How ABMs Evolve into ODEs/PDEs for System Dynamics

Habib Latifizadeh

Dr. Chan Group/ Duke University, Department of Biostatistics and Bioinformatics

Oct 10, 2024

1. Mapping ABMs into ODEs/PDEs for System Dynamics
2. Simulate and Collect Data from ABM
3. Identify Key Variables and Patterns
4. Use Data-Driven Techniques (SINDy)
5. Validate Derived ODE/PDE
6. Implement PINN

Mapping ABMs into ODEs/PDEs for System Dynamics

- **Agent-Based Models (ABMs):**
  - ABMs simulate interactions between individual agents based on defined rules.
  - Example: Predator-prey models simulate individual animals as agents.
- **ODEs/PDEs:**
  - ODEs/PDEs describe macroscopic behavior at the population level.
  - They allow for analytical predictions and generalizations of system dynamics.
  - Example: The Lotka-Volterra ODE describes population dynamics between predators and prey.

Step 2: Implementing Predator-Prey as Agent-Based Model

- **Agent Design:**
  - **Prey Agents:** Each prey agent follows the rules for movement, reproduction, and avoidance.
  - **Predator Agents:** Each predator agent follows rules for hunting, consuming prey, and reproduction.
- **Agent Interactions:**
  - **Predation Rule:** If a predator encounters a prey, it consumes the prey and its energy increases.
  - **Reproduction Rule:** Prey and predators reproduce based on energy levels and proximity to others.
- **Simulation Goal:** To observe emergent population dynamics from agent interactions over time.

Step 1: Simulate and Collect Data from ABM

Data Collection:

- Run the ABM to simulate individual agent interactions over time. Collect aggregate measures such as population density and mean field quantities.
- Record population data for key variables (e.g., prey and predator populations).
- **Example:**
  - At each time step, track the number of prey and predators.
  - This time-series data will be used to infer the governing ODEs/PDEs.

Output: Data set of population levels over time.

Step 2: Identify Key Variables and Patterns

- **Key Variables:** Identify the most important macroscopic quantities.
- Example in Predator-Prey Systems:
  - $x(t)$: Prey population at time t.
  - $y(t)$: Predator population at time t.
- **Pattern Detection:**
  - Look for oscillations, stable points, or equilibria in the time-series data.
  - This provides insight into the underlying structure of the system.

Goal: Detect patterns that hint at possible mathematical relationships between variables.

Step 3: Use Data-Driven Techniques (SINDy)

- **Sparse Identification of Nonlinear Dynamics (SINDy):**
  - SINDy is a method that uses sparse regression to identify governing equations from data.
  - It assumes that the system can be described by a small number of terms.
- **Process:**
  - Collect time-series data from ABM.
  - Build a library of candidate functions (e.g., polynomials, trigonometric functions).
  - Perform sparse regression to select the most important terms.

Output: Governing ODEs or PDEs with estimated parameters.

Step 4: Validate the Derived ODE/PDE

- **Validation:**
  - Run simulations of the derived ODE/PDE model.
  - Compare the results to the original ABM data.
- **Fine-tuning:** Adjust the parameters of the derived equations to improve accuracy.
- Example:

$$\frac{dx}{dt} = \alpha x - \beta xy \quad \text{(adjusted based on ABM data)}$$

- If the model fits both the short-term and long-term dynamics of the system!?

Step 5: Implement PINN for ODE Approximation

- **Physics-Informed Neural Networks (PINNs):**
  - PINNs use neural networks to solve differential equations by embedding physical laws into the loss function.
- **Incorporate ODE Residuals:**
  - The ODE residuals (e.g., $\frac{dx}{dt} - (\alpha x - \beta xy)$) are included in the loss function.
  - This ensures that the neural network solution adheres to the dynamics described by the ODEs.
- **Predict Future Dynamics:**
  - Once trained, the PINN can be used to predict future predator-prey dynamics.
  - It provides solutions that align with both the data and the governing equations.
- Example: Forecast prey and predator populations for 100 more time steps.

Vivarium: Stochastic Predator-Prey Model

Habib Latifizadeh

Dr. Chan Group/ Duke University, Department of Biostatistics and Bioinformatics

September 26, 2024

Predator-Prey Model (Lotka-Volterra)

- **Overview:** The model simulates the dynamics between two species: one as predator and the other as prey.
- **Prey Population** $x(t)$ & **Prey Equation:**

$$\frac{dx}{dt} = \alpha x - \beta xy$$

  - αx: Natural growth of prey.
  - $-\beta xy$: Reduction in prey due to predation.
- **Predator Population** $y(t)$ & **Predator Equation:**

$$\frac{dy}{dt} = \delta xy - \gamma y$$

  - δxy: Growth of predator population due to consuming prey.
  - $-\gamma y$: Natural death of predators.

Reactions and Stoichiometry: Gillespie Stochastic Algorithm in Predator-Prey Model

Reactions:

- Prey birth: Prey population increases by 1.
- Predation: Prey is eaten, and predator population increases by 1.
- Predator death: Predator population decreases by 1.
- Predator birth: Predator population increases by 1.

Stoichiometry Matrix:

$$\text{Stoichiometry} = \begin{bmatrix} 1 & 0 \\ -1 & 1 \\ 0 & -1 \\ 0 & 1 \end{bmatrix}$$

- Rows represent reactions (prey birth, predation, predator death, predator birth).
- Columns represent change in prey and predator populations.

