ENGINEERING 5

Lecture 4: Matrix Transformations, MatLab Scripts & Functions, Serial Communication

Professor Carr Everbach

Course web page:
http://www.swarthmore.edu/NatSci/Class/ceverba1/e5/E5Index.html

E5…

- Comments on Response Readings
- Thursday 9/22: Matlab I is due.
- Thursday 9/29: MatLab II lab is due.

- Wizards available Tuesdays/Wednesdays 7:00-9:00 pm (Hicks 213) for Engin (Matlab help) and Sundays/Tuesdays 7:00-9:00 pm (SC 128) Physics & Math

- This week in lab: make things out of stuff. If time, start assembling servos into robotic arm.

From *Elektronik*
(a Swedish Engineering Magazine)

Control of flow in MatLab (Ch 4)

- So far we have used MatLab as a fancy calculator
- The real power comes when we can have MatLab:
  - Perform operations many times consecutively:
    - `for… loop`
  - Make decisions:
    - `if…then…else…`
    - `if…then…elseif…then…elseif… … …else…`
    - `while… loop`

The for… loop (1)

Syntax:

```
for variable=start:finish
  …commands…
end
```

Example:

```
for i=1:10
  disp(i)
end
```

Output:

```
1
2
3
4
5
6
7
8
9
10
```

The for… loop (2)

Syntax:

```
for variable=start:increment:finish
  …commands…
end
```

Example:

```
for i=1:4:10
  disp(i)
end
```

Output:

```
1
5
9
```

Making Decisions in Matlab: `if`

Syntax:

```
if expression,
   … statements …
end
```

expression has to evaluate to TRUE or FALSE

Example:

```
if a>b,
   disp('a is greater than b');
end
```

```
if (a>b) & (a>0),
   disp('a>b and a is positive');
end
```

Logical expressions

expression has to evaluate to TRUE or FALSE

a>b	TRUE when a is *greater* than b, otherwise FALSE
a<b	TRUE when a is *less* than b, otherwise FALSE
a>=b	TRUE when a is *greater* than or equal to b, otherwise FALSE
a==b	TRUE when a is *equal* to b, otherwise FALSE
a~=b	TRUE when a is ***not*** equal to b, otherwise FALSE
(a>0) & (b>0)	TRUE when a is greater than 0 ***and*** b is greater than zero, otherwise FALSE
(a>0) | (b>0)	TRUE when a is greater than 0 ***or*** b is greater than zero, otherwise FALSE
(a>0) | ~(b>0)	TRUE when a is greater than 0 ***or*** b is ***not*** greater than zero, otherwise FALSE

Be careful if a or b are matrices, surprising results can occur!

Note: this list is not exhaustive, check you text for more.

Making Decisions: `if … else`

Syntax:

```
if expression,
   … statements …
else
   … statements
end
```

Example:

```
if a>b
   disp('a is greater than b');
else
   disp('a is less than or equal to b');
end
```

Making Decisions: `if … elseif … else`

Syntax:

```
if expression1
   … statements …
elseif expression2
   … statements …
else
   … statements …
end
```

Example:

```
if a>b
   disp('a is greater than b');
elseif a<b
   disp('a is less than b');
else
   disp('a is equal to b');
end
```

Making Decisions: `while`…

Syntax:

```
while expression
    … statements …
end
```

Example:

```
i=1;
while i<10
   disp(i);
   i=i+4;
end
```

Output:

```
1
5
9
```

Note: this is equivalent to the `for…loop` used earlier

Functions

- For our purposes a function is a special MatLab file that does a calculation(s) and returns a result
- It is different from a script file:
  - it returns a result
  - it can’t change other variables that you are using

Anatomy of a Function

Keyword "function" tells us this is a function file (called "perim.m")

Output variable(s) name(s); in this case only one: the variable "p"

Function name; also name of ".m" file

Input variables

```
function p = perim(xvals,yvals)

x1=[xvals xvals(1)];%Form augmented vector
y1=[yvals yvals(1)];

xd=diff(x1);        %Calculate differences
yd=diff(y1);

sqrd = xd.^2+yd.^2; %Square them
sqrtSqd=sqrt(sqrd); %Find distances

p=sum(sqrtSqd); %Find perimeter
```

Calculations

Output variable (must be the same variable name as is in function declaration)

Using Functions

Now we can call function as if it was built in:

```
>> q=perim(x,y)
q =    6.8284
```

A trick: if you add comments to the very top of the function file, they will appear if you ask for documentation (i.e., "`>>doc perim`").

Caveats

- Function name must not be the same as a variable name
- Function name must have no spaces in it
- Function must be in MatLab directory so MatLab can find it.
- If you edit a function, you must save the file before the changes will take effect in subsequent calls
- If you edit a function, you must save the file before the changes will take effect in subsequent calls

Serial Communication in Matlab

- Matlab communicates through the serial port via fprintf and fscanf commands writing to a "serial port object."
- s = serial('COM1'); % creates a serial port object
- set(s, 'Baudrate', 38400); % sets baud rate of COM1
- fprintf(s, '#0P1500'); % writes text in single quotes to COM1 serial port output
- a = fscanf(s); % reads text from COM1 serial port and puts the result in array 'a'

Matlab Demo…