UNIVERSITY OF EXETER

Computer Science Department

B.Sc. Computer Science
Computer Science Department

The Impact of Altruism on Ageing Populations: Unleashing the Potential of a Python Framework for the Sugarscape Model, and investigating how it can help us explore the complex dynamics of social and economic systems.

Candidate

David Ogunlesi

Student ID 700017447

Supervisor

Dr. Marcos Oliveira

University of Exeter

Co-supervisor

Academic Year
2022/2023

Abstract

This thesis explores the use of the Sugarscape model as a framework for understanding social systems. The motivation behind this research is to enhance the utility of the model by expanding its rule set to include altruism and exploring its impact on an ageing population. The goals of the thesis include developing a flexible and accessible Python framework, expanding the rule set, investigating altruistic agents, and informing policies to address ageing populations.

The thesis presents a case study of Japan that examines the influence of altruism, in an ageing population context. The results of the simulations highlight the importance of social support mechanisms and adequate resource allocation in addressing the challenges posed by ageing populations. The findings suggest the need to promote altruistic behaviour and allocate more resources to healthcare, elder care, and social welfare programs.

The thesis concludes with a discussion of the limitations of the study and the framework, suggesting future research directions such as better representing specific populations and optimizing the framework for grander simulations. The significance of this research extends beyond computer science, as it provides insights into the challenges faced by countries with ageing populations and offers a tool for developing tailored policies and interventions.

	Yes	No
I certify that all material in this dissertation which is not my own work has been identified.	☑	☐
I give the permission to the Department of Computer Science of the University of Exeter to include this manuscript in the institutional repository, exclusively for academic purposes.	☑	☐

Contents

List of Acronyms and Definitions

Sugarscape A simulated environment that represents a simplified social and economic system, where agents interact and compete for limited resources, typically depicted as patches of "sugar."

CA Cellular Automata. A discrete mathematical model of artificial agents.

ABM Agent Based Model

Agent A computational entity or individual within a system that is capable of perceiving its environment, making decisions, and taking actions based on its internal state and external stimuli.

Rational Agents Agents that make decisions based on logical reasoning and pursuit of their own self-interest.

Bounded Rationality The concept that agents have cognitive limitations and can only make decisions based on a limited amount of information and computational capacity.

Altruism The selfless concern and willingness to act for the benefit of others, even at a cost to oneself.

Heterogeneous Preferences The idea that different agents may have diverse and varied preferences, values, and priorities when making decisions or assessing outcomes.

GUI Graphical User Interface

MVC Model-view-controller

Scape A simulated landscape or environment where agents interact and operate on/in, often representing a simplified or abstracted version of a real-world system or phenomenon

Spatial Indexing A technique used to organize and optimize the storage and retrieval of spatial data, enabling efficient spatial queries and operations.

Cartesian Coordinates A system for representing points or locations in a two-dimensional or three-dimensional space using perpendicular axes and numerical values.

Cython A programming language that combines the simplicity and ease of Python with the speed and efficiency of C, allowing for high-performance code execution.

Prisoner's Dilemma A game theory scenario in which two individuals, acting in their own self-interest, face a choice between cooperation and betrayal, often resulting in suboptimal outcomes due to the conflict between individual and collective interests.

1 Thesis Motivation and Structure

1.1 Motivation

Achieving a comprehensive understanding of resource allocation and inequality in social and economic systems requires a modelling framework that can capture the complex dynamics and interactions between agents and resources. The ***Sugarscape*** has historically been proven to be a valuable tool for studying such systems, but we may further enhance its utility by expanding its rule set to more specific behaviours and exploring each rule's impact on the models' behaviour.[1] A flexible and accessible Python Framework for the ***Sugarscape*** model would provide a unique opportunity to push the model's limits and uncover insights into the intricacies of the socioeconomic systems we live in. By examining rules in isolation, or rather limited rulesets, we can build a more nuanced understanding of how different factors contribute to resource allocation and inequality, and better inform policies and interventions to address these issues. Moreover, we may probe into more profound emergent goals, such as altruism amongst agents to model social welfare systems.

1.2 Goals

- Develop a flexible and accessible Python framework for the ***Sugarscape*** model
- Build and expand the model's rule set to include more specific agent behaviours and resource distributions
- Investigate altruistic agents
- Inform policies and interventions to address resource allocation and inequality issues in real-world social and economic systems

1.3 Success Criteria

To evaluate the success of this thesis, the following criteria will be used:

Flexible and Accessible Python Framework

1. The developed Python framework should provide a flexible and accessible platform for simulating the ***Sugarscape*** model.
2. The framework should be well-documented and easy to understand, allowing researchers and practitioners to use and modify it for their own investigations.
3. The framework should support various configurations and parameter settings, enabling users to explore different scenarios and hypotheses.

Expanded Rule Set and Agent Behaviors

1. The model's rule set should be expanded to include additional agent behaviours that reflect specific socioeconomic dynamics or resource allocation mechanisms.
2. Each added rule should be implemented effectively within the framework, accurately capturing its intended effects on agents and the environment.
3. The added rules should be flexible and customizable, allowing users to activate or deactivate them as needed.

Impact Analysis of Individual Rules

1. The impact of each individual rule on the model's behaviour should be systematically analysed and documented.
2. Various metrics and indicators should be used to assess the effects of each rule on resource distribution, inequality, and emergent goals within the model.
3. The analysis should provide insights into the specific mechanisms through which each rule influences the overall dynamics of the system.

Investigation of Emergent Goals

1. The framework should allow for the exploration of emergent goals, such as technological development and language spread, within the ***Sugarscape*** model.
2. The impact of different combinations of rules on the emergence and propagation of these goals should be examined and discussed.
3. The findings should shed light on the underlying processes and conditions that lead to the achievement or hindrance of these emergent goals.

Informing Real-World Policies and Interventions

1. The insights gained from the analysis and exploration of the ***Sugarscape*** model should contribute to the development of policies and interventions that address resource allocation and inequality issues in real-world social and economic systems.
2. The implications and recommendations derived from the research should provide actionable strategies for policymakers and stakeholders.
3. The research should highlight the potential benefits and limitations of using agent-based modelling frameworks like ***Sugarscape*** to inform real-world decision-making processes.

1.4 Thesis Structure

The structure of this thesis is heavily focused on the design of the framework and the building of rulesets. The results will be discussed alongside the addition of rules. Below, you will see an outline of the thesis structure:

- Brief Literature Review
- Framework Design Goals
- Ruleset Implementation and Case Study Results
- Conclusions

2 Project Specification: A Foreword on Social Systems

2.1 Introduction: What are Social Systems

Social systems are fascinating structures that permeate our natural world. They consist of interconnected agents driven by personal motives, collectively giving rise to complex macro-level behaviours. In formal terms, social systems can be defined as "patterns of interrelationships that exist between individuals, groups, and institutions, forming a coherent whole" [2]. Much like the way a house is constructed from stacking bricks, social systems are built by the summation of their smaller parts, resulting in intricate and reactive systems.

2.2 Why We Study Social Systems

Understanding social systems is crucial for personal development and informed decision-making. By unravelling the organization and dynamics of these systems, we gain insights into the social, economic, and political factors that shape our world. As C. Wright Mills puts it, studying social systems enables us to comprehend the larger historical scene and its significance for individuals' inner lives and external endeavours [3]. This awareness empowers us to navigate the complexities of our society and uncover the unknown.

