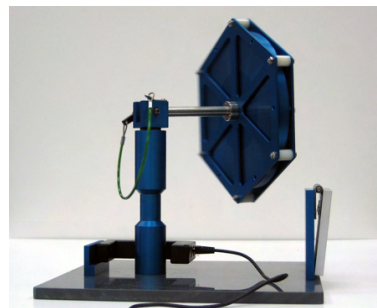
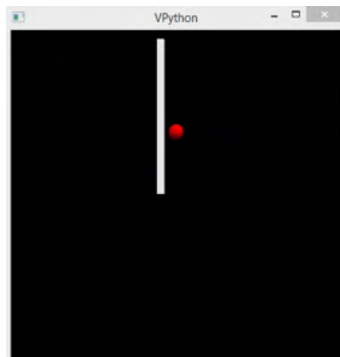
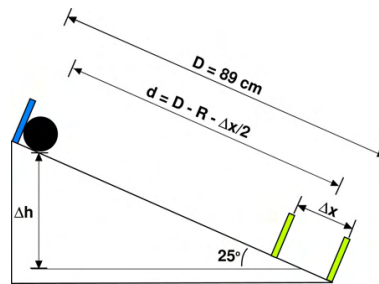
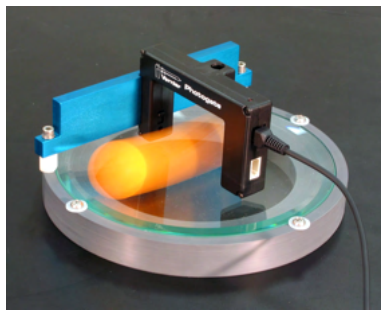


Introductory Physics Laboratory Manual

MECHANICS

Physics 141 - Elementary Laboratory I
Fall 2021



Blake Hipsley and Wolfgang Lorenzon, Editors
Department of Physics
University of Michigan
Ann Arbor, MI

Contents

Acknowledgments	ii
Introduction to Physics 141 Laboratories	iii
Laboratory 1 – Introduction to Statistics and Data Analysis	1
Laboratory 2 – Computational Kinematics	13
Laboratory 3 – Motion with Uniform Acceleration	35
Laboratory 4 – Projectile Motion	46
Laboratory 5 – Projectile Motion with Air Drag	55
Laboratory 6 – One-Dimensional Collisions	76
Laboratory 7 – Two-Dimensional Collisions	87
Laboratory 8 – Rotational Motion: The Inclined Plane	99
Laboratory 9 – Rotational Motion: The Gyroscope	110
Laboratory 10 – Modeling Gravitational Motion	125
Laboratory 11 – Waves and Sound	146
Appendix A – Software Tips and Tricks	159

Acknowledgments

The editors gratefully acknowledge the numerous suggestions for improvement provided by the University of Michigan graduate student instructors from the 2008-2021 academic years. Special thanks go to the previous Lead GSIs who developed and refined many of the introductory mechanics labs: Eric Gonzalez from Fall 2019 through Winter 2021, Shruti Paranjape from Fall 2017 through Spring 2019, Ojan Khatib-Damavandi from Fall 2016 through Spring 2017, Yun Suk Eo from Fall 2014 through Spring 2016, Daniel Shafer from Fall 2012 through Spring 2014, Jacob Ketchum from Fall 2010 through Spring 2012, and Bethany Percha from Fall 2008 through Spring 2010, who vastly improved and refined many of the introductory mechanics labs. The computational labs were developed by a team consisting of graduate students, postdoctoral fellows, staff, and instructors. The team members, in alphabetical order, were James Antonaglia, Xiyu Du, Dr. Joseph Gennaro, Ben Girodias, Tanvi Gujarati, Ojan Khatib-Damavandi, Dr. Nicole Michelotti, Ansel Neunzert, Dr. Yuri Popov, Alec Tewsley-Booth. Special thanks to Dr. Bradford Orr, Dr. Keith Riles, and Purdue University Physics Department for their reference materials. Thanks to REBUILD and the U-M Physics Department for their guidance and enabling support. Finally, very special thanks go to Michelle Coeman, our current supervisor of the introductory physics labs. Without her, the labs would not be in the shape they are today.

Introduction to Physics 141 Laboratories

Why take a physics lab?

Many of you are involved in degree programs for which the physics laboratories have been decreed mandatory. You may hope to pursue careers in medicine and engineering, highly technical fields which are growing more so every year. The ability to ask the right questions, perform appropriate measurements, and extract pertinent information from data will be critical to your success. Even those of you who have no intention of pursuing scientific careers will face decisions that require technical judgment, such as those regarding health care, technology, and safety. As a society, we are beset by technical problems such as global warming, nuclear and biological terrorism, genetic engineering, etc. Determining priorities for these issues will require good judgment in spite of our inevitable lack of access to detailed information. As these issues become more complex, you will find that previous experiences with physical phenomena and statistical analysis will improve your ability to estimate the probable validity of various claims.

Course Goals

The goals of Physics 141 are to

- improve your understanding of concepts taught in Physics 140.
- teach you quantitative techniques for reasoning from empirical data.
- teach you general methods for experimentally testing theoretical hypotheses.

Course Structure

Physics 140 lecture and Physics 141 lab are taken concurrently. Particular efforts have been made to synchronize lecture and lab. This means that labs will not only refer to material covered in lecture, but lecture exams will also cover material covered in labs. Students in Physics 141 collaborate in groups of two or three. Each laboratory class is self contained: all data must be taken, analyzed, interpreted and reported before leaving the classroom. Most labs contain computational components and require the use of Python in a Jupyter notebook environment, which students need to fill out and submit electronically. Students are expected to familiarize themselves with each experiment before class so that they can complete a short quiz at the beginning of each class, with the exception being for fully computational labs, which will have pre-laboratory assignments in place of the quizzes. Pre-laboratory assignments must be completed before the start of each class to receive credit. Partners will receive the same score for the laboratory reports, but individual scores for the quizzes and the pre-laboratory assignments. There is no final exam.

Lab Reports The laboratory reports will include three main elements:

- Raw experimental data.
- Data analysis, including calculations.
- Questions and conclusions.

Since students in Physics 141 collaborate in groups, each report represents a combination of two or three separate students' analyses and ideas. Try your best as laboratory partners to share the work fairly; more importantly, share your difficulties and insights related to the problem in front of you. You will find that this is not such bad advice for other collaborative endeavors. Your GSI will have you change laboratory partners every week to ensure that your average score is a function of your own ability and is not significantly affected by your laboratory partners.

The **laboratory reports** will account for **80%** of the lab score.

Saving and Submitting Lab Reports Lab reports are submitted via Canvas. You only need to make one submission of files as a group to the Canvas assignment and it will count for all group members. To submit an assignment, login to the 'PHYSICS 141 FA 2021' course page, select 'Assignments', select the lab assignment, then click on the option to upload a file. Lab reports are due at the end of the lab period. Students will lose a minimum of 4 points (10% of total points possible) for late submissions. Instructions for downloading the lab reports in the proper format will be provided at the end of each template and state which files will need to be submitted. Note that any files saved locally on the lab computer are deleted when a student logs off. In the event that your lab computer crashes and needs a reboot, all of your work is lost; the only way to backup a file is to copy it to the temp folder on the network drive, where it can be recovered by your instructor. It is the student's responsibility to save copies of the lab report periodically during class.

Pre-Lab Quizzes and Pre-Lab Assignments At the beginning of each laboratory (with the exception of the fully computational labs) you will be required to complete a 5-minute Canvas quiz before we begin. Note that you are responsible for lecture material up to the time of the lab; if you took the lecture in a previous term, you may need to go back and review some key concepts. The quizzes are intended to ensure that you are adequately prepared. They are mainly based on the laboratory manual chapter for that week's experiments, but they may occasionally test your understanding of the physics concepts as well. The quizzes will start exactly on the hour. There are no make-ups for the quizzes, and you will lose precious time if you show up late. Your lowest quiz score will be dropped when calculating your final lab score.

For the fully computational laboratories, there will be a pre-laboratory assignment that must be completed before class. For these labs, there will be no quiz. If you submit the pre-laboratory assignment more than 10 minutes (i.e., 10 minutes after the start of class) late, you will receive no

credit. Each pre-laboratory assignment will be graded and will receive the same amount of credit as a quiz.

The **pre-laboratory assignment and quizzes** will account for **20%** of the lab score.

Absences It is mandatory to attend every lab. Generally, laboratory make-ups are permitted only for absences due to medical reasons, family emergencies, religious holidays, and university sponsored events. Except for documented medical and family emergencies, permission **must** be obtained beforehand. Make-ups must be completed within one week of the regular schedule for the missed lab. To schedule a make-up, talk to your GSI as soon as possible and they will allow you to schedule a make-up lab through the calendar on Canvas. In order to receive credit, you must notify your GSI that the make-up is completed the same day that you complete it. If you receive a make-up, you are still required to complete the corresponding quiz on Canvas upon arriving to the make-up lab, or, for fully computational labs, are required to complete and submit the pre-laboratory assignment by the ten minute deadline following the start of your scheduled make-up lab.

Note: Make-up labs are **not** granted for exam scheduling conflicts. You must either ask for an alternate exam time that does not conflict with your lab schedule or enroll in an open section of lab that does not conflict with your exam schedule.

Since attendance is mandatory, we note that **any unexcused absence is grounds for a failing grade (F) in this course**, regardless of your average score. If you find yourself in this situation, please contact your instructor and/or the lead GSI Blake Hipsley (bhipsley@umich.edu) immediately to discuss the appropriate course of action.

In case your GSI has not arrived 10 minutes after the start of your scheduled class time,

1. Contact Michelle Coeman at 1241 Randall, if it is between 8 am - 5 pm
2. If she is unavailable, contact Student Services Office (SSO) located at 1440 Randall Lab, if it is between 8 am - 5 pm.
3. Leave, if it is after 5 pm.

Either way, your GSI will be in contact with you soon.

Tardiness It is imperative that each student be ready to begin class on the hour.

For labs with quizzes, if a student arrives a few minutes into the quiz, they may use the limited quiz time remaining to complete the quiz but will not be given extra time. Students who miss the pre-lab quiz entirely will receive zero points for the quiz. For the computational laboratories, if a

student submits the report 10 minutes after class begins (i.e. 10 minutes after the hour), no credit will be given for the pre-laboratory assignment. Late students will not receive any credit for the portions of the lab completed by their partners alone (i.e. before the student has arrived).

Grading Because each laboratory is different, they will each have different grading criteria in which data, analysis, and questions will be given differing weights. Your GSI will grade every laboratory report in the same format and provide feedback to you through Canvas. If at any point you have concerns about the way you were scored on a quiz or lab report, you are expected to contact your GSI and explain the situation within one week of receiving your graded assignment.

Final grades will be determined based on the following scale:

Total Percentage Achieved	Grade Range
92.0 – 100	A+ / A / A–
80.0 – 91.99	B+ / B / B–
70.0 – 79.99	C+ / C / C–
60.00 – 69.99	D+ / D / D–
≤ 59.99	F

Each section is graded independently by its own GSI; however, we do recognize that not all GSIs grade with exactly the same criteria, so if your section's class median is below 92.0%, we will lower the A– / B+ cutoff close to the class median. Note that we will never raise the cutoff (i.e. the letter grades above are an absolute guarantee, no matter how “easy” your GSI is). The assignment of + and – to a grade is determined by the grading distribution in each section.

Accommodations Please contact the Lead GSI, Blake Hipsley (bhipsley@umich.edu) before the term begins if you need any accommodations. The lab reports of students with SSD certification (and other students in their group) will be graded out of the work they have done, given that they have completed at least two-thirds of the lab. Quizzes will receive extra time, starting 5 minutes before the hour and ending 5 minutes past the hour.

Lab Cleaning Policy It is mandatory to clean the table and restore the equipment to good order before you leave the lab. About 2,000 students take this lab course every semester, so we want to make sure that everybody has a clean and orderly place to start their labs. This includes reporting to your GSI any equipment that does not work and needs to be replaced. Your group will lose points from your lab report if you fail to clean up.

Laboratory Safety We have endeavored to make every aspect of the experimental procedures as safe as possible, consistent with the physical requirements of the phenomena being investigated. This does not mean that safety issues can be ignored or disregarded. The two common hazards that one must be aware of are the 115-volt AC power supply used for most of the equipment and the bits of felt dust flying off of the felt wheels used in the gyroscope laboratories. It is also possible to knock over equipment, drop heavy objects on your toes, etc. (wearing shoes is thus a requirement). If you see any obvious hazards, please report them to your GSI immediately. If a student commits deliberate safety violations that might endanger other students, they will be barred from the lab for the remainder of the term.

Academic Integrity

Please familiarize yourself with the LSA policies on Academic Integrity (<http://www.lsa.umich.edu/academicintegrity/>). Note that suspected cases of academic dishonesty (cheating) will be forwarded directly to the Assistant Dean for Student Academic Affairs and any student found guilty will receive appropriate penalties decided based on the severity of cheating, in addition to penalties imposed by the College. Cheating includes copying answers to pre-lab assignments, using other groups' work during the lab, fabricating data, etc.

In case all else fails ...

For questions regarding details of course material, grading, attendance, performance, etc., the first person to talk to is your GSI. We expect that most issues will be resolved between you and your GSI without further intervention. However, from time to time, problems arise that the GSI is unable to adequately resolve. In such cases, you are invited to contact the Lead GSI, Blake Hipsley (bhipsley@umich.edu).

Physics Department Offices

Please visit (or contact) the appropriate office below if you are in need of assistance during business hours:

Introductory Physics Laboratories

1241 Randall

(Laboratory Manager: Michelle Coeman – mjlove@umich.edu)

(Lead GSI: Blake Hipsley – bhipsley@umich.edu)

Physics Student Services

1440 Randall

(physics.sso@umich.edu)

Computing and Technology Support (CaTS)

2428 Randall

(<https://cats.lsa.umich.edu/>)

***** Call 911 directly in the event of a medical emergency *****

Laboratory 1

Introduction to Statistics and Data Analysis

Introduction

Statistics is the science of obtaining meaningful information from data. The methodology of statistics allows scientists to separate information from noise and to quantify how certain they are about the results of their measurements. If scientists did not consistently analyze the results of their experiments within a statistical framework, they would frequently be misled by anecdotal evidence or the results of a few anomalous trials.

In this laboratory, we will explore a few of the most important concepts in statistics, and we will also discuss what it means for measurements to be “accurate” and “precise.” We will then use these tools to see how well some real data fit to a simple model of uniform acceleration. The concepts that we study here will provide a framework for our analysis of experimental data throughout the rest of the course.

Theoretical Background: Systematic and Statistical Error

In general, there are two types of error that creep into experimental data: systematic and statistical error. **Systematic errors** are biases in measurement that often lead to measured values being consistently too high or too low. These can be the result of poorly calibrated equipment or consistent human errors that repeat themselves in multiple trials. Systematic errors are distinguished from other errors because the addition of more data does *not* reduce systematic error.

Statistical errors are a very different type of error. They do not arise from any equipment miscalibration or other “mistake” on the experimenter’s part. A statistical error is any uncertainty that can be reduced by repeating the measurement a large number of times. Many statistical errors in experimental measurements are caused by unknown and unpredictable changes in the experiment or measurement apparatus that lead to “noise” in the measurement. Since the processes that generate noise tend to be random, it is expected that averaging over multiple trials will reduce the noise.

Now consider a situation in which you have an experimental quantity that you are trying to measure, such as g , the gravitational acceleration at Earth’s surface. We accept that the “true” value of

this quantity exists (it is out there somewhere), but you do not know what it is. So, you devise an experiment to measure it, and you perform a series of trials.

Precision refers to the degree of mutual agreement among your measurements. All of the measurements may be very far from the true value of whatever quantity you are trying to measure, but as long as the measurements themselves are close together, your data are precise. In Figure 1, the bottom-left target represents a set of measurements that is precise but not accurate.

Accuracy is the degree of conformity of a measured quantity to its true value. For a set of measurements to be considered accurate, the set must be centered on, while clustered relatively randomly around, the “true” value, but each individual measurement does not need to be close to other measurements or the true value. In Figure 1, the top-right target represents a set of measurements that is accurate but not precise.

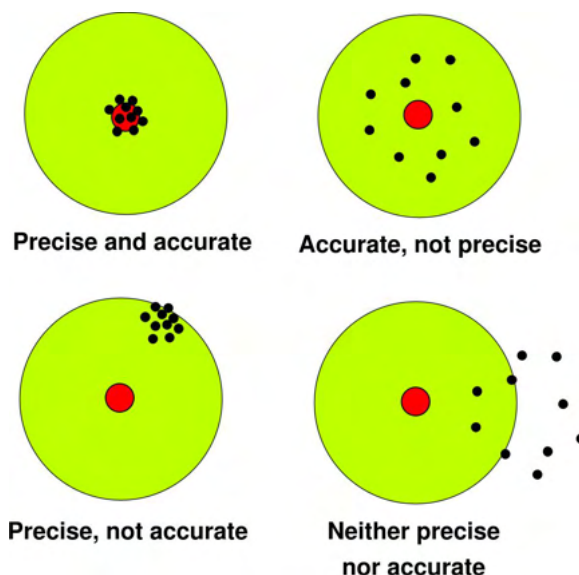


Figure 1: The target analogy illustrates the difference between precision and accuracy. The true value of the quantity is at the center of the target, and the black dots represent attempts to measure the quantity.

Problem 1.1 According to the discussion above, does systematic error have a greater impact on a measurement’s precision or its accuracy? What about statistical error?

Problem 1.2 Can the impact of statistical error be reduced by simply repeating a measurement many times and taking the average over all of those trials? Why or why not? Can the impact of systematic error be reduced by repeating a measurement? Why or why not?

Theoretical Background: Basic Statistical Concepts

The **mean** is defined as the arithmetic average of a set of values obtained from a series of N measurements or observations. The formula for the mean \bar{x} of quantity x for N observations is

$$\bar{x} \equiv \frac{1}{N} \sum_{i=1}^N x_i, \quad (1)$$

where the individual measurements of x are labeled x_i . The **variance** is closely related to the arithmetic average of the square of the difference between the value of each observation and the mean. It is given by

$$\sigma^2 \equiv \frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2. \quad (2)$$

It is not exactly equal to the arithmetic average because the denominator includes $N - 1$ instead of N . There is a good reason for this (related to the concept of **degrees of freedom**), and a full explanation can be found in any introductory statistics textbook. In Eq. 2, the differences are squared so that positive and negative differences do not cancel each other out. The variance is a measure of how closely the data cluster around the mean. If the variance is small, most data points are close to the mean. The **standard deviation** is the square root of the variance and is usually written as σ , defined as

$$\sigma \equiv \sqrt{\frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2}. \quad (3)$$

The units of σ are the same as those of whatever it is you are measuring, giving it a practical advantage over the variance. For example, if all of your x_i have units of meters, their standard deviation will also have units of meters. Their variance, however, will have units of meters *squared*, which makes it harder to interpret.

The **probability** associated with a particular outcome of a measurement is the likelihood that this outcome will occur. For example, if you toss a fair coin, the probability of it landing on “heads” is $1/2$, or 0.5 . Probability is always expressed as a number between 0 and 1. The **probability distribution function** for a measurement is the set of probabilities for every possible outcome of that measurement. For example, for a single coin toss, there are two outcomes {heads, tails}, and each has a probability of $1/2$ of occurring. Therefore, the probability distribution function for a single coin toss is $\{1/2, 1/2\}$. There are several important things to know about probability distribution functions:

- The sum of all the elements of a probability distribution must always be 1, because *something* will actually happen each time.
- Probability distributions fall into two categories: **discrete** and **continuous**. Discrete prob-

ability distributions occur when the outcome of a measurement can only be one of a fixed number of values, which are typically integers. The single coin toss is an example. Continuous probability distributions occur when the outcome can be any decimal value. An example would be measuring a person's exact height.

- One very important type of continuous probability distribution function is the **Gaussian** (or normal) distribution, which has the functional form

$$P(x) = A \exp\left(-\frac{(x - \bar{x})^2}{2\sigma^2}\right), \quad (4)$$

where A is the height of the distribution. This formula actually represents a set of distributions, since each unique choice of \bar{x} and σ results in a different distribution. These distributions are shaped like those in Figure 2, which have the classic “bell curve” shape. In Eq. 4, \bar{x} is the mean of the distribution and also the value of the variable x that is at the center of the distribution. The σ parameter is the standard deviation of the distribution, essentially a measure of the distribution's width.

- The **Central Limit Theorem** states that the mean of a large number of independent random processes (with some unknown probability distribution of their own) will be approximately normally distributed. This is the reason why so many random processes in nature (such as mechanical “noise”) follow normal distribution functions – they are the result of multiple independent processes acting together.
- The actual distribution of your set of data can be viewed by a **histogram** like that in Figure 3. A histogram is a bar chart of a quantitative measurement value that shows how many values are in various intervals (bins) of the data. The horizontal axis shows the scale of values that are measured. The vertical axis shows the number of times the measurement occurred (frequency). A typical histogram consists of many bars. The width represents the interval that a single bar covers. The height represents the frequency within that interval.

Problem 1.3 In an experiment, you take a fair coin and flip it twice. You measure the total number of heads that appeared in those two flips and call that number x . In this experiment, what is the probability distribution function for the number of heads that appear? What is the mean number of heads that appear?

Problem 1.4 Look at Figure 2. Do the two distributions on the left have different means, different standard deviations, or both? What about the distributions on the right?

Now, suppose that you are making measurements of a certain quantity which we will call x . If you measured x an infinite number of times and made a histogram of your experimental values, your

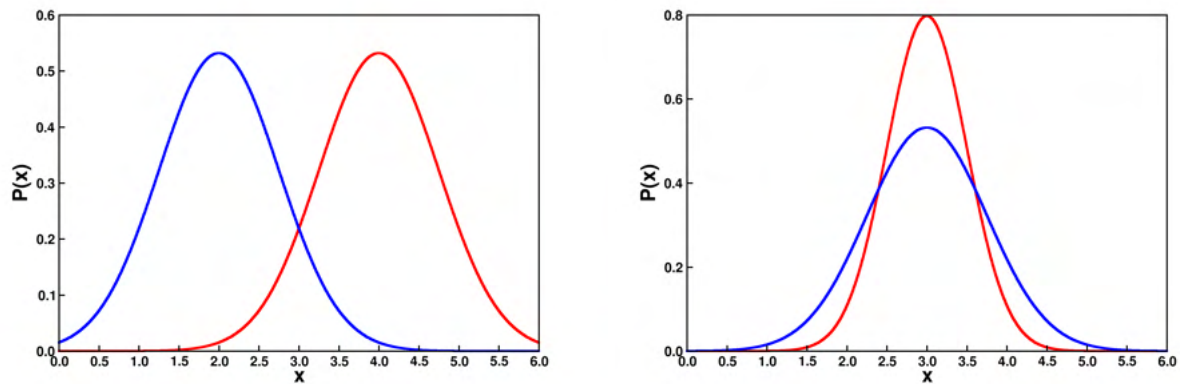


Figure 2: Examples of different normal distributions.

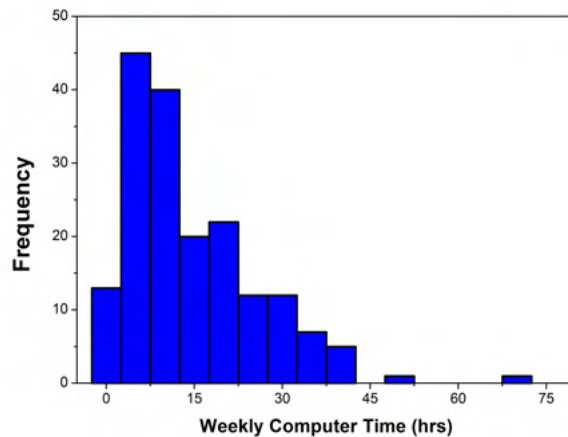


Figure 3: An example of a histogram showing weekly hours college students spend on a computer.

graph would look something like the one in Figure 4. Without worrying too much about the values on the vertical axis, think about what this graph means. It means that, for any single measurement, you are likely to get a result that is close to the true value of x (which is the value at the center of the distribution), and you are less and less likely to get results that are further and further from the true value.

As you can see from Figure 4, your chances of obtaining, from a single measurement, a value that is close to the true value of x are good, but they are not *that* good. There is still a sizeable probability that you will think x is something very far from its actual value. The severity of this

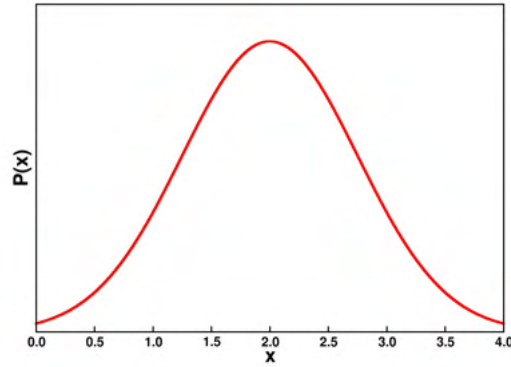


Figure 4: The red curve shows the result of taking an infinitely large number of measurements of quantity x and making a histogram of the values. It also represents the probability distribution function $P(x)$ for the outcome of a single measurement of x .

problem can be lessened by repeating the measurement several times and taking the mean of your experimental values. For example, assume that you try measuring x again, but this time you make six separate measurements and take their *mean* to be the measured value of x .

What would happen? It is true that some of those values might still be far from the true x . But it is very unlikely that *all* of your values would be far from the true x and even less likely that they would all be off in the same *direction*. More likely, some of your six measurements would be too low, and some would be too high. By taking their mean, you would cancel out some of the uncertainty (statistical error). *The chance that your reported value is close to the true value increases as you take the mean of more and more measurements.*

We quantify this by calculating a quantity known as the **error of the mean**. For a set of N measurements, it is given by

$$\sigma_{\bar{x}} \equiv \frac{\sigma}{\sqrt{N}}, \quad (5)$$

where σ is the standard deviation of the N measurements as defined in Eq. 3. Given that σ is **a measure of the uncertainty associated with taking a single measurement**, $\sigma_{\bar{x}}$ is **a measure of the uncertainty associated with taking the mean of N measurements**. As you can see, this formula claims that you can determine the mean to arbitrary precision by simply taking a larger and larger number of measurements: As N becomes larger and larger, the error of the mean becomes smaller and smaller.

This is a very important result because statisticians have shown that, provided certain assumptions

are met, we can expect the true value of whatever we are trying to measure to be within the interval

$$\begin{array}{ll} \bar{x} \pm \sigma_{\bar{x}} & 68.3\% \text{ of the time,} \\ \bar{x} \pm 2 \sigma_{\bar{x}} & 95.4\% \text{ of the time,} \\ \bar{x} \pm 3 \sigma_{\bar{x}} & 99.7\% \text{ of the time.} \end{array}$$

Note that when we use notation like $a \pm b$ for an interval, as in the above, we are referring to an interval $(a - b, a + b)$, where $a - b$ is the *lower bound* and $a + b$ is the *upper bound*.

In other words: If we repeat our measuring (and averaging) process an infinite number of times, the interval $\bar{x} \pm \sigma_{\bar{x}}$ will include the true value for 68.3% of our trials. Alternatively, for a single set of N measurements, there is a 68.3% chance that the mean of our measurements is within $\pm \sigma_{\bar{x}}$ of the true value. This is because the distribution of the means of repeated measurements is also a Gaussian but with a smaller standard deviation. Keep in mind the difference between a single measurement and the average of a set of N measurements: For a single measurement, we get one value of x , and the distribution of repeated single measurements is a Gaussian with a standard deviation of σ ; for a single set of N measurements, we get one value of the mean \bar{x} , and the distribution of many such means is a Gaussian with a standard deviation of $\sigma_{\bar{x}} = \sigma / \sqrt{N}$.

Perhaps you have heard the term **confidence interval** before; this is exactly what we are describing here. Scientists often report their results as a mean and a 95% confidence interval, although the 95% is just a common convention with no special significance. The 95% confidence interval is given by

$$\bar{x} \pm 1.96 \sigma_{\bar{x}}.$$

Practically speaking, this means that the scientists are “95% sure” that the interval they are reporting contains the unknown true value of the quantity of interest.

As an example, suppose we are interested in a certain physical quantity X . Also, assume that based on some model, we have obtained a theoretical value $X = x_{\text{th}}$. We want to see whether the model correctly predicts the true value of quantity X . After several measurements, we can get a mean experimental value \bar{x}_{ex} . We can also calculate the error of the mean and thus the 95% confidence interval for our mean value \bar{x}_{ex} . Assuming that there is no systematic error in our experiment to shift \bar{x}_{ex} away from the true value, we are 95% sure that the confidence interval contains the true value of X and if the model is “valid,” x_{th} .

But what if the confidence interval does *not* contain x_{th} ? In that case, we say the difference between the mean and the theoretical value of quantity X is “statistically significant” at the 95% confidence level. We may then conclude that the model we had at hand was not a valid one and could not produce the true value of X . However, note that in statistics, we can only *reject* hypotheses, never prove them to be true. So, in the example above, if the confidence interval includes x_{th} , we can only conclude that we have failed to reject the hypothesis that X is correctly predicted by our theoretical model.

Problem 1.5 If the standard deviation of four length measurements is 0.1 cm, what is the error of the mean?

Problem 1.6 If the error of the mean for four time measurements is 0.05 s, how many trials need to be performed for the error of the mean to be 0.01 s or less? What about 0.001 s or less?

Problem 1.7 Notice again that $\sigma_{\bar{x}}$ depends inversely on \sqrt{N} , where N is the number of independent measurements that go into the average value \bar{x} . What happens to the width of a particular confidence interval as N increases? What does this tell you about your confidence in the accuracy of your measurement as N increases?

Experimental Background: The Ultrasonic Sensor

In this laboratory, we will take position measurements of a falling object using a device called an *ultrasonic sensor* as seen in Figure 5. This device is capable of measuring the position of an object to a precision of a few millimeters. The sensor produces 1-millisecond bursts of 50 kHz sound waves. It works in a way similar to the locating system employed by bats: it measures the time delay of the echo to establish the distance from it to the object.

Experiments

Experiment 1: Fitting Data to a Model

To demonstrate some of the statistical concepts described above, we will begin by fitting several sets of data to a simple model. In this experiment, our model will be the y position as a function of time for an object in free-fall:

$$y = y_0 + v_0 t - \frac{1}{2} g t^2. \quad (6)$$

Our data will be a set of real measurements of g obtained by dropping a rubber ball several times and fitting Eq. 6 to the y vs. t measurements provided by the ultrasonic sensor. Each of the data sets will look like that in Figure 6. Since the ball is small and heavy, we can safely assume that the effect of air drag will be negligible.

1. Use the vacuum release system to clamp the ball in place directly above the ultrasonic sensor. You can apply pressure to the ball by compressing the vacuum hand pump a few times while it is in place.



Figure 5: A photograph of the ultrasonic sensor and vacuum release system used for measuring gravitational acceleration. The ultrasonic sensor is protected beneath a wire mesh housing in the bottom of the bucket.

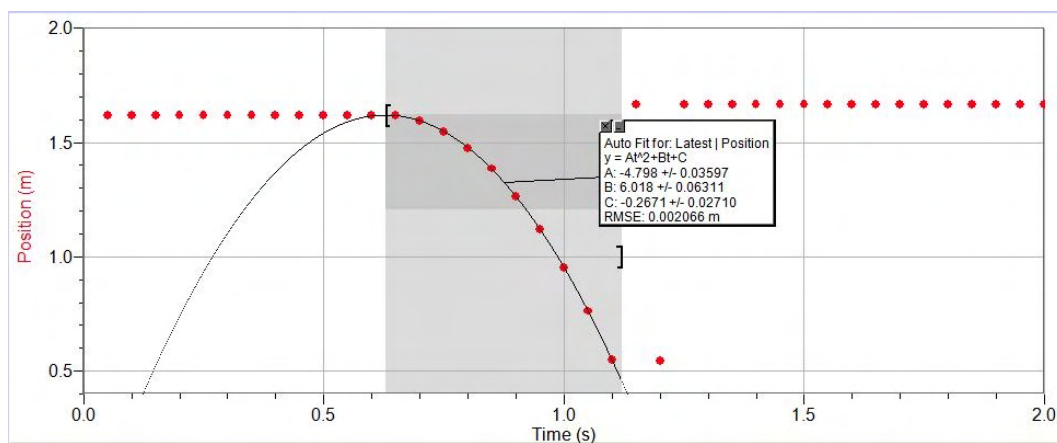


Figure 6: Raw position vs. time (y vs. t) data for one experimental trial. The region corresponding to the ball drop is highlighted in gray, and a best-fit quadratic curve is included.

2. Ensure that the ball falls directly onto the ultrasonic sensor by releasing it using the trigger on the hand pump. Adjust the position of the ultrasonic sensor if necessary.

3. Open the *LoggerPro* template file *free_fall.cmbl* from the “LoggerPro Templates” folder. This will open *LoggerPro* and set it up to take 3 seconds of position vs. time data at 20 samples per second.
4. Begin taking data by clicking the green *Collect* button in *LoggerPro*. As soon as the data points begin to plot across the screen, pull the trigger to release the ball. Make sure you are recording a complete drop.
5. If you are satisfied with how your data look, fit a quadratic function to the position vs. time graph (See Appendix A Section 1.4). Copy and paste the graph (including the fit line) into your lab report.
6. Drop the ball three more times. For each trial, record position vs. time data for the fall, perform the same fitting procedure, and paste the graphs into your lab report.

Problem 1.8 How can you measure gravitational acceleration (g) using the fit constants (A, B, C) that *LoggerPro* calculates for a quadratic fit to the position vs. time graph of a falling object?

7. Using your method from Problem 1.8, calculate the measured value of g for each of your trials.
8. Record your four values for g in table provided in the Jupyter notebook for this lab and using the online data collection tool (see Appendix A Sections 2.1-2.3). Note that g is technically the *magnitude* of gravitational acceleration and should be a *positive* quantity.

Experiment 2: Statistical Analysis

We will now analyze your group’s data and the class data statistically to determine which data set provides the better measurement of g , whose true value (we happen to know in advance) is 9.81 m/s^2 . We will also analyze larger data sets which represent data taken by hundreds of different groups of students.

1. Open the *LoggerPro* template file *statistics - F21.cmbl*. You will see two blank plots (see Figure 7).
2. Record your group’s four values of g in the “Your Values” column, and record the class values of g (from the online data collection tool and including your four values) in the “Class Values” column. Two histograms should appear

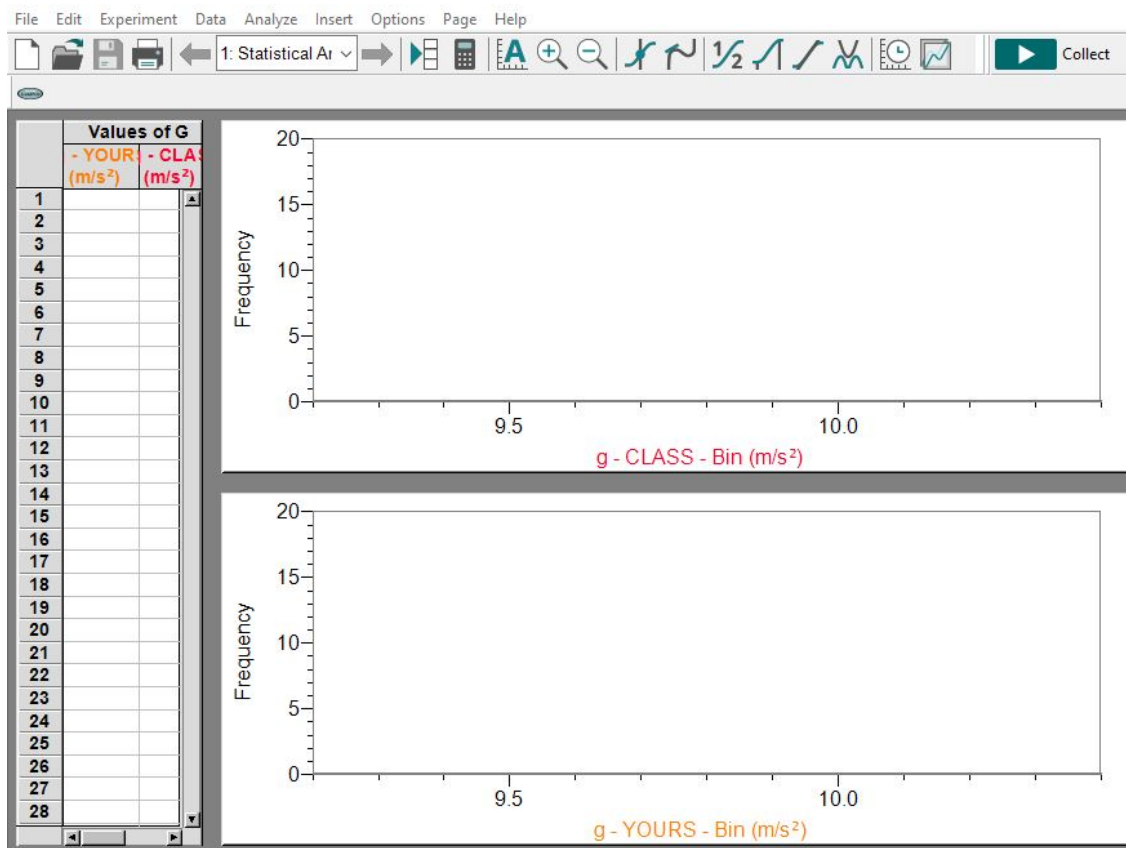


Figure 7: Screenshot of the statistical analysis template.

