Matlab Programming

a logical approach

Outline

- Announcements:
  - Homework I: due Wed. by 5, by e-mail
  - Remember: text & subject
  - Last day to add/drop or change credit/audit
- Iteration
- Conditionals and logic
- M-files

A Short Pep-Talk

- Some facts:
  - Computers are dumb.
  - No, actually, they're really dumb
  - Computers can only do a few stupid things, but
    - They can do them over & over again very quickly without tiring or complaining
- Some conclusions
  - You are way smarter than your computer--don't be intimidated
  - If you have to do something several times--get the stupid computer to do it.

Iteration

- For loops in Matlab use index-notation:

```
for j=st:step:en;
    <commands involving j> ;
end
```

- Example: inner product u'*v

```
tot=0;
for j=1:length(u);
    tot=tot+u(j)*v(j);
end
```

Conditionals

- Conditional statements control flow of a program

```
if(logic);
    <commands executed if true>;
else;
    <commands executed if false>;
end
```

- Matlab also has switch:

```
switch(j);
   case a: <commands if a==j>;
   case b: <commands if b==j>; …
   otherwise: <default commands>;
end
```

Logic

- Relational operators: (R x R -> B)
  - <, >, <=, >=, ==, ~=
  - isnan(a), isinf(a)
- Logical operators: (B x B -> B)
  - &, |, ~, xor (which is just ~=)

a	b	a&b	a|b	xor(a,b)	a~=b
T	T	T	T	0	0
T	0	0	T	T	T
0	T	0	T	T	T
0	0	0	0	0	0

Logic

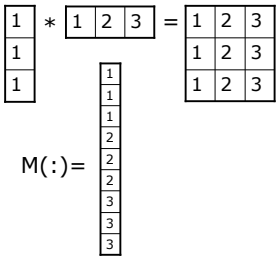
- Matlab's Boolean type is a logic array
  - Logical operators work with doubles, too
  - 0=> false; anything else => true
- Logical operators are defined for arrays:
  - a=1:5;
    - a = [1 2 3 4 5]
  - b=a<3
    - b= [1 1 0 0 0]

Logic--searching with find

- Find--searches for the truth (or not false)
  - b=[1 0 -1 0 0];
  - I=find(b)
    - I=[1 3] --b(I) is an array without zeros
  - I=find(b<1)
    - "b<1" -->[0 1 1 1 1];
    - I=[2 3 4 5]
  - I=find(b>1)
    - "b>1"-->[0 0 0 0 0];
    - I=[]

Find for matrices

- M=ones(3,1)*[1 2 3];
  - I=find(M==2)
    - I= 4 5 6
    - treats M as M(:)!
  - [I,J]=find(M==2)
    - I= 1 2 3
    - J= 2 2 2



Programming

- Matlab programs are stored in "m-files"
  - Text files containing Matlab commands and ending with .m
  - Script m-files--commands executed as if entered on command line
  - Function m-files--analogous to subroutines or methods, maintain their own memory (workspace)

Scripts are Evil

- Scripts are EXTREMELY poor programming
  - Application specific, so difficult to reuse
  - Dangerous: variables created in scripts can overwrite existing variables
- If you send me a script, I will send it back!

Functions

- First line of file must be
  function [outputs]=fname(inputs)
  - outputs and inputs can have 0,1, or many variables
  - Ex: [U,V]=SSHvel(SSH, lat,lon)
    - Requires 3 inputs (referred to in SSHvel.m as SSH, lat, lon)
    - Returns 2 outputs. Whatever U and V are when SSHvel finishes are returned to the workspace (or calling function)
    - >> [locU,locV]=SSHvel(locSSH, loclat,loclon)

Generic Function Format

```
function [U,V]=SSHvel(SSH,lat,lon);
%SSHvel.m-----Computes velocity from ssh-su
%
%[U,V]=SSHvel(SSH,lat,lon);
%
%SSH should be an m-by-n array of SSH obser
%The east-west spacing of the grid is defin
%north-south spacing is defined by lat (len

[m,n]=size(SSH);
p=length(lat);
if(p~=m);error('Length of lat must equal th
p=length(lon);
if(p~=n);error('Length of lon must equal th

I=1:m-1;J=1:n-1; %index vectors for differe
```

1. Function statement
2. 1st comment--used by lookfor
3. Comments returned by help SSHvel--Should indicate how to call, and describe what it does
4. Error checking--check that sizes are correct/consistent. Return a helpful message with error command
5. Code

Example: Geostrophy

- We want to create a function to compute velocity from SSH data

Summary

- Iteration with for (while exists too)
- Conditionals with if and switch
- Logical operators and vectorized
- 0=false
- find searches for truth
- Extend matlab with m-file functions
  - [output]=fname(input)