Including Assets in an ABM leads to emergent specialization behavior

Group 16: Nitai Nijholt (12709018), Calvin Law (11306734), Dan Dong (14989093), Jasper Timmer (12854328), Zijian Zhang (14851598)

***Abstract*—**

This project uses an Agent-Based Model (ABM) to investigate emergence of specialized agent behavior when assets that provide income over time and wealth heterogeneity are introduced. As secondary research question, we simulate the effects of static and redistributive dynamic tax policies on agent behaviors and wealth distribution. Agents, possessing assets like houses that generate income, can gather resources and engage in buying, selling, and building houses, which provide increased income when clustered. By contrasting Expected Value Maximizing agents with those using Neural Networks evolved through Neuroevolution, the study examines emergent behaviors in response to different tax policies, reflecting initial wealth disparities in the Netherlands. Findings show that agents exhibit emergent behavior by forming distinct economic classes: a wealthy building class, a low-activity poor class, and either a middle class (for Expected Value Maximizing agents) or a trader class (for Neuroevolution agents). Sensitivity analysis reveals that parameters related to the number, cost, and income of assets significantly influence the persistence of these classes. We find significant differences in Gini coefficients for wealth distribution across tax policies for expectation-maximizing agents, though these differences have a low effect size. Neuroevolution agents showed lower Gini coefficients and productivity, and neither agent's income distribution aligned with the Netherlands' data under the Dutch tax policy. The main contribution of this study is finding emergence of specialized building behavior among different agent intelligence types and tax policies, with initial wealth as the primary form of heterogeneity. As wealth is easier to measure than skill, our model is easier to parameterize while still showing the emergence of building behaviors found in other research employing skill heterogeneity. Future research should explore the impact of more aggressive redistributive tax policies on inequality and examine how class persistence changes through the parameter space by examining pairwise cluster feature differences of classes instead of the total Euclidean distance between the cluster feature vectors.

I. Introduction

Understanding the impact of tax policies on economic behavior and income distribution is a critical area of research in economics. Traditional economic models often fall short in capturing the complex and adaptive nature of human behavior in response to such policies [3]. To address these limitations, Agent-Based Models (ABMs) have emerged as a powerful tool for simulating and analyzing the interactions within an economy. By accounting for agent heterogeneity and complex local interactions, ABMs distinguish local behaviors that are not possible to capture in mean field approaches. In this way, analyzing ABMs can provide candidate explanations for the behavioral mechanisms that lead to economic outcomes of interest. In our case, identifying these behavioral mechanisms is relevant as we investigate the resulting wealth distribution and the underlying behavior of agents who move to optimize their own wealth with regard to different tax policies.

In this project, we utilize an ABM to investigate the effects of static and dynamic tax policies on agent behaviors and income distribution. By focusing on the emergent behavior of agents, we aim to discern how variations in tax policies influence economic outcomes, particularly considering how the heterogeneity in initial wealth among agents affects wealth distribution. Our study is inspired by the work of Zheng et al. [10], who employed a two-stage AI-controlled ABM to simulate the effects of various US and Saez tax policies, along with an AI-optimized tax policy. Zheng et al. hardcoded agent skill heterogeneity into the model, operationalized as increased income from building or gathering. They observed specialization in classes, finding distinctions between buyers and builders, gatherers, and sellers within the agents. While this is an interesting result, we believe that by hardcoding the skill and thus the action efficiency differential into the agents, the behavior is less emergent than it might seem. We address this by giving our agents the same income for each action, mirroring the equality of opportunity policy objective as closely as possible. However, capitalistic economies do exhibit large individual wealth differences, which can affect the opportunities available to an individual. Therefore, we give agents heterogeneous initial wealth, sampled from the Dutch wealth distribution, and aim to investigate if the same specializing behavior still emerges.

We also add another feature: whereas Zheng's building a house provides a one-time income, our houses provide

income over time, mirroring asset returns in capitalistic economies. Zheng's paper also explores a couple of tax policies, including those of the US, Saez, and an AI-optimized approach. We opt for a simpler taxing approach, using the static tax policy of the Netherlands, and compare it with a dynamic tax policy aimed at achieving equality by adjusting tax brackets to minimize the standard deviation of the wealth distribution, effectively redistributing more as the wealth disparity grows. The advantage is that applying such a simple dynamic tax policy would be more feasible than an AI-optimized one; the downside is that this simplified tax policy does not maximize overall social welfare (the product of equality and productivity) as effectively.

We test two types of agent decision intelligence separately: wealth maximizing and agents controlled by a neural network evolved using Neuroevolution. By employing Evolutionary Computing techniques to evolve the Neural Networks as opposed to training them, we minimize required training time and computational cost while still being able to adapt relatively complex neural networks successfully to the problem at hand.

Combining all these aspects, we aim to study which behaviors of agents manifest in our model and if specialized behavioral classes emerge through an experiment where we classify behavior.

The rest of the paper is laid out according to the ODD format [1], covering important design concepts, their implementation details, experiments, analysis methods, and results, finalized by a discussion and conclusion.

II. Design Concepts

A. Overview

1) Purpose: The aim of our project is to study the effect of introducing several mechanisms into an Agent-Based Model (ABM), including rational decision-making based on expected value and Neuroevolution, income generation over time from assets, and the application of Dutch wealth and tax distribution data to study the emergent behavior of agents and the resulting wealth and income distributions. This leads to the following research questions:

RQ1: How does introducing an asset that provides income over time influence the emergent behavior classes of agents?

This is quantified in our hypothesis H0a:

H0a: We observe clusters with significantly different characteristics in buying and selling, gathering and selling, and buying and building behaviors.

RQ2: What is the effect of static vs. dynamic tax policy on agent wealth distribution?

This is quantified in our hypothesis H0b:

H0b: There is no significant difference in the Gini coefficient and total welfare between simulation runs with dynamic tax policies and those with static tax policies.

Validation: We will also validate the model by comparing obtained incomes to the Dutch income distribution, which will be quantified by:

H0c: There is no significant difference between the income distribution in the model and that in the Netherlands.

Finally, we will answer the question of which parameters affect emergent agent behavior, through local Morris's Elementary Effects One-Factor-At-a-Time (OFAT) [7] and global Sobol sensitivity analysis using SALib in Python [4] [5].

Our model is parameterized using wealth values and tax brackets from the Dutch economy.

The intended contribution is for agent-based model builders interested in the criteria for the emergence of specializing behavior, and for policymakers who are looking to study the effect of different tax policies in the presence of wealth heterogeneity (which is easier to measure than skill heterogeneity) and assets which provide income over time (a core feature of a capitalist economy).

B. Entities, State Variables and Scales

The model includes the following types of entities and is characterized as follows:

- **Grid**: The spatial environment where agents operate, gather resources, trade, and build houses.
- **Agents**: Individuals who can gather resources, trade, or build houses to generate income over time. Initiated with an initial lifespan sampled from a gamma distribution but with a set mean and a starting wealth based on the wealth distribution from the Netherlands, obtained from the "Centraal Bureau van de Statistiek" [1].
- **Resources**: Items such as wood and stone that agents can gather to create houses or trade. These are spawned randomly on the grid with an initial set spawning rate.
- **Houses**: Structures that agents can build, which contribute to their income each timestep.
- **Market**: A dynamic market in which agents can trade via an orderbook using buy and ask orders (non local, can be accessed from anywhere on the grid).
- **Tax Policy**: A static or dynamic tax system that taxes agents' recevied income from trading and houses.

Agents have the following attributes:

- Unique identifier of the agent.
- Position of the agent on the grid.
- Time when the agent was created.
- Initial wealth of the agent drawn from the wealth distribution of the Netherlands.
- Initial number of wood resources of the agent.
- Initial number of stone resources of the agent.
- Agent's expectation of its own lifetime (options: 'gamma', 'lognormal').
- List of owned Houses.

The exogenous factor affecting the agents is the tax policy.

C. Spatial and Temporal Aspects of the Model

The spatial domain is represented by a 2D $N \times N$ grid with periodic boundaries. The grid spacing is uniform in both the x and y directions, given by $\Delta x = \Delta y = 1$. For a grid point (i, j), the periodic boundary conditions are defined as:

$$u(i + N, j) = u(i, j) \quad \text{and} \quad u(i, j + N) = u(i, j)$$

The temporal extent of the simulation spans 1000 timesteps with a uniform timestep size of $\Delta t = 1$.

D. Design Concepts

1) General Concepts and Theories: The general idea is to model a capitalist economy by introducing several core factors such as working to earn income (operationalized as gathering and selling resources) and purchasing assets that generate income (such as gathering or buying resources to build a house which then provides income). Agent heterogeneity include lifetime expectation differences and wealth heterogeneity, which is can affect equality of opportunity. Different tax policies (static and dynamic) are also considered and agents try to optimize their individual wealth taking into account these taxes.

The model simulates agents managing resources (wood and stone), making decisions about gathering, buying, selling, and building based on calculated earning rates, aiming to choose the action that increases wealth the most. This reflects principles of resource allocation and utilization.

Agents can trade on a market. The use of an order book mechanic for wood and stone mirrors real-world market mechanisms where prices are influenced by current supply and demand. Agents' buy and sell orders determine market prices, reflecting economic theories of market equilibrium and price formation.

Agents choose actions that maximize their wealth by selecting the action that maximizes their Expected Value (EV) per timestep. Agents are fully rational but have limited information available (only the prices at the previous timestep and the entities on the grid in a Von Neumann neighborhood around them). This model is based on the assumption that humans, like our agents, are fully rational but have limited information available (which, although a simplification with respect to reality, suffices for our purpose).

With these rules at the agent level decision making, we expect complex behaviors at the system level to emerge, with distinct agent behavioral classes.

2) Empirical Data and Level of Aggregation in the Model: For the ABM, we incorporated empirical data on a few levels. On the global level, we have two tax structures: static and dynamic, respectively. The realization of these two tax policies is based on the paper *The AI Economist* [10], but the tax rates and brackets are set referring to the current tax rates of the Netherlands. On the individual level, agents wealth is sampled from the wealth distribution of the Netherlands. Income distribution is validated [6] against the income distribution found in the Netherlands [9].

The level of aggregation the wealth data was available is household level, in our model we consider each agent to have the wealth of a single household.

Wealth is regarded on a per-agent level by simply observing the amount they have at the final timestep when they die. The wealth distribution aggregates all the wealth of the agents into a single distribution. In the section on tax policy, additional metrics which aggregate agent welfare into scalar metrics will be discussed.

3) Individual Decision-Making: We experiment using two different agent types: Expected Value Maximizing and Neuroevolution, which are described in the following sections II-E and II-F. It's important to note that we use these types of agent intelligence in separate simulations. This means that the EV agents' runs are conducted independently from the Neuroevolution runs, and the different agent types do not compete within a single simulation.

Before moving onto the agent decision intelligence description, the update sequence for agents which applies to both agents types is described.

4) Agent update sequence: In this subsection the agent update sequence of agents will be described. Note that agent attributes are described in section II-B

The agents is then updated as follows:

1) Collect fixed income at the beginning of each timestep.
2) Continue construction if currently building a house:
   a) Increment construction progress.
   b) Complete the house if required building time is reached.
3) Evaluate potential actions if not building:
   a) Calculate potential earning rates for actions: move, build, buy, sell, gather.
   b) Select the action with the highest earning rate.
4) Place market orders if buying or selling, considering order limits:
   a) Check current market prices.
   b) Calculate marginal prices based on expected income and building costs.
   c) Place buy orders at or below market price.
   d) Place sell orders at or above market price.
5) Perform the selected action and update agent's state.
6) Track metrics over time: wealth, number of houses, gathered resources, market transactions.
7) Monitor agent's lifespan and terminate if exceeded:
   a) Consolidate and record data.
   b) Remove agent from grid and delete market orders.
   c) Create a new agent to maintain population.

TABLE I: Update Sequence for Agent

E. Agent Type 1: Expected Value (EV) Based Decision Making

In the EV-based decision-making approach in this ABM model, individual agents make decisions by choosing actions that maximize their income per timestep, focusing on economic activities such as building, buying, selling, gathering, and moving. Agents consider their local environment (Von Neumann Neighborhood), personal state attributes, and market conditions (previous step prices) to calculate expected earning rates and choose the highest earning actions. Decisions are based on optimizing the expected value of outcomes, taking into account resource availability, market prices, and personal wealth. Agents adapt their behavior based on these factors, without social norms or cultural values influencing their choices. Spatial aspects are included as agents gather resources from specific grid locations and move to neighboring cells, while temporal aspects consider the agents' remaining lifetimes, modeled as random draws from a gamma distribution. Uncertainty arises from the stochastic nature of resource availability and initial wealth endowment.

Net extra income indicates the income after performing an action, after taking into account the impact of taxes. It is represented by the following formula:

$$\text{Net Extra Income} = [z_0 + \text{EV} - T(z_0 + \text{EV})] - [z_0 - T(z_0)] \tag{1}$$

where z_0 is the original income, EV is the expected additional income from the actions, and $T(\cdot)$ is the tax function. This formula calculates the change in income before and after considering taxes. For actions deemed impossible, $\text{EV} = -\infty$.

Earning rate refers to the expected income per unit time from building a house. It is represented in Equation 2, where the earning rate depends on the number of houses in its Von Neumann Neighbors, implying that more nearby houses increase the income a newly built house will bring.

$$r(i,j) = n_v + 1 \tag{2}$$

where $r(i,j)$ is the earning rate for a house in position (i,j), and n_v is the number of houses in the Von Neumann Neighbors of position (i,j).

The function of the expected value of building a house evaluates the profitability of building a house. Its mathematical formula is represented in Equation 3. The decision is based on available space on the grid, the earning rate, and the estimated remaining life of the agent after accounting for the time required to build. If resources are insufficient or the maximum house limit is reached, the function returns a prohibitive value. Otherwise, it computes the expected value based on the earning rate of building position (i,j) and the estimated remaining life, divided by the time required to build. The results are adjusted for taxation, so that agents take into account taxes into their decision making.

$$\text{EV}_{\text{building}}(i,j) = \frac{r_{building}(i,j) \cdot (\tilde{t_{\text{life}}} - a - t_{\text{building}})}{t_{\text{building}}} \tag{3}$$

where $r_{building}(i,j)$ is the earning rate of building a house in position (i,j), $\tilde{t_{\text{life}}}$ is life expectancy, a is the agent's age, and t_{building} is the time required to build a house.

