# Controle Ótimo Não Linear em Malha Fechada por Redes Neurais Profundas

**Róger Mateus Sehnem** [*] **Alexandre Sanfelici Bazanella** [**]

[*] *Programa de Pós Graduação em Engenharia Elétrica (PPGEE), Universidade Federal do Rio Grande do Sul, RS, (e-mail: roger@sehnem.com).*
[**] *Programa de Pós Graduação em Engenharia Elétrica (PPGEE), Universidade Federal do Rio Grande do Sul, RS, (e-mail: bazanella@ufrgs.br).*

**Abstract:** In this work, the design of an optimal closed-loop control of an inverted pendulum is performed using a deep neural network (DNN). A training database is created by solving several optimal control problems, resulting in an optimal open loop control, using the *MATLAB* platform. The DNN is then trained in this database and, after, used to calculate the optimal control given the plant states, effectively closing the loop. Simulations are performed and presented to demonstrate the optimal controller performance when applied to the system.

**Resumo**: Neste trabalho o projeto de um controle ótimo em malha fechada de um pêndulo invertido é realizado através da utilização de uma rede neural profunda (DNN). Uma base de dados é criada através da solução de diversos problemas de controle ótimo, o que resulta em um controle ótimo em malha aberta, utilizando a plataforma *MATLAB*. A DNN é então treinada nesta base e, após, utilizada para calcular o controle ótimo dado os estados da planta, efetivamente fechando a malha. Simulações são realizadas e apresentadas para demonstrar o desempenho da controladora ótima quando aplicada ao sistema.

*Keywords:* Optimal Closed Loop Control; DNN; Nonlinear Dynamics; Inverted Pendulum; Nonlinear Control.

*Palavras-chaves:* Controle Ótimo em Malha Fechada; DNN; Dinâmica não linear; Pêndulo Invertido; Controle Não Linear.

## 1. INTRODUÇÃO

Grande parte dos problemas de controle de sistemas pode ser descrito através da minimização de um índice de desempenho, função dos estados, controles e tempo. A regulação e o seguimento de referência dos estados da planta são dois exemplos clássicos onde os objetivos de controle são expressos através da minimização de um índice de desempenho. Para sistemas lineares e penalizações quadráticas dos estados e controles, as soluções destes problemas dão origem aos clássicos controles LQR e LQT (Bryson and Ho, 1975).

Já para situações mais genéricas, como sistemas não lineares e índices de desempenho quaisquer, a teoria de controle ótimo costuma fornecer como solução o sinal de controle ótimo $u^*(t)$ (Kirk, 2004). Entretanto nos sistemas de controle implementados em plantas reais é ideal obter um controle que seja função dos estados da planta ao invés de uma função do tempo, uma vez que o controle em malha fechada é naturalmente mais capaz de lidar com incertezas. Uma proposta para a solução deste problema é apresentada em Zhu et al. (2019) e Sanchez-Sanchez et al. (2017), onde uma rede neural profunda (DNN) é treinada com trajetórias ótimas e então utilizada para calcular o controle ótimo a partir dos estados em malha fechada. Esta mesma estratégia é aplicada e desenvolvida neste trabalho, onde faz-se uso de outra arquitetura e uma diferente função de perda para o treinamento da DNN, mais adequado ao problema de estabilização. A DNN deve ser capaz de mapear cada estado ótimo $\boldsymbol{x}^*(t)$ para um controle ótimo $\boldsymbol{u}^*(t)$. Caso a mesma seja capaz de realizar este mapeamento em toda a região do espaço de estados em que o sistema atua, tem-se então um controle ótimo em malha fechada.

Em suma, no presente trabalho, busca-se encontrar a lei de controle ótimo em malha fechada $\boldsymbol{u} = \boldsymbol{\gamma}(\boldsymbol{x})$ através da solução de um PCO. A solução do PCO, por sua vez, requer a definição de um índice de desempenho representativo e a obtenção de um modelo da planta que se deseja controlar. Para poder avaliar a viabilidade desta abordagem utiliza-se o modelo de um pêndulo invertido, uma planta bastante conhecida, não linear, de fácil modelagem e com estados de fácil interpretação.

No presente trabalho, diversas trajetórias ótimas para um pêndulo invertido são calculadas através da plataforma *MATLAB*. Cada passo de tempo destas trajetórias, ou

[⋆] O presente trabalho foi realizado com apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq). O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001 .

seja, cada conjunto $\{\boldsymbol{x}^*(t_i), \boldsymbol{u}^*(t_i)\}$, forma a base de dados que é utilizada para treinar a DNN que mapeia de $\boldsymbol{x}^*(t_i)$ para $\boldsymbol{u}^*(t_i)$.

