Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Mecânica

Sincronização Neural e Sub-atuada para um Sistema Hipercaótico Tetradimensional

Felipe Ofugi Hara

DISSERTAÇÃO DE MESTRADO

PROGRAMA DE PÓS-GRADUAÇÃO EM SISTEMAS MECATRÔNICOS

Brasília

2025

Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Mecânica

Sincronização Neural e Sub-atuada para um Sistema Hipercaótico Tetradimensional

Felipe Ofugi Hara

Dissertação de Mestrado submetida ao Departamento de Engenharia Mecânica da Universidade Brasília como parte dos requisitos necessários para a obtenção do grau de Mestre

Orientador: Prof. Dr. José Alfredo Ruiz Vargas

Brasília
2025

Hara, Felipe Ofugi.
H769s Sincronização Neural e Sub-atuada para um Sistema Hipercaótico Tetradimensional / Felipe Ofugi Hara; orientador José Alfredo Ruiz Vargas. -- Brasília, 2025.
97 p.

Dissertação de Mestrado (Programa de Pós-Graduação em Sistemas Mecatrônicos) -- Universidade de Brasília, 2025.

1. Análise de Lyapunov. 2. Controle não-linear. 3. Controle Neural. 4. Sincronização Caótica. I. Vargas, José Alfredo Ruiz, orient. II. Título

Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Mecânica

Sincronização Neural e Sub-atuada para um Sistema Hipercaótico Tetradimensional

Felipe Ofugi Hara

Dissertação de Mestrado submetida ao Departamento de Engenharia Mecânica da Universidade Brasília como parte dos requisitos necessários para a obtenção do grau de Mestre

Trabalho aprovado. Brasília, 7 de março de 2025:

Prof. Dr. José Alfredo Ruiz Vargas,
UnB/FT/ENM
Orientador

Prof. Dr. Guillermo Alvarez Bestard,
UnB/FGA
Examinador interno

Dr. Max Eduardo Vizcarra Melgar, TJCE
Examinador externo

Brasília
2025

Este trabalho é dedicado a
Fernanda

Agradecimentos

Agradeço profundamente aos meus pais, Neide e Jorge, e à minha irmã Lívia, por todo o apoio e incentivo ao longo de minha jornada acadêmica e profissional.

Expresso também minha gratidão ao meu orientador, professor Dr. José Alfredo Ruiz Vargas, pela orientação criteriosa, pela confiança depositada em meu trabalho e pela oportunidade de me envolver em diversos projetos nesta área de pesquisa. Seu conhecimento e dedicação foram determinantes para o meu crescimento intelectual e profissional.

Por fim, agradeço ao Dr. Kevin Herman Muraro Gularte pela prontidão em esclarecer dúvidas e oferecer auxílio sempre que necessário. Seu suporte contribuiu para que eu me sentisse mais empenhado em seguir adiante.

“Growth comes from chaos, not order.”
(Rakesh Jhunjhunwala)

Resumo

A sincronização de sistemas caóticos e hipercaóticos vem se destacando em diversas áreas da ciência e tecnologia, como comunicação segura, criptografia e modelagem de fenômenos naturais. Entretanto, grande parte das técnicas de sincronização descritas na literatura apresentam alto grau de complexidade, o que dificulta sua aplicação prática. Neste trabalho, propõe-se uma nova lei de controle neural e sub-atuada para a sincronização de um sistema hipercaótico de quatro dimensões, caracterizada por sua simplicidade de implementação e bom desempenho na presença de distúrbios limitados. A estratégia fundamenta-se na teoria de estabilidade de Lyapunov e foi validada por meio de simulações no MATLAB® e Simulink®. Além disso, demonstrou-se a aplicação da técnica de sincronização proposta na criptografia de sinais para comunicação segura. Os resultados revelaram que a lei de controle, atuando em apenas dois estados do sistema, é simples e robusta diante de distúrbios limitados, mostrando-se eficaz após o escalonamento em amplitude de alguns estados do sistema hipercaótico. Conclui-se que essa abordagem oferece uma solução prática e eficiente para a sincronização de sistemas hipercaóticos, com potencial para aplicações em diversas áreas que demandem controle de sistemas dinâmicos complexos. Esse estudo contribui para a ampliação do conhecimento na área e abre caminho para futuras investigações sobre controle simplificado e robusto em sistemas não lineares.

Palavras-chave: Análise de Lyapunov. Controle não-linear. Controle Neural. Sincronização Caótica.

Abstract

Synchronization of chaotic and hyperchaotic systems has gained prominence in various fields of science and technology, such as secure communication, cryptography, and modeling of natural phenomena. However, many of the synchronization techniques described in the literature exhibit a high degree of complexity, which complicates their practical implementation. In this work, we propose a new neural and underactuated control law for synchronizing a four-dimensional hyperchaotic system. This law stands out for its ease of implementation and good performance in the presence of bounded disturbances. The approach is based on Lyapunov stability theory and was validated through simulations in MATLAB® and Simulink®. In addition, the application of the proposed synchronization technique in signal cryptography for secure communication has been demonstrated. The results revealed that the control law, acting on only two states of the system, is both simple and robust against bounded disturbances, proving effective after amplitude scaling of certain states of the hyperchaotic system. We conclude that this approach provides a practical and efficient solution for synchronizing hyperchaotic systems, with potential applications in various domains requiring control of complex dynamical systems. This study contributes to advancing knowledge in the field and opens up new avenues for future research on simplified and robust control in nonlinear systems.

Keywords: Lyapunov Analysis. Nonlinear control. Neural Control. Chaotic Synchronization.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

4D	Tetradimensional	19
FNN	Rede Neural *Feedforward*	37
RNA	Rede Neural Artificial	37

Sumário

1 Introdução

1.1 Contextualização e motivação

Nas últimas décadas, o estudo de sistemas dinâmicos caóticos tem recebido atenção crescente em diversas áreas da ciência e engenharia, como comunicações seguras (GULARTE, K. H.; HARA et al., 2023a,b; WANG, A. et al., 2025; PRAJAPAT; KUMAR, D.; KUMAR, P., 2025; DHINGRA; DUA, 2025; GA; BHANU, 2025; MANHIL; JAMAL, 2024; BABANLI; KABAOGLU, 2024), processamento de sinais (DUARTE; EISENCRAFT, 2024; YU; CHEN, W.; POOR, 2024), teoria de controle (KARTAL, 2025; BASHIR; MALIK; HUSSAIN, S., 2025), reações químicas (RAZZAQ et al., 2025), meteorologia (DONG et al., 2024), entre outros. A presença de sensibilidade às condições iniciais e comportamento imprevisível são características marcantes dos sistemas caóticos (LORENZ, 1972), tornando-os atrativos para aplicações que demandam alta complexidade dinâmica. No entanto, quando se trata de sistemas hipercaóticos — isto é, sistemas caóticos com quatro ou mais dimensões (dois ou mais expoentes de Lyapunov positivos) (LETELLIER; ROSSLER, O. E., 2007) —, a investigação ainda se encontra em expansão, pois tais sistemas apresentam dinâmicas mais complexas e um comportamento com um maior grau de imprevisibilidade (JIN, M.; SUN; WANG, H., 2022).

Dentro desse contexto, a sincronização de sistemas caóticos ou hipercaóticos surge como um problema de grande interesse teórico e prático. O objetivo é garantir que as instâncias do sistema, mesmo quando sujeitas a perturbações ou incertezas, evoluam juntas ao longo do tempo. As aplicações dessa sincronização abrangem desde comunicações criptografadas (CLEMENTE-LOPEZ; JESUS RANGEL-MAGDALENO; MUÑOZ-PACHECO, 2024; WEN; LIN, 2024) até controle de robôs (MOYSIS et al., 2020; YANG; QIN; LIAO, 2023) e sistemas de energia (TADJ et al., 2024), passando por processamento de sinais biomédicos (PARBAT; CHAKRABORTY, 2021) e identificação de parâmetros em sistemas físicos (PENG; HE; SUN, 2022). Contudo, o projeto de leis de controle e observadores que assegurem a sincronização em sistemas de alta complexidade, como os hipercaóticos de quatro dimensões, apresenta inúmeros desafios, especialmente no que se refere a robustez à disturbios, tempo de convergência e implementação prática.

Para contornar essas dificuldades, diversas abordagens têm sido propostas, como métodos baseados em realimentação linear (LAAREM, 2021), controle adaptativo (SHAFIQ; AHMAD, 2025) e técnicas de observadores não lineares (HUSSAIN, M. M. et al., 2021). Recentemente, o uso de redes neurais tem emergido como uma alternativa promissora, devido à capacidade dessas estruturas de lidar com não linearidades complexas e de se

adaptarem dinamicamente às mudanças no sistema (ANH; DAT, 2024; JIN, J. et al., 2024). Além disso, a sub-ação (ou sub-atuador) traz uma perspectiva interessante ao permitir que o controle não precise atuar plenamente em todas as variáveis do sistema, diminuindo custos de implementação e, em alguns casos, tornando o método mais resiliente a falhas parciais (GULARTE, K. H. M.; GÓMEZ et al., 2023; GULARTE, K. H. M.; ALVES et al., 2021).

Neste trabalho, propõe-se combinar essas duas frentes — controle neural e sub-atuado — para sincronizar um sistema hipercaótico tetradimensional. A motivação principal decorre da necessidade de métodos mais eficientes e flexíveis para lidar com a alta complexidade e instabilidade desses sistemas. Ao empregar redes neurais, busca-se aproveitar sua capacidade de aproximar funções e adaptar parâmetros em tempo real, mantendo ao mesmo tempo a viabilidade de implementação, mesmo em cenários onde não haja atuadores em todas as variáveis de estado. Essa combinação inovadora visa não apenas assegurar a sincronização em cenários adversos, mas também oferecer uma abordagem robusta e generalizável a outros sistemas hipercaóticos que venham a ser estudados no futuro.

Desse modo, a relevância do tema torna-se evidente: as possíveis aplicações práticas e o avanço do conhecimento na área de sistemas hipercaóticos justificam a busca por novas estratégias de controle que sejam teoricamente sólidas e, ao mesmo tempo, tenham potencial de implementação em escala real. Acredita-se que o desenvolvimento e validação desse método de sincronização contribuirão para expandir as fronteiras do controle de sistemas hipercaóticos, fornecendo uma ferramenta útil para engenheiros, físicos e cientistas em geral que lidam com problemas de alta complexidade dinâmica.

1.2 Estado da arte

A Figura 1 ilustra a evolução dos estudos sobre caos e hipercaos, evidenciando os marcos históricos e conceituais que servem de base para o desenvolvimento das atuais técnicas de controle e sincronização. O ponto de partida é atribuído a (LORENZ, 1963), que descreveu o primeiro modelo efetivamente reconhecido como caótico, chamando atenção para o fenômeno da sensibilidade às condições iniciais em sistemas dinâmicos não lineares.

Em (ROSSLER, O., 1979), aprofundou-se no tema ao introduzir o conceito de hipercaos, caracterizado por apresentar pelo menos dois expoentes de Lyapunov positivos, ampliando a complexidade das trajetórias dinâmicas. Alguns anos mais tarde, em (PECORA; CARROLL, 1990) lançou-se as bases da sincronização caótica, demonstrando que sistemas caóticos podiam, sob certas condições de realimentação, evoluir de forma síncrona.

A partir desse alicerce, diversas vertentes de pesquisa começaram a florescer. Em (FRISON, 1992) aplicou-se redes neurais ao controle do caos, evidenciando o potencial de aprendizado e adaptação dessas estruturas para mitigar comportamentos caóticos indesejados ou instáveis. Paralelamente, em (BEDROSSIAN, 1991), propôs-se o controle sub-atuado

de sistemas não lineares, sugerindo a possibilidade de reduzir a complexidade e o custo de implementação ao usar menos atuadores do que o número de estados do sistema.

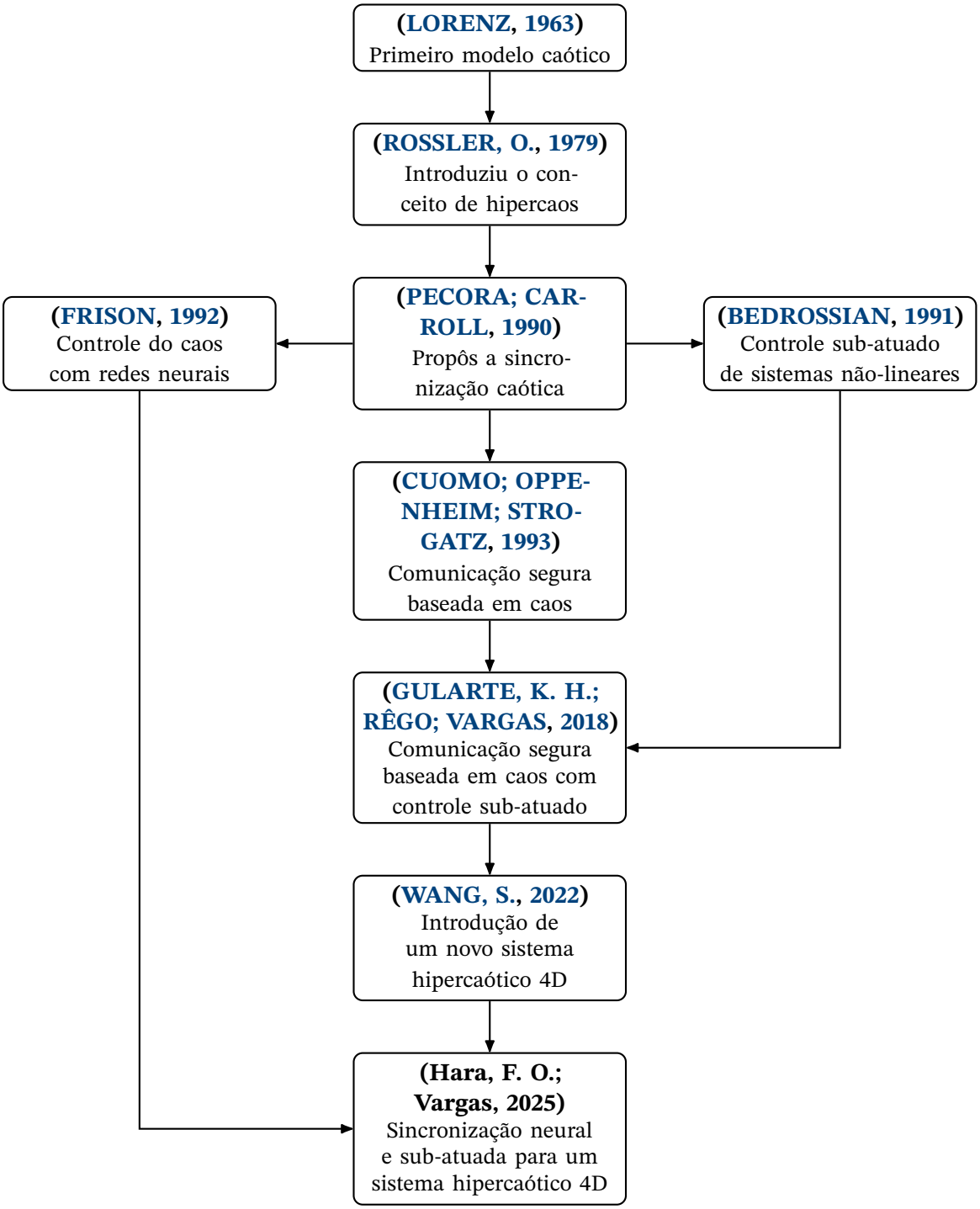

Figura 1 – Contribuições na área que culminaram na proposta deste trabalho.

Já em (CUOMO; OPPENHEIM; STROGATZ, 1993) popularizou-se a aplicação direta do caos em comunicação segura, apontando como a alta sensibilidade do comportamento caótico poderia ser explorada para criptografar informações. Seguindo essa linha, em (GULARTE, K. H.; RÊGO; VARGAS, 2018) demonstrou-se como a integração de técnicas de sub-atuadores poderia trazer maior robustez à comunicação segura baseada em caos.

Mais recentemente, em (WANG, S., 2022) introduziu-se um novo sistema hipercaótico 4D, evidenciando a continuidade das pesquisas em dinâmicas de alta dimensão, abrindo caminho para a exploração de sistemas ainda mais complexos. Finalmente, na culminância desta linha evolutiva, surge a proposta deste trabalho (Hara, F. O. e Vargas, 2025) – que consiste na aplicação conjunta de redes neurais e do controle sub-atuado para efetuar a sincronização de um sistema hipercaótico 4D.

Essa abordagem, objeto central da presente dissertação, visa combinar a adaptabilidade das redes neurais com a eficiência do controle sub-atuado, oferecendo um método promissor para lidar com a instabilidade inerente a sistemas de alta ordem. A proposta reflete não apenas o avanço conceitual ao longo dos últimos sessenta anos de pesquisa sobre caos e hipercaos, mas também a convergência de duas ferramentas potentes — redes neurais e controle sub-atuado — em prol de soluções inovadoras para problemas complexos de sincronização.

A análise dos trabalhos dispostos na figura acima evidencia a evolução desta área ao longo dos últimos anos, bem como a lacuna de pesquisa ainda existente. Especificamente, observa-se a ausência de abordagens combinadas que unam técnicas neurais a estratégias sub-atuadas em sistemas hipercaóticos de alta ordem. Essa lacuna motivou o presente estudo, que visa contribuir para o avanço do estado da arte propondo uma nova estrutura de controle neural e sub-atuada, a ser aplicada em um modelo tetradimensional de sistema hipercaótico. Espera-se, assim, reduzir a complexidade de projeto e incrementar a robustez, fornecendo subsídios para aplicações em larga escala e/ou cenários de incerteza elevada.

1.3 Objetivos

O presente trabalho tem como principal objetivo desenvolver e validar uma nova estratégia de controle capaz de promover a sincronização de um sistema hipercaótico tetradimensional, utilizando técnicas neurais e sub-atuadas. A proposta visa suprir lacunas encontradas no estado da arte, unindo a flexibilidade de aproximação oferecida por redes neurais à economia de recursos e robustez característica de esquemas sub-atuados.

Para alcançar este objetivo geral, os seguintes objetivos específicos foram estabelecidos:

- Modelar e analisar o sistema hipercaótico tetradimensional, destacando suas principais características dinâmicas e pontos críticos de instabilidade.
- Propor e desenvolver um controlador neural sub-atuado, descrevendo sua estrutura, parâmetros de projeto e leis de adaptação.

- Realizar a prova matemática de estabilidade, utilizando a Teoria de Lyapunov e demais conceitos teóricos necessários para garantir a convergência do erro de sincronização a valores próximos de zero.
- Implementar simulações computacionais no MATLAB, avaliando o desempenho da técnica proposta em termos de rapidez de convergência, robustez a perturbações e estabilidade.
- Aplicar a sincronização proposta em um esquema para comunicação segura e avaliar o seu desempenho.
- Discutir as contribuições e limitações do método, sugerindo possíveis extensões e aplicações práticas para pesquisas futuras.

Com base nestes objetivos, espera-se que a metodologia aqui proposta amplie as perspectivas de uso do controle sub-atuado e das redes neurais na sincronização de sistemas hipercaóticos, contribuindo para o avanço do conhecimento na área e incentivando novas aplicações e desenvolvimentos.

1.4 Organização deste trabalho

Esta dissertação está organizada em cinco capítulos, além das referências bibliográficas e apêndice. A seguir, descrevemos brevemente o conteúdo de cada capítulo:

Capítulo 1 – Introdução: Apresenta a contextualização e motivação do tema, o estado da arte relacionado aos estudos sobre sincronização de sistemas hipercaóticos, os objetivos da pesquisa e, por fim, esta visão geral da estrutura da dissertação.

Capítulo 2 – Conceitos Preliminares: Revisa os principais fundamentos teóricos necessários para o desenvolvimento do trabalho. São abordados temas como sistemas caóticos e hipercaóticos, teoria de estabilidade de Lyapunov, desigualdade de Young, escalonamento em amplitude e frequência e noções de redes neurais artificiais. Esses conceitos formam a base que justifica as metodologias e análises adotadas nos capítulos seguintes.

Capítulo 3 – Proposta de Controle Neural e Sub-atuado: Descreve em detalhes o sistema hipercaótico tetradimensional que serve de objeto de estudo e apresenta a técnica de controle proposta. São discutidas a modelagem da estratégia de controle, a formulação do controlador sub-atuado e as redes neurais utilizadas, culminando na prova matemática de estabilidade e sincronização.

Capítulo 4 – Simulações e Validações: Relata a implementação das simulações no MATLAB, destacando as configurações adotadas, os resultados obtidos e a sua aplicação em comunicação segura. As métricas de desempenho, bem como aspectos de robustez e convergência, são tratados neste capítulo.

Capítulo 5 – Conclusões e Trabalhos Futuros: Traz o fechamento do trabalho, sintetizando as principais contribuições e limitações encontradas, bem como possíveis desdobramentos e sugestões para pesquisas futuras.

Após esses capítulos, são apresentadas as referências bibliográficas. Em seguida, são disponibilizados apêndices contendo os códigos de simulação utilizados neste trabalho.

2 Conceitos preliminares

2.1 Sistemas Caóticos e Hipercaóticos

A teoria do caos dedica-se ao estudo de sistemas dinâmicos não lineares que exibem comportamentos aparentemente imprevisíveis, mesmo quando suas equações de evolução são perfeitamente conhecidas. Em tais sistemas, pequenas perturbações nas condições iniciais podem resultar em grandes diferenças na dinâmica ao longo do tempo, fenômeno frequentemente associado ao chamado *efeito borboleta* (LORENZ, 1963).

2.1.1 Definição e Caracterização de Sistemas Caóticos

Considere um sistema dinâmico contínuo no tempo, descrito por:

$$\dot{\mathbf{x}}(t) = \mathbf{f}\big(\mathbf{x}(t), t\big) \tag{2.1}$$

onde $\mathbf{x}(t) \in \mathbb{R}^n$ é o vetor de estados e $\mathbf{f} : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ é uma função não linear que define a dinâmica do sistema. Em termos gerais, podemos chamar esse sistema de *caótico* se ele apresenta:

1. **Sensibilidade às Condições Iniciais**: Dada uma condição inicial $\mathbf{x}_0$ e uma condição perturbada $\mathbf{x}_0 + \delta\mathbf{x}_0$ com $||\delta\mathbf{x}_0||$ muito pequena, as trajetórias $\mathbf{x}(t)$ e $\mathbf{y}(t)$ que partem dessas condições divergem exponencialmente no tempo. Formalmente, existe pelo menos um *expoente de Lyapunov* $\lambda > 0$ tal que

$$||\mathbf{x}(t) - \mathbf{y}(t)|| \approx ||\delta\mathbf{x}_0||\, e^{\lambda t}, \quad \lambda > 0 \tag{2.2}$$

2. **Densidade de Órbitas Periódicas**: Em muitos sistemas caóticos, podem existir órbitas periódicas dispersas pelo espaço de estados; entretanto, a presença simultânea de *sensibilidade às condições iniciais* faz com que essas órbitas não sejam dominantes na dinâmica.

3. **Mistura Topológica**: Qualquer região do espaço de estados eventualmente se "espalha" (ou se mistura) em todo o conjunto invariante do sistema. Em outras palavras, as trajetórias podem visitar complexamente diferentes partes do espaço de fases.

Em termos práticos, a sensibilidade às condições iniciais torna o longo prazo das trajetórias extremamente difícil de prever, mesmo que o sistema seja determinístico (ou seja, não haja incerteza ou aleatoriedade intrínseca nas equações).

2.1.1.1 Exemplo Clássico: Sistema de Lorenz

Um dos exemplos mais conhecidos de sistema caótico é o *Sistema de Lorenz* (LORENZ, 1963), dado pelas equações:

$$\begin{cases} \dot{x} = \sigma(y - x) \\ \dot{y} = x(\rho - z) - y \\ \dot{z} = xy - \beta z \end{cases} \tag{2.3}$$

onde σ, ρ e β são parâmetros positivos. Para certos valores (por exemplo, $\sigma = 10$, $\rho = 28$, $\beta = 8/3$), esse sistema exibe o característico *atrator de Lorenz* (Figura 2), que marcou o início do estudo detalhado de trajetórias caóticas em sistemas de três dimensões.

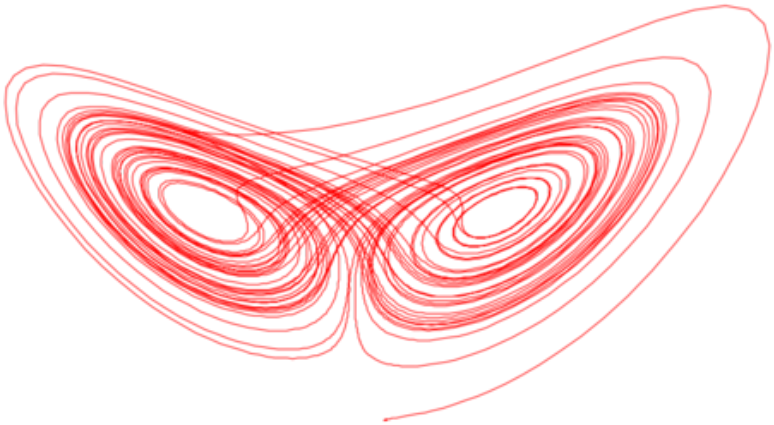

Figura 2 – *Atrator de Lorenz.*

2.1.2 Conceito de Hipercaos

Enquanto em um sistema caótico tipicamente há apenas *um* expoente de Lyapunov positivo, um sistema *hipercaótico* apresenta *dois ou mais* expoentes de Lyapunov positivos. Para analisar os expoentes de Lyapunov, costuma-se linearizar o sistema em torno de uma trajetória e estudar a evolução das variações infinitesimais, governadas pela derivada de $\mathbf{f}$. No caso contínuo, define-se a matriz Jacobiana:

$$\mathbf{J}(\mathbf{x}) = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}) \tag{2.4}$$

Os expoentes de Lyapunov $\{\lambda_i\}_{i=1}^{n}$ são, em essência, valores médios da taxa de expansão ou contração ao longo das direções dos autovetores de $\mathbf{J}$. Matematicamente, um sistema de dimensão n é considerado hipercaótico se:

$$\lambda_1 > \lambda_2 > 0, \quad \lambda_j \in \mathbb{R} \quad (j = 3, \ldots, n) \tag{2.5}$$

onde λ_1 e λ_2 são os primeiros dois maiores expoentes de Lyapunov *positivos*, indicando que o sistema diverge exponencialmente em pelo menos duas direções independentes do espaço de estados. Em geral, requer-se $n \geq 4$ para acomodar ao menos dois expoentes de Lyapunov positivos.

2.1.2.1 Exemplo de Sistema Hipercaótico

Um exemplo simples é o chamado *Sistema Hipercaótico de Rössler* modificado, que introduz termos adicionais para permitir a existência de múltiplos expoentes de Lyapunov positivos:

$$\begin{cases} \dot{x}_1 = -(x_2 + x_3) \\ \dot{x}_2 = x_1 + a\, x_2 \\ \dot{x}_3 = b + x_3(x_1 - c) + x_4 \\ \dot{x}_4 = -d\, x_3 \end{cases} \tag{2.6}$$

onde a, b, c e d são parâmetros ajustáveis. Para certas combinações desses parâmetros, observa-se a existência de dois expoentes de Lyapunov positivos, caracterizando o hipercaos.

