UNIVERSIDADE FEDERAL DO MARANHÃO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE ELETRICIDADE

CONVERGÊNCIA DO ESTIMADOR RLS PARA ALGORITMOS DE PROGRAMAÇÃO DINÂMICA HEURÍSTICA

ALLAN JAMES FERREIRA MACIEL

São Luís
2012

ALLAN JAMES FERREIRA MACIEL

CONVERGÊNCIA DO ESTIMADOR RLS PARA ALGORITMOS DE PROGRAMAÇÃO DINÂMICA HEURÍSTICA

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia de Eletricidade da Universidade Federal do Maranhão como parte dos requisitos necessários para obtenção do grau de Mestre em Engenharia Elétrica.

São Luís

2012

Maciel, Allan James Ferreira.

Convergência do estimador RLS para algoritmos de programação dinâmica heurística/ Allan James Ferreira Maciel – São Luís, 2012.

120 f.

Impresso por computador (fotocópia).

Orientador: João Viana da Fonseca Neto.

Dissertação (Mestrado) – Universidade Federal do Maranhão, Programa de Pós-Graduação em Engenharia de Eletricidade, 2012.

1. Controle ótimo. 2. Controle digital. 3. Programação dinâmica heurística. I. Título.

CDU 681.5.015.24

CONVERGÊNCIA DO ESTIMADOR RLS PARA ALGORITMOS DE PROGRAMAÇÃO DINÂMICA HEURÍSTICA

Allan James Ferreira Maciel

Dissertação aprovada em 28 de setembro de 2012.

Prof. João Viana da Fonseca Neto, Dr.
(Orientador)

Prof. Otacílio da Mota Almeida, Dr.
(Membro da Banca Examinadora)

Prof. Ginalber Luiz de Oliveira Serra, Dr.
(Membro da Banca Examinadora)

A Deus, fonte da vida.

A meus pais, pelo incentivo e carinho constantes.

Aos meus amigos, pelo apoio e companheirismo.

AGRADECIMENTOS

Primeiramente a Deus, pela oportunidade de viver este momento de realização acadêmica e profissional.

Aos meus pais: Maria das Graças Ferreira Maciel e Jornandes da Silva Maciel,por todo tipo de suporte dado durante a minha vida e por terem fornecido as condições para a completa execução deste trabalho.

Aos meus irmãos e familiares, pela confiança e apoio dados.

Ao meu orientador, professor João Viana da Fonseca Neto, pelaoportunidade, orientação, amizade e companheirismo que me foi dado para a realização deste trabalho e a todos os membros da banca examinadora.

Aos meus amigos do Laboratório de Controle de Processos da Universidade Federal do Maranhão, em especial a Gustavo Andrade, Leandro Lopes e Sarah Mesquita.

A minha namorada, Roberta Torres, pelo apoio e companheirismo.

A UFMA, PPGEE, DEE e a CAPES pelos recursos materiais e financeiros destinados a esse projeto.

Por fim, a todos aqueles que direta ou indiretamente, contribuíram para a elaboração desta dissertação de mestrado.

"*A vida só pode ser compreendida, olhando-se para trás; mas só pode ser vivida, olhando-se para frente*."

SorenKierkergaard

RESUMO

A união das metodologias de controle ótimo e de programação dinâmica tem impulsionado o desenvolvimento de algoritmos para realizações de sistemas de controle discreto do tipo regulador linear quadrático (DLQR). A metodologia utilizada neste trabalho é fundamentada sobre métodos de aprendizagem por reforço baseados em diferenças temporais e programação dinâmica aproximada. O método proposto combina a aproximação da função valor através do método RLS (mínimos quadrados recursivos) e iteração de política aproximada em esquemas de programação dinâmica heurística (HDP). A abordagem é orientada para a avaliação da convergência da solução DLQR e para a sintonia heurística das matrizes de ponderação Q e Rda função de utilidade associada ao DLQR. É realizada a investigação das propriedades de convergência relacionadas à consistência, excitação persistente e polarização do estimador RLS. A metodologia contempla realizações de projetos de forma online de controladores DLQR e é avaliada em um sistema dinâmico multivariável de quarta ordem.

Palavras-chave: Programação Dinâmica Heurística. Controle Multivariável. Controle Ótimo. Regulador Quadrático Linear Discreto. MínimosQuadradosRecursivos. Controle Digital.

ABSTRACT

The union of methodologies for optimal control and dynamics programming has stimulated the development of algorithms for realization of discrete control systems of the type linear quadratic regulator (DLQR). The methodology is based on reinforcement learning methods based on temporal differences and approximate dynamic programming. The proposed method combines the approach of the value function by method RLS (recursive least squares) and approximate policy iteration schemes heuristic dynamic programming (HDP). The approach is directed to the assessment of convergence of the solution DLQR and the heuristic weighting matrices Q and R of the utility function associated with DLQR. The investigation of convergence properties related to consistency, persistent excitation and polarization of the RLS estimator is performed. The methodology involved in a project achievements online DLQR controllers and is evaluated in a fourth order multivariable dynamic system.

Keywords: Heuristic Dynamic Programming. Multivariable Control.OptimalControl. Discrete Linear QuadraticRegulator. RecursiveLeastSquares. Digital Control.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

AD	*ActionDependent* (Dependente de Ação)
ADHDP	*ActionDependentHeuristicDynamicProgramming* (Programação Dinâmica Heurística Dependente de Ação)
ADP	*ApproximateDynamicProgramming* (Programação Dinâmica Aproximada)
AR	Aprendizagem por Reforço
DARE	*DiscreteAlgebraicRiccatiEquation* (Equação Algébrica Discreta de Riccati)
DLQR	*Discrete Linear QuadraticRegulator* (Regulador Linear Quadrático Discreto)
HDP	*HeuristicDynamicProgramming* (Programação Dinâmica Heurística)
LQR	*Linear QuadraticRegulator* (Regulador Linear Quadrático)
LS	*LeastSquares* (Mínimos Quadrados)
MIMO	*Multiple Input – Multiple Output* (Múltiplas Entradas – Múltiplas Saídas)
PD	Programação Dinâmica
PDM	Processo de Decisão Markoviano
RLS	*RecursiveLeastSquares* (Mínimos Quadrados Recursivos)
TD	*Temporal Difference* (Diferença Temporal)

SUMÁRIO

CAPÍTULO 1

INTRODUÇÃO

As grandes exigências para redução de custos de produção aliada às necessidades de aumento produtivo imposto pelo mercado consumidor têm forçado as indústrias a investirem cada vez mais no estudo de novas técnicas de controle para seus processos. O avanço industrial em resposta a estas exigências do mercado elevaram o grau de complexidade dos processos modernos, aumentando não apenas as interações entre suas variáveis, mas o número de malhas de controle necessárias para manter as condições de operação e qualidade final dentro dos limites estabelecidos [10] [28].

Nos processos complexos e com um alto grau de acoplamento entre as variáveis, não é possível atingir os altos níveis de desempenho utilizando apenas o controle regulatório simples [28]. Os projetos clássicos de sistemas de controle são geralmente processos de tentativa e erro em que diversos métodos de análise são utilizados de forma interativa para determinar os parâmetros aceitáveis de projeto. Um desempenho aceitável está intimamente definido em termos de margem de fase e ganho, largura de banda, máxima elevação, tempos de acomodação, subida e elevação, porém estas figuras de mérito não possuem valores otimizados [4] [5] [6]. Para sistema com múltiplas entradas e saídas, em que a complexidade se torna maior, necessita-se conhecer técnicas modernas de controle com novas abordagens[10].

Os métodos de controle ótimo têm alcançado grande sucesso para sistemas lineares (através das equações de *Riccati*), mas o sucesso é restrito a algumas aplicações, visto que os métodos exigem o conhecimento da dinâmica da planta, e não é de fácil aplicabilidade online. Em algumas aplicações, em que a

dinâmica da planta varia bastante, o projeto do controlador resultante do método clássico não é suficiente [32].

A programação dinâmica é um conjunto de técnicas de controle ótimo baseada no princípio da otimalidade de *Bellman* com aplicações em diversas áreas, desde a engenharia até finanças [7] [9] [24]. Podemos dizer ainda que a programação dinâmica é uma técnica computacional de otimização, e que freqüentemente, as soluções dos problemas são obtidas por um processo regressivo, trabalhando o problema do fim para o princípio. Deste modo a dificuldade será reduzida, ao decompor o problema numa seqüência de problemas interrelacionados mais simples [23] [24].

A utilização da quantização é de fundamental importância para a aplicação da programação dinâmica, e pode ser definida como o processo de atribuição de valores discretos para um sinal cuja amplitude varia entre infinitos valores, ou seja, variáveis são ditas quantizadas quando entre um valor que ela pode assumir e outro, existem valores proibidos [7] [15].

Os algoritmos para programação dinâmica adaptativa heurística (HDP) e dependente das ações heurísticas são implementados para o projeto de reguladores lineares quadráticos [29] [30] em sistemas multivariáveis que representam aeronaves.

Os desenvolvimentos do projeto são orientados para os esquemas críticos adaptativos em que se modela a função valorada em termos de recompensa de longo termo. Esta aproximação conduz ao desenvolvimento de métodos customizados [30] que atendem a determinadas situações, tais como: *deadline* de processos para o ajuste de ganhos em tempo real ou para o planejamento da operação. Ainda, outra vantagem do desenvolvimento de técnicas, baseados na metodologia adaptativa, é que não é necessárioa identificação explícita do sistema e os sistemas estocásticos podem ser inseridos (tratados) dentro deste contexto.

As variações do método adaptativo crítico são classificadas em diferenças temporais e aprendizado-*Q*. Por sua vez, estes métodos são classificados em programação dinâmica heurística (HDP) e programação dinâmica dual heurística (DHP). Por meio da abordagem dependente da ação os esquemas HDP e DHP são modificados de forma a realizar as estratégias

dependentes de ações (AD). A proposta de aproximação da função valorada para a determinação da solução da Equação *Hamilton-Jacobi-Bellman* (HJB) é utilizada para avaliar a política de controle (decisão). Em termos funcionais as HDPs são utilizadas na estimativa da função valorada e a programação dual heurística no gradiente da função valorada. As estratégias dependentes da ação estão associadas às estimativas da função Q e de seus gradientes em relação ao estado e a variável de decisão (lei de controle) [30].

Dentro desse contexto, propõe-se nesta dissertação, métodos de soluções de sistemas com controle discreto ótimo, utilizando-se de técnicas de otimização através de Programação Dinâmica Adaptativa. Uma das metas deste trabalho é a aplicação do controle ótimo discreto com ênfase no estimador *RLS* para sistemas dinâmicos multivariáveis.

1.1. Objetivos da dissertação

Abaixo são apresentados os objetivos geral e específico que este trabalho deseja alcançar.

1.1.1. Objetivo Geral

Pesquisar e desenvolver a teoria e sistematização do conhecimento para a realização do projeto de sintonia de um controlador discreto ótimo, por meio da programação dinâmica adaptativa para sistemas multivariáveis utilizando o aproximador dos mínimos quadrados recursivos (*RLS*) para estimação da função valor.

1.1.2. Objetivos Específicos

- Desenvolver algoritmos para o processo de convergência do estimador *RLS* em *ADP* para *LQR* em sistemas multivariáveis;
- Desenvolver algoritmos para a sintonia de controladores do tipo regulador linear quadrático discreto (*DLQR*);

- Investigar a teoria de programação dinâmica adaptativa (HDP e ADHDP) aplicada ao projeto de controladores;
- Investigar a aplicação e consistência dos estimadores LS (mínimos quadrados) e RLS (mínimos quadrados recursivos);
- Aplicar as técnicas de otimização através da programação dinâmica adaptativa em modelos de uma planta multivariável de quarta ordem;
- Implementar o projeto dos algoritmos propostos em plataforma do tipo MATLAB e verificar o seu desempenho em face a convergência.

1.2. Justificativas

O objetivo do controle ótimo é determinar o melhor sinal em um elenco de sinais de controle que levará o processo a satisfazer as restrições físicas e, ao mesmo tempo, minimizar (ou maximizar) algum índice de desempenho [5]. A determinação do sinal de controle ótimo pode ser feita através da programação dinâmica, que é um método de otimização matemática de processos utilizada para decisão de multiestágios. A condição do processo dentro de cada estágio é denominada de estado. Cada estágio inclui uma tomada de decisão que pode alterar o estado do processo representando uma transição do estado corrente e do estado futuro. Dentro do processo multiestágios, o objetivo da tomada de decisão é encontrar uma política ótima proveniente das decisões.

A política ótima para um processo de decisão multiestágios é determinada com base em uma abordagem alternativa que surgiu na década de 50, e recebeu o nome de princípio da otimalidade de *Bellman*, devido ao seu criador *Richard E.Bellman* [2] [7]. Portanto, independente do estado inicial e do controle nos estágios iniciais, o controle deve fornecer uma situação ótima com relação ao estado resultante, a partir do controle dos estágios iniciais.

O princípio da otimalidade desempenha um papel similar ao princípio mínimo de *Pontryagin* na abordagem variacional para controle de sistema, e serve para limitar, potencialmente, o número de estratégias de controle ótimo que devem ser investigadas. Sua importância está no fato de que permite otimizar sobre somente um vetor de controle ótimo devem ser determinadas operando-se

para trás no tempo. O problema de programação dinâmica é inerentemente a um problema recorrente no tempo [31].

Quando o sistema é modelado por uma dinâmica linear e a função de custo a ser minimizada é quadrática no estado e controle, então o controle ótimo é uma realimentação linear de estados, em que os ganhos são obtidos pela solução da equação de *Riccati*. Se o sistema for modelado por dinâmicas não-lineares ou a função de custo é não-linear quadrática, o controle ótimo de realimentação de estado dependerá da solução para a equação *de Hamilton-Jacobi-Bellman* (HJB), que geralmente é uma equação diferencial parcial não-linear ou equação a diferença para o caso discreto [8] [11].

Para que se possa utilizar a programação dinâmica para um projeto de um regulador quadrático linear discreto (DLQR), deve-se quantizar o estado e os valores de controle, isto é, devem ser restritos a fazerem parte de um conjunto de valores admissíveis. Portanto, quanto mais fina a quantização, mais fino será o controle, mas como o número de valores admissíveis para o controle e para os estados aumentam, o número de cálculos necessários para se obter a lei de controle ótima também aumentam significativamente [7]. Em função disto, o problema pode rapidamente tornar-se intratável, mesmo por um computador digital robusto. *Bellman* chamou este crescimento do número de cálculos necessários com o aumento dos níveis de quantização para entradas e estados de maldição da dimensionalidade [2] [3] [7] [9] [23] [24].Não importa o quão pequeno o estado é quantizado, a dinâmica do sistema pode resultar em valores de estados que não coincidem com os níveis de quantização. Nesse caso, deve-se utilizar a interpolação para que se possam encontrar os valores dos custos ótimos, e assim aplicar o princípio da otimalidade de *Bellman* [3] [7].

Ao longo dos anos, progressos têm sido feitos para contornar o problema da maldição da dimensionalidade [9] onde foi estabelecido um sistema chamado “crítico”, para a aproximação da função custo. A idéia principal desta solução é aproximar a solução da programação dinâmica utilizando uma estrutura de função de aproximação, tal como Redes Neurais, Lógica *Fuzzy*, ou Método dos Mínimos Quadrados, obtendo-se assim a função custo [32].

A programação dinâmica aproximada, denominada critica adaptativa, é apresentada no contexto do projeto do regulador linear quadrático discreto

(DLQR), levando em consideração as programações dinâmicas heurísticas ,dual e dependentes de ações (AD).

A HDP consiste de um arcabouço para o desenvolvimento de métodos (procedimentos) para estimar a função valorada. A estimação da função valorada para uma política ótima exige amostragens de função de recompensa. O modelo do ambiente e da recompensa instantânea é necessário para determinar a função valorada que corresponde da política ótima [30].

A programação dinâmica heurística dual (DHP) é um método que têm como objetivo a estimação do gradiente de custo, a partir de amostras da função que descreve o gradiente da função instantânea de recompensa.

Em aplicações industriais é verificado que a dinâmica da planta a ser controlada varia bastante durante a operação, e que os controladores projetados pelos métodos clássicos não apresentam um desempenho satisfatório pela dificuldade de ajuste dos parâmetros em decorrência dessa variação. Portanto, os métodos de controle ótimo utilizando inteligência computacional são propostos para solucionar essa necessidade de ajustes dos parâmetros do controlador, de forma mais eficiente através de dados reais da planta sem comprometer o seu desempenho.

1.3. Motivação

As indústrias vêm investindo cada vez mais nos estudos de novas técnicas para controle dos seus processos devido à necessidade do aumento da produção e principalmente pela redução dos custos de produção. Com o avanço industrial, o grau de complexidade dos processos modernos começou a envolver não apenas interações entre suas variáveis, mas interações entre um grande número de malhas de controle. A programação dinâmica adaptativa tem sido utilizada em uma vasta gama de sistemas industriais, nos quais podemos citar os sistemas de mísseis, aviões, sistemas de energia, sistemas de processos bioquímicos, etc [40].

Portanto, este avanço no campo industrial foi de grande estímulo para elaboração desta dissertação que utiliza técnicas modernas de controle e inteligência computacional para resolução de problemas práticos reais.

1.4 Contribuições

Dentre as principais contribuições dessa dissertação pode-se destacar:

1. Desenvolvimento e Avaliação do desempenho dos algoritmos de programação dinâmica adaptativa através dos aproximadores de função (LS e RLS).
2. Alocação de autoestrutura via programação dinâmica adaptativa para controle de sistemas multivariáveis.
3. Estudos de programação dinâmica adaptativa e aplicação de seus algoritmos em sistemas multivariáveis.

1.5 Organização da dissertação

Esta dissertação está organizada em Capítulos que descrevem o desenvolvimento da metodologia que será aplicada para obtenção e avaliação dos resultados computacionais. O Trabalho está organizado da seguinte forma:

Capítulo 2 – Neste capítulo são apresentados as teorias e conceitos sobre Controle Ótimo e Programação Dinâmica, que serão utilizados como base para o desenvolvimento da metodologia deste trabalho.

Capítulo 3 – Neste capítulo é apresentado a programação dinâmica adaptativa, assim como as principais definições para o início da implementação do algoritmo que será estudado nos capítulos posteriores. Também são apresentadas as estruturas básicas da programação dinâmica adaptativa que terá como foco neste trabalho, a programação dinâmica heurística.

Capítulo 4 – Dentro da teoria de controle ótimo alinhada com a teoria de programação dinâmica adaptativa, é necessário que se faça a estimação da função valor, daí a importância de um estudo detalhado dos aproximadores lineares, tendo como ênfase o estimador RLS, bem como seu estudo de convergência.

Capítulo 5 – Neste capítulo é feito um estudo detalhado do comportamento da planta que será utilizada para o teste do algoritmo desenvolvido posteriormente. Vale ressaltar que o sistema em questão, é um sistema multivariável de quarta ordem. O algoritmo HDP-RLS-DLQR é projetado neste capítulo e as explicações do passo a passo são baseadas nas equações apresentadas nos capítulos anteriores.

Capítulo 6 – Após o desenvolvimento do algoritmo e estudo do comportamento da planta na qual este será aplicado, deve-se fazer um estudo de avaliação de desempenho do algoritmo. Neste capítulo são adotadas várias abordagens para avaliar o algoritmo projetado no Capítulo 5, e um tópico que merece ser destacado é a convergência do algoritmo levando em consideração o fator de esquecimento do estimador RLS.

Capítulo 7 – Neste capítulo são apresentadas as conclusões, comentários e propostas futuras de trabalho.

Os apêndices são direcionados para complementar a fundamentação da abordagem considerada.

1.6 Artigos submetidos

João V. da Fonseca Neto, Leandro R. Lopes, Patrícia H. M. Rego, e Allan J. F. Maciel. *Algoritmos de Programação Dinâmica Heurística (HDP) e Dependente de Ação (ADHDP) para projetos de Sistemas de Controle DLQR.* São Luís, 2011.

João V. da Fonseca Neto, Leandro R. Lopes, Patrícia H. M. Rego, e Allan J. F. Maciel. *On the RLS Convergence of Heuristic and Action Dependent Dynamic Programming Algorithms for Tuning DLQR Controller*.São Luís, 2011.

CAPÍTULO 2

CONTROLE ÓTIMO E PROGRAMAÇÃO DINÂMICA

Os projetos utilizados em sistemas de controle clássico são geralmenteprocessos de tentativa e erro em que diversos métodos de análise são utilizados deforma interativa para determinar os parâmetros aceitáveis do projeto.

O controle ótimo é, geralmente, uma técnica de projeto *offline*que exigepleno conhecimento da dinâmica do sistema, por exemplo, no caso do sistema linear, deve resolver a equação algébrica de *Riccati*. Por outro lado, o controle adaptativo é um conjunto de técnicas de projeto *online*,que usa os dados medidos ao longo da trajetória do sistema para proporcionar um desempenho garantido por meioda compensação da dinâmica desconhecida, perturbações e erro de modelagem.

Este capítulo tem enfoque na caracterização do controlador ótimo para sistemas discretos utilizando programação dinâmica fundamentada no princípio da otimalidade e os conceitos básicos dos projetos de controladores que serãoutilizados no decorrer deste trabalho.

2.1. Controle Ótimo para Sistemas de Tempo Discreto

O problema do regulador é definido com referência a um sistema de entrada nula com o objetivo de guiar (orientar) os estados e saídas na vizinhança do estado de equilíbrio. As condições de entradas zero não são severas para o projeto. O regulador linear quadrático (tempo infinito) garante a estabilidade do sistema com algumas características de amortecimento e desempenho satisfatório quandoas entradas são diferentes de zero.

Os sistemas dinâmicos são representados através de equações diferenciais ordinárias (EDOs) não lineares,por análises físicas e matemáticas, da forma$\dot{x} = f(x,u)$, com o estado $x(t) \in R^n$ e entrada de controle $u(t) \in R^m$ no espaço contínuo. Alguns métodos padrões para a discretização dosistema continuo, convenientes para o controle por computador,podem ser utilizados. O resultado da discretização pode ser expresso em espaçode estados na forma de$x_{k+1} = F(x_k, u_k)$, com k sendo o tempo discreto.

A maioria dos estudos em ADP tem sido realizada para sistemas que operamem tempo discreto. Dessa forma, consideramos o seguinte modelo do sistema dinâmico discreto:

$$x_{k+1} = f(x_k) + g(x_k)u_k \tag{2.1}$$

com o estado $x_k \in R^n$ e a entrada de controle $u_k \in R^m$. A política de controle é definida como uma função de $h(.) : R^n \rightarrow R^m$. Isto é, para cada estado x_k tem-se uma ação de controle definida da seguinte forma:

$$u_k = h(x_k) = -Kx_k \tag{2.2}$$

Estes mapeamentos são também conhecidos como controladores de realimentação por estado. Vários métodos podem ser utilizados para a obtenção da política de controle de realimentação, dentre eles tem-se o controle ótimo através da resolução da equaçãode *Riccati*, controle adaptativo, controle no domínio da frequência, etc. No Aprendizado por Reforço, a política de controle éobtida (aprendida) em tempo real a partir de estímulos recebidos do ambiente.

Na Aprendizagem por Reforço, o ator é o agente que gera a política de controle, que matematicamente está representado pela Eq.(2.2), e tem como entrada x_ke saída u_k. É necessário ressaltar que tudo fora do ator é considerado como ambiente, ou seja, o sistema da Eq.(2.1), assim como todos os distúrbios possíveis gerados compõe o ambiente.

Partindo do princípio de aprendizagem por reforço (AR), define-se a função de custo, para horizonte infinito, como:

$$V_h(x_k) = \sum_{i=k}^{\infty} \gamma^{i-k} r(x_i, u_i) = \sum_{i=0}^{\infty} \gamma^{i} r(x_{i+1}, u_{i+1}) \tag{2.3}$$

Sendo γ um fator de desconto onde $0 < \gamma \leq 1$ e $u_k = h(x_k)$, a política de controle de realimentação. A função $r(x_k, u_k)$ é conhecida como fator de utilidade, ou aprendizado por reforço, sendo a medida de um passo da função custo. Esta função pode ser selecionada baseada em algumas considerações como energia mínima, risco mínimo, etc. Uma forma padrão utilizada na área de controle é a função quadrática dada por $x_k^T Q x_k + u_k^T R u_k$.

Um sistema é dito estabilizável em um conjunto $\Omega \in R^n$, se existir uma política de controle $u_k = h(x_k)$ em que o sistema da Eq. (2.1) em malha fechada seja assintoticamente estável em Ω. A política de controle $u_k = h(x_k)$ é dita admissível se é estabilizável e produz um custo finito $V_h(x_k)$. Portanto, o objetivo do controle ótimo é selecionar a política que minimiza:

$$V^*(x_k) = \min_{h(.)} \left(\sum_{i=k}^{\infty} \gamma^{i-k} r\big(x_i, h(x_i)\big) \right) \tag{2.4}$$

Logo a política ótima de controle é dada por:

$$u_k^* = h^*(x_k) = \arg\min_{h(.)} \left(\sum_{i=k}^{\infty} \gamma^{i-k} r\big(x_i, h(x_i)\big) \right) \tag{2.5}$$

Observando a Eq.(2.3), percebe-se que no campo da inteligência computacional, esta é interpretada como reforço e o objetivo é maximizá-la. Portanto, a maneira de como o valor de custo é encontrado, é a principal diferença entre controle com realimentação e aprendizagem por reforço.

2.2. Programação Dinâmica e Princípio da Otimalidade

A otimalidade dos controladores é garantida por meio da equação de*Riccati*, e através de uma discretização do sistema dinâmico, o DLQR

(regulador quadrático linear discreto) pode sercaracterizado como um problema de possíveis estágios e estados, tendo-se a possibilidadeem estabelecer um caminho das trajetórias através da Programação Dinâmica.O projeto DLQR clássico pode ser afetado pelo uso do princípio da Otimalidade, e este projeto é conhecido como método de Programação Dinâmica.

