Logic and **L**ogical **P**hilosophy (2020)
DOI: 10.12775/LLP.2020.011
Published online: June 28, 2020

Jeffrey Ketland

Computation and Indispensablity

Abstract. This article provides a computational example of a mathematical explanation within science, concerning computational equivalence of programs. In addition, it outlines the logical structure of the reasoning involved in explanations in applied mathematics. It concludes with a challenge that the nominalist provide a nominalistic explanation for the computational equivalence of certain programs.

Keywords: applicability of mathematics; computation; indispensability

1. Nominalism

Nominalism is the claim that there are no abstract entities:

> Nominalism is the doctrine that there are no abstract entities. The term 'abstract entity' may not be entirely clear, but one thing that does seem clear is that such alleged entities as numbers, functions and sets are abstract — that is, they would be abstract if they existed. In defending nominalism therefore I am denying that numbers, functions, sets or any similar entities exist.
> Since I deny that numbers, functions, sets, etc. exist, I deny that it is legitimate to use terms that purport to refer to such entities, or variables that purport to range over such entities, in our ultimate account of what the world is really like. [Field, 1980, p. 1]

From the point of view of modern science, the class of entities whose existence is thereby denied is extremely comprehensive: expression types, syntax trees, numbers, sets, functions, quantities, orderings, graphs, groups, rings, fields, metric spaces, manifolds, magnetic and electric fields, wavefunctions, Lie groups, gauge connections, structure groups, etc.

Received September 23, 2019

Yet, as Quine emphasized in his writing, particularly from [Quine, 1948] onwards, mathematics is an integral part of science. First, Quine set out his (correct) account of ontological commitment:

> To be assumed as an entity is, purely and simply, to be reckoned as the value of a variable. In terms of the categories of traditional grammar, this amounts roughly to saying that to be is to be in the range of reference of a pronoun. Pronouns are the basic media of reference; nouns might better have been named propronouns. The variables of quantification, "something", "nothing", "everything", range over our whole ontology, whatever it may be; and we are convicted of a particular ontological presupposition if, and only if, the alleged presuppositum has to be reckoned among the entities over which our variables range in order to render one of our affirmations true. [Quine, 1948, p. 32]

This criterion (of being assumed as an entity) is then applied first to mathematics:

> [...] classical mathematics, as the example of primes between 1000 and 1010 clearly illustrates, is up to its neck in commitments to an ontology of abstract entities. [Quine, 1948, p. 32]

and then to scientific theories (here Quine appeals to a *coherentist* account of *epistemology*):

> Now what of classes or attributes of physical objects, in turn? A platonistic ontology of this sort is, from the point of view of a strictly physicalistic conceptual scheme, as much a myth as that physicalistic conceptual scheme itself is for phenomenalism. This higher myth is a good and useful one, in turn, in so far as it simplifies our account of physics. Since mathematics is an integral part of this higher myth, the utility of this myth for physical science is evident enough.
> [Quine, 1948, p. 37]

As he later put it:

> If we subscribe to our physical theory and our mathematics, as indeed we do, then we thereby accept these particles and these mathematical objects as real; it would be an empty gesture meanwhile to cross our fingers as if to indicate that what we are saying doesn't count.
> [Quine, 1973, p. 65]

Likewise, Putnam emphasized similar points:

> [...] mathematics and physics are integrated in such a way that it is not possible to be a realist with respect to physical theory and a

> nominalist with respect to mathematical theory [...]. If talk of numbers and "associations" between masses, etc., and numbers is theology (in the pejorative sense), then the Law of Universal Gravitation is likewise theology. [Putnam, 1975, pp. 74–75]

If there are no abstract entities, science is false. But we are (epistemically) committed to science as the most coherent and explanatory systemization of our sensory experience. And it is difficult to see how the integration of abstract entities within science can be dispensed with or eliminated. The resulting argument is now sometimes called the Quine-Putnam indispensability argument. It amounts to the two claims: first, that science is *inconsistent* with nominalism; and, second, that attempts to "nominalize" science are hopeless — the mathematics is *indispensable*.

