HAL
archives-ouvertes.fr

The visibility of models: using technology as a bridge between mathematics and engineering

Richard Noss, Phillip Kent

To cite this version:

Richard Noss, Phillip Kent. The visibility of models: using technology as a bridge between mathematics and engineering. International Journal of Mathematics Education in Science and Technology, 1998, 31(1), pp.61-69. hal-00190101

HAL Id: hal-00190101

https://telearn.archives-ouvertes.fr/hal-00190101

Submitted on 23 Nov 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

The Visibility of Models: Using technology as a bridge between mathematics and engineering

Phillip Kent (Imperial College) and*
*Richard Noss** (Institute of Education)*
University of London, U. K.

Submitted to the proceedings of the ICMI Study Conference, "On the Teaching and Learning of Mathematics at University Level", Singapore, December 1998.

ABSTRACT: Engineering mathematics is traditionally conceived as a set of unambiguous mathematical tools applied to solving engineering problems, and it would seem that modern mathematical software is making the toolbox metaphor ever more appropriate. We question the validity of this metaphor, and make the case that engineers do in fact use mathematics as more than a set of passive tools—that mathematical models for phenomena depend critically on the settings in which they are used, and the tools with which they are expressed. The perennial debate over whether mathematics should be taught by mathematicians *or* by engineers looks increasingly anachronistic in the light of technological change, and we think it is more instructive to examine the potential of technology for changing the relationships between mathematicians and engineers, and for connecting their respective knowledge domains in new ways.

* *(Author for correspondence)*
Dr Phillip Kent
Mathematics Department
Imperial College
London SW7 2BZ, U.K.
Tel 0171 594 8503 Fax 0171 594 8517
Email p.kent@ic.ac.uk

** Professor Richard Noss
Mathematical Sciences
Institute of Education
20 Bedford Way
London WC1H 0AL, U.K.
Email: rnoss@ioe.ac.uk

Introduction

The statement that mathematics plays a central role in engineering and science is certainly true, and—in its bare bones formulation—certainly a truism. “Mathematical models” *are* everywhere, “modelling” *is* a central activity. However, in this paper we would like to ask in what ways is a model “mathematical”, and, at the same time, in what ways is it scientific, or part of engineering? How might these different aspects be connected in the minds of learners and experts?

Traditionally, these questions have unproblematic answers. If “service” mathematics is essentially a set of tools whose workings need not be visible to the user, then the difficulty is simply one of teaching “the mathematics” and learning to apply it later. The metaphor of application is ubiquitous. But what is it that is applied? And what, exactly, is it applied to? Further, if the toolbox metaphor is to be helpful, we need to have some idea of what different people will see when they look inside the box. Will they see the same thing, will it have the same structure? Will it have the same function?

The experience of engineering students entails more and more contact with sophisticated pieces of technology. For example, with the latest computer-aided design software for civil engineering it is possible not only to “build” structures such as bridges in the virtual space inside the computer, but also to test the integrity of a design against the effects of an earthquake. Underlying this computational power is a huge amount of *invisible mathematics*, and it is clear that technology is allowing students to *use* mathematics to an unprecedented degree—in the case of computer-aided structural design, the most advanced numerical techniques for solving nonlinear equations become available at the press of a button, and with barely a mathematical equation in sight. In these circumstances, the future role of mathematics teaching for engineers is uncertain, especially since most of the mathematical methods which form the staple diet of traditional mathematics service courses are now themselves available effortlessly in a computer mathematics system such as *Mathematica* [1].

In this respect, it seems that the computer is, if anything, making the toolbox metaphor ever more appropriate. If solving a nonlinear equation is a question of pressing the right buttons, it is not inappropriate to think of it as similar to selecting the right spanners — and we don’t seem to need much instruction about how spanners work (or are designed) to use one. On the contrary, we might be forgiven for asking how the connections between mathematics and its applications in engineering can be made more visible by using computer mathematics software, which, it is commonly acknowledged, hides mathematics inside general-purpose, black box functions for doing integration, equation solving and the like?

This role of technology, together with (in the UK at least) the well-documented decline in mathematical preparation of incoming students (e.g. [2], [3]), has led some to make a reasonable case for the downplaying of the role of mathematics in engineering (see, for example, [4, p. 264]). There is, undoubtedly, an argument that significant kinds of engineering can be done with mathematics which has already been done by someone else, and wrapped up into computational tools which the engineer needs only to use.

