Pontifícia Universidade Católica
do Rio de Janeiro

Eric Monteiro e Lobo Luz

Estudo de técnicas de aprendizado por reforço aplicadas ao controle de processos químicos

Dissertação de Mestrado

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre pelo Programa de Pós–graduação em Engenharia Elétrica, do Departamento de Engenharia Elétrica da PUC-Rio.

Orientador: Prof. Wouter Caarls

Rio de Janeiro
Agosto de 2021

Pontifícia Universidade Católica
do Rio de Janeiro

Eric Monteiro e Lobo Luz

Estudo de técnicas de aprendizado por reforço aplicadas ao controle de processos químicos

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre pelo Programa de Pós–graduação em Engenharia Elétrica da PUC-Rio. Aprovada pela Comissão Examinadora abaixo:

Prof. Wouter Caarls
Orientador
Departamento de Engenharia Elétrica – PUC-Rio

Prof. Bruno Didier Olivier Capron
UFRJ

Prof. Karla Tereza Figueiredo Leite
UERJ

Prof. Brunno Ferreira dos Santos
Departamento de Engenha Química e de Materiais – PUC-Rio

Rio de Janeiro, 27 de Agosto de 2021

Eric Monteiro e Lobo Luz

Graduou-se em Engenharia Química pela Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio). Durante a graduação participou de projeto nas áreas nanotecnologia, biotecnologia, química analítica, machine learning e processos químicos. Possui experiências em simulações e controle de processos químicos, análises químicas com enfase em inteligência computacional.

Ficha Catalográfica

M L Luz, Eric

Estudo de técnicas de aprendizado por reforço aplicadas ao controle de processos químicos / Eric Monteiro e Lobo Luz; orientador: Wouter Caarls. – 2021.

91 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Elétrica, 2021.

Inclui bibliografia

1. Engenharia Eletrica – Teses. 2. Engenharia Qúimica – Teses. 3. Aprendizado por Reforço. 4. Aprendizado por Reforço Profundo. 5. Controle de Processos Químicos. 6. Reator de Van de Vusse. 7. Tennessee Eastman Process. 8. Q-Learning. 9. Actor Critic. 10. Deep Q-Learning. 11. DDPG. 12. TD3. 13. SAC. I. Caarls, Wouter. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Elétrica. III. Título.

CDD: 620.11

Agradecimentos

Primeiramente, gostaria de agradecer especialmente ao meu orientador Prof. Wouter Caarls. Quando entrei no mestrado, era um verdadeiro "*noob*" no ambiente Linux. Toda semana aparecia em sua sala, pedindo ajuda para resolver alguns problemas relacionado a execução dos algoritmos no ambiente Linux. Suas sugestões e conselhos foram essenciais, não apenas para o avanço desta dissertação como também enriquecimento da minha vida acadêmica. A paciência e companheirismo que ele mostrou neste curto período, nunca vou esquecer, eu o considero um amigo. Eu me considero afortunado pela oportunidade de trabalhar com alguém de tão grande conhecimento e parceiro.

Também sou grato aos membros da minha banca examinadora, Bruno Didier Olivier Capron, Karla Tereza Figueiredo Leite e Brunno Ferreira dos Santos que comprometeram muito generosamente o seu tempo e conhecimento para avaliar a minha disertação.

Eu também gostaria de agradecer à minha familia pelo suporte, amor, paciência neste periodo tão tumultuado. Principalmente a minha irmã que teve a grande paciência de escutar diversas vezes a minha apresentação e relê essa dissertação.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

Resumo

M L Luz, Eric; Caarls, Wouter. **Estudo de técnicas de aprendizado por reforço aplicadas ao controle de processos químicos**. Rio de Janeiro, 2021. 91p. Dissertação de Mestrado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

A indústria 4.0 impulsionou o desenvolvimento de novas tecnologias para atender as demandas atuais do mercado. Uma dessas novas tecnologias foi a incorporação de técnicas de inteligência computacional no cotidiano da indústria química. Neste âmbito, este trabalho avaliou o desempenho de controladores baseados em aprendizado por reforço em processos químicos industriais. A estratégia de controle interfere diretamente na segurança e no custo do processo. Quanto melhor for o desempenho dessa estrategia, menor será a produção de efluentes e o consumo de insumos e energia. Os algoritmos de aprendizado por reforço apresentaram excelentes resultados para o primeiro estudo de caso, o reator CSTR com a cinética de Van de Vusse. Entretanto, para implementação destes algoritmos na planta química do *Tennessee Eastman Process* mostrou-se que mais estudos são necessários. A fraca ou inexistente propriedade Markov, a alta dimensionalidade e as peculiaridades da planta foram fatores dificultadores para os controladores desenvolvidos obterem resultados satisfatórios. Foram avaliados para o estudo de caso 1, os algoritmos *Q-Learning*, *Actor Critic* TD, DQL, DDPG, SAC e TD3, e para o estudo de caso 2 foram avaliados os algoritmos CMA-ES, TRPO, PPO, DDPG, SAC e TD3.

Palavras-chave

Aprendizado por Reforço; Aprendizado por Reforço Profundo; Controle de Processos Químicos; Reator de Van de Vusse; Tennessee Eastman Process; Q-Learning; Actor Critic; Deep Q-Learning; DDPG; TD3; SAC.

Abstract

M L Luz, Eric; Caarls, Wouter (Advisor). **Study of reinforcement learning techniques applied to the control of chemical processes**. Rio de Janeiro, 2021. 91p. Dissertação de Mestrado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

Industry 4.0 boosted the development of new technologies to meet current market demands. One of these new technologies was the incorporation of computational intelligence techniques into the daily life of the chemical industry. In this context, this present work evaluated the performance of controllers based on reinforcement learning in industrial chemical processes. The control strategy directly affects the safety and cost of the process. The better the performance of this strategy, the lower will be the production of effluents and the consumption of input and energy. The reinforcement learning algorithms showed excellent results for the first case study, the Van de Vusse's reactor. However, to implement these algorithms in the Tennessee Eastman Process chemical plant it was shown that more studies are needed. The weak Markov property, the high dimensionality and peculiarities of the plant were factors that made it difficult for the developed controllers to obtain satisfactory results. For case study 1, the algorithms Q-Learning, Actor Critic TD, DQL, DDPG, SAC and TD3 were evaluated, and for case study 2 the algorithms CMA-ES, TRPO, PPO, DDPG, SAC and TD3 were evaluated.

Keywords

Reinforcement Learning; Deep Reinforcement Learning; Chemical Process Control; Van de Vusse's Reactor; Tennessee Eastman Process; Q-Learning; Actor Critic; Deep Q-Learning; DDPG; TD3; SAC.

Sumário

Lista de figuras

Lista de tabelas

Lista de Abreviaturas

AC	*Actor Critic*
APR	Aprendizado por Reforço
APRP	Aprendizado por Reforço Profundo
CMA-ES	*Covariance matrix adaptation evolution strategy*
CSTR	*Continuous Stirred-Tank Reactor*
DDPG	*Deep Deterministic Policy Gradient*
DQL	*Deep Q-Learning*
GRL	*Generic Reinforcement Learning Library*
MDP	Processo de Decisão de Markov
MPC	*Model Predictive Control*
PFR	*Plug Flow Reactor*
PID	Controlador Proporcional, Integral e Derivativo
PPO	*Proximal Policy Optimization*
Q	Função de Valor de Ação
QL	*Q-Learning*
RVV	Reator com Cinética de Van de Vusse
S1	Seleção de Variáveis 1
S2	Seleção de Variáveis 2
SAC	*Soft Actor Critic*
Sp	*Setpoint*
SPIN	*Spinning Up*
TD	Diferença Temporal
TD3	*Twin Delayed DDPG*
TE	*Tennessee Eastman Process*
TRPO	*Trust Region Policy Optimization*
V	Função de Valor Estado

Viver
E não ter a vergonha
De ser feliz
Cantar e cantar e cantar
A beleza de ser
Um eterno aprendiz

Gonzaguinha, .

1
Introdução

A constante procura pela melhora da qualidade de seus produtos, a redução dos custos e o racionamento de matérias-primas são necessidades das indústrias para se adequarem à nova dinâmica mundial. Neste âmbito surge a indústria 4.0, como forma de agregar essas novas necessidades de forma integrada e inteligente.

Uma alternativa adotada nas últimas décadas é a automação e a implementação de estratégias de controle nas linhas industriais. Em relação à indústria química, a estratégia de controle impacta diretamente na segurança do processo, no consumo de insumos e energia, na produção de efluentes, na quantidade de etapas do processo e na qualidade e no valor do produto final.

1.1
Motivação

A estratégia de controle utilizada amplamente nas indústrias é o controlador Proporcional, Integral e Derivativo (PID). Porém, sua sintonia depende do grau de linearidade do processo, o qual é função das propriedades termofísicas e reológicas. Dependendo do grau de linearidade do processo, o uso do controlador PID pode não ser satisfatório.

Outra abordagem utilizada para controlar processos químicos complexos é o *model predictive control* (MPC). Este controlador utiliza um modelo para prever o comportamento futuro do sistema. Com base nestas previsões e um algoritmo de otimização, estabelece um conjunto de ações ótimas. Dependendo da complexidade do modelo ou do horizonte de predição este algoritmo pode apresentar um elevado custo computacional.

Os controles avançados baseados em técnicas de inteligência computacional podem ser uma das alternativas para processos que possuam alta complexidade e considerável grau de não linearidade. Em processos químicos com reações químicas, estas peculiaridades ocorrem devido à não linearidade da cinética química e a possibilidade de reações químicas em série e em paralelo.

Uma das técnicas de inteligência computacional mais relevantes na área de controle é a aprendizagem por reforço (APR). Esta técnica é baseada em teorias da psicologia sobre o aprendizado. Ela tenta reproduzir a forma de como

os seres humanos adquirem o conhecimento e tomam decisões, mas também possui uma forte base matemática na área de "*operations research*".

Com essa abordagem, o controle do processo é implementado através de uma política que mapeia os estados do processo em probabilidades de ações. O objetivo de um algoritmo APR é obter a política que maximiza uma função de valor, que é uma previsão das somas das recompensas futuras esperadas a partir de um determinado estado.

Um dos desafios na utilização de aprendizagem por reforço é o estabelecimento de uma função de recompensa. Essa função deve equilibrar as variáveis de processos de modo a otimizar a segurança, a produção e a qualidade do produto. Outro ponto que a função de recompensa impacta é na convergência do algoritmo de APR. O grau de complexidade da função pode facilitar ou até impossibilitar a convergência.

O avanço tecnológico na área da computação permitiu a viabilidade e a popularização das técnicas de *Deep-Learning*. Assim, uma nova área de pesquisa envolvendo algoritmos de aprendizado por reforço abriram novas possibilidades de soluções para problemas altamente complexos e de grande dimensionalidade.

Este avanço permitiu aplicar algoritmos de APR em processos químicos de alta dimensionalidade. Possibilitando o desenvolvimento de controladores baseados em APRP integrais, capazes de controlar vários equipamentos de processos químicos simultaneamente. Essa estrategia de controle integrada pode permitir uma atuação mais precisa e visando o processo na totalidade, diferentemente de controladores univariáveis que se limitam a um valor preestabelecido.

1.2 Objetivos

Este trabalho tem como objetivo principal desenvolver e avaliar controladores baseados em aprendizagem por reforço para processos envolvendo reações químicas, comparando o desempenho de diferentes algoritmos no controle dos estudos de casos. Os objetivos específicos propostos são:

1. Avaliar o desempenho de diferentes algoritmos de APR para estudo de caso 1: Reator de Van de Vusse;
2. Avaliar o desempenho de diferentes algoritmos de APRP para estudo de caso 2: *Tennesse Eastman Process*;
3. Avaliar abordagens para melhorar o desempenho em ambientes parcialmente observados;
4. Comparar o desempenho entre os algoritmos de APRP e a estratégia evolutiva no estudo de caso 2: *Tennesse Eastman Process*;

1.3 Contribuições

Nesta dissertação foi avaliado à possibilidade do emprego de modelos de controladores baseados em aprendizado por reforço em processos químicos. Essa análise, contemplou tanto controladores para um único equipamento quanto para controladores integrados de uma subunidade de uma planta química. Além disso, foram apresentadas algumas abordagens como *window* (uma janela do histórico de entrada) e *stride* (atraso entre obtenção dos dados) para trabalhar com sistemas parcialmente observáveis e com grande inércia. Estas duas características dificultam consideravelmente a utilização dos algoritmos de aprendizado por reforço.

Estudo de caso 1: Reator de Van de Vusse

1. Desenvolvimento de controladores multivariáveis capazes de simultaneamente otimizar a produção e a diminuição do erro operacional;
2. Avaliação da viabilidade da utilização de algoritmos de APR clássico e profundo em um sistema químico relativamente pequeno com pontos de inversão;

Estudo de caso 2: *Tennesse Eastman Process*

1. Avaliação de controladores integrados, capazes de encontrar novos pontos de operação com maior produção e menor custo.

2. Avaliação da utilização dos artifícios *window* e *stride* em um problema de grande dimensionalidade, parcialmente observáve e com grande inércia.

1.4 Estrutura da Dissertação

O conteúdo desta dissertação foi dividido em 7 capítulos. Os conceitos básicos de aprendizagem por reforço são apresentados no capítulo 2. Também são abordados neste capítulo, os algoritmos de APR clássicos como o *Q-learning* e *Actor-Critic*. No contexto de *Deep-learning*, são apresentados os algoritmos *off-policy* DQL, DDPG, SAC e TD3 e os *on-policy* TRPO e PPO.

O capítulo 3 apresenta uma revisão bibliográfica sobre a utilização de aprendizagem por reforço em processos químicos. Neste capítulo são apresentados os dois estudos de caso utilizado neste trabalho. O primeiro é o reator CSTR com cinética de Van de Vusse, um conhecido *benchmark* de controle de processos químicos. O segundo é o *Tennessee Eastman Process*, um modelo mais complexo baseado em uma planta de processo químico.

O capítulo 4 trata da metodologia adotada para o desenvolvimento dos modelos de processos químicos e dos algoritmos de controle baseados em APR. As simulações elaboradas para avaliar os desempenhos dos controladores encontram-se listadas no capítulo 5.

Os resultados das simulações e avaliação do desempenho de cada controlador desenvolvido são apresentados no capítulo 6. No capítulo 7 são apresentados as conclusões finais e sugestões para trabalhos futuros.

2
Fundamentos Teóricos: Aprendizado por Reforço

Das técnicas de *machine learning*, a aprendizagem por reforço é um dos métodos que mais se assemelham na forma de como os seres humanos adquirem conhecimentos e tomam decisões. Esta técnica é baseada em uma das teorias da psicologia sobre o aprendizado infantil. Uma criança nos seus primeiros anos de vida aprende sobre o ambiente em que está inserido através de estímulos. Esses estímulos são interações entre o ambiente e a criança. Por exemplo, quando uma criança deixa seus brinquedos bagunçado no quarto, normalmente a repreendemos. Em outras palavras, um estímulo negativo sobre o estado em que o quarto está após ação de deixar os brinquedos bagunçados. Em contrapartida, quando a criança arruma o quarto dela, normalmente a parabenizamos, isto é, um estímulo positivo sobre o estado em que o quarto está após a arrumação. Deste modo, o algoritmo de aprendizado por reforço aprende o que fazer, isto é, ele mapeia um conjunto de ações de modo a maximizar um sinal numérico de recompensa.

Um algoritmo de aprendizado por reforço é composto por um agente que aprende a executar certas ações em um ambiente, visando à maximização da soma das recompensas futuras. Ele faz isso explorando o conhecimento que aprende através de repetidas tentativas. Num problema clássico de aprendizado por reforço, o agente executa a ação a_t no tempo t e recebe como *feedback* do ambiente, uma recompensa r_{t+1} e um novo estado do ambiente s_{t+1} (Figura 2.1).

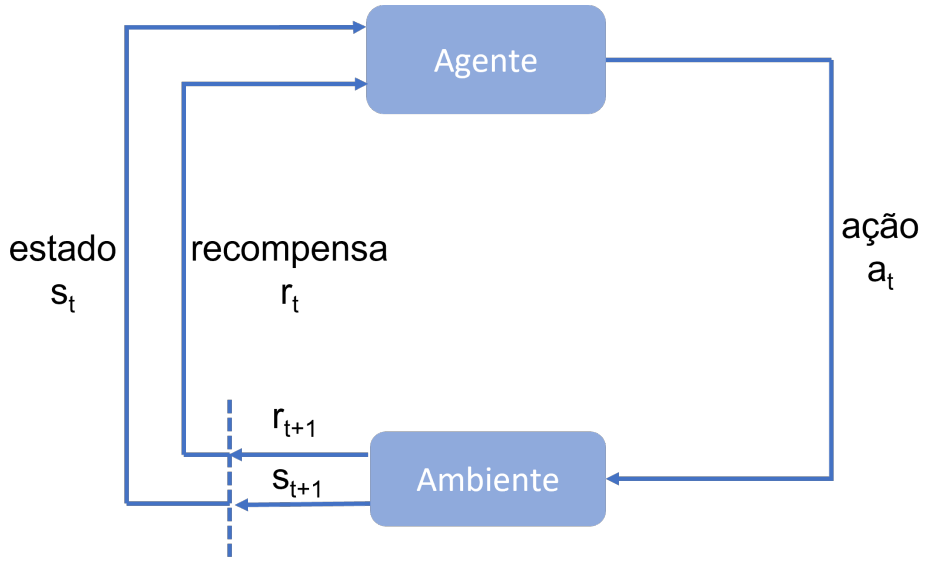

Figura 2.1: Fluxo de trabalho de um algoritmo de APR [1]

O aprendizado por reforço difere do aprendizado supervisionado. Neste tipo, o aprendizado ocorre a partir de exemplos fornecidos por um supervisor externo experiente onde cada entrada de dados tem uma determinada saída. Esse método é importante, entretanto não é adequado para aprender com interações. Em problemas interativos, muitas vezes é impraticável obter exemplos de comportamentos desejados que sejam corretos e representativos de todas as situações em que o agente deve agir. Em uma situação desconhecida, um agente deve ser capaz de aprender com sua própria experiência.

As duas principais características do APR são o aprendizado por tentativa e erro, e as recompensas atrasadas. Nos casos mais desafiadores, as ações podem afetar não apenas a recompensa imediata, mas também todas as recompensas subsequentes.

Um dos desafios que surgem no aprendizado por reforço é o *trade-off* entre "*exploration*" (exploração) e "*exploitation*" (extração). Para maximizar a soma das recompensas futuras, um agente tende de adotar uma postura gananciosa, preferindo ações que obtiveram altas recompensas no passado. Mas, para descobrir tais ações, ele precisa tentar novas ações e explorar mais estados. O agente deve aproveitar o que já sabe para obter maiores recompensas, mas também deve explorar para selecionar as melhores ações no futuro. O agente deve tentar uma variedade de ações e favorecer progressivamente aquelas que parecem ser as melhores. Em uma tarefa estocástica, cada ação deve ser tentada várias vezes para obter uma estimativa confiável de sua recompensa esperada.

Posteriormente, serão apresentados os conceitos básicos de APR e alguns algoritmos utilizados neste trabalho. Estes algoritmos foram agrupados

em aprendizado por reforço por diferenças temporais, aprendizado por reforço profundo e a estratégia evolutiva. Este último algoritmo não pertence à classe de algoritmos de APR, entretanto foi adicionado ao trabalho por apresentar características semelhantes aos demais algoritmos, podendo apresentar resultados promissores.

2.1 Conceitos básicos

Nesta seção serão apresentados os conceitos básicos de aprendizado por reforço. A maioria dos conteúdos desta seção foram baseados no capítulo 3 do livro do Sutton e Barto [1]. Primeiramente será denotado o conceito de Processo de Decisão de Markov (MDP). Posteriormente serão comentados a Equação de Belman, Função Valor de Estado e a Função Ação de Valor.

2.1.1 Processo de Decisão de Markov (MDP)

No fluxograma de aprendizado por reforço, o agente determina as suas ações em função de um sinal do ambiente denominado estado do ambiente (Figura 2.1).

Esses sinais podem estar além das percepções imediatas (medições sensoriais). As representações desses estados podem ser versões processadas de percepções originais ou podem ser estruturas complexas construídas ao longo do tempo a partir da sequência de percepções.

Deseja-se um sinal de estado que resuma as percepções passadas de forma compacta, preservando as informações relevantes. Isso normalmente requer mais do que as percepções imediatas e menos do que o histórico completo de todas as percepções. Um sinal de estado que consegue reter todas as informações relevantes, possui a propriedade de Markov.

Pode-se definir a dinâmica de um sistema finito, apenas especificando a distribuição de probabilidade (Equação 2-1).

$$P(s_{t+1} = s', r_{t+1} = r' | s_t, a_t, r_t, s_{t-1}, a_{t-1}, r_{t-1}, ..., s_0, a_0, r_1) \qquad (2\text{-}1)$$

Para todos os s', r e os valores possíveis dos eventos passados: $s_t, a_t, r_t, s_{t-1}, a_{t-1}, r_{t-1}, ..., s_0, a_0, r_1$. Se o sinal de estado tem a propriedade Markov, a resposta do ambiente em t +1, depende apenas das representações de estado e ação em t (Equação 2-2).

$$P(s_{t+1} = s', r_{t+1} = r'|s_t, a_t) \tag{2-2}$$

Em outras palavras, um sinal de estado tem a propriedade de Markov se e somente se Equação 2-1 é igual à Equação 2-2 para todos s' e r.

Se um ambiente tem a propriedade Markov, através de um processo iterativo, pode-se prever o próximo estado e a próxima recompensa esperada, dados o estado e a ação atuais. Ao iterar essa equação, é possível prever todos os estados futuros e as recompensas esperadas a partir do conhecimento apenas do estado atual.

Os estados de Markov fornecem a melhor base possível para a escolha das ações. Ou seja, a melhor política para escolher ações em função de um estado de Markov é tão boa quanto a melhor política para escolher ações em função do histórico completo.

Uma tarefa de aprendizado por reforço que satisfaça a propriedade de Markov é denominado processo de decisão de Markov (MDP). Se os espaços de estado e ação são finitos, então é denominado processo de decisão Markov finito (MDP finito).

Um determinado MDP finito é definido por seus conjuntos de estado e ação e pela dinâmica de uma etapa do ambiente. Dado qualquer estado e ação, s e a, a probabilidade de cada próximo estado possível, s', é dado pela Equação 2-3.

$$P^a_{ss'} = (s_{t+1} = s'|s_t = s, a_t = a) \tag{2-3}$$

Esse valor é denominado probabilidades de transição. Da mesma forma, dado qualquer estado e ações atuais, s e a , com qualquer próximo estado s' , o valor esperado da próxima recompensa é dado pela Equação 2-4.

$$R^a_{ss'} = E(r_{t+1}|s_t = s, a_t = a, s_{t+1} = s') \tag{2-4}$$

2.1.2 Equação de Bellman

A maioria dos algoritmos de aprendizado por reforço são baseados em estimar uma função que supõe a qualidade de um determinado estado, esta função é denominada função de valor estado. A noção de "qualidade" é definida em termos da soma das recompensas futuras esperadas a partir daquele estado.

Esta recompensa futura esperada depende da ação realizada pelo agente. Consequentemente, a função de valor estado é relacionada a uma política específica.

A política π é um mapeamento de cada estado $s \epsilon$ S, e ação $a \epsilon$ A(s), para a probabilidade $\pi(s, a)$ de realizar uma ação a quando no estado s. $V^{\pi}(s)$ é o retorno esperado a partir de s e seguindo a política π. O retorno é a soma das recompensas futuras esperada a partir daquele estado. Para MDPs, pode-se definir $V^{\pi}(s)$ como:

$$V^{\pi}(s) = E(R_t|s_t = s) = E_{\pi}(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}|s_t = s) \tag{2-5}$$

Onde $E_{\pi}(\dots)$ denota o valor esperado, dado o agente seguindo a política π, t é o tempo e γ é fator de desconto das recompensas futuras para evitar retornos infinitos em tarefas continuas. Denomina-se V^{π} como função de valor estado para a política π.

Outra forma de avaliar a qualidade de um estado e de uma ação é através da função valor de ação. $Q^{\pi}(s, a)$ é o retorno esperado a partir do estado s, realizando a ação a e seguindo a política π. Para MDPs, pode-se definir $Q^{\pi}(s, a)$ como:

$$Q^{\pi}(s, a) = E(R_t|s_t = s, a_t = a) = E_{\pi}(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}|s_t = s, a_t = a) \tag{2-6}$$

A Equação 2-6 é denominada como função de valor de ação para a política π (Q^{π}).

Tanto Q^{π} e V^{π} são estimados com base na experiência. Por exemplo, se um agente segue a política π e mantém uma média, para cada estado encontrado, dos retornos reais que seguiram aquele estado, então a média irá convergir para o valor do estado $V^{\pi}(s)$. Se média dos retornos daquele estado para cada ação forem mantidas, então essas médias convergirão de forma semelhante para os valores de $Q^{\pi}(s, a)$.

Os relacionamentos recursivos específicos são uma propriedade fundamental das funções Q^{π} e V^{π} que são utilizadas no aprendizado por reforço. Para qualquer política π e qualquer estado s, a seguinte condição de consistência é válida entre o valor de s e o valor de seus possíveis estados sucessores:

$$V^{\pi}(s) = \sum_{a} \pi(s, a) \sum_{s'} P^{a}_{ss'}[R^{a}_{ss'} + \gamma V^{\pi}(s')] \tag{2-7}$$

Na equação de Bellman para $V^{\pi}(s)$ (Equação 2-7), está implícito que as ações, a, são retiradas do conjunto A(s) e os próximos estados, s', são retirados do conjunto S. Essa equação calcula a média de todas as possibilidades, ponderando cada uma por sua probabilidade de ocorrência. A função de valor associado ao estado inicial s deve ser igual ao valor (descontado) do próximo estado esperado, mais a recompensa esperada ao longo do caminho.