Independente do estado inicial e do controle nos estágios iniciais deve-se fornecer um controle ótimo com relação ao estado resultante, a partir do controle dos estágios iniciais. De outra forma, qualquer estratégia de controle que é ótima no intervalo de $[j, N]$ é necessariamente ótima $[j + 1, N]$ para $j = 0, 1, 2, \dots, N - 1$.

Considerando o sistema linear em sua forma contínua dada por:

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{2.6}$$

sendo$x(t)$ o vetor de estado $n \times 1$ e $u(t)$ o vetor de controle $p \times 1$. O vetor de controle é dado por:

$$u(t) = u(kT), \tag{2.7}$$

sendo

$$kT \leq t \leq (k + 1)T \tag{2.8}$$

O desafio deste problema é encontrar o controle ótimo $u^*(kT)$, sendo $k = 0, 1, 2, \dots, N - 1$, para que o índice de desempenho dado pela Eq.(2.9) seja minimizado, para um horizonte de tempo $tf = NT$ dado.

$$J = \frac{1}{2}(tf)^T Sx(tf) + \frac{1}{2}\int_0^{tf} [x(t)^T Qx(t) + u(t)^T Ru(t)] \tag{2.9}$$

As matrizes Q e R do índice de desempenho são definida positiva e semi-definida positiva, respectivamente, T é o período de amostragem e N número de amostras. Então a equação de estado discreta é dada por:

$$x[(k+1)T] = A_d x(kT) + B_d u(kT), \tag{2.10}$$

sendo

$$A_d = e^{AT} \tag{2.11}$$

$$B_d = \int_0^T A_d(T-\tau)B d\tau \tag{2.12}$$

O índice de desempenho discretizado em termos dos operadores é dado por:

$$J_{DLQR} = \frac{1}{2}x(NT)^T S x(NT) +$$

$$+\frac{1}{2}\sum_{k=0}^{N-1}\left[x(kT)^T \hat{Q} x(kT) + 2x(kT)^T M(T) u(kT) + u(kT)^T \hat{R} u(kT)\right], \tag{2.13}$$

sendo

$$\hat{Q}(T) = \int_{kT}^{(k+1)T} A_d^T(t-kT) Q A_d(t-kT) dt, \tag{2.14}$$

$$M(T) = \int_{kT}^{(k+1)T} A_d^T(t-kT) Q B_d(t-kT) dt, \tag{2.15}$$

e

$$\hat{R}(T) = \int_{kT}^{(k+1)T} [B_d^T(t-kT) Q B_d(t-kT) + R] dt \tag{2.16}$$

Como ponto de vista de propriedades do índice de desempenho, temos que $\hat{Q}(T)$ e $\hat{R}(T)$ são simétricas, semi-definida positiva e definida positiva respectivamente, e nada pode-se dizer sobre a matriz $M(T)$ que é a matriz de ponderação do acoplamento entre o estado e a entrada. Em geral, o índice de desempenho quadrático é descrito sem a matriz M. É mais natural definir o índice de desempenho quadrático da seguinte forma:

$$J_{DLQR} = \frac{1}{2}x_N^T S x_N + \frac{1}{2}\sum_{k=0}^{N-1}(x_k^T Q x_k + u_k^T R u_k) \quad (2.17)$$

Para o caso de tempo infinito ou número infinito de estágios, $N = \infty$, o índice de desempenho é dado por:

$$J_{DLQR} = \frac{1}{2}\sum_{k=0}^{\infty}(x_k^T \hat{Q} x_k + x_k^T 2M u_k + u_k^T \hat{R} u_k) \quad (2.18)$$

Neste caso o custo final é eliminado, desde que N se aproxima do infinito, o estado final x_N deve aproximar do estado de equilíbrio nulo, de modo que a restrição terminal não é mais necessária. Para o projeto do regulador linear quadrático de tempo infinito, uma exigência é que, em malha fechada, o sistema seja assintoticamente estável. Para isso, o sistema dado pela Eq.(2.1), sendo $f(x_k) = A_d$ e $g(x_k) = B_d$precisa ser controlável e observável.

A solução do regulador infinito pode ser obtida definindo-se $k \to \infty$. Como $N \to \infty$, o ganho da matriz P_k de *Riccati* torna-se constante.

$$\lim_{k\to\infty} P_k = P \quad (2.19)$$

O controle ótimo é:

$$u_k^* = -\left(\hat{R} + B_d^T P B_d\right)^{-1}(B_d^T P A_d + M^T)x_k^* \quad (2.20)$$

Então a matriz de realimentação é uma matriz constante dada por:

$$K = \left(\hat{R} + B_d^T P B_d\right)^{-1}(B_d^T P A_d + M^T) \quad (2.21)$$

O índice de desempenho ótimo para $N = \infty$ através da minimização da Eq.(2.17), é dado por:

$$J_{\infty}^{*} = \frac{1}{2} x_0^T P x_0 \tag{2.22}$$

Para o projeto do regulador linear quadrático de tempo infinito, é necessário que o sistema seja controlável ou estável na realimentação de estados. A controlabilidade é uma exigência mais forte que a estabilidade, já que um sistema não controlável pode ser estabilizado se os estados não controláveis sejam estáveis [6].

O regulador discreto ótimo de tempo finito é dado por:

$$\min J = G(x_N) + \sum_{k=0}^{N-1} F(x_k, u_k) \tag{2.23}$$

sujeito a

$$x_{k+1} = A_d x_k + B_d u_k \tag{2.24}$$

sendo

$$G(x_N) = \frac{1}{2} x_N^T S x_N \tag{2.25}$$

$$\sum_{k=0}^{N-1} F(x_k, u_k) = \frac{1}{2} x_N^T \hat{Q} x_N + x_k^T M u_k + \frac{1}{2} u_k^T \hat{R} u_k \tag{2.26}$$

com o estado inicial dado, ou seja, x_0 é conhecido.

Se assumirmos que $J_{N-j}(x_j)$ seja o índice de desempenho no intervalo de $[j, N]$. Então:

$$J_{N-j}(x_j) = G(x_N) + \sum_{k=0}^{N-1} F_k(x_k, u_k) \tag{2.27}$$

$$J_{N-j}(x_j) = G(x_N) + \sum_{k=0}^{N-1} F_k(x_k, u_k) \tag{2.28}$$

com

$$j = 0, 1, 2, \ldots, N$$

O mínimo valor de $J_{N-j}(x_j)$ é representado por:

$$f_{N-j}(x_j) = \min_{u_i} J_{N-j}(x_j) \tag{2.29}$$

Para $j = N$, a Eq.(2.29) representa o índice de desempenho sobre o estágio 0, Logo:

$$f_0(x_N) = G(x_N) = \frac{1}{2} x_N^T S x_N \tag{2.30}$$

Para $j = N - 1$, tem-se um estágio ou um intervalo processado, que é o último estágio. Então o índice de desempenho ótimo é dado por:

$$f_1(x_{N-1}) = \min_{u_{N-1}} J_1(x_{N-1}) = \min_{u_{N-1}} G(x_N) + F_{N-1}(x_{N-1}, u_{N-1}) \tag{2.31}$$

Sabendo-se que:

$$G(x_N) = \frac{1}{2}(A_d x_{N-1} + B_d u_{N-1})^T S (A_d x_{N-1} + B_d u_{N-1}) \tag{2.32}$$

Considerando a Eq.(2.31), temos:

$$\begin{aligned} f_1(x_{N-1}) = \min_{u_{N-1}} & \left(\frac{1}{2} x_{N-1}^T (\hat{Q} + A_d^T S A_d) x_{N-1} \right) \\ & + \min_{u_{N-1}} \left(x_{N-1}^T \left(M + \frac{1}{2} A_d^T S B_d \right) u_{N-1} \right) \\ & + \min_{u_{N-1}} \left(\frac{1}{2} u_{N-1}^T B_d^T S A_d x_{N-1} \right) \\ & + \min_{u_{N-1}} \left(\frac{1}{2} u_{N-1}^T (\hat{R} + B_d^T S B_d) u_{N-1} \right) \end{aligned} \tag{2.33}$$

Então:

$$f_1(x_{N-1}) = \min_{u_{N-1}} J_1(x_{N-1}) \tag{2.34}$$

Portanto para calcular o mínimo, temos:

$$\frac{\partial J_1(x_{N-1})}{\partial u_{N-1}} = 0 \tag{2.35}$$

O resultado é dado por:

$$\left[\left(M + \frac{1}{2}A_d^T S B_d\right)^T + \frac{1}{2}B_d^T S A_d\right] x_{N-1}^* + \left(\hat{R} + B_d^T S B_d\right) u_{N-1}^* = 0 \tag{2.36}$$

o controle ótimo é dado por:

$$u_{N-1}^* = -\left(\hat{R} + B_d^T S B_d\right)^{-1}\left(M + \frac{1}{2}A_d^T S B_d\right) x_{N-1}^* \tag{2.37}$$

Substituindo a Eq.(2.37) na Eq.(2.33), e simplificando, temos:

$$\begin{gathered} f_1(x_{N-1}) = \frac{1}{2}x_{N-1}^T[\hat{Q} + A_d^T S A_d - \\ (M^T + B_d^T S A_d)^T\left(\hat{R} + B_d^T S B_d\right)^{-1}(M^T + B_d^T S A_d)]x_{N-1} \end{gathered} \tag{2.38}$$

Sabendo que $S = P_N$ (Matriz de *Riccati*):

$$P_{N-1} = \hat{Q} + A_d^T S A_d - (M^T + B_d^T S A_d)^T\left(\hat{R} + B_d^T S B_d\right)^{-1}(M^T + B_d^T S A_d) \tag{2.39}$$

Temos assim, respectivamente:

$$f_0(x_N) = \frac{1}{2}x_N^T P_N x_N \tag{2.40}$$

$$f_1(x_{N-1}) = \frac{1}{2} x_{N-1}^T P_{N-1} x_{N-1} \tag{2.41}$$

Continuando o processo, tem-se para $j = N - 2$, a caracterização de um problema de otimização consistindo nos dois últimos estágios:

$$f_2(x_{N-2}) = \min_{u_{N-2}, u_{N-1}} J_2(x_{N-2}) \tag{2.42}$$

Então

$$f_2(x_{N-2}) = \min_{u_{N-2}, u_{N-1}} F_{N-2}(x_{N-2}, u_{N-2}) + F_{N-1}(x_{N-1}, u_{N-1}) + G(x_N) \tag{2.43}$$

sabendo-se que:

$$f_2(x_{N-2}) = \min_{u_{N-2}} F_{N-2}(x_{N-2}, u_{N-2}) + f_1(x_{N-1}) \tag{2.44}$$

$$F_{N-2}(x_{N-2}, u_{N-2}) = \frac{1}{2} x_{N-2}^T \hat{Q} x_{N-2} + x_{N-2}^T M u_{N-2} + \frac{1}{2} u_{N-2}^T \hat{R} u_{N-2} \tag{2.45}$$

$$f_1(x_{N-1}) = \frac{1}{2} (A_d x_{N-2} + B_d u_{N-2})^T P (A_d x_{N-2} + B_d u_{N-2}) \tag{2.46}$$

Calculando o mínimo:

$$\frac{\partial J_2(x_{N-2})}{\partial u_{N-2}} = 0 \tag{2.47}$$

Logo, pode-se mostrar que o controle ótimo é dado por:

$$u_{N-2}^* = -\left(\hat{R} + B_d^T P_{N-1} B_d\right)^{-1} (M^T + B_d^T P_{N-1} A_d) x_{N-2}^* \tag{2.48}$$

e

$$f_2(x_{N-2}) = \frac{1}{2}x_{N-2}^T[\hat{Q} + A_d^T P_{N-1} A_d - (M^T + B_d^T P_{N-2} A_d)^T \left(\hat{R} + B_d^T P_{N-2} B_d\right)^{-1} (M^T + B_d^T P_{N-1} A_d)]x_{N-2} \tag{2.49}$$

Assumindo:

$$P_{N-2} = \hat{Q} + A_d^T P_{N-1} A_d - (M^T + B_d^T P_{N-1} A_d)^T \left(\hat{R} + B_d^T P_{N-1} B_d\right)^{-1} (M^T + B_d^T P_{N-1} A_d) \tag{2.50}$$

A Eq.(2.49) pode ser simplificada para:

$$f_2(x_{N-2}) = \frac{1}{2}x_{N-2}^T P_{N-2} x_{N-2} \tag{2.51}$$

Generalizando:

$$f_{N-j}(x_j) = \frac{1}{2}x_j^T P_j x_j \tag{2.52}$$

sendo

$$P_j = \hat{Q} + A_d^T P_{j+1} A_d - \left(M^T + B_d^T P_{j+1} A_d\right)^T \left(\hat{R} + B_d^T P_{j+1} B_d\right)^{-1} \left(M^T + B_d^T P_{j+1} A_d\right) \tag{2.53}$$

O controle ótimo é dado por

$$u_j^* = -\left(\hat{R} + B_d^T P_{j+1} B_d\right)^{-1} \left(M^T + B_d^T P_{j+1} A_d\right) x_j^* \tag{2.54}$$

Portanto, têm-se a equação de *Riccati* utilizando o princípio da Otimalidade. Este método de solução também é conhecido como Programação Dinâmica. Para simplificação das equações será considerado que $\hat{Q} = Q$,$\hat{R} = R$, e $M = 0$. Assim, têm-se:

$$P_j = Q + A_d^T P_{j+1} A_d - \left(B_d^T P_{j+1} A_d\right)^T \left(R + B_d^T P_{j+1} B_d\right)^{-1} \left(B_d^T P_{j+1} A_d\right) \quad (2.55)$$

e

$$u_j^* = -\left(R + B_d^T P_{j+1} B_d\right)^{-1} \left(B_d^T P_{j+1} A_d\right) x_j^* \quad (2.56)$$

2.3. Solução recursiva da DARE

A equação algébrica de *Riccati* é resolvida geralmente por métodos de cálculo numérico computacional, recursivos ou métodos de autovalores e autovetores. O cálculo utilizando o método computacional envolve soluções iterativas da equação a diferença não linear de *Riccati*. A Programação Dinâmica (PD) é um método de solução recursivo do problema de controle ótimo. Abaixo temos a matriz de ganho de realimentação do regulador linear discreto ótimo (DLQR):

$$K_j = \left(R + B_d^T P_{j+1} B_d\right)^{-1} \left(B_d^T P_{j+1} A_d\right) \quad (2.57)$$

O controle ótimo é dado por:

$$u_j^* = -K_j x_j^* \quad (2.58)$$

A Eq.(2.55) é simplificada:

$$P_j = Q + A_d^T P_{j+1} A_d - \left(B_d^T P_{j+1} A_d\right)^T K_j \quad (2.59)$$

Partindo da condição limite $P_N = S$ conforme é apresentado nas Eq.(2.30) e Eq.(2.40), as Eq.(2.57) e Eq.(2.59) são facilmente resolvidas pelo método da recursividade [7].

De acordo com as equações vistas anteriormente, podemos desenvolver um algoritmo que mostre a resolução do problema do regulador linear quadrático discreto utilizando-se da Programação Dinâmica:

ALGORITMO I – DLQR UTILIZANDO PD

1 ► **Inicialização**
2 Sistema Dinâmico Discreto
3 A_d, B_d, C_d e D_d
4 Condição Limite
5 $P_N = S$
6 Matrizes de Ponderação
7 QeR
8 Estados Iniciais
9 ► **Processo Iterativo**
10 Para j de N até 1 passo -1
11 ► **Ganho Ótimo de Realimentação**
12 $K_j = \left(R + B_d^T P_{j+1} B_d\right)^{-1}\left(B_d^T P_{j+1} A_d\right)$
13 ► **Recorrência de *Riccati***
14 $P_{j+1} = \left(A_d - B_d K_j\right)^T P_j \left(A_d - B_d K_j\right) + K_j^T R K_j + Q$
15 ► **Controle Ótimo**
16 $x_{j+1} = \left(A_d - B_d K_j\right) x_j$
17 $u_j^* = -K_j x_j$
18 Fim-para
19 ► **Fim do Processo Iterativo**

O algoritmo apresentado anteriormente deixa de maneira clara os passos para obtenção da solução do DLQR utilizando-se a Programação Dinâmica. Deve-se observar a inicialização do algoritmo que leva em consideração a condição limite, que nada mais é que o valor no instante final N assumido pela matriz P de *Riccati*, os estados iniciais, e ainda as matrizes de ponderação Q e R.

2.4. Aprendizagem por Reforço e Controle Adaptativo

A técnica de programação dinâmica é utilizada para otimização de processos de decisão que possuem multiestágios. A condição do processo dentro de cada estágio é denominada de estado. Cada estágio inclui uma tomada de decisão que pode alterar o estado do processo representando uma transição do estado corrente e o estado futuro. Dentro do processo multiestágios, o objetivo dafunção de decisão é encontrar uma política ótima proveniente das decisões.

A política ótima determinada para um processo de decisão de multiestágios está embasada no princípio da Otimalidade de Bellman, que diz que uma política ótima apresenta a propriedade segundo a qual, a respeito das

decisões tomadas para assumir um estado particular num certo estágio, as decisões restantes a partir deste estado devem constituir uma política ótima [7].

A aprendizagem por reforço é um tópico muito importante para a compreensão da programação dinâmica adaptativa. Um sistema típico de aprendizagem por reforço (AR) é constituído basicamente de um agente interagindo em um ambiente via sensores e atuadores, que representam a percepção e a ação, respectivamente. A ação tomada muda de alguma forma o ambiente, afetando o estado na tentativa de alcançar o seu objetivo, e as mudanças são comunicadas ao agente através de um sinal de reforço e o próximo estado.

A principal diferença entre a aprendizagem por reforço e o aprendizado supervisionado está relacionada ao tipo de resposta recebida do meio ambiente. No aprendizado supervisionado, está disponível uma função que conhece a saída correta para cada uma das saídas do agente, e a aprendizagem é baseada nos dados de erro da saída. Na aprendizagem por reforço, não há o conhecimento da saída correta, o agente recebe apenas a informação do ambiente por um reforço, e aplica uma ação, de maneira a aumentar a quantidade de recompensa que ele recebe ao longo do tempo. Um fator interessante em relação ao aprendizado supervisionado, é que no desempenho online, a avaliação do sistema ocorre paralelamente à aprendizagem. A aprendizagem por reforço é aplicada quando não é possível utilizar a aprendizagem supervisionada padrão, e exige menos conhecimento a princípio [29].

O sistema de AR tem um número de ações de saída formando um vetor denominado de vetor de ação ou controle, em que o agente usa para influenciar o ambiente. O problema é baseado em encontrar uma política ótima de controle que maximize o reforço acumulado no tempo.

O agente apresenta variações à medida que vai acumulando experiência a partir das interações com o ambiente. O aprendizado pode ser expresso em termos da convergência até uma política ótima, que conduza a solução do problema de forma ótima. As ações ótimas podem ser baseadas em condições máximas e mínimas.

O reforço é um sinal do tipo escalar, devolvido pelo ambiente ao agente e é emitido assim que uma ação tenha sido efetuada e uma transição de estado

tenha ocorrido. As funções de reforço servem para expressar o objetivo que o agente deve alcançar, assim o agente deve maximizar a quantidade total de reforços recebidos, denominado de reforço acumulado.

Na teoria de aprendizagem por reforço surge um conflito básico denominado de exploração e descoberta. Esse conflito ocorre em cada estado e na escolha pelo agente. No primeiro caso, o agente escolhe a ação para a qual o reforço esperado é de boa qualidade, e no segundo caso, o agente escolhe a ação em que a qualidade pode ser não tão boa, mas as aplicações podem direcionar para zonas promissoras que não foram exploradas.

2.4.1. Função Valor

A função valor é considerada como o mapeamento do estado, ou par estado-ação, em um valor que é obtido a partir do reforço atual e dos reforços futuros. A função valor considera apenas o estado x_k e é denotada por $V(x_k)$ e denominada de função valor-estado. A função que considera o par estado-ação (x_k, u_k) é denotada por $Q(x_k, u_k)$, e denominada função valor-ação $(Q - Function)$.

Para todo estado x_k, um agente escolhe uma ação de acordo com uma determinada política, $h(x_k) = u_k$. O valor do estado x_k sobre uma política $h(x_k)$ é dado por:

$$V_h(x_k) = E\left\{\langle \sum_{i=k}^{\infty} \gamma^{i-k} r\big(x_i, h(x_i)\big) \mid x_0 = x \rangle\right\} \quad \forall x \in X \tag{2.60}$$

A Eq.(2.60) representa a expectativa de reforço quando se aplica a ação proposta pela política pela política h no estado x_i. Adotando-se $E\{r(x_k, h(x_k))\} = R(x_k, u_k)$, temos:

$$V_h(x_k) = R\big(x_k, h(x_k)\big) + \gamma \sum_{y \in X} p_{xy}(h(x_k)) V_h(y) \tag{2.61}$$

Sabe-se que existe uma política ótima, $h^*(x_k)$ que define:

$$V_h^*(x_k) \geq V_h(x_k) \;\; \forall x \;\in X, \forall h \tag{2.62}$$

Logo o valor ótimo é dado por:

$$V_h^*(x_k) = \max_h E\left\{\langle \sum_{i=k}^{\infty} \gamma^{i-k} r_i \rangle\right\} \tag{2.63}$$

A função $V_h^*(x_k)$ satisfaz, no caso do horizonte infinito, a Eq.(2.64), conhecida como equação de Otimalidade de *Bellman* (Equação da Programação Dinâmica).

$$V_h^*(x_k) = \max_{u \in U_x}\left\{R(x_k, u_k) + \gamma \sum_y p_{xy}(u_k) V_h^*(y)\right\}, \qquad \forall\, x \;\in X \tag{2.64}$$

Considerando a Eq.(2.63), temos a seguinte política ótima $h^*(x_k)$ é dada por:

$$h^*(x_k) = \operatorname{argmax}_{u \in U_x}\left\{R(x_k, u_k) + \gamma \sum_y p_{xy}(u_k) V_h^*(y)\right\} \tag{2.65}$$

A programação dinâmica oferece um conjunto de métodos para resolver o problema de otimização, considerando a propriedade dos processos *markovianos*. Os dois principais métodos considerados na resolução dos problemas de otimização são chamados de política de iteração e valor de iteração.

2.4.2. Aprendizagem por Ator-Crítico

A Aprendizagem por Reforço está fortemente ligada do ponto de vista teórico ao controle adaptativo. Uma classe de AR é baseada na estrutura do Ator-Crítico, onde um componente Ator aplica uma política de ação ou controle para o

ambiente, e um componente Crítico avalia o valor dessa ação. De acordo com esta avaliação do valor, vários esquemas podem ser usados para modificar ou melhorar a ação no sentido de que a nova política possui um valor que é melhor em relação ao valor anterior [33].

Na figura 1, temos a estrutura do Ator-Crítico que é dada em duas etapas: avaliação de política realizada pelo crítico seguida da etapa de melhoria de política. A etapa de avaliação de política é feita através da observação do ambiente, dos resultados e das ações em curso. É de maior interesse que os sistemas de Aprendizagem por Reforço que utilizam como estrutura Ator-Crítico, em que o Crítico avalia o valor das atuais políticas seja baseado em algum tipo de critério de otimalidade.

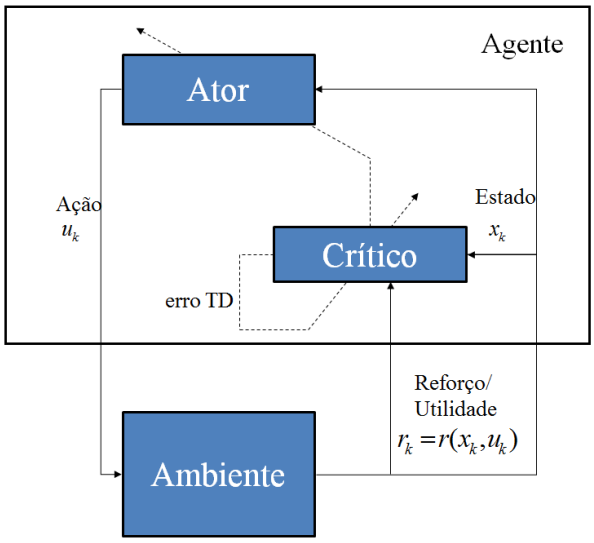

FIGURA 1 – Estrutura de AR por Ator-Crítico

Os métodos Ator-Crítico são uma forma de aprendizagem via diferenças temporais (TD) que possui uma estrutura de memória separada para representar a política de forma independente da função valor. Na aprendizagem TD, o conhecimento prévio do agente é utilizado para determinar qual decisão ele deve tomar. A atualização da função valor pode ocorrer a cada instante de tempo, não necessitando de uma estimativa confiável da função reforço. O algoritmo utilizado para atualizar $V(x_k)$ num dado instante k a uma ação, $h(x_k) = u_k$que faz passar do estado x_k para x_{k+1} para um reforço imediato $r_k = (x_k, u_k)$ é dada por:

$$V_h(x_k) = V_h(x_k) + \in [r_k + \gamma V_h(x_{k+1}) - V_h(x_k)] \quad (2.66)$$

sendo$\in$ é a taxa de aprendizagem. Expandindo-se a Eq.(2.59) temos:

$$V_h(x_k) = E\left\{r\left(x_k, h(x_k)\right) + \gamma \langle \sum_{i=k+1}^{\infty} \gamma^{i-(k+1)} r\left(x_i, h(x_i)\right) \mid x_0 = x\rangle\right\} \quad (2.67)$$

Logo:

$$V_h(x_k) = E\{r\left(x_k, h(x_k)\right) + \gamma V_h(x_{k+1}) - V_h(x_k)\} \quad (2.68)$$

Ao atualizar a função valor, é feita uma aproximação da equação anterior por utilizar V_k e não V_h real. Esta aproximação é utilizada como alvo na aprendizagem por diferença temporal (TD), de forma que V_k é atualizado a partir da sua diferença com a aproximação de V_h. O crítico assume a forma de um erro TD, sendo calculado de acordo com a Eq.(2.66). Logo o erro TD é dado por

$$e_k = r\left(x_k, h(x_k)\right) + \gamma V_h(x_{k+1}) - V_h(x_k) \quad (2.69)$$

A regra de aprendizado é dada por

$$V_h(x_k) = V_h(x_k) + \in e_k \quad (2.70)$$

Esta regra é denominada de regra de aprendizagem via diferença temporal. Pode ser considerada como um erro de previsão entre as estimativas $V(x_k)$ e $V(x_{k+1})$.

