

You don't look like Prof. Jose-Luis Jimenez...

Ingrid Ulbrich
Chem 5181
29 August 2009



Some Course Information

- **No syllabus today**
 - Jose will distribute next week
 - Read it carefully!
- **Work will include homework, labs, exams, final lab project, “journal skims,” in-class journal presentation**
 - Stay on top of the work and it's manageable

A Note about Jose's Style

From a past syllabus:

“If you don't go through the effort of understanding and solving every problem yourself (after discussing it with others if you want) you'll find yourself at a large disadvantage in the exams where you have to *solve new problems* very quickly.”

Clicker Q's 1&2

I'm a student in...

- a) Analytical Chemistry
- b) Physical Chemistry
- c) another division of Chemistry
- d) Engineering
- e) Another department

I'm in my ____ year of grad school.

- a) First
- b) Second
- c) Third
- d) > Third

Goals for First 2 Lectures (Ingrid)

- **Become familiar with Igor Pro software**
 - Storing data
 - Doing calculations
 - Making graphs
 - Writing simple functions
- **Short(ish) homeworks to work on these tasks**
 - Start with examples that should already be familiar
 - “Model” one MS component in last homework

Why do I have to use Igor instead of software I already know?

- **Powerful analysis package**
 - Superior to spreadsheets! Especially good for
 - Reproducible and traceable calculations
 - Graphs with more options that are easier to manipulate
 - Cheaper than Matlab
 - Good user-community mailing list, excellent email support from Igor developers
- **Used almost exclusively in Jimenez, Tolbert, and Volkamer groups for data analysis**
- **Good experience for learning programming**

Why should I learn programming? (1)

- Functions create a record of the steps used in a calculation
 - Help find errors in multistep calculations
- Functions make it easy to repeat the same calculation with different data
 - Standardized analysis

Why should I learn programming? (2)

- **Functions are easier to share with others than in a spreadsheet**
 - **Repeatable calculations for your colleagues**
- **Slice and dice big datasets**
 - **Aerosol Mass Spectrometer saves 500 mass spectral values every 2.5 minutes**
 - 10^4 values per hour $\rightarrow 3 \times 10^5$ per day $\rightarrow 2 \times 10^6$ per week
 - **Way too much data to look at everything by hand!**

Clicker Q's 3-6

My experience with spreadsheets
(e.g., Excel, etc.) is...

- a) Nil
- b) **Limited – I can do basic math, but not calculate complex functions**
- c) **Strong – I can calculate complex functions, make graphs, do fits, and modify graphs as I'd like (colors, line style, axis labels, etc.)**
- d) **Extensive – I can do the things above and am familiar with built-in functions and using Help.**

My experience with Igor, Matlab, IDL,
or other similar programs is...

- a) Nil
- b) Limited – I've worked with these programs, but don't feel very confident.
- c) Moderate – I can get around programs like this, but I'm not an expert.
- d) Extensive with non-Igor software – I just have to learn Igor's tweaks
- e) Extensive with Igor – I'm a pro

My experience with programming
(in any computational language/script) is...

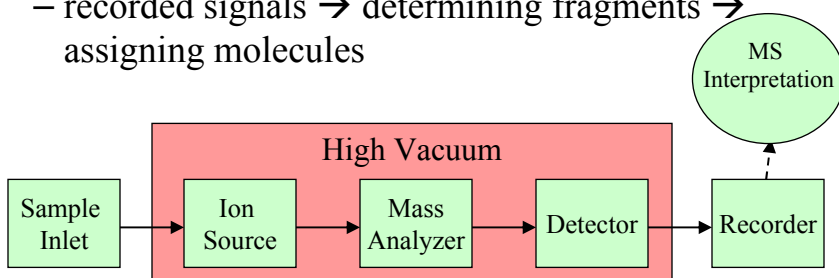
- a) Nil
- b) Limited – I can write simple functions
- c) Strong – I can decide how to break a problem into pieces that should be functions and can write nested functions.
- d) Extensive – I can code anything I want to.

My previous data analysis work mainly used...