2.3 Compressing Reality: Computers and Models

Advancements in computer technology have revolutionized the study of social systems. With the help of computational models, such as Cellular Automata (CA), we can now explore and analyze these complex systems in a controlled and inspectable manner. CA, a mathematical framework of discrete spatial and temporal systems with local interactions, provides an effective tool for capturing the essence of social systems. Through simulations and modelling, we can gain valuable insights into the behaviour and emergence of social systems.

2.4 The Sugarscape as a Social System

One such computational model, the Sugarscape, expands upon the principles of Cellular Automata by introducing the concept of "agents." In this bottom-up model, agents act based on rules and interact with their neighbouring agents and the environment. The Sugarscape, resembling a petri dish teeming with life, simulates the dynamics of social systems within a resource-limited environment. It captures the intricate interplay between agents' decision-making capabilities, social interactions, and the availability of resources.

2.5 What Can the Sugarscape Tell Us?

By employing this computational model, we can delve into the dynamics, patterns, and behaviours that emerge within resource-limited environments, thus gaining profound insights into the intricacies of social systems [4, 5].

3 Related Work and its Limitations: Building a Flexible and Accessible Python Framework for the Sugarscape Model

3.1 Similar Models

Several models, including NetLogo models such as Wolf Sheep Predation and Ants,[6] exist for simulating various scenarios. However, these models have limitations in terms of their outdated UI and limited expandability. Python-based models are sparse and often lack flexibility and expandability. [7]

3.2 Utility for a Python Framework

Listed below are the main benefits that a robust Python Framework would bring:

- **Improved flexibility:** A Python framework would allow for a more adaptable and customizable tool, addressing limitations and enabling the exploration of complex scenarios.
- **Improved scalability:** Python's scientific computing capabilities enable handling larger and more complex simulations than NetLogo.
- **Improved accessibility:** A Python framework would make the Sugarscape model accessible to researchers unfamiliar with NetLogo or who prefer Python, broadening its user base.
- **Improved analysis:** Incorporating data analysis and visualization tools from the familiar Python ecosystem enhances researchers' understanding and interpretation of simulation results, enhancing research quality.

3.3 Improving and Extending the Sugarscape Model

The Sugarscape model has limitations such as its simplicity and assumptions about agent behavior and static resource distribution.[4] Enhancements to the ***Sugarscape*** model, such as incorporating realistic agent behaviours (e.g., emotions, decision-making processes) and complex behaviours (e.g., language, memory), can improve accuracy and realism. A Python framework can facilitate these improvements and the exploration of alternative models.

3.4 Addressing the Limitations

Addressing the assumptions of purely ***Rational Agents*** can be done by incorporating more complex behaviours like ***Heterogeneous Preferences***, ***Bounded Rationality*** and ***Altruism***. Dynamic resource distribution, such as natural disasters or seasonal events, can also be considered. A Python framework allows for easier exploration of these alternative models.

3.5 Conclusion

Providing a flexible and accessible Python framework that allows researchers to easily create rulesets and extend functionality can allow for application to a broad range of research questions. The next chapter will discuss the design and implementation of the Python framework.

4 Utopia in the Scape: A Narrative of Ideal Behaviors, Aims and Design

4.1 Project Aims

4.1.1 Goals & Objectives

As discussed at the beginning of the dissertation, the goal is to present a robust and extendible Python Framework for the Sugarscape model, whilst additionally pushing and extending its functionalities to both validate its effectiveness as an extendible framework and address some of the limitations with the model.

4.1.2 Expected Outcomes

As a minimum, this thesis hopes to provide a framework that facilitates extensibility. Pushing the model is considered an experimental stretch and is not a core goal of the thesis.

4.2 Ideal Behaviours: Narrative Description

> *Deep in the heart of the Sugarscape, a newborn child entered the world. As the child grew, they learned the language of their tribe and became well-versed in the customs of their people. But as they matured, a restlessness grew within them, an insatiable desire to explore beyond the borders of their homeland.*
>
> *One day, the child ventured far into the mountains of sugar, leaving their home and their tribe behind. They traversed treacherous terrain and encountered new cultures, each with its own language and customs. The child learned quickly, assimilating with ease into each new group they encountered.*
>
> *But memories of the riches and abundance of their home on Sugar Mountain lingered in the child's mind, driving them to seek a way back to their birthplace. As they returned to their homeland, they discovered that their former people had grown complacent, hoarding their wealth and resources, while the child's experiences had taught them the value of trade and cooperation.*
>
> *With a new vision for their people, the child led a war against their former tribe, showing them the benefits of sharing resources and building together. Eventually, all the people of the Sugarscape assimilated into one, united by a common language and culture.*
>
> *But despite their newfound unity, the people of Sugarscape still faced challenges. Diseases spread quickly through their communities, threatening their very existence. However, they persevered, using their knowledge of building and trade to develop better infrastructure and medical practices to combat illness.*
>
> *In the end, Sugarscape became a thriving society, thanks to the courage and vision of a child who dared to venture beyond their borders and bring their people together.*

This story, though sentimental, exhibits complex, human-like behaviour. But this is a story that can be told within the confines of the Sugarscape. It is this level of complex behaviour, which I hope to achieve with this framework.

4.3 In Hindsight...

4.3.1 Achieving Project Aims

These proposed behaviours, such as movement, sex, ageing, combat, culture, trade, language, building, and disease, are fundamental aspects of human life. By incorporating them into the

Sugarscape model, we would be able to create more realistic simulations of social and economic systems. For example, movement and trade reflect the migration of people and the exchange of goods, while combat represents conflicts between groups with differing interests. Similarly, language and culture are essential to how humans interact and form communities, while building reflects the physical structures that support our societies.

Ageing and disease are also important factors that can affect resource allocation and inequality. As agents age and become less capable, they may be less effective at acquiring resources, leading to further inequalities. Meanwhile, the disease can have a devastating impact on populations and can disrupt the delicate balance of resource distribution. What we can do by modelling these phenomena is to better understand how they contribute to the dynamics of social and economic systems.

4.3.2 Motivation behind Behaviours

Overall, by incorporating these human behaviours into the Sugarscape model, we can create a more nuanced and realistic simulation of real-world systems. This will allow us to explore the complex interactions between agents and resources, and to gain insights into the root causes of inequality and other societal challenges.

4.4 Potential Implementation Challenges

Some of the limitations that may arise concern the complexity of the model. For one, the complexity of the model is often a crutch, as in the case of a study done on "Mixing beliefs among interacting agents"[8] where the research was limited by the complexity of the model, stating that an extension to the model that had a "historical perspective", would further the findings of the paper. However, this complexity can present difficulties in implementation, where rules become interconnected and form dependencies. Another issue that all Agent Based Model (ABM)s face is that models have to serve a specific purpose at a sufficient level of detail. [9] This seems opposed to the goals of this thesis, in which we are trying to build a general model. However, this is a natural byproduct of the attempt to create a *general framework* that is both easily accessible and modifiable. Since the model will be run mostly in small rulesets, the target behaviours will remain sufficiently isolated for appropriate study, However, the biggest challenge and a major limitation of ABMs is the computational expense. Due to ABM not focussing on an aggregate level but rather a process of constituent parts, ABMs are hard to simulate, and this may become a challenge considering Python is a relatively *"slow"* language, compared to its siblings.

