Ideas for a radical reform of mathematics education, après Arnold, Gelfand, and Vavilov

Alexandre Borovik

Introduction

I continue the discussion started by Nikolai Vavilov in his remarkable paper of 2021 (with Vladimir Khalin and Alexander Yurkov) *The skies are falling: Mathematics for non-mathematicians* [10] supported by their texbook [9]. It is about the crisis in mathematics education and the urgent need for a deep reform. I will invoke views on possible changes from other mathematicians, first of all, Vladimir Arnold and Israel Gelfand.

But my assessment is more stark: the mainstream mathematics education *for everyone* is dying. Its fate appears to be sealed by the emergence of AI with its promise that a smartphone will answer all mathematical questions which a person would ever encounter in his/her everyday life. This is happening for reasons beyond our control: the change of the role of mathematics in the very heart of the economy—in division of labour [3, 4]. I do not know how to save or reform the mainstream mathematics education. Instead, I focus on development of a much more narrow stream of academically selective education, *mathematics for makers* [4] build on the principles of *Deep Education*, starting in primary school [3, Sections 8–10]:

> Mathematics education in which every stage, starting from pre-school, is designed to fit the individual cognitive profile of the child and to serve as propaedeutics of his/her even deeper study of mathematics at later stages of education—including transition to higher level of abstraction and changes of conceptual frameworks.

This means running a network of mathematical circles, olympiads, Sunday, Winter, Easter, Summer Schools, etc. for identifying children who must be saved from their schools and directed to mathematical classes, schools, boarding schools of the *Deep Stream*. This is a colossal task, and it cannot be done without an active participation of the professional mathematical community. We have to realise that we have to ensure survival of our community and our culture. It is worth to remember a message from the times when a certain community of faith was fighting for its survival:

> *Then saith he unto his disciples, The harvest truly is plentious, but the labourers are few.* (Matthew 9:37 KJV)

1. Deep Stream: Arithmetic + Algorithmic + Algebra

With the administrative structure of the Deep Stream so loose, some shared understanding of its purpose and possible curricula is paramount. I emphasize: this is not about adapting the 20th century mathematics to the 21st century. As outlined in [6, 10], it means the 21st century mathematics education for the 21st century mathematics.

1.1. Arithmetic and Algorithmic

What I suggest in my talk is a fragment of one of possible curricula based on merging mathematics with computer science / programming, a course *Arithmetic, Algorithmic, Algebra* (AAA) which should start in primary school as soon as kids can read and write/type. Its first principle:

> From a relatively early stage of the course, Learner's answer to an arithmetic or algebraic problem (including mathematical problems described as 'real life' problems) should be an algorithm (and later—executable computer code developed, almost in its entirety, by Learner—without use of standard packages such as MATHEMATICA)—which
> - solves *all* problems of the same type;
> - helps to check, analyse, and generalise the solution.

At the earliest stage, this is achievable by applying the *questions procedure* to 'word problems' [5, Sections 3 and 4]: a sequence of questions produced in the solution is already an algorithm written in plain language. Together with attention to *named numbers* (future *typed variables* of coding) [5, Sections 3 and 4] this already meets the requirements.

I stick to two principles formulated in 1947 by Igor Arnold[1] [5, Section 1]:

> **1.** *Teaching arithmetic involves, as a key component, the development of an ability to negotiate situations whose concrete natures represent very different relations between magnitudes and quantities.*'

In modern mathematical language this means mapping the structures and relations of the real world into operations and relations of arithmetic.

> **2.** *The difference between the "arithmetic" approach to solving problems and the algebraic one is, primarily the need to make a concrete and*

[1]Igor Arnold was the father of Vladimir Arnold, who republished his paper and endorsed it in a touching foreword [1].

sensible interpretation of all the values which are used and/or which appear at any stage of the discourse.

Again, it means full use of "functorial" properties of the mapping from the real world to numbers and other mathematical structures, and development of depth and functionality of thinking within a limited language. My favourite simile is the *Essential English* method developed in 1930s by West, Palmer, and Faucett for teaching English as a second language, which was aimed at achieving *full language fluency* within a *limited*, but *functional vocabulary.*

1.2. Algebra

Algebra should be introduced simultaneously with switch from verbal algorithms to computer programming.

