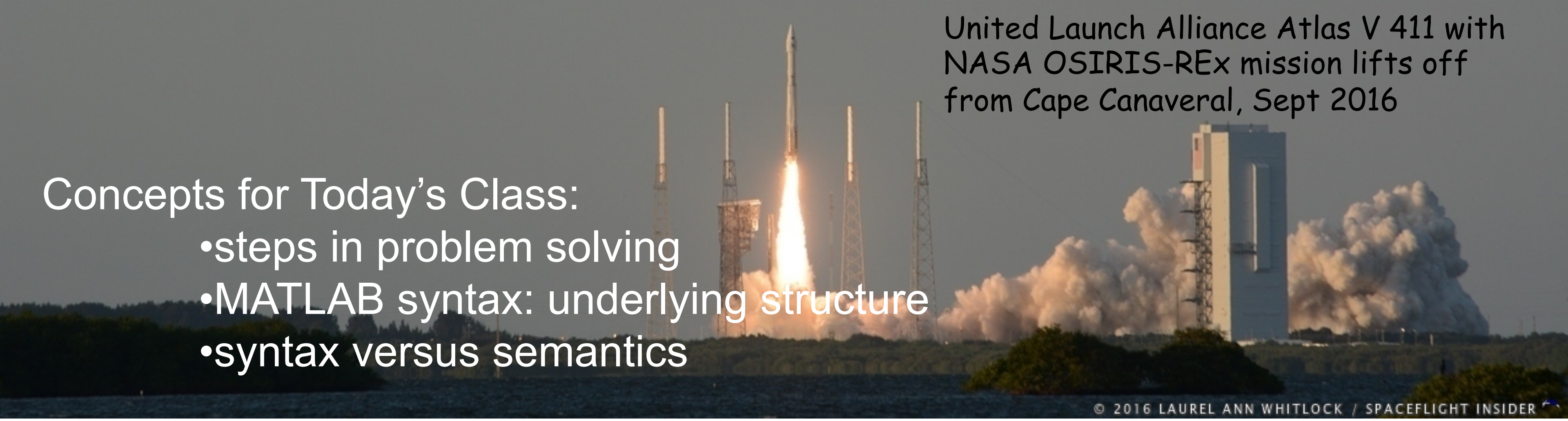


United Launch Alliance Atlas V 411 with NASA OSIRIS-REx mission lifts off from Cape Canaveral, Sept 2016

© 2016 LAUREL ANN WHITLOCK / SPACEFLIGHT INSIDER

Concepts for Today's Class:

- steps in problem solving
- MATLAB syntax: underlying structure
- syntax versus semantics

Reading:

- See course web site for details, we'll cover material in:
- pgs 6-40 (in class)
- pgs 3-6, 75-80, 89-101 (labs)
- pgs 557-564 (general reference, will show up in classes)

Lab:

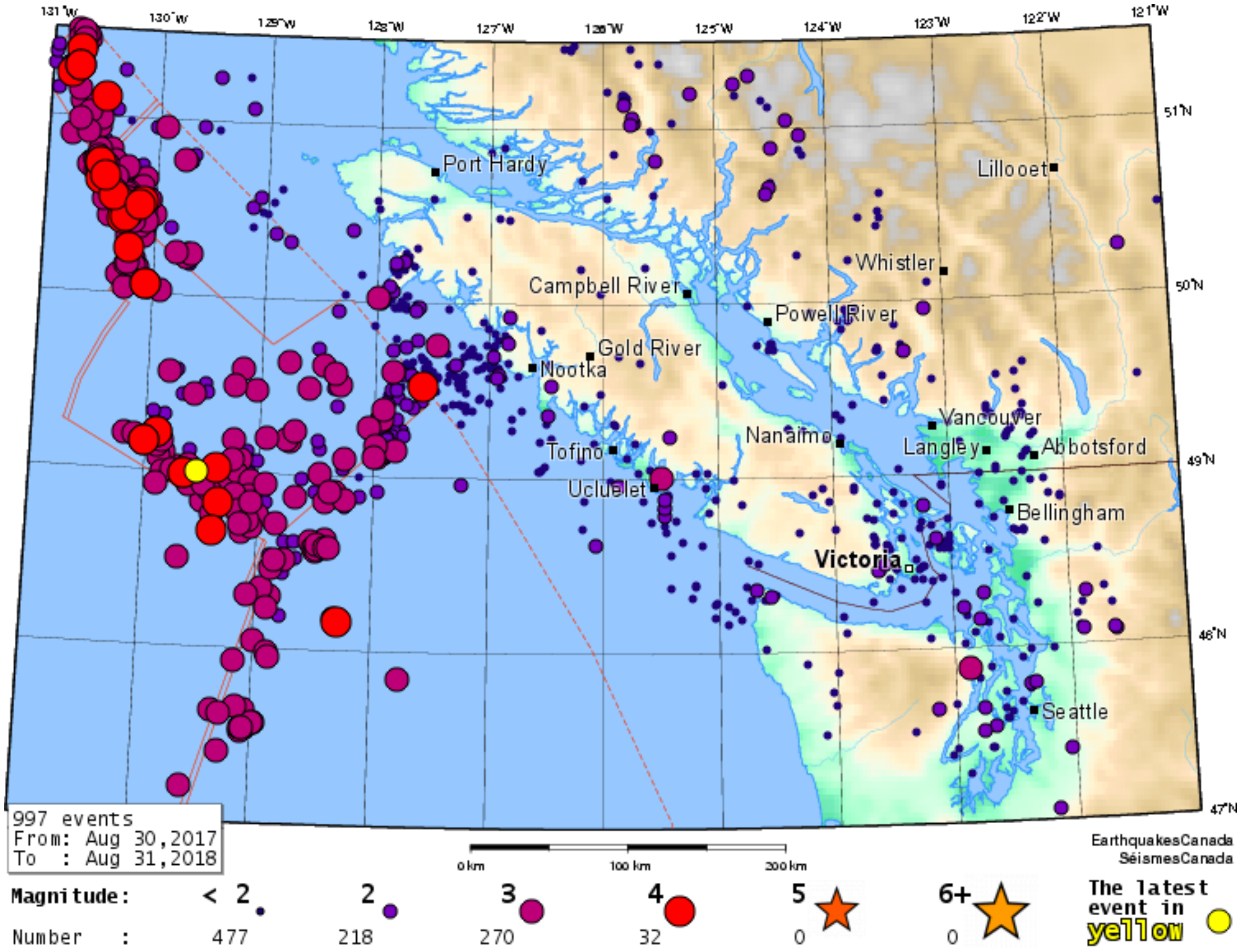
- bring text
- Labs due at **4pm on Fridays**