Resolver um problema de APR significa encontrar a política que proporcione maiores retornos a longo prazo. Uma política π é considerada superior a uma política π', se seu retorno esperado for maior ou igual ao de π' para todos os estados. Sempre existe pelo menos uma política que é melhor ou igual a todas as outras políticas. Denomina-se esta política como uma política ótima π^*. As funções V e Q desta política ótima são denotadas como V^* e Q^*.

$$V^*(s) = \max_{a \epsilon A(s)} Q^*(s, a) \tag{2-8}$$

A equação de Bellman para Q^* ótima é:

$$Q^*(s) = \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma \max_{a'} Q(s_{t+1}, a')] \tag{2-9}$$

2.2 Algoritmos de Aprendizado por Reforço por Diferença Temporais

Nesta seção serão apresentados de forma resumida os algoritmos de aprendizado clássicos. Estes algoritmos são os precursores dos algoritmos de aprendizado por reforço profundo que serão discutidos na subsecção 2.3. Primeiramente, será relatado o algoritmo *Q-Learning* (QL) desenvolvida por Watkins em 1989. Por último, será descrito as principais características do *Actor Critic* (AC).

2.2.1 QL: Q-Learning

O *Q-learning* é um dos algoritmos clássicos de aprendizado por reforço. As suas grandes vantagens são sua simplicidade e seu baixo custo computacional. O *Q-learning* é algoritmo de controle TD (diferenças temporais) *off-policy*. Um algoritmo *off-policy* aprende a política ótima, independentemente das ações do agente. Ele atualiza a função de valor usando ações que seriam da política ótima, o qual pode diferir da ação seguida pelo agente.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t)] \tag{2-10}$$

Onde Q é função de valor de ação, a_t é a ação atual, s_t é o estado atual, s_{t+1} é o próximo estado, a' é a próxima ação, γ é o fator de desconto, definido entre 0 e 1. Esse fator modela quanto as recompensas futuras valem menos que recompensas imediatas. α é a taxa de aprendizado, definida geralmente entre 0 e 1. $\max_{a'} Q(s_{t+1}, a')$ é a ação que maximiza a estimação da função Q atual, dado o estado s_{t+1}.

Esse algoritmo mapeia qual o retorno de uma ação em determinado estado e os organiza em uma tabela, denominada *Q-Table*. Desse modo, o agente precisa apenas consultar a tabela para escolher uma ação dado um estado. A Equação 2-10 atualiza os dados da *Q-Table*.

Em ambiente de estados contínuos, pode-se transformar a atualização da *Q-Table* em um problema de ajuste de função, de modo a obter pares de estados e ações semelhantes. Em outras palavras, cria-se uma função Q que se aproxima do valor da função Q^* ideal, atualizando o parâmetro θ (Equação 2-11).

$$Q(s, a; \theta) \approx Q^*(s, a) \tag{2-11}$$

Durante o aprendizado, é importante equilibrar a maximização das recompensas acumuladas com a exploração, o dilema do *trade-off* entre *exploration* e *exploitation*. O ε-*greddy* é dos algoritmos mais simples para resolver este dilema. Nele se estabelece uma probabilidade ε de executar uma ação aleatória (Equação 2-12).

$$\pi(a) = \begin{cases} 1 - \varepsilon - \dfrac{\varepsilon}{|A|} & \text{Se } a = \max_{a'} Q(a') \\ \dfrac{\varepsilon}{|A|} & \text{Se não} \end{cases} \tag{2-12}$$

Durante a progressão do algorítimo, a função Q aprendida se aproxima gradualmente da função de valor de ação ótima Q^*, independentemente da política que está sendo seguida. Isso simplifica drasticamente a análise do algoritmo e permite provas de convergência iniciais. A forma procedimental do *Q-learning* é mostrada no pseudocódigo 1.

Algoritmo 1: Q-Learning

```
início
    Inicialização arbitraria de Q(s,a)
    repita
        Inicialize um estado s ∈ S
        repita
            Execute uma ação a ∈ A baseado em Q(s, ·)
            Observe o r, s'
            Q(s,a)← Q(s, a) + α[r + γ max_a' Q(s', a') − Q(s, a)]
            s← s'
        até término do episódio
    até fim de todos episódios
fim
```

O *Q-learning* foi desenvolvido por Watkins em 1989 em sua tese de doutorado. Posteriormente, foram comprovados sua convergência por Watkins e Dayan (1992) [2, 3].

2.2.2
AC: Actor Critic

Os algoritmos *Actor-Critic* possuem uma estrutura de memória separada para representar de forma independente a política da função de valor. A estrutura da política é conhecida como ator, porque é utilizada para selecionar ações. A função valor é conhecida como crítico, porque critica as ações realizadas pelo ator.

Neste trabalho será utilizado o *Actor-Critic* por diferenças temporais (TD) baseado em gradiente de política. O crítico assume a forma de um erro de TD. Esse sinal escalar é a única saída do crítico e direciona o aprendizado tanto do ator quanto do crítico (Figura 2.2).

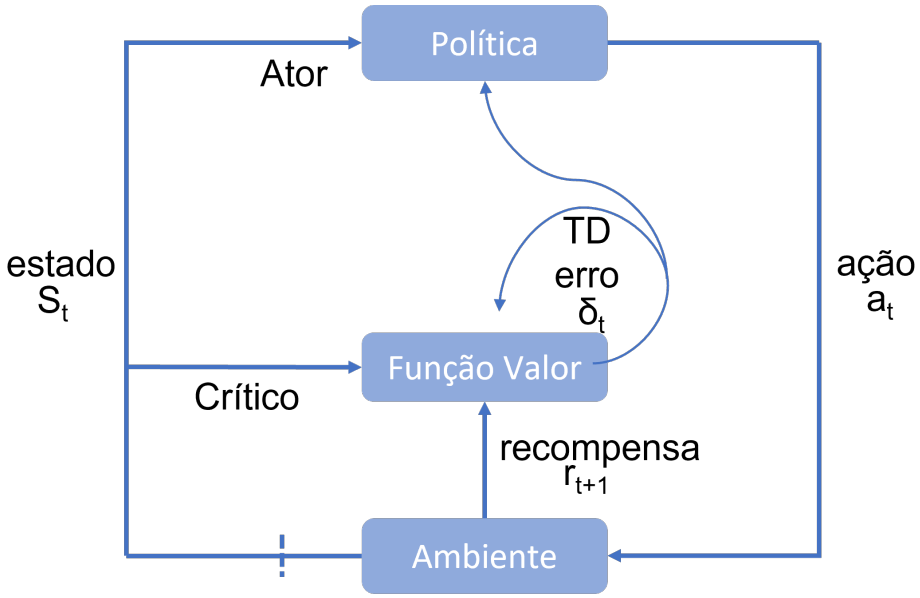

Figura 2.2: Arquitetura do *Actor-Critic*.

Após seleção de cada ação, o crítico avalia o novo estado para determinar se houve melhora ou não. Essa avaliação é denominada como o erro TD (Equação 2-13).

$$\delta_t = r_{t+1} + \gamma V(s_{t+1}) - V(s_t) \tag{2-13}$$

Onde V é a função valor implementada pelo crítico e γ é a taxa de desconto. Se o erro TD for positivo, sugere que a tendência deve ser fortalecida. Enquanto se o erro TD for negativo, sugere que a tendência deve ser enfraquecida. Neste presente trabalho será utilizado o *TD Actor-Critic* cujo gradiente de política encontra-se na equação 2-15. Este gradiente serve para

calcular os parâmetros da função aproximadora da função valor (Equação 2-14).

$$V(s;w) \approx V^*(s) \tag{2-14}$$

$$\nabla_\theta J(\theta) = E_{\pi\theta}[\nabla_\theta log_{\pi\theta}(s,a)\delta] \tag{2-15}$$

Onde $\nabla_\theta J(\theta)$ é o gradiente da política, δ é o erro TD, s é o estado, a é a ação, w é o vetor de parâmetros da função aproximadora de V e θ é o vetor de parâmetros da função aproximadora da política. O pseudocódigo 2 mostra a forma procedimental do algoritmo *TD actor-critic*.

Algoritmo 2: TD Actor Critc

```
início
    θ, w ← inicialização arbitrária
    repita
        Inicialize um estado s ∈ S
        repita
            Execute uma ação a ~ π(a|s;θ)
            Observe o r, s'
            δ_t = r_{t+1} + γV(s_{t+1}) − V(s_t)
            w = w + α∇_w V(s;w)δ
            θ = θ + β∇_θ V(s;θ)δ
            s ← s'
        até término do episódio
    até fim de todos episódios
fim
```

Onde α é a taxa de aprendizado do aproximador da função valor, β é a taxa de aprendizado do aproximador da política, γ é o fator de desconto, w é o vetor de parâmetros da função aproximadora de V e θ é o vetor de parâmetros da função aproximadora da política.

Os modelos *Actor-Critic* utilizando diferenças temporais foram primeiramente estudadas por Witten (1977) e posteriormente por Barto, Sutton e Anderson (1983; Sutton, 1984), que introduziram o uso dos termos "*actor*" e "*critic*". [1]

2.3 Algoritmos de Aprendizado por Reforço Profundo

Nesta seção serão apresentados de forma resumida os algoritmos de aprendizado profundo utilizados neste presente trabalho. A explicação será focada em suas principais característica e as contribuições destes modelos para a evolução do aprendizado por reforço profundo. Primeiramente será relatado o algoritmo *Deep Q-Learning* (DQL) desenvolvida pela DeepMind em 2013 [4, 5]. Posteriormente serão apresentados o *Deep Deterministic Policy Gradient* (DDPG) [7, 8], o *Twin Delayed DDPG* (TD3) [9] e o *Soft Actor Critic* (SAC) [10, 11] elaborados por Silver em 2014 e Fujimoto em 2018 e Haarnoja em 2018, respectivamente. Por último, será descrito as principais características dos algoritmos *on-policy Trust Region Policy Optimization* (TRPO) e *Proximal Policy Optimization* (PPO).

2.3.1 DQL: Deep Q-Learning

O algoritmo de *Q-Learning* cria uma tabela, a *Q-Table*, onde o agente consulta qual ação a ser executada para maximizar a soma das recompensas a longo prazo durante seu aprendizado. Em ambientes contínuos e com alta dimensionalidade de estados e ações, essa abordagem se torna ineficiente. Por exemplo, em um ambiente com 10.000 estados e com 5.000 ações por estado. Isso criaria uma tabela de 50 milhões de elementos.

O *Deep Q-Learning* utiliza uma rede neural profunda como aproximador da função Q para lidar com estes problemas. Ele cria uma rede neural que se aproxima do valor da função Q^* ideal atualizando os pesos sinápticos. Essas redes conseguem extrair automaticamente características implícitas do problema, além de serem capazes de trabalhar com espaços contínuos de alta dimensionalidade.

Em 2013, a DeepMind apresentou uma versão profunda do *Q-learning* capaz de jogar um videogame ATARI, o *Deep Q-Learning* (DQL). Eles usaram uma rede neural convolucional profunda para processar pixels RGB de 4 quadros (84x84) como entradas e a pontuação do jogo como recompensa. Esta abordagem obteve um desempenho comparável ou superior a um jogador humano profissional [4, 5].

Os autores propuseram uma rotina de treinamento por *replay* de experiência. Essa técnica conjuga a repetição e a memória na aprendizagem de reforço, onde se armazenam as experiências do agente em cada passo de execução no ambiente. Esse método elimina a correlação na sequência das observações (que pode ser prejudicial para o treinamento das redes) e suaviza as mudan-

ças na distribuição de dados. A Equação 2-16 é a função custo utilizada na atualização dos pesos θ_i da rede neural na i-ésima iteração do algoritmo.

$$L_i(\theta_i) = E_{(s,a,r,s')}[(r + \gamma \max_{a'} Q(s', a'; \theta_i^-) - Q(s, a; \theta_i)^2] \tag{2-16}$$

Este treinamento é normalmente realizado através do algoritmo otimização Adam (*Adaptive Moment Estimation*) [6]. Os parâmetros da rede alvo (θ_i^-) são atualizados a cada C passos com os parâmetros da Rede Q (θ_i) (Figura 2.3). Esta abordagem resulta em um treinamento mais estável devido à manutenção da função destino por um determinado tempo.

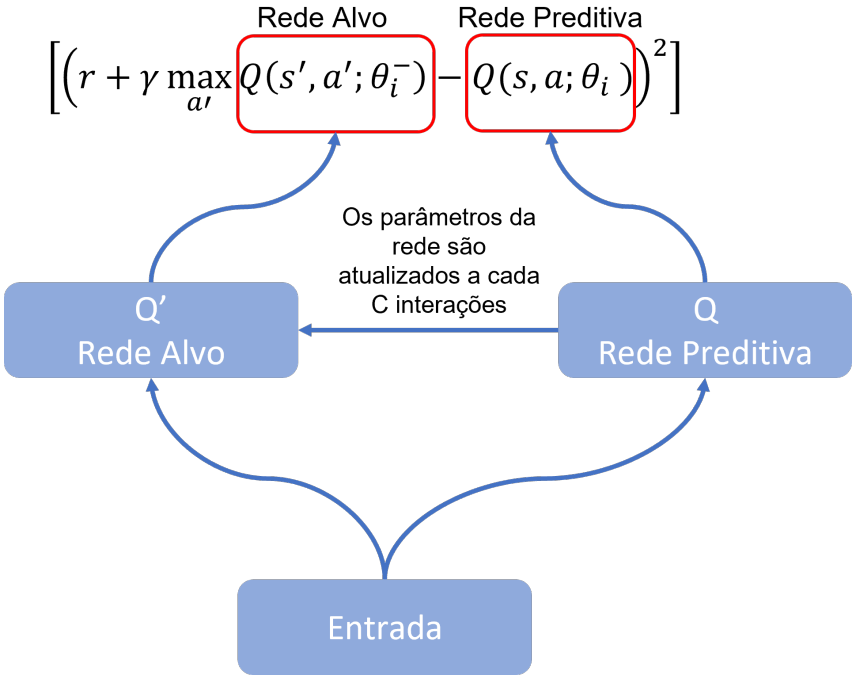

Figura 2.3: Cálculo da Função Perda

A forma procedimental do DQL é mostrado no pseudocódigo 3.

Algoritmo 3: Deep Q-Learning

início
  $\theta \leftarrow 0, \theta^t \leftarrow 0$
  repita
    Inicialize um estado s $\in S$
    repita
      Execute uma ação a $\sim \pi(a|s;\theta)$
      Observe o r, s'
      Adicione a transição (s,a) $\leftarrow (r, s')aD$
      Amostre aleatoriamente um mini-batch B $\subset D$
      repita
        $q_i = r_i + \gamma \max_{a'} Q(s_i, a')\delta$
      até *término das amostras:* $b_i : (s_i, a_i) \rightarrow (r_i, s'_i)$
      Treine Q(θ) para todas amostras $(s_i, a_i) \in B \rightarrow q_i$
      Atualize os pesos da rede Alvo a cada C passos
      $\theta^Q_{alvo} \leftarrow \theta^Q_{pred}$
      s $\leftarrow s', a \leftarrow a'$
    até *término do episódio*
  até *fim de todos episódios*
fim

No estudo dos autores, o DQL apresentou bons resultados em sistemas com dimensões elevadas, entretanto o espaço de ações do método ainda é discreto. Muitas tarefas, inclusive de controle, possuem um espaço de ação contínuo. Se a discretização de tal espaço de ações é muito fina, o espaço de ações torna-se muito grande e a complexidade do problema aumenta. Dificultando a convergência do método, além de aumentar exponencialmente o custo computacional [4, 5].

2.3.2 DDPG: Deep Deterministic Policy Gradient

O *Deep Deterministic Policy Gradient* (DDPG) é um algoritmo de aprendizado por reforço *off-policy* que combina o *Q-Learning* e gradiente de política. Como sendo uma técnica *actor-critic*, o DDPG é constituído de um ator e de um crítico. O ator é a rede neural das políticas. Ela recebe um determinado estado e responde com uma ação exata. O crítico é a rede da função ação de valor $Q^*(s, a)$. Ele assume determinados estado e a ação como

entrada e retorna com o valor da recompensa esperada para aquela entrada de dados.

O DDPG é utilizado em ambientes de ação contínua. O "determinístico" em DDPG se refere ao fato de que o ator calcula a ação exata em vez de uma distribuição de probabilidade sobre as ações como ocorre em outros algoritmos. Com base no conhecimento da função de valor de ação ideal $Q^*(s, a)$, a rede ator μ é capaz determinar a ação ideal, para qualquer estado s, resolvendo a Equação 2-17.

$$\mu(s) = arg\max_a Q^*(s, a) \tag{2-17}$$

Como no DQL, o DDPG utiliza o artifício do treinamento por *replay* de experiências e redes neurais Alvos para garantir estabilidade do aprendizado. São utilizadas redes Alvos tanto para o ator como para o crítico, totalizando quatro redes neurais. As redes Alvo são redes atrasadas em comparação com as redes principais. Os pesos dos alvos são atualizados periodicamente com base nas redes principais. Diferentemente do DQL onde ocorre uma atualização integral dos pesos da rede Alvo, no DDPG essa atualização é uma fração dos pesos da rede principal (Equação 2-19).

$$\theta^{\mu}_{alvo} \leftarrow \tau\theta^{\mu}_{alvo} + (1-\tau)\theta^{\mu} \tag{2-18}$$

$$\theta^{Q}_{alvo} \leftarrow \tau\theta^{Q}_{alvo} + (1-\tau)\theta^{Q} \tag{2-19}$$

Os treinamentos da rede Crítico (Q) e da rede Ator (μ) ocorrem pelas funções de perdas. A Equação 2-21 é a perda do ator. Ela é a soma dos valores de Q dos estados onde se utiliza a ação calculada pela rede neural Ator. A perda do crítico é um erro TD, utilizado pelas redes Alvos para calcular o valor Q para o próximo estado (Equação 2-20).

$$J_Q = \frac{1}{N}\sum_{i=1}^{B}(r_i + \gamma Q_{alvo}(s'_i, \mu_{alvo}(s'_i)) - Q(s_i, \mu(s_i)))^2 \tag{2-20}$$

O aprendizado deste algoritmo ocorre pela minimização da perda do crítico e maximização da perda do ator. No espaço de ações continuas, presume-se que a função $Q(s, a)$ seja diferenciável em relação ao argumento da ação. Isso permite definir uma regra de aprendizado eficiente baseado no gradiente da política (Equação 2-21).

$$J_\mu = \frac{1}{N}\sum_{i=1}^{|B|} Q(s_i, \mu(s_i)) \tag{2-21}$$

A forma procedimental do DDPG é mostrada no pseudocódigo 4.

Algoritmo 4: Deep Deterministic Policy Gradient

início
- inicialize $\tau << 1$
- inicialize arbitrariamente os pesos da redes Q e μ
- **repita**
  - Inicialize um estado s $\in S$
  - Inicialize o processo aleatório de exploração N
  - **repita**
    - Execute uma ação a $\sim \pi(a|s;\theta) + N$
    - Observe o r, s'
    - Adicione a transição (s,a) $\leftarrow (r, s')$ *a* D
    - Amostre aleatoriamente um mini-batch B $\subset D$
    - Calcules todos alvos
      y(r,s')=r$_i$ $+ \gamma Q_{alvo}(s'_i, \mu_{alvo}(s'_i)) - Q(s_i, \mu(s_i))$
    - Atualize os crítico pelo gradiente descendente
      $\nabla_{\theta^Q} \frac{1}{|B|} \sum_{(s,a,r,s')\epsilon B}(Q_{\theta^Q}(s,a) - y(r,s')^2$
    - Atualize o ator pelo gradiente ascendente
      $\nabla_{\theta^\mu} \frac{1}{|B|} \sum_{s\epsilon B} Q_{\theta^Q}(s, \mu_{\theta^\mu}(s))$
    - Atualize os pesos das rede alvos a cada C passos
      $\theta^\mu_{alvo} \leftarrow \tau\theta^\mu_{alvo} + (1-\tau)\theta^\mu$
    - $\theta^Q_{alvo} \leftarrow \tau\theta^Q_{alvo} + (1-\tau)\theta^Q$
    - s $\leftarrow s'$
  - **até** *término do episódio*
- **até** *fim de todos episódios*

fim

A teoria subjacente aos gradientes de política determinística foi estabelecida por Silver et al. em 2014 [7]. Lillicrap et al. 2016, adaptou o algoritmo DPG para a configuração de aprendizado por reforço profundo (APRP), desenvolvendo o DDPG [8]. Este algoritmo serviu como base para algoritmo de APRP de ações contínuas como o TD3 e o SAC.

2.3.3 TD3: Twin Delayed DDPG

O *Twin Delayed DDPG* (TD3) é um algoritmo *off-policy* que combina métodos de gradiente de política, *Actor-Crític* e *Double Deep Q-Learning* contínuo. O DDPG consegue obter um ótimo desempenho, entretanto ele costuma ser frágil em relação a hiper parâmetros e outras categorias de ajuste. Uma das falhas comum do DDPG é que a função Q aprendida começa a superestimar drasticamente os valores Q. Este comportamento leva à quebra da política, porque explora os erros na função Q. O *Twin Delayed DDPG* adiciona três artifícios para solucionar os problemas relatados acima.

O primeiro recurso adicionado ao TD3 é o uso de duas redes críticas. Ele utiliza a aprendizagem de Q duplo cortada onde considera o menor valor das duas redes críticas para atualização da rede Alvo. Este recurso favorece a subestimação dos valores de Q, ocasionando uma aproximação mais estável, melhorando assim a estabilidade de todo o algoritmo.

O segundo recurso adicionado ao TD3 é o atraso na atualização da política. A utilização das redes Alvos e das principais são uma ótima ferramenta para adicionar a estabilidade no treinamento, entretanto em métodos *Actor-Crític* essa estratégia pode ocasionar alguns problemas devido à interação entre as redes neurais Ator e Crítico. O treinamento pode divergir quando uma política ruim é superestimada devido à atualização conter muitos erros.

Uma solução apontada pelo autor deste algoritmo é a redução na frequência de atualização da rede Ator. Essa estratégia resulta em uma maior estabilidade da rede Ator e reduz os erros antes de serem utilizadas na rede Q. As atualizações de política menos frequentes terão estimativa de valor com menor variação, portanto podem resultar em uma política melhor.

O terceiro e último recurso adicionado ao TD3 é a regularização de ruído. Os métodos de política determinística durante a atualização da rede crítica tendem a produzir redes Alvos com alta variação. No DDPG, se o aproximador da função Q desenvolver um pico agudo incorreto para algumas ações, a política explorará rapidamente esse pico e terá um comportamento anormal. Para reduzir essa variação, o TD3 usa uma técnica de regularização conhecida como suavização de política alva. Essa técnica constitui na adição de uma pequena quantidade de ruído aleatório ϵ ao alvo a partir da média dos *mini-batch* (Equação 2-22). Adição deste ruído tende a tornar a política mais estável, suavizando as mudanças nos valores da ação.

$$a'(s') = corte(\mu_{\theta alvo}(s') + corte(\epsilon, -c, c), a_{inferior}, a_{superior}) \qquad (2\text{-}22)$$

Onde a' é próxima ação, $a_{inferior}, a_{superior}$ são limites numéricos da ação, c é o limite do ruído exploratório, s' é próximo estado, $\mu_{\theta alvo}$ é a rede neural Ator Alvo.

Juntos, esses três artifícios resultam em um desempenho substancialmente aprimorado em relação ao DDPG. O TD3 foi apresentado por Fujimoto et al. em 2018. Ele pode ser considerado um dos métodos sucessores do DDPG. Neste artigo, os autores comparam o TD3 com diferentes algoritmos em diferentes cenários implementados no OpenAI Gym. Ele obteve excelentes resultados ficando na frente de algoritmos avançados como PPO, TRPO e outros [10].

2.3.4 SAC: Soft Actor Critc

O *Soft Actor Critic* (SAC) é um algoritmo *off-policy* baseado na estrutura de aprendizado por reforço de entropia máxima. Nessa estrutura, o ator visa maximizar simultaneamente a recompensa e a entropia. Para este contexto, pode-se simplificar o termo "entropia" como a imprevisibilidade de uma variável aleatória. Se uma variável aleatória sempre assume um único valor, então ela possui entropia zero, porque ela é previsível. Se uma variável aleatória pode ser qualquer número real, então ela tem uma entropia elevada.

Ao maximizar entropia, a política é incentivada a explorar mais amplamente o ambiente, enquanto desiste de caminhos claramente pouco promissores. A política consegue reconhecer vários modos de comportamento quase ótimos. Em problemas onde várias ações parecem igualmente esplendidas, a política atribuirá igual massa de probabilidade a essas ações. Deste modo é garantido que ela não entrará em colapso devido à seleção repetida de uma ação particular. Assim, ocasionando a exploração de alguma inconsistência na função Q aproximada. A função objetivo do SAC consiste em um termo de recompensa e um termo de entropia H ponderado por α (Equação 2-23).

$$J(\pi) = \sum_{t=0}^{T} E_{(s_t,a_t)}[r(s_t, a_t) + \alpha H(\pi(.|s_t))] \tag{2-23}$$

Outro recurso introduzido pelo SAC é a Iteração de Política Suave. Ela adiciona o termo de entropia $H = -log\pi(a|s)$ à equação de Bellman. O agente busca maximizar a soma das recompensas esperada do ambiente e a entropia da política (Equação 2-24).

$$Q(s_t, a_t) = r(s_t, a_t) + \gamma E_{a' \approx \pi}[Q(s_{t+1}, a') - log\pi(a'|s_t)] \quad (2\text{-}24)$$

A estrutura do *Soft Actor Critic* é constituída: de uma rede neural para aprender a política, o ator. Como utiliza o artifício do *Double Deep Q-Learning*, possui quatro redes neurais para a função Q, as redes Q1 e Q2 e suas respectivas redes alvos. Três otimizadores para ajustar os pesos das redes ator, Q1, Q2 principais, similar a estrutura do DDPG. Além disso, exite um otimizador para o ajuste do parâmetro α da Equação 2-23.