2.1.3 Implicações e Aplicações

A presença de múltiplos expoentes positivos torna os sistemas hipercaóticos particularmente atraentes para aplicações que se beneficiam de uma dinâmica rica e não linear. Entre os principais campos de aplicação, destacam-se:

- **Comunicação Segura e Criptografia**: A elevada imprevisibilidade e *pseudoaleatoriedade* dos sinais hipercaóticos pode ser explorada para cifrar mensagens ou mascarar informações.
- **Processamento de Sinais**: Geração de sequências pseudoaleatórias, compressão ou análise de sinais que contenham comportamentos complexos.
- **Modelagem de Fenômenos Naturais**: Fenômenos em fluidodinâmica, química e biologia podem se manifestar como processos hipercaóticos, sobretudo quando existem inúmeras instabilidades acopladas.

2.1.4 Desafios para Controle e Sincronização

Apesar de seu valor científico e tecnológico, a alta complexidade dos sistemas hipercaóticos representa um obstáculo para o desenvolvimento de técnicas de controle e sincronização eficazes. A divergência em múltiplas direções impõe requisitos mais rígidos ao controlador, como:

- **Dimensionamento de Atuadores**: Garantir que mesmo um controle *sub-atuado* (com menos variáveis controláveis do que o total de estados) seja capaz de suprimir ou sincronizar a dinâmica hipercaótica.

- **Observadores Não Lineares**: A construção de observadores mais sofisticados para estimar estados não medidos, dada a possibilidade de crescimento exponencial dos erros em diferentes direções.
- **Tempo de Convergência e Robustez**: As múltiplas instabilidades requerem leis de controle com altas capacidades de reação e adaptação, além de robustez a perturbações externas e incertezas.

Esses fatores motivam a investigação contínua de métodos mais avançados de sincronização, incorporando abordagens como controle adaptativo, inteligência computacional, observadores robustos e arquiteturas de redes neurais cada vez mais elaboradas.

2.2 Teoria de estabilidade de Lyapunov

O estudo de estabilidade em sistemas dinâmicos não lineares é amplamente fundamentado na *Teoria de Lyapunov*, inicialmente desenvolvida por Aleksandr Mikhailovich Lyapunov no final do século XIX. Essa teoria fornece ferramentas conceituais e matemáticas para analisar se um sistema permanece próximo de um ponto de equilíbrio (ou trajetória de equilíbrio) ao sofrer pequenas perturbações em suas condições iniciais (KHALIL, 2009).

2.2.1 Definições de Estabilidade

Considere um sistema dinâmico contínuo descrito por:

$$\dot{\mathbf{x}}(t) = \mathbf{f}\big(\mathbf{x}(t)\big), \quad \mathbf{x}(t) \in \mathbb{R}^n \tag{2.7}$$

onde $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ é uma função contínua, não necessariamente linear. Suponha que exista um ponto de equilíbrio $\mathbf{x}^* \in \mathbb{R}^n$ tal que

$$\mathbf{f}\big(\mathbf{x}^*\big) = \mathbf{0} \tag{2.8}$$

A estabilidade desse ponto de equilíbrio pode ser definida de diferentes maneiras (SLOTINE; LI et al., 1991):

Estabilidade de Lyapunov (no sentido de Lyapunov): O ponto de equilíbrio $\mathbf{x}^*$ é estável se, para todo $\varepsilon > 0$, existe um $\delta > 0$ tal que, sempre que $\|\mathbf{x}(t_0) - \mathbf{x}^*\| < \delta$, então

$$\|\mathbf{x}(t) - \mathbf{x}^*\| < \varepsilon \quad \text{para todo } t \geq t_0$$

Em termos intuitivos, isso significa que, ao iniciar suficientemente próximo de $\mathbf{x}^*$, a trajetória permanece próxima a esse ponto.

Estabilidade Assintótica: O ponto de equilíbrio $\mathbf{x}^*$ é assintoticamente estável se, além de ser estável no sentido de Lyapunov, vale

$$\lim_{t\to\infty} ||\mathbf{x}(t) - \mathbf{x}^*|| = 0$$

isto é, as trajetórias não apenas permanecem próximas de $\mathbf{x}^*$, mas também convergem para ele com o passar do tempo.

Estabilidade Exponencial: O ponto de equilíbrio $\mathbf{x}^*$ é exponencialmente estável se existe um conjunto de constantes positivas $C > 0$ e $\alpha > 0$ tais que:

$$||\mathbf{x}(t) - \mathbf{x}^*|| \leq C\, e^{-\alpha(t-t_0)} ||\mathbf{x}(t_0) - \mathbf{x}^*||$$

para todo $t \geq t_0$. Nesse caso, a convergência até o ponto de equilíbrio se dá em velocidade exponencial.

2.2.2 Método Direto de Lyapunov

O chamado *método direto de Lyapunov* (ou *segunda* forma de Lyapunov) não exige a linearização do sistema e se baseia em uma *função de Lyapunov*, a qual desempenha um papel análogo ao de uma energia potencial. O procedimento consiste em:

1. **Escolher uma Função Candidata:** Seja $V(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ uma função continuamente diferenciável e **positiva definida** na vizinhança de $\mathbf{x}^*$. Isso significa que

   $$V(\mathbf{x}) > 0 \quad \text{para } \mathbf{x} \neq \mathbf{x}^*, \quad \text{e} \quad V(\mathbf{x}^*) = 0$$

2. **Verificar a Derivada de $V(\mathbf{x})$ ao longo das Soluções:** Calcule a derivada de $V(\mathbf{x}(t))$ em relação ao tempo, que pode ser expressa por

   $$\dot{V}(\mathbf{x}) = \nabla V(\mathbf{x})^\top \mathbf{f}(\mathbf{x})$$

   onde $\nabla V(\mathbf{x})$ é o gradiente de V. Analisa-se se $\dot{V}$ é **negativa semidefinida** ou **negativa definida**.

2.2.2.1 Critérios de Estabilidade

Com base na função de Lyapunov, têm-se os seguintes resultados (KHALIL, 2009):

- Se $V(\mathbf{x})$ é positiva definida e $\dot{V}(\mathbf{x})$ é **negativa semidefinida** em uma vizinhança de $\mathbf{x}^*$, então $\mathbf{x}^*$ é **estável** no sentido de Lyapunov.
- Se $V(\mathbf{x})$ é positiva definida e $\dot{V}(\mathbf{x})$ é **negativa definida**, então $\mathbf{x}^*$ é **assintoticamente estável**.
- Se, além disso, existem constantes $\alpha_1, \alpha_2, \alpha_3 > 0$ tais que

  $$\alpha_1 ||\mathbf{x}||^p \leq V(\mathbf{x}) \leq \alpha_2 ||\mathbf{x}||^q, \quad \dot{V}(\mathbf{x}) \leq -\alpha_3 ||\mathbf{x}||^r$$

  para algumas potências $p, q, r > 0$, pode-se demonstrar **estabilidade exponencial**.

2.2.3 Método Indireto de Lyapunov (Linearização)

A *primeira* abordagem de Lyapunov, também conhecida como *método indireto*, baseia-se na análise de autovalores da *parte linearizada* do sistema em torno do ponto de equilíbrio. Para o sistema (2.7), lineariza-se $\mathbf{f}(\mathbf{x})$ na vizinhança de $\mathbf{x}^*$:

$$\mathbf{A} = \left.\frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^*} \tag{2.9}$$

Se todas as partes reais dos autovalores de $\mathbf{A}$ forem negativas, conclui-se que $\mathbf{x}^*$ é **assintoticamente estável** para o sistema não linear. Entretanto, caso haja autovalores com parte real positiva ou nula, não se podem tirar conclusões definitivas sobre a estabilidade não linear apenas com esse método (SLOTINE; LI et al., 1991).

2.2.4 Aplicações em Sistemas Caóticos e Hipercaóticos

Em sistemas *caóticos* ou *hipercaóticos*, a aplicação da Teoria de Lyapunov desempenha um papel crucial na análise de *convergência* em problemas de *sincronização*. Por exemplo, em esquemas de controle que buscam sincronizar duas réplicas de um sistema hipercaótico, costuma-se projetar um controlador ou observador cujas leis de realimentação garantam a dissipação da chamada *função de Lyapunov* associada ao erro de sincronização (CHEN, G., 1999). Dessa maneira, demonstra-se formalmente que, mesmo diante da alta complexidade e do número de expoentes de Lyapunov positivos, a dinâmica de erro converge a zero, assegurando a sincronização.

Além disso, a escolha apropriada da função de Lyapunov é muitas vezes inspirada em formas quadráticas ou combinações polinomiais específicas, adaptadas às não linearidades presentes. Em alguns estudos, termos adicionais ou pesos variáveis são introduzidos para lidar com o *hipercaos* de maneira eficiente. Na sub-seção 2.2.1 se define a Estabilidade Exponencial que se revela particularmente relevante nesses cenários, pois o tempo de convergência assume papel fundamental em aplicações práticas, tais como comunicação segura e criptografia baseadas em caos.

Resumo

Em síntese, a Teoria de Lyapunov oferece uma estrutura robusta para analisar a estabilidade de pontos de equilíbrio e trajetórias de sistemas não lineares, sem recorrer somente à linearização ou a simplificações excessivas. Para fins de sincronização de sistemas caóticos e hipercaóticos — abordagem central deste trabalho — a formulação de uma função de Lyapunov adequada e a verificação de sua derivada negativa são etapas fundamentais na demonstração formal de estabilidade e convergência dos erros de sincronização. Essas

ferramentas matemáticas serão utilizadas nos capítulos subsequentes para desenvolver e validar o controlador proposto.

2.3 Desigualdade de Young

A *Desigualdade de Young* constitui uma ferramenta matemática relevante para manipular e estimar produtos de termos em diversas áreas da análise, incluindo sistemas de controle e teoria de estabilidade. Em sua forma mais simples, para números reais não negativos, ela estabelece que:

$$ab \leq \frac{a^p}{p} + \frac{b^q}{q} \tag{2.10}$$

onde $a, b \geq 0$ e $p, q > 1$ satisfazem

$$\frac{1}{p} + \frac{1}{q} = 1$$

Esse resultado é particularmente útil ao se lidarem com termos de produto que aparecem em equações diferenciais, principalmente em provas de estabilidade via métodos de Lyapunov (KHALIL, 2009).

2.3.1 Justificativa Geral

A idéia central por trás da Desigualdade de Young é fornecer uma maneira de majorar ab por uma expressão que separa as variáveis a e b, cada uma elevada a um expoente compatível. Dessa forma, é possível controlar cada termo de modo individual, o que se mostra conveniente ao construir funções de Lyapunov ou ao analisar sistemas com múltiplos graus de liberdade.

2.3.2 Aplicação em Sistemas Dinâmicos

Em sistemas dinâmicos — especialmente os não lineares — a Desigualdade de Young geralmente aparece quando se deseja impor limites sobre termos cruzados na derivada de uma função de Lyapunov. Por exemplo, ao avaliar $\dot{V}(\mathbf{x})$, podem surgir produtos como $x\,y$, que dificultam a análise do sinal dessa derivada. Por meio da Desigualdade de Young, esse produto pode ser substituído por uma soma de potências separadas de x e y, tornando mais clara a análise de negatividade de $\dot{V}$, além de possibilitar o uso de outros argumentos ou parâmetros de controle para garantir a estabilidade.

2.3.2.1 Exemplo Simples

Seja o termo ab, com a,$b \geq 0$. Escolhendo $p = q = 2$ (pois $1/p + 1/q = 1$), obtém-se:

$$ab \leq \frac{a^2}{2} + \frac{b^2}{2}$$

Dessa forma, “separa-se” o produto em duas parcelas quadráticas. Em provas de estabilidade, esse procedimento costuma facilitar a análise, pois a derivada de Lyapunov pode ser somada a outras parcelas quadráticas já presentes, contribuindo para estabelecer a negatividade de $\dot{V}(\mathbf{x})$.

2.3.3 Extensões e Outras Versões

Existem versões mais gerais da Desigualdade de Young, incluindo formas integrais ou conjugadas, que aparecem na análise de convoluções e transformadas de Fourier, bem como em aplicações que envolvem espaços de funções L^p. Em controle adaptativo, a Desigualdade de Young é frequentemente usada em conjunto com a Desigualdade de Cauchy-Schwarz para tratar termos de ajuste e incertezas nos parâmetros.

Em todos esses cenários, a utilidade fundamental da Desigualdade de Young reside em quebrar produtos de difícil manipulação em partes independentes que podem ser controladas ou estimadas separadamente. Essa característica faz dela um artifício matemático poderoso em demonstrações de estabilidade, otimização e análise de sistemas não lineares.

2.4 Escalonamento em Amplitude e na Frequência

O escalonamento em amplitude e na frequência é uma técnica frequentemente utilizada para ajustar sinais ou funções, de modo a tornar sua análise e/ou implementação mais simples e estável (OPPENHEIM; VERGHESE, 2017). Em aplicações de controle de sistemas dinâmicos, essas transformações podem servir, por exemplo, para:

- evitar saturações ou superaquecimentos ao manter valores de amplitude dentro de uma faixa segura;
- facilitar a análise em torno de uma frequência específica de interesse, reescalonando a dinâmica temporal para evidenciar períodos e características relevantes;
- ajustar ganhos de malha de controle em função de amplitude e frequência, garantindo melhor desempenho e robustez.

2.4.1 Escalonamento em Amplitude

O *escalonamento em amplitude* (por vezes chamado de “normalização” ou “ajuste de magnitude”) consiste em multiplicar o sinal ou variável de estado por um fator constante, A. Se $x(t)$ é o sinal original, o sinal escalonado pode ser expresso como:

$$x_{\text{esc}}(t) \;=\; A \cdot x(t)$$

- Se $A > 1$, o sinal sofre *amplificação*, aumentando proporcionalmente toda a sua faixa de valores.
- Se $0 < A < 1$, há *atenuação*, útil para manter o sinal dentro de limites de hardware ou dentro de modelos analíticos lineares aproximados.

Em muitos problemas de controle não linear, esse escalonamento é crucial para evitar que a variável controlada exceda os limites físicos de sensores ou atuadores, ou para simplificar as condições iniciais em simulações numéricas.

2.4.1.1 Exemplo – Sistema de Lorenz:

Considere as variáveis do sistema de Lorenz (caótico) que podem atingir amplitudes muito elevadas para certos valores do parâmetro ρ. Uma abordagem para contornar saturações de atuadores é:

$$x_{\text{esc}}(t) = \frac{1}{M}x(t) \quad y_{\text{esc}}(t) = \frac{1}{M}y(t) \quad z_{\text{esc}}(t) = \frac{1}{M}z(t)$$

onde M é escolhido de modo que as novas variáveis $x_{\text{esc}}, y_{\text{esc}}, z_{\text{esc}}$ permaneçam em uma faixa segura (por exemplo, entre -1 e 1). Isso não altera a natureza qualitativa do caos, mas diminui a escala dos valores para algo mais manejável.

2.4.2 Escalonamento na Frequência (ou no Tempo)

O *escalonamento na frequência* envolve a modificação da escala de tempo na qual o sistema é observado ou controlado. Se definimos $\tau = \alpha\, t$ com $\alpha > 0$, então um sinal $x(t)$ transformado na nova escala fica:

$$x_{\text{esc}}(\tau) = x\Big(\frac{\tau}{\alpha}\Big)$$

- Se $\alpha > 1$, o sistema é acelerado, equivalendo a compressão na escala de tempo (aumentando a frequência aparente das oscilações).
- Se $0 < \alpha < 1$, há uma desaceleração, alongando a evolução no tempo (reduzindo a frequência).

Esse procedimento pode ser útil para:

- *análise de fenômenos rápidos ou lentos*: realçar as oscilações de interesse, seja para observar altas frequências ou estudar dinâmicas que se desenvolvem muito devagar;
- *sintonia de controladores*: ao retardar virtualmente um sistema muito rápido, pode-se facilitar a identificação ou a implementação de leis de controle. Em sentido oposto, acelerar a dinâmica ajuda a estudar sistemas muito lentos.

2.4.2.1 Exemplo – Oscilador de Van der Pol:

O oscilador de Van der Pol pode ser descrito por:

$$\dot{x} = y, \quad \dot{y} = \mu\left(1 - x^2\right) y - x$$

Para μ muito grande, a convergência para a órbita periódica é muito rápida, tornando a simulação rígida. Se definimos $\tau = \mu t$, a evolução na variável τ se torna mais lenta, permitindo estudos de estabilidade e controle sem problemas de rigidez nos métodos numéricos.

2.4.3 Combinação de Escalonamentos

Em muitos casos práticos, é necessário combinar tanto o escalonamento em amplitude quanto o escalonamento na frequência. Isso ocorre, por exemplo, se há interesse em:

- normalizar estados em uma faixa específica, como $[-1, 1]$,
- ajustar a dinâmica temporal a um intervalo conveniente para medição ou para o algoritmo de controle.

2.4.3.1 Exemplo – Sistema Hipercaótico 4D:

Para um sistema hipercaótico tetradimensional em que as variáveis x_1, x_2, x_3, x_4 podem assumir valores elevados e oscilar em diversas frequências, pode-se definir:

- Fatores de amplitude γ_i para cada estado, a fim de mantê-los em faixas adequadas;
- Uma transformação de tempo $\tau = \alpha t$, compressora ou expansora, para realçar as dinâmicas relevantes.

Tais procedimentos tornam o sistema mais seguro para implementação em hardware e também facilitam a análise de controlabilidade e estabilidade.

Síntese

Em síntese, o escalonamento em amplitude e na frequência representa uma estratégia versátil para refinar o desempenho de simulações, contornar limitações físicas e ajustar diferentes modos dinâmicos. Como resultado, essas técnicas são adotadas tanto em pesquisas teóricas — a fim de facilitar análises de estabilidade e controle — quanto em implementações práticas, auxiliando na proteção de equipamentos e na otimização de recursos.

2.5 Correlação Não Linear de Kendall

Em várias situações de análise de dados e sinais, interessa avaliar se duas variáveis (ou sinais) apresentam uma relação *monotônica*, isto é, se é possível descrever uma tendência de crescimento ou decrescimento conjunto sem exigir linearidade estrita. Um método apropriado para isso é o *coeficiente de correlação de Kendall*, comumente denotado τ (tau). Diferentemente das técnicas de correlação lineares, Kendall captura relações monotônicas mais gerais e é considerada mais robusta em certos cenários estatísticos (KENDALL, 1938; CONOVER, 1999).

2.5.1 Definição e Interpretação

Considere um conjunto de n observações pareadas $\{(x_i, y_i)\}_{i=1}^{n}$. Diz-se que um par (x_i, y_i) e (x_j, y_j) é *concordante* se $x_i < x_j$ e $y_i < y_j$, ou se $x_i > x_j$ e $y_i > y_j$. Caso sejam "invertidos" (por exemplo, $x_i < x_j$ mas $y_i > y_j$), o par é dito *discordante*. Então, define-se o coeficiente τ de Kendall como

$$\tau_{\text{Kendall}} = \frac{(\text{número de pares concordantes}) - (\text{número de pares discordantes})}{\binom{n}{2}} \tag{2.11}$$

onde $\binom{n}{2} = \frac{n(n-1)}{2}$ é o total de pares possíveis. Assim, τ_{Kendall} varia de -1 (quando todos os pares são discordantes) até $+1$ (quando todos são concordantes). Valores próximos de 0 indicam que não há relação monotônica clara entre x e y.

2.5.2 Aplicação em Sistemas Dinâmicos e Criptografia

No contexto de sinais dinâmicos ou mensagens cifradas, a correlação de Kendall pode ser utilizada para aferir se dois sinais apresentam uma relação monotônica (ou se estão essencialmente "desalinhados"). Isso é particularmente útil em situações em que:

- **As variáveis podem ter relação não linear**: Quando suspeita-se que a mensagem cifrada ainda mantenha algum padrão com a mensagem original, mesmo que esse padrão não seja linear, τ_{Kendall} pode detectar essa dependência.
- **Deseja-se avaliar pequenas amostras**: O cálculo de Kendall tende a ser mais robusto em amostras menores, quando comparado a outras técnicas de correlações não lineares.

Em *comunicação segura*, se a correlação de Kendall entre a mensagem original e a mensagem cifrada é próxima de zero, isso sugere que não há relação monotônica residual. Assim, um interceptador sem a "chave" ou sem o conhecimento do esquema de sincronização encontrará dificuldade em extrair padrões que revelem a mensagem original.

Conclusão

De modo geral, o coeficiente de Kendall se mostra apropriado para quantificar a existência de qualquer relação monotônica (não necessariamente linear) entre dois sinais. Para este trabalho, avaliar τ_{Kendall} entre as mensagens originais e as mensagens criptografadas pode confirmar a ausência de padrões e a robustez da técnica de cifragem. Se os valores de τ_{Kendall} forem efetivamente reduzidos (próximos a 0), então a cifragem baseada em caos estará bem sucedida.

2.6 Entropia de Sinais

A entropia é uma medida fundamental no contexto da *teoria da informação* e serve para quantificar o grau de incerteza ou aleatoriedade presente em um conjunto de dados ou em um sinal. Em essência, quanto maior a entropia, mais imprevisível é a distribuição dos valores assumidos pelo sinal (SHANNON, 1948; COVER, 1999).

2.6.1 Definição de Shannon

Claude E. Shannon, em seu trabalho pioneiro sobre teoria da informação, definiu a entropia de uma variável aleatória X que pode assumir valores $\{x_1, x_2, \ldots, x_n\}$ com probabilidades $\{p_1, p_2, \ldots, p_n\}$ como:

$$H(X) \;=\; -\sum_{i=1}^{n} p_i \,\log_2\, p_i. \tag{2.12}$$

Essa expressão reflete a quantidade média de informação necessária para descrever o resultado de X. Quando todos os p_i são iguais, X é totalmente imprevisível — a entropia é máxima; se, ao contrário, X assume certo valor com probabilidade próxima de 1, há pouca incerteza, e a entropia se torna baixa.

2.6.2 Entropia Diferencial

A entropia discutida por Shannon para variáveis aleatórias discretas mede a incerteza em termos de uma soma sobre probabilidades. Para variáveis *contínuas*, em que a probabilidade é descrita por uma densidade $f_X(x)$, a noção análoga recebe o nome de *entropia diferencial*.[1]

[1] Em alguns textos também chamada de *entropia contínua*. A formulação foi introduzida pelo próprio Shannon na extensão contínua de sua teoria de informação.

2.6.2.1 Definição

Seja X uma variável aleatória contínua com densidade de probabilidade $f_X(x)$ definida em $\mathbb{R}$. A **entropia diferencial** de X é definida por

$$h(X) \;=\; -\int_{-\infty}^{+\infty} f_X(x)\,\log_2 f_X(x)\,dx. \tag{2.13}$$

O resultado é expresso em *bits* quando se usa logaritmo de base 2 (pode-se usar logaritmo natural, obtendo-se o valor em *nats*).

2.6.2.2 Observações Importantes

- Diferentemente da entropia discreta, $h(X)$ *pode assumir valores negativos*. Isso ocorre porque a densidade de probabilidade pode ser maior que 1 em regiões muito concentradas, de modo que $\log_2 f_X(x)$ se torna positivo.
- A entropia diferencial *não* é invariante a mudanças de escala. Se $Y = aX$ com $a \neq 1$, então $h(Y) = h(X) + \log_2 |a|$.
- Apesar dessas particularidades, $h(X)$ mantém várias propriedades úteis: maximiza-se para a densidade Gaussiana com variância fixa, aparece em limites de capacidade de canal e na definição de *informação mútua* contínua.

2.6.2.3 Interpretação em Processamento de Sinais

Em sinais analógicos pode-se estimar $h(X)$ a partir de amostras, utilizando histogramas finos ou técnicas de *kernel density estimation*. Valores altos de entropia diferencial sugerem sinais amplamente dispersos ou ruidosos; valores baixos indicam sinais mais concentrados ou previsíveis.

2.6.2.4 Aplicação a Sinais Caóticos

Sinais provenientes de sistemas caóticos ou hipercaóticos tendem a ocupar regiões complexas no espaço de fase, produzindo distribuições de amplitude com maior dispersão. Assim, espera-se que apresentem entropia diferencial elevada em comparação a sinais determinísticos regulares. Na prática:

- Um sinal caótico usado para mascarar uma mensagem deve manter ou aumentar $h(X)$, dificultando a detecção de padrões pelo adversário.
- A comparação de $h(X)$ *antes* e *depois* da inserção da mensagem permite avaliar se o processo de cifragem preserva a “aleatoriedade aparente” do sinal.

2.6.3 Relevância no Trabalho

Para este estudo, a análise de entropia pode servir como uma métrica de "qualidade" ou "força" do mascaramento de sinais. Se a entropia do sinal criptografado (resultante do caos + mensagem) permanecer elevada, então um interceptador (sem acesso aos parâmetros de sincronização) terá maior dificuldade em distinguir o conteúdo original. Além disso, a entropia pode indicar se a técnica de cifragem baseada em caos está produzindo um sinal adequadamente complexo, reduzindo a sua correlação com a mensagem original.

Em suma, a entropia fornece uma perspectiva quantitativa do quão imprevisível é um sinal. No âmbito de sistemas hipercaóticos e criptografia, manter ou aumentar a entropia do sinal é desejável para dificultar ataques de força bruta ou análises estatísticas que possam expor o conteúdo ou a chave de cifragem.

2.7 Redes Neurais Artificiais

As *Redes Neurais Artificiais (RNAs)* são estruturas computacionais inspiradas no comportamento do sistema nervoso biológico. Seu princípio fundamental consiste em combinar diversos "neurônios artificiais" interconectados, de modo a aprender padrões complexos de mapeamento entre entradas e saídas, sem a necessidade de especificar explicitamente um modelo matemático detalhado (HAYKIN, 1998).

2.7.1 Arquitetura Básica

Uma rede neural típica *feedforward* (FNN) pode ser organizada em *camadas*: uma camada de entrada, uma ou mais camadas intermediárias (ocultas) e uma camada de saída. Cada neurônio em uma camada oculta recebe entradas de todos (ou parte) dos neurônios da camada anterior, computa uma soma ponderada e aplica uma *função de ativação não linear* para produzir sua saída.

Considere $\mathbf{x} \in \mathbb{R}^n$ como o vetor de entrada da rede, e $\mathbf{y} \in \mathbb{R}^m$ como o vetor de saída. Em uma RNA com apenas uma camada oculta, cada neurônio da camada oculta efetua o cálculo:

$$z_j = \sum_{i=1}^{n} w_{ji}^{(1)} x_i + b_j^{(1)}, \quad \text{para } j = 1,2,\ldots,h,$$

onde h é o número de neurônios na camada oculta, $w_{ji}^{(1)}$ são os pesos que ligam a i-ésima entrada ao j-ésimo neurônio, e $b_j^{(1)}$ é o termo de *bias* do j-ésimo neurônio da primeira camada. Em seguida, uma função de ativação $S(\cdot)$ (ver Seção 2.7.2) é aplicada a cada z_j, gerando uma saída $\sigma_j = S(z_j)$.

Por fim, a camada de saída efetua nova soma ponderada das σ_j:

$$y_k \;=\; \sum_{j=1}^{h} w_{kj}^{(2)}\,\sigma_j \;+\; b_k^{(2)}, \quad \text{para } k = 1,2,\ldots,m,$$

onde $w_{kj}^{(2)}$ e $b_k^{(2)}$ são pesos e *bias* associados à camada de saída. Assim, o conjunto de parâmetros $\mathbf{W}$ e $\mathbf{b}$ engloba todos esses valores, constituindo o "conhecimento" da rede neural.

