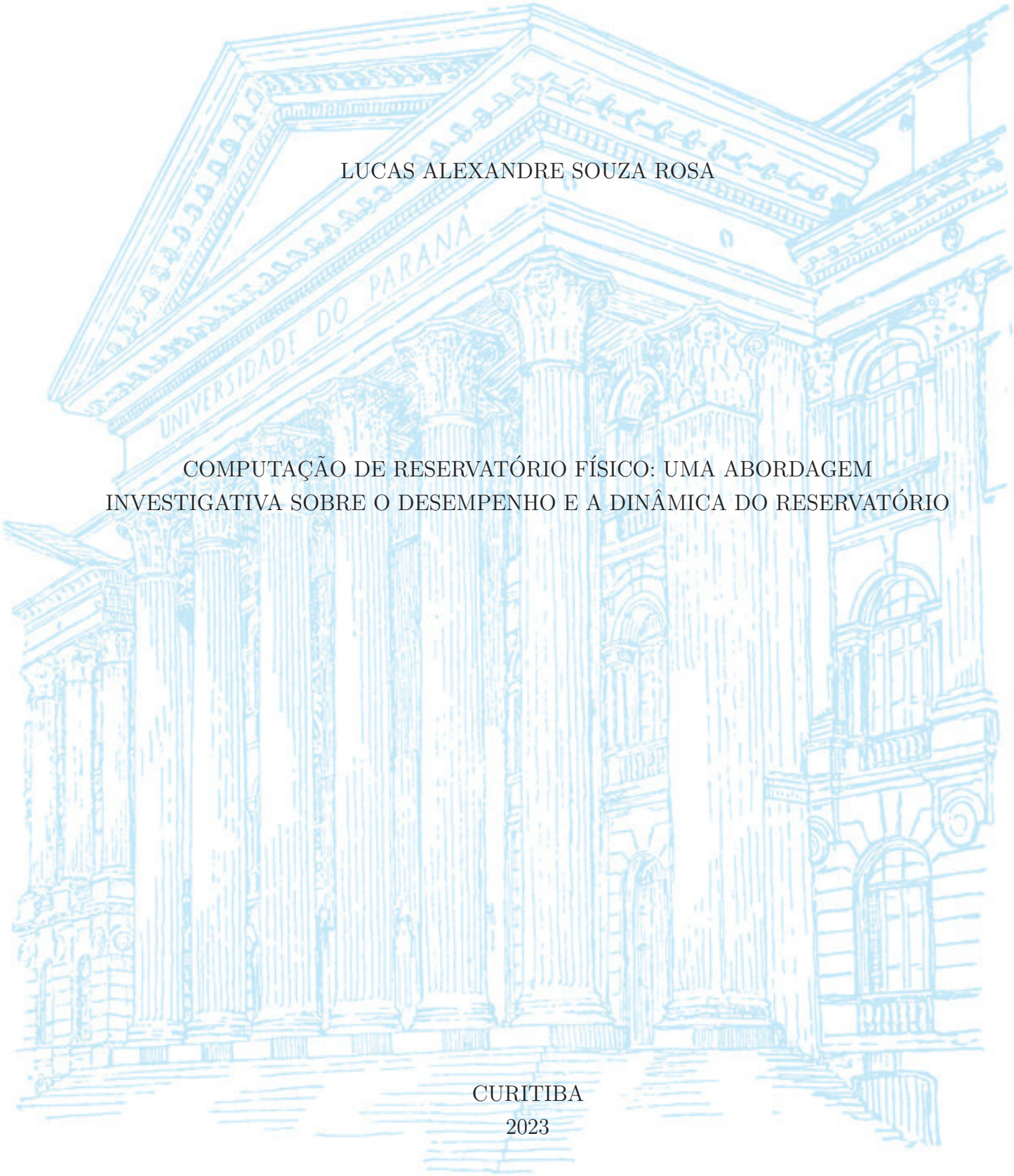
UNIVERSIDADE FEDERAL DO PARANÁ

LUCAS ALEXANDRE SOUZA ROSA



COMPUTAÇÃO DE RESERVATÓRIO FÍSICO: UMA ABORDAGEM INVESTIGATIVA SOBRE O DESEMPENHO E A DINÂMICA DO RESERVATÓRIO

CURITIBA
2023

LUCAS ALEXANDRE SOUZA ROSA

COMPUTAÇÃO DE RESERVATÓRIO FÍSICO: UMA ABORDAGEM INVESTIGATIVA SOBRE O DESEMPENHO E A DINÂMICA DO RESERVATÓRIO

Tese apresentada ao Programa de Pós-Graduação em Física do Setor de Ciências Exatas da Universidade Federal do Paraná como requisito parcial para a obtenção do título de Doutor em Física.
Orientador: Prof. Dr. Marcus Werner Beims.

CURITIBA
2023

DADOS INTERNACIONAIS DE CATALOGAÇÃO NA PUBLICAÇÃO (CIP)
UNIVERSIDADE FEDERAL DO PARANÁ
SISTEMA DE BIBLIOTECAS – BIBLIOTECA CIÊNCIA E TECNOLOGIA

Rosa, Lucas Alexandre Souza
Computação de reservatório físico: uma abordagem investigativa sobre o desempenho e a dinâmica do reservatório. / Lucas Alexandre Souza Rosa. – Curitiba, 2023.
1 recurso on-line : PDF.

Tese (Doutorado) - Universidade Federal do Paraná, Setor de Ciências Exatas, Programa de Pós-Graduação em Física.

Orientador: Prof. Dr. Marcus Werner Beims.

1. Inteligencia artificial. 2. Aprendizado por computador. 3. Algoritmos genéticos. 4. Sistemas dinâmicos. I. Beims, Marcus Werner. II. Universidade Federal do Paraná. Programa de Pós-Graduação em Física. III. Título.

Bibliotecária: Roseny Rivelini Morciani CRB-9/1585

MINISTÉRIO DA EDUCAÇÃO
SETOR DE CIÊNCIAS EXATAS
UNIVERSIDADE FEDERAL DO PARANÁ
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO FÍSICA - 40001016020P4

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação FÍSICA da Universidade Federal do Paraná foram convocados para realizar a arguição da tese de Doutorado de **LUCAS ALEXANDRE SOUZA** intitulada: **"Computação de Reservatório Físico: uma abordagem investigativa sobre o desempenho e a dinâmica do reservatório"** , sob orientação do Prof. Dr. MARCUS WERNER BEIMS, que após terem inquirido o aluno e realizada a avaliação do trabalho, são de parecer pela sua APROVAÇÃO no rito de defesa.

A outorga do título de doutor está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

CURITIBA, 04 de Maio de 2023.

Assinatura Eletrônica
04/05/2023 23:09:42.0
MARCUS WERNER BEIMS
Presidente da Banca Examinadora

Assinatura Eletrônica
09/05/2023 20:21:25.0
EDSON DENIS LEONEL
Avaliador Externo (UNIVERSIDADE EST.PAULISTA JÚLIO DE MESQUITA FILHO)

Assinatura Eletrônica
05/05/2023 09:06:23.0
CESAR MANCHEIN
Avaliador Externo (UNIVERSIDADE DO ESTADO DE SANTA CATARINA)

Assinatura Eletrônica
05/05/2023 07:27:51.0
RICARDO LUIZ VIANA
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

Documento assinado eletronicamente de acordo com o disposto na legislação federal Decreto 8539 de 08 de outubro de 2015.
Gerado e autenticado pelo SIGA-UFPR, com a seguinte identificação única: 281462
Para autenticar este documento/assinatura, acesse https://www.prppg.ufpr.br/siga/visitante/autenticacaoassinaturas.jsp e insira o codigo 281462

Dedico à minha mãe, ao meu vô, à minha vó, à Bruna e a toda a minha família, sem vocês comigo eu nada seria.

Agradecimentos

Agradeço a todos aqueles que, de alguma forma, direta ou indiretamente contribuíram para realização deste trabalho:

- Ao professor Marcus W. Beims pela orientação, pelos conselhos, pela paciência e pela ajuda sempre que eu precisei,
- Ao professor Rafael Marques Da Silva pela supervisão durante os estágios, pelas sugestões e pela amizade,
- Ao Eduardo por ser um grande amigo e parceiro de trabalho, desde os tempos de graduação e de mestrado, por todo o ensinamento compartilhado, pelas conversas sobre espiritualidade, pelos momentos de descontração, e principalmente por sua amizade,
- A minha família pelo incentivo e apoio incondicional que sempre me deram, e pelos valores que me foram passados, principalmente à minha mãe, que deu cada segundo da sua vida para que eu chegasse aqui e onde mais eu queira estar, aos meus avós por cuidarem de mim durante toda a minha vida como um filho, à Bruna por ser minha companheira e sempre me apoiar e incentivar em minha jornada, e aos meus amigos que sempre me fizeram rir mesmo nos momentos mais difíceis,
- Aos amigos da pós por todos os momentos de descontração e à parceria nas disciplinas e na pesquisa,
- Aos professores membros da banca, pelas correções e sugestões dadas ao trabalho,
- À UFPR, pela estrutura e pelos recursos disponibilizados aos alunos,
- À CAPES, pelo apoio financeiro.

