Epidemic Spread: A Multi-Paradigm Comparative Analysis using the Mesa Framework

CMCS Exam Project

Francesco Romeo - 634705

2026

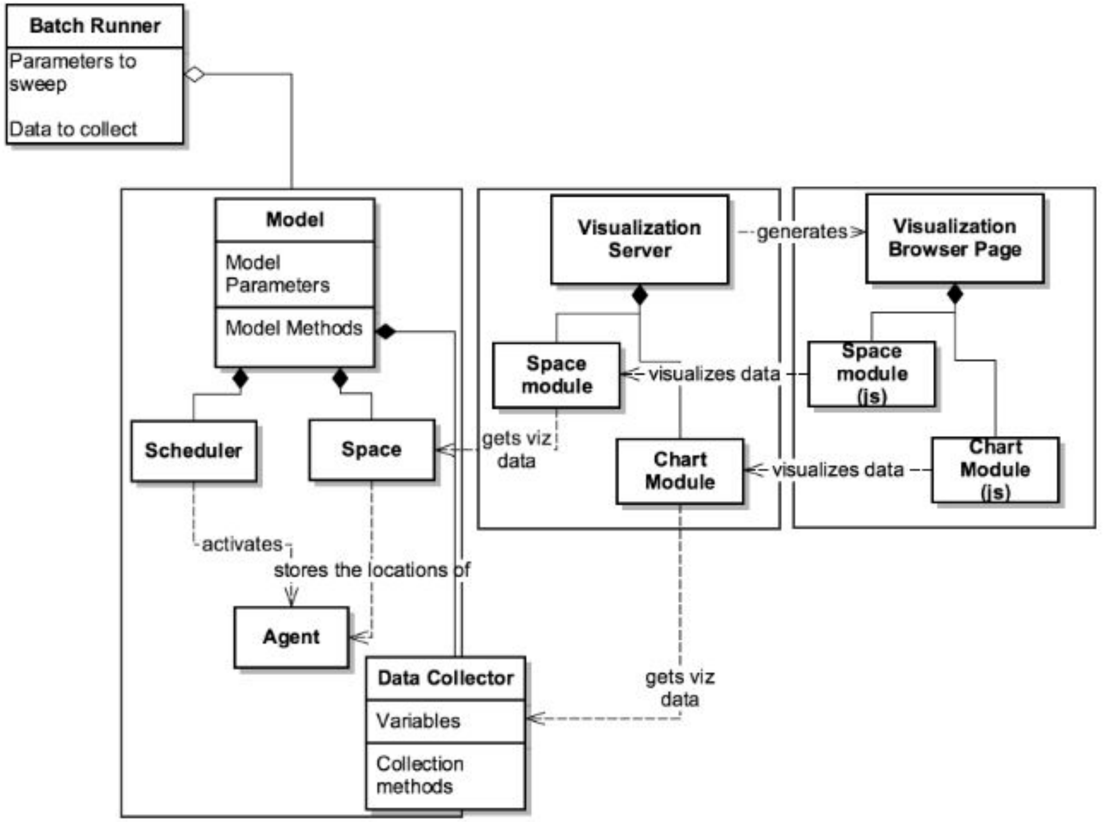
The "Python Gap" (Circa 2015)

While tools like NetLogo and Repast were standard, they lacked integration with Python's growing scientific ecosystem.

Mesa's Goal: To be a Python 3-based alternative built from scratch, allowing for modular components.
Mesa combines the speed of core components with the flexibility of custom implementations, leveraging web browsers for visualization and the Python stack for data analysis.

Philosophy: Mesa avoids rigid defaults (e.g., offering multiple flexible schedulers) and GUI dependencies (using JavaScript/Browsers).

Core Architecture



Core Architecture

The Model (Container & Architect)

- **Function:** The Model class holds global parameters (e.g., num_agents) and instantiates the core: Agents, Space, and Scheduler.
- **Integration:** Connects simulation to Analysis (DataCollector/BatchRunner) and Visualization (Browser-based via JavaScript).

The Agent (State & Action)

- **Definition:** Objects with a unique_id and internal state (e.g., wealth = 1).
- **Interface:** Must subclass Agent and implement a .step(model) method.
- **Logic:** Receives model state as an argument to execute logic (e.g., "give money") when triggered by the Scheduler.

The Scheduler (Time & Activation)

- **Regimes:** Supports **Synchronous** (Simultaneous), **Uniform** (Ordered), **Random** (Shuffled), and **Poisson** (Random Interval).
- **Mechanics:** Standard add/remove interface for dynamic agents. Tracks both **steps** (discrete ticks) and **time** (continuous/simulated clock).

Core Architecture

The space (Grids & Neighborhoods)

- **Space Categories:**
  - Discrete (Grids) vs. Continuous
- **Grid Occupancy Constraints:**
  - SingleGrid
  - MultiGrid
- **Topology Properties:**
  - Toroidal (Wrap-around boundaries)

Integrated Analysis: From Simulation to DataFrames

- **The DataCollector:**
  - **Generic:** No subclassing needed;
  - **Scopes:** Captures **Model-level** and **Agent-level** variables per step.
  - **Pandas Integration.**
- **The BatchRunner:**
  - **Automation:** Executes parameter sweeps and Monte Carlo replications.
  - **Purpose:** Essential for analyzing stochasticity and sensitivity in ABMs.

Visualization: The Client-Server Model

- **Architecture:** Decoupled design avoids both the developers and the users needing to deal with cross-system GUI programming
  - **Server (Python)**
  - **Client (JavaScript)**
- **Development:**
  - **Dual Requirement:** Custom visual elements require both a Python class (to parse data to JSON) and a JavaScript file (to render it).

Evolution of Core Mechanics (2015 vs. 2025)

From Manual Management to "Query-Based" Logic

- **Agent Management:**
  - • **2015:** Agents were managed via simple lists or stored directly *inside* the Scheduler. You had to explicitly add agents: schedule.add(agent).
  - • **2025:** Introduction of **AgentSet**. Agents auto-register upon creation. You can filter, group, and select them like a database (e.g., select, groupby), strictly separating the *population* from the *time*.
- **Time Management:**
  - **2015:** Time was controlled by a monolithic Scheduler Class (e.g., RandomActivation) that owned the agent list and dictated the execution order internally.
  - • **2025:** The Scheduler API is removed. Time is now an **action** applied to an agent set (e.g., model.agents.shuffle_do("step")). You have explicit control over *who* acts, *when*, and in *what order*.
- **Environment and Space:**
  - **2015:** Discrete grids (Single/MultiGrid) or standard continuous space.
  - **2025:** Introduction of **Property Layers**, efficient NumPy-based arrays for managing environmental properties (e.g., elevation, resources) at the cell level. Addition of specialized spatial agents (CellAgent, FixedAgent).

Modernization: Visualization & Ecosystem

From Custom Development to Modern Standards

- **Visualization:**
  - **2015:** Required maintaining custom JavaScript code coupled with Python, using a Tornado server and manually managed WebSockets.
  - **2025:** Migration to **Solara**, a standardized library based on React but managed entirely in Python, simplifying the creation of interactive dashboards and real-time controls.
- **Performance and Ecosystem:**
  - **2015:** Focused on creating a basic Python alternative to tools like NetLogo or Repast.
  - **2025:** Expansion into a complete ecosystem with high-performance extensions like **Mesa-Frames** (accelerated simulations) and **Mesa-Geo** (GIS modeling).
- **Stability and Development:**
  - **2025:** Adoption of a "two-track" development model: new features are released as **Experimental** (API may change) before graduating to **Stable** (Semantic Versioning), ensuring greater reliability for research.

