Transactions on Engineering Sciences vol 15, © 1997 WIT Press, www.witpress.com, ISSN 1743-3533

Mathematica in Education: Old Wine in New Bottles or a Whole New Vineyard?

P. Ramsden
METRIC Project, Mathematics Department, Imperial College of Science, Technology and Medicine, London, UK

Abstract

On its own, a new piece of technology is not enough to change anything very much. Things change, in any field, not through technology alone but through the way people use it. How people use it depends in turn on how they think about it. In the early history of many technological innovations that went on to shape modern life profoundly there was a period in which the innovation was seen, and therefore used, mainly as a new way of doing old things. The revolutionary potential of new technology lies, however, in our finding new things to do with it.

The impact upon educational practice of powerful software like *Mathematica* has been less profound than optimists hoped or pessimists feared. In many classrooms, I argue, it may be used as an adjunct to a curriculum and pedagogy unaltered in its essence. I here compare some possible approaches to the use of *Mathematica* with students, and ask of each one how close it comes to realising the potential of the software to transform the experience and nature of mathematical learning.

Version 3 presents our community with fresh challenges and fresh temptations. I show how Version 3, at the same time as it opens up new, exciting avenues for educators, also makes it easier than before to "bend" *Mathematica* to old-fashioned pedagogic strategies that leave much of its potential unexplored.

New tools for old jobs?

One can't spend much time in technology-in-education circles without running into phrases like "the effects of calculators on children's learning of arithmetic" or "the impact of computer algebra systems on the traditional calculus curriculum". But no technological innovation, on its own, has any "effects", still less any "impact". Things don't have effects: actions do. What matters isn't what technology there is, but the way that technology is used.

Transactions on Engineering Sciences vol 15, © 1997 WIT Press, www.witpress.com, ISSN 1743-3533

Someone might object that this is just pedantry: that a piece of technology is usually designed with a purpose in mind, and that uses we put it to will reflect that purpose. If I buy an alarm clock, say, I can certainly use it to help me get up in the morning; I could probably also use it for a few less conventional purposes, such as timing a roast. It's not open to me, however, to use it to make coffee or clean the living room. The uses of a piece of technology, it could be argued, flow from its nature, and in that sense technology *can* be said to have effects, albeit—if we're splitting hairs—indirect ones.

But not all technology is like the alarm clock. It's a truism that we can't use something for a purpose for which it can't be used, but that doesn't mean that technology can't have *emergent* uses: uses that no-one, not even the designers, could have predicted. *Mathematica* is an excellent example of a technological innovation that was built with precisely this kind of emergent use in mind. The whole point about a program like *Mathematica* is that the space of possible uses for it is vast and unexplored. If that were not the case, it would be without value for a research community whose *raison d'être* is innovation and discovery, and conferences like this would be redundant. What's true for *Mathematica* in research is, I'll argue, even more true for *Mathematica* in education.

When powerful technological innovations, in any field, first arrive on the scene, they tend to be thought of as new tools for performing old tasks. Only later do we devise fresh jobs for them: things that were perhaps prohibitively difficult, or impossible, or even inconceivable before. Writing in 1980, Seymour Papert put it like this.

> The first use of the new technology is quite naturally to do in a slightly different way what had been done before without it. It took years before designers of automobiles accepted the idea that they were cars, not "horseless carriages," and the precursors of modern motion pictures were plays acted as if before a live audience but actually in front of a camera. [Papert[1], p 36].

Stephen Wolfram's vision of the future of technical publications can be seen as an attempt to use *Mathematica* to do a wholly new thing, instead of just doing an old thing better. In the field of education, one might mention (to pick one among several) Jerry Uhl and his colleagues at the Calculus&*Mathematica* project, who have used *Mathematica* as an integral component of an entirely reconceived university calculus curriculum. But both Wolfram and Uhl would, I'm sure, agree that not everyone is following their lead. The world of technical publishing seems to possess a lot of inertia, and so, of course, does the world of education.

