Deep Reinforcement Machine Learning as a Driver of Agent Decision-Making in Agent-Based Models of Coupled Natural and Human Complex Systems

A Thesis Presented

by

Kevin Allen Andrew

to

The Faculty of the Graduate College

of

The University of Vermont

In Partial Fulfillment of the Requirements
for the Degree of Master of Science
Specializing in Computer Science

August, 2023

Defense Date: July 12th, 2023
Thesis Examination Committee:

Asim Zia, Ph.D., Advisor
Scott Hamshaw, Ph.D., Chairperson
Donna Rizzo, Ph.D.
Safwan Wshah, Ph.D.
Cynthia J. Forehand, Ph.D., Dean of Graduate College

Abstract

Agent-based models are becoming increasingly useful in studying the behavior of real-world complex multi-agent systems; however, one of the outstanding challenges in the modeling of coupled natural and human systems is the dearth of techniques for creating agents that are able to learn from their past failures and successes, as well as compounded environmental and social uncertainties. This research has been focused on the integration of traditional agent-based modeling with machine learning methodologies for modeling agent decision-making and its recursive impacts on economic, environmental, and societal outcomes, feeding into the dynamic co-evolution of the coupled natural and human system state variables within simulated worlds, resulting in the development of two models incorporating and exploring the use of deep reinforcement machine learning as a driver for decision-policy making in agent-based models.

The first of these models is a model of agricultural land use and the adoption of agricultural best-management practices by farmers in response to ecological and economic scenarios as a result of municipal regulation and variance in the occurrence of extreme weather events. The primary study area used for the model is a region of the Missiquoi Bay Area of Lake Champlain in Vermont, containing 480 farmer agents corresponding to agricultural land parcels within the region. A parameter sweep and sensitivity analysis on model hyperparameters was conducted to explore the effects of changes to agent calibration and training on agent decision-making and model performance.

The second model expands upon the scope of the first, including forester agents and commercial and residential urban agents within a larger region of the Lake Champlain Basin of Vermont. Additionally, the impacts of agent decision-making take place on the simulated landscape, resulting in gradual land cover change over time. Land cover data from the United States Geological Survey's National Land Cover Database was used for initial parameterization, calibration, and training of the model (years 2001, 2006) and model testing (year 2011).

Results suggest that with appropriate scoping and hyperparameter selection, the integration of deep reinforcement machine learning techniques into the development of agent-based models can increase predictive accuracy in the modeling of real-world phenomena; however, these gains must be weighed against the increased technical complexity of such a model and the associated risk of introducing model error.

Table of Contents

List of Figures

List of Tables

Chapter 1

Review of Related Work

TODO

- Additional to work into this section: Sutton and Barto (06); GRLA; Deep SARSA and Q (XU, CAO, CHEN et al, 2018); LOT of ABM Stuff
- Make more readable and less jumpy

The use of agent-based models to study the behavior of human agents in complex systems primarily dates back to the early 1970s, with some of the first formal models being Schelling's dynamic model of segregation [**?**], Reynolds' distributed herding model [**?**], and Axelrod and Hamilton's model for the iterative prisoner's dilemma [**?**]. While attempts to rationalize and describe human behavior date to antiquity, these models were among the first to demonstrate how reducing a complex system down to its elementary components and the simple rules that define it allows for its dynamic behaviors to be reliably, and repeatedly, observed and studied.

The study of emergent systematic behavior and large-scale system dynamics, as described in Anderson's *More is Different* [**?**], would quickly become known as complex systems studies. Over the following decades, interest in the field grew, and the modeling of multi-agent systems became more widespread, resulting in the development of larger, more complex agent-based models.

While early models primarily focused on studying small homogeneous systems, as work continued through the late 1990s and into the new millennium, researchers began to model the behavior of more heterogeneous agent populations [**?**] and explore how agents behave when given cooperative, competitive, or organizational tasks [**?**]. Work from this period began to focus less on solipsistic agents with information only about their independent state and more on how information sharing and networking can affect agent behavior [**?**].

The number of ways to define the behavior of agents within complex agent-based systems is myriad; however, some of the most common include probabilistic methods and rule-based approaches. For the majority of this project, the behavior of agents is going to be defined by artificial neural networks trained using deep reinforcement machine learning. Agent-based systems have previously incorporated reinforcement learning methods like SARSA and temporal difference learning (—, 1990); however, this project is one of the first to embed this type of neural network into agents within such a large-scale and heterogeneous model.

This specific application of reinforcement machine learning may be new, but its study is almost as old as the field of modern computer science. One of the first recorded mentions of reinforcement learning techniques for the development of artificial intelligence is in Turing's *Computing Machinery and Intelligence* [**?**], wherein he proposes that one possible way to construct an intelligent machine is to create a "child machine," that, through the application of punishments and rewards, is taught to behave such that “events which shortly preceded the occurrence of a punishment signal are unlikely to be repeated, whereas a reward signal [increases] the probability of repetition of the events which led up to it.” Computational learning of this sort was studied more seriously over the following decade, eventually being dubbed ’reinforcement learning’ in Minsky’s Steps Towards Artificial Intelligence [**?**]. While many of the techniques of this era have been supplanted by newer methodologies, some of its key theoretical concepts became mainstays and went on to form the backbone of modern reinforcement learning— perhaps most notably the development of temporal difference learning as described in Samuel’s *Some Studies in Machine Learning Using the Game of Checkers* [**?**].

Progress in the study of reinforcement machine learning saw little development over the following decade; however, a resurgence of interest in artificial intelligence during the 1970s revitalized the field and resulted in many new algorithms. Some of the more influential of these algorithms being the temporal difference learning algorithm (Sutton, 1988), the q-learning algorithm (Watkins, 1989), and the related SARSA algorithm (Rummery & Niranjan, 1994) for decision-policy making.

Notably, Sutton’s temporal difference learning algorithm category $TD(\lambda)$, where the historical discounting factor $0 \leq \lambda \leq 1$, is the basis for many of the techniques used in this project. Dayan (1992) proved that Sutton’s temporal difference learning algorithm family converges for discrete problem spaces [**?**]; however, the problem remains undecidable for continuous-valued problems, so consideration must be taken for model hyperparameter selection.