The function for the expected value of buying resources necessary to build a house, and the mathematical

formula is represented in Equation II-E. It calculates the cost of acquiring the needed resources and checks if the purchase is feasible with the agent's current wealth and market availability. If feasible, it computes the gross income expected from building the house, deducts the cost, and adjusts the result for the total time needed (including time to buy resources and building a house).

$$\mathrm{EV}_{\mathrm{buying}}(i,j) = \frac{r_{building}(i,j) \cdot (\tilde{t}_{\mathrm{life}} - a - (t_{\mathrm{building}} + 1)) - \mathrm{cost}}{t_{\mathrm{building}} + 1} \tag{4}$$

The function of expected value for selling resources, as represented by Equation 5, calculates the potential income from selling all held wood and stone at the current market price and adjusts for tax implications. If no resources are available, it returns a prohibitive value.

$$\mathrm{EV}_{\mathrm{selling}}(i,j) = r_{\mathrm{wood}} \cdot n_W + r_{\mathrm{stone}} \cdot n_S \tag{5}$$

where r_{wood} and r_{stone} are the prices of wood and stone, respectively, and n_W and n_S are the amounts of wood and stone owned by an agent.

Agents can also opt to gather resources directly from their current position on the grid. Equation 6 calculates the potential earnings from gathering, based on available resources at the position and their current market price, adjusted for any taxes.

$$\mathrm{EV}_{\mathrm{gathering}}(i,j) = r_{\mathrm{wood}} \cdot \min(1, n_{W_{ij}}) + r_{\mathrm{stone}} \cdot \min(1, n_{S_{ij}}) \tag{6}$$

where $n_{W_{ij}}$ and $n_{S_{ij}}$ are the amounts of wood and stone at position (i,j), respectively.

Equations 7 and 8 evaluate the benefit of moving to a new position, with two potential strategies: moving to build or moving to gather. If the new position allows for building (with sufficient resources and space), it calculates the expected income from building a house there, considering the time and resource requirements. If building isn't viable, it assesses the value of gathering resources at the new position, taking into account the travel and gathering time.

$$\mathrm{EV}_{\mathrm{moving,\ gather}}(i,j) = \frac{r_{\mathrm{wood}} \cdot n_{W_{ij}} + r_{\mathrm{stone}} \cdot n_{S_{ij}}}{\min(n_{W_{ij}}, n_{S_{ij}}) + 1} \tag{7}$$

$$\mathrm{EV}_{\mathrm{moving,\ build}}(i,j) = \frac{r(i,j) \cdot (\tilde{t}_{\mathrm{life}} - a - t_{\mathrm{building}} - 1)}{t_{\mathrm{building}} + 1} \tag{8}$$

Overall, the EV-based decision-making method utilizes a combination of current resource availability, market conditions, and the agent their personal circumstances, like their remaining estimated life and wealth, to calculate the most beneficial actions in an environment defined by both stochastic elements through the orderbook and interactions with the environment.

F. Agent Type 2: Neuroevolution Decision Making

As an alternative to the expected value equations detailed in section II-E, agent actions have been determined using a neural network. A network was designed with the same number of inputs as all the expected value equations combined in section II-E, producing outputs for the actions: *Start building*, *Buy*, *Sell*, *Gather*, *Stay* and *Move* to any cell in the agent's Von Neumann neighborhood. To facilitate the emergence of complex strategies, two hidden layers with 40 neurons each were incorporated into the network.

Training these networks for all individual agents in the simulation becomes unfeasible quickly with large numbers of agents. Therefore, these networks were not trained in the conventional sense but were evolved using a technique known as *Neuroevolution*.

Neuroevolution is a subfield of evolutionary computing where concepts like survival of the fittest and natural selection are applied to algorithmic problem-solving.[2] Solutions to a problem are recombined into new ones, where better solutions tend to have a higher chance to reproduce. Over time, poorly performing solutions will die out, as better-performing solutions reproduce more frequently, optimizing the solutions to best meet the problem requirements, based on defined performance metrics.

G. Employed Algorithms

A problem can be addressed with an evolutionary algorithm if the solutions can **be numerically represented** and their performance evaluated using a well-defined fitness function. Generally, an evolutionary algorithm follows this procedure:

1) Initialization
2) Repeat until termination:
   a) Parent selection
   b) Recombination
   c) Mutation
   d) Survivor selection

TABLE II: Update Sequence for the Evolutionary Algorithm

a) Representation: In our experiments, the goal is to evolve neural networks that provide agents with strategies to maximize their wealth over time. The agents, or rather solutions in this case, are represented by a neural network, specifically the weights and biases, as the architecture is predefined and immutable.

b) Fitness: Fitness is defined as the agent's wealth at the end of its lifetime minus its wealth at birth:

$$F^i = W_t^i - W_0^i, \tag{9}$$

where F^i is the fitness of agent i, and W_t is its wealth at time-step t.

c) Initialization and Termination: To simplify the algorithm, all agents of a generation spawn and die simultaneously. Initially, all agents are assigned networks with randomized weights and biases. The simulation ends after a predefined number of generations.

d) Parent Selection: Parent selection determines which pairs of agents reproduce to form new agents. The algorithm used is tournament selection, which involves randomly selecting a group of k individuals from the population and choosing the fittest among them. Hence, the tournament algorithm is executed twice per pair of agents. Selection pressure depends on k; a $k = 1$ results in random selection, while k equal to the population size results in all selected individuals being copies of the fittest individual in the population. For $k > 1$, better-performing agents have a higher chance of reproducing, mimicking natural selection observed in biological systems. However, since the k selected individuals might all perform poorly, this algorithm allows sub-optimally performing agents to reproduce, maintaining high diversity in the agents' networks. In our simulation, $k = 3$ is chosen, and each selected pair of agents produces two new agents as offspring.

e) Recombination: Crossover: To combine the networks of two parents into children, also referred to as crossover, we implemented line recombination. This approach involves connecting coordinates in the N-dimensional space representing the neural network of individuals. Each agent is represented by values for the weights and biases in their neural network, all within [-1, 1]. If two individuals reproduce, an imaginary line is drawn between those coordinates, connecting the parents in this hyper-dimensional space. In line recombination, each child receives a network which, in that hyper-dimensional space, lies on a randomly chosen point on that connecting line. This can be mathematically expressed as:

$$x_c = x_{p1} + \alpha(x_{p2} - x_{p1}), \tag{10}$$

where x_c, x_{p1}, x_{p2} are the coordinates of the child, parent 1, and parent 2, respectively, and α is a random uniform value between [-0.25, 1.25]. If $\alpha = 0$, the child will be an exact copy of x_{p1}, a value of 1 results in an exact copy of x_{p2}. If α lies between 0 and 1, the values of the weights will also lie between the values of those weights in the parents. If α lies outside of the [0, 1] range, the weights of the child will also lie outside the direct connection between x_{p1} and x_{p2}, but on an extension. This is to prevent the weights from gradually approaching zero.

f) Mutation: Each weight and bias in the newly created agent's network has a probability of $P = 0.001$ to be affected by mutation. When a mutation event occurs, the weight is assigned a new value, sampled from a uniform distribution between $[-1, 1]$.

g) Survivor Selection: Survivor selection involves deciding which agents survive to the next generation. As the fitness of our agents is highly dependent on the length of their lifetime, the decision was made to replace the 100% of the parent population of agents with their children each generation.

H. The Process of Evolving Agents

To expedite the process, instead of always selecting the output of the network with the highest value, the corresponding action is chosen only if it is legal. If the highest value action is not legal, the second highest is considered, and so on. Therefore, for the first generation, most actions of the agents will be chaotic and without clear strategy. After a certain number of timesteps, at the end of this generation, the fitness of all agents is calculated and used for parent selection. Due to the randomness of actions, agents tend to have low fitness; however, some will have higher fitness than others. Agents who manage to build a house early in the simulation will likely have accumulated the most wealth, thus having a higher probability of passing their network to the next generation. The same applies to agents that gathered resources and sold them on the market without building a house, or those who bought resources when prices were low and sold them when prices were high, although these scenarios are less likely in early generations.

1) Individual Learning: In terms of learning, there is a distinction between expected value maximizing agents, who do not learn, and Neuroevolution agents, who do learn. Expected value maximizing agents follow predefined rules and evaluate actions based on static parameters such as earning rates and market prices. Consequently, their decision-making process does not evolve dynamically with experience, and collective learning is not implemented for these agents. In contrast, for

Neuroevolution agents, learning occurs each generation; the information encoded in the weights of the fittest networks is passed on to the next generation through the previously described process. Learning, therefore, happens on a collective level, as individuals do not improve within their own lifetimes, but populations of individuals become more effective over time.

2) Individual Sensing & Prediction: Individuals are assumed to sense and consider both endogenous state variables, such as their own wealth, wood, stone, and current position, as described in II-B. They also consider exogenous state variables, including market prices for wood and stone, resource availability on the grid around them (Moore neighborhood), and potential earnings from different actions as described in II-E. Agents do not perceive the state variables of other agents; their decisions are based solely on their own state and market conditions. The spatial scale of their sensing is limited to their immediate position and neighboring cells for resource gathering. The mechanisms by which agents obtain information are not explicitly modeled: individuals are simply assumed to know these variables within their bounds. Costs for cognition and information gathering are not included in the model, as the focus is on decision-making based on predefined rules and available data.

Agents predict their expected value based on the actions available to them at a given timestep. They use the best prices on the orderbook or their predicted lifespan to determine buying and selling prices. Expectation-maximizing agents estimate their maximum age to calculate the expected earning rate from actions. For instance, building a house near the end of their life yields less wealth compared to building it early in their lifetime, as houses provide a set income per timestep. Additionally, agents calculate the expected value of building houses by considering the timesteps required for construction and their estimated lifespan.

Agents may make erroneous predictions about their lifetime due to limited information, as both the actual lifetime and their estimate are random draws from a gamma distribution. They only know the market price based on the order book's bids and asks at the previous timestep, $t-1$. They are also unaware of other agents' buying and selling intentions. Furthermore, agents can only see one grid space around them, preventing them from having perfect information about the entire grid. These limitations can lead to suboptimal decisions and model the limited information availability.

I. Interactions

Agents' interactions depend on several factors, including their current wealth, resource holdings, and income. These interactions occur indirectly through the order books of the market by submitting bids and asks, potentially matching with other agents' bids and asks. Income gained per timestep from a house increases by one for every other house in the Von Neumann neighborhood of that house. Additionally, when agents gather resources from the grid, those resources are no longer available to others. Our simulation does not include an explicit coordination network. Preliminary experiments also observed emergent clustering of houses, which could be explained by the fact that houses generate more income when located near other houses. Although this phenomenon warrants further investigation, it falls outside the scope of this project.

J. Tax Policy

The primary goal of the tax policy is to balance social equality and productivity to enhance social welfare. This involves defining income brackets and setting tax rates for each bracket to maximize total welfare. Revenue from taxes is used to redistribute wealth among taxpayers. High tax rates can demotivate taxpayers, as evidenced by agents' rational behaviors, where lower future income forecasts reduce work motivation. Consequently, agents optimize their actions based on taxes and redistribution, adjusting as the tax schedule changes.

Agents pay taxes based on a schedule $T(z,\tau)$, which calculates taxes using income z and marginal tax rates τ. The income from activities such as selling resources and building houses served as pretax income z to calculate taxes. Tax brackets are set to match the corresponding tax rates τ, adjustable based on changes in agents' income.

The tax policy employs a progressive tax system based on income, with higher rates for higher income ranges. Taxes on income z are computed as follows:

$$T(z) = \sum_{j=1}^{B} \tau_j \cdot ((b_{j+1} - b_j) 1\left[z > b_{j+1}\right] + (z - b_j) 1\left[b_j < z \leq b_{j+1}\right]) \tag{11}$$

where B denotes the number of income brackets, and τ_j and b_j correspond to the marginal tax rates and income limits for each bracket, respectively.

An agent's pretax income z_i for a given tax period is calculated as the change in their wealth C_i from the beginning of the period. Therefore, taxes are collected by deducting $T(z_i)$ from C_i at the end of each period. Collected taxes are redistributed evenly among all agents. Consequently, at the end of each period, an agent's wealth changes according to:

$$\Delta C_i = -T(z_i) + \frac{1}{N} \sum_{j=1}^{N} T(z_j) \tag{12}$$

where N is the total number of agents.

The equality of the society with an agent population N at time t is defined as:

$$\text{eq}(\boldsymbol{C}_t) = 1 - \frac{N}{N-1}\text{gini}(\boldsymbol{C}_t),\ 0 \leq \text{eq}(\boldsymbol{C}_t) \leq 1 \quad (13)$$

where $C_t = (C_{1,t}, \quad ..., \quad C_{N,t})$, and the Gini index is calculated as:

$$\text{gini}(\boldsymbol{C}_t) = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} \mid C_{i,t} - C_{j,t} \mid}{2N\sum_{i=1}^{N} C_{i,t}},\ 0 \leq \text{gini}(\boldsymbol{C}_t) \leq \frac{N-1}{N} \quad (14)$$

Productivity is defined as the sum of all agents' wealth:

$$\text{prod}(\boldsymbol{C}_t) = \sum_i C_{i,t} \quad (15)$$

The economy is assumed to be closed, and the collected tax is evenly divided and redistributed to every agent. No tax money leaves the system, and the total wealth of the society is only influenced by the agents' income, which further impacts the welfare of the society at time t. The policy planner adjusts the tax policy to optimize social welfare. For the objective of optimizing a trade-off between equality and productivity, the social welfare is defined as the product of equality and productivity.

$$\text{swf}_t = \text{eq}(\boldsymbol{C}_t) \cdot \text{prod}(\boldsymbol{C}_t) \quad (16)$$

The objective of the tax policy planner is to maximize expected social welfare:

$$\max_{\pi_p} \mathbb{E}_{\tau\sim\pi_p, a\sim\pi}\left[\sum_{t=1}^{H} \gamma^t r_{p,t} + \text{swf}_0\right], r_{p,t} = \text{swf}_t - \text{swf}_{t-1} \quad (17)$$

where swf_t is the social welfare at time t, balancing equality and productivity. $r_{p,t}$ represents the reward received by the planner at time t, typically reflecting the social benefits or costs under the chosen tax policy. And γ is the discounted factor, which influences the long-term rewards. $\tau \sim \pi_p$ indicates that the tax rates τ are chosen according to the policy π_p, which is used to determine optimal taxation rates. $a \sim \pi$ refers to actions a taken by the agents, which are selected according to their action policy π.