"O Senhor é o meu pastor; nada me faltará."
Salmos 23:1.

RESUMO

A Inteligência Artificial apresenta aplicabilidade em diversos campos do conhecimento, e levanta discussões em torno dos limites das suas capacidades cognitivas e de atuação, além dos riscos que o seu desenvolvimento pode acarretar para a humanidade. Especialistas e pessoas influentes nessa área indicam que, em vez de evitar, devemos aprender a lidar com os problemas que a sua evolução possa apresentar. Considerando essa busca por maior compreensão dos processos nas realizações das tarefas, além de somente obter o melhor desempenho, propusemos um modelo de computação de reservatório físico. Para as suas unidades, utilizamos osciladores de Kuramoto, porém adicionamos um termo de campo médio ao modelo matemático proposto originalmente. Nós investigamos como as mudanças nas suas dinâmicas interferem na eficiência do programa, focando no grau de sincronização e na caoticidade do reservatório. Os parâmetros do sistema de Kuramoto foram selecionados por meio do algoritmo genético, uma ferramenta especialmente útil quando há muitos parâmetros envolvidos no modelo. Testamos a aprendizagem de diferentes séries de referência mediante duas técnicas distintas de avaliação de desempenho. Notamos que, quando testamos a capacidade do nosso modelo em emulação e predição, a aprendizagem foi favorecida quando houve baixa sincronização na rede, enquanto na manutenção da distribuição das séries de referência, os parâmetros selecionados pelo algoritmo genético levaram o reservatório a apresentar alta sincronização. Observamos ainda que em ambos os casos, com esses parâmetros, a rede de osciladores apresentou dinâmica hipercaótica, uma vez que houve a presença de alguns expoentes de Lyapunov com sinal positivo.

Palavras-chave: Inteligência Artificial, Aprendizado de Máquina, Computação de Reservatório Físico, Sistemas Dinâmicos Não-Lineares, Caos, Algoritmo Genético.

ABSTRACT

It is known that Artificial Intelligence has applicability in several fields of knowledge. However, it also raises controversy about the limits of its cognitive and acting capacities, besides the risks its development can entail for humankind. Experts and influential people in that area suggest that, instead of avoiding it, we should learn how to deal with the problems that its evolution may present. Considering this pursuit for a greater understanding of task execution processes, in addition to just obtaining the best performance, we have proposed a physical reservoir computing model. We used Kuramoto oscillators as reservoir units, but with the addition of a mean-field term in the original mathematical model. We investigated how changes in their dynamics affect the program efficiency, focusing on the synchronization degree and the chaoticity of the network. The parameters of the Kuramoto system were selected with the genetic algorithm, a tool especially useful when there are many parameters involved in the model. We have tested the learning for different reference series using two performance evaluation techniques. We noticed that, when we evaluated the capacity of our model at emulation and prediction tasks, learning was favored when there was low synchronization in the network, while when the assignment was to maintain the distribution of the reference series, the parameters selected by the genetic algorithm led the reservoir to show high synchronization. We further observed that in both cases, with these parameters, the network of oscillators presented hyperchaotic dynamics, given the presence of some positive Lyapunov exponents.

Keywords: Artificial Intelligence, Machine Learning, Physical Reservoir Computing, Non-Linear Dynamical Systems, Chaos, Genetic Algorithm.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

IA	Inteligência Artificial;
ML	Aprendizado de Máquina;
ANN	Redes neurais artificiais;
FNN	Rede neural de alimentação para frente;
RNN	Rede neural recorrente;
RC	Computação de reservatório;
ESN	Redes de estado de eco;
LSM	Máquinas de estado líquido;
PRC	Computação de reservatório físico;
GA	Algoritmo genético;
fMRI	Imagem por ressonância magnética funcional;
k-NN	k-vizinhos mais próximos;
SVM	Máquina de vetores de suporte;
EDO	Equação diferencial ordinária;
LE	Expoentes de Lyapunov.

SUMÁRIO

1. Introdução

Temas envolvendo Inteligência Artificial (IA) têm sido abordados há anos em diversos setores de nossa sociedade. O avanço tecnológico proporcionado por essa, assim como o número crescente de aplicações, tem contribuído significativamente para o seu desenvolvimento em diversas diversas sub-áreas. Dentre essas destacam-se Visão Computacional [1], Processamento de Linguagem Natural [2], e Aprendizado de Máquina [3] (ML, do inglês: *Machine Learning*).

Aprendizado de máquina é um subcampo da IA cujo estudo aborda a construção de algoritmos capazes de construir modelos matemáticos, a partir de dados de entrada, sem serem explicitamente programados[1]. Isso é, que sejam capazes de fazer predições ou tomar decisões, e melhorar seu desempenho mediante repetição, sem a necessidade de o programador fornecer instruções explícitas em todas as etapas da computação.

"As máquinas são capazes de fazer o que nós, como entidades pensantes, podemos fazer?" [5]. Alan Turing propôs essa pergunta como uma alternativa à questão se as máquinas poderiam pensar, conduzindo a discussão relativa às tarefas realizadas por uma máquina a uma tendência mais operacional e menos cognitiva. Em concordância com essa abordagem, Tom Michael Mitchel, cientista da computação conhecido na área, definiu formalmente o Aprendizado de Máquina da seguinte maneira: "Diz-se que um programa de computador aprende pela experiência E, com respeito a algum tipo de tarefa T e performance P, se sua performance P na tarefa T, na forma medida por P, melhorar com a experiência E." [6].

Com o decorrer dos anos e a popularização da IA, foram levantadas diversas questões. Dentre essas, pode-se mencionar: qual a melhor abordagem a se seguir, quais os limites cognitivos e práticos que as máquinas podem atingir, e quais os malefícios que esse desenvolvimento pode acarretar. Para lidar com essas questões, especialistas e nomes conhecidos no setor tecnológico sugerem que a melhor solução será aprimorar o conhecimento em relação aos processos que levam os programas de IA às decisões e ações, em vez de atrasar esse avanço. Dessa maneira, nosso objetivo é ampliar a compreensão desses processos na realização de algumas tarefas, em detrimento a apenas buscar atingir o melhor desempenho possível.

[1] O termo "explicitamente programados"é comumente associado a Arthur Samuel. Apesar de não haver uma fonte específica do cientista da computação usando esse termo, essa associação pode ter ocorrido devido a uma interpretação do seu famoso artigo de 1959: *Some Studies in Machine Learning Using the Game of Checkers* [4], no qual o autor escreveu frases como: "Programar computadores para aprender com a experiência deverá por eliminar, em grande parte, a necessidade desse esforço de programação detalhada".

Dentre as mais amplamente utilizadas e eficientes técnicas de processamento de informação, dentro do campo do ML, destacam-se as redes neurais artificiais (ANN, do inglês: *Artificial Neural Networks*). Essas consistem em um conjunto de unidades de processamento de informações, modelos matemáticos que fazem analogia aos neurônios biológicos reais. Assim como os seus análogos os neurônios artificiais se conectam por ligações sinápticas, recebem estímulos através dos seus nós de entrada, e como resposta produzem uma ou mais saídas, sendo essa conversão realizada mediante funções de ativação determinadas de acordo com a arquitetura de cada rede[7, 8]. Há duas principais maneiras de se conectar esses neurônios: a rede neural de alimentação para frente (FNN, do inglês: *Feed-forward Neural Network*) e a rede neural recorrente (RNN, do inglês: *Recurrent Neural Network*), das quais o nosso modelo é um exemplo da segunda.

A principal característica das RNNs que as distinguem das FNNs é a existência de ciclos na topologia de suas conexões. Isso é, as conexões nas FNNs são realizadas sempre da camada anterior para a posterior, enquanto nas RNNs não há essa restrição, pois os estados das unidades podem depender até mesmo de seus próprios valores em tempos posteriores. Devido a essa característica, as RNNs apresentam uma dinâmica temporal autossustentada nas suas conexões mesmo sem a presença de um sinal de entrada externo, caracterizando-a como um sistema dinâmico. Caso haja um sinal de entrada, uma RNN preserva em seu estado interno uma transformação não linear do histórico desse, tornando-se capaz de processar informações que possuam dependência temporal [9, 10].

Uma clara desvantagem das RNNs é o alto custo do seu treinamento devido às suas conexões cíclicas, pois cada vez que os pesos das conexões internas são atualizados, altera-se a estrutura da rede. Uma nova abordagem, denominada computação de reservatório (RC, do inglês: *Reservoir Computing*) [11], surgiu para simplificar essa dificuldade. Inicialmente representado pelos seus dois principais modelos, redes de estado de eco (ESN, do inglês: *Echo State Networks*) [12] e máquinas de estado líquido (LSM, do inglês: *Liquid State Machines*) [13], os RCs tratam-se de estruturas semelhantes às RNNs convencionais, porém com a vantagem de serem aprendidas somente as conexões entre a saída e as unidades neuronais. Desta maneira, se mantêm fixas tanto as conexões internas entre os neurônios quanto as do sinal de entrada com essas. Além de facilitar o treinamento, esse aspecto fixo da rede torna viável a implementação de reservatórios compostos por sistemas físicos reais, estrutura conhecida como computação de reservatório físico (PRC, do inglês: *Physical Reservoir Computing*).