A função $V_h(x_k)$ representa a função valor atual considerada pelo Crítico. Após cada decisão tomada pelo Ator, o Crítico avalia o novo estado do ambiente, de forma a determinar se a decisão tomada foi adequada ou não. Para cada política, o Crítico manterá funções de valor diferentes, de forma a manter coerência e permitir que políticas diferentes possam ser utilizadas para diferentes

estados do ambiente. Após cada iteração, o Crítico atualiza a função valor de acordo com a Eq.(2.66).

2.5. Parametrizações do DLQR

O regulador linear quadrático discreto (DLQR) é caracterizado como um PDM (Processo de Decisão *Markoviano*), em que o modelo do sistema e a política de controle são mapeamentos lineares e são representados por combinadores lineares do estado dos estados e das saídas do sistema. As parametrizações dos estados $f(x_k, u_k)$ e da política de decisão $h(x_k)$ são dadas por

$$f(x_k, u_k) = A_d x_k + B_d u_k \tag{2.71}$$

$$h(x_k) = -K x_k \tag{2.72}$$

Sendo $A_d \in \Re^{nxn}$, n é a ordem do sistema, $B_d \in \Re^{n\,x\,n_e}$, n_e é a quantidade de entradas do sistema e $K \in \Re^{n_e\,x\,n}$ é a matriz de ganho de realimentação. Assume-se que (A, B) é estabilizável, ou seja, existe uma matriz K que garanta que o sistema de malha fechada seja assintoticamente estável. O sistema de malha fechada é dado por

$$x_{k+1} = (A - BK)x_k \tag{2.73}$$

A função de utilidade r associada a este sistema é uma forma quadrática, e pode ser representada pela equação abaixo:

$$r(x_k, u_k) = x_k^T Q x_k + u_k^T R u_k, \tag{2.74}$$

com as matrizes de ponderação $Q \in \Re^{n\,x\,n} \geq 0$ e $R \in \Re^{n_e\,x\,n_e} > 0$ simétricas.

O objetivo do DLQR é selecionar a política de controle K que minimiza a seguinte função de custo:

$$V(x_k) = \gamma^{i-k} \sum_{i=k}^{\infty} (x_i^T Q x_i + u_i^T R u_i) \tag{2.75}$$

$$V(x_k) = \gamma^{i-k} \sum_{i=k}^{\infty} x_i^T (Q + K^T R K) x_i, \quad \forall x_k \in X \tag{2.76}$$

No caso do LQR (*Linear QuadraticRegulator*), o valor ótimo da função de custo assume a seguinte forma quadrática [7]:

$$V(x_k) = x_k^T P x_k \tag{2.77}$$

para alguma matriz $P \in \Re^{n\,x\,n} > 0$ simétrica. Portanto a Equação de *Bellman* para o LQR discreto é

$$x_k^T P x_k = x_k^T Q x_k + u_k^T R u_k + \gamma (x_{k+1}^T P x_{k+1}) \tag{2.78}$$

Em termos de ganho de realimentação é dada por

$$x_k^T P x_k = x_k^T [Q + K^T R K + \gamma (A - BK)^T P (A - BK)] x_k \tag{2.79}$$

A Eq.(2.79) é satisfeita para todos os estados x_k se

$$\gamma (A - BK)^T P (A - BK) - P + Q + K^T R K = 0 \tag{2.80}$$

A Eq.(2.80), quando o ganho K é fixado, é conhecida como equação de *Lyapunov*.Dado um ganho K estabilizante, a solução desta equação fornece $P = P^T > 0$, tal que $V(x_k) = x_k^T P x_k$ é o custo usando a política K, ou seja:

$$V(x_k) = \sum_{i=k}^{\infty} \gamma^{i-k} x_i^T (Q + K^T R K) x_i = x_k^T P x_k \tag{2.81}$$

Portanto é definida a função valor em termos da política de controle K obtendo a matriz P de *Riccati*.

CAPÍTULO 3

PROGRAMAÇÃO DINÂMICA ADAPTATIVA

Na elaboração do projeto de sistemas de controle com realimentação, são necessários algoritmos e técnicas de análise que possam garantir um bom desempenhoe margens de segurança aceitáveis. Controle com realimentação com pouca estabilidade, sensibilidade para perturbações ou robustez não garantida, são métricas que são utilizadas pela indústria para avaliar o desempenho do sistema. A forma padrão para garantir critérios de qualidade aceitos no meio industrial é utilizar a estrutura e as ferramentas fornecidas pela matemática.

A programação dinâmica é uma técnica utilizada para a otimização de processos de decisão multiestágios. A condição do processo dentro de cada estágio é denominada de estado. Cada estágio inclui uma tomada de decisão que pode alterar o estado do processo representando uma transição do estado corrente e o estado futuro. Dentro do processo multiestágios, o objetivo do tomador de decisão é encontrar uma política ótima proveniente das decisões.

O controle adaptativo pode ser realizado da forma direta, em que os parâmetros do controlador são estimados diretamente, ou na forma indireta, em que os parâmetros do modelo do sistema são estimados primeiro e depois os ganhos do controlador são obtidos. A Programação Dinâmica como solução do controle ótimo, é um método de recorrência no tempo, sendo utilizado para planejamento *off-line*, ou seja, não estimação dos parâmetros em tempo real. A Equação de *Bellman* leva a vários métodos iterativos para aprendizado da solução do controle ótimo sem ter que resolver a equação de *Hamilton-Jacobi-Bellman* (HJB).

3.1. Processo de Decisão *Markoviano*

Na aprendizagem por reforço (AR), um agente é caracterizado como um processo de decisão *markoviano*, onde suas transições entre os estados são dadas de forma probabilísticas. Cada ação tem uma recompensa que depende do estado em que o processo se encontra. O processo de decisão *markoviano* é caracterizado por quatro componentes: estados, ações e distribuições de transição e recompensa [34].

Um processo de decisão *markoviano* determinístico é definido pelo espaço de estados X do processo, o espaço de ação U do controlador, a função de transição f do processo, que descreve como os estados mudam a partir das ações de controle, e a função de reforço r, que avalia o desempenho do controle imediato.

Assim, temos a função de transição $f: X \times U \rightarrow X$ dada por:

$$x_{k+1} = f(x_k, u_k) \tag{3.1}$$

O controlador recebe um sinal de reforço $r: X \times U \rightarrow \Re$ dado por:

$$r_k = r(x_k, u_k) \tag{3.2}$$

Considerando X como um processo estocástico com um número finito de valores, onde: $X = \{x_0, x_1, \ldots, x_k\}$.

As transições do estado x ao estado y em um processo de decisão *markoviano* dependem apenas das ações permitidas ao estado x. As probabilidades de transição são denotadas por:

$$p_{xy}(u) = P_r\left(x_{k+1} = x_y \mid x_k = x_k, u_k = u \right) \tag{3.3}$$

SendoP_r o operador de probabilidade do estado x_k passar para o estado x_{k+1}, quando a ação u_k for tomada no tempo k.

O processo de decisão markoviano pode ser caracterizado como:

- Ambiente que evolui probabilisticamente de acordo com um conjunto finito e discreto de estados;
- Para cada estado do ambiente, existe um conjunto finito de ações possíveis;
- A cada transição o agente recebe um retorno positivo ou negativo do ambiente em relação a ação tomada;
- Estados são observados, ações são executadas e reforços são relacionados.

3.2. Avaliação de Política Aproximada via Diferenças Temporais

Diversas contribuições da aprendizagem por reforço são feitas por meio da aproximação da função valor. Na aprendizagem da função valor por métodos de AR, é necessário armazenar o valor ótimo e o controle ótimo em função do vetor de estado $x \in \Re^n$. No processo de decisão *markoviano*, que são em sistemas de estados discretos, o estado pode assumir apenas um número finito de valores discretos prescritos, o que nos leva a chamada complexidade computacional, mais conhecido por meio de *Bellman* por *"Maldição da Dimensionalidade"*, portanto, com o aumento dos estados, mais informações devem ser armazenadas. Utilizando os aproximadores da função valor, onde o crítico e, se desejar, o ator são parametrizados usando aproximadores de função, este problema é atenuado [35].

O conceito chave para a implementação de controladores ótimos online em avanço no tempo é a diferença temporal do erro, que é definida em termos da equação de *Bellman* como:

$$e_k = H(x_k, h(x_k), \Delta V_k) = r\big(x_k, h(x_k)\big) + \gamma V(x_{k+1}) - V(x_k) \tag{3.4}$$

Percebe-se que o lado direito da equação nada mais é que o Hamiltoniano. A função e_k é conhecida como diferença temporal do erro. A intenção é levar a solução do Hamiltoniano de tal forma que $e_k = 0$ a cada intervalo de tempo k para a função valor $V(.)$.

$$0 = H(x_k, h(x_k), \Delta V_k) = r\big(x_k, h(x_k)\big) + \gamma V(x_{k+1}) - V(x_k) \tag{3.5}$$

A diferença temporal do erro pode ser considerada como, a previsão do erro entre o desempenho previsto e o desempenho observado em resposta a uma ação aplicada ao sistema. A Eq.(3.5) é um elemento muito importante na resolução online da equação não linear de *Lyapunov* utilizando-se somente dados medidos ao longo da trajetória do sistema. Para sistemas não lineares, a diferença temporal é de difícil solução.

Uma solução prática para resolução da diferença temporal, é fazendo a aproximação da função valor $V(.)$, utilizando-se aproximação paramétrica chamada de programação dinâmica aproximada (ou adaptativa). Uma aproximação da função valor para o caso do LQR discreto pode ser considerada.

Sabemos que $u_k = -Kx_k$ é quadrática nos estados, ou seja, vale para a matriz P. Logo para o DLQR, a diferença temporal do erro é dada da seguinte forma:

$$e_k = x_k^T Q x_k + u_k^T R u_k + \gamma(x_{k+1}^T P x_{k+1}) - x_k^T P x_k \tag{3.6}$$

Para simplificarmos a Eq.(3.6), utilizamos o produto de *Kronecker*:

$$V(x_k) = x_k^T P x_k = vec(x_k^T P x_k) = (x_k \otimes x_k)\big(vec(P)\big)^T = \bar{p}^T \bar{x}_k \tag{3.7}$$

sendo$\otimes$ o produto de *Kroneckere* $vec(P)$ o vetor formado pelos elementos da matriz P em um vetor coluna [36]. O produto de Kronecker dado por $\bar{x}_k = x_k \otimes x_k$, é um vetor quadrático contendo todos os produtos possíveis de n componentes de x_k. Vale ressaltar que a matriz P é simétrica, então tem-se somente $n(n+1)/2$ elementos independentes, assim, pode-se remover os elementos redundantes em $\bar{x}_k = x_k \otimes x_k$ para definir um conjunto de base quadrática, $\bar{x}_k$, com $n(n+1)/2$ elementos independentes. O vetor de parâmetros desconhecidos é $\bar{p} = vec(P)$constituídos pelos elementos da matriz P. Assim, o erro da diferença temporal passa a ser escrito como:

$$e_k = x_k^T Q x_k + u_k^T R u_k + \gamma(\bar{p}^T \bar{x}_{k+1}) - \bar{p}^T \bar{x}_k \tag{3.8}$$

$$e_k = r(x_k, u_k) + \gamma(\bar{p}^T \bar{x}_{k+1}) - \bar{p}^T \bar{x}_k \tag{3.9}$$

Para o regulador quadrático linear discreto (DLQR), um conjunto base completo para a estimação da função valor $V(x_k)$ é fornecida pela função quadrática dos componentes de x_k.

De acordo com *Bellman*, se as aproximações de valor fossem exatas, a diferença temporal seria zero. Assim, os valores das diferenças temporais podem ser usados para fazer ajustes incrementais de $\bar{p}^T$ (também chamado de θ) de forma a produzir uma boa aproximação entre a estimativa atual $\bar{p}^T \bar{x}_k$ de valor e a estimativa atualizada $r(x_k, u_k) + \gamma(\bar{p}^T \bar{x}_{k+1})$ no estado x_k ao longo das trajetórias amostradas.

3.3. Algoritmo da Iteração de Política Aproximada

A iteração de política é um método que busca a política ótima h^* usando o seguinte esquema:

1. Deve-se escolher uma política inicial admissível (h_0);
2. Determinar a função de valor-estado V associada à política atual h_j de acordo com a Eq.(3.10) (Equação de *Bellman*).

$$V(x_k) = r\big(x_k, h(x_k)\big) + \gamma V(x_{k+1}) \tag{3.10}$$

3. Definir a política melhorada h_{j+1} utilizando a Eq.(3.11).

$$h_{j+1}(x_k) = \underset{u(.)}{\operatorname{argmax}}\{r(x_k, u_k) + \gamma V(x_{k+1})\} \tag{3.11}$$

Este procedimento é descrito em [7] que prova que a nova política h_{j+1} é melhorada com respeito à h_j no sentido de que $V^{h_{j+1}}(x_k) \geq V^{h_j}(x_k)$. O método

requer um número de ciclos de avaliação e melhoria de política até atingir $h_j = h_{j+1}$, caso em que a sequência $\{h_j\}$ convergiu para h^*.

A iteração de política baseia-se sobre uma representação tabular da função valor que fornece a solução exata da equação de *Bellman*. Tais representações exatas são inviáveis para processos de decisão *markovianos* cujos espaços de estados e ações são grandes. Neste caso, considera-se uma forma aproximada de iteração de política em que é determinado, de maneira iterativa, uma função de valor aproximada $\hat{V}(x_k)$ da política atual h_j por meio da equação de atualização de valor de política dada pela Eq.(3.11). Em seguida, esta atualização é utilizada para determinar uma nova política gulosa h_{j+1}dada por:

$$h_{j+1}(x_k) = \underset{u(.)}{\operatorname{argmax}}\{r(x_k, u_k) + \gamma \hat{V}(x_{k+1})\} \tag{3.12}$$

A partir dessas equações mostradas anteriormente, desenvolve-se o algoritmo de iteração de política aproximada para funções de valor-estado.

ALGORITMO II– ITERAÇÃO DE POLÍTICA APROXIMADA

► **Inicialização**
Sistema Dinâmico Discreto e Matrizes de Ponderação
A_d, B_d, C_d,D_d, Q e R.
Selecionar qualquer política de controle admissível h_0
Selecionar o fator de desconto, $0 < \gamma \leq 1$.
$j \leftarrow 0$
► **Processo Iterativo**
Para $j = j + 1$
    ► **Etapa de avaliação de política aproximada**
    Função de Valor aproximada $\hat{V}^{hj}$ – Atualização de valor de política
    ► **Etapa de melhoria de política**
    Determinar uma política gulosa h_{j+1} utilizando a Eq.(3.12)
    Se $h_j = h_{j+1}$
► **Fim do Processo Iterativo**
Fim-para

Pode-se observar por meio destes passos que o projeto de controladores ótimos é quase universalmente um processo *offline,* envolvendo soluções de equações de *Riccati* em que deve-se ter o conhecimento da dinâmica do sistema, ou seja as matrizes A_d e $B_d$devem ser conhecidas. Métodos iterativos, incluindo a iteração de política aproximada apresentada nesta seção, podem ser utilizados para obter a solução da equação algébrica de *Riccati*. A idéia é aproximar soluções da equação de *Lyapunov* e utilizar procedimento de melhoria de política para determinação da política de controle ótima do regulador quadrático linear (DLQR).

3.3.1. Iteração de Política Aproximada para o DLQR

As equações para o desenvolvimento do algoritmo de iteração de política aproximada para o DLQR são baseadas na aproximação de P da solução da Eq.(3.13) de Lyapunov associada à política K atual e no esquema de iteração de política aproximada da seção 3.3. O algoritmo tem como etapa a melhoria de política dada pela Eq.(3.14).

$$\gamma(A - BK)^T P(A - BK) - P + Q + K^T RK = 0 \tag{3.13}$$

$$K = -\gamma(R + \gamma B^T PB)^{-1} B^T PA \tag{3.14}$$

Um fato interessante em relação as equações que deve ser destacado é que a matriz P é calculada no tempo k e a matriz de ganho K é atualizada no tempo $k + 1$, isto pode ficar mais claro no algoritmo mostrado a seguir.

ALGORITMO III– ITERAÇÃO DE POLÍTICA APROXIMADA PARA O DLQR

1 ► **Inicialização**
2 Sistema Dinâmico Discreto e Matrizes de Ponderação
3 A_d, B_d, C_d,D_d, Q e R.
4 Selecionar qualquer política de controle admissível K_0
5 Selecionar o fator de desconto, $0 < \gamma \leq 1$.
6 $j \leftarrow 0$

7 ► **Processo Iterativo**

8 Para $j = j + 1$

9 ► **Etapa de avaliação de política aproximada**

10 Determinar uma solução aproximada $\hat{P}$ da equação de *Lyapunov*

11 $\gamma\left(A - BK_j\right)^T P\left(A - BK_j\right) - P + Q + {K_j}^T R K_j = 0$

12 ► **Etapa de melhoria de política**

13 $K_{j+1} = -\gamma\left(R + \gamma B^T P B\right)^{-1} B^T P A$

14 Se $K_j = K_{j+1}$

15 ► **Fim do Processo Iterativo**

16 Fim-para

3.4. Método de Sintonia Heurística QR

Os autovalores são utilizados para verificar se as especificações de projeto estão sendo contempladas durante a operação do sistema. Estabelece-se uma relação entre as figuras de mérito do sistema dinâmico em função das matrizes de ponderação. A nova lei de controle em função das matrizes de ponderação são dadas por:

$$u_k(QR) = -K_{QR} x_k \tag{3.15}$$

sendoK_{QR} o ganho do controlador que depende diretamente da seleção das matrizes de ponderação. O método de sintonia heurística baseia-se nas relações entre as matrizes Q e R quando a recorrência de *Riccati* da Eq.(3.16) é substituída na Eq.(3.17) do ganho.

$$P = Q + A_d^T P A_d - A_d^T P B_d K_{ric} \tag{3.16}$$

$$K_{ric} = \left(R + B_d^T P B_d\right)^{-1} B_d^T P A_d \tag{3.17}$$

Desta forma tem-se o ganho ótimo que é dado por

$$K_{ric} = (R + B_d^T P B_d)^{-1} B_d^T (Q + A_d^T P A_d - A_d^T P B_d K_{ric}) A_d \tag{3.18}$$

$$\begin{aligned} K(Q,R) = \{&[(R + B_d^T P B_d)^{-1} B_d^T (Q + A_d^T P A_d)] \\ &- [(R + B_d^T P B_d)^{-1} B_d^T (A_d^T P B_d K_{ric})]\} A_d \end{aligned} \tag{3.19}$$

$$K(Q,R) = K_{f1}(QR) + K_{f2}(R) \tag{3.20}$$

sendo:

$$K_{f1}(QR) = [(R + B_d^T P B_d)^{-1} B_d^T (Q + A_d^T P A_d) A_d] \tag{3.21}$$

$$K_{f2}(R) = -[(R + B_d^T P B_d)^{-1} B_d^T (A_d^T P B_d K_{ric}) A_d] \tag{3.22}$$

Considerando as situações que $R \gg B_d^T P B_d$ e $Q \gg A_d^T P A_d$ em que as formas quadráticas da entrada e do estado são bem menores que as ponderações Q e R, tem-se que:

$$K_{f1}(QR) \approx \{[R^{-1} B_d^T Q] A_d\} \tag{3.23}$$

$$K_{f2}(R) \approx -\{[R^{-1} B_d^T A_d^T P B_d R^{-1} B_d^T P A_d] A_d\} \tag{3.24}$$

Observa-se que se $R \gg 0$ para o caso de matrizes diagonais, tem-se que $K_{f2}(R) \approx 0$ [31].

A análise da convergência QR consiste na avaliação dos autovalores das matrizes Q e R e suas relações com a alocação de autovalores de sistemas MIMO no plano Z por meio dos controladores ótimos.

Os resultados são apresentados na forma de uma tabela montada seguindo uma heurística que é estabelecida a partir da Eq.(3.23) e Eq.(3.24). O processo iterativo para variações sistemáticas nas matrizes Q e R da função de custo seguem um padrão de crescimento da matriz Q, enquanto que a matriz R é uma matriz identidade durante todo o processo de solução.

3.5. Estruturas Básicas de Programação Dinâmica Adaptativa

Os algoritmos de aprendizagem por reforço utilizando a estrutura Ator-Crítico, podem ser utilizados para determinação de uma lei de controle ótimo para um processo dinâmico tanto *online* quanto *offline*. Na implementação*offline*, é necessário que a dinâmica do sistema seja controlada e os autovalores dos estados são obtidos através de simulações. O algoritmo gera uma ação de controle u_k e uma função valor V_j, para todos os estados, que se repetem para $j = 0,1,2,\ldots$, até que o algoritmo produza valores ótimos de $h^*(x_k)$ e $V^*(x_k)$.

Na implementação*online*, o processo computacional fica dependente somente da variável de estado x_k, não envolvendo o conhecimento da condição final do processo. Os vetores de estado, tornam-se disponíveis progressivamente no tempo $\{x_k \mid k = 0,1,2,\ldots\}$. Assumindo-se que a planta é totalmente observável, o valor do atual estado x_k, é determinado a partir de medições de saída disponíveis. O próximo estado x_{k+1}, é previsto através do modelo da Eq.(2.1). A lei de controle ótima pode ser determinada *online* para sistemas em que suas características são expostas somente durante a operação [38].

No início dos anos 90 foi proposta por Werbos uma família de estruturas de programação dinâmica aproximada, sendo utilizada amplamente [39][40]. A formulação original é baseada em uma implementação utilizando-se Redes Neurais, porém qualquer estrutura de aprendizagem pode ser utilizada. As quatro principais estruturas de ADP propostas por Werbos são: *HeuristicDynamicProgramming* (Programação Dinâmica Heurística - HDP), *Dual HeuristicDynamicProgramming* (Programação Dinâmica Heurística Dual – DHP), *ActionDependentHeuristicDynamicProgramming* (Programação Dinâmica Heurística Dependente de Ação – ADHDP, também conhecida como *Q-Learning* ou Aprendizado-Q) e *ActionDependent Dual HeuristicDynamicProgramming* (Programação Dinâmica Heurística Dual Dependente de Ação – ADDHP) [41][42][43][44].

A figura seguinte mostra as características de cada tipo de categoria da Programação Dinâmica Adaptativa (ADP).

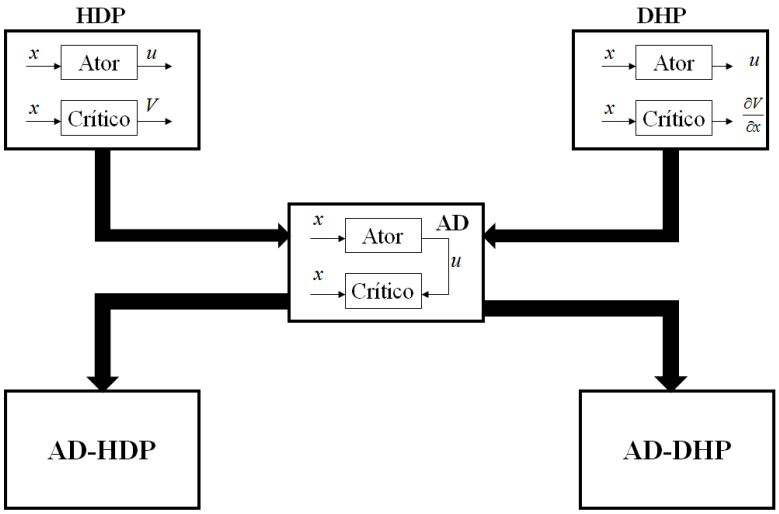

FIGURA 2 – Modelos de ADP propostos por Werbos.

Algumas características importantes devem ser notadas nas diferentes estruturas de ADP. A entrada do crítico recebe a informação do estado do sistema (e do modelo de referência da planta, se for o caso). Na estrutura Dependente de Ação (ActionDependent), o Crítico também promove as saídas do controlador. Na estrutura HDP, a saída do Crítico promove uma aproximação da função valor $V(x_k)$. Na estrutura DHP, tem-se uma aproximação do gradiente da função valor, denotado por:

$$\nabla V(x_k) = \frac{\partial V(x_k)}{\partial x_k} \tag{3.25}$$

Existem algumas formulações em que se necessita de um ciclo de treinamento do sistema, porém as categorias citadas necessitam de dois ciclos de treinamento, um para o Crítico e outro para a ação de controle. Dependendo da estrutura da ADP, um ou ambos os ciclos de treinamento, irão necessitar do modelo dinâmico do sistema.

Os componentes básicos no processo de programação dinâmica aproximada são ação/controlador e o modelo do sistema dinâmico a ser controlado. O controlador recebe as informações do estado x_k corrente da planta e tem como saída, a ação de controle u_k. A planta recebe a ação de controle u_k, e direciona-se para o próximo estado x_{k+1}. Os dados de x_k são passados para o

Crítico e para a função de utilidade $r(x_k, u_k)$. Todos esses dados são necessários para o cálculo de treinamento do Ator e Crítico. O treinamento é baseado na Equação de *Bellman* (Eq.(3.10)).

3.5.1. Programação Dinâmica Heurística (HDP)

A Programação Dinâmica Heurística é a estrutura mais básica da ADP e amplamente aplicada. Na figura abaixo podemos observar a estrutura HDP.

