InterCropGym
Intercropping expansion of CropGym environment for simulation with Reinforcement Learning techniques

I. Abstract

Intercropping, the agricultural practice of cultivating multiple crops simultaneously, offers significant potential for sustainable agriculture but has seen declining usage in favor of monoculture systems. This work presents an extension of the CropGym reinforcement learning environment to enable intercropping simulation. Building upon LINTUL3's crop growth engine, we developed a framework that models key interaction mechanisms between two crops, including competition for light, water, and nutrients, as well as facilitative effects.

The environment extends CropGym's OpenAI Gym interface to support simultaneous simulation and interaction of two independent crop systems. We evaluate the system using three reinforcement learning approaches: Double Deep Q-Network (DDQN), Soft Actor-Critic (SAC), and Proximal Policy Optimization (PPO), comparing their effectiveness in optimizing fertilization strategies for intercropped systems.

The work provides a foundation for applying reinforcement learning to complex agricultural systems, though we note that the current interaction models would benefit from agronomic expert validation. This framework enables researchers to explore sustainable farming practices through simulation, contributing to the development of data-driven decision support tools for regenerative agriculture.

II. Enviroment

The proposed Intercrop environment - *InterCropGym* - is based on CropGym[4], a RL environment focused on nitrogen optimization. In particular, it's based on the LINTUL-3[8] crop simulation engine.

A. CropGym

CropGym is an OpenAI environment for optimizing agricultural fertilization through reinforcement learning. The system operates as follows:

1) The environment simulates the growth of *winter wheat* using the LINTUL-3 process-based model, which tracks state variables including the stage of crop development, soil nitrogen levels, and biomass accumulation.
2) An agent performs periodic observations of the system state $s \in S$, comprising crop growth variables and weather data. The action space A consists of discrete fertilizer amounts:

$$A = \{20k\frac{\text{kg}}{\text{ha}} \mid k \in \{0, 1, 2, ..., 6\}\}$$

3) The reward at timestep t is calculated as:

$$r_t = m_{SO,t} - m_{SO,t-1} - (m^*_{SO,t} - m^*_{SO,t-1}) - \beta m_{fert,t}$$

where:

- $m_{SO,t}$ is the storage organ mass at time t
- $m^*_{SO,t}$ is the baseline storage organ mass without fertilization
- $m_{fert,t}$ is the applied fertilizer mass
- β is a penalty coefficient for fertilizer use

The objective is to maximize crop yield while minimizing the environmental impact of excess fertilization. The environment enables the development and evaluation of sustainable fertilization strategies using reinforcement learning.
LINTUL3 is a crop model which simulates the growth of biomass in a water and nitrogen limited environment, keeping track of a state consisting of dynamic parameters I.

TABLE I
State Variables

Name	Description	Unit
ANLV	Actual nitrogen content in leaves	g/m^2
ANRT	Actual nitrogen content in root	g/m^2
ANSO	Actual nitrogen content in storage organs	g/m^2
ANST	Actual nitrogen content in stem	g/m^2
CUMPAR	PAR accumulator	MJ/m^2
LAI	Leaf area index	m^2/m^2
NLOSSL	Total nitrogen loss by leaves	g/m^2
NLOSSR	Total nitrogen loss by roots	g/m^2
NNI	Nitrogen Nutrition Index	–
NUPTT	Total nitrogen uptake	gN/m^2
ROOTD	Rooting depth	m
TAGBM	Total aboveground biomass	g/m^2
TGROWTH	Total biomass growth (above and below ground)	g/m^2
TNSOIL	Amount of inorganic nitrogen available for crop uptake	g/m^2
WDRT	Dead roots	g/m^2
WLVD	Weight of dead leaves	g/m^2
WLVG	Weight of green leaves	g/m^2
WRT	Weight of roots	g/m^2
WSO	Weight of storage organs	g/m^2
WST	Weight of stem	g/m^2