Gillespie Stochastic Algorithm in Predator-Prey Model

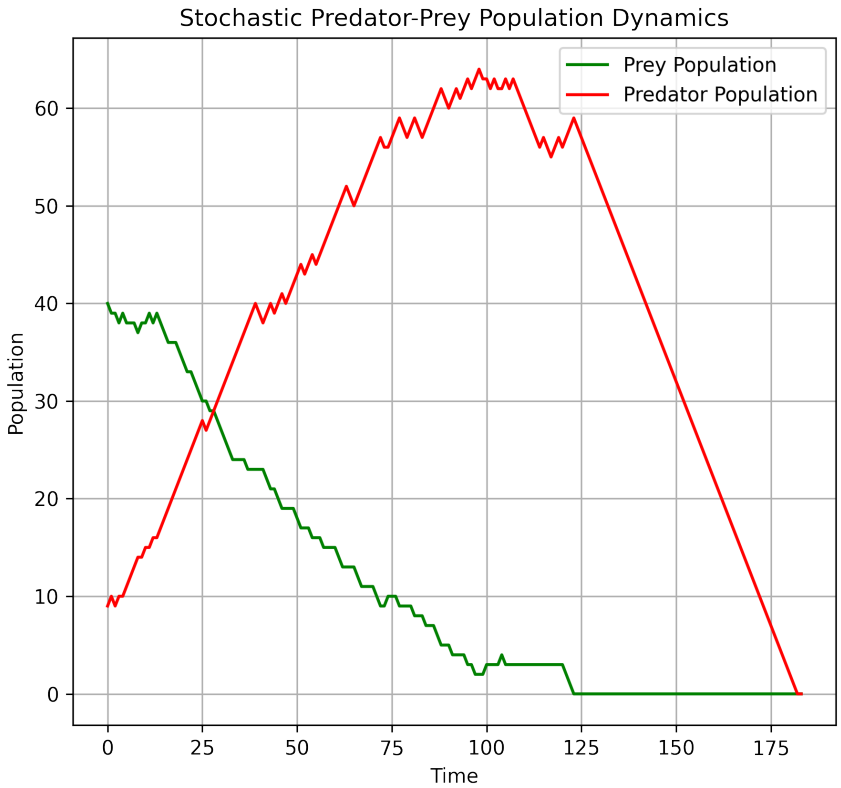
- Event propensities: Calculate rates for each event.
  - **Prey birth rate:** prey_birth_rate $= \alpha \cdot x$
  - **Predation rate:** predation_rate $= \beta \cdot x \cdot y$
  - **Predator death rate:** predator_death_rate $= \gamma \cdot y$
  - **Predator reproduction rate:** predator_reproduction_rate $= \delta \cdot x \cdot y$
- Calculate the total rate of all events.
- Randomly determine time to next event (exponentially distributed):

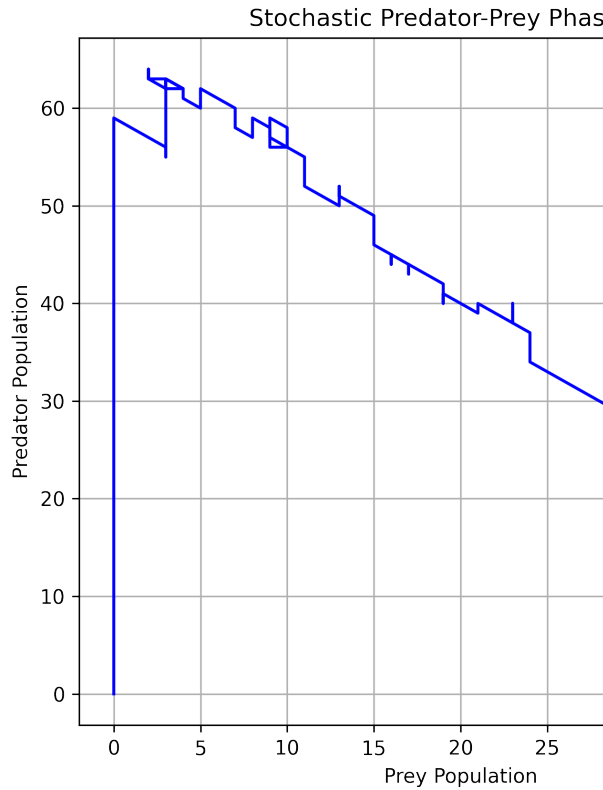
$$\tau = \text{Exponential}(1.0/\text{total\_rate})$$

- Event Occurrence: Determine the next event based on cumulative distribution of rates.
- Update the population of prey and predator accordingly.

Model Outputs

- **Stochastic result using Vivarium:** Time series of prey and predator populations.