The revolution that wasn't

Seventeen years have gone by since Seymour Papert wrote *Mindstorms*, the book from which the quotation from the last section is taken. In that time, computers have soared in power, speed and memory capacity while tumbling in price. Software has developed at the same dizzying rate, and the availability and accessibility of computers at all levels of education in the Western world is unprecedented. Yet the following quotation from *Mindstorms* would be just as true if it appeared in a brand new publication:

Transactions on Engineering Sciences vol 15, © 1997 WIT Press, www.witpress.com, ISSN 1743-3533

> In most contemporary educational situations where children come into contact with computers the computer is used to put children through their paces, to provide exercises of an appropriate level of difficulty, to provide feedback, and to dispense information. The computer programming the child. [Papert[1], p 19].

For "children" we can, of course, substitute "undergraduates". "Intelligent tutoring systems", "electronic textbooks" and multimedia CD-ROMs are, in many countries, the staple uses of computing in higher education.

It's not surprising, given the tendency noted in the last section, that we in the higher education community, when responding to the vast opportunities offered by personal computer technology, began by looking for electronic ways of doing the familiar jobs that had previously been done by textbooks or by lectures. What is odd is that these models have proved so robust, and that we have, collectively, been so slow to explore and invent in directions that were once entirely closed. While many areas of industry and commerce have been transformed by the advent of personal computers, education, for the most part, plods on with the same old stuff, sometimes half-heartedly and almost resentfully dressing it up in ill-fitting electronic garb. Even in mathematics, which we might expect to be particularly rich in possibilities, many students all over the industrialised world in 1997—perhaps most—are still learning the same old stuff and performing the same old tasks in pretty nearly the same old ways.

How should we be using *Mathematica*?

There are, however, exceptions to this dismal and uninspiring rule, and happily the *Mathematica* community continues to provide an especially high density of them. That, it should be said, is no accident. *Mathematica* is a piece of software that is built to be used, and to benefit its user. It attracts educators who want to give their students power, and not merely to exercise it over them. It's the kind of software that makes us think "how can my students use this to learn?" more often than "how can I use this to instruct?", and when we sit down to think about those two questions we generally find that it's the former that has the more natural and elegant answers.

My own conversations with *Mathematica*'s designers, and especially with Theo Gray, leave me in no doubt that all this is quite deliberate. The features of the Front End have always reflected a view of education based on the notion that learners should be active, creative and in control. *Mathematica* has never been conceived as a preformed, dead, "point-and-click" environment, as a bossy "tutor", or as a quiz machine.

But that hasn't stopped some of us trying to make it into one. My own early experiences as a designer of *Mathematica* learning materials are a good illustration of Papert's proposition that we begin by trying to shape new technology to old ends. I'll always be grateful that Theo and his colleagues made it so hard to do that with *Mathematica*, and that the very structure of the software made it so natural to place the learner at the centre of the picture. It wasn't long—it was too long, it was embarrassingly long, but it wasn't long—before I stopped wrestling with, say, how to set and mark electronic exercises in *Mathematica*, and started concentrating on a fresh set of problems, no easier perhaps but much more interesting. Problems like: what kinds of mathematics can students do with *Mathematica*?; what kinds of mathematics *should* they do now that they have *Mathematica*?; what kinds of *Mathematica* activities can one design that will get students thinking mathematically?; and so on.

Rather than exhibit my own efforts in this line (which I don't feel qualified to evaluate) I'd like to call upon some recent work of two colleagues of mine, Margaret James and Phillip Kent, as an example of what seems to me to be exciting and thoughtful innovation in the educational use of *Mathematica*. The problem that James and Kent were grappling with was the well-known tendency for students to carry pointwise, *local* conceptions of gradient into "advanced" contexts, such as differential equation theory, in which these conceptions are inadequate and must be replaced with *global* ones. They developed an educational tool in the form of a user-defined *Mathematica* function they called `TangentField`. This function takes a first-order differential equation, together with a list of coordinate pairs, and generates a figure consisting of a field of tangent stubs, each centred on one of the specified points and with a gradient defined by the differential equation. Figure 1 shows an example.