TABLE II

GYMCROP OBSERVATION VARIABLES

Name	Description	Unit
LAI	Leaf area index	m^2/m^2
WLVD	Weight of dead leaves	g/m^2
WLVG	Weight of green leaves	g/m^2
WRT	Weight of roots	g/m^2
WSO	Weight of storage organs	g/m^2
WST	Weight of stems	g/m^2
TAGBM	Total above ground biomass	g/m^2
TGROWTH	Total growth rate	g/m^2
NUPTT	Total nitrogen uptake	gN/m^2
TRAN	Transpiration rate	mm/d
TIRRIG	Total irrigation	mm
TNSOIL	Total nitrogen in soil	gN/m^2
TRAIN	Total rainfall	mm
TRANRF	Transpiration reduction factor	–
TRUNOF	Total runoff	mm
TTRAN	Total transpiration	mm
WC	Water content	m^3/m^3
DVS	Development stage (between 0.0 and 2.0)	–
DATE	Timestamp of observation	–

TABLE III

STATE VARIABLES OBSERVED BY GYMCROP

B. InterCropGym

The new proposed intercropping environment is an evolution of CropGym, which allows the interaction between two different crops, creating a regenerative agricolture setup. First of all, the CropGym environment is modified to create an OpenCropGym version. To do so, the step phase is divided in 3 steps:

1) *Pre-Step*: performs the preliminary step of applying an action and running the simulation, without updating the internal state.
2) *Update-step*: updates the environment's internal state with new input metrics and parameters.
3) *Step*: computes the actual step of the environment, calculating the reward and the termination state.

This subdivision allows to control directly the environment parameters, making possible the inter-cropping interaction. Finally, *InterCropGym* is explaiend in the following lines. During the simulation, two independent OpenCropGym enviroments are run with different crops. The *inter-cropping system module* combines the parameters of the two enviroment, allowing them to update the state. The fertilizer quantity chosen by the agent is given to both the crop - so It's doubled. The final reward is the sum of the reward of the two OpenCropGym modules.

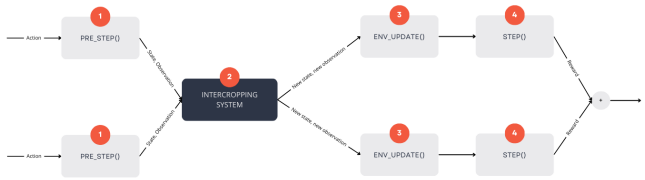

Fig. 1. Intercropping Enviroment

III. INTERCROPPING SYSTEM

The *intercropping system* models the interactions between two crops growing simultaneously through five primary interaction mechanisms. Each mechanism is quantified through specific mathematical formulations that capture both competitive and facilitative interactions.
Note: the model and formulas are not created by experts but just by unexpert handmade considerations, with the purpose of showing the potential of the RL technologies inside the agricoltural domain!

A. Light Competition

Light competition is modeled by considering both the relative canopy position and leaf area index (LAI). The taller crop factor is calculated using:

$$\text{taller\_crop\_factor} = \tanh\left(\frac{\text{WST}_1}{\text{WST}_2 + \epsilon}\right) \tag{1}$$

Where WST represents stem weight and ϵ is a small constant to prevent division by zero. The shading effects are then calculated using sigmoid functions:

$$\text{shading\_on\_crop}_2 = \alpha \cdot \frac{1}{1 + e^{-2(\text{LAI}_1 - \text{LAI}_2)}} \cdot \cdot \text{taller\_crop\_factor} \tag{2}$$

$$\text{shading\_on\_crop}_2 = \alpha \cdot \frac{1}{1 + e^{-2(\text{LAI}_1 - \text{LAI}_2)}} \cdot \text{taller\_crop\_factor} \tag{3}$$

Where α is the light competition factor.

