Measurements of Students' Performance on Computational Exercises in Introductory Mechanics

Marcos D. Caballero[1], Michael F. Schatz[1] and Matthew A. Kohlmyer[2]

[1]School of Physics, Georgia Institute of Technology, Atlanta, GA 30332 [2]Department of Physics, North Carolina State University, Raleigh, NC 27695

Abstract

We evaluated student performance on computational modeling exercises in our large (N $\sim$ 500) introductory mechanics course. The majority of students (60.4%) successfully completed the evaluation. Analysis of erroneous student-submitted programs indicated that a small set of student errors explained why most programs failed.

Background

Students taking introductory calculus-based mechanics at Georgia Tech learn to solve complex physics problems using computation in the lab and on their homework.

We delivered a proctored laboratory assignment during the last lab of three different semesters to evaluate students' computational skills basis. 1357 students were involved in the measurements. We reviewed students' erroneous code to uncover common difficulties.

Proctored Assignment

At the end of three different semesters, students were given a proctored assignment *with randomized given values*. The prompt for the assignment is shown below:

You will complete a partially constructed program that model the interaction of two particles.

One particle is much lighter than the other and both particles are far from all other objects. The force acting on the lighter particle is directed along the line connecting the two particles and is **attractive**. The magnitude is given by $F_{\text{mys}} = k/r^n$, where r is the separation between the particles, k is a positive constant, and n is an integer.

(2) After downloading this program, set the initial conditions of your computer model to the following:

Constants
Interaction for Graded Question: **repulsive**
Interaction coefficient, k: 6.3 N*m^2
Interaction strength, n: -2
Mass of less massive particle: 0.004 kg
Time step (deltat): 5e-6 s

Initial Conditions
Initial position of the more massive particle (remains still): <5,4,0> m
Initial velocity of the more massive particle (remains still): <0,0,0> m/s
Initial position of the less massive particle: <-2.9, -1.2, 0> m
Initial velocity of the less massive particle: <8.6, 8.9, 0> m/s

Keep the massive particle fixed.

Model the motion until 2.600 s have passed.

Double check all your initial conditions and constants before submitting your answers.

Overall Performance

Roughly 60% of students made no errors (syntactic or physics) and were able to solve this problem.

Sem.	Correct	Incorrect	% Correct
1	303	168	64.3
2	201	193	51.0
3	316	176	64.2
Overall	820	537	60.4

Systematically Unfolding Errors

We developed a rubric to systematically categorize errors in the programs submitted for Semester 2 and 3, $N = 369$.

Two raters reviewed the programs independently after coding a small sample section; they agreed 92% of the time.

Only Six Error Clusters Needed to Capture Most Student Mistakes

Our coding rubric allowed for $\sim$*4300 possible unique error patterns*, however, we discovered only 111 in our data. Commonalities between these error patterns was explored using cluster analysis. The common clusters that we found suggested *only a few dominant error clusters*. In fact, *over 80% of the erroneous programs appeared in just 6 clusters* (table below).

Dominant Error	%
Stuck on Test Case; Error in Force Calculation	23.8
Constructed Working Code; Error in Initial Conditions	19.8
Used Net Force Magnitude of Update	13.3
Stuck on Grading Case; Error in Force Calculation	10.8
Raised Separation Vector to Power	7.6
Force Calculation Outside Integration Loop	7.1

Students who appeared in most of these dominant clusters have made errors when calculating on the net force. Such errors might be avoided by using qualitative analysis before writing the program.

A number of students only made mistakes with their initial conditions. They did not use some of the initial conditions they were given.

Detailed Decomposition of Errors in Students' Programs

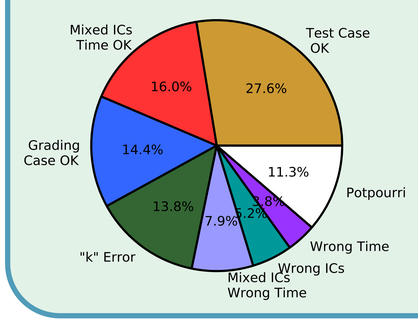
Within individual categories of the rubric, we counted the frequency of unique student errors. Only unique errors that affected more than 3% of students have been shown here.

Initial Conditions Students made a number of different errors (or typos) with their initial conditions (ICs). One interesting error involved students who mistakenly applied the exponent on the units of k to the value of k (the "k" error).

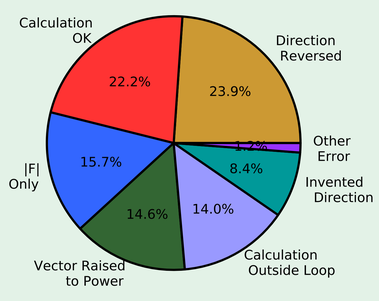
Force Calculation Most errors in calculating the force stemmed from a weak grasp of vectors or an inappropriate contextualization of the problem. Many of these errors could have been caught by sketching the problem.

Newton's Second Law Errors students made when updating with Newton's Second tended to occur because of the use of a scalar force or an inappropriate form of the update formula. Errors in this category involved a single line of the program.

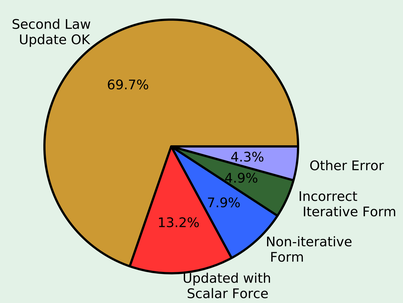
Correct Initial Conditions



Net Force Calculation



Second Law Update



Lessons Learned & Future Work

After solving a suite of computational homework problems, *a majority of students can successfully model a novel situation* without outside assistance. Success on this assignment did not depend on students' prior experience in computation.

Error that students made were typically similar to those they make on analytic problems, but some are unique to computational problems. These errors could have stemmed from a *weak grasp of vectors*, *poor contextualization* of the problem or an *inability to parse programming errors*.

Errors might be corrected by addressing each error in turn with additional practice. Our work suggests that instructional efforts should focus on developing teaching students how to debug programs. Debugging includes *identifying syntax errors*, but, more importantly, *learning the type of qualitative analysis* used for solving analytic problems.

Funding

Supported by the National Science Foundation DUE-0618519 & DUE-0942076