O *Soft Actor Critic* foi apresentado por Haarnoja et al., em 2018. Neste artigo, os autores do SAC apresentaram evidências na melhora da velocidade de aprendizagem, em relação a outros aos métodos de aprendizado por reforço [10]. No mesmo ano, foi publicado outro artigo focado nas aplicações do SAC. Os testes foram realizados no ambiente OpenAI Gym e no controle de um robô *Minitaur* no mundo real. O SAC obteve excelentes resultados ficando na frente de algoritmos avançados de aprendizado por reforço profundo [11].

2.3.5 TRPO: Trust Region Policy Optimization

O *Trust Region Policy Optimization* (TRPO) é um algoritmo *on-policy* que combina o método de gradiente de política conjugado e a pesquisa de linha de retrocesso. Este algoritmo pondera a taxa de atualização da política de modo a elevar o desempenho do aprendizado e garantir a estabilidade ao sistema. Um dos pontos que possibilitam este comportamento é a utilização do *KL-Divergence*, que quantifica e limita a proximidade entre a nova e antiga políticas. Essa restrição se assemelha a uma medida de distância entre distribuições de probabilidade.

No método de gradiente de política tradicional, existe uma certa proximidade entre as políticas novas e as antigas no espaço de parâmetros. Essas diferenças aparentemente pequenas podem gerar grandes impactos no desempenho, portanto, uma única etapa incorreta pode prejudicar o desempenho da política, assim prejudicando amostragem. Isso torna perigoso usar tamanhos de passo grandes com gradientes de política tradicional. O TRPO evita esse problema devido à incorporação de restrições na atualização da política.

Outra peculiaridade do método de gradiente de política implementada no TRPO é a amostragem por importância. Este método, permite a utilização de amostras de uma política antiga para calcular o gradiente da política. Esse processo aumenta a eficiência da amostragem, quando se comparando ao método tradicional que a política é alterada dependendo da aquisição de

novas amostras. Amostras antigas não são reutilizáveis nessa alteração.

Outra característica do TRPO é a otimização por região de confiança. Neste método, primeiramente determina-se o tamanho máximo da etapa a ser explorado. Em seguida, localiza-se o ponto ideal dentro dessa região de confiança. Esse ponto ideal será a partida para o cálculo da próxima iteração do otimizador. Essa região de confiança pode encolher ou expandir conforme a diferença entra nova e antiga política. Por exemplo, se a diferença entre a política aumentar consideravelmente, a região de confiança é encolhida.

Schulman em 2015 publicou um artigo onde apresenta e testa o algoritmo TRPO. Neste artigo, ele descreve um procedimento iterativo para otimizar políticas, com melhoria monotônica garantida. Segundo os autores, este algoritmo é semelhante aos métodos de gradiente de política natural sendo eficiente em otimizar grandes políticas não lineares, como as redes neurais. Ele apresentou bons resultados numa ampla variedade de tarefas na área da robótica [12].

2.3.6 PPO: Proximal Policy Optimization

O *Proximal Policy Optimization* (PPO) é um algoritmo *on-policy* baseado no método de gradiente de política. Este algoritmo alterna entre a amostragem de dados através da interação com o ambiente e a otimização de uma função objetiva "substituta" utilizando o gradiente estocástico ascendente.

Tanto o PPO quanto TRPO se preocupam em otimizar o aprendizado sem ocasionar um acidental colapso no sistema. O TRPO tenta resolver esse problema por um método complexo de segunda ordem, enquanto o PPO utiliza métodos de primeira ordem que incorpora alguns artifícios para controlar a atualização da política.

Como resultado o PPO possui alguns dos benefícios do TRPO, entretanto possui uma implementação mais simples e, em geral, melhor complexidade de amostra.

Schulman et al. 2017 publicou um artigo onde apresenta e testa o algoritmo PPO. Neste artigo, os autores apresentam bons resultados do algoritmo numa ampla variedade de tarefas incluindo locomoção robótica simulada e jogos Atari [13].

2.4 CMA-ES: Evolution Strategy with Covariance Matrix Adaptation

O *Evolution Strategy with Covariance Matrix Adaptation* (CMA-ES) é um método robusto de otimização livre de gradientes. Este método é derivado dos conceitos de auto adaptação da estratégia evolutiva e do CMA (*Covariance*

Matrix Adaptation) que adapta a matriz de covariância de uma distribuição normal de pesquisa multivariada [14].

Como está implícito no seu nome, este algoritmo pertence à classe de algoritmo de otimização inspirado na seleção natural. A seleção natural acredita que características benéficas para a sobrevivência do indivíduo é transmitida para a próxima geração. A evolução acontece pelo processo de seleção gradativo e a população cresce mais adaptada ao ambiente.

No aprendizado por reforço, o CMA-ES é utilizado na busca dos parâmetros ótimos da função aproximadora da política ideal. Neste trabalho, esse método foi utilizado para otimizar os pesos da rede neural que determina ação a, a partir do estado recebido. A forma procedimental do CMA-ES é mostrada no pseudocódigo 5.

Algoritmo 5: CMA-ES [15]

início
- Inicialize m, σ, λ, μ
- Inicialize C = I, $\mathrm{p}_c = 0, p_\sigma = 0$
- **repita**
  - Amostre: $\theta_i = m + \sigma y_i$, onde $y_i \sim N(0, C), i = 1, ..., \lambda$
  - Avalie: $\mathrm{f}(\theta_i), i = 1, ..., \lambda$
  - $\mathrm{m} \leftarrow m + \sigma\bar{y}$, onde $\bar{y} = \sum_1^\mu w_i y_i : \lambda$
  - $\mathrm{p}_\sigma \leftarrow (1 - c_\sigma)p_\sigma + \sqrt{c_\sigma(2 - c_\sigma)}C^{-0.5}\bar{y}$
  - $\sigma \leftarrow \sigma \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{||p_\rho||}{E||N(0,I)||}\right) - 1\right)$
  - $\mathrm{p}_c \leftarrow (1 - c_c)p_c + \sqrt{c_\sigma(2 - c_c)\mu_w}\bar{y}$
  - $\mathrm{C} \leftarrow (1 - C_1 - C_\mu)C + c_1 p_c p_c^T + c_\mu \sum_1^\mu w_i y_i : \lambda y_i^T : \lambda$
- **até** *convergir*

fim

Onde λ é o tamanho da população, μ são os melhores candidatos, m é a média, σ é o desvio padrão e C é a matriz de covariância.

Uma das primeiras utilizações de CMA-ES na área de aprendizado por reforço foi proposta por Christian Igel e Verena H. Meisner em 2009 [16]. Neste trabalho, os autores relataram bons resultados em ambiente total e parcialmente observados. Em trabalhos posteriores, os autores comparam o CMA-ES com *Natural Policy Gradient Ascent* na resolução de MDPs.

3
Aprendizado por Reforço em Processos Químicos

Neste capítulo serão apresentados alguns estudos relacionados à aplicação de aprendizado por reforço em processo químicos. Este relato foi organizado de forma cronológica, destacando os processos químicos e os algoritmos de APR utilizados. Posteriormente serão descritos os dois estudos de casos empregados nesta dissertação.

3.1
Revisão Bibliográfica

Apesar do advento da indústria 4.0, a maioria dos estudos sobre aplicações de aprendizado por reforço em processos químicos foram implementados em ambientes simulados. A resistência da implementação na indústria química deve-se principalmente a dois fatores: 1) a natureza de caixa preta da política de controle; 2) a conectividade da APR ao sistema de controle distribuído industrial (DCS) [17].

Primeiramente, a natureza de caixa preta do APR pode apresentar riscos operacionais devido à dificuldade de predição do comportamento do agente. No MPC *(Model Predictive Control)*, as ações de controle podem ser compreendidas através da análise do modelo do sistema. No APR, é quase impossível compreender o motivo ou como o agente aprendeu a política de controle. Pode-se avaliar o comportamento pontual do algoritmo de APR através da relação entre condições de operação e a resposta do agente. Entretanto, a maioria das funções aproximadoras utilizadas nas políticas de controle são altamente não lineares, principalmente as redes neurais. Isso implica que as condições operacionais próximas aos pontos testados ainda podem resultar em comportamentos muito diferentes. Este fato, coloca em xeque as provas de convergência e de estabilidade dos controladores baseados em APR, os quais são fundamentais para garantir a segurança do processo [17].

Segundo ponto, a comunicação confiável entre o algoritmo de APR e o sistema de controle distribuído industrial pode representar desafios. Segundo Nian et al. 2020, não existe nenhum *software* APR industrialmente aceito até 2020. Além disso, a falta de suporte a rede neurais profundas nos DCS moderno impossibilitam a incorporação de algoritmos de APRP. Uma solução

apontada pelos autores é construção do agente de APR em um *software* externo e aproveitar *Open Platform Communication* (OPC) para se comunicar com os sistemas de controle modernos[17].

As primeiras abordagens das aplicações de aprendizado por reforço em processos químicos foram os neurocontroladores. J.C.Hoskins et al. em 1992, controlou um reator CSTR genérico com controlador neural de Anderson [18]. Alex et al. em 2001, obteve excelentes resultados no controle completo do *Tennessee Eastam Process* utilizando o algoritmo SANE (*Symbiotic Adaptive Neuro-Evolution*) [19]. Este algoritmo não pode ser classificado como um algoritmo de aprendizado por reforço devido a sua estrutura baseada na preservação da melhor política, além de não seguir as propriedades de Markov.

Algoritmos baseados na *Approximate Dynamic Programming* (ADP) foram utilizados para controlar reatores e outros equipamentos. Por ser um algoritmo de estados discretos, nessas aplicações foram necessários a utilização de algoritmos como K-NN, redes neurais, RBF, para aproximar o ambiente contínuo. Jong MinLee et al em 2006, desenvolveu controladores para um modelo simplificado do reator Van de Vusse e um reator de polimerização de MMA [20]. Thidarat Tosukhowong et al em 2009, implementou um algoritmo baseado em ADP para controlar uma planta constituída de um reator e de uma coluna de destilação com reciclo [21]. Neste trabalho, realizou-se uma seleção de variáveis devido à alta dimensionalidade do modelo. Em seus trabalhos, Sudhakar Munusamy et al. e Christian D et al. desenvolveram controladores baseados em ADP para um reator PFR (*Plug Flow Reactor*) [22].

Na área de aprendizado por reforço profundo, alguns trabalhos utilizaram o DQL com algumas modificações para o controle de processos químicos. Zhuang Shao et al. em 2020, implementou um DQL híbrido com janelamento para um sistema de dessulfurização de gases de combustão úmida (WFGD) [23]. Soonho Hwangbo et al. utilizou um MC-DQL para um sistema de controle de separação a jusante em processos biofarmacêuticos. Nesta alteração do DQL, alterou-se o aprendizado por diferenças temporais (TD) para um aprendizado baseado pelo algoritmo de Monte-Carlos (MC) [24]. Dong-Hoon Oh et al. utilizou o algoritmo *Advantage Actor-Critic* (A2C) para estimar as condições operacionais ideais do processo de hidrocraqueamento. HaeunYoo et al., em seu trabalho propõe a alteração do aprendizado por diferenças temporais (TD) do DDPG para o aprendizado por Monte-Carlos (MC) [25].

Na Tabela 3.1 encontra-se um resumo de alguns trabalhos de aprendizado por reforço em processos químicos, destacando-se os algoritmos e a dimensionalidade do ambiente. Todos os trabalhos desta tabela foram estudados por simulações. Adotou-se a letra “C” para indicar que o estado é contínuo e “D”

para estados discretos.

Tabela 3.1: Resumo algumas de aplicação de APR em processo.

Trabalho	Modelo	Algoritmo	Nº obs	Tipo	Nº ações
[18]	CSTR	N. Controlador	8	C	2
[19]	Tennessee	SANE	63	C	12
[26]	Reator	ADP	5	C	1
[20]	RVV/Polimerização	ADP	2 e 8	D	1 e 2
[27]	Planta	ADP	13	D	1
[28]	PFR	ADP e QL	1	D	1
[29]	PFR	ADP e QL	1	D	1
[30]	Bomba	ADP	1	D	1
[22]	Processo	A2C		D	4
[23]	WFGD	DQL	≥ 100	C	4
[24]	Processo	MC-DQL	7	C	5
[31]	HCR	A2C	3	C	4
[25]	Polimerização	AC	11	C	2

Onde a coluna "Modelo" indica o tipo de processo utilizado no trabalho, "Nº obs" indica a quantidade de estados observados, "Nº ações" indica a quantidade de ações (válvulas, vazões, etc.).

A maioria dos estudos de aprendizado por reforço em processo químicos limitam-se a modelos complexos com a dimensionalidade reduzida, como é casos dos reatores PFR e das reações de polimerização. Em caso de alta dimensionalidade, normalmente simplifica-se o ambiente ou a atuação. Dificilmente, encontram-se estudos de alta dimensionalidade e alta complexidade simultaneamente.

3.2
Estudos de Caso

Nesta seção serão relatados os dois estudos de casos utilizados neste presente trabalho. Primeiramente, será apresentado o reator CSTR (*Continuous Stirred-Tank Reactor*) com a cinética de Van de Vusse. Este modelo é bastante conhecido e estudado na literatura devido às características de não linearidades. Por último, será comentado o *Tennessee Eastman Process*. Este processo é baseado em uma subunidade de planta química constituída de cinco equipamentos e sua respectiva malha de controle.

3.2.1
Reator CSTR de Van de Vusse

Na cinética de Van de Vusse, ciclopentenol(B) é produzido a partir do ciclopentadieno, com formação também dos subprodutos ciclopentanadiol(C) e diciclopentadieno(D), conforme as reações da Figura 3.1 [32].

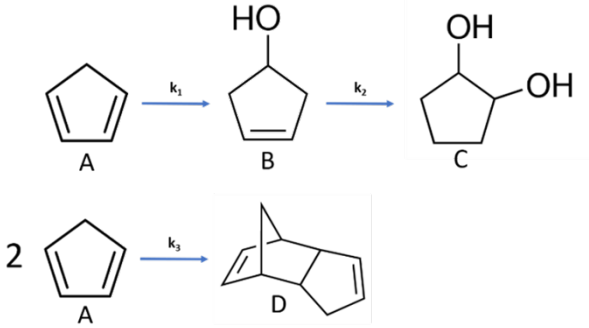

Figura 3.1: Reação química de Van de Vusse

A relação entre as constantes cinéticas k_1, k_2, k_3 e a temperatura são regulamentadas pela equação de Arrhenius (Equação 3-1).

$$ki(T) = ki_0.\exp\left(\frac{-EA_i}{R.T}\right) \tag{3-1}$$

Onde ki_0 é a constante cinética na temperatura de referência, EA_i é a energia de ativação, T é a temperatura e R é a constante universal dos gases. Conforme descrito em mais detalhes por Engell Klatt em 1993 [33], a reação ocorre em um reator CSTR encamisado, devido à natureza exotérmica da reação (Figura 3.2).

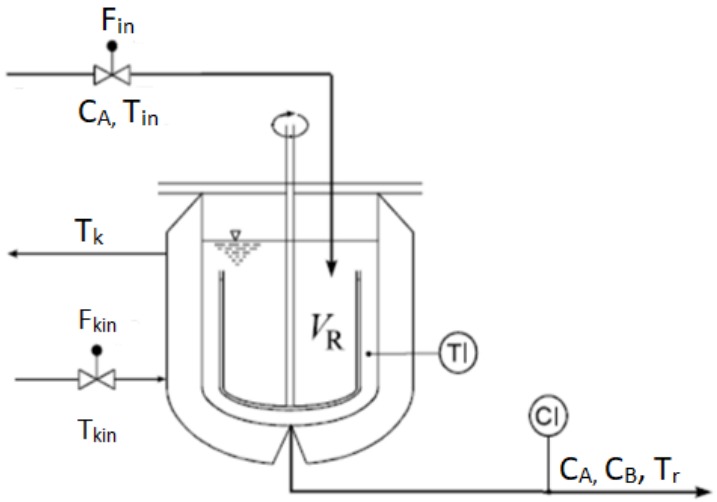

Figura 3.2: Reator CSTR de Van de Vusse [35]

Onde C_A é a concentração do composto A (ciclopentadieno), C_B é a concentração do composto B (ciclopentenol), F_{in} é a vazão de alimentação do reator, T_{in} é a temperatura da corrente de alimentação do reator, C_{Ain} é a concentração de A na alimentação, F_{kin} é a vazão de alimentação da camisa térmica, T_{kin} é a temperatura da corrente de alimentação da camisa térmica, V_r é o volume do reator, T_r é a temperatura do reator e T_k é a temperatura da camisa térmica.

A dinâmica do sistema é descrita pelas equações diferenciais, resultantes dos balanços de massa e energia do reator e da camisa de resfriamento (Equação 3-2). Para simular o sistema ao longo do tempo essas equações são integradas numericamente.

$$\begin{aligned}
\frac{dC_A}{dt} &= \frac{F_{in}}{V_r}[C_{Ain} - C_A] - k_1(T)C_A - k_3(T)C_A^2 \\
\frac{dC_B}{dt} &= \frac{F_{in}}{V_r}[C_{Bin} - C_B] + k_1(T)C_A - k_2(T)C_B \\
\frac{dT_r}{dt} &= \frac{F_{in}}{V_r}[T_{in} - T_r] + \frac{k_w.A_r}{\rho C_p V_r}[T_k - T_r] + \frac{1}{\rho C_p}[k_1 C_A \Delta H_1 + k_1 C_B \Delta H_2 + k_3 C_A^2 \Delta H_3] \\
\frac{dT_k}{dt} &= \frac{F_{kin}}{V_k}.[T_{kin} - T_k] - \frac{k_w.A_r}{m_k.C_{p_k}}.[T_k - T_r]
\end{aligned} \tag{3-2}$$

Nas Tabelas 3.2 e 3.3 encontram-se os parâmetros cinéticos da reação e propriedades do reator e camisa fornecidos por Engell Klatt (1993) [33]. Esses dados serviram como base em diversos outros estudos sobre este reator [34, 35]. Para simplificação do modelo, foram considerados a densidade, a capacidade

térmica e o nível constantes ao longo do reator.

Tabela 3.2: Propriedades físico-químicas e dimensões do reator.

Parâmetros	Símbolos	Valores	Unidades
Densidade da solução	ρ	0,9342	$kg.L^{-1}$
Capacidade térmica da solução	C_p	3,01	$kJ.kg^{-1}.K^{-1}$
Condutividade térmica	K_w	4032	$kJ.m^{-1}.h^{-1}K^{-1}$
Área de trocas térmica do reator	A_r	0,215	m^2
Volume do reator	V_r	10	L
Massa da camisa	m_k	5	kg
Capacidade térmica da camisa	C_{p_k}	2,0	$kJ.kg^{-1}.K^{-1}$

Tabela 3.3: Parâmetros cinéticos da reação.

Reação	k_{i0}	EA_i/R	ΔH_{ri}
A $\rightarrow$ B	1,287 x 10^{12} h^{-1}	-9758,3 K	4,2 $kJ.mol^{-1}$
B $\rightarrow$ C	1,287 x 10^{12} h^{-1}	-9758,3 K	-11 $kJ.mol^{-1}$
2A $\rightarrow$ D	9,043 x 10^{12} $L.mol^{-1}h^{-1}$	-8560 K	-41,85 $kJ.mol^{-1}$

Além da não linearidade, este processo possui uma inversão de ganho. Esta característica se revela quando um mesmo valor de estado estacionário para a variável controlada (C_B) é obtido para diferentes valores de variável manipulada (F_{in}). Este comportamento torna o controle do processo ainda mais desafiador, uma vez que perturbações de origens diversas podem fazer com que o processo perca estabilidade, podendo ser levado a um estado estacionário indesejado ou oscilar entre dois estados estacionários.

O comportamento no estado estacionário da concentração de ciclopentenol (C_B) em função vazão de alimentação (F_{in}) é apresenta na Figura 3.3. A concentração de ciclopentenol (C_B) em função da vazão da alimentação e da temperatura do meio reacional encontra-se na Figura 3.3.a. Os perfis longitudinais e transversais para determinadas temperaturas e vazões encontram-se nas Figuras 3.3.b e 3.3.c, respectivamente.

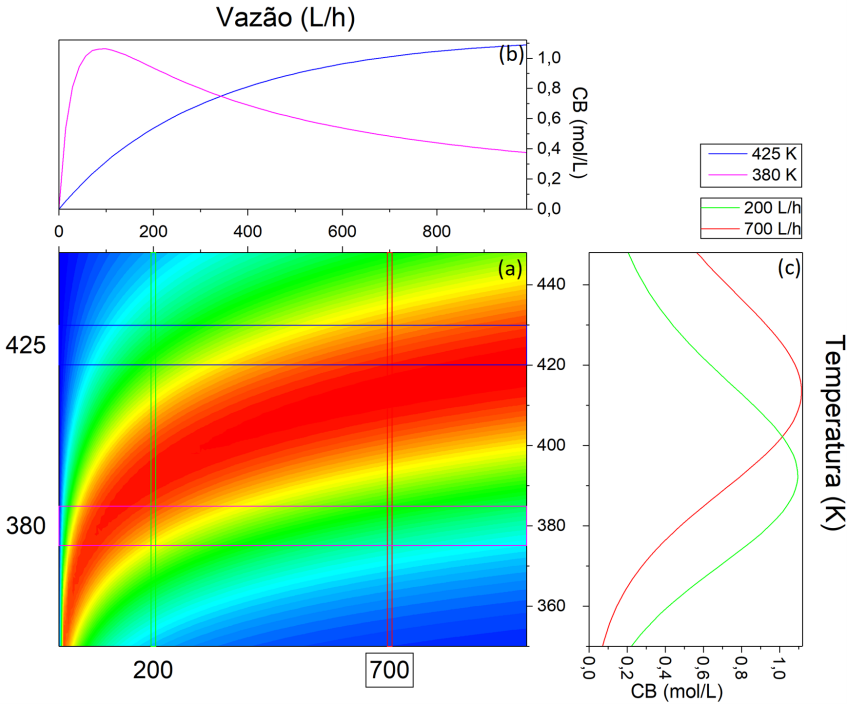

Figura 3.3: Concentração de B em função da vazão e temperatura [36]

O perfil não linear é mais facilmente evidenciado no corte de 380 K, onde a curva é crescente até a vazão de 100 L/h e para vazões maiores torna-se decrescente (Figura 3.3.b). Este aspecto é problemático para os controladores, pois nas regiões de vazões baixas, o C_B é crescente com a vazão e na região de vazões altas estas grandezas são decrescentes (Figura 3.3.b).

3.2.2 Tennessee Eastman Process

O *Tennessee Eastman Process* (TE) foi proposto por Downs e Vogel como um problema de avaliação de metodologias avançadas de controle de processo a partir de uma perspectiva geral de uma planta [37]. O modelo dinâmico do processo (baseado em um processo industrial real) integra a operação de cinco unidades operacionais: um reator exotérmico de duas fases, um condensador parcial, um compressor, um tanque *flash* e um *stripper*.

O processo consiste na produção de dois produtos a partir de quatro reagentes. Também estão presentes um inerte e um subproduto perfazendo um total de oito componentes: A, B, C, D, E, F, G e H. As reações químicas encontram-se na Equação 3-3.

$$\begin{aligned} A_{(g)} + C_{(g)} + D_{(g)} &\rightarrow G_{(liq)} \\ A_{(g)} + C_{(g)} + E_{(g)} &\rightarrow H_{(liq)} \\ A_{(g)} + E_{(g)} &\rightarrow F_{(liq)} \\ 3D_{(g)} &\rightarrow 2F_{(liq)} \end{aligned} \tag{3-3}$$

As reações presentes na Equação 3-3 são irreversíveis e exotérmicas. As taxas dessas reações são governadas pela equação de Arrhenius, em outras palavras, elas são funções da temperatura (Equação 3-1). A reação de produção do composto G tem uma energia de ativação mais alta, resultando em uma sensibilidade maior à temperatura [37].

Na Figura 3.4 encontra-se o diagrama do fluxograma de processo do TE.