Introduction to the problem

- **Complexity & Emergence:** Treat epidemics as complex systems where macro-level outcomes are driven by micro-level, non-linear human interactions.

- **The Modeling Gap:** Traditional deterministic models (ODEs) offer fast "mean-field" averages but fail to capture critical phenomena like **stochastic extinction** or the influence of **spatial clustering**.

- **Topological Validation:** Measuring how specific network structures (**Small World, Scale-Free, etc...**) deviate from the "well-mixed" assumptions of classical mathematics.

- **Assumptions**: constant population, random contacts between individuals, recovered are immune

Model Architecture & Dynamics

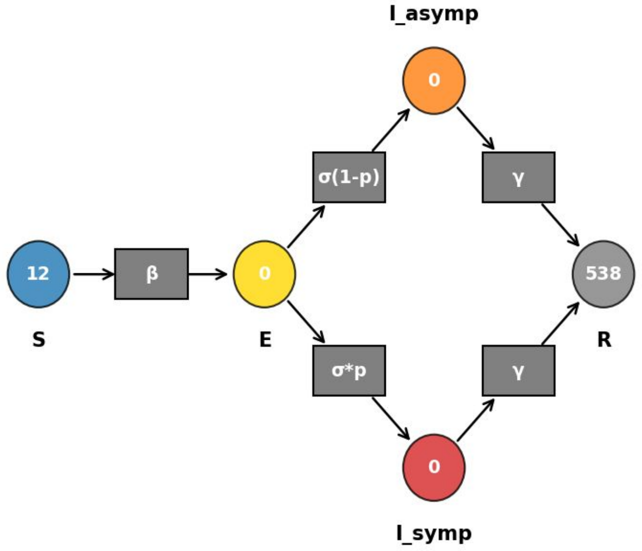
1. Methodology

The simulation validates epidemic trends by comparing three distinct modeling paradigms:

- **ABM (Agent-Based):** Stochastic, spatial (Grid/Network) modeling of individual interactions and heterogeneity.
- **ODE (Deterministic):** Mean-field SEIR differential equations for aggregate population trends.
- **Gillespie (SSA):** Exact stochastic event-time algorithm to capture fluctuations in finite populations.

2. The SEIR State & Vital Dynamics

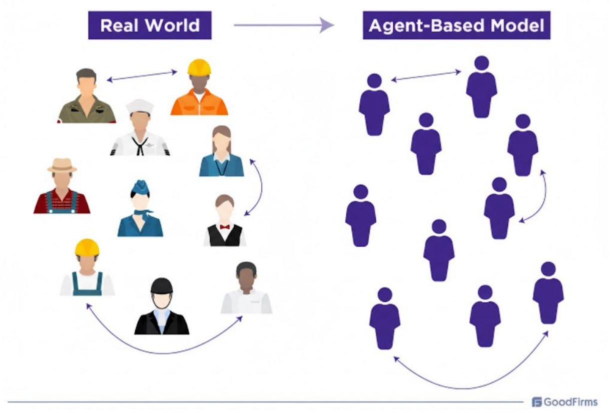
- **Agent States**: Agents transition through: Susceptible (0), Exposed (1), Infected Asymptomatic (2), Infected Symptomatic (3), and Recovered (4).
- **Disease-Specific Mortality (μdisease)**: Introduced a death probability for infected agents based on age-group distributions (Child, Teen, Adult, Senior).
- **Vital Dynamics & Respawn**: To maintain a constant population N, any death (natural or disease-induced) triggers the immediate "respawn" of a new agent.
- **Neonatal Vaccination**: New agents are initialized as Recovered (Immune) with probability p (vax_pct) or Susceptible with probability 1−p.



Agent-Based Modeling (ABM)

Definition: Agent-based modeling is a computational methodology used in social science, biology, and other fields, which involves simulating the behavior and interaction of many autonomous entities, or agents, over time.
Agents may represent individual organisms, humans, entire organizations, or abstract entities.

Key Advantage:
1. Unlike other modeling approaches, ABMs capture the path as well as the solution, so one can analyze the system's dynamic history.
2. Most social processes involve spatial or network attributes, which ABMs can incorporate explicitly.
3. When a model (A) produces a result (R), one has established a sufficiency theorem, meaning R if A.



Mathematical Baselines (ODE)

The standard SEIR system of ordinary differential equations is extended to include:

- vital dynamics (births and natural deaths),
- vaccination at birth,
- disease-induced mortality,
- explicit demographic replacement to keep the total population constant.
- a time-dependent control factor p(t) applied to the transmission rate to model non-pharmaceutical interventions (lockdown).

$$\begin{cases} \frac{dS}{dt} = \mu N(1-p) - \beta\, p(t)\, \frac{SI}{N} - \mu S + \mu_{\text{disease}} I \\ \frac{dE}{dt} = \beta\, p(t)\, \frac{SI}{N} - \sigma E - \mu E \\ \frac{dI}{dt} = \sigma E - \gamma I - \mu I - \mu_{\text{disease}} I \\ \frac{dR}{dt} = \gamma I + \mu N p - \mu R \end{cases}$$

The total population size N is assumed to be constant over time.
With:

S, E, I, R: susceptible, exposed, infectious, recovered

N: total population (constant)

β: transmission rate

σ: incubation rate

γ: recovery rate

μ: natural mortality/birth rate

$\mu_{(\text{disease})}$: disease-induced mortality

p: vaccination at birth

p(t): transmission control factor

If lockdown is not enabled, then p(t)=1 otherwise, p(t) is defined as

$$p(t) = 1 - (1 - p_{\text{lock}})\frac{1}{1 + e^{-k\left(\frac{I}{N} - \theta\right)}}$$

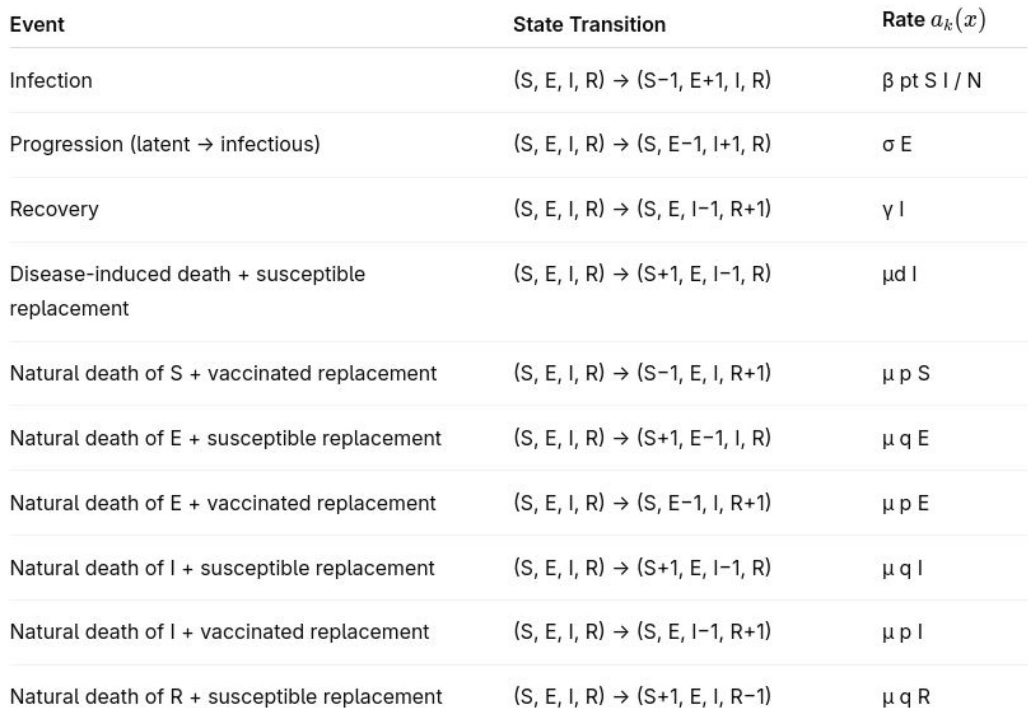
Stochastic Model (Gillespie SSA)

The standard SEIR model is adapted to a stochastic framework using the **Gillespie Algorithm** to capture discrete events and random fluctuations, which are lost in deterministic ODEs.