Alongside these developments in reinforcement learning, advancements in computing machinery and the production and training of artificial neural networks helped bypass many of the previous limiting factors in the study of artificial intelligence. For example, the best method to correct neural network output had been an open question since their first use. But, the development of algorithms for the backpropagation of network error revolutionized the field (Rumelhart, Hinton, & Williams, 1986). These methods allowed for the creation of networks that were more intricate and generalizable than ever before, and their increased performance made them a standard with derivatives still used today.

Entering the mid-to-late 1990s, development in artificial intelligence and reinforcement machine learning again began to stall. Problems like vanishing and exploding gradients within the hidden layers of networks, as well as physical limitations on the size and speed of machine memory, made the use and application of deep, large-scale neural networks infeasible for many potential use cases. Progress in the field remained incremental until the mid-2010s when advancements in GPU-enabled computing allowed for faster, more powerful, and more affordable high-performance computing to enter the mainstream. (—)

With this improvement in computing capabilities came several new reinforcement learning methods, including deep reinforcement learning, which makes use of the ability of artificial neural networks to perform function approximation to make the decision-policy for a problem space. By using deep neural networks in this way, the decision-policy table q-learning algorithms use to value decision-making in discrete problem spaces can be replaced with a neural network with a deep q-network architecture for decision-policy making in more continuous problem spaces (DQN).

DQN is a suitable algorithm for many reinforcement learning tasks, but it's not without its flaws. Overcoming its propensity towards biasing itself from outlier data early in training can be incredibly difficult. (Fujimoto, Hoof, & Meer, 2018)

To combat some of the difficulties that can arise from using DQN, several additions and variations to the algorithm have been developed. The addition of policy gradient (O'Donoghue, Munos, Kavukcuglu, & Mnih, 2017) and action replay (Zhao, Wang, Shao, & Zhu, 2016) to the algorithm can help to smooth the learning curve and encourage additional exploration of the problem space. Additionally, combination algorithms like double deep q-learning (DDQN) [**?**] and the rainbow algorithm [**?**] have been showing promising results; however, they are still fairly young algorithms and haven't been around long enough to do a proper meta-analysis of their reliability and accuracy across problem types.

Chapter 2

Machine Learning in Multi-Agent Systems

One of the outstanding challenges in ABMs of coupled natural and human systems concerns the lack of ABMs to simulate agents with the ability to learn from their past failures or successes and environmental and social uncertainties. [?]

In this chapter, a method of integrating machine learning into ABMs is resented as a potential solution to this problem using a modeling methodology incorporating elements of dee reinforcement machine learning with classical ABM techniques. This methodology is then applied to a simple ABM of a coupled natural and human system. The results of this applicaption are then discussed.

2.1 Methodology

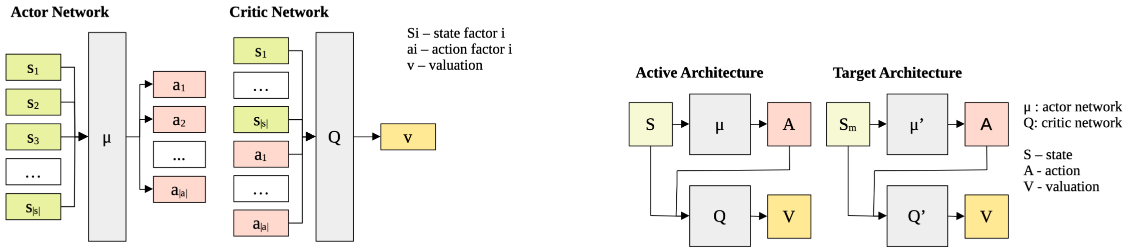
2.1.1 Modeling Approach

The dee reinforcement machine learning methods being used in this model are based on the dee q-learning methods developed by Hasselt, Guez, and Silver [?], incorporating some of the alterations to action relay and learning convergence as described in the rainbow algorithm developed by Hesel et al. [?] and integrating the episodic training structure into the runtime execution of an agent-based model.

In this regard, each agent has two aired actor-critic neural network architectures — one air, which is used for 'active' learning, and a 'target' air which is used for passive learning. The active air is used to drive agent decision-making within the current simulated model environment in any given time-step. The target air is updated periodically with the weights of the active network. This transfer learning is done to prevent the network from overfitting to circumstance and to prevent the networks from diverging during training.

Within each air, the actor network (μ) is responsible for selecting the next action to take given the current state of the agent and the critic network (Q) is responsible for estimating the value of the current state given the current state and action. The actor network is trained to maximize the value of the critic network, whereas the critic network is trained to minimize the difference between its estimated valuation for each state-action air with the valuation that would be consistent with the rewards received from past events.

High-level diagrams of these architectures and how they interact with agent states,



(a) Actor-Critic Pair

(b) Active-Target Pair

Figure 2.1: Diagram of (a) the actor-critic network layout and (b) the active-target transfer learning air used by agents

$s = \left(s_1, ..., s_{|s|}\right)$, and actions, $a = \left(a_1, ..., a_{|a|}\right)$, as vectorized components to produce value estimations can be seen in Figure 2.1.

2.1.2 Agent Decision-Making

Agents in this model make decisions according to an internal decision-policy function ß$(s) = a$ mapping the state of each agent to the potential actions that each agent can take. In this approach, the decision-policy function is being approximated by an artificial neural network (ANN), $\mu : S \to A$. The input to this ANN is the state of the agent, vectorized as a 1-dimensional array of length W_S. The output of the ANN is a vector of length W_A encoding the action that the ANN has decided the agent should take.

Each time an agent needs to take an action, it will ass its current state through the network to generate an action. It will perform this action with some probability $1 - \epsilon$. With probability ϵ, the agent will instead take a random action. This random action is used to encourage exploration of the state space and to prevent the agent from getting stuck in a local optimum.

2.1.3 Agent Memory

Agents in the model store a history of their past experience as a series of state transition records (s_t, a_t, r_t, s_{t+1}). These records are stored in a memory buffer B of fixed length N. When the memory buffer is full, new records overwrite the oldest records in the buffer. The memory buffer is used to train the agent's decision-policy and valuation.

Additionally, agents have a built-in 'forgetfulness factor', F, which has been incorporated in an attempt to capture some of the behavior patterns of human actors with imperfect memory. This factor is a real number between 0 and 1 and is used to linearly scale the amount of noise that is introduced into the memory record as the record ages within a run. An agent with $F = 0$ will have perfect recall of its entire state transition history, whereas an agent with $F = 1$ will have perfect recall of its most recent state transition with actions taken in the distant past being completely forgotten (noise term of equal range as actual term).