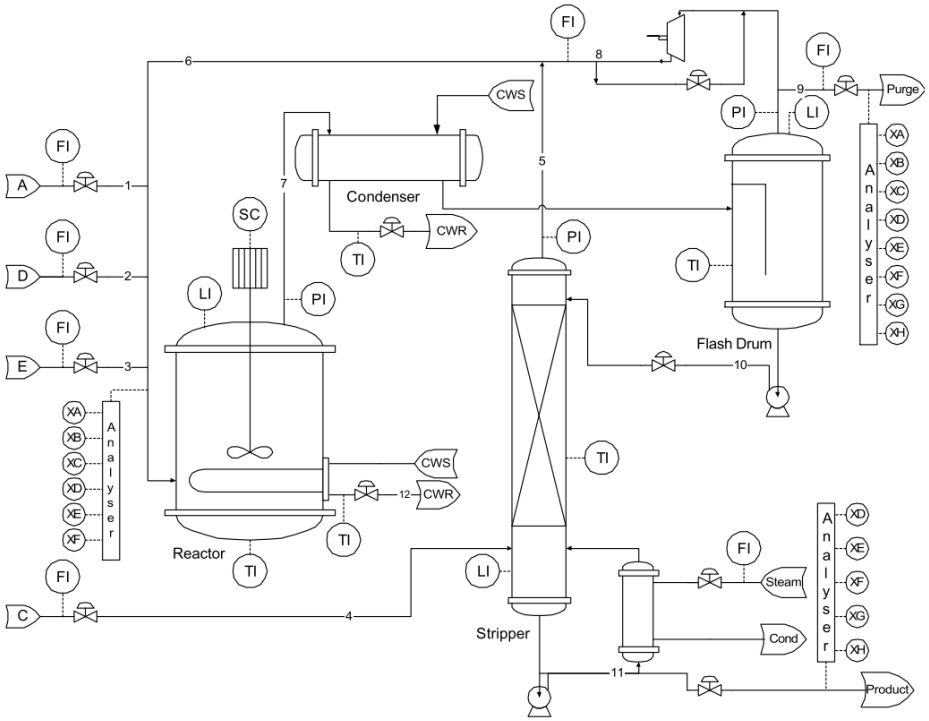

Figura 3.4: Diagrama do Fluxograma de Processo do TE [37]

Os reagentes gasosos A, B, C, D e E são misturados na corrente 6. Esta corrente, serve de alimentação do reator CSTR bifásico. As reações químicas descritas na Equação 3-3 ocorrem na fase gasosa do reator. Os produtos G, H e o subproduto F se acumulam no reator, sendo F inerte e indesejável. O nível do reator deve ser controlado de tal modo igualando a taxa química com a taxa de remoção do vapor. O produto é removido por meio do fluxo de vapor que sai para o condensador. A taxa de remoção de calor é ditada pela velocidade

de agitação das hélices e pelo fluxo do fluido de resfriamento nas serpentinas do reator (Figura 3.4).

A corrente gasosa do reator alimenta um condensador parcial, posteriormente esta corrente segue para um tanque *flash* (corrente 7). Esse tanque serve para separar a fase líquida e a fase gasosa. A fração líquida, rica em produtos, alimenta o *stripper* enquanto a fração gasosa, rica em reagente, retorna ao reator através de um compressor centrífugo, como uma corrente de reciclagem. Uma purga é necessária para evitar o acúmulo do composto B inerte (presente nas alimentações de A e C). A válvula de reciclagem do compressor permite o ajuste da taxa de alimentação líquida ao reator (Figura 3.4).

A corrente liquida 10, rica em G e H, entra na parte superior do *stripper* enquanto a corrente gasosa 4, rica no composto C, entra na parte inferior. Neste equipamento, a fase gasosa absorve as impurezas e os regentes não reagidos da fase líquida. A fase gasosa entra na corrente de alimentação do reator, enquanto a fase líquida, o produto final, entra na corrente 11 onde terá sua composição analisada (Figura 3.4).

As doze variáveis manipuladas são as quatro taxas de alimentação, a taxa de purga, a taxa de agitação, a taxa de vapor, a taxa do refrigerante do condensador, a taxa do refrigerante do reator, a reciclagem do compressor, a taxa de descarga do tanque *flash* e a taxa de produção do *stripper* (Tabela 3.4).

Tabela 3.4: Variáveis manipuladas do Tennessee Eastman Process.

ID	Unidades	Variáveis
XMV(1)	%	Vazão de Alimentação de D(corrente 2)
XMV(2)	%	Vazão de Alimentação de E (corrente 3)
XMV(3)	%	Vazão de Alimentação de A (corrente 1)
XMV(4)	%	Vazão de Alimentação de A e C (corrente 4)
XMV(5)	%	Válvula de Reciclo do Compressor
XMV(6)	%	Válvula da Purga (corrente 9)
XMV(7)	%	Vazão do Produto liq. do Separador(corrente 10)
XMV(8)	%	Vazão do Produto liq. do Stripper (corrente 11)
XMV(9)	%	Válvula do Vapor do Stripper
XMV(10)	%	Vazão do Fluido de resf. do Reator
XMV(11)	%	Vazão do fluido de resf. do Condensador
XMV(12)	%	Velocidade do Agitador

As 41 variáveis de processo monitoradas incluem indicadores de nível, pressão, temperatura, vazão e composição que podem ser consultadas nas Ta-

belas 3.5 e 3.6. Outro ponto que acrescenta maior grau de fidelidade e de complexidade ao TE são as diferentes taxas de amostragem das variáveis medidas. Por exemplo, o nível do reator possui uma amostragem de 3 segundos, enquanto a concentração do composto G na corrente 11 possui uma amostragem de 15 minutos. Esta diferença corresponde a casos reais, afinal é mais prático, rápido e barato medir um nível do reator do que a concentração de composto químico em uma corrente.

Essa divergência da amostragem, afeta a propriedade de Markov do modelo do *Tennessee Eastman Process*. Afinal, variáveis dessincronizadas e atrasadas podem destruir as relações de casualidades entre os estados e ações. Isso se torna mais crítico ao ter as variáveis fundamentais para o controle do processo como as concentrações dos compostos químicos em tempos tão diferentes dos demais.

Tabela 3.5: Variáveis Monitoradas do TE amostradas a cada 3 s.

ID	Variáveis	Corrente
XMEAS(1)	Alimentação de A	1
XMEAS(2)	Alimentação de D	2
XMEAS(3)	Alimentação de E	3
XMEAS(4)	Alimentação de A e C	4
XMEAS(5)	Vazão de Reciclo	8
XMEAS(6)	Taxa de Ali. do Reator	6
XMEAS(7)	Pressão do Reator	
XMEAS(8)	Nível do Reator	
XMEAS(9)	Temperatura do reator	
XMEAS(10)	Taxa de Purga	9
XMEAS(11)	Temperatura do Produto do Separador	
XMEAS(12)	Nível do Separador	
XMEAS(13)	Pressão do Separador	
XMEAS(14)	Vazão de fundo do separador	10
XMEAS(15)	Nível do Stripper	
XMEAS(16)	Pressão do Stripper	
XMEAS(17)	Vazão do Stripper Underflow	11
XMEAS(18)	Temperatura de Stripper	
XMEAS(19)	Vazão do Stripper	
XMEAS(20)	Trabalho do Compressor	
XMEAS(21)	Temp. da Camisa Resfr. do Reator	
XMEAS(22)	Temp. da Camisa Resfr. do Separador	3

Tabela 3.6: Variáveis Monitoradas do TE.

ID	Variáveis	Amostragem
XMEAS(23)	Corrente 6 - Componente A	6 min
XMEAS(24)	Corrente 6 - Componente B	6 min
XMEAS(25)	Corrente 6 - Componente C	6 min
XMEAS(26)	Corrente 6 - Componente D	6 min
XMEAS(27)	Corrente 6 - Componente E	6 min
XMEAS(28)	Corrente 6 - Componente F	6 min
XMEAS(29)	Corrente 9 - Componente A	6 min
XMEAS(30)	Corrente 9 - Componente B	6 min
XMEAS(31)	Corrente 9 - Componente C	6 min
XMEAS(32)	Corrente 9 - Componente D	6 min
XMEAS(33)	Corrente 9 - Componente E	6 min
XMEAS(34)	Corrente 9 - Componente F	6 min
XMEAS(35)	Corrente 9 - Componente G	6 min
XMEAS(36)	Corrente 9 - Componente H	6 min
XMEAS(37)	Corrente 11 - Componente D	15 min
XMEAS(38)	Corrente 11 - Componente E	15 min
XMEAS(39)	Corrente 11 - Componente F	15 min
XMEAS(40)	Corrente 11 - Componente G	15 min
XMEAS(41)	Corrente 11 - Componente H	15 min

Downs e Vogel, modelaram o processo como um conjunto de sub-rotinas escritas em FORTRAN, que descrevem as relações não lineares nas operações unitárias e os balanços de material e energia. Também foram implementados as condições de desligamento da planta, caso alguma variável de processo ultrapasse algumas faixas de operações pré estabelecida.

N. Lawrence Ricker em 2015 implementou uma estimativa do custo operacional da planta em seu modelo no MATLAB/Simulink [1]. Esta estimativa considera o consumo das variáveis XMEAS(10, 19, 20, 29, 31, 32, 33, 34, 35, 36, 37, 38, 39) e XMV(8) (Figura 3.5).

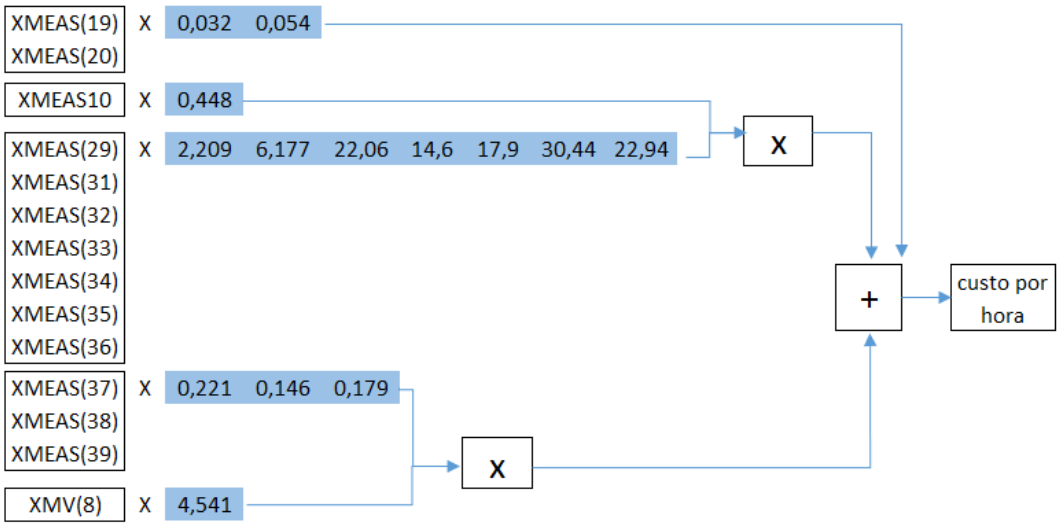

Figura 3.5: Custo Operacional do TE

[1] https://depts.washington.edu/control/LARRY/TE/download.html

4
Metodologia

Neste capítulo serão descritos as metodologias adotadas nos dois estudos de caso. Primeiramente, serão apresentados as metodologias das simulações do reator CSTR de Van de Vusse. Por último, a do *Tennessee Eastman Process*.

4.1
Reator de Van de Vusse

O processo foi estudado por simulação, integrando-se numericamente as equações diferenciais não lineares (Equação 3-2) que representam o reator CSTR com a cinética de Van de Vusse (Figura 3.2). Esse reator foi implementado no ambiente *Generic Reinforcement Learning Library* [1] (GRL) desenvolvido por Wouter Caarls.

O controlador baseado em aprendizado por reforço desenvolvidos para este processo são multivariáveis contendo 5 variáveis observadas e 2 variáveis de atuação. As concentrações dos compostos A e B, o *setpoint* da concentração de B e a temperatura do reator e da camisa térmica foram consideradas como variáveis observadas. As variáveis manipuladas foram as vazões de alimentação do reator e da camisa térmica. A função de recompensa está descrita na Equação 4-1.

$$r(t) = P_{F_{in}}.\frac{F_{in}(t)}{700} - (1 - P_{F_{in}}).|Sp_{C_B}(t) - C_B(t)| \tag{4-1}$$

Onde $r(t)$ é a recompensa por transição, $Sp_{C_B}(t)$ é o *setpoint* da concentração do composto B, $C_B(t)$ é a concentração de B na corrente de saída, $F_{in}(t)$ é vazão a alimentação do reator, $P_{F_{in}}$ é a influência da vazão sobre a recompensa. A inclusão do $P_{F_{in}}$ permite ao agente ponderar entre a maximização da produção e a qualidade do produto. O desejado é o obter a maior quantidade do produto na concentração de B especificada.

Este sistema por apresentar uma quantidade relativamente pequena de estado e ações é possível avaliar o desempenho dos algoritmos de APR clássicos como *Q-Learning* de ação discreta e *Actor Critic* TD de ação contínua. Apesar

[1] https://github.com/wcaarls/grl

dessa quantidade, uma discretização muito fina pode inviabilizar o estudo com estes algoritmos. Com este intuito, foram acrescentados à lista de algoritmos analisados, o DDPG, o SAC e o TD3 de ações e estados contínuos e o DQL de ações discreta e estado contínuo.

As simulações foram elaboradas para primeiramente avaliar a capacidade dos controladores de estabilizarem o reator em uma mudança de *setpoint* estável, isto é, mudar as condições operacionais do processo para um valor que se mantenha constante em regime estacionário. Por último, realizou-se o teste de estabilidade entorno do ponto de inversão de ganho, onde se encontra a verdadeira dificuldade de controle neste estudo de caso.

Foram realizados análises e testes paramétricos para os ajustes dos hiperparâmetros de cada algoritmo. A faixa de busca de alguns hiperparâmetros foram: a taxa de aprendizado (α) de 0,01 a 0,2, o fator de desconto (γ) de 0,8 a 0,999, o período de atuação do algoritmo (a_s) de 3 a 60 segundos e a fração de atualização das redes alvos (τ) de 0,01 a 0,001. Dentro deste espaço de busca, utilizaram-se os melhores hiperparâmetros para experimentos posteriores. Para o cálculo do fator γ, utilizou-se a Equação 4-2. Esta função permite traduzir o fator de desconto para termos de tempo e redução percentual da recompensa.

$$\gamma = [0,5^{h^{-1}}]^{a_s} \tag{4-2}$$

Onde γ é fator de desconto utilizado no cálculo da função valor, a_s é período de atuação do algoritmo, h é o horizonte de tempo que a recompensa é reduzida pela metade.

Apesar da recompensa acumulada ser uma métrica de desempenho, ela não é adequada para comparação de controladores APR com ambiente e com funções de recompensas diferentes. Neste âmbito, a utilização do *Integral of Time multiplied by Absolute of the Error* (ITAE) permite a comparação entre diferentes controladores APR (Equação 4-3).

$$ITAE(t) = \int_0^t t.|Sp_{C_B}(t) - C_B(t)|dt \tag{4-3}$$

4.2 Tennessee Eastman Process

O *Tennessee Eastman Process* (TE) foi estudado através de simulações nos ambientes de aprendizado por reforço GRL e *Spinning Up* (SPIN) [38]. Este estudo foi realizado através do acoplamento entre as sub-rotinas escritas em FORTAN por Downs e Vogel no ambiente GRL. Para os algoritmos *off-policy* DDPG, SAC e TD3 utilizaram as implementações fornecidas na biblioteca GRL e para os algoritmos *on-policy* TRPO e PPO utilizaram as da biblioteca do *Spinning Up*. Recorreu-se a biblioteca *OpenAi GYM* para converter e criar um ambiente de APR compatível com *Spinning Up* (Figura 4.1).

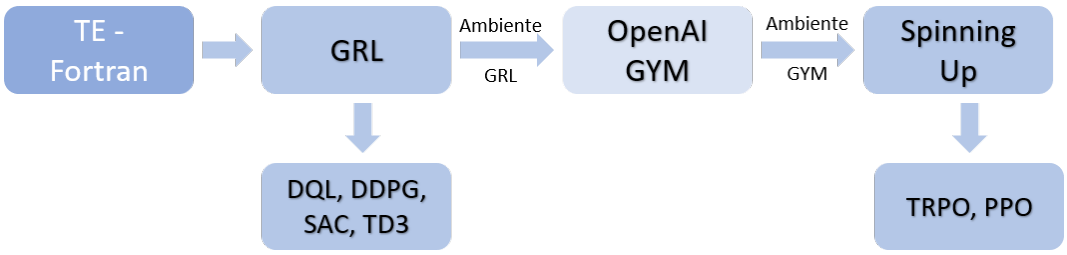

Figura 4.1: Fluxo de trabalho do GRL e *Spinning Up*

Foram evitadas comparações diretas entre os algoritmos do GRL e do *Spinning Up* devido às diferenças metodológicas de cada ambiente. A principal diferença entre eles está na definição de episódio. Enquanto para o GRL um episódio é igual a uma simulação, para o *Spinning Up* é uma quantidade fixa de transições. Esta divergência se torna relevante quando o modelo possui variadas durações de simulações conforme a atuação dos algoritmos.

As simulações do TE geram 51 estados internos do modelo, além das 41 variáveis monitoradas e 12 variáveis manipuladas descritas nas Tabelas 3.5, 3.6 e 3.4, respectivamente. Esta alta dimensionalidade impacta diretamente no processamento e no armazenamento dos resultados. Dependendo da quantidade de estados observados, do ambiente, de duração da simulação e de outros fatores, pode ocorrer geração de arquivos de saídas CSV na ordem de 30 GB. Nos casos extremos, chegou a ser gerado arquivos CSV de 90 GB. Estes arquivos foram salvos para averiguar o comportamento e a evolução dos controladores baseado em aprendizado por reforço.

O ambiente GRL permite selecionar as variáveis observadas e manipuladas utilizadas pelo agente do APR. Esta seleção permite definir quais equipamento e quais variáveis serão controladas por APR e quais serão controlados por outros algoritmos, por exemplo, um PID.

Para sintonia dos hiperparâmetros de cada algoritmo foram realizadas análises e testes paramétricos controlando apenas a temperatura e nível do reator. Nestas simulações, o restante da planta foi controlado pelos controladores PID implementados nas sub-rotinas do FORTRAN mas gerenciados por GRL. A faixa de busca de alguns hiperparâmetros foram: a taxa de aprendizado (α) de 0,001 a 0,2, o fator de desconto (γ) de 0,7 a 0,9999, a fração de atualização das redes alvos (τ) de 0,01 a 0,00001 e a taxa de aprendizado da rede crítica e ator de 10^{-3} a 10^{-7}. Dentro deste espaço de busca, utilizaram-se os melhores hiperparâmetros.

O *Tennessee Eastman Process* é processo pouco markoviano devido à grande diferença entre o número de variáveis observadas e o número de variáveis de processo [19]. Para tentar contornar esta dificuldade, avaliou-se a utilização do artifício do *window*. Este artifício se resume a uma janela dos históricos dos estados observados como entrada do algoritmo (Figura 4.2). Um dos inconvenientes da utilização do *window* é o aumento do número de estados observados.

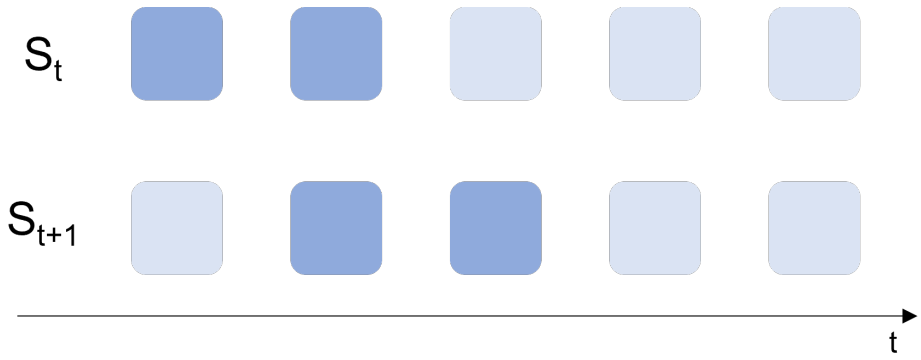

Figura 4.2: Exemplo de entradas com *Window* de tamanho 2

Outra característica peculiar do TE é a "inercia" química e física do processo. Cada grandeza química e física possui diferentes tempos de resposta. Por exemplo, a ação do controle de um nível do tanque é mais rápida e direta do que o controle da temperatura. Com este intuito, avaliou-se *window* com entradas espaçadas, denominadas *stride* (Figura 4.3).

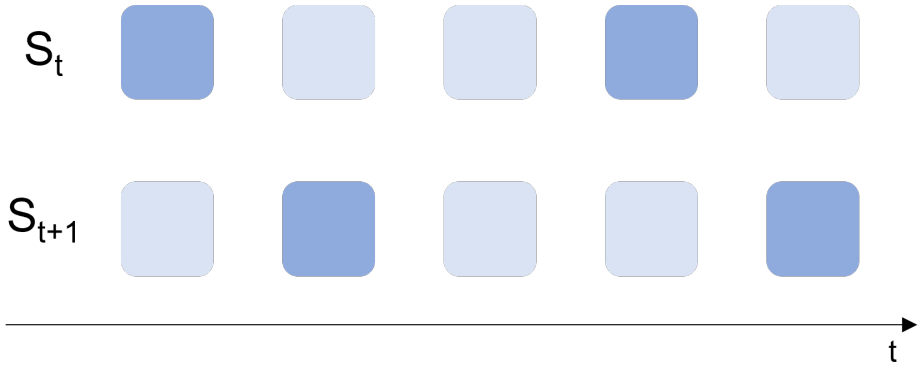

Figura 4.3: Exemplo de entradas com *Window* 2 e *Stride* 2

Estes dois recursos aumentam consideravelmente a quantidade de informação de cada estado. Eles adicionam as noções temporais aos estados sobre as evoluções das variáveis observadas. Por exemplo, o *window* possibilita o cálculo da derivada numérica de cada estado, enquanto o *stride* determina o intervalo sobre a derivada calculada. Em um sistema fracamente markoviano, essas informações podem ser cruciais para o sucesso do algoritmo.

Como recurso para contornar o possível problema do aumento da dimensionalidade ocasionado pelo *window* e o *stride*, realizou-se uma seleção de variáveis. Para esta seleção utilizou-se o método *filter* supervisionado "CfsSubsetEval" visando como saída a concentração do composto G no produto, a temperatura e a pressão do reator e os níveis do *stripper*, do separador e do reator. Utilizando a implementação do *sofware* WEKA 3.6 [39], foram selecionadas 18 variáveis das Tabelas 3.5 e 3.6: XMEAS(2, 4, 5, 6, 7, 8, 9, 12, 13, 14, 15, 16, 17, 32, 34, 37, 40).

As sub-rotinas do *Tennessee Eastman Process* possuem um sistema de desligamento da planta, caso certas variáveis de processo passem de um valor. Essas variáveis são a temperatura e pressão do reator e os níveis do *stripper*, do separador e do reator. Para melhor adequar o TE a um problema de aprendizado por reforço foram criados dois elementos, a penalidade por desligamento e a recompensa por tempo. A penalidade é um valor negativo acrescentado a função de recompensa, caso ocorra o término prematuro do episódio devido ao desligamento.

A recompensa por tempo é uma constante positiva acrescentada à função recompensa a cada transição. As funções de recompensa em problemas de controles de APR, em sua grande maioria possuem sinais negativos. Normalmente, elas são o valor negativo da diferença entre valor desejado (*setpoint*) e valor atual da variável controlada. O objetivo de um agente de APR é maximizar as recompensas acumuladas, em outras palavras, minimizar esta diferença. Como

este modelo possui um sistema de desligamento prematuro, ocasiona episódios mais curtos, consequentemente, estes possuem menor erro acumulado. O agente tende a exceder os limites operacionais ocasionando o desligamento da planta, apesar da penalidade por desligamento. O acréscimo da recompensa por tempo mostrou-se necessário para garantir que a recompensa por transição seja positiva, garantindo assim a preferência do agente pela continuidade do episódio.

Inicialmente avaliou-se o desempenho dos algoritmos de APR controlando apenas o nível e a temperatura do reator CSTR bifásico. As outras malhas foram controladas pelos PIDs incluso nas sub-rotinas originais do *Tennessee*. Para este experimento utilizou-se a Equação 4-4 como função de recompensa e as variáveis observadas foram as XMEAS(3, 7, 8, 20) e as variáveis de manipulação foram as XMV(3, 9). Estas variáveis foram escolhidas de tal modo a substituir os controladores PIDs, responsáveis pelo nível e temperatura do reator.

$$r(t) = \begin{cases} t_{rw} - |Sp_{nível} - \text{XMEAS(8)}| - |Sp_{temp} - \text{XMEAS(9)}| & \text{se } t \leq t_t \\ P & \text{se } t = t_f \lneq t_t \end{cases} \tag{4-4}$$

Onde P é penalidade por desligamento, t_{rw} é a recompensa por tempo, C é o custo de operação, t_t é o tempo do estado terminal da simulação, t_f é o tempo final da simulação, XMEAS(8) é o nível do reator, $Sp_{nível}$ é setpoint do nível, XMEAS(9) é a temperatura do reator e Sp_{temp} é setpoint da temperatura.

Para o controle integral da planta por algoritmo de aprendizado por reforço desenvolveu-se a Equação 4-5 como função de recompensa. Essa função foi elaborada para minimizar o custo de produção, manter o produto na qualidade especificada e manter a planta em funcionamento.

$$r(t) = \begin{cases} t_{rw} - 100.C - 0,1.|Sp_{C_G} - \text{XMEAS(40)}| & \text{se } t \leq t_t \\ P & \text{se } t = t_f \lneq t_t \end{cases} \tag{4-5}$$

Onde C é o custo de operação, XMEAS(40) é a fração molar percentual do composto G na corrente 11 e Sp_{C_G} é setpoint de G na corrente 11.

Devido à instabilidade do aprendizado do estudo de caso 2, adotou-se duas abordagens na apresentação dos resultados. Primeiro, a utilização do artifício da média móvel para observar a tendência do aprendizado e excluir eventuais ruídos e instabilidades. Segundo, denominou-se a maior média móvel da recompensa acumulada durante o aprendizado como recompensa de pico.

