Math 323
Linear Algebra and Matrix Theory I
Fall 1999

Dr. Constant J. Goutziers
Department of Mathematical Sciences
goutzicj@oneonta.edu

Lesson 1
Introduction to MATLAB

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows easy solution of many technical computing problems, especially those with matrix and vector formulations. For many colleges and universities MATLAB is the Linear Algebra software of choice. The name MATLAB stands for matrix laboratory.

1.1 Row and Column vectors, Formatting

In linear algebra we make a clear distinction between row and column vectors.

- Example 1.1.1
  Entering a row vector.
  The components of a row vector are entered inside square brackets, separated by blanks.

```
v=[1 2 3]

v =
     1     2     3
```

- Example 1.1.2
  Entering a column vector.
  The components of a column vector are also entered inside square brackets, but they are separated by semicolons.

```
w=[4; 5/2; -6]

w =
    4.0000
    2.5000
   -6.0000
```

- Example 1.1.3
  Switching between a row and a column vector, the transpose.

In MATLAB it is possible to switch between row and column vectors using an apostrophe. The process is called "taking the transpose". To illustrate the process we print **v** and **w** and generate the transpose of each of these vectors. Observe that MATLAB commands can be entered on the same line, as long as they are separated by commas.

```
v, vt=v', w, wt=w'

v =
     1     2     3
vt =
     1
     2
     3
w =
    4.0000
    2.5000
   -6.0000
wt =
    4.0000    2.5000   -6.0000
```

- Example 1.1.4
  Formatting.

You have probably already observed that the second component **w** was outputted as 2.5 and not as 5/2 as you might have hoped for. MATLAB is essentially numerically, not symbolically oriented. However there are three format options which can be used to your advantage. They are: **format long,** for the accuracy hungry; **format short,** the default for normal use; **format rat**(ional), the favorite of students. We illustrate all three.

```
format long
x=[2, 5/3, -131/107]

x =
   2.00000000000000   1.66666666666667  -1.22429906542056

format short
x

x =
    2.0000    1.6667   -1.2243

format rat
x

x =
      2            5/3       -131/107
```

The way you decide to print your output does not affect the accuracy of your computations. However, before you get to enthusiastic about the rational format, I should warn you that it merely produces a rational approximation of the underlying floating point number, **it does not represent an exact representation**. The latter is easily demonstrated if we ask the software to compute $\sqrt{2}$.

```
test=sqrt(2)

test =
  1393/985
```

Of course this is not mathematically correct, since √2 is irrational, but the underlying floating point number is a very good approximation for the square root of 2.

```
format long, test^2

ans =
   2.00000000000000
```

1.2 Computation with Vectors. Vector Addition; Scalar Multiplication; Linear Combinations; Norms and Dot Products

- Example 1.2.1
  Vector addition.

Vector addition in MATLAB is very intuitive. Just make sure that the vectors are both row vectors, or both column vectors, and have the same number of components.

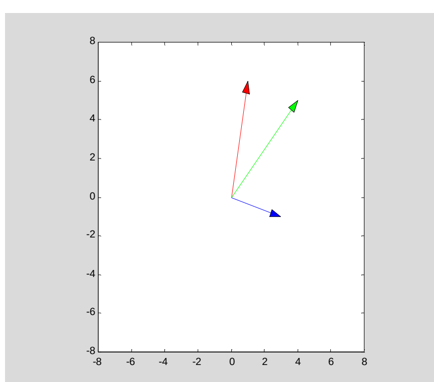
```
v=[1 6], w=[3 -1], s=v+w

v =
     1     6
w =
     3    -1
s =
     4     5
```

Of course the sum s can be graphically obtained from v and w by using the parallelogram rule, as is illustrated in the following picture.

```
drawvec(v, 'red', 8);, hold on, drawvec(w, 'blue', 8);, hold on,
drawvec(s, 'green', 8);
```

The semi colons suppress the unwanted last numerical output of the subroutine drawvec.



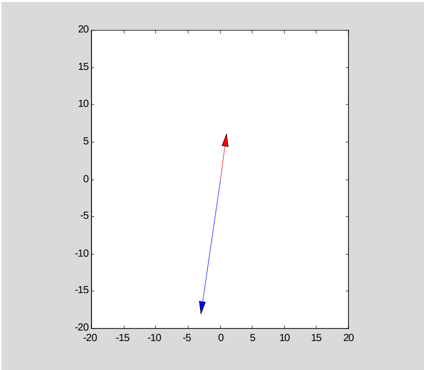
- Example 1.2.2

  Scalar multiplication.

  Scalar multiplication in MATLAB is just as intuitive as vector addition.

```
v, w=-3*v, drawvec(v, 'red', 20);, hold on, drawvec(w, 'blue', 20);
```

```
v =
     1     6
w =
    -3   -18
```



Observe that -3**v** is a vector three times as long as **v** itself and pointing in the opposite direction. Vector addition and scalar multiplication are **the** two important operations in linear algebra.

- Example 1.2.3

  Linear Combinations.

  By combining the operations of **vector addition** and **scalar multiplication**, we can now form expressions of the form c**v** +d**w.** Expressions of this type play a fundamental role in linear algebra and are known as **linear combinations**.

```
v=[1 2 3], w=[-3 5 -2], lc= 2*v-5*w
```

```
v =
     1     2     3
w =
    -3     5    -2
lc =
    17   -21    16
```

- Example 1.2.4

  Suppression of output.

  Sometimes you may want to perform a computation without displaying its result. That can be done by ending your command with a semicolon.

```
x=17^(1/3);
test=x^3
```

```
test =
   17.0000
```

- Example 1.2.5

  The length or norm of vectors.

Built into MATLAB is the so called p norm. For a vector **v** in n space it is the p-th root of the sum of the p-th powers of the absolute values of the components of **v**. We typically use the 2-norm: $||\mathbf{v}|| = \sqrt{(|v_1|^2 + |v_2|^2 + \ldots + |v_n|^2)}$. When working within the set of the **R**eal numbers, the absolute values are of course redundant and therefore usually omitted.

```
v=[1 2 -2], norm(v, 2)

v =
     1     2    -2
ans =
     3
```

- Example 1.2.6

  Dot products.

From calculus you are familiar with the dot product of two vectors. We have the **dotprod** command at our disposal.

```
v=[-2 1 3], w=[5 2 1], dp=dotprod(v, w)

v =
    -2     1     3
w =
     5     2     1
dp =
    -5
```

A dot product also can be obtained as the product of a row and a column vector. Please be advised that order is important, the product of a column and a row vector is **not a dot product**, next week you will learn the details of these operations. We now just illustrate this way of computing a dot product at the hand of the vectors **v** and **w**.

```
v, w, v*w'

v =
    -2     1     3
w =
     5     2     1
ans =
    -5
```

1.3 Saving the MATLAB Workspace on a floppy disk

MATLAB allows the user to save the workspace, that is the collection of names and values of all the variables you have used in a particular session. This gives you the opportunity to retain your work even if you get interrupted and have to continue at a later time. To save the workspace on a floppy disk, go to the file dropdown menu and click on **Save Workspace As**, then save the workspace on the a:\ drive in a file with a .mat extension. **Warning: Place the file in the root**

directory of the a:\ drive and make sure that your filename does not contain blanks. To retrieve the workspace once you start a new session, simply go back to the file menu, click on Load Workspace and load the appropriate .mat file from your floppy disk.