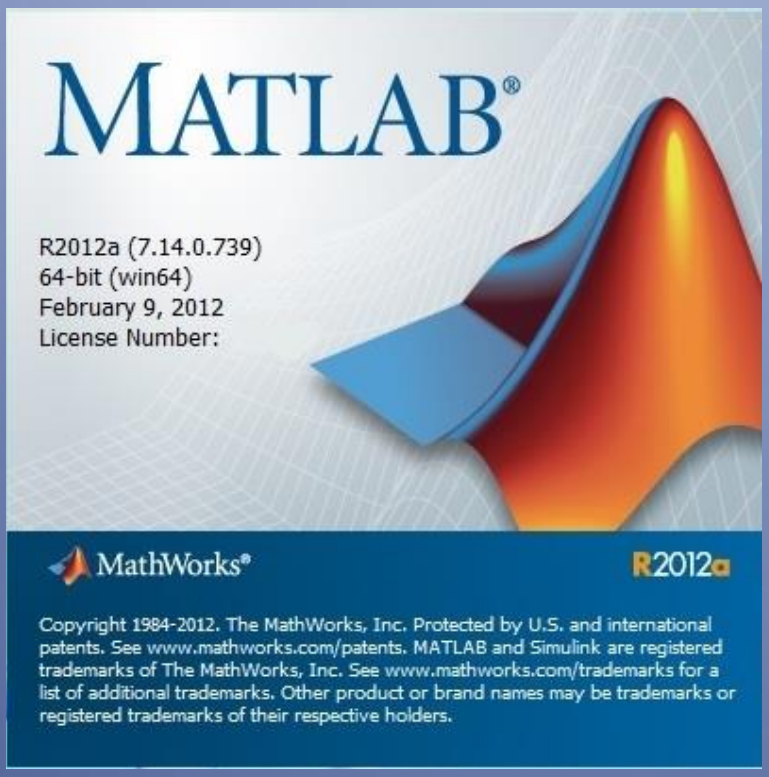


ELEKTRONİK VE BİLGİSAYAR BİLİMLERİNDE SAYISAL ÇÖZÜMLER DERS NOTLARI

Önsöz

MATLAB, MATrix LABoratory sözcüklerinden gelir. İlk defa 1985'de C.B. Moler tarafından geliştirilmiş etkileşimli bir paket programlama dilidir. Temelde sayısal ve analitik olarak matematiksel fonksiyonların ifadelerinin kullanıldığı başta mühendislik alanında olmak üzere sayısal analiz yöntemleri kullanılan bilimlerde son yıllarda oldukça sık kullanılan bir yazılımdır.

Elinizdeki bu kitap matlab ilgili bütün eğitimlerde yardımcı olması amacıyla derlenmiştir. Öncelikle danışman hocam Doç Dr. İlhan TARIMER'in ders notlarından yola çıkarak internet üzerinde yapılan geniş kapsamlı bir tarama sonucunda derlenen notların size yardımcı olmasını dilerim. İnternet üzerinden yararlandığım kaynakların hepsini kaynakçada belirtmeye çalıştım ancak yoğun tempo sırasında yazmayı atladığım kaynaklar olduysa şimdiden özür dilerim.

İçindekiler

Şekiller Listesi

Tablo Listesi

Tavsiye Edilen Ders Planı

HAFTA	BÖLÜM ADI
I. Hafta	Arayüz, Operatörler, İşlem Öncelikleri, Sabitler
II. Hafta	Matrislere Değer Yükleme, İndis, Boyutları
III. Hafta	Vektör ve Matrislerde Aritmetik İşlemler, Determinant, Inverse
IV. Hafta	Built In Fonksiyonlar
V. Hafta	M Dosyaları – Script
VI. Hafta	M Dosyaları - Function
VII. Hafta	Grafikler – 2D.
VIII. Hafta	Grafikler – 2D
IX. Hafta	Grafikler – 3D
X. Hafta	Matlab'da Programlama – Giriş Çıkış Deyimleri
XI. Hafta	Matlab'da Programlama – Koşul Deyimleri, Döngü Deyimleri
XII. Hafta	Polinomlar – Kökleri, Katsayıları
XIII. Hafta	Polinomlar – Çarpma, Bölme, Türev
XIV. Hafta	Polinomlar – İntegral, Diferansiyel

1. BAŞLARKEN

Mathwork firması tarafından geliştirilen MATLAB programının 2005 yılından itibaren tüm eklenti ve güncellemeleri 6 ayda bir yeni bir sürüm şeklinde şeklinde piyasaya sürülmektedir. Yılın ilk 6 ayında ki sürüm için a, ikinci 6 ayındaki sürüm için b kodunu kullanır. Bu döküman 2012 a versiyonu kullanılarak hazırlanmıştır.

Bu bölümde öncelikle MATLAB hakkında kısaca bilgi verdikten sonra sırasıyla MATLAB kullanıcı arayüzünü, MATLAB'da değişken tanımlarken dikkat edeceğimiz hususları ve değişkenlerle ilgili MATLAB komutlarını, kullanabileceğimiz aritmetiksel ve mantıksal öperatörler ile işlem önceliklerini ve son olarak da MATLAB'da tanımlı sabitler hakkında bilgi vereceğiz.

MATLAB numerik hesaplamarı, ileri düzey grafikleri ve görüntülemeyi, ve üst düzey programlama dilini birleştiren, entegre teknik hesaplama ortamıdır. MATLAB'ın temelindeki yapı boyutlandırma gerektirmeyen matrislerdir. Yapılan tüm girdi ve çıktılar hiçbir bildirim gerektirmeden bir matris olarak tanımlanır.

MATLAB'ın kullanım yerlerini saymak gerekirse;

- Denklem takımlarının çözümü, doğrusal ve doğrusal olmayan diferansiyel denklerinin çözümü, integral hesabı gibi sayısal hesaplamalar
- Veri çözümleme işlemleri
- İstatiksel hesaplamalar ve çözümlemeler
- Grafik çizimi ve çözümlemeler
- Bilgisayar destekli denetim sitemi tasarımı
- Devre analizinde düğüm kol denklemlerinin çözümü

MATLAB programının profesyonel ve öğrenci versiyonu olmak üzere iki farklı versiyonu mevcuttur.

MATLAB programını çalıştırdığımızda karşımaza gelen arayüzün(Şekil 1.1.1.) büyük kısmını komut ekranı(Command Window) penceresi oluşturur. Komut ekranında gördüğümüz ">>" işaretine prompt adı verilir. MATLAB programının öğrenci versiyonunda prompt "**EDU**>>" şeklinde görünür.

Prompt satırında istenilen komut kullanılıp sonucun görünmesi sağlanır. Ayrıca yazılan komutlardan oluşan programlar kaydedilir.

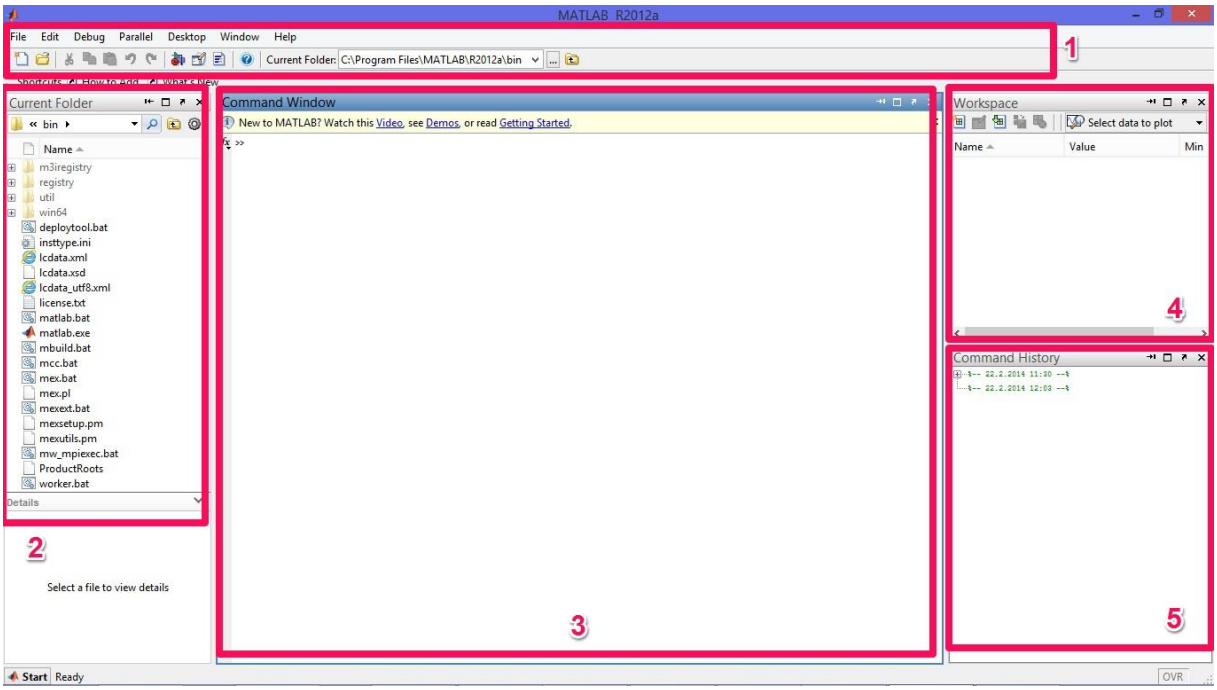
Başlangıç aşamasında aşağıdaki komutlar yardımıyla MATLAB'dan yardım alınılabilir.

- **info:** Bilgi almak için gerekli iletişim bilgilerini gösterir.
- **demo:** MATLAB tarafından hazırlanan eğitim videolarına erişilmesini sağlar.
- **help:** Herhangi bir komutun kullanımı hakkında bilgi alınmasını sağlar, help fonksiyon_adı yazarak komut hakkında ayrıntılı bilgi alınır.
- **helpbrowser:** Yardım ekranını açılmasını sağlar
- **lookfor:** Belirli bir sözcük yada deyim için yardım arar

MATLAB'ı kapatmak istenildiğinde komut satırında **quit** ya da **exit** yazmak veya Dosya menüsünden Exit MATLAB'ın seçilmesi gerekir.

1.1. Arayüz

Şekil 1.1. de MATLAB ayayüzü görülmektedir.



Şekil 1.1. MATLAB Arayüz

Şekil 1.1. de görüldüğü üzere MATLAB arayüzü 5 ana kısımdan oluşmaktadır. Bunlar;

- **Menu:** Ana menünün bulunduğu pencere
- **Current Folder:** MATLAB'ın çalışma klasörü içinde bulunan dosyaları gösterir.
- **Command Window:** MATLAB komutlarının çalıştırılabildiği pencere
- **Workspace:** Aktif olarak kullanılan değişkenlerin isimlerini ve değerlerini gösterir.
- **Command History :** Yapılan bütün işlemlerin kaydını görelebildiği pencere.

1.1.1 Command Window

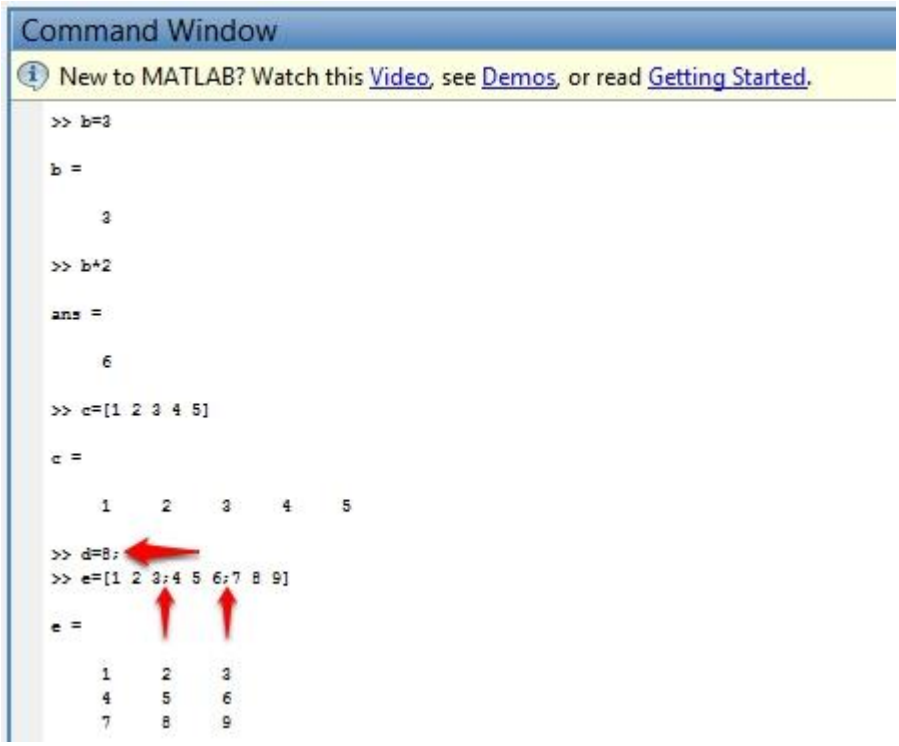
Şekil 1.2. de Command Window ekranı görülmektedir.

Şekil 1.2. Command Window

Command Window penceresinden komutlar veya programlar çalıştırılır ayrıca değişken işlemleri de bu pencereden gerçekleştirilir. Command Windows ekranını temizlemek için **clc** komutu kullanılır.

1.1.1.1 Command Window Ekranında Değişken İşlemleri

Şekil 1.3. de Command Window ekranında değişken tanımlama işlemleri görülmektedir.



Şekil 1.3. Değişken Tanımlama

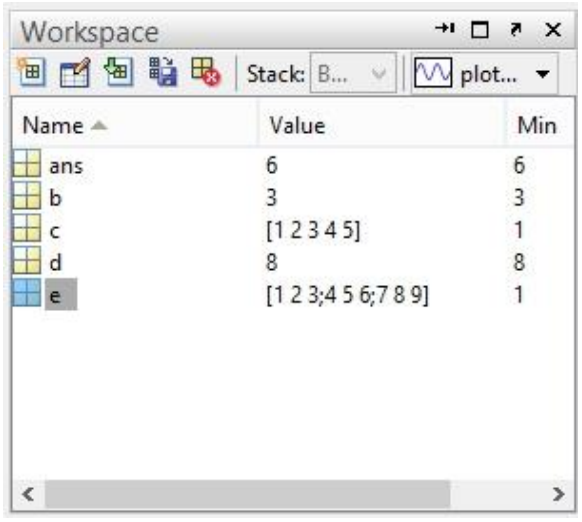
Değişkenler atanan ya da hesaplanan sonucu tutmakla görevli olan yapılardır. Şekil 1.3. de Command Window ekranında b, c, d, e olmak üzere 4 farklı değişken tanımlanıp bunlara değer atanmıştır. Değişkenlere değer atama operatörü olarak "=" işareti kullanılır. Değişkenlerimiz skaler, vektör ya da matris şeklinde olabilir. Değişkeni tanımladıktan sonra çıktının ekranda görünmesi istenmiyorsa satır sonuna "**;**" işareti eklenir. Değişken isimlerinde büyük küçük harf duyarlılığı vardır. MATLAB Command Window ekranında yapılan işlem herhangi bir değişken atanmazsa sonuç ans değişkeni içinde tutulur.

MATLAB da değişkenler için kullanılan komutlar;

- **who:** Hafizadaki değişkenleri listeler.
- **whos:** Hafizadaki değişkenleri boyutlarıyla beraber listeler.
- **clear:** Hafizadaki değişkenleri siler.
- **save:** Hafizadaki değişkenleri kaydeder.

1.1.2 Workspace

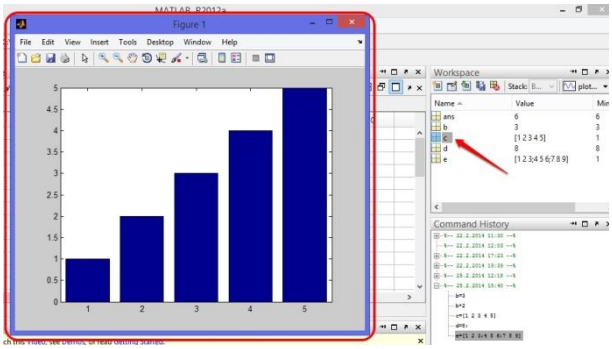
Şekil 1.4. de Workspace ekranı görülmektedir.



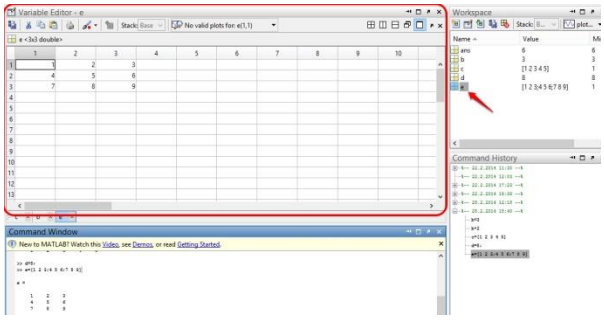
Şekil 1.4. Workspace

Workspace ekranı kullanılan değişkenlerin isimlerinin ve değerlerinin görüldüğü ekrandır. Bu ekranı kullanarak değişken değerlerini görebilmenin yanı sıra üzerinde ters tıklanarak değişkenleri silebilir, isimleri ya da değerleri değiştirilebilir.

Şekil 1.5. de bir değişkenin grafiği, Şekil 1.6. da ise Variable Editör ekranı görülmektedir.



Şekil 1.5. Değişken Grafiği



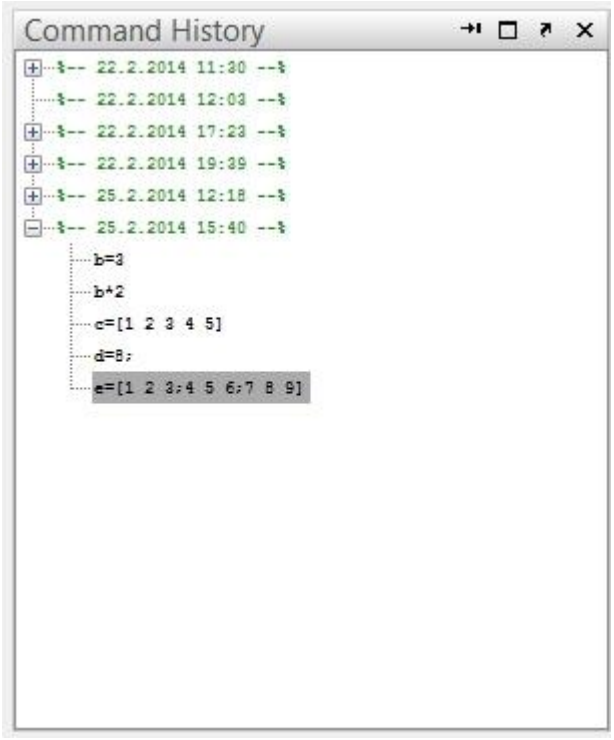
Şekil 1.6. Variable Editor

Şekil 1.5. de Workspace ekranında listelenen değişkenlerden c vektörü üzerinde ters tıklanıp değişkenin grafiği çizilmiştir. Şekil 1.6. da Workspace ekranında e matrisi üzerinde çift tıklanılıp Command Window penceresinin hemen üstünde Variable Editor penceresinin açılması sağlanmıştır.

Variable Editor penceresini kullanarak değişkenlerin değerleri değiştirilebilir.

1.1.3 Command History

Şekil 1.7. de Command History ekranı görülmektedir.



Şekil 1.7. Command History

Command Window ekranında yaptığımız bütün işlemleri Command History ekranında görebiliriz.

1.2 Operatörler

1.2.1 Aritmatiksel Operatörler

Tablo 1.1. Aritmetiksel Operatörler

İşlem	Cebirsel Biçimi	MATLAB Karşılığı
Toplama	a+b	a+b
Çıkarma	a-b	a-b
Çarpma	axb	a*b
Bölme	a/b	a/b
Sola Bölme	b/a	a\b
Üs Alma	a^b	a^b

1.2.2 İlişki Operatörleri

Tablo 1.2. İlişki Operatörleri

İşlem	Karşılığı
Eşit	==
Eşit Değil	~=
Küçük	<
Büyük	>
Küçük Eşit	<=
Büyük Eşit	>=

1.2.3 Mantıksal Operatörler

Tablo 1.3. Mantıksal Operatörler

İşlem	Karşılığı
Ve	&
Veya	|
Değil	~

1.3 İşlem Öncelikleri

Tablo 1.4. İşlem Öncelikleri

Öncelik	İşlem
1	Parantez
2	Üs alma, soldan sağa doğru
3	Çarpma ve bölme, soldan sağa doğru
4	Toplama ve çıkarma, soldan sağa doğru

1.4 Sabitler

Değişkenler isminden de anlaşılacağı gibi sabit olmayan değerleri depolamak için kullanılır. Değişkene atanan değeri ya direkt olarak kullanıcı belirler ya da bir işlem sonucu hesaplanabilir. Ancak bazı değerler vardır ki bunlar kullanıcının belirlemesine yada hesaplamasına gerek olmadan bir sabite atanır. Kısaca sabitlere değeri değişmeyen değişken denilebilir. MATLAB'da kullanılan sabitlerden bir kısmı aşağıda sıralanmıştır.

Tablo 1.5. Sabitler

Sabit Adı	Değeri
Pi	3,14
İnf	Sonsuz
Nan	Rakam olmayan, Tanımsız 0/0
Eps	İki sayı arası en ufak fark $2.2204*e^{16}$
i, j	Karmaşık sayıların sanal kısmı
Clock	Vektör olarak zaman bilgisi
Date	Dizi olarak tarih bilgisi

2.MATRİSLER

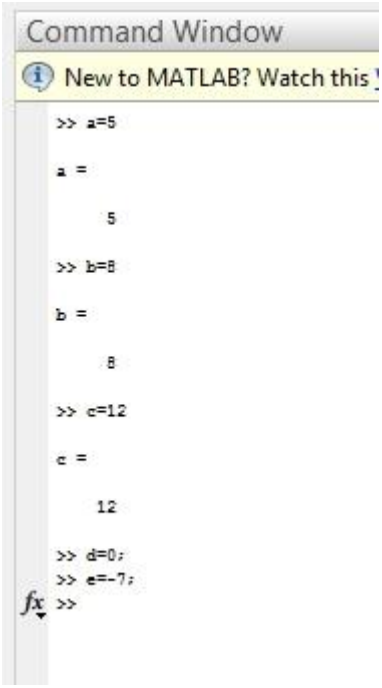
MATLAB programında bütün değişkenler matris olarak kabul edilir. Matrisler satır ve sütünlardan oluşan sayı dizileridir. MATLAB'da genel olarak kullanılan veri tipleri vektörler ve matrislerdir. Matris boyutları satır sayısı (row) x sütün sayısı (column) r x c şeklinde ifade edilir.

Boyutu 1x1 olarak ifade değilen matrislere skaler adı verilir. Skalerler sadece tek bir değer barındırır. Örneğin 4 sayısı boyutu 1x1 olan matristir.

Şekil 2.1. Skaler matris örnekleri, Şekil 2.2. de ise MATLAB da skaler değişken tanımlama işlemi görülmektedir.

5	8	12	0	-7

Şekil 2.1. Skaler Matris Örnekleri



Şekil 2.2. Skaler matrislere değer yükleme

Şekil 2.2. de Command Window ekranında a, b, c, d, e olmak üzere 5 farklı skaler değişken tanımlanıp bunlara değer atanmıştır.

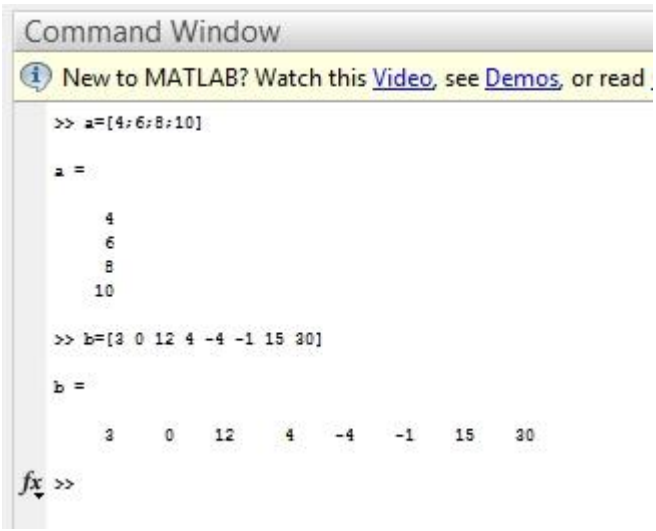
Tek bir satır (1 x n) ya da tek bir sütundan(n x 1) oluşan matrislere vektör adı verilir.

Şekil 2.3. de vektör örnekleri, Şekil 2.4. de ise MATLAB da vektörel değişkenlere değer atama işlemi görülmektedir.

4
6
8
10

3	0	12	4	-4	-1	15	30

Şekil 2.3. 4x1 ve 1x8 Vektör Örnekleri



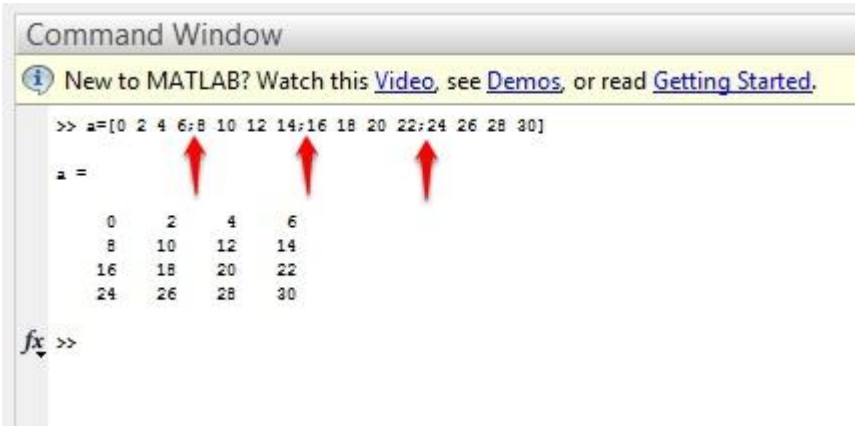
Şekil 2.4. 4x1 ve 1x8 Vektöre Değer Yükleme

Şekil 2.4. de Command Window ekranında a ve b olmak üzere 2 farklı vektörel değişken tanımlanıp bunlara değer atanmıştır. Sütün vektörü yaratmak için ‘;’ operatörü kullanılmıştır.

Şekil 2.5. de 3x4 lük bir matris , Şekil 2.6. da ise 4x4 lük bir matrisin MATLAB da tanımlanması görülmektedir.

0	2	4	6
8	10	12	14
16	18	20	22
24	26	28	30

Şekil 2.5. 3x4 Matris



Şekil 2.6. 4x4 Matrise Değer Yükleme

Şekil 2.6. da Command Window ekranında 4x4 boyutlarında bir matris tanımlanıp a değişkenine atanmıştır. Matrisde alt satıra geçmek için ';' operatörü kullanılmıştır.

2.1 Vektör ve Matrislere Değer Yükleme

Yukarı da görüldüğü gibi vektör yada matrislere değer yüklerken köşeli parantez kullanılır "**[.....]**" sütunlar ayırmak için boşluk bırakılır (istenirse boşluk bırakmadan virgül "**,**" işaretiyle de sütünlar ayrılabilir) satırları ayırmak için ise noktalı virgül "**;**" işareti kullanılır. Matrislerde her satırda aynı sayıda eleman olmak zorundadır.

Vektör ve matrisleri oluştururken en büyük kolaylıklardan birini iki nokta üst üste"**:**" yani kolon operatörü sağlar. Kolon operatörü **başlangıç_değeri:bitiş_değeri** veya **başlangıç_değeri:artış_miktarı:bitiş_değeri** biçiminde kullanılır.

Şekil 2.7. de '**:**' (kolon) operatörünün kullanılması görülmektedir.

Şekil 2.7. : (kolon) Operatörü

Şekil 2.7. de a vektörünü oluşturmak için kolon operatörü 1:10 şeklinde kullanılmış. Bu komut ile 1'den 10'a kadar olan 1 er artımlı bir a vektörü oluşturulmuştur. B vektörü de a vektörüyle aynı yöntem kullanılarak oluşturulmuştur. C vektörründe kolon operatörü 5:2:15 şeklinde kullanılmış. Bu komut ile 5'den 15'e kadar 2 şer artan bir vektör oluşturulmuştur. Artış miktarı pozitif ya da negatif olabilir.

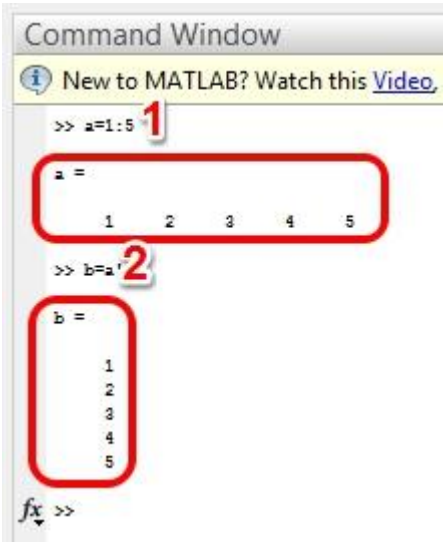
Kolon operatörüne benzer olarak **linspace** fonksiyonunu da kullanılabilir. Linspace fonksiyonu **linspace(başlangıç_değeri, bitiş_değeri, vektörün_eleman_sayısı)** şeklinde kullanılır. Artış miktarı başlangıç ve bitiş arasında eleman sayısını tutturacak şekilde fonksiyon tarafından otomatik olarak ayarlanır.

Şekil 2.8. de Linspace fonksiyonun kullanılışı görülmektedir.

Şekil 2.8. Linspace Fonksiyonunun Kullanımı

Şekil 2.8 de a vektörünü oluşturmak için **linspace(4, 20, 3)** komutu uygulanmıştır. Bu sayede a vektörü 4'den başlayıp 20'de biten 3 elemanlı bir vektör şeklinde tanımlanmıştır. Artış miktarı bu kurala uygun olarak otamatik olarak 8 hesaplanmıştır. Daha sonra a

vektörünü oluşturmak için **linspace(4, 20, 3)** komutu uygulanmıştır. Bu sayede b vektörü 4'den başlayıp 20'de biten 5 elemanlı bir vektör şeklinde tanımlanmıştır. Artış miktarı bu kurala uygun olarak otamatik olarak 4 hesaplanmıştır. En sonunda c vektörünü oluşturmak için **linspace(4, 20, 8)** komutu uygulanmıştır. Bu sayede c vektörü 4'den başlayıp 20'de biten 8 elemanlı bir vektör şeklinde tanımlanmıştır. Artış miktarı bu kurala uygun olarak otamatik olarak 2.2857 hesaplanmıştır.



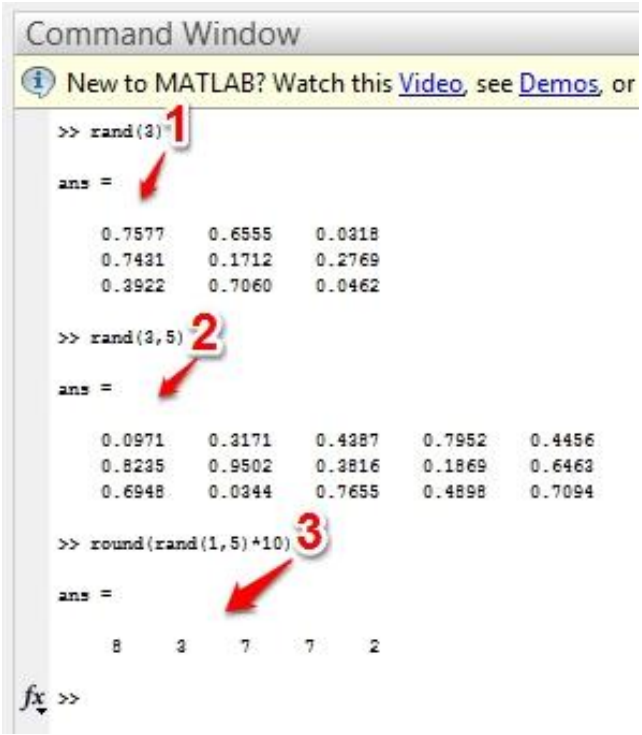
Şekil 2.7. ve Şekil 2.8. de kolon operatörü ve linspace fonksiyonuyla oluşturduğumuz vektörleri incelediğimizde hepsinin satır vektörü olduğunu görürüz, sizce bu komutlarla sütun vektörü yaratabilir miyiz?

Direkt olarak yaratamazsak bile dolaylı yoldan bu işlemi gerçekleştirebiliriz. Şekil 2.9. da görüldüğü gibi öncelikle 1. adımda satır vektörü olan a vektörü yaratılıp daha sonra 2. adımda transpozesini alınarak satır vektörü olarak yaratılan vektör sütun vektörü olan b ye dönüştürülmüştür. MATLAB programında bir vektörün ya da matrisin transpozesini "'" operatörü ile alabiliriz.

Şekil 2.9. Sütun Vektörü

: operatörü ve linspace fonksiyonu dışında matris yaratmak için kullanabilecek bir diğer fonksiyon da rand fonksiyonudur. Rand fonksiyonu **rand**, **rand(n)** yada **rand(r,c)** biçiminde kullanılabilir. Rand fonksiyonu ile elemanları 0-1 arasında rastgele sayılardan oluşan istenilen boyutta bir matris yaratılır.

Şekil 2.10. da Rand fonksiyonun kullanılışı görülmektedir.

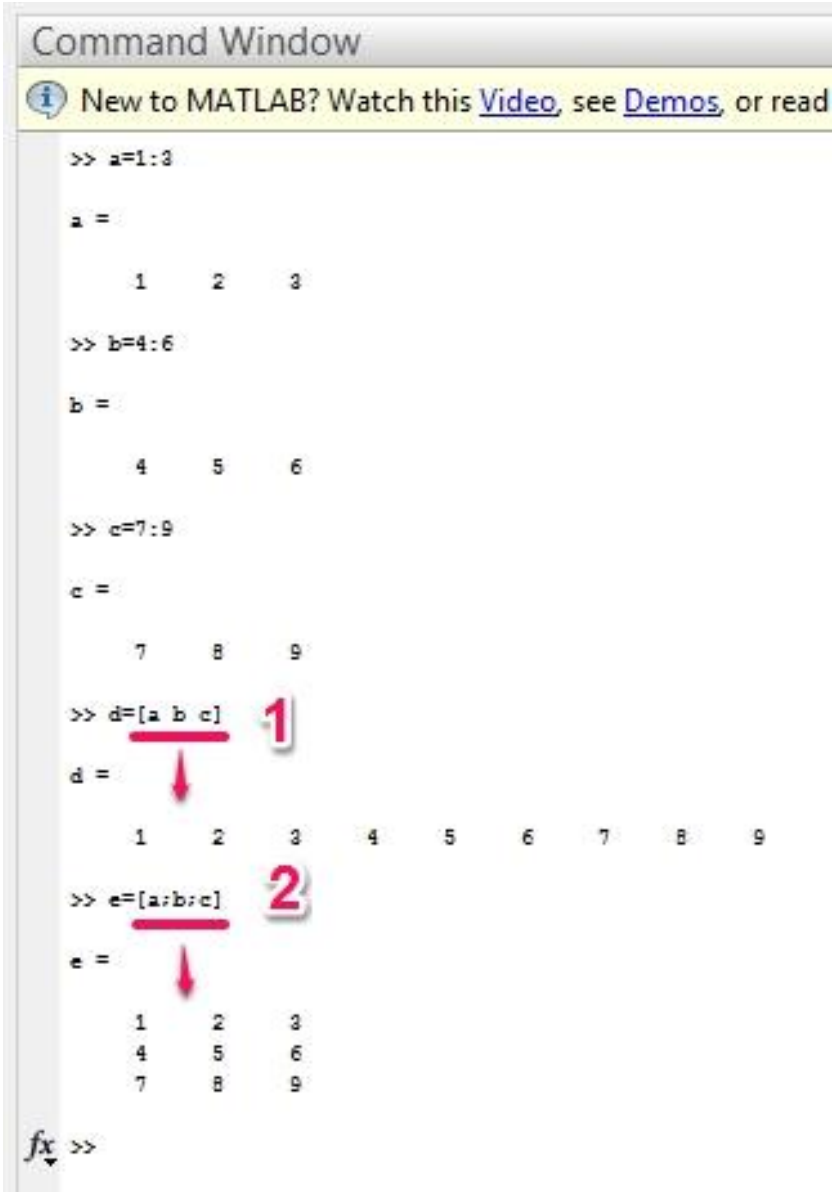


Şekil 2.10. Rand Fonksiyonun Kullanımı

Şekil 2.10. da fonksiyonu ilk olarak rand(3) şeklinde kullanarak 3 x 3 boyutunda bir kare matris, daha sonra rand(3,5) şeklinde kullanarak 3 x 5 boyutunda bir matris elde edilmiştir. Son olarak rand(1,5)*10 şeklinde kullanıp rand fonksiyonu ile elemanları sadece 0-1 arasında yaratılan matrislerden farklı olarak elemanlarının daha büyük sayılara ulaşması sağlanmıştır. Round fonksiyonu ile oluşan büyük sayılar yuvarlanarak matrisin bütün elemanların tam sayı olması sağlanmıştır. Her 3 işlemdede fonksiyon bir değişkene atanmadığı için oluşturulan matrisler ans değişkenine atanmıştır. Hatırlanacağı üzere MATLAB programında işlem sonucu bir değişkene atanmaz ise ans değişkeninde tutulur. Yukarıda ki örneklerden farklı bir şekilde fonksiyon sadece rand şeklinde kullanıldığında 0 –1 arasında skaker bir sayı üretir.