Weekly Report

Habib Latifizadeh

Dr. Chan Group/ Duke University, Department of Biostatistics and Bioinformatics

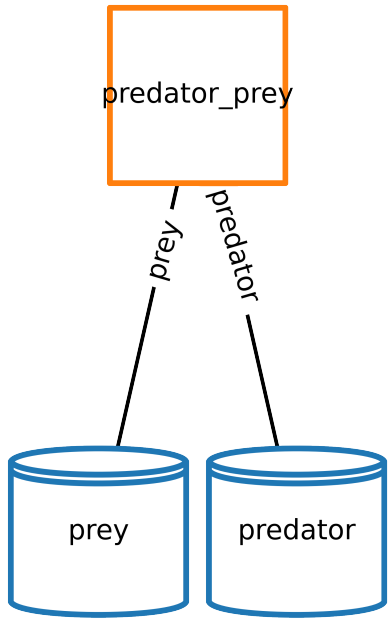
September 18, 2024

Predator-Prey Model (Lotka-Volterra)

- **Overview:** Model simulates dynamics between two species, one as predator and the other as prey.
- **Prey Population $x(t)$ & Prey Equation:** $\frac{dx}{dt} = \alpha x - \beta xy$**:**
  - αx: Natural growth of prey.
  - $-\beta xy$: Reduction due to predation.
- **Predator Population $y(t)$ & Predator Equation:** $\frac{dy}{dt} = \delta xy - \gamma y$**:**
  - δxy: Growth by consuming prey.
  - $-\gamma y$: Natural death in absence of prey.
- **parameters:**
  - α: Growth rate of prey.
  - β: Rate of predation.
  - γ: Natural death rate of predators.
  - δ: Predator growth rate from consuming prey.

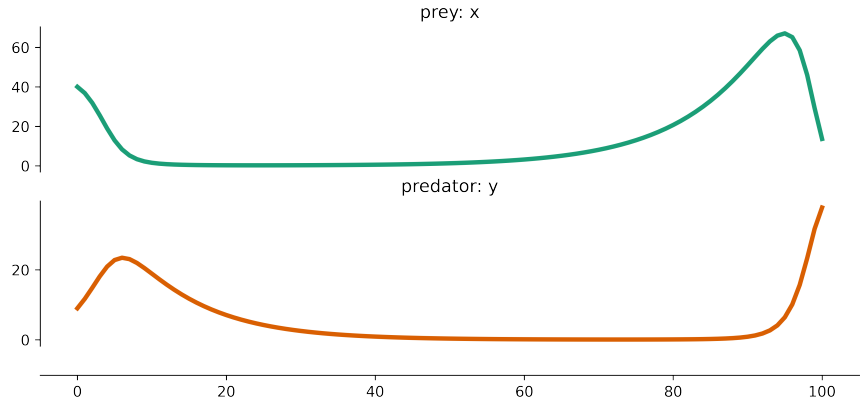
Vivarium Simulation Workflow and Results

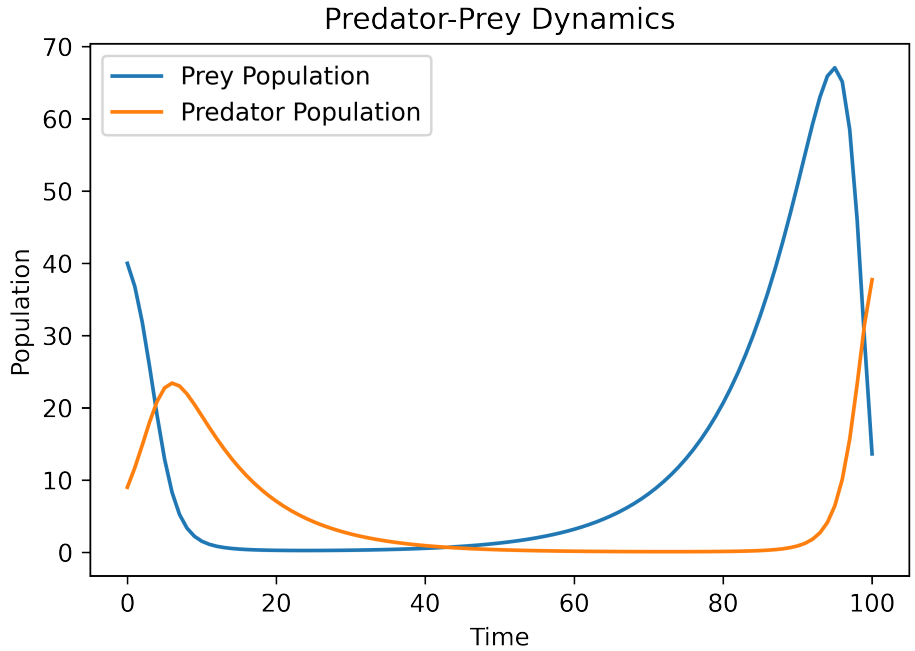
- **Vivarium Implementation & Topology Diagram:**
  - Processes: Prey growth, predation, predator growth, predator death.
  - Stores: Population sizes of predators and prey.
  - Composites: Combined processes simulate predator-prey interactions.
- **Workflow:**
  - Initialization of prey and predator populations.
  - Model execution using Lotka-Volterra equations.