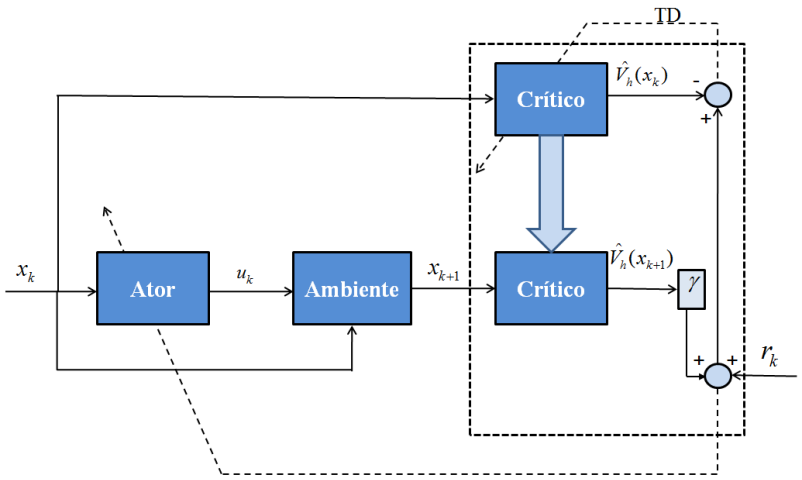

FIGURA 3 – Estrutura HDP.

De acordo com a figura podemos observar que:

- O crítico estima a função valor $V(x_k)$ baseado diretamente no estado x_k da planta.
- O crítico não necessita do modelo da planta para o cálculo.
- O treinamento do controlador necessita encontrar derivadas em cada instante k.

Portanto, o algoritmo HDP utiliza o modelo da planta somente para a atualização do controlador [39].

3.5.2. Programação Heurística Dual (DHP)

Neste caso, o Crítico estima a função valor $V(x_k)$ baseado no co-estado, ou seja, estima diretamente as derivadas de $V(x_k)$ em relação aos estados da planta de acordo com a Eq.(3.25). Na figura abaixo podemos observar a estrutura DHP.

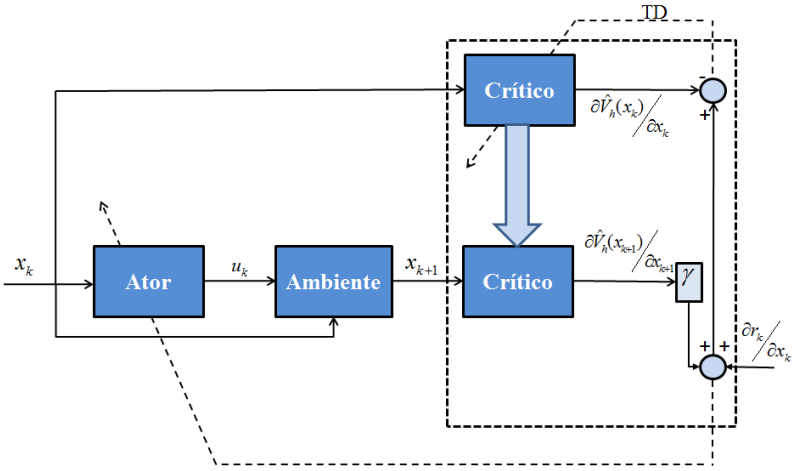

FIGURA 4 – Estrutura DHP.

Para a execução do algoritmo, é necessário encontrar a equação de ponto fixo para o co-estado. Então temos as seguintes equações:

$$\frac{\partial V(x_k)}{\partial x_k} = \frac{\partial r(x_k, u_k)}{\partial x_k} + \gamma \frac{\partial V(x_{k+1})}{\partial x_k} \tag{3.26}$$

ou

$$\nabla V(x_k) = \frac{\partial r(x_k, u_k)}{\partial x_k} + \frac{\partial r(x_k, u_k)}{\partial h(x_k)} \frac{\partial h(x_k)}{\partial x_k} + \gamma \left(\frac{\partial x_{k+1}}{\partial x_k} + \frac{\partial x_{k+1}}{\partial h(x_k)} \frac{\partial h(x_k)}{\partial x_k} \right) \nabla V(x_{k+1}) \tag{3.27}$$

Infelizmente, para a estrutura DHP, para estimar os valores do lado direito da Eq.(3.27), necessita-se do conhecimento da dinâmica do sistema, uma vez que:

$$\frac{\partial x_{k+1}}{\partial x_k} = f(x_k) \tag{3.28}$$

e

$$\frac{\partial x_{k+1}}{\partial x_k} = g(x_k) \tag{3.29}$$

Isto requer a implementação do método dos mínimos quadrados recursivos (*RLS*) para um n-ésimo vetor, sendo calculado computacionalmente intensivo. A ação de controle u_k é determinada da mesma forma que na HDP, porém como o crítico necessita do conhecimento da dinâmica do sistema, então, o controlador será dependente do conhecimento prévio do sistema a ser controlado [39].

3.5.3. Programação Dinâmica Heurística Dependente da Ação (ADHDP)

No algoritmo ADHDP, é utilizado tanto o estado x_k quanto o controle u_k como entradas para o crítico $Q(x_k, u_k)$, também denominado de função-Q. Na figura abaixo é representado o modelo da estrutura ADHDP da Programação Dinâmica Adaptativa.

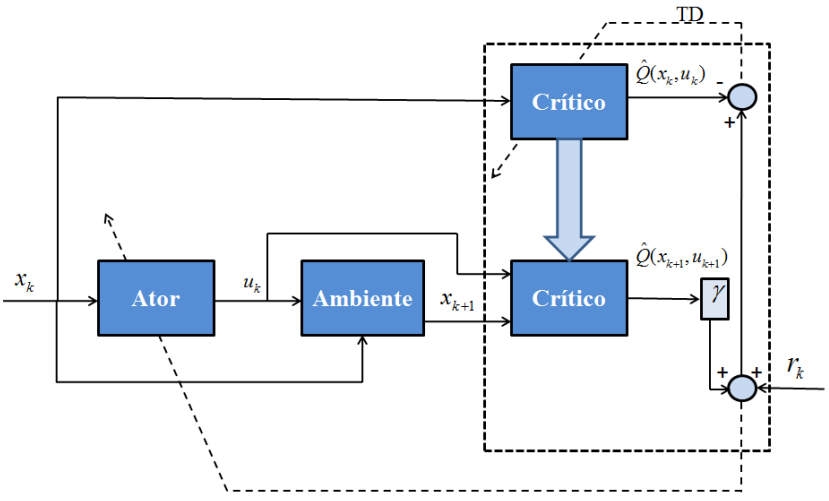

FIGURA 5 – Estrutura ADHDP.

O modo de operação é similar ao do algoritmo da programação dinâmica heurística (HDP). O controle é obtido através da derivada da função Q em relação à entrada de controle:

$$\frac{\partial Q(x_k, u_k)}{\partial u_k}$$

O algoritmo ADHDP, não necessita do conhecimento da dinâmica do sistema para treinamento do processo [45].

3.5.4. Programação Herística Dual Dependente da Ação (ADDHP)

No algoritmo que representa a estrutura ADDHP utiliza-se tanto o estado x_k quanto o controle u_k como entrada do Crítico, e como saída tem-se, o gradiente de $V(x_k)$ em relação aos estados e ao controle mostrados abaixo:

$$\frac{\partial V(x_k)}{\partial x_k}$$

$$\frac{\partial u_k}{\partial x_k}$$

Na figura a seguir, é representada a estrutura da ADDHP.

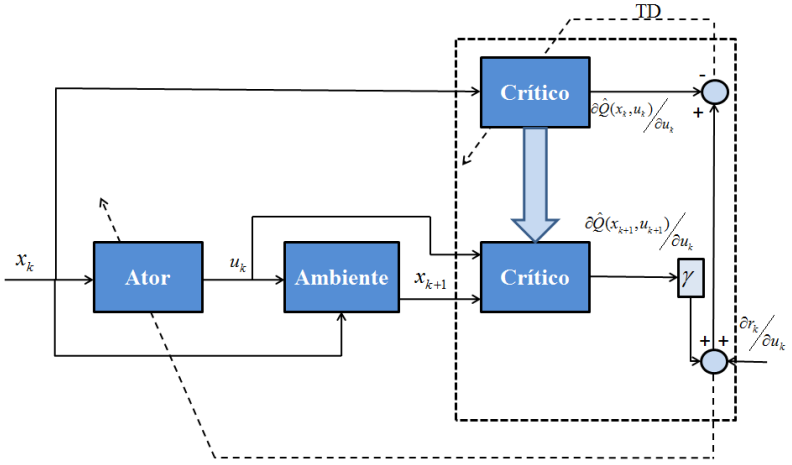

FIGURA 6 – Estrutura ADDHP.

O método ADDHP utiliza o mesmo treinamento do Crítico DHP, porém recebe derivadas necessárias para o treinamento do controlador diretamente da saída do Crítico. Por isso, a estrutura ADDHP necessita do modelo da planta para o treinamento do Crítico, mas para o treinamento do controlador não é necessário o conhecimento da dinâmica do sistema.

Os resultados das análises feitas para os diferentes tipos de estruturas da programação dinâmica aproximada que foram citadas anteriormente podem ser encontradas na Tabela abaixo, levando em consideração a necessidade do modelo dinâmico do sistema para o crítico da aprendizagem por reforço e para os ganhos do controlador [40]:

TABELA 1 – Estruturas ADP X Necessidade do modelo dinâmico do sistema.

Estrutura de ADP	Necessidade do modelo para treinamento:	
	Crítico	Controlador
HDP		X
ADHDP		
DHP	X	X
ADDHP	X	

CAPÍTULO 4

APROXIMADORES LINEARES

O Método dos mínimos quadrados, ou método LS (*Least Square*) é aplicado a inúmeros problemas de estimação encontrados nos mais diversos campos da ciência. Para este trabalho, vamos considerar dois algoritmos de estimação de parâmetros. O algoritmo de estimação em Batelada (*Batch*) e o algoritmo de estimação recursiva, chamado comumente de RLS, que significa *RecursiveLeast Square*, ou seja, mínimos quadrados recursivos.

A estimação em batelada e a estimação recursiva dizem respeito ao algoritmo. Se o mesmo processa os dados de uma só vez, diz-se que tal algoritmo faz estimação em batelada. Se os dados são processados sequencialmente, diz-se que a estimação é recursiva. Por outro lado, a denominação *online* refere-se ao fato do processamento (que pode ser recursivo como em batelada) ocorrer suficientemente rápido de maneira que o resultado esteja disponível para influenciar, dentro do período de amostragem, o processo sendo monitorado ou controlado.

Os aproximadores lineares para a função valor são classes usadas e que assumem a seguinte forma:

$$\hat{V}^h(x_k) = \varphi^T(x_k)\hat{\theta}_k \tag{4.1}$$

sendo$\varphi(x_k) = \left(\varphi_1(x_k), \varphi_2(x_k), \dots, \varphi_s(x_k)\right)^T$ o vetor de função de base e $\hat{\theta}_k = \left(\hat{\theta}_{1k}, \hat{\theta}_{2k}, \dots, \hat{\theta}_{sk}\right)^T$ o vetor de peso estimado. Condições de convergência requerem que as funções de base sejam linearmente independentes para garantir que as matrizes envolvidas nesses aproximadores tenham postos completos.

4.1. Método dos Mínimos Quadrados (LS)

Utilizando a identificação em Batelada, o sistema é excitado e em seguida é armazenado as medidas dos sinais de entrada e saída para aplicação e avaliação posteriormente. O processo de identificação é realizado “de uma só vez” e não existe limitação no tempo de processamento do algoritmo. Grande quantidade de memória pode ser necessária para armazenar os valores das amostras dos sinais de entrada e saída do sistema, além do esforço computacional aumentar proporcionalmente ao tempo do ensaio.

Consideremos uma coleção de pares ordenados obtidos em função de algum experimento, a colocação destes pares ordenados num plano cartesiano, depende dos valores de x_i e y_i, $(i = 1..n)$ fornecendo algum tipo de gráfico. Um fato que atrai pesquisadores aplicados das mais diversas áreas é a possibilidade de obter uma função real que passe nos pontos ou pelo menos passe próximo dos pontos (x_i, y_i) dados.

A *Teoria de Interpolação*é a área da matemática que estuda tais processos para obter funções que passam exatamente pelos pontos dados, enquanto que a *Teoria de Aproximação* estuda processos para obter funções que passem o mais próximo possível dos pontos dados. É óbvio que se pudermos obter funções que passem próximas dos pontos dados e que tenham uma expressão fácil de ser manipulada, teremos obtido algo positivo e de valor científico [46].

Dentre os processos matemáticos que resolvem tal problema com clareza, um dos mais utilizados é o *Método dos Mínimos Quadrados*(LS), que serve para gerar o que se chama em Estatística: *Regressão Linear* ou *Ajuste Linear*.

4.2. Método dos Mínimos Quadrados Recursivos (RLS)

O algoritmo do método dos mínimos quadrados recursivos (RLS), utiliza pouca memória e pequeno esforço computacional. A cada período de amostragem, novas medidas dos sinais de entrada e saídas tornam-se disponíveis e são processados de forma *online*, em tempo real, dependendo da aplicação, fornecendo-se assim estimativas dos parâmetros a cada período de amostragem. Isso é útil quando os parâmetros do processo variam lentamente em função de não linearidades, desgastes, aquecimento, falhas, dentre outros. Tais algoritmos são também úteis na resolução de problemas numéricos cuja solução em batelada seria difícil [49].

4.2.1. Demonstração do RLS

Suponha um sistema discreto que possa ser descrito da seguinte forma:

$$y_k = \varphi_k^T \theta + e_k \tag{4.2}$$

sendoy_k a saída do sistema, $\varphi_k = [x_{k1} \quad x_{k2} \quad \cdots \quad x_{k1}]^T$ um vetor determinístico conhecido, denominado de vetor de regressão, $\theta = [\theta_1 \quad \theta_2 \quad \ldots \quad \theta_n]^T$ o vetor de parâmetros que se deseja estimar e e_k é o erro do modelo, que é suposto de média nula[49].

A estimação do vetor de parâmetros θ, ocorre a partir de N experimentos:

$$\begin{aligned} y_1 &= \varphi_1^T \theta + e_1 \\ y_2 &= \varphi_2^T \theta + e_2 \\ &\vdots \\ y_N &= \varphi_N^T \theta + e_N \end{aligned} \tag{4.3}$$

O método dos mínimos quadrados tem por objetivo estimar $\hat{\theta}$ de modo a minimizar o funcional que é dado por

$$S = \sum_{k=1}^{N} \left(y_k - \varphi_k^T \hat{\theta}\right)^2 \tag{4.4}$$

A Eq.(4.3) pode ser rescrita na seguinte forma matricial

$$Y = \Phi\theta + E \tag{4.5}$$

sendo

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \qquad \Phi = \begin{bmatrix} \varphi_1^T \\ \varphi_2^T \\ \vdots \\ \varphi_N^T \end{bmatrix} \qquad \text{E} \qquad E = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_N \end{bmatrix} \tag{4.6}$$

Logo a Eq.(4.4) pode ser escrita como:

$$S = \left(Y - \Phi\hat{\theta}\right)^T \left(Y - \Phi\hat{\theta}\right) \tag{4.7}$$

Para minimizar o funcional S em relação ao vetor de parâmetros $\hat{\theta}$, é necessário fazer

$$\frac{\partial S}{\partial \hat{\theta}} = -(Y^T\Phi)^T - \Phi^T Y + 2\Phi^T\Phi\hat{\theta} = 0 \tag{4.8}$$

ou

$$\hat{\theta} = [\Phi^T\Phi]^{-1}\Phi^T Y \tag{4.9}$$

Pode-se provar que a estimativa de mínimos quadrados da Eq.(4.9) é não polarizada, ou seja, se o vetor de regressão φ_k é totalmente independente da saída y_k e deve ser garantido também, que o ruído seja branco:

$$E\left[\hat{\theta}\right] = E[(\Phi^T\Phi)^{-1}\Phi^T Y] = (\Phi^T\Phi)^{-1}\Phi^T E[Y] = (\Phi^T\Phi)^{-1}\Phi^T\Phi\theta = \theta \tag{4.10}$$

Podemos reescrever a Eq.(4.9) da forma a seguir, considerando o intervalo de tempo t.

$$\hat{\theta}(t) = [\Phi^T \Phi]^{-1} \Phi^T Y \tag{4.11}$$

sendo

$$\Phi(t) = \begin{bmatrix} \varphi^T(1) \\ \varphi^T(2) \\ \vdots \\ \varphi^T(t) \end{bmatrix} \quad \text{e} \quad Y = \begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(t) \end{bmatrix} \tag{4.12}$$

Supondo-se que no instante $(t + 1)$ obtém-se nova medida do sistema, então os vetores de medida e saída são reescritos como:

$$\Phi(t + 1) = \begin{bmatrix} \varphi^T(1) \\ \varphi^T(2) \\ \vdots \\ \varphi^T(t) \\ \varphi^T(t + 1) \end{bmatrix} = \begin{bmatrix} \Phi(t) \\ \varphi^T(t + 1) \end{bmatrix} \tag{4.13}$$

e

$$Y(t + 1) = \begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(t) \\ y(t + 1) \end{bmatrix} = \begin{bmatrix} Y(t) \\ y(t + 1) \end{bmatrix} \tag{4.14}$$

As estimativas no instante de tempo t são

$$\hat{\theta}(t) = [\Phi^T(t)\Phi(t)]^{-1} \Phi^T(t) Y(t) \tag{4.15}$$

enquanto que no instante $(t + 1)$, são dadas por

$$\hat{\theta}(t + 1) = [\Phi^T(t + 1)\Phi(t + 1)]^{-1} \Phi^T(t + 1) Y(t + 1) \tag{4.16}$$

sendo

$$\Phi^T(t+1)\Phi(t+1) = [\Phi^T(t) \quad \varphi(t+1)]\begin{bmatrix}\Phi(t)\\ \varphi^T(t+1)\end{bmatrix}$$
$$\Phi^T(t+1)\Phi(t+1) = \Phi^T(t)\Phi(t) + \varphi(t+1)\varphi^T(t+1) \tag{4.17}$$

Uma vez conhecido $\varphi(t+1)$ pode-se atualizar a matriz anterior das correlações $\Phi^T(t)\Phi(t)$ para obter a matriz atual $\Phi^T(t+1)\Phi(t+1)$. Entretanto, é necessário encontrar uma maneira de atualizar a inversa de $\Phi^T(t)\Phi(t)$ sem calcular a matriz inversa em cada instante de tempo.

Adicionalmente, necessita-se atualizar o termo $\Phi^T(t+1)Y(t+1)$, isto é

$$\Phi^T(t+1)Y(t+1) = [\Phi^T(t) \quad \varphi(t+1)]\begin{bmatrix}Y(t)\\ y(t+1)\end{bmatrix}$$
$$\Phi^T(t+1)Y(t+1) = \Phi^T(t)Y(t) + \varphi(t+1)y(t+1) \tag{4.18}$$

Considerando as seguintes definições:

$$P(t) = [\Phi^T(t)\Phi(t)]^{-1} \tag{4.19}$$

$$R(t) = \Phi(t)Y(t) \tag{4.20}$$

Substituindo a Eq.(4.19) e a Eq.(4.20) na Eq.(4.16), temos a seguinte equação:

$$\hat{\theta}(t+1) = P(t+1)R(t+1) \tag{4.21}$$

ou então

$$\hat{\theta}(t) = P(t)R(t) \tag{4.22}$$

enquanto que, das equações Eq.(4.17) e Eq(4.18), chegamos as seguintes equações:

$$P^{-1}(t+1) = P^{-1}(t) + \varphi(t+1)\varphi^{T}(t+1) \tag{4.23}$$

e

$$R(t+1) = R(t) + \varphi(t+1)y(t+1) \tag{4.24}$$

A Eq.(4.24) fornece uma atualização direta de $R(t)$ para $R(t+1)$. A atualização de $P(t)$para $P(t+1)$ pode ser obtida pela Eq.(4.23) aplicando-se o lema da inversa

$$(A + BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1} \tag{4.25}$$

Comparando-se o termo do lado direito da Eq.(4.23) com a Eq.(4.25), obtêm-se

$$P(t+1) = P(t) - \frac{P(t)\varphi(t+1)\varphi^{T}(t+1)P(t)}{1 + \varphi^{T}(t+1)P(t)\varphi(t+1)} \tag{4.26}$$

De acordo com a equação do erro de previsão

$$e(t+1) = y(t+1) - \varphi^{T}(t+1)\hat{\theta}(t) \tag{4.27}$$

Substituindo-se a Eq.(4.27) na Eq.(4.24), tem-se

$$R(t+1) = R(t) + \varphi(t+1) + e(t+1) + \varphi(t+1)\varphi^{T}(t+1)\hat{\theta}(t) \tag{4.28}$$

Substituindo-se as Eq.(4.17) e Eq.(4.22) na Eq.(4.28), temos

$$\begin{gathered} P^{-1}(t+1)\hat{\theta}(t+1) = \\ P^{-1}(t)\hat{\theta}(t) + \varphi(t+1)e(t+1) + \{P^{-1}(t+1) - P^{-1}(t)\}\hat{\theta}(t) \end{gathered} \tag{4.29}$$

$$\hat{\theta}(t+1) = \hat{\theta}(t) + P(t+1)\varphi(t+1)e(t+1) \tag{4.30}$$

O termo $P(t+1)\varphi(t+1)$ é um vetor coluna e é denominado ganho do estimador, ou seja,

$$K(t+1) = P(t+1)\varphi(t+1) = \frac{P(t)\varphi(t+1)}{1+\varphi^T(t+1)P(t)\varphi(t+1)} \tag{4.31}$$

O vetor de parâmetros estimados é calculado por

$$\hat{\theta}(t+1) = \hat{\theta}(t) + K(t+1)e(t+1) \tag{4.32}$$

Combinando-se as Eq.(4.26), Eq.(4.28), Eq.(4.27) e sendo $k = t+1$, obtém-se a fórmula recursiva de $\hat{\theta}(k)$ dada por

$$\begin{aligned} \hat{\theta}(k) &= \hat{\theta}(k-1) + \frac{P(k-1)\varphi(k)}{1+\varphi^T(k)P(k-1)\varphi(k)}\{y(k)-\varphi^T(k)\hat{\theta}(k-1)\} \\ P(k) &= P(k-1) - \frac{P(k-1)\varphi(k)\varphi^T(k)P(k-1)}{1+\varphi^T(k)P(k-1)\varphi(k)} \end{aligned} \tag{4.33}$$

4.2.2. Fator de Esquecimento

Se o sistema a ser identificado variar no tempo é necessário fornecer ao algoritmo de mínimos quadrados uma capacidade de ponderação diferenciada para as observações, dando-se uma maior importância às últimas medidas, uma vez que elas contêm informações mais atualizadas e precisam ter maior influência na estimação dos parâmetros.

O algoritmo dos mínimos quadrados recursivos (RLS) padrão minimiza o funcional apresentado na Eq.(4.4) e pondera cada erro de estimação igualmente. Assim, para ponderar as informações mais recentes o critério dos mínimos quadrados recursivos pode ser modificado para minimizar o funcional modificado,

$$S = \sum_{k=1}^{N} \lambda^{N-k}\left(y_k - \varphi_k^T\hat{\theta}\right)^2 \tag{4.34}$$

sendoλo fator de esquecimento. Se $\lambda = 1$, tem-se os mínimos quadrados recursivos padrão. Na prática usa-se valores na faixa $0{,}9 \leq \lambda \leq 1$. Deste modo, as medidas antigas são exponencialmente "esquecidas" e maior ênfase é atribuída as medidas mais recentes. O algoritmo de estimação dos mínimos quadrados recursivos com fator de esquecimento λ apresenta a seguinte forma recursiva [50]:

$$\hat{\theta}(k) = \hat{\theta}(k-1) + \frac{P(k-1)\varphi(k)}{\lambda + \varphi^T(k)P(k-1)\varphi(k)}\{y(k) - \varphi^T(k)\hat{\theta}(k-1)\}$$
$$P(k) = \frac{1}{\lambda}\left\{P(k-1) - \frac{P(k-1)\varphi(k)\varphi^T(k)P(k-1)}{\lambda + \varphi^T(k)P(k-1)\varphi(k)}\right\} \quad (4.35)$$

Abaixo pode ser visto os passos para a implementação do algoritmo do estimador RLS:

1. Atualizar o vetor de medidas $\varphi(k)$ (entrada e saída do sistema);
2. Calcular o vetor de parâmetros $\hat{\theta}(k)$;
3. Calcular a matriz de covariância $P(k)$;
4. Retornar ao passo 1.

4.3. Estabilidade dos Algoritmos dos Mínimos Quadrados

Em aplicações práticas do algoritmo dos mínimos quadrados é impossível não ter cuidado do seu comportamento com relação a estabilidade numérica. Os fatores que provocam a instabilidade, ou a divergência, dos algoritmos de mínimos quadrados são os erros de arredondamento consequentes da representação em precisão finita. Vale ressaltar, que outros fatores também contribuem severamente para a aceleração do processo de divergência. A escolha do fator de esquecimento e a característica de estacionaridade do sinal de entrada, são parâmetros às vezes tão influentes quanto a representação das variáveis.

O comportamento dos algoritmos de mínimos quadrados em uma implementação prática envolve, tipicamente, três fases [48]:

A primeira fase é a etapa de convergência, onde as variáveis evoluem dos valores iniciais até os valores de regime. A duração desta fase é breve (algumas dezenas de iterações) e praticamente isenta de ruído de quantização, devido ao pouco tempo que este teve para se acumular.

A segunda fase ocorre quando o algoritmo encontra-se em regime, e é caracterizado por uma flutuação das variáveis em torno dos valores esperados. Em condições hipotéticas de precisão infinita, tendo como entrada um sinal bem comportado, esta fase deveria durar indefinidamente. Entretanto, o ruído de quantização acumulado produz alterações nas variáveis. Estas alterações ao longo do tempo se tornam significativas até que, em um dado instante, observa-se abrupta nos valores das variáveis escalares e vetoriais. Esta divergência é uma particularidade dos algoritmos rápidos que pode se manifestar também no algoritmo RLS se alguns cuidados especiais não forem tomados na sua implementação.