Bir vektörü çoğaltarak ya da vektörleri birleştirerek de, yeni vektör ya da matrisler oluşturulabilir.

Şekil 2.11. de vektör birleştirme işlemi görülmektedir.

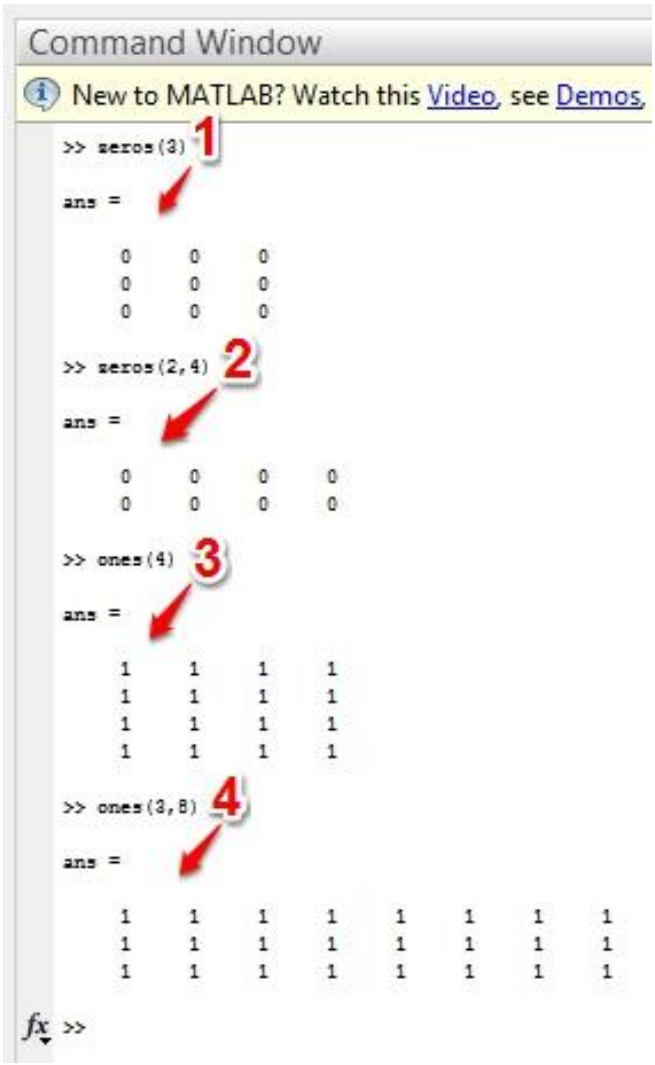


Şekil 2.11. Birleştirme Yöntemi

Şekil 2.11. de öncelikle a, b, c isminde 1x3 boyutunda 3 adet vektör tanımlanmıştır. Daha sonra bu vektörler birleştirilerek 1x9 boyutunda d vektörü ve 3x3 boyutunda e matrisi oluşturulmuştur. Eğer vektör ya da vektörlerden matris oluşturacaksak sütün sayılarının aynı olmasına dikkat etmeliyiz.

Yukardakilerin dışında özel matrisler yaratmak için kullanılan başka fonksiyonlarda mevcuttur. Bunlardan ilki zeros fonksiyonudur. Zeros fonksiyonu **zeros(n)** yada **zeros(r,c)** şeklinde kullanılabilir. Zeros fonksiyonu ile bütün elemanları 0 dan oluşan matrisler elde edebiliriz. Bir diğer özel matris yaratma fonksiyonu ones fonksiyonudur. Ones fonksiyonu **ones(n)** yada **ones(r,c)** şeklinde kullanılabilir. Ones fonksiyonu ile bütün elemanı 1 den oluşan matrisler elde edebiliriz.

Şekil 2.12. de özel matris yaratma fonksiyonlarından zeros ve ones görülmektedir.

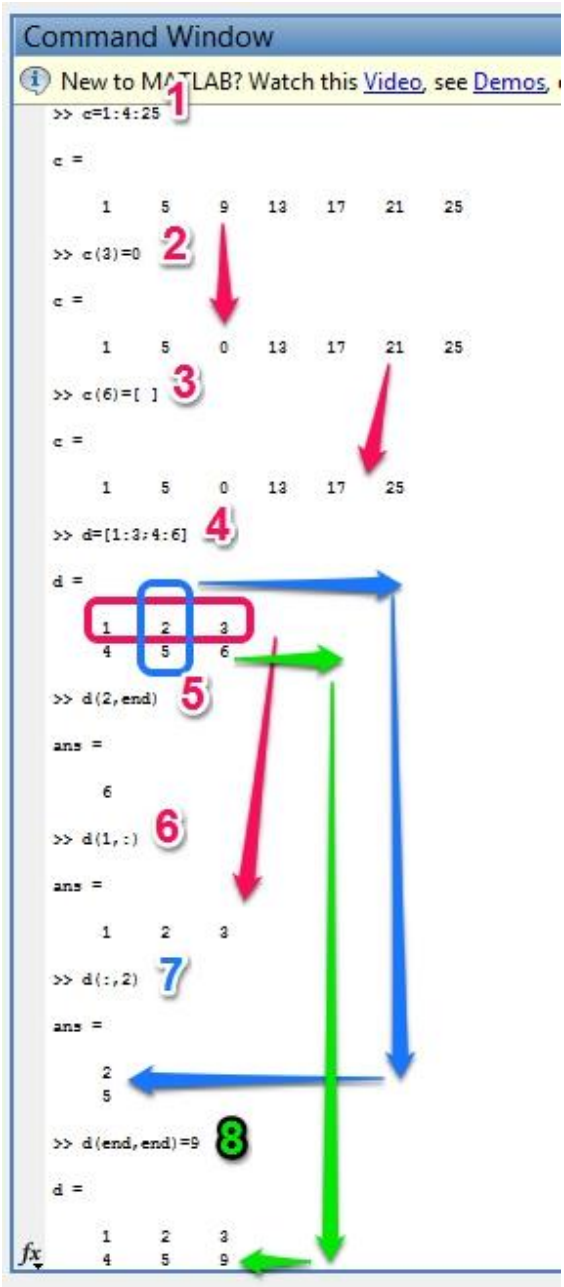


Şekil 2.12. Zeros ve Ones Fonksiyonlarının Kullanımı

Şekil 2.12. de birinci adımda zeros(3) komutu bütün elemanları 0 dan oluşan 3 x 3 boyutunda bit matris yaratılmıştır. İkinci adımda zeros(2,4) komutu bütün elemanları 0 dan oluşan 2 x 4 boyutunda bit matris yaratılmıştır. Üçüncü adımda ones(4) komutu bütün elemanları 1 den oluşan 4 x 4 boyutunda bit matris yaratılmıştır. Dördüncü ve son adımda ise ones(3,8) komutu bütün elemanları 1 den oluşan 3 x 8 boyutunda bit matris yaratılmıştır.

Bir diğer özel matris ise köşegen üzerindeki elemanları 1 geri kalan elemanlarının 0 olduğu kare matrislerdir. Bu matrislere birim matris denir. Birim matrisler **eye(n)** fonksiyonu ile oluşturulur. Diğerlerinden farklı olarak birim matris sadece n x n boyutunda oluşturulabilir. Şekil 2.13. de eye(3) komutu ile 3 x 3 boyutunda birim matris oluşturulmuştur.

Şekil 2.13 Eye Fonksiyonun Kullanımı



2.2 Vektör ve Matrislerde İndis

MATLAB programında vektör ya da matris değişkenlerine atanan değerler indisli olarak kaydedilir. Bu sayede vektörün ya da matrisin herhangi bir elemanını görebilir veya değiştirebiliriz. Vektör ve matrislerde ilk elemanın indisi 1 den başlamaktadır.

Şekil 2. 14. de indis işlemleri görülmek tedir.

1. adımda c vektörü oluşturulmuştur, **2. adımda** c vektörünün 3. elemanına 0 değeri atanmıştır **3. adımda** c vektörünün 6. Elemanı [ ] değerine eşitlenerek silinmiştir (Silme işlemi sadece vektörlerde yapılabilir matrislerde gerçekleşmez). **4. adımda** 2 x 3 boyutunda olan d matrisi tanımlanmıştır. Matris indisi belirtirken tıpkı matris boyutunu belirtirken yapıldığı gibi öncelikle satır sonra sütun indisini belirtilir.

Matrislerde indis kullanarak bir eleman veya bir satır yada sütun görelebilir, değiştirilebilir. Yandaki şekilde görüleceği gibi **6. adımda** 1. satır, **7. adımda** ise 2. sütun elemanları görülmüştür. Ancak değiştirme yapmak istendiğin de yazılan eleman sayısı ile satır ya da sütünda ki eleman sayısının aynı olması gerekmektedir. İndis numarası yazmadan "**end**" operatörü kullanılabilir. End operatörü satırın ya da sütunun son elemanın indisini belirtir. **5. ve 8. adımlarda** end oparatörü kullanılmıştır.

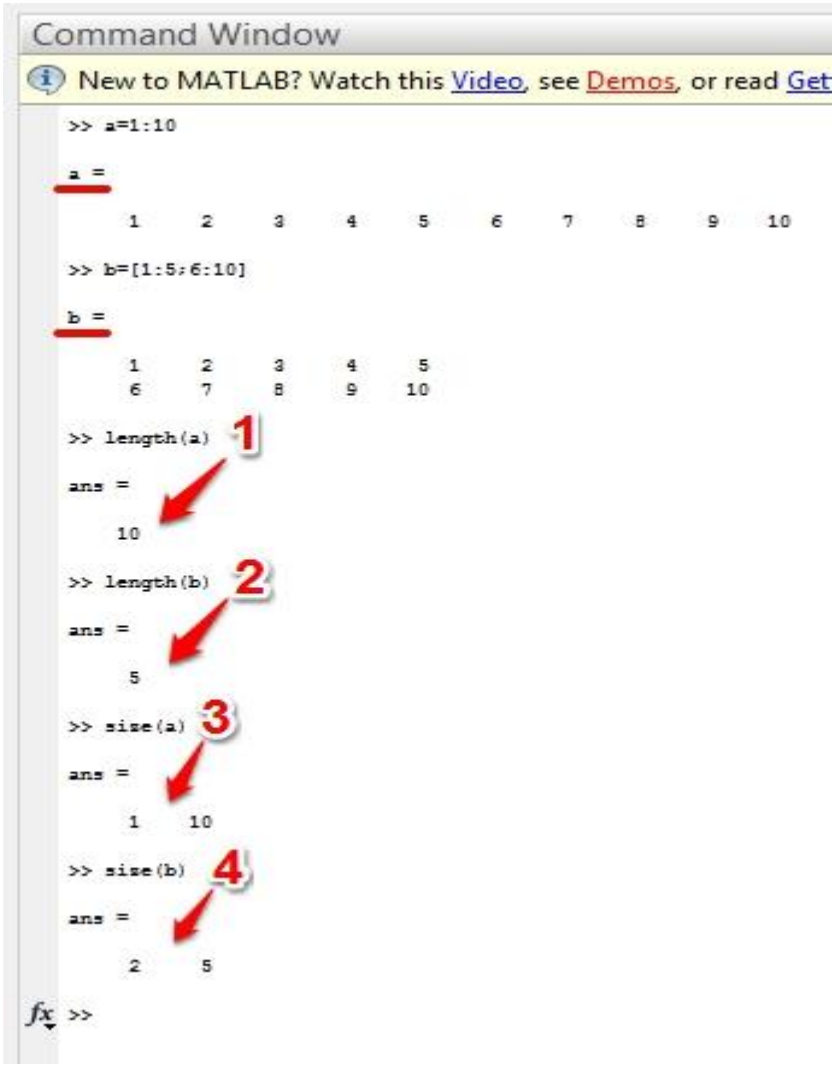
Şekil 2.14. İndis İşlemleri

Eğer belirttiğimiz indis matris ya da vektörün en büyük indisinden daha büyük değerde olursa "Index exceeds matrix dimensions" hata mesajını görürüz.

2.3 Vektör ve Matrislerin Boyutları

Vektör ve matrislerin boyutları **length** ve **size** komutlarıyla öğrenilir. Length komutu ile bir vektörün eleman sayısını verir. Komut matrislerde uygulandığında satır ya da sütunlarında bulunan elemanlardan sayısı büyük olanı verir. Size komutu ile vektör ya da matrislerin satır ve sütun sayılarını elde edilir.

Şekil 2.15. de length ve size fonksiyonlarının kullanımı görülmektedir.



Şekil 2.15. Length ve Size komutları

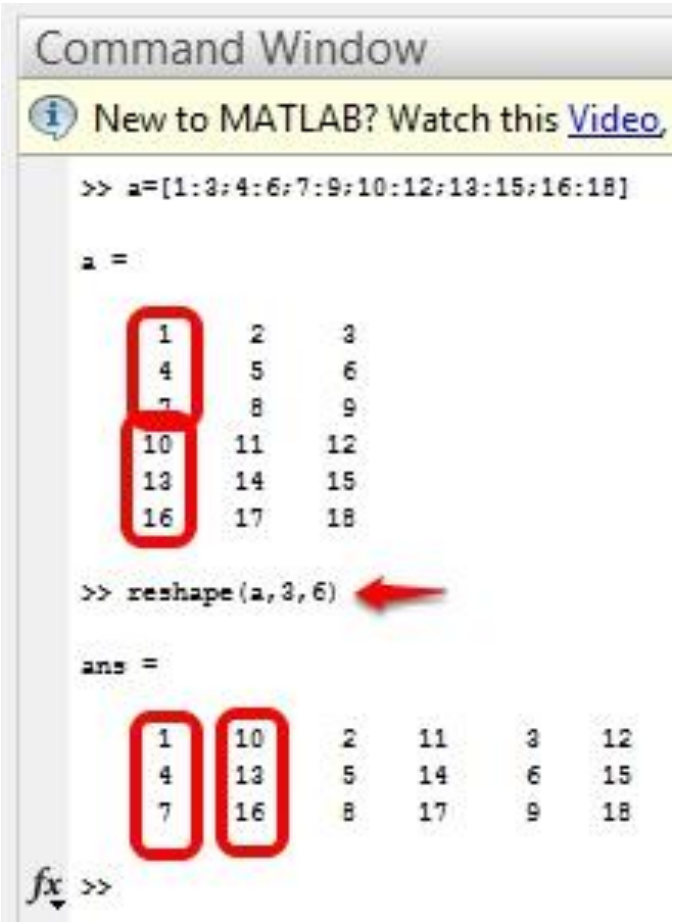
Şekil 2.15. de öncelikli 1 x 10 boyutunda a vektörü daha sonra ise 2 x 5 boyutlarında b matrisi oluşturulmuştur. **1. adımda** length(a) komutuyla a vektörünün eleman sayısı hesaplanmıştır. **2. adımda** length komutu b matrisine uygulandığında sütünlarında bulunan eleman sayısı büyük olduğundan 5 sonucu elde edilmiştir. **3. adımda** size(a) komutuyla a vektörünün satır ve sütün sayısı, **4. adımda** size(b) komutuyla b matrisinin satır ve sütün sayısı hesaplanmıştır.

Ayrıca bir vektör ya da matrisin toplam eleman sayısını **numel** komutuyla öğrenebiliriz.

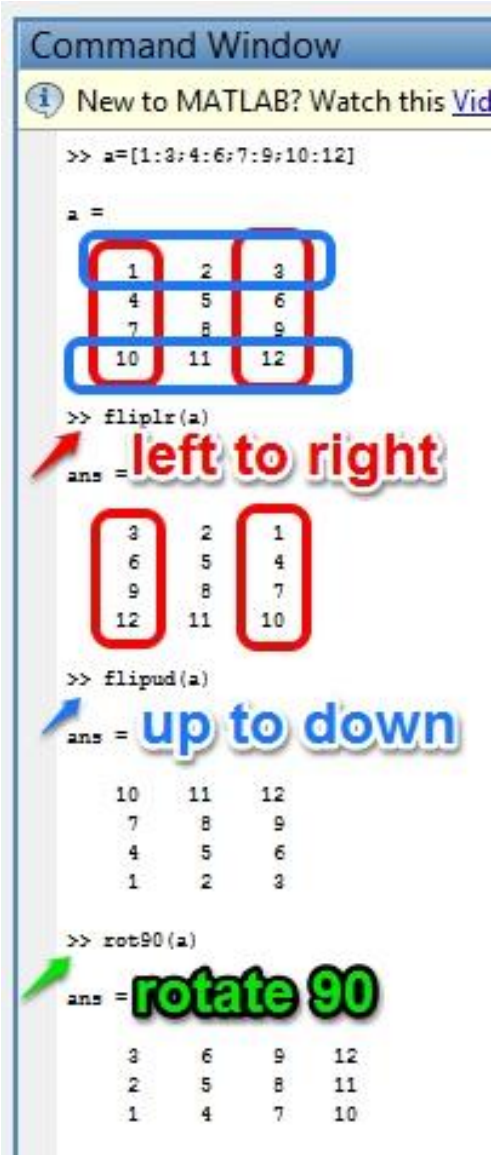
Mevcut bir matris yeniden boyutlandırılabilir. Bir matrisin yeniden boyutlandırılması için kullanılabilen fonksiyonlardan biri **reshape(değişken_adı,r,c)** fonksiyonudur. Burada dikkat edilmesi gereken en önemli husus, matrisi yeniden boyutlandırılırken toplam eleman sayısının çarpanlarına uygun bir şekilde hareket edilmesidir, örneğin 6 x 3 boyutlarında bir matrisin 18 elemanı olduğuna göre boyutlandırmadan sonra ancak ve ancak 1 x 18, 18 x 1, 2 x 9 ya da 9 x 2 boyutlarında bir matris oluşturulabilir. Boyutlandırmadan sonra elde edilen matris eski matrisin elemanlarının ilk sütundan başlayarak sırasıyla yerleştirilmesi sonucunda oluşturulur.

Matrisleri yeniden boyutlandırma için reshape fonksiyonundan başka sütunları soldan sağa doğru ters çeviren **fliplr(değişken_adı)** fonksiyonu satırları yukardan aşağıya ters çeviren **flipud(değişken_adı)** fonksiyonu ve matrisi saat yönünün tersinde 90 derece döndüren **rot90(değişken_adı)** fonksiyonları da kullanılabilir.

Şekil 2.16. da reshape, 2.17. de fliplr, flipud ve rot90 fonksiyonlarının kullanımı görülmektedir..



Şekil 2.16. Reshape Fonksiyonu



Şekil 2.17. Fliplr, Flipud ve Rot90 Fonksiyonları

Şekil 2.16. da öncelikle 6 x 3 boyutunda a matrisi oluşturulmuştur. Daha sonra reshape(a,3,6) komutuyla 3 x 6 boyutunda olacak şekilde yeniden boyutlandırılmıştır. Şekil 2.17. de öncelikle 4 x 3 boyutunda a matrisi oluşturulmuştur. Daha sonra 1. adımda fliplr(a) komutuyla a matrisinin sütunları soldan sağa ters çevrilmiştir. 2. adımda flipud(a) komutuyla a matrisinin satırları yukarıdan aşağıya ters çevrilmiştir. Son olarak 3. adımda ise rot90(a) komutuyla a matrisi saat yönünde 90 derece döndürülerek yeniden düzenlenmiştir.

Herhangi bir matrisi çoğaltarak daha büyük boyutlarda bir matris elde etmek istersek **repmat(değişken_adı,satır adeti,sütun adeti)** fonksiyonunu kullanabiliriz.

Bir vektörün elemanlarından biri ya da bir kaçını silerek de yeniden boyutlandırabiliriz. Bunun için silmek istediğimiz eleman ya da elemanların bulunduğu indis ya da indisleri [ ] değerine eşitlememiz yeterlidir. Bu işlem matrislerde gerçekleşmez.

2.4 Vektör ve Matrislerde Aritmetik İşlemler

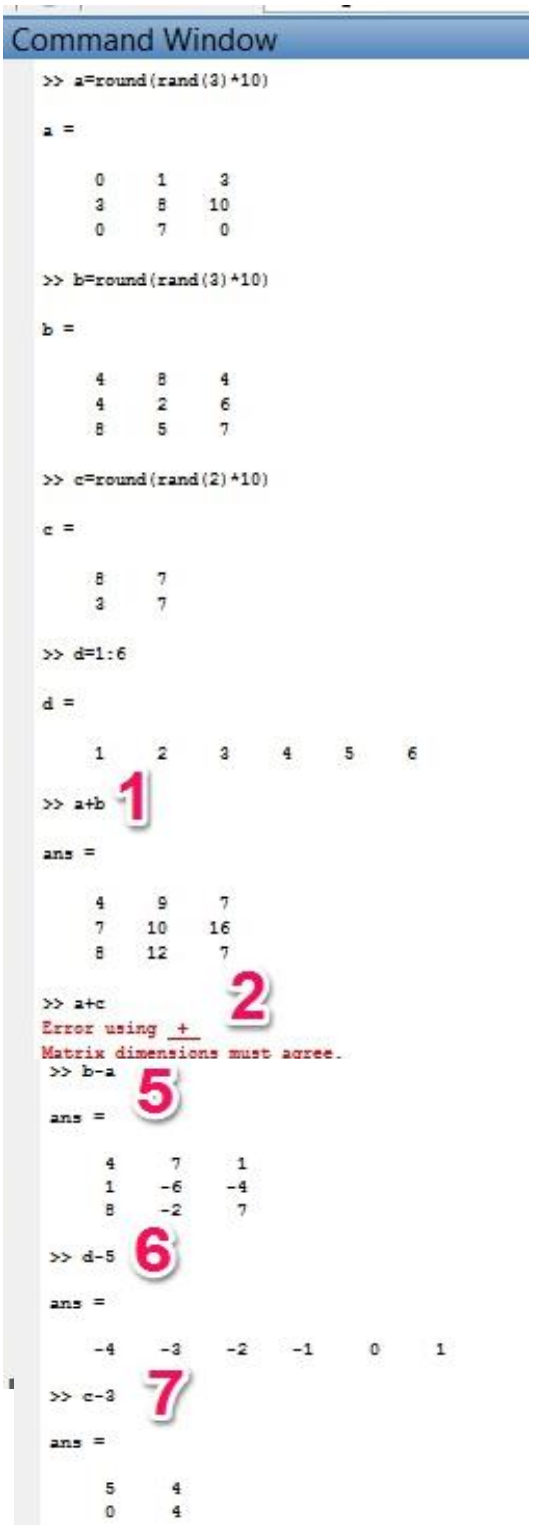
MATLAB' da iki farklı aritmetik işlem türü tanımlıdır. Bunlardan birincisi dizi işlemleri ya da diğer adıyla birebir işlemlerdir, ikincisi ise matris işlemleri ya da diğer adıyla cebirsel işlemlerdir.

Matris işlemleri lineer cebir kurallarına göre çalışır. Dizi işlemleri ise birebir elemanlar üzerinden hesaplama olanağı sağlar. Matris işlemleri ve dizi işlemleri arasındaki tek fark işlem işaretinden önce "." operatörünün bulunmasıdır. Birebir işlemlerde toplama ve çıkarma hariç işlem işaretinden önce "." operatörü yazılır. Toplama ve çıkarma işlemlerinde her iki aritmetik işlem türünde de aynı olduğundan böyle bir ayrıma gerek yoktur. Tablo 2.1. de her iki işlem türü için geçerli olan işlem operatörleri görülmektedir.

Matris transpoze ile birebir transpoze arasındaki fark sadece kompleks matrislerde görülebilir.

Tablo 2.1. Vektör ve Matrislerde Aritmetik İşlemler

Aritmetik İşlemler			
Dizi (Birebir) İşlemler		Matris (Cebirsel) İşlemler	
+	Toplama	+	Toplama
-	Çıkarma	-	Çıkarma
.*	Birebir Çarpma	*	Matris Çarpma
.^	Birebir Üs Alma	^	Matris Üs Alma
.\	Birebir Sağdan Bölme	\	Matris Sağdan Bölme
./	Birebir Soldan Bölme	/	Matris Soldan Bölme
.'	Birebir Transpoze	'	Matris Transpoze



2.4.1 Vektör ve Matrislerde Ortak Aritmetiksel İşlemler(Toplama – Çıkarma)

Yukarıda anlatıldığı gibi Toplama ve Çıkarma işlemleri her iki aritmetiksel hesaplamada da aynı yöntemle hesaplanır. Toplama veya çıkartma işlemlerinde sonucu karşılıklı olarak her elemanın birbiriyle toplanması yoluyla bulunur. Bir matris yada vektör yalnızca kendisiyle aynı sütun ve satır sayısına sahip yani aynı boyutlarda bir matris ve vektörle toplanabilir yada çıkartılabilir. Bu kurala uyulmaması durumunda "Matrix dimensions must agree" hatasını alırız. Bir matris ya da vektör skaler bir büyüklükle toplandığında veya çıkartıldığında sonuç skaler büyüklüğün matris ya da vektörün her elemanıyla ayrı ayrı toplanması veya çıkartılmasıyla hesaplanır.

Şekil 2. 14. de toplama ve çıkarma işlemleri görülmektedir.

Şekil 2.14. de öncelikle 3 x 3 boyutunda a matrisi, 3 x 3 boyutunda b matrisi, 2 x 2 boyutunda c matrisi ve 1 x 6 boyutunda d vektörü oluşturulmuştur **1. adımda** a ve b matrislerinin her elemanı birbiriyle ayrı ayrı toplanarak hesaplanmıştır. **2. adımda** 3 x 3 boyutlarında a matrisiyle 2 x 2 boyutlarında b matrisi toplanmak istendiğinde hata mesajı görülmüştür. **3. ve 4. adımlarda** d vektörünün ve c matrisinin skaler bir büyüklük olan 2 ile toplanmasının sonuçları hesaplanmıştır. **5. adımda** iki matrisin birbirinden çıkartılmıştır. **6. adımda** ise d vektöründen skaler bir büyüklük olan 5 in çıkartılmasının sonucu, 7. adımda ise c matrisinden skaler bir büyüklük olan 3 ün çıkartılmasının sonucu hesaplanmıştır.

Şekil 2.18. Vektör ve Matrislerde Toplama ve Çıkartma

2.4.2 Vektör ve Matrislerde Matris(Cebirsel) İşlemler

Tablo 2.2. Matris İşlemler

Matris(Cebirsel) İşlemler		
*	Matris Çarpma	C=A*B, A ve B matrislerin cebirsel çarpımı A matrisinin sütün sayısı ile B matrisinin satır sayısı eşit olmak zorundadır.
^	Matris Üs Alma	C=A^B A matrisinin üssünü alır. B skaler olmalıdır.
\	Matris Sağdan Bölme	C=A\B, C=B/A, A ve B matrislerinin sütun sayıları eşit olmak zorundadır.
/	Matris Soldan Bölme	C=A/B A ve B matrislerinin cebirsel bölümü. A ve B matrislerinin sütun sayıları eşit olmak zorunda
'	Matris Transpoze	C=A' A matrisinin cebirsel transpozu

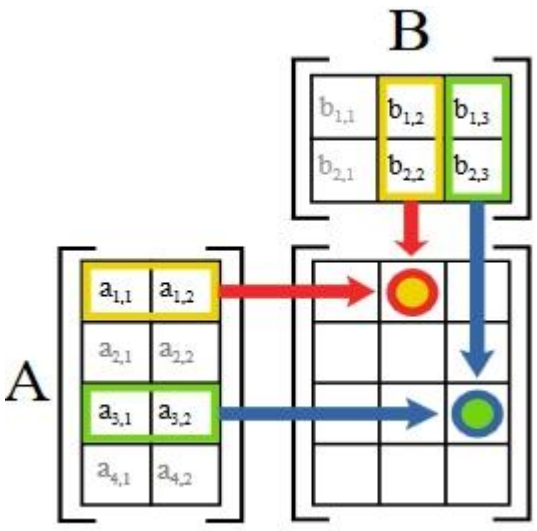
Matris işlemleri lineer cebir kurallarını sağlamak durumundadır, ve çok boyutlu diziler için uygun değildir. Örneğin iki matrisin çarpımı için birinci matrisin sütun sayısı ile ikinci matrisin satır sayısının eşit olması gerekir aksi halde işlem gerçekleşmez.

İki matrisin matris çarpımı hesaplandığında elde ettiğimiz matrisin satır sayısı birinci matrisin satır sayısı kadar, sütun sayısı ise ikinci matrisin sütun sayısı kadar olacaktır.

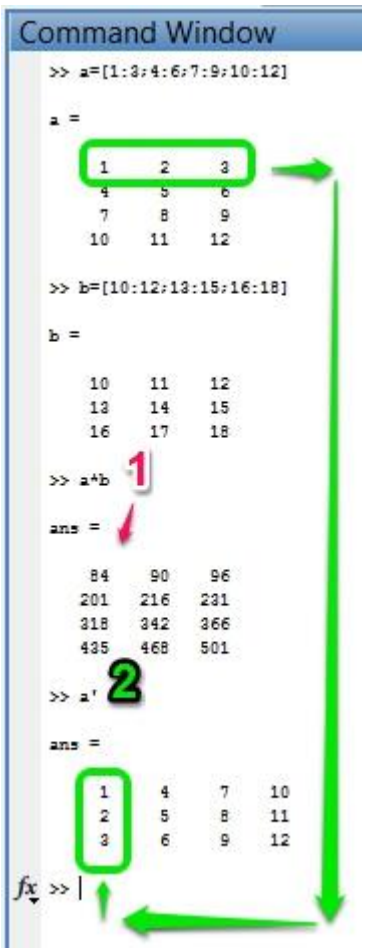
$$\begin{bmatrix} 1 & 0 & 2 \\ -1 & 3 & 1 \end{bmatrix} \begin{bmatrix} 3 & 1 \\ 2 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} (1\times 3+0\times 2+2\times 1) & (1\times 1+0\times 1+2\times 0) \\ (-1\times 3+3\times 2+1\times 1) & (-1\times 1+3\times 1+1\times 0) \end{bmatrix} = \begin{bmatrix} 5 & 1 \\ 4 & 2 \end{bmatrix}$$

Şekil 2.19. Matrislerde Matris Çarpım İşlemi

Şekil 2. 19. da 2 x 3 boyutunda olan bir matrisle 3 x 2 boyutunda olan bir matris çarpılarak sonuç matrisinin elemanları hesaplanmıştır. Şekil 2.20. de çarpma işleminin diagramı görülmektedir.



Şekil 2.20. Matrislerde Matris Çarpım İşleminin Diagramı



Şekil 2.21. de matrislerin cebirsel çarpımı ve cebirsel transpozunun hesaplanması görülmektedir.

Şekil 2.21.' de öncelikle 4 x 3 boyutunda a matrisi ve 3 x 3 boyutunda b matrisi oluşturulmuştur. **1. adımda** sütün ve satır sayısı uyumlu a ve b matrisleri çarpılıp sonuç hesaplanmıştır. **2. adımda** 4 x 3 boyutunda ki a matrisinin tranpozunun alınmasıyla 3 x 4 lük sonuç matrisi oluşturulmuştur.

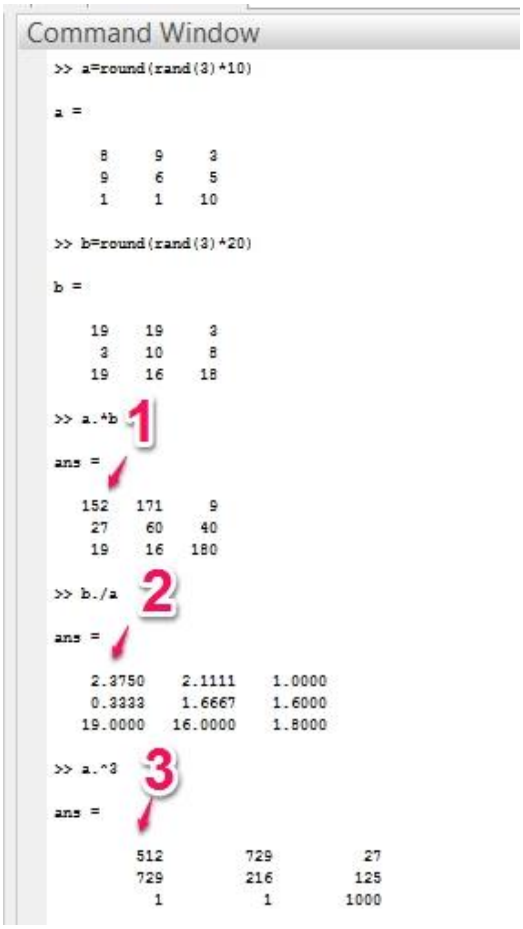
Şekil 2.21. Matris İşlemleri

2.4.3 Vektör ve Matrislerde Dizi(Birebir) İşlemler

Dizi işlemlerinin uygulanacağı bütün vektör ve matrisler satır ve sütun olarak eşit boyutta olmak zorundadır. Aksi halde hata mesajıyla karşılaşırız.

Şekil 2.22. de matrislerin birebir çarpımı, bölümü ve üs alma işlemleri görülmektedir

Şekil 2.22. de öncelikle 3 x 3 boyutlarında olan a ve b matrisleri oluşturulmuştur. **1. adımda** a ve b matrisleri biriber çarpılmıştır. Bu çarpmada bütün elemanlar kendisiyle aynı indisli elemanlarla çarpılarak sonuç matrisini oluşturulmuştur. **2. adımda** b matrisi a matrisine birebir bölünmüştür. Bu bölmede bütün elemanların kendisiyle aynı indisli elemana bölünerek sonuç matrisini oluşturmuştur. **3. adımda** ise a matrisinin birebir küpü alınmıştır. Bu işlemde de bütün elemanların tek tek küpü alınarak sonuç matrisi oluşturulmuştur.



Şekil 2.22. Vektör ve Matrislerde Birebir İşlemler

2.5. Diğer Matris İşlemleri

Yukarıda bahsedilen hususlar dışında matrislerde 2 önemli husus daha vardır. Bunlar;

1) Determinant
2) İnverse

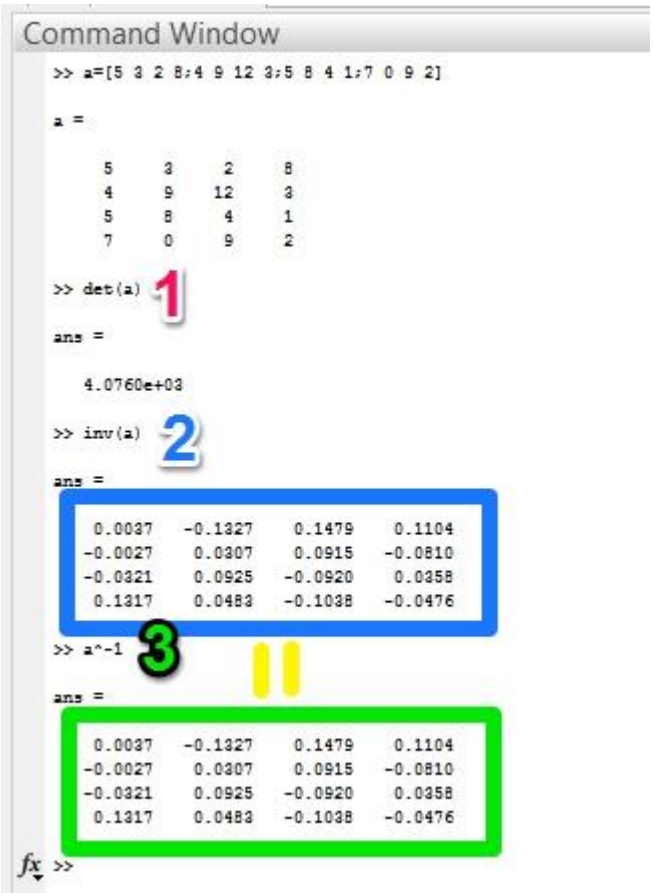
2.5.1 Determinat

Determinant kare matrislerle ilişkili özel bir sayıdır. MATLAB programında bir matrisin determinantını hesaplamak için **det(x)** fonksiyonunu kullanılır.

2.5.2 Inverse

Inverse bir matrisin tersinin hesaplanma işlemidir. MATLAB programında inverse almak için ya **inv(x)** fonksiyonu ya da **x^-1** olacak şekilde üs alma işlemi yapılır. Burada dikkat edilecek husus üs alma işleminin birebir değil cebirsel üs şeklinde olması gerektiğidir.

Şekil 2.5.1. de det ve inv fonksiyonlarının kullanımı görülmektedir.



Şekil 2.23. Matrislerde Determinant ve Inverse İşlemleri

Şekil 2.1. de önce 4 x 4 boyutunda olan a matrisi oluşturulmuştur. **1. adımda** det(a) komutu ile a matrisnin determinantı hesaplanmıştır. **2. adımda** inv(a), **3. adımda** a^-1 komutuyla a matrisinin inversi yani tersi oluşturulmuştur.

2.6. Ölçme ve Değerlendirme

Aşağıdaki soruları çözünüz.

1) Elemanları 0-100 arası rastgele tam sayılardan oluşan 3 x 6 boyutunda a matrisini yaratıp transpozunu alarak b matrisini oluşturun.
2) A matrisinin ilk üç sütunundan c matrisi son üç sütunundan d matrisi oluşturup bunları birbiriyle birebir çarparak e matrisini oluşturun. E matrisinin determinantını hesaplayın.
3) B matrisinin 3. ve 5. satırlarından 1 x 6 boyutunda f vektörünü oluşturup birebir karesini alaral g matrisini oluşturun.
4) A matrisini 2 x 9 boyutunda yeniden boyutlandırarak h matrisini oluşturun. Bu ana kadar yapılan işlemler sonucu hafizada tutulan bütün değişkenleri işlemler sonucu oluşan bütün değişkenleri bolum2.mat dosyasına kaydedin

Not: Soruların cevaplarını Ek2 de bulabilirsiniz.