There have been a variety of attempts to rebut these anti-nominalist arguments of Quine and Putnam. Quine himself had attempted just this in a 1947 paper with Nelson Goodman ("Steps Towards a Constructive Nominalism" [Quine and Goodman, 1947]), aiming to somehow eliminate quantification over syntactic types in the theory of syntax by quantification over *concrete syntactic token* (i.e., physical inscriptions, etc). But Quine quickly saw this as hopeless, and rapidly abandoned the program, as was clear in Quine 1948. As Church commented in a 1958 letter to Goodman:

> [...] additional tasks that 'nominalistic' (better, finitistic) syntax might be asked to accomplish [:] the deduction theorem; ... the proof of the rules of substitution as derived rules [...] Post's completeness theorem for the propositional calculus; the metatheorem that every quantifier-free theorem of first-order functional calculus has a quantifier-free proof; the principal results about the Skolem normal form; Gödel's incompleteness theorems ...; Gödel's relative consistency proof for the axiom of choice and the generalized continuum hypothesis; the results of Fraenkel, Lindenbaum, and Mostowski concerning the independence of the axiom of choice. [Church, 1958] in [Goodman, 1972]

Probably the most important, detailed and insightful challenge to Quine and Putnam's arguments is [Field, 1980], which gave a geometry-based programmatic nominalization of classical mechanics, as well as identifying the conservativeness property as both guaranteeing that mathematics itself adds no further non-mathematical content to a non-mathematical theory, while explaining the utility of mathematics in

speeding-up the deduction of non-mathematical sentences from other non-mathematical sentences.[1]

However, in the last fifteen years or so, debates concerning the indispensablity of mathematics have often turned away from attempted nominalistic *reformulations* of scientific theories, to questions about whether the role of mathematics in science is *explanatory*. See, for example, [Baker, 2005], and the ensuing debate [e.g., summarized in Mancosu, 2018]. So, by and large, nominalists have conceded to Quine and Putnam that science is inconsistent with nominalism — that is, nominalists consider our best scientific theories to be *false*. This was Quine's and Putnam's central point. But even so, they insist that these mathematicized scientific laws do not even *explain* their predictions.

For example, my favourite physical law is one of Maxwell's laws:

$$\nabla \cdot \mathbf{B} = 0$$

This asserts that a certain (axial) *vector field* (i.e., the magnetic field $\mathbf{B}$) has zero divergence, everywhere. What is a vector field? A vector field is a *function* which maps each point in space(time) to a *vector*. Nominalists assert that the magnetic field $\mathbf{B}$ does not exist. I assert that the magnetic field most certainly does exist.

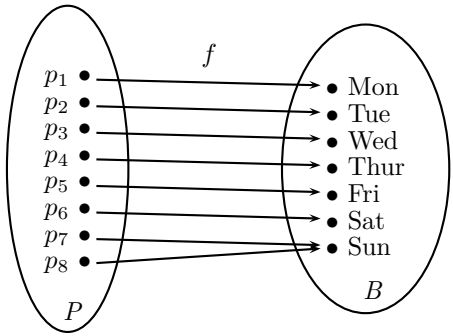
2. Explanations Using Mathematics

There is a trick mathematics lecturers use to surprise large lecture halls. Supposing the hall holds around, say, 400, if the lecture is nearly full, the lecturer can ask the audience: "I wonder if there are two people here with exactly the same birthday?". This causes some puzzlement. Supposing there are at least 367 in the hall and at most 366 days of the year (1 Jan up to 31 Dec) for birthdays, it follows, by what is known as the Pigeonhole Principle, that at least two members of the audience must share a birthday.

The trick works for days-of-the-week someone is born on; in that case, it requires at most eight people, since the size of the set of birthdays is now down to seven: a function $f\colon P \to B$ from a set P of eight people

[1] Still the best survey of many attempted nominalist reformulations of science is [Burgess and Rosen, 1997]. A recent extension of Field's geometric program to field theories in physics is [Arntzenius and Dorr, 2012], where fibre bundles over spacetime are treated as nominalistically acceptable.

to the set B of the seven (birth) days-of-the-week must be non-injective. So, there are at least two distinct people $p, q \in P$ such that $f(p) = f(q)$. The situation (illustrating the Pigeonhole Principle) can be visualized as follows:



It may seem odd that one can make this prediction, which after all is an empirical, contingent one. How could *mathematics alone* — the subject which deals with *abstract, causally inert, timeless, modally invariant objects* — have generated an empirical, contingent prediction? As Feynman noted:

> I find it quite amazing that it is possible to predict what will happen by mathematics, which is simply following rules which really have nothing to do with the original thing. (Feynman, 1965, p. 171)

If we look closely, we see that the prediction "*at least two persons must share a birthday*" is the consequent of a certain conditional:

(C_0) *If*

there are exactly 8 people,
there are exactly 7 days for birthdays,
and each person has exactly one birthday,

then

at least two persons share a birthday.