5
Simulações

Neste capítulo serão listadas as simulações dos estudos de caso do reator CSTR de Van de Vusse e do *Tennessee Eastman Process*. Essas simulações foram organizadas a partir de uma progressão crescente do nível de dificuldade e explorando as peculiaridades de cada estudo de caso.

5.1
Reator de Van de Vusse

As simulações envolvendo o estudo de caso do reator CSTR com a cinética de Van de Vusse foram elaboradas para avaliar a estabilidade e a robustez dos controladores baseados em aprendizado por reforço. O primeiro conjunto de simulações é constituído na resposta do controlador a mudança de *setpoint* do composto B (Sp_{C_B}). Para os episódios de treinamento, alteraram-se aleatoriamente o *setpoint* de C_B em um determinado tempo e as condições iniciais de operação do reator. Nos episódios de teste, manteve-se as mesmas condições iniciais de operação e alterou-se Sp_{C_B} de 0,9 para 1,1 mol/L no instante de 16 minutos, conforme a Equação 5-1.

$$Sp_{C_B}(t) = \begin{cases} 0,9 & \text{se } t \leq 16 \text{ min} \\ 1,1 & \text{se } t > 16 \text{ min} \end{cases} \tag{5-1}$$

Nas Tabelas 5.1 e 5.2 encontram-se as simulações referentes a este experimento para os algoritmos de aprendizado por reforço (APR) e aprendizado por reforço profundo (APRP), respectivamente. Cada simulação contém 1000 episódios. As simulações de APRP foram realizadas com γ de 0,994, τ de 0,001 e redes neurais com duas camadas *"full connect"* contendo 400 e 300 neurônios, respectivamente.

Tabela 5.1: Simulações da mudança de *setpont* para APR.

ID	Alg.	Epi.	Rep.	P_{Fin}	γ	$a_s(s)$	h(min)
1	AC	1000	20	0,0	0,994	30	60
2	AC	1000	20	0,1	0,994	30	60
3	AC	1000	20	0,3	0,994	30	60
4	AC	1000	20	0,5	0,994	30	60
5	QL	1000	20	0,0	0,994	30	60
6	QL	1000	20	0,1	0,994	30	60
7	QL	1000	20	0,3	0,994	30	60
8	QL	1000	20	0,5	0,994	30	60

Tabela 5.2: Simulações da mudança de *setpoint* para APRP.

ID	Alg.	Rep	P_{Fin}	T. Apr.	F. Ativação
1	DDPG	5	0,0	0,001 e 0,0001	relu-relu-tanh
2	DDPG	5	0,1	0,001 e 0,0001	relu-relu-tanh
3	DDPG	5	0,3	0,001 e 0,0001	relu-relu-tanh
4	DDPG	5	0,5	0,001 e 0,0001	relu-relu-tanh
5	DQL	5	0,0	0,001	relu-relu-linear
6	DQL	5	0,1	0,001	relu-relu-linear
7	DQL	5	0,3	0,001	relu-relu-linear
8	DQL	5	0,5	0,001	relu-relu-linear
5	SAC	5	0,0	0,001 e 0,0001	relu-relu-tanh
6	SAC	5	0,1	0,001 e 0,0001	relu-relu-tanh
7	SAC	5	0,3	0,001 e 0,0001	relu-relu-tanh
8	SAC	5	0,5	0,001 e 0,0001	relu-relu-tanh
5	TD3	5	0,0	0,001 e 0,0001	relu-relu-tanh
6	TD3	5	0,1	0,001 e 0,0001	relu-relu-tanh
7	TD3	5	0,3	0,001 e 0,0001	relu-relu-tanh
8	TD3	5	0,5	0,001 e 0,0001	relu-relu-tanh

Onde P_{Fin} é o peso da influência da vazão na função de recompensa (Equação 4-1), γ é o fator de desconto, a_s é tempo de atuação do controlador, h é o tempo necessário para influência da recompensa ser reduzida à metade (Equação 4-2), "T. Apr" é a taxa de aprendizado para o algoritmo de treinamento ADAM das redes ator e crítica, "Rep" é a quantidade de replicadas por experimento,"F. Ativação" é função de ativação dos neurônios de cada camada. Os valores de P_{Fin} foram escolhidos para avaliar a influência crescente

da vazão na função de recompensa de modo a equilibrar a maximização da produção e o menor erro operacional.

O segundo conjunto de simulações é constituído no teste do ponto de inversão do ganho. Neste experimento, ocorre a alteração do *setpoint* para a região entorno do ponto de inversão. Em 12 minutos e meio de simulação, eleva-se Sp_{C_B} de 1,0 para 1,2 mol/L. Em 25 minutos de simulação, diminui-se o Sp_{C_B} para o valor estável de 0,9 mol/L (Equação 5-2).

$$Sp_{C_B}(t) = \begin{cases} 1,0 & \text{se } t < 12{,}5 \text{ min} \\ 1,2 & \text{se } 12{,}5 \leq t \leq 25 \text{ min} \\ 0,9 & \text{se } t > 25 \text{ min} \end{cases} \tag{5-2}$$

Conforme se observa na Figura 3.3, o $Sp_{C_B} = 1{,}2$ mol/L não pode ser alcançado em regime estacionário. Portanto, o sistema ficará alternando entre um ganho positivo e negativo durante o período entre 12,5 e 25 minutos da simulação. Esse cenário é desafiador para qualquer controlador devido essas mudanças de ganho o que pode ocasionar um total descontrole do processo. Na Tabela 5.3 encontram-se as simulações referentes ao teste do ponto de inversão. Cada simulação contém 1000 episódios e 5 replicadas. Essas simulações foram realizadas com γ de 0,994, τ de 0,001 e redes neurais com duas camada *"full connect"* contendo 400 e 300 neurônios, respectivamente.

Tabela 5.3: Simulações do teste do ponto de inversão do ganho.

ID	Alg.	Rep	P_{Fin}	T. Apr.	F. Ativação
1	DDPG	5	0,0	0,001 e 0,0001	relu-relu-tanh
2	DDPG	5	0,1	0,001 e 0,0001	relu-relu-tanh
3	DQL	5	0,0	0,001	relu-relu-linear
4	DQL	5	0,1	0,001	relu-relu-linear
5	SAC	5	0,0	0,001 e 0,0001	relu-relu-tanh
6	SAC	5	0,1	0,001 e 0,0001	relu-relu-tanh
7	TD3	5	0,0	0,001 e 0,0001	relu-relu-tanh
8	TD3	5	0,1	0,001 e 0,0001	relu-relu-tanh

Para as simulações descritas nas Tabelas 5.1, 5.2 e 5.3, o aprendizado ocorreu de forma *on-line* com uma frequência de 1 teste para cada 10 episódios de treinamentos. Cada simulação foi replicada 5 vezes, assim permitindo o cálculo do erro padrão. Esse erro é uma medida que ajuda a verificar a confiabilidade da média amostral calculada. (Equação 5-3).

$$E_p \approx \frac{\sigma}{\sqrt{n}} \tag{5-3}$$

Para os resultados da atuação dos controladores, ordenou-se os 10 últimos episódios de cada replicada pela recompensa acumulada. O episódio com maior recompensa acumulada foi denominado como a melhor atuação. Denominou-se como atuação mediana, o episódio que alcançou a posição média na ordenação das recompensas acumulada.

5.2 Tennessee Eastman Process

As simulações do experimento de controle do nível e da temperatura do reator CSTR bifásico encontram-se na Tabela 5.4. Para estas simulações, manteve-se a quantidade de estados observados e ações próximo ao do estudo de caso 1 (4 estados observados e 2 ações). Essa manutenção é importante para avaliar a integração do *Tennessee* ao ambiente GRL e a viabilidade do controle por APR. Cada simulação contém 4000 episódios, 5 replicadas e duração de 60 minutos. Essas simulações foram realizadas com γ de 0,99309, τ de 0,001 e redes neurais com duas camadas *"full connect"* contendo 400 e 300 neurônios, respectivamente.

Tabela 5.4: Simulações do TE para o experimento 1.

ID	Alg.	Ambiente	Variáveis	Aproximador	F. Ativação
1	DDPG	GRL	4-2	Redes Neurais	relu-relu-tanh
2	SAC	GRL	4-2	Redes Neurais	relu-relu-tanh
3	TD3	GRL	4-2	Redes Neurais	relu-relu-tanh
4	AC	GRL	4-2	Tile Coding	-
5	QL	GRL	4-2	Tile Coding	-

No experimento do controle integral da planta, o primeiro conjunto de simulações do *Tennessee Eastman Process* foi elaborado para determinar quais eram os algoritmos mais promissores. Neste conjunto foram avaliados os algoritmos *off-policy* DDPG, TD3 e SAC e os *on-policy* TRPO e PPO. Na Tabela 5.5 encontram-se as simulações referente a este conjunto.

Tabela 5.5: Simulações do TE para *window* 1.

ID	Algoritmo	Ambiente	Variáveis	T. Apr.	Neurônios
1	DDPG	GRL	18-12	0,001 e 0,0001	400-300
2	SAC	GRL	18-12	0,001 e 0,0001	400-300
3	TD3	GRL	18-12	0,001 e 0,0001	400-300
4	CMA-ES	GRL	18-12	0,001 e 0,0001	50-50
5	TRPO	SPIN	18-12	0,001	64
6	PPO	SPIN	18-12	0,001	64
7	DDPG	GRL	41-12	0,001 e 0,0001	400-300
8	SAC	GRL	41-12	0,001 e 0,0001	400-300
9	TD3	GRL	41-12	0,001 e 0,0001	400-300
10	CMA-ES	GRL	41-12	0,001 e 0,0001	50-50
11	TRPO	SPIN	41-12	0,001	64
12	PPO	SPIN	41-12	0,001	64

O segundo conjunto de simulações foi elaborado para a determinar melhor quantidade de *strides* e *windows*. Por uma questão de tempo e por aparecerem mais promissores, apenas os algoritmos TD3, TRPO e CMA-ES foram avaliados neste conjunto de simulações. Na Tabela 5.6 encontram-se as simulações para *window* 2 e 3.

Tabela 5.6: Simulações do TE para *windows* 2 e 3.

ID	Algoritmo	Ambiente	Variáveis	T. Apr.	Neurônios	*Stride*
1	TD3	GRL	18-12	0,001 e 0,0001	400-300	1
2	TD3	GRL	41-12	0,001 e 0,0001	400-300	1
3	CMA-ES	GRL	18-12	0,001	50-50	1
4	CMA-ES	GRL	41-12	0,001	50-50	1
5	TRPO	SPIN	18-12	0,001	64	1
6	TRPO	SPIN	41-12	0,001	64	1
8	TD3	GRL	18-12	0,001 e 0,0001	400-300	3
9	TD3	GRL	41-12	0,001 e 0,0001	400-300	3
10	CMA-ES	GRL	18-12	0,001	50-50	3
11	CMA-ES	GRL	41-12	0,001	50-50	3
12	TRPO	SPIN	18-12	0,001	64	3
13	TRPO	SPIN	41-12	0,001	64	3
14	TD3	GRL	18-12	0,001	400-300	6
15	TD3	GRL	41-12	0,001	400-300	6
16	CMA-ES	GRL	18-12	0,001	50-50	6
17	CMA-ES	GRL	41-12	0,001	50-50	6
18	TRPO	SPIN	18-12	0,001	64	6
19	TRPO	SPIN	41-12	0,001	64	6

Para as simulações descritas nas Tabelas 5.4 e 5.6 foram estabelecidos a frequência de 1 teste para cada 10 episódios de treinamentos para o ambiente GRL. Cada simulação contém 4000 episódios, 5 replicadas e duração de 60 minutos.

6
Resultados e Discussões

Primeiramente, neste capítulo serão apresentados os resultados do reator de Van de Vusse. Estes resultados foram agrupados conforme as simulações descritas no capítulo anterior, constituídos do resultado da progressão do aprendizado e da atuação do controlador. Posteriormente, serão apresentados os resultados do *Tennessee Eastman Process*.

6.1
Reator de Van de Vusse

No experimento do Reator de Van de Vusse, utilizou-se a função de recompensa descrita na Equação 4-1, repetida na Equação 6-1.

$$r(t) = P_{F_{in}(t)}.\frac{F_{in}}{700} - (1 - P_{F_{in}}).|Sp_{C_B}(t) - C_B(t)| \tag{6-1}$$

Onde r é a recompensa por transição, Sp_{C_B} é o *setpoint* do composto B, C_B é a concentração de B na corrente de saída, F_{in} é vazão a alimentação do reator, $P_{F_{in}}$ é a influência da vazão sobre a recompensa.

6.1.1
Experimento 1 - Mudança de setpoint

O experimento 1 constitui na resposta do controlador a mudança do *setpoint* do composto B. Todos os episódios têm duração de 35 minutos e ocorre a alteração Sp_{C_B} de 0,9 para 1,1 mol/L no instante de 16 minutos, conforme a Equação 5-1, repetida na Equação 6-2.

$$Sp_{C_B}(t) = \begin{cases} 0,9 & \text{se } t \leq 16 \text{ min} \\ 1,1 & \text{se } t > 16 \text{ min} \end{cases} \tag{6-2}$$

Os gráficos de treinamento dos algoritmos de aprendizado por reforço para as simulações descrito na Tabela 5.1 do capítulo 5, encontram-se na Figura 6.1. Para calcular o intervalo de confiança, cada simulação foi replicada 20 vezes.

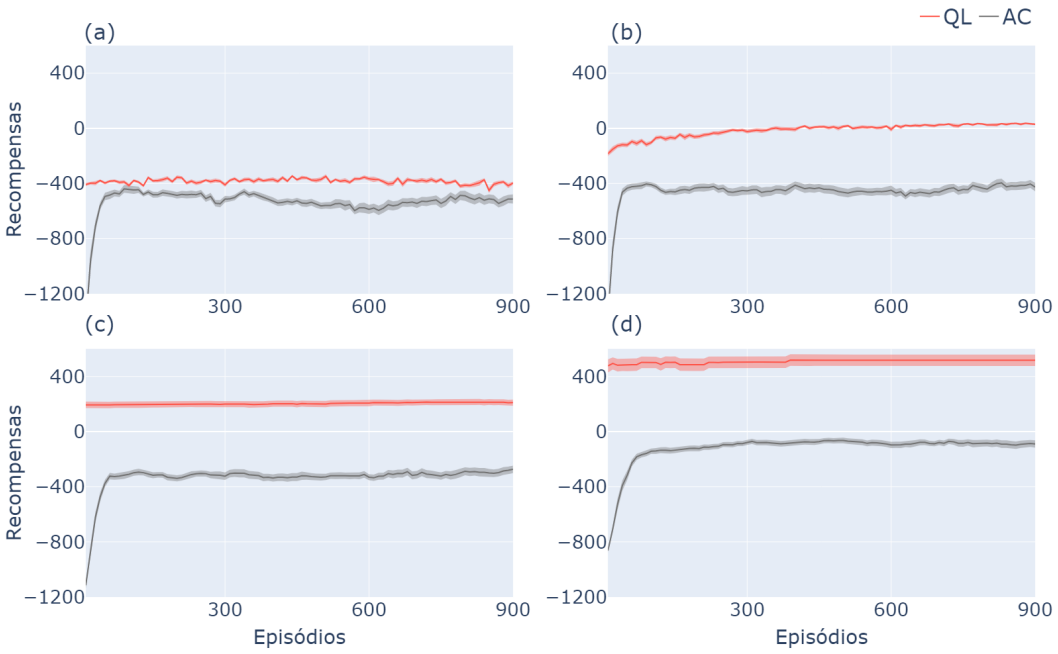

Figura 6.1: Aprendizado dos algoritmos de APR; (a) $P_{FIN} = 0{,}0$ (b) $P_{FIN} = 0{,}1$ (c) $P_{FIN} = 0{,}3$ (d) $P_{FIN} = 0{,}5$; nos eixos das abscissas encontram-se os números de episódios e nos eixos das coordenadas encontram-se as recompensas acumuladas durante o episódio de teste.

O algoritmo *Q-Learning* (QL) obteve as maiores recompensas com os quatro pesos de Fin. Para *Q-Learning*, observa-se uma diminuição do ruído de aprendizagem com o aumento de P_{FIN}. Entretanto, esse aumento está correlacionado com a ampliação da área do intervalo de confiança. Esta relação de proporcionalidade pode indicar convergências prematuras em diferentes mínimos locais, o que é prejudicial ao algoritmo. O AC obteve maior variação das recompensas acumuladas por episódio, pois provavelmente a política convergiu para um mínimo local. Este algoritmo possui uma certa tendência de convergência em mínimos locais (Figura 6.1).

O desempenho dos algoritmos AC e QL durante o episódio de teste para os diferentes pesos da vazão de alimentação encontram-se na Figura 6.2. As linhas contínuas representam o desempenho mediano enquanto a linha pontilhada a melhor atuação.

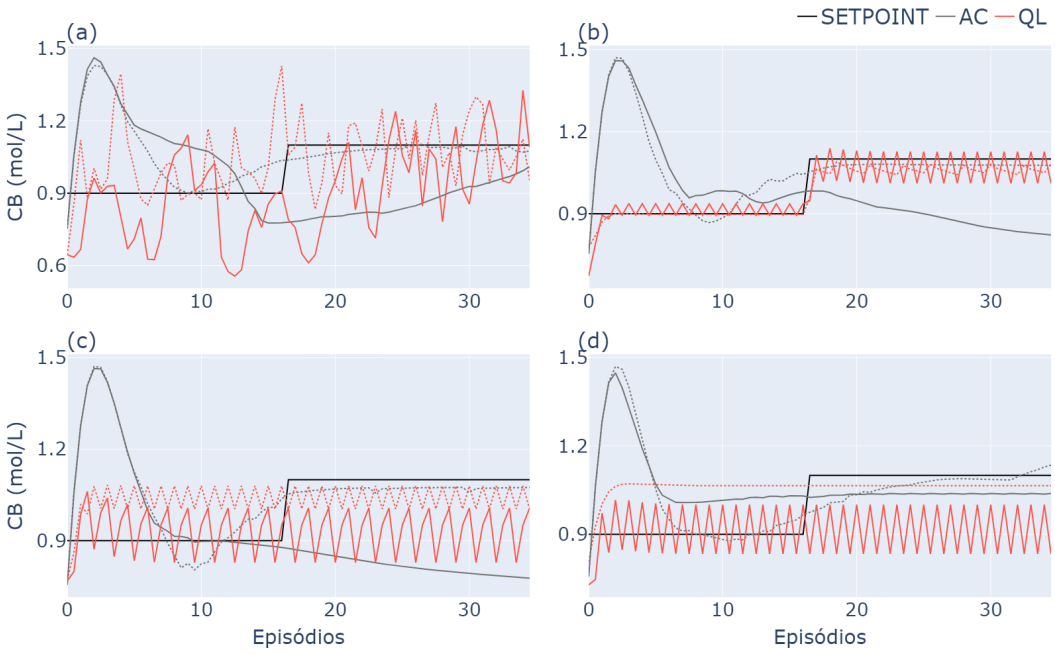

Figura 6.2: Desempenho dos algoritmos durante o episódio de teste para o peso da vazão de alimentação; (a) P_{FIN} = 0,0 (b) P_{FIN} = 0,1 (c) P_{FIN} = 0,3 (d) P_{FIN} = 0,5; nos eixos das abscissas encontram-se os números de episódios e nos eixos das coordenadas encontram-se as recompensas acumuladas durante o episódio de teste.

Analisando a Figura 6.2, observa-se que tanto o AC como QL não foram capazes de controlar o reator de forma satisfatória. Com P_{FIN} igual a "0", ambos algoritmos obtiveram uma atuação ruidosa e oscilatória. No P_{FIN} de "0,1", o QL conseguiu responder à mudança do *setpoint*, entretanto obteve uma ação oscilatória. Um melhor refinamento das malhas dos estados e das ações pode reduzir esta oscilação, entretanto essa estratégia pode aumentar o esforço computacional significativamente e dificultar o aprendizado (Figura 6.2.b).

Para o peso de "0,3", o QL manteve uma concentração constante de 1,00 mol/L ao longo de toda simulação. O *Actor Critic* apresentou um *overshoot* de 1,45 mol/L e oscilou entorno do *setpoint* de 1,00 mol/L a partir de 8 minutos. Percebe-se uma tentativa de aumento da concentração do composto B, porém alcançando o valor máximo de 1,05 mol/L (Figura 6.2.c).

No último e maior peso, o *Q-Learning* oscilou a concentração de B entre 0,98 e 1,06 mol/L durante maior parte do episódio. O *Actor Critic* apresentou um *overshoot* de mesma amplitude que o P_{FIN} "0,3", entretanto obteve um *offset* mais próximo de *setpoint* de 1,1 mol/L (Figura 6.2.d). Aparentemente a partir deste peso, ocorre uma redução significativa da importância do erro

operacional na função de recompensa (Equação 6-1). Essa redução é observada pela presença de *offset* nos pesos "0,3"e "0,5".

Os gráficos de treinamento dos algoritmos de aprendizado por reforço profundo para as simulações descritas na Tabela 5.2 do capítulo 5, encontram-se na Figura 6.3. Para cálculo do intervalo de confiança, cada simulação foi replicada 5 vezes.

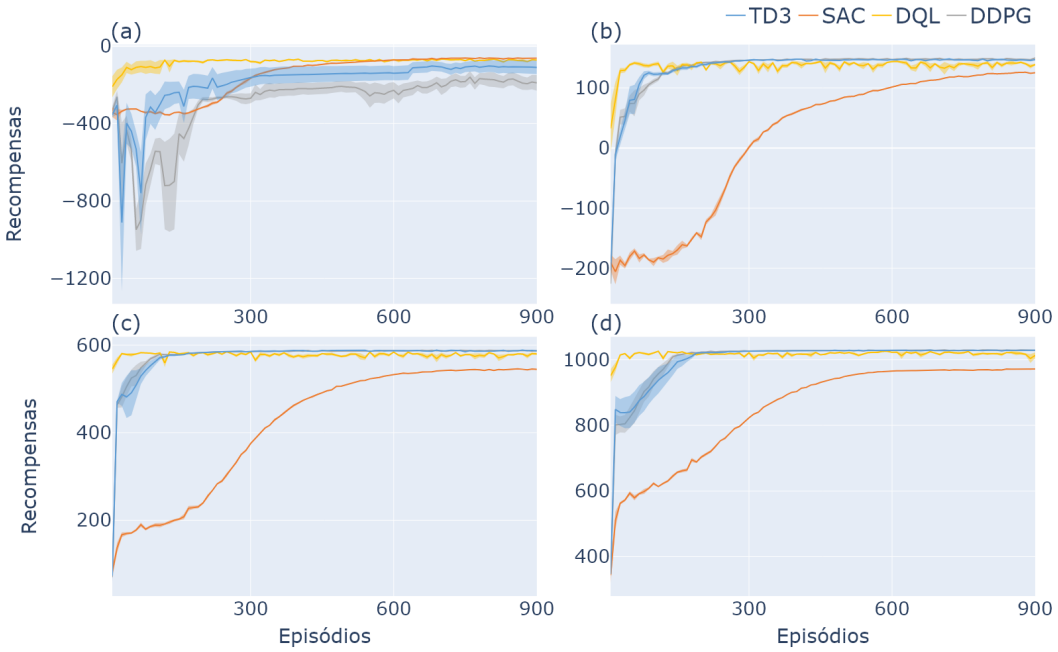

Figura 6.3: Aprendizado dos algoritmos de APRP (a) $P_{FIN} = 0$ (b) $P_{FIN} = 0{,}1$ (c) $P_{FIN} = 0{,}3$ (d) $P_{FIN} = 0{,}5$;

Para o P_{FIN} de 0, observa-se um aprendizado mais ruidoso e com maior erro padrão para todos os algoritmos. Este foi o peso que obteve maior diferença entre as recompensas médias acumuladas dos quatro algoritmos testados (Figura 6.3.a). Para os outros pesos da vazão de alimentação do reator, os algoritmos DDPG, DQL e TD3 convergiram para uma política com recompensas próximas. Para estes algoritmos, adição destes pesos a função de recompensa ajudaram a evitar mínimos locais, reduzir o ruído e o tempo de treinamento. O SAC apresentou um aprendizado mais lento e a política convergiu para um mínimo local, exceto para o peso da vazão de 0. Neste peso, o SAC obteve a recompensa acumulada de -62, enquanto DDPG, DQL e TD3 obtiveram -189, -71 e -109, respectivamente.

O desempenho e a atuação dos algoritmos durante o episódio de teste para o peso da vazão de alimentação $P_{FIN} = 0$ encontram-se na Figura 6.4. As linhas pontilhadas representam a melhor atuação e as linhas contínuas o desempenho mediano.