TA hours, EOS-Main 203 – make sure to note which weeks these carefully!

Friday (wks 3,4,5,6,8,9,10,11): 9:30 am – 10:30 am

Monday (wks 7,10 **only**): 11:00 am – noon.

Good Problem Solving Techniques



Good Problem Solving Techniques

We will adopt the 5-step program **(NOTE: this is more complete than the approach in the text book on page 76)**

1. State the problem clearly

2. Describe your input and output info / data

3. Write down the problem-solving procedure IN WORDS
   - pay special attention to the logic
   - if possible work the problem "by hand" for a simple set of data

4. Develop a MATLAB code to do (3)

5. Test your code with a variety of data or different cases

Example: What was the largest earthquake in the Vancouver / Vancouver Island region in the past year?

MATLAB: Programming elements (overview)

The structure of MATLAB

1. Lexical Elements

   - building blocks

Programming language
names, operators, special characters

Human Language
words, symbols, punctuation

2. Syntax

   - set of rules to combine lexical elements into legitimate constructs

Programming language
how to build expressions, statements, functions & programs

Human Language
how to build sentences & paragraphs

Lexical Elements (Building Blocks)

1. Names
   a) Variable names: store data: e.g., our user-defined variable: mag1
   b) Reserved words: have a special meaning, cannot be redefined; e.g. for
   c) Function names: sequences of code; e.g. sin

2. Operators
   a) Logical: e.g., && text p. 20
   b) Arithmetic: e.g., +, -, *, / text p. 12-13
   c) Relational: e.g., >, >=, <, <= text p. 19

3. Special characters: keyboard characters with special meaning in MATLAB.
   e.g., ; {} ()

Lexical Elements (Building Blocks)

1. Names: variable names, reserved words, function names
2. Operators: logical, arithmetic, relational
3. Special characters: keyboard characters with special meaning in MATLAB.

Lexical Elements (Building Blocks)

How are variables assigned in MATLAB?

```
>> mag1 = 4.9
```

What does this do?

1. Creates a space in memory
2. Gives the space the name “mag1”. We call mag1 a VARIABLE
3. Stores the value 4.9 in that space (variable)
4. Echoes back this information
5. The prompt reappears

```
>> mag1 = 3.7            overwrites “4.9” with “3.7”
```

Note

- “=“ does not mean = in the mathematical sense. Instead, it means “assign the number on the RHS to the variable on the LHS”

- We can do math on the RHS, mixing numbers and variables

Syntax

Lexical elements are combined using the set of rules known as “syntax”

I cdnuolt blveiee that I cluod aulacity uesdnatnrd what I was rdgnieg.