In fact, it seems that this is a very partial view. There remains a strong case for the inclusion of mathematics as more than a set of passive tools, catalysed by the

computer in new ways (see, for example, [5]). In this paper, we will outline our case that the computer, if appropriately conceived, affords an opportunity to make visible important parts of the mathematical agenda, rather than to relegate mathematics into a set of tools whose workings remain opaque.

Our position demands attention to epistemology rather than merely technology and its application. Of course, we will need to consider technology-focussed issues, such as what is possible with a piece of software such as *Mathematica*—what can be done with in the context of a given mathematical or engineering topic. But we want to focus on issues to do with the basic relationships between mathematical and engineering knowledge. These are fundamental in our attempt to rise to the challenge of designing and structuring activities which simultaneously lead students to use *and* understand the mathematics they are deploying in their computationally-based activities.

In the UK, students specialise in their degree subject from the start. For this reason, if no other, they meet some demanding mathematics as soon as they enter university, and are called upon to "apply" it almost immediately. The example student activities in this paper are drawn from a short (6 hour) introductory course in *Mathematica*, for first year undergraduates in the Civil Engineering department at Imperial College, which was designed and delivered, for the first time, by the METRIC Project[1] in early 1998.

Developing a "structural feel" for beams and bridges

An introductory *Mathematica* course for undergraduates can easily fall into the class of generic software training; there is so much that seems to need to be discussed (symbolic calculations, numerics, graphics, programming, using the document interface), that it is easy to spend the whole time exhibiting the functionality of the software. With the Civil Engineers, however, we wanted to use the course to present activities to the students where *Mathematica* is being applied in *specific* engineering contexts, and we enlisted the help of a colleague from Civil Engineering[2] to develop these contextual examples[3]. At the very least, seeing *Mathematica* applied to relevant situations in engineering is likely to be good motivation for students, but our hope is that the "bridging" effects can be more significant than this.

The two contexts we chose were both to do with structures. The idea of the engineers was that, by letting *Mathematica* take the mathematical strain, we could help students begin to get a "structural feel" for how structures behave (something that their present courses seem to be deficient on). After the first run, we can claim to have existence theorems for this; we will be seeking more substantial evidence when the course runs again in the current academic year.

The first *Mathematica* session for the students was a quick overview of the numeric, symbolic and graphical capabilities of the software. As a final exercise, the students were invited to "apply" their fresh knowledge to a typical loaded beam problem, such

---

[1] The project team is Phillip Kent and Phil Ramsden. See `http://metric.ma.ic.ac.uk/` .

[2] Dr David Lloyd-Smith, to whom we express our thanks.

[3] METRIC has developed a similarly "contextualised" approach for Chemical and Mechanical Engineers, and for Chemistry students; for details, see the web site already mentioned.

as they meet in their initial engineering course on structures. The “structural feel” idea prompted an emphasis not on the mathematics of the problem—which is given to them in full—but on estimating important structural quantities as a load (*W*, below) is varied, using whatever combination of graphical, numerical and symbolic methods they choose. These quantities include the point of maximum displacement along the beam, and the point of “contraflexure” (i.e. where the curvature changes sign).

The students were presented with the situation shown in Figure 1, where the up-arrows denote fixed supports, there is a distributed load between x = 3 and x =6, and W is a variable point load:

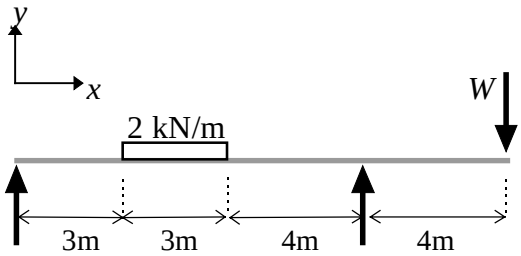

Figure 1: The loaded beam problem in Civil Engineering.

The students were given the solution for small deflections of the beam. This is conventionally written by engineers in the form:

$$EI\,y = \frac{5}{3}(4W - 22275)x + (33000 - 4W)\frac{x^3}{60} - \frac{250}{3}(x-3)^4 + \frac{250}{3}(x-6)^4 + (27000 + 14W)\frac{(x-10)^3}{60}$$

The constant term EI is typically around 10^7 so the deflections *are* very small, of order mm for y when x is of order m (Figure 2).

