UYGULAMA : MATLAB PROGRAMLAMA DİLİNE GİRİŞ

GIRIŞ

MATLAB yüksek seviyeli bir teknik programlama dili olmasının yanında algoritma geliştirme, verilerin görselleştirilmesi,veri analizi ve sayısal hesaplamalar için etkileşimli bir yazılım platformu sunar. MATLAB ile teknik hesaplama problemlerini, C, C++ ve Fortran gibi geleneksel programlama dillerinden daha hızlı bir şekilde çözebilirsiniz. MATLAB yazılımının birçok alanda uygulamaları vardır. İçerdiği "toolbox" adı verilen araçlar aracılığıyla sayısal işaret işleme, kontrol tasarımı, test ve ölçüm, finansal modelleme ve analiz, haberleşme gibi birçok alanda kullanılabilmektedir. MATLAB programlama paketi MATHWORKS firması (http://www.mathworks.com) tarafından geliştirilmiş ve günümüzde 14.03 sürümü kullanıma sunulmuştur.

Ana Özellikleri

• Teknik hesaplamalar için yüksek seviyeli bir dil,

• Kodların ,dosyaların ve verilerin düzenlenmesi için bir geliştirme ortamı,

• İteratif tasarım ve problem çözme yöntemleri için interaktif araçlar,

• Lineer cebir, İistatistik, Fourier Analizi, Filtreleme, Optimizasyon, Sayısal türev ve integral için matematik fonksiyonlar,

• Verilerin görselleştirilmesi için 2 ve 3 boyutlu grafik araçları,

• Grafik arayüzler tasarlamak için araçlar,

MATLAB'in kullanımı olmayan bir mühendislik alanı hemen hemen yok gibidir.

Araç Kutuları

MATLAB 'ın kullanılırlığı ve çok yönlülüğü, uygulamalara özgü çeşitli araç-kutuları eklemekle geliştirilebilir. Araç kutuları, çeşitli bilimsel alan ve konularda yazılan MATLAB fonksiyon dosyalarından oluşurlar. Aşağıda kısaca açıklanan, değişik bilim dalları ile ilgili olarak yazılmış hazır araç-kutuları yer almaktadır.

Signal Processing Toolbox (Sinyal İşleme Araç kutusu): 1-boyutlu ve 2-boyutlu sayısal sinyal işleme (zaman serilerinin analizi gibi) ile ilgili fonksiyonlardan oluşmaktadır. Ayrıca, sayısal filtreler için geliştirilen analiz ve tasarım fonksiyonları ile güç spektrumu analizine ilişkin fonksiyonları da içermektedir.

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

System Idendification Toolbox (Sistem Tanımlama Araç kutusu): Parametrik modelleme ve sistem tanımlama ile ilgili fonksiyonlardan oluşmaktadır.

Neural Network Toolbox (Yapay Sinir Ağları Araç kutusu): Yapay sinir ağları için tasarım ve benzetim (simülasyon) fonksiyonlarından oluşmaktadır. Bu fonksiyonlar birkaç kategoriye ayrılırlar. Bunlar;

1) Ağ benzetimi için transfer fonksiyonlarının belirtilmesi,

2) Ağ parametrelerini güncelleştirmek için kuralların belirtilmesi,

3) Veriler üzerinde ağın işlem yapabilmesi için fonksiyonların belirtilmesi. Bazı fonksiyonlar, lineer ve nonlineer ağların doğrudan tasarımı için kullanılabilirler.

Spline Toolbox (Oluşum Araç kutusu): Oluşumlar ile ilgili M-dosyalarından oluşmaktadır. Oluşum araç kutusu fonksiyonel modellemede kullanılmaktadır. Eğrilerin modellenmesi, verilere göre eğri uydurulması, fonksiyonel denklemlerin çözülmesi vb. için oldukça kullanışlıdır.

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

m-Analysis and Synthesis Toolbox (p.Analiz ve Sentez Araç kutusu): m -Analiz ve sentez tekniklerinin kullanılarak robot ve lineer kontrol sistemlerinin incelenmesi ve tasarlanması için yazılmış fonksiyonlardan oluşmaktadır.

MATLAB İLE ÇALIŞMAK

MATLAB' ı gözünüzde canlandırmanın en kolay yolu, onu tüm niteliklerle donatılmış bir hesap makinesi gibi düşünmektir. Basit bir hesap makinesinin yaptığı toplama, çıkarma, çarpma ve bölmeden ibaret dört işlemi kolaylıkla yapar. Bunlara ilaveten teknik bir hesap makinesinde bulunan karmaşık sayılar, karekök, ve üst alma ve sinüs, kosinüs ve tanjant gibi geometrik işlemlerde kolaylıkla yürütülür. Bunun dışında, programlanabilir bir hesap makinesinde olduğu gibi veri saklama ve geri yükleme gibi işlemler ile önemli bir sorunun hesaplamasını otomatik hale getirmek için komut satırlarını oluşturabilir, icra edebilir veya saklayabiliriz. Ayrıca çok güçlü bir hesap makinesinde olduğu gibi çok çeşitli yollardan veri grafiklerinin oluşturulması,matris aritmetiğinin icrası, polinomların incelenmesi, fonksiyonların integre edilmesi, denklemlerin sembolik olarak kullanılması v.b. işlemlerin yapılmasını olanaklı kılar.

Gerçekte, MATLAB çok daha fazla özelliklere sahip olup, herhangi bir hesap makinesinden daha çok yönlüdür. MATLAB matematik hesaplamalar yapmaya yarayan bir araçtır. FORTRAN, BASIC, PASCAL, C gibi bilgisayar programlama dillere göre kullanımı daha kolay ve daha gelişkin niteliklere sahip bir programlama dilidir. Güçlü grafik yetenekleri sayesinde verilerin görüntülenmesi ve canlandırılması için zengin bir ortam sağlar. MATLAB, özgün problemlerin çözümüne görsel yaklaşım sunan grafik kullanıcı ara birimleri (GUIS) oluşturulmasına olanak sağlayan bir uygulama geliştirme platformudur. Bütün bunlara ilaveten MATLAB Toolbox`lar adı altında özgün uygulama alanları için problem çözücü araç takımları sunar. Örneğin MATLAB öğrenci baskısı Control Sistem Toolbox, Signal

Processing Toolbox ve Symbolic Math Toolbox gibi üç önemli toolbox içermektedir. Ayrıca özgün problemle uğraşanlar kendi Toolbox`larını da oluşturabilirler.

MATLAB ÇALIŞMA ALANI

MATLAB çalışma alanı, MATLAB komut penceresinden(command window) kullanılabilecek değişkenler (dizimler olarak bilinen) takımını içerir. who veya whos komutlarını kullanarak o andaki çalışma alanı içindekiler görüntülenebilir. who komutu sadece değişkenlerin isimlerini kısa bir liste halinde verirken, buna karşılık whos komutu ayrıca boyutu ve veri türü bilgileri de içerir.