2.1.4 Agent Learning

2.2 Experimental Design

2.2.1 Simulation Environment

In order to test this modeling methodology, an experimental agent-based model was developed to explore the behavior of a multi-agent system of agricultural decision-makers and how that behavior may change in response to various external stimuli.

The real-world basis for this model is a study area in the Missisquoi Bay Area of the Lake Champlain Basin of Vermont, and the model is designed to represent the agricultural decision-making processes of farmers in this area. In articular, decisions pertaining to land use practices and the adoption or rejection of agricultural best management practices (BMPs) were studied.

The model was implemented using the FLAMEGPU framework (FLAMEGPU), which is an agent-based modeling framework that allows for the development of models that can be run on a GPU-device, and with CUDNN (CUDNN), a dee learning library for CUDA. These technologies were selected **EXPAND RATIONALE**

2.2.2 Agents

There are two types of agents resent in this model — 480 farmer agents, corresponding to the 480 agriculturally-zoned land parcels in the Missisquoi Bay Area, and a single regulatory agent. All agents in the model contain some internal information about their current state and history, a set of state-transition memories used to learn from experience, and a air of neural networks used to drive agent decision-making. As the agents make decisions over time, they gradually learn the correlation between the actions they take from each state using dee reinforcement machine learning.

The 480 farmer agents are used to model the behavior of agricultural land managers within the study area. These agents make annual decisions, once er time-step, about their farming practices, including whether they should adjust their productivity in one of four agricultural sectors (beef, dairy, corn, and hay) and whether they should implement an agricultural best management practice (BMP) to reduce phosphorus on their land.

Conditions that factor in as comonents of a farmer agent's state include the total land area the agent has devoted to cropland or pasture; the productivity of the agent in each of the four modeled agricultural industries along with their associated phosphorus byproduct productivity; an n-year history of the farm's profitability, storm losses, and BMP usage; and similar historical information from the agent's k-nearest neighboring farmer agents.

The one municipal regulatory agent is used to model a municipal government or regulatory agency's behavior managing agricultural practices on the landscape and the local environment and the policies that guide them. This agent acts more slowly than the agricultural agents, once every five time-steps, and decides if/how it should modify its incentive structure — changing its taxation rate, the subsidization given to an agent adopting a BMP, and the phosphorus runoff threshold at which a penalty is applied.

The municipal regulatory agent's state conditions include a history of extreme weather events in the region; and the aggregate profitability, phosphorus runoff, and storm loss across the last five time-steps for all agents.

Problem Space

The problem space for this model is the set of all possible states that the agents can be in and the set of all actions that can be taken from those states, detailed here.

The state factors used in the decision-making of each agent are listed in Table 2.1. For the farmer agents, these break down into a few main groups: information about their own land cover, information about their productivity in the given time-step, a 5-year history of their own experiences, and historical information from their 5-nearest

Table 2.1: A summary of the state factors being used as input to the agents' ANNs. Agents may have additional properties (listed in Appendix A); however, these are the ones that action selection is directly dependant on.

Farmer Agent

Group	Description	Detail
Land Cover	Cropland	Normalized Area (sq m)
	Pasture	Normalized Area (sq m)
Productivity	Corn	See A
	Hay	
	Dairy	
	Beef	
History (5-year)	Extreme Event Record	Occurrence
	Financial Record	A
	BMP Adoption Record	
Network Information	Financials	Losses (1-year, 5-year)
	BMP Adoption	Usage (1-year, 5-year)

Regulator Agent

Group	Description	
Aggregate Data	BMP Adoption	
	Financials	Net Profits, Losses
	P Runoff	
History	Extreme Event Record	5-year, 15-year
	BMP Adoption	
	Financials	Net Profits, Losses
	P Runoff	

neighbors.

The actions that the farmer agents can take are listed in Table 2.2. These actions are divided into two groups: adjusting the productivity of the agent in a given agricultural sector, and adopting a BMP to reduce phosphorus runoff from the agent's land.

The actions that the regulator agent can take are listed in Table **??**. These actions are divided into three groups: adjusting the tax rate applied to the farmer agents, adjusting the subsidy given to farmer agents adopting a BMP, and adjusting the phosphorus runoff threshold at which a penalty is applied.

The reward value for farmer agents is a function of their net profitability in the given year, after accounting for any storm losses and BMP adoption costs. The reward value for the regulatory agent is to minimize the $R_r = \left\langle G_* \sum_f P, G_l * \sum_f loss \right\rangle$

2.2.3 Hyperparameter Selection

A summary of model hyperparameters is listed in Table 2.3.

Preliminary model runs were conducted to determine the optimal values for the hyperparameters for machine learning within the model. The learning hyperparameters that were varied in these preliminary runs were the number of training episodes, the number of steps between target network updates, the number of inner layers in the neural networks, the number of neurons in each of those inner layer, the learning rate, and the batch size. The learning hyperparameters that were held constant were the exploration rate at $\varepsilon = 0.1$, the discount factor at $\gamma = 0.99$, the learning transfer rate at $\tau = 0.001$, the number of steps within a training episode at $N = 40$, the relay memory size at $M = 10000$.

A summary of the final model hyerparameters are listed in Table 2.3. Network-specific parameters are listed in Table 2.4.

2.2.4 Experimental Setup

The model was run for a variety of scenarios. All scenarios tested the variables BMP_e, ΔEE, and g. Then, there were two classes of test: tests with agents with uniform

Table 2.2: A summary of the action factors being used to drive agent decision-making for both types of agent resent in the model. Each group is an n-hot encoding set containing the subsequent possible actions. A A

Farmer Agent

Group	Action	Reference Index
BMP Usage	Adopt BMP	$(0, 0)$
	Don't Adopt BMP	$(0, 1)$
Corn Production	Increase by $[0, S_c^+)$	$(1, 0)$
	Maintain	$(1, 1)$
	Decrease by $[0, S_c^-]$	$(1, 2)$
Hay Production	Increase by $[0, S_h^+]$	$(2, 0)$
	Maintain	$(2, 1)$
	Decrease by $[0, S_h^-)$	$(2, 2)$
Dairy Production	Increase by $[0, S_d^+)$	$(3, 0)$
	Maintain	$(3, 1)$
	Decrease by $[0, S_d^-)$	$(3, 2)$
Beef Production	Increase by $[0, S_b^+)$	$(4, 0)$
	Maintain	$(4, 1)$
	Decrease $[0, S_b^-)$	$(4, 2)$