Sistemas físicos de diferentes naturezas têm se mostrado eficientes como PRCs, tais quais mecânicos [14, 15], biológicos [16, 17], quânticos [18, 19] e circuitos analógicos [20, 21]. Pode-se obter os estados neuronais diretamente dos sistemas físicos, construídos a partir de uma abordagem experimental, ou mediante simulações computacionais.

Neste trabalho, utilizamos uma abordagem numérica, na qual o reservatório é

composto por osciladores de Kuramoto [22]. Esses osciladores são amplamente utilizados para investigar os efeitos da sincronização em redes neurais biológicas [23]. Além disso, a simplicidade do modelo matemático dos osciladores de Kuramoto favorece o estudo dos efeitos da dinâmica do reservatório sobre o desempenho computacional do PRC. Com o intuito de aumentar a variabilidade das fases, incluímos um termo de campo médio ao modelo originalmente proposto.

Para escolhermos quais valores de parâmetros usaríamos, utilizamos a técnica metaheurística conhecida como algoritmo genético (GA, do inglês: *Genetic Algorithm*) [24, 25, 26]. Essa costuma ser escolhida quando há uma grande quantidade de parâmetros envolvidos e/ou limitações computacionais para a realização de determinada tarefa.

GA é uma técnica de escolha de parâmetros otimizada, no contexto da ciência da computação, a qual busca um conjunto de valores que levem o programa a obter um desempenho suficientemente bom, sem percorrer todos os valores possíveis para cada parâmetro [27]. Seu nome é dado devido ao fato de ser baseada no processo biológico de seleção natural proposto por Charles Darwin no século XIX.

Investigamos a dinâmica do reservatório, composto por osciladores de Kuramoto, na emulação e predição de séries binárias regidas pela dinâmica de um mapa logístico, séries binárias geradas aleatoriamente a cada passo, séries que seguem uma distribuição gaussiana, e séries trigonométricas.

Como mencionado anteriormente, nosso objetivo principal é estudar os processos intermediários adotados pela máquina na realização das tarefas. Nesta tese, escolhemos investigar os efeitos da dinâmica do reservatório no desempenho do PRC. Outros trabalhos tiveram sucesso em estudar algumas características que levam o PRC a apresentar bons desempenhos [28, 29]. Nesses estudos, normalmente, utiliza-se sinais de entrada, os quais são informações enviadas ao reservatório para serem processadas lá, e as séries alvo, isso é, as séries de referência a serem emuladas e aprendidas, costumam ser semelhantes aos sinais de entrada. No nosso trabalho, entretanto, propusemos diversas séries de referência sem a presença de nenhum sinal de entrada, utilizando o reservatório com a mesma configuração (exceto pela alteração nos parâmetros) para a aprendizagem dessas.

Portanto, a dinâmica do reservatório é o processo intermediário que escolhemos investigar para a realização da tarefa por parte do modelo, a qual no nosso estudo é a aprendizagem de séries de referência geradas por diferentes funções.

Na introdução apresentamos os conteúdos a serem abordados, nosso objetivo e a organização do texto. No segundo capítulo seguimos com uma revisão histórica acerca da evolução dos computadores até as versões atuais, os quais são as principais ferramentas, tanto para o desenvolvimento da IA e do aprendizado de máquina quanto para realizar computações na área de sistemas dinâmicos. Comentamos também a influência dessas

máquinas na sociedade e em eventos históricos, e apresentamos motivações que nos levaram ao interesse em compreender os processos, adotados pelos computadores, para a realização de algumas tarefas de aprendizado de máquina, o principal objetivo do nosso trabalho. No terceiro capítulo aprofundamos a discussão sobre a evolução das interpretações e conceitos envolvidos na área de IA, comentando sua crescente aplicabilidade em tempos recentes, assim como as vantagens e desvantagens que esse fenômeno pode oferecer. Apresentamos os subcampos da IA, em especial ML, com foco em redes neurais e RC. Ao fim desse capítulo, discorremos sobre sistemas dinâmicos e mostramos como podemos caracterizar seus comportamentos. O quarto capítulo contém a metodologia que adotamos para obter os resultados, assim como detalhes sobre a arquitetura do nosso modelo de PRC. No quinto capítulo, apresentamos e discutimos os resultados obtidos com a pesquisa desenvolvida ao longo do doutorado. Tínhamos como hipótese que dependendo de como variássemos os parâmetros do modelo, alteraríamos também sua dinâmica, e por consequência obteríamos desempenhos distintos do PRC, seja para pior ou para melhor. Além disso, observamos ainda que os valores dos parâmetros que favoreceram o aprendizado, isso é, a dinâmica do reservatório que torna o PRC mais eficiente, variam de acordo com o tipo de função que gera a série de referência a ser aprendida. O sexto capítulo compreende a conclusão do trabalho, baseada nos resultados, e indicações de possíveis caminhos a se seguir a partir desses. Na seção de apêndices, apresentamos um pseudocódigo contendo as etapas principais do GA.

Todos os programas utilizados ao decorrer do estudo desta tese foram desenvolvidos em Fortran 95 pelo autor, em parceria com o Dr. Eduardo L. Brugnago, ex-aluno de doutorado do Programa de Pós-Graduação em Física da Universidade Federal do Paraná, compilados com o gFortran, e todas as figuras geradas foram utilizando o Gnuplot, na versão 5.2.

2. História e motivação

Atualmente, entende-se por computador um conjunto de componentes eletrônicos capaz de realizar diversas tarefas, como sequências de operações lógicas e aritméticas, de acordo com o modo em que esse for programado. Entretanto, inicialmente o termo era atribuído a qualquer mecanismo que auxiliasse na tarefa de fazer cálculos. Dentre esses, pode-se citar as tábuas de argila do antigo oriente médio, e o ábaco, usado em diversas culturas através do tempo, inclusive nos dias atuais, principalmente em alguns países orientais [30].

Os computadores digitais, amplamente utilizados na atualidade, armazenam dados exclusivamente sob a forma de números, os quais são expressos no sistema binário. Por outro lado, os seus antecessores analógicos tratavam as grandezas envolvidas no problema a ser resolvido mediante fenômenos físicos reais, como elétricos, mecânicos ou hidráulicos, cujas equações que os regiam eram análogas às que determinavam o comportamento dos elementos do problema em questão.

A Máquina de Anticítera é considerada o primeiro computador analógico, criada no segundo século antes de Cristo, e utilizada para prever posições astronômicas determinantes para as fases da lua, as atividades de agricultura e os festivais religiosos [31]. No início do Século XVII, o matemático e clérigo inglês Willian Oughtred desenvolveu a Régua de Cálculo, na qual eram possíveis realizar operações matemáticas com base nas propriedades da função logarítmica, e que foi utilizada em algumas escolas e ambientes profissionais até a invenção das calculadoras manuais e portáteis que tornaram as réguas de cálculo obsoletas [32]. Nesse mesmo período foi criado por Blaise Pascal uma máquina denominada com o seu sobrenome, a qual era composta por rodas metálicas e realizava operações básicas, e foi inspiração para calculadoras inventadas posteriormente [33]. Ainda no contexto de computadores analógicos, é notável a figura de Charles Babbage. Considerado um cientista além do seu tempo, Babbage foi um matemático que viveu durante a Revolução Industrial e tinha o objetivo de automatizar os mecanismos de cálculos. Apesar de na época não ter sido possível construir suas invenções, seus projetos eram considerados ambiciosos. A Máquina Diferencial, por exemplo, seria capaz de calcular uma série de valores numéricos e imprimir os resultados de maneira automática, enquanto a Máquina Analítica funcionaria com base nas instruções de cartões perfurados, seria movida a vapor, e possuiria mecanismos de processamento e memória separados, característica que remete aos computadores modernos [34].

A revolução do computador é considerada a fase na qual foram produzidos os

primeiros computadores digitais, e um dos responsáveis por isso foi um nome bem conhecido na área, Alan Mathison Turing. Em 1936, o matemático inglês dedicou-se a formalizar matematicamente os passos que o ser humano procede na execução de um determinado cálculo, levando-o à construção formal da noção de algoritmo, a qual serviu como base para todo o desenvolvimento da computação. Além disso, Turing foi um dos principais responsáveis pelo Colossus, a máquina construída durante a Segunda Guerra Mundial (SGM), capaz de decifrar os códigos da Enigma, interceptando diversas vezes a comunicação alemã e contribuindo, assim, para o a vitória dos aliados. O Colossus era inteiramente eletrônico, com 1500 válvulas, e sua arquitetura era superior às das máquinas construídas até então [35].