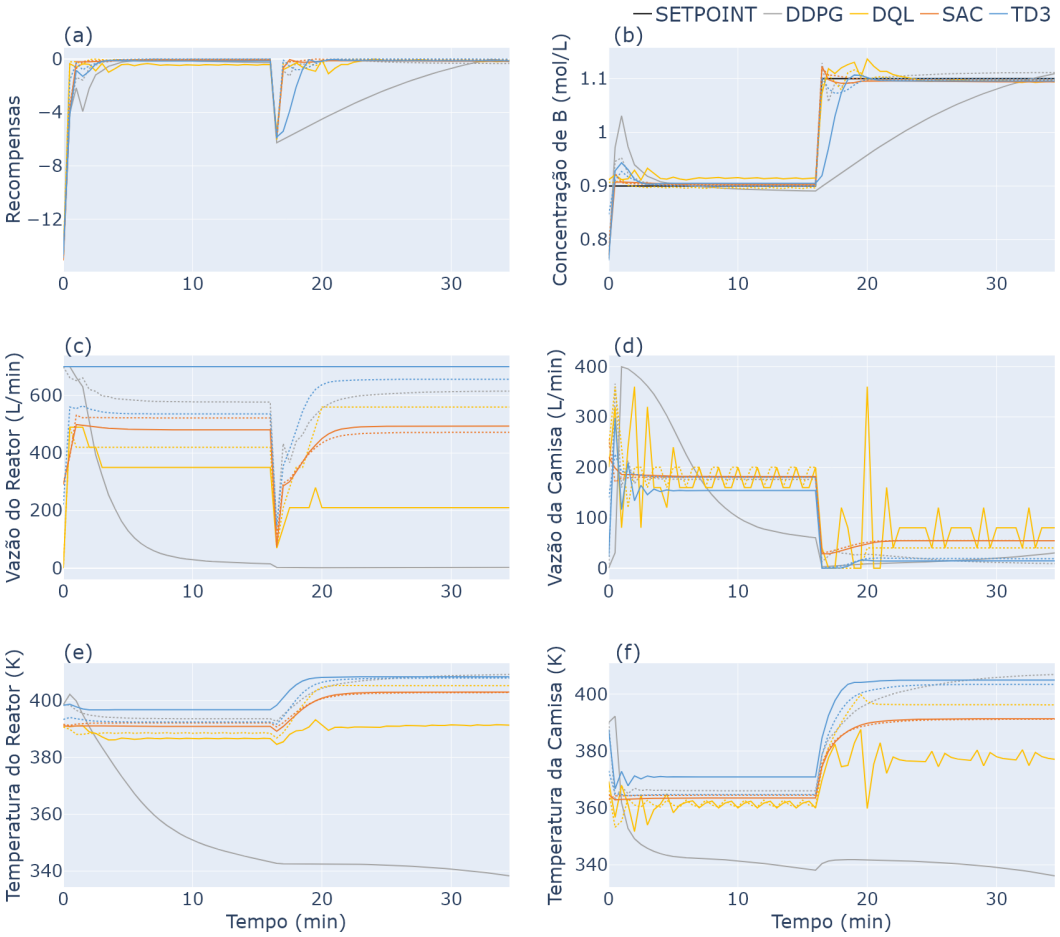

Figura 6.4: Resultados da simulação do episódio de teste para $P_{FIN} = 0{,}0$ a) Recompensas por passo; b) Concentração de B (variável controlada); c) Vazão de alimentação do Reator (variável manipulada); d) Vazão de alimentação da Camisa Térmica (variável manipulada); e) Temperatura do Reator; f) Temperatura da Camisa Térmica;

Para a simulação de mudança do *setpoint* estável com o $P_{FIN} = 0$, todos os algoritmos apresentaram desempenhos semelhantes, exceto o DDPG na atuação mediana que apresentou uma resposta lenta. (Figura 6.4.b).

Na mudança de *setpoint*, quase todos os algoritmos reduziram bruscamente a vazão de alimentação, sendo a maior delas por volta de 350 L/min. Essa mudança brusca pode ser indesejável em alguns processos devido ao maior desgaste das ferramentas de atuação, como válvulas, e pode ocasionar problemas nas próximas etapas do processo (Figura 6.4.c).

Os pontos de operação de cada algoritmo variaram, o DDPG apresentou uma vazão de alimentação de 577 L/min e temperatura do reator de 393 K, enquanto o DQL 490 L/min e temperatura de 391 K para o *setpoint* de 0,9 mol/L. Já no *setpoint* de 1,1 mol/L, o TD3 obteve uma vazão de 647 L/min e temperatura de 408 K, enquanto o SAC uma vazão de 571 L/min e temperatura de 404 K (Figuras 6.4.e 6.4.f). Para este peso, nenhum dos algoritmos otimizou a produção. As vazões de operações ficaram distantes da vazão máxima. O DQL por ser um algoritmo de ação discreta, nesse caso, com 11 ações para cada variável apresentou uma ação oscilatória. Maiorias dessas ações oscilaram entorno da atuação dos outros algoritmos (Figuras 6.4.d).

O desempenho e a atuação dos algoritmos durante o episódio de teste para o peso da vazão de alimentação $P_{FIN} = 0{,}1$ encontram-se na Figura 6.5. As linhas pontilhadas representam a melhor atuação e as linhas contínuas o desempenho mediano.

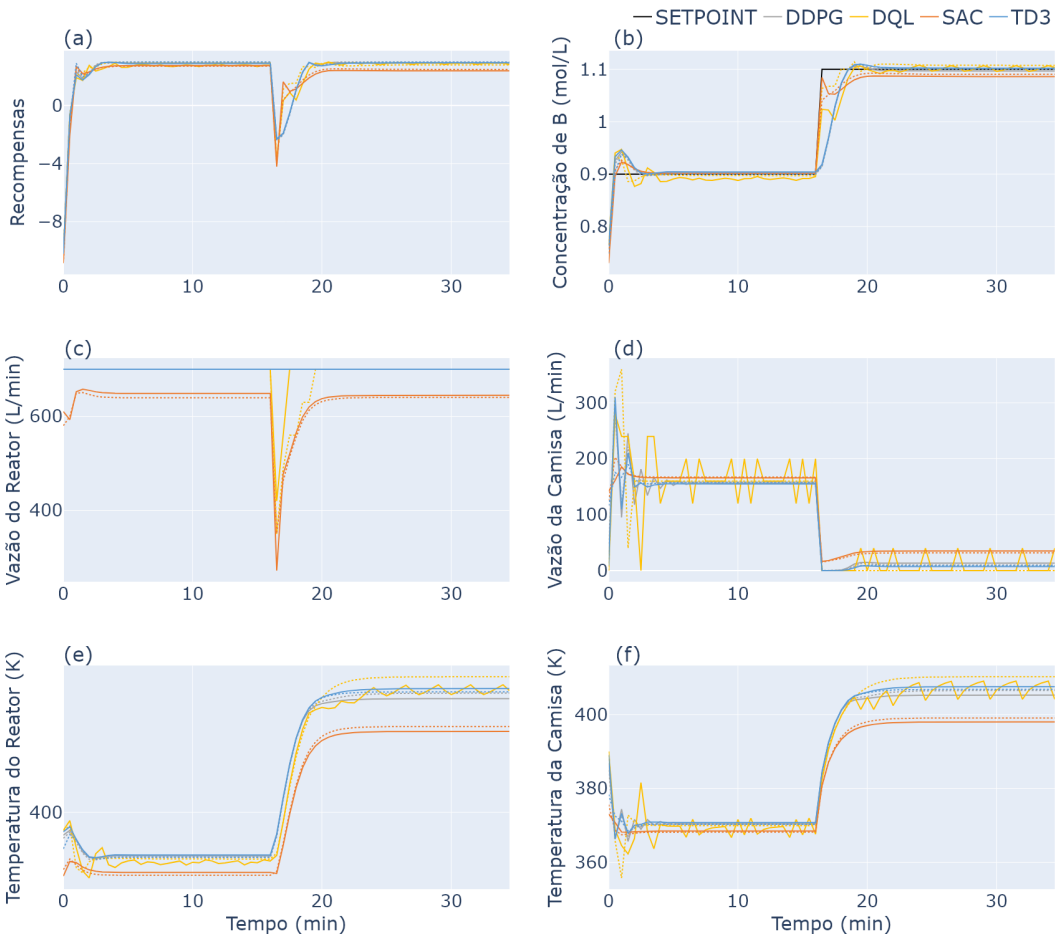

Figura 6.5: Resultados da simulação do episódio de teste para $P_{FIN} = 0{,}1$ a) Recompensas por passo; b) Concentração de B (variável controlada); c) Vazão de alimentação do Reator (variável manipulada); d) Vazão de alimentação da Camisa Térmica (variável manipulada); e) Temperatura do Reator; f) Temperatura da Camisa Térmica;

Para as simulações de $P_{FIN} = 0{,}1$, ambos os algoritmos apresentaram desempenhos e pontos de operações semelhantes. Na melhor atuação, o algoritmo DDPG apresentou um ITAE de 41, o DQL de 40, o SAC de 43 e TD3 de 42 (Figura 6.5.b). Estes valores são um pouco maiores quando comparados com as simulações sem a influência da vazão na função de recompensa. Adição da influência da vazão de alimentação à função de recompensa fez todos os algoritmos aumentarem o ponto de operação, para o SAC a vazão foi de 647 L/min e temperatura de 406 K, enquanto para o DDPG, TD3 e DQL a temperatura foi de 410 K e vazão de 700 L/min, o máximo estipulado pelo modelo (Figura 6.5.c).

Esse ponto de operação elevado pode ser um aspecto desejável devido à maximização da produção. Na mudança de *setpoint*, o DDPG e TD3 mantive-

ram a vazão de alimentação máxima de 700 L/min, enquanto o DQL temporariamente reduziu a vazão em 210 L/min e o SAC a reduziu em 368 L/min (Figuras 6.5.c e 6.5.d). A maximização da vazão de alimentação aumentou o tempo de resposta para todos os algoritmos de aprendizado por reforço profundo testados (Figura 6.5.b). Na camisa térmica, todos os algoritmos executaram ações parecidas. O DDPG apresentou uma ação oscilatória amortecida no *overshoot* inicial (Figura 6.5.d).

Na Figura 6.6 é apresentado o desempenho e a atuação dos algoritmos durante o episódio de teste para o peso da vazão de alimentação $P_{FIN} = 0{,}3$. As linhas pontilhadas representam a melhor atuação e as linhas contínuas o desempenho mediano.

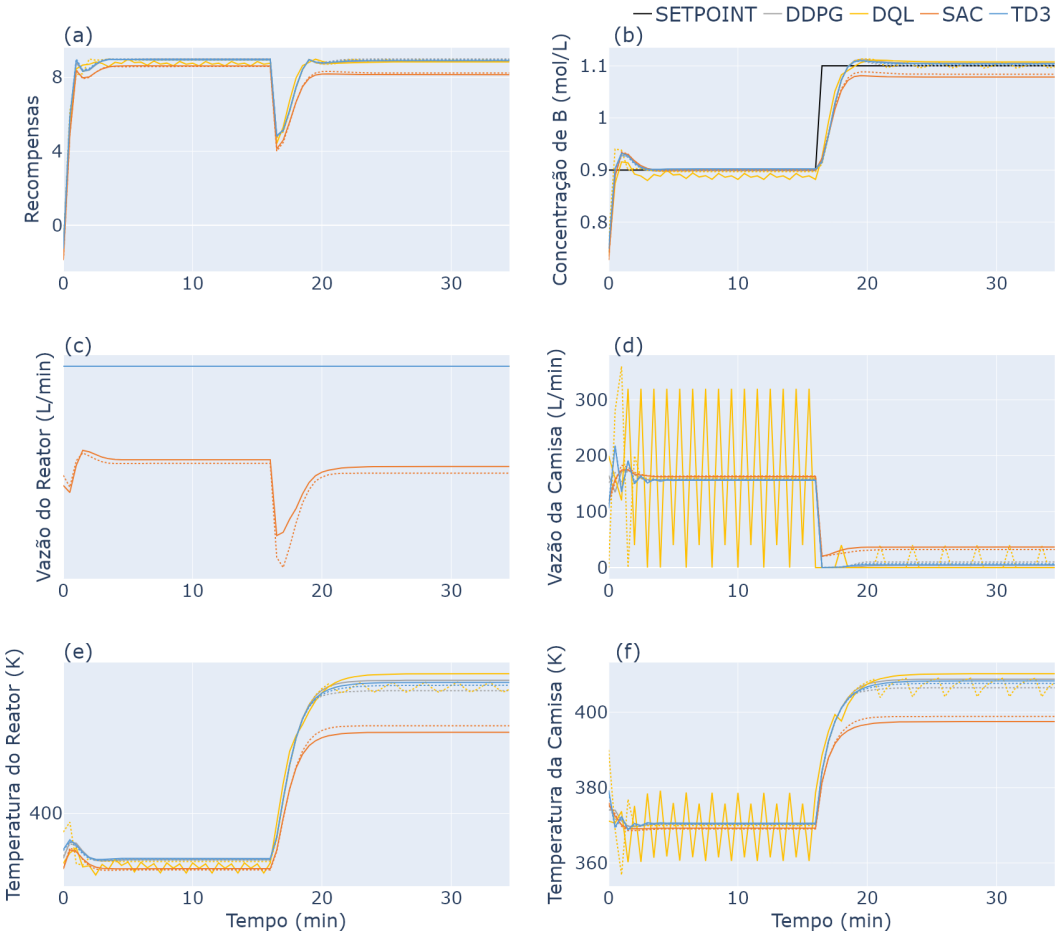

Figura 6.6: Resultados da simulação do episódio de teste para $P_{FIN} = 0{,}3$ a) Recompensas por passo; b) Concentração de B (variável controlada); c) Vazão de alimentação do Reator (variável manipulada); d) Vazão de alimentação da Camisa Térmica (variável manipulada); e) Temperatura do Reator; f) Temperatura da Camisa Térmica;

Todos os algoritmos testados apresentaram desempenhos similares (Figura 6.6.b). O aumento da influência da vazão de alimentação para 0,3 fez que DDPG, TD3 e DQL mantivessem à vazão máxima. No SAC, a redução da vazão foi de 35 L/min durante a mudança de setpoint (Figura 6.6.c). O SAC apresentou um *offset* de -0,02 mol/L em regime estacionário após a mudança do *setpoint*, entretanto a atuação dele, na vazão da camisa térmica apresentou oscilações com grande amplitude (Figura 6.6.d). O DQL apresentou um comportamento oscilatório com pequena amplitude entorno do *setpoint* (Figura 6.6.b). Dos algoritmos de APR testados, o DQL é único que possui uma atuação discreta. Esta característica pode ser causa deste comportamento oscilatório.

O comportamento dos algoritmos de controle para as simulações com $P_{FIN} = 0{,}5$ foram muito similares aos das simulações com $P_{FIN} = 0{,}3$. Por este motivo e para evitar redundâncias, optou-se por retirar os gráficos de atuação dos controladores deste presente trabalho.

Na Figura 6.7 encontra-se o gráfico de barra com ITAE (*integral of time-weighted absolute error*) para experimento de mudança do *setpoint*. Em cores mais escuras apresenta-se o desempenho da melhor atuação e em cores mais claro o desempenho mediano.

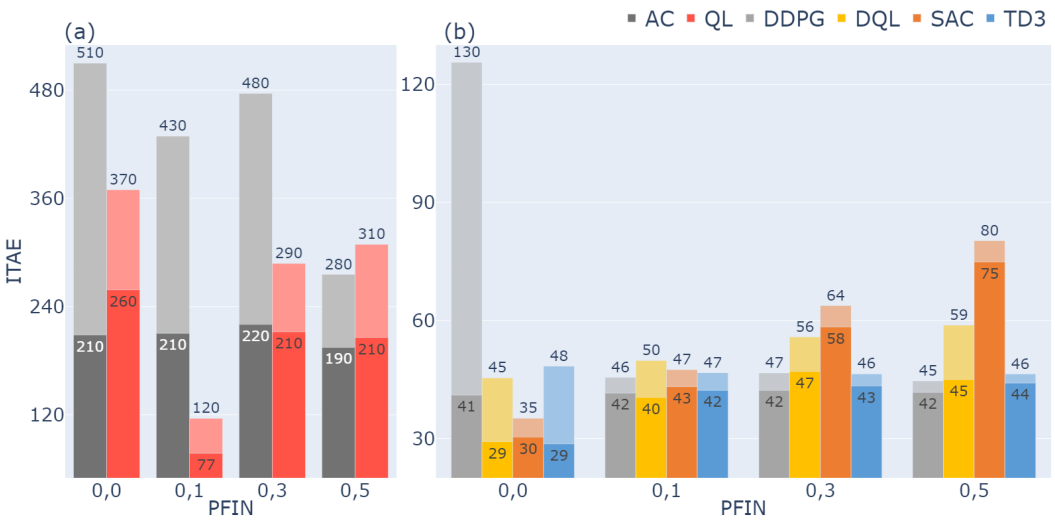

Figura 6.7: ITAE do Experimento 1: (a) APR (b) APRP

Analisando a Figura 6.7, percebe-se o desempenho superior dos algoritmos de aprendizado por reforço profundo em relação aos algoritmos clássicos. Um dos pontos a ser levantado em defesa aos algoritmos clássicos, é que não foram analisadas em profundidade técnicas diferentes de aproximação de ambiente como RBF, K-NN e outras. De modo geral, percebe-se uma diminuição

da diferença entre o desempenho da atuação mediana e ótima com aumento do P_{FIN}. Nos algoritmos de aprendizado por reforço profundo, observa-se um aumento do ITAE com peso, esse comportamento é esperado, pois, o agente está simultaneamente minimizando o erro em relação ao *setpoint* e maximizando a produção.

A presença de uma pequena influência da vazão de alimentação do reator na função de recompensa é o suficiente para o agente maximizar a produção com pequenos aumentos no ITAE. Outra vantagem desta adição é a maior estabilidade entre os melhores e medianos desempenhos, na maioria dos algoritmos. Devido ao desempenho insatisfatório dos algoritmos de APR clássicos e pequenas diferenças entre os maiores P_{FIN}, apenas os algoritmos de APRP e os pesos 0 e 0,1 serão avaliados no próximo experimento.

6.1.2 Experimento 2 - Ponto de Inversão

O segundo experimento do Reator de Van de Vusse constitui no teste do ponto de inversão do ganho. Neste teste, ocorre a alteração de *setpoint* para a região entorno do ponto de inversão. Em 12 minutos e meio de simulação, eleva-se Sp_{C_B} de 1,0 para 1,2 mol/L. Em 25 minutos de simulação, diminui-se o Sp_{C_B} para o valor estável de 0,9 mol/L (Equação 6-3).

$$Sp_{C_B} = \begin{cases} 1,0 & \text{se } t < 12{,}5 \text{ min} \\ 1,2 & \text{se } 12{,}5 \leq t \leq 25 \text{ min} \\ 0,9 & \text{se } t > 25 \text{ min} \end{cases} \tag{6-3}$$

Os gráficos de treinamento dos algoritmos de aprendizado por reforço profundo para as simulações entorno do ponto de inversão descrito na Tabela 5.3 do capítulo 5, encontram-se na Figura 6.8. Cada simulação contém 900 episódios e foi replicada cinco vezes.

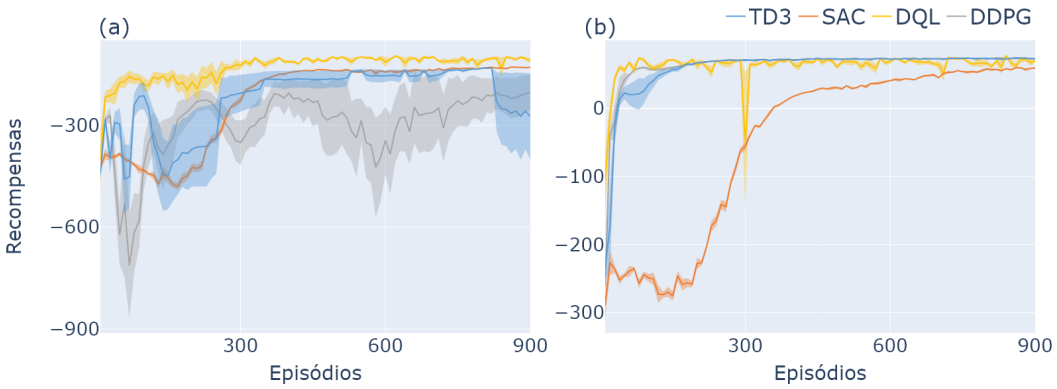

Figura 6.8: Aprendizado dos algoritmos de APRP. Nos eixos das abscissas são apresentados os números de episódios e nos eixos das coordenadas as recompensas acumuladas durante o episódio de teste; a) $P_{FIN} = 0$; b) $P_{FIN} =$ 0,1;

Analisando a Figura 6.8.a, percebe-se os algoritmos de DDPG e TD3 apresentaram grandes oscilações, chegando ocorrer em alguns momentos a desaprendizagem. Conforme foi observado no experimento anterior, o peso de vazão igual a zero apresentou ruído e intervalo de confiança maiores do que o peso igual a "0,1". Para o P_{FIN} de "0", o TD3 apresentou uma recompensa acumulada final de -274, o DDPG de -202, o SAC de -128 e o DQL de -108. Para P_{FIN} igual a "0,1", os algoritmos de DDPG, DQL e TD3 apresentaram uma convergência próxima e rápida. O DQL apresentou um aprendizado mais ruidoso. Apenas o SAC apresentou uma convergência mais lenta e uma recompensa menor. Para este peso, o SAC obteve uma recompensa final de 59, enquanto o DQL, TD3 e DDPG de 70.

O desempenho e a atuação dos algoritmos durante o episódio de teste para o $P_{FIN} = 0$, encontram-se na Figura 6.9. As linhas pontilhadas representam a melhor atuação e as linhas contínuas o desempenho mediano.

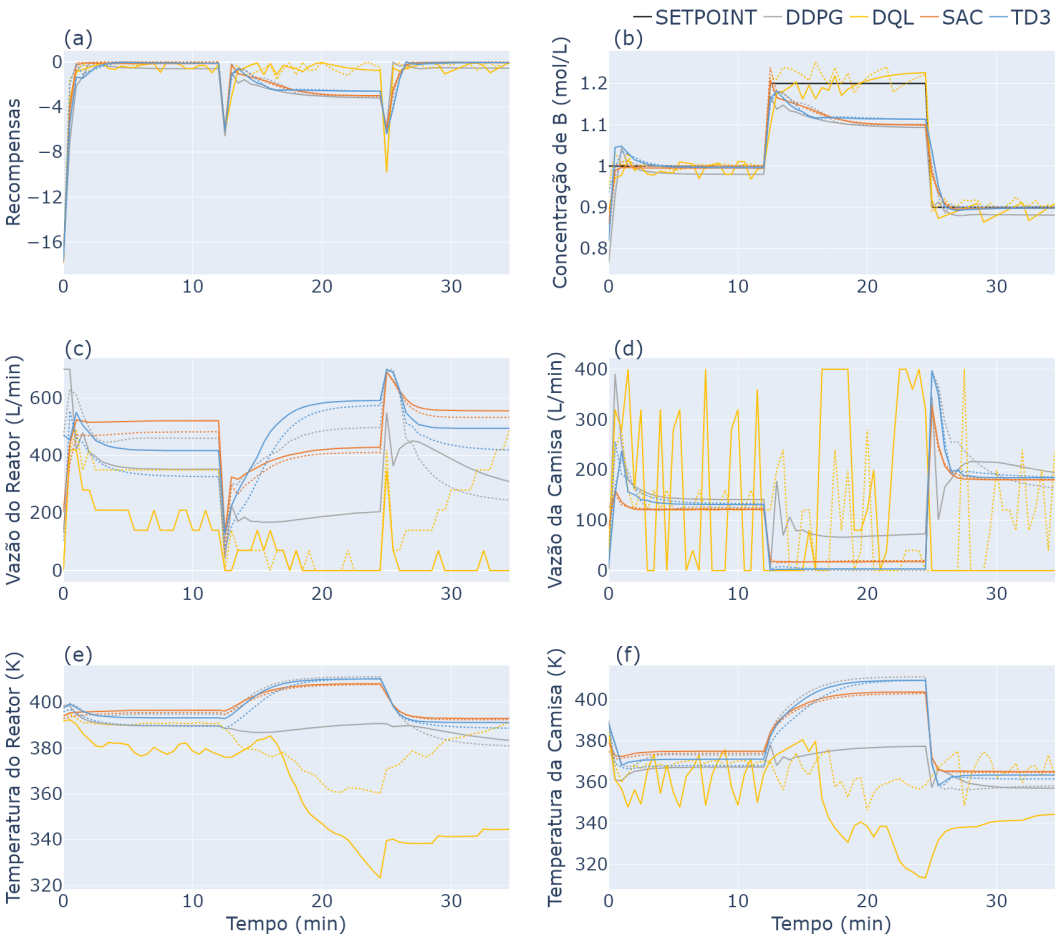

Figura 6.9: Resultados da simulação do episódio de teste para $P_{FIN} = 0{,}0$ a) Recompensas por passo; b) Concentração de B (variável controlada); c) Vazão de alimentação do Reator (variável manipulada); d) Vazão de alimentação da Camisa Térmica (variável manipulada); e) Temperatura do Reator; f) Temperatura da Camisa Térmica;

Todos os algoritmos conseguiram contornar o ponto de inversão e retornar ao *setpoint* estável de 0,9 mol/L. Na melhor atuação, os algoritmos SAC, TD3 e DDPG apresentaram ações parecidas. Entretanto, o SAC preferiu atuar em vazões mais baixas, enquanto o DDPG e TD3 em vazões maiores. O DDPG na atuação mediana, apresentou *offset* de cerca de 0,02 mol/L tanto no *setpoint* de 1 mol/L como no 0,9 mol/L.

O algoritmo *Deep Q-Learning* conseguiu alcançar o *setpoint* de 1,2 mol/L através do *overshoot*, praticamente fechando totalmente a alimentação do reator (Figura 6.9.c). O nível do reator constante é uma das considerações desta modelagem, ou seja, a vazão de entrada é igual vazão de saída. O DQL criou um sistema temporariamente fechado para realizar este feito. No desempenho mediano, o DQL apresentou uma atuação na vazão da camisa térmica com

oscilações com amplitude extremas, chegando a abrir e fechar totalmente vazão em poucos segundos.

