Why mathematics needs engineering

Raymond Boute, INTEC, Ghent University   `raymond.boute@pandora.be`

Abstract

Engineering needs mathematics, but the converse is also increasingly evident. Indeed, mathematics is still recovering from the drawbacks of several "reforms". Encouraging is the revived interest in proofs indicated by various recent *introduction to proof*-type textbooks. Yet, many of these texts defeat their own purpose by self-conflicting definitions. Most affected are fundamental concepts such as relations and functions, despite flawless accounts 50 years ago. We take the viewpoint that definitions and theorems are tools for capturing, analyzing and understanding mathematical concepts and hence, like any tools, require diligent engineering. This is illustrated for relations and functions, their algebraic properties and their relation to category theory, with the *Halmos principle* for definitions and the *Arnold principle* for axiomatization as design guidelines.

Keywords: algebra, analysis, calculus, category theory, codomain, definition, design, domain, engineering, function, logic, mathematics, relation, soundness

1. Introduction: Mathematics and Engineering

Mathematics has been intertwined with engineering since antiquity [12, 52].

Kline notes that "*More than anything else mathematics is a method*" [32]. Arguably, the primary purpose of this method is *effective reasoning*. This view best explains what Wigner calls *the unreasonable effectiveness of mathematics* [64], in particular its practical usefulness far beyond the originally intended application areas. From this perspective, the dichotomy between Platonism and formalism dissolves: mathematical objects *do* exist, albeit in an abstract universe. Formalism, definitions and theorems are the tools to study them.

Tools, being artifacts, deserve careful design, borrowing criteria and guidelines from engineering. Some of these also been discussed by José Oliveira [43] in another context. Here we focus on using engineering principles in mathematics.

Foremost is enhancing the effectiveness in reasoning. Symbolic notation properly designed and used yields extra guidance via the shape of the expressions. It should function like well-meshed gears in a Swiss precision clockwork.

Aptness and economy in capturing the abstract objects of interest ensures conceptual malleability, generality and practical usefulness. Human factors are influential here, and it is often overlooked that this is a highly individual matter of temperament and background. Even so, everyone benefits from clear con-

ceptualization and reasoning. For instance, *separation of concerns* avoids the common misconceptions caused by intellectual noise and conceptual tangling.

In classical mathematics, methods and notations were often thought-out carefully. In algebra, for instance, symbolic notation started with Diophantus and evolved into its current form via Viète and Descartes [6, 12], rarely violating good design practices, thus making symbolic calculation today's norm. In comparison, notations from "modern mathematics" as used in everyday practice are substandard, hampering symbolic reasoning and thus making it unpopular.

The cause of this stagnation is largely historical. When introducing so-called "modern mathematics", forgetting its roots caused serious educational mistakes, denounced in rather strong terms by Arnold [3]. In a severe overreaction, the view of mathematics as a method was sacrificed in favor of mathematics as a bag of tricks and attempting to elicit motivation by so-called "real-life" examples no more realistic than the *farmer-sells-potatoes*-type problems in grade school — and in PISA tests! The well-proven structure *definition-examples-theorems* was frowned upon, and mathematical exposition had to become a "narrative".

As a result, classics like Rudin's *Principles of Mathematical Analysis* [51] are, as Krantz observes, "*often no longer suitable, or appear to be inaccessible, to the present crop of students*" [34]. Here the blame does not fall on the students.

Narratives lack the punctuation provided by headings like "Definition" and "Theorem", which help novices to distinguish between, say, statements that can be deduced from earlier ones and statements introducing new elements.

If *definition-examples-theorems* expositions often deserve criticism, it is not for the usual reasons (take your pick), but because definitions are usually presented as "given", or as arbitrary points of departure for a game of logic. In fact, definitions are the result of *design decisions*. They also determine the flavor of the theorems (and proofs) derived from them. Hence it is crucial for understanding that these decisions are explained and justified.

In mathematics texts, this is all too rarely done. One of the few exceptions is Halmos's *Naive Set Theory* [26] which, if only for this reason (yet also for other reasons!), should be required reading for all beginning students — and many mathematicians as well. Halmos not only explains the design decisions and their shortcomings for most conventions, but also does not shrink back from calling some poor practices "unacceptable but generally accepted". Quine [47] even designates lesser offenses as "glaring perversity", which seems an apt characterization of mathematicians acting against better judgment.

Indeed, perceived "general acceptance" is often taken as a licence to perpetuate junk conventions. Users of inept designs typically defend them by feigning confusion at proper alternatives, calling them "nonstandard" even if they have been around for a long time and are routinely used by plenty of other authors.

If an engineer is sloppy, his design may fail, even catastrophically. Mathematicians often condone sloppiness, even if it sets bad examples and abuses confidence. Discerning students will be dissatisfied by the discrepancy between the reputation of mathematics as being precise and actual practice. Others may even get confused if insecure teachers insist on "doing things as in the book".

Yet, the engineering literature is not blameless either. Years ago Lee and

Varaiya [39] corrected many inept mathematical practices in signal processing.

Playing down such issues as "just a matter of notation" is misleading. Poor notation prevents the shape of expressions from giving guidance in reasoning. It also reflects poor understanding, according to Boileau's aphorism "*Ce que l'on conçoit bien s'énonce clairement – Et les mots pour le dire arrivent aisément*". If authors misunderstand their own definitions, what about their students?

"If it ain't broke, don't fix it" is another engineering maxim. Yet, as we shall see, even basic concepts that worked fine 50 years ago somehow got "broke".

This paper addresses the issue in the title by presenting a design view on various concepts from the literature, but it is *not* some linear, complete proposal.

Often references include page numbers to make them truly useful for the reader. For brevity, co-authors are omitted when mentioning names in the text.

2. Case study A – Relations: two logically equivalent definitions

2.1. Simple and safe formulations

The simplest "modern" definition of a *relation* is typical in older texts such as Bourbaki [9, p. 71], Suppes [60, p. 57], Tarski [61, p. 3], but only in a few current books, such as Jech [31, p. 10], Scheinerman [53, p. 73] and Zakon [65, p. 8].

Definition 1 (Relation). *A relation is a set of ordered pairs.*
Equivalently, in symbols [9, 60]: R isrel $\equiv \forall z \,.\, z \in R \Rightarrow \exists x \,.\, \exists y \,.\, z = (x, y)$.

Taking *set* and *ordered pair* colloquially, and with 'nonmathematical' examples, the word statement of Definition 1 is even accessible at grade school level.

In this paper, when saying just "pair", we always mean "ordered pair".

Some notational design issues arise here. First, an ordered pair is commonly written (x, y). Some authors use $\langle x, y \rangle$, a waste of symbols. In fact, one can even write x, y and reserve parentheses for emphasis or disambiguation, which also covers *n*-tuples like (x, y, z) and trees like $((x, y), z)$. Identifying (x, y, z) with $((x, y), z)$ as in Bourbaki [9, p. 70] is clearly a bad design decision.

Second, the literature diverges about writing (x, y) or (y, x) and $x \, R \, y$ or $y \, R \, x$. Quine [47, p. 24] offers many good reasons for following Peano and Gödel in using the *natural order* from spoken language, writing "*a* is the father of *b*" as $a \, F \, b$, and "*a* is smaller than *b*" as $a < b$. Similar reasons would favor writing, for instance, "velocity versus time" as (v, t). However, mathematicians used to writing the "independent variable" first might feel disoriented—unlike novices! Tradition can be reconciled with reason by writing $(x, y) \in R$ iff $y \, R \, x$. For human engineering reasons, such conventions should be stated conspicuously.

Intermezzo: Quine, the angry notational engineer. In an uncharacteristic diatribe of nearly two pages [47, p. 24–26], Quine deplores the "sorry business" and "glaring perversity" of ill-designed notations. He concludes (a) "*I have given much space to a logically trivial point of convention because in practice it is so vexatious.*"; (b) "*[Whoever] switched a seemingly minor point of usage out of willfulness or carelessness cannot have suspected what a burden he created.*".

These remarks reflect typical engineering concerns. Indeed, (a) reminds us that avoiding flaws during the design phase is easy compared with repair afterwards (one line versus lengthy arguments), and (b) emphasizes the importance of taking into account the interests of the users, namely, the future generations.

Auxiliary notions. Most authors using Definition 1 add Definitions 2, 3 and 4.

