)
            plotmanager.axes[0].plot(segmentTime, desiredG, '-b')
            for j in range(len(solverList)):
                calculatedOmega = omegaSolver.solve(solverList[j], desiredG);
                xxx = desiredG / 20.0;
                yyy = self.rbfn.predict(np.reshape(xxx, (len(xxx), 1)));
                plotmanager.axes[1].plot(segmentTime, calculatedOmega,
                                         curveTypeList[j])
                plotmanager.axes[1].plot(segmentTime, yyy, 'k-')

            initialG = gSegment.finalG;
            profileTime = profileTime + gSegment.duration;
        plt.show()
    def initialize_rbfn(self):
        sg         = SignalGenerator();
        omegaSolver = OmegaSolver();
        duration = 2.0
        t = np.linspace(0.0, duration, duration * (1.0/globals.dt))
        x = sg.linear(t,globals.MIN_G, 9.0, duration)
        y = omegaSolver.solve('actual', x);
        self.rbfn.train(np.reshape(x/20.0, (len(x), 1)), y)
```

EK-5. RTFA örnek

Kaynak Kod 6.6. RTFA örnek

```
from scipy import *
from scipy.linalg import norm, pinv
from matplotlib import pyplot as plt
class RBF:
    def __init__(self, in_dimention, number_of_centers, out_dimention):
        self.in_dimention = in_dimention
        self.out_dimention = out_dimention
        self.number_of_centers = number_of_centers
        self.centers = [random.uniform(-1, 1, in_dimention)
            for i in range(number_of_centers)]
        self.beta = 10
        self.W = random.random((self.number_of_centers, self.out_dimention))
    def _basis_function(self, c, d):
        assert len(d) == self.in_dimention
        return exp(-self.beta * norm(c-d)**2)
    def _calculate_activation_function(self, X):
        G = zeros((X.shape[0], self.number_of_centers), float)
        for ci, c in enumerate(self.centers):
            for xi, x in enumerate(X):
                G[xi,ci] = self._basis_function(c, x)
        return G
    def train(self, X, Y):
        random_centers = random.permutation(X.shape[0])[:self.number_of_centers]
        self.centers = [X[i,:] for i in random_centers]
        G = self._calculate_activation_function(X)
        self.W = dot(pinv(G), Y)
    def test(self, X):
        G = self._calculate_activation_function(X)
        Y = dot(G, self.W)
        return Y
if __name__ == '__main__':
    n = 100
    trainingStart = 0; testStart  = 0;
    trainingEnd  = 1; testEnd     = 1;
    t = mgrid[trainingStart:trainingEnd:complex(0,n)].reshape(n, 1);
    y = sin(2*(t+0.5)**3)
    rbf = RBF(1, 10, 1)
    rbf.train(t, y)
    x = mgrid[testStart:testEnd:complex(0,n)].reshape(n, 1);
    z = rbf.test(x)
    plt.figure(figsize=(12, 8))
    plt.plot(t, y, 'ko')
    plt.plot(x, z, 'r-', linewidth=1)
    plt.plot(rbf.centers, zeros(rbf.number_of_centers), 'cs')
    xAxisStart = min(trainingStart, testStart) -0.5;
    xAxisEnd  = min(trainingEnd, testEnd) +0.5;
    plt.xlim(xAxisStart, xAxisEnd)
    plt.xlabel('x'); plt.ylabel('y')
    plt.show()
```

Kaynak Kod 6.7. Gradyan iniş algoritması

```
#include <QtCore/QCoreApplication>
#include<iostream>
#include<cmath>
using namespace std;

double derivative(double x)
{
    return 2 * x;
}

void findMinimum(double &x, double minErrorLevel = 0.01,
                 double step = 0.1, int maxStepCount = 100)
{
    double prevX   = 0;
    double diff    = 1;
    int    stepCount = 0;
    bool   stop = false;
    while ( !stop && stepCount < maxStepCount )
    {
        if ( diff > minErrorLevel )
        {
            prevX = x;
            x     = x - step * derivative(x);
            diff  = abs(x - prevX);
            cout << "x: " << x << ", derivative :" << derivative(x) << endl;
        }
        else
        {
            stop = true;
        }
    }
}

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);
    double x = -10;
    findMinimum(x);
    cout << "Minimum:" << x;
    return a.exec();
}
```

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı : CENGİZ, Yakup

Uyruğu : T.C.

Medeni hali : Evli

Telefon : 0 (544) 785 33 10

e-mail : yakupcengiz@gmail.com

Eğitim

Derece	Eğitim Birimi	Mezuniyet Tarihi
Yüksek Lisans	Gazi Üniversitesi / Bilgisayar Mühendisliği	Devam ediyor
Lisans	Kocaeli Üniversitesi / Bilgisayar Mühendisliği	2008

İş Deneyimi

Yıl	Yer	Görev
2009-Halen	ETC Türkiye Şb.	Mühendis

Yabancı Dil

İngilizce

Yayınlar

Cengiz, Y. and Sagiroglu, S. (2017). *System safety of human centrifuge and solving angular velocity of main arm with artificial neural network*. 2017 5th International Symposium on Digital Forensic and Security, 1-6.

GAZİ GELECEKTİR...