UBC CHBE

Reinforcement Learning for Multi-Loop PID Control

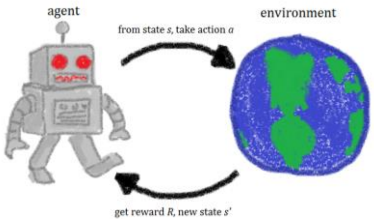
Kenechukwu Ene

Supervisor: Dr. Bhushan Gopaluni, UBC Chemical and Biological Engineering

Collaborators:
Dr. Philip Loewen, UBC Mathematics
Dr. Michael Forbes, Honeywell Process Solutions
Johan Backstrom, Backstrom Systems Engineering
Nathan Lawrence, PhD Candidate at UBC DAIS Lab

April 1, 2022

UBC Data Analytics and Intelligent Systems Lab



Project Recap

Recap

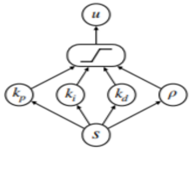
Motivation: Industry 4.0

- Expensive to refit model-based control strategies as plants evolve (Lawrence et al., 2020)
- Desired flexibility can be achieved via reinforcement learning (Sutton and Barto, 2018)

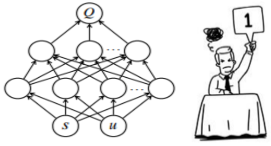
DAIS Lab Publication

- Reinforcement learning optimizes a controller’s policy through interactions with its environment
- Actor-Critic reinforcement learning algorithm that controls a PI controller

Actor-Critic System in PI Controller: Adjusting PI gains to achieve setpoint



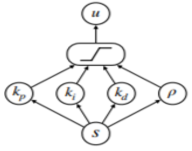
Actor generates initial input



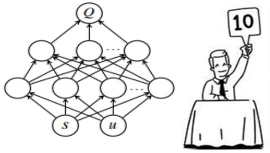
Critic assesses expected rewards

Critic advices Actor on policy changes

Policy gradient



Actor generates new input



Critic assesses expected rewards

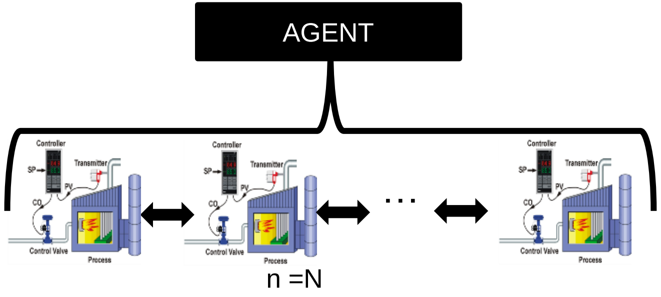
Recap

DAIS Lab

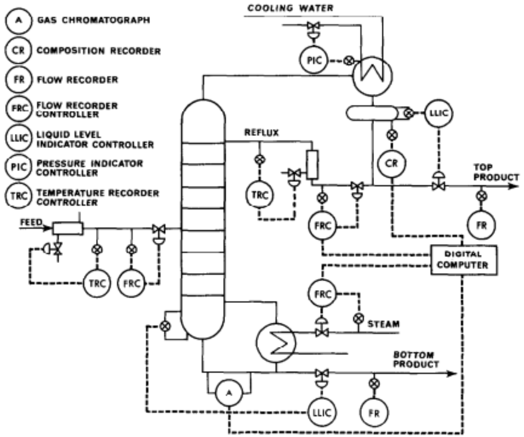
- RL agent for one PI controller

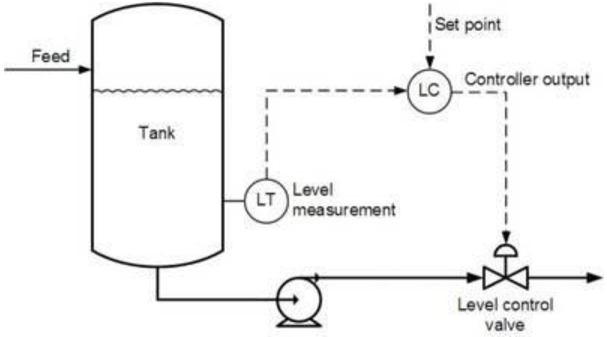
Goal

- RL agent for several, connected PI controllers



Chosen Scenario





Implementation

Implementation Changes – RL Libraries and Algorithms

Spinning Up	Stable Baselines 3
Vanilla Policy Gradient (VPG)	Advantage Actor Critic (A2C)
Deep Deterministic Policy Gradient (DDPG)	
Trust Region Policy Optimization (TRPO)	Hindsight Experience Replay (HER)
Proximal Policy Optimization (PPO)	
Soft Actor-Critic (SAC)	
Twin Delayed DDPG (TD3)	