Na Figura 6.10 encontra-se o desempenho e a atuação dos algoritmos durante o episódio de teste para o peso da vazão de alimentação igual a "0,1". As linhas pontilhadas representam a melhor atuação e as linhas contínuas o desempenho mediano.

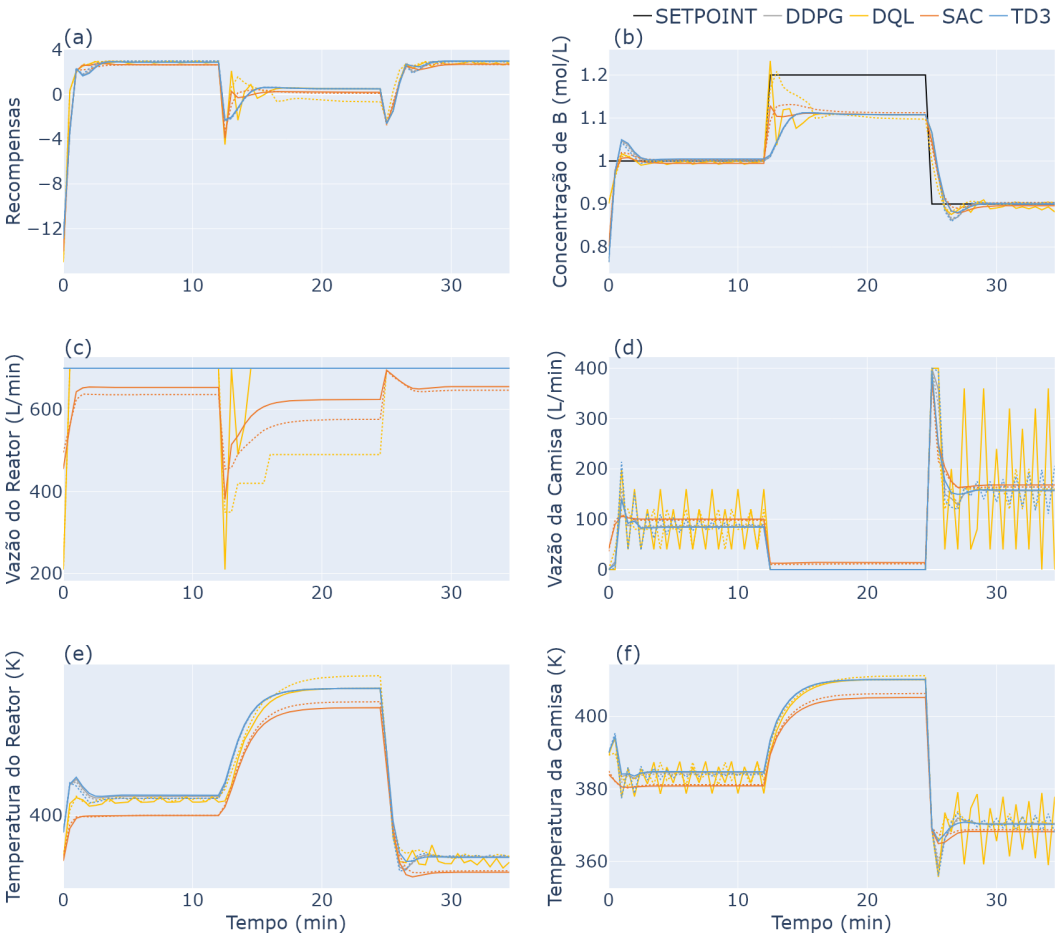

Figura 6.10: Resultados da simulação do episódio de teste para $P_{FIN} = 0{,}1$ a) Recompensas por passo; b) Concentração de B (variável controlada); c) Vazão de alimentação do Reator (variável manipulada); d) Vazão de alimentação da Camisa Térmica (variável manipulada); e) Temperatura do Reator; f) Temperatura da Camisa Térmica;

Todos os algoritmos conseguiram contornar o ponto inversão com excelência, lembrando que neste teste o mais importante é a capacidade de o controlador ser capaz de gerir o processo após alcançar o ponto de inversão. Os algoritmos TD3 e DDPG mantiveram a vazão de alimentação do reator no valor máximo ao longo de todo episódio. O DQL reduziu momentaneamente a

vazão para 630 L/min durante a mudança do *setpoint*. O SAC apresentou um ponto de operação com vazão inferior aos demais algoritmos (Figura 6.10.c).

Atuação na vazão de alimentação da camisa térmica foi similar para todos os algoritmos testados. Apenas o DQL e o TD3 apresentaram ações oscilatórias, principalmente no *setpoint* de 0,9 mol/L, enquanto o DDPG e SAC apresentaram ações contínuas e precisas (Figura 6.10.d)

O gráfico de barra com ITAE para experimento do ponto de inversão, encontra-se na Figura 6.11. Em cores mais escuras apresenta-se o desempenho da melhor atuação e em cores mais claro o desempenho mediano.

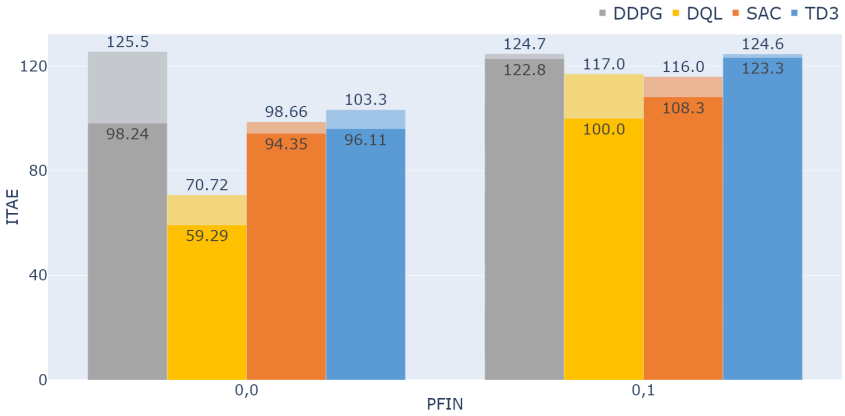

Figura 6.11: ITAE do experimento 2

Conforme visto no experimento anterior, percebe-se uma diminuição da diferença entre o desempenho da atuação médiana e melhor com aumento do P_{FIN}. Essa diminuição pode indicar maior estabilidade, a qual pode ser observada ao comparar as atuações dos gráficos das Figuras 6.9 e 6.10. Outra relação também observada é o aumento do ITAE com P_{FIN}, ocasionada pela atuação do agente que está simultaneamente minimizando o erro em relação ao *setpoint* e maximizando a produção.

Uma das vantagens do controlador multivariável é a possibilidade de ele enxergar diferentes modos de controlar um processo. Isso é consequência da sua capacidade de manipular várias variáveis de processo de forma integrada. Neste sistema, tanto a concentração de reagente como a temperatura conseguem interferirem drasticamente na produção de B. Sem a influência da vazão na função recompensa, o agente atua nas duas vazões simultaneamente obtendo uns curtíssimos tempos de resposta, entretanto para alguns algoritmos uma certa instabilidade. Quando se atribui pesos a diferentes, para cada uma das variáveis manipuladas, percebe-se nitidamente a preferência do agente em atuar na camisa térmica. Inesperadamente essa adição ocasionou uma maior robustez e estabilidade ao controlador.

6.2
Tennessee Eastman Process

Os resultados do controlador de APR para o controle do nível e da temperatura do reator do TE serão apresentados primeiro. Este experimento possui uma dimensionalidade próxima a do reator de Van de Vusse, 4 estados observados e 2 ações. Por último será exposto os resultados do controle integral da planta.

6.2.1
Experimento 1 — Controle do Reator

No experimento 1 do *Tennessee Eastman Process* utilizou-se a função de recompensa descrita na Equação 4-4, repetida na Equação 6-4. Para este experimento, a penalidade por desligamento foi -20k, a recompensa por tempo foi 50, o *setpoint* do nível foi 75 % e o *setpoint* da temperatura foi 120 °C.

$$r(t) = \begin{cases} t_{rw} - |Sp_{nível} - XMEAS(8)| - |Sp_{temp} - XMEAS(9)| & \text{se } t \leq t_t \\ P & \text{se } t = t_f \lneq t_t \end{cases} \tag{6-4}$$

Os gráficos de aprendizagem dos algoritmos de APR para as simulações descritas na Tabela 5.4, encontram-se na Figura 6.12. Cada simulação foi replicada 5 vezes.

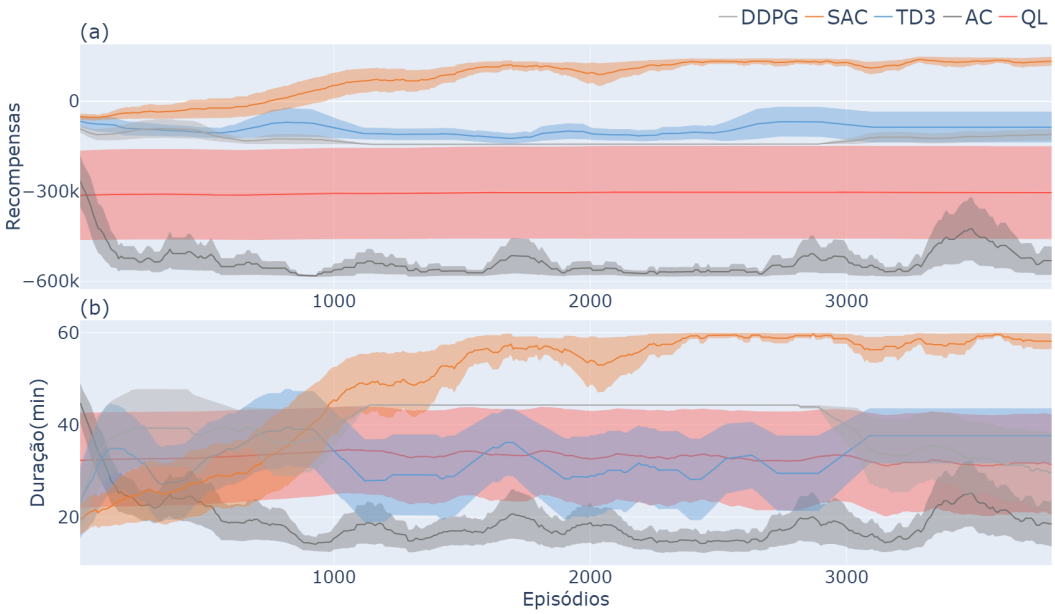

Figura 6.12: Gráficos de aprendizado para o experimento 1; a) Recompensas Acumuladas por episódio; b) Duração dos Episódios;

Para o experimento do controle do reator, apenas o algoritmo SAC convergiu para uma política que não ocasiona o desligamento da planta (Figura 6.12.b). O restante dos algoritmos apresentaram alguns episódios sem o desligamento da planta, entretanto a política não convergiu para este comportamento. A fraca propriedade Markov do ambiente pode ser uma das possíveis causas da falha na convergência para a política ótima.

O algoritmo *Q-Learning* apresentou um grande intervalo de confiança, praticamente constante. Este comportamento é ocasionado pela convergência em políticas sub-ótimas e mostra a dependência da inicialização da política para obter uma convergência ótima.

Os algoritmos de aprendizado profundo apresentaram um desempenho médio superior aos algoritmos clássicos. Os algoritmos profundos DDPG, SAC e TD3 apresentaram uma recompensa média final de -110k, 134k e -86k, respectivamente. Os algoritmos de APR clássicos AC e QL apresentaram uma recompensa média final de -531k e -304k, respectivamente (Figura 6.12).

O desempenho dos algoritmos durante o melhor episódio encontram-se na Figura 6.13. Para os algoritmos TD3, SAC e DDPG, as maiores recompensas acumuladas foram 164k, 167k e 104k, respectivamente. As maiores recompensas para QL foi 136k e o para AC de 145k. O desempenho do controlador PID, implementado no código FORTRAN do TE, está representado pela linha de cor preta.

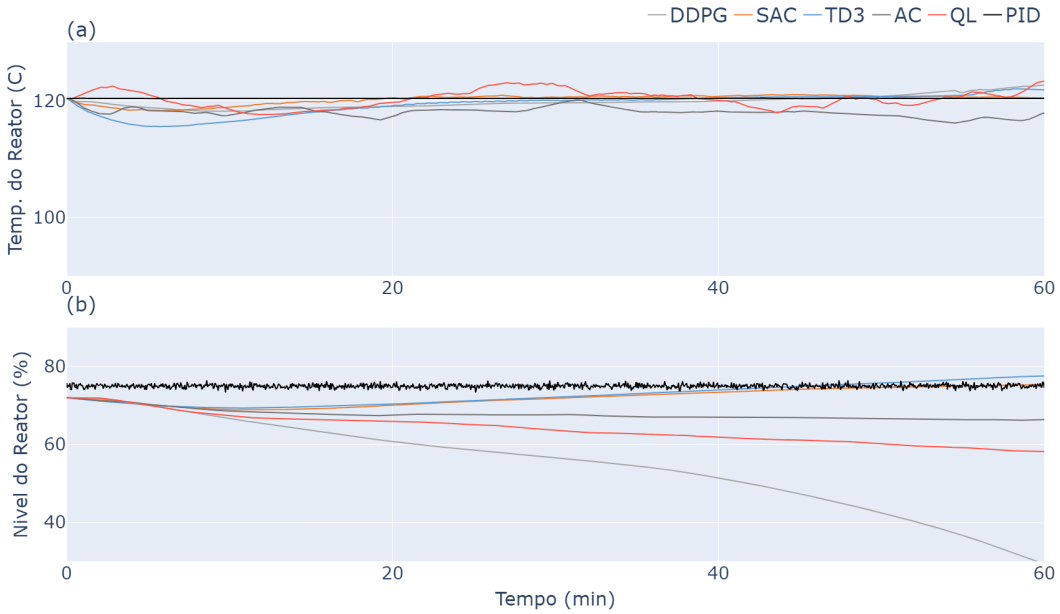

Figura 6.13: Desempenho do melhor episódio para o TE; a) Temperatura do reator; b) Nível do reator;

Analisando a Figura 6.13, percebe-se que os algoritmos de APR conseguiram manter a temperatura próximo ao *setpoint* de 120 °C, entretanto apenas o TD3 e o SAC conseguiram manter o nível próximo do *setpoint* de 75%. Provavelmente um ajuste da influência dos erros do nível e da temperatura na função recompensa possa melhorar o controle do nível do reator.

Os algoritmos AC e DDPG apresentaram um comportamento de decréscimo do nível do reator, sendo este comportamento mais acentuado no DDPG. Se a duração do episódio fosse maior, provavelmente ocorreria um desligamento da planta devido aos limites imposto pelo modelo para o nível do reator.

6.2.2 Experimento 2 — Controle Integral

No experimento de controle integral da planta do *Tennessee Eastman Process*, utilizaram os algoritmos de aprendizado por reforço profundo (APRP). Apesar de os algoritmos QL e AC serem capazes de não ocasionar o desligamento da planta no experimento 1, por serem algoritmos de estados discretos, sofrem do problema da dimensionalidade. Numa situação envolvendo um ambiente com mais de 41 estados observados como é o caso do controle integral, a utilização destes algoritmos se torna inviável.

Para este experimento, utilizou-se a função de recompensa descrita na Equação 4-5, repetida na Equação 6-5, onde a penalidade por desligamento

(P) foi -10k, a recompensa por tempo (t_{rw}) foi 25 e o *setpoint* de G (Sp_{C_G}) foi 90%.

$$r(t) = \begin{cases} t_{rw} - 100.C - 0,1.|Sp_{C_G} - \text{XMEAS}(40)| & \text{se } t \leq t_t \\ P & \text{se } t = t_f \leq t_t \end{cases} \quad (6\text{-}5)$$

Conforme descrito no capítulo 4 desta dissertação, estudou-se reduzir a quantidade de estados observados. Denominou-se como S1, a seleção de 18 variáveis observadas constituídas por XMEAS(2, 4, 5, 6, 7, 8, 9, 12, 13, 14, 15, 16,17, 32, 34, 37, 40) e S2 como a seleção contendo todas as 41 variáveis observadas.

Os gráficos de aprendizagem dos algoritmos de APRP para o primeiro conjunto de simulações descritos na Tabela 5.5 do capítulo 5, encontram-se na Figura 6.14. Para cálculo do intervalo de confiança, cada simulação foi replicada 5 vezes.

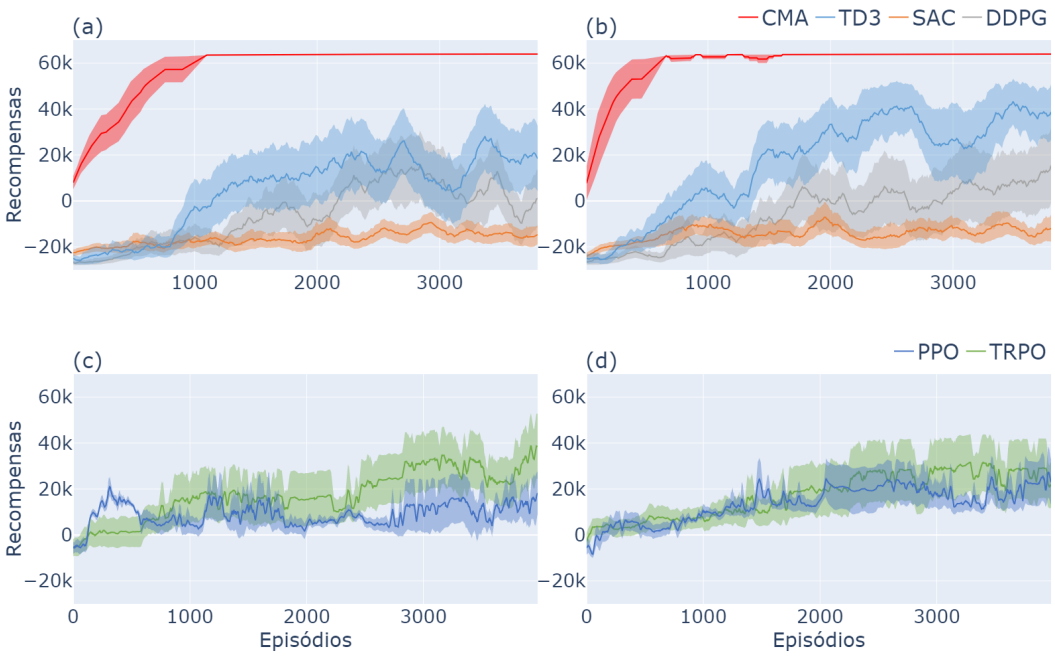

Figura 6.14: Gráficos de aprendizado do TE para *window* 1. Nos eixos das abscissas encontram-se os números de episódios e nos eixos das coordenadas encontram-se as recompensas acumuladas no final de cada episódio. a) Ambiente GRL e Seleção S1; b) Ambiente GRL e Seleção S2; c) Ambiente SPIN e Seleção S1; d) Ambiente SPIN e Seleção S2;

O algoritmo CMA-ES obteve as maiores recompensas acumuladas entre os algoritmos testados. Sua curva de aprendizado é sempre crescente devido à

preservação do melhor indivíduo a cada iteração do algoritmo. Esta preservação não é própria de algoritmos de APR, por isso é possível observar uma certa instabilidade no aprendizado (Figuras 6.14.a e 6.14.b).

Esta instabilidade e o grande intervalo de confiança mostram a dificuldade do controle do *Tennesse Estaman Process*. O agente aparenta estar andando numa corda bamba, qualquer deslize a planta desliga. Na Figura 6.14, a maioria das pontuações abaixo a 50k foi ocasionada pelo sistema de desligamento. Este mecanismo atrapalha consideravelmente a exploração do agente na busca da otimização da função valor.

Entre os algoritmos de APR *off-policy*, o TD3 apresentou o melhor resultado tanto na seleção S1 quanto na seleção S2 (Figuras 6.14.a e 6.14.b). Nos testes com as variáveis selecionadas, o DDPG, SAC e TD3 obtiveram a recompensa de pico igual 16k, -9k e 28k, respectivamente (Figura 6.14.a). Nos testes com todas as variáveis, o TD3 obteve a recompensa acumulada de pico igual 43k, enquanto o DDPG obteve 15k e o SAC obteve -7k (Figura 6.14.b).

Nos algoritmos *on-policy*, o TRPO apresentou o melhor resultado nos dois conjuntos de variáveis. Nas 18 variáveis selecionadas, o TRPO obteve uma recompensa acumulada de pico igual a 39k, enquanto o PPO obteve 21k (Figura 6.14.c). O TRPO e o PPO obtiveram 31k e 27k, respectivamente.

A Figura 6.15 apresenta o gráfico de aprendizado para *window* 2 dos algoritmos TD3, CMA-ES e TRPO, respectivamente.

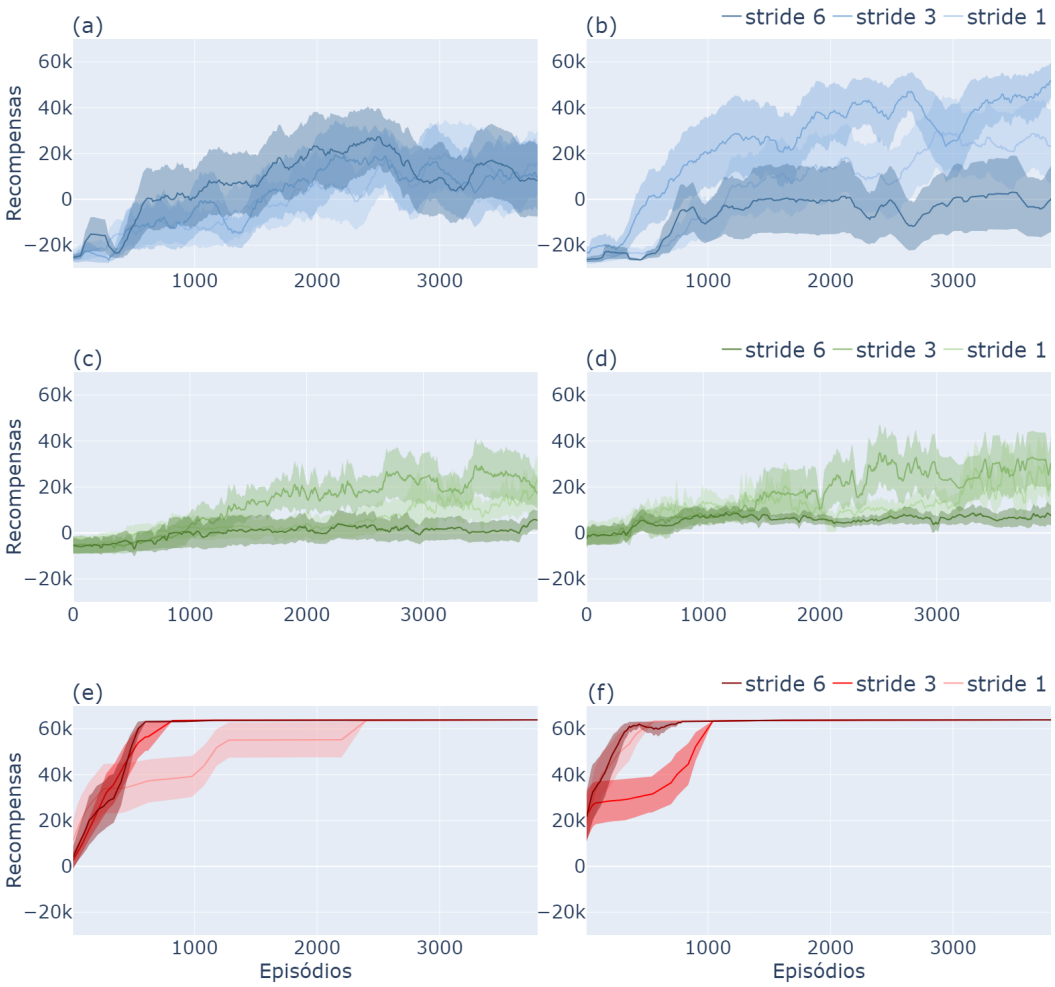

Figura 6.15: Gráficos de aprendizado para *window* 2. Nos eixos das abscissas encontram-se os números de episódios e nos eixos das coordenadas encontram-se as recompensas acumuladas no final de cada episódio; a) TD3 e Seleção S1; b) TD3 e Seleção S2; c) TRPO e Seleção S1; d) TRPO e Seleção S2; e) CMA-ES e Seleção S1; f) CMA-ES e Seleção S2;

Analisando a Figura 6.15, percebe-se que o *stride* igual a 3 apresentou o melhor resultado para maioria dos algoritmos e seleções de variáveis. Essa diferença aparenta ser menor na seleção S2 no algoritmo TD3. Para o TD3 S2, não se percebe uma diferença muito clara entre os *stride*, as áreas dos intervalos de confiança se sobrepõem em quase toda simulação (Figura 6.15.b). Nas simulações com seleção S1 para o *stride* 3, o CMA-ES, TD3 e TRPO obtiveram 64k, 19k e 29k, respectivamente (Figuras 6.15.e, 6.15.a e 6.15.c). Para a seleção S2, o TD3 obteve 51k, enquanto o CMA-ES, e TRPO obtiveram 64k e 35k, respectivamente (Figuras 6.15.b, 6.15.f e 6.15.d).

Não se percebe uma diferença considerável na maioria dos gráficos de

aprendizado das entre as seleções S1 e S2. A seleção S2 apresentou melhores resultados na maioria dos algoritmos para o *window* igual a 2, exceto TRPO, por uma diferença pequena. Aparentemente, o aumento da dimensionalidade do estado de 41 para 82 não foi prejudicial.