2.7.2 Notação e Definições Matemáticas

Nesta dissertação, adotaremos a seguinte *notação* para representar os pesos da rede neural ao longo das derivações de controle e sincronização:

- $\mathbf{W}^*$: denota o conjunto de *pesos ideais* ou ótimos, isto é, aquele que melhor aproxima a função alvo em um sentido teórico (por exemplo, sob certas hipóteses do Teorema da Aproximação Universal).
- $\widehat{\mathbf{W}}$: corresponde ao conjunto de *pesos estimados* em tempo real, que são adaptados por um algoritmo de aprendizado ou de controle adaptativo. Em outras palavras, $\widehat{\mathbf{W}}$ é a estimativa dos pesos que a rede possivelmente converge para se aproximar de $\mathbf{W}^*$.
- $\widetilde{\mathbf{W}}$: representa o conjunto de *erro de aproximação* entre os *pesos estimados* e os *pesos ideais* da rede neural. Usualmente, $\widetilde{\mathbf{W}} = \widehat{\mathbf{W}} - \mathbf{W}^*$.

Além disso, definimos a *função sigmoidal* (ou logística) $S : \mathbb{R} \to \mathbb{R}$, cujo papel é introduzir não linearidade na rede:

$$S(u) \;=\; \frac{1}{1+e^{-u}}.$$

Em alguns casos, também é conveniente adotar a forma derivada dessa função (para métodos de retropropagação), notada por:

$$S'(u) \;=\; S(u)\big(1 - S(u)\big).$$

Para redes com múltiplas camadas, a ideia é similar: cada camada ℓ terá pesos $\mathbf{W}^{(\ell)}$ e *bias* $\mathbf{b}^{(\ell)}$. No contexto específico desta dissertação, os símbolos $\mathbf{W}^*$, $\widetilde{\mathbf{W}}$ e $\widehat{\mathbf{W}}$ serão utilizados para destacar o papel de cada conjunto de pesos no esquema de controle adaptativo/neural.

2.7.3 Capacidade de Aproximação

Um resultado importante para o uso de RNAs em controle não linear e sincronização de sistemas (hiper)caóticos é o *Teorema da Aproximação Universal*. Em termos gerais, ele afirma que, sob certas condições (por exemplo, função de ativação contínua e ao menos uma

camada oculta com número suficiente de neurônios), uma rede neural pode aproximar arbitrariamente bem qualquer função contínua definida em um conjunto compacto (HORNIK; STINCHCOMBE; WHITE, 1989).

Essa *capacidade de aproximação universal* justifica empregar RNAs para lidar com incertezas não lineares em sistemas dinâmicos complexos, pois podemos presumir a existência de um conjunto de pesos $\mathbf{W}^*$ capaz de representar (com erro arbitrariamente pequeno) a dinâmica ou a função desconhecida.

2.7.4 Aprendizado e Ajuste dos Pesos

Para estimar $\widehat{\mathbf{W}}$ em direção a $\mathbf{W}^*$, usam-se processos de aprendizado baseados em métodos de gradiente ou em estratégias adaptativas. Se a rede for aplicada ao controle de um sistema, um algoritmo de adaptação *on-line* pode atuar sobre $\widehat{\mathbf{W}}$ de forma que a lei de controle dependa tanto das medições de saída quanto de alguma função de erro. Esse processo pode ser descrito genericamente por:

$$\dot{\widehat{\mathbf{W}}} \;=\; \Gamma\,\phi\big(\mathbf{x},\widehat{\mathbf{W}}\big),$$

onde Γ é uma matriz (ou ganho) de adaptação, e $\phi\big(\mathbf{x},\widehat{\mathbf{W}}\big)$ representa uma função de correção dos pesos, derivada de objetivos como minimizar um erro de aproximação ou garantir a estabilidade de Lyapunov do erro de controle.

2.7.5 Aplicações em Sistemas de Alta Complexidade

Redes neurais são especialmente promissoras em *sistemas caóticos e hipercaóticos*, pois:

- Permitem **aproximar** relações não lineares entre variáveis de estado e incertezas, fundamental para lidar com vários expoentes de Lyapunov positivos.
- Podem ser **treinadas on-line**, ajustando-se dinamicamente aos parâmetros do sistema que mudam ao longo do tempo.
- **Integram-se bem** a métodos de controle robusto ou sub-atuado, quando se deseja projetar leis de controle que não dependam de um modelo matemático exato do sistema.

Em síntese, ao longo deste trabalho, utilizaremos as notações $\mathbf{W}^*$, $\widehat{\mathbf{W}}$ e $\widetilde{\mathbf{W}}$ para representar, respectivamente, os pesos *ideais*, pesos *estimados* e *erro de aproximação* de uma rede neural. A função de ativação sigmoidal será denotada por $S(\cdot)$. Em capítulos posteriores, esses elementos serão fundamentais para a formulação do *controlador neural e sub-atuado*, garantindo simultaneamente a *aproximação* de incertezas e a *estabilidade* de um sistema hipercaótico de quatro dimensões.

3 Proposta de sincronização neural e sub-atuada para um sistema hipercaótico tetradimensional

3.1 Descrição do Problema

Nesta seção, propõe-se uma abordagem de *sincronização neural e sub-atuada* para um *sistema hipercaótico tetradimensional*. Primeiramente, apresenta-se o modelo original, conforme descrito em (WANG, S., 2022), e, na sequência, introduzem-se as versões *mestre* e *escravo* obtidas por meio de escalonamento e acréscimo de termos adicionais para controle e incertezas.

3.1.1 Sistema Hipercaótico Proposto por (WANG, S., 2022)

O sistema em estudo é composto por quatro variáveis de estado, $x(t)$, $y(t)$, $z(t)$ e $w(t)$. Suas equações de evolução são dadas por:

$$\begin{cases} \dot{x} = a(y - x) + gyz \\ \dot{y} = cx - dxz + y + w \\ \dot{z} = xy - bz \\ \dot{w} = -fy \end{cases} \tag{3.1}$$

onde $a = 35, b = 4.9, c = 25, d = 5, g = 35$ e $f = 100$ são constantes. Esse modelo apresenta múltiplos expoentes de Lyapunov positivos, caracterizando um regime hipercaótico. Como consequência, as trajetórias exibem elevada sensibilidade às condições iniciais, o que o torna adequado para avaliar métodos de sincronização em cenários de alta complexidade.

3.1.2 Planta Mestre (Versão Escalonada)

Para fins de controle e sincronização, considera-se uma forma *escalonada* do sistema, a fim de manter as variáveis em faixas adequadas e facilitar a análise teórica. A *planta mestre* passa a ser descrita pelas variáveis (x_m, y_m, z_m, w_m), que obedecem:

$$\begin{cases} \dot{x}_m = a(y_m - x_m) + \gamma_1 g y_m z_m \\ \dot{y}_m = c x_m - \gamma_2 d x_m z_m + y_m + w_m \\ \dot{z}_m = \gamma_3 x_m y_m - b z_m \\ \dot{w}_m = -f y_m \end{cases} \tag{3.2}$$

onde γ_1, γ_2, e γ_3 são fatores de escalonamento que não alteram a natureza qualitativa do hipercaos, mas podem melhorar a robustez ou a viabilidade numérica.

3.1.3 Planta Escravo (Versão Escalonada com Perturbações e Controle)

A *planta escravo* é construída de modo análogo, mas inclui termos adicionais relativos a incertezas e sinais de controle. Supondo as variáveis (x_s, y_s, z_s, w_s), o sistema segue:

$$\begin{cases} \dot{x}_s = a(y_s - x_s) + \gamma_1 g y_s z_s + d_1 \\ \dot{y}_s = c x_s - \gamma_2 d x_s z_s + y_s + w_s + d_2 + u_2 \\ \dot{z}_s = \gamma_3 x_s y_s - b z_s + d_3 \\ \dot{w}_s = -f y_s + d_4 + u_4 \end{cases} \tag{3.3}$$

onde:

- d_1, d_2, d_3, d_4 representam distúrbios externos ou perturbações que dependem do tempo e dos estados do sistema mestre e escravo.
- u_2, u_4 constituem as variáveis de *controle* (sub-atuado e neural), projetadas de modo a forçar o escravo a seguir as trajetórias do mestre.

3.1.4 Objetivo Geral de Sincronização

Com as duas versões (mestre e escravo), define-se formalmente um problema de *sincronização*: deseja-se projetar leis de controle para u_2, u_4 que garantam que ($x_s(t)$, $y_s(t)$, $z_s(t)$,$w_s(t)$) acompanhem ($x_m(t)$, $y_m(t)$, $z_m(t)$, $w_m(t)$) ao longo do tempo. Em outras palavras, objetiva-se que o *erro de sincronização* convirja para valores próximos de zero, mesmo na presença de incertezas, perturbações e não linearidades.

Nas seções seguintes, serão detalhados:

- A **função de erro** que quantifica a distância entre as variáveis mestre e escravo.
- O **controlador neural e sub-atuado**, cujo núcleo está na adaptação dos pesos neurais ($\widehat{\mathbf{W}}$) e na atuação parcial para otimizar custos e recursos.

- A **prova de estabilidade**, baseada em métodos de Lyapunov e na Desigualdade de Young, para assegurar convergência e robustez.

Dessa forma, o sistema hipercaótico tetradimensional aqui descrito, decomposto em plantas mestre e escravo escalonadas, servirá de base para a validação da estratégia de controle, demonstrando a viabilidade de *sincronizar* dinâmicas altamente instáveis por meio de redes neurais e de uma lei de controle sub-atuada.

3.2 Prova de estabilidade usando a Teoria de Lyapunov

Conforme apresentado na Seção 3.1, nosso objetivo é garantir que o *erro de sincronização* entre o sistema mestre e o escravo permaneça limitado e em valores próximos de zero. Diferentemente de uma convergência assintótica estrita, mostraremos aqui que, ao satisfazer certas condições, o erro e os parâmetros neurais permanecem em uma vizinhança (ou conjunto compacto) que asseguram a *estabilidade prática* do sistema.

3.2.1 Função de Erro

Sejam as variáveis do **sistema mestre** denotadas por (x_m, y_m, z_m, w_m) e as do **sistema escravo** por (x_s, y_s, z_s, w_s). Definimos então o *erro de sincronização*:

$$\begin{aligned} e_1 &= x_s - x_m \\ e_2 &= y_s - y_m \\ e_3 &= z_s - z_m \\ e_4 &= w_s - w_m \end{aligned} \tag{3.4}$$

Nosso intuito é mostrar que $\mathbf{e} = (e_1, e_2, e_3, e_4)$ permanece dentro de uma faixa pequena, ainda que perturbações e incertezas impeçam a convergência exata a $\mathbf{0}$. Em outras palavras, desejamos demonstrar a existência de um *conjunto* Ω ao redor de $\mathbf{0}$ para o qual o erro seja *limitado*, mantendo o escravo *próximo* do mestre em regime permanente.

Para atingir este objetivo, realiza-se as seguintes aproximações neurais:

$$\begin{aligned} \mathbf{W}_2^{*T} S_2 + \varepsilon_2 &= c x_s + y_s + w_s + d_2 \\ \mathbf{W}_4^{*T} S_4 + \varepsilon_4 &= -f y_s + d_4 \end{aligned} \tag{3.5}$$

onde:

- $\mathbf{W}_2^*$ e $\mathbf{W}_4^*$ são *pesos ideais* de uma rede neural, empregados para tratar incertezas ou dinâmicas não modeladas.
- S_2, S_4 denotam as funções sigmoidais dos neurônios da rede.

- $\varepsilon_2, \varepsilon_4$ são erros de aproximação ou termos auxiliares de aprendizado.

Derivando-se as equações de erro e substituindo os valores de 3.2, 3.3 e 3.5, obtém-se as seguintes equações para a *dinâmica dos erros*:

$$\begin{aligned}
\dot{e}_1 &= \dot{x}_s - \dot{x}_m = ae_2 - ae_1 + \gamma_1 g\Big(e_2e_3 + y_me_3 + z_me_2\Big) + d_1 \\
\dot{e}_2 &= \dot{y}_s - \dot{y}_m = \mathbf{W}_2^{*T}S_2 + \varepsilon_2 - cx_m - \gamma_2 d\big(e_1e_3 + x_me_3 + z_me_1\big) - y_m - w_m + u_2 \\
\dot{e}_3 &= \dot{z}_s - \dot{z}_m = \gamma_3\big(e_1e_2 + x_me_2 + y_me_1\big) - be_3 + d_3 \\
\dot{e}_4 &= \dot{w}_s - \dot{w}_m = \mathbf{W}_4^{*T}\,S_4 + \varepsilon_4 + fy_m + u_4
\end{aligned} \tag{3.6}$$

3.2.2 Controlador Neural e Sub-atuado

Neste trabalho, propõe-se um **controlador sub-atuado** que atua apenas em algumas das equações (neste caso, $\dot{y}_s$ e $\dot{w}_s$), deixando as demais variáveis livres ou apenas escalonadas. Adicionalmente, utilizam-se **redes neurais** para aproximar termos não lineares desconhecidos.

De modo geral, define-se:

$$u_2 = -\lambda_2 e_2 - \widehat{\mathbf{W}}_2^T S_2, \quad u_4 = -\lambda_4 e_4 - \widehat{\mathbf{W}}_4^T S_4 \tag{3.7}$$

onde λ_2 e λ_4 são constantes de realimentação de erro, e $\widehat{\mathbf{W}}_2$ e $\widehat{\mathbf{W}}_4$ são estimativas dos pesos ideais $\mathbf{W}_2^*$ e $\mathbf{W}_4^*$, respectivamente.

A ideia principal é que:

- Os ganhos λ_2 e λ_4 supram a parte linear do erro, garantindo *amortecimento* e um caminho para a estabilidade.
- As redes neurais $\widehat{\mathbf{W}}_2$ e $\widehat{\mathbf{W}}_4$ adaptam-se para compensar não linearidades e incertezas, via um *mecanismo de atualização* definido posteriormente na prova de Lyapunov.
- O controle seja *sub-atuado*, ou seja, não atue explicitamente em $\dot{x}_s$ e $\dot{z}_s$, com o objetivo de reduzir os custos e a complexidade da implementação, sem inviabilizar a sincronização.

3.2.3 Prova de Estabilidade

3.2.3.1 Função Candidata de Lyapunov

Para verificar a estabilidade (no sentido de manter o erro em uma vizinhança próxima de $\mathbf{0}$), definimos a seguinte *função candidata de Lyapunov*:

$$V\big(\mathbf{e}, \widetilde{\mathbf{W}}_2, \widetilde{\mathbf{W}}_4\big) = \frac{1}{2}\Big(e_1^2 + e_2^2 + e_3^2 + e_4^2\Big) + \frac{1}{2}\Big(\|\widetilde{\mathbf{W}}_2\|^2 + \|\widetilde{\mathbf{W}}_4\|^2\Big) \tag{3.8}$$

onde

$$\widetilde{\mathbf{W}}_2 = \widehat{\mathbf{W}}_2 - \mathbf{W}_2^*, \quad \widetilde{\mathbf{W}}_4 = \widehat{\mathbf{W}}_4 - \mathbf{W}_4^* \tag{3.9}$$

representam os *erros de estimação dos pesos* da rede neural e

$$\|\widetilde{\mathbf{W}}_2\|^2 = \widetilde{\mathbf{W}}_2^T \widetilde{\mathbf{W}}_2, \quad \|\widetilde{\mathbf{W}}_4\|^2 = \widetilde{\mathbf{W}}_4^T \widetilde{\mathbf{W}}_4$$

Observe que V é positivo definido em torno de $\mathbf{e} = \mathbf{0}$ e $\widetilde{\mathbf{W}}_n = \mathbf{0}$, satisfazendo o critério básico para funções de Lyapunov.

3.2.3.2 Derivada de Lyapunov e Uso das Desigualdades de Young

Calculemos $\dot{V}$ diferenciando cada parcela em função de $\dot{\mathbf{e}}$, $\dot{\widetilde{\mathbf{W}}}_2$ e $\dot{\widetilde{\mathbf{W}}}_4$. Após substituir as dinâmicas do sistema (mestre e escravo), obtemos a seguinte expressão:

$$\begin{aligned}
\dot{V} = {} & a e_1 e_2 - a e_1^2 + \gamma_1 g e_1 e_2 e_3 + \gamma_1 g y_m e_1 e_3 + \gamma_1 g z_m e_1 e_2 + e_1 d_1 \\
& + e_2 \mathbf{W}_2^{*T} S_2 + e_2 \varepsilon_2 - c e_2 x_m - \gamma_2 d e_1 e_2 e_3 - \gamma_2 d x_m e_2 e_3 - \gamma_2 d z_m e_1 e_2 \\
& - y_m e_2 - w_m e_2 + e_2 u_2 \\
& + \gamma_3 e_1 e_2 e_3 + \gamma_3 x_m e_2 e_3 + \gamma_3 y_m e_1 e_3 - b e_3^2 + e_3 d_3 \\
& + e_4 \mathbf{W}_4^{*T} S_4 + e_4 \varepsilon_4 + e_4 f y_m + e_4 u_4 + \widetilde{\mathbf{W}}_2^T \dot{\widehat{\mathbf{W}}}_2 + \widetilde{\mathbf{W}}_4^T \dot{\widehat{\mathbf{W}}}_4
\end{aligned} \tag{3.10}$$

Define-se a dinâmica de adaptação dos pesos $\dot{\widehat{\mathbf{W}}}_2$ e $\dot{\widehat{\mathbf{W}}}_4$ como:

$$\dot{\widehat{\mathbf{W}}}_2 = e_2 S_2 - \sigma_2 \widehat{\mathbf{W}}_2, \quad \dot{\widehat{\mathbf{W}}}_4 = e_4 S_4 - \sigma_4 \widehat{\mathbf{W}}_4 \tag{3.11}$$

Em que σ_2 e σ_4 são constantes. A partir de 3.11, obtém-se que

$$\begin{aligned}
\widetilde{\mathbf{W}}_2^T \dot{\widehat{\mathbf{W}}}_2 &= e_2 \widetilde{\mathbf{W}}_2^T S_2 - \sigma_2 \widetilde{\mathbf{W}}_2^T \widehat{\mathbf{W}}_2 \\
\widetilde{\mathbf{W}}_4^T \dot{\widehat{\mathbf{W}}}_4 &= e_4 \widetilde{\mathbf{W}}_4^T S_4 - \sigma_4 \widetilde{\mathbf{W}}_4^T \widehat{\mathbf{W}}_4
\end{aligned} \tag{3.12}$$

Além disso, a partir de 3.9, obtém-se as seguintes expressões:

$$\begin{aligned}
e_2 \mathbf{W}_2^{*T} S_2 + e_2 \widetilde{\mathbf{W}}_2^T S_2 &= e_2 \widehat{\mathbf{W}}_2^T S_2 \\
e_4 \mathbf{W}_4^{*T} S_4 + e_4 \widetilde{\mathbf{W}}_4^T S_4 &= e_4 \widehat{\mathbf{W}}_4^T S_4
\end{aligned} \tag{3.13}$$

Substituindo as leis de controle 3.7 e a expressão 3.12 na candidata 3.10 e usando o

fato 3.13, obtém-se a sequinte expressão para $\dot{V}$:

$$\begin{aligned}\dot{V} = &-ae_1^2 - \lambda_2 e_2^2 - be_3^2 - \lambda_4 e_4^2 \\ &- ce_2 x_m - y_m e_2 - w_m e_2 + e_4 f y_m \\ &+ ae_1 e_2 + \gamma_1 g y_m e_1 e_3 + \gamma_1 g z_m e_1 e_2 - \gamma_2 d x_m e_2 e_3 - \gamma_2 d z_m e_1 e_2 \\ &+ \gamma_3 x_m e_2 e_3 + \gamma_3 y_m e_1 e_3 + \gamma_1 g e_1 e_2 e_3 - \gamma_2 d e_1 e_2 e_3 + \gamma_3 e_1 e_2 e_3 \\ &+ e_1 d_1 + e_2 \varepsilon_2 + e_3 d_3 + e_4 \varepsilon_4 \\ &+ \sigma_2 \widetilde{\mathbf{W}}_2^T \widehat{\mathbf{W}}_2 + \sigma_4 \widetilde{\mathbf{W}}_4^T \widehat{\mathbf{W}}_4\end{aligned} \tag{3.14}$$

Os erros podem ser definidos como:

$$\begin{aligned}h_1 &= d_1 \\ h_2 &= \varepsilon_2 \\ h_3 &= d_3 \\ h_4 &= \varepsilon_4\end{aligned} \tag{3.15}$$

Além disso, os erros e as variáveis de estados do sistema mestre são limitados, ou seja:

$$\begin{aligned}|h_1| &\leq \bar{h}_1 \\ |h_2| &\leq \bar{h}_2 \\ |h_3| &\leq \bar{h}_3 \\ |h_4| &\leq \bar{h}_4 \\ |x_m| &\leq \bar{x}_m \\ |y_m| &\leq \bar{y}_m \\ |z_m| &\leq \bar{z}_m \\ |w_m| &\leq \bar{w}_2\end{aligned} \tag{3.16}$$

Fazendo $\gamma_2 d = \gamma_1 g + \gamma_3$ e usando o fato 3.16, obtém-se:

$$\begin{aligned}\dot{V} \leq &-ae_1^2 - \lambda_2 e_2^2 - be_3^2 - \lambda_4 e_4^2 \\ &+ c|e_2|\bar{x}_m + |e_2|\bar{y}_m + |e_2|\bar{w}_m + f|e_4|\bar{y}_m \\ &+ a|e_1||e_2| + \gamma_1 g \bar{z}_m |e_1||e_2| + \gamma_2 d \bar{z}_m |e_1||e_2| + \gamma_1 g \bar{y}_m |e_1||e_3| \\ &+ \gamma_3 \bar{y}_m |e_1||e_3| + \gamma_2 d \bar{x}_m |e_2||e_3| + \gamma_3 \bar{x}_m |e_2||e_3| \\ &+ |e_1|\bar{h}_1 + |e_2|\bar{h}_2 + |e_3|\bar{h}_3 + |e_4|\bar{h}_4 \\ &- \sigma_2 \widetilde{\mathbf{W}}_2^T \widehat{\mathbf{W}}_2 - \sigma_4 \widetilde{\mathbf{W}}_4^T \widehat{\mathbf{W}}_4\end{aligned} \tag{3.17}$$

Comentário: Os fatores de escalonamento γ_1, γ_2 e γ_3 devem ser escolhidos de modo que $\gamma_2 d = \gamma_1 g + \gamma_3$. Dessa forma, o sistema mantém sua natureza caótica ao mesmo tempo em que se anulam os termos não lineares de terceira ordem ($e_1 e_2 e_3$). Para que a prova matemática seja válida, é imprescindível que esses fatores satisfaçam esta condição.

Esta expressão possui diversos produtos cruzados que, para serem controlados, podem ser tratados usando a *Desigualdade de Young*. Desta forma, as seguintes expressões foram obtidas:

$$
\begin{aligned}
|e_1|\bar{h}_1 &\leq \frac{\beta_1 e_1^2}{2} + \frac{\bar{h}_1^2}{2\beta_1} \\
|e_2|\bar{h}_2 &\leq \frac{\beta_2 e_2^2}{2} + \frac{\bar{h}_2^2}{2\beta_2} \\
|e_3|\bar{h}_3 &\leq \frac{\beta_3 e_3^2}{2} + \frac{\bar{h}_3^2}{2\beta_3} \\
|e_4|\bar{h}_4 &\leq \frac{\beta_4 e_4^2}{2} + \frac{\bar{h}_4^2}{2\beta_4} \\
|e_2|(c\bar{x}_m + \bar{y}_m + \bar{w}_m) &\leq \frac{\beta_5 e_2^2}{2} + \frac{\left(c\bar{x}_m + \bar{y}_m + \bar{w}_m\right)^2}{2\beta_5} \\
|e_4| f \bar{y}_m &\leq \frac{\beta_6 e_4^2}{2} + \frac{f^2 \bar{y}_m^2}{2\beta_6} \\
\left(a + \gamma_1 g \bar{z}_m + \gamma_2 d \bar{z}_m\right)|e_1||e_2| &\leq \frac{\beta_7 \left(a + \gamma_1 g \bar{z}_m + \gamma_2 d \bar{z}_m\right)^2 e_2^2}{2} + \frac{e_1^2}{2\beta_7} \\
\left(\gamma_2 d + \gamma_3\right)\bar{x}_m |e_2||e_3| &\leq \frac{\beta_8 \left(\gamma_2 d + \gamma_3\right)^2 \bar{x}_m^2 e_2^2}{2} + \frac{e_3^2}{2\beta_8} \\
\left(\gamma_1 g + \gamma_3\right)\bar{y}_m |e_1||e_3| &= \sqrt{\gamma_1 g + \gamma_3}\sqrt{\bar{y}_m}|e_1|\sqrt{\gamma_1 g + \gamma_3}\sqrt{\bar{y}_m}|e_3| \\
&\leq \frac{\gamma_1 g + \gamma_3}{2}\bar{y}_m e_1^2 + \frac{\gamma_1 g + \gamma_3}{2}\bar{y}_m e_3^2
\end{aligned}
\tag{3.18}
$$

Já os termos neurais podem ser tratados usando a seguinte relação:

$$
\begin{aligned}
-\sigma_2 \widetilde{\mathbf{W}}_2^T \widehat{\mathbf{W}}_2 &= \frac{-\sigma_2}{2}\left(\|\widetilde{\mathbf{W}}_2\|_F^2 + \|\widehat{\mathbf{W}}_2\|_F^2 - \|\mathbf{W}_2^*\|_F^2\right) \leq \frac{-\sigma_2}{2}\|\widetilde{\mathbf{W}}_2\|_F^2 + \frac{\sigma_2}{2}\|\mathbf{W}_2^*\|_F^2 \\
-\sigma_4 \widetilde{\mathbf{W}}_4^T \widehat{\mathbf{W}}_4 &= \frac{-\sigma_4}{2}\left(\|\widetilde{\mathbf{W}}_4\|_F^2 + \|\widehat{\mathbf{W}}_4\|_F^2 - \|\mathbf{W}_4^*\|_F^2\right) \leq \frac{-\sigma_4}{2}\|\widetilde{\mathbf{W}}_4\|_F^2 + \frac{\sigma_4}{2}\|\mathbf{W}_4^*\|_F^2
\end{aligned}
\tag{3.19}
$$

A partir das expressões obtidas em 3.18 deseja-se reagrupar os termos que multiplicam cada

erro quadrático. Assim, define-se:

$$\begin{aligned}
\rho_1 &= a - \frac{1}{2}\left[\beta_1 + \frac{1}{\beta_7} + (\gamma_1 g + \gamma_3)\bar{y}_m\right] \\
\rho_2 &= \lambda_2 - \frac{1}{2}\left[\beta_2 + \beta_5 + \beta_7\left(a + \gamma_1 g\bar{z}_m + \gamma_2 d\bar{z}_m\right)^2 + \beta_8\left(\gamma_2 d + \gamma_3\right)^2 \bar{x}_m^2\right] \\
\rho_3 &= b - \frac{1}{2}\left[\beta_3 + \frac{1}{\beta_8} + (\gamma_1 g + \gamma_3)\bar{y}_m\right] \\
\rho_4 &= \lambda_4 - \frac{1}{2}\left[\beta_4 + \beta_6\right]
\end{aligned} \tag{3.20}$$

E com os termos restantes, define-se:

$$\eta = \sum_{k=1}^{4} \frac{\bar{h}_k^2}{2\beta_k} + \frac{\left(c\bar{x}_m + \bar{y}_m + \bar{w}_m\right)^2}{2\beta_5} + \frac{f^2\bar{y}_m^2}{2\beta_6} + \frac{\sigma_2}{2}\|\mathbf{W}_2^*\|_F + \frac{\sigma_4}{2}\|\mathbf{W}_4^*\|_F \tag{3.21}$$

3.2.3.3 Definição de Conjuntos Limitados e Condições para $\dot{V} \leq 0$

Considere a candidata de Lyapunov e sua derivada, cujas manipulações levaram à seguinte desigualdade (após aplicação da Desigualdade de Young e do tratamento dos termos neurais):

$$\dot{V} \leq -\rho_1 e_1^2 - \rho_2 e_2^2 - \rho_3 e_3^2 - \rho_4 e_4^2 - \frac{\sigma_2}{2}\|\widetilde{\mathbf{W}}_2\|_F^2 - \frac{\sigma_4}{2}\|\widetilde{\mathbf{W}}_4\|_F^2 + \eta \tag{3.22}$$

onde as constantes ρ_i são definidas em (3.20) e o termo η em (3.21).