Model Outputs

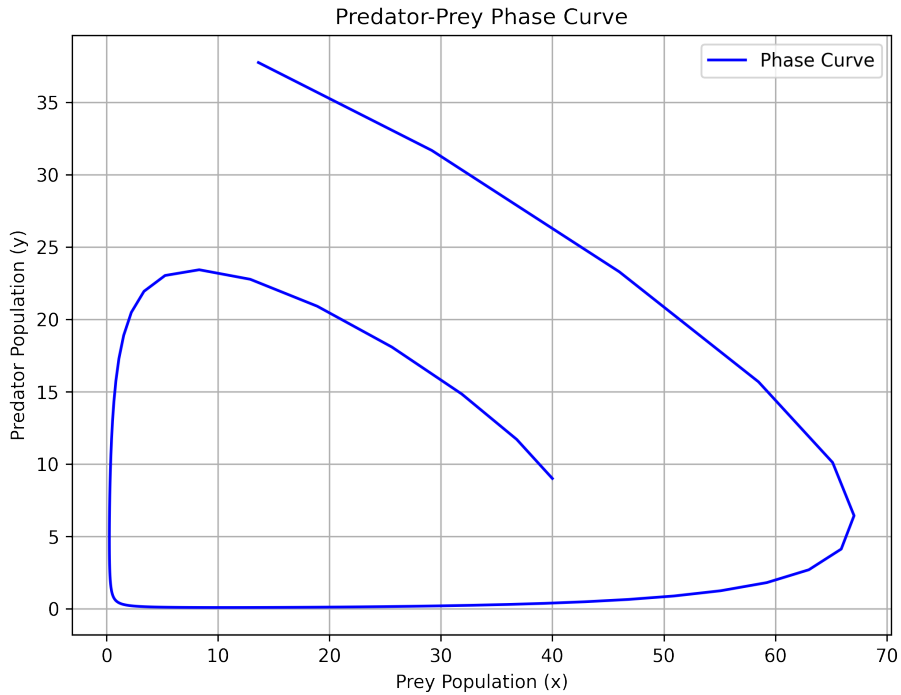
- **Vivarium result:** Time series of prey and predator populations.





Model Outputs

- **Phase Curve (Prey vs Predator):** The plot, the prey population on the x-axis and the predator population on the y-axis, shows how the populations change with respect to each other, rather than over time.



Stochastic Predator-Prey Model Explanation

- **Stochasticity:**
  - Introduced by multiplying the rates (e.g., α, β, γ, δ) by a normally distributed random variable.
  - Noise is controlled by the 'noise scale' parameter.
- **next_update() Function:**
  - Equations are updated to include noise in the growth and predation rates.
  - Simulates random fluctuations in predator and prey populations.

Weekly Report

Habib Latifizadeh

Dr. Chan Group/ Duke University, Department of Biostatistics and Bioinformatics

September 12, 2024

Research Progress

- **Vivarium Simulations:** Executed several Vivarium-based simulations modeling biological processes such as Transcription and Translation (Tx and TI) in both deterministic and stochastic frameworks with visualizations of mRNA and protein concentration changes over time. Additionally, a Composite Simulation (TxTI) with hierarchical embeding to simulate agent growth and division.
- **Visual Outputs:** Generated topology diagrams and timeseries data visualizing mRNA and protein concentration (included in the Jupyter notebook outputs)
- **Next Steps:** Further discussion on integrating these Vivarium simulations into CMV ODE model, and explorring any specific configurations for future runs.

Vivarium Overview

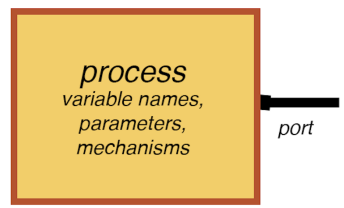
- **Modular Design:**
  - Flexible modeling by composing individual biological processes into composite models.
  - Biological processes modeled as independent modules connected through ports and stores.
- **Multiscale Integration:**
  - An integrative platform designed to support the simulation of biological processes across multiple scales (deterministic, stochastic, and agent-based models).
- **Scalable Computation:**
  - Parallel execution on multiple CPUs.
- **Libraries and Paradigms:**
  - vivarium-cobra: For flux balance analysis of metabolic networks using COBRA.
  - vivarium-bioscrape: For modeling chemical reaction networks (CRNs) like transcription, translation, and regulation using the Bioscrape.
  - vivarium-multibody: For spatial multicellular physics and diffusion, which models interactions between cells in a shared environment.

Vivarium's Interface Architecture

- **Process (rectangular flowchart):** A modular model contains parameters, an update function, and ports.
  - The building blocks of simulations and represent biological mechanisms (e.g., transcription and translation in gene expression).
  - Each process interacts with "ports" (data inputs/outputs) and "stores" (state variables).
- **Store (flowchart symbol):** Holds state variables and schemas, determining how updates to the system occur.

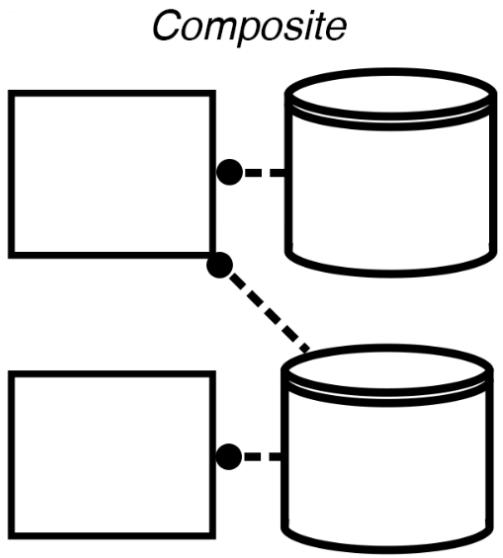
- Stores act as repositories where data (variables) used and generated by the processes are kept. Each variable in a store is defined with specific attributes that dictate how it behaves within the simulation.

- Schema: Each variable has a schema that defines its data type (integer, float, string) and the methods that can be applied to it. This schema ensures that different processes handle the data consistently and correctly.



