Lab Assignments: PH 150

Introductory Physics Laboratory

R. Todd Lines
Brigham Young University-Idaho

Preface

This class is a class on experimentation. The goals of the course are to help you understand the following:

1. Error and how to deal with and report error in experiments
2. How to communicate experimental results
3. Numerical Modeling and how error can be calculated in a numerical model
4. Experimental Design

With just 13 weeks of labs, one lab a week, it is a challenge to achieve all of these goals. It will greatly help if you will read the reading assignment and the lab introduction before coming to class each week.

Even though this is a somewhat daunting set of goals, this class is also a lot of fun. The class ends with you designing and performing your own experiments.

In a way, this class allows us to ask the question “how do we know things in the sciences?” This is an important question for our era.

I hope you do have fun in this class, but I hope you also think about how we think about science.

Acknowledgments

Contents

1 Measurement and Uncertainty I

Hopefully you have read the first three chapters of Taylor's book. This is a quick review.

The philosophy of experimental measurement

In experimental physics, we assume that at a single instant of time that there is a "correct" value for some physical quantity we wish to know. For example, how fast a car is going at a particular time. The correct value might be something like $64.99999\,\mathrm{mi/h}$ (because we don't speed here at BYU-I).
The problem is that we can never measure this "correct" value. That is because no instrument is perfectly accurate. We have to make do with the goal of getting as close as we can to the "correct" value. Your speedometer, for example, probably would report this speed as $65\,\mathrm{mi/h}$. But that is not exactly correct.
Several factors affect how well we can measure quantities.

- Proper instruments
- Instrument calibration
- Measurement repeatability
- Quantization error

Actually there are more factors, but we will deal with the first three in PH150. PH 250 will deal with computer control and so will likely introduce quantization error. If you are a physics major, you will take PH336 and deal with additional error factors.

values

In experimental physics we use the word *value* to mean a number and its units. For example, we might measure a metal rod and say that the length of the rod is

$$L = 97.6\,\mathrm{cm}$$

The number is 97.6 and the units are centimeters $(\,\mathrm{cm})$. Together they are a value. Numbers without units are not useful in experimental physics. **You should always report both a number and its units.**

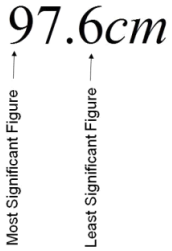
Errors

I give you this rod length, all you really know is that the rod is not longer than about

$97.7\,\mathrm{cm}$ or shorter than $97.5\,\mathrm{cm}$ (assuming you trust my ability to measure rods!). Could you be sure that the rod was not really $97.61\,\mathrm{cm}$ or maybe $97.62\,\mathrm{cm}$?

After making a measurement we have some uncertainty left over because of the limitation of our instrument. In this case our instrument is a meter stick, and you know that meter sticks have a smallest tick mark spacing, usually $1\,\mathrm{mm}$. I could probably judge to within half a millimeter. But it would be kind of crazy to say that you could be correct to within a $0.000001\,\mathrm{m}$ using a meter stick. You will learn in PH123 that meter sticks expand and contract with changes in temperature. So the meter stick, itself is not correct to within a hundredth of a millimeter!

One way to express this uncertainty in a measured value is by using significant figures. You have probably done this in high school.



We assume the last figure (in this case the 6) holds the uncertainty. Think of making this measurement. You would not be to uncertain about the $90\,\mathrm{cm}$ represented by the first digit. It is a big mark on the meter stick, and you could probably tell that the rod was almost a meter long without actually measuring. You are probably not to uncertain about the $7\,\mathrm{cm}$ represented by the second digit. You can easily read this from the meter stick. You will have to count millimeter marks to get the $0.6\,\mathrm{cm}$ represented by the last digit. But it is unlikely that the rod will end exactly at on the sixth millimeter mark. So this is where our uncertainty comes in. This is why we call the last digit the least significant digit, because it is the most uncertain.

If I give you a measurement like $97.6\,\mathrm{cm}$ you would assume that I could be off by $0.1\,\mathrm{cm}$ or by one millimeter. That is what stating $97.6\,\mathrm{cm}$ means. It would be better to state this explicitly

$$L = 97.6 \pm 0.1\,\mathrm{cm}$$

But you may object! You can do better than $\pm 0.1\,\mathrm{cm}$ with a meter stick. And so can I. This is one reason using significant figures is not such a great idea. It would be better to state our measurement and tell the person with whom we are communicating what we think the uncertainty really is. Say

$$L = 97.65 \pm 0.03\,\mathrm{cm}$$

This means that I measured $97\,\mathrm{cm}$, then counted up six millimeter marks, and noticed

that the rod end landed at just about half a mark more beyond the sixth mark. I am telling you that I think because of thermal expansion (and my poor eyesight) that I can only be sure of this measurement to within about a third of a millimeter.

We will always use this notation to report values and their errors in this class. If you report a value without an uncertainty, something is wrong (and the grader will surely notice!). This is because in your actual jobs as scientists not being clear about how well you know a value can be disastrous, even causing loss of life or property. So to keep you safe in future jobs, we will use best practices here.

Precision

So far when we have considered how correct our value is what we have really been talking about is the *precision* of our measurement. If I measure the metal rod 50 times, I will get about the same measurement, but not quite, each time. I might do a poor job of lining up the meter stick and the rod, or the temperature might change and the meter stick might shrink or expand. Whatever the problem, each measurement will be a little different. This small fluctuation about the "correct" value is called the precision of the measurement. It tells us how likely it is to get the same value each time we perform the experiment.

Think of throwing darts. The dots in the next figure represent the location of five darts from two dart players.

The blue dots show less spread in their locations than the red dots do. We would say that the blue dart player was more precise. Another way to say this is there is less uncertainty in where the blue player's darts will go. In our notation for reporting a value the plus-or-minus-part is this uncertainty.

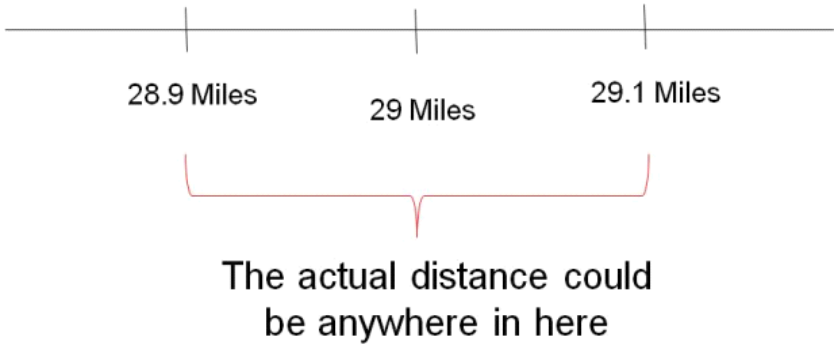
$$A = A_N \pm \delta A$$

Nominal Value Uncertainty

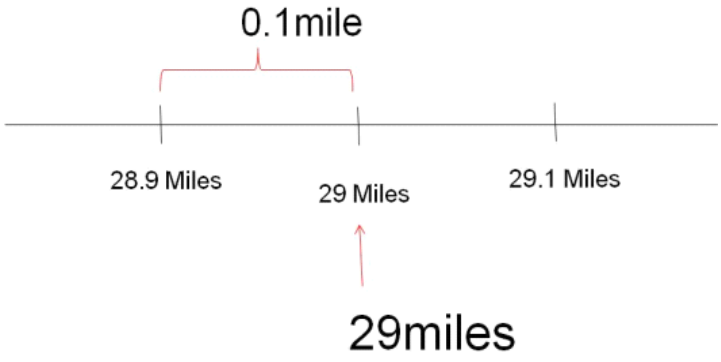
We call the actual measurement the *nominal value*. That would be where the dart thrower was aiming. If you know statistics, you can see that this is a little bit like an average and standard deviation.

Here is another example. Suppose we drive to Idaho falls. The distance is about 29

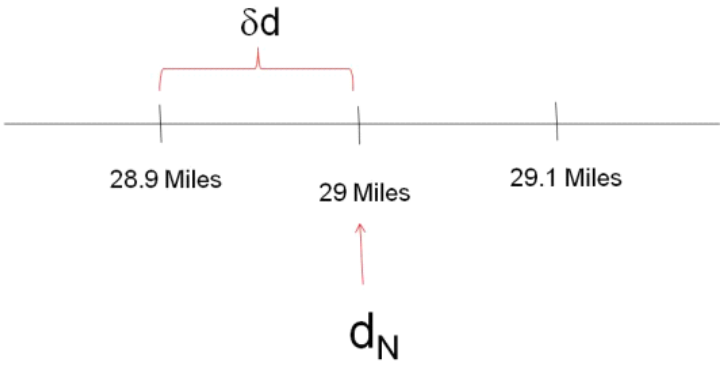
miles. If we use our odometer we might find it is marked in 10ths of a mile. So if I report a drive of 29miles, I might have gone 29.1miles, or 28.9 miles. Let's place these on a number line.



Notice that my uncertainty is 0.1mile.



If the distance is called d, then we write the uncertainty in the distance by placing the Greek letter δ in front of our name to get δd. This is read "delta d" or the "uncertainty in d."



We name the actual measurement, d_N where the N is for "nominal value" so then

$$\begin{aligned} d &= d_N \pm \delta d \\ &= 29 \pm 0.1\,\mathrm{mi} \end{aligned}$$

I report:

d=29±0.1 miles

This method of reporting a result is good most of the time, and we will always use it. But there is another way to report uncertainties that is helpful and we will often also add this new method to our reporting strategy. To illustrate the need for this second method, let's consider two measurements.

The first is the distance from Rexburg to Rochester NY. That distance is about 2082.90 miles or $3352100\,\mathrm{m}$. Further suppose that I tell you I have made this measurement to $\pm 1\,\mathrm{m}$. Is this measurement good?

Now suppose I measure one of our lab tables. I get that the table is $2\,\mathrm{m}$ long and I tell you that my measurement is good to $\pm 1\,\mathrm{m}$. Is this measurement good?

You can probably see that the first measurement is very good, while the second measurement could probably have been done better by guessing the length of the table. It is a terrible measurement with too much uncertainty.

But what makes the difference? The uncertainty is the same in both cases! Of course the difference is that in one case the uncertainty is a tiny fraction of the whole value, while in the table case the uncertainty is a large fraction of the measured value. If the uncertainty is large compared to the measured value, it is not a good measurement.

But we need a way to communicate this. The error is $\pm 1\,\mathrm{m}$ in both cases, let's call this the *absolute uncertainty*. Though the absolute uncertainty can't always tell us the quality of a measurement, we can use it to calculate

$$\frac{\delta L}{L} = \text{relative uncertainty}$$

This gives the error as a percentage of the total measurement. For our Rexburg to

Rochester measurement

$$\frac{\delta L}{L} = \frac{1\,\mathrm{m}}{352100\,\mathrm{m}} = 2.840\,1 \times 10^{-6}$$

and for the table

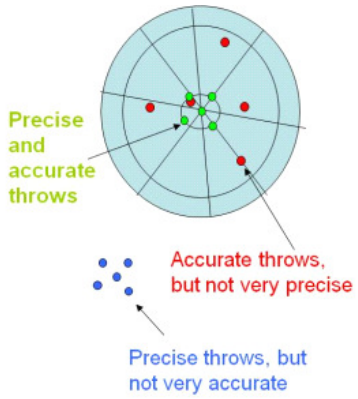
$$\frac{\delta L}{L} = \frac{1\,\mathrm{m}}{2\,\mathrm{m}} = 0.5$$

Since a good measurement has an error that is a small percentage of the total measurement, we can easily identify which measurement is better by calculating the percentage of the total value represented by the uncertainty and observing how small it is. We call this the "relative uncertainty."

The relative uncertainty will be small for that good measurement and large for a bad one. In this case we can easily see that the Rochester measurement is much better by looking at the relative uncertainty and noting that it is much smaller than the table relative uncertainty.

Accuracy

having a high precision measurement is good, but not enough. Here is a picture of our darts again, but this time I have included a target. Suppose you were trying to get a bull's-eye. We can see that our blue dart person is very precise, but he missed the bull's-eye–and the target! We need more than precision.



Notice that if we look at the average location of the red dart dots, the read dot thrower does seem to be aiming right at the target. We would say that he is accurate. Accuracy is whether or not you are aiming at the target. If we drive $29 \pm 0.1\,\mathrm{mi}$ as we discussed earlier, but we end up in Ashton instead of Idaho Falls, we would say that no matter how precise our driving distance is, we did not achieve our goal of getting to Idaho Falls. We mean that we are not accurate.

This might occur in the lab by having a scale that is not zeroed, or a ruler that is too

short or has extra material on the end. The measured amount given by an instrument when it should be measuring zero is called the "zero offset." If we know about the problem, we can just adjust the final number by this amount (or re-zero the instrument). But often we don't know about such problems and they can be hard to detect.
This type of error is called a *systematic error* because it affects the system each time it is used. The device will always be off by this amount until we fix it. We can improve our uncertainty estimate by taking many measurements and using the average as our value. But if we aim at the wrong place, no matter how many darts we throw, the accuracy error won't get better.
Of course what we want is both accuracy and precision in our measurements. The green dart dots have both.

Combining uncertainty

Suppose I want you to calculate the area of the cover of your text book. That is easy you say! it is the width times the height. And you are right

$$A = w \times h$$

Suppose we measure the book and find it has a height of $h = 28.4\,\mathrm{cm} \pm 0.2\,\mathrm{cm}$ and a width of $22.2\,\mathrm{cm} \pm 0.02\,\mathrm{cm}$. So the area is

$$\begin{aligned} A &= 28.4\,\mathrm{cm} \times 22.2\,\mathrm{cm} \\ &= 630.48\,\mathrm{cm}^2 \end{aligned}$$

but wait, what do we do with the uncertainties? If the initial measurements of the lengths are uncertain, then the area made from them must be more uncertain. We need a way to combine our uncertainties for the area.
Knowing what our uncertainty means now, the distance to the poorest measurements from a group, we can guess that if we were off by a positive $+0.2\,\mathrm{cm}$ on both measurements, then we would have the biggest area we could possibly get from our measurements. In some way it would be the most off we could get. Let's call this A_{max}. We would find it to be

$$\begin{aligned} A_{\mathrm{max}} &= (28.4\,\mathrm{cm} + 0.02\,\mathrm{cm}) \times (22.2\,\mathrm{cm} + 0.02\,\mathrm{cm}) \\ &= 28.6\,\mathrm{cm} \times 22.4\,\mathrm{cm} \\ &= 640.64\,\mathrm{cm}^2 \end{aligned}$$

Likewise, if we were off by $-0.2\,\mathrm{cm}$ on both measurements, then we would have the smallest area we could possibly get from our measurements. In another way it would

be the most off we could get. Let's call this $A_{\min}$. It would be

$$\begin{aligned} A_{\min} &= (28.4\,\text{cm} - 0.02\,\text{cm}) \times (22.2\,\text{cm} - 0.02\,\text{cm}) \\ &= 28.2\,\text{cm} \times 22.0\,\text{cm} \\ &= 620.4\,\text{cm}^2 \end{aligned}$$

To find the uncertainty, consider our trip to Idaho Falls. We found that the uncertainty was half the distance between our maximum estimate of our distance and the minimum estimate of our distance. We can use the same procedure for our area. We have the maximum and the minimum areas. The uncertainty is half the difference between these two extremes.

$$\delta A = \frac{A_{\max} - A_{\min}}{2}$$

using our numbers

$$\begin{aligned} \delta A &= \frac{640.64\,\text{cm}^2 - 620.4\,\text{cm}^2}{2} \\ &= 10.12\,\text{cm}^2 \end{aligned}$$

so our area should be reported as

$$A = 630\,\text{cm}^2 \pm 10\,\text{cm}^2$$

There are some tricks to this. We have to make sure we have the biggest value we can get when we get the maximum and the smallest value when we get the minimum. Suppose I measure two distances $x = 1.5\,\text{m} \pm 0.3\,\text{m}$ and $y = 3.0\,\text{m} \pm 0.2\,\text{m}$ and I want to calculate

$$z = \frac{y}{x}$$

then

$$\begin{aligned} z &= \frac{3.0\,\text{m}}{1.5\,\text{m}} \\ &= 2.0 \end{aligned}$$

what would the uncertainty in z be? Last time we chose adding the plus uncertainty to both values to get the maximum and we subtracted off both uncertainty values to get the minimum, but this time lets try every combination of plus and minus uncertainties

$$\begin{aligned} \frac{3.0\,\text{m} + 0.2\,\text{m}}{1.5\,\text{m} + 0.3\,\text{m}} &= 1.777\,8 \\ \frac{3.0\,\text{m} + 0.2\,\text{m}}{1.5\,\text{m} - 0.3\,\text{m}} &= 2.666\,7 \\ \frac{3.0\,\text{m} - 0.2\,\text{m}}{1.5\,\text{m} + 0.3\,\text{m}} &= 1.555\,6 \\ \frac{3.0\,\text{m} - 0.2\,\text{m}}{1.5\,\text{m} - 0.3\,\text{m}} &= 2.333\,3 \end{aligned}$$

Note that using both $+$ signs did not give the largest value. That is because for division a smaller denominator makes the fraction bigger. For the maximum we want a $+$ in the

numerator and a $-$ in the denominator. For the minimum we want a $-$ in the numerator and a $+$ in the denominator. It is a little tricky, but if we think, we can find the very biggest possible value and the very smallest possible value every time. To finish this off, lets find the uncertainty in z

$$\begin{aligned}\delta z &= \frac{z_{\max} - z_{\min}}{2} \\ &= \frac{2.666\,7 - 1.555\,6}{2} \\ &= 0.555\,55\end{aligned}$$

I will call this method of estimating the uncertainty the "High/Low method."
You may wonder if there is a more formal way to do all this. The answer is yes. I will call the more formal the *algebraic method.* We will tackle this next.

Algebraic method

We will develop an algebraic uncertainty combination equation for multiplication, division, addition, and subtraction. We will also tackle variables raised to a power. This will cover many simple situations, and we can use these formulae today. Eventually we will want to know how to estimate uncertainty for any function, That is explained in the reading from the book and we will tackle it in a future lab. But for now our goal is to have a provisional method for addition, subtraction, multiplication, and division. Taking a value to a power is really multiplication, so we will tackle it as well. Here is what we intend to find:

Function		Uncertainty Formula
Addition	$z = x + y$	$\delta z = \delta x + \delta y$
Subtraction	$z = x - y$	$\delta z = \delta x + \delta y$
Multiplication	$z = xy$	$\frac{\delta z}{\lvert z_N \rvert} = \left(\frac{\delta x}{x_N} + \frac{\delta y}{y_N}\right)$
Division	$z = \frac{x}{y}$	$\frac{\delta z}{\lvert z_N \rvert} = \left(\frac{\delta x}{x_N} + \frac{\delta y}{y_N}\right)$
Multiply by constant	$z = ax$	$\delta z = a\delta x$
Powers	$z = x^n$	$\frac{\delta z}{\lvert z_N \rvert} = n\frac{\delta x}{\lvert x_N \rvert}$

We can see that if you have $z = x + y$ then you just add the uncertainties $\delta z = \delta x + \delta y$. Of course z, x, and y stand for any variable. But how do we know this is true? Here are the details of how the algebraic method works

Multiplication

To multiply two measurements, say

$$m_{measured} = m_N \pm \delta m$$
$$v_{measured} = v_N \pm \delta v$$

We could write these as

$$m_{measured} = m_N \left(1 \pm \frac{\delta m}{|m_N|}\right)$$
$$v_{measured} = v_N \left(1 \pm \frac{\delta v}{|v_N|}\right)$$

This gives us the measurement in terms of the fractional uncertainties. If we wish to compute

$$p = mv$$

we use

$$p_N = m_N v_N$$

but what is the uncertainty in p?

The largest value of p is given by

$$p_{\text{large}} = m_{\mathbf{N}} v_{\mathbf{N}} \left(1 + \frac{\delta m}{|m_N|}\right)\left(1 + \frac{\delta v}{|v_{\mathbf{N}}|}\right)$$

which can be written as

$$p_{\text{large}} = m_{\mathbf{N}} v_{\mathbf{N}} \left(1 + \frac{\delta m}{|m_{\mathbf{N}}|} + \frac{\delta v}{|v_{\mathbf{N}}|} + \frac{\delta v}{|v_{\mathbf{N}}|}\frac{\delta m}{|m_{\mathbf{N}}|}\right)$$

We reason that fractional uncertainties should be small, so products of fractional uncertainties should be very small. We will ignore the very small term

$$\frac{\delta m}{|v_{\mathbf{N}}|}\frac{\delta m}{|m_{\mathbf{N}}|}$$

so we have

$$p_{\text{large}} = m_{\mathbf{N}} v_{\mathbf{N}} \left(1 + \frac{\delta m}{|m_{\mathbf{N}}|} + \frac{\delta v}{|v_{\mathbf{N}}|}\right)$$

The smallest value of p is likewise

$$p_{\text{small}} = m_{\mathbf{N}} v_{\mathbf{N}} \left(1 - \frac{\delta m}{|m_{\mathbf{N}}|} - \frac{\delta v}{|v_{\mathbf{N}}|}\right)$$

In each case we have $m_{\mathbf{N}} v_{\mathbf{N}}$ and then either plus or minus the term $\frac{\delta m}{|m_{\mathbf{N}}|} + \frac{\delta v}{|v_{\mathbf{N}}|}$ so we have, for our calculated value of p

$$p_{\text{calculated}} = m_{\mathbf{N}} v_{\mathbf{N}} \left(1 \pm \left(\frac{\delta m}{|m_{\mathbf{N}}|} + \frac{\delta v}{|v_{\mathbf{N}}|}\right)\right)$$

which we must be able to write as a nominal value and an uncertainty in p

$$p_{\text{calculated}} = p_{\mathbf{N}} \left(1 \pm \frac{\delta p}{|p_{\mathbf{N}}|}\right)$$

Comparing the previous two equations we can see that we must have

$$\frac{\delta p}{|p_{\mathbf{N}}|} = \left(\frac{\delta m}{m_{\mathbf{N}}} + \frac{\delta v}{v_{\mathbf{N}}}\right)$$

Then when we multiple measured quantities, we add fractional uncertainties. This is our algebraic rule for multiplication.

Don't forget, that we need to report δp, so to find δp we take

$$\delta p = \frac{\delta p}{|p_{\mathbf{N}}|} p_{\mathbf{N}}$$

Division

Let's start again with two measured values x and y with the form

$$x = x_{\mathbf{N}}\left(1 \pm \frac{\delta x}{|x_{\mathbf{N}}|}\right)$$
$$y = y_{\mathbf{N}}\left(1 \pm \frac{\delta y}{|y_{\mathbf{N}}|}\right)$$

We can find the quotient

$$q = q_{\mathbf{N}} \pm \delta q$$

as we did for multiplication

$$q = \frac{x_{\mathbf{N}}\left(1 \pm \frac{\delta x}{|x_{\mathbf{N}}|}\right)}{y_{\mathbf{N}}\left(1 \pm \frac{\delta y}{|y_{\mathbf{N}}|}\right)}$$

Again find the maximum quotient

$$q_{\text{large}} = \frac{x_{\mathbf{N}}\left(1 + \frac{\delta x}{|x_{\mathbf{N}}|}\right)}{y_{\mathbf{N}}\left(1 - \frac{\delta y}{|y_{\mathbf{N}}|}\right)}$$

and we will play a mathematical trick, we will multiple both top and bottom by $\left(1 + \frac{\delta y}{|y_{\mathbf{N}}|}\right)$, since

$$\frac{\left(1 + \frac{\delta y}{|y_{\mathbf{N}}|}\right)}{\left(1 + \frac{\delta y}{|y_{\mathbf{N}}|}\right)} = 1$$

this will not change our value for q_{large}

$$q_{\text{large}} = \frac{x_{\mathbf{N}}\left(1 + \frac{\delta x}{|x_{\mathbf{N}}|}\right)}{y_{\mathbf{N}}\left(1 - \frac{\delta y}{|y_{\mathbf{N}}|}\right)} \frac{\left(1 + \frac{\delta y}{|y_{\mathbf{N}}|}\right)}{\left(1 + \frac{\delta y}{|y_{\mathbf{N}}|}\right)}$$

The denominator is

$$y_{\mathbf{N}}\left(1 - \frac{\delta y}{|y_{\mathbf{N}}|}\right)\left(1 + \frac{\delta y}{|y_{\mathbf{N}}|}\right)$$

We can perform the multiplication to get

$$y_{\mathbf{N}}\left(1 - \frac{\delta y}{|y_{\mathbf{N}}|} + \frac{\delta y}{|y_{\mathbf{N}}|} - \frac{\delta y}{|y_{\mathbf{N}}|}\frac{\delta y}{|y_{\mathbf{N}}|}\right)$$
$$= \quad y_{\mathbf{N}}\left(1 - \frac{\delta y}{|y_{\mathbf{N}}|}\frac{\delta y}{|y_{\mathbf{N}}|}\right)$$

and we can perform the multiplication in the numerator

$$x_{\mathbf{N}}\left(1 + \frac{\delta x}{|x_{\mathbf{N}}|} + \frac{\delta y}{|y_{\mathbf{N}}|} + \frac{\delta x}{|x_{\mathbf{N}}|}\frac{\delta y}{|y_{\mathbf{N}}|}\right)$$

so

$$q_{\text{large}} = \frac{x_{\mathbf{N}}\left(1 + \frac{\delta x}{|x_{\mathbf{N}}|} + \frac{\delta y}{|y_{\mathbf{N}}|} + \frac{\delta x}{|x_{\mathbf{N}}|}\frac{\delta y}{|y_{\mathbf{N}}|}\right)}{y_{\mathbf{N}}\left(1 - \frac{\delta y}{|y_{\mathbf{N}}|}\frac{\delta y}{|y_{\mathbf{N}}|}\right)}$$

The term

$$\frac{\delta y}{|y_{\mathbf{N}}|}\frac{\delta y}{|y_{\mathbf{N}}|}$$

is very small compared to 1 (if $\delta y/\left|y_{\mathbf{N}}\right|$ is a small number, as we assume, then $\delta y^2/\left|y_{\mathbf{N}}\right|^2$ will be very tiny) so we will drop it from our calculations (we are calculating uncertainty, the fifth decimal place in the uncertainty is very uncertain!). We can do this as well with the term

$$\frac{\delta x}{|x_{\mathbf{N}}|}\frac{\delta y}{|y_{\mathbf{N}}|}$$

like we did with the multiplication case. Then

$$\begin{aligned} q_{\text{large}} &= \frac{x_{\mathbf{N}}\left(1+\frac{\delta x}{|x_{\mathbf{N}}|}+\frac{\delta y}{|y_{\mathbf{N}}|}\right)}{y_{\mathbf{N}}\,(1)} \\ &= \frac{x_{\mathbf{N}}}{y_{\mathbf{N}}}\left(1+\frac{\delta x}{|x_{\mathbf{N}}|}+\frac{\delta y}{|y_{\mathbf{N}}|}\right) \end{aligned}$$

We could go through this again for q_{small} and we would find

$$\begin{aligned} q_{\text{small}} &= \frac{x_{\mathbf{N}}\left(1-\left(\frac{\delta x}{|x_{\mathbf{N}}|}+\frac{\delta y}{|y_{\mathbf{N}}|}\right)\right)}{y_{\mathbf{N}}\,(1)} \\ &= \frac{x_{\mathbf{N}}}{y_{\mathbf{N}}}\left(1-\left(\frac{\delta x}{|x_{\mathbf{N}}|}+\frac{\delta y}{|y_{\mathbf{N}}|}\right)\right) \end{aligned}$$

We can then write

$$\begin{aligned} q_{\text{calculated}} &= \frac{x_{\mathbf{N}}}{y_{\mathbf{N}}}\left(1\pm\left(\frac{\delta x}{|x_{\mathbf{N}}|}+\frac{\delta y}{|y_{\mathbf{N}}|}\right)\right) \\ &= q_{\mathbf{N}}\left(1\pm\frac{\delta q}{|q|}\right) \end{aligned}$$

where

$$q_{\mathbf{N}}=\frac{x_{\mathbf{N}}}{y_{\mathbf{N}}}$$

and

$$\frac{\delta q}{|q|}=\frac{\delta x}{|x_{\mathbf{N}}|}+\frac{\delta y}{|y_{\mathbf{N}}|}$$

just like the formula for multiplication! The rule is when we divide measured quantities, we add fractional uncertainties

Addition

Let's take two measurements

$$x_{measured}=x_{\mathbf{N}}\pm\delta x$$

$$y_{measured}=y_{\mathbf{N}}\pm\delta y$$

and add them to get the largest value of the sum, z

$$\begin{aligned} z_{\text{large}} &= x_{measured}+y_{measured} \\ &= x_{\mathbf{N}}+y_{\mathbf{N}}+\delta x+\delta y \end{aligned}$$

We can also find the smallest value for z

$$\begin{aligned} z_{\text{small}} &= x_{measured}+y_{measured} \\ &= x_{\mathbf{N}}+y_{\mathbf{N}}-(\delta x+\delta y) \end{aligned}$$

so if we write this as $z_{\mathbf{N}} \pm \delta z$ we have

$$z_{\mathbf{N}} \pm \delta z = x_{\mathbf{N}} + y_{\mathbf{N}} \pm (\delta x + \delta y)$$

and we can identify

$$\begin{aligned} z_{\mathbf{N}} &= x_{\mathbf{N}} + y_{\mathbf{N}} \\ \delta z &= (\delta x + \delta y) \end{aligned}$$

The rule is that when we add measurements we add their uncertainties.

Subtraction

Let's take two measurements

$$x_{measured} = x_{\mathbf{N}} \pm \delta x$$
$$y_{measured} = y_{\mathbf{N}} \pm \delta y$$

and add them to get the largest value of the sum, z

$$\begin{aligned} z_{\text{large}} &= x_{measured} - y_{measured} \\ &= (x_{\mathbf{N}} + \delta x) - (y_{\mathbf{N}} - \delta y) \end{aligned}$$

We can also find the smallest value for z

$$\begin{aligned} z_{\text{small}} &= x_{measured} - y_{measured} \\ &= (x_{\mathbf{N}} - \delta x) - (y_{\mathbf{N}} + \delta y) \end{aligned}$$

so if we write this as $z_{\mathbf{N}} \pm \delta z$ we have

$$z_{\mathbf{N}} \pm \delta z = x_{\mathbf{N}} - y_{\mathbf{N}} \pm (\delta x + \delta y)$$

and we can identify

$$\begin{aligned} z_{\mathbf{N}} &= x_{\mathbf{N}} - y_{\mathbf{N}} \\ \delta z &= (\delta x + \delta y) \end{aligned}$$

The rule is that when we subtract measurements, we add their uncertainties just as we found for addition.

Measured quantity times an exact number

Suppose we want the circumference of a circle. We measure the radius to be

$$r_{measured} = r_{\mathbf{N}} \pm \delta r$$

and we calculate

$$C_{calculated} = 2\pi r_{measured}$$

how do we determine the uncertainty?