- **Vital Dynamics & Vaccination:** Every natural death (***μ***) or disease-induced death ($\mu_{disease}$) is coupled with an immediate **demographic replacement** to keep N constant.
  - Newborns are vaccinated with probability p=vax_pct (entering **R**).
  - Otherwise, they enter the system as Susceptible (**S**).
- **Dynamic Intervention:** A time-dependent factor p(t) modulates the transmission rate β when the infected ratio I/N exceeds the lockdown_threshold.
- **Event selection:** Instead of continuous flows, the system evolves through discrete events:
  - **Time step τ:** The time until the next event is drawn from an exponential distribution

    $\tau = \frac{1}{R_{\text{tot}}} \ln\left(\frac{1}{r_1}\right)$ representing the time until the next interaction.
  - **Event selection:** A specific transition (e.g., Infection, Recovery) is chosen with probability proportional to its rate $\mathbb{P}(E_i) = \frac{a_i}{R_{\text{tot}}}$

Stochastic Model (Gillespie SSA)



Event	State Transition	Rate $a_k(x)$
Infection	(S, E, I, R) → (S−1, E+1, I, R)	β pt S I / N
Progression (latent → infectious)	(S, E, I, R) → (S, E−1, I+1, R)	σ E
Recovery	(S, E, I, R) → (S, E, I−1, R+1)	γ I
Disease-induced death + susceptible replacement	(S, E, I, R) → (S+1, E, I−1, R)	μd I
Natural death of S + vaccinated replacement	(S, E, I, R) → (S−1, E, I, R+1)	μ p S
Natural death of E + susceptible replacement	(S, E, I, R) → (S+1, E−1, I, R)	μ q E
Natural death of E + vaccinated replacement	(S, E, I, R) → (S, E−1, I, R+1)	μ p E
Natural death of I + susceptible replacement	(S, E, I, R) → (S+1, E, I−1, R)	μ q I
Natural death of I + vaccinated replacement	(S, E, I, R) → (S, E, I−1, R+1)	μ p I
Natural death of R + susceptible replacement	(S, E, I, R) → (S+1, E, I, R−1)	μ q R

- **S**: Susceptible individuals
- **E**: Exposed (latent) individuals
- **I**: Infectious individuals
- **R**: Recovered individuals (recovered or vaccinated)
- **N**: Total population (constant)
- **β**: Transmission rate
- **pt**: Lockdown modifier (transmission reduction factor)
- **σ**: Progression rate (E → I)
- **γ**: Recovery rate
- **μd**: Disease-induced mortality rate
- **μ**: Natural death rate (equal to birth rate)
- **p**: Probability that a newborn is vaccinated (enters R)
- **q**: Probability that a newborn is susceptible (enters S)

Scheduling & Execution Strategies

Agent Architecture:

The simulation implements a modular agent logic to ensure flexibility across different execution regimes:

- **Split-Step Logic:** Actions are decoupled into **step_sensing()** (risk evaluation) and s**tep_apply()** (state transition).
- **Order-Bias Mitigation:** Decoupling ensures that an agent’s state change in the current step does not unfairly influence the decisions of others during the same cycle.
- **Sequential Wrapper:** A **step_sequential()** method is provided for schedulers that require immediate, one-by-one updates.

Scheduler Implementations:

- **Random (Shuffled):** Agents are activated in a randomized order each step, providing a stochastic approach to sequential updates.
- **Uniform (Ordered):** A fixed-order execution for comparative baseline testing.
- **Poisson (Random Interval):** Only a random subset of agents (following a Poisson distribution) is activated per step, approximating asynchronous time-step dynamics.
- **Synchronous (Simultaneous):** All agents "sense" the environment before any "apply" changes; this is the most robust method for preventing update-order bias.

Network Topologies & Space

The simulation supports five distinct interaction environments to evaluate how connectivity patterns influence viral transmission:

- **2D MultiGrid:** A classic lattice structure where agents move via random walks.
- **Network Topologies (NetworkX):** Structures where transmission occurs probabilistically across existing connections at each time step.:
  - **Small World (Watts-Strogatz):** High clustering with "shortcuts" that facilitate rapid, long-range infection jumps.
  - **Scale-Free (Barabási-Albert):** Highly heterogeneous degree distribution where "hubs" drive super-spreading events.
  - **Erdős-Rényi:** A baseline random graph model with uniform connection probabilities.
  - **Community Structure:** Distinct clusters (connected caveman graph) with sparse inter-connections to model localized outbreaks and targeted lockdowns.

Interventions (Lockdown & Vaccines)

1. Adaptive Lockdown (Dynamic Social Distancing)

- **Trigger Mechanism:** Automatically activates when the percentage of symptomatic agents exceeds the lockdown_threshold_pct.
- **Operational Effect:** Increases "Social Distancing" by reducing agent movement and contact probability by the lockdown_max_sd factor.
- **Multi-Model Response:**
  - **ABM:** Agents stop moving on the grid/network, creating local clusters of isolation.
  - **ODE/Gillespie:** The β parameter is scaled down globally to reflect the reduced contact rate.
- **Result:** Effectively "flattens the curve," delaying the peak.

2. Strategic Vaccination

- **Random Strategy:** Distributed uniformly, providing a baseline for herd immunity regardless of social structure.
- **Targeted Strategy (Network-Aware):** Specifically vaccinates nodes with the highest **degree centrality** (Hubs).
- **Efficiency:** Shows that targeted immunization can achieve herd immunity with significantly lower coverage compared to random distribution.

Multi-Paradigm Validation & Analysis

1. Live Simulation: Model Cross-Validation

- **Triangulation:** Overlays ABM (Solid), ODE (Dashed), and Gillespie (Dotted) curves to verify consistency across approaches.
- **Topology Impact:** Highlights how the network structure (e.g., Watts-Strogatz vs. Grid) alters the speed of contagion compared to the ODE and Gillespie.