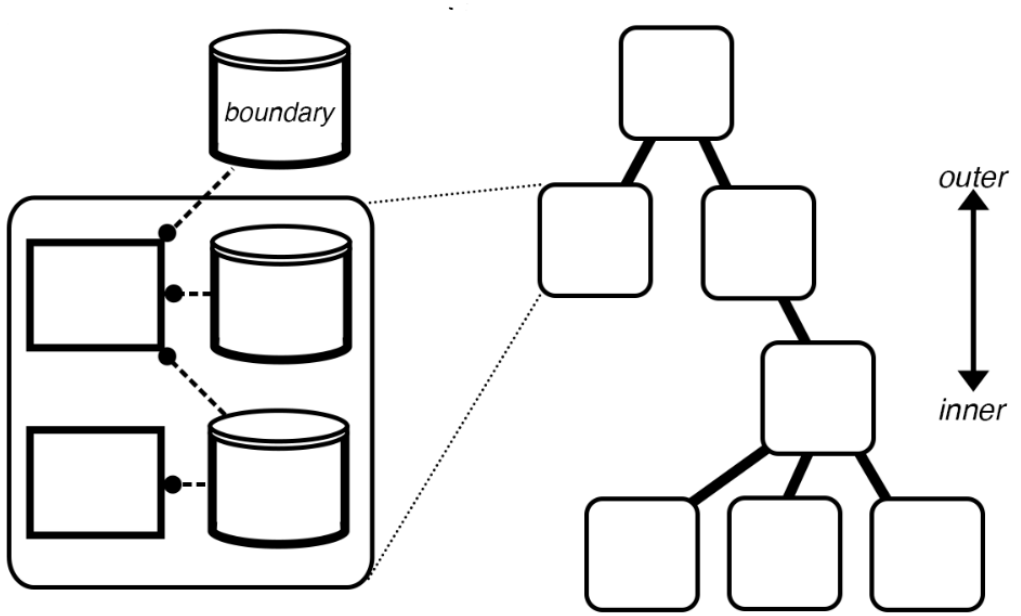
Vivarium's Interface Architecture

- **Composites:** Bundles of Processes and Stores wired together by a bipartite network/ Topology, with Processes connecting to Stores through their ports.
  - For instance, two processes, transcription and translation, are combined into a composite to simulate gene expression



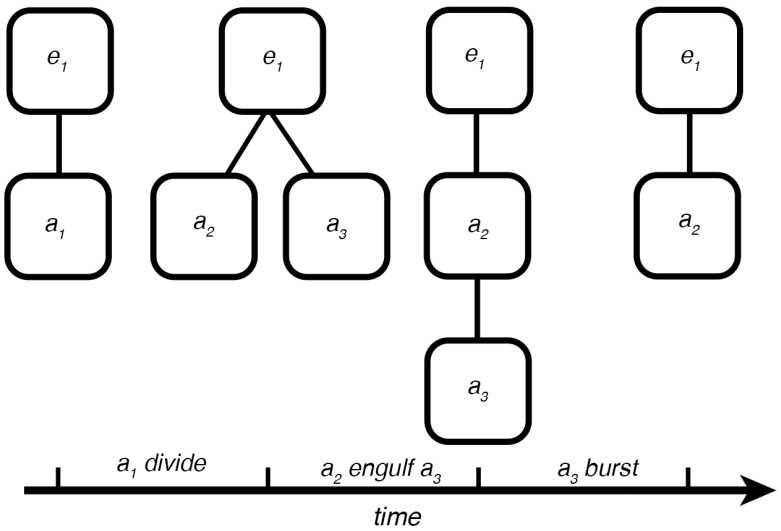
Vivarium's Interface Architecture

- **Compartments:** Stores with inner sub-Stores and Processes. Processes connect across compartments via boundary stores.
- **Hierarchy:** Compartments are embedded in a hierarchy as a place network with discrete layers, outer compartments above and inner compartments below.
- **Ports:** These serve as interfaces for the processes. Ports allow processes to connect to each other and to shared resources, facilitating the flow of data between them.



Hierarchy Updates in Modeling Systems

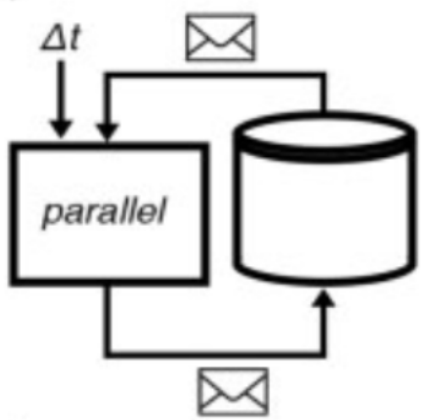
- **Divide Update:** Adds new compartment by splitting existing one.
  - *Purpose:* Simulates cell division—splits one cell into two distinct compartments.
- **Engulf Update:** One compartment absorbs another, which is then subsumed.
  - *Purpose:* Models processes like phagocytosis or integration of one system component into another.
- **Burst Update:** Deletes an engulfed compartment post-engulfment.
  - *Purpose:* Represents the breakdown or digestion of the engulfed entity within the system.



Vivarium Engine Operation

System Construction:

- The Vivarium engine constructs stores based on the declared schemas for each port of the processes.
- It assembles processes and stores into a hierarchy, executing their interactions over time.
- Aimed to support large models with thousands of mathematical equations.



Parallel Processing:

- Vivarium can distribute processes onto different OS processes (distinct from Vivarium processes) to handle complex and large-scale simulations efficiently.