There is not uncertainty in the 2 nor in the π. So we multiply

$$\begin{aligned} C_{calc} &= 2\pi \left(r_{\mathbf{N}} \pm \delta r\right) \\ &= 2\pi r_{\mathbf{N}} \pm 2\pi \delta r \end{aligned}$$

which gives

$$C_{\mathbf{N}} \pm \delta C = 2\pi r_{\mathbf{N}} \pm 2\pi \delta r$$

and we identify

$$C_{\mathbf{N}} = 2\pi r_{\mathbf{N}}$$

and

$$\delta C = 2\pi \left|\delta r\right|$$

In general, if we have

$$q = Bx$$

where B is a constant, then we should expect

$$\delta q = \left|B\right| \delta x$$

Powers For a power, we really are just multiplying

$$y = x^2 = x \times x$$

so, taking the fractional uncertainty rule for multiplication,

$$\begin{aligned} \frac{\delta y}{|y|} &= \frac{\delta x}{|x|} + \frac{\delta x}{|x|} \\ &= 2\frac{\delta x}{|x|} \end{aligned}$$

In general, if

$$y = x^n$$

then

$$\frac{\delta y}{|y|} = n\frac{\delta x}{|x|}$$

Judging success of an experiment

Now we know how to describe the uncertainty in a measurement and we can even judge if a measurement is a good one using relative uncertainties. We can find final uncertainties after a calculation. But how do we know, based on our measurements, if our experiment is a success?

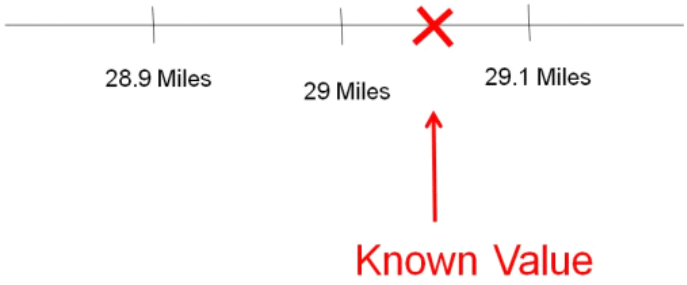
If we have a known value, we can compare our experimental results to that known value and judge our accuracy. We do this with a percent error

$$PE = \left(\frac{\text{measured value - accepted value}}{\text{accepted value}} \times 100\right)\%$$

We can compare this to our relative uncertainty

$$RE = \left(\frac{\delta(value)}{\text{nominal value}} \times 100\right)\%$$

Let's take our drive to IF as an example. Suppose we have a reliable study that shows the distance to IF is $29.05\,\mathrm{mi}$.



And we go to IF and find that our odometer measures $29 \pm 0.1\,\mathrm{mi}$.

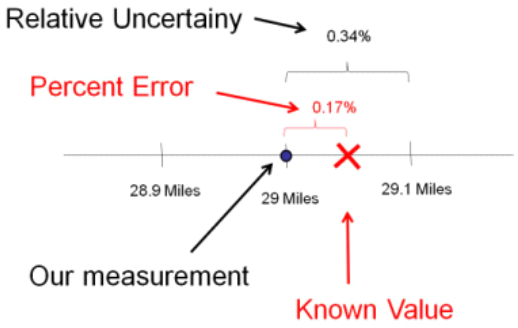
The percent error is

$$\begin{aligned} PE &= \left(\frac{29.05\,\mathrm{mi} - 29\,\mathrm{mi}}{29\,\mathrm{mi}} \times 100\right)\% \\ &= 0.172\,41\% \end{aligned}$$

This is roughly a 2% error.
Our relative uncertainty is

$$\begin{aligned} RE &= \left(\frac{0.1\,\mathrm{mi}}{29\,\mathrm{mi}} \times 100\right)\% \\ &= 0.344\,83\% \end{aligned}$$

Let's see what this means



We can see that we are off from the known value by 0.17%, but remember we are uncertain in our measurement. Our uncertainty tells us we can be anywhere within 0.34% of the value we measured. Since our percent error–how much we are off–is less than the fractional uncertainty—percent off we can be based on our equipment and our technique–we can say that this is an accurate value for the distance to Idaho Falls. More succinctly, if our percent error is smaller than our fractional uncertainty, we are accurate.
But suppose our percent error is larger than the relative uncertainty? Then we are not accurate. It is always good when this happens to try to figure out what the problem

could be. There may be a systematic error, or it may be that you failed to recognize some source of error.

Here is a rule of thumb for judging the accuracy of an experiment.

1. If the relative uncertainly is larger than the percent error
   - The experiment is accurate to within the uncertainty of the experimental technique
   - To improve this measurement, you need better equipment or better technique
2. The relative uncertainty is smaller than the percent error
   - The experiment is not accurate to within the uncertainty of the experimental technique
   - To improve this experiment, look for systematic errors
   - Consider if you have underestimated the uncertainty

We report this along with our results.

In our first lab, we will get some practice calculating uncertainties and judging accuracies.

Lab Notebooks

Hopefully you noticed that a lab notebook is required for this class. The lab notebook is designed to be a record of what you did. If you had to repeat today's experiment five years from now, could you do it based on what you write today?

At most professional labs and major engineering companies your lab notebook is considered the property of the company or organization. It is the proof that you did the experiment that you say you did, and that you got the results you say you got. It has to be readable and understandable to someone who did not participate in the lab with you. This is a pretty tall order. To help you plan your entries, here are the criteria I will use to grade your lab book:

- Describing the goal for the work
  - Usually this takes the form of a physical law we will test.
- Give predictive equations and uncertainties for the predictions based on the physical law.
  - This usually involves forming a mathematical model. You should record any assumptions that went into the model (e.g. no air resistance, point sources, massless ropes, etc.).
    - In lab today we will find the volume of the room. Your mathematical model will likely be $V = \ell \times w \times h$. The mathematical model is not necessarily something complicated, but the reader needs to know how you are doing your calculations.
- Give your procedure
  - Recording what you really did (not the lab instructions), tell what changes you make in your procedure as you make them.
  - Record as you do the work.
  - Record the equipment used and settings, values, etc. for that equipment (see next item).
  - Did you learn how to use any new equipment? What did you learn that you want to recall later (say, when taking the final, or when you are a professional and need to use a similar piece of equipment five years from now).
- Record the data you used. . The data are all the measurements you took plus your best estimate of the uncertainties in the measurements. Record any values you got from tables or published sources (or from your professor) and state where you got these values. You don't always want to write down all the data you use. If you have a large set of values, you can place them in a file, and then record the file name and location in your lab notebook. Make sure this is a file location that does not change (emailing the data to yourself is not a good plan).
- Give a record of the analysis you performed. You should have given some idea of how you got your predictive equation. Now, what did you do to get the data through the equation? Were there any extra calculations? Did you obtain a set of "truth data"

(data from tables or published sources, or from an alternate experiment) for your experiment? If so, did you do any calculations, have any uncertainty, etc. associated with the truth values?

- Give a brief statement of your results and their associated uncertainties.
- Draw conclusions
  - Do your results support the theory? Why or why not? What else did you learn along the way that you want to record.
  - This is where we may compare the percent error to our relative uncertainty.

Assignment: Practice with Measurement and Uncertainty calculations.

Part 1 Percent Error: Mass of a Cylinder–the hard way

- Given the density of a metal cylinder, use this density to determine the mass the cylinder.
  - You cannot directly measure the mass of the cylinder, You will be provided a mass of the cylinder by your instructor to compare with your calculated value.
  - Report your method for obtaining the mass of the cylinder in your lab notebook (not just your result, but tell yourself in your notebook *how you got your result*).
  - Report the following results: 1) Density of the cylinder, 2) Predicted Mass of the cylinder, 3) Actual Mass of the cylinder. Comment on the accuracy and precision of your measurement.
  - Resources: You may use any equipment or other resources found in the lab or on the internet

Part 2 Combining Uncertainty: Volume of the room

- Determine the volume of this room, including uncertainties. Describe your method fully in your lab notebook, including which measuring instruments you used and why, and the uncertainty in each of your measurements.
- The relative uncertainty in the volume is
$$\frac{\delta V}{V} = \frac{\delta L}{L} + \frac{\delta H}{H} + \frac{\delta W}{W}$$
from Taylor's equation 2.28 or from our algebraic method multiplication rule.
- Compare your answers with those from your neighboring research institutions at the other tables. Are your answers the same to within the values of your uncertainty? If not, explain why they aren't.

Part 3: Tie to Experimentation

- We will learn in this class that you should understand the uncertainties in our measuring devices *before* you start performing an experiment. From what you have experienced so far today, why do you think this is so?

Part 4 Combining Uncertainty: Determine the Volume of a Stack of Paper

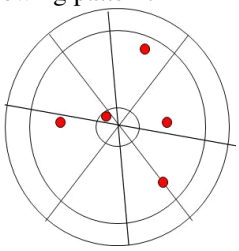
- Determine the volume of 20 pieces of paper (you can use more, but if you do, replace the number 20 with your actual number in the equation below).

- Determine the uncertainty in your measurement.
- Use your measurement to find the volume of one sheet of paper by dividing. Also determine the uncertainty in your calculation. This should be something like
$$\delta V_1 = \frac{\delta V_{20}}{20}$$
explain what this means in your lab notebook.
- Now measure the volume of one piece of paper directly using instruments (I might recommend a micrometer–ask if you have not used one before).
- How do your measurements compare?
- Which one is more accurate? Which is more precise? Why?

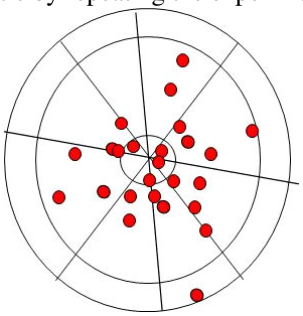
Part 5 Turn in your lab notebook.

2 Communicating Results I: Statistical Representation of Data

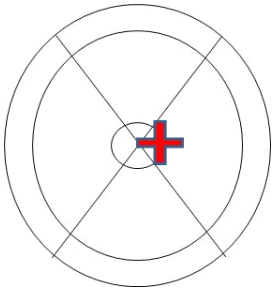
So far we have talked about repeating experiments, but we have been too pressed for time to actually do that. We should take the time to see how to report data from multiple results. Let's also tie the idea of multiple results to our ideas of uncertainty.
To do this, I would like to go back to our dart board. Suppose I throw the darts, trying for a bull's eye, and I get the following pattern.



We now know that this is fairly accurate, but not very precise. We say that there is a large uncertainty, but that we are aimed about the right direction. We could get a better estimate of how accurate we are by repeating the experiment many times



and finding an average location for the darts.



This average seems to be just a little right of center. Now we know that we should point the darts a little to the left. Many experiments are like this. We can repeat the experiment many times. The uncertainty might be larger than we want, but if we average over many trials of the experiment, we can find an average value that represents the actual value of the quantity we are trying to find.

Mean value as our best estimate value

The mathematical process we use to find the mean is simple and you are probably quite familiar with it. We simply add up all the values, and divide the sum by the number of values.

$$\begin{aligned} \bar{x} &= \frac{x_1 + x_2 + x_3 + \cdots x_N}{N} \\ &= \frac{1}{N}\sum_{i=1}^{N} x_i \end{aligned}$$

The last equation uses sigma notation. It is read as "one over N times the sum of x_i for $i = 1$ to N. It is a short-hand notation for the line above. We will use this notation because it makes writing our equations much easier. But that means it is very important that we understand what it means. So let's imagine that we have many values for the x-position for our darts.

$$\begin{aligned} x_1 &= 1.00 \pm 0.01\,\text{cm} \\ x_2 &= 0.50 \pm 0.01\,\text{cm} \\ x_3 &= -0.75 \pm 0.01\,\text{cm} \\ x_4 &= -2.25 \pm 0.01\,\text{cm} \\ x_5 &= 3.00 \pm 0.01\,\text{cm} \\ x_6 &= -0.80 \pm 0.01\,\text{cm} \\ x_7 &= 2.10 \pm 0.01\,\text{cm} \\ x_8 &= 1.2 \pm 0.01\,\text{cm} \end{aligned}$$

We have labeled each x with a number. That is what the x_i means. The "i" is an index. It stands for any number from 1 to N. Our sigma notation says we add up all these

positions, and divide by $N = 8$ since there are eight positions

$$\begin{aligned}\bar{x} &= \frac{(1.00 + 0.50 - 0.75 - 2.25 + 3.00 - 0.80 + 2.10 + 1.2)\ \mathrm{cm}}{8} \\ &= 0.5\,\mathrm{cm}\end{aligned}$$

which is a little bit to the right of our zero point.

Standard deviation as an estimate of our uncertainty

But what is our uncertainty? Each of our position measurements were good to $\pm 0.01\,\mathrm{cm}$. But this can't be what governs our uncertainty. We can see our points are spread out much more than $\pm 0.01\,\mathrm{cm}$. Something in the experiment (the bad dart thrower) is increasing the uncertainty. We could use our algebraic method to find the uncertainty, but that would be tedious and may not include the effects of the dart thrower. It would be great to have a way to use the spread of the points, itself, to obtain a numerical estimate of the uncertainty. The spread must include the effects of the dart thrower.

From your study of statistics, you can guess what we will use to represent uncertainty, but let's reason it out here. We could take how far each point is from where we aimed as an indication of how imprecise our throw was. That would be

$$\Delta x_i = \bar{x} - x_i$$

for each throw. In this equation we are using the Greek Δ to show a difference, and a bar over the x to mean "the average value of the x-position." Then Δx_i is how far off the i^{th} trow from the mean. Sometimes we are off to the right, and sometimes to the left. If we add up all the Δx_i values and average them, they will average to nearly zero most of the time. We can see that zero is not a good estimate of our uncertainty! So the average deviation won't work as a measure of uncertainty.

But we can play a trick. The quantity

$$\Delta x_i^2 = (\bar{x} - x_i)^2$$

is always positive. If we averaged Δx_i^2,

$$\overline{\Delta x_i^2} = \frac{1}{N}\sum_{i=1}^{N} \Delta x_i^2 = \frac{1}{N}\sum_{i=1}^{N} (\bar{x} - x_i)^2$$

nothing would cancel out. And we have solved our calcelation problem. But we have created another problem by doing this, $\overline{\Delta x_i^2}$ is like the square of our how far we are off. So let's take a square root

$$\sqrt{\overline{\Delta x_i^2}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \Delta x_i^2} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (\bar{x} - x_i)^2}$$

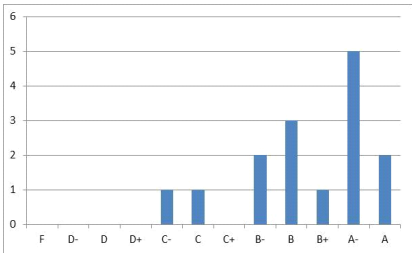
The quantity $\sqrt{\overline{\Delta x_i^2}}$, represents about how far off we are on average, it does not tend to zero, and has the same units as x_i so it can be an estimate of our uncertainty. It is about how far most of the points are off from the mean. But $\sqrt{\overline{\Delta x_i^2}}$ is a little hard to write, so we usually give this quantity the symbol σ, which is a Greek letter s and is pronounced "sigma." We also give σ a name. We call it the *standard deviation* because it is about how much the average point "deviates" from the mean position. So for our x-position we can write

$$\sigma_x = \sqrt{\sum_{i=1}^{N} \frac{(x_i - \bar{x})^2}{N}}$$

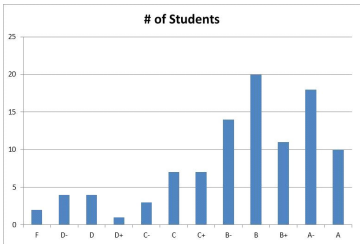
But what does this math symbology mean? To find σ_x, we must first find the average positions to find $\bar{x}$, then we take each x-position (x_i) and we subtract the mean from it $(x_i - \bar{x})$. We square the result. We do this for each of our x-positions. Then we have $(x_1 - \bar{x})^2$, $(x_2 - \bar{x})^2$, $(x_3 - \bar{x})^2$, $\cdots (x_N - \bar{x})^2$. We add these up, and divide by N to find the average $\sum_{i=1}^{N} \frac{(x_i - \bar{x})^2}{N}$. Then we take the square root.
In lab today, I will ask you to do this by hand once. That should be enough to convince you that you never want to do it by hand again! Normally we will use a computer to do this. I suggest you use one of our spreadsheet programs, or MatLab to do these calculations, and not your calculator.

Histograms

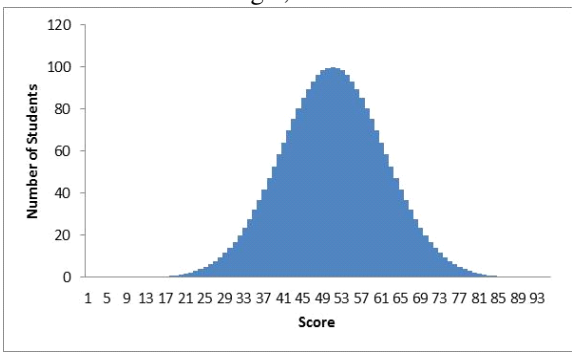
Suppose I plot the results of many, many dart throws. The way I want to plot this is something you have seen from grading for many years. I want the horizontal axis to show the x-position of the dart throws. I want the y-axis to show the number of darts that landed at a particular x-position. This type of graph is called a histogram. You often see grades given like this



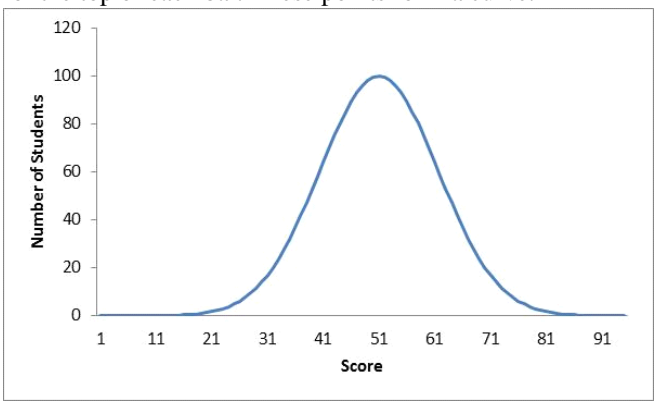
where we understand that the bars indicate how many students got an A (two in this case) and how many got an $A-$ (five in this case) etc.
If there are many students we can plot their scores and the shape of the histogram begins to smooth out some



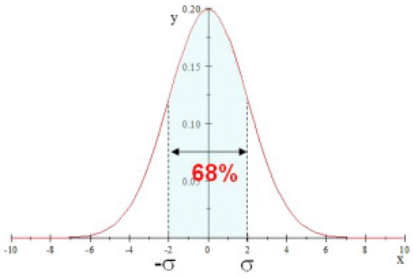
The more students or measurements we get, the more smooth the curve looks



If we had infinitely many students, we would get a perfectly smooth curve. You can see already that coloring in the bars in the graph is not useful any more. So usually we just draw a point for the top of each bar. These points form a curve.



Unlike student scores, dart positions can be negative. So our dart distribution should be centered on zero displacement. We will usually find that 68% of the darts will fall within $\pm\sigma$ of the mean.



We can see that our σ value is very like an uncertainty. But there is a difference. We still have 32% of our experiments outside of $\pm\sigma$, and if we give the uncertainty, δx, then all of the measurements should be within $\pm\delta x$. If you are building a space shuttle and absolutely need to guarantee that your error on your calcuation is within some limit, then you should use a true absolute uncertainty, $\pm\delta x$. But for most experiments, being that certain about our uncertainty is not required, and we can use $\pm\sigma$ as a good approximation to the uncertainty. We will often do this in this class. If losing 32% is not acceptable, but finding the true δx is not practical, it is often good enough to use 2σ or 3σ as the estimate of our uncertainty. 95% of the data will fall with $\pm 2\sigma$, and 99.7% of the data will fall within $\pm 3\sigma$. So these are more conservative estimates than using a single standard deviation. But in this class we will stick with just σ.

Standard deviation of the mean

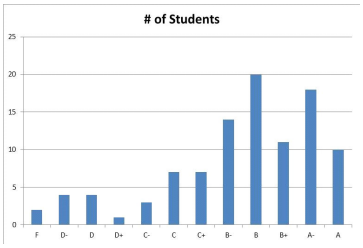
Now you may wonder, does the mean value get better as we take more measurements? That is, do we become more sure about where we are pointing if we throw more darts and include these many darts' locations in our average? I think you will see from our previous reasoning that this is the case. The more trials of an experiment that we take, the closer our mean value is to the "truth" value we are measuring. Since this is the case, shouldn't the uncertainty go down as we perform more trials?

The answer is yes. We won't derive this in our class. But the estimate of the uncertainty should be given by

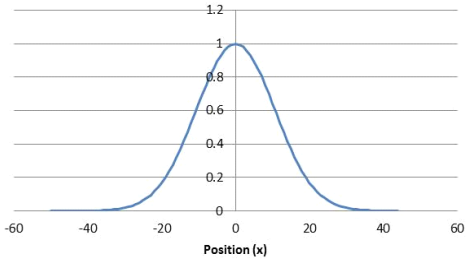
$$\sigma_{\bar{x}} = \frac{\sigma_x}{\sqrt{N}}$$

where σ_x is our standard deviation in our x-position values and N is the number of trials we took. The more trials that go into our average, the lower our uncertainty estimate. The value $\sigma_{\bar{x}}$ is called the *standard deviation of the mean*.

Notice that in some of our grade graphs, the most common score was not a C. Here is an example:



As students, this makes us all happier, but for our error analysis this causes a problem. The error analysis we have talked about so far assumes that our errors are distributed in a very uniform way. If I go back to this graph

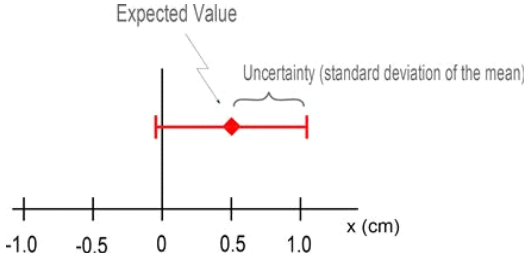


we can see that there are as many darts that landed to the left as there are to the right. This distribution of errors is called the *normal distribution*. Usually our errors in our labs will be normally distributed. That makes all the math we talked about work. But what if they are not, like our grade example? Well, that is a great topic for PH336. So for now we will just assume a normal distribution. But we can check to see how non-normal our data is. We can find the *mode* which is the value that occurs most frequently. For our grade distribution above it would be a B.
We can also find the place where half of the trials landed on one side and half on the other. This is called the *median* point. We will calculate both in our lab today. If we have a normal distribution, the average, median, and the mode will all be the same. If this is not the case, then we may worry a little about our error estimate–it may be too small.

Graphical reporting of the mean (expected value) and standard deviation (uncertainty)

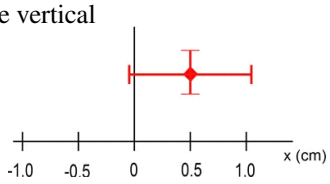
We now have a new view of measurement based on statistics. The mean value is the value that we will say is our measurement. We call this the *expected value*. The standard deviation is the representation of our uncertainty. We can plot this in a way that communicates both at once. If we take our eight data points that we started with earlier, we know the mean , $0.5\,\mathrm{cm}$, and we can find the standard deviation of the

mean to be $0.6\,\mathrm{cm}$. We plot this by making a dot or diamond or some larger point indicator. Then we make a line through the point with little ends that show the size of the uncertainty. The result looks like this.



We call the error indicators "error bars." Excel, and most plotting programs will allow you to add error bars to your graphs.

Of course, our example is only one dimensional. But we could have y-direction error bars as well. These would be vertical



and there is no reason the y-error would be the same as the x-error. We may encounter such situations in future labs.

Assignment

Complete this lab in an organized fashion in your lab notebook. *Part of the grade will be based on neatness and organization!* There are several histograms that need to be made during this lab. Make sure everyone in your group gets a chance to build a histogram.

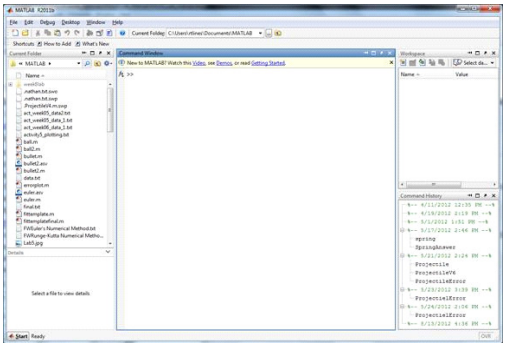
Statistical Data I: How long does it take to walk?

We will repeat one measurement, how long it takes to walk to a destination, many times. Each person in our class will take the measurement once.

1. We'll start by determining a walking destination as a class. Our destination is: _____.
2. Each person in the class should get a digital timer and time their walk to the destination and back. Walk at your normal walking speed. We will stagger when you leave, to avoid walking in groups. While you are not walking, you can begin working on part II.
3. When you return, record your walking time on the board to the nearest second.
4. Record the times for all class members in a table, and *by hand* determine the mean walking time, the median walking time, and (if appropriate) the modal walking time. Determine the standard deviation of the walking time *by hand* (show all your work).
5. Using a computer, make a histogram of the walking times (see instructions below)

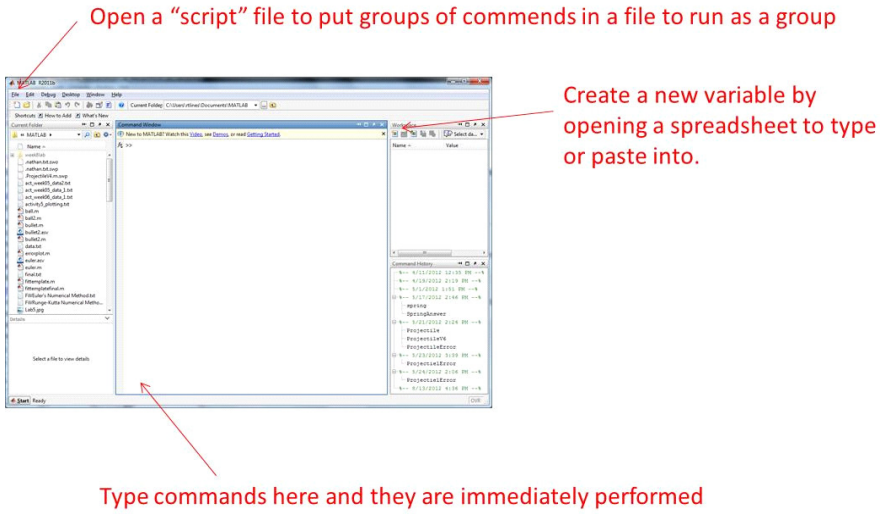
Making the Histogram in MatLab

We mentioned that part of this class is to learn how to solve physics problems using a computer. We have chosen the language called MatLab to do this. If you have not tried MatLab yet, now is the time. Start MatLab. You should see something like this

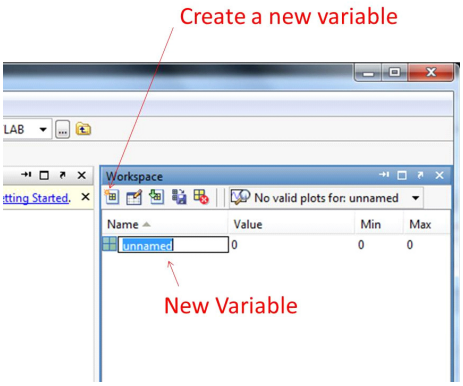


MatLab is like a calculator programing language. You can type commands in the main

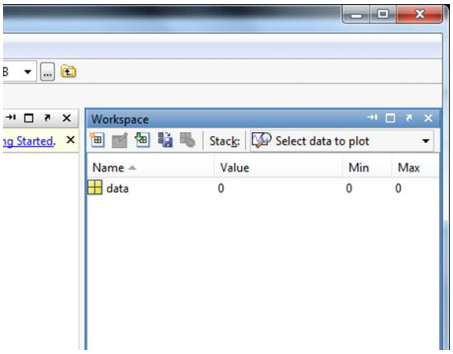
command window and it will perform them



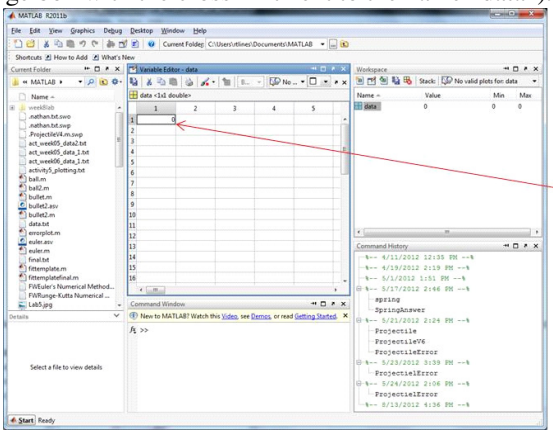
or you can save a list of commands into a file and have them run all at once. We will try both methods of using MatLab today. First, let’s make a variable with our data. To do this, use the mouse and click on the create new variable icon



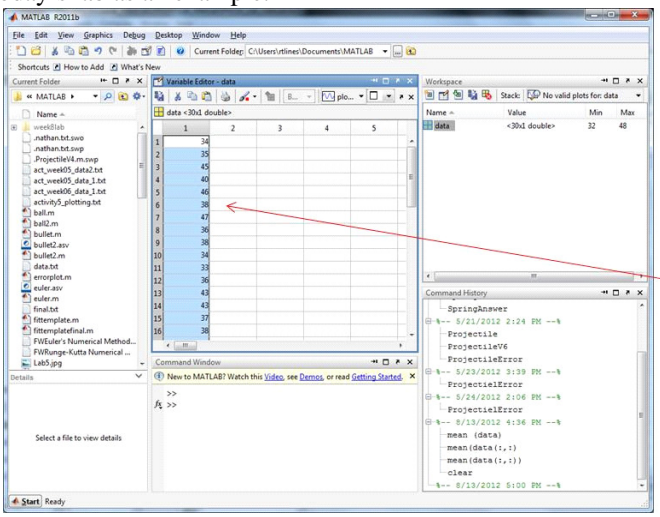
You need to type a name for your variable. We can just call it “data” for now. The result should look like this:



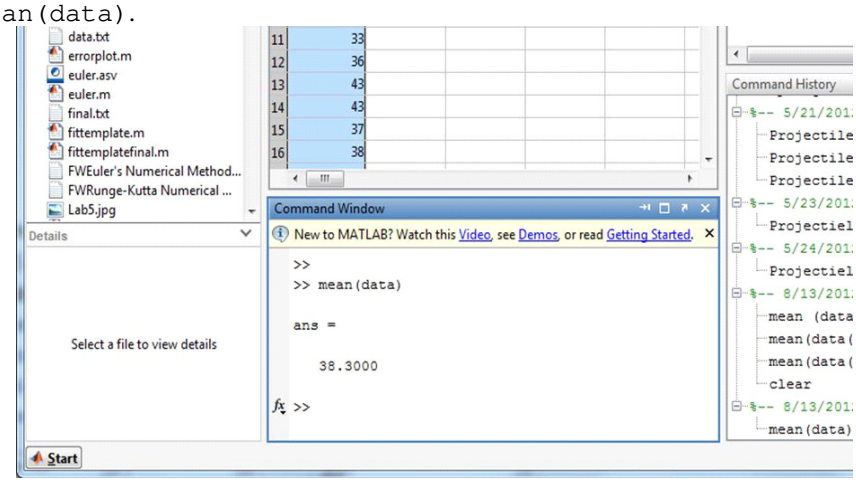
We can put values in this variable by double clicking the icon for our variable (the orange box with the cross in it next to the name “data”). A spreadsheet opens.



