Notes Set 1

CSS 455
Winter 2011

Find a Partner

- You will sign up as a group at the class break.

This set (week1)

- A few general introductory points
- Web Site Review
- Syllabus Review
- GA1, HW0
- General Matlab Issues
- Vector implementation in Matlab and simple plots
- Floating point error issues

General Matlab issues

- Interpreted language, but with considerable efficiency for vector and array operations
- Frequent choice for engineering and science problems of modest proportion.
- Mostly procedural – although objects are available. *What I do is mostly procedural.*
- Double precision throughout

General Matlab issues

- Main program is at command line or in an *m-file* starting with commands
- Subprograms are functions:
  - Primary functions – *m-files also, with name same as function.*
  - Subfunctions – follow primary functions in m-file
  - Nested functions – nested within primary functions
  - Private functions – primary functions with restricted access.

General Matlab issues

- Data structures and cell arrays are available
- User input: *input(), inputdlg, menu, ginput*
- User output: *disp, sprintf, fprintf,*
- Data import/export: spreadsheets, etc
- *csvread/csvwrite; dlmread/dlmwrite; fread/fwrite;* etc
- *tic, toc, profile*

General Matlab issues

- Array indexes *start with 1* *(not zero)!!!*
- 2-D arrays are indexed as A(row,col)
- 2D arrays can also be indexed linearly, as they are stored by column order.
- Will do activity to determine variable scope (global or local) and function scope (visibility)
- Will do activity to determine if arguments are passed by reference or by value.

General Matlab issues

- Really easy: no typing (all numbers are double precision, *regardless of format*)
- Semicolons supress output rather than terminate statements. Statements must be continued (…)
- Usual control logic (for loops, if-else blocks, case , while loops.
- Variable names are case sensitive

- In physics, a vector is a construct with both magnitude and direction.
- In linear algebra a vector is a column (row) of numbers.
- The latter is a representation of the former.
- Think about both 2- and 3- dimensional vectors and about n-dimensional generalizations in considering these notes.

Notation

- Vectors are often denoted by bold faced lower case symbols, such as **a**, or by singly underlined symbols, such as <u>a</u>. *(Matlab does not distinguish)*
- In 3D Euclidean space, the *unit vectors* in the *x, y, and z* directions are denoted, respectively, as {**i, j, k**} or {$\mathbf{e}_1$, $\mathbf{e}_2$, $\mathbf{e}_3$}.
- The vector **a** with components a_x, a_y, and a_z is expressed as $\mathbf{a} = a_x\mathbf{i} + a_y\mathbf{j} + a_z\mathbf{k}$.

Notation

- In column notation, **a** is expressed as:

$$\mathbf{a} = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} \text{ or } \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

- In row notation, **a** is expressed as:

$$\mathbf{a} = (a_x \; a_y \; a_z) \text{ or } (a_1 \; a_2 \; a_3)$$

Vector definitions in Matlab

```
» a = [1 5 7]
a =
     1     5     7
» b = [ 1; 5; 7]
b =
     1
     5
     7
```

- Notice square brackets for vector definition.
- Semicolons separate the rows of the vector.

Vector definitions in Matlab

```
» a = [1 5 7]

a =

     1     5     7

» b = [ 1; 5; 7]

b =

     1
     5
     7
```

```
>> a(1)

ans =

   1

>> b(2)

ans =

   5
```

•The vector index starts at 1, not 0.

Vector definitions in Matlab

```
EDU>> c = [1 15 20];
EDU>> c

c =

    1   15   20

EDU>>
```

•Trailing semicolons supress output

- Unit Vectors

$$\mathbf{e}_1 = \mathbf{i} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

$$\mathbf{e}_2 = \mathbf{j} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$$

$$\mathbf{e}_3 = \mathbf{k} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

- n - dimensional vectors

$$\mathbf{a} = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

Transposition interconverts row and column vectors.

- Transpose operator is denoted by superscript T or by prime indicator:
- Transpose of **x** is $\mathbf{x}^{\mathrm{T}}$ or $\mathbf{x}'$

$$\mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} \qquad \mathbf{b}^{\mathrm{T}} = \mathbf{b}' = \begin{pmatrix} b_1 & b_2 & \cdots b_n \end{pmatrix}$$

Transposition interconverts row and column vectors.