3. Add a statistics box to each histogram and paste all of the histograms into the appropriate spaces provided in your lab report (see Appendix A Section 1.1). In your lab report, use the information in the statistics box as well as the code provided in the Jupyter notebook for this lab to calculate the 95% confidence interval for the mean for each set of data.
4. You then will analyze a large data set collected from 600 groups within the past few years. Instead of using *LoggerPro*, we will utilize the power of Python. Run the code cell provided in this section of the lab to import the data, plot the histogram, and generate the statistics and 95% confidence interval for the mean of this large data set.

Problem 1.9 Look at the confidence intervals for your graphs. What confidence interval is the widest? Which is the narrowest? Why is this so? Your answer should show that you understand the two factors that affect the size of a confidence interval.

Problem 1.10 Which of the data sets above is the most precise? Which is the least precise? Which is the most accurate? Which is the least accurate? How do you know? Explain your answers thoroughly to each of these four questions.

Problem 1.11 The error of the mean is proportional to $1/\sqrt{N}$. Given this fact, what could explain a situation where the error of the mean for the class data is larger than the error of the mean for your group's data alone? Your explanation should refer to the concepts of statistical and systematic error.

Problem 1.12 If you did not know that the true value of g is 9.81 m/s^2 , could you determine how accurate your data set is? Why or why not? What implications does this have for experimental situations in which a researcher is measuring an unknown parameter?

Problem 1.13 Imagine instead of 600 groups we looked at 1000 groups. What advantage would we gain by having all that additional data? Remember that we are concerned with how close our mean is to the true (unknown) value.

Problem 1.14 What do you notice about the shapes of the histograms as more and more data points are added? What would the histogram look like if you had an infinite number of data points?

5. Using the code provided in the Jupyter notebook, add a Gaussian fit to the histogram of data from 600 groups. To get an appropriate fit, you will need to change the initial parameters of the fit until you succeed.
6. Calculate the **percent error** between this standard deviation from the fit and the standard deviation produced from the actual data set. The formula for percent error is

$$\% \text{ Error} = \frac{x - x_0}{|x_0|} \times 100, \quad (7)$$

where x is the measured or experimental value of a quantity (in this case the standard deviation of your data) and x_0 is the theoretical, expected, or standard accepted value of that same quantity. In our case, we theoretically expect to obtain a Gaussian, so the standard deviation of the Gaussian fit is our expected value.

Problem 1.15 In this experiment, you were asked to fit a Gaussian distribution to the largest data set. How does the standard deviation from the Gaussian fit compare to the standard deviation calculated directly from the data set? Does this seem consistent with what you expect from the Central Limit Theorem? Why or why not?

Laboratory 2

Computational Kinematics

Why Learn Numerical Modeling?

A ball in free-fall. A projectile subject to air drag. Gravitating bodies in our solar system. One can apply strictly analytical methods to the first of these physical situations with ease, to the second with great difficulty, and to the third only in special cases. When the benefit of the analytic approach grows small compared to the effort involved, physicists often utilize numerical methods to help achieve their research goals, as do researchers in many other disciplines.

Throughout this course, you will create computer models of the three situations listed above in computationally heavy labs. Creating these models will give you additional insight into constant acceleration, dissipative forces and the complexities inherent to central force problems that may not be able to be accessed within the confines of a traditional experiment or lab. Achieving this insight is, however, not the main purpose of these labs.

Common to simulating these and other physical situations with a computer is the process of model-building for numerical problem solving. It is our hope that completing these computationally heavy labs will equip you with the tools necessary to begin investigating the world around you with models, even when there are no straightforward analytical solutions.

As such, these labs do not assume that you have any prior knowledge in computer programming, and will cover only those syntax and programming techniques necessary to create efficient, readable, and physically accurate programs in Python.

Guide to Completing These Labs

You will find tasks located in boxes throughout each lab procedure. Follow these tasks in the order they appear. If you are unsure how to complete a task, read the surrounding text. Here are some tasks you should follow in general:

Box 1: General Tips

- ☐ Come prepared to these computational labs by having completed the pre-lab assignment associated with the lab.
- ☐ Ask your GSI or Learning Assistant for help sooner, rather than later. Interacting with them will help you learn the material covered in the lab (and, sometimes, more).
- ☐ You will be asked to complete both coding tasks and answer problems. In the Jupyter notebook, Coding Tasks will appear in red text and will require some code editing further than just inserting numbers where told. Problems will continue to appear in blue and require a written response.

Perhaps the most important benefit these labs could give you is the inspiration to explore. The world of computer simulation is vast and fascinating, and is by no means limited to physics.

Give these labs a chance to inspire you to not only become competent model-makers, but also producers, and not just consumers, of computer software.

Computational Kinematics: Pre-Lab Assignment

To receive full credit for this Pre-Lab Assignment: complete the Jupyter notebook file found in the Files tab of the Canvas page. Once completed, download the notebook as an html file and submit the assignment to Canvas before coming to class.

Welcome to the Pre-Lab Assignment for the Computational Kinematics Lab.

During the lab, you and your partner(s) will utilize computational methods to visualize functions and their derivatives, as well as learn to read and interpret code that perform numerical integration. This particular computationally intensive lab will give you the tools to model and predict physical phenomena related to kinematics. By investigating a computational approach to handling derivatives and integrals, we will be able to investigate further the relationship between position, velocity, and acceleration of a given system.

Box 2: In this Pre-Lab Assignment, you will:

- ☐ Define and Code the Kinematic Equations
- ☐ Create numerical position update and velocity update equations
- ☐ Understand the concept of Characteristic Time

Kinematic Definitions

The definition of the acceleration is the change in velocity with respect to time or $\vec{a} = d\vec{v}/dt$ while the definition of velocity is the change in displacement with respect to time $\vec{v} = d\vec{x}/dt$. These definitions are used when deriving a kinematic equation you may be familiar with under the assumption constant acceleration:

$$\vec{x}(t) = \vec{x}_0 + \vec{v}_0 t + \frac{1}{2} \vec{a} t^2. \quad (8)$$

This is done by integrating the acceleration over a time interval 0 to t twice. The first integral gives us the velocity $\left(\int_0^t d\vec{v} = \int_0^t \vec{a} \cdot dt'\right)$, while the second gives us the position $\left(\int_{x_0}^x d\vec{x}' = \int_0^t v(t') dt'\right)$. In this pre-lab assignment, we will code these two integral equations to find the position of a particle as a function of time, thus solving the kinematic equation numerically.

Part One: Position Update Equation with Constant Velocity

Suppose a particle moves in the $-\hat{y}$ direction starting at $y = 100$ m with a velocity $v_y = -10$ m/s.

Prelab Problem 2.1 With initial conditions $y = 100$ m and $v_y = -10$ m/s, how long will it take for the particle to reach $y = 0$ m? A code cell has been provided in the Jupyter notebook for calculations.

Let us confirm your answer to Prelab Problem 2.1 numerically. To do so, we will use the **position update equation** ($\vec{r}_{\text{new}} = \vec{r}_{\text{old}} + \vec{v} \cdot \Delta t$) which simply states that the position at a time t_{new} is equal to the old position at time t_{old} plus the change in position due to the velocity. This equation is essentially one step in the summation of an integral, we must perform it multiple times to do the entire integral. To to the full integral, we will use a while loop.

A while loop has the following construction:

```
while Condition:  
    Code Block
```

where Condition is a boolean variable (True or False) and Code Block is a set of code to be executed continuously while the condition Condition is True. To integrate, we want the condition to be True while we are integrating and False once we have summed all the pieces of the integral. In this case, we are starting at $y_0 = 100$ m and taking time steps of 1 s. We also have a constant velocity of -10 m/s in the \hat{y} direction.

Prelab Code Task 2.1 Input the initial conditions for the y position and y velocity. Then, run the cell to define the initial conditions.

Next we will set up the while loop to perform the integration. Since we are interested in knowing how long it takes for the particle to reach $y = 0$, we will keep track of time t . We will also define Δt , the time step size as `deltat`. When integrating, it is important to pick a good Δt or dt as it determines both how long and how accurate a numerical computation is. In this case, 1 second is sufficient but we will explore this more in part two.

```
#outside of while loop  
t=0  
#time step size  
deltat = 1 #seconds
```

and with each step of the sum (integral) we will add one unit of Δt to the time.

```
#inside while loop  
t = t + deltat
```

or, equivalently we can use the ‘+=’ python syntax, which redefines the left hand side to be its old value plus the right hand side.

```
#inside the loop  
t += deltat
```

We also need to consider the while loop 'Condition', we know it should take 10 seconds so we will let it run for up to 11 seconds and break the loop once $y \leq 0$. The code reads

```
while t <= 11:
    #code goes here

    #break statement
    if y<=0: break
```

Next we make sure to redefine y and \vec{r} both initially and at each step of the loop, so that the loop stops when $y = 0$:

```
#outside loop
r = r0

#inside loop
y = r[1] #since python is 0 indexed, 0 is x position, 1 is
        #y position and 2 is z position for the array r
```

finally, once the loop is complete we can print out the time it took for the particle to reach $y=0$ using the print function with the .format method.

```
print('Particle reached y={} after t={} seconds.'.format(y,t))
```

when we put this all together with the position update equation, we get a code that can perform an integration.

Prelab Code Task 2.2 Run the cell provided, without editing, to perform the integration.

Part Two: Characteristic Time

The characteristic time of a physical process is **not fixed**, but rather **an estimate** based on the physical properties of the problem. For example, if we consider the earth's orbit, it takes one year to complete an orbit therefore a good characteristic time would be 1 year. Another way to arrive at 1 year for an earth orbit would be to use the values of the problem's parameters in a combination

that has units of time. Force has units of $N = \text{kg} \cdot \text{m}/\text{s}^2$ which means $\text{s}^2 = (\text{kg} \cdot \text{m})/N$ so if we take the square root the product of the earth's mass (kg) and the distance from the earth to the sun (m), and divide by the force of gravity felt by earth (N) you would get seconds. The characteristic time for earth's orbit would then be estimated as $t_c = \sqrt{\frac{mr}{F_G}}$.

In a very similar manner, we can estimate the characteristic time for the process in part one under the influence of earth's gravity near the surface. Near the earth's surface $a = g = 9.81 \text{ m/s}^2$, and in the problem above $y_0 = 100 \text{ m}$.

Prelab Problem 2.2 Using the values of g and y_0 above, estimate the characteristic time for an object falling 100 m near the earth's surface where $g = 9.81 \text{ m/s}^2$. *Hint: What combination of g and y_0 has units of seconds?*

Part Three: Combining Velocity Update and Position Update Equations

Finally, we will include the influence of gravity on the object by using the acceleration due to gravity to update the velocity as the object falls. The **velocity update equation** is $\vec{v}_{\text{new}} = \vec{v}_{\text{old}} + \vec{a} \cdot \Delta t$. The acceleration due to gravity is $a_g = (0, -g, 0)$ where g is the magnitude of the gravitational acceleration. In this case the acceleration is simple and uniform, but in the future we will use the force with newton's second law to determine the acceleration for the velocity update, as $a = F/m$.

To include velocity updates, we will modify our previous code with the following. First we include the acceleration:

```
#define acceleration outside of loop. a is constant in this
#case so there is no need to update it.
g = 9.81
a = np.array([0, -g, 0])
```

Then we modify the while loop condition to instead use the characteristic time t_c as t_c

```
#define tc
tc = #combination of y0 and g.

#change while condition. use 1.2*tc to include the full motion
#in case we under estimate tc
while t < 1.2*tc:
```

We also modify Δt as it should be small compared to t_c i.e. $\Delta t/t_c \ll 1$. A systematic way to do this is to define another variable, the number of cycles as N_c or `Ncycles`, which determines the number of steps the computation is broken into. Here, $\Delta t = t_c/N_c$ and $N_c \gg 1$.

```
#define number of cycles
Ncycles = 10000
deltat = tc/Ncycles
```

Note that by changing `Ncycles` we change how accurate the computation is since it will break the problem into smaller steps, better approximating the infinitesimal steps dt in an integral. You might be tempted to change ‘`Ncycles`’ to a very large number such as 10^{10} but this can be counter productive as a larger number for `Ncycles` also means the computation takes longer. As you can see, `Ncycles` is a balance between computational time and accuracy.

Finally, we can no longer update the position based on the initial velocity, since the velocity will be changing due to gravitational acceleration. We will modify the position update to

```
r = r + v * deltat
```

and define the velocity vector to

```
#outside the loop
v = v0
```

Prelab Code Task 2.3 Fill in the gaps in the code cell provided in the Jupyter notebook to perform the computation of an object falling 100 m under the influence of gravity.

You should find that the particle reaches $y = 0$ after about 3.6 seconds.

Congratulations on completing this Pre-Lab Assignment; feel free to reward yourself with a tasty snack. You’ll complete the in-class portion both with your group and the guidance of your GSI. Be sure to submit the assignment to Canvas.

Computational Kinematics: Lab

Box 3: In this lab, you will:

- ☐ Emphasize the difference in scalars and vectors
- ☐ Implement Euler's Method of Integration
- ☐ Learn to solve systems with non-constant forces
- ☐ Complete a Culminating Assessment Task to solve for a ball bouncing on a trampoline

Part Zero: Notes on Scalars and Vectors

When coding it is important to keep in mind the data types you are working with since unexpected behavior will happen if you mix up your data types. It is very important in physics to be aware of which objects are **scalars** and which are **vectors**. Recall that a **scalar** is used to describe a **magnitude alone** and has only one numerical value. A **vector** on the other hand has **magnitude and direction** and when we work in 3D space, vectors are described by three values, typically an x magnitude, y magnitude and z magnitude.

Here are some examples of a scalar and vectors written in python

```
#scalar
g = 9.81 # m/s
```

It is always good practice to include units.

```
#vector 1
vec1 = np.array([1,1,0]) # (x,y,z)
```

As you can see, we have used `np.array` to create an array of numbers which we can treat like a vector. Typically the first position is the x direction, the second is the y direction, and the third is the z direction. Sometimes the z direction will be omitted when working in 2D. Another way to write a vector is by separating it into a **unit vector** (vector with magnitude 1, describes the overall

direction of the vector alone) and the **magnitude**. The same vector above, $(1, 1, 0)$ has a magnitude $|\vec{v}| = \sqrt{1^2 + 1^2 + 0^2} = \sqrt{2}$ and direction $\hat{v} = (1/\sqrt{2}, 1/\sqrt{2}, 0)$. We have used the pipes to indicate the magnitude of a vector in $|\vec{v}|$ and the hat to indicate a unit vector in \hat{v} . In python, we could then rewrite `vec1` as:

```
#vector 2
vec2mag = np.sqrt(2)
vec2dir = np.array([1/np.sqrt(2), 1/np.sqrt(2), 0])
vec2 = vec2mag * vec2dir #magnitude times direction gives the
                        #full vector
```

While this approach may seem like more work, it is often advantageous in situations where the magnitude is not changing but the direction is, e.g. if something rotates at a constant rate ω , the unit vector could read $\hat{u} = (\sin(\omega t), \cos(\omega t), 0)$.

When working on physical problem it is important to always be aware of which values are vectors and which are scalars, for example we can add two vectors by adding their components. If $\vec{v}_1 = (0, 1, 2)$ and $\vec{v}_2 = (1, 1, 1)$ then $\vec{u} = \vec{v}_1 + \vec{v}_2 = (1, 2, 3)$. Note that it does *not* make sense to add a vector and a scalar as $5 + (1, 1, 1)$ has no meaning. If you try to do this in python you will not be met with an error, but it will actually add 5 to each entry of the array,

```
np.array(1, 1, 1) + 5
```

which will return

```
([6, 6, 6])
```

Which could lead to an incorrect result in the physical process of interest since the expression $5 + \vec{v}$ has no meaning.

Part One: Euler's Method

Euler's method is a method to numerically compute integrals and derivatives. Numerical computation methods are used when analytic solutions do not exist. An analytic solution is one that we can algebraically solve for a solution. There exist many non-analytic problems, of which scientists and engineers want to know the solutions to. One of the methods used to determine solutions to

non-analytic problems is called Euler's Method. Many times in these labs we will still be working with functions that have known analytic solutions, and comparing to Euler's Method. By doing this with solutions we know, we can verify that our solution method will still work for problems we don't have known solutions to.

In this lab, we will be using Euler's Method to investigate the numerical relationship between acceleration, velocity, and position.

Problem 2.1 If you are given an initial velocity, an acceleration, and a time, how would you analytically (algebraically) calculate the final velocity?

In lecture, you will see that the acceleration due to gravity, g or a_g , can be written as $g = 9.81 \text{ m/s}$. This is constant in time.

The kinematic equation, $v_f = v_i + a \cdot t$ can also be written as $\Delta v = a \cdot \Delta t$, or the change in velocity due to an acceleration over a given time. For smaller increments of Δ , this becomes a small step in either velocity or time to give $\frac{dV}{dt} = a$. The computer can numerically calculate this relationship once, or many times moving by some Δt each time.

First, we need to tell the computer our initial conditions. These will be something like: A ball is falling towards the earth, and accelerating at a_g . The ball started from rest at some height h above the ground.

In lecture, you would apply the kinematic equation for one dimension (See Eq. 8) to solve this problem analytically.

Problem 2.2 Using kinematic equations, calculate the height an object will be at after time $T = 5$ seconds, if it started from rest at a height $h = 500 \text{ m}$.

Now lets use Euler's Method to numerically calculate this for us, and compare the results.

We will start with a given acceleration. We want to calculate $v_f = v_i + at$. We will start with the ball at rest falling for a given time. We will look at what the velocity is after 0.1 seconds.

Code Task 2.1 Run the code cell provided in the Jupyter notebook. Feel free to change the variables and run as many times as you need to understand what is going on.

*# Select this cell and press shift-enter to run it.
The output will be displayed in a new cell below this one.*

```
a = -9.81;
v = 0;
r = 500;
dt = 0.1;

v = v + a*dt

print('New velocity is: ', v, 'm/s')
```

Problem 2.3 Describe what this code cell does.

Code Task 2.2 Edit the provided code cell to calculate position.

This method works great for one time step, but we could easily calculate this by hand. It can be quite tedious when you want to look at more than a few time steps. We will utilize a for loop (see <https://wiki.python.org/moin/ForLoop> for information on for loops) to do these calculations for us. A for loop are similar to the while loop discussed in the pre-lab, but instead of repeating a code block and ending once a condition is false, a for loop iterates through a list or array and repeats a code block until the entire list has been iterated through.

To look at some total time T in increments of dt , we will create a time array. This array is a list of all the times at which we will query the velocity and position. This array is a list of all the times we will query the velocity and position at. Next we will use the for loop to iterate through this array we denote as `times`.

```
dt = 0.1

T = 5
times = np.arange(0, T+dt, dt)

for t in times[1:]:
    code block
```


Here we iterate through the array times (`[1:]` means we only index after the first element of the array) and update the variable `t` each time the code block is ran.

Code Task 2.3 Run the code cell provided after copying the position statement you added in Code Task 2.2.

Problem 2.4 If we wanted to plot the code above, what issues would we encounter?

In the next provided code cell, we have included syntax to plot the acceleration, velocity, and position over time.

```
#Syntax for plotting below
fig1, axs1 = plt.subplots(3,1)

axs1[0].plot(times, a_array[:,], '.')
axs1[0].set_title('Y acceleration')
axs1[0].set_xlabel('Time (s)')
axs1[0].set_ylabel('Acceleration (m/s^2)')
axs1[1].plot(times, v_array[:,], '.')
axs1[1].set_title('Y velocity')
axs1[1].set_xlabel('Time (s)')
axs1[1].set_ylabel('Velocity (m/s)')
axs1[2].plot(times, r_array[:,], '.')
axs1[2].set_title('Y position')
axs1[2].set_xlabel('Time (s)')
axs1[2].set_ylabel('Position (m/s)')

fig1.tight_layout()
```

Code Task 2.4 Run the code cell provided in the Jupyter notebook after copying the position statement you added in Code Task 2.2.

Problem 2.5 Discuss the difference in the code between Code Task 2.3 and 2.4, specifically the difference above the plotting syntax in Code task 2.4.

Since we live in a three dimensional world, we will acknowledge that our acceleration, velocity, and position can move independently in either of those three directions. Since right now we are

only interested in a ball falling, this is a one dimensional motion. We will just leave the other dimensions as zeros for now (to be continued in later labs).

```
a = np.array([0., -9.81, 0.])
v = np.array([0., 0., 0.])
r = np.array([0., 500., 0.])
```

The code above indicates that there are three directions (x , y , and z) to the acceleration, velocity, and position. This is why we choose to use r for position rather than x , since r is three dimensional, and x implies one dimension.

Code Task 2.5 Run the code cell provided to generate the same output as in Code Task 2.4 by utilizing three dimensional arrays. You will still have to copy the position statement you added in Code Task 2.2.

Now we are going to define a class. We will call this a ball, but we could call it a particle, box, ball, or even a dinosaur. Anything that will contain information that we determine. Here, we are defining a ball to have mass (m), position (r), velocity (v), and acceleration (a). To refer to the acceleration, rather than calling a we will call `ball.a`, or velocity will be `ball.v`.

We will not ask you to define a class yourself, but we do expect you to know (and be able to use) the various properties defined in the given class (m , r , v , and a).

Code Task 2.6 Run the code below without editing to define a ball class.

```
#Run this cell to define your ball class!
class ball:
    def __init__(self, m=1, r=np.array([0,0,0]),
                 v=np.array([0,0,0]), a=np.array([0,0,0])):

        self.m = m
        self.r = r
        self.v = v
        self.a = a

        self.r_array = np.array([])
        self.v_array = np.array([])
        self.a_array = np.array([])
```

Now using a ball as our falling object, we can use the exact same code from above with the ball syntax.

Code Task 2.7 Run the code cell provided after copying the position statement you added in Code Task 2.2. You will have to replace the variables used in earlier code cells with the new class object `ball_1` | .

Problem 2.6 Discuss the figure above. What do you notice about the relationship between acceleration, velocity, and position? *Hint: Are there (differential) equations which relate a , v , and r ?*

The plots created by the previous code cells will be static plots of acceleration, velocity, and position changing through time. We can, however, also make an animation of this ball traveling through time by plotting its position in time. The following code cells provided in the Jupyter notebook require all the code you needed to animate the graphs.

Code Task 2.8 Run the code provided to animate the ball falling, that you numerically calculated from the code cells in the Jupyter notebook found previously. Run the code in the both the provided cells without editing.

```
def init():
    line.set_data([], [])
    return(line)

def animate_ball_1(i):
    x = ball_1.r_array[i, 0]
    y = ball_1.r_array[i, 1]
    line.set_data(x, y)
    return (line)

def animate_ball_2(i):
    x = ball_2.r_array[i, 0]
    y = ball_2.r_array[i, 1]
    line.set_data(x, y)
    return (line)

def animate_ball_3(i):
    x = ball_3.r_array[i, 0]
    y = ball_3.r_array[i, 1]
    line.set_data(x, y)
    return (line)

def animate_ball_4(i):
    x = ball_4.r_array[i, 0]
    y = ball_4.r_array[i, 1]
    line.set_data(x, y)
    return (line)
```

```

fig2, axs2 = plt.subplots()
if np.min(ball_1.r_array[:,0])==np.max(ball_1.r_array[:,0]):
    axs2.set_xlim((np.min(ball_1.r_array[:,0])-0.05),
                  (np.max(ball_1.r_array[:,0]) + 0.05))
else:
    axs2.set_xlim((np.min(ball_1.r_array[:,0]) - \
0.1*np.abs(np.min(ball_1.r_array[:,0])),
                  np.max(ball_1.r_array[:,0]) + \
0.1*np.abs(np.max(ball_1.r_array[:,0]))))
axs2.set_ylim((np.min(ball_1.r_array[:,1]) - \
0.1*np.abs(np.min(ball_1.r_array[:,1])),
                  np.max(ball_1.r_array[:,1]) + \
0.1*np.abs(np.max(ball_1.r_array[:,1]))))
axs2.plot([np.min(ball_1.r_array[:,0]),
           np.max(ball_1.r_array[:,0])], [0,0], color='C1', lw=3)
line, = axs2.plot([], [], '.', markersize=24)
plt.close()

anim_1 = animation.FuncAnimation(fig2, animate_ball_1,
                                init_func=init, frames=N,
                                interval=30)

anim_1

```

Box 4: Tip for Controlling Animation

Once the animation has ran so that you understand what is going on, you can pause it by hovering your mouse over the figure and clicking the pause button.

It will take the computer a little bit of time to put the animation video together, so be patient. This specific animation shouldn't take too long, but future animations later in the semester will take a little bit of time to be generated.

Problem 2.7 Discuss how a change in the magnitude of the acceleration, initial velocity, and initial position each would be reflected in the animation above.

Earlier, you were asked to analytically calculate the position of the ball after a given time. Now we will compare this analytic result to our numerical result. We will only be looking at the ball's position for this.

Code Task 2.9 First run the code provided after copying the position statement you added in Code Task 2.2. Discuss with you lab partner(s) the difference between the analytical and numerical calculations of your position. Then, edit the times step, dt , in your code and run it again.

Problem 2.8 How does the step size dt affect the difference between the analytical and numerical calculation? What are the downsides to having very large and very small dt values?

Part Two: Bouncing Ball

In Part One, we used a constant acceleration. Now, since we know that $\vec{F} = m \cdot \vec{a}$ gives a relationship between the net force and the acceleration to a system, if the force is changing, the acceleration will also change. We will do this by defining a force, like we did below. The force here is a force due to gravity, $F_g = m \cdot g$.

Code Task 2.10 Run the code provided without editing to define a function, specifically the force due to gravity.

```
g = 9.81 # m/s^2

def force(m): #The variables you will give (variables you
              #pass) to your force calculation are in the
              #parentheses
    '''
    This force is dependent only on the y-coordinate,
    and represents gravity!
     $F_y = -mg$ 
    '''
    output_force = np.zeros(3) # initialize the returned force
                                # array with zeros
    output_force[1] = -m * g

    return output_force
```

Problem 2.9 After you read through the following code cell, describe what is being modeled. What does the `if` statement do, and why do you think the second part of the `and` statement is there?

Code Task 2.11 Run this code cell without editing.

```
ball_2 = ball(m=1, r=np.array([0,1,0]))

T = 3
dt = 0.01
times = np.arange(0, T+dt, dt)

N = times.size

ball_2.a_array = np.empty((N,3))
ball_2.a_array[0] = ball_2.a
ball_2.v_array = np.empty((N,3))
ball_2.v_array[0] = ball_2.v
ball_2.r_array = np.empty((N,3))
ball_2.r_array[0] = ball_2.r

i = 1
for t in times[1:]:

    # first update the accelerations
    ball_2.a = force(ball_2.m)/ball_2.m
    # then append the new value to the list
    ball_2.a_array[i] = ball_2.a
```

```

# if the ball is "in the floor", neglect gravity briefly
#in the calculation of v
# this keeps the ball from "sinking" into the ground over
#time
if ball_2.r[1] <= 0: ball_2.a = 0

# second update the velocities
ball_2.v = ball_2.v + ball_2.a * dt
# then append the new value to the list
ball_2.v_array[i] = ball_2.v

# third update the positions
ball_2.r = ball_2.r + ball_2.v * dt
# then append the new value to the list
ball_2.r_array[i] = ball_2.r

# if the ball is below y = 0 and traveling downwards,
#reverse direction of velocity
coeff_res = 0.7
if ball_2.r[1] <= 0 and ball_2.v[1] <= 0:
    ball_2.v[1] = -1 * coeff_res * ball_2.v[1]

# update iteration count
i = i + 1

```

The positive coefficient that multiplies the velocity is called the "coefficient of restitution." It takes a value between 0 and 1 (this has been set to 0.7 for you initially).

After running the code above and determining what is being modeled, run the plotting syntax below to plot.

Code Task 2.12 Run the code provided without editing.


```

fig1, axs1 = plt.subplots(3,1)

axs1[0].plot(times[1:], ball_2.a_array[1:,1], '.')
axs1[0].set_title('Y acceleration')
axs1[0].set_xlabel('Time (s)')
axs1[0].set_ylabel('Acceleration (m/s^2)')
axs1[1].plot(times[1:], ball_2.v_array[1:,1], '.')
axs1[1].set_title('Y velocity')
axs1[1].set_xlabel('Time (s)')
axs1[1].set_ylabel('Velocity (m/s)')
axs1[2].plot(times[1:], ball_2.r_array[1:,1], '.')
axs1[2].set_title('Y position')
axs1[2].set_xlabel('Time (s)')
axs1[2].set_ylabel('Position (m/s)')

```

Problem 2.10 Run the two blocks above a few more times, changing the coefficient of restitution a few times. What does a high coefficient correspond to? What about a low one? Give a few examples of objects with a high or low coefficient of restitution.

Code Task 2.13 Run the code provided in the Jupyter notebook to animate the ball bouncing you numerically calculated from above.

Problem 2.11 How does the animation from Code Task 2.13 represent the static figures you created in Code Task 2.12? Your answer should relate the features of the plots to behavior in the animation.

Part Three: Changing Forces

In Part Two, we determined our acceleration from a given force, $F_g = m \cdot g$, which was constant. Now we will explore systems where the force is dependant on some variable (we can determine our force to be dependent on anything we want, v , r , m , or even something else entirely!)

Code Task 2.14 Edit the code provided to have a force dependent both on position (r) **and** velocity (v). Try modeling the equation $F = -25r - 2v$ to start. Then run the code.

Problem 2.12 After you read through the code presented after Code Task 2.15, describe what is being modeled, given your updated force definition from Code Task 2.14. Predict and describe what your acceleration, velocity, and position graphs will look like due to this force.

Code Task 2.15 In the code cell provided, update the force to match the force you defined in Code Task 2.14. Then run the code.

Code Task 2.16 Run the next code cell provided in the Jupyter notebook to plot the acceleration, velocity, and position as a function of time for the simulation performed in Code Task 2.15.

Problem 2.13 Based on your prediction for Problem 2.12, was your prediction accurate? If not, what happened that you weren't expecting?

Problem 2.14 What was the importance of the velocity dependence in the force? Explain the changes you made, and the effect they had on the system.

Code Task 2.17 Run the code provided in the Jupyter notebook to animate the ball bouncing you numerically calculated for in Code Task 2.15.

Problem 2.15 Discuss the animation generated in Code Task 2.17. Does the animation represent the static figures you created in Code Task 2.16? What else do you notice?

Culminating Assessment Task (CAT)

In this section, you will combine the skills you used above to model a ball bouncing on a trampoline. A trampoline is an object that, when the ball is below the $y=0$ line, exerts a force proportional and opposite to the displacement. You can choose the coefficient in front of this factor. The ball also experiences the gravitational force at all times.

Problem 2.16 Write below how you plan to model a trampoline. (There are multiple ways to do this.)

Code Task 2.18 Edit the three code blocks provided in the Jupyter notebook to model the force(s) you want to define for a trampoline, as described above.

Box 5: Hints for Code Task 2.18.

- ☐ Look back at Parts 2 and 3 for inspiration.
- ☐ The code cells provided in this section are the same as used in Part Two, but you will need to update the force to represent a changing force as done in Part Three for the trampoline.
- ☐ After updating the force, be sure to edit the force update equation within the for loop so it matches how you defined it.

Problem 2.17 Does the ball act how you would expect? Look particularly at the plots of acceleration and velocity, and describe two distinct “regimes” of the motion.

Problem 2.18 Run the code blocks for Code Task 2.18 with different values for the constant in front of the position in the force function. How do different numbers change the different regimes? What physical changes to the trampoline does this represent?

Code Task 2.19 Run the code cell provided to animate the ball bouncing on a trampoline you numerically calculated in Code Task 2.18.

Problem 2.19 Discuss the animation above. Does the animation represent the static figures you created in Code Task 2.18? What else do you notice?

Laboratory 3

Motion with Uniform Acceleration

Introduction

In this laboratory, we will examine two different situations where the acceleration of an object is constant. The first will be a glider on an inclined plane; the second will be a glider attached to a hanging mass. The goal of this experiment is not to teach you any radically new physics; rather, it is to give you a chance to practice statistical analysis and to develop your ability to analyze various errors and their effects on your measurements. In each experiment, consider what factors will cause the magnitude of the acceleration to increase or decrease. Pay attention to the similarities in position vs. time graphs of uniformly-accelerating objects, no matter what the source of the acceleration is. Almost every problem you analyze in introductory mechanics will involve objects just like these.

Theoretical Background: Motion with Constant Acceleration

Much of the physics you will encounter in lecture will involve motion with uniform, or constant, acceleration. This means that the acceleration an object experiences does not depend on the object's position or velocity; it is just a single vector with constant magnitude and direction. Later on, you will learn more about situations that generate conditions of constant acceleration. There are four equations (shown in Table 1) that completely describe one-dimensional motion with constant acceleration. The quantities that appear in these equations are

x_0	Initial position
x	Position at time t
v_0	Object's velocity at x_0
v	Object's velocity at x
a	Acceleration (a constant)
t	Time it takes object to go from x_0 to x .

Equation	Variable You Can Ignore
$v = v_0 + a t$	x
$x = x_0 + v_0 t + \frac{1}{2} a t^2$	v
$v^2 = v_0^2 + 2 a (x - x_0)$	t
$x = x_0 + \frac{1}{2} (v_0 + v) t$	a

Table 1: Four equations relating key quantities for motion with constant acceleration.

Problem 3.1 Suppose you have a series of measurements of position vs. time for a falling object. Which one of the four equations in Table 1 should you use to obtain an estimate of the object’s acceleration? What’s wrong with the others? What if you had a series of measurements of velocity vs. time?

Theoretical Background: Newton’s Laws

No mechanics laboratory manual would be complete without some discussion of Newton’s Laws, so we list them here for reference purposes. Newton’s Second Law forms the foundation for nearly everything you will do in lecture and lab for the remainder of the course, so spend some time appreciating it!

Newton’s First Law states that a body remains at rest or in uniform motion unless acted upon by a net external force. The term “uniform motion” means motion in a straight line with constant speed (the object’s velocity vector is constant in time).

Newton’s Second Law states that a body acted upon by a net force moves in such a manner that the time rate of change of momentum (the product $m\vec{v}$) is equal to the force. If the mass of the body remains *constant* while the force is acting on it, we can write

$$\vec{F} = m\vec{a} ,$$

which is the familiar form of Newton’s Second Law that we all know and love.

Newton’s Third Law states that if two bodies exert forces on each other, these forces are equal in magnitude and opposite in direction. Another way of saying this is that “for every action, there

is an equal and opposite reaction.” It is important to remember that the “action” and “reaction” forces act on *different* bodies; if they didn’t, nothing would ever accelerate!

Problem 3.2 If a horse pulls on a cart with a force F , the cart necessarily pulls back on the horse with the same amount of force. Therefore, how does the cart ever move?

Theoretical Background: Kinetic Friction and Static Friction

In this experiment, we will try to make the experiment as ideal as possible. That is we will try to minimize the effect of friction. However, the effect of friction cannot be perfectly eliminated. You will also consider the role of friction when analyzing your experimental results.

Notice there are two types of frictional forces. The force that the sliding object must overcome so that it can “break free” and start sliding is the force of **static friction**. The force of static friction is what prevents an object from starting to slide along the surface. Therefore, static friction’s direction is opposite and the magnitude is the same as the force that tries to make the object move. Once the object has started to slide, the force of **kinetic friction** plays a role, acting to oppose the sliding object’s motion (i.e. opposing the velocity direction). It is equal in magnitude to the coefficient of kinetic friction, μ_K , times the normal force, F_N , that the object experiences while in contact with that surface. Mathematically, this is written as

$$f_K = \mu_K F_N .$$

Experimental Background: Tilted Air Track

In this laboratory, you will use an air track that has been specifically designed to provide you with a nearly-frictionless, one-dimensional environment. A photograph of the apparatus is shown in Figure 1. The track is armed with an ultrasonic sensor of the same type used in Laboratory 1. This sensor will measure the position of the glider to within a few millimeters.

In the first experiment of this laboratory, you will tilt the air track by an angle ϕ and release the glider from the high end of the track. The glider will bounce off the bottom a few times, and you will use a graph of the position vs. time measurements for a single bounce to measure a , the acceleration of the glider, and compare it to theory. How will you do this?

Consider the diagram of the glider on the air track shown in Figure 2. There are two forces acting on the glider: gravity and contact force. Remember that the normal force and friction are the two *components* of a single contact force. We will ignore friction for the purpose of simplifying our calculations; as mentioned before, it is negligible for this experimental setup. The directions of these forces are shown in Figure 2.

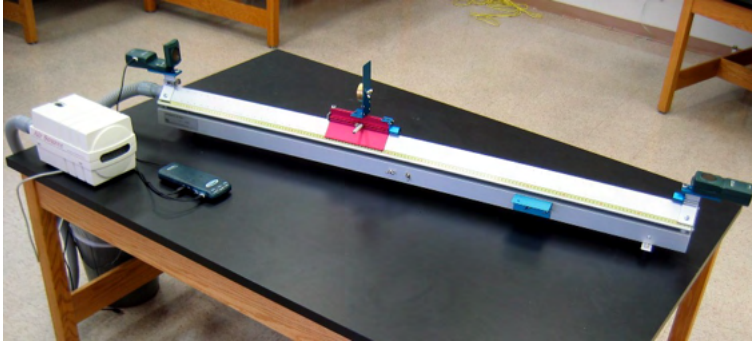


Figure 1: Photograph of the air track used in this and subsequent experiments. There are two sets of support screws on the underside of the track. The distance between these is used in calculating the tilt angle of the track.

