Martin-Luther-Universität
Halle-Wittenberg

Mathematical notation and the use of functions

Henning Thielemann

ICIAM 2007

1 Summary

In contrast to natural languages, mathematical notation is accepted as being exceptionally precise. It shall make mathematical statements unambiguous, it shall allow formal manipulation, it is model for programming languages, computer algebra systems and machine provers. However, what is traditional notation and is it indeed as precise as expected?
Following we discuss some examples of notation which require caution. How are they adapted in computer algebra systems? Can we replace them by something better? What can we learn from functional programming?

2 Comparisons

2.1 Do we need macros?

2.1.1 Traditional

A notation where some symbols do not denote mathematical objects, but the notation can be understood only as a whole, we will refer to as a *macro*. A macro represents a textual replacement rather than a mathematical object.

Examples of Macros:

$$\begin{aligned} F(x)+C &= \int f(x)\,\mathrm{d}x \\ \sin x &= x+O(x^3) \quad \text{for} \quad x\to 0 \\ 0 &= \lim_{n\to\infty}\frac{1}{n} \\ &\lim_{x\to y} f(x) \end{aligned}$$

Manipulation of macros requires to be aware of the whole macro. Basic calculation laws become invalid for local manipulations.

$$\begin{aligned} F(2)+C &= \int f(2)\,\mathrm{d}2 \\ 2\cdot\sin x &= 2\cdot x+O(x^3) \quad \text{for} \quad x\to 0 \\ \sin x &= x+O(x^3) \quad \text{for} \quad x\to 0 \end{aligned}$$

by subtraction:

$$\sin x = x \quad \text{for} \quad x\to 0$$

2.1.2 Functional

In contrast to macros, *functions* are mathematical objects. They do not depend on the description of their arguments. E.g., for all functions f, $f(2+2)$ and $f(4)$ have the same value. Functions can be itself arguments to other functions. In short: Functions are more safe and more flexible than macros.

Functional Programming provides higher order functions and Functional Analysis provides function operators to carry these properties over to more complex situations.

Functional replacements for the examples above:

Function functional: $\int_x^y \in (\mathbb{R}\to\mathbb{R})\to\mathbb{R}$

$$F(y)-F(x) = \int_x^y f$$

Map to function set: $O \in (\mathbb{R}\to\mathbb{R})\to\mathscr{P}(\mathbb{R}\to\mathbb{R})$

$$\sin-\mathrm{id} \in O(\mathrm{id}^3)$$

Sequence functional: $\lim \in (\mathbb{N}\to\mathbb{R})\to\mathbb{R}$

$$0 = \lim\left(\frac{1}{n} : n\in\mathbb{N}\right)$$

Continuous continuation: $\mathrm{cont} \in (\mathbb{R}\to\mathbb{R})\to(\mathbb{R}\to\mathbb{R})$

$$\mathrm{cont}\, f\; y$$

No problem with manipulations:

$$\begin{aligned} 2\cdot(\sin-\mathrm{id}) &\in O(\mathrm{id}^3) \\ \sin-\mathrm{id} &\in O(\mathrm{id}^3) \end{aligned}$$

by subtraction, since $O(\mathrm{id}^3)$ is a vector space:

$$\sin-\mathrm{id} \in O(\mathrm{id}^3)$$

2.2 Do we need differentials?

2.2.1 Traditional

Paper	$\left(\frac{\mathrm{d}}{\mathrm{d}x}\ln x\right)\Big\vert_{x=2}$
MuPad	`subs(diff(ln(x),x),x=2);`
Maple	`subs(x=2,diff(ln(x),x));`
Mathematica	`ReplaceAll[D[Log[x],x],x->2]`

Function composition is associative.

$$(f\circ g)(h(x)) = f((g\circ h)(x))$$

But here the order of evaluation counts.

$$\frac{\mathrm{d}}{\mathrm{d}2}\ln 2 \quad \text{vs.} \quad \frac{1}{x}\Big\vert_{x=2}$$

2.2.2 Functional

Instead of differentiation with respect to *variables*, differentiate *functions* with the operator D.

$$D\,(x\mapsto \ln x)\,(2) = D\ln 2 = \left(x\mapsto\frac{1}{x}\right)2 = \frac{1}{2}$$

Paper	$D\,(x\mapsto\ln x)\,(2) = D\ln 2$
MuPad	`D(x -> ln(x))(2) = D(ln)(2)`
Maple	`D(x -> ln(x))(2) = D(ln)(2)`
Mathematica	`Derivative[1][(Log[#])&][2]`
	`= Derivative[1][Log][2]`

2.3 Do we need variable manipulation?

2.3.1 Traditional

Some notations treat variables like mathematical objects.

