# Thinking Inside the Box:
# A Normal-Force Experiment Yields Unexpected Insights

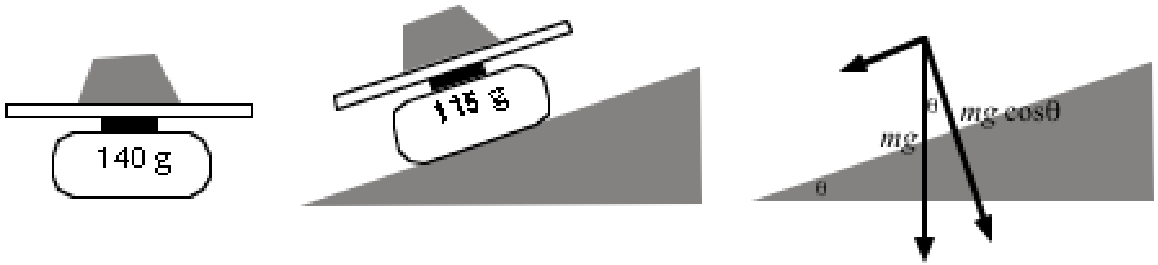
Tim Erickson, Senior Scientist
Epistemological Engineering

Every first-year physics student learns that the frictional force has a magnitude equal to the coefficient of friction times the normal force. To drive this relationship home, teachers and textbook authors invent a host of problems involving blocks on ramps.

Students also experience this relationship firsthand in the lab. As part of an NSF-sponsored project to develop physics labs and problem sets that feature more attention to data analysis[i], we were field-testing a lab in which students drag blocks up ramps at various angles when we realized that students were perplexed by the whole normal force idea.

And why shouldn't they be perplexed? First, the word "normal" does not suggest "force perpendicular to the surface" to the uninitiated. And the resulting frictional force is parallel to the surface, not normal at all. But finally, we tell students that the gravity vector decomposes into perpendicular and parallel components, but often give them no compelling reason to believe it. They never really see that forces work the way we say they do, that the block presses less against the ramp when the ramp is steeper.

A moment's reflection led us to an idea for a new lab: Put a weight on a digital scale. Now tilt the scale. See how the scale reading changes as a function of the tilt angle. If the scale's sensors read the force parallel to its axis, the tilted scale reading should be *less than the weight*. Figure 1 shows how this might work:




<note: this file is **Normal Fig.1.eps**>

Figure 1: A scale with a 140-gram mass should read somewhat less when it is tilted.

If you have done this a thousand times, we apologize. But to us, it was new, simple, and elegant. Students would see for themselves that the force on the scale gets smaller as the angle increases. In this case, with theta defined as in Figure 1, the force would decrease according to a cosine function. It seemed too good to be true.

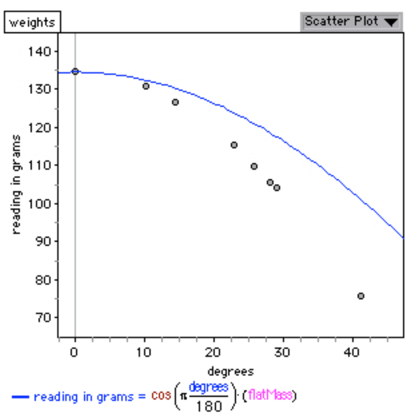
<Insert box about here, the text and illustration are at the end of this document.>

Let’s see what went wrong.

**<ScaleOnBox.tif>**

First, you have to keep the thing you're weighing from sliding off the scale. This we anticipated, so we chose to weigh a blob of clay. Then you have to find a good way to measure angles. A protractor will serve, though you should be careful that the vertex of the protractor (which is *not* usually on the protractor's bottom edge) is lined up on the slope that you're measuring. But you can be even more precise if you just measure the triangle. This proved a good choice. We set the scale on a box, and tilted the box. By measuring its length and the height of the raised end (call these distances **L** and **h**) we could derive the angle.[1]

All this went swimmingly. We got our data and plotted it in Figure 2, along with the obvious mathematical model (the "flat" mass times the cosine of the angle). We're using Fathom,[ii] a data analysis package we like.




**<normal force 1.jpg>**

Figure 2: Our first attempt at fitting the data

The model does not fit the data. We have to fix it, and that's where this story leads. We actually found three approaches to the problem: one simple and treacherous, a second revealing delicious new territory. After we discuss these two, we'll mention a third path, simple, accurate—and much less interesting.