Çalışma alanı ayrıca komut penceresinde yer alan araçlar üzerindeki çalışma alanı tarayıcısı (Workspace Brower) penceresini açarak da görüntülenebilir. Tüm bilgilerin görüntülendiği bu pencerenin araçlar üzerinde küp biçimde bir şekil vardır. Bu şekil üzerinde tıklandığında tarayıcı pencere açılır. Çalışma alanında yer alan tüm değişkenleri silmek için

clear

komutu kullanılır.

MATLAB komut penceresi

MATLAB açıldığında karşımıza gelen pencere MATLAB'ın komut penceresidir. Komut penceresi kullanıcı ile MATLAB komut yorumlayıcısı arasında iletişimi sağlayan bir ara yüzdür. Yorumlayıcı hazır hale geldiğinde >> iletisi karşımıza gelir. Bu ileti MATLAB'a komut ya da komut dizileri girilebileceğini gösterir.

Genel Komutlar

Demo komutu(demo): Eğer MATLAB ilk defa kullanılıyor ya da belli komutların çalışmasını merak ediliyorsa demo komutu ile demostrasyon listesini görüntülenebilir. Listeden yapacağınız bir seçimle seçtiğimiz işlevin icraatlarını adım adım izleme imkanı bulabilirsiniz.

Saklama ve geri çağırma komutları (save-load):

Bilgisayarınızda MATLAB ile çalışırken bilgisayarınızı kapatmayı arzulayabilirsiniz. Daha sonra geri dönerek kaldığınız yerden devam etmek isteyebilirsiniz. İşte bunu başarmak için kullanıyor olduğunuz bütün değişkenleri yeniden ayarlamadan bilgisayarınızı kapatmadan önce

>>save

komutunu kulanın.

Bu komut kullanımda olan MAT dosyasını alt dizininde veya MATLAB dosyanızda MATLAB.mat diye yapar veya üstüne yazar. Sonra MATLAB.mat da yer alan çalışma alanını yeniden çağırma ihtiyacı duyduğunuzda

>> load

komutunu girmelisiniz.

MATLAB.mat haricinde başka bir isim ile de değişkenleri saklayabilirsiniz. Örneğin;

>>save dosyam

Bunun yanında

>> load dosyam

komutu ile de saklamış olduğunuz değişkenleri geri çağırabilirsiniz.

Çalışma alanındaki verilerin kaydedilmesi ve yüklenmesi, save ve load komutları:

MATLAB`ın save ve load komutları bir oturumun her hangi bir anında MATLAB çalışma alanı içeriklerinin kaydedilmesi ve bu oturum sırasında veya daha sonraki bir oturumda kaydedilen bu verilerin tekrar çalışma alanına yüklenmesini sağlar. save ve load komutları aynı zamanda yazı (text) türü veri dosyalarının da çalışma ortamına ithal edilmesini sağlar.

save komutu çalışma alanı içeriğini bir ikili kod da (binary) MAT-dosyası olarak kaydeder. Bu dosya daha sonra load komutu ile geri çağırılabilir.

MATLAB Komut Penceresi Menüleri:

Bir çok programda olduğu gibi MATLAB'da da komut penceresi menüleri büyük kolaylıklar sağlar. Buna göre MATLAB'daki menüler ve işlevleri şu şekildedir.

File (Dosya) Menüsü: File menüsü dosya veya dosyaların oluşturulması ve yazdırma işleminin ayarlamalarının olduğu komutları içerir.

New: Bu komut şu seçenekleri içerir:

M-File: yeni M-dosyası oluşturmak için boş bir pencere açar.

Figure: yeni bir şekil penceresi oluşturur.

Model: yeni bir SIMULINK penceresi oluşturur.

Open M-File: Bir dosya seçebileğiniz pencere ekrana getirerek dosya adı girilmesini veya gereken dosyanın seçilmesini ister ve ardından seçilen yada ismi yazılan dosya metin düzenleyici programı çalıştırılarak açılır.

Open Selected: Komut penceresinde seçilerek belirtilen bir M-dosyasını varsayılan düzenleyiciyi çalıştırarak açar.

Save Workspace As...: Çalışma alanını kaydetmek için bir iletişim kutusu görüntülenir, yeni bir dosya adı girmeniz gerekmektedir.

Run M-File: Dosya Yöneticisi'nin Çalıştır... komutuna benzer. Bir M-dosyası adı girmeniz veya seçmeniz için bir iletişim kutusu görüntülenir ve belirtilen M-dosyasını çalıştırır.

Look for Selected: MATLAB'in lookfor komutunu çalıştırır. MATLAB'in arama yolunda bulunan tüm M-dosyalarının içindeki yardım metinlerinin ilk açıklama satırlarını tarayarak komut penceresi içinde seçilen katarı araştırır ve sonucu ekranda görüntüler.

Print...: Komut Penceresinde seçilen metni o an yüklü bulunan bir yazıcıya döker. Eğer seçilen metin yoksa, tüm MATLAB oturumu boyunca girilen metni yazdırır.

Printer Setup...: O andaki yazıcı ayarlarını ve seçeneklerini (renk tonu gibi) değiştirmek için bir iletişim kutusunu görüntüler.

Exit MATLAB: MATLAB oturumunu kapatır.

Edit (Düzen) Menüsü:

Edit (Düzen) menüsü komutları kullanıcıya düzenleme fonksiyonlarını uygulamak için büyük kolaylıklar sağlarlar.

Cut: Komut Penceresi'nde seçilen metni 'keser' ve ortamda saklar.

Copy: Komut Penceresi'nde seçilen metni 'kopyalar' ve ortamda saklar.

Paste: O andaki pano içeriğini komut satırına yapıştırır.

Clear Session: Komut penceresinin içeriğini siler. Bu komut, clc komutu ile aynı görevi icra eder.

Options (Seçenekler) Menüsü:

Bu menü; MATLAB'da pencere seçeneklerini ayarlamak, varsayılan düzenleme programını seçmek ve MATLAB'in format ve echo komutlarının işlevlerini değiştirmek için kullanılır.

Numeric Format: Ekran çıktı biçimlerini değiştirmek için bu komutu kullanabilirsiniz. Bu komut şu seçeneklerden oluşmaktadır.

Turn Echo On/Off (Yansıma Açık/Kapalı): Yansıma durumu için açık ve kapalı arasında geçiş yapar. Echo on ise verilen bir komutun sonucunun ekranda görüntülenmesini sağlar.

Enable/Disable Background Process: Artalan işlemlerinin olup olmaması arasında geçişi sağlar.