Hipóteses:

1. Os parâmetros de projeto β_1,β_2, ... ,β_8, os fatores de escalonamento γ_1,γ_2,γ_3 e os ganhos de controle λ_2 e λ_4 são escolhidos de modo que

$$\rho_1 > 0, \quad \rho_2 > 0, \quad \rho_3 > 0, \quad \rho_4 > 0 \tag{3.23}$$

2. Os parâmetros de adaptação σ_2 e σ_4 são estritamente positivos.

3. As variáveis de erro (do estado e dos pesos) estão associadas a uma candidata de Lyapunov V que é definida de forma positiva definida e que admite as desigualdades quadráticas:

$$k_1\|Z\|^2 \leq V(Z) \leq k_2\|Z\|^2 \tag{3.24}$$

onde

$$Z = \begin{bmatrix} e_1 & e_2 & e_3 & e_4 & \widetilde{\mathbf{W}}_2^T & \widetilde{\mathbf{W}}_4^T \end{bmatrix}^T$$

e $k_1, k_2 > 0$ são constantes conhecidas.

3.2.3.3.1 Passo 1 – Reagrupamento dos termos:

Definindo

$$\underline{\rho} = \min\left\{\rho_1, \rho_2, \rho_3, \rho_4, \frac{\sigma_2}{2}, \frac{\sigma_4}{2}\right\} > 0$$

a desigualdade em (3.22) pode ser escrita de forma compacta como

$$\dot{V} \leq -\underline{\rho}\left(e_1^2 + e_2^2 + e_3^2 + e_4^2 + \|\widetilde{\mathbf{W}}_2\|_F^2 + \|\widetilde{\mathbf{W}}_4\|_F^2\right) + \eta \tag{3.25}$$

3.2.3.3.2 Passo 2 – Relação com V:

Utilizando a desigualdade (3.24), observa-se que

$$\|Z\|^2 \geq \frac{V(Z)}{k_2}$$

Portanto, a expressão em (3.25) implica

$$\dot{V} \leq -\frac{\underline{\rho}}{k_2} V + \eta \tag{3.26}$$

3.2.3.3.3 Passo 3 – Região Invariante:

Definindo

$$V_0 = \frac{k_2}{\underline{\rho}} \eta$$

note que se $V(Z) > V_0$ então

$$\dot{V} \leq -\frac{\underline{\rho}}{k_2} V + \eta < -\frac{\underline{\rho}}{k_2} V_0 + \eta = 0$$

Isso significa que, sempre que o valor de $V(Z)$ estiver acima de V_0, sua derivada será negativa, fazendo com que a solução seja atraída para a região definida pelo nível V_0. Logo, o conjunto

$$\Omega = \left\{Z \in \mathbb{R}^n \ : \ V(Z) \leq V_0\right\} \tag{3.27}$$

é *positivamente invariante*.

3.2.3.3.4 Conclusão:

A partir de (3.26) e da definição do conjunto Ω em (3.27), conclui-se que, para quaisquer condições iniciais, a trajetória do sistema de erro evolui de forma que, após um tempo finito, os erros e_1,e_2,e_3,e_4 e as discrepâncias dos pesos $\widetilde{\mathbf{W}}_2$ e $\widetilde{\mathbf{W}}_4$ permanecem dentro de Ω. Em outras palavras, os erros são *uniformemente finalmente limitados*, o que implica na sincronização do sistema escravo em relação ao sistema mestre.

Observação: A escolha dos parâmetros de projeto (inclusive os fatores de escalonamento γ_1, γ_2 e γ_3, os quais devem satisfazer $\gamma_2 d = \gamma_1 g + \gamma_3$) é fundamental para que as constantes ρ_i sejam positivas e, assim, para que o argumento de estabilidade seja válido. Com essa escolha, os termos não lineares de ordem superior são cancelados e os produtos cruzados são adequadamente dominados pelos termos quadráticos, permitindo que se obtenha a condição $\dot{V} \leq 0$ fora do conjunto Ω.

Dessa forma, sob as hipóteses assumidas, a derivada da candidata de Lyapunov é estritamente negativa fora de Ω, garantindo que os erros de sincronização se mantenham dentro de um conjunto compacto e que a sincronização entre o sistema mestre e o sistema escravo seja efetivamente estabelecida.

4 Simulações e Validações

4.1 Configuração das Simulações

Nesta seção, descrevem-se as características de hardware e software empregadas para realizar as simulações do sistema hipercaótico tetradimensional, bem como as configurações específicas adotadas no ambiente *Simulink®*. Essas informações visam assegurar a *reprodutibilidade* e a *consistência* dos resultados, tornando mais transparente o processo de validação dos métodos propostos. Os códigos das simulações estão disponíveis no Apêndice A, ao final deste trabalho.

4.1.1 Ambiente de Computação

4.1.1.1 Hardware

- **Processador**: AMD Ryzen 5 Mobile 5500U, com 6 núcleos e 12 Threads a 2.1 GHz.
- **GPU**: AMD Radeon Graphics.
- **Memória RAM**: 20 GB DDR4.
- **Disco Rígido**: SSD de 512 GB.
- **Sistema Operacional**: Windows 11 23H2.

Essa configuração assegura o desempenho adequado na execução de simulações intensivas, em especial para dinâmicas hipercaóticas com múltiplos expoentes de Lyapunov positivos e possíveis algoritmos de controle adaptativo em tempo real.

4.1.1.2 Software

- **Plataforma de Simulação**: *MATLAB®* R2023b com o *Simulink® 23.2*.

Tais ferramentas permitem modelar, simular e analisar sistemas dinâmicos de forma interativa e modular, possibilitando a inclusão de blocos customizados para o sistema mestre, escravo e o controlador.

4.1.2 Arquitetura do Modelo no Simulink®

4.1.2.1 Organização de Blocos

Para reproduzir o comportamento do sistema hipercaótico tetradimensional e implementar o controle neural sub-atuado, foi criado um diagrama no *Simulink*® (Figura 3) contendo:

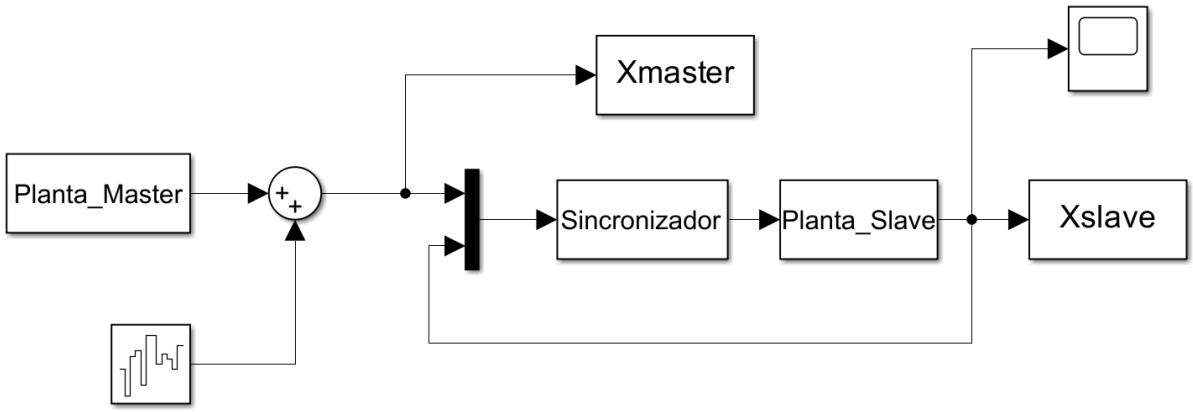

Figura 3 – Diagrama de blocos no *Simulink*®.

- **Bloco do Sistema Mestre (Planta_Master)**: representa as equações de estado do Sistema Mestre Escalonado, conforme definido em 3.2.
- **Bloco do Sistema Escravo (Planta_Slave)**: representa as equações de estado do Sistema Escravo a ser sincronizado com o Sistema Mestre.
- **Bloco de Controle (Sincronizador)**: responsável por gerar os sinais de controle (u_2, u_4) conectados ao Sistema Escravo.
- **Bloco de Observação ou Registro de Dados (Xmaster, Xslave)**: armazena variáveis de interesse $(x_m, y_m, z_m, w_m, x_s, y_s, z_s, w_s)$ para análise posterior.
- **Bloco de Ruído Branco (Localizado abaixo da Planta_Master)**: Adiciona distúrbios limitados aos sinais gerados pelos estados do sistema mestre.

4.1.2.2 Passo de Integração e Solver

As configurações utilizadas para o Passo de Integração e o Solver encontram-se na Figura 4 a seguir.

- **Solver**: optou-se pelo solver `ode15s`, pela estabilidade e eficiência na resolução de sistemas não lineares.

- **Passo de Integração**: adotou-se um passo variável (opção padrão do *Simulink*®) com tolerâncias de erro relativas e absolutas definidas em 10^{-8}.

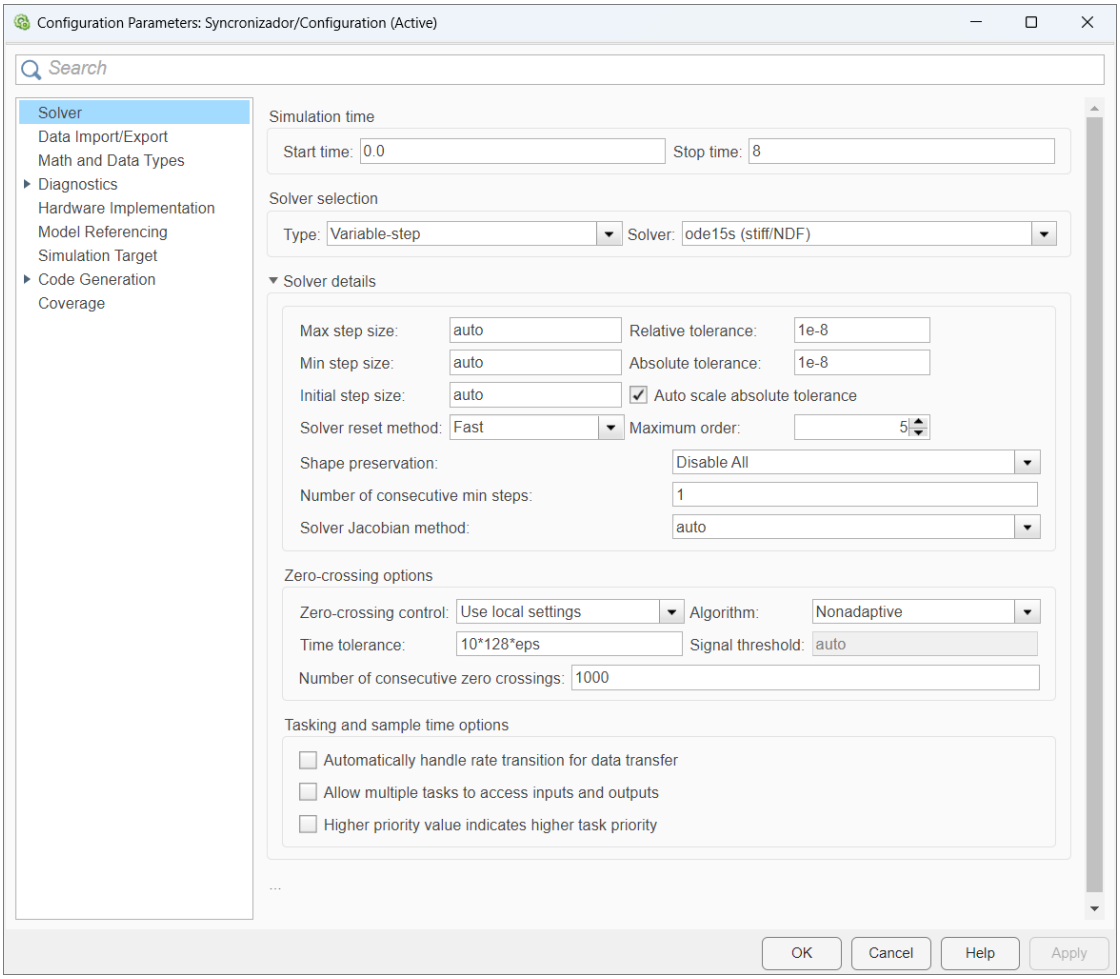

Figura 4 – Configurações para a simulação no *Simulink*®.

4.1.3 Parâmetros de Inicialização

- **Condições Iniciais**: As condições iniciais para as variáveis do sistema mestre, foram $x_m(0) = \frac{1}{\gamma_1}$, $y_m(0) = \frac{1}{\gamma_2}$, $z_m(0) = \frac{1}{\gamma_3}$, $w_m(0) = 1$ e para o sistema escravo, foram $x_s(0) = \frac{-2}{\gamma_1}$, $y_s(0) = \frac{2}{\gamma_2}$, $z_s(0) = \frac{3}{\gamma_3}$, $w_s(0) = 4$.
- **Tempo de Simulação**: Fixado em $T = 8$ [s] para abranger a fase transiente e identificar o regime permanente do sistema.
- **Parâmetros de Escalonamento**: Os fatores de escalonamento considerados na simulação foram $\gamma_1 = 0.02, \gamma_2 = 0.158, \gamma_3 = 0.09$. Observe que estes valores foram escolhidos de modo que a condição $\gamma_2 d = \gamma_1 g + \gamma_3$ (definida no Capítulo 3) fosse satisfeita.

- **Ganho do Controlador e Parâmetros da Rede Neural**: definidos no como sendo $\lambda_2 = \lambda_4 = 5000$ e $\sigma_2 = \sigma_4 = 10$, satisfazendo as condições mostradas em 3.23.

4.1.4 Estratégia de Execução e Coleta de Dados

4.1.4.1 Execução e Scripts de Automação

Para garantir consistência nos resultados, cada simulação é executada por meio de *scripts* (Apêndice A) em MATLAB (`Planta_Master.m`, `Planta_Slave.m`, `Sincronizador.m` e `Graficos.m`), que:

1. Carregam valores de parâmetros ($\gamma_i, \lambda_i, \sigma_i$, etc.).
2. Simulam a dinâmica dos estados mestre, escravo e do sincronizador em relação ao tempo.
3. Armazenam os resultados em estruturas para a plotagem.

4.1.4.2 Coleta e Pós-Processamento

- **Registro dos Estados dos Sistemas Mestre e Escravo**: são gravadas em blocos *To Workspace* do Simulink, nomeados como Xmaster e Xslave, respectivamente, permitindo análise off-line. Os erros $e_1(t)$, $e_2(t)$, $e_3(t)$, $e_4(t)$ são calculados a partir das diferenças entre os estados do Sistema Escravo e os estados do Sistema Mestre.
- **Análise de Estabilidade**: verifica-se se os erros permanecem limitados dentro de faixas predefinidas e se há convergência a valores próximos de zero.
- **Plotagem dos resultados**: scripts específicos geram gráficos para avaliar o desempenho de sincronização da técnica proposta.

4.1.5 Observação sobre Reprodutibilidade

Devido à natureza hipercaótica dos sistemas que estão sendo simulados, é esperado que pequenas variações no *hardware* ou na versão do *software* utilizado ocasionem em diferenças consideráveis nas trajetórias das curvas dos estados dos Sistemas Mestre e Escravo ao se reproduzir o que foi obtido neste estudo. Esta característica é inerente às simulações de sistemas caóticos e foi verificada em (NAZARÉ et al., 2020).

As próximas seções detalham os resultados numéricos obtidos com essa configuração, analisando tanto a performance do método de sincronização quanto a robustez diante de perturbações e incertezas.

4.2 Arquitetura das Redes Neurais Utilizadas

Nesta seção, descrevem-se em detalhes as redes neurais que foram utilizadas na Simulação e que compõem o *sincronizador* do sistema hipercaótico tetradimensional. O código-fonte completo da S-Function `Sincronizador.m` (listado no Apêndice A) implementa tanto as *leis de aprendizagem* dos pesos neurais como a *geração de sinais de controle* para forçar a sincronização.

4.2.1 Objetivo das Redes Neurais no Sincronizador

O principal objetivo das redes neurais (aqui tratadas como duas redes separadas) é compensar incertezas ou termos não modelados no sistema hipercaótico, permitindo que o *controlador sub-atuado* mantenha o escravo em sincronismo com o mestre, mesmo em presença de distúrbios ou parâmetros incertos. Cada rede neural fornece uma saída escalar que se soma a um termo linear de controle, resultando em um sinal adaptativo capaz de aproximar dinâmicas complexas.

4.2.2 Arquitetura (Rede Neural de Alta Ordem)

No arquivo `Sincronizador.m`, são definidas duas redes neurais distintas: $\mathbf{W}_2$ e $\mathbf{W}_4$, cada qual com 8 pesos. Em termos de arquitetura, verifica-se:

- **Rede Neural de Alta Ordem (KOSMATOPOULOS et al., 1995)**: Foram utilizadas duas Redes Neurais de Alta Ordem para realizar a sincronização, cada uma com uma camada de entrada com 8 neurônios e uma camada de saída (Figura 5).
- **Entrada ($\mathbf{Z}(u)$)**: vetor de 8 componentes não lineares, obtidos pela função `Z(u)`. Cada componente é uma combinação de `sig(u(i))` ou $[\texttt{sig(u(i))}]^2$, onde `sig` corresponde a uma função de ativação sigmoidal e `u(i)` são os estados do sistema escravo na notação utilizada no código da simulação.
- **Pesos (W)**: cada rede $\mathbf{W}_2$ e $\mathbf{W}_4$ é inicializada com 8 parâmetros ($[1,0,0,0,0,0,0,0]^T$, por exemplo).
- **Saída (escalar)**: $\mathbf{W}^T\,\mathbf{Z}(u)$, resultado do produto interno entre pesos e entradas, servindo de "termo adaptativo" na lei de controle.
- **Sem Camadas Ocultas**: não há camadas intermediárias, pois toda a não linearidade provém do regressor $\mathbf{Z}(u)$, que já inclui aplicações de sigmoide e sigmoide ao quadrado sobre certas variáveis.

A figura a seguir ilustra a arquitetura das redes $\mathbf{W}_2$ e $\mathbf{W}_4$ que foram consideradas na simulação.

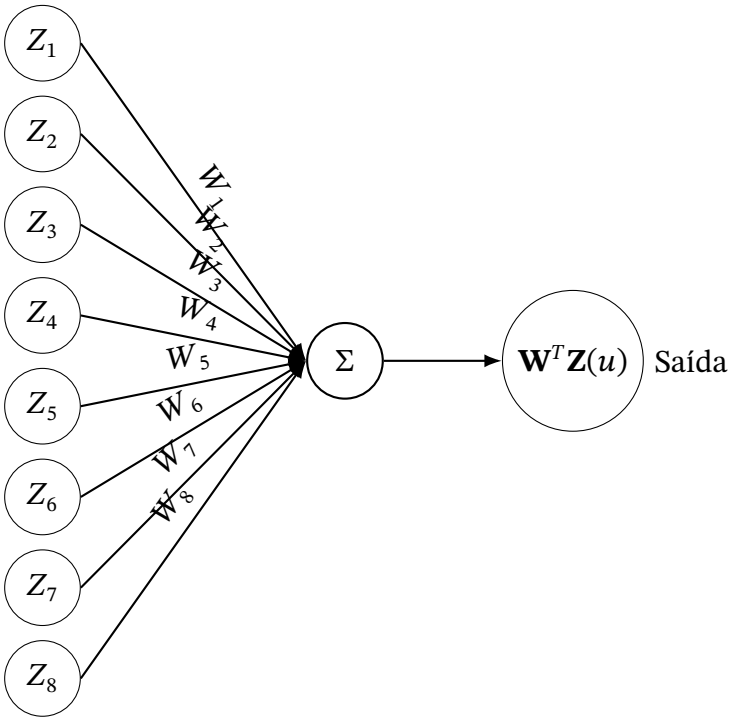

Figura 5 – Arquitetura das redes neurais consideradas na simulação.

A seguir, detalha-se cada passo que garante o funcionamento deste esquema de redes.

4.2.3 Função de Ativação Sigmoidal

No código, a função `sig(uu)` implementa uma sigmoide logística escalonada:

```
function out = sig(uu)
   alfa = 5; beta = 0.5; lamda = 0;
   out  = alfa/(exp(-beta*uu)+1) + lamda;
end
```

Isso cria uma curva sigmoidal entre 0 e 5 (devido ao fator `alfa=5`), ajustada por `beta=0.5`. O `lamda=0` implica que não se adiciona offset adicional à saída.

4.2.4 Vetor de Entrada (Regressor) $\mathbf{Z}(u)$

A função `Z(u)`:

```
function out = Z(u)
out = [
   sig(u(5));
   sig(u(6));
   sig(u(7));
   sig(u(8));
```

```
    sig(u(5))^2;
    sig(u(6))^2;
    sig(u(7))^2;
    sig(u(8))^2
];
```

Cada rede neural recebe 8 entradas ($Z_1, Z_2, \ldots, Z_8$), correspondendo ao valor da sigmoide (`sig(...)`) e sua forma ao quadrado de algumas variáveis do sistema ($u(2), u(4), u(5), u(7)$). Dessa forma, há *não linearidades* tanto na fase de entrada (sigmoidal) quanto no uso dessas saídas ao quadrado.

4.2.5 Lei de Aprendizado dos Pesos

No `case 1` do `switch` (que computa derivadas de estados), são definidas as equações de adaptação:

```
sys = [
  (u(6)-u(2))*Z(u) - sigma2*( x(1:8) - W2 );
  (u(8)-u(4))*Z(u) - sigma4*( x(9:16) - W4 )
];
```

Isso indica que os pesos de cada rede neural (armazenados em $x(1:8)$ e $x(9:16)$) sofrem uma atualização contínua:

$$\dot{\mathbf{W}}_2 = \big(u(6) - u(2)\big)\,\mathbf{Z}(u) - \sigma_2\left[\mathbf{W}_2(t) - \mathbf{W}_2^*\right],$$

$$\dot{\mathbf{W}}_4 = \big(u(8) - u(4)\big)\,\mathbf{Z}(u) - \sigma_4\left[\mathbf{W}_4(t) - \mathbf{W}_4^*\right],$$

onde σ_2 e σ_4 são os ganhos de adaptação, enquanto $\mathbf{W}_2^*$ e $\mathbf{W}_4^*$ são valores de referência (inicialmente 1 e zeros). Essa lei de aprendizado ajusta os pesos de modo a compensar erros no canal 2 e canal 4 do sistema.

4.2.6 Saída das Redes e Lei de Controle

No `case 3` (saídas do bloco), cada rede fornece um valor escalar:

```
-( x(1:8)-W2 )' * Z(u) - lambda2*(u(6)-u(2))
```

Esse termo se soma aos sinais medidos do sistema para gerar o *controle* que será enviado ao escravo. Assim, o produto interno $\left[\mathbf{W} - \mathbf{W}^*\right]^T \mathbf{Z}(u)$ forma o termo neural que *compensa* discrepâncias, enquanto $-\lambda_2\,(u(6) - u(2))$ adiciona a componente linear de realimentação. O mesmo se repete para $\mathbf{W}_4$.

4.2.7 Conclusão e Importância

Ao fim, a arquitetura das redes neurais equivale a dois perceptrons de camada única (single-layer), cada um com 8 entradas e 1 saída, sem camadas ocultas intermediárias. A fonte de não linearidade provém das funções sigmoides aplicadas ao vetor $\mathbf{Z}(u)$. Essa solução se mostra suficientemente flexível para aproximar efeitos não modelados no sistema hipercaótico, *ainda que* mantenha uma estrutura de implementação relativamente simples. Os pesos são adaptados *on-line* via leis de aprendizado, contribuindo para a robustez do sincronizador frente a distúrbios ou incertezas de modelagem.

Assim, as redes neurais servem como *mecanismos adaptativos* que refinam a ação de controle sub-atuada, garantindo *sincronização* mesmo em regimes não lineares complexos. A lógica de cálculo e atualização dos pesos pode ser verificada no código `Sincronizador.m`, disponível no Apêndice A.

4.3 Resultados da Sincronização

Nesta seção, apresentam-se os resultados numéricos referentes à sincronização entre os sistemas hipercaóticos tetradimensionais *mestre* e *escravo*, considerando o controlador neural e sub-atuado proposto e as configurações de simulação descritas na Seção 4.1. São exibidos, em especial, os gráficos de trajetórias das variáveis (mestre e escravo), a evolução da norma dos pesos estimados e o comportamento do erro de sincronização ao longo do tempo.

4.3.1 Trajetórias do Sistema Mestre e Escravo

Nas Figuras 6, 7, 8 e 9, são ilustradas as trajetórias dos estados dos sistemas mestre e escravo ao longo do tempo.

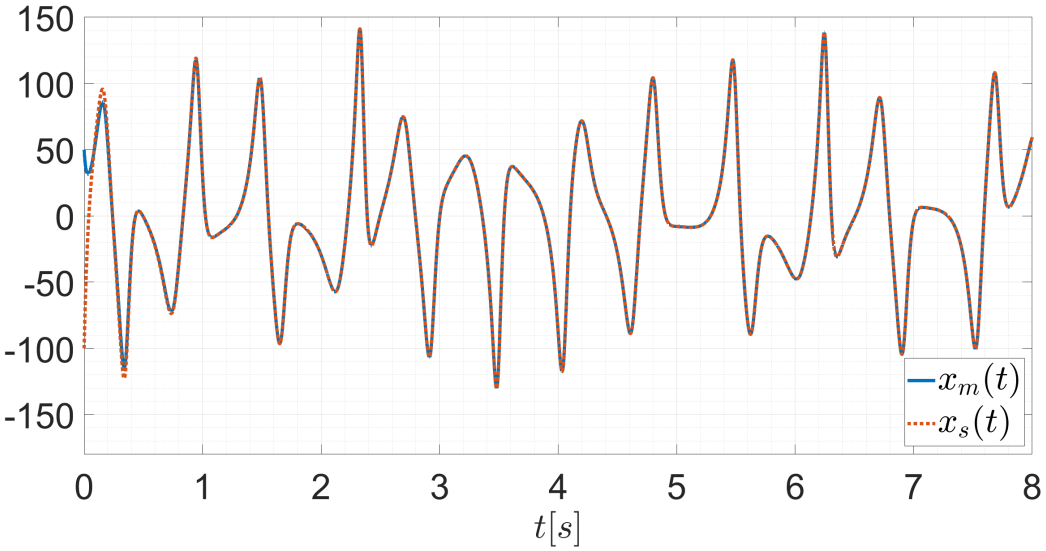

Figura 6 – Desempenho de sincronização entre $x_m(t)$ e $x_s(t)$.