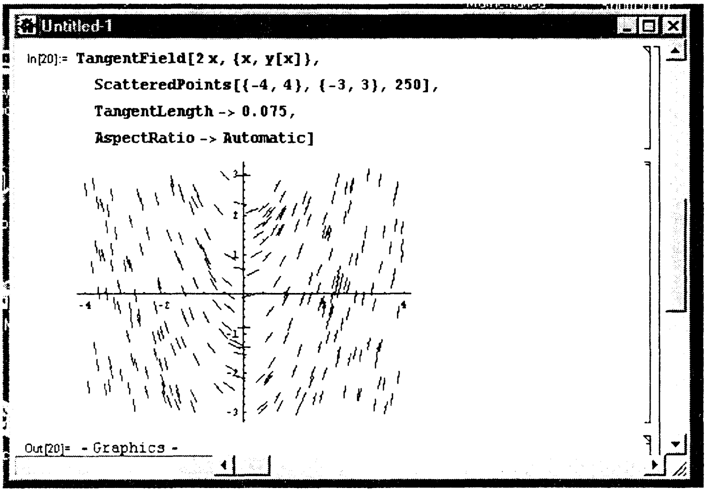

Figure 1: Field of randomly scattered tangents for the equation $\frac{dy}{dx} = 2x$.

The tangent field representation of differential equations has obvious links with the existing pointwise conceptions that students often bring to this area of mathematics. But at the same time, as Figure 1 illustrates, it provides a strong visual sense of *families of curves*: the global conception James and Kent were after: the tangent stubs, aligning themselves like iron filings in a magnetic field, form patterns whose long-range order is strikingly apparent. Students have complete control over the lists of points they use as input (one of the important differences, this, between James and Kent's tool and *Mathematica*'s own `PlotVectorField` function), and the accompanying activities are designed to allow these long-range patterns to emerge as the learner moves from single points, to small sets, to larger ones. The tool therefore has the potential to enable students to build, as they explore differential equations in *Mathematica*, strong conceptual links between the local and the global.

What are the important features of James and Kent's work with `TangentField`? They saw their job as creating a flexible, mathematically powerful tool: a tool whose features were designed with one eye on the nature of the knowledge domain and one on what they knew about the needs of learners in that area. This tool was to "belong" to the students, who would use it for doing mathematics; it wasn't designed to "belong" to the system—to set exercises, for instance, or to give explicit instruction. And armed with `TangentField`, students could do mathematics of a type that has no exact "traditional" counterpart.

And then there was 3.0

Now, of course, we have *Mathematica* 3.0. This is a very special piece of software, and we can be sure that exciting and innovative uses for it will soon come along, and that some of these uses will be educational ones. "Papert's Law", though, tells us to expect that people will begin by using it, in some sense, *conservatively*. That might simply mean that the way we use the new version in education will begin by looking very like the way we used the old one, and that only later will we come up with inspiring ways of making use of, say, palettes or typesetting. Or it might mean something a bit stronger: that there will be a tendency for people to bring a "textbook-quiz-and-lectures" mindset to *Mathematica* 3, rather like the one I began by bringing to *Mathematica* 2. It's therefore legitimate to ask whether the structure of the software still militates against that way of using it. Is it still the case that *Mathematica* is most naturally used as a powerful exploratory tool in the hands of students, rather than as a substitute for lectures and tutorials?

Yes, is the answer to that question. On the basis of my limited explorations of the new version, it's clear that the Front End design team have remained true to their educational principles, and that *Mathematica* remains one of those rare pieces of educational software that it's almost easier to use innovatively than otherwise. But I do have some disquiet: I'm not sure that "otherwise" is as hard as it used to be.

Of all the new features of *Mathematica* 3.0, I'd like to focus on three, namely:

- buttons and palettes;
- hypertext linking;
- kernel control of the Front End.

Buttons and palettes

Let's look at James and Kent's `TangentField` function again. It's clear from Figure 1 that students are being asked to grapple with some quite complicated, difficult and potentially irksome syntax when they use the tool. With *Mathematica* 3.0, James and Kent might, in principle, have decided to spare the student all this heartache by building a suitable button. An example of the sort of thing I mean is shown in Figure 2. Here the student, instead of typing the whole command, now simply has to type, and select,

```
2x
```

(or whatever) and then hit the button.

