

A Reinforcement Learning Framework for Simultaneous Chemical Process Flowsheet Generation, Design and Control

by

Simone Reynoso Donzelli

A thesis

presented to the University of Waterloo

in fulfillment of the

thesis requirement for the degree of

Master of Applied Science

in

Chemical Engineering

Waterloo, Ontario, Canada, 2024

© Simone Reynoso Donzelli 2024

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Integration of process design and control of chemical process flowsheets (CPFs) is a key focus in chemical engineering, receiving extensive research attention. The main objective in this area is to identify optimal process design and control variables for a CPF, ensuring both economic viability and dynamic feasibility of plant operations. This integration presents a complex optimization problem, which is challenging to solve using traditional optimization methods. Additionally, the problem becomes even more intricate when discrete decisions or logical constraints, which give rise to Boolean variables, are considered—common in integrated design and control of CPFs problems. Therefore, the development of new methodologies is needed to effectively address these challenges. The emerging trend in Machine Learning (ML), particularly in Reinforcement Learning (RL), for solving such problems serves as the foundation for this thesis. The limited studies regarding the solution of the integrated problem using RL techniques motivates the exploration and development of novel methodologies.

Before addressing the integrated problem, it is important to understand the potential of RL as a tool for solving design and optimization problems of CPFs under steady-state conditions. A RL methodology that introduces two novel RL agents: a discrete masked Proximal Policy Optimization (mPPO) and a hybrid masked Proximal Policy Optimization (mHPPO) has been proposed. In this framework, the agents are capable of autonomously generate, design and optimize CPFs utilizing an inlet flowrate and a set of unit operations (UOs) as initial information. A key feature of this approach is the use of masking – an underexplored yet promising area for solving the present problem – which involves the incorporation of expert knowledge or design rules to exclude certain actions from the agent's decision-making process, enhancing the agent's performance. Adding to that, this method stands out by seamlessly integrating masked agents with rigorous models of UOs, including advanced thermodynamic and conservation equations, within its simulation environment. The effectiveness of these agents was evaluated through several case studies, including two that utilized commercial simulation suites as part of the RL environment. The resulting CPFs generated by the RL agents present viable flowsheet designs that meet the pre-specified design requirements.

Recognizing the potential of RL for designing CPF, this thesis also introduces a novel RL approach for generating, designing, and controlling CPFs. Similar to the previous methodology, the proposed framework generates CPFs directly from an inlet stream, eliminating the need for predefined arrangements of UOs. Furthermore, the framework leverages surrogate models, specifically Neural

Networks (NNs), to accelerate the learning process of the RL agent and avoid dependence on mechanistic dynamic models. These surrogate models approximate key process variables and closed-loop performance metrics for complex dynamic UO models. The results obtained using this methodology were compared with model-based optimization results to assess the accuracy and validity of the proposed approach in approximating well-established methodologies. Consistency with the model-based approach was assessed. Additional case studies involved formulations with multiple UOs to further demonstrate the approach's flexibility to deal with various scenarios. Results from those case studies demonstrate that the RL agent can effectively learn to maintain the dynamic operability of the UOs under disturbances, adhere to equipment design and operational constraints, and generate viable and economically attractive CPFs. The high adaptability offered by the surrogate models enables this methodology to approximate the dynamic behavior of the most common UO. As a result, the proposed framework is sufficiently explicit and flexible to be used in more intricate design and control problems involving multiple UOs.

Acknowledgments

First and foremost, I would like to thank my supervisor, Prof. Luis A. Ricardez-Sandoval, for the opportunity to pursue my master's studies. I appreciate your guidance and for constantly challenging me to reach my full potential.

I would also like to thank my examining committee: Prof. Hector Budman and Prof. Nasser Mohieddin Abukhdeir for their time and valuable feedback.

My gratitude extends to the University of Waterloo and Cooper-Standard for the financial support provided throughout my master's program. Without this support, my studies would not have been possible.

I want to express my deepest gratitude to my entire family. Without their support, obtaining my degree would not have been possible. Thank you, beloved Mamma, for your love, all your prayers, patience, and support. Thank you, beloved Fiorella, for your advice, encouragement, and unconditional love. Thank you, beloved Nonna, for your love and tireless prayers. Thank you, beloved Zia Gaby, for your strength, advice, and love. Thank you, Matteo and Zio Francesco, for always supporting me. Thank you, Tio Pepe and Tia Lore, for your love and support. Thank you, Tio Alex and Tia Pivis, for your support.

Thanks to my dearest friends, Carla Aguilar, Juan de la Luz Enriquez, Armando Gaxiola and Alfonso Llaca, for always being there. Thanks to my new friends Pablo Kowalski, Jesús Minakata, Miguel Angel Guerra and Alfonso Izquierdo. A special thanks to Daniel Rangel, for being a great friend and a consultant on existential and academic problems. You were like a co-supervisor to me. A special thanks to the all the members of the research group (David, Sam, Carlos, Hao, Shayesteh, Meagan, Oscar, Gabriel, Fernando, Jaime), it was nice sharing joys and pains with all of you.

Infinite thanks to Mrs. Clara for welcoming me into her home, caring for me like a son, and bringing me closer to the ways of God.

Finally, I know you are very proud. Thank you, Papito, for watching over me from heaven. Thank you, Vita, Totis, and Nonno. Thank you, God, for your love and for teaching me so much about life in these two years. Thank you Virgencita de Guadalupe for always protecting me and praying for me.

Dedication

This thesis is dedicated to my parents: to Adrián Dagoberto Reynoso Del Valle (1962-2018), my role model, the man I aspire to become, the best father I could have wished for; and to Claudia Donzelli Palmieri, my comfort in sorrow, my encouragement in adversity, the greatest maternal love I could ever desire.

Table of Contents

Author's Declaration	ii
Abstract	iii
Acknowledgments	v
Dedication	vi
List of Figures	ix
List of Tables	xi
List of Abbreviations	xiii
List of Symbols	xv
Chapter 1 Introduction.....	1
1.1 Research objectives and contributions	4
1.2 Structure of the Thesis.....	6
Chapter 2 Literature Review	7
2.1 Reinforcement Learning and its Applications in Chemical Engineering	7
2.1.1 Reinforcement Learning: An Overview	8
2.1.2 Proximal Policy Optimizer (PPO)	11
2.1.3 Applications of Reinforcement Learning in Chemical Engineering	15
2.2 Chemical Process Flowsheet Design and Optimization	15
2.3 Simultaneous Generation, Design and Control of Chemical Process Flowsheet	20
2.4 Summary	25
Chapter 3 A Reinforcement Learning approach with masked agents for chemical process flowsheet design.....	26
3.1 Solution framework.....	27
3.1.1 Agents.....	28
3.1.2 Environment	30

3.2 Case studies	35
3.2.1 Case study 1.....	35
3.2.2 Case study 2.....	40
3.2.3 Case study 3.....	51
3.3 Summary	57
Chapter 4 An integrated Reinforcement Learning framework for simultaneous generation, design, and control of chemical process flowsheets	59
4.1 Solution framework.....	60
4.1.1 Surrogate models.....	61
4.1.2 Environment	67
4.2 Case studies	71
4.2.1 Case study 1.....	72
4.2.2 Case study 2.....	81
4.2.3 Case study 3.....	89
4.2.4 Case study 4.....	99
4.3 Summary	110
Chapter 5 Conclusions and Future Work	112
5.1 Recommendations for future work.....	114
References	116
Letters of Copyright Permissions	127
Appendix A Chapter 3 – Case study 1	134
Appendix B Chapter 4 – Case study 1.....	139
Appendix C Chapter 4 – Case study 2.....	149
Appendix D Chapter 4 – Case study 3	162
Appendix E Chapter 4 – Case study 4.....	168

List of Figures

Figure 2.1: RL diagram.....	8
Figure 3.1: Agent-environment interaction.....	27
Figure 3.2: Illustrative example of the masking functionality.....	29
Figure 3.3. Network architecture for the mHPPO actor.....	29
Figure 3.4: Learning curve for a single run: A) mHPPO B) mPPO.....	37
Figure 3.5: Reward and losses profiles for mHPPO.....	38
Figure 3.6: CPF obtained for case study 1: A) mHPPO B) mPPO	39
Figure 3.7: Individual learning curve for HPPO: A) unmasked B) masked	46
Figure 3.8: Average learning curve for 10 runs: A) mHPPO B) mPPO	47
Figure 3.9: CPF obtained for case study 2: A) mHPPO B) mPPO	48
Figure 3.10: Individual learning curve for mHPPO case study 3	54
Figure 3.11: CPF obtained for case study 3	55
Figure 4.1: RL diagram.....	60
Figure 4.2: Dataset building	64
Figure 4.3: Actor network architecture.....	70
Figure 4.4: PPO agent's learning curve.....	75
Figure 4.5: Dynamic responses Scenario A of: A) CSTR 1 B) CSTR 2.....	76
Figure 4.6: Dynamic responses in Scenario B of: A) CSTR 1 B) CSTR 2	79
Figure 4.7: . Dynamic response of the CSTR for Scenario C	80
Figure 4.8: PPO agent's learning curve	85
Figure 4.9: Final flowsheet for both scenarios	86
Figure 4.10: Dynamic responses Scenario D of: A) CSTR 1 B) CSTR 2 C) CSTR 3.....	87
Figure 4.11: Dynamic responses Scenario D of DC: A) Bottoms B) Overheads.....	87
Figure 4.12: Dynamic responses Scenario E of: A) CSTR 1 B) CSTR 2 C) CSTR 3	88
Figure 4.13: Dynamic responses Scenario E of DC: A) Bottoms B) Overheads	89
Figure 4.14: PPO agent's learning curve	92
Figure 4.15: Dynamic responses Scenario F of: A) CSTR 1 B) CSTR 2 C) CSTR 3	94
Figure 4.16: Dynamic response of DC Scenario F	94
Figure 4.17: PPO agent's learning curve	96
Figure 4.18: Final flowsheet Scenario G	97

Figure 4.19: Dynamic responses Scenario G of: A) CSTR 1 B) CSTR 2 C) CSTR 3.....	97
Figure 4.20: Dynamic response of FT Scenario G	98
Figure 4.21: PPO agent's learning curve	102
Figure 4.22: Final flowsheet Scenario H	104
Figure 4.23. Dynamic response Scenario H of CSTR	104
Figure 4.24. Dynamic responses Scenario H of DC 1: A) Bottoms B) Overheads	105
Figure 4.25. Dynamic responses Scenario H of DC 2: A) Bottoms B) Overheads	106
Figure 4.26. Final flowsheet for Scenario I	107
Figure 4.27. Dynamic responses Scenario I of: A) CSTR 1 B) CSTR 2	108
Figure 4.28. Dynamic responses Scenario I of DC 1: A) Bottoms B) Overheads.....	109
Figure 4.29. Dynamic responses Scenario I of DC 2: A) Bottoms B) Overheads.....	109
 Figure A.1: Learning curves of mPPO for different number of discretization points.....	 137
 Figure B.1: Correlations matrix.....	 143
Figure B.2: Temperature constraint violation Scenario A: A) CSTR 1 B) CSTR 2	146
Figure B.3: Learning curve for PPO agent Scenario C.....	147
Figure B.4: Temperature constraint violation Scenario C.....	148
 Figure C.1: Correlation matrix.....	 152
Figure C.2: Volume constraint violation Scenario D: A) CSTR 1 B) CSTR 2 C) CSTR 3	153
Figure C.3: Volume constraint violation Scenario E: A) CSTR 1 B) CSTR 2 C) CSTR 3	154
Figure C.4: Correlation matrix.....	160
 Figure D.1: Correlation matrix.....	 165
Figure D.2: Volume constraint violation Scenario F: A) CSTR 1 B) CSTR 2 C) CSTR 3	166
Figure D.3: Volume constraint violation Scenario G: A) CSTR 1 B) CSTR 2 C) CSTR 3.....	167
 Figure E.1 Volume constraint violation scenario H.....	 169
Figure E.2 Volume constraint violation Scenario I: A) CSTR 1 B) CSTR 2	169
Figure E.3 Correlation matrix for DC with $\alpha=3.5$	171

List of Tables

<i>Table 2.1: Studies that have applied RL for CPF design and optimization</i>	<i>18</i>
<i>Table 3.1: Action space example.....</i>	<i>31</i>
<i>Table 3.2: Reward shaping.....</i>	<i>33</i>
<i>Table 3.3: Sub-rewards case study 1.....</i>	<i>36</i>
<i>Table 3.4: UOs designs and specifications case study 1</i>	<i>40</i>
<i>Table 3.5: UOs description case study 2.....</i>	<i>41</i>
<i>Table 3.6: Sub-rewards case study 2.....</i>	<i>43</i>
<i>Table 3.7: UOs designs and specifications for case study 2</i>	<i>49</i>
<i>Table 3.8. Cumulative reward breakdown</i>	<i>50</i>
<i>Table 3.9: UOs for case study 3</i>	<i>52</i>
<i>Table 3.10: Sub-rewards case study 3.....</i>	<i>53</i>
<i>Table 3.11: UOs design and specifications for case study 3.....</i>	<i>56</i>
<i>Table 4.1: Sub-rewards for case study 1</i>	<i>73</i>
<i>Table 4.2: UOs designs and specifications for case studies of scenarios A, B and C.....</i>	<i>76</i>
<i>Table 4.3. Cumulative reward breakdown</i>	<i>77</i>
<i>Table 4.4: Objective function results for Scenario B</i>	<i>78</i>
<i>Table 4.5: Surrogate models used in case study 2</i>	<i>81</i>
<i>Table 4.6: Sub-rewards for case study 2</i>	<i>83</i>
<i>Table 4.7: UOs designs and specifications for case study 2 scenarios D and E.....</i>	<i>86</i>
<i>Table 4.8: FT's surrogate models</i>	<i>90</i>
<i>Table 4.9: Sub-rewards for FT.....</i>	<i>91</i>
<i>Table 4.10: UOs designs and specifications for case study 3 scenarios F and G.....</i>	<i>93</i>
<i>Table 4.11 Relative volatilities.....</i>	<i>99</i>
<i>Table 4.12: UOs designs and specifications for case study 3 scenarios F and G.....</i>	<i>103</i>
 <i>Table A.1: Parameters for CSTR.....</i>	 <i>135</i>
 <i>Table B.1 Parameters values.....</i>	 <i>141</i>
<i>Table B.2: Design variables' bounds</i>	<i>142</i>
<i>Table B.3: Surrogate models' details</i>	<i>144</i>

<i>Table B.4: Surrogate model validation</i>	145
<i>Table C.1: Parameters' values</i>	151
<i>Table C.2: Design variables' bounds</i>	151
<i>Table C.3: Surrogate models' details</i>	153
<i>Table C.4: Parameters' values</i>	157
<i>Table C.5: Design variables' bounds</i>	158
<i>Table C.6: Surrogate models' details</i>	160
<i>Table D.1 Parameters' values</i>	163
<i>Table D.2: Design variables' bounds</i>	163
<i>Table D.3: Surrogate models' details</i>	166
<i>Table E.1 Design variables</i>	170
<i>Table E.2 Surrogate models' details for DCs</i>	172

List of Abbreviations

Abbreviation	Definition
A2C	Advantage Actor-Critic
AI	Artificial Intelligence
APFR	Adiabatic Plug Flow Reactor
C	Cooler
CPF	Chemical Process Flowsheet
CPU	Central Processing Unit
CSTR	Continuous Stirred Tank Reactor
DAE	Differential Algebraic Equations
DCR	Distillation Column with Recycle
DC	Distillation Column
DDPG	Deep Deterministic Policy Gradient
DDQN	Double Deep Q-Network
DME	Dimethyl Ether
DQN	Deep Q-Network
FT	Flash Tank
GAE	Generalized Advantage Estimator
HEX	Heat Exchanger
HK	Heavy Key
HPPO	Hybrid Proximal Policy Optimizer
IAE	Integral Absolute Error
IPOPT	Interior Point OPTimizer
ISE	Integral Squared Error
LHS	Latin Hypercube Sampling
LK	Light Key
M	Mixer
MeOH	Methanol
mHPPO	Masked Hybrid Proximal Policy Optimizer
MIDO	Mixed-Integer Dynamic Optimization
MINLP	Mixed-Integer Nonlinear Programming
ML	Machine Learning

mPPO	Masked Proximal Policy Optimizer
MPC	Model Predictive Control
NLP	Nonlinear Programming
NN	Neural Network
NRTL	Non-Random Two-Liquid
P	Pump
PFR	Plug Flow Reactor
PI	Proportional-Integral
PID	Proportional-Integral-Derivative
PINN	Physics-Informed Neural Network
PG	Propylene Glycol
PO	Propylene Oxide
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
SARSA	State-Action-Reward-State-Action
SMOTE	Synthetic Minority Over-sampling Technique
Tanh	Hyperbolic Tangent
TC	Ternary Column
TCR	Ternary Column with Recycle
UO	Unit Operation
UNIQUAC	Universal Quasichemical Activity Coefficient

List of Symbols

Symbol	Definition
$\mathbf{A}$	Action space
a_i	Action at step i
$\hat{A}_i$	Advantage function at step i
A_R	Heat transfer area of the reactor
α	Relative volatility
α_B, β_B	Beta distribution parameters
A_{CSTR}	Cross-sectional area of CSTR
$\mathbf{A}_{ex}$	Action space example
$\mathbf{A}_{FD}$	Action space fully discrete
β	Tray time constant
B	Beta distribution
B^L	Bottoms flow rate in DC
C_A	Outlet concentration of A
$\mathbf{C}_A$	Concentration of A vector
$C_{A_{in}}$	Inlet concentration of A
C_{A_L}	Outlet concentration of A at L_{FT}
$C_{A_{rec}}$	Recycle concentration of A
C_A^{sp}	Set-point concentration of A
C_{A_0}	Initial concentration of A
C_B	Outlet concentration of B
$C_{B_{in}}$	Inlet concentration of B

C_c	Outlet concentration of C
$C_{C_{in}}$	Inlet concentration of C
C_p	Liquid heat capacity
C_{cool}	Collant heat capacity
$cons$	Constraint
$cons_{DC}$	Constraint design of DC
$cons_T$	Temperature constraint
$cons_V$	Reaction volume constraint
$cost_{CSTR}$	Cost CSTR
$cost_{DC}$	Cost DC
$cost_{FT}$	Cost FT
$cost_M$	Cost mixer
$cost_R$	Cost reactor
χ	Intermediate continuous variables calculated by the chemical process simulator
Δ	Difference in distinctive parameter
ΔH_R	Heat of reaction
D^{APFR}	Diameter APFR
D_C	Column diameter
D_{CSTR}	Diameter of CSTR
D^L	Distillate flowrate in DC
D^{PFR}	Diameter PFR
$\mathbf{d}(t)$	Disturbances affecting the system
$dist$	Distillate flowrate

δ_i	Temporal difference error at step i
E_A/R_{cte}	Energy of activation per gas constant
E_A	Activation energy
err_R	Error in outlet concentration of A for reactor
err_{FT}	Error in molar fraction at the liquid outlet for FT
ε	Clipping hyperparameter
$\epsilon_1, \epsilon_2, \epsilon_3$	Thresholds for clauses in Algorithm 4.1
η	Time-independent continuous design and control variables
η^L, η^U	Lower and upper bounds for time-independent continuous design variables
f_a	Algebraic equality constraints
f_d	Differential equality constraints
f_0	Initial conditions for the system
$feed_N$	Feed stage
F^L	Inlet molar flowrate to DC
F	Total molar flowrate
F_{cool}	Coolant flowrate
F_{in}	Inlet flowrate to Mixer
F_{rec}	Recycle flowrate
F_0	Inlet molar flowrate to CSTR
Φ	Logical constraints
g	Inequality constraints
g_{cons}	Constraint limit
γ	Integer design variables

$\tilde{\gamma}$	Discount factor
$\mathbf{h}$	Equality constraints
H_2O	Water
H_{CSTR}	Height of CSTR
i	Step index
I	Maximum number of steps in an episode
I_B	Integral error molar fraction difference at the bottoms DC
I_D	Integral error molar fraction difference at the distillate DC
I_{FT}	Integral error molar fraction difference in FT
I_R	Integral error concentration difference in reactor
$inter_{dist}$	Intermediate distillate flowrate
ISE_D	ISE for the distillates
ISE_B	ISE for bottoms
ISE_{FT}	ISE for FT
ISE_R	ISE for reactor
k	Reaction rate constant
$\tilde{k}$	Index for various constraints associated with each disjunctive decision
k_0	Kinetic rate constant
$\kappa(t)$	Continuous operating conditions
κ^L, κ^U	Lower and upper bounds for continuous operating conditions
K_C^B	Proportional gain PI controller in bottoms
K_C^D	Proportional gain PI controller in distillate
K_C^{FT}	Proportional gain PI controller in FT
K_C^R	Proportional gain PI controller in reactor

$\mathbf{l}$	Logits
l_{mask}	Masked logit
L^{APFR}	Length APFR
L^{PFR}	Length PFR
L_{FT}	Liquid outlet stream of FT
$\mathbf{L}_{FT}$	Liquid outlet vector
L_i	Liquid flow for each tray
$\widehat{Label}$	Predicted <i>Label</i>
$\widehat{Label}_{DC}$	Predicted <i>Label</i> for DC
$\widehat{Label}_{FT}$	Predicted <i>Label</i> for FT
$\widehat{Label}_R$	Predicted <i>Label</i> for reactor
$\ln(\widehat{DM})$	Predicted natural logarithm of a dynamic metric
λ	GAE parameter
m, n	Dimensions of $\mathbf{M}_{UO}$
M^L	Molar holdup
M_C	Molar holdup in the condenser
M_R	Molar holdup in the reboiler
$\mathbf{M}_{Resp}$	Matrix storing the response for each $\mathbf{z}^m$ realization
$\mathbf{M}_{UO}$	Data matrix for UO
mv_i	Masking vector at step i
μ_θ	Mean of Gaussian distribution parameterized by the policy
$\tilde{\mu}$	Decision variable to choose between FT or DC
N	Number of stages
N_{min}	Minimum number of stages

$N_{reactors}$	Total number of reactors
n_{x_d}	Dimension of discrete design variables
n_{x_c}	Dimension of continuous design variables
n_{χ}	Dimension of intermediate continuous variables
n_{ϕ}	Dimension of Boolean variables
n_{η}	Dimension of time independent continuous design and control variables
n_{γ}	Dimension of integer design variables
n_{κ}	Dimension of continuous operating conditions variables
n_u	Dimension of controlled variables
NN_c	Neural network classifier
NN_{DP}	Neural network dynamic performance metric regressor
NN_{EP}	Neural network endpoint regressor
$\mathcal{N}$	Gaussian distribution
ν	Binary decision variable for separation sequence
$\mathcal{O}$	Observation space
o_i	Observation at step i
Ω	Feasible region for continuous and discrete variables
ω_{DME}	Molar fraction of DME
ω_{H_2O}	Molar fraction of Water
ω_{MeOH}	Molar fraction of MeOH
ω_{PG}	Molar fraction of PG
ω_{PO}	Molar fraction of PO
$P_{\tilde{q}}$	Set of constraints or logical propositions involved in disjunction $\tilde{q}$
$Press$	Pressure

π_{θ}	Policy
Ψ	MIDO objective function
ψ	MINLP objective function
Ψ_{SB}	Objective function for <i>Scenario B</i>
φ	Boolean variable
ϖ_i	Process operating variables at step i
$\tilde{Q}$	Total number of disjunctive decisions considered in the problem
q	Volumetric flowrate
$\tilde{q}$	Index for $\tilde{Q}$
Q_R	Rate of heat transfer in the reactor
q_{in}	Inlet volumetric flowrate
R	Reflux flow rate
r_A	Rate of reaction
r_b	b -individual sub-rewards
r_{CC}	Capital cost sub-reward
r_{CV}	Classifier value sub-reward
r_{DC}	Dynamic cost sub-reward
r_{FC}	Final constraint sub-reward
r_i	Reward at step i
$reflux_N$	Reflux stage
rr	Recycle ratio
R_{cte}	Gas constant
rp	Set that contains reactant and products
R_{ss}	Reflux flow in steady-state

ρ	Liquid density
ρ_{cool}	Coolant density
$q_{i,rp}$	Chemical components tracker at step i
s_i	State at step i
σ_θ	Standard deviation of Gaussian distribution for parameterized by the policy
t	Time
T	Temperature
$\mathbf{T}$	Temperature vector
t_0	Initial time
T_0	Feed temperature
T_C	Operating temperature of Cooler
T_C^R	Coolant temperature in the reactor
T_{Cnom}^R	Nominal coolant temperature
T_{cool}	Coolant temperature
T_{cool_0}	Feed coolant temperature
t_f	Final time
T_{HEX}	Operating temperature of HEX
T_{in}	Inlet temperature
T_{rec}	Recycle temperature
τ_I^B	Time integral PI controller in bottoms
τ_I^D	Time integral PI controller in distillate
τ_I^{FT}	Time integral PI controller in FT
τ_I^R	Time integral PI controller in reactor

θ	Policy parameters
θ_{old}	Old policy parameters
$\boldsymbol{\vartheta}(t)$	Auxiliary vector representing deviations between g_{cons} and g
U	Heat transfer coefficient
$\mathbf{u}(t)$	Manipulated control variables
$\mathbf{u}^L, \mathbf{u}^U$	Lower and upper bounds for manipulated control variables
V	Vapor boilup flowrate in DC
$\tilde{V}$	Value function
V^L	Liquid reaction volume in FT
$\mathbf{V}^L$	Liquid reaction volume vector
V_{cool}	Coolant volume
V_{CSTR}	Volume of CSTR
V_{cons}^L	Liquid reaction volume constraint
V_{DC}	Volume of DC
$V_{DC}^{min}, V_{DC}^{max}$	Minimum-and-maximum volumes of DC
V_{FT}	Vapor outlet stream of FT
V_{FT}^{SS}	Vapor outlet of FT steady-state
V_{in}^L	Initial reaction volume
$V_{in_{min}}^L, V_{in_{max}}^L$	Minimum-and-maximum reaction volumes
$\dot{V}_{in}$	Inlet volumetric flowrate
$\dot{V}_{out}$	Outlet volumetric flowrate
$\dot{V}_{out_{nom}}$	Nominal value for outlet volumetric flowrate
V_R	Volume of reactor (generic can be CSTR or PFR)
V_R^{min}, V_R^{max}	Minimum-and-maximum volumes of reactor

V_{ss}	Vapor boilup in steady-state
vf	Vapor/feed fraction
$\mathbf{w}(t)$	System output variables
$\mathbf{x}(t)$	State variables of the system as a function of time
$\dot{\mathbf{x}}(t)$	Time derivatives of the state variables
x_A	Molar conversion of reactant A
x_B	Liquid molar fraction of B in the bottoms
$\mathbf{x}_B$	Liquid molar fraction of B in the bottoms vector
x_B^{sp}	Bottoms set-point
$\mathbf{x}_c$	Continuous design variables MINLP
$\mathbf{x}_c^L, \mathbf{x}_c^U$	Lower and upper bounds for continuous design variables
$\mathbf{x}_d$	Discrete design variables MINLP
$\mathbf{x}_d^L, \mathbf{x}_d^U$	Lower and upper bounds for discrete design variables
x_D	Liquid molar fraction of B in the distillate
x_D^{sp}	Distillate set-point
$\mathbf{x}_D$	Liquid molar fraction of B in the distillate vector
x_i	Liquid fractions for each tray
x_L	Molar fraction of B at the liquid outlet for FT
$\mathbf{x}_L$	Molar fraction of B at the liquid outlet for FT vector
x_L^{sp}	Molar fraction set-point of B at the liquid outlet for FT
$\xi_i(\theta)$	Probability ratio between the new policy and the old policy at step i
$\mathbf{y}(t)$	Time-dependent output variables produced by the mechanistic models
$y(t_f)$	Endpoint value
$\widehat{y(t_f)}$	Predicted endpoint value

y_B	Vapor fraction at the bottoms
y_i	Vapor fractions for each tray
y_{sp}	Set-point value of the output variable $\mathbf{y}(t)$
y_V	Molar fraction of B at the vapor outlet for FT
z	Liquid fraction at the feed
z_A	Molar fraction of A
z_B	Molar fraction of B
z_C	Molar fraction of C
$\mathbf{z}^m$	Instance of $\mathbf{M}_{UO}$

Chapter 1

Introduction

Optimal process integration, and in particular, integration of process design and control, is a major topic within chemical engineering, attracting significant research attention (Yuan et al., 2012; Pistikopoulos and Diangelakis, 2016; Rafiei and Ricardez-Sandoval, 2020a). The aim is to identify optimal process design and control variables for a chemical process flowsheet (CPF) that ensure both the economic viability and dynamic feasibility of the plant's operation. However, the complexities involved in combining design and control have driven the search for more efficient solution methods (Seferlis and Georgiadis, 2004; Iftakher et al., 2022).

CPFs are typically designed to operate at economically optimal that can result in a feasible plant operation at steady state. Consequently, extensive research has been devoted to the design and optimization of CPFs under steady-state conditions. Flowsheet optimization is generally considered a two-step process: the optimal synthesis problem is addressed first followed by the optimal equipment design problem. The synthesis optimization problem focuses on determining the arrangement of unit operations (UOs) to produce a desired product from specified reactants and products. In contrast, the design optimization problem aims to establish the optimal design and operating conditions of the corresponding units, including both discrete and continuous variables, for a given flowsheet (Liñán et al., 2024). Usually, these steps are solved separately and following a set of rules derived from a combination of process data, and the insights and experiences of experts; this approach is referred to as the heuristic methodology. Although the designs generated through this method may be reliable, in many instances, the information available may be inaccurate or outdated concerning standards and quality parameters (d'Anterrockes and Gani, 2005).

With the development of advanced optimization and computational tools, CPF design underwent through a significant improvement, enabling the integration of process synthesis and process design decisions (Bertran et al., 2016). With this integration, it is possible to define the CPF design problem as a single optimization problem, giving way to model-based optimization methods. The primary challenge in this approach lies in defining the superstructure, a crucial element of the problem formulation that represents all possible arrangements of UOs considered in the analysis. The success of obtaining an optimal CPF is heavily reliant on the initial formulation of the superstructure, which is often influenced by the experience and expertise of the user (Brüggemann and Wolfgang, 2004). The superstructure often includes a comprehensive network of interconnections between various UOs, along with decision variables and structural design parameters. This detailed formulation typically

results in a mixed-integer nonlinear programming (MINLP) problem (Grossmann et al., 2005). Solving an MINLP is particularly challenging due to the combinatorial nature of discrete decision variables, the presence of disjunctive decisions that create logical constraints, the nonlinearity of equations governing process behavior, and potential non-convexities, which can lead to multiple local optima. As a result, advanced optimization algorithms and significant computational resources are often required to find feasible or near-optimal solutions within a reasonable time.

As an alternative to the traditional methodologies, a new approach has recently emerged that integrates both heuristic and mathematical methods. This hybrid, or model-free optimization method, combines the rules of heuristic techniques with mathematical methods for synthesizing and designing CPFs. The versatility of the hybrid methods allows them to address both equation-driven and data-driven problems effectively. The growing interest in hybrid methods is closely linked to advancements in Machine Learning (ML) and Artificial Intelligence (AI). Among these, Reinforcement Learning (RL) is particularly notable for CPF design due to its ability to make interrelated sequential decisions while optimizing complex processes. Note that RL is considered a model-free optimization method as it does not rely on explicit mathematical models in the RL environment. Instead, it learns optimal strategies from data, which can originate from different sources, such as, advanced simulators, mathematical models, or databases. Various studies utilizing RL are documented in the literature, with a more detailed review provided in Chapter 2. Despite the works in this area, there are still research avenues to explore. For instance, the application of advanced techniques such as masking – associated with the incorporation of expert input or design rules similar to those used in heuristic methods – remains underexplored. To the authors' knowledge, integrating masking into a hybrid RL agent and providing a formal explanation of its functionality represent major gaps in the literature.

Once the CPF is defined at steady-state conditions, the next step is to conduct a controllability analysis, which aims to assess the dynamic operability of the system in closed-loop. Despite its effectiveness and widespread application, the dynamic nature of real-world environments often causes deviations from steady-state operation, which can adversely affect both the economics and sustainability of the plant (Andrés-Martínez and Ricardez-Sandoval, 2022). In extreme cases, a design that is optimal under steady-state conditions may become suboptimal or even infeasible when subjected to transient disturbances or model uncertainty (Mohideen et al., 1996; Schweiger and Floudas, 1998). Therefore, it is essential to adopt an integrated approach to design and control in order to identify feasible plant designs that consistently achieve design objectives during both steady-state and transient phases, while accounting for external disturbances and model uncertainties.

The integration of process design and control frequently involves balancing conflicting objectives between process economics, sustainability, and dynamic operability. For instance, there is often a trade-off between selecting smaller units to reduce capital costs and opting for larger units to enhance dynamic feasibility. Addressing this trade-off can be highly complex, especially in the context of CPF design, where decisions encompass both continuous variables and integer or disjunctive choices. These decisions occur at various levels: for instance, at the UOs level, decisions might involve selecting reactor types or incorporating recycle streams, while at the equipment level, choices may include determining the number of stages in a distillation column or the placement of feed trays. Additionally, the modeling of UOs typically involves differential-algebraic equations (DAE), further complicating the design process (Palma-Flores and Ricardez-Sandoval, 2023). In contrast to steady-state CPF design and optimization, incorporating the system's dynamics requires integrating DAEs as constraints into the optimization formulation. This results in intricate problems commonly known as mixed-integer dynamic optimization (MIDO) problems. In addition, stochastic events taking place during operation, e.g. time-varying model uncertainty, can increase the complexity of those problems (Bahakim and Ricardez-Sandoval, 2014; Koller et al., 2018). Consequently, various model-based optimization methods have been proposed to address the integrated design and control problem. Chapter 2 reviews these methods as presented in the literature. Despite these efforts, current methodologies for process flowsheet design are often limited to case studies involving a small number of UOs (Chawankul et al., 2005; Hamid et al., 2010; Ricardez-Sandoval, 2012; Sánchez-Sánchez and Ricardez-Sandoval, 2013). These cases are frequently modeled using non-rigorous methods due to high computational costs and mathematical complexity. Furthermore, case studies that use rigorous models and involve additional challenging tasks, e.g. safety considerations, are typically confined to specific feasible regions (Bansal et al., 2002; Bernal et al., 2018; Tian and Pistikopoulos, 2019; Tian et al., 2021), making it either prohibitively expensive or impractical to scale or interpolate these models to other regions.

Instead of relying on rigorous models, surrogate models are introduced as data-driven tools capable to approximate expensive rigorous models by correlating relevant input and output data. In addition to reducing the complexity of the optimization problem, these models can provide computational advantages over rigorous models. However, the overall computational cost is highly dependent on the implementation strategy; for instance, frequent online updates of surrogate models may become computationally demanding. On the other hand, rigorous models often face challenges such as numerical divergence or excessive CPU time requirements, particularly when applied to large-

scale, complex systems. With the increasing popularity of ML in the scientific community, Neural Networks (NNs) have become the preferred surrogate model (Schweidtmann and Mitsos, 2019). Although some studies have investigated the application of NNs as surrogate models, there is a gap in the literature regarding the use of these models in conjunction with RL for the integrated design and control problem. Additionally, there are very few studies in the literature that address the integrated design and control problem using RL (Sachio et al., 2022; Mendiola-Rodriguez and Ricardez-Sandoval, 2022). As previously mentioned, the majority of existing works focus on steady-state CPF design. Therefore, solving the integrated problem using NNs as surrogate models embedded within a RL environment represents a novel and intriguing research avenue that has not yet been explored.

1.1 Research objectives and contributions

The growing interest in model-free optimization methods, particularly RL, has greatly propelled advancements in CPF design. Despite the research conducted, there are still important gaps to fill, especially in the integrated design and control problem. Accordingly, this thesis aims to advance the subject by presenting a novel methodology for steady-state CPF systems and, more notably, a new framework to tackle the problem under transient operation. The current research is centered around the following specific objectives:

- Present a new methodology capable of generating, designing, and optimizing a steady-state CPF, starting from an inlet flow rate and a set of UOs with rigorous mathematical models, while adhering to design and operational constraints.
- Develop a methodology capable of linking non-adjacent UOs within the process flowsheet, thereby enabling interactions between them.
- Present a comprehensive methodology for the simultaneous generation, design, and control of CPFs in transient states, starting from an inlet flow rate and a set of UOs. The approach must ensure compliance with design constraints, effectively manage system disturbances, and optimize flowsheets for economic viability.
- Develop a methodology that captures the dynamic behavior of UOs using a series of representative data to generate surrogate models for these units.

As outlined above, the contributions of this thesis encompass not only the design and optimization of steady-state CPF but also in the field of integrated design and control of CPF. Regarding steady-state CPF design, this thesis makes the following contributions:

- This research introduces a masked hybrid RL agent that, despite employing a masking process only for the discrete component, simultaneously handles discrete and continuous actions. This dual-action approach, not previously explored in the literature, adds complexity to the methodology as the effect of masking is not solely reliant on the discrete part but also encompasses the continuous part. Masking is regarded as the incorporation of an expert input, process knowledge, or design (heuristic) rules that allow the exclusion of actions from the agent's decision spectrum.
- An advanced RL environment is presented, compatible with commercial chemical engineering simulators. This environment uses rigorous UO models and offers flexibility to interact with various operations offered by the commercial simulator suite.
- The study provides an in-depth explanation of how RL generates flowsheets and learns complex structural relationships between interconnected UOs, such as recycle streams, using the masking technique.

In addition to these contributions a comparative analysis of discrete and hybrid masked agents is performed highlighting their respective advantages and limitations. On the other hand, this thesis advances the field of integrated design and control of CPF with the following contributions:

- A methodology is developed to model UOs dynamically using representative data to create surrogate models that effectively capture output variables, dynamic performance metrics, and operational constraints. This approach is designed to be applicable across various UOs.
- This study proposes a new step-reward system, flexible enough to be adapted to various case studies, that effectively accounts for design and dynamic costs, as well as for constraint violations.
- A novel RL framework is proposed, where the agent interacts with an environment primarily composed of surrogate models. This approach enables the generation, design, and control of CPFs while ensuring compliance with design constraints and producing economically viable flowsheets. Unlike current state-of-the-art methods, this framework is not limited to a specific range of UOs. As long as a dynamic model of an UO is available, its transient behavior can be adequately approximated using surrogate models, broadening the applicability and flexibility of the methodology.

1.2 Structure of the Thesis

A description of the structure of this thesis is presented next:

Chapter 2 presents a comprehensive literature review relevant to this thesis. First, an overview of RL, its agents and applications in chemical engineering is provided, paying special attention to the Proximal Policy Optimization (PPO) agent. Following that, the generation, design, and control of CPFs in both steady and transient states problems are posed, discussing relevant methodologies for the solution of these problems. The gaps in knowledge that motivate this research are discussed at the end of this chapter.

Chapter 3 introduces a novel framework using masked RL agents for the design and optimization of CPFs at steady state. This framework includes a comparative study where the performance of these agents is evaluated across various case studies, two of which utilize advanced chemical simulation suites within the RL environment. To the author's knowledge, the integration of masking within a hybrid RL agent, as well as the use of advanced simulation suites involving multiple UOs, has not been previously explored. The work presented in this chapter was showcased at the 12th IFAC Symposium on Advanced Control of Chemical Processes (ADCHEM 2024) (Reynoso-Donzelli and Ricardez-Sandoval, 2024c), published in the AIChE Journal (Reynoso-Donzelli and Ricardez-Sandoval, 2024d), and presented at the AIChE Annual Meeting 2024 (Reynoso-Donzelli and Ricardez-Sandoval, 2024a). Permission from the publisher has been granted to include this content in the thesis.

Chapter 4 introduces a novel RL approach for generating, designing, and controlling CPFs in the transient domain. CPFs are created directly from an inlet stream, eliminating the need for predefined UOs arrangements. Moreover, the framework leverages surrogate models – specifically NNs – to approximate the dynamic behavior of UOs in closed-loop UO. This approach accelerates the RL agent's learning process and reduces dependence on mechanistic dynamic models. The efficacy of this framework is demonstrated through its application to three distinct case studies. To the author's knowledge, integrating RL with surrogate models in this context has not been previously attempted. The outcomes of this work have been submitted for journal publication (Reynoso-Donzelli and Ricardez-Sandoval, 2024e) and presented at the AIChE Annual Meeting 2024 (Reynoso-Donzelli and Ricardez-Sandoval, 2024b).

Chapter 5 presents the concluding remarks from this research on CPF design in both steady and transient states. Additionally, recommendations for potential future work are provided.

Chapter 2

Literature Review

Chemical process flowsheets (CPFs) are essential tools in chemical engineering offering a visual representation of chemical processes by illustrating the arrangement of UOs, their stream connections, and the materials involved. Designing and optimizing CPFs in both steady and dynamic states remains an active area of research, given its importance for improving process efficiency, minimizing resource consumption, enhancing safety, and ensuring process flexibility and sustainability. CPFs are typically solved through model-based optimization, where mathematical models represent the system and guide decision-making process. However, these methods can face challenges such as long computational times, difficulty in handling non-linearities and uncertainties, and limited scalability for complex or large systems. Model-free methods, in particular RL strategies, have emerged as a promising approach for addressing optimization problems in chemical engineering. Its ability to learn and adapt from dynamic environments without relying on detailed process models offers a compelling alternative to traditional optimization methods, particularly in handling complex systems. Section 2.1 provides an overview of RL and its applications in chemical engineering. In this section special emphasis will be placed on PPO algorithm, its structure and various modifications; as this agent, in its different variants, was employed throughout this thesis. RL is discussed first to establish its potential as a novel optimization approach, providing the foundation for understanding its application to process flowsheet synthesis and design. Section 2.2 examines the steady-state design and optimization of CPFs, reviewing various methods for addressing this problem, with particular emphasis on RL-based approaches. Section 2.3 extends the study of CPF design by considering transient processes, thereby incorporating control decisions into the design. This section highlights methods used to address this challenge, with a particular focus on RL-driven solutions. Each subsection concludes by summarizing the key findings and identifying the gaps revealed in the literature for each topic. A summary of this chapter is provided at the end.

2.1 Reinforcement Learning and its Applications in Chemical Engineering

In recent years, RL has garnered increasing interest within the field of chemical engineering due to its potential to enhance process optimization, control strategies, and decision-making processes. As a subset of ML, RL provides a robust framework for developing algorithms that learn optimal behaviors through interactions with their environment. RL is particularly valuable in chemical engineering, where

complex, dynamic systems require sophisticated control mechanisms and decision-making tools. The concept of RL and its basic constituents are presented next.

2.1.1 Reinforcement Learning: An Overview

RL is formally defined as a type of ML where an agent learns to make decisions by interacting with an environment and receiving feedback in the form of rewards or penalties. The agent's objective is to maximize the cumulative reward over time by learning the most effective actions to take in various states of the environment (Sutton and Barto, 2014). This approach contrasts with supervised learning, where models are trained on predefined datasets with explicit labels. Key components that constitute the RL framework include the agent, which is the learner or decision-maker; the environment, which encompasses all external factors with which the agent interacts; the policy (π_θ), which defines the agent's behavior by mapping states to actions; and the value function ($\tilde{V}$), which estimates the long-term return or expected reward from a given state or action.

A complete interaction between the agent and the environment is termed a step (i). During this step, the agent performs an action (a_i), which influences the environment and may alter its state. In response, the environment provides an observation (o_i) containing sufficient information to describe the state without returning all the variables that define it, and a reward (r_i), which serves as feedback on the immediate value of the action taken. The sequence of steps from an initial state to a terminal state or until a predefined condition is met constitutes an episode. The relationship between all these elements can be schematically represented as in Figure 2.1:

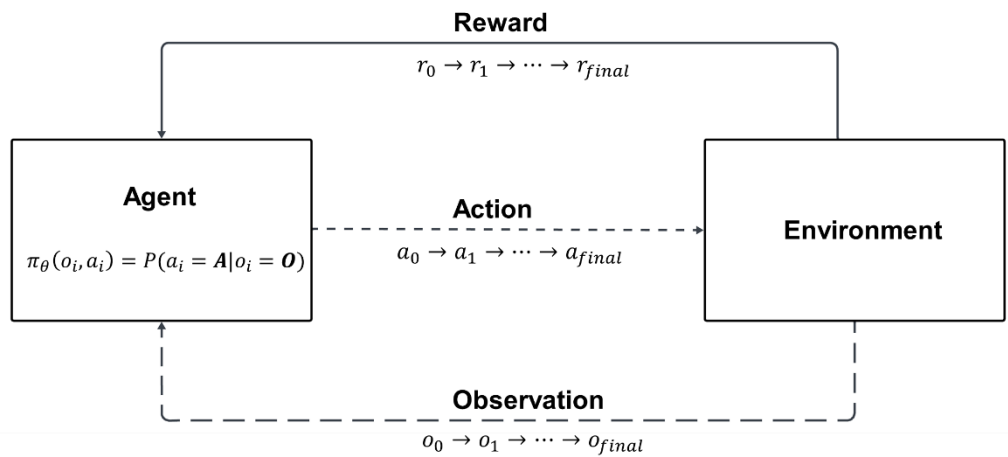


Figure 2.1: RL diagram

Figure 2.1 illustrates the fundamental interaction between an agent and its environment in a RL framework. The agent observes the environment's state through a series of observations, $o_0 \rightarrow o_1 \rightarrow \dots \rightarrow o_{final}$, and based on these observations, it selects and performs a sequence of actions, $a_0 \rightarrow a_1 \rightarrow \dots \rightarrow a_{final}$. The policy of the agent, represented by $\pi_{\theta}(o_i, a_i) = P(a_i = \mathbf{A} | o_i = \mathbf{O})$, defines the probability of selecting step-action a_i , from the action space $\mathbf{A}$, given the current step-observation o_i , found in the observation space $\mathbf{O}$. The environment reacts to the agent's actions by updating its state and providing new observations and rewards, which reflect the immediate value of the agent's actions. The step-reward, r_i , serves as feedback, allowing the agent to adjust its behavior to maximize future rewards. The cycle of observations, actions, and rewards forms the foundation of the learning process. The agent interacts iteratively with the environment, where it observes the current state, takes an action based on its policy, and receives feedback in the form of rewards or penalties. This feedback informs the agent about the effectiveness of its actions. Learning occurs through a framework, where the agent continuously refines its policy – a mapping of states to actions – based on the accumulated feedback. As the agent explores various actions, it evaluates the outcomes and adjusts its approach to maximize cumulative rewards. This iterative process fosters an adaptive learning mechanism, allowing the agent to optimize its performance over time. The learning continues until a terminal condition or final state is reached, denoted as subscript *final* in this context.

In RL, agents are generally classified into several categories based on their learning strategies and environments. This classification is presented next.

1. Model-Free Agents: These agents learn directly from their interactions with the environment without constructing an explicit model of the environment's dynamics. Instead, they rely on trial-and-error methods, where they explore various actions and observe the resulting outcomes to infer knowledge about the environment. This experiential learning allows them to improve their decision-making policies based on the feedback received from their actions over time. This can be further partitioned as follows:
 - a. Value-Based: These agents focus on estimating the value of actions or states, which helps inform their decision-making process.
 - i. Off-Policy: Agents like Deep Q-Network (DQN) (Mnih et al., 2013), Double DQN (DDQN) (van Hasselt et al., 2015), and Q-Learning (Jang et al., 2019) learn from actions taken by other policies or behaviors, enabling them to leverage a wider range of experiences. This flexibility allows them to update

their value estimates based on both historical actions and the current policy, facilitating more efficient learning.

- ii. On-Policy: Agents like SARSA (State-Action-Reward-State-Action) (Jang et al., 2019) learn from actions taken under the current policy. This approach means that the learning is directly tied to the policy being executed, which helps the agent refine its strategy in real-time based on the rewards received from its own actions.
- b. Policy-Based: These agents focus on learning a policy that directly maps states to actions, optimizing their performance in a more intuitive manner.
 - i. Off-Policy: Agents like Deep Deterministic Policy Gradient (DDPG) (Lillicrap et al., 2019) use past experiences to update their policies. By learning from historical actions and corresponding outcomes, these agents can explore a wider range of potential actions without being limited to the current policy, leading to improved policy stability and performance.
 - ii. On-Policy: Agents like A2C (Advantage Actor-Critic) (Mnih et al., 2016) and Proximal Policy Optimizer (PPO) (Schulman et al., 2017) update their policies based on the current policy. This real-time adjustment allows them to make incremental improvements, directly enhancing their decision-making capabilities as they gather new information from their environment.
- 2. Model-Based: These agents construct a model of the environment to predict future states based on current actions. This model can take various forms (e.g., NNs, mechanistic models), which capture the underlying dynamics of the environment and allow the agent to simulate potential outcomes before acting. By leveraging this predictive capability, model-based agents can optimize their actions more effectively by anticipating the consequences of different strategies.
 - a. Policy-Based: Model-based agents like Model Predictive Control (MPC) (García et al., 1989) utilize a model of the environment to optimize actions. By forecasting future states, they can plan a sequence of actions that maximize performance while considering constraints, leading to improved decision-making in dynamic environments.

2.1.2 Proximal Policy Optimizer (PPO)

Each of the methods discussed above has its own advantages and disadvantages. Model-Free Agents are typically simpler to implement and do not require an understanding of the underlying dynamics of the environment. However, they often require extensive exploration and can be inefficient in terms of learning, as they rely heavily on trial-and-error methods. Value-based agents are effective at estimating the value of actions, yet they may struggle with large state spaces and can suffer from convergence issues. Policy-based agents, on the other hand, can directly optimize policies and handle continuous action spaces more effectively, though they may also face challenges with sample efficiency and stability during training. Among the various RL algorithms, PPO has gained significant attention due to its ability to balance computational efficiency with policy stability, making it particularly well-suited for complex environments encountered in process systems engineering (Yu et al., 2022). Additionally, PPO minimizes the risk of unstable updates, a common challenge in RL, by using a clipped objective function that restricts the extent of policy changes between updates. This characteristic enhances training stability, ensuring consistent performance (Schulman et al., 2017). Moreover, its flexibility allows it to deal with both discrete and continuous action spaces. Likewise, recent advancements have demonstrated that PPO can also handle hybrid action spaces (Fan et al., 2019), which involve a combination of discrete and continuous actions.

PPO strikes a balance between computational efficiency and policy stability by employing a clipped surrogate objective function. This design ensures that updates to the policy are controlled, preventing excessive deviations that could destabilize training. The objective function for PPO is defined as follows:

$$L^{PPO}(\theta) = \mathbb{E}[\min(\xi_i(\theta) \hat{A}_i, \text{clip}(\xi_i(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_i)] \quad (2.1)$$

The probability ratio $\xi_i(\theta)$ is a critical component in the PPO objective function, as it quantifies how the updated policy (π_θ) compares to the old policy ($\pi_{\theta_{old}}$) with respect to assigning probabilities to specific actions. This term is defined as follows:

$$\xi_i(\theta) = \frac{\pi_\theta(a_i|s_i)}{\pi_{\theta_{old}}(a_i|s_i)} \quad (2.1a)$$

The probability ratio can be described as follows:

- When $\xi_i(\theta) > 1$, The new policy assigns a higher probability to the action (a_i) in state (s_i) than the old policy did. This typically means that the action is deemed more favorable under the updated policy, perhaps due to higher associated rewards or advantages.
- When $\xi_i(\theta) < 1$, the new policy assigns a lower probability to the action (a_i). This implies that the action is less preferred in the new policy, possibly due to lower rewards or advantages.

By directly comparing these probabilities, PPO can evaluate whether the updated policy is moving in a desirable direction in terms of action preferences. The probability ratio is multiplied by the advantage function $\hat{A}_i$, which measures how much better or worse an action is compared to the policy's average performance. The sign of $\hat{A}_i$ determines the desired direction of change, i.e.,

- If $\hat{A}_i > 0$, the goal is to increase the probability of the action, so PPO pushes $\xi_i(\theta)$ toward values greater than 1.
- If $\hat{A}_i < 0$, the goal is to decrease the probability of the action, so PPO encourages $\xi_i(\theta)$ toward values less than 1.

The ratio thus adjusts the gradient step based on how the policy change aligns with the advantage. In standard policy gradient methods, large updates can cause $\xi_i(\theta)$ to deviate significantly from 1. This can lead to drastic changes in the policy, destabilizing training and reducing sample efficiency.

PPO addresses this issue with its clipping mechanism, which ensures that $\xi_i(\theta)$ remains within a bounded range $[1 - \epsilon, 1 + \epsilon]$. This prevents the policy from deviating too far from the old policy during a single update. Without this constraint, large updates could lead to poor exploration, overfitting, or policy collapse.

- If $\xi_i(\theta)$ remains within the range $[1 - \epsilon, 1 + \epsilon]$, the objective function remains unaltered.
- If $\xi_i(\theta)$ exceeds this range, the objective function "clips" the ratio to either $1 - \epsilon$ or $1 + \epsilon$ effectively capping the contribution of the advantage $\hat{A}_i$.

This mechanism ensures that even with large advantage values, the update remains stable. The hyperparameter ϵ controls how conservative or aggressive the updates are. The clipping hyperparameter ϵ takes typically values between 0.1 and 0.3. Higher values allow larger updates, which may accelerate learning but risk instability. While lower values enforce stricter constraints, favoring stability at the cost of slower convergence.

To compute more accurate and robust estimates of the advantage function, PPO uses Generalized Advantage Estimation (GAE). The advantage function is critical for determining how much better or worse an action is compared to the average policy performance at a given state. GAE is defined as follows:

$$\hat{A}_i = \sum_{l=0}^{\infty} (\tilde{\gamma}\lambda)^l \delta_{i+l} \quad (2.2)$$

Components of GAE include:

1. Temporal difference error (δ_i):

$$\delta_i = r_i + \tilde{\gamma}\tilde{V}(s_{i+1}) - \tilde{V}(s_i) \quad (2.2a)$$

The temporal difference error quantifies the difference between the reward received plus the discounted value of the next state, $\tilde{V}(s_{i+1})$, and the current state's value $\tilde{V}(s_i)$.

2. Discount factor ($\tilde{\gamma}$): Controls how much future rewards contribute to the current estimate. A higher $\tilde{\gamma}$ places more emphasis on long-term rewards. Typically ranges between 0.95 and 0.99
3. Smoothing factor (λ): Balances the trade-off between bias and variance. A smaller value of λ reduces variance by relying more on immediate temporal difference errors but introduces bias. While a larger λ aggregates information over multiple time steps, reducing bias but increasing variance. Typical values for λ range between 0.9 and 0.95.

GAE allows PPO to leverage a combination of advantages over multiple time steps, thus providing a more robust estimate of the expected return. This helps in reducing variance and improving the overall performance of the policy.

