ТЕХНИЧЕСКИ УНИВЕРСИТЕТ-СОФИЯ
ЕЛЕКТРОТЕХНИЧЕСКИ ФАКУЛТЕТ
КАТЕДРА ЕЛЕКТРОЕНЕРГЕТИКА

Лабораторни упражнения

по

УСТОЙЧИВОСТ НА ЕЛЕКТРОЕНЕРГИЙНИТЕ СИСТЕМИ

Рад Христов Станев

София, 2008 г.

Съдържание

Лабораторно упражнение № 1

КРАТКО ВЪВЕДЕНИЕ В MATLAB

Добре дошли в лаборатория 12/401. Това е първото лабораторно упражнение по устойчивост на електроенергийните системи (УЕЕС). Неговата цел е бързо да Ви въведе в средата на Matlab. Това лабораторно упражнение съдържа основните операции и функции, които ще Ви бъдат необходими, за да можете да решите поставените Ви задачи при всички следващи лабораторни упражнения. При всяко следващо лабораторно упражнение е необходимо носенето на:

- настоящия свитък и
- записките от семинарните упражнения по УЕЕС.

Няколко думи за *MATLAB*

Тенденциите в развитието на техниката и математиката през последните двадесет години утвърдиха софтуерният пакет ***MATLAB***, създаден от **Math Works Inc**. като водещ софтуер и език за технически изчисления. ***MATLAB*** разполага с множество математически методи, програмирани като готови функции, които позволяват решаването на широк кръг от задачи от областта на матричната алгебра, комплексната аритметика, линейните системи, диференциалното и интегрално смятане, нелинейните системи, оптимизацията и др. Сам по себе си ***MATLAB*** е обектно ориентиран език за програмиране. Основен обект в този език е матрицата.

Предимството на ***MATLAB*** се състои в това, че той разполага с готови решения за най- често срещаните математически задачи. Също така, поради гъвкавата си структура, при създаването на нов метод в дадена област на математиката, скоро след публикуването му, потребителите на този метод, които най-често не са математици, могат бързо и лесно да ползват метода, без да се налага те да минават през тежката задача сами да вникнат в същността на метода, да създадат компютърна програма, да изпитат тази програма, да отстранят допуснатите грешки при програмирането и след това да го сравнят със съществуващите досега методи.

Заедно с това ***MATLAB***, чрез своята мощна програма за симулация на динамични системи ***SIMULINK***, дава възможност на ползвателите, които не се занимават с математика и програмиране, но все пак имат известни познания в съответната област на техниката, чрез достъпния интерфейс на ***SIMULINK*** лесно и ефективно да моделират и симулират сложни процеси в различни линейни и нелинейни системи.

Изложените предимства и перспективи, които ***MATLAB*** предлага пред другите среди за програмиране, са причина именно той да бъде избран за среда, в която ще бъдат провеждани лабораторните упражнения по УЕЕС.

1. Пускане на Matlab

Matlab в Windows се стартира чрез двукратно щракане на левия бутон на мишката върху иконата на Matlab. 

Следва зареждане на командния прозорец (Command Window) и след няколко встъпителни съобщения като intro, demo, help help и т.н. се изобразява знака “**>>**”. Този знак указва, че програмата е готова за работа и очаква нашите команди да бъдат записани в командния ред, който се намира непосредствено след знака “**>>**”. Matlab е организиран в интерактивен команден режим и подобно на разговор след като зададем определена команда, програмата извежда резултата от изпълнението на командата отново в командния прозорец. Например, ако в командния ред запишем командата **`help`** и след това натиснем клавиша Enter, в командния ред ще бъде изведен списък на файловете, функциите и операторите, за които има помощ. Ако запишем:

`help` *име на функцията*

ще бъде изведена информация за функцията и за нейното използване.

Символите, които следват знака “%” се игнорират и всичко записано след този знак не се взема под внимание от програмата. По този начин могат да се въвеждат обяснителни коментари.

Matlab може да изпълнява поредица от команди, които се записват във файлове с разширение ".m", които ще наричаме за краткост Script- файлове или M-файлове. За да създадем такъв файл е необходимо да щракнем на **File** -> **New** -> **M-File**. Това отваря прозореца за редактиране (Edit Window), в който се пише програмата (или поредицата от команди). Когато решим да изпълним написаната програма е необходимо:

- да укажем на Matlab да работи с директорията, в която сме записали нашия M-File.
- да запишем името на създадения от нас M-File в командния ред (непосредствено след знака “>>”) и да натиснем клавиша Enter.

2. Променливи

Променливите в Script-файловете (M-файловете) по подразбиране са глобални.

Имената на променливите се задават като комбинация от букви и цифри. Изисква се името на променливата да започва с буква и освен това всички букви трябва да са латински (използването на кирилица е възможно само за въвеждане на пояснителни коментари). Ако например в командния ред запишем:

```
x = 2.5*sin(pi/2)
```

ще бъде изведен резултата:

```
x =
     2.5
```

Десетичният делител е символът “**.**”. По подразбиране всички ъглови аргументи, които се въвеждат или извеждат от програмата Матлаб са в радиани!