A terceira fase seguida após a divergência é caracterizada por uma total instabilidade, com as variáveis atingindo valores aleatórios e extremos, sem qualquer tendência de retornar as condições anteriores, ou de regime. Esse processo de total descontrole pode também determinar uma ocorrência de *overflow*, o que representa uma situação inaceitável, sobretudo quando se processa sinais em tempo real.

A instabilidade das variáveis, a ocorrência de *overflow*, e outros inconvenientes, como a divisão por zero, são eventos que devem ser evitados a qualquer custo, mesmo tendo que se prescindir da aplicação rigorosa do critério dos mínimos quadrados. As pequenas alterações no algoritmo são toleráveis, desde que eliminem o problema da instabilidade. Muitos artifícios foram propostos, visando principalmente a estabilização dos algoritmos rápidos.

Existem três fatores que, conjunta ou separadamente, provocam a divergência dos algoritmos dos mínimos quadrados. Estes fatores são listados abaixo:

- A falta de excitação persistente;
- O fator de esquecimento;
- O ruído de quantização;

4.3.1. Excitação persistente no RLS

A falta da excitação persistente está ligada às características do sinal de entrada e é comum ao algoritmo dos mínimos quadrados recursivos (RLS) e aos algoritmos rápidos. Portanto, a falta de excitação persistente está ligada às características de estacionaridade do sinal de entrada.

A falta de excitação persistente consiste na variação extrema do ganho no algoritmo RLS, fazendo com que pequenos ruídos (de arredondamento) sejam amplificados ao nível de produzirem a instabilidade. Uma definição matematicamente precisa da condição de excitação persistente, pode ser encontrada nas referências [50] e [51].

Uma maneira simples de solucionar o problema da falta de excitação persistente consiste em adicionar um ruído branco de pouca amplitude ao sinal de entrada, eliminando-se assim as regiões nulas do espectro. Este procedimento equivale a acrescentar valores positivos aos elementos da diagonal principal da matriz de covariância$R(t)$, tornando a matriz melhor condicionada. Os sinais encontrados em aplicações práticas são, de forma geral bem comportados e, prescindem deste artifício.

A instabilidade ocasionada pela falta de excitação persistente não tem relação com o ruído de arredondamento, pois a matriz de autocorrelação do sinal de entrada tenderia à singularidade ainda que as variáveis tivessem precisão infinita.

O algoritmo RLS é consagrado por sua robustez numérica, mas sua implementação computacional exige cuidados na preservação da simetria da matriz inversa, $P(t)$. Caso contrário, o algoritmo torna-se extremamente instável, apresentando um desempenho muito inferior aos demais algoritmos. O artifício necessário para manter o algoritmo RLS estável consiste em se garantir que a matriz $P(t)$ permaneça simétrica ao longo do tempo.

A manutenção da simetria da matriz $P(t)$, é caracterizada pelo aumento do número de operações no algoritmo RLS. Existe uma outra possibilidade de se garantir a simetria da matriz $P(t)$, que corresponde a utilizar somente os elementos situados na diagonal principal e acima dela. Com isto, a implementação é mais adequada pois além da simetria intrínseca, exige um

número menor de variáveis e minimiza o número de operações aritméticas por iteração.

Uma análise do algoritmo dos mínimos quadrados recursivos pode ser encontrada em [51], onde é mostrado sob certas condições, a preservação da simetria de $P(t)$ e garante que esta se mantém positiva definida, mesmo com o acúmulo de erros.

CAPÍTULO 5

PROJETO DO ALGORITMO HDP-RLS

O algoritmo de programação dinâmica adaptativa que será apresentado neste capítulo é modelo por críticos adaptativos e serão utilizados na resolução do regulador quadrático linear discreto (DLQR) através das formulações matemáticas que foram demonstradas nos capítulos anteriores. Vale ressaltar que o algoritmo adaptativo que será implementado é baseado na programação dinâmica heurística (HDP) e será direcionado para a resolução da equação de *Riccati* de forma *online*. O modelo de aproximador linear baseado no método dos mínimos quadrados recursivos (RLS) é utilizado para aproximar a função valor que representa o custo da política de controle corrente.

No esquema da programação dinâmica heurística, deve-se conhecer a dinâmica do sistema para a atualização da ação de controle. A planta que será utilizada no projeto deste algoritmo é um circuito elétrico que representa um modelo dinâmico multivariável de quarta ordem. As análises preliminares da planta serão mostradas neste capítulo antes da elaboração do algoritmo HDP-RLS (baseado em programação dinâmica heurística e, utilizando o para estimar a função valor o método dos mínimos quadrados recursivos).

5.1. Sistemas Dinâmicos Multivariáveis

As grandes exigências para redução de custos de produção aliada às necessidades de aumento produtivo, impostas pelo mercado consumidor têm forçado as indústrias a investirem cada vez mais no estudo de novas técnicas de controle dos seus processos. O rápido avanço industrial em resposta a estas exigências de mercado elevou o grau de complexidade dos processos modernos,

aumentando não apenas as interações entre as suas variáveis, como o número de malhas de controle necessárias para manter as condições de operação e qualidade do produto final dentro dos limites pré-estabelecidos [10][28].

Em processos complexos e com alto grau de acoplamento entre as variáveis, os mais altos níveis de desempenho, não são mais atingidos utilizando apenas o controle regulatório simples (que é utilizado em sistemas de entrada e saída simples) [28]. Nestes casos, estratégias de controle para sistemas multivariáveis tornam-se indispensáveis, pois surgem neste cenário buscando incrementar a operabilidade e a produção dos processos diminuindo a sua variabilidade e permitindo, desta forma, que estados estacionários mais próximos dos limites de especificação e operação sejam atingidos.

Os Sistemas Lineares e Invariantes no Tempo (SLIT) são de importância central no estudo da engenharia elétrica, principalmente nas áreas de processamento de sinais e sistemas de controle. Um modelo de sistema é um conjunto de expressões matemáticas que determinam o valor de sinais de saída a partir dos valores de variáveis de entrada. Estas expressões podem representar, por exemplo o modelo matemático de um filtro eletrônico, ou de um dispositivo mecânico.

Sistemas complexos podem ser decompostos em subsistemas mais simples. Um diagrama de blocos de um sistema é a sua representação a partir de elementos mais simples de processamento de sinais [12].

Um sistema é dito invariante no tempo quando seus parâmetros não variam no tempo.A linearidade dos sistemas implica que todas as operações utilizadas no processamento dos sinais de entrada são lineares.

A propriedade mais importante de um sistema linear invariante no tempo é o princípio da superposição. Este princípio diz que a saída de um sinal formado pela combinação linear de diferentes sinais, é igual à mesma combinação aplicada aos sinais de saída gerados por cada sinal original separadamente. Ou seja, suponha a existência de dois sinais $x(t)$ e $y(t)$. Cada um desses sinais, se aplicarmos individualmente a um certo sistema H, criam as saídas $x'(t)$ e $y'(t)$. Se a entrada do sistema for a combinação linear $ax(t) + by(t)$, a saída será $ax'(t) + by'(t)$.

Os sistemas lineares e invariantes no tempo são classificados de acordo com os tipos abaixo:

- *SISO (Single Input, Single Output):* O sistema possui uma única entrada e uma única saída.
- *MISO (Multiple Input, Single Output):* A equação equivalente do sistema depende de todas as entradas. Para um sistema MISO ser linear, os diferentes sinais só podem ser somados entre si. Não pode haver multiplicação entre eles ou derivação de um sinal em relação a outro.
- *SIMO (Single Input, Multiple Output):* Percebemos que este tipo de sistema pode ser decomposto em sistemas SISO individuais.
- *MIMO (Multiple Input, Multiple Output):* Da mesma forma do anterior, podemos decompor este sistemaem sistemas do tipo MISO.

5.1.1. Descrições do Modelo

A modelagem matemática da planta que será utilizada neste trabalho pode ser vista no Apêndice A, e serve como base para as descrições do modelo que serão apresentadas nesta secção. Estas descrições são divididas em obtenção da matriz de transferência, obtenção das equações de espaço de estados e o cálculo dos autovalores e autovetores.

5.1.1.1.Matriz de Transferência

As Eq.(A.19) e Eq.(A.38) serão utilizadas para a determinação da matriz de transferência do circuito elétrico que representa o sistema dinâmico multivariável. Colocando as Eq.(A.19) e Eq.(A.38) na forma matricial, temos:

$$\begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix} = A \begin{bmatrix} Y_1(s) \\ Y_2(s) \end{bmatrix} \tag{5.1}$$

Portanto para encontrarmos a matriz de transferência deve-se inverter a matriz A que é dada por:

$$A = \begin{bmatrix} \left(\frac{L_1}{R_2}s + L_1C_1s^2 + 1 + \frac{R_1}{R_2} + R_1C_1s\right) & -\left(\frac{R_1}{R_3} + R_1C_2s\right) \\ -\left(\frac{R_1}{R_2} + R_1C_1s\right) & \left(\frac{L_2}{R_3}s + L_2C_2s^2 + \frac{R_1}{R_3} + R_1C_2s + 1\right) \end{bmatrix} \quad (5.2)$$

Se admitirmos que os componentes $R_1 = R_2 = R_3 = R$, $L_1 = L_2 = L$ e $C_1 = C_2 = C$, temos:

$$A = \begin{bmatrix} \left(\frac{L}{R}s + LCs^2 + 1 + \frac{R}{R} + RCs\right) & -\left(\frac{R}{R} + RCs\right) \\ -\left(\frac{R}{R} + RCs\right) & \left(\frac{L}{R}s + LCs^2 + 1 + \frac{R}{R} + RCs\right) \end{bmatrix} \quad (5.3)$$

$$A = \begin{bmatrix} \left(\frac{L}{R}s + LCs^2 + 2 + RCs\right) & -(1 + RCs) \\ -(1 + RCs) & \left(\frac{L}{R}s + LCs^2 + 2 + RCs\right) \end{bmatrix} \quad (5.4)$$

Invertendo a matriz A, temos:

$$A^{-1} = \begin{bmatrix} \frac{\left(\frac{L}{R}s + LCs^2 + 2 + RCs\right)}{d(s)} & \frac{(1 + RCs)}{d(s)} \\ \frac{(1 + RCs)}{d(s)} & \frac{\left(\frac{L}{R}s + LCs^2 + 2 + RCs\right)}{d(s)} \end{bmatrix} \quad (5.5)$$

sendo

$$d(s) = L^2C^2s^4 + \left(2\frac{L^2C}{R} + 2RLC^2\right)s^3 + \left(\frac{L^2}{R^2} + 6LC\right)s^2 + \left(4\frac{L}{R} + 2RC\right)s + 3 \quad (5.6)$$

Se considerarmos todos os elementos do circuito elétrico com valores unitários, ou seja, $R = L = C = 1$ na sua unidade respectivamente, temos:

$$A^{-1} = \begin{bmatrix} \frac{s^2 + 2s + 2}{s^4 + 4s^3 + 7s^2 + 6s + 3} & \frac{s + 1}{s^4 + 4s^3 + 7s^2 + 6s + 3} \\ \frac{s + 1}{s^4 + 4s^3 + 7s^2 + 6s + 3} & \frac{s^2 + 2s + 2}{s^4 + 4s^3 + 7s^2 + 6s + 3} \end{bmatrix} \quad (5.7)$$

Portanto temos a matriz de transferência para o circuito elétrico de quarta ordem:

$$\begin{bmatrix} Y_1(s) \\ Y_2(s) \end{bmatrix} = \begin{bmatrix} \frac{s^2 + 2s + 2}{s^4 + 4s^3 + 7s^2 + 6s + 3} & \frac{s + 1}{s^4 + 4s^3 + 7s^2 + 6s + 3} \\ \frac{s + 1}{s^4 + 4s^3 + 7s^2 + 6s + 3} & \frac{s^2 + 2s + 2}{s^4 + 4s^3 + 7s^2 + 6s + 3} \end{bmatrix} \begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix} \quad (5.8)$$

$$\begin{bmatrix} Y_1(s) \\ Y_2(s) \end{bmatrix} = \frac{1}{s^4 + 4s^3 + 7s^2 + 6s + 3} \begin{bmatrix} s^2 + 2s + 2 & s + 1 \\ s + 1 & s^2 + 2s + 2 \end{bmatrix} \begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix} \quad (5.9)$$

5.1.1.2. Espaço de Estados

Para a determinação das equações de espaço de estados para o sistema dinâmico multivariável, devemos selecionar as variáveis de estado de acordo com os elementos do circuito (capacitores e indutores). Logo, temos as seguintes variáveis:

$$\begin{aligned} V_{C1}(t) &= x_1 \\ V_{C2}(t) &= x_2 \\ i_{L1}(t) &= x_3 \\ i_{L2}(t) &= x_4 \end{aligned} \quad (5.10)$$

Observando o circuito, e utilizando a lei das correntes de *Kichhoff,* temos a seguinte equação:

$$i_{L1}(t) = i_{C1}(t) + i_{R2}(t) \quad (5.11)$$

sende:

$i_{L1}(t)$a corrente do Indutor 1 (L_1);

$i_{C1}(t)$a corrente do Capacitor 1 (C_1);

$i_{R2}(t)$a corrente do Resistor 2 (R_2);

Temos assim:

$$C_1 \frac{dV_{C1}(t)}{dt} = i_{L1}(t) - \frac{V_{C1}(t)}{R_2} \tag{5.12}$$

Substituindo as variáveis de estado da Eq.(5.10) na Eq.(5.12):

$$C_1 \dot{x_1} = x_3 - \frac{x_1}{R_2} \tag{5.13}$$

Do mesmo modo:

$$C_2 \dot{x_2} = x_4 - \frac{x_2}{R_3} \tag{5.14}$$

Utilizando a lei das tensões de *Kirchhoff*no circuito elétrico da figura A.1, temos:

$$u_1(t) = V_{L1}(t) + V_{R1}(t) + V_{C1}(t) \tag{5.15}$$

$$u_1(t) = V_{L1}(t) + V_{C1}(t) + R_1(i_{L1}(t) - i_{L2}(t)) \tag{5.16}$$

Substituindo as variáveis de estado da Eq.(5.10) na Eq.(5.16):

$$u_1(t) = L\dot{x}_3 + x_1 + R_1(x_3 - x_4) \tag{5.17}$$

$$u_2(t) = L\dot{x}_4 + x_2 + R_1(x_4 - x_3) \tag{5.18}$$

Isolando as variáveis $\dot{x}_1, \dot{x}_2, \dot{x}_3$ e $\dot{x}_4$ nas Eq.(5.13), Eq.(5.14), Eq.(5.17) e Eq.(5.18) respectivamente, temos:

$$\dot{x_1} = \frac{1}{C_1}x_3 - \frac{1}{R_2C_1}x_1 \tag{5.19}$$

$$\dot{x_2} = \frac{1}{C_2}x_4 - \frac{1}{R_3C_2}x_2 \tag{5.20}$$

$$\dot{x_3} = -\frac{1}{L_1}x_1 - \frac{R_1}{L_1}x_3 + \frac{R_1}{L_1}x_4 + \frac{1}{L_1}u_1(t) \tag{5.21}$$

$$\dot{x_4} = -\frac{1}{L_2}x_2 + \frac{R_1}{L_2}x_3 - \frac{R_1}{L_2}x_4 + \frac{1}{L_2}u_2(t) \tag{5.22}$$

As equações de estado para um dado sistema dinâmico é dado pelo seguinte sistema de equações matriciais:

$$\dot{x} = Ax + Bu \tag{5.23}$$

$$y = Cx + Du \tag{5.24}$$

Utilizando as Eq.(5.18), Eq.(5.19), Eq.(5.20) e Eq.(5.21), e fazendo analogia com a Eq.(5.23), temos a seguinte equação de espaço de estados:

$$\begin{bmatrix}\dot{x_1}\\ \dot{x_2}\\ \dot{x_3}\\ \dot{x_4}\end{bmatrix} = \begin{bmatrix}-1/R_2C_1 & 0 & 1/C_1 & 0\\ 0 & -1/R_3C_2 & 0 & 1/C_2\\ -1/L_1 & 0 & -R_1/L_1 & R_1/L_1\\ 0 & -1/L_2 & R_1/L_2 & -R_1/L_2\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{bmatrix} + \begin{bmatrix}0 & 0\\ 0 & 0\\ 1 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}u_1\\ u_2\end{bmatrix} \tag{5.25}$$

Como a saída do sistema é dada pelas tensões nos capacitores C_1 e C_2, temos as seguintes considerações:

$$y_1 = V_{C1}(t) = x_1 \tag{5.26}$$

$$y_2 = V_{C2}(t) = x_2 \tag{5.27}$$

Utilizando as Eq.(5.26) e Eq.(5.27), e fazendo analogia com a Eq.(5.24), temos a seguinte equação no espaço de estados:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{5.28}$$

Se considerarmos que todos os elementos do circuito são unitários nas suas respectivas unidades, ou seja, $R_1 = R_2 = R_3 = L_1 = L_2 = C_1 = C_2 = 1$, temos as seguintes equações para o espaço de estados do sistema:

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \\ \dot{x_4} \end{bmatrix} = \begin{bmatrix} -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ -1 & 0 & -1 & 1 \\ 0 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{5.29}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \tag{5.30}$$

As matrizes do espaço de estado do sistema são dadas por

$$A = \begin{bmatrix} -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ -1 & 0 & -1 & 1 \\ 0 & -1 & 1 & -1 \end{bmatrix} \tag{5.31}$$

$$B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{5.32}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{5.33}$$

$$D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{5.34}$$

5.1.1.3. Autovalores e Autovetores

Os autovalores do sistema dinâmico de quarta ordem são determinados a partir das raízes do polinômio característico da matriz de transferência, dado pela Eq.(5.6), ou seja, a solução da equação:

$$s^4 + 4s^3 + 7s^2 + 6s + 3 = 0 \tag{5.35}$$

Portanto, resolvendo a Eq.(5.35), temos os autovalores que são mostrados a seguir:

$$\begin{aligned} \lambda_1 &= -1.5 + 0.866\mathrm{i} \\ \lambda_2 &= -1.5 - 0.866\mathrm{i} \\ \lambda_3 &= -0.5 + 0.866\mathrm{i} \\ \lambda_4 &= -0.5 - 0.866\mathrm{i} \end{aligned} \tag{5.36}$$

Os autovetores associados a estes autovalores são mostrados a seguir, e satisfazem a equação abaixo:

$$(\lambda_i I - A)v_i = 0 \tag{5.37}$$

sendo

$\lambda_i$representa o autovalor i;

$v_i$representa o autovetor associado ao autovalor i;

Ié a matriz identidade;

Aé a matriz do espaço de estados;

Os autovetores são:

$$v_1 = \begin{bmatrix} -0.25 - 0.433i \\ 0.25 + 0.433i \\ 0.5 \\ -0.5 \end{bmatrix} \tag{5.38}$$

$$v_2 = \begin{bmatrix} -0.25 + 0.433i \\ 0.25 - 0.433i \\ 0.5 \\ -0.5 \end{bmatrix} \tag{5.39}$$

$$v_3 = \begin{bmatrix} 0 \\ 0 \\ -0.5 + 0.866i \\ 0 \end{bmatrix} \tag{5.40}$$

$$v_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -0.5 - 0.866i \end{bmatrix} \tag{5.41}$$

5.1.2. Análise Dinâmica

Na análise dinâmica do sistema multivariável trabalhada nesta seção, será levada em consideração, a resposta do sistema ao estado nulo e a estabilidade assintótica.

A resposta ao estado nulo é BIBO estável se e somente se todo pólo da função de transferênciativer parte real negativa (isto é, todos os pólos dos elementos $G_{ij}(s)$da matriz de transferência).

Para calcularmos os pólos da matriz de transferência utilizamos o polinômio característico (Eq.(5.35)), que é o mesmo para todos os elementos da matriz, então:

$$\begin{aligned} p_1 &= -1.5 + 0.866\text{i} \\ p_2 &= -1.5 - 0.866\text{i} \\ p_3 &= -0.5 + 0.866\text{i} \\ p_4 &= -0.5 - 0.866\text{i} \end{aligned} \tag{5.42}$$

Verificamos assim, que os pólos são iguais aos autovalores Eq.(5.36), e que a resposta do sistema ao estado nulo é BIBO estável, pois todos os pólos possuem parte real negativa.

5.1.3. Especificações de Projeto

Para a análise de especificações do projeto, será utilizada a resposta ao degrau do sistema, resposta ao impulsoe o diagrama de valores singulares para a análise no domínio da frequência, pois este sistema trata-se de um sistema dinâmico multivariável, onde o diagrama de bode não é aplicado.

As respostas ao impulso do sistema são mostradas nas figuras abaixo para as duas saídas que representam as tensões nos capacitores C_1 e C_2 da figura A.1.

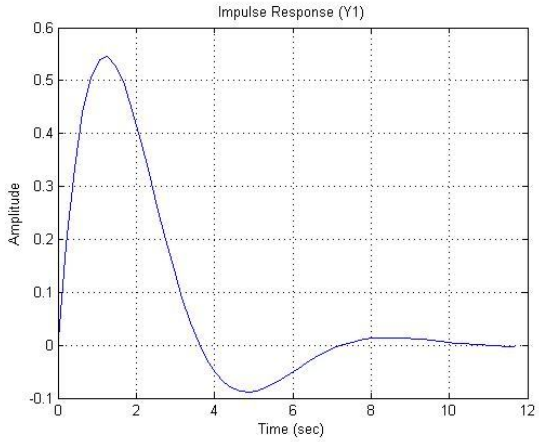

FIGURA 7 – Resposta ao Impulso do Sistema MIMO (Saída 1)

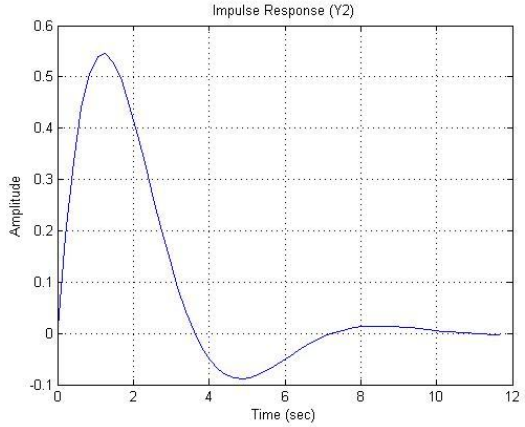

FIGURA 8 – Resposta ao Impulso do Sistema MIMO (Saída 2)

A resposta ao degrau do sistema multivariável, será apresentada de duas formas: a primeira será utilizando o teorema da superposição e a segunda com as duas fontes ativas (considerando as duas entradas).

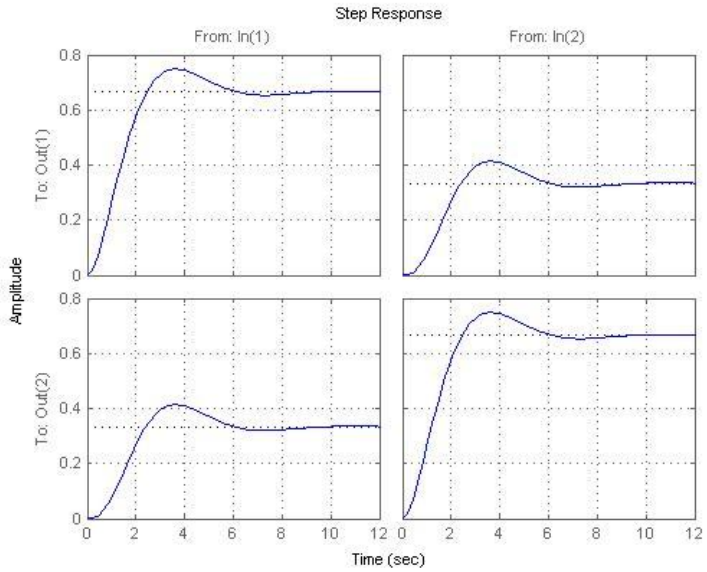

FIGURA 9 – Resposta ao degrau (teorema da superposição).

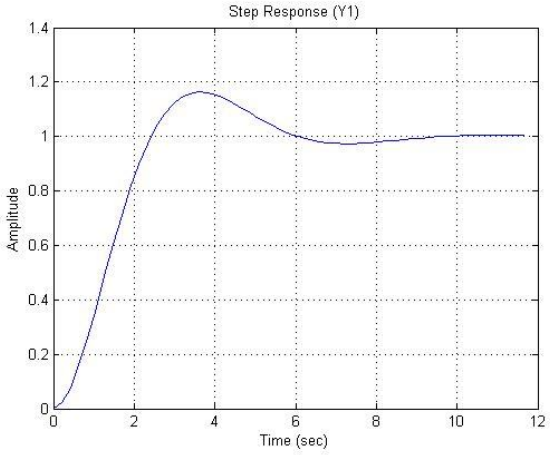

FIGURA 10 – Resposta ao degrau (Saída 1).

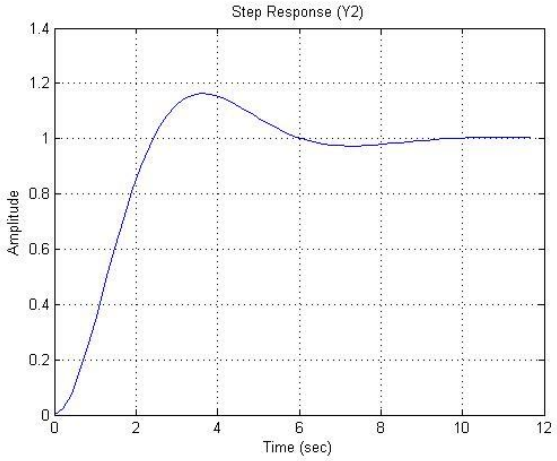

FIGURA 11 - Resposta ao degrau (Saída 2).

Observando as figuras 7,8,9,10 e 11 fica claro que o sistema é estável, o que pode ser confirmado através das análises de estabilidade na seção anterior que se referia a análise dinâmica do sistema multivariável.