Vivarium Applications

- The difference equation $\Delta x = f(r, x) \cdot \Delta t$ describes the dynamics of the system.
- A **Vivarium store** is a computational object that holds the system's state variables x.
- A **Vivarium process** is a computational object containing the update function f, which governs how variables evolve over time, describes the inter-dependencies between variables, and maps them from one time t to the next time $t + \Delta t$.
- **Processes** are configured by parameters r, which define how the update functions map input values to output values, giving them a distinct shape.

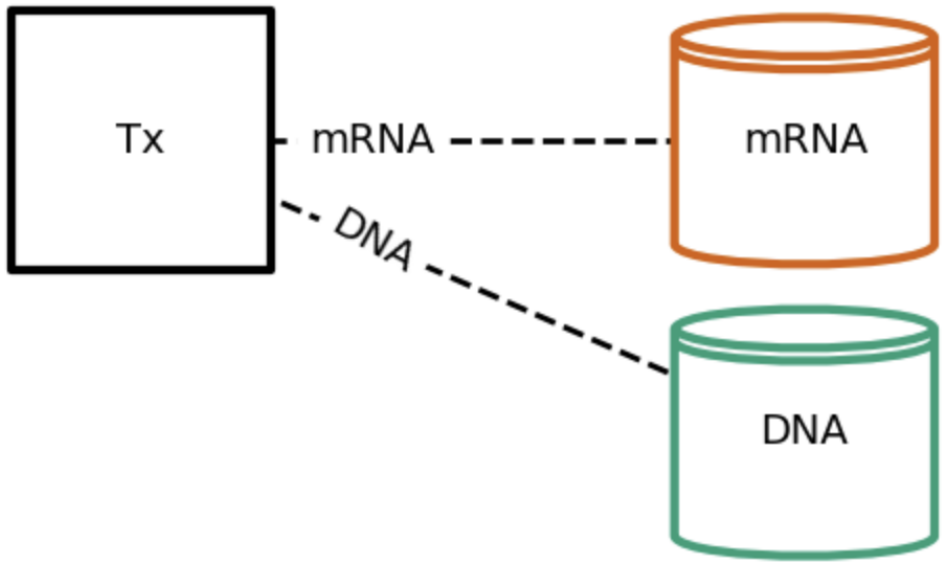
Vivarium Simulation

Simulated Biological Processes:

- **Transcription (Tx):** DNA to mRNA.
- **Translation (Tl):** mRNA to protein.
- **Stochastic Tx:** Randomized transcription process.
- **TxTl Composite:** Combined Tx and Tl, includes agent-based growth and division.

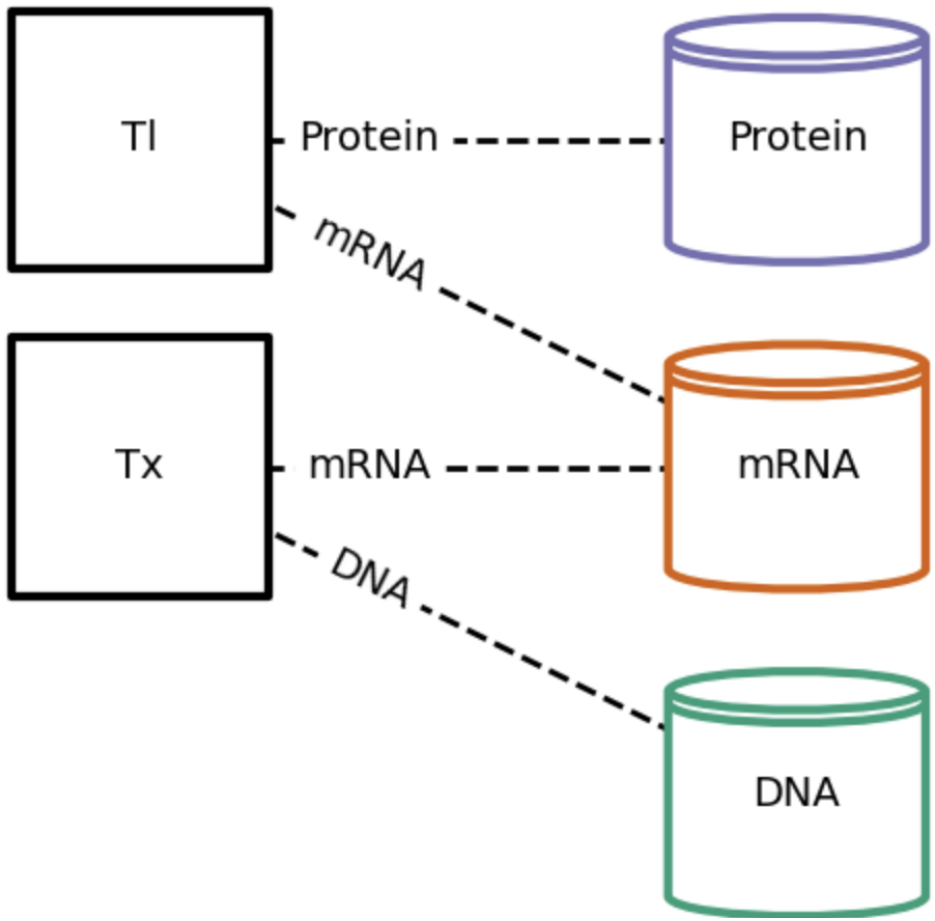
Vivarium Topology Outputs:

- Topology of Transcription (interactions between DNA, mRNA).