PPO demonstrates remarkable adaptability as it can be applied to both discrete and continuous action spaces due to its flexible policy parameterization. This adaptability arises from the ability of PPO to represent policies using different probabilistic distributions tailored to the type of action space. In discrete action spaces, the policy is typically modeled using a categorical distribution. This distribution assigns probabilities to each discrete action, enabling the agent to choose one action from a finite set. The categorical distribution is derived by applying a *SoftMax* function to the policy network's output logits. This ensures that the resulting probabilities add to the unity and represent a valid probability distribution. The policy outputs the probabilities as follows:

$$\pi_{\theta}(a_i|s_i) = \text{SoftMax}(s^T \theta) \quad (2.3)$$

For continuous action spaces, the policy is often represented by parameterized probability distributions such as Gaussian, multivariate Gaussian, or even Beta distributions, depending on the application. The most common approach involves modeling the policy as a Gaussian distribution ($\mathcal{N}$), where the mean (μ_{θ}) and standard deviation (σ_{θ}) learned by the agent. The policy is expressed as follows:

$$\pi_{\theta}(a_i|s_i) = \mathcal{N}(\mu_{\theta}(s), \sigma_{\theta}(s)) \quad (2.4)$$

This formulation allows the agent to sample actions continuously, capturing the uncertainty inherent in the decision-making process. Other distributions, like the Beta distribution, can be used when the action space is bounded within a specific range, offering additional flexibility for specialized tasks. In both discrete and continuous action spaces, the sampling process evolves as the agent learns. For discrete action spaces, the categorical distribution shifts to favor higher-probability actions as the agent improves its policy. Over time, less probable actions are deprioritized, sharpening the distribution around the optimal actions. In continuous action spaces, the standard deviation σ_{θ} of the Gaussian distribution tends to decrease as the agent gains confidence in its actions. This results in a slimmer Gaussian distribution, concentrating the samples closer to the mean μ_{θ} . This dynamic adjustment in both discrete and continuous spaces ensures that PPO effectively balances exploration and exploitation while converging towards optimal behavior.

Recent works have extended PPO to handle hybrid action spaces, where the environment requires both discrete and continuous actions (Fan et al., 2019). This is especially useful in robotic control tasks or games where some decisions are categorical (e.g., movement directions) and others are continuous (e.g., torque or speed). The hybrid policy structure can be represented as a mixture of a categorical distribution for the discrete actions and a Gaussian distribution for the continuous actions, combining both types in a single objective.

In this research, all versions of PPO – discrete, continuous, and hybrid – were considered. Specifically, in Chapter 3, both discrete and hybrid PPOs are employed for the generation, design and optimization of CPF. Notably, these agents incorporate a slight variation in the discrete component of their policies through the application of masking (Huang and Ontañón, 2022). This technique will be further explained in Chapter 3. Conversely, a continuous PPO is utilized in Chapter 4 to simultaneously generate, design, and control CPF. Some variations were applied to the structure of the agent to obtain

better exploration and results; these modifications will be explained in detail in the aforementioned chapter.

2.1.3 Applications of Reinforcement Learning in Chemical Engineering

Recently, RL has demonstrated its ability to tackle complex sequential dynamic decision-making problems at levels comparable to or surpassing human performance (Silver et al., 2018). These advancements have significantly impacted various research areas within chemical engineering, leading to the development of innovative methods for addressing complex challenges. RL is primarily designed for solving discrete-time optimization problems modeled as Markov Decision Processes and relies on the Bellman optimality equation, which is similar to the Hamilton–Jacobi–Bellman equation and Pontryagin’s maximum principle used in continuous state and action spaces (Gao and Schweidtmann, 2024). This foundation explains why early applications of RL in chemical engineering focused mainly on process control (Hoskins and Himmelblau, 1992) and continue to be prevalent (Nian et al., 2020; Bao et al., 2021; Alhazmi et al., 2022; Patel, 2023; Faria et al., 2024; Hassanpour et al., 2024; Wang and Ricardez-Sandoval, 2024; Agyeman et al., 2024). However, RL has also advanced other areas such as scheduling (Waschneck et al., 2018; Hubbs et al., 2020; Mowbray et al., 2022; Mousa et al., 2024; Rangel-Martinez and Ricardez-Sandoval, 2024), dynamic optimization (D. Machalek et al., 2020; Petsagkourakis et al., 2020; Boulesnane and Meshoul, 2021), and the design of CPF in both steady and transient domains. The subsequent sections of this chapter will discuss these advancements in CPF design in greater detail.

2.2 Chemical Process Flowsheet Design and Optimization

In the context of steady-state CPF design, the optimization problem is typically formulated as a MINLP (Grossmann, 1985; Trespalacios and Grossmann, 2014; Mencarelli et al., 2020; Chen et al., 2021). Equation 2.5 presents the mathematical formulation of the optimization problem at hand. The primary objective is to minimize an objective function, denoted as ψ , which may represent a combination of factors such as process economics, sustainability incentives, environmental constraints, and other operational considerations. Mathematically, ψ is defined as a function of three types of decision variables: 1) Continuous design variables, denoted as $\mathbf{x}_c \in \mathbb{R}^{n_c}$, englobe all the continuous decisions taken for the CPF design, e.g., equipment sizing, required purities, flow rates, among others. 2) Discrete design variables, denoted as $\mathbf{x}_d \in \mathbb{Z}^{n_d}$, represents all the discrete design variables in the problem such as the number of stages in a distillation column or the number of tubes in a shell and tube heat exchanger. 3) Intermediate variables, $\mathbf{x} \in \mathbb{R}^{n_x}$, denote the intermediate continuous variables calculated

internally by the chemical process simulator, e.g., material flows, compositions, and operating conditions. Additionally, $\boldsymbol{\varphi} \in \{True, False\}^{n_\varphi}$ represents the Boolean variables that appear in disjunctions, determining whether a specific constraint vector is enforced or not. This disjunctive formulation is commonly applied in CPF design to capture structural and logical decisions. Examples include determining the presence or absence of a UO, the inclusion or exclusion of a flow at a particular location, or the implementation or omission of a recycle stream at a specific point in the flowsheet.

$$\min_{\mathbf{x}_d, \mathbf{x}_c, \boldsymbol{\chi}} \psi(\mathbf{x}_d, \mathbf{x}_c, \boldsymbol{\chi}) \quad (2.5a)$$

$$s. t. \mathbf{h}(\mathbf{x}_d, \mathbf{x}_c, \boldsymbol{\chi}) = 0 \quad (2.5b)$$

$$\mathbf{g}(\mathbf{x}_d, \mathbf{x}_c, \boldsymbol{\chi}) \leq 0 \quad (2.5c)$$

$$\Phi(\boldsymbol{\varphi}) = True$$

$$\forall \tilde{k} \in P_{\tilde{q}} \left[\varphi_{\tilde{k}, \tilde{q}} \left[g_{\tilde{k}, \tilde{q}}(\mathbf{x}_d, \mathbf{x}_c, \boldsymbol{\chi}) \leq 0 \right] \right], \forall \tilde{q} \in \tilde{Q} \quad (2.5d)$$

$$[\mathbf{x}_d, \mathbf{x}_c, \boldsymbol{\chi}] \in \Omega \quad (2.5e)$$

$$\mathbf{x}_d^L \leq \mathbf{x}_d \leq \mathbf{x}_d^U, \mathbf{x}_c^L \leq \mathbf{x}_c \leq \mathbf{x}_c^U \quad (2.5f)$$

Following Equation 2.5, the equality constraints ($\mathbf{h}$) and inequality constraints ($\mathbf{g}$) encapsulate all the operational and design limitations associated with the UOs in the CPF. These constraints include restrictions such as the permissible operating pressure or temperature for a given UO, as well as process design targets like the desired level of reactant conversion or product purity. Moreover, Boolean variables $\varphi_{\tilde{k}, \tilde{q}}$ indicates if constraint $g_{\tilde{k}, \tilde{q}}$ is enforced or not. These constraints are related to the logical proposition $\Phi(\boldsymbol{\varphi}) = True$ through the logical operators (e.g., AND, OR, XOR, negation, implication, equivalence), which are defined based on the specific disjunctive decisions considered in the problem. Note that Boolean variables are represented as binary variables in the MINLP and the upper limit for n_φ is given by $n_\varphi = \sum_{\tilde{q} \in \tilde{Q}} |P_{\tilde{q}}|$. The index $\tilde{k}$ represents the various constraints associated with each disjunctive decision. Depending on the problem context, $\tilde{k}$ typically takes values from a set of integers representing specific constraints or conditions. The domain of $\tilde{k}$ can be defined as follows: $\tilde{k} \in \{1, 2, \dots, |P_{\tilde{q}}|\}$, $\forall \tilde{q} \in \tilde{Q}$, where $|P_{\tilde{q}}|$ is the cardinality of the subset $P_{\tilde{q}}$ corresponding to each disjunctive decision $\tilde{q}$. The index $\tilde{q}$ represents the different disjunctive decisions made in the optimization problem; the domain can be defined as $\tilde{q} \in \{1, 2, \dots, |\tilde{Q}|\}$ where $|\tilde{Q}|$ is the total number of disjunctive decisions considered in the problem. Moreover, $[\mathbf{x}_d, \mathbf{x}_c, \boldsymbol{\chi}] \in \Omega$ represents the constraints handled by the chemical simulation suite; these restrictions enclose all the mass and energy balances, thermodynamic models, chemical equilibria and reaction rates, and more restrictions considered internally by the

chemical simulator. Hence, the variables within the set Ω represent all the possible variable combinations that ensure a convergence of the model in the simulator (Liñán et al., 2024). Lower bounds (superscript L) and upper bounds (superscript U) for the discrete and continuous variables are also included in Equation 2.5.

Numerous studies have been conducted to address the design and optimization of CPF using model-based optimization methods (Caballero and Grossmann, 1999; Grossmann et al., 2005; Chen and Grossmann, 2017; Tian and Pistikopoulos, 2019; Liñán et al., 2020; Mencarelli et al., 2020; Ma et al., 2021; Tian et al., 2021; Bernal et al., 2022; Liñán and Ricardez-Sandoval, 2023; Stinchfield et al., 2024). However, these approaches face several inherent challenges. A key limitation is the reliance on a predefined superstructure, which often introduces bias due to the user's subjective definition of system boundaries and possible configurations. In the absence of a superstructure, the spatial arrangement of UOs within the CPF is typically fixed, thereby constraining the exploration of alternative flowsheet configurations and limiting design flexibility. Moreover, for any configuration, mechanistic models of the UOs must be explicitly programmed to enable access to all the necessary equations governing the process, adding significant complexity to the modeling effort. Furthermore, the limited availability of open-source software and the significant computational demands—especially for large-scale optimization problems—pose further challenges. These issues are compounded by the difficulty in integrating such models seamlessly with chemical simulation suites, reducing flexibility and increasing the complexity of solving CPF optimization problems. On the other hand, the application of global optimization methods offers a significant advantage. These techniques ensure global optimality, providing the assurance that the best possible solution is found within the defined problem space, thus improving the reliability and performance of CPF design and optimization. However, global optimization solvers can suffer from high computational costs, particularly in large-scale problems, leading to long CPU times and resource demands. Additionally, issues such as numerical instability, sensitivity to initial conditions, and convergence challenges can further complicate the optimization process, necessitating careful consideration of solver selection and problem formulation.

As discussed in Chapter 1, a new alternative to model-based optimization methods has emerged. This approach seeks to integrate the principles of heuristic methods into its formulation while employing mathematical techniques for the synthesis and design of CPFs. This method, also referred to as hybrid or model-free optimization, offers the flexibility to address both equation-driven and data-driven problems effectively. d'Anterrockes and Gani (d'Anterrockes and Gani, 2005) were among the first authors to explore this hybrid methodology, developing a framework for synthesizing, designing

and optimizing CPFs by employing group contribution—a strategy widely applied in molecular design. Other notable techniques such as deterministic stochastic-algorithms (Herrera Velázquez et al., 2022; Liñán et al., 2024), genetic algorithms (Contreras-Zarazúa et al., 2017, 2019) and Bayesian optimization (Chanona et al., 2021) have been proposed to synthesize and design CPF. However, the greatest interest in hybrid methods emerged with the increased studies and advancements in ML and AI techniques. RL is particularly notable as a framework for CPF design due to its capability to make interrelated sequential decisions while effectively optimizing complex processes. Table 2.1 provides an overview of various studies that have employed RL for the design and optimization of CPFs.

Table 2.1: Studies that have applied RL for CPF design and optimization

Authors	Contributions
(Midgley, 2020)	Proposed an agent capable of designing distillation trains to separate non-azeotropic mixtures. The RL agent used was a hybrid soft actor-critic that interacted with an environment coupled with the COCO (van Baten et al., 2023) simulation suite
(Khan and Lapkin, 2020, 2022; Seidenberg et al., 2023)	Introduced the use of hierarchical RL agents for searching optimal process routes. Framework applied to three separation processes. Hierarchical agents enabled enhanced interaction between the agent and the process, yielding superior results in comparison to baseline methods.
(Plathottam et al., 2021)	RL applied to optimize complex solvent extraction processes. Authors implemented a configurable learning environment that can effectively match or surpass the performance of traditional optimization methods, achieving comparable results in process design across various feed compositions.
(van Kalmthout et al., 2022)	Extended Midgley’s (Midgley, 2020) work by modifying the framework, allowing the interaction with the widely used ASPEN Plus® (Aspen Technology Inc, n.d.) simulation suite
(Göttl et al., 2022a, 2022b, 2024)	Authors have made significant contributions by developing "SynGameZero," a turn-based two-player game environment for CPF design, which utilizes a tree search RL algorithm from DeepMind to improve agent exploration. They advanced this approach by incorporating recycles and convolutional neural networks for more complex problems. Their latest work introduces a hierarchical structure of RL agents, enabling effective handling of both discrete and continuous variables. All their frameworks have been tested with short-cut models to represent UOs.
(Stops et al., 2023)	Represented CPF as graphs, enabling the incorporation of graph neural networks into their agent's architecture. Alongside this feature, a hybrid PPO agent was considered. Framework applied for methyl acetate production and purification.

(Gao et al., 2023)	Extended Stops et al. (Stops et al., 2023) work by leveraging transfer learning and linking the environment to a more rigorous simulation suite such as DWSIM (Medeiros, n.d.)
(Kim et al., 2023)	RL was applied to optimize Single Mixed Refrigerant cycles with the goal of minimizing refrigeration power. A DQN agent, interfaced with a process simulator and MATLAB® (The MathWorks Inc., 2022), demonstrated comparable performance to Genetic Algorithms while effectively managing complex interactions and nonlinearities.

Beyond RL, but still within the area of AI, a few works involving the design, completion or correction of CPF have been proposed using generative transformers (Vogel et al., 2023) and large language models (Schulze Balhorn et al., 2024). For those type of works, a considerable number of flowsheets – complete or incomplete –, as well as a tokenization method – similar to the one presented by d'Anterrosches and Gani (d'Anterrosches and Gani, 2005) – is required. To this regard, Mann et al. (Mann et al., 2023, 2024a, 2024b) have further advanced d'Anterrosches and Gani (d'Anterrosches and Gani, 2005) work by incorporating Machine Learning and Artificial Intelligence techniques.

As shown in Table 2.1, there are numerous studies that utilize RL for the design and optimization of CPFs. However, few of these studies consider a key tool for enhancing the agent's learning process: the incorporation of masking. Masking refers to a technique that dynamically restricts the agent's action space by disabling infeasible or suboptimal actions based on prior knowledge, expert rules, or contextual constraints. This approach improves learning efficiency, reduces exploration of irrelevant or impossible actions, and accelerates convergence toward optimal solutions. Huang and Ontañón (Huang and Ontañón, 2022) were the first to introduce the concept of masking in a RL context; they utilized a discrete PPO agent and, through various case studies, demonstrated the effectiveness of this technique compared to its standard version. In CPF design, masking could be particularly valuable for incorporating domain expertise and design rules, similar to the heuristic method, ensuring that the RL agent avoids configurations that violate physical or operational constraints. Göttl et al. (Göttl et al., 2022a, 2022b, 2024) and Seidenberg et al. (Seidenberg et al., 2023) also mentioned the use of the masking technique in their framework, however a detailed explanation of its functionality was not described in those works. Adding to that, the integration of a masking technique with a hybrid PPO approach remains unexplored in the literature. Chapter 3 will delve into the application of masking, offering a clearer understanding of how it influences the agent's learning process; as well as exploring the use of masking in hybrid agents, with a particular focus on hybrid PPO.

2.3 Simultaneous Generation, Design and Control of Chemical Process Flowsheet

Integration of design and control often involves the solution of a MIDO problem. This approach combines steady-state design and dynamic control considerations into a unified optimization step. Process design decisions involve equipment sizing and nominal operating conditions, while process control focuses on minimizing closed-loop variability and ensuring product quality by keeping key output variables within specified limits. Typically, the main objective in integrated CPF design and control problems is the minimization of an objective function (Ψ). This function usually combines economic or environmental costs combined with metrics that accounts for the system's controllability and dynamic variability. Equation 2.6 presents the primary formulation of the integrated design and control MIDO problem, posed as follows (Rafiei and Ricardez-Sandoval, 2020a):

$$\min_{\eta, \kappa(t), \gamma, u(t)} \Psi(x(t), \dot{x}(t), w(t), \eta, \kappa(t), \gamma, u(t), d(t), t) \quad (2.6a)$$

$$s. t. f_d(x(t), \dot{x}(t), w(t), \eta, \kappa(t), \gamma, u(t), d(t), t) = 0 \quad (2.6b)$$

$$f_a(x(t), w(t), \eta, \kappa(t), \gamma, u(t), d(t), t) = 0 \quad (2.6c)$$

$$f_0(x(t_0), \dot{x}(t_0), w(t_0), \eta, \kappa(t_0), \gamma, u(t_0), d(t_0), t_0) = 0 \quad (2.6d)$$

$$h(x(t), \dot{x}(t), w(t), \eta, \kappa(t), \gamma, u(t), d(t), t) = 0 \quad (2.6e)$$

$$g(x(t), \dot{x}(t), w(t), \eta, \kappa(t), \gamma, u(t), d(t), t) \leq 0$$

$$\Phi(\varphi) = True$$

$$\forall \tilde{k} \in P_{\tilde{q}} \left[\varphi_{\tilde{k}, \tilde{q}} \left[g_{\tilde{k}, \tilde{q}}(x_d, x_c, \chi) \leq 0 \right] \right], \forall \tilde{q} \in \tilde{Q} \quad (2.6f)$$

$$\eta \in [\eta^L, \eta^U] \quad (2.6g)$$

$$\kappa \in [\kappa^L, \kappa^U] \quad (2.6h)$$

$$u \in [u^L, u^U] \quad (2.6i)$$

$$\gamma \in \{\gamma_1, \dots, \gamma_N\} \quad (2.6j)$$

$$t \in (t_0, t_f] \quad (2.6k)$$

MIDO problems are typically constrained by DAEs, represented by f_d and f_a respectively, and their initial condition (f_0). In chemical engineering, particularly in process design and control, these functions include all continuity, mass and energy balances, thermodynamics, reaction kinetics, as well as any DAE representing the physical-chemical processes of the system. The problem can also be constrained by equality (h) and inequality (g) constraints, which may possess a differential-algebraic nature similar to f_d and f_a . The system states and their derivatives, represented as functions of time,

are denoted by $\mathbf{x}(t), \dot{\mathbf{x}}(t) \in \mathbb{R}^{n_x}$ while $\mathbf{w}(t) \in \mathbb{R}^{n_w}$ represents the system's output variables. Similar to Equation 2.5, Equation 2.6 also incorporates disjunctive decisions. The notation and structure used in both Equation 2.5d and Equation 2.6f are consistent, adhering to the same principles. In addition to addressing disjunctive decisions related to the inclusion or exclusion of a unit in the CPF, Equation 2.6f can also account for decisions such as the choice between a proportional-integral (PI) or proportional-integral-derivative (PID) control system. The vector $\boldsymbol{\eta} \in \mathbb{R}^{n_\eta}$ includes all time independent continuous design and control variables of the system, such as the diameter of a distillation column or the gain of a PI controller. In contrast, $\boldsymbol{\gamma} \in \mathbb{Z}^{n_\gamma}$ contains all integer design variables of the system, such as the number of stages in a distillation column, or the number of passes in a shell and tube heat exchanger. The time-dependent vector $\boldsymbol{\kappa}(t) \in \mathbb{R}^{n_\kappa}$ represents continuous operating conditions, which can vary over time, such as a change in the purity set-point of a distillate in a distillation column. The variables available for control are denoted by $\mathbf{u}(t) \in \mathbb{R}^{n_u}$, representing changes in manipulated variables used for process control, such as the reflux ratio in a distillation column or the jacket temperature in a reactor. In addition to the inequality and equality constraints, $\boldsymbol{\eta}$, $\boldsymbol{\kappa}(t)$, and $\mathbf{u}(t)$ are bounded by upper and lower limits, i.e., L and U , respectively. Furthermore, $\mathbf{d}(t)$ denotes the disturbances affecting the system over time (t), which are typically defined a priori and span from an initial time t_0 to a final integrating time t_f .

Model-based optimization methods are well-suited for solving problems like the one described in Equation 2.6. Over the past several years, there has been considerable attention directed towards the integration of design and control in CPF, prompting researchers to explore diverse approaches. Methods for integrating design and control are categorized in various ways in the literature. Most use steady-state models to estimate capital and operating costs. The main difference between them lies in how they assess the dynamic performance of the closed-loop system, focusing on how dynamic behavior affects costs. These methods fall into three categories (Ricardez-Sandoval et al., 2009a; Yuan et al., 2012):

- Controllability index-based approach

The controllability index-based approach integrates process design and control by using indicators like the relative gain array, condition number, and disturbance condition number to assess closed-loop performance. It aims to minimize capital and operating costs through steady-state models while meeting controllability objectives. However, these methods often treat controllability indicators as constraints or apply ad hoc weightings in the cost function without clear links to economic impact. Additionally, reliance on steady-state or linear dynamic models limits their accuracy to small deviations around

nominal conditions, potentially leading to suboptimal solutions, especially by overlooking product variability costs. Relevant works that fall within this category are (M.L. Luyben and Floudas, 1994; Luyben and Floudas, 1994; Papalexandri and Pistikopoulos, 1994; Schweiger and Floudas, 1998; Alhammadi and Romagnoli, 2004; Blanco and Bandoni, 2007).

- Dynamic optimization-based approach

To address the inherent limitations of the controllability index-based methodology, a variety of dynamic optimization techniques emerged in the literature. These methodologies shift focus from steady-state or matrix-based assessments to evaluating process variability through simulations utilizing comprehensive nonlinear dynamic models (Perkins and Walsh, 1996). However, the significant computational demands associated with these nonlinear simulations often confine analyses to predetermined time horizons, thereby imposing constraints on their applicability. The intricate nature of solving nonlinear dynamic optimization problems also limits their practicality in real-world industrial contexts. Furthermore, the influence of dynamic variability is not explicitly integrated into the overall economic cost function, unlike capital and operational costs. While dynamic optimization approaches provide valuable advantages in managing process nonlinearity and time-dependent disturbances, they face considerable challenges related to computational complexity, particularly regarding the absence of a well-defined economic cost framework for transient variability and insufficient explicit consideration of system's closed-loop stability. The dynamic optimization-based approach received significant attention developing and advancing various solution techniques: back-off approach and its various modifications such as power series expansions or uncertainty (Perkins, 1989; Narraway et al., 1991; Kookos and Perkins, 2001, 2016; Mehta and Ricardez-Sandoval, 2016; Koller et al., 2018; Rafiei and Ricardez-Sandoval, 2018, 2020a, 2020b, 2020c; Patilas and Kookos, 2020, 2021), decomposition methods of the MIDO problem into primal and master problems (Bansal et al., 2002, 2003; Hamid et al., 2010; Pistikopoulos and Diangelakis, 2015; Mansouri et al., 2016a, 2016b; Diangelakis and Pistikopoulos, 2017; Andersen et al., 2018; Burnak et al., 2019), parametric programming to incorporate uncertainty (Sakizlis et al., 2004), mixed-integer stochastic optimization frameworks (Mohideen et al., 1996), bi-level optimization (Palma-Flores and Ricardez-Sandoval, 2022) and control Q-parameterization techniques that account for parametric uncertainties in linear control problems (Swartz, 2004).

- Robust model-based approach

To address the challenges inherent in dynamic optimization, methodologies were proposed that focused on estimating bounds for process variables using surrogate models. These bounds play a critical role in evaluating the system's flexibility, stability, and controllability and are derived from linear models characterized by uncertainty, rather than relying on comprehensive nonlinear dynamic models. However, a notable limitation of this approach is their tendency to generate conservative and suboptimal designs, as it employs estimated bounds instead of true maximum values for flexibility and controllability. Furthermore, disturbances are analyzed primarily in terms of their maximum magnitudes, rather than their actual time-dependent profiles, potentially resulting in designs that are more conservative compared to those generated through dynamic optimization methodologies. Nevertheless, this robust approach offers an advantage in terms of computational efficiency, requiring substantially less computational effort than dynamic optimization-based formulations. Relevant works that fall into this category are (Mönnigmann and Marquardt, 2002, 2003; Marquardt and Mönnigmann, 2005; Chawankul et al., 2005, 2007; Malcolm et al., 2007; Ricardez-Sandoval et al., 2008, 2009b, 2009c, 2010, 2011; Wirth et al., 2011; Ricardez-Sandoval, 2012; Muñoz and Marquardt, 2013; Kastsian and Mönnigmann, 2013) .

In the studies discussed above, a common factor is their dependency on a pre-established flowsheet, i.e., the arrangement of the UOs is known *a priori*. Consequently, there are no decision variables regarding the inclusion or exclusion of a specific stream or UO. To address this limitation, the concept of a superstructure has been considered in integration of design and control. The key idea is to add a decision layer by considering multiple possible configurations for the placement of flows and UOs within a process flowsheet. This approach allows for the optimization of not only the process parameters but also the overall design and configuration of the process network. Relevant works that dealt with an integrated superstructure design and control problem include (M.L Luyben and Floudas, 1994; Schweiger and Floudas, 1998; Hamid et al., 2010; Sánchez-Sánchez and Ricardez-Sandoval, 2013; Trainor et al., 2013).

Despite these efforts, current methodologies that involve process flowsheet design are limited to case studies with a small number of UOs, often modeled using non-rigorous methods due to their high computational costs and mathematical complexity. Adding to that, case studies employing rigorous models are restricted to specific feasible regions, making their scalability or interpolation to other regions either prohibitively expensive or unfeasible. Instead of relying on rigorous models, surrogate models are introduced as data-driven tools capable to approximate expensive rigorous models by correlating relevant input and output data. With the increasing popularity of ML in the scientific

community, NNs have become the preferred surrogate model, as they can approximate any function given the correct and sufficient data and an appropriate architecture (i.e., network depth). Schweidtmann and co-workers have demonstrated that NNs not only improve computational efficiency by reducing the number of optimization variables but also significantly decrease solution times (Schweidtmann et al., 2019; Schweidtmann and Mitsos, 2019). Key applications of surrogate models in the integrated design and control problem include power series expansions for optimal design (Bahakim and Ricardez-Sandoval, 2014; Rafiei and Ricardez-Sandoval, 2020c), adaptive state-space models for control optimization (Malcolm et al., 2007) and dynamic model approximation for integrated design and control (Pistikopoulos and Diangelakis, 2015; Diangelakis and Pistikopoulos, 2017; Burnak et al., 2019). In addition to reducing the complexity of the optimization problem, the computational cost associated with these models is significantly lower than those that employ rigorous models. Despite their benefits, surrogate models have limitations, such as potential inaccuracies in estimating functions over wide operating regions. To address this challenge, techniques like piecewise surrogate modeling can be employed to capture varying system behavior more effectively (Rafiei and Ricardez-Sandoval, 2018; Pedrozo et al., 2020; Shokry et al., 2020; Obermeier et al., 2021). Notably, these examples do not incorporate NNs as surrogate models within the context of integrated design and control, highlighting an interesting research avenue.

As mentioned in Chapter 1, there are very few studies that address the integrated problem using RL. Sachio et al. (Sachio et al., 2022) were among the first to incorporate RL into the solution methodology for the integrated problem. Their approach involved reformulating the design and control problem as a bi-level problem, decoupling it by initially solving the optimal control using RL, and subsequently integrating it into the optimization problem for resolution. This reformulation effectively transformed the bi-level problem into a single-level problem while preserving optimal outcomes. This approach was demonstrated on tank and continuous stirred-tank reactor (CSTR) designs, with the REINFORCE algorithm (Williams, 1992) guiding the control actions. Mendiola-Rodriguez and Ricardez-Sandoval (Mendiola-Rodriguez and Ricardez-Sandoval, 2023) used RL entirely, employing a DDPG agent to find an optimal design and control scheme for an anaerobic digestion system, designing it initially and then controlling it throughout the process. In the fields of ML and RL, most research has focused on the generation and design of steady-state CPF (Table 2.1). Therefore, addressing the integrated design and control problem using RL represents a key avenue of research. Adding to that, in this context, approximating the dynamic behavior of UOs with NNs has never been explored before. Acknowledging the gaps in the existing literature, Chapter 4 presents the application

of RL to address the integrated design and control problem, leveraging NN as surrogate models to approximating the dynamic behavior of UOs.

2.4 Summary

This chapter presented the key methodologies, mathematical formulations, and theoretical background essential for this thesis. The first section opened with an overview of RL, offering a formal definition and classification of various agent types according to their learning strategies. Special emphasis was placed on the PPO agent, exploring its structure, various versions, and the advantages that makes this agent the preferred choice for addressing RL problems. Relevant works where RL techniques could be effectively applied within chemical engineering were presented. In the second section, the conceptual formulation for the design and optimization of CPFs was introduced. Model-based and model-free optimization methods were discussed, with particular emphasis on state-of-the-art RL methods as a solution to this problem. Despite the current interest in RL methods, which had led to the publication of numerous studies, several areas of research remain underexplored. One clear example is the integration of masking into the agent's formulation. Another example is the integration of advanced chemical process simulators, capable of handling multiple UOs, into a RL environment. To the author's knowledge, the implementation of masked agents interacting with rigorous simulation environments to solve design and optimization problems has not yet been addressed in the literature. Similar to the previous section, the third section introduced the conceptual formulation for the integrated design and control of CPFs. A literature review of the development and innovations within the integrated design and control framework was presented. Most studies focused on model-based optimization methods to address the integrated problem, with only two studies exploring this issue using RL. The implementation of RL has been identified as a challenging task; however, it represents a highly promising research avenue. Additionally, the use of NN surrogate models to approximate mechanistic models was discussed, as they reduce the number of variables to optimize and consequently decrease computation time. To the authors' knowledge, the application of RL in conjunction with NN surrogate models to address the integrated problem has not yet been explored. The next chapter presents a methodology that is proposed in this study for the generation, design and optimization of CPF using masked RL agents.

Chapter 3

A Reinforcement Learning approach with masked agents for chemical process flowsheet design

As discussed in Chapters 1 and 2, model-free optimization methods, particularly RL, offer a novel and effective approach to addressing design and optimization problems. Despite substantial research utilizing RL methods, a significant gap remains in the literature. Specifically, there is limited investigation into the application of masked RL agents and their integration with advanced chemical simulation suites, such as ASPEN Plus®, which handle multiple rigorous UOs within the RL environment. This gap underscores the necessity for further exploration and assessment to fully uncover and demonstrate the potential of RL in these complex contexts. To address this gap, this work presents a CPF design and optimization framework using RL, where the employed agents are equipped with the masking technique, enabling them to effectively discard infeasible actions during the decision-making process. Through a reward shaping process, the masked agents interact with the environment (comprised of an advanced chemical engineering simulation suite) and learn to design attractive and economically viable CPFs. The integration of masking with an environment that utilizes advanced process simulators, ensures a closer alignment with a more realistic CPF design practices, as it allows the consideration of interdependence between UOs or a specific hierarchical order in their arrangement, while modeling UOs with rigorous thermodynamic and conservation balance equations. The agents are trained to design CPF at steady state, adhering to general process design, operational, environmental and safety constraints. This study provides insights on the way RL generates flowsheets and learns complex structural relationships between linked UOs such as recycle streams utilizing the masking technique. A comparative study between the discrete and hybrid agent is performed by testing the agents in different case studies, shedding light into their benefits and limitations. The major outcomes of this chapter were showcased at the 12th IFAC Symposium on Advanced Control of Chemical Processes (ADCHEM 2024) (Reynoso-Donzelli and Ricardez-Sandoval, 2024c), published in the AIChE Journal (Reynoso-Donzelli and Ricardez-Sandoval, 2024d) and presented at the AIChE Annual Meeting 2024 (Reynoso-Donzelli and Ricardez-Sandoval, 2024a).

This chapter is organized as follows: the subsequent section presents the proposed RL methodology. Section 3.2 details the implementation of this methodology through three case studies, with two of these case studies utilizing ASPEN Plus® within their simulation framework. The chapter concludes with a summary in Section 3.3.

3.1 Solution framework

Solving a MINLP problems like that depicted in Equation 2.5 presents significant challenges due to the combination of discrete and continuous variables, along with nonlinear relationships. The inclusion of integer variables introduces combinatorial complexity, as the solution space grows exponentially with the number of discrete decisions. Additionally, these problems often involve disjunctive constraints—logical conditions that enforce specific operating modes or configurations—further complicating the optimization process. Nonlinearities can result in non-convex regions, leading to multiple local optima, making it difficult to identify the global optimum reliably.

To address the issues discussed above, this section presents a RL approach that is used to solve the CPF design problem. In RL, the environment is the external system or context with states, actions, and rewards, where an agent interacts to learn a policy that maximizes cumulative rewards over time. In parallel to traditional optimization, the environment corresponds to the mathematical problem formulation, encompassing all the variables, process model equations, constraints and the objective function, while the solver functions as the agent interacting within this environment to acquire (i.e., learn) a strategy, i.e., a policy, that aims to maximize cumulative rewards.

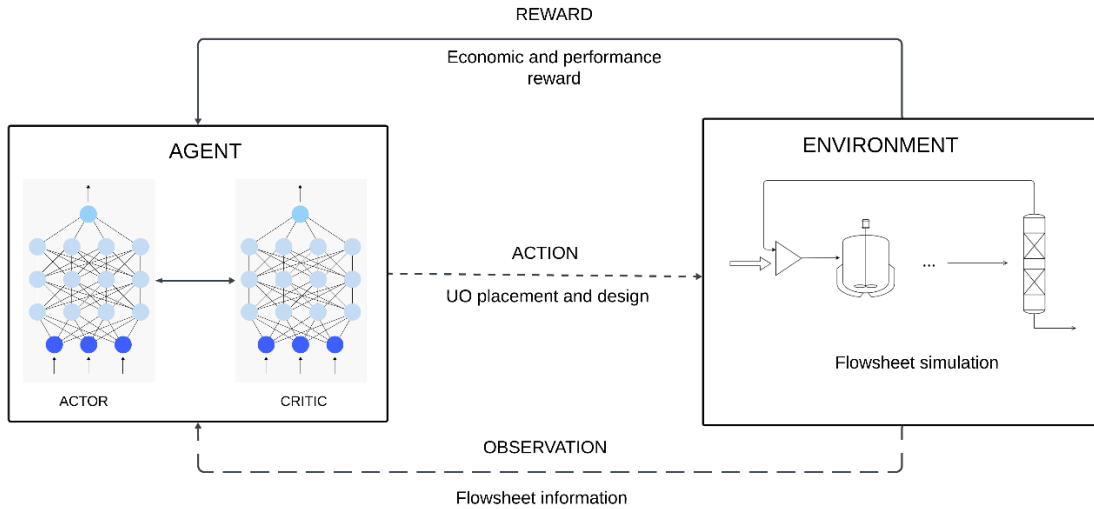


Figure 3.1: Agent-environment interaction

The interaction between the agent and the environment for CPF design is schematically represented in Figure 3.1. Section 3.1.1 provides details on the two agents presented in this work and their structures. Section 3.1.2 presents the action space, the observation, and the reward function. The addition of the novel masking technique proposed in this work is described within these sections.

3.1.1 Agents

In this study, the foundational structure of the agents is based on the typical PPO architecture proposed by Schulman et al. (Schulman et al., 2017). The agent's structure varies based on the nature of the addressed variables, adopting either a hybrid structure (i.e., a combination of discrete and continuous variables) or a conventional discrete PPO structure involving only discrete variables. Regardless of the agent, as a novel feature, both architectures incorporate the use of the masking methodology adapted from Huang and Ontañón (Huang and Ontañón, 2022).

3.1.1.1 Masked Discrete Proximal Policy Optimizer (mPPO)

The architecture of this agent presents a conventional structure, featuring a fully connected or dense neural network with one hidden layer having the same number of neurons. This was determined after several experiments where it was observed that increasing the number of hidden layers was not improving the training and instead leading to overfitting towards the training buffer and reducing the agent's exploration capabilities. Hyperbolic tangent ($\tanh$) activation functions are applied to all layers except for the last one; this choice is made to avoid disparities in the raw output scores (logits). By doing so, it prevents the creation of a strong bias towards one action over another during the network training, ultimately empowering the agent with enhanced exploration capabilities. Before reaching the activation function in the final layer, the masking filter is applied. The way this technique operates is as follows: as shown in Figure 3.2, at each step the agent receives an observation vector o_i containing information about the current state of the environment. In addition to the observation vector, the agent also receives a masking vector (mv_i) from the environment, which indicates with Boolean variables the position of the actions to be masked. An illustrative demonstration of the functioning of the masking function in the environment is presented in the next section. The integrated masking function in the agent works by setting the logit values ($\mathbf{l}$) from the last layer of the neural network, with the same position as the actions to be masked, to a large negative number ($\mathbf{l} \rightarrow -\infty$). These modified values are then propagated through the activation function – *SoftMax* – of the final layer of the actor network, transforming large negative values into the probability domain, effectively setting them to zero, i.e.,

$$\text{Softmax}(\mathbf{l}) = \frac{e^{l_{mask}}}{\sum_{j=1}^{n_{x_d}} e^{l_j}} \quad \forall \text{ mask} \in n_{x_d} \text{ and } \mathbf{l} \in \mathbb{R}^{n_{x_d}} \quad (3.1)$$

$$\lim_{l \rightarrow -\infty} e^{l_{mask}} = 0 \quad (3.2)$$

During training, the agent needs to randomly sample discrete actions, which is not directly feasible from this output. Therefore, once the probability vector is obtained, it is transformed into a categorical distribution, from which the agent can sample discrete actions at each step.

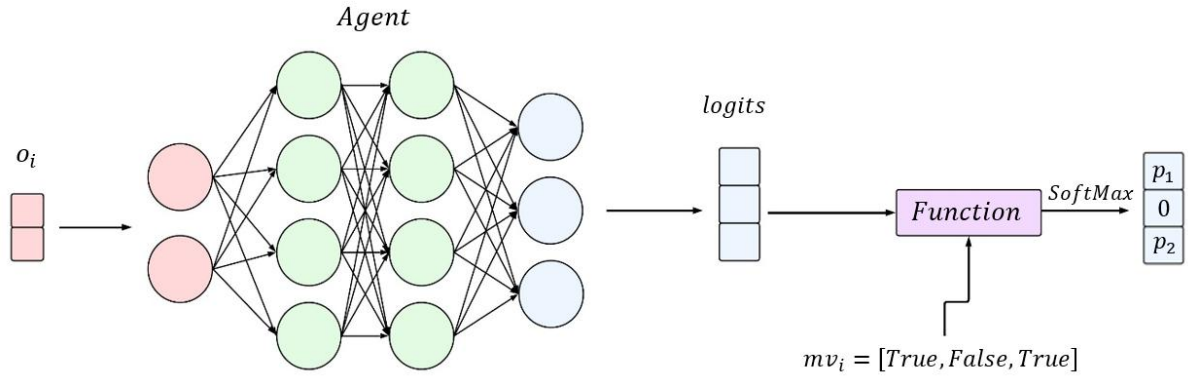


Figure 3.2: Illustrative example of the masking functionality

3.1.1.2 Masked Hybrid Proximal Policy Optimizer (mHPPO)

The actor network architecture for this agent, shown in Figure 3.3, is based on the Hybrid Proximal Policy Optimizer (HPPO) proposed by Fan et al. (Fan et al., 2019), with the addition of a masking technique in the discrete component. Masking is only applied to discrete actions, meaning continuous actions cannot be masked. As a result, the agent may consistently select certain continuous actions, even when corresponding discrete actions are masked.

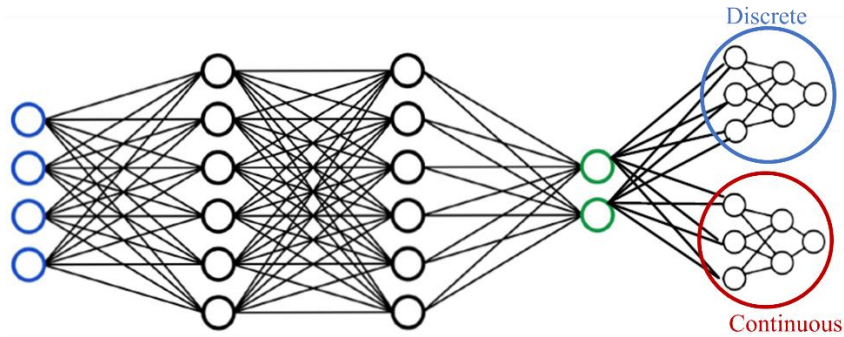


Figure 3.3. Network architecture for the mHPPO actor

The architecture of the actor network in the mHPPO agent, as depicted in the Figure 3.3, can be described in two main sections. The first part functions as an encoder, where the network progressively reduces the number of neurons through layers, culminating in a two-neuron layer. From

this layer, two separate fully connected networks emerge. The first network, responsible for discrete decision-making, adopts a similar architecture to the one described in section 3.1.1.1 and includes the masking mechanism mentioned earlier. This component handles discrete action selection, typically employing a categorical distribution to sample actions. The second network, designed for continuous decision-making, processes actions using continuous probability distributions. The network consists of hidden layers with hyperbolic tangent activation functions, and it outputs two parameters, α_B and β_B , which define the shape of a Beta distribution: $B(\alpha_B, \beta_B)$. The activation function used to change the logits into these parameters is a *SoftPlus* function.

During training, the total loss function combines the losses from both the discrete and continuous parts to update the whole network via backpropagation. Experiments exploring sequential updates, where the discrete or continuous parts are frozen during the other's update, revealed no significant difference in performance compared to joint updates. This suggests that the network efficiently learns both discrete and continuous policies in parallel.

3.1.2 Environment

The environment is the segment of the RL framework where the optimization problem (Equation 2.5) is translated into RL terms. This subsection delineates its distinct components: the action space, the observation vector, and the reward function. The action space is the sole component whose structure varies depending on the agent, which has been described above.

3.1.2.1 Action space

The action space ($\mathbf{A}$) of a hybrid agent (mHPPO) is composed of the set of all discrete ($\mathbf{x}_d \in \mathbb{N}$) and continuous ($\mathbf{x}_c \in \mathbb{R}$) variables, i.e.,

$$\mathbf{A} = [\mathbf{x}_d, \mathbf{x}_c] \quad (3.3)$$

Note that $\mathbf{A}$ does not encompass the combination of all discrete and continuous variables, but rather probability distributions. As the RL method requires the selection of an action at each step i , actions are sampled from these distributions, resulting in the step-action a_i . For the discrete variables, there is a categorical distribution from which only one value is stored in a_i . In the case of the continuous variables, each has its own distribution, i.e., a Beta distribution. Thus, a_i is composed of n_{x_c} values, even if some of them are not strictly used in that particular step.

For a fully discrete agent (mPPO) the continuous variables can be discretized and unlike the previous action space, this discretized version ($\mathbf{A}_{FD}$) contains all the combinations of the former discrete variables with their corresponding realizations of the discretized continuous variables. The primary motivation to discretize continuous variables within the action space is to streamline the agent's exploration domain, thus enhancing its interaction with specific actions. Note that this may not be feasible in a direct manner with large infinite search spaces produced by $\mathbf{A}$. This simplification comes with drawbacks, including limited flexibility – discrete action spaces may not adequately represent the continuous nature of the variables – loss of precision –, discretizing actions can lead to a loss of precision in decision-making, as the granularity of actions is limited by the discretization – and the requirement for prior knowledge of the criterion used to perform the discretization. The discretization criterion depends on the case study under consideration, the computational budget, and the expected accuracy in the solution. Typically, a refined discretization will likely return accurate results at the expense of increasing the search space domain thus leading to a lengthy training, and consequently larger computational costs. Note that the present RL framework can be adapted to multiple discretization methods in the continuous variables' domain, e.g., uniform, normal discretization. $\mathbf{A}_{FD}$ is defined as a database, consisting of keys and values; the keys represent the discrete actions available for the mPPO to take, while the values associated with these keys denote the different combinations of the discretized continuous realizations. Once the key is chosen by the discrete agent, its value is mapped in the database, and the associated value is used.

Let us consider the following illustrative example to demonstrate the application of the two agents delivering discrete and hybrid actions spaces: a heater and a cooler must be arranged in series. The design variable for both UOs is their operating temperature, ranging from 100°C to 105°C for the heater and from 5°C to 10°C for the cooler. For simplicity, continuous actions within the discrete action space are discretized into two possible realizations: high and low. Therefore, for the heater, values are set to 100°C and 105°C, while for the cooler, they are 5°C and 10°C.

Table 3.1: Action space example

Hybrid action space (mHPPO)	Discrete action space (mPPO)
$\mathbf{A} = [\mathbf{x}_d, \mathbf{x}_c]$ $\mathbf{x}_d \in \{Heater, Cooler\}$ $\mathbf{x}_c \in \{[100, 105], [5, 10]\}$	$\mathbf{A}_{FD} = [1, 2, 3, 4]$ where: Database = {1: 100°C, 2: 105°C, 3: 5°C, 4: 10°C}

As shown in Table 3.1, the mPPO includes only 4 discrete actions; the first two represent the heater whereas the last two represent the cooler. On the other hand, the mHPPO has an infinite number of actions due to the use of continuous variables. For this case, the mHPPO step-action vector is composed of three random selected values, while for the mPPO only one discrete realization selected at random is chosen per step. Let us take as an example a random step n , the outputted mHPPO step-action vector could be: $a_n = [Heater, 103.43, 7.34]$, where the UO selected is the heater and will be used at a temperature of 103.43°C. Note that also a value for the temperature of the cooler is calculated; however, it is not used in that specific step. In the case of the mPPO the action vector, if a random action is selected, e.g., $a_n = [3]$, this realization corresponds to the low-temperature cooler.

3.1.2.2 Observation vector

The step-observation vector (o_i) is the portion of the state that is conveyed to the agent as information about the environment at each step (Sutton and Barto, 2014). This is defined as follows:

$$o_i = [\varpi_i, q_{i,rp}, i] \quad (3.4)$$

The structure of o_i may change depending on the setting of the problem; however common features remain. For instance, at each step we keep track of: a) process operating variables (ϖ_i), e.g., temperature, pressure; b) chemical components (reactants or products) tracker ($q_{i,rp}$), which could be molar, mass or volumetric flowrates, as well as molar or mass fractions, where $rp \in \{reactants, products\}$; c) the step of the current observation (i), to add temporal context to the vector and differentiate potential identical action responses. The elements in o_i are normalized, thus preventing larger values from having a greater influence on data processing within the agent's neural network.

3.1.2.3 Reward function

The reward function represents a numerical measure of the immediate benefit or cost derived from the agent's action in a specific state (Sutton and Barto, 2014), thus, the reward is assigned at each step (r_i). Designing an effective reward function is crucial since it needs to be explicit enough to facilitate the agent's assertive learning and sufficiently flexible to generalize well across the task at hand. For simple processes, such as the maximization of the agent's interaction with the environment or reaching a specific state, the design of the reward function is relatively straightforward. However, for more complex problems where the objective is not as punctual as the other cases – e.g., opposing maximization and minimization objectives – and there are constraints to uphold, designing an effective

reward function becomes a more challenging task. In order to tackle this challenge, a method referred to as reward shaping is used in this work:

$$r_i = \sum_{b \in \mathcal{B}} r_b \quad (3.5)$$

The step reward (r_i) used in this problem is decomposed into b -individual sub-rewards ($b \in \mathcal{B}$) which promote different objectives in the policy (Wiewiora, 2010). This approach is applied to the objective function and constraints included in Equation 2.5, i.e., assign separate sub-rewards to each of these elements, as illustrated in Table 3.2. Additional to these elements, and to make the agent acquire a better understanding of the problem, extra sub-rewards are added. One of these extra sub-rewards is the driving force, defined as the difference (Δ) in one distinctive parameter, e.g., conversion, molar fraction, purity, etc. This sub-reward empowers the agent to actively progress toward the creation of a CPF rather than opting to remain stationary. In CPF design, the RL agent employs a specific product, as its driving force; however, the process may also produce other by-products of interest. To prevent the agent from losing track of these by-products, an additional sub-reward is introduced, where the agent is rewarded for achieving a specific objective with these by-products, e.g., reaching a specified purity, produce a certain quantity of by-product, etc. Another sub-reward is introduced to motivate the agent to design short and effective CPF, so in the fewest steps possible. This sub-reward adds the unused steps ($I - i$) to r_i , where I represents the maximum number of steps allowed in an episode.

Table 3.2: Reward shaping

Optimization problem equations	Translation to RL sub-reward
Minimization of $f(\mathbf{x}_d, \mathbf{x}_c, \mathbf{x})$	Objective: The penalty increases as the values obtained from evaluating the function becomes more negative.
Constraints $\mathbf{h}/\mathbf{g}(\mathbf{x}_d, \mathbf{x}_c, \mathbf{x})$	Process design goal: The penalty lowers as the constraint violation approaches the desired value.
-	Specific constraint (UO constraint): Constant value provided for the constraint satisfaction
-	Driving force: Varying reward depending on the magnitude of the driving force.
-	By-product reward: Varying reward depending on the achievement of the specified goal
-	Short flowsheets: Varying reward depending on the remaining steps.

As presented in Equation 2.5, the main goal is to minimize the objective function. To avoid the agent from choosing the actions with the lowest penalty, the previously defined driving force is used to balance out the rewards, thus avoiding trivial solutions. As mentioned in the problem statement section, the constraints ($\mathbf{h}(\mathbf{x}_d, \mathbf{x}_c, \boldsymbol{\chi})$ and $\mathbf{g}(\mathbf{x}_d, \mathbf{x}_c, \boldsymbol{\chi})$) can be divided into two categories, the process design goal requirements, and the specific constraints. The former only become relevant in the final steps of the episode. Failing to meet those constraints compels the agent to continue constructing the CPF, potentially leading to incorrect designs—resulting in highly negative rewards—or failing to satisfy the specified constraint within I , resulting also in a negative reward. Specific design constraints, which are only activated for certain actions, offer an additional constant sub-reward to the agent when the constraint is not violated. Treating these constraints as penalties, where the agent is penalized for any violation (similar to a constraint programming approach) may lead to poor learning rates. This formulation hinders the agent's exploration, resulting in convergence toward suboptimal results or, in some cases, it may lead to divergence. Therefore, a constant value reward is chosen; as it does not limit the agent's exploration freedom and rewards it when operating within the established criteria. Note that disjunctive constraints and constraints managed by the chemical simulator do not directly translate into the RL sub-reward shaping system. The latter is managed within the simulator, representing a black box for the agent, thereby denying direct access, while the former is addressed through the masking technique explained before.

In the proposed environment, the masking function generates a masking vector mv_i , consisting of Boolean values that control the activation or deactivation of discrete actions within the action space. This vector is then passed to the agent to enable logits masking, ensuring that only valid actions are considered during decision-making. As an illustrative example, consider the following case: the dependency between the binary variables x_{d_1}, x_{d_2} and x_{d_3} . The variables x_{d_2} and x_{d_3} can only be chosen if x_{d_1} has been previously selected, i.e.,

$$x_{d_1} \Rightarrow (x_{d_2} \wedge x_{d_3}) \quad (3.6)$$

$$x_{d_2} + x_{d_3} \leq x_{d_1} \quad (3.7)$$

$$x_{d_1}, x_{d_2}, x_{d_3} \in \{0, 1\} \quad (3.8)$$

Note that the direct translation between Boolean and binary variables is straightforward: 0 represents False and 1 represents True. Regardless of the action space nature – hybrid or fully discrete – the masking process is performed the same way in the environment. Following the previous example, given the state of the environment, it would create a starting masking vector: $mv_i =$

$[True, False, False,]$, corresponding to an action space: $\mathbf{A}_{ex} = [x_{d_1}]$. This action space would be passed to the agent, making x_{d_1} the only available action to be chosen. Once chosen, the masking vector changes to $mv_i = [False, True, True]$ and the action space to $\mathbf{A}_{ex} = [x_{d_2}, x_{d_3}]$. Making x_{d_2} and x_{d_3} available in the action space for the agent to choose for.

3.2 Case studies

In this section, the methodology presented in section 3.1 is tested on three case studies adapted from the literature, each highlighting distinct features of the proposed framework. The first two case studies employ both discrete and hybrid approaches to design the process flowsheet, enabling a comparative analysis of the results obtained through each methodology. The third case study, in contrast, utilizes only the hybrid agent for the design and optimization of the CPF. The first case study serves as a proof of concept and focuses on providing insights into the operation of the hybrid and discrete agents when synthesizing a relatively simple process. This case explores key aspects such as the interaction between discrete and continuous variables within the hybrid agent, as well as the role of masking in guiding agent decision-making. These findings are essential for understanding the proposed methodology and serve as a foundation for applying it to more complex scenarios. The second and third case studies aim to design more advanced processes, further exploring the potential of the RL framework as a viable tool for CPF design. A notable aspect of these case studies is the integration of masking within a simulation environment that incorporates advanced process simulators, such as ASPEN Plus®. This approach ensures a closer alignment with realistic CPF design practices, allowing for the consideration of interdependencies between UOs and the specific hierarchical order in which they are arranged, while rigorously modeling UOs based on thermodynamic and conservation balance equations.

All the case studies were implemented on a PC with Intel® Core™ i7-3770 CPU @ 3.40 GHz and 32 GB of RAM, using Python as the programming language. The main libraries used were torch for the development of the neural networks and gym for creating the RL environment.

3.2.1 Case study 1

The objective of this case was to design a CPF that enables the conversion of reactant A into product B up to a 97.5% conversion rate, while minimizing the process economics. For this case three UOs (discrete actions) with their associated design variables (continuous actions) were considered. Note that the equations modeling each of these operations stem from simple mass and energy balances. This simplification enables a deeper focus on comprehending the design and optimization capabilities of the

agents introduced in the previous section. The UOs involved in this case study include a mixer (M); a continuous stirred-tank reactor (CSTR) (Schweiger and Floudas, 1998), with design variables being the diameter (D_{CSTR}) and height (H_{CSTR}) of the reactor; and a flash tank (FT), with the design variable being the vapor/feed fraction (vf). A detailed description of the UOs, their models and cost functions are provided in Appendix A.