You can copy and paste values into the spreadsheet cells. Let’s use the data table from today’s lab as an example.

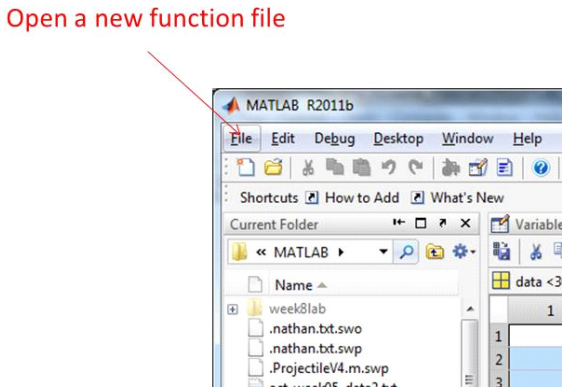


Now we have a variable, we can do calculations with this variable. Say we want to calculate the mean value of these numbers. We type in the command window `mean(data)`.

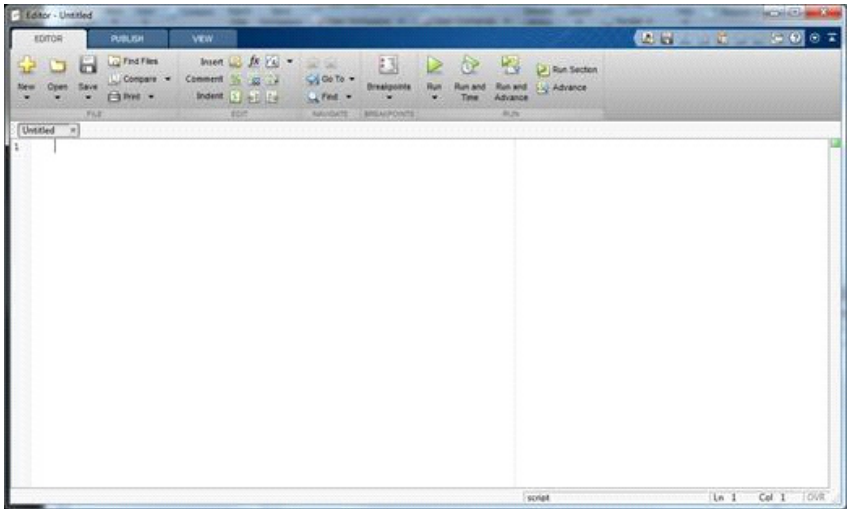


The answer appears on the next line in the command window. This is very like what happens on your calculator. The command window is like your calculator display. So once the variable "data" is made you can find the standard deviation, mode, and many other things using the variable and the results appear on the command window.

Now let's create a file of commands that we can give to MatLab to run all together. Open a file using the file menu



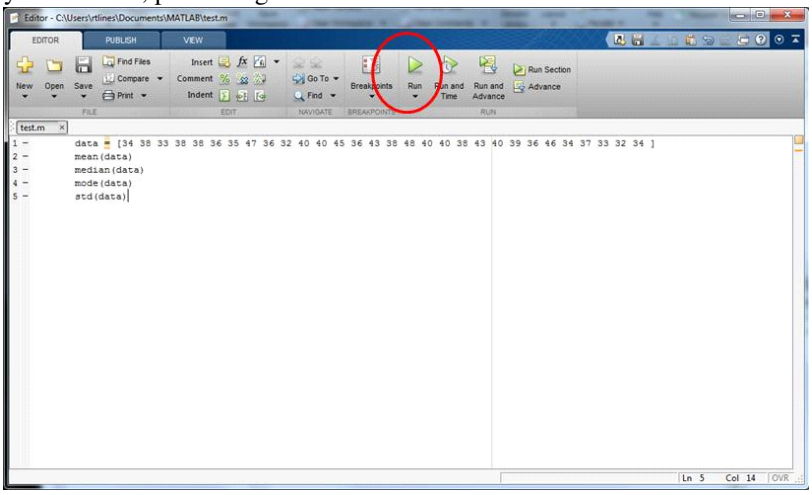
the result should look like something like this:



This is MatLab's file editor window. We will type our commands here, and then tell MatLab to run them. So in this window type the following commands

```
data = [34 38 33 38 38 36 35 47 36 32 40 40 45 36 43 38 48 40 40 38 43 40 39 36 46 34 37 33 32 34 ]
mean(data)
median(data)
mode(data)
std(data)
```

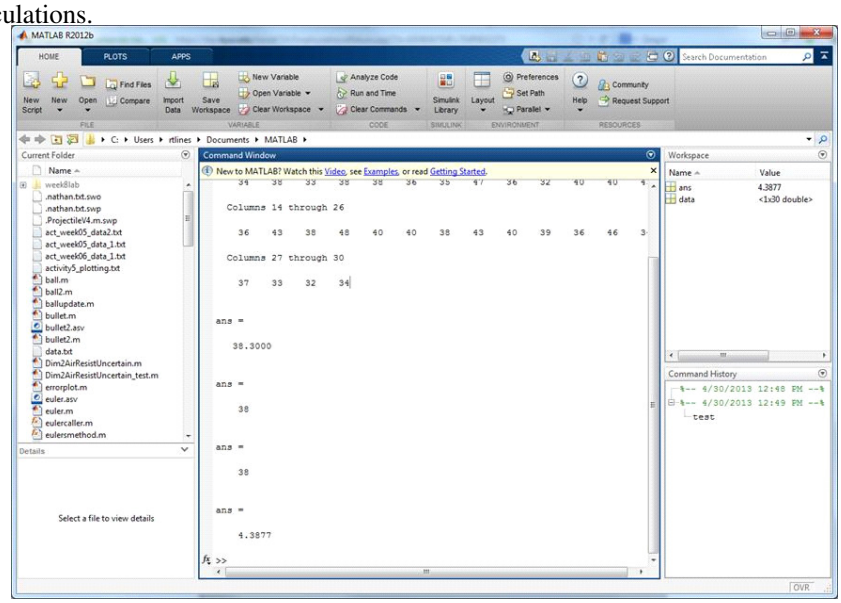
Once you are done, press the green "run" arrow.



MatLab will tell you that you need to save this file. You can choose any name, but don't

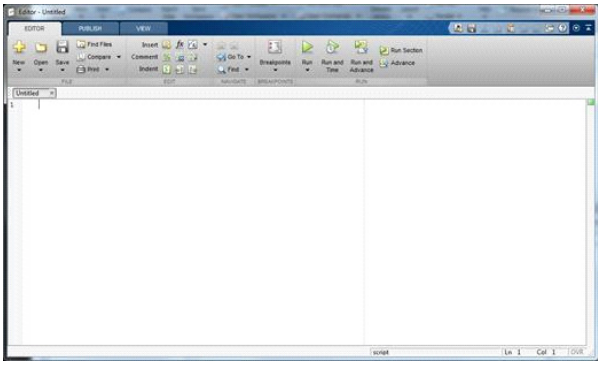
use spaces or strange special characters, and let MatLab choose the name of the file extension and where it puts the file You can save the file to your Cloud drive or USB drive later.

If you switch back to the command window, you will see the answers to all these calculations.



MatLab did all the commands in the file at once!

But there is another way to make MatLab perform multiple commands. That is to write a “function.” This is a little bit more formal than a script. Making a function is adding a capability to MatLab–adding a new way to do something. Start by opening a new file, just as before

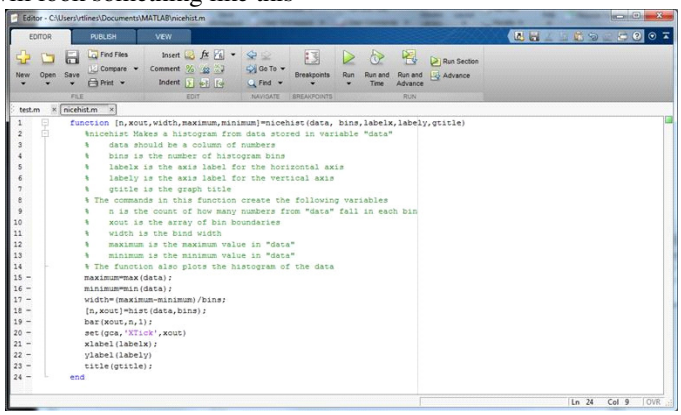


Now type the following commands in the file:

```
function [n,xout,width,maximum,minimum]=nicehist(data, bins,labelx,labely,gtitle)
    %nicehist Makes a histogram from data stored in variable "data"
```

```
%       data should be a column of numbers
%       bins is the number of histogram bins
%       labelx is the axis label for the horizontal axis
%       labely is the axis label for the vertical axis
%       gtitle is the graph title
% The commands in this function create the following variables
%       n is the count of how many numbers from "data" fall in each bin
%       xout is the array of bin boundaries
%       width is the bind width
%       maximum is the maximum value in "data"
%       minimum is the minimum value in "data"
% The function also plots the histogram of the data
maximum=max(data);
minimum=min(data);
width=(maximum-minimum)/bins;
[n,xout]=hist(data,bins);
bar(xout,n,1);
set(gca,'XTick',xout)
xlabel(labelx);
ylabel(labely)
title(gtitle);
end
```

The result will look something like this

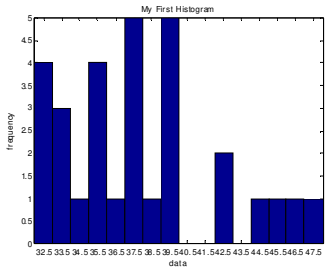


You need to save this file. But MatLab is kind of dumb in that it needs to choose the name of function files (so use the default it gives you–nicehist.m) and wants the file in a particular place. Let it put the new file where it wants. You can save it to your Cloud

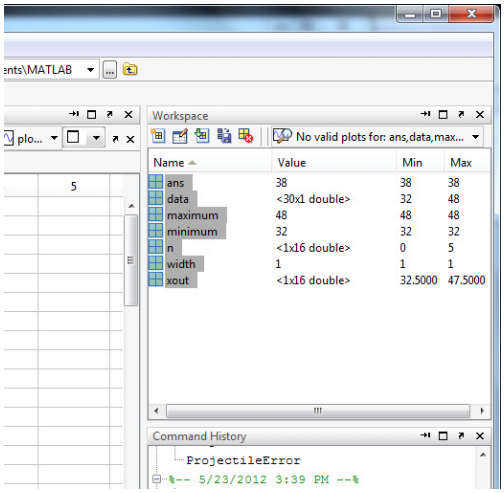
drive or USB drive later. You can't use the green arrow for this function file. Use this code by typing in the command window

[n,xout,width,maximum,minimum]=nicehist(data, 16,'data','frequency','My First Histogram')

You should see a picture that looks like



This corresponds to the bin size of 1. You should also notice that the new variables have been created.



You can see what number is in each of the variables by typing the variable name in the command window, or by double clicking on the variable's icon.

Make sure you record how to do this in your lab notebook and see if you can record what the individual lines of code do. Note that any line with a "%" sign is a comment. MatLab skips it, but it can be very useful to leave yourself some notes on what your commands do. You should always put some comments in a set of commands.

Copy your working file to your cloud or USB drive and note where you put it in your lab notebook.

We call a group of commands a "code" sometimes. So tell where your histogram code is in your lab notebook.

Statistical Data II: Using a Computer to Determine Statistical Values

After this week, you will be allowed to use your computer to calculate statistical values for you. Now would be a good time to practice, using a convenient software package. If you know R or SPSS that is fine. Otherwise, we have Excel on our lab computers. The commands below are for Excel.

1. As individuals, do the following:
   a. Enter the following set of numbers into a spreadsheet. Record them in a single column.

34	38	33	38	38	36
35	47	36	32	40	40
45	36	43	38	48	40
40	38	43	40	39	36
46	34	37	33	32	34

   b. Sort the data, using the sorting function of Excel.
   c. Using the AVERAGE function, calculate the mean value of this set.
   d. Using the MEDIAN function, determine the median value.
   e. Using the MODE function, determine the mode. Comment on the result given by Excel.
   f. Using the STDEV function, determine the standard deviation.

2. As a group of two or three, repeat this process using MatLab.
   a. To do this, input the data into a linear matrix
      i `data = [34 38 33 38 38 36 35 47 36 32 40 40 45 36 43 38 48 40 40 38 43 40 39 36 46 34 37 33 32 34 ]`
      ii Then use the commands `mean(data)`, `median(data)`, `mode(data)` and `std(data)` and compare to the Excel results.

3. As a group, do the following:
   a. Compare your individual means, medians, and modes. Resolve any discrepancies.
   b. Generate a histogram of the data, using a bin size of . Use your nicehist.m file you built before.
   c. Generate a histogram of the data, using a bin size of 5.
   d. Comment on the advantages/disadvantages of each bin size.

3 Measurement and Uncertainty II

I hope that by now you have been taught what a derivative is in your Math 112 class. But if not, we will learn what it is today. For our purposes, a derivative is a slope of a line. You should recognize the equation of a straight line as

$$y = mx + b$$

The slope m can be written as

$$m = \frac{dy}{dx}$$

This is nothing magic. It is just a strange way to write m. With the slope written this way, the equation of the line could be written as

$$y = \frac{dy}{dx}x + b$$

But why dy/dx? Think of how we find a slope of a line. Back in junior high school we called the slope the "rise over run." That is, the change in y-value divided by the change in the x-value.

$$m = \frac{y_2 - y_2}{x_2 - x_1}$$

In physics, we write the change in a variable using the greek letter delta, Δ. So we could write the slope as

$$m = \frac{y_2 - y_2}{x_2 - x_1} = \frac{\Delta y}{\Delta x}$$

We need to get used to this delta notation, so let me write out Δy

$$\Delta y = y_2 - y_1$$

and Δx.

$$\Delta x = x_2 - x_1$$

So our straight line equation should be written

$$y = \frac{\Delta y}{\Delta x}x + b$$

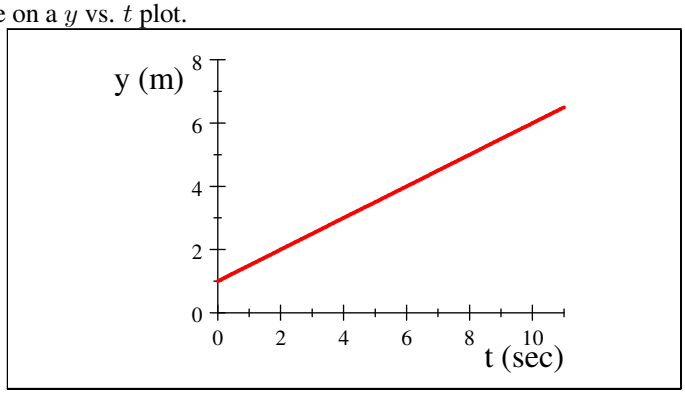
but if we take Δx to be very, very small it is customary to write the Δx as just dx (I guess a "d" is smaller than a "Δ"). If this is not familiar from Math 112, is should be by now from PH121 (if is not familiar at all, call me over for help, but still don't panic–we are just writing slope in a strange way).

In PH121 you have or will shortly learn that the velocity is the slope of the plot of x vs.

t, for example,

$$y = \frac{1}{2}\frac{\text{m}}{\text{s}}t + 1\,\text{m}$$

is an equation giving the y position of an object as a function of time. Note that it is a straight line on a y vs. t plot.

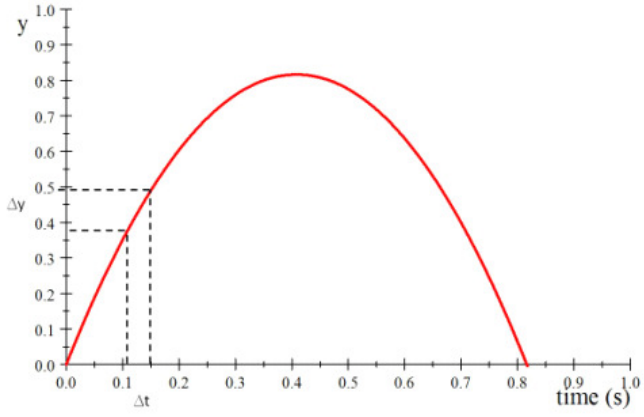


The slope of the line is

$$\frac{dy}{dt} = \frac{1\,\text{m}}{2\,\text{s}}$$

We can verify that this works by looking at the plot and noting that for every two units of time, we go up one position unit. The slope is $1/2\frac{\text{m}}{\text{s}}$.

But not all curves are straight lines. What do we do with curves that, well, curve? One idea is that we could split up the curve into little line segments, each with its own slope. We can think of dy/dt as an instantaneous slope, a slope of one of the tiny line segments that make up our curve. This is the sort of speed measurement that your speedometer gives. The speed might be different a short time later. But right now the speed is, say, $0.5\,\text{m/s}$.



Really, in defining an instantaneous slope we have assumed that the slope near our point

on the curve is essentially a straight line if Δt is small enough.

We can use this idea to interpret our error calculations. Suppose I throw a ball in the air with a initial speed of $4\,\mathrm{m/s}$ straight up starting from $y_o = 0$. From PH121 you have learned (or will soon learn) that the equation for predicting how high the ball will go is

$$y = y_o + v_o t + \frac{1}{2} a t^2$$

It says that starting at y_o the ball will go higher depending on the initial velocity, v_o, and the acceleration, a. That make sense.

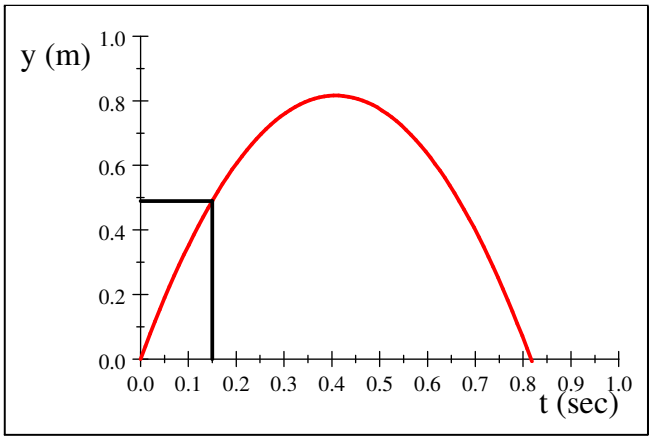
At a time, t, the ball should be at

$$y = 0 + 4\frac{\mathrm{m}}{\mathrm{s}} t - \frac{1}{2}\left(9.8\frac{\mathrm{m}}{\mathrm{s}^2}\right) t^2$$

where $a = -9.8\frac{\mathrm{m}}{\mathrm{s}^2}$ is the acceleration due to gravity. So, knowing this, I could predict how high the ball would go if I pick a particular time, say, $0.15\,\mathrm{s}$. The result should be

$$\begin{aligned} y &= 0 + 4\frac{\mathrm{m}}{\mathrm{s}}\,(0.15\,\mathrm{s}) - \frac{1}{2}\left(9.8\frac{\mathrm{m}}{\mathrm{s}^2}\right)(0.15\,\mathrm{s})^2 \\ &= 0.489\,75\,\mathrm{m} \end{aligned}$$

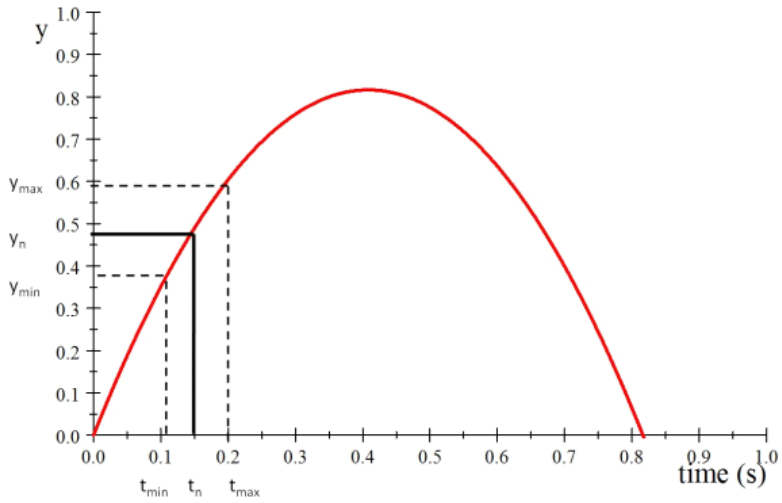
This is shown in the next figure with a black line. Solving the equation for y is equivalent to drawing a line up to the curve, then from our spot on the curve over to the y-axis to find the position.



For our case we plot a line upward from $0.15\,\mathrm{s}$ to the curve, and then plot a horizontal line from the intersection to the y-axis. We can see that we get $4.9\,\mathrm{m}$. Suppose I try to verify this by taking a picture of the ball in flight at $0.015\,\mathrm{s}$, but my stop watch is only good to ± 0.005 seconds. I try to take the picture when the watch is at $0.015\,\mathrm{s}$, but I might have taken the picture at $0.01\,\mathrm{s}$ or at $0.02\,\mathrm{s}$ or anywhere in between. My time has some uncertainty. What does the uncertainty in my stop watch time mean for the uncertainty in my y value?

We can get a good approximation by graphically drawing vertical lines up from $t_{\min}$

and $t_{\max}$ to the curve, and then extending horizontal lines from the intersections to the y-axis. This gives us a $y_{\min}$ and $y_{\max}$. Our actual height could be anywhere in between these. This is a way to view our uncertainty in y.



We can use this idea to find a general way to calculate uncertainties. We could define $\Delta t = t_{\max} - t_{\min}$. If our Δt is small enough (so we can write it just dt), the curve is essentially a straight line in the region between $t_{\min}$ and $t_{\max}$. So if we knew the slope of that line (the derivative dy/dt) we could easily figure out the $y_{\max}$ and $y_{\min}$ points to get our uncertainty range, at least if we stay near our t_n part of the curve. Recall that our uncertainty in y using the high/low method is

$$\delta y = \frac{y_{\max} - y_{\min}}{2} = \frac{\Delta y}{2}$$

Remembering that

$$y = \frac{dy}{dt}t + b$$

then

$$\begin{aligned} \Delta y &= y_{\max} - y_{\min} \\ &= \frac{dy}{dt}t_{\max} + b - \frac{dy}{dt}t_{\min} - b \\ &= \frac{dy}{dt}\Delta t \end{aligned}$$

From your reading, you will recognize this as almost the uncertainty in a function of one variable! But even if you don't recognize it, we can show that this is true using our high/low method. The quantity Δt is

$$\Delta t = t_{\max} - t_{\min}$$

so our uncertainty in t would be

$$\delta t = \frac{t_{\max} - t_{\min}}{2} = \frac{\Delta t}{2}$$

then

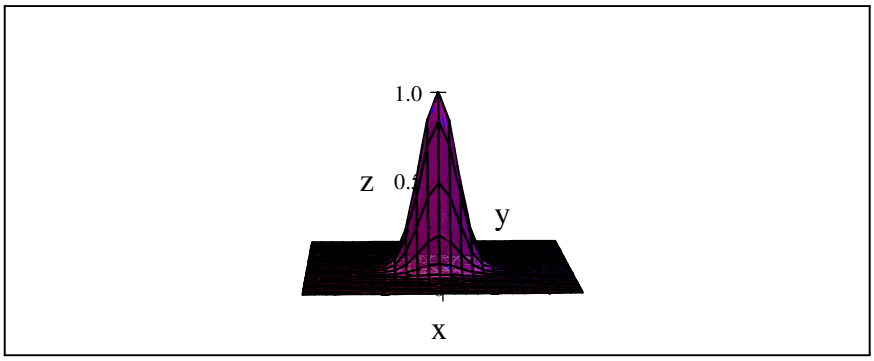
$$\begin{aligned} \delta y &= \frac{y_{\max} - y_{\min}}{2} \\ &= \frac{1}{2}\frac{dy}{dt}\Delta t \\ &= \frac{dy}{dt}\frac{\Delta t}{2} \\ &= \frac{dy}{dt}\delta t \end{aligned}$$

so

$$\delta y = \frac{dy}{dt}\delta t$$

So our uncertainty in y is just the slope at our point on the curve multiplied by our uncertainty in t.

But what if we have more than one variable? Say, we have a function $y(x, z)$, we essentially have a two dimensional slope. Think of a hill, you can go down a hill in more than one direction. So we need slope parts for each direction we can go.



$$\Delta y = \frac{dy}{dx}x + \frac{dy}{dz}z$$

But there is a fix we need to make to this equation that you won't learn for several math classes to come. We want to have a slope in the x and z direction, but we want the slopes to be independent (if you have already taken PH121, think of two dimensional motion problems, we split the problem into components). The notation for this is

$$\Delta y = \frac{\partial y}{\partial x}x + \frac{\partial y}{\partial z}z$$

where

$$\frac{\partial y}{\partial x}$$

means the component of the slope just in the x direction. We take a derivative of

the function y, but assume only x is a variable (treat z and all z terms with no $x's$ as constants). This lets us separate the x and z parts. A special, one variable derivative like $\partial y/\partial x$ is called a *partial derivative* because you only take one dimension of the derivative at a time. The reading from Chapter 3 uses this type of derivative to find the general formula for error propagation. If we wish to find the error in some general function $z\left(x,y\right)$ the error is given by

$$\delta y = \sqrt{\left(\frac{\partial y}{\partial x}\right)^2 \delta x^2 + \left(\frac{\partial y}{\partial z}\right)^2 \delta z^2}$$

This looks a lot like our slope equation What we are doing is to assume the function $y\left(x,z\right)$ is □at in a small region around the point we are studying. then the function has a slope $\partial y/\partial x$ in the x-direction, and $\partial y/\partial z$ in the y-direction. Each term like

$$\left(\frac{\partial y}{\partial x}\right)\delta x$$

gives how far off we could be in that direction (the x-direction in this case). Remember that we have assumed that $y\left(x,z\right)$ is essentially □at near our point of interest. The square root may be something of a mystery, but remember what you have learned or are learning about adding vectors in PH121. We add components of a vector to find the magnitude like this

$$V = \sqrt{V_x^2 + V_y^2}$$

This comes from the Pythagorean theorem. The x and y parts of the vector form two sides of a triangle. We want the remaining side. So we use the Pythagorean theorem to find the length of the remaining side.

We are doing the same for our small uncertainty lengths. We are just adding the x and the y components of the error. We could write our error formula for the general case of a function f, that depends on N different variables.

$$\delta f = \sqrt{\sum_{i=1}^{N}\left(\frac{\partial f}{\partial x_i}\right)^2 \delta x_i^2}$$

We will use this formula a lot, so make sure you understand what it means (ask your instructor for help if it is not clear).

How do we find the slope?

But now we have an equation in terms of slope written as dy/dx or dy/dz, but how would we ever find these slops? Your calculus class has or will teach you this. So we will just give you some quick formulas that will work for most equations, then you can ask your instructor if you have something odd like an $\arctan$ function.

For polynomials like,

$$f(x) = ax^2 + bx + c$$

each term follows the rule

$$\frac{d}{dx}(ax^n) = anx^{n-1}$$

that is, if I have a constant, a, times x^n the slope of this curve is the constant, a, times the power, n, times x to the $n-1$ power.

Let's take an example. What is the slope of the function $y = 5x^3$?

$$\frac{d}{dx}\left(5x^3\right) = (5)(3)x^{3-1} = 15x^2$$

How about finding the slope of $y = 7x^2 - 2x + 1$

$$\frac{d}{dx}\left(7x^2 - 2x + 1\right) = (7)(2)x^1 - (2)(1)x^0 + 0$$

The last term illustrates that the slope of a constant is zero. That makes sense. Constants don't change. So the change in y just due to the last term (1) should be zero. We also remember $x^0 = 1$. So we are left with

$$\frac{d}{dx}\left(7x^2 - 2x + 1\right) = 14x - 2$$

This one rule will take care of most of our functions for now. Let's try one more, say $y = x^{\frac{1}{2}}$

$$\frac{d}{dx}\left(x^{\frac{1}{2}}\right) = \frac{1}{2}x^{-\frac{1}{2}}$$

This could be written as

$$\frac{d}{dx}\left(\sqrt{x}\right) = \frac{1}{2}\frac{1}{\sqrt{x}}$$

So we see we can handle square roots with this rule.

Tie to statistics

We need to tie our statistical ideas into what we have learned about error propagation. Lets go back to our function $f(x, y)$ the error is given by

$$\delta f = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 \delta x^2 + \left(\frac{\partial f}{\partial z}\right)^2 \delta z^2}$$

but now we know we could express this in terms of standard deviations (provided you don't need to ensure all data are within your uncertainty range). We can write our uncertainties as

$$\sigma_f = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 \sigma_x^2 + \left(\frac{\partial f}{\partial z}\right)^2 \sigma_z^2}$$

We can use this to show that the standard deviation of the mean (the best estimate of our uncertainty) is given by

$$\sigma_{\bar{x}} = \frac{\sigma_x}{\sqrt{N}}$$

Think of calculating a mean value

$$\bar{x} = \frac{x_1 + x_2 + \cdots x_N}{N}$$

We can find the uncertainty in this function $\sigma_{\bar{x}}$

$$\sigma_{\bar{x}} = \sqrt{\left(\frac{\partial \bar{x}}{\partial x_1}\right)^2 \sigma_{x_1}^2 + \left(\frac{\partial \bar{x}}{\partial x_2}\right)^2 \sigma_{x_2}^2 + \cdots + \left(\frac{\partial \bar{x}}{\partial x_N}\right)^2 \sigma_{x_N}^2}$$

You see we just take the partial derivative of our function $\bar{x}$ with respect to each of the variables x_i and multiply by the uncertainty in that variable written now as a standard deviation σ_i.

For this special case, all of the x_i are the same (we are measuring the same value over and over in taking an average) and all of the σ_i are the same so we just have

$$\sigma_{\bar{x}} = \sqrt{N\left(\frac{\partial \bar{x}}{\partial x_1}\right)^2 \sigma_{x_1}^2}$$

and we can take the derivative using our rule. Only x_1 is a variable, so we can write the average $\bar{x}$ as

$$\bar{x} = \frac{x_1}{N} + \frac{x_2 + \cdots x_N}{N}$$

This is a polynomial! The first term is $\frac{1}{N}x_1$ and the whole second term is a constant if we take a partial derivitive with respect to x_1. The derivative is

$$\begin{aligned} \frac{\partial \bar{x}}{\partial x_1} &= \frac{\partial}{\partial x_1}\left(\frac{x_1 + x_2 + \cdots x_N}{N}\right) \\ &= \frac{1}{N}x_1^0 + 0 \\ &= \frac{1}{N} \end{aligned}$$

so our statistical error function is just

$$\begin{aligned} \sigma_{\bar{x}} &= \sqrt{N\left(\frac{1}{N}\right)^2 \sigma_{x_1}^2} \\ &= \sqrt{\frac{\sigma_{x_1}^2}{N}} \\ &= \frac{\sigma_{x_1}}{\sqrt{N}} \end{aligned}$$

or, since all the σ_{x_i} are the same, we can just write this as

$$\sigma_{\bar{x}} = \frac{\sigma_x}{\sqrt{N}}$$

Notice that in this example we had many x_i and that to find the uncertainty we just extended our equation from two variables

$$\sigma_f = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 \sigma_x^2 + \left(\frac{\partial f}{\partial z}\right)^2 \sigma_z^2}$$

to N variables

$$\sigma_f = \sqrt{\sum_{i=1}^{N} \left(\frac{\partial f}{\partial x_i}\right)^2 \sigma_i^2}$$

In this special case, we were trying to show a special result, but we can do this for any function with any number of variables. If your function is complicated, you just need to take more partial derivative terms under the square root.