$$\text{light\_effect}_{1/2} = 1 - \text{shading\_on\_crop}_{1/2} \tag{4}$$

B. Water Competition

The water competition model incorporates both transpiration demand and root characteristics:

$$\text{water\_competition}_{1/2} = \frac{\text{TRAN}_{1/2}}{\text{TRAN}_{\text{total}}} \cdot \cdot \frac{\text{ROOTD}_{1/2} \cdot \text{WRT}_{1/2}}{\text{ROOTD}_1 \cdot \text{WRT}_1 + \text{ROOTD}_2 \cdot \text{WRT}_2} \tag{5}$$

Where:

- TRAN is the transpiration rate
- ROOTD is the root depth
- WRT is the root weight

$$\text{water\_effect}_{1/2} = 1 - \psi(1 - \text{water\_competition}_{1/2}) \tag{6}$$

Where ψ is the water interaction factor.

C. Nitrogen Interaction

The nitrogen dynamics are modeled through uptake ratios and inter-crop transfer:

$$\text{N\_ratio}_{1/2} = \frac{\text{NUPTT}_1}{\text{TNSOIL}_{1/2} + \epsilon} \tag{7}$$

$$\text{N\_transfer}_{1/2\to 2/1} = \max(0, (\text{N\_ratio}_{1/2} - \text{N\_ratio}_{2/1}) \cdot \beta) \tag{8}$$

Where:

- NUPTT is nitrogen uptake
- TNSOIL is soil nitrogen
- β is the nitrogen transfer factor

$$\text{nitrogen\_effect}_{1/2} = 1 - \Phi(1 - \text{N\_transfer}_{1/2\to 2/1}) \tag{9}$$

Where Φ is the nitrogen interaction factor.

D. Root Interaction

Root interaction is quantified through spatial overlap and density metrics:

$$\text{overlap\_factor} = \frac{\min(\text{ROOTD}_1, \text{ROOTD}_2)}{\max(\text{ROOTD}_1, \text{ROOTD}_2)} \tag{10}$$

Where ROOTD is the root depth.

$$\text{root\_density}_{1/2} = \frac{\text{WRT}_{1/2}}{\text{ROOTD}_{1/2} + \epsilon} \tag{11}$$

$$\text{total\_density} = \text{root\_density}_1 + \text{root\_density}_2 \tag{12}$$

$$\text{density\_effect}_{1/2} = \frac{\text{root\_effect}_{1/2}}{\text{total\_density}} \tag{13}$$

$$\text{root\_effect}_{1/2} = 1 - \gamma(1 - \text{density\_effect}_{1/2}) \tag{14}$$

Where γ is the root interaction factor.

E. Biomass Effects

Overall biomass competition is determined by relative proportions:

$$\text{biomass\_ratio}_{1/2} = \frac{\text{TAGBM}_{1/2}}{\text{TAGBM}_1 + \text{TAGBM}_2} \tag{15}$$

Where TAGBM represents total above-ground biomass.

$$\text{biomass\_effect}_{1/2} = 1 - \delta(1 - \text{biomass\_ratio}_{1/2}) \tag{16}$$

Where δ is the biomass competition factor.

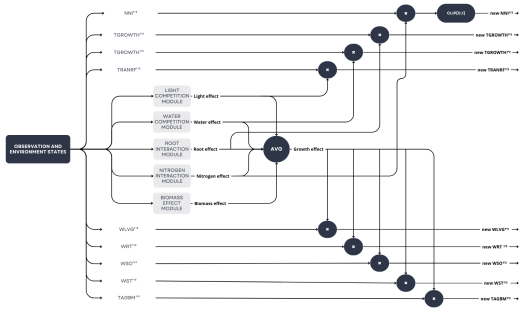

Fig. 2. Intercropping System

F. Combined Growth Effect

The final growth effect for each crop is calculated as the mean of all interaction effects:

$$\begin{aligned}\text{growth\_effect} = \frac{1}{5}(&\text{light\_effect}+\\ &\text{water\_effect}+\\ &\text{nitrogen\_effect}+\\ &\text{root\_effect}+\\ &\text{biomass\_effect})\end{aligned} \tag{17}$$