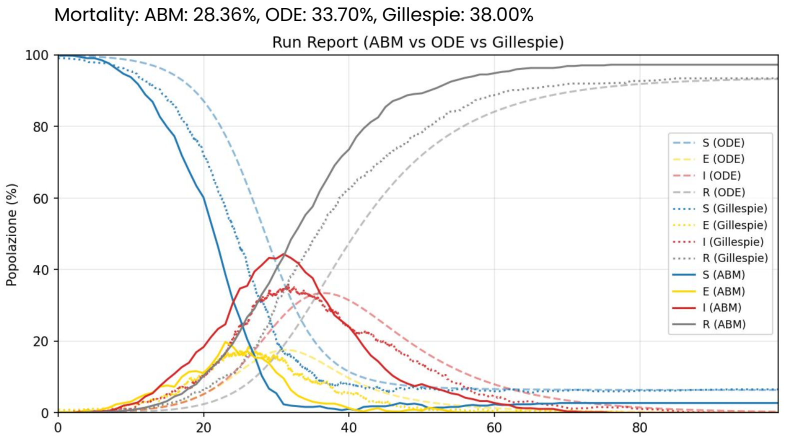
2. Batch Analysis: Monte Carlo & Risk Management

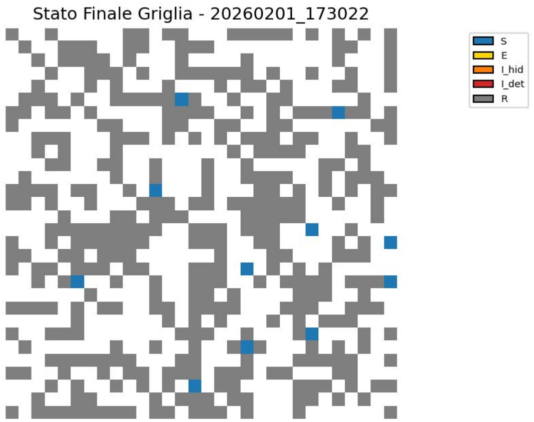
- **Stochastic Variance:** Runs 50+ iterations to capture the range of possible outcomes.
- **Peak Distribution:** Generates a histogram of maximum infections to assess the tail-end risks.
- **Capacity Planning:** Calculates the probability that infections will breach a safety threshold.

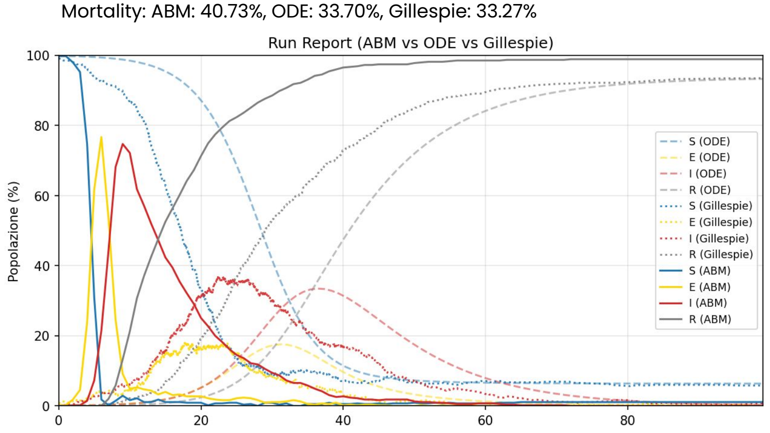
3. Parameter Sweep: Sensitivity & Policy Design

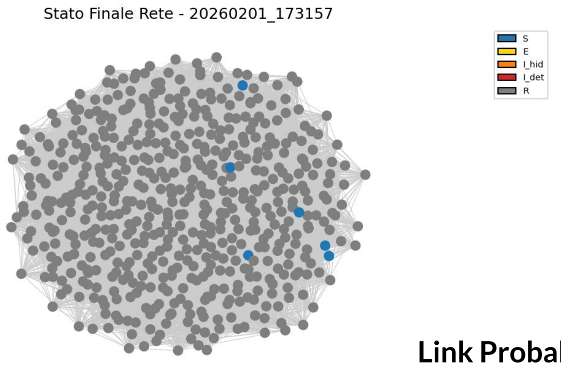
- **Variable Interaction:** Tests parameters like Transmission Rate (β) or Vaccination Pct to map the system's behavior.
- **Critical Thresholds:** Pinpoints the exact values where interventions (like Lockdown) successfully shift the system into a recovery phase.
- **Optimization:** Guides the selection of the most cost-effective intervention strategy (e.g., random vs. targeted vaccination).

Results (Grid & Erdos Renyi)





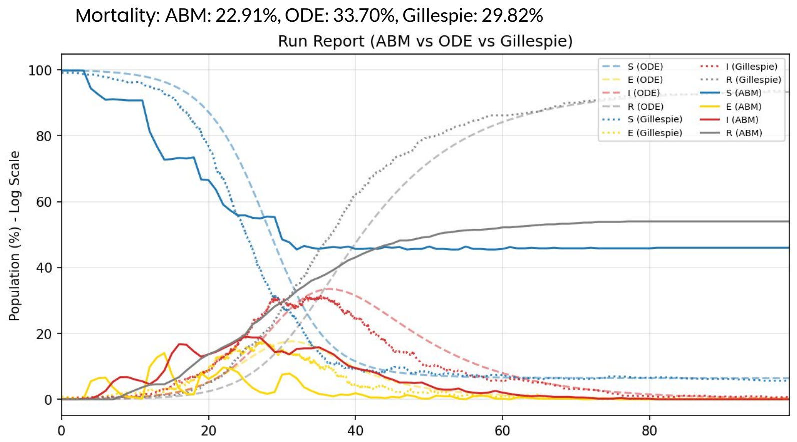


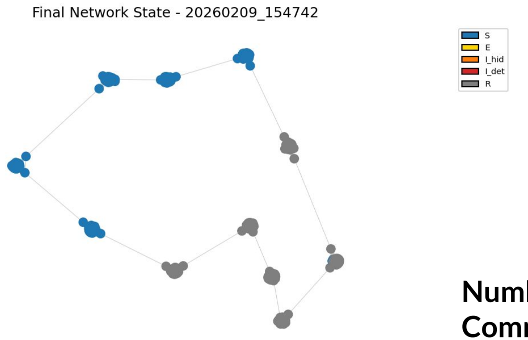


- **Population (N):** 550 agents.
- **Duration:** 100 simulation steps.
- **Transmission (β):** 0.6.
- **Recovery (γ):** 0.1.
- **Incubation:** 3 days (mean).
- **Symptomatic Probability:** 0.6.
- **Activation:** Synchronous (Simultaneous).
- **Interventions:** No initial vaccination; Dynamic Lockdown disabled.

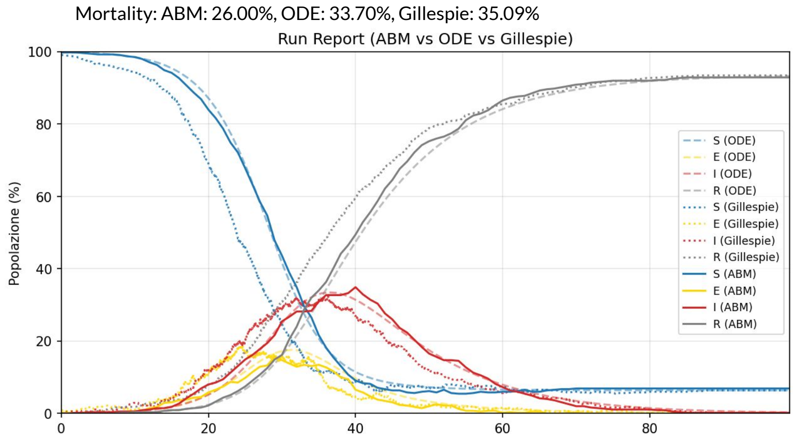
Link Probability: 0.1

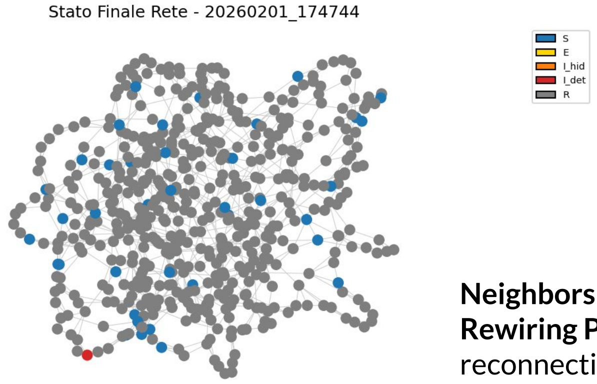
Results (Communities & Small World)





Number of Communities: 11
Community Size: 50 agents per group.