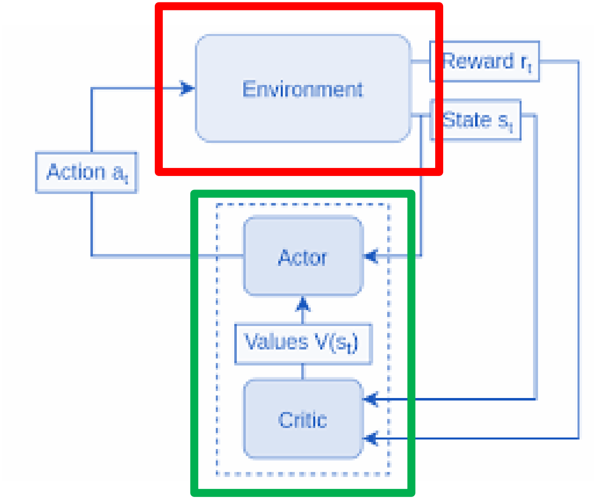
TD3:

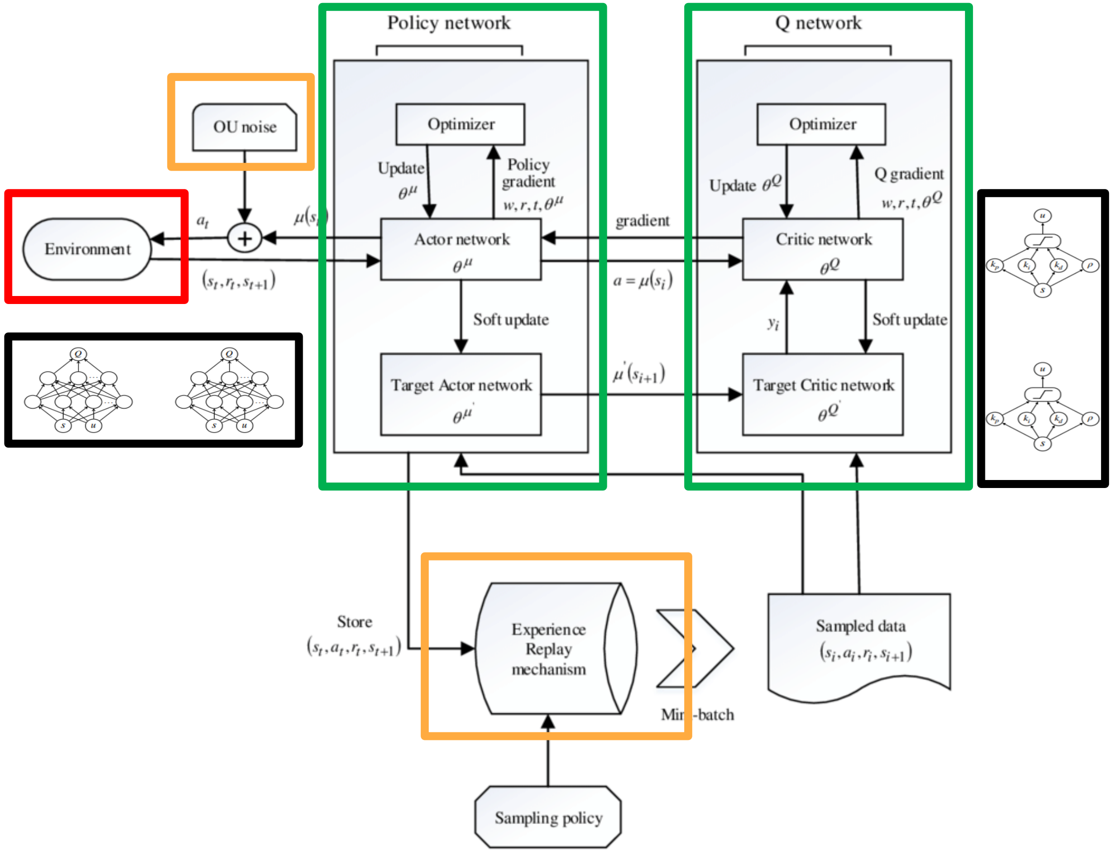
- Addresses the instability of DDPG (Raffin, 2021)
- Can have more critics so less biased than DDPG