- Transpose of **x** is $\mathbf{x}^{\mathrm{T}}$ or $\mathbf{x}'$

$$\mathbf{c} = \begin{pmatrix} c_1 & c_2 & \cdots c_n \end{pmatrix} \qquad \mathbf{c}^{\mathrm{T}} = \mathbf{c}' = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{pmatrix}$$

Matlab uses single quote for transposition

```
» b = [1 5 8 2]

b =

     1     5     8     2

» btr = b'

btr =

     1
     5
     8
     2
```

```
» c = [5;3;4;10]

c =

     5
     3
     4
    10

» ctr=c'

ctr =

     5     3     4    10
```

Vector Operations

- Addition

$$\mathbf{c}=\mathbf{a}+\mathbf{b}=\begin{pmatrix}c_1\\c_2\\\vdots\\c_n\end{pmatrix}=\begin{pmatrix}a_1+b_1\\a_2+b_2\\\vdots\\a_n+b_n\end{pmatrix}$$

- Subtraction

$$\mathbf{c}=\mathbf{a}-\mathbf{b}=\begin{pmatrix}c_1\\c_2\\\vdots\\c_n\end{pmatrix}=\begin{pmatrix}a_1-b_1\\a_2-b_2\\\vdots\\a_n-b_n\end{pmatrix}$$

- Multiplication by constant λ:

$$\lambda\mathbf{a}=\begin{pmatrix}\lambda a_1\\\lambda a_2\\\vdots\\\lambda a_n\end{pmatrix}$$

Vector Addition

```
>> a = [4;6;8;21]

a =

     4
     6
     8
    21

>> b = [1;3;2;5]

b =

     1
     3
     2
     5

>> c = a+b

c =

     5
     9
    10
    26
```

```
>> x=a'

x =

     4     6     8    21

>> y=b'

y =

     1     3     2     5

>> z=x+y

z =

     5     9    10    26
```

Vector Subtraction

```
a =

     4
     6
     8
    21

>> b

b =

     1
     3
     2
     5

>> c=b-a

c =

    -3
    -3
    -6
   -16
```

```
>> x

x =

     4     6     8    21

>> y

y =

     1     3     2     5

>> z=x-y

z =

     3     3     6    16
```

```
a =

     5
     3
    21
     4

>> s=3

s =

     3

>> b=s*a

b =

    15
     9
    63
    12
```

```
a =

    15
     6
     4

>> s

s =

     3

>> v = a/s

v =

    5.0000
    2.0000
    1.3333
```

Scalar Multiplication /division

Vector Magnitudes

- In ordinary two- and three- dimensional space, the magnitude of a vector is given by its Euclidean Norm

$$|\mathbf{a}| = \sqrt{a_1^2 + a_2^2 + a_3^2}$$

In *n* dimensions*:*

$$|\mathbf{a}| = \left(\sum_i^n a_i^2 \right)^{1/2}$$

Vector Magnitudes

- In ordinary two- and three- dimensional space, the magnitude of a vector is given by its Euclidean Norm

$$|\mathbf{a}| = \sqrt{a_1^2 + a_2^2 + a_3^2}$$

This is an example of the more general concept of a *vector norm.*

Vector Norms

- If **x** is an *n*-dimension vector

$$\mathbf{x} = (\mathrm{x}_1 \;\; \mathrm{x}_2 \cdots \mathrm{x}_n)$$

P=2 is the usual Euclidean Norm

- Then the *p*-norm of **x** is given by:

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |\mathrm{x}_i|^p\right)^{1/p} = \left(|\mathrm{x}_1|^p + |\mathrm{x}_2|^p + \cdots + |\mathrm{x}_n|^p\right)^{1/p}$$

The summation operator Σ will be encountered frequently.

Vector Norms - measure of size or "length" of vector

- p-norm

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |\mathrm{x}_i|^p\right)^{1/p}$$

- 2-norm (Euclidean)

$$\|\mathbf{x}\|_2 = \left(\sum_{i=1}^{n} |\mathrm{x}_i|^2\right)^{1/2}$$

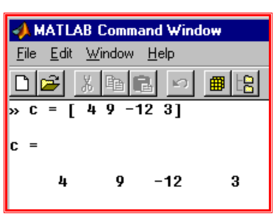
- 1-norm

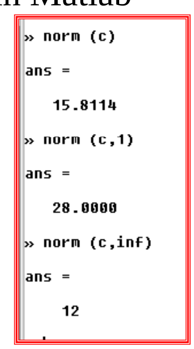
$$\|\mathbf{x}\|_1 = \left(\sum_{i=1}^{n} |\mathrm{x}_i|\right)$$