This growth effect modifies key crop state variables according to:

$$\begin{aligned}\text{WLVG}_{\text{new}} &= \text{WLVG} \cdot \text{growth\_effect}\\ \text{WRT}_{\text{new}} &= \text{WRT} \cdot \text{growth\_effect}\\ \text{WSO}_{\text{new}} &= \text{WSO} \cdot \text{growth\_effect}\\ \text{WST}_{\text{new}} &= \text{WST} \cdot \text{growth\_effect}\\ \text{TAGBM}_{\text{new}} &= \text{TAGBM} \cdot \text{growth\_effect}\\ \text{TGROWTH}_{\text{new}} &= \text{TGROWTH} \cdot \text{light\_effect}\\ \text{TRANRF}_{\text{new}} &= \text{TRANRF} \cdot \text{water\_effect}\\ \text{NUPTT}_{\text{new}} &= \text{NUPTT} \cdot \text{root\_effect}\\ \text{NNI}_{\text{new}} &= \min(1.0, \max(0.0, \text{NNI} \cdot \text{nitrogen\_effect}))\end{aligned} \tag{18}$$

All effects are properly bounded between 0 and 1, representing the range from complete suppression to no interaction effect. This comprehensive modeling approach enables the study of various intercropping combinations and their potential benefits for regenerative agriculture.

IV. Agents

In the presented work, 3 agents have been tested: DDQN, PPO and SAC.

A. Double DQN

The double-DQN[3] (double deep Q-Network) model is a reinforcement learning algorithm which aims to estimate the Q-value of the given state-action pairs, allowing to chose the best action as the one with the highest Q value.
The key features are the following.

- Support of Double Q-Learning. It uses two networks, an *online* for selecting the action and a *target* network to evaluate an action. Only the first one is updated with gradient descent, while every fixed number of steps is copied in the second model. This reduces oscillations and overestimation of the Q value.
- *Replay buffer* support . The past experiences are saved in a circular memory, in such a way that previous examples are used in future parameters' updates. This makes the learning process more stable and allows to re-use multiple times rare events.
- It uses the TD-error δ to update the online network's parameters.
- Balance exploration and exploitation by a decaying variable ϵ.

The action selection algorithm is:

Algorithm 1: DDQN action selection

Input: A current state s
Output: The selected action a
Let $Q_o^\theta(s,a)$ be the online network;
Let ϵ be the exploration rate;
$a \leftarrow argmax_a(Q_o^\theta(s,a))$;
$a_{rand} \leftarrow random(action\_space)$
if $random[0,1] \leq \epsilon$ **then**
    return a_{rand};
else
    return a;

The DDQN training algorithm is described in the following scheme:

Algorithm 2: DDQN training step

Input: A new experience s, a, r, s'
Let $Q_o^\theta(s,a)$ be the online network;
Let $Q_t(s,a)$ be the target network;
Let B be the replay buffer;
Let γ be the future discount factor;
Let ϵ be the exploration rate;
$B \leftarrow push(B, (s,a,r,s'))$;
$S, A, R, S' \leftarrow sample(B, batch\_size)$;
$A' \leftarrow argmax_A(Q_o^\theta(s,a))$;
$\delta = r + \gamma Q_t(S', A') - Q_o^\theta(s,a)$;
$L(\theta) = \mathbb{E}\left[\delta^2\right]$;
$\theta \leftarrow \theta - \alpha \nabla_\theta L(\theta)$;
$\epsilon \leftarrow decay(\epsilon)$;
if *target_update_steps reached* **then**
    $Q_t \leftarrow copy(Q_o^\theta)$;

B. SAC

The SAC [2] (Soft Actor Critic) model is a actor-critic model that encourages exploration by maximizing both the reward and the entropy.
It was originally designed to work with continuous actions, but it has been adapted to work with the discrete domain. The key features are the following.