3. FONKSİYONLAR

MATLAB'ın çok güçlü ve çok kapsamlı bir fonksiyon yapısı vardır. Bu yapıyı temelde 3 grupta sınıflandırmak mümkündür. Bunlar;

1) Yerleşik(Built-In) fonksiyonlar
2) M-dosyalarından oluşan MATLAB toolboxlarında tanımlı fonksiyonlar.
3) Özel uygulamalar için kullanıcıların tarafından oluşturulan M-dosyaları.

MATLAB'da mevcut fonksiyonlar genel olarak aşağıdaki kategorilere ayrılır.

- Temel matematiksel fonksiyonlar
- Özel fonksiyonlar
- Temel matrisler ve matris işlemleri
- Veri analizleri
- Polinomlar
- Diferensiyel denklem çözümleri.
- Lineer olmayan denklemler ve optimizasyon
- Sayısal integral hesaplamaları.
- Sinyal işleme

Fonksiyonlar hususunda unutmaması gereken en önemli husus MATLAB' ın geniş ve kapsamlı yapısından dolayı hepsinin tek bir dökümanda işlenmesinin neredeyse imkansızlığıdır. Burada tanıtılanlar dışında da birçok MATLAB fonksiyonu mevcuttur. Fonksiyonların tam listesine http://www.mathworks.com/help/matlab/functionlist.html adresinden ulaşılabilir.

3.1. Yerleşik(Built-In) Fonksiyonlar

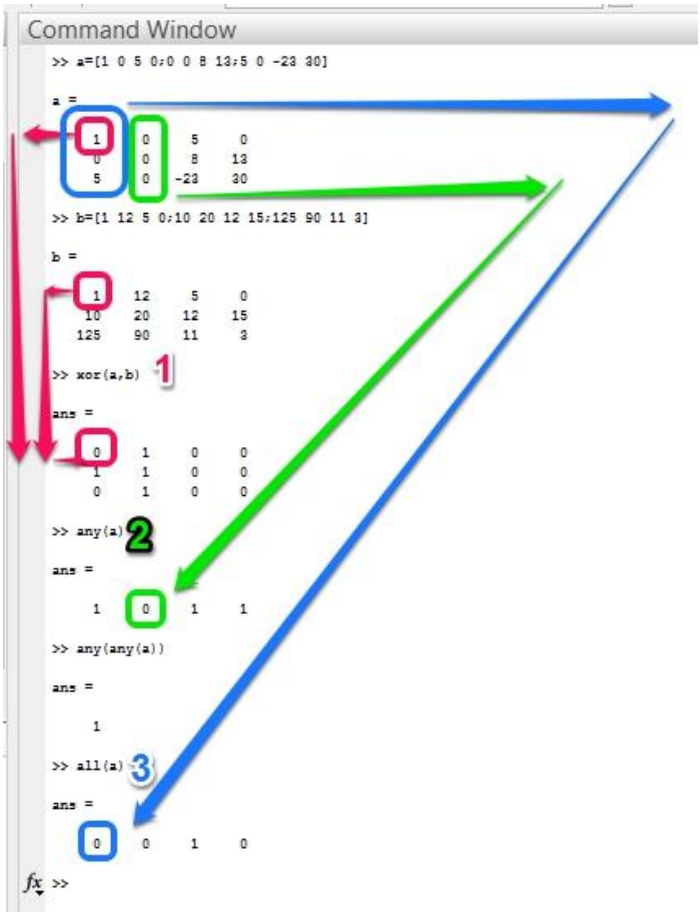
MATLAB' da birçok yerleşik fonksiyon vardır. **help** komutunu kullanarak gerek fonksiyonların nasıl kullanılacağı gerekse yerleşik fonksiyon grupları görülebilir. Command window ekranında **help** yazıp enter tuşuna basıldığında fonksiyon gruplarının listesini görülür. Komut satırında **helpwin** yazıldığında aynı listeyi komut satırında görüntülemek yerine yeni bir pencerede görüntülenmesi sağlanır. Bu liste oldukça uzun olmakla beraber grupları önem sırasına göre listelemiştir. Fonksiyon grupları birbiriyle ortak özellikleri olan fonksiyonların bir araya gelmesiyle oluşmuştur. En bilinen fonksiyon grupları arasında temel matematik fonksiyonlarının oluşturdurğu **elfun,** temel matris fonksiyonlarının oluşturduğu **elmat,** özel matematik fonksiyonlarının oluşturduğu **specfun** veya random fonksiyonlarının oluşturduğu **randfun** grupları söylenebilir.

Yukarda ki basit ya da karmaşık matematiksel fonksiyonlar dışında MATLAB programında tanımlı olan mantıksal fonksiyonlar da mevcuttur. Aşağıda ki tabloda MATLAB programında kullanılan mantıksal fonksiyonlar ve işlevleri görülmektedir.

Tablo 3.1. Mantıksal Fonksiyonlar

Mantıksal Fonksiyonlar	
xor(a,b)	**A ve b matrislerinin elemanları birebir xor işlemine tabi tutulur.** Bu işlemde yalnız biri 0 dan farklıysa sonuç 1, her ikisi de 0 veya her ikisi de 0 dan farklıysa sonuç 0 olur
any(x)	**X vektörünün elemanlarının içinde 0 dan farklı bir eleman varsa sonuç 1, aksi halde sonuç 0.** Any fonksiyonunu bir matrise tabi tutarsak matrisin her sütunu ayrı bir vektör olarak alıp ortaya hesaplanan skaler büyüklüklerle bir sonuç vektörü oluşturur.
all(x)	**X vektörünün bütün elemanları 0 dan farklıysa sonuç 1, aksi halde sonuç 0.** All fonksiyonunu bir matrise tabi tutarsak matrisin her sütunu ayrı bir vektör olarak alıp hesaplanan skaler büyüklüklerle bir sonuç vektörü oluşturur.

Şekil 3.1. de mantıksal fonksiyon uygulamaları görülmektedir.



Şekil 3.1. Mantıksal Fonksiyon Uygulamaları

Şekil 3.1. de öncelikle 3 x 4 boyutunda a matrisi ve yine 3 x 4 boyutunda b matrisi oluşturulmuştur. 1. Adımda iki matrise xor işlemi, 2 adımda iki matrise any işlemi, 3. Adımda iki matrise all işlemi uygulanmıştır.

Aşağıda temel matematik fonksiyonlarının oluşturduğu elfun grubunun bazı üyeleri hakkında açıklama verilecektir.

Tablo 3.2. Elfun Grubu Fonksiyonları

Elfun			
Grup	Fonksiyon Adı	Görevi	Şekil
Complex	**abs(x)**	Mutlak değer hesaplar ve karekterleri sayıya dönüştürür	1
	angle(x)	X karmaşık sayısının faz açısını radyan cinsinden hesaplar	2
	complex(x,y)	x + yi formunda karmaşık sayı oluşturur.	3
	conj(x)	X karmaşık sayısının eşleniği hesaplar	4
	imag(x)	X karmaşık sayısının sanal kısmını verir	5
	real(x)	X karmaşık sayısının gerçek kısmını verir	6
	Unwrap	Faz açısını düzenler	
	isreal(x)	X sayısı reel sayı ise "1" değilse "0" değerini verir	7
	Cplxpair	Karmaşık eşlenik çiftleri içine sayıları sıralama	
Rounding	**fix(x)**	Sıfıra doğru tam sayıya yuvarlar	8
	floor(x)	Sayının tam kısmını alır	9
	ceil(x)	En yakın büyük tam sayıya yuvarlar	10
	round(x)	En yakın tam sayıya yuvarlar	11
	mod(x,y)	Bölümünden kalanı hesaplar(işaretli olarak)	12
	rem(x)	Bölümünden kalanı hesaplar(işaretsiz olarak)	
	sign(x)	Verinin pozitif mi yoksa negatif mi olduğunu belirler.	13
Exponential	**exp(x)**	e^x üzeri değerini hesaplar	14
	log(x)	ln(x) değerini hesaplar.	15
	log10 (x)	$Log_{10}(X)$ değerini hesaplar.	16
	pow2(x)	2^x değerini hesaplar	17
	realpow(x,y)	x^y değerini hesaplar.	18
	reallog(x)	Real sayının e tabanında logunu alır.	
	sqrt(x)	Karekökünü hesaplar	19
Trigonemetric	**sin(x)**	X radyan değerinin sinüs değerini hesaplar	20
	sind(x)	X derece değerinin sinüs değerini hesaplar	21
	sinh(x)	X radyan değerinin hiperbolik sinüs değerini hesaplar	22
	asin(x)	X radyan değerinin ters sinüs değerini hesaplar	23
	asind(x)	X derece değerinin ters sinüs değerini hesaplar	
	asinh(x)	X radyan değ. ters hiperbolik sinüs değerini hesaplar	
	cos(x)	X radyan değerinin kosinüs değerini hesaplar	
	tan(x)	X radyan değerinin tanjant değerini hesaplar	
	cot(x)	X radyan değerinin cotanjant değerini hesaplar	
	sec(x)	X radyan değerinin sekand değerini hesaplar	
	csc(x)	X radyan değerinin kosekand değerini hesaplar	
	hypot(x,y)	X ve y sayılarının karelerini toplayıp karekökünü alır.	24

Şekil 3.2. de elfun grubu üyelerinden bazılarının uygulamalrı görülmektedir.

Şekil 3.2. Elfun Grubu Fonksiyonları Uygulamaları

Şekil 3.2. de tablo 3.2 de listelenen elfun grubu fonksiyonlarının bazılarının uygulaması yapılmıştır.

3.2. M-Dosyaları

M-dosyaları MATLAB ortamında kullanılan komutlardır. Bu dosyalar .m uzantıya sahiptirler. Her kullanıcı amacına göre M-File oluşturabileceği gibi program ile birlikte gelen veya diğer kullanıcıların hazırladığı ve TOOLBOX adı verilen dosya grubunda bulunan M-File'larını kullanabilir.

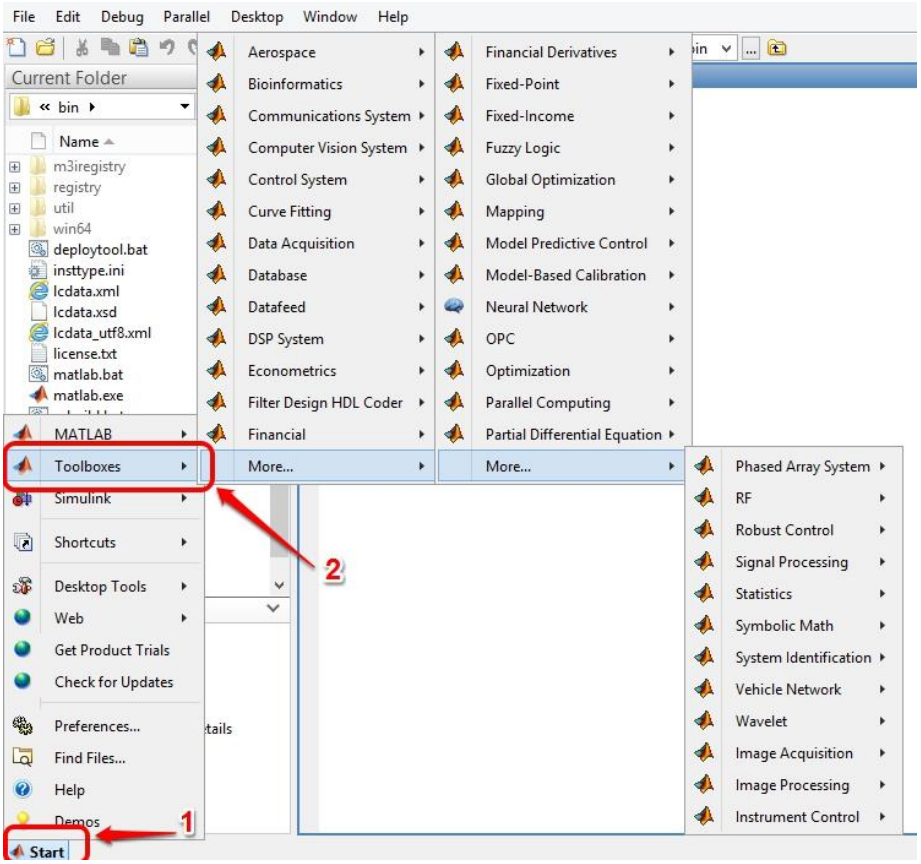
M-dosyaları 2 gruba ayrılır. Bunlar;

1) MATLAB sürümlerinde tanımlı ya da 3. parti geliştiricilerin oluşturduğu toolboxlarda bulunan M-dosyaları
2) Kullanıcı tarafından oluşturulan M-dosyaları.

3.2.1. ToolBox'larda Tanımlı M-Dosyaları

MATLAB'ın son sürümünde tanımlı olan toolbox'ların listesine http://www.mathworks.com/help/relnotes/ adresinden ulaşılabilir. MATLAB oturumunda aktif olarak kullanılan toolbox'ların listesi **license('inuse')** komutuyla görülebilir. Sistemde kurulu olan MATLAB sürümünde herhangi bir toolbox'ın kurulu olup olmadığı **license checkout <Toolbox_Name>** komutuyla öğrenilir. Sonuç olarak 1 değerini elde edersek belirtilen toolbox sistemde kurulu haldedir. Sistemde kurulu olan tüm toolbox'ların listesi **ver** komutuyla elde edilir. Tablo 3.3. de MATLAB 2012 a sürümünde tanımlı olan toolboxların tam listesini görülebilir. Ayrıca **Start** (Burada kastedilen MATLAB programına ait Start butonudur.) butonu tıklanıp **Toolboxes** seçeneği altında da bu liste görülebilir.

Şekil 3.3. de Start-Toolbox seçeneği altındaki toolbox'lar görünmektedir.



Şekil 3.3. Toolbox Listesi

MATLAB da tanımlı olmayan 3. parti geliştiriciler tarafından geliştirilen toolbox'ları kurmak ve kullanmak için **File-Set Path** seçeneğini tıklanarak gelen ekranda **Add Folder** veya **Add with Subfolders** butonunu tıklanıp toolbox'ın bulunduğu klasör işaretlenmelidir.

Tablo 3.3. Matlab 2012 a Versiyonunda Tanımlı Toolbox Listesi

ToolBox Name	ToolBox Name	ToolBox Name
MATLAB	Image Acquisition T.	SimEvents
Simulink	Image Processing T.	SimHydraulics
Aerospace Blockset	Instrument Control T.	SimMechanics
Aerospace Toolbox	MATLAB Builder EX	SimPowerSystems
Bioinformatics Toolbox	MATLAB Builder JA	SimRF
Communications System T.	MATLAB Builder NE	Simscape
Computer Vision System T.	MATLAB Coder	Simulink 3D Animation
Control System Toolbox	MATLAB Compiler	Simulink Code Inspector
Curve Fitting Toolbox	MATLAB Distributed Computing Server	Simulink Coder
DO Qualification Kit	MATLAB Report Generator	Simulink Control Design
DSP System Toolbox	Mapping Toolbox	Simulink Design Optimization
Data Acquisition Toolbox	Model Predictive Control T.	Simulink Design Verifier
Database Toolbox	Model-Based Calibration T.	Simulink Fixed Point
Datafeed Toolbox	Neural Network Toolbox	Simulink PLC Coder
Econometrics Toolbox	OPC Toolbox	Simulink Report Generator
Embedded Coder	Optimization Toolbox	Simulink Verification and Validation
Filter Design HDL Coder	Parallel Computing Toolbox	Spreadsheet Link EX
Financial Derivatives T.	Partial Differential Equation T.	Stateflow
Financial Toolbox	Phased Array System T.	Statistics Toolbox
Fixed-Income Toolbox	RF Toolbox	Symbolic Math Toolbox
Fixed-Point Toolbox	Real-Time Windows Target	System Identification T.
Fuzzy Logic Toolbox	Robust Control Toolbox	SystemTest
Global Optimization T.	Signal Processing Toolbox	Vehicle Network Toolbox
HDL Coder	SimBiology	Wavelet Toolbox
HDL Verifier	SimDriveline	xPC Target
IEC Certification Kit	SimElectronics	xPC Target Embedded Option

Yukarıda listelenen toolboxlar arasında bazılarını açıklamak gerekirse;

Control Systems Toolbox (Kontrol Sistemleri Araç kutusu): Durum uzayı tekniklerini kullanarak kontrol mühendisliği ve sistemler teorisi ile ilgili fonksiyonlardan oluşmaktadır.

Robust-Control Toolbox (Robot Kontrol Araç kutusu): Robot kontrol sistemleri tasarımı ile ilgili fonksiyonlardan oluşmaktadır.

3.2.2. Kullanıcı Tarafından Oluşturulan M-Dosyaları

MATLAB'ta kullanıcıların çalıştırılmak üzere oluşturduğu M-dosyaları iki biçimde hazırlanabilir.

- Birincisi, komutlar dizisi biçiminde yazılan düz yazı (script) M-dosyaları.
- İkincisi MATLAB'ın kendi M-dosyalarına benzer biçimde hazırlanan fonksiyon(function) M-dosyalarıdır. Bu dosyalar herhangi bir ana program içinde bir alt program gibi kullanılabilirler.

3.2.2.1. Script M-Dosyaları

Scriptler bir nevi birden fazla satırın yani komutun bir kerede otomatik olarak çalıştırılması gibi düşünülebilir. Bu tanım itibariyle Scriptler bir çeşit program dosyasıdır. Dolayısı ile Matlab'de, M-Dosyaları oluşturularak programlama yapılabilir.

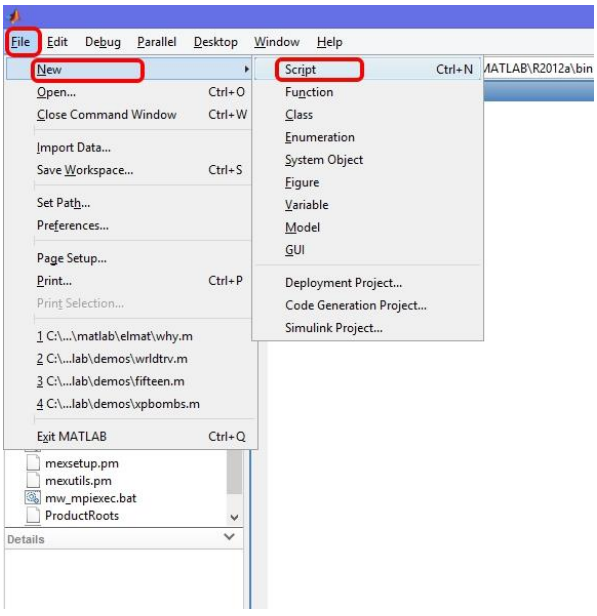
Scriptler fonksiyonlara göre daha basit yapıda programlardır çünkü giriş paremetresi almaz veya çıkışa değer döndürmezler. Yalnız bu durum bazen sorunlara yol açabilir. Script içinde yapılan işlemlerde kullanılan değişkenlere ya script içinde işlemden önce ya da script çalıştırılmadan önce Command Window ekranında mutlaka değer atanması gerekmektedir(Şekil 3.13).

Bir scripti iki şekilde oluşturabiliriz. Bunlar;

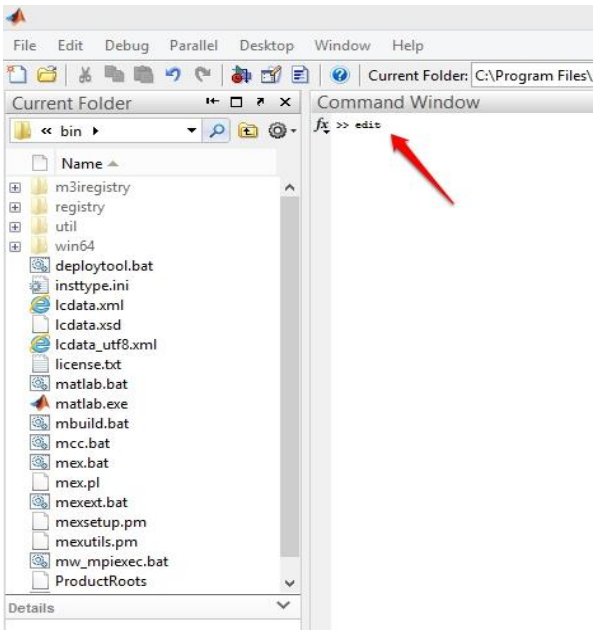
- File-New-Script i tıklamak(Şekil 3.4.)
- Komut satırında **edit** yazıp entere basmak.(Şekil 3.5.)

Her iki durum da script yazmak için kullanılacak editör ekranı ortaya çıkacaktır. Editör ekranı bağımsız bir pencerede(Şekil 3.6.) ya da MATLAB ekranında bitişik olarak(Şekil 3.7.) çalıştırılabilir.

Şekil 3.4. ve 3.5. de yeni script penceresi açma yolları görülmektedir.

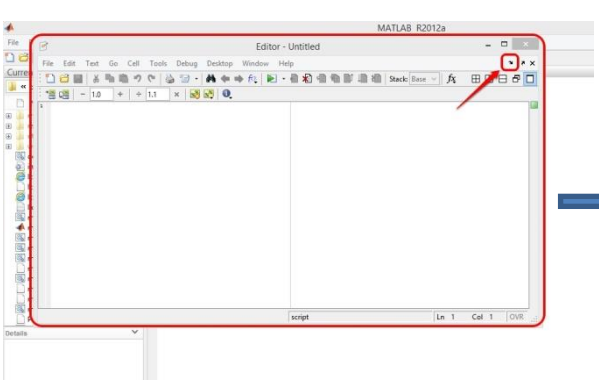


Şekil 3.4. Yeni Script Penceresi Açma

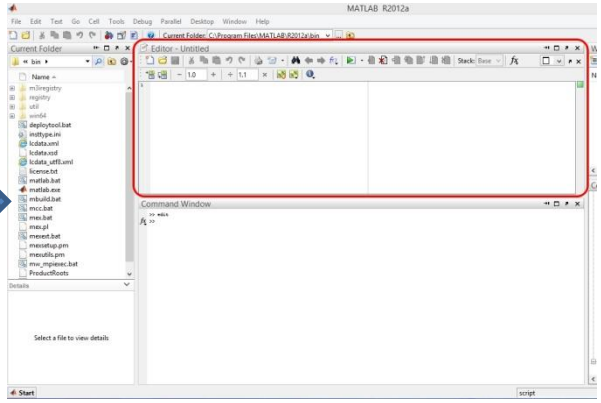


Şekil 3.5. Yeni Script Penceresi Açma

Şekil 3.6. ve 3.7. de yeni script penceresinin ekrandaki konumlanabileceği yerler görülmektedir



Şekil 3.6. Yeni Script Penceresi



Şekil 3.7. Yeni Script Penceresi

Hangi yöntemle açılırsa açılsın script yazıldıktan sonra .m uzantısıyla kaydedilmelidir(Şekil 3.12). Burada dikkat edilecek en önemli husus scripte verilen isimle aynı isimde bir değişken adı **olmayacağıdır.** Kaydedilen M-Dosyasını çalıştırmak için dosya adının Command Window da yazılıp enter tuşuna basılması yeterlidir(Şekil 3.14). MATLAB 2012 a sürümünde M-Dosyalarının default saklanma konumu C:\Program Files\MATLAB\R2012a klasörüdür. Ancak Windows 7 ve daha üst sürümlerde bu klasöre kayıt yapılmak istenildiğinde yetki hatası verebilir(Şekil 3.10). Bu klasörden farklı bir klasöre kaydedilen ya da farklı bir klasörde bulunan M-Dosyaları ismini yazarak çalıştırmak istenildiğinde tıpkı toolbox kurulumunda olduğu gibi **File-Set Path** seçeneğini tıklanarak gelen ekranda **Add Folder** veya **Add with Subfolders** butonunu tıklanıp M-Dosyalarımızın bulunduğu klasörü işaretlenmesi gerekmektedir.

Aktif klasördeki M-Dosyalarını listelemek için Command Window ekranında **what** komutu girilir.

Bir M-Dosyasının içeriğini **type <dosya adı>** komutuyla görülebilir(Şekil 3.16), eğer bir M-Dosyasının içeriğin, değiştirmek için **edit <dosya adı>** komutuyla M-Dosyasını editör ekranınında açarak istenilen değişiklikler yapılabilir. Bu yöntemle var olmayan bir M-Dosyası açılmak istendiğinde MATLAB bu dosyanın var olmadığını ve yaratmak istenilip istenilmediğini soracaktır(Şekil 3.8). Bir M-Dosyasını silmek için **delete <dosya adı>** komutunu kullanılır.

Yazılacak scriptte MATLAB fonksiyonlarının tamamını kullanılabilir. Ancak bu noktada I/O işlemleri için kullanılan 3 temel fonksiyon hakkında bilgi verilecektir. Bunlar;

- **Input**: Bilgi girişi komutu. Girilen verinin bir değişkene atanmasını sağlar.
  Ör: a=input('Birinci sayiyi giriniz: ');
- **Disp**: Ekrana her seferinde sadece bir ifadenin yada değişkenin yazdırılmasını sağlayan komut. Formatlı bir ekran çıkışı istenildiğinde **fprintf** komutu kullanılır. Fprintf komutu ile aşağıdaki ekran çıkış formatları kullanılabilir;
  - %d: Tamsayı
  - %c: Tek karakter
  - %f: Gerçek Sayı
  - %s: String

  Ör: disp('---------Iki sayinin toplamini bulan program----------');
  Ör: fprintf('Bu işlemde kullanılan rakam %d ve karekter %c dir',25, 's');

NOT: fprintf komutu kompleks sayıların sadece gerçek kısmını gösterir, bu nedenle kompleks sayılarda disp komutu kullanılmalıdır.

- **Plot:** Verilerin grafiğini çizmek için kullanılan komuttur. Oldukça geniş bir uygulama alanı olan bu komutu ileride daha ayrıntılı bir şekilde işlenecektir.
  Ör: Plot(x,y)
- **%**: Programın derlenmesi dışında tutulacak açıklama ifadelerin yazılmasını sağlayan komut. % komutunun bir diğer kullanımı ise oluşturulan M-Dosyası için help komutu kullanıldığında görüntülenecek yardım bilgisini oluşturabilmektir. M-Dosyasının başında boşluk bırakmadan yazılan % satırları **help <dosya adı>** komutu girildiğinde Command Window ekranında görüntülenir. Bu satırlarda yazan yardım ifadeleri üzerinden lookfor komutunu kullanarak arama yaptırılabilir. Bunun için **lookfor <yardım ifadesi> -all** komutu kullanılır.
  Ör: %Bu program bir silindirin hacmini hesaplar

NOT: Input, Disp ve fprintf komutu ile ekrana yazdırılacak ifadelerin '.....' işaretleri arasında yazılması gerekir. Ayrıca Input ve fprintf komutlarında \n parametresi ile alt satıra geçişi sağlanabilir. \n parametresinin '... ' ifadeleri arasında yazılması gerekir.

Script dosyada kullanılan bütün değişkenler MATLAB ekranında değişkenlerin ve değerlerinin listelendiği Workspace ekranında listelenir(Şekil 3.14). Yukarıda belirtilen hususu tekrar hatırlatmak gerekirse script içinde yapılan işlemlerde kullanılan değişkenlere ya script içinde işlemden önce ya da script çalıştırılmadan önce Command Window ekranında mutlaka değer atanması gerekmektedir.

Değişkenlere değer yüklenirken kullanabilecek bir diğer yöntem de değişken değerlerini bir dosyaya yüklemektir. MATLAB da değişken değerlerini kaydetmek ve geri yüklemek için iki farklı türde dosyaya kullanılabilir.

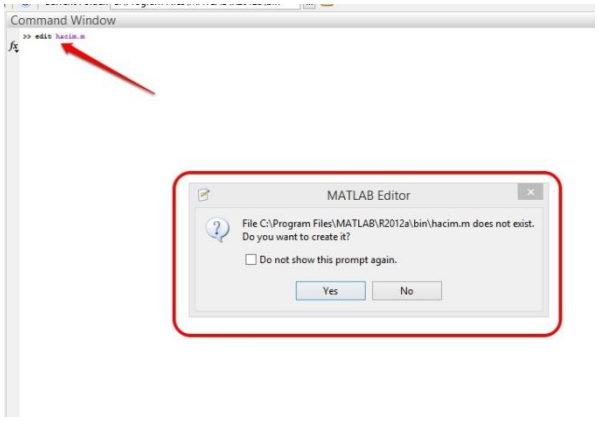
Bunlardan birincisi .dat dosyaları, ikincisi ise .mat dosyalarıdır. Bir .dat dosyası içinde değişken değerlerini kaydetmek için **save <filename> <matrixvariablename> –ascii,** geri yüklemek için **load <filename>**, dosya içeriğini görmek için **type <filename>** komutları kullanılır. Burada dikkat edilmesi gereken husun dosya adının .dat uzantısına sahip olması zorunluluğudur.

Diğer tür olan .mat dosyalarına ise .dat dosyalarına göre daha basit kayıt gerçekleştirilir. İki dosya türü arasında ki en önemli fark .dat dosyalarında kaydedilecek değişkenleri seçebilirken .mat dosyalarına o anda hafizada ki bütün değişkenleri kaydedilmesidir. Bir .mat dosyasına değişken değerlerini kaydetmek için **save<filename>**, geri yüklemek için **load<filename>** komutları kullanılır. Dosya adı yazarken uzantı yazılmasına gerek yoktur. Dosya .mat uzantısını otomatik olarak alır.

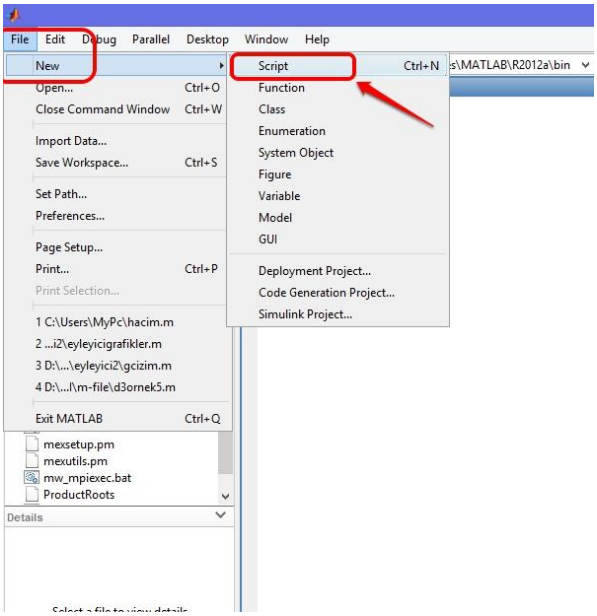
Ayrıca excel dosyalarından da veri okuyarak değişkenlere değer yüklenebilir, bunun için **değişken_adı=xlsread('<dosya_adı>')** komutu uygulanır.

Not: Bir ufak hatırlatma, **who** komutu ile hafizada bulunan değişkenlerin listesi görülebilir, **clear** komutu ile tamamını silinebilir.

Şekil 3.8. ve 3.9. da yeni script penceresi açma yolları görülmektedir.

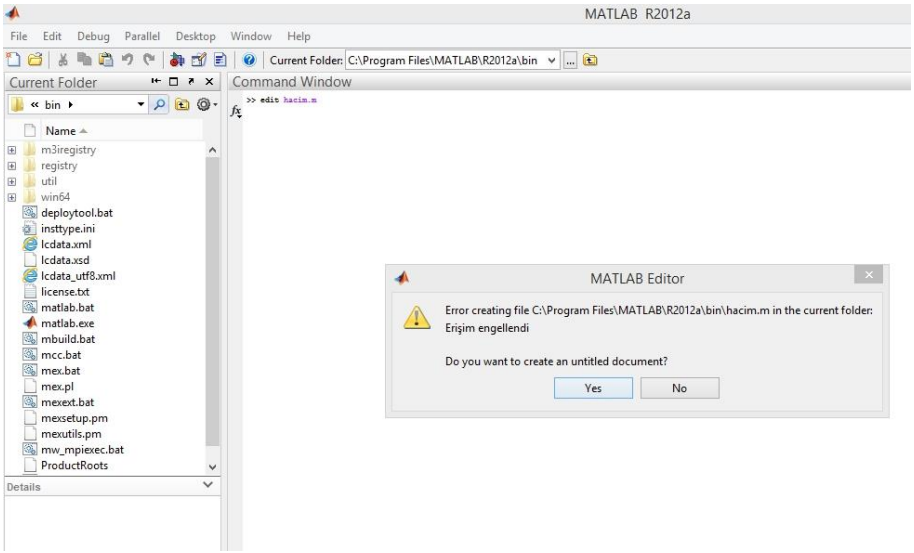


Şekil 3.8. Edit Komutu ile M-Dosyası Oluşturma



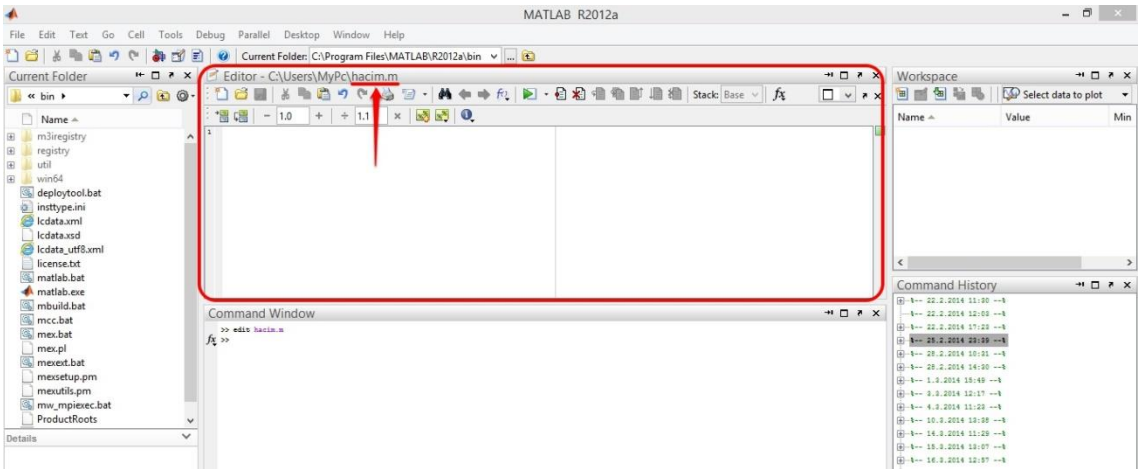
Şekil 3.9. File-New-Script ile M-Dosyası Oluşturma

Şekil 3.10. da Program Files klasörüne erişim hatası görülmektedir.



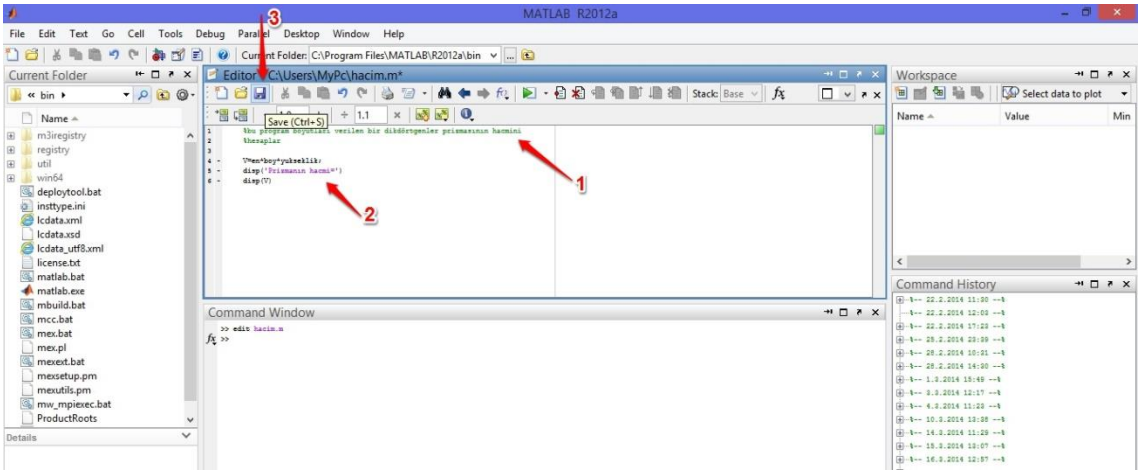
Şekil 3.10. Erişim Hatası Uyarısı

Şekil 3.11. de Editör penceresi görülmektedir.



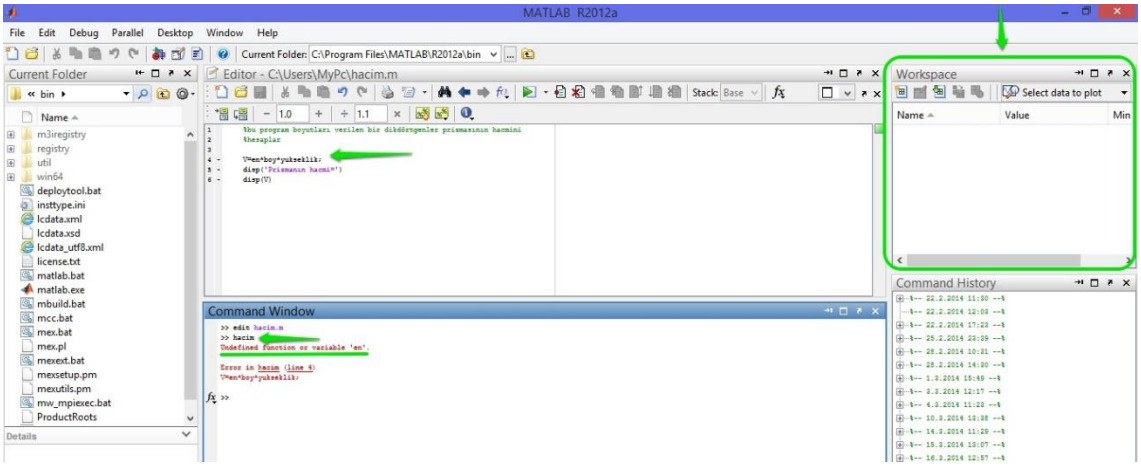
Şekil 3.11. Editör Penceresi

Şekil 3.12. de M-Dosyasının kaydeilmesi görülmektedir.