Regulator Agent

Group	Action	Reference Index
Tax Rate	Increase by $[0, T_g^+)$	$(0, 0)$
	Decrease by $[0, T_g^-)$	$(0, 1)$
BMP Subsidy	Provide/Increase	$(1, 0)$
	Remove/Decrease	$(1, 1)$
Phosphorous Threshold$\star$	Scale	2

Table 2.3: Hyperparameters and associated values with source or rationale for the agricultural land use model

Parameter	Value	Source/Rationale
Learning Rate (α)	0.00025	
Exploration Rate (ε)	0.1	
Discount Factor (γ)	0.99	
Transfer Rate (τ)	0.001	cite *Transfer learning*
Relay Memory Size (M)	10000	
Batch Size (B)	32	[?]
Number of Episodes (N)	1000	Testing **Expand**
Number of Steps er Episode (T)	40	Econ model limitations **Expand**

Table 2.4: Network parameters for the ANNs used by the agents in the agricultural land use model, where the μ_a and Q_a columns correspond to the values for the agricultural agents' networks and μ_r and Q_r correspond to the values for the regulatory agent networks ***Include bias line?***

Parameter	μ_a	Q_a	μ_r	Q_r
Input Nodes	15	32	12	22
Inner Layers	5	5	5	5
Inner Nodes	10	16	10	16
Activation Function	ReLU	ReLU	ReLU	ReLU
Connectivity	Full	Full	Full	Full
Output Nodes	17	1	5	1
Output Activation	n-hot	Linear	n-hot	Linear
Output Groups (n)	5	—	3	—

memory accuracy (Table 2.5) and tests with agents with heterogeneous memory accuracy (Table 2.6).

BMP Efficacy (BMP_e) was varied from 0.0 to 1.0 in increments of 0.1. This parameter represents the effectiveness of BMPs in reducing nutrient loading from agricultural fields. A value of 0.0 indicates that BMPs have no effect on nutrient loading, while a value of 1.0 indicates that BMPs completely eliminate nutrient loading. This parameter was varied in order to determine the effect of BMP efficacy on the behavior of the system.

Change in weather event frequency (ΔEE) was varied from -0.2 to 0.2 in increments of 0.05. This parameter represents the change in the frequency of extreme weather events, such as heavy rainfall, that may be induced by climate change compared to a historical baseline.

Change verbiage from threshold to scale? The regulation change threshold (g) represents the maximum rate at which the regulatory agent will adjust the regulatory environment. Three values were tested: an aggressive threshold ($g = 0.2$), a moderate threshold ($g = 0.05$), and a restrictive case ($g = 0$) for testing the model's ability to operate in a static regulatory environment.

Change wording: F' for accuracy and F for forgetfulness is bad. The impact of agent memory accuracy was tested for two types of agent populations. In uniform agent populations, all agents had the same memory recall accuracy (F'), where F' is the probability that a memory will be recalled correctly. In heterogeneous agent populations, agents had different memory recall accuracies, where a proportion of agents (P) had accuracy $F = 1$ and all other agents ($1 - P$) had accuracy $F = 0$.

Table 2.5: Table listing experimental parameters for uniform population runs

Variable	Values
BMP Efficacy (BMP_e)	0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0
Change in Event Frequency (ΔEE)	-0.2, -0.15, -0.1, -0.05, 0.0, 0.05, 0.1, 0.15, 0.2
Regulation Change Threshold (g)	0, 0.05, 0.2
Recall Accuracy (F')	0, 0.25, 0.5, 0.75, 1

Table 2.6: Table listing experimental parameters for mixed population runs

Variable	Values
BMP Efficacy (BMP_e)	0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0
Weather Event Frequency (ΔEE)	-0.2, -0.15, -0.1, -0.05, 0.0, 0.05, 0.1, 0.15, 0.2
Regulation Change Threshold (g)	0, 0.05, 0.2
Population Mixing (P)	0.25, 0.5, 0.75

2.3 Results

2.3.1 Model Performance

For each model parameterization, agents were trained for 1000 training episodes. If more than 10% of the agents ($n = 48$) in the model failed to converge to a stable policy within 1000 training episodes, the model was discarded and retrained; however, this occurred in less than 2% of model runs. A lot showing the distribution of number of agents which converged across model parameterizations is shown in Figure 2.2.

In this training, an agent's networks were considered to have converged if after 50 initial training episodes, the net change in the weights of the network during a transfer learning step was less than 10^{-5}. This threshold was chosen to be small enough to ensure that the networks had reached some stable policy, but large enough to avoid overfitting.

Models which were successfully trained and passed through this screening were

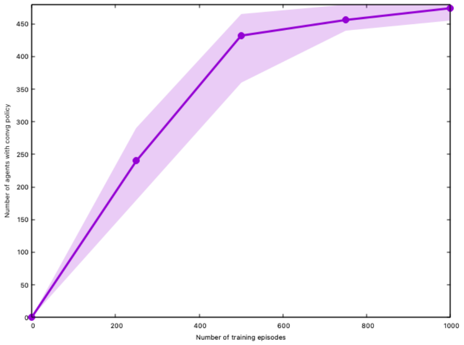
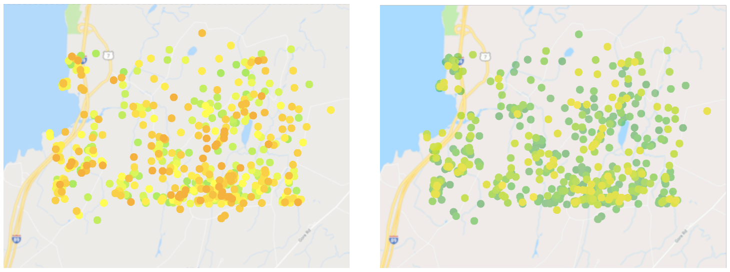


Figure 2.2: Plot of the number of agents that converged to a stable policy for each parameterization of the model