Without any doubt, students could save themselves a great deal of trouble this way. But would it be worth it? Doesn't the fact that learners can now ignore all the other inputs make TangentField more mysterious, more opaque, more of a "black box"? Isn't it less likely that students will use the function in ways not foreseen by the designers: on problems of their own, perhaps? On such "rare events" [diSessa[2], p320] are real breakthroughs in learning often built. What we have here is an example of something that might end up making them even rarer.

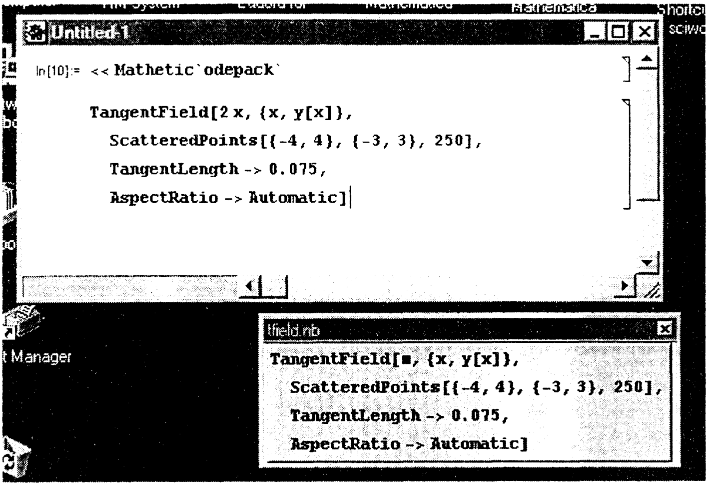

Figure 2: A "button-based" approach to `TangentField`.

None of the above applies, I think, if the button was built by the student (as a sort of home-made labour-saving device) rather than built *in* by the author. It's in students building their own buttons that the most exciting possibilities for the future perhaps lie.

Hypertext linking

With the advent of Version 3.0, hypertext functionality at last comes to *Mathematica*. And not before time, many might say. The electronic documents of the future must, of course, have electronic footnoting and cross-referencing, and hypertext is probably the best idea anyone's yet come up with for how to do that. What applies to scholarly papers also applies, surely, to *Mathematica* notebooks intended for students. Here, too, we can make use of the facilities the new version gives us for hotlinking key words and phrases to, for example, their definitions. Figures 3 and 4 illustrate the principle in action.

But while I can see that this is all to the good, I can't see that it's all that great, educationally speaking. In the end, what matters about a program like *Mathematica* aren't the ways it offers students to read things, but the ways it offers them to *do* things. It does no harm to hyperlink one bit of text with another in this way, but there's a danger of our convincing ourselves, as a community, that it does some real good. And that might lead to our concentrating on things that really aren't important, and diluting our real work, which I think, as I've suggested, is designing *tools* and *activities*.

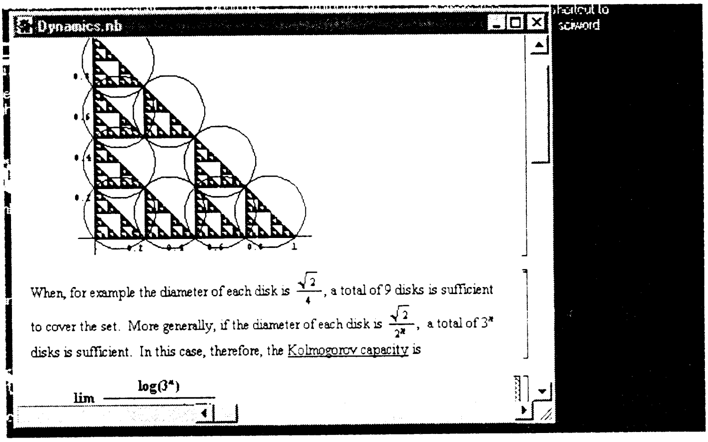

Figure 3: A hypertext link.