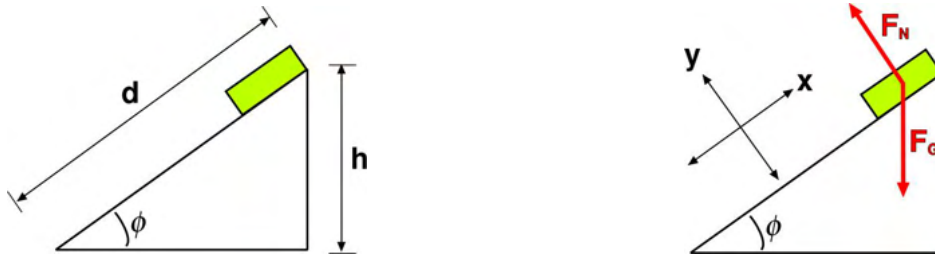


Figure 2: Diagram of a glider on an air track. The distance d between the support feet is 140 cm or 55.1 inches. The glider experiences two forces while sitting on the air track. The normal force is perpendicular to the surface of the track, and the gravitational force points straight down, toward the Earth.

We can break these forces into their **components** using the coordinate system shown in Figure 2. We can then write an equation relating force and acceleration in the x direction (the direction recorded by the ultrasonic sensor) using Newton's Second Law. We obtain

$$a_x = -\frac{g h}{d} \quad (9)$$

for the acceleration of the glider. This equation makes sense: if the track is flat ($h = 0$), the glider won't accelerate; if the track is vertical ($h = d$), the glider is in free fall.

Problem 3.3 Derive Eq. 9 on your own using Newton's Second Law.

In this laboratory, we will obtain a series of measurements of the glider's position x as a function of time t using the ultrasonic sensor. We will then fit these data to one of the equations from Table 1

to obtain an experimental estimate for a . To do this, we need an equation that relates position to time, acceleration, and other known quantities. We can use the second equation in Table 1,

$$x = x_0 + v_0 t + \frac{1}{2} a t^2, \quad (10)$$

where x is the glider's displacement from the sensor, which is at the lower end of the track, and x_0 and v_0 are the glider's initial displacement and velocity.

As you know from lecture, we can choose any point of the glider's motion as our starting point for the model fit. If you look at the raw data for this experiment that is displayed in Figure 6, you will see that the data is a series of parabolas, each representing the motion of one "bounce" of the glider off the bottom of the track and back. Because we will want to include as many data points as possible in our fit, we will fit one entire bounce to the model equation, Eq. 10. The time t will be the time since the point at which the glider first bounces off the bottom of the track. Note that if we use this definition and the coordinate system defined in Figure 2, the initial velocity of the glider (just after it bounces) will be positive, and the acceleration, appearing in Eq. 10, will be negative.

Problem 3.4 *LoggerPro* will fit your data to the functional form $y = A t^2 + B t + C$. What does each of these fit constants represent?

Experimental Background: Level Air Track with Pulley

In the first experiment of this laboratory, the force of gravity acts directly on the glider and produces a constant acceleration (see Eq. 9). For our second experiment, we investigate a different situation in which an object experiences a constant acceleration. We will use a level air track with a pulley attached to one end (see Figures 3-5). We will run a string from the glider over the pulley and attach it to a small mass m . If the glider has mass M , the acceleration of the glider will be given by

$$a_x = \frac{m}{M + m} g. \quad (11)$$

Problem 3.5 Derive Eq. 11 on your own.

Problem 3.6 How much hanging mass would there need to be for the glider to accelerate at g ?

Problem 3.7 In Experiment 1, what force(s) are responsible for the acceleration of the glider along the surface of the air track? What object exerts each force? Repeat this analysis for the force(s) in Experiment 2.

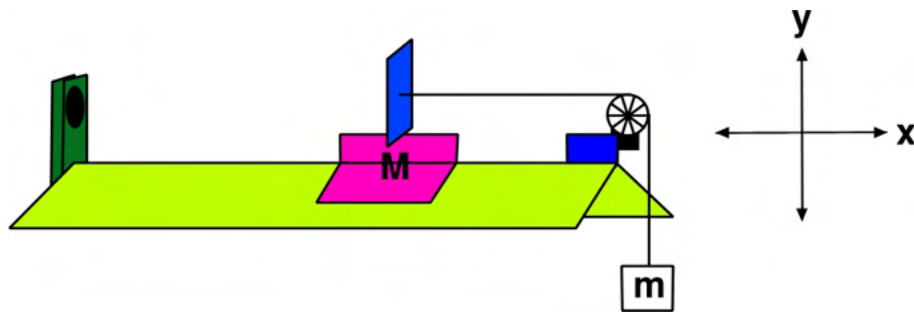


Figure 3: A schematic diagram of the glider on the air track with the pulley clamped onto the right side. The green object on the left is the ultrasonic sensor.

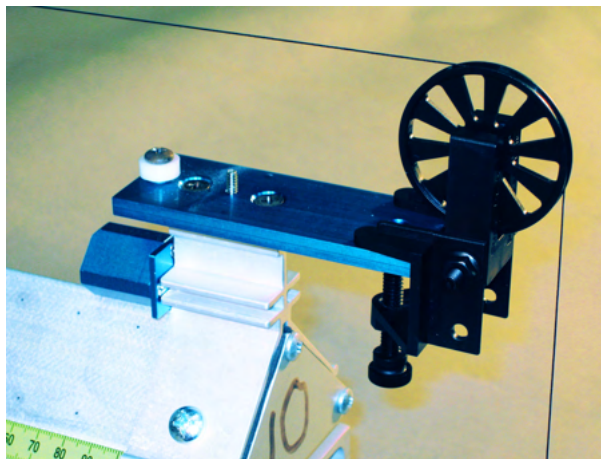


Figure 4: A close-up of the pulley mounted on the end of the air track.

Experiments

Experiment 1: Tilted Air Track

In this experiment, we will fit our data for x and t using *LoggerPro* and then determine an experimental value for the acceleration of a glider on a tilted air track. We will compare this acceleration to the theoretical value given by Eq. 9. Before you start, notice that leaving the glider on the air track while the air compressor is turned off can seriously damage the air track. Make sure the glider is off the air track when the air compressor is turned off.

1. Turn on the air compressor and place the glider on the track.
2. Level the air track so that the glider has no tendency to move left or right. There is a leveling

screw on the underside of the air track at one end.

3. Tilt the air track by sliding a 1-inch-thick block (or two 1/2-inch blocks) under the side opposite the ultrasonic sensor. The glider should be oriented on the track such that the metal bumper on the glider faces the metal bumper beneath the ultrasonic sensor. Record the height of the actual block used where prompted in the Jupyter notebook.
4. Open the *LoggerPro* template file *uniform_v_a.cml*.
5. Bring the glider to the raised end of the track and hold it there.
6. Start the data acquisition and release the glider. Position vs. time data will start to plot across the screen. The data will look similar to that in Figure 6.
7. Fit a quadratic curve to the first complete glider bounce (see Appendix A Section 1.4). Paste the graph into your lab report.
8. Calculate the glider's acceleration using the fit constants from your position vs. time graph using the code cell provided in the Jupyter notebook. Also calculate a theoretical value for the acceleration using Eq. 9. Note that the distance between the leveling feet on the track is 140 cm or 55.1 inches.
9. Repeat this procedure with the end of the track raised by half an inch instead of one inch. Be sure to also record the height of the new block used as well as copy and paste your second plot into your lab report as well.

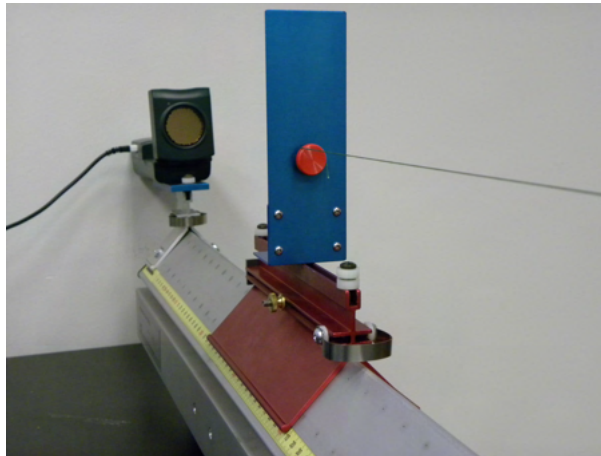


Figure 5: The string should be looped around the end of a screw attached to the glider's "fin".

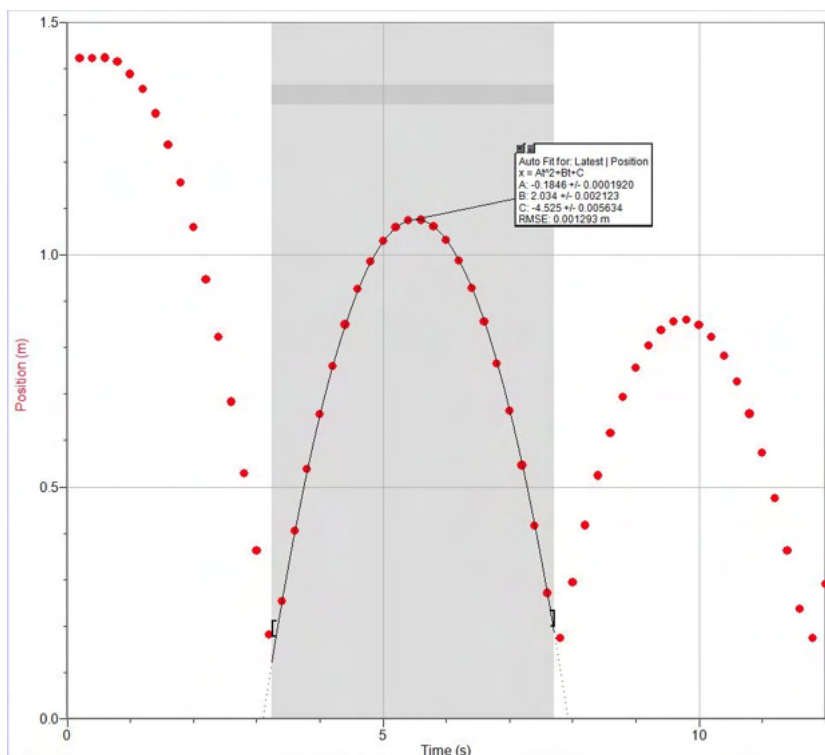


Figure 6: Raw position vs. time data for the tilted air track. A single complete bounce is selected for fitting, and a best-fit quadratic curve is included.

10. When you are done taking data, turn off the air compressor so you do not unnecessarily deafen your classmates.
11. Measure the mass of the glider. There are two scales in the lab.

Problem 3.8 Using the graph of the 1/2-inch tilt height data, describe what is happening during each part of a single glider bounce (a single parabola).

- (a) Specifically, what are the time intervals during which the glider is moving with a positive velocity? A negative velocity? Zero velocity?
- (b) What are the time intervals during which its acceleration is positive? Negative? Refer to specific (approximate) times from the graph.

Problem 3.9 How would the *magnitude* of your experimental measurement of acceleration change if you were to instead perform this experiment on the Moon? Would it be greater than, less than, or no different than the intended values described in the procedures? Write a one-sentence explanation (using physical reasoning rather than mathematical reasoning).

Problem 3.10 Try using *LoggerPro* to fit only the first half of the bounce. Record the acceleration determined from this fit.

Problem 3.11 Now fit only the second half of the bounce and record the acceleration determined from this new fit.

The two accelerations should differ as a result of friction. When the glider moves up the ramp, kinetic friction works with gravity to slow the glider down even faster, leading to a larger overall acceleration. When the glider moves down the track, friction now works against gravity. Kinetic friction always opposes motion, so when the glider starts moving in the same direction as gravity, friction will reduce the overall acceleration experienced from gravity. Using the free-body diagrams shown in Figure 7, we can determine the acceleration for each section:

$$a_x^{up} = -\frac{h}{d}g - \frac{f_k}{m}, \quad a_x^{down} = -\frac{h}{d}g + \frac{f_k}{m},$$

where $\frac{h}{d}$ is the ratio of the force of gravity along the x-axis.

Problem 3.12 Use the two accelerations from the fits in Problems 3.10 and 3.11 to determine the magnitude of the force due to kinetic friction acting on the glider. Is it okay we neglect friction earlier in this experiment? (Answer this based on whether friction is less than 1% of the component of gravity along the x-axis).

Problem 3.13 How would our results from this section (accelerations measured and proportion of friction to gravity) change if we used a larger mass? (*Hint: Think about what happens to friction as we increase the mass*).

Experiment 2: Level Air Track with Pulley

We now investigate a slightly different system where the glider experiences constant acceleration. The air track will be level, and a mass hanging from a string suspended over a pulley will pull the glider down the track with constant acceleration.

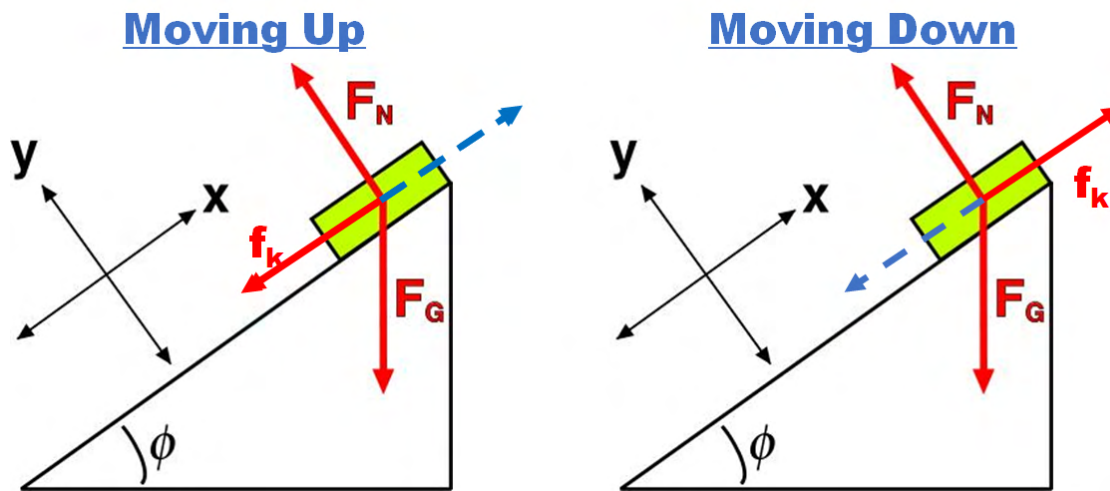


Figure 7: Diagram of a glider on the track with labeled forces. This diagram includes the force of friction, which varies direction depending on the motion of the glider. When moving up the track, friction points down in the $-x$ -direction. When moving down the track, friction points instead in the $+x$ -direction.

1. Set up the glider and air track as in Figure 3. In particular, remove all tilt blocks from beneath the air track, connect the glider to the small 10 g mass hanger via a length of string, and pass the string over the top of the pulley attached to the end of the track.
2. Turn on the air compressor. Move the glider toward the end of the track with the ultrasonic sensor, and hold it steady.
3. Begin taking data, and release the glider.
4. You will see a position vs. time graph similar to the one you made in Laboratory 1. Select and fit a quadratic curve to the region of the graph that corresponds to the glider moving *toward the pulley only* (before hitting the bumper beneath the pulley).
5. Paste the graph into your lab report.
6. Calculate the glider's acceleration using the fit constants from your position vs. time graph. Also calculate a theoretical value for the acceleration using Eq. 11. You can use the electronic scale on the center table to determine the true mass of your hanging mass.
7. Repeat this procedure using a 20 g hanging mass.
8. Record the *magnitude* of the glider's acceleration for all four trials (1 inch and 1/2 inch tilt heights, 10 g and 20 g hanging masses) using the online data collection tool (see Appendix A Sections 2.1-2.3).

Problem 3.14 How would the *magnitude* of your experimental measurement of acceleration change under the following conditions? Would it be greater than, less than, or no different than the intended values described in the procedures? Write a one-sentence explanation (using physical reasoning rather than mathematical reasoning) after each of your answers.

- (a) You performed the experiment on Jupiter (Hint: Is the gravitational acceleration larger or smaller than on the Earth?).
- (b) The glider's mass is greater.
- (c) You start the glider in the middle of the track rather than at the end.
- (d) The glider experiences more friction/air resistance as it moves toward the pulley.
- (e) The glider is slightly off from being on horizontal track (the track is not level).

Experiment 3: Model Fitting and Analysis

1. When all of the class data has been collected, follow the instructions provided in the Jupyter notebook to download and save the class data as a '.csv' file so that it can be imported into Jupyter notebook.
2. Once the data is imported into Jupyter notebook, use the provided code cell to calculate the mean, standard deviation, error of the mean, and 95% confidence interval for the mean of the class data for each of the four experimental setups.
3. Record these statistics in the table provided in your lab report and use it to answer the following questions.

Problem 3.15 For each of the four data sets: is the difference between the class mean and the theoretical value of acceleration “statistically significant” at the 95% confidence level (meaning that the 95% confidence interval for the class mean does *not* include the theoretical value)? What statistical conclusion can you draw from your result?

Problem 3.16 Which one of the four experimental methods used here appears to yield the most reproducible results? Justify your choice.

Laboratory 4

Projectile Motion

Introduction

In Laboratory 3, we looked at a situation where an object starting from rest was subject to a constant downwards force (and therefore had constant acceleration). The motion of the ball and the force of gravity happened to be *parallel*; gravity accelerated the ball downwards, but the ball had no horizontal velocity, and therefore did not stray from a one-dimensional line.

In many situations, things are not this simple. In projectile motion, which we will explore today, things get more complicated because the acceleration can be perpendicular to the object's velocity, parallel to it, or somewhere in between, depending on how the object is launched. This laboratory marks the start of our exploration of two-dimensional motion. In two-dimensional motion, the net force and velocity vectors lie in a plane but are not necessarily parallel to each other. This describes the majority of situations you will encounter in the rest of the course.

Theoretical Background: Two-Dimensional Motion

In Table 1 of Laboratory 3, we encountered four equations relating the variables x (displacement), v (velocity), a (acceleration), and t (time). Recall that each equation allows you to ignore one of these variables. To extend these equations to two-dimensional motion, we must simply apply the principle of **dimensional independence**, which means that forces, displacements, accelerations, etc. happening in one direction do not affect those happening in a perpendicular direction. Because of this very important principle, we can write two sets of equations, using subscripts to denote velocities and accelerations happening in the x and y directions (see Table 1 of this lab). The only variable that is the same for both dimensions is the time t ; we will exploit this fact repeatedly in our analyses of two-dimensional motion.

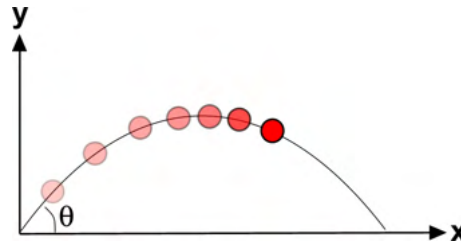
One cannot study two-dimensional motion effectively without understanding the ideas of **vector addition** and **component vectors**, which we already encountered briefly in Laboratory 3. A full discussion of these concepts can be found in any introductory physics textbook.

Parameter	x dimension	y dimension
Initial position	x_0	y_0
Position	x	y
Initial velocity	v_{0x}	v_{0y}
Final velocity	v_x	v_y
Acceleration	a_x	a_y
Time	t	t

Table 1: All of the variables for the x and y dimensions are distinct, except for time.

Theoretical Background: Projectile Motion

A **projectile** is any object propelled through space by a force that ceases after it is launched. While it is in flight, the only force that acts on it is gravity. Using the coordinate system shown in Figure 1, we can simplify the relevant x and y kinematic quantities listed in Table 1.



Parameter	x dimension	y dimension
Initial position	0	0
Position	x	y
Initial velocity	$v_0 \cos \theta$	$v_0 \sin \theta$
Final velocity	v_x	v_y
Acceleration	0	a_y
Time	t	t

Figure 1: The typical coordinate system used for analysis of projectile motion. Normally we think of a projectile as an object flying through the air. In that case, $a_y = -g$.

The angle θ above the horizontal is called the **launch angle**. The horizontal distance the projectile travels before it reaches the vertical height from which it was launched is called its **horizontal range**, denoted r_x . The maximum vertical displacement of the projectile from its starting position is called its **vertical range**, denoted r_y .

We can obtain a formula for the vertical range r_y as follows. Assume the projectile is launched with

initial coordinates $x_0 = 0$ and $y_0 = 0$ at a launch angle θ and an initial speed v_0 . The vertical range of the trajectory is then equal to the displacement y at the precise moment when the projectile is turning around. At this instant, it is motionless in the y direction; in other words, the y component of velocity equals zero. In the kinematic equation

$$v_y^2 = v_{0y}^2 + 2 a_y y ,$$

we can then substitute $y = r_y$ for $v_y = 0$ to obtain

$$\begin{aligned} r_y &= -\frac{v_{0y}^2}{2 a_y} \\ &= -\frac{v_0^2 \sin^2 \theta}{2 a_y} . \end{aligned} \tag{12}$$

This formula for the vertical range will be useful in lab.

We can obtain a formula for the horizontal range r_x as follows. Assume the projectile is launched with initial coordinates $x_0 = 0$ and $y_0 = 0$ at a launch angle θ and an initial speed v_0 . First, let us find the *time* t_{end} that the projectile returns to its initial height $y_0 = 0$, since the projectile's horizontal range r_x is simply the horizontal displacement x at this time. In the kinematic equation

$$y = y_0 + v_{0y} t + \frac{1}{2} a_y t^2 ,$$

we can substitute $y_0 = 0$ and $t = t_{\text{end}}$ for $y = 0$. One solution is $t = 0$ (reflecting the fact that the projectile did indeed *start* at initial height $y_0 = 0$), but the relevant solution is

$$t_{\text{end}} = \frac{-2 v_{0y}}{a_y} .$$

Since we are assuming that $a_x = 0$, the kinematic equation

$$x = x_0 + v_{0x} t + \frac{1}{2} a_x t^2$$

tells us that

$$\begin{aligned}
 r_x &= x(t_{\text{end}}) \\
 &= v_{0x} t_{\text{end}} \\
 &= -\frac{2 v_{0x} v_{0y}}{a_y}.
 \end{aligned}
 \tag{13}$$

This formula for the horizontal range will be useful in lab. Using trigonometric identity $\cos \theta \sin \theta = 1/2 \sin 2\theta$ this can be expressed alternatively as

$$r_x = -\frac{v_0^2 \sin 2\theta}{a_y}.
 \tag{14}$$

Experimental Background: The Air Table

To study two-dimensional motion, we will use an air table, which provides us with a two-dimensional version of the frictionless surface we used for our study of motion with constant acceleration in Laboratory 3. A diagram that includes the coordinate system we will use for this experiment is shown in Figure 2.

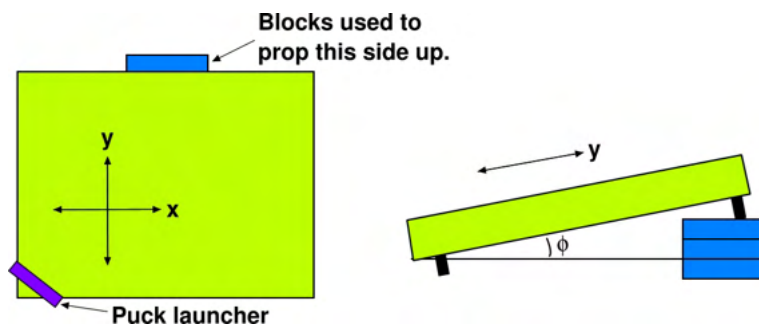


Figure 2: A diagram of the air table used in this laboratory. Air expelled from dozens of small holes in its surface creates a nearly-frictionless surface on which aluminum pucks can slide.

A puck launcher (see Figure 3) is attached to the air table; it allows you to launch aluminum pucks with fairly reproducible speed. Four sets of pegs across which a rubber band is stretched lead to different initial puck velocities.

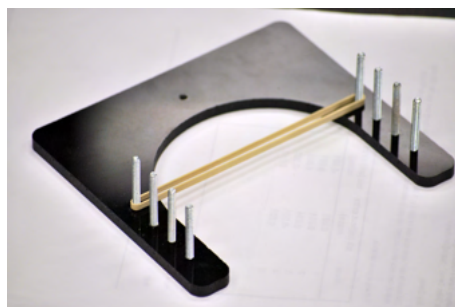


Figure 3: A photograph of the puck launcher, which is attached to the air table as shown in Figure 2.

There is one very important fact you must realize before studying projectile motion on an air table: the force in the y direction is no longer the object's weight, but rather the component of that weight which is parallel to the surface of the air table. If ϕ is the tilt angle of the air table (produced by a block placed under one of the leveling feet), the component of gravitational acceleration in the y direction becomes

$$a_y = -g \sin \phi \quad (15)$$

instead of $a_y = -g$, as we normally expect for projectiles in flight.

Problem 4.1 What happens to a_y when $\sin \phi = 0$? How about when $\sin \phi = 1$? What two situations do these correspond to?

Experimental Background: Video Capture

So far we have used an ultrasonic sensor to capture the motion of moving objects. Unfortunately, this technique does not work for two-dimensional motion, since the ultrasonic sensor can only capture motion in one direction. Therefore, we must use a different technique for this lab: video capture. Above your air table, you will notice a small webcam. This camera will record video at a set rate within *LoggerPro*. You can analyze your video by clicking on the object of interest in each frame; *LoggerPro* will automatically make graphs of position vs. time for both the x and y directions.

Experiments

Experiment 1: Projectile Motion

For our first experiment, we will experimentally verify certain important properties of projectile motion (for example, dimensional independence).

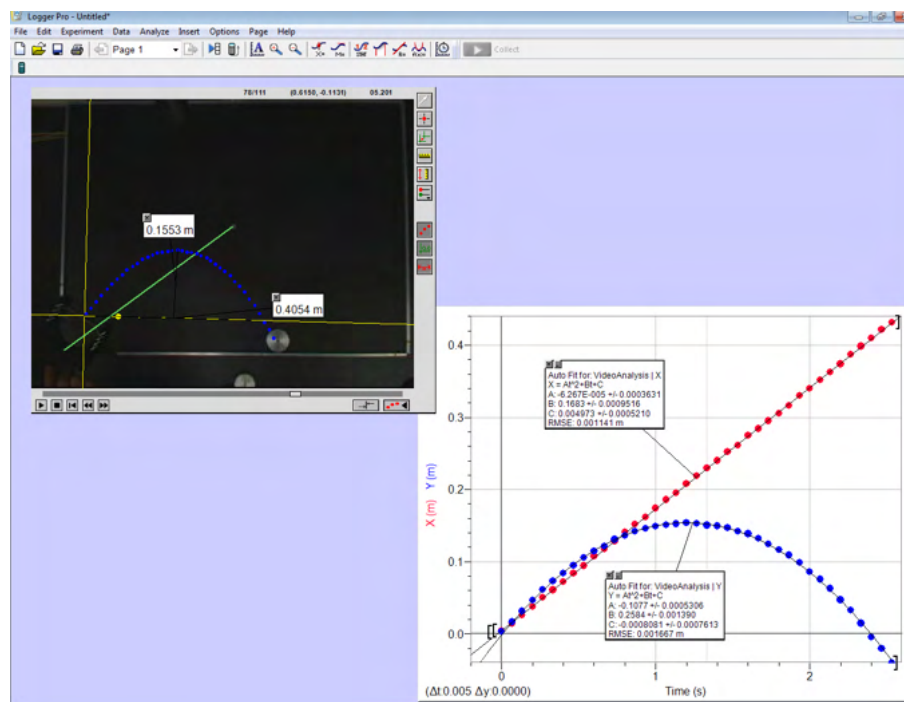












Figure 4: A screenshot of raw data from Laboratory 4.

1. There is no *LoggerPro* template for this laboratory. Open *LoggerPro* from desktop and insert a new video capture window (see Appendix A Section 1.8)¹. Also, the reflection from the table might not allow you to see the blue dot of the puck. If this is the case, click 'Options' and then go to 'Camera Settings' to change the brightness and contrast of the capture window.
2. Level the air table; then tilt up the end of the table with the camera attachment using a 1-inch-thick plastic block (or two 1/2-inch-thick blocks). Make sure the camera is aligned correctly. The camera can be adjustable while the camera is still clamped to the stand. In the

¹If the camera reports an error or glitches a lot, you need to check the camera resolution. Click 'Set Up Camera' and try changing the resolution.

video capture window, the edges of the air table should fit within, and approximately line up with, the edges of the camera's view.

3. Practice launching the puck so that the puck's trajectory forms a big parabola that terminates along the bottom rail of the air table. If the puck's trajectory does not terminate along the bottom rail, you will not be able to obtain an experimental horizontal range.
4. Ready the puck for launch and click *Start Capture* in the video capture window. Give *LoggerPro* a moment to begin recording video, and then launch the puck at an angle between 0 and 90 degrees. *LoggerPro* will record a video of the motion. (Caution: When pressing the Stop Capture button, sometimes the recording may fail with a Quicktime error. This happens mostly when you double click the stop button, or when clicking the button too fast after the previous click. This is an unfixable bug in LoggerPro.)
5. Obtain position vs. time data (for both x and y dimensions) by analyzing the video in *LoggerPro*. The following step-by-step procedures will help you.
 - (a) Click the  *Enable / Disable Video Analysis* button at the bottom right of the video capture window to display the video analysis toolbar.
 - (b) Use either the slider bar or the double-arrowed buttons   in the video capture window to queue the first frame where the puck is *no longer in contact with the rubber band*. The center of the puck in this frame will be its initial position.
 - (c) With this frame queued, click  *Set Origin*, and then click on the center dot of the puck.
 - (d) Set the time t equal to zero in this frame of the video by clicking the  *Synchronize Video* button and specifying the **Graph Time** to be 0. This step is crucial in order to get the correct initial velocities of the pucks. Do not set the Movie Time equal to 0. Also, make sure the 'Synchronize Video Analysis Data Set with Selected Data Set' box is checked.
 - (e) Set the size scale: Click the  *Set Scale* button; then click and drag between the air table's center dot and the screw head that fastens the puck launcher to the air table. This distance is **0.475 m**.
 - (f) Track the puck's coordinates: Click the  *Add Point* button, and then click once on the center dot of the puck (where the origin should be located). The video will advance one frame for each point you add in this way. Continue this until the puck comes into contact with one of the rails on the air table. You should now have two sets of data points, x and y position vs. t , and they will be displayed on the graph. (If you made a mistake and would like to delete a point, click Select Point  and click on the point. Then press delete.)

- (g) Fit *both* the x and y data sets to a quadratic equation.
- (h) Now we want to see how the position vs time graph looks like. Resize both the graph and the video capture window by grabbing one of the edges of the window with the mouse cursor so they can be viewed on the screen simultaneously. Return to the video capture window and click the  *Set Origin* button again. Click and hold the yellow dot located on the positive x -axis (near the origin) and slowly spin the axes around the origin. As you spin the axes, you'll notice that the graph and associated fit parameters update automatically to reflect the changing axes. Spin the axes until the x fit curve is as linear as possible (fit constant A should be close to zero). This adjustment should be *small* and is intended to correct for any camera alignment imperfections.
- (i) Obtain experimental horizontal and vertical ranges using the  *Photo Distance* button. Record these values in your lab report. Also, copy your graph (with fit curves) into your lab report.
6. Repeat steps 4-5 to do two more trials, for a total of three trials. Remember, it is important to make sure that your data includes a “complete” parabola; otherwise, you will not be able to obtain an experimental measurement of the horizontal range.
7. For your x and y fit equations for each trial, copy the fit parameters into the appropriate spaces in your lab report.
8. Record the puck's mass and the tilt block thickness in your lab report.

Problem 4.2 Calculate the sine of the tilt angle using the block thickness and the dimensions of the air table (already measured for you in the lab report). Using this sine angle, calculate the theoretical y acceleration using Eq. 15

Problem 4.3 Qualitatively, how would your position vs. time graphs for the x and y directions change, relative to normal experimental conditions, in the following scenarios? Assume you are launching the puck with exactly the same initial speed and launch angle each time.

- (a) The table is flat (not tilted).
- (b) The table is tilted upward by 1/2 inch instead of 1 inch.
- (c) The table is tilted along the x -axis and the y -axis, instead of just the y -axis.
- (d) You use a more massive puck.

Experiment 2: Model Fitting and Analysis

1. Using the results from your fits, fill in the data table provided in your lab report with experimental values for the initial x and y velocities, x and y accelerations, launch angle (as measured above the positive x -axis), vertical range, and horizontal range. In addition to the ranges you measured in Experiment 1, calculate theoretical ranges using the range equations derived in the Theoretical Background section (Eqs. 12 and 13) using the experimental values for v_x and v_y . You can save time creating a function with the inputs being the fit coefficients, but this is not required.
2. Choose what you consider to be the “best” trial, and recopy the graph into the space provided in your lab report directly above Problem 4.4 (this will make it easier to answer the problem).

Problem 4.4 For your chosen trial, is your experimental value for the y component of the acceleration greater or less (in magnitude) than the theoretical value you calculated using Eq. 15? Could the following sources of systematic error explain the deviation you observed? Justify each answer with a brief explanation.

- (a) The tilt block is thinner than 1 inch.
 - (b) The puck’s initial position was not coincident with the origin.
 - (c) The puck’s mass is greater than what was measured by the (perhaps miscalibrated) scale.
 - (d) We’re actually on TRAPPIST-1g, an earth-like exoplanet slightly more massive than earth.
3. First tilt the angle with a 1-inch-thick block. Without recording data, vary the launch angle, θ , and launch the puck several times. Try to maintain a constant initial velocity between launches (Do the best you can. You don’t need to be perfect.).

Problem 4.5 Which angle seems to give the highest vertical range? The longest horizontal range? Can you justify your guesses with the formulae for vertical and horizontal range (Hint: Use Eqs. 12 and 14 in the Theoretical Background section)?

Laboratory 5

Projectile Motion with Air Drag

Introduction

Welcome to the Projectile Motion with Air Drag Lab. In each of the preceding labs, we investigated objects subject to constant acceleration. In Lab 1, we looked at an object in free-fall, neglecting the effects of air drag. We created a model for this motion in Lab 2. In Lab 4 you investigated the projectile motion of a puck gliding across a tilted air table (referred to here as the “puck launch experiment”). For Lab 5, we will analyze motion where the acceleration is *not* uniform, which will demonstrate the power of Newton’s Second Law to predict behavior for more complicated systems. We will begin by creating a model of the two-dimensional projectile motion, then go on to incorporate air drag into the model for an object in free-fall from Lab 2 before finally applying it to our projectile model.

As we’ll learn in the Pre-Lab, the air drag force has variable direction and magnitude. Incorporating air drag into an analytical model entails fancy algebraic footwork. Modeling it numerically, on the other hand, is relatively straightforward.

Discuss any questions you have about the material on the Pre-Lab with your group or GSI at the beginning of Lab or someone in the Physics Help Room. The Pre-Lab is designed to assist your completion of the Lab within the time constraints presented by class time; effort you spend now to improve your understanding of these topics will strongly benefit your performance in class and your skill as a model-builder.

In class, we’ll assemble a model of the puck launch experiment initially assuming that the only forces acting on our projectile are the normal force from the table and gravity, then attempt to reproduce our results from the puck launch experiment, thereby determining whether it was reasonable for us to ignore air drag when analytically modeling the puck launch experiment. To make this comparison, we’ll need the experimental data from one of your puck launches, so make sure you are able to access your previous lab report from Canvas.

After getting our model up and running without air drag, we’ll incorporate the air drag physics from this pre-lab assignment into our program. This will enable us to investigate a physical situation with very different parameters, so that we can investigate how these parameters affect how strongly air drag affects the projectile’s trajectory.

Projectile Motion with Air Drag: Pre-Lab Assignment

To receive full credit for this Pre-Lab Assignment: complete the Jupyter notebook file found in the Files tab of the Canvas page. Once completed, download the notebook as an html file and submit the assignment to Canvas before coming to class.

In this Pre-Lab, we'll learn about the physics of a force with variable direction and magnitude: air drag. The translations from algebra to code performed here will prove crucial to your group's performance during Lab. If you aren't certain about any of the answers, discuss them with your group or GSI at the beginning of Lab or in the Physics Help Room.

We'll also revisit the concept of **characteristic time** at the end of this Pre-Lab. A better understanding of this concept will help you understand the models we develop in Lab, as well as aid you in developing models of your own.

Box 1: Pre-Lab Objectives

- ☐ Begin investigating the physics of motion with air drag.
- ☐ Translate crucial formulae and properties of the puck launch experiment from algebra to Python code.
- ☐ Review the concept of characteristic time and how it underpins numerical simulation.

