

Why Is DragginMath Different?

... *and what do I get for it?*

©2020 brising.com

In 1964, author Marshall McLuhan wrote a sentence that immediately became famous: “The medium is the message.” If you were alive back then, you heard this a lot. What does it mean?

Partly, it means that our media create much of our culture simply by being what they are. For example, making marks on a page both creates and limits the possibilities for what we can communicate through writing. There is the old proverb: “A picture is worth a thousand words.” What does it tell us? That words are not so good at conveying images, which are a *different medium*. What about sound? It is possible to write music, but few people read music well enough to know how it sounds just by looking at marks on a page. Written music works, sort of, but for most of us it is not as effective as sound recording, which is a *different medium*. And books and movies can both tell stories, but they do it in different ways. In each case, the medium itself determines the messages you can make with it.

Mathematics has traditionally been written on paper, and the mechanisms of writing on paper played a large part in the creation of the notation we use today. If you have ever tried to do algebra with someone over the phone, you have experienced how badly our written notation works in that medium.

With the advent of computers in the 1950s, the way math was written had to change, and quickly: the computers of that era simply couldn’t use traditional math notation. As computer designs matured, it eventually became possible to write the traditional notation on them. But *writing* the math is still not the same as *doing* the math.

DragginMath is a computer program that doesn’t just help you *write* the math. It helps you *do* the math. And it is implemented on the iPad. At first glance, you might think this means it will be just like math on paper.

But here is an interesting question: while this is something that *can* be done, is it something that *should* be done? Why is this question interesting? Because, while in some ways the iPad is similar to paper, *it is not paper*. It is not even *mega-paper* or *paper-on-steroids*. It is a *different medium*.

A good design for an iPad app like DragginMath needs to notice this, and then make use of it. A bad design might try to ignore this altogether.

DragginMath *writes* math much as you write math on paper, but it is not quite the same, and with good reason. DragginMath *does* math much as you do math on paper, but it is not quite the same, and with good reason. Here is the reason: *DragginMath uses the iPad as a different medium*. Slavishly copying what math-on-paper is and does would only make it less effective. Is that really what you want?

If you are learning algebra for the first time, DragginMath was designed to help you past some problems people often have with this new experience. It shows you a useful way of thinking about algebra that is hard to show on paper, but easy to show on an iPad. And it helps you understand that *what math means* is not the same thing as *how we write it*.

You still need to learn to do algebra on paper. DragginMath is not a substitute for that, but

it can help you get ready for it. And while DragginMath was not deliberately designed to be fun to use, most people discover that it is. If that makes your learning task easier, we won't complain. You probably won't, either.

If you already know math-on-paper, you have some new things to learn. Will it be like learning math all over again? No. You are already most of the way there. What you need to do now is bend your existing knowledge to fit this different medium. You will quickly find that not much bending is required. You may also find that math facts you never thought much about before have now become more important, and even useful.

The medium is different. So is the message. The underlying math remains the same.

Which Learning Issues Does DragginMath Address?

First and foremost, whatever results you get in math must be correct. Unlike most human endeavors, math is one in which answers to questions are either right or wrong. This is one reason some people like math.

When you are learning math, it helps to have an oracle that can tell you if what you just did is correct. DragginMath is such an oracle: anything DragginMath lets you do *is correct*. Whether it is what you *want* is a different question, and that is the thought space you get to play in. If DragginMath can't verify that what you are trying to do is correct, it simply won't agree to do it. So even if you are doing algebra on paper, DragginMath can help you check your work, one step at a time.

There is a difference between *what math means* and *how we write it*. Math is often taught as if these two distinct aspects were the same thing. But they are nowhere near the same thing. DragginMath pulls algebra far enough away from its traditional written form to make it clear that they are different things. Then you can learn each of these things for what they are, instead of trying to learn some confused combined concept that doesn't really work.

DragginMath presents math expressions as tree diagrams of operators and operands. How it translates written expressions into diagrams is shown to you, step by step, for every expression you enter. Learning how to translate a diagram back to a written expression is a classroom exercise that is both simple and fun.

For lack of an alternative, expression parsing has always been taught a particular way. With enough practice, most students figure it out eventually, although many learn it only imperfectly. Some never get it. DragginMath provides the needed alternative: a simpler technique from Computer Science.

Much of traditional algebra instruction focuses on *rewriting rituals*: for any equation that looks like *this*, you can rewrite it as an equation that looks like *that*.

But learning to see all the possible variations in *this* and *that* causes many students to simply give up. Someone who understands the structure hidden in our math notation has some hope of succeeding. Someone who doesn't understand that structure has none. DragginMath shows you that structure explicitly. It also gives you tools to work with that structure directly.