1) Tax Rates and Brackets: The tax rates and brackets used in the simulation are based on the income data distribution of Dutch people in 2022 and the brackets for wage tax/national insurance contributions in 2023. The tax brackets are set as percentiles, which have the values of [5.12, 65.67, 96.93, 100], and the corresponding tax rates are set as [0, 0.1903, 0.3693, 0.4950]. Both are aligned with the actual tax data in the Netherlands.

2) Difference between Static and Dynamic Tax Policies: The basic mechanisms of static and dynamic tax policies are similar; both aim to regulate the wealth of individuals in the economy through taxation and redistribution, promoting social welfare and economic equality. In the static tax policy, the tax brackets are divided only according to income percentiles, with a corresponding fixed tax rate applied to each bracket. The tax rates do not adjust for income fluctuations.

In contrast, the tax rates in a dynamic tax policy are adjusted dynamically based on the standard deviation of income. When the standard deviation of pretax incomes σ exceeds a certain threshold θ, the adjustment factors α modify the tax rates τ so that high-income earners face a higher tax rate and low-income earners face a lower tax rate, thereby balancing the income gap. The corresponding mathematical formula is as follows:

$$\tau_j' = \max(0, \tau_j + \alpha_j) \quad (18)$$

where τ_j is the base tax rate, τ_j' is the adjusted tax rate, and α_j is the adjustment factor α for each tax bracket j.

In theory, this dynamic adjustment mechanism makes tax policies more adaptable and responsive to changes in income distribution, thereby more effectively achieving income redistribution and improving social welfare.

K. Market

1) Orderbook: The market implemented in the simulation mirrors the principles of a stock market orderbook, designed to simulate trading as closely as possible to real-world operations. The primary function of the orderbook is to record all buy (bid) and sell (ask) orders from agents. Each order specifies the quantity and price at which an agent is willing to transact. The orderbook manages these orders through order placement, price matching, transaction execution, and order expiry processes. The market is accessible from every location on the grid, making it non-spatial.

2) Transactions and Order Expiry: Agents participate in the market by placing orders through the orderbook. A bid is placed by an agent who intends to purchase resources, specifying the price they are willing to pay. Conversely, an ask is placed by an agent willing to sell, indicating the price they are prepared to accept.

The orderbook's functionality lies in its ability to match buy and sell orders based on price. Orders are prioritized and matched according to their price competitiveness; bids are sorted in descending order of price, while asks are sorted in ascending order. The matching process occurs continuously, with the highest bid matched to the lowest ask if the bid price meets

or exceeds the ask price. When a match is made, a transaction is executed, transferring the resource to the buyer and payment to the seller. The price is determined by whether an agent places a bid or an ask, which cannot be instantly matched and then becomes visible to other agents, influencing their actions.

To maintain market efficiency and prevent order saturation, each order is assigned a lifespan, after which it is considered expired if not executed. Expired orders are removed from the orderbook, ensuring that the trading environment remains updated and reflective of the current market state.

3) Price Calculation and Determination: Prices in the market are determined through a combination of agent-specific calculations and market conditions:

a) Order Placement and Matching:

- **Order Books**: The market maintains order books for each resource (wood and stone), where agents can place bids (buy orders) and asks (sell orders).
- **Placing Orders**: Agents specify the maximum price they are willing to pay for bids or the minimum price they are willing to accept for asks.
- **Matching Orders**: Bids and asks are continuously matched based on price. Transactions occur when the highest bid meets or exceeds the lowest ask.

b) Agent-Specific Calculations:

- **Earning Rate and Marginal Price**:
  - Agents calculate their total expected income over their remaining lifetime.
  - This total income is divided by the required building time to determine the earning rate.
  - The marginal price (*m_price*) is then calculated by dividing the earning rate by the sum of the costs of the resources needed for building a house.
- **Market Order Book Prices**:
  - Agents check the current best bid and ask prices in the order books.
  - For selling, agents set a price at least as high as the highest bid price or their marginal price, whichever is higher.
  - For buying, agents set a price no higher than the lowest ask price or their marginal price, whichever is lower.

c) Dynamic Adjustments:

- **Update Prices**: Agents update their market rates for wood and stone based on current order book prices and their calculated marginal price.
- **Market Dynamics**: Prices adjust according to supply and demand. Higher bids raise bid prices, and lower asks reduce ask prices.
- **Order Expiry**: Orders expire after a set number of timesteps if not matched, which helps keep the order books current and reflective of the latest market conditions.

By integrating these mechanisms, the simulation creates a dynamic and responsive price determination system, ensuring that market prices for wood and stone reflect ongoing economic activities and strategic decisions of the agents.

L. Collectives

Agents in the model form and belong to aggregations that influence and are influenced by their actions and interactions. These aggregations manifest both as directly modeled aspects and as emergent properties that develop during the simulation due to the dynamics established through the model's rules and interactions. Imposed aggregations include market participation, tax policies, and grid and resource allocation. For example, the spatial distribution of agents and resources on a grid imposes a form of aggregation where proximity and resource availability can dictate economic opportunities and constraints. This setup affects how agents move, gather resources, and interact with their environment and each other. Similarly, agents interact within a market framework through orderbooks for buying and selling resources. This market setup is an imposed structure where agents are aggregated by the model to facilitate trade, affecting prices and the availability of resources like wood and stone. The market itself acts as an aggregation point for economic activity, which can lead to complex trading interactions. Additionally, the use of dynamic or static tax policies is an imposed aggregation where the agents are subjected to collective economic rules that impact wealth distribution and social welfare. The choice of tax policies and the manner in which taxes are applied and collected are predefined by the modeler and affect the agents' decision-making as decision are made considering post tax expected values.

Emergent aggregations include market dynamics and price formation, social structures and economic inequality, clustering of houses into city like formations and collective behaviors. nAs agents interact, trade, and grow their wealth by building houses, patterns of inequality and economic stratification can emerge. These patterns are not explicitly defined by the modeler but develop as agents pursue their objectives within the imposed rules of the simulation. Over time, these emergent properties can lead to new social structures within the model, such as classes of wealth or clusters of highly productive agents.

M. Heterogeneity

Agents exhibit both heterogeneity in their state variables and their decision-making processes. Agents start with different amounts of wealth, randomly assigned based on the Dutch wealth distribution, to endow agents with starting wealth similar to that found in the Netherlands. Agents are initialized at (uniform) random positions on the grid, affecting their access to resources and interaction opportunities with other agents. The lifespan of agents is determined randomly using a gamma distribution, affecting how long they can operate within the simulation. Agents also have an estimate of their own lifetime, drawn from a gamma distribution with the same parameters, which can affect their expected earning rates from building a house. This is because if an agent expects to die sooner, they will expect to receive less income from the house, making the action of building a house carry lower expected value.

Agents are divided into two types based on their decision making mechanisms: Expected Value (EV) agents and Neuroevolution agents, each employing different decision-making strategies as described previously. The decision making mechanism for EV agents is the same for each EV agents, while for the Neuroevoluton agent decision making mechanism can differ based on the specific settings of the neural network.

Agents make individual decisions about where to move based on the availability of resources and spatial distribution, and they decide when and how much to buy or sell based on their current needs, wealth, and market prices. Similarly, agents decide whether to build houses based on their resource holdings and the expected value of the action which can differ per agent due to heterogeneity in agents attributes such as lifetime expectation. These decisions are influenced by their personal attributes (inventory) and the market dynamics. Additionally, agents collect income based on their houses and are subjected to taxation, which may vary depending on their tax bracket. Neuroevolution agents also carry heterogeneity in network weights, which is affected by their parents, mutation rate, and stochasticity in the recombination. Neuroevolution agents dont explicitly pick the action with highest expeced value, although their networks are evolved over time through the generations to pick the actions which provide them with the most wealth implicitly.

N. Stochasticity

The lifetimes of the agents and agents own estimation of their lifetime are modeled as random variables with a Γ distribution. Parameters such as the mean and standard deviation are specified during the initialization of an agent. Resources spawn randoml (uniformly) on the grid both initially, and according to a spawnrate throughout the simulation. The initialization of new agents, including their starting positions are also random. Their intial wealth is randomly sampled from the Dutch wealth distribution. The evolutionairy process of the Neuroevolution agents contain stochasticity in ways described in more detail in II-F and omitted here to avoid repetition.

O. Observation

a) Data Collection: The simulation framework systematically captures data at both the individual agent level and the overall simulation level. Key parameters defining the simulation environment include the number of agents, the number of time steps, the number of resources, grid dimensions, resource spawn rates, agent lifetimes, order expiry times, tax periods, and income per time step. These parameters are dynamically adjusted and recorded for each simulation run to study their impact on the outcomes.

b) Individual Agent Metrics: Throughout each simulation run, the state and actions of each agent are tracked. This includes recording the wealth of agents at various time steps, the number of houses owned, amounts of wood and stone resources held, income generated, position on the grid, and the specific actions performed at each time step. These observations allow analysis of the behavior and strategies of individual agents over time.

c) Simulation-Level Metrics: At the end of each run, individual agent metrics are aggregated to compute overall simulation metrics. These include total welfare (sum of the normalized final wealth of all agents), the Gini coefficient (measure of wealth inequality), total productivity (sum of normalized final wealth), social equality (derived from the Gini coefficient), and social welfare (calculated as the product of social equality and total productivity). These metrics provide an overview of the system's performance and social dynamics.

d) Feature Analysis: When feature analysis is enabled, additional metrics related to agent behavior are tracked. This includes aggregated action counts (frequency of different actions performed by agents, e.g., buy, sell, gather, start building). Data related to clusters of agent behavior are also collected and normalized for comparability. Euclidean distances between clusters of agent behavior and reference vectors are calculated to quantify alignment with predefined behavioral patterns.

e) Results Documentation and Visualization: The simulator saves and visualizes various results for post-simulation analysis. Run data is saved as CSV files,

capturing the raw data for each simulation run. Metrics data, including calculated metrics for each run, are also saved as CSV files. Sensitivity analysis results, including Sobol sensitivity indices, are saved as JSON files. Additionally, various plots are generated to visualize wealth distribution, number of houses, income distribution, and aggregated income distribution over time. These visualizations aid in interpreting the simulation outcomes and identifying key trends and patterns.

III. Experimental Details

A. Experimental Design Analysis methods

1) Experiment 1: Classifying Agent Behavior:

a) Analysis: Clustering to Classify Emergent Behavior: To quantify emergent behavior, we analyze agents' actions, income, and wealth differentials. Agents are clustered based on their actions (buy, build, sell, gather - excluding move and doing nothing) and their wealth differential relative to the mean of a specific run. The agents are classified in each cluster according to the highest feature importances. For instance, an agent with a high build action count and above-average wealth is termed a builder, so the agents in the cluster with highest feature importances (both within and between clusters) are considered a wealthy builder class. Due to varying action counts across parameter settings, designing static thresholds to define classes is challenging. For example, an agent might need to build 10 times to be in the 90th percentile in one simulation, but the 10th percentile in another, making them a non-builder relative to other agents in that run. Therefore, we employ a dynamic method that adjusts class boundaries relative to the dataset of that specific run using agglomerative clustering with the Ward criterion, which minimizes within-cluster variance). Agglomerative algorithm works as follows:

1) Initialize Clusters:
   a) Start with each data point as its own cluster.
2) Find and merge the closest pair of clusters:
   a) Update distances based on WARD linkage criterion (minimum variance within clusters).
3) Repeat until stopping criterion:
   a) Repeat until $n_{clusters}$ equals a researcher-chosen number.

TABLE III: Agglomerative Clustering Algorithm

A stopping criterion of $n_{clusters} = 3$ is chosen because initial experiments showed significant heterogeneity and the emergence of distinct classes: wealthy buyers, a poor class, and traders or middle class depending on agent intelligence type. This number optimally captured behavioral differences using the specified parameters. Lower and higher values for $n_{clusters}$ (such as 2, 4, and 5) led to less heterogeneity between clusters, resulting in less meaningful classifications.

b) Analysis: Statistical Testing on Feature Importances: To verify if these classes persist across runs, similar clusters are grouped based on the lowest Euclidean distance between their feature importances across runs (as cluster labels are arbitrary). The resulting plots display mean feature importances per cluster and per feature with a 95% confidence interval. Then, pairwise ANOVA tests are performed on the feature importances, with Tukey Post hoc tests to account for multiple comparisons between clusters (per feature) to determine if cluster features (and their corresponding actions) significantly differ, indicating persistent class differences across runs. If so, this constitutes evidence for a robust emergence of a class when using the chosen parameters across multiple runs.

Now that the analysis method has been explained, the experimental setup will be discussed. We will compare the feature importance of agents' clusters within four separate simulation setups:

1) EV decision making with a static tax.
2) EV decision making with a dynamic tax.
3) Neuroevolution decision making with a static tax.
4) Neuroevolution decision making with a dynamic tax.

Each simulation will be run 30 times, and feature importance will be recorded for each cluster.

Statistical analyses, including ANOVA and Tukey's post-hoc tests, will be conducted on pairwise samples of feature importance on a per-feature basis. This will determine statistically significant differences between each cluster within one of the four simulation runs and establish (average) differences in classes that are significant and persist across runs within that simulation setup. Between simulation differences in features are not directly tested for statistical significance, but visually observed differences will be discussed.

2) Experiment 2: Comparing Agent Wealth Distributions under Different Tax Policies: In this experiment, samples of Gini coefficients, equality, productivity, and social welfare will be compared between simulations of dynamic vs. static tax policies across 30 runs.

This will be done separately for each intelligence type, yielding the following setup:

1) EV agents: Static vs. Dynamic tax policy

2) Neuroevolution agents: Static vs. Dynamic tax policy

For each setup, the metrics obtained from simulations under static and dynamic tax policies will be compared using a non-parametric Kolmogorov-Smirnov test. The null hypothesis (H0) of this test asserts that the samples come from the same distribution. Rejection of H0 would indicate a statistically significant difference between the samples, implying a significant impact of the tax policy on the respective metric.