Assignment

Measure the acceleration due to gravity, g, four different ways. For each case, determine an experimental value for g along with its uncertainty. Record how you find g and its uncertainty for each method in your lab notebook. Try to obtain the best value you can for each method.

Method 1: Timing a ball drop

Using a stop watch and a tennis ball, drop the ball over a known height and determine a value for g.

Method 2: Using a pendulum

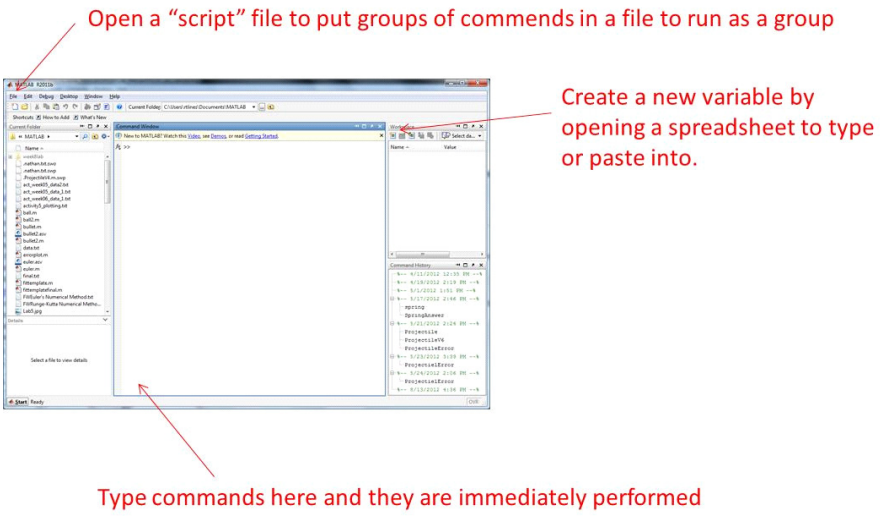
You will learn in PH123 that a pendulum oscillates back and forth at a certain rate. If you don't plan to take PH123, you still know that the pendulum of a grandfather clock sets the rate at which the clock will run. The time it takes the pendulum to go back and forth is called the *period of oscillation*. That period is given by the following equation

$$T = 2\pi\sqrt{\frac{L}{g}}$$

where for some reason the letter T stands for period, and L is the length of the pendulum string, and g is the acceleration due to gravity. Build your pendulum, and measure the period of oscillation using a photogate. From this obtain a value for g.

Plot Your Results

A spreadsheet program (e.g. MS Excel or LibreCalc) can graph data, and so can a piece of software on our lab computers named LoggerPro. You man know how to make a graph in one of these tools.
But we are learning MatLab this semester, and we need to know something about making graphs in MatLab.
Start MatLab. The result should look like this



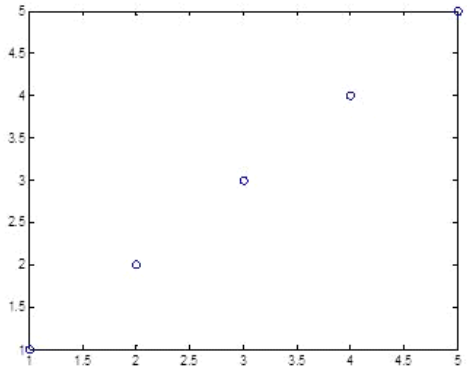
So far we have learned that MatLab is like a calculator programing language. You can type commands in the main command window and it will perform them very like your calculator. Let’s try a simple plot first. In the command window type

x = [1 2 3 4 5];

y=x;

plot (x, y, ’o’)

This should make a plot that looks like this



You could copy these lines and put them in a script like we did this last lab. The pressing the green arrow on the editor window will save and run the script. All the commands in the script will run one after the other.

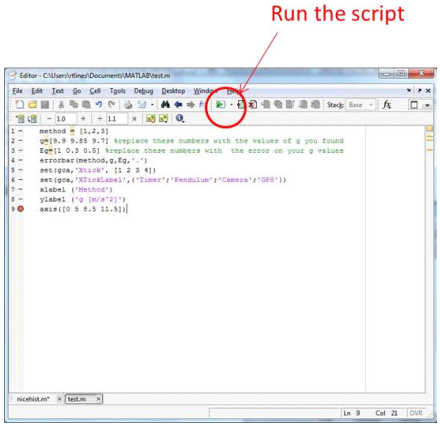
Let’s make a fancier graph that will plot our acceleration due to gravity data.

You could copy and paste the following lines into the command window and that will

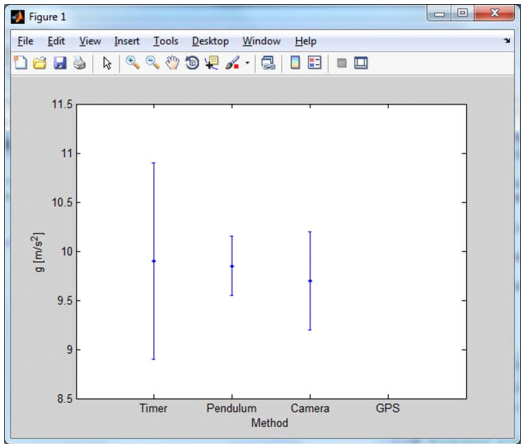
work just fine.

```
method = [1,2,3]
gdata=[9.9 9.85 9.7]   %replace these numbers with the values of g you found
Eg=[1 0.3 0.5]         %replace these numbers with  the error on your g values
errorbar(method,gdata,Eg,'.')
set(gca,'Xtick', [1 2 3])
set(gca,'XTickLabel',{'Timer';'Pendulum';'Camera'})
xlabel ('Method')
ylabel ('g [m/s^2]')
axis([0 5 8.5 11.5])
```

Alternately you could open a script file by choosing to create a new script from the file menu and pasting or typing the commands into the editor. I suggest you try this



You can run all the commands by clicking on the green arrow on the toolbar. MatLab will ask you to save the file. It needs a name (you should give it one instead of letting MatLab name it “untitled”) but MatLab is a little silly in that it insists you save your script in it’s chosen directory. After you get it working you can copy your file to your cloud or USB drive. You should see a graph like this, but the numbers will be different than mine.



These commands may seem mysterious now, but we will learn what many MatLab commands mean in the next few labs. So for now, being able to create a script to produce a graph is a good first step even if the commands are not all clear that do the graphing.

Let's see what each line of this "code" does.

The first line

method = [1,2,3]

defines a variable called "method." The variable is a vector or a one dimensional matrix that holds three values, 1, 2, and 3.

$$method = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

We will measure g using three different methods. This just gives us an index for each method we will use.

The next line does just about the same thing.

gdata=[9.9 9.85 9.7] %replace these numbers with the values of g you found

defines a variable called "gdata." The variable is a vector or a one dimensional matrix that holds three values, 9.9, 9.85, and 9.7.

$$gdata = \begin{pmatrix} 9.9 \\ 9.8 \\ 9.7 \end{pmatrix}$$

These are measurements of the acceleration due to gravity. You should replace these value with the ones from today's experiments. Note that we have used the "%" sign to allow us to write a reminder of what this line means. MatLab ignores anything on the line after a "%" sign. So we can make notes on what the line does after a "%."

The next line does just about the same thing again.

Eg=[1 0.3 0.5] %replace these numbers with the error on your g values

defines a variable called "Eg." The variable is a vector or a one dimensional matrix that holds three values, 1, 0.3, and 0.5.

$$Eg = \begin{pmatrix} 1 \\ 0.3 \\ 0.5 \end{pmatrix}$$

These are the uncertainties in the measured values of g. So you should replace these with your calculated uncertainties.

The next line

```
errorbar(method,gdata,Eg,'.')
```

Makes the plot with error bars. You can see that we gave this function our method, gdata, and Eg variables. The "." tells MatLab to plot a point at each of the gdata values.

The final two lines set up the vertical or y-axis.

```
ylabel ('g [m/s^2]')
axis([0 5 8.5 11.5])
```

The first labels the y-axis with $g(\,\mathrm{m}/\,\mathrm{s}^2)$ and the second sets up the range of values to display from $8.5\,\mathrm{m}/\,\mathrm{s}^2$ to $11.5\,\mathrm{m}/\,\mathrm{s}^2$ in increments of $0.5\,\mathrm{m}/\,\mathrm{s}^2$.

Of course you can change any of the axis labels are could even change the plot style (say, change the "." to a "o" and see what happens when you run the script again).

Method 3: High Speed Camera

Take high speed video of a falling ball. Include a meter stick or something of known length in your video. Use the *Logger Pro* software to analyze the video. The steps to do this are outlined in the Logger Pro help under "video analysis."

Fit a curve to your data that comes from the video. From you PH121 experience you know that the acceleration due to gravity is constant, so we can use the equation

$$y = y_o + v_o t + \frac{1}{2} a t^2$$

to indicate the type of curve to use for our fit. If you have trouble finding the curve fit function in Logger Pro, or have trouble using Logger Pro, call your instructor over.

The next two lines

```
set(gca,'Xtick', [1 2 3])
set(gca,'XTickLabel',{'Timer';'Pendulum';'Camera'})
```

format the horizontal axis. They tell matLab to use tick marks on the x-axis at locations 1, 2, and 3. then the next line labels the ticks "Timer", "Pendulum", and "Camera."

The next line

```
xlabel ('Method')
```

labels the whole x-axis "Method"

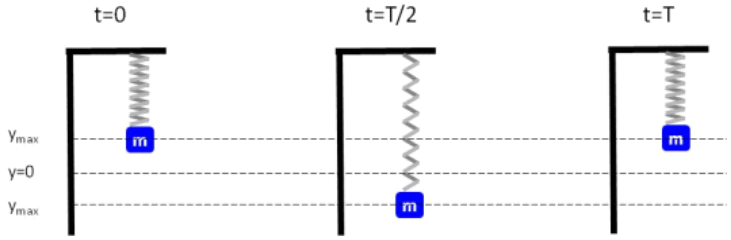
4 Experimental Design I: Harmonic Oscillators (masses and springs)

Introduction

We are going to tackle the subject of experimental design today, but it helps to have an experiment in mind. You are learning or have learned Hook's law for springs in your PH121 class. You understand that when we attach a mass to a spring and stretch or compress a spring we have a force

$$F_s = -kx$$

on the mass, where by k we mean the spring stiffness constant, and by x we mean the displacement from the equilibrium position of the mass-spring system. We can make such a system oscillate. This is really a PH123 problem, but let's pretend that we are scientists in Newton's day and we don't know much about oscillation (because most of us don't yet). We wish to find out more. We know that if we build a mass-spring system we can get oscillation and we define the time it takes for the mass to travel through one full oscillation (so the mass, say, starts from the highest point and it returns to it's highest point) as the *period* of oscillation and abbreviate it with the letter T.



Let's further pretend that you have read Hook's work and from this work have reason to believe that period might be proportional to the square root of the mass.

$$T \propto \sqrt{m}$$

You want to verify this report and build your own model for the period of oscillation of

a mass-spring system. You will use this model to make predictions, and by doing so, you will see how well your model works. This is what we want to investigate, now let's see how to design our experiment.

Designing an experiment

One of our objectives in this course is to learn how to design an experiment so that it will be successful.

Back in grade-school, an experiment was any science related activity (the proverbial building of a volcano model was considered an experiment). But for a scientist, an experiment is a specific thing. It is the testing of a hypothesis. You must test a hypothesis with care because the entire foundation of science depends on the integrity of how we do this testing.

In today's lab I will give you a hypothesis to test (the period of oscillation for a mass-spring system depends directly on the square root of the mass). The following steps will help your experiment be successful.

1. Identify the system to be examined. In our case it is a mass-spring system. We should identify all the *inputs* to the system. For example, we know there is a mass, m, and you have heard about a spring constant, k. There is also the force due to gravity and tension on a spring. These are inputs. You should describe your system in your lab notebook and list the inputs. These inputs are the things you can possibly change in the design of your experiment.

2. Identify the model to be tested. The word "model" means our mental picture of how something works. As physicists, we would prefer to express a model in a mathematical equation. For example, we have a model of how force depends on acceleration. The bigger the acceleration, the more the force. But our model also includes mass. The larger the mass, the larger the force needed to create the same acceleration. This mental model can be expressed in an equation

   $$F = ma$$

   It is valuable to use both the word description of the model as well as our mathematical representation. In our case today, our mental model is that period of oscillation for a mass-spring system depends directly on the square root of the mass. Think about what this means. If I increase the mass, the period should get longer. But if I double the mass, the period won't double. This is a mental model that allows us to do predictions of behavior. In physics it is almost required to reduce this model to an algebraic equation that can be used to calculate a prediction and an uncertainty on that prediction. For today's experiment that equation is

   $$T = C\sqrt{m}$$

   where C is a factor that does not depend on mass, but for your experiment that your group designs later in this course you will have to come up with your own mathematical expression of your model. Record your model and the model equation

in your lab notebook.

3. Plan how you will know if you are successful in your experiment. You are testing a hypothesis, and you are much more likely to succeed in your test if you plan what that success would look like. One way to do this is to plan how you will communicate your results. It is a great idea to think of what graph you will make at the end of the experiment to communicate whether your model works (or not). In today's experiment, a graph of T vs. m or even T vs.$\sqrt{m}$ might be useful along with a curve fit. Notice I am suggesting you plan this before you perform the experiment. I am not suggesting you decide on what the results will be, only how you will report them. This focuses your attention on deciding what measurement you will make. In our case today it is hard to plot T vs. m if you don't measure T and m, planning the graph in advance helps you plan the experiment. Mock up your graph or figure in your lab notebook. Give axis titles and even units (but of course no data yet).

4. Rectify your equation. It would be good to be able to use a curve fit to analyze our data. The strongest and most reliable curve fits are straight line fits where the fit equation is something like
$$y = mx + b$$
So if at all possible, we would like to reduce our equation to the equation of a straight line. In today's lab we can do this. We call this *linearizing the equation*. If we can't find a way to linearize the equation, we at least need to render our equation into a form that we can use to predict the outcome of our experiment. Record your new equation in your lab notebook.

5. Choose ranges of the variables. For today's experiment we might have several, but m and k are principal variables. It should be clear when you see the spring that putting a thousand kilograms of mass on the spring would be a bad idea. but how much mass is right? What will give you good results in testing your theory? Hooks law is not valid for all m and k (if you doubt this, think of your Christmas slinky after your brother got to it; it never looked the same again!). What values of m are best for performing the experiment? An error analysis based on your equation is invaluable in making this decision. Changes in mass that produce a change in T that is smaller than the uncertainty in T will not be noticeable. So taking measurements for such small mass changes would be a waste of time and effort. We would like to avoid this. Changes that are likely to break the equipment are also not desirable. And of course you want to plan this before you do the experiment and find that you did not get good data, and therefore must repeat all your work! Record your variable ranges in your lab notebook. As you perform the experiment note any deviations from this plan.

6. Plan the experimental procedure. As a group talk your way through the experiment. You might find yourselves saying something like "then you take the stopwatch and measure the period.." and you realize that you did not get a stop watch. For your experiment that you design, you need to find out in advance if we have the equipment you need. So get in the habit of working through the procedure in advance to see if you have forgotten anything. Record your planned procedure in

your lab notebook. As you perform the experiment, note any deviations from the plan and the reason for the deviation. Deviations are fine, just make sure you record them.

7. Perform the experiment and report on it in your lab notebook. This involves all the things we have been including in our lab notebooks to date

- Describing the goal for the work. (This is probably already done in your plan)
- Give predictive equations and uncertainties for the predictions based on the physical law. (This is probably already done in your plan)
- Give your procedure you actually followed, recording what you really did as you do it. This will probably not be just a restatement of the plan because things will change as you go. Record the equipment used and settings, values, etc. for that equipment. Did you learn how to use any new equipment? What did you learn that you want to recall later (say, when taking the final, or when you are a professional and need to use a similar piece of equipment five years from now).
- Record the data you used. . If you have a large set of values, you can place them in a file, and then record the file name and location in your lab notebook. Make sure this is a file location that does not change (emailing the data to yourself is still not a good plan).
- Give a record of the analysis you performed. You planned this above, now record what you actually did
- Give a brief statement of your results and their associated uncertainties.
- Draw conclusions: Do your results support the theory? Why or why not? What else did you learn along the way that you want to record. (This is where we may compare the percent error to our relative uncertainty).

Assignment

This week's assignment follows the experimental design process outlined above. I have mostly designed this experiment for you. So this week I want you to identify the design parts and put them in our design process order. For our next deign lab, you will have to design the experiment yourself. This week's lab is to get familiar with the process. Perform this experiment as a group.

First Part: Data Collection

1. Our system will be the mass-spring system and it's hanger. Obtain a set of weights, a spring, a weight hanger, a stand, and a stopwatch. Attach the spring to the stand, and the weight hanger to the spring. Determine the inputs to this mass-spring system that may affect the output quantity of interest (the period of oscillation). Determine whether each of these inputs will affect the period of oscillation. If so, explain how you will control for that input. If not, give justification for why you can ignore that input.

2. Build a mathematical model beginning with the suggestion you got from reading Hook's work (above).

3. Determine how you will measure the period of oscillation. Remember that you want to minimize the amount of uncertainty in your measurement. Techniques we have learned in previous labs may help. Record your method. You should plan any graphs you will make and in general plan how you will report your data and whether or not your experiment is successful.

4. Discuss how you might go about making your equation look linear by a proper substitution of variables. Explain why this might be useful.

5. Select a range of variables. (e.g. $m = 20\,\mathrm{g}$, $30\,\mathrm{g}$, $40\,\mathrm{g}$, $50\,\mathrm{g}$, and $100\,\mathrm{g}$). Don't use $80\,\mathrm{g}$ because I want to reserve this value for a special purpose below. Stop at about $100\,\mathrm{g}$.

6. Plan your procedure and record your plan in your lab notebook.

7. Perform the experiment. Your plan probably includes determining the period of oscillation for masses that you have selected. Be sure to record any measurement uncertainty. Make the graphs of your data that you planned including the appropriate error bars. Attach the graphs in your lab notebook. Record what you do and highlight any deviations from your planned procedure. Record your data or your data file name and location. Show your analysis and give your results. Draw conclusions. We will check these conclusions in the next part. But state whether you believe that $T \propto \sqrt{m}$

Hints on graphing:

1. a. It might help to plan to fit a smooth curve through your graphed points.
   b. You will need to tell your fitting routine what equation to use, so you should pick a theoretical mathematical form before you try a curve fit. Hopefully you linearized your equation, so the theoretical mathematical for is a straight line $y = mx + b$.
   c. In MatLab the fit routine is the `fit()` command. If our data are y and x then the command is `FO=fit(x,y,'poly1')`. The result is m and b, the values for the constants in the equation you are using $y = mx + b$ for a straight line which is a polynomial of order 1 (thus the `'poly1'` designation). The `fit` command also gives what MatLab calls a *confidence interval*. This we will interpret as the uncertainty in the fit parameters, but MatLab dies not give the uncertainty directly. Instead, the confidence interval gives the high and low points for m and b. But from lab 1 and the high/low method of calculating uncertainties we know what to do
   $$\delta m = \frac{m_{\max} - m_{\min}}{2}$$
   The fit parameters, and their high/low values are stored in the variable `FO`.
   d. You can check your fit equation by using your value of m and b to calculate a new set of y values `yfit=m*x+b` and then plotting them `hold on; plot(x,yfit); hold off;`. This should place a straight line through your plot of your data points if all went well.

Second Part: Interpolation and Extrapolation

We would like to test the equation or "law" you developed in the last part. We will use the equation to predict periods for masses you have not yet used.

1. **By interpolation, predict the period of oscillation for an** $80\,\mathrm{g}$ **mass.** Record your methods and results. Interpolation means to predict an output value (in this case, a period) for an input value that falls within the range of the input values you have used in your measurements. If you measured periods for $20\,\mathrm{g}$, $30\,\mathrm{g}$, $40\,\mathrm{g}$, $50\,\mathrm{g}$, and $100\,\mathrm{g}$, then $80\,\mathrm{g}$ is within this range. Using the curve fit equation generated by the data we measured, we can plug in $80\,\mathrm{g}$ and predict the period for our spring with an $80\,\mathrm{g}$ mass. This is interpolation. This will test our model to see if it works for new inputs. If it does not, our model is probably not good.

2. **By extrapolation, predict the period of oscillation for a** $300\,\mathrm{g}$ **mass.** Record your methods and results. Extrapolation means to predict an output value (in this case, a period) for an input value that falls outside the range of the input values you have previously measured. If you measured periods for $20\,\mathrm{g}$, $30\,\mathrm{g}$, $40\,\mathrm{g}$, $50\,\mathrm{g}$, and $100\,\mathrm{g}$, then $300\,\mathrm{g}$ is outside this range. Using the curve fit equation generated by the data we measured, we can input $300\,\mathrm{g}$ and predict the period for our spring with an $300\,\mathrm{g}$ mass. Extrapolation is more risky. The conditions of our experiment might change outside our range (think, in a limiting case, we could break the spring, and get an infinite period!). But if things are done carefully, this is also a test of the validity of our model.

3. Measure the period of oscillation for the $80\,\mathrm{g}$ and $300\,\mathrm{g}$ masses. Be sure to account for all uncertainties. Compare your measurements with your predictions, and comment on the level of agreement.

4. Now that we have tested our mathematical model for the relationship between period and mass for a mass-spring system, you can report it. Determine values for your constants, including uncertainties . Record your methods and results.

Third Part: Further Discussion

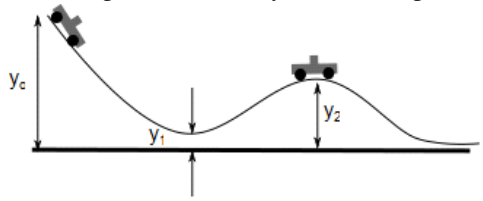
1. An often useful tool, especially when your data is not naturally linear, is to plot it on a logarithmic scale. Create such a graph using Logger Pro or Excel (or MatLab) and attach it to your lab notebook. For Logger Pro or Excel this is a graph axis option. The command in MatLab is `semilogy(x,y)` where x and y are the arrays of x and y data values. Comment on what you see.

2. Don't forget to make good comments on what you did and how you did it in your lab book.

5 Experimental Design II: Conservation of Energy

This week we will practice experimental design with a new context. I wont spell out all the steps, so your lab group and you will have to work through the experimental design steps.

Suppose you have been told that energy is conserved (I hope you have by now in PH121). This is our model–the idea that energy is conserved. That is, that it is never lost, just transferred from one form of energy to another. A colleague suggests a method to test this model. He builds a pine-wood derby track and a pinewood derby car.



Your colleague suggests to you that if energy is conserved, you should be able to predict the velocity of the car at points y_1 and y_2.

Another colleague steps in and suggests that you need to be concerned about the energy tied up in the rotational kinetic energy of the wheels. You may not have heard about this in your PH121 class yet. She says that is OK, because you can do a quick fix. She suggests that you should include a factor of about 10% of the translation kinetic energy. That is, compute the kinetic energy in this case as

$$K = (1.1)\,\frac{1}{2}mv^2$$

that should account for the wheel rotational kinetic energy.

Assignment

Part 1:

As a group design an experiment to test the model. Your design should include the steps from lab 4 (briefly repeated here)

1. Identify the system to be examined. Identify the inputs and outputs. Describe your system in your lab notebook.
2. Identify the model to be tested. Express the model in terms of an equation representing a prediction of the measurement you will make. Record this in your lab notebook. (If you have not solved this problem in your PH121 class yet, call me over and we will go through it together).
3. Plan how you will know if you are successful in your experiment. Plan graphs or other reporting devices. Record this in your lab notebook. For today's lab, I will provide photogates and the car and track. If you need other equipment, ask.
4. Rectify your equation if needed. Record this in your lab notebook.
5. Choose ranges of the variables. Record this in your lab notebook.
6. Plan the experimental procedure. Record this in your lab notebook.
7. Perform the experiment. Record this in your lab notebook. Go through all the steps of performing an experiment. End with a conclusion that clearly states whether your experiment supported the model, falsified the model, or, if neither was possible, try to explain why.

Part 2

Work on your proposals. They are due in two weeks. Here are some tips on writing your proposal:

Proposal writing:

A Proposal is a document that is intended to persuade someone (your professor, funding agency, yourself, etc.) that you should be given the resources and support to perform the experiment. For our class, the proposal consists of the following parts:

1. Statement of the experimental problem
2. Procedures and anticipated difficulties
3. Proposed analysis and expected results

4. Preliminary List of equipment needed

Note that most of the steps involved in planning an experiment are contained in these for parts of the proposal. Each of the steps is explained in more detail below.

Statement of the experimental problem: This is a physics class, so our experiment should be a physics experiment. The job of an experimental physicist is to test physics models. So your statement of the experimental problem should include what model you are testing and a brief, high level, overview of what you plan to do to test this model.

Procedures and anticipated difficulties: Hopefully, your reader will be so excited by the thought of you solving your experimental problem that he or she will want to know the details of what you plan to do. You should describe in some detail what you are planning. If there are hard parts of the procedure, tell how you plan to get through them. This is essentially steps 1-6 of our experimental design strategy.

Proposed analysis and expected results: You might think this is unfair, how are you supposed to know what analysis will be needed and what the results should be until you take the data? But really you both can, and should make a good plan for your data analysis and figure out what your expected results should be before you start taking data. After all, you have a model you are testing. You can encapsulate that model into a predictive equation for your experiment. Then you can use that predictive equation to obtain predicted results and uncertainties. Using this, you can design your experimental apparatus by putting in the numbers from your experimental design and seeing what the outcome should be. You can see if there is a chance that your experiment will measure what you want with the equipment you have (this is where our differential form of error calculation comes in).

If you don't do this, you don't know what equipment you will need or how sensitive that equipment needs to be. If you are trying to measure the size of your text book, an odometer that only measures in whole miles may not be the best choice of equipment. This might be obvious, but depending on how well you need to measure your text book, a ruler may not work either. You don't know until you have an estimate of the uncertainty. So to know what you need, do the calculations in advance with your range of inputs as the values you take for the prediction.

Of course this means you must include a predictive calculation of the uncertainty. Uncertainty in your result is governed by the uncertainty inherent in the measurements you will take. The uncertainty calculation tells you what sensitivity you will need in your measurement devices. In our text book case, you could see immediately that

you need a different apparatus than the odometer. You might also find our ruler to be problematic depending on what precision you need.

I remember a time in my career when the US Naval Research Labs asked us to build a microwave radiometer to measure the sea wind direction from space. We spent some time and our predicted analysis and uncertainty said that it would be a very expensive instrument to be able to successfully measure the wind direction—it would take more money than they were offering. NRL disagreed and built the device themselves at less cost, but to lesser specifications with much greater uncertainties. Then they spent a billion (yes, a billion with a "b") dollars to launch the device into space. The device was a total failure. The uncertainty was so big that the data was totally useless. We want to find out whether our experiment will work before we risk our grades (or a billion dollars) on it. So we will do the prediction ahead of taking the data.

You should do all of this symbolically if you can, numerically if you must, but almost never by hand giving single value results. Some measurements will come back poorer than you anticipated, or some piece of equipment will be unavailable. You don't want to have to redo all your calculations from scratch each time this happens. For example, in the event of an equipment problem, your analysis tells you if another piece of equipment is sufficiently sensitive, or if you need to find an exact replacement. When I perform an analysis like this, I try for a symbolic equation for uncertainty. I like to program these equations into Scientific Workplace, or Maple, or SAGE, or MathCAD, or Mathmatica or whatever symbolic math processor I have. Then, as actual measurements change, I instantly get new predictions. In the absence of a symbolic package, a spreadsheet program will do fine (and we have Excel on our computers). A numerical program also is quick and easy to re-run with new numbers when no symbolic answer is found.

Preliminary List of equipment needed

Once you have done your analysis, you are ready to list the equipment you need and the sensitivity of the equipment you need (that is, list the uncertainties you need to achieve). Final approval of the project and the ultimate success of your experiment depends on the equipment you choose or are granted. You want to do a good job analyzing so you know what you need, and a good job describing the experiment so you are likely to have the equipment you want available when you start.

Performing the experiment

Once your proposal is accepted, I will provide you with the equipment we have agreed upon from your proposal. You will have three weeks to perform your experimentation. I will be available for advice and to watch for problems or safety issues. But you and your team will perform the experiment. **You will want to keep good notes in your lab**

notebook. You will likely have to change your procedure after you start because of problems. Take careful note of what was actually done, and what your measurements were. Give the reason for the change. Note any unusual things that happen. **Carefully record what you do.**

Written report

The written report is designed to match a normal format for an applied physics article in a journal like *Applied Optics* or the *IEEE Transactions* journals. It is useful to know now how I will grade the report later so you can make sure you design in all the parts I will look for. There should be an introduction, description of the procedure, description of the data and results, a description of the analysis, and a conclusion. These sections are described in detail in the following table.

Experimentation is a lot of fun if done right. It can be frustrating and discouraging if not done well. Our goal is to learn how to perform and report on an experiment, so that is what will be graded. If you show something new and interesting, that is just more fun. If you show that your original model was not correct–that is science! If you have done a good job designing and reporting your experiment, a negative result is just as good as a positive result.