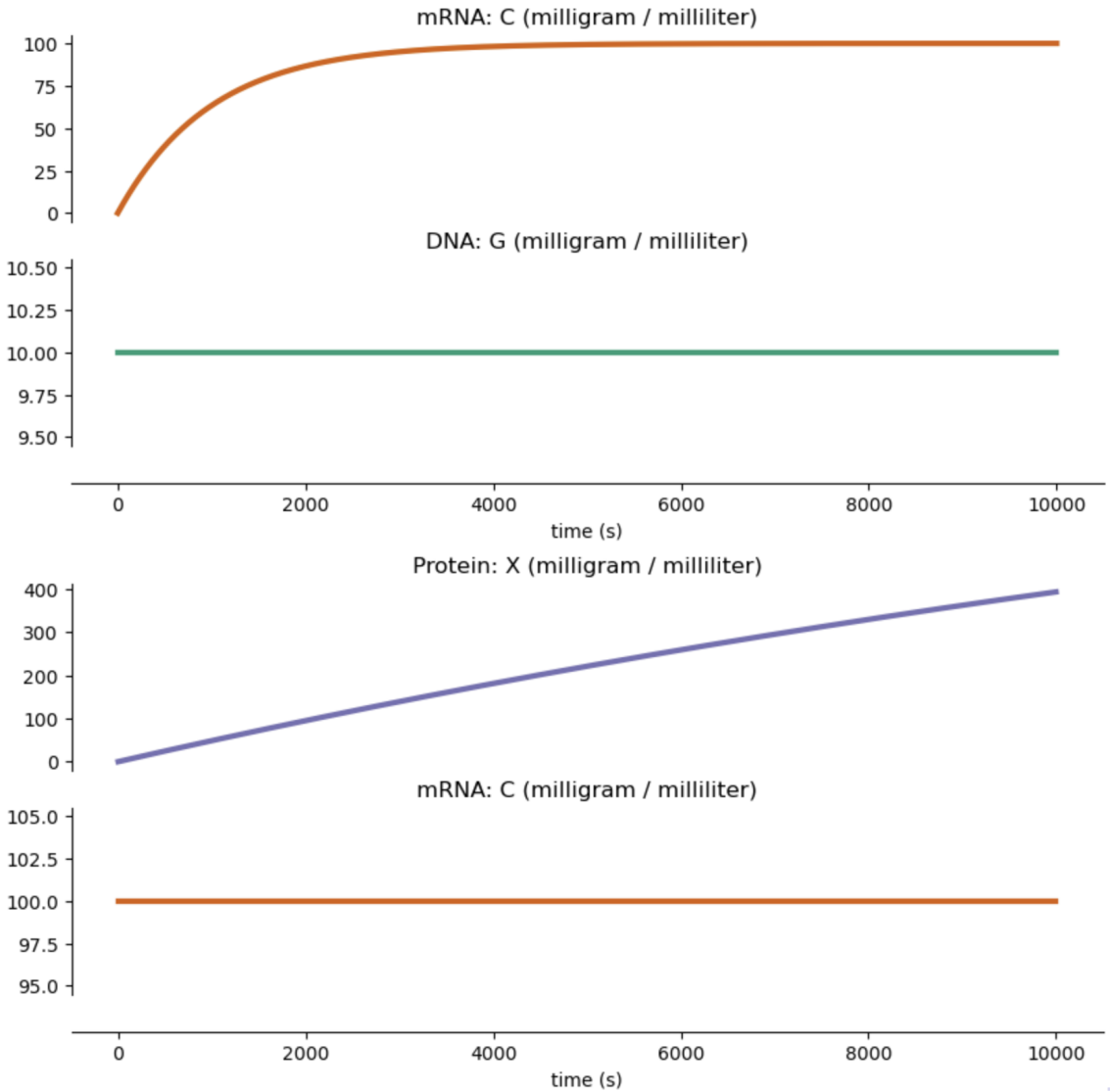
Vivarium Topology Outputs

- Topology diagrams of Translation (interactions between DNA, mRNA, protein).



Vivarium Topology Outputs:

- Timeseries plots of mRNA and protein concentrations over time.



Challenges and Future Directions

- **Implementing CMV ODE System in Vivarium:** Integrating the CMV ODE system into the Vivarium framework to model viral dynamics and immune interactions.
- **Agent-Based and ODE Hybrid Modeling:** Combining agent-based models and ODE systems to simulate both individual-level cell interactions and population-level dynamics in the CMV infection model.
- **Stochastic Processes in Vivarium:** Incorporating stochastic elements into the Vivarium ODE system to reflect random events in viral transmission and immune responses.
- **Real-World Data Validation in Vivarium:** Using clinical and experimental data to validate and fine-tune the Vivarium-based CMV ODE model for accurate real-world predictions.

Weekly Report

Habib Latifizadeh

Dr. Chan Group/ Duke University, Department of Biostatistics and Bioinformatics

September 5, 2024

Research Progress

- **Vaccine-Induced Antibody Responses:**
  - Reviewed gB/MF59 CMV vaccine study, showing antibodies that prevent cell-to-cell spread.
  - Plan to integrate antibody data into CMV transmission models to improve predictions.
  - Also reading Dr. Permar's study on mRNA-1647 vaccine, which shows stronger neutralization and antibody responses.
  - Both studies could help refine our CMV models and improve prevention strategies.
- **Agent-Based Modeling (ABM):**
  - Exploring ABM to simulate immune cell responses.
  - Discuss ABM strategies with a researcher to enhance model accuracy.
- **Immune Checkpoints and Vaccine Development:**
  - Investigating how targeting immune checkpoints could enhance vaccine responses.