3) Validation: Comparing Agent Income Distributions under Different Tax Policies and Decision Intelligences: Given that wealth heterogeneity is built into the model, comparing the simulated wealth distribution to its real-world counterpart in the Netherlands is less suitable than comparing it to the income distribution. Although the wealth distribution is an input to the model and evolves over time, the income distribution is a better candidate for validation because it is not a direct model input and is relevant for investigating income taxes. To validate the model, the Kolmogorov-Smirnov test is conducted on the income distribution, comparing Neuroevolution and Expected Value agents under a static tax policy. In this experiment, the income distribution of each simulation setup for each run is compared against the Dutch income distribution obtained from the CBS, resulting in the following sub experiments.

1) Income distribution of EV decision making with a static tax vs. Dutch income distribution (normalized)
2) Income distribution of EV decision making with a dynamic tax vs. Dutch income distribution (normalized)
3) Income distribution of Neuroevolution decision making with a static tax vs. Dutch income distribution (normalized)
4) Income distribution of Neuroevolution decision making with a dynamic tax vs. Dutch income distribution (normalized)

This is again done for 30 runs. For each run, a KS test is conducted to determine if the two income samples obtained are from the same distribution, with rejection again indicating that they are not from the same underlying distribution.

B. Main Experimental Parameters

The following parameters are used in the aforementioned experiments unless otherwise mentioned:

- `num_agents`: 30
- `grid_width`: 40
- `grid_height`: 40
- `n_timesteps`: 1000
- `num_resources`: 500
- `stone_rate`: 1
- `wood_rate`: 1
- `lifetime_mean`: 80
- `lifetime_std`: 10
- `resource_spawn_rate`: 0.5
- `order_expiry_time`: 5
- `tax_period`: 1
- `income_per_timestep`: 1
- `house_cost`: (2,2)
- `num_houses`: 3

Note that the tuple for `house_cost` indicates the wood and stone cost, respectively.

C. Reproducibility

The results can be reproduced by consulting the `readme.md` and running the accompanying Python code available in the project's Github repository at: `https://github.com/NitaiNijholt/ABM_project`. The necessary Python packages and their respective versions are listed in `requirements.txt`. These can be installed by navigating to the project directory and executing `pip install -r requirements.txt` in the terminal.

IV. Results

1) Wealth & Number of Houses Over Time for Both Agent Types: We first examine the wealth and number of houses for EV agents using a static tax policy only to see if the model shows expected behavior in terms of buidling and agent wealth.

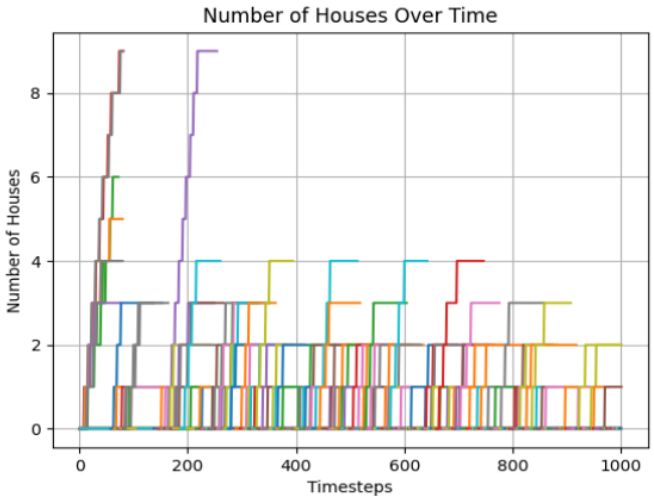


Fig. 1: Houses Over Time of EV Agents

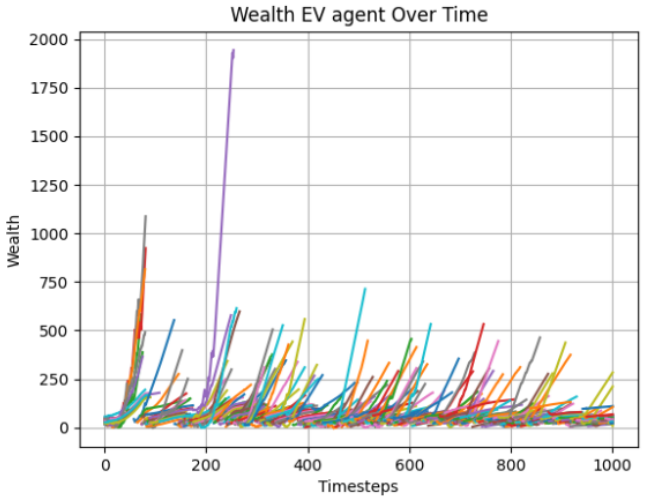


Fig. 2: Wealth Over Time of EV Agents

In Figure 1, we observe that agents own between 0 and 9 houses. More houses early in the simulation generate the most income, as reflected in Figure 2; agents who build houses early tend to continue doing so. When examining Neuroevolution agents, this dynamic persists, but agent behaviour changes over the simulation lifespan.

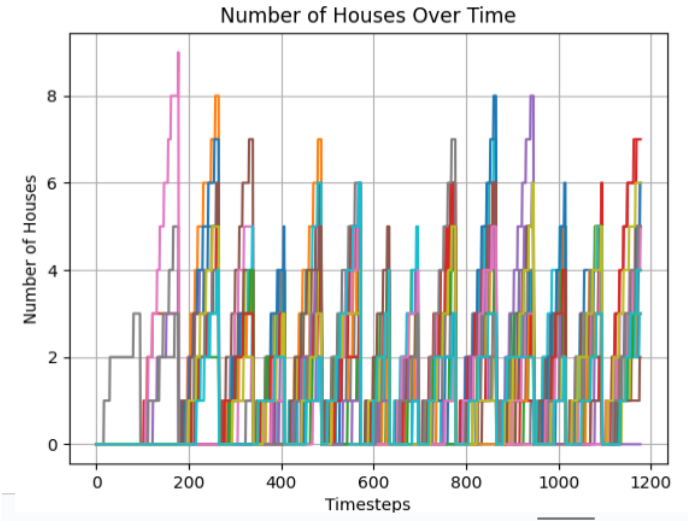


Fig. 3: Houses Over Time of Neuroevolution Agents

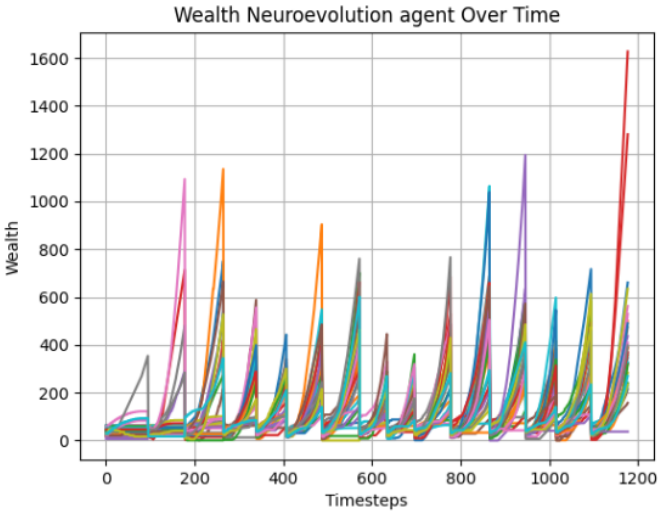


Fig. 4: Wealth Over Time of Neuroevolution Agents

Figure 4 shows that while the wealthiest agents typically have the most houses, wealth is no longer the only factor; the wealth of agents also increases over generations as shown in Figure 4. In the simulation, agents gain wealth by owning houses or trading resources. Figure **??** shows the wealth of Neuroevolution agents over time, with oscillation occurring when a parent generation is fully replaced by their offspring. Collective learning is evident as the offspring of more adept agents have a higher likelihood of surviving and passing their neural net weights and biases to the next generation. This is evidenced by the increasing density of wealth peaks over time, indicating a higher proportion of the agent population effectively gaining wealth.

Concluding, both agent types display expected behavior in terms of building houses and gaining wealth.

2) Experiment 1: Classifying Agent Behavior: The following figures illustrate the feature importances for different agent-based models under various tax policies and decision-making intelligence types (EV maximizing and Neuroevolution agents). These feature importances represent the within-cluster action counts (and significant metrics such as income and wealth) relative to the mean of that run. We average these feature importances across 30 runs and plot the mean feature importances per cluster over time. Error bars represent the 95% confidence interval (CI). The x-axis represents the respective features, and the y-axis represents the mean value in the dataset on which the 3 clusters are fitted. Pairwise ANOVA tests with Tukey post hoc tests correct for multiple comparisons on a per feature basis (e.g., comparing the 'buy' feature samples between all clusters to determine

if the means are statistically significantly different). Results of these tests will be discussed, although for conciseness, specific p-values are not stated in the text and can instead be found by navigating to `final_result_images/Feature_importance` in the project folder.

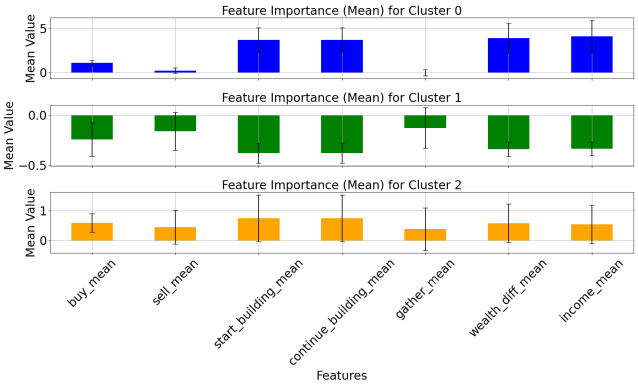


Fig. 5: EV Agents Static Tax

Figure 5 shows that Cluster 0 has the highest building and wealth/income features out of all clusters, with higher buying, indicating a wealthy builder class. Cluster 1 exhibits the lowest feature importances of all actions, with low building and wealth/income, indicating a low-activity 'poor' class. Cluster 2 displays average feature importances across the board and also average wealth, resulting in a medium-activity 'middle' class. Pairwise ANOVA and Tukey post-hoc tests indicate that the differences between clusters in mean wealth, income, and counts of buying and building are statistically significant at the $\alpha = 0.05$ level, although the differences in gathering are not statistically significant.

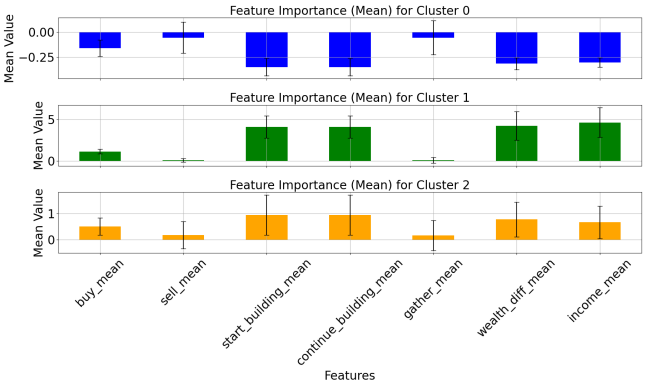


Fig. 6: EV Agents Dynamic Tax

Figure 6 shows similar patterns as Figure 5, with wealthy builder, average middle, and low-activity poor classes emerging. We observe that the poor class appears slightly less poor, although not significantly so. Pairwise ANOVA and Tukey post-hoc tests indicate that the differences between clusters in all features are statistically significant at the $\alpha = 0.05$ level, except for selling between Clusters 0 and 2, and the gathering rate between Clusters 0 and 1, where the differences in means are not statistically significant.

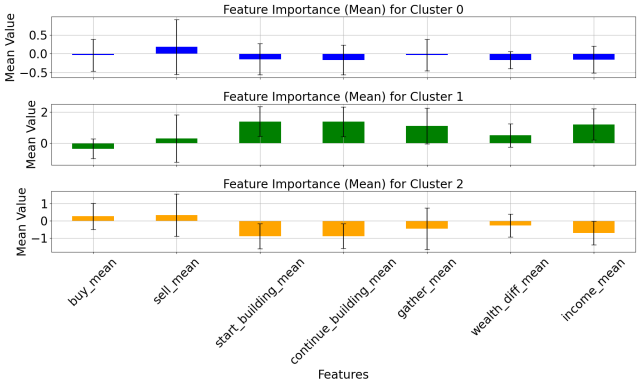


Fig. 7: Neuroevolution Agents Static Tax

Figure 7 illustrates feature importances for clusters of Neuroevolution agents. Neuroevolution agents also display a wealthy builder class in Cluster 1 (note that cluster labels are arbitrary), which shows increased gathering behavior along with some selling. Cluster 0 is characterized by above-average selling behavior and low average wealth, indicating a poor trading class that exhibits positive feature importances for selling, while other features, including wealth, are negative. Cluster 2 reveals a class with positive buying and selling behaviors, though they are the poorest in terms of income. Pairwise ANOVA and Tukey post-hoc tests indicate statistically significant differences in building behavior and income. Differences in selling and buying behavior are not significant, except for the difference in buying between Clusters 1 and 2. Differences in gathering are significant except between Clusters 0 and 2. Wealth differences between clusters are also significant, except between Clusters 0 and 2.

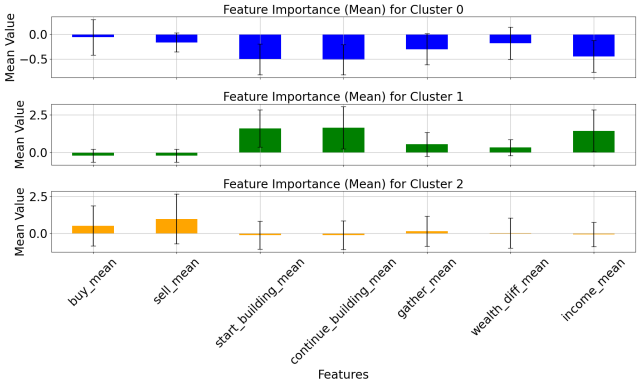


Fig. 8: Neuroevolution Agents Dynamic Tax

Figure 8, similar to Figure 7, shows consistent classes: a low activity poor class emerging in Cluster 0, and the highest builder and wealth class in Cluster 1, with trading occurring less often for this class. In cluster 2, a trading class again emerges, and in Cluster 3, trading occurs more frequently than in the other classes (although the high variability means this is not statistically significant). Pairwise ANOVA and Tukey post-hoc tests indicate statistically significant differences in buying behavior and selling between clusters, except for Cluster 0 vs. 1, where the differences are not significant. Differences in building are significant between clusters, except for Cluster 1 vs. 2. Cluster 1 also gathers significantly less than Cluster 0, The wealth difference between Clusters 0 and 1 is significant, while income differences between Clusters 0 and 2 are also significant.