- Similarly to DDQN, it uses Double Q-Learning to avoid overestimating the Q value and the replay buffer. Also it uses the TD error to update the Q-networks.
- It uses a separate network as policy to select an action.
- An entropy term is introduced in the policy loss to encourage exploration
- Soft updates are used to train the target network instead of deep copy, meaning that the weights represent a convex combination of the target and online networks.

Algorithm 3: SAC training step

Input: A new experience s, a, r, s'
Let $Q_o^\theta(s,a)$ be the online network;
Let $Q_t^\Theta(s,a)$ be the target network;
Let $\pi_\phi(s)$ be the policy;
Let B be the replay buffer;
Let γ be the future discount factor;
Let ξ be the entropy factor.
Let τ be the soft update factor.
$B \leftarrow push(B, (s,a,r,s'))$;
$S, A, R, S' \leftarrow sample(B, batch\_size)$;
$A', \log P(A') \leftarrow \pi_\phi(S')$;
$\delta = r + \gamma Q_t^\Theta(S', A') - (Q_o^\theta(s,a) - \xi \log P(A'))$;
$L(\theta) = \mathbb{E}\left[\delta^2\right]$;
$\theta \leftarrow \theta - \alpha \nabla_\theta L(\theta)$;
$\hat{A}, \log P(\hat{A}) \leftarrow \pi_\phi(S)$;
$L(\phi) = \mathbb{E}[\xi \log P(\hat{A}) - Q_o^\theta(S, \hat{A})]$;
$\phi \leftarrow \phi - \alpha \nabla_\phi L(\phi)$;
$\alpha \leftarrow decay(\alpha)$
if *target_update_steps reached* **then**
    $\Theta \leftarrow (1-\tau)\Theta + \tau\theta$;

C. PPO

The PPO[7] (Proximal Policy Optimization) algorithm is an on-policy reinforcement learning method that directly optimizes a policy network while ensuring the policy updates remain within a trusted region. It uses an actor-critic architecture where the actor learns the policy and the critic evaluates state values.
The key features are the following.

- Uses separate networks for policy (actor) and value estimation (critic)
- Employs a clipped objective function to prevent too large policy updates
- Performs multiple optimization epochs on each batch of data

- Uses Generalized Advantage Estimation (GAE) for more stable training
- Normalizes states and advantages to improve training stability

The PPO training algorithm operates as follows:

Algorithm 4: PPO training step

Input: A batch of experiences (s, a, r, s')

Let π_θ be the actor network;
Let V_ϕ be the critic network;
Let ϵ be the clip range;
Normalize states $s_{norm} = Normalize(s)$;
Compute action distribution $\pi_\theta(s_{norm})$;
Compute current values $V_\phi(s_{norm})$;
Compute log probs $\log \pi_\theta(a|s_{norm})$;
$r(\theta) = \exp(\log \pi_\theta - \log \pi_{old})$;
Compute advantages A using GAE;
$L^\pi = -\text{mean}(\min(r(\theta)A, \text{clip}(r(\theta), 1-\epsilon, 1+\epsilon)A))$;
$L^V = \frac{1}{2}\text{mean}(\max((V_\phi - R)^2,$
$\quad (V_{clipped} - R)^2))$;
$L = L^\pi + c_1 L^V - c_2\text{entropy}$;
Clip gradients and optimize;
Update learning rates;

V. Data Generation

The generation of crop files for the intercropping environment initially aimed to use real-world crop parameters in a format compatible with the LINTUL3 engine. While the lintul3_winterwheat.crop file from CropGym[4][6] provided a template, extensive research revealed that similarly formatted data for other crops, particularly legumes, was not publicly available. Main sources online, such as DSSAT[1], APSIM[5] and PCSE[9] provided data examples in different formats only, requiring significant domain expertise to translate between the different parameter representations. Given that our project focused primarily on reinforcement learning rather than agricultural sciences, we developed an alternative approach to generate representative crop parameter files, as detailed agricultural modeling fell outside both our expertise and the core scope of this work.