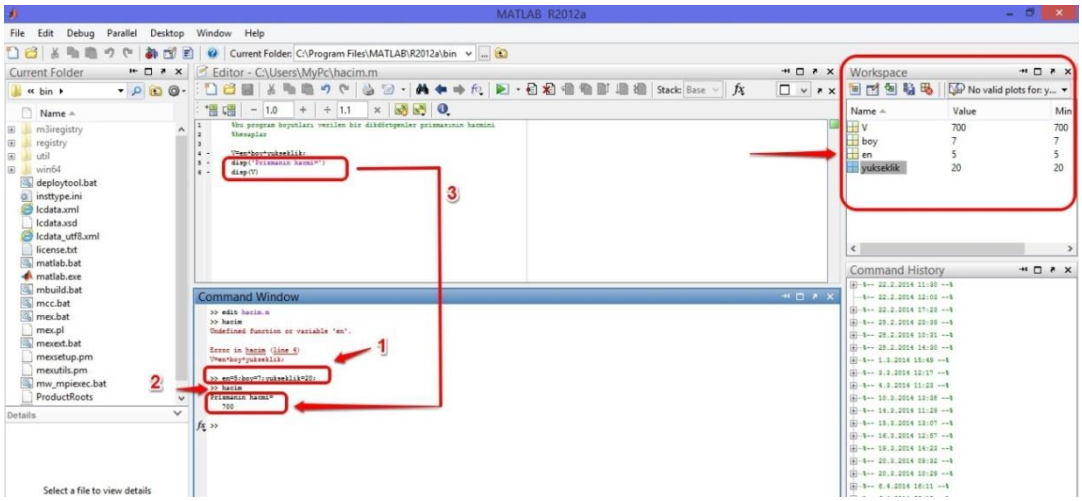
Şekil 3.12. M-Dosyasının Kaydedilmesi

Şekil 3.13. de işlemde kullanılmasına rağmen değer yüklenmemiş değişkenin yol açtığı hata mesajı görülmektedir.



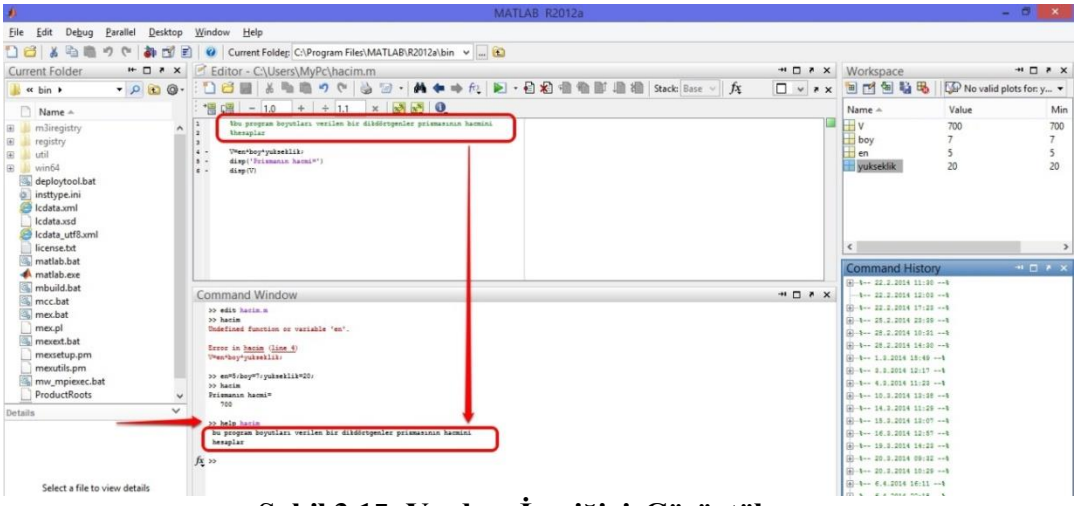
Şekil 3.13. Değişken Hatası

Şekil 3.14. de bir M-Dosyasının çalıştırılması görülmektedir.



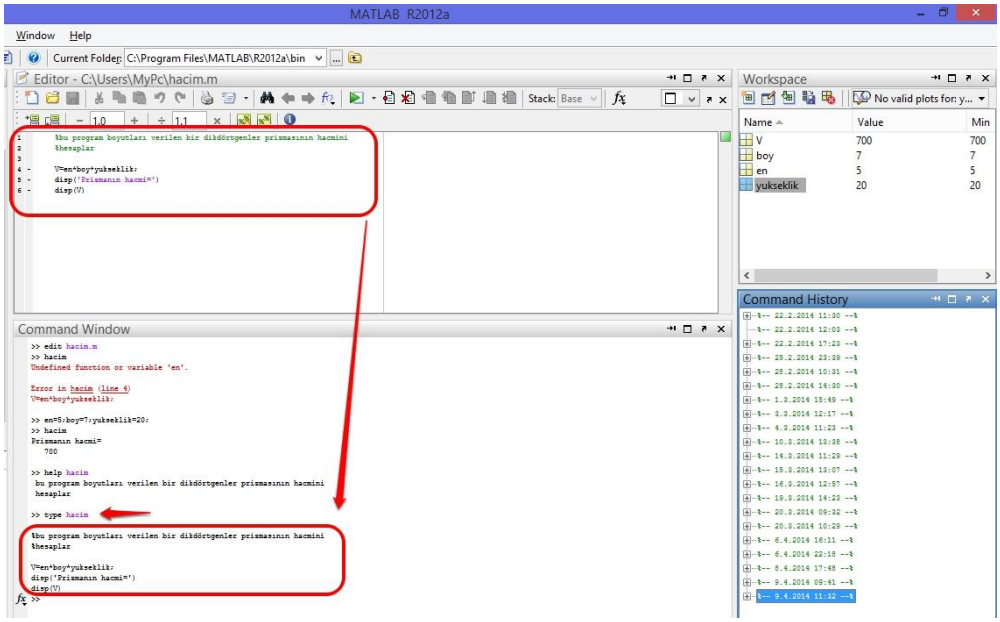
Şekil 3.14. M-Dosyasının Çalıştırılması

Şekil 3.15.de M-Dosyasının yardım içeriğini oluşturma ve görüntüleme görülmektedir.



Şekil 3.15. Yardım İçeriğini Görüntüleme

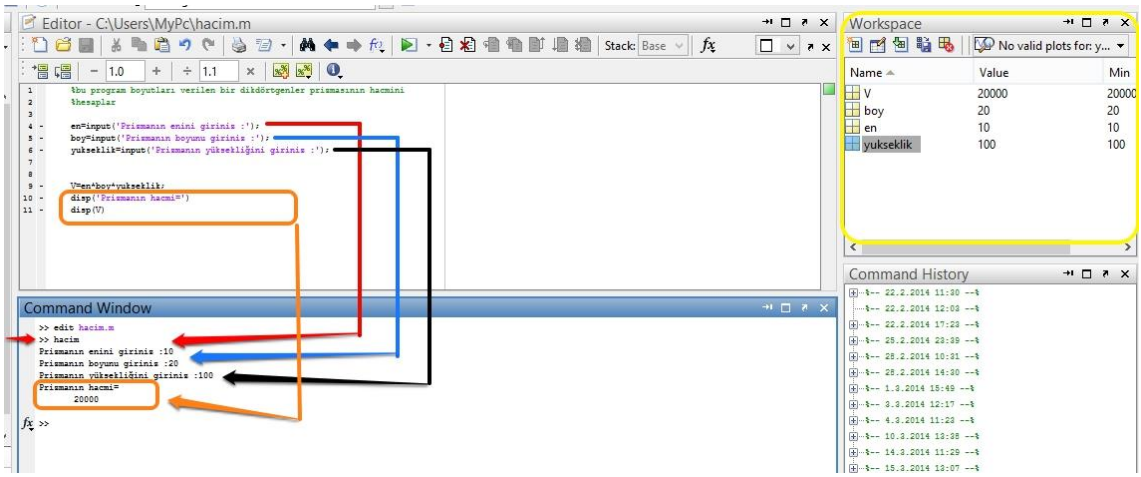
Şekil 3.16. da type komutu ile bir M-Dosyasının içeriğinin görülmesi görülmektedir.



Şekil 3.16. Type Komutu

Yukarıdaki şekillerde basit bir M-Dosyasının oluşturulması ve çalıştırılması ekranları görülmektedir. Şekil 3.8. veya Şekil 3.9. yöntemlerinden herhangi biriyle açılan editör penceresinde oluşturulan scriptin yazımı tamamladıktan sonra Şekil 3.12. olduğu gibi kaydedilir. Burada tekrar hatırlaması gereken husus kesinlikle scriptle aynı isimde bir değişken adının olmaması gerektiğidir. Şekil 3.13. da görüldüğü üzere scripti çalıştırmak için Command Windows ekranında scriptin adı yazılıp enter tuşuna basılır ancak scriptte herhangi bir işlemde kullanılan değişkenlere bir değer yüklenmemesi durumunda script çalışmaz. Şekil 3.14. de 1. adımda görüldüğü üzere bu değişkenlere Command Window ekranında da değer yüklenebilir. Ancak bu durum birçok soruna yol açabilecek oldukça kullanışsız bir yöntemdir. Bu sebeple scriptte yapılacak işlemlerde kullanılan değişken değerlerinin scriptin içinde girilmesi sağlanmalıdır. Bir değişkene değer girişi için **input** fonksiyonunu kullanılır.

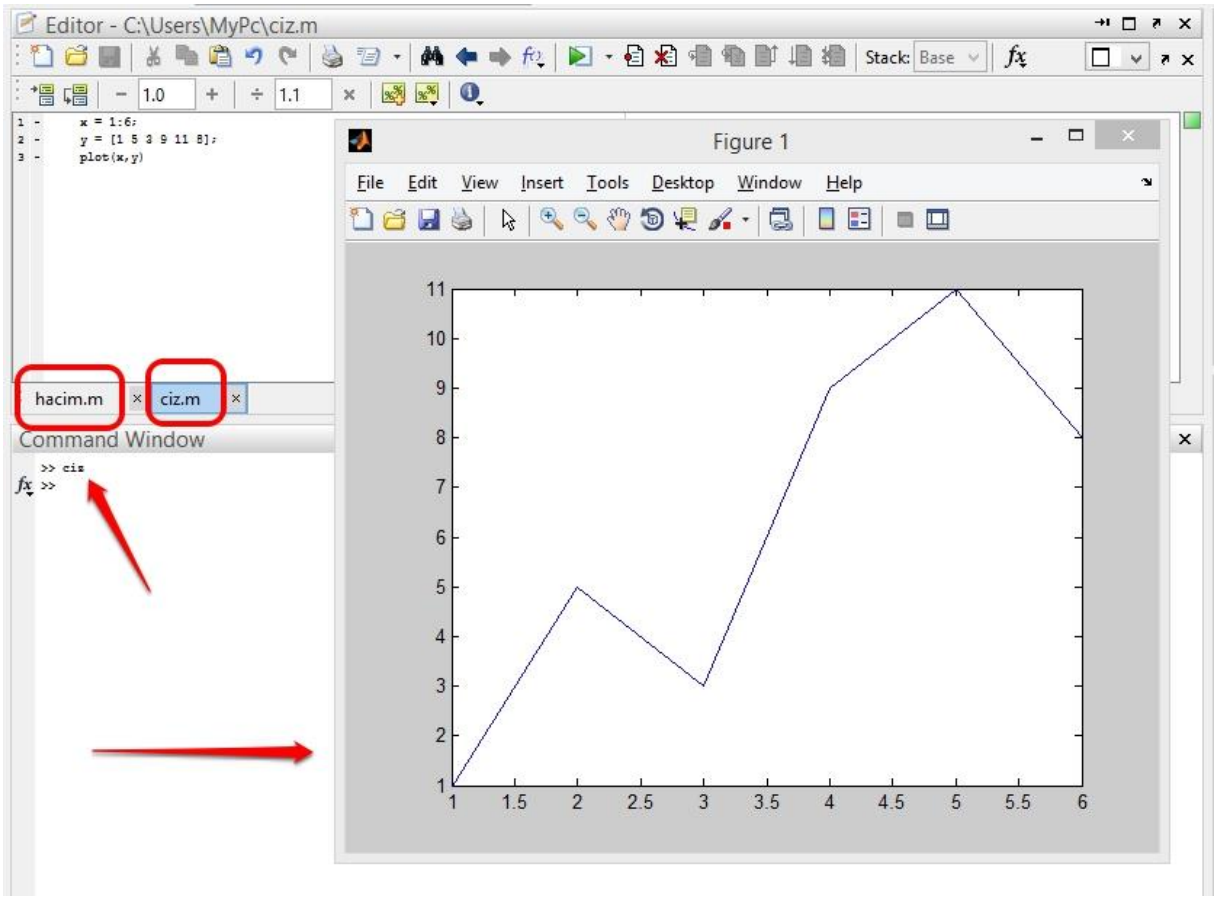
Şekil 3.17. de input fonksiyonun uygulanması görülmektedir.



Şekil 3.17 Input Fonksiyonu

Şekil 3.17. de scriptte kullanılan değişkenlere kullanıcının değer ataması için input fonksiyonunu kullanılmaktadır. Script çalıştırıldıktan sonra sırasıyla değişkenler için değer ataması yapılıp hesaplanan sonuç ekrana yazılmaktadır. Değişken değerleri Workspace ekranınında da görülebilir.

Şekil 3.18. de plot fonksiyonun uygulaması görülmektedir.



Şekil 3.18. Plot Fonksiyonu

Şekil 3.18. de scripte grafiksel destek için plot fonksiyonunu kullanılmaktadır, şekilde görüldüğü üzere scripti düzenlemek için kullandığımız Editör ekranın da düzenlenilen her script ayrı bir sekme olarak görülmektedir. Plot fonksiyonu ile Figure penceresinde istenilen değişkenlerin grafiğini görülür.

Plot fonksiyonunun kullanımında grafiğe renk verilebilir. Bunun için **Plot(x,y, 'renk_kodu')** komutu uygulanır.

Tablo 3.4. Renk Kodları

Renk Kodları	
Renk Kodu	Değeri
b	Blue(Mavi)
c	Cyan(Açık Mavi)
g	Green(Yeşil)
k	Black(Siyah)
m	Magenta(Mor)
r	Red(Kırmızı)
y	Yellow(Sarı)

Plot fonksiyonu ile çizgisel grafiklerden farklı işaretli grafiklerde oluşturulabilir. Bu işaretli grafikler renkli ya da renksiz olarak kullanılabilir. Bunun için **Plot(x,y, 'işaret_kodu')**komutunu uygulanır.

Tablo 3.5. İşaret Kodları

İşaret Kodları	
İşaret Kodu	Değeri
o	Circle(Daire)
d	Diamond(Elmas)
h	Hexagram(Altı Köşeli Yıldız)
p	Pentegram(Beş Köşeli Yıldız)
+	Plus(Artı)
.	Point(Nokta)
s	Square(Kare)
*	Star(Yıldız)
v	Down Triangle(Aşağı Doğru Üçgen)
<	Left Triangle(Sola Doğru Üçgen)
>	Right Triangle(Sağa Doğru Üçgen)
^	Up Triangle(Yukarı Doğru Üçgen)
x	X Mark(X işareti)

Plot fonksiyonunda çizgisel grafiklerde kullanılan çizginin stili değiştirilebilir. Bunun için **Plot(x,y, 'çizgi_stili')**komutu uygulanır.

Tablo 3.6. Çizgi Stilleri

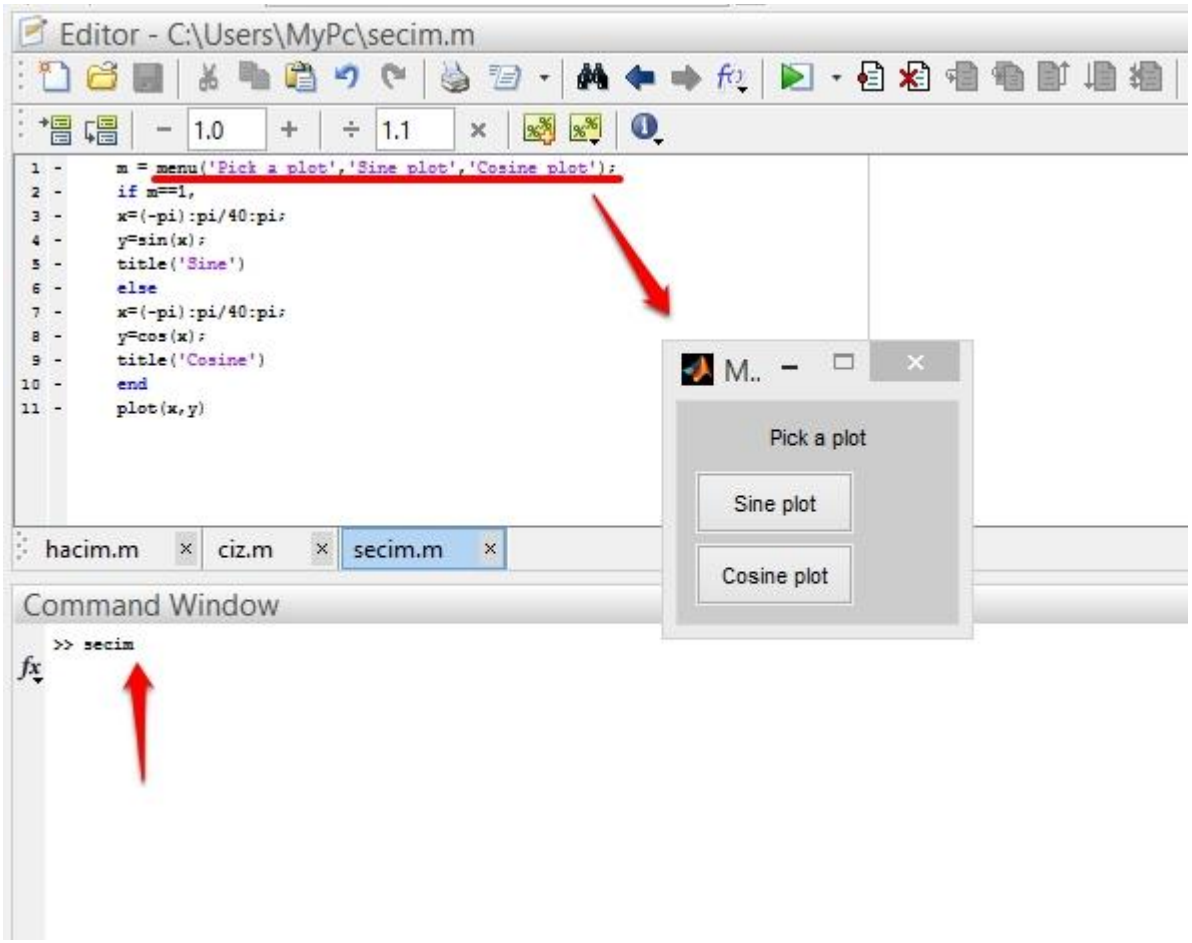
Çizgi Stilleri	
Çizgi Stili	Değeri
--	Dashed(Kesik Çizgi)
-.	Dash dot(Noktalı Kesik Çizgi)
:	Dotted(İki Nokta Üst Üste Çizgi)
-	Solid(Normal Çizgi)

İşaret ya da çizgi stiline karar verildikten sonra renk kodları da değiştirilebilir. Ayrıca **Grid on** ya da **Grid off** komutlarıyla grafiğin arka planında grid işaretlerinin görünüp görünmemesi de ayarlanabilir. Ayrıca grafik başlığı için **title**, x ekseni başlığı için **xlabel**, y ekseni başlığı için **ylabel** ve gösterge paneli için **legend** fonksiyonlarını kullanılır. Title, xlabel, ylabel ve legend fonksiyonu ile ekranı yazdırılacak ifadeler tırnak içinde yazılmak zorundadır.

Plot fonksiyonu dışında **bar** ve **pie** gibi 2 boyutlu ya da **plot3** ve **surface** gibi 3 boyutlu grafik oluşturmaya yarayan fonksiyonlar da MATLAB'da kullanılabilir.

Grafik fonksiyonları bir sonraki bölümde daha ayrıntılı olarak işlenecektir.

Şekil 3.19. da menu fonksiyonun uygulaması görülmektedir.



Şekil 3.19. Menu Fonksiyonu

Şekil 3.19. da scriptte kullanıcının seçim yapabilmesi için **menu** fonksiyonunu kullanılmaktadır, kullanıcı scripti çalıştırdığında karşısına iki seçenekli bir menü gelecek ve istediği seçim doğrultusunda script çalışmaya devam edecektir. Menu komutundan sonra kullanıcının yaptığı seçime göre dallanma yapabilmek için **if** komutunu kullanılmalıdır. İf komutu bölüm 4.2.1 de incelenebilir.

3.2.2.2. Function M-Dosyaları

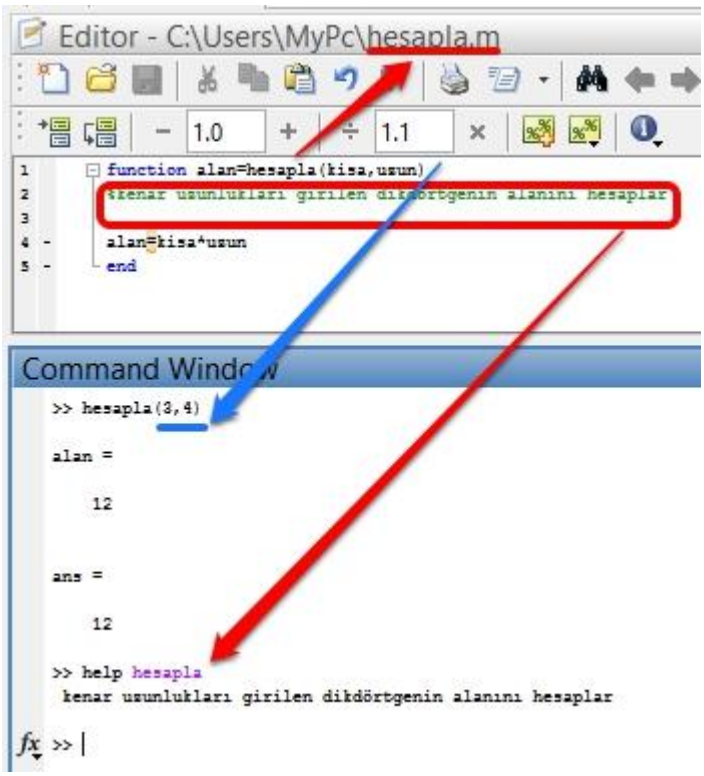
Aslında bu ana kadar birçok function yapısında dosya kullandık gerek MATLAB'ın built-in fonksiyonları gerekse dahili ya da harici toolboxlarda kullanılan M-Dosyalarının tamamı function yapısındadır. Functionlar giriş paremetresi alabilen ve geri değer döndürebilen scriptler olarak düşünülebilir. Functionlar scriptlere göre birçok açıdan daha fonksiyoneldir. Functionlar da scriptler gibi Editör ekranında oluşturulur ve .m uzantısıyla kaydedilir.

Function yapısında dikkat edilmesi gereken 4 önemli husus vardır. Bunlar;

- Dosya "**Function**" ifadesiyle başlamalıdır.
- Dosya fonksiyona verilen isimle kaydedilmek zorundadır. Örneğin aşağıdaki örnekte dosya adı **my_fun.m** olmak zorundadır.
- Dosya içinde tanımlanan tipler sadece dosya içinde geçerlidir yani localdir.
- Giriş ve çıkış parametlerinin sayısı dosyanın ilk satırında deklare edilmek zorundadır.

Tablo 3.7. Function yapıları

Function Yapıları
Function outputs = my_fun(inputs) %Fonksiyon hakkındaki açıklamalar code . . code outputs=... end



Şekil 3.20. de basit yapıda işlem yapabilen bir fonksiyon görülmektedir. Dikkat edilmesi gereken hususlara uygun olarak dosya function ifadesiyle başlamış çıkış parametresi olarak alan, giriş paremetreleri olarak ise kisa ve uzun değişkenleri tanımlanmıştır. Dosya fonksiyona verilen isimle yani hesapla.m olarak kaydedilmiştir. Command Window ekranında tıpkı scriptler gibi çalıştırılan function'lar verilen giriş paremetlerine uygun olarak sonucu hesaplayıp çıkış değerini göstermiştir.

Şekil 3.20. Function Dosyaları

3.3. Ölçme ve Değerlendirme

Aşağıdaki soruları çözünüz.

1) Kullanıcının girdiği ifadenin uzunluğunu bulan scripti yazınız.
2) Santigrad olarak girilen sıcaklık ölçümünü Kelvin ve Fahrenhayt olarak çeviren scripti yazınız.
3) Giriş parametresi olarak girilen a b c kenar uzunluklarına göre üçgenin çevresini hesaplayıp yazdıran function u yazınız.
4) Giriş paremetresi olarak girilen x_{bas}, x_{bit}, y_{bas}, y_{bit} ve el_say sayılarına göre x ve y vektörlerini oluşturup bunların çizgi grafiğini çizen function u yazınız

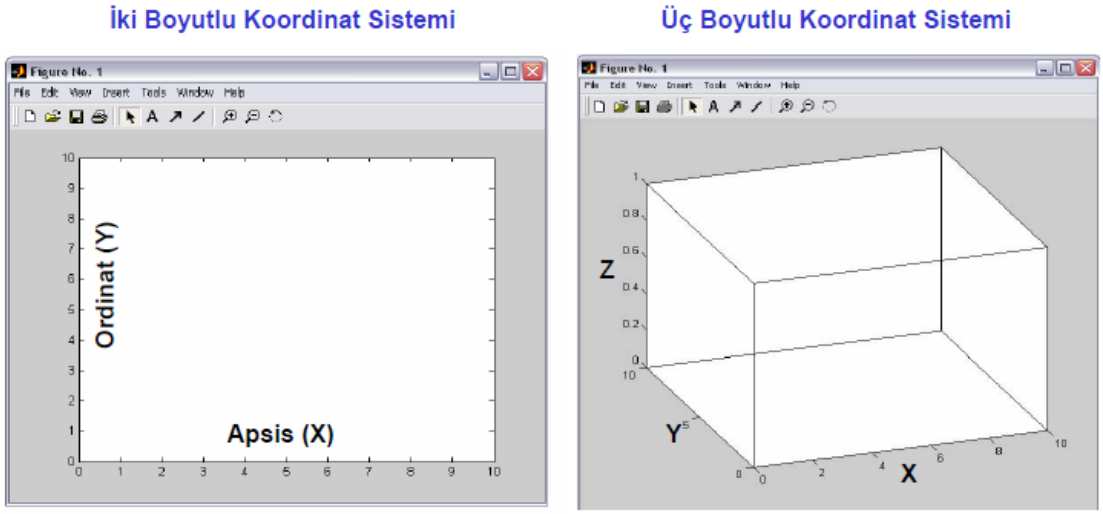
Not: Soruların cevaplarını Ek2 de bulabilirsiniz.

4.GRAFİKLER

MATLAB grafik sistemi, verilerin hazırlanmasında ve görselleştirilmesinde çok değişik ve kendine has özellikleriyle bir kullanıcılara büyük kolaylık sağlamaktadır. MATLAB programında grafikler temel olarak 2 ana grupta incelenebilir. Bunlar;

1) 2 Boyutlu(2D) Grafikler
2) 3 Boyutlu(3D) Grafikler

Şekil 4.1. de 2D ve 3D koordinat sistemleri görülmektedir.



Şekil 4.1. 2D ve 3D Koordinat Sistemi

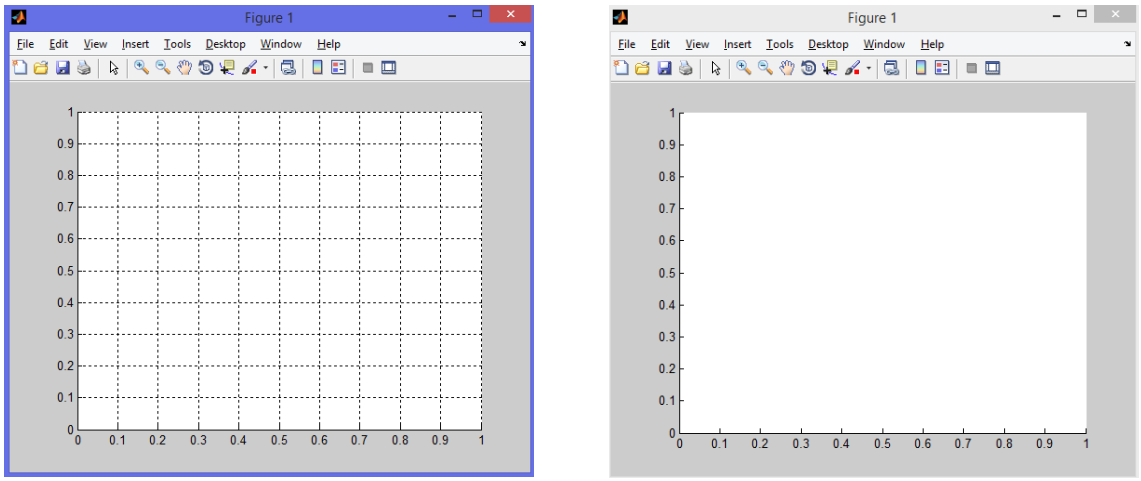
MATLAB 2D ve 3D başta olmak üzere çok gelişmiş grafik araçları sunar. Bunlar;

- Çizgi (line) grafikler (plot, plot3, polar)
- Yüzey (surface) grafikler (surf, surfc)
- Ağ (mesh) grafikler (mesh, meshc, meshgrid)
- Eş yükselti (contour) grafikler (contour, contourc, contourf)
- Çubuk (bar), pasta(pie) vb. özel grafikler (bar, bar3, hist, rose, pie, pie3)
- Animasyonlar (movein, movie)
- Resim ve videolar (imread, image, mplay, Videoreader)

MATLAB' da bütün grafik, resim, video ya da animasyonların çıktısı "figure" adı verilen pencerelerde verilir. X ve y eksenlerinin sayısal sınırları aksi belirtilmedikçe grafiği çizilecek verilere göre ayarlanır.

2D ya da 3D olmasına bakılmaksızın Figure penceresi için geçerli olan komutlar şunlardır;

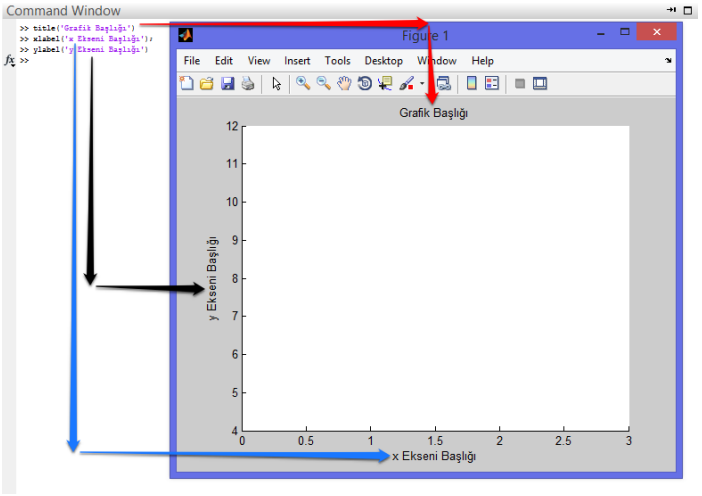
1) **Grid:** Figure penceresinde ki grafiğin arka planın ölçekli olarak gösterilmesini sağlar. **Grid** ve ya **Grid On** yazarak ölçekli gösterimi aktif hale getirilebilir. Bu özelliği kapatmak için **Grid Off** komutu kullanılır.
   Şekil 4.2. de Grid On ve Grid Off durumları görülmektedir



Şekil 4.2. Grid On, Grid Off

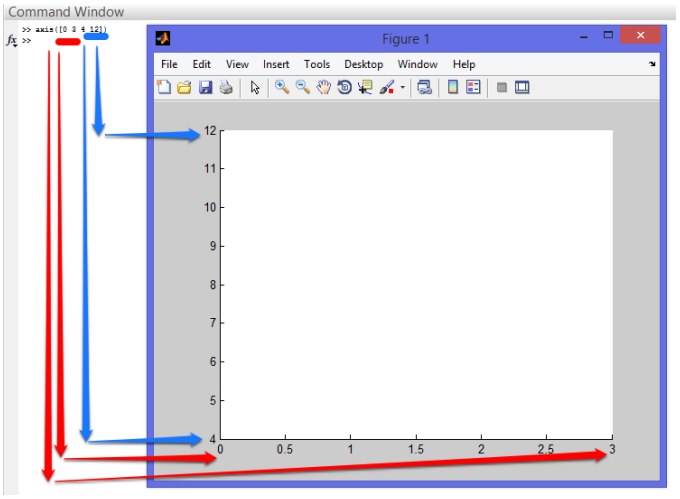
2) **title('....'):** Grafik başlığını belirlemek için kullanılan komuttur(Şekil 4.3).
3) **xlabel('....'):** Çizilen grafiğin x ekseni başlığını ayarlar(Şekil 4.3).
4) **ylabel('....'):** Çizilen grafiğin y ekseni başlığını ayarlar(Şekil 4.3).
5) **zlabel('....'):** Çizilen grafiğin z ekseni başlığını ayarlar(3 boyutlu çizimlerde kullanılabilir).
6) **axis([$x_{başlangıç}$ $x_{bitiş}$ $y_{başlangıç}$ $y_{bitiş}$]):** Yukarı da belirtildiği gibi x ve y eksenlerinin sayısal sınırları grafiği çizilecek verilere göre ayarlanır. Şekil 4.2. de görüldüğü gibi boş grafik ekranların da x ve y eksenleri 0-1 arasında ayarlanmıştır. Grafik eksenlerindeki sayı değerlerinin başlangıç ve bitiş noktalarını ayarlamak için axis komutu kullanılır(Şekil 4.4).
7) **Hold:** Birden fazla grafiğin tek bir figure penceresinde ve tek eksen kullanarak göstermek için kullanılan komuttur. **Hold On** yazarak bu özelliği açar, **Hold Off** yazarak da kapanmasını sağlanabilir(Şekil 4.5).
8) **figure:** Yeni bir figure penceresinin oluşturulup açılmasını sağlar(Şekil 4.6).

Şekil 4.3. de title, xlabel ve ylabel komutlarının kullanımı görünmektedir.



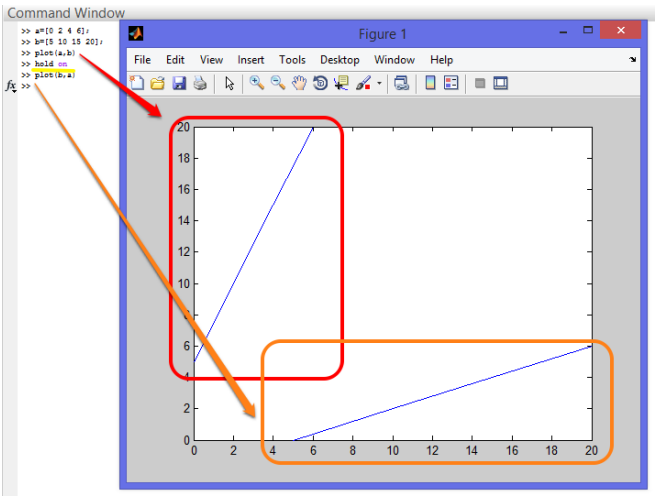
Şekil 4.3. Title, xlabel ve ylabel Komutlarının Kullanımı

Şekil 4.4. de axis komutunun kullanımı görünmektedir.