In conclusion, the feature importance analysis reveals distinct patterns of agent behavior across different tax policies and agent types, which are statistically significant in many cases. Thus we reject the null hypothesis (**H0a**): We do get clusters with (within simulation) significantly different characteristics in buying, selling, gathering, and building behaviors. In 59 out of 84 pairwise feature comparisons, we reject **H0a** in favor of **H1a**: There is a significantly different feature importance's in selling, gathering and selling, and buying and building behaviors. Static and dynamic tax policies slightly influence the variability of actions but maintain the general importance hierarchy within each cluster. Neuroevolution agents seem to evolve classes that show trading behavior, although the evidence for this is not statistically significant in all cases, it is still an interesting observation warranting further investigation. A builder class emerges and is consistently the wealthiest, regardless of tax policy or agent decision intelligence type, indicating that this is a robust strategy for gaining wealth in this model across multiple decision intelligence types.

3) Experiment 2: Wealth Distribution Under Different Tax Policies: The following results were derived from implementing different tax policies and comparing wealth distributions metrics between agents types. Results can be found by navigating to `final_result_images/wealth_dist` in the project folder.

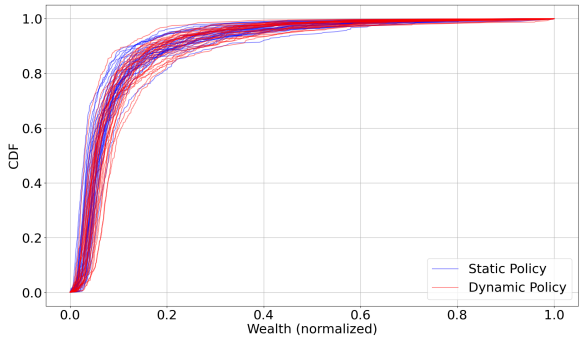


Fig. 9: Wealth CDF EV Agents (Static vs. Dynamic Tax Policies)

Metric	Static Policy Mean [95% CI]	Dynamic Policy Mean [95% CI]	KS Statistic	P-Value
Gini	0.43 [0.42, 0.44]	0.41 [0.40, 0.42]	0.50	0.0009
Productivity	52000 [50000, 54000]	49000 [48000, 51000]	0.27	0.24
Equality	0.57 [0.56, 0.57]	0.59 [0.58, 0.59]	0.50	0.0009
Social Welfare	29000 [28000, 30000]	29000 [28000, 30000]	0.13	0.96

TABLE IV: Comparison of Static and Dynamic Policies (Table 1)

Figure 9 compares the wealth distribution for EV agents. The CDF shape indicates that most agents are relatively poor, with approximately 80% of the agents owning 20% of the wealth and vice versa. Gini coefficients indicate a slight preference for the dynamic tax policy with a coefficient of 0.41 compared to 0.43 under the static policy; however, while statistically significant when using a KS test, the practical significance is limited. Productivity is higher under the static tax policy, although the difference is not statistically significant. Equality shows a statistically significant difference, albeit small in practical terms. Social welfare metrics are approximately equal and not statistically different.

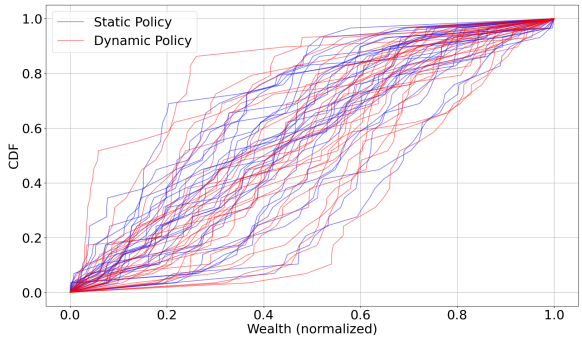


Fig. 10: Wealth CDF Neuroevolution Agents (Static vs. Dynamic Tax Policies)

Metric	Static Policy Mean [95% CI]	Dynamic Policy Mean [95% CI]	KS Statistic	P-Value
Gini	0.23 [0.21, 0.25]	0.24 [0.21, 0.26]	0.17	0.81
Productivity	8900 [7200, 11000]	6700 [5200, 8200]	0.30	0.14
Equality	0.76 [0.74, 0.79]	0.76 [0.73, 0.78]	0.17	0.81
Social Welfare	6700 [5500, 7900]	4900 [3900, 5900]	0.30	0.14

TABLE V: Comparison of Static and Dynamic Policies (Table 2)

Figure 10 shows the wealth distribution under different tax policies for Neuroevolution agents. The CDF shape depicts a more equitable wealth distribution under both tax types compared to EV agents, as confirmed by lower Gini coefficients of 0.23 for static and 0.24 for dynamic tax policies. While static tax policy exhibited higher productivity and social welfare, equality remained similar in both cases, with a score of 0.76, higher than in the EV agent case. Statistical tests conducted on samples of the metrics over 30 runs yielded no significant differences in any of the chosen metrics for static vs. dynamic tax policies. Thus, we do not reject the null hypothesis **H0b**: There is no significant difference in the Gini coefficient and total welfare between simulation runs with dynamic tax policy and those with static tax policy.

However, it should be noted that overall productivity and social welfare were considerably lower for Neuroevolution agents (52000 vs. 8900 and 29000 vs. 6700 respectively). This indicates that despite being more equitable, Neuroevolution agents are less effective at accumulating wealth than EV agents.

In conclusion, the dynamic tax policy did not lead to materially significant changes in the wealth metrics chosen, and EV agents, although resulting in a less equitable distribution, led to a richer society.

4) Validation: To validate the model, we compare the obtained income distributions of the agents over their lifetime against the Dutch income distribution in Figure 11. Results can be found by navigating to `final_result_images/validation` in the project folder. The colored lines represent the different runs, while the black dotted line indicates the empirical result. Using a Kolmogorov-Smirnov test, we find that for the chosen parameters, the null hypothesis — that the empirical and simulated income distributions come from the same distribution — is rejected for the EV agents with a found p-value < 0.001 (for each run) at the $\alpha = 0.05$ level.

Overall, the simulated income distribution for the Neuroevolution agents did not match the empirical income distribution, except in 2 out of 30 cases. In these instances, the null hypothesis of the Kolmogorov-Smirnov (KS) test — that the samples come from the same distribution — was not rejected. However, the high variability in the results suggests that these matches may be due to chance, indicating that our model does not fit the Dutch empirical income distribution well with the current parameters.

In conclusion, we reject **H0c**: There is no significant difference between the income distribution in the model and in the Netherlands. We indeed find a significant difference in 116 out of 120 tested cases, with the only exceptions being some runs for Neuroevolution agents. There is no strong evidence to suggest any difference in these results for dynamic vs. static tax policies.

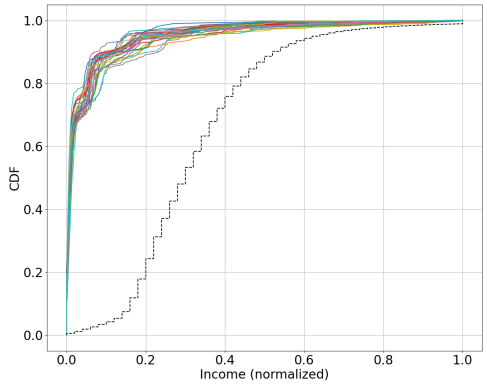
V. Sensitivity Analysis

One-Factor-At-a-Time (OFAT) and Sobol methods are two techniques for sensitivity analysis (SA) in ABMs. Sobol analysis, a global SA approach, samples the entire parameter space and quantifies the sensitivity of output parameters to input variations, accounting for interaction effects among parameters. On the other hand, OFAT is a local SA technique where a baseline parameter configuration is established, and each parameter is varied individually while the others remain fixed at their baseline values. This method is effective for detecting linear and non-linear responses to parameter changes and identifying potential tipping points; however, it is less effective at detecting interactions. OFAT is less computationally demanding compared to the Sobol method and can be used as a precursor to find non-influential parameters that can then be excluded to reduce the parameter space for the global sensitivity analysis. To conduct the sensitivity analysis, we use Python's SA lib [4] for both local and global SA. due

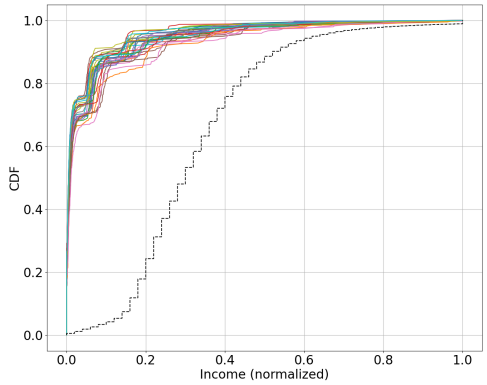
A. Local Sensitivity Analysis Results

For the local SA, we analyzed how changes in parameters affected the Gini coefficient. We used the Morris method, also known as the Elementary Effects method. The Morris method provides two key measures: μ^*, representing the mean of the absolute values of the elementary effects, and σ , representing the standard deviation of the elementary effects. High μ^* values indicate a significant influence of the parameter on the model output, while high σ values suggest variability due to non-linear effects or interactions with other parameters.

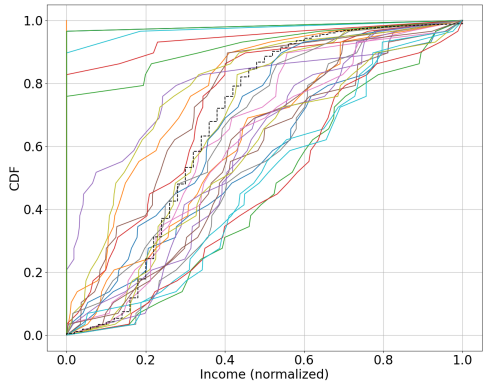
We first start off by varying the parameters for the grid, which are the *house_cost* and the number of houses allowed in each square, *max_house_amount*.



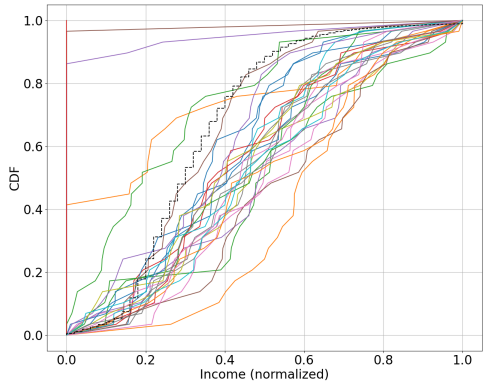
(a) Dynamic EV



(b) Static EV



(c) Dynamic Evolve



(d) Static Evolve

Fig. 11: Comparison of Income CDF vs. Empirical for Different Tax and Agent Types

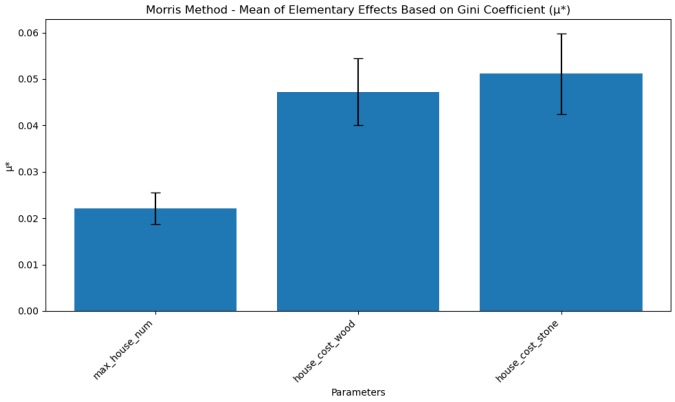


Fig. 12: The Variability Caused by Changes in Different Parameters of the Grid on the Gini-Coefficient (static tax only)

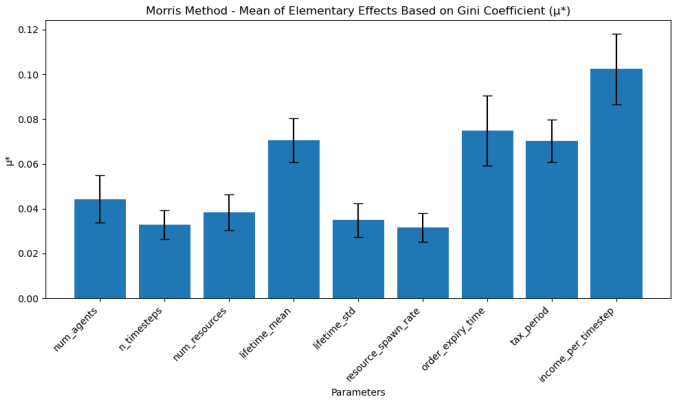


Fig. 13: The Variability Caused by Changes in Different Parameters of the Simulation on the Gini Coefficient with a Static Tax Policy

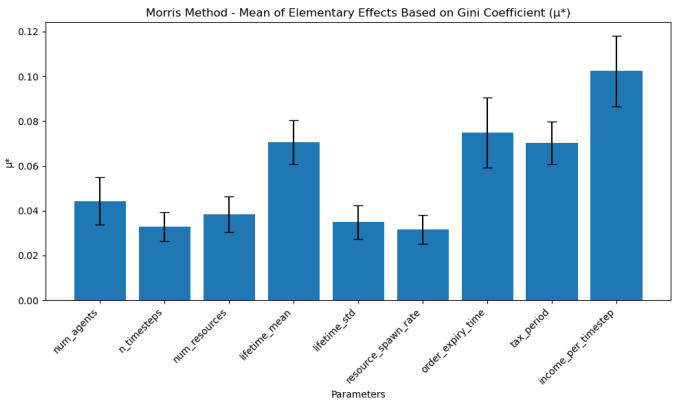


Fig. 14: The Variability Caused by Changes in Different Parameters of the Simulation on the Gini Coefficient with a Dynamic Tax Policy

a) Number of Runs & Parameter Ranges: For the local SA, we ran each parameter 100 times and, since there are 11 parameters tested, the total number of runs is $100\times(11+1) = 1200$ where 1 is the initial starting point. Each parameter range is discretized into a specified number of levels. The parameter spacing Δ for a given parameter is calculated using the following formula:

$$\Delta = \frac{b-a}{L-1}$$

where:

- a is the minimum value of the parameter,
- b is the maximum value of the parameter,
- L is the number of levels.
- `num_agents`: [5, 50]
- `n_timesteps`: [100, 1200]
- `num_resources`: [50, 5000]
- `lifetime_mean`: [70, 90]
- `lifetime_std`: [5, 15]
- `resource_spawn_rate`: [0.1, 5]
- `order_expiry_time`: [3, 7]
- `tax_period`: [1, 10]
- `income_per_timestep`: [0.5, 5]
- `house_cost`: [1, 10]
- `num_houses`: [1, 10]

Note that some values needed to be integers, therefore, those were cast as integers.

b) mu^* and σ Values of the Grid Parameters: The grid parameters have low mu^* values and also a low σ value. This indicates that the grid parameters do not have a large effect on the simulation output.

c) High μ^* Values: The parameters *lifetime_mean* and *income_per_timestep* exhibited the highest μ^* values, indicating that these parameters have the most significant influence on the model output. Furthermore, *order_expiry_time* and *tax_period* also show high μ^* values, suggesting that these parameters have the most influence over the simulations.

d) Low μ^* Values: The parameters *num_agents*, *lifetime_std*, *n_timesteps*, *num_resources*, and *resource_spawn_rate* have low μ^* values, indicating that perturbations in these parameters do not cause significant changes in the output. The parameters *n_timesteps*, *num_resources*, and *resource_spawn_rate* have the lowest μ^* values, exhibiting the least influence on the model output among those considered. Changes in these parameters do not lead to significant changes in the output.

e) Error Bars (σ): The parameters *order_expiry_time* and *income_per_timestep* have relatively large error bars, indicating variability in their effects. This suggests that their influence on the output might be due to non-linear effects or interactions with other parameters. Parameters with smaller error bars, such as *num_agents* and *lifetime_std*, have more consistent effects, indicating less variability and potentially more linear effects.

We observe that changes in tax policy do not affect the sensitivities to the parameters as shown in Figure 13 and Figure 14. Furthermore, due to the mu^* and σ findings, one could choose to only vary *lifetime_mean*, *income_per_timestep*, *order_expiry_time*, and *tax_period* for the global sensitivity analysis, as these parameters have shown to have the largest effect on the simulations, and fix other parameters which have low μ^* values.

Our overarching conclusion is that we see differing sensitivities with some parameters having smaller effects according to Morris' OFAT; however, because all parameters are still relatively similar in magnitude (with the highest magnitude being 0.1 and the lowest being 0.3, a 3.33 factor difference), we find it more reasonable to consider all parameters for the global sensitivity analysis instead of excluding the smaller effect ones, even though we realize this will come at a high computational cost reducing possible run numbers.

B. Determining Agent Behavior Class Persistence Across Parameter Space

This section will provide an explanation about the devised measure for class persistence across the parameter space, clarifying the concept for the reader, which is necessary for meaningful interpretation of the results.

To investigate where in the parameter space the classes observed in our main experiment persist during the global sensitivity analysis, we face the challenge of quantitatively determining class membership when the threshold for belonging to a class can change according to the parameter space combination.

To address this problem, recall that clustering was chosen to determine class membership because it allows for the distinction of classes in different action spaces relative to that specific run. Classification then occurs according to the highest importance features of the cluster.

To compare different clusters across the action space, we perform 20 runs with a certain parameter combination and average the feature importances to obtain $\mathbf{v}^c_{\text{sim}}$ for each cluster c. We then compare this cluster's averaged feature importance vector to a reference vector to determine the Euclidean distance. We define the average feature importance vectors obtained in our main experiments as the reference vectors, $\mathbf{v}^c_{\text{ref}}$, for each cluster c, obtained using 30 runs. The rationale behind averaging the feature importances of the runs is to

average away the stochastic effects of single runs that are present due to the multiple levels of stochasticity in the model. Importantly, before computing distances, feature importances are normalized such that their sum is 1.

Comparing the Euclidean distance between average feature importance vectors of clusters can then serve as a quantitative measure of class persistence across runs for different parameter combinations, with a low Euclidean distance indicating a closer fit and thus more class persistence than a high Euclidean distance between feature importance vectors. The advantage of this measure is its straightforwardness. However, its downside is that by aggregating all feature distances into a single scalar, it obscures which specific features have changed more than others. This method only indicates that a class has changed, not how it has changed. We accept this limitation for now, as a high variance in Euclidean distance across the parameter space for a specific cluster still indicates that the parameter influences class formation.

Clusters from the current run are aligned with the reference clusters by computing a Euclidean distance matrix and solving the assignment problem using the Hungarian algorithm, which minimize the total distance. This means we are matching up clusters with their most similar reference clusters in terms of the lowest Euclidean distance. We then record that Euclidean distance per cluster, measuring how well the simulation clusters match the reference clusters in terms of feature importance. The variability in these distances is then the output metric for the global sensitivity analysis. This effectively measures the impact of varying the parameter of interest on the variability of the distance of the clusters, giving a rough measure of the impact on class persistence across the parameter space of the classes observed in the main experiments.

Formally, let:

$$\mathbf{v}_{\text{ref}}^{c} = \text{Average feature importance vector for the reference cluster} \tag{19}$$

$$\mathbf{v}_{\text{sim}}^{c} = \text{Average feature importance vector for the simulation cluster of a specific parameter combination} \tag{20}$$

$$d(\mathbf{v}_1, \mathbf{v}_2) = \|\mathbf{v}_1 - \mathbf{v}_2\|_2 = \sqrt{\sum_{i=1}^{n} (v_{1,i} - v_{2,i})^2} \tag{21}$$

The objective is to minimize the total Euclidean distance between the reference vectors and the simulation vectors by solving the assignment problem:

$$\min_{\sigma} \sum_{c=1}^{k} d(\mathbf{v}_{\text{ref}}^{c}, \mathbf{v}_{\text{sim}}^{\sigma(c)}) \tag{22}$$

where σ is a permutation of the indices representing the optimal assignment of clusters. The minimum distances found are then recorded, and the variability in the distances $d(\mathbf{v}_{\text{ref}}^{c}, \mathbf{v}_{\text{sim}}^{c})$ across different parameter combinations is analyzed to assess the impact on class persistence.

C. Global Sensitivity Analysis Results

For our global sensitivity analysis, we vary multiple parameters simultaneously, capturing of the effect of interactions using the Sobol method [8]. The Sobol method is a quesi-Monte Carlo sampling method that ensures a lower discrepancy sequence with more uniform coverage of the input space, providing accurate sensitivity indices with fewer samples. To maintain its convergence properties, the sample size must be a power of 2. The convergence rate of the Sobol method is $O\left(\frac{(\log N)^d}{N}\right)$, where N is the number of samples and d is the number of input dimensions.

The following parameter ranges were used for the global sensitivity analyses, using 20 runs and a base number of Saltelli samples of 128:

- `num_agents`: [5, 50]
- `n_timesteps`: [100, 1200]
- `num_resources`: [50, 5000]
- `grid_width`: [40, 100]
- `grid_height`: [40, 100]
- `lifetime_mean`: [70, 90]
- `lifetime_std`: [5, 15]
- `resource_spawn_rate`: [0.1, 5]
- `order_expiry_time`: [3, 7]
- `tax_period`: [1, 10]
- `income_per_timestep`: [0.5, 5]
- `house_cost`: [1, 10]
- `num_houses`: [1, 10]

a) Static Tax Policy Global SA Results: The global sensitivity analysis using a static tax policy yielded the following results.

The relevant files can be found in `sensitivity_analysis_result/global_sa_ev_static/` in the main project folder.

Figure 15 displays the first order Sobol indices for Static tax and EV agents. Sobol first order sensitivity indices should be between 0 and 1, so the appearance of negative indices suggests numerical instability in the model, which likely causes these anomalies for certain

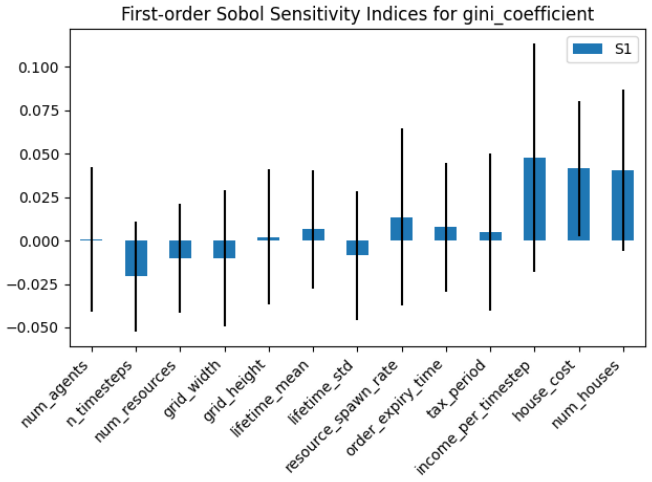


Fig. 15: Global SA Gini Coefficient First Order Sensitivities for EV Agents & Static Tax Policy

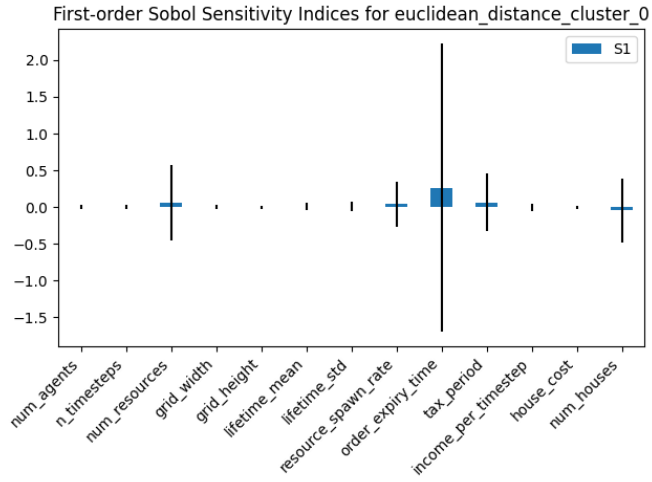


Fig. 17: Global SA Cluster 0 Euclidean Distance First Order Sensitivities for EV Agents & Static Tax Policy

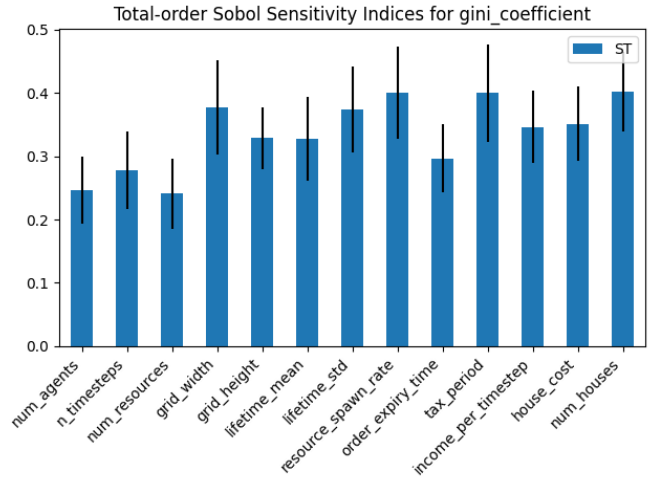


Fig. 16: Global SA Gini Coefficient Total Order Sensitivities for EV Agents & Static Tax Policy

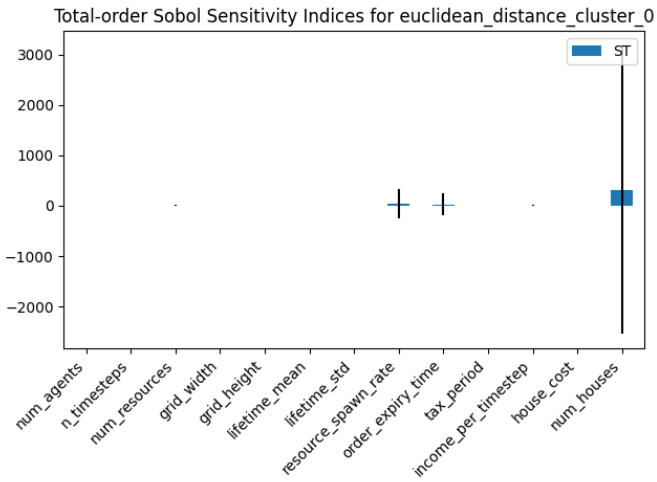


Fig. 18: Global SA Cluster 0 Euclidean Distance Total Order Sensitivities for EV Agents & Static Tax Policy

parameter combinations. While we will proceed with interpretation, the results might be spurious. According to this plot, the most influential parameter for the Gini coefficient is income per timestep, followed by house cost and then the number of houses per grid cell, which is logical given that assets provide the income.

Figure 16 shows that the tax period, resource spawn rate, and number of houses most influence the parameter variance in the Gini coefficient. However, all parameters have total order sensitivity indices between 0.25 and 0.45, indicating comparable influence when considering interactions.

Figure 17 identifies which parameters have the largest direct influence on the variance in the Euclidean distance of Cluster 0, the wealthy builder cluster (also depicted in Figure 5). throughout the parameter space. Order expiry time, number of resources, resource spawn rate, tax period, and number of houses, with other parameters having negligible influence. Note that while Sobol indices are positive, a variance higher than 2 suggests that more runs might be needed for a more definitive result.

