

Math Drag'n: An Application for Eliminating Algebra Mistakes
in Physics Homework

David Ludlow

A capstone report submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Bachelor of Science

Keith Leatham, Advisor

Department of Physics and Astronomy

Brigham Young University

April 2013

Copyright © 2013 David Ludlow

All Rights Reserved

ABSTRACT

Math Drag'n: An Application for Eliminating Algebra Mistakes in Physics Homework

David Ludlow

Department of Physics and Astronomy
Bachelor of Science

I present a new program, Math Drag'n, that allows students to solve or manipulate equations interactively. Students apply transformations on their equations by dragging the variables or numbers around the equation. This program is designed to help students eliminate algebraic errors from their homework and tests. It has potential to boost students' productivity and attitude, while doing homework. Traditional computer algebra systems are not well adapted for introductory physics coursework or proof problems. The workflow and paradigm behind Math Drag'n are fundamentally different from those of any other program currently available. They allow students to control the problem, almost as much as they can on paper. That control is important for the students learning experience and necessary for proof problems. I argue that Math Drag'n, or a program based on Math Drag'n's principles, should eventually replace paper and pencil as the main medium of calculation for physics homework, especially in introductory physics classes or when working proofs of any level of math.

Keywords: physics education, symbolic manipulation, CAS, computer algebra system, dragging, user experience engineering, interaction design, computer-aided math, step-by-step, math education, engineering education

ACKNOWLEDGMENTS

Dr. James Hart wrote most of the code for Math Drag'n when he was an undergraduate at BYU.

I would not have had the time to program the dragging part of Math Drag'n on my own, so if it had not been for James, I would have had to make a hotkey and toolbar based interface. The underlying math logic that he wrote is phenomenally accurate. The GUI he made was astoundingly fleshed out.

Dr. Keith Leatham is my capstone project advisor. There was once a time when I thought that I might not get approval to do this project, because I could not find an advisor. I asked every physics professor and computer science professor that I knew, and I asked many that I did not know. No one wanted to take on the extra work. Dr. Leatham was the first professor that I tried in the Math Education Department. He accepted the role after only one small meeting, even though he knew very little about what was required of a physics capstone project advisor. It is good to have an advisor who knows how to do educational research. See his website at <http://mathed.byu.edu/~kleatham/index.html>.

Dr. Cheryl Davis was brave enough to be the first teacher to give Math Drag'n to her students.

Dr. Dallin Durfee was brave enough to be the second teacher to give Math Drag'n to his students.

Alex Safsten is working on the log function, and is helping in designing a unit conversion feature.

Other students will keep the project alive while I am gone. I will not write their names until they do something that becomes part of the project.

Dr. J. Ward Moody did a lot of editing on this document.

Dr. Jean-François Van Huele did a lot of editing on this document.

Excelsior is a company that gave me an expensive software packaging tool for free.

Contents

Table of Contents	vii
List of Figures	ix
1 The Goal of Math Drag'n	1
1.1 The Problem: when math minutia distracts from physics concepts	1
1.2 The Solution: a nail gun	2
2 Why Math Drag'n is Different	5
2.1 Programs that just give the answer	5
2.2 Programs that show how to get the answer	7
2.3 CAS calculators and paper	7
2.4 Programs that check your work as you go	10
2.5 Programs that suggest possible individual transformations	10
2.6 Summary	13
3 Math Drag'n's Interface and Workflow	15
3.1 Overview	15
3.2 Enter the equations	15
3.3 Manipulate and mix equations	17
3.4 Get symbolic and numerical answers	18
3.5 Turn it in	20
4 The Story of Math Drag'n	23
4.1 Before I started Math Drag'n	23
4.2 Finding Math Drag'n	24
4.3 A code style issue	25
4.4 Changes I made	25
5 Math Drag'n's Effect and Future	27
5.1 Preliminary testing	27
5.2 Keeping work symbolic	28

5.3	The good practices, skills, and algorithms that Math Drag'n teaches	29
5.4	What I would do differently	31
5.5	But why does Math Drag'n not do such-and-such?	31
5.6	Other future work	32
5.7	Vision of the future	34
Bibliography		37
A Changes I Made to Math Drag'n		39
A.1	Creating an easy way to view output	39
A.2	Adding tabs	40
A.3	Rewriting the user license agreement	40
A.4	Rewriting help documentation	41
A.5	Making the program ready for distribution	41
Index		45

List of Figures

2.1	Screenshot: Wolfram Alpha	8
3.1	Screenshot: entering an equation	16
3.2	Screenshot: display's equations with fancy formatting	16
3.3	Screenshot: the left panel, where values are substituted into variables	19
5.1	Screenshot: verbose example of Math Drag'n showing steps	33

List of Tables

5.1 Two tables that describe the classes that gave Math Drag'n to their students. 27

Chapter 1

The Goal of Math Drag'n

1.1 The Problem: when math minutia distracts from physics concepts

How much time have you spent looking for math mistakes? Students spend a great deal of time looking for basic arithmetic and algebra errors. I spent about a third of my homework time looking for mistakes in my first years of physics. Math mistakes are a problem.

Have you ever noticed that, towards the end of most physics problems, there is a thicket of basic arithmetic and algebra? Consider algebra-based physics coursework. The beginning of the problem typically requires that the student employ the new physics concepts that they are learning. They need to obtain accurate initial equations and figure out how they will use them to get the answer. But after that physics exercise, they have to mix those equations together, solve for the desired variable, and substitute in numbers to produce a final answer. This thicket of tedious math is only as educationally useful as it reinforces the subject matter and is typically not the best way for a student to spend their time.

Basic math often distracts students from the physics concepts that they are supposed to be

learning. In fact, it is my opinion that math is the average physics student's greatest distraction.

The thicket can even distract students emotionally. For many of my friends and me, math is the scariest part of doing physics homework. It sometimes brings feelings of insecurity, because even if you know the physics—the math, waiting for you at the end of the problem, could still prevent you from solving the problem quickly or accurately. Even for advanced students, lots of basic math in homework is a problem.

1.2 The Solution: a nail gun

In an effort to mitigate the tedious half of math, I set out to do for math-based homework what the nail gun did for stick framing. My solution is a Java program called Math Drag'n.

It had to be a new program, because existing tools, such as Mathematica [1], perform a different function than what I was looking for. Mathematica, in our analogy, knows how to stick frame most types of houses, and it will complete that entire process for you. You just need to tell it what walls you want. There are many ways to frame a 8 ft by 15 ft wall with a 4 ft by 4 ft hole for a window at the center. Mathematica will frame it however it wants to frame it.

Nail guns, on other hand, do not know how to create entire walls, that is the carpenter's job; however, nail guns join boards together much faster than hammer and nail, which, in our analogy, represent paper and calculator. Math Drag'n does not know how to solve a system of equations, but it is good at putting in individual nails, or rather, it is good at executing individual algebra steps. If you tell Math Drag'n to subtract a variable named *nail* on both sides of an equation, it will do that step for you, and it will not forget the negative sign. The student is left as the mastermind behind solving the problem.