For the present case study, the action space ($\mathbf{A}$) is composed by the discrete actions $\mathbf{x}_d \in [\text{M}, \text{CSTR}, \text{FT}]$ and the continuous actions $\mathbf{x}_c \in [\mathbb{R}_{D_{CSTR}}, \mathbb{R}_{H_{CSTR}}, \mathbb{R}_{vf}]$. For the mPPO, $\mathbf{A}_{FD}$ was discretized as follows. D_{CSTR} and H_{CSTR} in CSTR were each discretized into three uniform realizations between their upper and lower limits. The resulting combinations between each of these realizations in D and H resulted in a total of 9 discrete actions. Regarding FT, the domain of vf was uniformly divided into 6 discrete values. Taking into consideration the discrete action associated with the choice of a M, combined with the discrete actions of CSTR and FT, a total of 16 discrete actions were considered for this problem. The observation vector at each step contains information about the outlet stream of the current UO: concentration of the reactant A, temperature, total molar flowrate, current step, and the achieved conversion, i.e., $\mathbf{o}_i = [C_A, T, F, i, x_A]$. The reward function considered in this case study follows the method described in the section 3.5 and is represented by four sub-rewards shown in Table 3.3.

Table 3.3: Sub-rewards case study 1

Sub-reward	Reward or penalty	Assigned	Description
Driving force	Reward	Every step	Quantified as the conversion difference between consecutive steps, it signals the UO's efficacy in transforming reactant A into product B.
UO cost	Penalty	Every step	Quantifies the cost associated with each UO.
Process design requirement	Penalty	Final step	Examines whether the desired conversion was attained. Depends on the gap between the achieved conversion and the desired conversion.
Short flowsheets	Reward	Final step	Denoted as the difference between the maximum number of steps I (which for this case was 10) and i .

Note that all sub-rewards listed in Table 3.3 were normalized to prevent an individual term from having a disproportionate impact, e.g., the operating cost of a UO. Regarding the masking, to prevent the agent from selecting a trivial sequence like a M followed by a FT – where no conversion

occurs –, the condition to unmask the FT requires the existence of at least one reactor between these operations. Additionally, to avoid confusion for the agent concerning the recycling, once M is chosen, that operation is masked and unmasked again only after FT is selected by the agent.

All the hidden layers in both agents consisted of 64 neurons. Both agents were trained ten times for 75,000 steps, taking around 30 minutes per training. The networks were updated every 2048 steps with a discount factor of 0.99 and optimized using Adam (Kingma and Ba, 2017) optimizer with a starting learning rate of $2.5e-4$, which was adjusted throughout the training and from preliminary simulation tests.

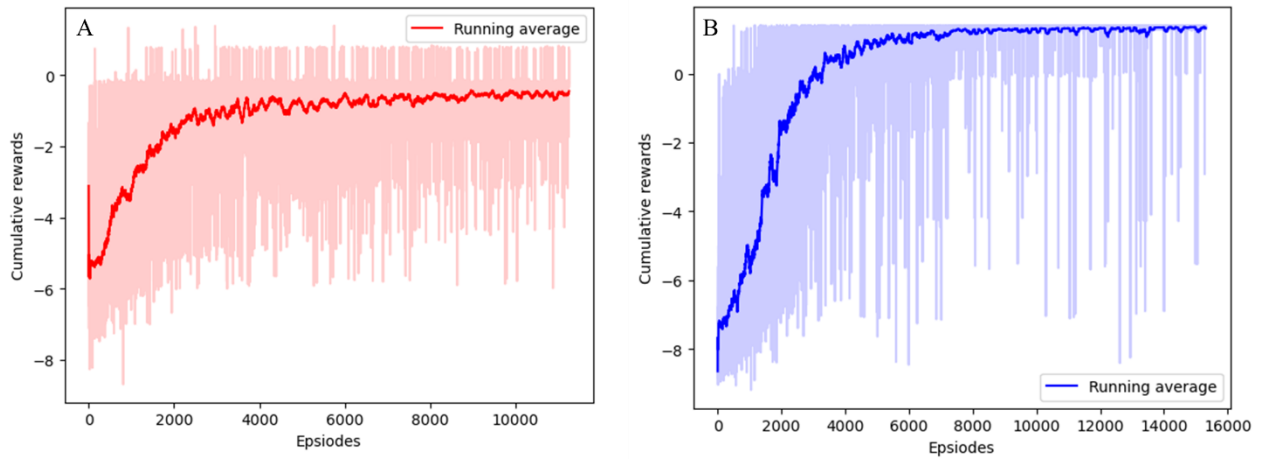


Figure 3.4: Learning curve for a single run: A) mHPPO B) mPPO

Figure 3.4 illustrates the learning curves of both agents for a single training run. The solid line, representing the running mean, depicts the trend of cumulative rewards throughout the learning process, while the faded region indicates the cumulative rewards obtained at the end of each episode. In both cases, the running mean exhibits a similar behavior—an exponential growth in the initial episodes followed by a plateau when reaching a certain value. On the other hand, the behavior of cumulative rewards is notably different. In mPPO, the variability is almost reduced at the end of training, whereas mHPPO shows a slight reduction compared to the initial episodes. Also, the plateau reached by mHPPO indicates a potential room for improvement, as it does not reach the maximum cumulative rewards achieved by the agent, whereas mPPO does achieve this condition. The differences in behavior between agents can be attributed to the types of variables considered and their corresponding relationships.

mPPO operates with a single type of variable—discrete variables. Discrete variables generate a finite action space; hence, the agent has a finite number of actions to choose from. In addition to this, and boosted by the masking technique, the agent is capable of finding the optimal combination of

actions to achieve maximum rewards. A convergence test on the number of discretization points is presented in the Appendix A.

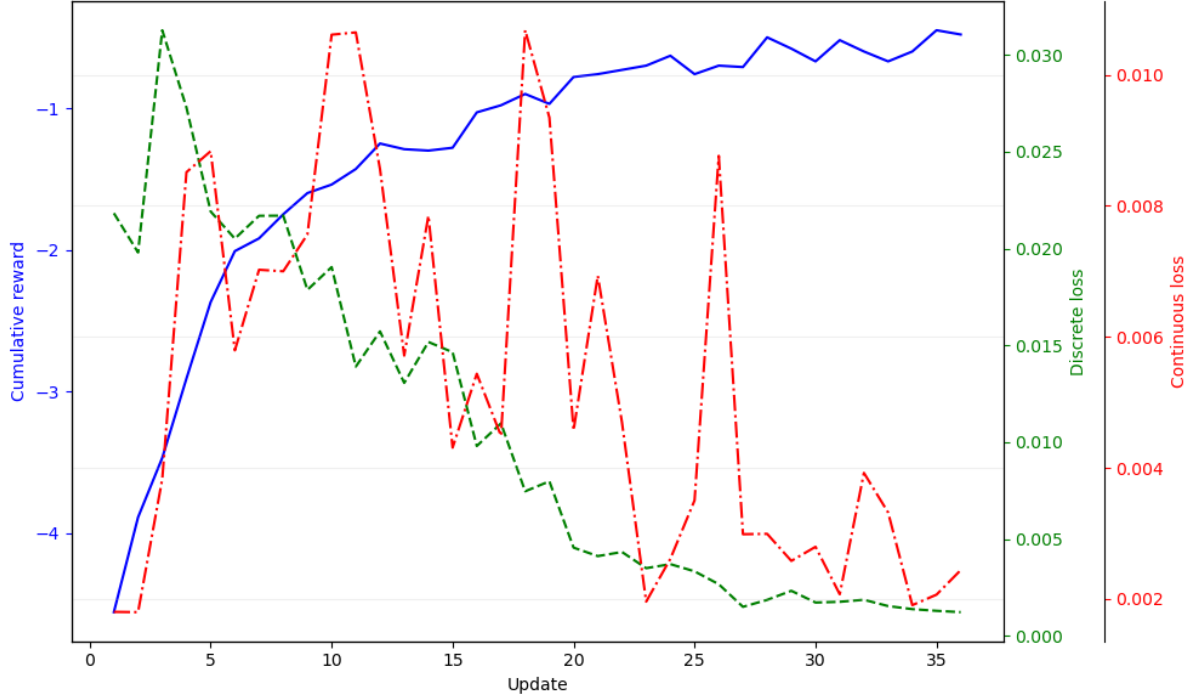


Figure 3.5: Reward and losses profiles for mHPPO

On the other hand, mHPPO faces a more complicated task as it handles both discrete and continuous variables. In addition to creating an infinite search space, the combination of discrete and continuous variables has a more substantial impact on the agent’s learning process. In the neural network, the agent’s loss function consists of two components—one discrete and one continuous. Figure 3.5 presents the cumulative reward and losses profiles with respect to the agent’s updates – optimization via backpropagation of the neural network. The losses often have opposing objectives, potentially leading to a trade-off. Figure 3.5 clearly illustrates this conflict, wherein as the agent learns—manifested by the increase in cumulative rewards—the discrete loss decreases while the continuous loss fluctuates. Throughout the training process, it was observed that at the initial updates of the network, as the continuous loss decreased, there was a corresponding increase in the discrete loss. This prompted a modification in the agent’s strategy for the subsequent update, resulting in a reversal of roles. While both losses show a decreasing pattern over the updates, it suggests that the behavior of the continuous loss is influenced by the discrete loss, preventing it from exhibiting a consistently decreasing pattern. Moreover, the cumulative reward ceases to grow once the discrete loss reaches a

plateau, also hindering the decrease in the continuous loss. This is reflected in the decision-making process of actions, where the agent engages in a trade-off between both discrete and continuous decisions, ultimately arriving at a response that could be deemed as the best trade-off between the two types of variables. Note that the behaviors exhibited in Figure 3.5 are consistent for the rest of all mHPPO runs.

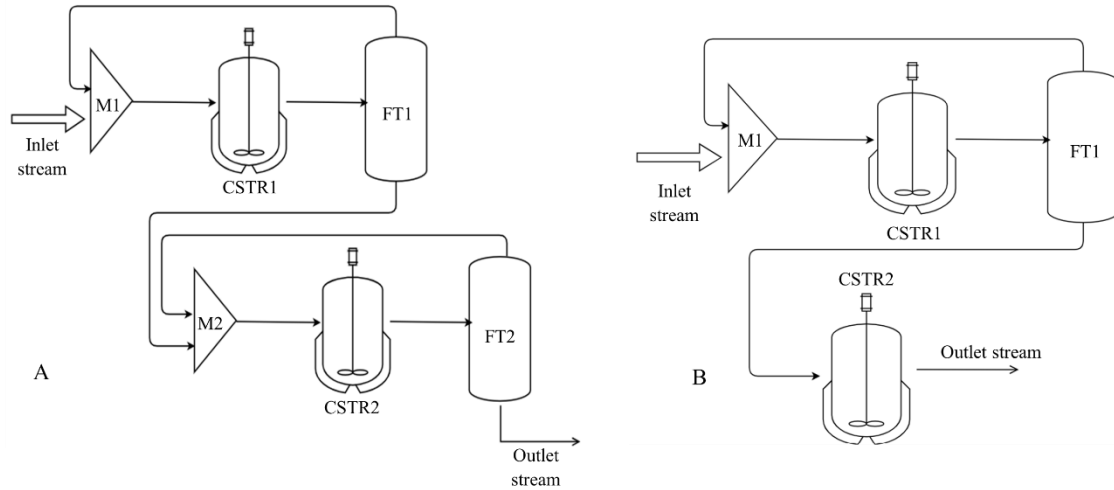


Figure 3.6: CPF obtained for case study 1: A) mHPPO B) mPPO

When evaluating the best-performing agents, the resulting CPFs (Figure 3.6 and Table 3.4) are capable to reach the desired conversion while learning the advantage of adding recycling to the process flowsheet. Achieving shorter and superior flowsheets than the naïve assumption of merely placing reactors in series, which would have also achieved the desired conversion. The significant difference between the two flowsheets shown in Figure 3.6 lies in the sizing of the equipment, thereby affecting the length of the CPF. In the discrete case (mPPO), the agent selects the first reactor with the largest dimensions, the flash tank with the smallest liquid-vapor fraction, and the second reactor with the largest diameter and smallest height. Due to the previously explained behavior, the mHPPO agent seeks to find for a balance (trade-off) in its decisions by opting for reactors of medium size, compelling the process to have more units.

Table 3.4: UOs designs and specifications case study 1

Unit operation	Design variable	mHPPO	mPPO
CSTR1	Diameter (D_{CSTR}) [m]	2.2073	2.3622
	Height (H_{CSTR}) [m]	2.0409	2.3622
CSTR2	Diameter (D_{CSTR}) [m]	2.0193	2.3622
	Height (H_{CSTR}) [m]	2.0203	1.6764
FT1	Vapor/Feed (vf)	0.3501	0.2000
FT2	Vapor/Feed (vf)	0.2559	-
Achieved conversion (x_A)		0.9816	0.9768
Cumulative reward		-0.0951	1.3993

Outcomes from this case study provide insights that can aid in understanding the behavior of the PPO agents, and the type of rewards they should consider. When translating the optimization problem into RL terms, it is crucial to design a non-sparse reward system for the agent, i.e., the agent should not receive all the rewards at the end of the episode. Instead, the agent must receive a reward at each step that is descriptive enough to allow it to learn the consequence of choosing an action in a specific position. In the case of the mHPPO, the actions taken align with this idea, as each action has the greatest impact on the conversion of the reactant, while seeking to minimize the process economics. Hence, the designed CPF may not be the shortest, but each action seeks to yield the highest reward. The mPPO operates on the same principle; however, given the simplicity of the case study – a reduced and discretized search space – the agent is capable of identifying an attractive combination within the feasible action space.

3.2.2 Case study 2

This case study aims to design the CPF for a dimethyl ether (DME) production plant, based on the work proposed by the authors Caballero et al. (Caballero et al., 2007) and Luyben (Luyben, 2017), while minimizing the process economics. In contrast to case study 1, this case study features the use of a chemical simulation software (i.e., ASPEN Plus® v8.8) as part of the simulation environment. The key motivation behind the development of this hybrid platform that combines a computer programming language (i.e., Python) with a chemical simulation software such as ASPEN Plus® lays in harnessing the thermodynamic packages and equations governing all declared UOs within the latter. Notably, this approach facilitated the incorporation of a broader spectrum of UOs, resulting in processes that closely emulate actual design procedures. This study stands out from others employing ASPEN Plus® in the simulation environment by virtue of its incorporation of multiple types of UOs, distinguishing itself from works that solely report the utilization of a single UO (van Kalmthout et al., 2022). Additionally, the proposed RL framework is capable of designing CPFs from an empty flowsheet, automatically

connecting and simulating the selected UOs. To the authors’ knowledge, this has not been attempted before; hence, the novelty of this work.

The goal of this case study is to formulate a CPF for the production of DME – obtained through the dehydration of methanol – that can achieve outflows with a purity of 99% for both the produced DME and water. The dehydration of methanol (Equation 3.9) is a reversible reaction carried out in the presence of a catalyst, considering the catalyst efficiency, the temperature within the catalytic reactors employed in this process must be maintained below 400°C. In this case study, only one type of catalyst is being considered. The exploration for optimal catalyst materials falls outside the scope of this study but could be expanded within the current framework by incorporating reactors with different catalyst materials. This can potentially lead to the specification of a set of reactors, each with a different catalyst material and performance, e.g., product yield or conversion.



The thermodynamic model selected for this case study was the Universal Quasichemical Activity Coefficient (UNIQUAC) model with Redlich-Kwong equation of state. Firstly, this choice was made in alignment with the decision of the authors Caballero et al. (Caballero et al., 2007) who specifically opted for this model. Additionally, the UNIQUAC model combined with the Redlich-Kwong equation of state offers a comprehensive approach for accurately predicting phase equilibria and thermodynamic properties of complex systems. The UNIQUAC model is well-suited for describing non-ideal liquid mixtures, particularly those containing polar and associating molecules, while the Redlich-Kwong equation of state provides a reliable framework for estimating vapor-phase behavior. Together, these models offer a robust and versatile toolset for capturing the thermodynamic behavior of the dehydration of methanol, making them an ideal choice for this study.

The UOs and their corresponding design variables are described in Table 3.5.

Table 3.5: UOs description case study 2

Unit operation	ASPEN Plus® block	Design variable	Bounds
Mixer (M)	<i>MIXER</i>	-	-
Pump (P)	<i>PUMP</i>	Pressure	Fixed to 10 bar
Heat exchanger (HEX)	<i>HEATER</i>	Operating temperature (T_{HEX})	150-400 [°C]

Cooler (C)	<i>HEATER</i>	Operating temperature (T_C)	5-50 [°C]
Plug flow reactor with constant cooling (PFR)	<i>RPLUG</i>	Diameter (D^{PFR})	0.5-3.5 [m]
		Length (L^{PFR})	6.5-12 [m]
Adiabatic plug flow reactor (APFR)	<i>RPLUG</i>	Diameter (D^{APFR})	0.5-3.5 [m]
		Length (L^{APFR})	6.5-12 [m]
Binary distillation column (DC)	<i>RADFRAC</i>	Number of stages (N)	5-25
		Distillation rate ($dist$)	20-110 [kmol/hr]
Binary distillation column with recycle (DCR)	<i>RADFRAC</i> + <i>SPLITTER</i>	Number of stages (N)	5-25
		Distillation rate ($dist$)	20-110 [kmol/hr]
		Recycle ratio (rr)	0.5-0.95
Ternary distillation column (TC)	<i>RADFRAC</i>	Number of stages (N)	5-25
		Distillation rate ($dist$)	70-110 [kmol/hr]
		Intermediate distillation rate ($inter_{dist}$)	20-60 [kmol/hr]
		Number of stages (N)	5-25
Ternary distillation column with recycle (TCR)	<i>RADFRAC</i> + <i>SPLITTER</i>	Distillation rate ($dist$)	70-110 [kmol/hr]
		Intermediate distillation rate ($inter_{dist}$)	20-60 [kmol/hr]
		Recycle ratio (rr)	0.5-0.95

To simplify the analysis and prevent an excessive number of degrees of freedom, a few UO has some variables specified *a priori*. For instance, specifications for the reactor are the reaction that takes place (Equation 3.9), heat transfer coefficients and the catalytic load, which is a function of the reactor's design. For the distillation columns, the feed stage is located at the midpoint of the tower. Additionally, for ternary columns, the intermediate flow outlet is established in the first third of the column. However,

it is important to note that if the user intends to leave these variables as degrees of freedom, the proposed framework is sufficiently flexible to accommodate this choice.

In the RL domain, the hybrid action space consists of 10 discrete variables and 19 design actions – 15 continuous variables and 4 discrete variables. Due to the large number of discrete actions and their corresponding continuous actions, in order to maintain a reasonably sized action space, each continuous variable was uniformly discretized into three realizations, resulting in a fully discretized action space of 170 actions. For each step, the observation vector returned information about the outlet stream of the selected UO, i.e., temperature (T), pressure ($Press$), the current step, and the molar fractions (ω) of methanol ($MeOH$), water (H_2O) and dimethyl ether (DME), i.e., $o_i = [T, Press, \omega_{MeOH}, \omega_{H_2O}, \omega_{DME}, i]$. The reward function considered in this case study is similar to that used for the previous case study and follows a similar format for the sub-rewards, as shown in Table 3.6. As in the previous case study, all sub-rewards were normalized to prevent an individual term from having a disproportionate impact.

Table 3.6: Sub-rewards case study 2

Sub-reward	Reward or penalty	Assigned	Description
Driving force	Reward	Every step	Quantified by the molar fraction difference between the current and previous steps, its magnitude reflects the UO's effectiveness in reducing the current stream's MeOH content.
UO total costs	Penalty	Every step	Composed of the capital and operational costs of the UO. The capital costs of each UO were calculated using functions derived from Douglas (Douglas, 1988).
Process design requirement	Penalty	Final step	Ensures that the desired purities of DME and water are met. Magnitude depends on the gap of the achieved purities and desired purities.
Specific design constraint	Reward	Specific step	Constant reward granted to the agent when the UO is a reactor and meets the design specification (operating below 400°C). No penalty is awarded if this constraint is not met.
Short flowsheets	Reward	Final step	Denoted as the difference between the maximum number of steps I (which for this case was 15) and i .
DME extra	Reward	Specific step	Granted to the agent upon achieving the target DME purity in the current step. The magnitude hinges on the distilled amount of pure DME, with a higher value for larger

flows. Notably, this sub-reward is awarded only once, contingent on attaining the desired water purity for a complete flowsheet.

Unlike case study 1, the masking mechanism in this case study is composed of different process stages, as shown in Algorithm 3.1. Within each of these stages (see Algorithm 3.1), specific UOs are temporarily unmasked. Once the stage is completed, the UOs and the stage itself are then masked. All the required values to complete the stage and advance to the subsequent stage are derived from Luyben's (Luyben, 2017) work and set to prevent ASPEN Plus® simulation errors.

Algorithm 3.1 Pseudocode for the masking process, case study 2

```

done = False
stage = preparation
avail_units = []

while not done do
  if stage = preparation then
    avail_units = [M, P, HEX]
    if  $T \geq 250^{\circ}\text{C}$  and  $Press = 10\text{ bar}$  then
      stage = reaction
    end if
  elif stage = reaction then
    avail_units = [PFR, APFR]
    if  $conv_{MeOH} \geq 0.75$  then
      stage = cooling
    end if
  elif stage = cooling then
    avail_units = [C]
    stage = separation
  elif stage = separation then
    avail_units = [DC, DCR, TC, TCR]
    if  $purity_{DME} \geq 0.99$  and  $purity_{water} \geq 0.99$  then
      done = True
    end if
  else
    Error
    done = True
  end if
end while

```

A more detailed description of each stage, and the derivation of intermediate stage constraints, is presented as follows. Based on the works of Caballero et al. (Caballero et al., 2007) and Luyben (Luyben, 2017), different parameters for the masking process were derived. The dehydration of methanol is a reaction carried out in a catalytic medium and must be performed at specific temperature and pressure conditions. Caballero et al. (Caballero et al., 2007) specify that, the flow entering the reactive process must be a gaseous flow compressed to a pressure of 1000-1500 kPa and at a

temperature above 200°C. These were precisely the conditions specified for the first stage of masking, where the agent had to design the preprocessing stage able to achieve these conditions. Note that, for this case, the desired minimum temperature was not 200°C but 250°C, as argued by Luyben (Luyben, 2017), who states that with an inlet temperature of around 290°C, the reactant conversion is even more effective. For the reactive section, Luyben (Luyben, 2017) presents the possibility of using two types of reactors, one adiabatic and one with a constant cooling flow. Regardless of the reactor type, due to the nature of the reaction, the highest achievable conversion is around 80%. As a specification to proceed to the next stage, the agent must achieve a conversion of 75% or higher. The threshold was set lower than that reported in the experiment by Luyben (Luyben, 2017) with the idea that the agent should learn to approach these conversion values. Additionally, both reactors must comply with the design constraint of not exceeding the operating temperature of 400°C at any time. To avoid simulation issues and thereby improve the agent's learning process, a section for cooling and condensing the outlet flow from the reactor was introduced. Without this stage, the agent would often select distillation columns before the flow is condensed, leading to errors in the ASPEN Plus® simulation and thus severely affecting the agent's learning. This section is composed by only one UO, a condenser. The value of the operating temperature of the condenser was chosen to ensure a completely liquid flow; hence, the temperature to be cooled is 50°C or lower. Finally, there is the separation stage, which comprises various types of distillation columns. The agent remains in this stage until achieving the desired purity of the outlet streams, which is set at 99%.

Note that the masking technique does not fix or pre-establish any ordering or configuration of UOs; in fact, each time an episode starts, an empty flowsheet is initialized in the environment. This allows the agent to learn more effectively and avoid issues when simulating in ASPEN Plus® multiple environments, i.e., process flowsheets. Also, when an error happens during the simulation, the episode is terminated, and the agent is rewarded with a heavy penalty. Figure 3.7 presents the individual training curves for an unmasked and a mHPPO agent.

The insights derived from Figure 3.7 are multifaceted, with the most significant being the critical role of masking when simulating in advanced environments, such as the one presented in this case study. The training curve for the unmasked agent exhibits a pronounced decreasing trend, with the mean reward approaching -8, a penalty assigned to the agent for designing a failed CPF. In this scenario, the absence of an active masking function reveals that the agent learns to minimize the penalty as rapidly as possible, thereby enforcing failure at the earliest opportunity. In this case the agent opts to select a reactor that instantly fails when evaluated, due to not adhering to the specified conditions, i.e.,

a gaseous inlet flowrate at a designated temperature and pressure. Consequently, this results in single-step episodes characterized by direct failures, thus circumventing larger penalties. Furthermore, Figure 3.7A illustrates that certain flowsheets achieve a cumulative reward of zero; however, the agent's persistent failures and the limited representation of such runs in the experience buffer leads to the swift obsolescence of these experiences, which are replaced by failed runs.

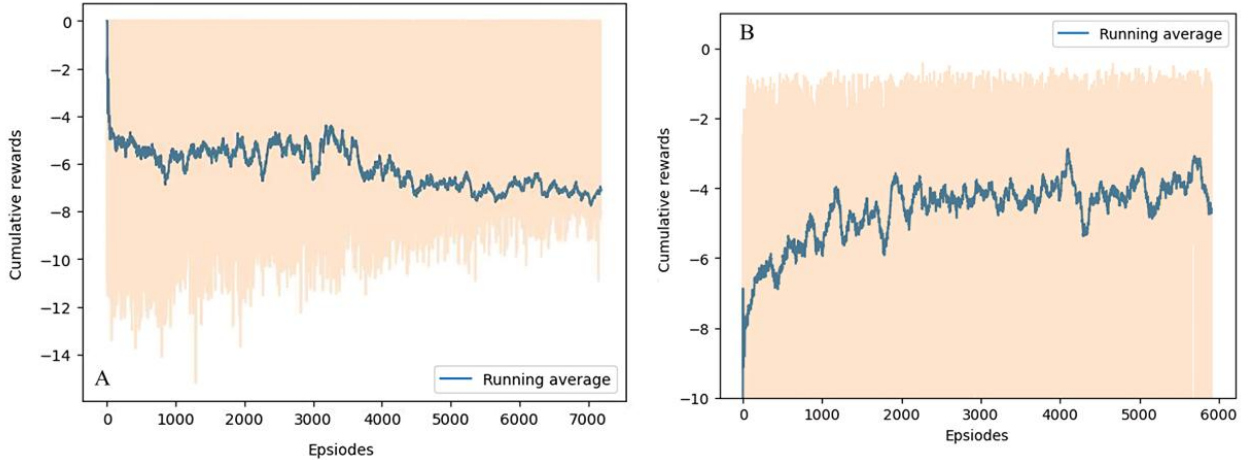


Figure 3.7: Individual learning curve for HPPO: A) unmasked B) masked

Figure 3.7B illustrates the learning curve of a masked agent. In this instance, despite significant variability in the obtained responses (indicated by the shaded orange area), the running average demonstrates a trend of growth and convergence towards a value of approximately -4. This cumulative reward not only indicates that the proposed flowsheet operates without failure in ASPEN Plus®, but also confirms that it is a completed flowsheet capable of meeting the design and operational conditions outlined in the case study. In contrast to Figure 3.7A, Figure 3.7B does not achieve cumulative rewards exceeding -2. However, when evaluating the best-performing agent, the cumulative rewards consistently approximate -2. This stands in stark contrast to the unmasked agent, where the evaluation of the best-performing agent yields a cumulative reward close to -8.

All the hidden layers in the mPPO and mHPPO agents consisted of 64 neurons. Both agents were trained ten times for 50,000 steps. The networks were updated every 512 steps with a discount factor of 0.99 and optimized using Adam (Kingma and Ba, 2017) optimizer with a starting learning rate of 2.5×10^{-4} which was adjusted throughout the training. The agents' exploration was encouraged by the inclusion of an entropy factor of 0.025, which progressively decayed at a rate of 0.9. These hyperparameters were obtained from preliminary testing with both agents.

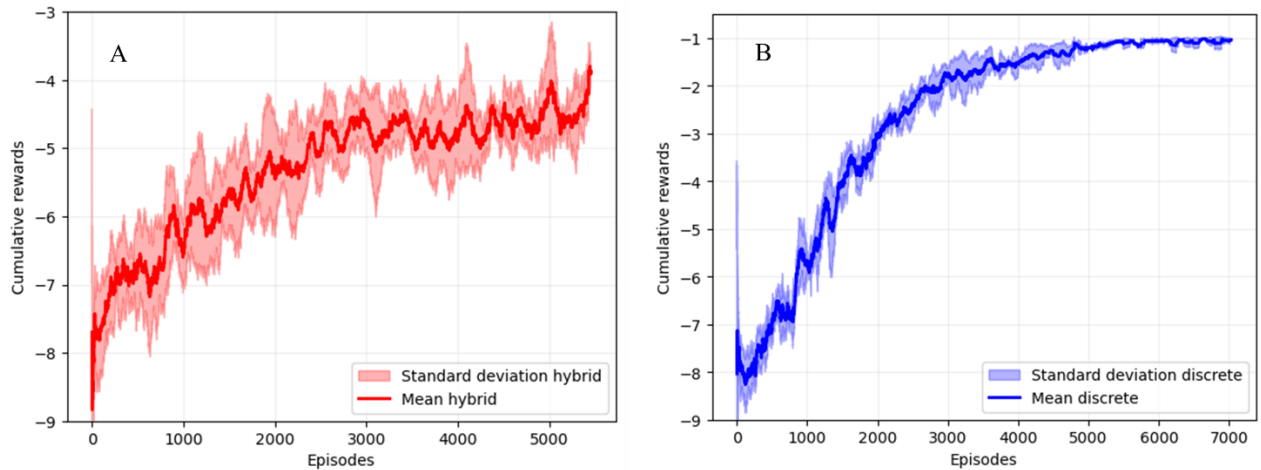


Figure 3.8: Average learning curve for 10 runs: A) mHPPO B) mPPO

Figure 3.8 presents the learning curves of both agents for 10 runs. The solid line represents the mean of the running means of the 10 runs, while the faded region represents the standard deviation around the different running means in those 10 runs. As in the previous case study, the mPPO achieves higher cumulative rewards with lower variability, whereas the mHPPO fluctuates around lower cumulative rewards with higher variability. Despite the mHPPO showing high variability in its learning (Figure 3.8A), when evaluating the mHPPO agents individually, the CPF obtained was the same. The only difference lied in the sizing of the UOs. The showcased variability arises from significant fluctuations in cumulative rewards during individual training sessions of mHPPO agent (Figure 3.7B). The source of these fluctuations arises from various factors. First, there's the inherent variability due to the implementation of continuous decision variables. Second, the interaction between continuous and discrete variables, which causes changes in the design and amount of UO present in the flowsheet, thus affecting the rewards. Third, the agent's difficulty in converging the flowsheet, particularly with units that can be considered as challenging. Operations such as distillation columns or reactors are often regarded as challenging units since the rigorous equations describing their behavior are highly non-linear models that are challenging to converge. Despite being well-specified, a discrepancy in the input flow or an imbalance in mass or energy balance can trigger warnings or convergence errors, resulting in simulation failures. These failures are attributed in the environment as significant penalties, thus prematurely ending episodes. On the other hand, the learning curve of the mPPO (Figure 3.8B) shows small to almost negligible variability in the last steps, suggesting that the 10 trained agents presented similar learning curves and converged to the same CPF. Note that the difference in episode lengths is precisely due to these variations in learning. While mHPPO exhibits extensive exploration, the length

of the designed CPFs may not be constant and may vary from one episode to another. On the other hand, once the mPPO finds a flowsheet that meets all the requirements, the agent starts to exploit that arrangement of UOs, keeping the number of units of the CPF constant and only varying the equipment sizing and operating conditions.

When evaluating the best-performing agents, the designed CPFs shown in Figure 3.9 and Table 3.7 demonstrate the ability to achieve the desired DME and water purities, maintain the reactor's operating temperature below the specified threshold, and yield an economically attractive process flowsheet.

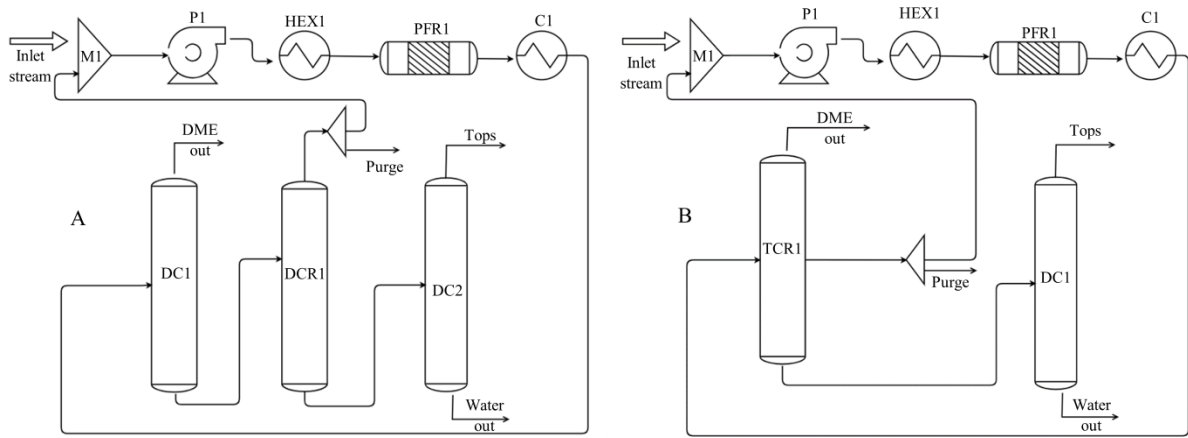


Figure 3.9: CPF obtained for case study 2: A) mHPPO B) mPPO

From the two CPF presented in Figure 3.9, it can be seen that the overall structure of both agents is similar. In fact, the first 5 UOs selected by both agents are the same, only varying in their sizing. The differences arise in the purification section, where the number and type of distillation columns differ. The mPPO opts for a TCR, where all the DME is distilled to the desired purity (i.e., 99%). Hence, the only task left for the agent is to distill the water using another column. Conversely, the mHPPO selects a series of columns that simultaneously distill both DME and water to their desired purities. The difference between the two agents lies not so much in the percentage of DME lost during the process, i.e., the percentage is quite similar, as shown in Table 3.7, but in the amount of MeOH recycled into the system. Despite achieving a lower cumulative reward, the mHPPO manages to reintegrate MeOH more effectively, allowing the agent to produce more DME than that obtained by the mPPO's CPF. For this case study, MeOH losses were not directly considered in the reward function, as the primary objective was to produce streams of DME and water that meet the desired specifications.

depicted in Table 3.7. However, its proper reintegration into the system is positively reflected in the sub-reward associated with the amount of DME distilled.

Table 3.7: UOs designs and specifications for case study 2

Unit operation	Design variable	mHPPO	mPPO
HEX1	T_{HEX} [°C]	277.2498	275.0000
PFR1	D^{PFR} [m]	2.0294	3.5000
	L^{PFR} [m]	9.2744	6.5000
C1	T_c [°C]	28.1489	50.0000
DC1	N	21	20
	$dist$ [kmol/hr]	84.0929	60.0000
DCR1	N	21	-
	$dist$ [kmol/hr]	39.7867	-
	rr	0.7295	-
DC2	N	21	-
	$dist$ [kmol/hr]	40.1090	-
	N	-	20
TCR1	$dist$ [kmol/hr]	-	80.0000
	$inter_{dist}$ [kmol/hr]	-	60.0000
	rr	-	0.8500
Operating reactor temperature [°C]		329.1773	314.6445
Output DME purity		0.9998	0.9999
Output water purity		0.9999	0.9999
DME wasted / DME produced [%]		5.0911	4.5352
Lost methanol [kmol/hr]		32.1410	39.8877
Cumulative reward		-2.0805	-1.1404

Note that a key insight achieved by the agents lies in their ability to acknowledge the need to include a recycle stream to improve the CPF. The recycle intricately connects various UOs and manages to yield a better outcome than without their consideration in the flowsheet. For instance, with the design specifications for the first distillation column, the mHPPO agent is unable to achieve the desired DME purity. Faced with this scenario, this agent can either mark the action as erroneous and adjust the specifications in the next episode, or, in the subsequent step, implement recycling to assess if it can

meet the expected design conditions. If the recycling option is not chosen, the alternative is to decrease the magnitude of the distillate rate. Given that DME is the most volatile compound, this action ensures that the distillation column only contains this compound in the overheads. However, the downside of taking this action is that the agent loses some of the converted DME to other streams and also forfeits the opportunity to convert more DME from the MeOH that is not recirculated. Accordingly, recycling allows the agent to achieve the desired purity while maintaining the initially proposed distillate rate.

To better understand the results produced by the best-performing agents, it is important to evaluate the contribution of each of the sub-rewards to the total cumulative reward. Table 3.8 presents the values of each sub-reward in a more granular form, providing deeper insights into their individual contributions. This analysis reveals the impact of each sub-reward on the cumulative reward, highlighting their role in guiding the agents toward economically and operationally efficient flowsheet designs.

Table 3.8. Cumulative reward breakdown

Sub-reward	mHPPO	mPPO
Driving force	0.7977	0.8019
UO capital cost	-3.1290	-2.5645
UO operating cost	-2.3489	-2.1425
Temperature constraint	0.4000	0.4000
DME produced	0.7996	0.7648
Design objective	0.0000	0.0000
Compact flowsheets	1.4000	1.6000
Total cumulative reward	-2.0805	-1.1404

The results presented in Table 3.8 highlight several key observations. Most notably, there are significant differences in capital and operational costs between the agents, with the mHPPO flowsheet having higher costs due to the inclusion of an additional UO. In terms of sub-reward contributions to the cumulative reward, the negative effect of the capital and operational cost are counterbalanced by the rest of sub-rewards. Note that determining the proportion in which a sub-reward affects the cumulative reward is complicated due to the difference in signs. However, it can be observed that the sub-reward with the most negative impact on the cumulative reward is the UO capital cost, while its counterpart is the reward granted for short flowsheets. Regarding the other sub-rewards, they show similar values for both agents, such as driving force, temperature constraints, and DME production.

However, mPPO presents a slightly higher reward for producing shorter flowsheets, which further contributes to its less negative total cumulative reward. Additionally, mHPPO achieves a slightly higher sub-reward for DME production, attributed to the previously mentioned effective MeOH mass integration into the system.

When comparing the flowsheets designed by the proposed framework with those presented in the works of Caballero et al. (Caballero et al., 2007) or Luyben (Luyben, 2017), it is evident that the results are somewhat similar. Note that the overall design, economic and operational objectives are the same; however, the approach used to arrive to the flowsheet design is different. Caballero et al. (Caballero et al., 2007) considered a superstructure approach, while Luyben (Luyben, 2017) fixed the arrangement of UOs, thus the goal was to size the units. In contrast, this study begins with an empty flowsheet and its goal is to propose the arrangement and design of UOs without any *a priori* knowledge of potential super-structures or CPF. Another point to consider is regarding the optimization methods; RL differs from conventional optimization techniques. Replicating the results reported in those studies using RL techniques may be quite challenging. As mentioned above, the agent aims to find a trade-off in its decisions, maximizing its impact on the flowsheet—specifically, obtaining 99%-pure water and DME streams—while minimizing operating costs. Translating results from traditional optimization methods to RL may not always seem the best option for the agent at a particular step and may even lead to their potential rejection. Note that the agent does not have access to the entire flowsheet at once but rather operation by operation, i.e., step by step when building the flowsheet. RL offers advantages in CPF design over conventional methods, with similar results validating its effectiveness. Among its various advantages, RL demonstrates flexibility and adaptability in handling complex systems without prior knowledge. It also capable to identify new processing routes and the ability to generate CPFs with limited information, serving as a starting point for more advanced studies. Additionally, its flexibility to interact with various simulation environments (process simulators) and advanced dynamic exploration techniques is a key feature that becomes critical in uncertain design contexts where input variables may not be known *a priori*. Nevertheless, an inherent limitation of the framework lies in the extensive training time. Training times could extend to more than 36 hours, posing a significant practical challenge and area for future improvement; particularly for larger and more complex case studies.

3.2.3 Case study 3

In contrast to the previous case studies, where CPFs generated by different masked agents were compared, this section aims to evaluate the proposed hybrid platform (Python-ASPEN Plus®) using a

new case study. Given the promising results obtained from case study 2 with mHPPO, it was decided to use the same agent. Note that the problem proposed is not as advanced as the one presented in Section 3.2.2, which is why the results from this section (Reynoso-Donzelli and Ricardez-Sandoval, 2024c) were not published alongside the previous two (Reynoso-Donzelli and Ricardez-Sandoval, 2024d).

The aim of this case study is to design a CPF for the production of propylene glycol from the reaction between propylene oxide and water (Equation 3.10), with the goal of achieving an outlet purity of 99%. The reaction is irreversible and occurs in the liquid phase, with methanol serving as the solvent. To prevent a runaway reaction, the reactor’s temperature must be maintained below 77°C.



For this case study, the Non-Random Two-Liquid (NRTL) model was selected as the thermodynamic package due to its suitability for systems operating at atmospheric pressure and entirely within the liquid phase. The UOs and their corresponding design variables are described in Table 3.9.

Table 3.9: UOs for case study 3

Unit operation	ASPEN Plus® block	Design variable	Bounds
Mixer (M)	<i>MIXER</i>	-	-
Cooler (C)	<i>HEATER</i>	Operating temperature (T_C)	5-15 [°C]
Plug flow reactor with constant cooling (PFR)	<i>RPLUG</i>	Length (L^{PFR})	5.0-8.5 [m]
		Diameter (D^{PFR})	1.0-3.5 [m]
Distillation column (DC)	<i>DSTWU</i>	Number of stages (N)	2-30
Distillation column with recycle (DCR)	<i>DSTWU</i> + <i>SPLITTER</i>	Number of stages (N)	2-30
		Recycle ratio (rr)	0-1

Note that more rigorous models, such as *RADFRAC* or *HeatX*, could have been employed in the current framework to achieve more accurate results and address multicomponent or multiphase situations. However, these models were not explored in this study to simplify the analysis.

The action space for this case study consisted of 5 discrete, $\mathbf{x}_d \in [M, C, PFR, DC, DCR]$ and 5 design actions – 4 continuous and 1 discrete variables – $\mathbf{x}_c \in [\mathbb{R}_{D^{PFR}}, \mathbb{R}_{L^{PFR}}, \mathbb{R}_N, \mathbb{R}_{rr}, \mathbb{R}_{T_C}]$. The

observation vector returned at each step contained information of the outlet stream of the selected UO, i.e., temperature, molar fractions (ω) of water (H_2O), propylene oxide (PO), propylene glycol (PG), and methanol ($MeOH$), and the current step; thus, $o_i = [T, \omega_{H_2O}, \omega_{PO}, \omega_{PG}, \omega_{MeOH}, i]$. The reward function considered in this case study is similar to that used for the previous case studies and follows a similar format for the sub-rewards, as shown in Table 3.10. As in the previous case study, all sub-rewards were normalized to prevent an individual term from having a disproportionate impact.

Table 3.10: Sub-rewards case study 3

Sub-reward	Reward or penalty	Assigned	Description
Driving force	Reward	Every step	Quantified by the conversion difference between the current and previous steps, its magnitude reflects the UO's effectiveness in producing propylene oxide.
UO total costs	Penalty	Every step	Composed of the capital and operational costs of the UO. The capital costs of each UO were calculated using functions derived from Douglas (Douglas, 1988).
Process design requirement	Penalty	Final step	Ensures that the desired purity of propylene glycol is reached. Magnitude depends on the gap of the achieved purities and desired purities.
Specific design constraint	Reward	Specific step	Constant reward granted to the agent when the UO is a reactor and meets the design specification (operating below 77°C). No penalty is awarded if this constraint is not met.
Short flowsheets	Reward	Final step	Denoted as the difference between the maximum number of steps I (which for this case was 10) and i .

For this case study masking was applied to 3 UOs: DC, DCR and C. For DC a straightforward masking mechanism was implemented, once a substantial amount of reactant was converted into propylene glycol, the UO was unmasked. This was implemented to prevent the agent from selecting short sequences of UOs that would result in reactant wastage, particularly given that this column lacked recycling capabilities. Similarly, masking was applied to the cooler selection process; once a C was chosen, another could not be selected consecutively. This restriction was designed to prevent the agent from placing coolers in series without a defined purpose, thereby optimizing the efficiency of the process design. For the DCR, a masking strategy similar to that applied in the first case study was implemented. In this instance, instead of a M – FT pair, a M – DCR pair was considered. The UO was

unmasked once some reactant was converted, and a mixer was present in the CPF. To prevent reactant wastage in the distillate stream, the concentration of reactant in the purge stream (the complementary stream to the recycle) needed to reach a specific threshold to conclude the flowsheet; otherwise, it would have been used to further extend the flowsheet.

Same as the past case studies, all the hidden layers of the mHPPO agent consisted of 64 neurons and the agent was trained for 50,000 steps. The networks were updated every 512 steps with a discount factor of 0.99 and optimized using Adam optimizer (Kingma and Ba, 2017) with a starting learning rate of 2.5×10^{-4} which was adjusted throughout the training. The agent's exploration was encouraged by the inclusion of an entropy factor of 0.025, which progressively decayed at a rate of 0.9. These hyperparameters were obtained from preliminary testing with both agents.

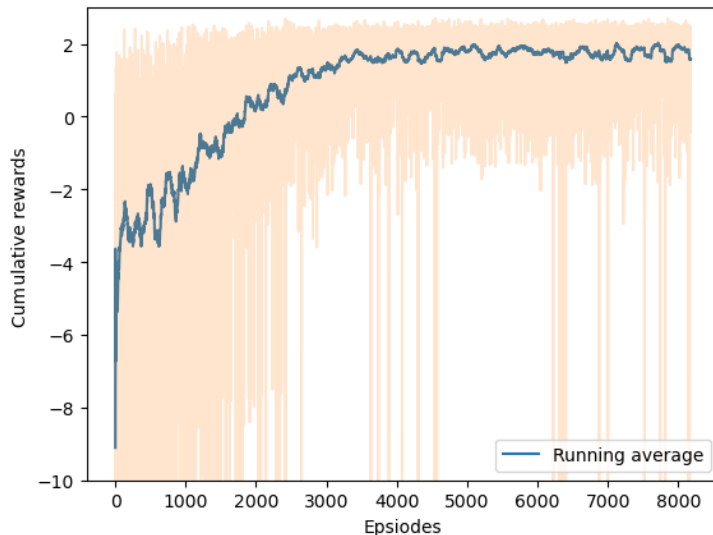


Figure 3.10: Individual learning curve for mHPPO case study 3

Figure 3.10 presents the individual learning curve of the mHPPO agent used in this case study. The solid blue line, representing the running mean, illustrates the trend of cumulative rewards throughout the learning process, while the faded orange region highlights the cumulative rewards obtained at the end of each episode. The behavior of the running mean in this case exhibits an exponential growth during the initial 4,000 episodes, followed by a plateau as it approaches a cumulative reward close to 2. In contrast to the learning curves observed for the mHPPO agents in previous case studies (Figures 3.4A and 3.7B), the plateau reached by the running average in Figure 3.10 closely corresponds to the maximum rewards achieved by the agent during training, indicating that the UO arrangement identified is the most reward-efficient configuration the agent could find. The

slight deviation from the maximum rewards represents the margin of error in the continuous component of the hybrid agent, responsible for determining the design of the UOs.

When evaluating the best-performing agents, the designed CPFs shown in Figure 3.11 and Table 3.11 demonstrate the ability to achieve the desired propylene glycol purity, maintain the reactor's operating temperature below the specified threshold, and yield an economically attractive process flowsheet.

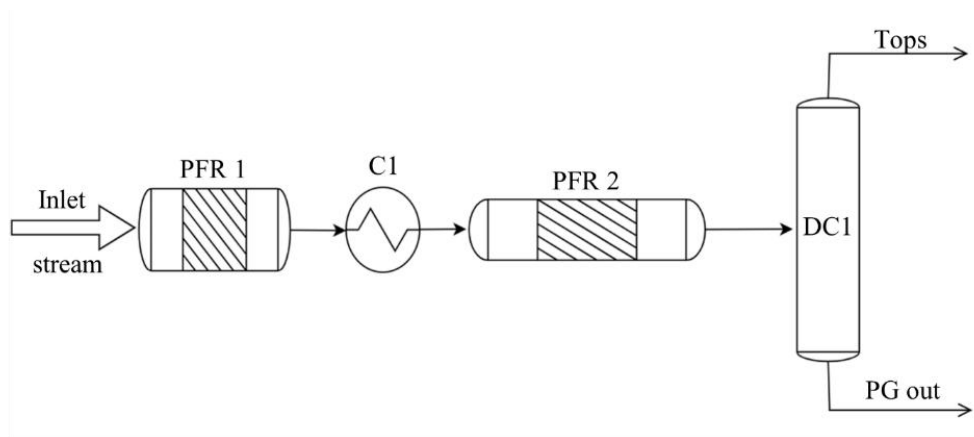


Figure 3.11: CPF obtained for case study 3

As shown in Figure 3.11, the flowsheet identified by the agent did not require recycling. This decision is reasonable because, unlike the previous case study, the reaction in question is irreversible. Additionally, the cost associated with the distillation column is significantly higher than that of the reactor, making a configuration similar to that of case study 1 impractical. Thus, the agent chose to fully convert the reactant first, meeting the reactor design constraints, operating the reactors below 77°C, before proceeding with the separation process.

A detailed analysis of the flowsheet revealed that the agent strategically maximized reactant conversion in the early stages. To comply with design constraints, the agent selected two medium-sized reactors and incorporated a cooler between them. Notably, had the agent not detected the need for the cooler, the inlet temperature of the second reactor would have reached around 75°C, likely leading to a violation of the temperature constraint.

Table 3.11: UOs design and specifications for case study 3

Unit operation	Design variable	Value
PFR1	D^{PFR} [m]	2.9578
	L^{PFR} [m]	5.6003
C1	T_c [°C]	9.6213
PFR2	D^{PFR} [m]	1.5907
	L^{PFR} [m]	7.2016
DC1	N	13
PFR1 operating temperature [°C]		75.6421
PFR2 operating temperature [°C]		69.0674
Output propylene glycol purity		0.9999
Cumulative reward		2.1785

Preliminary experiments had shown that all reactants could be converted using a single large reactor, but this approach exceeded the temperature limits, necessitating the use of multiple reactors and a cooler to maintain operational boundaries. A distillation column of 13 stages is chosen at the end to obtain the desired product with a purity of 99% in the bottoms. Note that tests were conducted with the inclusion of a recycling column. However, the rewards obtained under those conditions were lower, e.g., the agent needed more than just a single column, the reactors were not able to meet the constraints, or the simulation failed. One notable aspect from the learning curve is the significant reduction in reward variability. Unlike the previous case studies, once the agent found the optimal configuration, it focused on exploring the continuous part while exploiting the discrete component. To prevent flowsheets that had runs with warnings or errors, a penalty of -10 was assigned to the agent when faced with these issues. Note that as the agent learned, such flowsheets were mitigated, with only a few remaining in the later stages of learning.

As in the previous case study, a significant limitation of this framework is the extended computation time required to achieve results. In this case, approximately one day was needed to train the agent. Techniques such as early stopping – where the CPF configuration is observed at a certain point during training and compared with previously proposed designs to assess how different they are – could be applied to reduce training time. In this particular case, such a technique could have been beneficial, as the agent's training might have been halted around episode 4,500 based on the trend of the learning curve shown in Figure 3.11. However, the proposed framework offers significant

advantages. It closely aligns with how CPFs are typically designed in the industry, automating the entire process. Additionally, it is flexible enough to connect with other simulation suites that might run faster than ASPEN Plus®, providing an alternative to model-based optimization methods that require detailed equations. Furthermore, once the agent is trained, its evaluation is extremely fast, significantly reducing computational time for subsequent tasks.

3.3 Summary

This chapter introduced a novel RL methodology that was capable of designing and optimizing steady-state CPFs starting from a given inlet flowrate and a predefined set of UOs. The first section of the chapter introduces the solution framework, emphasizing in the various components of the RL framework and the application of the masking functionality in each of them. In this section, the two innovative agents developed – mPPO and mHPPO – were introduced, detailing their architecture and the pertinent modifications made to facilitate the use of masking. The remaining components of the RL framework were also described: the action space for both discrete and hybrid environments; the observation vector; and the reward function, which outlined the different components that comprise it and enable the capture of design objectives and constraints of the problem. The proposed framework was applied to three case studies; the first two compared the performance of the proposed agents. The first case study involved a relatively simple problem, which facilitated an understanding of the masking functionality and the learning processes of both the agents. When the problem involved fewer discrete and continuous variables, and the discretization of continuous variables was straightforward, mPPO demonstrated significantly better performance than mHPPO. The rewards achieved by mPPO aligned with the maximum possible rewards, and due to the restricted search (combinatorial) space, it was able to find the optimal configuration. Part of the observed difference in the agents' responses can be attributed to their learning processes. Notably, the learning process of the mHPPO was characterized by a trade-off between discrete and continuous losses, wherein minimizing one type often led to an increase in the other. This dynamic interplay underscores that the agent's decision-making is governed by the necessity to balance both types of variables, ultimately striving for an optimal configuration. The second case study incorporated a hybrid platform (Python-ASPEN Plus®) to address a more realistic scenario. In this case, the number of discrete and continuous variables was significantly large. Despite having an infinite search space, mHPPO outperformed mPPO. Even though it yielded lower rewards, the generated CPFs not only met all specified requirements but also exhibited novel arrangements to attain additional benefits, such as minimizing reactant losses or enhancing flow reintegration more

effectively. An inherent limitation of the framework lies in the extensive training time, particularly when the RL framework is connected to an external chemical simulation software such as ASPEN Plus® to simulate the environment. Training times could extend to more than 36 hours under these conditions, posing a significant practical challenge and area for future improvement; particularly for large and more complex case studies. On a positive side, the CPFs generated through this methodology provide results that are sufficiently robust to meet primary objectives and serve as a solid starting point when seeking to optimize a process flowsheet. Similarly, when dealing with a process whose flowsheet is unknown, this technique is ideal for initiating the outline of UOs' arrangements and gaining insights on potential (attractive) configurations. Once a RL-agent designs a CPF, this can be used as a starting point for the optimization process and refine its performance. A third case study was addressed using mHPPO coupled with the hybrid platform to assess the agent's performance in a new scenario, yielding results that aligned with the specifications and proposed an economically attractive outcome. This case study further confirmed the advantages and disadvantages of the methodology. On one hand, it highlighted the extended training periods required to train the agent when the methodology was integrated with a hybrid platform. On the other hand, it demonstrated the significant benefit of employing this methodology in designing CPFs that closely mirrors real-life scenarios, providing an initial or general understanding of how a process might be arranged.