A DNN é criada e treinada utilizando o TensorFlow, devido a sua capacidade de utilização de GPU, o que torna o custoso processo de treinamento mais rápido e eficiente. Os dados utilizados são tratados em Python, e várias análises são também realizadas utilizando esta linguagem de programação.

## 2. CONTROLE ÓTIMO

A obtenção das condições necessárias para uma trajetória ótima foram extraídas de Kirk (2004), Lewis et al. (2012) e Bryson and Ho (1975), que as apresentam com uma maior profundidade. Seja a planta de interesse dada por

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), t), \tag{1}$$

o problema de controlar a planta (1) é o de encontrar o controle $\boldsymbol{u}(t)$ que faz com que os estados $\boldsymbol{x}(t)$ apresentem um comportamento desejado.

Em controle ótimo, o comportamento desejado é aquele que minimiza um índice de desempenho que depende dos estados, controles e do tempo. O problema de controle ótimo é, portanto, encontrar a função $\boldsymbol{u}^*(t)$ que minimiza um índice de desempenho $J(\boldsymbol{x}(t), \boldsymbol{u}(t), t)$, sujeito à restrição (1).

Seja o índice de desempenho dado por

$$J(\boldsymbol{x}(t), \boldsymbol{u}(t), t) = \phi(\boldsymbol{x}(T), T) + \int_{t_0}^{T} L(\boldsymbol{x}(t), \boldsymbol{u}(t), t)dt, \tag{2}$$

onde $T$ corresponde ao tempo final e $\phi(\boldsymbol{x}(T), T)$ é uma função escalar que depende do estado e tempo finais. Além disso, $\boldsymbol{u}^*(t)$ deve ser escolhido de modo que uma restrição do estado final $\boldsymbol{\psi}$ seja satisfeita. Esta última tem a seguinte forma:

$$\boldsymbol{\psi}(\boldsymbol{x}(T), T) = \boldsymbol{0}, \tag{3}$$

onde $\boldsymbol{\psi}(\boldsymbol{x}(T), T)$ é um campo vetorial de dimensão $p$.

Assim, o objetivo é minimizar o índice de desempenho, de modo que os estados respeitem as restrições dinâmicas (1) e de estado final (3). Ver a dinâmica como uma restrição permite criar um índice de desempenho aumentado $J_a$, no qual as restrições são adicionadas ao índice de desempenho com multiplicadores de Lagrange. Deste modo, de acordo com a teoria de Lagrange, o mínimo de $J_a$ é o mínimo de $J$ que respeita as restrições (1) e (3).

É útil, para resolver este problema, definir a função **Hamiltoniana** como

$$H(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\lambda}(t), t) \coloneqq L(\boldsymbol{x}(t), \boldsymbol{u}(t), t) + \boldsymbol{\lambda}^T(t)\boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), t). \tag{4}$$

Com isso, o índice de desempenho aumentado é, portanto,

$$J_a(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\lambda}(t), \boldsymbol{\nu}, t) = \phi(\boldsymbol{x}(T), T) + \boldsymbol{\nu}^T\boldsymbol{\psi}(\boldsymbol{x}(T), T) + \int_{t_0}^{T} \left[H(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\lambda}(t), t) - \boldsymbol{\lambda}^T(t)\dot{\boldsymbol{x}}\right] dt, \tag{5}$$

onde $\boldsymbol{\lambda}(t) = [\lambda_1(t),\ \lambda_2(t),\ \cdots,\ \lambda_n(t)]^T$ e $\boldsymbol{\nu} = [\nu_1,\ \cdots,\ \nu_p]^T$ são os multiplicadores de Lagrange.

Para um extremo de $J_a$, é necessário que variações independentes quaisquer em seus argumentos resultem em uma variação nula do funcional. Assim, de acordo com Lewis et al. (2012), as condições para um extremo são:

$$\begin{aligned} \left(\phi_{\boldsymbol{x}} + \boldsymbol{\psi}_{\boldsymbol{x}}^T\boldsymbol{\nu} - \boldsymbol{\lambda}\right)|_T &= 0; \\ \left(\phi_t + \boldsymbol{\psi}_t^T\boldsymbol{\nu} + H\right)|_T &= 0; \\ \dot{\boldsymbol{\lambda}} = -H_{\boldsymbol{x}}, &\quad t \in [t_0, T]; \\ H_{\boldsymbol{u}} = 0, &\quad t \in [t_0, T]; \end{aligned} \tag{6}$$

onde o subíndice indica a diferenciação com relação àquela variável.

