Computing for Math and Stats

Lecture 9.

Conditional execution

- We often want our program to decide whether to execute one command or another (or none)
  - Example: The absolute value of a real number x is x if positive and -x if negative
- Matlab (like any other language) has conditional statements
- Conditional statements check a condition and if true execute a group of statements, otherwise execute another group of statements (or no statements)

Conditional Statements

- Absolute value of x
  - if x>=0
    - absx=x;
  - else
    - absx=-x;
  - end

Conditions

- Conditions can be simple like
  - x>0
- We often need quite complex conditions
- We have a whole set of relational operators
  - They are: >, <, >=, <=, ==, ~=
- We also have a set of logical operators
  - They are: and() [&], or() [|], not() [~]

Conditions

- When we want both x and y to be positive
  - x>0 & y>0
- When we want at least one of them to be positive
  - x>0 | y>0
- When we want none of them to be positive
  - x<=0 & y<=0   or...
  - ~(x>0 | y>0)
- There can be more than one way to express a condition. Usually one of them is less computationally expensive

What is True?

- This is not a philosophy question
- What do we get when we run
  - A = x>2
- In Matlab 1 is true and 0 is false
  - Question: what does Matlab think of numbers other than 0 or 1?
  - This is a Matlab (and C and a few other) convention
  - In other systems/languages we have different conventions
    - T and nil in Lisp
    - 0 and non-zero in bash
    - True and False in other systems
- It is defined so for convenience (not philosophy)

Precedence

- When we have a numerical expression
  - a+b*c
- We all understand it as
  - a+(b*c)
- Old calculators understood it as
  - (a+b)*c
- Matlab understands it the way we all do
- See weird.m

Precedence

- Matlab precedences are as follows
  - Parentheses
  - Exponentiation
  - Logical not (~)
  - Multiplication, division
  - Addition, subtraction
  - Relational ops. (<, >, etc)
  - Logical and (&)
  - Logical or (|)

How to Remember Precedences

- Hard
- Even if you do, the one that reads your code may not
- The golden rule:
  - When in doubt, parenthesize
- Different languages may have different (very slightly) precedences
- One thing to remember is that Matlab has weird precedence for not (~)

Other Logical Operators

- In theory there are 16 logical operators (with two operants/arguments)
- Most of them are not needed in programming
- We need some others that take as operand/argument a vector

Other Logical Operators

- These are
  - xor(a,b)
    - Exclussive or
  - all(A)
    - Returns true (1) if all elements of A are true
  - any(A)
    - Returns true (1) if any element of A is true
  - find(A)
    - Returns the indices for which A is true (non zero)

The “if” statement

- Conditional statements use conditional expressions
- The most common conditional statement is the if statement aka if ... end
- Allows a block of statements to be executed or not executed depending on a condition.

The simplest version of if...end

- We use this to execute or not execute a group of statements
  - if <some condition>
    - Stmnt 1
    - Stmnt 2...
  - end
- We can have as many statements in the block as we want

The if-then-else form

- We use this to choose between two (or more) blocks of statements
- if <condition>
  - Stmnt1
  - Stmnt2
- elseif <condition>
  - Stmnt5...
- else
  - Stmnt3
  - Stmnt4
- end
- See piecewise.m

Indentation

- Blocks of statements whithin the if-else (ot the if-end or the else-end, etc), have to be indented
- Indentation means prepending a “few” blanks to statements within enclosed blocks
- “few” means usually 2-4 blanks. Asways the same number
- Real editors do it automatically
  - The Matlab editor does a pretty good job.
- Indentation helps with code readability
- Lack of indentation screams “Amateur”