This is the basic workflow of Math Drag'n:

1. The user types in the equation(s).

2. Using the mouse, they drag variables around the equation.
3. Math Drag'n accurately manipulates the equation according to each drag.
4. When the desired variable is solved for, the user specifies the values of the remaining variables.
5. Math Drag'n shows evaluated and unevaluated forms of the equation, as well as each step.

Learning physics interactively with computer help is a natural technological progression. Consider, for example, when hand-held calculators came along. Some educators were reluctant to stop using slide rules, and many people had become attached to their slide rule. However, the transition from slide rule to calculator was necessary. With calculators, people could get an answer with less effort, and could, therefore, spend more thought on the high-level concepts. Although some people may find it hard to leave the use of paper for computation, those feelings are just growing pains, as we move on to the next generation of tools and techniques [2].

The next chapter, Chapter 2, explains that there is no other math program or tool that does what Math Drag'n does. Chapter 3 goes over how Math Drag'n works, and why it was designed the way it was. Chapter 4 tells the story of developing Math Drag'n. Chapter 5 describes the small amount of field testing performed on Math Drag'n and discusses conclusions I have drawn from the tests. That chapter also describes our future plans for the project, especially our plans to make Math Drag'n a valuable assistant for work using almost any level of math, in any setting, even on exams.

Chapter 2

Why Math Drag'n is Different

Math Drag'n is the only computer algebra systems (CAS's) of its type. There are several other categories of CAS's, but they each have specific reasons why they have failed to replace paper in physics homework. With the possible exception of the tools I mention in Section 2.4, each of these tools have powerful application, but none of them are optimized for helping with the specific type of coursework that this document is discussing, i.e., mostly algebra-based physics and engineering homework and tests.

2.1 Programs that just give the answer

Many upper-level undergraduate students at Brigham Young University (BYU) use Mathematica [1] when they need a little computer help for their traditional paper calculations. If you type in an equation, Mathematica will solve it. Wolfram|Alpha [3], Maple [4], Maxima [5], Sage [6], TI-89 [7], TI-Nspire CAS [8], and many other CAS's do this as well.

Mathematica can be very useful, especially for more advanced students. It is not suitable for students who still need the practice solving the particular type of equation that they are required to use.

The students I am targeting have book knowledge on how to solve, for example, the lens equation,

$$\frac{1}{f} = \frac{1}{p} + \frac{1}{q} \quad (2.1)$$

for q , but most of them do not have the practical experience in solving Eq. (2.1) to know that they should get

$$q = \frac{1}{\frac{1}{f} - \frac{1}{p}} \quad (\text{correct}) \quad (2.2)$$

instead of

$$q = \frac{1}{f - p} \quad (\text{incorrect}) \quad (2.3)$$

or

$$q = f - p. \quad (\text{incorrect}). \quad (2.4)$$

These incorrect answers are often encountered. If students had practice solving this type of equation, they would not make these mistakes nearly as often. Programs like Mathematica (excluding Wolfram|Alpha which will be discussed later) do not show steps, so students do not gain experience in seeing this type of problem solved. The students are less likely to acquire the skills needed to find the solution on their own.

Another problem with programs that just give the answer is that they tend to over-simplify output. When Mathematica is asked

$$\text{Solve}[1/f == 1/p + 1/q, q] \quad , \quad (2.5)$$

it outputs

$$\left\{ \left\{ q \rightarrow -\frac{fp}{f-p} \right\} \right\} \quad (\text{correct}). \quad (2.6)$$

I do not consider this output very useful to the student's education, because it is so over-simplified. Because there are multiple instances of f and p in that equation, it is hard to see and imagine how f and p really do affect q . The un-simplified lens equation, Eq. (2.2), is much better for helping

student's understand how lenses work. Mathematica frequently over-simplifies equations, making the output less meaningful to the student.

2.2 Programs that show how to get the answer


On the internet, there are many programs that will solve an equation and then show the steps, so one can learn how to do it independently. Normally these programs do one thing each, such as integrate a certain type of expression, or solve a primitive algebraic equation. By far, the most polished program of this type is Wolfram|Alpha [3]. From the creators of Mathematica, this free web service will show you the steps for solving many, but definitely not all, math problems. It is nice, but as Fig. 2.1 shows, the steps are unnatural, lengthy, and easy for a student to ignore, in favor of just looking at the answer. Even if the student had to copy the steps down, there is nothing to force the student to learn them.

Do not try to use Mathematica or Wolfram|Alpha to do a proof. It is almost always impossible, because that is not what they were built for. Proofs are an essential part of physics, but right now the only good way to do almost any proof is by manually writing each step. Math Drag'n, on the other hand, can assist in doing a proof just as easily as it can assist in solving an equation.

2.3 CAS calculators and paper

Calculators with symbolic manipulation capability (CAS calculators) are very easy to use compared to programs on PC's. Calculators fit on any desk space; they have button layouts that are optimized for the task; they are often less complicated; they do not have as many battery issues; they can be taken on most physics tests; they cannot easily be used for social media; etc.

CAS calculators can typically solve equations, like Mathematica can. They have most of the same weaknesses as PC based CAS's, like Mathematica, because a CAS calculator just gives the

 solve($1/f = 1/p + 1/q$, p)

Input interpretation:

solve	$\frac{1}{f} = \frac{1}{p} + \frac{1}{q}$	for	p
-------	---	-----	-----

Results: [Hide steps](#)

$f \neq q$ and $p = -\frac{fq}{f-q}$ and $f \neq 0$

Possible intermediate steps:

Solve for p :

$$\frac{1}{f} = \frac{1}{q} + \frac{1}{p}$$

$\frac{1}{f} = \frac{1}{q} + \frac{1}{p}$ is equivalent to $\frac{1}{p} + \frac{1}{q} = \frac{1}{f}$:

$$\frac{1}{q} + \frac{1}{p} = \frac{1}{f}$$

Bring $\frac{1}{p} + \frac{1}{q}$ together using the common denominator pq :

$$\frac{p+q}{pq} = \frac{1}{f}$$

Cross multiply:

$$f(p+q) = pq$$

Expand out terms of the left hand side:

$$fp + fq = pq$$

Subtract $f q + p q$ from both sides:

$$p(f-q) = -fq$$

Divide both sides by $f - q$:

Answer:

$$p = -\frac{fq}{f-q}$$

WolframAlpha

Figure 2.1 This is a screenshot of Wolfram|Alpha showing the steps to solve the lens equation. Notice that the steps appear to be slightly arbitrary—in other words, most humans would not solve this problem this way. Also, notice that the final result is of the form that I criticized in Section 2.1.

answer that it calculated its way and simplified however it wanted to.

There are many benefits of using paper. Almost all of our mathematical syntax is optimized for use on paper; you can draw pictures on paper; you can define new mathematical operators, and immediately use them, if you are using paper; etc. But there are also many disadvantages of using paper and calculator.