De 1936 em diante foram criadas algumas máquinas que merecem destaque. Apesar do desinteresse do governo alemão em sua primeira criação, o Z1, por aparentemente não ser útil na já eminente SGM, o engenheiro Konrad Zuse é conhecido como um pioneiro na área da computação. O principal dispositivo responsável por esse reconhecimento é o Z3, de 1941, o primeiro computador totalmente automático e programável [36]. Nos Laboratórios de Harvard, em 1943, Howard Aiken e outros engenheiros da IBM desenvolveram o primeiro dentre os computadores automáticos de larga escala, conhecido como Harvard Mark I, dando início à era dos computadores digitais nos Estados Unidos [35, 37]. Esse período teve continuidade com as máquinas sucessoras ENIAC, EDIVAC, EDSAC E UNIVAC, as quais apresentavam melhorias significativas na memória e no desempenho quando comparadas ao Harvard Mark I. Essas máquinas foram desenvolvidas na Universidade da Pensilvânia e tiveram a arquitetura modelada pelo matemático húngaro John von Neumann, a qual é utilizada como referência até os dias atuais [37].

A segunda geração de computadores foi caracterizada pela invenção dos transistores, substitutos das válvulas a vácuo amplamente utilizadas nas máquinas antecessoras. Além de muito menores, os transistores não exigiam tempo de pré-aquecimento, consumiam menos energia e eram mais rápidos. A terceira geração foi marcada pela utilização dos circuitos integrados, também conhecidos como microchips, os quais eram construídos integrando um grande número de transistores. Além de possibilitar a redução dos equipamentos, os circuitos integrados tinham um processo de fabricação que viabilizava a produção de várias unidades simultaneamente, e por consequência, a produção em massa. Os processadores foram o diferencial das máquinas da quarta geração, o quais se tornaram mais rápidos e com maior capacidade de armazenamento, e pela possibilidade da redução de tamanho, nesse período, se tornou popular a venda de computadores pessoais. A quinta geração, que começou em 1991 e se estende até os dias atuais, continua a apresentar inovações e melhorias em todos os componentes, de tal forma que possibilite a conectividade do computador a outros dispositivos como o celular, a televisão ou sistemas de segurança, e também o avanço da IA, a qual vem demonstrando ter vasta aplicabilidade nos meios científico e tecnológico [35, 38].

Além da computação, evidentemente, outras áreas do conhecimento contribuíram para o desenvolvimento da IA. Desde a antiguidade, filósofos como Aristóteles se dedicam a entender de onde vem o conhecimento e como este conduz à ação [8].

Contemporâneo às máquinas analíticas do Século XVII, o matemático e filósofo inglês, Thomas Hobbes propôs que o raciocínio era semelhante à computação numérica e sugeriu a ideia de um "animal artificial", fazendo comparações de órgãos humanos com dispositivos mecânicos. Além disso, outros nomes conhecidos, como René Descartes, Francis Bacon e John Locke, se dedicaram a entender melhor a relação entre a mente e a parte física do cérebro, e o processo de compreensão de novos conceitos [8].

Os matemáticos contribuíram para a formalização da ciência, definindo a base para a compreensão da computação e do raciocínio sobre algoritmos. No campo da lógica, nomes como George Boole e Gottlob Frege se destacaram com suas teorias, desenvolvidas no século XVIII, sendo utilizadas até os dias atuais. Na teoria da probabilidade, uma contribuição importante foi a de Thomas Bayes, cuja regra que leva seu sobrenome possibilitou a atualização das probabilidades à luz de novas evidências, formando a base de parte das abordagens para o raciocínio incerto em sistemas de IA. Ainda na matemática, as noções de computabilidade e tratabilidade ajudam a identificar se determinada função, ou problema, podem apresentar dificuldade nas suas resoluções devido às características próprias desses ou à limitação na tecnologia do computador que está realizando essas tarefas [8].

A economia, comumente associada ao modo de se lidar com o dinheiro, na verdade aborda de maneira mais profunda o processo de tomada de decisões, tendo em vista a otimização do resultado esperado. Nessa área, vale destacar a teoria dos jogos idealizada por Von Neumann e Oskar Morgenstern, a qual demonstrou benefícios de políticas aleatórias ao tomar determinadas decisões. Outro trabalho que merece destaque é o que premiou com o Nobel de economia de 1978 um dos pioneiros em IA, Herbert Simon, valorizando a satisfação em detrimento a uma decisão que busque um resultado ótimo, fornecendo assim uma descrição mais realística do comportamento humano [8].

Desde a identificação de que o cérebro era responsável pelos pensamentos, nos tempos antigos, devido à observação das consequências de pancadas fortes na cabeça, até os detalhados mapeamentos de atividade cerebral com processamento de imagem por ressonância magnética funcional (fMRI, do inglês: *functional Magnetic Resonance Imaging*) [39, 40], a neurociência estuda o sistema nervoso, em especial o cérebro, sendo cada vez mais capaz de atribuir às suas regiões a função específica dentro do organismo estudado. Em um estudo envolvendo IA, pode-se escolher abordagens distintas. Dentre essas, uma que seja centrada nos seres humanos, tentando imitá-los ou distanciando desses e buscando um modelo ideal. Ou ainda, tendo como objetivo o desenvolvimento de processos de pensamento e raciocínio, ou a tentativa reproduzir um comportamento desejado. Portanto, o papel da neurociência é fundamental na compreensão do funcionamento da mente e

quais as suas semelhanças e diferenças quando comparada aos computadores [8].

A psicologia científica, que teve origem no século XVII, ganhou notoriedade no século seguinte devido ao movimento chamado Behaviorismo. Apesar de conseguir obter medidas objetivas das ações dos animais em resposta aos estímulos, esse método apresentava limitações quando aplicado a humanos, pois esses demonstravam realizar processos introspectivos além da relação simples entre estímulo e resposta. A psicologia cognitiva propôs teorias que levassem em consideração esses processos. Em 1943, o filósofo e psicólogo escocês Kenneth Craik dividiu em três passos o caminho que leva um indivíduo à ação. Inicialmente, um estímulo externo deve ser traduzido em uma representação interna, a qual será manipulada por processos cognitivos para ser transformada em novas representações internas, de acordo com a visão da realidade adquirida mediante as próprias experiências, e aquelas serão, por fim, traduzidas em ações. Alguns anos após essa proposta, na segunda geração de computadores, foi criado o campo da ciência cognitiva, o qual começou a fazer analogias entre seres humanos, e animais, com máquinas, e a partir disso modelar fenômenos psicológicos como processamento de informações.

Na linguística, ocorreu algo semelhante com a crítica de Noam Chomsky à teoria de Burrhus Skinner, questionando a incapacidade do Behaviorismo em explicar o porquê de uma criança ter a capacidade de compreender e formar frases que ainda não tivesse ouvido, por exemplo. Abordando a IA, a linguística computacional parecia simples no início, porém logo se percebeu que para se compreender uma outra língua não bastava conhecer as estruturas das frases. A partir disso, esse campo passou a considerar outros fatores, como o conteúdo e o contexto, e continua a se desenvolver atualmente. Outro campo que se afastou da teoria behaviorista foi a teoria de controle, que considerou o comportamento consciente como o resultado de um mecanismo regulador, que busca minimizar a diferença entre os estados atual e objetivo. Apesar de haver exemplos de máquinas auto controladas desde a antiguidade, é a moderna teoria do controle que declara buscar a maximização da função objetivo, o que remete a uma parte dos estudos envolvendo IA. Todavia, ambas as áreas se diferenciam devido à segunda considerar alguns problemas como linguagem, visão e planejamento [8, 41].

O primeiro trabalho reconhecido como de IA, posteriormente, foi o modelo de neurônios artificiais proposto em 1943 por Warren McCulloch e Walter Pitts, o qual era um caso particular de um discriminador linear de n entradas e uma saída, pois nesse, além da saída, os componentes do vetor de entrada também eram binários. Os autores demonstraram que com um número suficiente desses neurônios, e com conexões sinápticas ajustadas apropriadamente, uma rede assim seria possível realizar o cálculo de qualquer função computável. Apesar de simples, esse trabalho foi e continua sendo uma referência na área, tendo influenciado outros personagens importantes, como os já citados Alan Turing e von Neumann. Donald Hebb propôs, em 1949, que a conectividade do cérebro é

continuamente modificada conforme um organismo aprende novas tarefas funcionais, e demonstrou uma regra de atualização simples para modificar a intensidade de conexão entre neurônios. Em 1950, dois alunos de Harvard, Marvin Minsky e Dean Edmonds, criaram o primeiro computador, constituído por redes neurais artificiais e capaz de aprender alguns conceitos simples [42]. Ainda nesse mesmo ano, Turing abordou temas como algoritmos genéticos, aprendizagem por reforço, aprendizagem de máquina, e apresentou o teste que recebeu o seu nome, o qual buscava responder se uma determinada máquina tem a capacidade de exibir um comportamento inteligente semelhante a um ser humano. Apesar desses trabalhos, o ano mais aceito como a origem da IA é 1956 [5, 8, 43, 7].