Chapter 4

An integrated Reinforcement Learning framework for simultaneous generation, design, and control of chemical process flowsheets

Chapter 3 introduced a methodology for generating, designing, and optimizing steady-state CPFs. Nevertheless, most chemical engineering processes are inherently dynamic, requiring more advanced approaches to account for their transient nature. As discussed in Chapter 2, extensive research addresses the integrated design and control problem, particularly through model-based optimization methods. However, these approaches are typically limited to case studies with few UOs, relying on non-rigorous models due to high computational costs and inherent mathematical and computational complexities. Recently, model-free optimization methods have emerged as an alternative to solve various research problems. RL stands out as an attractive framework for tackling the integrated design and control problem, particularly for its ability to take sequential decisions that are interrelated, while optimizing complex processes. However, few studies have explored this area (Sachio et al., 2022; Mendiola-Rodriguez and Ricardez-Sandoval, 2023). Additionally, surrogate models, particularly NNs, have demonstrated their capacity to approximate rigorous UO models, thereby significantly reducing the number of variables and mitigating computational costs (Schweidtmann et al., 2019). While integrating NNs into RL frameworks is feasible, this has yet to be fully explored in the context of the integrated design and control problem. Recognizing these research opportunities, this study introduces a framework capable of simultaneously generating, designing, and controlling CPF using RL. The RL agent is tasked with optimizing a user-defined objective function – comprising both economic and dynamic components, such as disturbance rejection and tracking errors – while adhering to process and equipment design and operational constraints. These objectives and constraints are enforced through a reward shaping strategy. The environment in which the RL agent operates is composed of various NN surrogate models, which approximate the rigorous dynamic models that simulate the closed-loop behavior of the UOs. The major outcomes of this chapter have been submitted for publication (Reynoso-Donzelli and Ricardez-Sandoval, 2024e) and presented at the AIChE Annual Meeting 2024 (Reynoso-Donzelli and Ricardez-Sandoval, 2024b).

This chapter is structured as follows: Section 4.1 outlines the proposed methodology, emphasizing the approach employed to capture the dynamic behavior of UOs through a series of representative data used to generate surrogate models. Section 4.2 elaborates on the implementation of

this methodology via three case studies, two of which involve multiple UOs. The chapter concludes with a summary in Section 4.3.

4.1 Solution framework

Solving a MIDO problem like that depicted in Equation 2.6 is inherently challenging due to the combination of discrete and continuous variables alongside system dynamics described by DAEs. The interplay between these elements results in a large, complex search space that is difficult to explore efficiently. Furthermore, the problem must satisfy both time-independent constraints – such as design or operational limits – and account for the dynamic behavior of the system over time. Balancing these conflicting requirements while identifying feasible and optimal solutions often requires sophisticated methodologies and significant computational resources.

This section presents a new methodology that addresses the integrated problem of generation, design, and control using RL. Figure 4.1 schematically illustrates the interaction among the various components of the proposed RL framework. In this context, the agent interacts with the environment, primarily composed of surrogate models, through the exchange of rewards, observations, and actions.

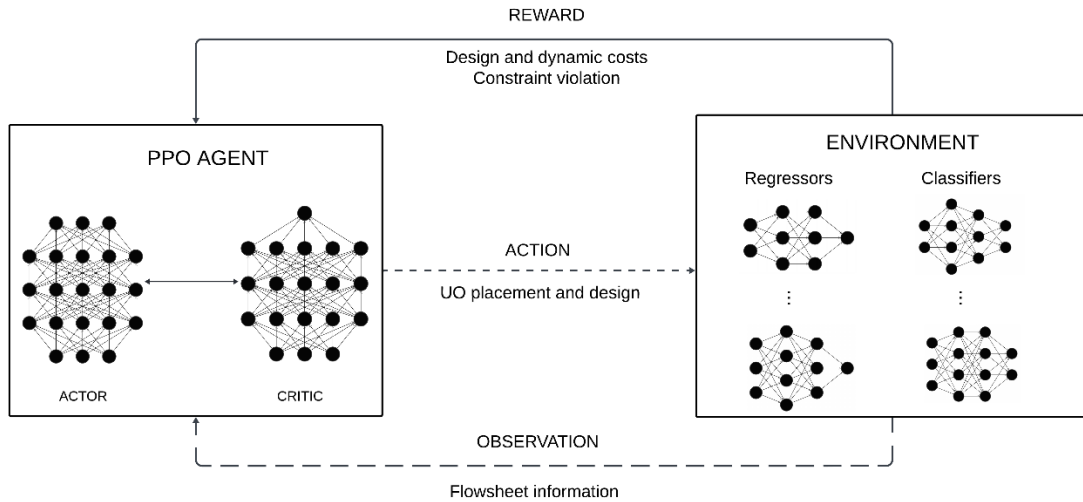


Figure 4.1: RL diagram

The solution framework section is structured as follows: Section 4.1.1 describes the methodology employed to construct surrogate models, which capture the behavior of key process variables, dynamic performance metrics, and constraint violations of the UOs. Following this, Section

4.1.2 outlines the constituent elements of the RL environment, including the action space, observation vector, and reward function. Section 4.1.3 provides a description of the PPO agent utilized in this study, highlighting the key features of its architecture and modifications made relative to the original version (Schulman et al., 2017).

4.1.1 Surrogate models

One of the core components of this study is the use of surrogate models embedded within the RL environment. The primary objective of these models is to approximate the closed-loop transient operation of UOs through descriptive metrics that are used to evaluate their potential to be integrated into the final process flowsheet. The use of surrogate models aims to accelerate the RL agent's learning by providing fast predictions of key output variables. Although mechanistic models could theoretically be used within the RL environment, this study demonstrates that surrogate models have the potential to not only replace mechanistic models but also serve as efficient alternatives to costly plant data and complex dynamic simulators. A surrogate model can range from a simple linear model to a complex NN, the latter being the most commonly used due to its high levels of achievable accuracy. The use of NNs in this study was driven by their ability to model complex, nonlinear relationships with high accuracy, while also offering practical advantages in terms of flexibility, using similar NNs' architectures for different regression problems, compared to other surrogate modeling techniques. A surrogate model can range from a simple linear model to a complex NN, the latter being the most commonly used due to its high levels of achievable accuracy. In the RL environment, surrogate models act like any analytical function; they take input variables and return metrics or performance values that are used within the environment to assess the performance of the proposed UO for subsequent calculations or to design the reward function. The use of surrogate models is fundamental, as it allows to compress all the information from a transient process into a series of individual descriptive values, e.g., the Integral Squared Error (ISE) or the outlet temperature at the final integration time. This approach contrasts with conventional RL applications, where the agent functions as a controller, continuously adjusting actions based on real-time system feedback. Instead of directly controlling the process, the agent in this framework determines the type and position of the UOs in the CPF, defines the sizing of the chosen UOs and determines the controller tuning parameters – making the control process inherently offline. To ensure accurate surrogate model training, the data must comprehensively represent the full dynamic response of the UO or system, enabling the extraction of key descriptive values. To build these surrogate models (NNs), two steps were followed: data generation and mechanistic model approximation. These steps are explained next.

4.1.1.1 Data generation

The objective in data generation is to create a matrix for each UO ($\mathbf{M}_{UO}$), as the input variables differ between different UOs. The generated matrix consists of m different vectors, denoted as $\mathbf{z}^m$, each containing a realization of the n input variables. Hence, $\mathbf{M}_{UO}$ has dimensions of $m \times n$, i.e.,

$$\mathbf{M}_{UO} = \begin{pmatrix} z_1^1 & \cdots & z_n^1 \\ \vdots & \ddots & \vdots \\ z_1^m & \cdots & z_n^m \end{pmatrix} \quad (4.1)$$

This matrix considers the essential input data required for evaluation within the mechanistic model, which is then utilized to train the surrogate model. The vector $\mathbf{z}^m$ comprises input variables provided by the feed stream to the UO, such as temperature or concentration ($\mathbf{x}(t)$); design variables specific to the UO, e.g., reactor diameter ($\boldsymbol{\eta}$) or the number of stages in a distillation column ($\boldsymbol{\gamma}$); as well as variables related to the control system of the UO, e.g., set-point values or controllers' tuning parameters.

Regardless of the UO, when designing $\mathbf{M}_{UO}$, it is essential to consider that the feed stream to the UO can be situated at any point within the CPF. For the design variables, the limits are specified by their lower and upper bounds, as it is performed in an optimization problem. For the design variables, the limits are specified by their lower and upper bounds. For the inlet stream variables, the bounds are determined in two ways: 1) CPF design or operational goals, e.g., if the goal is to achieve a mole fraction of less than or equal to 0.99, the lower bound for this variable would be 0.99, while the upper bound would be the mole fraction at the stream's entry point. 2) Design or operational constraints, e.g., if the system must operate below 10 bar, the upper bound for the inlet pressure to a unit operation cannot exceed this limit. All these bound estimations are made *a priori*, implying that the upper and lower bounds of $\mathbf{z}^m$ are assumed to be known beforehand. Note that during these estimations, the arrangement of UOs within the CPF is yet to be determined.

There are various methods by which matrix $\mathbf{M}_{UO}$ can be populated, e.g., enumeration or random point selection (Kimaev and Ricardez-Sandoval, 2017). In this work, we employed Latin Hypercube Sampling (LHS) as the sampling method. This method was deemed superior for training surrogate models due to its ability to efficiently cover the multidimensional input space while ensuring a representative sampling of the parameter ranges. This method minimizes redundancy and maximizes the diversity of input combinations, thereby enhancing the accuracy and robustness of the surrogate models trained on the generated data (Song and Kawai, 2023). Note that other sampling techniques can

be employed to populate matrix $\mathbf{M}_{UO}$. Also, it is key to verify that the generated combinations of $\mathbf{z}^m$ are logical and exclude those that could detrimentally skew the agent's learning process. To achieve this, a preliminary data analysis is conducted, discarding any combination that exhibits illogical values. For instance, in a conversion process, an illogical combination might be one where the proposed set-point concentration exceeds the proposed input concentration.

4.1.1.2 Mechanistic model approximations

Mechanistic models are predefined representations of a UO that can be expressed through a series of DAEs that describe the conservation balance equations of that UO, an advanced chemical engineering simulator, or a dataset. In this work, mechanistic models are expressed through the mathematical framework of a UO in a dynamic state (DAEs). Note that each UO is equipped with an embedded control system. These models typically accept input variables (stored in $\mathbf{z}^m$) and produce a series of time dependent output variables ($\mathbf{y}(t) \in \mathbb{R}^{n_y}$). Due to the inability of surrogate models to fully replicate the dynamic responses of the mechanistic models, for each output variable a surrogate model is designed to approximate a point in time of these dynamic responses; for instance, we denote the output response at the endpoint as $y(t_f)$. Note that advanced data-driven surrogate models, e.g., Physics-Informed Neural Networks (PINNs), have the ability to capture more accurately the entire dynamic response (Chaffart et al., 2024). The use of such surrogate models is beyond the scope of this work.

In addition to predicting these specific time points in the output variables, surrogate models are also employed to forecast dynamic performance metrics such as the ISE (Equation 4.2) or Integral of Absolute Error (IAE), as well as to provide a label indicating whether the proposed process flowsheet configuration violates design and operational constraints. Both the dynamic performance metrics and the labels are dependent on the output variables $\mathbf{y}(t)$ specified for every UO.

$$ISE = \int_{t_0}^{t_f} (\mathbf{y}(t) - \mathbf{y}_{sp})^2 dt \quad (4.2)$$

The labels and their formulation will be explained in greater detail later in this section when discussing the classifiers. As discussed above, for each UO, multiple surrogate models are employed. This approach ensures a more accurate description of the model's behavior and enhances the predictive performance of the surrogate models.

Figure 4.2 schematically illustrates the process of constructing the database that is subsequently used to train the surrogate models. In this case, the mechanistic model of the UO is simulated with each $\mathbf{z}^m$ realization included in the matrix $\mathbf{M}_{UO}$. For each combination in $\mathbf{M}_{UO}$ (i.e., for each $\mathbf{z}^m$ realization), the response consists of output variables $\mathbf{y}(t)$ – whose endpoint ($y(t_f)$) is chosen –, a dynamic performance metric – in this case ISE – and the classifier label. The responses for each of the m combinations are stored in a matrix called $\mathbf{M}_{Resp}$, which, together with $\mathbf{M}_{UO}$, will later be used to train the surrogate models.

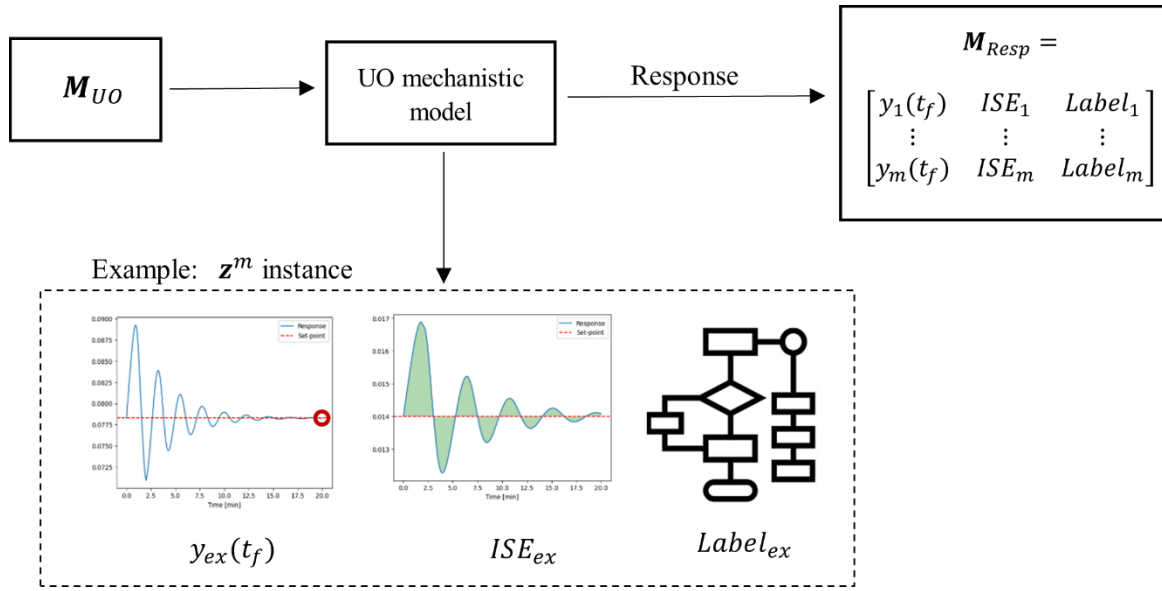


Figure 4.2: Dataset building

An illustrative example of how the values for $y(t_f)$, ISE , and $Label$ are obtained from a $\mathbf{z}^m$ instance is presented next. As shown in Figure 4.2, for $y_{ex}(t_f)$ the final value is located and recorded (red circle); for ISE_{ex} to obtain the value the integral of the response curve is calculated; and for $Label_{ex}$ a series of logical tests are evaluated to assign the corresponding value. After building the dataset, a NN surrogate model was fitted for each of the responses, i.e., $y(t_f)$, ISE , and $Label$. Given the differences in the responses, both in terms of magnitude and nature (continuous vs. Boolean), it was determined that training the networks individually provided a more accurate fit for each output response. In this case, the Mean Squared Error (MSE) was used as the loss function to train the surrogate model. Moreover, Adam's optimizer (Kingma and Ba, 2017) was employed to compute the gradients and update the network weights. Each type of surrogate models, i.e., endpoint regressors, dynamic performance metrics regressors and label classifiers, are described next.

The objective of endpoint regressors is to accurately predict the endpoint value of a dynamic response ($y(t_f)$), e.g., the outlet temperature or concentration from a system at the final integration time. The endpoint is selected because the system is expected to operate at steady state. Note that other intermediate points could also be considered within the present framework. Mathematically, the endpoint regressor can be expressed as $\widehat{y(t_f)} = NN_{EP}(\mathbf{z}^m)$, where a neural network (denoted as NN_{EP}) takes an instance of $\mathbf{z}^m$ as input and regresses the endpoint of an output variable. In contrast, dynamic performance metric regressors, predict the model's response accuracy relative to control variables, typically employing metrics such as ISE or IAE. To enhance predictive precision, the regressor computes the natural logarithm of these metrics rather than their nominal values. This transformation aids in distinguishing values more effectively, especially when ISE or IAE values are near zero. This regressor is represented as $\ln(\widehat{DM}) = NN_{DP}(\mathbf{z}^m)$, where the neural network NN_{DP} takes an instance of $\mathbf{z}^m$ and predicts the natural logarithm of a dynamic metric (denoted as DM). Unlike the regressors discussed above, the classifier exhibits a more complex structure. In this study, the classifier aims to determine whether the proposed UO violates design or operational constraints while maintaining the operation on target in closed-loop. Mathematically, the classifier can be represented as $\widehat{Label} = NN_C(\mathbf{z}^m)$, where the label is predicted through NN_C which takes an instance of $\mathbf{z}^m$. The method for assessing compliance with constraints and ensuring controlled operation involves various logical tests depicted in Algorithm 4.1.

Algorithm 4.1 Pseudocode for label classification

```

if  $|y(t_f) - y_{sp}| \leq \epsilon_1$  then
  if  $\frac{1}{r} \sum_{j=1}^r \left( \frac{dy}{dt} \right)_j \leq \epsilon_2$  then
    if  $cons \leq \epsilon_3$  then
       $pass = 1$ 
    else
       $pass = 0$ 
    end if
  else
     $pass = 0$ 
  end if
else
   $pass = 0$ 
end if

```

As illustrated in Algorithm 4.1, the first logical test aims to determine whether the distance between the endpoint predicted by the mechanistic model for an output variable ($y(t_f)$) and its corresponding set-point proposed by the design (y_{sp}) exceeds a specified threshold (ϵ_1). Note that the

value of y_{sp} is an input variable that needs to be specified in $\mathbf{z}^m$. In this case, even if the endpoint regressor has high accuracy in predicting $y(t_f)$, if the value does not fall within the established threshold, the design is labeled as unsatisfactory. If this logical test holds, the next step involves verifying that the system is operating near their desired operating conditions. In this work, a UO is assumed to be working at their target condition if the averaged gradient of the last r time-points fall within a specified threshold (ϵ_2). The objective of this metric is to prevent the agent from selecting designs that are far from their desired target conditions, even if they exhibit values of ISE or IAE smaller than the unit. Having confirmed that the system operates near the established set-points, the next step is to examine whether the proposed UO complies with design, logical or operational constraints, i.e., $cons \leq \epsilon_3$. Depending on the nature of the constraint, e.g., dynamic path or end-point constraints, $cons$ can be estimated in various ways. To capture a more general framework, i.e., dynamic path constraint violations, a structure similar to the ISE (Equation 4.2) is employed, i.e.,

$$g(\mathbf{x}(t), \dot{\mathbf{x}}(t), \mathbf{w}(t), \boldsymbol{\eta}, \boldsymbol{\kappa}(t), \boldsymbol{\gamma}, \mathbf{u}(t), \mathbf{d}(t), t) \leq g_{cons} \quad (4.3a)$$

$$\boldsymbol{\vartheta}(t) = \begin{cases} g(t) - g_{cons}, & g(t) \geq g_{cons} \\ 0, & g(t) < g_{cons} \end{cases} \quad (4.3b)$$

$$cons = \int_{t_0}^{t_f} \boldsymbol{\vartheta}(t)^2 dt \quad (4.3c)$$

where g_{cons} represents the input constraint limit, e.g., maximum allowed temperature or liquid level in a UO. Note that $g(t)$ can take different forms, i.e., it may be expressed as a single function of an output variable $y(t)$, or as a function or a combination of multiple design and control variables, including output variables, $\mathbf{y}(t)$. Regardless of the form, for each point in the time domain, the difference between $g(t)$ and g_{cons} needs to be computed. As shown in Equation 4.3, an auxiliary vector ($\boldsymbol{\vartheta}$) is employed to collect the deviations between $g(t)$ and g_{cons} . If the difference is positive, indicating that the constraint was violated at time t , the value of this difference is assigned to $\boldsymbol{\vartheta}$; otherwise, a value of 0 is assigned to $\boldsymbol{\vartheta}$ (i.e., constraint is satisfied). Accordingly, $cons$ is estimated as shown in Equation 4.3. Due to potential class imbalance, where violation instances are typically less frequent, Synthetic Minority Over-sampling Technique (SMOTE) is applied to balance the dataset (Chawla et al., 2002), thus improving the predictive power of the classifier and preventing it from naïvely choosing the overrepresented class.

4.1.2 Environment

As illustrated in Figure 4.1, the RL environment primarily consists of surrogate models. However, it is important to note that not all surrogate models are used at each step; only those required for the selected UO are activated and utilized. During this process, the agent interacts with the RL environment—primarily composed of surrogate models—learning to place, design, and control the UOs within the CPF. During this process, the RL agent aims to determine the sequence, type, and number of UOs present in the CPF. Initially, the agent may propose infeasible CPFs as it explores the environment and learns the relationships between UOs and process constraints. Through feedback from the environment, e.g., penalties due to constraint violation, the agent progressively refines its decisions. During training, the agent learns to build CFP that are not only feasible but that can also improve the reward function by ensuring that each UO's outputs serve as appropriate inputs for subsequent steps, adhering to overall CPF goals and requirements. While building the CPF, the agent also takes into consideration the sizing of the selected UOs (i.e., design), and the tuning parameters of the integrated control system embedded to the UOs (i.e., control). The selection of these three activities is performed simultaneously by the agent. To the best of the authors' knowledge, no previous work has addressed the fully integrated problem of flowsheet generation, design, and control using RL, which is a key novel aspect in our research. This section aims to provide a more detailed description of the components through which the environment exchanges information with the agent: the action space, the observation vector, and the reward function.

4.1.2.1 Action space

In this work, the action space ($\mathbf{A}$) encompasses all decisions (both discrete and continuous) related to the design and control of all UOs present in the problem, i.e.,

$$\mathbf{A} = [\mathbf{x}_d, \mathbf{x}_c] \quad (4.4)$$

The discrete variables, stored in the vector $\mathbf{x}_d \in \mathbb{Z}$, can be divided into two categories. The first category is concerned with the selection of a UO. Unlike model-based approaches, where the arrangement of UOs or a superstructure defines the potential connections between them, the RL agent only has knowledge of the available UOs without specific details about their placement. These operations are stored in a "UOs' pool," which assigns an integer value to each operation, allowing the agent to distinguish between them and make selections accordingly. The second category involves all the discrete designed variables, denoted as $\boldsymbol{\gamma}$ in Equation 2.6; e.g., the number of stages in a distillation column, or the specific feed or reflux stages. Continuous variables, stored in the vector $\mathbf{x}_c$, represent

all the design and control variables for the UOs considered in the system. For example, the diameter of a distillation column, the tuning parameters of a PI controller that maintains in closed-loop the boil-up ratio of a column, or the liquid hold-up of a reactor. The discrete design variables are approximated using continuous distributions, as seen in other RL works that generate and design CPF (Midgley, 2020; van Kalmthout et al., 2022). While this approach improves computational efficiency and makes the analysis straightforward, a potential downside is that the continuous probability distributions may not fully capture the trend of the original discrete distribution, leading to approximation errors and a loss of detail. Nonetheless, this method simplifies the model and eliminates the need for a multi-discrete agent, which falls outside the scope of this work.

Moreover, coupled with the action space, the environment incorporates a series of functions that interpolate values sampled from the probability distributions, into the actual domain of those actions. Note that the action space consists of individual probability distributions rather than a combination of different continuous and discrete actions. In RL, when the agent interacts with the environment, it receives information and takes an action based on that information. The selection of the action at each step i (i.e., a_i) involves sampling values from predefined probability distributions, as detailed in the agent sub-section (Petrizzini and Antonelo, 2021).

4.1.2.2 Observation vector

The observation vector (o_i) for this methodology is consistent with that utilized in the previous chapter (Equation 3.4). The overall structure of the vector is contingent upon the specific problem setting; however, it generally includes the following components: operating variables of the process (such as temperature, pressure, and total flow), a tracker for chemical components (reactants or products), and the current observation step. As in the previous methodology, the elements in o_i are normalized to prevent larger values from disproportionately influencing data processing within the agent's neural network.

4.1.2.3 Reward function

Depending on the RL agent, the type of reward can be sparse, meaning all time steps have a reward of zero except for the final step, or non-sparse, where each step has a reward. PPO agents have demonstrated more effective learning when dealing with non-sparse rewards compared to sparse rewards (Yu, 2020). Therefore, a non-sparse reward system was designed to address this problem. For complex problems, such as the integrated generation, design and control problem discussed in Chapter

2.3, the step reward in a non-sparse system must incorporate various elements. These elements enable the agent to consistently avoid violating constraints (e.g., operating below a certain temperature) and move towards its primary objective (e.g., achieving the desired reactant conversion at the lowest possible cost). This approach is known as reward shaping, as illustrated in Equation 3.5.

Note that the sub-reward expressions are unique for each UO; however, a common structure is followed, where r_i is primarily composed of three terms. 1) Capital cost (r_{CC}): This sub-reward aims to quantify the cost related to the UO and process design variables chosen by the agent. Consequently, this sub-reward is expressed as a function of variables such as $\boldsymbol{\gamma}$ and $\boldsymbol{\eta}$ or variables regressed by endpoint output regressors, $r_{CC} = f(\boldsymbol{\gamma}, \boldsymbol{\eta}, \widehat{y(t_f)})$. 2) Dynamic cost (r_{DC}): This sub-reward aims to quantify the cost associated with the dynamic performance of the UO. The value of the sub-reward is determined based on the output from the surrogate model responsible for regressing the dynamic performance metric, $r_{DC} = f(\ln(\widehat{DM}))$. 3) Constraint violation cost (r_{CV}): This sub-reward is designed to indicate whether the design and control proposed by the agent may lead to a constraint violation. The value of this sub-reward is determined by the classifier, where based on the output (Boolean value), the agent receives a penalty of a fixed amount, $r_{CV} = f(\widehat{Label})$. Unlike the previous sub-rewards, at the final step, the agent receives an additional penalty if the process flowsheet proposed by the agent fails to achieve the expected design objective (r_{FC}), e.g., reach a desired reactant conversion or product purity. More details about the design of rewards are provided in the case studies.

4.1.2.4 Agent

PPO operates as an actor-critic algorithm, where the actor selects actions based on the current state, and the critic estimates action values. Both actor and critic functions are NNs, adept at approximating complex mappings between observations, actions, and rewards. The actor learns a policy mapping observations to actions, while the critic evaluates those actions returning values, i.e., the expected cumulative reward. PPO's algorithm utilizes a clipped surrogate objective to balance reward maximization and policy stability, ensuring efficient and stable training. In this study, the architectures of the actor and critic were tailored to the specific needs, while retaining the core algorithmic structure as outlined in Schulman et al.'s formulation (Schulman et al., 2017). Figure 4.1 shows the two networks that comprise the actor-critic agent. The actor in this study is designed as a fully connected NN comprising an input layer, hidden layers, and two output layers, as illustrated in Figure 4.3.

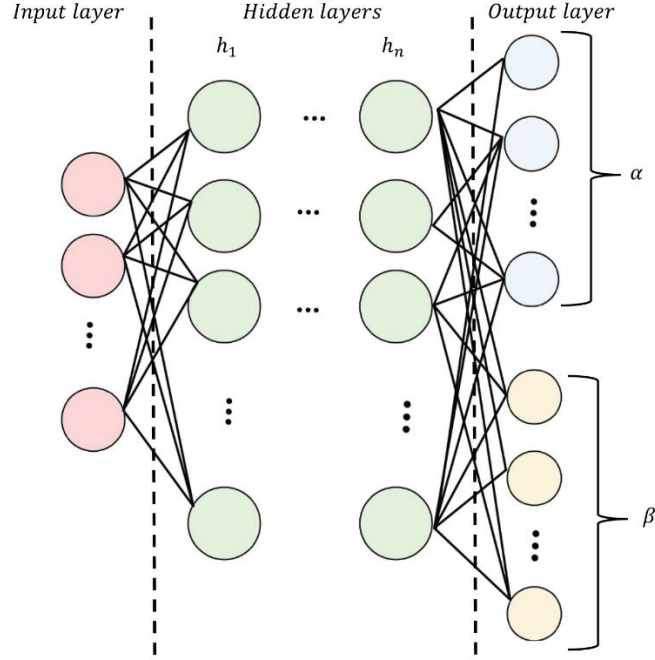


Figure 4.3: Actor network architecture

The number of neurons in the input layer matches the size of the o_i . The hidden layers possess the same number of neurons. While the two output layers, responsible for determining the parameters of a Beta distribution (α and β), each have a number of neurons equal to the total number of possible continuous actions. Note that the decision to use a Beta distribution stems from the work presented by Petrazzini and Antonelo (Petrazzini and Antonelo, 2021), which demonstrated superior performance of agents using this distribution compared to the typically used Gaussian distribution. Beta distributions are defined over a finite range of values, which makes it well-suited for real-world applications where continuous variables are bounded. Following the authors' work (Petrazzini and Antonelo, 2021), the activation functions for all layers except the output layer use a hyperbolic tangent ($\tanh$) activation function. This choice was made over the typically used Rectified Linear Unit (ReLU) activation function due to the fact that the $\tanh$ activation function bounds values in the range of -1 to 1, suiting well real-world applications where variables are bounded. Additionally, the results with these activation functions showed superior performance compared to ReLU. Following the approach used by Petrazzini and Antonelo (Petrazzini and Antonelo, 2021), a Softplus activation function, $f(x) = \log(1 + e^x)$, was employed for the output layers. To prevent multi-modal Beta distributions, a value of 1 was added to the Softplus layer, ensuring that α and β output values were greater than 1.

The critic is implemented as a fully connected NN with three types of layers: an input layer, hidden layer, and an output layer. Unlike the original implementation, the first two types of layers utilize a tanh activation function, deemed to superior results compared to ReLU activation function. While the output layer, as in the original implementation, uses a linear activation function.

The present RL framework offers advantages for CPF synthesis, design and control. It enables the construction of a CPF starting from an inlet flowrate, utilizing a pool of UOs with the aim to accomplish specific CPF goals and operational targets. The RL agent is supported in decision-making through the use of masking, which aims to introduce process knowledge into the learning process. This approach enhances the efficiency of the learning phase, allowing the agent to neglect process configurations that may not be feasible. Also, the control structure of each UO must be specified a priori thus limiting the adaptability of the resulting CPF to other control schemes. Moreover, the present RL framework does not involve multiple streams or ramifications within the CPF, preventing the consideration of disturbances that could occur within the system or at points other than the inlet. While the hybrid methodology offers significant computational advantages, it does not guarantee the optimality of the solution. Furthermore, this framework assumes that sufficient UO data can be collected to build accurate surrogate (NN) models.

4.2 Case studies

The RL framework outlined in the previous section is evaluated through four case studies adapted from the literature, each highlighting different features. The first case study serves as a proof of concept and aims to demonstrate the methodology and its effectiveness in addressing the integrated design and control problem. This methodology is compared with a model-based optimization technique to evaluate the proposed approach's efficiency. In contrast, the second case study addresses a more complex integrated problem, involving the simultaneous design and control of two UOs. In the third case study, an additional UO is incorporated into the previous formulation, granting the agent the ability to select from a broader array of UOs. The fourth case study addresses a design and control problem more closely aligned with real-world scenarios, incorporating advanced design decisions such as distillation sequences and the inclusion of a recycle. Note that for all case studies, there is no *a priori* knowledge of the arrangement or design of the UOs within the CPF. Only information about the possible UOs and their corresponding upper and lower bounds is available *a priori*. Additionally, as a starting point, the agent only has an inlet flow and a pool of UOs, which are represented in the environment as surrogate

NN models. The agent's task is to place, design, and connect these operations to develop a flowsheet that operates in closed-loop near the design target goals and outputs a cost-effective solution.

The case studies were implemented on a PC with Intel® Core™ i7-3770 CPU @ 3.40 GHz and 32 GB of RAM, using Python as the programming language. The main libraries used were torch for the development of the RL agent, scikit-learn for the training and validation of the surrogate models and gym for creating the RL environment.

4.2.1 Case study 1

The primary objective of this case study was the generation, design, and control of a CPF for a first order reaction ($A \rightarrow B$) subject to a step disturbance of -5% with respect to the nominal operation of the inlet concentration. Through this illustrative example, the aim is to outline and establish the foundation of our framework for addressing the integrated design and control problem using RL. Note that the reaction considered in this case study can be likened to the isomerization of *n*-butane to *iso*-butane; further explanation is provided in the Appendix B. The agent is tasked to establish the optimal number of CSTRs required to achieve a desired conversion, determine their sizing and control tuning parameters, effectively reject disturbances in the inlet concentration, and ensure that the reaction temperature remains below 400 K, thereby avoiding runaway reactions in any of the CSTRs in the process flowsheet, all while considering the dynamic models of the CSTRs in closed-loop operation. In this setup, the “UO pool” consists of only one type of UO – a CSTR. The maximum number of reactors the agent can use is five, which corresponds to the maximum number of steps allowed in an episode. The thermodynamic parameters and dynamic model of these reactors were adapted from the literature (Seaborg et al., 2004). A detailed formulation of the mathematical model of the CSTRs is provided in Appendix B. In this scenario (*Scenario A*) the agent was tasked to achieve a 95% conversion of the reactant A, smoothly reject disturbances in the inlet concentration, and ensure that the reaction temperature remained below 400 K to prevent a runaway reaction in any of the CSTRs included in the process flowsheet. The integrated control system in the CSTRs included a PI controller that aims to regulate the outlet concentration of reactant A by adjusting the reactor jacket's temperature. Note that the actual system is represented by the mechanistic models, meaning the surrogate models are trained using data from this representation rather than directly from the real system. However, the proposed framework is flexible to allow for training the surrogate models with data from a variety of sources such as advanced dynamic simulation tools, databases, or real-time data.

For this problem, four surrogate models were embedded within the RL environment: three regressors and one classifier. The regressors were employed to predict the endpoint of the outlet

concentration profile of the reactant A (C_A), the endpoint of the outlet temperature profile (T), and the natural logarithm of the ISE of the outlet concentration ($\ln(ISE_R)$). The classifier, on the other hand, was used to determine whether the reactor design complies with the temperature constraint (i.e., whether the temperature exceeded or not 400 K at any point during the operation) and controlled the operation around a concentration set-point specified by the agent. The dataset used to train the surrogate models was generated using LHS and comprised around 80,000 combinations of the inlet variables of the reactor's model, i.e., inlet concentration (C_{Ain}) and temperature (T_{in}), set-point concentration (C_A^{sp}), reactor's volume (V_R), and the controllers' proportional gain (K_C^R) and the time integral (τ_I^R). Note that the chosen number of combinations allows for a thorough exploration of the feasible regions around which the problem develops. More detailed information about the identification of the surrogate models, correlations between variables and the coefficient of determination of the models can be found in the Appendix B.

Among the input variables of the model, those determined by the agent and constituting the action space were: V_R , C_A^{sp} , K_C^R and τ_I^R . Referring to the structure in Figure 4.3, the total number of neurons in the output layer is 8 (2 parameters for each of the 4 actions). The observation vector, on the other hand, is composed of information about the current state of the CPF. In this case, the information provided to the agent includes the outlet concentration of reactant A and the outlet temperature of the reactor – both values calculated using the surrogate models – as well as the current step, i.e., $o_i = [C_A, T, i]$. The reward function, as mentioned in Section 4.1.2.3, comprises various sub-rewards, mainly penalties, as detailed in Table 4.1.

Table 4.1: Sub-rewards for case study 1

Sub-reward	Mathematical formulation	Awarded
UO capital cost	$cost = -\frac{V_R - V_R^{min}}{V_R^{max} - V_R^{min}}$	Every step
Dynamic cost	$\begin{cases} -1, & ISE_R \geq 1 \\ -\left(1 - \frac{\ln(ISE_R)}{\ln(ISE_R)_{min}}\right), & ISE_R < 1 \end{cases}$	Every step
Design and control constraint	$\begin{cases} 0, & \widehat{Label}_R = 1 \\ -2.0, & \widehat{Label}_R = 0 \end{cases}$	Every step
Process design requirement	$\begin{cases} 0, & x_A \geq 0.95 \\ -10 * (0.95 - x_A), & x_A < 0.95 \end{cases}$	Final step

The cost sub-reward, composed by the reactor's volume, was normalized using the maximum (V_R^{max}) and minimum (V_R^{min}) allowable volumes defined *a priori*, thereby calculating a normalized equipment cost. For the dynamic cost sub-reward, values obtained from the dynamic performance surrogate model were first converted back to their original scale by exponentiation. Subsequently, it was determined that ISE values greater than or equal to 1 received a fixed penalty of -1, while those less than the unity were re-transformed and scaled to a range between 0 and 1 by dividing it by a constant term $\ln(ISE_R)_{min}$. In this case, preliminary testing using the sampling techniques indicated that the minimum ISE achieved by the $\ln(ISE_R)$ surrogate model was e^{-11} , which corresponds to -11. Hence, the latter value was assigned to $\ln(ISE_R)_{min}$. For the design and control constraint sub-reward, a fixed penalty of -2.0 (determined from preliminary testing) was imposed based on the Boolean value ($\widehat{Label}_R$) determined by the classifier. Equation S10 in the supplementary material details how the *cons* term in Algorithm 4.1 is constructed for this case. In the event that the agent was unable to achieve the desired conversion by the end of the episode, a final sub-reward was awarded based on the deviation from the expected conversion of 95%.

The PPO agent was trained for 100,000 steps taking around 57 minutes. The actor network consisted of 2 hidden layers, each with 64 neurons, while the critic network had only 1 hidden layer with 64 neurons. The actor-critic networks were updated every 2048 steps with a discount factor of 0.99 and optimized using Adam optimizer (Kingma and Ba, 2017) with a starting learning rate of 2.5×10^{-4} , which was adjusted throughout the training.

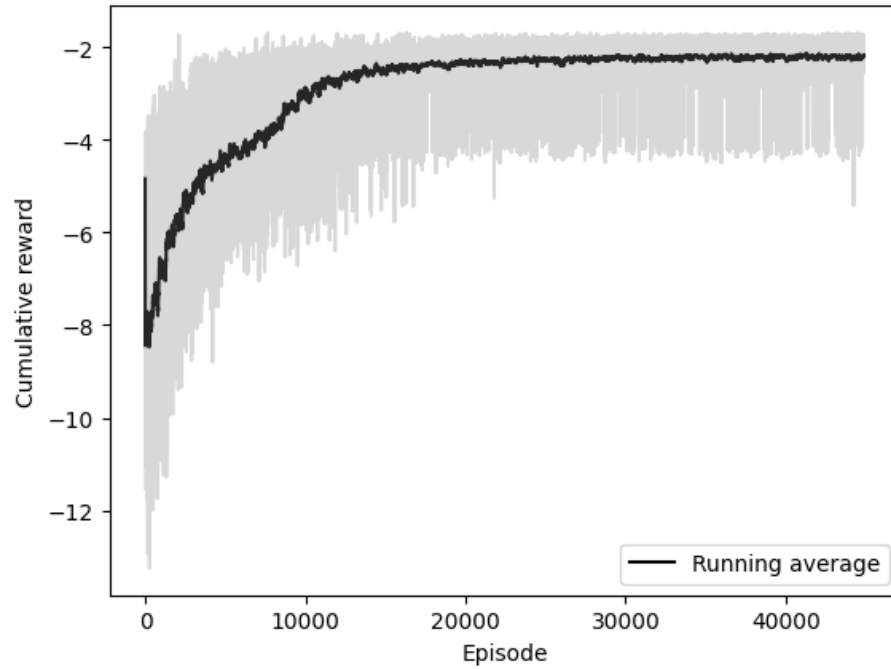


Figure 4.4: PPO agent's learning curve

Figure 4.4 shows the learning curve of the PPO agent. The solid black line represents the running average for the last 100 episodes and aims to describe the trend in the behavior of the cumulative reward throughout the agent's learning. The faded grey region represents the cumulative rewards obtained at the end of each episode. As observed in Figure 4.4 during the first 10,000 episodes, the agent's learning is highly oscillatory, averaging episodes with the maximum number of allowed steps, which in this case were five. During this exploration phase, the agent tends to design CPF systems with many CSTRs that either fail to achieve the expected conversion or violate the maximum allowed temperature constraint. As the agent's learning improves, the variability in the cumulative reward decreases, indicating refinement in the proposed process flowsheets. This includes reducing the number of CSTRs used and ensuring they operate below 400 K. This improvement is reflected in the increase and stabilization of the running mean. Note that the running mean does not reach the maximum rewards achieved by the agent, as depicted by the grey faded region. This disparity is due to the continuous nature of the action space. With a continuous action space, the search space for the agent is infinite, resulting in a persistent offset between the mean and the cumulative rewards once the running mean hits a plateau. However, the value at which the cumulative rewards plateau is very close to the maximum rewards, i.e., near the upper bound, as shown in Figure 4.4.

When evaluating the best-performing agent, the resulting CPF is capable of achieving the desired conversion, adhering to the operational constraints (temperature constraint), and producing an economically attractive solution. For this scenario, the agent required two reactors with design specifications outlined in Table 4.2 (*Scenario A*).

Table 4.2: UOs designs and specifications for case studies of scenarios A, B and C

Scenario	CSTR	C_A^{sp} [mol/L]	V_R [L]	K_C^R [$K * L/mol$]	τ_l^R [min^{-1}]
A	1	0.0999	176.3943	74.7324	0.3093
	2	0.0493	57.4667	130.2703	0.0573
B	1	0.3684	50.0026	68.2984	0.3803
	2	0.0500	132.0144	23.3307	0.0387
C	1	0.0847	175.8063	143.8290	0.3081

The cumulative reward obtained from this flowsheet design was -2.0291. Furthermore, the dynamic responses obtained when evaluating the CSTR mechanistic model with these specifications are shown in Figure 4.5.

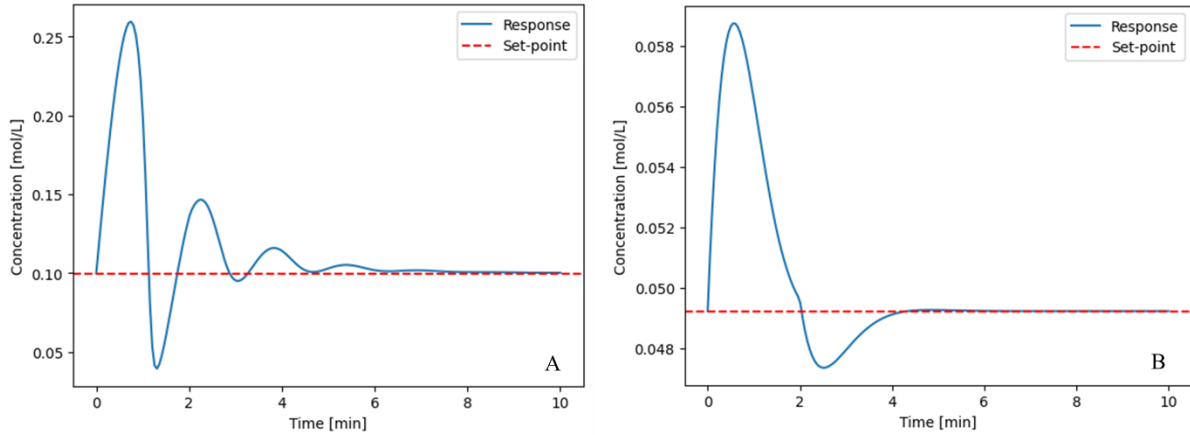


Figure 4.5: Dynamic responses Scenario A of: A) CSTR 1 B) CSTR 2

In both cases, the reactors designed by the agent successfully reject the disturbance and operate at the established set-point. As shown in Figure B.2 in Appendix B, neither of the two reactors violate the temperature constraint, consistently operating below 400 K. For the first reactor, the agent selects a large CSTR, while for the second reactor, it chooses a small-sized CSTR (bounds can be found in Table B.2 in Appendix B). The control actions taken in the first reactor are less aggressive than those in the second, resulting in a longer oscillation period. However, with a larger volume to work with, the

agent does not require aggressive control decisions to manage that CSTR unit. Conversely, the second reactor employs more aggressive control actions, allowing it to operate near their desired set-point in a shorter time.

To understand the results produced by the best-performing agent, it is important to evaluate the contribution of each of the sub-rewards to the total cumulative reward. Table 4.3 presents the values of each sub-reward in a more granular form, providing deeper insights into their individual contributions.

Table 4.3. Cumulative reward breakdown

Sub-reward	Value
UO capital cost	-0.8440
Dynamic cost	-1.1851
Design and control constraint	0.0000
Process design requirement	0.0000
Total cumulative reward	-2.0291

The results presented in Table 4.3 highlight several key observations. The total cumulative reward is mainly composed by the capital and the dynamic costs, with the latter having a larger impact. Despite both costs being normalized, the agent's selection of a small second reactor, which incurs a low normalized cost, increases the impact of the dynamic cost into the total cumulative reward. In terms of sub-reward contributions, the capital cost accounts for a 41.59% of the cumulative reward, while the dynamic cost for a 58.41%. Notably, no violations occurred in the design and control constraint or process design requirement, therefore the assigned sub-reward is 0. Experiments for case study 1 demonstrated that incorporating a mechanistic model within the RL environment yielded CPF designs comparable to those generated using surrogate models. However, the mechanistic model significantly increased the training time, requiring approximately three hours compared to just one hour with surrogate models, transferring the computational burden of designing the $\mathbf{M}_{UO}$ and $\mathbf{M}_{resp}$ matrices into the training of the RL agent.

To verify the effectiveness of the proposed methodology, the results obtained from the RL framework were compared against those from a model-based optimization approach (*Scenario B*). The MIDO problem was formulated and solved using Pyomo (Bynum et al., 2021). Initially, the MIDO problem was transformed into a MINLP through discretization methods; however, given that the maximum number of CSTRs the RL agent could select was five, enumeration was employed to avoid directly solving the MINLP. Having fixed the flowsheet, the resulting non-linear programming (NLP)

optimization problem was solved using the Interior Point Optimizer (IPOPT) solver (Wächter and Biegler, 2006) and discretized using the Lagrange-Radau method.

The optimization problem was formulated as presented in Equation 2.6, considering all the DAE, constraints and bounds. The objective function that is to be minimized was adapted from the reward function resulting in the following equation:

$$\Psi_{SB} = \sum_{j=1}^{N_{reactors}} \frac{V_{R,j} - V_R^{min}}{V_R^{max} - V_R^{min}} + \left(1 - \frac{\ln(ISE_{R,j})}{\ln(ISE_R)_{min}} \right) \quad (4.5)$$

where $N_{reactors}$ denoted the total number of reactors in the flowsheet. The full formulation of the optimization problem can be found in Equation B.11 in the Appendix B. The resulting objective function values for the five different flowsheets are presented in Table 4.4.

Table 4.4: Objective function results for Scenario B

$N_{reactors}$	Ψ_{SB}
1	Infeasible
2	1.9341
3	3.1289
4	3.4153
5	3.4420

The results of the model-based methodology indicate that the optimal configuration consists of two reactors, see Table 4.2 (*Scenario B*) for the design specifications. This is because, as the number of reactors increases, the objective function also increases. Conversely, a single reactor configuration is deemed infeasible. The reason behind this infeasibility lies in the fact that, at some point during its operation, the operating temperature exceeds the constraint of 400 K.

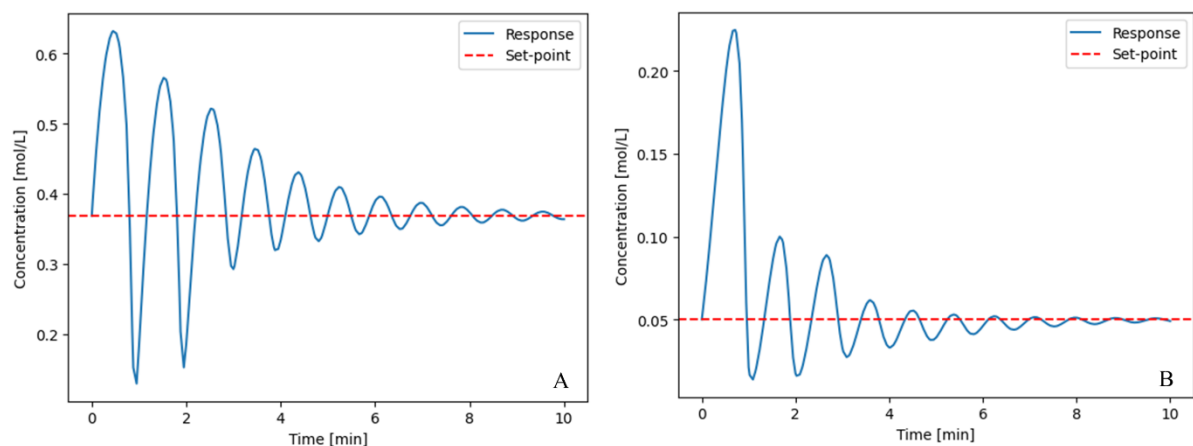


Figure 4.6: Dynamic responses in Scenario B of: A) CSTR 1 B) CSTR 2

When comparing the results of both scenarios, it is observed that the RL agent is capable of reaching a configuration with the same number of reactors. However, its solution differs somewhat from that obtained by the model-based optimization. The first major difference is the order of the reactors: the RL agent places a large reactor first in the flowsheet, followed by a smaller unit. In contrast, the model-based optimization method designs the CPF with a small reactor first, followed by a medium/large reactor. Another significant difference is the aggressiveness of the control strategy. The model-based optimization method favors less aggressive responses, which is evident from the extended periods of oscillation (Figure 4.6). Conversely, the RL agent adopts a more aggressive approach, particularly with the second reactor, where stabilization is nearly immediate.

Note that the RL agent does not achieve the same results as the model-based optimization method for two reasons. First, the agent's learning strategy plays a significant role. The PPO agent seeks to maximize its rewards at each step, so if it discovers regions that allow it to maximize rewards, the agent tends to remain in those regions. To encourage the agent to explore beyond these regions, advanced exploration techniques, such as curiosity methods, can be employed. However, these techniques are beyond the scope of this work. Another important factor is the configuration of the environment in the present framework, i.e., consideration of surrogate models instead of the explicit mechanistic process models. Although surrogate models can closely approximate mathematical models, they carry inherent errors that introduce bias to the agent. Consequently, the agent often adopts a more conservative approach, avoiding the risk of exploring flowsheets that might lead to constraint violations or fail to achieve the design objectives. Therefore, these factors contribute to the deviation between the optimal result and that obtained by the agent. Nevertheless, it is worth to emphasize that despite not achieving the optimum, the CPF proposed by the agent remains economically competitive,

adheres to all constraints, and achieves the desired conversion. This is reflected in the slight difference between the cumulative rewards obtained in both scenarios, where the difference is 0.09.

One of the advantages that the proposed RL framework aims to demonstrate is the ability to solve for different case studies using the same surrogate models without the need to generate new data. For the present scenario (*Scenario C*), the objective is to generate, design, and control a CPF capable of achieving a 90% conversion – instead of the 95% conversion of *Scenario A* –, smoothly reject the same inlet concentration disturbance and maintain the reactors' operating temperature below 400 K. Having established the new conversion goal, the PPO agent, with the same specifications, was trained for 100,000 steps (Figure B.3 in Appendix B) taking approximately 50 minutes to complete. The actor-critic networks were updated every 2048 steps with a discount factor of 0.99 and optimized using the Adam optimizer (Kingma and Ba, 2017), starting with a learning rate of 2.5×10^{-4} , which was adjusted throughout the training. The resulting CPF was composed of one CSTR with a cumulative reward of -1.6893; see Table 4.2 (*Scenario C*) for the design specifications.

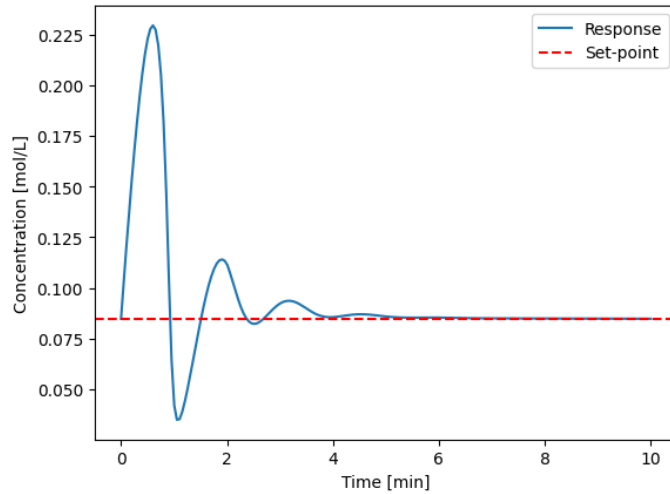


Figure 4.7: . Dynamic response of the CSTR for Scenario C

As shown in Figure 4.7, the agent effectively controls the step disturbance in the inlet concentration while maintaining the operating temperature constraint (Figure B.4 in Appendix B). In this case, the agent could have chosen a smaller reactor, which would still achieve the desired 90% conversion. However, the control actions for such a reactor might have led the agent to violate the temperature constraint, as it would require more aggressive control actions to manage the inlet disturbance. The reason the agent selects a larger reactor is due to a design-control trade-off: with a

larger volume, the control actions are less aggressive, the system experiences fewer abrupt changes, and the agent is less likely to violate the temperature constraint.

4.2.2 Case study 2

The objective of this case study is to generate, design and control a CPF that efficiently produces two high-purity products (B and C) from reactant A. The entire process is conducted isothermally and experiences a +50% step disturbance in the inlet flow rate. In this case, the RL agent has access to two UOs: a CSTR where two first-order reactions ($A \rightarrow B$ and $A \rightarrow C$) occur in parallel, and a binary distillation column (DC) that can separate products B and C. Dynamic models for the CSTR and DC were adapted from the literature (Bequette, 2002; Schweiger and Floudas, 1998), with detailed mathematical formulations provided in the Appendix C. For the first scenario (*Scenario D*), the agent was tasked with achieving two outflows, each with a high concentration of products B and C, respectively. The first outflow was required to have a molar fraction of B of 0.99 or higher, while the second outflow needed to have a molar fraction of C of no less than 0.9. To avoid unnecessary mass loss in streams containing residual reactant A, and given that the DC model is binary, the agent was expected to ensure a minimum 95% conversion of reactant A into products. Furthermore, the agent was tasked with effectively rejecting disturbances in the inlet flow rate and maintaining the reaction volume within the design limit of 8 liters in the CSTR. Each UO selected by the agent is equipped with its own integrated control system. For the reactors, a PI controller is included, which regulates the outlet concentration of reactant A by adjusting the outlet flow rate. In contrast, the DC is equipped with two controllers: one at the tops to regulate the molar fraction of product B by manipulating the reflux flow rate, and another at the bottoms to regulate the molar fraction of product B by manipulating the vapor boil-up rate. The maximum number of steps available to the agent per episode was 10. Within this limit, the agent had complete freedom to choose the number of reactors required to meet the specified conversion condition as well as the DC necessary to separate the binary mixture.

For this case study, nine surrogate models were used: four for the CSTR and five for the DC. The type and performance metric of these models is presented as follows (Table 4.5):

Table 4.5: Surrogate models used in case study 2

Unit operation	Type	Performance metric
CSTR	Regressor	Predict the endpoint of the outlet concentration of reactant A (C_A)
	Regressor	Predict the endpoint of the liquid reaction volume profile within the reactor (V^L)

DC	Regressor	Regress the natural logarithm of the ISE which measures the difference between C_A and the specified set-point (ISE_R)
	Classifier	Asses whether the operation was conducted under control around the specified set-point and determine if the reactor design violated the design constraint, i.e., whether the liquid reaction volume exceeded the design volume at any point during operation
	Regressor	Predict the endpoint of the liquid molar fraction of product B in the distillate (x_D)
	Regressor	Predict the endpoint of the liquid molar fraction of product B in the bottoms (x_B)
	Regressor	Regress the natural logarithm of the ISE at the distillate (ISE_D)
	Regressor	Regress the natural logarithm of the ISE at the bottoms (ISE_B)
	Classifier	Determine whether the chosen design violates any inherent constraints of the DC (e.g., process simulation failure due to DAEs non-compliance) and assess whether the system remains under control around the established set-points.