Ако даден израз е записан в командния ред или в дадена програма, без да е зададено име на променливата, Matlab изчислява, запазва и извежда резултата от този израз като променлива наименована **`ans`**. Например въвеждането на:

```
12*5
```

ще изведе резултата:

```
ans =
     60
```

Ще отбележим, че в Matlab се прави разлика между главни и малки букви, т.е. Matlab е *case sensitive* по подразбиране. Така например променливите F и f, които са

записани с главна и съответно с малка буква ще бъдат възприети от Matlab като две различни променливи.

Ако искаме да изчислим даден израз, но не желаем резултатът от него да бъде изведен в командния прозорец, е необходимо след израза да поставим знака точка и запетая ";".

3. Векторни операции

- **Създаване на вектор-ред**

За да създадем един вектор- ред трябва да напишем елементите му в средни скоби и да ги отделим с интервали. Елементите на вектора могат да бъдат, както числа, така и изрази. Така например резултатът от въвеждането на:

```
R = [  tan(pi/4)   sqrt(9)   -5 ]
```

е

```
R = 1.0000   3.000   -5.0000
```

Разделянето на елементите на вектора- ред може да стане и със запетаи.

- **Създаване на вектор-стълб**

За да създадем един вектор- стълб, трябва да напишем елементите му в средни скоби и да ги отделим с точка и запетая. Така например резултатът от въвеждането на:

```
X = [  2; -4; 8]
```

е

```
X =
     2
     -4
     8
```

Разделянето на елементите на вектора- стълб може да стане и с натискане на клавиша Enter.

- **Транспониране**

Транспонирането на един вектор- ред води до получаване на вектор- стълб и обратно. Например:

```
Y = R'
```

дава

```
Y =
     1.0000
     3.0000
     -5.0000
```

- **Операции с матрици и операции с масиви**

В Matlab се прави разлика между операции с матрици (матрични операции) и операции с масиви. Матричните операции се извършват по правилата на матричната алгебра, докато операциите с масиви се извършват елемент по елемент. Матричното умножение на два вектора X и Y, всеки от които има по n- на брой елемента, се определя като $\sum_{i=1}^{n} x_i y_i$. Така матричното произведение на векторите X и Y, дефинирани по горе се дава с

```
S = X' * Y
```

и резултатът е

```
S =
     -50
```

Чрез оператора `.*` се задава умножение елемент по елемент. Така например

```
E = X .* Y
```

Дава

```
E =
     2
     -12
     -40
```

Аналогично чрез оператора ./ се задава деление елемент по елемент, чрез оператора .^ се задава степенуване елемент по елемент и т.н.

● Създаване на някои характерни вектори

Нулев вектор:

```
Z = zeros(1, 4)
```

дава

```
Z =
     0   0   0   0
```

Единичен вектор:

```
I = ones(1, 4)
```

дава

```
I =
     1   1   1   1
```

Вектор с нарастващи целочислени елементи:

```
x = 1:8
```

дава

```
x =
     1   2   3   4   5   6   7   8
```

4. Основни матрични операции

В Matlab една матрица се задава като правоъгълен масив от стойности, записани между среди скоби. Елементите във всеки ред се отделят с интервали или запетаи. За означаване на края на всеки ред се ползва знакът **;** или се натиска клавишът Enter. Например:

```
A = [ 6   1   2;   -1   8   3;   2   4   9  ]
```

дава матрицата

```
A =
     6   1   2
     -1  8   3
     2   4   9
```

● Извличане на редове и стълбове от матрица

За да извлечем даден ред от една матрица се ползва символа **:** . Той означава „всички елементи от зададения ред или стълб". Например, ако искаме да извлечем трети ред от матрицата A, можем да запишем:

```
r3 = A(3, : ),
```

което дава

```
r3 =
     2    4    9
```

Аналогично, за да извлечем втори стълб на A трябва да напишем `A(:, 2)`.

● Матрично умножение и деление

Ако броят на редовете на една матрица $\mathbf{A}$ е равен на броя на стълбовете на друга матрица $\mathbf{B}$, тези две матрици могат да бъдат умножени и да бъде получено тяхното произведение $\mathbf{AB}$. Разбира се ще припомним, че разместителното свойство не е в сила при матрично умножение и деление. За матрично умножение се ползва знака (*), а за матрично деление се ползва (/) . Така $\mathbf{A}\backslash\mathbf{B}$ е равносилно на $\mathbf{A}^{-1}\mathbf{B}$, а съответно $\mathbf{A}/\mathbf{B}$ е равносилно на $\mathbf{AB}^{-1}$.

