Mapping programs to the brain

Joey Velez-Ginorio. Department of Brain & Cognitive Sciences, MIT

Research Question. Programs describe computations. Properly construed, they describe human behavior too. An outpour of work in cognitive science shows this. Yet it's not clear how these programs manifest in brains. This is odd. Because human behavior *is* brain behavior.

I want to show what programs could look like in brains. To do so, I need a mapping from programs to the mathematics familiar to neuroscience. The form $W\hat{x} \mapsto \hat{y}$ from linear algebra describes an input vector $\hat{x}$, a matrix W which transforms the input, and the resultant output vector $\hat{y}$. It's a general way to describe linear transformations. Brain behaviors are often construed this way. To map programs in this form would show how they could manifest in brains, and offer an alternative way to describe and predict brain behaviors.

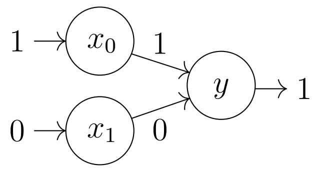
Courtesy of recent programming language research in *denotational semantics*, the mapping from programs to linear algebra has already been made. The aims of this project are to cash out that work in the context of cognitive science—to show what programs could look like in brains.

Background. The *lambda calculus* is a model of computation, akin to Turing machines. In programming language research, lambda calculi amount to formal specifications of a programming language. I elide the minutiae of their presentation, but consider the following example: $(\lambda x.x)t \mapsto t$.

We have a program $(\lambda x.x)$ which computes the identity function $f(x) = x$. And we apply it to an argument t, which reduces (or equivalently, computes) t. The example illustrates what programs look like in the lambda calculus, and how they behave.

As formal specifications, lambda calculi offer rigorous ways to interpret programs in different mathematics. These mappings are known as denotational semantics. They give programs meaning (semantics) by describing what they denote (their denotation) in different mathematics.

In my project I use a denotational semantics which maps programs to vector spaces, the objects of linear algebra. As a result, programs map to the form $W\hat{x} \mapsto \hat{y}$ familiar to neuroscience. Again, avoiding minutiae but to convey intuition, consider these interpretations of the earlier program.



(a) Circuit

$$\begin{bmatrix}1 & 0\end{bmatrix} \begin{bmatrix}1 \\ 0\end{bmatrix} \mapsto \begin{bmatrix}1\end{bmatrix}$$

(b) Equation

$$(\lambda x.x)t \mapsto t$$

(c) Program

Figure (a) is a circuit of 3 neurons, connected as shown. Figure (b) is the equational form of that circuit, as $W\hat{x} \mapsto \hat{y}$. And with denotational semantics, Figure (c) is the programmatic form of that circuit. Crucially, the program offers a new lens into the circuit. It does not merely rehash the numbers in new notation, as the equational form does. It does not even use numbers. In this way, it's a higher-level descriptor of the circuit. It tells us that the circuit computes the identity function.

PLANNED WORK. Because denotational semantics are formal, I can prove that a similar correspondence between circuit, equation, and program exists for any neural circuitry described by the form $W\hat{x} \mapsto \hat{y}$ over finite vector spaces. This is a key theoretical result.

To make this practical, I will implement a programming language whose equational and circuit denotations are computable. For any program you write, you can see what it would look like in neural circuitry. Because a handful of similar denotational semantics exist, I will explore each in the same way—determining how they alter our interpretation of neural circuitry. These efforts establish both the theory and practice of mapping programs to brains.

These are highly interdisciplinary efforts. Thankfully, my PhD program offers me the bureaucratic freedom to research beyond disciplinary and institutional bounds. With the Ford foundation's support, I hope too for financial freedom. Together these freedoms enable my work with Nada Amin and Joshua Tenenbaum, who jointly offer expertise in programming languages and computational cognitive science. They will help me lay these foundations for a new kind of brain science, one I will pursue for life.