Calculators cannot directly interact with information on paper. Assuming you write your homework with what is commonly accepted at BYU as “good style”—you already wrote on your paper what you are going to type into your calculator. This redundancy of typing into your calculator exactly what you have already written on your paper uses lots of valuable time, but students still exercise this “good style” because it greatly reduces the chance of making a mistake when typing into the calculator. This exercise also makes group collaboration easier. However, even with “good style”, students frequently enter their math into their calculator incorrectly. It does not matter how many advances we make in paper making, this will always be a problem with paper.

One may argue that one could take a picture of handwritten math and use optical character recognition to convert that picture into code that a calculator could understand. Even with moderately futuristic optical character recognition, there are so many technical complications with the practice of converting handwritten math into calculator code that it will take seriously creative innovation to make software that can do that efficiently enough to actually be used by a typical student. The “technical complications” mentioned are beyond the scope of this document. My argument still stands, that because paper and calculator cannot talk to each other, students spend too much time and make too many errors as they transcribe in between paper and calculator. We need a better tool.

Because the user has to type out initial equations into Math Drag’n, users can also make input mistakes. With paper, if you make a correction at the beginning or in the middle of the work, you have to carefully propagate the change down through the rest of the steps. With Math Drag’n, if

you change the first line, the program deletes all the steps after that. That is not good, but can be fixed in future editions; we just need more programmer man-hours. Even in Math Drag'n's current version, it is typically not too hard to redo the steps. Redoing the steps mostly just means re-dragging things, which does not take that long. The longest step of using Math Drag'n is typing the initial equation.

People who work on a long equation on paper typically spend a lot of time copying a lot of unchanged, or barely changed, information from line of work to line of work. Math Drag'n would save them time by writing the next line of work for them.

The largest fault of using paper is that you are prone to error from line to line. Math Drag'n does not make mathematical mistakes, with the exception that it does not check for divide by zero in the intermediate steps. There is also a tiny bug, where sometimes the thing that you are dragging around disappears, but this is infrequent, and when it happens it is obvious.

2.4 Programs that check your work as you go

There are many programs on the internet that check your work as you go. Some examples are Aplusix [9] and Algy [10]. These programs make you retype every line of your work, and they tell you if what you typed is true, based upon the previous line. They are mostly used in algebra classes.

This type of tool is not worth using in a physics class. The ones I found are underdeveloped and are much more tedious to use than paper.

2.5 Programs that suggest possible individual transformations

There are a few programs that, upon inspecting an equation, suggest possible transformations that can be performed on the equation. When the user selects one of the options, the program performs

the transformation.

Although this is similar to Math Drag'n's workflow, it is different and it does not completely solve the problem that Math Drag'n intends to solve. These programs do a great deal of high-level reasoning in order to come up with a list of options of possible transformations. This makes it inappropriate to use these tools in many educational settings, because students need to practice looking at an equation and recognizing possible steps forward in the problem. If a program is always analyzing the problem and giving suggestions based on that, then what assurance do we have at the end of the problem that the student was the mastermind behind solving it?

Because these programs operate by giving a list of suggestions of what transformations can be done, it is possible that some of the ideas needed to solve the problem are not coming from the student but are coming from the computer. It is important in academia that ideas needed to solve a problem originate from the student and that student's education. Math Drag'n does not offer suggestions, so it does not have this problem.

An example of a program that offers suggestions is Symbolic Math Guide (SMG) which comes as a pre-installed app on the TI-89 calculator. This app has been around for a long time and was not even perpetuated on the predecessor of the TI-89, the TI-Nspire CAS [8]. Part of the reason that it was not perpetuated is that it is tedious to operate. Selecting the subexpression that you wish to transform can be a frustrating [11] and long process. For more info about SMG, see the study done by S. Asly Özgün Koca and Michael Todd Edwards [11].

The closest program to Math Drag'n currently available is part of Maple called Clickable Math [4]. As of Maple 16, the user can drag terms to do the most basic addition, subtraction, multiplication, and division, but the only place a term can be drug is to the other side of the equals sign.

Drags in Math Drag'n can do much more. Its drags can do commutative transformations, associative transformations, distribution. It can do factorization as long as the term being factored

out is a one-polynomial. Math Drag'n also allows you to take the square root of both sides of an equation by dragging, and will, in the future, allow you to take the inverse of functions like trigonometric and logarithmic functions with a drag.

Math Drag'n may be more advanced at transformation via dragging, but unlike Math Drag'n, Maple's Clickable Math gives suggestions of possible transformations when you highlight part of an output expression. These suggestions can be useful, and can perform powerful transformations like applying a trigonometric identity or solving an equation for a variable. These context sensitive menus somewhat make up for the lack of ability in dragging.

However, opening up a context sensitive list of suggested transformations is like opening a box that someone else packed for you to provide you with tools you will later need. It can be frustrating when it turns out that that box does not contain the tool you were counting on being there. Both SMG [11] and Maple's Clickable Math often do not offer the transformation that the user expects to be there. This lack of showing the wanted transformation is a problem that is fundamental to the situation and cannot be fixed appropriately by hiring more programmers. Even if you had the time to make a program that could create a complete list of every normal transformation or class of normal transformations that could be done to a given equation, the resulting list would be too large to be looked at and efficiently be used efficiently in a program like this. (Strong innovation might someday prove that last statement untrue.)

Math Drag'n, with the exception of a few right click menu items that I regret, always keeps its figurative tools in the same place in the shed so that you know where to get them, rather than hoping the program will bring you the tool you want. Right now this is only evident in the fact that you can always try a drag transformation without being offered to try a drag transformation. When the user learns a new type of drag transformation, it is like they learned where a tool is kept so that they know where to get it when they need it. Only so many transformations can be easily controlled by dragging, so in the future, Math Drag'n will also use ribbons to organize transformations, as

discussed in Section 5.4. Each ribbon will be like a shelf with unmoving tools.

Both SMG and Maple's Clickable Math were optimized primarily for math as opposed to physics. Because of that, neither of these programs have a way to substitute values in for variables and then later change those values as conveniently as can Math Drag'n.

2.6 Summary

The main idea of each type of math tool:

1. Mathematica solves equations completely, and the student is left not knowing how it was solved.
2. Wolfram|Alpha shows you how to solve equations completely, but it can't do proofs or expressions that are too complicated.
3. Paper and calculator help you do whatever, even if the whatever is wrong.
4. Some programs can check your work as you manually slave away typing each step.
5. Maple's Clickable Math suggests possible things that you can do to the selected part of the equation.
6. Math Drag'n does your work incrementally, after you figure out what each new step should be.

Chapter 3

Math Drag'n's Interface and Workflow

3.1 Overview

The best way to learn how Math Drag'n works is to see it in action. This can be done by watching a 7.5 minute internet video screencast at <http://ludlowcloud.com/math/2013capstone/index.html> (if that link breaks, try <http://www.youtube.com/watch?v=XV6JP20rxjw>). A picture is worth a thousand words, and a screencast is worth even more. The screencast shows *how* it works and what using it *is like*. The text of this chapter only describes how it works enough to defend the *why* behind its design.