Задача. 1

Дадена е система линейни уравнения, която се описва еднозначно чрез матричното уравнение $\mathbf{AX} = \mathbf{B}$, разписано по-долу:

$$\begin{vmatrix} 4.x_1 & -2.x_2 & -10.x_3 = & -10 \\ 2.x_1 & +10.x_2 & -12.x_3 = & 32 \\ -4.x_1 & -6.x_2 & +16.x_3 = & -16 \end{vmatrix} \Leftrightarrow \underbrace{\begin{bmatrix} 4 & -2 & -10 \\ 2 & 10 & -12 \\ -4 & -6 & 16 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}}_{\mathbf{X}} = \underbrace{\begin{bmatrix} -10 \\ 32 \\ -16 \end{bmatrix}}_{\mathbf{B}},$$

където с $\mathbf{X}$ е означен векторът на неизвестните x_1, x_2 и x_3.

Да се реши уравнението по отношение на $\mathbf{X}$, като се използва оператора за матрично деление (\).

Решение:

Въвеждаме:

```
A = [ 4   -2   -10;   2   10   -12;    -4   -6   16];
B = [ -10;   32;   -16];
X = A\B
```

и изведеният резултат е:

```
X =
     2.0000
     4.0000
     1.0000
```

● Обръщане на матрица

Решаването на една система линейни уравнения може да стане чрез операцията обръщане на матрица.

Задача. 2

Дадено е матричното уравнение $AX = B$ от зад.1. Да се реши уравнението по отношение на X, като се използва оператора за обръщане на матрица **inv** .

Решение:

Въвеждаме:

```
A = [ 4   -2   -10;   2   10   -12;    -4   -6   16];
B = [ -10;   32;   -16];
X = inv(A)*B
```

и изведените резултати са:

```
X =
     2.0000
     4.0000
     1.0000
```

● Създаване на някои характерни матрици

За създаването на матрица с **m** реда и **n** стълба е необходимо да се зададе вида на матрицата и нейната размерност **(m,n)**.

Нулева матрица:

`N = zeros(m,n)` създава матрица с нулеви елементи и размерност mxn

Единична матрица:

`E = ones(m,n)` създава матрица с единични елементи и размерност mxn

Диагонална матрица:

`D = diag(x)` създава диагонална матрица чиито елементи са елементите на вектора **x (x** трябва да е дефиниран по-горе от **D)**.

5. Работа с комплексни числа

За работа с комплексни числа в Matlab са запазени комплексните оператори **i** и **j**. Така например:

```
Z = 10 + j*20
```

дава

```
Z =
     10.0000 +20.0000i
```

Синтаксисът, който се ползва за операциите с компексни числа е аналогичен със синтаксиса за операциите за реални числа. Например:

```
Y = 1/Z
```

дава

```
Y =
     0.0200 - 0.0400i
```

За да бъдат извлечени реалната и съответно имагинерната съставка на комплексната матрица `Z` се ползват командите `real(Z)` и `imag(Z)`. За да бъдат извлечени модулът и фазата на `Z` могат да се използват функциите `abs(Z)` и съответно `angle(Z)`.

6. Тригонометрични функции

При изчисленията в редица случаи ще бъде удобно използването на някои основни тригонометрични функции, дадени втаблиците по-долу. Напомняме, че всички ъглови аргументи са в радиани!

Функция	синтаксис	Функция	синтаксис
синус	`y=sin(x)`	аркус синус	`y=asin(x)`
косинус	`y=cos(x)`	аркус косинус	`y=acos(x)`
тангенс	`y=tan(x)`	аркус тангенс	`y=atan(x)`
котангенс	`y=cot(x)`	аркус котангенс	`y=acot(x)`

7. Създаване на графики

В работата ни често ще се налага да представяме графично, някои от по-важните резултати. Ще покажем, как се създава една двумерна графика на функцията y=f(x), чиито променливи по двете оси са X и Y:

```
x = [ 6  3   6   4  4.1  4   5   4  0  0  9   9  4 ];
y = [11  5  5.5  2   3   2  2.5  2  2  12 12  2  2 ];
plot(x,y) , grid
xlabel('x'), ylabel('y'), title('Нашата първа графика')
```

Броят на елементите от двата вектора трябва да е един и същ!

Ако на една и съща графика желаем да начертаем повече от една функция на дадена променлива, например графиките на функциите y1=f1(x1), y2=f2(x2), y3=f3(x3), ... , записваме:

```
plot(x1,y1, x2,y2, x3,y3,… )
```

8. Цикли и логически условия

Матлаб предлага възможности за програмиране чрез изразите **`for, while`** и **`if`**. Чрез **`for`** даден израз се изпълнява, докато се достигне до **`end`**, последователно, толкова пъти, колкото е определено отнапред. Така например, написвайки:

```
for i = 1:n
     израз
end
```

даденият *израз* ще бъде изпълнен **n** на брой пъти.

Чрез **`while`** даден израз се изпълнява неограничен брой пъти, докато се достигне до удовлетворяване на определено, предварително зададено условие.

Чрез **`if, elseif`** и **`else`** даден израз се изпълнява тогава, когато е изпълнено дадено условие.

Матлаб има 6 релационни и 4 логически оператора , които са дадени в таблицата по долу.

Релационни оператори	*Логически оператори*
== равно	& и
~= различно	| или
< по-малко	~ логическо добавяне
<= по-малко или равно	xor изключващо или
> по-голямо	
>= по-голямо или равно	