DDPG

TD3

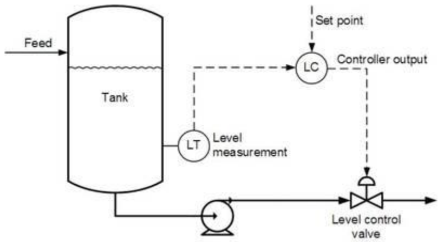
Explaining TD3 using DDPG





The Environment

Creating the Environment



Class Environment_Name(gym.Env):

Initialization Function:

- Transfer function and Matlab control toolbox
- Action and observation spaces

$$Transfer\ Function: \frac{H'(s)}{Q'(s)} = \frac{0.12}{11.31s + 1}$$

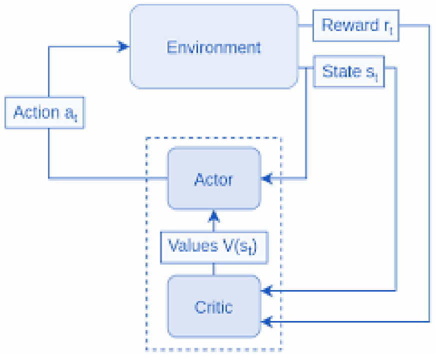
Reward Function:

- Conditions agent to achieve goal
- Want convergence at maxima

$$Reward = -(y - y_{sp})^2$$

```python
## TANK LEVEL CONTROL ENVIRONMENT
class FirstTank(gym.Env):

  # Initialization Function
  def __init__(self, max_ep_len = 200, timestep=0.1, setpoint = 1):

    # Initialization of variables:
    self.num = [0.12]
    self.den = [11.31, 1]
    self.max_ep_len = max_ep_len #[in seconds]
    self.ts = timestep #[in seconds]
    self.sp = setpoint #[in metres]
    self.t = 0 #[in seconds]

    # Matlab's Control Toolbox Functions:
    self.sysc = control.matlab.tf(self.num, self.den)
    self.sys = control.matlab.c2d(self.sysc, self.ts, method='zoh')
    self.ss_sys = control.tf2ss(self.sys)
    self.x = np.zeros(np.transpose(self.ss_sys.C).shape)

    # Action and Observation Spaces:
    self.size_act = np.array([-1000.0, 1000.0])
    self.action_space = spaces.Box(low=self.size_act[0],
                                   high=self.size_act[1], shape=(1,))
    size_obs = np.array([-1000.0, 1000.0])
    self.obs_dim = size_obs.shape
    self.observation_space = spaces.Box(low=size_obs[0],
                                        high=size_obs[1], shape=self.obs_dim)
```

Creating the Environment



Class Environment_Name(gym.Env):

…

Step Function:

- Paves agent's path through environment
- Iterative component e.g., time = time + 1
- Stopping criteria

$$State\ Space\ Representation:$$

$$x' = -0.0884173x + u$$
$$y = 0.0106101x$$

Render Function: For visualization

Reset Function: Resets the environment

Close Function: Closes the environment

```python
# Step Function
def step(self, action):

  # Iterative Aspect:
  self.t = self.t + 1
  self.action = action.squeeze()
  self.u = self.action
  self.x = np.dot(self.ss_sys.A, self.x) + np.dot(self.ss_sys.B, self.u)
  self.y = np.dot(self.ss_sys.C, self.x).item()
  self.error = self.sp - self.y
  self.int += self.ts*self.error
  self.state = np.array([self.error, self.int])

  # Stopping Criteria:
  if self.t > self.max_ep_len:
    self.done = True

  return self.state, self.reward(), self.done, {}
```