The procedure involved three main steps:

1) Collection of reference data from two primary sources:
   - INI format files from the Rlintul repository[10], which provided core physiological parameters
   - YAML format files from WOFOST crop parameters[11], which supplied complementary growth parameters
   - Parameters were extracted where direct correspondence was found between the formats
2) Development of conversion tools:
   - Scripts were coded to handle the automated conversion and merging of parameters from different sources
   - The scripts implemented parameter mapping and conflict detection between different data sources
3) Validation and completion:
   Following steps were executed through the use of a chat-bot model of generative A.I. avaiable online (Claude.ai)
   - Missing parameters were generated through consultation with domain literature
   - Basic parameter sanity checks were performed by comparing against publicly available crop characteristics
   - Parameters were adjusted to maintain consistency with known physiological differences between cereals and legumes

This methodology was applied to generate crop files for four legume species: chickpea, soybean, faba bean, and cowpea. Special attention was given to critical parameters such as nitrogen fixation rates, root development, and growth stage timing.

The resulting files maintain the format required by CropGym while incorporating the specific physiological characteristics of legume crops. This enables the intercropping environment to simulate semi-realistic interactions between different crop combinations.

VI. Experiments and results

In this section we will describe the experiments done on the different agents. In particular the hyperparameters used will be found in the appendix. As a reference, the baseline in the CropGym original paper reaches an average reward value of around -360, and It's also important to take into consideration that the reward of the Intercropping enviroment takes into account the rewards of both the crops. Also, since the data and the intercropping system are approximated, so the results are purely demonstrative.

A. DDQN training

In the DDQN training process, the early stopping has been triggered after 65 episodes. As it's possible to notice from the plots in fig 3 the networks converged easily after only 10 episodes, and the model capacity has been confirmed by a lower exploration rate in the later episodes.
A test experiment on 100 episodes has shown the following average values:

- Reward -695.0
- Weight of storage organs of crop 1 $20.64 \frac{g}{m^2}$
- Weight of storage organs of crop 2 $579.56 \frac{g}{m^2}$
- Total fertilizer used $64.44 \frac{kg}{ha}$

B. SAC training

As It's possible to see from fig 4, the considerations about the learning time are similair to the DDQN training process. The policy loss has steadily decreased in less than 5 episodes, while the Q-networks converged in less than 2 episodes.
A test experiment on 100 episodes has shown the following average values:

- Reward -372.29
- Weight of storage organs of crop 1 $20.64 \frac{g}{m^2}$

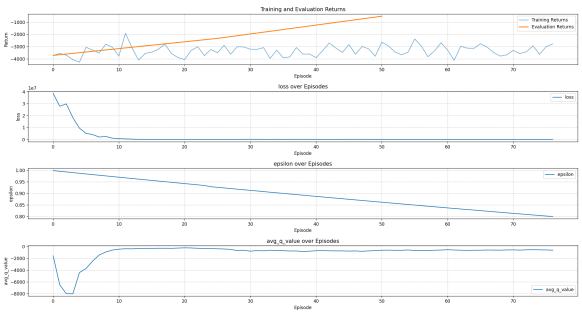

Fig. 3. DDQN training plots

- Weight of storage organs of crop 2 $579.52 \frac{g}{m^2}$
- Total fertilizer used $33.04 \frac{kg}{ha}$