It is this conditional (C_0) which is implied by the mathematics. In this case, it is a *logical truth* and a *necessity*.[2] I shall call it an "*application conditional*".

[2] Though not all such examples are logical truths.

To try and understand the logical situation, suppose **Math** is some system of applicable mathematics. In particular, suppose **Math** proves various *comprehension theorems*, of the form

$$\exists R \forall x_1 \dots x_n ((x_1, \dots, x_n) \in R \leftrightarrow \phi)$$

including instances where the defining formula ϕ applies to non-mathematical objects, along with some system of axioms governing set operations, properties of sets, their cardinalities, functions between them and, typically, an incorporated theory of number systems, etc. What generally makes application possible, besides the assumption that **Math** is true, is the fact that

$$\mathbf{Math} \vdash (C_0).$$

More generally, **Math** will logically imply certain "application conditionals":

$$\mathbf{Math} \vdash \theta(X_1, \dots) \to \psi(X_1, \dots),$$

where the formulas $\theta(X_1, \dots)$, $\psi(X_1, \dots)$ are contingent/empirical sentences, but which may have some set, relation or function variables free (as well as variables ranging over concrete individuals). Since such an "application conditional" is logically implied by **Math**, it is a necessity and can be read *counterfactually*:

If θ were the case, then ψ would be the case.

Compare this with, for example, the following analysis from a standard classical mechanics textbook:

> In fact, mechanics — and indeed all theoretical science — is a game of mathematical make-believe. We say:
>
> "*If the Earth were a homogeneous rigid ellipsoid acted on by such and such forces, how would it behave*?"
>
> Working out the answer to this mathematical question, we compare our results with observation. If there is agreement, we say we have chosen a good model; if disagreement, then the models or laws assumed are bad. [Synge and Griffith, 1959, p. 5]

Under certain conditions, when no other *mathematical* terms appear in θ or ψ (e.g., no formulas of the form $x \in Y$ occur), the conditional $\theta \to \psi$ is a theorem of *logic* alone (e.g., as above, (C_0) is logically true). In that

scenario, a certain system of restricted set-theoretical comprehension axioms is conservative.[3]

But that is a highly *unusual* situation with respect to the typical applications of mathematics in science. This is because the scientific theory, hypothesis or equation which we add to **Math** (in order to deduce consequences from the law, boundary conditions, etc.: see below) is almost never "*nominalized*". That is, the formulas θ and ψ are not "purely nominalistic" sentences. If one examines the relevant examples of such conditionals, the antecedent and consequent are themselves mathematically formulated or "mixed", as we show below.

For example, if I reason in classical mechanics, based on equations of motion

$$\frac{dp_i}{dt} = -\frac{\partial V}{\partial x^i} \tag{1}$$

I will use position co-ordinates, mass & momentum values/functions, potential functions, etc. To use the technical jargon, the equations involved refer to, and relate, *mixed mathematical entities*: sets, relations and functions: these connect physical objects/systems to abstract values (or "containers" in the case of sets).

Similarly, if I reason in quantum theory, based on the equation of motion

$$i\hbar\frac{\partial \Psi}{\partial t} = \hat{H}\Psi \tag{2}$$

I will refer to some wavefunction Ψ on spacetime, and perhaps to operators like $\hat{H}$ on that space of wavefunctions, and to eigenvalues of such operators, etc. Here, likewise, the equations involved refer to, and relate, *mixed mathematical entities*: sets, relations, functions, wavefunctions, etc. It is these *mixed functions, sets and relations* — call them *mixed quantities* — which are the engine of applicability. The *laws* of various

[3] This is analogous to the well-known result that adding predicative second-order comprehension to Peano arithmetic, yielding a system called ACA_0, is a conservative extension wrt arithmetic sentences [see, e.g., Simpson, 2010]. This kind of conservativeness result is one of the two main parts of Field's programme [Field, 1980] of showing how mathematics can be dispensed with in scientific applications. The other part consists in aiming to show that scientific theories, laws of motion, etc., can be *reformulated nominalistically*, dispensing with mathematical notions and quantification over mathematical entities. In [Field, 1980], this approach focuses on classical mechanics and gravitation. Similar ideas are introduced and extended in [Arntzenius and Dorr, 2012], looking at the geometric (fibre-bundle) formulation of field theories.

specific sciences are (typically) equations, about such mixed quantities, such as (1) and (2).