Os gráficos de aprendizado das simulações descritas na Tabela 5.6 para *window* 3 encontram-se na Figura 6.16.

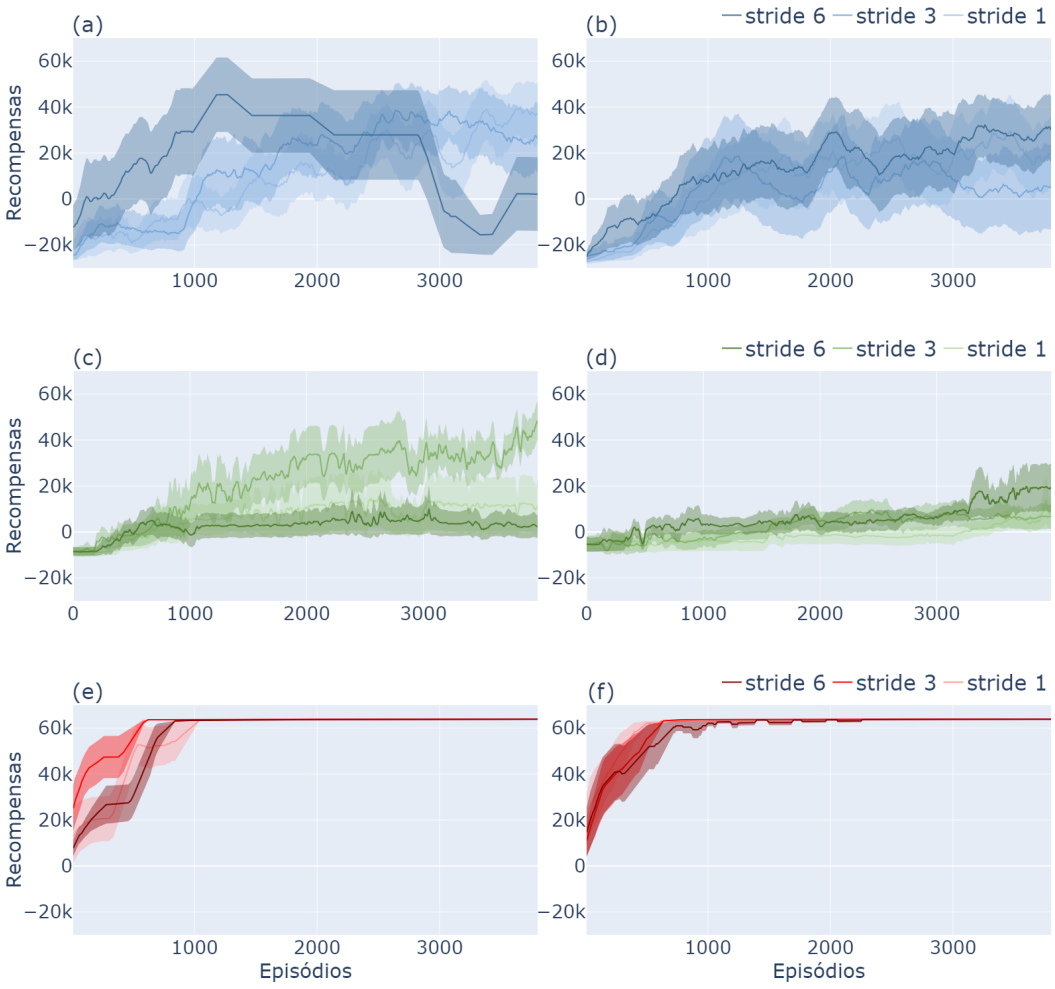

Figura 6.16: Gráficos de aprendizado para *window* 3; a) TD3 e Seleção S1; b) TD3 e Seleção S2; c) TRPO e Seleção S1; d) TRPO e Seleção S2; e) CMA-ES e Seleção S1; f) CMA-ES e Seleção S2;

Analisando a Figuras 6.16, percebe-se um resultado melhor na seleção S1 (3x18 = 54 estados observados). Aparentemente, a alta dimensionalidade do *window* 3 na seleção S2 (3x41 = 123 estados observados) impactou nos resultados dos algoritmos TD3 e TRPO. No *stride* 6, o algoritmo TD3 obteve a maior recompensa acumulada de pico de 45k, enquanto o S2 obteve uma recompensa de 28k (Figura 6.16.a). Para TRPO, a pontuação obtida na seleção

S1 foi 48k, enquanto a pontuação na seleção S2 foi de 13k (Figura 6.16.b). No TD3 seleção S2, não se percebe uma diferença clara entre os *stride*, as áreas dos intervalos de confiança se sobrepõem em quase toda simulação (Figura 6.16.a).

As configurações de seleções de variáveis, *window* e *stride* não afetaram a pontuação final do algoritmo CMA-ES. Um dos motivos que pode explicar este comportamento é o fato do CMA-ES não utilizar a propriedade de Markov no aprendizado. A diferença entre as recompensas acumuladas obtidas entre as configurações ficou na ordem das centenas. Estas configurações impactaram no tempo de convergência do algoritmo.

Nas Tabelas 6.1 e 6.2 encontram-se as médias das recompensas acumuladas médias dos últimos 20 episódios para cada configuração e algorítimo. Os valores encontrados nas tabelas encontram-se na ordem de milhares.

O primeiro aspecto peculiar dos valores contidos nas Tabelas 6.1 e 6.2 são os altos valores dos erros padrão na maioria das configurações e dos algoritmos. Essa característica mostra a dificuldade da convergência das políticas dos algoritmos de APRP para controlar o *Tennessee Eastman Process*. Em contrapartida, o algoritmo CMA-ES apresentou um pequeno erro padrão e praticamente convergiu para os mesmo valores, independente das configurações de *stride*, de *window* e de seleções de variáveis. Este comportamento é ocasionado pela preservação da melhor política ao longo do aprendizado.

Para a maioria das configurações, a seleção de variáveis S2 apresentou os melhores resultados. Essa tendência é invertida quando se comparam os valores das Tabelas 6.1 e 6.2 nas configurações de *window* 3. Uma das causas desse aspecto, deve ser a alta dimensionalidade do ambiente com *window* 3. Talvez outros métodos de seleção de variáveis diferentes do "CfsSubsetEval", possam apresentar ganhos significativos e possibilitar aplicações de maiores *window*.

Tabela 6.1: Médias das Recompensas para Seleção de Variáveis S1 ($\times 10^3$).

Alg.	Window 1	Window 2			Window 3		
		Stride 1	Stride 3	Stride 6	Stride 1	Stride 3	St6
CMA	63,9±0,03	63,9±0,01	63,9±0,03	63,9±0,02	63,9±0,02	63,9±0,03	63,9±0,04
TD3	18,4±14,2	15,9±13,8	9,2±13,9	8,3±15,9	37,5±13,2	26,4±15,1	2,±16,0
TRPO	-0,05±6,2	-5,1±4,05	-8,41±3,57	-6,03±3,96	-8,74±1,96	-5,35±5,18	-6,5±4,39
DDPG	1,05±12,6						
SAC	-14,7±4,15						
PPO	9,7±2,75						

Tabela 6.2: Médias das Recompensas para Seleção de Variáveis S2 ($\times 10^3$).

Alg.	Window 1	Window 2			Window 3		
		Stride 1	Stride 3	Stride 6	Stride 1	Stride 3	Stride 6
CMA	63,9±0,03	63,9±0,02	63,9±0,05	63,9±0,01	63,9±0,06	63,9±0,02	63,9±0,04
TD3	38,7±11,9	23,3±16,0	51,9±7,59	0,26±14,4	20,7±15,3	5,09±18,4	30,0±18,4
TRPO	-0,87±3,65	0,22±5,86	-3,5±4,11	-1,22±3,79	-4,72±4,24	-6,87±3,72	-7,91±3,71
DDPG	15,4±16,2						
SAC	-11,8±5,8						
PPO	4,54±4,14						

O desempenho e a atuação dos algoritmos durante o melhor episódio encontram-se na Figura 6.17. Para o algoritmo TD3, a maior recompensa acumulada foi 64 251 na configuração de seleção S1, *window* 3 e *stride* 3. Para CMA-ES, a maior pontuação foi 64 051 na configuração de seleção S2 de *window* 2 e *stride* 3. O TRPO obteve a recompensa de 63 863 para seleção S1 e *window* 1. O desempenho do controlador PID, implementado no código FORTRAN do TE, está representado pela linha de cor preta.

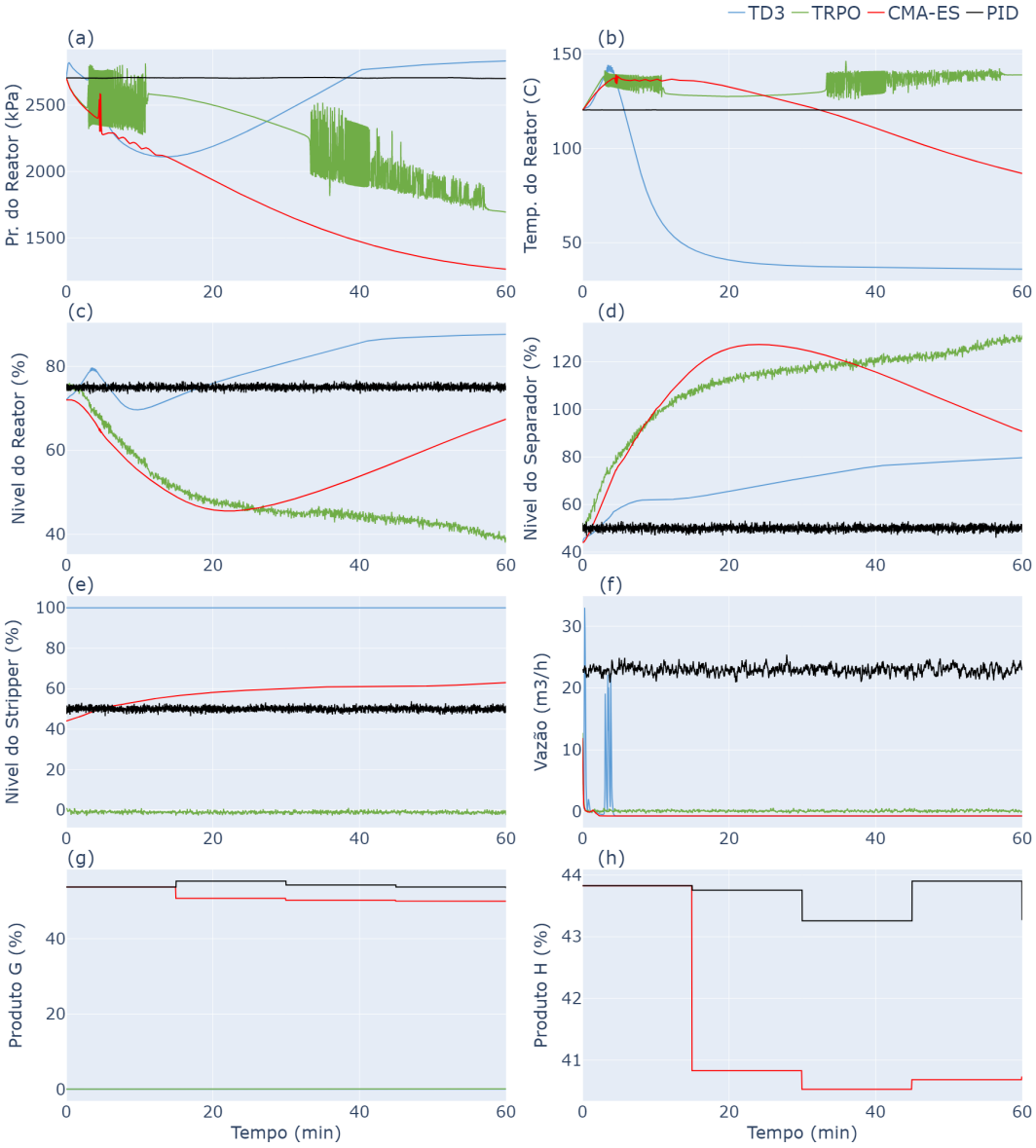

Figura 6.17: Desempenho do melhor episódio para o TE; a) Pressão do reator; b) Temperatura do reator; c) Nível do reator; d) Nível do separador; e) Nível do *stripper*; f) Vazão do produto; g) Concentração de G no produto; h) Concentração de H no produto

Analisando a Figura 6.17, percebe-se que nenhum algoritmo foi capaz de controlar a planta de forma estável. Provavelmente ocorreria um desligamento da planta, se aumentasse a duração do episódio. A maioria das variáveis de processo não alcançaram um estado estacionário durante o período da simulação, e algumas delas possuem tendência de ultrapassar os limites operacionais da planta. Atuação de todos os algoritmos diferiram consideravelmente do controlador PID (Figuras 6.17.a, 6.17.c e 6.17.d).

A baixa vazão da corrente 11 foi outro aspecto preocupante observado (Figura 6.17.f). Esta variável de processo indica a produção da planta. Aparentemente, este comportamento deve-se a inclusão e a minimização do termo do custo operacional na função de recompensa. Afinal, o menor custo operacional de uma planta é produzir nada.

Outro ponto preocupante da atuação dos controladores foi incapacidade da manutenção da concentração do composto G no *setpoint* de 90 %. O TD3 e TRPO mantiveram o percentual da concentração de G próximo a 0 na corrente 11, enquanto o CMA-ES manteve esse valor próximo 45 %. Em alguns casos estes valores podem ser consequência da baixa da formação dos produtos G e H no reator devido à baixa temperatura e baixa pressão, outros casos podem ser incapacidade de purificar dos produtos ou até a redução na alimentação dos reagentes devido à redução do custo operacional.

Um dos principais pontos que os resultados deste experimento mostra é que nem sempre uma alta recompensa acumulada é similar a um bom controlador. Era esperado um desempenho similar ao do controlador PID, entretanto nenhum deles chegou próximo. Provavelmente uma nova função de recompensa considerando mais variáveis e outras relações seja necessário para alcançar um desempenho satisfatório.

Os recursos de *window* e *stride* não foram capazes de compensar a fraca propriedade de Markov do sistema do TE. Esta incapacidade é mostrada pela não convergência dos algoritmos de APR. Todos eles possuem um aprendizado ruidoso e com um grande intervalo de confiança. Talvez modificação nos algoritmos de APR como apresentados por Yoo, Haeun et al. [25], sejam necessários para garantir a convergência da política.

Outro ponto a ser considerado é a quantidade de episódios de treinamento. Em modelos complexos e de alta dimensionalidade, como o *Tennessee Eastman Process*, é comum observar treinamentos extensos com mais de 10000 episódios. Entretanto, as simulações do TE contendo 4000 episódios demoram em média uns 3 dias, sendo executadas em uma NVIDIA RTX 2080 TI. Este tempo de execução, dificulta o escalonamento do número de episódios em estudo paramétrico.

7
Conclusões e Trabalhos Futuros

Neste trabalho avaliou-se o desempenho de controladores baseado em aprendizado por reforço frente a dois estudos de casos. O primeiro deles, o reator CSTR com cinética de Van de Vusse, um conhecido "*benchmark*" de controle de processo químico. Conforme descrito no capítulo 3, este processo apresenta características que dificultam o controle do mesmo. Entre essas peculiaridades encontram-se características não lineares como inversão de ganho, multiplicidade de entradas, resposta inversa e entre outros.

A maioria dos controladores APR multivariáveis apresentaram um excelente desempenho nos testes de mudança do *setpoint* e do ponto de inversão para o reator de Van de Vusse. Nestes testes, quase todos os controladores de APRP foram capazes de completar as tarefas de forma estável e minimizando o erro operacional. A influência de 10% do valor da vazão se mostrou suficiente para garantir uma pequena influência no erro e maximização da produção do reator.

No segundo estudo de caso, o *Tennessee Eastman Process* (TE) apresentou-se como um grande desafio devido à complexividade do modelo, a alta dimensionalidade e ao sistema de desligamento. Essas características, dificultaram a convergência dos algorítimos de APRP e o próprio pós-processamento de dados devido ao tamanho dos arquivos de resultados gerados. Inicialmente avaliou-se o comportamento dos algoritmos de APR controlando apenas o nível e a temperatura do reator, 4 estados observados e 2 ações. Os algoritmos SAC e TD3 conseguiram controlar o reator de forma satisfatória, entretanto com um desempenho inferior ao controlador PID incluso nas sub-rotinas originais do TE. Posteriormente, avaliou-se o desempenho dos algoritmos de aprendizado por reforço para o controle integral do TE, 41 estados observados e 12 ações. Infelizmente, apesar de alguns algoritmos alcançarem altos valores de recompensa acumuladas por episódio, não conseguiram controlar o processo de forma satisfatória e estável. Em algumas situações observou comportamentos preocupantes como a vazão do produto próxima a zero, os níveis de equipamentos instáveis e próximos do limite operacional. Estas situações talvez possam ser contornadas com uma função de recompensa mais restritiva e com maior tempo de aprendizagem.

Para trabalhos futuros, sugere-se a elaboração de uma nova função de recompensa com maiores restrições para o *Tennessee Eastman Process*. Provavelmente, modificações nos algoritmos de APRP talvez sejam necessários para garantir estabilidade ao aprendizado e a convergência para a política ótima. Outra possibilidade é um aumento significativo do tempo de aprendizagem. Praticamente quase nenhum algoritmo de APR apresentou uma convergência para o controle integral do TE.

Referências bibliográficas

[1] ANDREW, A. M.. **Reinforcement Learning: An Introduction**, volumen 27. 1998.

[2] WATKINS, C. J. C. H.; DAYAN, P.. **Q-Learning**. Technical report, 1992.

[3] JAAKKOLA, T.; JORDAN, M. I. ; SINGH, S. P.. **On the Convergence of Stochastic Iterative Dynamic Programming Algorithms**. Neural Computation, 6(6):1185–1201, 1994.

[4] MNIH, V.; KAVUKCUOGLU, K.; SILVER, D.; GRAVES, A.; ANTONOGLOU, I.; WIERSTRA, D. ; RIEDMILLER, M.. **Playing Atari with Deep Reinforcement Learning**. Technical report.

[5] MNIH, V.; KAVUKCUOGLU, K.; SILVER, D.; RUSU, A. A.; VENESS, J.; BELLEMARE, M. G.; GRAVES, A.; RIEDMILLER, M.; FIDJELAND, A. K.; OSTROVSKI, G.; PETERSEN, S.; BEATTIE, C.; SADIK, A.; ANTONOGLOU, I.; KING, H.; KUMARAN, D.; WIERSTRA, D.; LEGG, S. ; HASSABIS, D.. **Human-level control through deep reinforcement learning**. Nature, 518(7540):529–533, feb 2015.

[6] KINGMA, D. P.; BA, J.. **Adam: A Method for Stochastic Optimization**. dec 2014.

[7] SILVER, D.; HEESS, N.; DEGRIS, T.; WIERSTRA, D. ; RIEDMILLER, M.. **Deterministic Policy Gradient Algorithms**. Technical report.

[8] LILLICRAP, T. P.; HUNT, J. J.; PRITZEL, A.; HEESS, N.; EREZ, T.; TASSA, Y.; SILVER, D. ; WIERSTRA, D.. **Continuous control with deep reinforcement learning**. sep 2015.

[9] FUJIMOTO, S.; VAN HOOF, H. ; MEGER, D.. **Addressing function approximation error in actor-critic methods**. CoRR, abs/1802.09477, 2018.

[10] HAARNOJA, T.; ZHOU, A.; ABBEEL, P. ; LEVINE, S.. **Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor**. jan 2018.

[11] HAARNOJA, T.; ZHOU, A.; HARTIKAINEN, K.; TUCKER, G.; HA, S.; TAN, J.; KUMAR, V.; ZHU, H.; GUPTA, A.; ABBEEL, P. ; LEVINE, S.. **Soft Actor-Critic Algorithms and Applications**. dec 2018.

[12] SCHULMAN, J.; LEVINE, S.; MORITZ, P.; JORDAN, M. I. ; ABBEEL, P.. **Trust region policy optimization**, 2017.

[13] SCHULMAN, J.; WOLSKI, F.; DHARIWAL, P.; RADFORD, A. ; KLIMOV, O.. **Proximal policy optimization algorithms**, 2017.

[14] HANSEN, N.. **The cma evolution strategy: A tutorial**, 2016.

[15] DANG, V. H.; VIEN, N. A. ; CHUNG, T. C.. **A covariance matrix adaptation evolution strategy in reproducing kernel hilbert space**. Genetic Programming and Evolvable Machines, 20:479–501, 12 2019.

[16] HEIDRICH-MEISNER, V.; IGEL, C.. **Neuroevolution strategies for episodic reinforcement learning**. Journal of Algorithms, 64(4):152–168, 2009. Special Issue: Reinforcement Learning.

[17] NIAN, R.; LIU, J. ; HUANG, B.. **A review On reinforcement learning: Introduction and applications in industrial process control**. Computers Chemical Engineering, 139:106886, aug 2020.

[18] VITTHAL, R.; DURGAPRASADA RAO, C.. **Process control via artificial neural networks and learning automata**. In: PROCEEDINGS OF IEEE/IAS INTERNATIONAL CONFERENCE ON INDUSTRIAL AUTOMATION AND CONTROL, volumen 16, p. 329–334. IEEE, 1995.

[19] ALEX; ALEX; ALDRICH, C. ; ALDRICH, C.. **Plant-Wide Neurocontrol of the Tennessee Eastman Challenge Process using Evolutionary Reinforcement Learning**. Proceedings of the Third International Conference on Intelligent Processing and Manufacturing of Materials, 2001.

[20] LEE, J. M.; KAISARE, N. S. ; LEE, J. H.. **Choice of approximator and design of penalty function for an approximate dynamic programming based control approach**. Journal of Process Control, 16(2):135–156, feb 2006.

[21] TOSUKHOWONG, T.; LEE, J. H.. **Approximate dpamic programming based optimal control applied to an integrated plant with a reactor and a distillation column with recycle**. AIChE Journal, 55(4):919–930, apr 2009.

[22] HUBBS, C. D.; LI, C.; SAHINIDIS, N. V.; GROSSMANN, I. E. ; WASSICK, J. M.. **A deep reinforcement learning approach for chemical production scheduling**. Computers Chemical Engineering, 141:106982, oct 2020.

[23] SHAO, Z.; SI, F.; KUDENKO, D.; WANG, P. ; TONG, X.. **Predictive scheduling of wet flue gas desulfurization system based on reinforcement learning**. Computers and Chemical Engineering, 141, 2020.

[24] HWANGBO, S.; AL, R. ; SIN, G.. **An integrated framework for plant data-driven process modeling using deep-learning with monte-carlo simulations**. Computers Chemical Engineering, 143:107071, 2020.

[25] 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 and Chemical Engineering, 144:107133, jan 2021.

[26] KAISARE, N. S.; LEE, J. M. ; LEE, J. H.. **Simulation based strategy for nonlinear optimal control: Application to a microbial cell reactor**. International Journal of Robust and Nonlinear Control, 13(3-4):347–363, 2003.

[27] JOY, M.; KAISARE, N. S.. **Approximate dynamic programming-based control of distributed parameter systems**. In: ASIA-PACIFIC JOURNAL OF CHEMICAL ENGINEERING, volumen 6, p. 452–459, may 2011.

[28] MUNUSAMY, S.; NARASIMHAN, S. ; KAISARE, N. S.. **Approximate dynamic programming based control of hyperbolic PDE systems using reduced-Order models from method of characteristics**. Computers and Chemical Engineering, 57:122–132, oct 2013.

[29] SIDHU, H. S.; SIDDHAMSHETTY, P. ; KWON, J. S.. **Approximate dynamic programming based control of proppant concentration in hydraulic fracturing**. Mathematics, 6(8), aug 2018.

[30] ALVES GOULART, D.; DUTRA PEREIRA, R.. **Autonomous pH control by reinforcement learning for electroplating industry wastewater**. Computers Chemical Engineering, 140:106909, sep 2020.

[31] OH, D.-H.; ADAMS, D.; VO, N. D.; GBADAGO, D. Q.; LEE, C.-H. ; OH, M.. **Actor-critic reinforcement learning to estimate the optimal**

operating conditions of the hydrocracking process. Computers Chemical Engineering, 149:107280, jun 2021.

[32] VUSSE., J. G.. **Plug flow type reactor versus tank reactor**. 19:994–997, 1993.

[33] ENGELL, S.; KLATT, K.-U.. **Nonlinear control of a non-minimum-phase cstr**. p. 2941–2945, 1993.

[34] MONTANHEIRO, C. E.. **Estudo de um controlador preditivo não linear multivariável baseado em redes neuronais**. Master's thesis, Universidade Federal Do Rio de Janeiro, Rio de Janeiro, 2014.

[35] ALVARISTO, E. L.. **Controle preditivo adaptativo de processos**. Master's thesis, Universidade Federal Do Rio de Janeiro, Rio de Janeiro, 2014.

[36] LUZ, E. M. L.. **Desenvolvimento de controladores inteligenes para reator van de vusse**, 2018.

[37] DOWNS, J. J.; VOGEL, E. F.. **A plant-wide industrial process control problem**. Computers and Chemical Engineering, 17(3):245–255, 1993.

[38] ACHIAM, J.. **Spinning Up in Deep Reinforcement Learning**. 2018.

[39] HALL, M.; FRANK, E.; HOLMES, G.; PFAHRINGER, B.; REUTEMANN, P. ; WITTEN, I. H.. **The weka data mining software: An update**. SIGKDD Explor. Newsl., 11(1):10–18, Nov. 2009.