Definition 2 (Domain, range).
The domain *of a relation R is the set of first members of the pairs in R. The* range *of a relation R is the set of second members of the pairs in R.*
Notation: the literature mentions various self-explanatory notations such as $\mathcal{D} R$ or $\mathrm{Dom}(R)$ for the domain of R and $\mathcal{R} R$ or $\mathrm{Ran}(R)$ for the range of R.

Definition 3 (Relation from X to Y). *A* relation from X to Y *is a relation whose domain is a subset of X and whose range is a subset of Y.*
Equivalently, in symbols: $R \, \mathrm{isrel}\, (X, Y) \equiv R \, \mathrm{isrel} \wedge \mathcal{D} R \subseteq X \wedge \mathcal{R} R \subseteq Y$.
Notation: Writing $R : X \leftrightarrow\!\!\!\rightarrow Y$ or $R : X \leftrightarrow Y$ introduces a relation from X to Y.

Aside As in Meyer [42, p. 23], $X \leftrightarrow\!\!\!\rightarrow Y$ denotes the set of relations from X to Y. Also, the symbol : clearly distinguishes bindings from statements. A *binding* $i : S$, read "i <u>in</u> S", *introduces* an identifier i for an object in a set S, whereas $i \in S$, read "i <u>is in</u> S" (or similar) is a *statement* in which i is *used*. Proper symbolism passes the *prose test*: transliterating formulas into words must yield sentences with correct grammar. The RHS of $S \subseteq T \equiv \forall \, x : S \,.\, x \in T$ is read "for all x <u>in</u> S, x <u>is in</u> T" or "every x <u>in</u> S <u>is in</u> T". Lamport [36, p. 289] notes that the common forms $\{x \in S \,|\, p\}$ and $\{e \,|\, x \in T\}$, where p is a boolean expression and e is any expression, are ambiguous if p is $x \in T$ and e is $x \in S$. Writing $\{x : S \,|\, p\}$ and $\{e \,|\, x : T\}$ yields $\{x : S \,|\, x \in T\} = S \cap T$ and $\{x \in S \,|\, x : T\} \subseteq \mathbb{B}$.

Finally, S in $i : S$ is called a *type* and expresses a range for i, not an attribute of i. Hence the X and Y figuring in $R : X \leftrightarrow\!\!\!\rightarrow Y$ are attributes of the type $X \leftrightarrow\!\!\!\rightarrow Y$, not of R. This concludes the aside.

The following formulation is independent of the (x, y) versus (y, x) issue.

Definition 4 (Composition, converse). *The* composition $S \circ R$ *of relations S and R is the relation such that $z(S \circ R)x \equiv \exists y . z \, S \, y \wedge y \, R \, x$.*
The converse $R^{\smile}$ *of a relation R is the relation defined by $x \, R^{\smile} \, y \equiv y \, R \, x$.*

Composition is called *relative product* by Suppes [60, p. 63] and Tarski [61, p. 3], and *resultant* by Quine [47, p. 22]. Suppes writes S/R, the others $S|R$.

2.2. Entering murky waters: misconceptions, unsoundness and poor judgement

All books in our *introduction to proof* sample [8, p. 172], [14, p. 101] , [15, p. 93], [19, p. 155], [20, p. 86], [23, p. 51], [24, p. 267], [27, p. 192], [49, p. 176], [55, p. 135], [62, p. 171] except [53, p. 73] combine Definitions 1 and 3 as follows.

Definition 5 (Relation from X to Y). *A* relation from X to Y *is a subset of $X \times Y$.* Equivalently, in symbols: $R \, \mathrm{isrel}' \, (X, Y) \equiv R \subseteq X \times Y$.

This forces defining *relation* backwards, as a relation from X to Y for some X, Y. Also, using Cartesian products at this stage may be an educational burden [54].

A more serious problem is that disregarding *separation of concerns* carries a heavy price in understanding, apparently even for the authors. Most fail to recognize that Definitions 3 and 5 are equivalent: R isrel $(X, Y) \equiv R$ isrel$'$ (X, Y).

Indeed, many textbooks strongly suggest that Definition 5 somehow "glues" X and Y to the relation, and that one cannot even define just *relation* without adding "*from X and Y*". Some texts remain vague here, but the litmus test for one's understanding of a mathematical concept is the view on *equality*, in this case: when is a relation R from X to Y equal to a relation S from U to V?

The answer was evident 50 years ago: with relations defined as sets, $R = S$ iff both contain the same elements, regardless of X, Y, U, V. Defining equality anew often causes unsoundness, typically by stating that $R = S$ also requires $X = U$ and $Y = V$ [49, p. 179]. Indeed, if $R \subseteq X \times Y$, $X \neq \emptyset$ and $Y \subset V$, then $R \subseteq X \times V$ and $Y \neq V$, so $R = R$ would require $Y = V$, a contradiction.

Many texts [49, p. 180], [55, p. 141], [62, p. 236], [63, p. 94] define $S \circ R$ for a relation R from X to Y and a relation S from U to V only for the case $U = Y$. Since $R \subseteq X \times Y \subseteq X \times (Y \cup U)$ and $S \subseteq U \times V \subseteq (Y \cup U) \times V$, this is not restrictive, unless one accepts the aforementioned unsound view on equality.

Aside As a restrictive variant of a *relation from X to Y*, Bourbaki [9, p. 72] defines a *correspondence from X to Y* as a triple (R, X, Y) where $R \subseteq X \times Y$, and restricts composition of (R, X, Y) and (S, U, V) to the case $U = Y$.

3. Case study B – Functions

Poor design decisions for relations reappear for functions with a vengeance. This is especially unfortunate since, as Herstein puts it, *Without exaggeration this* [namely, a *mapping* or *function*] *is probably the single most important and universal notion that runs through all of mathematics.* [29].

The "modern" definition of *function* was issue-free 50 years ago, and still is in analysis/calculus books [1, 5, 17, 33, 34, 38, 50, 51, 58, 59, 65], but unsoundness appears since 2005 in *transition to proof* texts [8, 13, 14, 15, 19, 20, 27, 49, 55].

In passing, we mention some harmful myths that can be read between the lines in textbooks but surface explicitly in oral and written conversations. Myth #0 (a meta-myth actually) holds that divergences in definitions just reflect different needs in various disciplines [54]. However, our samples come from algebra, analysis/calculus, discrete math, logic, set theory, and reveal that nearly *all* of them use the *same* function concept, differing only in the care devoted to design and formulation. Myth #0 is harmful in trying to divert closer scrutiny.

Recurrent points of interests are: (i) defining *function*, (ii) function equality, (iii) function from X to Y, (iv) onto-ness, (v) function composition and inverse.

3.1. Once again: simple and safe formulations
(i) A relation R is called *functional* [9, 42] iff no two pairs in R have the same first member. Hence the following phrasings are equivalent; the choice depends on whether *relations* are skipped, as in analysis/calculus texts, or defined first.

Definition 6 (Function). (A) Apostol [1, p. 53]: *A function f is a set of ordered pairs (x, y) no two of which have the same first member.*
(B) Bourbaki [9, p. 77]: *A function is a functional relation.*

Definition 6 is also found in Dasgupta [16, p. 10], Flett [17, p. 4], Jech [31, p. 11], Mendelson [41, p. 6], Scheinerman [53, p. 167], Suppes [60, p. 86], Tarski [61, p. 3], Zakon [65, p. 10]. Functionality justifies writing $y = f(x)$ iff $(x, y) \in f$. As in [40, p. 1], one may write $f\,x$ instead of $f(x)$ when no ambiguity results. Authors using Definition 6 introduce the domain and range as in Definition 2.
  (ii) This results in the following theorem, quoted from Apostol [1, p. 54].

Theorem 1 (Equality). *Functions f and g are equal iff* (a) *f and g have the same domain, and* (b) *$f(x) = g(x)$ for every x in the domain of f.*

  (iii) Authors starting from Definition 6, including Apostol [1, p. 578], Dasgupta [16, p. 10], Flett [17, p. 5], Jech [31, p. 11], Scheinerman [53, p. 169] and Zakon [65, p. 10], use the following common notions for classifying functions.

Definition 7 (Function from X to Y). *A function from X (in)to Y is a function with domain X and range included in Y.*
Notation: writing $f : X \to Y$ introduces a function f from X to Y.