Section/Value	50-40pts	40-30 pts	30-20 pts	20-0 pts
Introduction: **Answers the question "what is this lab about?"**	• Answers the question "what is this lab about?" sufficiently that a person who did not perform the lab would understand • Gives enough background so that the lab report makes sense as a stand-alone document • Tells the reader what your expected outcome is based on theory.	• Answers the question "what is this lab about?" sufficiently that a person who was part of your lab group would understand • Gives enough background so that the lab report makes sense to someone who knows the lab topic well	• Mentions what the lab is about • Gives some background	• It is difficult to tell from the introduction what the lab is about • Little or no background provided
Procedure: **Answers the question "what did you do?"**	• This section answers the question "what did you do?" sufficiently so a non-expert can understand what was done. • Describe the entire procedure, especially indicate any deviations from your plan and explain why those deviations were necessary.	• This section answers the question "what did you do?" sufficiently so your lab partner could understand what was done. • Tells where you deviated from the plan	• Major points of the procedure are listed	• It is difficult to tell what you did from your description
Data: **Answers the question "what did you measure?"**	• Each measured value is given with units • Each value is given with a good estimate of uncertainty • Only measured values that are needed are given • The data is presented in a way that is easy for the reader to find and read. (e.g. label graphs and table columns)	• Each measured value is given with units • Each value is given with an estimate of uncertainty • Extra values that were not needed are given	• measured values are given	• It is not clear what you measured
Analysis: **Answers the question "how did I get from my data to my results?"**	• It is clear how you got from your measured values to your results • Major equations are given and discussed. • The method of determining uncertainties is discussed	• It is possible to tell how you got from your measured values to your results • Major equations are given • The method of determining uncertainties is discussed	• It is possible to tell how you got from your measured values to your results • Major equations are given • Method of determining uncertainty is not discussed	• It is not possible to tell how you got from your measured values to your results • Major equations are missing • Method of determining uncertainty is not discussed
Results: **Gives the results of your analysis**	• There is a clear, understandable answer to the question the lab asks. For example, if I ask you how fast a car is going, the result would be a calculated speed, with its calculated uncertainty and units. • Report percent error or percent difference • Report fractional uncertainty	• There is a an answer to the question the lab asks with uncertainty and units • Report percent error • Report fractional uncertainty	• There is a an answer to the question the lab asks • uncertainty and units are missing • Percent error or fractional uncertainty is missing	• There is no clear answer to the question the lab asks • Percent error or fractional uncertainty is missing
Conclusion: **Answers the question "did the experiment show what was intended?"**	• There is a clear discussion of whether the experiment was supported or falsified the theory. • This discussion includes a comparison of the percent error and fractional uncertainty • If there were difficulties, they are discussed here • There is a statement of what you learned from this experiment. Note any problems and how you would resolve them if you were to redo this experiment.	• There is a general discussion of accuracy (often with percent errors quoted) • There is some mention of whether the predictive theory is supported • Problems are noted and how you would resolve them if you were to redo this experiment is discussed.	• There is no comparison of the percent error and fractional uncertainty • There is a statement of what you learned from this experiment.	• There is no outcome of the accuracy of the experiment • There is no comparison of fractional uncertainty and percent error • There is no clear conclusion about the predictive theory • There is little mention of what was learned

6 Experimental Design III: Standing Waves on Strings

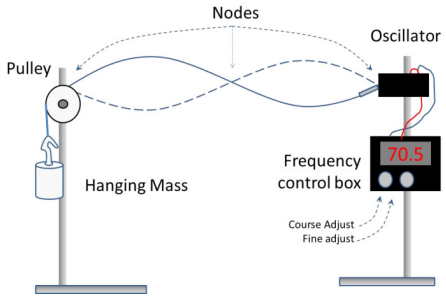
Introduction

Standing waves result from making waves that re□ect back on themselves, or by making waves on both ends of a string. When we were children we formed a type of standing wave with jump ropes.

This standing wave had a part that went up and down in the middle and two parts that did not move much on each end (called nodes). But if you had a bored kid, you might have seen a standing wave that looks like this.

This is not so good for jumping, but makes an interesting picture. The part in the middle that does not seem to be moving is called a node. Really there are also nodes on each end of the rope as well. So altogether there are three nodes in this picture. We can make standing waves that have many nodes. If you try this with a jump rope, you will find that the more nodes you have, the faster you have to shake your end of the rope. Another way to say this is that the frequency of your wave you are making must increase with the number of nodes. This is part of today's model.

In the setup on your table, ring stands are holding strings, and there is an oscillator on one end that has a frequency control box. The other end has a pulley and a hanging mass to provide tension on the string.



Experimentally we find that not all frequencies will make standing waves. So our model includes the idea that only some frequencies produce standing waves. Our model also includes the idea that the weight of the string will change which frequency will make a standing wave. If you play a stringed instrument, you may have noticed that some strings are thicker than others. Thicker strings have different standing wave frequencies than thinner strings. If you study this model (some of you will in PH123), you can derive an equation that tells which frequencies will work

$$f = \frac{n}{2L}\sqrt{\frac{Mg}{\mu}} \tag{6.1}$$

where n is an integer ($n = 1, 2, 3 \ldots$). This integer for strings is the number of nodes minus one $n = n_{nodes} - 1$. So you can form a standing wave, then count the places that don't seem to move (remember the ends!) and subtract one to find n. The frequency that creates the standing wave should be a function of n. **Our model tells us that if we know n, we should be able to predict the frequency**. You will find that for each way you can make a standing wave, a small range of frequencies will make the standing wave, not just one, single frequency. But the frequency that produces the largest standing wave is the one we want (biggest amplitude–or the one for which the wave looks bigger) . That was the f that was included in forming our model equation. Of course there are other variables in our equation, so we should find out what they are. The quantity, μ, is the linear mass density. It is defined as mass of the string divided by the length of the string. So μ tells us how massive the string is.

The quantity, L is the length of the string that is participating in the waving, $g = 9.8004\,\mathrm{m/s^2}$ is the acceleration due to gravity, and M is the hanging mass tied to the end of the string beyond the pulley.

One way we could verify our model equation is to use it to predict one of the input values. Let's use μ. **The idea is to use our model equation to somehow find μ and then measure μ to see if the model equation prediction is good.**

It might be tempting to just solve the above equation for μ and report the answer from

one measurement. And of course that will work. But since this is a final exam I want to see if you can use some of the things we learned.

We learned earlier that we can take more than one measurement, and use those measurements together with a curve fit to solve for a fit parameter. I want you to do this. The quantity μ should be in one of the fit parameters. Then you can solve for μ using the fit parameter given by your MatLab linear fit code, or LoggerPro (using your code gives the most points). This is a more robust way to find μ, and it is the way I want you to proceed (Even if you solve for μ several times and take a mean and standard deviation–it will work and it is a good experimental technique–but I want to see if you can find and use your linear least squares code, so using your code gives the most points).

You may have to adjust the amplitude knob on the frequency controller for some frequencies to keep the apparatus from shaking itself apart. The frequency controller has a fine and a course frequency adjust knob, and a digital frequency display.

Assignment

As a group design an experiment to test the model. Your design should include the steps from lab 4.

1. Identify the system to be examined. Identify the inputs and outputs. Describe your system in your lab notebook.
2. Identify the model to be tested. Express the model in terms of an equation representing a prediction of the measurement you will make. Record this in your lab notebook. (If you have not solved this problem in your PH121 class yet, call me over and we will go through it together).
3. Plan how you will know if you are successful in your experiment. Plan graphs or other reporting devices. Record this in your lab notebook. For today's lab, I will provide photogates and the car and track. If you need other equipment, ask.
4. Rectify your equation if needed. Record this in your lab notebook.
5. Choose ranges of the variables. Record this in your lab notebook.
6. Plan the experimental procedure. Record this in your lab notebook.
7. Perform the experiment. Record this in your lab notebook. Go through all the steps of performing an experiment. End with a conclusion that clearly states whether your experiment supported the model, falsified the model, or, if neither was possible, try to explain why.

7 Numerical modeling of Projectile motion

There are times when we would like to predict the motion of an object, but we would like to make a computer do the hard work involved in solving the equations to get a numeric answer, so we don't have to do it. For example, we might want to calculate the position of a satellite as it orbits. Or we might want to calculate the position of the planets as they orbit the Sun. For simple cases, we might be able to do this by hand, but predicting by hand where all the planets will be on July 4, 2200 might get tedious without a computer.

As physics students, it is good to know how to approach solving problems on a computer.[1]

Let's start with some simple problems that we know how to do algebraically so we can tell if the computer solution is reasonable by comparing to the exact answer. Then we will tackle things we can't do algebraically.

Consider a ball moving under constant acceleration in one dimension. This is a kinematics problem. So we can use the kinematic equations you have learned (or will very shortly learn) in PH121. Let's remind ourselves of what these are:

$$\begin{aligned} x &= x_o + v_o t + \frac{1}{2} a t^2 \\ v &= v_o + at \\ v^2 &= v_o^2 + 2a\left(x - x_o\right) \\ x &= x_o + \frac{1}{2}\left(v_o + v\right) t \end{aligned}$$

The first equation tells us the position, x, as a function of time. This could be position in any direction, up and down, or side to side. If it is a ball being shot into the air, it might be better to write y instead of x. But for now, we will use just x for position in any direction. What this first equation tells us is that, once the ball is moving, we can find the new position of the ball by starting with the initial position of the ball, x_o, and adding to this initial position how far it has gone in t seconds.

[1] Most calculators qualify as computers now, since they are usually programmable. But we are talking about going beyond the built in functions of calculators or even spreadsheet functions on computers.

$$x = x_o + v_o t + \frac{1}{2} a t^2$$

Where we started

How far we went in t seconds

If the ball were going at constant speed, the additional distance would be just $v_o t$. This comes from speed being distance over time.

$$\begin{aligned} v &= \frac{d}{t} \\ d &= vt \end{aligned}$$

But our ball is accelerating, so we have to add in a little more distance because our speed is changing. That is what $\frac{1}{2}at^2$ does.

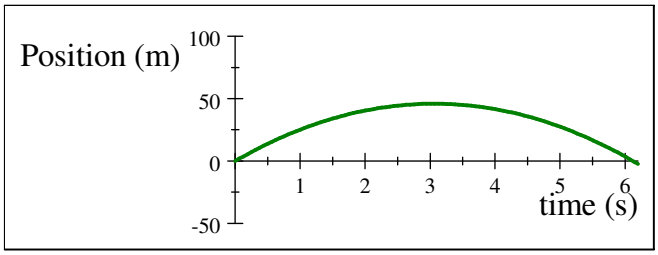
$$x = x_o + v_o t + \frac{1}{2} a t^2$$

Where we started

How far we would go at a constant speed

How much farther we went because we are accelerating

Since the position is a function of t^2, we know that the position vs. time graph for our ball will be a parabola.



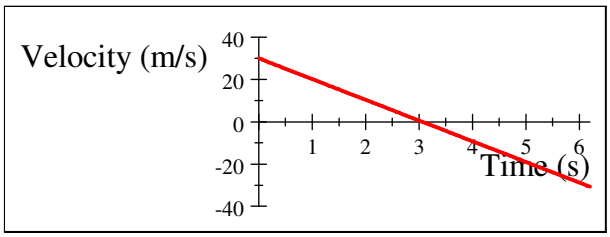
The second and third equations from our kinematic set give the speed of the object. The second is the speed as a function of time. Since for this first problem in programming we will only allow one dimensional motion (at first) we could say this is the velocity of the object and allow negative values to mean going the opposite way of positive values. Because the ball is accelerating, the speed will change. We can see that the velocity changes linearly with time. Think of a straight line

$$y = mx + b$$

our velocity equation is

$$v = at + v_o$$

on a velocity vs. time graph, this has the form of a straight line. So the velocity vs. time graph will be a straight line.



Our goal is to get the computer to predict this motion. To do this we will use an idea from calculus[2] that we can take an infinitesimally small amount of time for our experiment. We will call this very small amount of time a "time step" and label it Δt. Over this time interval, the change in velocity is essentially zero, that is, over our very small time interval, we can take $a \approx 0$, then our first kinematic equation becomes

$$x_f \approx x_i + v_o \Delta t$$

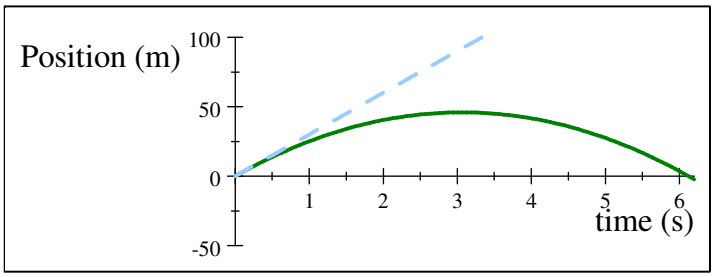
Remember that we have only let our experiment run for a very small time, Δt, so even if the velocity changed, it would not have changed much. So our equation will nearly work. In the limit as $\Delta t \rightarrow 0$ our equation will be exact.

Now suppose I want to go an additional small time interval, a second mini-experiment. That is, I let the experiment keep going now starting from the end of the first mini-experiment from the specific time $t_1 = \Delta t$ and end this second mini-experiment at the time $t_2 = t_1 + \Delta t$ (that is the second mini-experiment runs from times Δt to $2\Delta t$). We know where the ball is at the end of the first mini-experiment that lasted Δt seconds. But now we let the ball keep falling. Where will it be after a second Δt seconds? We can predict that the next position will be

$$\begin{aligned} x(t_2) &= x(t_1) + v(t_1)\Delta t \\ x(t + \Delta t) &= x(t_1) + v(t_1)\Delta t \end{aligned}$$

where $x\,(t_1)$ is the position at the end of the first time interval (our old x_f). Likewise, $v\,(t_1)$ is the velocity at the end of the first time interval. At this point, we have to acknowledge that the speed of the ball is really changing. Otherwise the ball's position vs. time graph will go in a straight line as shown in the next figure

[2] Not to worry if you are concurrently taking calculus. We will use very little calculus in this reading, so little that you probably won't notice it.



The dotted line is what we get when we don't change the velocity. The solid line is what we should have. But we don't yet know $v\,(t_1)$, we will have to find it.
We can find $v\,(t_1)$ using the second of our set of kinematic equations. We can modify the equation

$$v = v_o + at$$

for the first time interval,

$$v(t_1) = v_o + a\Delta t$$

where $v(t_1)$ means that v is a function of t and we have evaluated v at t_1. This last equation says that at time, Δt, the ball is really going at velocity $v\,(t_1)$ where we have to add (or subtract, depending on the sign of a) the change in velocity due to the acceleration. That change is velocity is $a\Delta t$ at the end of the first mini-experiment.
At the end of the second mini-experiment the velocity will have changed even more. I can write this as

$$v(t + \Delta t) = v(t_1) + a(t_1)\Delta t$$

where again $v(t)$ means that v is a function of t, and $a\,(t)$ means that a is a function of t.In our case, we know the acceleration is $a\,(t) = -g$, and we know that it really does not depend on time, but let's see how we would find the acceleration if we did not know this. Let's take Newtons's second law. In one dimension,[3]. This may be something you have not seen yet in PH121, so let me give you Newton's second law. It simply states that if you add up all the pushes and pulls on an object, the amount of push or pull left over after opposing pushes cancel is what makes the object change it's motion. In physics, we call pushes and pulls "forces." In a tug-o-war, both sides pull, but they pull in opposite directions. The winning team has to pull a little bit harder than the loosing team. This difference in the pull of the two teams is called the net pull or the net force. That difference in the forces is what makes the rope accelerate. For our ball, we can write (in calculus notion)

$$\Sigma F_x = \frac{d}{dt}\,(mv) = ma$$

This means to sum up all the forces in the x-direction and it says that that sum will be

[3] OK, I know it is weird that my x-axis is vertical. Usually the vertical axis is the y-axis. But that is just tradition, we can pick our axes any way we want. And I wanted x to go up and down.

equal to the mass of the ball multiplied by the acceleration. If we know the mass of the ball and the sum of all the pushes and pulls, we can find the acceleration. If there is no air resistance, all we have is the force of gravity pulling on our moving ball.

$$ma = -F_g$$

Then

$$ma = -mg$$

or

$$\begin{aligned} a &= \frac{-mg}{m} \\ &= -g \end{aligned}$$

So our ball's acceleration is just the acceleration due to gravity.

$$a\left(t\right) = -g$$

which is not time dependent. This is not very surprising. But what is important here is the method for obtaining the acceleration, and this method is general—we can use it in more complicated situations. We will do this each time we need to find an acceleration for our systems.[4]

For our falling ball, then, we could write our second kinematic equation as

$$v(t + \Delta t) = v(t_1) - g\Delta t$$

But note that in our computer formulation, we did not have to assume the acceleration was constant. In our problem the acceleration *is* constant, but in the problems to follow, it will not be! Our method can handle non-constant acceleration. That is more than the kinematic equations can do. This is exciting!

At this point we have two equations that calculate a new position and velocity forward in time, based on what the ball's position and speed currently are:

$$\begin{aligned} x(t_2) &= x(t + \Delta t) = x(t_1) + v(t_1)\Delta t \\ v(t_2) &= v(t + \Delta t) = v(t + \Delta t) = v(t_1) + a(t_1)g\Delta t \end{aligned}$$

So far we have only talked about calculating two positions (two mini-experiments), but we can use these equations to get a general expression for the next position, and the next, and the next to as many mini-experiments as we want.

$$\begin{aligned} x(t + \Delta t) &= x(t) + v(t)\Delta t \\ v(t + \Delta t) &= v(t) + a(t)g\Delta t \end{aligned}$$

These mean to take the position at the current time, $x(t)$, and add the current speed, $v\left(t\right)$ times Δt to get the next x position $x\left(t + \Delta t\right)$. And also take the speed at the current time, $v(t)$, and add the current acceleration, $a\left(t\right)$ times Δt to get the next speed,

[4] For those who are taking PH 121 this semester, have a more experienced student work with you on this part until your PH121 class catches up to net force.

$v\left(t+\Delta t\right)$. By doing this, over and over we can find the position of the ball at each time for a whole ball □ight.

You will notice that these expressions are coupled: the new position depends on the previous velocity, and the new velocity depends on the previous position. We have to use the equations together. To find the solution we will take a small step in time, Δt, and calculate a new position. We will also calculate a new speed. Then we will use the new speed to find a new new position an additional Δt later, and the new new position to find a new new speed. We step these two quantities forward in time one Δt at a time. We can repeat the process indefinitely.

Let's try an example. Suppose we throw a ball straight up with an initial velocity of $v_o = 30\,\mathrm{m/s}$ and an initial position of $x_o = 0$ (measured vertically) at $t_0 = 0\,\mathrm{s}$. Let's take time steps of size $\Delta t = 0.001\,\mathrm{s}$. We can calculate the next position at $t_1 = \Delta t$

$$\begin{aligned} x(t_1) &= x(t_0) + v(t_0)\Delta t \\ x(t_1) &= 0 + 30\frac{\mathrm{m}}{\mathrm{s}} \times 0.001\,\mathrm{s} \\ &= 0.03\,\mathrm{m} \end{aligned}$$

our ball has moved to up to the position $x = 0.03\,\mathrm{m}$. Our ball will have changed speed. We can find the new speed as

$$\begin{aligned} v(t_1) &= v(t_0) + a(t)\Delta t \\ v(t_1) &= 30\frac{\mathrm{m}}{\mathrm{s}} - 9.8\frac{\mathrm{m}}{\mathrm{s}^2} \times 0.001\,\mathrm{s} \\ &= 29.99\frac{\mathrm{m}}{\mathrm{s}} \end{aligned}$$

as we would expect, the ball has slowed down a little. In the next time interval, $t_2 = t_1 + \Delta t$, the ball will move to the new position

$$\begin{aligned} x(t_2) &= x(t_1) + v(t_1)\Delta t \\ x(t_2) &= 0.03\,\mathrm{m} + 29.99\frac{\mathrm{m}}{\mathrm{s}} \times 0.001\,\mathrm{s} \\ &= 0.059\,99\,\mathrm{m} \end{aligned}$$

Notice that we used the new velocity at t_1 to find this second position at t_2. We need to find a new velocity at this position

$$\begin{aligned} v(t_2) &= v(t_1) + a(t_1)\Delta t \\ v(t_2) &= 29.99\frac{\mathrm{m}}{\mathrm{s}} - 9.8\frac{\mathrm{m}}{\mathrm{s}^2} \times 0.001\,\mathrm{s} \\ &= 29.98\frac{\mathrm{m}}{\mathrm{s}} \end{aligned}$$

We can use this to find the third position

$$\begin{aligned} x(t_3) &= x(t_2) + v(t_2)\Delta t \\ x(t_3) &= 0.059\,99\,\mathrm{m} + 29.98\frac{\mathrm{m}}{\mathrm{s}} \times 0.001\,\mathrm{s} \\ &= 0.089\,97\,\mathrm{m} \end{aligned}$$

and the velocity at this position

$$\begin{aligned} v(t_3) &= v(t_2) + a(t_2)\Delta t \\ v(t_3) &= 29.98\frac{\mathrm{m}}{\mathrm{s}} - 9.8\frac{\mathrm{m}}{\mathrm{s}^2} \times 0.001\,\mathrm{s} \\ &= 29.97\frac{\mathrm{m}}{\mathrm{s}} \end{aligned}$$

We can keep marching along, one Δt at a time finding new positions, then finding new speeds, then using the new speed to find a new position, and finding the new position to find a new speed, and the new speed to find a new position, etc.
There is a typical short hand for these equations

$$\begin{aligned} x_{n+1} &= x_n + v_n\Delta t \\ v_{n+1} &= v_n - g\Delta t \end{aligned}$$

What this means is that we have labeled each new x and v by how many time steps, Δt, we have taken. The first Δt will be $n = 1$, the second $n = 2$, and so on. Just to be clear, let's compare

$$\begin{aligned} x(t + \Delta t) &= x(t) + v(t)\Delta t \\ x_{n+1} &= x_n + v_n\Delta t \end{aligned}$$

This means that we start with the initial velocity, v_0 at the initial position x_0. We use these to calculate the position x_1 and the speed v_1 a time $t = \Delta t$ later.

$$\begin{aligned} x_1 &= x_0 + v_0\Delta t \\ v_1 &= v_0 - g\Delta t \end{aligned}$$

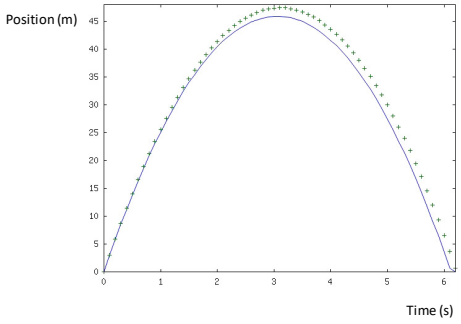
Then we start over with the new initial velocity, v_1 at the initial new position x_1 and use them to calculate the position x_2 and the speed v_2 an additional time Δt later ($t = 2\Delta t$).

$$\begin{aligned} x_2 &= x_1 + v_1\Delta t \\ v_2 &= v_1 - g\Delta t \end{aligned}$$

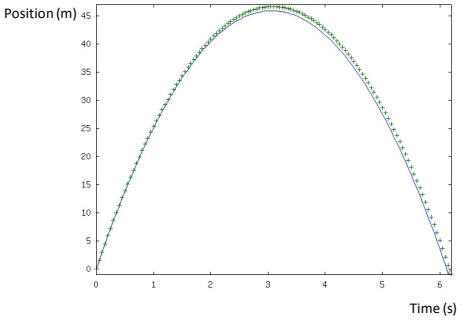
Then we use x_2 and v_2 to find x_3 and v_3

$$\begin{aligned} x_3 &= x_2 + v_2\Delta t \\ v_3 &= v_2 - g\Delta t \end{aligned}$$

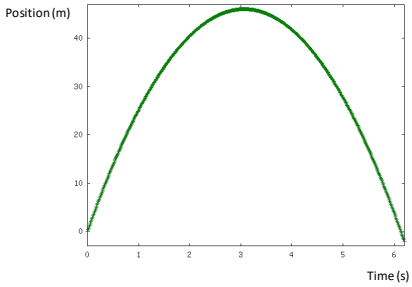
and so forth. Here is what we get after many iterations:



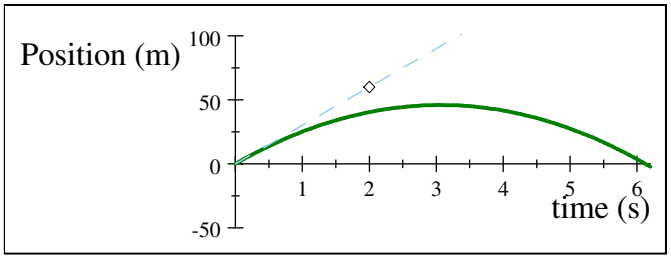
the solid line is the result of the kinematic equation (the correct result). The little plus signs are our computer solution. It is not too bad, but it is not too good either. Of course, if we let our Δt get smaller, we will have a better solution.



Notice that there are many more small plus signs now. We had to do more calculations to get the improvement. This is the cost of the smaller Δt. If we let Δt be very small the curves lie right on top of each other.



Euler came up with this method back in the 1700's and we still call this method of numerically finding results *Euler's method* (pronounced like"oiler").
Let's take a moment and see what went wrong when Δt was not small enough.



In the figure, we can see a solid curve (green if you are seeing this in color). Let's suppose the solid curve represents the actual position of the ball as a function of time. Lets also suppose $\Delta t = 2\,\mathrm{s}$. If we use the method we have described to find the position as a function of time, we will get an x_1 that is shown as a diamond on the graph. We have

$$\begin{aligned} x_1 &= x_0 + v_0 \Delta t \\ v_1 &= v_0 - g\Delta t \end{aligned}$$

if we let $v_o = 30\,\mathrm{m/s}$ and $x_o = 0$ like before, we get

$$\begin{aligned} x_1 &= 0 + 30\frac{\mathrm{m}}{\mathrm{s}} \times 2\,\mathrm{s} = 60\,\mathrm{m} \\ v_1 &= 30\frac{\mathrm{m}}{\mathrm{s}} - 9.81\frac{\mathrm{m}}{\mathrm{s}^2} \times 2\,\mathrm{s} = 10.38\frac{\mathrm{m}}{\mathrm{s}} \end{aligned}$$

This is much too high. That is because our method assumes

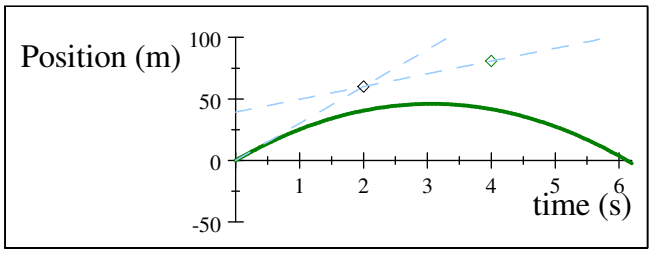
$$\frac{dx}{dt} = v_o$$

for this entire time step. But observing the solid curve shows us that dx/dt is not constant along the path. The slope changes because there is a negative acceleration. So we have over-estimated x_1. This error will compound as we go. For the next Δt we start with x_1 and our new slope v_1 and try to find x_2.

$$\begin{aligned} x_2 &= x_1 + v_1 \Delta t \\ v_2 &= v_1 - g\Delta t \end{aligned}$$

or, given our initial conditions,

$$\begin{aligned} x_2 &= 60\,\mathrm{m} + 10.38\frac{\mathrm{m}}{\mathrm{s}} \times 2\,\mathrm{s} = 80.76\,\mathrm{m} \\ v_2 &= 10.38\frac{\mathrm{m}}{\mathrm{s}} - 9.81\frac{\mathrm{m}}{\mathrm{s}^2} \times 2\,\mathrm{s} = -9.24\frac{\mathrm{m}}{\mathrm{s}} \end{aligned}$$



We can see that if Δt is too big, we take a step that is too high because our slope is too big. This continually happens at each step. Our solution gets poorer and poorer.
We can see that using numerical techniques requires some caution. The farther from your initial conditions you get, the poorer the result will be. In other words, if I look at $t = 0.5$ seconds, the soliton is not too bad. But at $t = 4$ seconds our solution was $39.163\,\mathrm{m}$ off.
You might be tempted to ask "then why do this?" The answer is, we often do not have easy kinematic equations that will solve our problems. Suppose, for example, the acceleration is not constant.

Euler and Simple Systems with Not-So-Simple Dynamics.

Suppose we throw a ball, but we have air resistance. Your PH121 class only has you throw balls in a vacuum–something that is only fun if you have a space suit. Real balls have air resistance. To model air resistance is really a PH123 problem. So I will just give you a formula for the force due to air resistance here. There is a resistive force

$$F_R = \frac{1}{2} D \rho A v^2$$

that depends on the cross-sectional area of the ball, A, the density of the air, ρ, the speed of the ball, v, and the ball's drag coefficient, D, that contains the effects like surface ruffness and shape of the ball. So we have two forces working on the ball now, the force due to gravity, and the resistive force. If we throw the ball straight up, then we only have forces in one dimension. We can use our Newton's second law method to find the acceleration!

$$\Sigma F_x = ma = -F_g + F_R$$

so

$$a = \frac{-F_g + F_R}{m}$$

or

$$\begin{aligned} a &= \frac{-mg + \frac{1}{2}D\rho A v^2}{m} \\ &= -g + \frac{D\rho A v^2}{2m} \end{aligned}$$

Notice that this acceleration changes when v changes. And we know that v does change as the ball goes up. We can't use the kinematic equations at all with a changing acceleration. But we can use our Euler method.

$$\begin{aligned} x(t + \Delta t) &= x(t) + v(t)\Delta t \\ v(t + \Delta t) &= v(t) + a\,(t)\,\Delta t \end{aligned}$$

or

$$\begin{aligned} x_{n+1} &= x_n + v_n \Delta t \\ v_{n+1} &= v_n + a_n \Delta t \end{aligned}$$

The only difference is that now our acceleration is not $-g$, but $a = -g + \frac{D\rho A v^2}{2m}$. So instead of

$$\begin{aligned} x_{n+1} &= x_n + v_n \Delta t \\ v_{n+1} &= v_n - g\Delta t \end{aligned}$$

we will have

$$\begin{aligned} x_{n+1} &= x_n + v_n \Delta t \\ v_{n+1} &= v_n + \left(-g + \frac{D\rho A\,(v(t))^2}{2m} \right) \Delta t \end{aligned}$$

The rest of the method stays the same. In fact, to change physical systems we usually only need to change the acceleration function. This implies that we could build a general Euler solver, and then just modify the acceleration part. That is such a good idea that there are standard notations for this.

Standard Euler Notation

This is not really needed for our lab today, but might be helpful if you have to write a Euler code in the future. It is convenient if you are a computer programer (which you now are!) to write this method more generally, in particular, if we have two coupled

differential equations of the form[5]

$$\begin{aligned}\frac{dx}{dt} &= v(t) = f(x,v,t)\\ \frac{dv}{dt} &= a(t) = g(x,v,t)\end{aligned}$$

All we have done is to write the speed as $f(x,v,t)$ and the acceleration as $g(x,v,t)$. Then we step them forward in time according to

$$\begin{aligned}x_{n+1} &= x_n + f(x_n,v_n,t_n)\Delta t\\ v_{n+1} &= v_n + g(x_n,v_n,t_n)\Delta t\end{aligned}$$

where the subscript n refers to the number of time steps we have taken, and t_n refers to the time of the n^{th} time step. This is really nothing new for us. We just wrote a and v a different way.

$$\begin{aligned}f(x,v,t) &= v(t)\\ g(x,v,t) &= a(t)\end{aligned}$$

so we have only invented a more difficult way to write our equations. We are right back to

$$\begin{aligned}x_{n+1} &= x_n + v_n\Delta t\\ v_{n+1} &= v_n + a_n\Delta t\end{aligned}$$

But if I have more than two coupled equations (say, we let the ball move in two dimensions), the extension is quite straightforward to incorporate the additional equations in the new notation. If

$$\begin{aligned}\frac{dx}{dt} &= f_x(x,y,v_x,v_y,t)\\ \frac{dy}{dt} &= f_y(x,y,v_x,v_y,t)\\ \frac{dv_x}{dt} &= g_x(x,y,v_x,v_y,t)\\ \frac{dv_y}{dt} &= g_y(x,y,v_x,v_y,t)\end{aligned}$$

then the Euler method notation would dictate

$$\begin{aligned}x_{n+1} &= x_n + f_x(x_n,y_n,v_{x_n},v_{y_n},t_n)(\Delta t)\\ y_{n+1} &= y_n + f_y(x_n,y_n,v_{x_n},v_{y_n},t_n)(\Delta t)\\ v_{x_{n+1}} &= v_{x_{n+1}} + g_x(x_n,y_n,v_{x_n},v_{y_n},t_n)(\Delta t)\\ v_{y_{n+1}} &= v_{y_{n+1}} + g_y(x_n,y_n,v_{x_n},v_{y_n},t_n)(\Delta t)\end{aligned}$$

[5] In computer programming books and Numerical Methods Books, they usually switch the v to the letter y to be more general. That is because you may want to solve a differential equation that is not in terms of position vs time. But for this first view of Euler's method, I will stick with v.