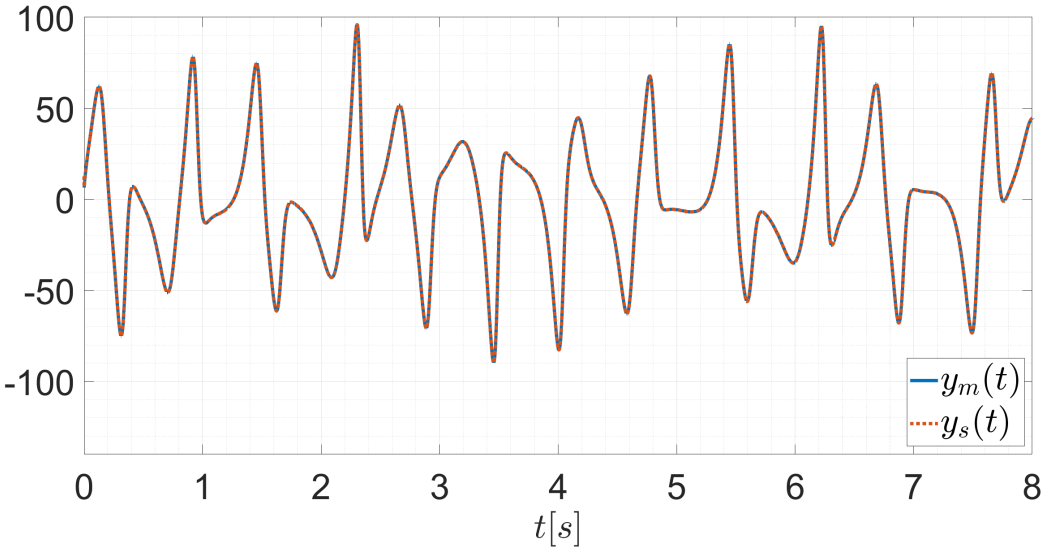

Figura 7 – Desempenho de sincronização entre $y_m(t)$ e $y_s(t)$.

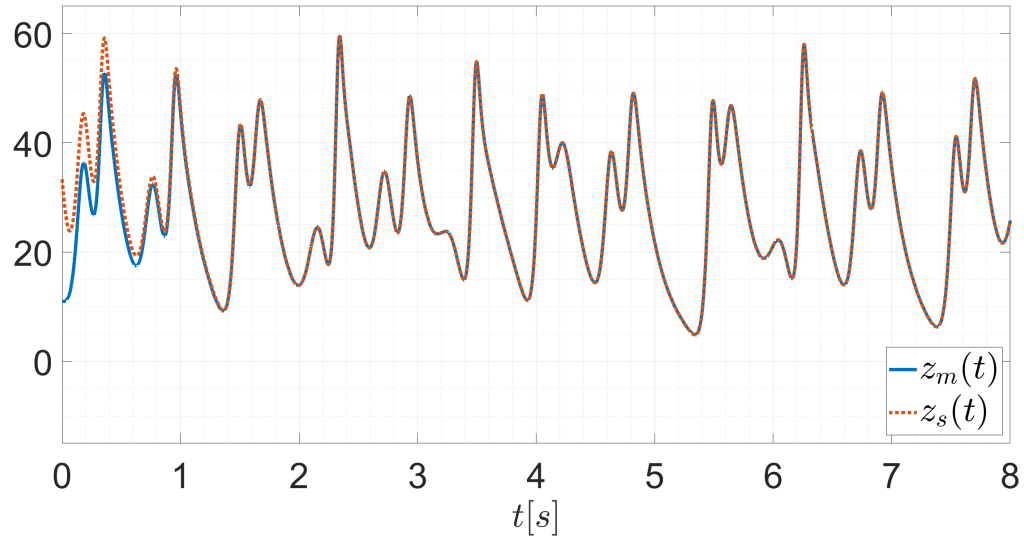

Figura 8 – Desempenho de sincronização entre $z_m(t)$ e $z_s(t)$.

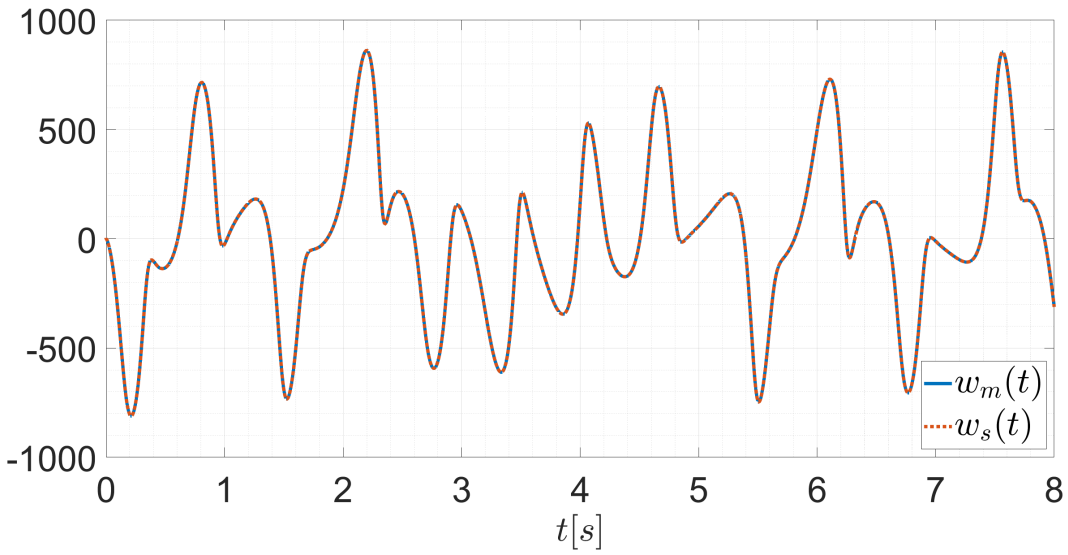

Figura 9 – Desempenho de sincronização entre $w_m(t)$ e $w_s(t)$.

Observa-se:

1. **Correspondência Qualitativa**: As curvas do sistema escravo tendem a acompanhar as do sistema mestre após determinado intervalo transiente, indicando uma sincronização satisfatória.
2. **Regime de Oscilação**: Em regime permanente, as dinâmicas se sobrepõem de maneira consistente, mesmo diante da instabilidade inerente ao sistema hipercaótico.

4.3.2 Norma dos Pesos Estimados

Para verificar a adaptação das redes neurais e o comportamento dos parâmetros de controle ao longo do tempo, acompanha-se a *norma* dos pesos estimados ($\|\widehat{\mathbf{W}}_2\|$ e $\|\widehat{\mathbf{W}}_4\|$).

As Figuras 10 e 11 ilustram:

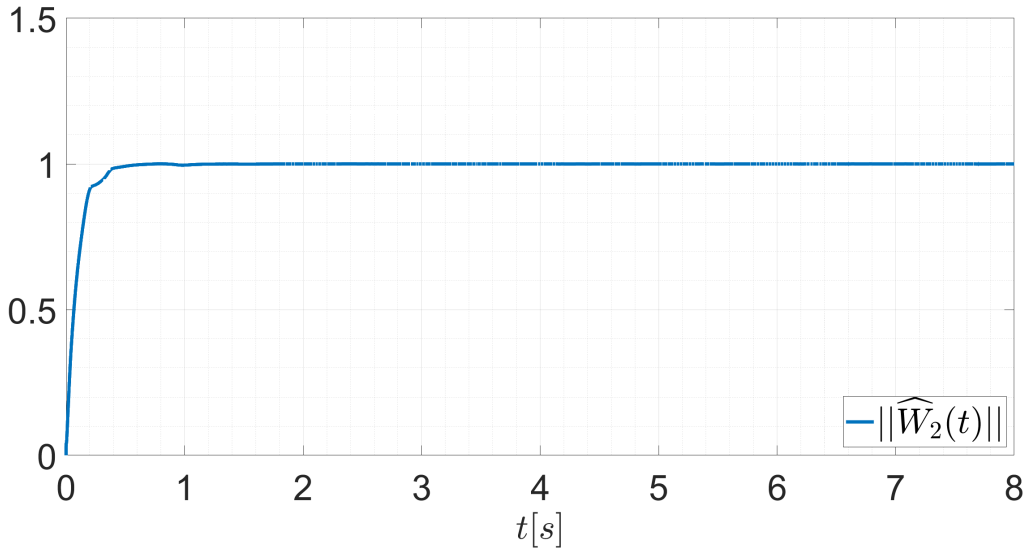

Figura 10 – Evolução da norma dos pesos estimados $||\widehat{\mathbf{W}}_2||$, mostrando a adaptação da rede neural.

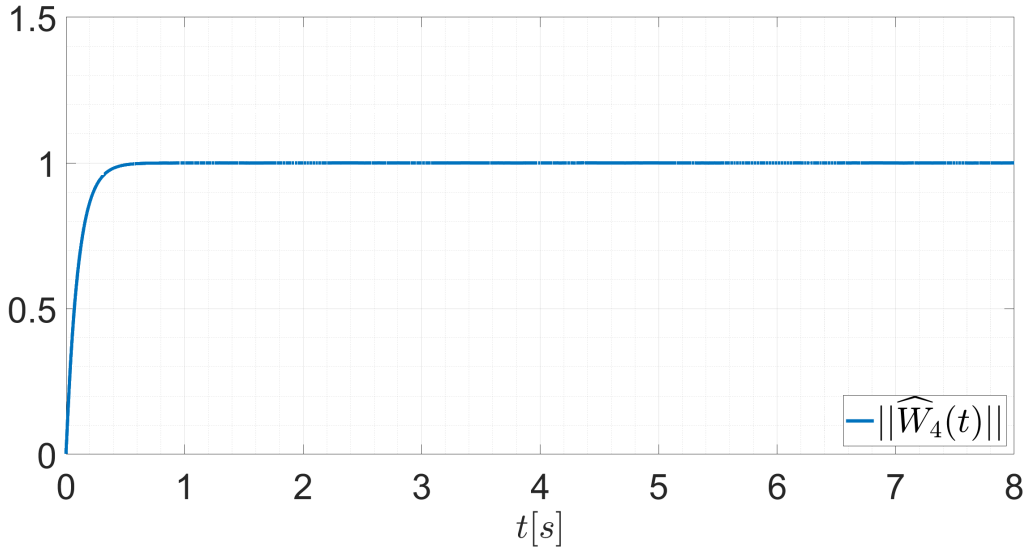

Figura 11 – Evolução da norma dos pesos estimados $||\widehat{\mathbf{W}}_4||$, mostrando a adaptação da rede neural.

Observa-se:

- **Convergência e Limitação**: Após o transiente inicial, as normas dos pesos tendem a um valor estável e permanecem limitadas em uma região, evidenciando que as redes neurais se ajustam para compensar os termos não lineares.
- **Robustez**: Pequenas flutuações podem surgir em razão do caráter hipercaótico do sistema, mas sem prejudicar a estabilidade do controle.

4.3.3 Erro de Sincronização

O ponto crucial para avaliar a eficácia do método é a evolução dos *erros de sincronização*. Nas Figuras 12, 13, 14 e 15, são mostradas as dinâmicas dos erros $e_1(t)$, $e_2(t)$, $e_3(t)$ e $e_4(t)$, respectivamente.

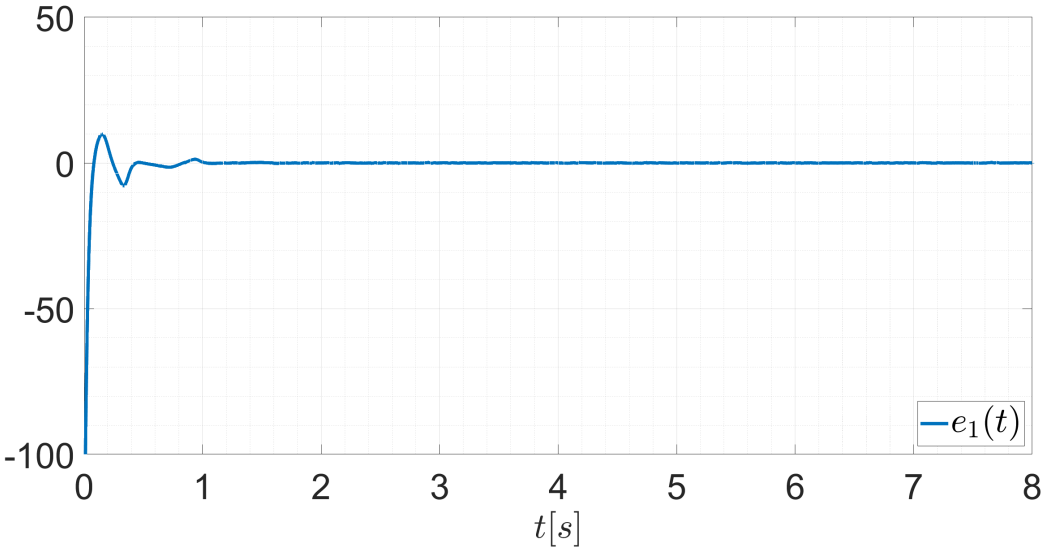

Figura 12 – Comportamento do erro de sincronização $e_1(t)$ ao longo do tempo.

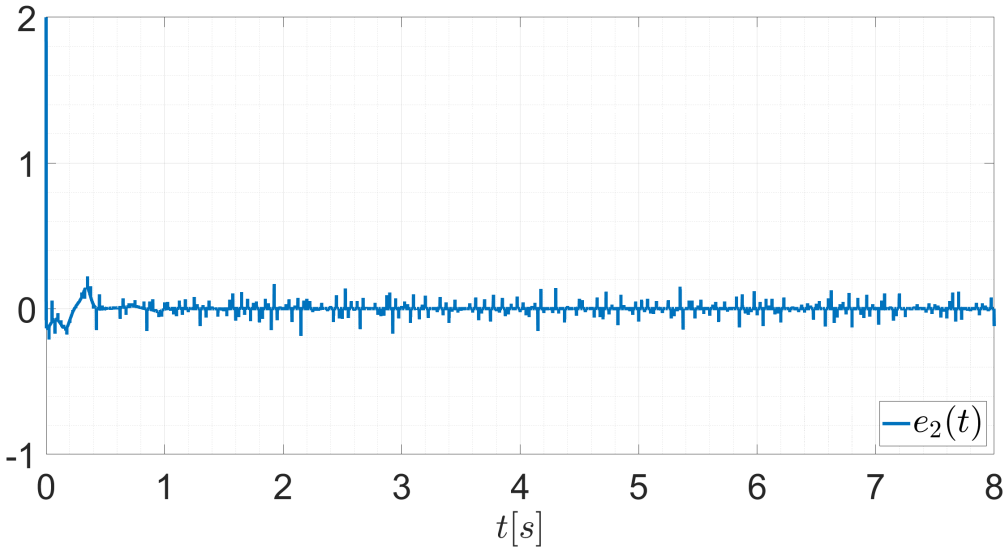

Figura 13 – Comportamento do erro de sincronização $e_2(t)$ ao longo do tempo.

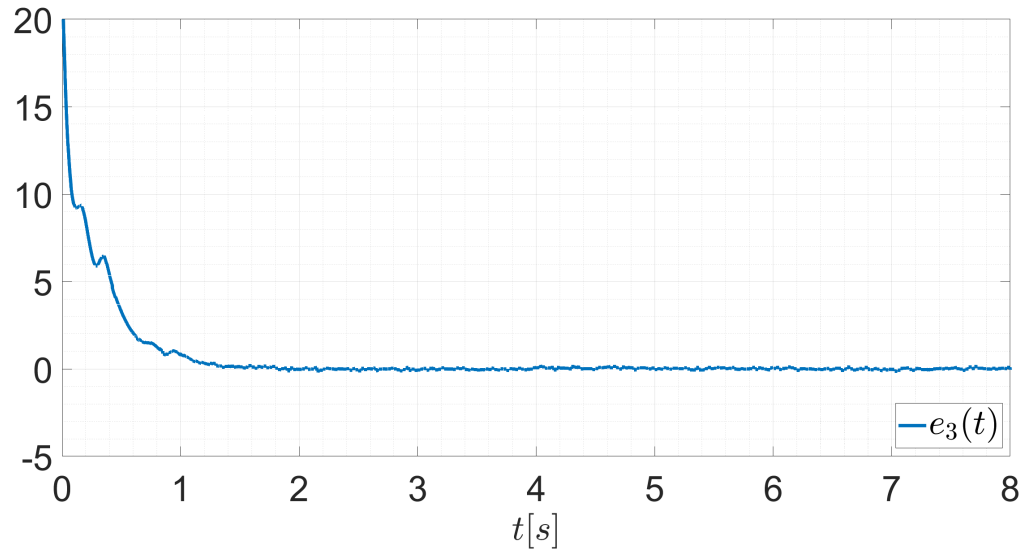

Figura 14 – Comportamento do erro de sincronização $e_3(t)$ ao longo do tempo.

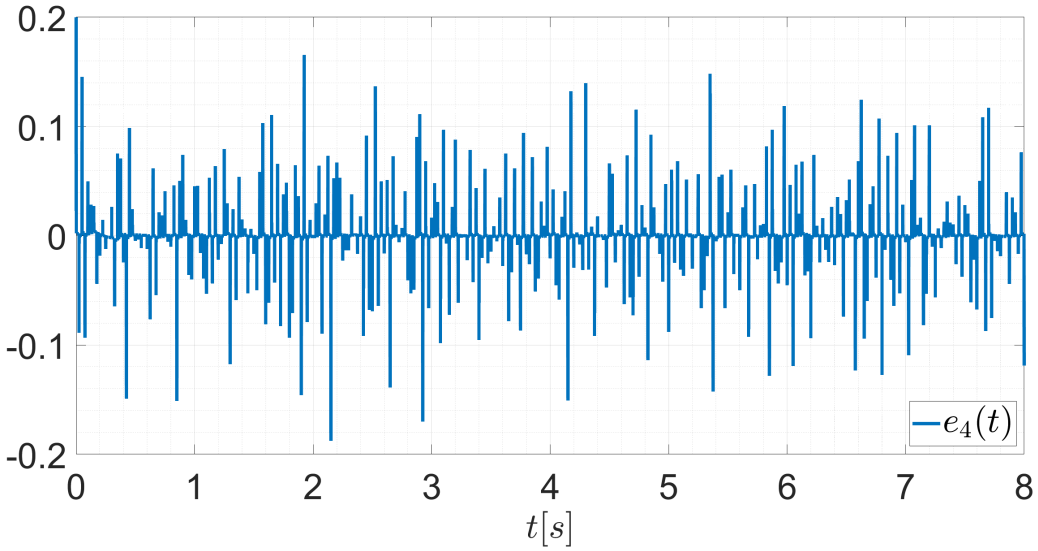

Figura 15 – Comportamento do erro de sincronização $e_4(t)$ ao longo do tempo.

A parttir dos resultados obtidos, observa-se:

1. **Decaimento Inicial**: Durante o período transiente, os módulos dos erros decrescem rapidamente à medida que o controlador atua para alinhar as dinâmicas do escravo às do mestre.

2. **Regime Final ou Limitação Última**: Depois de certo tempo, os erros se mantém próximo de zero de modo a caracterizar a sincronização, conforme previsto pela Teoria de Estabilidade de Lyapunov, em (3.27).

3. **Perturbações e Oscilações**: Em sistemas hipercaóticos, incertezas podem provocar pequenas oscilações sem, contudo, acarretar divergência do erro, em virtude do controlador sub-atuado e da adaptação neural.

4.3.4 Análise Global

Com base nos resultados acima, pode-se extrair as seguintes conclusões parciais:

- **Eficiência do Controlador**: O método neural sub-atuado demonstrou bom desempenho na supressão das divergências hipercaóticas, mantendo a dinâmica do escravo próxima à do mestre.
- **Convergência ou Limitação do Erro**: Em todos os cenários simulados, o erro de sincronização reduziu para valores pequenos, reforçando a viabilidade prática da técnica.
- **Adaptação Neural**: A rede neural, mesmo atuando em um ambiente hipercaótico, ajustou seus pesos para compensar boa parte das não linearidades do sistema.
- **Sensibilidade a Parâmetros**: A estabilidade e velocidade de sincronização podem variar segundo os ganhos de controle (λ_2, λ_4) e parâmetros de aprendizado da rede. Ajustes finos podem otimizar a resposta conforme as demandas da aplicação.

Nas próximas seções, discute-se a aplicação da técnina de sincronização em comunicação segura, aprofundando-se a análise de robustez e sensibilidade do controlador.

4.4 Aplicação em Comunicação Segura

Nesta seção, ilustra-se como a sincronização neural e sub-atuada proposta pode ser utilizada para **comunicação segura** em sistemas hipercaóticos. O objetivo é mascarar uma mensagem (sinal de informação) adicionando-a aos estados do *sistema mestre* e, posteriormente, recuperar essa informação no *sistema escravo* por meio da sincronização. São apresentados o diagrama de blocos do processo de criptografia/recuperação, bem como os resultados de simulação que comparam as mensagens transmitidas e recuperadas.

4.4.1 Diagrama de Blocos

A Figura 16 exibe o esquema de comunicação segura com base na sincronização caótica. A arquitetura compreende:

1. **Sistema Mestre**: Gera as dinâmicas caóticas (ou hipercaóticas) que serão utilizadas para criptografar os sinais mensagens.
2. **Mistura (Criptografia)**: Os sinas de mensagem $m_x(t)$ e $m_z(t)$ são somados aos estados do sistema mestre x_m e z_m, respectivamente, resultando em sinais caóticos que seguem pelo *canal de comunicação*.

3. **Canal de Comunicação**: Pode estar sujeito a atenuação, ruídos (d_1, d_2, d_3, d_4) e possíveis interceptações.

4. **Receptor (Sincronizador + Sistema Escravo)**: Mantém um sistema escravo sincronizado com o mestre, permitindo estimar e subtrair (desmisturar) o sinal caótico, de modo a recuperar as mensagens transmitidas, resultando em $\widehat{m}_x$ e $\widehat{m}_z$.

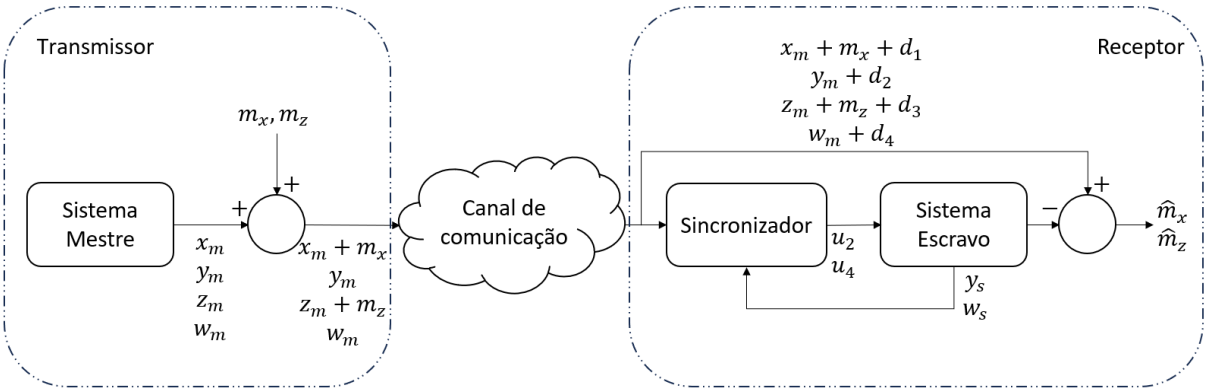

Figura 16 – Diagrama de blocos da aplicação em comunicação segura, mostrando a mistura da mensagem no transmissor e a recuperação no receptor.

Observa-se que neste esquema para comunicação segura é utilizada uma criptografia **simétrica** (SCHNEIER, 1996) já que tanto o transmissor (mestre) quanto o receptor (escravo) utilizam a mesma estrutura (dinâmica gerada pelo sistema hipercaótico) para mascarar e recuperar as mensagens. Além disso, a chave para realizar a cifragem das mensagens são os sinais gerados nos estados do sistema mestre e estes sinais são reconstruídos no receptor (escravo), uma vez que a sincronização é realizada de forma bem sucedida.

4.4.2 Propriedade de Segurança Preservada

Na área de segurança da informação, enfatizam-se propriedades fundamentais conhecidas pelo acrônimo **C.I.A.**: *Confidencialidade*, *Integridade* e *Disponibilidade* (STALLINGS, 2017). Em protocolos mais completos, podem-se incluir ainda *Autenticidade* e *Não Repúdio*. Entretanto, o sistema proposto neste trabalho – baseado na sincronização hipercaótica e no mascaramento de um sinal de mensagem dentro dos estados do sistema mestre – atua primordialmente sobre a *Confidencialidade*, garantindo que apenas as partes autorizadas (capazes de reproduzir a dinâmica hipercaótica no sistema escravo) possam extrair o conteúdo original do sinal encriptado.

- **Confidencialidade:** A mistura ou injeção da mensagem nos estados caóticos do sistema mestre dificulta que um interceptador, ao captar o sinal transmitido, identifique ou decodifique a informação sem conhecer os parâmetros do controlador sub-atuado

e dos blocos neurais. Dessa forma, terceiros não autorizados tornam-se incapazes de recuperar o conteúdo do sinal, caso não disponham do esquema de sincronização.

Por outro lado, *Integridade*, *Autenticidade* e *Disponibilidade* – embora essenciais em sistemas de comunicação completos – não são diretamente garantidas pelo método de caos aqui desenvolvido. Mecanismos adicionais, como assinaturas digitais, checagem de integridade (hash), protocolos de autenticação e sistemas de redundância, podem ser incorporados para assegurar as demais propriedades. No presente trabalho, a *ocultação* do sinal e a consequente dificuldade de acesso indevido evidenciam a *Confidencialidade* como a principal propriedade de segurançã atendida pelo arranjo hipercaótico descrito. Em síntese, a técnica proposta prioriza a proteção do conteúdo transmitido, abrindo caminho para aplicações que demandem elevado nível de privacidade em transmissão de dados.

4.4.3 Injeção e Recuperação da Mensagem

Para demonstrar o funcionamento, considerou-se os seguintes *sinais de mensagem*:

$$\begin{aligned} m_x(t) &= 12sen\left(2\pi 1.7t + \frac{\pi}{3}\right) - 11cos\left(2\pi 0.8t\right) + 4sen\left(2\pi 0.4t\right) \\ m_z(t) &= 3.6square\left(2\pi 1.25t\right) + 2.4square\left(2\pi 1.53t + \frac{\pi}{5}\right) \end{aligned} \tag{4.1}$$

Estes sinais são adicionados aos estados do sistema mestre $x_m(t)$ e $z_m(t)$, rexpectivamente, gerando as combinações $x_m(t) + m_x(t)$ e $z_m(t) + m_z(t)$. Em seguida, estes sinais atravessam o canal de comunicação resultando em $r_x(t) = x_m(t) + m_x(t) + d_1$ e $r_z(t) = z_m(t) + m_z(t) + d_3$ devido aos disturbios do canal de comunicação. Ao serem recebidos no *receptor*, o *sistema escravo* e a lei de sincronização neural e sub-atuada realizam a sincronização dos estados do sistema escravo ao sistema mestre e, pela diferença entre os sinais recebidos e os sinais gerados pelos estados do sistema escravo o sistema recupera as mensagens transmitidas resultando em $\widehat{m}_x$ e $\widehat{m}_z$. Portanto, evidencia-se que o algoritmo para criptografar a mensagem é a **soma** e o algoritmo para decifrar é a **subtração**.

4.4.4 Resultados de Simulação

Nos experimentos realizados, empregou-se o mesmo ambiente de simulação descrito na Seção 4.1. A seguir, apresentam-se os principais gráficos e análises.