That said, partial differential equations, such as (1) and (2), with quantities taking highly abstract value ranges (and perhaps with various linear operators appearing too, etc.) require quite advanced methods to understand. And needlessly so in relation to the point being made here about the *underlying logical structure of applied mathematical reasoning, involving such laws and predictions*. In general, such reasoning takes the following schematic form:

$$\textbf{Math} \vdash \left[\overbrace{\mathsf{Law}(X_1, \dots)}^{\text{law/differential equation}} \quad \wedge \quad \overbrace{\mathsf{BC}(X_1, \dots)}^{\text{boundary condition}} \right] \rightarrow \overbrace{\mathsf{Sol}(X_1, \dots)}^{\text{solution}}.$$

Informally, the applicable mathematics *implies* a conclusion to the effect that:

> If we have quantities $X_1, \dots, X_n$, satisfying some law $\mathsf{Law}(X_1, \dots)$, and boundary conditions $\mathsf{BC}(X_1, \dots)$, then the solution (or prediction) is given by: $\mathsf{Sol}(X_1, \dots)$.

I will briefly sketch a fairly simple example. In university level mathematics and physics, one learns how to solve various differential equations, ranging from very simple ones at the outset, to highly complicated ones in advanced courses: such as many-particle Schrödinger equations, the Navier-Stokes equation, Einstein's field equations, or Yang-Mills field equations, or perhaps more esoteric differential equations that pop up in chaos theory. But here is a simpler example. This would be an exponential growth/decay law:

$$\frac{dF}{dt}(t) = kF(t)$$

subject to a boundary condition, such as,

$$F(0) = p$$

In this case, one can figure out that the solution is

$$F(t) = pe^{kt}$$

In effect, one has a *mathematical theorem*:[4]

$$\left(\forall t \in T(\frac{dF}{dt}(t) = kF(t)) \wedge F(0) = p\right) \rightarrow \forall t \in T(F(t) = pe^{kt}).$$

This is another example of an *application conditional*. Indeed, it is of the kind that is ubiquitous in theoretical physics, engineering, chemistry, etc. And, typically, *applied mathematical reasoning is, as we have noted above, the deduction of such application conditionals from some (implicit) system of axioms for applicable mathematics.*[5]

The current example looks like this:

$$\textbf{Math} \vdash \left[\overbrace{\forall t \in T(\frac{dF}{dt}(t) = kF(t))}^{\text{differential equation}} \wedge \overbrace{F(0) = p}^{\text{boundary condition}}\right] \rightarrow \overbrace{\forall t \in T(F(t) = pe^{kt}))}^{\text{solution}}.$$

Schematizing even further:

$$\textbf{Math} \vdash \overbrace{\theta(F, \dots)}^{\text{assumption about } F, \dots} \rightarrow \overbrace{\psi(F, \dots)}^{\text{conclusion about } F, \dots} .$$

This kind of schema, I claim, will account for a high proportion of cases of reasoning in applied mathematics.

3. A Computational Prediction

Below, we discuss the logical situation in some more detail, but first I would like to give another example of mathematical explanation in science. It is relatively easy to explain and also highlights the logical structure of the underlying reasoning involved. It involves the closed form expression for any finite geometric series: $1 + x + x^2 + \cdots + x^N$.

Suppose you and your friend have the R programming language on your laptops and you are going to take some lectures by me on, say,

[4] This oversimplifies a bit, by omitting some implicit clauses. For example, a clause expressing that F is a real-valued differentiable function on T.

[5] I do not claim that every example of mathematical reasoning in applied mathematics is the deduction of a *solution* from a *law* constrained by a *boundary condition*. Even so, it is fairly typical.

regression models. My unpopularity is so high that there are only two students: you and your friend. I ask you to type the following code into R, defining a function f:

Program P_f

```
f <- function(x, N) {
    S <- rep(NA,N+1)
    S[1] <- x^0
    for(i in 1:N){
        S[i+1] <- S[i]+x^{i}
        }
    S[N+1]
}
```

Given input arguments x, N, this short program first initializes S to be a vector/sequence $S[1], \ldots, S[N+1]$, of length $N+1$, with each component set as "NA" ("not available"). It next sets the first component $S[1]$ of S to be x^0. It then loops through the remaining range of indices on the vector S, from 1 to N, by defining $S[i+1]$ to be $S[i] + x^i$. Finally, it outputs the final component $S[N+1]$. So, the program P_f computes $1 + x + x^2 + \cdots + x^N$.[6]

I ask your friend to type the following code, defining a function g:

Program P_g

```
g <- function(x, N) {
    (1 - x^{N+1})/(1 - x)
}
```

In conventional mathematical terminology, the programs P_f and P_g compute the functions f, g, where:[7]

[6] R code like this is sometimes called a "script", which can be "run" without separate compiling. Languages like Python, Perl and many others are similar.