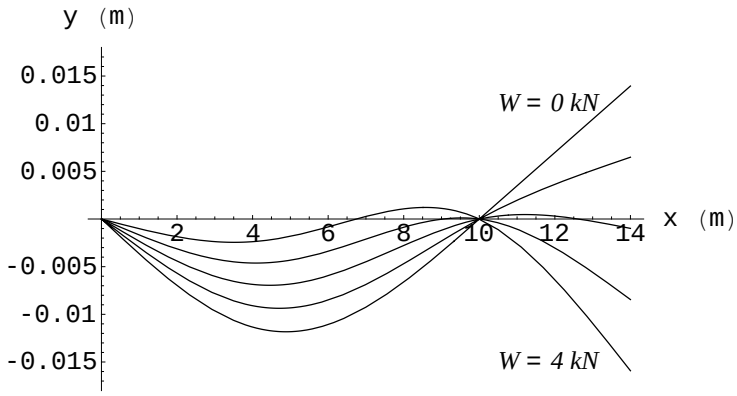

Figure 2: Graphs of beam displacement (for *W* ranging from 0 to 4 kN).

The y-vs-x equation is not quite as it seems, because the polynomial terms $(x-3)^4$ etc, are written in the normal way, but in fact represent piecewise-defined *ramp functions*, defined to be zero when $x < 3$ and to be $(x-3)^4$ when $x \geq 3$, etc. (Figure 3).

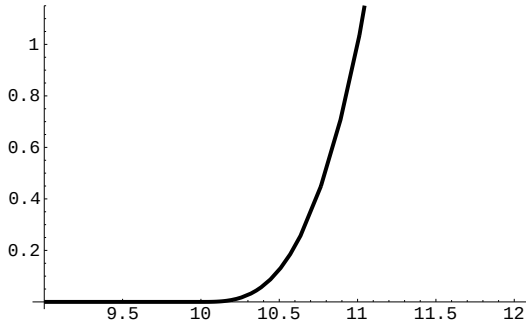

Figure 3: The ramp function "$(x-10)^3$".

While this notational shift is implicit in much of the spoken and written language used with the students, it is made explicit, indeed it *must* be made explicit, in the *Mathematica* expression for *y* which the students are given:

```
y = (1/EI)*(5(-22275+4W)x/3 + (33000-4W)x³/60 -
  (250/3)*If[x<3,0,(x-3)⁴] + (250/3)*If[x<6,0,(x-6)⁴] +
  (27000+14W)*If[x<10,0,(x-10)³]/60)
```

From the engineer's point of view, there is nothing strange; as our colleague put it: "*of course*, these terms here are ramp functions...". But for us it was surprising to discover something new about polynomials: basic and boring mathematical objects, but when looked at in a certain (engineer's) way, they *are* "ramps". Moreover, this is true in a dual sense, both as a visual metaphor, and as an expression of the role that the functions serve in the structural analysis: that the terms containing ramp functions have zero effect on the beam displacement until *x* reaches a threshold value.

We don't want to overstate the importance of a small episode, but it does point up the fact that mathematics is not a passive agent. In use, mathematics becomes a means of making sense of the underlying engineering principles. But reciprocally, the mathematics itself is shaped by its application—it takes on meanings which are derived from the setting in which it is used.

Incidentally, we discovered one other curious (to us) phenomenon: in situations like this, the beam's weight is often negligible in magnitude relative to the other forces in the problem. So the weight is abstracted into the form of a distributed load (pressure). In effect, the beam is abstracted to an "ideal structure" defined only by its geometry, flexibility and material strength.

The Rainbow Bridge

The second *Mathematica* activity for the Civil Engineers is based on second-year mathematics material, and it represents an instance of *didactical inversion*: using the capability of the technology to allow students to carry out some task using mathematics which the students don't know yet in order that they can focus on some conceptual points which the mathematics makes accessible. Two pre-written *Mathematica* functions generate animations of a test load moving across two different simple bridge structures; at each step in the animation, the colour of each of the struts in a bridge represents the magnitude of the force in that strut induced by the test load (Figure 4).