Introduction to Air Drag

When an object moves through a fluid, such as a ball moving through air, it experiences a force opposing its motion. The general effect of such a **drag** force is to resist the body's motion and to slow it down. Despite differences in the detailed behavior of the many different models for the force of air drag, most share the following properties:

- The drag force is directed opposite the object's motion
- The drag force's magnitude increases with the object's speed
- The drag force's magnitude increases with fluid density and the object's cross-sectional area
- The drag force's magnitude also depends on details about the object's shape (a person and sphere with the same total cross-sectional area still experience different drag forces)

The exact form an air drag force model takes is an extremely interesting area of physics (for a wryly written example, check out <http://physics.info/drag/>). We will assume the following model for the drag force on a ball moving through air with velocity \vec{v} :

$$\vec{F}_{drag} = -\frac{1}{2}\rho AC|\vec{v}|^2\hat{v}. \quad (16)$$

Imagine a bowling ball moving through a smattering of ping pong balls. During each collision, the ping pong balls apply a small force to the bowling ball, oriented (on average) opposite to its direction of motion. The more often the bowling ball collides with a ping pong ball, and the more force applied during each collision, the more drag force it experiences. In the puck launch experiment, the bowling ball represents the puck, while the ping pong balls represent the stationary molecules of air the puck collides with as it moves along the table. With that analogy in mind, let's review the meaning of each component of this force law:

- The drag force is always oriented directly opposite the object's motion ($\vec{F} \parallel -\hat{v}$). Recall that \hat{v} is the **unit vector** which points in the direction of the object's instantaneous velocity. Mathematically, we write $\hat{v} = \frac{\vec{v}}{|\vec{v}|}$. The unit vector which points directly *opposite* the object's velocity is, therefore, $-\hat{v}$.
- ρ is the density of the fluid. If the fluid is more dense, the object will collide with more objects per unit distance traveled, and experience a higher drag force.
- The cross-sectional area, A , refers to the area of a three-dimensional shape when viewed from a certain direction. For example, a cube of side length l has cross-sectional area l^2 when viewed from directly above one of its faces. A sphere with radius r has $A = \pi r^2$ (the area of a circle with radius r) when viewing it from any direction.² The *larger* A is, the *more* air an object has to push aside per unit distance traveled through the fluid.
- The drag coefficient, C , is a constant which is characteristic of the shape of the moving object. For an ideal sphere, $C = \frac{1}{2}$. The curious may find appropriate values of C for other shapes at <http://physics.info/drag/>. For our simple model, let's pretend that someone, somewhere experimentally determined these values for us to use. Actually deriving them is much more involved and would require a more detailed model.
- The dependence of air drag force magnitude on speed we've assumed here is quadratic ($F \propto v^2$). (That symbol, \propto , is mathematical shorthand for "is proportional to.")

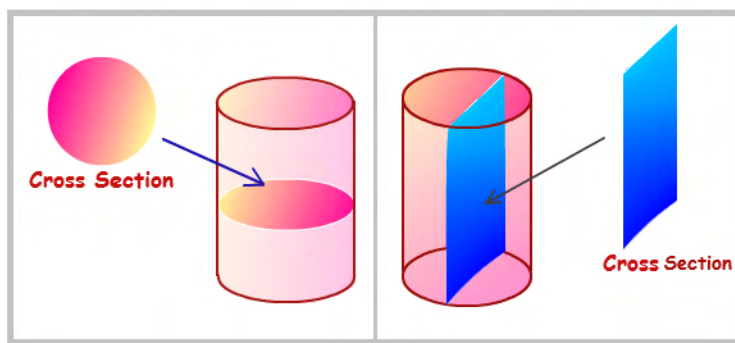
²Fritz Zwicky (1898-1974), the first physicist to postulate the existence of the still-mysterious Dark Matter, famously decried his critics as "spherical idiots" because they were idiots no matter which way he looked at them. The actual quote is slightly more colorful.

The relationship $F \propto v^2$ is generally appropriate for objects moving at medium speeds through dilute fluids (such as pucks on an air table, moving through air). Though this relationship can be more rigorously derived in various ways, the following heuristic might give you some intuition about what's going on physically. Returning to our bowling ball/ping-pong ball analogy, as v increases, the bowling ball will collide with more ping-pong balls per unit time. Not only that, the ping pong balls strike the bowling ball with more force during each collision. Thus it seems *reasonable* (but is by no means proven) that there are two multiplicative factors of v at work. Other possible choices (which we will not investigate) include $F \propto v$, $F \propto v^n$ (where n is an empirically-estimated parameter), or even a polynomial relationship $F \propto \sum_{i=1}^n a_i v^i$.

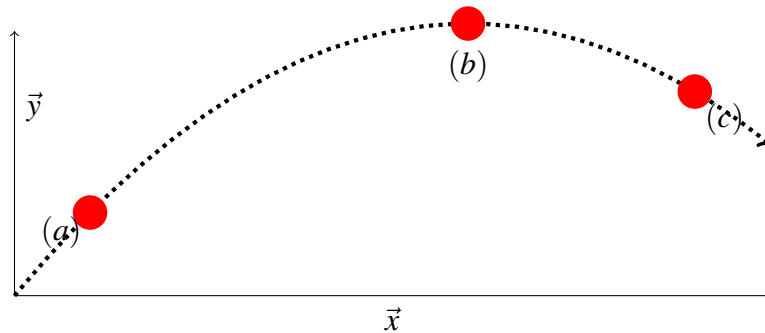
Drag force on a spherical object in projectile motion

Prelab Problem 5.1 Look up, and write here, a reasonable value for the density of air in the lab, expressed in units of kg/m^3 . (We'll need this value in various places during this lab.)

Prelab Problem 5.2 Consider a cylinder with height h and radius r . What is its cross sectional area when it is moving parallel to its axis (below, left)? What about when it is moving perpendicular to its axis (below, right)? If you aren't sure why these values are different, make sure to discuss this concept with your group or GSI during the lab.



Prelab Problem 5.3 Below is plotted the (near-parabolic) trajectory of a ball thrown near the Earth's surface. \hat{y} points vertically upwards, and \hat{x} points horizontally. At each labeled point, draw vectors representing \vec{v} , \vec{F}_g , and \vec{F}_{drag} : (a): on the way up after launch, (b) at the peak of its trajectory, and (c) on the way back down. Don't worry about the magnitude of the vectors, just the directions.



Prelab Problem 5.4 Will the magnitude of the drag force be larger at (a) or (b)?

Prelab Problem 5.4 should get you thinking about the magnitude of the drag force. Increasing our velocity, say when riding a bike downhill compared to walking on flat ground, drastically increases the magnitude of the drag force. One way to quantify this magnitude is to compare it to the magnitude of the force of gravity. Let's use the ball drop experiment as a framework for our imagination.

Prelab Problem 5.5 What is the speed of an initially stationary ball after falling 2 meters (the approximate height of the stand in the ball drop experiment) near Earth's surface and ignoring air drag?

Prelab Problem 5.6 Calculate the magnitude of the air drag force acting on a ball with this speed. You'll need to know the density of air in kg/m^3 (you wrote it in Prelab Problem 5.1), the value of A for a sphere with radius .025 m (similar to what you did in Prelab Problem 5.2, except for a sphere, not a cylinder), and the value of C for a sphere (0.5).

Prelab Problem 5.7 Calculate the ratio of this magnitude to the magnitude of the gravitational force acting on the ball (assume it has a mass of 75 g).

Prelab Problem 5.8 Based on the result to the previous problem, is it valid to ignore air resistance for this problem? Justify your answer.

Congratulations on completing this Pre-Lab assignment! Although you will not be required to submit the remainder of questions asked during the rest of this background section, it is important to read through and try to answer them on your own in preparation for the lab. There are many useful concepts discussed that relate to simulating air drag and the puck launch experiment in python. Hopefully, answering these questions now will enable you and your partner to efficiently assemble a working model in Lab, evaluate the importance of air drag in the ball drop and puck launch experiments, then explore how air drag could affect something you didn't do in Lab: the flight of a rocket in the Earth's atmosphere.

Calculating the Drag Force: from Algebra to Code

When attempting to determine whether air drag will play an important role in a given physical situation, always start with the air drag force equation:

$$\vec{F}_{drag} = -\frac{1}{2}\rho AC|\vec{v}|^2\hat{v}. \quad (17)$$

Notice in particular that the object's mass doesn't appear in this equation. Contrast with the force of gravity

$$\vec{F}_{grav} = -m\vec{g}, \quad (18)$$

which scales linearly with mass. For a given mass and velocity, having smaller air drag constants (that is, A and/or C) will reduce the importance of air drag, and vice versa. Suppose a rock and a parachute with the same mass have the same downward velocity at a given moment. They both experience the same gravitational force, but the parachute's larger area will result in a larger drag force, and will fall more slowly when unfurled.

Another way for air drag to become more important relative to other forces is by increasing velocity. It was reasonable for us to ignore air drag during our analysis of the ball drop experiment because the ball had a relatively small drag force compared to gravity (you'll do a calculation along these lines in Lab). If the ball had been moving faster, say as fast as a golf ball, air drag would have played a much more important role.

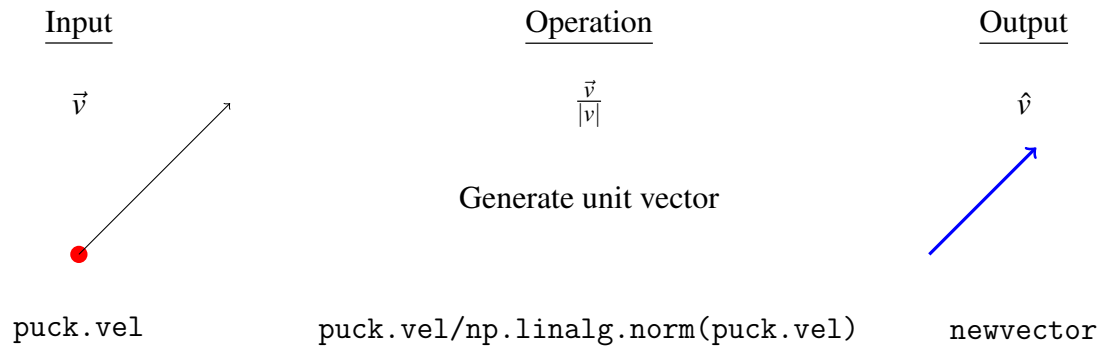
Now let's go about how we'd calculate \vec{F}_{drag} in our program. The moving object will be called puck, and `puck.vel` will represent its velocity vector. As we keep track of the puck's motion inside our `while` loop, `puck.vel` will change magnitude and direction. This means that \vec{F}_{drag} will *also* vary in magnitude and direction (as you should have concluded in Prelab Problem 5.3). Together, this means that we'll need to re-calculate \vec{F}_{drag} *every cycle* of our `while` loop, as a function of the *current value* of `puck.vel`.

Since the air drag force depends so closely on velocity (for both magnitude and direction), it will prove convenient to create `Fdrag` by starting with a copy of `puck.vel`, then modifying it. The next few diagrams will demonstrate how to use an input vector as a starting point to generate new vectors and scalars.

Let's begin by creating a vector parallel to `puck.vel`. In algebra, we would write this as \hat{v} . The following code demonstrates how to turn \vec{v} into \hat{v} :

Direction of a Vector

```
newvector = puck.vel/np.linalg.norm(puck.vel)
```



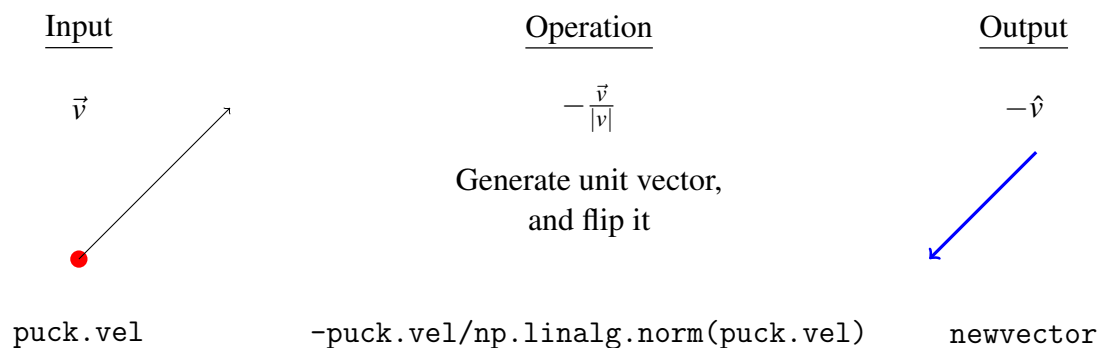
The action of this code snippet is diagrammed above: it takes as input the puck's velocity into a function called `norm` from the `numpy.linalg` package, which then outputs the magnitude of a vector, which we can use to get a unit vector parallel to `puck.vel` by dividing `puck.vel` by the magnitude. We **assign** this value to the aptly-named new vector variable `newvector`. Remember that the `=` sign in Python does not represent algebraic equality. It stands for "store whatever is on the right in the variable on the left."

But we actually need a vector pointing anti-parallel to the puck's velocity, not parallel. So, let's add a negative sign:

Anti-Direction of a Vector

```
newvector = -puck.vel/np.linalg.norm(puck.vel)
```

As diagrammed below, this does the same thing as before, except that the end result is flipped in orientation:

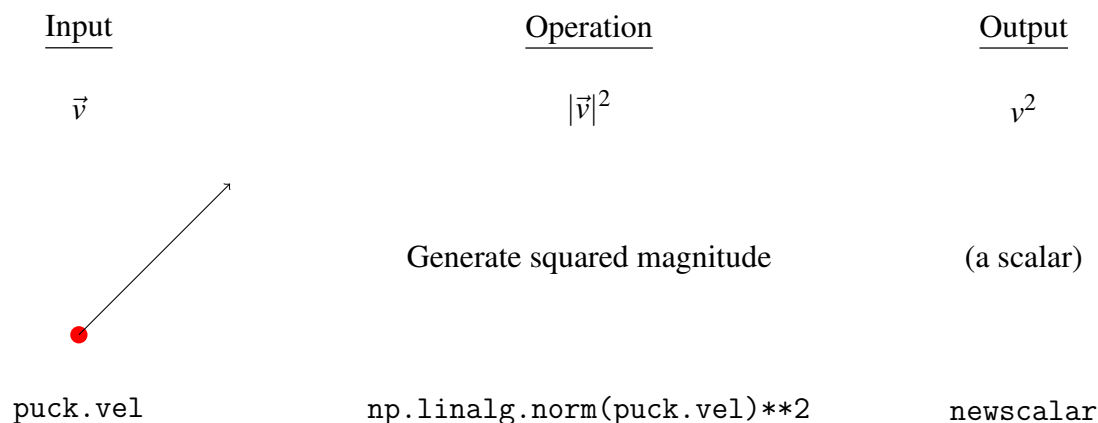


All we did was multiply the output of `norm` by a scalar (namely, -1). This is often called “scaling a vector.” In this case we “scaled” it by the value -1 . Notice we aren’t mucking about with the *components* of `puck.vel` — we’re manipulating it just like we manipulate vectors algebraically in Lecture.

Let’s look at the drag force equation again:

$$\vec{F}_{drag} = -\frac{1}{2}\rho AC|\vec{v}|^2\hat{v}. \quad (19)$$

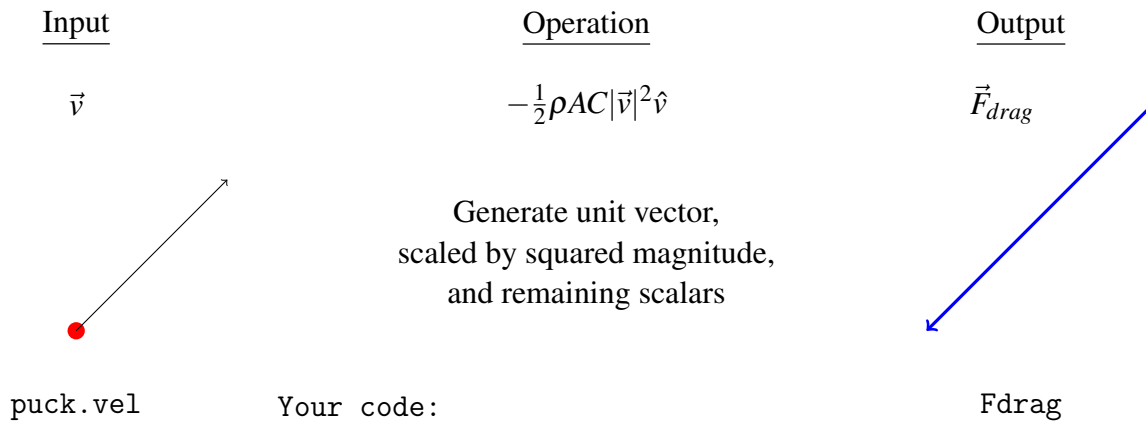
We already have the $-\hat{v}$ part down. All that’s left are some scalars: ρ , A , and C will just be constants defined in our model. Computing $|\vec{v}|^2$ will require us to square the output of `np.linalg.norm`. Here’s how that works:



This turns a vector input into a scalar output, which we can then use to do whatever we want. Well, not *whatever* we want. But we *could* scale a vector with it, for example. Note that you *could* compute the squared magnitude of a vector using an expression like `v.x**2+v.y**2+v.z**2`, but this takes more computation time for the computer than using numpy’s built-in functions. Give your fingers, and the computer, a break, and just use the built-in functions whenever you can.

If you put the above diagrams together, you should be able to combine them into the next diagram, which takes `puck.vel` as an input and follows the steps in the drag force equation (in the Operation column) to produce \vec{F}_{drag} . Just take it step by step:

1. Generate the unit vector parallel to velocity and the magnitude using `np.linalg.norm`
2. Scale the unit vector by the squared magnitude of velocity
3. Scale that result by the other constants in the drag force equation (Eq. 16). Don’t forget the minus sign!



Given the vector `puck.vel` and scalar constants `rho`, `A`, and `C` (representing the constants ρAC), how would you compute the drag force vector in Python code? There should be no individual *components* of the velocity vector in your answer, only functions of the velocity vector scaled by the other scalars. After thinking about this on your own for a bit, take a look below to see if what you came up with matches with what is provided.

Algebra: $\vec{F}_{drag} = -\frac{1}{2}\rho AC|\vec{v}|^2\hat{v} = -\frac{1}{2}\rho AC|\vec{v}|\vec{v}$

Algebra to Code Translation: Drag Force Equation

```
Fdrag = -1/2*rho*A*C*np.linalg.norm(puck.vel)*puck.vel
```

Take a moment here to check that `Fdrag` is a vector. This means that the result of evaluating the right-hand side of your equation should be a vector. If your right-hand side instead resolves to a scalar, then `Fdrag` will become a scalar. And this would wreak absolute havoc upon our Velocity Update Equation. Why? Because velocity is a vector, and we need to modify velocity with an acceleration, which is also a vector. And since $\vec{a} = \vec{F}/m$, we definitely need a vector force to compute a vector acceleration.

Lastly, since `Fdrag` depends directly on `puck.vel`, and `puck.vel` changes every cycle, this computation will need to be performed *inside* the `while` loop when we assemble our program so that it is repeated for every new value of the velocity vector. Putting this statement outside the loop would erroneously result in a constant force (and is a very common error). Keep this in mind when assembling your program in Lab!

Initial Velocity and the Inclined Plane

In this section we'll review the experimental setup for the puck launch experiment, and generate some key expressions we'll need to accurately model it.

Recall that the puck launch took place on an air table elevated at one end, so that the net effect of normal force and gravity was to accelerate the puck “down” the table. We're going to construct our model such that our view of the 3-D scene will be identical to your view of the air table from the camera. This means the x -axis points to the right, the y -axis points up, and the z -axis points out of the screen (towards the camera, perpendicular to the air table). This means that “down” is in the $-y$ direction.

Since our model is two-dimensional, and we've cleverly chosen our coordinate system to align the x - y plane parallel to the table, we won't need to worry about the z -component of any position or velocity vectors. We do, however, need to worry about the x - and y -components of other vectors, such as initial velocity.

Puck Initial Velocity

In the puck launch experiment we launched the puck with arbitrary initial speed and launch angle, then measured the x - and y - components of the initial velocity using the information gleaned from the video. How can we use this information to set the initial velocity of the puck in our model?

Recall that in our simulation of the ball drop experiment, we assigned a **vector** initial velocity to our ball object, by writing `ball.vel = np.array([0, -4, 0])` using the numpy package. This set the y -component of ball's initial velocity to -4 m/s, and the other two components to zero. To model the puck's two-dimensional motion, we'll need its initial velocity to have two non-zero components.

Given the x - and y -components of puck's initial velocity (as v_x and v_y , respectively), we would instead write the puck's initial velocity vector in Python as the following:

```
puck.vel = np.array([v_x, v_y, 0])
```

Net Force Acting on the Puck

In the ball drop model the only force was gravity, and it pointed straight down. In that model's coordinate system, “down” was equivalent to the $-y$ -direction, so we represented gravitational acceleration as below.

```
g=np.array([0,-9.8,0])
```

Characteristic Time

While creating the ball drop model, we encountered the concept of **characteristic time**, abbreviated as t_c , which characterizes the time scale over which the objects in a physical situation significantly alter their positions, velocities or accelerations.

Think about the characteristic times associated with the following:

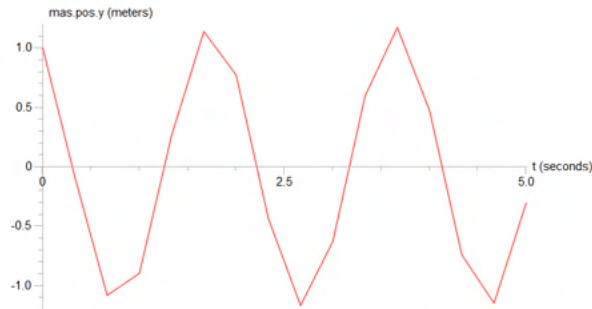
- (a) Grocery shopping
- (b) Amount of time it takes to recover from the flu
- (c) Duration of winter

To reduce numerical error, we would want to model trajectories with a small enough time interval between calculations. But *how* small is small enough? Determining this time interval is important: it's the Δt we'll use in the position and velocity update equations. Doing so is often more of an art form than an exact science.

For example, suppose we wished to model the oscillatory motion of a mass hanging from a spring. Perhaps this mass moves back and forth about once per second, and we decide to model it's motion in the vertical direction, then compare the results to a theoretical model consisting of a sine function (e.g., $y(t) = \sin(\omega t)$). Here's how we could go about determining a workable Δt :

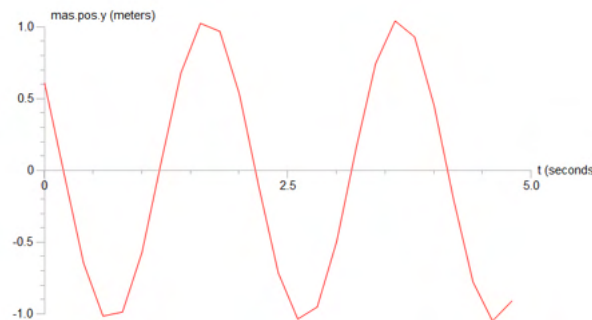
1. Get a ballpark figure for t_c . In this case, the relevant time scale is the oscillatory period, one second. Fine, let's choose $t_c = 1$ second.
2. Divide t_c by a fairly large number N . Perhaps $N = 100$ or $N = 1000$. The exact value isn't important.
3. Compute $\Delta t = t_c/N$. In this example, that's $\Delta t = 1/100$ or $\Delta t = 1/1000$ seconds, depending on N .
4. Run the model, and see if it either a) suffers from numerical inaccuracy, in which case we should increase N , or b) takes forever to run, in which case we should decrease N .

Let's look together at the results of such a model with four different values of N (3, 10, 100, and 1,000,000). Here's $N = 3$, or $\Delta t = 1/3^{rd}$ of a second:

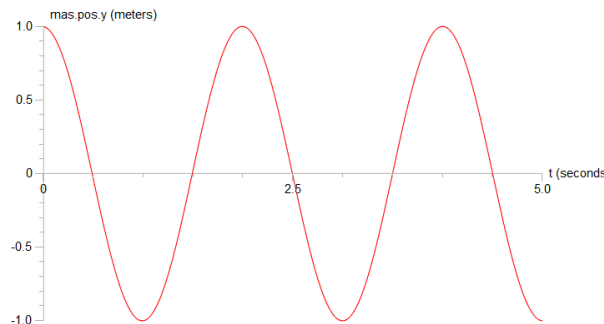


The results very significantly from theory, most noticeably by virtue of the many un-physical sharp corners. Δt is definitely too large.

Here's $N = 10$, or $\Delta t = 1/10^{th}$ of a second:



The trajectory has gotten closer to theory, but the persistence of the sharp corners impels us to shrink Δt even further. The plot below was generating using $N = 100$, or $\Delta t = 1/100$ seconds.



It completes quickly, and has negligible numerical error. Although we haven't seen this in theory yet (we'll learn about simple harmonic motion later on in lecture; in fact, we'll perform this exact experiment in a later Lab), this does in fact closely recreate the theoretical (friction-less) trajectory. Compare to the next case, where we've cranked up N all the way to 1,000,000, so that $\Delta t = 1 \mu s$, $= 10^{-6}$ s:



There's no visual difference between the latter two cases. Using a Δt of $1/100^{th}$ of a second already results in essentially no numerical error, so simulating the motion in chunks of microseconds is overkill, and succeeds only in making the program take much, much longer to run. In this case, it took my machine about 30 minutes to complete simulating 5 seconds of motion for the $N = 1,000,000$ case, instead of a negligible amount of time in the $N = 100$ case. Some simulations do require that level of detail, but none of the programs we'll write in this course will take more than a few seconds to complete.

So can we just always set $N = 100$ and be done with it? In a word, no. Remember, t_c is at best an estimate of the relevant timescale. If we over-estimate t_c by a factor of 100, and then still use $N = 100$, chances are your model will have a lot of numerical error or unphysical behavior. Therefore it's essential to do testing similar to the above, then evaluate your model's performance and adjust t_c and N as needed to produce a workable Δt .

Think back to last Lab with the puck launch experiment. What would you estimate here a t_c for the experiment to be? Remember that t_c is an estimate, and there are typically many different ways to make that estimate.

Using this t_c , what might be a reasonable value for Δt ?

These are things to keep in mind during this lab, as well as whenever you are simulating an event numerically.

Ready for the lab?

We will use the parameters in the following table to perform our simulation, so keep this chart handy.

Box 2: Simulation Parameters

Quantity	Codename	Units	Value
Density of air in the lab (ρ)	rho	kg/m ³	1.20
Width between table legs	w	m	0.570
Puck mass	puck.mass	kg	0.036
Total height of elevating blocks	h	m	0.025
x –component of puck initial velocity	v_x	m/s	0.354
y –component of puck initial velocity	v_y	m/s	0.249
Horizontal projectile range (experimental)		m	0.312
Horizontal projectile range (theoretical)		m	0.322

Projectile Motion with Air Drag: Lab

Box 3: In this lab, you will:

- ☐ Simulate the puck launch experiment without air drag.
- ☐ Compare your simulated results to your experimental results.
- ☐ Incorporate air drag.
- ☐ Investigate how air drag affects a large, fast object moving through the atmosphere.

Introduction

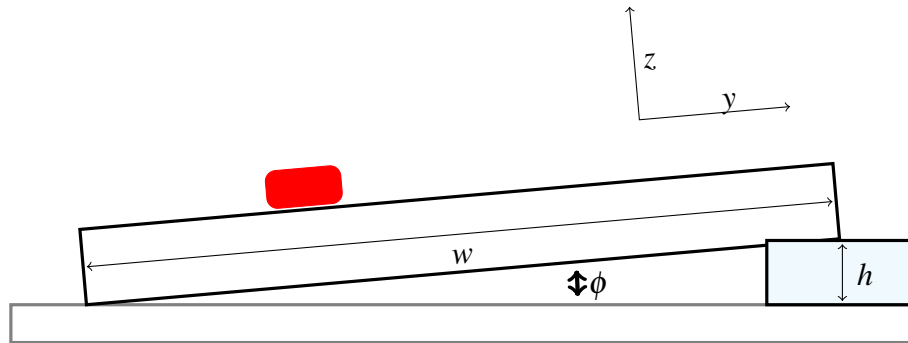
Welcome to the in-lab portion of Projectile Motion with Air Drag, in which we'll model the puck launch experiment without (at first) and (later on) with air drag, decide whether air drag significantly influenced trajectory of the puck, then model the flight of a rocket in the Earth's atmosphere with and without air drag.

Part One: Modeling Projectile Motion

Our first order of business is to get our model of the puck launch experiment up and running.

The diagram below depicts an air table of width w , with one end elevated to a height h , resulting in an inclination of ϕ radians.

Problem 5.1 For the below figure, try and draw a force diagram for the puck, and use it to try and define the net force due gravity and normal force as a function of m , g , w and h only.



You should find that we can cancel the z -component (with respect to the rotated coordinate system) of gravity with the normal force provided by the air table, leaving us with the component of gravity along the rotated y direction. (Note: This is using the rotated coordinate system shown in the diagram, not the coordinate system used for the code cell where we defined g above.)

If you get confused, go back to your lecture slides and review force diagrams for the inclined plane. You could also review the puck launch experiment in the manual and review how the force was derived there. Think about which way the puck accelerated during its motion: that way is our “down.”

Now let’s figure out how to write this in our code. For simplicity’s sake, in our program we’ll call the net force due gravity and normal force `netF`. Remember that `netF` is a vector, and due to our clever choice of coordinate system, only one of its components should be non-zero.

Code Task 5.1 Given the scalar variables m , g , w and h and your answer to Problem 5.1, write a line of code that would compute the net force from gravity and normal force. The result should be a vector (use `np.array([])`) with the correct direction — it should match the direction the puck accelerated during the puck launch experiment.

Code Task 5.2 Using your defined `netF` function, define the acceleration of the puck and edit the values for the total time of the simulation (which we will equate with our characteristic time `t_c`) and time interval between calculations `dt` until a smooth trajectory is observed that shows the puck returning to past its initial height of $y=0$.

Now that we have created the simulation to model the puck launch experiment, we can use the results to determine the vertical and horizontal range. The code we used to simulate the experiment generates lists of position and velocity of the puck at different times, which we can iterate through to find the ranges.

The vertical range can just be found using the `max()` function, which outputs the maximum of a list. Therefore, we can just use this function with the list of all the y-component positions of the puck. However, we have our simulation end at $y < 0$, which means our maximum x-component will be larger than our horizontal range (since it includes values representing the puck phasing through the edge of the air-table).

Problem 5.2 How can we use the position and velocity arrays we defined to determine the horizontal range. In your answer, be sure to address: What is the y-component of position when the x-component of position has reached the horizontal range? How can this be used with `r.array` to find the horizontal range?

Code Task 5.3 Complete the code provided in the Jupyter notebook to calculate the horizontal range and the vertical range of the simulation for the puck launch experiment. Use your answer to Problem 5.2 to finish to condition needed for the horizontal range.

Problem 5.3 Compare the horizontal range from the simulation to the experimental horizontal range and to the theoretical horizontal range values provided in Box 2. For each of the two comparisons, calculate the percent error between the values ($\% \text{ Error} = \frac{x - x_0}{|x_0|} \times 100$). Treat the experimental and theoretical ranges as the 'expected' value (x_0) and the simulation range as the 'measured' value (x).

Problem 5.4 Is the range predicted by your numerical model greater or smaller than the experimental value? Which experimental errors could cause the discrepancy you observe?

Part Two: Adding Air Drag to Free-fall

The drag force is usually written in the form $\vec{F}_{drag} = -\frac{1}{2}\rho AC\|\vec{v}\|^2\hat{v}$, or equivalently $\vec{F}_{drag} = -\frac{1}{2}\rho AC\|\vec{v}\|\vec{v}$. The drag force is always oriented directly opposite to the object's direction of motion ($\vec{F}_{drag} \parallel -\hat{v}$), hence the "-" sign in the front.

The faster an object falls, the larger the air drag force is, which decreases the acceleration. Over time, this acceleration will decrease to zero through this process. Once the object is no longer accelerating, the velocity will not change. This constant velocity is referred to as **terminal velocity**.

Problem 5.5 Using Newton's second law, solve for the terminal velocity for an object in free-fall experiencing drag. (assume here 1-dimensional motion). Your answer should be in terms of ρ , A , C , m , and g .

Problem 5.6 If the drag force F_{drag} were parallel to velocity instead of anti-parallel, what would that imply about this system?

To numerically determine what the analytic terminal velocity should be, values for ρ , A , and C need to be given.

The symbol ρ (pronounced rho) is the density of the medium. The higher the density, the more mass per unit volume. This also equates to being harder for anything to fly through. Imagine swimming through honey (higher density) versus swimming through water (lower density). Some common fluid densities are listed below:

Parameter	Value	Units
ρ_{air}	1.225	kg/m ³
ρ_{water}	1000	kg/m ³
ρ_{ice}	916.7	kg/m ³
ρ_{helium}	0.179	kg/m ³
ρ_{hydrogen}	0.0898	kg/m ³

A represents the cross-sectional area, which is what the 3 dimensional object would look like projected into a 2D plane. Imagine looking at the shadow an object creates, this is the cross-sectional area of this object. This is involved in drag force, because a larger cross sectional area, the more fluid can push against the object to cause a larger drag force. Skydivers in free fall have a relatively small cross sectional area of just their body. When they release their parachute, they significantly increase their cross sectional area with the fabric of their parachute, which increases their drag force. Formula's cross-sectional areas for certain 3-dimensional objects are shown below:

Parameter	Value	Units
A_{sphere}	πr^2	m ²
A_{cube}	l^2	m ²
A_{box}	lw	m ²
$A_{\text{cylinder (length)}}$	$2rh$	m ²
$A_{\text{cylinder (width)}}$	πr^2	m ²

C is a drag coefficient constant that is determined by the shape of the moving object. Although it is complex to derive this coefficient from models, we can assume that the coefficients are already given to us in this chart:

Parameter	Value	Units
C_{sphere}	0.50	N/A
$C_{half-sphere}$	0.42	N/A
C_{cube}	0.80	N/A
$C_{rectangular\ box}$	2.10	N/A
$C_{streamlined\ body}$	0.04	N/A

(Also feel free to google the geometric factor for anything you want). This coefficient is why engineers and designers of cars and jets create the shape in streamline shapes.

We are assuming the dependence of air drag force magnitude on speed is quadratic here ($\|\vec{F}\| \propto \|v\|^2$), because it is generally appropriate for objects moving at medium speeds through dilute fluids (ex: something moving through air), but the relation can change depend on the exact motion. For example ($\|\vec{F}\| \propto \|v\|^n$) where n is an empirically determined number that's greater than 0. We will not be investigating anything other than $n = 2$ today however.

Problem 5.7 For the 75 g ball with a radius of $r = 0.025$ m used in Lab 1, calculate what its terminal velocity would have been if it was allowed to continue to fall instead of hitting the ground. Use the information provided in the previous tables for what values to use for ρ , A , and C . Be sure to show your work.

Code Task 5.4 In the code provided in the Jupyter notebook, we simulate dropping the ball while ignoring air drag (as we did in Lab 2) and compare it with the results of a simulation that takes air drag into account. Before running the code, edit the update acceleration equation for the ball experiencing drag. Recall that to get the magnitude of velocity, use `np.linalg.norm()`. If you are stuck, look back at the Prelab Assignment.

Problem 5.8 How would you describe the differences in shape of acceleration, velocity, and position graphs between the two balls? How do the shapes of the graphs for the ball with drag indicate its reaching a terminal velocity?

Problem 5.9 How would you determine the terminal velocity based on only the *position* graph? *Hint: Could you fit a function to all/part of the data? What function would that be?*

Problem 5.10 How would you determine the terminal velocity based on the *velocity* graph? Use this method to determine the terminal velocity of the ball with drag and record it in your lab report.

Part Three: Adding Air Drag to Projectile Motion

Now we will implement drag in the simulation for the puck launch experiment we created in Part One of this lab.

Code Task 5.5 Write a line of code that would compute the net force `netF2` from gravity, the normal force, and air drag acting on the puck from the puck launch experiment. We have already generated a line of code in Part One called `netF` that incorporates gravity and the normal force, so we call it below and add onto it air drag. You are responsible for implementing air drag into this function.

Code Task 5.6 Using your defined `netF2` function, define the acceleration of the puck with drag and edit the values for the total time of the simulation (which we will equate with our characteristic time) `t_c` and time interval between calculations `dt` based on what you used for Code Task 5.2.

Problem 5.11 How would you describe the differences in shape of acceleration, velocity, and position graphs between the two pucks? Does the puck with drag reach a terminal velocity like we saw with the ball in free-fall? Explain why or why not using the plots generated.

Code Task 5.7 Run the function defined in Code Task 5.3 `findRanges` in the code cell provided in the Jupyter notebook to solve for and print out the horizontal and vertical range of the ball with drag.

Problem 5.12 Compare the horizontal and vertical ranges for from the simulations of the puck with and without drag. What is the percent error between the corresponding values? (Assume the expected values are the horizontal and vertical ranges calculated *with* drag). Based on these percent errors, is it reasonable we ignored air resistance in the previous lab? Explain your choice.

Now we will simulate what the puck launch experiment would have looked like had we submerged the puck in water to see the effect of more dense fluids on projectile motion.

Code Task 5.8 Copy your code from Code Task 5.6 and replace the density for air with the density for water. Run the simulation again.

Problem 5.13 How do the graphs for position, velocity, and acceleration differ from when the simulation was done in air? Do you think it is reasonable to neglect the drag force due to water in this situation? Explain based on the Trajectory plots generated.