As condições expressas em (6) são apenas condições necessárias para um mínimo (ou máximo). Condições necessárias adicionais para um mínimo podem ser encontradas aplicando o **princípio do mínimo de Pontryagin**. Em suma, este princípio requer que, ao longo de toda trajetória, o controle minimize a Hamiltoniana (Bryson and Ho, 1975). Ou seja, além de ser necessário que $H_{\boldsymbol{u}} = 0$, é também necessário que

$$H_{\boldsymbol{uu}} \geq 0, \quad t \in [t_0, T]. \tag{7}$$

Se a dinâmica $\boldsymbol{f}$ e a função $L$ não forem funções do tempo, isto é, se o problema for o de um sistema invariante no tempo, a Hamiltoniana não é função explícita do tempo. Desse modo, de acordo com Bryson and Ho (1975)

$$H_t = 0, \quad t \in [t_0, T]. \tag{8}$$

A dinâmica da planta, junto das condições necessárias (6) e (7), formam um Problema de Valor de Contorno em Dois Pontos (PVCDP). Para resolvê-lo pode-se criar um sistema aumentado com a dinâmica da planta e dos coestados (terceira Equação em (6)). Assim,

$$\dot{\boldsymbol{X}} = \begin{bmatrix} \dot{\boldsymbol{x}} \\ \dot{\boldsymbol{\lambda}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \\ -H_{\boldsymbol{x}}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\lambda}(t), t) \end{bmatrix}, \tag{9}$$

tem as condições de contorno dadas em (6) e pelas condições iniciais do sistema. Com isso, a solução deste PVCDP gera os estados e controles ótimos necessários para o treinamento.

## 3. CONTROLE ÓTIMO EM MALHA FECHADA POR REDE NEURAL

Caso $\boldsymbol{u}(t)$ fosse obtido a partir de uma função que mapeasse os estados atuais para o controle ótimo, ter-se-ia um controle ótimo em malha fechada, o que, como já comentado, é de grande interesse para a maioria dos sistemas. Como a solução do PCO apresenta, para cada passo de tempo, os estados e correspondente controle ótimo, é natural que se tente obter a função $\boldsymbol{u} = \boldsymbol{\gamma}(\boldsymbol{x})$, a qual leva dos estados para o controle, ou seja, um controle em malha fechada. Neste trabalho, isto é feito através de uma rede neural profunda, uma vez que a mesma é capaz de aproximar qualquer função.

Deste modo, o objetivo é treinar uma rede neural profunda (DNN) que seja capaz de, dado os estados atuais, predizer o controle ótimo atual. Aplicando esta rede neural treinada para realizar o controle, pode-se fechar a malha de controle e obter um controle ótimo em malha fechada por redes neurais profundas ($OPTDNN$). Para o treinamento gerou-se uma base de dados composta de 2744 trajetórias ótimas. Cada trajetória ótima parte de uma condição inicial diferente, e é composta dos controles ótimos $\boldsymbol{u}^*(t_i)$, dos estados ótimos $\boldsymbol{x}^*(t_i)$, gerados pela aplicação deste controle ótimo

na planta, e dos instantes tempo $t_i$. A base de dados completa totalizou $1,8\ GB$.

Foram testadas várias arquiteturas de DNN para este problema. A abordagem utilizada foi a de implementar e treinar DNNs cada vez mais complexas, até que a função de custo de treinamento (10) obtida fosse suficientemente pequena. Esta abordagem é necessária para evitar que se obtenha uma DNN desnecessariamente complexa, a arquitetura da DNN obtida através deste processo é a seguinte:

(1) Camada de entrada com 4 neurônios;
(2) Camada oculta com quatro neurônios de função de ativação linear;
(3) Seis camadas ocultas com cinquenta neurônios de função de ativação sigmoide;
(4) Camada oculta com 40 neurônios de função de ativação sigmoide;
(5) Camada oculta com 30 neurônios de função de ativação sigmoide;
(6) Camada oculta com 20 neurônios de função de ativação sigmoide;
(7) Camada oculta com 10 neurônios de função de ativação sigmoide;
(8) Camada de saída com 1 neurônio de função de ativação linear.

A camada de entrada possui um neurônio para cada estado. A ideia da primeira camada oculta é proporcionar às camadas seguintes a melhor combinação linear dos estados de entrada. Uma vez que todas as entradas tem unidades diferentes, essa camada também fica responsável por, de certo modo, escalar as entradas.

As próximas camadas da rede utilizam todas uma função de ativação sigmoide. Esta função foi escolhida pois foi a que gerou melhores resultados para a OPTDNN. A estrutura em pirâmide (50-40-30-20-10 neurônios) foi utilizada para que, segundo Géron (2019), em cada camada subsequente, a DNN seja "forçada" a reconhecer padrões cada vez mais complexos e que, na última camada, resulta no controle propriamente.