Como o sistema analisado é multivariável, a análise de resposta em freqüência é feita através do método dos valores singulares que ocorre através da decomposição em valores singulares da matriz A das equações de espaço de estado, dado por:

$$A = U_{svd} S_{\sigma} V'_{svd} \tag{5.43}$$

sendo

σ_{svd} é a matriz diagonal de dimensão $n \times n$ com elementos não negativos na diagonal.

U_{svd} e V'_{svd} são matrizes unitárias.

A decomposição da matriz A em valores singulares é dada pelas seguintes matrizes:

$$U_{svd} = \begin{bmatrix} 0.2049 & 0.6015 & -0.6768 & -0.3717 \\ -0.2049 & 0.6015 & 0.6768 & -0.3717 \\ -0.6768 & 0.3717 & -0.249 & 0.6015 \\ 0.6768 & 0.3717 & 0.2049 & 0.6015 \end{bmatrix} \tag{5.44}$$

$$V'_{svd} = \begin{bmatrix} 0.2049 & -0.6015 & 0.6768 & -0.3717 \\ -0.2049 & -0.6015 & -0.6768 & -0.3717 \\ 0.6768 & 0.3717 & -0.2049 & -0.6015 \\ -0.6768 & 0.3717 & 0.2049 & -0.6015 \end{bmatrix} \quad (5.45)$$

$$\sigma_{svd} = \begin{bmatrix} 2.3028 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 1.6180 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 1.3028 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.6180 \end{bmatrix} \quad (5.46)$$

A resposta em frequência dos valores singulares é utilizada para avaliar o comportamento do sistema linear invariante no tempo. Na figura abaixo se apresenta os maiores e menores valores singulares em *dB* em função da frequência em *rad/sec*.

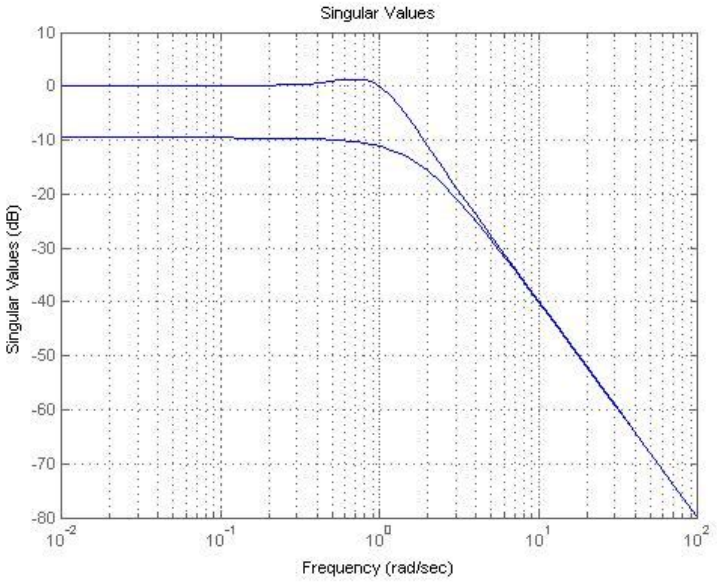

FIGURA 12 – Diagrama de Valores Singulares do Sistema MIMO

5.2. Algoritmo HDP-RLS-DLQR

A estimação da função de custo V^K para uma dada política K associada ao DLQR exige amostragens da recompensa instantânea r_k. Enquanto que os modelos f do ambiente e r_k da recompensa instantânea são necessários

para determinar a função de custo corresponde à política ótima. Neste contexto, métodos RLS abordam o problema de encontrar um vetor de parâmetros θ^K que nos permite estimar V^K, o aproximador linear é dado por:

$$V^K(x_k) = \varphi^T(x_k)\theta^K \tag{5.47}$$

e, portanto, deve satisfazer a seguinte condição de consistência

$$V^K(x_k) = r\big(x_k, h(x_k)\big) + \gamma V^K\Big(f\big(x_k, h(x_k)\big)\Big) \tag{5.48}$$

A forma quadrática da função de custo, dada pela Eq.(2.22), é representada utilizando-se definições de produto de *Kronecker* e vetorização para obtenção de uma estrutura de aproximação linear no vetor de parâmetros θ^K,

$$V^K(x_k) = x_k^T P x_k = \overline{x_k}^T vec(P) \tag{5.49}$$

sendo$\overline{x_k} \in \Re^{n(n+1)/2}$ definido de acordo com o produto de *Kronecker*

$$\overline{x_k}^T = x_k^T \otimes x_k^T = [x_{1k}x_k^T \ \dots \ x_{nk}x_k^T] \tag{5.50}$$

$vec(P) \in \Re^{n(n+1)/2}$contém as n entradas diagonais de P e as $[n(n+1)/2] - n$ somas distintas dos elementos $p_{ij} + p_{ji}$ da matriz P. O vetor de parâmetros θ^K corresponde à forma vetorizada da matriz P, i.e. $\theta^K = vec(P)$e$\varphi(x_k) = \overline{x_k}$.

Reordenando a Eq.(5.48) e usando a Eq.(5.49), tem-se:

$$r(x_k, u_k) = \overline{x_k}^T vec(P) - \gamma\overline{x_{k+1}}^T vec(P) = \Phi_k{}^T\theta^K \tag{5.51}$$

sendo$\Phi_k = \overline{x_k} - \gamma\overline{x_{k+1}}$.

Assim, a função de utilidade instantânea r_k pode ser vista como a função objetivo desejada para a qual necessita-se ajustar o vetor de parâmetros θ^K no sentido dos mínimos quadrados, isto é, minimizando-se a seguinte função de custo.

$$J_N = \sum_{i=1}^{N} \mu^{N-i} \left| r_{k-N+i} - \Phi_{k-N+i}^T \right|^2 \tag{5.52}$$

sendo$N$ a quantidade de dados amostrados $(r_{k-N+i}, \Phi_{k-N+i})$ e a constante $0 < \mu \leq 1$ é o fator de esquecimento, parâmetro que pondera a influência das amostras mais recentes. Para satisfazer a condição de excitação do problema dos mínimos quadrados necessita-se que a quantidade N de dados amostrados seja no mínimo igual a $n(n+1)/2$. A estimativa dos mínimos quadrados é baseada sobre N observações é dada por

$$\hat{\theta}(N) = \Lambda^{-1}(N)\Theta(N) \tag{5.53}$$

sendo

$$\Lambda(N) = \sum_{i=1}^{N} \mu^{N-i} \Phi_{k-N+i}\, \Phi_{k-N+i}^T \tag{5.54}$$

a matriz de autocorrelação que deve ser inversível, e

$$\Theta(N) = \sum_{i=1}^{N} \mu^{N-i} \Phi_{k-N+i} r_{k-N+i} \tag{5.55}$$

a matriz de correlação cruzada.

A solução dos mínimos quadrados dada na Eq.(5.53) pode ser reescrita em uma forma recursiva, de modo que a estimativa obtida na etapa de tempo $i-1$ é usada para obter a estimativa na etapa de tempo i, com $i = 1,2,\ldots,N$. Denotando por $\hat{\theta}(N-1)$ a estimativa dos mínimos quadrados baseada sobre $N-1$ observações e $\Gamma(N) = \Lambda^{-1}(N)$, segue da Eq.(5.54) que

$$\Gamma^{-1}(N) = \mu \sum_{i=1}^{N-1} \mu^{N-1-i} \Phi_{k-N+i} \Phi_{k-N+i}^T + \Phi_k \Phi_k^T \tag{5.56}$$

$$\Gamma^{-1}(N) = \mu\Gamma^{-1}(N-1) + \Phi_k\Phi_k^T \tag{5.57}$$

Reescrevendo a Eq.(5.53) como

$$\hat{\theta}(N) = \Gamma(N)\left(\mu\sum_{i=1}^{N-1}\mu^{N-1-i}\Phi_{k-N+i}r_{k-N+i} + \Phi_k\Phi_k^T\right) \tag{5.58}$$

e observando-se que

$$\sum_{i=1}^{N-1}\mu^{N-1-i}\Phi_{k-N+i}r_{k-N+i} = \Gamma^{-1}(N-1)\hat{\theta}(N-1) \tag{5.59}$$

Substituindo Eq.(5.57) em Eq.(5.59) para obter

$$\hat{\theta}(N) = \hat{\theta}(N-1) - \Gamma(N)\Phi_k\Phi_k^T\hat{\theta}(N-1) + \Gamma(N)\Phi_k r_k \tag{5.60}$$

Para deduzir uma equação recursiva para $\Gamma(N)$ ao invés de $\Gamma^{-1}(N)$ como na Eq.(5.56), utiliza-se o lema da inversa: *Sejam* A, C *e* $C^{-1} + DA^{-1}B$ *matrizes quadradas não singulares. Então* $A + BCD$ *é inversível, e* $(A + BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} + DA^{-1})B^{-1}DA^{-1}$. Aplicando-se o lema da inversa na Eq.(5.56), tem-se:

$$\Gamma(N) = (\mu\Gamma^{-1}(N-1) + \Phi_k\Phi_k^T)^{-1} \tag{5.61}$$

$$\begin{gathered}\Gamma(N) = \mu^{-1}\Gamma(N-1) \\ -\mu^{-1}\Gamma(N-1)\Phi_k\Big(I + \Phi_k^T\mu^{-1}\Gamma(N-1)\Big)^{-1}\Phi_k^T\mu^{-1}\Gamma(N-1)\end{gathered} \tag{5.62}$$

$$\Gamma(N) = \mu^{-1}\left(\Gamma(N-1) - \frac{\Gamma(N-1)\Phi_k\Phi_k^T\Gamma(N-1)}{\mu + \Phi_k^T\Gamma(N-1)\Phi_k}\right) \tag{5.63}$$

Isto implica na seguinte forma recursiva da solução dada na Eq.(5.53):

$$L_j(i) = \Gamma_j(i)\Phi_k = \frac{\Gamma_j(i-1)\Phi_k}{\mu + \Phi_k^T\Gamma_j(i-1)\Phi_k} \tag{5.64}$$

$$\hat{\theta}_j(i) = \hat{\theta}_j(i-1) + L_j(i)\left(r_k - \Phi_k^T \hat{\theta}_j(i-1)\right) \tag{5.65}$$

$$\Gamma_j(i) = \mu^{-1}\left(\Gamma_j(i-1) - \frac{\Gamma_j(i-1)\Phi_k\Phi_k^T\Gamma_j(i-1)}{\mu + \Phi_k^T\Gamma_j(i-1)\Phi_k}\right) \tag{5.66}$$

sendo$j$ o índice de atualização de política, i o índice das recursividades do RLS, k o tempo discreto, $\theta_j = vec(P)$ o vetor de parâmetro verdadeiro para a função V^{K_j}, $\hat{\theta}_j(i)$ a i-ésima estimativa de θ_j, $\Gamma_j(0) = \Gamma_0$, com $\Gamma_0 = \delta I$ para alguma constante positiva δ, e $\Gamma_{j+1}(0) = \Gamma_j$.

Observando na Eq.(5.65) que a estimativa $\hat{\theta}_j(i)$é obtida adicionando-se uma correção à estimativa anterior $\hat{\theta}_j(i-1)$ que é proporcional à diferença temporal $\Delta_k = r_k - \Phi_k^T\hat{\theta}_j$. As componentes do vetor ganho $L_j(i)$ são ponderações que nos dizem como o vetor de parâmetros $\hat{\theta}_j(i)$deve ser ajustado adequadamente.

O algoritmo HDP-RLS-DLQR possui dois laços principais que são dos processos iterativos da HDP e do RLS para a determinação da política de decisão baseada nos métodos de aprendizagem por reforço. Nesta situação, o algoritmo customizado é formado pelos seguintes passos principais:

ALGORITMO IV– ALGORITMO HDP-RLS-DLQR

1 ► **Inicialização dos Parâmetros**
2 ►Sistema Dinâmico Discreto e Matrizes de Ponderação
3 A_d, B_d, C_d,D_d, Q e R.
4 ► Parâmetros do RLS
5 $\theta_0(0) \leftarrow 0$
6 $\Gamma_0(0) \leftarrow \Gamma_0$
7 $\mu \leftarrow$fator de esquecimento
8 ► Selecionar qualquer política de controle admissível, K_0 e matriz P_0
9 $k \leftarrow 0$
10 ► **Início do Processo Iterativo**
11 Para $k \leftarrow k + 1$ passo 1
12 ► Ação de Controle
13 $u_k \leftarrow K^{\theta} x_k$

14 ► Estados

15 $x_{k+1} \leftarrow A_d x_k + B_d u_k$

16 ► Montagem do Vetor Alvo

17 $\theta(x, r, f, P^\theta) \leftarrow x_k^T Q x_k + u_k^T R u_k + x_{k+1}^T P^\theta x_{k+1}$

18 ► Conjunto Base – Produto de *Kronecker*

19 $\Phi_k \leftarrow [x_{1k}^2;\ x_{1k}x_{2k};\ x_{1k}x_{3k};\ x_{1k}x_{4k}x_{2k}^2;\ x_{2k}x_{3k};\ x_{2k}x_{4k};\ x_{3k}^2;\ x_{3k}x_{4k};\ x_{4k}^2]$

20 ► Mínimos quadrados recursivos (RLS)

21 $\theta \leftarrow \left(\Phi_k {\Phi_k}^T\right)^{-1} \Phi_k\, \theta(x, r, f, P^\theta)$

22 Calcular o valor de θ utilizando as Eq.(5.123), Eq.(1.24) e Eq.(5.125)

23 ► Matriz P^θ para recuperação do vetor θ

24 $P^\theta \leftarrow [\ \theta_1 \quad \theta_2/2 \quad \theta_3/2 \quad \theta_4/2\ ;\ \theta_2/2 \quad \theta_5 \quad \theta_6/2 \quad \theta_7/2\ ;$

25 $\quad \theta_3/2 \quad \theta_6/2 \quad \theta_8 \quad \theta_9/2\ ;\ \theta_4/2 \quad \theta_7/2 \quad \theta_9/2 \quad \theta_{10}]$

26 ► Melhoria de Política

27 $K^\theta \leftarrow \left(R + B_d^T P^\theta B_d\right)^{-1} B_d^T P^\theta A_d$

28 ► Revitalização dos Estados

29 $x_{i+1} \leftarrow x_{revit}$

30 ► Reinicialização dos Parâmetros do RLS

31 $\theta_{i+1}(0) \leftarrow \theta_i$

32 $\Gamma_{i+1}(0) \leftarrow \Gamma_i$

33 Se $K_k^\theta == K_{k+1}^\theta$

34 ► **Fim do Processo Iterativo**

35 Fim-para

A inicialização do algoritmo IV consiste nas matrizes do sistema dinâmico, ponderações de recompensa instantânea, os parâmetros do processo iterativo, as condições da solução da equação algébrica de*Riccati*, ganho ótimo e parâmetros de inicialização do RLS.

O algoritmo implementa as atualizações do vetor de regressão no passo 19 e a função de valor no passo 17 para cada leitura do intervalo de amostragem. Como também implementa a atualização do vetor θ no passo 21 e da política K^θ no passo 27. Esta sequência de passos ocorre uma janela de tempo chamada de passos de recorrência, mas apenas para inicialização do RLS, e é dada por $n(n + 1)/2$ unidades de tempo.

5.2.1. Convergência RLS para Esquemas HDP

O resultado de convergência para esquemas HDP utilizando aproximadores baseados no RLS tem sido provados por [8] para problemas que envolvam o regulador quadrático linear (LQR). O teorema que será mostrado a seguir pode ser encontrado em [8] e mostra que a sequência $\{K_j\}_{j=1}^{\infty}$ de políticas geradas pelo algoritmo HDP-RLS-DLQR converge para a política ótima.

Teorema: Suponhamos que $\{A, B\}$ é um par controlável, K_0 é um controle estabilizante, e o vetor Φ_k é excitado persistentemente segundo a equação:

$$\varepsilon_0 I \leq \frac{1}{N}\sum_{i=1}^{N} \Phi_{k-i}\Phi_{k-i}^T \leq \bar{\varepsilon}_0 I \tag{5.67}$$

para todo $k \geq N_0,\ N \geq N_0,\ \varepsilon_0 \leq \bar{\varepsilon}_0$, com ε_0 e $\bar{\varepsilon}_0$ inteiros positivos e N_0 uma constante positiva. Então, existe um intervalo de estimação $N < \infty$ de modo que o esquema de iteração de política aproximada descrita anteriormente, gera uma sequência $\{K_j\}_{j=1}^{\infty}$de controles estabilizantes, tal que

$$\lim_{j\to\infty} \left\| K_j - K^* \right\| = 0 \tag{5.68}$$

sendoK^* a matriz de controle de realimentação ótima.

A falta de excitação persistente pode ser entendida, no domínio do tempo quando a matriz de autocorrelação$\Lambda(N)$ é singular. Uma definição equivalente é assumir que o vetor Φ_k apresenta excitação persistente quando sua matriz de autocorrelação, $\Lambda(N)$, for definida positiva. Neste projeto de HDP utilizando o RLS, uma forma de solucionar o problema da falta de excitação persistente é usar a revitalização de estados, assim os estados são reinicializados a cada intervalo de recorrência. Esses procedimentos conduzem a uma matriz $\Lambda(N)$ cujos autovalores são positivos.

Uma medida que representa o bom condicionamento da matriz $\Lambda(N)$ pode ser dada pelo fator de positividade p, que é definido por:

$$p = 1 - L^T(N)\Phi_k \tag{5.69}$$

Usando a definição do vetor ganho e o fato de que $\Lambda(N)$ é simétrica, isto é $\Lambda^T(N) = \Lambda(N)$, então a Eq.(5.69) pode ser reescrita por

$$p = 1 - \Phi_k^T \Lambda^{-1}(N)\Phi_k \tag{5.70}$$

Substituindo-se a Eq.(5.63) na Eq.(5.70), temos após algumas simplificações:

$$p = \frac{\mu}{\mu + \Phi_k^T \Lambda(N-1)\Phi_k} \tag{5.71}$$

Observando que a forma simétrica na Eq.(5.71) possui um valor real não-negativo, tem-se $0 < p \leq 1$. Portanto, uma condição necessária para que a matriz de autocorrelação seja definida positiva é dada por $0 < p \leq 1$. A ocorrência de um valor fora desta faixa indica que a matriz de autocorrelação perdeu a propriedade positiva definida.

CAPÍTULO 6

AVALIAÇÃO DE DESEMPENHO DO ALGORITMO

O algoritmo de programação dinâmica heurística (HDP) que utiliza o estimador linear baseado no método dos mínimos quadrados recursivos (RLS) e aplicado no problema do regulador quadrático linear discreto (DLQR), proposto no capítulo anterior será avaliado por meios de procedimentos que quantificam o tempo e exatidão para a obtenção da política ótima de controle. Neste capítulo é aplicado ao algoritmo desenvolvido, o modelo do sistema dinâmico de quarta ordem estudado no início do capítulo anterior. As metodologias de convergência das matrizes de ponderação Q e R e as análises dos resultados, são obtidos por meio das simulações computacionais desenvolvidas na plataforma MATLAB.

O modelo do sistema dinâmico e a inicialização dos experimentos computacionais constituem a primeira parte do procedimento para avaliação do desempenho do algoritmo para o projeto de controladores via HDP.

A segunda parte do procedimento consiste de comparações que avaliam a exatidão das (decisões) soluções dos controladores, o resultado do algoritmo proposto é confrontado com o resultado obtido pelo método de *Schur* e pela recorrência de *Riccati*.

A terceira parte consiste em fazer testes e análises que são realizadas para avaliar a sensibilidade da heurística para varrer o plano Z, por meio de variações nas matrizes de ponderação do projeto DLQR. O traço da matriz de malha fechada é utilizado para avaliar o desempenho da alocação da autoestrutura.

6.1. Inicialização do Processo Iterativo

O modelo de sistema dinâmico que será utilizado para avaliação do algoritmo HDP-RLS-DLQR será o circuito elétrico de quarta ordem estudado no capítulo anterior, cujo modelo contínuo é dado por:

$$\dot{x}(t) = \begin{bmatrix} -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ -1 & 0 & -1 & 1 \\ 0 & -1 & 1 & -1 \end{bmatrix} x(t) + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} u(t) \tag{6.1}$$

As variáveis de estado são as tensões nos capacitores (x_1 e x_2) e correntes nos indutores (x_3 e x_4). A discretização do modelo é obtida pelo método do segurador de ordem zero (*zero orderhold* –zoh) com intervalo de amostra de $0.8\ s$. O modelo discreto é mostrado abaixo:

$$x_{k+1} = \begin{bmatrix} 0.305 & -0.037 & 0.358 & -0.136 \\ -0.037 & 0.305 & 0.136 & -0.358 \\ -0.358 & -0.136 & 0.441 & -0.321 \\ 0.136 & 0.358 & -0.321 & 0.441 \end{bmatrix} x_k + \begin{bmatrix} 0.191 & -0.046 \\ 0.046 & -0.191 \\ 0.549 & -0.182 \\ -0.182 & 0.549 \end{bmatrix} u_k \tag{6.2}$$

O modelo discreto mostrado na Eq.(6.2) será utilizado para o projeto do controlador ótimo. Para a composição do setup iterativo, devemos informar os seguintes parâmetros para o algoritmo:

- Ordem do Sistema: 4
- Intervalo de Amostragem: 0.8s
- Número de Iterações: 500
- Fator de Recorrência: 10

A ordem do sistema é um dado de entrada, pois na elaboração do produto de *Kronecker*, o produto entre os valores dos estados varia de acordo com a ordem do sistema. O intervalo de amostragem é necessário para a discretização das matrizes contínuas do espaço de estados do sistema dinâmico e deve ser adequadamente determinado. O número de iterações serve como uma condição de segurança para que o algoritmo não entre em laço infinito, ou seja,

não atinja a convergência para a política ótima durante os testes. O fator de recorrência varia em função da ordem do sistema, sendo de fundamental importância para a inicialização do método dos mínimos quadrados recursivos.

Após a definição dos parâmetros de inicialização do algoritmo, devemos fornecer os parâmetros para o RLS que faz parte do núcleo do algoritmo HDP-RLS-DLQR. Os parâmetros são:

- Matriz Inicial da Equação de *Riccati/Lyapunov*.
- Vetor de Parâmetros θ
- Fator de Esquecimento

A inicialização da matriz deve corresponder a ordem do sistema, logo:

$$P_{HDP} = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix} \tag{6.3}$$

O vetor de parâmetros também deve seguir a ordem do sistema, o número de elementos deve ser calculado pela expressão $n(n+1)/2$ onde n é a ordem do sistema

$$\theta = [\theta_1 \quad \theta_2 \quad \theta_3 \quad \theta_4 \quad \theta_5 \quad \theta_6 \quad \theta_7 \quad \theta_8 \quad \theta_9 \quad \theta_{10}] \tag{6.4}$$

. A inicialização do vetor de parâmetros θ para este algoritmo é dada por:

$$\theta = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0] \tag{6.5}$$

O fator de esquecimento é utilizado como estratégia de convergência para o método RLS, e neste algoritmo assumirá dois valores que serão definidos nos próximos tópicos. Vale ressaltar que o fator de esquecimento pode assumir valores entre 0.9 e 1.

Outro parâmetro importante na inicialização do algoritmo que deve ser citado é a condição inicial do sistema, e os estados de revitalização para o

algoritmo. Os valores mostrados abaixo para a condição inicial e os estados de revitalização foram estimados através da simulação do circuito elétrico com os parâmetros (valores dos componentes) utilizados neste projeto. Os valores para x_0 e x_{revit} são:

$$x_0 = [0.22 \quad 0.22 \quad 0.0005 \quad 0.0005] \tag{6.6}$$

$$x_{revit} = [0.5 \quad 0.5 \quad 0.002 \quad 0.002] \tag{6.7}$$

As matrizes de ponderação Q e R da função de utilidade associada ao DLQR são definidas através de heurísticas, mas seguem uma metodologia para variação e avaliação do algoritmo dada em [31]. As matrizes utilizadas para inicialização são:

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.8}$$

$$R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{6.9}$$

Considerando as matrizes A_d e B_d do sistema multivariável dado na Eq.(6.2), e as matrizes Q e R como identidades (Eq.(6.8) e Eq.(6.9)) para fins de recorrência, tem-se que a solução de *Riccati* pelo método de *Schur* é dada por:

$$P_{SCHUR} = \begin{bmatrix} 1.2940 & 0.1123 & -0.0479 & 0.1105 \\ 0.1123 & 1.2940 & -0.1105 & 0.0479 \\ -0.0479 & -0.1105 & 1.3663 & -0.2962 \\ 0.1105 & 0.0479 & -0.2962 & 1.3663 \end{bmatrix} \tag{6.10}$$

Consequentemente a política ótima é dada por:

$$K_{SCHUR} = \begin{bmatrix} -0.1466 & -0.1273 & 0.3020 & -0.2240 \\ 0.1273 & 0.1466 & -0.2240 & 0.3020 \end{bmatrix} \tag{6.11}$$

6.2. Experimentos HDP-RLS

A análise dos resultados obtidos pelo algoritmo baseado em programação dinâmica, especificamente HDP, para o DLQR do sistema MIMO de quarta ordem será mostrada nesta seção. Análises sobre os aspectos de convergência do algoritmo em relação a número de iterações e a relação dos valores das matrizes Q e R com alocação de autovalores no plano Z serão feitas.

Os aspectos de estabilidade na implementação da programação dinâmica aproximada para um sistema realimentado tem sido estudado durante os anos. A análise de convergência QR consiste na avaliação dos autovalores das matrizes Q e R e suas relações com a alocação de autovalores de sistemas multivariáveis no plano Z por meio de controladores ótimos. Os resultados são apresentados na forma de Tabela, seguindo a metodologia estabelecida a partir das equações mostradas no Capítulo 3 (Seção 3.4) [31]. O processo iterativo para variações sistemáticas nas matrizes Q e R da função de custo seguem um padrão de crescimento da matriz Q, enquanto que a matriz R é uma matriz identidade durante todo o processo de solução.

O estimador linear baseado no método dos mínimos quadrados será avaliado através da presença e ausência do fator de esquecimento para análise da convergência do algoritmo.