We saw during Part Two that drag will lead to the object having a terminal velocity, as the drag force gets larger proportional to the square of the magnitude of the velocity. Let's observe the effects of the initial speed on the trajectory of the puck.

Code Task 5.9 Copy your code from Code Task 5.6 and replace the `multfactor` to change the initial magnitude of the velocity of the puck. You can try values less than 1, or values as large as 1,000,000 ($1e6$).

Problem 5.14 How do the trajectories of the simulations with and without drag compare at low speeds? What about high speeds? Do the ranges change with higher speeds?

Happy Landings

Congratulations on completing this Lab! You've successfully modeled projectile motion with air drag, a situation which is not trivial to analyze analytically. As advertised in the Introduction to Numerical Modeling Lab, it doesn't matter how hard the theoretical analysis is: if we can write down a force law and solve for acceleration, then we can simulate the resulting motion.

Laboratory 6

One-Dimensional Collisions

Introduction

In this laboratory, we introduce the concepts of work, energy, and energy conservation. Consider a system of objects (such as two gliders on a frictionless air track). When no outside work is done on the system, the total energy of the system is “conserved” or constant. In this laboratory, we look at energy conservation in the context of one-dimensional collisions on a frictionless surface. We also investigate another conservation law, the conservation of linear momentum, which can be derived directly from Newton’s Second Law.

Theoretical Background: Work and Energy

The **work** done on an object by a constant force \vec{F} is defined as

$$W = Fs \cos(\theta) ,$$

where F is the magnitude of the force, s is the magnitude of the object’s displacement (due to the action of the force), and θ is the angle between the force and displacement vectors. Work has units of **joules**, where $1 \text{ J} = 1 \text{ N m}$. In vector notation, we write

$$W = \vec{F} \cdot \vec{s}$$

and say that the work done is equal to the **dot product** of the force and displacement vectors. If the force varies over a general path, we add up many small pieces $\vec{F} \cdot d\vec{s}$ and get

$$W = \int \vec{F} \cdot d\vec{s}$$

Work is intimately related to **energy**; in fact, the term energy is defined as the capacity of an object to do work. Work is also defined (somewhat circularly) as the transfer of energy from one system to another. We all have an intuitive understanding of what “energy” and “work” refer to; all that remains is to describe them mathematically.

There are two basic types of energy: **potential energy** (PE) and **kinetic energy** (KE). We can think of potential energy as energy stored within a physical system; it depends on the position or configuration of a system rather than its motion. For example, gravitational potential energy is stored in an object when it is moved from a lower to a higher position relative to Earth's surface. The change in gravitational potential energy that occurs when an object is moved a vertical distance h is

$$\Delta GPE = mgh . \quad (20)$$

Another form of potential energy is elastic potential energy, which is stored when one stretches or compresses a spring away from its equilibrium position. The amount of energy stored when the spring is stretched or compressed a distance x from equilibrium is

$$EPE = \frac{1}{2}kx^2 , \quad (21)$$

where k is the spring constant. Kinetic energy, on the other hand, is associated with the motion of a system. For a single object of mass m moving at a speed v , the kinetic energy is

$$KE = \frac{1}{2}mv^2 . \quad (22)$$

All of these formulas should be familiar from lecture.

Problem 6.1 What are the two basic types of energy?

Problem 6.2 Give an example of a situation where gravitational potential energy is converted into kinetic energy. What about a situation where kinetic energy is converted into gravitational potential energy? Elastic potential energy to kinetic energy?

Theoretical Background: Conservation of Energy

According to the **Law of Conservation of Energy**, energy can neither be created nor destroyed, but can only be converted from one form to another. Conservation energy for **mechanical** systems states that the sum of all forms of mechanical energy (KE , GPE , EPE) in the initial state plus work of non-conservative forces (forces that are path dependent, such as friction) equals sum of all form of mechanical energy in the final state. Mathematically, this is written as

$$KE_i + GPE_i + EPE_i + W_{non-cons} = KE_f + GPE_f + EPE_f . \quad (23)$$

Here $W_{non-cons}$ denotes the work of non-conservative forces and could be positive, negative, or zero.

What does this mean in the context of this lab, which involves head-on collisions of two gliders? In this case, we will take the initial state to be the state of the system before collision, and the final state to be the state of the system after collision. Because we are dealing with two objects, in addition to the KE and PE of each object before and after collision, it is useful to also define an **internal energy** for the system. The internal energy of a body is generally defined as the sum total of the kinetic energies of all of its component molecules (translational, rotational, and vibrational) plus the potential energies associated with the vibrational and electrostatic energy of the atoms within these molecules.

We will examine collisions where the gravitational potential energy of the colliding objects does not change (because they collide on a level surface), and there are no springs involved. We will also minimize frictional forces by using an air track, so we can ignore $W_{non-cons}$ as well, but kinetic energy can still be converted into the internal energy of the system.

Physicists usually refer to two types of collisions: **elastic collisions** and **inelastic collisions**. The defining difference between these two types is that in an elastic collision, the total kinetic energy of the two colliding objects is the same before and after the collision. The objects simply transfer some of the kinetic energy between themselves, and no kinetic energy is converted into internal energy (or vice versa). In other words, kinetic energy is conserved in elastic collisions.

In inelastic collisions, however, kinetic energy is *not* conserved. Instead, while the collision is taking place, some of the kinetic energy is transformed into the internal energy of the colliding objects.

The difference is somewhat intuitive. An example of a completely inelastic collision would be one between two balls made of putty – the objects are “squishy” and somewhat deform during the collision, and they stick together afterward. A series of snapshots of two inelastic collisions is shown in Figure 1. An example of an elastic collision, in contrast, would be one between two pool balls – the objects are hard, do not change shape during the collision, and bounce off each other cleanly. A series of snapshots of two elastic collisions is shown in Figure 2. There are also many types of collisions that are in the intermediate range between elastic and inelastic; there is some kinetic energy loss, but the objects do not stick together.

Problem 6.3 What energy conversion takes place during an inelastic collision that does not take place during an elastic collision?

Problem 6.4 What are some differences between inelastic and elastic collisions? Give an example of each.

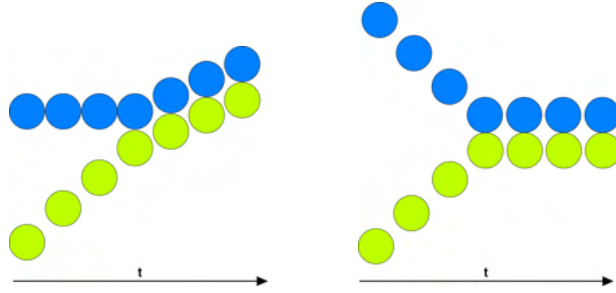


Figure 1: A series of snapshots of two different inelastic collisions.

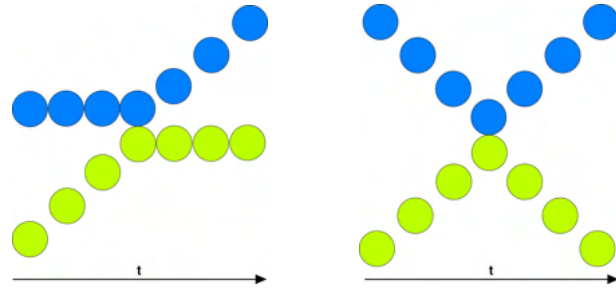


Figure 2: A series of snapshots of two different elastic collisions.

Theoretical Background: Conservation of Linear Momentum

Linear momentum, in contrast to energy, is a vector quantity, and it is defined as the product of an object's mass and its velocity, or

$$\vec{p} = m\vec{v}. \quad (24)$$

Therefore, linear momentum has both magnitude and direction, and the mathematics of momentum conservation involves breaking momentum vectors into their components, just like we did with forces.

The **Law of Conservation of Linear Momentum** is basically just a restatement of Newton's Second Law, which states that

$$\vec{F}_{\text{ext}} = \sum_i \frac{\Delta \vec{p}_i}{\Delta t} = \sum_i m_i \vec{a}_i$$

if the masses remain constant. In the absence of any net external force,

$$\sum_i \frac{\Delta \vec{p}_i}{\Delta t} = 0,$$

and therefore

$$\sum_i \vec{p}_i = \text{constant} ,$$

and momentum is conserved. Thus, by Newton's Second Law, momentum is conserved in any situation in which there is no net external force.

Problem 6.5 Momentum and kinetic energy both depend on mass and velocity and nothing else. So how is it possible then that momentum is *always* conserved (in both inelastic and elastic collisions, for example) and kinetic energy is not conserved in inelastic collisions?

Experimental Background: Two Gliders

In this laboratory, we will be using the air track setup of Laboratory 3, but with one important difference: there will be *two* gliders on the track, as shown in Figure 3.

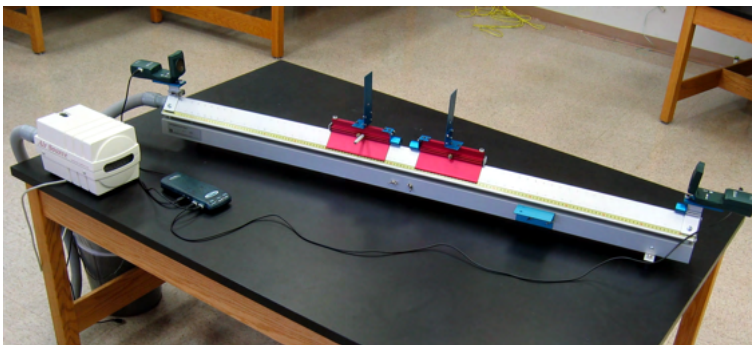


Figure 3: Air track with two gliders.

The gliders themselves can be positioned in two ways. If the sides with velcro on them are facing each other, any collision between the gliders will be an inelastic collision – the gliders will stick together and travel as one object after the collision. However, if the sides with metal bumpers on them are facing each other, the gliders will bounce off each other cleanly, and any collision between them will be an elastic collision.

Problem 6.6 If we collide two gliders on an air track, will the total linear momentum of the two-glider system be conserved if the environment of the air track is not perfectly frictionless? Why or why not?

Problem 6.7 If we collide two gliders on an air track, will the total linear momentum of the two-glider system be conserved if the track is not perfectly level? Why or why not?

The motion of each glider is detected using the same type of ultrasonic sensor that we used in previous laboratories. Because there are two gliders, there must be two sensors; they are positioned at opposite ends of the air track, and each “sees” the glider closest to it.

Experiments

Experiment 1: Inelastic Collisions

We begin by investigating whether kinetic energy and momentum are conserved in the context of *inelastic* collisions. We will generate experimental data for two separate inelastic collisions.

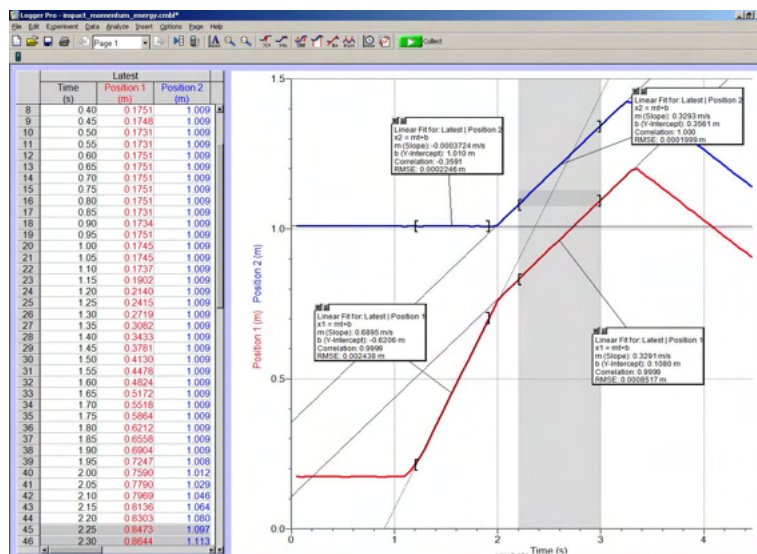


Figure 4: Plot of position vs. time for two inelastically-colliding gliders.

1. Turn on the air compressor and place a single glider on the track.
2. Ensure that your track is level by adjusting the leveling screw so that the glider has no systematic tendency to move toward one end.
3. Once the track is level, place the second glider on the track. Make sure that the gliders are oriented with the textbfvelcro ends facing each other. Remove any additional brass weights from the gliders.
4. Open the *LoggerPro* template file *impact_momentum_energy.cmbl*. This will open *LoggerPro* and set it up to collect 5 seconds of data from both sensors simultaneously at a rate of 10

samples per second. Practice collecting data and moving the gliders around so you can see which glider corresponds to which sensor. If the data look bumpy instead of smooth, adjust the tilts of the sensors to ensure they are “seeing” the gliders and not the track or surrounding objects.

5. Weigh both gliders, and record their masses in your lab report.
6. Position one glider in the middle of the track and the other at one end. Begin taking data, and give the glider at the end of the track a gentle push toward the one in the middle. The gliders will collide, and your data will look similar to that in Figure 4.
7. Fit lines to the regions of the graph immediately before the gliders collided and immediately after they collided (see Appendix A Section 1.2). Copy the graph into your lab report, and record the values of the slopes of these lines in the code cell provided in the Jupyter notebook.

Problem 6.8 What do the values of the slopes of your fits represent?

8. Repeat this procedure, but add 100 g of brass weights to the glider that is at rest before the collision (see Figure 5).

Problem 6.9 Calculate the total momentum of the system before and after the collision for both with and without the additional mass.

Problem 6.10 Calculate the total kinetic energy of the system before and after the collision for both with and without the additional mass.

Problem 6.11 Calculate the ratio of total momentum after the collision to total momentum before the collision, and do the same for kinetic energy. Do this for both data sets taken with and without the additional mass.

Experiment 2: Elastic Collisions

We next investigate whether kinetic energy and momentum are conserved in the context of *elastic* collisions. The procedure for Experiment 2 is identical to the procedure for Experiment 1, with one very important difference. In Experiment 1, you oriented the gliders so that the velcro ends were facing each other; this produced the right conditions for inelastic collisions. In Experiment 2, however, you will orient the gliders so that the ends with the metal bumpers are facing each other. This will produce the right conditions for elastic collisions, in which the gliders bounce off each other when they collide.

Repeat the procedure for Experiment 1 with the metal bumpers facing each other. Copy both graphs (with fit lines included) into your lab report, and record the fit parameters.

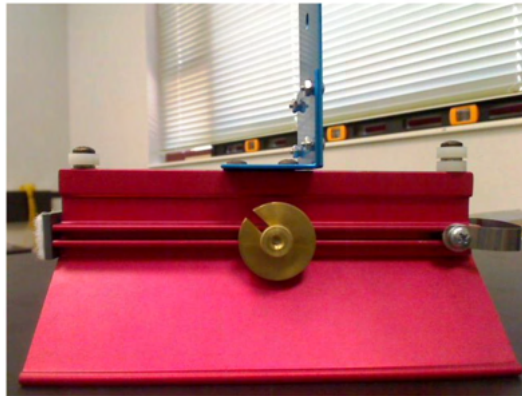


Figure 5: Attach two 50 g brass weights to the glider's base, one on either side.

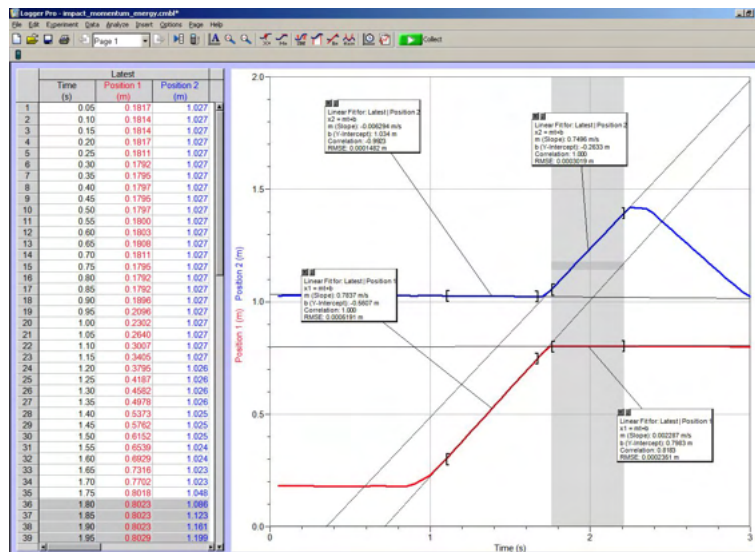


Figure 6: Plot of position vs. time for two elastically-colliding gliders.

Problem 6.12 Calculate the total momentum of the system before and after the *elastic* collision for both with and without the additional mass.

Problem 6.13 Calculate the total kinetic energy of the system before and after the *elastic* collision for both with and without the additional mass.

Problem 6.14 Calculate the ratio of total momentum after the *elastic* collision to total momentum before the *elastic* collision, and do the same for kinetic energy. Do this for both data sets taken with and without the additional mass.

Experiment 3: Model Fitting and Analysis

Use the online data collection tool to record the ratios of momentum and kinetic energy after the collision to before the collision, where

$$\text{ratio} = \frac{\text{quantity after}}{\text{same quantity before}} ,$$

for the inelastic collisions and the elastic collisions. If a given quantity is conserved, the ratio will equal one; however, keep in mind that these are experimental quantities subject to statistical and systematic error.

1. Enter your data into the online data collection tool (see Appendix A Sections 2.1-2.3). We will use both charts this time. Enter the momentum ratios into Chart 1 and the kinetic energy ratios into Chart 2. Within each chart, enter the ratios in the following order: inelastic (no added mass), inelastic (with 100 g added), elastic (no added mass), elastic (with 100 g added).
2. When all of the class data has been collected, follow the instructions provided in the Jupyter notebook to download and save the class data as a '.csv' file so that it can be imported into Jupyter notebook.
3. Once the data is imported into Jupyter notebook, use the provided code cell to calculate the mean, standard deviation, error of the mean, and 95% confidence interval for the mean of the momentum and kinetic energy ratios for all four collision types.

Problem 6.15 From the class's experimental results, does it appear that *linear momentum* is conserved in inelastic collisions? How about in elastic collisions? Explain how you reached your conclusion. Use statistical arguments and be careful about your wording. In statistics, we can only *reject* hypotheses, never prove them to be true.

Problem 6.16 From the class's experimental results, does it appear that *kinetic energy* is conserved in inelastic collisions? How about in elastic collisions? Explain how you reached your conclusion (use statistical arguments).

Problem 6.17 Will momentum and kinetic energy still be conserved in the presence of these sources of error? Write a one-sentence explanation for each of your answers.

- (a) The gliders experience air resistance.
- (b) The track is not level.
- (c) The glider has a non-zero velocity at the beginning of each trial (rather than being at rest).

Experiment 4: Mechanics of Car Crashes

Major strides have been taken over the past half-century or so towards increasing car passenger safety, especially in car crashes. The framework of one-dimensional momentum conservation provides an excellent means to examine the efficacy of safety devices and design elements to reduce the force felt by crash victims during impact.

We will look at two data sets which represent the position of the center of mass of a car versus time for two different cars, one of which is a modern safety marvel with crumple zones, and the other of which is a 1950s econo-clunker. To access the data sets, open the *CarCrash.cmb1* file in the *LoggerPro Templates/* folder. Note that these are two different collisions, i.e. the blue car is not crashing into the red car.

Problem 6.18 We are interested primarily in the average acceleration felt by the car passengers. Come up with a way to estimate this for each of the data sets. Remember that average acceleration can be approximated as $a \approx \Delta v / \Delta t$, where Δt is the time interval during which the velocity changes and Δv is the amount by which the velocity changes.

Problem 6.19 Examine Table 1 regarding humans and acceleration norms and decide the likely fate of the unfortunate passengers in our experiment. Explain your conclusions using your analysis from the previous problem.

Problem 6.20 The position vs. time data you examined describes the center of mass motion of the car, which is not identical to the center of mass motion of its passengers. What further means do car manufacturers have at their disposal to exploit this fact to increase passenger safety, and how would each of these methods protect crash victims?

Table 1: Accelerations Experienced by Humans and Their Consequences

a	Duration	Potential Consequences	Experienced by...
1 g	Permanent	Proper bone growth	You
3-4 g	A few seconds	None to healthy riders	Roller-coaster thrill seekers
4-8 g	< a few seconds	Blackout	Fighter pilots
Up to 15 g	< 1 s	Severe injury if prolonged	Pilots and astronauts in training
30-35 g	\ll 1 s	Broken ribs	Crash victims with seat belt
70-100 g	\ll 1 s	Violent death	Crash victims with no seat belt

Laboratory 7

Two-Dimensional Collisions

Introduction

In Laboratory 6, we showed that linear momentum is conserved in both inelastic and elastic collisions, whereas kinetic energy is only conserved in elastic collisions. We will be looking at collisions again this week, but in the context of two-dimensional collisions. The major difference in the analysis is that, in Laboratory 6, we used a single calculation of total momentum before and after the collision to establish whether momentum was conserved. In this week's laboratory, we will use two calculations: the total momentum in the x and y directions. This illustrates an important point: every extra dimension in a problem leads to one more equation for vector quantities like momentum. However, for scalar quantities like energy, the conservation law can be represented by a single equation no matter how many dimensions a problem involves. This is why conservation of energy is such a powerful tool in multidimensional problems.

In this laboratory, we will focus on inelastic collisions, which have an added feature in two dimensions: When two extended objects collide inelastically in two dimensions, they usually begin to rotate about their combined center of mass. This makes their motion after the collision much more challenging to analyze, which is why the collision problems you see in lecture usually deal only with point masses. This lab is an introduction to the complex nature of collisions as they occur in the real world.

Theoretical Background: Collisions Revisited

The most important facts to remember when solving collision problems are:

1. Linear momentum is always conserved, in every dimension, for every type of collision, as long as no external forces act on the colliding objects. When a collision involves only two objects, we write this as

$$\vec{p}_1 + \vec{p}_2 = \text{constant}.$$

Because momentum is a vector, we can break this equation into components and say that the total linear momentum in the x direction is constant *and* that the total linear momentum in the y direction is constant.

2. Total energy is always conserved. This is clear in elastic collisions, where the initial and final energy can be measured in the form of kinetic energy. In inelastic collisions, however, some of the energy is transferred to heat and sound. Without knowing the exact microphysics we are incapable measuring all of the energy. Although total energy is conserved in this case, total mechanical energy (gravitational, kinetic, elastic) is not. The Law of Conservation of Energy, like the Law of Conservation of Linear Momentum, cannot be violated.
3. In elastic collisions, kinetic energy is conserved, so that

$$KE_i = KE_f.$$

4. In inelastic collisions, kinetic energy is not conserved because some of it is converted into the internal energies of the colliding objects. Since total energy must still be conserved, you can calculate exactly how much internal energy was gained during the collision by measuring how much of the total kinetic energy was lost during the collision.

Theoretical Background: The Center of Mass

In discussing momentum conservation, it is important to define one further quantity, the **center of mass** of a system of particles. The center of mass of a system is a specific point at which we can consider all of the system's mass to be concentrated when solving certain types of problems. For example, in physics we routinely consider extended objects such as cars and elephants to be “point particles” when describing their motion using kinematic equations or Newton's Laws. What we are really doing is treating these objects as though all of their mass is located at the center of mass.

The center of mass of a system of particles is defined as the average of the particles' vector positions \vec{x}_i weighted by their masses m_i , or

$$\vec{x}_{\text{cm}} = \frac{\sum_i m_i \vec{x}_i}{m_{\text{tot}}}, \quad (25)$$

where

$$m_{\text{tot}} = \sum_i m_i.$$

This means that the center of mass of a system of particles will be closer to heavier particles and farther from lighter ones. This definition works for any number of point particles, and we can also use it with extended objects (such as elephants) by considering them to be infinite collections of point particles.

Problem 7.1 If a hockey puck of mass m_1 is situated on the x -axis at x_1 , and another hockey puck of mass m_2 is situated on the x -axis at x_2 , where is the center of mass of the system located? Where is the center of mass for two pucks of *equal* mass?

We can also define a quantity called the **center of mass velocity**:

$$\vec{v}_{\text{cm}} = \frac{\sum_i m_i \vec{v}_i}{m_{\text{tot}}} = \frac{\Delta \vec{x}_{\text{cm}}}{\Delta t}. \quad (26)$$

Note that the numerator in the middle equation is just the total linear momentum of the system. Therefore, if momentum is conserved in a particular collision, the center of mass velocity should be the same before and after the collision.

Theoretical Background: Mass vs. Moment of Inertia

We can think of **mass** as a measure of how hard it is to get something to move (translationally, that is). **Moment of inertia**, in contrast, is a measure of how hard it is to get something to *rotate*. Both are scalars. This fact can make your life a lot easier when solving problems because, if you have a system of multiple rotating objects, you can break that system into its component parts, calculate their individual moments of inertia, and add them up to determine the combined moment of inertia.

Two important moments of inertia to consider are: a) that of a **point mass** of mass m located at a distance R from the axis of rotation. Its moment of inertia is just

$$I = mR^2 \quad (\text{for a point mass}). \quad (27)$$

b) that of a **disc** of mass m and radius R . The moment of inertia about an axis passing through the center of the disc perpendicular to its plane is

$$I_{\text{disc}} = \frac{1}{2}mR^2 \quad (28)$$

Theoretical Background: Rotational Kinetic Energy

Because a rotating object has mass, and that mass is moving, we intuitively sense that the object should have some kinetic energy. However, we are used to thinking of kinetic energy in terms of a point object moving along a straight path, so it is not immediately obvious how we can quantify rotational kinetic energy or understand it in terms of concepts we have learned before. The trick is to look at a rotating object as a large set of point particles, each of which moves around the axis of rotation at a certain radius.

Each of the point particles has a different speed, but the same angular speed ω . The speed of the i^{th} particle, which we will call v_i , is given by $v_i = r_i \omega$.

Problem 7.2 If we assume our rotating object is rigid, such that all of its component particles have the same ω , do particles whose distances r_i from the axis of rotation are large have larger or smaller v_i than particles closer to the axis? Why?

Problem 7.3 Show that the product $r_i \omega$ has units of speed or velocity (m/s in SI units).

If we use the standard formula $\frac{1}{2} m v^2$ for the kinetic energy of a point particle, we can write the kinetic energy of the i^{th} particle as

$$\frac{1}{2} m_i v_i^2 = \frac{1}{2} m_i r_i^2 \omega^2.$$

If we then add up the kinetic energies of all of the different particles, we obtain

$$KE_{\text{rot}} = \sum_i \frac{1}{2} m_i r_i^2 \omega^2.$$

Pulling the constants $1/2$ and ω^2 out of the sum, we are left with the sum of the masses (of the individual point particles) times their radii squared. This is actually the formal definition of the moment of inertia

$$I \equiv \sum_i m_i r_i^2.$$

where m_i are the masses of the components of the object and r_i are their distances from their common center of mass. With this definition, we can rewrite the **rotational kinetic energy** formula as

$$KE_{\text{rot}} = \frac{1}{2} I \omega^2, \quad (29)$$

which is the formula you have seen in class for rotational kinetic energy.

The important thing to remember when you are doing problems involving rotational kinetic energy is that Eq. 29 already takes into account the distance of each part of the object from the rotational axis via its dependence on the moment of inertia. Using the moment of inertia means you can avoid having to constantly worry about what all of those individual point masses that make up the extended object are doing.

Problem 7.4 If mass is measured in kilograms and distance is measured in meters, what are the units of I , the moment of inertia?

Problem 7.5 What are the units of rotational kinetic energy? Using Eq. 29 and your answer to Problem 7.4, show that the units of rotational kinetic energy are the same as the units of translational kinetic energy.

Theoretical Background: More About Moment of Inertia

Moment of inertia is also important for any analysis of two-dimensional collisions between extended objects. In particular, in an inelastic collision between two extended objects, some of the initial translational kinetic energy is converted into the rotational kinetic energy of the combined object after the collision.

As we already know, one cannot obtain an object's rotational kinetic energy from its rotational speed ω without also knowing its moment of inertia. In a situation where multiple extended objects are colliding inelastically and sticking together, the moment of inertia of the combined object can become quite difficult to calculate. We can exploit two properties of the moment of inertia to simplify our calculations.

First, for a rigid object rotating in two dimensions, the moment of inertia is a **scalar**. This means that we can find the moments of inertia of the different parts of the combined object separately and then add them up algebraically to obtain the total moment of inertia. Therefore, we should start by looking for ways to break the combined object into pieces which are simple geometric shapes.

Second, we can use the **parallel axis theorem**. The parallel axis theorem is a very powerful tool in the study of rotational motion. If you know the moment of inertia for a body rotating about an axis that passes through its center of mass, you can find its moment of inertia for rotation about any other axis *parallel* to that axis using the parallel axis theorem, given by

$$I = I_{\text{cm}} + mh^2, \quad (30)$$

where m is the mass of the object and h is the perpendicular distance between the two parallel axes. In other words, the moment of inertia of an object about any axis is equal to the moment of inertia it *would* have about that axis if all of its mass were concentrated at its center of mass (mh^2) plus its moment of inertia for rotation about its own center of mass (I_{cm}).

Problem 7.6 Suppose you have two identical circular pucks (solid disks) of mass m and radius r sitting next to each other such that their edges are just barely touching. What is their combined moment of inertia about the point at which they are touching? Now suppose you have three such pucks sitting in a row with their edges just barely touching. What is their combined moment of inertia about the center of the middle puck?

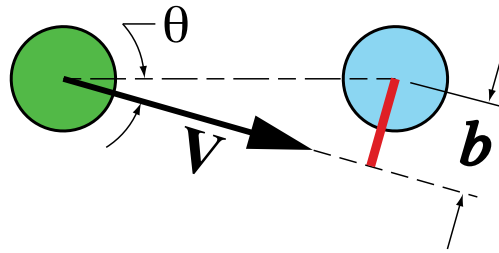


Figure 1: An impending inelastic collision. The impact parameter b is the perpendicular distance between the projected velocity vector of the moving particle and the center of mass of the stationary particle (the length of the red line). If $b > 0$, rotation will occur.

Theoretical Background: Rotational Motion in Inelastic Collisions

When the collision of two extended objects is perfectly inelastic, the objects will rotate about their combined center of mass. The combined center of mass may or may not reside within one of the objects; this depends on the relative masses and geometries of the objects.

To see why this rotation occurs, consider the two-puck system shown in Figure 1. The blue puck is at rest, while the green puck has velocity V . The angle between the velocity vector and the line joining each puck's center of mass is θ . If the velocity vector is collinear with the center of mass line ($\theta = 0$), the two pucks will not rotate about their combined center of mass after the collision (this is the situation we encountered in Laboratory 6). However, if $\theta \neq 0$, the two pucks will rotate about their combined center of mass. The rate of rotation resulting from this sort of two-dimensional encounter is related to the **impact parameter** b , which is the shortest distance between the velocity vector of the moving object and the center of mass of the other. The larger the impact parameter, the higher the rate of rotation after the collision. To keep things as simple as possible, we will only analyze collisions with one puck initially at rest.

Theoretical Background: Reference Frames

When a physics problem involves the rotation of an extended object or the motion of multiple particles, it is often helpful to separate any motion of the center of mass from the motion *relative to* the center of mass. When analyzing the two-dimensional collisions in this laboratory, we will subtract off the motion of the center of mass and only look at the motion of the components of the system relative to that center of mass. Because momentum is conserved in the absence of external forces, the center of mass velocity should be the same before and after any collision. Therefore, removing that motion from the analysis will make things simpler but should not affect

conservation of energy or any other important aspects of the collision. Analyzing data in this way is called “working in the **center of mass reference frame**.”

In general, whenever you take a measurement of an object’s motion, you are doing it relative to some **reference frame** which you take to be stationary. For example, in Laboratory 3, we collected data in the reference frame of the ultrasonic sensor, which we took to be stationary. In Laboratory 4, our reference frame was the air table. In this laboratory, our reference frame will start out as the air table. By subtracting the motion of the pucks’ combined center of mass *relative to* the air table, we will end up in the reference frame of the pucks’ center of mass. You will learn more about reference frames if you study special relativity in the future.

Problem 7.7 Suppose one puck moves toward another (stationary) puck along a straight line on an air table. In the reference frame of the air table, only the first puck is moving. In the reference frame of the pucks’ combined center of mass, how many pucks are moving? Explain.

Problem 7.8 A straight river flows along at a speed v_r , and a crocodile in the river swims upstream at a speed v_c , where v_c is measured from the reference frame of the river water. What is the speed (and direction) of the crocodile from the reference frame of the river bank? What is the speed (and direction) of the river bank from the reference frame of the crocodile? What is the speed (and direction) of the river water from the reference frame of the crocodile?

Theoretical Background: Rotational Motion and Angular Speed

In this lab, you will be analyzing the pucks’ rotation after collision. This will be difficult in the reference frame of the air table (laboratory frame) because the motion of the pucks have both translational and rotational motion. Instead, you will want to convert to a reference frame where the pucks only rotate without any translational motion. This reference frame is the pucks’ center of mass reference frame. With only rotational motion remaining in this reference frame, the angular speed of rotation of the two pucks can be found by fitting a sine function to the x and y positions vs time.

As shown in Figure 2, the projection of an object in circular motion (as shown in the left panel) onto the y -axis follows a sinusoidal pattern (as shown in the right panel). If the object is rotating with an angular speed, ω , the projection onto the y -axis will be

$$y = h \sin(\omega t + \phi),$$

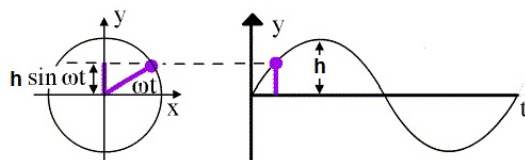


Figure 2: Projection of an object in circular motion.

where h is the radius of the circular path and ϕ is the initial phase that depends on the start of the measurement. Similarly, the projection onto the x -axis will be a cosine function instead of a sine function.

Therefore, in this lab, after you convert your data from the laboratory frame to the pucks' center of mass reference frame, the angular speed of rotation of the two pucks can be found by fitting a sine function to the x or y position vs time. When we analyze our circular motion data, we perform find a fit of the form

$$y = A \sin(Bx + C) + D.$$








Comparing this to the previous equation, we see that $B = \omega$ is the parameter of interest, since ω with the moment of inertia determine the rotational kinetic energy.

Experiments

Experiment: Two-Dimensional Inelastic Collisions

In this lab, we will only study an inelastic collision in two dimensions. Reviewing elastic collisions in two dimensions will not provide us with significantly more insight than the one-dimensional inelastic collisions we performed in Lab 6 gave us. However, we can explore rotational motion in addition to reinforcing concepts covered in last lab by exploring inelastic collisions in two dimensions.

1. There is no *LoggerPro* template for this laboratory. Open *LoggerPro* from desktop and insert a new video capture window (see Appendix A Section 1.8). Also, the reflection from the table might not allow you to see the blue dots of the puck. If this is the case, either estimate the center of the puck during data analysis or click 'Options' and go to 'Camera settings' to change the brightness and contrast of the capture window.
2. Level the air table. A puck placed in the center should not move in any preferred direction. If necessary, make use of the leveling screws on the underside of the table. Also, make sure that the camera is aligned correctly; in the video capture window, the edges of the air table should fit within, and approximately line up with, the edges of the camera's view.

3. Position two equal-mass pucks (**with velcro edges**) on the air table, with one puck stationary near the center of the table and the other placed in the puck launcher. You will want to launch this puck with moderate speed, so you may want to use one of the front sets of launcher pegs.
4. Practice launching the puck toward the other with the goal of maximizing the impact parameter of the collision. For later analysis, it is important that the pucks complete **at least one full rotation** about their combined center of mass after sticking together and before hitting any rail on the edge of the air table.
5. Ready the puck for launch and click *Start Capture* in the video capture window. Give *LoggerPro* a moment to begin recording video, and then launch the puck toward the stationary puck. Record the puck leaving the launcher, colliding with the stationary puck, and ultimately hitting one of the rails on the edge of the air table.
6. Obtain position vs. time data for each puck separately (and for both x and y dimensions) by analyzing the video in *LoggerPro*. The following step-by-step procedures will help you.
 - (a) Click the  *Enable / Disable Video Analysis* button at the bottom right of the video capture window to display the video analysis toolbar.
 - (b) Use either the slider bar or the double-arrowed buttons   in the video capture window to queue the first frame where the puck is *no longer in contact with the rubber band*. The center of the puck in this frame will be its initial position.
 - (c) With this frame queued, click  *Set Origin*, and then click on the center dot of the (moving) puck.
 - (d) Set the size scale: Click the  *Set Scale* button, and then click and drag between the air table's center dot and the screw head that fastens the puck launcher to the air table. This distance is **0.475 m**.
 - (e) Track the moving puck's coordinates first: Click the  *Add Point* button, and then click once on the center dot of the puck (where the origin should be located). The video will advance one frame for each point you add in this way. Continue this only until either one of the pucks comes into contact with one of the rails on the air table.
 - (f) After tracking the motion of the moving puck, click the  *Set Active Point* button, add a new point series, rewind the video to the frame where you started tracking the first puck, and track the motion of the other (initially stationary) puck. *LoggerPro* will automatically create a graph containing four separate data sets (x and y positions for each puck).
7. Separate the graph in *LoggerPro* into two separate graphs, one for the x dimension and one for the y dimension (see Appendix A Section 1.3 for instructions on how to make extra

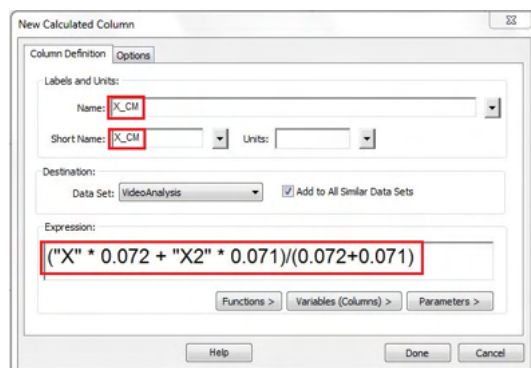


Figure 3: Screenshot of creating a new calculated column for the center of mass x position. Click on the 'Variable (Columns)' button to select the columns you want to include in your column calculation.

graphs). Paste these graphs into your lab report, and state what is moving and what the reference frame is.