Font...: Yazı fontları iletişim kutusunu açarak buradan komut penceresinde kul¬lanılan font ve artalan rengini seçebiliriz.

MATLAB Değişkenleri:

MATLAB' da, herhangi bir tip tanımlaması veya boyut ifadesine gerek yoktur. MATLAB, yeni bir değişken ismi ile karşılaştığında, otomatik olarak ans isminde bir değişken oluşturur ve uygun bir bellek miktarı ayırır. Eğer değişken zaten varsa, MATLAB gerekli bir bellek ayırdığında içeriği değişir. Örneğin,

ogrenci_sayı=51
ogrenci_sayı diye isimlendirilen 1x1 matrisi oluştur ve 51'i yükler.

Değişken isimleri; bir harfden veya alt çizgiden oluşur.MATLAB, sadece değişken isminin ilk 31 karakterini kullanır. MATLAB, büyük ve küçük harfe duyarlıdır, büyük harf ile küçük harfi ayırdeder. A ve a değişkenleri aynı değildir.

Sayılar: MATLAB'da sayılar yaygın olarak kullanılan onluk tabanda ifade edilirler. Bunun yanısıra onluk tabanda üstel olarak veya i veya j olarak kompleks sayı biçimlerinde de ifade edilebilirler. Örnek olarak,

3, -99, 0.0001
9.6397238, 1.60210e-20, 6.02252e+23
1i, 3.141592j
sayıları gösterilebilir.

Sayıların duyarlılığını belirtmek için kullanılan eps sayısı onluk tabanda 16 basamaklı olarak gösterilmektedir.

Operatörler(sayısal işlemciler):

Matematiksel ifadeleri oluşturmak için operatörler ve önceden tanımlanmış sembolleri kullanabilirsiniz. Operatörler özetle şunlardır:

İki skaler sayı arasındaki aritmetiksel işlemler Bir deyim aşağıda olduğu gibi değer atanarak belli bir değer içinde saklanabilir.

x=a + b

Bu ifadede a ve b nin toplandığı ve x değişkeni içinde saklandığı belirtilmektedir. Bu atama işlemini; a içindeki bir değerin b içindeki bir değerle toplanarak bu toplamın x değişkenine atanacağı şeklinde yorumlamak mümkündür.

Eğer atama işlemi bu şekilde yorumlanacak olursa, aşağıda verilen bir MATLAB bildirimi de geçerli olur.

sayı = sayı + 1;

Açık bir şekilde bu bildirim geçerli bir cebirsel bildirim olamaz, fakat MATLAB içindeki 1' in sayı içindeki bir değere ilave edileceğini ve sonucun tekrar sayı içinde saklanacağını belirtir. Sonuçta sayı içindeki değerin her seferinde 1 artacağının belirtilmesine denktir.

Belli bir değişken tanımlamadan girilen deyimlerin icrasında ans isimli bir değişken içinde otomatik olarak saklanır. Her defasında ans içindeki değer bir öncekinin yerini alır. Burada ans İngilizce cevap anlamına gelen answer kelimesinin kısaltılmış şeklidir.

Matrislerle yapılan işlemlerde bölme işlemi için iki farklı sembol kullanılmaktadır. Bunun yanında eğer sayılar skaler ise iki bölme işleminin sonucu da aynı değeri gösterecektir. Örneğin 3/2 ile 2\3 ifadelerinin sonuçları aynı olup 1.5' dir.

Aritmetiksel İşlemlerde Öncelik Durumu: Tek bir aritmetiksel durum içinde birden fazla durum bir arada bulunabildiğine göre hangi işlemin öncelik hakkına sahip olunduğunun bilinmesi yerinde olacaktır.

Disp FONKSİYONU

MATLAB' da bir matematiksel ifadeyi argüman olarak alıp, bu ifadenin sonucunu ekrana aktaran bir fonksiyon mevcuttur bu da disp fonksiyonudur. Ancak disp fonksiyonunu kullanmaksızın sadece ifadeyi yazarsak ta sonucu görebiliriz.

```
» 2+8
ans = 10
» disp(2+8 )
10
»
```

, SEMBOLÜ

Dizi ya da matris elemanları arasına ayraç olarak yerleştirilir. Bu sembol yerine boşluk sembolü kullanılması da aynı etkiyi sağlar.

```
» [ 5 , 7 ]
ans = 5  7
» disp( [ 3 , 4 ] )
3  4
» disp( [3 4] )
3 4
»
```

Karakter türü veriler ' ' sembolleri içine alınır. disp fonksiyonu ile aşağıdaki gibi görüntülenirse bitişik olarak ekrana aktarılırlar.

```
 » disp([3 4])
3 4
» disp(['a' 'c'])
ac
»
```

; SEMBOLÜ

; Sembolü, aralarında yerleştirildiği iki skaleri iki farklı satıra yazar.

```
» disp([4;8])
4
8
»
```

; sembolü ayrıca ilerde görüleceği gibi matrislerde satır ayracı olarak kullanılır.

: SEMBOLÜ

: Sembolü başlangıç ve son değerleri belirten bir sayı dizisini 1'er artımlarla üretilir. Başlangıç ve son değerler yanında bir de artım değerleri üçüncü parametre olarak verilirse bu durumda da belirten artımı kullanarak bir sayı dizisi üretir(örneğin for döngüsünde olduğu gibi). Üç parametre kullanılırsa ilk parametre başlangıç, ikinci parametre artım ve üçüncü parametre ise son değerdir.

```
» 1:6
ans = 1 2 3 4 5 6
» disp(2:5)
2 3 4 5
»
```

ve aynı zamanda

```
» 2:3:18
ans =
2 5 8 11 14 17
» 0.4:0.7:10
ans =
Columns 1 through 7
0.4000 1.1000 1.8000 2.5000 3.2000 3.9000 4.6000
Columns 8 through 14
5.3000 6.0000 6.7000 7.4000 8.1000 8.8000 9.5000
»
```

AYNI UZUNLUKLARDAKİ VEKTÖRLER ÜZERİNDE İŞLEMLER

TOPLAMA VE ÇIKARMA

+ ve – sembolleri iki vektör arasında da kullanılabilir; a ve b üçer elemanlı iki vektör olsun:

```
» a=[2  1 -1]

a =
2 1 -1
» b=[4 -2 3]
b = 4 -2 3
» a+b
ans = 6 -1 2
» a-b
ans = -2 3 -4
»
```

yukardaki örneklerde görüldüğü gibi, toplama ve çıkarma işlemlerinde bilinen vektör toplamı ve farkı işlemi gerçekleştirilecektir.

