Dynamic Modelling of Memory Interactions and Behavioural Prediction

Python Simulation: Differential Equations & OOP Design

Presented by:- Sagnik Barman, Pranay Saha, Lucas Haobam, Debadatta Panda

Department of Mathematics, IIT Madras

Introduction and Objectives

1

Model Memory Dynamics

Simulate memory interaction and evolution using coupled ODEs to capture cognitive interdependencies.

2

Implement with Python OOP

Develop a modular, OOP-based architecture for flexible, scalable, and maintainable memory simulations.

3

Predict Behavioural Outcomes

Infer approach/avoidance behaviour by analyzing dominant memory states, linking neurodynamics to observable actions.

4

Generalize to Complex Systems

Extend the framework from two-memory to multi-memory networks for realistic behavioural decision-making.

Mathematical Model

Memory Mi(t) evolves based on coupled linear differential equations:

$$dM_i/dt = \alpha_i M_i + \Sigma_j \beta_{ij} M_j + b_i$$

Self-Dynamics (αi)

Intrinsic decay or growth rate of memory Mi. Determines how Mi changes in isolation.

Cross-Memory Coupling (βij)

Influence of memory Mj on memory Mi. Captures excitatory or inhibitory interactions between memories.

External Bias (bi)

External stimuli or factors affecting memory Mi. Represents constant environmental or internal inputs.

The system of ODEs is numerically integrated using `scipy.integrate.odeint` in Python.

Methodology

Model Design

Establish abstract base class for simulation interfaces and a concrete LinearMemorySystem subclass for the coupled ODE framework.

Numerical Simulation

Solve coupled ODE system using user-specified parameters (α, β, b) and initial conditions over a defined temporal domain.

Visualization

Generate time-series plots for memory evolution and phase portraits to reveal dynamical structure and attractor behaviour.

Behavioural Prediction

Extract dominant memory at simulation end to infer behavioural tendencies (approach/avoidance), offering cognitive - behavioural insights.

Implementation and Results

Core Architecture

SimulationInterface: Abstract Base Class for flexible simulation definition.

LinearMemorySystem: Concrete subclass implementing the coupled ODE model.

MemoryState: Data structure for tracking memory values over time.

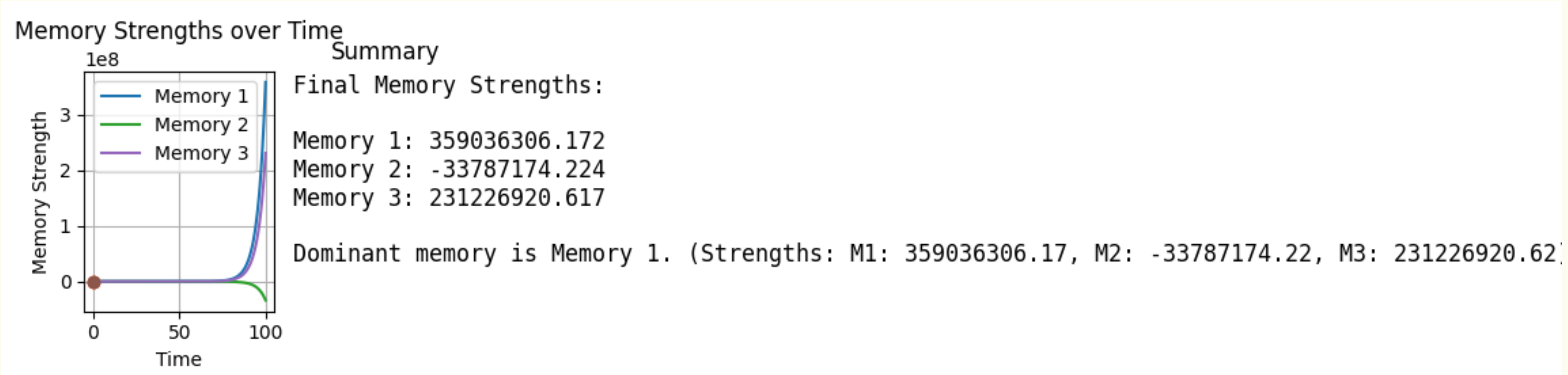
ParameterManager: Utility for loading and managing α, β, and b parameters.

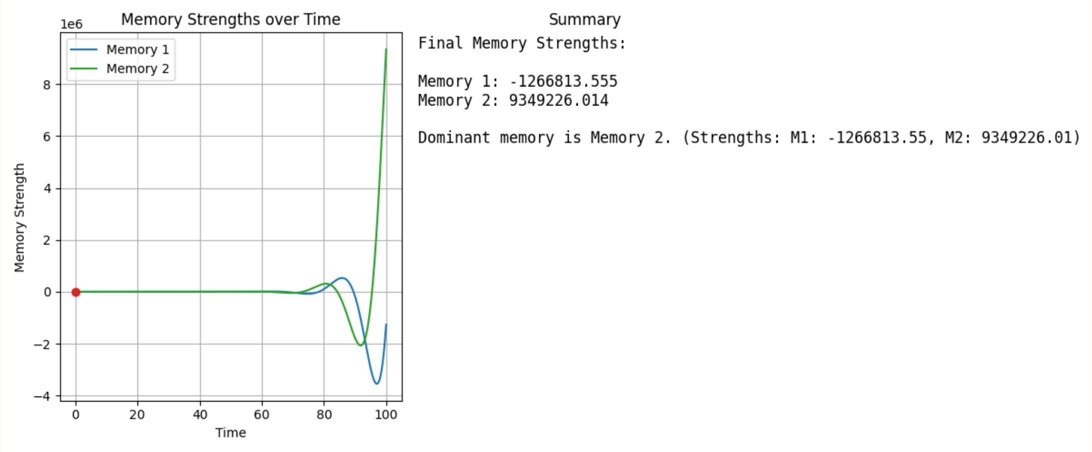
Python OOP: Ensures modular, scalable, and maintainable code.