This is the ISO standard [30, p. 15], where a *function* is defined in broader terms, mentioned later. Divergent views on $f : X \to Y$ are *nonstandard*.
  It is convenient reading $X \to Y$ as the set of functions from X to Y and $X \nrightarrow Y$ as the set of functional relations from X to Y [42, pp. 25–26]. Such types serve as *partial specifications* for functions. Tighter types are defined later.
  (iv) Next, we consider *onto-ness* as defined by authors using Definitions 6 and 7, for instance, Flett [17, p. 5], Jech [31, p. 11], Mendelson [41, p. 6], Scheinerman [53, p. 172], Tarski [61, p. 3] and Zakon [65, p. 11].

Definition 8 (Onto Y). *For any set Y, a function is said to be* onto Y, *or surjective on Y, iff its range is Y.*

Note that *onto* is a preposition, and appears as such in Definition 8, which is also used by many authors (listed later) who do *not* start from Definition 6.
  The dual notion of "f is *onto* Y" ($\mathcal{R}f = Y$) is "f is *total on* X ($\mathcal{D}f = X$). The dual notion of "f is *into* Y" ($\mathcal{R}f \subseteq Y$) is "f is *partial on* X" ($\mathcal{D}f \subseteq X$). A function $f : X \nrightarrow Y$ is often called a *partial function*, but is $\{(0,1), (2,3)\}$ a partial function? Grammatically correct is: a *function from part of X to Y*.
  (v) For composition, we mention two equivalent formulations. Formulation (A) skips relations, as in Apostol [1, p. 140], Flett [17, p. 11], Mendelson [41, p. 7] and many others, listed later. Formulation (B) is based on relations.

Definition 9 (Function composition $g \circ f$). (for *any* functions g and f)
(A) *$g \circ f$ is the function whose domain consists of all x in $\mathcal{D}f$ that satisfy $f(x) \in \mathcal{D}g$ and whose value $(g \circ f)(x)$ for arbitrary x in that domain is $g(f(x))$.*
(B) $g \circ f$ follows Definition 4 assuming natural order as in Quine [47, p. 24]; otherwise f and g must be swapped, e.g., $g \circ f = f/g$ in Suppes [60, p. 87], $g \circ f = f|g$ in Tarski [61, p. 3]. Proof obligation: showing that $g \circ f$ is functional.

3.2. Wearing the ice thin: convoluted formulations

(i–iii) Definitions 6 and 7 are sometimes crammed together, starting as early as Halmos [26, p. 30] and Herstein [29, p. 10], and more often in current texts including Krantz [34, p. 20], Velleman [62, p. 226] and others, listed later.

Definition 10 (Function from X to Y). *Let X and Y be sets.*

(a) *A function f from X to Y, written $f : X \to Y$, is a relation $f \subseteq X \times Y$ satisfying the property that for each x in X the relation f contains exactly one ordered pair of the form (x, y).*

(b) *The set X is called the* domain of f.

As a warning against uncritical copying, we reproduced the widespread but unacceptable phrasing, which suggests that the function is written $f : X \to Y$ (in fact, the function is written just f), and that $f \subseteq X \times Y$ is a relation (in fact, it is a statement about the relation f). Proper phrasings are evident.

All authors using Definition 10 overlook that part (b) requires proving that X is fully determined by f as defined in (a). This is easy; the result is $X = \mathcal{D}f$.

More importantly, users of Definition 10 fail to realize its logical equivalence to Definition 6 (proof: exercise), including the standard meaning of $f : X \to Y$. Still, the different formulation has a huge impact on clarity. By disregarding separation of concerns, Definition 10 has given rise to Myth #1, which holds that one cannot define *function* by itself, but only *function from X to Y*.

The common pitfalls are exposed by the litmus test: equality. Surprisingly few present-day texts using Definition 10 mention Theorem 1, found only in Daepp [14, p. 152], Gerstein [20, p. 113] and Smith et al. [55, p. 189]. Instead, many define equality anew, all too often unsoundly, as demonstrated later.

(iv) "Classical" authors using Definition 10 (or similar), including Bartle [5, p. 13], Halmos [26, p. 31], Herstein [29, p. 12], Kolmogorov [33, p. 5], say that f is *onto Y* iff $\mathcal{R}f = Y$, as in Definition 8, using "onto" as a *preposition*. Some of the few "modern" users of Definition 10 who write "onto Y" are Gerstein [20, p. 118] and Smith [55, pp. xvii, 205], but their formulation lacks generality.

(v) Most "classical" authors using Definition 10 (or similar), including Bartle [26, p. 40] and Halmos [26, p. 40], define $g \circ f$ for arbitrary functions f and g, as in Definition 9. Only a few classical texts [29, p. 13][50, p. 9] restrict coverage of $g \circ f$ for functions $f : X \to Y$ and $g : U \to V$ to the special case $U = Y$.

Most "modern" texts succumb to this restriction, including Bloch [8, p. 146], Roberts [49, p. 226], Scheinerman [53, p. 183], Smith [55, p. 197], Velleman [62, p. 231]. An intermediate form requiring $\mathcal{R}f \subseteq \mathcal{D}g$ appears in Daepp (2003) [14, p. 175], Daepp (2011) [15, p. 167], Jech [31, p. 11] and Krantz [34, p. 22]. The general form appears in Larson [38, p. 25] and Stewart [58, p. 40] which, not surprisingly, are calculus texts, since restricted composition is impractical.

3.3. Falling through the ice: common yet unsound additions to Definition 10

(i–iii) Definition 10 is the sound part of definitions in *transition to proof* texts, but Bloch [8, p. 131], Chartrand [13, p. 216], Daepp [14, p. 147][15, p. 143], Garnier [19, p. 224] Gersting [20, p. 383], Hammack [27, p. 195], Roberts [49, p. 220], Smith [55, p. 185], Gilbert [22, p.13] and Wallis [63, p. 106], add

Definition 11 (Codomain). Definition 10(c) Y *is called the* codomain of f.

However, just like Definition 10(b), adding 10(c) entails a proof obligation. Recognizing this clearly reveals a logical contradiction. Indeed, the definiendum is *codomain of f*, the definiens is Y, but Y is not uniquely determined by f. As for relations, letting $f \subseteq X \times Y \subset X \times Y'$ reveals a contradiction.

Still, some authors uphold Myth #2: writing $f : X \to Y$ makes Y an attribute of f by specifying $f \subseteq X \times Y$. Yet, all this says about Y is $\mathcal{R}f \subseteq Y$.

Myth #3 maintains that Definition 10 contains ambiguities allowing multiple views, making logical contradictions just a matter of interpretation. Yet, in Definition 10(a), the definiendum and the definiens are clear (except as written in [8, p. 131]), using the unambiguous concepts *subset* and *Cartesian product*.

Also, insofar as Definition 11 makes the perceptive reader wonder if the authors really mean "codomain *of f*", further context indicates they mostly do.

Again equality is most revealing. Apart from three exceptions mentioned, all *transition to proof* texts using Definition 11 overlook Theorem 1 and define equality anew. Roberts [49, p. 223] avoids conflict with Definition 10 by using the statement of Theorem 1. Others, e.g., Bloch [8, p. 136], Garnier [19, p. 224], Hammack [27, p. 198] extend this statement with $\operatorname{cod} f = \operatorname{cod} g$, indirectly contradicting Definition 10. Interestingly, Smith [55, p. 189] explicitly states that function equality does *not* require equal codomains!

(iv) Unsoundness also results from improper use of "*onto*" as an adjective, as in Bloch [8, p. 155], Daepp [14, p. 163][15, p. 157], Gries [24, 282], Hammack [27, p. 199], Krantz [34, p. 22], Roberts [49, p. 231] and Velleman [62, p. 236].

Definition 12 (Onto). *A function $f : X \to Y$ is* onto (surjective) *iff $\mathcal{R}f = Y$.*

With Definition 12, the same function can be both onto and not onto depending on whether or not the set Y appearing in $f : X \to Y$ happens to be $\mathcal{R}f$.

(v) All texts adding Definition 11 require for $g \circ f$ that $\mathcal{D}g = \mathcal{C}f$ and for the inverse f^- that $\mathcal{R}f = \mathcal{C}f$. This is impractical for applications, e.g., in calculus.