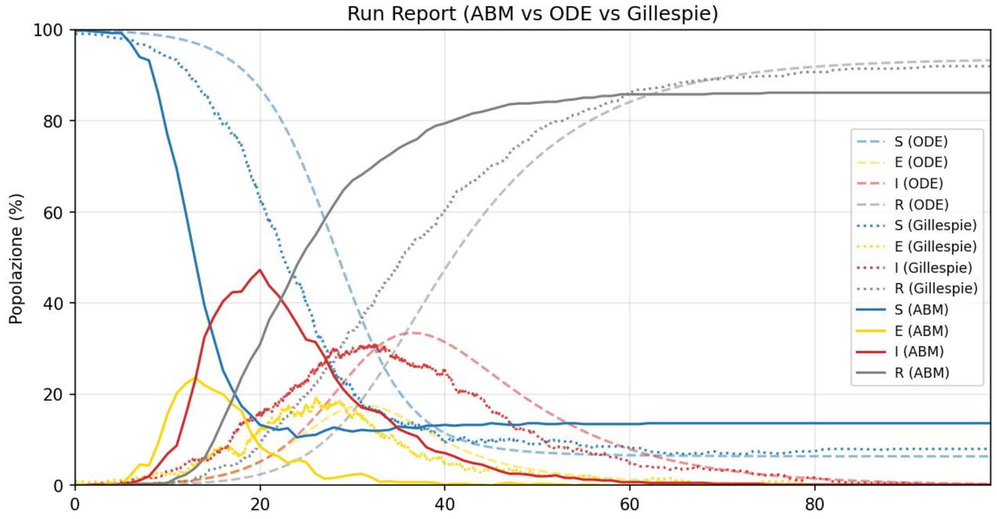


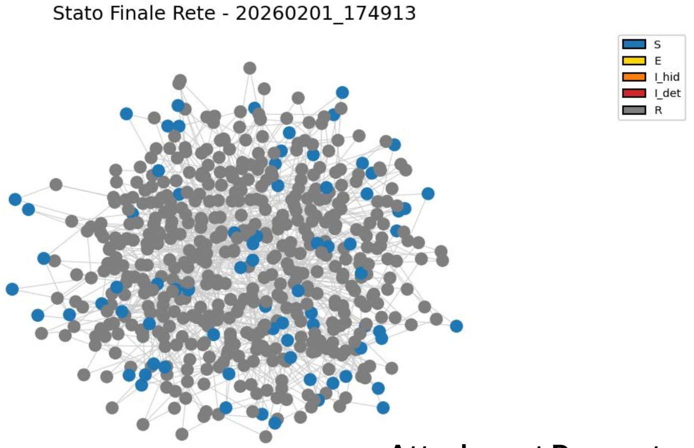


Neighbors: 4
Rewiring Probability: 0.1 (Chance of reconnecting an edge to a random node).

Results (Scale-Free)

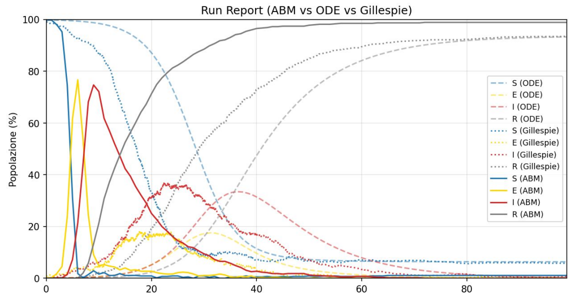
Mortality: ABM: 27.82%, ODE: 33.70%, Gillespie: 30.91%





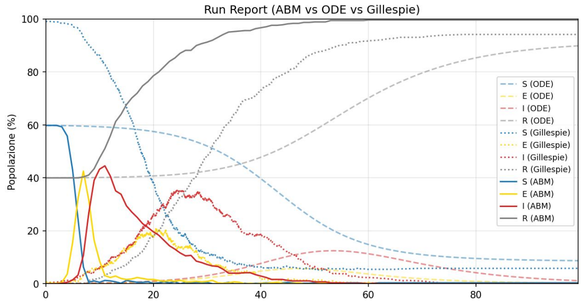
Attachment Parameter: 2

Results Vaccination Strategy Comparison



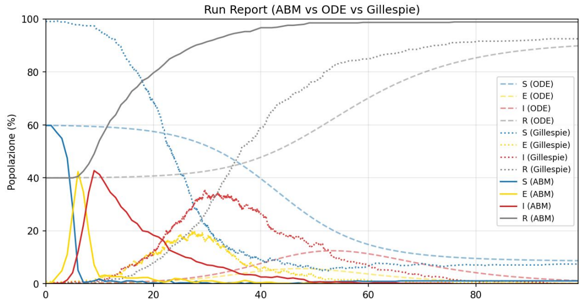
No Vaccination Scenario

Mortality: ABM: 40.73%, ODE: 33.70%, Gillespie: 33.27%



Random Vaccination (40% Coverage)

Mortality: ABM: 10.18%, ODE: 18.06%, Gillespie: 37.27%

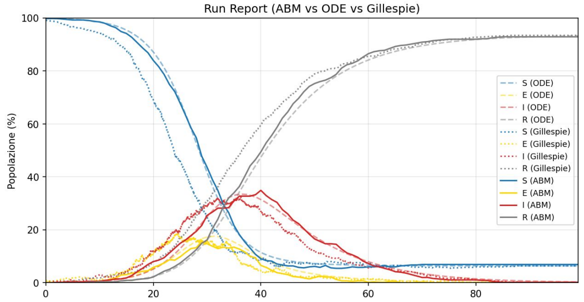


Targeted Vaccination (40% Coverage)

Mortality: ABM: 10.36%, ODE: 18.06%, Gillespie: 30.36%

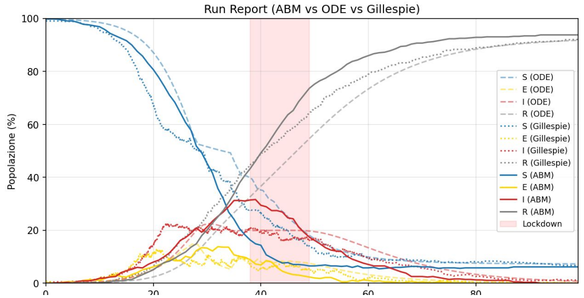
- **Erdos Renyi**

Results Lockdown & Vaccination.