Um grupo de pesquisadores de instituições como Princeton, Stanford, IBM e MIT, interessados em teoria dos autômatos, redes neurais e estudo da inteligência, liderado pelo cientista da computação estadunidense John McCarthy, se reuniu durante dois meses no verão de 1956 para um seminário. A proposta era tentar desenvolver máquinas que usem a linguagem, a partir de abstrações e conceitos, para resolver tipos de problemas até então reservados aos seres humanos, e se possível, de maneira mais rápida e mais eficiente. Dentre os pesquisadores envolvidos, dois se destacaram. Allen Newell e Herbert Simon apresentaram um programa de raciocínio denominado Logic Theorist, o qual era capaz de demonstrar teoremas matemáticos, por exemplo. Apesar de o seminário não ter trazido nenhuma grande novidade, ocorreu um importante intercâmbio científico entre aqueles que seriam os principais responsáveis pelos avanços na área, pelo menos, durante os 20 anos seguintes. Além disso, a própria proposta do seminário apontava a necessidade que havia de tornar a IA um campo separado de outros como matemática, teoria do controle, pesquisa operacional ou teoria da decisão. Além da metodologia, que está intimamente relacionada à ciência da computação na tentativa de construir máquinas autônomas e que funcionam em ambientes complexos e mutáveis, a IA difere dos outros campos ao apresentar a intenção de reproduzir faculdades humanas, como a criatividade e a linguagem [8].

Em sequência ao Logic Theorist, Newell e Simon criaram, em 1957, o GPS (solucionador de problemas gerais), programa capaz de resolver mais problemas que seu antecessor e de maneira mais semelhante a um ser humano. Em 1958, Frank Rosenblatt, psicólogo e pesquisador na Universidade de Cornell, prosseguiu com as ideias de McCulloch e Pitts lançadas aproximadamente 15 anos antes. Rosenblatt desenvolveu uma nova abordagem para o problema de reconhecimento de padrões, ampliando a aplicação dos discriminadores lineares, propondo assim o Perceptron, e ainda o Perceptron de múltiplas camadas, o qual é capaz de reconhecer também padrões que não fossem linearmente separáveis. Esses e outros trabalhos na área foram tomando o lugar da incredulidade, até então existente, principalmente provinda da classe intelectual, a qual fazia questão de levantar tarefas que supostamente não poderiam ser realizadas por máquinas, afirmações que foram sendo derrubadas pelos pesquisadores em IA [8, 43, 7].

Apesar do avanço considerável da IA durante esses anos citados, os prognósticos otimistas de seus pesquisadores pioneiros não foram confirmados, pelo menos não no tempo que esperavam. Hebert Simon em 1957, por exemplo, tentou responder ao questionamento de Turing afirmando que, em um futuro breve, a variedade de problemas que as máquinas poderiam lidar seria correspondente à variedade de problemas com os quais a mente humana é capaz, provavelmente motivado pelo sucesso dos primeiros sistemas de IA ao resolver problemas simples. Ainda, Simon tentou fazer previsões mais específicas como a de que, em 10 anos, um computador seria campeão de xadrez, ou que um teorema matemático significativo seria provado por uma máquina. Entretanto, passaram-se pelo menos 40 anos para que essas se concretizassem. A maioria dos primeiros programas trabalhava experimentando diferentes combinações de passos até encontrar a solução desejada, mas conforme se aumentava a complexidade dos problemas, esses programas se mostravam ineficientes. Acreditava-se que o motivo era a limitação tecnológica, porém problemas não tão complexos que seriam, em teoria, possíveis de serem tratados com o que se dispunha na época em um número maior de etapas, não foram resolvidos. Além disso, os programas não tinham conhecimento do assunto que abordavam. Por exemplo a tradução, que não basta a substituição das palavras, e sim uma interpretação do contexto da frase na língua original e uma representação coerente na língua a ser expressa. De qualquer modo, a baixa disponibilidade de computadores pessoais e estações de trabalho também foi uma dificuldade para o desenvolvimento dos projetos na área [8, 7].

Outro fator que contribuiu para o atraso nos avanços em IA foi o desinteresse por grande parte dos pesquisadores e investidores envolvidos, principalmente provocado pelo livro escrito por Minsky e Papert em 1969 [44], o qual apontou limitações quanto aos cálculos que tanto o percéptron de camada única quanto o de múltiplas camadas eram capazes de realizar. Apesar de pouco tempo após essa publicação já existirem soluções para grande parte das limitações indicadas, cientistas e investidores abandonaram as pesquisas na área por pelo menos dez anos seguintes, restando apenas alguns poucos, principalmente aqueles que trabalhavam com psicologia e neurociência. Curiosamente, os primeiros algoritmos de aprendizado por retro propagação para redes de várias camadas (técnica abordada em mais detalhes na seção 3.1), os quais foram importante no ressurgimento das redes neurais na década de 80, também já haviam sido desenvolvidos em torno do ano da publicação de Minsky e Papert [8, 7, 44].

Na década de oitenta, a IA voltou a despertar interesse em diversos setores. No ramo da indústria, o primeiro sistema especialista[1] de sucesso comercial foi o R1, criado em 1982 pela *Digital Equipment Corporation*. Logo, o R1 ajudou a configurar pedidos de novos computadores e fez a empresa economizar cerca de quarenta milhões de dólares por ano. Essa prática se tornou popular, e dentro de alguns anos todas as grandes empresas

[1] Programa computacional capaz de realizar tarefas de uma especialista humano em determinado domínio.

dos Estados Unidos passaram a utilizar e pesquisar sobre sistemas especialistas. Nesse período, os Estados Unidos, o Japão e a Inglaterra investiram grandes quantias, levando a indústria da Inteligência Artificial a movimentar bilhões de dólares em 1988, construindo e comercializando sistemas especialistas, sistemas de visão, robôs, e software e hardware especializados para esses projetos. Apesar disso, os projetos foram muito ambiciosos e as empresas que não cumpriram o que prometeram, acabaram perdendo investimento e não conseguiram levá-los adiante [8].

Como mencionado, o algoritmo de aprendizado por retro propagação foi descoberto em 1969. Porém, só nos anos oitenta foi amplamente utilizado, principalmente em problemas de aprendizado em ciência da computação e psicologia. Nesse período, os estudos envolvendo redes neurais se dividiram em dois campos, um que focava na criação e compreensão de algoritmos que buscavam realizar cálculos específicos, tal como suas propriedades matemáticas, e outro que tentava modelar as propriedades de neurônios reais. A partir dessa década, os pesquisadores de IA adotaram uma metodologia mais científica e menos intuitiva, preferindo utilizar, com maior frequência, teorias já existentes e difundidas a propostas completamente novas, e buscando maior aplicabilidade aos projetos, os levando a serem utilizados também em indústrias. Passou-se a analisar com mais cuidado quais tarefas as redes neurais eram capazes de realizar, e quais as suas vantagens em relação às outras técnicas, levando à consolidação do seu uso, e os sistemas especialistas a se comportarem de uma maneira mais condizente com a Teoria da Decisão e menos imitando os passos de especialistas humanos [8, 45].

Com a popularização do uso da internet nos anos 90 também se tornaram comuns as aplicações na web de sistemas de IA, as quais podem agir em mecanismos de pesquisa para coletar informações, sistemas de recomendação de produtos e na construção de sites, por exemplo. A ênfase dos pesquisadores de IA era nos detalhes da construção do algoritmo. Entretanto, recentemente, sugere-se que a quantidade dados disponíveis seja mais determinante no sucesso do programa, o que se tornou evidente devido à vasta disponibilidade de informações que se encontram na internet [8, 46, 47, 48].

Nos dias atuais, há aplicações para IA em diversas áreas. Um exemplo são veículos robóticos autônomos equipados com câmeras, radares e telêmetros a laser, capazes de decidir o quanto acelerar, quando frear e qual o trajeto a se adotar, evitando colisões com objetos, pedestres e outros veículos [49, 50]. Acima da superfície terrestre, há exemplos de programas que controlam naves espaciais [51] e dispositivos de exploração de superfície em Marte [52], os quais conseguem realizar planejamentos e efetuar detecção, diagnóstico e resoluções de problemas durante as missões.

Nos jogos, o caso mais conhecido é o do *DEEP BLUE* da IBM, que em 1996 se tornou o primeiro programa de computador a vencer o campeão mundial de xadrez, o russo Garry Kasparov, em uma partida de exibição [53].

Um problema que a popularização da internet trouxe foram os spam, mensagens eletrônicas enviadas em massa, podendo ser propagandas de produtos ou ofertas de serviços, por exemplo, reconhecidas automaticamente pela IA e movida a um destino separado, o qual não teria a mesma eficiência se fosse um programa com uma abordagem convencional, devido à constante atualização das táticas por parte dos remetentes para driblar essa inspeção [54].

No planejamento logístico vale mencionar que, em 1991, as forças armadas dos Estados Unidos realizaram essa tarefa utilizando IA, e os planos que antes demoravam semanas, devido ao elevado número de veículos e rotas possíveis, passaram a ser concluídos em algumas horas [55]. A tradução automática, que custou a se desenvolver no início dos estudos em IA, agora conta com o amplo banco de dados disponível na internet contendo uma vastidão de palavras, de diversas línguas, faladas ao redor do globo terrestre [56, 57].