[7] As aficionados will recognise but philosophers may be interested to know, these function definitions are lambda-abstractions, which go back to the ideas of Frege [1891] and Russell [1903-05] concerning the syntactic and semantic analysis of function terms, and to Church's early papers [Church, 1932, 1933] on lambda-calculus (the system given there in fact turned out to be inconsistent, as shown in [Kleene and Rosser, 1935]. See [Klement, 2003] for a detailed discussion of those historical developments.

$$f(x,N) := \sum_{i=0}^{N} x^i,$$
$$g(x,N) := \frac{1-x^{N+1}}{1-x}.$$

I then ask you to compute $f(0.5, 10)$ and I ask your friend to compute $g(0.5, 10)$. What ought we predict about these computations?

Prediction: You both get the same answer!

But why is this so?[8] The answer is:

(i) your laptop runs a program P_f computing the function f;
(ii) your friend's laptop runs a program P_g computing the function g;
(iii) you are both operating the inputs and outputs correctly;
(iv) and, for all $x \in \mathbb{R}$, for all $N \in \mathbb{N}$, $f(x,N) = g(x,N)$.

The first three parts (i)–(iii) of this explanans are *contingent*. Maybe you mistype the function definition, and at run-time, get an error message. Maybe you typed it correctly, but your laptop's battery might give up a microsecond after you press "run". Likewise, may you enter the inputs incorrectly. To make the prediction, we must suppose none of this happened, and suppose that (i)–(iii) are true.

The fourth part, (iv), however, is a mathematical necessity. We know from algebra that these computations always yield equal results. The algebra is fairly straightforward. We wish to work out the sum $\sum_{i=0}^{N} x^i$ and show it is equal to $\frac{1-x^{N+1}}{1-x}$.

Since the 1960s, lambda-abstraction is now an integral part of modern programming languages. The code

$$\texttt{g <- function(x, N) \{ (1 - x}^{\texttt{N+1}}\texttt{)/(1 - x) \}}$$

says: let g be the binary function, with arguments x, N, whose value, given x, N, is $\frac{1-x^{N+1}}{1-x}$.

To illustrate a lambda-abstraction *inside* another lambda abstraction, there is a shorter "vectorized" program definition of the function f:

$$\texttt{f <- function(x, N) \{ sum(sapply(0 : N, FUN = function(n)\{x}^{\texttt{n}}\texttt{\})) \}}$$

This uses $\texttt{sapply}(., \texttt{FUN} = .)$ to *apply* a defined function to a sequence, $\texttt{0 : N}$. However, this program is less intuitive than the one given above. In fact, it runs about 5 times slower too.

[8] The answer, by the way, is: $f(0.5, 10) = 1.999023 = g(0.5, 10)$.

First, we have an obvious recurrence relation:

$$\sum_{i=0}^{N+1} x^i = \sum_{i=0}^{N} x^i + x^{N+1}. \tag{3}$$

Second, factorizing gives:

$$\sum_{i=0}^{N+1} x^i = 1 + x + x^2 + \cdots + x^{N+1} = 1 + x(1 + x + \cdots + x^N),$$

Thus, it follows that a second recurrence relation holds:

$$\sum_{i=0}^{N+1} x^i = 1 + x \sum_{i=0}^{N} x^i. \tag{4}$$

Equating the right-hand-sides of (3) and (4),

$$\sum_{i=0}^{N} x^i + x^{N+1} = 1 + x \sum_{i=0}^{N} x^i.$$

So, rearranging,

$$(1 - x) \sum_{i=0}^{N} x^i = 1 - x^{N+1}.$$

This implies, so long as $x \neq 1$:

$$\sum_{i=0}^{N} x^i = \frac{1 - x^{N+1}}{1 - x}.$$

Thus, the functions f, g defined by the given code are coextensive.[9]

Putting this all together, the mathematical statement:

(E_0) For all $x \in \mathbb{R}$, and all natural numbers N,

$$1 + x + x^2 + \cdots + x^N = \frac{1 - x^{N+1}}{1 - x}.$$

[9] There is a caveat that attempting to compute g will yield an error message when $x = 1$. That is, when a program call `g(1,N)` is made, it will throw an error message. This is because the ratio of functions defining g is indeterminate at $x = 1$ (i.e., $0/0$). In fact, it converges. There is no difficulty of course in working out $f(1, N)$ (i.e., $\sum_{i=0}^{N} x^i$ when $x = 1$). It is $N + 1$. So, $f(1, N) = N + 1$. For the ratio of functions defining g, the same value may also be obtained using L'Hôspital's Rule, by differentiating top and bottom expressions. This gives $\frac{-(N+1)x^N}{-1}$, whose limit at $x = 1$ is $N + 1$.

explains, given the definitions of f and g, why:

(E_1) For all $x \in \mathbb{R}, N \in \mathbb{N} : f(x, N) = g(x, N)$.

And this explains, given the auxiliary assumptions that the program P_f computes f and the program P_g computes g, why:

(E_2) P_f and P_g are computationally equivalent.

Finally, this explains why:

(C_1) For any machines m_1, m_2, *if* m_1, m_2 run the programs P_f, P_g and m_1 and m_2 receive the same inputs, *then* m_1 and m_2 yield the same outputs.

The final claim (C_1) is the *application conditional* involved, and its consequent, "m_1 and m_2 yield the same outputs" is the relevant *prediction*.

4. Logical Structure

We can now sketch the overall structure of this computational explanation:

A computational explanation

(E_0) $(\forall N \in \mathbb{N})(\forall x \in \mathbb{R}) \left[\sum_{i=0}^{N} x^i = \frac{1-x^{N-1}}{1-x}\right]$

explains ↓ (given the definitions of f and g)

(E_1) $(\forall N \in \mathbb{N})(\forall x \in \mathbb{R})(f(x, N) = g(x, N))$

explains ↓ (given that P_f computes f and P_g computes g)

(E_2) P_f is computationally equivalent to P_g.

explains ↓ (given the definition of "m runs P")

(C_1) If m_1, m_2 run P_f, P_g, then same inputs to m_1, m_2 give same outputs.

The following (which simplifies the reasoning a little) is a deductively valid argument showing how the explanation from (E_2) to (C_1) works:[10]

(Df_1)	Machine m runs program P iff, m performs in sequence each instruction of P and given an input representing n, and outputs the return value of P on n.
(Df_2)	P_1 and P_2 are computationally equivalent programs iff, for any input n, the value of P_1 on n = the value of P_2 on n.
(1)	P_1 and P_2 are computationally equivalent programs.
(2)	Machine m_1 runs program P_1 and m_2 run programs P_2.
(3)	m_1 and m_2 receive same input (z, say).
(4)	m_1 and m_2 yield the same output (w, say).

The proximate explanation of (C_1) is the fact that programs P_f and P_g are computationally equivalent. But *this is a mathematical fact about abstract programs.* And this fact is explained by the *mathematical fact* (E_1) that the functions computed by programs P_f and P_g are co-extensive. And the co-extensiveness of these functions is explained by the *mathematical fact* (E_0).

Finally, suppose we absorb the definitions and the explanans "inside" **Math**, as it were.[11] The result is an instance of the application conditional schema described above:

[10] The formalization of this argument is:

(Df_1)	$\forall x \forall p\ (\mathrm{run}(x,p) \leftrightarrow (\mathrm{perf}(x,p)) \wedge \forall y \forall z((\mathrm{inp}(x,y) \wedge \mathrm{val}(p,y) = z) \rightarrow \mathrm{out}(x,z)))$.
(Df_2)	$\forall p_1 \forall p_2(\mathrm{equiv}(p_1,p_2) \leftrightarrow \forall z(\mathrm{val}(p_1,z) = \mathrm{val}(p_2,z)))$.
(1)	$\mathrm{equiv}(p_1,p_2)$.
(2)	$\mathrm{run}(a,p_1) \wedge \mathrm{run}(b,p_2)$.
(3)	$\exists z(\mathrm{inp}(a,z) \wedge \mathrm{inp}(b,z)$.
(4)	$\exists w(\mathrm{out}(a,w) \wedge \mathrm{out}(B,w))$.

This can be shown to be valid. And (C_1) is, in effect, $(2) \wedge (3) \rightarrow (4)$.

