Uncovering neural mechanisms of mental simulation by symbolically programming RNNs.

Summary: How is it that through the distributed and dynamic activity in our brain's neural circuits, we think thoughts about objects, mentally simulate how they will move and react to forces, and plan actions? We present a multi-level modeling framework that natively inter-operates in both cognitive hypotheses (simulatable object representations) and neural mechanisms (distributed codes and dynamic attractors). Using this new framework, we reverse-engineer a "mental simulation circuit" in prefrontal cortex populations of macaques playing the video game pong.

In computational neuroscience, deep neural networks (DNNs) have made strides by providing models mappable to neural data. DNNs offer scientific hypotheses with respect to a model's architecture, training data, and training objective, while obscuring the algorithms and representations at play. We introduce a different approach that directly encodes explicit algorithmic and representational hypotheses into the weights and dynamics of recurrent neural networks (RNNs) and test them against neural data.

We do so by extending a recent approach in the physics of dynamical systems that allows us to specify a cognitive hypothesis—the approximate dynamics of the game of pong, in terms of objects and surfaces— and *program* a biologically plausible RNN to encode these dynamics and perform goal-directed control.

We compare prefrontal cortex activity recorded in a pair of macaques to the dynamics of these RNNs under the same set of pong trials. We find that the hidden states of programmed RNNs explain substantial variance in neural data. Beyond a quantitative alignment, programmed RNNs recapitulate a key non-linearity in prefrontal populations that machine-learning style trained RNNs don't. These results provide support for the multi-level hypothesis of an object-based mental simulation system implemented in the neural machinery of the prefrontal cortex.

This multi-level modeling framework holds significant promise for computational neuroscience, offering new ways to reverse-engineer fundamental neural mechanisms underpinning mental life.

Detail: We hypothesize that monkeys playing pong utilize an internal model for game dynamics, such as ball movement and wall collisions, to inform paddle control (Fig. 1a, b). Utilizing ref. [1]'s method, we embed this hypothesis in a biologically plausible RNN, which our results suggest captures prefrontal neural activity more effectively than alternative models (Fig. 1c, d).

Our approach deconstructs the RNN's governing equation through series expansion around an initial state (Fig. 2a, b), enabling the hidden RNN state to symbolically represent network inputs. This permits target functions, including those commonly found in physical simulation, to be directly encoded into the read-out weights of RNNs through least-squares without training on input-output pairs (Fig. 2d; shown for a NOR logic gate). When these outputs are fed back into the RNN, via a mapping from output to input, the programmed read-out weights integrate with the read-in matrix, compiling our functions into the hidden units' connectivity and dynamics (Fig. 2e-left).

Despite the approximation inherent in decomposition, the RNN's tanh activation allows it to sustain non-linear dynamics, vital for tasks such as collision detection. We demonstrate this with a pitchfork bifurcation, which underlies discrete state transitions needed for collision resolution in physical simulation, within the RNN (Fig. 2c, 2e-middle with the NOR gate). This composes to a hysteretic circuit that reverses ball velocity at collision (Fig. 2e-right). Finally, we simulate gameplay by projecting the paddle's position from the RNN's hidden state and feed this prediction back into the network at each time step (Fig. 1d).

The model is evaluated against neural recordings from the dorsomedial frontal cortex of monkeys during pong, across 79 conditions of ball positions and velocities (Fig 1a). By calculating the correlation of pairwise distances following ref. [2], we find our programmed RNNs (n=100) explain a substantial portion of neural dynamics, surpassing control RNNs and machine learning (ML) style, traditionally trained models (Fig 3a). Moreover, the RNN's game proficiency is directly proportional to its neural data similarity (Fig. 3b). Critically, programmed RNNs show the key non-linear decodability of the ball's final position early on in a trial's progression, as seen in the neural data and not in traditionally trained RNNs, although the programmed RNN lags behind the actual neural timing (Fig 3c).

Figure 1: Experimental setup and computational model of pong dynamics. Using a model-based policy to support gameplay, the programmed RNNs show high correlation with DMFC activity.