Let's see what our ball with air resistance would look like in this notation.

$$\begin{aligned} x_{n+1} &= x_n + f(x_n, y_n, v_{x_n}, v_{y_n}, t_n)(\Delta t) \\ v_{n+1} &= v_n + g(x_n, y_n, v_{x_n}, v_{y_n}, t_n)(\Delta t) \end{aligned}$$

with

$$\begin{aligned} f(x_n, y_n, v_{x_n}, v_{y_n}, t_n) &= v\,(t_n) \\ g(x_n, y_n, v_{x_n}, v_{y_n}, t_n) &= -g + \frac{D\rho A v_n(t_n)^2}{2m} \end{aligned}$$

We really just write out the acceleration equation on a different line, and that makes the code easier to write.

In the event that your time steps are infinitely small (i.e. $\Delta t \to 0$), the Euler method will give you an exact solution. In reality, we cannot take infinitely small time steps. We can imagine making them really, really small, but then we have to take many, many calculations. For example, if my time step were as small as $1\,\mathrm{ns}$, I would have to take one billion steps forward in time just to cover one second of experimental time!

The issue of taking large number of steps is not the only problem we face. Only infinitely small time steps will give you an exact solution. Any time step that is not infinitely small will only give you an approximate solution. The larger the time step, the less accurate the approximation. It turns out that these approximate solutions turn sour quickly, as we will see when we tackle mass-spring systems. Even with a very small time step, an Euler's method simulation of the mass-spring system becomes unphysical after just one or two oscillations!

So we should ask, is this really useful? Is there some way to make numerical modeling more accurate? The answer, of course, is yes to both questions.

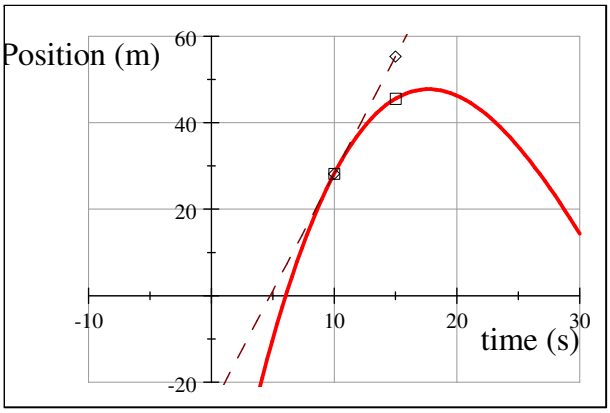
Beyond Euler: The Runge-Kutta Method.

We won't go beyond the Euler method in this class, unless you want to try for the extra credit. But it is useful to think of how we would. So read this next part, but look fop the concepts of how to make Euler better, and don't worry about the details of writing computer code to make the improvement happen.

The nastiness that evolves from the Euler method boils down to one simple fact: we assume that the rate of change of the variable is constant over a given time step. Obviously, this assumption is wrong in practically every circumstance[6]. To make matters even worse, we take our "constant" rate of change from the endpoint of the time interval, and the rate of change at the endpoint is by no means representative of the

[6] The only exception would be, for example, where velocity was constant. In this case a numerical solution would be completely unwarranted.

entire interval. Here is an example. In the figure below, the solid curve is the function we are trying to find. We will start at $t = 10\,\mathrm{s}$.

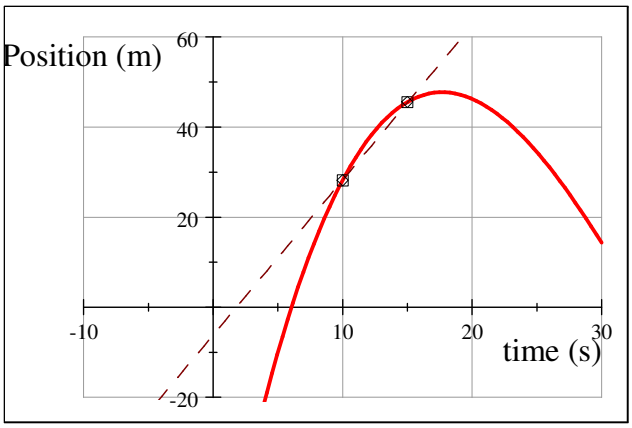


I want to use a time step of $\Delta t = 5\,\mathrm{s}$ (way to large). We see that our Euler's method starts with our initial time at $t = 10\,\mathrm{s}$ and makes a step

$$x_1 = x_0 + v_0 \Delta t$$

that is on the straight line. The slope of the straight line is given by $v_{0.} = \frac{dx}{dt}$ right at our starting point, $(x_o, 10\,\mathrm{s})$. We get the position x_1 marked by the small diamond. But, as we expect, this is not on the true curve at all!

An obvious improvement would be made if we could somehow estimate what the average value of the slope (derivative) over the time interval would be. It would logically be better to use this average value while progressing a given variable over time (for example, using the average velocity over a time interval while evaluating the position).



This is quite an improvement as you can see.

There are methods for doing just this. One way is to take a trial step across the entire interval, estimate the values of the derivatives at the midpoint of the interval, and then

use the midpoint derivatives to step over the entire interval. For the set of coupled differential equations from before,

$$\begin{aligned}\frac{dx}{dt} &= f(x,v,t)\\ \frac{dv}{dt} &= g(x,v,t)\end{aligned}$$

this process is re□ected by numerical iterations of the form

$$\begin{aligned}k_{x1} &= (\Delta t) f(x_n, v_n, t_n)\\ k_{y1} &= (\Delta t) g(x_n, v_n, t_n)\\ k_{x2} &= (\Delta t) f\left(x_n + \frac{k_{x1}}{2}, v_n + \frac{k_{v1}}{2}, t_n + \frac{\Delta t}{2}\right)\\ k_{y2} &= (\Delta t) g\left(x_n + \frac{k_{x1}}{2}, v_n + \frac{k_{v1}}{2}, t_n + \frac{\Delta t}{2}\right)\\ x_{n+1} &= x_n + k_{x2}\\ v_{n+1} &= y_n + k_{v2}\end{aligned}$$

Lets take this apart to see what it means

The last two equations project our position and another variable (for us velocity) forward to the next position.

$$\begin{aligned}x_{n+1} &= x_n + k_{x2}\\ v_{n+1} &= v_n + k_{v2}\end{aligned}$$

If we were using Euler's method we would know what these look like from our previous work:

$$\begin{aligned}x_{n+1} &= x_n + v_n \Delta t\\ v_{n+1} &= v_n + a_n \Delta t\end{aligned}$$

We could identify

$$\begin{aligned}k_{x2} &= v_n \Delta t\\ k_{v2} &= a_n \Delta t\end{aligned}$$

or we could write these as

$$\begin{aligned}k_{x2} &= (\Delta t) f(x_n, v_n, t_n)\\ k_{v2} &= (\Delta t) g(x_n, v_n, t_n)\end{aligned}$$

because

$$\begin{aligned}f(x_n, v_n, t_n) &= v_n\\ g(x_n, v_n, t_n) &= a_n = -g\end{aligned}$$

This would just be a more obscure way to write Euler's method. We would use the derivative or slope at x_1 to find x_2. But we want to find an average slope over the

interval x_1 to x_2. To do this we start with the Euler steps

$$\begin{aligned} k_{x1} &= (\Delta t) f(x_n, v_n, t_n) \\ k_{v1} &= (\Delta t) g(x_n, v_n, t_n) \end{aligned}$$

(note the subscript of the $k's$ changed) and use it to find the average slope. This is done in the middle equations from our set

$$\begin{aligned} k_{x2} &= (\Delta t) f\left(x_n + \frac{k_{x1}}{2}, v_n + \frac{k_{v1}}{2}, t_n + \frac{\Delta t}{2}\right) \\ k_{v2} &= (\Delta t) g\left(x_n + \frac{k_{x1}}{2}, v_n + \frac{k_{v1}}{2}, t_n + \frac{\Delta t}{2}\right) \end{aligned}$$

what this means is that to find the steps k_{x2} and k_{v2} we find the slope function $f(x_n, v_n, t_n)$ but at the points $x'_n = x_n + \frac{k_{x1}}{2}$, $v'_n = x_n + \frac{k_{v1}}{2}$. Since k_{x1} is the distance we would go in the x direction if we used the Euler method, $k_{x1}/2$ is half that distance. We are calculating the slope half way along the Euler step! This is not the true average slope, but it is close to it. Of course we do the same thing for v, calculating the slope (this time our acceleration) half way between v_n and v_{n+1} that we find from the Euler's method.

Once we have the derivatives (slopes), we can multiply by Δt to get our new step size. Using

$$\begin{aligned} x_{n+1} &= x_n + k_{x2} \\ v_{n+1} &= v_n + k_{v2} \end{aligned}$$

we project to the next x_{n+1} and v_{n+1} and start over to find the next set.

The extension to systems of more than two coupled equations, just as with the Euler method, is straightforward. This method is referred to as the ***midpoint method*** or ***second order Runge-Kutta method***.

It is fair to point out that this method is still □awed. We're still getting approximate values for the midpoint derivatives based on some sort of extrapolation from the conditions at the beginning of the interval. However, this method is a vast improvement over the Euler method. The solution is far more stable, even for larger step sizes, and thus more accurately approximates the exact solution.

We don't need to stop here, however. A person could consider using the midpoints of the half-intervals to further refine the numerical process. In so doing, one generally estimates the value of the derivatives at four points and then takes a weighted average of those derivatives. The resulting process is called the ***fourth-order Runge-Kutta***

method. Iterations of the form

$$\begin{aligned}
k_{x1} &= (\Delta t) f(x_n, v_n, t_n) \\
k_{y1} &= (\Delta t) g(x_n, v_n, t_n) \\
k_{x2} &= (\Delta t) f\left(x_n + \frac{k_{x1}}{2}, y_n + \frac{k_{v1}}{2}, t_n + \frac{\Delta t}{2}\right) \\
k_{v2} &= (\Delta t) g\left(x_n + \frac{k_{x1}}{2}, y_n + \frac{k_{v1}}{2}, t_n + \frac{\Delta t}{2}\right) \\
k_{x3} &= (\Delta t) f\left(x_n + \frac{k_{x2}}{2}, y_n + \frac{k_{v2}}{2}, t_n + \frac{\Delta t}{2}\right) \\
k_{v3} &= (\Delta t) g\left(x_n + \frac{k_{x2}}{2}, y_n + \frac{k_{v2}}{2}, t_n + \frac{\Delta t}{2}\right) \\
k_{x4} &= (\Delta t) f(x_n + k_{x3}, v_n + k_{v3}, t_n + \Delta t) \\
k_{v4} &= (\Delta t) g(x_n + k_{x3}, v_n + k_{v3}, t_n + \Delta t) \\
x_{n+1} &= x_n + \frac{k_{x1}}{6} + \frac{k_{x2}}{3} + \frac{k_{x3}}{3} + \frac{k_{x4}}{6} \\
v_{n+1} &= v_n + \frac{k_{v1}}{6} + \frac{k_{v2}}{3} + \frac{k_{v3}}{3} + \frac{k_{v4}}{6}
\end{aligned}$$

In many physical situations, a fourth order Runge-Kutta solution is indistinguishable from the exact solution, even when "relatively" large steps are used. I emphasize the word "many," though, because there will always be exceptions.

There are even further refinements that can be made to these methods. For example, you can use a combination of second and fourth order Runge-Kutta steps and from those extrapolate an n^{th} order Runge-Kutta step. You can imagine ways in which you would use the endpoint of the interval, as opposed to the beginning point, as the basis for estimating your derivatives. You can implement numerical strategies that will adjust the step size so that it is really small in places where the derivatives are large and really large in places where the derivatives are small. You might even try combining all of those methods together. You will probably be grateful to know that we won't do any of these things in our lab. That would take all semester to work out (and then some). But you should be aware that there are ways to increase the accuracy of your numerical solutions.

Implementation

You are probably wondering how you would actually make a computer do all of this calculation. In this section let's consider how to write an Euler method code for our lab, then I will comment on versions to use in further studies (later in your career). You don't need to read this section before class. What follows is a tutorial that will guide you step-by-step through the first part of the lab. But if you want to, read on just to get a feel for what we will do.

First let me give you a more proper introduction to the computer language we use in our class. It is called MatLab[7] and it is a commercial product[8]. One of the objectives of this course is to introduce you to ways to do calculations with high level computer programing languages. Our department has chosen MatLab as our language because it is widely used in industry and academia.

So far we have made plots in MatLab, and we have done some simple calculations, but truthfully, they might have been easier to do on your calculator or in a spreadsheet program. Hopefully, you will see that this week's assignment is not easier in a spreadsheet or on your calculator. That is why we use MatLab or a similar computer language to do calculations. Let's take some time and get acquainted with MatLab.

MatLab–turn your computer into a giant calculator

We can think of MatLab as a tool that turns your computer into a very powerful calculator. Like many powerful things, it takes some time to learn to use it. There are no specific buttons for functions like on your calculator. In the last few labs we have learned how to start MatLab. Do so now. You used MatLab's statistical commands last lab. We found that we had to type commands to get the functions we want. Let's try a new function now. Type

```
sin(90)
```

in the command window. You should see the result

```
ans=0.89400
```

[7] If you are a computer science major, you may prefer another language to MatLab. That is fine, get approval before you start so we can make sure your entire group can work together.

[8] However there is a free programing language called *Octave* that you can use if you want to run your code on your computer, but don't want to pay for MatLab.

We have taken the sine function of the number 90. On our calculator, we would type 90, then press the `sin` button to get the same result.
You may wonder about this answer. Usually $\sin(90\,^\circ) = 1$. Why did MatLab give us 0.89400? MatLab, like most powerful mathematical languages, assumes we are using radians. Try typing

```
sin(pi/2)
```

you should see

```
ans=1
```

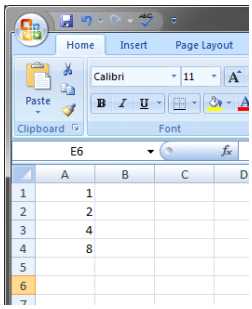
What we mean by `sin(pi/2)` is of course

$$\sin\left(\frac{\pi}{2}\right)$$

MatLab has some common constants predefined, like `pi`$=\pi$.
This may all seem nice, but not as convenient as the calculator that you have used for many years. But, as we said before, MatLab can do some things your calculator cannot do, and that is why we will begin to learn to use it this semester. Think of it as using a large programmable calculator. The MatLab language is the calculator programming language for our lab computers.

MatLab implementation

We will write our lab implementation of numerical integration in MatLab. Let's tackle the Euler method first, then you can modify your Euler code to produce the second order Runge-Kutta version if you are going to attempt the extra credit (Euler is hard enough if you are new at programming).
There are some new ideas in programing that might make this easier. The first is the idea of an *array*. This is a column of data, like you might see on a spreadsheet. In last lab we used an array variable that we named "data" to calculate means and standard deviations. We will use the notation `x(i)` to mean a column of x values. For example, the data in the following figure



could be named `x(i)` where the `i` part is the "index" telling you which element of the

array you will get. The `i` part is like the row number in a spread sheet. In the spread sheet, if I asked for the third row element of our first column, you would tell me the value was 4. Likewise, if we put this data into an array, then `x(3)` = 4. You could mentally think of `x(i)` as

$$\texttt{x(i)} = \begin{array}{|c|}\hline 1 \\ \hline 2 \\ \hline 4 \\ \hline 8 \\ \hline \end{array}$$

We also need a way to do calculations over and over again. MatLab has a way to do the same thing many times. It is called a "for" loop and we have seen it before. Let's take an example from life of something that we do over and over again and write it in MatLab terms so we can see how it works. We go to class for 13 weeks. We meet every week for lab either on Tuesday or Thursday. Let's pick Thursday in our example. We want to go *for* 13 weeks. So we write

```
for week=1:13
    go to class on Thursday
end;
```

This says to go to class on Thursday for 13 weeks, then after the 13th week we end (quit). Everything between the `for` and the `end` is done 13 times.
We can get fancier. We could include doing the reading and even celebrate when lab is over:

```
for week=1:14
    read the reading assignment
    go to class on Thursday
    do the lab assignment
    shout "hurray week ", week, "is done!"
end;
```

This set of instructions will do all the things between the `for` and the `end`, this time four separate things, fourteen times. Notice in the last instruction we shout "hurray," but in the statement the word `week` appears on its own with no quotation marks. This word, `week`, is also used in the `for` statement. This is a variable called the loop counter. It changes value by one every time the instructions that are in the loop are run. The first time the loop instructions run the variable `week` has 1 stored in it. The second time it will have 2 stored in it. The next time 3 and the next time 4 up to 13 for the last time the instructions are run. The idea of a loop counter is very useful. If we want to

work on the data stored in an array, the loop counter can be used as the array index to access the data one "row" at a time.

Our computers can't really execute instructions like "go to class," so let's try something that a computer can do. Let's take something from a previous lab as an example. To make an average, we add up all the numbers in a set, and then divide by the how many numbers we have. Let's see how to make MatLab do this:

```
data = [34 38 33 38 38 36 35 47 36 32 40 40 45 36 ...
        43 38 48 40 40 38 43 40 39 36 46 34 37 33 32 34 ];
N=30;
sx=0;
for n=1:N
    sx=sx+data(n);
end;
ave=sx/N
```

This set of instructions starts by defining an array of numbers called `data`. Next, it defines the variable N and sets it equal to 30. This is the number of data points in the array `data`. Next a variable `sx` is defined and set equal to zero. This will be a memory place to put the sum as we make the sum in the loop.

Next we have a `for` statement. The loop counter is the variable n. The loop will run 30 times, because $N = 30$. Inside the loop we just have one statement, `sx=sx+data(n);`. You may be saying, "wait, that last equation does not make sense" and you would be right. It is not an equation but is a MatLab *assignment* statement. It means "take the current value of variable, `sx`, and add the value of `data(n)` to it , and place the answer back in the variable, `sx`." Here we are using the loop counter to choose the "row" in our array to use. Each time we run the statement `sx=sx+data(n);` we want to use a different "row" in the array. This is like using a different row in the column in our spreadsheet. We need to add up all the numbers in `data` so we need to have a different "row" from the array each time. The loop counter lets us do that. After N loops, we have added up the whole array and placed the result in `sx`. To finish our average, we need to divide by N. The last statement does this, and assigns the result toe the variable `ave`.

Notice that we started by setting the variable `sx` to zero. We may want to do that for an entire array. A way to do that is the `zeros()` command.

```
t=zeros(1,N+1);
```

This command makes an array called `t` that is `N` elements long and sets every element of `t` to zero.
For our Euler code, you will want to define a variable for Δt, say, `deltat`. You will want to be able to change this so you can see the effect of making Δt smaller. Since the total number of calculations for a given total time and Δt are related, you might choose to calculate N from Δt,

```
deltat=0.01; %in seconds
ti=0;        %initial time
tf=20;       %final time for our calculation
N = (tf-ti)/deltat;
```

Note that I am writing useful, informative things after a `%` sign. The `%` sign tells MatLab to ignore anything after it. It allows us to make useful comments that we can understand, but MatLab can't. In this case, I included units (seconds) and told you what data I would store in the variables `tf` and `ti.`
Another useful MatLab function is the array filler:

```
t = (0:deltat:tf)
```

This command fills the array, `t` starting with `t(1)=0`, and adding `deltat` each time so you have `t(2)=deltat` and `t(3)=2*deltat` and `t(4)=3*deltat` and so forth up to `t(N)=(N-1)*deltat` which should be `tf`. For example,

```
t=(0:0.5:2)

t =
 0 0.5000 1.0000 1.5000 2.0000
```

This is very like the spreadsheet ability to start a calculation and copy it down a column to make it calculate many rows the same way.
Yet another help is to realize that the condensed notation we have used

$$\begin{aligned} a &= -g \\ x_{n+1} &= x_n + v_n \Delta t \\ v_{n+1} &= v_n + a\Delta t \end{aligned}$$

is very like, the assignment notation in a loop. We could write these as

```
a=-g;
x(n+1)=x(n)+v(n)*deltat;
```

```
v(n+1)=v(n)+a*deltat;
```

where `x(n)` and `v(n)` are arrays. MatLab will understand these equations! These statements would be included inside a loop, of course, so we could increase the index `n` so we could calculate the `x(n)` and `v(n)` for every time step, Δt. Alternately, we could use our fancier notation

$$\begin{aligned} f(x_n, v_n, t_n) &= v_n \\ g(x_n, v_n, t_n) &= -g \\ x_{n+1} &= x_n + (\Delta t) f(x_n, v_n, t_n) \\ v_{n+1} &= v_n + (\Delta t) g(x_n, v_n, t_n) \end{aligned}$$

which requires four lines of assignments

```
f=v(n);
g=-9.8004; %m/s^2
x(n+1)=x(i)+deltat*f;
v(n+1)=v(n)+deltat*g;
```

which might be better. The point is, that our condensed notation is very like MatLab code and is a guide in how to write our MatLab statements.

Step by step instructions

Armed now with a few MatLab commands, we can make the Euler code in MatLab.
Start by opening a new MatLab script file.
We want to be sure we know what the instructions will do later, so we should start by making some comments. I suggest something like this

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% One dimensional free-fall Euler code                                    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% PH150
% Brother Lines (<-your names here)
% 2012 August 14
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  this code will calculate the exact solution for ball in free fall
%  having been shot straight up using Euler's method
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Since all of these lines start with % MatLab will ignore all of these lines. But they will help us a lot to remember what we have done. Next, let's define some of the variables we need so they will be all in one place. This makes it easier to change the numbers we put in these variables later when we compare our numerical prediction to an actual experiment.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initial conditions and physical setup constants
v0=30.0; %initial velocity (m/s)
x0=70;    %initial position

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Set up the time steps and number of calculations
deltat=0.01;  %Time step in seconds
to=0;          %Start up time in seconds t=0
tf=7.0;       %Final time in seconds
```

The first two statements set up the initial speed of the ball and it's initial height (given the variable x in this code, like in our reading, but it is really the y-direction). The next three statements define Δt, and the initial time and final time to consider in our calculation. Notice that there are other lines that start with %. These are more comments.
Next, let's calculate how many times we need the loop to run to make our Euler solution. A little thought tells us that the total time the ball will be in the air is $t_f - t_o$. The number of time steps is then $(t_f - t_o)/\Delta t$. Here it is in MatLab:

```
% calculate how many time steps to take
N=(tf-to)/deltat;
```

We need some arrays as places to put our Euler solution as we build it. This is like making columns in a spreadsheet. MatLab does not have columns already built like as spreadsheet does. We need to make them. We will use the array zero fill command.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% define arrays and fill with zeros
t=zeros(1,N+1);  % time array
x=zeros(1,N+1);  % x positions (vertical position)
v=zeros(1,N+1);  % velocity array
```

Now we have an array of times, it would be good to put in time values. We can do this using the array fill command:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% fill the time array with times that are deltat apart
t=(0:deltat:tf);
```

Now we are ready to build the loop that performs the Euler calculation.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%now perform an Euler's Method Calculation
v(1)=v0;  % initial velocity
x(1)=x0;  % initial position
for n=1:N
    f=v(n);
    g=-9.8004;
    x(n+1)=x(n)+deltat*f;
    v(n+1)=v(n)+deltat*g;
end;
```

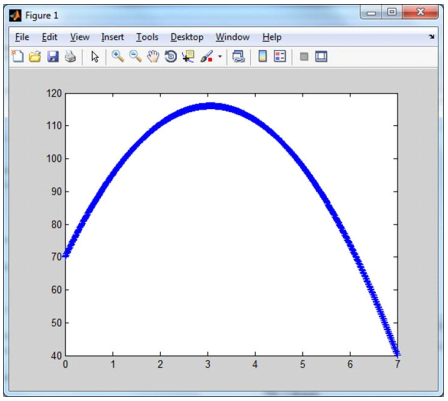
The first two lines put our initial position and initial speed into the first part of the array. This is like putting the initial conditions in the first row of the position and speed columns in our spreadsheet. The loop counter is n. Then we have MatLab versions of our Euler equations

$$\begin{aligned} f(x_n, v_n, t_n) &= v_n \\ g(x_n, v_n, t_n) &= -g \\ x_{n+1} &= x_n + (\Delta t) f(x_n, v_n, t_n) \\ v_{n+1} &= v_n + (\Delta t) g(x_n, v_n, t_n) \end{aligned}$$

At the end of the loop we have a position and speed for each time in our t array. To see how our ball traveled, we can plot position vs. time.

```
% plot the results
plot(t,x,'+');
```

The plot should look something like this.