Agent-Based Modelling

- **A bottom-up modelling approach:**
  - Systems' behavior emerges from actions and interactions of autonomous agents (e.g., patients and providers) with each other and the environment over time.
- **An experimental approach to:**
  - Testing explanatory hypotheses of real-world patterns.
  - Translating individual-level assumptions to population models.
  - Examining population-level effects as a decision support tool.
  - Guiding data collection.

Introduction to ABMs

- **Definition:** ABMs analyze complex systems by simulating individual interactions (micro-level) to reveal system-level phenomena (macro-level).
- **Key Components:**
  - **Agents:** Autonomous entities interacting with the environment and other agents based on rules, adapt based on interactions.
  - **Environment:** The space where agents interact. It can be real-world locations (GIS integration) that determine how agents move and connect.
  - **Rules:** Govern agent behavior and interactions which Define agent interactions and environment dynamics.

Why ABMs? & Challenges in ABM Development

Why ABMs?

- **Advantages:**
  - Effective for studying systems involving individual interactions.
  - Emergent behavior through agent interactions provides insights into complex phenomena.
- **Complexity:**
  - Difficulty in developing for non-computer scientists.
  - Need for easy-to-use, efficient platforms for model development.
- **ABM Tools:**
  - Provide frameworks to abstract development complexity, allowing users to focus on simulation outcomes.

Key Desiderata for ABM Tools

- **Efficiency and Ease of Use:** Balancing performance and accessibility.
- **Visualization and Analysis:** Real-time monitoring and data visualization are critical.
- **Stochastic Processes:** Handling randomness in simulations through robust random number generation.

Popular ABM Tools Overview

- **ActressMAS**: .NET-based, easy-to-use, not for high-performance needs.
- **AgentPy**: Python library, Jupyter integration, supports parallel simulations.
- **Agents.jl**: Julia-based, efficient and scalable, supports GIS data.
- **Care HPS**: C++ tool for high-performance parallel computing.

Advanced ABM Tools

- **FLAME & FLAME GPU**: For parallel computing, supports XML-based modeling for high-performance simulations on supercomputers and GPUs.
- **GAMA**: Integrated with Eclipse IDE, supports massive simulations with GIS capabilities.
- **NetLogo**: Standard ABM platform, includes VPL, integrates with Python (PyNetLogo) and R (RNetLogo).

Conclusions

- ABMs provide a powerful method for analyzing complex systems using a bottom-up approach.
- Choosing the right tool depends on the user's technical skills, model complexity, and performance needs.
- **Future Directions**: Exploring distributed computing applications like federated learning and blockchain systems in ABM research.

Vivarium Overview

- **Modular Design:**
  - Flexible modeling by composing individual biological processes into composite models.
  - Biological processes modeled as independent modules connected through ports and stores.
- **Multiscale Integration:**
  - An integrative platform designed to support the simulation of biological processes across multiple scales (deterministic, stochastic, and agent-based models).
- **Scalable Computation:**
  - Parallel execution on multiple CPUs.
- **Libraries and Paradigms:** Vivarium integrates several libraries to extend its functionality, such as:
  - vivarium-cobra: For flux balance analysis of metabolic networks using COBRA.
  - vivarium-bioscrape: For modeling chemical reaction networks (CRNs) like transcription, translation, and regulation using the Bioscrape.
  - vivarium-multibody: For spatial multicellular physics and diffusion, which models interactions between cells in a shared environment.

Vivarium Processes and Example Applications

- **Processes, Stores and Composites:**
  - Processes are the building blocks of simulations and represent biological mechanisms (transcription and translation in the gene expression example).
  - Each process interacts with "ports" (data inputs/outputs) and "stores" (state variables).
  - Composites are higher-level constructs that combine multiple processes. For instance, transcription and translation are combined into a composite to simulate gene expression, integrating the two processes into a unified model.
- **E. coli Colony Simulation:**
  - Integrated flux balance analysis (FBA), gene expression (transcription and translation), and multicell physics.
  - Demonstrates multiscale modeling potential for CMV simulations.

Relevance to CMV Research

- **Viral Dynamics and Immune Response:**
  - Use Vivarium to simulate viral replication, immune responses, and even vaccine effects at multiple biological scales.
- **Scalability:**
  - Applicable for both within-host dynamics and population-level CMV studies using Agent-Based Modeling.
    1. Within-host ABM simulates interactions between viral particles, immune cells, and other biological entities inside the host (the spread of CMV within tissues or between cells).
    2. Population-level ABM models interactions between individuals in a population, including infection transmission, vaccination effects, and epidemiological outcomes.

Challenges and Future Directions

- **CMV Model Refinement and Vivarium integration:**
  - Refining agent-based models (ABM) for simulating immune cell behaviors.
  - Implementation of CMV modeling progress and Vivarium integration.
- **Experimental Data Integration:**
  - Exploring dynamic integration of patient data to improve model accuracy.
- **Grant Writing:**
  - Planning to develop an idea for writing a T32 postdoc grant proposal to secure funding for further research and model development.