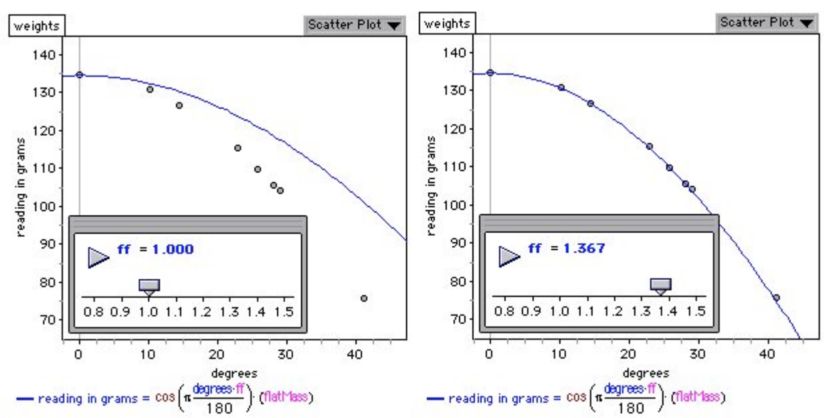
---

[1] Ultimately we want the angle only to take its cosine. That is, we can do the problem with similar triangles and the Pythagorean theorem without ever treading on trig

## *Simple and Treacherous: Curve Fitting*

Looking at Figure 2, we observed that the shape of the data was about right, but that the cosine function, as we had written it, did not curve down fast enough.

One possible way to fix that is to squish the cosine horizontally: we'll invent a parameter named **sf** for "squish factor" and use cos(**sf** * θ) instead of cosθ. But how big is **sf**? In Fathom, we create a *slider*—a variable parameter. Then we move the slider, which changes the function continuously, until we match the data as well as we can. [2] Figure 3 shows the original graph with the slider (see the function at the bottom with the variable degrees multiplied by **sf**), and then the same graph after we have moved the slider to a new value:




<note: this figure is from 2 files: **normal force ff_1.000.jpg** and **normal force ff_1.367.jpg**>

Figure 3: A "squish factor" **sf** of 1.367 appears to bring the model to the data.

Before we comment on what's wrong with this, let's acknowledge what works.

[2] You can so a similar thing in Excel by inserting a "scroll bar" and using its value in a calculation. Alas, the vanilla scroll bar takes only integer values, so you should, for example, divide it by 100.

First, just dragging the slider and seeing the function change gives students an almost kinesthetic feel for how changing the value of a parameter changes the appearance of a function.

Second, many students put the squish factor in the wrong place, or at least move the slider first in the wrong direction. But the software gives them immediate correction and feedback by being dynamic. They move the slider the direction they think; the function goes the other way. If they think about it, they get a quick slap of cognitive dissonance that supports a correct understanding of functions (e.g., "oh, yeah, if I increase the coefficient, the waves squish together, so the curve moves *in* instead of *out*…").

Finally, the function we have chosen does match these values well. This means that our initial observation that the data seem to be the same shape—by no means a sure thing—was correct. This fact may be surprising if you were worrying whether a real-life electronic scale, which is designed to work on the flat, would measure only the perpendicular force when we tilted it. The excellent match of this fudged curve suggests strongly that something is right about that idea.

### *What's Not to Like?*

Plenty, as it turns out. When we have computing power, we often try to fit data by using more and more parameters. You can bludgeon any data into submission with enough fudge factors. (This is especially dangerous if students can get polynomial regression at the touch of a button: if a quadratic doesn't look good, try a cubic or quartic function!) So we ask the students, whenever possible, the *meanings* of those parameters, and what their values tell us. For example, **flatMass**—the amplitude of our cosine, the reading of the scale when it's flat—is the actual mass of the clay blob. But what is **sf**? Why is it greater than one? What does it mean that it is 1.367? That's not so easy to say. Was there something wrong with our model of how vector components work, so that to find the normal component, you always multiply the angle by 1.367? Hardly.

It is not necessarily a bad thing to look at the data first and fit a curve. We often explore the data to figure out the functional form and find the values of parameters, and explain them afterwards. Frequently, learning that a relationship is quadratic (say) is a great help in understanding what's going on. But in this case we had a conceptual model before we ever took measurements. We had better admit that there is something wrong—or missing—in our conceptual scheme, and then match the data with a corrected mathematical model rather than simply inserting a fudge factor into the formula.