3.4. Design considerations, variant concepts, and evaluation

Many textbooks and countless blogs use the term *codomain*, typically in the unsound way described. Even though most authors using Definition 10/11 seem to sense the problems with squeezing in codomains, they somehow feel obliged to try. Of course, they needn't! Clearly, a proper account for *codomain* is overdue.

In view of the earlier analysis, the term *codomain* is best (i) simply discarded, or (ii) used as the symmetric counterpart of *domain*, thus far called *range*, or (iii) recognized as an attribute of a type like $X \to Y$ or $X \nrightarrow Y$, not of a function.

A quite different approach is defining a variant of the function concept such that the set Y in $f : X \to Y$ is truly part of f, safely called *the codomain of f*.

For instance, Bourbaki initially defines a function [9, p. 76] as a triple (F, A, B) where F is a functional relation with domain A and range included in B. Such a triple is subsequently called an *application from A into B* [9, p. 76], which avoids confusion with using *function* for a functional relation [9, p. 77]. The term *codomain* is not mentioned in [9] — so let's not blame Bourbaki!

The engineering question is: what purpose might such a variant serve?

For the sake of generality, this issue is best disassociated from the set of pairs view, whose predominance in the discussion thus far just reflects random literature samples from diverse areas of mathematics. Many authors, including Lang [37, p. 38], Lee [39, p. 48], Royden [50, p. 8] and Spivey [56, p. 29], note that a set of pairs is really a *representation* of a function, called its *graph*. Hence let's broaden the *representational* Definition 6 to a *conceptual* one, inspired by the Goursat/Courant style, but generalized to arbitrary domains and properly distinguishing f from $f(x)$. For instance, the ISO standard [30, p. 15] just says that a *function f* assigns to each x in its domain a unique value $f(x)$. Here "assigns" can be made precise by an *assertion* of the form $A(x, f(x))$, called *relation* by Bourbaki [9, p. 47], but not to be confused with a set of pairs.

Equality is pivotal for mathematical objects. Distinguishing functions on the basis of possible assignments outside their domain would be an useless complication, hence these values are best declared irrelevant for a function.

(A) The minimalist, "no frills" design reflecting this view is the following.

Definition 13 (The *function* concept: minimalist design).

i. *A function f is an object fully specified by* (a) *a set $\mathcal{D}f$, called* the domain of f, *and* (b) *for each x in $\mathcal{D}f$ a unique* value, *written $f(x)$ or $f\,x$.*

ii. *The stipulation* "fully specified" *means that $f = g$ if* (a) $\mathcal{D}f = \mathcal{D}g$ *and* (b) $f(x) = g(x)$ *for all x in $\mathcal{D}f$.* (Note: "only if" by Leibniz's principle [24])

iii. *A function f from X to Y is a function such that* (a) $\mathcal{D}f = X$ *and* (b) $f(x) \in Y$ *for all x in X. Such a function is introduced by writing $f : X \to Y$.*

Definition 13.iii simply follows the ISO standard: $\mathcal{D}f = X$ and $\mathcal{R}f \subseteq Y$, where $\mathcal{R}f = \{f(x) \,|\, x : \mathcal{D}f\}$. Types of the form $X \to Y$ are partial specifications. An illustration is defining sqrt : $\mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ with $(\text{sqrt}\,x)^2 = x$. *Composition* is unrestricted: $\mathcal{D}\,(g \circ f) = \{x : \mathcal{D}f \,|\, f(x) \in \mathcal{D}g\}$ and $(g \circ f)\,x = g(f\,x)$ as usual.

Composition also supports *specification by proxy*: specifying $g : Y \to Z$ via $f : X \twoheadrightarrow Y$ ($\twoheadrightarrow$ indicating onto) and $h : X \to Z$ by $g(f\,x) = h\,x$. *Well-definedness* (functionality) of g requires $h(x) = h(x')$ whenever $f(x) = f(x')$.

An example is defining the inverse: let $g := f^-$, $Y := \mathcal{R}f$, $Z := X$ and $h := \text{id}_X$. Well-definedness of f^- amounts to f being 1-1. In the general scheme, if f is 1-1, then g is well-defined and $g \circ f = h$ is equivalent to $g = h \circ f^-$. Pattern matching is an instance: compare $g(\text{cons}(a, x)) = h(a, x)$ and $g\,s = h(\text{cons}^- s)$.

The set of pairs view will remain a useful analogy, e.g., in defining f^- as the object represented by the inverse relation, which is functional iff f is 1-1.

(B) A typical non-minimalist design variant of Definition 13 would add: i.(c) a set $\mathcal{C}f$, called *the codomain of f*; ii.(c) $\mathcal{C}f = \mathcal{C}g$; iii.(c) $\mathcal{C}f = Y$.

Logically, composition and inverses *could* still be defined without restriction (exercise). However, authors using codomains do restrict $g \circ f$ by $\mathcal{D}g = \mathcal{C}f$ and define inverses for 1-1 functions only if the latter are "onto" [their codomain].

So the issue boils down to: what are the costs and the merits of codomains?

Shuard [54] published perhaps the only paper evaluating the function-with-codomain variant. Her single (!) argument *in favor* is using *onto* as an adjective. Yet, the ability to say that "f is onto Y but not onto Z" is more selective.

Shuard's argument *against* is more solid: simplicity. She states: "*Flett's definition wins hands down as simplicity in analysis is concerned*". Her next statement, "*In algebra, however, it is more convenient to start by mentioning the codomain of a function*" (resembling Myth #0), is left unsubstantiated. Excellent algebra texts such as Herstein [29] do fine with the standard variant.

In Shuard's paper and all other sources consulted, suggestions that attaching a codomain *might* be convenient turns out to be fallacious, typically overlooking that the standard view regarding $f : X \to Y$ already implies $\mathcal{R}f \subseteq Y$. This view covers all sensible purposes of $f : X \to Y$, "mentioning" Y included, and without making Y a function attribute. Only for topology further study is needed to determine whether viewing Y as an attribute of a function $f : X \to Y$ is just some tradition based on similar oversights or has genuine advantages.

Still, what's the harm in a function-with-codomain, beside complexity?

From a conceptual and practical viewpoint, burdening a function with a codomain affects all other definitions, complicates equality and impoverishes the function algebra for inverses, composition, merge, override and so on [42]. Shuard [54, p. 10] notes that the only analysis book that she knows to use codomains [57] gets into trouble by defining the inverse of a 1-1 function f to have domain $\mathcal{R}f$ as usual and ignoring that codomain users require surjectivity. She adds that "*the distinction between $f : A \to \mathbb{R}$ and $f_1 : A \to f(A)$ is so tedious that it is clearly better forgotten at this stage*". One might say: "*clearly better avoided from the start*", matching Smith's view on equality [55, p. 189].

Aside: programming versus mathematics Types and signatures of "functions" in programming [19, Section 3.8] and some proof assistants typically are unique attributes by design. They are easier to implement than general symbolic computation of, say, $\mathcal{D}(g \circ f)$, but remain rather crude approximations of types as partial specifications following the ISO standard. Indeed, the generality provided free of charge by the minimalist/standard view (illustrated for $g \circ f$ and f^-) is common fare in *mainstream mathematics*, by which we mean: the mathematics routinely applied by the large majority of users, ranging from mathematicians active in analysis/calculus, linear algebra, discrete math etc. to engineers active in signals and systems. Such users would be justified in dismissing as impractical any definition that infringes on these "acquired rights".

Similar considerations in the context of specification languages are found in [35]. In the specification language TLA$^+$ [36, p. 48], the notation $[X \to Y]$ has the meaning of $X \to Y$ as defined by the ISO standard.

Not surprisingly, nearly all calculus/analysis texts avoid codomains and simply proceed from Definition 6 or 13 or equivalent, the most complete picture being presented by Flett [17, pp. 4–6]. In this manner, calculus/analysis books succeed in giving a proper account in about one page, without being too terse, and often before page 10. Functions are too important to postpone their introduction beyond page 100, only to get them bogged down in unsoundness.

Conclusion For mainstream mathematics, attaching a codomain onto a function has no verified merits but increases complexity and reduces generality. Hence any definition that accepts such drawbacks entails a heavy obligation of justifying the design, even if it concerns only some niche area.

4. Engineering mathematical abstraction

Abstraction is a very useful intellectual tool in science, especially engineering. It allows reasoning about the essentials, without sidetracking by elements causing unnecessary complications. Of course, *unnecessary* is the operative word.