6.2.1. Algoritmo com fator de esquecimento unitário

Na inicialização do algoritmo, consideramos o fator de esquecimento $\mu = 1$, ou seja, todos os valores estimados possuirão os mesmos pesos (importância) a cada iteração. Para manter a condição de excitação, pode-se usar vários esquemas padrões incluindo a reinicialização dos estados ou injeção de um sinal de ruído branco. Neste algoritmo, a excitação persistente foi implementada a através da revitalização dos estados dada por x_{revit} definido anteriormente e ocorrendo a cada $n(n+1)/2$ intervalos de tempo. Se não for considerada a excitação persistente o algoritmo RLS perde a estabilidade e não consegue atingir a condição de convergência.

O setup desta simulação utilizou todos os parâmetros definidos na seção 6.1. A seguir temos os estados e as entradas de controle do sistema dinâmico em relação ao tempo, ou seja, na medida em que as iterações do algoritmo vão acontecendo. Percebemos que os estados possuem valores oscilatórios, o que caracteriza a reinicialização dos estados através de x_{revit} e condição de excitação persistente para o RLS.

Na Figura 13 é apresentada a evolução dos estados, e a evolução das ações de controle para o sistema multivariável de quarta ordem é mostrada na Figura 14.

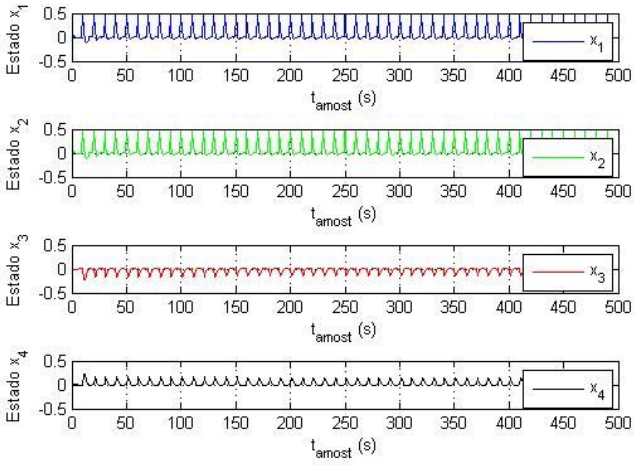

FIGURA 13 – Trajetórias dos Estados x_k com reinicialização por HDP (1º caso).

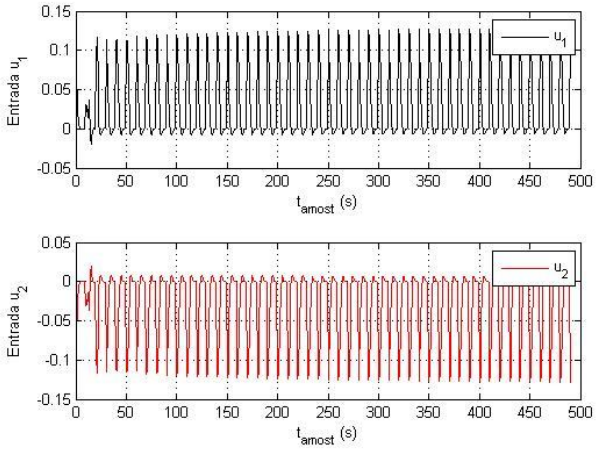

FIGURA 14 – Ação de Controle u_k por HDP (1º caso).

Na Figura 15 é mostrado o coeficiente de condicionamento ω do estimador RLS, que é dado através da seguinte expressão:

$$\omega_j(i) = \frac{\lambda_{maximo}[\Gamma_j^{-1}(i)]}{\lambda_{minimo}[\Gamma_j^{-1}(i)]} \tag{6.12}$$

As matrizes com o coeficiente de condicionamento ω muito alto possuem também uma dispersão muito grande nos valores de seus elementos. Esse fato pode trazer problemas computacionais na solução de sistemas de equações envolvendo Γ_j^{-1} ou Γ_j. Por este motivo dizemos que uma matriz com ω grande é uma matriz mal condicionada.

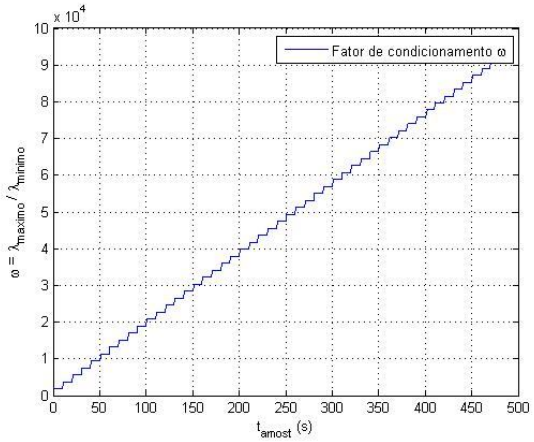

FIGURA 15 – Fator de condicionamento ω (1º caso).

Abaixo é mostrada a convergência dos parâmetros do crítico pelo método HDP. Percebemos que os parâmetros do crítico não convergem para a solução P de *Riccati*, ou seja, com este número de iterações não é possível atingir a política ótima, fato que pode ser comprovado pelo valor elevado do fator de condicionamento da matriz utilizada no RLS, e pela ausência do fator de esquecimento.

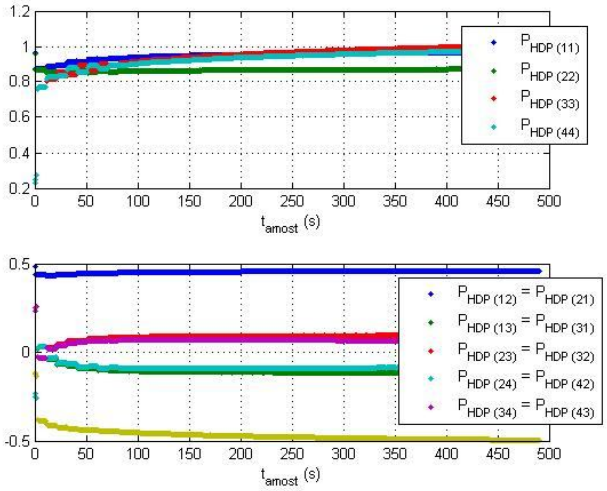

FIGURA 16 – Convergência dos parâmetros P do crítico por HDP (1º caso).

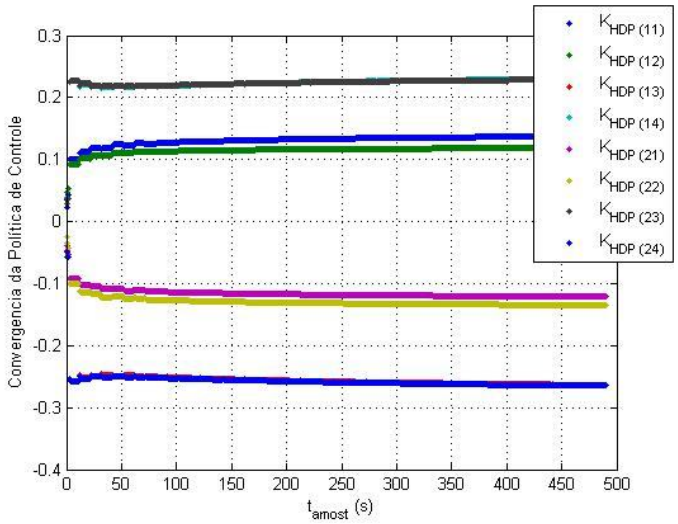

FIGURA 17 – Convergência da política ótima$K$por HDP (1º caso).

Os gráficos da convergência da ação são mostrados abaixo, onde os elementos K_{ij} representam os elementos da matriz da política ótima.

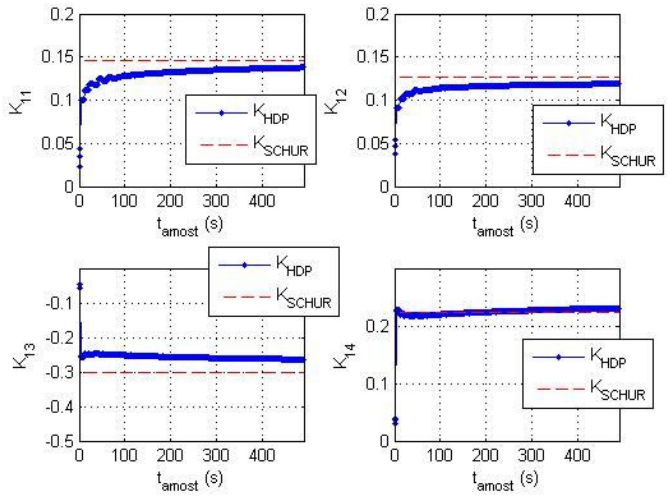

FIGURA 18 – Elementos da política ótima por HDP (primeira linha)(1º caso).

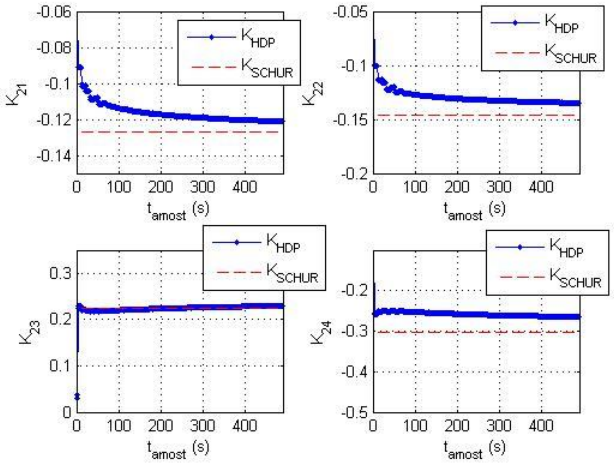

FIGURA 19 – Elementos da política ótima por HDP (segunda linha)(1º caso).

Observando as Figuras 18 e 19 fica mais clara a percepção que o algoritmo não atinge a política ótima, utilizando 500 iterações. Vale ressaltar que, se aumentarmos o número de iterações, o set point é alcançado.

Os autovalores do sistema e o traço da matriz de malhada fechada podem ser verificados a cada iteração de acordo com a figura 20 mostrada abaixo:

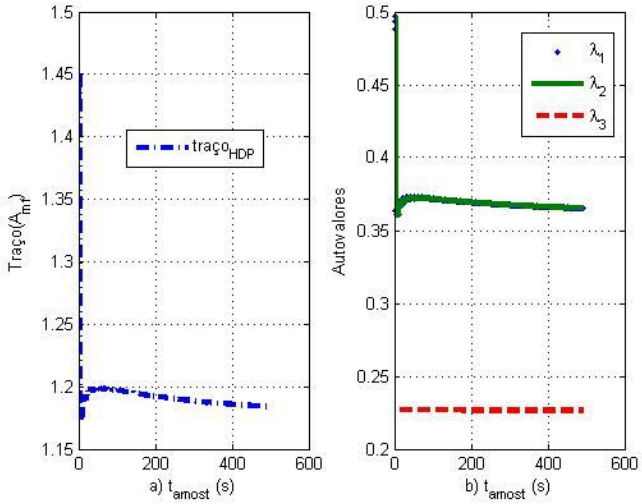

FIGURA 20 – Traço e Autovalores a cada iteração por HDP (1º caso).

Os traços das matrizes de malha fechada são utilizados para avaliar o comportamento do processo de recorrência para mapear as ponderações no plano Z estável. Estes valores podem ser utilizados para comparação de convergência entre algoritmos.

A análise de convergência QR por HDP é mostrada na tabela abaixo. Para esta situação seguiu-se a heurística de $q_i = \{-2,1,0,1,-2\}$.

TABELA 2 – Convergência QR (1º caso).

$Q(q_i)$	$R(r_i)$	Autovalores λ
2	0	0.02600.0519 0.2709 0.2639
1	0	0.20200.2020 0.1965 0.1965
0	0	0.36540.3654 0.2267 0.2267
-1	0	0.48740.4874 0.2312 0.2312
-2	0	0.51400.5140 0.2320 0.2320

Observamos que as variações na matriz Q, conduziram ao mapeamento de pólos reais limitados ao eixo real Z e ao semi plano direito. Uma investigação para mapeamento em outras regiões do plano Z envolve outras heurísticas para variações de Q e R. Verifica-se também que se obtém solução mais próxima da origem para valores acima de $q_i > 0$. Para valores de $q_i < 0$, os autovalores tendem a ficarem mais distantes da origem e as iterações aumentam.

A motivação para a implementação do RLS para estimação da função valor no problema do DLQR foi motivada pelo insucesso da estimação através do método dos mínimos quadrados em batelada (LS), pois o mesmo, utilizando os mesmos parâmetros que foram usados para inicialização deste algoritmo, a matriz de regressores do método LS estava mal condicionada, impossibilitando a estimação dos parâmetros θ.

6.2.1. Algoritmo com fator de esquecimento variável

Neste segundo caso, para a simulação é considerado como parâmetro de entrada no setup inicial, o fator de esquecimento. O valor do fator de esquecimento é alterado de 1 para 0.975, que está dentro do intervalo estipulado na teoria para este fator ($0.9 < \mu \leq 1$). A trajetória dos estados no decorrer do tempo pode ser vista na figura abaixo.

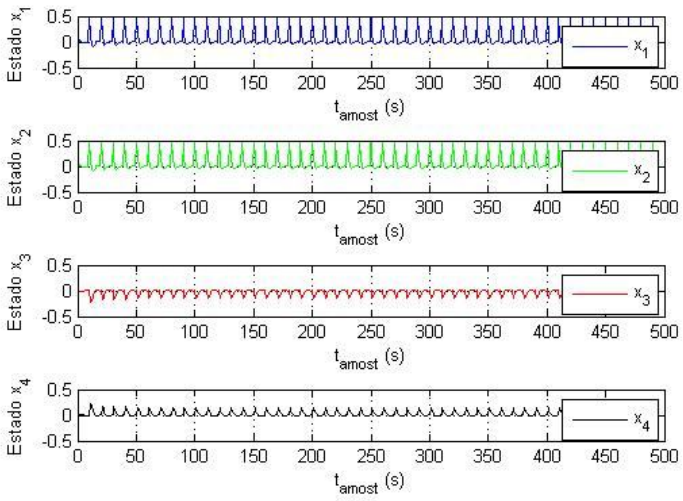

FIGURA 21 – Trajetórias dos Estados x_k com reinicialização por HDP (2º caso).

A figura a seguir mostra a evolução da ação de controle ao longo tempo (iterações do algoritmo). Comparando as Figuras 21 e 22 com as Figuras 13 e 17 do algoritmo que não utiliza o fator de esquecimento, percebemos que os gráficos são bem próximos devido à revitalização dos estados.

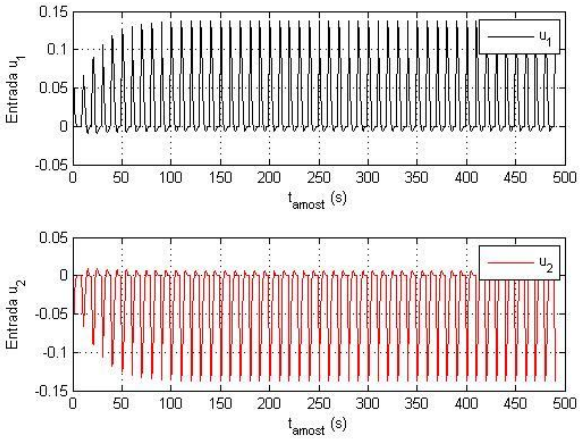

FIGURA 22– Ação de Controle u_k por HDP (2º caso).

Um ponto importante e que vale ser destacado para este segundo caso, é que o fator de condicionamento da matriz do RLS reduziu consideravelmente depois da inserção no fator de esquecimento no algoritmo. Isto sinaliza um bom condicionamento da matriz e pode ser visto a seguir.

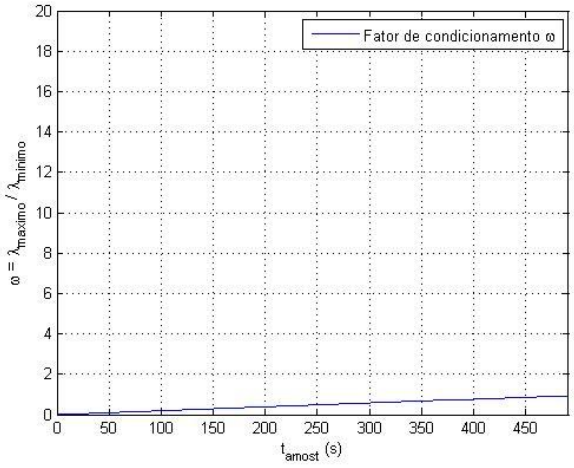

FIGURA 23– Fator de condicionamento ω (2º caso).

Os parâmetros de P são mostrados na Figura 24, e percebemos que os elementos da matriz convergem para os valores da solução de *Schur*a partir da 50ª iteração, representando assim a eficiência do algoritmo em relação ao anterior.

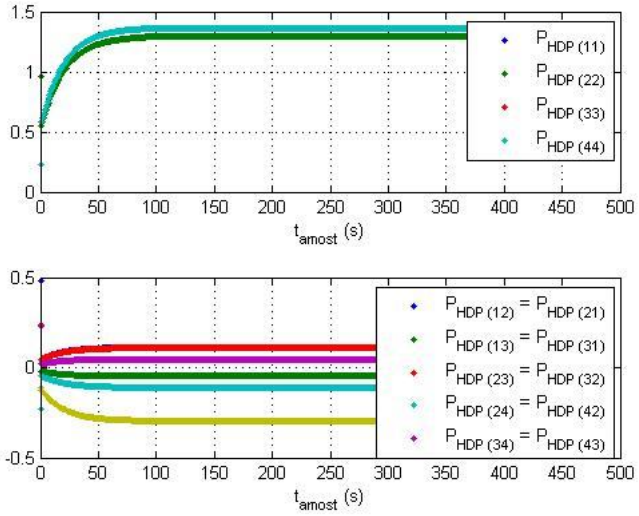

FIGURA 24 – Convergência dos parâmetros P do crítico por HDP (2º caso).

Para o esquema HDP, a política de controle é encontrada a partir da matriz P. Como a matriz P começa atinge a convergência próxima a 50ª iteração, a política ótima K segue o mesmo comportamento.

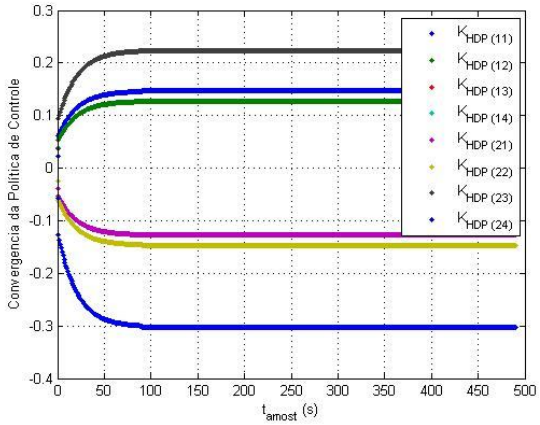

FIGURA 25 - Convergência da política ótima K por HDP (2º caso).

A seguir é mostrada a evolução dos elementos individuais e a solução de *Schur* para a política de controle K.

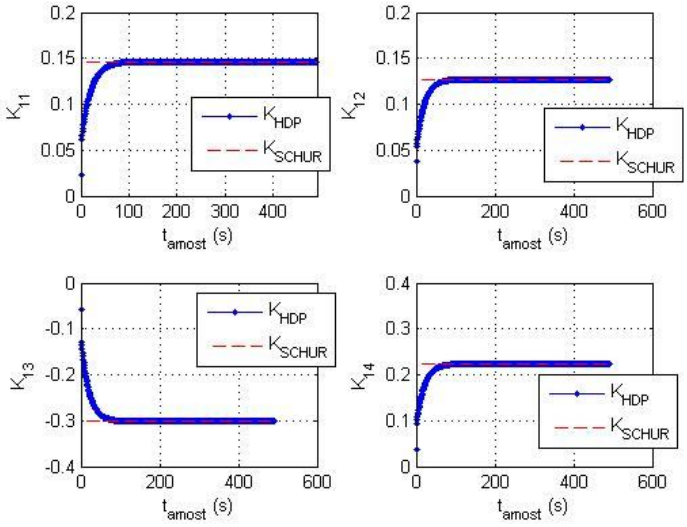

FIGURA 26 - Elementos da política ótima por HDP (primeira linha)(2º caso).

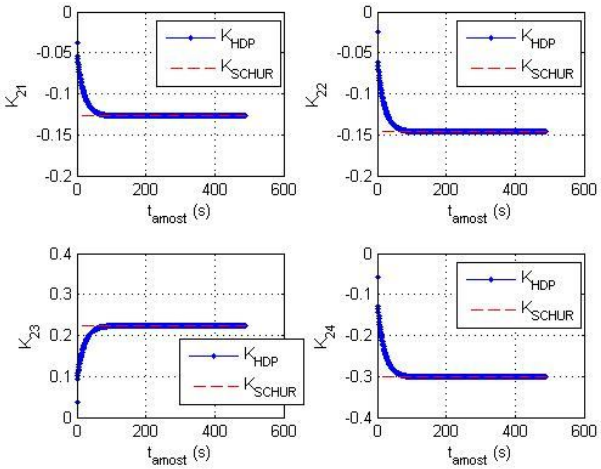

FIGURA 27 – Elementos da política ótima por HDP (segunda linha)(2º caso).

Observando as Figuras 26 e 27 fica clara a percepção de que o algoritmo atinge a política ótima, utilizando 50 iterações. Vale ressaltar que, para as 500 iterações consideradas, o algoritmo converge com a solução de *Schur* em todas as casas decimais consideradas.

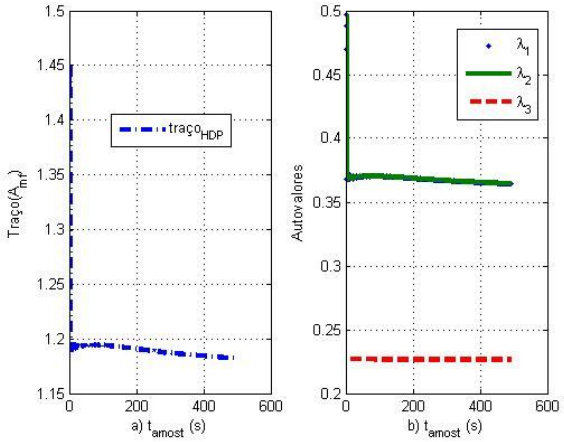

FIGURA 28 - Traço e Autovalores a cada iteração por HDP (2º caso).

Observando o traço da matriz de malha fechada e o gráfico dos autovalores, percebemos que não houve uma grande mudança, pois os autovalores são bem próximos nos dois algoritmos.

A análise de convergência QR por HDP considerando a presença do fator de esquecimento é exposta na tabela abaixo. Para esta seguiu-se a heurística da escolha de $q_i = \{-2, -1, 0, 1, 2\}$

TABELA 3 – Convergência QR (2º caso).

$Q(q_i)$	$R(r_i)$	Autovalores λ		
2	0	0.15720.1671	0.0180	0.0245
1	0	0.19360.1936	0.1760	0.1760
0	0	0.35590.3559	0.2188	0.2188
-1	0	0.48510.4851	0.2303	0.2303
-2	0	0.51230.5123	0.2316	0.2316

Verifica-se que se obtém solução mais próxima da origem para valores acima de $q_i > 0$. Para valores de $q_i < 0$, os autovalores tendem a ficarem mais distantes da origem e as iterações aumentam.

As simulações dos algoritmos mostradas neste capítulo com as duas varrições considerando o fator de esquecimento, validam a teoria apresentada e mostram que o algoritmo que considera a presença do fator de esquecimento

torna-se mais robusto em relação ao outro que considera apenas o RLS padrão, e atinge a convergência de forma rápida e suave (sem muitas variações na busca da solução).

CAPÍTULO 7

CONCLUSÃO

Nesta dissertação foi apresentado o projeto de um controlador ótimo baseado no regulador quadrático linear discreto (DLQR) através da programação dinâmica adaptativa, mas precisamente utilizando o esquema da programação dinâmica heurística, e utilizando para estimação da função valor, o aproximador linear dos mínimos quadrados recursivos (RLS). Tópicos importantes utilizados nesta dissertação e que sevem de apoio para esta metodologia é a diferença temporal, processo de decisão *markoviano* e a técnicas de aproximação da função valor.

A programação dinâmica aproximada é responsável pela viabilização da sintonia da política ótima de controle e da solução por programação dinâmica com avanço no tempo. Portanto, com a junção de todas estas metodologias, este trabalho é caracterizado como um controle ótimo direto.

O método heurístico baseado na programação dinâmica adaptativa foi apresentado desde sua formulação pela abordagem crítica adaptativa até as avaliações do algoritmo com a presença do estimador RLS aplicado em sistemas multivariáveis, neste caso um circuito de quarta que teve toda a sua formulação matemática apresentada no decorrer deste trabalho. O algoritmo de HDP foi formulado como uma aproximação da função valor para uma dada política admissível. A aproximação da função valor foi realizada pelo método dos mínimos quadrados recursivos, como já foi dito anteriormente, e apoiada pela formulação da teoria da vetorização de matrizes e álgebra de *Kronecker*. A eficiência do algoritmo seguiu a metodologia para determinação dos coeficientes da matriz P de *Riccati* e os ganhos do controlador DLQR ótimo.

Os experimentos computacionais analisados mostraram as soluções alcançadas pelo algoritmo em relação ao método de *Schur* para o sistema dinâmico multiariável representado por um circuito elétrico de quarta ordem. A análise de convergência pelo número de iterações e pela sintonia das matrizes Q e R foram apresentadas. Com a elaboração e simulação do algoritmo fica claro que para métodos HDP, a dependência da dinâmica do sistema e que a determinação do ganho do controlador é obtida de maneira explícita.

