Matlab Programming

Outline

• Announcements:
  – Homework 1: 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:  
  ```matlab
  for j=start:step:end;
    <commands involving j> ;
  end
  ```
- Example: inner product $u^\ast v$  
  ```matlab
  tot=0;
  for j=1:length(u);
    tot=tot+u(j)*v(j);
  end
  ```

**Conditionals**

- Conditional statements control flow of a program  
  ```matlab
  if(logic);
    <commands executed if true> ;
  else;
    <commands executed if false> ;
  end
  ```
- Matlab also has switch:  
  ```matlab
  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)  
  - $<,>,<\leq,\geq,\leq,\geq$  
  - isnan(a), isinf(a)
- Logical operators: (B x B -> B)  
  - $\&$, $\lor$, $\neg$, xor (which is just $\neg=)$

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>$&amp;$</th>
<th>$\lor$</th>
<th>xor(a,b)</th>
<th>$\neg=a$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>0</td>
<td>0</td>
<td>T</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>T</td>
<td>T</td>
<td>0</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
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];
    - I=[2 3 4 5]
    - I=find(b>1)
      - "b>1"--->[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(:)
  - [1,2]=find(M==2)
    - I= 1 2 3
    - J= 2 2 2
    - M(:,):=
**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
  ```matlab
  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**

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

```
function [U,V]=SSHvel(SH,lat,lon)

SSH should be an m-by-n array of SSH data
The east-west spacing of the grid is defined by lon.
North-south spacing is defined by lat

[x,y]=size(SH);
[px,wy]=size(lon);
[py,xn]=size(lat);
if px==x
error('Length of lon must equal to size of SSH data')
end
if py==y
error('Length of lat must equal to size of SSH data')
end
l=1:m;1:n=1;1:m-1; Waven vectors for differ
```

**Example: Geostrophy**

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

```
SSH  \rightarrow  SSHvel.m \rightarrow  U  V
```

**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)