4.5 Conclusion

Building a robust and extensible framework for ABMs can be a challenging task that requires careful consideration. One of the most critical aspects of this process is ensuring that the model adheres to a complex *"narrative"* of behavioural goals, which may involve balancing competing objectives and optimizing for multiple outcomes simultaneously. This a challenge, given the inherent complexity of human behaviour and the wide range of factors that can influence it. Furthermore, the computational costs of implementing such models can be significant, requiring careful attention to issues such as optimization, scalability, and efficiency. Nonetheless, by carefully balancing these various factors and leveraging advanced computational techniques, it is possible to build powerful and flexible simulations and generate valuable insights into human behaviour and social dynamics. Evidently, from *"SugarScape on steroids: Simulating over a million agents at interactive rates"*. [10]

5 Genesis: Designing and Implementing the Framework

5.1 Design Principles

There are various design principles that we could use to build this framework, before we can tackle constructing it, we first need to establish a good design standard that satisfies all our design goals.

5.1.1 Modularity and Extensibility

One of our target design goals is modularity and extensibility, to achieve this, we need an efficient way to easily extend behaviours. Since the ***Sugarscape*** is built around discrete rules, we can design the system so that rules can be added without needing to modify the underlying code. That is, rules should be able to be created, ideally in external files, and then registered on the ***Sugarscape*** model before the simulation is executed.

To achieve this, a **dynamic variable system** should be used. The framework interface should allow for variables to be dynamically allocated during initialisation and run-time, removing the need for editing underlying class data structures. This would lead to an "On-the-fly" behaviour resolution which would make developing rules trivial, whilst allowing for a full expression of logic.

5.1.2 Robustness and Error Handling

Another major consideration should allow for the framework to be transparent about errors. Since users will create the rules themselves, certain aspects of error handling must be delegated to said users. However, if the framework provides a suite of error-logging tools, this would provide significant ease of use for those developing, testing, and interfacing rules.

5.1.3 Performance Optimization

Optimisation is also a significant factor in the design of this framework. However, optimisation will not be a major focus in this thesis in its current form. However, the framework ideally should simulate at high speeds, though given the nature of rule stacking and user-created rules, some of the optimisations are delegated to the users. We can, however, optimise certain background processes, and provide optimised utility functions that may be regularly needed; such as currently theoretical ***GetAllNeighboursAroundAgent()*** which as described returns all the neighbours around an agent on the ***Sugarscape***.

5.1.4 User-Friendly Interface

The last, and most important aspect of the framework's design should be the accessibility and friendliness of the framework. The framework should provide a suite of simple functions to construct simulation scenarios, run them, and analyse them.

Additionally, the framework should have an extensive plotting system to easily visualise and graph variables pertaining to the simulation. A Graphical User Interface (GUI) would be the most desirable outcome which could be applied to initialising models and viewing results; however, rules will require an IDE to write. Therefore, a GUI may only be useful for displaying results.

5.2 Architecture and Implementation

5.2.1 Model-View-Controller (MVC) Design Pattern

Separation of concerns: MVC provides a clear separation between the model, view, and controller, making the framework more modular, maintainable, and easier to understand and modify. **Reusability and extensibility:** Components can be designed to be reusable and extensible, allowing for flexibility in adapting the framework to different requirements and future enhancements. **Testability:** MVC promotes testability by decoupling the components, enabling easier unit testing and integration testing of the framework. **Scalability and maintainability:** MVC provide a scalable architecture that can handle the growth and complexity of Sugarscape models, allowing for easier maintenance and future enhancements. **User interface flexibility:** The view and controller components can be tailored to different user interface requirements, such as command-line or graphical interfaces, while the model remains the same. Collaboration: MVC promotes collaboration among developers by allowing teams to work simultaneously on different components without conflicts, improving productivity during development. [11]

5.2.2 Overall System Architecture

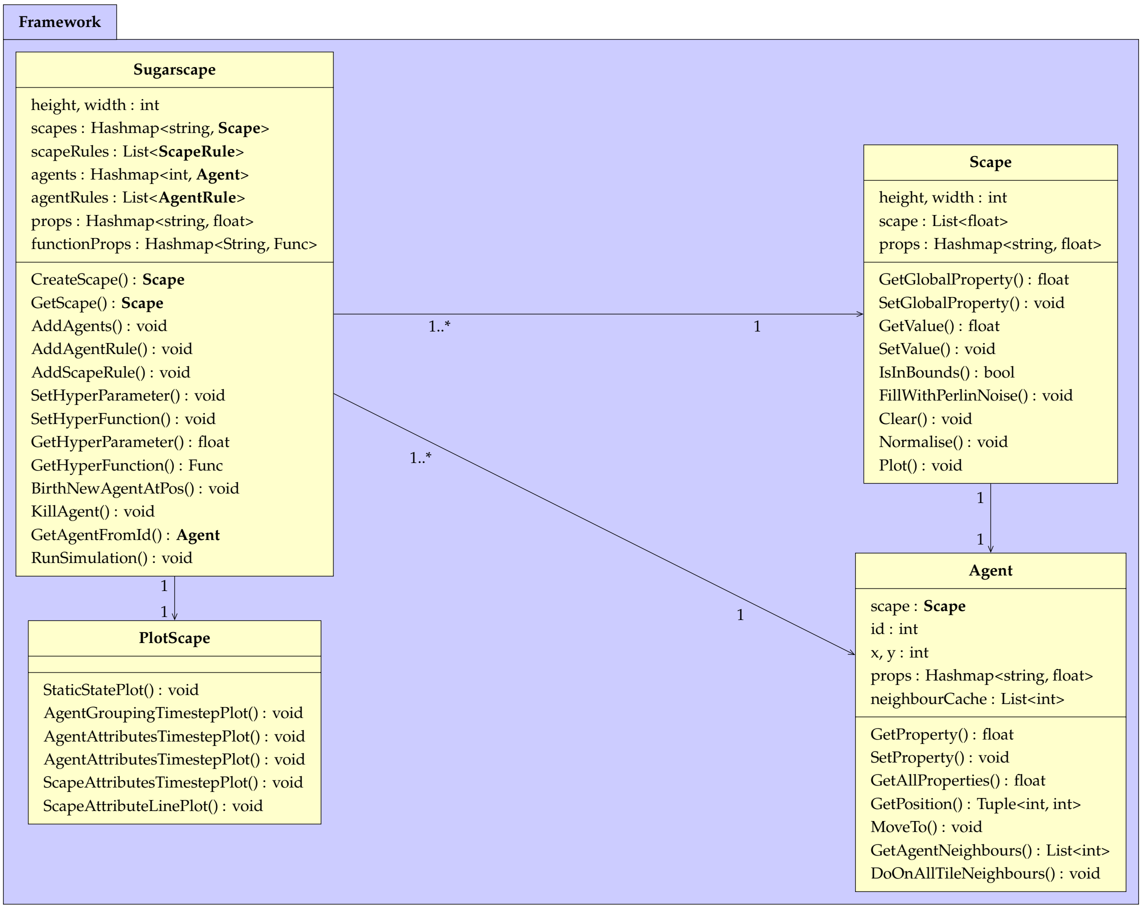


Figure 5.1: Simplified UML diagram describing core relationships between systems

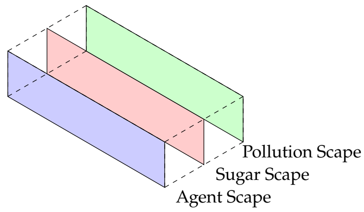


Figure 5.2: An illustration of scape stacking

5.2.3 Core Components

Agent Class

The agent class represents individual agents that exist within the environment. It encapsulates crucial attributes that enable agents to interact with their surroundings. To this end, it has a ***scape*** attribute, which serves as a reference to the ***Scape*** object on which the agent resides, which facilitates a connection between the agent and its environment. Furthermore, the agent has attributes ***x*** and ***y*** attributes, denoting its position within the environment.