Appendix C provides more information on the input matrices to the mechanistic models, the correlations between variables, the architecture of the surrogate models, and the training results of these models.

The action space for this problem was composed of all the design decisions related to the reactor and the column (Equation 4.6). For the reactor: set-point of the concentration of A (C_A^{sp}), initial liquid reaction volume (V_{in}^L), and controller parameters (K_C^R, τ_I^R). For the DC: the number of stages (N), column diameter (D_C), feed stage ($feed_N$), reflux stage ($reflux_N$), and controller parameters at the distillate (K_C^D, τ_I^D) and bottoms (K_C^B, τ_I^B).

$$\mathbf{A} = [C_A^{sp}, V_{in}^L, K_C^R, \tau_I^R, N, D_C, feed_N, reflux_N, K_C^D, \tau_I^D, K_C^B, \tau_I^B] \quad (4.6)$$

Note that in each step, a_i is composed of all these design variables; however, not all variables are used in every step. To simplify the analysis, the RL environment is designed such that the DC remains inactive until the agent has successfully converted 95% of reactant A into the product. Once this conversion is achieved, the reactor is deactivated, and the DC is activated. Note that removing the masking from the framework will interfere with the agent's learning process, potentially extending the time required to converge to a solution or making the learning noisier. One possible outcome is that the agent may fail to fully convert reactant A into products, leading to erroneous flowsheets that incorrectly utilize a binary DC to separate a ternary mixture. During the steps when the DC is inactive, although

the agent samples actions from the probability distributions of the column's design variables, the sampled values are not utilized, occurring the same with the reactor when this UO is inactive. The observation vector for this case study contains all pertinent information from the environment so that it can be used for both UOs. The information stored in the observation vector includes the outlet concentration of reactant A, the outlet concentration of product B, the outlet concentration of product C, the molar fraction of product B, and the current step (Equation 4.7). Note that the outlet concentrations become the inlet concentrations for the subsequent UO.

$$o_i = [C_A, C_B, C_C, z_B, i] \quad (4.7)$$

For both UOs, the reward function varies; however, in general terms, it contains the same elements: a capital cost, a dynamic cost, and a design and control constraint cost. If the agent is unable to meet the process design requirements—namely, achieving the desired molar fractions of products B and C after converting reactant A to a certain extent within the permitted number of steps—a further penalty is imposed on the agent. The different sub-rewards for this case study are detailed in Table 4.6.

Table 4.6: Sub-rewards for case study 2

Sub-reward	UO	Mathematical formulation	Awarded
UO capital cost	CSTR	$cost_R = -\frac{V_{in}^L - V_{in_{min}}^L}{V_{in_{max}}^L - V_{in_{min}}^L}$	Every step
	DC	$cost_{DC} = -\frac{V_{DC} - V_{DC}^{min}}{V_{DC}^{max} - V_{DC}^{min}}$	
Dynamic cost	CSTR	$\begin{cases} -1, & ISE_R \geq 1 \\ -\left(1 - \frac{\ln(ISE_R)}{\ln(ISE_R)_{min}}\right), & ISE_R < 1 \end{cases}$	Every step
	DC	$\begin{cases} -1, & ISE_B \geq 1 \\ -\left(1 - \frac{\ln(ISE_B)}{\ln(ISE_B)_{min}}\right), & ISE_B < 1 \end{cases}$	
	DC	$\begin{cases} -1, & ISE_D \geq 1 \\ -\left(1 - \frac{\ln(ISE_D)}{\ln(ISE_D)_{min}}\right), & ISE_D < 1 \end{cases}$	
Design and control constraint	CSTR	$\begin{cases} 0, & \widehat{Label}_R = 1 \\ -1.0, & \widehat{Label}_R = 0 \end{cases}$	Every step
	DC	$\begin{cases} 0, & \widehat{Label}_{DC} = 1 \\ -5.0, & \widehat{Label}_{DC} = 0 \end{cases}$	

Process design requirement	Both	$\begin{cases} 0, & i < I \\ -5.0, & i \geq I \end{cases}$	Final step
----------------------------	------	--	------------

The capital cost sub-reward, similar to the previous case study, is composed of a volume normalized by the maximum and minimum possible values. In the case of the CSTR, the considered volume is the initial liquid reaction volume chosen by the agent. For the DC, the volume is composed as follows:

$$V_{DC} = \left(\pi * \frac{D_C}{2} \right)^2 * (1.2 * 0.61 * (N - 2)) \quad (4.8)$$

The dynamic sub-reward follows the same structure and logic as that used in the previous case study. For this case, the value of $\ln(ISE_R)_{min}$ was -8, while for both $\ln(ISE_D)_{min}$ and $\ln(ISE_B)_{min}$ a value of -17 was used. Note that these values are not chosen arbitrarily but are based on an analysis of the minimum values at which the surrogate models can perform regression. For the design and control constraint sub-reward, the agent received two fixed penalties depending on the UO. For the CSTR, a design or control violation resulted in a penalty of -1. This penalty aims to indicate that the proposed design was incorrect without significantly hindering the agent's exploration capabilities. In contrast, a more severe penalty was associated with the DC to indicate that the proposed design was incorrect. The difference in penalties arises from the fact that violations of constraints by the DC are more undesirable than those by the reactor, indicating that the proposed design leads to a failure to satisfy the DAEs. Additionally, a larger penalty value must be assigned to this UO because its reward function comprises more terms, and if the penalty value is too small, its impact may be diminished or overlooked. In the case where the agent is unable to convert all the reactant into the product and distill the product within the maximum number of permitted steps in an episode (I), a final penalty is imposed on the agent. This penalty indicates that the proposed CPF is incapable of achieving the intended design objectives within the allowed number of steps.

The training of the PPO agent took 59.17 minutes to complete 100,000 steps. The actor network consisted of 2 hidden layers, each with 64 neurons, while the critic network had only 1 hidden layer with 64 neurons. The actor-critic networks were updated every 2048 steps with a discount factor of 0.99 and optimized using Adam (Kingma and Ba, 2017) optimizer with a starting learning rate of $2.5e-4$, which was adjusted throughout the training.

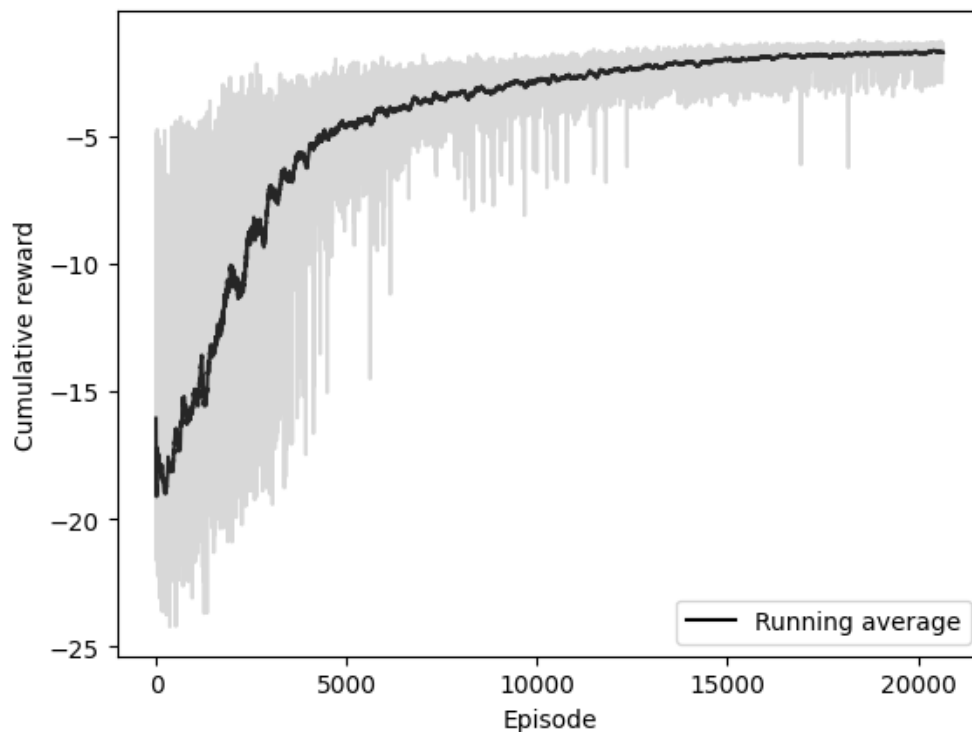


Figure 4.8: PPO agent's learning curve

The learning curve of the PPO agent, depicted in Figure 4.8, shows the cumulative reward versus the number of episodes, highlighting the agent's performance over time. The agent's exploration phase, characterized by the wide spread in the faded grey region, extends over the first 5,000 episodes. Throughout this phase, the agent learns to design CPFs that do not violate the design constraints and that correctly control the system. This learning process is reflected in the steady increase of the running mean. After exploring various CPFs, the next phase of the agent's learning corresponds to the refinement of its designs (exploitation). In this phase, the rate of learning growth is not as immediate as in the exploration phase, but the variability in the designed flowsheets is much lower, as clearly reflected in the shrinkage of the area of the grey faded region – cumulative rewards are more similar.

When evaluating the best-performing agents, the resulting CPF is capable of distilling products B and C to molar fractions of 0.99 and 0.9, respectively, having previously converted reactant A to 95%; adhere to the design constraints (reaction volume smaller than 8 liters); and produce an economically attractive solution. For this case, the agent required three CSTRs and a DC with the design specifications shown in Table 4.7 (*Scenario D*) and Figure 4.9.

Table 4.7: UOs designs and specifications for case study 2 scenarios D and E

Scenario	Variable	CSTR 1	CSTR 2	CSTR 3	DC
D	V_{in}^L [L]	3.4175	3.3842	3.4020	-
	C_A^{sp} [mol/L]	0.3581	0.1247	0.0482	-
	K_C^R [$L^2/mol \min^{-1}$]	-157.9105	-174.1107	-180.4943	-
	τ_I^R [$\min^{-1}$]	5.1789	5.8452	6.2748	-
	N	-	-	-	13
	D_C [m]	-	-	-	0.3336
	$feed_N$	-	-	-	4
	$reflux_N$	-	-	-	12
	K_C^D [mol/min]	-	-	-	-15.7539
	τ_I^D [$\min^{-1}$]	-	-	-	0.0493
	K_C^B [mol/min]	-	-	-	16.2139
	τ_I^B [$\min^{-1}$]	-	-	-	0.0477
E	V_{in}^L [L]	3.1359	3.2970	3.6038	-
	C_A^{sp} [mol/L]	0.3015	0.1146	0.04977	-
	K_C^R [$L^2/mol \min^{-1}$]	-156.5406	-179.6652	-180.6296	-
	τ_I^R [$\min^{-1}$]	5.4388	5.7206	5.0507	-
	N	-	-	-	14
	D_C [m]	-	-	-	0.3158
	$feed_N$	-	-	-	7
	$reflux_N$	-	-	-	13
	K_C^D [mol/min]	-	-	-	-16.6067
	τ_I^D [$\min^{-1}$]	-	-	-	0.0162
	K_C^B [mol/min]	-	-	-	13.3983
	τ_I^B [$\min^{-1}$]	-	-	-	0.0166

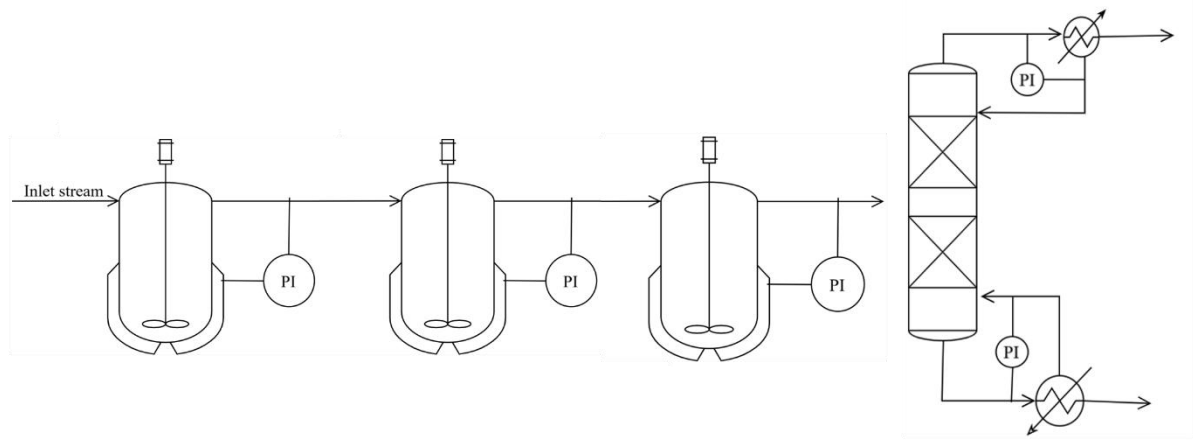


Figure 4.9: Final flowsheet for both scenarios

Furthermore, the dynamic responses obtained when evaluating the mechanist models of the CSTR and DC with these specifications are shown in Figure 4.10 and 4.11, respectively.

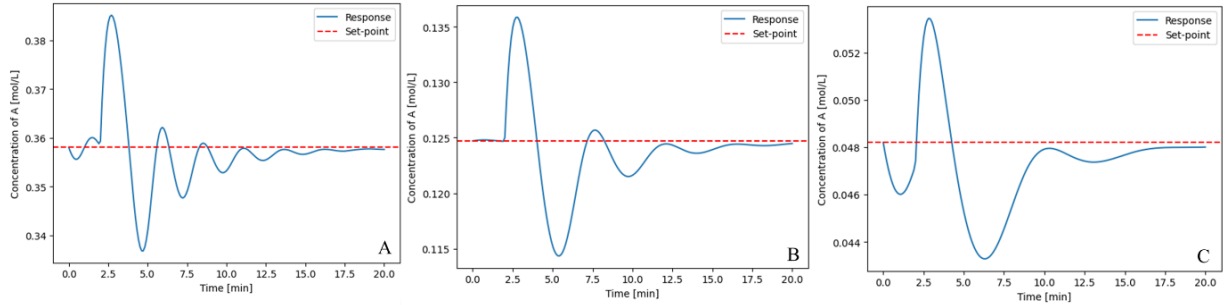


Figure 4.10: Dynamic responses Scenario D of: A) CSTR 1 B) CSTR 2 C) CSTR 3

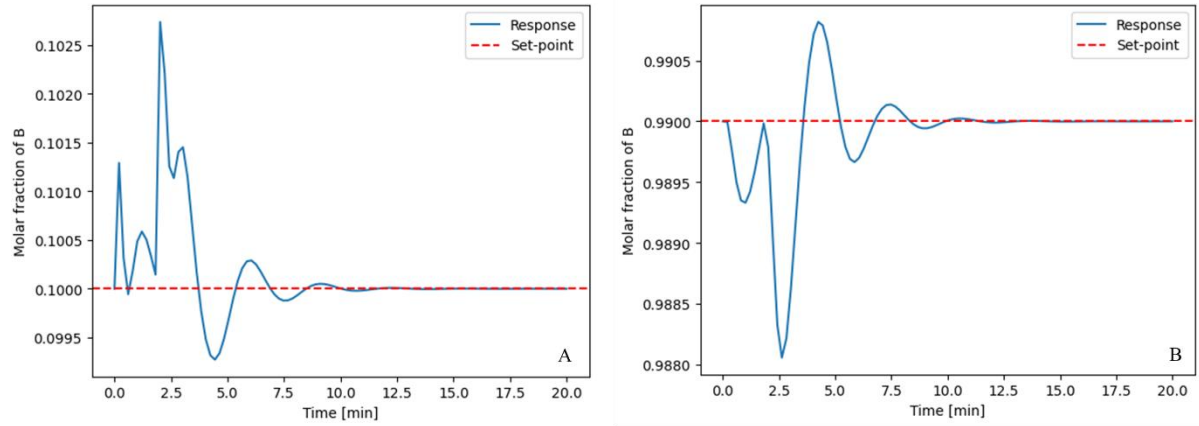


Figure 4.11: Dynamic responses Scenario D of DC: A) Bottoms B) Overheads

For the three CSTRs designed by the agent, the disturbance is successfully rejected while operating at the established set-point (Figure 4.10). Notably, the agent could have chosen two reactors to achieve the desired conversion of A with fewer units. However, due to the volume constraint, the agent opted to include three reactors with initially small liquid reaction volumes. The first two reactors experience significant volume swings, reaching up to 6.5 liters before stabilizing at a lower volume. Nevertheless, none of the reactors violate the design constraints (see Figure C.2 in Appendix C). Regarding process control, the three reactors employ large K_C^R and τ_I^R values (see Table C.2 in Appendix C for variable bounds). This control scheme induces oscillatory behavior in the response variable, specially for the first reactor, before stabilizing at the set-point. For the DC, as with the reactors, the disturbance is successfully rejected while operating at the established set-points (Figure 4.11). Regarding process design, the agent selects a column with a small diameter and an intermediate

number of stages (see Table C.5 in Appendix C for variable bounds). Since the molar fraction required for product B is higher than for product C, the agent feeds the column in the lower middle section to allow for more enrichment stages. The chosen control strategy is aggressive, where the response variable is quickly controlled with minimal oscillations, characterized by low τ_I values.

As mentioned in case study 1 *Scenario C*, one of the main advantages of the present RL framework is the ability to employ surrogate models across different problems, provided that the prediction ranges of these models align with the modifications required by the new problem. To further illustrate this feature, the case study presented in this scenario (*Scenario D*) was modified by tasking the agent to achieve 0.99 molar fractions of B and C in both of the CPF's outflows, while maintaining the 95% conversion of reactant A. The design constraint of 8 liters remained unchanged, and the agent was allowed to use a maximum of 10 steps to achieve the corresponding design specifications. Having established the new molar fraction goal, the PPO agent, with the same specifications, was trained for 100,000 steps for the course of 58.34 minutes. The actor-critic networks were updated every 2048 steps with a discount factor of 0.99 and optimized using the Adam optimizer (Kingma and Ba, 2017), starting with a learning rate of 2.5×10^{-4} , which was adjusted throughout the training.

The overall structure of the resulting CPF closely mirrored that of *Scenario D*, with some variations in the design specifications of the UOs, detailed in Table 4.7 (*Scenario E*).

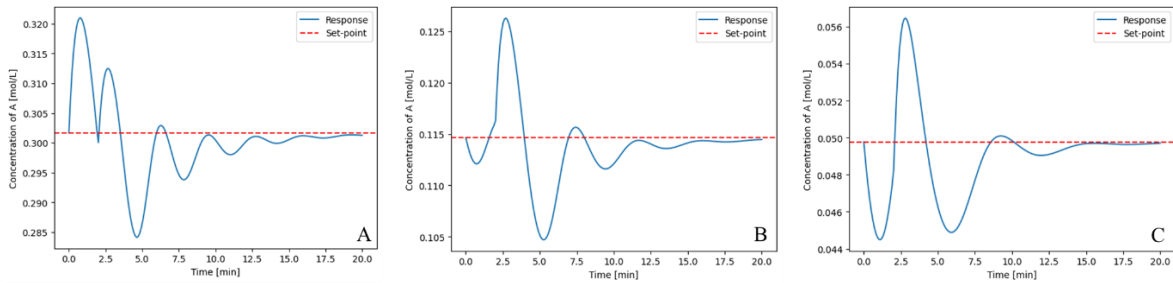


Figure 4.12: Dynamic responses Scenario E of: A) CSTR 1 B) CSTR 2 C) CSTR 3

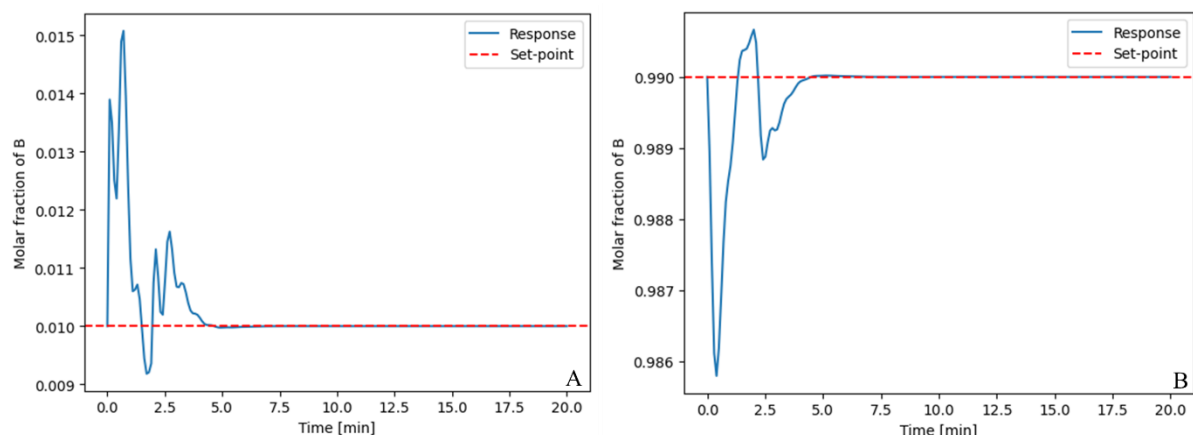


Figure 4.13: Dynamic responses Scenario E of DC: A) Bottoms B) Overheads

Similar to *Scenario D*, all UOs successfully reject disturbances in the inlet flow, maintaining control around the specified set point, as presented in Figures 4.12 and 4.13. As shown in Table 4.7 (*Scenario E*), the reactor designs for both case studies exhibit a very similar configuration, reflected in the dynamic response where the dynamic responses for the concentration of A is similar between both cases. Similarly, all reactors adhered to the volume design constraint, maintaining a reaction volume below 8 liters. Note that in this scenario, the first reactor exhibits the largest fluctuations in volume, reaching values above 7 liters, which are higher than those observed in the reactors of the previous scenario (see Figure C.3 in the Appendix C). On the other hand, the DC shows variations between the two case studies. The primary difference lies in the feed stage, where in *Scenario E*, the agent decides to feed the column at the midpoint, resulting in an equal number of enriching and stripping stages. Additionally, there is a change in the total number of stages, with the DC designed in *Scenario E* having more stages. This behavior is logical, as the agent recognizes the need to add more stages to achieve higher purity at both ends of the column. Regarding control, the approach adopted in this case study is more aggressive compared to the previous one, evidenced by minimal to no oscillations and rapid system stabilization in both the overheads and bottoms. This aggressiveness in the dynamic response is expected given that the time constant (τ_I) for both PI controllers in *Scenario E* are four times smaller than in *Scenario D*.

4.2.3 Case study 3

This case study builds upon the problem outlined in the previous section, with the key difference being the introduction of a third UO, a flash tank (FT). The inclusion of this unit enhances the flexibility of the process, enabling exploration of alternative separation strategies for the products. As in the previous

case study, the entire process is conducted isothermally and subjected to a +50% step disturbance in the inlet flow rate. The dynamic model for the FT was adapted from the literature (Schweiger and Floudas, 1998), with detailed mathematical formulations provided in Appendix D. Recognizing the limitations of FT in the separation process compared to DC, the focus of this case study was narrowed to a single outlet stream. Specifically, attention was placed on the molar fraction of product B in the bottoms' outlet stream for the DC and the liquid outlet stream for the FT.

In the first scenario (*Scenario F*), the target molar fraction for product B was set at 0.1. Similar to the previous case study, to avoid unnecessary mass loss in streams containing residual reactant A and given that the separation process in both units is binary, the agent was expected to ensure a minimum 95% conversion of reactant A into products. Furthermore, the agent was tasked with effectively rejecting disturbances in the inlet flow rate and maintaining the reaction volume within the design limit of 8 liters in the CSTR. Each UO selected by the agent is equipped with its own integrated control system. The control structure of the reactor and the DC is described in section 4.2.2. Conversely, the FT is equipped with a PI controller that regulates the liquid molar fraction of product B by adjusting the vapor outlet flow rate. The maximum number of steps available to the agent per episode was set at 10, allowing complete freedom to choose the necessary reactors to meet the specified conversion condition, as well as the DC or FTs needed to separate the binary mixture.

For this case study 13 surrogate models were employed, including the nine previously introduced in Table 4.4, along with four additional models specifically tailored for the FT. Note that the surrogate models for the DC were retrained, extending the range of x_B^{sp} by adjusting the upper bound from 0.1 to 0.4. The type and performance metrics for the FT surrogate models are presented in Table 4.8:

Table 4.8: FT's surrogate models

Unit operation	Type	Performance metric
FT	Regressor	Predict the endpoint of the liquid molar fraction of product B in the liquid outlet flow rate (x_B)
	Regressor	Predict the endpoint of the liquid outlet flowrate (L_{FT})
	Regressor	Regress the natural logarithm of the ISE ($\ln(ISE_{FT})$) which measures the difference between x_B and the specified set-point
	Classifier	Determine whether the chosen design violates any inherent constraints of the FT (e.g., process simulation failure due to DAEs non-compliance) and assess whether the system remains under control around the established set-points.

More detailed information about the identification of the surrogate models, correlations between variables and the coefficient of determination of the models can be found in Appendix D.

The action space for this problem builds upon the action space outlined in the previous section (Equation 4.6), incorporating the design decisions specific to the FT, which are as follows: molar fraction set-point of B at the liquid outlet (x_L^{sp}) and controller parameters at the vapor outlet (K_C^{FT}, τ_I^{FT}). In addition to the design variables of the three UOs, $\mathbf{A}$ also includes a binary variable that indicates the choice between DC and FT, denoted as $\tilde{\mu}$.

$$\mathbf{A} = [C_A^{sp}, V_{in}^L, K_C^R, \tau_I^R, N, D_C, feed_N, reflux_N, K_C^D, \tau_I^D, K_C^B, \tau_I^B, x_L^{sp}, K_C^{FT}, \tau_I^{FT}, \tilde{\mu}] \quad (4.9)$$

As in the previous case study, the binary and discrete variables in this case were approximated using continuous distributions to enable the application of the same framework discussed in earlier sections. To simplify the analysis, the RL environment is designed such that the DC and FT remain inactive until the agent has successfully converted 95% of reactant A into the product. Once this conversion is achieved, the reactor is deactivated, and the previously inactive UOs are activated. The observation vector for this case study is the same as the one presented in Equation 4.7.

The reward functions for the reactor and DC from case study 2 were retained for the current analysis. In line with this approach, the reward function for the FT was similarly constructed, encompassing sub-rewards for capital cost, dynamic operating cost, and design and control constraint. If the agent was unable to meet the process design requirements—namely, achieving the desired molar fractions of products B after converting reactant A to a certain extent within the permitted number of steps—a further penalty was imposed on the agent. The different sub-rewards for the FT are detailed in Table 4.9.

Table 4.9: Sub-rewards for FT

Sub-reward	Mathematical formulation	Awarded
UO capital cost	$cost_{FT} = -0.1$	Every step
Dynamic cost	$\begin{cases} -1, & ISE_{FT} \geq 1 \\ -\left(1 - \frac{\ln(ISE_{FT})}{\ln(ISE_{FT})_{min}}\right), & ISE_{FT} < 1 \end{cases}$	Every step
Design and control constraint	$\begin{cases} 0, & \widehat{Label}_{FT} = 1 \\ -5.0, & \widehat{Label}_{FT} = 0 \end{cases}$	Every step

In this case, since there are no sizing variables for the FT, a constant capital cost of -0.1 was assigned. This value was chosen to symbolically represent the cost of adding a FT to the flowsheet, indicating that it incurs an expense but is considerably smaller compared to the cost of a DC. The dynamic sub-reward follows the same structure and logic as that used in the previous case study. For this case, the value of $\ln(ISE_{FT})_{min}$ was -8, value based on an analysis of the minimum values at which the surrogate model can perform regression. For the design and control constraint sub-reward, a similar approach to that of the DC was employed. In this case, the objective was to ensure that the agent violated the constraints of the FT, as such violations would result in failure to satisfy the DAEs. The penalty assigned was sufficiently large to significantly impact the agent's reward, discouraging constraint violations and preventing the agent from converging on suboptimal designs that yield comparable rewards without violating the design and control constraints.

The PPO agent was trained for 100,000 steps taking around 52.27 minutes. The actor network consisted of 2 hidden layers, each with 64 neurons, while the critic network had only 1 hidden layer with 64 neurons. The actor-critic networks were updated every 2048 steps with a discount factor of 0.99 and optimized using Adam optimizer (Kingma and Ba, 2017) with a starting learning rate of 2.5×10^{-4} , which was adjusted throughout the training.

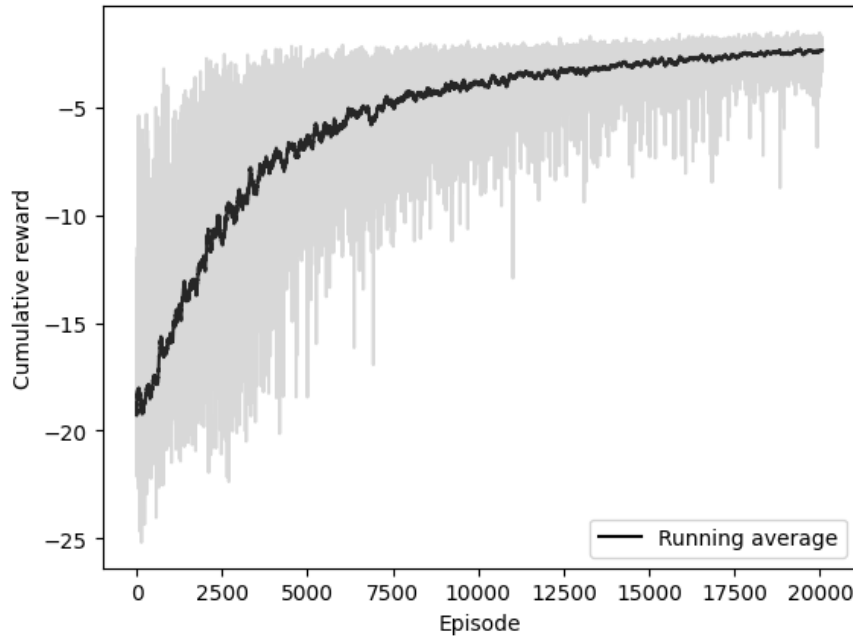


Figure 4.14: PPO agent's learning curve

A comparison of the computational times required for training agents in both *Scenario D* and *Scenario E* revealed no substantial differences, suggesting that the inclusion of an additional UO in the problem formulation did not significantly impact computational time. In contrast, the learning process was significantly influenced by the presence of the additional UO. This effect was clearly illustrated in Figure 4.14, which presented the learning curve for the PPO agent in *Scenario F*. A comparison of Figures 4.8 and 4.14 yielded the following observations. First, *Scenario F* exhibited a noisier learning process than *Scenario D*, with a prolonged exploration phase; specifically, the exploration phase in *Scenario F* continued until approximately 6,000 episodes, whereas in *Scenario D*, it concluded around 4,000 episodes. Furthermore, the increased noise was not limited to the exploration phase, as the exploitation phase was also markedly noisier in *Scenario F* compared to *Scenario D*. The increase in variability of cumulative rewards did not imply that the agent failed to converge to a solution; rather, it indicated the expanded search space available to the agent, represented by a larger action space. This broader action space required the agent to explore a wider array of alternatives, resulting in more pronounced fluctuations in cumulative rewards. Despite these differences, the cumulative reward trends remained similar, with exponential growth during the initial episodes followed by a plateau at a reward level of approximately -2.5.

When evaluating the best-performing agents, the resulting CPF is capable of distilling product B to a molar fraction of 0.1, having previously converted reactant A to 95%; adhere to the design constraints (reaction volume smaller than 8 liters); and produce an economically attractive solution. For this case, the agent required three CSTRs and a DC with the design specifications shown in Table 4.10 (*Scenario F*) and Figure 4.9.

Table 4.10: UOs designs and specifications for case study 3 scenarios F and G

Scenario	Variable	CSTR 1	CSTR 2	CSTR 3	DC/FT
F	V_{in}^L [L]	3.8988	3.7787	3.8763	-
	C_A^{sp} [mol/L]	0.3352	0.1032	0.0480	-
	K_C^R [$L^2/mol \min^{-1}$]	-164.6513	-172.4885	-173.0276	-
	τ_I^R [$\min^{-1}$]	5.2697	5.9563	6.3171	-
	N	-	-	-	13
	D_C [m]	-	-	-	0.3454
	$feed_N$	-	-	-	5
	$reflux_N$	-	-	-	12
	K_C^D [mol/min]	-	-	-	-13.7983
	τ_I^D [$\min^{-1}$]	-	-	-	0.1381
	K_C^B [mol/min]	-	-	-	14.6740

G	$\tau_I^B [\text{min}^{-1}]$	-	-	-	0.0785
	$V_{in}^L [L]$	4.0549	4.0239	4.1461	-
	$C_A^{sp} [\text{mol}/L]$	0.3482	0.1044	0.0498	-
	$K_C^R [L^2/\text{mol min}^{-1}]$	-148.5977	-162.3069	-171.0000	-
	$\tau_I^R [\text{min}^{-1}]$	4.9108	5.7208	6.1898	-
	x_L^{sp}	-	-	-	0.3635
	$K_C^{FT} [\text{mol}/\text{min}]$	-	-	-	7.7068
	$\tau_I^{FT} [\text{min}^{-1}]$	-	-	-	0.4628

The dynamic responses obtained when evaluating the mechanist models of the CSTR and DC with these specifications are shown in Figure 4.15 and 4.16, respectively.

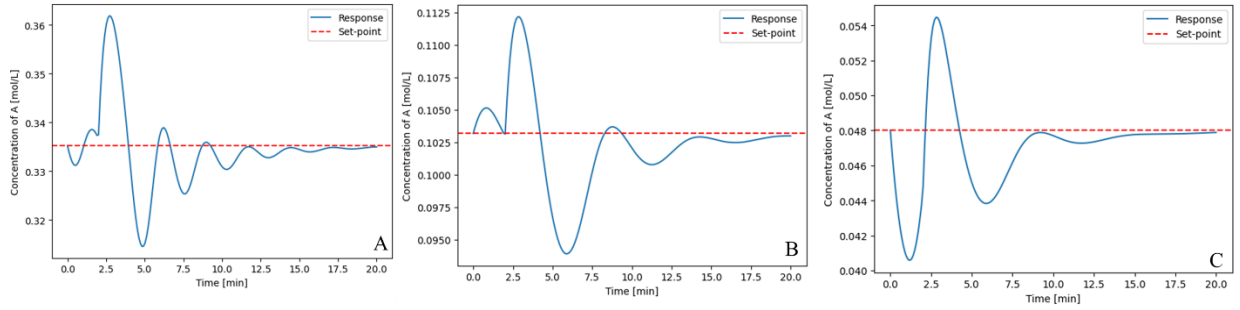


Figure 4.15: Dynamic responses Scenario F of: A) CSTR 1 B) CSTR 2 C) CSTR 3

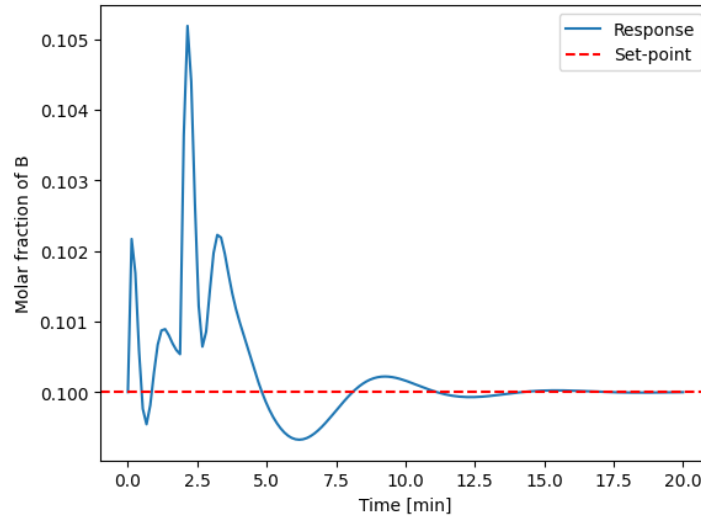


Figure 4.16: Dynamic response of DC Scenario F

The results for this scenario indicate that the designed CPF requires a DC instead of a flash separator. Although the capital cost of a FT is lower, the agent learned during training that using multiple flash units in sequence failed to achieve the desired product purity. To experimentally validate

this finding, attempts were made to reach the required purity using only FTs. It was observed that, to reach such high purities, the product-rich stream (containing product C) had very low flow rates, while the opposing stream (liquid or vapor) contained most of the mass flow with practically the same composition as the inlet flow. This made the separation process impractical for achieving the target purity.

Similar to the previous case study, the designed UOs were capable of rejecting the step disturbance in the inlet flow, maintaining control around the specified set point, as shown in Figures 4.15 and 4.16. In this scenario, the agent selected reactors with slightly larger initial reaction volumes compared to *Scenario D*. However, this increase did not violate the volume constraint (see Figure D.2 in Appendix D), as the total reaction volume of each reactor remained below the 8-liter limit. Notably, for the second reactor, the increased volume brought it closer to the permitted maximum, with values approaching 7.7 liters. Regarding the DC, the analysis paralleled that of the reactors. The overall design was similar, with the only significant changes being a feed stage positioned one tray higher than in *Scenario D* and a slightly larger diameter. Additionally, a minor adjustment was observed in the controller design: a less aggressive PI controller was used, leading to prolonged oscillations that settled after 15 minutes.

To further enhance the work, and considering the significant advantage offered by the proposed framework – which allows the use of surrogate models across various problems – a new scenario was addressed. In *Scenario G*, the agent was tasked with achieving a target molar fraction of 0.4 for product B, while maintaining the 95% conversion of reactant A. The design constraint of 8 liters remained unchanged, and the agent was allowed to use a maximum of 10 steps to achieve the corresponding design specifications. Having established the new molar fraction goal, the PPO agent, with the same specifications from *Scenario F*, was trained for 100,000 steps for the course of 50.35 minutes. The actor-critic networks were updated every 2048 steps with a discount factor of 0.99 and optimized using the Adam optimizer (Kingma and Ba, 2017), starting with a learning rate of $2.5\text{e-}4$, which was adjusted throughout the training.

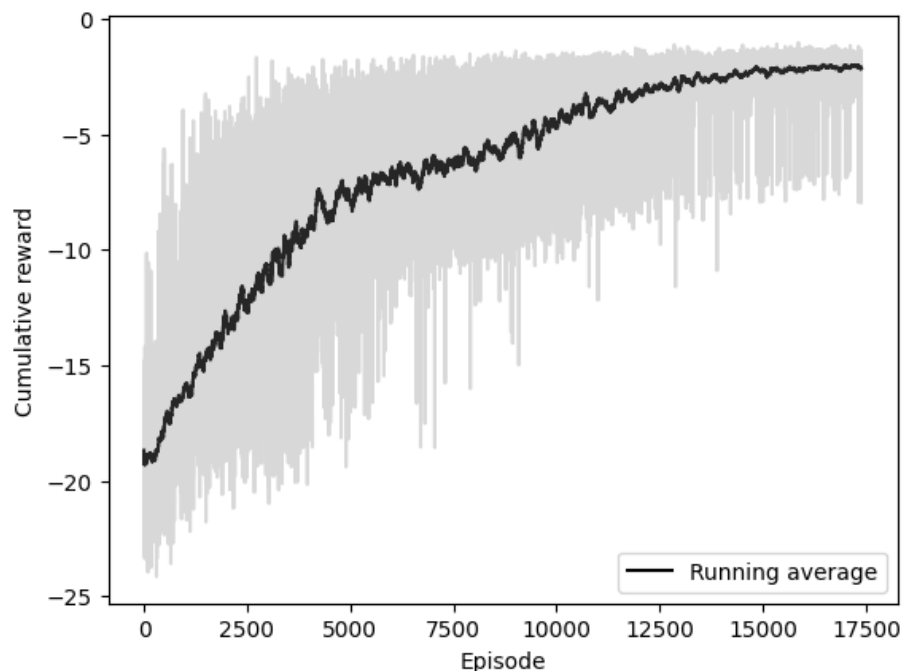


Figure 4.17: PPO agent's learning curve

Figure 4.17 shows the learning curve for the PPO agent trained in *Scenario G*, with the running average showcasing the trend of the agent's learning progress. The learning pattern in this case is distinct, as it does not follow the typical trend of transitioning from exponential growth to a plateau. Instead, after an initial period of exponential growth during the first 5,000 episodes, the agent experiences an intermediate phase of linear growth. This allows the agent to plateau around episode 12,500, reaching a cumulative reward of approximately -3.0. This difference in learning is also reflected in the variability of cumulative rewards. Unlike previous scenarios (Figures 4.8 and 4.14), where variability significantly decreases after 5,000 episodes, this reduction does not occur until around episode 12,500. While there is a slight decrease in variability between episodes 5,000 and 12,500, it is less evident than in the other cases. Several factors contribute to this variability. First, the DC is more prone to simulation errors when trying to distill component B at a lower purity. This was confirmed through post-training tests. Second, the agent struggles to find the optimal flowsheet and often selects multiple FTs in series, as observed during its learning process. This approach typically results in errors, as the FT has a very narrow operational range, and even minor deviations in conditions lead to simulation errors.

When evaluating the best-performing agents, the resulting CPF is capable of distilling product B to a molar fraction of 0.4, having previously converted reactant A to 95%; adhere to the design

constraints (reaction volume smaller than 8 liters); and produce an economically attractive solution. For this case, the agent required three CSTRs and a FT with the design specifications shown in Table 4.10 (*Scenario G*) and the obtained CPF shown in Figure 4.18.

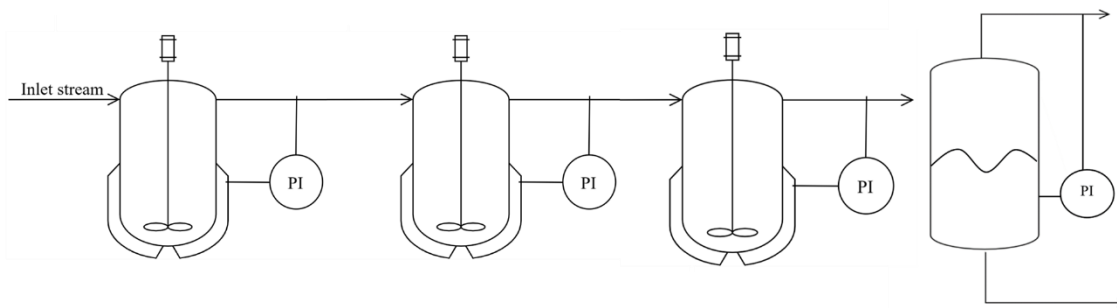


Figure 4.18: Final flowsheet Scenario G

The dynamic responses obtained when evaluating the mechanist models of the CSTR and FT with these specifications are shown in Figure 4.19 and 4.20, respectively.

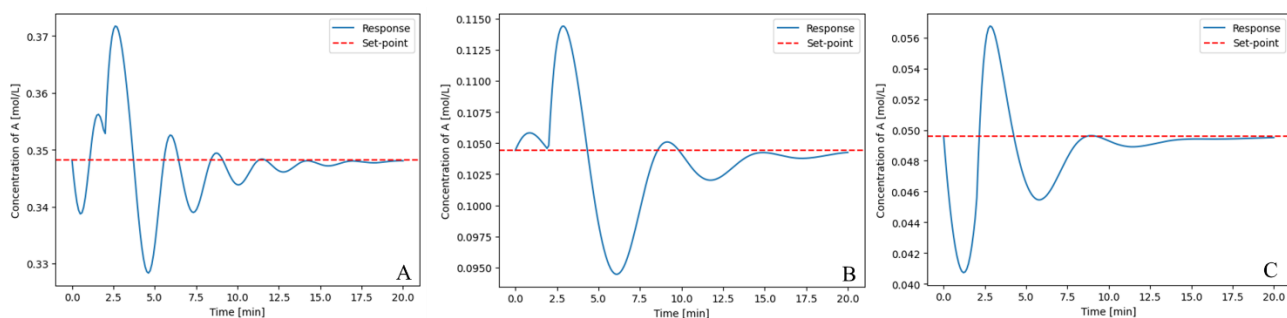


Figure 4.19: Dynamic responses Scenario G of: A) CSTR 1 B) CSTR 2 C) CSTR 3

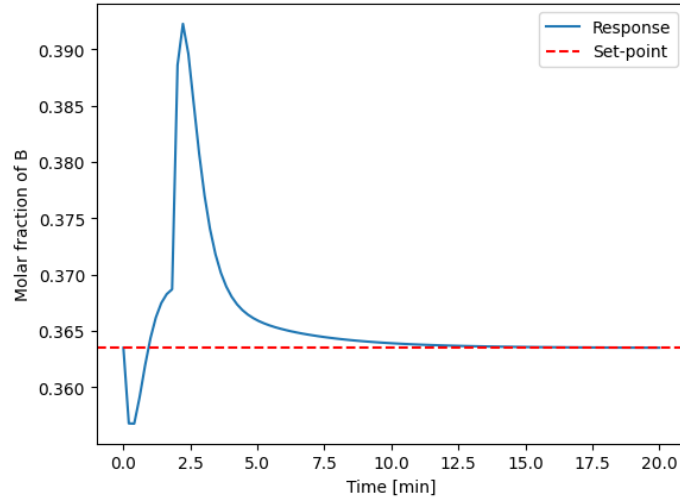


Figure 4.20: Dynamic response of FT Scenario G

Analyzing the CPF obtained from *Scenario G*, several key observations emerge. The three reactors designed by the agent successfully rejected the disturbance in the inlet flow rate while maintaining operation at the established set point, as shown in Figure 4.19. Compared to previous scenarios, the selected V_{in}^L are larger, with initial values around 4 liters. Despite the increased volumes, none of the reactors violate the 8-liter design constraint (see Figure D.3 in Appendix D). The second reactor comes closest to the limit, exhibiting oscillations that peak at approximately 7.9 liters. In terms of process control, same as the other scenarios, the three reactors utilize relatively large values for both K_C^R and τ_I^R (refer to Table C.2 in Appendix C for variable bounds). For the FT, similar to the reactors, the disturbance is successfully rejected while maintaining operation at the established set points, as illustrated in Figure 4.20. In terms of process control, the agent adopts a relatively aggressive strategy, allowing for a swift response of the controlled variable with minimal oscillations, despite selecting medium values for K_C^{FT} and τ_I^{FT} (refer to Table D.2 in Appendix D for variable bounds). In this scenario, the agent was able to achieve a molar fraction of component B using only one FT. It is important to note that while the target value of 0.4 was not attained, the agent proposed a molar fraction of 0.365. Given that the design constraint required the molar fraction to be equal to or less than 0.4 (i.e., $x_L \leq 0.4$), the design proposed by the agent remains valid. Furthermore, since x_L^{sp} is a decision variable that the agent must select, it is unlikely that the agent will precisely choose 0.4 unless the problem formulation explicitly necessitates this value.

4.2.4 Case study 4

This case study builds on the second case study by presenting a process that more closely mirrors real-world applications. The primary objective of this case study is to generate, design and control a CPF that efficiently produces two high-purity products (B and C) from reactant A. Unlike the previous case study, this example presents a more realistic scenario by introducing the agent to more complex synthesis decisions. Key considerations in this case include determining the optimal separation sequence for the ternary mixture, where the agent must decide on the types and order of distillation columns to be included in the flowsheet. Additionally, a recycle stream is incorporated into the separation process, making the synthesis decision more intricate. In this case, the agent must not only choose the separation sequence but also determine when to perform the recycle, further integrating the mass reintegration into the process design. The entire process is conducted isothermally and experiences a +50% step disturbance in the inlet flow rate. This case study is analogous to the isomerization of *n*-pentane into its isomers, *iso*-pentane and neopentane, where separation is facilitated by their boiling point differences; more information is provided in the Appendix E.

In this case, the RL agent has access to four UOs: a CSTR where two first-order reactions ($A \rightarrow B$ and $A \rightarrow C$) occur in parallel, and three different binary DC, each operating at a constant relative volatility to obtain pure streams of the ternary mixture components, all while considering the dynamic models of the CSTR and DCs in closed-loop operation. In this case B is the most volatile component – light key (LK), while A is the least volatile – heavy key (HK). Table 4.11 presents the different values of the relative volatilities (α) of a component with respect to the HK:

Table 4.11 Relative volatilities

Component	α	α
A	1.00	-
B	4.50	2.50
C	3.50	1.00

The dynamic models for the CSTR and DC used in this case study were the same as those employed in case study 2 (Bequette, 2002; Schweiger and Floudas, 1998), only varying the value of the relative volatility in the DC model accordingly. The same mechanistic models as in case study 2 were used, more detailed mathematical formulations of the UO models are provided in Appendix C.

To prevent unnecessary mass loss, the distilled pure A stream was recycled into the system's inlet. It is assumed that the recycle stream passes through a dampening tank before being reintroduced, which mitigates any dynamic effects. For simplicity, the design of the dampening tank is not explicitly considered; however, it is assumed that the tank's volume is large enough to effectively dampen the dynamics. The tank is designed as an overflow tank, which ensures a constant inlet and outlet flow, this assumption eliminates disturbances in the inlet flow caused by the recycling process. By recycling A, mass conservation is maintained, as the mass lost during separation is recovered. Consequently, the initial flowrate for the column performing the recycling will be the same for subsequent UOs. This approach simplifies the analysis while preserving mass balance throughout the system.

For the first scenario (*Scenario H*), the agent was tasked with achieving two outflows, each with a molar fraction of products B and C of 0.99 or higher, having converted A at least to a 50% beforehand. The agent was tasked with effectively rejecting the beforementioned step disturbance of +50% in the inlet flow rate and maintaining the reaction volume within the design limit of 8 liters in the CSTR. Each UO selected by the agent is equipped with its own integrated control system. For the reactor, a PI controller is included, which regulates the outlet concentration of reactant A by adjusting the outlet flow rate. In contrast, the DC is equipped with two controllers: one at the top to regulate the molar fraction of the LK at the distillate by manipulating the reflux flow rate, and another at the bottoms to regulate the molar fraction of the LK at the bottoms by manipulating the vapor boil-up rate. The maximum number of steps available to the agent per episode was 10. Within this limit, the agent had complete freedom to choose the number of reactors required to meet the specified conversion condition as well as the DC necessary to separate the ternary mixture.

For this case study, 19 surrogate models were employed: four for the CSTR and five for each DC. The type and performance of the surrogate models were consistent with those described in Table 4.4. The only difference was that for the DC surrogate models, an additional 10 models were developed to account for the two other relative volatility values, 3.5 and 4.5. The action space (Equation 4.10) for this problem builds upon the action space outlined in the previous section (Equation 4.6), incorporating the binary decision variable (v) that defines the separation sequence to be followed by the agent. Specifically, it determines whether the process should first separate product B from the ternary mixture, followed by a separation of A and C; or alternatively, separate products B and C from A first, with a subsequent separation of the products.

$$\mathbf{A} = [C_A^{sp}, V_{in}^L, K_C^R, \tau_I^R, N, D_C, feed_N, reflux_N, K_C^D, \tau_I^D, K_C^B, \tau_I^B, v] \quad (4.10)$$

As in the previous case study, the binary and discrete variables in this case were approximated using continuous distributions to enable the application of the same framework discussed in earlier sections. The observation vector for this case study contains all pertinent information from the environment so that it can be used for both UOs' types. The information stored in the observation vector includes the outlet concentrations of reactant A and products B and C, the molar fractions of all three components, and the current step (Equation 4.11). Note that the outlet concentrations become the inlet concentrations for the subsequent UO.

$$o_i = [C_A, C_B, C_C, z_A, z_B, z_C, i] \quad (4.11)$$

The reward functions and all specifications for the reactor and DC from case study 2 were retained in the current analysis. It is important to note that the reward functions for the DCs remain identical across all three types of DCs, regardless of their relative volatility values.

The PPO agent was trained for 100,000 steps taking around 62.34 minutes. The actor network consisted of 2 hidden layers, each with 64 neurons, while the critic network had only 1 hidden layer with 64 neurons. The actor-critic networks were updated every 2048 steps with a discount factor of 0.99 and optimized using Adam optimizer (Kingma and Ba, 2017) with a starting learning rate of 2.5e-4, which was adjusted throughout the training.

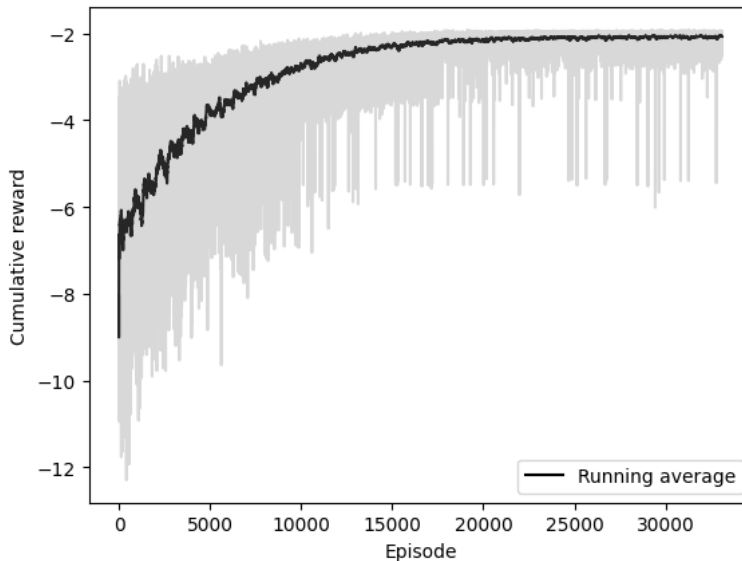


Figure 4.21: PPO agent's learning curve

A comparison of the computational times required for training agents in the previous two case studies showed no significant differences. This finding supports the claim that incorporating additional UOs into the problem formulation did not substantially affect computational time. The learning curve of the PPO agent, depicted in Figure 4.21, shows the cumulative reward versus the number of episodes, highlighting the agent's performance over time. The agent's exploration phase, characterized by the wide spread in the faded grey region, extends over the first 12,000 episodes. Throughout this phase, the agent learns to design CPFs that do not violate the design constraints and that correctly control the system. This learning process is reflected in the steady increase of the running mean. After exploring various CPFs, the next phase of the agent's learning corresponds to the refinement of its designs (exploitation). In this phase, the rate of learning growth is not as immediate as in the exploration phase, but the variability in the designed flowsheets is much lower, as clearly reflected in the shrinkage of the area of the grey faded region – cumulative rewards are more similar.

When evaluating the best-performing agents, the resulting CPF is capable of distilling both products to a molar fraction of 0.99, having previously converted reactant A to 50%; adhere to the design constraints (reaction volume smaller than 8 liters); and produce an economically attractive solution. For this case, the agent required one CSTR and two DCs with the design specifications shown in Table 4.12 (*Scenario H*) and Figure 4.22.