- Asymptotic complexity

$$\begin{aligned} f(x) &= O(g(x)) \\ f(x) &\lesssim g(x) \end{aligned}$$

- Ordinary differential equation

$$y' = \lambda\cdot y\cdot x$$

- Polynomial [2]

$$\begin{aligned} f &\in K[x] \\ \tilde{f}(x) &= 1+2\cdot x+x^2 \end{aligned}$$

2.3.2 Functional

Functions allow modeling of dynamic situations by static objects.

- Instead of saying what happens with performance when the size of an computational problem increases, we consider relations between complexity functions.

$$\begin{aligned} f &\in O(g) \\ f &\lesssim g \end{aligned}$$

- Instead of a time dependent variable y, we consider a function y, $y\in\mathbb{R}\to\mathbb{R}$. This allows formulation of a differential equation like

$$\begin{aligned} y' &= \lambda\cdot y\cdot\mathrm{id} \\ \text{or} \quad y' &= x\mapsto\lambda\cdot y(x)\cdot x \end{aligned}$$

which can be solved by usual equation transformations.
- There is no need to associate polynomials with variables.

$$\begin{aligned} p &\in P(K) \\ p &= (1,2,1) \end{aligned}$$

Introduce variables by the evaluation homomorphism φ.

$$\begin{aligned} \varphi(p)(x) &= 1+2\cdot x+x^2 \\ \varphi(p)(x) &\in K[x] \end{aligned}$$

2.4 Do we need context?

2.4.1 Traditional

Independent random variables X and Y imply

$$\begin{aligned} E(X\cdot Y) &= EX\cdot EY \\ F_{X+Y} &= (F_X * F_Y)' \end{aligned}$$

Independence needs context, namely the probability space.

2.4.2 Functional

What about dropping random variables and relying merely on distribution functions? Associate a distribution function F with a "*distribution*" X_F. Let functions operate on these distributions. Whether distributions represent independent random variables depends on the probability operation.

$$\begin{aligned} E(X\cdot Y) &= EX\cdot EY \\ X_F+X_G &= X_{(F*G)'} \end{aligned}$$

2.5 Is traditional easier?

Is traditional notation more concise than functional one?

variable oriented	functional	
$f(\cdot)$	f	
$f:(x,y)\mapsto f(x,y)$	f	
$x[k]*y[k]$	$x*y$	
$\lim_{n\to\infty} a_n$	$\lim a$	
$\frac{\mathrm{d}f(x)}{\mathrm{d}x}$	$f'(x)$	
$f(\cdot+k)$	$f\leftarrow k$	
$O\left(n^2\right)$	$O\left(\mathrm{id}^2\right)$	
$x^2\lesssim x^3$	$\mathrm{id}^2\lesssim\mathrm{id}^3$	
$g(z^2)=\frac{1}{2}\cdot(h(z)+h(-z))$	$g=h\downarrow 2$	[6]
$f(x)\in\mathscr{L}(\mathbb{R})$	$f\in\mathscr{L}(\mathbb{R})$	

Is traditional notation more intuitive than functional one?

variable oriented	functional
$\varphi(2\cdot -k)$	$\varphi\to k\downarrow 2$

Is traditional notation more precise than functional one?

variable oriented	functional
$f'(g(x))$	$(f'\circ g)\,x$
	$(f\circ g)'\,x$
$f_x(x,x+y)$	$(\xi\mapsto f(\xi,x+y))'\,x$
	$(\xi\mapsto f(\xi,\xi+y))'\,x$
$f(g(\cdot))$	$x\mapsto f(g(x))$
	$f(x\mapsto g(x))$
	$f(g(x\mapsto x))$

3 Conclusions

Several traditional notations complicate formal manipulations. Notations based on functions are often less ambiguous and safer for manipulation.

References

[1] F. Cajori, *A History of Mathematical Notation*, Dover, New York, 1993.

[2] G.-M. Greuel and G. Pfister, *A* `Singular` *Introduction to Commutative Algebra*, Springer-Verlag, 2002.

[3] J. Harrison, *Formalized mathematics*, tech. rep., Åbo Akademi University, Department of Computer Science, Lemminkäisenkatu 14a, 20520 Turku, Finland, August 1996.

[4] H. Thielemann, *Quantorenmechanik*, i-mail, Zeitschrift der Fachschaft Mathematik/Informatik an der Martin-Luther-Universität Halle-Wittenberg, Wintersemester 2000/2001 (2000), pp. 15–16.

[5] ——, *Klein, aber o*, i-mail, Zeitschrift der Fachschaft Mathematik/Informatik an der Martin-Luther-Universität Halle-Wittenberg, Sommersemester 2002 (2002), pp. 11–16.

[6] ——, *Optimally matched wavelets*, PhD thesis, Universität Bremen, March 2006.