- a) Pencil and paper
- b) Spreadsheets
- c) Non-Igor analysis packages
- d) Igor

Mass Spectrometry Overview

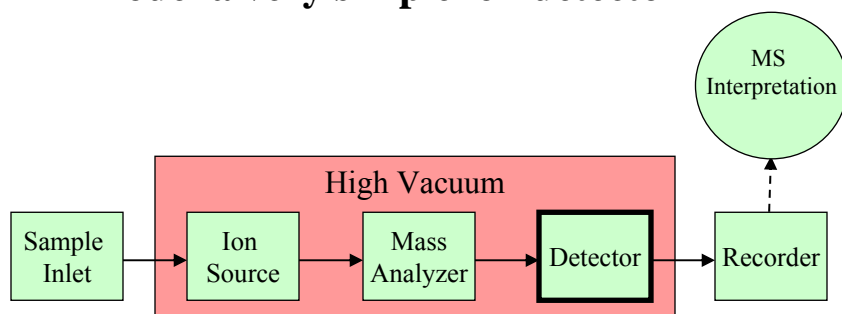
- Everything in the mass spectrometer is *Physics*
 - Make ions, move them in an electric field
 - Study some physics!
- Mass spec interpretation is *Chemistry*
 - recorded signals → determining fragments → assigning molecules



Mass Spectrometry Overview

Our goal after these two lectures and two homeworks:

Model a very simple ion detector



Igor Pro: Beyond “Getting Started”

In-Class Today:

Writing Simple Functions

Clicker Q's 7-8

Did you do the Igor “Getting Started” Tour?

- a) Yes, all of it.
- b) Yes, most of it.
- c) Yes, some of it.
- d) No, or not very much of it.

How do you feel about Igor so far?

- a) Ready and rarin' to go!
- b) Lots to learn, but I'm ready.
- c) I'd rather use tools I already know.

Igor File Vocabulary

- **Experiment:** an Igor file where you store data and graphs (.pxp) *Let's open a new experiment.*
- **DataFolder:** subdirectory in an experiment
 - Red Arrow in Data Browser shows current DataFolder
- **Notebook:** “file” in an experiment where you can write notes, paste graphs
 - *Windows → New → Notebook* (Formatted Text)
- **Procedures:** functions written by you (or others)
 - Some exist only in the experiment they are in
 - Can be shared with friends/colleagues
 - “Local” procedure window (*ctrl+m*)

More Igor File Vocabulary

- Data Browser
 - Can set Preferences to sort by “Name and Type”
- Help Browser (F1)
 - Look for help on functions
 - Can also get function help by right-clicking on a function name, choosing “Help on *functionName*”
 - Most functions have Examples
 - Many functions have Related Functions

Data is usually stored in Waves

- **Wave**: a vector (array) containing data
 - Exists until you “kill” it (even if you’re not looking at it in a graph or table)
 - Numbers (single precision by default) or text
- Every wave has some number of “points” (length)
- Waves have inherent “x” values = point number (0, 1, 2...)
 - When you **display** a wave, it’s plotted against the x values (unless you tell it to plot vs. something else)

Wave Names have Rules

- Wave names
 - Cannot include spaces, operators (+ - * /), special characters
 - Can use underscore (_)
 - Cannot begin with numbers
 - Must be ≤ 32 characters
 - Are not case sensitive
 - Cannot have the same name as functions, variables, Igor-used terms

Making Waves

- Let's make a really simple waves so that we can practice some simple operations

`Make/N=20 y_vals`

Igor function that creates a wave Flag for make that sets Number of points in wave Name of wave

- Now let's look at the values in the wave in a table:

`edit y_vals.id`

(special way to see the “x” and “data” values”)

Giving values to y_vals

- Now we can take advantage of the inherent “x” values and make $y_vals = f(x)$

*Y_vals = x
Display y_vals*

*Fix the axes to have range -10, 40 for x and y
Add “zero lines” (Ticks and Grids tab)*

Copy and Paste this plot in your Notebook

*Change your line by changing the equation in the command line
(e.g., $y_vals = 2*x$ $y_vals = 2*x + 3$)*

Copy and Paste a new plot in your Notebook

Note that the y_vals.d changes in the table, too!

Kill the table!