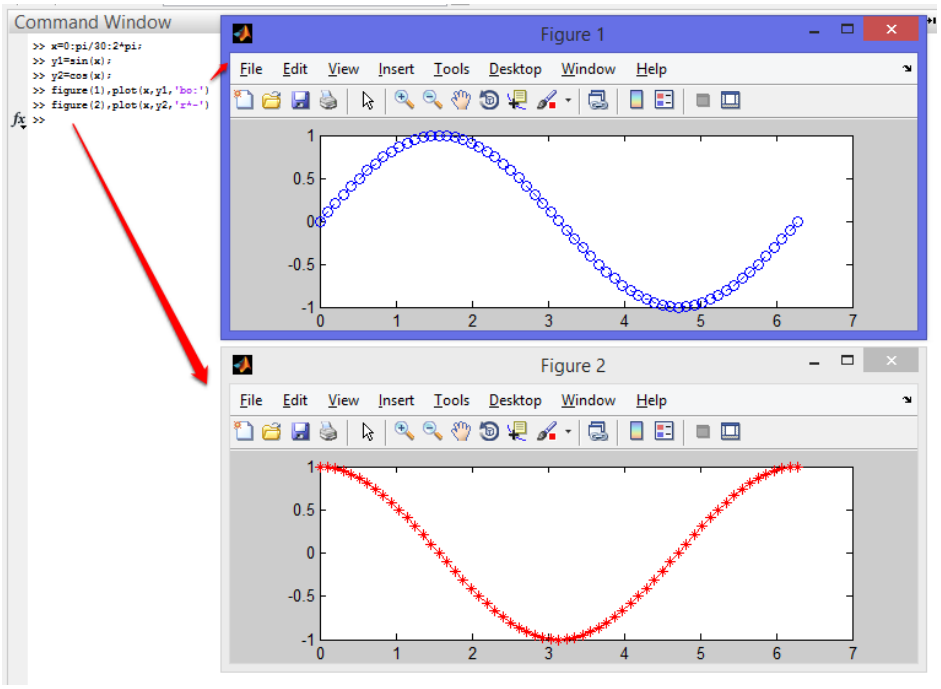
Şekil 4.4. axis Komutunun Kullanımı

Şekil 4.5. de hold komutunun kullanımı görünmektedir



Şekil 4.5. hold Komutunun Kullanımı

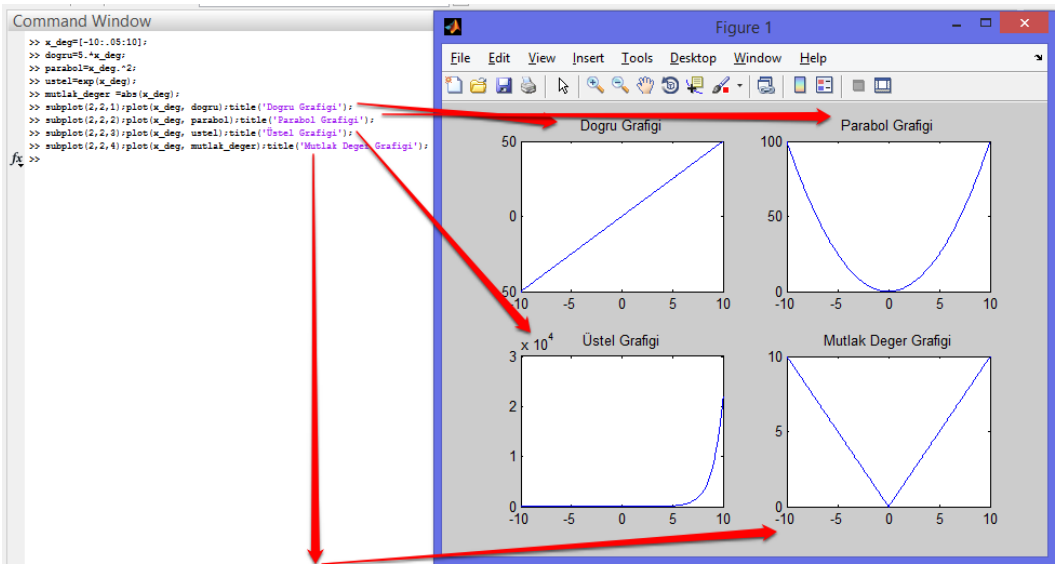
Şekil 4.6. da figure komutunun kullanımı görünmektedir



Şekil 4.6. figure Komutunun Kullanımı

Şekil 4.6. da figure komutu kullanılarak birden fazla grafik penceresinin oluşturulması sağlanmıştır.

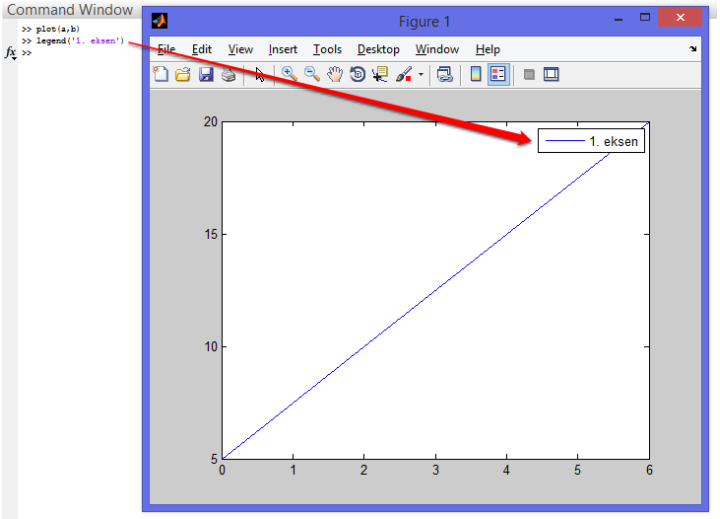
9) subplot(r,c,n): Birden fazla grafiği tek bir figure penceresinde ve ayrı ayrı eksenler kullanarak(yani bir nevi ekranı bölerek) göstermek için kullanılan komuttur. Ekran kaç satır ve kaç sütüna bölünecekse r(row) ve c(column) paremetrelerinde belirtilip kaçıncı sırada görüntüleneceği n parametresinde bildirilip ekranda görüntülenmesi sağlanır. Şekil 4.7. de subplot komutunun kullanımı görünmektedir.



Şekil 4.7. subplot Komutunun Kullanımı

Şekil 4.7. de subplot komutu kullanılarak tek bir figure ekranında birden fazla grafik oluşturulması sağlanmıştır. Grafiğin ekrandaki görünme sırası işlem sırasından bağımsız tamemen n parametresine bağlıdır.

10) **text(x,y, '....'):** x,y koordinatlarında istenilen yazının görünmesini sağlar.
11) **gtext('....'):** İstenilen yazının figure penceresinde ki grafik üzerinde te ilk tıklanan noktada kalıcı olarak görüntülenmesini sağlar
12) **legend('....', '....', '....'.....):** Grafik için gösterge kutusunun gösterilmesini sağlar. Kapatmak için legend off komutunu kullanılır.
Şekil 4.8. de legend komutunun kullanımı görünmektedir.



Şekil 4.8. legend Komutunun Kullanımı

13) **close all:** Açık bulunan bütün figure pencerelerini kapatır.

4.1. 2 Boyutlu(2D) Grafikler

MATLAB da birçok 2 boyutlu(2D) grafik çizim fonksiyonu bulunmaktadır. Bunlar;

1) Plot, Ezplot
2) Bar, Barh
3) Hist
4) Stairs
5) Stem
6) Pie
7) Area
8) Quiver(Vektör)
9) Kanava
10) Polar
11) Loglog
12) Semilogx, Semilogy
13) Contour

4.1.1. Plot, Ezplot

Plot x-y koordinat düzlemine lineer grafik çizmek için kullanılır. **Plot(x)** ya da **Plot(x,y)** şeklinde kullanılır. Grafik çizgisinin özelliklerini değiştirmek istediğimizde **Plot(x,y,'s')** şeklinde de kullanılabilir. S parametresinde çizginin rengi stili ya da biçimi belirtilir. Bu özelliklerden sadece biri veya ikisi yada üçü birden s parametresinde ayarlanabilir.

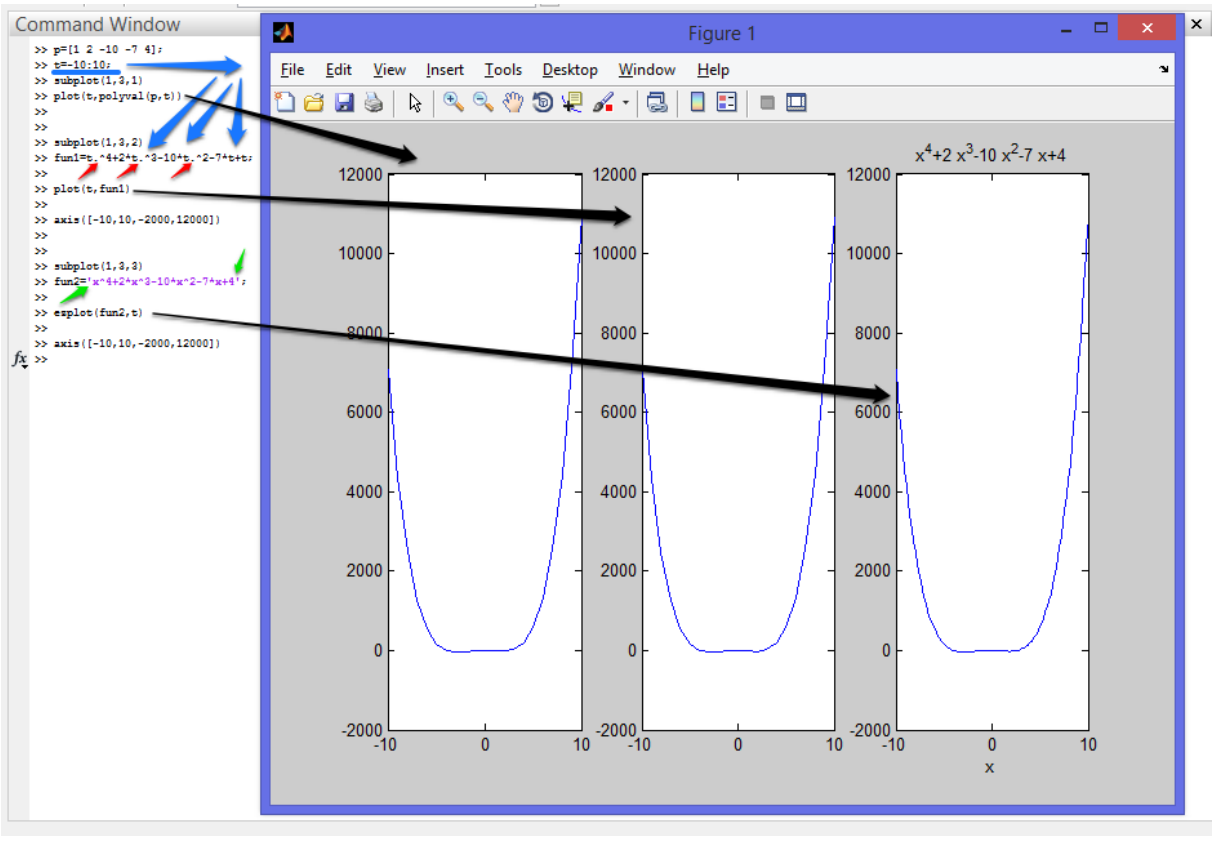
Bir fonksiyonun istenilen aralıktaki grafiğini çizmek için 3 farklı yol izlenebilir. Bunlar;

1) Plot komutu Polyval komutu ile birlikte kullanarak bir fonksiyonun istenilen aralıktaki grafiği çizdirebilir. Bunun için **plot([aralık],polyval([katsayılar],[aralık]))** komutu uygulanır.
2) Plot komutunu yalnız başına kullanarak da fonksiyonun istenilen aralıktaki grafiğini çizelebilir. Bunun için **plot([aralık],fun)** komutu uygulanır. Burada dikkat edilecek üç önemli husus vardır. Bunlardan birincisi aralığı tanımladığımız vektör değişkeninin ismi ile grafiği çizilecek fonksiyonun değişken ismi aynı olması, ikincisi fonksiyonda ki bütün üs alma işlemleri birebir işlem şeklinde yani '.' operatörü kullanılarak yapılması ve son olarak üçüncüsü ise fun değişkeninde tanımlanan fonksiyon kesinlikle tırnak içinde yazılmamasıdır.
3) Bir fonksiyonun grafiğini çizmek için kullanabileceğimiz bir diğer komut da Ezplot'dur. Ezplot komut **ezplot(fun, [aralık])** şeklinde uygulanır. Ezplot komutu **ezplot(fun)** şeklinde uygulandığında aralığı -2π 2π olarak alır. Bu aralık -6 6 aralığına karşılık gelir.

NOT: Direkt olarak fonksiyonu yazarak çözüme ulaşabileceğimiz komutlarda(Bu kuralın tek istasnası plot komutu için yazacağımız fonksiyondur)fonksiyonu '...' arasına yazmalı ya da komuttan hemen önce fonksiyonda kullanılan harfleri(x,y vb) **syms** fonksiyonu ile sisteme tanıtılmalıdır. Aksi halde hata mesajıyla karşılaşılır.

Ör: $y=x^4 + 2x^3 - 10x^2 - 7x + 4$ fonksiyonu -10 – 10 aralığındaki grafiğini yukarıda ki 3 yöntemi de kullanarak çiziniz.

Çözüm için geçerli olan aralık vektörü [-10:10], katsayı vektörü [1 2 -10 -7 4], plot için fun değeri x.^4+2*x.^3-10*x.^2-7*x+4 ve ezplot için fun değeri 'x^4+2*x^3-10*x^2-7*x+4' dür.



Yukarıda $y=x^4 + 2x^3 - 10x^2 - 7x + 4$ fonksiyonu 3 farklı yöntem kullanılarak çizdirilmiştir. Grafiklerin tek bir figure penceresinde görünmesini sağlamak için subplot komutu kullanılmıştır. Çizimler plot ve ezplot komutları kullanılarak gerçekleştirilmiştir. Çizimlerden önce katsayı vektörü p, aralık vektörü t, plot komutu için kullanılacak fun1 ve ezplot komutu için kullanılacak fun2 fonksiyonlarının tanımlanması sağlanmıştır

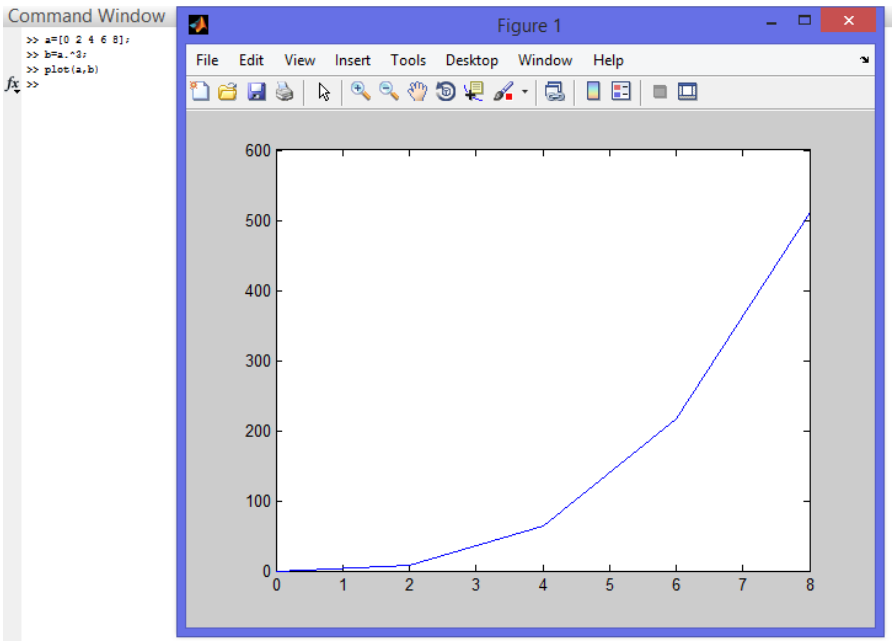
Tablo 4.1. Plot Fonksiyonu Çizgi Ayarları

Renk Kodları	
Renk Kodu	Değeri
B	Blue(Mavi)
C	Cyan(Açık Mavi)
G	Green(Yeşil)
K	Black(Siyah)
M	Magenta(Mor)
r	Red(Kırmızı)
y	Yellow(Sarı)

İşaret Kodları	
İşaret Kodu	Değeri
o	Circle(Daire)
d	Diamond(Elmas)
h	Hexagram(Altı Köşeli Yıldız)
p	Pentegram(Beş Köşeli Yıldız)
+	Plus(Artı)
.	Point(Nokta)
s	Square(Kare)
*	Star(Yıldız)
v	Down Triangle (Aşağı Doğru Üç.)
<	Left Triangle(Sola Doğru Üçgen)
>	Right Triangle(Sağa Doğru Üçgen)
^	Up Triangle(Yukarı Doğru Üçgen)
x	X Mark(X işareti)

Çizgi Stilleri	
Çizgi Stili	Değeri
--	Dashed(Kesik Çizgi)
-.	Dash dot(Noktalı Kesik Çizgi)
:	Dotted(İki Nokta Üst Üste Çizgi)
-	Solid(Normal Çizgi)

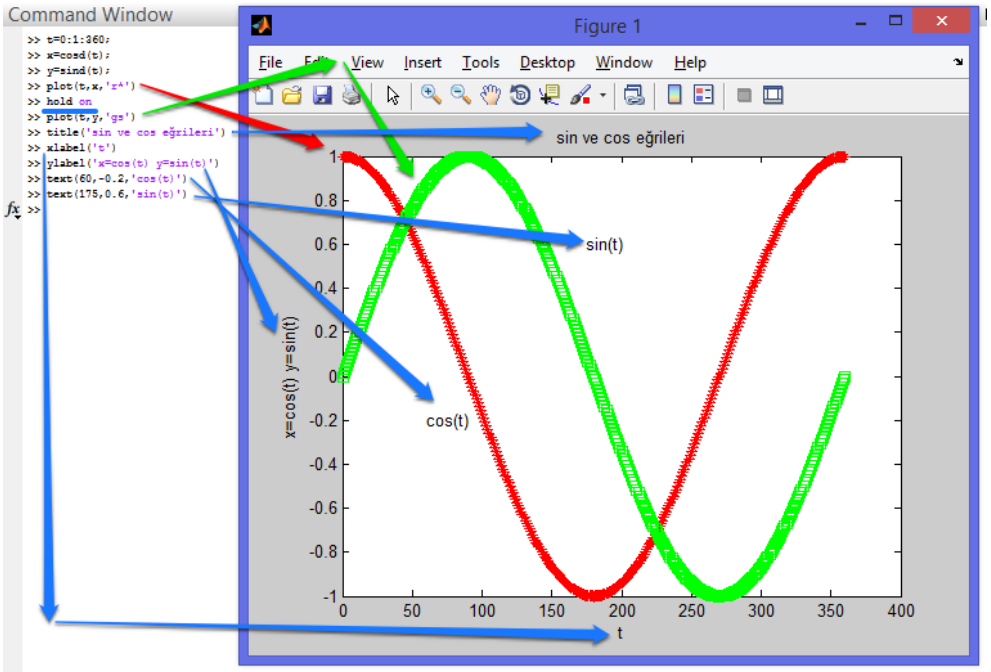
Şekil 4.9. da plot fonksiyonun kullanımı görünmektedir.



Şekil 4.9. Plot Fonksiyonun Basit Kullanımı

Şekil 4.9. da öncelikle a ve b vektörü yaratılıp daha sonra plot komutu ile grafikleri çizilmiştir

Şekil 4.10. da plot fonksiyonun hold fonksiyonuyla beraber kullanımı görünmektedir.



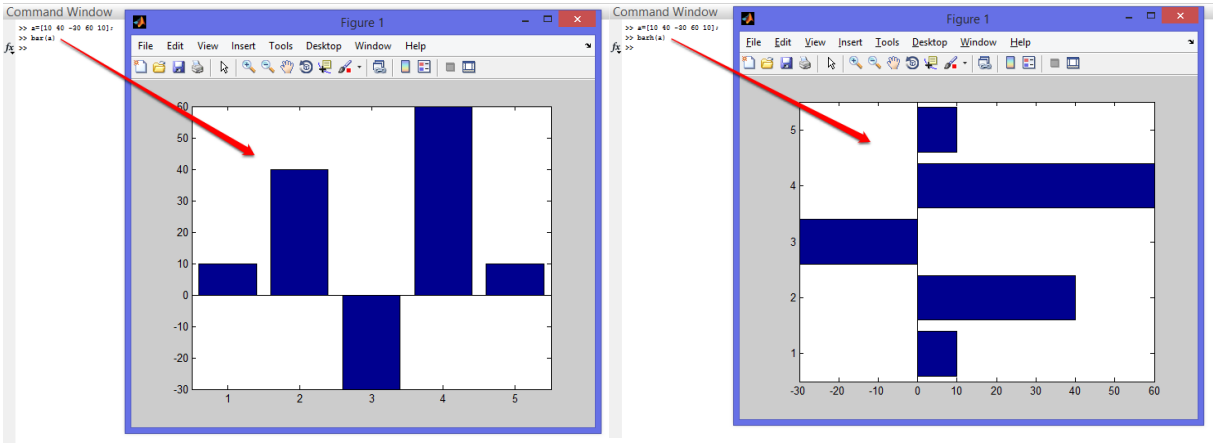
Şekil 4.10. Plot Fonksiyonun Gelişmiş Kullanımı

Şekil 4.10. da öncelikle t vektörü yaratılmış, daha sonra cos grafiği çizmek için x, sinüs grafiği çizmek için y eğrileri tanımlanmıştır. Hold komutunun kullanılmasıyla iki grafiğin tek bir ekseni kullanması sağlanmıştır. Grafik, x ekseni ve y ekseni başlıkları oluşturulmuştur. Son olarak text komutu kullanılarak grafik üzerinde istenilen noktalara yazı yazılması sağlanmıştır.

4.1.2. Bar, Barh

Bar grafiklerini kullanarak grafik çiziminde kullanılan komuttur. **Bar(x)** ya da **Bar(x,y)** şeklinde kullanılır. Ayrıca yatay olarak kullanılmasını sağlayan **Barh(x)** ve **Barh(x,y)** komutları da mevcuttur.

Şekil 4.11. de Bar ve BarH fonksiyonlarının kullanımı görünmektedir.

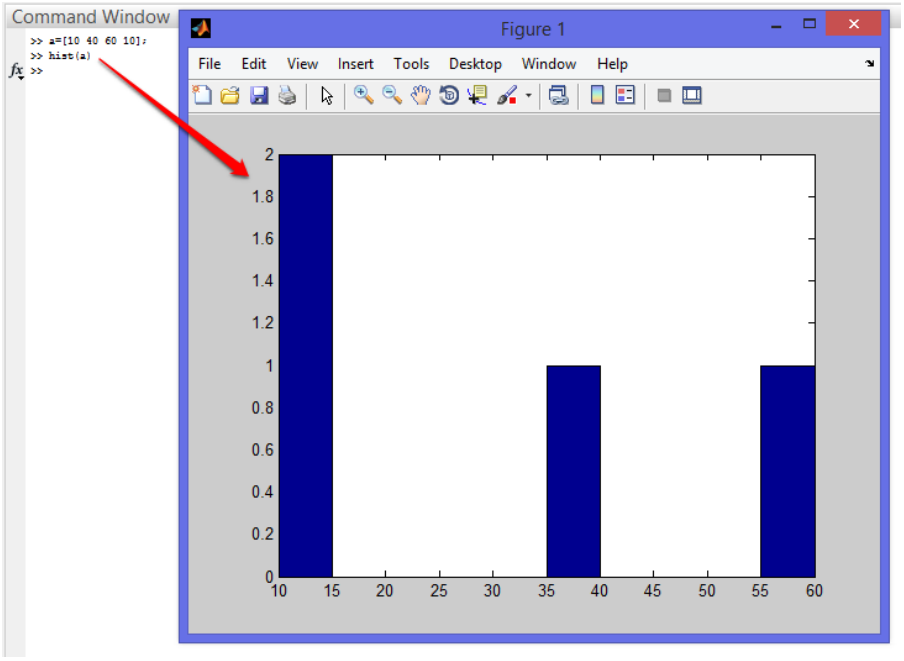


Şekil 4.11. Bar ve Barh Fonksiyonları

Şekil 4.11. de öncelikle a vektörü oluşturulmuş daha sonra bu vektörün bar(a) komutuyla dikey, barh(a) komutuyla yatay bar grafikleri çizdirilmiştir.

4.1.3. Hist

Histogram grafiği çiziminde kullanılan komuttur. **Hist(x)** şeklinde kullanılır. Şekil 4.12. de hist fonksiyonun kullanımı görünmektedir.



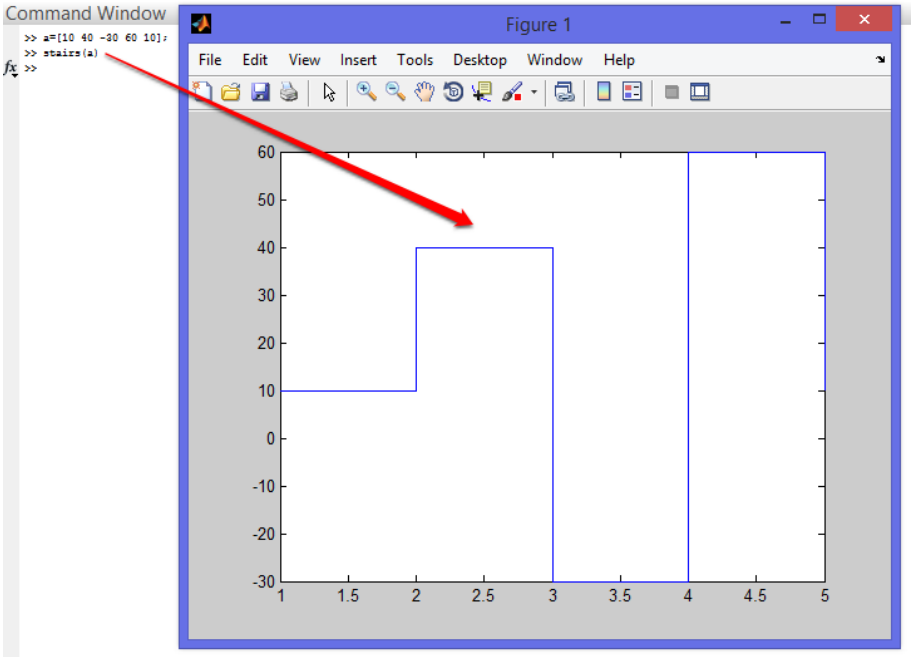
Şekil 4.12. Hist Fonksiyonu

Şekil 4.12. de öncelikle a vektörü yaratılmış, daha sonra bu vektörün hist(a) komutuyla histogram(sıklık) grafiği çizdirilmiştir.

4.1.4. Stairs

Basamak şeklinde grafik çiziminde kullanılan komuttur. **Stairs(x)** ya da **Stairs(x,y)** şeklinde kullanılır.

Şekil 4.13. de stairs fonksiyonun kullanımı görünmektedir.



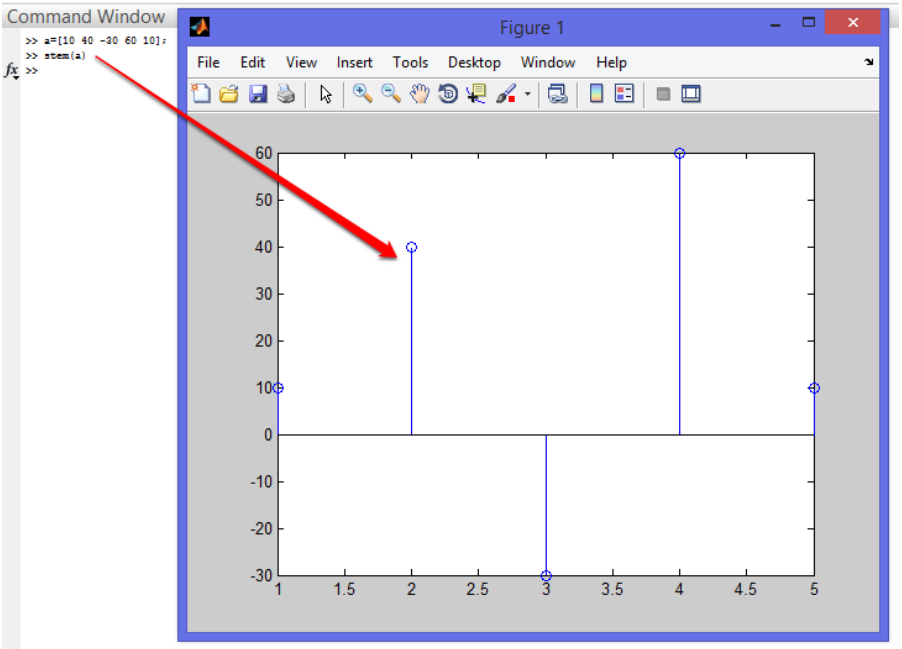
Şekil 4.13. Stairs Fonksiyonu

Şekil 4.13. de öncelikle a vektörü yaratılmış, daha sonra bu vektörün stairs(a) komutuyla basamak grafiği çizdirilmiştir.

4.1.5. Stem

Nokta şeklinde grafik çiziminde kullanılan komuttur. **Stem(x)** ya da **Stem(x,y)** şeklinde kullanılır.

Şekil 4.14. de stem fonksiyonun kullanımı görünmektedir.



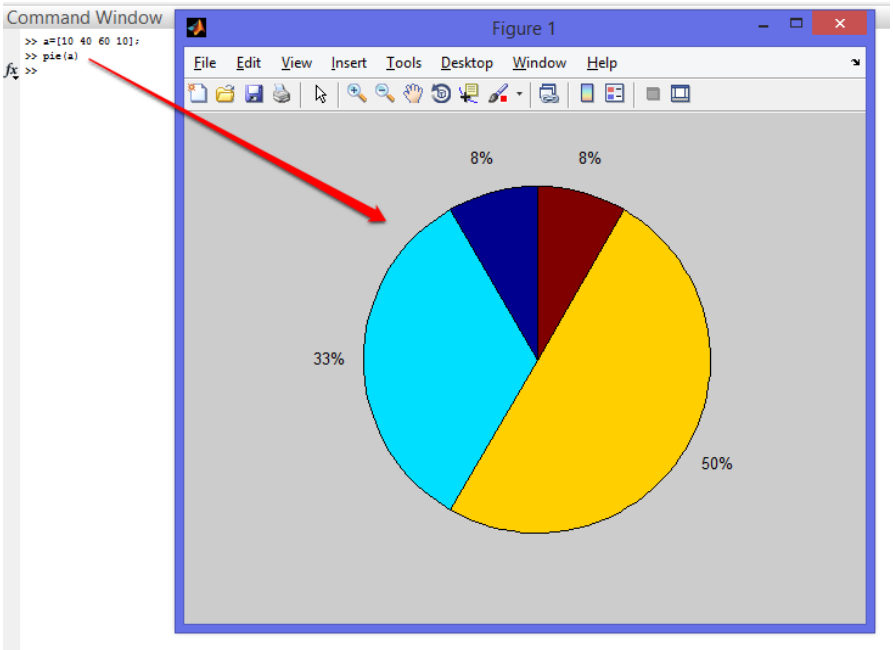
Şekil 4.14. Stem Fonksiyonu

Şekil 4.14. de öncelikle a vektörü yaratılmış, daha sonra bu vektörün stem(a) komutuyla nokta grafiği çizdirilmiştir.

4.1.6. Pie

Yüzdelik dilim şeklinde grafik çiziminde kullanılan komuttur. **Pie(x)** şeklinde kullanılır. Ancak bu şekilde grafiği çizilecek vektörde negatif değerli eleman bulunmamalıdır.

Şekil 4.15. de pie fonksiyonun kullanımı görünmektedir.



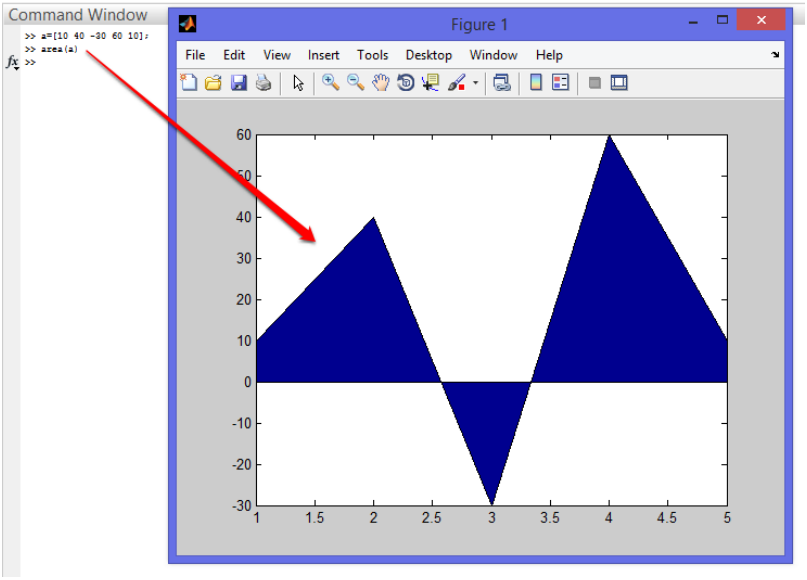
Şekil 4.15. Pie Fonksiyonu

Şekil 4.15. de öncelikle a vektörü yaratılmış, daha sonra bu vektörün pie(a) komutuyla pasta grafiği çizdirilmiştir.

4.1.7. Area

Alan grafiği çiziminde kullanılan komuttur. **Area(x)** ya da **Area(x,y)** şeklinde kullanılır.

Şekil 4.16. da area fonksiyonun kullanımı görünmektedir.



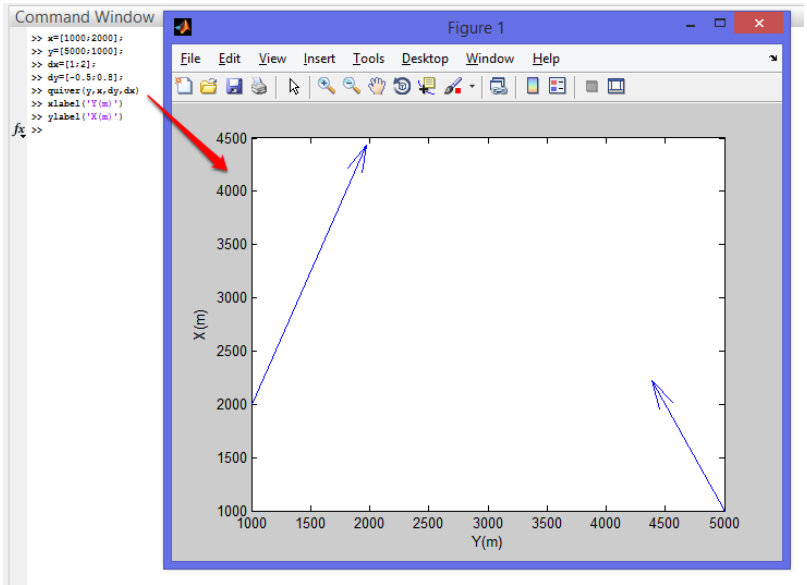
Şekil 4.16. Area Fonksiyonu

Şekil 4.16. da öncelikle a vektörü yaratılmış, daha sonra bu vektörün area(a) komutuyla pasta grafiği çizdirilmiştir.

4.1.8. Quiver

x ve y koordinat değerlerine sahip bir noktanın dx ve dy kadar yer değiştirdiği düşünülsün. Bu noktadaki (dx,dy) vektörünü çizdirmek istenildiğinde, quiver fonksiyonu kullanılır. **quiver(y,x,dy,dx)** komutu ile bir jeodezik dik koordinat sisteminde vektör çizimi gerçekleştirilir. Vektörleri ölçeklendirmek için, s ölçek faktörü quiver fonksiyonuna beşinci bir değişken olarak eklenmelidir; **quiver(y,x,dy,dx,s)**

Şekil 4.17. de quivery fonksiyonun kullanımı görünmektedir.



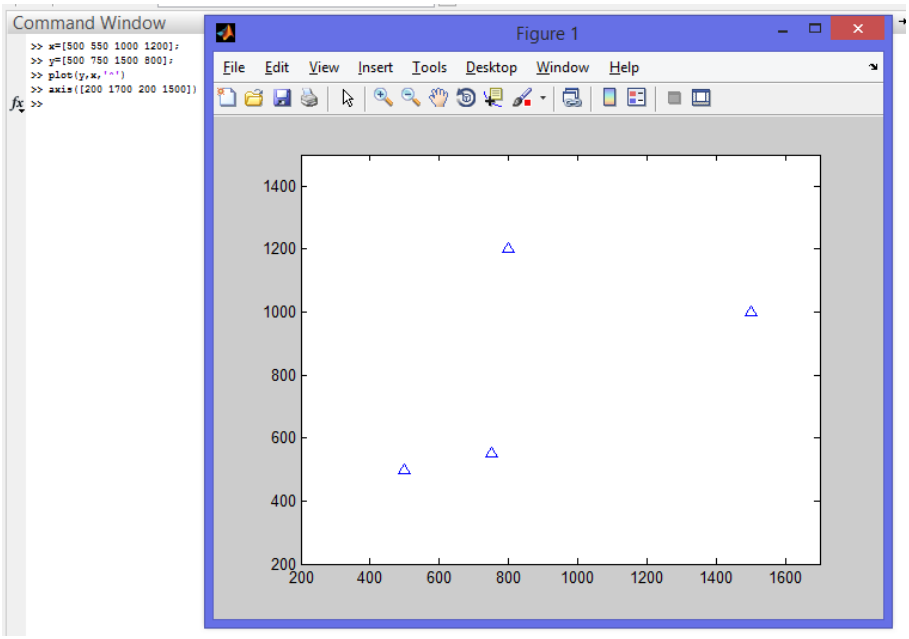
Şekil 4.17. Quiver Fonksiyonu

Şekil 4.17. de öncelikle x, y, dx ve dy vektörleri yaratılmış, daha sonra bu vektörlerin quiver(y,x,dy,dx) komutuyla vektör grafikleri çizdirilmiştir.

4.1.9. Kanava

Kanava jeodezik çizimlerde kullanılan bir tekniktir. Plot fonksiyonu kullanılarak ekrana üçgen noktalar çizilerek oluşturulan bir grafik çeşididir.

Şekil 4.18. de kanava grafiğinin kullanımı görünmektedir.



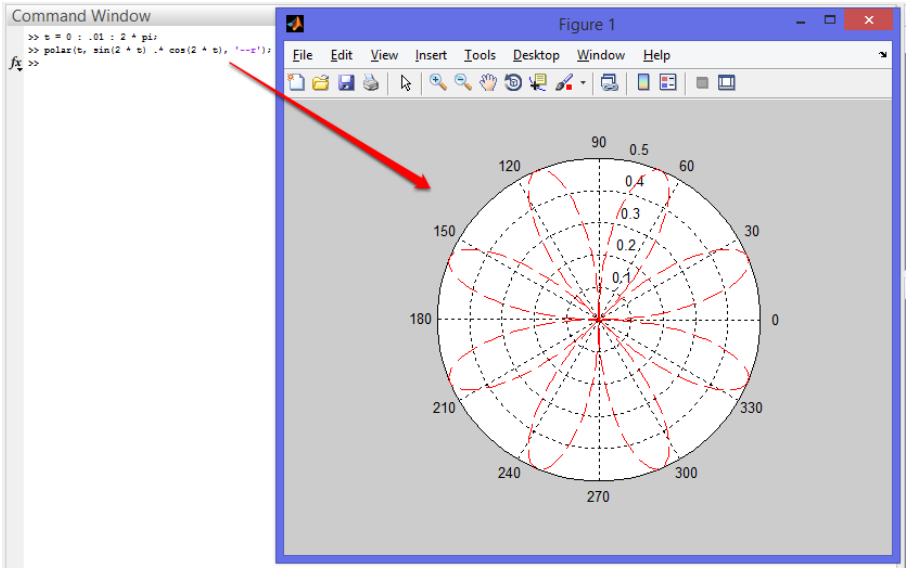
Şekil 4.18. Kanava Grafiği

Şekil 4.18. de öncelikle x ve y vektörleri yaratılmış, daha sonra bu vektörlerin plot(y,x) komutuyla kanava grafiği çizdirilmiştir.