Table 4.12: UOs designs and specifications for case study 3 scenarios F and G

Scenario	Variable	CSTR 1	CSTR 2	DC 1	DC 2
H	V_{in}^L [L]	3.0516	-	-	-
	C_A^{sp} [mol/L]	0.3172	-	-	-
	K_C^R [$L^2/mol \min^{-1}$]	-131.7546	-	-	-
	τ_I^R [$\min^{-1}$]	4.0099	-	-	-
	N	-	-	13	14
	D_C [m]	-	-	0.3865	0.3611
	$feed_N$	-	-	6	5
	$reflux_N$	-	-	12	13
	K_C^D [mol/min]	-	-	-14.5769	-18.7366
	τ_I^D [$\min^{-1}$]	-	-	0.1479	0.0577
	K_C^B [mol/min]	-	-	16.2761	16.7803
	τ_I^B [$\min^{-1}$]	-	-	0.0102	0.0662
I	V_{in}^L [L]	3.8198	6.1889	-	-
	C_A^{sp} [mol/L]	0.3374	0.2407	-	-
	K_C^R [$L^2/mol \min^{-1}$]	-194.7158	-94.9062	-	-
	τ_I^R [$\min^{-1}$]	5.2575	4.5057	-	-
	N	-	-	16	18
	D_C [m]	-	-	0.3046	0.3140
	$feed_N$	-	-	8	8
	$reflux_N$	-	-	15	17
	K_C^D [mol/min]	-	-	-8.7482	-18.9527
	τ_I^D [$\min^{-1}$]	-	-	0.0964	0.1566
	K_C^B [mol/min]	-	-	15.3053	19.3028
	τ_I^B [$\min^{-1}$]	-	-	0.1687	0.0617

In this case, the separation sequence chosen by the agent involved initially isolating the least volatile component of the mixture, corresponding to the simplest separation task, followed by the subsequent separation of the remaining products. Notably, the agent arrived at this strategy through its learning process, without prior knowledge of or explicit access to heuristic rules that typically guide such decisions. This outcome highlights the agent's ability to independently identify an efficient

separation strategy, aligning with established process engineering principles that often favor starting with the simplest separation to minimize operational complexity (Douglas, 1988; Smith, 2005).

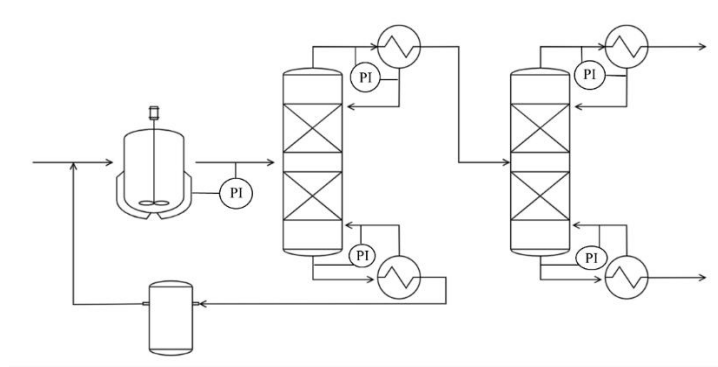


Figure 4.22: Final flowsheet Scenario H

The dynamic responses for the mechanistic models of the CSTR and the first and second DCs using the design parameters outlined in Table 4.12 are illustrated in Figures 4.23, 4.24, and 4.25, respectively.

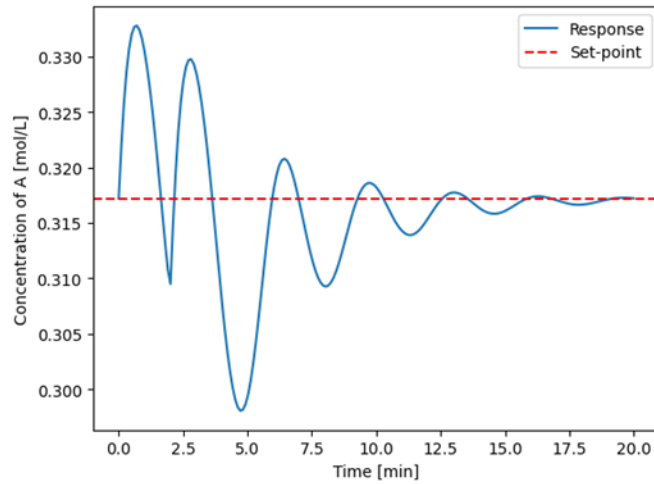


Figure 4.23. Dynamic response Scenario H of CSTR

For the CSTR designed by the agent, the disturbance is successfully rejected while operating at the established set-point (Figure 4.23). In this case, the agent selects a relatively small V_{in}^L to avoid exceeding the design volume limit of 8 liters (see Figure E.1 in Appendix E). Despite this precaution, the volume undergoes significant fluctuations, reaching up to 7 liters before eventually reaching a new steady-state condition. In RL, there is no single correct answer. Unlike model-based optimization,

which returns a single solution, RL samples actions from a probability distribution, leading to slight variations in the agent's responses. In some tests, the agent selected a C_A^{sp} value closer to 0.5. Combined with relatively stable values for other design variables, this resulted in responses slightly different from those shown in Figure 4.23 and Figure E.1. Specifically, the dynamic response exhibited greater oscillations, while volume fluctuations were minimal, reaching a maximum of 4 liters. This highlights a clear trade-off in the agent's learning process: higher dynamic costs (reflected in increased oscillations and a higher ISE value) corresponded to smaller volume fluctuations, reducing the risk of violating the design constraint. Regarding process control, the reactor employs medium-sized K_C^R and τ_I^R values (see Table C.2 in Appendix C for variable bounds). This control scheme induces oscillatory behavior in the response variable before approaching the set-point.

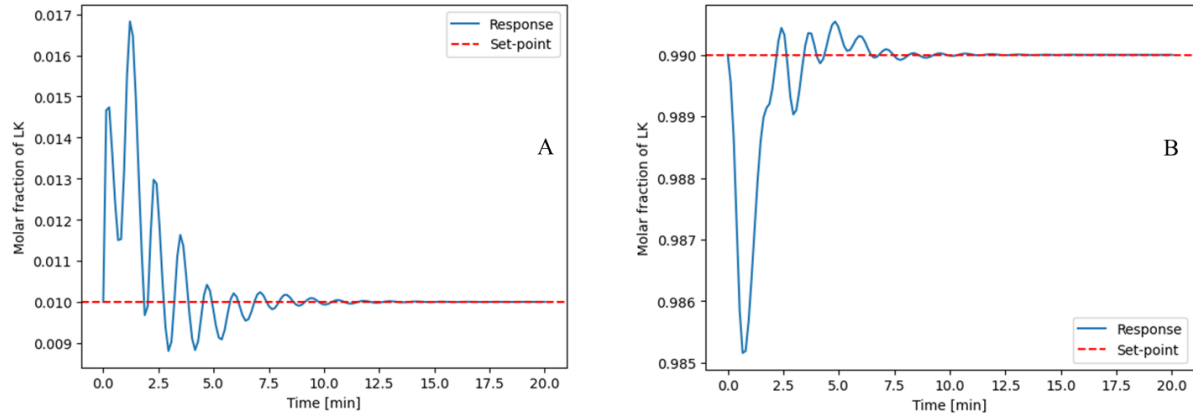


Figure 4.24. Dynamic responses Scenario H of DC 1: A) Bottoms B) Overheads

As previously mentioned, the first DC separates products B and C from reactant A. The dynamic profiles in Figure 4.24 illustrate the response of the LK at both ends of the column, where the LK in this case consists of the two products. Similar to the reactor, the first distillation column successfully rejects inlet disturbances while maintaining operation at the designated set point. For its design, the agent selected a column with a small diameter and a number of stages close to the minimum – which was 10 in this case. Post-analysis of the results confirmed that achieving molar fractions of 0.99 at both ends of the column did not require an excessive number of stages, aligning with the agent's design decisions. However, the agent could have opted for a column with 12 stages, which would also have met the separation requirements. Additionally, while the selected column diameter was sufficient to handle the operating conditions, a slightly smaller diameter could have been chosen to reduce costs without compromising performance. The control strategy chosen by the agent allows for small-

amplitude oscillations in the response before approaching the set point, which occurs within 10 minutes. Notably, the dynamic response for the bottoms exhibits more oscillations compared to that of the overheads. This difference may be attributed to the fact that the time integral value for the bottoms is ten times smaller than that for the overheads.

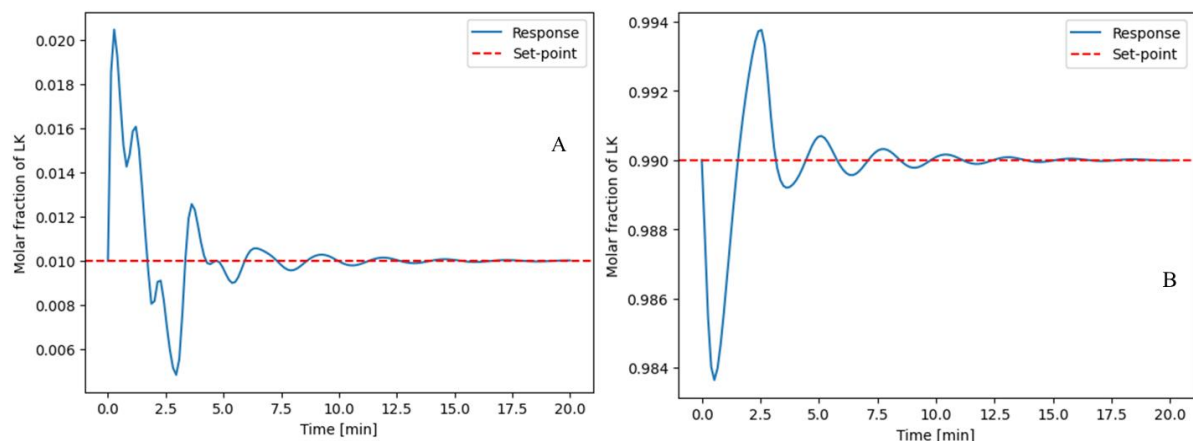


Figure 4.25. Dynamic responses Scenario H of DC 2: A) Bottoms B) Overheads

In the second DC, products B and C were separated, with the dynamic profiles of the LK, in this case product B, shown in Figure 4.25. Similar to the previous UOs, this column successfully rejects disturbances in the inlet stream while maintaining operation around the specified set points. The design of the second column is similar to that of the first, with the only difference being the number of stages, as the agent selects one additional stage. Notably, the feed stage is positioned at the middle of the column. This decision aligns with expectations, as the molar fraction of product B is 0.53, providing adequate enrichment and depletion stages for both products, thereby ensuring effective separation. Regarding the dynamic response, unlike the first column, the control performed on both the tops and the overhead is similar, as reflected in the response profiles. Both exhibit an oscillation with a large amplitude at the start of the simulation, followed by a series of subsequent oscillations with decreasing amplitude until reaching a steady state around 12.5 minutes. The gain constant values for both controllers are at the upper end of the absolute value range, while the time integral values are closer to the lower end, near the minimum (see Table E.1 in the Appendix E).

To further enhance the work, and considering the significant advantage offered by the proposed framework – which allows the use of surrogate models across various problems – a new scenario was addressed. In *Scenario I*, the agent was tasked to achieve a 75% conversion of reactant A, while maintaining the 0.99 molar fraction value for the outlet streams of products B and C. The design

constraint of 8 liters remained unchanged, and the agent was allowed to use a maximum of 10 steps to achieve the corresponding design specifications. Having established the new conversion goal, the PPO agent, with the same specifications, was trained for 100,000 steps for the course of 65.78 minutes. The actor-critic networks were updated every 2048 steps with a discount factor of 0.99 and optimized using the Adam optimizer (Kingma and Ba, 2017), starting with a learning rate of $2.5e-4$, which was adjusted throughout the training.

For this case, the agent required two CSTRs and two DCs with the design specifications shown in Table 4.12 (*Scenario I*) and an overall flowsheet as depicted in Figure 4.26.

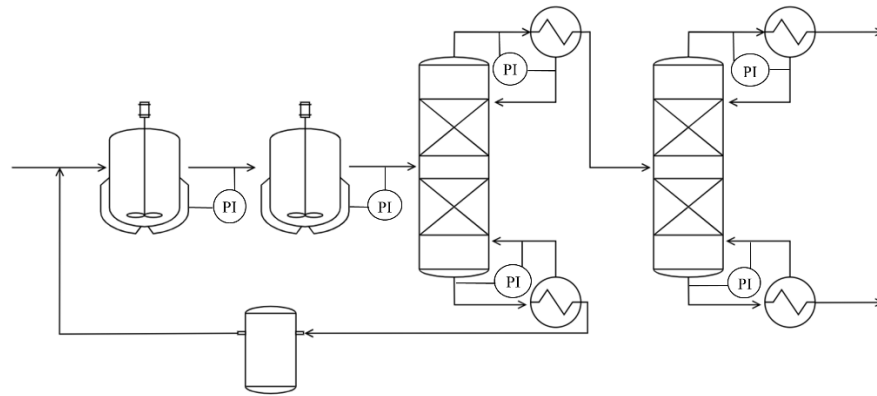


Figure 4.26. Final flowsheet for Scenario I

For the two CSTRs designed by the agent, the disturbance is successfully rejected while operating at the established set-point (Figure 4.27). Notably, the agent could have chosen one reactor to achieve the desired conversion of A with fewer units; however, due to the volume constraint, the agent opted to include two reactors.

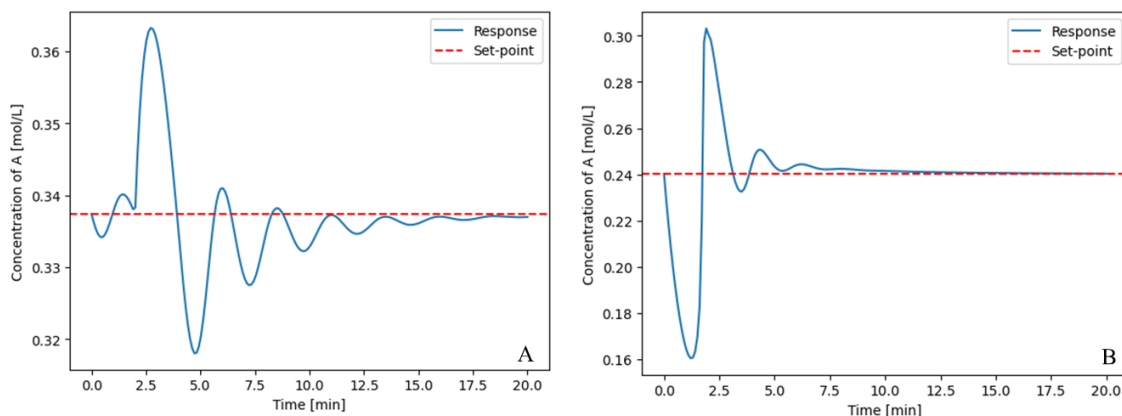


Figure 4.27. Dynamic responses Scenario I of: A) CSTR 1 B) CSTR 2

The first reactor experiences a significant volume swing, reaching up to 7 liters before reaching a new lower steady-state liquid volume in the reactor. In contrast, the second reactor exhibits the opposite behavior, with a substantial volume decrease, starting from a high V^L value and reaching a liquid level around 2 liters. Nevertheless, neither reactor violates the design constraints (see Figure E.2 in the Appendix E). Regarding control behavior, the reactors exhibit distinct dynamic responses. The first reactor demonstrates an oscillatory response that reaches a new steady-state at around 17.5 minutes, whereas the second reactor shows a pronounced oscillation in the initial moments, followed by smooth decay towards the set-point. This difference in behavior can be attributed to variations in their relative gains, with the first reactor's gain being 100 units higher than that of the second. Additionally, the degree of conversion achieved in each reactor differs significantly: the first reactor handles the majority of the conversion, while the second only further reduces the concentration of reactant A from 0.33 to 0.24.

In the purification process, as in the previous scenario, the agent prioritizes separating reactant A, recycling it back into the input stream. Subsequently, the more complex separation of products B and C is performed. Figure 4.28 illustrates the dynamic profiles of the LK component in the bottoms and overhead streams of the first DC, which operates with a constant relative volatility of 3.5.

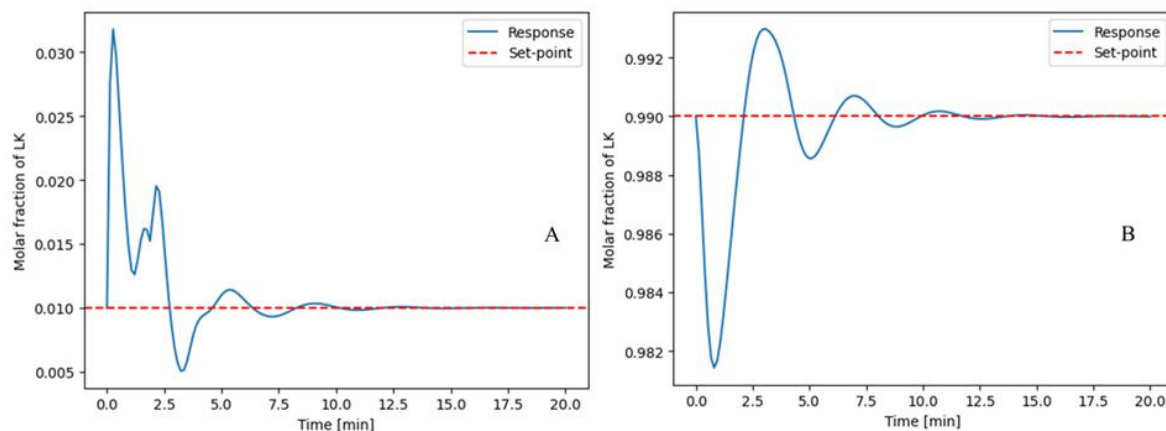


Figure 4.28. Dynamic responses Scenario I of DC 1: A) Bottoms B) Overheads

The proposed DC effectively rejects disturbances in the inlet flowrate while maintaining operation around the specified set points. Regarding its design, the column selected by the agent differs from the two previously proposed designs for *Scenario H*. In this case, the DC features a higher number of stages – 16 – and a narrower diameter, close to the lower bound of 0.3 meters. The increased number of stages is justified by the higher concentration of the mixture to be separated compared to the previous scenario, as a greater number of stages enhances separation efficiency in systems with higher feed concentration. The control strategy implemented by the agent is relatively mild, resulting in some oscillatory behavior in the dynamic response at both ends of the column. Notably, the oscillations in the bottoms exhibit molar fraction swings of up to 0.02, compared to the maximum swings of 0.01 observed in the overheads. This behavior can be attributed to a smaller relative gain in the distillates, a key difference from the DCs in the previous scenario.

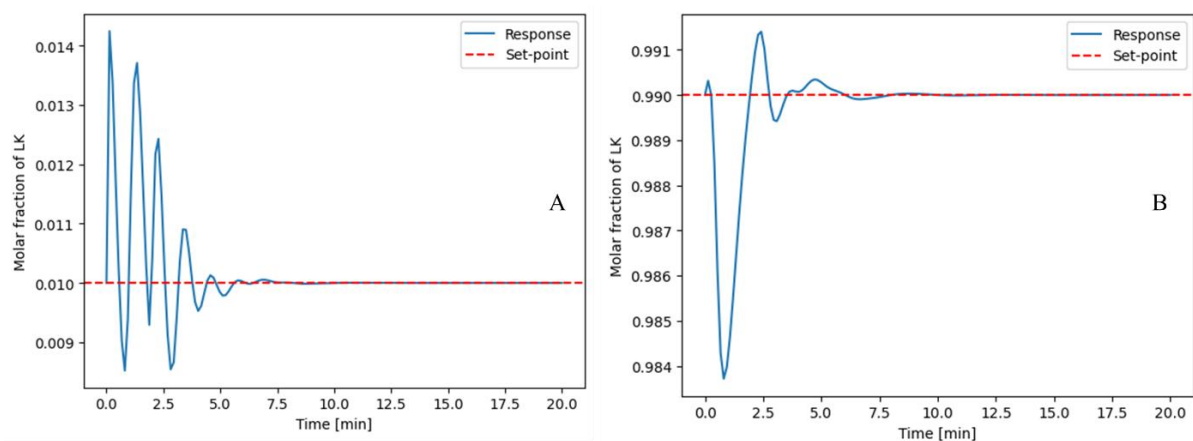


Figure 4.29. Dynamic responses Scenario I of DC 2: A) Bottoms B) Overheads

The second DC effectively rejects disturbances in the inlet flowrate while maintaining operation near the specified set points. Compared to the first DC, the second column features a slightly larger number of stages and a marginally wider diameter. In terms of control, the second DC employs a significantly more aggressive control strategy than both the first DC and the columns from the previous scenario. This is clearly reflected in the short closed-loop settling times, with steady-state conditions achieved in both the overheads and bottoms within approximately 6 minutes. A comparison of controller parameters reveals that the relative gain for the overheads in the second DC is 10 units larger than in the first DC, as shown in Table 7. Similarly, the bottoms controller also exhibits a larger gain for the second DC, enabling sharper responses and shorter closed-loop settling times.

4.3 Summary

In this chapter, a novel RL strategy was introduced for the simultaneous generation, design, and control of CPFs starting from an inlet stream and a set of UOs. The first section outlines the solution framework of the proposed methodology, placing particular emphasis on the innovative aspect of this method: the surrogate models. This section elaborates on the data generation process utilizing LHS to train the surrogate models, detailing the various types of surrogate models employed and their techniques for approximating the mechanistic models of UOs in dynamic states. The remaining components of the RL framework are described, emphasizing the reward function and its various sub-rewards, which capture the economic and dynamic elements, as well as the satisfaction of the problem's constraints. The proposed framework was illustrated using four case studies. The first case study aimed to validate the proposed framework through a comparative analysis with a model-based optimization method. Although the agent did not attain the optimal solution identified by the model-based optimization approach, it successfully replicated the essential flowsheet configuration – i.e., the arrangement of UOs. Despite this discrepancy, the proposed CPF adhered rigorously to all prescribed design and operational constraints, effectively controlled operations at the specified set points, and yielded an economically attractive solution. The second, third and fourth case studies addressed a more intricate problem involving the simultaneous design and control of multiple UOs. In these case studies, the advantage of using surrogate models was demonstrated by simplifying a complex mechanistic model, like the DC, into multiple and more manageable surrogate models for evaluation. In the three case studies the proposed CPFs achieved designs that met design constraints, effectively controlled operations around specified set points and proposed economically attractive solutions. Notably, these case studies revealed that adding an extra UO to the agent's action space does not impact the training time of the agent;

however, it does impact the agent’s learning process, resulting in greater variability in cumulative rewards. Case studies 3 and 4, in particular, highlighted this phenomenon. One of the disadvantages of this method is the long simulation times. The most time-consuming process is the generation of the data need for training the surrogate models, taking up to 8 hours for the case studies considered in this work. In contrast, the training of the surrogate models themselves and the RL agent do not impose a considerable time burden, typically requiring no more than one hour. A notable benefit of this methodology is that once a surrogate model is trained, it can be applied to various problems, demonstrating the flexibility of the proposed approach. However, users must ensure that the new problem falls within the predictive range covered by these surrogate models.

Chapter 5

Conclusions and Future Work

The primary objective of this thesis was to investigate the potential of RL as a tool for addressing the generation, design, and control of CPFs in both steady and transient domains. Two distinct methodologies were developed in this thesis, both capable of generating CPFs from an initial flowrate and a predefined set of UOs.

In Chapter 3, a novel methodology for steady-state CPF design was introduced. Two advanced RL agents were implemented: a fully discrete agent (mPPO) and a hybrid agent (mHPPO). Key contributions include the integration of masking – an underexplored technique in CPF design – and the development of a hybrid platform (Python-ASPEN Plus®) for rigorous simulation of UOs. Three case studies were conducted, demonstrating that both agents successfully learned to design CPF that attained to design/operational constraints while generating economically viable solutions. In all cases, masking proved to be essential, allowing the agents to capture interdependencies between non-adjacent UOs, enhancing their learning and the design outcomes. The following insights emerged from this research:

- The choice of the agent depends on the specific problem under consideration. For scenarios with fewer discrete and continuous decisions, the mPPO agent outperforms the mHPPO by achieving near-optimal solutions, leveraging a more constrained combinatorial search space. In contrast, when the problem involves a large number of discrete and continuous variables, the mHPPO, despite obtaining lower cumulative rewards, generates superior CPF designs by enhancing mass integration and minimizing product losses.
- The mHPPO agent balances discrete and continuous losses during training, with the discrete loss playing a dominant role in shaping the final flowsheet. The arrangement of UOs is heavily influenced by discrete decisions, which define the structural composition early in the design process. However, the trade-off between discrete and continuous losses limits further optimization, as continuous improvements are constrained once the discrete loss stabilizes. This trade-off explains why the mHPPO does not achieve the same results as the mPPO, which benefits from a simpler, fully discrete search space.
- Masking was crucial for enabling the agent to learn effectively. In complex problems involving multiple UOs, incorporating heuristics in the form of masking significantly improved the agent's learning process. Without masking, the agent would naïvely select flawed flowsheet

designs that failed immediately, minimizing penalties rather than producing viable solutions. Notably, while masking aids learning by guiding the agent toward better decisions, it does not predetermine or fix the flowsheet design *a priori*.

In Chapter 4, a novel RL strategy was introduced for the simultaneous generation, design, and control of CPFs in transient states. The primary contribution of this research is a methodology that captures the dynamic behavior of UOs through representative data, enabling the generation of surrogate models – embedded in the RL environment – applicable to a wide variety of UOs. These models effectively represent output variables, dynamic performance metrics, and operational constraints – a gap not previously addressed in the literature on integration of design and control. Four case studies demonstrated that the RL approach successfully designed and controlled CPFs while adhering to design and operational constraints and producing economically viable solutions. Additionally, the use of surrogate models significantly expedited simulation times and simplified the evaluation of environmental impacts. The following insights emerged from this research:

- The use of surrogate models was demonstrated by simplifying complex mechanistic models, such as DC, into multiple manageable surrogate models for evaluation. Their flexibility allows these models to be applied to various scenarios, provided that the data used for training remains within the predictive range covered by the surrogate models.
- While the proposed methodology was not able to achieve the optimal solution, it offers a close approximation, as validated by a comparative analysis with a model-based optimization method. This underscores the effectiveness of the methodology in delivering practical solutions in more complex scenarios where model-based optimization might be computational expensive.
- Case studies revealed that adding an extra UO to the agent’s action space does not affect the agent’s training time. However, it does influence the learning process, leading to greater variability in cumulative rewards.
- Normalizing the cost and dynamic cost sub-rewards in the reward function is crucial for maintaining balanced contributions from both components. This ensures that the agent accurately navigates the trade-offs encountered in the integrated design and control problem, avoiding bias towards a certain size of unit or control strategy.

5.1 Recommendations for future work

The insights gained through this thesis have demonstrated an important number of applications for the proposed methodologies; however, potential improvements can be considered as part of future work. The mayor recommendations derived from this research are as follows:

- Reducing computational times

One of the primary areas for improvement and a significant avenue for future research is the reduction of computational times required for both proposed methodologies outlined in Chapters 3 and 4. Beginning with Chapter 3, considerable time was spent training the agent within the hybrid Python-Aspen Plus® platform. A key research direction would be to enhance the agent's training efficiency by refining training algorithms or incorporating techniques such as parallel processing or adaptive learning strategies. Focusing on these improvements would accelerate the workflow and enhance the agent's learning from the surrogate models, resulting in more robust and efficient design and control of chemical process flowsheets. Conversely, for the methodology presented in Chapter 4, the most time-consuming aspect is not the training of the RL agent but rather the data generation required to train the surrogate models, particularly when employing optimization programs such as Gekko (Beal et al., 2018). In this context, a potential area for enhancement would be to find ways to streamline the simulations, perhaps by utilizing parallel computing across multiple machines to obtain the necessary data without compromising the functionality of Gekko. Another possibility could involve the use of historical data from similar processes instead of mechanistic models. This approach is particularly attractive for reducing CPU times, as surrogate models can be directly trained on pre-existing datasets rather than waiting for synthetic data to be generated through simulations. As discussed above, the generation of simulation data is often the most time-consuming step in the modeling process. Addressing these computational time challenges will enhance the practicality and applicability of the proposed methodologies in real-world scenarios.

- Process uncertainty

An intriguing avenue for future research involves the integration of process uncertainty into the methodologies developed in this thesis. The nature of chemical processes often entails inherent uncertainties that can significantly affect operational outcomes. RL has demonstrated considerable promise in managing such uncertainties due to its adaptive learning capabilities. By incorporating stochastic elements or uncertain parameters within the RL framework, future investigations could assess the robustness and resilience of the proposed methodologies under varying conditions. This

exploration could lead to a deeper understanding of the methodologies' performance in realistic settings, thus expanding their applicability in industry.

- RL with human expert feedback

The potential incorporation of human expert feedback into the RL framework presents another valuable research avenue. In particular, this could significantly enhance the learning process of agents, especially in the context of the methodology outlined in Chapter 3, where the learning phase is both time-consuming and exhibits considerable variability. By integrating expert knowledge, agents could be guided more effectively, accelerating their learning processes and improving their decision-making capabilities. However, it is important to acknowledge the trade-off involved in this approach; relying on human feedback may compromise the goal of full automation in the process.

- CPF retrofitting

Plant retrofitting represents an interesting avenue for research. The primary idea is to modify existing CPFs rather than designing them from scratch, as is typically done in the proposed methodologies. A potential modification worth considering is the implementation of hierarchical agents that can navigate the CPF and facilitate the application of relevant changes. This approach could enable a more adaptive and efficient means of optimizing existing processes, leveraging the foundational work already established in the CPF while integrating advanced methodologies for improved performance.

- RL integrated with a Dynamic simulator

An exciting yet challenging research direction involves establishing a hybrid platform that integrates a connection between a dynamic simulator and Python within the RL environment. In this case, the use of surrogate models would be eliminated, allowing for direct simulation within the dynamic simulator, which could enable a more accurate representation of dynamic behaviors in chemical processes. Integrating Aspen Plus® dynamics would be particularly intriguing; however, it requires the manual creation of the file where the dynamic process will be simulated, in addition to requiring a steady-state version beforehand. Aspen HYSYS® presents an interesting alternative, as this platform simulates operation by operation in dynamic state, without the need of generating a steady-state file.

References

- Agyeman, B.T., Naouri, M., Appels, W.M., Liu, J., Shah, S.L., 2024. Learning-based multi-agent MPC for irrigation scheduling. *Control Engineering Practice* 147, 105908. <https://doi.org/10.1016/j.conengprac.2024.105908>
- Alhammadi, H.Y., Romagnoli, J.A., 2004. Process design and operation: Incorporating environmental, profitability, heat integration and controllability considerations, in: *Computer Aided Chemical Engineering*. Elsevier, pp. 264–305. [https://doi.org/10.1016/S1570-7946\(04\)80063-4](https://doi.org/10.1016/S1570-7946(04)80063-4)
- Alhazmi, K., Albalawi, F., Sarathy, S.M., 2022. A reinforcement learning-based economic model predictive control framework for autonomous operation of chemical reactors. *Chemical Engineering Journal* 428, 130993. <https://doi.org/10.1016/j.cej.2021.130993>
- Andersen, B.B., Nielsen, R.F., Udugama, I.A., Papadakis, E., Gernaey, K.V., Huusom, J.K., Mansouri, S.S., Abildskov, J., 2018. Integrated Process Design and Control of Cyclic Distillation Columns. *IFAC-PapersOnLine* 51, 542–547. <https://doi.org/10.1016/j.ifacol.2018.09.368>
- Andrés-Martínez, O., Ricardez-Sandoval, L.A., 2022. Integration of planning, scheduling, and control: A review and new perspectives. *The Canadian Journal of Chemical Engineering* 100, 2057–2070. <https://doi.org/10.1002/cjce.24501>
- Aspen Technology Inc, n.d. ASPEN Plus.
- Bahakim, S.S., Ricardez-Sandoval, L.A., 2014. Simultaneous design and MPC-based control for dynamic systems under uncertainty: A stochastic approach. *Computers & Chemical Engineering* 63, 66–81. <https://doi.org/10.1016/j.compchemeng.2014.01.002>
- Bansal, V., Perkins, J.D., Pistikopoulos, E.N., 2002. A Case Study in Simultaneous Design and Control Using Rigorous, Mixed-Integer Dynamic Optimization Models. *Ind. Eng. Chem. Res.* 41, 760–778. <https://doi.org/10.1021/ie010156n>
- Bansal, V., Sakizlis, V., Ross, R., Perkins, J., Pistikopoulos, E., 2003. New Algorithms for Mixed-Integer Dynamic Optimization. *Computers & Chemical Engineering* 27, 647–668. [https://doi.org/10.1016/S0098-1354\(02\)00261-2](https://doi.org/10.1016/S0098-1354(02)00261-2)
- Bao, Y., Zhu, Y., Qian, F., 2021. A Deep Reinforcement Learning Approach to Improve the Learning Performance in Process Control. *Ind. Eng. Chem. Res.* 60, 5504–5515. <https://doi.org/10.1021/acs.iecr.0c05678>
- Beal, L., Hill, D., Martin, A., Hedengren, J., 2018. GEKKO optimization suite. *Processes* 6, 106. <https://doi.org/10.3390/pr6080106>
- Bequette, B., 2002. *Process control: modeling, design, and simulation*, First. ed. Prentice Hall Press, USA.
- Bernal, D.E., Carrillo-Díaz, C., Gómez, J.M., Ricardez-Sandoval, L.A., 2018. Simultaneous Design and Control of Catalytic Distillation Columns Using Comprehensive Rigorous Dynamic Models. *Ind. Eng. Chem. Res.* 57, 2587–2608. <https://doi.org/10.1021/acs.iecr.7b04205>
- Bernal, D.E., Ovalle, D., Liñán, D.A., Ricardez-Sandoval, L.A., Gómez, J.M., Grossmann, I.E., 2022. Process Superstructure Optimization through Discrete Steepest Descent Optimization: a GDP Analysis and Applications in Process Intensification, in: Yamashita, Y., Kano, M. (Eds.), *Computer Aided Chemical Engineering, 14 International Symposium on Process Systems Engineering*. Elsevier, pp. 1279–1284. <https://doi.org/10.1016/B978-0-323-85159-6.50213-X>
- Bertran, M.-O., Frauzem, R., Zhang, L., Gani, R., 2016. A Generic Methodology for Superstructure Optimization of Different Processing Networks, in: Kravanja, Z., Bogataj, M. (Eds.), *Computer Aided Chemical Engineering, 26 European Symposium on Computer Aided Process Engineering*. Elsevier, pp. 685–690. <https://doi.org/10.1016/B978-0-444-63428-3.50119-3>

- Blanco, A.M., Bandoni, J.A., 2007. Eigenvalue optimization-based formulations for nonlinear dynamics and control problems. *Chemical Engineering and Processing: Process Intensification* 46, 1192–1199. <https://doi.org/10.1016/j.cep.2006.04.014>
- Boulesnane, A., Meshoul, S., 2021. Reinforcement learning for dynamic optimization problems, in: *Proceedings of the Genetic and Evolutionary Computation Conference Companion, GECCO '21*. Association for Computing Machinery, New York, NY, USA, pp. 201–202. <https://doi.org/10.1145/3449726.3459543>
- Brüggemann, S., Wolfgang, M., 2004. Rapid screening of design alternatives for nonideal multiproduct distillation processes. *Computers & Chemical Engineering, PSE* 29, 165–179. <https://doi.org/10.1016/j.compchemeng.2004.07.009>
- Burnak, B., Diangelakis, N.A., Katz, J., Pistikopoulos, E.N., 2019. Integrated process design, scheduling, and control using multiparametric programming. *Computers & Chemical Engineering* 125, 164–184. <https://doi.org/10.1016/j.compchemeng.2019.03.004>
- Bynum, M.L., Hackebeil, G.A., Hart, W.E., Laird, C.D., Nicholson, B.L., Sirola, J.D., Watson, J.-P., Woodruff, D.L., 2021. *Pyomo — Optimization Modeling in Python*, Springer Optimization and Its Applications. Springer International Publishing, Cham. <https://doi.org/10.1007/978-3-030-68928-5>
- Caballero, J., Odjo, A., Grossmann, I., 2007. Flowsheet optimization with complex cost and size functions using process simulators. *AIChE Journal* 53, 2351–2366. <https://doi.org/10.1002/aic.11262>
- Caballero, J.A., Grossmann, I.E., 1999. Aggregated Models for Integrated Distillation Systems. *Ind. Eng. Chem. Res.* 38, 2330–2344. <https://doi.org/10.1021/ie980803j>
- Chaffart, D., Yuan, Y., Ricardez-Sandoval, L.A., 2024. Multiscale Physics-Informed Neural Network Framework to Capture Stochastic Thin-Film Deposition. *J. Phys. Chem. C* 128, 3733–3750. <https://doi.org/10.1021/acs.jpcc.3c07168>
- Chanona, E.A. del R., Petsagkourakis, P., Bradford, E., Graciano, J.E.A., Chachuat, B., 2021. Real-time optimization meets Bayesian optimization and derivative-free optimization: A tale of modifier adaptation. *Computers & Chemical Engineering* 147, 107249. <https://doi.org/10.1016/j.compchemeng.2021.107249>
- Chawankul, N., Budman, H., Douglas, P.L., 2005. The integration of design and control: IMC control and robustness. *Computers & Chemical Engineering* 29, 261–271. <https://doi.org/10.1016/j.compchemeng.2004.08.034>
- Chawankul, N., Sandoval, L.R., Budman, H., Douglas, P.L., 2007. Integration of Design and Control: A Robust Control Approach Using MPC. *The Canadian Journal of Chemical Engineering* 85, 433–446. <https://doi.org/10.1002/cjce.5450850406>
- Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P., 2002. SMOTE: Synthetic Minority Over-sampling Technique. *jair* 16, 321–357. <https://doi.org/10.1613/jair.953>
- Chen, Q., Grossmann, I.E., 2017. Recent Developments and Challenges in Optimization-Based Process Synthesis. *Annual Review of Chemical and Biomolecular Engineering* 8, 249–283. <https://doi.org/10.1146/annurev-chembioeng-080615-033546>
- Chen, Q., Liu, Y., Seastream, G., Sirola, J.D., Grossmann, I.E., 2021. Pyosyn: A new framework for conceptual design modeling and optimization. *Computers & Chemical Engineering* 153, 107414. <https://doi.org/10.1016/j.compchemeng.2021.107414>
- Chi, K., Liu, H., Ma, H., Qu, W., Wang, C., Lv, G., Tian, Z., 2017. Skeletal Isomerization of n -Pentane: A Comparative Study on Catalytic Properties of Pt/WO_x-ZrO₂ and Pt/ZSM-22. *Applied Catalysis A: General* 537. <https://doi.org/10.1016/j.apcata.2017.03.005>
- Contreras-Zarazúa, G., Sánchez-Ramírez, E., Vázquez-Castillo, J.A., Ponce-Ortega, J.M., Errico, M., Kiss, A.A., Segovia-Hernández, J.G., 2019. Inherently Safer Design and Optimization of

- Intensified Separation Processes for Furfural Production. *Ind. Eng. Chem. Res.* 58, 6105–6120. <https://doi.org/10.1021/acs.iecr.8b03646>
- Contreras-Zarazúa, G., Vázquez-Castillo, J.A., Ramírez-Márquez, C., Segovia-Hernández, J.G., Alcántara-Ávila, J.R., 2017. Multi-objective optimization involving cost and control properties in reactive distillation processes to produce diphenyl carbonate. *Computers & Chemical Engineering, Process Intensification* 105, 185–196. <https://doi.org/10.1016/j.compchemeng.2016.11.022>
- d’Anterrosches, L., Gani, R., 2005. Group contribution based process flowsheet synthesis, design and modelling. *Fluid Phase Equilibria, PPEPPD 2004 Proceedings* 228–229, 141–146. <https://doi.org/10.1016/j.fluid.2004.08.018>
- D. Machalek, T. Quah, K. M. Powell, 2020. Dynamic Economic Optimization of a Continuously Stirred Tank Reactor Using Reinforcement Learning, in: 2020 American Control Conference (ACC). Presented at the 2020 American Control Conference (ACC), pp. 2955–2960. <https://doi.org/10.23919/ACC45564.2020.9147706>
- Diangelakis, N.A., Pistikopoulos, E.N., 2017. A multi-scale energy systems engineering approach to residential combined heat and power systems. *Computers & Chemical Engineering* 102, 128–138. <https://doi.org/10.1016/j.compchemeng.2016.10.015>
- Douglas, n.d. J.M. Douglas. *Conceptual Design of Chemical Processes*, chapter 8. McGraw-Hill, 1988.
- Douglas, J., 1988. *Conceptual Design of Chemical Processes*. McGraw-Hill Book Company.
- Fan, Z., Su, R., Zhang, W., Yu, Y., 2019. Hybrid Actor-Critic Reinforcement Learning in Parameterized Action Space.
- Faria, R.R., Capron, B.D.O., Secchi, A.R., De Souza, M.B., 2024. A data-driven tracking control framework using physics-informed neural networks and deep reinforcement learning for dynamical systems. *Engineering Applications of Artificial Intelligence* 127, 107256. <https://doi.org/10.1016/j.engappai.2023.107256>
- Gao, Q., Schweidtmann, A.M., 2024. Deep reinforcement learning for process design: Review and perspective. *Current Opinion in Chemical Engineering* 44, 101012. <https://doi.org/10.1016/j.coche.2024.101012>
- Gao, Q., Yang, H., Shanbhag, S.M., Schweidtmann, A.M., 2023. Transfer learning for process design with reinforcement learning, in: Kokossis, A.C., Georgiadis, M.C., Pistikopoulos, E. (Eds.), *Computer Aided Chemical Engineering, 33 European Symposium on Computer Aided Process Engineering*. Elsevier, pp. 2005–2010. <https://doi.org/10.1016/B978-0-443-15274-0.50319-X>
- García, C.E., Prett, D.M., Morari, M., 1989. Model predictive control: Theory and practice—A survey. *Automatica* 25, 335–348. [https://doi.org/10.1016/0005-1098\(89\)90002-2](https://doi.org/10.1016/0005-1098(89)90002-2)
- Göttl, Q., Grimm, D.G., Burger, J., 2022a. Automated synthesis of steady-state continuous processes using reinforcement learning. *Front. Chem. Sci. Eng.* 16, 288–302. <https://doi.org/10.1007/s11705-021-2055-9>
- Göttl, Q., Grimm, D.G., Burger, J., 2022b. Using Reinforcement Learning in a Game-like Setup for Automated Process Synthesis without Prior Process Knowledge, in: Yamashita, Y., Kano, M. (Eds.), *Computer Aided Chemical Engineering, 14 International Symposium on Process Systems Engineering*. Elsevier, pp. 1555–1560. <https://doi.org/10.1016/B978-0-323-85159-6.50259-1>
- Göttl, Q., Pirnay, J., Burger, J., Grimm, D.G., 2024. Deep Reinforcement Learning Enables Conceptual Design of Processes for Separating Azeotropic Mixtures Without Prior Knowledge. <https://doi.org/10.2139/ssrn.4776784>
- Grossmann, I.E., 1985. Mixed-integer programming approach for the synthesis of integrated process flowsheets. *Computers & Chemical Engineering* 9, 463–482. [https://doi.org/10.1016/0098-1354\(85\)80023-5](https://doi.org/10.1016/0098-1354(85)80023-5)

- Grossmann, I.E., Aguirre, P.A., Barttfeld, M., 2005. Optimal synthesis of complex distillation columns using rigorous models. *Computers & Chemical Engineering, Selected Papers Presented at the 14th European Symposium on Computer Aided Process Engineering* 29, 1203–1215. <https://doi.org/10.1016/j.compchemeng.2005.02.030>
- Hamid, M.K.A., Sin, G., Gani, R., 2010. Integration of process design and controller design for chemical processes using model-based methodology. *Computers & Chemical Engineering* 34, 683–699. <https://doi.org/10.1016/j.compchemeng.2010.01.016>
- Hassanpour, H., Wang, X., Corbett, B., Mhaskar, P., 2024. A practically implementable reinforcement learning-based process controller design. *AIChE Journal* 70, e18245. <https://doi.org/10.1002/aic.18245>
- Herrera Velázquez, J.J., Zavala Durán, F.M., Chávez Díaz, L.A., Cabrera Ruiz, J., Alcántara Avila, J.R., 2022. Hybrid two-step optimization of internally heat-integrated distillation columns. *Journal of the Taiwan Institute of Chemical Engineers* 130. <https://doi.org/10.1016/j.jtice.2021.06.061>
- Hoskins, J.C., Himmelblau, D.M., 1992. Process control via artificial neural networks and reinforcement learning. *Computers & Chemical Engineering* 16, 241–251. [https://doi.org/10.1016/0098-1354\(92\)80045-B](https://doi.org/10.1016/0098-1354(92)80045-B)
- Huang, S., Ontañón, S., 2022. A Closer Look at Invalid Action Masking in Policy Gradient Algorithms. *FLAIRS* 35. <https://doi.org/10.32473/flairs.v35i.130584>
- Hubbs, C.D., Li, C., Sahinidis, N.V., Grossmann, I.E., Wassick, J.M., 2020. A deep reinforcement learning approach for chemical production scheduling. *Computers & Chemical Engineering* 141, 106982. <https://doi.org/10.1016/j.compchemeng.2020.106982>
- Iftakher, A., Liñán, D.A., Mansouri, S.S., Nahid, A., Hasan, M.M.F., Choudhury, M.A.A.S., Ricardez-Sandoval, L.A., Lee, J.H., 2022. RD-toolbox: A computer aided toolbox for integrated design and control of reactive distillation processes. *Computers & Chemical Engineering* 164, 107869. <https://doi.org/10.1016/j.compchemeng.2022.107869>
- Jang, B., Kim, M., Harerimana, G., Kim, J.W., 2019. Q-Learning Algorithms: A Comprehensive Classification and Applications. *IEEE Access* 7, 133653–133667. <https://doi.org/10.1109/ACCESS.2019.2941229>
- Kastsian, D., Mönnigmann, M., 2013. Robust optimization of periodically operated nonlinear uncertain processes. *Chemical Engineering Science* 106. <https://doi.org/10.1016/j.ces.2013.11.023>
- Khan, A., Lapkin, A., 2020. Searching for optimal process routes: A reinforcement learning approach. *Computers & Chemical Engineering* 141, 107027. <https://doi.org/10.1016/j.compchemeng.2020.107027>
- Khan, A.A., Lapkin, A.A., 2022. Designing the process designer: Hierarchical reinforcement learning for optimisation-based process design. *Chemical Engineering and Processing - Process Intensification* 180, 108885. <https://doi.org/10.1016/j.cep.2022.108885>
- Kim, S., Jang, M.-G., Kim, J.-K., 2023. Process design and optimization of single mixed-refrigerant processes with the application of deep reinforcement learning. *Applied Thermal Engineering* 223, 120038. <https://doi.org/10.1016/j.applthermaleng.2023.120038>
- Kimaev, G., Ricardez-Sandoval, L.A., 2017. A comparison of efficient uncertainty quantification techniques for stochastic multiscale systems. *AIChE Journal* 63, 3361–3373. <https://doi.org/10.1002/aic.15702>
- Kingma, D.P., Ba, J., 2017. Adam: A Method for Stochastic Optimization.
- Koller, R.W., Ricardez-Sandoval, L.A., Biegler, L.T., 2018. Stochastic back-off algorithm for simultaneous design, control, and scheduling of multiproduct systems under uncertainty. *AIChE Journal* 64, 2379–2389. <https://doi.org/10.1002/aic.16092>
- Kookos, I.K., Perkins, J.D., 2016. Control structure selection based on economics: Generalization of the back-off methodology. *AIChE Journal* 62, 3056–3064. <https://doi.org/10.1002/aic.15284>

- Kookos, I.K., Perkins, J.D., 2001. An Algorithm for Simultaneous Process Design and Control. *Ind. Eng. Chem. Res.* 40, 4079–4088. <https://doi.org/10.1021/ie000622t>
- Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D., 2019. Continuous control with deep reinforcement learning.
- Liñán, D.A., Bernal, D.E., Ricardez-Sandoval, L.A., Gómez, J.M., 2020. Optimal design of superstructures for placing units and streams with multiple and ordered available locations. Part I: A new mathematical framework. *Computers & Chemical Engineering* 137, 106794. <https://doi.org/10.1016/j.compchemeng.2020.106794>
- Liñán, D.A., Contreras-Zarazúa, G., Sánchez-Ramírez, E., Segovia-Hernández, J.G., Ricardez-Sandoval, L.A., 2024. A hybrid deterministic-stochastic algorithm for the optimal design of process flowsheets with ordered discrete decisions. *Computers & Chemical Engineering* 180, 108501. <https://doi.org/10.1016/j.compchemeng.2023.108501>
- Liñán, D.A., Ricardez-Sandoval, L.A., 2023. A Benders decomposition framework for the optimization of disjunctive superstructures with ordered discrete decisions. *AIChE Journal* 69, e18008. <https://doi.org/10.1002/aic.18008>
- Luyben, M.L., Floudas, C.A., 1994. Analyzing the interaction of design and control—1. A multiobjective framework and application to binary distillation synthesis. *Computers & Chemical Engineering* 18, 933–969. [https://doi.org/10.1016/0098-1354\(94\)E0013-D](https://doi.org/10.1016/0098-1354(94)E0013-D)
- Luyben, M.L., Floudas, C.A., 1994. Analyzing the interaction of design and control—1. A multiobjective framework and application to binary distillation synthesis. *Computers & Chemical Engineering* 18, 933–969. [https://doi.org/10.1016/0098-1354\(94\)E0013-D](https://doi.org/10.1016/0098-1354(94)E0013-D)
- Luyben, M.L., Floudas, C.A., 1994. Analyzing the interaction of design and control—2. reactor-separator-recycle system. *Computers & Chemical Engineering* 18, 971–993. [https://doi.org/10.1016/0098-1354\(94\)85006-2](https://doi.org/10.1016/0098-1354(94)85006-2)
- Luyben, W.L., 2017. Improving the conventional reactor/separation/recycle DME process. *Computers & Chemical Engineering* 106, 17–22. <https://doi.org/10.1016/j.compchemeng.2017.05.008>
- Ma, Y., Yang, Z., El-Khoruy, A., Zhang, N., Li, J., Zhang, B., Sun, L., 2021. Simultaneous Synthesis and Design of Reaction–Separation–Recycle Processes Using Rigorous Models. *Ind. Eng. Chem. Res.* 60, 7275–7290. <https://doi.org/10.1021/acs.iecr.1c00250>
- Malcolm, A., Polan, J., Zhang, L., Ogunnaike, B.A., Linninger, A.A., 2007. Integrating systems design and control using dynamic flexibility analysis. *AIChE Journal* 53, 2048–2061. <https://doi.org/10.1002/aic.11218>
- Mann, V., Gani, R., Venkatasubramanian, V., 2023. Intelligent Process Flowsheet Synthesis and Design using Extended SFILES Representation, in: Kokossis, A.C., Georgiadis, M.C., Pistikopoulos, E. (Eds.), *Computer Aided Chemical Engineering*. Elsevier, pp. 221–226. <https://doi.org/10.1016/B978-0-443-15274-0.50036-6>
- Mann, V., Sales-Cruz, M., Gani, R., Venkatasubramanian, V., 2024a. Hybrid Artificial Intelligence-based Process Flowsheet Synthesis and Design using Extended SFILES Representation, in: Manenti, F., Reklaitis, G.V. (Eds.), *Computer Aided Chemical Engineering*. Elsevier, pp. 1279–1284. <https://doi.org/10.1016/B978-0-443-28824-1.50214-3>
- Mann, V., Sales-Cruz, M., Gani, R., Venkatasubramanian, V., 2024b. eSFILES: Intelligent process flowsheet synthesis using process knowledge, symbolic AI, and machine learning. *Computers & Chemical Engineering* 181, 108505. <https://doi.org/10.1016/j.compchemeng.2023.108505>
- Mansouri, S.S., Huusom, J.K., Gani, R., Sales-Cruz, M., 2016a. Systematic integrated process design and control of binary element reactive distillation processes. *AIChE Journal* 62, 3137–3154. <https://doi.org/10.1002/aic.15322>
- Mansouri, S.S., Sales-Cruz, M., Huusom, J.K., Gani, R., 2016b. Systematic integrated process design and control of reactive distillation processes involving multi-elements. *Chemical Engineering Research and Design* 115, 348–364. <https://doi.org/10.1016/j.cherd.2016.07.010>

- Marquardt, W., Mönnigmann, M., 2005. Constructive nonlinear dynamics in process systems engineering. *Computers & Chemical Engineering* 29, 1265–1275. <https://doi.org/10.1016/j.compchemeng.2005.02.009>
- Medeiros, D.W., n.d. DWSIM - Open Source Process Simulator.
- Mehta, S., Ricardez-Sandoval, L.A., 2016. Integration of Design and Control of Dynamic Systems under Uncertainty: A New Back-Off Approach. *Ind. Eng. Chem. Res.* 55, 485–498. <https://doi.org/10.1021/acs.iecr.5b03522>
- Mencarelli, L., Chen, Q., Pagot, A., Grossmann, I.E., 2020. A review on superstructure optimization approaches in process system engineering. *Computers & Chemical Engineering* 136, 106808. <https://doi.org/10.1016/j.compchemeng.2020.106808>
- Mendiola-Rodriguez, T.A., Ricardez-Sandoval, L.A., 2023. Integration of design and control for renewable energy systems with an application to anaerobic digestion: A deep deterministic policy gradient framework. *Energy* 274, 127212. <https://doi.org/10.1016/j.energy.2023.127212>
- Mendiola-Rodriguez, T.A., Ricardez-Sandoval, L.A., 2022. Robust control for anaerobic digestion systems of Tequila vinasses under uncertainty: A Deep Deterministic Policy Gradient Algorithm. *Digital Chemical Engineering* 3, 100023. <https://doi.org/10.1016/j.dche.2022.100023>
- Midgley, L.I., 2020. Deep Reinforcement Learning for Process Synthesis.
- Mnih, V., Badia, A.P., Mirza, M., Graves, A., Lillicrap, T.P., Harley, T., Silver, D., Kavukcuoglu, K., 2016. Asynchronous Methods for Deep Reinforcement Learning.
- Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M., 2013. Playing Atari with Deep Reinforcement Learning.
- Mohideen, M.J., Perkins, J.D., Pistikopoulos, E.N., 1996. Optimal design of dynamic systems under uncertainty. *AIChE Journal* 42, 2251–2272. <https://doi.org/10.1002/aic.690420814>
- Mönnigmann, M., Marquardt, W., 2003. Steady-state process optimization with guaranteed robust stability and feasibility. *AIChE Journal* 49, 3110–3126. <https://doi.org/10.1002/aic.690491212>
- Mönnigmann, M., Marquardt, W., 2002. Normal Vectors on Manifolds of Critical Points for Parametric Robustness of Equilibrium Solutions of ODE Systems. *Journal of Nonlinear Science* 12, 85–112. <https://doi.org/10.1007/s00332-001-0400-1>
- Mousa, M., van de Berg, D., Kotecha, N., del Rio Chanona, E.A., Mowbray, M., 2024. An analysis of multi-agent reinforcement learning for decentralized inventory control systems. *Computers & Chemical Engineering* 188, 108783. <https://doi.org/10.1016/j.compchemeng.2024.108783>
- Mowbray, M., Zhang, D., Chanona, E.A.D.R., 2022. Distributional Reinforcement Learning for Scheduling of Chemical Production Processes.
- Muñoz, D.A., Marquardt, W., 2013. Robust control design of a class of nonlinear input- and state-constrained systems. *Annual Reviews in Control* 37, 232–245. <https://doi.org/10.1016/j.arcontrol.2013.09.003>
- Narraway, L.T., Perkins, J.D., Barton, G.W., 1991. Interaction between process design and process control: economic analysis of process dynamics. *Journal of Process Control* 1, 243–250. [https://doi.org/10.1016/0959-1524\(91\)85015-B](https://doi.org/10.1016/0959-1524(91)85015-B)
- Nian, R., Liu, J., Huang, B., 2020. A review On reinforcement learning: Introduction and applications in industrial process control. *Computers & Chemical Engineering* 139, 106886. <https://doi.org/10.1016/j.compchemeng.2020.106886>
- Obermeier, A., Vollmer, N., Windmeier, C., Esche, E., Repke, J.-U., 2021. Generation of linear-based surrogate models from non-linear functional relationships for use in scheduling formulation. *Computers & Chemical Engineering* 146, 107203. <https://doi.org/10.1016/j.compchemeng.2020.107203>