The content of the algebra course should change and reflect the demands of computer programming, and include, for example, Boolean algebra, elements of number theory, modular arithmetic, and finite fields. Shaping the algebraic curriculum should go in step with development of the bespoke, and very child-friendly, Domain-Specific Language (DSL)[2] for use in the course. Most likely, existing general purpose languages are not suitable for this role. In particular, DSL is needed for bridging the conceptual gap between

- formulation of an algorithm and its implementation in a code, and
- representation of the answer as a closed algebraic formula.

1.3. Elementary Geometry

AAA includes elementary geometry as a theory of the 2-dimensional vector space over $\mathbb{R}$ with two non-degenerate bilinear forms, one symmetric—*dot product*, another skew-symmetric—*wedge product*. This approach was developed by Israel Gelfand in his proto-textbook of elementary geometry for schools, left, unfortunately, unfinished. The use of the wedge product allows us to easily handle, at the level of algorithms and computer code, orientation of the plane and concepts of left/right and clockwise/counterclockwise. Moreover, this solves problems unapproachable in the traditional school geometry, for example, this one:

> **Problem.** A scalene triangle (that is, a triangle in which all three sides are in different lengths) is drawn on a piece of paper. It was carefully cut out and turned over. Prove that if you move this triangle along a sheet of paper, then it cannot be used to cover the hole left behind without any gaps.

You can experiment with a cut out and flipped over scalene triangle on a Möbius strip; if you drag it near the hole, you can't cover the hole. But if you drag it along the entire length of the strip, it covers it perfectly. Locally, a Möbius strip is a piece of the Euclidean plane; but, unlike the latter, it is not orientable [8].

[2] A Domain-Specific Language (DSL) is a computer language specialized to a particular application domain. This is in contrast to a general-purpose language (GPL), which is broadly applicable across domains. Famous examples include html and LaTeX. Some books: [2, 11].

Orientability of a surface is its *global* property, but school geometry was always *local*, made on a small sheet of paper.

1.4. The rest of school mathematics

Other parts of mathematics, for example, mechanics, combinatorics, probability theory, and statistics need a separate discussion and are not touched here. Also, interactions with physics deserve most serious attention.

Conclusion

This abstract is only a very brief outline of a relatively short about talk proposed course. Some additional details could be found in [7].

References

[1] I. V. Arnold, *Principles of selection and composition of arithmetic problems.* MCNMO, Moscow, 2008. (In Russian).

[2] M. Boersma, *Domain-specific languages made easy.* Manning Publication, 2020.

[3] A. V. Borovik, *Calling a spade a spade: Mathematics in the new pattern of division of labour*. In B. Larvor (Ed.), *Mathematical cultures: The London meetings 2012–2014*; Springer International Publishing, Cham, pp. 347–374, 2016.

[4] A. V. Borovik, *Mathematics for Makers and Mathematics for Users.* In B. Sriraman (Ed.), *Humanizing mathematics and its philosophy: Essays celebrating the 90th birthday of Reuben Hersh.* Springer International Publishing, Cham, pp. 309–327, 2017.

[5] A. V. Borovik, *Economy of thought: a neglected principle of mathematics education.* In R. Kossak and Ph. Ording (Eds.), *Simplicity: Ideals of practice in mathematics and the arts*; Springer, pp. 241–265, 2017.

[6] A. Borovik, Z. Kocsis, and V. Kondratiev, *Mathematics and mathematics education in the 21st century*, 2022. arXiv:2201.08364 [math.HO].

[7] A. Borovik and V. Kondratiev, *A new course 'Algebra + Computer Science': What should be its outcomes and where it should start*. 2022. arXiv:2212.12257v1 [math.HO]

[8] V. V Nikulin and I. R. Shafarevich, *Geometries and groups.* Moscow, Nauka, 1983. (In Russian.)

[9] N. A. Vavilov, V. G. Khalin, and A. V. Yurkov, *Mathematics for non-mathematicians.* Moscow, MCMNO, 2021. (In Russian.)

[10] N. A. Vavilov, V. G. Khalin, and A. V. Yurkov, *The skies are falling: Mathematics for non-mathematicians.* Doklady Mathematics **107**, Suppl. 1, pp. S138–S151, 2023.

[11] M. Voelter et al, *DSL engineering. Designing, implementing and using domain-specific languages.* 2010. `http://dslbook.org`.

Alexandre Borovik
University of Manchester, Manchester, UK
e-mail: `alexandre@borovik.net`