4.1.10. Polar

Polar koordinatları kullanarak grafik çiziminde kullanılan komuttur.

Şekil 4.19. da polar fonksiyonun kullanımı görünmektedir.



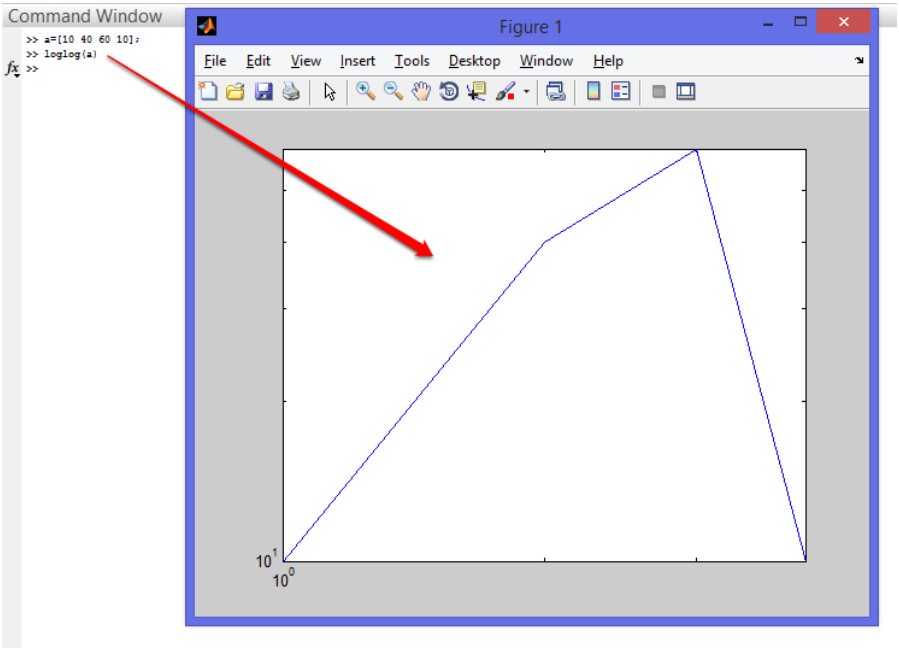
Şekil 4.19. Polar Fonksiyonu

Şekil 4.19. da öncelikle t vektörü yaratılmış, daha sonra polar komutuyla polar kordinat sisteminde grafiği çizdirilmiştir.

4.1.11. Loglog

X ve y eksenleri için logaritmik ölçeklendirme kullanarak grafik çiziminde kullanılan komuttur. **Loglog(x)** ya da **Loglog(x,y)** şeklinde kullanılır. Ancak bu şekilde grafiği çizilecek vektörde negatif değerli eleman bulunmamalıdır.

Şekil 4.20. de loglog fonksiyonun kullanımı görünmektedir.



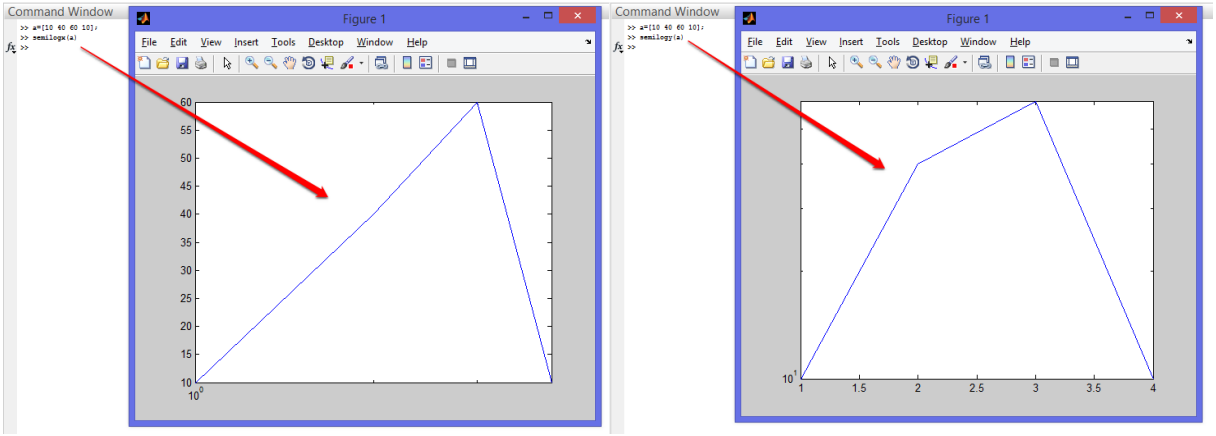
Şekil 4.20. Loglog Fonksiyonu

Şekil 4.20. de öncelikle a vektörü yaratılmış, daha sonra loglog(a) komutuyla logaritmik grafiği çizdirilmiştir.

4.1.12. Semilogx ya da Semilogy

Semilogx x eksenini logaritmik y eksenini lineer olarak ölçeklendirerek grafik çiziminde kullanılan komuttur. **Semilogx(x)** ya da **Semilogx(x,y)** şeklinde kullanılır. Semilogy ise y eksenini logaritmik x eksenini lineer olarak ölçeklendirerek grafik çiziminde kullanılan komuttur. **Semilogy(x)** ya da **Semilogy(x,y)** şeklinde kullanılır.

Şekil 4.21. de semilogx ve semilogy fonksiyonlarının kullanımı görünmektedir.



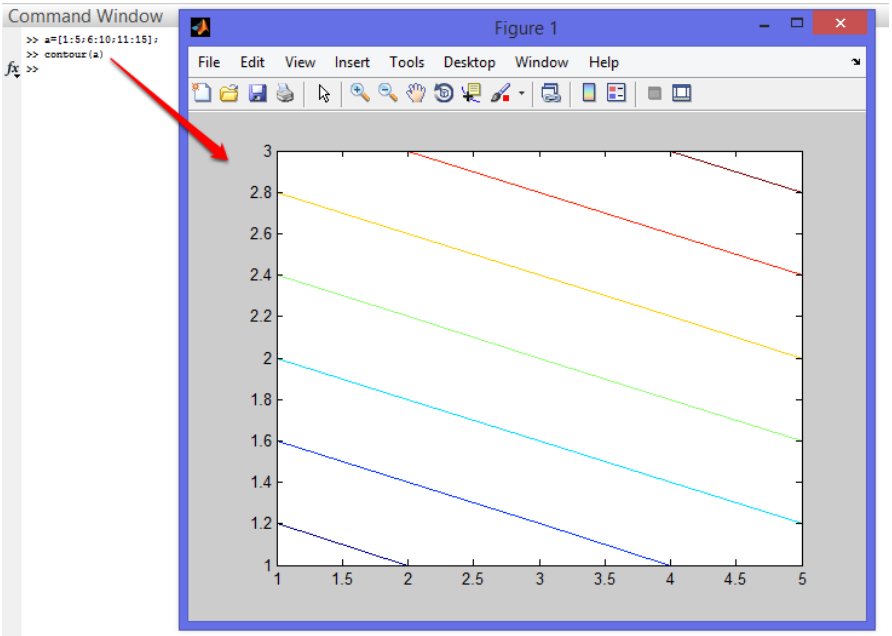
Şekil 4.21. Semilogx ve Semilogy Fonksiyonu

Şekil 4.21. de öncelikle a vektörü yaratılmış, daha sonra semilogx(a) komutuyla x eksenli logaritmik, semilogy(a) komutuyla y eksenli logaritmik grafikleri çizdirilmiştir.

4.1.13. Contour

Contour grafik çiziminde kullanılan komuttur. **Contour(x)** şeklinde kullanılır. Ancak X matris olmak zorundadır.

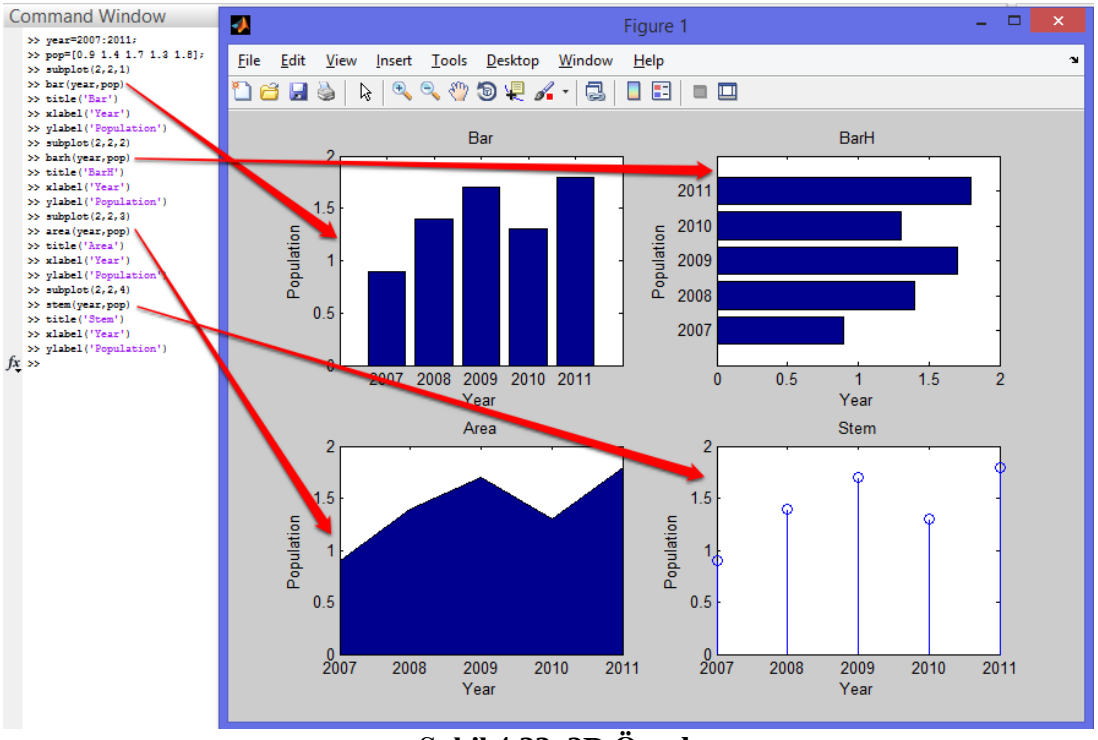
Şekil 4.22. de contour fonksiyonunun kullanımı görünmektedir.



Şekil 4.22. Contour Fonksiyonu

Şekil 4.22. de öncelikle a vektörü yaratılmış, daha sonra contour(a) komutuyla eş yükselti grafiği çizdirilmiştir.

Yukarıda bahsedilen 2D grafiklerin tamamında aynı vektör kullanılarak(negatif değer barındırmaması gereken ve polar koordinat düzlemini kullanan grafik türleri hariç) çizim yapılmıştır. Aşağıdaki örnekte ise yukarıda ki grafiklerin bazıları kullanılarak oluşturulan yıl – doğum oranı grafikleri görülmektedir.



Şekil 4.23. 2D Örnek

4.2. 3 Boyutlu(3D) Grafikler

MATLAB da birçok 3 boyutlu(3D) grafik çizim fonksiyonu bulunmaktadır. Bunlar;

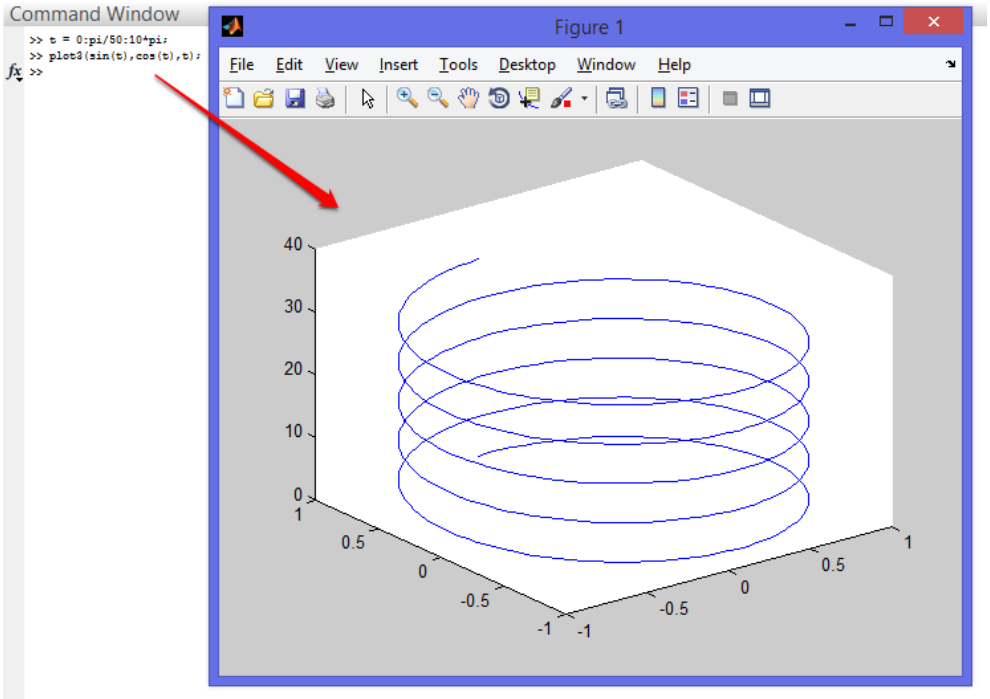
1) Plot3
2) Bar3, Bar3h
3) Pie
4) Mesh
5) Surface

3D grafiklere başlamadan önce bu grafiklerde kullanacağımız bir fonksiyondan bahsetmek istiyorum. **Meshgrid** fonksiyonu grafik çiziminde direkt olarak kullanılan bir komut değildir. Ancak 3 boyutlu çizimlerde verilerin oluşturulmasında kullanılmaktadır. Meshgrid ile elde edilen koordinat verileri [X,Y] matrisine aktarılır. Bu işleme interpolasyon denir. Daha sonra bu matrise bağlı olarak Z=f(X,Y)gibi bir matris elde ederiz. Z fonksiyonu uzayda bir yüzey belirtir. Z fonksiyonu matematiksel bir formülle hesaplanabileceği gibi MATLAB ın hazır fonksiyonlarıyla da elde edilebilir. Bu fonksiyonlar arasında **Peaks**, **Cylinder**,**Sphere** ve **Ellipsoid** i sayılabilir. Herhangi bir koordinat verilerinde tepe ve alt tepe değerlerini elde etmek istiyorsak **Peaks(X,Y)** komutu uygulanır. **Cylinder** dairesel, **Sphere** küresel, **Ellipsoid** ise elipsel sonuçlar üretir.

4.2.1. Plot3

x-y-z koordinat düzlemine çizmek için kullanılır. **Plot(x,y,z)** şeklinde kullanılır. Grafik çizgisinin özellikleri değiştirilmek istendiğinde **Plot(x,y,z's')** şeklinde de kullanılabilir. S parametresinde çizginin rengini stilini ya da biçimini belirtir. Bu özelliklerden sadece biri herhangi ikisi yada üçü birden s parametresinde ayarlanabilir. S paremetresinin değerleri yukarıda Tablo 4.1 de olduğu gibidir

Şekil 4.24. de plot3 fonksiyonunun kullanımı görünmektedir.



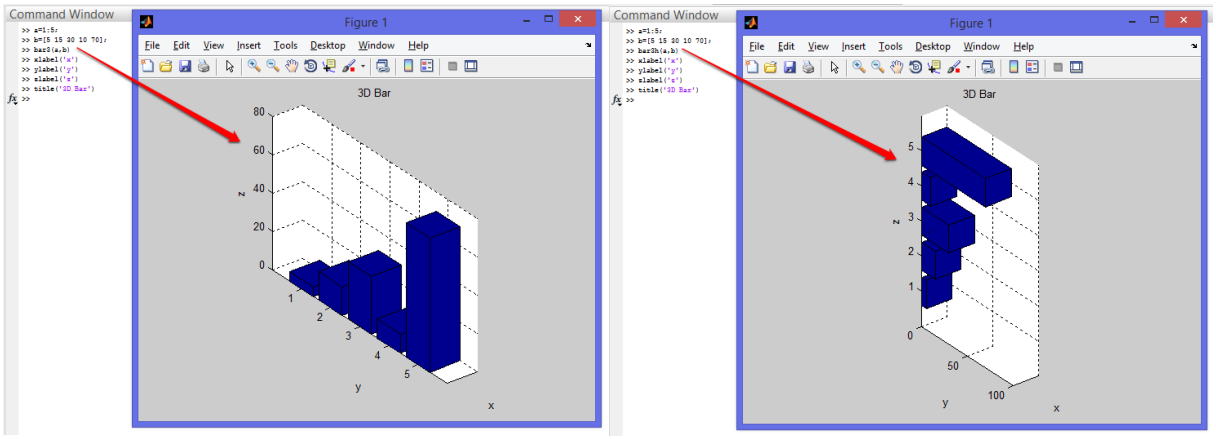
Şekil 4.24. Plot3 Fonksiyonu

Şekil 4.24. de öncelikle t vektörü yaratılmış, daha sonra plot3 komutuyla 3 boyutlu trigonometrik grafiği çizdirilmiştir.

4.2.2. Bar3, Bar3h

3 boyutlu bar grafiklerini kullanarak grafik çiziminde kullanılan komuttur. **Bar3(x,y)** şeklinde kullanılır. Ayrıca yatay olarak kullanılmasını sağlayan **Bar3h(x,y)** komutları da mevcuttur.

Şekil 4.25. de Bar3 ve Bar3H fonksiyonlarının kullanımı görünmektedir.



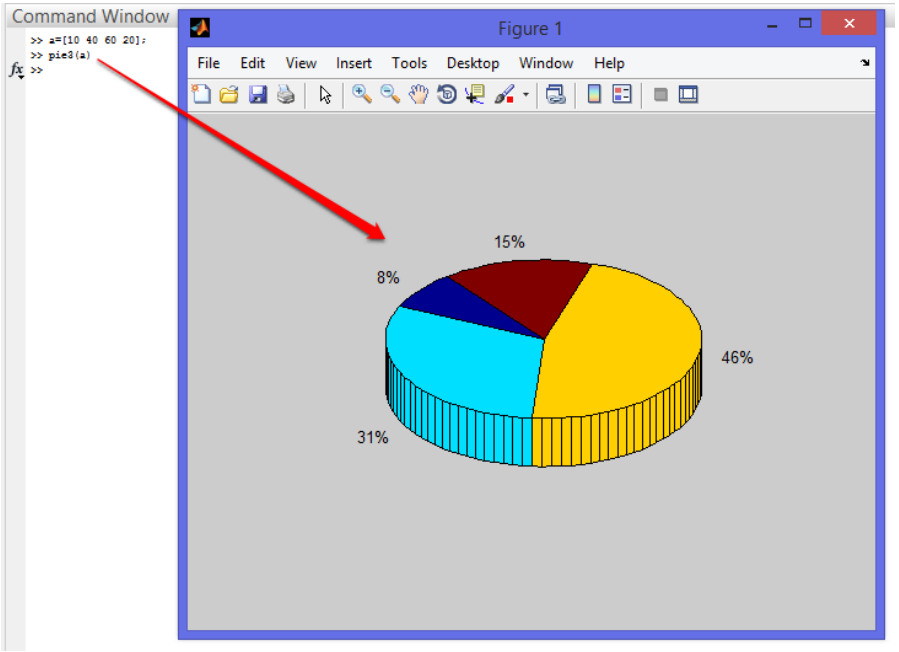
Şekil 4.25. Bar3 ve Bar3h Fonksiyonları

Şekil 4.25. de öncelikle a ve b vektörleri oluşturulmuş daha sonra bu vektörün bar3(a,b) komutuyla dikey, bar3h(a,b) komutuyla yatay bar grafikleri çizdirilmiştir.

4.2.3. Pie3

3 boyutlu yüzdelik dilim şeklinde grafik çiziminde kullanılan komuttur. **Pie3(x)** şeklinde kullanılır. Ancak bu şekilde grafiği çizilecek vektörde negatif değerli eleman bulunmamalıdır.

Şekil 4.26. da pie3 fonksiyonunun kullanımı görünmektedir.



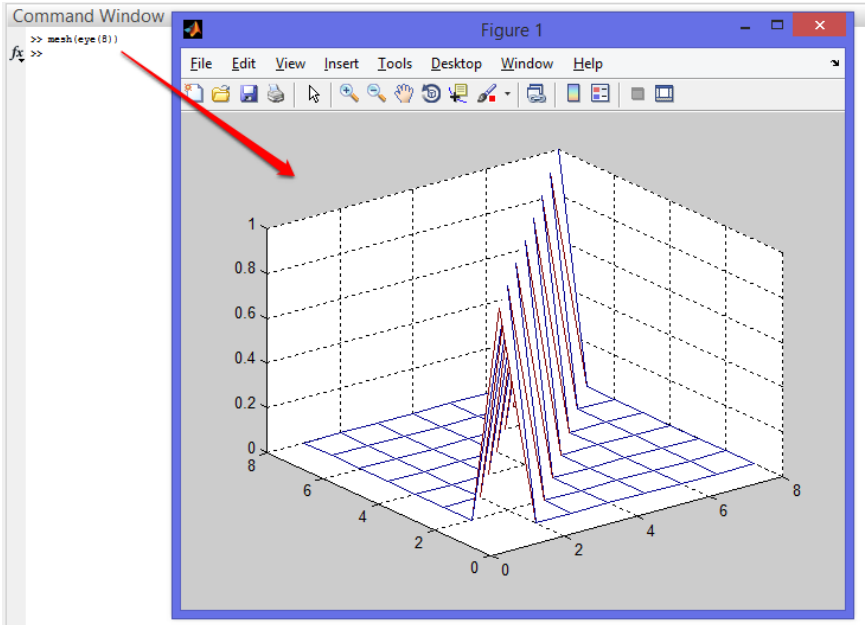
Şekil 4.26. Pie3 Fonksiyonu

Şekil 4.26. da öncelikle a vektörü oluşturulmuş daha sonra bu vektörün pie3(a) komutuyla 3 boyutlu pasta grafiği çizdirilmiştir.

4.2.4. Mesh

3 boyutlu ağ ve yüzey çizimlerinde kullanılan komuttur. **Mesh(x)** ya da **Mesh(a,b,c)** şeklinde kullanılır. X matris olmak zorundadır.

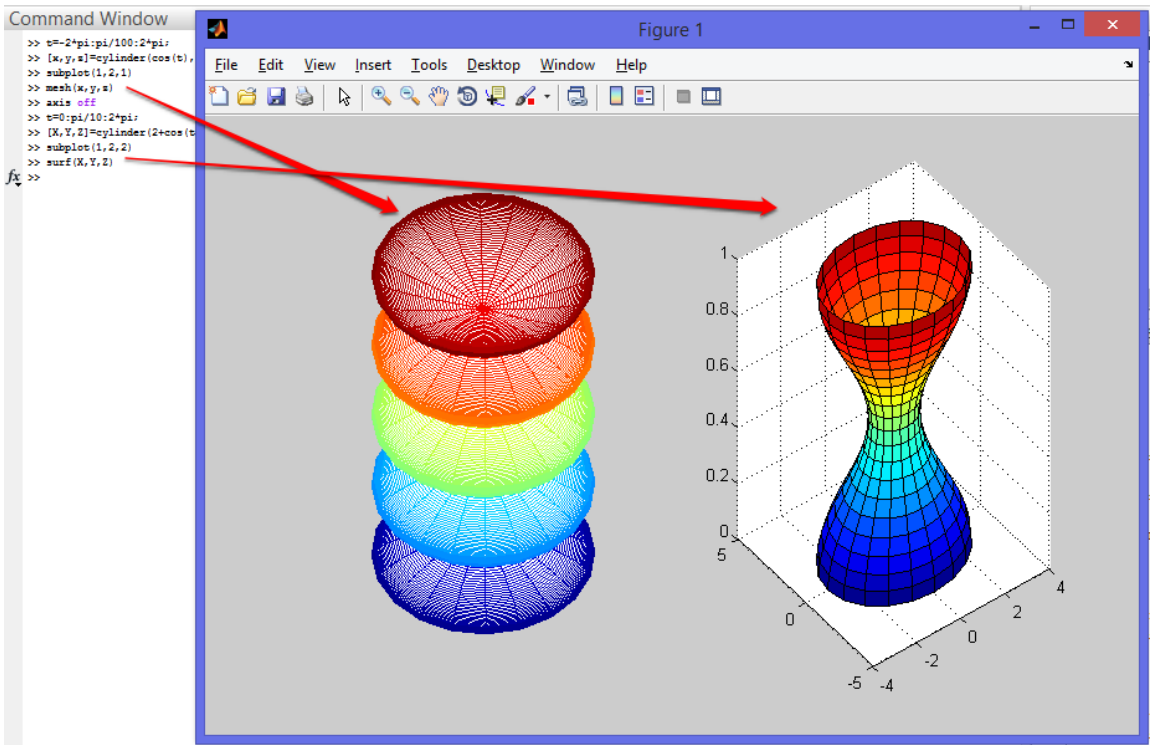
Şekil 4.27. de mesh fonksiyonunun kullanımı görünmektedir.



Şekil 4.27. Mesh Fonksiyonu

Şekil 4.27. de mesh(eye(8)) komutuyla 8 x 8 boyutundaki birim matrisin 3 boyutlu yüzey grafiği çizdirilmiştir.

Şekil 4.28. de mesh fonksiyonunun başka bir uygulaması görünmektedir.



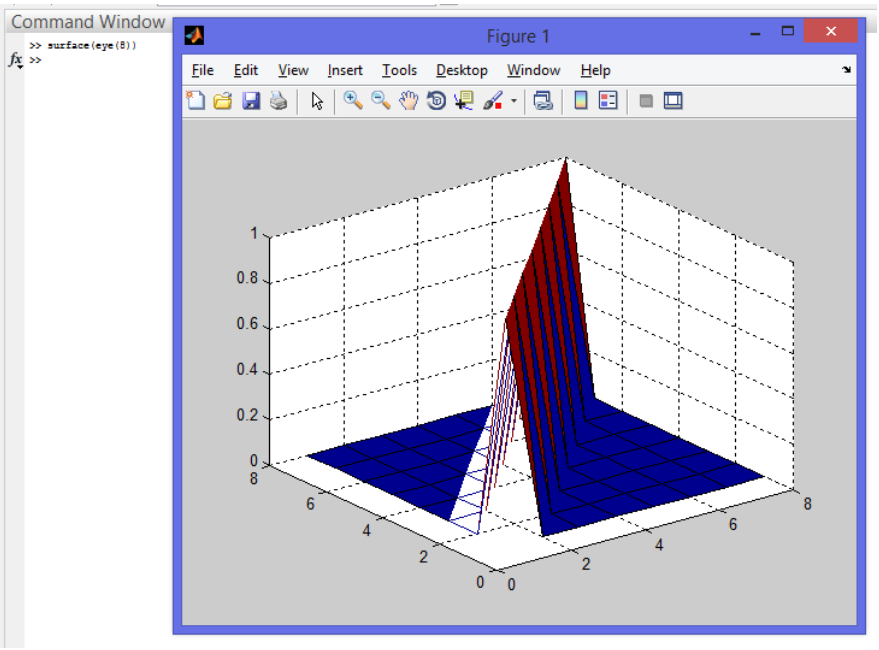
Şekil 4.28. Mesh Fonksiyonu

Şekil 4.28. de öncelikle t vektörü yaratılmış ardından subplot komutuyla figure penceresinde 2 grafiğin çizilmesi sağlanmıştır. Cylinder ile yaratılan silindir düzlemin 3 boyutlu yüzey grafikleri mesh komutuyla çizilmiştir.

4.2.5. Surf

3 boyutlu ağ ve yüzey çizimlerinde kullanılan komuttur. **Surf(x)** ya da **Surf(a,b,c)** şeklinde kullanılır. X matris olmak zorundadır. Mesh fonksiyonu ile aynı çıktıyı verir ek olarak yüzeyi boyar.

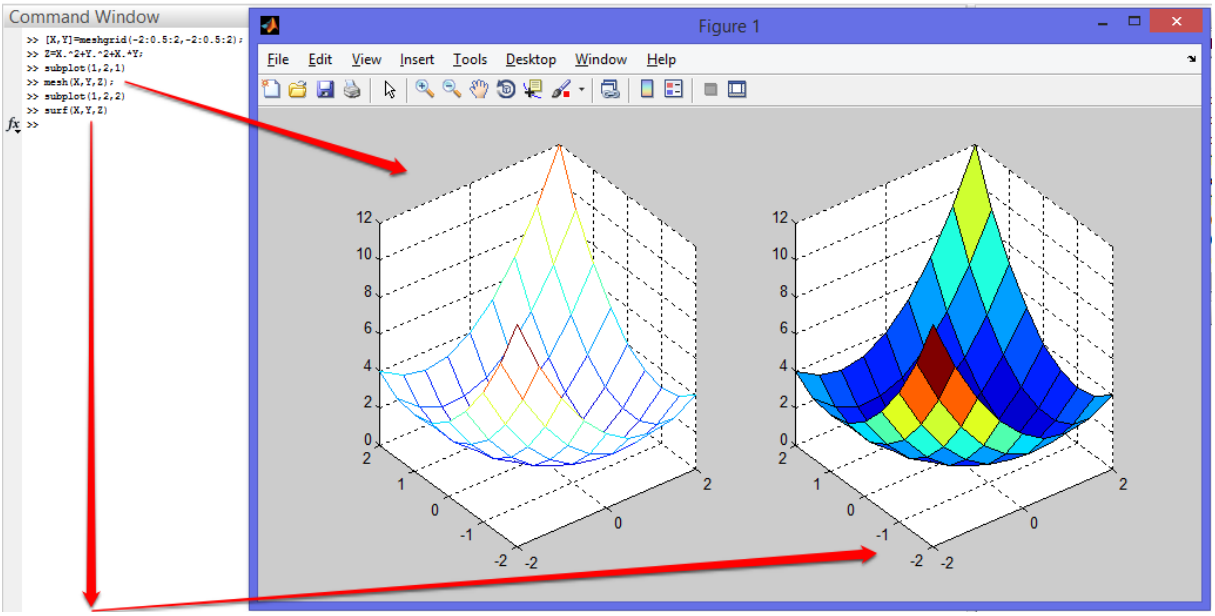
Şekil 4.29. da surf fonksiyonunun uygulaması görünmektedir.



Şekil 4.29. Surface Fonksiyonu

Şekil 4.29. da surface(eye(8)) komutuyla 8 x 8 boyutundaki birim matrisin 3 boyutlu yüzey grafiği çizdirilmiştir.

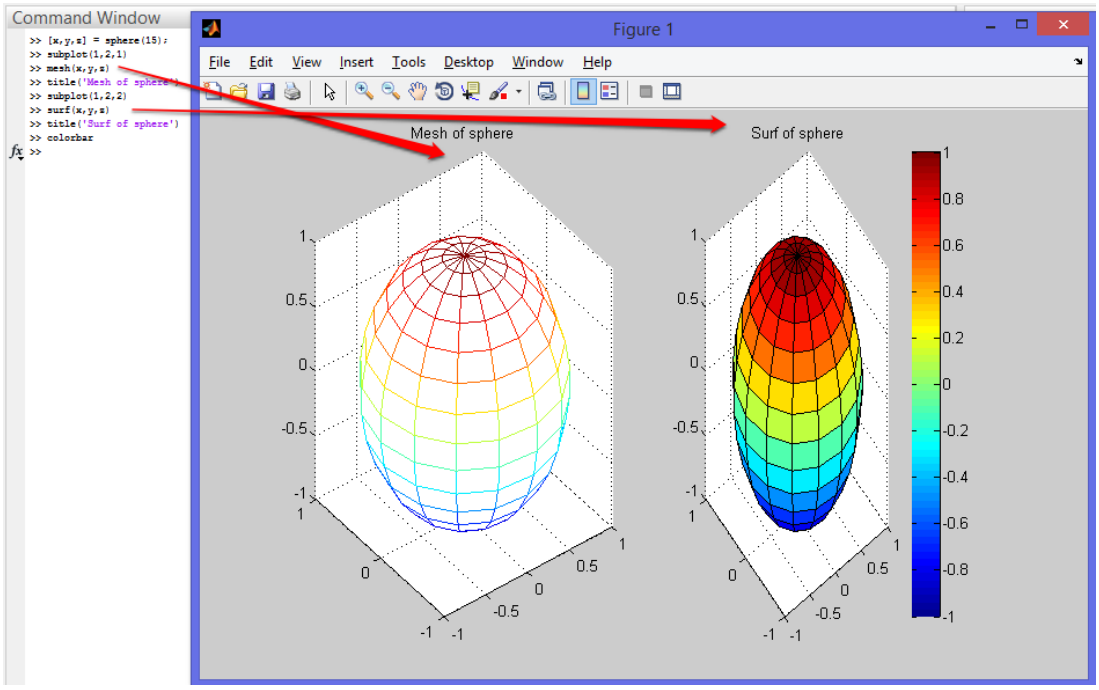
Şekil 4.30. da surf ve mesh fonksiyonlarının uygulamaları görünmektedir.



Şekil 4.30. Surf ve Mesh Fonksiyonu

Şekil 4.30. da öncelikle meshgrid komutu ile interpolasyonu yapılarak sonuç vektörünü elde etmek için X ve Y vektörleri elde edilmiştir. Daha sonra Z ile yüzey fonksiyonu oluşturulmuştur. Subplot komutu ile figure ekranı 2 ye bölündükten sonra aynı eğrinin önce mesh komutuyla daha sonra surf komutuyla 3 boyutlu yüzey grafikleri çizilmiştir.

Şekil 4.31. de surf ve mesh fonksiyonlarının başka bir uygulaması görünmektedir.



Şekil 4.31. Surf ve Mesh Fonksiyonu

Şekil 4.31. de sphere komutu ile direkt olarak grafik için gerekli olan x,y,z vektörleri oluşturulmuştur. Subplot komutu ile figure ekranı 2 ye bölündükten sonra aynı eğrinin önce mesh komutuyla daha sonra surf komutuyla 3 boyutlu yüzey grafikleri çizilmiştir. Son olarak colorbar komutuyla ekrana renk çubuğu eklenmiştir.

4.3. Ölçme ve Değerlendirme

Aşağıdaki soruları çözünüz.

1) f(x) = x2 + 5x – 3, g(x) = x2 + 3, h(x)= x olmak üzere birinci fonksiyonu kırmızı ikinci fonksiyonu yeşil ve üçüncü fonksiyo mavi renkli olarak tek eksen üzerinde 0 – 10 aralığında çiziniz. Çizgilerin üzerinde ait oldukları fonksiyonun harfi yazılsın (f, g ve h).
2) $f(x) = x.e^{(-x2-y2)}$ fonksiyonunun x için -2 – 2 , y için -2 – 3 aralığında eş yükselti (contour) grafiğin çiziniz.
3) $f(x) = x^2 – y^2$ fonksiyonun x, y için -2 – 2 aralığında yüzey (mesh) grafiğini çizdirin.
4) $z = x^3 – y^3 + \cos(xy)$ fonksiyonunu -4 – 4 aralığında 3D olarak çiziniz

Not: Soruların cevaplarını Ek2 de bulabilirsiniz.

5. MATLAB'DA PROGRAMLAMA

MATLAB programı sunduğu geniş komut ve fonksiyon yapısının yanında program yazmayı da destekler. Bölüm 3.2 de kısaca değinilen giriş çıkış komutlarının dışında koşul ve döngü komutları da mevcuttur. MATLAB da program yazarken kullanılan komutlar 3 ana grup altında toplanıp incelenebilir. Bunlar;

1) Giriş – çıkış deyimleri
2) Koşul deyimleri
3) Döngü deyimleri

MATLAB programında program yazarken açıklama satırlarının % işareti ile başlaması gerekir.

5.1. Giriş – Çıkış Deyimleri

Bölüm 3.2 de değinilen giriş – çıkış deyimlerini tekrar kısaca hatırlayalım. MATLAB da sık kullanılan giriş – çıkış deyimleri 3 adettir. Bunlar;

1) input
2) disp
3) fprintf
4) Mesaj Kutuları

Giriş çıkış komutlarında **\n** parametresi kullanılarak alt satıra geçişi sağlanabilir.

5.1.1. input

MATLAB da bilgi giriş komutu olarak **input** fonksiyonunu kullanılır. Kullanıcının girdiği bilgi **input** fonksiyonu ile okunduktan sonra bir değişkene aktarılabilir. String ifade girişi için 's' parametresi kullanılmak zorundadır. MATLAB programında değişkenleri tanıtmaya gerek yoktur.

Ör:

>> a=input('Birinci Sayıyı Giriniz');	>> ad=input('Adınızı Giriniz','s');
Birinci Sayıyı Giriniz: 5	Adınızı Giriniz: sinan
a=	ad=
5	sinan

5.1.2. Disp

Formatsız çıkış komutu olarak kullanılan komuttur. Disp komutunu kullanarak ekrana her seferinde sadece tek bir ifade yazdırılabilir.