ÇARPMA VE BÖLME

Eşit uzunlukta iki vektör için * ve / operatörleri kullanılırken dikkatli olunmalıdır.

```
» a=[4 5]
a = 4 5
» b=[3 -2]
b = 3 -2
» c=a*b
??? Error using ==> *
Inner matrix dimensions must agree.
»
```

Burada MATLAB için * sembolü matris çarpımı sembolüdür ve a ve b çarpılabilecek tipte matrisler olmadıkları için yukarıdaki hata mesajını alınmaktadır.

.* sembolü, elemanları, iki vektörün karşılıklı elemanların çarpımından oluşan aynı uzunlukta yeni bir vektör üretecektir. Yani dizey veya vektörlerin eleman eleman çarpımında kullanılır.

```
» c=a.*b
c = 12 -10
»
```

Benzer biçimde ./ ve .\ operatörleri de geçerlidir. Aşağıda örneklerde inceleyelim:

```
» a=[4 5]
a = 4 5
» b=[3 -2]

b = 3 -2
» d=a./b
d = 1.3333 -2.5000
» e=a.\b
e = 0.7500 -0.4000
```

```
»
```

EŞİT UZUNLUKTA İKİ VEKTÖR ARASINDA ^ OPERATÖRÜ

```
» a=[4 5]
a = 4 5
» b=[3 -2]
b = 3 -2
» h=a^b
??? Error using ==> ^
Matrix dimensions must agree.
»
```

Yukarıda görüldüğü gibi iki vektör arasında ^ işlemi tanımsızdır. .^ sembolü geçerlidir ve birinci vektörün bileşenleri taban ve ikinci vektörün bileşenlerini de üst kabul ederek üst alma işlemi sonucu aynı boyutta yeni bir vektör oluşturacaktır. Aşağıda örneği inceleyelim.

```
» h=a.^b
h = 64.0000 0.0400
»
```

BİR VEKTÖR VE SKALER ARASINDAKİ İŞLEMLER

TOPLAMA VE ÇIKARMA İŞLEMLERİ:

Aşağıda örneklerde görüldüğü gibi, bir skaler ile bir vektör operatörü ile işleme sokulursa, skaler vektörün her iki bileşeni ile de toplanır.

```
» 4+[2 -2]
ans = 6 2
»
```

- operatörü için de aynı şey söz konusudur.

```
» 7-[2 -1]
ans =  5 8
»
```

ÇARPMA VE BÖLME

Bir skaler bir vektör operatörü ile işlem sokulursa sonuçta bileşenleri skaler ile vektörün bileşenlerinin ayrı ayrı çarpılması ile oluşan yeni bir vektör elde edilir.

```
» 3*[2 -1]
ans = 6 -3
»
```

Bir skalerin bir vektöre bölümü ise tanımsızdır (/ sembolü ile):

```
» 3/[2 -1>
??? Error using ==> /
Matrix dimensions must agree.
»
```

Ters bölme (\ ) sembolü kullanılırsa vektörün bileşenlerinin skaler ile bölünmesinden elde edilen iki sayı yeni bir vektör oluşturacaktır.

```
» 2\[4 8]
ans = 2 4
»
```

Oysa bir vektörün bir skalere bölünmesi tanımlıdır ve bileşenleri vektörün bileşenlerinin skalere bölünmesinden elde edilen yeni bir vektör elde edilecektir.

```
» [4 -8]/2
ans = 2 -4
```

MATLAB' TA PROGRAM HAZIRLANMASI

M-dosya Programcılığı

MATLAB' da algoritmaları bilinen programları hazırlamak ve çalıştırmak çok kolaydır. Ayrıca FORTRAN, BASIC, C/C++ ve PASCAL gibi programlama dillerinde hazırlanmış programları MATLAB için uyarlamak mümkündür. Bu durumda çoğunlukla aynı program için daha az sayıda satır kullanmak yeterlidir. MATLAB' ın hazır M-dosya paketlerini kullanmak suretiyle programlamayı çok kısa tutmak mümkündür.

M-dosyaları

MATLAB dil kodu içeren dosyalara M-dosyası (M-files) adı verilir. M-dosyalarının iki türü mevcuttur. Bunlar;

• Fonksiyonlar (functions): Giriş olarak argümanlar kabul eder ve çıkış olarak bu argümanlara karşılık gelen çözümü üretir.

• Düzyazı (scripts) dosyaları: Bir dizi MATLAB deyimini otomatik olarak icra eder.

Bir M-dosyasının MATLAB tarafından bir dosya olarak onaylanması için ".m" uzantısına sahip olması gerekir.

MATLAB' ın kendisi büyük oranda herhangi bir program içinde çağırılıp kullanılabilen fonksiyon (alt program) M-dosyalarından oluşmuştur. Bunların dışında ayrıca yine program içinde çağırılabilen kendi özüne gömülü (built-in) özel fonksiyonlara sahiptir. M-dosyaları ASCII karakterinde hazırlanmış okunabilir ve yazılabilir dosyalardır. Bu nedenle MATLAB' a ait M-dosyaların yanlışlıkla değiştirilmemesi gerekir. Aksi taktirde orijinal görevini yerine getiremez. Bu nedenle kullanıcının kendi hazırladığı M-dosyalarını kendisine ait bir klasörde saklaması tavsiye edilir.

M-dosyalarının Oluşturulması, Metin Editörüne Giriş: M-dosyalarını bir metin olarak kullanarak oluşturulabilecek sıradan metin dosyalarıdır. M-dosyaları bu editörde hazırlanabileceği gibi herhangi bir başka metin (text) editöründe de hazırlanabilir.

Düzyazı M-Dosyaları

Bir dizi komutlardan ibaret düzyazı dosyaları, MATLAB içinde çağrıldığında dosya içinde bulunan komutlar otomatik olarak çalıştırılır. Böylece her seferinde klavyeden komutları tekrar tekrar girmeye gerek kalmaz. Bir düzyazı dosyası içinde yer alan deyimler MATLAB çalışma ortamında yer alan verileri işletir ve sonuçlandırır. Düzyazı dosyaları, MATLAB ortamında etkileşimli biçimde çalışılmayacak uzun komutlar dizisi gerektiren analizlerin icrası, problemlerin çözümü veya tasarım yapılmasında kullanışlı olmaktadır.

Örneğin aşağıda verilen MATLAB komutları dizisi `hata_example1.m` adı verilen bir dosyada oluşturulup saklanabilir.