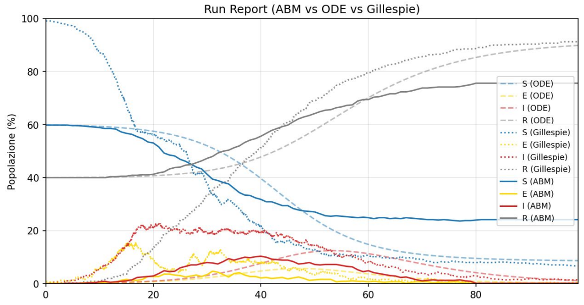
No Intervention Scenario

Mortality: ABM: 26.00%, ODE: 33.70%, Gillespie: 35.09%



Lockdown Intervention

Mortality: ABM: 19.45%, ODE: 32.97%, Gillespie: 31.45%

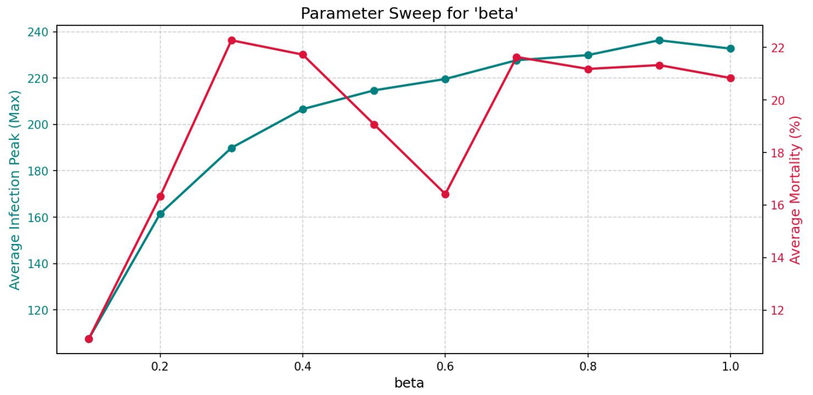


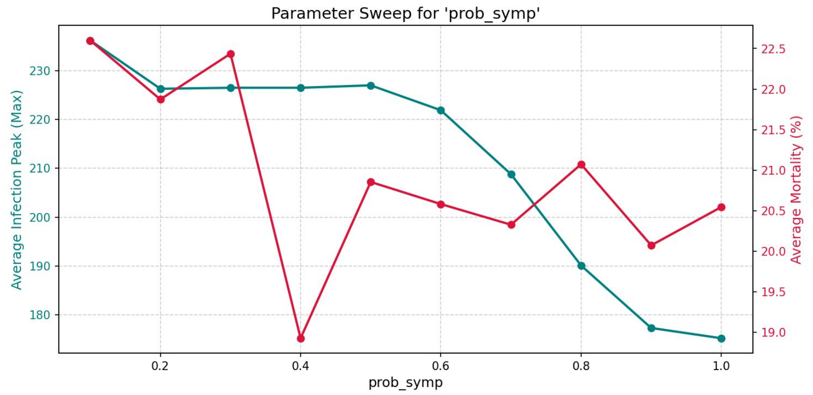
Combined Intervention Strategy

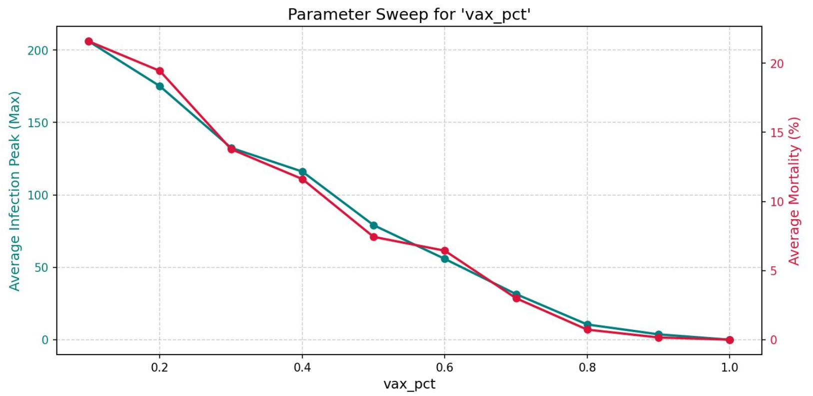
Mortality: ABM: 4.55%, ODE: 18.05%, Gillespie: 31.27%