Ör:

Disp('---------Iki sayinin toplamini bulan program----------'):

5.1.3. fprintf

Formatlı çıkış komutu olarak kullanılan komuttur. fprintf komutu ile kullanılabilecek ekran çıkış formatları arasında en çok kullanılanlar;

- %d: Tamsayı
- %f: Gerçek Sayı
- %c: Tek karakter
- %s: String

Ör:

```
>> fprintf('Yapılan işlemin sonucu=%d\n İşlem başarıyla tamamlanmıştır\n',32);
Yapılan işlemin sonucu 32
İşlem başarıyla tamamlanmıştır
>>
```

```
>> fprintf('%2.3f\n',1234.5678);
1234.567
>>
```

```
>> vec=2:4;
>>fprintf('%d\n',vec)
2
3
4
>>
```

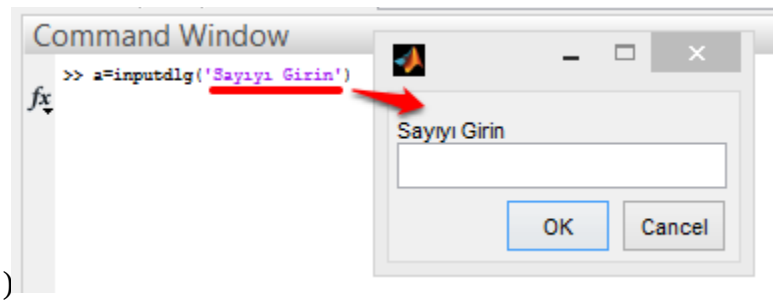
```
>> vec=2:4;
>>fprintf('%d',vec)
2 3 4 >>
```

5.1.4. Mesaj Kutuları

Mesaj kutuları da bir çeşit giriş – çıkış deyimi olarak görülebilir. MATLAB programında bir çok farklı türde mesaj kutusu formu mevcuttur. Aşağıda bunlardan bazılarının uygulaması gerçekleştirilmiştir.

1) msgbox('İçerik', 'Başlık')

2) İnputdlg('İçerik')



5.2. Koşul Deyimleri

Bütün programlama çözümlerinde en önemli husus programın koşullara göre dallanmasını sağlayabilmektir. Ancak bu sayede programın beklenen ihtimallere uygun cevap vermesi sağlanabilir. MATLAB da koşul deyimleri 3 adettir. Bunlar;

1) if else end
2) switch, case
3) try/catch

5.2.1. if

if(eğer) bir koşulun gerçekleşmesi durumunda işlemi yaptırmak için sıklıkla kullanılır. Tablo 5.1. de if komutunun farklı kullanım yapıları görünmektedir.

Tablo 5.1. if Komutunun Kullanım Yapısı

Komutun yapısı:		
if (koşul) işlem **end**	**if** (koşul) işlem **else** işlem **end**	**if** (koşul-1) işlem **else if** (koşul-2) işlem **else if** (koşul-3) işlem **else** işlem **end**

Tam bu noktada iki operatörü tekrar hatırlatmakta fayda var. Bunlardan birincisi koşul yazarken kullanılan ilişki operatörleri diğeri de eğer gerekirse koşulları birleştirirken kullanılan mantıksal operatörlerdir. Tablo 5.2. de İlişki ve mantıksal operatörler görünmektedir.

Tablo 5.2. İlişki ve Mantıksal Operatörler

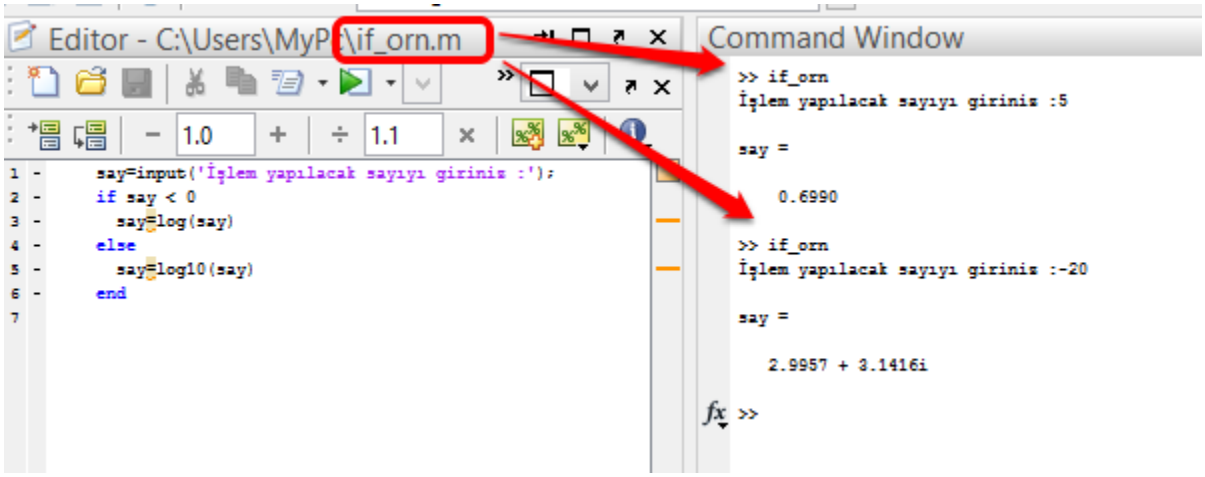
İlişki Operatörleri	
İşlem	Karşılığı
Eşit	==
Eşit Değil	~=
Küçük	<
Büyük	>
Küçük Eşit	<=
Büyük Eşit	>=

Mantıksal Operatörler	
İşlem	Karşılığı
Ve	&
Veya	|
Değil	~

Ör:

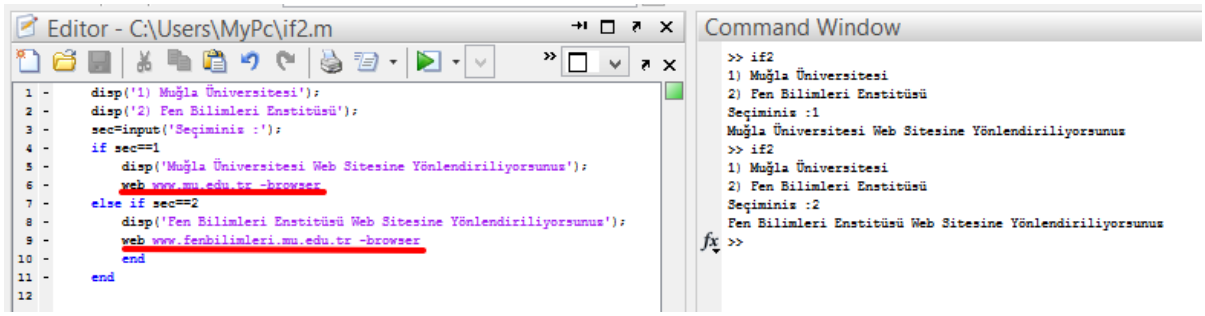
if ((x>5) & (y==3))	**if** (x<9)	**if** (x<4)
.....	....	.....
End	**else**	**else if** (y~=8)
	....	....
	end	**end**

Ör:Girilen sayı negetifse doğal logaritmasını, pozitifse 10 tabanında logoritmasını alan programı yazınız.



Yukarıda ki örnekte öncelikle if_orn script dosyası oluşturulmuştur. Bu dosyada kullanıcının girdiği sayı say değişkenine aktarılmış ve 0 dan küçükse log(say) komutuyla doğal logaritması 0 dan büyükse log10(say) komutu ile 10 tabanında logaritması alınmıştır. Dosya kaydedildikten sonra command window ekranında if_orn yazılarak çalıştırılmıştır.

Ör: Kullanıcı 1 girerse Muğla Üniversitesinin web sayfasını 2 girerse Muğla Üniversitesi Fen Bilimleri Enstitüsünün web sitesini açan programı yazın.



Yukarıda ki örnekte öncelikle if2 script dosyası oluşturulmuştur. Bu dosyada disp komutlarıyla önce kullanıcıya bilgi verilmiş ve bu doğrultuda kullanıcının yapacağı seçim sec değişkeninde tutulmuştur. Sec değişkeni 1 ise web www.mu.edu.tr – browser komutu ile Muğla Üniversitesinin web sayfasının, 2 ise web www.fenbilimleri.mu.edu.tr – browser komutu ile Fen Bilimleri Enstitüsünün web sayfasının açılması sağlanmıştır. Dosya kaydedildikten sonra command window ekranında if2 yazılarak çalıştırılmıştır.

5.2.2. switch case

Switch if yapısına benzer. Burada daha çok bir değişkenin sözel olarak belirtilen durumlara göre yönlendirme işlemi yapılır. Case ile beraber kullanılır. Tablo 5.3. de switch – case komutunun kullanım yapısı görünmektedir.

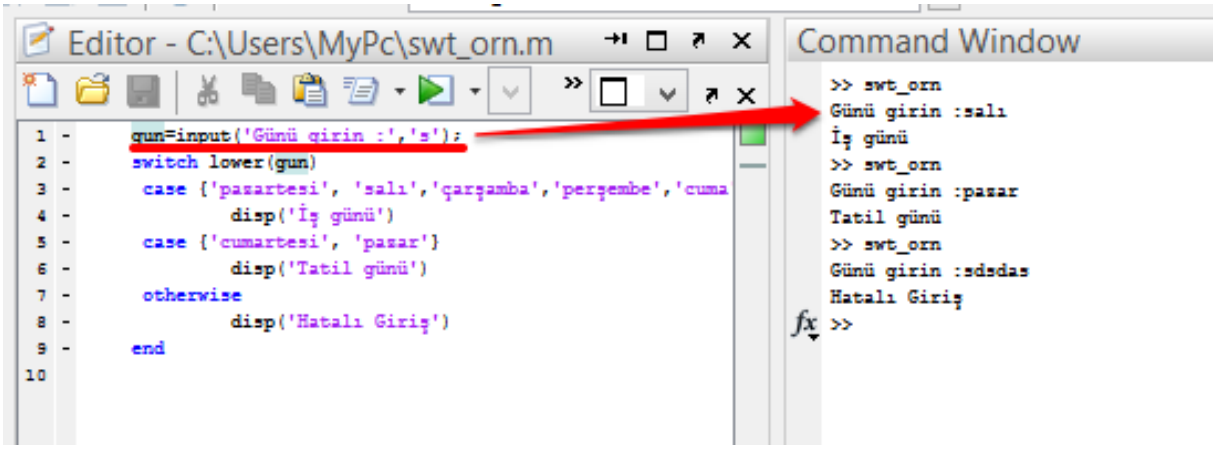
Tablo 5.3. switch – case Komutunun Kullanım Yapısı

Komutun yapısı:

switch değisken_adı
  case durum1
  İşlem1
  case durum2
  İşlem2
  case durum3
  İşlem3
  otherwise
  İşlem4
end

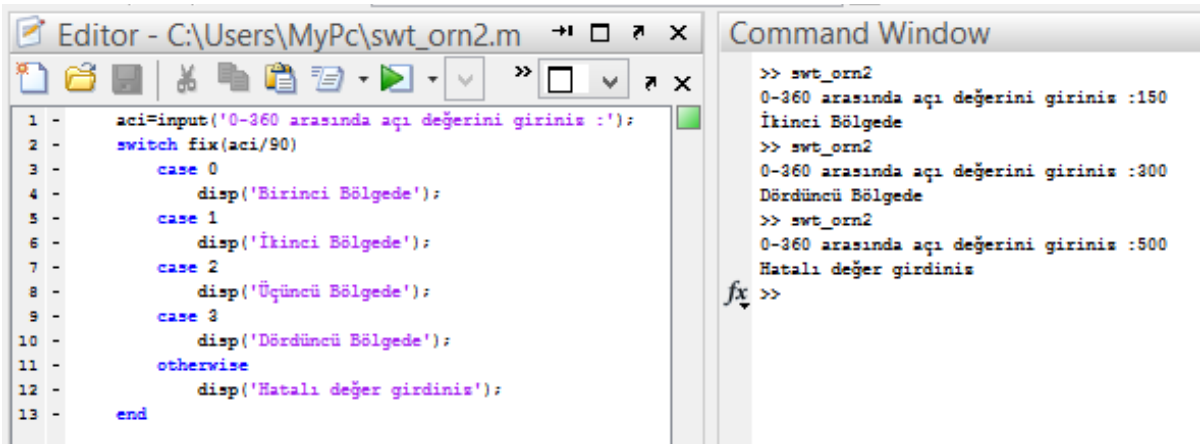
Eğer belirtilen durumların hiç biri koşulu sağlamıyorsa otherwise satırı devreye girer. Otherwise satırının kullanımı zorunlu değildir, kullanıcaya bağlıdır.

Ör:Girilen günün iş günümü yoksa tatil mi olduğunu ekrana yazan programı yazın.



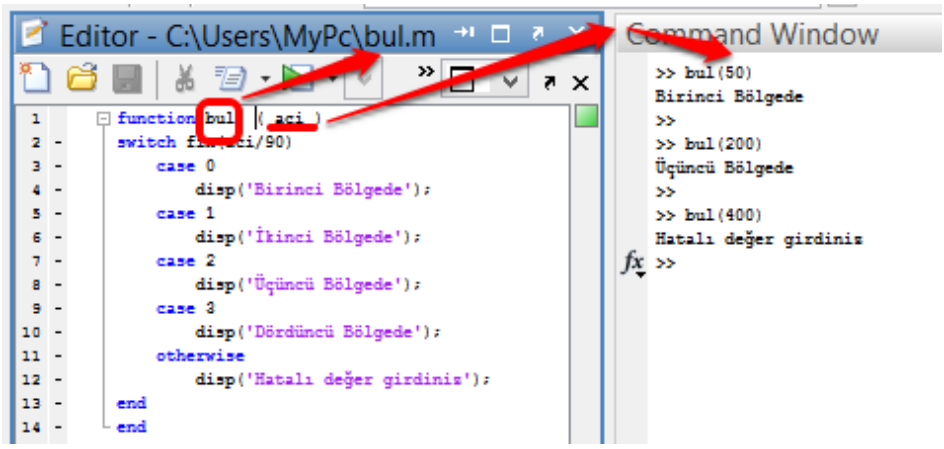
Yukarıda ki örnekte öncelikle swt_orn script dosyası oluşturulmuştur. Bu dosyada önce kullanıcının girdiği gün ismi gun değişkeninde tutulmuştur. Daha sonra switch – case komutuyla dallanma yapılarak duruma göre ekranda İş günü, Tatil günü yada Hatalı Giriş yazılması sağlanmıştır. Dosya kaydedildikten sonra command window ekranında swt_orn yazılarak çalıştırılmıştır.

Ör: Girilen açının kaçıncı bölgede olduğunu bulan programı yazın.



Yukarıda ki örnekte öncelikle swt_orn2 script dosyası oluşturulmuştur. Bu dosyada önce kullanıcının girdiği açı aci değişkeninde tutulmuştur. Girilen açı 90'a bölünüp fix komutuyla bölümün tam kısmı alındıktan sonra switch – case komutuyla dallanma yapılarak duruma göre ekranda Birinci Bölgede, İkinci Bölgede, Üçüncü Bölgede, Dördüncü Bölgede yada Hatalı değer girdiniz yazılması sağlanmıştır. Dosya kaydedildikten sonra command window ekranında swt_orn2 yazılarak çalıştırılmıştır.

Ör: Girilen açının kaçıncı bölgede olduğunu bulan programı function yapısını kullanara yazın.



Yukarıda ki örnekte öncelikle bul function dosyası oluşturulmuştur. Bir üstteki script örneğinden farklı olarak function çözümünde kullanıcının gireceği açı değeri input komutuyla değil functionın giriş parametresi olarak alınmıştır. Parametre olarak alınan açı 90'a bölünüp fix komutuyla bölümün tam kısmı alındıktan sonra switch – case komutuyla dallanma yapılarak duruma göre ekranda Birinci Bölgede, İkinci Bölgede, Üçüncü Bölgede, Dördüncü Bölgede yada Hatalı değer girdiniz yazılması sağlanmıştır. Dosya kaydedildikten sonra command window ekranında bul(deger) yazılarak çalıştırılmıştır.

5.3. Döngü Deyimleri

Programlama da tekrar eden durumlar için döngü yapıları kullanılır. Ancak unutulmaması gereken nokta tekrar sayılarının çok yüksek olmaması gerektiğidir. MATLAB programı hakkında dile getirilen en önemli eksikliklerden biri büyük döngülerde programın kasması yani bekleme süresinin uzun olmasıdır. MATLAB' da iki çeşit döngü komutu kullanılır. Bunlar;

1) for, end döngüsü
2) while, end döngüsü

Her iki döngüde de döngüyü tekrar sayısından daha önce sonlandırmak için **break** komutu kullanılır, yine her iki döngüde de döngüyü işlem yaptırmadan bir sonraki değere taşımak için **continue** komutu kullanılır.

5.2.1. for, end

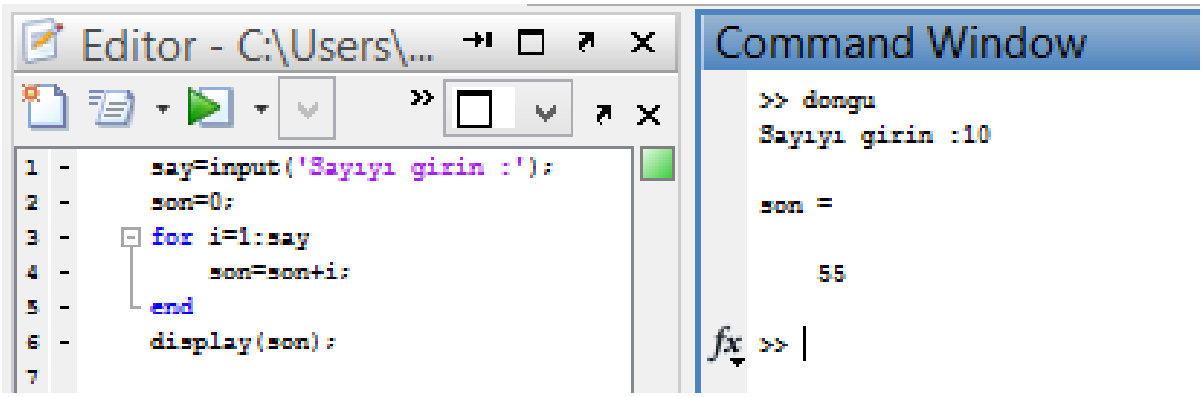
For, end döngüleri tekrar sayısı belli olan durumlarda kullanılan döngülerdir. Tablo 5.4. de for, end komutunun farklı kullanım yapıları görünmektedir.

Tablo 5.4. for, end Komutunun Kullanım Yapısı

Komutun yapısı:	
for döngü_değişkeni=bas_deg**:**bit_deg işlem **end**	**for** döngü_değişkeni=bas_deg**:**artış_mik**:**bit_deg işlem **end**

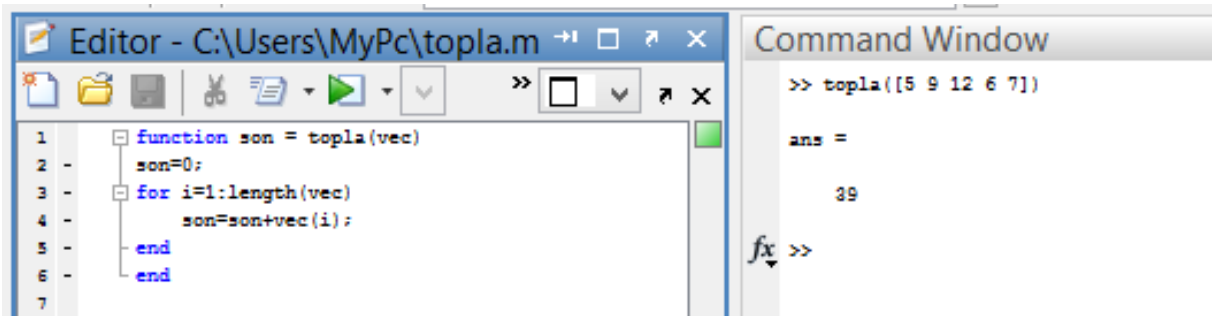
Döngü değişkeni integer tipinde tam sayı bir değişken olmak zorundadır.

Ör: Birden girilen sayıya kadar olan sayıların toplamını hesaplayıp yazdıran.



Yukarıda ki örnekte öncelikle dongu script dosyası oluşturulmuştur. Bu dosyada önce kullanıcının girdiği sayı say değişkeninde tutulmuştur. Birden girilen sayıya kadar bir döngü oluşturularak bu sayıların toplanması sağlanmıştır. Hesaplanan sonuç display komutuyla ekrana yazdırılmıştır. Dosya kaydedildikten sonra command window ekranında dongu yazılarak çalıştırılmıştır.

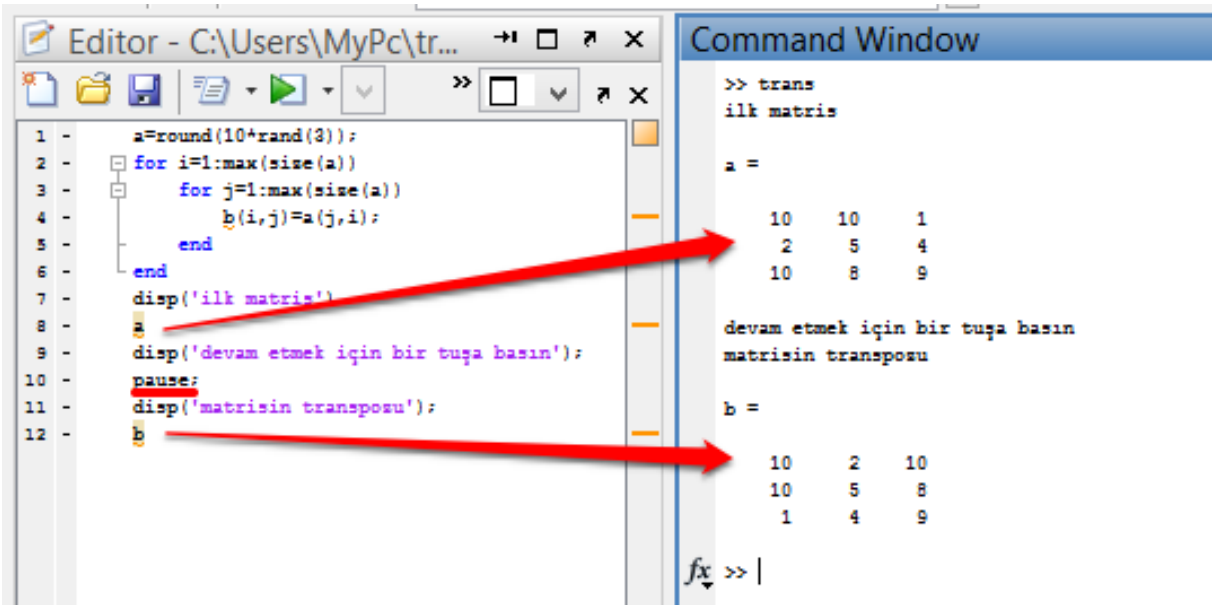
Ör: Bir vektörün elemanlarını toplayan programı function yapısında hesaplayın.



Yukarıda ki örnekte öncelikle topla function dosyası oluşturulmuştur. Elemanlarının toplanılması istenen vektör functionın giriş parametresi olarak alınmıştır. Birden vektörün eleman sayısı kadar bir döngü kurulduktan sonra, döngü değişkenini indis olarak kullanarak vektörün bütün elemanlarının okunması ve daha sonra toplanması sağlanmıştır. Bulunan sonuç çıkış parametresi olarak dışa aktarılmıştır. Dosya kaydedildikten sonra command window ekranında topla(vektör) yazılarak çalıştırılmıştır.

NOT: Yukarıda yapılan işlem sum(vektör) komutuyla da sağlanır.

Ör: 3x3 lük rastgele elemanlardan oluşan bir matris oluşturduktan sonra bunun transpozunu döngü kullanarak alan programı yazın.



Yukarıda ki örnekte öncelikle trans script dosyası oluşturulmuştur. Bu dosyada transpozu alınacak matrisin kullanıcı tarafından girilmesi yerine rastgele 3 x 3 boyutlarında bir matris oluşturulması tercih edilmiştir. İç içe döngü kurularak indisler üzerinden matrisin transpozu hesaplanmıştır. İlk oluşan matris ekrana yazdırılıp kullanıcıdan bir tuşa basması istenmiş ve daha sonra hesaplanan transpoz matrisi ekrana yazdırılmıştır. Dosya kaydedildikten sonra command window ekranında trans yazılarak çalıştırılmıştır.

Not: Yukarıda yapılan işlem b=a' komutuyla da sağlanır.

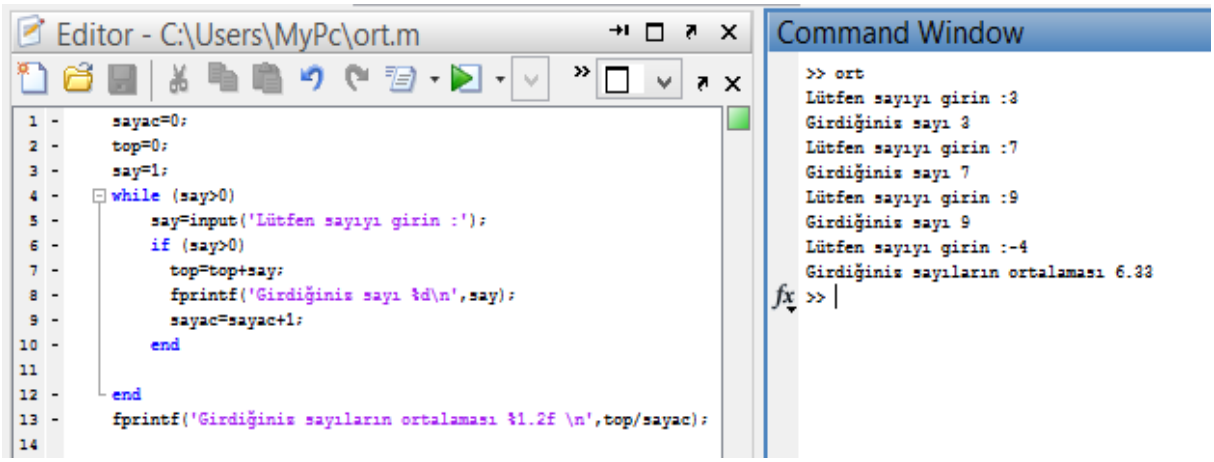
5.2.2. while, end

while, end döngüleri tekrar sayısı belli olmayan durumlarda kullanılan döngülerdir. Döngümüz bir koşula bağlı olarak çalışır ve koşul sağlandığı sürece çalışmaya devam eder. Koşul sağlanmadığında döngü sona erer. While döngülerinde döngünün ilk sefer çalışması için döngüden önce koşul da kullandığımız değişkene koşulu sağlayan bir öndeğer atamalıyız. Tablo 5.5. de while, end komutunun kullanım yapısı görünmektedir.

Tablo 5.5. for, end Komutunun Kullanım Yapısı

Komutun yapısı:
while (koşul) işlem **end**

Ör: Negatif sayı girilene kadar girilen pozitif sayıların ortalamasını hesaplayan programı yazın.



Yukarıda ki örnekte öncelikle ort script dosyası oluşturulmuştur. Bu dosyada ilk olarak programda kullanılacak yardımcı değişkenler olan sayac(girilen sayı adetini tutacak değişken) ve top(girilen sayıların toplamını tutan değişken) değişkenlerine 0 değeri atanmıştır hem girilen sayıyı tutan hemde döngü koşulunda kullanılan say değişkenine döngünün çalışmaya başlaması için while koşulunu sağlayan bir ilk değer atanmıştır. Döngü içinde önce sayı girilmiş sonra 0 dan büyük olması durumunda hem önceki sayılarla toplanmış hem de ortalama hesabında gerekli olan girilen sayı adetinin tespiti için sayac değişkeni bir arttırılmıştır. Ayrıca girilen sayının ekrana tekrar yazılması sağlanmıştır. Döngü bittikten sonra girilen sayıların ortalaması sayıların toplamının sayı adetine bulunmasıyla hesaplanıp yazdırılmıştır. Dosya kaydedildikten sonra command window ekranında ort yazılarak çalıştırılmıştır.

5.3. Ölçme ve Değerlendirme

Aşağıdaki soruları çözünüz.

1) Bir polinomun türev katsayılarını veren fonksiyonu yazınız.
2) Bir vektörün en küçük elemanının değerini bulan fonksiyonu yazınız.
3) Kullanıcının bir menü yardımıyla çorba, anayemek yada tatlı sipariş edebileceği ve seçiminin ekranda yazacağı scripti switch – case yapısını kullanarak yazınız.
4) Girilen açının sinusunu hesaplayan scripti yazınız. Açı derece yada radyan cinsinden girilebilir. Kullanıcı açı ya da derecenin sayısal bilgisini girdikten sonra hemen peşine derece için "d" radyan için "r" ibaresini girmek zorundadır (60d 3.1r gibi). Açının radyan ya da derece olduğunun belirtilmemesi durumunda belirtilene kadar girişin tekrar yapılmasını while, end döngüsüyle sağlayınız.

Not: Soruların cevaplarını Ek2 de bulabilirsiniz.

6. POLİNOMLAR

Polinomlar genellikle tek değişkenli ve sabit katsayılı fonksiyonlardır. Bu fonksiyonların en genel hali;

$$a_n x^n + a_{n-1} x^{n-1} + ... + a_1 x \ + a_0 = 0$$

Denklemde verilen eşitliğin kökleri gerçek veya karmaşık sayı olabilir.

MATLAB da polinomlar bir vektörle temsil edilirler. Polinom oluşturmak için yüksekten düşük dereceliye doğru azalan sırada polinom katsayıları yazılır.

Yukarıda genel ifadeyle yazılan polinoma bu işlem uygulandığında;

$$k = [a_n, a_{n-1}, ..., a_1, a_0]$$

6.1. Polinomun Kökleri

Polinomu temsil eden vektörü **roots(p)** komutuna tabi tutarak polinomun kökleri hesaplanır. Başka bir yöntem ise fonksiyonun tamamını **solve(fun)** komutuna tabi tutmaktır.

Solve ve benzeri direkt olarak fonksiyonun yazılarak çözüme ulaşalan komutlarda ya fonksiyon '...' arasına yazılır ya da komutdan hemen önce fonksiyonda kullanılan harfleri(x,y vb) **syms** fonksiyonu ile sisteme tanıtılır. Aksi halde hata mesajıyla karşılaşılır.

Ör: $2x^2 + 5x - 3$ denkleminin köklerini hesaplayınız.

P=[2 5 -3]

```
Command Window
>> p=[2 5 -3];
>> roots(p)

ans =

   -3.0000
    0.5000

fx >>
```

Yukarıda ki örnekte öncelikle polinomu temsil eden p vektörü oluşturulmuş daha sonra roots(p) komutuyla polinomun kökleri bulunmuştur.

```
Command Window
>> solve('2*x^2+5*x-3')

ans =

  -3
 1/2

>> double(ans)

ans =

   -3.0000
    0.5000

fx >>
```

```
Command Window
>> solve(2*x^2+5*x-3)
Undefined function or variable 'x'.

>> syms x
>> solve(2*x^2+5*x-3)

ans =

  -3
 1/2

fx >>
```

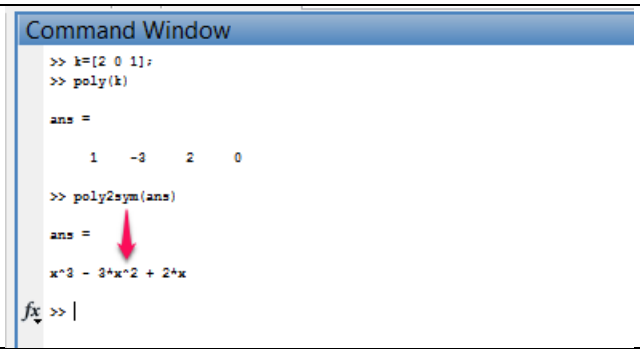
Yukarıda ki örnekte polinomun kökleri solve(fun) komutuyla hesaplanmıştır. Sol tarafta fonksiyon tırnak içinde yazılmış sağ tarafta ise önce x değişkeni tanımlanıp daha sonra fonksiyon direkt olarak parantez içinde yazılmıştır.

Not: double(x) fonksiyonu ile sembolik olarak bulunan polinom kökünün sayısal karşılığı hesaplanır.

6.2. Kökleri Bilinen Polinomun Katsayılarının Bulunması

MATLAB da kökleri bilnen bir polinomun katsayıları hesaplanabilir. Bunun için poly([$kök_1$, $kök_2$, $kök_3$...$kök_n$]) fonksiyonu uygulanır.

Ör: Kökleri 2, 0, 1 olan polinimu bulunuz.

K=[2 0 1]	

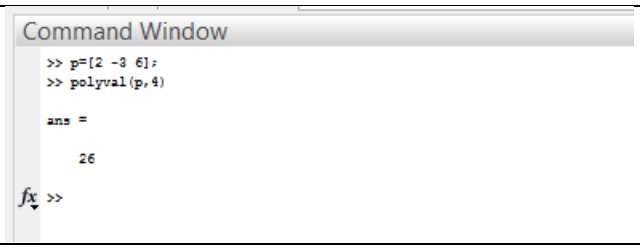
Yukarıda ki örnekte önce polinomun köklerinden oluşan k vektörü oluşturulmuş daha sonra poly(k) komutuyla polinomun katsayıları bulunmuştur.

Not: poly2sym(x) fonksiyonu ile polinomun sembolik bir şekilde görünmesi sağlanır.

6.3. Bir Polinomun Herhangi Bir Değer İçin Sonucunun Bulunması

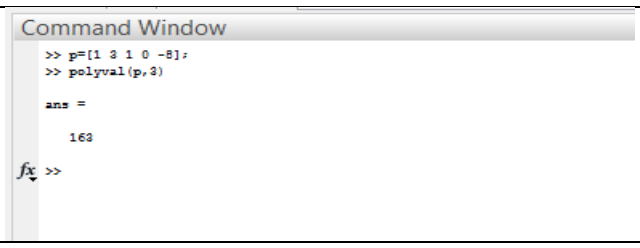
Bir polinomun herhangi bir değer için sonucunu hesaplamak için **polyval(p,değer)** fonksiyonunu uygularız.

Ör: $f(x)=2x^2 - 3x + 6$ ise $f(4)=?$

P=[2 -3 6]	

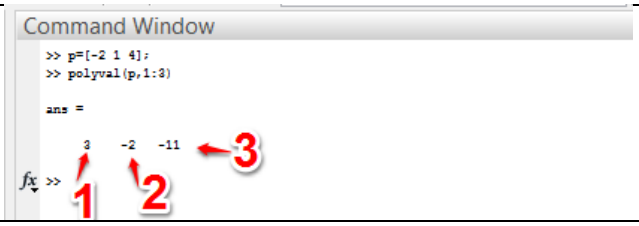
Yukarıda ki örnekte öncelikle polinomu temsil eden p vektörü oluşturulmuş daha sonra polyval(p,4) komutuyla polinomun 4 için sonucu hesaplanmıştır.

Ör: $f(x)=x^4 + 3x^3 + x^2 - 8$ ise $f(3)=?$

P=[1 3 1 0 -8]	

Yukarıda ki örnekte öncelikle polinomu temsil eden p vektörü oluşturulmuş daha sonra polyval(p,3) komutuyla polinomun 3 için sonucu hesaplanmıştır.

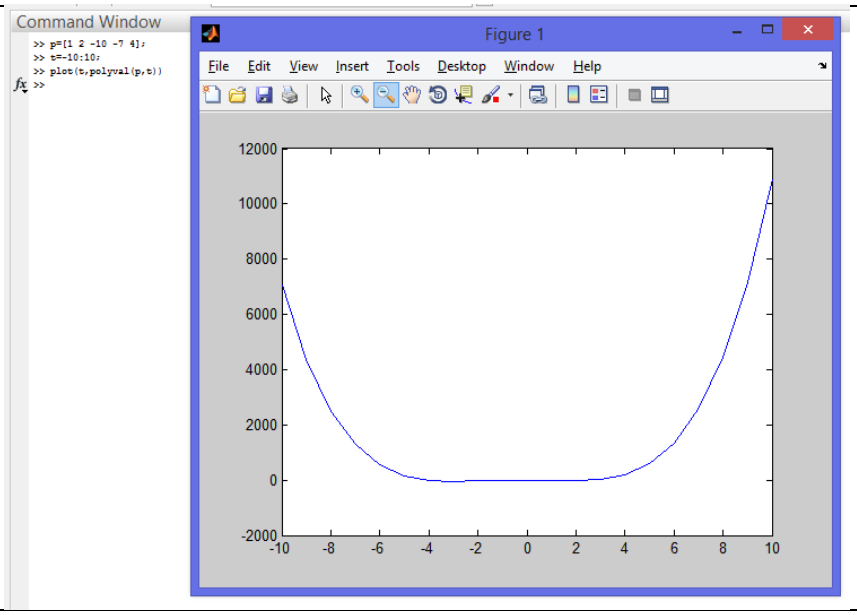
Ör: f(x)=-$2x^2$ + x + 4 ise f(1)=? f(2)=? f(3)=?

P=[-2 1 4]	

Yukarıda ki örnekte öncelikle polinomu temsil eden p vektörü oluşturulmuş daha sonra polyval(p,1:3) komutuyla polinomun 1, 2 ve 3 için sonuçları ayrı ayrı hesaplanmıştır.

Hesaplanan değerler üzerinden polinomun grafiği de çizilebilir.