Being a tool, abstraction requires engineering. Criteria include not just soundness, but also effectiveness in reasoning. Parnas [45] deplored 25 years ago that "*Those working in theoretical computing science lack an appreciation for the simplicity and elegance of mature mathematics*". Little has changed. The context in [45] makes clear that *mature* refers to mainstream mathematics as characterized earlier. Symbolic reasoning has been kept from reaching maturity by decades of stifling in favor of "narrative" prose [62], but that is another story.

Engineering mathematical abstraction involves *balance*. Abstraction easily degenerates into obfuscation, making it ineffective as a tool for general use.

5. Illustration A: functions and the Halmos principle

The function concept is the result of a long evolution. Its "modern" formulation as in Definition 6 is a one-liner in the most positive sense: a degree of simplicity and clarity that is unlikely to be surpassed. This is typical for a final design. Arguably, non-equivalent variants need a different name to avoid confusion[1].

Yet even when using the set-of-pairs definition as a reference basis, in practice one rarely thinks about functions as sets, and set theory serves only as a handy framework, as noted by Halmos [26, p. 31], and illustrated by Definition 13.

Reducing concepts to a set-theoretic representation is conceptually chafing and causes "freak properties" or "accidental facts" [26, pp. 25, 45]. Two examples: defining a function as a set (of pairs) yields $(x, y) \in f$, and defining an ordered pair (a, b) as $\{\{a\}, \{a, b\}\}$ yields $\{a, b\} \in (a, b)$. Halmos considers such effects "*a small price to pay for conceptual economy*" and offers two solutions.

One solution is axiomatization. Bourbaki [9, p. 68] characterizes pairs by the equality axiom $(x, y) = (x', y') \Rightarrow x = x' \wedge y = y'$. Definition 13 characterizes functions by the equality axiom $f = g \Leftarrow \mathcal{D}f = \mathcal{D}g \wedge \forall x : \mathcal{D}f . f(x) = g(x)$. In both examples, the converse is trivial by Leibniz's principle [24, p. 60].

The other solution consists in using overly concrete definitions only to derive theorems that capture the essence of a concept, and then declaring "*the definition has served its purpose by now and will never be used again*" [26, p. 25]. Although many others tacitly follow this approach as well, we name it the *Halmos principle* after the author who made it explicit as a design strategy.

According to Halmos, "*The mathematician's choice is between having to remember a few more axioms and forgetting a few accidental facts; the choice is pretty clearly a matter of taste*". Still, expressiveness and support for reasoning are more reliable criteria. The choice also depends on each specific concept.

---

[1]Halmos [25] is quite amused by Bourbaki's habit of abandoning their "innovations" in favor of common terminology. Still, Bourbaki's term *application* for the "triples" variant [9, p. 76] at least avoids confusion with using *function* for the common variant [9, p. 77].

For instance, distinguishing functions from sets as in Definition 13 helps disambiguating common expressions such as f^n and S^n. This is secondary.

For reasoning, the "set of pairs" view remains a powerful simplifying analogy, as shown for composition and inverses. It is linked to Definition 13 as the *graph* of a function: $\mathcal{G}f = \{(x, f\,x)\,|\,x : \mathcal{D}f\}$. Clearly, $f = g \Rightarrow \mathcal{G}f = \mathcal{G}g$ by Leibniz's principle. By Definition 13.ii, $\mathcal{G}f = \mathcal{G}g \Rightarrow f = g$. Hence $\mathcal{G}$ is invertible and can be omitted (left implicit) when using the "set of pairs" view as an analogy.

6. Illustration B: Preparing category theory for engineering

6.1. Categories and the Arnold Principle

The practical potential of category theory has been demonstrated in [7]. This book, as well as [2], are among the best introductions to the subject.

Still, an engineer would ask: *why* use category theory? A non-evasive answer is: to support the algebraic elegance and practical benefits of the point-free style, which has proven useful throughout engineering. For instance, in systems modeling, this style has helped making common notations more general [11] and free of errors that "undermine the students' confidence in mathematics" [39].

Why *category* theory? Because it is ready-made and seems plug-and-play.

Yet, the literature based on this view reveals grave mismatches. Category theory is about *arrows*. In the *strict* (i.e., textbook) variant, each arrow a has a unique *source* src a and *target* tgt a. Composition $b \circ a$ requires src $b =$ tgt a. In this setup, a *function f* corresponds to many *arrows* (one for each triple f, A, B), as noted by Bird [7, p. 26] and Pierce [46, p. 2]. The target attribute inherits all burdens of the codomain. Hence, in strict category theory, arrows do not fit *relations* and *functions* but *correspondences* and *applications* instead.

Pierce [46, p. 3] adds that arrows fit functions in their "ordinary mathematical meaning" only in variants of category theory where one cannot always tell for an arrow what its source and target are. But uniqueness of source and target is precisely why users of strict category theory feel forced to restrict composition!

Such design decisions bring to mind the following familiar parable.

The Suit A man buys a new suit. When he finds it too tight at some places and too loose at others, the tailor shows him how twisting his arms, turning his feet and bending backwards brings relief. When the customer proudly hobbles along in his new suit, a passerby observes "How sad, this man must have had a terrible accident". Says his companion, "Yes, but he is lucky to have found a true master tailor who can make such a perfectly fitting suit".

The moral is that well-designed axiomatization is fitted to existing concepts, not the other way around like a straitjacket. Typical examplary designs are group, ring and field theory [29]: they capture commonalities of existing concepts of quite different nature, without distorting them to fit the axioms.

Arnold [3], one of the greatest 20th century mathematicians, denounces improper use of abstraction by "the criminal algebraists-axiomatisators". Despite this hyperbole, one can hardly deny his observation that often "*the so-called 'axioms' are in fact just (obvious) properties* [of the concepts of interest]". This

suggests an apt guideline, here called the *Arnold Principle*: unless an axiomatization is applied regularly to objects more abstract than those of primary interest, it is obfuscation. Arnold mentions Whitney's theorem to warn that more abstract objects may not always exist. Even if they do exist, proper axiomatization unifies them with the objects of primary interest, but never constrains the latter, which would amount to upside-down design. Inability to adequately capture functions, "the single most important and universal notion that runs through all of mathematics" [29], inhibits the practicality of category theory.

Rather than going for extremes such as dismissing either category theory or its flaws, we show how categorical concepts can suit mainstream mathematics.

6.2. Reaping the benefits of category theory without the constraints
Flawed axiomatization by upside-down design is most effectively avoided by starting from the objects of interest and deriving theorems reflecting abstract properties [3, 10]. When appropriate (by Arnold's principle), such theorems can be recast as axioms afterwards. The proofs then become evidence that the objects of interest do satisfy these axioms. Thus, no effort is wasted.

Proper design avoids misleading terminology by using *relation* and *function* only in their ordinary mathematical meaning. For the concepts captured by strict category theory, safe terms are *correspondence* and *application* (or alternatives). Abstractions of correspondences are often called *allegories* [7, 18].
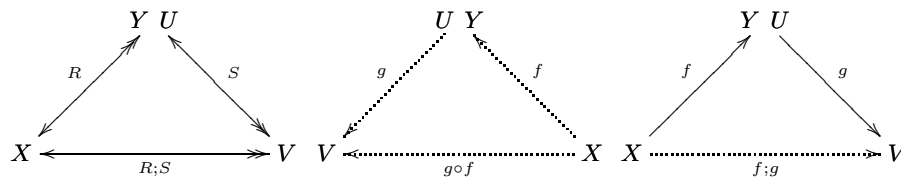
The relevant benefits of category theory are point-free expression and reasoning. Still, good design should not cause a rift between point-free and point-wise styles, since practical applicability requires safe and smooth mixing of styles.

One approach along these design guidelines is *concrete category theory*, which starts from the primary objects of interest (relations and functions as usual, with the standard view on $f : X \to Y$), and hence is "ordinary mathematics".

(i) We assume a *relation* R is defined conceptually in the style of Definition 13 (exercise), with the set of pairs view as an analogy: $y\,R\,x \equiv (x, y) \in \mathcal{G}R$ and $R = S \equiv \mathcal{G}R = \mathcal{G}S$. The domain $\mathcal{D}\,R$ and the range $\mathcal{R}\,R$, defined via Definition 2, are proper attributes of R. Bringing them into the picture often yields sharper results ($=$ instead of $\subseteq$). Composition $S \circ R$ (or $R\,;S$) and converse $R^{\smile}$ follow Definition 4. The *identity relation* id_A *on* A with $\mathrm{id}_A = \{(a, a)\,|\,a : A\}$. Predicate calculus yields the usual collection point-free formulas.