Na Medicina, a IA vem demonstrando muitos benefícios devido à essa grande variedade de informações compartilhadas na internet, dentre os quais vale citar a escolha mais eficiente dos tratamentos, para determinadas doenças, e maior precisão nos diagnósticos, seja por imagem, comparando exames radiológicos ou imagens de doenças dermatológicas, por exemplo, seja no reconhecimento e associação de sintomas, dos quais podem ser retirados da literatura conhecida na área médica, a dados dos pacientes que um ser humano teria muita dificuldade em considerar, como etnia ou locais de residência e de trabalho. Além disso, há projeção de que aparelhos como os relógios inteligentes possam melhorar a relação médico-paciente, gerando notificações, nos casos mais graves, ou armazenando os dados de saúde para análise posterior, nos casos mais leves [58].

Observa-se que a robótica, a qual dificilmente se dissocia da Inteligência Artificial, além de despertar a curiosidade por seus feitos, sejam esses imitando seres humanos ou não, tem se tornado cada vez mais presente na indústria, exercendo principalmente funções que exijam esforço repetitivo, que sejam de risco, ou que envolvam planejamento e logística, por exemplo [59].

Há perspectivas de aplicabilidade da IA em outras diversas áreas, como Economia e Direito [60], as quais a exemplo das citadas enfrenta o problema da desconfiança, porém este será um tópico abordado posteriormente.

Apesar do estudo em IA ter se desenvolvido muito devido ao interesse crescente nesses campos, os pioneiros da área demonstraram descontentamento com os rumos que foram seguidos. Nomes como John McCarthy, Marvin Minsky, dentre outros, alegaram que a ênfase das pesquisas em IA deveria ser no processo de aprendizado e de criatividade da máquina, e não a eficiência na realização de tarefas específicas. Esses têm organizado congressos e publicado estudos, nos últimos anos, na tentativa de promover e reforçar seus argumentos. Todavia, observa-se que essa abordagem mais aplicada tende a continuar despertando interesse em diversas áreas [8].

A abordagem que os pioneiros criticam é a conhecida atualmente como Inteligência Artificial Fraca. Mais operacional, essa trata a hipótese de que as máquinas possam agir de maneira inteligente, portanto, preocupa-se mais em discutir suas capacidades e as limitações de sua atuação. A maioria dos pesquisadores abordam essa hipótese. Turing por exemplo, em seu teste, propõe que um programa deva ser capaz de manter uma conversação por cinco minutos. Para esse passar no teste, o interrogador deve permanecer por, pelo menos, trinta por cento do tempo sem conseguir identificar se está conversando com uma máquina ou um ser humano, demonstrando que o objeto de interesse é o comportamento da máquina e não o ato de raciocinar. Aqueles que adotam essa abordagem se dedicam a tratar da realização de tarefas por parte das máquinas e discutir suas limitações, sem se preocupar com os processos que essas executaram até tomarem as decisões ou agirem [8].

A Inteligencia Artificial Forte, por sua vez, apresenta características mais cognitivas, isso é, aborda questões relativas às suas capacidades de raciocínio. Um questionamento, por exemplo, seria se uma máquina que passa pelo teste de Turing está mesmo pensando ou realizando uma simulação de pensamento. Os pesquisadores que se propõem a responder perguntas dessa natureza abordam temas como a consciência, ou seja, se a máquina está ciente dos próprios estados mentais e de suas ações, a fenomenologia, ligada à necessidade de haver emoção envolvida, a intencionalidade, relacionada às representações da máquina e o mundo real, e as semelhanças e diferenças entre as mentes humana e construída artificialmente, levando a discussões filosóficas em torno da interpretação da relação mente e corpo, se dualista ou monista, por exemplo. Há cientistas que não consideram necessária a análise minuciosa dos estados mentais internos das máquinas, pois essa não é realizada nos seres humanos, em geral. Todavia, há questões éticas a serem discutidas que podem servir para contestar essa linha de pensamento [8].

Uma consequência negativa do avanço tecnológico é de pessoas perderem seus empregos para a automação, porém esse não é um problema novo. A indústria tem se tornado dependente dos computadores e dos programas de IA há algum tempo e há funções nas quais esses programas, de fato, substituem uma quantidade considerável de trabalhadores humanos. Entretanto, em vez de tentar frear o avanço tecnológico, uma alternativa é aprender a lidar com esse e investir na qualificação profissional. Dessa maneira, além de criar novas vagas, os empregos seriam mais abrangentes e melhor remunerados.

Outra consequência indesejada poderia ser a perda de responsabilidade em decisões equivocadas. Atualmente, os sistemas especialistas na medicina podem recomendar um tratamento, por exemplo, porém é o médico quem decide se vai adotar a sugestão. Conforme a confiança nas decisões das máquinas aumentar, e se for constatado que seus diagnósticos forem mais precisos que os humanos, há uma questão ética em torno de quem deve ser a decisão final. Entretanto, os médicos humanos costumam se basear na literatura conhecida e relacionada ao problema do paciente, já um programa de IA pode adotar outros caminhos

para tomar sua decisão que sejam menos convencionais, e que ainda assim não convençam os médicos. Mesmo na situação hipotética de comprovada a maior taxa de acerto para a máquina, em todos os tratamentos médicos há um percentual de sucesso envolvido. Em caso de piora no quadro de um paciente que recebeu um tratamento escolhido por um robô, ou até mesmo morte, é difícil saber quem culpar, se é que há um culpado. Há autores estudando a transparência do processo de decisão e a eficiência desses programas [61, 62, 63, 64, 65]. De maneira semelhante, não é uma questão simples julgar quem é o responsável caso um carro autônomo infrinja as leis de trânsito vigentes, ou mesmo dentro da lei, se envolva em algum acidente [66], além de outros processos judiciais envolvendo programas de IA [67].

Outro aspecto que gera desconfiança é o uso intencional de sistemas de IA para fins indesejáveis. Tem-se observado um aumento no uso aeronaves e carros autônomos em guerras [68], assim como drones [69]. Apesar de colocar menos militares humanos em campos de batalha, preservando assim suas vidas, essa forma de combate pode encorajar os governantes a entrarem em guerras de forma mais imprudente, aumentando assim o número de conflitos, e por consequência os riscos às vidas de mais civis. Além disso, há a promessa do aprimoramento da segurança coletiva, que cumprindo ou não o seu propósito, abre espaço para a divulgação de informações pessoais, levando ao comprometimento da privacidade do indivíduo [70].

Os casos mencionados envolvem consequências adversas não intencionais dos avanços em IA, ou cuja intenção é humana. Contudo, para a discussão referente ao nosso estudo, é mais interessante abordar os riscos relacionados ao arbítrio da decisão da própria máquina em suas ações.

É comum encontrar exemplos no cinema e na literatura que abordem questões éticas e conflitos entre robôs e a raça humana, como nos filmes *O Exterminador do Futuro* (1984), *Matrix* (1999) e *A.I. Inteligência Artificial* (2001). Dentre as obras desse gênero, a mais influente é o livro *Eu, Robô* (1950), de Isaac Asimov, no qual um jornalista entrevista a psicóloga de robôs, Dra. Susan Calvin, que fala sobre os 9 contos que experienciou durante sua carreira na empresa construtora de robôs *U.S. Robots and Mechanical Men*.

Um motivo de haver um interesse da cultura no tema de IA, é o desconhecimento de como a humanidade lidaria com o aumento do número de robôs na nossa sociedade, assim como na quantidade de funções delegadas a esses e seus comportamentos a partir do momento que tivessem autonomia. Apesar de ficção, muitos dos eventos abordados em filmes e livros são discutidos como possíveis, pelo menos de modo semelhante, em uma realidade futura. Um sistema de defesa autônomo, por exemplo, que não possua uma boa série de pesos e contrapesos para amenizar os erros de estimativa no processo de tomada de decisão, pode interpretar erroneamente um ataque de mísseis e tentar contra-atacar, levando a uma guerra desnecessária [8]. Outro problema que poderíamos enfrentar é a

função de aprendizagem de sistemas de IA levá-los a evoluir de maneira que apresentem comportamentos indesejados. Alguns cientistas discutem sobre isso utilizando o termo denominado "singularidade tecnológica" [71], que de maneira simplificada, representa o momento em que os seres humanos perdem o controle sobre o avanço tecnológico. Isso é, um hipotético ponto de ruptura no qual as capacidades tecnológica superariam de maneira significativa as capacidades humanas, podendo ocorrer por meio da criação de uma superinteligência artificial, por exemplo, ou por avanços que ampliam a capacidade humana de forma significativa. O matemático Irving John Good, por exemplo, propõe a hipótese da máquina ultrainteligente [72], a qual é muito mais inteligente que qualquer ser humano, e dentre suas tarefas consta a de projetar máquinas ainda melhores. Nesse caso, aconteceria o que o autor chama de "explosão de inteligência", na qual não há mais a necessidade de que o homem projete novas invenções, pois teríamos substitutos mais capazes para essa função. A dificuldade, pois, não está mais "apenas" em construir máquinas cujas decisões não sejam nocivas à humanidade, mas também que suas criações tenham essa consciência.