Ör: f(x)=x^4 + $2x^3$ – $10x^2$ – 7x + 4 polinomunun -10 – 10 arasındaki grafiğini çiziniz.

p=[1 2 -10 -7 4] t=-10:10	

Yukarıda ki örnekte öncelikle polinomu temsil eden p vektörü, daha sonra grafiği çizilecek aralığın tanımlandığı t vektörü oluşturulmuştur. Son olarak plot komutuyla polinomun grafiği çizilmiştir.

Not: Bir polinomun grafiğini çizdirilmesi bölüm 4.1.1. de ayrıntılı olarak işlenmiştir.

6.4. Sonuçları ve Derecesi Bilinen Polinomun Katsayılarının Bulunması

MATLAB programında kökleri bilenen bir polinomun katsayılarının hesaplanabildiği gibi sonuç kümesi ve derecesi bilinen bir polinomun da katsayıları hesaplanabilir. Bunun için **polyfit([$x_{değerleri}$], [$y_{değerleri}$],derece)** fonksiyonu uygulanır.

Ör: x=[1, 3, 5, 6, 9, 12, 15], y=[4, 8, 15, 9, 12, 20, 24] olan 3. derece polinimu bulunuz.

X=[1 3 5 6 9 12 15]
Y=[4 8 15 9 12 20 24]

```
Command Window
>> x=[1,3,5,6,9,12,15];
>> y=[4,8,15,9,12,20,24];
>> polyfit(x,y,3)

ans =

    0.0138   -0.3082    3.0503    1.6163

>> pretty(poly2sym(ans))

                    3                    2
  7967699627078907 x    347041253266833 x    3434376007881833 x
  ------------------ - ----------------- + ----------------- + 7279247654042999/4503599627370496
  576460752303423488     1125899906842624     1125899906842624
fx >>
```

Yukarıda ki örnekte öncelikle verilen sonuç kümelerinin tanımlandığı x ve y vektörleri oluşturulmuştur. Daha sonra polyfit(x, y, 3) komutuyla bu sonuç kümelerinin sağlanabildiği 3. dereceden polinomun kökleri elde edilmiştir.

Not: poly2sym(x) fonksiyonu ile polinomun sembolik bir şekilde görünmesi sağlanır.
Not: pretty(x) fonksiyonu ile polinomun daha düzenli şekilde görünmesi sağlanır.

6.5. Polinomların Çarpımı ve Bölümü

İki polinomu çarpmak için **conv(f,g),** bölmek için **deconv(f,g)** fonksiyonları uygulanır.

Ör: f(x)=x + 8 g(x)= x^2 + 4x +8 ise f(x).g(x) ve f(x)/g(x) polinomlarını hesaplayınız.

f=[1 8]
g=[1 4 8]

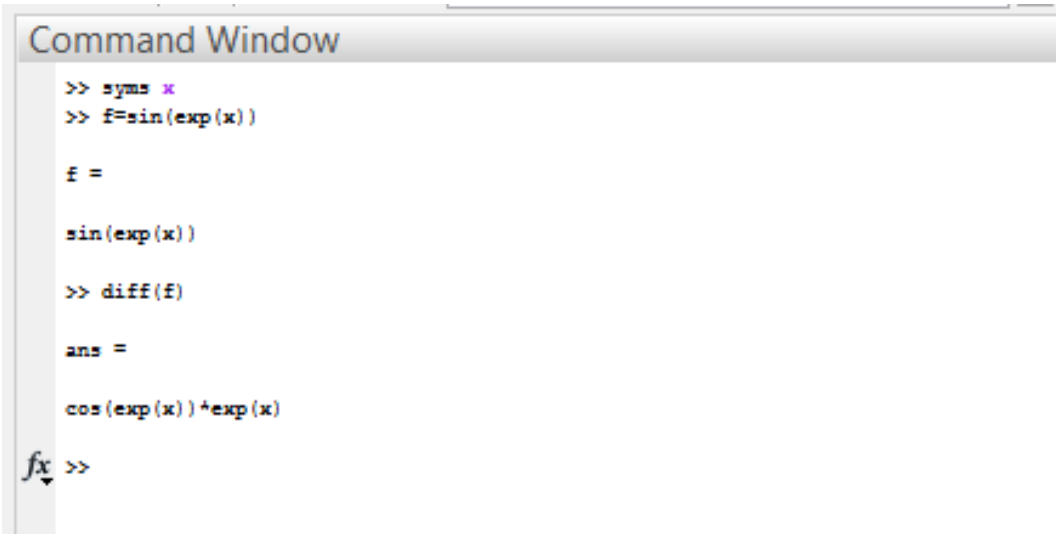
```
Command Window
>> f=[1 8];
>> g=[1 4 8];
>> conv(f,g)

ans =

     1    12    40    64

>> poly2sym(ans)

ans =

x^3 + 12*x^2 + 40*x + 64

>> deconv(g,f)

ans =

     1    -4

>> poly2sym(ans)

ans =

x - 4
fx >>
```

Yukarıda ki örnekte öncelikle f(x) polinomunu temsil eden p vektörü, daha sonra g(x) polinomunu temsil eden g vektörü oluşturulmuştur. Conv(f, g) komutuyla iki polinomun çarpıldığında ortaya çıkacak polinomun katsayıları hesaplanmıştır. Poly2sym(ans) komutuyla katsayıları görünen bu polinom sembolik olarak yazdırılmıştır. Deconv(g, f) komutuyla iki polinomun bölündüğünde ortaya çıkacak polinomun katsayıları hesaplanmıştır. Poly2sym(ans) komutuyla katsayıları görünen bu polinom sembolik olarak yazdırılmıştır.

6.6. Polinomlarda Türev

Türev matematiksel olarak bir f(x) fonksiyonunun x'e göre değişim oranı olarak tanımlanır. MATLAB' da bir polinomun sayısal türevi **polyder(katsayılar)** komutu ile alınır. Simgesel türevi ise **diff(x)** fonksiyonu ile hesaplanır.

Ör: f(x)=sin(e^x) fonksiyonunun simgesel türevini hesaplayınız.



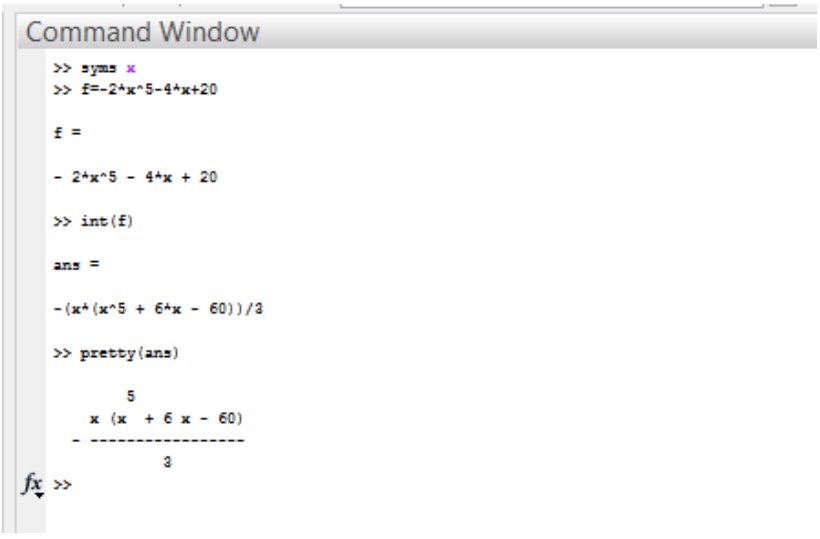
Yukarıda ki örnekte öncelikle syms x komutu ile x değişkeni, daha sonra f(x) polinomu f değişkeninde tanımlanmıştır. Diff(f) komutuyla polinomun simgesel türevi hesaplanmıştır.

Not: Simgesel bütün türev ve integral işlemleri için syms fonk. ile x'i tanımlamak zorundayız.

6.7. Polinomlarda İntegral

Simgesel integral almak için **int(f(x))** fonksiyonuyla **syms(x)** ile belirlenen simgesel değişkene göre f(x)'in benzersiz integrali alınır. Sayısal integral almak için ise **trapz(x,y), quad(f, xmin, xmax), dblquad(f, xmin, xmax, ymax, ymin)** fonksiyonları kullanılır.

Ör: $\int -2x^5 - 4x + 20$ integralini hesaplayınız.



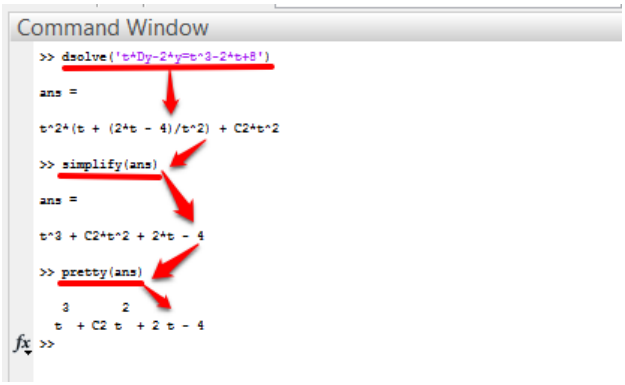
Yukarıda ki örnekte öncelikle syms x komutu ile x değişkeni, daha sonra f(x) polinomu f değişkeninde tanımlanmıştır. int(f) komutuyla polinomun simgesel türevi hesaplanmış pretty(ans) komutuyla düzenli basım şeklinde gösterilmiştir.

6.8. Polinomlarda Diferansiyel

Simgesel genel diferansiyel almak için **dsolve('dif_denk')** komutu kullanılır. Burada dikkat edilmesi gereken iki önemli husus vardır. Bunlardan birincisi ve en önemlisi diferansiye denklemi yazarken fonksiyon değişkeni olarak 'x' değil 't' kullanılması, ikincisi ise y' 1. mertebeden türev fonksiyonu için Dy, y'' 2. mertebeden türev fonksiyonu için D2y, y''' 3. mertebeden türev fonksiyonu için D3y... yazılmasıdır. Ayrıca **dsolve('dif_denk', 'özel_değer_1', 'özel_değer_2', ...)** komutuyla da belirtilen diferansiyel denklemin özel değerlerine karşılık gelen özel çözümleri bulunur.

Sayısal diferansiyel almak için ise **ode23(f, x_bas, x_bit,y_bas), ode45(f, x_bas, x_bit,y_bas)** adında iki adet Runge-Kutta fonksiyonu mevcuttur. Ode23 fonksiyonu ikinci ve üçüncü dereceden integrasyon denklemlerini, Ode45 ise dördüncü ve beşinci dereceden integrasyon denklemlerini kullanır.

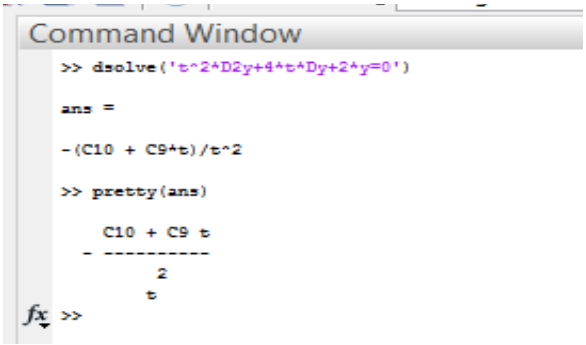
Ör: xy'-2y=x3 – 2x + 8 diferansiyel denkleminin genel çözümünü hesaplayınız.



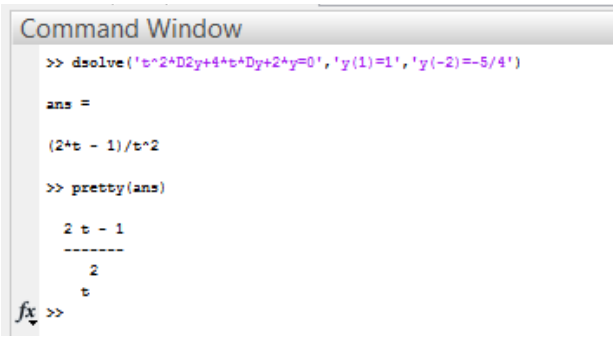
Yukarıda ki örnekte öncelikle diferansiyel denklem dsolve komutu ile oluşturulurken değişken olarak 't', y' ile 1. mertebeden türeve karşılık da Dy kullanılmıştır. Hesaplanan sonuç önce simplify(x) komutu ile sadeleştirilmiş, daha sonra pretty(x) komutu ile düzgün formatta yazılması sağlanmıştır.

Ör: x2y'' + 4xy' + 2y = 0 diferansiyel denkleminin

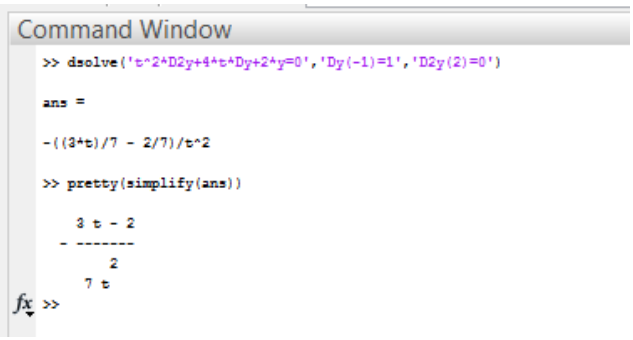
a) Genel çözümünü



b) X = 1 için y = 1 ve x= -2 için y = -5/4 değerini veren özel çözümünü hesaplayınız



c) X= -1 için y'=1 ve x= 2 için y'' = 0 değerini veren özel çözümünü hesaplayınız



6.9. Ölçme ve Değerlendirme

Aşağıdaki soruları çözünüz.

1) f(x) = $x^5 + 2x^4 - 5x^3 - 10x^2 - 36x - 72$ polinomunun kökleri bulup -5 – 5 arasındaki grafiğini çiziniz.
2) Kökleri 2 3 0 ve 9 olan polinomun katsayılarını hesaplayıp simgesel gösterdikten sonra düzenli görünümünü yazdırınız.
3) x=[2 3 4 5 6] y=[65 67 72 71 63] çözüm kümesine sahip f(x) polinomunun katsayılarını bulup simgesel olarak gösterdikten sonra f(12) değerini hesaplayınız.
4) $x^3 + 6x^2 - 13x + 19$ fonksiyonun türev fonksiyonu hesaplayıp türev fonksiyonun 10 18 ve 23 sayıları için değerini hesaplayınız

Not: Soruların cevaplarını Ek2 de bulabilirsiniz.

KAYNAKLAR

- Tr.wikipedia.org
- www.mathworks.com
- Matlab ve Uygulamaları, Yrd. Doç. Dr. Hikmet ÇAĞLAR
- Matlab Dersleri – Ders II Matlab İçerisindeki Temek Fonksiyonlar, Arş. Gör. Selman Fatih AVŞAR
- Advanced Matlab Graphics and GUI, Yair MOSHE
- Matlab Programlamaya Giriş
- Introduction to MATLAB R2012b
- A Pratical Introduction to Programing and Problem Solving Second Edition, Stormy ATTAWAY
- Matlab Giriş, Mehmet SiraçÖZERDEM
- Matlab Kullanım Klavuzu, Arş. Gör. Tolga Ensari – Öğr. Gör. Koray ÖZPOLAT
- Matlab Ders Notları, Hasan KORKMAZ
- Matlab Programlamaya Giriş
- Matlab
- Yeni Başlayanlar İçin Matlab Yardımcı Ders Notları, Yrd. Doç. Dr. Cüneyt AYDIN
- Matlab ile Grafik Çizimi, Doç. Dr. İrfan KAYMAZ
- Matlab M – Files
- Matlab Programing Fundemantals R2014a, MathWorks
- Matlab ve Simulink Kullanımına Giriş, Arş. Gör. Barış DOĞAN
- Matlab Matematik Programlama Dili, Osman ÖZTÜRK
- Matlab, Özgür KOCA
- Matlab
- Matlab'da Genel İşlemler, Çoşkun TAŞDEMİR
- CORS-TR Eğitim Seminerleri Serisi: Matlab'a Giriş – Ders 1
- CORS-TR Eğitim Seminerleri Serisi: Matlab'a Giriş – Ders 2
- CORS-TR Eğitim Seminerleri Serisi: Matlab'a Giriş – Ders 5
- Matlab Diferansiyel Denklemler, Emre ÖZBAY
- Matlab, Tolga BEKLER
- Matlab Tutorial
- Matlab'ın Temelleri, Muharrem TÜMÇAKIR
- Matlab

EK 1

Matlab Hızlı Erişim Kılavuzu

Temel Komutlar

help y	Bir y fonksiyonu için yardım
clear	Atanan tüm değişkenlerin silinmesi
clear x	Bir x değişkeneninin silinmesi
pwd	Çalışma klasör yolu
demo	Matlab demo penceresi
save	Matris kaydetme
load	Matris geri çağırma
clc	Ekranın temizlenmesi

Matematiksel Operatörler

+	Toplama
-	Çıkarma
*	Çarpma
/	Bölme
^	Üst alma
.*	Elemanter çarpım
./	Elemanter bölüm
.^	Elemanter üst alma
sqrt	Kök alma
abs	Mutlak değer

Mantıksal Operatörler

&	Ve
|	Veya
~	Değil
/	Bölme

Karar Operatörleri

>	Büyüktür
<	Küçüktür
>=	Büyük eşittir
<=	Küçük eşittir
==	Eşittir
~=	Eşit değildir

Semboller

%	Açıklama getirme ifadesi
[ ]	Matris girme ifadesi
( )	İndis ve değer girme ifadesi
=	Değer atama ifadesi
;	Matrislerde satır ayracı, değer atamada kullanılırsa değişken değerinin command window ekranında görünmesini engeller
:	Kolon ifadesi
ans	değişken atanmamış işlem için sonucu tutan değişken
{ }	Hücre Dizisi
x.adi	X yapı dizisi

Sabit Terimler

pi	pi sayısı
eps	$2.2204*e^{16}$
inf	Sonsuz(belirsiz) ifadesi

Trigonometrik Fonksiyonlar

sin, cos, tan, cot	Trigonometrik fonksiyonlar
asin, acos, atan, acot	Ters trigonometrik fonksiyonlar

Logaritmik Fonksiyonlar

log	Doğal logaritma
exp	Eksponansiyel

Lineer Cebir Fonksiyonları

det(x)	Bir x matrisinin determinantı
inv(x)	Bir x matrisinin tersi
trace(x)	Bir x matrisinin izi
diag(x)	Bir x matrisinin köşegen elemanları
zeros	Sıfır matris
ones	Birler matrisi
eye	Birim matris
eig	Özdeğer ve özvektör bulma

Yazdırma Fonksiyonları

fprintf	Bir ifadenin yazdırılması
sprintf	Bir ifadenin bir karakter dizisine aktarılması
disp	Bir ifadenin ekran çıktısı olarak gösterilmesi

Karakter Dizisi Fonksiyonları

num2str(x)	X sayısının bir karaktere atanması
str2num(x)	Karekter olan bir x sayısının sayısal bir değişkene atanması
char(a)	a hücresini bir karaktere atama
char(s1,s2)	s1, s2, ... karakterlerinden yeni bir karakter oluşturulması
lower(s)	Bir s karakter dizisinin tüm harflerinin küçük harfle yazdırılması
upper(s)	Bir s karakter dizisinin tüm harflerinin büyük harfle yazdırılması
isnumeric	Bir değişkenin sayı olup olmadığını sorgulama
ischar	Bir değişkenin karakter olup olmadığını sorgulama
iscell	Bir değişkenin hücre olup olmadığını sorgulama

Hazır GUI'ler

msgbox	İleti penceresi
inputdlg	Değer girme penceresi
questdlg	Soru diyalog penceresi
uigetfile	Open file diyalog penceresi
uigetdir	Open directory diyalog penceresi
uiputfile	Save file diyalog penceresi

Sayı Yuvarlama Fonksiyonları

fix	Sıfıra yuvarlatma
floor	Negatif sonsuza yuvarlama
ceil	Pozitif sonsuza yuvarlama
round	En yakın tam sayıya yuvarlama

Bazı Matematiksel Fonksiyonlar

sum(x)	Bir x vektörünün elemanlar toplamı
diff(x)	Bir x vektörünün elemanlarının ardışık farklandırılması
mean(x)	Bir x vektörünün elemanlarının ortalaması
median(x)	Bir x vektörünün elemanlarının orta değeri(medyanı)
sort(x)	Bir x vektörü elemanlarının küçükten büyüğe sıralanması
max, min	En büyük ve en küçük değer bulma
sort rows	İlgili sütuna göre küçükten büyüğe sıralama

Programlama

if/end	Eğer koşulu
for/end	Döngü
while/end	Şartlı döngü
input	Değişken girdirme

Çizim

plot	2 boyutlu grafik
plot3	3 boyutlu grafik
hist, bar	Histogram ve bar grafikleri
surf, mesh	Yüzey grafikleri
figure	Çizim penceresi oluşturma
hold on, hold off	farklı grafikleri aynı eksen takımında çizdirme
axis	Eksen komutu
axis equal	Eksenleri eşit ölçekli katsayı ile öl.
stem	Çubuk grafik
plotxy	Çift y eksenli grafik
errorbar	Hata bar grafiği

EK 2

Bölüm 2 Cevapları			
1	>>a=round(rand(3,6)*100) >>b=a’	3	>>f=[b(3,:),b(5,:)] >>g=f.^2
2	>>c=a(1:3,1:3) >>d=a(1:3,4:6) >>e=c.*d >>det(e)	4	>>h=reshape(a,2,9) >>save bolum2.mat

Bölüm 3 Cevapları		
	Editor Ekranı	Command Window
1	ifade=input('Uzunluğu hesaplanacak ifadeyi giriniz ','s'); fprintf('Girilen ifade %d karakter uzunluğundadır\n', length(a)) Yukarıda ki komutlar yazıldıktan sonra script uzbul.m olarak kaydedilir	>>uzbul

2	isi=input('Santigrad cinsinden sıcaklığı giriniz ');
	kel=isi+273.15;
	fah=32+(isi*1.8);
	fprintf('%1.2f santigrad sıcaklık %1.2f fahrenhat veya %1.2f
	kelvine eşittir\n',isi,fah, kel)
	Yukarıda ki komutlar yazıldıktan sonra script cevir.m olarak kaydedilir
		>>cevir

3	function [cevre] = ucgen (a,b,c) cevre=a+b+c; end Yukarıda ki komutlar yazıldıktan sonra function ucgen.m olarak kaydedilir	>>ucgen(4,2,5)
4	function [X,Y] = cizim(x1,x2,y1,y2,es) X=linspace(x1,x2,es) Y=linspace(y1,y2,es) plot(X,Y) end Yukarıda ki komutlar yazıldıktan sonra function cizim.m olarak kaydedilir	>>cizim(2,5,4,8,4)

Bölüm 4 Cevapları			
1	>> t=0:0.5:10; >> f=t.^2+5*t-3; >> g=t.^2+3; >> h=t; >> hold >> plot(f,t,'r'); >> plot(g,t,'g'); >> plot(h,t,'b'); >> text(f,t,'f'); >> text(g,t,'g'); >> text(h,t,'h');	3	>> [x,y] = meshgrid([-2:.25:2]); >> z=(x.^2-y.^2); >> mesh(x,y,z) >> colormap('jet') >> colormap jet
2	>> [X,Y] = meshgrid(-2:.2:2,-2:.2:3); >> Z = X.*exp(-X.^2-Y.^2); >> [C,h] = contour(X,Y,Z); >> clabel(C,h) >> colormap cool	4	>> x=-4:.5:4; >> y=-4:.5:4; >> z=x.^3-y.^3+cos(x+y); >> plot3(x,y,z) >> figure >> [X,Y] = meshgrid([-4:.5:4]); >> z=x.^3-y.^3+cos(x+y); >> plot3(x,y,z)

Bölüm 5 Cevapları		
	Editor Ekranı	Command Window

1	function [tr] = turev (p)
	% tr türev polinomu
	% p verilen polinom
	n=length(p)-1;
	p=p(:)'; % p yi satır haline getirir
	tr=p(1:n).*(n:-1:1); % katsayıları hesapla
	k=find(tr~=0);
	if ~isempty(k)
	tr=tr(k(1):end); %sıfırları sil
	else
	tr=0
	end
	Yukarıda ki komutlar yazıldıktan sonra function turev.m olarak kaydedilir
		>>turev([1 2 0 4])

2	function min = kucuk(vec) % kucuk vectörün en küçük elemanını bulur % Kullanımı: kucuk(vector) min = vec(1); %başlangıç olarak vektörün ilk elemanı ...en küçük farzediliyor for i = 2:length(vec) %ikinci elemandan başlayarak ...vektörün bütün elemanlarına bakmak için döngü kuruluyor if vec(i) < min min = vec(i); end	>>kucuk([1 5 8 -2])

	end end Yukarıda ki komutlar yazıldıktan sonra function kucuk.m olarak kaydedilir	
3	%Bu script yemek sipariş etmenizi sağlar secim = menu('Lütfen Siparişinizi Seçiniz','Çorba','Yemek','Tatlı'); switch secim case 1 disp('Çorba sipariş ettiniz.') case 2 disp('Yemek sipariş ettiniz') case 3 disp('Tatlı sipariş ettiniz') end Yukarıda ki komutlar yazıldıktan sonra scripti yemek.m olarak kaydedilir	>>yemek

4	% Kullanıcı açıyı girdikten sonra derece için 'd'
	% veya radyan için 'r' girer.; Açının işareti yazıldıktan
	% sonra diğer türe çevrim yapılır.
	aci = input('Açıyı girin ve d/r: ', 's');
	derece = aci(1:end-1);
	birim = aci(end);
	% Kullanıcının d yada r girmesi için hata kontrolu
	while birim ~= 'd' && birim ~= 'r'
	disp('Hatalı bilgi girişi. Derece yada radyan belirtilmemiş ')
	aci = input('Açıyı girin ve d/r: ', 's');
	derece = aci(1:end-1);
	birim = aci(end);
	end
	% Açı bilgisi sayısal olarak dönüştürülüyor
	derecesayi = str2num(derece);
	fprintf('Açı %.1f ', derecesayi)
	% Radyan yada derece olmasına göre sonuç hesaplanıyor
	if birim == 'd'
	fprintf('Sonuç :%.3f.\n', sind(derecesayi))
	else
	fprintf('Sonuç :%.3f.\n', sin(derecesayi))
	end
	Yukarıda ki komutlar yazıldıktan sonra scripti sinhes.m olarak kaydedilir
		>>sinhes

Bölüm 6 Cevapları			
1	>> p= [1 2 -5 -10 -36 -72]; >> roots(p) ans = 3.0000 -0.0000 + 2.0000i -0.0000 - 2.0000i -3.0000 -2.0000 >> t=-5:.5:5; >> plot(t,polyval(p,t))	3	>> x=2:6; >> y=[65 67 72 71 63]; >> p=polyfit(x,y,2) p = -1.8571 14.8571 41.6000 >> poly2sym(p) ans = (104*x)/7 - (13*x^2)/7 + 208/5 >> polyval(p,12) ans = -47.5429
2	>> poly ([2 3 0 7]) ans = 1 -12 41 -42 0 >> poly2sym(ans) ans = x^4 - 12*x^3 + 41*x^2 - 42*x >> pretty(ans) 4 3 2 x - 12 x + 41 x - 42 x	4	>> syms x >> f= x^3 + 6*x^2 -13*x +19; >> diff(f) ans = 3*x^2 + 12*x - 13 >> p=sym2poly(ans) p = 3 12 -13 >> polyval(p,[10 18 23]) ans = 407 1175 1850

EK 3

Eyleyici Uygulama Örneği:

"bobinveri" isimli excel tablosu

Güç (**W**)	Kutup	Boy (mm)	Çap (mm)	İç Çap (mm)	Derinlik (mm)	Bilezik genişlik (mm)
100	2	100	100	40	5	3
185	2	100	100	40	5	3
375	2	100	100	40	5	3
500	2	100	100	40	5	3
746	2	100	100	40	5	3
930	2	100	100	40	5	3
1120	2	100	100	40	5	3
1305	2	100	100	40	5	3
1492	2	100	100	40	5	3
1680	2	100	100	45	5	3
1865	2	100	100	45	5	3
2052	2	100	100	45	5	3
2240	2	100	100	46	5	3
2425	2	100	100	48	5	3
2611	2	100	100	48	5	3
2800	2	100	100	49	5	3
2984	2	100	100	50	5	3
3171	2	100	100	51	5	3
3357	2	100	100	51	5	3
3730	2	100	100	52	6	3
4476	2	100	100	52	6	4

5968	2	100	100	52	6	5
6714	2	100	100	53	6	6
7460	2	100	100	53	6	7
11190	2	100	100	55	6	8
14920	2	132	132	54	6	10
18650	2	140	140	57	6	11
22380	2	143	143	56	6	12
26110	2	148	148	57	6	13
29840	2	151	151	58	6	14
37300	2	154	154	60	7	14
44760	2	159	159	64	7	15
52220	2	164	164	64	7	16
63410	2	168	168	68	7	17
74600	2	170	170	70	7	18
89520	2	174	174	74	7	19
104440	2	180	180	77	7	20

1. 2 boyutlu grafiğinin oluşturulması;
   Adımlar:
   - Matlab "Başlat→Tüm Programlar→MATLAB→Matlab.exe" yolundan çalıştırılır.
   - Veri girişleri **bobinveri.xls** isimli excel dosyası **import** edilerek matlab'a dahil edilecek.
   - "Home→New→Script" yolundan **gcizim.m** fonksiyon dosyası oluşturulur.
   - Aşağıdaki kodlar yazılır.

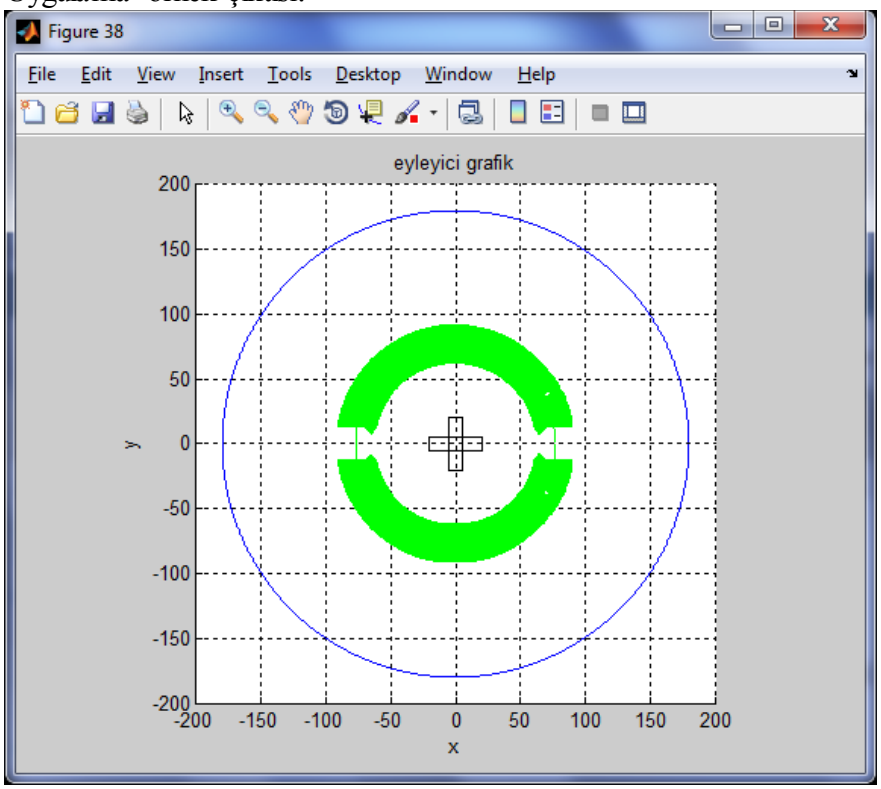
Adımlar	**Örnek**
Fonksiyon tanımlaması ve giriş-çıkış değişkenleri belirlenir	function [x,y,x1,y1,x2,y2,bilezikgen] = gcizim(boy,cap, iccap,derinlik,bilezikgen)
Verinin hazırlanması	boy1=boy;
Verinin hazırlanması	derinlik1=derinlik;
Verinin hazırlanması	x=-cap:0.1:cap;
Verinin hazırlanması	x1=-iccap+1:0.1:iccap-1;
Verinin hazırlanması	x2=-iccap:0.1:iccap;
Verinin hazırlanması	y=sqrt(cap^2-x.^2);
Verinin hazırlanması	y1=sqrt(iccap^2-x1.^2);
Verinin hazırlanması	y2=sqrt(iccap^2-x2.^2);
Fonksiyonun sonlandırılması	end

   - "Home→New→Script" yolundan **eyleyicigrafikler.m** fonksiyon dosyası oluşturulur.
   - Aşağıdaki kodlar yazılır.

Adımlar	**Örnek**
Matrix deki satır sayısı belirlenir	[r,c]=size(bobinveri);
Matrix deki satır sayısı adedince for döngüsü çalıştırılır	for i=1:1:r;
Fonksiyonun çağrılması	[x,y,x1,y1,x2,y2,bilezikgen]= gcizim(bobinveri(i,3), bobinveri(i,4),bobinveri(i,5), bobinveri(i,6),bobinveri(i,7));
Yeni grafik penceresi açılması	figure(i);
Çoklu grafik çizme aktif	hold on
Çizim fonksiyonuna girilmesi,	plot(x,y,'b',x,-y,'b');
Çizim fonksiyonuna girilmesi,	plot(x1,y1,'g',x1,-y1,'g','LineWidth',bilezikgen);
Çizim fonksiyonuna girilmesi,	plot(x2,y2,'g',x2,-y2,'g');
Çizim fonksiyonuna girilmesi,	rectangle('Position',[-5,-20,10,40]);
Çizim fonksiyonuna	rectangle('Position',[-20,-5,40,10]);

girilmesi,	
Çoklu grafik çizme sonlandırma	hold off
Grafik ayarlama	sekil=get(0,'currentfigure');
Grafik ayarlama	eksen=get(sekil,'currentaxes');
Grafik ayarlama	get(eksen,'dataaspectratio');
Grafik ayarlama	set(eksen,'dataaspectratio',[1 1 1]);
Grafik başlığı	title('eyleyici grafik');
X eksen ismi	xlabel('x');
Y eksen ismi	ylabel('y');
Izgara aktif	grid on;
Fonksiyon sonlandırma	end

- Run butonu çalıştırılır ve grafiğin oluşması sağlanmış olur.
- Uygulama örnek çıktısı:



2. 3 boyutlu grafiğinin oluşturulması;
Adımlar:

- Matlab "Başlat→Tüm Programlar→MATLAB→Matlab.exe" yolundan çalıştırılır.
- Veri girişleri **bobinveri.xls** isimli excel dosyası **import** edilerek matlab'a dahil edilecek.
- "Home→New→Script" yolundan **gcizim3.m** fonksiyon dosyası oluşturulur.
- Aşağıdaki kodlar yazılır.

Adımlar	**Örnek**
Fonksiyon tanımlaması ve giriş-çıkış değişkenleri belirlenir	function [x,y,x1,y1,x2,y2,bilezikgen, z,z1,z2] = gcizim3(boy,cap, iccap,derinlik,bilezikgen)
Verinin hazırlanması	boy1=boy;
Verinin hazırlanması	derinlik1=derinlik;
Verinin hazırlanması	x=-cap:0.1:cap;
Verinin hazırlanması	x1=-iccap+1:0.1:iccap-1;
Verinin hazırlanması	x2=-iccap:0.1:iccap;
Verinin hazırlanması	y=sqrt(cap^2-x.^2);
Verinin hazırlanması	y1=sqrt(iccap^2-x1.^2);
Verinin hazırlanması	y2=sqrt(iccap^2-x2.^2);
Verinin hazırlanması	z=meshgrid(x,y);
Verinin hazırlanması	z1=meshgrid(x1,y1);
Verinin hazırlanması	z2=meshgrid(x2,y2);
Fonksiyonun sonlandırılması	end

- "Home→New→Script" yolundan **eyleyicigrafikler3.m** fonksiyon dosyası oluşturulur.
- Aşağıdaki kodlar yazılır.

Adımlar	**Örnek**
Matrix deki satır sayısı belirlenir	[r,c]=size(bobinveri);
Matrix deki satır sayısı adedince for döngüsü çalıştırılır.(bellek dolması olduğu için 1 de sonlandırılmıştır)	for i=1:1:1;
Fonksiyonun çağrılması	[x,y,x1,y1,x2,y2, bilezikgen,z,z1,z2]= gcizim3(bobinveri(i,3), bobinveri(i,4),bobinveri(i,5), bobinveri(i,6),bobinveri(i,7));
Yeni grafik penceresi açılması	figure(i);
Çoklu grafik çizme aktif	hold on
Çizim fonksiyonuna girilmesi,	plot3(x,y,z,'b',x,-y,z,'b');
Çizim fonksiyonuna girilmesi,	plot3(x1,y1,z1,'g',x1,-y1,z1,'g','LineWidth',bilezikgen);

Çizim fonksiyonuna girilmesi,	plot3(x2,y2,z2,'g',x2,-y2,z2,'g');
Çizim fonksiyonuna girilmesi,	rectangle('Position',[-5,-20,10,40]);
Çizim fonksiyonuna girilmesi,	rectangle('Position',[-20,-5,40,10]);
Çoklu grafik çizme sonlandırma	hold off
Grafik ayarlama	sekil=get(0,'currentfigure');
Grafik ayarlama	eksen=get(sekil,'currentaxes');
Grafik ayarlama	get(eksen,'dataaspectratio');
Grafik ayarlama	set(eksen,'dataaspectratio',[1 1 1]);
Grafik başlığı	title('eyleyici grafik');
X eksen ismi	xlabel('x');
Y eksen ismi	ylabel('y');
Izgara aktif	grid on;
Fonksiyon sonlandırma	end

- Run menüsünden çalıştırılır ve grafiğin oluşması sağlanmış olur.
- Uygulama örnek çıktısı: