UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE CIÊNCIAS E TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA QUÍMICA

TESE DE DOUTORADO

SISTEMA DE CONTROLE DATA DRIVEN PARA COLUNAS DE DESTILAÇÃO UTILIZANDO DEEP REINFORCEMENT LEARNING

Gladson Euler Lima Júnior

Campina Grande – Paraíba
2023

Gladson Euler Lima Júnior

SISTEMA DE CONTROLE DATA DRIVEN PARA COLUNAS DE DESTILAÇÃO UTILIZANDO DEEP REINFORCEMENT LEARNING

Tese de doutorado apresentada ao Programa de Pós-graduação em Engenharia Química da Universidade Federal de Campina Grande como parte dos requisitos necessários para obtenção do título de Doutor em Engenharia Química.

Orientador

Prof. Dr. Luís Gonzaga Sales Vasconcelos

Campina Grande – Paraíba

2023

FICHA CATALOGRÁFICA

L732s Lima Júnior, Gladson Euler.

Sistema de controle data driven para colunas de destilação utilizando deep reinforcement learning / Gladson Euler Lima Júnior – Campina Grande, 2023.

111 f. : il. color.

Tese (Doutorado em Engenharia Química) - Universidade Federal de Campina Grande, Centro de Ciências e Tecnologia, 2023.

"Orientação: Prof. Dr. Luís Gonzaga Sales Vasconcelos."

Referências.

1. Controle Baseado em Dados. 2. Aspen Plus Dynamics. 3. Deep Q-Network. 4. Q-learning. 5. Algoritmo de Treinamento Genérico. I. Vasconcelos, Luís Gonzaga Sales. II. Título.

CDU 66.011(043)

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECÁRIA ITAPUANA SOARES DIAS GONÇALVES CRB-15/093

FOLHA DE ASSINATURAS

SISTEMA DE CONTROLE DATA DRIVEN PARA

COLUNAS DE DESTILAÇÃO UTILIZANDO DEEP REINFORCEMENT LEARNING

Gladson Euler Lima Júnior

Tese apresentada em 25/09/2023

Banca Examinadora:

gov.br Documento assinado digitalmente
LUIS GONZAGA SALES VASCONCELOS
Data: 17/10/2023 09:28:49-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Luis Gonzaga Sales Vasconcelos (Orientador)

gov.br Documento assinado digitalmente
ROMILDO PEREIRA BRITO
Data: 17/10/2023 10:32:52-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Romildo Pereira Brito (Examinador Interno)

gov.br Documento assinado digitalmente
KAROLINE DANTAS BRITO
Data: 17/10/2023 16:34:36-0300
Verifique em https://validar.iti.gov.br

Profª. Dra. Karoline Dantas Brito (Examinadora Interna)

Prof. Dr. Arioston Araújo de Morais Júnior (Examinador Externo)

gov.br Documento assinado digitalmente
THIAGO GONCALVES DAS NEVES
Data: 18/10/2023 21:54:29-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Thiago Gonçalves das Neves (Examinador Externo)

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
UNIDADE ACADEMICA DE ENGENHARIA QUIMICA
Rua Aprigio Veloso, 882, - Bairro Universitario, Campina Grande/PB, CEP 58429-900
Telefone: (83) 2101-1100
Site: http://cct.ufcg.edu.br

REGISTRO DE PRESENÇA E ASSINATURAS

ATA DA DEFESA PARA CONCESSÃO DO GRAU DE DOUTOR EM ENGENHARIA QUÍMICA, REALIZADA EM 25 DE SETEMBRO DE 2023 (Nº 143)

CANDIDATO(A): **Gladson Euler Lima Júnior**. COMISSÃO EXAMINADORA: Prof. Dr. Romildo Pereira Brito (**Presidente da Sessão e Examinador Interno/UFCG**), Prof. Dr. Luis Gonzaga Sales Vasconcelos (**Orientador/UFCG**), Profa. Dra. Karoline Dantas Brito (Examinadora Interna/UFCG), Prof. Dr. Thiago Gonçalves das Neves (Examinador Externo/IFRN) e Prof. Dr. Arióston Araújo de Morais Júnior (Examinador Externo/UFPB). TITULO DA TESE: "Sistema de Controle Data Driven para Colunas de Destilação Utilizando Deep Reinforcement Learning". ÁREA CONCENTRAÇÃO: Desenvolvimento de Processos Químicos - HORA DE INÍCIO: **14:00 horas** – LOCAL: por videoconferência. Em sessão pública, após exposição de cerca de 50 minutos, o(a) candidato(a) **Gladson Euler Lima Júnior** foi arguido(a) oralmente pelos membros da Comissão Examinadora, tendo demonstrado suficiência de conhecimento e capacidade de sistematização, no tema de sua tese, sendo **APROVADO.** Face à aprovação, declara o(a) Presidente da Comissão, achar-se a examinada, legalmente habilitado a receber o **Grau de Doutor**, no domínio da Engenharia Química, cabendo a Universidade Federal de Campina Grande, como direito, providenciar a expedição do Diploma, o que o mesmo faz jus. Na forma regulamentar, foi lavrada a presente ata, que é assinada por mim, Maricé Pereira de Araújo - Secretária - do PPGEQ e os membros da Comissão Examinadora e o(a) candidato(a). Campina Grande, 25 de setembro de 2023.

Maricé Pereira de Araújo
Secretária

Romildo Pereira Brito, Prof., Dr., UFCG
Presidente da Comissão e Examinador Interno

Luis Gonzaga Sales Vasconcelos, Prof.,Dr., UFCG
Orientador

Karoline Dantas Brito, Profa., Dra., UFCG
Examinadora Interna

Thiago Gonçalves das Neves, Prof., Dr., IFRN
Examinador Externo

Arióston Araújo de Morais Júnior, Prof., Dr., UFPB
Examinador Externo

Gladson Euler Lima Júnior
Candidato

Documento assinado eletronicamente por **LUIS GONZAGA SALES VASCONCELOS, PROFESSOR 3 GRAU**, em 25/09/2023, às 17:37, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **KAROLINE DANTAS BRITO, PROFESSOR 3 GRAU**, em 25/09/2023, às 17:37, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **Thiago Gonçalves das Neves, Usuário Externo**, em 25/09/2023, às 17:41, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **ROMILDO PEREIRA BRITO, PROFESSOR 3 GRAU**, em 25/09/2023, às 17:43, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **MARICE PEREIRA DA SILVA, SECRETÁRIA**, em 26/09/2023, às 09:14, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **Gladson Euler Lima Júnior, Usuário Externo**, em 26/09/2023, às 12:00, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

A autenticidade deste documento pode ser conferida no site https://sei.ufpe.edu.br/autenticidade, informando o código verificador **3822877** e o código CRC **66569B75**.

Dedico essa tese primeiramente à Deus, e a minha família, em especial a meus pais, Maria do Socorro Barboza da Silva e Gladson Euler Lima da Silva.

AGRADECIMENTOS

Agradeço a Deus pela Sua infinita bondade em conceder-me, dia após dia, a possibilidade de crescer em todos os aspectos, mantendo-me firme nas atribulações e forte para superar os desafios impostos pela vida.

Aos meus pais, Maria do Socorro Barboza da Silva e Gladson Euler Lima da Silva, pela confiança ao longo dessa caminhada e que sempre serão meus exemplos de moral, respeito e dignidade.

À minha irmã Mylena Euler Barboza da Silva, por toda a ajuda nos momentos difíceis e por sempre acreditar no meu esforço e potencial.

À minha namorada Javanna Lacerda, por toda a paciência e suporte para superar esse e tantos outros desafios ao longo dos últimos anos.

Aos meus amigos e colegas da Legasys pelo companheirismo e suporte em todos os momentos, além de me proporcionarem momentos de alegria insubstituíveis.

Ao professor Dr. Luis Gonzaga Sales Vasconcelos por todo o apoio e conhecimento transmitido.

Aos professores Dra. Karoline Dantas Brito e Dr. Romildo Pereira Brito pelo suporte ao longo da construção desse trabalho e pela atenção e apoio ao longo da minha trajetória pessoal e profissional.

Aos professores Dr. Thiago Gonçalves das Neves e Dr. Arioston Araújo de Morais Júnior pelas valiosas contribuições ao longo da construção deste trabalho.

Aos professores da Unidade Acadêmica de Engenharia Química e do Programa de Pós-Graduação em Engenharia Química, pelos conhecimentos transmitidos ao longo da graduação e da pós-graduação.

A todos que contribuíram diretamente ou indiretamente para a realização deste trabalho.

À CAPES, pelo apoio financeiro.

"Demore o tempo que for para decidir o que você quer da vida, e depois que decidir não recue ante nenhum pretexto, porque o mundo tentará lhe dissuadir."

Friedrich Nietzsche

LIMA JÚNIOR, GLADSON EULER. **Sistema de controle data driven para colunas de destilação utilizando deep reinforcement learning**. 2023. 111 p. Tese (Doutorado em Engenharia Química) – Universidade Federal de Campina Grande, Paraíba, 2023.

RESUMO

O controle preciso da composição em colunas de destilação é essencial para garantir a qualidade do produto e o desempenho do processo. Contudo, os controladores feedback do tipo PID, que são comumente utilizados nestes processos, podem apresentar restrições, dependendo da malha de controle utilizada. Neste sentido, controles baseados em dados utilizando técnicas de *reinforcement learning* (RL) tem sido uma solução atrativa dada a capacidade de adaptação do algoritmo à diferentes condições de controle. Trabalhos recentes indicam um foco no desenvolvimento de estudos voltados para a performance dos algoritmos de RL negligenciando a robustez na modelagem do ambiente. Neste trabalho foi proposto o desenvolvimento de um algoritmo de treinamento dinâmico integrado utilizando Python e Aspen Plus Dynamics para avaliação de diferentes modelos de RL. A adoção do Aspen Plus Dynamics para treinamento e validação assegurou a complexidade, não-linearidade e aspecto transiente do processo de destilação. Para avaliação do algoritmo foram aplicadas as metodologias do Q-Learning e Deep Q-Network (DQN), acopladas a um controlador PID. A primeira etapa do trabalho consistiu na avaliação do Q-Learning, explorando duas estratégias: uma taxa de atuação de controle fixa de 0,5% e outra com taxa de atuação flexível entre 0,1% e 5%. Na segunda etapa, propôs-se substituir o Q-Learning pelo DQN, mantendo a melhor estratégia da taxa de manipulação. A partir da comunicação do tipo COM foi possível rastrear as variáveis observáveis no software Aspen Plus Dynamics e realizar alterações no modelo dinâmico a partir do Python ao longo de toda a simulação. Os resultados confirmam a limitação da malha de controle inferencial na manutenção da composição de isobutano na base da coluna para distúrbios na composição de propano na alimentação. A estratégia com Q-Learning e taxa de atuação variável apresentou aproximadamente o dobro de assertividade em relação à taxa fixa, ampliando as regiões de acerto. Por outro lado, a utilização do controle DQN permitiu manter a composição dentro da especificação em 96% dos cenários de teste avaliados, com um IAE 52,9 % menor em comparação com o Q-Learning com taxa de atuação variável. Neste sentido, a abordagem DQN mostrou-se capaz de lidar com um processo de alta dimensão e não linear de forma mais robusta.

Palavras-chave: Controle baseado em dados, Aspen Plus Dynamics, Deep Q-Network, Q-learning, algoritmo de treinamento genérico.

LIMA JÚNIOR, GLADSON EULER. **Data-driven control system for distillation columns using deep reinforcement learning**. 2023. 111 p. Thesis (PhD in Chemical Engineering) - Federal University of Campina Grande, Paraiba, 2023.

ABSTRACT

Precise control of composition in distillation columns is essential to ensure product quality and process performance. However, PID-type feedback controllers, commonly used in these processes, may have limitations depending on the control loop used. In this regard, data-based controls using *reinforcement learning* (RL) techniques have become an attractive solution due to the algorithm's adaptability to various control conditions. Recent research indicates a focus on the development of studies aimed at the performance of RL algorithms, often overlooking the robustness in environmental modeling. In this work, the development of an integrated dynamic training algorithm using Python and Aspen Plus Dynamics for evaluating different RL models was proposed. The adoption of Aspen Plus Dynamics for training and validation ensures the complexity, non-linearity, and transient nature of the distillation process. To evaluate the algorithm, the methodologies of Q-Learning and Deep Q-Network (DQN), coupled with a PID controller, were applied. The first stage of the work involved evaluating Q-Learning, exploring two strategies: a fixed control action rate of 0.5% and another with a flexible action rate between 0.1% and 5%. In the second stage, the proposal was to replace Q-Learning with DQN while retaining the best control action rate strategy. Through COM-type communication, it was possible to track the observable variables in the Aspen Plus Dynamics software and make changes to the dynamic model from Python throughout the simulation. The results confirm the limitations of the inferential control loop in maintaining the isobutane composition at the base of the column during disturbances in the propane composition in the feed. The strategy with Q-Learning and a variable action rate showed approximately twice the accuracy compared to the fixed rate, expanding the regions of correctness. On the other hand, the use of DQN control allowed maintaining the composition within specification in 96% of the evaluated test scenarios, with a 52.9% lower IAE compared to Q-Learning with a variable action rate. In this sense, the DQN approach has proven capable of handling a high-dimensional and nonlinear process more robustly.

Keywords: Data-driven control, Aspen Plus Dynamics, Deep Q-Network, Q-Learning, generic training algorithm.

LISTA DE FIGURAS

LISTA DE TABELAS

Sumário

Capítulo 1 – Introdução

1.1 Contextualização

A obtenção de um controle preciso da composição em colunas de destilação de alta pureza é um desafio crítico do ponto de vista econômico. Além da otimização do processo, a implementação de uma estrutura de controle adequada é fundamental para garantir o desempenho da coluna e, consequentemente, reduzir os custos associados e aumentar a viabilidade do processo (RAMOS et al., 2016).

Tradicionalmente, os controladores clássicos do tipo PID têm sido amplamente utilizados na indústria para garantir a pureza dos produtos em colunas de destilação. No entanto, devido à complexidade, não linearidade e natureza multivariável desses processos, desvios nas características do processo ou mudanças nos pontos de ajuste podem levar à deterioração do desempenho do controlador ao longo do tempo (TULSYAN; GARVIN; UNDEY, 2018).

Nesse contexto, estratégias de controle baseadas em dados (data driven), como o uso de algoritmos de *reinforcement learning* (RL), têm sido propostas como alternativas aos controladores regulatórios clássicos ou baseados em modelos. O RL se destaca por sua capacidade adaptativa em lidar com processos não lineares, permitindo a obtenção de modelos a partir de processos complexos sem a necessidade de um conjunto inicial de dados de treinamento (SELVI; PIGA; BEMPORAD, 2018).

Adicionalmente, o RL é baseado em uma política de identificação do ambiente, tornando a lei de controle auto-organizada e capaz de se adaptar com base em experiências anteriores (MENDEL e MCLAREN, 1970).

Considerando as limitações e as oportunidades apresentadas pelos estudos anteriores, este trabalho propõe o desenvolvimento de um algoritmo de treinamento dinâmico integrado utilizando Python o Aspen Plus Dynamics para avaliação de diferentes modelos de reinforcement learning.

A abordagem utilizando o Aspen Plus Dynamics garante a complexidade, não linearidade e aspecto transiente do sistema, permitindo a inclusão de variáveis importantes, como a abertura das válvulas do sump e condensador, além de uma avaliação mais robusta do controlador.

1.2 Objetivos

1.2.1 Geral

Construir um algoritmo integrado utilizando Python e Aspen Plus Dynamics para o treinamento de modelos de *reinforcement learning* em simulações de processos dinâmicos.

1.2.2 Específicos

- Avaliar as principais variáveis para compor o conjunto de variáveis observáveis do algoritmo de *reinforcement learning* (RL) utilizando como caso de estudo uma coluna de destilação.
- Desenvolver o algoritmo de comunicação entre Python e Aspen Plus Dynamics para treinamento dos modelos de RL.
- Implementar o algoritmo Q-*learning* utilizando a Q-table como técnica inicial para correção de offsets avaliando as estratégias de taxa de manipulação fixa e variável.
- Implementar o algoritmo DQN com taxa de manipulação variável.
- Avaliar o desempenho nas etapas de treinamento e teste dos diferentes algoritmos de DDC por meio de distúrbios na composição da alimentação.
- Realizar uma avaliação dinâmica entre o comportamento do controlador PID convencional e a utilização conjunta do PID com os algoritmos avaliados.

1.3 Contribuições deste trabalho

No âmbito dos trabalhos envolvendo *reinforcement learning* este estudo visa contribuir com a literatura existente a partir de dois pontos fundamentais. O primeiro refere-se à utilização de um modelo de coluna destilação dinâmico construído no software *Aspen Plus Dynamics*.

De uma forma geral, estudos envolvendo RL para controle em colunas de destilação modelam o processo (ambiente) utilizando três abordagens: 1) Modelos linearizados a partir de funções de transferências (MIMO) simplificadas (KRETCHMAR et al., 2001; SPIELBERG et al., 2019); 2) Equações matemáticas que avaliam as relações entre as variáveis observáveis do sistema (PATEL, 2023); 3) Simuladores de processos químicos robustos como o Aspen Plus, em regime estacionário (HWANGBO; SIN, 2020).

A utilização das abordagens anteriores, focam na avaliação do algoritmo de *reinforcement learning*, simplificando o ambiente de estudo e negligenciando variáveis importantes na dinâmica de controle. Neste sentido, a utilização de um modelo de coluna de destilação simulado *Aspen Plus Dynamics* garante a complexidade, não-linearidade e aspecto transiente do processo.

O segundo ponto está relacionado ao algoritmo de comunicação construído. A tese propõe um procedimento para treinamento de modelos de *reinforcement learning* voltado para ambientes dinâmicos que pode ser adaptado para diferentes processos e softwares de análise dinâmica.

1.4 Organização do Trabalho

O capítulo 2 apresenta a revisão bibliográfica da tese, na qual é abordada uma visão geral histórica do reinforcement learning, seguida das aplicações iniciais e mais relevantes, como jogos e robótica. Em seguida, a revisão apresenta os trabalhos desenvolvidos na área de controle de processos, convergindo para os trabalhos realizados em colunas de destilação.

O capítulo 3 aborda os principais conceitos necessários para o entendimento do trabalho, incluindo as estratégias básicas de controle em colunas de destilação, redes neurais e reinforcement learning. Para o reinforcement learning, é realizado um aprofundamento no equacionamento, desde o processo de decisão de Markov até o surgimento do Q-learning e DQN.

O capítulo 4 apresenta o primeiro artigo submetido para a revista *Modelling and Simulation in Engineering*. O trabalho concentra-se na criação de um sistema de controle baseado em dados (DDC) que utiliza Q-Learning acoplado a um controlador PID para correção de desvios na composição do produto em colunas de destilação, causados por uma malha de controle inferencial utilizando o PID convencional.

O capítulo 5 contém o segundo artigo pronto para submissão. O trabalho busca corrigir as limitações relacionadas à utilização da Q-learning em conjunto com o controlador feedback do tipo PID. Nesse sentido, o capítulo apresenta uma comparação entre as estratégias utilizando Q-Learning e o Deep Q-Network (DQN).

O capítulo 6 apresenta a conclusão do trabalho a partir dos resultados obtidos, enquanto o capítulo 7 propõe linhas de pesquisas que podem ser desenvolvidas posteriormente a partir do presente estudo.

O capítulo 8 contempla uma abordagem teórica de outros tipos de redes neurais que podem ser utilizadas.

Capítulo 2 – Revisão Bibliográfica

Este capítulo apresenta a revisão bibliográfica referente ao *reinforcement learning*, com o propósito de contribuir com os fundamentos para o qual essa pesquisa se propõe. Desta forma, é apresentado desde uma visão geral e histórica até as aplicações na engenharia química, mais especificamente em colunas de destilação.

2.1 Visão Geral

O *reinforcement learning* é considerado um dos três paradigmas básicos do aprendizado de máquina, juntamente com o supervised learning e o unsupervised learning. No RL, o agente tem o objetivo de aprender a melhor maneira de executar uma tarefa por meio de interações repetidas com um ambiente, avaliando o valor de longo prazo das ações realizadas.

Em contraste, no supervised learning, o agente aprende um mapeamento a partir de dados de entrada e saída previamente conhecidos, permitindo prever valores de saída para novos valores de entrada. No unsupervised learning, o agente recebe dados não rotulados e aprende a distribuição probabilística desses dados (SHIN et al., 2019).

A história do *reinforcement learning* como técnica de inteligência artificial, mais especificamente no campo do aprendizado de máquina, surgiu a partir de estudos relacionados à psicologia do aprendizado animal e à teoria do controle ótimo. A consolidação do *reinforcement learning* ocorreu com a unificação dessas linhas de pesquisa, através do trabalho de Watkins (1992) e a criação do algoritmo Q-learning.

Nesse contexto, a psicologia da aprendizagem contribuiu com conceitos relacionados ao aprendizado de uma tarefa por meio de tentativa e erro, fundamentada em trabalhos como o de Thorndike (1911). Por outro lado, a teoria do controle ótimo, com o uso de funções de valor e programação dinâmica, ofereceu técnicas para melhorar o desempenho da tarefa, além de estruturar formulações que indicam a conclusão e o sucesso da tarefa realizada (SUTTON e BARTO, 2018).

Em 1957, Bellman introduziu conceitos do processo de decisão de Markov para definir a interação entre um agente de aprendizado e seu ambiente, considerando políticas, ações e recompensas (SUTTON e BARTO, 2018). A solução ótima para um problema de *reinforcement learning* refere-se à política que gera a maior recompensa ao longo de uma trajetória (NIAN; LIU e HUANG, 2020).

Nesse sentido, diversos métodos de aprendizado têm sido desenvolvidos para solucionar esses problemas, destacando-se três famílias de algoritmos: i) programação dinâmica; ii) Monte Carlo; e iii) diferença temporal.

Os métodos de programação dinâmica fornecem soluções exatas para a política ideal, assumindo a disponibilidade de um modelo perfeito do sistema. No entanto, o custo computacional é geralmente alto para problemas não triviais. Por essa razão, muitas aplicações têm utilizado métodos como Monte Carlo (MICHIE e CHAMBERS, 1968; ALLEN, ROYCHOWDHURY e LIU, 2018) e diferença temporal (SUTTON, 1988; TESAURO, 1992; CARTA, 2021), que não assumem a presença de um modelo perfeito e buscam soluções aproximadas.

O primeiro algoritmo de *reinforcement learning* que combina estados e ações contínuas foi desenvolvido por Silver et al. (2014) e é chamado de gradiente de política determinística (DPG, do inglês: deterministic policy gradient). O algoritmo DPG usa métodos de Monte Carlo para mapear deterministicamente estados contínuos em ações contínuas por meio de uma rede neural. No entanto, o custo computacional desse algoritmo é alto.

Os algoritmos baseados em diferença temporal (DT) são os mais aplicáveis, pois não requerem um modelo do sistema e o aprendizado ocorre a partir de interações dinâmicas. Além disso, diferentemente do método de Monte Carlo, esses algoritmos são capazes de ajustar as previsões finais nos estágios posteriores de iteração, antes que o resultado seja conhecido.

Os dois métodos de diferença temporal mais comuns diferem principalmente na avaliação do agente: i) Q-learning (WATKINS, 1992), que avalia todas as ações possíveis no momento atual para decidir qual ação maximiza a recompensa na próxima etapa; e ii) SARSA (RUMMERY; NIRANJAN, 1994), que aprende as políticas e avalia as consequências das ações tomadas no momento atual.

O interesse da comunidade científica se intensificou após o trabalho de MNIH et al. (2013), que propôs combinar o uso de *Q-learning* com redes neurais profundas (DQN, do inglês: *deep Q-learning network*).

Lillicrap et al. (2016) propôs associar as ideias de Mnih et al. (2013) e Silver et al. (2014) em um algoritmo chamado gradiente de política determinística profunda (DDPG, do inglês: *deep deterministic policy gradient*), que mapeia estados e ações contínuos. Nesse caso, o DPG foi usado para mapear estados em ações e o DQN foi usado para identificar os valores de ação para atualizar o DPG sem usar métodos de Monte Carlo. Essa estratégia reduziu tanto o enviesamento quanto a variância das experiências, além de reduzir o custo computacional do

treinamento substituindo o método de Monte Carlo por *Q-learning* (NIAN; LIU; HUANG, 2020).

2.2 Aplicações

As aplicações envolvendo os métodos de *reinforcement learning* têm atraído atenção da comunidade científica e da indústria para lidar com tarefas desafiadoras de tomada de decisão. Ao longo do desenvolvimento do RL, diversos domínios de aplicação foram explorados, sendo os jogos um dos mais relevantes.

Dentre os primeiros algoritmos desenvolvidos para este campo destaca-se o sistema para jogo de damas proposto por Samuel (1959, 1967), que aprende uma função de valor representada por um aproximador de função linear e emprega um esquema de treinamento usando *Q-learning*.

Nos anos subsequentes, os métodos de *reinforcement learning* foram amplamente aplicados em jogos, principalmente, naqueles conhecidos por ter uma alta complexidade como o gamão (TESAURO, 1992, 1994, 1995, 2002) e Go (SCHRAUDOLPH; DAYAN; SEJNOWSKI, 1994; SILVER et al., 2016, 2018).

O primeiro programa de inteligência artificial a derrotar humanos no Go foi o AlphaGo proposto por Silver et al. (2016). Posteriormente, Silver et al. (2018) propuseram o AlphaZero, um sistema que substitui heurísticas feitas à mão por uma rede neural profunda e algoritmos que não recebem nada além das regras básicas do jogo, desenvolvendo, assim, seu próprio estilo de jogo de Go, xadrez e shogi.

Recentemente, Schrittwieser et al. (2020) desenvolveram o MuZero, um passo significativo na busca de algoritmos de uso geral, que domina Go, xadrez, shogi e Atari sem a necessidade de saber as regras, graças à sua capacidade de planejar estratégias de vitória em ambientes desconhecidos.

As aplicações dos métodos de *reinforcement learning* se estendem a áreas multidisciplinares como robótica, mercado financeiro e engenharia.

A aplicação do RL no âmbito da robótica, tem, em geral, o objetivo de permitir que um robô descubra de forma autônoma um comportamento ideal por meio de interações de tentativa e erro com seu ambiente (KOBER; BAGNELL; PETERS, 2013).

Embora a manipulação de robôs seja uma aplicação particularmente desafiadora, enfrentando problemas sobretudo com a geração de funções de recompensa apropriadas para aprender boas políticas, desempenhos promissores têm sido obtidos em tarefas como,

empilhamento (NAIR et al., 2018) e manipulação de mão robótica (ANDRYCHOWICZ et al., 2020).

No contexto do mercado financeiro, os algoritmos de *reinforcement learning*, em geral, tratam a negociação no mercado externo, de câmbio ou de ações, como um problema de decisão markoviana, e foi introduzida pela primeira vez por Neuneier (1996).

Fischer (2018) conduziu uma pesquisa sobre as principais abordagens de modelos nesta área, que incluem trabalhos como DENG, 2016; PENDHARKAR e CUSATIS, 2018.

Entre as contribuições mais recentes, Carta et al. (2021) propôs usar o *reinforcement learning* para maximizar a função de retorno usando um agente *Q-learning* e investigou o comportamento do modelo em importantes mercados de ações do mundo real. Dessa forma, os resultados alcançados na negociação intradiária indicaram melhor desempenho do que a estratégia convencional de *Buy-and-Hold.*

O termo *reinforcement learning* foi usado pela primeira vez por pesquisadores da área da engenharia para aplicar tal técnica em sistemas de controle na década de 1960. Mendel e Mclaren (1970) descreveram que, uma classe de sistema de controle usando *reinforcement learning* é aquela cuja planta ou o ambiente podem não ser conhecidos a priori, nesse caso, a lei de controle é auto-organizada; ou seja, é capaz de mudar em função de sua experiência.

Por razões de direcionamento do conhecimento, a próxima seção abordará o *reinforcement learning* no contexto da engenharia química, especificamente, na aplicação dessas técnicas em processos industriais.

2.3 Aplicações em Processos Industriais

No contexto do controle de processos industriais, as estratégias convencionais de controle costumam incluir controladores PID e controle adaptativo, os quais são projetados para atender a especificações sob condições conhecidas de ambiente e processo. No entanto, variações nas características do processo ou mudanças deliberadas nas condições de ajuste podem afetar o desempenho do sistema, mesmo quando um controlador bem ajustado é instalado inicialmente.

De acordo com Spielberg et al. (2019), os controladores industriais modernos exigem um modelo de processo de alta qualidade. Isso significa que, em casos de perda de desempenho, pode ser necessário reidentificar o modelo, o que demanda tempo, interrompe a operação normal do processo e pode ser particularmente complicado para processos multivariáveis. Diante dessas limitações, a comunidade científica tem explorado como as técnicas de *reinforcement learning* podem contribuir para a solução desses problemas.

Embora as ideias de *reinforcement learning* sejam aplicadas há algumas décadas em diversas áreas do conhecimento, sua aplicação no controle de processos industriais emergiu apenas nas últimas duas décadas. Alguns dos primeiros trabalhos que se concentraram na aplicação do *reinforcement learning* para tais problemas foram os estudos de Hoskins e Himmelblau (1988, 1992).

Nesses estudos, foi proposta uma arquitetura de redes neurais artificiais para o controle de um reator CSTR, comparando sua performance com a de um controlador PID sintonizado. Os autores destacaram uma importante relação entre o tempo necessário para alcançar o setpoint e a quantidade de iterações no processo de aprendizagem. Em outras palavras, o tempo de atraso poderia ser reduzido para que a trajetória se aproximasse do comportamento do controlador PID, mas isso exigiria um tempo maior para aprender a estratégia de controle do processo

Além disso, o *reinforcement learning* também tem sido aplicado em outros cenários como: a) controle de composição em uma coluna de destilação (MARGAGLIO et al., 1997); b) modelagem e otimização de reatores em batelada (WILSON e MARTINEZ, 1997; MARTINEZ, 2000); c) otimização da operação do processo de uma coluna de destilação em batelada comparando modelos preditivos de função linear, polinômio de segunda ordem, rede neural e modelo preditivo híbrido generalizado (MUSTAFA e WILSON, 2012); d) controle de termostato considerando ações de várias etapas e *Q-learning* (SCHOKNECHT; RIEDMILLER, 2003); e) controle de pH usando o método de controle de aprendizagem livre de modelos (SYAFIIE, TADEO e MARTINEZ, 2007, 2008); f) controle baseado em *Q-Learning* para determinação do fluxo de alimentação em um processo de fermentação de levedura (CHUO et al., 2013).

As técnicas de *reinforcement learning* aplicadas nessas áreas têm atraído o interesse da comunidade científica. No caso do controle de reatores em batelada, algumas das abordagens mais recentes incluem o trabalho de Zhang et al. (2019), que aplicou um algoritmo de Q-learning de ação de múltiplas etapas modificado; o estudo de Singh e Kodamana (2020), que concluiu que um controlador baseado em DQN alcançou o ponto de ajuste mais rapidamente do que um controlador baseado em Q-learning; e a pesquisa de Yoo et al. (2021), que demonstrou melhorias no controle utilizando DDPG modificado com a aprendizagem Monte-Carlo.

Goulart e Pereira (2020) desenvolveram um controlador autônomo de pH para efluentes líquidos na indústria de galvanoplastia, baseado em reinforcement learning. Nesse caso, um simulador de neutralização de reator de tanque agitado (CSTR) e um algoritmo de otimização de enxame de partículas foram adaptados para automatizar a escolha dos hiperparâmetros. A

validação do controlador foi realizada ao estabilizar o pH do efluente em uma faixa neutra em diferentes cenários durante as operações regulatórias, apresentando resultados superiores quando comparados a um controlador PID.

Machalek, Quah e Powell (2020) também investigaram algoritmos de *reinforcement learning* no ambiente de um CSTR, incluindo DDPG, DDPG duplo-atrasado (ou TD3) e otimização de política proximal. Embora os três algoritmos tenham demonstrado um desempenho satisfatório, apenas o TD3 demonstrou convergência para uma solução estável.

Hwangbo e Sin (2020) propuseram um modelo DQN para sistemas de controle de uma coluna de extração líquido-líquido em processos biofarmacêuticos. Essa abordagem resultou em um desempenho de operação com um melhor rendimento de recuperação de ativos (32% maior do que a operação em malha aberta) e desvios mais baixos (23% menor do que a operação em malha aberta) contra distúrbios.

Deng et al. (2022) propuseram uma abordagem inovadora para o controle de laminação de tiras utilizando *reinforcement learning*. Com base na otimização de política proximal (PPO), os autores apresentaram um PPO multi-ator., onde atores interagiram com o ambiente em paralelo e apenas a experiência do ator com a maior recompensa foi utilizada para atualizar os demais atores. O controle de laminação de tiras é um processo complexo que exige abordagens de controle baseadas em modelos matemáticos de primeiros princípios ou empíricos. Os resultados da simulação mostraram que o método proposto superou os métodos de controle convencionais e outras abordagens de RL avançadas, como DDPG e TD3 em termos de capacidade de processo e suavidade.

Dutta e Upreti (2023) apresentaram uma estratégia de controle de processo utilizando redes neurais artificiais múltiplas (MNN) e *reinforcement learning* para gerar um controlador MNNRL. Os estudos de caso realizados em processos contínuos (reatores) com diferentes níveis de não-linearidade demonstraram que o controlador MNNRL foi capaz de fornecer um rastreamento aprimorado do setpoint e suprimiu overshoots e undershoots nas variáveis controladas. O controlador MNNRL mostrou um desempenho superior em relação ao MNN e ao controle preditivo de modelo não linear (NMPC), com cerca de 32% menos erro absoluto integral (IAE) e tempo de estabilização reduzido em 50%, apresentando-se como uma abordagem eficaz para o controle de processos não lineares.

Liu, Tsai e Chen (2023) desenvolveram um controlador baseado em *deep reinforcement learning* (DRL) integrado a um modelo substituto construído com o uso de uma rede neural do tipo sequence-to-sequence e dados históricos de um processo industrial Claus. Os resultados mostraram que o controlador baseado em DRL proposto reduziu em até 55% o desvio padrão

da variável de controle (razão $\frac{H_2S}{SO_2}$ no gás residual), em comparação com a estratégia de controle padrão. Adicionalmente, os controladores DRL mantiveram consistentemente a média da razão $\frac{H_2S}{SO_2}$ em diferentes modos de operação, além de reduzir efetivamente as variações na distribuição dos dados.

Alhazmi e Sarathy (2023) utilizaram redes neurais e RL para propor uma estratégia para a estimação online de parâmetros não lineares de modelos físicos. O RL foi aplicado para aprender a política de estimação de parâmetros representada por uma rede neural. O algoritmo foi testado em uma simulação de hidrogenação seletiva de acetileno, um sistema altamente não linear simulado por um modelo pseudo-homogêneo unidimensional. A política de estimação de parâmetros aprendida foi capaz de prever corretamente os estados do sistema com um erro de previsão inferior a 1% em várias condições, incluindo ruído de medição e diferenças estruturais nos modelos. Adicionalmente os autores sugerem que o método de estimação de parâmetros baseado em RL pode ser combinado com métodos de controle adaptativo para alcançar um desempenho de controle adaptativo ótimo para sistemas com dinâmicas variáveis.

2.4 Aplicações em Colunas de Destilação

As colunas de destilação são utilizadas na indústria química e petrolífera para separar líquidos usando suas propriedades físico-químicas. Essas unidades são amplamente empregadas pelas empresas do setor petroquímico (NIZAMI, 2011).

Dentro do contexto do reinforcement learning, Glorennec (1994) foi um dos pioneiros a propor a versão fuzzy do Q-learning, conhecida como Fuzzy Q-Learning (FQL) e Dynamic Fuzzy Q-Learning (DFQL). Glorennec e Jouffe (1997) propuseram uma adaptação do Q-learning para sistemas de inferência fuzzy, onde as ações são inferidas a partir de regras fuzzy. Eles compararam essa técnica de aprendizado com a abordagem do algoritmo genético e demonstraram que o Q-learning é mais eficaz na abordagem proposta.

Margaglio et al. (1997) propuseram o uso de sistemas de inferência fuzzy para controlar as composições dos produtos em uma coluna de destilação binária, utilizando o Q-learning para ajustar as saídas fusificadas. No entanto, os autores identificaram como desvantagem desse método a independência dos controladores, além de não considerarem a interferência e o acoplamento das variáveis.

Recentemente, Spielberg et al. (2019) demonstraram a eficácia de um controlador DQN ao realizar o rastreamento bem-sucedido dos setpoints em um sistema MIMO. Nesse caso, o controlador DQN foi projetado para rastrear dois produtos em uma coluna de destilação,

utilizando um modelo linearizado simplificado mal condicionado. Embora o sistema possuísse múltiplas variáveis manipuladas, os autores simplificaram considerando apenas duas entradas: carga térmica e razão de refluxo.

Wang e Ge (2020) apresentaram um esquema baseado em *Q-Learning* para controle do nível de líquido, pressão e temperatura em uma coluna de destilação DMF modelada empiricamente, a partir do fluxo de entrada e saída.

Oh et al. (2021) utilizaram *reinforcement learning* para a otimização de um processo de hidrocraqueamento (HCR). A estratégia consistiu em três partes principais: modelagem matemática rigorosa do HCR (equações de conservação de massa e energia), construção de um modelo substituto de redes neurais profundas (DNN) para o desenvolvimento do ambiente de RL e formulação do algoritmo de RL. O modelo matemático foi validado e o modelo substituto DNN mostrou-se preciso. O algoritmo de RL A2C foi treinado para determinar as condições de operação ótimas para objetivos de rendimento de produto. Os resultados mostraram que a estratégia de otimização de RL foi capaz de determinar com precisão as condições operacionais ótimas para o processo de HCR.

Patel (2023) apresentou um método sistemático para o controle contínuo de processos industriais usando RL, buscando uma implementação segura, rápida e explicável. Em vez de propor um novo algoritmo de RL, o estudo se concentrou em definir estados, ações e recompensas, além de modificações para algoritmos existentes que suportam estados e ações contínuas. A abordagem foi aplicada nos processos de separação de ciclohexano e n-heptano em colunas de destilação e controle de temperatura em um setup laboratorial, demonstrando sua eficácia em ambientes multivariáveis, ruidosos e não lineares, evitando operações inseguras.

Capítulo 3 - Fundamentação teórica

Este capítulo apresentará os conceitos fundamentais para o entendimento das principais técnicas utilizadas neste trabalho. Adicionalmente, outros tópicos relacionados às redes neurais estão disponíveis no Apêndice A (8.1.1).

3.1 Coluna de Destilação

A destilação é uma operação unitária que continua sendo o principal método de separação em plantas de processamento industrial, apesar de sua inerente baixa eficiência termodinâmica (KISTER, 1992)

Nesse processo, uma mistura de alimentação composta por dois ou mais componentes é separada em dois ou mais produtos, geralmente limitados a um destilado de topo e um produto de fundo, ambos com composições diferentes da mistura de alimentação (SEADER; HENLEY; ROPER, 2010). A eficácia desse processo de separação depende de dois fatores principais:

1)A criação de uma segunda fase, permitindo que líquido e vapor coexistam e entrem em contato à medida que fluem em direções opostas em uma coluna.

2)A diferença nas volatilidades dos componentes, que determina como eles se distribuem entre as fases, levando à separação desejada.

Os componentes mais voláteis, com pontos de ebulição mais baixos, são retirados como destilado no topo da coluna, enquanto os componentes menos voláteis, de pontos de ebulição mais altos, são obtidos como produto de fundo (LUYBEN, 2020).

No entanto, a destilação é uma técnica energeticamente intensiva, especialmente quando a volatilidade relativa (α) dos componentes-chave a serem separados é baixa, ou seja, quando suas diferenças de volatilidade são pequenas.

Devido ao alto consumo de energia envolvido a implementação de um controle eficiente é crucial para aumentar a produtividade e, consequentemente, melhorar a lucratividade do processo (MISHRA; KUMAR; RANA, 2015). De acordo com Skogestad (1992) a destilação é provavelmente a operação unitária mais estudada em termos de controle.

3.1.1 Controle em Colunas de Destilação

O comportamento dinâmico, não linear e o elevado grau de acoplamento tornam o controle de colunas de destilação um problema de resolução complexa. Nesse sentido, estratégias de controle convencionais, como controladores feedback do tipo P, PI e PID, podem ter sua performance comprometida, dependendo das perturbações do processo (LUYBEN, 1990; MARLIN, 1995; STEPHANOPOULOS, 1984).

Kumar (2011) elencou três estratégias convencionais de controle utilizadas para o controle de composição em colunas de destilação, que são as seguintes:

1. Controle através da manutenção da temperatura do prato sensível (Figura 3-1): após identificar o estágio sensível da coluna por meio da avaliação do ganho em estado estacionário, utiliza-se um controlador para manter a temperatura do prato em um valor que resulte na composição especificada.
2. Controle direto de composição (Figura 3-2): o controlador feedback convencional é utilizado para controlar a composição de topo da coluna a partir da manipulação da carga térmica do reboiler (Q).
3. Controle cascata (Figura 3-3): nesta estratégia de controle utiliza-se o controlador de temperatura do prato sensível (estratégia 1) como controlador secundário. O *setpoint* do controlador secundário é definido a partir do sinal de saída do controlador de composição (controlador primário).

As três estratégias mencionadas requerem uma prévia sintonia dos parâmetros dos controladores. No entanto, em certos casos, a magnitude das perturbações pode levar o sistema a uma região na qual os ganhos (proporcional, integral e derivativo) do controlador não são suficientes para manter a composição no setpoint, resultando em instabilidades e/ou ocorrência de offsets.

Nesse contexto, estratégias de controle que utilizam técnicas de inteligência artificial têm se mostrado uma solução viável para contornar essas limitações.

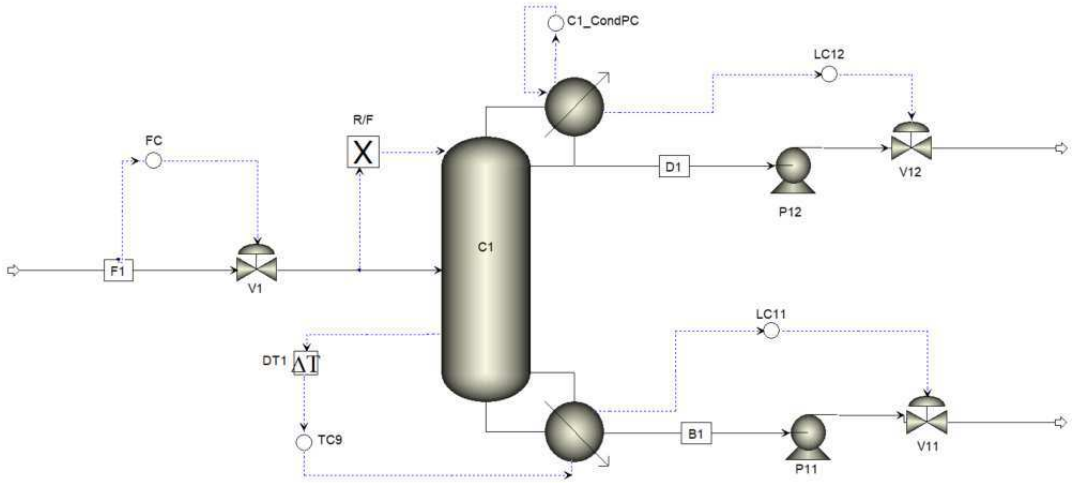


Figura 3-1-Fluxograma para a estratégia de controle 1.

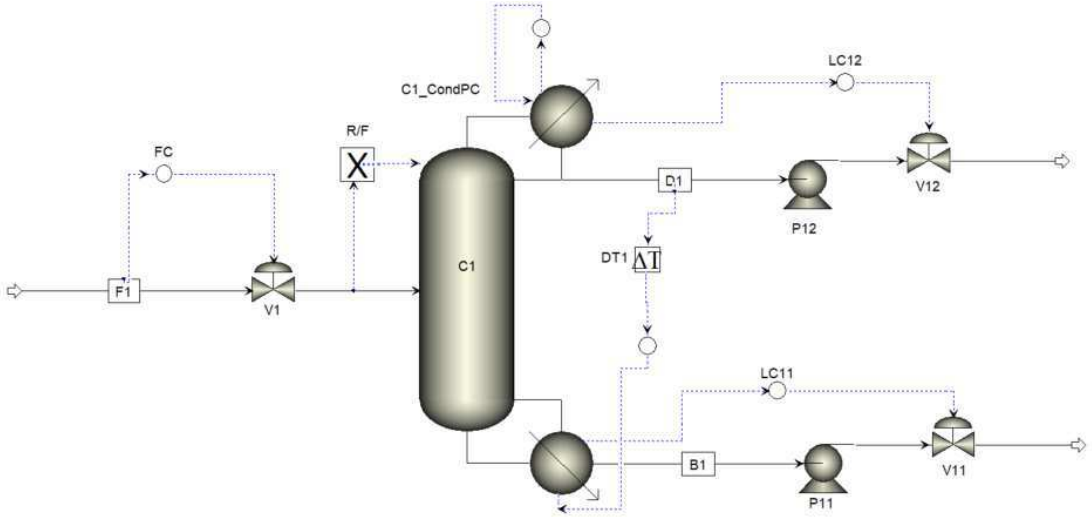


Figura 3-2- Fluxograma para a estratégia de controle 2.

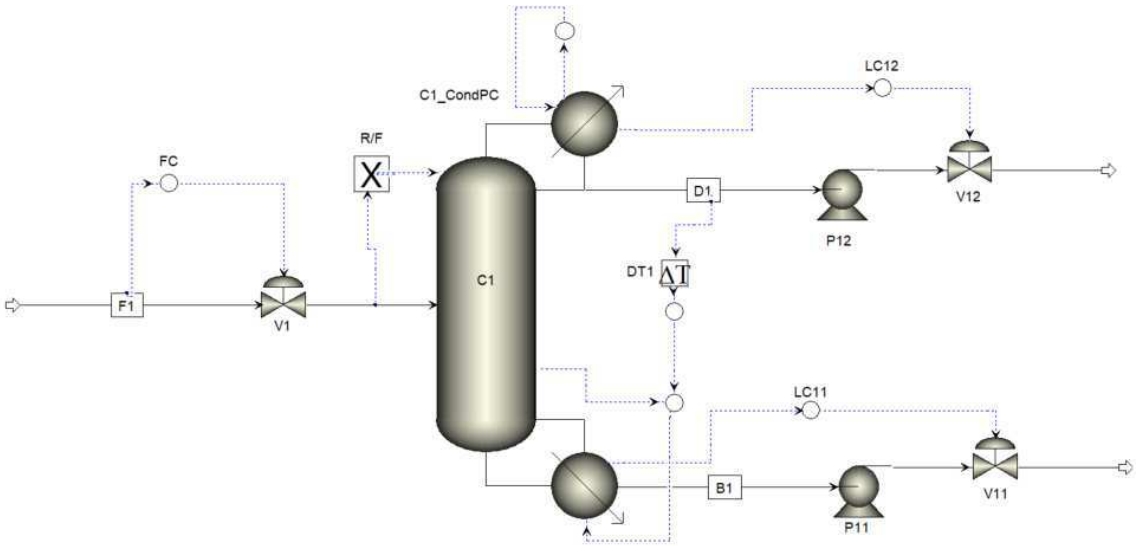


Figura 3-3- Fluxograma para a estratégia de controle 3.

3.2 Redes neurais

3.2.1 Definição Geral

Em 1943, o matemático Watter Pitts e o neurofisiologista Warren McCulloch, modelaram a primeira rede neural conhecida utilizando circuitos elétricos visando mostrar o funcionamento de um neurônio humano (MCCULLOCH e PITTS, 1943). A partir deste trabalho, diversos avanços foram realizados na vertente do entendimento do funcionamento do neurônio bem como na aplicabilidade das redes neurais no âmbito da inteligência artificial.

De forma mais relevante, destaca-se Rosenblatt (1957), com a criação do Perceptron para o reconhecimento de padrões baseado em uma rede neural computacional de duas camadas. Além de Rosenblatt, Bernard Widrow e Marcian Hoff (1960) destacam-se pelo desenvolvimento dos modelos ADALINE e MADALINE. Esses modelos foram nomeados devido ao uso de ***M**ultiple **ADA**ptive **LIN**ear **E**lements* sendo o MADALINE a primeira rede neural a ser aplicada a um problema do mundo real como um filtro adaptativo que elimina ecos nas linhas telefônicas, sendo este utilizado até os dias atuais.

A estrutura básica da uma rede neural pode ser descrita sendo uma generalização do modelo desenvolvido por McCulloch e Pitts, conforme apresentado na Figura 3-4.

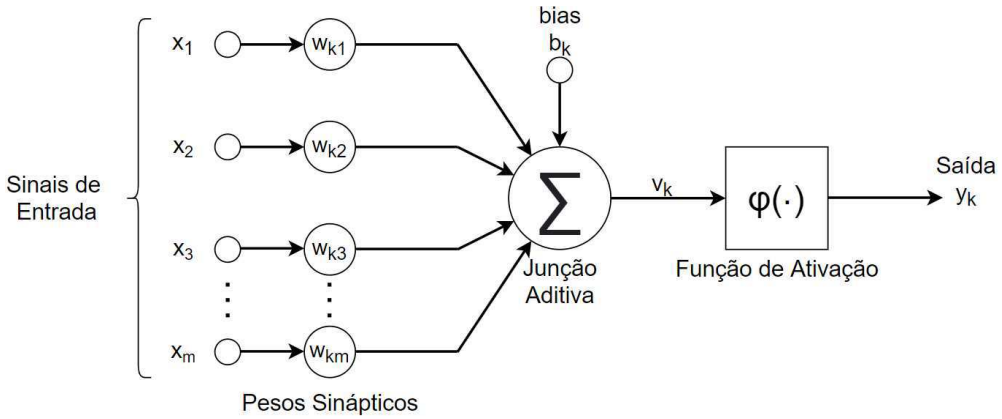


Figura 3-4 - Estrutura básica de uma rede neural padrão
Fonte: Fonte: Adaptado de HAYKIN, 2001

Onde:

- w_{k1}-w_{km}: Pesos sinápticos atrelados ao neurônio k que ponderam os sinais das entradas x_1 - x_m.
- Junção aditiva: Possui a função de somar os sinais de entrada ponderados pelos pesos sinápticos.

- Bias (b_k): Parâmetro que introduz uma compensação b_k ou deslocamento para as saídas dos neurônios, permitindo que a rede aprenda e generalize melhor a partir dos dados de treinamento.
- Função de ativação$\varphi(.)$: Modela a forma como o neurônio responde a excitação, introduzindo não-linearidades no modelo, permitindo que a rede neural aprenda e modele relações complexas nos dados.

O modelo apresentado na Figura 3-4 pode ser descrito matematicamente de acordo com a Eq. (3-1).

$$y_k = \varphi(v_k); Onde\ v_k = \sum_{j=1}^{m} w_{kj} x_j + b_k \tag{3-1}$$

3.2.2 Funções de ativação

Conforme descrito previamente, as funções de ativação têm como objetivo modelar a resposta de um neurônio à excitação, restringindo a amplitude do sinal de saída. As principais funções de ativação, de acordo com Sharma et al. (2020), estão listadas a seguir:

1. Função degrau binária
2. Linear
3. Sigmóide
4. Tangente hiperbólica
5. ReLU (Rectified Linear Unit)
6. Leaky ReLU
7. Parametrized ReLU
8. Unidade Linear Exponencial
9. Swish
10. SoftMax

As funções de ativação mencionadas possuem vantagens e desvantagens, dependendo do sistema que está sendo representado. No entanto, algumas observações gerais são relevantes:

- Para problemas de classificação, uma combinação de funções sigmoides tende a produzir melhores resultados.

- Devido a problemas relacionados à diferenciabilidade, funções sigmoides e tangente hiperbólica costumam ser evitadas em problemas de regressão.
- A função de ativação ReLU é amplamente utilizada e geralmente apresenta um desempenho superior às outras funções de ativação em termos de correlação. No entanto, essa função de ativação deve ser empregada apenas na camada oculta da rede.

Uma das principais características de uma função de ativação é que ela deve ser preferencialmente diferenciável. Isso está relacionado à estratégia de retropropagação do erro e aos métodos de otimização dos pesos sinápticos, que geralmente usam algoritmos baseados em derivadas, como o gradiente descendente, para minimizar os erros durante o treinamento.

Neste sentido, os tópicos 3.2.2.1 e 3.2.2.2 fornecem uma breve explanação sobre as funções de ativação ReLU e Leaky ReLU, pois são diferenciáveis em todo o domínio. A descrição das demais funções de ativação é apresentada no trabalho de Sharma et al. (2020).

3.2.2.1 Função ReLU

A unidade linear retificada (ReLU) descrita pela Eq.(3-2) é uma função de ativação não linear amplamente utilizada na criação de redes neurais. A principal vantagem na utilização da função ReLU é que apenas parte dos neurônios são ativados por vez, ou seja, caso a entrada da função de ativação seja negativa, ela é convertida em zero e o neurônio não é ativado, tornando a rede mais eficiente. A Figura 3-5 apresenta graficamente a função ReLU.

$$\varphi(x) = \max(0, x) \tag{3-2}$$

Em alguns casos (x<0), tem-se que o valor da derivada da função é igual a zero, podendo ocasionar problemas para a função ReLU, como por exemplo a não atualização dos pesos durante a etapa de retropropagação no treinamento. Visando contornar essa limitação, pode-se utilizar a função de ativação *Leaky* ReLU.

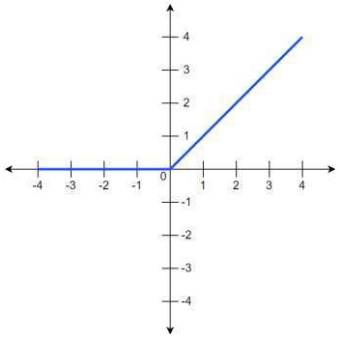


Figura 3-5 - Representação gráfica da função de ativação ReLU

3.2.2.2 Função *Leaky* ReLU

A função *Leaky* ReLU é uma modificação da função ReLU padrão. Nela, a resposta da função para valores de x < 0, é dada como um componente linear (α) extremamente pequeno de x. Matematicamente pode-se expressar esta função como apresentado na Eq.(3-3).

Observa-se, portanto, que enquanto a ReLU inativa os neurônios em regiões onde x<0, a *Leaky* ReLU permite que eles sejam ativados. Além disso, a derivada da Relu nessa região é igual a zero enquanto a *Leaky* ReLU possui derivada igual a α. A Figura 3-6 apresenta graficamente a função de ativação considerando o componente linear α = 0.01.

$$\begin{cases} \varphi(x) = \alpha \mathrm{x}\,, \mathrm{x} < 0 \\ \varphi(x) = \mathrm{x}\,, \mathrm{x} \geq 0 \end{cases} \tag{3-3}$$

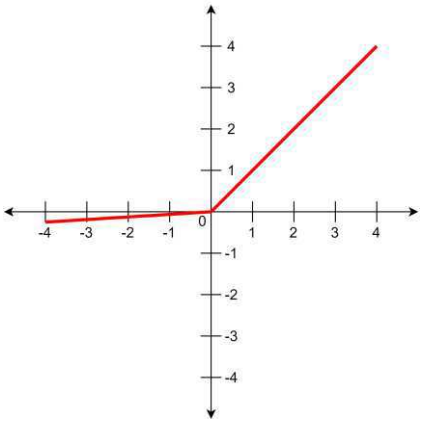


Figura 3-6 - Representação gráfica da função de ativação *Leaky* ReLU

3.2.3 Classificação e Arquitetura das Redes Neurais

Em geral, as redes neurais artificiais podem ser classificadas em feedforward, recorrentes e simetricamente conectadas. O tópico 3.2.4 apresenta a classificação feedforward utilizada neste trabalho. Os outros dois tipos de classificação, bem como as principais arquiteturas envolvidas, estão descritos no apêndice A.

3.2.4 Redes Neurais Feedforward (Multi Layer Perceptron)

As arquiteturas de redes classificadas como *feedforward* (FFNN) recebem esse nome devido ao fluxo de informação que ocorre sempre em uma única direção (entrada para saída) sem conexões entre os elementos de uma mesma camada.

As redes do tipo multi layer perceptron (MLP) possuem uma estruturação padrão composta pela camada de entrada, uma ou mais camadas intermediárias (ocultas) e a camada de saída, conforme apresentado na Figura 3-4 (ROSENBLATT, 1957). MLPs que possuem mais de uma camada oculta são chamadas de redes neurais profundas ou *Deep Learning*.

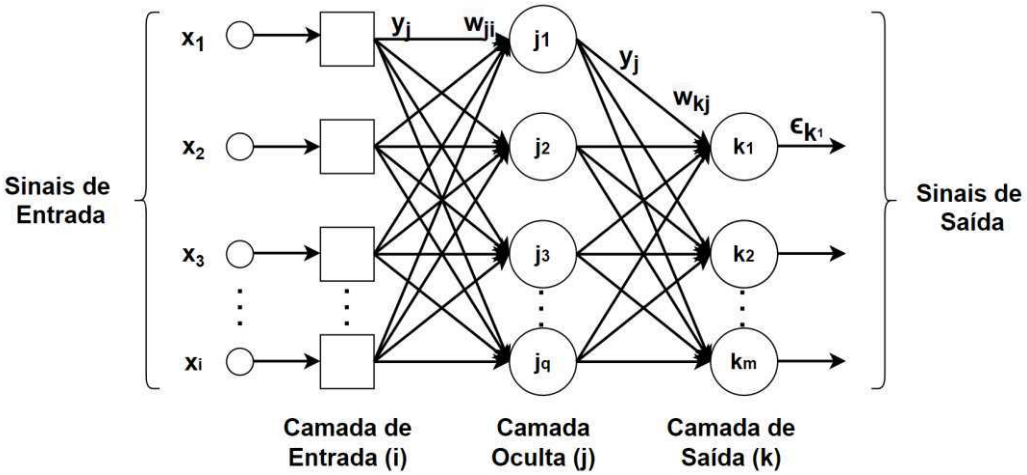


Figura 3-7 - Estruturação MLP com 1 camada oculta
Fonte: Adaptado de HAYKIN, 2001.

O algoritmo para treinamento de uma MLP envolve inicialmente um passo de propagação direta (forward propagation), seguido por um passo de retropropagação (backward propagation).

O sinal de erro na saída de um neurônio k da camada de saída, é calculada através da diferença entre a saída desejada $yd_k(n)$ e a resposta da rede $y_k(n)$. O cálculo do erro $\epsilon_k(n)$ é obtido através da Eq.(3-4) (RODRIGUES, 2019).

$$\epsilon_k(n) = yd_k(n) - y_k(n) \tag{3-4}$$

Conforme mostrado anteriormente, a saída do neurônio k é um valor real produzido através da função de ativação não linear, conforme apresentado na Eq.(3-5)

$$y_k(n) = \varphi_k\big(v_k(n)\big);\ Onde\ v_k(n) = \sum_{j=1}^{m} w_{kj} y_j(n) \tag{3-5}$$

Sendo o erro quadrático para o neurônio k definido por $\frac{1}{2}\left(\epsilon_k(n)\right)^2$, a soma dos termos de todos os neurônios da camada de saída dado pela Eq.(3-6) representa o erro quadrático total instantâneo para o padrão (n).

$$\varepsilon(n) = \frac{1}{2}\sum_{k=1}^{p}\left(\epsilon_k(n)\right)^2 \tag{3-6}$$

O ajuste do peso w_{kj} a ser realizado é proporcional a derivada parcial $\frac{\partial \varepsilon\,(n)}{\partial w_{kj}}$ em relação a cada peso da rede. De acordo com a regra da cadeia oriunda do cálculo diferencial, o gradiente pode ser obtido através da seguinte Eq.(3-7).

$$\frac{\partial \varepsilon\,(n)}{\partial w_{kj}} = \frac{\partial \varepsilon\,(n)}{\partial \epsilon_k(n)}\frac{\partial \epsilon_k(n)}{\partial y_k(n)}\frac{\partial y_k(n)}{\partial v_k(n)}\frac{\partial v_k(n)}{\partial w_k(n)} \tag{3-7}$$

Calculando cada termo separadamente, temos:

$\frac{\partial \varepsilon\,(n)}{\partial \epsilon_k(n)} = \epsilon_k(n)$; $\frac{\partial \epsilon_k(n)}{\partial y_k(n)} =$ - 1; $\frac{\partial y_k(n)}{\partial v_k(n)} = \varphi'_k\left(v_k\,(n)\right)$; $\frac{\partial v_k(n)}{\partial w_k(n)} = y_j(n)$

Substituindo o resultado de cada termo na Eq.(3-7), tem-se:

$$\frac{\partial \varepsilon\,(n)}{\partial w_{kj}} = -\epsilon_k(n)\varphi'_k\left(v_k\,(n)\right)y_j(n) \tag{3-8}$$

É importante destacar que $y_j(n)$ refere-se a saída oriunda do neurônio j que é a entrada do neurônio k da camada de saída. Os ajustes nos pesos de Δw_{kj} aplicado a w_{kj} é realizado em direção oposta ao gradiente através da Regra Delta. A constante η na Eq.(3-9) representa a taxa de aprendizagem, o sinal de negativo indica a descida do gradiente em relação ao espaço de pesos.

$$\Delta w_{kj} = -\,\eta\,\frac{\partial \varepsilon\,(n)}{\partial w_{kj}} \tag{3-9}$$

Substituindo a Eq.(3-8) na Eq.(3-9), tem-se:

$$\Delta w_{kj} = \eta\, \delta\,(n) y_j(n) \tag{3-10}$$

Neste caso, observa-se que a atualização dos pesos depende do sinal do erro $\epsilon_k(n)$ para o gradiente local. Na descrição realizada, o parâmetro $\delta\,(n)$ está relacionado com o neurônio k localizado na camada de saída. Para este caso, $\delta_k(n)$ é dado pela Eq. (3-11).

$$\delta_k(n) = -\frac{\partial \varepsilon\,(n)}{\partial \epsilon_k(n)} \frac{\partial \epsilon_k(n)}{\partial y_k(n)} \frac{\partial y_k(n)}{\partial v_k(n)} = \epsilon_k(n) \varphi'_k\big(v_k\,(n)\big) \tag{3-11}$$

Caso, o parâmetro $\delta\,(n)$ estivesse sendo calculado para o neurônio j da camada intermediária, o sinal do erro $\epsilon_j(n)$ seria calculado em termos dos neurônios em que j esteja conectado. Neste sentido, para calcular o gradiente local $\delta_j(n)$, a Eq.(3-11) pode ser reescrita da seguinte forma:

$$\delta_j(n) = -\frac{\partial \varepsilon\,(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} = -\frac{\partial \varepsilon\,(n)}{\partial y_j(n)} \varphi'_j\left(v_j\,(n)\right) \tag{3-12}$$

Aplicando os passos supracitados, considerando os pesos w_{kj} relacionados aos neurônios da camada de saída k ligada ao neurônio j, tem-se que $\frac{\partial \varepsilon\,(n)}{\partial y_j(n)}$ pode ser dado pela Eq.(3-13)

$$\frac{\partial \varepsilon\,(n)}{\partial y_j(n)} = -\sum_k \epsilon_k(n) \varphi'_k\big(v_k\,(n)\big) w_{kj}(n) \tag{3-13}$$

Substituindo a Eq.(3-11) na Eq.(3-13) e em seguida na Eq.(3-12), tem-se

$$\frac{\partial \varepsilon\,(n)}{\partial y_j(n)} = -\sum_k \delta_k(n)\; w_{kj}(n) \tag{3-14}$$

$$\delta_j(n) = \varphi'_j\left(v_j\,(n)\right) \sum_k \delta_k(n)\; w_{kj}(n) \tag{3-15}$$

Observa-se, portanto que enquanto $\delta_k(n)$ é impactado por $\epsilon_k(n)$, $\delta_j(n)$ é impactado pelo gradiente local $\delta_k(n)$ oriundo da camada de saída em decorrência do backpropagation do erro.

De forma resumida, o processo de backpropagation ocorre em duas fases distintas. Na primeira fase, chamada de fase forward, as informações fluem da camada de entrada até a

camada de saída sem que os pesos das camadas sejam alterados. Isso começa com a entrada de dados na camada de entrada e termina com o cálculo do erro na camada de saída.

A segunda fase, conhecida como fase backward, inicia na camada de saída e propaga-se através de todas as camadas intermediárias até chegar à camada de entrada. Durante essa fase, os pesos são ajustados pela primeira vez na camada de saída e, em seguida, são propagados recursivamente para ajustar os pesos de todas as camadas intermediárias, até finalmente chegar à camada de entrada.

3.3 Machine Learning

Machine Learning (ML) ou aprendizado de máquina é um ramo da Inteligência Artificial (IA) cujo objetivo é desenvolver técnicas computacionais para o aprendizado e construção de sistemas capazes de adquirir conhecimento automaticamente (MONARD, 2003). As técnicas relacionadas ao ML surgiram por volta de 1950, mas têm ganhado destaque recentemente devido à capacidade atual de processamento e armazenamento de grandes volumes de dados (big data) (HALL, 2016). Existem três principais tipos de aprendizado de máquina:

- Aprendizado supervisionado: Nesse tipo de aprendizado, o algoritmo utiliza um conjunto de dados de treinamento rotulados, nos quais há um vetor de entrada com características e atributos do sistema e um vetor de saída conhecido, que está correlacionado com as entradas propostas. O objetivo é reconhecer os padrões existentes entre os dados de entrada e saída, de modo a conseguir determinar corretamente as saídas para novos dados ainda não rotulados.
- Aprendizado não supervisionado: Nesse caso, os dados não possuem saídas específicas ou rótulos que sejam utilizados para o reconhecimento de padrões. O objetivo é encontrar características comuns nos dados e agrupá-los em conjuntos distintos.
- Aprendizado por reforço (*reinforcement learning*): Conforme descrito por Ottoni et al. (2015), no RL, a ideia principal é otimizar a tomada de decisão por meio de retornos de sucessos e fracassos dentro de um ambiente, com base na definição de um objetivo e na execução de um processo de treinamento para maximizar recompensas ou minimizar perdas.

3.4 *Reinforcement learning*

A Figura 3-8 apresenta um esquema simples no qual problemas de *reinforcement learning* (RL ou AR) podem ser enquadrados. Em geral podem esses problemas são definidos a partir da determinação dos elementos numerados abaixo (SILVA, 2016):

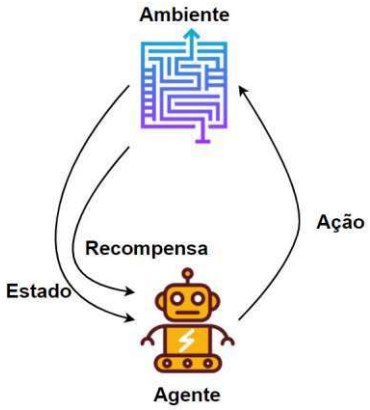


Figura 3-8-Esquema básico de um problema baseado em *reinforcement-learning*

1. Ambiente: Refere-se ao domínio onde o algoritmo de RL opera e adquire conhecimento por meio das ações disponíveis. Este ambiente pode ser representado por diversos cenários, como tabuleiros de jogos, processos industriais ou sistemas específicos.
2. Agente: Representa o elemento central do sistema, encarregado de tomar decisões. O agente é submetido a treinamento para aprender a otimizar seu comportamento dentro do ambiente observando o estado atual e buscando maximizar os reforços positivos ao longo do tempo.
3. Política de Controle/Decisão: Consiste em uma estratégia composta por um conjunto de regras de decisão que o agente emprega para selecionar a próxima ação, baseando-se na observação corrente do ambiente
4. Reforço: Refere-se à métrica de sucesso ou fracasso de uma ação tomada pelo agente em um estado específico do ambiente. O sinal de reforço, que pode assumir a forma de recompensa ou penalidade, indica a qualidade da ação executada e seu impacto no ambiente.
5. Função Valor: Trata-se de uma função que quantifica o valor de um determinado estado ou par estado-ação em um ambiente. Essa função é fundamental para o agente avaliar a qualidade de suas ações e tomar decisões relativas a ações futuras.

De uma forma geral os algoritmos de *reinforcement learning* podem ser divididos entre *based-model* e *model-free*, conforme apresentado na Figura 3-9 (HABIB, 2019). Algoritmos de RL *based-model* assumem que em cada interação o agente possui disponível um modelo que contêm toda a informação necessária para que o agente consiga obter êxito em uma determinada

tarefa. Em geral, não é possível fazer suposições precisas sobre o ambiente pouco conhecido ao qual o agente está inserido, logo o uso de algoritmos de RL *based-model* possui pouco uso prático.

Em contrapartida, algoritmos *model-free* permitem que o agente aprenda à medida que explora e executa as ações dentro de um ambiente pouco conhecido, ajustando o conhecimento do agente sobre o ambiente a cada interação. Dentre os métodos inseridos na categoria *model-free* destacam-se:

- *State-Action-Reward-State-Action* (SARSA): É considerado um método de controle TD *on-policy*, pois o método escolhe a ação para cada estado durante o aprendizado, seguindo uma determinada política.
- Q-Learning: É um método de controle de diferença temporal (TD) *off-policy*. É semelhante ao SARSA, entretanto, a escolha da próxima ação é realizada avaliando qual das ações possíveis dente a fornecer o melhor resultado, diferentemente do SARSA que escolhe a ação baseado em uma determinada política de decisão. Logo, o algoritmo de *Q-learning* toma as decisões de movimentação dentro do ambiente baseado em uma regra "gananciosa", sempre buscando a maior recompensa.

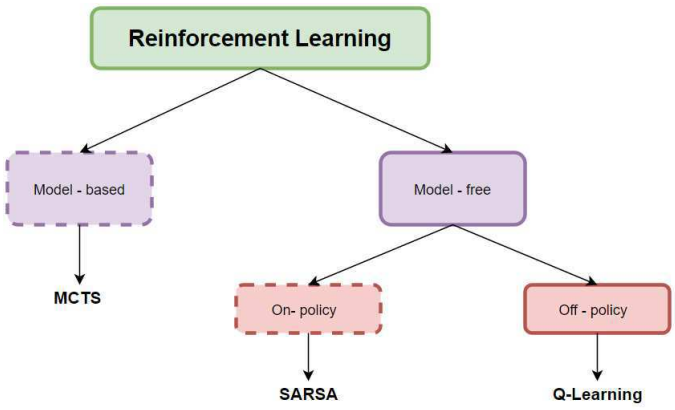


Figura 3-9 - Principais categorias e métodos na área de *reinforcement learning*

Para uma compreensão abrangente da teoria e do equacionamento envolvido na técnica de *reinforcement learning* e deep reinforcement learning, é necessário explanar os conceitos fundamentais. Os tópicos 3.4.1 a 3.4.3 têm como objetivo apresentar os fundamentos, conceitos e equações do reinforcement learning, enquanto o tópico 3.4.4 concentra-se em demonstrar como esses conceitos convergem para a construção da técnica de deep reinforcement learning.

3.4.1 Processo de decisão de Markov

No início do século 20, o matemático Andrey Markov realizou estudos envolvendo processos estocásticos sem memória chamados de "**cadeias de Markov**". A Figura 3-10 apresenta um exemplo simples de uma "cadeia de Markov" com um número fixos de estados (S). O processo de transição de um estado S_i para outro S_j e a probabilidade de transição desses estados é fixa e depende apenas do par (S_i, S_j). Desta forma, chega-se ao conceito da *propriedade de Markov*: O efeito de uma ação em um estado depende apenas da ação e do estado atual do sistema (e não de como o processo chegou a tal estado) (WEINER e PELLEGRINI, 2007).

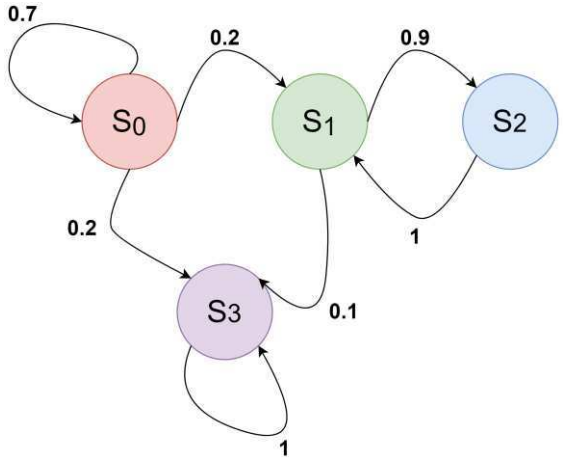


Figura 3-10 - Exemplo de uma cadeia de Markov
FONTE: Adaptado de GERÓN, 2017

A *propriedade de Markov* pode ser comprovada no exemplo apresentado na Figura 3-10. Observa-se que a transição do estado S_1 para o próximo estado (S_2 ou S_3) dependem apenas da ação tomada, sem nenhuma relação com o estado S_0, ou seja, sem nenhuma memória do estado anterior. É possível observar que existe uma maior probabilidade de transição do estado S_1 para S_2 (0.9) do que para S_3(0.1), entretanto, ao executar a cadeia de Markov n-vezes, eventualmente ocorrerá a transição do estado S_1 para S_3 o qual, para este caso, entrará em um loop permanente no próprio estado.

O conceito do **processo de decisão de markov (MDP)** foi descrito pela primeira vez em 1957 por Richard Bellman (BELLMAN, 1957). A diferença primordial entre a cadeia de Markov e o MDP baseia-se no fato de que no MDP o agente que realiza a transição entre os estados pode escolher dentre as várias ações possíveis, desta forma, as probabilidades de transição dependem agora do estado atual do agente no processo e da ação escolhida. Ademais,

a transição entre os estados pode retornar para o agente recompensas ou penalidades a depender do novo estado alcançado.

A escolha da ação que deve ser tomada é realizada por meio da definição de uma política (π). A Figura 3-11 apresenta os elementos básicos da dinâmica de um sistema modelado como processo de decisão de markov. (WEINER e PELLEGRINI, 2007).

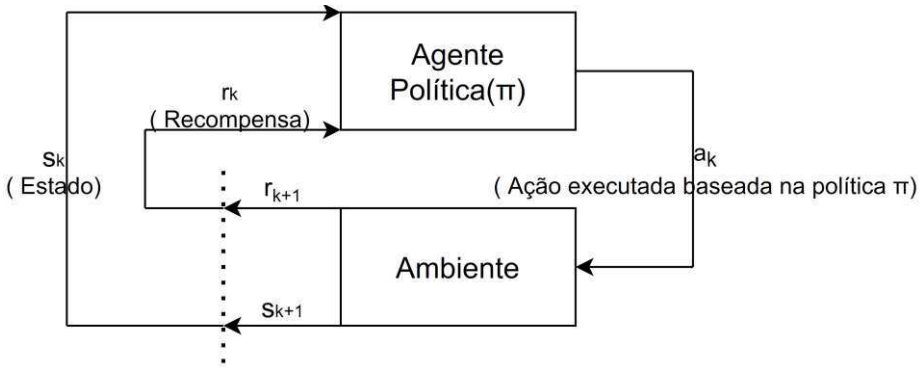


Figura 3-11 - Dinâmica de um sistema modelado como um MDP
Fonte: Adaptado de SUTTON e BARTO (2018).

A partir do estado s_k, o agente decide (através de uma regra de decisão) qual a próxima ação (a_k) a ser tomada, dentro de um conjunto de ações possíveis. A ação a_k realizada no ambiente deslocará os estados observáveis pelo agente para o ponto s_{k+1}.

Após o deslocamento, o agente é reforçado por meio de uma recompensa ou penalidade (r_k) e o processo torna a se repetir. O conjunto de regras de decisão é chamado de política (π). Em geral, opta-se pelas políticas que promovam ações que maximizam as recompensas ou minimizam as penalidades obtidas pelo agente.

A Eq.(3-16) apresenta a equação de otimalidade de Bellman a partir da qual é possível estimar o valor de estado ideal ($V^*(s)$) para qualquer estado s (GERÓN,2017).

$$V^*(s_k) = max_a \sum_{s'} T(s_k, a, s_{k+1})[r(s_k, a, s_{k+1}) + \gamma . V^*(s_{k+1})]; para\ todo\ s \qquad (3\text{-}16)$$

Onde,

- $T(s_k, a, s_{k+1})$ é a probabilidade de transição do estado s para s' dado a escolha da ação "a".
- $R(s_k, a, s_{k+1})$ é a recompensa destinada ao agente quando ele sai do estado s para s' por meio da ação "a".
- γ é taxa de desconto.

Desta forma, a equação expressa que se o agente atua de forma ótima, a **função de valor de estado ideal** será dada pela recompensa obtida após a realização da ação ótima mais o valor esperado do estado s_{k+1} descontado por γ. O conceito relacionado ao parâmetro γ será explicitado posteriormente.

Observa-se que, partindo de uma estimativa inicial dos valores atribuídos para cada estado (todos iguais a zero, por exemplo) é possível utilizar a Eq. (3-16) de forma interativa visando a determinação dos valores ótimos para todos os estados. Logo, o *algoritmo de interação de valor* pode ser expresso pela Eq. (3-17).

$$V(s_k) \leftarrow max_a \sum_{s_{k+1}} T(s_k, a, s_{k+1})[r(s_k, a, s_{k+1}) + \gamma . V(s_{k+1})]; para\ todo\ s \qquad (3\text{-}17)$$

Onde,

- $V(s_k)$ é o valor estimado do estado s na k-ésima interação do algoritmo.

A partir da Eq. (3-17) pode-se inferir que, quando $k \rightarrow \infty$ as estimativas iniciais tendem a convergir paras os valores de estado ótimo, que culminam na determinação da melhor política. Entretanto, a consideração de que o agente sempre age de forma ótima, não deixa explícito qual o caminho ou as ações que devem ser tomadas pelo agente.

Para este tipo de avaliação, Bellman definiu um algoritmo similar para determinação dos **valores estado-ação** ótimos chamado de *Q-values*.

3.4.2 Q-Learning

A partir do *algoritmo de interação Q-value* descrito por Bellman (Eq.(3-18)) é possível determinar o valor Q ideal ($Q(s_k, a)$) para o par estado-ação quando $k \rightarrow \infty$. Logo, após a determinação dos valores ótimos, é possível uma definição clara da melhor política ($\pi^*(s)$) a ser tomada dependendo do estado. Esta política está vinculada à escolha do maior *Q-value* esperado no estado futuro, ou seja, $\pi^*(s) = argmax_a Q(s_{k+1}, a)$.

$$Q(s_k, a_k) \leftarrow \sum_{s'} T(s_k, a, s_{k+1})[r(s_k, a, s_{k+1}) + \gamma . max_a Q(s_{k+1}, a)]; para\ todo\ (s, a) \qquad (3\text{-}18)$$

Para facilitar o entendimento desta etapa, a Figura 3-12 apresenta um sistema hipotético com 6 valores de estado possíveis (s, s_1, s_2, s_3, s_4 e s_5) para uma mesma variável observável (ex: temperatura). Três ações são possíveis a partir do estado s_0. A escolha de uma ação a_i (i = 1, 2 ou 3) a partir do estado s_0 desloca o processo para um dos estados possíveis s_1, s_2 ou s_3. De forma análoga, a escolha das ações disponíveis a partir de s_1, s_2 ou s_3 determinará o próximo estado.

Os *Q-values* apresentados em cada par estado-ação são os valores finais encontrados após k-ésimas interações do algoritmo descrito pela Eq.(3-18). Tendo em vista que a política para a escolha da ação é baseada nos maiores *Q-value*s para cada estado, as ações tomadas para este processo, independente do estado inicial do agente, convergem para o caminho indicado pelas setas destacadas em vermelho.

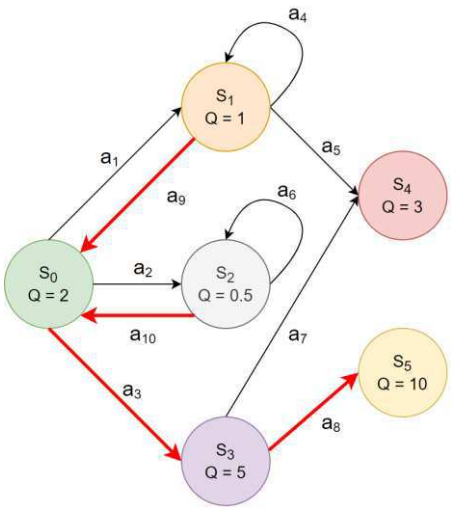


Figura 3-12 - Sistema modelado como um MDP depois de k-ésimas interações.
FONTE: Adaptado de GERÓN, 2017.

Embora a Eq.(3-18) seja útil para a determinação da melhor política a ser tomada em um processo com ações discretas modelado a partir de um MDP, é natural concluir que, em um processo desconhecido, o agente não tem nenhuma referência atrelada às probabilidades de transição dos estados ($T(s, a, s')$) e nem das recompensas que podem ser alcançadas a partir das ações tomadas.

Na prática, é necessário que o agente explore o ambiente ao menos uma vez em cada estado para identificar as recompensas possíveis e n-vezes para que seja possível estimar a probabilidade de transição entre os estados.

O *algoritmo de aprendizado por diferença temporal* (*TD-Learning*) apresentado pela Eq.(3-19) é semelhante ao *algoritmo de interação de valor* (Eq.(3-17)) modificado para levar

em consideração o fato de que o agente conhece apenas parcialmente do MDP. Neste caso, assume-se que o agente conhece apenas os estados observáveis do sistema e as ações possíveis.

$$V(s_k) \leftarrow (1-\alpha)V(s_k) + \alpha\left(r + \gamma.V(s_{k+1})\right) \tag{3-19}$$

- Onde α é a taxa de aprendizado.

De forma similar, a Eq.(3-20) é uma adaptação do *algoritmo de interação Q-value* incluindo a consideração de que inicialmente, as recompensas e probabilidades de transição entre os estados são desconhecidas.

$$Q(s_k, a_k) \leftarrow Q(s_k, a_k) + \alpha\left(r + \gamma.max_a Q(s_{k+1}, a) - Q(s_k, a_k)\right) \tag{3-20}$$

A Eq.(3-20) representa o *algoritmo Q-learning*. Para cada par estado-ação (s,a) o algoritmo mantém o controle das recompensas que o agente obtém ao deixar o estado s por meio da ação a. Considerando que a política de destino visa atuar de forma ótima, têm-se que o algoritmo sempre atualiza o valor do estado $Q(s_k, a_k)$ baseado no valor atual e no maior valor das estimativas de *Q-value* para o estado seguinte.

A inclusão da premissa de que o agente conhece apenas os estados observáveis do sistema e as ações possíveis, fomenta a inclusão de uma *etapa de exploração*. Conforme descrito anteriormente, o agente precisa explorar o ambiente para conhecer as recompensas e os valores que podem ser alcançados para os estados observáveis dado as ações possíveis.

Para isso, pode-se determinar uma taxa de exploração (ε), na qual o agente opta pela escolha randômica das ações sem se importar com as recompensas recebidas, visando a exploração do ambiente. As regras de decisão podem ser descritas pela Eq.(3-21).

$$\pi^*(s) = \begin{cases} random(a), Se\ random(0,1) < \varepsilon_n, Onde\ n\ \text{é}\ o\ episódio \\ argmax_a Q^*(s,a) \end{cases} \tag{3-21}$$

- $\varepsilon_{n+1} = \varepsilon_n - T_D;\ Sendo\ \varepsilon_1 = \varepsilon_{Inicial} = 1$
- T_D é a taxa de decaimento que em geral é linear e pode ser calculada da forma $T_D = \frac{1}{N_{eposódios}}$

Vale ressaltar que as equações descritas para o cálculo de ε e T_D bem como o valor inicial para ε podem variar dependendo do problema. Observa-se que, seguindo a política

descrita pela Eq.(3-21) o agente inicialmente optará pela realização de escolhas randômicas tendo em vista que ε_n será próximo de "1" e, portanto, $random(0,1) < \varepsilon_n$. À medida que os episódios vão sendo realizados, o valor de ε_n é reduzido devido a taxa de decaimento T_D, fazendo com que o agente opte pela escolha da ação atrelada ao maior valor de *Q-value*.

3.4.2.1 Fator de desconto e taxa de aprendizado

O fator de desconto γ é um parâmetro que pode assumir valores entre [0,1]. A função deste parâmetro é definir qual o peso será dado para os valores de ação esperados nos passos seguintes em relação ao passo atual, visando maximizar a soma de recompensas no futuro (OTTONI et al., 2015). A partir da análise da Eq.(3-20), têm-se que se $\gamma = 0$ a equação pode ser reescrita na forma apresentada pela Eq.(3-22).

$$Q(s_k, a_k) \leftarrow Q(s_k, a_k) + \alpha\big(r - Q(s_k, a_k)\big) \tag{3-22}$$

Neste caso, o agente não tem nenhum interesse nas recompensas futuras e a atualização do *Q-value* é realizada inteiramente a partir da avaliação do valor atual do *Q-value* no par (s, a). Em contrapartida, para o caso em que $\gamma = 1$ o agente tende a valorizar significativamente os valores relacionados aos passos seguintes.

Por outro lado, a taxa de aprendizado α pondera o quão rápido o agente deve aprender sobre o estado futuro frente ao estado atual. Observa-se que quando $\alpha = 0$, a Eq. (3-20) converge para a Eq.(3-23) , ou seja, não existe aprendizado, logo o *Q-value* assume valor do passo atual.

$$Q(s_k, a_k) \leftarrow Q(s_k, a_k) \tag{3-23}$$

Quando $\alpha = 1$ a Eq. (3-20) pode ser reescrita na forma apresentada pela Eq. (3-24). Logo, o agente desconsidera completamente o estado atual e foca apenas nas possíveis recompensas alcançadas nos passos seguintes por meio da avaliação do termo $\gamma . max_{\mathrm{a}} Q(s_{k+1}, a_{k+1})$.

Em resumo, se α for muito pequeno o agente pode nunca chegar na política ótima, pois supervaloriza o *Q-value* do estado atual, enquanto se α for próximo a 1 o agente pode acabar desconsiderando um *Q-value* ótimo por supervalorizar o *Q-value* futuro.

$$Q(s_k, a_k) \leftarrow \ r + \ \gamma . max_a Q(s_{k+1}, a) \tag{3-24}$$

3.4.3 *Q-table*

A *Q-table* é uma estrutura de dados utilizada para atualizar o cálculo dos *Q-value*s para cada par (s, a) através da Eq.(3-20). Após n-interações é possível mapear, através da tabela, qual a melhor ação a ser tomada para alcançar a meta e maximizar as recompensas independente do estado inicial do agente.

Para exemplificar a aplicação da Q-*table,* a Figura 3-13 apresenta o estado inicial de um tabuleiro 3x3 na qual o agente (robô) pode se descolar através da escolha de uma das ações possíveis. Como mencionado anteriormente, o algoritmo *Q-learning* parte da premissa de que o agente conhece apenas os estados observáveis do sistema e as ações possíveis. Desta forma, é preciso definir as condições iniciais e regras do sistema para que o agente possa interagir com o ambiente. Neste sentido, pode-se estabelecer as seguintes regras:

- Estado observável: posição do robô no tabuleiro;
- As ações possíveis: esquerda, direita, baixo, cima;
- Cada ação do robô recebe uma recompensa de -1.
- Ao atingir a posição L_2, C_2 (bomba) a recompensa recebida é -10.
- Ao atingir a posição L_3, C_2 (ouro) o episódio acaba, a META é alcançada e a recompensa recebida é +10.
- Número de episódios: 200
- Dentro de cada episódio o robô possui 100 tentativas para alcançar a META.
- $\alpha = 0.1$, $\gamma = 0.95$, $\varepsilon_{Inicial} = 1$

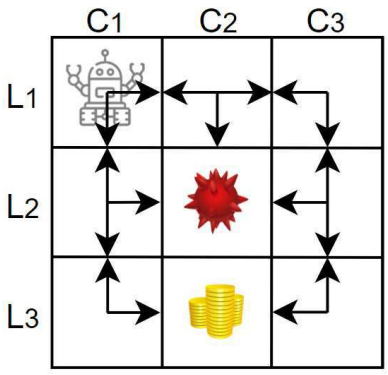


Figura 3-13 - Ambiente a ser explorado pelo agente(robô)

A *Q-table* para esse sistema pode ser representada de acordo com a Tabela 3-1 na qual cada linha representa uma posição do robô no sistema e cada coluna da tabela representa as

ações possíveis do agente para o par (L, C). Os elementos da tabela representam os valores de *Q-value* que serão atualizados ao longo das interações. Os campos com * representam as ações que levariam o robô para fora do sistema proposto, logo, estão fora do universo de ações possíveis.

Tabela 3-1 - Representação geral da Q-table para o sistema descrito na Figura 3-13

Posição\Ação	Esquerda	Direita	Cima	Baixo
L_1C_1	*	$Q(L_1C_1, D)$	*	$Q(L_1C_1, B)$
L_1C_2	$Q(L_1C_2, E)$	$Q(L_1C_2, D)$	*	$Q(L_1C_2, B)$
L_1C_3	$Q(L_1C_3, E)$	*	*	$Q(L_1C_3, B)$
L_2C_1	*	$Q(L_2C_1, D)$	$Q(L_2C_1, C)$	$Q(L_2C_1, B)$
L_2C_2	$Q(L_2C_2, E)$	$Q(L_2C_2, D)$	$Q(L_2C_2, C)$	$Q(L_2C_2, B)$
L_2C_3	$Q(L_2C_3, E)$	*	$Q(L_2C_3, C)$	$Q(L_2C_3, B)$
L_3C_1	*	$Q(L_3C_1, D)$	$Q(L_3C_1, C)$	*
L_3C_2	$Q(L_3C_2, E)$	$Q(L_3C_2, D)$	$Q(L_3C_2, C)$	*
L_3C_3	$Q(L_3C_3, E)$	*	$Q(L_3C_3, C)$	*

Inicialmente, a *Q-table* é preenchida com um valor inicial como apresentado na Tabela 3-2. Em seguida, inicia-se o episódio com o robô partindo de uma posição aleatória, neste caso considerou-se a posição L_1C_1. Partindo de uma política exploratória, é possível supor que, no início, as ações do robô serão completamente aleatórias. Neste caso, considerando a escolha do robô pela ação "direita", o valor de *Q-value* pode ser atualizado segundo a Eq.(3-20), conforme apresentado na Eq.(3-25). A Tabela 3-3 apresenta a *Q-table* atualizada após o primeiro movimento.

Tabela 3-2 – Estado inicial dos valores da *Q-table*

Posição\Ação	Esquerda	Direita	Cima	Baixo
L_1C_1	*	0	*	0
L_1C_2	0	0	*	0
L_1C_3	0	*	*	0
L_2C_1	*	0	0	0
L_2C_2	0	0	0	0
L_2C_3	0	*	0	0
L_3C_1	*	0	0	*
L_3C_2	0	0	0	*
L_3C_3	0	*	0	*

$$Q(L_1C_1, D) = (1 - 0{,}1)0 + 0{,}1(-1 + 0{,}95.0) \quad (3\text{-}25)$$

$$Q(L_1C_1, D) = -0{,}1 \quad (3\text{-}26)$$

Tabela 3-3 - Q-table atualizada após o primeiro movimento

Posição\Ação	Esquerda	Direita	Cima	Baixo
L_1C_1	*	-0,1	*	0
L_1C_2	0	0	*	0
L_1C_3	0	*	*	0
L_2C_1	*	0	0	0
L_2C_2	0	0	0	0
L_2C_3	0	*	0	0
L_3C_1	*	0	0	*
L_3C_2	0	0	0	*
L_3C_3	0	*	0	*

No próximo *step* (tentativa) o agente atualizará a posição atual L_1C_2 com base no valor de *Q-value* para a posição e na ação escolhida. O procedimento se repete até que o agente atinja a meta ou o número de tentativas máxima estipulado (100). Caso um dos dois pontos supracitados ocorra, o episódio termina, o valor de ε é atualizado de acordo com a expressão $\varepsilon_{n+1} = \varepsilon_n - T_D;\ T_D = \frac{1}{Nº Episódios}$ e um novo episódio é iniciado. A Tabela 3-4 apresenta uma configuração hipotética dos valores de *Q-table* atingida após os 200 episódios.

Tabela 3-4 - Configuração final dos valores de Q-table após os 200 episódios

Posição\Ação	Esquerda	Direita	Cima	Baixo
L_1C_1	*	-50	*	15
L_1C_2	20	22	*	-15
L_1C_3	-10	*	*	30
L_2C_1	*	-100	-40	50
L_2C_2	-20	-30	-50	10
L_2C_3	-50	*	-12	80
L_3C_1	*	60	-20	*
L_3C_2	0	0	0	*
L_3C_3	40	*	-10	*

Observa-se, que após os 200 episódios, pode-se utilizar apenas a Tabela 3-4 para determinar as melhores ações do agente, independentemente do estado inicial. A utilização da *Q-table* mostra-se eficiente para sistemas modelados por meio de um MDP onde existe um número finito de posições da variável observável, como no exemplo apresentado onde existem 9 posições possíveis dentro do ambiente explorado pelo agente.

Supondo que o ambiente seja agora descrito pela Figura 3-14 onde o agente seja o trocador de calor, a variável observável seja a composição de benzeno na base do *flash* e meta seja alcançar a composição de 0,96 a partir das ações de aumentar, reduzir ou manter a temperatura de saída do trocador. Nota-se que existem infinitos pontos possíveis de composição que podem ser atingidos a partir na manipulação da temperatura.

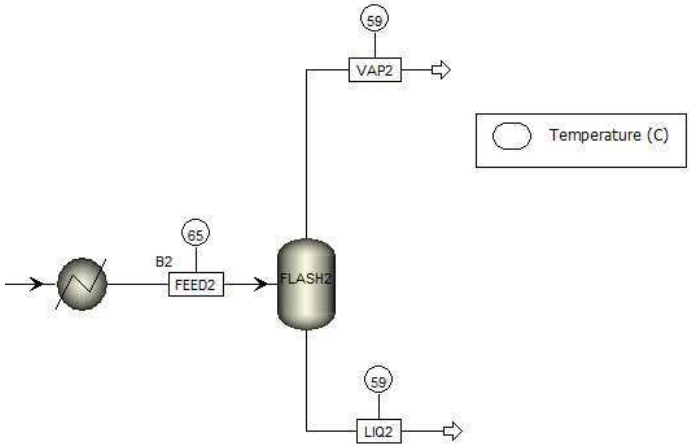


Figura 3-14 - Trocador-Flash para o sistema Benzeno-Tolueno-Metano

Neste caso, a utilização da *Q-table* só é possível a partir de um zoneamento do espaço de busca definindo um valor mínimo e máximo de composição possível bem como o espaçamento entre esses valores.

Sendo $Comp_{Mínimo} = 0,9$, $Comp_{Máximo} = 1$ e o número de espaços especificados N = 5, é possível construir uma *Q-table* (N+1)xM, onde M são as ações possíveis (3 neste caso), com a primeira linha sendo os *Q-values* atrelados a composição de 0,9 e as demais linhas os *Q-values* atrelados às composições 0,92; 0,94; 0,96; 0,98 e 1.

É possível verificar o quão sensível torna-se a *Q-table* quando a complexidade do sistema é ligeiramente aumentada. Nota-se que na teoria, quando mais espaços existirem entre os valores mínimo e máximo da variável observável, mais preciso será o deslocamento dentro da *Q-table*. Entretanto, essa estratégia tende a aumentar consideravelmente o esforço computacional.

Além disso, vale ressaltar que nos dois exemplos apresentados, existe apenas 1 variável observável. Quando a complexidade do sistema aumenta juntamente com o número de variáveis observáveis, a dimensão da *Q-table* cresce de significativamente, podendo tornar sua utilização inviável.

Visando contornar esta limitação, é possível aplicar estratégias que utilizam *Deep Learning* como parte do algoritmo de aprendizagem, substituindo a função de estruturação de dados da *Q-table*.

3.4.4 Deep Q-Learning (DQL)

A resolução de problemas complexos utilizando *reinforcement learning* pode ser modulado a partir substituição da estrutura de dados estratificada (Q-*table*) por uma rede neural que possui uma maior capacidade de generalização bem como uma maior facilidade de lidar com problemas não-lineares complexos e com diferentes dimensionalidades.

Neste sentido a Eq.(3-20) pode ser reescrita conforme apresentado na Eq.(3-27) .O desaparecimento do termo atrelado à taxa de aprendizado α se da pelo fato da atribuição desta função para o otimizador de pesos da própria rede.

$$\begin{aligned} Q(s_k, a_k) \leftarrow\ & Q(s_k, a_k) + r + \gamma . max_a Q(s_{k+1}, a) - Q(s_k, a_k) \\ & Q(s_k, a_k) \leftarrow r + \gamma . max_a Q(s_{k+1}, a) \end{aligned} \tag{3-27}$$

A Figura 3-15 apresenta os percentuais de acerto cumulativo para as duas estratégias avaliadas, calculado a cada episódio com base na equação $\frac{\sum_1^i n_{Sucess}}{n_{Episodes}}$, onde i é o episódio atual, n_{sucess} é o número de episódios finalizados com sucesso até o momento e $n_{Episodes}$ é o número de episódios totais finalizados.

As condições utilizadas para delimitação do ambiente e da rede são descritas na Tabela 3-5.

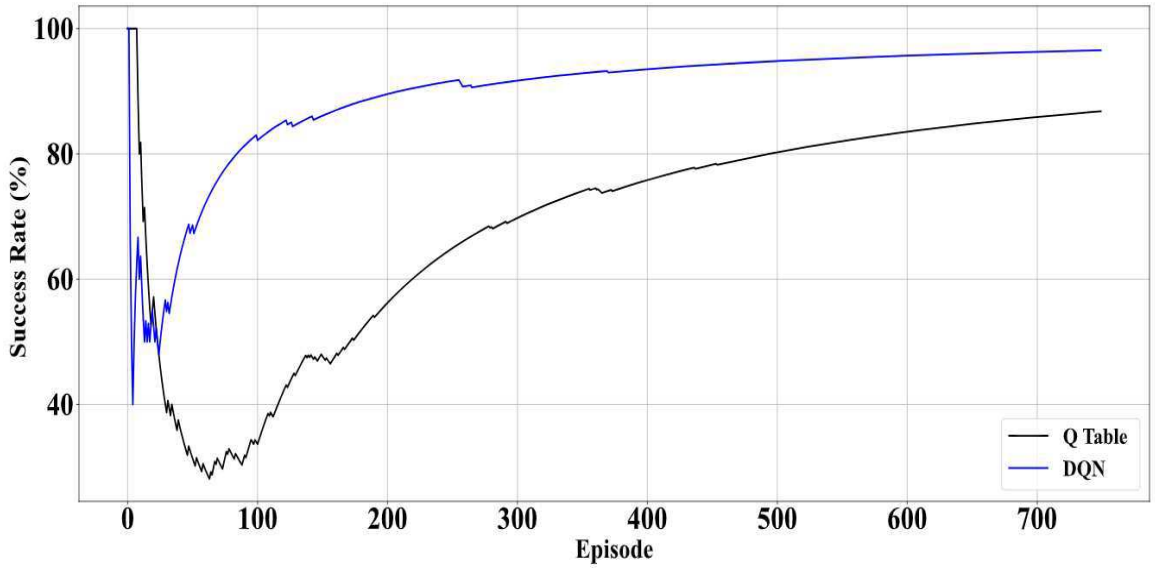


Figura 3-15 - Percentual de acertos calculados ponto a ponto

Tabela 3-5 - Parâmetros para construção das arquiteturas

Parâmetro	DQN	Q-Table
Meta inicial	0,92	
Tolerância	1e-4	
N° de Episódios	750	
N° de passos por episódios	100	
N° Ações	3	
Taxa de aprendizado	0,01	
Epsilon	1	
Decaimento Epsilon	1/250	
Epsilon Mínimo	0,01	
Fator de desconto (γ)	0,99	
Taxa de atuação	0,3	
Recompensa inicial	0	
Recompensa para acerto	+10	
Recompensa para erro	-1	
Dimensão da Matriz	-	1000
Limite inferior de busca	-	0,9
Limite superior de busca	-	1

Observa-se que ao longo dos episódios, o percentual de acertos calculado é maior utilizando a estrutura DQN atingindo um percentual acima de 90% após 400 episódios, enquanto a estrutura Q-*Table* alcança um valor próximo a 85% ao final dos 750 episódios.

A partir da Figura 3-16 observa-se que, não só é possível construir um sistema para atuação e controle da composição na base do flash por meio da temperatura como também é possível (caso necessário) um ajuste da meta indicando uma melhor condição operacional frente à meta inicial.

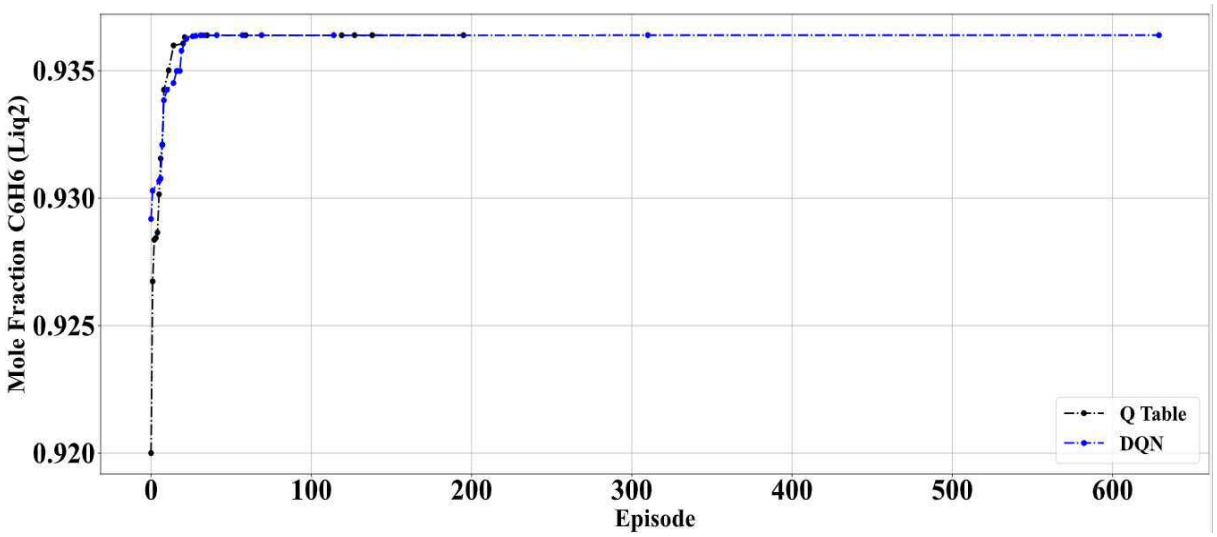


Figura 3-16 - Ajuste da condição ótima de composição de Benzeno na base do flash.

Capítulo 4 Aplicação Q-Table

Data-driven control system for distillation columns using Q- learning.

Gladson Euler, Luís Gonzaga, Karoline Brito, Romildo Brito

Federal University of Campina Grande, Chemical Engineering Department, Campus Universitario, Campina Grande, PB, Brazil, 58109-970

Abstract:

A precisão do controle de composição em colunas de destilação é fundamental para garantir a qualidade do produto e o desempenho do processo. No entanto, os controladores feedback do tipo PID comumente usados nesses processos podem apresentar limitações devido à malha de controle, aos distúrbios nas variáveis de processo e à configuração dos parâmetros do controlador. Este estudo propõe a criação de um sistema de controle baseado em dados (DDC) que utiliza Q-Learning acoplado a um controlador PID para corrigir desvios na composição do produto causados por uma malha de controle inferencial. Duas estratégias de controle foram avaliadas com base nas ações escolhidas pelo algoritmo: a primeira manteve uma taxa de atuação fixa de 0.5%, enquanto a segunda flexibilizou a taxa de atuação entre 0.1% e 5% com base no desvio entre o estado atual e a meta escolhida. Os resultados dos testes confirmaram a limitação da malha de controle inferencial na manutenção da composição de isobutano (IC_4) na base da coluna para distúrbios na composição de propano na alimentação. Por outro lado, o Q-learning com taxa de atuação variável apresentou aproximadamente o dobro de assertividade em relação à taxa fixa, indicando que a flexibilização da taxa de atuação permite a expansão das regiões em que o algoritmo pode manter a composição dentro da especificação. O ambiente dinâmico utilizado para treinamento e validação das propostas de controle, o Aspen Plus Dynamics, garantiu a robustez, não-linearidade e aspecto transiente do processo, permitindo a inclusão das principais variáveis observáveis (carga térmica do reboiler e a composição de IC_4) e restrições (saturação das válvulas do sump e condensador), tornando as estratégias de controle propostas mais viáveis para a aplicação prática.

Keywords: Controle data-driven; coluna de destilação, Aspen plus dynamics, Q-learning, Algoritmo de treinamento genérico.

1.Introdução

O controle de composição em colunas de destilação de alta pureza é um aspecto crítico do ponto de vista econômico. A implementação de uma estrutura de controle adequada é tão importante quanto a otimização do processo, impactando diretamente no desempenho da coluna e, consequentemente, nos custos econômicos associados e na viabilidade do processo (Ramos et al., 2016).

Neste sentido, estruturas robustas de controle devem ser aplicadas para garantir as especificações requeridas. Tradicionalmente, controladores clássicos do tipo PID são utilizados na grande indústria, especialmente em colunas de destilação para garantir a pureza dos produtos de base e topo. Devido à natureza complexa, não linear e multivariável, desvios nas características do processo ou alterações de setpoint em detrimento de mudanças de especificação do produto podem deteriorar o desempenho do controlador com o tempo (Tulsyan et al., 2019; Tulsyan, Garvin, & Ündey, 2018; Tulsyan, Garvin, & Undey, 2018).

A utilização de controles industriais modernos baseados em modelos matemáticos fundamentados em princípios físico-quimicos (first-principles) ou obtidos por identificação de sistemas, podem ser particularmente caros e demorados (Hou & Wang, 2013). A reidentificação do sistema, quando necessária, pode durar semanas (Kano & Ogawa, 2009) e, normalmente, envolve a injeção de excitações externas (Tulsyan et al., 2013) que introduzem um custo pela interrupção do funcionamento normal do processo.

Como alternativa aos controladores regulatórios clássicos ou baseados em modelo, estratégias de controle do tipo data-driven utilizando algoritmos de *reinforcement learning* (RL) tem sido proposta devido ao desenvolvimento da ciência e tecnologia da informação (Selvi et al., 2018).

O RL baseia-se em uma política de identificação do ambiente, portanto, não necessitam de um conjunto inicial de dados de treinamento para obtenção do modelo do processo, tornando a lei de controle auto-organizada; ou seja, capaz de mudar em função de sua experiência (Mendel & McLaren, 1970). Desta forma, o RL é considerado um poderoso método data-driven que fornece a política ótima para resolver problemas de controle (Radac & Precup, 2019).

Embora seja considerado uma área recente frente ao aprendizado supervisionado e não supervisionado, os algoritmos de RL tem sido utilizado em processos industriais para diversas aplicações com resultados promissores: a) modelagem e otimização de reatores em batelada (Martinez, 2000; Wilson & Martinez, 1997); b) controle de pH (Syafiie et al., 2007, 2008) ;c) controle em colunas de extração líquido-líquido em processos biofarmacêuticos (Hwangbo & Sin, 2020), etc.

Em geral, estudos envolvendo RL para controle em colunas de destilação modelam o processo (ambiente) utilizando três abordagens: 1) Modelos linearizados a partir de funções de transferências (MIMO) simplificadas (Kretchmar et al., 2001; Spielberg et al., 2019); 2) Equações matemáticas que avaliam as relações entre as variáveis observáveis do sistema (Wang & Ge, 2020); 3) Simuladores de processos químicos robustos como o Aspen Plus, em regime estacionário, desconsiderando a natureza transiente do controle (Hwangbo & Sin, 2020).

Nesse contexto, este trabalho propõe a utilização de um ambiente dinâmico (Aspen Plus Dynamics) para o desenvolvimento de um sistema de controle baseado em dados (DDC) utilizando o algoritmo Q-Learning acoplado a um controlador PID convencional para a redução de *offsets* de composição gerados em decorrência da malha de controle inferencial utilizada.

O Aspen Plus Dynamics apresenta-se como uma fonte confiável para construção e obtenção de informações do modelo dinâmico da coluna de destilação. A manutenção da complexidade, não-linearidade e aspecto transiente do processo aumenta a viabilidade de aplicação prática das estratégias propostas além de permitir a inclusão de variáveis de treinamento importantes na dinâmica de controle como temperatura do prato sensível, carga térmica do reboiler e composição do produto na base da coluna.

2.Q-Learning – Conceitos

Um problema de *reinforcement learning* pode ser formulado partindo da premissa de um processo de decisão de markov (MDP), ou seja, o efeito de uma ação em um estado do sistema depende apenas da ação realizada e do estado atual, sem qualquer relação com os estados anteriores (Pellegrini & Wainer, 2007). Os principais elementos que compõem um problema de *reinforcement learning* são descritos abaixo e apresentados na Figura 4-1.

1. Ambiente: Espaço onde o algoritmo de RL irá mapear e aprender a partir das ações possíveis. Este ambiente pode ser um tabuleiro, processo, equipamento dentre outros.
2. Política de Controle/Decisão: É a estratégia composta por um conjunto de regras de decisão que o agente utiliza para selecionar a próxima ação com base na observação atual do ambiente.
3. Reforço: É a medida de sucesso ou fracasso de uma ação tomada pelo agente em um determinado estado do ambiente. O sinal (recompensa ou penalidade) atribuído pelo ambiente indica a qualidade da ação tomada e o seu impacto no ambiente.
4. Função Valor: É uma função que mede o valor de um determinado estado ou par estado-ação em um ambiente. Ela é utilizada pelo agente para avaliar a qualidade de suas ações e

decidir as próximas ações. A função valor que considera o par estado-ação (s, a) é denotada por Q(s,a) e denominada função valor-ação.

5. Agente: É o elemento que toma as decisões dentro do sistema. O agente é treinado para aprender a melhor maneira de agir em um determinado ambiente observando o estado atual e maximizando os reforços positivos ao longo tempo.

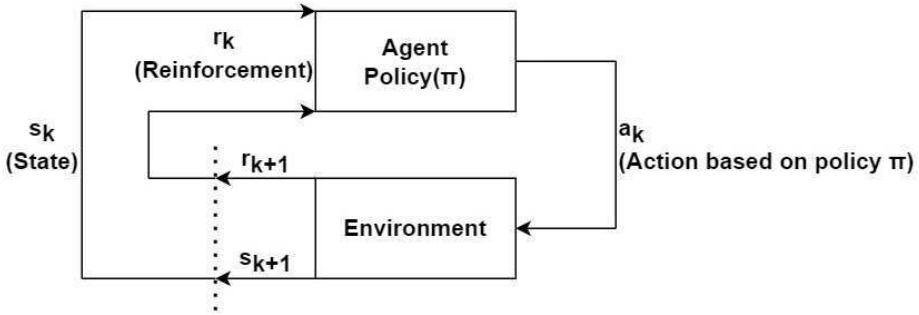


Figura 4-1 - Formulação geral de um problema de RL
Fonte: Adaptado de SUTTON e BARTO (2018).

A partir do estado s_k, o agente decide (através de uma regra de decisão) qual a próxima ação (a_k) a ser tomada, dentro de um conjunto de ações possíveis. A ação a_k realizada no ambiente deslocará os estados observáveis pelo agente para o ponto s_{k+1}.

Após o deslocamento, o agente é reforçado por meio de uma recompensa ou penalidade (r_k) e o processo torna a se repetir. O conjunto de regras de decisão é chamado de política (π). Em geral, opta-se pelas políticas que promovam ações que maximizam as recompensas ou minimizam as penalidades obtidas pelo agente.

A Eq. (4-1) apresenta a função valor do algoritmo *Q-learning*, que é empregado para atribuição de Q-values para cada par estado-ação(s,a) (Géron, 2017). O algoritmo do *Q-learning* corrige o valor atual do estado $Q(s_k, a_k)$ baseado no maior valor das estimativas de *Q-value* para o estado a posteriori ($max_a\ Q(s_{k+1}, a)$), ponderado pelo fator de desconto γ.

O parâmetro α determina a rapidez com que o modelo aprende a partir dos dados enquanto γ está atrelado ao peso da recompensa futura em relação a recompensa imediata.

$$Q(s_k, a_k) \leftarrow\ Q(s_k, a_k) + \alpha\big(r_k + \gamma . max_a\ Q(s_{k+1}, a) - \ Q(s_k, a_k)\big) \quad (4\text{-}1)$$

Neste trabalho, a aplicação do algoritmo Q-Learning foi realizada utilizando a estrutura básica de dados Q-table para o zoneamento do ambiente. A Tabela 4-1 apresenta a Q-table para um sistema genérico com n estados e z ações possíveis. À medida que o agente se desloca pelos

estados do ambiente representados pelas linhas da Q-table, os Q-values para cada par (s, a) são atualizados através da Eq. (4-1).

Tabela 4-1 - Q-table para um sistema genérico

Estado\Ação	Ação 1	Ação 2	⋯	Ação z
s_1	$Q(s_1, \text{Ação}_1)$	$Q(s_1, \text{Ação}_2)$	⋯	$Q(s_1, \text{Ação}_z)$
s_2	$Q(s_2, \text{Ação}_1)$	$Q(s_2, \text{Ação}_2)$	⋯	$Q(s_2, \text{Ação}_z)$
s_3	$Q(s_3, \text{Ação}_1)$	$Q(s_3, \text{Ação}_2)$	⋯	$Q(s_3, \text{Ação}_z)$
⋮	⋮	⋮	⋮ ⋯	⋮
s_n	$Q(s_n, \text{Ação}_1)$	$Q(s_n, \text{Ação}_2)$	⋯	$Q(s_n, \text{Ação}_z)$

3.Abordagem do problema

Conforme mostrado anteriormente, o conceito de *reinforcement learning* envolve uma série de elementos. A maneira como esses elementos se encaixam no controle da coluna de destilação é resumida na Tabela 4-2 e explicados em detalhes em seguida.

Tabela 4-2 - Resumo dos elementos RL para implementação do DDC.

Elemento	**Contexto no processo de destilação**	**Contexto do RL**
Ambiente	Processo de destilação	Modelo em Aspen Plus Dynamics
Tarefa	Encontrar as condições de operação para um alvo de composição	Eqs. (4-2)-(4-6)
Estado	Condições de operação	State $s \in$ {[IC_4], SUMP%, CD%, Q_{REB}}
Reforço	Baseado no estado atual	Eq. (4-7)
Agente	Algoritmo Python-Aspen	Figura 4-4
Ações	Baseado no desvio de $IC_{4\ Alvo}$	Eqs. (4-8) e (4-9)
Política	Regra seguida para determinar as ações para correção do offset de composição.	Figura 4-5.b

3.1.Ambiente

A Figura 4-2 apresenta o fluxograma do processo de destilação para o sistema propano (C_3) - isobutano (IC_4) proposto por Luyben, 2006. A coluna de destilação, modelada a partir do bloco RadFrac, foi implementada no software Aspen Plus, utilizando o modelo termodinâmico CHAO-SEADER. Posteriormente, a coluna foi exportada para o Aspen Plus Dynamics.

O modelo termodinâmico CHAO-SEADER foi escolhido devido à sua capacidade comprovada de lidar com as relações de equilíbrio líquido-vapor (VLE) em sistemas de hidrocarbonetos em diferentes faixas de temperatura e pressão (CHAO e SEADER, 1961).

Os controles regulatórios adicionados estão vinculados aos níveis do condensador e do *sump* juntamente com o controle de pressão da coluna. Um controle de razão(R/F) foi adicionado para manter a proporção mássica (kg/s) entre a alimentação e o refluxo da coluna.

O propano é retirado no primeiro estágio da coluna com uma pureza de 98%. O isobutano é retirado na base da coluna com pureza especificada de 99%. A malha de controle utilizada para manter a composição de IC_4 na base da coluna dentro da especificação é do tipo inferencial por meio do controle de temperatura do prato sensível (estágio 9) utilizando a carga térmica do reboiler como variável manipulada. As especificações técnicas do sistema proposto estão descritas na Tabela 4-3.

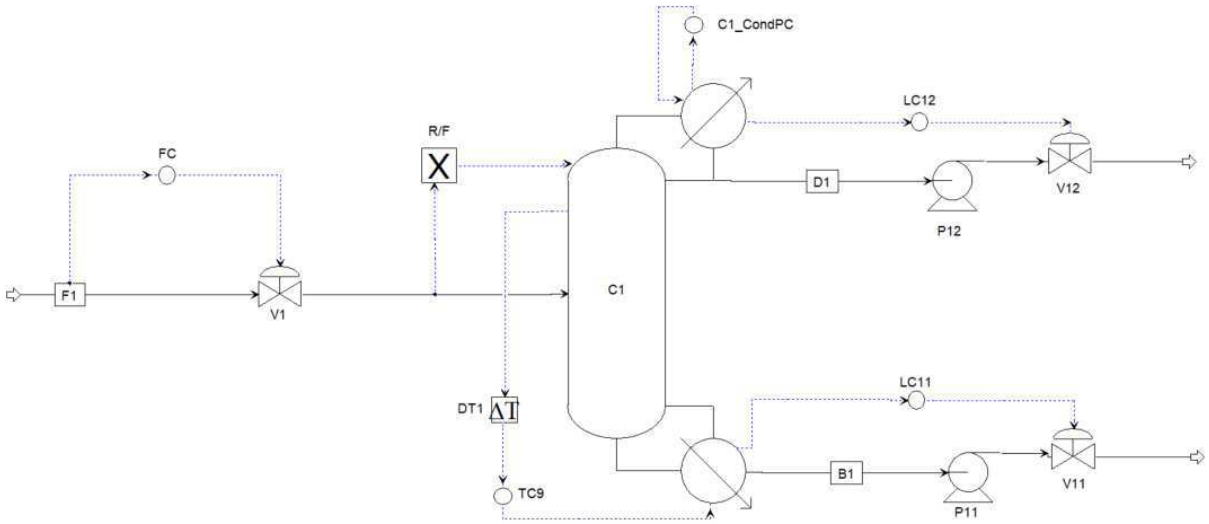


Figura 4-2 - Fluxograma do processo de destilação(C3-IC4)

Tabela 4-3 - Especificações técnicas do processo.

Variable	**Valor**	**Unidade**
Taxa de alimentação (F1)	1	kmol/s
Temperatura (F1)	322	K
Pressão (F1)	20	atm
Composição C_3 (F1)	0.4	-
Composição IC_4 (F1)	0.6	-
Número de estágios (C1)	32	
Taxa de destilado (C1)	0.4	kmol/s
Refluxo (C1)	3.46	-
Estágio de alimentação (C1)	14	-
Pressão no condensador (C1)	16.8	atm
Queda de pressão no estágio (C1)	6.8×10^{-3}	atm
Nível no condensador – Setpoint (LC12)	5.10	m
Nível no sump setpoint (LC11)	6.35	m
Pressão no condensador Setpoint (C1_CondPC)	16.8	atm
Temperatura do estágio 9° e Setpoint (TC9)	337.36	K
R/F	1.18	-

3.2.Tarefa

No contexto do processo de destilação proposto, a tarefa do algoritmo pode ser formulada conforme apresentado nas Eqs. (4-2)-(4-6) abaixo.

$$Encontrar\ \ s|\theta\ com\ Desvio \leq Tol; \tag{4-2}$$
$$Tol = 1\ x\ 10^{-4} \tag{4-3}$$
$$Desvio = |IC_{4\ COMP} - IC_{4\ Alvo}|;\ IC_{4\ Alvo} = 0.99 \tag{4-4}$$

Sujeito a:

$$20\% < SUMP_{\%} < 80\% \tag{4-5}$$
$$20\% < \mathrm{CD}_{\%} < 80\% \tag{4-6}$$

Onde s representa as condições operacionais que compõem as variáveis observáveis do ambiente, sendo $s \in$ {Composição de Isopropano ([IC_4]); Abertura da válvula - Sump ($SUMP_{\%}$), Abertura da válvula - condensador ($CD_{\%}$), Carga térmica do Reboiler (Q_{REB})}.

O objetivo das restrições atreladas à $SUMP_{\%}$ e $\mathrm{CD}_{\%}$ é evitar que, ao longo do processo de aprendizado, o algoritmo opte por regiões onde as aberturas das válvulas relacionadas ao condensador/sump estejam próximas dos limites especificados nas Eqs. (4-5) e (4-6), ou seja, próximas da saturação.

θ representa os parâmetros da taxa de aprendizado (α) e fator de desconto (γ) atrelados ao reinforcement learning, ou seja, $\theta \in \{\alpha, \gamma\}$. Os valores de α, γ foram definidos empiricamente e mantidos fixos em 0.01 e 0.99, respectivamente, para as estratégias avaliadas neste trabalho.

3.3.Reforço

Os reforços determinam que tipo de retorno o ambiente entregará para o agente em decorrência da ação tomada e do novo estado s_{k+1} obtido. Para o problema proposto, as funções de reforço são descritas pela Eq. (4-7).

$$R_s^a = \begin{cases} 0; se\ k = 1; \\ 10, se\ a\ tarefa\ (Eqs.\ (4\text{-}2) - (4\text{-}6)) foi\ concluída; Status: Concluído \\ R_s^a - 1, se\ \ k \neq 1\ ;\ k < k_{max}\ e\ o\ Desvio > Tol; Status: Não\ Concluído \\ -k_{max}, se\ as\ restrições\ (Eqs.\ (4\text{-}5)\ ou\ (4\text{-}6)) não\ forem\ atendidas\ ou\ k = k_{max}; Status: Falha \end{cases} \tag{4-7}$$

Os reforços foram estruturados de forma a garantir uma bonificação para os casos em que o algoritmo alcança um estado s_{k+1} que satisfaz a tarefa especificada, e fornecendo uma penalidade para cada step *k* onde algoritmo não conclui a tarefa.

Tendo em vista que os valores dos reforços serão utilizados para atualização dos Q-values, a penalização a cada step cria uma espécie de superfície. Desta forma, ao final do treinamento, o algoritmo escolherá as ações que maximizam os valores de Q, a partir da escolha do caminho mais rápido para realização da tarefa, independente do estado inicial do sistema.

Para os casos em que o algoritmo propõe uma ação que culmina na obtenção de um estado s_{k+1} no qual as restrições atreladas às válvulas do sump e/ou do condensador (Eqs. (4-5) e (4-6)) não são atendidas, o agente recebe a penalidade correspondente ao número de steps máximo.

3.4Ações

No presente trabalho, a mudança do estado s_k para s_{k+1} ocorre a partir da alteração do setpoint do controlador referente a temperatura do prato sensível (estágio 9). A escolha da manipulação do setpoint de temperatura no estágio 9 decorre da influência dessa variável na composição de isobutano na base da coluna. A Figura 4-3 mostra a variação da composição de IC_4 com a alteração da temperatura do prato sensível.

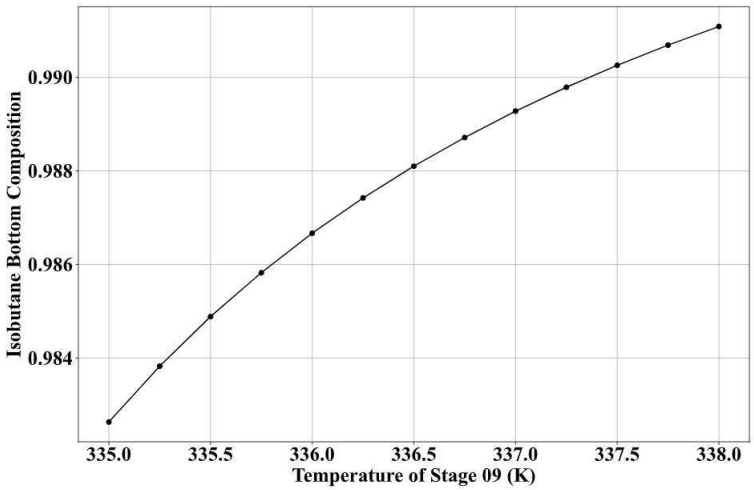


Figura 4-3 - Variação de IC_4 na base da coluna com a alteração da temperatura do estágio 9

Existem três ações possíveis, formuladas conforme apresentado na Eq. (4-8), ou seja, é possível aumentar/reduzir a temperatura de set-point (T_{SP}) à uma taxa AR ou não alterar a temperatura.

Neste trabalho serão avaliadas duas estratégias na execução das ações de controle do Q-Learning. Na primeira estratégia, a taxa de alteração (AR) é fixa em 0.5%, enquanto na segunda ela é variável (Eq. (4-9)) e depende do desvio entre a composição de isopropano atual e a especificada para a corrente de produto na base da coluna.

A utilização combinada das Eq. (4-9) e (4-10), permite que o agente se desloque em um ambiente contínuo por meio de ações discretas. A escolha por uma forma simplificada para compor as ações do algoritmo em ambas as estratégias teve como objetivo reduzir a complexidade do problema de RL.

$$A = \begin{cases} T_{SP}(1 - AR) \\ T_{SP} \\ T_{SP}(1 + AR) \end{cases} \tag{4-8}$$

$$AR = \begin{cases} 0.1\%; 1\ x\ 10^{-4} \leq Desvio \leq 5\ x\ 10^{-4} \\ 0.5\%;\ 5\ x\ 10^{-4} \leq Desvio \leq 1\ x\ 10^{-3} \\ 1\%; 1\ x\ 10^{-3} \leq Desvio \leq 5\ x\ 10^{-3} \\ 5\%;\ Desvio > 5\ x\ 10^{-3} \end{cases} \tag{4-9}$$

3.5.Agente e Política

A coluna de destilação é um ambiente dinâmico modelado no Aspen Plus Dynamics. Neste sentido, a utilização da Q-table precisa estar integrada à um algoritmo recursivo que avalie os estados e tome as ações ao longo da simulação. A Figura 4-4 apresenta o algoritmo construído em Python que integra a Q-table e o ambiente dinâmico na etapa de treinamento para ambas as estratégias (AR fixo e variável).

Neste caso, o agente é o próprio algoritmo, responsável por tomar as decisões com base nas informações recebidas do ambiente, aprendendo com as experiências para alcançar a tarefa especificada. A maneira como o agente interage com o ambiente e atualiza os valores da Q-table ao longo dos episódios de treinamento é apresentado a seguir.

Inicialmente os parâmetros gerais da simulação e a estruturação da Q-table são definidos na etapa “Set Parameters” com base na Tabela 4-4 e Tabela 4-5, respectivamente. Na Tabela 4-4, os números de episódios e passos foram definidos empiricamente, enquanto os valores iniciais dos estados observáveis estão relacionados com o estado estacionário do sistema. Os distúrbios foram definidos considerando um range de $\pm$15% em relação a condição nominal da composição de propano na alimentação (0.4).

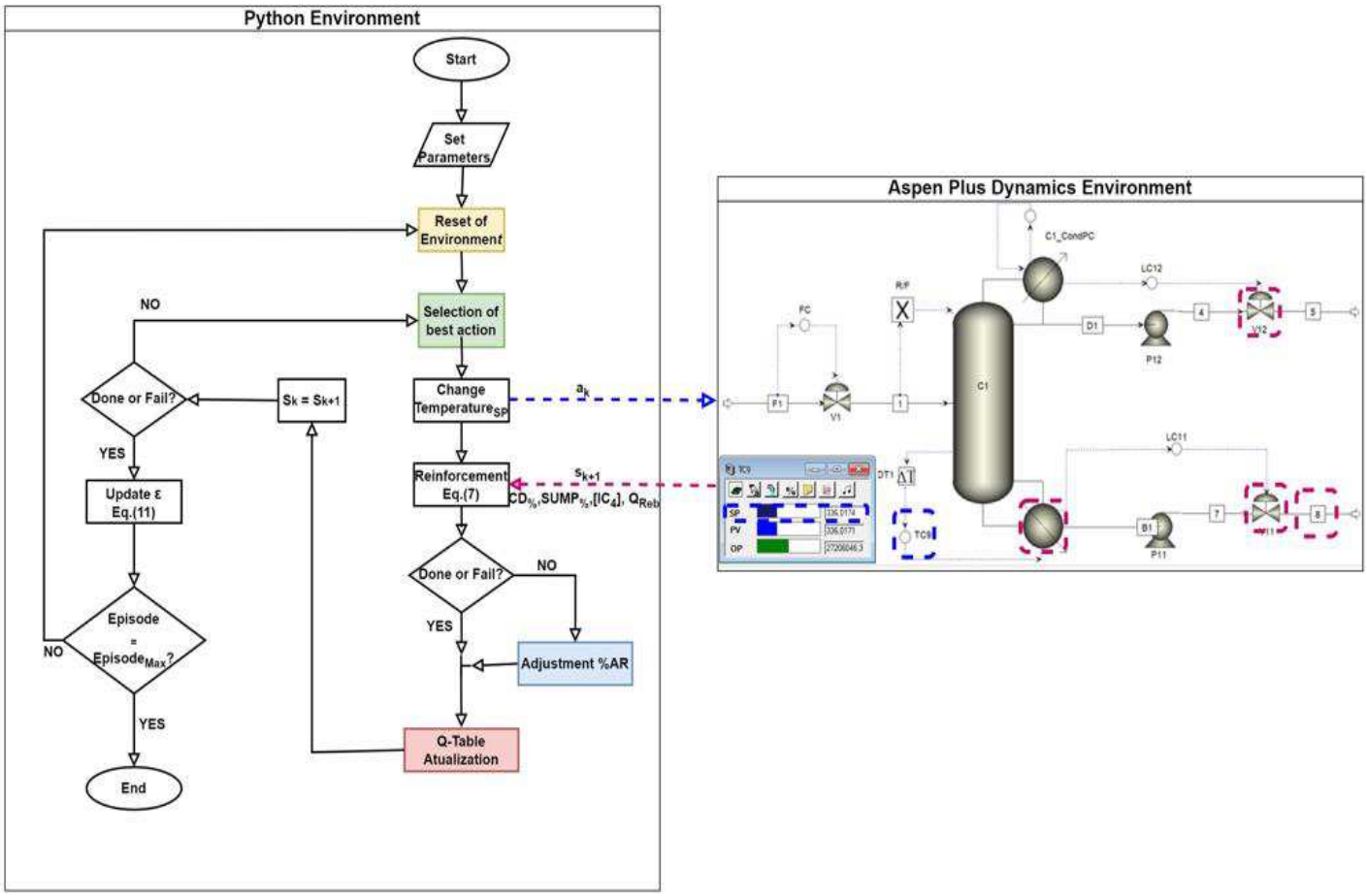


Figura 4-4 - Fluxograma geral do algoritmo de treinamento

Tabela 4-4 - Parâmetros gerais do algoritmo

Parâmetro	Valor	Unidade
Episódios	600	-
Passos	150	-
$\varepsilon_{\text{Inicial}}$	1	-
$\varepsilon_{\text{Decay}}$	$\frac{1}{Episodes/2}$	-
$\varepsilon_{\text{Mínimo}}$	0.01	-
Estado Inicial IC_{4COMP}	0.99	-
Estado Inicial SUMP%	50	%
Estado Inicial CD%	50	%
Estado Inicial Q_{REB}	2.71×10^7	W
Distúrbio na Alimentação	$[C_3]$ following $U\sim(0.34 – 0.46)$	-

Tabela 4-5 - Parâmetros utilizados para geração da Q-table

Parâmetro	Valor	Unidade
S_{MIN}	$[0.9; 0; 0; 2.08 \times 10^7]$	[-, %, %, W]
S_{MAX}	$[1; 100; 100; 2.92 \times 10^7]$	[-, %, %, W]
Nº Divisões da tabela	200	-
Nº Colunas da tabela	3	-
Valores Iniciais $Q(s_k,a_k)$	0	-

Os limites mínimos e máximos das variáveis observáveis apresentados na Tabela 4-5 foram definidos para o zoneamento do ambiente. O número de divisões (linhas) da Q-table foi definido com base na memória consumida pelo computador, enquanto o número de colunas é igual ao número de ações que o agente dispõe para se locomover no ambiente, neste caso três, conforme descrito na Eq. (4-8).

Após a definição dos principais parâmetros, o algoritmo segue para o módulo "Reset of Environment" (Figura 4-5.a) que controla o início dos novos episódios. Neste módulo, a simulação é resetada (para Episódio >1) e posteriormente iniciada. Após 1h de simulação, um distúrbio randômico é aplicado na composição de propano na corrente de alimentação F1 da coluna seguindo U~ (0.34-0.46).

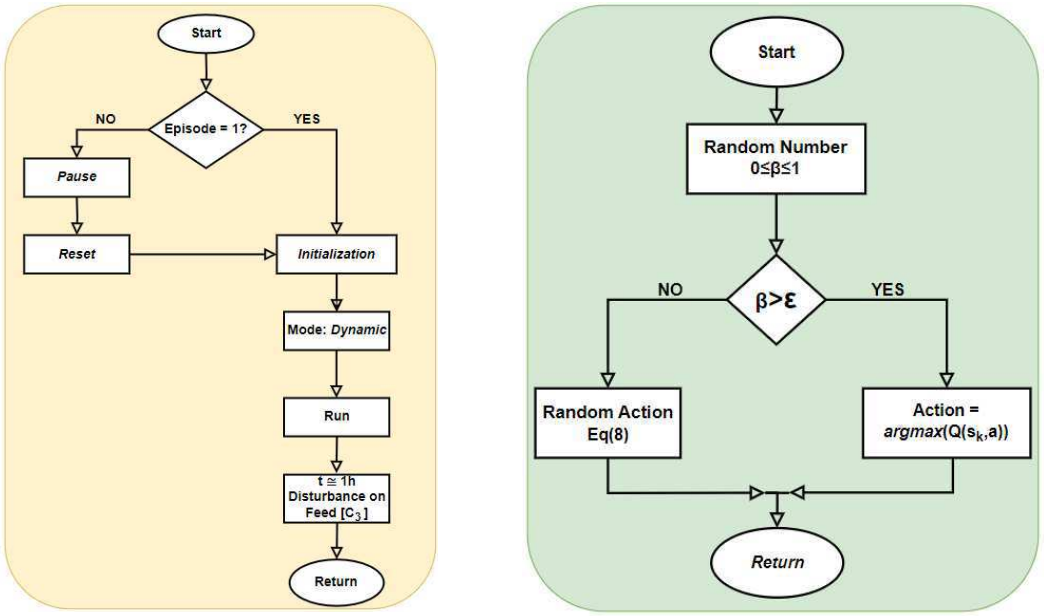
A partir deste ponto, as ações do algoritmo são definidas no módulo "Selection of best action" (Figura 4-5.b) com base nas relações de exploração e aproveitamento seguindo a política (π) ε-greed.

Nos episódios iniciais (fase de exploração), o valor de ε ainda é próximo ao valor de ε_Inicial apresentado na Tabela 4-4, logo, o valor de β gerado randomicamente tende a ser inferior a ε, convergindo para ações aleatórias. Esse comportamento é provocado buscando explorar o ambiente.

A cada novo episódio, o valor de ε é reduzido seguindo o valor de ε_Decay, conforme apresentado na Eq. (4-10) até atingir o $\varepsilon_{\text{Mínimo}}$ na metade do treinamento, iniciando a fase de aproveitamento.

Nesta fase, o algoritmo utiliza os valores de Q(s,a) aprendidos durante a fase de exploração para selecionar a melhor ação possível em cada estado, refinando os valores de Q(s,a) a partir das informações adquiridas na fase anterior buscando maximizar a recompensa esperada.

$$\varepsilon = \begin{cases} \varepsilon_{\text{Inicial}}; Episódio = 1 \\ \varepsilon - \varepsilon_{\text{Decay}}; 1 < Episódio < Episodios/2 \\ \varepsilon_{\text{Mínimo}}; Episódio \geq 300 \end{cases} \tag{4-10}$$



(a) (b)

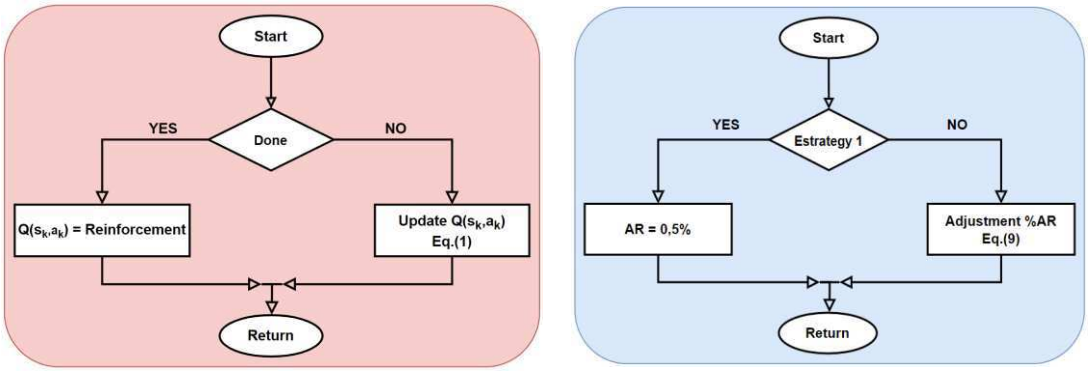
Figura 4-5 - Subetapas do algoritmo referentes ao controle de novos episódios (4-5.a) e a determinação das novas ações (4-5.b)

Após a determinação da ação (a_k), um novo contato com a simulação no Aspen Plus Dynamics é realizado para a alteração do setpoint do controlador da temperatura do estágio 09 da coluna. Em seguida, o novo estado s_{k+1} do sistema é coletado 1h após a alteração do setpoint e o reforço do agente é fornecido com base na Eq. (4-7). O tempo de 1h é necessário para o sistema atingir o novo estado estacionário.

Com o reforço definido, o algoritmo pode seguir diretamente para a etapa de atualização da Q-table no módulo "Q-Table Atualization" (Figura 4-6.a) ou para atualização prévia da taxa de manipulação no módulo "Adjustment %AR" (Figura 4-6.b) dependendo do status do episódio.

O ajuste de $Q(s_k, a_k)$ realizado na Q-table também depende do status do episódio. Para os casos em que o agente alcança a meta especificada, o Q-value para estado-ação (s_k, a_k) assume o valor do reforço alcançado. Caso contrário, o valor de $Q(s_k,a_k)$ é atualizado com base no reforço atual e na recompensa futura ($max_a\ Q(s_{k+1}, a)$) a partir da equação base do Q-learning (Eq. (4-1)).

Após a atualização do passo s_k, o algoritmo pode retornar para o módulo "Selection of best action" ou seguir para o próximo episódio, dependendo do status. Na segunda opção, o valor de ε é atualizado com base na Eq. (4-10) e um novo episódio é iniciado no módulo "Reset of Environment".



(a) (b)

Figura 4-6 - Subetapas referente ao update de Q-Table (4-6.a) e Ajuste do AR (4-6.b)

4.Resultados

4.1.Treinamento

A etapa de treinamento foi realizada seguindo o fluxograma geral da Figura 4-4. A Figura 4-7.a apresenta os valores de composição de IC_4 na base da coluna obtidos durante o treinamento para o Q-learning com AR fixo, enquanto a estratégia com AR variável é apresentada na Figura 4-7.b. Os distúrbios que culminaram em falha ao longo dos episódios para as estratégias com AR fixo e variável são apresentados na Figura 4-7.c e Figura 4-7.d, respectivamente.

Os valores iniciais da Q-table definidos na Tabela 4-5 não fornecem uma boa estimativa dos valores de Q para cada ação. Neste sentido, espera-se que o agente aprenda cometendo mais erros do que acertos na fase de exploração.

No final da etapa de aprendizado apresentado na Figura 4-7.a e Figura 4-7.b, a utilização da Q-Table foi capaz de fornecer resultados que corrigiram o offset de composição gerado pelo distúrbio na alimentação nas duas estratégias avaliadas, mantendo a composição de isobutano na base da coluna dentro da tolerância (1×10^{-4}) especificada.

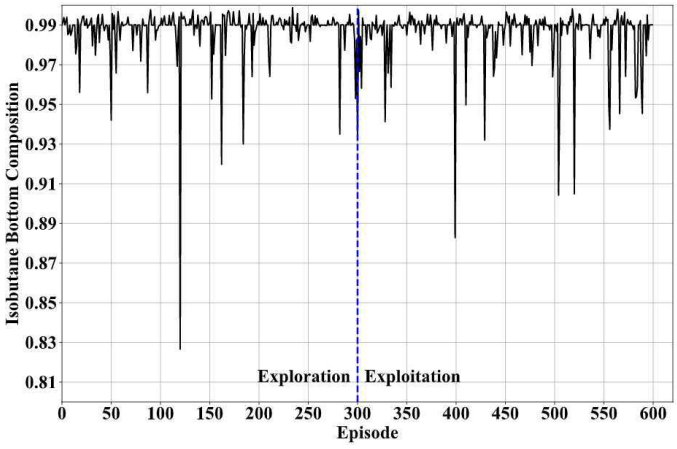
No entanto, na estratégia com AR fixo, as composições finais de IC_4 se mantiveram, em sua maioria, próximas à 0.99, porém, fora da tolerância requerida, provocando um comportamento ruidoso na avaliação conjunta dos episódios (Figura 4-7.a). Esse comportamento está relacionado com a taxa de manipulação fixa. A Figura 4-7.b indica que, as regiões próximas às composições de distúrbio de 0.35-0.38-0.40-0.42 apresentaram poucas falhas mesmo na fase de exploração, apontando uma boa aderência do AR fixo. Em contrapartida, as demais regiões apresentam falhas generalizadas durante todo o treinamento, sugerindo que valores diferentes de AR seriam necessários para correção do offset.

Em contrapartida, a estratégia utilizando Q-table com AR flexível, permite aproximações mais precisas para a região de tolerância de IC4 conforme apresentado na Figura 4-7.b. A análise da Figura 4-7.d apontam apenas 2 falhas após o episódio 550.

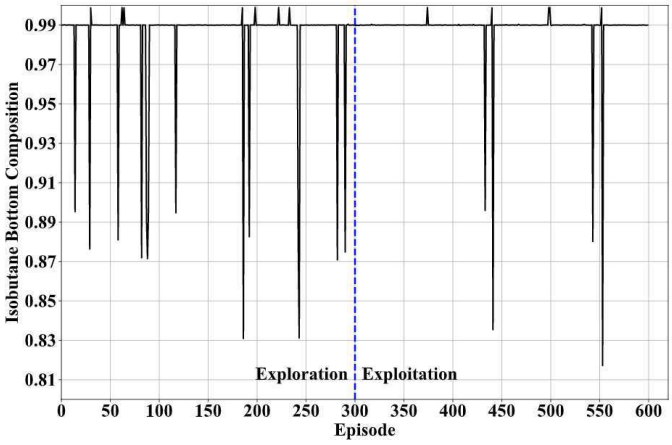
Nesta segunda estratégia, a ocorrência das falhas está primordialmente atrelada à estratificação do ambiente em um número finito de possibilidades. O deslocamento do agente dentro do ambiente para estados de composição de IC_4 abaixo de 0.9 culminam automaticamente na falha do episódio, uma vez que, o intervalo da Q-Table para essa variável é 0.9-1. Esse comportamento pode ser visualizado na maioria das falhas de treinamento da estratégia com AR variável (Figura 4-7.b).

A Figura 4-8 apresenta os percentuais de acerto cumulativo para as duas estratégias avaliadas, calculado a cada episódio com base na equação $\frac{\sum_1^i n_{Sucess}}{n_{Episodes}}$, onde i é o episódio atual, n_{sucess} é o número de episódios finalizados com sucesso até o momento e $n_{Episodes}$ é o número de episódios totais finalizados.

Para o Q-learning com AR fixo, o número acertos em relação a quantidade de episódios mantem uma relação 44.13 ±0.74 após a fase de exploração. Em contrapartida, os percentuais de acertos utilizando AR flexível, se mantem em 92.76 ±0.25 mesmo antes do término da fase de exploração.



(a)



(b)

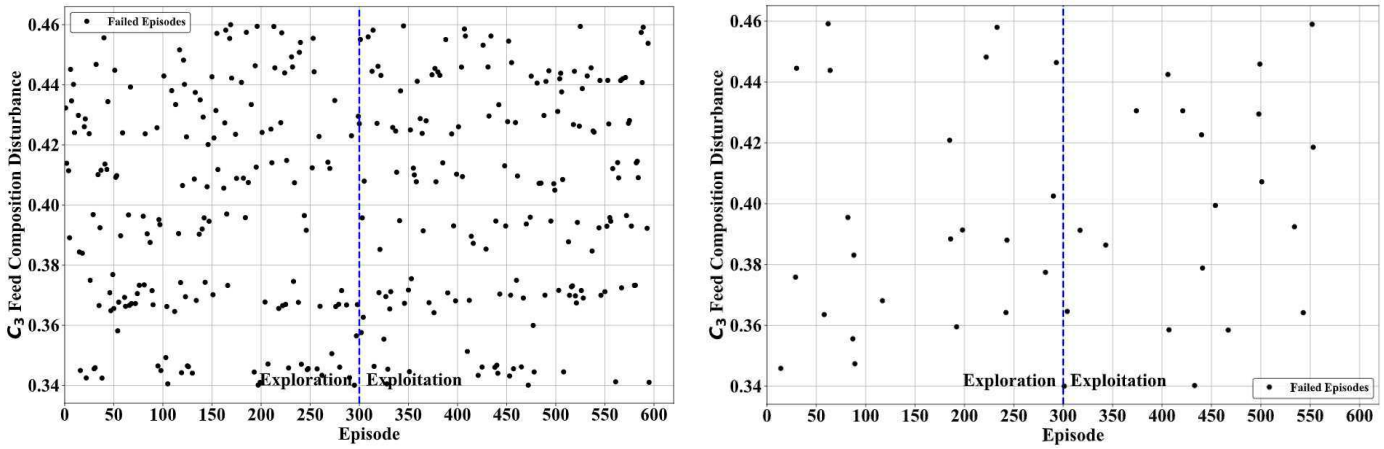


Figura 4-7 - IC_4 na base da coluna (4-7.a - 4-7.b) e dispersão dos distúrbios que provocaram falha (4-7.c – 4-7.d), para as estratégias com AR fixo e variável, respectivamente.

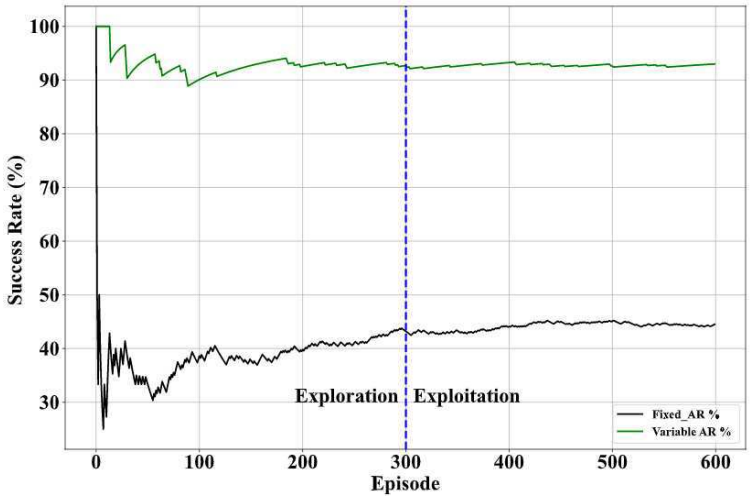


Figura 4-8 - Percentual de acerto cumulativo para as estratégias com AR fixo e variável.

4.2.Teste

Para a etapa de teste, foram avaliados 100 pontos de distúrbio na composição de propano da alimentação, gerados seguindo uma distribuição uniforme $U\sim$ (0.34-0.46). Os testes foram realizados com as Q-tables de cada estratégia obtidas após o último episódio.

A Tabela 4-6 apresenta o comparativo entre as duas estratégias avaliando o percentual de casos em que o modelo obteve sucesso e a mediana das tentativas realizadas para alcançar a meta nos episódios que foram bem-sucedidos.

Tabela 4-6 - Teste comparativo entre as estratégias utilizadas

Estratégia	%Acertos	Mediana das tentativas até o sucesso
1 (Ar fixo)	32	3
2 (Ar variável)	65	3

Observa-se que o número de ações realizadas foi igual nas duas estratégias avaliadas. Entretanto, os acertos da estratégia utilizando Q-table com AR variável foi aproximadamente o dobro em relação ao AR fixo.

Avaliando o histograma dos distúrbios realizados nos episódios bem-sucedidos (Figura 4-9), observa-se que, a estratégia 1 obteve sucesso nas regiões com baixa ocorrência de falhas no treinamento identificada na Figura 4-7.c, em decorrência da boa aderência do AR fixo para correção de offsets gerados nessas regiões de distúrbio.

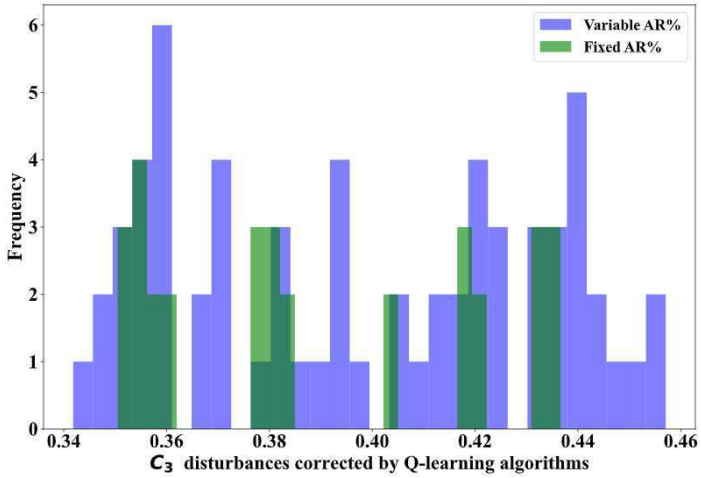


Figura 4-9 - Histograma dos distúrbios de C3 na alimentação que foram corrigidos

Em contrapartida, utilização de um AR variável, viabilizou a expansão das regiões com episódios bem-sucedidos, além do aumento dos acertos em algumas regiões atendidas pelo AR fixo.

Para avaliação comparativa da dinâmica das estratégias, analisou-se a resposta do sistema a um distúrbio no qual ambas atingiram a meta com o número representativo de tentativas apresentado na Tabela 4-6. O distúrbio ocorre em 1h de simulação e as atuações das estratégias de RL ocorrem a cada hora.

A avaliação da Figura 4-10.a confirma a limitação da malha de controle inferencial utilizada para manter a composição dentro da especificação requerida (0.99) após o distúrbio realizado na composição de propano (C_3) na corrente F1, de 0.4 para 0.4328.

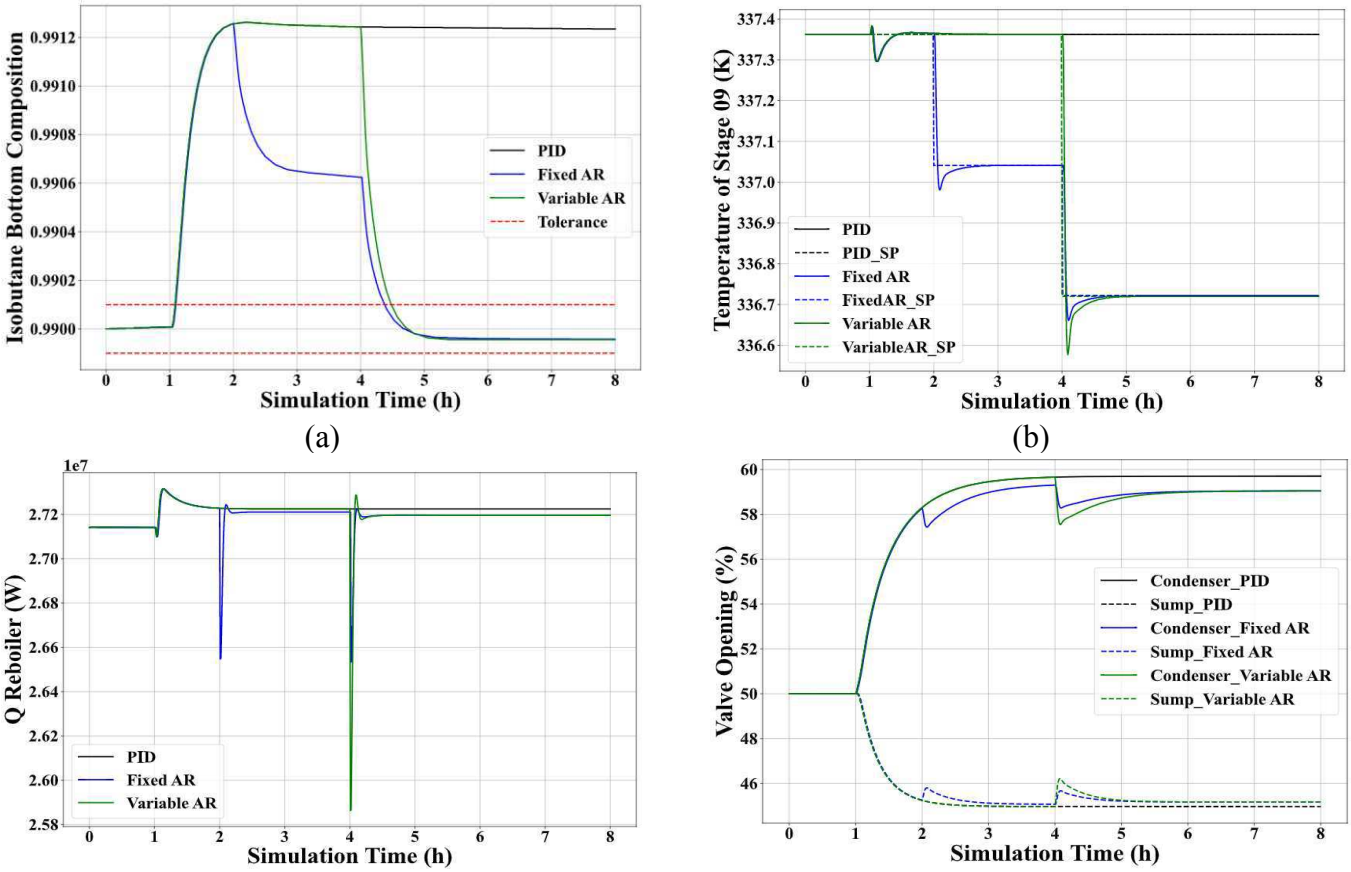


Figura 4-10 – Comportamento da [IC_4] na base da coluna (4-10.a); Temperatura do estágio 9 (4-10.b); QReb (4-10.c) e aberturas de válvula para o condensador e sump (4-10.d) para as estratégias avaliadas.

O controlador PID exerce a função de manter a temperatura do estágio 9 no setpoint especificado, conforme observado na Figura 4-10.b. Entretanto, perturbações na composição de propano em F1 podem exigir um novo setpoint de temperatura para a obtenção do produto de base com composição de 0.99 de IC_4. No cenário avaliado, o novo setpoint é de 336.72 K. Conforme esperado, esta mudança não é enxergada diretamente pelo controlador.

A incorporação dos algoritmos de *reinforcement learning* utilizando a Q-Table demonstram a capacidade de correção dos offsets utilizando as estratégias com AR fixo e variável, com tempos de retorno similares (5h) para a região de tolerância.

O tempo de retorno para a composição correta na base da coluna está relacionado com os valores de $Q(s_k,a_k)$ obtidos ao final do treinamento por cada estratégia. A Q-table utilizando AR fixo realizou duas alterações no setpoint intercaladas por uma ação de manutenção da temperatura de SP no tempo de simulação entre 3-4h. Em contrapartida, a estratégia utilizando AR variável manteve a temperatura no setpoint original até 4h de simulação, realizando apenas 1 ação de manipulação para correção do offset com AR = 1%.

Em relação aos níveis do sump e do condensador (Figura 4-10.d), observa-se que, em ambas as estratégias, os algoritmos não sugeriram alterações intermediárias que levassem o sistema para regiões próximas a saturação das válvulas.

5.Conclusão

O controle de composição em colunas de destilação é uma tarefa importante, geralmente realizada com controladores feedback do tipo PID. A utilização deste tipo de controlador pode não ser eficiente em todas as condições. Neste trabalho foram identificadas limitações atreladas a offsets na composição do produto provocados pela malha de controle inferencial.

A análise acoplada do controlador convencional (PID) com o algoritmo de Q-learning mostrou-se uma alternativa atraente para corrigir limitações de offset sem alterar a malha de controle principal da coluna. Além disso, a utilização de um ambiente dinâmico (Aspen Plus Dynamics) viabilizou a inclusão das principais variáveis de controle e de restrição durante o treinamento do modelo.

Os resultados apontam a limitação da malha de controle inferencial utilizada para manter a composição de IC_4 na base da coluna. Em contrapartida, a utilização conjunta do controlador PID + Q-learning com AR variável permitiu a manutenção da composição dentro da especificação em 65% dos cenários avaliados, um percentual duas vezes maior em comparação com a estratégia de AR fixo.

Embora os tempos de resposta das duas estratégias tenham sido próximos no cenário dinâmico avaliado, o algoritmo com AR fixo está limitado a atuações em regiões específicas em que a taxa de manipulação é suficiente para corrigir o offset. Neste sentido, a abordagem com AR flexível mostrou-se capaz de lidar com diferentes regiões de distúrbio, mostrando-se uma estratégia mais robusta para problemas transientes e não lineares.

Por fim, este trabalho apresentou um algoritmo procedural para treinamento de modelos de *reinforcement learning* voltado para ambientes dinâmicos que pode ser adaptado para diferentes processos e softwares de análise dinâmica.

6.Referências

Chao, K. C., & Seader, J. D. (1961). **A General Correlation of Vapor-Liquid Equilibria in Hydrocarbon Mixtures**. AIChE Journal, 7, 598. https://doi.org/10.1002/aic.690070414

Géron, A. (2017). **Hands-On Machine Learning with Scikit-Learn and TensorFlow** (1st ed.). O'Reilly Media, Inc.

Hou, Z. S., & Wang, Z. (2013). **From model-based control to data-driven control: Survey, classification and perspective.** Information Sciences, 235, 3–35. https://doi.org/10.1016/J.INS.2012.07.014

Hwangbo, S., & Sin, G. (2020). **Design of control framework based on deep reinforcement learning and Monte-Carlo sampling in downstream separation**. Computers & Chemical Engineering, 140, 106910. https://doi.org/10.1016/J.COMPCHEMENG.2020.106910

Kano, M., & Ogawa, M. (2009). **The State of the Art in Advanced Chemical Process Control in Japan**. IFAC Proceedings Volumes, 42(11), 10–25. https://doi.org/10.3182/20090712-4-TR-2008.00005

Kretchmar, R. M., Young, P. M., Anderson, C. W., Hittle, D. C., Anderson, M. L., & Delnero, C. C. (2001). **Robust reinforcement learning control with static and dynamic stability.** International Journal of Robust and Nonlinear Control, 11(15), 1469–1500. https://doi.org/10.1002/rnc.670

Luyben, W. L. (2006). **Distillation design and control using Aspen simulation**. Wiley-Interscience.

Martinez, E. C. (2000). **Batch process modeling for optimization using reinforcement learning.** Computers & Chemical Engineering, 24(2–7), 1187–1193. https://doi.org/10.1016/S0098-1354(00)00354-9

Mendel, J. M., & McLaren, R. W. (1970). 8 **Reinforcement-Learning Control and Pattern Recognition Systems. Mathematics in Science and Engineering**, 66(C), 287–318. https://doi.org/10.1016/S0076-5392(08)60497-X

Pellegrini, J., & Wainer, J. (2007). **Processos de Decisão de Markov: um tutorial. Revista de Informática Teórica e Aplicada**, 14(2), 133–179. https://doi.org/10.22456/2175-2745.5694

Radac, M. B., & Precup, R. E. (2019). **Data-driven model-free tracking reinforcement learning control with VRFT-based Adaptive Actor-Critic**. Applied Sciences (Switzerland), 9(9). https://doi.org/10.3390/app9091807

Ramos, W. B., Figueirêdo, M. F., Brito, K. D., Ciannella, S., Vasconcelos, L. G. S., & Brito, R. P. (2016). **Effect of Solvent Content and Heat Integration on the Controllability of Extractive Distillation Process for Anhydrous Ethanol Production**. Industrial and

Engineering Chemistry Research, 55(43), 11315–11328. https://doi.org/10.1021/acs.iecr.6b03515

Selvi, D., Piga, D., & Bemporad, A. (2018). **Towards direct data-driven model-free design of optimal controllers**. In 2018 European Control Conference (ECC). https://doi.org/10.0/Linux-x86_64

Spielberg, S., Tulsyan, A., Lawrence, N. P., Loewen, P. D., & Bhushan Gopaluni, R. (2019). **Toward self-driving processes: A deep reinforcement learning approach to control**. AIChE Journal, 65(10). https://doi.org/10.1002/aic.16689

Sutton, R. S., & Barto, A. G. (2018). **Reinforcement Learning: An Introduction**. The MIT Press.

Syafiie, S., Tadeo, F., & Martinez, E. (2007). **Learning to control pH processes at multiple time scales: Performance assessment in a laboratory plant.** Chemical Product and Process Modeling, 2(1). https://doi.org/10.2202/1934-2659.1024

Syafiie, S., Tadeo, F., & Martinez, E. (2008). **Model-Free Learning Control of Chemical Processes**. In C. Weber, M. Elshaw, & N. M. Mayer (Eds.), Reinforcement Learning. IntechOpen. https://doi.org/10.5772/5287

Tulsyan, A., Garvin, C., & Ündey, C. (2018). **Advances in industrial biopharmaceutical batch process monitoring: Machine-learning methods for small data problems**. Biotechnology and Bioengineering, 115(8), 1915–1924. https://doi.org/10.1002/bit.26605

Tulsyan, A., Garvin, C., & Undey, C. (2018). **Machine-learning for biopharmaceutical batch process monitoring with limited data.** 51(18), 126–131. https://doi.org/10.1016/j.ifacol.2018.09.287

Tulsyan, A., Garvin, C., & Undey, C. (2019). **Industrial batch process monitoring with limited data. Journal of Process Control**, 77, 114–133. https://doi.org/10.1016/j.jprocont.2019.03.002

Tulsyan, A., Khare, S. R., Huang, B., Gopaluni, R. B., & Forbes, J. F. (2013). **Bayesian identification of non-linear state-space models: Part I- Input design**. IFAC Proceedings Volumes, 46(32), 774–779. https://doi.org/10.3182/20131218-3-IN-2045.00105

Wang, H., & Ge, Z. (2020). **Internal environment controlling algorithm of DMF distillation column based on reinforcement learning**. 2020 IEEE 9th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), 1479–1483. https://doi.org/10.1109/ITAIC49862.2020.9338759

Wilson, J. A., & Martinez, E. C. (1997). **Neuro-fuzzy modeling and control of a batch process involving simultaneous reaction and distillation**. Computers & Chemical Engineering, 21(SUPPL.1), S1233–S1238. https://doi.org/10.1016/S0098-1354(97)87671-5

Capítulo 5 Aplicação Deep Q-Network

Data-driven control system for distillation columns using Deep reinforcement learning.

Gladson Euler, Karoline Brito, Wagner Brandão, Luís Gonzaga, Romildo Brito

Federal University of Campina Grande, Chemical Engineering Department, Campus Universitario, Campina Grande, PB, Brazil, 58109-970

Abstract:

Para garantir a qualidade do produto e o desempenho do processo em colunas de destilação, o controle preciso da composição é essencial. No entanto, os controladores PID comumente usados podem ter limitações dependendo da malha de controle utilizada. Embora os estudos recentes apresentem estratégias promissoras utilizando *reinforcement learning* (RL) para controle de colunas, a modelagem do processo é simplificada, negligenciando aspectos importantes na dinâmica de controle. Este trabalho propôs a criação de um algoritmo de treinamento dinâmico integrado utilizando Python e Aspen Plus Dynamics para avaliação de diferentes modelos de RL. O estudo avaliou duas metodologias de RL, Q-Learning e Deep Q-Network (DQN), ambas acopladas a um controlador PID. A partir da comunicação do tipo COM foi possível rastrear as variáveis observáveis no software Aspen Plus Dynamics e realizar alterações no modelo dinâmico a partir do Python ao longo de toda a simulação. A comparação entre as metodologias DQN, Q-Learning e o controle regulatório (PID) confirmou a limitação da malha de controle inferencial utilizada para manter a composição dentro da especificação de isobutano (IC_4) na base da coluna. Em contrapartida, a utilização conjunta do PID com o controle DQN permitiu manter a composição dentro da especificação em 96% dos cenários avaliados, com um com um IAE 52,9 % menor em comparação com o Q-Learning. O ambiente dinâmico utilizado para treinamento e validação das propostas de controle, o Aspen Plus Dynamics, manteve a complexidade, não-linearidade e aspecto transiente do processo, permitindo a inclusão das principais variáveis observáveis (carga térmica do reboiler, temperatura do prato sensível e a composição do produto) e restrições (saturação das válvulas do sump e condensador), tornando as propostas de controle avaliadas mais viáveis para a aplicação prática.

Keywords: Data-driven control; Aspen Plus Dynamics, Deep Q-Network, Q-learning, Generic training algorithm.

1.Introdução

O controle de composição em colunas de destilação de alta pureza desempenha um papel crucial na viabilidade econômica desses sistemas. A implementação de uma estrutura de controle adequada e a otimização do processo têm um impacto direto no desempenho da coluna, nos custos associados e na viabilidade global do processo (RAMOS et al., 2016).

Os controladores clássicos do tipo PID (Proporcional, Integral e Derivativo) são amplamente utilizados em colunas de destilação para assegurar a pureza dos produtos de base e topo. No entanto, devido à natureza complexa, não linear e multivariável desse processo, a estrutura de controle construída utilizando esse tipo de controlador, pode afetar negativamente desempenho do equipamento (TULSYAN; GARVIN; UNDEY, 2018, 2018, 2019).

Alternativamente, métodos de controle avançado podem ser utilizados. Entretanto, métodos de controle baseados em modelos matemáticos, fundamentados em princípios físico-químicos (first-principles) ou obtidos por meio de identificação de sistemas, podem ser caros e demorados (HOU; WANG, 2013). Neste caso, estratégias de controle baseadas em dados (data-driven) surgem como alternativa aos controladores regulatórios clássicos ou baseados em modelos.

Dentre elas, os algoritmos de *reinforcement learning* (RL) destacam-se por serem baseados em uma política de interação com o ambiente, não exigindo um conjunto inicial de dados de treinamento para obter o modelo do processo. Isso permite que a lei de controle se adapte e mude com base em sua experiência, tornando-se auto-organizada (MENDEL; MCLAREN, 1970). Portanto, o RL é considerado um poderoso método data-driven que pode fornecer a política ótima para resolver problemas de controle (RADAC; PRECUP, 2019).

Embora seja uma área mais recente em relação ao aprendizado supervisionado e não supervisionado, os algoritmos de RL têm sido amplamente utilizados em processos industriais, demonstrando resultados promissores em diversas aplicações, por exemplo: 1) Controle do processo de laminação de tiras (DENG et al., 2022); 2) Controle de reatores contínuo (DUTTA e UPRETI, 2023), simulação de hidrogenação seletiva de acetileno (ALHAZMI e SARATHY, 2023), dentre outros.

A modelagem do ambiente onde o algoritmo de RL irá aprender, é uma etapa crucial para obtenção de uma estratégia de controle robusta e viável. A Tabela 5-1 apresenta a forma com a qual os trabalhos recentes abordam a modelagem do processo de destilação em colunas.

Tabela 5-1 - Estratégia utilizada para modelagem do ambiente (coluna de destilação)

Trabalho	Ano da publicação	Regime	Modelagem do ambiente
Kretchmar	2001	Transiente	Modelos linearizados a partir de funções de transferência simplificadas (MIMO)
Spielberg	2019		
Hwangbo; Sin	2020	Estacionário	Simuladores de processos químicos (Aspen Plus)
Wang; GE	2020	Transiente	Fórmulas empíricas simplificadas para aproximar a relação de controle.
Patel	2023	Transiente	Equações matemáticas simplificadas que não consideram mudança de cenários.

De uma maneira geral, os estudos concentram-se no desenvolvimento de algoritmos de reinforcement legarning para o controle, porém, modelam o ambiente (coluna de destilação) de maneira simplificada.

Equacionamentos matemáticos simplificados podem negligenciar a abertura de válvulas, comportamento transiente e respostas a perturbações, limitando a compreensão das operações reais da coluna. Da mesma forma, simulações em regime estacionário podem não representar adequadamente as variações nas condições operacionais, tornando-as menos adequadas para otimizar estratégias de controle e identificar pontos críticos de operação.

Nesse contexto, este trabalho propõe o desenvolvimento de um algoritmo de treinamento dinâmico integrado utilizando Python o Aspen Plus Dynamics adaptável para avaliação de diferentes modelos de reinforcement learning. Para avaliação do algoritmo, dois modelos de RL simplificados utilizando Q-learning e Deep Q-Network (DQN) foram avaliados.

O Aspen Plus Dynamics é uma ferramenta confiável que permite a construção e obtenção de informações robustas sobre o modelo dinâmico da coluna de destilação. A manutenção da complexidade, não-linearidade e aspecto transiente do processo aumenta a viabilidade prática das propostas de controle avaliadas.

2.*Reinforcement learning* – Conceitos

Um problema de RL pode ser formulado a partir de um processo de decisão de Markov (MDP), onde a ação realizada em um estado depende apenas do estado atual e da ação tomada, sem levar em consideração estados anteriores (PELLEGRINI; WAINER, 2007). Os principais

elementos que compõem um problema de *reinforcement learning* são apresentados na Figura 5-1.

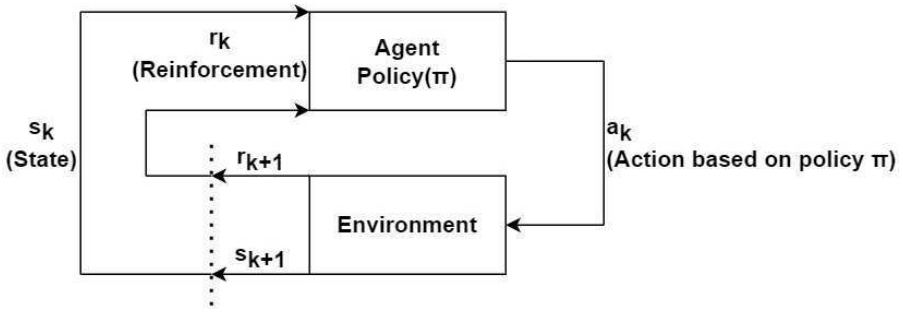


Figura 5-1 - Formulação geral para um problema de RL.
Fonte: Adaptado de SUTTON e BARTO, 2018.

Nesse tipo de aprendizado, o agente aprende de forma interativa a melhor maneira de agir em um determinado ambiente, baseando-se em reforços positivos ou negativos recebidos ao longo do tempo.

O agente começa em um estado s_k e decide, com base em uma regra de decisão, qual ação a_k deve ser executada, selecionando-a de um conjunto de ações possíveis. A ação a_k afeta o ambiente, resultando na transição para um novo estado observável s_{k+1}. Após essa transição, o agente é reforçado por meio de uma recompensa ou penalidade r_k, e o processo se repete.

A política (π) é o conjunto de regras de decisão adotadas pelo agente. Em geral, busca-se uma política que maximize as recompensas cumulativas, com base na Eq. (5-1), onde γ é o fator de desconto que reflete a importância da recompensa futura em relação à recompensa imediata.

$$V^{\pi}(s_k) = r_k + \gamma r_{k+1} + \gamma^2 r_{k+2} + \cdots; \text{ onde } \gamma \in [0,1] \qquad (5\text{-}1)$$

No contexto deste trabalho, foram avaliadas duas metodologias de RL, Q-Learning e Deep Q-Network (DQN).

Q-learning

A função estado-valor relacionada ao Q-learning é similar à apresentada na Eq. (5-1) e pode ser pela Eq. (5-2), onde a_k é a ação selecionada no conjunto de ações A e executada no instante k. Considerando o objetivo de maximizar a recompensa futura (s_{k+1}), a Eq. (5-2) pode ser reescrita conforme Eq. (5-3). Após a tomada de ação e a avaliação do novo estado

observável s_{k+1}, a atualização do valor de $Q(s_k, a_k)$ é realizado a partir da Eq. (5-4), sendo α a taxa de aprendizado.

$$Q(s_k, a_k) \leftarrow r_k + \gamma V(s_{k+1}) \quad (5\text{-}2)$$

$$Q(s_k, a_k) \leftarrow r_k + \gamma \, maxQ_{a \in A}(s_{k+1}, a) \quad (5\text{-}3)$$

$$Q(s_k, a_k) \leftarrow Q(s_k, a_k) + \alpha\big(r_k + \gamma . maxQ_{a \in A}(s_{k+1}, a) - Q(s_k, a_k)\big) \quad (5\text{-}4)$$

O mapeamento do ambiente é realizado utilizando uma estrutura de dados chamada Q-table. A Tabela 5-2 apresenta a Q-table para um sistema genérico com n estados e "z" ações possíveis. Conforme o agente transita pelos estados do ambiente, os valores Q para cada par (s, a) são atualizados usando a Eq. (5-4).

Tabela 5-2 - Q-table para um sistema genérico

State	**Ação 1**	**Ação 2**	⋯	**Ação z**
s_1	$Q(s_1, \text{Ação}_1)$	$Q(s_1, \text{Ação}_2)$	⋯	$Q(s_1, \text{Ação}_z)$
s_2	$Q(s_2, \text{Ação}_1)$	$Q(s_2, \text{Ação}_2)$	⋯	$Q(s_2, \text{Ação}_z)$
s_3	$Q(s_3, \text{Ação}_1)$	$Q(s_3, \text{Ação}_2)$	⋯	$Q(s_3, \text{Ação}_z)$
⋮	⋮	⋮	⋮ ⋯	⋮
s_n	$Q(s_n, \text{Ação}_1)$	$Q(s_n, \text{Ação}_2)$	⋯	$Q(s_n, \text{Ação}_z)$

Deep Q-Network (DQN)

A DQN é uma extensão do Q-learning e utiliza uma rede neural para mapear o ambiente no lugar da Q-table. Na metodologia utilizando DQN, a taxa de aprendizado α pode ser omitida da Eq. (5-4) uma vez que ela é definida no algoritmo de otimização dos pesos sinápticos presente na rede neural. Neste sentido, a função estado-valor para a DQN pode ser expressa pela Eq. (5-5) enquanto a função de perda da rede neural a ser minimizada pode ser descrita conforme Eq. (5-6).

$$Q(s_k, a_k) \leftarrow r_k + \gamma . maxQ_{a \in A}(s_{k+1}, a) \quad (5\text{-}5)$$

$$loss\ function = \left[\left(r_k + \gamma . max_{a \in A}\ \hat{Q}(s_{k+1}, a) - Q(s_{k+1}, a)\right)^2\right] \quad (5\text{-}6)$$

3.Abordagem do problema

Conforme descrito anteriormente, o conceito de *reinforcement learning* envolve uma série de elementos. A maneira como esses elementos se encaixam no controle da coluna de destilação são explicados em detalhes a seguir.

3.1. Ambiente

O fluxograma da Figura 5-2 apresenta o processo de destilação de uma mistura de propano (C_3) - isobutano (IC_4), proposto por (Luyben, 2006). A coluna de destilação foi modelada a partir do bloco RadFrac do software Aspen Plus, utilizando o modelo termodinâmico CHAO-SEADER. Após o dimensionamento do equipamento e definição das quedas de pressão nas válvulas de controle, a coluna foi exportada para o Aspen Plus Dynamics.

O modelo CHAO-SEADER foi selecionado devido à sua comprovada capacidade para lidar com equilíbrio líquido-vapor (VLE) em sistemas de hidrocarbonetos em diversas faixas de temperatura e pressão (CHAO e SEADER, 1961).

A operação da coluna ocorre sob pressão devido à baixa volatilidade relativa entre os componentes, e os controles regulatórios estão relacionados aos níveis do condensador e do sump, juntamente com o controle de pressão da coluna. Um controle de razão (R/F) foi implementado para manter a proporção mássica entre a alimentação e o refluxo.

O propano é retirado no primeiro estágio da coluna com uma pureza de 98%. O isobutano é retirado na base da coluna com pureza especificada de 99%. A malha de controle utilizada para manter a composição de IC_4 na base da coluna dentro da especificação é do tipo inferencial por meio do controle de temperatura do prato sensível (estágio 9) utilizando a carga térmica do reboiler como variável manipulada.

As especificações técnicas do sistema proposto bem como os parâmetros e variáveis dos controladores utilizados estão descritos nas Tabela 5-3 e Tabela 5-4 respectivamente. Vale destacar que os controladores foram sintonizados utilizando o método Tyreus-Luyben.

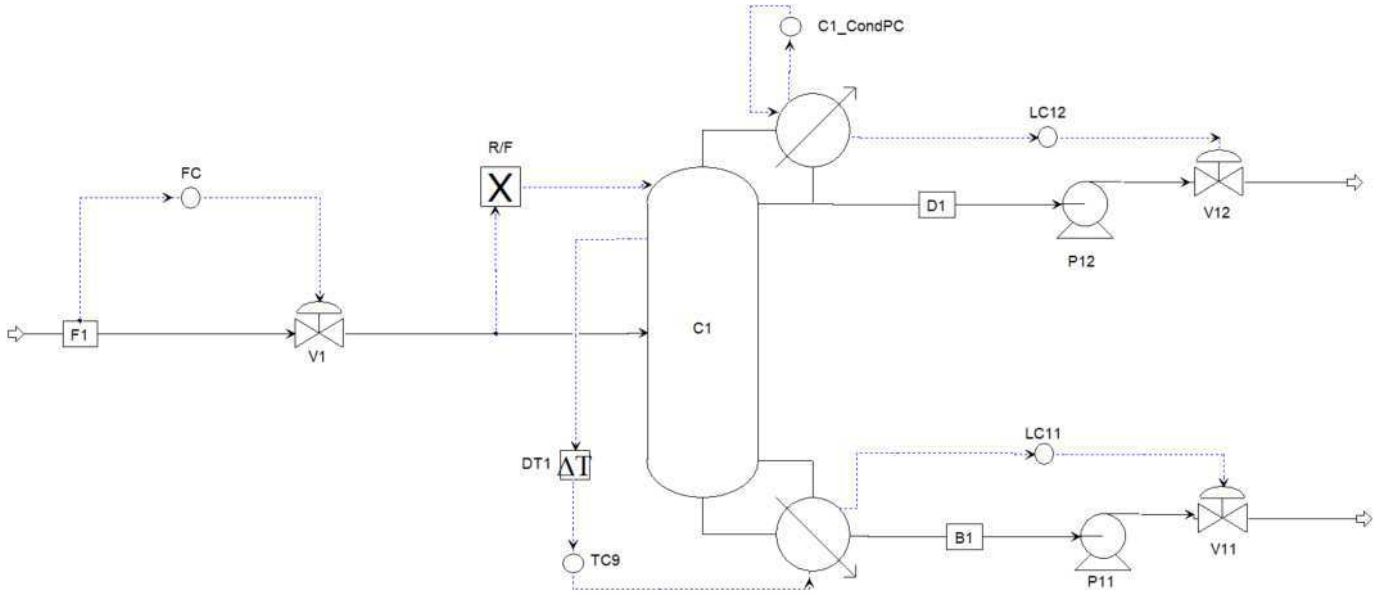


Figura 5-2 - Fluxograma do processo de destilação (C_3-IC_4).

Tabela 5-3 - Especificações de processo.

Variable	Valor	Unidade
Taxa de alimentação (F1)	1	kmol/s
Temperatura (F1)	322	K
Pressão (F1)	2.0265 x 10^6	Pa
Composição C_3 (F1)	0.4	-
Composição IC_4 (F1)	0.6	-
Número de estágios (C1)	32	
Taxa de destilado (C1)	0.4	kmol/s
Refluxo (C1)	3.46	-
Estágio de alimentação (C1)	14	-
Pressão no condensador (C1)	1.7022 x 10^6	Pa
Queda de pressão no estágio (C1)	689.01	Pa
Nível no condensador – Setpoint (LC12)	5.10	m
Nível no sump setpoint (LC11)	6.35	m
Pressão no condensador Setpoint (C1_CondPC)	16.8	atm
Temperatura do estágio 9° e Setpoint (TC9)	337.36	K
R/F	1.18	-

Tabela 5-4 - Parâmetros e variáveis dos controladores utilizados.

Controlador	Variável Controlada	Variável Manipulada	Tipo de Ação	K_P	K_I	K_D
FC	Vazão de alimentação	Abertura da válvula (%)	Reversa	0.5	0.3	-
LC12	Nível Condensador		Direta	2	-	-
LC11	Nível Sump		Direta	2	-	-
C1_CondPC	Pressão da coluna	Carga térmica – Condensador	Reversa	20	12	-
TC9	Temperatura do Estágio 9	Carga térmica - Reboiler	Reversa	1.84	9.9	-

3.2.Tarefa

No contexto do processo de destilação proposto, a tarefa do algoritmo pode ser formulada conforme apresentado na Tabela 5-5:

Tabela 5-5 - Objetivos e restrições que norteiam a atuação do agente

Objetivo	$Encontrar\ \ s\|\theta\ com\ Desvio \leq Tol;$	(5-7)
	$Tol = 1\ x\ 10^{-4}$	(5-8)
	$Desvio = \|[IC_4] - [IC_4]_{Alvo}\|;\ [IC_4]_{Alvo} = 0.99$	(5-9)
Restrições	$20\% < SUMP_{\%} < 80\%$	(5-10)
	$20\% < \mathrm{CD}_{\%} < 80\%$	(5-11)

Onde s representa as condições operacionais que compõem as variáveis observáveis do ambiente, sendo $s \in$ {Composição de Isopropano ([IC_4]); Abertura da válvula - Sump ($SUMP_{\%}$), Abertura da válvula - condensador ($CD_{\%}$), Carga térmica do Reboiler (Q_{REB})}.

O objetivo das restrições atreladas à $SUMP_{\%}$ e $CD_{\%}$ é evitar que, ao longo do processo de aprendizado, o algoritmo opte por regiões onde as aberturas das válvulas estejam próximas da saturação. Na prática, isso pode levar a problemas de estabilidade do sistema e agarramento de válvula.

Os hiperparâmetros $\theta \in \{\alpha, \gamma\}$. foram definidos empiricamente e mantidos fixos em 0.01 e 0.99, respectivamente, para as metodologias avaliadas neste trabalho.

3.3. Reforço

Os reforços determinam que tipo de retorno o ambiente entregará para o agente em decorrência da ação tomada e do novo estado s_{k+1} obtido. Para o problema proposto, as funções de reforço são descritas pela Eq. (5-12).

$$R_s^a = \begin{cases} 0; se\ k = 1; \\ 10, se\ a\ tarefa\ (Eqs.\ (5\text{-}7) - (5\text{-}9))\ foi\ concluida; Status: Concluído \\ R_s^a - 1, se\ \ k \neq 1\ ;\ k < k_{max}\ e\ o\ Desvio > Tol; Status: Não\ Concluído \\ -k_{max}, se\ as\ restrições\ (Eqs.\ (5\text{-}10)\ ou\ (5\text{-}11))\ não\ forem\ atendidas\ ou\ k = k_{max}; Status: Falha \end{cases} \quad (5\text{-}12)$$

Os reforços foram estruturados de forma a garantir uma bonificação para os casos em que o algoritmo alcança um estado s_{k+1} que satisfaz a tarefa especificada, e fornecendo uma penalidade para cada step k onde algoritmo não conclui a tarefa. A violação das restrições do sistema é penalizada com o valor máximo possível. Isso é realizado para garantir que o agente evite tomar ações que possam convergir cenários onde críticos como a saturação das válvulas.

Tendo em vista que os valores dos reforços serão utilizados para atualização dos Q-values, a penalização a cada step cria uma espécie de superfície. Desta forma, ao final do treinamento, o algoritmo escolherá as ações que maximizam os valores de Q, a partir da escolha do caminho mais rápido para realização da tarefa, independente do estado inicial do sistema.

3.4. Ações

No presente trabalho, a mudança do estado s_k para s_{k+1} ocorre a partir da alteração do setpoint do controlador referente a temperatura do prato sensível (estágio 9). A escolha do estágio 9 para controle indireto de composição do produto a partir da temperatura está relacionada com o perfil térmico da coluna.

A Figura 5-3 apresenta o perfil térmico da coluna e a diferença de temperatura entre o estágio atual e o posterior. Observa-se que o estágio 9 está na região mais sensível em termos de temperatura, apresentando a região de maior inclinação no perfil em conjunto com a maior variação de temperatura observada.

A Figura 5-4 mostra a variação da composição de IC_4 com a alteração da temperatura do prato sensível. Desta forma, fica evidente a necessidade de alteração do setpoint de temperatura para corrigir desvios na composição de IC_4 na base da coluna uma vez que essa necessidade não é enxergada diretamente pelo controlador devido à malha de controle inferencial.

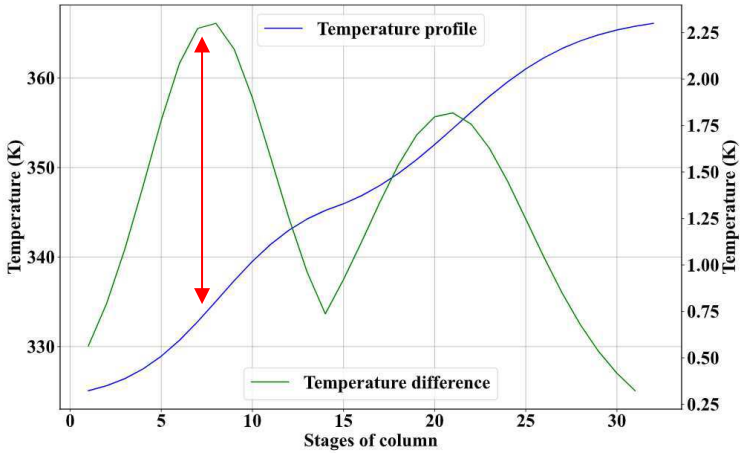


Figura 5-3 - Perfil e variação de temperatura entre os estágios da Coluna

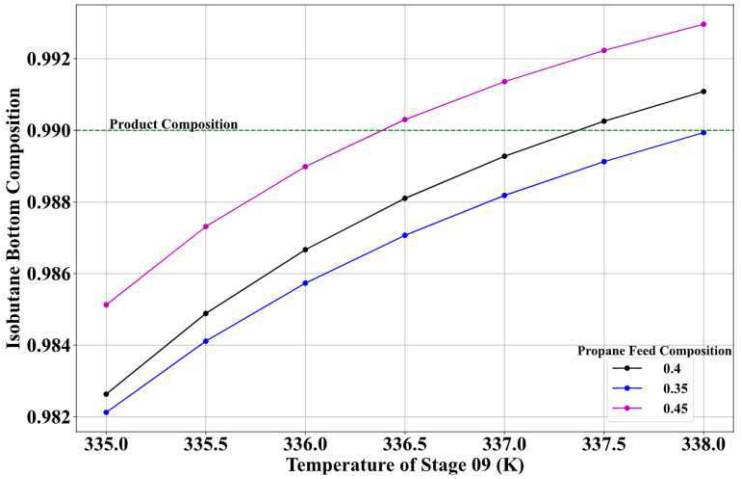


Figura 5-4 - Variação de IC_4 na base da coluna com a alteração da temperatura do estágio 9 para diferentes distúrbios.

Existem três ações possíveis, formuladas conforme apresentado na Eq. (5-13), ou seja, é possível aumentar/reduzir a temperatura de set-point (T_{SP}) à uma taxa AR ou não alterar a temperatura.

Neste trabalho, optou-se por uma AR variável, para a Q-table e DQN, conforme apresentado na Eq. (5-14). Desta forma, é possível realizar manipulações de até 5% na temperatura do estágio 9 da coluna com base no desvio entre a composição de isopropano atual e a especificada para a corrente de base da coluna.

$$A = \begin{cases} T_{SP}(1 - AR) \\ T_{SP} \\ T_{SP}(1 + AR) \end{cases} \tag{5-13}$$

$$AR = \begin{cases} 0.1\%; 1\ x\ 10^{-4} \leq Desvio \leq 5\ x\ 10^{-4} \\ 0.5\%;\ 5\ x\ 10^{-4} \leq Desvio \leq 1\ x\ 10^{-3} \\ 1\%; 1\ x\ 10^{-3} \leq Desvio \leq 5\ x\ 10^{-3} \\ 5\%;\ Desvio > 5\ x\ 10^{-3} \end{cases} \tag{5-14}$$

A utilização combinada das Eq. (5-13) e (5-14), permite que o agente se desloque em um ambiente contínuo por meio de ações discretas, com flexibilidade na capacidade de atuação do sistema com base na proximidade da meta. A escolha por uma forma simplificada para compor as ações do algoritmo foi realizada visando reduzir a complexidade do problema de RL.

4.Projeto do Algoritmo; Política

A coluna de destilação é um ambiente dinâmico modelado no Aspen Plus Dynamics. Neste sentido, as metodologias avaliadas, neste caso Q-learning e DQN precisam estar integradas à um algoritmo recursivo que avalie os estados e tome as ações no decorrer da simulação. A Figura 5-5 apresenta o algoritmo construído em Python que integra as metodologias de RL e o ambiente dinâmico na etapa de treinamento.

Neste caso, o agente é o próprio algoritmo, responsável por tomar as decisões com base nas informações recebidas do ambiente, aprendendo com as experiências para alcançar a tarefa especificada. A maneira como o agente interage com o ambiente e atualiza os valores da Q-table e dos pesos sinápticos na DQN ao longo dos episódios de treinamento é apresentada a seguir.

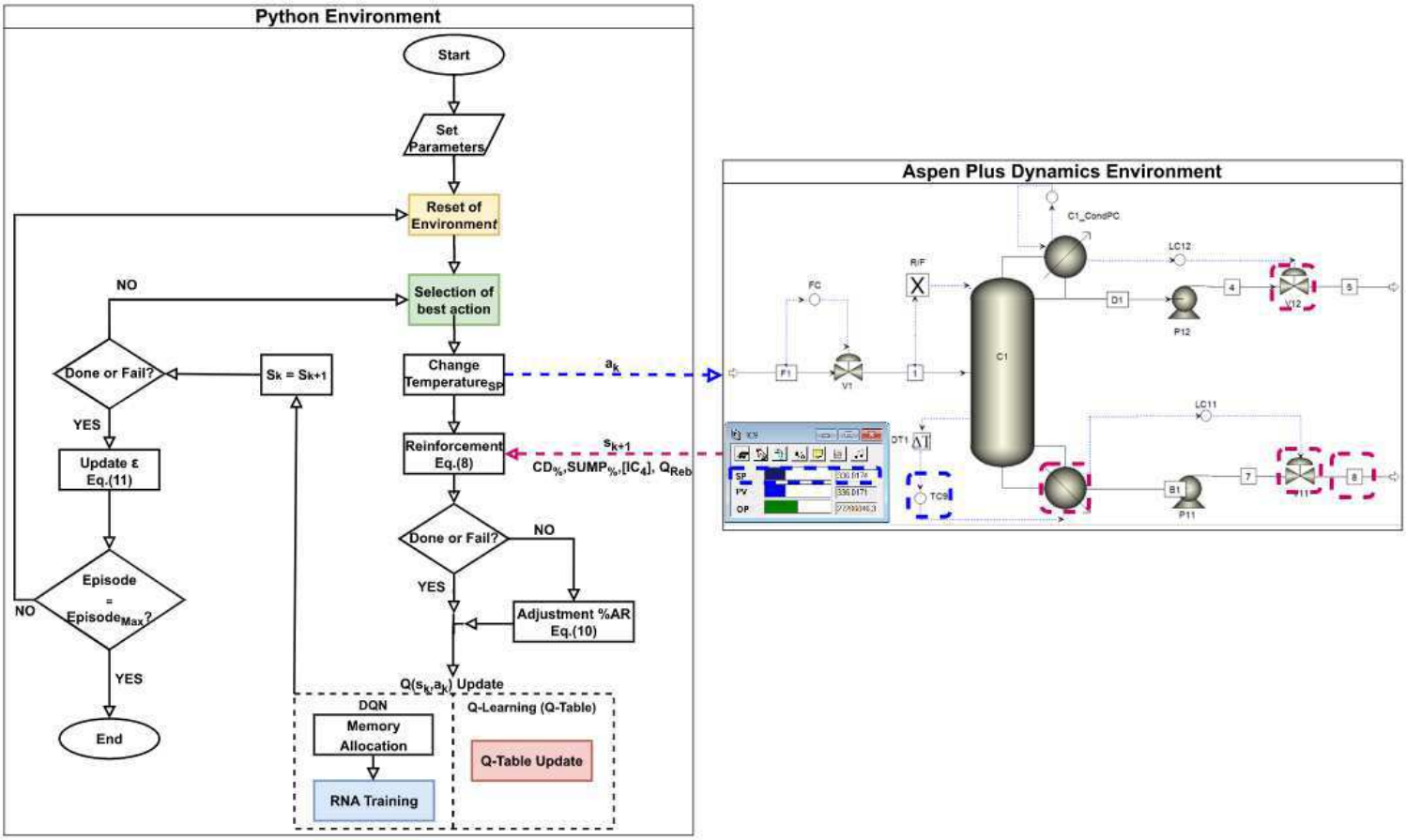


Figura 5-5 - Fluxograma geral do algoritmo de treinamento

Inicialmente os parâmetros gerais do algoritmo, Q-table e da DQN são definidos na etapa "***Set Parameters***" com base na Tabela 5-6, Tabela 5-7 e Tabela 5-8, respectivamente. Na Tabela 5-6, os números de episódios e passos foram definidos empiricamente, enquanto os valores iniciais dos estados observáveis estão relacionados com o estado estacionário do sistema. Os distúrbios foram definidos considerando um range de $\pm$15% em relação a condição nominal da composição de propano na alimentação (0.4).

Tabela 5-6 - Parâmetros gerais do algoritmo

Parâmetros	Valor	Unidade
Episódios	600	-
Passos	150	-
Estado Inicial [IC_4]	0.99	-
Estado Inicial SUMP%	50	%
Estado Inicial CD%	50	%
Estado Inicial Q_{REB}	2.71 x 10^7	W
Distúrbio na Alimentação	[C_3] seguindo $U\sim$ (0.34 – 0.46)	-

Tabela 5-7 - Parâmetros utilizados para construção da Q-table

Parâmetros	Valor	Unidade
S_{MIN}	[0.9; 0; 0; 2.08 x 10^7]	[-, %, %, W]
S_{MAX}	[1; 100; 100; 2.92 x 10^7]	[-, %, %, W]
Número de linhas da tabela	200	-
Número de colunas da tabela	3	-
Valores Iniciais $Q(s_k,a_k)$	0	-

Tabela 5-8 - Topologia da RNA utilizada.

Elemento	Valor
Nº de Neurônios Camada de entrada	4
Nº de Neurônios camada oculta 1	24
Nº de Neurônios camada oculta 2	48
Nº de Neurônios camada de saída	3
Função de ativação camada oculta 1	ReLU
Função de ativação camada oculta 2	ReLU
Função de ativação camada de saída	Linear
Otimizador de pesos	ADAM
Épocas	1

Para o Q-learning os limites mínimos e máximos das variáveis observáveis apresentados na Tabela 5-7 foram definidos para o zoneamento do ambiente. O número de divisões (linhas) da Q-table foi definido com base na memória consumida pelo computador, enquanto o número de colunas é igual ao número de ações que o agente dispõe para se locomover no ambiente.

Na metodologia utilizando a DQN, a topologia da rede neural (Figura 5-6) proposta para substituição da Q-table consiste em uma rede do tipo *Multi Layer Perceptron* (MLP) com 2 camadas ocultas. A entrada da rede consiste nas variáveis observáveis que compõem o estado s_k enquanto a saída da rede tem a função de estimar os valores de $Q(s_k, a)$ para cada ação possível do agente. A Figura 5-6 apresenta a representação gráfica simplificada da arquitetura da rede neural utilizada.

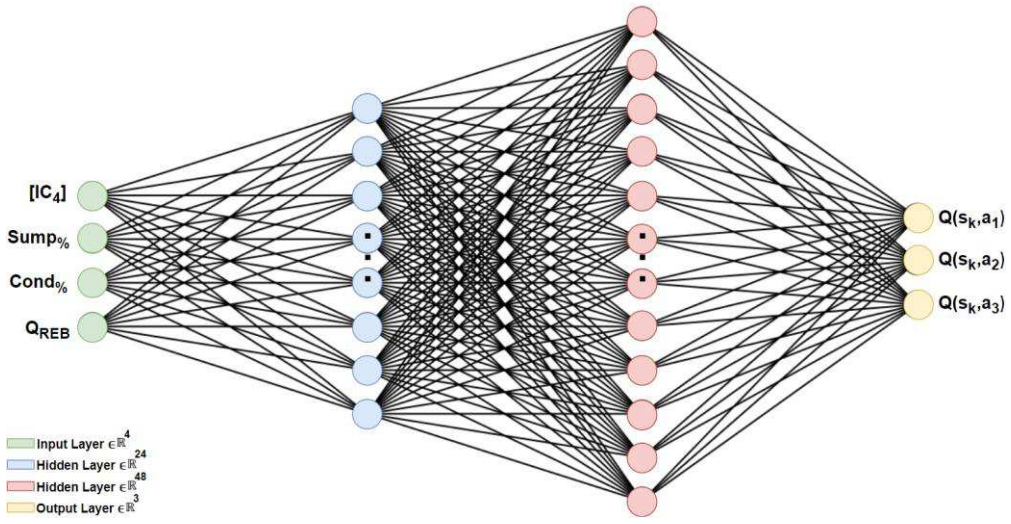


Figura 5-6 - Arquitetura simplificada da RNA utilizada na metodologia DQN

Após a definição dos principais parâmetros, o algoritmo segue para a etapa de "***Integration Python-Aspen***" responsável por realizar a comunicação Python-Aspen Plus Dynamics através da tecnologia de Component Object Model (COM) desenvolvida pela Microsoft. A comunicação é realizada por meio da biblioteca *win32com.client* do python através do comando *ACMObj = win32.GetObject(r"directory\Simulation.dynf").*

Ainda nessa etapa, é preciso buscar o caminho de cada variável dentro da simulação Aspen Plus Dynamics para que seja possível monitorar e atuar na simulação. A Figura 5-7 apresenta em destaque o nome da variável de interesse para a abertura de válvula (%) do condensador, por exemplo. A Tabela 5-9 apresenta o caminho de todas as variáveis utilizadas.

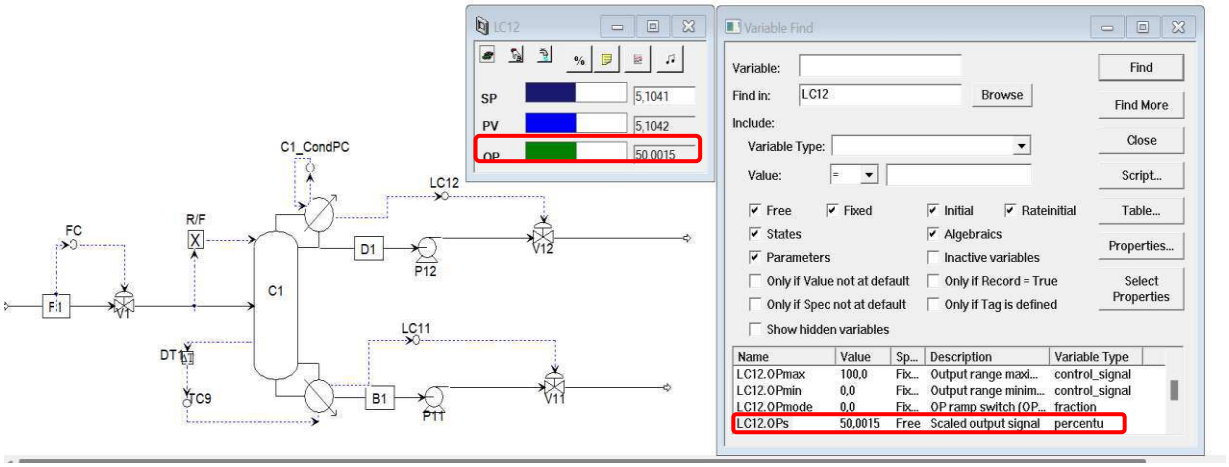


Figura 5-7 - Identificação do caminho das variáveis dentro do Aspen Plus Dynamics.

Tabela 5-9 - Caminho para cada variável utilizada no treinamento

Variável	Caminho
$[IC_4]$	ACMObj.Application.Simulation.Flowsheet.STREAMS("8").Zn("IC4").value
$SUMP_{\%}$	ACMObj.Application.Simulation.Flowsheet.LC11.OPs.value
$CD_{\%}$	ACMObj.Application.Simulation.Flowsheet.LC12.OPs.value
Q_{REB}	ACMObj.Application.Simulation.Flowsheet.TC9.OP.value

Após a definição dos parâmetros e integração entre Python-Aspen o algoritmo segue para o módulo "***Reset of Environment***" (Figura 5-8.a) que controla o início dos novos episódios. No início de cada novo episódio, a simulação é reiniciada e um distúrbio aleatório é aplicado na composição de propano na corrente de alimentação F1 da coluna, seguindo uma distribuição U~ (0.34-0.46).

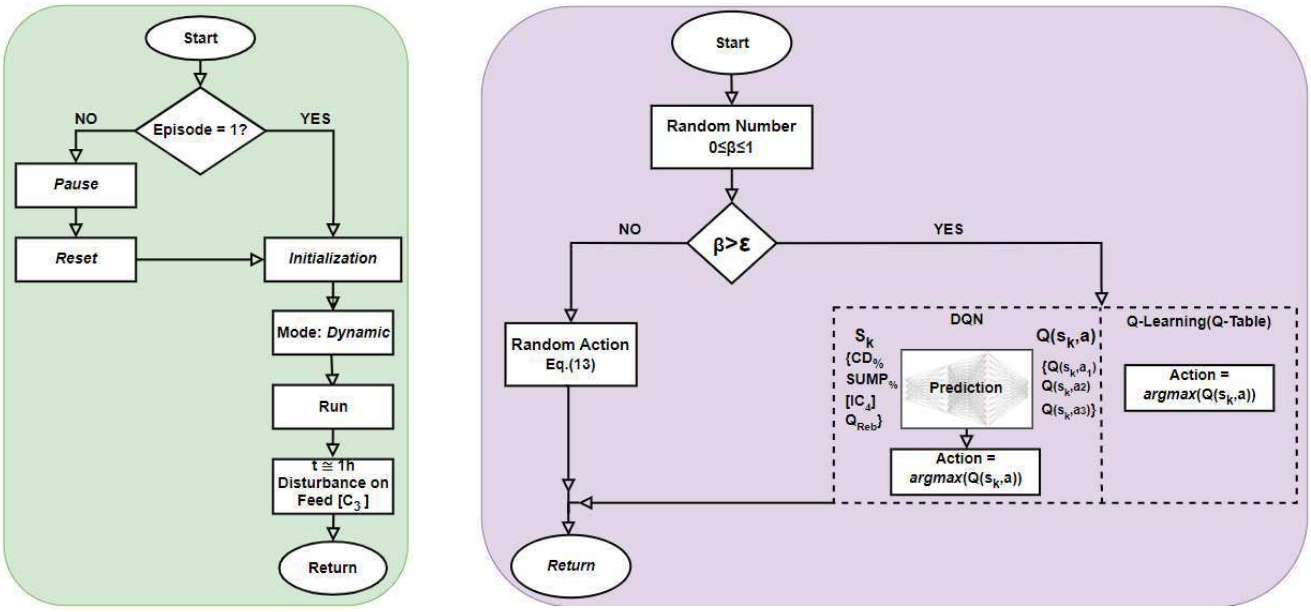
A partir deste ponto, as ações do algoritmo são definidas no módulo "***Selection of best action***" (Figura 5-8.b) com base nas relações de exploração e aproveitamento seguindo a política (π) ε-greedy, independente da metodologia avaliada (Q-learning ou DQN).

Na fase de exploração, o valor de ε é alto, neste caso o algoritmo tem uma maior probabilidade de escolher ações aleatórias objetivando avaliar o maior número de estados possível. Na fase de aproveitamento, o valor de ε é reduzido ao longo dos episódios conforme apresentado na Eq. (5-15), até atingir o valor mínimo de 0.01 na metade do treinamento. Nesta fase, o objetivo do algoritmo é refinar os valores de Q(s,a) obtidos na fase de exploração a partir da seleção da melhor ação possível em cada estado.

$$\varepsilon = \begin{cases} 1; Episódio = 1 \\ \varepsilon - \dfrac{1}{Episodios/2}; 1 < Episódio < Episodios/2 \\ 0.01; Episódio \geq 300 \end{cases} \quad (5\text{-}15)$$

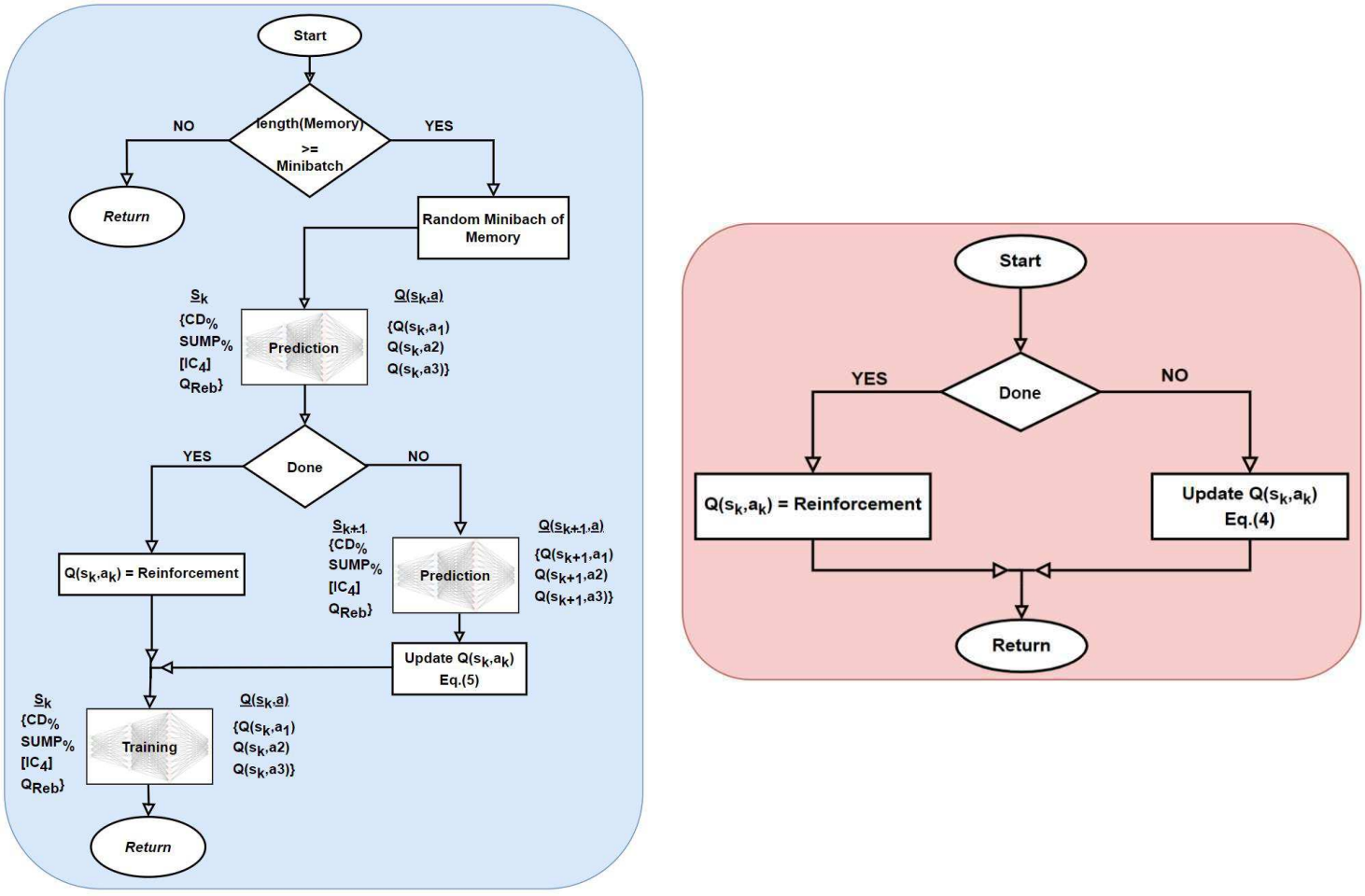
Após a determinação da ação (a_k), um novo contato com a simulação no Aspen Plus Dynamics é realizado para a alteração do setpoint do controlador da temperatura do estágio 09 da coluna. Em seguida, o novo estado s_{k+1} do sistema é coletado 1h após a alteração do setpoint e o reforço do agente é fornecido com base na Eq. (5-12). O tempo de 1h é necessário para o sistema atingir o novo estado estacionário.

A atualização da taxa de manipulação com base na Eq. (5-14) é realizada em seguida dependendo do status do episódio. A atualização dos valores de $Q(s_k,a_k)$ apresentado na Figura 5-9 ocorre de maneiras distintas dependendo da metodologia avaliada.



(a) (b)

Figura 5-8 - Subetapas do algoritmo referentes ao controle de novos episódios (5-8.a) e a determinação das novas ações (5-8.b)



(a) (b)

Figura 5-9 – Subetapas do treinamento da RNA (5-9.a) e atualização da Q-Table (5-9.b)

No modulo "***Q-Table Atualization***", o Q-value para o estado-ação atual é atualizado para R_s^a caso a meta tenha sido alcançada. Caso contrário, o valor de $Q(s_k,a_k)$ é atualizado com base no reforço atual e na recompensa futura ($max_a\ Q(s_{k+1},a)$) a partir Eq. (5-4).

Em contrapartida, no módulo "***RNA training***",os valores de $Q(s_k, a_k)$ são ajustados com base nas informações armazenadas em uma memória, que é preenchida à medida que o algoritmo explora o ambiente. Após coletar informações suficientes na memória, o DQN seleciona aleatoriamente um minibatch de dados para atualizar os valores de $Q(s_k, a_k)$.

Esse treinamento em minibatch ajuda a melhorar a estabilidade do processo de treinamento, evitando a correlação temporal entre os dados. A Tabela 5-10 apresenta as informações armazenadas na memória, enquanto a Tabela 5-11 apresenta as dimensões da memória e do minibatch.

Para cada dado no minibatch, o valor de $Q(s_k, a_k)$ é atualizado com base no status do ambiente. Se o status for "Done", o valor é definido como a recompensa obtida; caso contrário, ele é atualizado de acordo com a Eq. (5-5). A RNA é então treinada usando o conjunto de

estados "s" do minibatch como entrada e os valores atualizados de Q(sk, ak) como saída, utilizando a função de perda descrita pela Eq. (5-6) para atualização dos pesos sinápticos.

Tabela 5-10 - Informações armazenadas na memória.

Parâmetros
Estado Atual (s_k)
Reforço atribuído (R_s^a)
Status (*Done* ou *Fail*)
Ação tomada (novo setpoint de temperatura do estágio 9)

Tabela 5-11 - Dimensões da memória e do minibatch

Parâmetro	**Valor**
Tamanho da Memória	20000
MiniBatch	300

Após a atualização do passo s_k, o algoritmo pode retornar para o módulo "***Selection of best action***" ou seguir para o próximo episódio, dependendo do status. Na segunda opção, o valor de ε é atualizado com base na Eq. (5-15) e um novo episódio é iniciado no módulo "***Reset of Environment***".

5.Resultados

5.1.Treinamento

A etapa de treinamento foi realizada seguindo o fluxograma geral da Figura 5-5. A Figura 5-10.a apresenta os valores de composição de IC_4 na base da coluna obtidos durante o treinamento para o Q-learning, enquanto a metodologia utilizando DQN é apresentada na Figura 5-10.b. Os distúrbios que culminaram em falha ao longo dos episódios para as metodologias Q-learning e DQN são apresentados nas Figura 5-10.c e Figura 5-10.d, respectivamente.

Os valores iniciais da Q-table e dos pesos sinápticos da rede neural não fornecem uma boa estimativa dos valores de Q para cada ação. Neste sentido, espera-se que o agente aprenda cometendo mais erros do que acertos na fase de exploração.

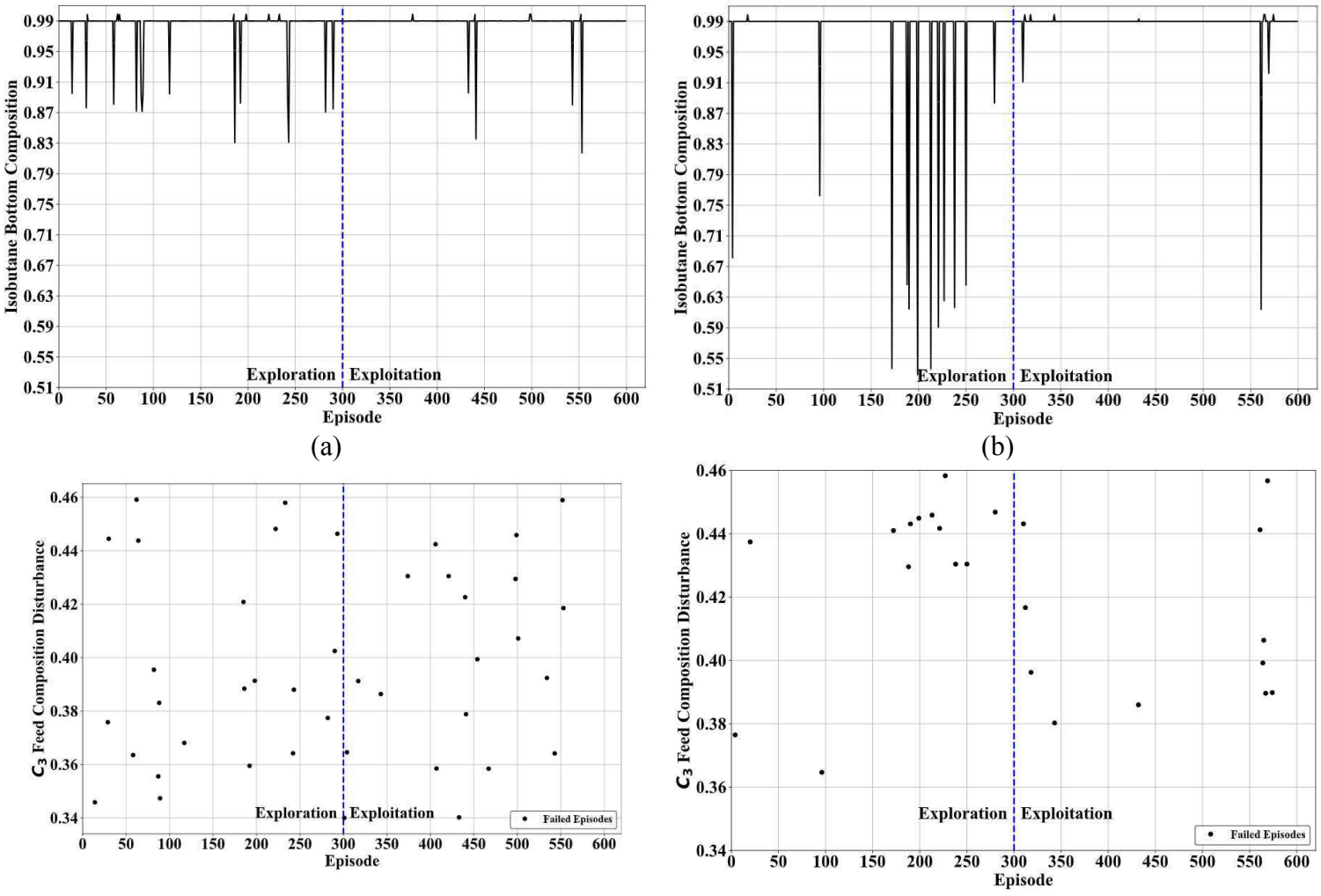


Figura 5-10 - IC_4 na base da coluna (5-10.a – 5-10.b) e dispersão dos distúrbios que provocaram falha (5-10.c – 5-10.d), para as metodologias Q-learning e DQN, respectivamente.

No final da etapa de aprendizado apresentado nas Figura 5-10.a-b, as metodologias avaliadas foram capazes de fornecer resultados que corrigiram o offset de composição gerado pelo distúrbio na alimentação, mantendo a composição de isobutano na base da coluna dentro da tolerância (1×10^{-4}) especificada.

Entretanto, no Q-learning, não houve uma melhora gradativa ao longo do treinamento na maioria das regiões de distúrbio, mesmo com o refino dos valores de $Q(s_k,a_k)$ após a etapa de exploração. Neste caso, as falhas estão primordialmente ligadas à estratificação do ambiente em um número finito de possibilidades.

O deslocamento do agente dentro do ambiente para estados de composição de IC_4 abaixo de 0.9 culminam automaticamente na falha do episódio, uma vez que, o intervalo da Q-Table para essa variável é 0,9-1. Esse comportamento pode ser visualizado na maioria das falhas de treinamento do Q-learning (Figura 5-10.a).

Em contrapartida, a abordagem utilizando DQN permite a correção dessa fragilidade, a partir da utilização de uma rede neural. A RNA é capaz de lidar com os problemas de alta dimensão e não lineares de uma forma mais robusta e com um menor esforço computacional.

A análise da Figura 5-10.b aponta a redução da frequência de falhas após a etapa de exploração, indicando uma melhoria após o refino dos valores de Q-value na fase de aproveitamento.

Nos episódios que apresentaram falha (Figura 5-10.d), a composição de IC_4 na base da coluna chegou a valores menores utilizando a DQN. Isso ocorre, pois, diferentemente da Q-table, a DQN aprende ao longo de toda a faixa de composição, sendo limitada pela apenas pela convergência dos balanços de massa e energia da coluna.

A Figura 5-11 apresenta os percentuais de acerto cumulativo para o Q-learning e a DQN, calculado a cada episódio com base na equação $\frac{\sum_1^i n_{Sucess}}{n_{Episodes}}$, onde i é o episódio atual, n_{sucess} é o número de episódios finalizados com sucesso até o momento e $n_{Episodes}$ é o número de episódios totais finalizados.

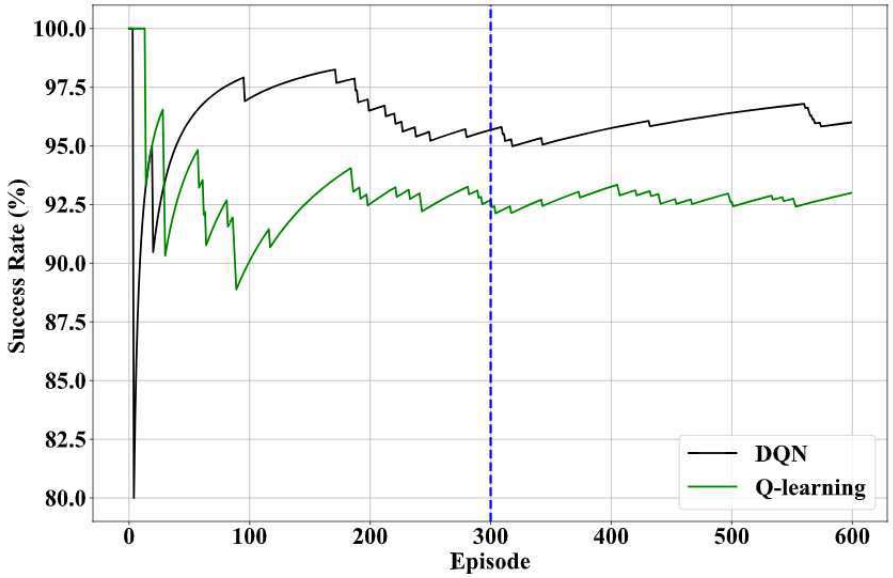


Figura 5-11 - Percentual de acerto cumulativo para as metodologias avaliadas

Para o Q-learning, o percentual de acerto se mantém em 92.76 ± 0.25 enquanto a DQN apresenta percentuais superiores mesmo nos episódios iniciais. Próximo ao final da fase de exploração da metodologia DQN, entre os episódios 150 e 250, ocorre uma redução do percentual de acertos em decorrência das falhas ocorridas nessa região e posteriormente, se mantem em um patamar de 95.94 $\pm$ 0.49.

5.2.Teste

Para a etapa de teste, foram avaliados 100 pontos de distúrbio na composição de propano da alimentação, gerados seguindo uma distribuição uniforme $U\sim$ (0,34-0,46). Os testes foram realizados com os modelos finais gerados em cada metodologia. A Tabela 5-12 apresenta o comparativo entre as duas metodologias avaliando o percentual de casos em que o modelo obteve sucesso e a mediana das tentativas realizadas para alcançar a meta.

Tabela 5-12 - Teste comparativo entre as metodologias Q-learning e DQN.

Metodologia	%Sucesso	Mediana de tentativas até o sucesso
Q-learning	66	3
DQN	96	2

Observa-se que a DQN apresentou uma assertividade 96% dos casos avaliados contra 66% de acerto utilizando o Q-learning, confirmando a melhoria do modelo ao alterar a estrutura de dados da Q-Table para DQN.

Para avaliação comparativa da dinâmica das metodologias, analisou-se a resposta do sistema a um distúrbio no qual ambas atingiram a meta com o número representativo de tentativas apresentado na Tabela 5-12. O distúrbio ocorre em 1h de simulação e as atuações dos algoritmos de Q-learning e DQN ocorrem a cada hora.

A avaliação da Figura 5-12.a confirma a limitação da malha de controle inferencial utilizada para manter a composição dentro da especificação requerida (0.99) após o distúrbio realizado na composição de propano (C_3) na corrente F1, de 0.4 para 0.3541.

O controlador PID exerce a função de manter a temperatura do estágio 9 no setpoint especificado, conforme observado na Figura 5-12.b. Entretanto, perturbações na composição de propano em F1 exigem um novo setpoint de temperatura para a obtenção do produto de base com composição de 0,99 de [IC_4]. No cenário avaliado, o novo setpoint é de 338 K.

A incorporação dos algoritmos de *reinforcement learning* demonstram a capacidade de correção dos offsets em ambas as metodologias (Q-learning e DQN) utilizadas. Entretanto, a partir da Figura 5-12.a observa-se que o controle PID acoplado ao Q-learning levou aproximadamente 1,2 horas a mais para retornar à composição alvo, em comparação ao PID acoplado ao DQN.

O tempo de retorno para a composição correta na base da coluna está relacionado com os valores de $Q(s_k,a_k)$ obtidos ao final do treinamento por cada metodologia. Enquanto o Q-learning realizou apenas uma alteração no tempo t= 4h para correção do offset o algoritmo do

DQN realizou alterações parciais no setpoint do controlador nos tempos t = 2h e 3h reduzindo o tempo de correção para a composição alvo na base da coluna.

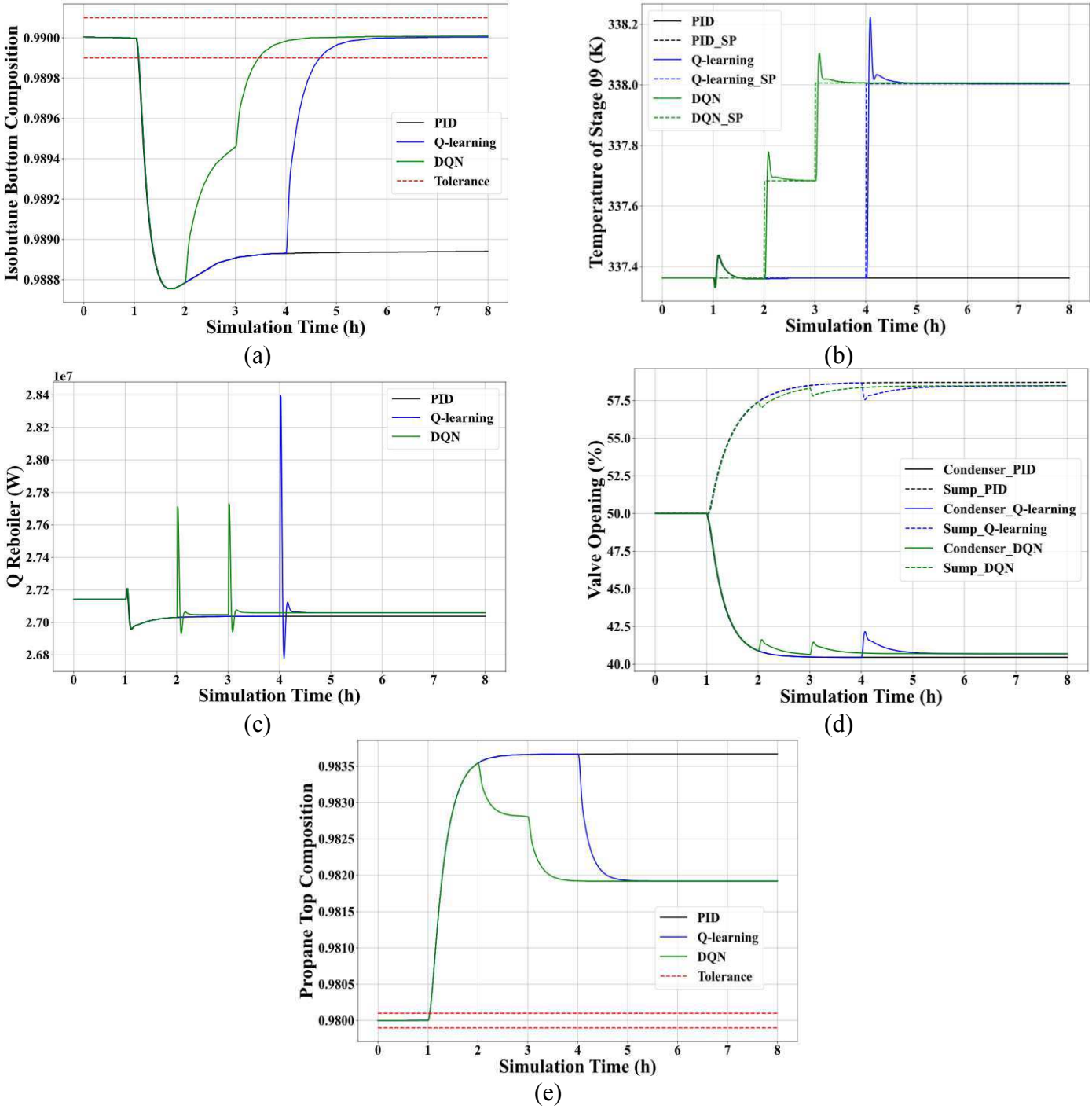


Figura 5-12 - Comportamento da [IC_4] na base da coluna (5-12.a); Temperatura do estágio 9 (5-12.b); QReb (5-12.c); aberturas de válvula para o condensador e sump (5-12.d) e Comportamento da [C_3] no topo da coluna (5-12.d) para as metodologias avaliadas.

A superioridade da estratégia utilizando o controle feedback em conjunto com o DQN em comparação com as demais é evidenciada na Tabela 5-13, que avalia o desempenho do sistema de controle usando a métrica da integral do erro absoluto (IAE), conforme a Eq. (5-16).

Ao analisar a tabela, fica evidente que o PID apresenta o maior erro de magnitude devido ao offset gerado, enquanto a estratégia DQN exibe um IAE menor, destacando sua superioridade em relação à Q-Table.

Tabela 5-13 - IAE índice para as diferentes estratégias de controle

Estratégia de Controle	IAE
PID	$7{,}44 \times 10^{-3}$
Q-learning	$3{,}46 \times 10^{-3}$
DQN	$1{,}83 \times 10^{-3}$

$$IAE = \int_{t_0}^{t} \left| [IC_4] - [IC_4]_{target} \right| \mathrm{dt};\ [IC_4]_{target} = 0.99\ e\ \Delta t = 0.01 \qquad (5\text{-}16)$$

Em contrapartida, não foi possível corrigir completamente a composição de propano no topo da coluna. Neste caso, mantendo a estratégia do controle feedback em conjunto com o DQN seria necessário aumentar o número de variáveis observáveis e controladas para correção simultânea dos *offsets* das composições de topo e base.

Em relação aos níveis do sump e do condensador (Figura 5-12.d), observa-se que, em ambas as metodologias, os algoritmos não sugeriram alterações que levassem o processo para regiões próximas a saturação das válvulas. Por outro lado, a redução da carga térmica está relacionada com a redução da razão de refluxo da coluna.

6.Conclusão

O controle de composição em colunas de destilação é uma tarefa importante, geralmente realizada com controladores feedback do tipo PID. A utilização deste tipo de controlador em malhas de controle inferencial pode não ser eficiente para correção de distúrbios. Neste trabalho foram identificados offsets na composição do produto provocados por essa limitação.

Estratégias de controle do tipo data-driven utilizando RL são alternativas promissoras, entretanto, trabalhos recentes indicam um foco na performance dos algoritmos de RL negligenciando a robustez na modelagem do ambiente.

O trabalho apresentou um algoritmo de treinamento dinâmico integrado utilizando Python e Aspen Plus Dynamics para avaliação de diferentes modelos de reinforcement learning. Embora o ambiente avaliado tenha sido uma coluna de destilação, o algoritmo é genérico e pode ser adaptado para diferentes processos e softwares de análise dinâmica. A partir da comunicação

do tipo COM foi possível rastrear as variáveis observáveis no software Aspen Plus Dynamics e realizar alterações no modelo dinâmico ao longo de toda a simulação.

No estudo de caso utilizando as abordagens Q-learrning e DQN, a análise acoplada do controlador convencional (PID) com esses algoritmos mostrou-se uma alternativa atraente para corrigir limitações de offset sem alterar a malha de controle principal da coluna. Além disso, a utilização Aspen Plus Dynamics como ambiente de simulação viabilizou a inclusão das principais variáveis de controle e de restrição durante o treinamento do modelo.

Os resultados apontam a limitação da malha de controle inferencial utilizada para manter a composição dentro da especificação de IC_4 na base da coluna, enquanto a utilização conjunta do controlador PID e abordagem DQN permite a manutenção da composição dentro da especificação em 96% dos cenários avaliados e com um IAE 52,9 % menor em comparação com o Q-Learning.

Neste sentido, a abordagem DQN mostrou-se capaz de lidar com os problemas de alta dimensão e não lineares de uma forma mais robusta e com um menor esforço computacional. Em contraste, a abordagem Q-learning apresentou fragilidades na generalização da estrutura da Q-table, que pode afetar a capacidade do agente de lidar com novas situações.

7.Referências

Alhazmi, K., & Sarathy, S. M. (2023). **Nonintrusive parameter adaptation of chemical process models with reinforcement learning. Journal of Process Control**, 123, 87–95. https://doi.org/10.1016/J.JPROCONT.2023.02.001

Chao, K. C., & Seader, J. D. (1961). **A General Correlation of Vapor-Liquid Equilibria in Hydrocarbon Mixtures**. AIChE Journal, 7, 598. https://doi.org/10.1002/aic.690070414

Deng, J., Sierla, S., Sun, J., & Vyatkin, V. (2022). **Reinforcement learning for industrial process control: A case study in flatness control in steel industry**. Computers in Industry, 143, 103748. https://doi.org/10.1016/J.COMPIND.2022.103748

Dutta, D., & Upreti, S. R. (2023). **A multiple neural network and reinforcement learning-based strategy for process control**. Journal of Process Control, 121, 103–118. https://doi.org/10.1016/J.JPROCONT.2022.12.004

Géron, A. (2017). **Hands-On Machine Learning with Scikit-Learn and TensorFlow** (1st ed.). O'Reilly Media, Inc.

Hou, Z. S., & Wang, Z. (2013). **From model-based control to data-driven control: Survey, classification and perspective.** Information Sciences, 235, 3–35. https://doi.org/10.1016/J.INS.2012.07.014

Hwangbo, S., & Sin, G. (2020). **Design of control framework based on deep reinforcement learning and Monte-Carlo sampling in downstream separation**. Computers & Chemical Engineering, 140, 106910. https://doi.org/10.1016/J.COMPCHEMENG.2020.106910

Kano, M., & Ogawa, M. (2009). **The State of the Art in Advanced Chemical Process Control in Japan**. IFAC Proceedings Volumes, 42(11), 10–25. https://doi.org/10.3182/20090712-4-TR-2008.00005

Kretchmar, R. M., Young, P. M., Anderson, C. W., Hittle, D. C., Anderson, M. L., & Delnero, C. C. (2001). **Robust reinforcement learning control with static and dynamic stability.** International Journal of Robust and Nonlinear Control, 11(15), 1469–1500. https://doi.org/10.1002/rnc.670

Luyben, W. L. (2006). **Distillation design and control using Aspen simulation**. Wiley-Interscience.

Martinez, E. C. (2000). **Batch process modeling for optimization using reinforcement learning.** Computers & Chemical Engineering, 24(2–7), 1187–1193. https://doi.org/10.1016/S0098-1354(00)00354-9

Mendel, J. M., & McLaren, R. W. (1970). 8 **Reinforcement-Learning Control and Pattern Recognition Systems. Mathematics in Science and Engineering**, 66(C), 287–318. https://doi.org/10.1016/S0076-5392(08)60497-X

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

Pellegrini, J., & Wainer, J. (2007). **Processos de Decisão de Markov: um tutorial. Revista de Informática Teórica e Aplicada**, 14(2), 133–179. https://doi.org/10.22456/2175-2745.5694

Radac, M. B., & Precup, R. E. (2019). **Data-driven model-free tracking reinforcement learning control with VRFT-based Adaptive Actor-Critic**. Applied Sciences (Switzerland), 9(9). https://doi.org/10.3390/app9091807

Ramos, W. B., Figueirêdo, M. F., Brito, K. D., Ciannella, S., Vasconcelos, L. G. S., & Brito, R. P. (2016). **Effect of Solvent Content and Heat Integration on the Controllability of Extractive Distillation Process for Anhydrous Ethanol Production**. Industrial and Engineering Chemistry Research, 55(43), 11315–11328. https://doi.org/10.1021/acs.iecr.6b03515

Selvi, D., Piga, D., & Bemporad, A. (2018). **Towards direct data-driven model-free design of optimal controllers**. In 2018 European Control Conference (ECC). https://doi.org/10.0/Linux-x86_64

Spielberg, S., Tulsyan, A., Lawrence, N. P., Loewen, P. D., & Bhushan Gopaluni, R. (2019). **Toward self-driving processes: A deep reinforcement learning approach to control**. AIChE Journal, 65(10). https://doi.org/10.1002/aic.16689

Sutton, R. S., & Barto, A. G. (2018). **Reinforcement Learning: An Introduction**. The MIT Press.

Syafiie, S., Tadeo, F., & Martinez, E. (2007). **Learning to control pH processes at multiple time scales: Performance assessment in a laboratory plant.** Chemical Product and Process Modeling, 2(1). https://doi.org/10.2202/1934-2659.1024

Syafiie, S., Tadeo, F., & Martinez, E. (2008). **Model-Free Learning Control of Chemical Processes**. In C. Weber, M. Elshaw, & N. M. Mayer (Eds.), Reinforcement Learning. IntechOpen. https://doi.org/10.5772/5287

Tulsyan, A., Garvin, C., & Ündey, C. (2018). **Advances in industrial biopharmaceutical batch process monitoring: Machine-learning methods for small data problems**. Biotechnology and Bioengineering, 115(8), 1915–1924. https://doi.org/10.1002/bit.26605

Tulsyan, A., Garvin, C., & Undey, C. (2018). **Machine-learning for biopharmaceutical batch process monitoring with limited data.** 51(18), 126–131. https://doi.org/10.1016/j.ifacol.2018.09.287

Tulsyan, A., Garvin, C., & Undey, C. (2019). **Industrial batch process monitoring with limited data. Journal of Process Control**, 77, 114–133. https://doi.org/10.1016/j.jprocont.2019.03.002

Tulsyan, A., Khare, S. R., Huang, B., Gopaluni, R. B., & Forbes, J. F. (2013). **Bayesian identification of non-linear state-space models: Part I- Input design**. IFAC Proceedings Volumes, 46(32), 774–779. https://doi.org/10.3182/20131218-3-IN-2045.00105

Wang, H., & Ge, Z. (2020). **Internal environment controlling algorithm of DMF distillation column based on reinforcement learning**. 2020 IEEE 9th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), 1479–1483. https://doi.org/10.1109/ITAIC49862.2020.9338759

Wilson, J. A., & Martinez, E. C. (1997). **Neuro-fuzzy modeling and control of a batch process involving simultaneous reaction and distillation**. Computers & Chemical Engineering, 21(SUPPL.1), S1233–S1238. https://doi.org/10.1016/S0098-1354(97)87671-5

Capítulo 6 Conclusão

O controle preciso da composição em colunas de destilação é crucial para garantir a qualidade do produto e o desempenho do processo. Tradicionalmente, controladores do tipo PID têm sido amplamente utilizados, mas sua eficácia pode ser limitada dependendo da malha de controle utilizada.

Recentemente, estratégias de controle data-driven com o uso de *reinforcement learning* surgiram como alternativas promissoras. No entanto, os estudos recentes focam primordialmente no desempenho dos algoritmos RL, negligenciando a robustez na modelagem do ambiente.

Neste trabalho, foi desenvolvido um algoritmo de treinamento dinâmico integrado em Python e Aspen Plus Dynamics para avaliar o controle de uma coluna de destilação com malha de controle inferencial para composição. Para avaliar o algoritmo, foram utilizadas duas técnicas de RL: Q-learning e DQN.

No Q-learning, foram exploradas duas estratégias de controle: uma com uma taxa de atuação fixa de 0,5% e outra com uma taxa de atuação flexível variando de 0,1% a 5%, dependendo do desvio entre o estado atual e a meta especificada. Os resultados revelaram a limitação da malha de controle inferencial utilizada para manter a composição de IC_4 dentro da especificação na base da coluna.

No entanto, a abordagem que combina o controlador PID com o Q-Learning, adotando uma taxa de atuação variável, mostrou-se promissora, mantendo a composição dentro das especificações em 65% dos cenários avaliados, um aumento significativo em relação à estratégia de AR fixo.

Apesar dos avanços com o Q-Learning, foram observadas limitações relacionadas à generalização da estrutura da Q-table. Para superar esse desafio, a segunda parte do trabalho buscou substituir o Q-Learning pela metodologia Deep Q-Network (DQN). A transição para a abordagem DQN, mantendo a taxa de manipulação variável, obteve resultados ainda mais promissores. O uso do controle PID com DQN permitiu manter a composição dentro da especificação em 96% dos cenários de teste, com um IAE 52,9 % menor em comparação com o Q-Learning.

Essa significativa melhoria na eficiência do controle, proporcionada pela DQN, destacou sua capacidade de lidar com problemas de alta dimensão e não-lineares de forma mais robusta, com menor demanda computacional.

A partir da comunicação do tipo COM estabelecida foi possível rastrear as variáveis observáveis no software Aspen Plus Dynamics e realizar alterações no modelo dinâmico a partir do Python ao longo da simulação em todos os cenários avaliados nesse estudo.

Embora o ambiente proposto tenha sido uma coluna de destilação, o algoritmo é genérico e pode ser adaptado para diferentes processos e softwares de análise dinâmica

Capítulo 7 Sugestões para trabalhos futuros

Com base no trabalho desenvolvido, surgem oportunidades para a exploração de outras linhas de pesquisa em projetos futuros. Abaixo, destaca-se algumas sugestões promissoras:

- Avaliação comparativa entre a estratégia utilizando DQN e algoritmos mais complexos de RL como DDPG, ou Double Q-learning.
- Avaliação comparativa entre estratégia utilizando DQN e estratégias de controle do tipo MPC e GMC.
- Ampliação do número de distúrbios e de variáveis observáveis do sistema buscando o controle das composições de topo e base.
- Avaliar a flexibilidade do algoritmo para controle de outros processos/equipamentos (reatores, trocadores de calor etc.).
- Avaliação comparativa entre a estratégia utilizando DQN e o controle cascata utilizando PID.
- Avaliar a inserção de um controle dual de temperatura para controle mútuo da composição de topo e base. Em seguida, comparar o controle dual com o controle utilizando DQN para controle das composições de topo e base.

Capítulo 8 Apêndices

8.1.1 Apêndice A - Classificação e Arquitetura das Redes Neurais

8.1.1.1.1 Redes Neurais Feedforward (Autoencoders)

As redes classificadas como autoencoders possuem características similares às redes MLP. A diferença entre elas está mais relacionada à aplicação do que à arquitetura em si. A ideia fundamental desse tipo de rede é codificar informações automaticamente usando estruturas de rede que se assemelham a uma ampulheta, conforme ilustrado na Figura 8-1 (BOURLARD, 1998).

A estrutura de um autoencoder é sempre simétrica em relação à camada intermediária. As camadas que vão da entrada até a parte central são chamadas de codificação, enquanto a parte entre o centro e a camada de saída é chamada de decodificação. Esse tipo de rede é útil no processamento de imagens, pois permite aprender representações eficientes (codificações) das entradas da rede sem a necessidade de supervisão.

Essas codificações geralmente possuem uma dimensão bem menor do que a entrada original, o que torna esse tipo de rede útil para redução de dimensionalidade. A Figura 8-2 apresenta um esquema ilustrativo de como esse tipo de rede pode ser utilizado para identificar e representar uma imagem inicialmente ruidosa.

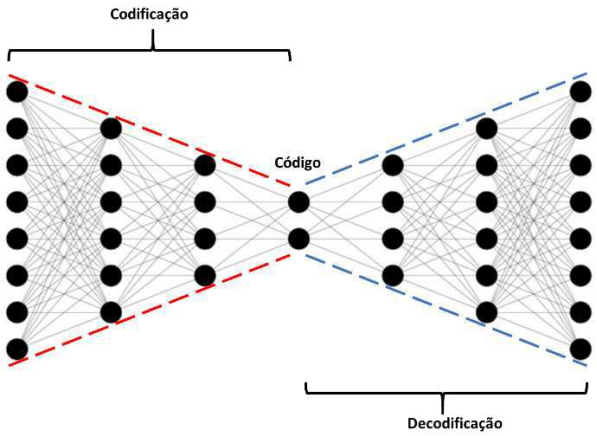


Figura 8-1 - Arquitetura simplificada de uma rede *Autoencoder*

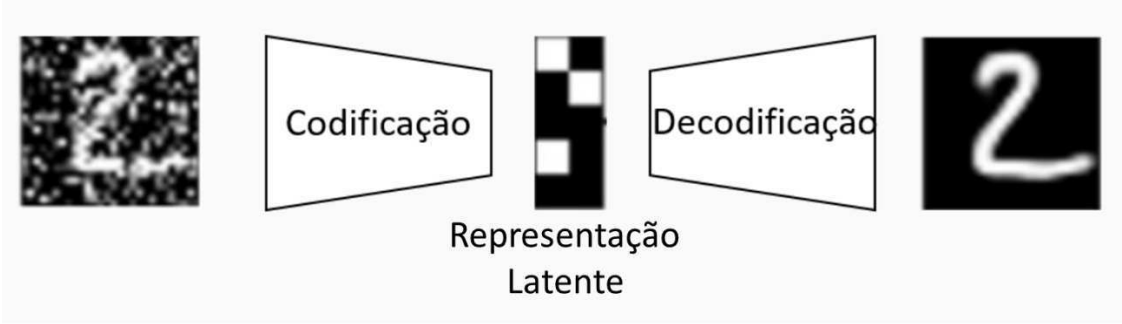


Figura 8-2 - Exemplo de aplicação da rede do tipo *autoencoder*.

8.1.1.2 Redes Neurais Recorrentes (RNN)

Redes neurais classificadas como recorrentes são consideradas as mais biologicamente realistas. Embora a arquitetura possa variar dentro dessa classificação, a principal característica dessas redes é que os neurônios não recebem apenas informações das camadas anteriores, mas também informações deles mesmos em um tempo ou instância anterior (ELMAN, 1990).

Além disso, podem existir interconexões entre neurônios da mesma camada, tornando as RNNs uma escolha adequada para complementar informações, como preenchimento automático de texto ou imagens, além da predição do comportamento dinâmico de sinais e processos.

A Figura 8-3 apresenta a estrutura de uma célula recorrente (a), juntamente com as dependências ao longo do tempo (b). A camada azul recebe a entrada x e gera a saída y, sendo U o peso da camada de entrada e w o peso da camada de saída. Além de propagar a informação para a próxima camada, a célula também transmite informações para si mesma, configurando o estado interno da rede, que flui para o próximo estado de tempo/instância, conforme observado em (b).

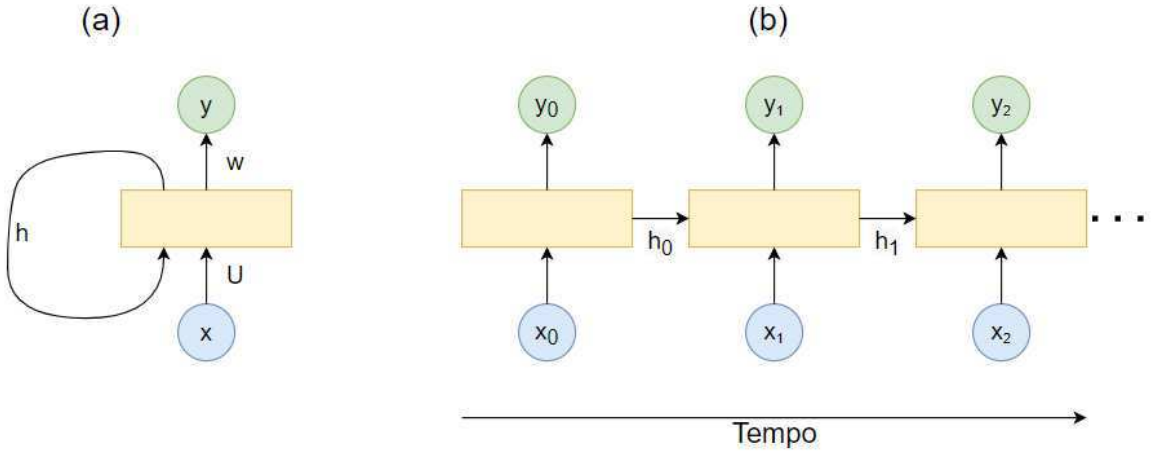


Figura 8-3 - Estrutura padrão de uma rede neural recursiva.
FONTE: Adaptado de GERÓN, 2017.

Em instâncias subsequentes ou ao longo do tempo, é natural que a informação associada ao estado interno h0 se dissipe. Em casos em que as informações de longo prazo, relacionadas a períodos anteriores, sejam importantes para a predição dos novos pontos, o uso da arquitetura padrão pode ser ineficiente.

Nesse sentido, redes neurais recorrentes que permitem o condicionamento de uma memória de longo prazo podem ser utilizadas para contornar esse problema. Dentre essas redes neurais recorrentes, destacam-se:

- Long Short-Term Memory (LSTM): Nesse tipo de arquitetura (Figura 8-4), cada neurônio é composto por uma célula de memória com três portas, descritas abaixo (HOCHREITER, 1997). Vale ressaltar que a quantidade de informações inseridas na rede por meio da porta de entrada é independente das informações retidas pela porta de esquecimento.
    - Porta de Entrada: Determina quanto da informação da camada anterior será armazenado na célula atual.
    - Porta de Saída: Determina quanto da informação será transmitido da camada atual para a próxima camada.
    - Porta de Esquecimento: Determina qual parte do estado da célula anterior, que já está armazenada na célula atual, deve ser esquecida/deletada.
- Gated Recurrent Units (GRU): São uma variação da arquitetura LSTM. As GRUs (Figura 8-5) possuem apenas duas portas, conforme descrito abaixo (CHUNG, 2014). Nesse caso, a porta de atualização é responsável por controlar a nova memória a ser adicionada e, consequentemente, a adição de novas informações (reinicialização). Portanto, a retenção de memória e a inclusão de novas informações não são independentes nesse tipo de arquitetura.
    - Porta de Reinicialização: Decide quais partes do estado oculto anterior devem ser combinadas com a entrada atual para propor um novo estado oculto.
    - Porta de Atualização: Determina quanto do estado oculto anterior deve ser retido e qual parte do novo estado oculto proposto (derivado da porta de reinicialização) deve ser adicionado ao estado oculto final.

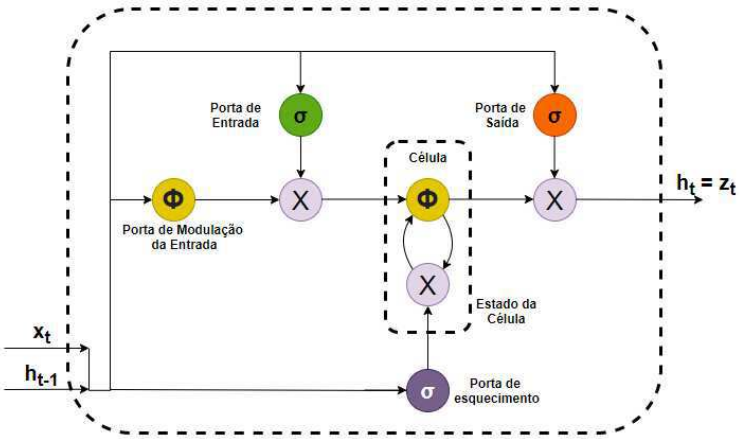


Figura 8-4 - Estrutura padrão LSTM

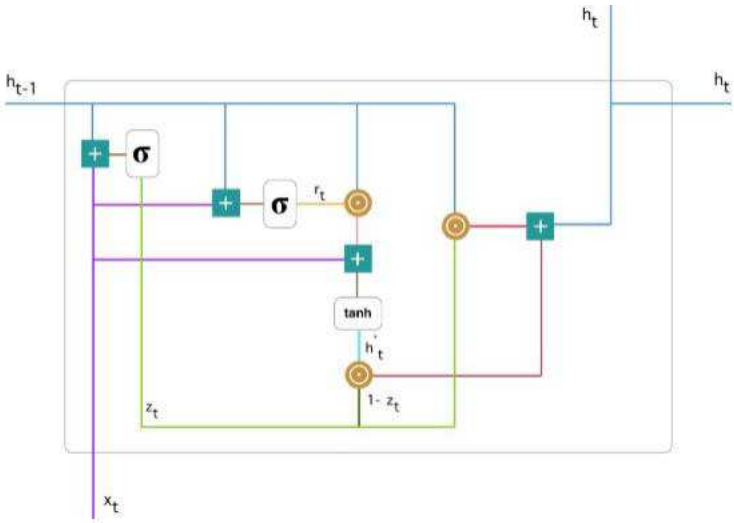


Figura 8-5 - Estrutura padrão GRU

8.1.1.3 Redes Neurais Simetricamente Conectadas

Redes neurais de arquitetura simétrica funcionam como redes recorrentes. No entanto, devido ao seu aspecto simétrico, os pesos são iguais em ambas as direções, o que facilita sua atualização. Por outro lado, essa simetria limita o universo de possibilidades, uma vez que os pesos dentro das redes não podem ser ajustados de maneiras diferentes. Dentro dessa categoria, destacam-se:

- Cadeias de Markov: Esse tipo de arquitetura não possui memória (Figura 3-10), pois segue a propriedade de Markov, ou seja, cada estado seguinte depende unicamente das informações do estado atual (HAYES, 2013). Por vezes, as cadeias de Markov não são consideradas redes neurais, mas formam a base teórica para as redes do tipo Hopfield e máquina de Boltzmann.
- Redes Hopfield (HN): Também conhecidas como redes auto-associativas, todos os neurônios desse tipo de rede estão interligados, como apresentado na Figura 8-6

(HOPFIELD, 1982). Como característica principal, as RH buscam reconhecer padrões e retornar os próprios padrões, sendo indicadas para preencher lacunas de padrões incompletos ou corrompidos.

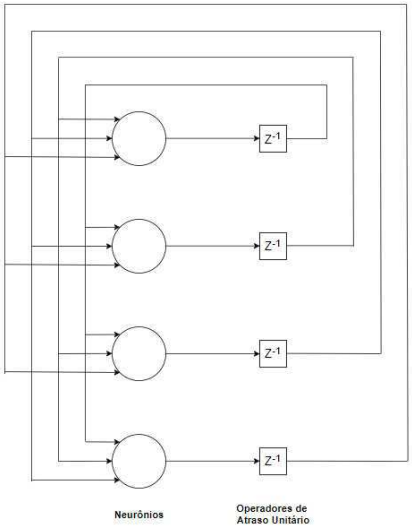


Figura 8-6 - Arquitetura convencional das redes *Hopfield*

- Máquina de Boltzmann (BM): São redes estocásticas semelhantes às HN, compostas principalmente por duas camadas: visível e oculta (Figura 8-7). A camada de unidades visíveis representa os dados observados e está conectada à camada oculta, que, por sua vez, extrai as características desses dados de forma não supervisionada (MEMISEVIC e HINTON, 2010). Além disso, os neurônios de entrada tornam-se neurônios de saída ao final de uma atualização completa da rede, que começa com pesos aleatórios e aprende por retropropagação.

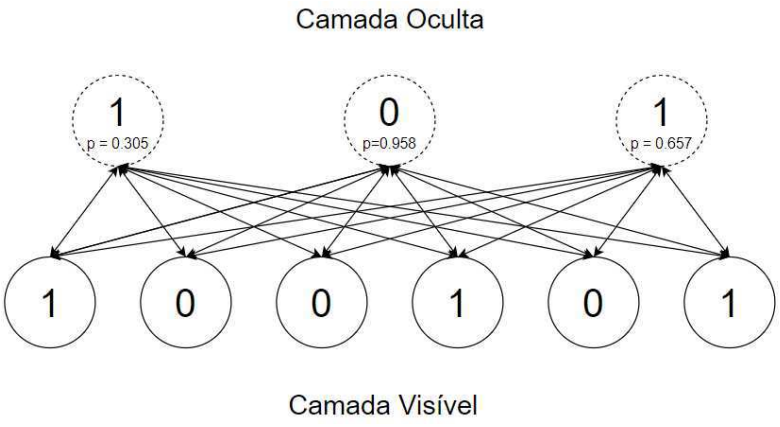


Figura 8-7 - Arquitetura convencional da máquina de *Boltzmann*

Capítulo 9 Referências

ALHAZMI, K.; SARATHY, S. M. **Nonintrusive parameter adaptation of chemical process models with reinforcement learning**. Journal of Process Control, v. 123, p. 87–95, 1 mar. 2023.

ALLEN, T. T.; ROYCHOWDHURY, S.; LIU, E. **Reward-based Monte Carlo-Bayesian reinforcement learning for cyber preventive maintenance**. Computers & Industrial Engineering, v. 126, p. 578-594, 2018.

ANDRYCHOWICZ, M.; BAKER, B.; CHOCIEJ, M.; JÓZEFOWICZ, R.; MCGREW, B.; PACHOCKI, J.; PETRON, A.; PLAPPERT, M.; POWELL, G.; RAY, A.; SCHNEIDER, J.; SIDOR, S.; TOBIN, J.; WELINDER, P.; WENG, L.; ZAREMBA, W**. Learning dexterous in-hand manipulation**. The International Journal of Robotics Research, v. 39, n. 1, p. 3-20, 2020.

BELLMAN, R., **'A Markovian Decision Process'**, Journal of Mathematics and Mechanics, vol. 6, no. 5, pp. 679-84, ISSN: 00959057, 1957.

BOURLARD, H., KAMP, Y., **Auto-association by multilayer perceptrons and singular value decomposition**. Biological cybernetics, v. 59, n. 4, p. 291-294, 1988.

CARTA, S.; FERREIRA, A.; PODDA, A. S.; RECUPERO, D. R.; SANNA, A. **Multi-DQN: An ensemble of Deep Q-learning agents for stock market forecasting**. Expert Systems with Applications, v. 164, p. 113820, 2021.

CHUNG, Junyoung et al. **Empirical evaluation of gated recurrent neural networks on sequence modeling**. arXiv preprint arXiv:1412.3555, 2014.

CHUO, H. S. E.; TAN, M. K.; THAM, H. J.; TEO, K. T. K. **Q-learning-based controller for fed-batch yeast fermentation**. In: Developments in Sustainable Chemical and Bioprocess Technology. Springer, Boston, MA, p. 219-225, 2013.

DENG, Y.; BAO, F.; KONG, Y.; REN, Z.; DAI, Q. **Deep direct reinforcement learning for financial signal representation and trading**. IEEE transactions on neural networks and learning systems, v. 28, n. 3, p. 653-664, 2016.

DENG, J. et al. **Reinforcement learning for industrial process control: A case study in flatness control in steel industry**. Computers in Industry, v. 143, p. 103748, 1 dez. 2022.

DUTTA, D.; UPRETI, S. R. **A multiple neural network and reinforcement learning-based strategy for process control.** Journal of Process Control, v. 121, p. 103–118, 1 jan. 2023.

ELMAN, Jeffrey L. **Finding structure in time**. Cognitive science, v. 14, n. 2, p. 179-211, 1990.

FISCHER, T. G. **Reinforcement learning in financial markets-a survey**. FAU Discussion Papers in Economics, 2018.

GERÓN, A., **Hands-On Machine Learning with Scikit-Learn and TensorFlow**. Ed. 1,2017

GLORENNEC, P. Y., JOUFFE, L., **Fuzzy Q-learning**, Proceedings of 6th International Fuzzy Systems Conference, pp. 659-662 vol.2, doi: 10.1109/FUZZY.1997.622790, 1997.

GLORENNEC, P. Y.; **Fuzzy Q-learning and dynamical fuzzy Q-learning.** Proceedings of 1994 IEEE 3rd International Fuzzy Systems Conference, pp. 474-479 vol.1, doi: 10.1109/FUZZY.1994.343739, 1994.

GOMIDE, R. **Operações Unitárias: operações de transferência de massa**. 4.ed. São Paulo:Ed. Reynaldo Gomide, 1988. 444p. v4

GOULART, D. A.; PEREIRA, R. D. **Autonomous pH control by reinforcement learning for electroplating industry wastewater**. Computers & Chemical Engineering, v. 140, p. 106909, 2020.

HABIB, N., **Hands-On Q-Learning with Python: Practical Q-learning with OpenAI Gym, Keras, and TensorFlow**.2019.

HALL, P.; PHAN, W.; WHITSON, K.; **The evolution of Analytics: Opportunities and Challenges for Machine Learning in Business.** 1 ed. O'Reilly, 2016.

HAYES, Brian et al. **First links in the Markov chain**. American Scientist, v. 101, n. 2, p. 252, 2013.

HAYKIN, S. **Redes Neurais: Princípios e Prática**. 2. ed. Porto Alegre: Bookman, 2001.

HOCHREITER, Sepp; SCHMIDHUBER, Jürgen. **Long short-term memory**. Neural computation, v. 9, n. 8, p. 1735-1780, 1997.

HOPFIELD, John J. **Neural networks and physical systems with emergent collective computational abilities**. Proceedings of the national academy of sciences, v. 79, n. 8, p. 2554-2558, 1982.

HOSKINS, J. C.; HIMMELBLAU, D. M. **Automatic chemical process control using reinforcement learning in artificial neural networks**. Neural Networks, v. 1, p. 446, 1988.

HOSKINS, J. C.; HIMMELBLAU, D. M. **Process control via artificial neural networks and reinforcement learning**. Computers & chemical engineering, v. 16, n. 4, p. 241-251, 1992.

HWANGBO, S.; SIN, G. **Design of control framework based on deep reinforcement learning and Monte-Carlo sampling in downstream separation**. Computers & Chemical Engineering, v. 140, p. 106910, 2020.

KISTER, H. Z. **Distillation Design**. [s.l.] McGraw-Hill Education, 1992.

KOBER, J.; BAGNELL, J. A.; PETERS, J. **Reinforcement learning in robotics: A survey**. The International Journal of Robotics Research, v. 32, n. 11, p. 1238-1274, 2013.

KRETCHMAR, R. M. et al. **Robust reinforcement learning control with static and dynamic stability. International Journal of Robust and Nonlinear Control**, v. 11, n. 15, p. 1469–1500, 30 dez. 2001.

KUMAR, ABHISHANKAR & MUNSHI, BASUDEB & TECH, M.; **Control of distillation column using aspen dynamics**. 10.13140/2.1.2762.2080, 2011.

LAMANNA R., HUMPHREY I., SDAZA S., **Simulacón estática y dinamica e identificación con redes neurales recurrentes de una columna de destilación binaria metanol-agua.** Undergraduate thesis. Universidad Simón Bolivar. Caracas, 1996

LAMANNA R.; MARGAGLIO E., **Neural networks for estimation of product properties in a pilot plant distillation column**. In Proceedings of the IASTED International Conference, pages 5-8. IASTED-Acta Press, 1995.

LILLICRAP, TIMOTHY P. et al. **Continuous control with deep reinforcement learning**. In: International Conference on Learning Representations (ICLR). 2016.

LIU, J.; TSAI, B. Y.; CHEN, D. S. **Deep reinforcement learning based controller with dynamic feature extraction for an industrial claus process.** Journal of the Taiwan Institute of Chemical Engineers, v. 146, p. 104779, 1 maio 2023.

LUYBEN, W. L, **Process Modeling, Simulation, and Control for Chemical Engineers**, 2nd ed., McGraw-Hi11, New York, 1990.

LUYBEN, W. L. Design and control of distillation columns with inert venting. **Computers & Chemical Engineering**, v. 134, p. 106725, 4 mar. 2020.

MACHALEK, Derek; QUAH, Titus; POWELL, Kody M. **Dynamic Economic Optimization of a Continuously Stirred Tank Reactor Using Reinforcement Learning**. In: 2020 American Control Conference (ACC). IEEE, p. 2955-2960, 2020.

MARGAGLIO, E.; LAMANNA, R.; GLORENNEC, P. Y. Control of a distillation column using fuzzy inference systems. In: **Proceedings of 6th International Fuzzy Systems Conference**. IEEE, p. 995-999, 1997.

MARLIN, T. E., **Process Control: Designing Processes and Control Systems for Dynamic Peformance**, McGraw-Hill, New York, 1995.

MARTINEZ, E. C. **Batch process modeling for optimization using reinforcement learning.** Computers & Chemical Engineering, v. 24, n. 2-7, p. 1187-1193, 2000.

MCCULLOCH; W. S.; PITTS. W.; **A logical calculus of the ideas immanent in nervous activity. Bulletin of mathematical biophysics**, vol. 5 (1943), pp. 115–133.

MEMISEVIC, Roland; HINTON, Geoffrey E. **Learning to represent spatial transformations with factored higher-order Boltzmann machines**. Neural computation, v. 22, n. 6, p. 1473-1492, 2010.

MENDEL, J. M.; MCLAREN, R. W. **Reinforcement-learning control and pattern recognition systems**. In: Mathematics in Science and Engineering. Elsevier, p. 287-318, 1970.

MICHIE, D.; CHAMBERS, R. A. BOXES: **An experiment in adaptive control**. Machine Intelligence, v. 2, n. 2, p. 137-152, 1968.

MISHRA, P.; KUMAR, V.; RANA, K. P. S. **A fractional order fuzzy PID controller for binary distillation column control**. Expert Systems with Applications, v. 42, n. 22, p. 8533–8549, 1 dez. 2015.

MNIH, V.; KAVUKCUOGLU, K.; SILVER, D.; GRAVES, A.; ANTONOGLOU, I.; WIERSTRA, D.; RIEDMILLER, M. **Playing atari with deep reinforcement learning**. arXiv preprint arXiv:1312.5602, 2013.

MONARD, M.C.; BARANAUSKAS, J.A.; **Conceitos sobre aprendizado de máquina**; Sistemas inteligentes-Fundamentos e aplicações; 2003.

MUSTAFA, M. A.; WILSON, J. A. **Application of Reinforcement Learning to Batch Distillation**. In: The Sixth Jordan International Chemical Engineering Conference. JIChE06, 2012.

NAIR, A.; MCGREW, B., ANDRYCHOWICZ, M.; ZAREMBA, W.; ABBEEL, P. **Overcoming exploration in reinforcement learning with demonstrations.** In: IEEE International Conference on Robotics and Automation (ICRA). IEEE, p. 6292-6299, 2018.

NEUNEIER, R. **Optimal asset allocation using adaptive dynamic programming**. Advances in Neural Information Processing Systems, p. 952-958, 1996.

NIAN, R.; LIU, J.; HUANG, B. **A review on reinforcement learning: Introduction and applications in industrial process control.** Computers & Chemical Engineering, p. 106886, 2020.

NIZAMI, M. **Development of a fuzzy logic controller for a distillation column using rockwell software,** University of Guelph, PHD Thesis, 2011.

OH, D. H. et al. **Actor-critic reinforcement learning to estimate the optimal operating conditions of the hydrocracking process.** Computers & Chemical Engineering, v. 149, p. 107280, 1 jun. 2021.

OTTONI, A. L. C., et al. **Análise do desempenho do aprendizado por reforço na solução do problema do caixeiro viajante**. XII SBAI-Simpósio Brasileiro de Automaçao Inteligente, p. 43-48, 2015.

PATEL, K. M. **A practical Reinforcement Learning implementation approach for continuous process control.** Computers & Chemical Engineering, v. 174, p. 108232, 1 jun. 2023.

PENDHARKAR, P. C.; CUSATIS, R. **Trading financial indices with reinforcement learning agents**. Expert Systems with Applications, v. 103, p. 1-13, 2018.

RAMOS, W. B. et al. **Effect of Solvent Content and Heat Integration on the Controllability of Extractive Distillation Process for Anhydrous Ethanol Production**. Industrial and Engineering Chemistry Research, v. 55, n. 43, p. 11315–11328, 2 nov. 2016.

ROCHA, R. S. **Determinação Experimental de Correntes do Processo de Destilação Molecular de Resíduos de Petróleo e Extensão da Curva PEV**, 2008. 190 f. Tese (Mestrado

em Engenharia Química) - Departamento de Engenharia Química, Universidade Estadual de Campinas, Campinas, 2008.

RODRIGUES, W. G. **Predição de Diâmetros e Cálculo de Volume de Clones de Eucalipto.** Goiânia: Universidade Federal de Goiás, 2019.

ROSENBLATT, F., **The perceptron, a perceiving and recognizing automaton**. Project Para. Cornell Aeronautical Laboratory, 1957.

RUMMERY, G. A.; NIRANJAN, M., **On-line Q-learning using connectionist systems.** Cambridge, UK: University of Cambridge, Department of Engineering, 1994.

SAMUEL, A. L. **Some studies in machine learning using the game of checkers**. IBM Journal of research and development, v. 3, n. 3, p. 210-229, 1959.

SAMUEL, A. L. **Some studies in machine learning using the game of checkers. II—Recent progress.** IBM Journal of research and development, v. 11, n. 6, p. 601-617, 1967.

SCHRITTWIESER, J.; ANTONOGLOU, I.; HUBERT, T.; SIMONYAN, K.; SIFRE, L.; SCHMITT, S.; GUEZ, A.; LOCKHART, E.; HASSABIS, D.; GRAEPEL, T.; LILLICRAP, T.; SILVER, D. et al. **Mastering atari, go, chess and shogi by planning with a learned model.** Nature, v. 588, n. 7839, p. 604-609, 2020.

SEADER, J. D.; HENLEY, E. J.; ROPER, D. K. **Separation process principles : chemical and biochemical operations**. 3. ed. [s.l.] John Wiley & Sons, 2010.

SELVI, D.; PIGA, D.; BEMPORAD, A. **Towards direct data-driven model-free design of optimal controllers**. [s.l: s.n.].

SHARMA, SIDDHARTH; SHARMA, SIMONE; ATHAIYA, ANIDHYA. A**ctivation Functions in Neural Networks.** International Journal of Engineering Applied Sciences and Technology. 04. 310-316. 10.33564/IJEAST. 2020. v04i12.054, 2020

SHIN, J.; BADGWELL, T. A.; LIU, K. H.; LEE, J. H. et al. **Reinforcement Learning–Overview of recent progress and implications for process control.** Computers & Chemical Engineering, v. 127, p. 282-294, 2019.

SILVA, L. M. D.; **Proposta de Arquitetura em Hardware para FPGA da Técnica Q-learning de Aprendizagem por Reforço**, Universidade Federal do Rio Grande do Norte, Dissetação de mestrado, Natal-Brasil, 2016.

SILVER, D.; HUANG, A.; MADDISON, C. J.; GUEZ, A.; SIFRE, L.; DRIESSCHE,G.; SCHRITTWIESER, J.; ANTONOGLOU, I.; PANNEERSHELVAM, V.; LANCTOT, M.; DIELEMAN, S.; GREWE, D.; NHAM, J.; KALCHBRENNER, N.; SUTSKEVER, I.; LILLICRAP, T.; LEACH, M.; KAVUKCUOGLU, K.; GRAEPEL, T.; HASSABIS, D. **Mastering the game of Go with deep neural networks and tree search**. Nature, v. 529, n. 7587, p. 484-489, 2016.

SILVER, D.; HUBERT, T.; SCHRITTWIESER, J.; ANTONOGLOU, I.; LAI, M.; GUEZ, A.; LANCTOT, M.; SIFRE, L.; KUMARAN, D.; GRAEPEL, T.; LILICRAP, T.; SIMONYAN,

K.; HASSABIS, D. **A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play**. Science, v. 362, n. 6419, p. 1140-1144, 2018.

SILVER, D.; LEVER, G.; HEESS, N.; DEGRIS, T.; WIERSTRA, D.; RIEDMILLER, M. **Deterministic policy gradient algorithms.** In: International conference on machine learning. PMLR, p. 387-395, 2014.

SINGH, V.; KODAMANA, H. **Reinforcement learning based control of batch polymerisation processes**. IFAC-PapersOnLine, v. 53, n. 1, p. 667-672, 2020.

SKOGESTAD S, MORARI M, DOYLE JC. **Robust control of ill-conditioned plants: high-purity distillation**. IEEE Trans Autom Control, 33(12): 1092-1105, 1988.

SKOGESTAD, S. **Dynamics and Control of Distillation Columns - A Critical Survey**. IFAC Proceedings Volumes, v. 25, n. 5, p. 11–35, 1 abr. 1992.

SOARES, C.; **Avaliação experimental dos coeficientes de transferência de massa e calor em uma coluna com pratos perfurados.** 2000. 290f. Tese (Mestrado em Engenharia Química) – Departamento de Engenharia Química, Universidade Estadual de Campinas, Campinas, 2000.

SPIELBERG, S.; TULSYAN, A.; LAWRENCE, N. P.; LOEWEN, P. D.; GOPALUNI, R. B. **Toward self-driving processes: A deep reinforcement learning approach to control**. AIChE Journal, v. 65, n. 10, p. e16689, 2019.

STEPHANOPOULOS, G., **Chemical Process Control**, Prentice-Hall, Englewood Cliffs, NJ, 1984.

SUTTON, R. S., **Learning to predict by the method of temporal differences.** Machine Learning, v. 3, p. 9-44, 1988.

SUTTON, R. S.; BARTO, A. G. **Reinforcement learning: An introduction**. MIT press, 2018.

SYAFIIE, S.; TADEO, F.; MARTINEZ, E**. Learning to control pH processes at multiple time scales: performance assessment in a laboratory plant**. Chemical Product and Process Modeling, v. 2, n. 1, 2007.

SYAFIIE, S.; TADEO, F.; MARTINEZ, E. **Model-free learning control of chemical processes.** In: Reinforcement Learning. IntechOpen, p. 295-310, 2008.

TESAURO, G. **Practical issues in temporal difference learning**. Machine learning, v. 8, n. 3, p. 257-277, 1992.

TESAURO, G. **Programming backgammon using self-teaching neural nets.** Artificial Intelligence, v. 134, n. 1-2, p. 181-199, 2002.

TESAURO, G. **TD-Gammon, a self-teaching backgammon program, achieves master-level play**. Neural computation, v. 6, n. 2, p. 215-219, 1994.

TESAURO, G. **Temporal difference learning and TD-Gammon.** Communications of the ACM, v. 38, n. 3, p. 58-68, 1995.

THORNDIKE, E., **Animal Intelligence: Experimental Studies**, ISBN: 9780765804822, The Macmillan Company, 1911

TULSYAN, A.; GARVIN, C.; UNDEY, C. **Machine-learning for biopharmaceutical batch process monitoring with limited data**. Elsevier B.V., 1 jan. 2018.

TULSYAN, A.; GARVIN, C.; UNDEY, C. **Advances in industrial biopharmaceutical batch process monitoring: Machine-learning methods for small data problems. Biotechnology and Bioengineering,** v. 115, n. 8, p. 1915–1924, 1 ago. 2018.

TULSYAN, A.; GARVIN, C.; UNDEY, C. **Industrial batch process monitoring with limited data. Journal of Process Control**, v. 77, p. 114–133, 1 maio 2019.

WAINER, J.., PELLEGRINI, J.; **Processos de Decisão de Markov: um tutorial**. Revista de Informática Teórica e Aplicada, v. 14, n. 2, p. 133-179, 2007.

WANG, H.; GE, Z. **Internal Environment Controlling Algorithm of DMF Distillation Column based on Reinforcement Learning**. In: 2020 IEEE 9th Joint International Information Technology and Artificial Intelligence Conference (ITAIC). IEEE, p. 1479-1483, 2020.

WATKINS C., **Leaning from Delayed Rewards**. PhD thesis, Cambridge University, Cambridge, England, 1989.

WATKINS, C. J. C. H.; DAYAN, P. **Q-learning.** Machine learning, v. 8, n. 3-4, p. 279-292, 1992.

WIDROW, B. et al. **Adaptive" Adaline" neuron using chemical" memistors"**. Number Technical Report 1553-2. Stanford Electron. Labs., Stanford, CA, October 1960.

WILSON, J. A.; MARTINEZ, E. C. **Neuro-fuzzy modeling and control of a batch process involving simultaneous reaction and distillation**. Computers & chemical engineering, v. 21, p. S1233-S1238, 1997.

YOO, H.; KIM, B.; KIM, J. W.; LEE, J. H. **Reinforcement learning based optimal control of batch processes using Monte-Carlo deep deterministic policy gradient with phase segmentation.** Computers & Chemical Engineering, v. 144, p. 107133, 2021.

ZHANG, Peng et al. **An improved reinforcement learning control strategy for batch processes.** In: 2019 24th International Conference on Methods and Models in Automation and Robotics (MMAR). IEEE, p. 360-365, 2019.

ZUMACH, F. C.; **Aplicação do método da continuação para simulação de colunas de destilação**. Campinas.