O algoritmo com o RLS apresentado mostra a robustez do estimador através das simulações, pois primeiramente foi feita a tentativa de implementação da aproximação da função valor com o estimador dos mínimos quadrados em batelada, que obteve sucesso apenas para sistema até terceira ordem, e a partir dos sistemas de quarta ordem a matriz de regressores perdia a consistência ao longo das iterações e o estimador não atingia os objetivos. Já com a utilização do RLS foi mostrada dois casos com ausência e presença do fator de esquecimento, onde com apenas o uso do RLS já foi obtida a convergência do algoritmo mas não para a solução de *Schur*. E com o fator de esquecimento o algoritmo convergiu de maneira eficaz para a solução ótima.

De maneira geral, a programação dinâmica adaptativa nos mostra que dentro de suas limitações, que seus processos são de soluções viáveis e com suas soluções aplicáveis na prática. Portanto fica claro para ADP que obtendo apenas uma pequena informação dos estados do sistema, em momentos específicos é possível obter a solução de Riccati online, assim é possível a implementação de um controlador ótimo discreto baseado em programação dinâmica adaptativa.

7.1. Trabalhos Futuros

Os trabalhos futuros que serão citados abaixo, seguem alguns estudos e desenvolvimentos com foco em programação dinâmica adaptativa que foram abordados neste trabalho de dissertação. A pesquisa pode seguir os seguintes desenvolvimentos e investigações:

- Aplicar o estimador RLS na programação dinâmica Heurística Dependente de Ação (ADHDP).
- Desenvolver o estudo para aplicação prática dos outros métodos de programação dinâmica adaptativa (DHP e ADDHP).
- Aplicar outras técnicas para aproximação da função valor através da teoria de programação neurodinâmica e lógica *Fuzzy*.
- Verificar outros métodos para sintonia das matrizes de ponderação Q e R investigando a sua convergência com o algoritmo ADP.
- Analisar o impacto da maldição da dimensionalidade através da relação de quantização de entradas e estados para sistemas dinâmicos multivariáveis.

APÊNDICE A

MODELAGEM MATEMÁTICA DA PLANTA

Para a análise e controle de sistemas dinâmicos multivariáveis, foi escolhido para este trabalho um circuito elétrico que representa um sistema MIMO de quarta ordem, através da presença de quatro elementos armazenadores de energia (dois capacitores e dois indutores). Observando a figura seguinte do circuito elétrico, podemos perceber que as fontes de tensão $u_1(t)$ e $u_2(t)$ representam as entradas do sistema, e as tensões nos capacitores C_1 e C_2 dadas por V_{C1}e V_{C2}, respectivamente, representam as saídas do sistema, caracterizando assim, um sistema MIMO composto por duas entradas e duas saídas.

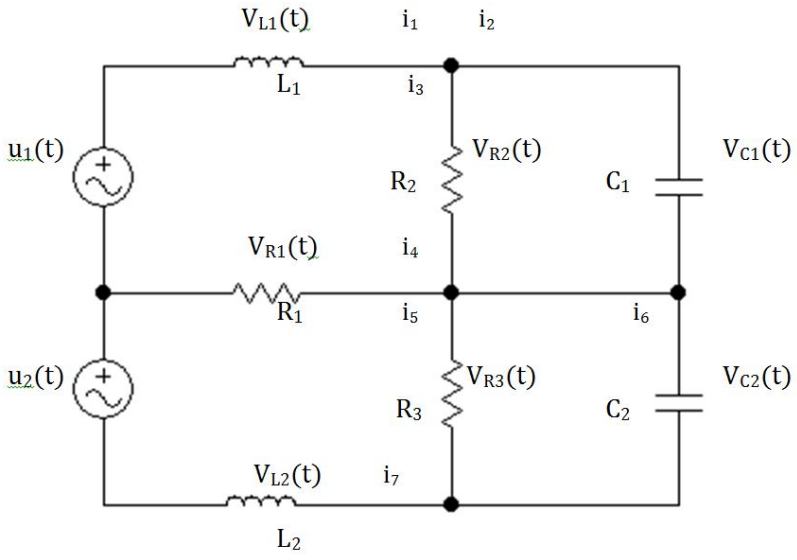

FIGURA A.1 – Circuito Elétrico de Quarta Ordem.

Como temos um sistema linear e invariante no tempo, todas as propriedades vistas anteriores são aplicáveis a este circuito. Logo, para a obtenção do modelo matemático do sistema dinâmico devemos utilizar o teorema da superposição.

Aplicando o teorema da superposição ao circuito da figura A, temos a figura abaixo com a fonte de tensão $u_2(t)$ curto-circuitada.

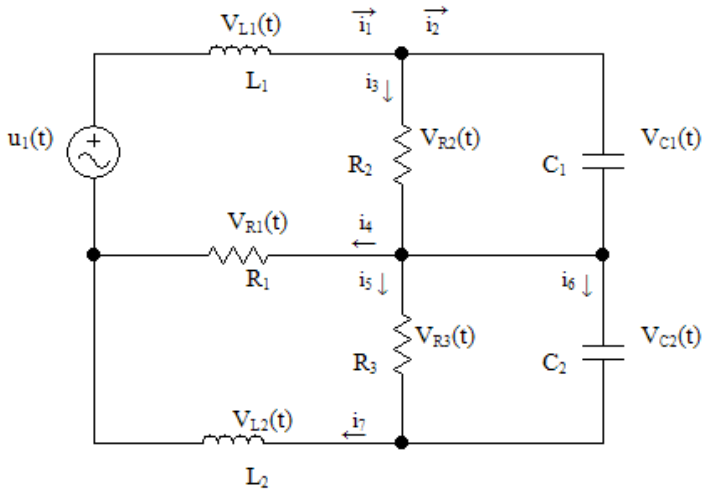

FIGURA A.2 – Circuito Elétrico de Quarta Ordem ($u_2(t) = 0$).

Utilizando a lei das tensões de *Kirchhoff,* temos:

$$u_1(t) = V_{L1}(t) + V_{R2}(t) + V_{R1}(t) \tag{A.1}$$

Utilizando a lei das correntes de *Kirchhoff,* temos:

$$i_1(t) = i_2(t) + i_3(t) \tag{A.2}$$

$$i_2(t) + i_3(t) = i_4(t) + i_5(t) + i_6(t) \tag{A.3}$$

Sabemos que a tensão no Indutor 1(V_{L1}) é dada por:

$$V_{L1}(t) = L_1 \frac{di_1(t)}{dt} \tag{A.4}$$

As correntes $i_3(t)$ e $i_2(t)$, são dadas por:

$$i_3(t) = \frac{V_{R2}(t)}{R_2} = \frac{V_{C1}(t)}{R_2} \tag{A.5}$$

$$i_2(t) = C_1 \frac{dV_{C1}(t)}{dt} \tag{A.6}$$

Substituindo as Eq.(A.5) e Eq.(A.6) na Eq.(A.2), temos:

$$i_1(t) = \frac{V_{C1}(t)}{R_2} + C_1 \frac{dV_{C1}(t)}{dt} \tag{A.7}$$

Substituindo a Eq.(A.7) na Eq.(A.4):

$$V_{L1}(t) = L_1 \frac{di_1(t)}{dt} = \frac{L_1}{R_2}\frac{dV_{C1}(t)}{dt} + L_1 C_1 \frac{d^2V_{C1}(t)}{dt^2} \tag{A.8}$$

As correntes $i_5(t)$ e $i_6(t)$, são dadas por:

$$i_5(t) = \frac{V_{R3}(t)}{R_3} = \frac{V_{C2}(t)}{R_3} \tag{A.9}$$

$$i_6(t) = C_2 \frac{dV_{C2}(t)}{dt} \tag{A.10}$$

Isolando $i_4(t)$ na Eq.(A.3), temos:

$$i_4(t) = i_2(t) + i_3(t) - i_5(t) - i_6(t) \tag{A.11}$$

Substituindo as Eq.(A.5), Eq.(A.6), Eq.(A.9) e Eq.(A.10) na Eq.(A.11), temos:

$$i_4(t) = \frac{V_{C1}(t)}{R_2} + C_1\frac{dV_{C1}(t)}{dt} - \frac{V_{C2}(t)}{R_3} - C_2\frac{dV_{C2}(t)}{dt} \quad \text{(A.12)}$$

A tensão no Resistor 1(V_{R1}) é dada por:

$$V_{R1}(t) = R_1 i_4(t) \quad \text{(A.13)}$$

Substituindo a Eq.(A.12) na Eq.(A.13):

$$V_{R1}(t) = R_1\left(\frac{V_{C1}(t)}{R_2} + C_1\frac{dV_{C1}(t)}{dt} - \frac{V_{C2}(t)}{R_3} - C_2\frac{dV_{C2}(t)}{dt}\right) \quad \text{(A.14)}$$

$$V_{R1}(t) = \frac{R_1}{R_2}V_{C1}(t) + R_1C_1\frac{dV_{C1}(t)}{dt} - \frac{R_1}{R_3}V_{C2}(t) - R_1C_2\frac{dV_{C2}(t)}{dt} \quad \text{(A.15)}$$

Substituindo as Eq.(A.8), Eq.(A.5) e Eq.(A.15) na Eq.(A.1):

$$\begin{aligned} u_1(t) = {} & \frac{L_1}{R_2}\frac{dV_{C1}(t)}{dt} + L_1C_1\frac{d^2V_{C1}(t)}{dt^2} + V_{C1}(t) + \frac{R_1}{R_2}V_{C1}(t) \\ & + R_1C_1\frac{dV_{C1}(t)}{dt} - \frac{R_1}{R_3}V_{C2}(t) - R_1C_2\frac{dV_{C2}(t)}{dt} \end{aligned} \quad \text{(A.16)}$$

Sabemos que $V_{C1}(t) = y_1(t)$ e $V_{C2}(t) = y_2(t)$, e substituindo na Eq.(A.16) temos:

$$\begin{aligned} u_1(t) = {} & \frac{L_1}{R_2}\frac{dy_1(t)}{dt} + L_1C_1\frac{d^2y_1(t)}{dt^2} + y_1(t) + \frac{R_1}{R_2}y_1(t) \\ & + R_1C_1\frac{dy_1(t)}{dt} - \frac{R_1}{R_3}y_2(t) - R_1C_2\frac{dy_2(t)}{dt} \end{aligned} \quad \text{(A.17)}$$

Aplicando a Transformada de Laplace na Eq.(A.17), temos:

$$U_1(s) = \frac{L_1}{R_2}sY_1(s) + L_1C_1s^2Y_1(s) + Y_1(s) + \frac{R_1}{R_2}Y_1(s) + R_1C_1sY_1(s) - \frac{R_1}{R_3}Y_2(s) - R_1C_2sY_2(s) \quad \text{(A.18)}$$

$$U_1(s) = \left(\frac{L_1}{R_2}s + L_1C_1s^2 + 1 + \frac{R_1}{R_2} + R_1C_1s\right)Y_1(s) - \left(\frac{R_1}{R_3} + R_1C_2s\right)Y_2(s) \quad \text{(A.19)}$$

Aplicando o teorema da superposição ao circuito da figura A.1, temos a figura abaixo com a fonte de tensão $u_1(t)$ curto-circuitada. O desenvolvimento das equações para este caso é bem próximo do que foi apresentado anteriormente coma fonte $u_2(t)$ curto-circuitada, e segue a mesma linha de raciocínio.

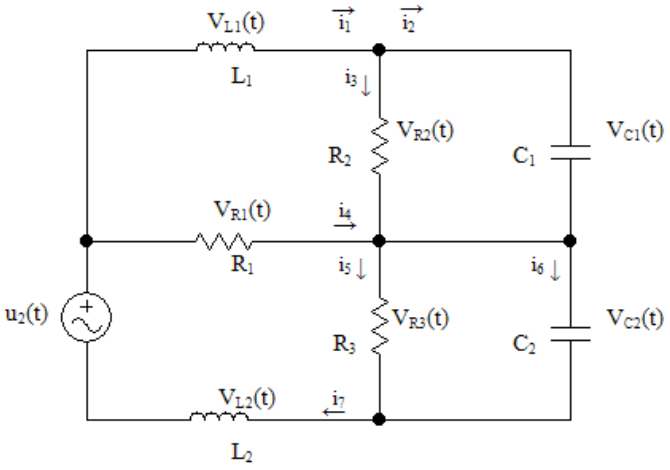

FIGURA A.3 – Circuito Elétrico de Quarta Ordem ($u_1(t) = 0$).

Utilizando a lei das tensões de *Kirchhoff,* temos:

$$u_2(t) = V_{R1}(t) + V_{R3}(t) + V_{L2}(t) \quad \text{(A.20)}$$

Utilizando a lei das correntes de *Kirchhoff,* temos:

$$i_7(t) = i_5(t) + i_6(t) \quad \text{(A.21)}$$

$$i_5(t) + i_6(t) = i_4(t) + i_2(t) + i_3(t) \tag{A.22}$$

Sabemos que a tensão no Indutor 2(V_{L2}) é dada por:

$$V_{L2}(t) = L_2 \frac{di_7(t)}{dt} \tag{A.23}$$

As correntes $i_5(t)$ e $i_6(t)$, são dadas por:

$$i_5(t) = \frac{V_{R3}(t)}{R_3} = \frac{V_{C2}(t)}{R_3} \tag{A.24}$$

$$i_6(t) = C_2 \frac{dV_{C2}(t)}{dt} \tag{A.25}$$

Substituindo as Eq.(A.24) e Eq.(A.25) na Eq.(A.21), temos:

$$i_7(t) = \frac{V_{C2}(t)}{R_3} + C_2 \frac{dV_{C2}(t)}{dt} \tag{A.26}$$

Substituindo a Eq.(A.26) na Eq.(A.23), temos:

$$V_{L2}(t) = L_2 \frac{di_7(t)}{dt} = \frac{L_2}{R_3}\frac{dV_{C2}(t)}{dt} + L_2C_2 \frac{d^2V_{C2}(t)}{dt^2} \tag{A.27}$$

Isolando $i_4(t)$ na Eq.(A.22):

$$i_4(t) = i_5(t) + i_6(t) - i_2(t) - i_3(t) \tag{A.28}$$

como:

$$i_3(t) = \frac{V_{R2}(t)}{R_2} = \frac{V_{C1}(t)}{R_2} \tag{A.29}$$

$$i_2(t) = C_1 \frac{dV_{C1}(t)}{dt} \tag{A.30}$$

Substituindo as Eq.(A.24), Eq.(A.25), Eq.(A.29) e Eq.(A.30) na Eq.(A.28), temos:

$$i_4(t) = \frac{V_{C2}(t)}{R_3} + C_2\frac{dV_{C2}(t)}{dt} - \frac{V_{C1}(t)}{R_2} - C_1\frac{dV_{C1}(t)}{dt} \tag{A.31}$$

A tensão no Resistor 1(V_{R1}) é dada por:

$$V_{R1}(t) = R_1 i_4(t) \tag{A.32}$$

Substituindo a Eq.(5.31) na Eq.(5.32):

$$V_{R1}(t) = R_1\left(\frac{V_{C2}(t)}{R_3} + C_2\frac{dV_{C2}(t)}{dt} - \frac{V_{C1}(t)}{R_2} - C_1\frac{dV_{C1}(t)}{dt}\right) \tag{A.33}$$

$$V_{R1}(t) = \frac{R_1}{R_3}V_{C2}(t) + R_1C_2\frac{dV_{C2}(t)}{dt} - \frac{R_1}{R_2}V_{C1}(t) - R_1C_1\frac{dV_{C1}(t)}{dt} \tag{A.34}$$

Substituindo as Eq.(A.27), Eq.(A.24) e Eq.(A.34) na Eq.(A.20):

$$\begin{aligned} u_2(t) = {} & \frac{R_1}{R_3}V_{C2}(t) + R_1C_2\frac{dV_{C2}(t)}{dt} - \frac{R_1}{R_2}V_{C1}(t) - R_1C_1\frac{dV_{C1}(t)}{dt} \\ & + V_{C2}(t) + \frac{L_2}{R_3}\frac{dV_{C2}(t)}{dt} + L_2C_2\frac{d^2V_{C2}(t)}{dt^2} \end{aligned} \tag{A.35}$$

Sabemos que $V_{C1}(t) = y_1(t)$ e $V_{C2}(t) = y_2(t)$, e substituindo na Eq.(5.35) temos:

$$\begin{aligned} u_2(t) = {} & \frac{R_1}{R_3}y_2(t) + R_1C_2\frac{dy_2(t)}{dt} - \frac{R_1}{R_2}y_1(t) - R_1C_1\frac{dy_1(t)}{dt} \\ & + y_2(t) + \frac{L_2}{R_3}\frac{dy_2(t)}{dt} + L_2C_2\frac{d^2y_2(t)}{dt^2} \end{aligned} \tag{A.36}$$

Aplicando a Transformada de Laplace na Eq.(5.36), temos:

$$U_2(s) = \frac{R_1}{R_3}Y_2(s) + R_1C_2sY_2(s) - \frac{R_1}{R_2}Y_1(s) - R_1C_1sY_1(s) + Y_2(s) + \frac{L_2}{R_3}sY_2(s) + L_2C_2s^2Y_2(s) \tag{A.37}$$

$$U_2(s) = -\left(\frac{R_1}{R_2} + R_1C_1s\right)Y_1(s) + \left(\frac{L_2}{R_3}s + L_2C_2s^2 + \frac{R_1}{R_3} + R_1C_2s + 1\right)Y_2(s) \tag{A.38}$$

REFERÊNCIAS

[1] Richard E.Bellman. *Dynamic programming and stochastic control processes*. Information and Control. 1958.

[2] Richard E.Bellman. *Dynamic Programming*.Dover Publications. 1957.

[3] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control, One Volume*.3rd Edition. Athena Scientific, 2005.

[4] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control, Two Volume*.2nd Edition.Athena Scientific, 1995.

[5] Donald E. Kirk. *Optimal Control Theory: An Introduction*. Prentice-Hall NetworkSeries. Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1970.

[6] Benjamin C. Kuo. *Digital Control Systems*.Harcourt Brace College Publishers, 1980.

[7] Frank L. Lewis and Vassilis L. Syrmos. *Optimal Control*.2nd Edition.John Wiley & Sons. 1995.

[8] Rush D. Robinett III, David G. Wilson, G. Richard Eisler and John E. Hurtado. *Appiled Dynamic Programming for Optimization of Dynamical Systems*.Society for Industrial and Applied Mathematics, Philadelphia. 2005.

[9] Warren B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*.John Wiley & Sons. 1955.

[10] Oleg N. Gasparyan. *Linear and Nonlinear Multivariable Feedback Control: A Classical Approach.*John Wiley & Sons. 2008.

[11] Martino Bardi and ItaloCapuzzo-Dolcetta.Optimal *Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations.*Birkhauser. 1997.

[12] P. Albertosand A. Sala. *Multivaribale Control Systems: Na Engineering Approach.*Springer. 2004.

[13] M. Freimer, *A dynamic programming approach to adaptive control processes,Automatic Control, IRE Transactions on*, vol. 4, no. 2, pp. 10 – 15, Nov. 1959.

[14] K. Hosking, *Dynamic programming and synthesis of linear optimal control systems*,*Electrical Engineers, Proceedings of the Institution of*, vol. 113, no. 6, pp. 1087 –1090, 1966.

[15] D. Corona, A. Giua, and C. Seatzu, *Quantized optimal control of discrete-time systems, in Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on*, vol. 1, 2005, pp. 4 pp. –276.

[16] Gene F. Franklin, J. David Powell and Michael L. Workman. *Digital Control of Dynamic Systems*.3rd Edition.Addison-Wesley. 1998.

[17] Gene Franklin, J. David Powell, Abbas Emami-Naeini. *Feedback Control of Dynamic Systems.*4th Edition.Prentice-Hall. 2002.

[18] Dimitri P. Bertsekas and Tsitsiklis, J. *Neuro-dynamic Programming*. Athena Scientific. 1996.

[19] Brian D. O. Anderson and John B. Moore. *Optimal Control: Linear Quadratic Methods*. Prentice-Hall. 1989.

[20] Shun-ichi Azuma and Toshiharu Sugie. *An Optimal Dynamic Quantization Scheme for Control With Discrete-Valued Input*. *Proceedings of the 2007 American Control Conference*. New York City, USA. 2007.

[21] R. Zoppoli, M. Sanguineti and T. Parisini.*Can We Cope with the Curse of Dimensionality in Optimal Control by Using Neural Approximators?*.*Proceedings of the 40th IEEE Conference on Decision and Control*. Florida, USA. 2001.

[22] M. S. Fadali. *Digital Control Engineering: Analysis and Design.*Academic Press. 2009.

[23] M. Sniedovich. *Dynamic Programming: Foundations and Principles*. 2nd Edition.CRC Press. 1991.

[24] Stuart E. Dreyfus and Averill M. Law.*The Art and Theory of Dynamic Programming.*Academic Press. 1977.

[25] C. Y. Seong and B. Widrow, *Neural dynamic optimization for control systems—Part I: Background,IEEE Trans. Syst., Man, Cybern. B*, vol.31, pp. 482–489, Aug. 2001.

[26] C. Y. Seong and B. Widrow, *Neural dynamic optimization for control systems—Part II: Theory,IEEE Trans. Syst., Man, Cybern. B*, vol. 31, pp. 490–501, Aug. 2001.

[27] C. Y. Seong and B. Widrow,*Neural dynamic optimization for control systems—Part III: Applications, IEEE Trans. Syst., Man, Cybern.*, vol. 31, pp. 502–513, Aug.2001.

[28] S. Skogestad and I. Postlethwaite.*Multivariable Feedback Control: Analysis and Design.*John Wiley & Sons. 1997.

[29] L. Busoniu, R. Babuska, B. De Schutter, and D. Ernst.“*Reinforcement Learning and Dynamic Programming Using Function Approximators”*. Boca Raton, Florida: CRC Press. ISBN: 1439821089, 2010.

[30] T. Landelius. *“Reinforcement Learning and Distributed Local Model Synthesis”*.Ph.D. dissertation. Linköping University, Sweden. SE-581 83 Linkpöping.Sweden, 1997.Dissertation No 469, ISBN 91-7871-892-9.

[31] J. V. Fonseca Neto and L.R. Lopes. *“On the convergence of DLQR control and recurrences of Riccati and Lyapunov in Dynamic Programming.”*In UKSim 13th Internacional Conference on Computer Modelling and Simulation (UKSim2011). Cambridge, United Kingdom.

[32] Lendaris, G. G. *A retrospective on adaptive dynamic programming for control. In: Neural Networks, 2009*. IJCNN 2009.International Joint Conference on.Vol. 0.Pp.1750-1757.

[33] Frank L. Lewis and DragunaVrabie*. Adaptative dynamic programming for feedback control*. In: Asian Control Conference, 2009. ASCC 2009. 7TH. PP. 1402-1409.

[34] Buşoniu, L., R. Babuška B. de Schutter and D. Ernst*. Reinforcement Learning and Dynamic Programming Using Function Approximators.*CRC Press. Boca Raton, Florida. 2010.

[35] Vamvoudakis, K.G. e and F.L. Lewis. *Onlie actor critic algorithm to solve the continuous-time infinite horizon optimal control problem.*In Neural Networks, 2009.IJCNN 2009.International Joint Conference on.Pp.3180-3187.

[36] Brewer, J. *Kroneckerprodutcs and matrix calculus in system theory.* Circuits and Systems, IEEE Transactions on 25(9), 772-781. 1978.

[37] R. Howard. *Dynamic programming and markov processes.* New York: John Wiley and Sons, Inc. 1960.

[38] Si, Jennie, Andrew G. Barto, Warren Buckler Powell e Don Wunsch. *Handbook of Learning and Approximate Dynamic Programming.*IEEE Press Series on Computational Intelligence.Wiley-IEEE Press. 2004.

[39] Wang, Fei-Tue, Huaguang Zhang and Derong Liu. *Adaptive dynamic programming: An introduction. Computational Intelligence Magazine*, IEEE 4(2), 39-47. 2009.

[40]Lendaris, G.G. *A retrospective on adaptive dynamic programming for control.*In: Neural Networks, 2009. IJCNN 2009.International Joint Conference on.Vol. 0. Pp. 1750-1757. 2009.

[41] Werbos, P. J. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis.Harvard University. Cambridge, MA, USA. 1974.

[42] Werbos, P. J. *Foreword – ADP: The key direction for future research in intelligent control and understanding brain intelligence*. Systems, Man and Cybernetics, Part B: Cybernetics, IEEE Transactions on 38(4), 898-900. 2008.

[43] Werbos, P. J. *Neural networks for control and system identification. In: Decision and Control, 1989.*, Proceedings of the 28th IEEE Conference on. Pp. 260-265 vol. 1. 1989.

[44] Werbos, P.J. *Neural networks for control. Chap. A menu of designs for reinforcement learning over time, pp. 67-95*.MIT Press. Cambridge, MA, USA. 1990.

[45]Al-Tamimi, A. e F. Lewis. *Discrete-time nonlinear HJB solution using approximate dynamics programming: Convergence proof. In: Approximate Dynamic Programming and Reinforcement Learning*. ADPRL 2007.IEEE Internatitonal Symposium on.Pp.38-43. 2007.

[46]L. A. Aguirre. *Introdução a Identificação de Sistemas: Técnicas Lineares e Não-Lineares Aplicadas a Sistemas Reais*. Ed. UFMG. 2nd Ed. 2004.

[47] L. Ljung. *System Identification: Theory for the Use*. Prentice Hall, 2nd Edition. 1998.

[48]Gimenez, J. R. B. *Estabilidade Numérica dos Algoritmos de Mínimos Quadrados Rápidos.* Tese de Doutorado. UNICAMP. 1995.

[49] Âström, K. J. and Wittenmark, B. *Adaptive Control*. Addison Wesley. 2nd Ed. 1995.

[50] Ljung, S., Ljung, L. *Error propagation properties of recursive least-squares adaptation algorithms*. Automatica, vol 21, pp 157-167, 1985.

[51] Verhaegen, M. W. *Round-off error propagation in four generally-applicable, recursive, least-squares estimation schemes*. Automatica, vol 25, pp 437-444, 1989.

[52]S. Bradtke, B. Ydstie, e A. Barto. *Adaptive linear quadratic control using policy iteration*.In Computer Science Department University of Massachusetts. 1994.