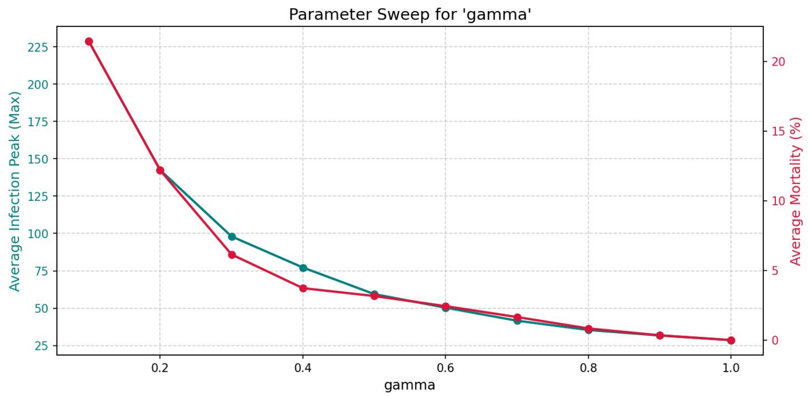
- **Small World**

Results sweep

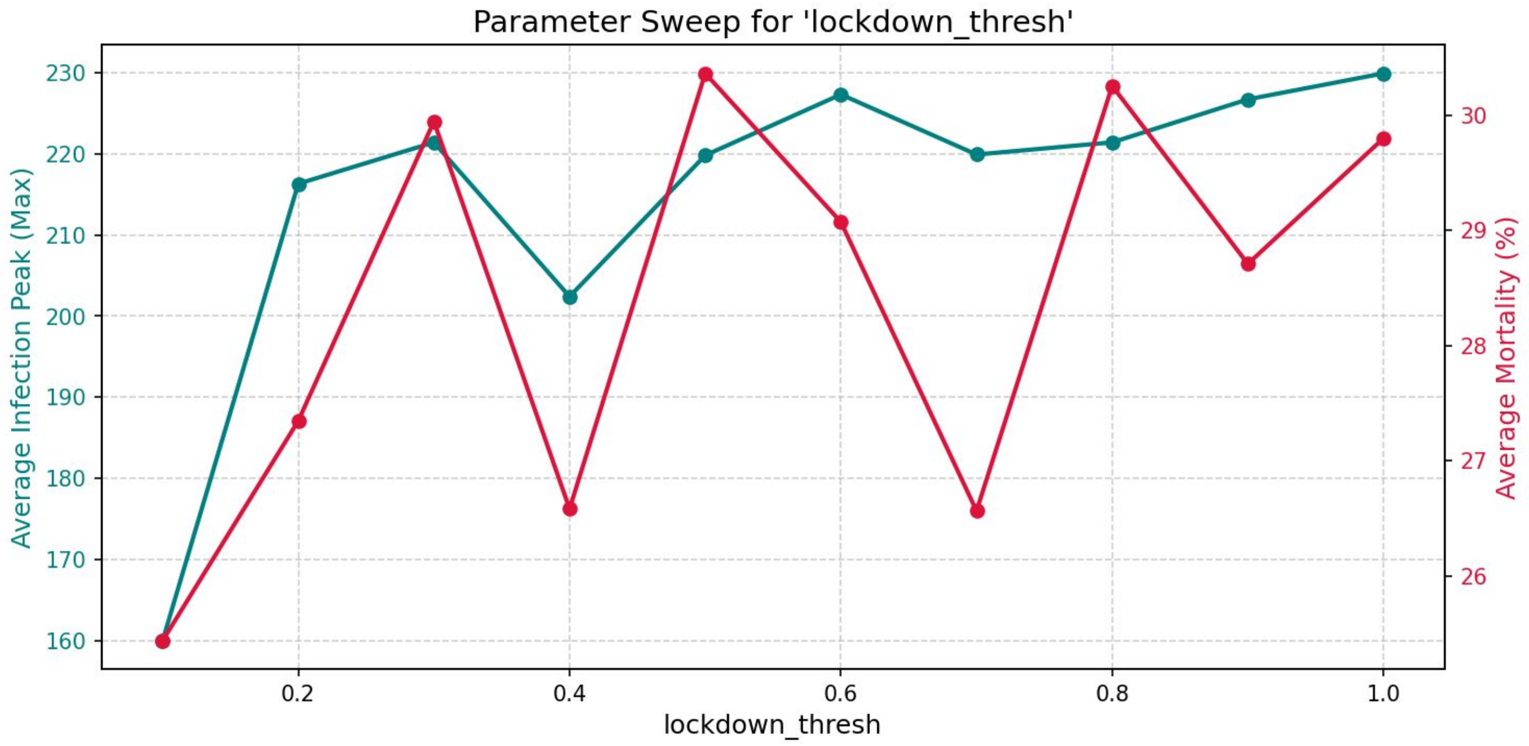






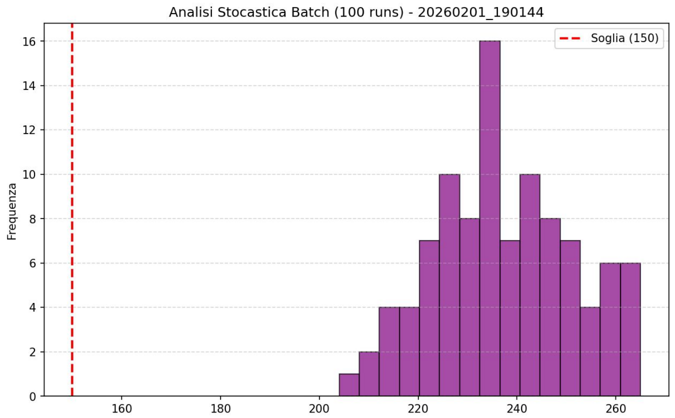


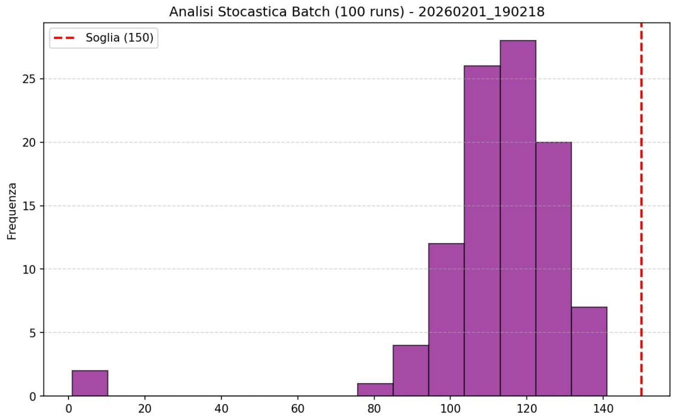
Results sweep



Results batch

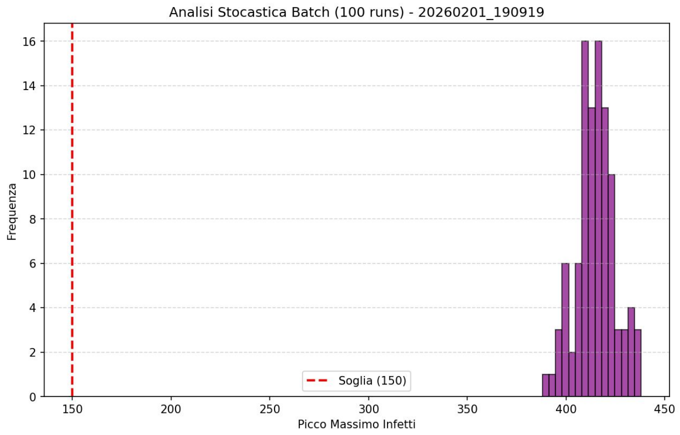
- **Left**: Baseline Batch Results
- **Right**: Combined Intervention Results (Vaccination + Lockdown)