Model Realization

Stages of RL Implementation

Training → **Evaluation** → **Testing**

Training
- Teach the agent how to achieve the goal

Evaluation
- Assess how well the agent trained

Testing
- Deploy the trained agent

Training

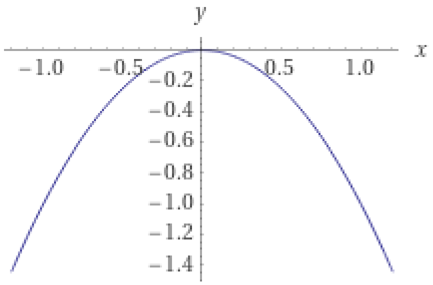
- Involves tuning hyperparameters to ensure the RL agent is set up for success
- Too many hyperparameters to tune and can assign values from endless list of values

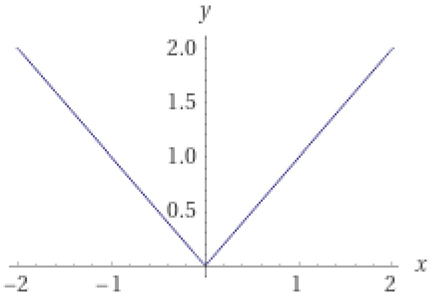
Hyperparameter Tuning: An Imperfect Science

Select hyperparameters → Train agent → Evaluate → Complete or Repeat

Choosing the Reward Function

- Linear or quadratic?
- How much to incorporate actor?
- Conditional?





$$Reward = -|\mathrm{y} - y_{sp}|^2 - \mathrm{a}|\mathrm{u}|$$

$$Reward = -|\mathrm{y} - y_{sp}|^2$$

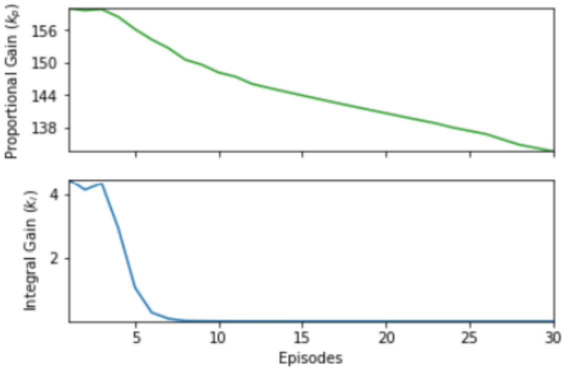
$$Reward = -|\mathrm{y} - y_{sp}|$$

where e.g., a = 1E-10 to 1E +10

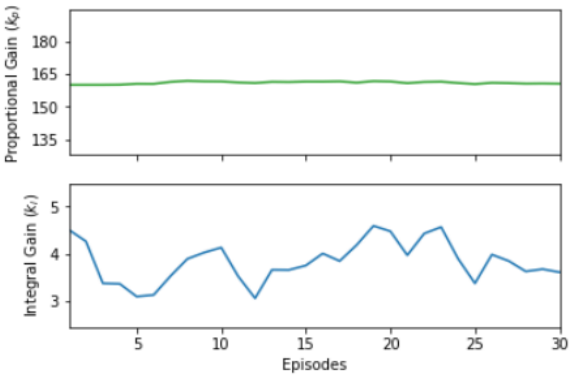
Hyperparameter Tuning: OU Noise

```
# Noise
n_actions = env.action_space.shape[0]
action_noise = OrnsteinUhlenbeckActionNoise(mean=np.zeros(n_actions),
                                            sigma=(0)*np.ones(n_actions))
```