The ***GetPosition()*** method provides the agent's current position in the form of a tuple, representing its coordinates within the environment. When an agent needs to relocate within the environment, the ***MoveTo()*** method facilitates smooth movement to a new position. The ***GetAgentNeighbours()*** method returns a list of neighbouring agent positions, enabling agents to interact with their immediate surroundings. Lastly, ***DoOnAllTileNeighbours()*** is a utility method that allows agents to execute specific actions on all neighbouring tiles.

Scape Class

The scape class represents a 2D lattice for a certain singular property. So a Sugarscape with the resources **{Water, Tree}** would have a separate scape for **Water** and **Tree** respectively. This separation allows for easy visualisations of specific layers and reduces the complexity of referencing resources when building rules. A scape is represented by a 2D array of float values but internally this uses a single-dimensional array. Float values were chosen as all meaningful concepts can be represented as floats, whereas having a dynamic datatype system would introduce a lot of overhead whilst being rarely utilised.

Amongst the host of functions, the ***Scape*** class provides, the most important are ***GetValue()***, ***SetValue()*** which set and retrieve float values on the scape respectively. The ***Plot()*** method creates and displays a plot of the current scape as a heatmap. For this use the ***Normalise()*** function to normalise the float values in the 2D array between a discrete range. This is used extensively for heatmap plots to ensure readability. The other methods listed are self-explanatory in function.

Sugarscape Class

The ***Sugarscape*** class is the main orchestrator class which handles creating multiple resource scapes and agent scapes, respectively. A key design decision was to keep the environment of the Sugarscape discretely separated for all established finite cell states. This design decision came due to difficulties in initial prototype designs where representing internal data structures for the scapes in a way which could allow for easy indexing as well as easy interfacing was shown to be a non-trivial problem. Separating resources into separate scapes, though introducing a minor overhead in the simulation loop, proved to be much simpler to work with. See Fig. **??**.

The ***Sugarscape*** class notably contains the functions ***AddAgentRule()*** and ***AddScapeRule()***, these functions are essential to registering user-created rules, which can be imported from external Python files and passed into the Sugarscape model.

5.2.4 Data Structures and Algorithms

Grid Representation

As mentioned previously, the environment, which requires to be modelled as a grid lattice of discrete states, is encapsulated by the ***Scape*** object. Internally the grid representation follows that of a one-dimensional array of floating point numbers, which is read as a two-dimensional array at the interface level. This was chosen over the alternative of a dynamic datatype system because the floating numbers are more versatile in terms of data representation. Where a boolean datatype perfectly represents *True* and *False*, floats can represent likewise with *1.0* and *0.0*. So, there is a delegation of responsibility for users to handle representation within the limitation of floats.

Spatial Indexing

Spatial Indexing uses the conversion of ***Cartesian Coordinates*** to indexes. The index of the position is simply found by the equation:

$$i = (x + y)\mathrm{w}$$

where w is the width of the scape. This means that scapes are limited to dimensions of w^2 and cannot be irregular in shape. The bounds of the scape do not connect to each other, so an agent from the far east side of a scape cannot cross into the far west. This design decision is necessary as keeping agents with a defined boundary allows us to analyse the spatial distribution of agents and their movements within the Sugarscape. This can provide valuable insights into patterns of movement and interactions, especially if the edges represent meaningful boundaries or barriers. Ideally, however, there would be an option to enable a boundless Sugarscape.

Optimisations

To facilitate efficient retrieval of neighbouring positions, the agent employs the ***neighbourCache*** attribute, a list that acts as a cache. In addition, the agent class includes the ***props*** attribute, a hashmap that stores various properties associated with the agent.

However, the implementation is still slow, and it does not scale well with additional rules or increased agent populations. This is partly due to limitations with Python, which could be overcome by integrating ***Cython*** a suite of tools that allow Python to be complied with considerably faster C. Some of the contributions to the limited speed of the simulation can also be attributed to saving state snapshotting. Due to the large amounts of data being copied during each snapshot, and the regularity of snapshots for each epoch, this adds considerable overhead to a simulation run.

Buffers and Update Operation Order

Buffers are used to prevent the paradoxical issue of agent ghosts. When updating the agent's positions based on the ruleset, there are two options: Stale update and live update. A stale update is where you update the agents based on the knowledge of the previous epoch end state, it then follows that a live update would be updating the agents based on the knowledge of the previous update. Both have their drawbacks, but for this framework, I chose to implement the latter.

This means buffers for agent birth and death are required, as creating new agents whilst updating current ones can potentially create an infinite feedback loop or a myriad of undesired edge cases. These buffers facilitate the required pause for agent population changes, ensuring agent lifecycles are resolved at the end of the agent update phase of the simulation lifecycle.

Save States

Save States are taken every epoch. Internally, they are represented by a Tuple of two Time series. The former Timeseries of scape data evolution and the latter is a time series of agent data evolution. Each time series is represented as a list of scapes and agent populations. All the relevant data is copied at the end of each simulation cycle, and stored for later retrieval and analysis by the provided tools, or by manual means.

Dynamic Properties

The dynamic properties accessible in the framework is one of the most important aspects, making it easy and fast to prototype and develop rules. Agents, Scapes and the Sugarscape as a whole contain various different types of dynamic properties which will be specified below:

```
s = Sugarscape(100)
...
s.SetHyperParameter("initial_endowment_range", (50, 100))
s.SetHyperParameter("disease_infection_chance", 0.005)
s.SetHyperParameter("disease_pool", [{"name":"sugardeath", "string": "010110", "
    infection_chance": 0.005}])
...
s.SetHyperFunction("cultural_similarity_function",functions.CulturalSimilarityFunction)
s.SetHyperFunction("welfare_function", functions.GetWelfare)
```

Code 5.1: Example of setting dynamic global properties

In the figure above you can see how the interface allows you to initialise hyperparameters with a string reference and a dynamic variable value. These can then be utilised when creating rules as you will see in the following section. Agents allow for dynamic properties which will be shown later.

Simulation Lifecycle

The lifecycle of the simulation is constructed as follows. The order allows for all the required elements of the Sugarscape to be resolved without any undesired clashes between processes. To this end, the structure maintains the live complexities of the system, whilst providing a structure to each epoch.