- (Kill the wabbit, kill the wabbit...)
- Check for the wave in the Data Browser
 - Waves exist until you kill them, even if you’re not looking at the them.
 - Very different from Excel, of course!
- Make table again if you want to watch the data change

Can you make the line go into the $x < 0$ range?

- Recall from “Getting Started” that you can *change* the inherent x-scaling
Data → Change Wave Scaling
- The command for this change was printed in the History in the Command Window!
 - Useful for using this command in a function, since you can’t use the pulldown menu in a function.

Make a function for the line

1. Simple version

Our Sample Function

```
Function LineValues()  
    make/O/N=20 y_vals = 0.1 * x + 1  
end
```

Function Name (points to `LineValues`)
Function Arguments (points to `()`)
Stuff for the function to do (points to `y_vals = 0.1 * x + 1`)
Gotta end the function (points to `end`)

And let's compile!

And let's run it (from the command line).

Change the values. Does the line change?

Make a function for the line

1. Simple version *with variables*

Our Sample Function, with Variables

```
Function LineValues2()  
    variable m = 0.1, b = 1  
    make/O/N=20 y_vals = m * x + b  
end
```

Make a function for the line

1. Simple version with variables
2. Generalize function with inputs

Our Sample Function, with Inputs

```
Function LineValues_input(m,b)
    variable m, b
    make/O/N=20 y_vals = m * x + b
end
```

Make a function for the line

1. Simple version
2. Generalize function with inputs
3. Print the equation of the line
 - a) Printing equation is an option in the function

Our Sample Function, Print Equation

```
Function LineValues(m,b)
    variable m, b
    make/O/N=20 y_vals = m * x + b
    print "Equation is y=", m, "*x + ", b
end
```

Our Sample Function, Print Equation Sometimes

```
Function LineValues(m, b, printFlag)
    variable m, b, printFlag
    make/O/N=20 y_vals = m * x + b
    if(printFlag == 1)
        print "Equation is y=", m, "*x + ", b
    endif
end
```

Make a function for the line

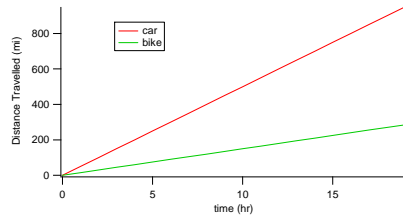
1. Simple version
2. Generalize function with inputs
3. Print the equation of the line
 - a) Printing equation is an option in the function
4. Add ability to define x range

Our Sample Function, Print Equation Sometimes

```
Function LineValues(m,b, x_min, x_max,  
printFlag)  
    variable m, b, x_min, x_max, printFlag  
    setScale/I x x_min, x_max,"", y_vals  
    make/O/N=20 y_vals = m * x + b  
    if(printFlag == 1)  
        print "Equation is y=", m, "*x + ", b  
    endif  
end
```

Other Examples

- When you make a lot of waves with the same x data (e.g., `time_hr`), clearer to just use a separate x wave



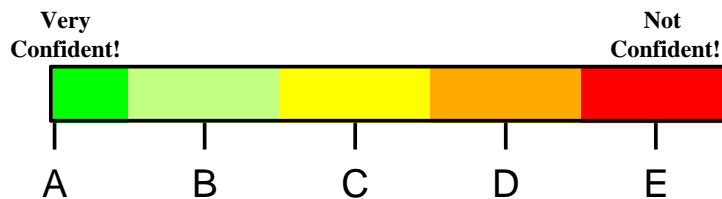
- Calculate the distance that a car goes (50 mph) and that a bike goes (15 mph) over 10 hours

`display dist_car_mi, dist_bike_mi vs time_hr`

Plotted vs.
same x wave

- Plot $\ln(x)$, warn user if `x_min < 0`

How do you feel about doing homework in Igor?



Confidence Scale

Lots of Help This Week!

~ Unlimited office hours (10 am – 6pm) in M112
(+ Sunday, time TBA)

Not available then? ulbrich@colorado.edu

Bring questions, and laptop with your work!

Check out my Igor Quick Reference Wiki

- http://cires.colorado.edu/jimenez-group/wiki/index.php/Igor_Quick_Reference
- Still a work in progress! Please make suggestions for improvements!

Ingrid's Sample Functions