3.2 Enter the equations

To start using Math Drag'n, a user first types in their equations. They type only plain text i.e. there is no fancy math formatting (Fig. 3.1). I define fancy formatting to be the bar in expressions like $\frac{a}{b}$, the square root in expressions like \sqrt{a} , superscripts, Greek letters, and other special mathematical syntax. When the user presses Enter, the equation is displayed with fancy formatting (Fig. 3.2). The user will perform their transformations on this fancy formatted version.

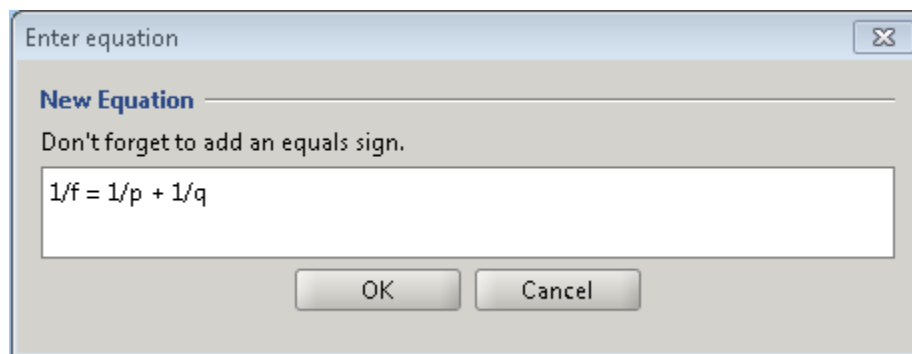


Figure 3.1 The user types an equation in plain text. This equation is the lens equation.

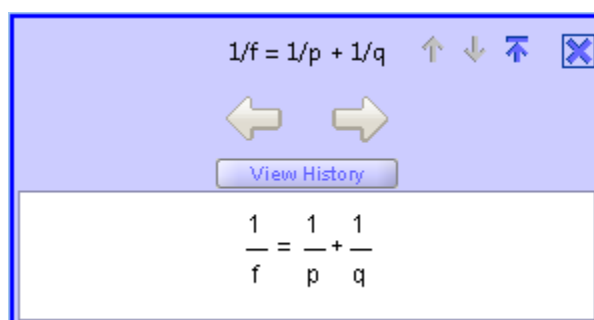


Figure 3.2 Math Drag'n displays the equation with fancy formatting for fractions.

Math Drag'n avoids fancy formatting in equation input on purpose. In general, plain text input is faster to type, easier to edit, and more widely accepted by other programs. It is my opinion that only the most polished programs—such as Mathematica, Maple, and Grapher for Macs—have an interface that makes entering equations via fancy formatting even close to worth the effort.

Entering in equations via plain text is an important skill. All regular programming languages use only plain text, and there are many programs and calculators that do not have either the specialty or the sophistication required to warrant allowing fancy formatted input. Students may, at first, complain about having to put parentheses around the entire denominator of a fraction, but looking at mathematical expressions in plain text becomes easy with practice. Or at least it becomes easy compared to physics! Another benefit of plain text is that it makes “copy” and “paste” commands much simpler.

3.3 Manipulate and mix equations

Now that the equation is entered into Math Drag'n, the user needs to manipulate, or in other words, transform the equation. Most often, the goal is to solve for a specific variable. The user decides what steps to take; they tell Math Drag'n those steps by dragging; and Math Drag'n executes each primitive step. It is important to note that the decision of what steps should be done originates completely from the user. The only hint that Math Drag'n gives is that it shows the equation's current state. One could also argue that Math Drag'n gives hints by refusing to carry out invalid drags.

Most transformations can be specified with a drag. In a drag, the user selects the part of the equation they want to move, and they move it to another location. These gestures are intuitive. This follows a common mental pattern that many students already use to solve algebra problems.

All the steps are shown in Math Drag'n's history panel. The user can go back to any step. If

they change a step, then all subsequent steps are deleted.

There are some transformations that cannot be done with a drag. A common example is taking both sides of an equation to the power of two. In fact, for most functions, there is no way to apply that function on both sides of the equation with a drag. Transformations that do not involve a drag are done by selecting the equation, right-clicking, and then using the options in the right-click menu to specify what you want. In Section 5.4, I explain why right-click menu transformations are not optimal and why future versions of Math Drag'n will not have transformations in the right-click menu.

Typically, multiple equations are needed to solve a physics problem. Users can substitute an equation into another equation by dragging the first equation onto the second one. To illustrate the point more specifically: If the user drags $A = B$ onto $C = D$ then Math Drag'n will replace every instance of A within C and D with the value of B .

3.4 Get symbolic and numerical answers

Via step-by-step transformations, the user has now solved for the variable that they originally wanted. At this time, the user probably has a large, chunky symbolic answer. This is on purpose. Users may elect to spend more time manipulating their equation to simplify it, but often the chunky answer is nicer. For example, I prefer using chunky Eq. (2.2), even though most mathematicians would consider it not fully simplified.

The left side of the main screen is dedicated to giving values to variables and to getting the final result (see Fig. 3.3). The table on the top-left shows all the variables that can be found. There, the user can enter in values for the variables. This does not affect the math in the center panel. That is to say—the copy of the equation that is in the place where dragging happens (the center panel) will still stay in symbolic form, even when some of its variables know what their value will be.

a.

Variable Substitutions	
Variable	Value
f =	
p = -4	
q =	

Equ. with Substitutions

$$q = \frac{1}{\frac{1}{f} - \frac{1}{(-4)}}$$

Equ. Evaluated as Much as Possible

$$q = \frac{1}{\frac{1}{f} + 0.250}$$

Treat substituted integers as decimals

b.

Variable Substitutions	
Variable	Value
f = 6.4	
p = -4	
q =	

Equ. with Substitutions

$$q = \frac{1}{\frac{1}{(6.4)} - \frac{1}{(-4)}}$$

Equ. Evaluated as Much as Possible

$$q = 2.46$$

Treat substituted integers as decimals

Figure 3.3 Both (a) and (b) are screenshots of the left panel of Math Drag'n. The only difference between (a) and (b) is that (b) has a specified value for f . Users type values for variables at the top of the left panel. The middle of the left panel shows the numbers substituted into the currently active equation, but it does not evaluate itself. The bottom of the left panel shows the active equation with all the available values for variables, and the equation is evaluated as much as it can be. The bottom of the left panel, therefore, is where the user will find the answer, when the problem is finished.

It is good style to keep equations in symbolic form while you perform transformations on them. Instructors of introductory physics often have a hard time getting students to follow that style rule. Physics 106 at BYU is one such class. It consists of mainly pre-med students. As a teaching assistant of physics 106, I found that even when the students had completed physics 105 and were almost done with physics 106, that around half of students who came to get help still had not developed the habit of keeping their work in symbolic form. If those students had used Math Drag'n, then Math Drag'n would have naturally given them training in keeping their work symbolic, a training that they refused to receive when they were using paper and calculator. That is one of the largest potential educational benefits of Math Drag'n.