(a) Initial State (y=2001) *(b) End State (y=2040, g=0.05)*

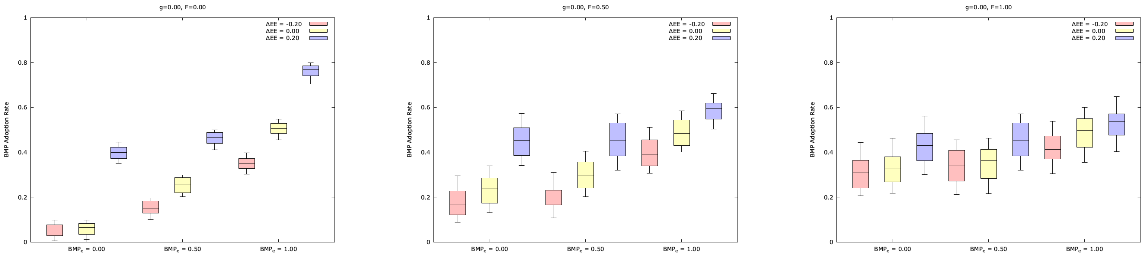
Figure 2.3: Sample model output showing the change in BMP adoption likelihood from a characteristic initial model state (a) to a characteristic end state (b) for a model run with a moderate regulatory change threshold ($g = 0.05$). The color of each dot represents the likelihood that the agent will adopt BMPs, with green indicating a high likelihood and red indicating a low likelihood.

then run for 40 testing runs.

2.3.2 Agent Behavior

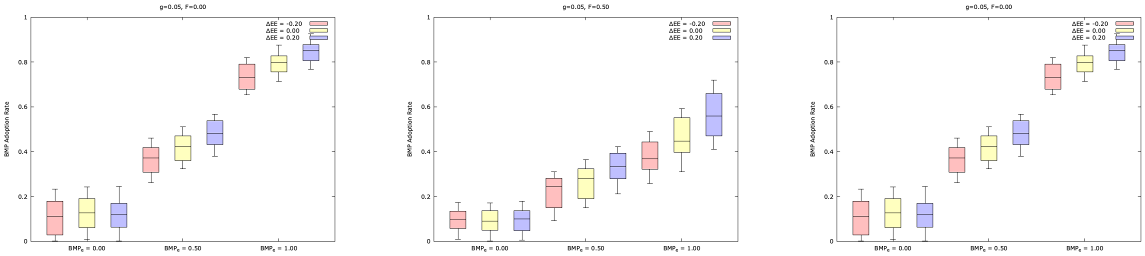
Uniform Population Runs

For model parameterizations with uniform agent populations, the proportion of agents which adopted a BMP in each testing model run was recorded and used to generate a distribution of BMP adoption rates for each parameterization. Summaries of the



(a) $F = 0$ *(b)* $F = 0.5$ *(c)* $F = 1.0$

Figure 2.4: Distribution of mean BMP adoption rate for uniform population runs of the agricultural land use model, where $g = 0.0$*, for (a)* $F = 0$*, (b)* $F = 0.5$*, and (c)* $F = 1.0$



(a) $F = 0$ *(b)* $F = 0.5$ *(c)* $F = 1.0$

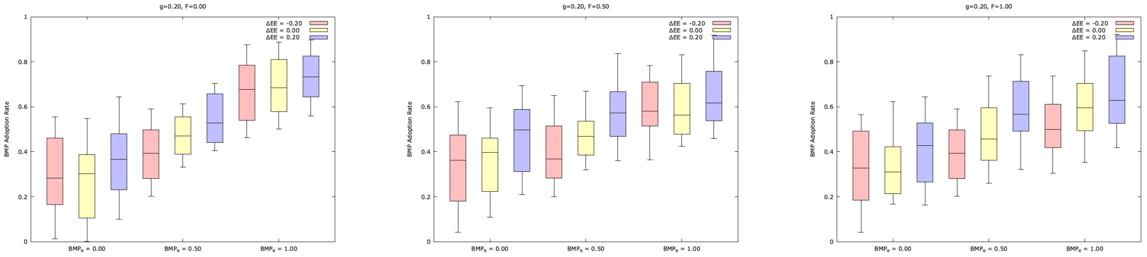
Figure 2.5: Distribution of mean BMP adoption rate for uniform population runs of the agricultural land use model, where $g = 0.05$*, for (a)* $F = 0$*, (b)* $F = 0.5$*, and (c)* $F = 1.0$

results of these runs are shown in Figure 2.4 for the case where the regulation change threshold (g) was set to 0.0, Figure 2.5 for when g was set to 0.05, and Figure 2.6 for when g was set to 0.2.

Some parameterizations have been omitted for readability. The full listing of results can be found in Appendix **??**.

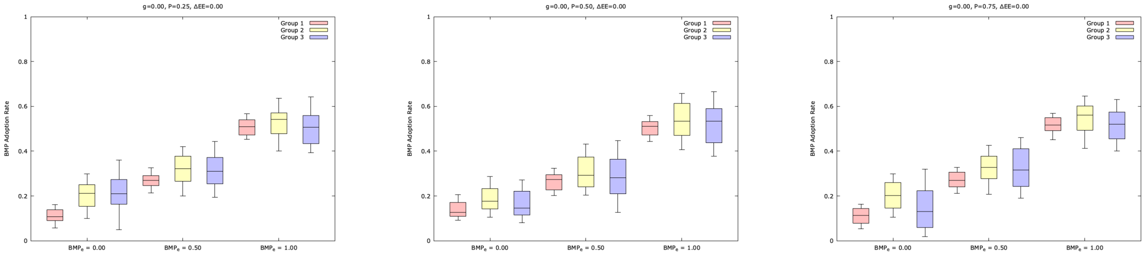
Mixed Population Runs

For model parameterizations with mixed agent populations, agents were divided into three groups: group 1, where $F = 0$ for the agent and all neighbors, group 2, where $F = 1$ for the agent and all neighbors, and group 3, for agents with neighbors where



(a) $F = 0$ *(b)* $F = 0.5$ *(c)* $F = 1.0$

Figure 2.6: Distribution of mean BMP adoption rate for uniform population runs of the agricultural land use model, where $g = 0.2$*, for (a)* $F = 0$*, (b)* $F = 0.5$*, and (c)* $F = 1.0$



(a) $P = 0.25$ *(b)* $P = 0.5$ *(c)* $P = 0.75$

Figure 2.7: Distribution of mean BMP adoption rate for mixed population runs of the agricultural land use model, where $g = 0.0$*,* $\Delta EE = 0$*, for (a)* $P = 0.25$*, (b)* $P = 0.5$*, and (c)* $P = 0.75$

$F = 0$ and $F = 1$. The proportion of agents in each group which adopted a BMP in each testing model run was recorded and used to generate a distribution of BMP adoption rates for each parameterization. Results of one set of parameterizations of these runs are shown in Figure 2.7 where $g = 0$, $\Delta EE = 0$. The results indicate generally that this method of introducing heterogeneity into the population can introduce variance in agent behavior, but that it is unclear if it leads to any of the desired emergent behavioral patterns. Further testing, specifically targeting these types of populations would be needed to draw stronger conclusions.