- Palma-Flores, O., Ricardez-Sandoval, L.A., 2023. Selection and refinement of finite elements for optimal design and control: A Hamiltonian function approach. *AIChE Journal* 69, e18009. <https://doi.org/10.1002/aic.18009>
- Palma-Flores, O., Ricardez-Sandoval, L.A., 2022. Integration of design and NMPC-based control for chemical processes under uncertainty: An MPCC-based framework. *Computers & Chemical Engineering* 162, 107815. <https://doi.org/10.1016/j.compchemeng.2022.107815>
- Papalexandri, K.P., Pistikopoulos, E.N., 1994. Synthesis and Retrofit Design of Operable Heat Exchanger Networks. 1. Flexibility and Structural Controllability Aspects. *Ind. Eng. Chem. Res.* 33, 1718–1737. <https://doi.org/10.1021/ie00031a012>
- Patel, K.M., 2023. A practical Reinforcement Learning implementation approach for continuous process control. *Computers & Chemical Engineering* 174, 108232. <https://doi.org/10.1016/j.compchemeng.2023.108232>
- Patilas, C.S., Kookos, I.K., 2021. Algorithmic Approach to the Simultaneous Design and Control Problem. *Ind. Eng. Chem. Res.* 60, 14271–14281. <https://doi.org/10.1021/acs.iecr.1c01855>
- Patilas, C.S., Kookos, I.K., 2020. A quadratic approximation of the back-off methodology for the control structure selection problem. *Computers & Chemical Engineering* 143, 107114. <https://doi.org/10.1016/j.compchemeng.2020.107114>
- Pedrozo, H.A., Rodriguez Reartes, S.B., Chen, Q., Diaz, M.S., Grossmann, I.E., 2020. Surrogate-model based MILP for the optimal design of ethylene production from shale gas. *Computers & Chemical Engineering* 141, 107015. <https://doi.org/10.1016/j.compchemeng.2020.107015>
- Perkins, J.D., 1989. Interactions Between Process Design and Process Control. *IFAC Proceedings Volumes* 22, 195–203. [https://doi.org/10.1016/S1474-6670\(17\)53357-1](https://doi.org/10.1016/S1474-6670(17)53357-1)
- Perkins, J.D., Walsh, S.P.K., 1996. Optimization as a tool for design/control integration. *Computers & Chemical Engineering* 20, 315–323. [https://doi.org/10.1016/0098-1354\(95\)00022-4](https://doi.org/10.1016/0098-1354(95)00022-4)
- Petrazzini, I.G.B., Antonelo, E.A., 2021. Proximal Policy Optimization with Continuous Bounded Action Space via the Beta Distribution.
- Petsagkourakis, P., Sandoval, I.O., Bradford, E., Zhang, D., del Rio-Chanona, E.A., 2020. Reinforcement learning for batch bioprocess optimization. *Computers & Chemical Engineering* 133, 106649. <https://doi.org/10.1016/j.compchemeng.2019.106649>
- Pistikopoulos, E., Diangelakis, 2015. PAROC—An integrated framework and software platform for the optimisation and advanced model-based control of process systems. *Chemical Engineering Science* 136, 115–138. <https://doi.org/10.1016/j.ces.2015.02.030>
- Pistikopoulos, E.N., Diangelakis, N.A., 2016. Towards the integration of process design, control and scheduling: Are we getting closer? *Computers & Chemical Engineering*, 12th International Symposium on Process Systems Engineering & 25th European Symposium of Computer Aided Process Engineering (PSE-2015/ESCAPE-25), 31 May - 4 June 2015, Copenhagen, Denmark 91, 85–92. <https://doi.org/10.1016/j.compchemeng.2015.11.002>
- Plathottam, S.J., Richey, B., Curry, G., Cresko, J., Illoeje, C.O., 2021. Solvent extraction process design using deep reinforcement learning. *Journal of Advanced Manufacturing and Processing* 3, e10079. <https://doi.org/10.1002/amp2.10079>
- Rafiei, M., Ricardez-Sandoval, L.A., 2020a. New frontiers, challenges, and opportunities in integration of design and control for enterprise-wide sustainability. *Computers & Chemical Engineering* 132, 106610. <https://doi.org/10.1016/j.compchemeng.2019.106610>
- Rafiei, M., Ricardez-Sandoval, L.A., 2020b. Integration of design and control for industrial-scale applications under uncertainty: a trust region approach. *Computers & Chemical Engineering* 141, 107006. <https://doi.org/10.1016/j.compchemeng.2020.107006>
- Rafiei, M., Ricardez-Sandoval, L.A., 2020c. A trust-region framework for integration of design and control. *AIChE Journal* 66, e16922. <https://doi.org/10.1002/aic.16922>

- Rafiei, M., Ricardez-Sandoval, L.A., 2018. Stochastic Back-Off Approach for Integration of Design and Control Under Uncertainty. *Ind. Eng. Chem. Res.* 57, 4351–4365. <https://doi.org/10.1021/acs.iecr.7b03935>
- Rangel-Martinez, D., Ricardez-Sandoval, L.A., 2024. A recurrent reinforcement learning strategy for optimal scheduling of partially observable job-shop and flow-shop batch chemical plants under uncertainty. *Computers & Chemical Engineering* 188, 108748. <https://doi.org/10.1016/j.compchemeng.2024.108748>
- Reynoso-Donzelli, S., Ricardez-Sandoval, L., 2024a. A Reinforcement Learning Approach with Masked Agents for Chemical Process Flowsheet Design. Presented at the 2024 AIChE Annual Meeting, AIChE.
- Reynoso-Donzelli, S., Ricardez-Sandoval, L., 2024b. A Reinforcement Learning Framework for the Simultaneous Generation, Design and Control of Chemical Process Flowsheets. Presented at the 2024 AIChE Annual Meeting, AIChE.
- Reynoso-Donzelli, S., Ricardez-Sandoval, L.A., 2024c. Enhancing process flowsheet design through masked hybrid Proximal Policy Optimization. *IFAC-PapersOnLine* 58, 646–651. <https://doi.org/10.1016/j.ifacol.2024.08.410>
- Reynoso-Donzelli, S., Ricardez-Sandoval, L.A., 2024d. A reinforcement learning approach with masked agents for chemical process flowsheet design. *AIChE Journal* n/a, 16. <https://doi.org/10.1002/aic.18584>
- Reynoso-Donzelli, S., Ricardez-Sandoval, L.A., 2024e. An integrated reinforcement learning framework for simultaneous generation, design, and control of chemical process flowsheets. *Manuscript submitted for publication*.
- Ricardez-Sandoval, L.A., 2012. Optimal design and control of dynamic systems under uncertainty: A probabilistic approach. *Computers & Chemical Engineering* 43, 91–107. <https://doi.org/10.1016/j.compchemeng.2012.03.015>
- Ricardez-Sandoval, L.A., Budman, H.M., Douglas, P.L., 2010. Simultaneous Design and Control: A New Approach and Comparisons with Existing Methodologies. *Ind. Eng. Chem. Res.* 49, 2822–2833. <https://doi.org/10.1021/ie9010707>
- Ricardez-Sandoval, L.A., Budman, H.M., Douglas, P.L., 2009a. Integration of design and control for chemical processes: A review of the literature and some recent results. *Annual Reviews in Control* 33, 158–171. <https://doi.org/10.1016/j.arcontrol.2009.06.001>
- Ricardez-Sandoval, L.A., Budman, H.M., Douglas, P.L., 2009b. Application of Robust Control Tools to the Simultaneous Design and Control of Dynamic Systems. *Ind. Eng. Chem. Res.* 48, 801–813. <https://doi.org/10.1021/ie800378y>
- Ricardez-Sandoval, L.A., Budman, H.M., Douglas, P.L., 2009c. Simultaneous design and control of chemical processes with application to the Tennessee Eastman process. *Journal of Process Control* 19, 1377–1391. <https://doi.org/10.1016/j.jprocont.2009.04.009>
- Ricardez-Sandoval, L.A., Budman, H.M., Douglas, P.L., 2008. Simultaneous design and control of processes under uncertainty: A robust modelling approach. *Journal of Process Control* 18, 735–752. <https://doi.org/10.1016/j.jprocont.2007.11.006>
- Ricardez-Sandoval, L.A., Douglas, P.L., Budman, H.M., 2011. A methodology for the simultaneous design and control of large-scale systems under process parameter uncertainty. *Computers & Chemical Engineering* 35, 307–318. <https://doi.org/10.1016/j.compchemeng.2010.05.010>
- Sachio, S., Mowbray, M., Papathanasiou, M.M., del Rio-Chanona, E.A., Petsagkourakis, P., 2022. Integrating process design and control using reinforcement learning. *Chemical Engineering Research and Design* 183, 160–169. <https://doi.org/10.1016/j.cherd.2021.10.032>
- Sakizlis, V., Perkins, J.D., Pistikopoulos, E.N., 2004. Recent advances in optimization-based simultaneous process and control design. *Computers & Chemical Engineering* 28, 2069–2086. <https://doi.org/10.1016/j.compchemeng.2004.03.018>

- Sánchez-Sánchez, K., Ricardez-Sandoval, L., 2013. Simultaneous process synthesis and control design under uncertainty: A worst-case performance approach. *AIChE Journal* 59, 2497–2514. <https://doi.org/10.1002/aic.14040>
- Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O., 2017. Proximal Policy Optimization Algorithms.
- Schulze Balhorn, L., Caballero, M., Schweidtmann, A., 2024. Toward autocorrection of chemical process flowsheets using large language models. pp. 3109–3114. <https://doi.org/10.1016/B978-0-443-28824-1.50519-6>
- Schweidtmann, A.M., Bongartz, D., Huster, W.R., Mitsos, A., 2019. Deterministic Global Process Optimization: Flash Calculations via Artificial Neural Networks, in: Kiss, A.A., Zondervan, E., Lakerveld, R., Özkan, L. (Eds.), *Computer Aided Chemical Engineering*, 29 European Symposium on Computer Aided Process Engineering. Elsevier, pp. 937–942. <https://doi.org/10.1016/B978-0-12-818634-3.50157-0>
- Schweidtmann, A.M., Mitsos, A., 2019. Deterministic Global Optimization with Artificial Neural Networks Embedded. *J Optim Theory Appl* 180, 925–948. <https://doi.org/10.1007/s10957-018-1396-0>
- Schweiger, C.A., Floudas, C.A., 1998. Interaction of Design and Control: Optimization with Dynamic Models, in: Hager, W.H., Pardalos, P.M. (Eds.), *Optimal Control: Theory, Algorithms, and Applications*, Applied Optimization. Springer US, Boston, MA, pp. 388–435. https://doi.org/10.1007/978-1-4757-6095-8_19
- Seaborg, D.E., Edgar, T.F., Mellichamp, D.A., 2004. *Process Dynamics and Control*, 2nd ed. Wiley.
- Seferlis, P., Georgiadis, M.C., 2004. Introduction: The integration of process design control – Summary future directions, in: Seferlis, P., Georgiadis, M.C. (Eds.), *Computer Aided Chemical Engineering, The Integration of Process Design and Control*. Elsevier, pp. 1–9. [https://doi.org/10.1016/S1570-7946\(04\)80052-X](https://doi.org/10.1016/S1570-7946(04)80052-X)
- Seidenberg, J.R., Khan, A.A., Lapkin, A.A., 2023. Boosting autonomous process design and intensification with formalized domain knowledge. *Computers & Chemical Engineering* 169, 108097. <https://doi.org/10.1016/j.compchemeng.2022.108097>
- Seider, W.D., 2004. *Product and process design principles : synthesis, analysis, and evaluation*. Second edition. New York : Wiley, [2004] ©2004.
- Shokry, A., Baraldi, P., Zio, E., Espuña, A., 2020. Dynamic Surrogate Modeling for Multistep-ahead Prediction of Multivariate Nonlinear Chemical Processes. *Ind. Eng. Chem. Res.* 59, 15634–15655. <https://doi.org/10.1021/acs.iecr.0c00729>
- Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., Hassabis, D., 2018. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science* 362, 1140–1144. <https://doi.org/10.1126/science.aar6404>
- Smith, R., 2005. *Chemical process design and integration*. Wiley.
- Song, C., Kawai, R., 2023. Monte Carlo and variance reduction methods for structural reliability analysis: A comprehensive review. *Probabilistic Engineering Mechanics* 73, 103479. <https://doi.org/10.1016/j.probengmech.2023.103479>
- Stinchfield, G., Morgan, J.C., Naik, S., Biegler, L.T., Eslick, J.C., Jacobson, C., Miller, D.C., Sirola, J.D., Zamarripa, M., Zhang, C., Zhang, Q., Laird, C.D., 2024. A mixed integer linear programming approach for the design of chemical process families. *Computers & Chemical Engineering* 183, 108620. <https://doi.org/10.1016/j.compchemeng.2024.108620>
- Stops, L., Leenhouts, R., Gao, Q., Schweidtmann, A.M., 2023. Flowsheet generation through hierarchical reinforcement learning and graph neural networks. *AIChE Journal* 69, e17938. <https://doi.org/10.1002/aic.17938>

- Sutton, R.S., Barto, A., 2014. Reinforcement learning: an introduction, Nachdruck. ed, Adaptive computation and machine learning. The MIT Press, Cambridge, Massachusetts.
- Swartz, C.L.E., 2004. Chapter B3 - The use of controller parametrization in the integration of design control, in: Seferlis, P., Georgiadis, M.C. (Eds.), Computer Aided Chemical Engineering. Elsevier, pp. 239–263. [https://doi.org/10.1016/S1570-7946\(04\)80062-2](https://doi.org/10.1016/S1570-7946(04)80062-2)
- The MathWorks Inc., 2022. Statistics and machine learning toolbox. MATLAB version: 9.13.0 (R2022b).
- Tian, Y., Pappas, I., Burnak, B., Katz, J., Pistikopoulos, E.N., 2021. Simultaneous design & control of a reactive distillation system – A parametric optimization & control approach. Chemical Engineering Science 230, 116232. <https://doi.org/10.1016/j.ces.2020.116232>
- Tian, Y., Pistikopoulos, E.N., 2019. Synthesis of Operable Process Intensification Systems—Steady-State Design with Safety and Operability Considerations. Ind. Eng. Chem. Res. 58, 6049–6068. <https://doi.org/10.1021/acs.iecr.8b04389>
- Trainor, M., Giannakeas, V., Kiss, C., Ricardez-Sandoval, L.A., 2013. Optimal process and control design under uncertainty: A methodology with robust feasibility and stability analyses. Chemical Engineering Science 104, 1065–1080. <https://doi.org/10.1016/j.ces.2013.10.017>
- Trespalcios, F., Grossmann, I.E., 2014. Review of Mixed-Integer Nonlinear and Generalized Disjunctive Programming Methods. Chemie Ingenieur Technik 86, 991–1012. <https://doi.org/10.1002/cite.201400037>
- van Baten, J., Baur, R., Kooijman, H., 2023. COCO (CAPE-OPEN to CAPE-OPEN).
- van Hasselt, H., Guez, A., Silver, D., 2015. Deep Reinforcement Learning with Double Q-learning.
- van Kalmthout, S.C.P.A., Midgley, L.I., Franke, M.B., 2022. Synthesis of separation processes with reinforcement learning.
- Villegas, J.I., Kumar, N., Heikkilä, T., Lehto, V.-P., Salmi, T., Murzin, D.Yu., 2006. Isomerization of n-butane to isobutane over Pt-modified Beta and ZSM-5 zeolite catalysts: Catalyst deactivation and regeneration. Chemical Engineering Journal 120, 83–89. <https://doi.org/10.1016/j.cej.2006.03.011>
- Vogel, G., Schulze Balhorn, L., Schweidtmann, A.M., 2023. Learning from flowsheets: A generative transformer model for autocompletion of flowsheets. Computers & Chemical Engineering 171, 108162. <https://doi.org/10.1016/j.compchemeng.2023.108162>
- Wächter, A., Biegler, L.T., 2006. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math. Program. 106, 25–57. <https://doi.org/10.1007/s10107-004-0559-y>
- Wang, H., Ricardez-Sandoval, L.A., 2024. A Deep Reinforcement Learning-Based PID Tuning Strategy for Nonlinear MIMO Systems with Time-varying Uncertainty. IFAC-PapersOnLine 58, 887–892. <https://doi.org/10.1016/j.ifacol.2024.08.449>
- Waschneck, B., Reichstaller, A., Belzner, L., Altenmüller, T., Bauernhansl, T., Knapp, A., Kyek, A., 2018. Optimization of global production scheduling with deep reinforcement learning. Procedia CIRP 72, 1264–1269. <https://doi.org/10.1016/j.procir.2018.03.212>
- Wiewiora, E., 2010. Reward Shaping, in: Sammut, C., Webb, G.I. (Eds.), Encyclopedia of Machine Learning. Springer US, Boston, MA, pp. 863–865. https://doi.org/10.1007/978-0-387-30164-8_731
- Williams, R.J., 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Mach Learn 8, 229–256. <https://doi.org/10.1007/BF00992696>
- Wirth, B., Gerhard, J., Marquardt, W., 2011. Robust Optimisation with Normal Vectors on Critical Manifolds of Disturbance-Induced Stability Loss. Journal of Nonlinear Science 21, 57–92. <https://doi.org/10.1007/s00332-010-9076-8>
- Yu, C., Velu, A., Vinitzky, E., Gao, J., Wang, Y., Bayen, A., WU, Y., 2022. The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games, in: Koyejo, S., Mohamed, S.,

- Agarwal, A., Belgrave, D., Cho, K., Oh, A. (Eds.), *Advances in Neural Information Processing Systems*. Curran Associates, Inc., pp. 24611–24624.
- Yu, E.Y., 2020. Coding PPO from Scratch with PyTorch (Part 1/4) [WWW Document]. Medium. URL <https://medium.com/analytics-vidhya/coding-ppo-from-scratch-with-pytorch-part-1-4-613dfc1b14c8> (accessed 7.4.24).
- Yuan, Z., Chen, B., Sin, G., Gani, R., 2012. State-of-the-art and progress in the optimization-based simultaneous design and control for chemical processes. *AIChE Journal* 58, 1640–1659. <https://doi.org/10.1002/aic.13786>

Letters of Copyright Permissions

JOHN WILEY AND SONS LICENSE

TERMS AND CONDITIONS

Sep 16, 2024

This Agreement between Simone Reynoso-Donzelli ("You") and John Wiley and Sons ("John Wiley and Sons") consists of your license details and the terms and conditions provided by John Wiley and Sons and Copyright Clearance Center.

License Number	5870940545567
License date	Sep 16, 2024
Licensed Content Publisher	John Wiley and Sons
Licensed Content Publication	AIChE Journal
Licensed Content Title	A reinforcement learning approach with masked agents for chemical process flowsheet design
Licensed Content Author	Simone Reynoso-Donzelli, Luis Alberto Ricardez-Sandoval
Licensed Content Date	Aug 30, 2024
Licensed Content Volume	0
Licensed Content Issue	0
Licensed Content Pages	16
Type of use	Dissertation/Thesis
Requestor type	Author of this Wiley article
Format	Electronic

Portion	Full article
Will you be translating?	No
Title of new work	A Reinforcement Learning methodology for chemical process flowsheet generation, design and control in steady and transient states
Institution name	University of Waterloo
Expected presentation date	Dec 2024
The Requesting Person / Organization to Appear on the License	Simone Reynoso-Donzelli
	University of Waterloo 200 University Avenue West
Requestor Location	Waterloo, ON N2L 3G1 Canada Attn: University of Waterloo
Publisher Tax ID	EU826007151
Total	0.00 CAD
Terms and Conditions	

TERMS AND CONDITIONS

This copyrighted material is owned by or exclusively licensed to John Wiley & Sons, Inc. or one of its group companies (each a "Wiley Company") or handled on behalf of a society with which a Wiley Company has exclusive publishing rights in relation to a particular work (collectively "WILEY"). By clicking "accept" in connection with completing this licensing transaction, you agree that the following terms and conditions apply to this transaction (along with the billing and payment terms and conditions established by the Copyright Clearance Center Inc., ("CCC's Billing

and Payment terms and conditions"), at the time that you opened your RightsLink account (these are available at any time at <http://myaccount.copyright.com>).

Terms and Conditions

- The materials you have requested permission to reproduce or reuse (the "Wiley Materials") are protected by copyright.
- You are hereby granted a personal, non-exclusive, non-sub licensable (on a stand-alone basis), non-transferable, worldwide, limited license to reproduce the Wiley Materials for the purpose specified in the licensing process. This license, **and any CONTENT (PDF or image file) purchased as part of your order**, is for a one-time use only and limited to any maximum distribution number specified in the license. The first instance of republication or reuse granted by this license must be completed within two years of the date of the grant of this license (although copies prepared before the end date may be distributed thereafter). The Wiley Materials shall not be used in any other manner or for any other purpose, beyond what is granted in the license. Permission is granted subject to an appropriate acknowledgement given to the author, title of the material/book/journal and the publisher. You shall also duplicate the copyright notice that appears in the Wiley publication in your use of the Wiley Material. Permission is also granted on the understanding that nowhere in the text is a previously published source acknowledged for all or part of this Wiley Material. Any third party content is expressly excluded from this permission.
- With respect to the Wiley Materials, all rights are reserved. Except as expressly granted by the terms of the license, no part of the Wiley Materials may be copied, modified, adapted (except for minor reformatting required by the new Publication), translated, reproduced, transferred or distributed, in any form or by any means, and no derivative works may be made based on the Wiley Materials without the prior permission of the respective copyright owner. **For STM Signatory Publishers clearing permission under the terms of the [STM Permissions Guidelines](#) only, the terms of the license are extended to include subsequent editions and for editions in other languages, provided such editions are for the work as a whole in situ and does not involve the separate exploitation of the permitted figures or extracts**, You may not alter, remove or suppress in any manner any copyright, trademark or other notices displayed by the Wiley Materials.

You may not license, rent, sell, loan, lease, pledge, offer as security, transfer or assign the Wiley Materials on a stand-alone basis, or any of the rights granted to you hereunder to any other person.

- The Wiley Materials and all of the intellectual property rights therein shall at all times remain the exclusive property of John Wiley & Sons Inc, the Wiley Companies, or their respective licensors, and your interest therein is only that of having possession of and the right to reproduce the Wiley Materials pursuant to Section 2 herein during the continuance of this Agreement. You agree that you own no right, title or interest in or to the Wiley Materials or any of the intellectual property rights therein. You shall have no rights hereunder other than the license as provided for above in Section 2. No right, license or interest to any trademark, trade name, service mark or other branding ("Marks") of WILEY or its licensors is granted hereunder, and you agree that you shall not assert any such right, license or interest with respect thereto
- NEITHER WILEY NOR ITS LICENSORS MAKES ANY WARRANTY OR REPRESENTATION OF ANY KIND TO YOU OR ANY THIRD PARTY, EXPRESS, IMPLIED OR STATUTORY, WITH RESPECT TO THE MATERIALS OR THE ACCURACY OF ANY INFORMATION CONTAINED IN THE MATERIALS, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTY OF MERCHANTABILITY, ACCURACY, SATISFACTORY QUALITY, FITNESS FOR A PARTICULAR PURPOSE, USABILITY, INTEGRATION OR NON-INFRINGEMENT AND ALL SUCH WARRANTIES ARE HEREBY EXCLUDED BY WILEY AND ITS LICENSORS AND WAIVED BY YOU.
- WILEY shall have the right to terminate this Agreement immediately upon breach of this Agreement by you.
- You shall indemnify, defend and hold harmless WILEY, its Licensors and their respective directors, officers, agents and employees, from and against any actual or threatened claims, demands, causes of action or proceedings arising from any breach of this Agreement by you.
- IN NO EVENT SHALL WILEY OR ITS LICENSORS BE LIABLE TO YOU OR ANY OTHER PARTY OR ANY OTHER PERSON OR ENTITY FOR ANY SPECIAL, CONSEQUENTIAL, INCIDENTAL, INDIRECT, EXEMPLARY OR PUNITIVE

DAMAGES, HOWEVER CAUSED, ARISING OUT OF OR IN CONNECTION WITH THE DOWNLOADING, PROVISIONING, VIEWING OR USE OF THE MATERIALS REGARDLESS OF THE FORM OF ACTION, WHETHER FOR BREACH OF CONTRACT, BREACH OF WARRANTY, TORT, NEGLIGENCE, INFRINGEMENT OR OTHERWISE (INCLUDING, WITHOUT LIMITATION, DAMAGES BASED ON LOSS OF PROFITS, DATA, FILES, USE, BUSINESS OPPORTUNITY OR CLAIMS OF THIRD PARTIES), AND WHETHER OR NOT THE PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION SHALL APPLY NOTWITHSTANDING ANY FAILURE OF ESSENTIAL PURPOSE OF ANY LIMITED REMEDY PROVIDED HEREIN.

- Should any provision of this Agreement be held by a court of competent jurisdiction to be illegal, invalid, or unenforceable, that provision shall be deemed amended to achieve as nearly as possible the same economic effect as the original provision, and the legality, validity and enforceability of the remaining provisions of this Agreement shall not be affected or impaired thereby.
- The failure of either party to enforce any term or condition of this Agreement shall not constitute a waiver of either party's right to enforce each and every term and condition of this Agreement. No breach under this agreement shall be deemed waived or excused by either party unless such waiver or consent is in writing signed by the party granting such waiver or consent. The waiver by or consent of a party to a breach of any provision of this Agreement shall not operate or be construed as a waiver of or consent to any other or subsequent breach by such other party.
- This Agreement may not be assigned (including by operation of law or otherwise) by you without WILEY's prior written consent.
- Any fee required for this permission shall be non-refundable after thirty (30) days from receipt by the CCC.
- These terms and conditions together with CCC's Billing and Payment terms and conditions (which are incorporated herein) form the entire agreement between you and WILEY concerning this licensing transaction and (in the absence of fraud) supersedes all prior agreements and representations of the parties, oral or written. This Agreement may not be amended except in writing signed by both parties. This Agreement shall be binding upon

and inure to the benefit of the parties' successors, legal representatives, and authorized assigns.

- In the event of any conflict between your obligations established by these terms and conditions and those established by CCC's Billing and Payment terms and conditions, these terms and conditions shall prevail.
- WILEY expressly reserves all rights not specifically granted in the combination of (i) the license details provided by you and accepted in the course of this licensing transaction, (ii) these terms and conditions and (iii) CCC's Billing and Payment terms and conditions.
- This Agreement will be void if the Type of Use, Format, Circulation, or Requestor Type was misrepresented during the licensing process.
- This Agreement shall be governed by and construed in accordance with the laws of the State of New York, USA, without regards to such state's conflict of law rules. Any legal action, suit or proceeding arising out of or relating to these Terms and Conditions or the breach thereof shall be instituted in a court of competent jurisdiction in New York County in the State of New York in the United States of America and each party hereby consents and submits to the personal jurisdiction of such court, waives any objection to venue in such court and consents to service of process by registered or certified mail, return receipt requested, at the last known address of such party.

WILEY OPEN ACCESS TERMS AND CONDITIONS

Wiley Publishes Open Access Articles in fully Open Access Journals and in Subscription journals offering Online Open. Although most of the fully Open Access journals publish open access articles under the terms of the Creative Commons Attribution (CC BY) License only, the subscription journals and a few of the Open Access Journals offer a choice of Creative Commons Licenses. The license type is clearly identified on the article.

The Creative Commons Attribution License

The [Creative Commons Attribution License \(CC-BY\)](#) allows users to copy, distribute and transmit an article, adapt the article and make commercial use of the article. The CC-BY license permits commercial and non-

Creative Commons Attribution Non-Commercial License

The [Creative Commons Attribution Non-Commercial \(CC-BY-NC\)License](#) permits use, distribution and reproduction in any medium, provided the original work is properly cited and is not used for commercial purposes.(see below)

Creative Commons Attribution-Non-Commercial-NoDerivs License

The [Creative Commons Attribution Non-Commercial-NoDerivs License](#) (CC-BY-NC-ND) permits use, distribution and reproduction in any medium, provided the original work is properly cited, is not used for commercial purposes and no modifications or adaptations are made. (see below)

Use by commercial "for-profit" organizations

Use of Wiley Open Access articles for commercial, promotional, or marketing purposes requires further explicit permission from Wiley and will be subject to a fee.

Further details can be found on Wiley Online

Library <http://olabout.wiley.com/WileyCDA/Section/id-410895.html>

Other Terms and Conditions:

v1.10 Last updated September 2015

Appendix A Questions? customercare@copyright.com.

Appendix A

Chapter 3 – Case study 1

In this appendix, a detailed description of the UOs presented in case study 1 is provided. Note that when defining the equations of the UOs, input flows are considered. These flows either come from previous operations or the main inlet flow. The subscript for these incoming flows is always *in*, regardless of their origin.

1. Mixer (M)

The purpose of the mixer is to mix material flows without considering any enthalpy, temperature, or pressure changes during the process. The mixer has two possible inlet streams and an outlet stream. If only one inlet stream is accepted, the UO operates as a bypass block; so, the total molar flow (F), the concentration of reactant A (C_A) and the temperature (T) are kept:

$$F_{in} = F \quad (A1)$$

$$C_{A_{in}} = C_A \quad (A2)$$

$$T_{in} = T \quad (A3)$$

If a recycle current (F_{rec}) is present in the mixer, the equations change as following:

$$F_{in} + F_{rec} = F \quad (A4)$$

$$\frac{F_{in} * C_{A_{in}} + F_{rec} * C_{A_{rec}}}{F} = C_A \quad (A5)$$

$$\frac{F_{in} * T_{in} + F_{rec} * T_{rec}}{F} = T \quad (A6)$$

The mixer is one of the two essential components for performing recycling thus leading into a disjunctive decision, i.e., if this UO is not present in the flowsheet, material recirculation cannot be carried out. Associating a cost with a mixer is complicated, as it is not the same as comparing this operation with a reactor or a distillation column. However, to indicate to the agent that placing a mixer in the flowsheet is important, a cost was associated with it. For this case:

$$cost_M = 0.1 \quad (A6)$$

2. CSTR

The reactor type included in the set of available UOs is a CSTR. In this setup, R is considered non-isothermal with a constant flow of coolant. The reaction that takes place inside the reactor is the direct conversion of A into B ; $A \rightarrow B$. Additionally, it is assumed that this unit operates at steady state, featuring one inlet and one outlet stream. The CSTR model parameters as well as the reaction kinetics were derived from Schweiger et al. (Schweiger and Floudas, 1998)

Table A.1: Parameters for CSTR

Description	Parameter	Value
Heat of reaction	ΔH_R	-3000.0 BTU/lbmol
Heat transfer coefficient	U	300.0 BTU/(hr ft ² °R)
Energy of activation	E/R	15075.0°R
Kinetic rate constant	k_0	$4.08 \times 10^{10} \text{ hr}^{-1}$
Liquid density	ρ	50.0 lb/ft ³
Liquid heat capacity	C_p	0.75 BTU/(lb °R)
Coolant density	ρ_{cool}	62.3 lb/ft ³
Coolant heat capacity	C_{cool}	1.0 BTU/(lb °R)
Feed flow rate	F_0	100 ft ³ /hr
Flow rate coolant	F_{cool}	30 ft ³ /hr
Feed concentration of A	C_{A0}	1 lbmol/ft ³
Feed temperature	T_0	600°R
Feed coolant temperature	T_{cool_0}	530°R

The associated design variables for this UO are the diameter (D_{CSTR}) and the height (H_{CSTR}). Both design variables are bounded within the range of 5.5 to 7.75 ft. and were derived from the authors solution. When solving the model, the values of the concentration of reactant A (C_A), the temperature (T) and the coolant temperature (T_{cool}) at the outlet were obtained. The molar flow is considered constant in this UO. The mathematical model is as follows:

$$k = k_0 e^{-\frac{E}{R^* T}} \quad (A7)$$

$$V_{CSTR} = \frac{\pi}{4} * D_{CSTR}^2 * H_{CSTR} \quad (A8)$$

$$A_{CSTR} = \pi * D_{CSTR} * H_{CSTR} \quad (A9)$$

$$V_{cool} = \frac{A_{CSTR}}{3} \quad (A10)$$

$$\frac{F_0}{V_{CSTR}} * (C_{A_0} - C_A) = k * C_A \quad (A11)$$

$$\frac{F_0}{V_{CSTR}} * (T_0 - T) + \frac{(-\Delta H_R * k * C_A)}{C_p * \rho} = \frac{U * A_{CSTR}}{V * C_p * \rho} * (T - T_{cool}) \quad (A12)$$

$$\frac{F_{cool}}{V_{cool}} * (T_{cool_0} - T_{cool}) = -\frac{U * A_{CSTR}}{V_{cool} * C_{cool} * \rho_{cool}} * (T - T_{cool}) \quad (A13)$$

The cost function for the CSTR was derived from Douglas (Douglas, n.d.):

$$cost_{CSTR} = \frac{\left(\frac{D_{CSTR}}{7.75}\right)^{1.05} + \left(\frac{H_{CSTR}}{7.75}\right)^{0.82}}{2} \quad (A14)$$

To avoid a greater effect of one action over another, the cost function for this operation is normalized.

3. Flash tank (FT)

This operation seeks to separate the mixture of A and B, assuming a relative volatility (α) of 4.5. In this operation, a constant operating temperature is assumed, so the temperature at the inlet to the FT is the same as the outlet temperature in both the liquid and vapor phases. This value of α was chosen to facilitate the separation process, thereby enabling the use of a simple UO like FT instead of a distillation column. The design variable for this UO is the vapor/feed fraction of the flash tank (vf), bounded between 0.1 and 0.5. These values of vf were chosen because at a higher value, the model yielded incorrect results and the effect of the FT was counterproductive rather than offering an alternative UO to the agent.

This unit operation involves one input and two outputs, namely the vapor and the liquid streams. In the FT model, the complete vapor stream is recycled back to the mixer, assuming that the stream undergoes condensation before reaching the mixer to align with the composition of the input flow. The liquid stream functions as the material stream for the subsequent development of the process flowsheet. The FT model returns the values for the liquid (L_{FT}) and vapor (F_{rec}) stream, as well as the

concentrations of A associated to both streams, C_{A_L} and $C_{A_{rec}}$ respectively. The model for FT is the following:

$$L_{FT} = vf * F_{in} \quad (A15)$$

$$F_{rec} = F_{in} - L_{FT} \quad (A16)$$

$$C_{A_{rec}} = \frac{\alpha * C_{A_L}}{1 + C_{A_L} * (\alpha - 1)} \quad (A17)$$

$$C_{A_{rec}} = \frac{vf}{vf - 1} * C_{A_L} + \frac{C_{A_{in}}}{1 - vf} \quad (A18)$$

Deriving the cost function for the FT was challenging because there was no possibility of calculating the diameter and height of the tank, as there were no thermodynamic data available for the problem, given the hypothetical reactants. Therefore, the cost function was as follows:

$$cost_{FT} = 0.5 * (1 + vf) \quad (A19)$$

4. Convergence test

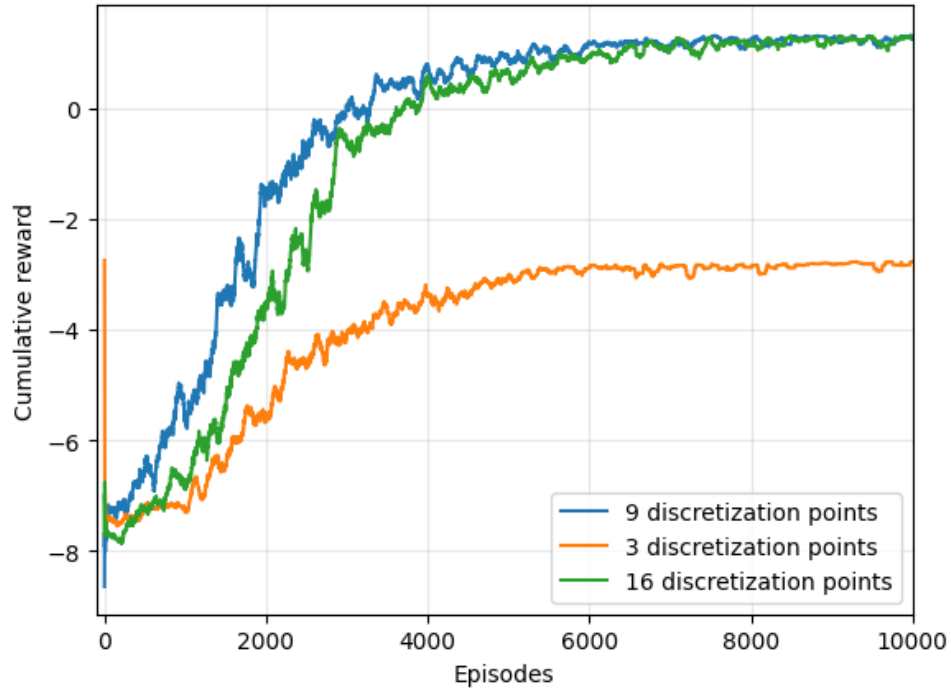


Figure A.1: Learning curves of mPPO for different number of discretization points

Figure A.1 displays the learning curves of mPPO for case study 1, where the design variables of the CSTR are discretized by a different number of discretization points. As shown in the figure above, the agent fails to find an appealing flowsheet with very few discretization points. Despite achieving the design specification (i.e., a conversion of 97.5% or higher), the flowsheet proposed by this agent only requires reactors in series; hence, the low cumulative reward obtained. As the number of discretization points increases, the problem converges to the same cumulative reward. In this case, there is not much difference between 9 and 16 discretization points, since in both cases the agent converges to the same cumulative reward of around 1.3. This behavior is key, as for problems with a large number of continuous variables, finding the smallest number of discretization points that also allow reaching the maximum cumulative reward significantly simplifies the problem.

Appendix B

Chapter 4 – Case study 1

This appendix provides a more in-depth examination of various components of case study 1. Among these components is a thorough description of the mathematical model used for the CSTR, as well as a more detailed analysis of the surrogate models employed, the whole mathematical formulation of the optimization problem and some supporting images for the various scenarios.

1. Mechanistic model

The model of the CSTR was taken from (Seaborg et al., 2004) and considered a liquid-phase, irreversible, first-order chemical reaction. The chemical species involved in the reaction are the reactant A and the product B. The reaction can be written as follows: $A \rightarrow B$. It is assumed that the rate of reaction is first order with respect to the concentration of reactant A (C_A):

$$r_A = kC_A \quad (\text{B1})$$

$$k = k_0 e^{-\frac{E_A}{R_{cte}T}} \quad (\text{B2})$$

where k_0 is the frequency factor, E_A is the activation energy, and R_{cte} is the gas constant. Note that the kinetic scheme remains consistent across all reactors in the process flow. A cooling coil is used to maintain the desired operating temperature of the reaction mixture by removing the heat generated during the exothermic reaction. The initial development of the CSTR model is based on the following assumptions:

- a) The CSTR is perfectly mixed
- b) The mass densities of the feed and product streams are equal and constant. They are denoted by ρ
- c) The liquid volume V in the reactor is kept constant by an overflow line. So the inlet flow equals the outlet flow ($q_{in} = q$)
- d) The liquid volume of the tank and the total volume of the tank are considered to be the same.

Note that when defining the equations of the CSTR, input flows are considered. These flows may originate from previous reactors or the main inlet flow. The subscript for these incoming flows and their properties is denoted as *in*, regardless of their origin. The mass balance for reactant A, in terms of molar units, derived from these assumptions is:

$$V_R \frac{dC_A}{dt} = q(C_{A_{in}} - C_A) - V_R k C_A \quad (\text{B3})$$

where V_R is the volume and q the volumetric flowrate. For the energy balance the following assumptions were made:

- e) The thermal capacitances of the coolant and the cooling coil wall are negligible compared to the thermal capacitance of the liquid in the tank.
- f) The coolant is at a uniform temperature, T_C^R . (In other words, the increase in coolant temperature as it passes through the coil is neglected.)
- g) The rate of heat transfer from the reactor contents to the coolant is expressed as

$$Q_R = UA_R(T_C^R - T) \quad (\text{B4})$$

where U is the overall heat transfer coefficient, and A_R is the heat transfer area. Both of these model parameters are assumed to be constant.

- h) The enthalpy change associated with the mixing of the feed and the liquid in the tank is negligible compared to the enthalpy change for the chemical reaction. In other words, the heat of mixing is insignificant compared to the heat of reaction.
- i) Shaft work and heat losses to the ambient can be neglected.

The energy balance derived from these assumptions is as follows:

$$V_R \rho C_P \frac{dT}{dt} = \rho q (T_{in} - T) + (-\Delta H_R) V_R k C_A + UA(T_C^R - T) \quad (\text{B5})$$

where ΔH_R is the heat of reaction per mole of A that is reacted.

A suitable real-world analogy for this setup is the isomerization of n-butane to *iso*-butane. In this reaction, *n*-butane (reactant A) is converted to its structural isomer, *iso*-butane (product B), through a catalytic process that is both irreversible and typically follows first-order kinetics with respect to n-butane concentration. The operational requirements of the isomerization process align closely with the design conditions of the CSTR model. For example, this reaction, like the CSTR model, is conducted in a liquid phase and often within a controlled-temperature environment, making it a suitable analogy

for scenarios involving exothermic reactions that require heat removal via cooling. The choice of n-butane and isobutane is particularly relevant, as these isomers have similar physical properties, including density, which simplifies assumptions in the CSTR model and makes it adaptable for both theoretical and practical optimization of process parameters in isomerization reactions (Villegas et al., 2006).

In this problem, it is assumed that there is a disturbance with a 5% decrease in the inlet concentration ($C_{A_{in}}$) at time $t = 2$. To mitigate the effects of this disturbance, a control system is implemented on the equipment. This system aims to control the outlet concentration of the reactor by manipulating its jacket temperature. The controller integrated into the reactor is a PI controller represented by the following equations:

$$err_R = C_A - C_A^{sp} \quad (B6)$$

$$\frac{dI_R}{dt} = err_{C_A} \quad (B7)$$

$$T_C^R = T_{C_{nom}}^R + K_C^R \left(err_{C_A} + \frac{err_R}{\tau_I^R} \right) \quad (B8)$$

where C_A^{sp} is the desired concentration set-point of reactant A, $T_{C_{nom}}^R$ is the nominal cooling temperature (determined beforehand), K_C^R is the proportional gain of the PI controller and τ_I^R is the integrated time of the PI controller.

The parameters used in this case study were the following (Table B.1):

Table B.1 Parameters values

Parameter	Value
q_{in} [L/min]	100
C_{A_0} [mol/L] (first inlet stream)	1
T_0 [K] (first inlet stream)	350
ρ [g/L]	1000
$T_{C_{nom}}^R$ [K]	300
C_p [J/gK]	0.239
ΔH_R [J/mol]	-5.0e4
k_0 [1/min]	7.2e10
UA_R [J/minK]	5.0e4
E_A/R_{cte} [K]	8750

The bounds for the feed conditions were established based on the operational goals and constraints of the system. The feed streams to each unit operation can originate from various points in the flowsheet; hence when defining these bounds, it is crucial to consider the potential maximum and minimum values that the feed conditions can take, such as temperature, pressure, and mole fraction. These values were set to ensure that the system operates within feasible and realistic parameters. The relative width of the bounds is determined by the system's flexibility and objectives, with narrower ranges typically applied to more constrained variables and broader ranges used for those with greater variability in operational conditions. As for the design variables and their bounds, they were as follows (Table B.2):

Table B.2: Design variables' bounds

Design variable	Bounds
V_R [L]	50 – 200
C_A^{sp} [mol/L]	0.045 – 1.0
K_C^R [K * L/mol]	20 – 150
τ_I^R [1/min]	0.01 – 0.5

2. Surrogate models

The dataset used to train the surrogate models was generated using Latin Hypercube Sampling (LHS) and comprised 80,000 combinations of the inlet variables of the reactor's model, i.e., inlet concentration and temperature, set-point concentration, volume, and the controllers' proportional gain and time integral. During the generation of this matrix, it was ensured that the input concentration was always greater than the set-point concentration. This precaution was taken to avoid any errors during the simulations. Once the matrix was constructed, the various combinations were evaluated using the CSTR model presented in the previous section. The outputs of the model included a vector of concentrations of reactant A ($\mathbf{C}_A$) and vector of temperatures ($\mathbf{T}$) at the reactor's outlet. From $\mathbf{C}_A$, the Integral Squared Error (ISE) was calculated (Equation B9), while with $\mathbf{T}$, a metric ($temp_{cons}$) was calculated to determine the extent of temperature constraint violations (Equation B10).

$$ISE_R = \int_{t_0}^{t_f} (C_A - C_A^{sp})^2 dt \quad (B9)$$

$$\boldsymbol{\vartheta} = \begin{cases} T - 400, & T \geq 400 \\ 0, & T < 400 \end{cases} \quad (B10)$$

$$cons_T = \int_{t_0}^{t_f} \boldsymbol{\vartheta}^2 dt$$

Note that $cons_T$ represents the variable $cons$ in Algorithm 4.1 presented in the manuscript. The previous logical tests (i.e., the distance from the stabilization point and the set-point, and the slope of the last five values) are also calculated in the mechanistic model.

As mentioned in the manuscript, surrogate models only determine a single value for the variables, not the entire dynamic profile. Therefore, the endpoints of the vectors $\mathbf{C}_A$ and $\mathbf{T}$ were chosen to be regressed by the surrogate models, C_A and T respectively. Prior to the surrogate model training, a correlation matrix was constructed to assess the impact of the inlet variables on the regressed variables (Figure B.1).

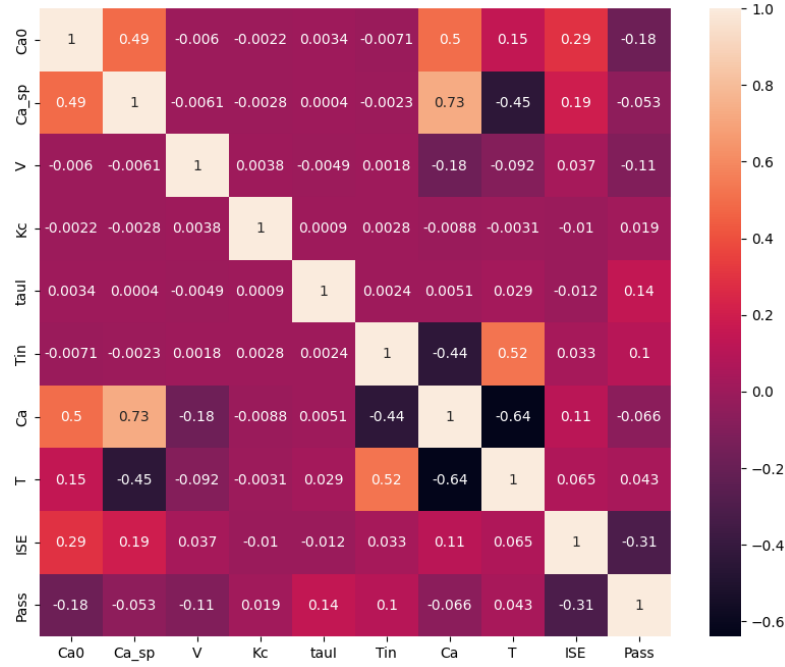


Figure B.1: Correlations matrix

From this matrix, the necessary input variables were determined for each of the regressed variables. For instance, to enhance the regression of T , C_A was included as an input variable due to its

strong correlation coefficient with this variable. Similarly, the classifier’s power was enhanced by including the ISE as part of the inlet variables to the *Pass* classifier’s model. The input values to the surrogate models were standardized to ensure that no single variable disproportionately influenced the model. It was not possible to standardize all values using the same scaler, as each model had a different number and size of input variables. Therefore, a separate scaler was used for each model to appropriately match its specific input requirements. Note that in this case, there was an overrepresented class, the “pass” class. To prevent the agent from naïvely selecting “pass”, the Synthetic Minority Over-sampling Technique (SMOTE) was applied to balance the dataset and improve the classifier’s performance. SMOTE is a technique that can be easily found in Python libraries such as scikit-learn.

The surrogate models were implemented as fully connected NN utilizing a ReLU activation function. Details regarding the hidden layer sizes, the R^2 score and the root mean squared error (RMSE) on the testing dataset are provided in the following table (Table B.3):

Table B.3: Surrogate models’ details

Model	Architecture (hidden layers)	R^2 score	RMSE/F1-score
C_A regressor	[300, 200, 100]	0.9347	0.0514
T regressor	[300, 200, 100]	0.9268	5.0113
ISE_R regressor	[100, 100]	0.9904	0.2506
$\widehat{Label}_R$ classifier	[100, 100]	0.9895	0.99/0.96

The architecture was determined through various experiments, revealing that for the regressors of C_A and T , a structure with a decreasing number of neurons was more effective than one with a fixed number. Conversely, for the ISE_R regressor and the $\widehat{Label}_R$ classifier, the hidden layers consisted of the same number of neurons. Note that the for the $\widehat{Label}_R$ classifier instead of calculating the RMSE the F1-score is presented, 0.99 for class $\widehat{Label}_R = 1$ and 0.96 for class $\widehat{Label}_R = 0$.

It is important to acknowledge that there is some discrepancy in the predictions of the output variables by the surrogate models. A table comparing the values of C_A , T and ISE determined by the mechanistic model and those regressed by the NN surrogate models is presented as follows:

Table B.4: Surrogate model validation

Model	Parameter	CSTR 1	CSTR 2
Mechanistic	C_A [mol/L]	0.10022	0.04923
	T [K]	373.4695	356.9482
	ISE	0.3423	1.53e-3
Surrogate	C_A [mol/L]	0.09802	0.04889
	T [K]	377.5939	353.6407
	ISE	0.3469	2.638e-4

As observed, the differences between the predictions of the surrogate models and the values from the mechanistic models fall within the range specified by the surrogate models' RMSE.

3. Optimization formulation

The optimization formulation for Case Study 1 is presented in Equation B.11. This problem was translated into Pyomo (Bynum et al., 2021), a Python-based optimization modeling language, for its solution.

$$\min \sum_{j=1}^{N_{reactors}} \frac{V_{R,j} - V_R^{min}}{V_R^{max} - V_R^{min}} + \left(1 - \frac{\ln(ISE_{R,j})}{\ln(ISE_R)_{min}} \right) \quad (B11.1)$$

$$s. t. r_j = k_j C_{A,j} \quad \forall j \in J \quad (B11.2)$$

$$k_j = k_0 e^{-\frac{E_A}{RT_j}} \quad \forall j \in J \quad (B11.3)$$

$$V_{R,j} \frac{dC_{A,j}}{dt} = q(C_{A_{in,j}} - C_{A,j}) - V_{R,j} k_j C_{A,j} \quad \forall j \in J \quad (B11.4)$$

$$V_{R,j} \rho C_P \frac{dT_j}{dt} = \rho q(T_{in,j} - T_j) + (-\Delta H_R) V_{R,j} k_j C_{A,j} + U A_R (T_{C,j}^R - T_j) \quad \forall j \in J \quad (B11.5)$$

$$err_{R,j} = C_{A,j} - C_{A,j}^{sp} \quad \forall j \in J \quad (B11.6)$$

$$\frac{dI_{R,j}}{dt} = err_{R,j} \quad \forall j \in J \quad (B11.7)$$

$$T_{C,j}^R = T_{C,nom}^R + K_{C,j}^R \left(err_{R,j} + \frac{I_{R,j}}{\tau_{I,j}^R} \right) \quad \forall j \in J \quad (B11.8)$$

$$ISE_{R,j} = \int_{t_0}^{t_f} (C_{A,j} - C_{A,j}^{sp})^2 dt \quad \forall j \in J \quad (B11.9)$$

$$T_j \leq 400 \quad \forall j \in J \quad (B11.10)$$

$$C_{A,N_{reactors}} \leq 0.05 \quad (B11.11)$$

$$V_{R,j} \in [50, 200] \quad \forall j \in J \quad (B11.12)$$

$$C_{A,j}^{sp} \in [0.045, 1.0] \quad \forall j \in J \quad (B11.13)$$

$$K_{C,j}^R \in [20, 150] \quad \forall j \in J \quad (B11.14)$$

$$\tau_{l,j}^R \in [0.01, 0.5] \quad \forall j \in J \quad (\text{B11.15})$$

$$t \in (t_0, t_f] \quad (\text{B11.16})$$

$$J \in \{1, N_{\text{reactors}}\} \quad (\text{B11.17})$$

The set $J \in \{1, N_{\text{reactors}}\}$ refers to the index of reactors, where J can take any integer value between 1 and N_{reactors} , inclusive, with N_{reactors} representing the total number of reactors allowed in the system.

4. Supporting figures

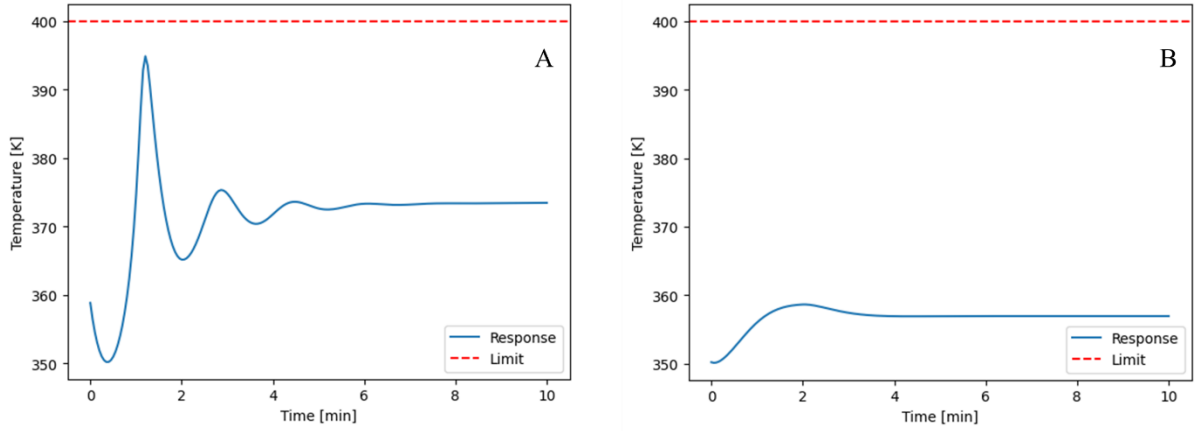


Figure B.2: Temperature constraint violation Scenario A: A) CSTR 1 B) CSTR 2

Figure B.2 presents two charts illustrating the temperature constraint violations for *Scenario A*. In both cases, it is observed that the design chosen by the agent ensures no violations of this constraint, with temperatures consistently staying below 400 K. It is evident that the greatest temperature swing is observed in the first reactor, where the maximum temperature peak reaches just over 390 K before stabilizing around 372 K. In contrast, the second reactor only experiences a slight increase in temperature, stabilizing around 355 K.