What Does DragginMath *Not* Do?

DragginMath does not do the math *for you*. It helps you along by handling the so-called *busy work* of rewriting and redrawing, but you must tell it what to do. You will do that mostly by dragging parts of pictures around on the screen with your fingertip.

Several apps that are currently available ask you to enter an equation, then show you all the details of how you *could have* solved it. But here's the problem with that: *you* didn't solve it. If your goal is to learn to do algebra yourself, that might not be the kind of help you need.

DragginMath is not a function-graphing app. It is about symbolic algebra *only*. If function graphing is what you want, there are apps for that. Many of them are excellent, and for some topics in math instruction, they are the tools of choice. But before you can use those tools effectively, you need to know the kinds of things DragginMath can help you learn.

DragginMath is not a calculator. Its screen keyboard does not even have a decimal point. It does the kind of arithmetic needed for solving *algebra* problems. This includes integer factorization for fraction and root reduction, which it does without being asked. If you want to teach or learn *computational* math, there are apps for that.

DragginMath does not implement or show the *tricks* that are often found in algebra books or passed along by teachers. That is because such tricks are not really algebra. They may be helpful in some situations, but they also confuse many students, leaving them unsure what algebra is. DragginMath does not know tricks. It knows the properties of algebra, and exercising those properties under your direction is the *only* thing it does. That turns out to be enough.

DragginMath is not curriculum. Whatever algebra book you have, DragginMath can help you teach it, or learn it.

DragginMath does not do all of algebra. It does quite a lot of it, and upcoming versions of the app will do more. Expect to see new operators covering more of the algebraic landscape. Also, we plan enhanced versions that take the user interface in even more powerful directions. But in our estimation, once you know enough to start a traditional calculus course, even some future version of DragginMath has probably taught you everything it has to offer. We could be wrong about that, though.

What Are The Tradeoffs?

Designers and engineers have a lot of experience with tradeoffs: to get more of *this*, one may have to give up some of *that*. And it often isn't just a matter of this and that, but several different things, all pulling in different directions. It is nice when one can have everything, but usually one must make choices. Hopefully, they end up being good choices.

We want you to use DragginMath. We hope you like it. We hope you find it helpful and useful. And we want you to know and understand the tradeoffs we had to make when designing it, and that you will have to make when using it. There is no point in pretending such tradeoffs do

not exist, and you will get more out of this product if you know what they are.

The primary design intent of DragginMath is to show and do algebra in a Direct Manipulation Interface: the app converts written equations into *pictures with behavior*, where you cause the behavior to happen by dragging parts of pictures around on the screen with your fingertip.

Should these pictures mimic traditional math notation exactly, or are *operator trees* more useful? We decided to use operator trees. They show the structure of equations explicitly. Also, it is easier to guess what drag gestures within a picture are likely to mean.

Translating the traditional notation of binary operators ($+$ $-$ $*$ \div) into operator trees is simple. It becomes harder when we talk about exponents and roots. Exponentiation traditionally has no visible operator symbol. Instead, it uses a *typographic convention* (x^n), where you are expected to *infer* the existence of the operator when you see characters written that way. Roots are traditionally represented with the radical symbol alone for square roots (\sqrt{x}), and a combination of the radical symbol and a typographic convention for other roots ($\sqrt[n]{x}$). Traditional notation needs to do similar things to talk about logarithms ($\log_n x$). These traditions work well enough with pen and paper or in typesetting systems, but they work badly when interacting with a computer, which is a *different medium*.

We know of two ways to approach this problem, and they both involve a tradeoff. One approach looks more traditional, but it creates special cases for you to learn in the user interface. It also forces practical limits on the algebra problems you can write. The other approach looks less traditional, but the user interface works in a uniform way, and it opens up more algebra for teaching and learning. We chose the second approach, treating all functions in DragginMath as operators with explicit symbols: $\text{raise}\uparrow$, $\text{root}\downarrow$, and $\log\downarrow$.

Using these symbols for raise and \log is not much to ask. It is more of a problem for roots, where $n\sqrt{x}$ traditionally means n times the square root of x . In DragginMath, it means the n th root of x . As a tradeoff, you get two things for this: 1) a wider range of mathematical behaviors to work with, and 2) everything in the user interface looks and works the same. There are no special or odd cases in DragginMath. This makes the app easier to learn and use. Students also have an easier time understanding that these things really are all pretty much the same, even if traditional notation chose to write them differently.

Each of these operators has both binary and unary forms, where $\uparrow x = e\uparrow x$, $\sqrt{x} = 2\downarrow x$, and $\downarrow x = e\downarrow x$.