Now, let us go back to describing the function of the left panel (Fig. 3.3): the middle of the left panel shows the equation that is active in the center panel, but the equation is shown in a special form that does not use any arithmetic to simplify itself but still has numbers substituted in for variables. Many students who use paper and calculator already have a habit of writing down this special form of the equation to help improve their computational accuracy. When these students find that they do not have the same final answer as someone else in their study group, this special form of the equation is often the first thing that they will compare with their buddies. This is a perfect example of how Math Drag'n, unlike most *math* programs, is specifically optimized for *physics* coursework.

3.5 Turn it in

We have not yet given Math Drag'n the ability to print. This is simply because we have not had the man power to implement it yet.

We are lucky that so far, this has not been a problem. It just so happens that the classes that Math Drag'n has been deployed in do not require their students to turn in a copy of the steps they

used to get the answer. They just need to submit their numerical answer online.

Math Drag'n does make a file that you can save, but, as it is, that file is best used for personal records, because we have not yet created an easy way to mark an expression as the answer to a problem.

Chapter 4

The Story of Math Drag'n

4.1 Before I started Math Drag'n

I started designing the Math Drag'n user experience in 2007. As I did my math-based homework, I would imagine exactly what type of tool would be most useful for what I was doing. I would think of what weaknesses my current homework workflow had, and what could be done to improve the workflow. As I studied the reasons why my homework was so hard and boring, I learned most of the content presented in Chapter 2, “Why Math Drag'n is Different,” and I conceptually designed most of the content presented in Chapter 3, “Math Drag'n's Interface and Workflow.”

Before I wrote any code, I tried to find a program that already did what I wanted. As of at least the beginning of 2013, there is no other program available that has the qualities that I deemed necessary to someday replace paper as the main medium of calculation. If a program like Math Drag'n does indeed exist out there, it must be hiding under a rock and, therefore, cannot help us.

I, therefore, decided to develop a program myself. I obtained approval to do it as a capstone project. But the program that I started making was not Math Drag'n. Its name was Equation Transformer (ET). I wanted to make a program that used dragging for manipulation, but I thought

that, as the only developer, it would be too much work for me to handle. Instead, to specify a transformation when using ET, the users would select part of an expression, then they would press a hotkey or a button to specify which transformation to do to that part of the expression. For example, the user selects p and then presses the “Divide on both sides” button or the hotkey “d” to divide p on both sides of the equation.

This would be more efficient and user friendly for power-users than Math Drag'n is now. The user could select parts of expressions with the mouse in the right hand, and press hotkeys with the left hand. Each hotkey would be exactly one regular letter. It just so happens that almost every letter that I would want to use as a hotkey can be easily accessed by the left hand, as it sits in its default position on the keyboard. This would make it so the user would not even need to look down at the keyboard to press hotkeys.

The only reason why Math Drag'n does not have similar hotkeys, in addition to the ability to drag, is a lack of time. I plan on adding hotkeys later, for the sake of power-users. For regular users, I believe that the intuition and coolness factor that dragging affords is more valuable than the time that could be saved by using keyboard shortcuts.

4.2 Finding Math Drag'n

When I first told Dr. Grant Hart about my project, he said that it sounded a lot like something that his son had made. That son, now Dr. James Hart, wrote most of the code that is in Math Drag'n today back in 2003-2006, when he was an undergraduate at BYU. Unfortunately, James did not publicize the program, so the project stagnated. It was a miracle that I found out about it.

The code that James wrote was powerful, extensive, and, for the most part, reliable. It was so much more developed than ET, that I decided to scrap ET and go with Math Drag'n instead. Everything that involves dragging was already there. The math was surprisingly accurate. It had

every aspect of a typical program. It had menus named “File,” “Edit,” “View,” and “Help” that were full of the usual amenities. It had a user license agreement. It even had artwork for icons and a logo. The list goes on and includes many other vital components, both small and large. If you looked at the main window of Math Drag’n then and compared it to today’s version, all you would probably notice different is that the left panel is new, there are tabs for different problems, and there is a slider on the toolbar for changing the number of significant figures shown.

4.3 A code style issue

The code that I started with had a huge programming style mistake. The front-end of the program that people see, called the GUI (Graphical User Interface), was mixed with the logic of the program. When James first started this program, he was not a computer scientist, he was a physicist who taught himself how to program. He did not know that you must separate the GUI (the front end) from the main logic of the program (the back end), to facilitate maintaining the code. For this and other reasons, it takes a lot of effort to edit any of James’ code.

I estimate that it would take about 150 man-hours to separate these two halves of the program now. James is now both a computer scientist and a physicist. He would not make the same mistake again!

4.4 Changes I made

Unfortunately, when I received Math Drag’n, it was not user friendly. When I first tried it, it took me two hours to figure out how to solve the lens equation (2.1) for q . I have changed the user experience significantly since then. There are still many large parts of the program that need to be made more user friendly, but I have not had the resources to implement those changes.

The time that I have spent programming Math Drag’n far exceeds the requirements for a cap-

stone project. When I got the program, it had 30 151 lines of Java code, excluding blank lines and libraries. (Libraries are reusable code, often made by someone else.) Now Math Drag'n has 33 862 such lines. That makes a difference of 3711, but I changed and deleted many lines of James' code. For every line of code that I affected, I had to learn many lines of James' code.

An account of some of the main changes that I made can be found in Appendix A. It would be a distraction to list those changes here.

Chapter 5

Math Drag'n's Effect and Future

5.1 Preliminary testing

We made Math Drag'n available to three algebra-based physics classes. Table 5.1 describes these classes.

As the table shows, not that many people tried Math Drag'n. Each of the classes made Math

Course	Course Description
Physics 123	Intro to waves, optics, thermodynamics, and relativity
Physics 105	Algebra-based mechanics, thermodynamics, waves, and acoustics

Course	Instructor	Semester	Students that tried
Physics 123	Dr. Cheryl Davis	Fall 2012	5
Physics 123	Dr. Dallin Durfee	Winter 2013	2
Physics 105	Dr. Cheryl Davis	Winter 2013	10 (rough estimate)

Table 5.1 Two tables that describe the classes that gave Math Drag'n to their students.

Drag'n optional.

I watched students as they used the program. Of the students that used it Fall Semester 2012, I only really received one piece of usable feedback. It was from student who was having a hard time with doing the algebra on paper. He looked real stressed. I showed him how to use the program. After a couple of problems, he happily announced to me that he had obtained correct answers on the first try. He looked excited about his work and the appearance of stress was gone. This is the exact effect that this program was designed to induce.