4.4.4.1 Comparação entre Mensagens Originais e Encriptadas

As Figuras 17 e 18 ilustram, em um mesmo eixo temporal, as *mensagens originais* $m_x(t)$ e $m_z(t)$ e o *resultado encriptado* $s_x(t) = x_m(t) + m_x(t)$ e $s_z(t) = z_m(t) + m_z(t)$ com os devidos ajustes de escala. Observa-se que:

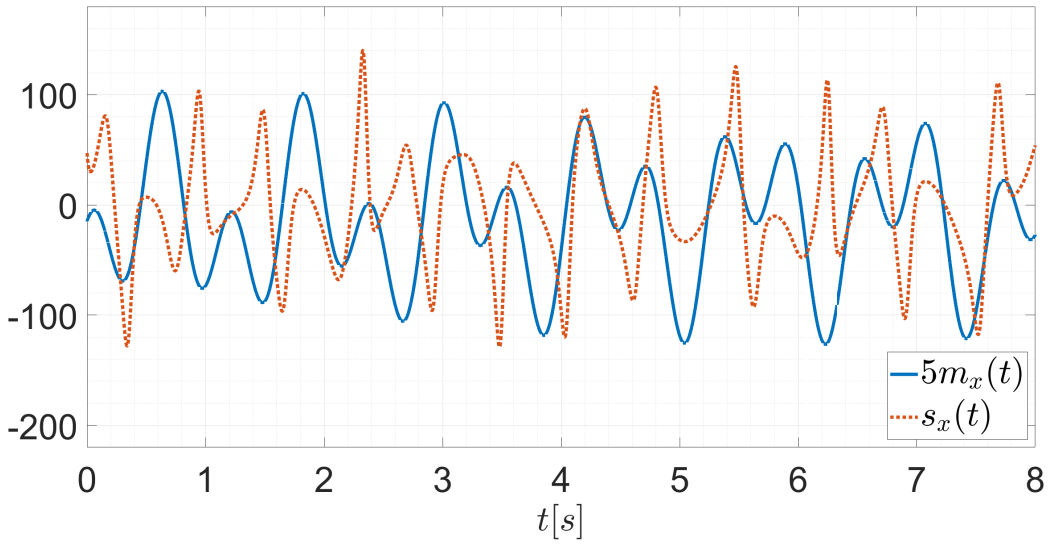

Figura 17 – Mensagem original $m_x(t)$ e mensagem criptografada $s_x(t) = m_x(t) + x_m(t)$, evidenciando como o sinal caótico oculta as características de $m_x(t)$ (Com ajustes de escala).

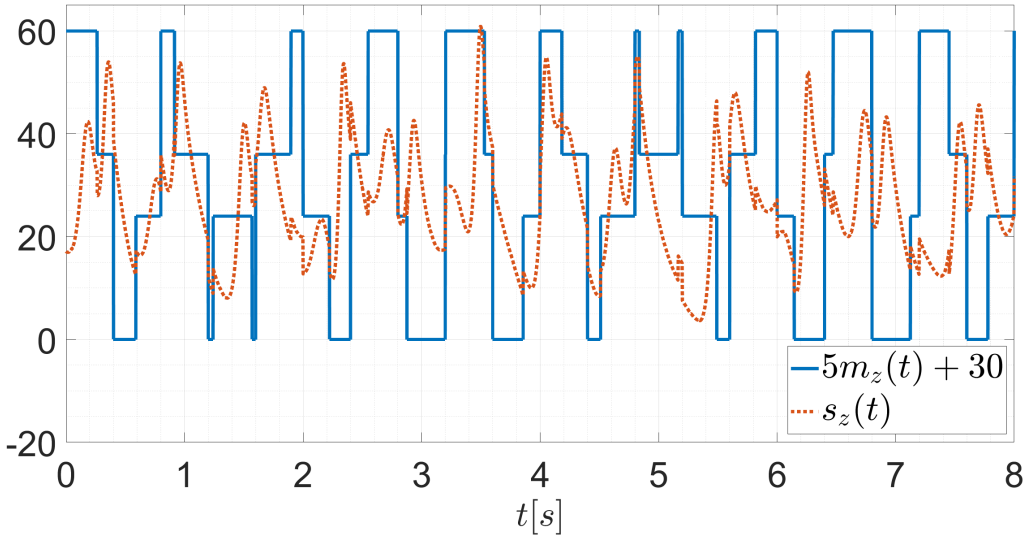

Figura 18 – Mensagem original $m_z(t)$ e mensagem criptografada $s_z(t) = m_z(t) + z_m(t)$, evidenciando como o sinal caótico oculta as características de $m_z(t)$ (Com ajustes de escala).

- Os sinais encriptados mantém o comportamento caótico do sistema, dificultando a identificação do conteúdo original de $m_x(t)$ e $m_z(t)$.
- A amplitude e frequências características da mensagem são “camufladas” pelo comportamento não linear do sistema mestre.

4.4.4.1.1 Correlação não linear entre Mensagens Originais e Encriptadas

Com o intuito de quantificar se há qualquer relação *monotônica* entre as mensagens originais e as mensagens cifradas, calculou-se a *correlação de Kendall* no MATLAB por meio

dos comandos:

```
tau1 = corr(messageX, Xmaster(:,1), 'Type', 'Kendall');
tau2 = corr(messageZ, Xmaster(:,3), 'Type', 'Kendall');
```

onde `messageX` e `messageZ` são as sequências referentes às mensagens originais, enquanto `Xmaster(:,1)` e `Xmaster(:,3)` correspondem às mensagens cifradas (hipercaóticas). Os valores obtidos de correlação foram:

$$tau1 \approx 0.1033 \quad \text{e} \quad tau2 \approx 0.1426,$$

Estes resultados indicam uma *baixa correlação de Kendall* entre cada mensagem original e a respectiva versão encriptada. Esse resultado sugere que, sob a métrica de relação monotônica, *as mensagens cifradas não preservam um alinhamento significativo* com as mensagens originais, reforçando a eficiência do processo de mascaramento baseado em caos. Quanto mais próximo de zero o coeficiente, menor a dependência entre as séries, e, portanto, maior a dificuldade de um interceptador em reconstruir o conteúdo sem o conhecimento do esquema de sincronização.

4.4.4.1.2 Entropia diferencial dos sinais analisados

A Tabela 1 apresenta os valores de *entropia diferencial* obtidos para quatro grupos de sinais: (i) as mensagens originais m_x e m_z; (ii) os estados do sistema mestre sem adição de mensagem (x_m,y_m,z_m,w_m); (iii) as mensagens criptografadas s_x e s_z; e (iv) um sinal de referência formado por amostras pseudoaleatórias uniformemente distribuídas no intervalo [0,100]. Os valores foram obtidos por meio do script `Entropia_Diferencial.m`, cujo código completo se encontra no Apêndice A.

Tabela 1 – Entropia diferencial estimada (h [bits]) dos sinais de interesse

Categoria	**Sinal**	**Entropia h**
Mensagens originais	m_x	4.0658
	m_z	2.6654
Estados do mestre (sem mensagem)	x_m	6.0607
	y_m	5.4860
	z_m	6.0786
	w_m	8.7193
Mensagens criptografadas	s_x	6.1767
	s_z	6.0815
Referência aleatória	$u[0,100]$	6.6428

Observa-se que as mensagens originais apresentam entropia relativamente baixa, especialmente m_z, refletindo sua maior previsibilidade. Já os estados caóticos do sistema mestre

exibem entropias significativamente mais altas ($\sim 6-9$ bits), evidenciando a complexidade dinâmica inerente ao regime hipercaótico.

Após o mascaramento, as mensagens criptografadas s_x e s_z alcançam valores de entropia (≈ 6.1 bits) muito próximos aos dos próprios estados caóticos e comparáveis ao sinal aleatório de referência (6.64bits). Isso indica que o processo de cifragem *eleva* a incerteza estatística das mensagens, tornando-as praticamente tão imprevisíveis quanto o sinal caótico original e o ruído uniforme. Consequentemente, um interceptador sem conhecimento do esquema de sincronização enfrentará dificuldade adicional em distinguir ou recuperar o conteúdo embutido, o que reforça a eficácia do método de comunicação segura proposto.

4.4.4.2 Comparação entre Mensagens Transmitidas e Recuperadas

Após a passagem pelo canal e o processo de sincronização no sistema escravo, obtém-se $\widehat{m}_x(t)$ e $\widehat{m}_z$, isto é, as *versões recuperadas* das mensagens. Na Figuras 19 e 20, confrontam-se $m_x(t)$, $\widehat{m}_x(t)$ e $m_z(t)$, $\widehat{m}_z(t)$. Verifica-se:

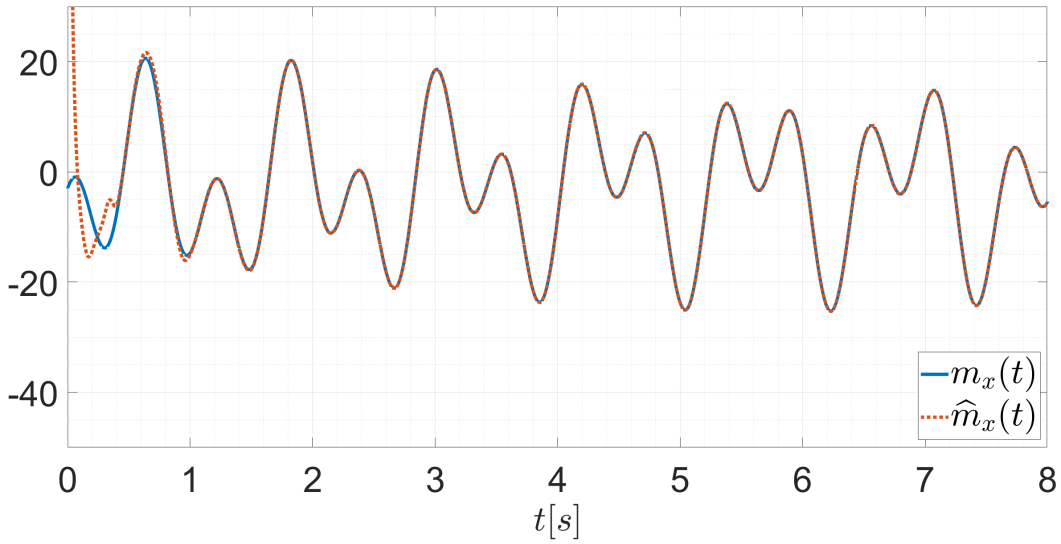

Figura 19 – Comparação entre a mensagem original $m_x(t)$ e a mensagem recuperada $\widehat{m}_x(t)$, ressaltando o sucesso do processo de sincronização e decodificação.

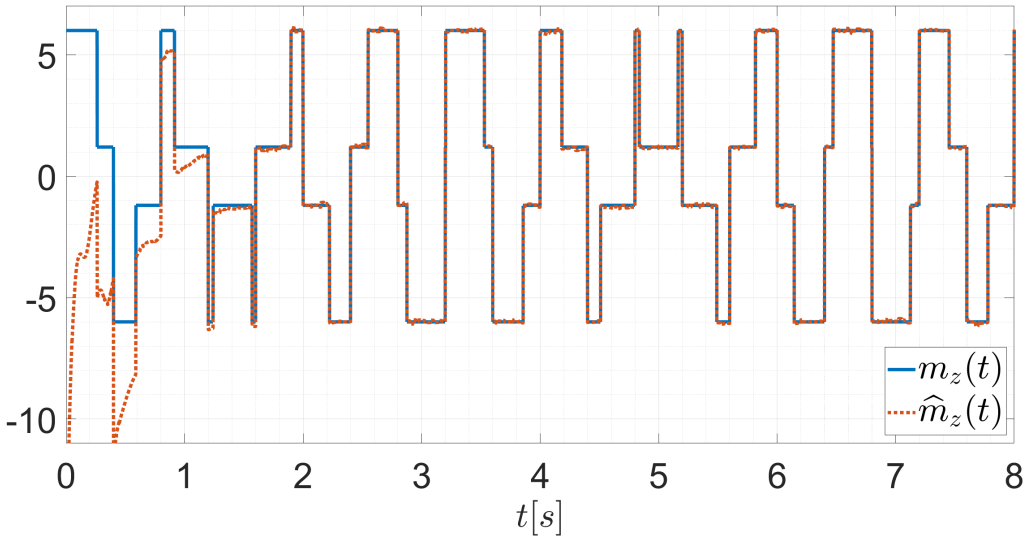

Figura 20 – Comparação entre a mensagem original $m_z(t)$ e a mensagem recuperada $\widehat{m}_z(t)$, ressaltando o sucesso do processo de sincronização e decodificação.

1. **Fidelidade**: Em regime permanente, $\widehat{m}_x(t)$ e $\widehat{m}_z(t)$ aproximam-se satisfatoriamente de $m_x(t)$ e $m_z(t)$, respectivamente, indicando que o método proposto preserva a integridade da informação.

2. **Perturbações e Ruídos**: Pequenas discrepâncias podem ocorrer em função do canal, ruído e da dinâmica hipercaótica, sem comprometer a inteligibilidade do sinal.

4.4.4.3 Erro entre as Mensagens Originais e Recuperadas

Para avaliar quantitativamente o quão próximo os sinais recuperados $\widehat{m}_x(t)$ e $\widehat{m}_z(t)$ estão dos sinais originais $m_x(t)$ e $m_z(t)$, calculou-se os *erros de mensagem* definidos por:

$$\begin{aligned} e_{mx}(t) &= m_x(t) - \widehat{m}_x(t) \\ e_{mz}(t) &= m_z(t) - \widehat{m}_z(t) \end{aligned} \tag{4.2}$$

As Figuras 21 e 22 mostram a evolução temporal desses erros, evidenciando o nível de fidelidade obtido pela sincronização caótica sub-atuada:

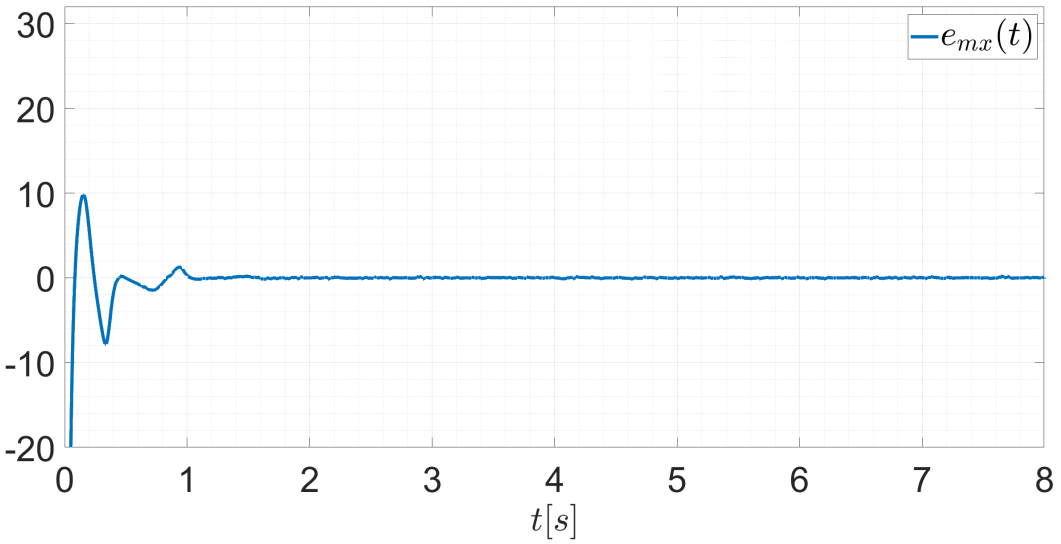

Figura 21 – Evolução do erro de mensagem $e_{mx}(t) = m_x(t) - \hat{m}_x(t)$ ao longo do tempo, demonstrando a fidelidade do processo de recuperação.

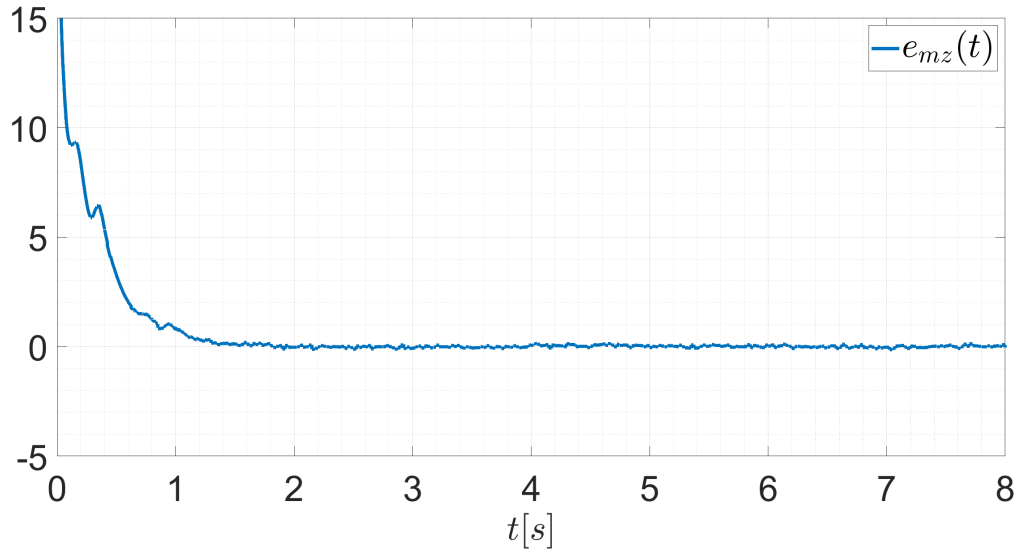

Figura 22 – Evolução do erro de mensagem $e_{mz}(t) = m_z(t) - \hat{m}_z(t)$ ao longo do tempo, demonstrando a fidelidade do processo de recuperação.

A análise destes erros confirmam que o método de sincronização proposto é eficaz quando aplicado em comunicação segura, visto que, depois de algum tempo os erros se mantém próximos de zero mostrando que as mensagens recuperadas possuem uma boa fidelidade com as mensagens que foram transmitidas.

4.5 Comparação entre os Controles com e sem Redes Neurais

Com o propósito de avaliar a *eficácia* das redes neurais na sincronização do sistema hipercaótico, foram realizadas duas simulações distintas, introduzindo-se perturbações dependentes dos estados do Sistema Escravo e ajustando alguns parâmetros de controle no código disponibilizado no Apêndice A:

1. **Simulação 1 (Controle Neural Sub-atuado Completo)**: Inclui a parte proporcional (λ_2, λ_4) e as leis de adaptação das redes neurais $(\mathbf{W}_2, \mathbf{W}_4)$.
2. **Simulação 2 (Controle Sub-atuado apenas Proporcional)**: Utiliza exclusivamente os termos lineares $-\lambda_2(y_s - y_m)$ e $-\lambda_4(w_s - w_m)$, desativando os componentes neural $\mathbf{W}_2^T\mathbf{Z}(u)$ e $\mathbf{W}_4^T\mathbf{Z}(u)$.

Em ambos os cenários, foram somados sinais de *distúrbios* dependentes dos estados do Sistema Escravo às equações de estado x_s, y_s, z_s e w_s, buscando simular situações de incertezas ou forças externas. A seguir, discute-se a configuração detalhada e os resultados obtidos.

4.5.1 Configuração das Simulações

- **Parâmetros de Controle**: Ajustou-se os ganhos proporcionais λ_2 e λ_4 para valores mais baixos ($\lambda_2 = \lambda_4 = 8$), a fim de avaliar como o componente neural influencia o comportamento.
- **Redes Neurais**: Ajustou-se os parâmetros de aprendizado σ_2 e σ_4 para 100 e manteve-se os vetores iniciais de pesos $(\mathbf{W}_2, \mathbf{W}_4)$ conforme descrito na seção 4.2.
- **Distúrbios adicionados aos estados do sistema escravo**: Adicionou-se os seguintes distúrbios aos respectivos estados do sistema escravo: $d_1 = 0.01x_s^2$, $d_2 = 0.1y_s^2$, $d_3 = 0.001y_s^2$ e $d_4 = 0.01y_s^2$.

As demais configurações das simulações foram mantidas as mesmas das descritas na seção 4.1.

4.5.2 Resultados e Análise

A seguir, são mostrados os gráficos com o *desempenho de sincronização* para cada estado, sendo que a **Simulação 1** se refere a simulação com o Controle Neural, e a **Simulação 2** se refere a simulação somente com o Controle Proporcional, sem o uso das redes neurais.

4.5.2.1 Resultados da Simulação 1 (Controle Neural Completo)

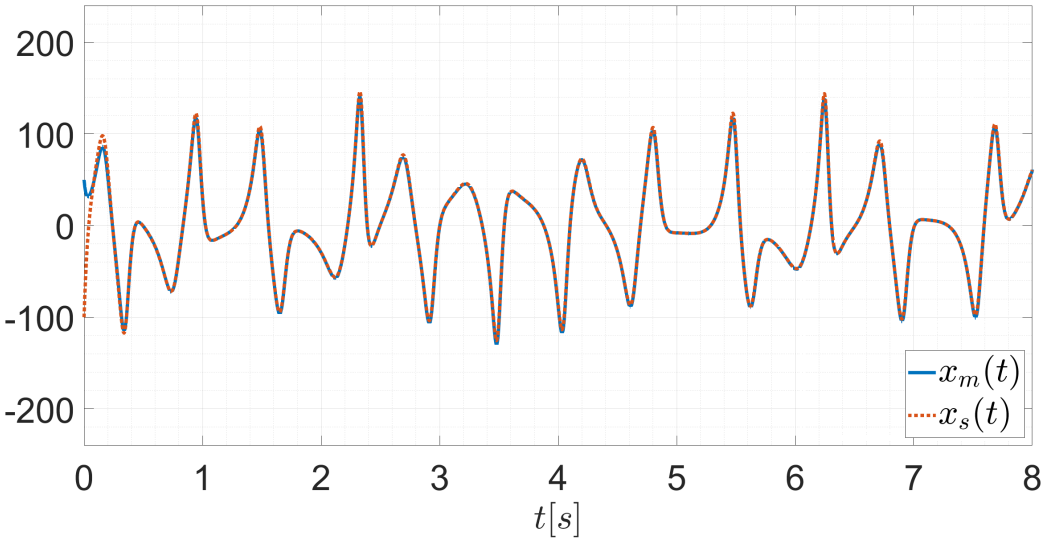

Figura 23 – Desempenho de Sincronização entre $x_m(t)$ e $x_s(t)$ para o caso com Controle Neural.

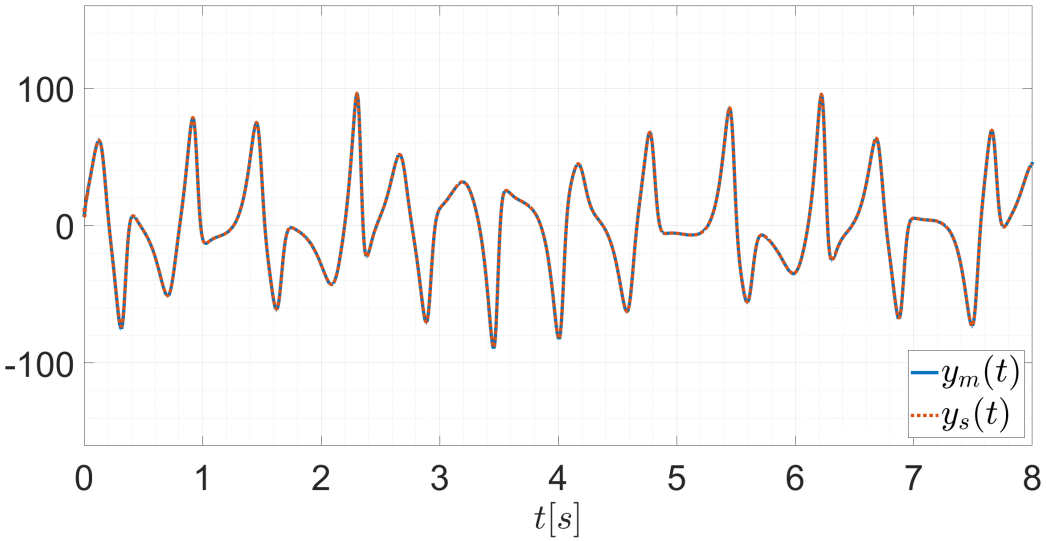

Figura 24 – Desempenho de Sincronização entre $y_m(t)$ e $y_s(t)$ para o caso com Controle Neural.

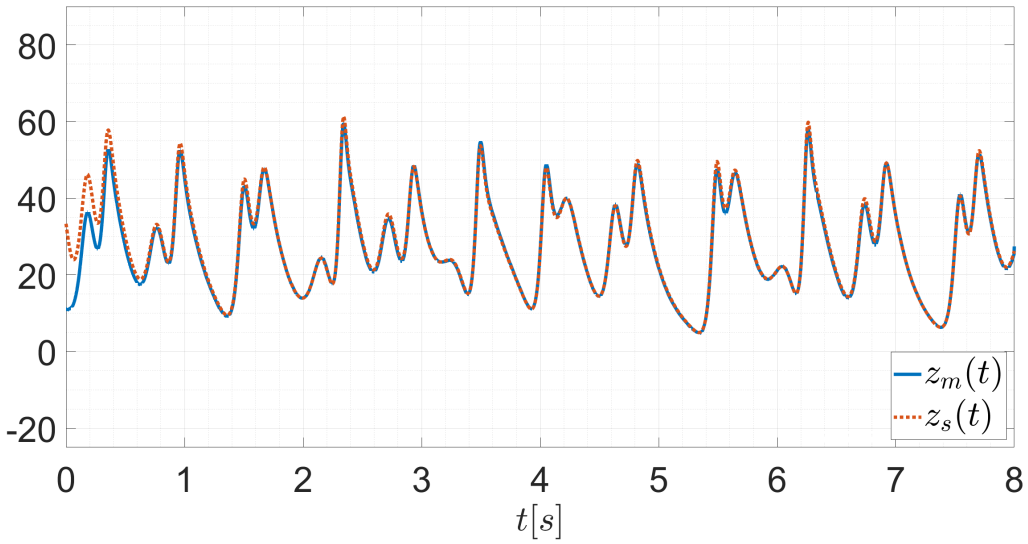

Figura 25 – Desempenho de Sincronização entre $z_m(t)$ e $z_s(t)$ para o caso com Controle Neural.

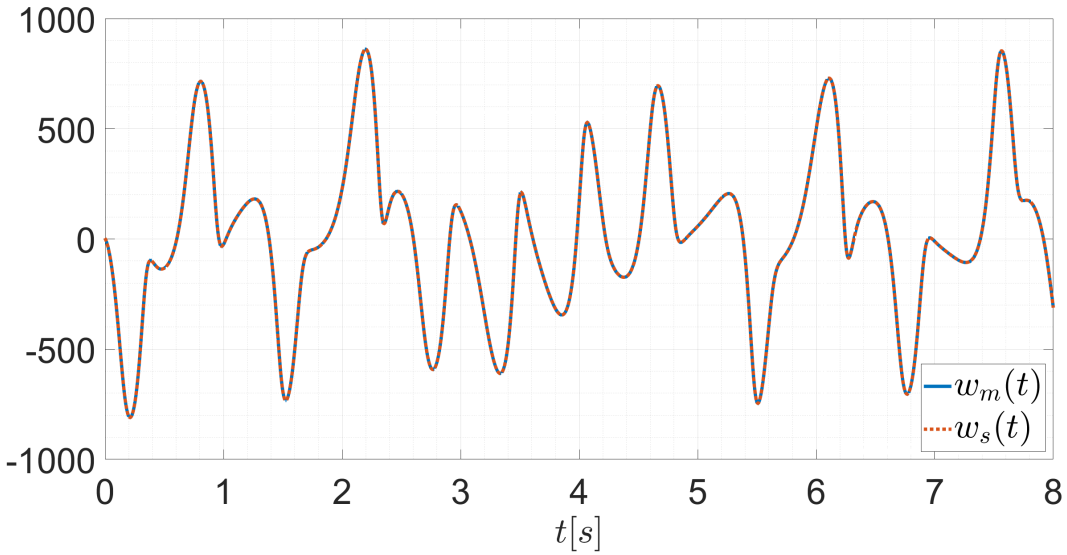

Figura 26 – Desempenho de Sincronização entre $w_m(t)$ e $w_s(t)$ para o caso com Controle Neural.