A full table listing the results for all parameterizations can be found in Table **??**.

2.4 Discussion

2.4.1 Sensitivity and Limitations

The purpose of the regulatory agent was to help incentivize agent learning, but in results with $g = 0.2$, the increased variability in model performance was high and the impact dominated all other parameters. **Detail, Scaling in results, mean and variance, why these results even matter**

Chapter 3

Increasing ABM Integration

The previous chapter demonstrated how an agent-based model can be implemented with machine learning in order to induce a model of individual behavior. This chapter will go on to show how this kind of agent-based model can be integrated with a land cover model in order to predict land cover change.

3.1 Methodology

The methodology here expands on the methodology presented in Chapter **??**. This model is also an agent-based model, but instead of focusing primarily on training the decision-policy of agents in the model, the focus is on the land cover change that results as a byproduct of the behavior of the agents.

3.1.1 Modeling Land Cover Change

Within the model, land cover is represented as a grid of cells. Each cell has a land cover category, an NLCD land cover class, a land use, and an associated land parcel.

Table 3.1: Land Cell Features

Parameter	Values
Land Cover Category	Agricultural, Forested, Urban, Other
NLCD Cover Class	
Land Usage Type	Managed/In-Use, Adjacent, Unmanaged

Table 3.2: Land Cover Categories

Cover Category	NLCD Cover Class	Encoding
Urban	Open Space Urban	21
	Low Density Urban	22
	Medium Density Urban	23
	High Density Urban	24
Forested	Deciduous Forest	41
	Evergreen Forest	42
	Mixed Forest	43
Agricultural	Pasture	81
	Crops	82
Barren	Barren	31
Grassland/Scrub	Scrub	52
	Grassland	71
Other	Water	11
	Wetlands Woody	90
	Wetlands Other	95

See Table 3.1.

Land cover change is modeled as a stochastic byproduct of agent decision-making. For example, an agricultural agent deciding to increase its productivity with regard to grazing animals may result in clearing forested land cells and transitioning them to pasture.

These land-cover transitions

Agent behavior is trained according to internal incentive structures, the model and was trained on the validation accuracy of transitions between land cover types in the validation dataset.

Details

3.2 Experimental Design

3.2.1 The Model

Real world land cover datasets for years: 2006, 2011, 2016. Real world land cover transitions between 2006 and 2011 were used as training and validation data for the land cover model. Real world land cover transitions between 2011 and 2016 were used as test data for the land cover model.

The real world transitions mapped from cover A to cover B, which is referred to as $\Delta_{A,B}$. The modeled transitions for a parameterization $M(...)$ from cover A to cover $\hat{B}$ is referred to as $\Delta_{A,\hat{B}}M(...)$.

Explain meaning

Agents

There are four types of agent present in this model: agricultural agents, forestry agents, commercial agents, and residential agents.

Agricultural agents model the behavior of farmers, herders, and other kinds of agricultural land managers within the study area. They make annual decisions about their farming practices, including whether they should change production in one of the four modeled agricultural industries (beef, dairy, corn, and hay) and whether they should implement an agricultural best management practice (BMP) to reduce phosphorous runoff on their land. This agent type is very similar to how it was implemented in the previous model; although, now, its productivity decisions can change the land cover of the model.

Forester agents model the behavior of loggers and other kinds of forested land managers within the study area. They make annual decisions about their practices and whether to implement an advised management practice (AMP).

Commercial agents model the behavior of shops, factories, offices, and other kinds of commercial land-holders within each study area. They make decisions trimonthly about their workforce, including their available jobs and the associated salaries Byproducts of their actions impact the density and sprawl of urban land cover on the landscape.

Residential agents model the behavior of renters and landowners within each study area. They make two decisions annually: whether to attempt a job change and whether to try to move house. Household satisfaction is valued as a combination of financial stability and mental satisfaction. Each household earns wages provided by a commercial agent — these wages are determined by a stochastic process and can be adjusted by the job over time. The decisions of these agents do not directly impact land cover change on their associated parcel, but land cover can transition within their parcel as a result of the decisions of other agents.

Agricultural and forestry agents are connected to and share information with their n-nearest neighbors of the same agent type; these networks are static throughout each model run. Commercial and residential agents exist in a bipartite network with one another. This network is initialized via a stochastic process and is updated as agents make decisions.

The learning architecture for agents of each type is listed in Table 3.3.

Table 3.3: Network parameters for the ANNs used by agents in each class for the land cover model

Parameter	Agricultural		Forestry		Commercial		Residential	
	μ	Q	μ	Q	μ	Q	μ	Q
Input Nodes	15	32	10	15	4	10	5	9
Inner Layers	4	3	4	3	2	2	2	2
Inner Nodes	10	16	7	7	5	5	4	5
Output Nodes	17	1	5	1	6	1	4	1

Execution Overview

Add Textual description of model execution and the timing of model subsystems. Agents make decisions, then act and interact, then learn.

3.2.2 Hyperparameter Selection

A summary of model hyperparameters is listed in Table **??**.

3.2.3 Experimental Setup

Parameters were varied for experimental scenarios.

Explain why this is more of a hyperparameter analysis. Because we want to explore how agent interactivity and agent decision-making can have secondary effects.

Include other parameters, or simple version?