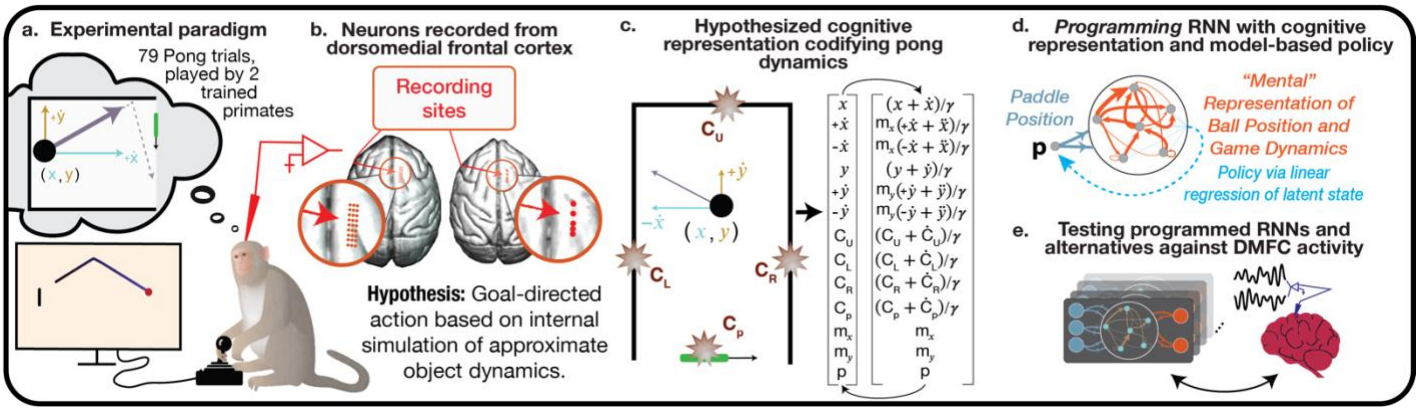


Figure 2: RNN Architecture and Pipeline for programming a neural Pong engine

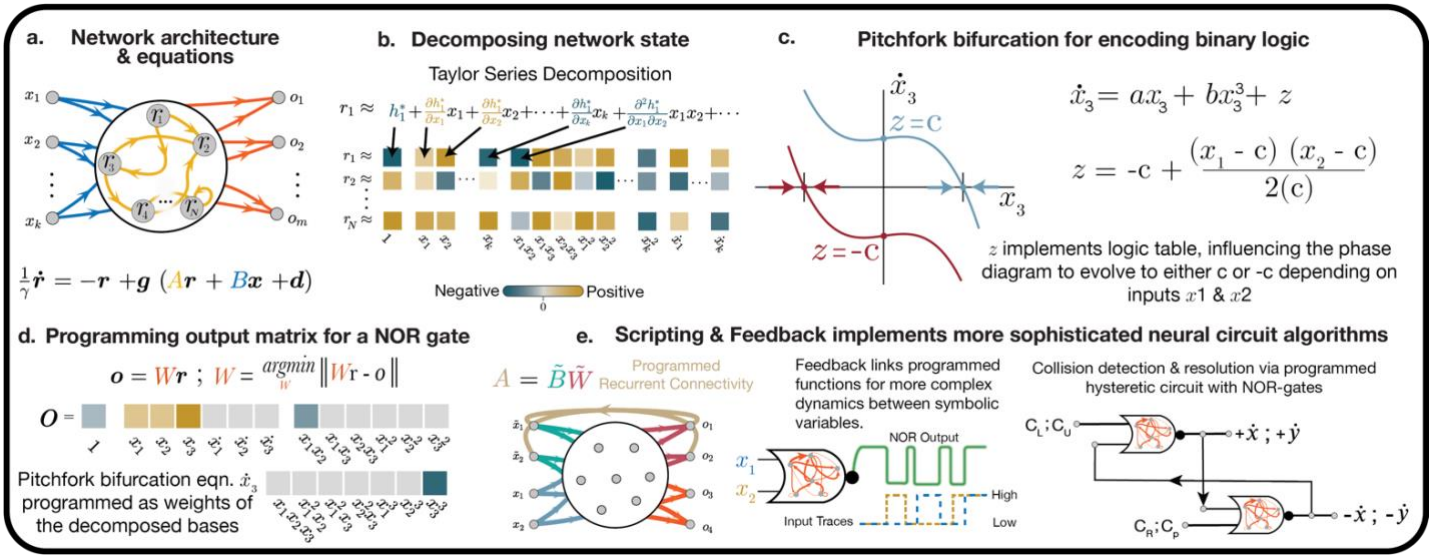
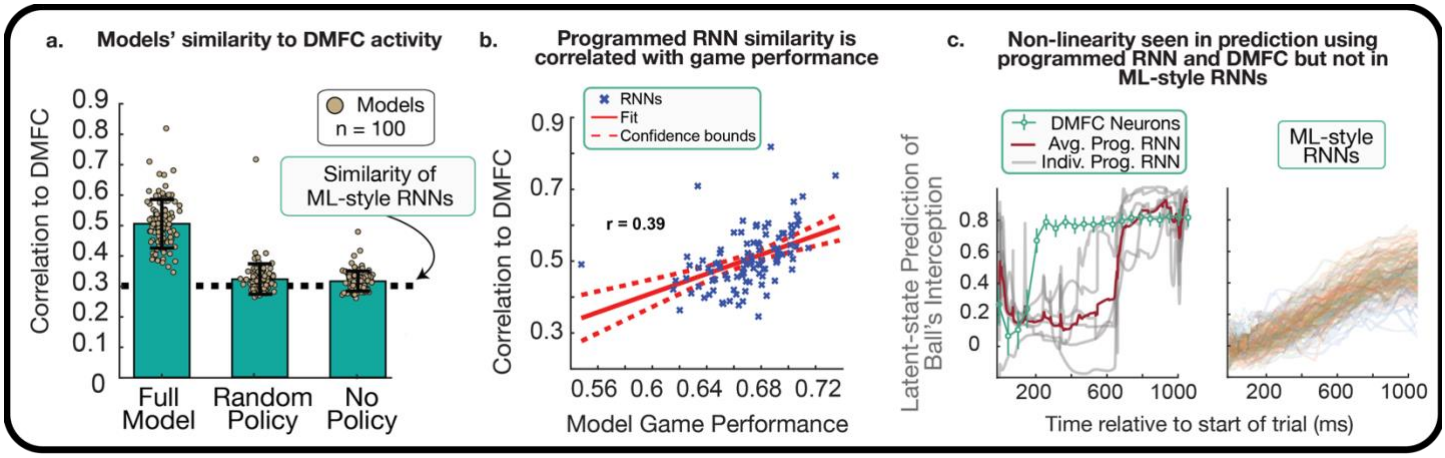


Figure 3: Programmed RNNs show performant gameplay and recover key patterns in neural dynamics.



References: [1] Kim, J. Z., & Bassett, D. S. (2023). A neural machine code and programming language for the reservoir computer. Nature Machine Intelligence.
[2] Rajalingham, R., Sohn, H., & Jazayeri, M. (2022b). Dynamic tracking of objects in the macaque dorsomedial frontal cortex. bioRxiv, 2022–06.