### *Getting It Right and Understanding It Too*

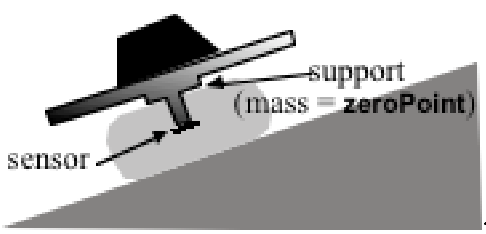
How shall we correct our model? In this case, it helps to think about the instrument itself. How does an electronic scale (or a force probe) work? Here's one possible mechanism: The blob of clay sits on a platform, which in turn has a support that sits on some kind of sensor (in fact, a strain gauge), made of a material whose resistance changes depending on pressure. The scale's circuitry converts that resistance to a voltage and feeds it to an analog-to-digital converter. A chip takes this digital value, applies any corrections and scaling factors (to convert voltage to grams, for example) and displays the reading. This conversion is probably linear, as in

**reading = (conversion factor) * (sensor voltage) – zeroPoint**

The key to our problem is the intercept we call **zeroPoint**. Consider the "tare" function of the scale. When you put an empty beaker on the scale and press "zero," the scale will subsequently display the mass of whatever you put in the beaker. That is, *it subtracts the mass of the beaker from future readings*. How does the scale accomplish this? It remembers the mass from when you pressed the button, and incorporates it into a new, larger **zeroPoint**.

What is the actual value of **zeroPoint** when there is no beaker? Probably not zero. Consider: with a glob of clay on the flat scale, the voltage from the sensor reflects the mass of the glob plus the "support" (everything upstream of the sensor itself: the platform and whatever supports it). That means that the scale must subtract the support mass to yield the mass of the clay. So this **zeroPoint** is the mass of the support.

What happens when we tilt the scale? If our understanding of the scale is correct, the pressure on the sensor comes from the normal component of the weight above it—both the glob and the support. Thus the sensor's output is the cosine of the angle times the *combined* mass. But then the scale subtracts the *un-tilted* **zeroPoint**, the full mass of the support. That is, the scale subtracts too much to account properly for the weight of the support, so it will display a weight lower than the "true" weight of the glob. This is what we see in the graph: the points are below the curve.

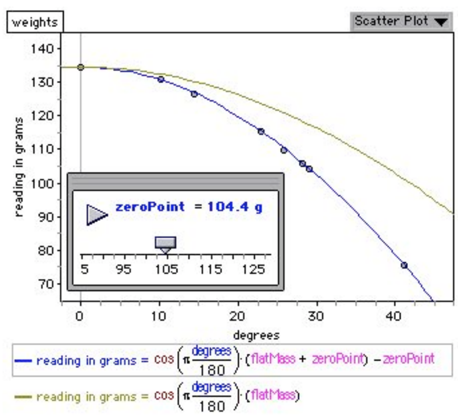


<this figure is **support illus.eps**>

Figure 4: The sensor reads the tilted combined mass of the support and the glob, but then subtracts the *untilted* support.

To account for this effect in our data, we need the mass of the support. While we might open up the box and disassemble the scale to weigh its parts, let's figure it out indirectly. As before, we make a variable parameter: a slider, which we call **zeroPoint**. Though we do not yet know its value, this slider has a physical meaning—it is not a miscellaneous fudge factor. And we can use it in calculations. For example, **flatMass + zeroPoint** is the combined mass of the glob and the support. We can use a function to model the data, and simply drag the slider until the curve matches the points. In Fathom, this looks like Figure 5, which also shows the uncorrected cosine curve. The function is

**reading = [cos(angle) * (flatMass + zeroPoint)] – zeroPoint**




**<normal force zero.jpg>**

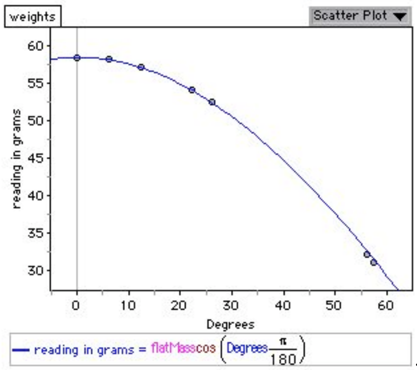
Figure 5: Our improved model fits the data *and* yields additional information.

Our altered function again fits the data, but this time we got additional information: the mass of the sensor apparatus—104.4 grams.