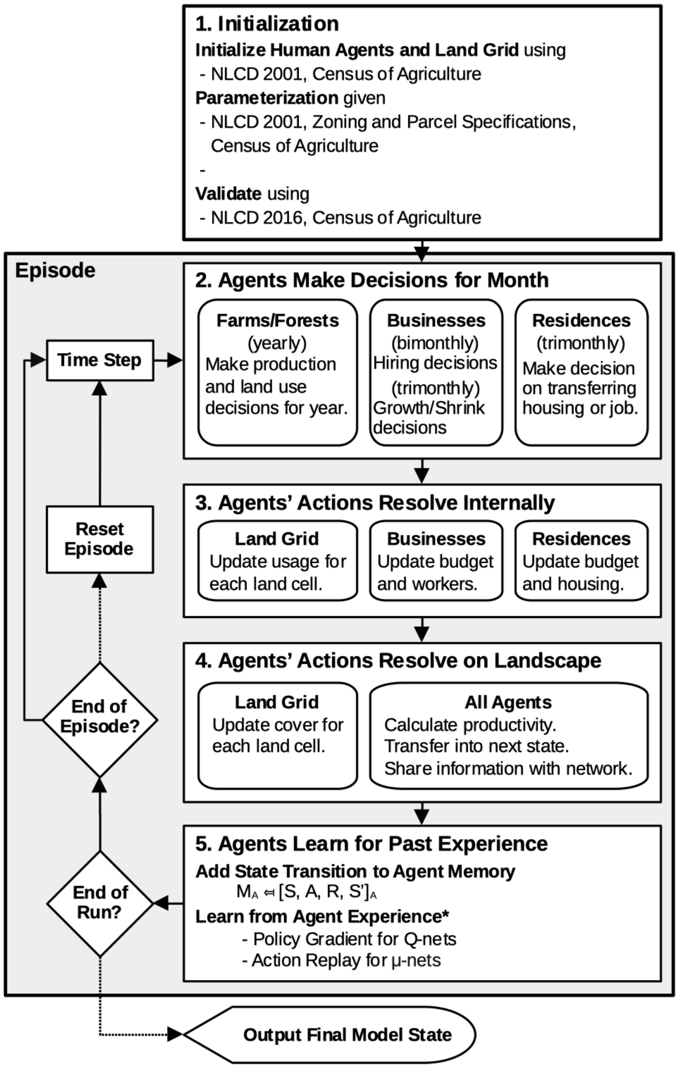

1. Initialization
Initialize Human Agents and Land Grid using
- NLCD 2001, Census of Agriculture
Parameterization given
- NLCD 2001, Zoning and Parcel Specifications, Census of Agriculture
-
Validate using
- NLCD 2016, Census of Agriculture
Episode
2. Agents Make Decisions for Month
Farms/Forests
(yearly)
Make production and land use decisions for year.
Businesses
(bimonthly)
Hiring decisions
(trimonthly)
Growth/Shrink decisions
Residences
(trimonthly)
Make decision on transferring housing or job.
Time Step
3. Agents' Actions Resolve Internally
Land Grid
Update usage for each land cell.
Businesses
Update budget and workers.
Residences
Update budget and housing.
Reset Episode
4. Agents' Actions Resolve on Landscape
Land Grid
Update cover for each land cell.
All Agents
Calculate productivity.
Transfer into next state.
Share information with network.
End of Episode?
5. Agents Learn for Past Experience
Add State Transition to Agent Memory
M_A ⇐ [S, A, R, S']_A
Learn from Agent Experience*
- Policy Gradient for Q-nets
- Action Replay for μ-nets
End of Run?
Output Final Model State

Figure 3.1: Flowchart demonstrating the overall execution of the agent-based model and its coupling with the machine learning process

Table 3.4: Experimental parameters for the land cover transition model

Variable	Values
Batch Size (B)	8, 16, 32, 64
Discount Factor (γ)	0.5, 0.9, 0.99
Recall Accuracy (F')	0.0, 0.25, 0.5, 0.75

3.2.4 Known Limitations

Some types of land-cover transition either do not exist in the training data or are not well-represented. These types of transitions are rare within the study area,

move and elaborate

3.3 Results

3.3.1 Model Performance

Model performance under each experimental parameterization was evaluated by comparing the land cover in model year 2016 to the recorded/observed land cover for the study area for real year 2016.

The Nash-Sutcliffe efficiency index (NSE) was used to evaluate the goodness-of-fit of the model under each experimental parameterization. The NSE is a measure of the relative magnitude of the residual variance of modeled data compared to the residual variance of the observed data. The value of the index ranges from $-\inf$ to 1, where a score of 1 indicates a perfect fit, a score of 0 indicates that the model is no better than the mean of the observed data, and a score less than 0 indicates that the mean of the observed data is a better predictor than the model.

This index was calculated in relation to three forms of model performance. The first is the ability of the model to appropriately predict the proportional coverage of each land cover type in the target year, $\mathrm{NSE}_{\mathrm{plc}}$. The second is the ability of the model to appropriately predict the categorical transitions of land cover types from the start year to the target year, $\mathrm{NSE}_{\mathrm{cat}}$. The third is the ability of the model to appropriately predict the absolute transitions of land cover types from the start year to the target year, NSE$_{\text{abs}}$. These indices are detailed below.

A majority of land cells in the study area do not transition land cover between the start year and target year ($92.6\%, n = 69124$), which would heavily bias any analysis of model performance. Therefore, the NSE was calculated only for those cells that transitioned land cover.

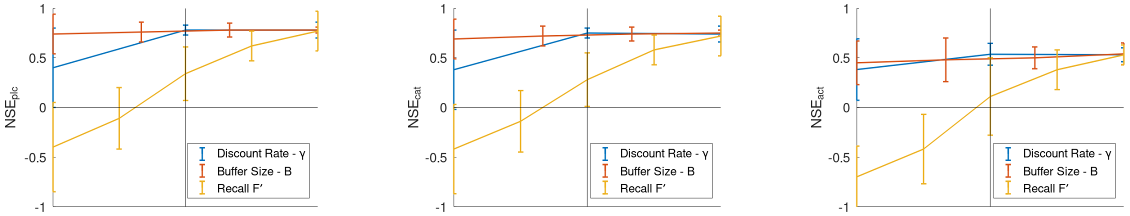
The NSE measure of proportional coverage (NSE$_{\text{plc}}$), shown in Equation 3.1, where P_b represents the observed proportional coverage of land cover type b in the target year, $\hat{P}_b$ represents the simulated proportional coverage of land cover type b in the target year, and $\bar{P}_b$ represents the mean observed proportional coverage of land cover type b in the target year.