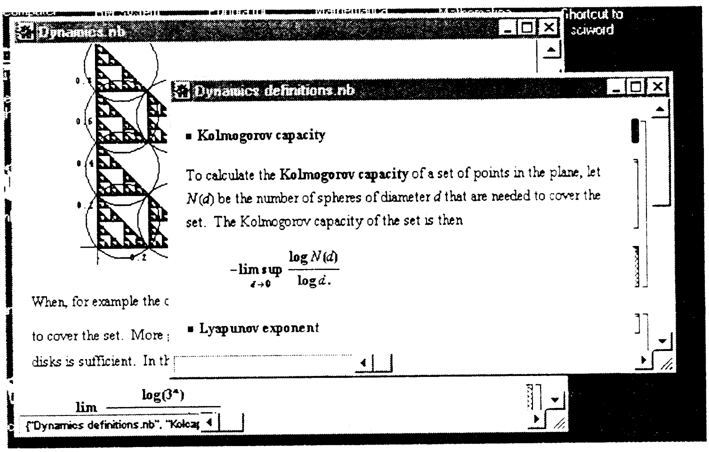

Figure 4: Following a hypertext link.

Kernel control of the Front End

I've saved the best, or the worst, till last. Earlier, I referred to the time I spent, as a newcomer to *Mathematica* in education, tangling with the problem of how to set up Q-and-A type exercises in the old Front End. It can be done, and I'm sure it can be done in many different ways, but I'm sure none of them are remotely natural. That, I'm pleased to say, is still true: to bend *Mathematica* 3.0 to this purpose, when it's so well suited to so many others, still feels contrary and perverse. But it's easier than it was.

What's made the difference is that the functions of the Front End have now been made largely subordinate to those of the Kernel, and in particular that the Notebook can now be represented as a *Mathematica* expression, which

Transactions on Engineering Sciences vol 15, © 1997 WIT Press, www.witpress.com, ISSN 1743-3533

makes it a programmable object. This is an elegant piece of computer science, which ties up several loose ends and unifies the structure of the software beautifully. One side-effect of it is that we authors now have such control over everything that, paradoxically, it's easier than ever before for us to take control *away* from the user. Figure 5 illustrates a multiple-choice test that uses Kernel control in this way, generating fresh windows and killing old ones in response to the "point-and-click" activity of the student.

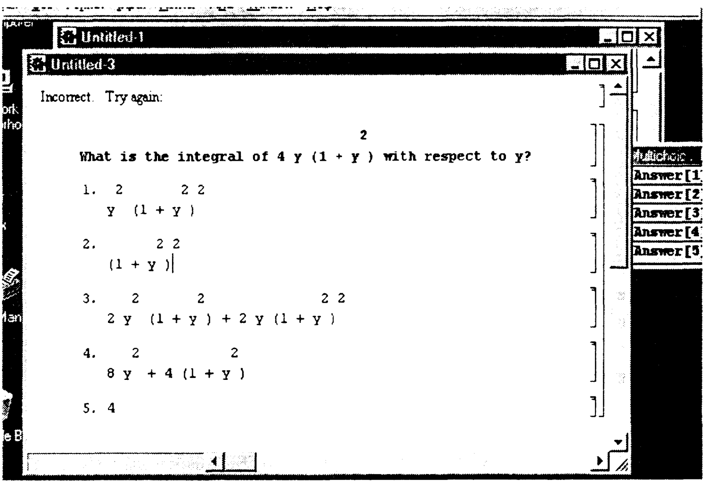

Figure 5: A multiple-choice test that uses kernel control of the Front End.

Of course, *Mathematica* is still *Mathematica*, and the user hasn't really ceded control to the system. It just feels that way, which is almost as bad.

Final comments

I've been privileged to work with *Mathematica* in education over the last few years. It's been exciting, challenging work. Now there's another excitement, a fresh challenge, for all of us working in this area. But every fresh challenge carries with it fresh opportunities to get things wrong. Version 3.0 will take us to places we don't even know exist yet. But we'll get to those places quicker if we avoid going down the many well-mapped blind alleys whose entrances are now, as a side-effect of all the improvements, perhaps slightly wider than they were.

References

1. Papert, S. *Mindstorms: Children, Computers and Powerful Ideas*, HarperCollins, New York, 1980.
2. diSessa, A. Social niches for future software, Chapter 16, *Toward a Scientific Practice of Science Education*, ed M. Gardner et al., pp 301-322, Hillsdale, NJ, 1990.