[11] I.e., the explanans (E_2) (that P_f and P_g are computationally equivalent, along with the various mathematical results used to prove that) is now hidden inside **Math**. Clearly, as we have shown, **Math** implies that P_f and P_g are computationally equivalent.

$$\textbf{Math} \vdash \Big[\overbrace{(m_1 \text{ runs } P_f \text{ and } m_2 \text{ runs } P_g)}^{\text{“law”}} \wedge \overbrace{(m_1 \text{ and } m_2 \text{ receive same input})}^{\text{initial condition}}\Big] \rightarrow \overbrace{(m_1 \text{ and } m_2 \text{ yield same output})}^{\text{prediction}}.$$

As soon as one sees the basic explanatory pattern used here, it is easy to give similar computational examples.

For example, there are plenty of computable functions f, g, which are *defined differently*, but which are such that one can prove that $f(n) = g(n)$, for all n. A nice example would be programs P_1 and P_2 which, given input n, print the first n decimal digits of π, or perhaps print the nth decimal digit of π. Typically, the equivalence of programs P_1 and P_2 follows from a mathematical theorem:

$$\forall n \in \mathbb{N} : P_1(n) = P_2(n).$$

As programmers are well aware, it is routine for the properties of some specified program P (i.e., a piece of abstract syntax in some fixed programming language) to be analysed *mathematically*. Often one is interested estimating the running time of P, measured as a function of the size of the input. One of the most important open problems in mathematics concerns whether there exists a program P, which computes the satisfiability of given Boolean formulas in CNF in polynomial time.

5. Is There a “Nominalistic” Alternative?

The application conditional that we explained mathematically is the following:

(C_1) If machines m_1, m_2 run programs P_f, P_g, then same inputs to m_1, m_2 give same outputs.

As is typical in such cases, this is a *mixed* statement. It is specifically about two *abstract entities* — namely, the *programs* P_f and P_g. Thus, (C_1) is, at best, vacuously true for a nominalist, as the nominalist view is that only *tokens* (such as physical inscriptions, utterances, or perhaps specific concrete encodings in digitized patterns in computer memory or

on screen) exist. The nominalist may of course talk of tokens of these programs: let us say they are t_f and t_g, perhaps defined by ostension. Then, the corresponding "nominalization" of (C_1) would be:

(C_1^{nom}) If machines m_1, m_2 run *tokens* t_f, t_g, then same inputs to m_1, m_2 give same outputs.

And the modification of the proximate explanation (E_2) is:

(E_2^{nom}) Token t_f is computationally equivalent to token t_g.

It seems to me that one *can* reasonably argue that (E_2^{nom}) explains (C_1^{nom}).[12]

But this is not the problem. For consider what explanations might be given for (E_2^{nom}) itself. The mathematician or computer scientist may *explain* (E_2^{nom}) as follows. First, t_f is a token of P_f and t_g is token of P_g. Second, the program P_f is computationally equivalent to the program P_g, *indeed because the functions f, g are coextensive.* Moreover, if programs are computationally equivalent, then so are their tokens. Ergo, token t_f is computationally equivalent to token t_g. Q.E.D.[13]

However, this explanation is not nominalistically admissible since it refers to abstract entities: programs. Is there a nominalistic explanation of (E_2^{nom})? This might somehow proceed via a discussion of microchips, wires, atoms, etc., but nowhere advert to abstract entities such as functions, programs or numbers? Or, for that matter, lengths, times, masses, charges, electric & magnetic field strengths, etc.? This appears to me to be doubtful.[14] The explanation of the (necessarily) true application conditional (C_1), given above, is (E_2). The parallel "nominalistic" explanation of (C_1^{nom}) would be (E_2^{nom}). However, it seems that (E_2^{nom}) is *nominalistically inexplicable.*

So, I end with a challenge:

[12] Note that (E_2^{nom}) and (C_1^{nom}) are now contingent statements. In nearby worlds, where the physical inscriptions t_f or t_g are a teeny bit different, they do not in fact compute the right functions. In fact, nearby tokens of either will not even be syntactically well-formed and so will not run (an error with be thrown).

[13] Admittedly, "t is a concrete token of syntactical entity P" is a notion which has not received sufficient attention. The only work I know of which has attempted a sustained analysis of such notions is [Wetzel, 2009].

[14] The only manageable approach I can picture would be to try to "nominalize" the description of the physical machine which "runs" a given *token*, and then somehow show that various non-obviously-equivalent "dynamical trajectories" of the physical machine's states sometimes take equal input tokens to equal output tokens.

Challenge: provide a *nominalistic explanation* of the fact that token t_f is computationally equivalent to token t_g.