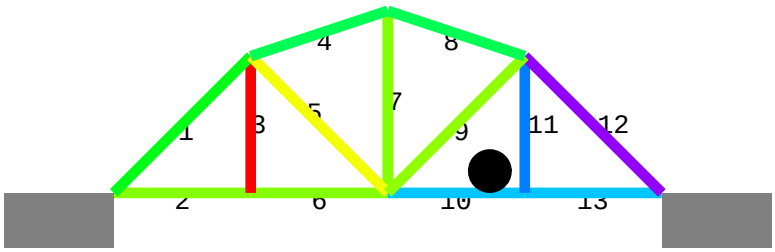

Figure 4: Frame from a "bridge movie"; a test load (black disc) moves across a (2-D) bridge, whose struts change colour according to the magnitude of the force caused by the test load (the numbers just index the struts for reference).

Again in the interests of "structural feel", the students are not required to understand the *Mathematica* details of how the animation is generated, nor the mathematical details of how the strut forces are calculated (which involves the solution of systems of linear equations). What they do have control over are the magnitude of the test load and the "colour function" which maps a numerical force value onto a range of output colours. They are asked to consider how to design a colour function which yields the most useful information about what is going on in the bridge as the test load moves across it, and to design a function which would allow them to detect the maximum safe load that can cross a bridge given a maximum safe force for any strut.

It should be clear what are the engineering lessons from this activity: the students can get experience in how the patterns of forces vary in a loaded structure, and they are invited to consider, albeit for a toy example, a central engineering design question of determining what loads a given structure can safely support. The mathematical lessons may not be so obvious. Indeed, one might ask, where is the mathematics at all? Haven't we hidden all the relevant mathematics inside the *Mathematica* functions? In the most obvious sense, we *have* hidden the mathematics of the problem—the solving of systems of linear equations. But in fact, the mathematics can be made visible in two ways. First, the didactic inversion allows us to hide the *details* of the mathematical processes whilst keeping visible the very useful *results* of that process. Second, the students are invited to engage in an interplay between bottom-level *Mathematica* programming—defining colour functions—and high-level visualisation. For example, the "maximum safe load" question demands some kind of piecewise-defined function along the lines of (taking 35000 N as the max. safe force in a strut):

```
overloadFun[force_]:=If[Abs[force]>35000,
  GrayLevel[1], (*safe load exceeded – output white*)
  Hue[0.8*Abs[force]/35000] (*else output a colour spectrum*)
]
```

This process of making and criticising representations (i.e. the colour function mappings) is not conventionally recognised as mathematical work, at least not for non-expert, beginning students. But, we think that it reflects a kind of mathematical thinking that has a great deal to do with having a good structural feel. (And we think that it has a strong relationship with diSessa et al's [6] idea of "meta-representational competence").

These structures activities highlight a problem of visibility in design. They illustrate the complexity of the questions we asked at the outset: for now it should be clear that the question is not only whether or not to make the mathematics visible, but *what* mathematics, and in what form, to make visible? Designers must choose to make

certain pieces of the mathematics visible and functional, and it is functionality which is the very real contribution of the technology. At the same time, students have to map the expression of the mathematics (in whatever form) into the results they see—and to try to re-represent those results in terms of the (*Mathematica*-based) mathematics. This is not simply a matter of multiple representation, it is a matter of construction.

Discussion: Designing for visibility in a mathematical software environment

In this section, we want to consider the visibility of mathematical calculations in three different software packages—*Mathcad*, *Mathematica* and *Maple*—as well as how, and to what degree, different teachers of engineering mathematics are choosing to make mathematics visible whilst using those packages with students. Insight into the latter was gained from the proceedings of a recent workshop on the use of mathematical software packages in undergraduate engineering education[4]. This happened to allow us some rather intriguing views on the relationship between epistemology and visibility—in other words, how the intentions of the designer are translated into the conceptual mathematical models developed by the user/learner. As we shall see, the relationship is not straightforward.

Clearly, all the various software manufacturers are interested in appealing to as large an audience of mathematics users as possible, and their "box top" slogans express this: *Mathcad*—"the worldwide standard for technical calculations"; *Mathematica*—"the world's only fully integrated technical computing system"; *Maple*—"complete mathematics and visualisation system". But if one looks inside the box, we think that the different epistemologies of engineers and (applied) mathematicians can be made out in the software designs.

Mathematica and *Maple* are examples of "computer algebra systems", and represent what an applied mathematician might expect of "computer mathematics": comprehensive sets of symbolic, numerical and graphical functions, expressed in a precise, extensible mathematics-like programming language. (It should be said that neither are much up to doing formal proofs automatically, but then again they don't claim to be replacing human mathematical thought—fortunately!—but to provide an environment to support it.)