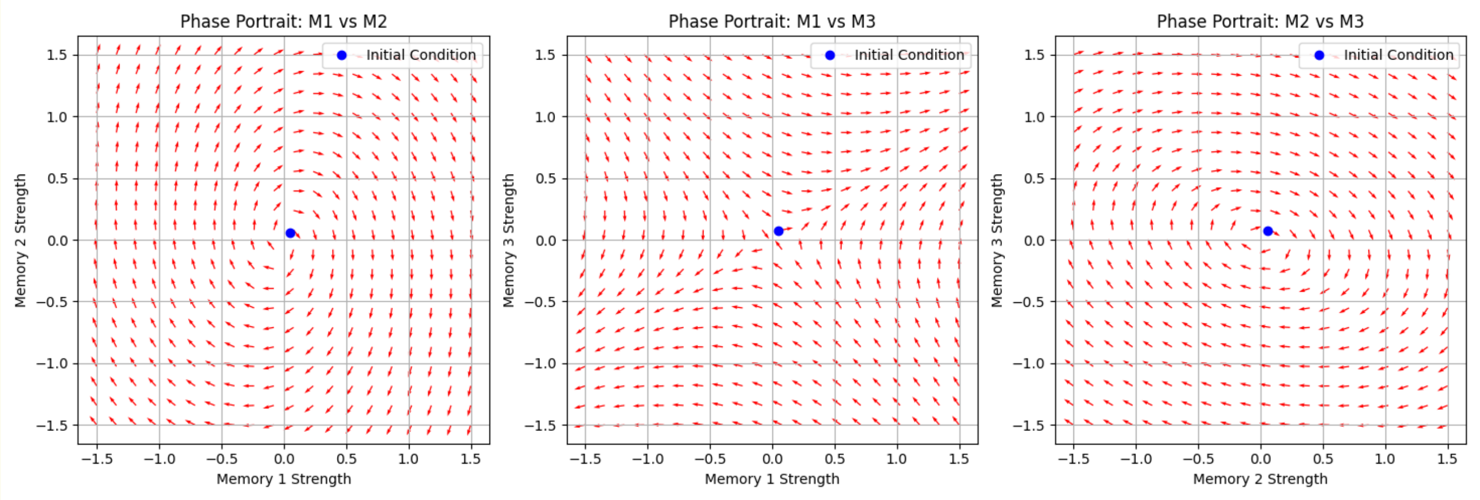
Key Results

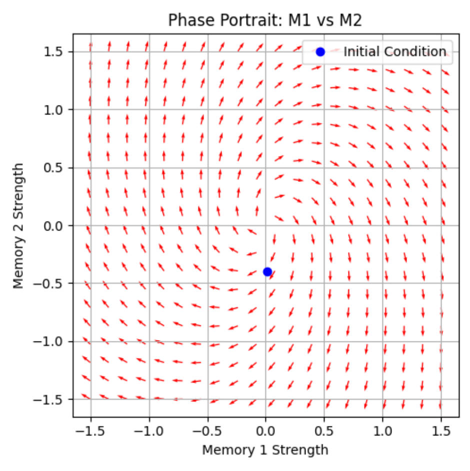
- Successfully simulated diverse memory dynamics, including oscillatory patterns.
- Demonstrated prediction of approach/avoidance behaviours based on dominant memory states.
- Validated model stability and sensitivity to parameter variations.
- Confirmed framework's extensibility to multi-memory systems for complex decision-making.
- Generated rich visualizations aiding in dynamic behaviour analysis.

Results and Visual Outputs









Key Observations:

- Time-series plots (3D and 2D) confirm convergence to stable memory states.
- Phase portraits (3D and 2D) distinctly reveal heteroclitic connections between attractors.
- Coupling strength is quantitatively shown to modulate memory transition dynamics.

Conclusion

Successful Modelling

Developed differential equation framework for memory interactions, linking neurobiology to behaviour.

Computational Rigor

OOP Python implementation offers modularity and extensibility, facilitating collaboration and future enhancements.

Theory-Computation Bridge

Model translates cognitive theory into concrete computational predictions, advancing interdisciplinary neuroscience.

Future Directions

Integrate non-linear memory dynamics, Hebbian learning, contextual modulation, and validate against empirical data.