Figure 18 shows which parameters, when considering interactions, have the largest influence on the variance in the Euclidean distance of Cluster 0, the wealthy builder cluster (also depicted in Figure 5) throughout the parameter space. The total variance explained indicates that the number of houses allowed per grid cell contributes to a significant amount of variance in the result, followed by the resource spawn rate and order expiry time. The high variance suggests a possible numerical error might be

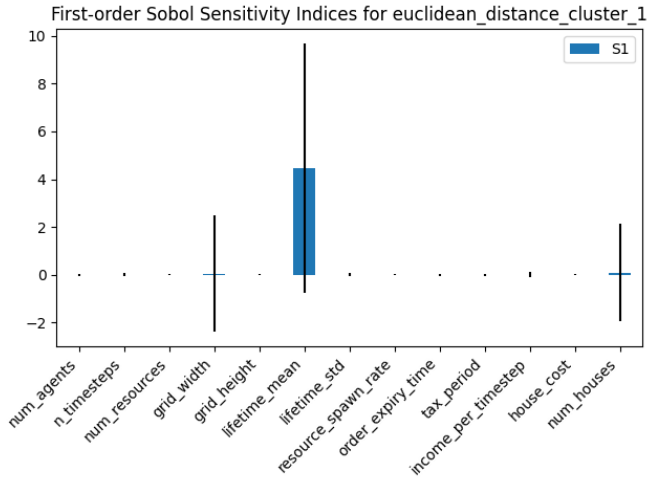


Fig. 19: Global SA Cluster 1 Euclidean Distance First Order Sensitivities for EV Agents & Static Tax Policy

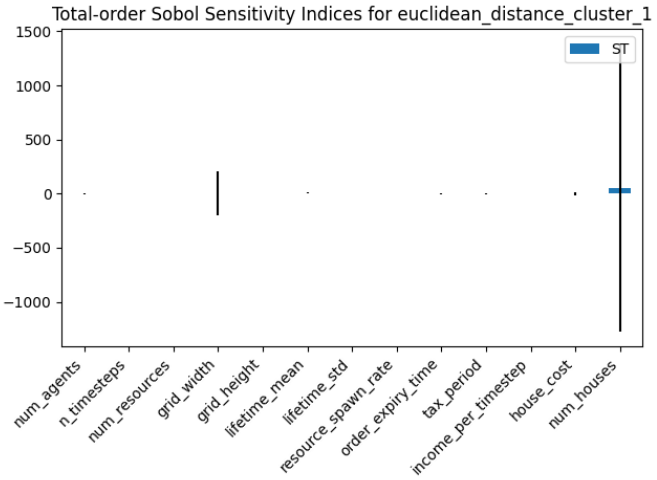


Fig. 20: Global SA Cluster 1 Euclidean Distance Total Order Sensitivities for EV Agents & Static Tax Policy

affecting the results.

Figure 19 illustrates which parameters have the largest direct influence on the Euclidean distance of Cluster 1, the low activity poor cluster (also shown in Figure 5) throughout the parameter space. For this cluster, it is primarily the lifetime that contributes to variance in the distance from their reference cluster, although a first order sensitivity index above 1 suggests a possible spurious result.

Figure 20 displays which parameters, when considering interactions, have the largest influence on the variance in the Euclidean distance of Cluster 1, the low activity poor cluster (also shown in Figure 5) throughout the parameter space. The number of houses per grid cell is the most influential parameter, with grid width

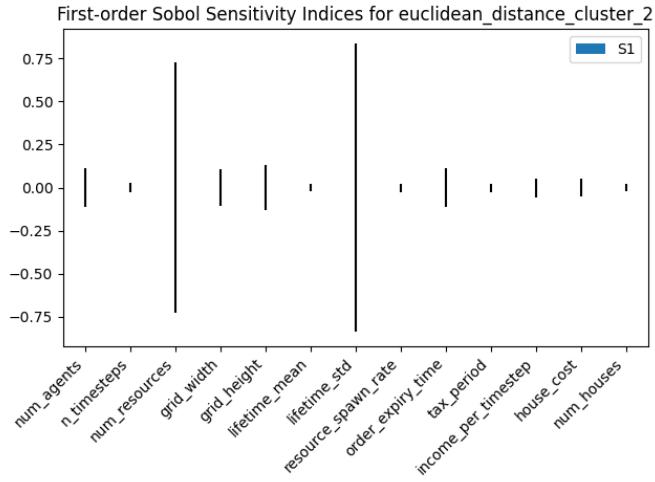


Fig. 21: Global SA Cluster 2 Euclidean Distance First Order Sensitivities for EV Agents & Static Tax Policy

also having some influence, while all other parameters have negligible effects according to the plot. Observing the y-axis, the variances are again very high, suggesting possible numerical errors.

Figure 21 indicates which parameters have the largest direct influence on the variance in the Euclidean distance of Cluster 2, the medium activity middle class (also depicted in Figure 5) throughout the parameter space. The plot reveals that there are almost no first-order effects significantly contributing to the variance in Euclidean distance. This could be due to feature importances for this cluster being averaged, so changes in one cluster feature might be offset by an opposite change in another, causing the overall change in feature importance distances to be low, even though individual feature importance distances may have shifted. This low sensitivity highlights a shortcoming of the chosen measure but also suggests that the class may be persistent regardless of parameter changes, or there could be a numerical error.

Figure 22 shows which parameters, when considering interactions, have the largest influence on the variance in the Euclidean distance of Cluster 2, the medium activity middle class (also directly shown in Figure **??**) throughout the parameter space. We again observe large variances in the number of resources and grid height, which complicates the interpretation of the effect sizes. Manual inspection of the data (found in `sensitivity_analysis_results.json` within the `sensitivity_analysis_result/global_sa_ev_st` folder) reveals that despite variations in variance, all total order sensitivities are around 1, indicating that each parameter similarly contributes to the variance in

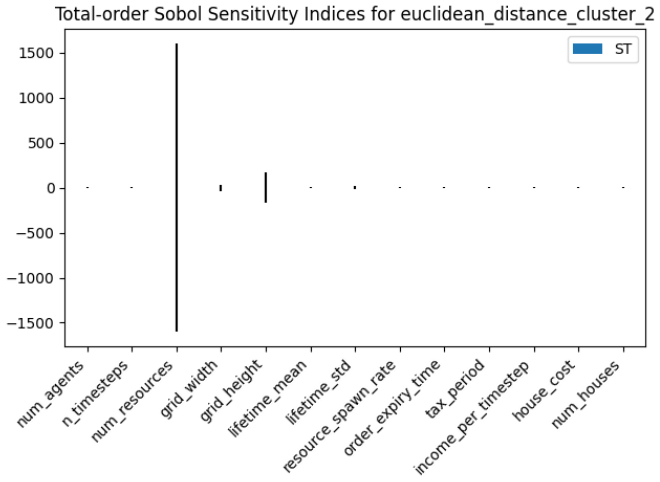


Fig. 22: Global SA Cluster 2 Euclidean Distance Total Order Sensitivities for EV Agents & Static Tax Policy

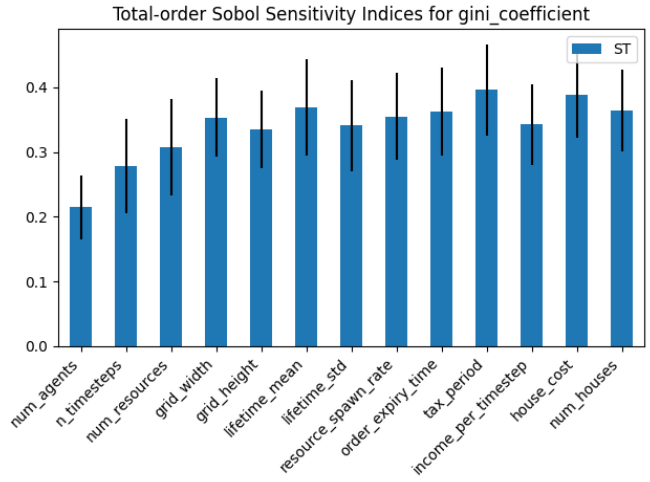


Fig. 24: Global SA Gini Coefficient Total Order Sensitivities for EV Agents & Dynamic Tax Policy

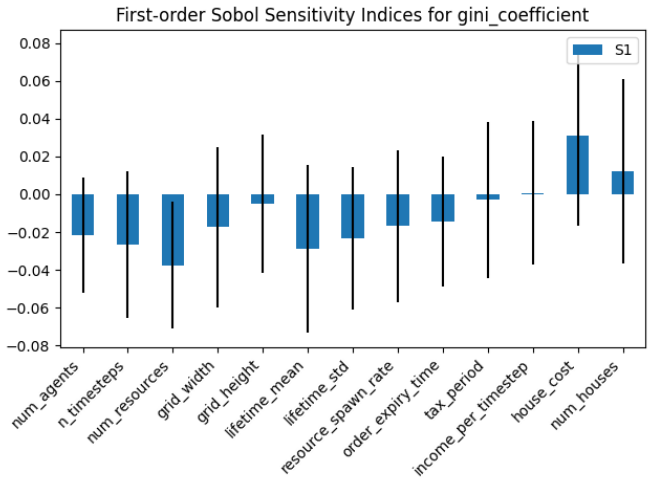


Fig. 23: Global SA Gini Coefficient First Order Sensitivities for EV Agents & Dynamic Tax Policy

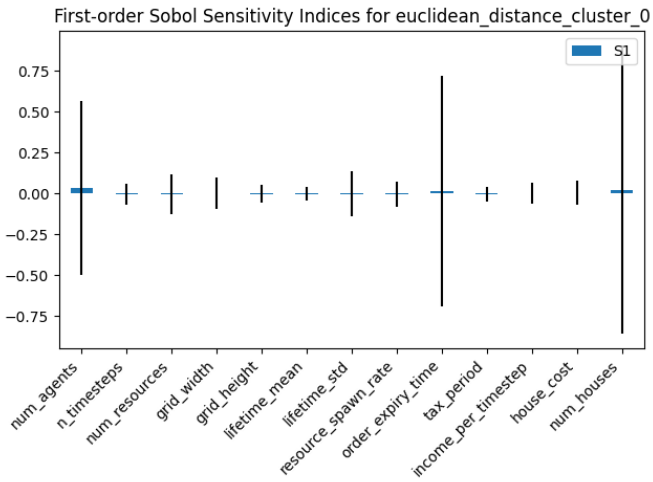


Fig. 25: Global SA Cluster 0 Euclidean Distance First Order Sensitivities for EV Agents & Dynamic Tax Policy

Euclidean distance from the reference cluster features.

b) Dynamic Tax Policy Global SA Results: For the global sensitivity analysis using the dynamic tax policy, we obtain the following results. The relevant files can be found in `sensitivity_analysis_result/global_sa_ev_dynamic/` in the main project folder.

Figure 23 shows the first order Sobol indices using Dynamic tax and EV agents. Sobol first order sensitivity indices should be between 0 and 1, so the presence of negative Sobol indices here suggests likely numerical instability in the model, which causes these indices for certain parameter combinations. We will proceed with interpretation but caution the reader to keep in mind that results may be spurious. According to this plot, the most influential parameter for the Gini coefficient is house cost, followed by the number of houses per grid cell, which makes sense given that assets provide income. In contrast with the static policy, house income is not one of the most important features, in fact having a slight positive influence around 0.

Figure 24 shows total order sensitivity indices indicating that the tax period, house cost, and the number of houses influence the variance in the Gini coefficient the most. All parameters have a total order sensitivity index between 0.2 and 0.4, showing that all parameters have comparable influence when considering interactions.

Figure 25 shows which parameters have the largest direct influence on the variance in the Euclidean distance of Cluster 0, the wealthy builder cluster (also shown

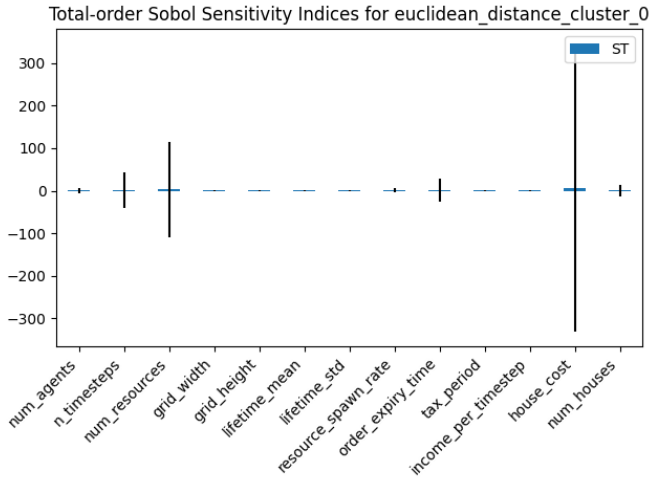


Fig. 26: Global SA Cluster 0 Euclidean Distance Total Order Sensitivities for EV Agents & Dynamic Tax Policy

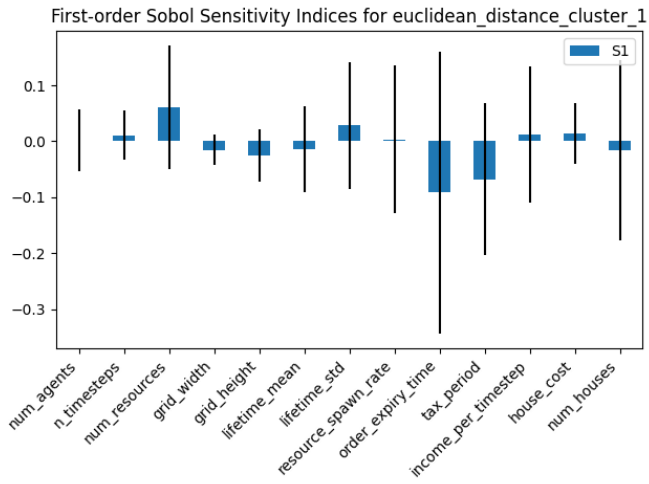


Fig. 27: Global SA Cluster 1 Euclidean Distance First Order Sensitivities for EV Agents & Dynamic Tax Policy

in Figure 6) throughout the parameter space. Overall, the first order sensitivities are negligible. However, the number of agents and houses have the largest influence, with order expiry time also having some effect and these parameters showing the highest variance.

Figure 26 illustrates which parameters, when considering interactions, have the largest influence on the variance in the Euclidean distance of Cluster 0, the wealthy builder cluster (also directly shown in Figure 6) throughout the parameter space. Overall, the variances are quite high. House cost, followed by the number of resources and houses per grid cell, have the highest influence on the variance of the distance from the reference cluster, aligning with these variables' impact on building activities, the key feature of the wealthy builder class.