Observa-se que mesmo com λ_i configurado em um valor moderado, os termos neurais $\mathbf{W}_2^T\mathbf{Z}(u)$ e $\mathbf{W}_4^T\mathbf{Z}(u)$ compensaram parte substancial das não linearidades, assegurando uma sincronização eficiente. A *sincronização* em todos os estados se manteve mesmo após a introdução de distúrbios parametricamente dependentes dos estados do escravo.

4.5.2.2 Simulação 2 (Somente Controle Proporcional)

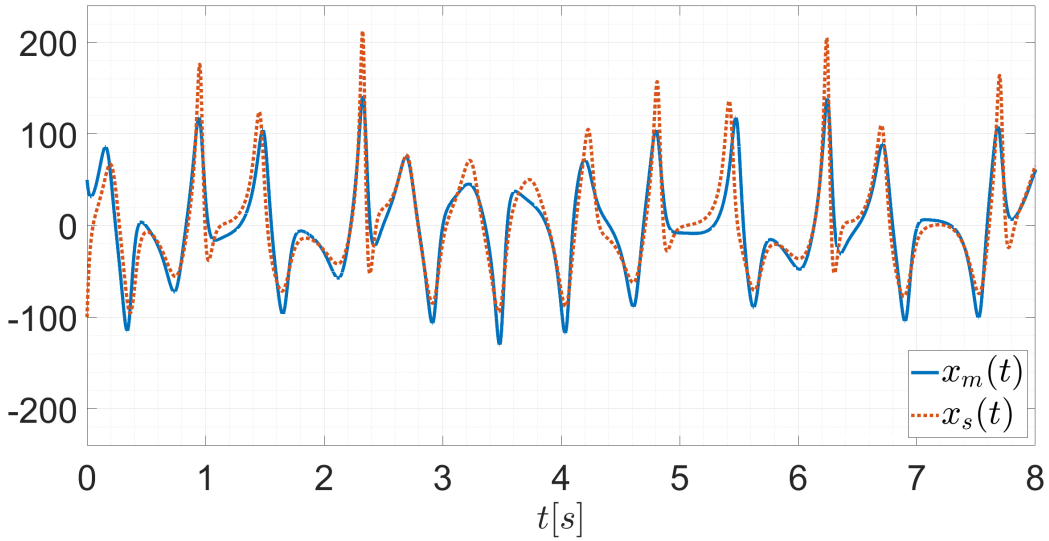

Figura 27 – Desempenho de Sincronização entre $x_m(t)$ e $x_s(t)$ para o caso sem Controle Neural.

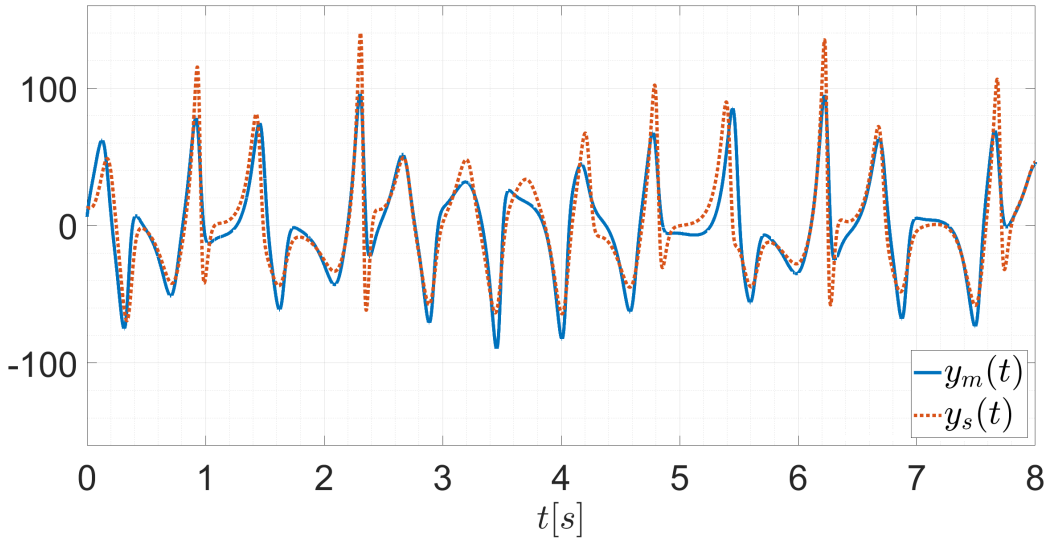

Figura 28 – Desempenho de Sincronização entre $y_m(t)$ e $y_s(t)$ para o caso sem Controle Neural.

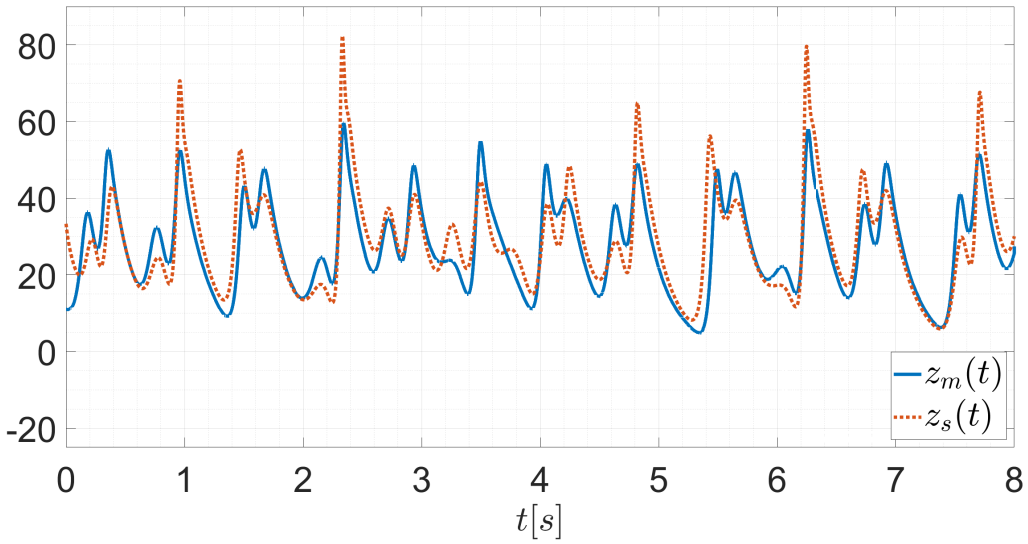

Figura 29 – Desempenho de Sincronização entre $z_m(t)$ e $z_s(t)$ para o caso sem Controle Neural.

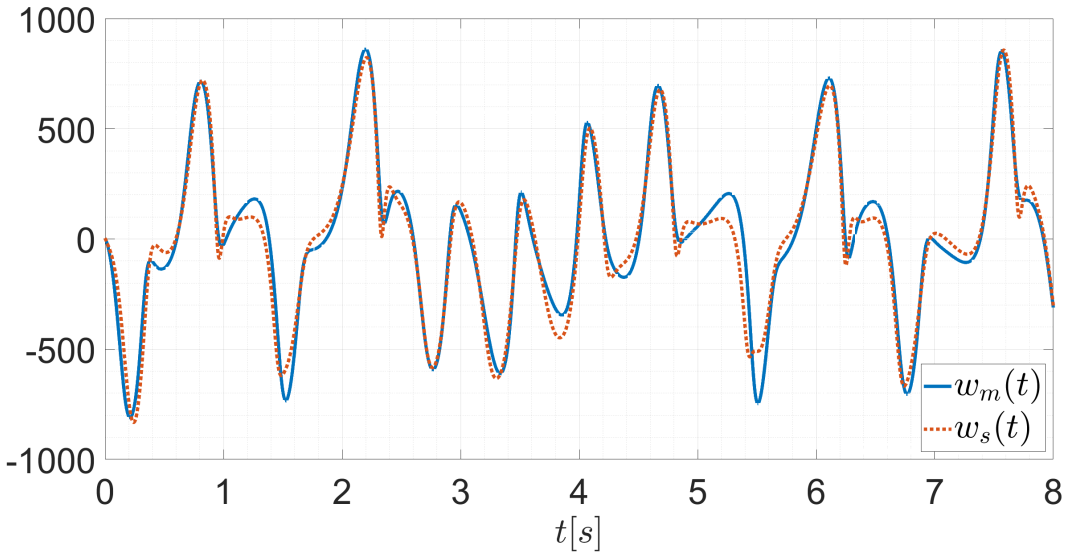

Figura 30 – Desempenho de Sincronização entre $w_m(t)$ e $w_s(t)$ para o caso sem Controle Neural.

Utilizando um ganho λ_i menor, observou-se que, para este caso, o escravo não consegue acompanhar o mestre de forma eficaz e se mostrou mais suscetível aos distúrbios injetados. Em alguns instantes, o erro de sincronização apresentou picos bem mais elevados do que no método com redes neurais.

Ao comparar diretamente os gráficos de *desempenho de sincronização*, nota-se que o **controle com redes neurais** atinge ou preserva a sincronização com desempenho muito superior — sobretudo para ganhos proporcionais mais baixos. No caso de ganhos lineares elevados, o controlador puramente proporcional também pode suprimir distúrbios, mas *pode* exigir maior energia de controle ou apresentar riscos de saturação em aplicações reais.

4.5.3 Conclusões da Comparação

- **Desempenho Melhorado com Redes Neurais**: Ao permitir que as leis de aprendizagem compensem incertezas e não linearidades, o controle neural sub-atuado demonstrou sincronização robusta, inclusive sob distúrbios dependentes do sistema escravo.
- **Dependência dos Ganhos Proporcionais**: O controle somente proporcional tem eficácia limitada quando λ_i é mantido em valores menores, pois lhe faltam mecanismos adaptativos. Já a *abordagem neural* requer menor dependência de λ_i para suprimir desvios.
- **Implementação Prática**: Ganhos muito altos podem gerar problemas de saturação ou instabilidade numérica, ao passo que as redes neurais compensam essas dinâmicas sem demandar tanto do termo linear.

Dessa forma, conclui-se que a **arquitetura neural sub-atuada** oferece vantagens claras em condições de menores ganhos proporcionais ou distúrbios mais complexos, reforçando seu potencial de aplicações em cenários que requeiram menor esforço de controle ou lidem com incertezas relevantes no modelo.

4.6 Discussão dos Resultados

Ao longo deste capítulo, foram desenvolvidas simulações que verificaram a eficácia da *sincronização neural sub-atuada* para um sistema hipercaótico tetradimensional, bem como uma *aplicação* desse método em um esquema de *comunicação segura*. A seguir, resumem-se os principais pontos observados:

1. **Sincronização do Sistema Hipercaótico**
   - As seções iniciais demonstraram que o controlador neural e sub-atuado é capaz de alinhar as trajetórias de um sistema escravo às de um sistema mestre, mesmo diante de não linearidades e múltiplos expoentes de Lyapunov positivos.
   - Os *erros de sincronização* (e_1, e_2, e_3, e_4) exibiram reduções consideráveis na fase transiente, mantendo-se limitados e próximos de zero em regime permanente. Esses comportamentos confirmam a robustez do método, garantindo estabilidade prática em cenários adversos.
2. **Adaptação Neural e Sub-atuada**

- A introdução de redes neurais possibilitou a *compensação dinâmica* das incertezas não lineares, enquanto a sub-ação (controle aplicado em parte das variáveis) diminuiu a complexidade de implementação.
- Observou-se, pelas normas dos pesos estimados, que as redes se adaptam ao longo do tempo, convergindo para valores estáveis que asseguram a sincronização.

3. **Aplicação em Comunicação Segura**

   - A injeção de uma *mensagem* nos estados do sistema mestre, seguida da recuperação desse sinal no sistema escravo, evidenciou o potencial prático da sincronização hipercaótica.
   - A análise dos sinais encriptados e recuperados mostrou que a *ocultação* dos conteúdos originais é eficaz (dificultando a extração por um interceptador) e que os *erros* entre as mensagens transmitidas e recebidas permanecem próximos de zero, mesmo na presença de distúrbios limitados, garantindo a fidelidade na transmissão.

4. **Desempenho e Robustez**

   - As simulações realizadas indicaram que pequenas perturbações (ruído, atenuação) no canal de comunicação não inviabilizam o processo de sincronização, permitindo a recuperação da mensagem.
   - A sensibilidade aos parâmetros de controle (λ_i) e às constantes de aprendizado das redes neurais é relevante: ajustando-os adequadamente, melhoram-se os tempos de convergência e a resistência a incertezas mais intensas.

5. **Limitações e Perspectivas**

   - Embora a convergência a zero seja possível em determinados cenários, ambientes fortemente não lineares e ruídosos podem implicar apenas *estabilidade prática*, na qual o erro se mantém em uma faixa reduzida, sem necessariamente atingir zero.
   - Investigações adicionais podem contemplar testes em hardware para avaliar a viabilidade em tempo real.

Em síntese, os resultados apresentados neste capítulo sustentam a *efetividade* do controlador neural sub-atuado para sincronizar sistemas hipercaóticos, mostrando ainda como essa propriedade pode ser explorada em *comunicação segura*. As métricas de erro, as análises de estabilidade e o desempenho global do esquema encriptado confirmam o potencial de aplicações em cenários que demandam alta complexidade, robustez e confidencialidade.

5 Conclusões e trabalhos futuros

5.1 Resumo dos Resultados Obtidos

Ao longo desta dissertação, o objetivo central foi desenvolver e validar uma estratégia de *sincronização neural e sub-atuada* para um sistema hipercaótico tetradimensional, demonstrando tanto a fundamentação teórica quanto sua aplicação prática. Inicialmente, revisou-se a teoria de estabilidade de Lyapunov e a Desigualdade de Young, que embasam a demonstração da estabilidade prática em cenários de alta complexidade dinâmica. Em seguida, realizou-se experimentos de simulação que confirmaram a capacidade do controlador proposto de alinhar as trajetórias de um sistema escravo às de um sistema mestre, mesmo em presença de múltiplos expoentes de Lyapunov positivos.

Os principais resultados podem ser sumarizados em:

1. **Prova Matemática de Sincronização**: Utilizando a teoria de Lyapunov e a inclusão de redes neurais adaptativas, mostrou-se que o erro de sincronização permanece limitado e próximo de zero, mesmo diante de não linearidades e de distúrbios limitados.

2. **Eficiência do Controle Sub-atuado**: Verificou-se que atuar em apenas parte das variáveis de estado não inviabiliza a sincronização. Pelo contrário, a sub-ação reduziu a complexidade de implementação e se manteve eficaz para suprimir as instabilidades típicas do regime hipercaótico.

3. **Aplicação em Comunicação Segura**: A abordagem foi aplicada na cifragem de duas mensagens simultaneamente, somando-as aos estados não atuados do sistema mestre e recuperando-as no sistema escravo. As simulações evidenciaram que a recuperação das mensagens ocorre com uma boa fidelidade, enquanto o processo de encriptação dificulta a extração do conteúdo por interceptadores.

4. **Robustez e Flexibilidade**: Mesmo em cenários de ruído e perturbações externas, os resultados confirmaram a capacidade do método em manter o erro de sincronização dentro de faixas seguras. Além disso, o ajuste dos parâmetros de controle e das redes neurais demonstrou flexibilidade para lidar com diferentes configurações e velocidades de convergência.

Em síntese, o conjunto de experimentos e análises realizadas comprova a eficácia da lei de controle neural e sub-atuada não apenas para a sincronização de sistemas hipercaóticos complexos, mas também para aplicações práticas envolvendo comunicação segura,

sinalizando o potencial de extensão e inovação tecnológica na área de controle de sistemas não lineares.

5.2 Sumário de Contribuições do Trabalho

Nesta dissertação, foi introduzida e validada uma abordagem de *sincronização neural e sub-atuada* aplicada a um sistema hipercaótico tetradimensional. Ao longo de seu desenvolvimento, destacam-se as seguintes contribuições específicas:

1. **Proposição de um Controlador Sub-atuado para Sistemas Hipercaóticos**
    - Ao contrário de métodos convencionais que exigem a atuação em todas as variáveis de estado, o controlador desenvolvido atua apenas sobre parte das variáveis, reduzindo significativamente a complexidade de implementação.
    - Essa redução de atuadores não compromete a estabilidade do sistema, e a prova de estabilidade evidenciou a eficácia em cenários de não linearidades elevadas.
2. **Integração de Redes Neurais como Elemento de Compensação Adaptativa**
    - As redes neurais foram empregadas para lidar com incertezas e termos não modelados, adaptando-se dinamicamente à evolução do sistema hipercaótico.
    - A utilização das redes neurais, em conjunto com os fundamentos de Lyapunov, formou um arcabouço robusto para garantir que o erro de sincronização permanecesse baixo, mesmo na presença de perturbações ou parâmetros incertos.
3. **Demonstração de Estabilidade Prática em Regime Hipercaótico**
    - Em cenários altamente instáveis, a estabilidade prática (em vez de estritamente assintótica) mostrou-se suficiente para manter o sistema escravo em proximidade das trajetórias do sistema mestre.
    - Isso abre caminho para aplicações em situações onde ruídos externos impeçam uma convergência exata.
4. **Aplicação em Comunicação Segura**
    - Foi desenvolvida e validada uma estratégia para encriptar e recuperar duas mensagens simultaneamente utilizando o comportamento hipercaótico, evidenciando a potencialidade do método em mascarar sinais de forma que um terceiro não autorizado não consiga extrair o conteúdo.
    - Os resultados das simulações demonstraram valores de erros baixos na recuperação do sinal, reforçando a viabilidade prática em cenários de transmissão confidencial de dados.

5. **Flexibilidade e Extensibilidade**

    - A abordagem de controle sub-atuado e redes neurais pode ser estendida a outros sistemas não lineares de maior dimensão, desde que sejam respeitadas as hipóteses de projeto.
    - Parâmetros como os ganhos de controle (λ_i) e as taxas de aprendizado neural podem ser ajustados para atender diferentes requisitos de robustez, rapidez de convergência e limitações de hardware.

Em conjunto, essas contribuições reforçam a ideia de que a sincronização sub-atuada apoiada em redes neurais não apenas colabora para o avanço teórico sobre controle de sistemas hipercaóticos, mas também sinaliza aplicações práticas em áreas como comunicação segura, processamento de sinais não lineares e engenharia de controle em geral.

5.3 Publicação de Trabalhos com Resultados Preliminares

No decorrer do desenvolvimento deste mestrado, foram alcançados resultados preliminares relacionados à sincronização de sistemas hipercaóticos sub-atuados, sob a ênfase em *comunicação segura*. Esses resultados foram consolidados em duas publicações científicas, listadas nas Referências Bibliográficas:

1. *Hyperchaos-Based Secure Communication Using Lyapunov Theory* (GULARTE, K. H.; HARA et al., 2023a).
2. *Secure Communications in the Presence of Disturbances Based on Lyapunov Theory* (GULARTE, K. H.; HARA et al., 2023b).

Em ambas as publicações, propõe-se a utilização de técnicas de controle sub-atuado fundamentadas na teoria de Lyapunov para manter a sincronização entre um sistema mestre e um sistema escravo hipercaóticos, possibilitando a cifragem de mensagens através do mascaramento das variáveis de estado. Embora esses trabalhos não adotem redes neurais em sua estrutura de controle, as soluções apresentadas demonstraram que o *controle sub-atuado* — mesmo sem camadas de adaptação — pode garantir a sincronização em diversos cenários, mostrando-se viável para aplicações de comunicação segura onde a redução de complexidade e custo (em termos de atuação) seja primordial.

O material publicado reforça a idéia de que o uso de sistemas hipercaóticos e de métodos de controle baseados em Lyapunov forma uma base sólida para garantir a *confidencialidade* das comunicações. A partir dessas contribuições iniciais, o presente trabalho

evoluiu para incorporar *redes neurais adaptativas*, aprofundando a robustez e a versatilidade da solução de sincronização, conforme discutido nos capítulos anteriores. Dessa forma, a pesquisa atual pode ser considerada uma extensão e aprimoramento natural das técnicas preliminares, ampliando o escopo e a eficácia no tratamento de não linearidades e distúrbios mais complexos.

5.4 Trabalhos Futuros

Apesar dos resultados positivos obtidos ao longo desta dissertação, há alguns aspectos que podem ser aprofundados ou estendidos em pesquisas futuras:

1. **Implementação Discreta do Controlador** Uma possibilidade de evolução consiste em desenvolver uma *versão discreta* do controlador neural sub-atuado, adaptada a sistemas amostrados. Nesse cenário, as equações diferenciais e as leis de adaptação neural precisariam ser reformuladas em termos de diferenças finitas ou em função de um período de amostragem definido. Essa abordagem permitiria a *execução do controle em plataformas digitais* (por exemplo, microcontroladores, DSPs ou FPGAs), viabilizando testes em tempo real e assegurando que atrasos e quantizações inerentes a processos discretos não comprometam a convergência e a robustez do método. Além disso, a implementação discreta abriria caminho para estratégias de otimização e ajuste de parâmetros em algoritmos de controle adaptativo *on-line*, levando a soluções ainda mais eficientes e compatíveis com restrições práticas de hardware.

2. **Aplicação em Outros Sistemas Dinâmicos**
   Estender o método a sistemas hipercaóticos de dimensão superior ou a outros modelos não lineares complexos, verificando se a estabilidade prática e a sub-atuação mantêm sua eficácia em cenários ainda mais desafiadores.

3. **Análise de Desempenho em Hardware**
   Implementar o sistema utilizando componentes de eletrônica e verificar o seu desempenho e suas limitações em um cenário em que os sinais são influenciados pelas limitações e não-idealidades dos componentes.

4. **Comunicação Segura em Larga Escala**
   Ampliar os testes de comunicação caótica para cenários multiusuário (Como em comunicações *vehicle-to-everything* (V2X), por exemplo) ou de redes, com diferentes níveis de ruído e atenuação. Analisar a capacidade de canal e a taxa de transmissão quando se emprega a sincronização hipercaótica em diferentes protocolos de comunicação.

Essas direções sinalizam oportunidades de evolução tanto no âmbito teórico quanto na aplicação prática, consolidando e expandindo o potencial da sincronização neural sub-

atuada em sistemas hipercaóticos para diferentes nichos de pesquisa e desenvolvimento tecnológico.

Referências

ANH, H. P. H.; DAT, N. T. Online Adaptive Neural Observer for Prescribed Performance Hyper-Chaotic Systems. **Knowledge-Based Systems**, Elsevier, p. 112021, 2024. Citado na p. 18.

BABANLI, K. M.; KABAOGLU, R. O. Synchronization of fuzzy-chaotic systems with Z-controller in secure communication. **Information Sciences**, Elsevier, v. 657, p. 119988, 2024. Citado na p. 17.

BASHIR, Z.; MALIK, M. A.; HUSSAIN, S. A computational study of fractional variable-order nonlinear Newton–Leipnik chaotic system with radial basis function network. **The Journal of Supercomputing**, Springer, v. 81, n. 1, p. 152, 2025. Citado na p. 17.

BEDROSSIAN, N. S. **Nonlinear control using linearizing transformations**. 1991. Tese (Doutorado) – Massachusetts Institute of Technology. Citado nas pp. 18, 19.

CHEN, G. **Controlling chaos and bifurcations in engineering systems**. CRC press, 1999. Citado na p. 28.

CLEMENTE-LOPEZ, D.; JESUS RANGEL-MAGDALENO, J. de; MUÑOZ-PACHECO, J. M. A lightweight chaos-based encryption scheme for IoT healthcare systems. **Internet of Things**, Elsevier, v. 25, p. 101032, 2024. Citado na p. 17.

CONOVER, W. J. **Practical nonparametric statistics**. john wiley & sons, 1999. Citado na p. 33.

COVER, T. M. **Elements of information theory**. John Wiley & Sons, 1999. Citado na p. 34.

CUOMO, K. M.; OPPENHEIM, A. V.; STROGATZ, S. H. Synchronization of Lorenz-based chaotic circuits with applications to communications. **IEEE Transactions on circuits and systems II: Analog and digital signal processing**, IEEE, v. 40, n. 10, p. 626–633, 1993. Citado na p. 19.

DHINGRA, D.; DUA, M. Novel multiple video encryption scheme using two-chaotic-map-based two-level permutation and diffusion. **Nonlinear Dynamics**, Springer, p. 1–27, 2025. Citado na p. 17.

DONG, X.; WANG, D.; LU, J.; HE, X. A wind power forecasting model based on polynomial chaotic expansion and numerical weather prediction. **Electric Power Systems Research**, Elsevier, v. 227, p. 109983, 2024. Citado na p. 17.

DUARTE, A. L.; EISENCRAFT, M. Denoising of discrete-time chaotic signals using echo state networks. **Signal Processing**, Elsevier, v. 214, p. 109252, 2024. Citado na p. 17.

FRISON, T. W. Controlling chaos with a neural network. In: IEEE. [PROCEEDINGS 1992] IJCNN International Joint Conference on Neural Networks. 1992. v. 2, p. 75–80. Citado nas pp. 18, 19.

GA, T.; BHANU, S. M. S. Chebyshev chaotic map with attribute based encryption on session based data-sharing in fog environment. **Peer-to-Peer Networking and Applications**, Springer, v. 18, n. 1, p. 1–25, 2025. Citado na p. 17.

GULARTE, K. H. M.; ALVES, L. M.; VARGAS, J. A. R.; ALFARO, S. C. A.; DE CARVALHO, G. C.; ROMERO, J. F. A. Secure communication based on hyperchaotic underactuated projective synchronization. **Ieee Access**, IEEE, v. 9, p. 166117–166128, 2021. Citado na p. 18.

GULARTE, K. H. M.; GÓMEZ, J. C. G.; SANTOS RABELO, H. dos; VARGAS, J. A. R. Minimal underactuated synchronization with applications to secure communication. **Communications in Nonlinear Science and Numerical Simulation**, Elsevier, v. 125, p. 107376, 2023. Citado na p. 18.

GULARTE, K. H.; HARA, F. O.; VARGAS, J. A.; GUIMARÃES, F. O. Hyperchaos-Based Secure Communication Using Lyapunov Theory. In: IEEE. 2023 15th IEEE International Conference on Industry Applications (INDUSCON). 2023a. P. 747–751. Citado nas pp. 17, 79.

GULARTE, K. H.; HARA, F. O.; VARGAS, J. A.; GUIMARÃES, F. O. Secure Communications in the Presence of Disturbances Based on Lyapunov Theory. In: IEEE. 2023 15th IEEE International Conference on Industry Applications (INDUSCON). 2023b. P. 512–517. Citado nas pp. 17, 79.

GULARTE, K. H.; RÊGO, L. N.; VARGAS, J. A. Scheme for chaos-based encryption and lyapunov analysis. In: IEEE. 2018 IEEE International Conference on Automation/XXIII Congress of the Chilean Association of Automatic Control (ICA-ACCA). 2018. P. 1–7. Citado na p. 19.

HAYKIN, S. **Neural networks: a comprehensive foundation**. Prentice Hall PTR, 1998. Citado na p. 36.

HORNIK, K.; STINCHCOMBE, M.; WHITE, H. Multilayer feedforward networks are universal approximators. **Neural networks**, Elsevier, v. 2, n. 5, p. 359–366, 1989. Citado na p. 38.

HUSSAIN, M. M.; SIDDIQUE, M.; ALMOHAIMEED, Z. M.; SHAMSHAD, R.; AKRAM, R.; ASLAM, N. Synchronization of Chaotic Systems: A Generic Nonlinear Integrated Observer-Based Approach. **Complexity**, Wiley Online Library, v. 2021, n. 1, p. 4558400, 2021. Citado na p. 17.