Significant upgrade respect to DDQN case,

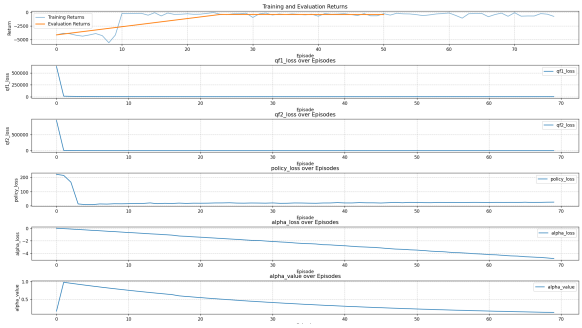

Fig. 4. SAC training plots

C. PPO training

As show in fig 5, the training process has been much slower, taking over 1250 episodes to trigger the early stopping. After analyzing metrics from multiple training sessions, techniques like value function clipping and state normalization were fundamental in order to achieve consistent training.

A test experiment on 100 episodes has shown the following average values:

- Reward -657.94
- Weight of storage organs of crop 1 $20.64 \frac{g}{m^2}$
- Weight of storage organs of crop 2 $579.56 \frac{g}{m^2}$
- Total fertilizer used $60.76 \frac{kg}{ha}$

The results are similair to the DDQN case, but with a slower convergence.

VII. Conclusion

This work demonstrates the application of reinforcement learning techniques to agricultural systems, specifically for modeling and optimizing intercropping practices. By extending the CropGym environment to create InterCropGym, we've established a framework that enables the simulation of interactions between multiple crops growing simultaneously.

Our primary contribution is the development of a baseline implementation that models five key interaction mechanisms between crops: competition for light, water, and nutrients,

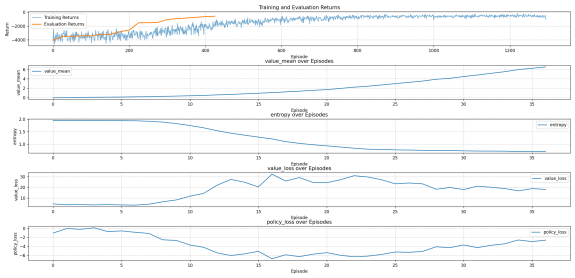

Fig. 5. PPO training plots 1

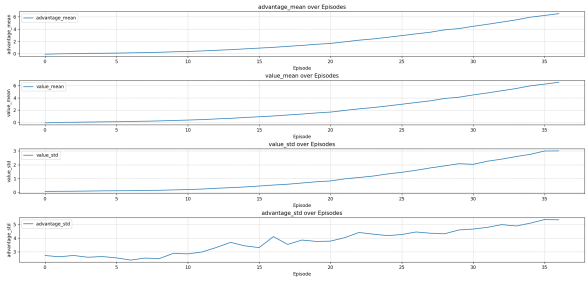

Fig. 6. PPO training plots 2

as well as facilitative effects. This provides a foundation for future research in applying RL to complex agricultural systems.

The comparative analysis of three reinforcement learning approaches (DDQN, SAC, and PPO) aim to provide references for future improvements. The SAC algorithm demonstrated superior performance in optimizing fertilization strategies with higher rewards and lower fertilizer usage, though all three approaches successfully learned viable strategies.

While our current interaction models serve as useful approximations, we acknowledge they would benefit from expert agronomic validation. The environment we've created serves as a starting point that researchers can build upon, refine, and extend to explore sustainable farming practices through simulation.

This work establishes a reference implementation that bridges reinforcement learning and agricultural sciences, potentially contributing to the development of data-driven decision support tools for regenerative agriculture. Future work could focus on incorporating expert domain knowledge, validating the models with field data, and expanding the environment to include additional crops and interaction mechanisms.

VIII. Appendix

A. Architectures

In this chapter are explained the details of the architectures used in the agents.

1) DDQN Q-Networks: The Q-Network used in the DDQN architecture is a 3-layers multi layer perceptron with ReLu activation function and 128 hidden size.

2) SAC Q-Networks: In the SAC agent, the same architecture is used with 256 hidden size. Differently for the previous, 2 identical networks are used and updated simultaneously, but It's used always the smallest value between the two to avoid overestimation.