This lifecycle is reflected in how rules are added. See below:

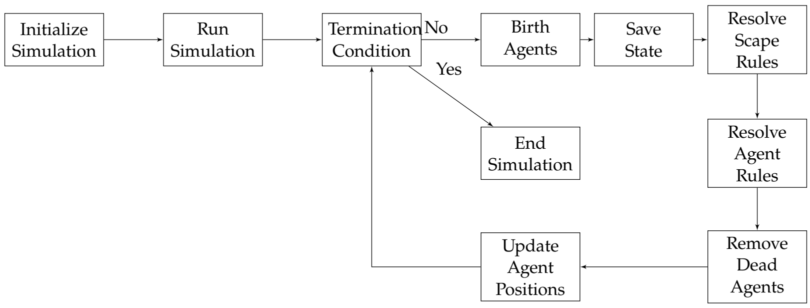


Figure 5.3: Life cycle of the sugarscape simulation

```
def Init(sugarscape: Sugarscape, agent: Agent):
    max = sugarscape.GetHyperParameter("max_vision")

    if agent.GetProperty("vision") == None:
        agent.SetProperty("vision", random.randint(1, max))

def Step(sugarscape: Sugarscape, agent: Agent):
    ...
    # Move to the best sugar value
    newx, newy = bestSugarValue[1][0], bestSugarValue[1][1]
    if bestSugarValue[0] != -1 and agent.scape.IsInBounds(newx, newy) and agent.scape.IsCellDefault(newx, newy):
        agent.MoveTo(newx, newy)
```

Code 5.2: Example of lifecycle in rule file

You can see that there are distinctive phases. ***Init()*** gets executed during the **Simulation Initilisation** phase and the **Birth Agents** phase. By hooking onto these events, you can create complex behaviour very simply. In the figure you will also notice the use of dynamic variables on agents. Here we can set the property ***"vision"*** in the init, and it will be accessible from all other rules. This is the utility power of this property system. Instead of needing to extend an agent class and add variables in the constructor, you can instead initialise them on the fly allowing for rapid rule development.

5.3 Performance Evaluation

The performance evaluation results indicate a notable difference in initialization time compared to execution time as the agent count increases. The initialization time shows an increasing trend, surpassing the simulation time by a factor ranging from 1.52 to 8.66. This suggests that the model's initialization process becomes more resource-intensive and time-consuming with a larger number of agents. Consequently, optimizing the initialization phase could be beneficial in reducing the overall computational burden and improving the efficiency of the Sugarscape simulation.

Agent Count	Init Time (ms)	Exec Time (ms)	Memory Usage (MB)	Init-Exec Ratio
100	1800	1184	12.56	1.52
500	8900	2265	15.37	3.93
1000	16700	3384	19.01	4.94
5000	88300	10181	48.82	8.66

Table 5.1: Performance Evaluation of the Sugarscape Model by Agent Count

6 Building The Universe: Constructing the ruleset that governs agents

6.1 Bringing Life to the Sugarscape

To continue further we need to build the ruleset. Using the framework now established, we can start creating rules to emerge from the narrative that we constructed in Chapter 4. See 4.2. Below are the core behavioural features, inspired by the work done in Growing Artificial Societies by Joshua M. Epstein and Robert Axtell.

Agent Metabolism Rule M_e

- Start with an initial endowment of *sugar* and *spice*.
- Collect all sugar and spice at the current site.
- Consume (metabolise) all stored sugar and spice.
- If there is no more sugar or spice to metabolise, die.

Agent Movement Rule M_o

- Look out as far as vision allows and find the cell with the best welfare
- Move to cell with the best welfare

The welfare function is denoted as:

$$W = w_1^{\frac{m_1}{M}} * w_2^{\frac{m_2}{M}}$$

where 1: sugar, 2: spice, w: resource wealth m: resource metabolic rate $M = m_1 + m_2$. The welfare function can be described as a utility function that when maximised over neighbouring sites maximises the gain of the prioritised resource.

Agent Aging Rule A_g

- Every unit of time, age one year.
- If age is greater than a predetermined life span, die.

Scape Seasonal Growback Rule S_e

- Every fixed span of time units flip seasons
- If season is *Winter*, reduce the sugar grow back rate
- Every unit of time, grow sugar back by the sugar grow back rate

Agent Reproduction Rule R_e

- If within fertile age range, look for a fertile neighbour as far as vision permits
- If the fertile neighbour is the opposite sex, and there is a suitable site for child breed
- Mix agent properties by Mendelian inheritance to produce offspring

Scape Pollution Rule P_o

- When sugar is collected or consumed by an agent, create pollution.

Scape Diffusion Rule D_i

- Diffuse pollution to neighbouring cells every unit of time.

Agent Cultural Transmission Rule C_u

- For every neighbour select a random flag and compare it to self
- If neighbour disagrees, adopt neighbours flag

Cultural transmission is modelled using binary strings. So, a flag would be a bit of this binary string. Agents use a utility function to determine whether other agents are in the same cultural group. This function is 0 when the strings match and 1 when they are maximally different.

Agent Combat Rule C_o

- Look out as far as vision allows and ignore sites with members of my own tribe
- Ignore all sites occupied by members of my own tribe
- Throw out all sites occupied by members of other tribes who are wealthier than me
- If the site has an agent next to it (vulnerable to retaliation), ignore it
- If there is an agent at the site and they are of a different tribe, ignore it
- If the agent is the enemy, kill the agent and take all sugar

Agent Trading Rule T_r

- Compute MRS. Look out as far as vision allows and find a neighbour with $MRS! = MRS_{self}$
- Spice flows from high MRS to low MRS, sugar flows from low to high
- Calculate the geometric mean of MRS to get price p
- If $p > 1$ then trade p units of spice for 1 unit of sugar
- If $p < 1$ then trade $\frac{1}{p}$ units of sugar for 1 unit of spice
- If trade makes agents better off and does not cause MRSs to cross, then trade

The MRS is simply a ratio of $\frac{m_2}{m_1}$ where m_1: time to spice death, m_2: time to sugar death.

Agent Disease Rule D_i

- Randomly infect the agent from the disease pool
- If not infected, randomly infect the agent from neighbours
- If the disease is a substring of the immune string, the agent is immune
- If the agent is not immune, find the closest hamming distance substring to the immune string and flip the first bit of the substring

6.2 Conclusion

In this chapter, we have constructed the ruleset that governs the agents in the Sugarscape model, building upon the narrative we developed in Chapter 4. We have outlined various behavioural features inspired by the work done in Growing Artificial Societies by Joshua M. Epstein and Robert Axtell. With the ruleset in place, we have laid the foundation for simulating and exploring the dynamics of the Sugarscape model, incorporating various elements of agent behaviour, social interactions, environmental factors, and cultural exchange. In the next chapters, we will delve deeper into analyzing the emergent patterns and outcomes that arise from these rules, further expanding our understanding of complex social systems through a case study.

7 Trial By Sugar: Exploring Population Growth Dynamics in Japan's Aging Society

7.1 Introduction

The dynamics of population growth and its impact on society have become crucial areas of study, particularly in the context of ageing populations in modern countries. Japan, in particular, has been facing significant challenges due to its rapidly ageing society, with implications for various sectors such as healthcare, labour markets, and social welfare. [12]

In this case study, we will explore Japan's rapidly ageing society using the Sugarscape model. Our objective is to assess the effectiveness of policy interventions and ultimately inform them. By adapting the model to represent Japan's demographic landscape, introducing altruistic agents and conducting simulations, we aim to gain insights into the complex interactions between population dynamics, societal factors, and policy choices.