MATLAB ortamında "`hata_example1`" yazmak (.m uzantısı yazılmaz) sureti ile program çalıştırılabilir. Yalnız bunun için program içinde yer alan x ve dx parametre değerlerinin program çalıştırılmadan önce atanmış olması gerekir. Bu, ya "`hata_example1`" komutu yazılmadan önce klavyeden yukarıda belirtilen parametre değerlerini girmek sureti ile ya da hazırlanmış bir veri dosyasını yüklemek sureti ile atanabilir.

```
%
% PROGRAM hata_example1.m
% JFM224 Sayısal Analiz ve Programlama III Dersi uygulama
% Konu: Hata Analizi
%
%
% Program f(x)=exp(x) fonksiyon değerini verilen bir x ve dx değeri
% ve adım aralığı için Taylor serisinin ilk bir, iki ve üç terimi
% kullanarak yaklaşık değeri, bağıl ve yaklaşım hatalarını hesaplar.
%
% Yazan : Dr.Unal Dikmen
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
function = hata_example1(x,dx)
fclose('all'); clc
format long
 f = 'exp(dx)';
 x = 0.0;
dx = 0.5 ;
% exp(x) fonksiyonun x=0.5 için gercek değeri
 g = eval(f,dx) ;
% Taylor acılımı:
%                          2             3                     n
% f(x+dx)=f(x)+dxf'(x)+( dx/2!)f''(x)+dx/6)f'''(x)...+...(dx/n!)fn(x)
%
% Seri açılımında ilk üç terim için fonksiyonun yaklaşık değerleri
f1t = 'exp(x)' ;
f2t = 'exp(x) + 0.5*exp(x)';
f3t = 'exp(x) + 0.5*exp(x) + (0.125)*exp(x)' ;
ft1 = eval(f1t,x);
ft2 = eval(f2t,x);
ft3 = eval(f3t,x);
%yuzde olarak bagıl hatalar
re1 = 100.*(g-ft1)/g ;
```

```
re2 = 100.*(g-ft2)/g ;
re3 = 100.*(g-ft3)/g ;
% yuzde olarak yaklaşım hataları
ye21 = 100.*(ft2-ft1)/ft2 ;
ye32 = 100.*(ft3-ft2)/ft3 ;
% sonucların ekrana aktarılması
clc
disp(' Fonksiyon = exp(x) ')
disp(sprintf(' x=0.5 için gerçek değeri = %15.12f\n',g));
disp(sprintf(' Taylor serisinin ilk terim yaklaşık degeri = %5.4f',ft1));
disp(sprintf(' Taylor serisinin iki terim yaklaşık degeri = %5.4f',ft2));
disp(sprintf(' Taylor serisinin üç terim yaklaşık degeri  = %5.4f\n',ft3));
%
disp('Bağıl Hataları :')
disp(' ')
disp(sprintf(' 1 terim yaklaşımındaki bağıl hata yüzdesi  = %4.2f',re1));
disp(sprintf(' 2 terim yaklaşımındaki bağıl hata yüzdesi  = %4.2f',re2));
disp(sprintf(' 3 terim yaklaşımındaki bağıl hata yüzdesi  = %4.2f\n',re3));
%
disp('Yaklaşım Hataları:')
disp(' ')
disp(sprintf(' iki terim hesaplamada yaklaşım hata yüzdesi= %4.2f',ye21));
disp(sprintf(' üç terim hesaplamada yaklaşım hata yüzdesi = %4.2f',ye32));
```

bu şekilde "`hata_example1`" çalıştırıldığı zaman MATLAB dosyası içinde yazılı komutları icra eder ve sonuçları ekrana aktarılır.

Fonksiyon M-Dosyaları

Fonksiyon dosyaları ilk satırda "function" kelimesi bulunan ".m" uzantısı bulunan dosyalardır. MATLAB içinde bulunan tüm M-dosyaları fonksiyon dosyaları biçiminde olup, bunlar hazırlanan herhangi bir program içinde çağırılabilir. Gerçekte bunlar FORTRAN, BASIC, C ve benzeri programlama dillerinde kullanılan alt programlar (subroutines) gibi işlem görürler. MATLAB' ta normal M-dosyaları biçiminde ve özünde gömülü fonksiyon olmak üzere iki tür fonksiyon mevcuttur. Fonksiyon dosyası; "function" satırında yer alan giriş argümanlar (function tarafından bulunan sonuçlar) istenirse diğer hesaplamalarda kullanılabilir. Fonksiyon dosyaları, MATLAB komut iletisindeki çalışma alanından ayrılmış, kendi çalışma alanı içindeki değişkenleri işletirler.

Fonksiyon dosyasının oluşturulması ve çalıştırılması aşağıda verilen basit bir örnekle açıklayabiliriz. Burada MATLAB' ta mevcut bir vektörün ortalamasını hesaplayan "`ortalama.m`" dosyasına ait bildirimler listesi verilmiştir.

```
Function y = ortalama(x)
%ortalama (x)her bir sütunda ortalama değeri olan bir satır
vektörüdür.
[m,n]=size (x) ;
if m=1
m=n;
end;
y=sum (x ) /m;
```

bu şekilde yazılıp yine "`ortalama.m`" olarak saklanan (dosya adı ile fonksiyon adı aynı olmalıdır) yeni fonksiyon dosyasının kullanımı herhangi MATLAB dosyasının kullanımından farklı değildir.

• Bir fonksiyon dosyasının belli başlı bölümleri ve özellikleri örnek "ortalama.m" fonksiyon dosyası üzerinden aşağıdaki şekilde açıklayabiliriz.

• Birinci satır fonksiyon tanım satırı olup, fonksiyon adını, giriş ve çıkış argümanlarının sayısını ve sırasını tanımlar. Bu satır bulunmadığı zaman, dosya düzyazı dosyası gibi işlem görür.

• % işareti ile başlayan satırlar yardım açıklama satırlarıdır. Bu satırların birincisi 1.yardım (help1) satırıdır. lookfor fonksiyonu veya tüm klasör yolu üzerinden yardım talebi olduğunda bir fonksiyon için MATLAB bu satırı görüntüler. Bundan sonraki satırlar yardım metnidir. Belli fonksiyon hakkında yardım istendiğinde MATLAB yardım satırı ile birlikte tüm yardım metnini görüntüler. Bu satırlara yazılanlar MATLAB tarafından icra edilmez.

• Fonksiyon gövdesi, % işareti ifade eden satırlardan sonraki satırlardır. Fonksiyonun bu kısmı hesaplamaları yerine kod içerir ve herhangi çıkış argümanları için değerleri saptar.

• Fonksiyon dosyası içinde yer alan m, n ve y gibi değişkenler "ortalama" dosyasının çalışması sırasında kendi içinde geçerli olup daha sonra çalışma ortamında görüntülenemez ve kullanılamaz.

Fonksiyon dosyaları hakkında daha ayrıntılı bilgi alma ve çeşitli MATLAB fonksiyon dosyalarını incelemek için MATLAB ortamında edit fonksiyon adı yazılarak biçimleri görüntülenebilir. Bazı komutların isimleri ve açıklamaları Çizelge 1' de verilmiştir.