Now you will create some new calculated columns for your data that “filter out” the motion of the center of mass.

1. Create two new calculated columns in *LoggerPro* for the x and y positions of the center of mass, using the formula for the center of mass given in the theoretical section of this lab (see Appendix A Section 1.5).
2. Make a graph displaying both the x and y positions of the combined center of mass as a function of time. Do a linear fit and check if your data is linear. Paste this graph into your lab report, and state what is moving and what the reference frame is.
3. Create four more calculated columns that contain the following functions:

$$y_1 - y_{\text{cm}}$$

$$x_1 - x_{\text{cm}}$$

$$y_2 - y_{\text{cm}}$$

$$x_2 - x_{\text{cm}}$$

Make sure you understand what these calculated columns are doing.

4. Make two more graphs containing these newly calculated quantities. One graph should show the x positions, and the other should show the y positions. Fit a line to the region *before* the collision for each data set.

5. Save a backup copy of your *LoggerPro* file to the desktop before proceeding to the sine fit in the next step.
6. In order to find the final rotational kinetic energy, you will need to calculate the angular speed of rotation ω of the combined puck system. You can do this by fitting a sine curve to the region *after* the collision for the four position vs. time graphs. The fitting function will be of the form

$$y = A \sin(Bx + C) + D ,$$

and the parameter B is equal to ω (see Appendix A Section 1.4 for how to fit a curve to a graph). Do this fitting for all of the four position vs. time graphs and average the four values.

7. Paste the two graphs (with the linear and sine fits - be careful about which data points you use for the linear and sine fits, since the transition can be subtle) for the inelastic collision into your lab report, and state what is moving and what the reference frame is.
8. Measure the mass of the pucks and record them in your report. Then use the linear fits from your first two plots to determine the initial velocity of each puck and use them to calculate the initial translational kinetic energies of each puck. Record these values in your report as well.

Problem 7.9 Calculate the moment of inertia of the combined object for rotation about its center of mass (which is the point at which the two pucks meet, since they have equal mass and radius). You will need to use Eq. 30 in your calculation (radius of the puck is 1 inch).

9. Use your sinusoidal fits from your last two plots to determine the average angular velocity of the puck system after the collision and use it to calculate the final rotational kinetic energy. Record these values in the table provided.

Problem 7.10 Is kinetic energy conserved in your inelastic collision? Is this the result you expect?

Problem 7.11 Is it possible to have an inelastic collision in which there is no rotation about the combined center of mass after the collision? If so, what would such a collision look like and how it would produce no rotation? If not, explain.

Problem 7.12 Hopefully you observed that the x and y positions of the center of mass were straight lines with respect to time. What would it imply if one of them (the y position graph, for instance) were parabolic instead of straight? Would momentum be conserved in this system? Why or why not?

Problem 7.13 Suppose someone gave you data from the center of mass frame only. From this data, can you tell whether the puck in the lab frame was stationary or not? In order to tell, you need to convert the center of mass frame to the lab frame without any additional information. If you cannot tell, what additional information do you need?

Laboratory 8

Rotational Motion: The Inclined Plane

Introduction

With this laboratory, we begin a study of rotational motion. While all of the rotating systems you investigate initially may appear very different, you will find that they can all be analyzed using the same basic principles of energy and momentum conservation that we have seen in previous laboratories. The equations of rotational motion will look slightly different, but they are very similar in structure to the equations that describe translational (linear) motion because they are built around these same two powerful conservation laws.

Until this point, we have analyzed all linear motion by treating the objects involved as point particles; that is, we assume that all forces acting on an object act at exactly the same spot. But in rotational motion, it matters where on the object the forces act. Forces acting on one part of the object may cause it to rotate, but those same forces acting on another part may cause no rotation at all.

Theoretical Background: Moment of Inertia of Different Objects

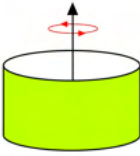
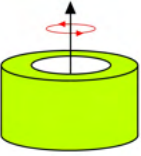
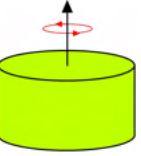
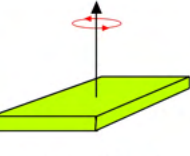
Take a look at the moments of inertia listed in Table 1 for some different object geometries, where the same total mass is distributed in different ways about the axis of rotation.

From these expressions, it is easy to see that adding additional mass to an object will increase I ; however, it is more important to realize that you can also increase I by keeping the mass the same and just changing how it is distributed around the axis of rotation.

Problem 8.1 Why is the expression for the moment of inertia of a point mass the same as the expression for the moment of inertia of a thin-walled cylinder?

Recall the formal definition of moment of inertia,

$$I \equiv \sum_i m_i r_i^2.$$

			
thin hoop or ring of radius R and mass m	thick ring of inner radius R_1 , outer radius R_2 , and mass m	solid cylinder or disk of radius R and mass m	flat plate with sides of length A and B and mass m
$m R^2$	$\frac{1}{2} m (R_1^2 + R_2^2)$	$\frac{1}{2} m R^2$	$\frac{1}{12} m (A^2 + B^2)$

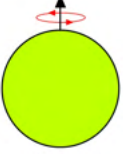

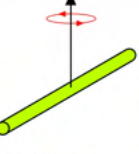
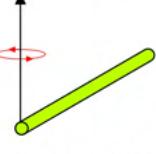
			
solid sphere of radius R and mass m (shown in cross-section)	hollow sphere of radius R and mass m (shown in cross-section)	slender rod of length L and mass m , spinning around center	slender rod of length L and mass m , spinning around end
$\frac{2}{5} m R^2$	$\frac{2}{3} m R^2$	$\frac{1}{12} m L^2$	$\frac{1}{3} m L^2$

Table 1: Moments of inertia for some common object geometries.

from Laboratory 7. Compare this formula to that for the moment of inertia of a point mass (Eq. 27). It must be the case then that the common formulae shown in Table 1 for the moments of inertia of various objects are really just simplifications of this sum (someone used calculus to do the sum in advance for different object geometries).

In this discussion, we will focus specifically on thin-walled and solid spheres and cylinders. Notice that all of these objects have similar-looking expressions for their moments of inertia about the axes shown in Table 1. The expressions all take the form

$$I = f_I m R^2, \quad (31)$$

where m is the object's mass, R is its radius, and f_I is some number, which we call the **geometric factor**. It depends on how the object's mass is distributed around the axis of rotation. We define the geometric factor as

$$f_I \equiv \frac{I}{m R^2}. \quad (32)$$

For example, the solid sphere has a f_I of $2/5$. f_I does not depend on the size or radius of the object. Both a small and large solid sphere has $f_I = 2/5$. On the other hand, different shapes do have different values of f_I . Notice that f_I gets larger as more of the object's mass is placed further away from the axis of rotation. It reaches its maximum value of 1 for a thin-walled cylinder. An equation like Equation 31 applies to other simple objects as well, though R might be replaced by some other relevant length, such as the length L of the thin rod in Table 1.

Problem 8.2 How can you change a cylindrical object's moment of inertia without changing its mass or its radius?

The message here is that, in rotational motion, how far something is from the axis of rotation matters, whether that thing is a force or a bit of mass. Moment of inertia makes this adjustment for mass, and the concept of *torque* does the same for force.

Theoretical Background: Angular Momentum

So far we have investigated two quantities in rotational motion that have analogues in linear motion: moment of inertia (mass) and rotational kinetic energy (translational kinetic energy). Here we will investigate another quantity, momentum, that takes a similar form in both translational and rotational motion.

Recall that the linear momentum of an object of mass m can be written as

$$\vec{p} = m\vec{v},$$

where \vec{v} is the linear velocity of the object. It is natural, therefore, to define a similar quantity for rotational motion called the **angular momentum**, defined as

$$\vec{L} = I\vec{\omega}, \quad (33)$$

where \vec{L} is the angular momentum.

Angular momentum, like linear momentum, obeys a conservation principle. The **Law of Conservation of Angular Momentum** states that, in the absence of a net external torque, the total angular momentum of a system will remain constant.

A comparison of all of the equations for linear and rotational motion is shown in Table 1.

	Translational/Linear	Rotational/Angular
Kinetic energy	$\frac{1}{2}mv^2$	$\frac{1}{2}I\omega^2$
Momentum	$\vec{p} = m\vec{v}$	$\vec{L} = I\vec{\omega}$
Acceleration	$\vec{a} = \frac{\Delta\vec{v}}{\Delta t}$	$\vec{\alpha} = \frac{\Delta\vec{\omega}}{\Delta t}$
Newton's 2nd Law	$\vec{F} = m\vec{a} = m\frac{\Delta\vec{v}}{\Delta t}$	$\vec{\tau} = I\vec{\alpha} = I\frac{\Delta\vec{\omega}}{\Delta t}$

Table 2: A comparison of some of the equations for translational and rotational motion.

Experimental Background: The Inclined Plane and Photogates

Now we discuss the inclined plane, down which we will roll a variety of objects. The first goal of this laboratory is to verify that energy conservation holds in situations where rotational kinetic energy is included. We will then use this fact to measure some unknown moments of inertia. To do this, we will need the theoretical tools we just discussed (moment of inertia and rotational kinetic energy) as well as one from a previous laboratory (conservation of energy). A photograph of the experimental setup is shown in Figure 1.

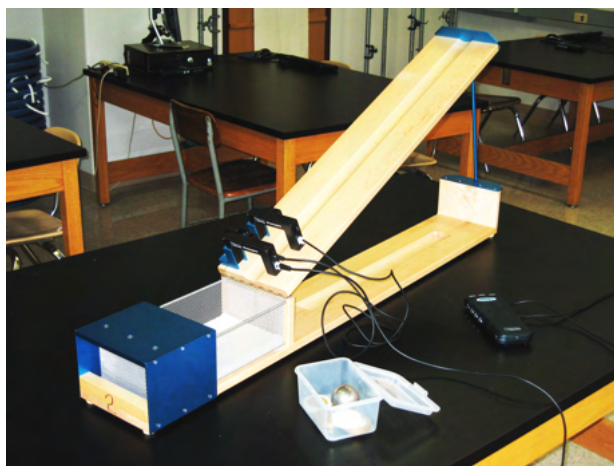


Figure 1: Photograph of the experimental setup for the inclined plane laboratory.

The two **photogates** at the bottom of the track work a bit like the safety mechanism of an automatic garage door opener. There is a laser beam that passes from one side of the photogate to the other, and if the beam is broken, the computer records the time at which this occurred. Here, two photogates are used in series, and *LoggerPro* can automatically detect the amount of time that elapses between an object breaking the beam of the first photogate and the object breaking the beam of the second photogate. This elapsed time, combined with a knowledge of how far apart the photogates are, allows us to determine the object's average velocity in the region between the photogates. For the purposes of this laboratory, we will assume that this average velocity is actually the object's instantaneous velocity at the midpoint of the two photogates, a slightly crude approximation.

Consider a spherical or cylindrical object of mass m and radius R sitting (at rest) at the raised end of a plane of length d which is inclined at an angle ϕ (see Figure 2). There are two ways in which this object can move to the bottom of the plane.

Way #1 The object *slides*. This is the situation we have encountered so far in energy conservation problems. It is rather straightforward: the object loses a certain amount of potential energy by sliding to the bottom, and that potential energy is converted into translational kinetic energy. Thus,

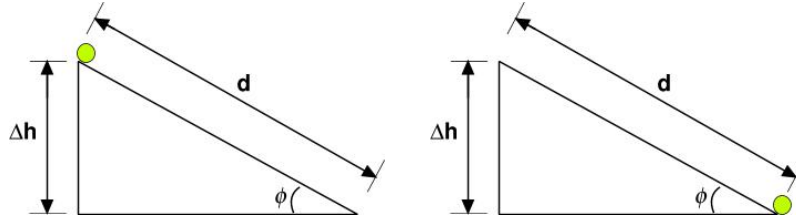


Figure 2: The two object positions we are concerned with for conservation of energy calculations.

change in potential energy = final kinetic energy

$$mg\Delta h = \frac{1}{2}mv_t^2.$$

Way #2 The object *rolls*. This is slightly more complicated. If the object also rolls, the object gains angular velocity *omega* at the bottom. The potential energy that is changed by moving to the bottom is still converted into kinetic energy, but now there are two types of kinetic energy: translational and rotational. Thus,

$$mg\Delta h = \frac{1}{2}mv_t^2 + \frac{1}{2}I\omega^2. \quad (34)$$

We can simplify this equation even further if we assume that the object **rolls without slipping**, so that $\omega = v_t/R$. Then we have

$$\begin{aligned} mg\Delta h &= \frac{1}{2}v_t^2 (m + I/R^2) \\ &= \frac{1}{2}mv_t^2 (1 + f_I), \end{aligned}$$

where f_I is the geometric factor from above.

We say an object **rolls with slipping**, when it almost always rolls down the plane, but at certain instants, the object slips, causing it to slide. This means that the slipping object's translational velocity is no longer fixed to be ωr . In fact for an object rolling with slipping,

$$v_t \geq \omega r \Rightarrow \omega \leq \frac{v_t}{r} \quad (35)$$

Experimental Background: Rotation and Static Friction

Although this chapter focuses on rotational energy, it is important to note that **static friction** is actually hiding in Equation 34. When the object is not slipping, static friction causes the object to rotate. Remember, static friction is the force of that an object must overcome so that it can “break free” and start sliding. Because static friction does not appear in the two equations above, a common misconception is that static frictional force does not do work (i.e contribution to energy conservation) because a rotating object’s displacement is zero. Actually, static friction does do work, both rotational, and translational, but they cancel each other because they are equal in magnitude and opposite in sign. The rotational work W_{rot} done by static frictional force F_{st} can be calculated by the torque τ sweeping a total angle ψ such that

$$W_{rot} = \tau\psi = (Rf)\psi = F_{st}s,$$

where the total angle ψ is equal to s/R (distance/radius). On the other hand, the translational work done by static frictional force moving a distance s is

$$W_{trans} = -F_{st}s.$$

Because static friction does work, some potential energy is now converted to static frictional work. Therefore, the change in potential energy is equal to the kinetic energy, and static frictional work. Thus,

change in potential energy = final kinetic energy + static frictional work

$$\begin{aligned} mg\Delta h &= \frac{1}{2}mv_t^2 + \frac{1}{2}I\omega^2 + W_{rot} + W_{trans} \\ &= \frac{1}{2}mv_t^2 + \frac{1}{2}I\omega^2 \end{aligned}$$

The rotational and translational work cancel out because they are equal in magnitude but opposite sign. Therefore, this equation is identical to Equation 34.

Problem 8.3 If the object is partially slipping while it is rolling down the inclined plane, would the angular velocity be larger, smaller, or same?

Problem 8.4 How is velocity measured in this laboratory? Why is our measurement of velocity considered to be the velocity at the midpoint of the two photogates rather than at the second photogate? *Hint:* Consider the definition of “average velocity.”

Experiments

Experiment 1: Rotational Kinetic Energy

In the first experiment, we will verify energy conservation for a rolling object as it travels down an inclined plane.

Object	Radius (cm)
Steel ball	2.540
Solid brass disk	2.540
Solid nylon disk	2.540
Disk (brass rim, nylon center)	2.515
Disk (nylon rim, brass center)	2.515

Table 3: Outer radii (R) of the objects used with the inclined plane.

1. Type the number on the back plate of your inclined plane into the provided code cell in your Jupyter notebook. A value for Δx , the distance between the two photogates, will appear. Notice that the length D of the inclined plane (89 cm) is already recorded in your template.

Problem 8.5 Convert the given value of the angle of inclination ϕ of the plane (25°) into radians.

2. Weigh the steel ball and record its mass. Be sure to keep track of units.
3. Use the meter scale for a rough estimate of Δh , the vertical distance the ball travels from the top of the plane to the midpoint of the two photogates.

Problem 8.6 Now calculate Δh using the expression for d in Figure 3 ($d = D - R - \Delta x/2$). Your calculated answer should be similar to your rough measurement.

4. Calculate the amount of potential energy that the steel ball loses as it travels from the top of the inclined plane to the midpoint of the two photogates.
5. Open the *LoggerPro* template file *inclined_plane.cmb*. This will open *LoggerPro* and set it up to record the time delay between the object's passage through one sensor and the other.
6. Hold the steel ball carefully against the aluminum back plate at the top of the inclined plane, centering it in the groove.
7. **Note:** The software is setup to take 4 trials for each object. In the following steps you will perform those four trials for each object, but **you do not need to start and stop the data acquisition** for each trial. The acquisition will stop automatically.

8. Start the data acquisition. Wait a few seconds for the software to initialize the photogates. Release the ball cleanly so that it does not wobble on its way down the track. If LoggerPro does not record any data, check that the photogate has been turned on.
9. Repeat the rolling procedure three more times. *LoggerPro* should be recording the times it takes the steel ball to roll between the two photogates, and data collection will stop automatically after four trials.
10. Copy these four individual measurements into your lab report.
11. Repeat this procedure for the two solid disks (brass and nylon).

Problem 8.7 For each object, separately calculate the mean Δt of the four trials. Then, using the approximation $v = \Delta x / \Delta t$, calculate the translational velocity of that object at the midpoint of the two photogates.

Problem 8.8 Use your measurements of v and the objects' radii (see Table 3) to calculate ω for each object at the bottom of the plane, assuming that no slipping has occurred.

Problem 8.9 Calculate each object's moment of inertia using Table 1. Also calculate experimental values for the objects' translational, rotational, and total kinetic energies at the midpoint of the two photogates.

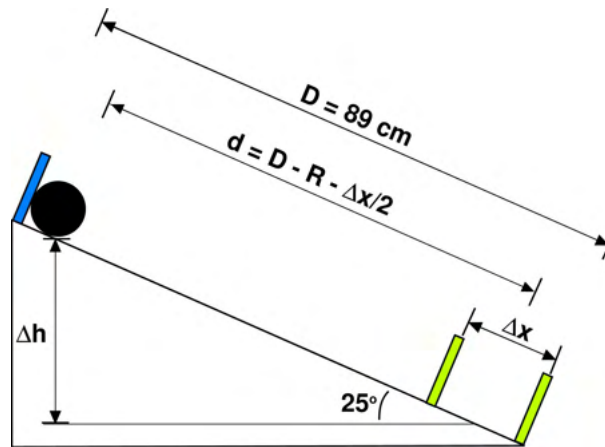


Figure 3: Diagram of the experimental setup. In this diagram, R is the radius of the rolling object. Note that you can find Δh easily if you know d and ϕ , since $\Delta h = d \sin \phi$.

Problem 8.10 (a) Name the forces that act on the rolling object (even if they don't do work). (b) Explain why (each of) the forces the object experiences does or does not do work.

Problem 8.11 Calculate the change in potential energy for each object rolling from the top of the plane to the midpoint of the photogates (using the measured Δh). Compare the final total KE with the change in PE for each of the rolling objects. Are the final KEs generally greater or less than the corresponding PEs? What could cause this effect?

Problem 8.12 How would the following changes (relative to normal experimental conditions) affect the steel ball's translational kinetic energy at the bottom of the incline? Write a one-sentence explanation in each case.

- (a) The ball slips as it rolls down the plane, and the energy losses due to friction are negligible.
- (b) The ball is hollow instead of solid, but the total mass remains unchanged.
- (c) The angle of inclination of the plane is less than 25° .
- (d) The coefficient of static friction between the plane's surface and the ball is increased. (The ball was rolling without slipping before this change happened.)
- (e) The mass of the ball is increased, but its shape and size are unchanged.

Experiment 2: Measuring the Geometric Factor

From Experiment 1, we verified energy conservation (or work-kinetic energy theorem) holds for rotating objects. Now we will use this to compare the motion of objects with the same mass but different geometric factors. The mass distributions of these objects are more complicated than those of the solid ball and disks from Experiment 1. Here we use a procedure similar to that of Experiment 1, but assuming that energy conservation holds in order to measure the geometric factors of two disks with the same mass but more complicated geometries. To do this, we will repeat the data acquisition procedure from Experiment 1 for the two composite disks (one has a brass rim, and the other has a nylon rim).

1. Record the masses of the two disks.

Problem 8.13 Calculate the amount of potential energy change by the disks as they move from the top of the plane to the midpoint of the two photogates (again using the measured Δh).

2. Collect four measurements of Δt for each of the two disks and copy them into your lab report.

Problem 8.14 Calculate the mean Δt for each disk. Then, using the approximation $v = \Delta x / \Delta t$, calculate the translational velocities of the disks at the midpoint of the two photogates.

Problem 8.15 Using your measurement of v and the radii of the disks (see Table 3), calculate ω for the disks at the bottom of the plane, assuming that no slipping has occurred.

Problem 8.16 Calculate the translational kinetic energies of the disks at the midpoint of the two photogates. Then, assuming that conservation of energy holds, use your knowledge of the change in potential energy and the translational kinetic energy to obtain experimental measurements of the rotational kinetic energies and moments of inertia of the disks.

Problem 8.17 Use the moments of inertia, masses and radii of the disks to obtain experimental values for the geometric factors of the two disks.

Experiment 3: Model Fitting and Statistics

1. Using the online data collection tool, record your values for the geometric factors of the two composite disks (see Appendix A Sections 2.1-2.3). Enter the ratios in the following order: Brass rim, nylon center, then the nylon rim, brass center.
2. When all of the class data has been collected, follow the instructions provided in the Jupyter notebook to download and save the class data as a '.csv' file so that it can be imported into Jupyter notebook.
3. Once the data is imported into Jupyter notebook, use the provided code cell to calculate the mean, standard deviation, error of the mean, and 95% confidence interval for the geometric factors of both disks.

Problem 8.18 Which composite disk has the greater geometric factor? Why does this make sense?

Problem 8.19 In your own words, explain why a rod rotating about its end has a different moment of inertia than the same rod rotating about its midpoint (see Table 1 for an illustration of this).

Problem 8.20 Suppose you are given two balls, one hollow and one solid, but both with the same mass and radius. Using the same experimental setup as today's Lab (inclined plane and photogates), outline a list of steps one can follow to determine which ball is hollow and which one is solid. Explain your reasoning.

Laboratory 9

Rotational Motion: The Gyroscope

Introduction

A **gyroscope** is a device used for measuring or maintaining orientation. Its characteristic motion, while somewhat nonintuitive to the casual observer, is actually a very straightforward example of the principle of angular momentum conservation. In its most basic form, a gyroscope consists of a spinning wheel or disk whose axle is mounted on pivoted supports (“gimbals”) such that the gyroscope’s axis of rotation can take on any orientation with respect to the surface it is mounted on.

For reasons we will investigate in a moment, the orientation of a spinning gyroscope’s rotational axis changes much less in response to a given external torque than it would if the gyroscope were not spinning. In short, that means that a spinning gyroscope is very stable; you can push and pull it in ways that would cause a stationary gyroscope to fall over, but it will not fall. Instead, it will undergo a type of motion called *precession*. A spinning bicycle wheel, a spinning top, and even the Earth itself exhibit this unusual precessive motion.

Theoretical Background: Force vs. Torque

Simply put, *force* is something that can change the velocity of a body. Similarly, *torque* is something that can change the *angular* velocity of a body. Even though forces are required to generate torques, it is important to remember that torque is *not* a force. Torques and forces are both vectors, so torques have both a *magnitude* and a *direction*. Using vector algebra, we find torque using something called the **cross product**, written as

$$\vec{\tau} = \vec{r} \times \vec{F} , \quad (36)$$

where $\vec{\tau}$ is the torque vector.

The *magnitude* of a torque experienced by a rigid object about a particular axis of rotation can be calculated using

$$\tau = r F \sin(\theta) , \quad (37)$$

where τ is the magnitude of the torque, F is the magnitude of the applied force, and r is the magnitude of the vector \vec{r} (also known as the *lever arm*) that points from the axis of rotation to the point at which

Eq. 36 shows that a torque's *direction* is completely determined by the directions of \vec{r} and \vec{F} . You can find the *direction* of $\vec{\tau}$ using something called the **right-hand rule**. To use the right-hand rule, you must lay your right hand flat so that your fingers first (i) point in the direction of \vec{r} , and then (ii) can curl to point in the same direction as \vec{F} . If you do this correctly, your outstretched thumb will point in the direction of $\vec{\tau}$. Figure 1 contains 4-step instructions on how to properly employ the right hand rule as described above.

Problem 9.1 If \vec{r} points in the positive x -direction, and \vec{F} points in the negative y -direction, in which direction does the torque vector point? What if \vec{r} points in the positive y -direction and \vec{F} points in the negative x -direction?

Problem 9.2 Find the magnitude and direction of the torque for each of the following situations, where the force, \vec{F} , has magnitude F , and the radius vector, \vec{r} , has magnitude r .



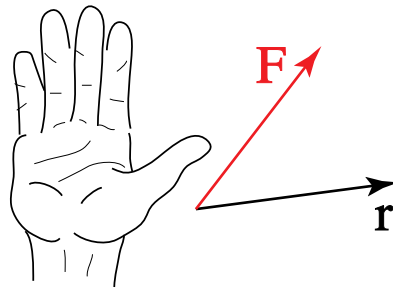
Torque is important in rotational motion because it is the rotational analogue of force (but it is not a force). In translational motion, a force is needed to make an object accelerate, and the amount of acceleration a given force can produce is inversely proportional to the object's mass. Rotational motion is similar. A torque is needed to create an **angular acceleration**, and the amount of angular acceleration a given torque can produce is inversely proportional to the object's moment of inertia. Compare the equation

$$\vec{\tau} = I \vec{\alpha} \quad (38)$$

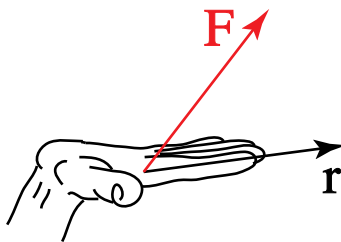
to Newton's second law for translational motion, $\vec{F} = m\vec{a}$, and you will get the idea.

Theoretical Background: Angular Momentum

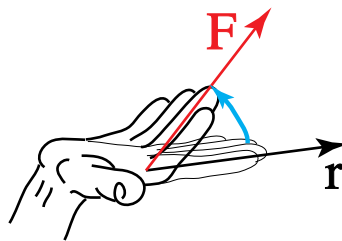
So far we have investigated two quantities in rotational motion that have analogues in linear motion: torque (force) and rotational kinetic energy (translational kinetic energy). Here we will investigate another quantity, momentum, that takes a similar form in both translational and rotational motion.



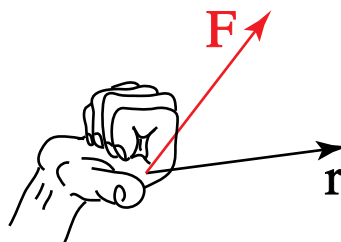
1 Right hand flat, Thumb extended



2 Point fingers in direction of r



3 Curl Fingers into direction of F



4 Thumb points in direction of τ
(here, out of the page)

Figure 1: The Right Hand Rule : A four step method for determining torque's direction.

Recall that Newton's second law can also be written as

$$\vec{F} = m\vec{a} = \frac{\Delta\vec{p}}{\Delta t},$$

where \vec{p} is the linear momentum. It is natural, therefore, to define a similar quantity for rotational motion called the **angular momentum**, defined as

$$\vec{L} = I\vec{\omega}, \quad (39)$$

where \vec{L} is the angular momentum. We can also rewrite Newton's second law for rotational motion as

$$\vec{\tau} = I\vec{\alpha} = \frac{\Delta\vec{L}}{\Delta t}.$$

In addition, just as force and torque are related through the equation $\vec{\tau} = \vec{r} \times \vec{F}$, linear and angular momentum are related via

$$\vec{L} = \vec{r} \times \vec{p}. \quad (40)$$

This means that a particle with a certain linear momentum also has an angular momentum about any point it is not moving directly toward or away from. (If \vec{r} and \vec{p} are parallel, \vec{L} is zero.)

Angular momentum, like linear momentum, obeys a conservation principle. The **Law of Conservation of Angular Momentum** states that, in the absence of a net external torque, the total angular momentum of a system will remain constant. If there is a net external torque, the change in \vec{L} is always in the direction of the torque.

Problem 9.3 In Laboratory 8, was the torque experienced by the rolling objects constant?

A comparison of all of the equations for linear and rotational motion is shown in Table 1.

	Translational/Linear	Rotational/Angular
Kinetic energy	$\frac{1}{2}mv^2$	$\frac{1}{2}I\omega^2$
Momentum	$\vec{p} = m\vec{v}$	$\vec{L} = I\vec{\omega}$
Acceleration	$\vec{a} = \frac{\Delta\vec{v}}{\Delta t}$	$\vec{\alpha} = \frac{\Delta\vec{\omega}}{\Delta t}$
Newton's 2nd Law	$\vec{F} = m\vec{a} = m\frac{\Delta\vec{v}}{\Delta t}$	$\vec{\tau} = I\vec{\alpha} = I\frac{\Delta\vec{\omega}}{\Delta t}$

Table 1: A comparison of some of the equations for translational and rotational motion.

Theoretical Background: Angular vs. Linear Momentum

From our study of linear momentum, we know that if the force and momentum vectors point in the same direction, the object speeds up or slows down, but its direction of motion does not change. Conversely, if the force and momentum vectors are perpendicular (uniform circular motion is an example), the object's direction changes but its speed stays constant. See Figure 2 for an illustration of this.

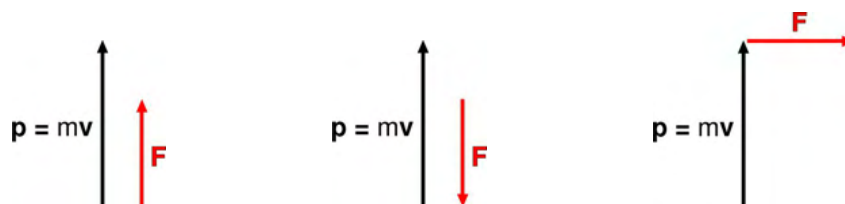


Figure 2: An illustration of how force can cause changes in linear momentum. If the force is parallel to the linear momentum vector, the object will speed up (left) or slow down (middle). If the force is perpendicular to the linear momentum vector, the object's direction of motion will change, but its speed will remain constant (right). The idea is the same for rotational motion: if the torque is parallel to the angular momentum vector, the object's rotational speed will increase or decrease. If the torque is perpendicular to the angular momentum vector, the object's rotational speed will stay the same, but its angular momentum vector will change direction.

We can apply this same idea to our study of rotational motion. In rotational motion, if the torque and angular momentum vectors are parallel to each other, the object's angular velocity will increase or decrease. We looked at a situation like this in Lab 8. In this lab, we will study a situation where the torque is perpendicular to the angular momentum vector. Just as with linear momentum, this causes the angular momentum vector to change direction without changing in magnitude.

Theoretical Background: Precession

Precession is a term used to describe the type of motion that occurs when a rotating object experiences a torque that is perpendicular to its angular momentum vector. In this laboratory, we will use a gyroscope to investigate precession. The trickiest part of this experiment is that it includes motion in all three dimensions rather than just two (the inclined plane was actually two-dimensional systems because all of the motion happened in a single plane). In three-dimensional rotational motion, we consider a system's rotation with respect to a **center of rotation** or **pivot point**, not an axis of rotation as we have seen previously.

As a simple example of a gyroscope, consider a spinning top. For simplicity, we will assume that our top is constructed out of a solid, circular disk of material of mass m and radius R and a stick

that passes through its middle. For all of our analysis, we will consider the angular momentum of the system about a pivot point that we place at the point where the top touches the ground, as shown in Figure 3.

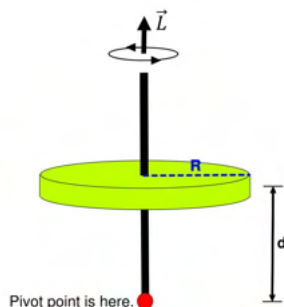


Figure 3: Example of a simple gyroscope. The pivot point is shown in red, and the distance d from the pivot point to the gyroscope's center of mass is labeled in the figure. The gyroscope's angular velocity $\vec{\omega}$ and angular momentum vector \vec{L} points upward in this diagram.

When the top shown in Figure 3 is stationary, the initial angular momentum about the pivot point is zero. If it is spinning, the initial angular momentum about the pivot point is equal to the moment of inertia I_{top} of the top about its axis of rotation times the top's angular velocity $\vec{\omega}$.

Problem 9.4 What two forces does the top shown in Figure 3 experience when it is upright, regardless of whether or not it is spinning? Do either of these forces generate torques about the pivot point? Why or why not?

Now consider what happens when we tilt the top when it is stationary vs. spinning, as shown in Figure 4. When it is stationary, the top just falls over, as we have all experienced. But when it is spinning, something else happens. Recall that

$$\vec{\tau} = \vec{r} \times \vec{F} = mgd \sin \theta$$

when the top is tilted an angle θ from the vertical, since its weight generates a torque about the pivot point.

Problem 9.5 In which direction does the torque vector $\vec{\tau}$ point in Figure 4?

Problem 9.6 How do we expect the angular momentum vector \vec{L} to change in the case of the spinning top? Will it increase in magnitude? Why or why not? Will it change direction? If so, how will the direction change?



Figure 4: Tops slightly tilted; the top on the left is stationary, while the top on the right is spinning.

In the above questions, you should have found that in the case of the spinning top, the tip of the angular momentum vector \vec{L} will move in the direction of the torque, that is, into the page. In fact, because the pivot point is fixed, the axis of rotation of the spinning top will *precess* in a counterclockwise direction when viewed from above.

The situation we will encounter in this laboratory is similar; however, the pivot point will actually be slightly elevated so that the axis of rotation of the gyroscope will be *perpendicular* to the force; this will simplify our calculations by omitting the $\sin\theta$ term. A diagram of our experimental setup can be found in Figure 5.

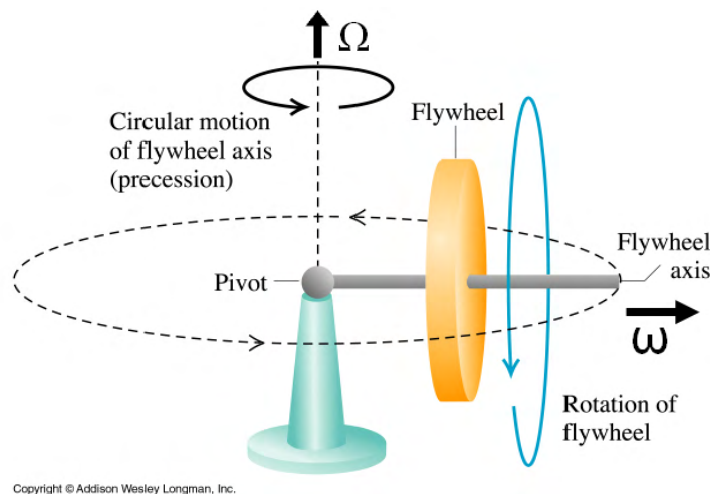


Figure 5: The direction of the angular momentum vector \vec{L} is to the right in this picture, and the torque vector $\vec{\tau}$ is pointing into the page, perpendicular to the flywheel axis (Fig. 10.29, p. 315 from *University Physics* (10th ed.) by Hugh D. Young and Roger A. Freedman. Copyright 2000 by Addison Wesley Longman, Inc. Reprinted by permission of Pearson Education, Inc.)

The **angular precession speed** of a gyroscope can be derived quite simply by considering the magnitude of the torque and its relationship to the rate of change of angular momentum. Let Ω be the angular precession speed, and let ω be the angular speed of the gyroscope spinning about its axis. Note that

$$\vec{\tau} = \frac{\Delta \vec{L}}{\Delta t},$$

where in this case the magnitude L of \vec{L} is not changing, but the rate of change of its direction is equal to Ω , so

$$\left| \frac{\Delta \vec{L}}{\Delta t} \right| = L \Omega.$$

Solving for Ω , we obtain

$$\begin{aligned} \Omega &= \frac{\tau}{L} \\ &= \frac{\tau}{I \omega}. \end{aligned} \tag{41}$$

Note that $\vec{\Omega}$ is a vector, just like $\vec{\omega}$. If you curl your fingers of your right hand along the gyroscope's path of precession, your thumb will point in the direction of $\vec{\Omega}$.

Problem 9.7 What are the directions of \vec{L} , $\vec{\tau}$, and $\vec{\Omega}$ in Figure 5?

Experimental Background: The Gyroscope

The gyroscopes originally used in these laboratories were designed about 30 years ago by Professor Richard Crane, a physicist who made some exquisitely precise measurements of the gyromagnetic ratio of the electron, confirming our understanding of quantum electrodynamics. The sculpture on the south side of the courtyard between Randall and West Hall commemorates this important contribution to 20th-century physics.