7.2 Applicability of Sugar Scape Model

The Sugarscape model is useful for studying population dynamics and simulating individual behaviours within a population. However, it has limitations as it doesn't fully capture the physiological and cognitive changes of ageing, and it doesn't adequately represent societal factors and policies that influence population dynamics. Factors like healthcare, social welfare, culture, and economy have significant impacts on ageing populations but are not well-represented in the Sugarscape model.

7.3 Modifying the Sugar Scape Model for Aging Population Study

7.3.1 Resource Distribution

To represent the varying needs of different age groups in the Sugarscape model, we can increase the metabolism as agents age. Older agents have higher sugar requirements due to physiological changes associated with ageing, such as decreased metabolism and energy efficiency. [13] By increasing the metabolism of older agents, we simulate their higher sugar needs, reflecting real-world scenarios where older individuals require more resources to support their ageing bodies and daily activities.

7.3.2 Disease

To incorporate disease susceptibility, agents can become more vulnerable to diseases as they age. This reflects the real-world observation of weakened immune systems in older individuals. By adjusting the susceptibility to disease based on age, we can simulate the increased risk and impact of illnesses on older agents in the Sugarscape model.

7.3.3 Reproduction and Mortality

To reflect declining birth rates in ageing populations, we adjust the reproductive rates in the model. This accounts for the decrease in fertility among older individuals. [14]

Additionally, we incorporate increasing mortality rates with age to simulate the higher mortality risk among older individuals. This adjustment aligns with the observation that mortality rates tend to rise as individuals grow older. [15]

7.3.4 Social Support Mechanisms

We incorporate social support mechanisms to simulate healthcare, elder care facilities, and social welfare programs. Altruistic agents are introduced to represent individuals who provide support and care to others in the population, as a percentage of the agent populus.

By including these altruistic agents, we can attempt to simulate the influence of social welfare programs on the dynamics and well-being of the population.

7.3.5 Intergenerational Dynamics

Modelling intergenerational relationships in the Sugarscape model can be challenging due to its focus on individual agents and their local interactions. The model does not explicitly capture the complex dynamics and influences that occur within families or across different age groups in society.

To simulate the impact of intergenerational interactions, we can use the properties of cultural transmission. As agents age, we can introduce random noise to their cultural tags, a **degenerative factor**, which will lead to isolation for agents with "weak" social circles.

By incorporating this degeneration factor and considering proximity or social connections, we can capture some aspects of intergenerational dynamics in the model. however, the Sugarscape model may still have limitations in fully capturing the complexity and nuances of intergenerational relationships in real-world scenarios.

7.3.6 Formalised Model Modifications

Modified Aging Rule A_g^2

- Every unit of time, age one year.
- If age is greater than a predetermined life span, die.
- Increase metabolism, random mortality chance, random cultural degeneration factor
- Decrease immune system strength, and vision with ageing

Agent Atrulism Rule A_l

- If above the adulthood age threshold, allow the agent to become altruistic
- If designated reciprocative, and resource welfare is above a threshold, allocate these resources as shared
- For all neighbours as for as vision permits, find the one with the lowest ***Age-Adjusted Welfare Index***
- Transfer sugar/spice to this neighbour
- If the majority of neighbours are reciprocating, randomly designate altruism on the agent
- If the majority of neighbours are defective, randomly designate defective on the agent
- If resource welfare falls below a threshold, agent becomes defective

> The Age-Adjusted Welfare Index (AAWI) is simply the welfare-to-age ratio. It represents the agent's well-being in relation to its age, accounting for the changes in resource requirements and metabolic rates throughout the agent's lifespan.

This altruism rule incorporates elements of the Prisoner's Dilemma game[16], where agents adjust their behaviour based on the reciprocity of their neighbours.[17] The allocation of resources as shared between deficient neighbours, the selection of neighbours based on AAWI, and the random designation of altruism or defection, aims to capture the dynamics of cooperation and defection observed in the Prisoner's Dilemma.

Agents designated as reciprocative allocate their resources as shared, promoting cooperation with neighbours who have lower AAWI. Random designation of altruism or defection based on the majority behaviour of neighbours introduces an element of strategy and adaptability to the model.

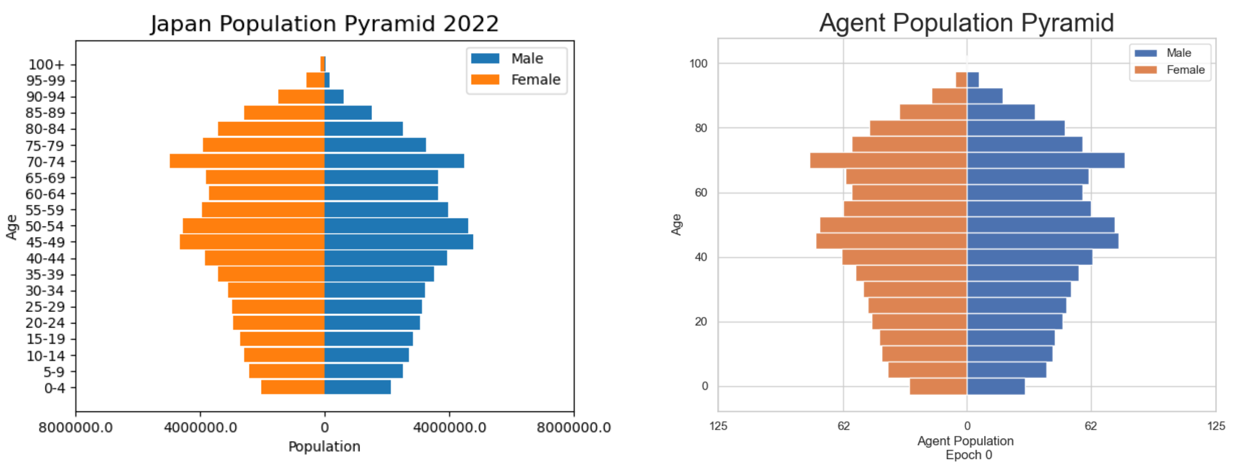
The condition for becoming defective when resource welfare falls below a threshold represents a scenario where agents prioritize their own survival and well-being over cooperation, mimicking the self-interest aspect of the Prisoner's Dilemma. Known as a *"TIT FOR TAT"* system. [16]

These modifications add complexity and strategic decision-making to the altruistic interactions in the Sugarscape model, allowing for the exploration of emergent dynamics and the study of how reciprocity influences the distribution of resources and the overall well-being of the agents.

7.4 Observations and Insights

7.4.1 Experimental Setup

For this case study, we step up the simulation with the following parameters, to reflect the socioeconomic landscape of Japan, producing a rising ageing population in line with empirical data. To achieve this, we pulled the 2022 demographic data as a Population Pyramid. Then by splitting up the agents into their respective age groups, we could seed the simulation with the current ageing population conditions Japan faces today. The data was sourced from *populationpyramids.net* [18]



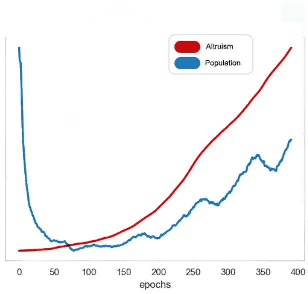
(a) Population (123,951,692) Pyramid of Japan C.2022. (b) Agent Population (1000) Pyramid at Epoch 0

Figure 7.1: Comparison of Japan demographic and Sugarscape demographic on initialisation.