- ∞-norm

$$\|\mathbf{x}\|_\infty = \max_i |\mathrm{x}_i|$$

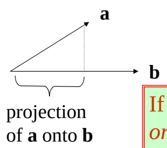
Vector Norms in Matlab





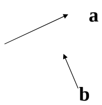
Scalar (dot or inner) Product

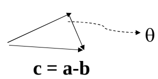
- In physics it is useful to define the scalar product of two vectors:
  a·b = |**a**| |**b**| *cos(θ), where θ is the angle between* **a** *and* **b.**



If **a** and **b** are *perpendicular* or *orthogonal*, their scalar product is *zero.* (Cos 90º = 0)

Law of Cosines from Trig





$$|\mathbf{c}|^2 = |\mathbf{a}|^2 + |\mathbf{b}|^2 - 2|\mathbf{a}||\mathbf{b}|\cos(\theta)$$

$$|\mathbf{c}|^2 = |\mathbf{a}|^2 + |\mathbf{b}|^2 - 2\mathbf{a}\cdot\mathbf{b}$$

$$\mathbf{a}\cdot\mathbf{b} = \tfrac{1}{2}\left\{|\mathbf{a}|^2 + |\mathbf{b}|^2 - |\mathbf{c}|^2\right\}$$

$$\boxed{|\mathbf{a}|^2 = \left[\left(\mathbf{a}_1^2 + \mathbf{a}_2^2\right)^{1/2}\right]^2 = \mathbf{a}_1^2 + \mathbf{a}_2^2}$$

- Substitute component expressions for vector magnitudes:

$$\mathbf{a}\cdot\mathbf{b} = \tfrac{1}{2}\left[\left(a_1^2 + a_2^2\right) + \left(b_1^2 + b_2^2\right) - \left(c_1^2 + c_2^2\right)\right]$$

•**c = a - b,** so $c_1 = a_1 - b_1$ and $c_2 = a_2 - b_2$

$$\mathbf{a}\cdot\mathbf{b} = \tfrac{1}{2}\left[\left(a_1^2 + a_2^2\right) + \left(b_1^2 + b_2^2\right) - \left(\left(a_1 - b_1\right)^2 + \left(a_2 - b_2\right)^2\right)\right]$$

$$\mathbf{a}\cdot\mathbf{b} = \tfrac{1}{2}\left[\left(a_1^2 + a_2^2\right) + \left(b_1^2 + b_2^2\right) - \left(a_1^2 + b_1^2 - 2a_1b_1\right) - \left(a_2^2 + b_2^2 - 2a_2b_2\right)\right]$$

$$\boxed{\mathbf{a}\cdot\mathbf{b} = a_1b_1 + a_2b_2}$$

The inner (scalar, dot) vector product is generalized to *n*-dimensions:

- If **a** and **b** are both *n* dimension column vectors.

$$\mathbf{a}^T = \mathbf{a}' = (a_1\ a_2 \ldots a_n) \qquad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

$$\mathbf{a} \cdot \mathbf{b} = \mathbf{a}^T\mathbf{b} = \mathbf{a}'\mathbf{b} = a_1b_1 + a_2b_2 + \cdots + a_nb_n = \sum_{i=1}^{n} a_ib_i$$

- Note that by convention the first vector of the product is a *row* vector and the second is a *column* vector. The transpose is taken if necessary to assure this.
- This convention is consistent with the matrix multiply operation we will introduce.

$$\mathbf{a} \cdot \mathbf{b} = \mathbf{a}^T\mathbf{b} = (a_1\ a_2 \ldots a_n)\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

Scalar (inner) product in Matlab

```
a =

     1     5     9

» b

b =

     3
     2
     7

» a*b

ans =

    76
```

There are other ways to do this in Matlab, using the element-by-element multiply (**.***) and the sum function (sum)

Note: the elementary multiply operator becomes a scalar product when both operands are vectors.

Product Operator

The product operator (*) is overloaded. It's operation depends upon the nature of the operands.

```
>> s1=3;
>> s2=5;
>> a = [1 5 9];
>> b = [3;2;7];
```

Initialization:

Scalar times scalar

```
>> s1*s2
ans =
   15
```

Product Operator

The product operator (*) is overloaded. It's operation depends upon the nature of the operands.

```
>> s1=3;
>> s2=5;
>> a = [1 5 9];
>> b = [3;2;7];
```

Initialization:

Scalar times vector

```
>> s1*a
ans =
   3   15   27
```

Product Operator

The product operator (*) is overloaded. It's operation depends upon the nature of the operands.

```
>> s1=3;
>> s2=5;
>> a = [1 5 9];
>> b = [3;2;7];
```

Initialization:

Scalar times vector

```
>> s1*b
ans =
    9
    6
   21
```

Product Operator

The product operator (*) is overloaded. It's operation depends upon the nature of the operands.

```
>> s1=3;
>> s2=5;
>> a = [1 5 9];
>> b = [3;2;7];
```

Initialization:

```
>> a*b
ans =
   76
```

Vector times vector

Matlab "pointwise" operators

- The operators ".*" and "./" operate on each element of the two vector operands and place the result in corresponding element.

c = a .* b, then
c(1) = a(1)*b(1)
c(2) = a(2)*b(2), etc.

c = a ./ b, then
c(1) = a(1)/b(1)
c(2) = a(2)/b(2),
etc.