Apesar dos que se preocupam, há aqueles que são otimistas em relação ao avanço tecnológico, a exemplo dos transumanistas. Esses acreditam que a biotecnologia será capaz de nos fazer mais resistentes às doenças, mais fortes, mais inteligentes, e apesar dos aspectos bioéticos que esse avanço envolve [73], os defensores dessa linha de pensamento costumam destacar as diversas evoluções às quais estaríamos sujeitos. O inventor e cientista da computação Ray Kurzweil sugere que a singularidade nos levará a ter controle sobre nossa própria mortalidade, além de compreender totalmente o pensamento humano e expandir muito o seu alcance [74]. Todavia, alerta para a ampliação na capacidade de agirmos de acordo com nossa inclinação destrutiva.

Asimov foi o primeiro a abordar a questão de diretrizes para o comportamento dos robôs, com as três leis da robótica: a primeira diz que um robô não pode ferir um ser humano ou, por omissão, permitir que um ser humano sofra algum mal; a segunda impõe que um robô deva obedecer às ordens que lhe sejam dadas por seres humanos, exceto nos casos em que tais ordens contrariem a primeira; e a terceira versa sobre a proteção obrigatória do robô à sua própria existência, desde que para isso não haja conflito com a primeira ou a segunda lei [2]. Mesmo que em ficção, essa proposta pode servir até hoje como base de uma solução para o problema da imprevisibilidade comportamental desses. Contudo, em diversos contos presentes em *Eu, Robô*, o próprio livro onde foram escritas as três leis, surgem situações conflitantes com o modo que as máquinas classificam essas em prioridade, principalmente quando os pedidos feitos pelos humanos não são bem claros, ou são mal formulados. Um exemplo é, caso vigente essas mesmas três leis a um robô que hipoteticamente já existisse, poderíamos pedir algo a esse sem sugerir um momento para

[2] As três Leis da Robótica foram propostas no livro *Eu, Robô*, de Isaac Asimov

que desistisse da tarefa. Desta maneira, para obedecer à segunda lei, a máquina poderia causar danos a propriedades, à natureza, ou a ela própria, mesmo sem violar a primeira lei. Portanto, mesmo que haja diretrizes bem estabelecidas para os programas de IA, deve-se ter muita cautela no processo de interpretação desses, em relação aos nossos comandos, e da conversão em ação.

Desejamos também que as máquinas hajam de acordo com uma moral, mas com o passar do tempo podem surgir dúvidas quanto a isso. James Moore propôs a ideia de agentes éticos explícito e implícito [75]. O primeiro age de acordo com as máximas éticas implementadas explicitamente nos seus códigos. Um problema desse tipo de máquina é a mudança na moral ao decorrer dos anos. Por exemplo, se supostamente essas existissem antes de 1888 no Brasil, iriam tentar manter a escravidão como algo legal. Já o segundo analisa as situações e desenvolve seus próprios julgamentos éticos, de maneira plausível, e é capaz de justificá-los, assim como um programa que joga xadrez e analisa seu passo futuro da maneira que acha mais eficiente para ganhar o jogo. Moore exemplifica situações nas quais agentes éticos explícitos podem superar os humanos na tomada de decisões, como definir a prioridade na tentativa de salvamento de pessoas em desastres naturais ou grandes acidentes automobilísticos [76]. Entretanto, como sugerem Russel e Norving [8], se dissermos a uma máquina que evoluísse a função utilidade, não teríamos como garantir que essa nos observe matando insetos, por os considerarmos seres mais primitivos que nós, e reproduza o mesmo comportamento conosco, caso nos julguem da mesma maneira.

O multibilionário, inventor e entusiasta da tecnologia, Elon Musk, adverte sobre os males que o desenvolvimento da IA pode nos causar. Em uma palestra no MIT, em 2014, Musk afirma que se tivesse de apostar qual seria a próxima ameaça a nível global, sua escolha seria a IA, além de comparar esse desenvolvimento à invocação de um demônio. Em uma reunião com governantes dos Estados Unidos, Musk pede por regulamentação da pesquisa nessa área por parte das autoridades, justificando que em IA precisamos ser pró-ativos, pois dependendo de como acontecer o erro talvez seja muito tarde para tentar consertá-lo. Esse, em conjunto com outros líderes do mercado da tecnologia, assinou uma promessa de não desenvolverem armas letais autônomas. Além desses tópicos, Musk reitera frequentemente em entrevistas e em suas redes sociais que não deve haver monopólio nessa área, pois uma inteligência não humana super desenvolvida, uma vez pendendo para o mal, poderá ser ainda mais perigosa não tendo concorrentes à sua altura. O que esses empresários e cientistas alegam querer não é o boicote ao desenvolvimento da IA, mas um planejamento responsável para esse processo. Como Isaac Asimov escreveu na parte final de *Eu, Robô*, perante o perigo do conhecimento, a melhor solução deve ser a sabedoria, e não a ignorância. Ou seja, em vez de evitar o perigo, o correto seria aprender a lidar com esse, buscando conhecer mais sobre os processos intermediários das máquinas durante as execuções de suas tarefas. Como mencionado na Introdução, essa é a principal motivação do nosso trabalho. Nos próximos capítulos desta tese, explicamos como desenvolvemos

nosso estudo.

3. Inteligência Artificial e Sistemas Dinâmicos

Para entender os métodos que utilizamos, é necessário apresentarmos os subcampos da Inteligência Artificial, explicarmos o que são sistemas dinâmicos, e como caracterizar seus comportamentos, pois é um modelo desse tipo de sistema o nosso principal objeto de estudo nos subcampos da IA que escolhemos abordar. Uma maneira de se investigar a dinâmica do sistema de Kuramoto, além das apresentadas neste capítulo, é mediante o parâmetro de ordem. Entretanto, assim como o próprio sistema, explicaremos melhor esse método no próximo capítulo.

3.1 Inteligência Artificial

Provavelmente devido ao fato de ser um campo de estudo novo e estar em rápido desenvolvimento, os subcampos da IA ainda não estão bem delimitados, havendo diversas interseções entre esses. Dentre os mais reconhecidos podemos citar Processamento de Linguagem Natural, Visão Computacional e Aprendizado de Máquina.

O Processamento de Linguagem Natural estuda a capacidade das máquinas em analisar e manipular a linguagem de maneira mais natural possível, isso é, de forma que se pareça com um ser humano. O principal objetivo desse subcampo é levar os computadores a conseguirem entender e redigir textos, ou discursos verbais, seja em uma língua específica ou traduzindo em outras, de maneira a extrair a informação, reconhecer o contexto e compreender os sentidos e os sentimentos contidos, as figuras de linguagem e a estrutura gramatical [77, 78].

A Visão Computacional, por sua vez, aborda processamento digital de imagens, buscando a construção de algoritmos capazes de interpretar o conteúdo visual de imagens, mesmo que determinado elemento tenha diferentes formas, por exemplo, ou hajam imperfeições e ruídos [79].

O Aprendizado de Máquina é o que escolhemos estudar em nosso trabalho, portanto o abordaremos de modo mais detalhado.

Aprendizado de Máquina é o subcampo da IA que recebe mais destaque nos meios científico e tecnológico. Como explicado no início deste capítulo, esse aborda a construção de algoritmos que de forma autônoma são capazes de fazer predições ou tomar decisões, a partir dos dados de entrada, e de melhorar seu desempenho mediante repetição. Além disso, um outro objetivo é o estudo de como são regidos o raciocínio e o comportamento dos sistemas de aprendizagem, considerando o quão bem sucedido um programa pode ser

ao realizar uma determinada tarefa, levando em consideração a influência do volume de dados fornecidos, a qualidade da técnica utilizada e a capacidade da máquina que a realiza [3].

É comum classificar Aprendizado de Máquina em três abordagens. Na mais utilizada, denominada aprendizado supervisionado, é fornecido um conjunto de dados de entrada e também os dados de saída esperados, para que haja uma fase de treinamento, no qual a máquina faz o tratamento dos dados de entrada, e compara o resultado obtido com o resultado esperado. Dessa maneira, faz correções e aumenta sua eficiência, se tornando capaz de fazer predições para outros dados de entrada que não estejam rotulados, ou seja, relacionados a um resultado predeterminado. No caso do aprendizado não supervisionado, não há rótulos fornecidos para os dados de entrada, portanto, é útil para identificar padrões e categorias em um conjunto de dados cujo resultado é desconhecido ou imprevisível. A terceira é o aprendizado por reforço, no qual os dados de entrada na fase de treinamento servem apenas para indicar se as ações do programa são corretas ou não, em vez no fim de cada processo, como no supervisionado. Nesse caso, há reforços positivos e negativos para que a máquina aja de acordo com um comportamento esperado, ao decorrer do tempo, e até certo limite, maximizando as recompensas totais, normalmente em um ambiente dinâmico e desconhecido [3].