JIN, J.; FANG, J.; CHEN, C.; LI, Z.; YU, F. A complex-valued time varying zeroing neural network model for synchronization of complex chaotic systems. **Nonlinear Dynamics**, Springer, p. 1–21, 2024. Citado na p. 18.

JIN, M.; SUN, K.; WANG, H. Hyperchaos, extreme multistability, and hidden attractors in the novel complex nonlinear system and its adaptive hybrid synchronization. **Nonlinear Dynamics**, Springer, v. 110, n. 4, p. 3853–3867, 2022. Citado na p. 17.

KARTAL, S. A discrete fractional order cournot duopoly game model with relative profit delegation: Stability, bifurcation, chaos, 0-1 testing and control. **Journal of Computational and Applied Mathematics**, Elsevier, v. 457, p. 116284, 2025. Citado na p. 17.

KENDALL, M. G. A new measure of rank correlation. **Biometrika**, Oxford University Press, v. 30, n. 1-2, p. 81–93, 1938. Citado na p. 33.

KHALIL, H. K. Lyapunov stability. **Control systems, robotics and automation**, v. 12, p. 115, 2009. Citado nas pp. 26, 27, 29.

KOSMATOPOULOS, E. B.; POLYCARPOU, M. M.; CHRISTODOULOU, M. A.; IOANNOU, P. A. High-order neural network structures for identification of dynamical systems. **IEEE transactions on Neural Networks**, IEEE, v. 6, n. 2, p. 422–431, 1995. Citado na p. 53.

LAAREM, G. A new 4-D hyper chaotic system generated from the 3-D Rösslor chaotic system, dynamical analysis, chaos stabilization via an optimized linear feedback control, it's fractional order model and chaos synchronization using optimized fractional order sliding mode control. **Chaos, Solitons & Fractals**, Elsevier, v. 152, p. 111437, 2021. Citado na p. 17.

LETELLIER, C.; ROSSLER, O. E. Hyperchaos. **Scholarpedia**, v. 2, n. 8, p. 1936, 2007. Citado na p. 17.

LORENZ, E. Chaos in meteorological forecast. **Journal of the Atmospheric Sciences**, v. 20, n. 2, p. 130–141, 1963. Citado nas pp. 18, 19, 23, 24.

LORENZ, E. Predictability: Does the flap of a butterfly's wing in Brazil set off a tornado in Texas? na, 1972. Citado na p. 17.

MANHIL, M. M.; JAMAL, R. K. A novel secure communication system using Duffing's chaotic model. **Multimedia Tools and Applications**, Springer, p. 1–14, 2024. Citado na p. 17.

MOYSIS, L.; PETAVRATZIS, E.; MARWAN, M.; VOLOS, C.; NISTAZAKIS, H.; AHMAD, S. Analysis, synchronization, and robotic application of a modified hyperjerk chaotic system. **Complexity**, Wiley Online Library, v. 2020, n. 1, p. 2826850, 2020. Citado na p. 17.

NAZARÉ, T. E.; NEPOMUCENO, E. G.; MARTINS, S. A.; BUTUSOV, D. N. A note on the reproducibility of chaos simulation. **Entropy**, MDPI, v. 22, n. 9, p. 953, 2020. Citado na p. 52.

OPPENHEIM, A. V.; VERGHESE, G. C. **Signals, systems & inference**. Pearson London, 2017. Citado na p. 30.

PARBAT, D.; CHAKRABORTY, M. A novel methodology to study the cognitive load induced EEG complexity changes: Chaos, fractal and entropy based approach. **Biomedical Signal Processing and Control**, Elsevier, v. 64, p. 102277, 2021. Citado na p. 17.

PECORA, L. M.; CARROLL, T. L. Synchronization in chaotic systems. **Physical review letters**, APS, v. 64, n. 8, p. 821, 1990. Citado nas pp. 18, 19.

PENG, Y.; HE, S.; SUN, K. Parameter identification for discrete memristive chaotic map using adaptive differential evolution algorithm. **Nonlinear Dynamics**, Springer, v. 107, n. 1, p. 1263–1275, 2022. Citado na p. 17.

PRAJAPAT, S.; KUMAR, D.; KUMAR, P. Quantum image encryption protocol for secure communication in healthcare networks. **Cluster Computing**, Springer, v. 28, n. 1, p. 3, 2025. Citado na p. 17.

RAZZAQ, R.; KHAN, Z.; ABRAR, M.; ALMOHSEN, B.; FAROOQ, U. Chemical reaction and radiation analysis for the MHD Casson nanofluid fluid flow using artificial intelligence. **Chaos, Solitons & Fractals**, Elsevier, v. 190, p. 115756, 2025. Citado na p. 17.

ROSSLER, O. An equation for hyperchaos. **Physics Letters A**, Elsevier, v. 71, n. 2-3, p. 155–157, 1979. Citado nas pp. 18, 19.

SCHNEIER, B. **Applied Cryptography: Protocols, Algorithms, and Source Code in C**. 2nd. New York, NY, USA: John Wiley & Sons, 1996. Citado na p. 63.

SHAFIQ, M.; AHMAD, I. Continuous-Time Robust Adaptive Controller Design for Nonlinear Chaotic Jerk Circuit System Stabilization. **Arabian Journal for Science and Engineering**, Springer, p. 1–12, 2025. Citado na p. 17.

SHANNON, C. E. A mathematical theory of communication. **The Bell system technical journal**, Nokia Bell Labs, v. 27, n. 3, p. 379–423, 1948. Citado na p. 34.

SLOTINE, J.-J. E.; LI, W. et al. **Applied nonlinear control**. Prentice hall Englewood Cliffs, NJ, 1991. v. 199. Citado nas pp. 26, 28.

STALLINGS, W. **Cryptography and Network Security: Principles and Practice**. 7th: Pearson, 2017. Citado na p. 63.

TADJ, M.; CHAIB, L.; CHOUCHA, A.; ALHAZMI, M.; ALWABLI, A.; BAJAJ, M.; DOST MOHAMMADI, S. A. Improved chaotic Bat algorithm for optimal coordinated tuning of power system stabilizers for multimachine power system. **Scientific Reports**, Nature Publishing Group UK London, v. 14, n. 1, p. 15124, 2024. Citado na p. 17.

WANG, A.; WANG, J.; JIANG, L.; WANG, L.; WANG, Y.; YAN, L.; QIN, Y. Experimental demonstration of 8190-km long-haul semiconductor-laser chaos synchronization induced by digital optical communication signal. **Light: Science & Applications**, Nature Publishing Group UK London, v. 14, n. 1, p. 40, 2025. Citado na p. 17.

WANG, S. A novel hyperchaotic system with fast and slow attractors. **AIP Advances**, AIP Publishing, v. 12, n. 10, 2022. Citado nas pp. 19, 20, 39.

WEN, H.; LIN, Y. Cryptanalysis of an image encryption algorithm using quantum chaotic map and DNA coding. **Expert Systems with Applications**, Elsevier, v. 237, p. 121514, 2024. Citado na p. 17.

YANG, Y.; QIN, S.; LIAO, S. Ultra-chaos of a mobile robot: A higher disorder than normal-chaos. **Chaos, Solitons & Fractals**, Elsevier, v. 167, p. 113037, 2023. Citado na p. 17.

YU, S.; CHEN, W.; POOR, H. V. Real-Time Monitoring of Chaotic Systems With Known Dynamical Equations. **IEEE Transactions on Signal Processing**, IEEE, 2024. Citado na p. 17.

Apêndices

APÊNDICE A – Códigos de simulação no MATLAB

A.1 Código para a Planta Master (Mestre)

Código A.1 – Planta_Master.m

```
function [sys,x0,str,ts] = Sistema(t,x,u,flag)

%constantes
a = 35;
b = 4.9;
c = 25;
d = 5;
e = 35;
f = 100;

gamma3 = 0.09;
gamma1 = 0.02;
gamma2 = (gamma1*e + gamma3)/d;

%mensagens a serem transmitidas
messageX = 12*sin(2*pi*1.7*t+pi/3)+11*cos(2*pi*0.8*t);
messageZ = 3.6*square(2*pi*1.25*t)+2.4*square(2*pi*1.53*t+pi/5);

switch flag,
    %%%%%%%%%%%%%%%%%%
    % Inicialização  %
    %%%%%%%%%%%%%%%%%%
    case 0,
      sizes = simsizes;
        sizes.NumContStates = 4; %Número de estados contínuos
        sizes.NumDiscStates = 0; %Número de estados discretos
        sizes.NumOutputs = 4;    %Número de saídas
        sizes.NumInputs = 0;     %Número de entradas
        sizes.DirFeedthrough = 1;
        sizes.NumSampleTimes = 1;
        sys = simsizes(sizes);
        x0=[1/gamma1 1/gamma2 1/gamma3 1]; %Condições iniciais
      str=[];
      ts=[0 0];
        %%%%%%%%%%%%%%%
        % Diretivas   %
        %%%%%%%%%%%%%%%
    case 1,      %Planta do sistema mestre
  sys = [a*(x(2)-x(1))+gamma1*e*x(2)*x(3);
```

```
          c*x(1)-gamma2*d*x(1)*x(3)+x(2)+x(4);
          gamma3*x(1)*x(2)-b*x(3);
          -f*x(2)];
        %%%%%%%%%%
        % Saídas %
        %%%%%%%%%%
    case 3,
        sys = [x(1) + messageX; x(2); x(3) + messageZ; x(4)];
        %%%%%%%%%
        %  Fim  %
        %%%%%%%%%
    case {2,4,9},
        sys = []; % Não faz nada
    otherwise
        error(['unhandled flag = ',num2str(flag)]);
end
```

A.2 Código para a Planta Slave (Escravo)

Código A.2 – Planta_Slave.m

```
function [sys,x0,str,ts] = Sistema(t,x,u,flag)

%constantes
a = 35;
b = 4.9;
c = 25;
d = 5;
e = 35;
f = 100;

gamma3 = 0.09;
gamma1 = 0.02;
gamma2 = (gamma1*e + gamma3)/d;

%OBS não foi necessário incluir o disturbio pois o mesmo foi
   adicionado no Simulink

switch flag,
   %%%%%%%%%%%%%%%%%%
   % Inicialização  %
   %%%%%%%%%%%%%%%%%%
case 0,
  sizes = simsizes;
    sizes.NumContStates = 4; %Número de estados contínuos
    sizes.NumDiscStates = 0; %Número de estados discretos
    sizes.NumOutputs = 6;    %Número de saídas
    sizes.NumInputs = 7;     %Número de entradas
    sizes.DirFeedthrough = 1;
    sizes.NumSampleTimes = 1;
```

```
    sys = simsizes(sizes);
    x0=[-2/gamma1 2/gamma2 3/gamma3 4]; %Condições iniciais
  str=[];
  ts=[0 0];
   %%%%%%%%%%%%%%%
   % Diretivas   %
   %%%%%%%%%%%%%%%
    case 1,      %Planta do sistema escravo
  sys = [a*(x(2)-x(1))+gamma1*e*x(2)*x(3)+u(1)+d1;
         c*x(1)-gamma2*d*x(1)*x(3)+x(2)+x(4)+u(2)+d2;
         gamma3*x(1)*x(2)-b*x(3)+u(3)+d3;
         -f*x(2)+u(4)+d4];
   %%%%%%%%%%
   % Saídas %
   %%%%%%%%%%
case 3,
   sys = [x(1);x(2);x(3);x(4);u(6);u(7)];
   %%%%%%%%%
   %  Fim  %
   %%%%%%%%%
case {2,4,9},
    sys = []; % Não faz nada
 otherwise
   error(['unhandled flag = ',num2str(flag)]);
end
```

A.3 Código para o Sincronizador

Código A.3 – Sincronizador.m

```

function [sys,x0,str,ts] = Sincronizador(t,x,u,flag)

%constantes sincronizador
a = 35;
b = 4.9;
c = 25;
d = 5;
e = 35;
f = 100;

%constantes definidas pelo usuario
sigma2 = 10;
sigma4 = 10;

lambda2 = 5000;
lambda4 = 5000;

% parametros iniciais matriz w
W2=[1 0 0 0 0 0 0 0]';
```

```
W4=[1 0 0 0 0 0 0 0]';

switch flag
    %%%%%%%%%%%%%%%%%
    % Inicialização %
    %%%%%%%%%%%%%%%%%
    case 0

        sizes = simsizes;
        sizes.NumContStates  = 16;   %Número de estados continuos
        sizes.NumDiscStates  = 0;   %Número de estados discretos
        sizes.NumOutputs     = 7;   %Número de saídas
        sizes.NumInputs      = 10;   %Número de entradas
        sizes.DirFeedthrough = 1;
        sizes.NumSampleTimes = 1;
        sys = simsizes(sizes);
        x0=zeros(16,1);              %Condições iniciais
        str=[];
        ts=[0 0];

        %%%%%%%%%%%%%%
        % Diretivas  %
        %%%%%%%%%%%%%%
    case 1  %aqui ficam os estimadores dos pesos de uma rede neural
            sys = [(u(6)-u(2))*Z(u) - sigma2*(x(1:8)-W2);
                   (u(8)-u(4))*Z(u) - sigma4*(x(9:16)-W4)];
                  % lei de aprendizado

        %%%%%%%%%%
        % Saídas %
        %%%%%%%%%%
    case 3  %saidas u enviadas para o sistema escravo
        sys =  [0;%-lambda1*(u(5)-u(1));
                -(x(1:8)-W2)'*Z(u)-lambda2*(u(6)-u(2));
                0;
                -(x(9:16)-W4)'*Z(u)-lambda4*(u(8)-u(4));
                u(2);
                norm(x(1:8));
                norm(x(9:16))]; %pesos das redes
    case {2,4,9}
        sys = [];

    otherwise
        error(['unhandled flag = ',num2str(flag)]);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function out = Z(u)     %Regressor
out=[1*(sig(u(5)));
    1*(sig(u(6)));
    1*(sig(u(7)));
    1*(sig(u(8)));
```

```
    1*(sig(u(5))^2);
    1*(sig(u(6))^2);
    1*(sig(u(7))^2);
    1*(sig(u(8))^2)];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function out = sig(uu)  %funcao de ativacao
lamda=0;
alfa=5;
beta=.5;
out=alfa/(exp(-beta*uu)+1)+lamda;
```

A.4 Código para a obtenção dos gráficos

Código A.4 – Graficos.m

```
%Executando esse arquivo --> automaticamente mostra os gráficos da
%simulação e salva na pasta em formato png e epsc
clc
close all

format1 = 'png';
format2 = 'epsc';

fSize = 38;
axesSize = 38;
lSize = 3.6;
dvlsize = 2;
dhlsize = 2;
fonte = 38;
largura_linha = 2;
color1 = [0 0.4470 0.7410];
color2 = [0.8500 0.3250 0.0980];
color3 = [0.4660 0.6740 0.1880];

messageX = 12*sin(2*pi*1.7*t+pi/3)+11*cos(2*pi*0.8*t);
messageZ = 3.6*square(2*pi*1.25*t)+2.4*square(2*pi*1.53*t+pi/5);

set(0,'DefaultAxesFontSize',axesSize);

%Figura 1
fig=figure('visible','off', 'DefaultAxesPosition', [0.096, 0.172,
   0.890, 0.800]);
plot(t,Xmaster(:,1),t, Xslave(:,1),':','LineWidth',lSize);
grid on
grid minor
h=legend('$x_m(t)$','$x_s(t)$','Location','southeast');
set(h,'Interpreter','Latex','FontSize',fSize);
ylim([-180 150]);
xlabel('$t[s]$','Interpreter','Latex','Fontsize',fSize);
```

```
34 set(gcf,'units','normalized','outerposition',[0 0 1 1]);
35 saveas(gcf,'Figuras/1_XsXm', format1);
36 saveas(gcf,'Figuras/1_XsXm', format2);
37 close(fig)
38
39 %Figura 2
40 fig=figure('visible','off', 'DefaultAxesPosition', [0.096, 0.172,
      0.890, 0.800]);
41 plot(t,Xmaster(:,2),t, Xslave(:,2),':','LineWidth',lSize);
42 grid on
43 grid minor
44 h=legend('$y_m(t)$','$y_s(t)$','Location','southeast');
45 set(h,'Interpreter','Latex','FontSize',fSize);
46 ylim([-140 100]);
47 xlabel('$t[s]$','Interpreter','Latex','Fontsize',fSize);
48 set(gcf,'units','normalized','outerposition',[0 0 1 1]);
49 saveas(gcf,'Figuras/2_YsYm', format1);
50 saveas(gcf,'Figuras/2_YsYm', format2);
51 close(fig)
52
53 %Figura 3
54 fig=figure('visible','off', 'DefaultAxesPosition', [0.096, 0.172,
      0.890, 0.800]);
55 plot(t,Xmaster(:,3),t, Xslave(:,3),':','LineWidth',lSize);
56 grid on
57 grid minor
58 h=legend('$z_m(t)$','$z_s(t)$','Location','southeast');
59 set(h,'Interpreter','Latex','FontSize',fSize);
60 ylim([-15 65]);
61 xlabel('$t[s]$','Interpreter','Latex','Fontsize',fSize);
62 set(gcf,'units','normalized','outerposition',[0 0 1 1]);
63 saveas(gcf,'Figuras/3_ZsZm', format1);
64 saveas(gcf,'Figuras/3_ZsZm', format2);
65 close(fig)
66
67 %Figura 4
68 fig=figure('visible','off', 'DefaultAxesPosition', [0.096, 0.172,
      0.890, 0.800]);
69 plot(t,Xmaster(:,4),t, Xslave(:,4),':','LineWidth',lSize);
70 grid on
71 grid minor
72 h=legend('$w_m(t)$','$w_s(t)$','Location','southeast');
73 set(h,'Interpreter','Latex','FontSize',fSize);
74 xlabel('$t[s]$','Interpreter','Latex','Fontsize',fSize);
75 set(gcf,'units','normalized','outerposition',[0 0 1 1]);
76 saveas(gcf,'Figuras/4_WsWm', format1);
77 saveas(gcf,'Figuras/4_WsWm', format2);
78 close(fig)
79
80 %Figura 5
81 fig=figure('visible','off', 'DefaultAxesPosition', [0.096, 0.172,
      0.890, 0.800]);
```

```
82  plot(t, Xslave(:,5),'LineWidth',lSize);
83  grid on
84  grid minor
85  h=legend('$||\widehat{W}_{2}(t)||$','Location','southeast');
86  set(h,'Interpreter','Latex','FontSize',fSize);
87  ylim([0 1.5]);
88  xlabel('$t[s]$','Interpreter','Latex','Fontsize',fSize);
89  set(gcf,'units','normalized','outerposition',[0 0 1 1]);
90  saveas(gcf,'Figuras/5_Norma_Pesos_Estimados', format1);
91  saveas(gcf,'Figuras/5_Norma_Pesos_Estimados', format2);
92  close(fig)
93
94  %Figura 6
95  fig=figure('visible','off', 'DefaultAxesPosition', [0.096, 0.172,
        0.890, 0.800]);
96  plot(t, Xslave(:,6),'LineWidth',lSize);
97  grid on
98  grid minor
99  h=legend('$||\widehat{W}_{4}(t)||$','Location','southeast');
100 set(h,'Interpreter','Latex','FontSize',fSize);
101 ylim([0 1.5]);
102 xlabel('$t[s]$','Interpreter','Latex','Fontsize',fSize);
103 set(gcf,'units','normalized','outerposition',[0 0 1 1]);
104 saveas(gcf,'Figuras/6_Norma_Pesos_Estimados', format1);
105 saveas(gcf,'Figuras/6_Norma_Pesos_Estimados', format2);
106 close(fig)
107
108 %Figura 7
109 fig=figure('visible','off', 'DefaultAxesPosition', [0.096, 0.172,
        0.890, 0.800]);
110 plot(t,Xslave(:,1)-Xmaster(:,1),'LineWidth',lSize);
111 grid on
112 grid minor
113 h=legend('$e_1(t)$','Location','southeast');
114 set(h,'Interpreter','Latex','FontSize',fSize);
115 ylim([-100 50]);
116 xlabel('$t[s]$','Interpreter','Latex','Fontsize',fSize);
117 set(gcf,'units','normalized','outerposition',[0 0 1 1]);
118 saveas(gcf,'Figuras/7_e1', format1);
119 saveas(gcf,'Figuras/7_e1', format2);
120 close(fig)
121
122 %Figura 8
123 fig=figure('visible','off', 'DefaultAxesPosition', [0.096, 0.172,
        0.890, 0.800]);
124 plot(t,Xslave(:,2)-Xmaster(:,2),'LineWidth',lSize);
125 grid on
126 grid minor
127 h=legend('$e_2(t)$','Location','southeast');
128 set(h,'Interpreter','Latex','FontSize',fSize);
129 ylim([-1 2]);
130 xlabel('$t[s]$','Interpreter','Latex','Fontsize',fSize);
```

```
131 set(gcf,'units','normalized','outerposition',[0 0 1 1]);
132 saveas(gcf,'Figuras/8_e2', format1);
133 saveas(gcf,'Figuras/8_e2', format2);
134 close(fig)
135
136 %Figura 9
137 fig=figure('visible','off', 'DefaultAxesPosition', [0.096, 0.172,
      0.890, 0.800]);
138 plot(t,Xslave(:,3)-Xmaster(:,3),'LineWidth',lSize);
139 grid on
140 grid minor
141 h=legend('$e_3(t)$','Location','southeast');
142 set(h,'Interpreter','Latex','FontSize',fSize);
143 ylim([-15 15]);
144 xlabel('$t[s]$','Interpreter','Latex','Fontsize',fSize);
145 set(gcf,'units','normalized','outerposition',[0 0 1 1]);
146 saveas(gcf,'Figuras/9_e3', format1);
147 saveas(gcf,'Figuras/9_e3', format2);
148 close(fig)
149
150 %Figura 10
151 fig=figure('visible','off', 'DefaultAxesPosition', [0.096, 0.172,
      0.890, 0.800]);
152 plot(t,Xslave(:,4)-Xmaster(:,4),'LineWidth',lSize);
153 grid on
154 grid minor
155 h=legend('$e_4(t)$','Location','southeast');
156 set(h,'Interpreter','Latex','FontSize',fSize);
157 ylim([-0.2 0.2]);
158 xlabel('$t[s]$','Interpreter','Latex','Fontsize',fSize);
159 set(gcf,'units','normalized','outerposition',[0 0 1 1]);
160 saveas(gcf,'Figuras/10_e4', format1);
161 saveas(gcf,'Figuras/10_e4', format2);
162 close(fig)
163
164 % %Figura 11
165 fig=figure('visible','off', 'DefaultAxesPosition', [0.060, 0.172,
      0.920, 0.800]);
166 plot(t,5*messageX,t,Xmaster(:,1),':','LineWidth',lSize);
167 grid on
168 grid minor
169 h=legend('$$5 m_{x}(t)$$', '$$s_{x}(t)$$','Location','southeast');
170 set(h,'Interpreter','Latex','FontSize',fSize);
171 ylim([-220 180]);
172 xlabel('$t[s]$','Interpreter','Latex','Fontsize',fSize);
173 set(gcf,'units','normalized','outerposition',[0 0 1 1]);
174 saveas(gcf,'Figuras/11_encrypted_message_comparison_X', format1);
175 saveas(gcf,'Figuras/11_encrypted_message_comparison_X', format2);
176 close(fig)
177
178 %Figura 12
```

```
179 fig=figure('visible','off', 'DefaultAxesPosition', [0.096, 0.172,
        0.890, 0.800]);
180 plot(t,5*messageZ + 30,t,Xmaster(:,3),':','LineWidth',lSize);
181 grid on
182 grid minor
183 h=legend('$$5 m_{z}(t)+30$$',
        '$$s_{z}(t)$$','Location','southeast');
184 set(h,'Interpreter','Latex','FontSize',fSize);
185 ylim([-20 65]);
186 xlabel('$t[s]$','Interpreter','Latex','Fontsize',fSize);
187 set(gcf,'units','normalized','outerposition',[0 0 1 1]);
188 saveas(gcf,'Figuras/12_encrypted_message_comparison_Z', format1);
189 saveas(gcf,'Figuras/12_encrypted_message_comparison_Z', format2);
190 close(fig)
191
192 % %Figura 13
193 fig=figure('visible','off', 'DefaultAxesPosition', [0.060, 0.172,
        0.920, 0.800]);
194 plot(t,messageX,t,Xmaster(:,1) -
        Xslave(:,1),':','LineWidth',lSize);
195 grid on
196 grid minor
197 h=legend('$$m_{x}(t)$$',
        '$$\widehat{m}_{x}(t)$$','Location','southeast');
198 set(h,'Interpreter','Latex','FontSize',fSize);
199 ylim([-50 30]);
200 xlabel('$t[s]$','Interpreter','Latex','Fontsize',fSize);
201 set(gcf,'units','normalized','outerposition',[0 0 1 1]);
202 saveas(gcf,'Figuras/13_retrieved_message_comparison_X', format1);
203 saveas(gcf,'Figuras/13_retrieved_message_comparison_X', format2);
204 close(fig)
205
206 %Figura 14
207 fig=figure('visible','off', 'DefaultAxesPosition', [0.096, 0.172,
        0.890, 0.800]);
208 plot(t,messageZ,t,Xmaster(:,3) -
        Xslave(:,3),':','LineWidth',lSize);
209 grid on
210 grid minor
211 h=legend('$$m_{z}(t)$$',
        '$$\widehat{m}_{z}(t)$$','Location','southeast');
212 set(h,'Interpreter','Latex','FontSize',fSize);
213 ylim([-11 7]);
214 xlabel('$t[s]$','Interpreter','Latex','Fontsize',fSize);
215 set(gcf,'units','normalized','outerposition',[0 0 1 1]);
216 saveas(gcf,'Figuras/14_retrieved_message_comparison_Z', format1);
217 saveas(gcf,'Figuras/14_retrieved_message_comparison_Z', format2);
218 close(fig)
219
220 %Figura 15
221 fig=figure('visible','off', 'DefaultAxesPosition', [0.060, 0.172,
        0.920, 0.800]);
```

```
plot(t,messageX - Xmaster(:,1) + Xslave(:,1),'LineWidth',lSize);
grid on
grid minor
h=legend('$$e_{mx}(t)$$','Location','northeast');
set(h,'Interpreter','Latex','FontSize',fSize);
ylim([-20 32]);
xlabel('$t[s]$','Interpreter','Latex','Fontsize',fSize);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'Figuras/15_message_X_error', format1);
saveas(gcf,'Figuras/15_message_X_error', format2);
close(fig)

%Figura 16
fig=figure('visible','off', 'DefaultAxesPosition', [0.096, 0.172,
    0.890, 0.800]);
plot(t,messageZ - Xmaster(:,3) + Xslave(:,3),'LineWidth',lSize);
grid on
grid minor
h=legend('$$e_{mz}(t)$$','Location','northeast');
set(h,'Interpreter','Latex','FontSize',fSize);
ylim([-5 15]);
xlabel('$t[s]$','Interpreter','Latex','Fontsize',fSize);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'Figuras/16_message_Z_error', format1);
saveas(gcf,'Figuras/16_message_Z_error', format2);
close(fig)
```

A.5 Código para a obtenção dos valores de entropia diferencial

Código A.5 – Entropia_Diferencial.m

```
[x_pdf, f_pdf] = ksdensity(sinal);  % 'sinal' é o vetor do sinal
    contínuo e deve ser substituído pelo sinal que se deseja
    calcular a entropia diferencial

% Calcular a entropia diferencial (integral numérica)
H_diff = -trapz(x_pdf, f_pdf .* log2(f_pdf + eps)); % Adiciona eps
    para evitar log(0)
fprintf('Entropia diferencial: %.4f bits\n', H_diff);
```