(ii) A relation R is *functional* iff $R \circ R^{\smile} = \mathrm{id}_{\mathcal{R}R}$ (equivalently, $R \circ R^{\smile} \subseteq \mathrm{id}_Y$ for $R : X \leftrightarrow Y$). Such a relation is called a *function*, typically written f, g, ...

Symbolism can be augmented by diagrams of arrows labeled by relation or function expressions, e.g., $\xrightarrow{f}\xrightarrow{g} = \xrightarrow{f;g}$. Optionally, types can be specified by labels at endpoints of arrows and shaping the latter like the type arrows $\leftrightarrow\!\!\!\rightarrow$, $\to$ and $\nrightarrow$, more conveniently drawn as $\cdots\!\!\rightarrow$. Here are some illustrations:

13

Recall that types are partial specifications for, not attributes of, the arrows. Normally, extra structure naturally merges endpoint labels at a common vertex. Then diagrams look exactly as in strict category theory, but without constraints. In the following examples, it will become evident that attaching a target or codomain to a function or relation would only provide negative added value.

6.3. Some illustrations of concrete category theory — selected topics

Convention In some contexts, *family* is a graphic synonym for *function* [9, p. 77] [26, p. 34]. An *I-family* is a function with domain I, called *index set*.

Product. The first topic has significant fundamental and practical interest.

Definition 14 (Product). *The* Cartesian product ΠT *of a family T of sets is the set of functions f such that $\mathcal{D}f = \mathcal{D}T$ and $f(x) \in T(x)$ for all x in $\mathcal{D}f$.*

Products support *dependent types* and *tolerances on functions* [11] that can be tight, e.g., if T is a family of singletons, declaring $f : \Pi T$ fully specifies f. Hence types as partial specifications can be arbitrarily fine, and $\Pi T \subseteq \mathcal{D}T \to \cup T$.

Another example is $T := (X, Y)$. An *n-tuple* is a function whose domain consists of the first n natural numbers [26, p. 45][34, p. 75][50, p. 9]. For an n-tuple T of sets, one can use infix notation, e.g., $\Pi(X, Y) = X \times Y$. This view also resolves the "terminological friction" discussed by Halmos [26, p. 36].

To obtain the usual point-free characterization of ΠT, we let $I := \mathcal{D}T$ and define an I-family π of *projection functions* by $\pi_i \in \Pi T \to T_i$ and $\pi_i t = t_i$. Note: an illustrative equivalent declaration is $\pi : \Pi_{i:I}(\Pi T \to T_i)$.

Theorem 2. *Let T be an I-family of sets, all nonempty* (to avoid $\Pi T = \emptyset$).
(i) *For any set S and I-family f of functions with $f_i \in S \to T_i$, there exists a unique $g : S \to \Pi T$ satisfying $f_i = \pi_i \circ g$. Specifically, $g = f^\mathsf{T}$* (transpose).
(ii) *Let γ be an I-family of functions with $\gamma_i \in C \to T_i$ and the property that, for any set S and I-family f of functions with $f_i \in S \to T_i$, there exists a unique $h : S \to C$ satisfying $f_i = \gamma_i \circ h$. Then there is a bijection between C and ΠT.*
Proof (i) Solving $f_i = \pi_i \circ g$ for g: for any i in I and s in S, $f_i s = (\pi_i \circ g)\,s = \pi_i(g\,s) = (g\,s)\,i$. By definition, $(f^\mathsf{T}s)i = f_i s$, so $g = f^\mathsf{T}$. *Example*: let $f := \pi$, so $\pi_i = \pi_i \circ \pi^\mathsf{T}$. Note that $(\pi^\mathsf{T}t)i = \pi_i t = t_i = (\mathrm{id}_{\Pi T}t)i$. In fact, $\pi^\mathsf{T} = \mathrm{id}_{\Pi T}$.
(ii) Letting $f := \gamma$ in (i), $\gamma_i = \pi_i \circ \gamma^\mathsf{T}$. Letting $f := \pi$ in (ii), $\pi_i = \gamma_i \circ h$. Hence $\gamma_i = \gamma_i \circ h \circ \gamma^\mathsf{T}$ and $\pi_i = \pi_i \circ \gamma^\mathsf{T} \circ h$. By uniqueness, $h \circ \gamma^\mathsf{T} = \mathrm{id}_C$ and $\gamma^\mathsf{T} \circ h = \mathrm{id}_{\Pi T}$. So γ^T is a bijection from C to ΠT and h is its inverse.

This compact yet detailed proof was made possible by the *generic operator* $^\mathsf{T}$ for *transposition* [11], defined by $(f^\mathsf{T}s)i = f_i s$, as recalled inside the proof. Dependencies may be easier to trace by writing g_f for g in part (i) and $\mathrm{h}_{\gamma,f}$ for h in part (ii) of the statement. Thus, $f := \pi$ makes $\mathrm{h}_{\gamma,\pi}$ the inverse of γ^T.

Part (i) is depicted in Figure 1. Dashed lines reflect multiple instances, one for each i in I, forming a 3D cone. The proof of (ii) shows that every I-family γ with the stated property is isomorphic to π, hence "unique up to isomorphism".
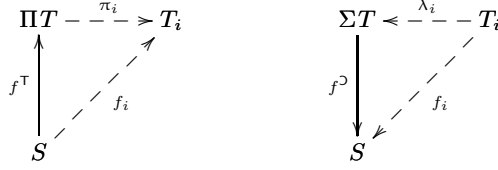
Figure 1: Concrete Categorical Cartesian Product and Disjoint Union

This *theorem* faithfully reflects the *definition* of products in category theory [46, p. 19], while avoiding "unacceptable but generally accepted" notations like $(T_i)_{i \in I}$ for just T and, worse, $\langle f_i \rangle_{i \in I}$ for just $\langle f \rangle$, which was proven to be f^T.

Aside Clearly, $\gamma : \Pi_{i:I}(C \to T_i)$ depends on C and T. Similarly, π depends on T, which is why some authors write π^T. One can also see the π_i as *operators* (Lamport [36, p. 69]) which, unlike functions, do not have a domain. A third view is offered by theories allowing for the set (or *class*) of all pairs, such as the algebra of relations in Tarski [61] or set theory with a universal set $\mathcal{U}$ in Holmes, where axioms [28, p. 30] assert the existence of the *equality relation* $\{(x, x) \mid x : \mathcal{U}\}$ and *projection relations* such as $\{((x, y), x) \mid x : \mathcal{U}; y : \mathcal{U}\}$.

Sum. This topic illustrates how objects with quite different point-wise definitions resemble each other in point-free form, more specifically as duals.

Definition 15 (Disjoint union). *The* disjoint union ΣT *of a family T of sets is the set of (ordered) pairs such that* $(i, x) \in \Sigma T$ *iff* $i \in \mathcal{D} T$ *and* $x \in T_i$.

This concept is used to express choice in formal language semantics [42, p. 74].

To obtain the usual point-free characterization of ΣT, we let $I := \mathcal{D} T$ and define an I-family λ of *labeling functions* by $\lambda_i \in T_i \to \Sigma T$ and $\lambda_i x = (i, x)$.

Theorem 3. *Let T be an I-family of sets.*
(i) *For any set S and I-family f of functions with $f_i \in T_i \to S$, there exists a unique $g : \Sigma T \to S$ satisfying $f_i = g \circ \lambda_i$. Specifically, $g = f^\circ$ (uncurry).*
(ii) *Let δ be an I-family of functions with $\delta_i \in T_i \to D$ and the property that, for any set S and I-family f of functions with $f_i \in T_i \to S$, there exists a unique $h : D \to S$ satisfying $f_i = h \circ \delta_i$. Then there is a bijection between D and ΣT.*
Proof (i) Solving $f_i = g \circ \lambda_i$ for g: for any i in I and s in S, $f_i s = (g \circ \lambda_i)\, s = g(\lambda_i s) = g(i, s)$. By definition, $f^\circ(i, s) = f_i s$, so $g = f^\circ$. *Example*: let $f := \lambda$, so $\lambda_i = \lambda^\circ \circ \lambda_i$. Note that $\lambda^\circ(i, x) = \lambda_i x = (i, x) = \mathrm{id}_{\Sigma T}(i, x)$. In fact, $\lambda^\circ = \mathrm{id}_{\Sigma T}$.
(ii) Letting $f := \delta$ in (i), $\delta_i = \delta^\circ \circ \lambda_i$. Letting $f := \lambda$ in (ii), $\lambda_i = h \circ \delta_i$. Hence $\delta_i = \delta^\circ \circ h \circ \delta_i$ and $\lambda_i = h \circ \delta^\circ \circ \lambda_i$. By uniqueness, $\delta^\circ \circ h = \mathrm{id}_D$ and $h \circ \delta^\circ = \mathrm{id}_{\Sigma T}$. So δ° is a bijection from D to ΣT and h is its inverse.