The equipment we will use in this experiment has a more modern design with a few more safety considerations; however, some precautions are still in order. Please take note of the following:

1. Be careful with the gyroscopes. When fully spun up, the gyroscopes used in this laboratory store almost 500 J of kinetic energy, equivalent to a baseball moving at 180 mph. They are also somewhat heavy, so you do not want to drop them on your feet.
2. A small electric motor with a 1-inch-diameter felt wheel (Dremel tool) is provided to spin up the gyroscope to a high angular velocity. To spin up the gyroscope, turn on the motor and bring the edge of the felt wheel into contact with the rim of the rotor, as shown in Figure 6.



Figure 6: The technique for spinning up the gyroscope's rotor. Do not press the Dremel tool very hard! For best performance, hold the Dremel tool perpendicular to the gyroscope as shown.

3. Keep a very light, steady pressure on the felt wheel while spinning up the gyroscope. **Note that pressing too hard will burn the surface of the felt** (and will actually slow the rotor's spin).
4. Keep your eyes away from the line of contact between the felt wheel and the rotor rim. Small bits of the felt will flake off and may irritate your eyes. Because of this, *all* group members *must* wear safety goggles when spinning up the rotor.
5. It will take a couple of minutes to bring the rotor speed above 3200 RPM, which is the speed we will need for these experiments.
6. If you notice that the diameter of your felt disk is significantly less than 1 inch or that it exhibits significant burn marks around its edge, notify your instructor.
7. To stop the gyroscope from spinning, gently touch a finger (perhaps covered by a piece of paper towel) to the rotor rim to bring it slowly to a halt.

There is a piece of reflective tape attached to one side of the gyroscope rotor. By aiming the beam of a small **photo tachometer** at this tape while the gyroscope is spinning, you can measure its angular speed in RPM. An illustration of this technique is shown in Figure 7.

The tachometer should be held a few inches from the rotor so that it illuminates the reflective strip as shown in Figure 7. Aim the beam of light at an angle of 30° or so from the normal to the rotor surface. This enhances the contrast between the light reflected from the reflective strip and stray light from the rest of the rotor surface. The angular velocity in RPM is reported by the display.

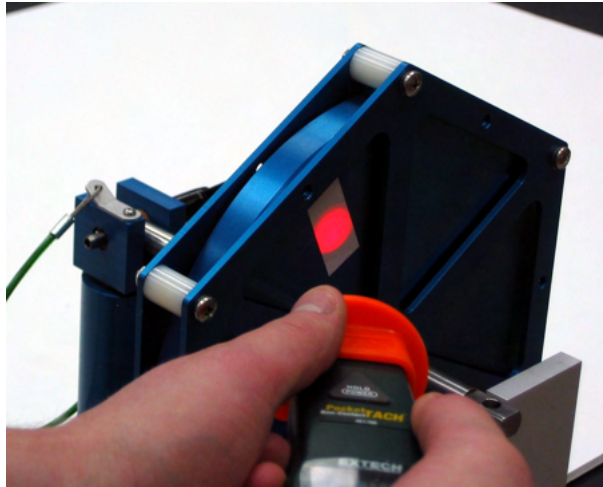


Figure 7: Hold the photo tachometer so that the red LED illuminates the reflective strip on the side of the rotor. Try to keep the beam of light at an angle of 30° from the rotor normal.

Experiments

Experiment 1: Angular Momentum and the Spinning Chair

Note: There is only one setup for this experiment in each classroom. If the chair is in use, please move on to the next part until it is available. You are not required to do this before moving on to the rest of the lab, but should complete it before the end of class. In this experiment, you will conduct a tactile demonstration of angular momentum conservation by holding a spinning gyroscope while sitting in a swivel chair (see Figures 8 and 9) and then flipping the gyroscope over so that the direction of \vec{L} is reversed.

1. Spin up a heavy gyroscope to above 3200 RPM using the Dremel tool, which your instructor will demonstrate. You will be able to tell when it is going fast enough because you will start to hear an ominous hum from the bearings.
2. Sit (or have your lab partner sit) on the swivel chair while holding the gyroscope shaft with both hands, as shown in Figure 9. Note the initial direction of \vec{L} for the gyroscope.
3. Quickly flip the gyroscope's axis of rotation by 180° . Note the final direction of \vec{L} for the gyroscope, and the direction of the new \vec{L} for the person sitting in the chair.



Figure 8: The swivel chair and stainless steel rotor used to demonstrate the conservation of angular momentum.

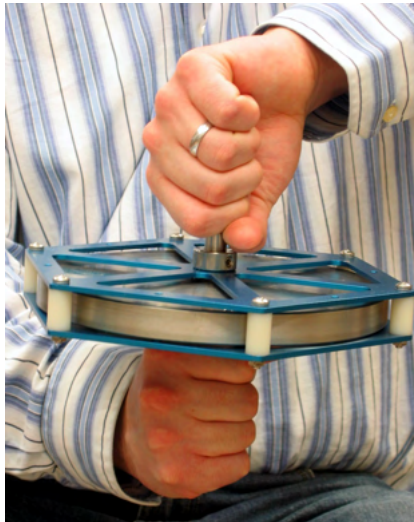


Figure 9: Initial position for holding the gyroscope while sitting in the swivel chair. With the rotor spinning rapidly, twist the axis 180° from up to down.

Problem 9.8 If the direction of the angular momentum vector, \vec{L} is initially upward, and you flip the gyroscope while sitting in the chair so that \vec{L} now points downward, does the chair spin clockwise or counter-clockwise as viewed from above? Describe why this happens using terms and concepts from angular momentum conservation.

Problem 9.9 If we were to increase the gyroscope rotor's mass but spin it up to the same initial speed, how would this affect the motion of the chair when the gyroscope is flipped? Why?

Problem 9.10 If we were to increase the mass of the person sitting in the chair, how would this affect the chair's motion when the gyroscope is flipped? Why?

Experiment 2: Quantitative Analysis of Precession Rate

In this experiment, we quantitatively test the prediction for the precession rate given by Eq. 41. Two separate angular velocities will need to be measured for this exercise: the rotor speed ω and the precession rate Ω . We will measure ω using the photo tachometer, and we will measure Ω using a photogate and the *LoggerPro* software.

1. Open the *LoggerPro* template file *gyroscope.cml*. This will open *LoggerPro* and set it up to record values of the precession rate Ω in rad/s.
2. Mount the gyroscope in the vertical stand as shown in Figure 10 by fixing the 5.00-inch end of the gyroscope shaft in the fork on the stand using the retainer pin.
3. Spin up the gyroscope to at least 3200 RPM using the Dremel tool.

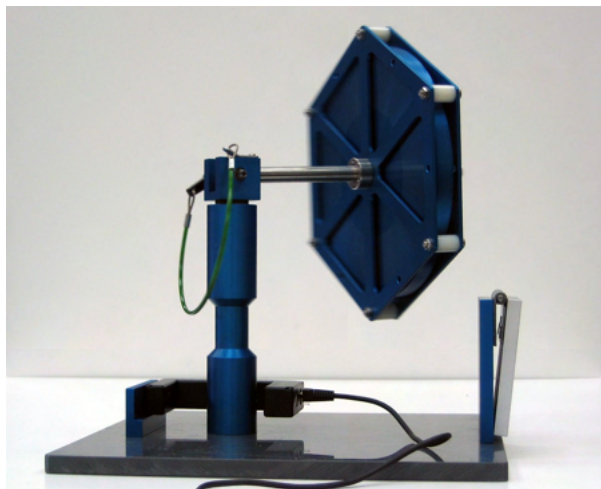


Figure 10: Release the spinning rotor assembly and the gyroscope will precess (rotate) about the vertical axis. The precession rate Ω is measured by the photogate at the base.

4. Measure the angular velocity of the rotor using the photo tachometer and record this value in your lab report.
5. Immediately after measuring the angular velocity, release the free end of the gyroscope from the stand and start taking data with *LoggerPro*. Let the gyroscope precess until *LoggerPro* takes a reading of the precession rate. Record this precession rate in the appropriate cell in your lab report. Note that *LoggerPro* will automatically stop collecting data after making *two* measurements of the precession rate. If the two values look similar, use the average value; if one of the values is clearly incorrect, ignore it and use the other.
6. Immediately after *LoggerPro* is finished taking data, carefully grab the gyroscope and take another measurement of the rotor's angular velocity, and record this value in your lab report. Use the average of the tachometer measurements from before and after the gyroscope precesses as your value of ω .
7. Spin up the gyroscope again and repeat this procedure for two more trials.
8. We will now measure the force exerted on the gyroscope by gravity. Stop the rotor from spinning. Use a spring scale attached to the end of the gyroscope's shaft opposite the pivot point to measure the upward force necessary to cancel the torque caused by the gyroscope's weight (lift until the shaft is perfectly horizontal).

Problem 9.11 Use this measured force due to gravity and its distance from the pivot point (7.50 inches) to calculate the torque acting on the gyroscope.

9. Type the number stamped on the surface of the rotor into the provided code cell in your Jupyter notebook. A value for the mass of the rotor m_{rotor} will appear. Notice that the radius R_{rotor} of the rotor (3.95 in) is already recorded in your template.

Problem 9.12 Compute the moment of inertia using the formula for a disk (refer to Table 1 of Lab 8 if needed.)

Problem 9.13 Calculate the angular momentum as well as the theoretical precession rate (using Eq. 41) for each trial. (Remember to use the average ω_{rotor})

Problem 9.14 In step 8 of the procedure, you measured torque caused by the gyroscope's weight by lifting up the end of the gyroscope (at 7.5 inches). If you measured at a different distance, say you attached the spring scale at 5 inches from the pivot point, would the spring scale read a different value? Would the torque be a different value? Does this explain why you do not need to measure the spring scale at the center of mass location, where the moment of gravitational force is located? (provide an argument as well as your measurement report.)

Problem 9.15 The precession rate given by Eq. 41 assumes that friction is negligible. If friction plays a role while the rotor is spinning (such that ω is constantly decreasing), will our observed precession rate be greater or less than what we expect? Why?

Problem 9.16 Let us assume that there is an additional torque associated with air resistance while the gyroscope is precessing. How would the additional torque influence the precession direction of the gyroscope?

Problem 9.17 If we reversed the spin direction of the rotor, would the gyroscope precess in the same or the opposite direction when it is released? Why?

Experiment 3: Direction of Precession about the Center of Mass

Now we investigate the direction of precession, rather than its magnitude. A gyroscope that is not fixed at any particular pivot point can still experience torque and still precess, but the precession will occur about its center of mass rather than a fixed pivot point.

1. Place the gyroscope in the cradle assembly (see Figure 11) such that it balances easily. Insert the retainer pins to ensure that the assembly does not loosen and drop the gyroscope on the floor.
2. Spin up the rotor to at least 3200 RPM. Use the photo tachometer to keep track of the rotor's speed as shown in Figure 7, although the exact angular speed is unimportant here.
3. Orient the gyroscope so that the angular momentum vector \vec{L} points to the left when you are facing the gyroscope (along the $-x$ -axis as shown in Figure 11).
4. In the next two problems, you will apply torques to the gyroscope assembly by exerting the specified forces. To exert each force, gently push or pull the end of the shaft with a single finger. You will feel the gyroscope beginning to precess. Observe the gyroscope's direction of precession in each case.

Problem 9.18 Using the coordinate system in Figure 11, what are the directions of \vec{F} , \vec{L} , $\vec{\tau}$, and $\vec{\Omega}$ when you are

- (a) Pushing the left side of the shaft down?
- (b) Pushing the left side of the shaft up?
- (c) Pulling the left side of the shaft toward you?
- (d) Pushing the left side of the shaft away from you?

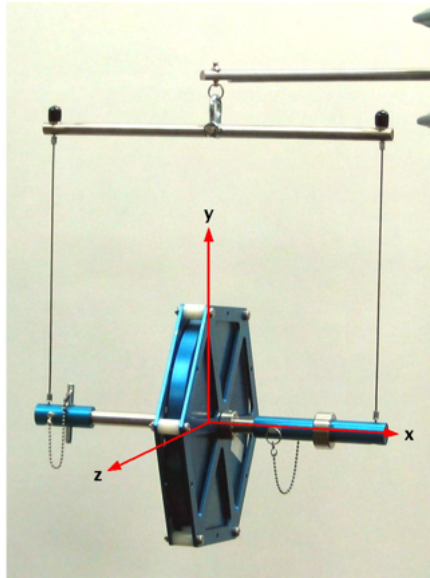


Figure 11: The gyroscope mounted in a torque-free cradle. Note that the longer end of the gyroscope shaft is inserted in the shorter blue hanger to properly balance the assembly. A coordinate system is included for reference.

5. Spin up the gyroscope again, but orient it such that its angular momentum vector points to the right when you are facing it (along the $+x$ -axis).

Problem 9.19 Repeat Problem 9.18 for the situation where \vec{L} initially points along the $+x$ -axis.

Problem 9.20 As you push or pull on the shaft in each of these situations, the gyroscope will begin to precess about its center of mass. Describe how you can use the directions of the \vec{L} and $\vec{\tau}$ vectors to predict the gyroscope's direction of precession (the direction of $\vec{\Omega}$).

Problem 9.21 If you are rounding a corner on a bicycle, should you lean inward or outward to keep from falling over? Why? Think of the front wheel of the bicycle as a gyroscope and ignore any other effects that may be relevant.

Laboratory 10

Modeling Gravitational Motion

Introduction

Welcome to the Modeling Gravitational Motion Lab, in which we'll model Earth orbiting the Sun by modifying a template program to include the physics-enabling lines of code you wrote while doing the Pre-Lab assignment. Then, we'll use our model to investigate the relationship between energy and orbital shape. Finally, we'll investigate the time evolution of Earth's kinetic and potential energy, and linear and angular momentum, during circular and elliptical orbits around the Sun before investigating a three-body system.

Modeling Gravitational Motion: Pre-Lab Assignment

To receive full credit for this Pre-Lab Assignment: complete the Jupyter notebook file found in the Files tab of the Canvas page. Once completed, download the notebook as an html file and submit the assignment to Canvas before coming to class.

In previous labs, we have already explored the nature of the Euler's Method and using it to simulate kinematic processes. In this Pre-Lab Assignment, we'll review the nature of the gravitational force as it acts on a large scale.

Box 1: Pre-lab Objectives

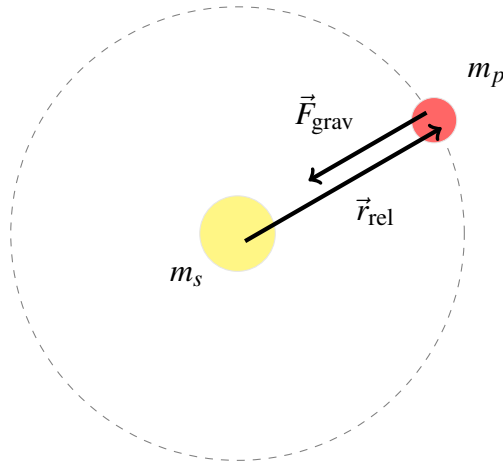
- ☐ Review Newton's law of universal gravitation.
- ☐ Derive properties of circular orbits and make predictions for real planets.
- ☐ Estimate the time scale of the simulation you'll do in lab.
- ☐ Translate Newton's law of universal gravitation from algebra into Python code.

The Universal Law of Gravitation

In previous experiments this semester, we treated gravity near Earth's surface as a constant force pointing “downward”. But that was only an approximation (albeit a very useful one). Once we consider motion over distances large compared to typical separations between gravitating bodies (e.g., the distance between Sun and Earth), the error introduced by using this constant-force approximation becomes very large. We need to use Newton's universal law of gravitation to get more accurate results. Recall the gravitational force experienced by a planet orbiting the Sun:

$$\vec{F}_{\text{grav}} = -G \frac{m_p m_s}{|\vec{r}_{\text{rel}}|^2} \hat{r}_{\text{rel}},$$

where $G = 6.67 \times 10^{-11} \text{ N m}^2/\text{kg}^2$ is the Gravitational Constant, m_p is the mass of the planet, m_s is the mass of the sun, \vec{r}_{rel} is their relative spatial separation, and \vec{F}_{grav} is the force of gravity exerted on the planet by the sun (see diagram below).



The $1/|\vec{r}_{\text{rel}}|^2$ factor (that is, the “inverse square” part) causes the magnitude of the gravitational force to change depending on how far apart the objects are. The $-\hat{r}_{\text{rel}}$ factor tells us that the direction of the force will also change, depending on where the objects are.

Note that due to Newton’s 3rd Law, there is also a force exerted by the planet on our Sun, having the same magnitude and opposite direction. We will ignore this force in our program, and treat the Sun as having a fixed location at the origin of our coordinate system. This approximation works well when the mass of the orbiting planet is very small relative to the Sun (for example, the Earth and Venus). Larger planets (such as Jupiter) exert a large enough force on the Sun to cause it to wobble perceptibly. For example, imagine whirling a rock on a string around your head. If the rock were a pebble, your hand would not feel a large tension force from the string, and would stay fairly stationary. If you were whirling, say, a medium-sized rock instead, your hand would experience a large force and would likely not be stationary. (Unless, of course, you are a professional rock whirler fully capable of whirling medium-sized rocks without any sign of wobbling, in which case we ask you to sympathize with the plight of our slightly-more-wobbly-prone compatriots.)

One way to understand this approximation is to imagine that there is some infinitely strong force preventing the Sun from wobbling, even a little, in any direction. Alternatively, you can imagine that we are viewing the system from a set of coordinates moving perfectly with the Sun, so that we always measure its position as constant and its velocity as zero. Either way you look at it, keep in mind that a more realistic model would take into account the motion of the Sun, and that our chosen method of modeling the system may impact the system’s total linear momentum.

Relative Position Vectors

The gravitational force pulls objects together. But “together” is a relative direction which depends on the locations of those objects. When we assemble our model of Earth orbiting the Sun in lab,

we'll update (\mathbf{r}) every cycle, so that it always points from the Sun's last known position to Earth's last known position.

Circular Motion

Maintaining circular motion requires that the body experience centripetal acceleration. This means that the sum of all the forces on the body add up to a force that points toward the center of the circle.³ The name “centripetal” tells us nothing about the nature of the forces creating this acceleration—they could be tension, normal forces, gravity, electromagnetism, or anything else (including a combination of forces).

Regardless (of course) of the source of the force, in order to keep an object of mass m moving precisely in a circle with speed v , the acceleration of the body must have the following magnitude:

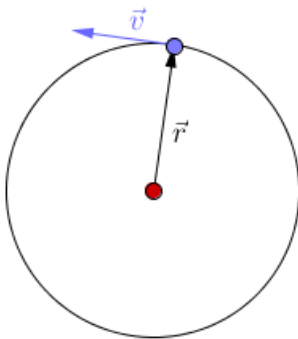
$$a_c = \frac{v^2}{r}.$$

For an orbiting planet, the gravitational force is the *only* force, and so must be the source of the centripetal acceleration. Thus, by Newton's Second Law we conclude that:

$$F_{\text{grav}} = m_p \frac{v_p^2}{r_{\text{rel}}}.$$

(Notice we are dealing only with magnitudes here.)

Use the following diagram of a blue object orbiting circularly a red object to answer questions based on the relative position:



Prelab Problem 10.1 In which direction does the gravitational force act?

Hint: you can use parallel or anti-parallel to \vec{r} or \vec{v} in your response.

³The word “centripetal” comes from the Latin *centrum* (“center”) and *petere* (“to seek”). As with most Latin-derived terms, it's pretty literal.

Prelab Problem 10.2 During its circular orbit, in a very small amount of time dt , a planet moves an infinitesimal amount $d\vec{s}$ in the same direction as \vec{v} in at that moment. What is the work done on the planet by the gravitational force during a circular orbit? Recall work done by gravity is calculated as $dW = \vec{F} \cdot d\vec{s}$.

Prelab Problem 10.3 Will the object's kinetic energy change as it moves in a circular orbit?

Prelab Problem 10.4 Use the equation for acceleration during circular motion $|F| = m|\vec{v}|^2/|\vec{r}|$ and the equation for gravitational acceleration to express the speed of an object in circular motion in terms of the large mass M (red in the GIF), the small mass m (blue in the diagram), the magnitude of the relative position $|\vec{r}|$, and Newton's constant G .

All About Time

This section addresses two questions:

1. How long do we want our simulation of Earth to run?
2. How small should the time step, Δt , be?

As we saw in previous labs, we needed to be careful in choosing these parameters when using Euler's Method for creating Numerical simulations. The first question relates to solving for the characteristic time and is the most straight forward to answer.

Prelab Problem 10.5 From the speed you found in Prelab Problem 10.4, find a characteristic time of orbital motion in terms of the same variables.
Hint: with a speed we can find a related time using $v = d/t$ and solve for t with the appropriate d (distance) and v (velocity/speed).

The second question for Δt (or dt as we define in the Python code, is more difficult. It depends on the speed the objects are moving at: the faster they move, the smaller Δt should be such that the objects don't move too far in a straight line that the estimation of circular motion (or as we will see in lab even elliptical) is accurate. To address this in lab, we mainly will deal with trial and error.

Modeling Gravitational Motion

Box 2: In this lab, you will:

- ☐ Simulate Earth's circular orbit.
- ☐ Change the initial conditions of the system to produce various orbital shapes, and investigate the relationship between energy and orbital shape.
- ☐ Investigate conservation laws.

Part One: Simulating Earth's Orbit

In this section, we will simulate the Earth in orbit around the Sun. This will follow a similar approach as in previous computational labs: we simply need to calculate the force acting on an object moving in two dimensions and update its acceleration, velocity, and position at discrete time steps. This same general approach allows us to model a wide variety of systems, by simply defining the appropriate force function.

To model a planet orbiting a sun, the relevant force is given by *Newton's law of universal gravitation*. This gives the gravitational force \vec{F}_{12} acting on mass m_1 due to mass m_2 and takes the form

$$\vec{F}_{12} = -G \frac{m_1 m_2}{r^2} \hat{r},$$

where r is the distance between m_1 and m_2 , \hat{r} is a unit vector pointing from m_2 to m_1 , and G is the *gravitational constant*.

Problem 10.1 In previous computational labs, your force function could depend on various parameters, such as position and velocity. To model the Earth orbiting the Sun, what parameters should your force function depend on?

After importing the necessary packages for this lab, we define a *class* called `ball` as we did in previous labs. We will create planets as instances of the `ball` class. Each instance of the `ball` class will have variables `r`, `v`, and `a`, for instantaneous values of position, velocity, and acceleration.

Likewise, the arrays `r_array`, `v_array`, and `a_array` store the *history* of position, velocity, and acceleration.

```
class ball:

    def __init__(self, m=1, r=np.array([0,0,0]), \
                v=np.array([0,0,0]), a=np.array([0,0,0])):

        self.m = m

        self.r = r
        self.v = v
        self.a = a

        self.r_array = np.array([])
        self.v_array = np.array([])
        self.a_array = np.array([])
```

Now we are ready to define our force function. When complete, this function will take two objects of the `ball` class (such as a planet and a sun) as input, and will return the force between them as output.

Code Task 10.1 Complete the force function defined in the Jupyter notebook by entering Newton’s law of universal gravitation in the location marked by replacing ‘?’ under #YOUR CODE HERE. Then run the cell.

Hints:

1. You can use the variables `G`, `ball_1.m`, `ball_2.m`, `r`, and `r_hat`.
2. You can calculate the dot product of two vectors `a` and `b` as `a.dot(b)`.

Next up, we need to initialize our system. We create `earth` and `sun` objects as instances of the `ball` class. We need to give the `earth` and `sun` properties such as mass, initial position, and initial velocity. We will place the `sun` at the origin and the `earth` on the $+x$ -axis, separated from the `sun` by a distance equal to the Earth’s mean orbital radius. For now, assume the initial velocity of the `sun` is zero. Let’s make the initial velocity of the `earth` point in the $+\hat{y}$ direction at some speed you get to choose! **To get you in the right ballpark, we recommend choosing an initial velocity between 12,000 m/s and 25,000 m/s.**

Code Task 10.2 Choose an initial velocity for the Earth and enter it in the cell provided in the Jupyter notebook. Then run the cell.

Now we need to split up time into discrete steps. We will simulate events happening over a total time T , using time steps of size dt .

Problem 10.2 What do you think would be appropriate values for T and dt ? Explain why you chose these particular values, and also note the values you chose. *Hint: A year is about 3×10^7 s.*

Code Task 10.3 Enter your values for T and dt in the cell provided in the Jupyter notebook. Then run the cell.

After defining the values for our force function, the time parameters of our simulation, and our objects `earth` and `sun`, we can now generate a code for simulating the orbit of the Earth around the Sun. The following code does just this:

```

i = 1
for t in times[1:]:

    # first update the accelerations
    earth.a = force(earth, sun)/earth.m
    sun.a = 0
    # then append the new value to the list
    earth.a_array[i] = earth.a
    sun.a_array[i] = sun.a

    # second update the velocities
    earth.v = earth.v + earth.a * dt
    sun.v = sun.v + sun.a * dt
    # then append the new value to the list
    earth.v_array[i] = earth.v
    sun.v_array[i] = sun.v

    # third update the positions
    earth.r = earth.r + earth.v * dt
    sun.r = sun.r + sun.v * dt
    # then append the new value to the list
    earth.r_array[i] = earth.r
    sun.r_array[i] = sun.r

    # update iteration count
    i = i + 1

```

Which we can plot the results of the simulation using the following code:

```

plt.plot(earth.r_array[:,0], earth.r_array[:,1], '.')
plt.plot(sun.r_array[:,0], sun.r_array[:,1], '.')
plt.axis('equal')
plt.show()

```

Take a look at the graph generated by the code provided above, and reassess your choices of simulation parameters T and dt . Does your Earth make at least one complete orbit? If not, you

may need a larger T . Does it trace the same path several times? Maybe make T smaller. Do the points on the planet's path seem too far apart? Or does the orbit not meet up in a closed loop? You may need to make dt smaller. Did the cells take too long to run, and the data points are so close together that the graph looks like a solid line? Maybe make dt larger.

Code Task 10.4 Go back to Code Task 10.3 and try different values for T and dt until you are satisfied. (Be sure to run all cells between Code Task 10.3 and here to see your results.)

Throughout this lab, we hope that you will ask yourself these questions to make sure you are optimizing your choices of T and dt for each simulation.

Problem 10.3 Look at the graph generated from your simulation and describe the orbit you have modeled. Is it circular? Elliptical? How does this differ from the Earth's orbit? Note that we chose realistic values for the masses and orbital radius, but *not* the Earth's initial velocity.

Next, we will animate the orbit by rendering a video of all our simulated data points. (We do not expect you to dig into *how* the animation code works.)

```
def init():
    line.set_data([], [])
    return(line)

def animate_ball_list(ball_list):
    def animation_function(i):
        x = []
        y = []
        for j in range(0, len(ball_list)):
            x.append(ball_list[j].r_array[i, 0])
            y.append(ball_list[j].r_array[i, 1])
        line.set_data(x, y)
        return (line)
    return animation_function
```

```

fig3, axs3 = plt.subplots()
axs3.set_xlim((np.min(earth.r_array[:,0]) - \
0.1*np.abs(np.min(earth.r_array[:,0])),
              np.max(earth.r_array[:,0]) + \
              0.1*np.abs(np.max(earth.r_array[:,0]))))
axs3.set_ylim((np.min(earth.r_array[:,1]) - \
0.1*np.abs(np.min(earth.r_array[:,1])),
              np.max(earth.r_array[:,1]) + \
              0.1*np.abs(np.max(earth.r_array[:,1]))))
)
axs3.set_aspect('equal')
axs3.plot([-1,1], [0,0], color='C1', lw=2)
line, = axs3.plot([], [], '.', markersize=24, lw=2)

plt.close()

anim_2 = animation.FuncAnimation(fig3, \
animate.ball_list([sun, earth]), init_func=init, \
frames=N, interval=40)
anim_2

```

Problem 10.4 For an elliptical orbit, discuss the velocity of the Earth when it is near the Sun (perihelion) versus far from the Sun (aphelion). What do you notice? If you did not run the animation, you can find the index of the earth array at its max and min to find the velocity at those points using the following code.


```

rMag = np.sqrt(earth.r_array[:,0]**2 + \
earth.r_array[:,1]**2)
rMaxIndex = np.argmax(rMag)
rMinIndex = np.argmin(rMag)

vMag = np.sqrt(earth.v_array[:,0]**2 + \
earth.v_array[:,1]**2)

print('At its maximum distance, the speed \
of earth is {:.3f} m/s'.format(vMag[rMaxIndex]))
print('At its minimum distance, the speed \
of earth is {:.3f} m/s'.format(vMag[rMinIndex]))

```

Problem 10.5 Why is the velocity larger or smaller at aphelion (furthest point from the sun) vs perihelion (closest point to the sun)? *Hint: Think about whether or not energy is conserved and what happens to the kinetic and potential energy at these points.*

Part Two: Circular Orbit

You might be looking at the orbit above and wondering why it's so elliptical. If we actually lived on a planet in that orbit we would burn up once a year. The remedy lies in the initial conditions. Next we're going to play with the initial conditions to try and make a circular orbit.

Playing with this means compressing a lot of the steps above so that we can see everything at once. For example, we can compress all the lines adjusting the planets' accelerations, velocities, and positions into an update function (much like how we compressed all the calculations for the gravitational force into a force function). Similarly, we want to move all the tedious lines of code setting initial conditions into an initialization function.

Read and understand what is happening in the lines of code provided below before continuing on with the lab.

```
def initialize_planet(planet, N):
    '''
    This takes in a planet and initializes it with
    empty arrays for the position, velocity, and
    acceleration histories. N denotes the total number
    of time steps these histories will contain.
    '''

    planet.a_array = np.empty((N,3))
    planet.a_array[0] = planet.a
    planet.v_array = np.empty((N,3))
    planet.v_array[0] = planet.v
    planet.r_array = np.empty((N,3))
    planet.r_array[0] = planet.r
```

```
def update_planet(planet, force_felt, dt):
    '''
    This takes in a planet and the force it felt and
    updates the planet by one time step.
    '''

    # first update the acceleration
    planet.a = force_felt/planet.m
    # then append the new value to the list
    planet.a_array[i] = planet.a
    # second update the velocities
    planet.v = planet.v + planet.a * dt
    # then append the new value to the list
    planet.v_array[i] = planet.v
    # third update the positions
    planet.r = planet.r + planet.v * dt
    # then append the new value to the list
    planet.r_array[i] = planet.r
```

Code Task 10.5 Change the velocity of the earth to get a roughly circular orbit (you could go up to 35,000 m/s, faster than suggested in part one). You can run this cell as many times as you need! *Hint: Your orbit may not "look" circular, even when it is! Be sure to note the horizontal vs. vertical scaling of the graph.*

Problem 10.6 What orbital velocity did you settle on to get a circular orbit? Look up the Earth's actual orbital velocity and see how close you were! Describe how far off you were with a percent error. (<https://www.wolframalpha.com/input/?i=earth+orbital+velocity+in+m%2Fs>)

Part Three: Changing Parameters

Now that we have some sense of how Earth's orbit works, and how changing our parameters changes the results. Before we begin, we should introduce some quantitative measure of our orbit to serve as evidence of the changes we introduce. One such measure is the orbital period (year), which we will calculate below.

Read and understand what is happening in the following code before continuing on with the lab.

```

def length_of_year(planet, dt, tolerance=R_Earth*0.01):
    '''
    This takes in a planet after it has orbited and calculates
    the length of the year based on how long it takes for it to
    return back to its starting point. Because our simulation
    is not perfect, some tolerance is allowed in case it misses
    its starting point by a little
    '''

    def distance_from_start(index):
        '''
        This calculates the distance at a given time the planet
        is from its start location
        '''
        start_loc = planet.r_array[0]
        curr_loc = planet.r_array[index]
        distance = curr_loc - start_loc
        return np.sqrt(distance.dot(distance))

    # This loop seems a little complicated, but really it
    # is just first checking that the planet got away from
    # start and then came back, so it doesn't just think
    # it has finished a loop immediately
    near_start = True
    for i in range(len(planet.r_array)):
        if near_start and distance_from_start(i) \
            > tolerance:
            near_start = False
        if not near_start and distance_from_start(i) \
            < tolerance:
            return i*dt
    return 0 # If something went wrong, it will say the
            # year was instant

```

Now, our first modification is going to be to make the sun feel force. Up to now, we have treated the Sun as completely static, which is rather artificial. Remember Newton's third law: the force on the Earth due to the sun is matched with an equal and opposite force on the Sun due to the Earth. We're going to account for the force on the Sun by treating the Sun as a really heavy planet. Then

we can generalize our code a bit, treating the two objects on a truly equal footing.

In particular, we are going to change our code by looping over a list of "planets" (including the Sun) rather than listing out each one every time we update it, apply force, or initialize it.

Again, read and understand what is happening in the following code before continuing on with the lab.

```

v_initial_earth = 29800 # m/s

earth = ball(m=M_Earth, r=np.array([R_Earth,0,0]),
            v=np.array([0, v_initial_earth, 0]))
sun = ball(m=M_Sun)

planets = [earth, sun]

for planet in planets:
    initialize_planet(planet, N)

i = 1
for t in times[1:]:

    # first find the force
    for planet in planets:
        planet.force = 0
        for second_planet in planets:
            planet.force += force(planet, second_planet)

    # Update the planet(s)
    for planet in planets:
        update_planet(planet, planet.force, dt)

    i = i + 1

print(length_of_year(earth, dt))

plt.plot(earth.r_array[:,0], earth.r_array[:,1], '.')
plt.plot(sun.r_array[:,0], sun.r_array[:,1], '.')
plt.axis('equal')
plt.show()

```

Note the numerical output from this code cell is the output of our `length_of_year` function (in seconds). Compare to the actual Earth year of 3.154×10^7 s.

Problem 10.7 Compare your result from the previous code block to the actual length of an Earth year.

Problem 10.8 We made our simulation account for the force on the Sun, but the Sun still isn't moving. Why do you think this is? Is it realistic?

Cumulative Assessment Task (CAT)

For the Culminating Assessment Task (CAT), you will simulate a three-body system. Specifically, the system should have two bodies, both with mass M_{Sun} , orbiting each other (like the system above). A third body, with mass M_{Earth} , should closely orbit one of the first two bodies.

For this simulation, we have chosen starting values for T and dt :

```
T = 4e7
dt = 1e5
```

We also have chosen the following initial conditions to get you started:

- Body #1:
 - Mass: M_{Sun}
 - Initial position: a distance R_{Earth} to the left of the origin
 - Initial velocity: 9,000 m/s in the $-\hat{y}$ direction
- Body #2:
 - Mass: M_{Sun}
 - Initial position: a distance R_{Earth} to the right of the origin
 - Initial velocity: 9,000 m/s in the $+\hat{y}$ direction
- Body #3:
 - Mass: M_{Earth}
 - Initial position: a distance $0.9 \cdot R_{\text{Earth}}$ to the left of the origin (just to the right of Body #1)
 - Initial velocity: 100,000 m/s in the $+\hat{y}$ direction

Code Task 10.6 Update the code cell provided in the Jupyter notebook to create a list of three balls using the initial conditions given above. Then run the cell.

With the initial conditions set, we can simulate the three-body orbit:

```
times = np.arange(0, T+dt, dt)
N = times.size

for sun in suns:
    initialize_planet(sun, N)

i = 1
for t in times[1:]:

    # first find the force
    for planet in suns:
        planet.force = 0
        for second_planet in suns:
            planet.force += force(planet, second_planet)

    # Update the planet(s)
    for planet in suns:
        update_planet(planet, planet.force, dt)

    i = i + 1

print(length_of_year(suns[2], dt, tolerance=R_Earth*0.1))
for sun in suns:
    plt.plot(sun.r_array[:,0], sun.r_array[:,1], '. ')
plt.xlim(-1.5*R_Earth, 1.5*R_Earth)
plt.ylim(-R_Earth, R_Earth)
plt.axis('equal')
plt.show()
```

That's pretty neat! If everything has gone well, you should see that the two massive bodies (blue and yellow) make complete ellipses. Meanwhile, the green planet tightly orbits the blue body, but eventually rockets off into the distance.

Simulation is a powerful tool. Today, we have simulated systems that we could never fit in a lab. This allowed us to get a "hands-on" experience with gravitational systems in a way that would not be possible without simulation. However, we need to be careful. In this case, due to the parameters

we gave you, the simulation does not give us a physically accurate result. Given the masses, initial positions, and initial velocities, it turns out the green planet should not actually rocket off into the distance. Instead, it should continue orbiting the blue planet.

Notice how the simulated points for the green planet are spaced a somewhat far apart. Given that this planet is moving quite fast, perhaps we have chosen too large a value for the time step dt .

Code Task 10.7 Go back to the code cell where dt is defined at the beginning of the CAT and try different values for dt until you are satisfied. (Be sure to run all cells from the start of the CAT up to this Code Task to see your results.)

Problem 10.9 How can you feel confident that your simulation is showing a physically accurate result, as opposed to an artifact of the simulation parameters? How do you know if you chose a "good" value for dt ?

Problem 10.10 What physical system could this represent?