Mathcad is a package very much designed for, and commercially targeted at, engineers. It works rather like a "sketchpad" combination of word processor, spreadsheet and mathematical (mostly numerical) toolbox: inputs and results can be placed quite freely on the screen/page, but they are causally connected behind the scenes. It is interesting to trace the evolution of *Mathcad* over its past three or four versions**.** As usual, more functions, menus and palettes have appeared, but a couple of developments seem more indicative of a particular design philosophy: the first is the way that *Mathcad*'s developers are acquiring the electronic rights to many of the standard engineering data books, and making them available as $200 "electronic library" add-ons to the basic system. *Mathcad* is data-oriented, and proud of it.

[4] Organised by METRIC at Imperial College, June 1998. Proceedings are available from `http://metric.ma.ic.ac.uk/symposium/` .

Second, there is a "symbolic toolbox", that performs a selection of symbolic algorithms, which has grown in mathematical coverage with each new version (in fact, it is a portion of the *Maple* "mathematics engine" running in the background). We presume that this growth depends not least on the fact that enough users have requested a particular symbolic function to be added. Also, presumably, the developers of *Mathcad* have to pay the developers of *Maple* more to use more symbolic functions, which implies a certain conservatism on the part of the former.

Now, from the point of view of a typical *Mathcad* user, this growth process must seem quite natural. An engineer is faced with a problem to solve, and needs to apply mathematical techniques to solve it; chances are it won't yield to a symbolic technique anyway (few mathematical equations of practical use do possess exact analytic solutions outside of special cases), but having an improved package of symbolic techniques to hand is going to turn out to be useful some of the time.

However, from the viewpoint of a mathematician, this haphazard growth process could seem pretty worrying. Mathematicians make strong distinctions between symbolic and numerical procedures, so much so that the latter are often treated as a separate field of the discipline (i.e. numerical analysis). This is natural, too; surely it's the business of mathematicians to make such distinctions?

The message that we take from this comparison of perspectives is that visibility is not a simple issue: it depends on what designers, and users, think "mathematics" is. While it is no surprise that engineers and mathematicians see the function of mathematics differently, it is perhaps more surprising that they may not be thinking about the *same* mathematics: if that is true, it raises some difficult questions about the nature of applied mathematics itself, and surely indicates that the metaphor of *application* is, at best, limited.

Should it be the business of engineers to make the same distinctions as mathematicians—to work with the same mathematical epistemology? In particular, do engineering students get the most appropriate mathematical training by following traditional mathematics courses which give pride of place to symbolic techniques, and relegate numerical methods to second place?

For example, a speaker at the workshop, a teacher of civil engineering students, declared that a particular bugbear of his is having to re-orientate students who have been taught in school that integration is firstly about backward differentiation (a symbol-oriented view), and secondly (if at all) about determining areas under curves.

In principle (institutional finances and academic politics notwithstanding), a lecturer in engineering mathematics can choose between offering students a package like *Mathcad*, or a package like *Mathematica* (or *Maple*). *Mathcad* has the advantage of being a tool tuned for engineers, whilst *Mathematica* may be less so (it is certainly difficult to get to grips with it if you're not willing to think in explicit mathematical terms).

Either way, the educator has to come to terms with the design choices that the software developers have made: which mathematical aspects are visible enough, which need to be made more visible, or indeed less visible (as in our rainbow bridge example above).

It was interesting to note the high degree of unanimity at the workshop—amongst users of *Mathcad*, *Mathematica*, *Maple*, *Derive* and other systems—that the most important issue is how to develop competence in the use of appropriate mathematical

technology, which all agreed was grounded in a combination of traditional and technologically-based mathematical understanding. One of the *Mathcad* users put it like this:

> I let [the students] work in the way that an engineer would: they tend to work backwards to the mathematical questions from *Mathcad*'s answers: what do these things mean? If the answer isn't what you expected, did you expect a positive number, or a real number? Can you find by hand a bracketing answer? Then you've got confidence in the computer result.

And this from a *Maple* user:

> What we need to ensure is that our students become failsafe users of these powerful mathematical tools. The problem is very similar to (but more complex than) that which we have with the uncritical use of calculators to compute unlikely numerical answers. … With computer algebra systems the problem of validation is much greater and the techniques of validation are more varied and more sophisticated. …. What rules or heuristics do you teach to your students to ensure that they become discriminating users of computer algebra systems and failsafe engineers?