5.2 Keeping work symbolic

Dr. Cheryl Davis of BYU has had a difficult time trying to get her students to solve equations symbolically. Most of her students use paper, and sometimes it is daunting to copy a large symbolic equation from line to line. Dr. Davis and I think Math Drag'n may help with that problem. Right now, the homework for physics 123 at BYU is submitted almost entirely in the form of numerical answers, mainly because that is straightforward for a student to submit, and it is easy for a computer to grade. Math Drag'n would make it more practical for a student to generate a symbolic answer, especially if the answer is bulky. That symbolic answer could be graded with a symbolic equivalence checker, such as can be found in STACK [12] which is a plugin for Moodle [13], a famous open-source framework used to build class websites. Dr. Dallin S. Durfee of BYU independently came up with the same general idea, while he was teaching the same physics class, so there is a good chance we will be able to actually test the idea in a future semester.

5.3 The good practices, skills, and algorithms that Math Drag'n teaches

In watching many people's reactions to the program, I have noticed that it definitely appears that this program is only useful to people who initially believe that it will be helpful to them. This may be because people normally do not try Math Drag'n if they think it would not help them; after all, in every test, the use of this program has been optional.

This result, however, could have deeper causes. Using Math Drag'n on homework can change the way a user processes problems. As mentioned in various parts of Chapter 3, Math Drag'n structures the workflow of a physics problem in a specific way. In this way, using the program forces the user to follow an algorithm or in other words template for solving problems. Some people are uncomfortable with being forced to solve a problem in a certain way. This is especially true when people feel uncomfortable with structuring their personal thoughts to fit academically taught thought algorithms.

However, many of these thought algorithms were purposely designed into Math Drag'n. The idea is to make good habits natural to follow. For example, Sections 3.4 and 5.2 discussed how Math Drag'n naturally helps students keep their work symbolic. Also, the middle of the left panel, as discussed in Section 3.4, makes it easy for students to establish a habit of checking that special form of the equation with their peers. As Section 3.2 discusses, Math Drag'n will help students develop the skills of typing math in plain text and balancing parenthesis.

As mentioned in Section 3.3, many students who have never seen Math Drag'n think of manipulating equations as moving/dragging variables around. Students who are weak in algebra may benefit from adopting this paradigm. Solving equations by dragging is a rich way to solidify mental patterns of what can legally be done to an equation. I hypothesize that, since students can execute dragging transformations quickly, it will be easier for them see the beginning from the end of a

derivation. That is, it will be easier for the student to see the big picture of how their transformations progressed them towards their answer. Perhaps Math Drag'n has a place in an Algebra I class as well as physics. We are looking for an opportunity to test this hypothesis.

There are some rules that Math Drag'n forces student's to follow that are not as educationally beneficial. For example, the user must type all of their initial equations with consistent and non-conflicting names for variables, or else there may be problems when the user tries to substitute one equation into another. This can be frustrating, especially to those who are not as familiar with how strict computers can be with syntax. This will be less frustrating when we make it so users can edit a variable name in an initial equation without Math Drag'n deleting all the transformations they performed on that equation.

There are many people who are averse to using computers, in general, especially for things that they feel can be done well with paper. Sometimes they feel this way because they are not confident that they can easily learn how to make the computer do the task. A lot of times they are just stubborn. In the case of math-based homework, many students have fallen in love with paper and don't want to change. Math Drag'n may help users who are uncomfortable with computers to gain confidence in relying on computers. After all, if people do not let computers help them in their work, then in the future, computers might replace their work.

Alex Safsten, a junior at BYU, and I have started designing a unit conversion feature. We think we have a design that will work somewhat automatically, give the user total control, and help the user become more familiar with dimensional analysis. When using Wolfram|Alpha is not an option, dimensional analysis is the most popular and easy algorithm for doing complicated unit conversions. When we implement the unit conversion feature, we hope it will help students develop the valuable skill of dimensional analysis.

It is good that Math Drag'n teaches all these good practices, skills, and algorithms, but these are just side bonuses. The main point and benefit of Math Drag'n is that it helps students learn

physics concepts more efficiently, by providing an environment that uninhibited by the distractions of computational mistakes and tedious math.

5.4 What I would do differently

For the most part I am satisfied with how the project has gone so far, but there still are some things that I think should have been done differently.

I am, for the most part, the only student that has contributed material that is now part of the project (unless you count what Dr. James Hart did when he was a student). I did try to get help from other students, but I could not get them to join enthusiastically enough to contribute. I am starting to build a team of students.

I regret that some transformations are found in the right-click menu. Part of it is context specific. Users have less surety that they will be able to find what they want in the right-click menu. If I had the resources, I would move the transformations in the right-click menu onto the toolbar. As the program develops and becomes more capable, the toolbar would fill up. Eventually, I think that the toolbar would become not a toolbar, but a set of ribbons. Ribbons are thick toolbars that have tabs, like on the newer versions of Microsoft Word. Each ribbon tab will contain transformations that have to do with a certain branch of mathematics.

5.5 But why does Math Drag'n not do such-and-such?

There are a couple questions that people often ask about the design of Math Drag'n. I want to address these enquiries here.

Lots of people say that they want Math Drag'n to accept handwritten input. This could be done with a stylus on a touchscreen or by taking a picture of what they wrote on their paper. Handwritten input is not a good idea for Math Drag'n because optical character recognition software is

not reliable enough yet. Also, the required hardware for fine stylus input is not readily available enough. There are many devices with touchscreens, but most of them do not have the touch resolution to allow you to write on them with small or even medium font. When ambient technology overcomes these issues, only then would it be appropriate to allow input via handwriting. When that day comes, handwritten input for Math Drag'n will benefit students by helping them make the transition from paper to computer.

Many say that they will not use Math Drag'n unless I make a mobile version for tablets. They will be waiting for a long time, because I do not think that it is possible to make a Math-Drag'n-like program worth using on a device that does not have a physical keyboard. That is, until technology gives us something better than a standard virtual keyboard e.g. handwritten input.

5.6 Other future work

There are several basic features that we have not had the man power to implement yet. Right now, Math Drag'n cannot print. The classes that currently used Math Drag'n have not needed printing, but printing will be necessary for other classes.

There is not an easy-enough way to "move $\sin()$ to the other side of the equation." In other words, it is too hard to apply an inverse trig function to both sides of the equation. James is supposed to be working this now. However, to change anything about how functions like $\sin()$ work, the code needs to be restructured. Alex Safsten, a Junior at BYU, is supposed to be working on this now.