Çizelge 1. komutlar

Komut	İşlev
help	MATLAB'ın operatör ve fonksiyonlarını tanımlar
who	Değişkelerin isimlerini listeler
whos	Değişkenlerin simlerini ve boyularını listeler
what	Diskinizdeki M-dosyalarını listeler
size	Argümanların boyutlarını verir
lenght	Argümanların maksimum boyutlarını verir
clear	Çalışma ortamındaki tüm değişkenleri temizler
quit	MATLAB ortamını sona erdirir
save	MATLAB çaışma ortamında bir MAT-dosyasını saklar

MATLAB' da OPERATÖRLER

A ve B skaler veya matrisin toplanmasında : **A+B** veya **plus(A,B)**
A ve B skaler veya matrisin çıkartılmasında : **A-B** veya **minus(A,B)**
A ve B skaler veya matrisin çarpılmasında : **A*B** veya **mtimes(A,B)**
A ve B skaler veya matrisin eleman düzeyinde çarpılmasında : **A.*B** veya **times(A,B)**
A ve B skaler veya matrisin sağdan bölünmesi: **A/B** veya **mrdivide(A,B)**
A ve B skaler veya matrisin eleman düzeyinde sağdan bölünmesi : **A./B** veya **rdivide(A,B)**
A ve B skaler veya matrisin soldan bölünmesi: **A\B** veya **mldivide(A,B)**
A ve B skaler veya matrisin eleman düzeyinde soldan bölünmesi: **A.\B** veya **ldivide(A,B)**

Matrisin gücü: **A^b** veya **mpower(A,b)**
Matrisin eleman düzeyinde gücü: **A.^b** veya **power(A,b)**
A dizeyinin karmaşık devriği: **A'** veya **transpose(A)**
A dizeyinin karmaşık (komplex) devriği: **A'** veya **ctranspose(A)**

MATLAB ' TA BAZI TEMEL KOMUTLAR

ls veya dýr : çalýþýlan dizindeki dosya isimlerini görüntüler
who veya whos: o an bellekteki deðiþkenleri görüntüler
pwd: çalýþýlan dizin yolunu ekrana verir.
clear: belleði temziler
clc: ekraný temizler (command window' u)
size(A): A dizeyinin boyutunu verir.
length(a) : a vektörünün uzunlugunu verir.
max(a): a vektörü içerisindeki maksimum deðeri bulur. a bir dizey ise
her bir sütünun maksimum deðerini bulur.
min(a): a vektörü içerisindeki mimumum deðeri bulur. a bir dizey ise
her bir sütünun mimumum deðerini bulur.

norm(A) : A dizeyinin normunu verir.
[d,ind]find(A>k): A dizeyi içerisinde k deðerinden büyük elemanlarý bulur.
d: k dan büyük deðerleri tutar ve ind komutu hangi satýr
ve sütün' da oldugunu gösteren indisi tutar.
diag(A): A matrisinin diagonal elemanlarýný verir.
trace(A): A dizeyinin izini verir.
inv(A): A dizeyinin tersini alýr.
[U D V]=svd(A) : A dizeyini SVD (Singular Value Decomposition) yöntemine göre çözer
cross(A,B): A ve B gibi iki matrisi çarpar
dot(A,B): A ve B gibi iki matrisi skaler çarpar
disp('string'): yazýlan string ifadeyi ekranda görüntüler
Linspace(a,b,N): a ve b arasýnda N adet eþit aralýklý nokta içeren vektör üretir.
logspace(a,b,N): a ve b arasýnda N adet eþit aralýklý logaritmik nokta içeren vektör üretir.
%%%%%%%%%%%%%%%%%%%%
plot(x,y,'-b','LineWidth',2): x,y veri çiftini plot eder. '-b' plotun sürekli cizgi ve
mavi renkli olmasýný söyler. 'LineWidth' ise cizgi kalýnlýgýný gösterir

MATLAB' TA MATRİS İŞLEMLERİ

MaTLAB programlama dili farklı yapıda matris ve vektör oluşturmak için çeşitli hazır fonksiyonlar kullandığı gibi kullanıcı da kendisine özgü matris ve vektörleri kolaylıkla oluşturabilir. Öncelikle MATLAB' ın hazır matris ve vektör oluşturma fonksiyonlarına kısaca bakacak olursak;

x=pascal(n) veya x=pascal(n,opt)

komut [nxn] boyutlu pozitif tam tanımlı bir simetrik matris oluşturur. opt seçeneği verilmeyebilir(bu durumda 0 değeri default olarak vardır)
pascal(n,1): Pascal matrisinin alt üçgen Cholesky katsayılarını içerir., ascal(n,2): pascal(n,1)' in döndürülmüş halidir.

Örnek:

```
x=pascal(3)
x =
   1    1    1
   1    2    3
   1    3    6

x=pascal(3,1)
x =
   1    0    0
   1   -1    0
   1   -2    1

x=pascal(3,2)
x =
   1    1    1
  -2   -1    0
   1    0    0
```

x=magic(n)
Komut [nxn] boyutlu 1' den n^2 ' ye kadar değerlerden eşit satır, eşit sütün ve diagonal elemaların toplamından oluşan simetrik olmayan bir kare matris oluşturur.

Örnek:

```
x=magic(3)
x =
   8    1    6
   3    5    7
   4    9    2
```

eye(n) veya **eye(n,m)**
Komut (nxn) veya (n,m) boyutlu birim matris oluşturur.

Örnek:

```
 x = eye(2,3)
x =
    1    0    0
    0    1    0
```

zeros(n,m)
Komut (nxm) boyutlu sıfır matrisi oluşturur.

Örnek:

```
x=zeros(2,3)
x =
    0    0    0
    0    0    0
```

rand(n) veya rand(n,m)
Komut nxn veya nxm boyutlarında üniform matris üretir.

Örnek:

```
x= rand(3)
x =
   0.9501    0.4860    0.4565
   0.2311    0.8913    0.0185
   0.6068    0.7621    0.8214

y=rand(3,2)
y =
   0.4447    0.9218
   0.6154    0.7382
   0.7919    0.1763
```

randn(n) veya randn(n,m)
komut normal dağılımlı nxn veya nxm boyutlu matrisler üretir.

Örnek:

```
x=randn(3)
x =
  -0.4326    0.2877    1.1892
  -1.6656   -1.1465   -0.0376
   0.1253    1.1909    0.3273
```

```
y=randn(3,2)
y =
   0.1746   -0.5883
  -0.1867    2.1832
   0.7258   -0.1364
```

sprand(n)
Komut üniform seyrek (sparse) yapıda nxn boyutlu matris oluşturur.

Örnek:

```
x=sprand(3)
x =
   (1,1)       0.4186
```

sprandn(n,m,density)
komut nxm boyutunda normal dağılıma sahip seyrek yapıda matris oluşturur.