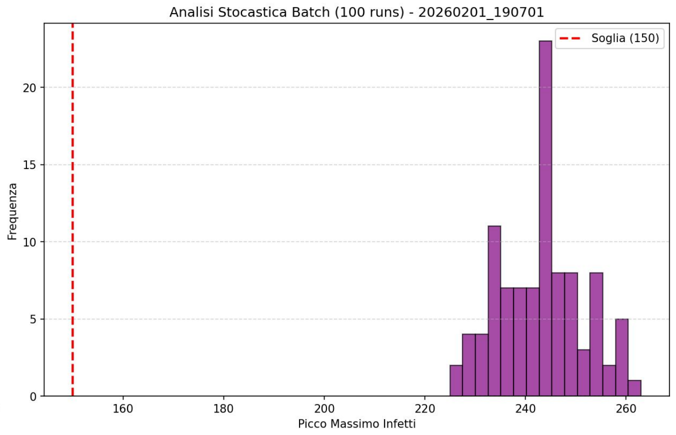




Avg. Mortality: 28.89% | Avg. Mortality: 11.96%

- **Grid**

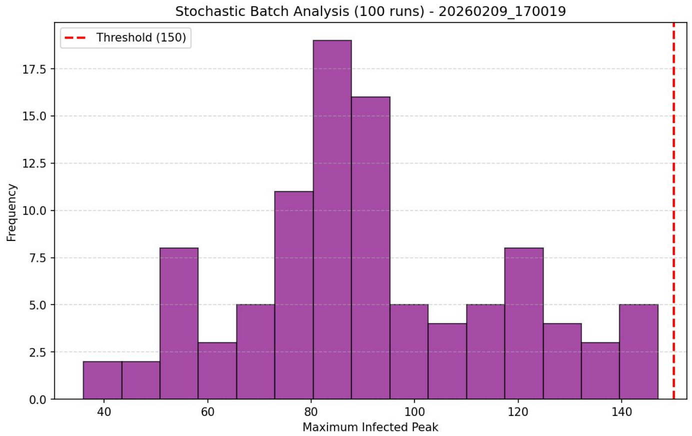


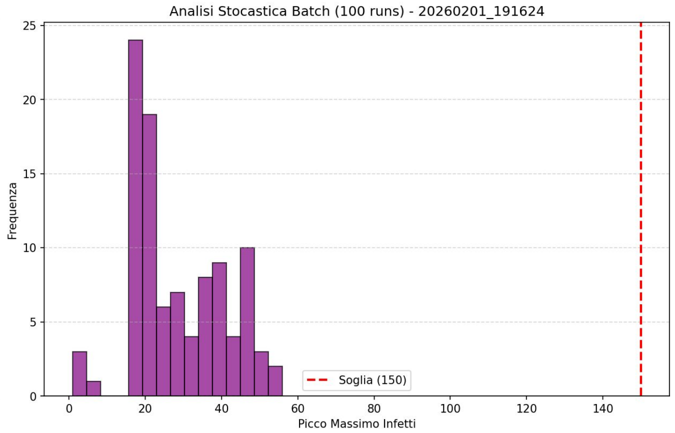


Avg. Mortality: 22.09% | Avg. Mortality: 13.02%

- **Erdos Renyi**

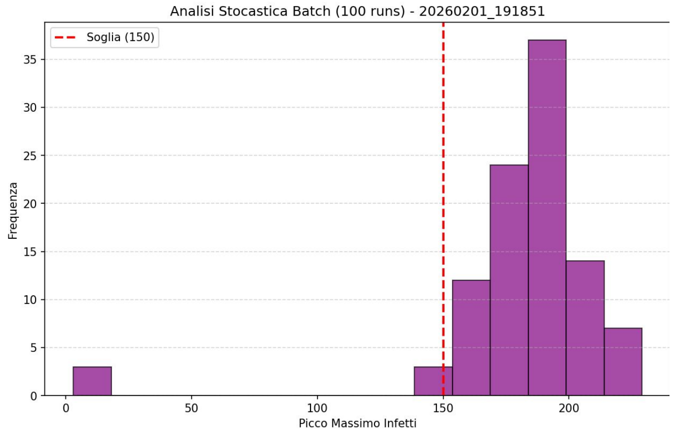
Results batch

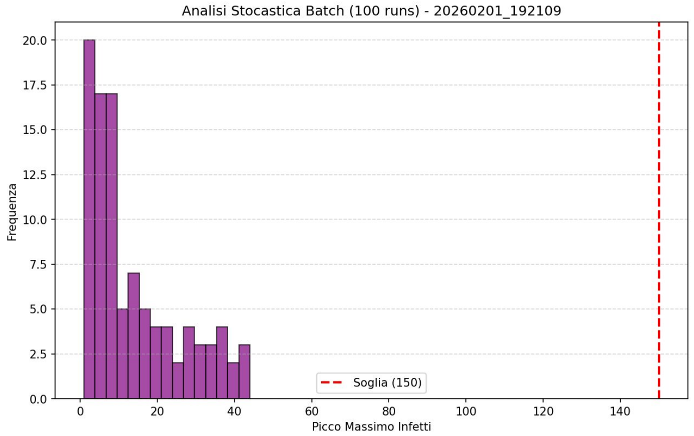




Avg. Mortality: 15.69% | Avg. Mortality: 2.21%

- **Communities**

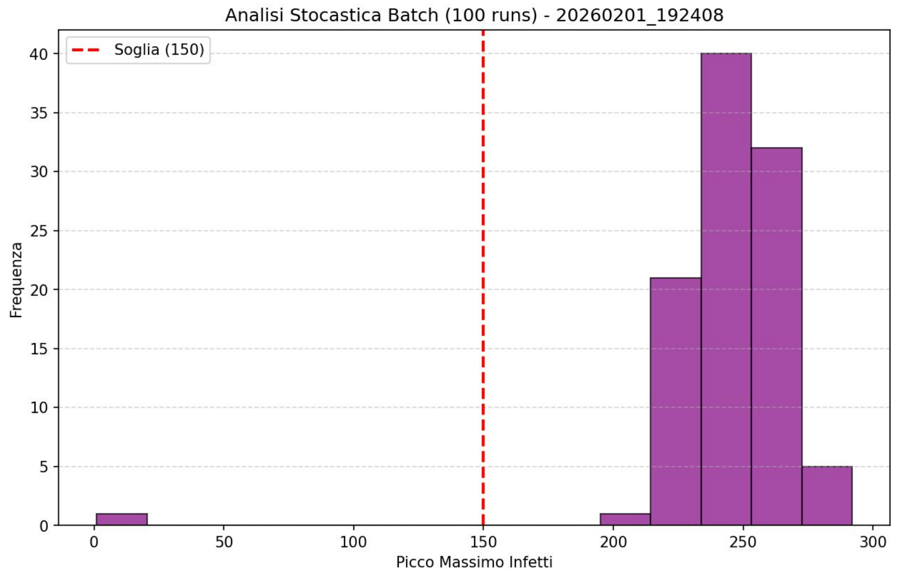


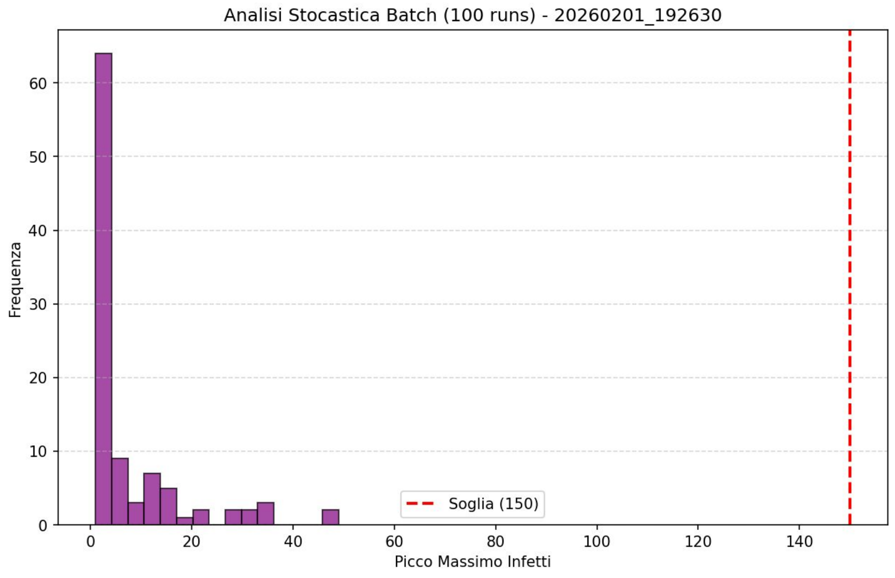


Avg. Mortality: 19.67% | Avg. Mortality: 1.59%

- **Small World**

Results





Avg. Mortality: 17.54% | Avg. Mortality: 0.19%

- **Scale-Free**

Demo

Limitations & Future Work

Technical Architecture:

- **Current State:** The environment is passive; complexity resides entirely in Agents.
- **Why not PropertyLayer?** Only necessary for cell-specific traits (e.g., local pollution, static transmission risk maps), which are not currently modeled.

Model Limitations:

- **Simplified Mobility:** Agents move randomly or strictly along network edges, lacking realistic commuting patterns (e.g., Home $\leftrightarrow$ Work).

Future Directions:

- **Calibration:** Fitting β and γ parameters to real datasets(e.g., COVID-19 or Influenza time-series).
- **GIS:** Integration with Mesa-Geo for real-world geospatial simulation.
- **Optimization:** Integration with Mesa-Frames to accelerate the agent loop for larger populations ($N>10{,}000$).

References

- Compartmental models in epidemiology – Theoretical overview on Wikipedia.
- Mesa: Agent-based modeling in Python (2015)– Research paper via CiteSeerX.
- Mesa: Agent-based modeling in Python – Official scientific publication on JOSS.
- Mesa Framework (Official Documentation) – Core documentation for Python ABM.
- Mesa GitHub Repository – Source code and community development.
- Mesa-Geo Documentation – Extension for GIS and geographical data analysis.
- Mesa-Frames GitHub – High-performance extension for Mesa.
- Matplotlib User Guide – Standard library for data visualization in Python.
- NetLogo Home Page – Multi-agent programmable modeling environment.

Questions?

Thank you.