There really should be a way to hide some of the steps. For instance, most often a grader doesn't need a whole new line to show that the student multiplied both sides of the equation by negative one. When students do work on paper, it is a normal and good thing to do more than one small-enough step at a time. Skipping steps may sound dangerous, but consider Fig. 5.1. It is a

>The derivation starts with:
$m_{Pb} C_{Pb} (T_f - T_{Pb}) = - m_w C_w (T_f - T_w)$
>
$m_{Pb} C_{Pb} (T_f - T_{Pb}) = - (m_w C_w T_f - m_w C_w T_w)$
>
$m_{Pb} C_{Pb} (T_f - T_{Pb}) = - m_w C_w T_f + m_w C_w T_w$
>
$m_{Pb} C_{Pb} T_f - m_{Pb} C_{Pb} T_{Pb} = - m_w C_w T_f + m_w C_w T_w$
>
$m_{Pb} C_{Pb} T_f = - m_w C_w T_f + m_w C_w T_w + m_{Pb} C_{Pb} T_{Pb}$
>
$m_{Pb} C_{Pb} T_f + m_w C_w T_f = m_w C_w T_w + m_{Pb} C_{Pb} T_{Pb}$
>
$T_f (m_{Pb} C_{Pb} + m_w C_w) = m_w C_w T_w + m_{Pb} C_{Pb} T_{Pb}$
>
$T_f = \frac{m_w C_w T_w + m_{Pb} C_{Pb} T_{Pb}}{m_{Pb} C_{Pb} + m_w C_w}$

Figure 5.1 This is a screenshot of some of the steps to solve one of the problems for Physics 123 class. It would be easier to read it if some of the smaller steps were hidden.

screenshot of real work that a TA did to become familiar with the particular physics problem. Step #2 really should be hidden from view. The whole would be more readable if it just skipped from Step #1 to Step #3. Step #7 was totally unnecessary to the progression of the problem, so the user should have probably just undone the step before moving on. I have friends that would most likely rather re-write the assignment by hand, then turn in something as verbose and bulky as Fig. 5.1.

This project needs, in general, continued development. If you, the reader, want to help develop this project, please email me at physicsprogrammer@gmail.com.

This product needs to be tested out on a lot more students. Please email me if you want to try.

5.7 Vision of the future

I will continue to develop Math Drag'n after I graduate, in December 2013. I have built a small team of students that will promote it at Brigham Young University while I am gone. I have change-the-world hopes for Math Drag'n.

Most teachers, when they show proofs in class, will use a chalkboard or white board. Students often scramble to keep up in their notes. Math Drag'n is not optimized for this yet, but in the future, it might be a tool used for lectures.

Imagine this hypothetical story of what Math Drag'n may do in the future: the teacher prepares the lecture notes in a Math Drag'n file, and distributes it to the students before class. During the lecture, students take extra personal notes in Math Drag'n. Since this is a future version of Math Drag'n, the teacher can play the transformations as animations. These animations show things like the variable b in $ab = c$ swooshing across the equals sign to come to rest under the c like in $a = \frac{c}{b}$. The student can keep his eyes up and off of his notes, because there is no need to copy every line of math. Later, the student studies for the test, which will be administered in the testing center's computer lab (BYU has such a computer lab), because they will be using Math Drag'n then. The student is able to review everything that the teacher did in lecture, because even though teachers like skipping steps on the chalkboard, they will not skip steps on Math Drag'n.

Math Drag'n will need enormous changes before it is optimized for classroom presentation, but the hypothetical story I just told is a viable future.

I speculate that this program can make it possible for people with dyslexia and other disabilities

to take technical classes that they otherwise would not be able to pass. I never intended to specifically test the program's effect on people with disabilities, but even if it had turned out that the program was not helpful to average students, I strongly believe it would be helpful to those with certain learning disabilities. Take Autism for example. People with autism tend to learn better in settings that involve computers [14].

There is no reason why this program could not be expanded to include every level of math. This can be done with ribbons. Imagine ribbon tabs with the names: "Home," "Algebra," "Trigonometry," "Calculus," "Approximations," "Differential Equations," etc.. In the "Trigonometry" tab, you will find buttons for applying identities such as $\cos^2(\theta) + \sin^2(\theta) = 1$. The "Calculus" tab will have identities that will help you get your integral into a form that can be found on the integration table, which will be on a separate window or panel in the program. These ideas have not yet been tested with mockups or prototypes though, so they may change later.

But will not all the advanced functionality confuse certain people, e.g. High School students? No. When Math Drag'n gains those advanced capabilities, it will also gain settings that will allow the user, or a teacher, to filter-out the functionality that is too advanced or powerful. If there is one version of Math Drag'n for every grade level, students will not have to learn new products as they advance through school, they will just learn more about the product that they already own, as they filter-out less and less functionality.

With advanced math capabilities, Math Drag'n could be used, for example, in a quantum mechanics class. It could be used by a math professor to handle a large proof in their research. These are things that, for the most part, computers can not currently be of much help in; however these advanced tasks are made out of basic steps, and so computers should be able to help more than they already do.

As we have already done with so many other daily tasks, we need to offload our more tedious work onto computers. Math Drag'n is a great example of how we can do that to our math-based

coursework. Despite its shortcomings, I really think that Math Drag'n or a program like it will change the world someday for physics students.

Bibliography

- [1] “Wolfram Mathematica: Technical Computing Software—Taking You from Idea to Solution,” <http://www.wolfram.com/mathematica/> (Accessed April 6, 2013).
- [2] J. W. Moody, Private communication (2013).
- [3] “Wolfram|Alpha: Computational Knowledge Engine,” <http://wolframalpha.com> (Accessed April 6, 2013).
- [4] “Maple 17 - Technical Computing Software for Engineers, Mathematicians, Scientists, Instructors and Students - Maplesoft,” <http://www.maplesoft.com/products/maple/> (Accessed April 6, 2013).
- [5] L. Beshenov, “Maxima, a Computer Algebra System,” <http://maxima.sourceforge.net/> (Accessed April 6, 2013).
- [6] “Sage: Open Source Mathematics Software,” <http://www.sagemath.org/> (Accessed April 6, 2013).
- [7] “TI-89 Titanium - Features Summary by Texas Instruments - US and Canada,” <http://education.ti.com/en/us/products/calculators/graphing-calculators/ti-89-titanium/features/features-summary> (Accessed April 6, 2013).

- [8] “TI-Nspire™ CAS with Touchpad - Features Summary by Texas Instruments - US and Canada,” <http://education.ti.com/en/us/products/calculators/graphing-calculators/ti-nspire-cas-with-touchpad/features/features-summary> (Accessed April 6, 2013).
- [9] “APLUSIX website,” <http://www.aplusix.com/en/> (Accessed April 5, 2013).
- [10] “ALGY software,” <http://www.stepsinlogic.com/> (Accessed April 6, 2013).
- [11] S. A. O. Koca and M. T. Edwards, “Symbolic Math Guide : an Innovative Way of Teaching and Learning Algebra Using TI-89 and TI-92+ Graphing Calculators,” In , (2001).
- [12] C. Sangwin, “STACK,” <http://www.stack.bham.ac.uk/> (Accessed April 6, 2013).
- [13] “Moodle.org: open-source community-based tools for learning,” <https://moodle.org/> (Accessed April 6, 2013).
- [14] C. Putnam and L. Chong, “Software and technologies designed for people with autism: what do users want?,” In *Proceedings of the 10th international ACM SIGACCESS conference on Computers and accessibility*, Assets ’08 pp. 3–10 (ACM, New York, NY, USA, 2008).
- [15] “Excelsior JET - Java Virtual Machine (JVM) and Native Code Compiler,” <http://www.excelsior-usa.com/jet.html> (Accessed April 6, 2013).