Örnek:

```
x=sprandn(3,3,2)
x =
   (1,1)       1.6236
   (2,1)       0.8580
   (3,1)      -1.5937
   (1,2)      -0.6918
   (3,2)      -1.4410
   (2,3)       1.2540
   (3,3)       0.5711
```

sprandsym(n,density)

komut, nxn boyutunda seyrek simetrik random matris üretir.

Örnek:

```
sprandsym(3,1)
ans =
   (2,1)       1.1908
   (3,1)       2.0022
   (1,2)       1.1908
   (3,2)       2.1742
   (1,3)       2.0022
   (2,3)       2.1742
```

Matrislerde toplama ve çıkarma

Matrislerde toplama ve çıkarma işlemi matrisin veya vektörün elemanları üzerinde gerçekleştirilir.

Örnek:

```
a = pascal(3);
a =
   1   1   1
   1   2   3
   1   3   6

b = magic(3);
b =
   8   1   6
   3   5   7
   4   9   2

x = a +b
x =
   9   2   7
   4   7  10
   5  12   8

y = x - a
y =
    8   1   6
    3   5   7
    4   9   2
```

MATLAB' ta kolon işlemleri

Kolon operatörü olarak **:** kullanılır.

örneğin

```
X=1:10
X
1   2   3   4   5   6   7   8   9  10
```

veya

```
X=0:pi/4:pi
X=
0   0.7854   1.5708   2.3562   3.1416
```

gibi kolon operatörü ile matris ve vektörlerin istediğimiz kolon veya satırları üzerinde işlemler yapabiliriz. Örneğin A gibi bir matris üzerinde,

A(1:k,j)

işlemi ile A matrisinin j. ci sütün üzerinde 1' den k' ıncı satırlarını alabiliriz. Örnek olarak,

```
a =
   0.6902   0.3487   0.0742
   0.6897   0.9634   0.1656
   0.4450   0.1259   0.3717

>> a(1:2,3)
ans =
   0.0742
   0.1656
```

Örneğin A matrisinin 2. sütününu 4 ile çarpmak ve 5' e bölmek istersek,

```
a=rand(3)
a =
   0.8168   0.7339   0.9250
   0.0185   0.4776   0.6458
   0.5357   0.1875   0.8134

>> a(:,2)=4.*a(:,2)./5
a =
   0.8168   0.5871   0.9250
   0.0185   0.3821   0.6458
   0.5357   0.1500   0.8134
```

veya A dizeyinin 3 sütunundan 2. sütununu çıkartmak istersek

```
a=rand(3)
a =
   0.8708   0.9405   0.6957
   0.4817   0.2945   0.0910
   0.1039   0.2842   0.9548

>> a(:,3)=a(:,3)-a(:,2)

a =
   0.8708   0.9405  -0.2447
   0.4817   0.2945  -0.2035
   0.1039   0.2842   0.6706
```

yada A dizeyinin 2.satırını 1.satır ile çarpmak istersek,

```
a=rand(3)
a =
   0.1129   0.0585   0.0565
   0.4949   0.1158   0.4606
   0.0588   0.1154   0.9807

>> a(2,:)=a(2,:).*a(1,:)
a =

   0.1129   0.0585   0.0565
   0.0559   0.0068   0.0260
   0.0588   0.1154   0.9807
```

A matrisinin son sütününu kullanmak için

```
a=rand(3)
a =
   0.9851   0.2750   0.4407
   0.1145   0.0556   0.3954
   0.4202   0.8172   0.4408

>> a(:,end)
ans =
   0.4407
   0.3954
   0.4408
```

A matrisinin son satırını kullanmak için

```
a=rand(3)
a =

   0.1188   0.0603   0.9717
   0.6402   0.7386   0.6048
   0.8138   0.0243   0.6677

>> a(end,:)
ans =

   0.8138   0.0243   0.6677
```

Bu ve bunlara benzer matris üzerinde her türlü işlemde : operatörünü kullanabiliriz.

Matrisin Transpoze(devriği), vektörel çarpımı ve skaler çarpımı

Bir A dizeyinin transpozesi(devriği) ‘ operatörü ile verilir.

Örnek:

```
A=rand(3)
A =

   0.1692   0.5504   0.3248
   0.1558   0.0067   0.9042
   0.7333   0.5068   0.1569
```

A’ nın transpozesi A'

```
=A’
ans =
   0.1692   0.1558   0.7333
   0.5504   0.0067   0.5068
   0.3248   0.9042   0.1569
```

Aynı satır ve sütün boyundaki iki matrisin vökter çarpımı,

```
x = [3; 1; 4];
y = [2  0 -1];

z=x*y
z =
    6    0   -3
    2    0   -1
    8    0   -4
```

Skaler çarpımı

```
z=y*x
z =
    2
```

Eleman düzeyinde iki aynı boyutlu matrisi çarpmak için .* operatörü kullanılır.

```
a=randn(3)
a =
   0.2193   -0.0592    0.5077
  -0.9219   -1.0106    1.6924
  -2.1707    0.6145    0.5913

b=randn(3)
```

```
b =
  -0.6436   -0.0195   -0.3179
   0.3803   -0.0482    1.0950
  -1.0091    0.0000   -1.8740

c=a.*b
c =
  -0.1412    0.0012   -0.1614
  -0.3506    0.0487    1.8532
   2.1905    0.0000   -1.1081
```

MATLAB' ta matris ve vektör işlemlerinde kullanılan bazı fonksiyonlar:

norm(a):

bir matrisin veya vektörün normunu (max(svd(X)) verir.

```
a=rand(3)
a =
   0.4399    0.0319    0.0310
   0.9473    0.9757    0.4143
   0.2067    0.8049    0.0762
>> c=norm(a)
c =
   1.6296
```

svd(A)

komut A dizeyi üzerinde tekil değer ayrışım yöntemini uygular

```
[u,s,v] = svd(a)

a=rand(3)

a =

   0.1953    0.9944    0.7043
   0.6653    0.6614    0.3014
   0.7367    0.3337    0.3126

>> [u,s,v]=svd(a)

u =

  -0.6853    0.6975    0.2091
```

```
  -0.5697   -0.3348   -0.7505
  -0.4535   -0.6335    0.6268


s =

   1.6777         0         0
        0    0.6412         0
        0         0    0.1414


v =

  -0.5049   -0.8629    0.0232
  -0.7210    0.4068   -0.5609
  -0.4746    0.2999    0.8275
```

MATLAB programlama dilinde matrislerin satır veya sütünları arasındaki işlemlerde yine : operatörü kullanılır.