Esta rede foi treinada utilizando, de forma randômica, os conjuntos de estados e controles ótimos, ou seja, dos conjuntos $\{\boldsymbol{x}^*(t_i), \boldsymbol{u}^*(t_i)\}$, de todas as trajetórias do banco de dados. Em outras palavras, pega-se um passo de tempo aleatório de uma trajetória aleatória para cada passo do treinamento. Essa randomização é importante para garantir que a rede treinada não seja enviesada para alguma trajetória específica. Além disso, a rede é treinada com várias épocas e com a condição de parada de não melhora da função de custo de treinamento, da Equação (10), por 5 épocas consecutivas.

A função de custo utilizada para treinar a DNN é dada pela seguinte equação:

$$c(\boldsymbol{u}*, \boldsymbol{u}_p) = \alpha m_{sle}(\boldsymbol{u}^*, \boldsymbol{u}_p) + (1-\alpha) m_{se}(\boldsymbol{u}^*, \boldsymbol{u}_p), \quad (10)$$

onde $\boldsymbol{u}_p$ é o controle predito pela DNN.

Esta função de custo é formada por uma combinação convexa do erro médio quadrático $m_{se}$, com o erro médio quadrático logarítmico $m_{sle}$, definidas como:

$$\begin{aligned} m_{sle}(\boldsymbol{u}^*, \boldsymbol{u}_p) &= \frac{1}{n_b} \sum_{i=0}^{n_b-1} (log_e(1+\boldsymbol{u}^*) - log_e(1+\boldsymbol{u}_p))^2. \\ m_{se}(\boldsymbol{u}^*, \boldsymbol{u}_p) &= \frac{1}{n_b} \sum_{i=0}^{n_b-1} (\boldsymbol{u}^* - \boldsymbol{u}_p)^2. \end{aligned} \quad (11)$$

É interessante notar que a função $m_{sle}$, em (11), penaliza algo próximo ao erro percentual. Diferente de $m_{se}$, que penaliza mais os valores maiores. Isso é facilmente visto se, por exemplo, $\boldsymbol{u}^*(t) \equiv 1000$ e $\boldsymbol{u}_p(t) \equiv 999$. Nesse caso, $m_{sle}(\boldsymbol{u}^*, \boldsymbol{u}_p) = 9,9900 \times 10^{-7}$ e $m_{se}(\boldsymbol{u}^*, \boldsymbol{u}_p) = 1$, deixando claro que a função penaliza muito mais valores $(\boldsymbol{u}^*, \boldsymbol{u}_p)$ grandes do que diferenças percentuais. Utilizar a função custo (10) propicia um melhor comportamento na região de equilíbrio. Uma vez que os controles nesta região tendem a ser muito próximos de zero, uma função que não penaliza corretamente esta região terá uma maior tendência de obter um controle oscilatório em malha fechada.

## 4. MODELAGEM MATEMÁTICA DO PÊNDULO INVERTIDO

O pêndulo invertido é um sistema mecânico composto de um carro que suporta, através de um pino, uma haste. Este sistema é apresentado na Fig. 1.

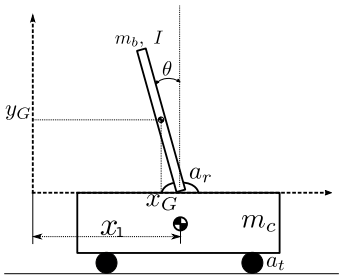


Figura 1. Representação esquemática de um pêndulo invertido.

A haste é livre para girar em torno do pino, possuindo um ângulo com a vertical de $\theta$, coeficiente de atrito viscoso de rotação $a_r$, massa $m_b$ e momento de inércia $I$. O carro, por sua vez, possui posição horizontal $x$, massa $m_c$ e coeficiente de atrito viscoso de translação $a_t$. Assume-se que o mesmo não altera a sua altura, ou seja, desconsidera-se seu movimento vertical.

O sistema apresentado na Fig. 1 possui duas condições de equilíbrio: $\theta = 180°$ e $\theta = 0°$. O objetivo é manter a haste com ângulos $\theta$ próximos de $0°$ através da aplicação de uma força de controle $F(t)$, aplicada no carro no sentido horizontal.