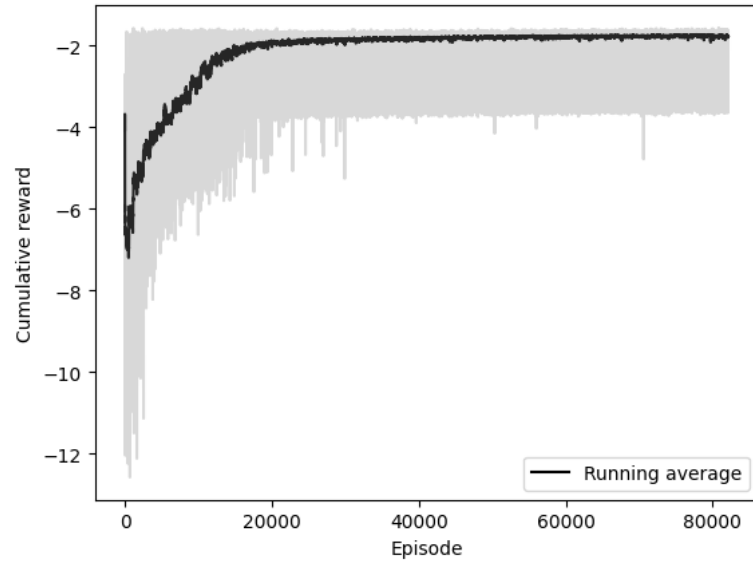


Figure B.3: Learning curve for PPO agent Scenario C

Figure B.3 presents the learning curve of the agent for *Scenario C*. In this case, it is observed that after the initial 20,000 episodes, which serve as the exploration phase for the agent, the curve reaches a plateau around a value of -2. In this case, it is observed that the agent stagnates in a reward region that spans from -4 to -2. This indicates that the agent encounters various flowsheets to address the problem, with the one featuring a single reactor being predominant. This is evident when evaluating the agent, particularly by noting that the running mean levels off at the top of the reward region.

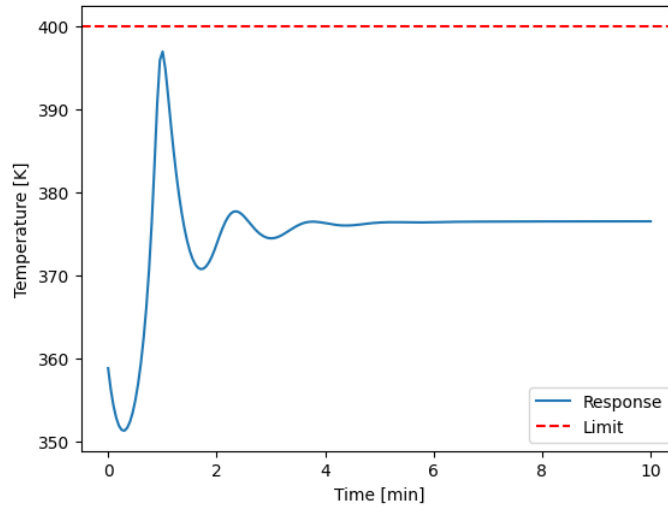


Figure B.4: Temperature constraint violation Scenario C

Similar to Figure B.2, Figure B.4 shows the temperature constraint violation for *Scenario C*. In this case, it is observed that the selected reactor design consistently remains within the established limits, without violating the 400 K constraint, and reaches a peak temperature close to 395 K. After this peak, the temperature in the reactor decreases, stabilizing around 375 K.

Appendix C

Chapter 4 – Case study 2

This appendix provides a more in-depth examination of various components of case study 2. Among these components is a thorough description of the mathematical model used for the isothermal CSTR, the isothermal binary distillation column and a more detailed analysis of the surrogate models employed for both UOs.

CSTR

1. Mechanistic model

The dynamic model of the CSTR was taken from (Bequette, 2002) and considered a CSTR with variable holdup. Inside of the reactor two chemical reactions were carried out in parallel: $A \rightarrow B$ and $A \rightarrow C$. It is assumed that the rate of reaction of both reactions is first order with respect to the concentration of reactant A (C_A):

$$r_1 = k_1 C_A \quad (C1)$$

$$r_2 = k_2 C_A \quad (C2)$$

The rate constants k_1 and k_2 are known. This model does not have any associated energy balance, which is why the reaction rates of the two reactions are independent of the temperature. The initial development of the CSTR model is based on the following assumptions:

- a) The CSTR is perfectly mixed
- b) The mass densities of the feed and product streams are equal and constant.
- c) The liquid volume of the tank (V^L) is variable

$$\frac{dV^L}{dt} = \dot{V}_{in} - \dot{V}_{out} \quad (C3)$$

where $\dot{V}_{in}$ is the inlet volumetric flow rate and $\dot{V}_{out}$ is the outlet volumetric flow rate. The mass balance for the reactant can be written as follows:

$$\frac{dV^L C_A}{dt} = \dot{V}_{in} C_{A_{in}} - \dot{V}_{out} C_A - \sum_{j=1}^{N_{par}} k_j C_A \quad (C4)$$

where N_{par} is the total number of parallel reactions. Expanding the left-hand side of C4:

$$\frac{dV^L C_A}{dt} = V^L \frac{dC_A}{dt} + C_A \frac{dV^L}{dt} \quad (C5)$$

Combining C3-C5, the following equation is obtained:

$$\frac{dC_A}{dt} = \frac{\dot{V}_{in}}{V^L} (C_{Ain} - C_A) - \sum_{j=1}^{N_{par}} k_j C_A \quad (C6)$$

The mass balances for the products (B and C) are derived in a similar manner to that of the reactant:

$$\frac{dC_B}{dt} = \frac{\dot{V}_{out}}{V^L} (-C_B) + k_1 C_A \quad (C7)$$

$$\frac{dC_C}{dt} = \frac{\dot{V}_{out}}{V^L} (-C_C) + k_2 C_A \quad (C8)$$

Note that in the mechanistic model, equations C7 and C8 were not solved directly, but rather their steady-state approximations were used.

$$C_B = C_{Bin} + \frac{V^L k_1 C_A}{\dot{V}_{out}} \quad (C9)$$

$$C_C = C_{Cin} + \frac{V^L k_2 C_A}{\dot{V}_{out}} \quad (C10)$$

The values of V^L , $\dot{V}_{out}$, and C_A were taken as the final values from the vectors generated by the mechanistic model. Meanwhile, the values of C_{Bin} and C_{Cin} were tracked by the model and stored in the observation vector received by the RL agent. Note that this modification was implemented to simplify the model and to avoid errors in the simulation. By doing so, the need to estimate the initial values of C_B and C_C at intermediate steps was eliminated, specially because the exact quantity previously converted of C_B and C_C depends on a previous CSTR and it is not known during the input variable determination. Also, through various experiments, it was observed that the steady-state approximation with respect to the final value of the dynamic response was practically the same. By making this

simplification, the reactor model is expressed solely in terms of the reactant, thereby avoiding infeasible initial values for C7 and C8, which could lead to infeasible results.

In this problem, it is assumed that there is a disturbance with a 50% increase in the inlet flowrate ($\dot{V}_{in}$) at time $t = 2$. To mitigate the effects of this disturbance, a control system is implemented on the equipment. This system aims to control the outlet concentration of the reactor by manipulating the outlet flowrate. The controller integrated into the reactor is a PI controller represented by the following equations:

$$err_R = C_A - C_A^{sp} \quad (C11)$$

$$\frac{dI_R}{dt} = err_R \quad (C12)$$

$$\dot{V}_{out} = \dot{V}_{out_{nom}} + K_C^R \left(err_R + \frac{I_R}{\tau_I^R} \right) \quad (C13)$$

where C_A^{sp} is the desired concentration set-point of reactant A, $\dot{V}_{out_{nom}}$ is the nominal outlet volumetric flow rate, K_C^R is the proportional gain of the PI controller and τ_I^R is the integrated time of the PI controller. The parameters used in this case study were the following (Table C.1):

Table C.1: Parameters' values

Parameter	Value
$\dot{V}_{in}$ [L/min]	1
C_{A0} [mol/L] (first inlet stream)	1
k_1 [min ⁻¹]	0.3
k_2 [min ⁻¹]	0.25
$\dot{V}_{out_{nom}}$ [L/min]	1

As for the design variables and their bounds, they were as follows (Table C.2):

Table C.2: Design variables' bounds

Design variables	Bounds
V_{in}^L [L]	3 – 6
C_A^{sp} [mol/L]	0.045 – 1.0
K_C^R $\left[\frac{L^2}{mol \cdot min} \right]$	-200 – -80
τ_I^R [1/min]	1 – 7

2. Surrogate model

The dataset used to train the CSTR surrogate models was generated using LHS and comprised around 85,000 combinations of the inlet variables of the reactor's model, i.e., inlet concentration, initial liquid reaction volume, set-point concentration, and the controllers' proportional gain and time integral. Like in the previous case study, it was ensured, when generating the combinations matrix, that the input concentration was always greater than the set-point concentration. All the combinations were evaluated in the isothermal CSTR model presented in the previous sub-section. The outputs of the model for each combination included a vector of the outlet concentrations of reactant A (C_A) and a vector containing the profile of the liquid volume (V^L). From C_A , the ISE was calculated (Equation B9), while with V , a metric (V_{cons}^L) was calculated (Equation C14), this metric determined if the design proposed violated the reactor design volume constraint (which in this case was 8 liters) or not.

$$cons_V = \begin{cases} 0, & V^L \geq V_{cons}^L \\ 1, & V^L < V_{cons}^L \end{cases} \quad (C14)$$

Similar to the previous case study, $cons_V$ represents the variable $cons$ in Algorithm 4.1 presented in the manuscript. From the vectors outputted by the mechanistic model, the endpoint values for C_A and V^L were taken, C_A and V^L respectively. Then a correlations matrix was built:

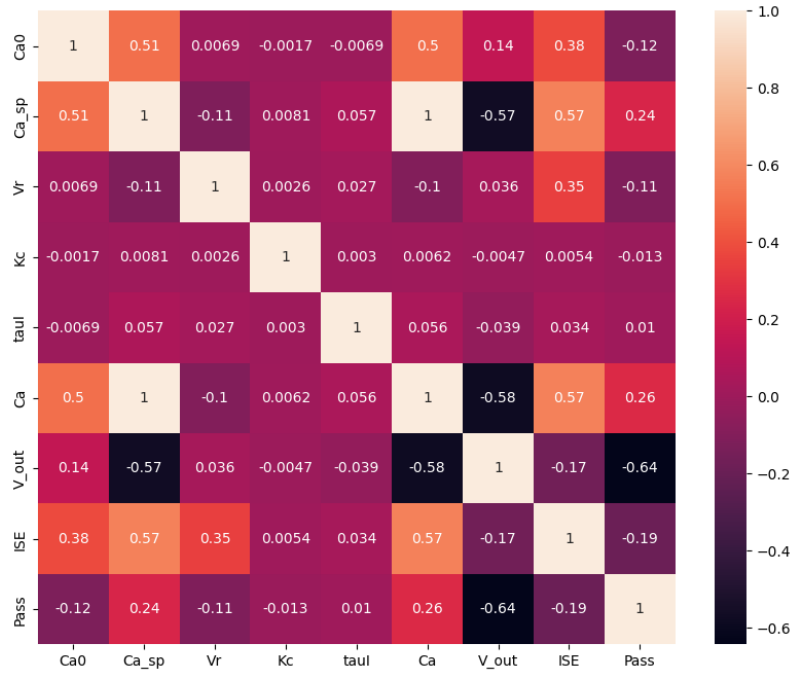


Figure C.1: Correlation matrix

Similar to the previous case, the correlation matrix helps determine which variables should be used as input variables in the surrogate models to predict the regressed variable. In this case, no regressed variable was used as an output variable, so all models used C_{A_0} , C_A^{sp} , V_{in}^L , K_C^R , and τ_I^R as input variables.

The surrogate models were implemented as fully connected NN utilizing a ReLU activation function. Details regarding the hidden layer sizes, the R^2 score and the RMSE on the testing dataset are provided in the following table (Table C.3):

Table C.3: Surrogate models' details

Model	Architecture (hidden layers)	R^2 score	RMSE/F1-score
C_A regressor	[300, 200, 100]	0.9960	0.0125
V^L regressor	[300, 200, 100]	0.9990	0.2849
ISE_R regressor	[100, 100]	0.9985	0.0607
$\widehat{Label}_R$ classifier	[100, 100]	0.9803	0.99/0.95

The architecture of the surrogate models for both the previous case study and the present one was the same. The reason behind this decision was the good results obtained in the previous case, which were sought to be replicated for the current study. All models achieved high test R^2 score which indicated a good selection of the models' architecture. Same as in the past case study, for the classifier the F1-score is presented, with values of 0.99 for class $\widehat{Label}_R = 1$ and 0.95 for class $\widehat{Label}_R = 0$.

3. Supporting figures

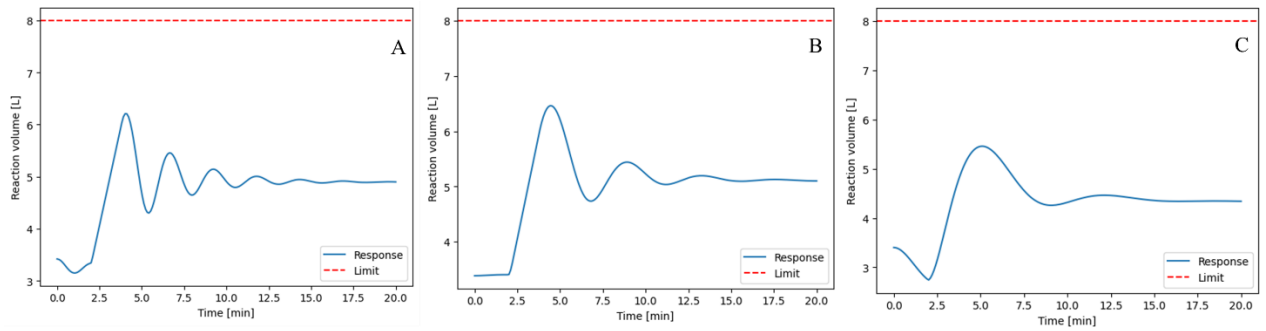


Figure C.2: Volume constraint violation Scenario D: A) CSTR 1 B) CSTR 2 C) CSTR 3

Figure C.2 presents three graphs illustrating the volume constraint violation for *Scenario D*. In all three cases, it is observed that the design chosen by the agent ensures no violations of this constraint, with volumes consistently remaining below 8 liters. Significant fluctuations in volume are observed in

the first two reactors, with the second reactor reaching volumes close to 6.5 liters. This behavior occurs because, in the initial reactors, the agent primarily aims to damp the disturbance while maximizing the conversion of the reactant. Ultimately, after 10 minutes, all three reactors stabilize at different volumes.

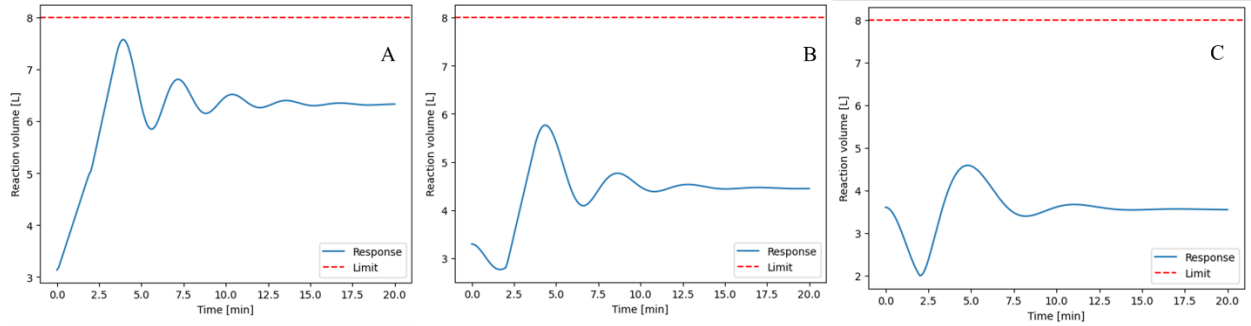


Figure C.3: Volume constraint violation Scenario E: A) CSTR 1 B) CSTR 2 C) CSTR 3

As in the previous figure, this figure presents three graphs illustrating the volume constraint violation for *Scenario E*. In all three cases, it is observed that the design selected by the agent prevents any violations of the volume constraint. It is worth noting the volume profile in the first reactor, which undergoes significant changes, increasing from 3 to 7 liters in a short period, with fluctuations approaching the imposed volume limit, reaching up to 7.5 liters. Despite this behavior, after 10 minutes, the value stabilizes around 6.5 liters. In the other reactors, fluctuations and changes in volume are not as pronounced, once again stabilizing after around 10 minutes.

Distillation column (DC)

1. Mechanistic model

The dynamic model of the DC was taken from (Schweiger and Floudas, 1998). Note that the entire model is formulated in terms of the most volatile component, which in this case is product B. The goal of this DC is to separate a saturated liquid feed into bottoms and distillate products of specified purity. For this column the total number of trays, the location of the feed, the diameter of the column and the set-point purities for both the tops and bottoms must be determined. The model of the DC is based on the following assumptions: equimolar overflow, constant relative volatility (α), a partial reboiler and total condenser are considered. The full model is presented as follows:

- Reboiler component balance

$$M_R \frac{dx_B}{dt} = L_1(x_1 - x_B) + V(x_B - y_B) \quad (C15)$$

- Tray component balance

$$M^L \frac{dx_i}{dt} = L_{i+1}(x_{i+1} - x_i) + V(y_{i-1} - y_i) + feed_n F^L(z - x_i) + reflux_n R(x_D - x_i) \quad (C16)$$

- Condenser component balance

$$M_c \frac{dx_D}{dt} = V(y_N - x_D) \quad (C17)$$

- Reboiler total balance

$$\beta \frac{dB}{dt} = L_1 - V - B^L \quad (C18)$$

- Tray total balance

$$\beta \frac{dL_i}{dt} = L_{i+1} - L_i + feed_n F^L + reflux_n R \quad (C19)$$

- Condenser component balance

$$\beta \frac{dD}{dt} = V - R - D^L \quad (C20)$$

- Reboiler equilibrium

$$y_B = \frac{\alpha x_B}{1 + x_B(\alpha - 1)} \quad (C21)$$

- Tray equilibrium

$$y_i = \frac{\alpha x_i}{1 + x_i(\alpha - 1)} \quad (C22)$$

- Molar holdup

$$M^L = 7.538115 \left(\left(\frac{0.0014134}{D_C} \right)^{\frac{2}{3}} + 0.0254 \right) D_C^2 \quad (C23)$$

- Tray time constant

$$\beta = 0.05271 D_C^{\frac{4}{3}} \quad (C24)$$

- Molar holdup in the condenser and the reboiler

$$M_C = M_R = 100 M^L \quad (C25)$$

where x_i represent the liquid fractions from each tray (i can take values from 1 to N which is the maximum number of trays in the DC), x_D the liquid fraction at the distillate, x_B the liquid fraction at the bottoms, z the liquid fraction at the feed, y_B the vapor fraction at the bottoms, y_i the vapor fractions from each tray, L_i the liquid flow rate from each tray, R the reflux flow rate, V the vapor boilup, B^L the bottoms flow rate, D^L the distillate flow rate, F^L the feed rate and D_C the diameter of the DC. The structural alternatives are represented by the binary variables $feed_n$ and $reflux_n$, which denote the presence of the feed and reflux to tray n , respectively.

When designing the input matrix for the mechanistic model, the following logical constraints were considered:

- The existence of only one feed stream

$$\sum_{n=1}^N feed_n = 1 \quad (C26)$$

- The existence of only one reflux stream

$$\sum_{n=1}^N reflux_n = 1 \quad (C27)$$

- The reflux enters at least 4 trays above the feed stream

$$\sum_{n=1}^N (n)(reflux_n) - (n)(feed_n) \geq 4 \quad (C28)$$

In this problem, it is assumed that there is a disturbance with a 50% increase in the inlet flowrate (F) at time $t = 2$. To account for the same disturbance in both the CSTR and the DC, it was assumed that the conversion between volumetric flow rate and molar flow rate was 1:1. To mitigate the effects of this disturbance and achieve the desired purity set-points, two PI control systems are implemented: one on the condenser and one on the reboiler. In the condenser x_D is controlled by manipulating R , while in the reboiler x_B is controlled by manipulating V . The PI controllers' equations are as follows:

$$\frac{dI_B}{dt} = x_B - x_B^{sp} \quad (C29)$$

$$\frac{dI_D}{dt} = x_D - x_D^{sp} \quad (C30)$$

$$V = V_{ss} + K_C^B(x_B - x_B^{sp}) + \frac{K_C^B}{\tau_I^B} I_B \quad (C31)$$

$$R = R_{ss} + K_C^D(x_D - x_D^{sp}) + \frac{K_C^D}{\tau_I^D} I_D \quad (C32)$$

where x_B^{sp} denoted the set-point in the bottoms liquid fraction, x_D^{sp} the set-point in the distillate liquid fraction, K_C^B and τ_I^B the vapor boilup controller inputs, and K_C^D and τ_I^D the reflux rate controller inputs. The parameters used in this case study were the following (Table C.4):

Table C.4: Parameters' values

Parameter	Value
F [mol/min]	1
α	2.5
V_{ss} [mol/min]	1.258
R_{ss} [mol/min]	0.8097

Table C.5 lists the design variables and their corresponding bounds.

Table C.5: Design variables' bounds

Design variable	Bounds
z	0.3 – 0.6
N	10 – 20
D_C [m]	0.3 – 0.6
$feed_N$	2 – 10
$reflux_N$	8 – 19
K_C^B [mol/min]	5 – 20
τ_I^B [1/min]	0.01 – 0.3
K_C^D [mol/min]	-20 – -5
τ_I^D [1/min]	0.01 – 0.3
x_B^{sp}	0.01 – 0.1
x_D^{sp}	0.9 – 0.99

2. Surrogate model

The dataset used to train the DC surrogate models was generated using LHS and comprised around 22,000 combinations of the inlet variables of the column: z , N , D_C , $feed_N$, $reflux_N$, K_C^B , τ_I^B , K_C^D , τ_I^D , x_B^{sp} and x_D^{sp} . For this case, it was verified that $feed_n$ was always less than $reflux_N$ by a difference of 4, as presented in the previous section (Equation C28). All combinations were evaluated in the isothermal DC model presented in the previous sub-section. For each combination, the model outputted a series of vectors: N vectors corresponding to each of the liquid fraction profiles at each stage, N vectors of each of the molar flow profiles at each stage, and N vectors of each of the vapor fraction profiles at each stage. In this case, we focused only on the vector of the output profile in the distillate ($\mathbf{x}_D$) and the vector of the output profile in the bottoms ($\mathbf{x}_B$). From $\mathbf{x}_D$, the ISE for the tops was calculated (Equation B9), for the bottoms the ISE was also calculated with $\mathbf{x}_B$ (Equation B9). The dynamic model of the DC was programmed in Python, primarily using the Gekko package (Beal et al., 2018). A key advantage of using Gekko for solving optimization problems is that if the proposed design violates constraints or results in an infeasible solution, e.g., nonpositive concentrations, Gekko generates an error thus providing a direct identification of the infeasible designs. Adding to that, the necessity to handle a substantial number of equations, initializations, and constraints made Gekko an attractive simulation framework for this unit operation.

The constraints considered in this case included compliance of all DAEs (equality constraints) C15-C20 and C29-C30, as well as prevention of crossover in the mole fractions, i.e.,

$$x_B \leq x_1 \leq \dots \leq x_N \leq x_D \quad (C33)$$

Furthermore, there are constraints requiring that all process streams in the model must be positive, ensuring that no stream dries up:

$$\begin{aligned} V &\geq 0 \\ L_i &\geq 0 \quad \forall i \in I \\ R &\geq 0 \\ I &\in \{1, N\} \end{aligned} \quad (C34)$$

For this case, $cons_{DC}$ represents the variable $cons$ in Algorithm 4.1 presented in the manuscript. For this unit operation $cons_{DC}$ determined whether the proposed design violated any of the constraints outlined above, i.e.,

$$cons_{DC} = \begin{cases} 0, & \text{Constraint violation} \\ 1, & \text{No constraint violation} \end{cases} \quad (C35)$$

The primary drawback of this approach was the significant computation time required to simulate all scenarios, taking approximately 8 hours to complete. As a result, only 22,000 combinations could be simulated for the DC, compared to the 85,000 combinations simulated for the reactors. For the DC, five surrogate models were determined: four regressors and one classifier. The regressors were tasked with determining the endpoint of $\mathbf{x}_B$ (x_B), the endpoint of $\mathbf{x}_D$ (x_D), the ISE of the distillates (ISE_D) and the ISE for the bottoms (ISE_B). The classifier determined whether the simulation in Gekko failed or not.

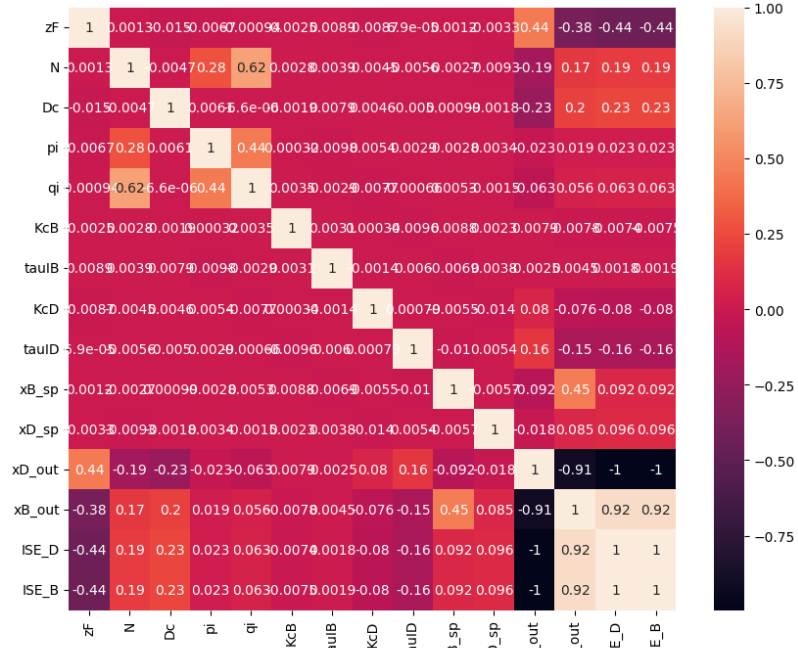


Figure C.4: Correlation matrix

Similar to the other unit operations, a correlation matrix was constructed for the DC. In this case, Figure C.4 shows a matrix featuring designs that did not violate any constraints, meaning they were labeled as "No constraint violation" by the classifier. This approach was taken because many designs failed during the simulation and including both "No constraint violation" and "Constraint violation" designs would obscure the true effect that input variables have on the regressed variables.

Details regarding the hidden layer sizes, activation functions, the R^2 score and the RMSE on the testing dataset are provided in the following table.

Table C.6: Surrogate models' details

Model	Activation function	Architecture (hidden layers)	R^2 score	RMSE/F1-score
x_B regressor	Tanh	[512, 256, 128, 64]	0.9725	0.0044
x_D regressor	Tanh	[512, 256, 128, 64]	0.9833	0.0034
ISE_B regressor	ReLU	[100, 100]	0.9635	0.3071
ISE_D regressor	ReLU	[100, 100]	0.9549	0.3265
Classifier	ReLU	[100, 100]	0.9592	0.98/0.88

As observed in Table C.6, the architecture of the surrogate models for predicting x_B and x_D differs from the previous surrogate models. The primary change lies in the type of activation function

used; in this case, a tanh activation function was chosen due to the restricted domain of the problem, which required a well-bounded response. Additionally, the decreasing architecture in the number of neurons proved effective in making predictions. Regarding the regressors for ISE_B and ISE_D , the architecture and activation function remained the same as in the previous cases, as did the classifier's architecture and activation function. Same as in the past case study, for the classifier the F1-score is presented, with values of 0.98 for class “No constraint violation” and 0.88 for class “Constraint violation”.

Appendix D

Chapter 4 – Case study 3

This appendix provides a more in-depth examination of various components of case study 3. Among these components is a thorough description of the mathematical model used for the isothermal flash tank (FT) and a more detailed analysis of the surrogate models employed for that UO. Supporting images are also included, showing the volume constraint satisfaction for *Scenarios F* and *G*.

1. Mechanistic model

The dynamic model of the FT was derived from (Schweiger and Floudas, 1998). The primary objective of the FT is to separate a saturated liquid feed into two distinct outflows: a vapor stream and a liquid stream, each with a specified molar fraction. To achieve this, the liquid molar fraction set-point must be determined. The FT model operates under the following key assumptions: constant relative volatility (α), no mass accumulation within the tank, and constant temperature and pressure throughout the process. These assumptions simplify the system dynamics, allowing for more straightforward analysis and control of the separation process. Note that the entire model is formulated in terms of the most volatile component, which in this case is product B.

The full model is presented as follows:

- Component balance equation

$$\frac{dx}{dt} = F_{in}z - V_{FT}y_V - L_{FT}x_L \quad (D1)$$

- Total balance equation

$$\frac{dL_{FT}}{dt} = F_{in} - V_{FT} - L_{FT} \quad (D2)$$

- Equilibrium equation

$$y_V = \frac{\alpha x_L}{1 + x_L(\alpha - 1)} \quad (D3)$$

Here, x_L and y_V represent the molar fractions in the liquid and vapor phases, respectively. F_{in} is the inlet flowrate, V_{FT} is the vapor outflow, L_{FT} is the liquid outflow, and z denotes the feed's liquid

fraction. Note that the equations for the FT were derived from the DC's reboiler equations presented in the previous appendix. This approach is justified based on the underlying physical principles shared by both processes. Reboilers and flash tanks operate on similar thermodynamic concepts, primarily the phase equilibrium between liquid and vapor phases under specific pressure and temperature conditions.

In this problem, it is assumed that there is a disturbance with a 50% increase in the inlet flowrate (F_{in}) at time $t = 2$. To mitigate the effects of this disturbance, a control system is implemented on the equipment. This system aims to control the outlet liquid molar fraction of the most volatile component by manipulating the vapor the outlet flowrate. The controller integrated into the flash tank is a PI controller represented by the following equations:

$$err_{FT} = x_L - x_{L_{sp}} \quad (D4)$$

$$\frac{dI_{FT}}{dt} = err_{FT} \quad (D5)$$

$$V_{FT} = V_{FT}^{ss} + K_C^{FT} \left(err_{FT} + \frac{I_{FT}}{\tau_I^{FT}} \right) \quad (D6)$$

where $x_{L_{sp}}$ is the desired set-point of the molar fraction in the liquid phase, V_{ss} is the steady-state vapor outflow, K_C^{FT} is the proportional gain of the PI controller and τ_I^{FT} is the integrated time of the PI controller. The parameters used in this case study were the following (Table D.1):

Table D.1 Parameters' values

Parameter	Value
$V_{FT}^{ss} [mol/min]$	0.5
α	2.5

As for the design variables and their bounds, they were as follows (Table D.2):

Table D.2: Design variables' bounds

Design variables	Bounds
$F_{in} [mol/min]$	0.01 – 1
z	0.1 – 0.9
x_L^{sp}	0.05 – 0.85
$K_C^{FT} [mol/min]$	1 – 10
$\tau_I^{FT} [1/min]$	0.01 – 0.9

In contrast to the previous UO, the inlet flowrate to the FT model must be treated as a design variable. This consideration is primarily necessary to accommodate the possibility of multiple FTs operating in series. For the design of FTs operating in series, the inlet flowrate is taken as the outflow with the highest molar quantity from the preceding unit. This approach accounts for the inevitable material loss that occurs during the separation process.

2. Surrogate models

The dataset used to train the FT surrogate models was generated using LHS and comprised around 15,000 combinations of the inlet variables of the tank: z , F_{in} , x_L^{FT} , K_C^{FT} and τ_I^{FT} . All the combinations were evaluated in the FT model presented in the previous sub-section. The outputs of the model for each combination included a vector of the molar fraction in the liquid phase ($\mathbf{x}_L$) and a vector containing the profile of the liquid outflow ($\mathbf{L}_{FT}$). The ISE (Equation B9) was calculated from $\mathbf{x}_L$. Similar to the DC from Appendix C, the dynamic model of the FT was programmed in Python, using the Gekko package (Beal et al., 2018).

The constraints considered in this case encompassed the satisfaction of the DAEs as equality constraints, specifically equations D1, D2, and D5. Additionally, a constraint was imposed to ensure that all process streams in the model remain positive, thereby preventing any stream from drying up.

$$\begin{aligned} V_{FT} &\geq 0 \\ L_{FT} &\geq 0 \end{aligned} \quad (D7)$$

For this case, $cons_{FT}$ represents the variable $cons$ in Algorithm 4.1 presented in the manuscript. For this unit operation $cons_{FT}$ determined whether the proposed design violated any of the constraints outlined above, i.e.,

$$cons_{FT} = \begin{cases} 0, & \text{Constraint violation} \\ 1, & \text{No constraint violation} \end{cases} \quad (C35)$$

As previously mentioned, one of the main drawbacks of using Gekko is the high computational time, as simulating the 15,000 combinations required approximately 3 hours to complete. For the FT, four surrogate models were determined: three regressors and one classifier. The regressors were tasked with determining the endpoint of $\mathbf{x}_L$ (x_L), the endpoint of $\mathbf{L}_{FT}$ (L_{FT}) and the ISE for the molar fraction in the liquid outlet. It is important to note that, based on the values of x_L and L_{FT} , the values of y_V and V_{FT} were also calculated to ensure that the entire FT model was solved at the endpoint. These calculations were made because y_V and V_{FT} could be used as input variables for a hypothetical next FT, as the

determining factor for selecting one outflow over the other is its magnitude (the outflow with the most amount of material is chosen as the hypothetical next input flow).

The classifier determined whether the simulation in Gekko failed or not.

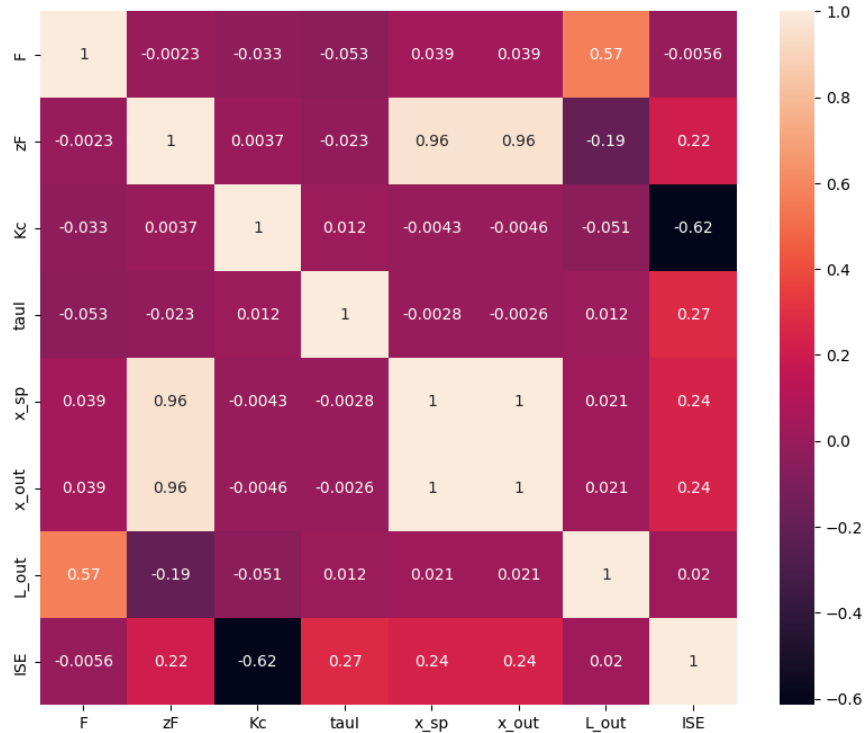


Figure D.1: Correlation matrix

As with the other unit operations, a correlation matrix was created for the FT. In this instance, Figure D.1 presents a matrix that includes only those designs that did not violate any constraints, which were classified as "No constraint violation" by the classifier. This strategy was adopted because many designs encountered failures during the simulation, and incorporating both classes' designs would obscure the true impact of the input variables on the regressed variables.

Details regarding the hidden layer sizes, activation functions, the R^2 score and the RMSE on the testing dataset are provided in the following table.

Table D.3: Surrogate models' details

Model	Activation function	Architecture (hidden layers)	R^2 score	RMSE/F1-score
x_L regressor	Tanh	[100,100,100,100]	0.9969	0.0109
L_{FT} regressor	ReLU	[100,100,100,100]	0.9860	0.0254
ISE_{FT} regressor	ReLU	[100, 100]	0.9899	0.1174
Classifier	ReLU	[100, 100]	0.9871	0.99/0.98

As observed in Table D.3, the x_L and L regressors employ a greater number of hidden layers compared to other surrogate models. However, unlike other unit operations, their architecture does not exhibit a decreasing trend; maintaining a consistent number of neurons per layer has yielded favorable results. All surrogate models demonstrate excellent performance, with an average R^2 score of approximately 0.99. Notably, only the x_L regressor utilizes the Tanh activation function; attempts to implement ReLU did not produce satisfactory outcomes. The Tanh function also assists in bounding the output variable. Additionally, the constructed classifier performs effectively, clearly distinguishing between the “No Constraint Violation” and “Constraint Violation” classes, indicating its robustness in classification tasks.

3. Supporting figures

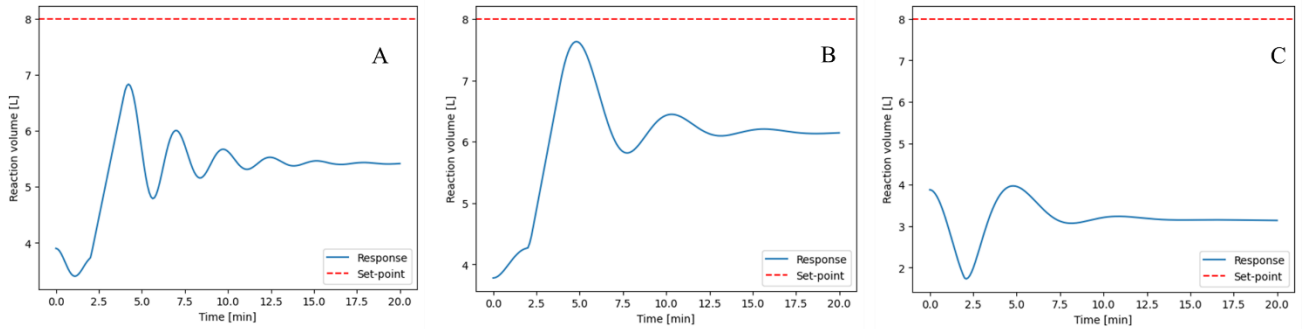


Figure D.2: Volume constraint violation Scenario F: A) CSTR 1 B) CSTR 2 C) CSTR 3

Figure D.2 presents three graphs illustrating the volume constraint violation for *Scenario F*. In the case of all three reactors, the agent ensures adherence to the design constraint of 8 liters. Notably, both the first and last reactors achieve this constraint comfortably. Conversely, the second reactor exhibits the most significant fluctuations, characterized by an exponential increase in volume during the initial 2.5 minutes, reaching a peak of 7.5 liters before stabilizing around 6.5 liters. Despite not having the highest initial volume among the three reactors, the dynamics of the process lead to this

behavior, which the controller designed by the agent effectively dampens. After 10 minutes, all reactors stabilize at a fixed operational volume.

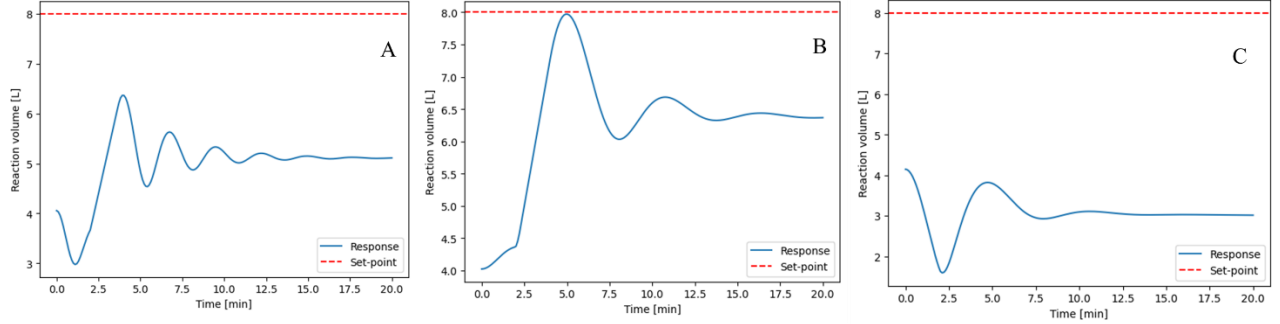


Figure D.3: Volume constraint violation Scenario G: A) CSTR 1 B) CSTR 2 C) CSTR 3

As in the previous figure, this figure presents three graphs illustrating the volume constraint violation for *Scenario G*. In this case, the three reactors preceding the FT consistently adhere to the volume constraint throughout the entire operation. Similar to the previous scenario, the second reactor exhibits the most significant challenges, experiencing an exponential increase in volume during the initial minutes of operation. However, this volume is subsequently regulated by the controller implemented by the agent. The maximum volume attained in this instance is 7.95 liters, which remains within the allowed operating volume. Additionally, while both the first and last reactors stabilize at approximately 10 minutes, the second reactor requires slightly more time to stabilize, achieving equilibrium around the 14-minute mark.

Appendix E

Chapter 4 – Case study 4

This appendix provides a more in-depth examination of various components of case study 4. It is important to note that all mechanistic models used in this case study are the same as those presented in Appendix C. Readers are encouraged to refer to Appendix C for a comprehensive description. The only modification pertains to the mechanistic model of the distillation column, where the relative volatility value (α) was adjusted to match the values presented in Table 4.11; however, the overall mechanistic model remains unchanged.

Note that a suitable real-world analogy for this setup is the isomerization of *n*-pentane into *iso*-pentane and neopentane. In this reaction, *n*-pentane (reactant A) is converted into its structural isomers, *iso*-pentane (product B) and neopentane (product C), through a catalytic process that is typically irreversible and follows first-order kinetics with respect to *n*-pentane concentration. This analogy reflects the reaction mechanisms and operational considerations of the model, as the isomerization process is conducted in the liquid phase. The choice of *n*-pentane and its isomers is particularly relevant, as their physical properties, including density, are nearly identical, simplifying modeling assumptions in a CSTR setup. Additionally, this isomerization is well-suited for separation processes due to the differences in boiling points among the three isomers, with neopentane being the most volatile, followed by *iso*-pentane, and finally *n*-pentane (Chi et al., 2017).

With these clarifications, this appendix includes the updated design variables table, revised to accommodate all three column designs. Additionally, it presents supporting figures for the volume constraints satisfaction for *Scenarios H* and *I* and the detailed specifications of the surrogate models architectures and score metrics for the three distillation columns.

1. Supporting figures

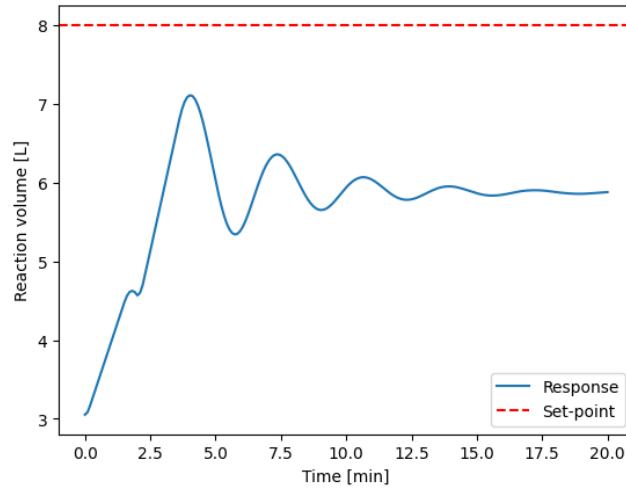


Figure E.1 Volume constraint violation scenario H

Figure E.1 presents a graph illustrating the volume constraint violation for *Scenario H*. In this graph, it can be observed that the design selected by the agent ensures no violations of the constraint, with volumes consistently remaining below 8 L. The reactor volume increases exponentially during the first few minutes, reaching a peak near 7 L, after which it stabilizes around 6 L after approximately 12.5 minutes.

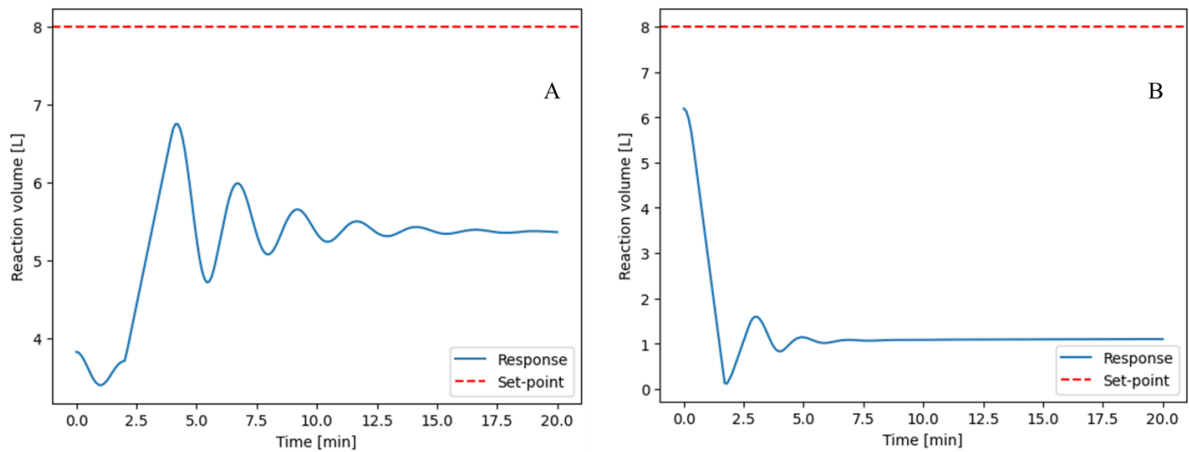


Figure E.2 Volume constraint violation Scenario I: A) CSTR 1 B) CSTR 2

As in the previous figure, this figure presents two graphs illustrating the volume constraint violation for *Scenario I*. In all two cases, it is observed that the design selected by the agent prevents any violations of the volume constraint. It is worth noting the volume profile in the first reactor, which

undergoes significant changes, increasing from 4 to 6 liters in a short period, with fluctuations approaching the imposed volume limit, reaching up to 6.8 liters. Despite this behavior, after 12.5 minutes, the value stabilizes around 5.5 liters. The second reactor exhibits a more intriguing behavior, as its volume decreases instead of increasing. It starts with a high reaction volume, near 7 liters, followed by an abrupt drop before stabilizing around 1.5 liters. This behavior is believed to occur because the conversion in this reactor is relatively low, and the outlet concentration was subject to near-immediate control (see Figure 4.27 in the manuscript).

2. DC design table

Table E.1 lists the design variables and their corresponding bounds. In order to determine the lower and upper bounds for the number of stages required for the separation of components in the distillation process, short-cut methods were utilized. Specifically, Fenske's equation was applied to calculate the minimum number of stages, N_{min} , needed at total reflux to achieve the desired separation between the more volatile component and the less volatile component.

$$N_{min} = \frac{\ln \left(\frac{x_D * (1 - x_B)}{x_B * (1 - x_D)} \right)}{\ln (\alpha)} \quad (E1)$$

Assuming molar fractions of 0.99 at the top and 0.01 at the bottom, as well as the relative volatility with the smallest value to achieve the highest number of trays (in this case, 2.5), the minimum number of stages required to achieve the separation is 11. This value is used as a starting point to determine the bounds required for the actual number of stages required for the separation.

Table E.1 Design variables

Design variable	Bounds
z	0.3 – 0.9
N	10 – 20
D_C [m]	0.3 – 0.6
p_k	2 – 10
q_k	8 – 19
K_C^B [mol/min]	5 – 20
τ_I^B [1/min]	0.01 – 0.3
K_C^D [mol/min]	-20 – -5
τ_I^D [1/min]	0.01 – 0.3
x_B^{sp}	0.01 – 0.1
x_D^{sp}	0.10 – 0.99

3. DC surrogate models

The regressors were tasked with determining the endpoint of x_B (x_B), the endpoint of x_D (x_D), the ISE of the distillates (ISE_D) and the ISE for the bottoms (ISE_B). The classifier determined whether the simulation in Gekko failed or not.

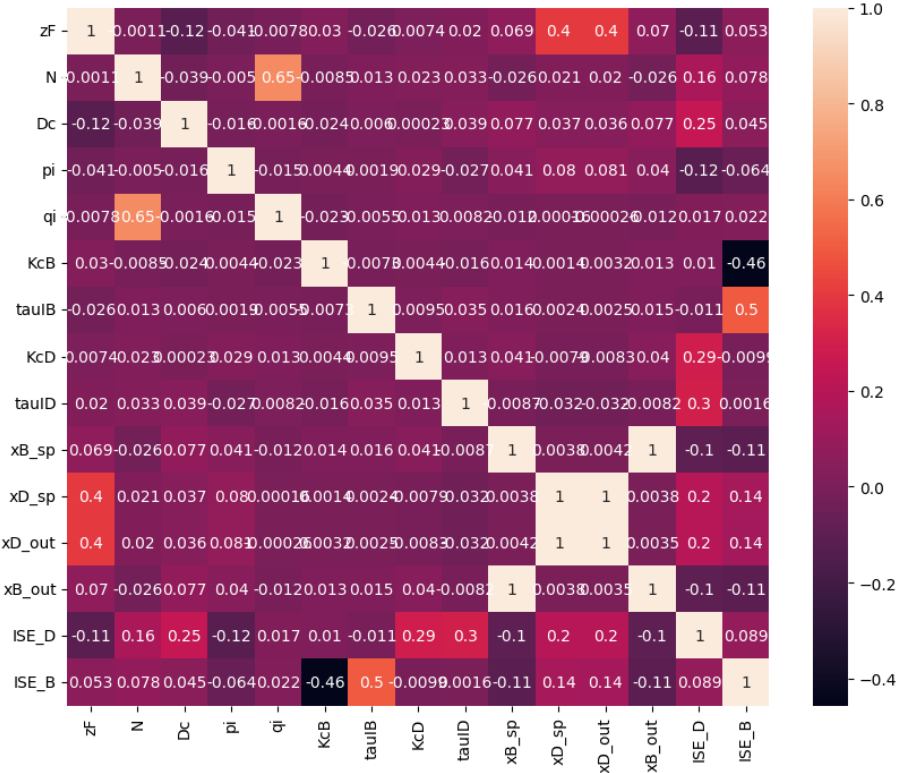


Figure E.3 Correlation matrix for DC with $\alpha=3.5$

Figure E.3 presents the correlation matrix of the different variables in the distillation column model with a relative volatility of 3.5. To observe the true effect of the input variables on the output variables, only designs that did not violate any constraints were considered for this matrix, meaning they were labeled as "pass" by the classifier. As seen, many variables have low or nearly zero correlations with the output variables. For example, x_D has a very high correlation with the input composition and with x_D^{sp} , but no correlation with the rest. The same behaviour is observed for x_B , having high correlation with x_B^{sp} and practically no correlation with the rest of the input variables. As expected, ISE_D and ISE_B show a correlation with the controller tuning parameters in the distillates and

in the bottoms respectively. It is worth noting that this correlation increased when the ISE value was transformed to $\ln(\text{ISE})$. For the surrogate models, all design variables were taken as input, excluding any regressed variables, even though some output variables may have certain correlations among them. Note that correlation matrices for the other two distillation columns with different relative volatility show similar trends and values to those observed in Figure E.3. Therefore, to avoid repetition, they are not included in this supplementary material.

Table E.2 presents the architecture details for the models used in the study, including information on hidden layer sizes, activation functions, the R^2 score, and the RMSE on the testing dataset.

Table E.2 Surrogate models' details for DCs

Model	Activation function	Architecture (hidden layers)	R^2 score	RMSE or F1-score
$\alpha = 2.5$				
x_B regressor	Tanh	[512, 256, 128, 64]	0.9725	0.0044
x_D regressor	Tanh	[512, 256, 128, 64]	0.9833	0.0034
ISE_B regressor	ReLU	[100, 100]	0.9635	0.3071
ISE_D regressor	ReLU	[100, 100]	0.9549	0.3265
Classifier	ReLU	[100, 100]	0.9592	0.98/0.88
$\alpha = 3.5$				
x_B regressor	Tanh	[512, 256, 128, 64]	0.9700	0.0044
x_D regressor	Tanh	[512, 256, 128, 64]	0.8637	0.0086
ISE_B regressor	ReLU	[100, 100]	0.8987	0.4880
ISE_D regressor	ReLU	[100, 100]	0.8887	0.6170
Classifier	ReLU	[100, 100]	0.9142	0.92/0.91
$\alpha = 4.5$				
x_B regressor	Tanh	[512, 256, 128, 64]	0.9726	0.0044
x_D regressor	Tanh	[512, 256, 128, 64]	0.9307	0.0067
ISE_B regressor	ReLU	[100, 100]	0.9469	0.3701
ISE_D regressor	ReLU	[100, 100]	0.6258	1.2096
Classifier	ReLU	[100, 100]	0.9288	0.95/0.89

The table presents the performance of various surrogate models across different α values in terms of the R^2 score and RMSE or F1-score. As observed, most models achieve R^2 scores above 85%, with the exception of the ISE_D regressor for $\alpha = 4.5$, which demonstrates lower performance. This discrepancy can be attributed to the need for additional simulations for that particular model, which resulted in some outliers that the model could not accurately predict. Furthermore, the choice of

activation function—specifically the Tanh function – for x_D and x_B was influenced by the restricted domain of the variables, which required a well-bounded response. The decreasing architecture in the number of neurons (i.e., [512, 256, 128, 64]) was also found to be effective in enhancing model performance, particularly for regression tasks. The classifiers show good accuracy. Despite the presence of overrepresented the overrepresented “No violation” class, the application of SMOTE effectively mitigates bias, as evidenced by the high F1 scores.