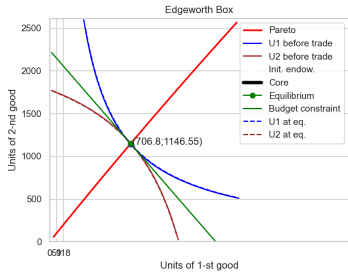
With this experimental setup, we establish the baseline that by 2500 [19], the population of Japan will have disappeared. Given this, as an epoch in our simulation is one year, we start with the conditions that collapse society in roughly 478 epochs. By tuning the parameters, we can achieve this cutoff point that is representative of this number. The rulesets chosen for the simulations runs were:

$$M_e, M_o, A_g, S_e, R_e, C_u, C_o, T_r, D_i$$

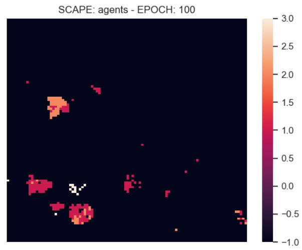
7.4.2 Simulation results



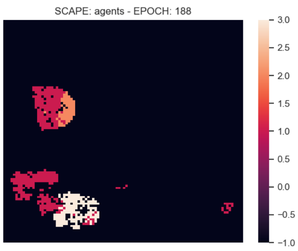
(a) Altrusm vs Population



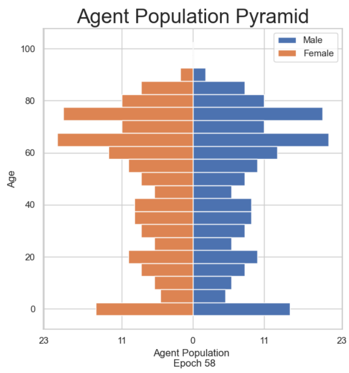
(b) Trading Edgeworth Plot



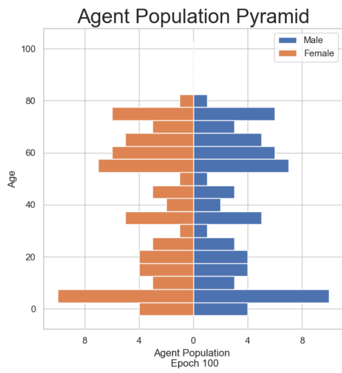
(c) Social Clumping, E100



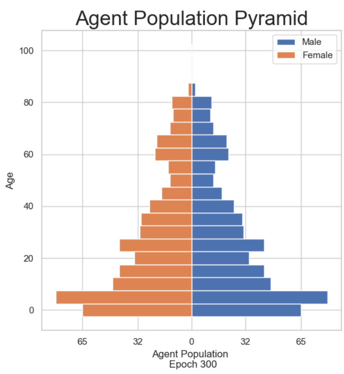
(d) Social Clumping, E188



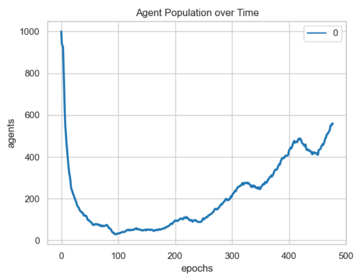
(e) Pop Distribution, E58



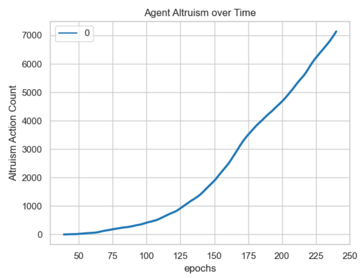
(f) Pop Distribution, E100



(g) Pop Distribution, E300



(h) Agent Pop over years



(i) Agent Altruism over years

Figure 7.2: 4 x 4

7.5 Conclusion

We defined and conducted simulations using the modified Sugar Scape model to explore population dynamics in the context of Japan's ageing society. The simulations provided valuable insights into the interactions between various factors and their influence on population dynamics which we discuss in the next and final chapter.

8 Results, Discussion and Conclusions

8.1 Discussion: Case Study

8.1.1 Evaluation of Results

The modified Sugar Scape model simulations produced interesting findings on the influence of altruism, metabolism, and resource distribution in an ageing population context.

Age Structure Shift: The simulations mirrored Japan's ageing population, showing a rise in older individuals and a decline in younger ones.

Fertility Decline: Rates dropped as agents aged, reflecting real-world trends. This led to a smaller working-age population and a higher dependency ratio. Work is akin to extracting sugar.

Importance of Social Support: Altruistic agents (representing caregivers and healthcare providers) positively affected overall welfare, especially among older individuals. Robust social support systems are crucial for addressing ageing population challenges.

Altruism Impact: Introducing altruism prevented population decline by extending the lifespan of older individuals. Adequate resource allocation from altruistic agents sustained their well-being and facilitated a population boom by maintaining status quo and giving time for younger agents to accumulate wealth and acquire better conditions for bearing children.

Resource Allocation: Insufficient altruistic donations led to population struggles and hindered recovery. Sufficient resource allocation and support systems are vital for population stability.

Metabolism and Carrying Capacity: Higher metabolism reduced the environment's carrying capacity. This signifies the strain older individuals place on society, demanding more resources than the environment can sustain.

Research suggests that diminishing energy returns on investment (EROI) [20] and increased resource demands by older individuals can strain societal resources, potentially leading to societal collapse. [21]

8.1.2 Policy Implications

This case study suggests that policies promoting altruism and social support mechanisms are crucial for addressing the challenges posed by ageing populations. To address the challenges posed by population aging, Japan has implemented a range of initiatives. These include policies like the New Angel Plan(1999) and the Plus One Policy(2009), which aim to facilitate child-rearing through measures such as funding childcare facilities and reducing educational costs. The Abenomics economic package focuses on increasing female labor force participation, technological innovation, and reducing healthcare costs. Additionally, Japan is considering raising retirement and pension eligibility ages, relaxing immigration restrictions, investing in children's education, and implementing technological advancements like robotic automation. These solutions aim to expand the productive workforce years, ease the fiscal burden of pensions, and ensure older adults can work and contribute effectively.[22] These align with the importance of social support mechanisms identified in the simulation. However, the simulation emphasizes the need to increase resource allocation to these areas, which could provide further guidance for policy decisions.

Japan's efforts to encourage intergenerational interactions and community engagement also align with the simulation's findings on the positive impact of intergenerational dynamics. However, the simulation highlights the potential of promoting altruistic behaviour, which Japan could further incorporate into policies by incentivizing volunteerism or community involvement.

Furthermore, the simulation underscores the significance of adequate resource allocation to healthcare, elder care, and social welfare programs. Japan could consider allocating more resources to meet the increasing needs of the ageing population and ensure their well-being.

By drawing parallels between the simulation findings and Japan's existing policies, policymakers can identify areas of alignment and areas that may require adjustments or additional measures. This can inform policy decisions and contribute to the development of comprehensive strategies to effectively address the challenges of an ageing population.