3) SAC Policy: In the SAC agent, the architecture is modified by adding a softmax activation function after the output layer. Also it supports exploration by choosing a random action with probability ϵ.

B. Training hyperparameters

Parameter		Value
Optimizer	Learning Rate	1e-3
	Optimizer	Adam
	Max gradient norm	10.0
Training Process	Batch Size	256
	Epochs	660
	Eval Frequency	25
	Eval episodes	80
Algorithm	γ	0.99
	Q_t Update Frequency	5
	τ	0.005
Exploration	α	0.2
	$update\_frequency$	1
Replay Buffer	Capacity	5000

TABLE IV
SAC training Hyperparameters

Parameter		Value
Optimizer	Learning Rate	1e-4
	Optimizer	Adam
	Max gradient norm	20.0
Training Process	Batch Size	32
	Epochs	660
	Eval Frequency	25
	Eval episodes	80
Algorithm	γ	0.99
	Q_t Update Frequency	40
Exploration	ϵ_0	1.0
	ϵ_f	0.05
	$decay\_rate$	0.9999
Replay Buffer	Capacity	5000

TABLE V
DQN training Hyperparameters

Parameter		Value
Optimizer	Learning Rate	5e-4
	Optimizer	Adam
	Max gradient norm	0.5
Network	Actor hidden sizes	[256, 256]
	Critic hidden sizes	[64, 64]
	Activation	ReLU
Algorithm	γ	0.99
	GAE λ	0.95
	Clip_range	0.2
	Entropy coefficient	0.05
	Value function coefficient	0.5
Training Process	Buffer size	4096
	Batch Size	512
	Epochs per update	10
	Eval Frequency	25
	Eval episodes	80

TABLE VI
PPO training Hyperparameters

References

[1] Phillip D Alderman. *DSSAT: A Comprehensive R Interface for the DSSAT Cropping Systems Model*. R package version 0.0.9. 2024. DOI: 10.5281/zenodo.4091381. URL: https://CRAN.R-project.org/package=DSSAT.

[2] Tuomas Haarnoja et al. "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor". In: *arXiv:1801.01290* (2018).

[3] Hado van Hasselt, Arthur Guez, and Google DeepMind David Silver. "Deep Reinforcement Learning with Double Q-learning". In: *arXiv:1509.06461v3* (2015).

[4] Ioannis N. Athanasiadi Hiske Overweg Herman N. C. Berghuijs. "CROPGYM: A REINFORCEMENT LEARNING ENVIRONMENT FOR CROP MANAGEMENT". In: *arXiv:2104.04326v2* (2021).

[5] Dean P. Holzworth et al. "APSIM – Evolution towards a New Generation of Agricultural Systems Simulation". In: *Environmental Modelling & Software* 62 (Dec. 2014), pp. 327–350. DOI: 10.1016/j.envsoft.2014.07.009.

[6] Michiel G.J. Kallenberg et al. "Nitrogen management with reinforcement learning and crop growth models". In: *Environmental Data Science* 2 (2023), e34. DOI: 10.1017/eds.2023.28.

[7] John Schulman et al. "Proximal Policy Optimization Algorithms". In: *arXiv:1707.06347* (2017).

[8] M. E. Shibu et al. "LINTUL3, a simulation model for nitrogen-limited situations: Application to rice". In: *European Journal of Agronomy* (2010).

[9] Allard de Wit. *PCSE - Python Crop Simulation Environment*. Accessed: 2024-02-21. 2024. URL: https://github.com/ajwdewit/pcse.

[10] Allard de Wit. *Rlintul: R implementation of LINTUL crop growth models*. https://github.com/cropmodels/Rlintul/tree/master/inst/lintul/crop. Accessed: 2024-02-14. 2019.

[11] Allard de Wit. *WOFOST crop parameters*. https://github.com/ajwdewit/WOFOST_crop_parameters/tree/master. Accessed: 2024-02-14. 2019.