Acknowledgements. The author acknowledges support by a grant from the National Science Centre in Cracow (NCN), Poland, grant number 2018/29/B/HS1/01832.

References

Arntzenius, Frank, and Cian Dorr, 2012, "Calculus as geometry", Chapter 8 of F. Arntzenius (ed.), *Space, Time, and Stuff*, Oxford: Oxford University Press. DOI: 10.1093/acprof:oso/9780199696604.001.0001

Baker, Alan, 2005, "Are there genuine mathematical explanations of physical phenomena?", *Mind* 114: 223–238. DOI: 10.1093/mind/fzi223

Burgess, J. P., and Gideon Rosen, 1997, *A Subject with No Object. Strategies for Nominalistic Interpretation of Mathematics*, Oxford: Clarendon Press. DOI: 10.1093/0198250126.001.0001

Church, Alonzo, 1932, "A set of postulates for the foundations of logic", *Annals of Mathematics* 33: 346–366.

Church, Alonzo, 1933, "A set of postulates for the foundations of logic" (second paper), *Annals of Mathematics* 34: 839–864.

Church, Alonzo, 1958, "Letter to Goodman (1 Dec 1958)". Reprinted in [Goodman 1972].

Field, Hartry, 1980, *Science Without Numbers: A Defence of Nominalism*, Princeton, N.J.: Princeton University Press.

Frege, Gottlob, 1891, "Function and concept". English translation of [Frege, 1891] "Funktion und Begriff" (*der Jenaischen Gesellschaft für Medizin und Naturwissenschaft*) in P. Geach and M. Black (eds.), *Translations from the Philosophical Writings of Gottlob Frege*, Oxford: Blackwell, 1980.

Feynman, Richard, 1965, *The Character of the Physical Law*, London: Penguin Books, 1992. Page reference to the 1992 edition. DOI: 10.7551/mitpress/11068.001.0001

Goodman, Nelson, 1972, *Problems and Projects*, New York: The Bobbs-Merrill Company Inc.

Kleene, S. C., and J. B. Rosser, 1935, "The inconsistency of certain formal logics", *Annals of Mathematics* 36: 630–637. DOI: 10.2307/1968646

Klement, Kevin, 2003, "Russell's 1903–1905 anticipation of the lambda calculus", *History and Philosophy of Logic* 24 (1): 15–37. DOI: 10.1080/0144534031000076237

Mancosu, Paolo, 2018, "Explanation in mathematics", Stanford Encyclopedia of Philosophy. URL: https://plato.stanford.edu/entries/mathematics\protect\unhbox\voidb@x\hbox-explanation/.

Putnam, Hilary, 1971, *Philosophy of Logic*. Reprinted in [Putnam 1979]. Page reference to [Putnam, 1979].

Putnam, Hilary, 1975, "What is mathematical truth?", in [Putnam, 1979]. Page reference to [Putnam, 1979].

Putnam, Hilary, 1979, *Mathematics, Matter and Method. Philosophical Papers*, Vol. 1, Cambridge University Press, Cambridge. Second edition. DOI: 10.1017/CBO9780511625268.022

Quine, W. V., 1948, "On what there is", *Revue of Metaphysics* 2: 21–38.

Quine, W. V., 1973, "The limits of knowledge", Radio Talk. Published in [Quine, 1976], *The Ways of Paradox*. Revised enlarged second edition. Harvard University Press, Cambridge, Ma.

Quine, W. V., 1995, *From Stimulus to Science*, Harvard University Press, Cambridge, Ma.

Quine, W. V., and Nelson Goodman, 1947, "Steps towards a constructive nominalism", *Journal of Symbolic Logic* 12: 105–122. Reprinted in [Goodman, 1972].

Russell, Bertrand, 1903–05, *The Collected Papers of Bertrand Russell, Vol. 4, Foundations of Logic 1903–05*, edited by Alasdair Urquhart, London: Routledge, 1994.

Simpson, S., 2010, *Subsystems of Second-Order Arithmetic*, 2nd edition, Cambridge: Cambridge University Press. DOI: 10.1017/CBO9780511581007

Synge, J. L., and B. A. Griffith, 1959, *Principles of Mechanics*, Third edition, McGraw-Hill Book Company Inc., New York.

Wetzel, L., 2009, *Types and Tokens: On Abstract Objects*, Cambridge, Ma.: MIT Press. DOI: mitpress/9780262013017.001.0001

JEFFREY KETLAND
Institute of Philosophy
University of Warsaw, Poland
jeffreyketland@gmail.com