Relations, point-free style. Relational properties of the Cartesian product can be similarly expressed and explored in point-free style. For instance, given an I-family R of relations R_i from S to T_i, let us aim for a relation G from S to ΠT such that $\pi_i \circ G = R_i$. This requires $\pi_i^\smile \circ \pi_i \circ G = \pi_i^\smile \circ R_i$ and, since

$\mathrm{id}_{\Pi T} \subseteq \pi_i^{\smile} \circ \pi_i$, also $G \subseteq \pi_i^{\smile} \circ R_i$ (for all i in I), hence $G \subseteq \bigcap i : I . \pi_i^{\smile} \circ R_i$. Let us define $\langle R \rangle$ as $\bigcap i : I . \pi_i^{\smile} \circ R_i$, so $t\langle R \rangle s \equiv I = \emptyset \vee (t \in \Pi T \wedge \forall i : I . t_i R_i s)$.

In general, this only yields $\pi_i \circ \langle R \rangle \subseteq R_i$, as in category theory. Still, if all R_i share a common domain, a sharper result is $\pi_i \circ \langle R \rangle = R_i$, the design goal. Even sharper is the pointwise form $y (\pi_i \circ \langle R \rangle) s \equiv y R_i s \wedge s \in \bigcap i : I . \mathcal{D} R_i$.

Assuming $I \neq \emptyset$, let T and T' be I-families of sets and R an I-family of relations R_i from T_i to T_i'. Define $||R$ from ΠT to $\Pi T'$ by $||R = \langle i : I . R_i \circ \pi_i \rangle$, generalizing [7, p. 114]. Now $t'(||R)t \equiv t' \in \Pi T' \wedge t \in \Pi T \wedge \forall i : I . t_i' R_i t_i$.

Functors. A *functor* in category theory maps objects to objects and arrows to arrows. Often one uses the same symbol for both maps, e.g., writing $\times$ for $||$, but that is not necessary [7, p. 30]. Although notational economy is commendable, in "working mathematics" one prefers combining various concepts as convenient, yet still avoid notational confusion. Hence, if one views relations as sets, the need to distinguish between $R || S$ and $R \times S$ is evident. The operator $||$ is called *shuffle* in some functional languages and *parallel* by Meyer [42, p. 36].

Tabulations. Tabulations are helpful in point-free reasoning [7]. Let $R : X \leftrightarrow Y$. Now $f : R \to X$ with $f(x, y) = x$ and $g : R \to Y$ with $g(x, y) = y$ satisfy both $R = g \circ f^{\smile}$ and $(f^{\smile} \circ f) \cap (g^{\smile} \circ g) = \mathrm{id}_R$.

In category theory, a *tabulation* of a correspondence arrow r from X to Y is a pair of application arrows f from Z to X and g from Z to Y jointly satisfying $r = g \circ f^{\smile}$ and $(f^{\smile} \circ f) \cap (g^{\smile} \circ g) = \mathrm{id}_Z$. Similar concepts in a more general setting are discussed in Tarski [61, p. 96].

Currying. In its most basic form, Currying transforms a function $f : X \times Y \to Z$ into a function f^{C} of type $X \to (Y \to Z)$ such that $f^{\mathsf{C}} x y = f(x, y)$. For a point-free characterization [7, p. 72][46, p. 33], one uses an *evaluation* or *application* function $\alpha_{Y,Z} : (Y \to Z) \times Y \to Z$ defined by $\alpha_{Y,Z}(g, y) = g(y)$. One can verify that $f = \alpha_{Y,Z} \circ (f^{\mathsf{C}} || \mathrm{id}_Y)$; in fact, $h = f^{\mathsf{C}} \equiv \alpha_{Y,Z} \circ (h || \mathrm{id}_Y) = f$ (uniqueness).

$$X \times Y \xrightarrow{\ f\ } Z$$

$$f^{\mathsf{C}} || \mathrm{id}_Y \quad \alpha_{Y,Z}$$

$$(Y \to Z) \times Y$$

The categorical view is that the object $X \to Y$ (often written Y^X) is fully characterized by the existence of an arrow $\alpha_{Y,Z} : (Y \to Z) \times Y \to Z$ such that there is a unique arrow f^{C} satisfying $f = \alpha_{Y,Z} \circ (f^{\mathsf{C}} || \mathrm{id}_Y)$.

In the wider setting of common mathematics, f^{C} is defined for any function f whose domain is a set R of pairs as follows: $\mathcal{D} f^{\mathsf{C}} = \mathcal{D} R$ and, for any x in $\mathcal{D} f^{\mathsf{C}}$, $f^{\mathsf{C}} x \in \{y : \mathcal{R} R \,|\, (x, y) \in R\} \to \mathcal{R} f$ with $f^{\mathsf{C}} x y = f(x, y)$. *Example*: if $\mathcal{D} f = \{(0, 5), (3, 1), (3, 2)\}$ then $\mathcal{D} f^{\mathsf{C}} = \{0, 3\}$, $\mathcal{D} (f^{\mathsf{C}} 3) = \{1, 2\}$ and $f^{\mathsf{C}} 3\, 2 = f(3, 2)$.

Conversely, uncurrying $F^{\mathsf{\supset}}$ is defined for any family of functions F as follows: $F^{\mathsf{\supset}} \in \{(x, y) : \mathcal{D} F \times \bigcup (\mathcal{D} \circ F) \,|\, y \in \mathcal{D} (F x)\} \to \bigcup (\mathcal{R} \circ F)$ with $F^{\mathsf{\supset}}(x, y) = F x y$. Note that $(f^{\mathsf{C}})^{\mathsf{\supset}} = f$, but $(F^{\mathsf{\supset}})^{\mathsf{C}} = F$ only if $\mathcal{R} F$ contains no empty function.

7. Concluding remarks

The first part of this paper argued that mathematical definitions and notations are the result of *design* and hence benefit from engineering principles.

This was illustrated for some very basic concepts that have fallen into disrepair, apparently due to habits that noted mathematicians consider "unacceptable" or evidence of "glaring perversity". Indeed, only perversity explains the neglect for so many impeccable accounts that have been around since decades.

Simple design guidelines go a long way. In particular, definitions require diligent engineering. The discipline of always explicitly justifying design decisions incites more thoughtfulness. Halmos [26] provides an unsurpassed example.

Judicious use of symbolism is another invaluable tool for sanity checking. Defining concepts not just in prose but also by very simple formulas, as in Suppes [60, p. 57, 86], arguably would have avoided the *codomain* blunder.

One empirical engineering principle that does *not* hold in mathematics is that performance requires complexity. A "crystal radio" needs just one (passive) semiconductor; a software-defined radio (SDR) performs better but uses billions of transistors. For relations and functions, the standard variant is the simplest, yet was found more versatile and general than variants with codomains. Clearly a very onerous burden of justification rests on such variant definitions!

The second part focused on making category theory practical. Historically, practical use was not a concern, so the need for re-engineering is no surprise.

Appreciating this fact took a scholar like José Oliveira, supported by his background in electrical engineering. One of his recent papers [44] points out how relation algebra requires adequate *preparation* to suit the intended purpose.

Whereas [44] is more advanced, we have shown how liberating category theory from unjustified constraints (similar to codomains) extends its practical applicability while preserving its essential ideas, and how full compatibility with "ordinary mathematics" makes the style of expression and reasoning accessible to anyone with a high school background and a taste for algebraic elegance.

In this way, the basic ideas may become useful and appealing to a wider scientific community rather than remain jargon for a small group of aficionados.