The operands must be vectors of same dimension, and the result is also a vector that dimension.

Pointwise Multiplication

```
a =

     1     5     9    10

» b

b =

     2     3     7    11

» c=a.*b

c =

     2    15    63   110
```

Pointwise Division

```
a =

     1     5     9    10

>> b

b =

     2     5     3     2

>> c=a./b

c =

    0.5000    1.0000    3.0000    5.0000
```

Activity 1

Orthogonality and normalization in terms of the scalar product

- **a** and **b** are ***orthogonal*** if $\mathbf{a}^{\mathrm{T}}\mathbf{b} = \mathbf{0}$
- Unit vectors are one example of an orthogonal set

$$\mathbf{i}\cdot\mathbf{j} = \mathbf{e}_1\cdot\mathbf{e}_2 = \begin{pmatrix}1 & 0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix} = 0$$

- **a** is ***normalized*** if its magnitude is unity:

$$|\mathbf{a}| = \sqrt{\mathbf{a}^{\mathrm{T}}\mathbf{a}} = 1$$

$$\mathbf{a}^{\mathrm{T}}\mathbf{a} = 1$$

Unit vectors are normalized:

$$|\mathbf{i}| = \sqrt{\mathbf{i}\cdot\mathbf{i}} = \sqrt{\mathbf{i}^{T}\mathbf{i}} = \left[\begin{pmatrix}1 & 0\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}\right]^{\frac{1}{2}} = 1$$

Vectors as arguments

- When a built-in function operates on a vector, it produces a vector of results in Matlab

  If **c** = sin(**x**), then

  c(1) = sin(**x**(1))

  c(2) = sin(**x**(2)), etc.

- You can vectorize in-line functions with the *vectorize* function.

Vector argument for *exp*

```
>> v = [0 1 2 3]

v =

     0     1     2     3

>> z = exp(v)

z =

    1.0000    2.7183    7.3891   20.0855
```

e^0 e^1 e^2 e^3

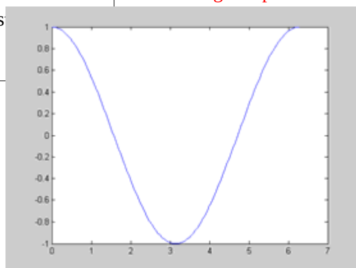
Use Vectors for Plotting in Matlab

- To plot the function $f(x) = cos(x), x=0 - 2\pi$
- First: set up a vector **x** of the independent variable closely spaced over the range.
- Second: Evaluate a vector **y** of the dependent variables: (**y** = *f*(**x**))
- Third plot the curve using *plot(**x**,**y**)*

```
xlo=0;
xhi = 2*pi;
nstep = 120;
x = linspace(xlo,xhi,ns
y = cos(x);
plot (x,y,'-')
```

Curve is smooth because of using 120 points.

Plot of cosine function.

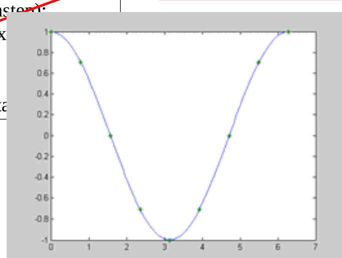


Here, the vectors are simply data structures and do not have physical meaning.

```
xlo=0;
xhi = 2*pi;
nstep = 100;
x = linspace(xlo,xhi,nstep);
xdata = [xlo:0.25*pi:x
y = cos(x);
ydata = cos(xdata);
plot (x,y,'-',xdata,ydata
```

- What is xdata?
- How many elements?

some data points



One vector contains the selected data points. (xdata, ydata)
The other contains the many points to give a smooth curve. (x,y)

```
plot (x,y,'-',xdata,ydata,'*')
title ('Cos(x) vs x')
xlabel('X')
ylabel ('Cosine
```

- These functions have parameters to control size, fonts, etc.

Include labels