With the simulation parameters properly set, we can run the following code to animate the simulation:

```

fig, axs = plt.subplots()
xMin = 0
xMax = 0
yMin = 0
yMax = 0
for sun in suns:
    xMin = min(xMin, np.min(sun.r_array[:,0]))
    yMin = min(yMin, np.min(sun.r_array[:,1]))
    xMax = max(xMax, np.max(sun.r_array[:,0]))
    yMax = max(yMax, np.max(sun.r_array[:,1]))
axs.set_xlim(xMin, xMax)
axs.set_ylim(yMin, yMax)
axs.set_aspect('equal')
axs.plot([-1,1], [0,0], color='C1', lw=2)
line, = axs.plot([], [], '.', markersize=24, lw=2)
plt.close()

anim_many = animation.FuncAnimation(fig, \
    animate_ball_list(suns), init_func=init, \
    frames=N, interval=1)
anim_many

```

Laboratory 11

Waves and Sound

Introduction

We are now in a position to study wave mechanics, which is directly related to our previous studies of circular motion and simple harmonic motion. A **wave** is defined as any periodic disturbance that propagates through a medium. For example, a sound wave is a disturbance in pressure that propagates through a medium. At different parts of the wave, the molecules of the medium are closer together or farther apart, creating areas of higher and lower pressure. A sinusoidal sound wave is considered a wave, rather than just a disturbance, because these areas of high and low pressure are **periodic**: they occur at regular intervals of position and time. An understanding of wave mechanics is crucial to your understanding of light and optical phenomena in future physics courses.

Theoretical Background: Waves and Simple Harmonic Motion

We will consider only sinusoidal waves in this laboratory. A sinusoidal wave looks like Figure 1. The displacement d of a single particle of the medium located at a position x at a time t is given mathematically by the expression

$$d(t) = A \sin(kx - \omega t + \delta) \quad (42)$$

where ω is the angular frequency, and k is called the wave number defined as $k = 2\pi/\lambda$.

Problem 11.1 In Figure 1, the red circles represent a single point on the wave that changes position as the wave moves through the medium. If the first of the four images (the top image) is the earliest, which direction is the wave traveling? What if the bottom image is the earliest?

This expression has several important features:

- If we look at a “snapshot” of the wave at a single point in time (say $t = 0$), it is a sine curve. This is true no matter the time at which we take the picture. As time goes on, the wave simply shifts position within the medium, but its shape stays constant.

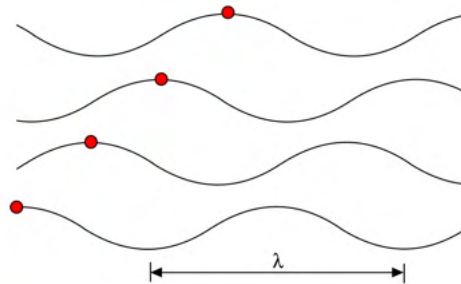


Figure 1: A picture of a sinusoidal wave moving through a region of a medium. The four “snapshots” show the displacement of the medium at four different points in time. The red circle represents a single point on the wave that changes position with time.

- An individual particle of the medium can never be displaced from its equilibrium position by more than the **amplitude** A of the wave.
- The parameter δ is called the **phase** of the wave. It represents the offset of the wave from zero displacement at $t = 0$. For example, two waves can look identical, but can be shifted relative to each other in space at a particular point in time. The phase governs how much of a shift there is. For example, in Figure 2, the top image shows two waves that are perfectly **in phase** ($\delta_1 = \delta_2$). The bottom image shows two waves that are perfectly **out of phase** ($\delta_1 = \delta_2 + \pi$).
- Waves that have the form of Eq. 42 are also called traveling waves. This will be in contrast to **standing waves**, which will be discussed later.

The important connection between waves and simple harmonic motion is this: the individual particles of the medium oscillate in simple harmonic motion, but they never leave their equilibrium positions by more than A . The pattern of their combined displacements is the “wave” that actually travels through the medium.

Theoretical Background: Wave Properties

So far, we know that when a sinusoidal wave (Eq. 42) travels through a medium, the individual particles of the medium oscillate with amplitude A and angular frequency ω . There are a few other properties of waves that one should remember. First, a wave has a property that is related to its frequency called its **wavelength**, which is represented by λ . The wavelength is just the distance between identical parts of the wave (see Figure 1, which includes a labeled wavelength). Second,

just like in simple harmonic motion, the angular frequency is related to the frequency via

$$\omega = 2\pi f,$$

where f is the frequency in Hz (cycles per second). In the context of a wave, this means that f wavelengths of the wave pass a given point on the x -axis each second. If we then multiply by the wavelength, we can determine the **wave speed**

$$v = \frac{\omega}{k} = f\lambda. \quad (43)$$

The wave speed is constant for a given medium and is not dependent on any properties of the wave itself. Therefore, if λ increases within a given medium, the frequency f must decrease, and vice versa.

Problem 11.2 Why can't you determine the wave speed by taking the derivative of Eq. 42?

Transverse waves are a class of waves in which the medium oscillates *perpendicular* to the direction of wave motion. A special exception to this is the electromagnetic wave, which is transverse, but in which electric and magnetic fields oscillate instead of matter. Transverse waves describe phenomena such as the vibration of strings on stringed instruments, for which the oscillating medium is the string, and light, which is an electromagnetic wave. In the context of Eq. 42, the displacements d of individual particles are in either the y or z direction.

Longitudinal waves are a separate class of waves in which matter oscillates *parallel* to the direction of the wave's motion. Probably the most important type of longitudinal wave you will encounter is the sound wave, which is any type of longitudinal pressure wave traveling in a medium (such as air or water). For a longitudinal wave, the displacements described by Eq. 42 are in the x direction, and the wave also travels in the x direction. This is a bit of a tricky concept because the particles are oscillating parallel to the direction of wave travel, but they do *not* move through the medium along with the wave. Oscillations in a slinky accurately demonstrate this.

Problem 11.3 What is the key difference between transverse and longitudinal waves? Draw a picture of a transverse wave and a longitudinal wave at a single point in time. *Hint:* Think about the waves on a slinky.

Theoretical Background: More on Wave Speed

One important aspect to remember about waves is that the speed at which they propagate through a particular medium is a property of the *medium*, not the wave. This means that the speed of a

wave in a particular medium is constant (This is not *technically* true, but it is a good approximation for the phenomena we will be studying here. **Dispersion** occurs whenever wave speed depends on wavelength, and you can find an extensive treatment of it in any intermediate physics text.).

We will be studying two types of waves in this lab: longitudinal waves propagating through air and transverse waves propagating along a string. The speeds of these waves have simple formulae that are related to certain material properties of the air and the string. For example, the speed of a wave on a string is given by

$$v = \sqrt{\frac{F}{\mu}}, \quad (44)$$

where F is the tension in the string and μ is the string's mass per unit length. For sound waves in air, the speed is given by

$$v = \sqrt{\frac{\gamma P_{\text{atm}}}{\rho}}, \quad (45)$$

where ρ is the density of the medium, P_{atm} is the atmospheric pressure of the air, and γ is a dimensionless factor from thermodynamics that is related to the heat capacity of the medium. We will use 340.3m/s for the speed of sound in air at sea level, but we will test Eq. 44 by varying the tension in an inelastic string and seeing how it affects the wave speed in that string.

Problem 11.4 For the same atmospheric pressure, why is the sound in helium much faster than air?

Problem 11.5 Imagine a massive string hanging vertically from one of its ends. Because the string has mass, there is actually more tension at the top of the string than at the bottom. Will the speed of a wave change as it moves through this string? If so, will it go faster at the top or at the bottom?

Theoretical Background: Principle of Linear Superposition and Standing Waves

What happens when two waves pass through a medium at the same time? The answer is simple: at every point in the medium, the displacements of the two waves add. This simple concept is called the **Principle of Linear Superposition** (see Figure 2). If the waves are perfectly in phase, the result is a wave that looks like the original waves but with twice the amplitude. If the waves are perfectly out of phase, the result a wave with zero amplitude.

On a side note, the Principle of Linear Superposition is another approximation that is sufficiently accurate for this course but which has many exceptions. In general, it is true only in *linear* media.

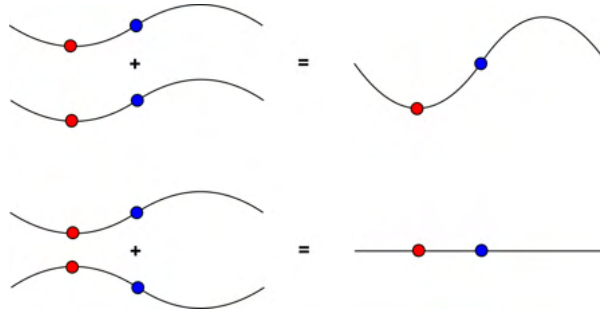


Figure 2: Illustration of the Principle of Linear Superposition. The top image shows what happens when two waves are perfectly in phase; the bottom image shows what happens when they are perfectly out of phase.

You can read about the distinction between linear and nonlinear media in any intermediate physics text.

When two sinusoidal waves of the same wavelength ($\lambda = \frac{2\pi}{k}$) and amplitude are traveling in the opposite directions, their superposition creates a wave that do not move left or right. This superpositioned wave is called a **standing wave**.

This can be seen by combining the displacements of the two sinusoidal waves:

$$d(t) = A \sin(kx - \omega t) + A \sin(kx + \omega t). \quad (46)$$

Applying trigonometric identities, Eq. 46 leads to

$$d(t) = [2A \sin(kx)][\cos(\omega t)]. \quad (47)$$

The quantity $[A \sin(kx)]$ can be viewed as the amplitude oscillating at position x . The quantity $[\cos(\omega t)]$ is the oscillating term. Note that this oscillating term is different from a traveling wave represented as Eq. 42.

Theoretical Background: Resonance

A **resonator** is a device or system that naturally oscillates at some frequency or set of frequencies. These special frequencies are the **resonant frequencies**. If a resonator is excited by the introduction of a wave that has the same frequency as one of its resonant frequencies, the phenomenon of **resonance** occurs.

Resonators are usually physical objects that oscillate at certain frequencies because their dimen-

sions are equal to an integral multiple of the wavelength (or half-wavelength) at those frequencies. For example, imagine a cavity (a medium of fixed length) in which a wave travels. An example of a cavity for transverse waves is a string fixed at both ends. Another cavity might be an air-filled pipe open at both ends or a pipe open at only one end. There are certain wavelengths that lead to resonance in these cavities. For an inelastic string fixed at both ends, these wavelengths are given by the formula

$$\lambda_n = \frac{2L}{n}, \quad (48)$$

where $n = 1, 2, 3, \dots$ and L is the length of the string. Figure 3 shows (in red) waves with wavelengths corresponding to the first three resonant frequencies in Eq. 48. Note that for this type of cavity, the waves corresponding to the resonant frequencies have zero amplitude at the ends of the cavity. A place where a wave has zero amplitude is called a **node**. Likewise, a place where a wave is at maximum amplitude is called an **antinode**.

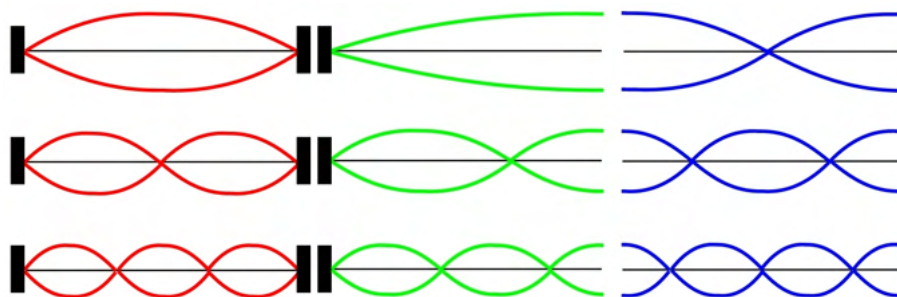


Figure 3: The first three resonant wavelengths for a cavity with two fixed ends are shown here in red. The green waves represent the first three resonant wavelengths for a cavity with one fixed end and one open end. Finally, the blue waves represent the first three resonant wavelengths for a cavity with two open ends. Note that the distance between any two successive nodes is always $\lambda/2$.

Another example of a resonant cavity is a pipe open at both ends. In this case, the resonant frequencies correspond to waves with antinodes at both ends of the cavity. The expression for the resonant wavelengths is the same as that for a string fixed at both ends (see Eq. 48). For a pipe open at both ends, the wavelengths are

$$\lambda_n = \frac{2L}{n}, \quad (49)$$

where $n = 1, 2, 3, \dots$ and L is the length of the pipe. Figure 3 shows (in blue) the first three resonant wavelengths for a cavity with two open ends.

Problem 11.6 Why are Eq.s 48 and 49 identical, even though they describe very different physical setups?

The expression for the resonant wavelengths of a pipe open at only one end, or a string fixed at only one end, looks somewhat different. This is because these situations are asymmetric; the resonant frequencies correspond to waves with nodes at the fixed end and antinodes at the open end. For a cavity of length L with one fixed end and one open end, the resonant wavelengths are

$$\lambda_n = \frac{4L}{n}, \quad (50)$$

where $n = 1, 3, 5, \dots$ (odd n only). Figure 3 shows (in green) the first three resonant wavelengths for this type of cavity.

Experimental Background: Waves on a String

In our first experimental setup, a string is attached to a variable frequency mechanical wave driver, which in turn is attached to a sine wave generator capable of adjusting the frequency of the output wave to the nearest tenth of a Hz. The other end of the string passes over a pulley and attaches to a variable hanging mass. This accurately reproduces a cavity with a string fixed at both ends. One node occurs (approximately) where the string is attached to the wave driver, and the other occurs where the string meets the pulley. The hanging mass can be varied to adjust the tension in the string and hence the wave speed.

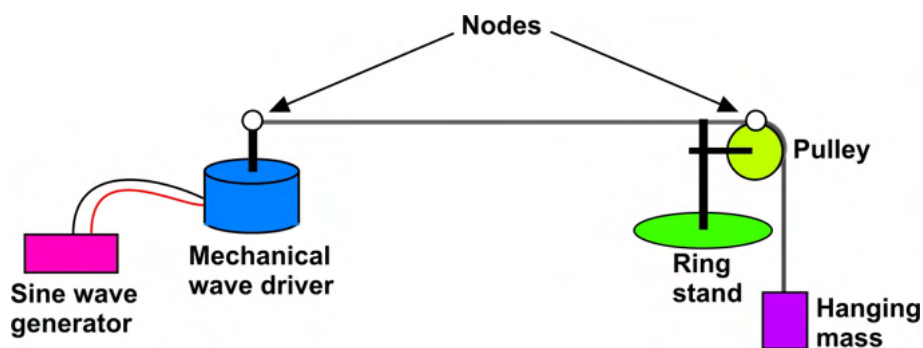


Figure 4: The experimental setup for producing standing waves on a string.

A diagram of the experimental setup is shown in Figure 4, and photographs of some individual components are shown in Figure 5.

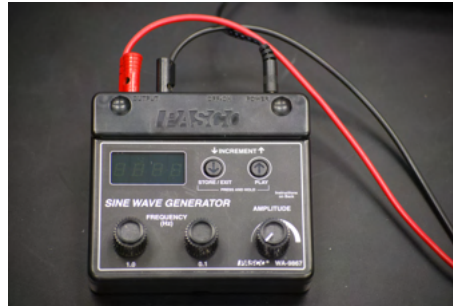
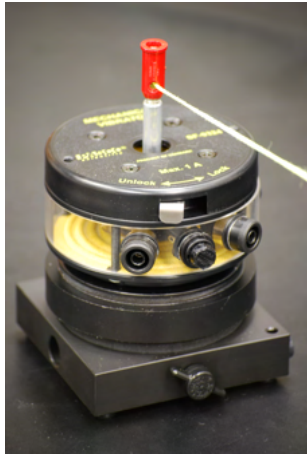


Figure 5: *Left:* A photograph of the variable frequency mechanical wave driver. *Right:* A photograph of the sine wave generator used in both experiments.

Experimental Background: Waves in a Tube

We can generate standing waves in a tube using a setup involving a speaker and a plunger that can be moved in and out of the tube to alter the length of the resonant cavity. A microphone placed inside the tube is used to detect resonance; the microphone can be placed at different locations in the tube to find the nodes and antinodes of the standing wave. What the microphone actually measures is sound pressure, and the sound pressure, it turns out, is highest at a node and lowest at an antinode. So pressure nodes are displacement anti-nodes and vice versa. Since there is always an antinode at the open end of a tube, a microphone placed there will barely detect any sound pressure when resonance occurs. Placing the microphone several inches into the tube (toward a node) will help solve this problem.

Experiments

Experiment 1: Standing Waves on a String

For our first experiment, we will verify the relationship between wavelength and cavity length as well as the relationships between wave velocity, tension, and mass per unit length for a string with two fixed ends.

1. One can measure the mass per unit length μ of the string by weighing a known length of it with a scale and dividing the mass by the length. This has been done for you, and the value is given in the lab report template.

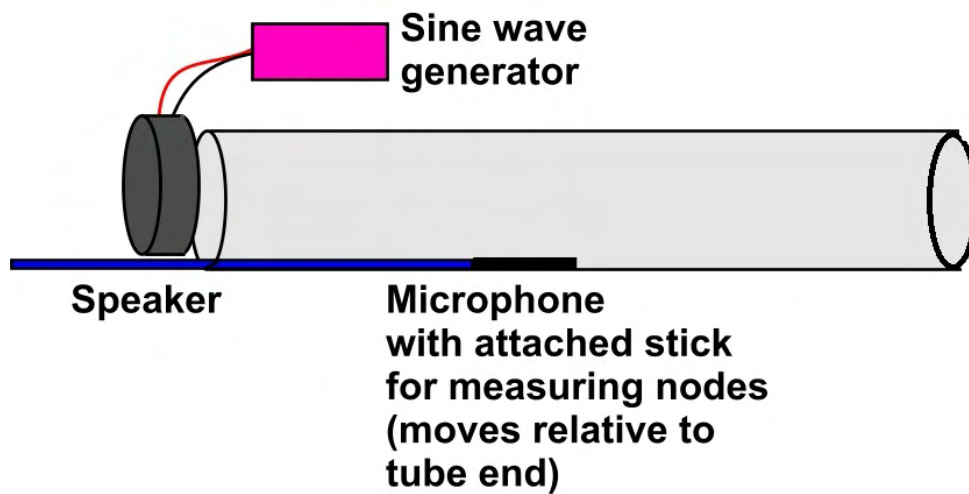


Figure 6: The experimental setup for generating standing waves in a tube.



Figure 7: The speaker used to create standing waves in a tube. Like the mechanical wave driver, it is powered by the sine wave generator.

2. Assemble the experimental setup shown in Figure 4. Start with a total hanging mass of 100 g. Make sure the mechanical vibrator attached to the string is “unlocked” so that the oscillating pin can move freely.
3. Adjust the equipment so that the length of the string between the two nodes (fixed end points) shown in Figure 4 is approximately 1 meter.
4. Turn the amplitude dial of the sine wave generator all the way down. Turn on the sine wave generator and dial the frequency down to 1 Hz.

5. Turn up the amplitude until the string oscillates visibly. Dial up the frequency gradually using the coarse adjustment knob until you observe the first normal mode, as shown in Figure 3. Fine-tune your measurement to the nearest tenth of a Hz using the fine adjustment knob. Record this frequency in your lab report.
6. Continue turning up the frequency until you have observed a total of five normal modes. Record the resonant frequencies associated with these normal modes in your lab report. You may have to adjust the amplitude of the sine wave generator to more easily observe all of the normal modes.
7. Repeat this procedure with a total hanging mass of 200 g.
8. Use the online data collection tool to record your measured values for the first five resonant frequencies for the case of a 100 g hanging mass (see Appendix A Sections 2.1-2.3) in Chart 1.
9. When all of the class data has been collected, follow the instructions provided in the Jupyter notebook to download and save the class data as a '.csv' file so that it can be imported into Jupyter notebook.
10. Once the data is imported into Jupyter notebook, use the provided code cell to calculate the mean, standard deviation, error of the mean, and 95% confidence interval for the first five resonant frequencies.

Problem 11.7 Using Eqs. 43 and 44, calculate the wavelengths that correspond to your measured resonant frequencies for both sets of data taken with different hanging masses.

Problem 11.8 Using Eq. 48 along with Eqs. 43 and 44, calculate the theoretical values of the first five resonant wavelengths (and the corresponding theoretical frequencies) associated with this string's L , μ , and F . Also calculate the percent error between your observed frequencies and the theoretical frequencies. Do this for both sets of data taken with different hanging masses.

Problem 11.9 Do any of the 95% confidence intervals generated from the class data *not* include the corresponding theoretical frequency found in Problem 11.8? What would it mean if this were the case?

Problem 11.10 Give an example of a musical instrument that operates based on the principles you have observed here. How would you make different notes (different frequencies) using this instrument? Explain.

Problem 11.11 Consider the fundamental frequency ($n = 1$) for the case of the string with two fixed ends. Would the following sources of error cause your experimental fundamental frequency to be greater than, less than, or no different than the theoretical frequency? Write a one-sentence explanation in each case.

- (a) The end of the string attached to the mechanical vibrator is not actually a node (because it is not actually sitting still as the string vibrates). This is indeed the case for our experiment.
- (b) The mass per unit length of your string is slightly greater than what was measured.
- (c) The part of the string that is attached to the hanging mass has significant mass of its own.

Experiment 2: Waves in a Tube with Two Open Ends

We now verify the relationship between wavelength and cavity length for a tube with two open ends that has the same length as the string used in Experiment 1.

1. Feed the output of the sine wave generator to the speaker instead of the mechanical vibrator. Once again, start with the amplitude at its lowest possible value and keep it as low as possible throughout the experiment.
2. Position the speaker very near (but not directly against) one end of the tube, leaving the other end open. This setup creates a tube with both ends open. Your task is to measure the resonant frequencies for the first five standing waves.
3. To determine when you have found a resonance, you can either use your ear, a microphone, or (preferably) both. Keep in mind that, at resonance, the sound can be rather intense, even at low input amplitudes. Adjust the amplitude accordingly.
4. Record the resonant frequency in your lab template.
5. Open *LoggerPro*. If a microphone is attached, you should be ready to take sound pressure data. Set the length of data collection to 600 seconds at a sampling rate of 60 samples per second, and click the box marked “Continuous Data Collection” (see Appendix A Section 1.7).
6. Position the microphone about a foot into the end of the tube opposite the speaker, start collecting data, and observe the change in sound pressure as you alter the frequency coming from the sine wave generator. You are looking for a frequency at which the amplitude of the sound pressure observed in *LoggerPro* is a maximum. Note that *LoggerPro* does not know

what frequency your sine wave generator is feeding the speaker, so make sure you record the resonant frequency yourself before moving on.

7. As you slowly dial up the frequency, your graph should eventually contain a number of peaks that represent the resonant frequencies. Label these peaks with their associated frequencies using *LoggerPro*'s text annotation feature. Paste your graph into your lab report. Record the first five measured resonant frequencies in the table in your lab report as well.

You may end up with a graph without nice and clean peaks if you have changed the amplitude while recording or dialing the frequency knob back and forth. That is fine. You can easily get a nice looking graph by programming the sine wave generator to sweep across the frequencies with a constant speed. To do so, use the following instructions for AUTO PLAY:

1. First set the frequency to 2 Hz. This will be the increment frequency.
2. Next, press and hold the left increment button until it blinks, then release.
3. Set the frequency to 100 Hz. This will be your starting frequency.
4. Have a lab partner get ready to write down resonant frequencies. These occur at pressure wave peaks, you might have to start over if you miss one so be ready!
5. Finally, press and hold the right increment button. Autoplay should begin when you release. If it does not begin, press the right button once to start it.
6. You can adjust how fast it increases by tuning the middle knob. A lower number increases the rate, and a bigger number decreases the rate. We recommend starting at 0.3.

The result is a graph with clear peaks which you can then label with the frequencies you've found. Note that you should only use the auto play function after finding all five resonances because in auto play mode, you will not be able to read the frequency with enough accuracy before it is changed.

7. Use the online data collection tool to record your measured values for the first five resonant frequencies (see Appendix A Sections 2.1-2.3) in Chart 2.
8. When all of the class data has been collected, follow the instructions provided in the Jupyter notebook to download and save the class data as a '.csv' file so that it can be imported into Jupyter notebook.
9. Once the data is imported into Jupyter notebook, use the provided code cell to calculate the mean, standard deviation, error of the mean, and 95% confidence interval for the first five resonant frequencies.

Problem 11.12 Using the known speed of sound in air, calculate the wavelengths that correspond to your measured resonant frequencies. Using the length of the tube and again the speed of sound in air, calculate the first five theoretical resonant wavelengths and frequencies.

Problem 11.13 What exactly is the microphone in this experiment measuring? Is this quantity a maximum or a minimum at the end of the tube? Is it a maximum or a minimum at the nodes within the tube? (*Hint: Refer to the section Experimental Background: Waves in a Tube.*)

Problem 11.14 If we lengthened the tube, would the fundamental frequency increase or decrease? Give an example of a musical instrument that operates based on this principle, and describe how it works in a few sentences.

Problem 11.15 Would the following sources of error cause your experimental fundamental frequency to be greater than, less than, or no different than the theoretical frequency? Write a one-sentence explanation in each case.

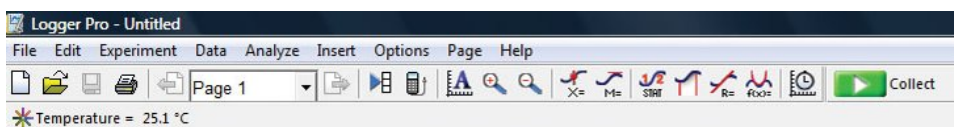
- (a) Some helium gas has leaked into the room, so the air density is lower than assumed.
- (b) The speaker is quite far (a few inches) from the end of the tube.
- (c) The tube is actually shorter than it was measured to be.

Problem 11.16 The length of the tube and the length of the string in Experiment 1 are identical. Therefore, the theoretical wavelengths for the first five normal modes should be the same. Why then are the frequencies corresponding to these normal modes different? In which experiment are the frequencies higher? Why?

Appendix A

Software Tips and Tricks

1 LoggerPro



1.1 To add a statistics box to a histogram ...

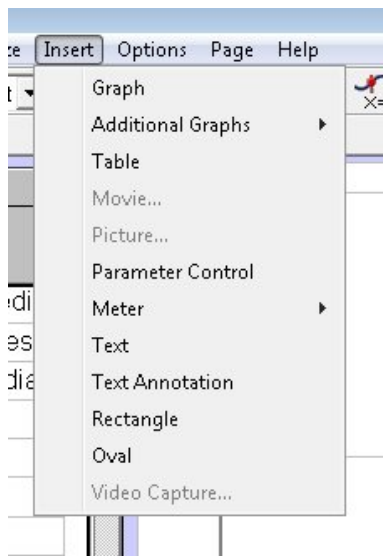
1. Highlight all of the data points on the histogram.
2. Choose *Analyze > Statistics*.
3. A box will appear that includes the minimum, maximum, mean, median, standard deviation, and number of samples in your data set.

1.2 To fit a line to a graph ...

1. Highlight the region of the graph you want to fit.
2. Choose *Analyze > Linear Fit* from the *LoggerPro* menu.
3. A line will appear with a box that includes its slope and intercept.

1.3 To create a graph ...

1. Choose *Insert > Graph* from the *LoggerPro* main menu.



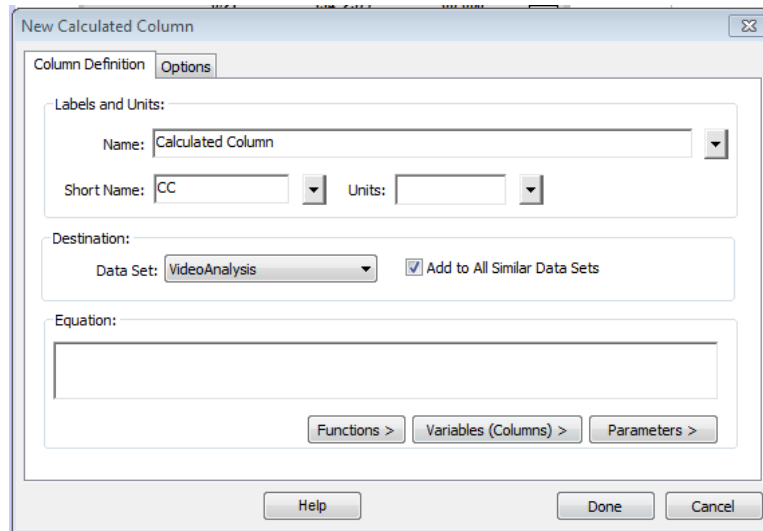
2. *LoggerPro* will create a new graph without first asking you what you want on the axes. Don't worry: this is normal.
3. Right-click on the graph and choose *Graph Options*, then choose the *Axes Options* tab. Under this tab you can...
 - Choose the data sets you want to represent the dependent (y-axis) and independent (x-axis) variables. You can choose more than one dependent variable to plot on the same graph, but you can only choose one independent variable.
 - Adjust the axes scales, or have *LoggerPro* assign them automatically.

1.4 To fit an arbitrary function to a graph ...

1. Highlight the region of the graph you want to fit.
2. Choose *Analyze > Curve Fit* from the *LoggerPro* menu.
3. If the function you want is one of the functions listed on the *Curve Fit* menu, choose it.
4. Otherwise, click *Define Function* and enter your desired function.
5. Press *Try Fit...* and click *OK* once the fitting algorithm is finished.

1.5 To create a new calculated data column ...

1. Choose *Data > New Calculated Column*.



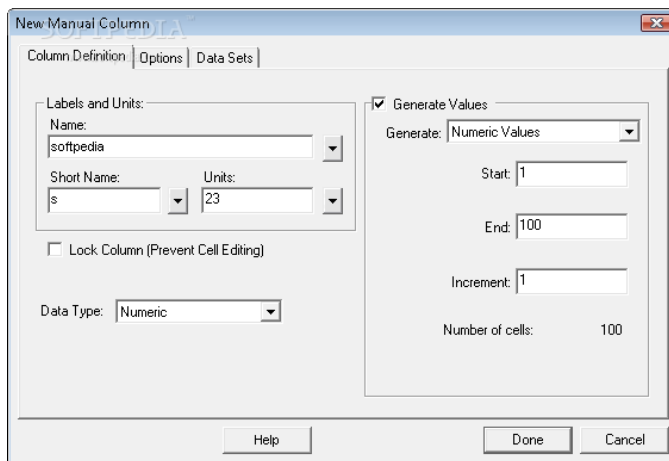
2. In the dialog box that appears, name your new column something descriptive. Be sure to fill in a “Short Name” too, since this is what will appear to you on the data table (the two names can be identical).
3. Select the data set you want to work with. If you have more than one, make sure “Add to All Similar Data Sets” is selected.
4. In the *Equation* box, you will specify what values *LoggerPro* should populate the column with. For example, if you have previously existing columns of data with short names “D1” and “D2”, and you wanted to compute the difference between them, you would select “D1” from the *Variables* drop down list, then type “-”, and then select “D2”. You can access more complicated functions in the *Functions* drop down list, though the standard operations (+, -, *, /) and any parentheses () can be typed using the keyboard and should be sufficient for our purposes.

1.6 To create a custom data table ...

1. Open a blank *LoggerPro* file. You can do this by simply opening *LoggerPro* after unplugging all devices from the green hub on your table.
2. You will see a single graph and two columns of data: *X* and *Y*.

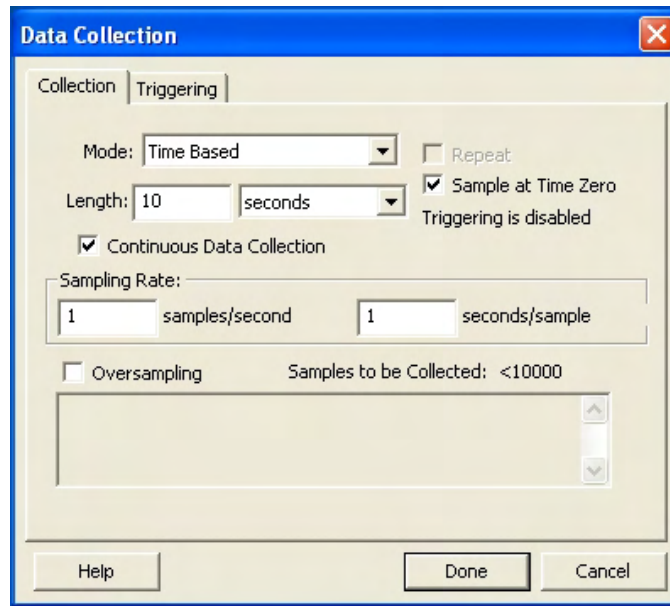
3. You can either ...

- (a) Double-click on a column title in the table, which will allow you to edit the properties of that column (name, units, etc.), or...
- (b) Make an entirely new column by choosing *Data > New Manual Column*.



1.7 To adjust the data collection settings ...

1. To adjust the sampling parameters in *LoggerPro*, choose *Experiment > Data Collection*.
2. A variety of settings will appear.
3. Choose *Mode > Time-Based* if you want to take data for a specific length of time. You can set the sampling rate as well (in samples/second).
4. If you click the box marked *Continuous Data Collection*, *LoggerPro* will continue taking data at the specified sample rate until you click *Stop*.



1.8 To set up your video camera for video capture ...

1. Make sure your video camera is plugged in to one of the regular USB ports on your computer.
2. Open *LoggerPro* and choose *Video Capture* from the *Insert* menu.
3. You will see what the camera sees in a box in the upper left of the video capture window. Move the camera until it is positioned to capture most of the air table and is looking directly down at it.
4. If the image of the air table is too dark or too bright, or motion appears very blurry, try changing some of the camera's settings. If you are having trouble setting the proper exposure time and brightness, ask your GSI for assistance.





1.9 To record video ...

1. After setting up your video, click *Start Capture*. After a short moment, video capture will begin.
2. *LoggerPro* will then record video for the next 10 seconds. It will then store your video automatically.

1.10 To analyze video ...

After you insert or capture a movie, you need to make some adjustments so that your desired data sets (usually the x and y positions of some object) will show up correctly on a graph. The following buttons are useful. They appear to the right of the video in *LoggerPro*.

 **Enable / Disable Video Analysis** Either displays or hides many of the buttons used for analyzing your video. The buttons are initially hidden by default, so this button will be one of the first buttons you will click before analyzing a video.

 **Select Point** Highlight a trace point in the movie object. Use this button to edit existing objects. For example, if you mis-click a point using *Add Point*, you can select the point using this tool and delete it.



Add Point Allows you to mark the location of an object in a frame. Allows you to choose the points in the video that you want to record, automatically advancing through the video as you click.



Set Origin Tells *LoggerPro* where the origin is. Click it and then click on the point on the video that you wish to be the origin. A set of axes will appear. Click on the large yellow dot on the x -axis and move it around to rotate the axes relative to the video frame.



Set Scale Allows you to choose two points in the video, for example the two ends of the air table, and tell *LoggerPro* how far apart those points are in physical units, like meters, instead of pixels. Now the axis of your graph will be in the units of your choosing, rather than pixels.



Photo Distance Use this to measure a distance in a video, once you have set the scale. Click the button, and drag across the distance of interest on the image.



Set Active Point *LoggerPro* can track multiple objects. For example, in Laboratory 7 you will need to create separate data sets for the motion of two separate pucks. Use this tool to add a new data series or track a new object, or to return to a previous series.



Toggle Trail Displays or hides all the points that have been added up to the current time.



Show Origin Display the origin of the movie object. Click again to remove the origin. Remember: the origin is set using the *Set Origin* button.



Show Scale This will show the line used in setting the scale.



Synchronize Video Sometimes there will be extraneous video frames before the event you care about. You'll want to synchronize the video with your graph so that $t = 0$ on the graph corresponds to the start of the "action" in the video, not some random earlier frame. To do this, make sure that the video is set to the frame that shows the point just before the puck launches (or whatever event you care about occurs). Then click the button shown here. Set the Graph Time equal to zero.

1.11 To add a FFT (Fast Fourier Transform) graph to sound data ...

1. Choose *Insert > Additional Graphs* and choose *Fast Fourier Transform*.

2. A FFT graph will appear for your sound data. This graph will update as you take more sound data. Therefore, every time you get a nice graph, be sure to copy it to your lab report before collecting more data.

2 Online Data Collection Tool

2.1 Location ...

<https://webapps.lsa.umich.edu/introlabs/labcharts/>

When you open Google Chrome from the desktop, the first tab that opens is the Online Data Collection Tool, so you don't have to type in the link above. On this page, you first need to type in your username and password.

2.2 Entering data ...

1. Choose your room number, your section number, the number of today's lab, and the "chart number" (on some days, you may need to submit more than one kind of data; the chart number is used for this purpose).
2. Click Edit on the row corresponding to your Station/Group you are working in.
3. Input your values in cells 1-10 (or however your GSI wants you to input them), and click *Update* when finished.
4. Your data is now visible to the entire class.

2.3 Viewing class data ...

1. From the main menu, choose *View Data*.
2. Select the room number, section number, lab number, and chart number that correspond to the data set you wish to view.
3. You can download the data to use in your lab report by selecting the "Download Report to Excel" button and then in the pop-up window give the data file the appropriate name ending in '.csv' in the appropriate Lab data folder. Improper naming of the file, failure to convert it to .csv file by adding '.csv', or downloading the data into the wrong folder will result in the data not being properly utilized in the Jupyter notebook.