Definindo $\boldsymbol{x} = [x_1\ x_2\ x_3\ x_4]^T = [x\ \dot{x}\ \theta\ \dot{\theta}]^T$, de acordo com Sanchez-Sanchez et al. (2017), o sistema de Equações diferenciais que descreve a planta é:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} x_2 \\ f_2(\boldsymbol{x}, t) \\ x_4 \\ f_4(\boldsymbol{x}, t) \end{bmatrix}, \quad (12)$$

Tabela 1. Valores das constantes do sistema.

| Parâmetro | Valor | Unidade |
|---|---|---|
| $l$ | $0,3$ | $m$ |
| $I$ | $2$ | $m^4$ |
| $m_b$ | $1$ | $kg$ |
| $m_c$ | $3$ | $kg$ |
| $a_t$ | $0,2$ | $kg\ s^{-1}$ |
| $a_r$ | $0,2$ | $kg\ m^2\ s^{-1}$ |

onde

$$f_2(\boldsymbol{x},t) = \frac{1}{C_2C_3 - C_1^2c(\theta)^2}\Big[gC_1^2c(\theta)s(\theta) + C_2(F(t) - a_t\dot{x}) - a_rC_1c(\theta)\dot{\theta} - C_1C_2s(\theta)\dot{\theta}^2\Big] \quad (13)$$

e

$$f_4(\boldsymbol{x},t) = \frac{1}{C_2C_3 - C_1^2c(\theta)^2}\Big[gC_1C_3s(\theta) + C_1c(\theta)(F(t) - a_t\dot{x}) - a_rC_3\dot{\theta} - C_1^2c(\theta)s(\theta)\dot{\theta}^2\Big]. \quad (14)$$

Nesses, $C_1 = lm_b$, $C_2 = I + l^2m_b$, $C_3 = m_b + m_c$, $s(\cdot) = \sin(\cdot)$ e $c(\cdot) = \cos(\cdot)$ e $l$ é o comprimento da haste.

Pode-se ainda definir a função de saída como

$$\boldsymbol{g}(\boldsymbol{x},\boldsymbol{u}) = \boldsymbol{x}, \quad (15)$$

onde $\boldsymbol{u} = F(t)$ é o controle do sistema, uma força horizontal aplicada no CG do carro.

Os parâmetros físicos do pêndulo são apresentados na Tabela 1.

## 5. PROBLEMA DE VALOR DE CONTORNO RESULTANTE DA SOLUÇÃO DO PROBLEMA DE CONTROLE ÓTIMO DO PÊNDULO INVERTIDO

A primeira parte do processo de obtenção do PVCDP resultante do problema de controle ótimo (PCO) é a definição do índice de desempenho (2). Um índice de desempenho que costuma ser de interesse em aplicações de controle em engenharia é o índice de desempenho quadrático:

$$J(\boldsymbol{x}(t),\boldsymbol{u}(t),t) = \int_{t_0}^{T}\left[\boldsymbol{x}^T(t)Q\boldsymbol{x}(t) + \boldsymbol{u}^T(t)R\boldsymbol{u}(t)\right]dt, \quad (16)$$

onde $Q$ é a matrix que penaliza os estados e $R$ é a matriz que penaliza os controles. As matrizes $Q$ e $R$ utilizadas no problema são:

$$Q = qI_{4x4}\ \ R = r, \quad (17)$$

onde $q$ e $r$ são escalares e $I_{3x3}$ é a matriz identidade de dimensão 3.

O índice de desempenho (16), contudo, não apresenta uma penalização direta ao tempo levado para atingir estados e controles nulos, ou seja, para atingir o equilíbrio. Assim, propõe-se a seguinte modificação:

$$J(\boldsymbol{x}(t),\boldsymbol{u}(t),t) = \int_{t_0}^{T}\left[\boldsymbol{x}^T(t)Q\boldsymbol{x}(t) + \boldsymbol{u}^T(t)R\boldsymbol{u}(t) + q\right]dt. \quad (18)$$

O índice de desempenho (18) agora penaliza o tempo total levado para atingir o estado final desejado, uma vez que a última parte da integral pode ser resolvida como

$$\int_{t_0}^{T} qdt = q(T - t_0), \quad (19)$$

onde fica claro que quanto maior o tempo final $T$, maior será o resultado da integral.

Os índices de desempenho (16) e (18) ambos têm a penalização do estado final $\phi(\boldsymbol{x}(T),T) = 0$.

Utilizando o índice de desempenho (18) e a dinâmica da planta (12) em (4) obtém-se a seguinte Hamiltoniana:

$$H = q + qx_1^2 + qx_2^2 + qx_3^2 + qx_4^2 + x_2\lambda_1 + x_4\lambda_4 + rF^2 + \frac{1}{w}\Big[\lambda_2((I + l^2m_b)F + lm_bc(x_3)(-a_rx_4 + glm_bs(x_3)) - (I + l^2m_b)(a_tx_2 + lm_bx_4^2s(x_3)))\Big] + \frac{1}{w}\Big[\lambda_4(-lm_bc(x_3)F + (m_b + m_c)(a_rx_4 - glm_bs(x_3)) + lm_bc(x_3)(a_tx_2 + lm_bx_4^2s(x_3)))\Big], \quad (20)$$

onde $w = (I + l^2m_b)(m_b + m_c) - l^2m_b^2\cos(x_3)^2$. Nos experimentos foi utilizado $q = r = 1$.