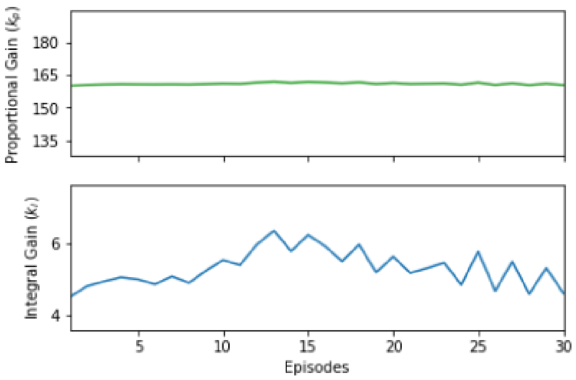
Here, **a = 0**



a = 0.01

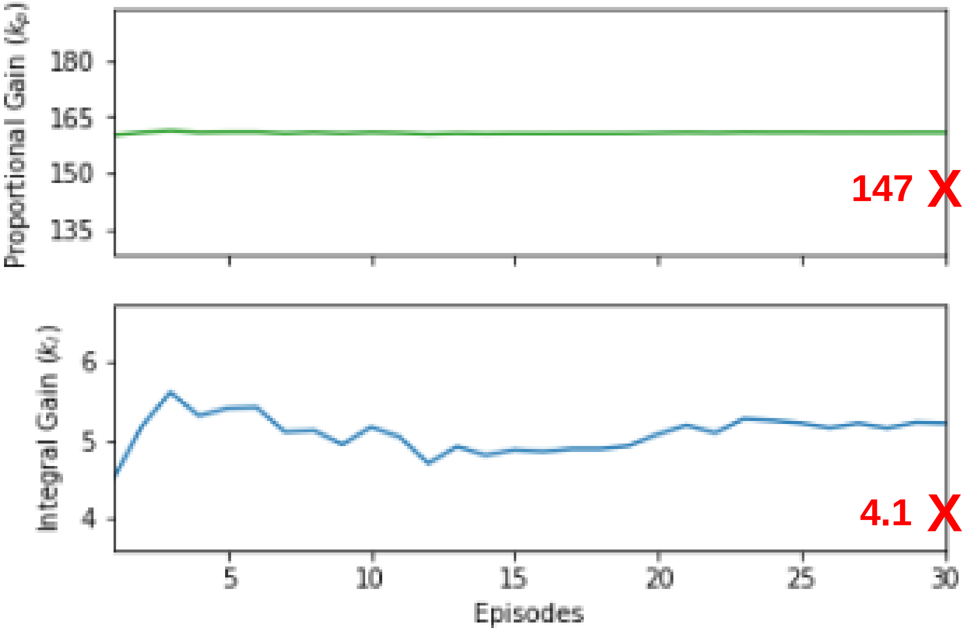


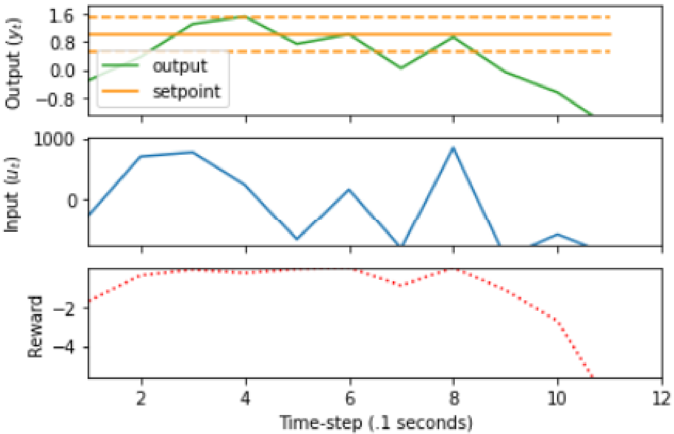
a = 0.1

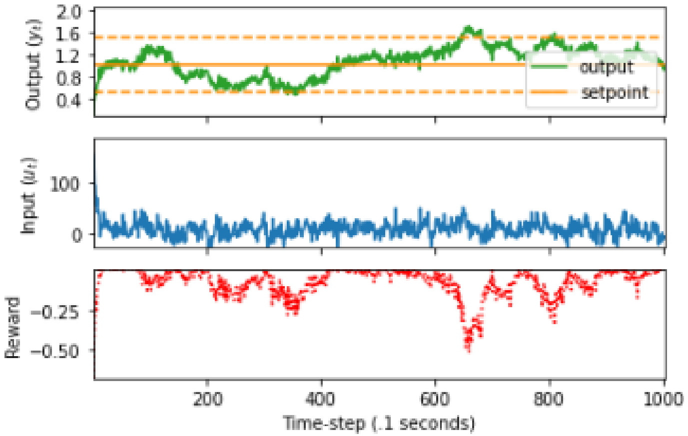


a = 0.06

Validation





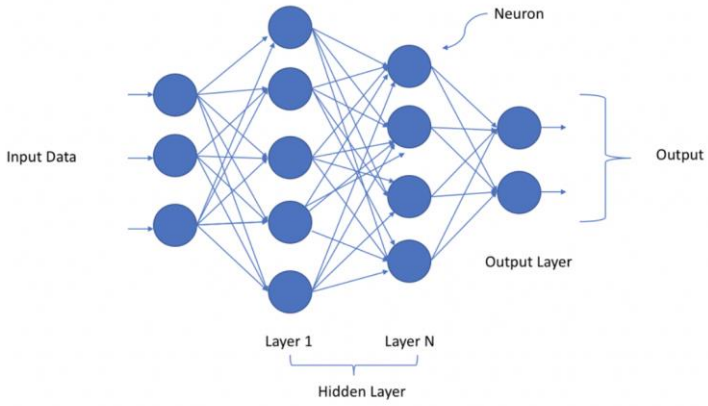


MATLAB

RL against the World

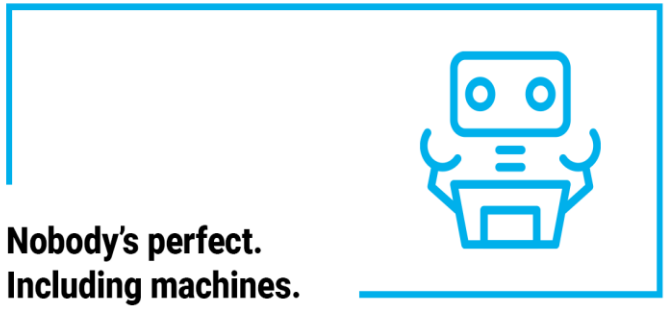
From Obstacles to Improvements

- Knowledge and experience
  - More steps to have taken before RL
  - Difficulty determining what success looked like
- Hyperparameter tuning
  - Initial values of k_P and k_I of actor network
  - Unknown seed values of critic DNN [400, 300]
  - Inconsistency of results
  - Difficult to track hyperparameters
- RL Libraries
  - Inconsistencies between libraries and authors
  - Open-source code sometimes not updated



Feasibility in Industry

- Unpredictability – Black box
- Need for a representative environment
- Lots of data required for RL agent to learn
- Better alternatives exist – MPC
- Troubleshooting requires uncommon expertise
- Scalability – Curse of dimensionality



Conclusion

Has the project's objective been achieved?

SEP | OCT | NOV | DEC | JAN | FEB | MAR | APR

- Installation and course on practical RL
- Legacy code and TD3 architecture
- Adaptation of legacy code to new RL library
- Creation of environment and training

- Optimize implementation of SISO system
- Consider extending to a MIMO system
- Thesis Report

Expert Consultation (DAIS Lab)

Research Papers

Stack Overflow, RL in practice course

Process Control: CHBE 356, 456

Acknowledgments

Special thanks to the supportive community at the UBC Data Analytics and Intelligent Systems (DAIS) Laboratory