Appendix A

Changes I Made to Math Drag'n

A.1 Creating an easy way to view output

I made the entire left panel (see Fig. 3.3). This is the panel where you enter values for variables and where you see the numbers plugged-in. It required me to change the way variables were handled. I also had to change the way functions evaluated, so that they would evaluate thoroughly in the bottom of the left panel.

When I made that functional, the panel often displayed numbers with 15, 30, or more significant figures. (Note that floating point numbers in Math Drag'n are not limited to the standard 15 decimal digits of precision. The numbers expand their precision when they detect that it is necessary.) There were several readily available algorithms for formatting numbers to a reasonable amount of significant figures, but I saw weaknesses in each of them. I made my own formatting algorithm and implemented it. Also, there is now a slider on the toolbar that allows you change the number of significant figures displayed.

A.2 Adding tabs

I added tabs so users could have multiple problems open once, like the tabs on Firefox for multiple webpages. Even before I added tabs, you could have more than one equation per window, but multiple tabs allow you to keep your work local to each problem. It would be inconvenient, for example, for each variable to be allowed only one value for an entire assignment. This was a pain to implement, because of the code style issue described in Section 4.3.

In order to make the tabs, I had to change the type of file that Math Drag'n saves and loads, so it could hold multiple problems. I upgraded it to an XML file type. I used Java Bean XML serialization, which is good, because now I could slightly change the file type rules, and newer versions of Math Drag'n will still be able to load the old files. This required learning a great deal of computer science theory to get right.

Math Drag'n, the way James wrote it, used to be able to invisibly connect with Mathematica so that Mathematica could do some of the heavy lifting. It could also connect to Maxima [5], which is like Mathematica, except it is free, open-source, and much less pretty. With the help of one of these external CAS's, Math Drag'n could instantly solve equations, or even systems of equations for a given variable.

I ripped out Math Drag'n's ability to connect with an external CAS, because of the problems that I mentioned in Section 2.1 with using these types of CAS's to solve equations. Ripping them out also makes it easier to use Math Drag'n the first time you start it up, because you do not have to help Math Drag'n find your installation of Mathematica or Maxima.

A.3 Rewriting the user license agreement

I wrote a custom user license agreement. Math Drag'n uses libraries that were made elsewhere, I had to research the licenses of those libraries to get Math Drag'n's license right.

A.4 Rewriting help documentation

James gave Math Drag'n a professional-looking help documentation window. It had well organized and indexed pages of text that described how to use certain parts of Math Drag'n. Unfortunately, it said almost nothing about how to drag. This hole in the help was the main reason why it took me 2 hours to solve the lens equation, Eq. (2.1), the first time that I tried to use Math Drag'n. However, I am glad that James did not attempt to describe how to drag things in Math Drag'n, because like I said at the beginning of Chapter 4, the best way to learn about Math Drag'n is to *see* it in action.

I made a webpage with screencasts showing how to use Math Drag'n. You can find it at <http://ludlowcloud.com/math/help/>. I purchased webhosting and learned how to author a website, so that I could make this page. The screencasts on the page are good, but they could be better, and I know more how to do that with the hindsight I gained. I am also currently working on making the website not look so much like it was made by a physicist. Right now, the page is ugly but useful.

I removed the existing help window. Instead of that offline help system James made, I wanted an online help system so I could change the help documentation without releasing a new version Math Drag'n, especially at this beginning stage of Math Drag'n, when the help documentation can evolve quickly. More importantly, I did not want users to get stuck reading help documentation, when they should be watching screencasts.

A.5 Making the program ready for distribution

Math Drag'n was written in Java code. All java programs need something called the the Java Runtime Environment (JRE) to run. The JRE is installed on a lot of computers, but Math Drag'n needed to be easy for everyone to run. I knew how to run the program when I got it from James only because I knew how to use Java. I needed to make it easy to run for everyone.

On a Windows machine, files with the extension .exe can run without the JRE installed. There

is an \$2500 program called Excelsior JET [15] that can compile Java into an .exe file. I sent an email to Excelsior explaining my project. They gave me a slightly chopped down but free copy of Excelsior JET. Now every fast enough Windows computer can run Math Drag'n, without even installing anything.

Files with the extension .exe do not run on Macs, but all somewhat modern and unchanged Macs have the JRE already installed anyway. This makes it so I could just compile Math Drag'n into a .jar file (Java ARchive file), and it would run by double-clicking on it. Math Drag'n is complex enough, though, that it does not naturally compile into just one .jar file. Math Drag'n uses several libraries of Java code that each have their own .jar file. I figured out a way to compile all the .jar files into one.

Unfortunately, it is easy to decompile .jar files. That means that it would be easy for someone to unscrew the .jar file, take out all the original source code, and use that code to make their own program. I wanted to control who has access to the code, especially since I want to start a business from this someday. The best option was to obfuscate the code. That means to change the names of everything in the code to junk names, so that when the .jar was decompiled into Java code, the code would not make any sense. I keep the code in an un-obfuscated form so that I can easily edit it, then right before I create a new distributable version, I use a program called ProGuard to obfuscate it.

It was unusually hard to successfully obfuscate the code, in a way that would not break it, because of how James programmed it. Some of his code, while the program is running, uses word manipulation to piece together the names of which sections of code should be run next. This is unusual, and took a long time to figure out. All in all, it took about 60 hours just to make the program into a single obfuscated .jar file, so that it could easy to run on a Mac.

The .jar file that I made for Macs actually runs on Linux and Windows machines as well, as long as a complete version of the JRE is installed.

Both the .exe and the .jar versions of Math Drag'n are standalone. That means that you do not need to install Math Drag'n to use it. You can even run the program while it is sitting on your flash drive. This is helpful when using school computers, especially since students typically do not have the administrative privileges that are needed to install programs on school computers.

Index

Algy, 10
Aplusix, 10
calculator, 7
CAS, 5, 7, 40
Clickable Math, 12, 13
Computer Algebra System, 5
Equation Transformer, 24
Excelsior JET, 42
exe, 41, 43
Firefox, 40
flash drive, 43
Grapher, 17
GUI, 25
jar, 42, 43
Java Beans, 40
Java Runtime Environment, 41, 42
JRE, 41, 42
Linux, 42
Macs, 17, 42
Maple, 5, 11–13, 17
Mathematica, 2, 5–7, 13, 17, 40
Maxima, 5, 40
Moodle, 28
obfuscate, 42
paper, 2, 3, 5, 7, 9, 10, 13, 23, 28, 30–32
printing, 20, 32
ProGuard, 42
ribbons, 31, 35
Sage, 5
serialization, 40
SMG, 11, 13
STACK, 28
Symbolic Math Guide, 11–13
TI-89, 5
TI-Nspire CAS, 5, 11
user experience, 25
Windows, 42
WolframAlpha, 5–7, 30
XML, 40