A dinâmica dos coestados, dada pela terceira equação em (6), é longa e, por este motivo, não será apresentada.

Aplicando (20) na quarta Equação em (6) e resolvendo para $F$, obtém-se

$$F = -\frac{(I + l^2m_b)\lambda_2 + lm_b\lambda_4\cos(x_3)}{(I + l^2m_b)(m_b + m_c) - 2l^2m_b^2r\cos(x_3)^2}. \quad (21)$$

Aplicando (20) em (7)

$$H_{uu} = 2r > 0, \quad (22)$$

onde fica claro que, contanto que se escolha $r > 0$, (21) é um controle que minimiza o índice de desempenho da Equação (18).

Neste problema deseja-se levar estados quaisquer no tempo inicial $\boldsymbol{x}_0$ para zero no tempo final $\boldsymbol{x}_T$, que é a condição de equilíbrio do pêndulo quando invertido. Assim a restrição de estado final é da forma

$$\boldsymbol{\psi} = [x_1\ \ x_2\ \ x_3\ \ x_4]^T. \quad (23)$$

Aplicando-se (23) na primeira Equação de (6), obtém-se

$$\boldsymbol{\lambda} = [\nu_1\ \ \nu_2\ \ \nu_3\ \ \nu_4]^T. \quad (24)$$

Assim, devem-se escolher coestados $\boldsymbol{\lambda}$ finais de modo a satisfazer a Equação (23). No problema de valor de contorno em dois pontos, essa escolha é equivalente à dizer que os estados finais devem ser nulos e que os coestados finais são livres.

Por fim, calculando a segunda Equação em (6), observa-se que

$$H(T) = 0. \quad (25)$$

Observando que a Hamiltoniana (20) não é função explícita do tempo e utilizando (8), obtém-se que, para uma trajetória ótima, a Hamiltoniana é nula ao longo de todo o tempo.

## 6. IMPLEMENTAÇÃO NUMÉRICA

Para a solução numérica do PVCDP resultante do PCO, é necessário montar um sistema aumentado com a dinâmica dos estados e coestados. As condições de contorno são, neste caso, dadas pelas condições dos estados no tempo

inicial $\boldsymbol{x}(0) = \boldsymbol{x}_0$ e pelas condições dos estados no tempo final $\boldsymbol{x}(T) = \mathbf{0}$.

Para resolver este PVCDP utilizou-se a Linguagem *MATLAB* e sua função *bvp4c*, que utiliza o método da colocação polinomial para a resolução de problemas de valor de contorno (Kierzenka and Shampine, 2001).

Para avaliar a qualidade do controlador proposto, foi implementado um simples simulador em Python. Esse implementa a dinâmica da Equação (12), onde o controle é gerado através dos estados pela OPTDNN treinada. O resultado da simulação é a trajetória no tempo dos estados e controle.

## 7. RESULTADOS E DISCUSSÕES

Existem dois grupos de resultados a serem avaliados, os resultados das soluções do PVCDP provenientes do PCO e os resultados referentes à qualidade do controle gerado pela OPTDNN. Por questão organizacional, ambos são apresentados em diferentes subseções.

### *7.1 Dados para o treinamento*

Um exemplo de trajetória ótima da base de dados é apresentado na Figura 2.

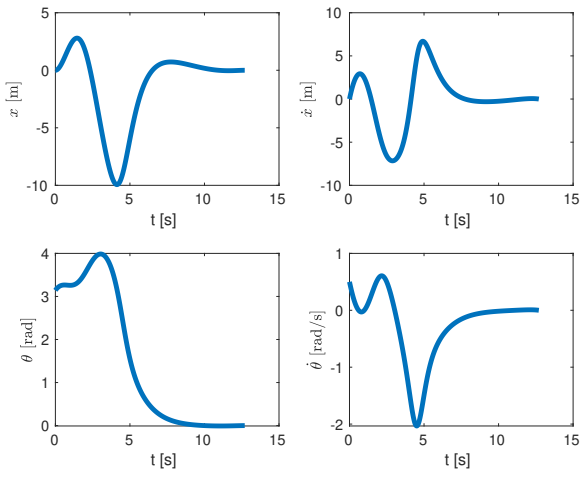


Figura 2. Trajetória ótima dos estados.

Observa-se que o pêndulo parte de posição e velocidade nulas, com ângulo de $\pi$ $[rad]$ e velocidade angular de 0.5 $[rad/s]$. É interessante observar que esta condição corresponde à condição em que o pêndulo encontra-se completamente para baixo e com uma pequena velocidade angular. Os coestados, Hamiltoniana e controle referentes a esta trajetória são apresentados na Figura 3.