However, as we already pointed out in the introduction, there is certainly not unanimity in the engineering community as a whole about the way to do, and to learn, mathematical modelling:

> We encourage students not to be linear toward the answer, and to go back around the loop to understand things better, but when the students graduate into industry they're usually told that "academic approaches" can't be tolerated, "here in industry we're results-oriented", it's a case of getting the answer and then moving onto the next problem. The point is to know when you've done enough to be sure of your answer.

Summing up, it is clear that the epistemological decisions built into software design far from determine the user's activities. Epistemological structures shape and are shaped by what the user does, but these are not straightforwardly linked.

Conclusions

We have presented our examples of mathematics teaching and educators' discussion with the aim of challenging the traditional view of mathematics: that it is either studied in its own right *or* must inevitably be viewed as a succession of recipes, preferably wrapped in computational dressing. The former view may be attractive to mathematicians, but it has consistently failed engineering and science students. On the other hand, it seems increasingly likely that the latter view will render invisible crucial parts of the scientific and technological endeavour, in ways which relegate mathematics only to the privileged few who design the programs. This is, we think, an increasingly problematic issue, and one which is facing all those whose work involves—implicitly or explicitly—mathematical knowledge and techniques.

Michael Clayton [7], a mathematician working in the multidisciplinary environment of the telecommunications industry, has pointed to the "bridging" effects of technology on the relationships between mathematicians and engineers in industrial practice, overturning the traditional roles of mathematicians as makers of models, and other people in the design and production process as consumers of models:

> General-purpose IT tools such as spreadsheets, and mathematically based environments and workbenches such as *Mathcad* and *Matlab* have made it easier for engineers, dealers, salesmen, managers and others to construct their own models and refine them for specific applications. When time is of the essence, the value of these tools lies in the rapid prototyping they allow: initial modelling ideas can be investigated by the potential users, and the resulting interaction often leads to an improved match between the model and the users' requirements. Modern graphical user interfaces … can be designed to make even the most sophisticated special-purpose models accessible to the people who need to use them, helping to remove the "ivory tower" and "back room" images that have sometimes been attached to mathematicians in the past. [7, p. 25]

Clayton's insight may be crucial for effective university mathematics teaching in the future. The perennial debate over whether mathematics should be taught by mathematicians *or* by engineers looks increasingly anachronistic in the light of technological change, and modern industrial working practice. We think it is more instructive to examine the potential of mathematical technology to change the relationships between mathematicians and engineers, and to connect both people, and the knowledge domains in which they work, in new ways.

The tools we use, as much as the activities we design, shape the kinds of understandings our students construct. Moreover, the mathematical models for phenomena—however straightforward they are to mathematicians—are not straightforward at all: they depend critically on the settings in which they are used, and the tools with which they are expressed. Provided we are explicit (at least to each other, and perhaps to our students) we see this as a mathematical opportunity: in contrast, leaving this issue (and the mathematics) invisible must, we think, be a source of difficulty.

References

[1] Wolfram, S. , 1996, *The Mathematica Book*, third edition (Cambridge University Press).

[2] Sutherland, R. and Pozzi, S., 1995, The Changing Mathematical Background of Undergraduate Engineers: A review of the issues, The Engineering Council (London).

[3] LMS, IMA and RSS, 1995, Tackling the Mathematics Problem, a joint report by the London Mathematical Society, Institute for Mathematics and its Applications, and the Royal Statistical Society.

[4] Cox, W., Norris, W. T. and Penny, J. E. T., 1995, Mathematics in engineering teaching at university—a practitioner's dilemma, in *Mathematical Education of Engineers*, edited by L. R. Mustoe and S. Hibberd (Oxford University Press), pp. 251-266.

[5] Wilkinson, T. S., 1995, Mathematics for engineers: an industrial view, in *Mathematical Education of Engineers*, edited by L. R. Mustoe and S. Hibberd (Oxford University Press), pp. 3-14.

[6] diSessa, A., Hammer, D., Sherin, B., & Kolpakowski, T., 1991, *Journal of Mathematical Behavior*, **10**(2), 117-160.

[7] Clayton, M., 1998, Industrial applied mathematics is changing as technology advances: What skills does mathematics education need to provide?, in *Rethinking the Mathematics Curriculum*, edited by C. Hoyles, C. Morgan and G. Woodhouse (Falmer Press), pp. 22-28.