$$\text{NSE}_{\text{plc}} = \frac{\sum_b \left(P_b - \hat{P}_b\right)^2}{\sum_b \left(P_b - \bar{P}_b\right)^2} \tag{3.1}$$

The NSE measure of categorical land cover transitions (NSE$_{\text{cat}}$), shown in Equation 3.2, where $\Delta_{A,B}$ represents the number of observed transitions from land cover category A in the starting year to land cover category B in the target year, where $\widehat{\Delta_{A,B}}$ represents the number of simulated transitions from A to B, and where $\overline{\Delta_{A,B}}$ represents the mean observed number of transitions from A to B. **Not sure I'm saying this correctly.**

$$\text{NSE}_{\text{cat}} = \frac{\sum_{A,B} \left(\Delta_{A,B} - \widehat{\Delta_{A,B}}\right)^2}{\sum_{A,B} \left(\Delta_{A,B} - \overline{\Delta_{A,B}}\right)^2}, A \neq B \tag{3.2}$$

The NSE measure for absolute land cover transition (NSE$_{\text{act}}$), shown in Equation 3.3, is very similar to the calculation of NSE$_{\text{cat}}$, except that it is calculated



(a) NSE_{plc} *(b) NSE_{cat}* *(c) NSE_{act}*

Figure 3.2: NSE index sensity for each index showing the variance in model classification accuracy by each metric under different model parameterizations.

for the absolute land cover class of each land cell and not just its categorical class.

NLCD Cover Class vs Coverage Category, f.x. for-deciduous vs forested

$$\mathrm{NSE}_{\mathrm{abs}} = \frac{\sum_{a,b}\left(\Delta_{a,b} - \widehat{\Delta_{a,b}}\right)^2}{\sum_{a,b}\left(\Delta_{a,b} - \overline{\Delta_{a,b}}\right)^2}, a \neq b \tag{3.3}$$

The maximum NSE values seen during testing runs were $\mathrm{NSE}_{\mathrm{plc}} = 0.84$, $\mathrm{NSE}_{\mathrm{cat}} = 0.76$, and $\mathrm{NSE}_{\mathrm{act}} = 0.64$.

The sensitivity of NSE to each model parameterization across model runs was evaluated by calculating the mean and variance of the NSE indices under each parameterization. Figure 3.2

3.4 Discussion

Explain why it matters.

Name	Type	Range
Econ	float	-10 10

3.5 Agents

3.5.1 Agents

Listing of agent state parameters.

Bibliography

[1] Thomas C Schelling. Dynamic models of segregation. *Journal of mathematical sociology*, 1(2):143–186, 1971.

[2] Craig W Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 25–34, 1987.

[3] Robert Axelrod and William D Hamilton. The evolution of cooperation. *science*, 211(4489):1390–1396, 1981.

[4] Philip W Anderson. More is different: Broken symmetry and the nature of the hierarchical structure of science. *Science*, 177(4047):393–396, 1972.

[5] E. Bonabeau. Application of simulation to social sciences. *Hermes Sciences*, pages 451–461, 2000.

[6] Robert Axelrod. The complexity of cooperation: Agent-based models of competition and collaboration. *Princeton Univ. Press*, 1997.

[7] Michael J Prietula, Kathleen M Carley, and Les Gasser. *Simulating organizations: Computational models of institutions and groups.* Mit Press, 1998.

[8] Computing Machinery. Computing machinery and intelligence-am turing. *Mind*, 59(236):433, 1950.

[9] Marvin Minsky. Steps toward artificial intelligence. *Proceedings of the IRE*, 49(1):8–30, 1961.

[10] Arthur L Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, 3(3):210–229, 1959.

[11] Peter Dayan. The convergence of td (λ) for general λ. *Machine learning*, 8:341–362, 1992.

[12] Hado V. Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[13] Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[14] Egemen Sert, Yaneer Bar-Yam, and Alfredo J Morales. Segregation dynamics with reinforcement learning and agent based modeling. *Scientific reports*, 10(1):1–12, 2020.

[15] Carla Gomes, Thomas Dietterich, Christopher Barrett, Jon Conrad, Bistra Dilkina, Stefano Ermon, Fei Fang, Andrew Farnsworth, Alan Fern, Xiaoli Fern, et al. Computational sustainability: Computing for a better world and a sustainable future. *Communications of the ACM*, 62(9):56–65, 2019.

[16] Hoo-Chang Shin, Holger R Roth, Mingchen Gao, Le Lu, Ziyue Xu, Isabella Nogues, Jianhua Yao, Daniel Mollura, and Ronald M Summers. Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. *IEEE transactions on medical imaging*, 35(5):1285–1298, 2016.

[17] Lisa Torrey and Jude Shavlik. Transfer learning. *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*, 1:242, 2009.

[18] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. Lstm neural networks for language modeling. In *Thirteenth Annual Conference of the International Speech Communication Association*, 2012.

[19] Gavin A Rummery and Mahesan Niranjan. *On-line Q-learning using connectionist systems*, volume 37. University of Cambridge, Department of Engineering, 1994.

[20] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

[21] Ladislav Rampasek and Anna Goldenberg. Tensorflow: Biology gateway to deep learning? *Cell systems*, 2(1):12–14, 2016.

[22] Asim Zia, Ciara Low, and Danielle Shaw. Hedonic value of water quality in lake champlain basin: A multi-level modeling approach. *Ecological Economics*, 2017.

[23] Asim Zia, Arne Bomblies, Andrew W Schroth, Christopher Koliba, Peter D F Isles, Yushiou Tsai, Ibrahim N Mohammed, Gabriela Bucini, Patrick J Clemins, Scott Turnbull, Morgan Rodgers, Ahmed Hamed, Brian Beckage, Jonathan Winter, Carol Adair, Gillian L Galford, Donna Rizza, and Judith Van Houten. Coupled impacts of climate and land use change across a river-lake continuum: insights from an integrated assessment model of lake champlain's missisquoi basin, 2000–2040. *Environmental Research Letters*, 11(11), 2016.

[24] Bongghi Hong, Karin E Limburge, Jon D Erickson, John M Gowdy, Audra A Nowosielski, John M Polimeni, and Karen M Stainbrook. Connecting the ecological-economic dots in human-dominated watersheds: Models to link socio-economic activities on the landscape to stream ecosystem health. *Landscape and Urban Planning*, 91:78–87, June 2009.

Appendix A
Farm Model ODD+D Document

text