É interessante avaliar quão bem explorado o espaço de condições iniciais foi. Uma boa base de dados teria trajetórias ótimas com condições iniciais bem distribuídas ao longo da região em que se espera que o controlador seja efetivo.

A Figura 4 apresenta a avaliação da base de dados, nela cada ponto representa a condição inicial de cada par de estados de uma simulação, como indicado nos eixos.

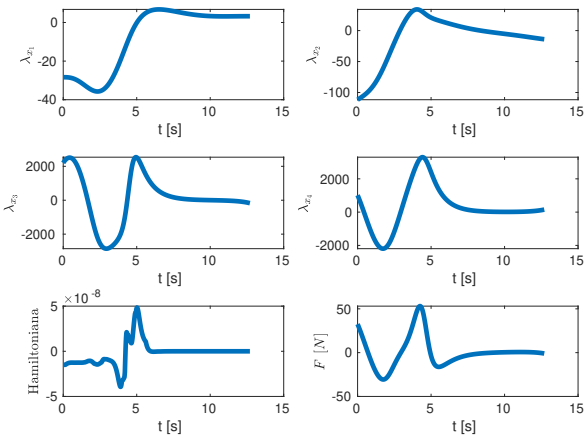


Figura 3. Trajetória ótima dos coestados, Hamiltoniana e controle.

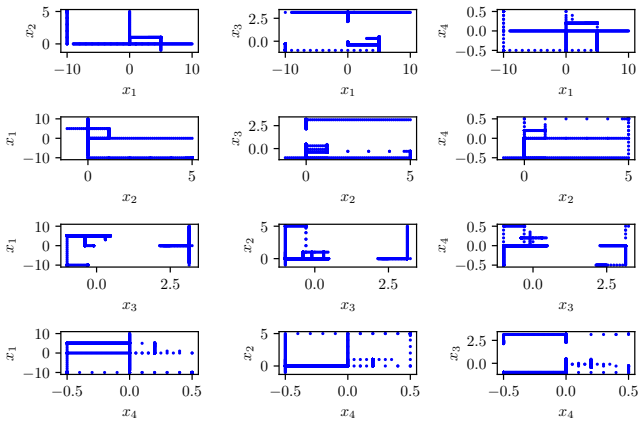

Figura 4. Avaliação das condições iniciais da base de dados.

Observa-se que, na Figura 4, os pontos estão bastante concentrados em linhas ou regiões, não apresentando o comportamento que seria considerado ideal, ou seja, completamente distribuído.

Essa característica da base de dados se dá pelo fato de que o algoritmo utilizado para gerá-la é semi automático, ou seja, precisa-se setar um valor de CI para variar por vez, o que favorece o aparecimento de linhas. Essa característica do algoritmo, por sua vez, é justificada pela facilidade de se implementar o método da continuação e de se obter uma base de dados com diferentes CIs.

### *7.2 Operação da rede em malha fechada*

A Figura 5 apresenta os resultados da OPTDNN para a CI de $\boldsymbol{x}_0 = [0\ 0\ -0,1415\ 0]^T$, que não está contida no conjunto de treinamento. Observa-se que neste caso a OPTDNN é capaz de fazer com que o sistema convirja para o ponto de equilíbrio em um curtíssimo período de tempo e de forma bastante similar ao controle ótimo em malha aberta.

Para a CI de $\boldsymbol{x}_0 = [0\ 0\ \pi\ 0]^T$ o comportamento é apresentado na Figura 6. Neste caso a OPTDNN também se mostra extremamente efetiva em estabilizar o ponto de equilíbrio, com comportamento bastante similar ao ótimo.

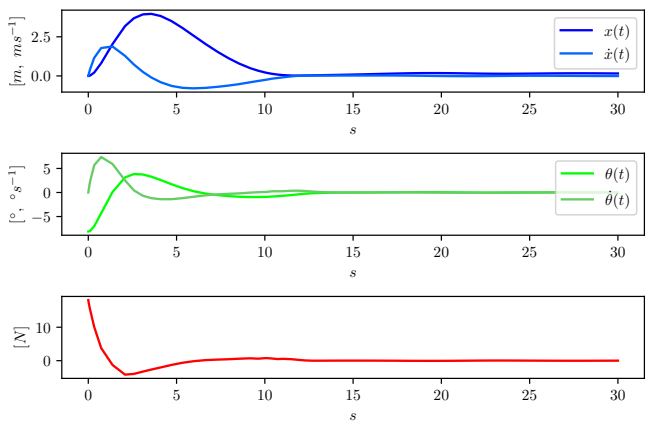


Figura 5. OPTDNN em malha fechada com $\boldsymbol{x}_0 = [0\ 0\ -0,1415\ 0]^T$.