8.2 Discussion: Framework, Our Contribution & its Limitations

The framework has been shown to be useful in studying societies on an artificial landscape. This case study contributes to the understanding of population dynamics in the context of an ageing population using the modified Sugar Scape model. It highlights the importance of incorporating societal factors, such as social support mechanisms, in population models. However, it is important to acknowledge the limitations of the model. The small population size (1,000 agents) and various assumptions made in the simulation may not fully capture the complexity of real-world scenarios.

8.3 Conclusion and Significance: Future Research & Limitations of this project

As a final conclusion: future research in this field can focus on refining the model to better represent the characteristics of specific populations, and incorporating additional factors that influence population dynamics. Furthermore, exploring alternative modelling approaches and integrating real-world data can enhance the accuracy and applicability of the simulations.

Two major limitations are the missing geographical context and unrepresentative population size. The former was not explored, due to the latter which was not possible due to a lack of optimisations in the system. Moving forward, focusing on optimising the framework to allow for quicker and larger simulation populations and scopes would expand the scope of research possible. However, a move from Python, to underlying C internally would be a recommended approach concerning optimisation.

It is clear the Implementation challenges we discussed in Chapter 4, have surfaced during this case study. There is an aspect of the model currently, which will always fail the capture the complexities of real life. As mentioned in Chapter 1, ABM's effectively *compress reality*, but this compression is lossy. Models will never be able to capture the full extent and complexities of real-world social systems, however, the more rules we add, the closer we can come to increasing the realism a,d utility of models in informing real-world actions and policies. This reiterates the main motivation, building this Python framework acts as a first step in providing an extendible tool that can allow researchers to create more complex Sugarscape simulations, and highlight the likely near-unlimited capacity for this model to capture social systems from an emergent basis.

The significance of this study extends beyond the field of computer science, as it provides insights into the challenges faced by countries with ageing populations, such as Japan. This model could easily be applied to more research questions. Such endeavours will provide valuable insights for developing tailored policies and interventions to address the challenges posed by ageing populations, ultimately contributing to the well-being and sustainability of societies.

References

[1] Nigel Gilbert and Steven Bankes. "Platforms and methods for agent-based modeling". In: *Proceedings of the National Academy of Sciences* 99.suppl_3 (2002), pp. 7197–7198.

[2] *Social System Definition amp; meaning*. URL: `https://www.merriam-webster.com/dictionary/social%5C%20system`.

[3] C Wright Mills. *The sociological imagination*. Oxford University Press, 2000.

[4] Joshua M Epstein and Robert Axtell. *Growing artificial societies: social science from the bottom up*. Brookings Institution Press, 1996.

[5] Rosaria Conte and Mario Paolucci. "On agent-based modeling and computational social science". In: *Frontiers in psychology* 5 (2014), p. 668.

[6] Seth Tisue and Uri Wilensky. "Netlogo: A simple environment for modeling complexity". In: *International conference on complex systems*. Vol. 21. Citeseer. 2004, pp. 16–21.

[7] Graeme Stuart. *Pysugarscape*. June 2013. URL: `https://pypi.org/project/pysugarscape/`.

[8] Guillaume Deffuant et al. "Mixing beliefs among interacting agents". In: *Advances in Complex Systems* 3.01n04 (2000), pp. 87–98.

[9] Eric Bonabeau. "Agent-based modeling: Methods and techniques for simulating human systems". In: *Proceedings of the national academy of sciences* 99.suppl_3 (2002), pp. 7280–7287.

[10] Roshan M D'Souza, Mikola Lysenko, and Keyvan Rahmani. "Sugarscape on steroids: simulating over a million agents at interactive rates". In: *Proceedings of Agent2007 conference. Chicago, IL*. 2007.

[11] James Bucanek. "Model-view-controller pattern". In: *Learn Objective-C for Java Developers* (2009), pp. 353–402.

[12] Hamid Faruqee and Martin Mühleisen. "Population aging in Japan: demographic shock and fiscal sustainability". In: *Japan and the world economy* 15.2 (2003), pp. 185–210.

[13] CJK Henry. "Mechanisms of changes in basal metabolism during ageing". In: *European journal of clinical nutrition* 54.3 (2000), S77–S91.

[14] Karen Glaser et al. "Fertility decline, family structure, and support for older persons in Latin America and Asia". In: *Journal of Aging and Health* 18.2 (2006), pp. 259–291.

[15] Elke Raum et al. "Changes of cardiovascular risk factors and their implications in subsequent birth cohorts of older adults in Germany: a life course approach". In: *European Journal of Preventive Cardiology* 14.6 (2007), pp. 809–814.

[16] Robert Axelrod. "Effective choice in the prisoner's dilemma". In: *Journal of conflict resolution* 24.1 (1980), pp. 3–25.

[17] Toh-Kyeong Ahn, Elinor Ostrom, and James M Walker. "Heterogeneous preferences and collective action". In: *Public choice* 117.3-4 (2003), pp. 295–314.

[18] *Population pyramids of the world from 1950 to 2100*. 2022. URL: `https://www.populationpyramid.net/japan/2022/`.

[19] Brandon A. Weber. *At current birth rates, the Japanese population will disappear by 2500*. Apr. 2022. URL: `https://bigthink.com/culture-religion/the-japanese-population-is-crashing-faster-than-any-other-country-why-2/`.

[20] David J Murphy and Charles AS Hall. "Year in review—EROI or energy return on (energy) invested". In: *Annals of the New York Academy of Sciences* 1185.1 (2010), pp. 102–118.

[21] Charles AS Hall, Stephen Balogh, and David JR Murphy. "What is the minimum EROI that a sustainable society must have?" In: *Energies* 2.1 (2009), pp. 25–47.

[22] Andrew Stawasz et al. *Japan's age wave: Challenges and solutions*. Dec. 2018. URL: `https://cepr.org/voxeu/columns/japans-age-wave-challenges-and-solutions`.

Acknowledgments

Hey Marcos!

I just wanted to drop you a quick note to say a massive thank you for all your help and guidance throughout this project. Seriously, I couldn't have done it without you!

You've been an amazing supervisor, always there to answer my questions and provide the right direction. Your expertise in modelling and simulation is seriously impressive, and I've learned so much from you. I mean, who knew the Sugarscape model could be so fascinating?

I really appreciate how you've pushed me to explore different behaviours and expand the rule set. It's been a real eye-opener to see how those specific factors impact resource allocation and inequality. Your suggestions have definitely made this project more nuanced and interesting.

I'm genuinely grateful for the time and effort you've put into mentoring me. You've been patient, and approachable, and your passion for this subject is contagious. Working with you has been an awesome experience, and I feel lucky to have had such a supportive supervisor like you.

Moreover, I wanted to express my gratitude for your unwavering support and understanding during my personal journey. You've created a safe and inclusive environment that has allowed me to grow not just as a researcher, but also as an individual. Through your mentorship, I've gained valuable insights and learned to overcome personal challenges, which have positively influenced my overall well-being and productivity.

Once again, thank you so much for everything! This project has been a fantastic learning journey, and I'm excited to see how the insights we've gained can contribute to future research in the field.

Take care and stay awesome!

Cheers,
David