And the completed code might look like this:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% One dimensional free-fall Euler code %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% PH150
% Brother Lines
% 2012 August 14
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% this code will calculate the exact solution for ball in free fall
% having been shot straight up using Euler's method
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initial conditions and physical setup constants
v0=30.0; %initial velocity (m/s)
x0=70; %initial position
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Set up the time steps and number of calculations
deltat=0.01; %Time step in seconds
to=0; %Start up time t=0
tf=7.0; %Final time
% calculate how many time steps are in 20 seconds
N=(tf-to)/deltat;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% define arrays and fill with zeros
t=zeros(1,N+1); % time array
x=zeros(1,N+1); % x positions (vertical position)
v=zeros(1,N+1); % velocity array
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% fill the time array with times that are deltat apart
t=(0:deltat:tf);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%now perform an Euler's Method Calculation
v(1)=v0; % initial velocity
x(1)=x0; % initial position
for i=1:N
 f=v(i);
 g=-9.8004;
 x(i+1)=x(i)+deltat*f;
 v(i+1)=v(i)+deltat*g;
end;
% plot the results
plot(t,x,'+');
```

Implementation in your future work

Numerical schemes commonly employed by professional scientists are often more complicated out of necessity. How those methods work, however, is beyond the scope of our present discussion. For more information, I would direct your attention to chapter 16 of the book *Numerical Recipes in C*, written by William H. Press, et al., and published by Cambridge Press.[1] Although the algorithms in the text are written in the C programming languages (there are also versions available in C++ and ForTran, I use the ForTran edition), the text very clearly describes these numerical processes in a way that is essentially independent of the programming language. There are several sources of pre-written code that you can include in your programs to perform numerical calculations. *Numerical Recipes in C* includes many. The Gnu Science Library (GSL) also contains many good numerical routines. But you should be careful to only use such routines when you know exactly how they work. Otherwise, you may have a terrible surprise at some point when the routine fails because you have used it outside it's valid range.

Assignment: Numerical modeling of Projectile motion

Because we are learning a major new skill. we will take three weeks to complete the experiment we are starting today. This will affect your lab notebook. Complete each part of the lab each week in an organized fashion in your lab notebook. Make graceful end-of-class entries so you can start up again the following week. As always, part of the grade will be based on neatness and organization!

1. Create a MatLab script that will numerically model the motion of a spherical projectile shot straight up and then falling back down using Euler's method. Assume there is no air resistance. As input quantities you should provide the following:
   - The initial y position of the projectile $y = 70.00$ meters
   - The initial speed of the projectile. $30.0\,\mathrm{m/s}$
   - The time step size.0.1 seconds
   - The acceleration due to gravity $9.8\,\mathrm{m/s^2}$

   Make these quantities variables so you can easily change values and recalculate. There is a step-by-step tutorial in today's reading that will walk you through this part. Save your script and record where you saved it. Include a scatter plot of y vs t in your lab notebook.

2. Copy and then modify your MatLab script to numerically model the motion of a spherical projectile being launched from a cannon using Euler's method. As input quantities you should add the following (Make these quantities variables so you can easily change values and recalculate):
   - The initial x position of the projectile $x = 0.00$
   - The launch angle for the projectile (measured from the positive x axis) 45 degrees Include a plot of y vs x in your lab notebook. Save your script and record where you saved it.

3. Copy and modify your script to include air resistance. Air resistance will add a new resistive force that is proportional to the square of the projectile's velocity, i.e.
   $$F_R = \frac{1}{2} D\rho A v^2$$
   where D is the drag coefficient, ρ is the density of the air, A is the cross-sectional area of the projectile presented to the air (in our case $A = \pi r^2$), and v is the speed. The force is directed opposite the velocity of the projectile. You will need components of this force. You should convince yourself that
   $$\begin{aligned} F_{R_x} &= -\frac{1}{2} D\rho A v\left(n\right) v_x\left(n\right) \\ F_{R_y} &= -\frac{1}{2} D\rho A v\left(n\right) v_y\left(n\right) \end{aligned}$$
   where
   $$v\left(n\right) = \sqrt{\left(v_x\left(n\right)\right)^2 + \left(v_y\left(n\right)\right)^2}$$
   and $v_x\left(n\right)$ and $v_y\left(n\right)$ are the components of the velocity. This form of the components of the resistive force avoids having to calculate the angle at each of our Euler steps. As input quantities you should provide the following (Make these quantities variables so you can

easily change values and recalculate):

- The mass of the projectile.$0.020\,\mathrm{kg}$
- The radius of the projectile.0.02 meters
- The drag coefficient for the projectile 0.2
- The density of the air the projectile travels through $1.23\,\mathrm{kg}/\,\mathrm{m}^3$

4. Include a scatter plot of your x and y positions for each time step in your lab notebook

If you also do this using the second order Runge-Kutta will receive extra credit–but this is not required. Groups that use fourth order Runge-Kutta will receive extra extra credit. In order to make sure all groups get through with the Euler's method solution, I will try to limit my help just to the Euler method.

8 Numerical modeling of a Mass-Spring System

In our last lab we started using Euler's method to solve physics problems. We started with a problem we could do algebraically (a ball toss with no air resistance), but then considered a problem that we could not do with just algebra (a ball toss with air resistance). We did this by using the coupled equations

$$\begin{aligned} x(t+\Delta t) &= x(t) + v(t)\Delta t \\ v(t+\Delta t) &= v(t) + a\,(t)\,\Delta t \end{aligned}$$

where for the first case

$$a\,(t) = -g$$

and for the second

$$a\,(t) = -g + \frac{D\rho A v(t)^2}{2m}$$

We said that to take on different problems using Euler's method, we only have to change the acceleration term. And we can see that this was true for our moving ball cases. We are going to take on a very different physical system today, and we will see that even though it is very different, we can still use Euler's method to describe the motion, and all we have to do is to change the acceleration term again.

Mass Spring Systems

Let's review Hook's law for springs. Consider a system consisting of a mass attached to a spring lying on a frictionless surface, as depicted in next figure.[9]

[9] A mass-spring system is often called a harmonic oscillator, it will show up under this name in your PH121 book.

We assume that the spring strictly obeys Hook's law, $F = -kx$, where k is the spring constant and x is the displacement from the equilibrium position and F is the spring force. There are only four inputs to this system: the spring constant, the mass, and the initial velocity and initial displacement of the mass. We turn to Newton's second law again to find the acceleration. There is only one horizontal force.

$$\Sigma F_x = ma = -kx$$

so

$$a = -\frac{k}{m}x$$

so we can write a set of coupled equations for the mass-spring system

$$\begin{aligned} x(t+\Delta t) &= x(t) + v(t)\Delta t \\ v(t+\Delta t) &= v(t) + -\frac{k}{m}x(t)\Delta t \end{aligned}$$

This is our Euler's method solution. We can use the same code as we started with last lab for our one dimensional ball motion, but change the acceleration to $a = -\frac{k}{m}x$.

The exact mass-spring solution

Of course, if you have taken PH123 you know we can also find an exact solution for the motion of a mass-spring system's motion. Let's introduce or review this here. It has quite a bit of calculus, so if you are concurrently taking M12, don't worry. all you need is the answer at the end.

Given the simplicity of this system, it is fairly easy to write down an equation that describes the motion. If your calculus skills are up to it, we can rewrite this using differentials. Recognizing that acceleration is the second time derivative of position,

$$a = \frac{dv}{dt} = \frac{d}{dt}\left(\frac{dx}{dt}\right) = \frac{d^2x}{dt^2}$$

we can rewrite our spring acceleration as

$$\frac{d^2x}{dt^2} = -\frac{k}{m}x$$

We call this type of equation a *differential equation*. In our case, we have an equation that involves a quantity, x, along with it's second derivative. Because we have the second derivative of x, it is called a second order ordinary differential equation. If you stare at this equation long enough, the solution becomes somewhat apparent. Ask yourself: for what function can I take a derivative of twice and get the same function back along with a negative sign and a constant? If you are far enough in your M112 class the answer is obvious, a sine or a cosine.[10] Of course it could also be a

[10] But if you are not far along in M112, don't worry. I am just trying to motivate the calculus savvy reader to believe the position as a function of time will be a sine or cosine function.

combination of sines and cosines, so we can write a general solution

$$x(t) = a\sin(\omega t) + b\cos(\omega t)$$

where A and B are constants and either could be zero. If we substitute this in for x in our differential equation, and take a couple of quick derivatives we will see that our solution works so long as

$$\omega = \sqrt{\frac{k}{m}}$$

Note that there are two undetermined constants in this expression, namely a and b. Using a little trig[11], we can rewrite this in an equivalent way[12]

$$x(t) = A\sin\left(\sqrt{\frac{k}{m}}t + \phi\right)$$

with the undetermined constants A, which represents the amplitude or maximum displacement of the oscillation, and ϕ which is referred to as a phase angle. This is true because the sum of a sine and a cosine function is another sinusoidal function (think of all those trig identities you memorized in your trig class!).

If we know what the initial displacement and velocity are, it is easy enough to determine the values of A and ϕ. Doing so at this point would distract from our discussion, so we will forgo the exercise for now. In any event, the solution is oscillatory, as you would expect from a mass on a spring.

Adding friction (damping)

A frictionless mass-spring system is obviously not too realistic. Many oscillating systems will not fit our exact equation too well. Real masses have frictional forces acting on them. A mass spring that experiences friction is said to be *damped*.

To modify our Euler code to use damping, we go back to Newton's second law one more time. We need a new expression for the acceleration. There are two forces that act on the spring: the spring force $F = -kx$ which acts opposite the displacement and a damping force $F = -bv$ that acts opposite the velocity. Here b is the damping coefficient and is constant for a given situation.

The dynamic equation that describes the system is then

$$\Sigma F_x = ma = -kx - bv$$

We get a new acceleration

$$a = \frac{-kx - bv}{m}$$

and use this in our coupled Euler equations.

[11] OK, a lot of trig.

[12] This is where you should start paying attention if you are concurrently taking M112.

Review of the Euler Method

Let's review how to turn this into an Euler solution.

Starting from the basic kinematic equation

$$\Delta x = v_o t + \frac{1}{2} a t^2$$

and assuming that for our time interval the change in velocity (acceleration) is essentially zero, then

$$\Delta x = v_o \Delta t$$

or

$$x_f = x_i + v_o \Delta t$$

remember that we have only let our experiment run for a very small Δt, so even though it is accelerating, and the acceleration varies with x, it would not have changed velocity much. And our equation will nearly work. In the limit as $\Delta t \to 0$ our equation will be exact, just as we said before.

Of course we want go an additional small time interval. That is, let the experiment run from $t = \Delta t$ to $t = t + \Delta t$. Then we can predict that the next position will be

$$x(t + \Delta t) = x(t) + v(t)\Delta t$$

As before we need to estimate a new velocity at the end of our time step, Δt. We again use

$$v = v_o + at$$

assuming a very small time interval (small enough that the acceleration does not change a whole lot over the interval) I can write

$$v(t + \Delta t) = v(t) + a(t)\Delta t$$

where I don't necessarily know that the acceleration is constant in time. But just like in the last equation for position vs. time we did not assume the velocity was constant (we just assumed it was essentially constant over a small Δt) we will assume the acceleration is nearly constant over the interval Δt.

We found (above) that for a frictionless mass-spring system Newton's second law gave us

$$a = -\frac{k}{m} x$$

This acceleration depends on position. It is not constant so technically we can't use the kinematic equations. But it is constant enough over a very small Δt.

Knowing the acceleration, we can use it in our velocity equation.

$$v(t + \Delta t) = v(t) - \frac{k}{m} x(t)\Delta t$$

Now, we have two equations that project the position and speed forward in time, based

on what the position and speed currently are:

$$\begin{aligned} x(t+\Delta t) &= x(t) + v(t)\Delta t \\ v(t+\Delta t) &= v(t) - \frac{k}{m}x(t)\Delta t \end{aligned}$$

Just like before, these expressions are coupled: the new position depends on the current velocity, and the new velocity depends on the current position. To find the solution we will take a small step in time, Δt, and calculate a new position. We will also calculate a new speed. Then we will use the new speed to find the next new position Δt later, and the new position to find another new speed. Once we step these two quantities forward in time, we can repeat the process indefinitely.

In short hand these equations become

$$\begin{aligned} x_{n+1} &= x_n + v_n\Delta t \\ v_{n+1} &= v_n - \frac{k}{m}x_n\Delta t \end{aligned}$$

We have labeled each new x and v by how many time steps, Δt, we have taken. The first Δt will be $n = 1$, the second $n = 2$, and so on. Just to be clear, let's again compare

$$\begin{aligned} x(t+\Delta t) &= x(t) + v(t)\Delta t \\ x_{n+1} &= x_n + v_n\Delta t \end{aligned}$$

This means that we start with the initial velocity, v_0 at the initial position x_0. We use these to calculate the position x_1 and the speed v_1 a time $t = \Delta t$ later. Then we start with the initial velocity, v_1 at the initial position x_1 and use the equations to calculate the position x_2 and the speed v_2 an additional time Δt later ($t = 2\Delta t$). Then we use x_2 and v_2 to find x_3 and v_3 and so forth.

General form for the mass-spring system

We can write our coupled equations in our general form for Euler's method if we want to. Here is how they would look.

Starting with the second order differential equation

$$\frac{d^2x}{dt^2} = -\frac{k}{m}x$$

we write this as two coupled first order differential equations:

$$\begin{aligned} \frac{dx}{dt} &= v \\ \frac{dv}{dt} &= -\frac{k}{m}x = a \end{aligned}$$

To relate this to our general form,

$$\begin{aligned}\frac{dx}{dt} &= v(t) = f(x,v,t)\\ \frac{dv}{dt} &= a(t) = -\frac{k}{m}x\,(t) = g(x,v,t)\end{aligned}$$

We can write these for time t_n as

$$\begin{aligned}f(x_n,v_n,t_n) &= v(t_n)\\ g(x_n,v_n,t_n) &= -\frac{k}{m}x\,(t_n)\end{aligned}$$

or more compactly,

$$\begin{aligned}f(x_n,v_n,t_n) &= v_n\\ g(x_n,v_n,t_n) &= -\frac{k}{m}x_n\end{aligned}$$

which gives us what we need to understand our general form

$$\begin{aligned}x_{n+1} &= x_n + (\Delta t) f(x_n,v_n,t_n)\\ v_{n+1} &= v_n + (\Delta t) g(x_n,v_n,t_n)\end{aligned}$$

Of course this is really just

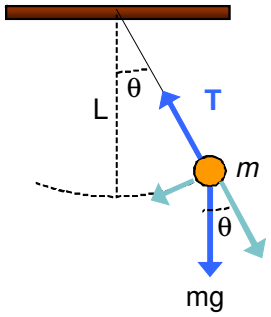
$$\begin{aligned}x_{n+1} &= x_n + (\Delta t) v_n\\ v_{n+1} &= v_n + (\Delta t)\left(-\frac{k}{m}x_n\right)\end{aligned}$$

for our specific case. But sometimes it is easier to write a general routine and then provide $f(x_n,y_n,t_n)$ and $g(x_n,y_n,t_n)$ for our specific case in a subroutine[13]. This way the main routine can be easily used over and over and only the subroutines need to be rewritten if we change specific cases.

Harder oscillation systems

There are problems worse than damping. After all, if you have taken PH 123 you know that we can find a solution even to the damped case that is a single equation (but it is hard to do). If I can solve a differential equation, why would I need to work out a numerical solution? The answer is that not all differential equations have nice, neat analytical solutions like the one that describes the mass-spring system. For example, consider a pendulum bob on a massless string, as shown in next figure. We confine the motion of the bob to the plane of the page, so that the displacement of the bob can be conveniently described by the angle θ that the string makes with the vertical.

[13] The terms "routine" and "subroutine" will be familiar to the computer science majors. For the rest of us they mean "program" and "sub-program" or in MatLab "script" and "function."



A Simple Pendulum

There are two forces acting on the pendulum bob: gravity, which pulls downward, and the tension in the string. The acceleration of the bob can be expressed in terms of a centripetal acceleration (parallel to the string) and a tangential acceleration (perpendicular to the string). There is never a change in velocity in the direction of the string, so the component of gravity parallel with the string must equal the tension. The tangential acceleration of the bob, which is what changes the bob's speed, is thus provided by the component of gravity perpendicular to the string. If you do the geometry, you find that this force has a magnitude of

$$F = -mg\sin\theta$$

Putting this into Newton's second law, we end up with the relation

$$ma_{tan} = -mg\sin\theta$$

The tangential acceleration is related to the angular acceleration by[14]

$$a_{tan} = L\alpha$$

where L is the length of the string, and the angular acceleration (α) is the second time derivative of the angular position, i.e.

$$\alpha = \frac{d^2\theta}{dt^2}$$

Inserting these latter two expressions into our dynamic equation, we get

$$mL\frac{d^2\theta}{dt^2} = -mg\sin\theta$$

At this point, the masses cancel and we can isolate the derivative on the left hand side of the equation:

$$\frac{d^2\theta}{dt^2} = -\frac{g}{L}\sin\theta$$

Despite it's relatively simple appearance, this second order nonlinear ordinary differential equation does *not* have a simple analytic solution[15]. At this point, most

[14] Understanding the details here is not so important as noticing that we are again using Newton's second law to find the acceleration.

[15] The analytic solution involves a few not-so-straightforward operations involving differential calculus, as

introductory physics texts use what is known as a small angle approximation, i.e. for small angles $\sin\theta \approx \theta$, where θ is in radians. Once that approximation is made, the differential equation has exactly the same form as the equation for the mass-spring system, and can be solved analytically. However, we recognize that the solution will only be valid for small angles, generally less than about 15° (it is only really very good for very small angles $< \tilde{}\, 4^\circ$).

But what if you want to know something about the motion of the pendulum for larger angles? You have two choices. Either you can go through the painful process of solving the differential equation and numerically solving the resulting integral (you physics majors will be able to do this by the time you are seniors), or you can numerically approximate the solution to the differential equation in a way that gives us useful information about how the system behaves, even when the angles are large. In other words, you can numerically model the pendulum. It is generally true that, with the introduction of some intermediate variable (in our case v), that any second order differential equation can be broken down into two coupled first order ordinary differential equations. For the pendulum, these first order equations are

$$\begin{aligned} \frac{d\theta}{dt} &= \omega \\ \frac{d\omega}{dt} &= -\frac{g}{L}\sin\theta \end{aligned}$$

This system (or any system like it) can be progressed forward in time using Euler's method just as simply as the mass-spring system. Thus we can solve any second order differential equation using Euler. For the pendulum, we can find the motion without the pesky requirement that the angle be small. Where no simple or easy analytic solution exists, we can still solve the system numerically. This is a fantastic benefit that you physicists and engineers will appreciate more as you take higher level classes!

It is worth noting that there are many more physical systems for which analytic solutions simply cannot be found (not all differential equations have analytic solutions). In those cases, you must solve numerically, there is no other choice![16]

The easiest way to numerically model the dynamics of a system is to ask three questions:

1. What is the current position or state of my system?
2. How fast is it moving, and in what direction?
3. Can I guess where the system will be a short time from now?

This is just what we did in using Euler's method.

well as integrating the differential equation twice. The second integration is an elliptical integral of the first kind which generally can't be done without resorting to a numerical method.

[16] You computer science majors have great job security!

Assignment: Numerical modeling of mass-spring system

Complete this lab as a group and record your experience in an organized fashion in your lab notebook. Part of the grade will be based on neatness and organization!

1. Copy, then modify your one dimensional Euler code and then modify it to numerically model the motion of a mass-spring system with a mass of 200.0 grams, a spring constant of 0.500 N/m, an initial displacement from equilibrium of 15.0 cm, and an initial velocity of zero. Use a time step of 0.01 seconds, and simulate the motion for a total of 20 seconds. Graph your results, and comment.

2. Repeat problem 1, but this time using a damped oscillator. Damping is a resistive force that is opposite the direction the mass is traveling. The resistive force has the form $F_R(n+1) = bv(n)$ where $v(n)$ is the mass speed and b is a constant that tells how much resistance the system has. In your modeling, use $b = 0.05$ kg/s.

3. Investigate what happens with each of the above modeling scenarios when you increase or decrease the time step.

4. If there is time, try different masses, spring constants, damping coefficients, and so on.

5. Extra Credit: Model the same system as in the previous problem, this time using a second order Runge-Kutta algorithm. Graph your results, and comment.

6. Extra Extra Credit: Repeat problem 2, this time using a fourth order Runge-Kutta algorithm.

9 Measurement and Uncertainty III

We should tie our numerical modeling knowledge into our understanding of uncertainty. We want to be able to numerically predict the outcome of an experiment. But that prediction should come with an uncertainty. How can we find an uncertainty for something we found numerically?

Let's think about what we have been calling the high/low method of finding an uncertainty. Each measurement we make has an uncertainty. If we use those measurements to calculate things like we did a few weeks ago to find the value of g, we have to combine or *propagate* the uncertainties. We used large equations with derivatives to do this when we knew the function that described our prediction. But now we don't have an equation that tells us how a ball will fall with air resistance. We need another method for finding the uncertainty in a numerical solution like our Euler method. But we can use our high/low method to develop a new, computational method of finding the error.

Think back to our lab where we measured g. One way to find the uncertainty was to use our equation to find the combination of values that gave the largest estimate for g. Then find the combination of values that gave the smallest estimate for g. For example, when you found g using the pendulum your equation should have been something like

$$g = 4\pi^2 \frac{L}{T^2}$$

we can find the uncertainty by taking

$$g_{\max} = 4\pi^2 \frac{L_{\max}}{T_{\min}^2}$$

where $L_{\max} = L + \delta L$ and $T_{\min} = T - \delta T$. We can also find

$$g_{\min} = 4\pi^2 \frac{L_{\min}}{T_{\max}^2}$$

And then an estimate for the uncertainty would be half the difference between $g_{\max}$ and $g_{\min}$.

Of course we can't exactly do this for our ball case because we don't know the function for our ball fall with air resistance. We don't know whether to use max or min input vales (like $L_{\max}$ and $T_{\min}$ above). Combinations that make the maximum distance and

the minimum distance do exist. We just don't know how to form them. We have lots of inputs to our Euler's method. Each has uncertainty. How do we know which ones to use as maximum values and which ones to use as minimum values? We know that things get funny when we divide, so without knowing the actual function we don't know how to combine the input values to get the maximum or minimum distance.

The answer is that we will have to guess, but guess systematically. That may sound unsettling, but we have a computer that can help us to guess. We can choose random values for each of our inputs such that the value we choose is within the uncertainty for each measurement. Then run the code and see how far away the ball gets when it lands. Then choose a new random set of inputs and see if it get's farther. The computer can do this quickly, so we can run hundreds of combinations in just a few minutes. For our ball-shot-out-of-a-cannon model, say, we used

- The initial y position of the projectile $y = (27.00 + 1)\ \mathrm{cm}$
- The initial speed of the projectile. $(6.8 + 0.2)\ \mathrm{m/s}$
- The launch angle for the projectile (measured from the positive x axis) $(45 + 1)\ ^\circ$
- The mass of the projectile.$(0.0003 + 0.00001)\ \mathrm{kg}$
- The radius of the projectile.$(0.0125 + 0.000)\,1\,\mathrm{m}$
- The drag coefficient for the projectile $0.46 + 0.2$
- The density of the air the projectile travels through $(1.23 + 0.1)\ \mathrm{kg/m^3}$
- the acceleration due to gravity is $(9.8004 + 0.001)\ \mathrm{m/s}$

Then we could try again with the choices

- The initial y position of the projectile $y = (27.00 - 1)\ \mathrm{cm}$
- The initial speed of the projectile. $(6.8 - 0.2)\ \mathrm{m/s}$
- The launch angle for the projectile (measured from the positive x axis) $(45 - 1)\ ^\circ$
- The mass of the projectile.$(0.0003 - 0.00001)\ \mathrm{kg}$
- The radius of the projectile.$(0.0125 - 0.000)\,1\,\mathrm{m}$
- The drag coefficient for the projectile $0.46 - 0.2$
- The density of the air the projectile travels through $(1.23 - 0.1)\ \mathrm{kg/m^3}$
- the acceleration due to gravity is $(9.8004 - 0.001)\ \mathrm{m/s}$

If we do this hundreds of times, each time with a different combination input parameters, we can take a mean and standard deviation and have our predicted value and uncertainty for our distance that the ball will go.

We might not get the actual maximum (or minimum distance) combination without trying millions of combinations (maybe more!). But knowing some statistics, we can still estimate the uncertainty We can average our predicted distances and use this average as our prediction for where the ball will hit. We can take a standard deviation

of our predicted distances and use this for an uncertainty. A standard deviation of the mean might be even better.

This is a little like throwing darts at a target and taking the mean and standard deviation to tell where we were aiming and with what precision. But now we are doing that with our numerical prediction. We make many predictions with our inputs within their respective uncertainties, and then average at the end.

We know that when we use a standard deviation for our uncertainty, we miss some cases, only 68% of the predicted distances will fall within that standard deviation. So it is not as important to run our code millions of times if we can make due with a standard deviation as our estimate of uncertainty. A few hundred combinations should be enough.

Assignment: Modeling and Uncertainty

In this lab we will finally test our numerical modeling of a projectile. We will use our spring cannons and shoot balls that have large drag coefficients. **Be careful not to shoot people, including you–don't look down the barrel of the cannon! Also, don't load the cannon by pushing down on the ball in the cannon, set the spring first and then gently put in the ball If you push on the Styrofoam balls you will have □at balls that don't match our models.**

There is too much to do in this lab to have everyone do all the parts. You will have to split up your group. So how do you record all the experiment if you only do part of it? The way this is done is to write down the results obtained by the other subgroups, and refer to their lab notebooks for the details.

1. **Predict** the maximum range for the path of a Styrofoam ball (or a plastic ball if we don't have any Styrofoam balls) shot out of the spring cannon. Do this with a numerical model of the ball's □ight path written by one (or more) of the members of your group that has been changed so that it can find the error estimate as described in today's lab reading.

2. Determine the uncertainty in your prediction with your new code that chooses values for the input parameters randomly from within the parameter uncertainty ranges. This is actually more efficient than trying to pick values in a non-random way to cover all possible combinations of uncertainties! You will need to take about 100 or more sets of parameters to get a good estimate, but if you built your code so you can easily change your parameters, this will go quickly (if you put actual numbers into your equations in the previous labs, it might be faster to fix your MatLab code first to make this part go more quickly. If you need help, call me over). I will have an example code for you to look at if you get stuck.
   - You will need a maximum and a minimum value for each parameter. Here is an example for the drag coefficient.

     ```
     Dmax=0.48; %kg/s Maximum value of the drag coefficient
     Dmin=0.45; %kg/s Minimum value of the drag coefficient
     ```

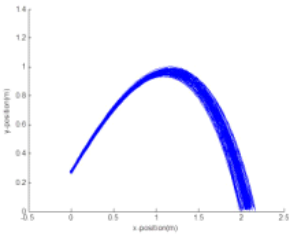
   - Then you will need to pick a value in between the max and the min randomly

     ```
     D=Dmin+(Dmax-Dmin)*rand(1); % kg/s Current value for the drag coefficient
     ```

     the `rand(1)` command gives you a random value between 0 and 1. So

     $$D_{\min} + (D_{\max} - D_{\min}) * \mathrm{rand}\,(1)$$

     gives a random value between $D_{\max}$ and $D_{\min}$.
   - Of course, you want to do this for every variable that has uncertainty, and you want to do the calculation many times so you get a good idea of the precision of your answer. Here is a figure showing an example result



Part of your team should do this work and report back to the rest of the group.

3. You will need to find the drag coefficient. A subteam should find a value (and uncertainty) for the drag coefficient. One way to do this is to use our high speed cameras and film the ball dropping. Since there is significant air resistance, the ball will reach terminal speed. At this point, there is no acceleration, so

$$\begin{aligned} \sum F &= 0 = R - F_g \\ F_g &= R \\ mg &= \frac{1}{2} D \rho A v^2 \end{aligned}$$

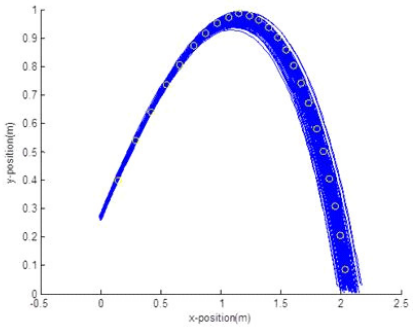
so

$$D = \frac{2mg}{\rho A v^2}$$

Since we know or can measure everything except D, we can solve for D. Use $g = (9.8004 \pm 0.0001)\ \mathrm{m/s}$ and $\rho = (1.23 \pm 0.1)\ \mathrm{kg/m^3}$.

- The rest of the values you will need to measure. Use the digital camera and Logger Pro to estimate the terminal velocity. You will have to drop the ball from high up to get a good value (you have to give it time to reach terminal velocity). Near the ground there can be problems due to the air □ow hitting the ground, so don't use positions that are close to the ground..
- I suggest you have a team of people from your group do this while a second team modify your MatLab code. Remember you need an uncertainty in D.

4. Verify the prediction. Did the ball land within your error range? If you use the digital cameras again for this part, you can determine if the □ight path falls within the range of possible □ight paths. A figure like the following might help you decide



5. Does the data support your modeling prediction? If not, what is likely the problem?
   - In your discussion make sure your other team members understand the part of the project that you did.
   - You might consider having a third team collect data for the ball shoot with the digital camera while the first two teams are working on the drag coefficient and the numerical prediction.

6. If there is time, polish your proposal.

10 Communicating Results II: Curve fits and Least Squares

To report our results, we often incorporate a curve fit. We have done this in the previous lab. But there is more to comparing data to a curve than just saying "it fit the equation." Immediately your esteemed colleagues should ask "*how well* did the equation fit the data."

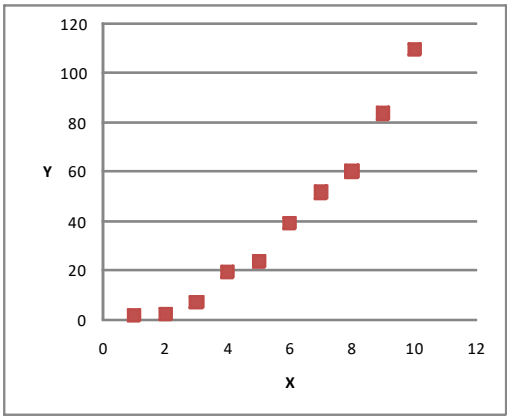
From a curve fit, we generally want three things:[1]

1. The fit parameters that our curve fit software found that make the equation type fit our particular set of data.
2. The uncertainty in those fit parameters
3. Some measure of how well the resulting fit equation matches the data.

In what follows I have derived the equations for Linear Least Squares. I have done this is excruciating detail. I think it is helpful to see where the method comes from so you know it's limitations. But you will not have to code up the derivation. You are principally interested in the results of the nasty math below. So take heart, and wade through the development of the Linear Least Squares method, but with the hope that the actual implementation will be fairly simple.

Least Squares

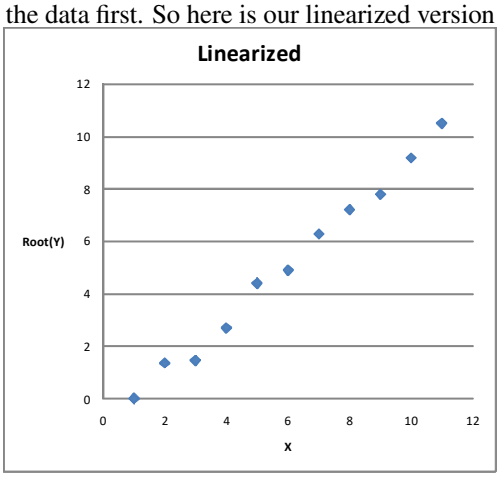
The idea behind Least Squares is to find a "best" function that fits a set of data.
For example, lets take the following data set.



We would like to fit this data with a line that in some way is the "best fit" for the data. But it is not clear what the "best fit" means. There are many choices we might make. We really need an agreed-on method that others will understand so we can communicate with the general science community.

Finding the fit parameters

Let's use what we have learned in lab so far. Over the last two weeks we learned that it is useful to linearize the data first. So here is our linearized version of the data:

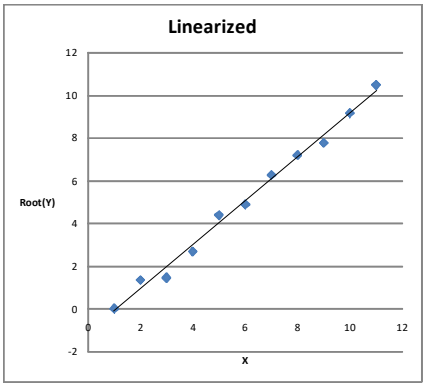


Now we will apply the idea of least squares on our linear data. Our data consists of many x and y points. From our MatLab experience, we recognize the notation x_i. This means that we have a column or array of x values and we have a counter, i, that tells us which row in the column (or which item in the array) we are talking about.
For every x_i we have a measured value y_i. Since we linearized our data, we want to find a straight line that predicts y-values for every x-value (all the x_i-values that we

measured, and all other x-values we did not measure). We will call the predicted values that we get from the straight line fit, Y_i (with a capital letter to show it is the predicted value instead of the measured value). So

$$Y_i = mx_i + b$$

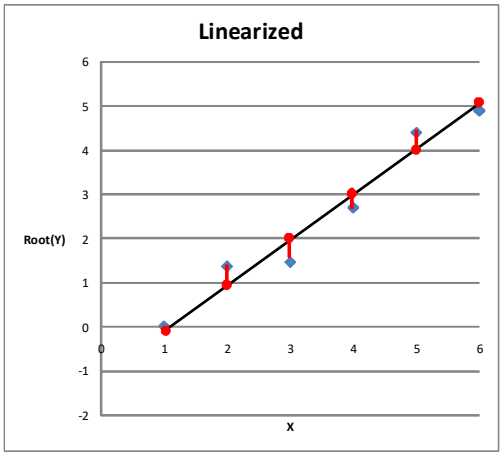
is the equation that predicts the Y_i. values. What we mean is that we take each of our x_i values and plug them into the equation $Y_i = mx_i + b$ to get our predicted Y_i values. The problem is, that we don't know m and b. If we did, we could just plot the line as shown in the next figure:



But we don't know the m and b. That is what we are trying to find.
Let's consider a way to do this. Suppose we take the difference between the two y-values, the measured and predicted

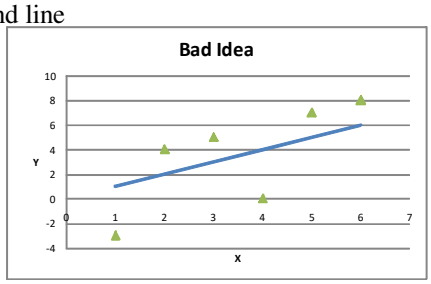
$$\Delta y_i = y_i - Y_i$$

where here Δy_i is just a name for the difference between the two y's. The differences Δy_i are shown on the next graph. the red circles are the predicted Y_i values. The blue diamonds are the measured y_i values, and the red lines connecting them represent the differences $\Delta y_i = y_i - Y_i$



This is where we can begin to define a criteria for "best" fit. We want to minimize all the Δy_i at the same time. If you have taken M112 or if your M112 class is far enough along, you know that we can minimize a function using derivatives, so is believable that we could actually find the minimum of our Δy_i values. But we have to think carefully about how we do this.

It would be tempting to just add up all the Δy_i and minimize this sum. But because some Δy_i values are negative and some are positive, this sum would usually be very close to zero no matter how big the Δy_i values. They all cancel each other out, but this would not minimize the distance between the line and the measurements. For example, consider this data set and line



The line does go through the data, and the quantity $\sum \Delta y_i \approx 0$, but the distance between *each* measured point and the line is not minimized. It is hard to say that this is a "best fit."

Instead, let's minimize the square of Δy_i. This eliminates the problem with negative values. We want to find the minimum difference between the Y_i's and the y_i's, so let's minimize

$$\Sigma_i \left(\Delta y_i\right)^2 = \Sigma_i \left(y_i - Y_i\right)^2$$

For convenience, I want to give this a name, say, the Greek letter χ or, since it is traditional, χ^2 as our variable (which is a little silly, so if you prefer, just call χ^2, M for

minimum). The point is that χ^2 is just a variable name.

$$\chi^2 = \Sigma_i \frac{(y_i - Y_i)^2}{\sigma_{y_i}^2}$$

Well, actually I divided what we had before by $\sigma_{y_i}^2$ which is the uncertainty in the y value for each y value. This makes some sense if we think about it. We are adding up the square of the deviation point by point. But if some points have larger error than others, it would be nice to weight the points that have the lowest error more strongly. We can do that if we divide the term for each point by the error in that point. This situation is ideal, but makes the math more difficult. So we are going to assume that all the σ_{y_i} are the same, so we can write χ^2 as

$$\chi^2 = \frac{1}{\sigma_y^2} \Sigma_i \left(y_i - Y_i\right)^2$$

Now, the Y_i's really depend on m and b. We can use our equation of the line,

$$Y_i = mx_i + b$$

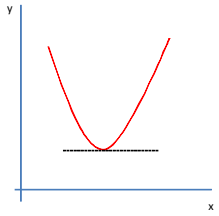
to write χ^2 as

$$\chi^2 = \frac{1}{\sigma_y^2} \Sigma_i \left(y_i - (mx_i + b)\right)^2$$

We want the m and the b that make χ^2 a minimum. We can take derivatives of χ^2 with respect to m and b and set the result to zero in order to find these values. This is our standard way of finding a minimum of a function using calculus,

$$\begin{aligned} \frac{\partial \chi^2}{\partial m} &= 0 \\ \frac{\partial \chi^2}{\partial b} &= 0 \end{aligned}$$

Then we solve the resulting equations for m and b. If this is not a familiar practice, think of what a derivative means. The derivative is the slope of the line. When the line has a minimum, the slope is zero.



so we can find the minimum by looking for the place where the slope is zero. That is where the derivative is zero.

To find the derivative it is useful to write out χ^2 in detail

$$\chi^2 = \frac{1}{\sigma_y^2} \Sigma_i \left(y_i - (mx_i + b)\right)^2$$

$$\chi^2 = \frac{1}{\sigma_y^2} \Sigma_i \left(y_i - (mx_i + b)\right)\left(y_i - (mx_i + b)\right)$$

$$\chi^2 = \frac{1}{\sigma_y^2}\Sigma_i \left(y_i^2 - 2y_i\left(mx_i + b\right) + \left(mx_i + b\right)^2\right)$$
$$\chi^2 = \frac{1}{\sigma_y^2}\Sigma_i \left(y_i^2 - 2y_i m x_i - 2y_i b + \left(mx_i + b\right)\left(mx_i + b\right)\right)$$
$$\chi^2 = \frac{1}{\sigma_y^2}\Sigma_i \left(y_i^2 - 2y_i m x_i - 2y_i b + m^2x_i^2 + 2bmx_i + b^2\right)$$
$$\chi^2 = \frac{1}{\sigma_y^2}\left(\Sigma_i y_i^2 - \Sigma_i 2y_i m x_i - \Sigma_i 2y_i b + \Sigma_i m^2 x_i^2 + \Sigma_i 2bmx_i + \Sigma_i b^2\right)$$
$$\chi^2 = \frac{1}{\sigma_y^2}\left(\Sigma_i y_i^2 - 2m\Sigma_i y_i x_i - 2b\Sigma_i y_i + m^2\Sigma_i x_i^2 + 2bm\Sigma_i x_i + Nb^2\right)$$
$$\chi^2 = \frac{1}{\sigma_y^2}\left(\Sigma_i y_i^2 + m^2\Sigma_i x_i^2 + Nb^2 + 2bm\Sigma_i x_i - 2m\Sigma_i y_i x_i - 2b\Sigma_i y_i\right)$$

Now we can just take the derivatives. Note that these are partial derivatives. We just take the derivative with respect to one variable at a time.

This is the important thing to remember. We have found χ^2 as a function that represents how far away the points of our fit line are from the actual data points, and we will find the slope, m and y-intercept, b, by minimizing χ^2. The rest is just mathematics.

Let's perform the minimizations starting with respect to m

$$\begin{aligned}\frac{\partial\chi^2}{\partial m} &= 0\\ &= \frac{1}{\sigma_y^2}\left(2m\Sigma_i x_i^2 + 2b\Sigma_i x_i - 2\Sigma_i y_i x_i\right)\end{aligned}$$

so

$$0 = 2m\Sigma_i x_i^2 + 2b\Sigma_i x_i - 2\Sigma_i y_i x_i$$

and now with respect to b

$$\begin{aligned}\frac{\partial\chi^2}{\partial b} &= 0\\ &= \frac{1}{\sigma_y^2}\left(2Nb + 2m\Sigma_i x_i - 2\Sigma_i y_i\right)\end{aligned}$$

so we have two messy equations

$$\begin{aligned}0 &= 2m\Sigma_i x_i^2 + 2b\Sigma_i x_i - 2\Sigma_i y_i x_i\\ 0 &= 2Nb + 2m\Sigma_i x_i - 2\Sigma_i y_i\end{aligned}$$

and we have two variables, m, and b. We can solve them simultaneously. Say, we solve the first one for b

$$b = \frac{\Sigma_i y_i - m\Sigma_i x_i}{N}$$

and plug it into the first equation and solve for m

$$0 = 2m\Sigma_i x_i^2 + 2\left(\frac{\Sigma_i y_i - m\Sigma_i x_i}{N}\right)\Sigma_i x_i - 2\Sigma_i y_i x_i$$
$$0 = 2m\Sigma_i x_i^2 + \frac{2\Sigma_i y_i \Sigma_i x_i}{N} - \frac{2m\Sigma_i x_i \Sigma_i x_i}{N} - 2\Sigma_i y_i x_i$$

$$0 = 2m\Sigma_i x_i^2 - \frac{2m\Sigma_i x_i \Sigma_i x_i}{N} + \frac{2\Sigma_i y_i \Sigma_i x_i}{N} - 2\Sigma_i y_i x_i$$
$$0 = m\left(\Sigma_i x_i^2 - \frac{\Sigma_i x_i \Sigma_i x_i}{N}\right) + \frac{\Sigma_i y_i \Sigma_i x_i}{N} - \Sigma_i y_i x_i$$
$$0 = m\left(N\Sigma_i x_i^2 - \Sigma_i x_i \Sigma_i x_i\right) + \Sigma_i y_i \Sigma_i x_i - N\Sigma_i y_i x_i$$
$$m\left(N\Sigma_i x_i^2 - \Sigma_i x_i \Sigma_i x_i\right) = -\Sigma_i y_i \Sigma_i x_i + N\Sigma_i y_i x_i$$

Then

$$m = \frac{N\Sigma_i \left(y_i x_i\right) - \Sigma_i x_i \Sigma_i y_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}$$

This is our equation for m. We can use this to find b now

$$b = \frac{\Sigma_i y_i - m\Sigma_i x_i}{N}$$
$$b = \frac{\Sigma_i y_i}{N} - \frac{m\Sigma_i x_i}{N}$$
$$b = \frac{\Sigma_i y_i}{N} - \frac{\Sigma_i x_i}{N}\left(\frac{N\Sigma_i \left(y_i x_i\right) - \Sigma_i x_i \Sigma_i y_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}\right)$$
$$b = \frac{\Sigma_i y_i}{N} - \left(\frac{\Sigma_i x_i \Sigma_i \left(y_i x_i\right) - \Sigma_i x_i \Sigma_i x_i \Sigma_i y_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}\right)$$
$$b = \frac{\Sigma_i y_i \left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}{N\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)} - \left(\frac{\Sigma_i x_i \Sigma_i \left(y_i x_i\right) - \Sigma_i x_i \Sigma_i x_i \Sigma_i y_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}\right)$$
$$b = \frac{N\Sigma_i y_i \Sigma_i x_i^2 - \Sigma_i y_i \left(\Sigma_i x_i\right)^2}{N\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)} - \left(\frac{N\left(\Sigma_i x_i \Sigma_i \left(y_i x_i\right) - \Sigma_i x_i \Sigma_i x_i \Sigma_i y_i\right)}{N\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}\right)$$
$$b = \frac{N\Sigma_i y_i \Sigma_i x_i^2 - \Sigma_i y_i \left(\Sigma_i x_i\right)^2 - N\left(\Sigma_i x_i \Sigma_i \left(y_i x_i\right) - \Sigma_i x_i \Sigma_i x_i \Sigma_i y_i\right)}{N\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}$$
$$b = \frac{N\Sigma_i y_i \Sigma_i x_i^2 - N\Sigma_i x_i \Sigma_i \left(y_i x_i\right)}{N\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}$$

So

$$b = \frac{\Sigma_i y_i \Sigma_i x_i^2 - \Sigma_i x_i \Sigma_i \left(y_i x_i\right)}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}$$

We have achieved our goal of finding the fit parameters,

$$\begin{aligned} m &= \frac{N\Sigma_i \left(y_i x_i\right) - \Sigma_i x_i \Sigma_i y_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)} \\ b &= \frac{\Sigma_i y_i \Sigma_i x_i^2 - \Sigma_i x_i \Sigma_i \left(y_i x_i\right)}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)} \end{aligned} \tag{10.1}$$

This may look bad, but remember we can use a computer to sum things up easily, and sums like $\Sigma_i \left(y_i x_i\right)$ are just numbers when the summing is done. So really m and b are

very easy to calculate.

The strategy should be to calculate $\Sigma_i (y_i x_i)$ and $\Sigma_i x_i$ and $\Sigma_i y_i$ and $\Sigma_i x_i^2$ first, then, just use the result to find m and b.

We have achieved our first goal. We have the fit parameters m and b for a straight line fit. We still need to find the uncertainty in the fit parameters and a measure of the goodness of the fit.

Finding the uncertainty in the fit parameters

Many graphing programs give fit parameters like m and b, but really we should not be satisfied until we have an uncertainty on m and b. This is often hard to get out of our graphing program. But we know how to do this also! Remember our standard error propagation equation

$$\sigma_f = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 \sigma_x^2 + \left(\frac{\partial f}{\partial z}\right)^2 \sigma_z^2 \cdots}$$

We can use this to find the uncertainty in m and b.

$$\sigma_m^2 = \left(\frac{\partial f}{\partial x_1}\right)_{x_1}^2 \sigma_{x_1}^2 \cdots \left(\frac{\partial f}{\partial x_N}\right)_{x_N}^2 \sigma_{x_2}^2 + \left(\frac{\partial f}{\partial y_1}\right)^2 \sigma_{y_1} \cdots \left(\frac{\partial f}{\partial y_N}\right)^2 \sigma_{y_N}$$

You don't see the square root sign because on the left hand side I have the square of the uncertainty (I have squared both sides). For our lab this week, we will assume the measurement uncertainty in the x_i is small enough we can ignore it (if not, we can use our knowledge of σ_{x_i}, but it requires more math). Then for the uncertainty in m we would have

$$\sigma_m^2 = \left(\frac{\partial m}{\partial y_1}\right)^2 \sigma_{y_1} + \ldots \left(\frac{\partial m}{\partial y_N}\right)^2 \sigma_{y_N}$$

Since we know that or our case all of the σ_{y_i} are the same we know σ_y^2. But if we don't know the uncertainty in each y_i we can still give a rough estimate the uncertainty based on the spread of the data.

Consider

$$\sigma_y = \sqrt{\frac{\Sigma_i \left(y_i - (mx_i + b)\right)^2}{N - 2}}$$

This gives us the spread of the Δy_i which is an estimate of the uncertainty (kind of an average of how far away the points are from the line–this is kind of an optimistic uncertainty estimate, much like a standard deviation). LoggerPro does this, and so does Excel. It's not a perfect estimate of our uncertainty in y, but it makes sense that the data can be at least as far off as this, since this is very like the average distance the data are from the fit line. The error must at least be as big as this.

But, there is a huge cost to this. If we substitute this σ_y into χ^2 we get

$$\begin{aligned}\chi^2 &= \frac{1}{\left(\sqrt{\frac{\Sigma_i (y_i-(mx_i+b))^2}{N-2}}\right)^2}\Sigma_i \left(y_i-(mx_i+b)\right)^2 \\ &= N-2\end{aligned}$$

no matter what, which is not helpful. By using this estimate of σ_y we have given up our goodness of fit estimate. This is why Excel and LoggerPro use a correlation estimate (called R^2) as their goodness of fit. This estimate of σ_y is much more risky, and should be avoided where possible. We will stick with actual uncertainties for our calculations. This limits us to equal uncertainties (all the $\sigma_{y_i} = \sigma_y$, a constant value). For those of you who take PH336 we will show you how to go beyond this, but for everyone else a good curve fitting routine will also do this right, so you just need to buy a better software package to go beyond what we will do in PH 150.

For the uncertainty in m we have

$$\begin{aligned}\sigma_m^2 &= \left(\frac{\partial m}{\partial y_1}\right)^2 \sigma_y^2 + \ldots \left(\frac{\partial m}{\partial y_N}\right)^2 \sigma_y^2 \\ &= \sigma_y^2 \left[\left(\frac{\partial m}{\partial y_1}\right)^2 + \ldots \left(\frac{\partial m}{\partial y_N}\right)^2\right]\end{aligned}$$

which will reduce to just

$$\sigma_m^2 = \sigma_y^2 \left[\Sigma_k \left(\frac{\partial m}{\partial y_k}\right)^2\right]$$

where k is an additional summation index.

All that is left is taking the derivative of m. But this looks like a difficult task. Remember that

$$m = \frac{N\Sigma_i \left(y_i x_i\right) - \Sigma_i x_i \Sigma_i y_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}$$

Let's write out our expression for m a bit.

$$\begin{aligned}m &= \frac{N\left(y_1 x_1\right) - y_1 \Sigma_i x_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)} + \frac{N\left(y_2 x_2\right) - y_2 \Sigma_i x_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)} + \ldots \\ &\quad + \frac{N\left(y_k x_k\right) - y_k \Sigma_i x_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)} + \ldots + \frac{N\left(y_N x_N\right) - y_N \Sigma_i x_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}\end{aligned}$$

This only makes it look messier! But consider, we need to take the derivative with respect to a particular y_k, holding all the other $y_i's$ constant. Only one term in our expression for m depends on y_k. So then we have only

$$\frac{\partial m}{\partial y_k} = \frac{N\left(x_k\right) - \Sigma_i x_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}$$

Again, this is because the partial derivative assumes all the y_i where $i \neq k$ are constants

along with all the x_i. Squaring this is not too hard

$$\left(\frac{\partial m}{\partial y_k}\right)^2 = \frac{N^2 x_k^2 - 2N\left(x_k\right)\Sigma_i x_i + \left(\Sigma_i x_i\right)^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}$$

To find our uncertainty in m we need a term like this for every value of k. So we wish to sum this over k

$$\sum_k \left(\frac{\partial m}{\partial y_k}\right)^2 = \sum_k \frac{N^2 x_k^2 - 2N\left(x_k\right)\Sigma_i x_i + \left(\Sigma_i x_i\right)^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}$$

$$\sum_k \left(\frac{\partial m}{\partial y_k}\right)^2 = \frac{\sum_k N^2 x_k^2 - \sum_k 2N\left(x_k\right)\Sigma_i x_i + \sum_k \left(\Sigma_i x_i\right)^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}$$

$$\sum_k \left(\frac{\partial m}{\partial y_k}\right)^2 = \frac{N^2 \sum_k x_k^2 - 2N \sum_k \left(x_k\right)\Sigma_i x_i + N\left(\Sigma_i x_i\right)^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}$$

Now both i and k run over the same numbers, so we can switch from k to i in the sums like $\sum_k \left(x_k\right)$

$$\sum_k \left(\frac{\partial m}{\partial y_k}\right)^2 = \frac{N^2 \sum_i x_i^2 - 2N \sum_i x_i \Sigma_i x_i + N\left(\Sigma_i x_i\right)^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}$$

$$\sum_k \left(\frac{\partial m}{\partial y_k}\right)^2 = \frac{N^2 \sum_i x_i^2 - N \sum_i x_i \Sigma_i x_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}$$

$$\sum_k \left(\frac{\partial m}{\partial y_k}\right)^2 = \frac{N\left(N \sum_i x_i^2 - \sum_i x_i \Sigma_i x_i\right)}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}$$

$$\sum_k \left(\frac{\partial m}{\partial y_k}\right)^2 = \frac{N}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}$$

which gives

$$\sigma_m = \sigma_y \sqrt{\frac{N}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}}$$

That was a lot of work, but we only need the result, and calculating it is fairly easy. We already have $\Sigma_i x_i^2$ and $\Sigma_i x_i$ from calculating m, itself, and we know σ_y.

We need to do the same thing for b

$$b = \frac{\Sigma_i y_i \Sigma_i x_i^2 - \Sigma_i x_i \Sigma_i \left(y_i x_i\right)}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}$$

$$\begin{aligned} b \quad = \quad & \frac{y_1 \Sigma_i x_i^2 - \left(y_1 x_1\right)\Sigma_i x_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)} + \frac{y_2 \Sigma_i x_i^2 - \left(y_2 x_2\right)\Sigma_i x_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)} + \ldots \\ & + \frac{y_k \Sigma_i x_i^2 - \left(y_k x_k\right)\Sigma_i x_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)} + \ldots \frac{y_N \Sigma_i x_i^2 - \left(y_N x_N\right)\Sigma_i x_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)} \end{aligned}$$

Again only one term survives if we take the derivative with respect to a particular y_k

$$\frac{\partial b}{\partial y_k} = \frac{\Sigma_i x_i^2 - x_k \Sigma_i x_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}$$

and we square this to put it in our uncertainty formula

$$\left(\frac{\partial b}{\partial y_k}\right)^2 = \frac{\left(\Sigma_i x_i^2\right)^2 - 2x_k \Sigma_i x_i \Sigma_i x_i^2 + \left(x_k \Sigma_i x_i\right)^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}$$

and sum up the contribution for each y_k

$$\sum_k \left(\frac{\partial b}{\partial y_k}\right)^2 = \frac{\sum_k \left(\Sigma_i x_i^2\right)^2 - \sum_k 2x_k \Sigma_i x_i \Sigma_i x_i^2 + \sum_k \left(x_k \Sigma_i x_i\right)^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}$$

We can simplify this a little

$$\sum_k \left(\frac{\partial b}{\partial y_k}\right)^2 = \frac{N\left(\Sigma_i x_i^2\right)^2 - 2\Sigma_i x_i \Sigma_i x_i^2 \sum_k x_k + \sum_k x_k^2 \left(\Sigma_i x_i\right)^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}$$

$$\sum_k \left(\frac{\partial b}{\partial y_k}\right)^2 = \frac{N\left(\Sigma_i x_i^2\right)^2 - 2\Sigma_i x_i \Sigma_i x_i^2 \sum_k x_k + \left(\Sigma_i x_i\right)^2 \sum_k x_k^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}$$

$$\sum_k \left(\frac{\partial b}{\partial y_k}\right)^2 = \frac{N\left(\Sigma_i x_i^2\right)^2 - 2\left(\Sigma_i x_i\right)^2 \Sigma_i x_i^2 + \left(\Sigma_i x_i\right)^2 \Sigma_i x_i^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}$$

$$\sum_k \left(\frac{\partial b}{\partial y_k}\right)^2 = \frac{N\left(\Sigma_i x_i^2\right)^2 - \left(\Sigma_i x_i\right)^2 \Sigma_i x_i^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}$$

$$\sum_k \left(\frac{\partial b}{\partial y_k}\right)^2 = \frac{\Sigma_i x_i^2 \left(N\left(\Sigma_i x_i^2\right) - \left(\Sigma_i x_i\right)^2\right)}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}$$

$$\sum_k \left(\frac{\partial b}{\partial y_k}\right)^2 = \frac{\Sigma_i x_i^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}$$

so, putting this in our formula for the uncertainties we get

$$\sigma_b = \sigma_y \sqrt{\frac{\Sigma_i x_i^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}}$$

Which gives the second part of our goal, we have the errors on the fit parameters.

$$\begin{aligned} \sigma_m &= \sigma_y \sqrt{\frac{N}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}} \\ \sigma_b &= \sigma_y \sqrt{\frac{\Sigma_i x_i^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}} \end{aligned} \tag{10.2}$$

We have achieved the second of our goals. we have the uncertainties in our fit

parameters. One goal to go, the goodness of fit parameter.

We should brie□y return to what we would do if we had σ_{y_i} that differed for each measured y_i point. Then we would use

$$\sigma_m^2 = \left(\frac{\partial m}{\partial y_1}\right)^2 \sigma_{y_1} \ldots \left(\frac{\partial m}{\partial y_N}\right)^2 \sigma_{y_N}$$

where we can now use our derivatives

$$\left(\frac{\partial m}{\partial y_k}\right)^2 = \frac{N^2 x_k^2 - 2N\left(x_k\right)\Sigma_i x_i + \left(\Sigma_i x_i\right)^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}$$

and

$$\left(\frac{\partial b}{\partial y_k}\right)^2 = \frac{\left(\Sigma_i x_i^2\right)^2 - 2x_k \Sigma_i x_i \Sigma_i x_i^2 + \left(x_k \Sigma_i x_i\right)^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}$$

to obtain

$$\sigma_m^2 = \left(\frac{N^2 x_1^2 - 2N\left(x_1\right)\Sigma_i x_i + \left(\Sigma_i x_i\right)^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}\right)^2 \sigma_{y_1} + \ldots \left(\frac{N^2 x_N^2 - 2N\left(x_N\right)\Sigma_i x_i + \left(\Sigma_i x_i\right)^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}\right)^2 \sigma_{y_N}$$

and

$$\begin{aligned} S_b^2 \quad = \quad & \left(\frac{\left(\Sigma_i x_i^2\right)^2 - 2x_1 \Sigma_i x_i \Sigma_i x_i^2 + \left(x_1 \Sigma_i x_i\right)^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}\right)^2 \sigma_{y_1} + \\ & \ldots + \left(\frac{\left(\Sigma_i x_i^2\right)^2 - 2x_N \Sigma_i x_i \Sigma_i x_i^2 + \left(x_N \Sigma_i x_i\right)^2}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)^2}\right)^2 \sigma_{y_N} \end{aligned}$$

This procedure is a little bit harder to implement in code, so we will stay with our assumption that all the σ_{y_i} values are the same and wait until PH336 to implement this.

Finding a measure of goodness of fit

We really already know how to do this too! We started out to minimize

$$\chi^2 = \frac{1}{\sigma_y^2}\Sigma_i \left(y_i - Y_i\right)^2$$

if we calculate χ^2 by taking our m and b

$$\chi^2 = \frac{1}{\sigma_y^2}\Sigma_i \left(y_i - \left(mx_i + b\right)\right)^2$$

we can compare χ^2 to zero and see how well we did. The value of χ^2 should be small–the smaller, the better the fit. This completes our requirements for a useful curve fit for a linear equation.

Summary

Our job, then, is to calculate

$$\begin{aligned} m &= \frac{N\Sigma_i (y_i x_i) - \Sigma_i x_i \Sigma_i y_i}{\left(N\Sigma_i x_i^2 - (\Sigma_i x_i)^2\right)} \\ b &= \frac{\Sigma_i y_i \Sigma_i x_i^2 - \Sigma_i x_i \Sigma_i (y_i x_i)}{\left(N\Sigma_i x_i^2 - (\Sigma_i x_i)^2\right)} \end{aligned}$$

and

$$\begin{aligned} \sigma_m &= \sigma_y \sqrt{\frac{N}{\left(N\Sigma_i x_i^2 - (\Sigma_i x_i)^2\right)}} \\ \sigma_b &= \sigma_y \sqrt{\frac{\Sigma_i x_i^2}{\left(N\Sigma_i x_i^2 - (\Sigma_i x_i)^2\right)}} \end{aligned}$$

and

$$\chi^2 = \frac{1}{\sigma_y^2}\Sigma_i (y_i - (mx_i + b))^2$$

This gives us a complete set of information for our linear curve fit. Note that terms like $\Sigma_i x_i$ and $\Sigma_i (y_i x_i)$ show up in more than one place in these equations. That means we can precalculate them, and just use the result. It would be good to precalculate $\Sigma_i x_i$, $\Sigma_i y_i$, $\Sigma_i (y_i x_i)$, $\Sigma_i x_i^2$, $(\Sigma_i x_i)^2$ then use them to calculate m, b, and finally $\sigma_{m,}\sigma_b$ and χ^2.

Finally, then, we have all we need to know about our straight line fit equation

$$y = mx + b$$

because we know m and b and we know our uncertainty in m and b and our goodness of fit χ^2.

A note about implementations in the field

In the built-in MatLab least squares routine and many other commercial products that tackle least squares use iterative methods to find m and b. This means that they need to start with initial guesses and work to find the minimums, given the uncertainties σ_{y_i}. You might guess that more uncertainty makes it harder to find the m and b values. Standard software like Excel and LoggerPro only use an estimate of the uncertainty σ_y. We should prefer routines that can use our best estimated uncertainties, so wherever we can, we choose more robust products (or write our own). If you are writing your own software, and in it you need to fit data, you may despair at having to write a least-squares type routine every time you do a calculation. But don't worry, there are great libraries that have least squares routines already coded that you can use. The Gnu

Science Library (GSL) is an example.

What happens if I can't linearize my equation?

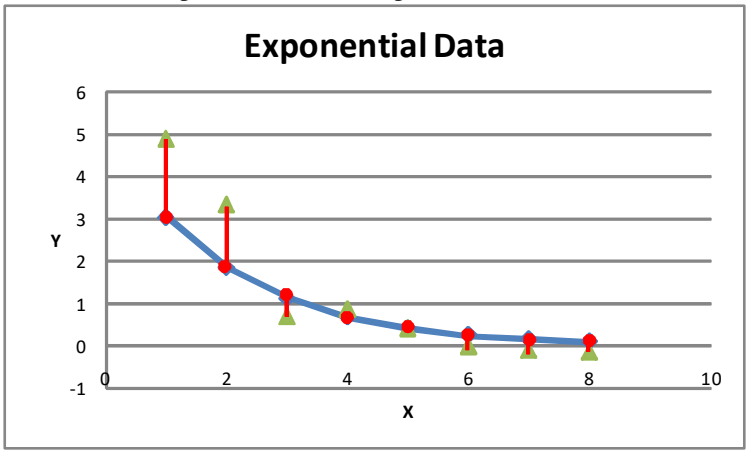
If you can't linearize your equation you can still use least squares, but you will have to do all this math to find your coefficients for each type of equation you wish to fit. This is what Excel, Logger Pro, and the MatLab easy fitting GUI have done. They pre-calculate the fit equations for several standard functions. We used

$$Y_i = mx_i + b$$

but you could do the same thing with

$$Y_i = ae^{-bx_i} + c$$

This works adequately for most beginning physics functions. But there are some things to worry about. For example, let's take our exponential function.



Because of the steep decent on the left hand side of the function, the errors are often larger. But on the right hand side, the Δy values are small. Situations like this will fool our least squares routine. It will do a great job on the □atter part of the curve, and not so good a job on the steep part. There are ways to help fix this, but it is always a concern with non-linear least squares. MatLab, Octave, Scilab, (and other programs like Gnuplot) will all do a general non-linear least squares to any function but it is often hard to get them to converge to a truly good fit. They require good initial guesses for the fit parameters. We will return to this problem in PH336.

Homework Assignment:

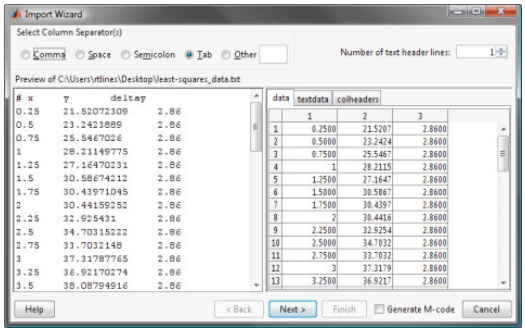
1. What is the equation for the slope in a linear least-squares fit?
2. What is the equation for the y-intercept in a linear least-squares fit?
3. What is the equation for the uncertainty in the slope parameter in a linear least-squares fit?
4. What is the equation for the uncertainty in the y-intercept parameter in a linear least-squares fit?
5. What is the equation for the "goodness of fit" parameter for linear-least squares.

Assignment

1. Transfer your homework to your lab notebook.
2. On I-learn, you will find a table of x and y values. Download this table onto a laptop.
3. Open the table using Excel, Notepad, the MatLab editor, or Word, or whatever.
4. Using the methods described above, determine the least squares linear fit (slope and intercept) to this data. This is easy in MatLab and there are instructions below. Report the slope, m and the y-intercept, b.
5. Determine the uncertainty (standard deviation) of the slope and intercept.
6. Calculate the goodness of the fit.
7. Print your work, and attach it to your lab notebook..
8. For the grader's convenience, record your final results in a table at the end of the lab entry. Include the following:
   - slope:
   - standard deviation of slope:
   - intercept:
   - standard deviation of intercept:
   - sum of squares of residuals:
9. Begin to brainstorm topics for your group experimental design project

MatLab Hints

To get your data into MatLab we need to import it. Under `File` choose `Import Data`. A dialog box will come up and you should choose the file least-squares_data.txt. A new dialog box will come up that looks like this



You should see the data on the right hand side. When you click the `Next` button, your

data will be saved in the variable named `data`. Our function will be used by typing `[m,b,deltam, deltab, CHI2]=lstsq(x,y,deltay, N)`, so we need to build the variables `x`, `y`, `deltay`, and `N`. The variable `data` is a two dimensional array, that is, structured as rows and columns like a spread sheet. Specifically, it is three columns of data that are 50 rows long. We need the first column to be our `x` values, the second to be `y` values, and the third to be our `deltay` values. The following code will create these variables:

```
x=data(:,1)
y=data(:,2)
deltay=data(:,3)
N=50;
```

Now that we have our data in the proper variables, we should consider how to code our formulas. Let's take our m equation as an example

$$m = \frac{N\Sigma_i \left(y_i x_i\right) - \Sigma_i x_i \Sigma_i y_i}{\left(N\Sigma_i x_i^2 - \left(\Sigma_i x_i\right)^2\right)}$$

We have several mini-calculations to do in this formula. We need to calculate

$$\Sigma_i x_i$$

$$\Sigma_i y_i$$

$$\Sigma_i \left(y_i x_i\right)$$

$$\Sigma_i x_i^2$$

and

$$\left(\Sigma_i x_i\right)^2$$

We have done such things before. The first calculation is just the sum of all the x_i. We could do this with the command

```
sum(x);
```

The same is true for the sum of the `y` values

```
sum(x);
```

Or we could write our sums as a loop. Let's try this for the x_i values

```
N=50; %the number of data points
sx=0; %the place to put our sum of the x values
for i=1:N
 sx=sx+x(i);
end;
```

This takes more lines, but is more flexible. Suppose we take on $\Sigma_i \left(y_i x_i\right)$. We could write this as

```
%the place to put our sum of the product of the x and y
values
sxy=0;
```

% now perform the sum

```
for i=1:N
 sxy=sxy+x(i)*y(i);
end;
```

We might do something similar for $(\Sigma_i x_i)^2$

If we call the sum of the x values sx, and the sum of the y values sy and use sxy as the sum of the product of the x and y values and finally $sx2$ as the sum of the x values squared, then our m equation is

$$m = \frac{N \times sxy - sx \times sy}{\left(N \times sx2 - (sx)^2\right)}$$

We do the same thing for b and σ_m, σ_b, and χ^2.

It also might be a good reminder to see how to plot the data. The command

```
plot(x,y,'o')
```

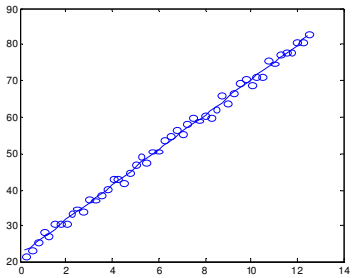
will do this. We would also like to plot our line through the data after we run our least-squares code. Our code created the slope `m` and the y-intercept, `b`. We can generate our line by calculating y values for each of our x values based on `m` and `b`.

```
yfit=m.*x+b
```

Then

```
hold on;
plot(x,yfit);
hold off;
```

Will complete the plot. It should look something like this:



11 Work on Student Designed Experiments

No Assignment

12 Work on Student Designed Experiments

No Assignment

13 Work on Student Designed Experiments: Final Paper

No Assignment

14 Final Exam

No Assignment

References

[1] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in Fortran 77*. Cambridge, UK: Cambridge University Press, 1992.