Nota-se que este é o caso do movimento de swing up, isto é, o movimento de trazer o pêndulo da posição vertical inferior para a posição vertical superior. A OPTDNN é capaz de controlar o sistema mesmo nestas condições extremas e ainda assim de forma extremamente próxima à ótima.

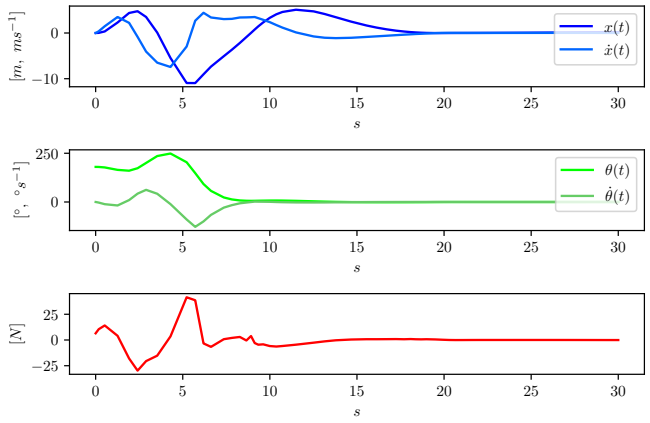


Figura 6. OPTDNN em malha fechada com $\boldsymbol{x}_0 = [0\ 0\ \pi\ 0]^T$.

Contudo, não é sempre que a OPTDNN é capaz de estabilizar o ponto de equilíbrio. Como no caso de, por exemplo, uma CI de $\boldsymbol{x}_0 = [0\ 0\ 0,1415\ 0]^T$. Nesta CI, a OPTDNN não é capaz de fazer com que o sistema convirja para o ponto de equilíbrio e essa apresenta um comportamento muito longe do que é o ideal. Tal comportamento é, provavelmente, causado pelo fato de que a base de dados é bastante pobre nesta região de CIs, como pode-se observar na Figura 4. Este resultado também mostra que a região de atração do ponto de equilíbrio em questão é não simétrica e, por vezes, tem uma fronteira que se aproxima bastante do mesmo.

A trajetória da Figura 2 tem o índice de desempenho (18) de $J = 4,076 \times 10^3$, enquanto que o índice de desempenho para a mesma CI, utilizando o controle em malha fechada com a OPTDNN, é $J = 4,205 \times 10^3$. Este resultado indica que a OPTDNN tem um controle em malha fechada bastante próximo do controle ótimo, uma vez que o índice de desempenho real ótimo é apenas, aproximadamente, 3% menor que o controle em malha fechada.

Assim, a OPTDNN se mostra bastante efetiva em efetuar o controle ótimo em malha fechada. Ainda mais considerando a baixa qualidade da base de dados, como apresentado na Figura 4. Além disso, o controle em malha fechada aparenta ser próximo ao controle ótimo, como indicado pela avaliação dos índices de desempenho.

## 8. CONCLUSÃO

Neste trabalho uma rede neural foi treinada para calcular o controle ótimo do pêndulo invertido com base nos seus estados atuais. Esta rede neural foi então utilizada para controlar o sistema em malha fechada, sistema de controle que foi denominado OPTDNN.

A arquitetura da DNN utilizada é bastante simples e pouco profunda. O treinamento foi feito utilizando uma base de dados pouco dispersa e os resultados são bastante promissores. O controlador obtido é efetivamente um controlador não linear para o sistema do pêndulo invertido, um problema de reconhecida difícil solução. Não obstante, o controlador obtido produz, como demonstrado, resultados bastante similares aos do controle ótimo.

## AGRADECIMENTOS

O presente trabalho foi realizado com apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

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

## REFERÊNCIAS

Bryson, A.E. and Ho, Y.C. (1975). *Applied optimal control: optimization, estimation and control*. Taylor and Francis.

Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. "O'Reilly Media, Inc.".

Kierzenka, J. and Shampine, L.F. (2001). A bvp solver based on residual control and the maltab pse. *ACM Transactions on Mathematical Software (TOMS)*, 27(3), 299–316.

Kirk, D.E. (2004). *Optimal control theory: an introduction*. Dover Publications.

Lewis, F.L., Vrabie, D., and Syrmos, V.L. (2012). *Optimal control*. John Wiley & Sons.

Sanchez-Sanchez, C., Izzo, D., and Hennes, D. (2017). Learning the optimal state-feedback using deep networks. *2016 IEEE Symposium Series on Computational Intelligence, SSCI 2016*. doi:10.1109/SSCI.2016.7850105.

Zhu, L., Ma, J., and Wang, S. (2019). Deep neural networks based real-time optimal control for lunar landing. In *IOP Conference Series: Materials Science and Engineering*, volume 608, 012045. IOP Publishing.