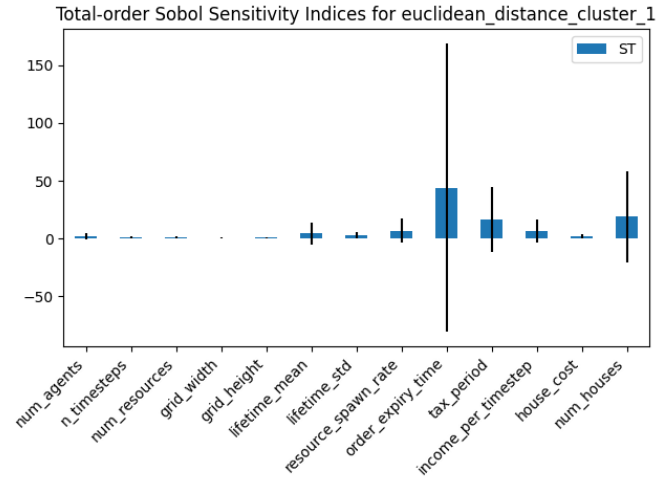


Fig. 28: Global SA Cluster 1 Euclidean Distance Total Order Sensitivities for EV Agents & Dynamic Tax Policy

Figure 27 indicates which parameters have the largest direct influence on the Euclidean distance of Cluster 1, the low activity poor cluster (also directly shown in Figure 6) throughout the parameter space. This shows slight negative first order sensitivities, suggesting a lack of runs or numerical instability, though the variances appear within reasonable bounds and the sensitivities remain below their upper limit of 1. The number of initial resources has the most direct impact on the variance in the distance from the reference cluster, which is plausible considering gathering is the primary source of income for the poor.

Figure 28 shows which parameters, when considering interactions, have the largest influence on the variance in the Euclidean distance of Cluster 1, the low activity poor cluster (also directly shown in Figure 6) throughout the parameter space. Order expiry time significantly affects the persistence of this class the most, followed by the number of houses allowed per grid cell and the tax period.

Figure 29 shows which parameters have the largest direct influence on the variance in the Euclidean distance of Cluster 2, the medium activity middle class (also shown in Figure 6) throughout the parameter space. The number of resources significantly impacts the persistence of the class, though the first order sensitivity index above 1 indicates possible numerical instability.

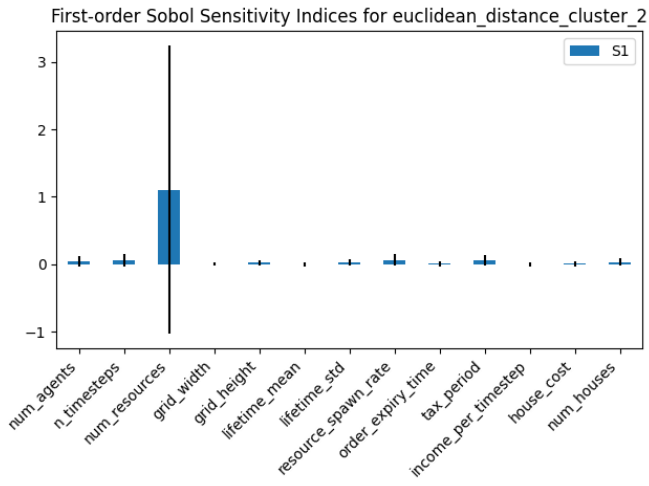


Fig. 29: Global SA Cluster 2 Euclidean Distance First Order Sensitivities for EV Agents & Dynamic Tax Policy

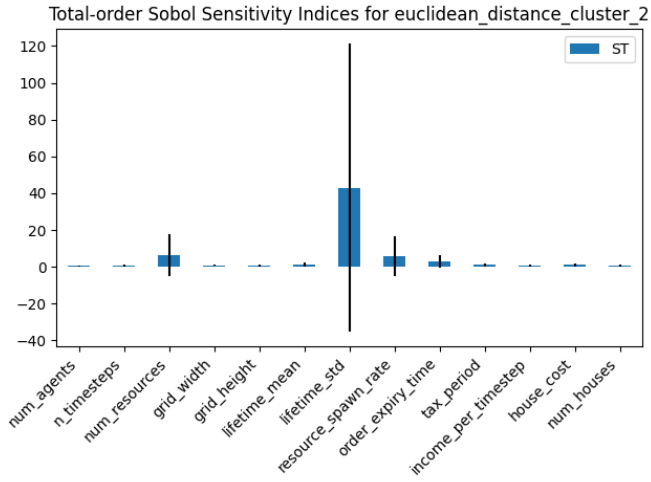


Fig. 30: Global SA Cluster 2 Euclidean Distance Total Order Sensitivities for EV Agents & Dynamic Tax Policy

Figure 30 illustrates which parameters, when considering interactions, have the largest influence on the variance in the Euclidean distance of Cluster 2, the medium activity middle class (also directly shown in Figure 6) throughout the parameter space. Lifetime mean, which influences both agent lifetimes and their expectations, impacts class persistence the most, followed by the number of resources, resource spawn rate, and order expiry time. This contrasts with using a static tax policy, where all total order sensitivity indices were around 1 for Cluster 2.

VI. Discussion

This report investigated the emergent behavior that occurs when assets providing income over time are added to an ABM, based on [10]. Our first research question concerned which types of emergent behavior would occur as a result of adding assets. We found that a wealthy builder class and a low-activity poor class emerged for both agent intelligence types. Distinct classes between intelligence types included a Trader class for Neuroevolution agents and a balanced activity average wealth class for EV agents. Global sensitivity analysis results revealed some numerical instability, indicating possible spuriousness. Following total order sensitivity analysis, variables that materially affected conditions related to houses on the grid (our assets), number of resources traded (order expiry time), and mean lifetime seemed to affect variance in the distance from the reference classes, builders, and the poor class, which is related to their distinctive feature importances. The middle class seemed largely unaffected by parameters, which could indicate persistence of the class across the parameter space, but further research should be carried out not only to consider the total Euclidean distance but to check if each feature-wise distance difference (before being summed to the total Euclidean distance) remains constant, as these pairwise feature distances changing would affect class persistence as defined in this paper but cannot be captured by the total Euclidean distance measure.

Overall, the introduction of an income-generating asset over time led to the emergence of builder classes significantly wealthier than other agent classes. This occurred without directly hardcoding skill differentials, unlike in Zhang et al.'s paper, where different incomes were gained from various actions. This modification to the model and the emergent behavior is a tangible contribution of our research, creating a more realistic representation of assets while still achieving similar emergent specialization behavior. Importantly, this specialization arises naturally from the decision dynamics of multiple types of agent intelligence, including Evolutionary Value (EV) and Neuroevolution, rather than being pre-programmed. Achieving this result with wealth heterogeneity of agents only makes our model easier to parameterize using real wealth data rather than a skill differential, which is harder to define and measure for policymakers.

The hint of trading classes emerging in the case of Neuroevolution, while requiring further evidence and experimentation to validate robustness through sensitivity analysis, is an interesting suggestion for future research. The Neuroevolution approach we employed also carried significant computational cost advantages. Zheng et al.'s paper ran a maximum of 10 agents at a time and required weeks of training, while we ran 30 agents for our

main experiment, and effective agents evolved within a single simulation run, thereby gaining much higher 'training' efficiency. A limitation of our research is that simulation differences in feature importances between simulations with differing decision intelligences and tax policies were not tested for statistical significance, but only visually observed differences have been discussed. Nonetheless, the tax policy did not seem to affect emergent class behaviors. Resulting wealth distributions compared across different tax simulation setups also showed no material differences but will be discussed in more detail next.

Our second research question focused on the impact of a static tax policy (using the same tax rates and corresponding brackets as those in the Netherlands) versus a dynamic tax policy, which adjusted tax rates for each bracket to reduce wealth inequality. We found that the dynamic tax policy did not significantly alter the wealth distribution, equality, or productivity metrics for any type of agent. This contrasts with the findings from Zheng et al.'s paper, which inspired this work, where the AI-optimized dynamic tax policy effectively promoted wealth equality and maximized social welfare by balancing equality and productivity [10]. Our simulation results revealed statistically significant differences in Gini coefficients between static and dynamic policies, but the effect size was too small to be materially consequential. Preliminary results with fewer runs indicated far more significant differences between tax policies in Gini coefficients in our model using EV agents; however, these were not reproduced with a higher number of runs, so we attribute these to stochasticity. For the Neuroevolution agents, the differences between different tax policies were also not significant.

The dynamic tax policy may have failed to significantly impact welfare and productivity metrics because it was not parameterized aggressively enough to redistribute wealth effectively. Future research could explore lowering the standard deviation threshold for redistribution or more substantially adjusting tax rates when this threshold is exceeded.

In contrast, Zheng et al.'s paper showed that a reinforcement learning algorithm to optimize the tax policy successfully increased social welfare. This suggests that a more nuanced dynamic parameterization of the tax policy or an alternative method for adjusting tax rates might be necessary to achieve the desired reduction in inequality and increases in social welfare, as adaptive learning mechanisms allow tax policies to evolve based on the dynamic responses of economic agents.

Another perspective is that reducing inequality through tax policy is challenging because the introduction of assets often leads to a "rich get richer" phenomenon, which is hard to avoid regardless of tax choices. Making such a strong claim based on the current evidence may be premature. We nonetheless suggest this as an interesting area for future research and encourage researchers to explore various characterizations of dynamic tax to see if the "rich get richer" phenomenon is indeed robust.

Validation showed that using our main experimental parameters, the model does not reproduce the income distribution of the Netherlands when the initial wealth endowment for agents was sampled from the Dutch wealth distribution. However, further exploration of the parameter space could lead to a combination that better fits real data, although this would not guarantee predictive validity. Other limitations include that wealth taxes are not considered; all income (working, trading, etc.) is treated equally, which is not the case under the tax code of the Netherlands. Income distributions are normalized and thus represent only the same distribution shape, not actual income numbers. For the Neuroevolution simulations, we did not use burn-in time, leading to runs where the agents displayed barely evolved behavior, which influenced outputs. This could be fixed by allowing the system to evolve for a certain number of runs before starting the measurements.

Also, a sensitivity analysis for Neuroevolution agents was not performed due to time and computational constraints but would be an interesting candidate for future research. Preliminary experiments indicated the possible emergence of a trader class, which could be confirmed by finding high buying, selling, and wealth feature importances for certain parameter settings. Persistence of that class could then be measured by varying relevant trading parameters. The fact that order expiry time was also an important variable explaining class persistence in some of the EV agents provides a further hint that this might be an interesting line of research. Some evolutionary computing-specific considerations include that the entire generation of Neuroevolution agents was replaced by offspring, leading to a higher risk of the agents getting stuck in local optima. This contrasts with forcing more diversity in the population by, for example, keeping a proportion of the parents or giving mutant strategies that show some success and differ from the population in action counts artificially higher fitness. This would facilitate a search of the space more likely to find a global optimum. A final limitation we will discuss is that the original number of clusters determined was a researcher's choice based on the observation of initial runs under one set of parameters, which might not be representative of other parts of the parameter space

but nonetheless is consequential for the classification of classes across the rest of the parameter space.

VII. Conclusion

This study found that introducing assets, which provide income over time, is associated with the formation of distinct behavioral classes in agents, with wealthy builders and a low-activity poor class emerging, as well as hints of an emergent trading class (in the case of Neuroevolution agents) and a middle class (in the case of Expected Value agents). Dynamic tax policy did reduce the inequality observed under a static policy statistically significantly for the EV agents, but not materially significantly due to the low effect size. As for validation, parameterizing the model with the wealth distribution found in the Netherlands did not reproduce the empirical income distribution found in the Netherlands. We encourage future researchers to consider assets that generate income over time as model components to see if emergent behaviors observed (such as the emergence of traders, a middle class, and a wealthy asset-owning class) can be replicated. Further investigation into how individual feature importance's change across the parameter space can also offer deeper insights into the conditions underlying observed emergent behavior. This approach may also reveal new emergent behaviors, such as the indications of a trading class noted in our preliminary results. For further research into Neuroevolution agents, we recommend a full sensitivity analysis and taking into account the burn-in (learning) time of Neuroevolution agents before recording outputs and investigating the possible emergence of a trading class using different parameters. Lastly we recommend research into a more aggressive implementation of dynamic tax policy to determine if the wealth distribution of agents is materially affected.

References

[1] Statistics Netherlands (CBS). *Income Distribution (Standardised Income)*. Accessed: 2024-06-07. n.d. URL: https://www.cbs.nl/en-gb/visualisations/income-distribution.

[2] A. E. Eiben and J. E. Smith. *Introduction to evolutionary computing*. 2nd. Springer, 2015. DOI: 10.1007/978-3-662-44874-8.

[3] Joshua M. Epstein. *Generative Social Science: Studies in Agent-Based Computational Modeling*. Princeton, NJ: Princeton University Press, 2006. ISBN: 978-0691125473.

[4] J. Herman and W. Usher. "SALib: An open-source Python library for sensitivity analysis". In: *Journal of Open Source Software* 2.9 (2017). DOI: 10.21105/joss.00097.

[5] T. Iwanaga, W. Usher, and J. Herman. "Toward SALib 2.0: Advancing the accessibility and interpretability of global sensitivity analyses". In: *Socio-Environmental Systems Modelling* 4 (2022), p. 18155. DOI: 10.18174/sesmo.18155.

[6] Ministerie van Financiën. *Vermogensverdeling in beeld*. Retrieved from https://www.rijksoverheid.nl/documenten/begrotingen/2023/04/28/voorjaarsnota-2023. 2023.

[7] *Morris Method*. Accessed: 2024-07-09. URL: https://www.sciencedirect.com/topics/engineering/morris-method.

[8] Ilya M Sobol. "Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates". In: *Mathematics and computers in simulation* 55.1-3 (2001), pp. 271–280.

[9] Statistics Netherlands (CBS). *Income distribution (standardised income)*. Retrieved June 7, 2024, from https://www.cbs.nl/en-gb/visualisations/income-distribution. 2022.

[10] Stephan Zheng et al. "The AI Economist: Taxation policy design via two-level deep multiagent reinforcement learning". In: *Science Advances* 8.18 (2022), eabk2607. DOI: 10.1126/sciadv.abk2607. eprint: https://www.science.org/doi/pdf/10.1126/sciadv.abk2607. URL: https://www.science.org/doi/abs/10.1126/sciadv.abk2607.

VIII. Appendix