Dr. Bhushan Gopaluni
Principal Investigator, UBC Chemical and Biological Engineering

Dr. Philip Loewen
Professor, UBC Department of Mathematics

Dr. Michael Forbes
Lead Research Engineer, Honeywell Process Solutions

Johan Backstrom
President, Backstrom Systems Engineering Ltd.

Nathan Lawrence
PhD Candidate

Shuyuan Wang
PhD Candidate

Daniel McClement
MASc Candidate

Frida Bjornstad Konow
Visiting Scholar

UBC Data Analytics and Intelligent Systems Lab

References

N. P. Lawrence, G. E. Stewart, P. D. Loewen, M. G. Forbes, J. U. Backstrom, and R. B. Gopaluni, “Optimal PID and Antiwindup Control Design as a Reinforcement Learning Problem,” *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 236–241, 2020. [Accessed: 29-Nov-2021]

N. P. Lawrence, M. G. Forbes, P. D. Loewen, D. G. McClement, J. U. Backstrom, and R. B. Gopaluni, “Deep Reinforcement Learning with Shallow Controllers: An Experimental Application to PID Tuning,” *arXiV,* 2021. [Accessed: 29-Nov-2021]

Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., Dormann, N. (2021). Stable-Baselines3: Reliable Reinforcement Learning Implementations. *Journal of Machine Learning Research, 22 (268).* http://jmlr.org/papers/v22/20-1364.html.

R. K. Wood and M. W. Berry, "Terminal composition control of a binary distillation column," *Chemical Engineering Science,* vol. 28, *(9),* pp. 1707-1717, 1973. [Accessed: 29-Nov-2021].

R. S. Sutton, F. Bach, and A. G. Barto, *Reinforcement Learning: An Introduction*. Massachusetts: MIT Press Ltd, 2018.

Thank you for listening!

Questions?

UBC Data Analytics and Intelligent Systems Lab