## *The Third Way*

Do you really have to go though all this to show that the normal force works the way it should? No. If you take the clay off the scale between measurements, and *zero the scale at each new tilt*, the data match the cosine astonishingly well. You can see one teacher's data in

Figure 6. (One excuse is that we had left the clay stuck to the scale so it would not slide off; another is that this scheme does not work with all scales.)



<screen shot. **Normal force BHS.jpg**>

Figure 6: Zeroing the tilted scale simplifies matters.

Remember that the original point of the activity was to verify the idea of decomposing vectors when we find the normal force. To avoid the whole **zeroPoint** rigamarole, simply zero the scale when it is tilted.

## Conclusions

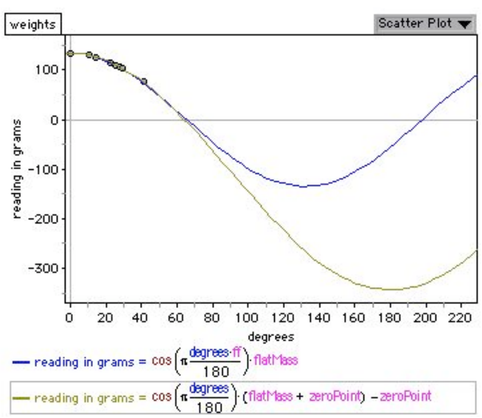
But what glorious rigamarole it is! Sometimes our mistakes result in the most surprising insights. Look what happened here: we had a conceptual model and took data to verify it. But the data did not conform. So we expanded our model, this time looking at the measurement apparatus itself. By thinking of how the apparatus could work, we came up with yet another conceptual model to graft onto the first. We expressed that model mathematically as a function. With that function, we matched the data.

Note that, of course, we did not prove that either of our models was correct, but we have evidence now that they are more plausible. It is more believable that vectors really do decompose, and electronic scales really do subtract out the zero point—and all that without resorting to “received” knowledge of the textbook or the scale’s internal diagrams. In fact, we learned important things about the inside of the scale without ever opening the box.

The preceding discussion is also redolent of the process of science at its juiciest. Let us go one step further.

Were you surprised that our final model—where we moved the curve *down* to meet the data—was the same as our first, fudged correction, where by increasing the “squish factor,” we moved the model *left*? Isn’t it odd that those two cosine functions are the same?

Fortunately, they are not. Figure 7 shows both models. Both fit the data well over the range that we have, but diverge for larger angles. This means that while we distrust the first model because the fudge factor has no physical basis, we can actually test “fudge” against “zeroPoint” because we have a quantitative prediction.



<screen shot. Appears in **normal force 2 models.jpg**>

Figure 7: Comparing data with both the squish factor model (upper curve) and with the improved model that gave us the support mass (lower curve).

As often occurs in real science, the test to distinguish these rival models must take place under extreme conditions. You will need great precision or unusually large angles, nearly—or past—vertical, where the scales read negative, and where additional measurement problems (e.g., keeping the scale's platform from falling off) will surely arise.

### An Additional Note

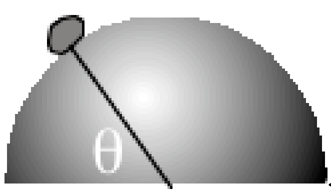
You might have noticed that the variable was named **reading**. This name helps clarify what's really going on, but came to us only after we had used up other names that turned out to be bad (such as **mass** or **normalForce**). It's curious that a name like **mass**, which is often what we read from an electronic balance, can be so misleading.


### Acknowledgements

The author is grateful for helpful comments from Bryan Cooley, Richard White, and especially Jeffrey Adkins and his students of Deer Valley High School in Antioch, California, who tried this activity out with real live, lively students.


### The Box (appears on earlier page)

Some students remember the formula ($F_{normal} = mg\cos\theta$) and not the concept. It helps to give them problems where the usual formula does not apply, such this one:



<note: illus is in **bag on sphere.eps**>

A beanbag sits on a hemisphere as shown. The coefficient of static friction between the hemisphere and the bag is $\mu = 0.5$. For what values of $\theta$ will the bag stay put?

---

[i] Erickson, T. and Cooley, B. 2004. *A Den of Inquiry*. Oakland, CA: eeps media. In preparation.

[ii] KCP Technologies. 2000. *Fathom Dynamic Statistics Software*. Emeryville, CA: Key Curriculum Press.