Örnek:

A matrisi 6x6 lık bir matris ise

1. A matrisinin 3. Satırı ile 5. Satırını yer değiştirelim.
   Tmp=A(3,:);
   A(3,:)= A(5,:);
   A(5,:)=Tmp ;

Yada aynı işlemi başka bir ifade ile

```
A= [ A(1, : )  ;  A(2, : ) ;  A(5, : ) ; A(4, : ) ; A(3, : ) ] ;
```

İlede yapabiliriz.

2. A kare matrisinin 1. Sütünü ile 3. Sütününu yer değiştirelim.
   Tmp=A(:, 3) ;

   A(:, 3) = A(:,1) ;

   A(:,1)= Tmp ;

Yada aynı işlemi başka bir ifade ile

```
A= [ A( :, 3 )  ;  A(2, : ) ;  A( : , 1 ) ; A(4, : ) ; A(5, : ) ] ;
```

ilede yapabiliriz.

3. A matrisinin 2.satırının 2.ci elemanından sonraki elemanları (2,3,4,5,6. cı elemanları) ile A matrisinin 4. Sütunun 2.ci elemanından sonraki elemanlarını (2,3,4,5,6. cı elemanları) yerdeğiştirirsek,

```
Tmp=A(2,2:6) ;

A(2,2:6) = A(2:6, 4);

A(2:6, 4)=Tmp;
```

Genel olarak NxN boyutlu bir A matris üzerinde elemanter düzende işlem yapacagimiz zaman,

A(:,a) ile a sütununu (tüm satırın)' na ait elemanları

A(a,:) ile a satırının(tüm sutunların)' na ait elemanlarını

A(a:b,:) ifadesi ile tüm a ile b satırlarını içeren tüm sutun elemanlarını

A(:,a:b) ilede a ile b sütunları arasındaki tüm satır elemanları

üzerinde işlem yapıyor oluruz.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
A=[3 2; 2 1]
[V,D]=eig(A) % A matrisinin özdeger ve özvektörlerini bulur

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Ax=b denklem sisteminin Cholesky yöntemiyle çözümü

```
a=[1 1 -3; 2 -1 3; -2 2 -1] %Katsayý dizeyi
b=[-11;8;1]              % bilinen vektör
[L,U]=lu(a);             % alt ve üst üçgen dizeyler
z=inv(L)*b;
x=inv(U)*z;              % aranan sonuç çözüm vektörü
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Ax=b denklem sisteminin EÞHELON (birim matrise indirgeme) yöntemiyle çözümü

```
c=[a b]
d=rref(c);
x=d(:end); % burada end ile d matrisinin son kolonu görüntülenir.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

1. Jacobi Yöntemi
2. Gauss-Seidel Yöntemi

1. Jacobi yöntemi Matlab Kodu:

```
function x=jacobi(A, b, N)
% Jacobi(A, b, N) sistemi çözümü
%
% A katsayı dizeyi
% b sağ kolon vektörü
% N yineleme sayısı
%
% baslangıç vektörü sıfır alınmıştır.
% Jacobi dönüşüm dizeyi A = inv(D)*(L+U)
%                    c= inv(D)*b.
%
n = size(A,1);
L = zeros(n);
U = zeros(n);

tol = 1e-05;
k = 1;
x(:,1) = zeros(n,1);          %baslangıc vektör


%A dizeyini L, U ve D dizeylerine bol
for i=1:n
  if A(i,i)==0
    disp('sifir kosegen elemanı ile karşılaşıldı')
    return
  else
    D(i,i) = A(i,i);
  end;
end;

for i=2:n
  for j=1:i-1
    L(i,j)= - A(i,j);
  end;
end;

for i=1:n-1
  for j=i+1:n
    U(i,j)= - A(i,j);
  end;
end;
T = inv(D)*(L+U);
c = inv(D)*b;

while k <= N
```

```
  x(:,k+1) = T*x(:,k) + c;

  if ( norm(x(:,k+1)-x(:,k)) < tol )
    disp('islem tamam')
    x = x'
    return
  end;
  k = k+1;
end;

if ( norm(x(:,k)- x(:,k-1)) > tol) | (k > N)
  disp('yineleme tamamlandi fakat istenen hata düzeyine ulaşılamadı:')
  x = x';
end;
```

Örnek:

```
>>a =

   4   0   1   1
   0   4   0   1
   1   0   4   0
   1   1   0   4

>>b =

   1
   2
   3
   4

>>jacobi(a,b,10)
>>
ans =
      0       0       0       0
  0.2500  0.5000  0.7500  1.0000
 -0.1875  0.2500  0.6875  0.8125
 -0.1250  0.2969  0.7969  0.9844
 -0.1953  0.2539  0.7813  0.9570
 -0.1846  0.2607  0.7988  0.9854
 -0.1960  0.2537  0.7961  0.9810
 -0.1943  0.2548  0.7990  0.9856
 -0.1962  0.2536  0.7986  0.9849
 -0.1959  0.2538  0.7990  0.9856
 -0.1962  0.2536  0.7990  0.9855
```

2. Gauss-Seidel Matlab kodu:

```
function x=gauss_seidel(A, b, N)
```

```
% Gauss_seidel(A, b, N) çözümü

% A katsayı dizeyi
% b sag vektör
% N yineleme sayısı
%
% Tg = inv(D-L)*U
% cg = inv(D-L)*b.
%
n = size(A,1);
L = zeros(n);
U = zeros(n);

tol = 1e-05;
k = 1;
x = zeros(n,1);                    % başlangıç vektörü
%L, U and D dizeyleri
%
for i = 1:n
  if A(i,i) == 0
    disp('sifir diagonal eleman')
    return
  else
    D(i,i) = A(i,i);
  end
end
for i = 2:n
  for j = 1:i-1
    L(i,j) = -A(i,j);
  end
end
for i = 1:n-1
  for j = i+1:n
    U(i,j) = -A(i,j);
  end
end

T = inv(D-L)*U;
c = inv(D-L)*b;

while k < N
  x(:,k+1) = T*x(:,k) + c;

  if norm(x(:,k+1)-x(:,k)) < tol
    disp('işlem tamam')
    x = x'
    return
  end
  k = k+1;
end
```

```
if norm(x(:,k)- x(:,k-1)) > tol | k > N
  disp('yineleme tamamlandi fakat istenen hata düzeyine ulaşılamadı:')
  x = x';
end;
```

Örnek:

```
>>a =

    2    1    1
    2    3    1
    3    2    5

>>b =

    4
    6
   10

>> gauss_seidel(a,b,10)

ans =

       0         0         0
   2.0000    0.6667    0.5333
   1.4000    0.8889    0.8044
   1.1533    0.9630    0.9228
   1.0571    0.9877    0.9707
   1.0208    0.9959    0.9891
   1.0075    0.9986    0.9961
   1.0027    0.9995    0.9986
   1.0009    0.9998    0.9995
   1.0003    0.9999    0.9998
```