We did not make these choices casually. We understand there is a cost. After a lengthy examination of these tradeoffs, it became clear that DragginMath can do much more math much more simply by taking the less traditional tradeoff in this different medium.

For some of the work you do in algebra, paper is easier than any app will ever be. For other things, DragginMath will make you laugh at even the thought of working on paper. Sometimes, these two categories live very close to each other in your work. You will often feel the pull in both directions when solving even a single algebra problem.

Paper is easier to use when making simple rearrangements of terms in an equation. It's like you throw all the terms into the air, mentally swat them around while they are up there, then see them fall back onto the paper in their new arrangement. This works amazingly well,

assuming you do it correctly and you don't mind the actual rewriting.

Rearrangements like this are based mostly on the associative and commutative properties. They assume two things: 1) you *understand* these properties, and 2) you use them correctly *every time*. This is not hard for experienced people, who don't have to give it much thought. In the back of our minds, we secretly tell ourselves that "Through some application of the associative and commutative properties, this all works out. Just don't ask me for the specific details of how it is done in this equation. I know the result is right." And *usually*, it is.

If the person doing the algebra doesn't fully understand these properties, or doesn't use them carefully enough as those math symbols are flying through their mental atmosphere, bad things happen. The safer assumption with beginning algebra students is that they *don't* fully understand and *often* make mistakes.

When rearranging terms in an equation, DragginMath feels like more work than paper. But it *does* understand the associative and commutative properties, it *never* makes mistakes with them, and it won't let the student make mistakes either.


In the course of developing DragginMath, we discovered a way of using the associative and commutative properties on the screen that may surprise you. Term rearrangements that initially seem daunting turn out to be easier than expected. But you would probably never think of this technique, or bother to use it, when working on paper. It was news to us. It is only in this *different medium* that this technique becomes useful.

For which parts of algebra is DragginMath easier than working on paper? Pretty much *everything else*. Complicated distributions, factorizations, evaluations, and solutions can be done reliably with single small motions of your finger on the screen. Or you can break them into smaller transformations to see or show how these compose together.

There are things DragginMath *could do* that we chose not to have it do. For example, with the expression $a \div a * b$, cancelling the $a \div a$ leaves $1 * b$. DragginMath could eliminate the 1, leaving just the b . Why doesn't it? Because DragginMath is a teaching tool, not a professional tool. We believe it is important for students to see and know about the existence of the 1 in this situation, even if the next step will almost always be to eliminate it. There are several places where we made this kind of design choice. As you become more proficient in algebra, you will become more aware of these. You might choose to view that awareness as your growing mastery of the subject.

With some exceptions, traditional math notation is one-dimensional: characters written in a line. But paper is a two-dimensional medium. You can put that second dimension to good use by writing one version of an equation after another, neatly down the page, as you work on it. This allows you to examine the path of your thinking about that equation.

The iPad is also a two-dimensional medium. DragginMath uses those two dimensions differently, instead showing the internal structure of equations as a tree diagram. You can undo and redo changes as much as you like, but DragginMath does not currently show successive versions of an operator tree at the same time, the way paper can. Perhaps a later version of DragginMath will do this.

What you *can* do now is tap the  button to see the history of your work. On this screen, you can see the successive versions of an equation as you worked on it. These are shown in DragginMath's linear notation, not in operator trees. We plan to make this even easier to access.

Is it possible to make this kind of user interface work on linear notation? Yes. DragginMath has this capability as a natural outgrowth of the way it is coded. We experimented with direct manipulation of linear notation. Then we added code to stop you from doing it. It is counterproductive to DragginMath's goals as a teaching tool. If you want a direct manipulation interface on algebra, we believe two-dimensional operator trees are a better way to go.

DragginMath does allow you to convert operator trees into linear notation and back again whenever you like. Tap the \Uparrow button to see a tree fold up into a line of characters, with parentheses injected as necessary. Tap it again to see the linear notation fold back down into a tree. Use this to persuade yourself that operator trees and linear notation really do mean the same thing.

Earlier in its development, DragginMath let you fold any subtree into linear notation. This complicated the user interface, and users found it more troublesome than helpful.

In the first lecture of Dr. Marion Diamond's human anatomy course on YouTube (one of the most-viewed academic courses ever on that website), she tells you that she deliberately moves slowly through the material so you will have time to copy all of it accurately into your notes. This is not just so you can read your notes later. It is because the act of writing and rewriting helps you learn.

DragginMath's most obvious benefit is that you write a lot less, and maybe not at all. This is a drawback also, as the previous paragraph makes clear.

DragginMath is a powerful tool that can help you teach, or learn. It is not the only tool, and there will be times when *you* need to choose the right tool for the job.