References

References

[1] Tom M. Apostol, *Calculus, Vol. I*, 2nd. ed. John Wiley (1967)

[2] Michael A. Arbib and Ernest G. Manes, *Arrows, structures and functors — The Categorical Imperative.* Academic Press (1975)

[3] Vladimir I. Arnold, *On Teaching Mathematics.* Lecture on mathematics education, Paris (1997) http://pauli.uni-muenster.de/~munsteg/arnold.html

[4] Luís S. Barbosa, "Towards a Calculus of State-based Software Components", *Journal of Universal Computer Science 9*, 8, pp. 891–909 (2003)

[5] Robert G. Bartle, *The Elements of Real Analysis*. John Wiley (1964)

[6] Isabella G. Bashmakova and Galina S. Smirnova, "The literal algebra of Viète and Descartes", *The American Mathematical Monthly 106*, 3, pp. 260–263 (Mar. 1999)

[7] Richard Bird and Oege De Moor, *Algebra of Programming*. Prentice Hall (1997)

[8] Ethan Bloch, *Proofs and Fundamentals*. Springer (2011)

[9] Nicolas Bourbaki, *Théorie des ensembles*. Hermann & c^{ie} (1954)

[10] Raymond Boute, "On the shortcomings of the axiomatic approach as presently used in Computer Science", in: *CompEuro '88. 'Design: Concepts, Methods and Tools'*, pp. 184-193 (Apr. 1988)

[11] Raymond Boute, "Concrete Generic Functionals", in: Jeremy Gibbons and Johan Jeuring, Eds., *Generic Programming*, pp. 89-119. Kluwer (2003)

[12] Carl B. Boyer and Uta C. Merzbach, *A History of Mathematics*. Wiley (1991)

[13] Gary Chartrand, Albert Polimeni and Ping Zhang, *Mathematical Proofs: A Transition to Advanced Mathematics (3rd. ed.)*. Pearson (2012)

[14] Ulrich Daepp and Pamela Gorkin, *Reading, Writing and Proving: a Closer Look at Mathematics*. Springer (2003)

[15] Ulrich Daepp and Pamela Gorkin, *Reading, Writing and Proving: a Closer Look at Mathematics (2nd. ed.)*. Springer (2011)

[16] Abhijit Dasgupta, *Set Theory*. Birkhäuser (2014)

[17] Thomas M. Flett, *Mathematical Analysis*. McGraw-Hill (1966)

[18] Peter J. Freyd and Andre Scedrov, *Categories, Allegories*. North Holland (1990)

[19] Rowan Garnier and John Taylor, *Discrete Mathematics — Proofs, Structures and Applications*. CRC Press (2010)

[20] Larry Gerstein, *Introduction to Mathematical Structures and Proofs (2nd. ed.)*. Springer (2012)

[21] Judith L. Gersting, *Mathematical Structures for Computer Science (7th. ed.)*. W. H. Freeman (2013)

[22] Linda Gilbert and Jimmie Gilbert, *Elements of Modern Algebra (7th. ed.)*. Cengage Learning (2008)

[23] Edgar G. Goodaire and Michael M. Parmenter, *Discrete Mathematics with Graph Theory, 3rd. ed.*. Pearson Prentice Hall (2006)

[24] David Gries and Fred B. Schneider, *A Logical Approach to Discrete Math*. Springer (1993)

[25] Paul R. Halmos, "Nicolas Bourbaki", *Scientific American, 196*, 5, pp. 88-99 (May 1957)

[26] Paul R. Halmos, *Naive Set Theory*. Van Nostrand Reinhold (1960)

[27] Richard Hammack, *Book of Proof*. CC BY-ND (2009)

[28] Randall Holmes, *Elementary Set Theory with a Universal Set*. (on the Web)

[29] Israel Herstein, *Topics in Algebra*. Xerox College Publishing (1964)

[30] ISO/IEC, *Quantities and units — Part 2: Mathematical signs and symbols to be used in the natural sciences and technology*. ISO 80000-2 (2009)

[31] Thomas Jech, *Set Theory*. Springer (2003)

[32] Morris Kline, *Mathematics in Western Culture*. Oxford University Press (1953)

[33] Andrey L. Kolmogorov and Sergey V. Fomin, *Introductory Real Analysis*. Dover (1970)

[34] Steven G. Krantz, *Real Analysis and Foundations*. Chapman & Hall/CRC (2005)

[35] Leslie Lamport and Lawrence C. Paulson, "Should your specification language be typed?", *ACM TOPLAS 21*, 3, pp. 502–526 (May 1999)

[36] Leslie Lamport, *Specifying Systems — The TLA+ Language and Tools for Hardware and Software Engineers*. Pearson (2003)

[37] Serge Lang, *Undergraduate Analysis*. Springer-Verlag (1983)

[38] Ron Larson and Bruce Edwards, *Calculus 9e*. Brooks/Cole (2009)

[39] Edward A. Lee, Pravin Varaiya, "Introducing Signals and Systems, the Berkeley Approach". *First Signal Processing Education Workshop*, Hunt, Texas (Oct. 2000)

[40] Saunders Mac Lane, *Categories for the Working Mathematician*. Springer (1971)

[41] Eliott Mendelson, *Introduction to Mathematical Logic (3rd. ed.)*. Wadsworth & Brooks/Cole (1987)

[42] Bertrand Meyer, *Introduction to the Theory of Programming Languages*. Prentice Hall (1991)

[43] José N. Oliveira, *Formal Specification, Rapid Prototyping and Program Calculation — an Industrial Experiment using the CAMILA/SETS Approach*. UNU/IIST Seminar, Macau (May 1997)

[44] José N. Oliveira, "Preparing Relational Algebra for 'Just Good Enough' hardware". In: Peter Höfner et al. (Eds.), *Relational and Algebraic Methods in Computer Science*, pp. 119–138. Springer LNCS 8428 (2014)

[45] David L. Parnas, "Education for Computing Professionals", *IEEE COMPUTER 23*, 1, pp. 17–22 (Jan. 1990)

[46] Benjamin C. Pierce, *Basic Category Theory for Computer Scientists*. The MIT Press (1991)

[47] Willard V. Quine, *Set Theory and Its Logic*. Belknap Press of Harvard University Press (1969)

[48] John C. Reynolds, *Theories of Programming Languages*. Cambridge University Press (1998)

[49] Charles E. Roberts, Jr., *Introduction to Mathematical Proofs — A Transition*. CRC Press (2010)

[50] Halsey L. Royden, *Real Analysis*. Macmillan (1968)

[51] Walter Rudin, *Principles of Mathematical Analysis*. McGraw-Hill (1964)

[52] Lucio Russo, *The Forgotten Revolution*. Springer (2004)

[53] Edward R. Scheinerman, *Mathematics — A Discrete Introduction (3rd. ed.)*. Cengage Learning (2012)

[54] Hilary S. Shuard, "Does it matter?", *The Mathematical Gazette 59*, 407, pp. 7–15 (Mar. 1975)

[55] Douglas Smith, Maurice Eggen and Richard St. Andre, *A Transition to Advanced Mathematics*. Cengage Learning (2010)

[56] John Michael Spivey, *The Z Notation – A Reference Manual*. Prentice-Hall (1989)

[57] David Sprecher, *Elements of Real Analysis*. Academic Press (1970)

[58] James B. Stewart, *Calculus: Early Transcendentals (7th. ed.)*. Cengage Learning (2010)

[59] Michael P. Sullivan and Kathleen Miranda, *Single Variable Calculus — Early Transcendentals*. W. H. Freeman (2014)

[60] Patrick Suppes, *Axiomatic Set Theory*. Dover (1972)

[61] Alfred Tarski and Steven Givant, *A Formalization of Set Theory Without Variables*. The American Mathematical Society (1987, reprinted with corrections 1988)

[62] Daniel J. Velleman, *How To Prove It: A Structured Approach (2nd. ed.)*. Cambridge (5th printing 2009)

[63] Walter D. Wallis, *A Beginner's Guide to Discrete Mathematics (2nd. ed.)*. Birkhäuser (2012)

[64] Eugene Wigner, "The unreasonable effectiveness of mathematics in the natural sciences", *Communications on Pure and Applied Mathematics 13*, 1, pp. 1–14 (Feb. 1960)

[65] Elias Zakon, *Mathematical Analysis, Vol. I*. Trillia (2004)