⇒Important difference between human and programming languages is tolerance to syntax errors versus tolerance to spelling/grammar errors

Examples of Syntax Errors

Usually / often your code will not run if you make a syntax error:

```
>> x=10;                                defined my variable x and assigned it a value of 10


>> y=cos(x                              trying to compute cos(x) and assign it to variable y
??? y=cos(x                             Correct: y = cos(x)
           |
Error: Expression or statement is incorrect--possibly unbalanced (, {, or [.



>> y=2x                                 trying to compute “2 times x” and assign it to “y”
??? y=2x                                Correct: y = 2*x (spaces around = don’t matter)
       |
Error: Unexpected MATLAB expression.
```

Syntax versus Semantics

When a program doesn’t work this can happen because:

a) SYNTAX is incorrect - e.g.,

```
>> distance = speed *
```

- multiplication (*) needs numbers or variables on both sides of *

b) SEMANTICS are incorrect

```
>> distance = speed / time
```

- may or may not run but will certainly give you the wrong answer!! The program you wrote is not the program you meant to write!

can reduce both types of errors using good problem solving techniques

Wrap Up

Steps in problem solving

1. state problem clearly
2. define the input/output
3. write down algorithm by hand: ie., think the problem through
4. code in MATLAB
5. test, test, test…

Underlying structure of MATLAB syntax

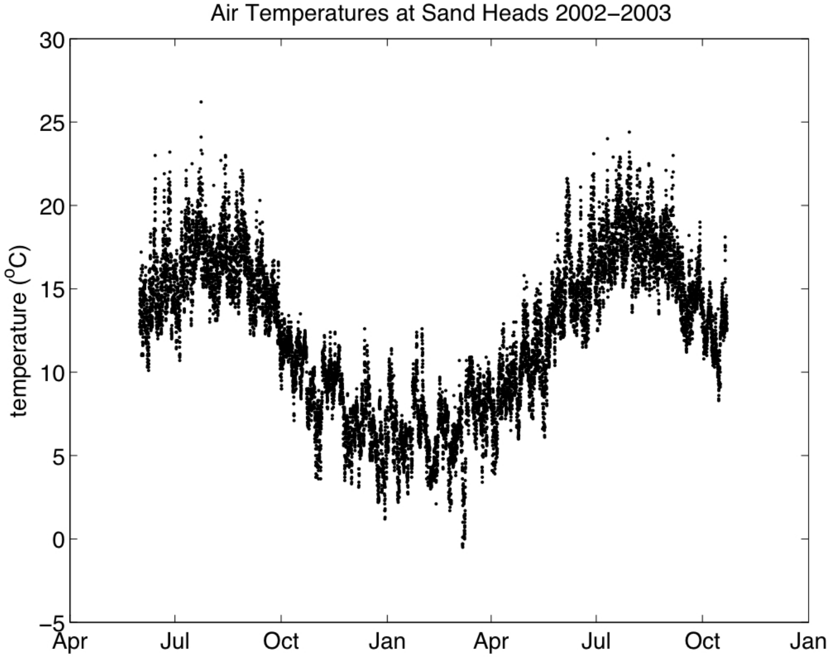
- building blocks: names, operators, special characters

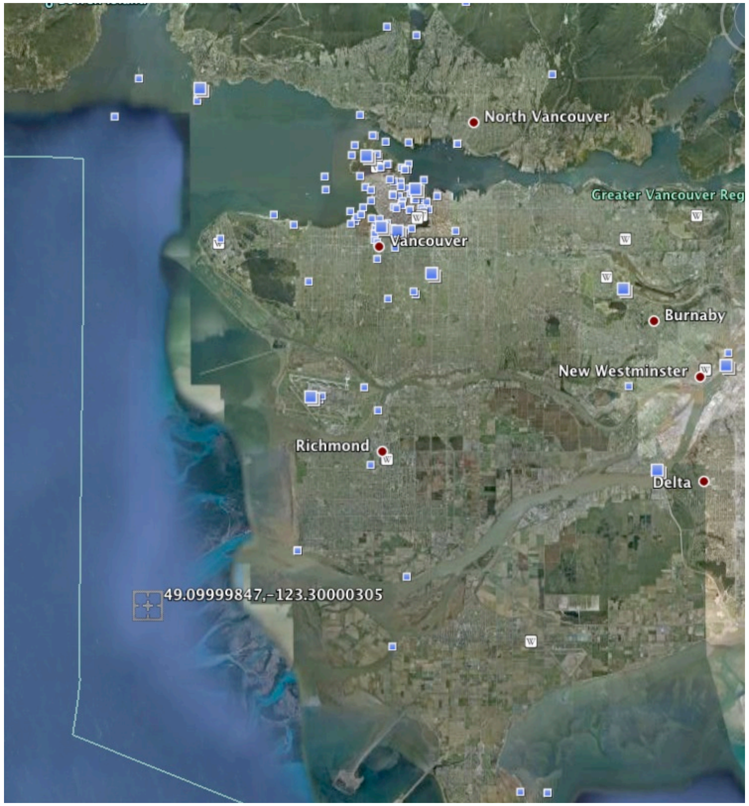
Syntax versus semantics

- syntax errors are about bugs in step 4 above
- semantics errors usually result from skipping step 3

Lab Overview

- become familiar with MATLAB environment
- load, plot and save data
- try some simple operations
- write MATLAB scripts





a) pair programming
b) need either an account, or your own version of MATLAB
c) turn in requested file on Canvas (not graded but will use later in term)