Dentro do aprendizado não supervisionado, há duas técnicas mais comumente usadas. Uma dessas, denominada agrupamento, separa os dados de entrada em grupos, sem que esses grupos sejam rotulados previamente. Essa é utilizada em sistemas de recomendação de músicas e séries, classificação de filmes por gênero, ou clientes por comportamento de compra, por exemplo. A outra é conhecida por redução de dimensionalidade. Essa é útil quando há uma grande quantidade de variáveis, e muitas dessas estão altamente correlacionadas ou não apresentam informação útil ao problema. Dessa maneira, a tentativa é de criar uma nova representação dos dados, com menos dimensões, mantendo a maior variância possível entre essas [3]. O aprendizado por reforço funciona a partir de tentativa e erro. Para executar determinada tarefa, mesmo que tenha êxito, o programa pode executar muitas ações inúteis ou prejudiciais durante o processo, dependendo do contexto. Desta maneira, o envolvimento humano é importante nas mudanças no ambiente e nos ajustes ao sistema de recompensas e penalidades, com o objetivo de levar a máquina a cumprir a tarefa da maneira mais adequada. Além disso, é necessário cautela na construção do ambiente de simulação, pois um erro no treinamento de um programa que jogue Xadrez, por exemplo, não tem as mesmas consequências de outro que dirige um carro de maneira autônoma. Portanto, na preparação da simulação, deve-se tentar compreender no modelo o maior número de variáveis semelhantes ao que se espera na realidade. Escolhemos trabalhar com o aprendizado supervisionado, sendo assim, será a abordagem que nos aprofundaremos [7].

No aprendizado supervisionado pode-se ter como objetivo realizar regressão ou classificação, dependendo do problema que está sendo tratado. Quando os dados de saída esperados são variáveis contínuas, trata-se de um problema de regressão. Um exemplo é a previsão do preço de uma casa, cujos dados de entrada são o tamanho da casa, o tamanho do terreno, o número de quartos e banheiros, a qualidade da localização de forma quantificada, dentre outros. Outros exemplos são a previsão da idade de uma pessoa, dada uma foto dessa, ou a previsão da temperatura a partir dos dados meteorológicos. O tipo da função para melhor representar uma tendência comportamental dos dados de entrada vai ser aquela que reduza a diferença entre a hipótese e os dados de saída, na fase de treinamento, desde que não haja sobre ajuste (ou uma superadaptação), também conhecido como problema de alta variância, isso é, uma função que se ajuste muito aos pontos de treinamento, mas que não representa bem um comportamento geral. Na classificação, os dados de saída formam um conjunto finito de valores discretos, cujo objetivo é separar as entradas em diferentes categorias, de maneira semelhante ao agrupamento do modo não supervisionado, porém as categorias são determinadas previamente na fase de treinamento. Essas categorias podem ser somente duas, caracterizando uma classificação binária, como nos casos em que se busca avaliar se um e-mail é spam ou diagnosticar tumores malignos a partir da análise de imagens, ou mais categorias, por exemplo na distinção em uma foto de uma rua, entre carros, bicicletas, pedestres, postes, hidrantes, dentre outros. Há muitos tipos de algoritmos, em aprendizado supervisionado, que representam diferentes técnicas de tratar os dados de entrada, dos quais os mais conhecidos serão abordados a seguir.

A regressão é um método para descobrir uma relação entre duas ou mais variáveis, podendo ser linear ou não linear. A função custo é comumente o erro quadrático médio, medido entre a hipótese obtida do modelo e a previsão. O objetivo é obter o conjunto de coeficientes que minimizem essa função custo, com técnicas como o gradiente descendente ou da equação normal, por exemplo [80]. Na classificação linear, se constrói funções lineares que separam o hiperespaço das variáveis independentes em diferentes classes. A regressão logística estabelece uma relação não linear entre as variáveis, e apesar do seu nome, costuma ser utilizada em problemas de classificação. A função logística limita os valores da hipótese entre 0 e 1, tornando possível a interpretação dessa como a probabilidade de os valores de entrada serem associados a uma saída 1, seja qual for a interpretação desejada em cada caso, como spam, tumor maligno, dentre outras [7]. A decisão do valor limite para considerar a hipótese como sendo 0 ou 1 vai depender de quão preciso, sensível ou específico se espera que o modelo seja. Nesses modelos, para tratar problemas de classificação, a abordagem em relação à função custo e à obtenção dos coeficientes é semelhante, o que muda nesse caso é a forma que essa função se apresenta.

O método dos k-vizinhos mais próximos (k-NN, do inglês: *k-Nearest Neighbors*) pode ser usado para regressão, o qual consiste em atribuir a um determinado dado de entrada, a saída correspondente, a média dos k vizinhos mais próximos no hiperespaço

das variáveis independentes, porém, seu uso é mais comum para fins de classificação. Nesse caso, é fornecido inicialmente um conjunto de dados classificados, representados por vetores no hiperespaço das variáveis independentes, os quais vão ter suas distâncias medidas em relação a um novo vetor não classificado. Para classificar esse novo vetor o algoritmo ordena, por distância os dados já classificados, e escolhe os k vizinhos mais próximos, onde k pode ser calculado manualmente ou por algum método automatizado. Dentre esses k vizinhos, são observadas as suas classes e contabilizados como votos para o dado não classificado, o qual vai ter sua classe atribuída àquela que receber mais votos [8].

O método conhecido como máquina de vetores de suporte (SVM, do inglês: *Support Vector Machine*), comumente usado para classificação, semelhantemente ao classificador linear constrói um hiperplano no hiperespaço das variáveis independentes, o qual serve como um limite de decisão para separar as classes. Porém, nesse modelo, as distâncias do hiperplano em relação aos pontos mais próximos são maximizadas. Esses pontos mais próximos ao hiperplano são alguns dos dados de entrada, os quais são denominados vetores de suporte por servirem como referência para o cálculo das distâncias. Para determinar a posição desse limite de decisão costuma-se usar a função de Kernel, normalmente na forma gaussiana. Esse método também é utilizado para regressão, porém em vez de separar as classes das variáveis, o hiperplano tem a função de representar a tendência dos dados de entrada [8].

A Árvore de Decisão cumpre seu objetivo executando uma sequência de testes a partir dos dados de entrada. Cada nó de decisão, na árvore, corresponde a um teste do valor de uma das variáveis de entrada, cujas ramificações determinam intervalos de valores que delimitam regiões no hiperespaço das variáveis independentes, e os nós de término retornam a decisão a ser tomada. Esse tipo de algoritmo é capaz de abordar variáveis discretas e contínuas [8].

A evolução da neurociência nos levou à compreensão da atividade mental como a atividade eletroquímica, entre redes de células cerebrais, denominadas neurônios (figura 1).

Essa noção inspirou o surgimento de uma das formas mais populares e eficazes de aprendizagem, denominada redes neurais artificiais, cujo modelo pioneiro é atribuído a McCulloch e Pitts, como já mencionado [8].

As ANNs são compostas por unidades de processamento de informações, modelos matemáticos que fazem analogia aos neurônios. A eles é aplicado um estímulo m-dimensional $\vec{x}(i)$ através dos m nós de entrada, cuja resposta é produzir um escalar $v(i)$, dado pela soma ponderada dos componentes de entrada da seguinte maneira:

$$v(i) = \sum_{k=1}^{m} w_k(i) x_k(i), \tag{3.1}$$

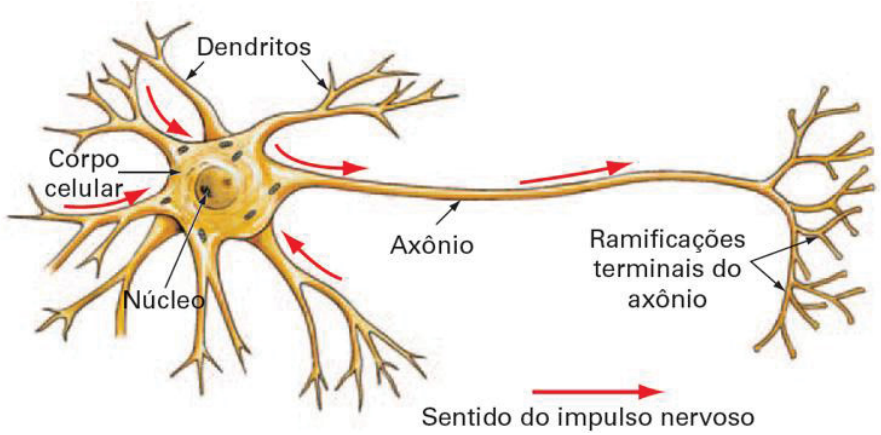


Figura 1 – Cada neurônio consiste em um corpo celular, o qual contêm um núcleo. A partir do corpo celular, ramificam-se fibras chamadas dendritos, e uma fibra longa denominada axônio, o qual se estende por uma distância muito maior do que é possível representar nessa figura. Os sinais se propagam pelo axônio de um neurônio, em uma reação eletroquímica, aos dendritos de outros, através de junções conhecidas por sinapses. Acredita-se que esses mecanismos sejam a base para o aprendizado no cérebro. Figura retirada de [81].

onde $w_0(i)$, $w_1(i)$, $w_2(i)$, ..., $w_m(i)$ são os m pesos sinápticos do neurônio, e i pode representar tanto a evolução temporal de um conjunto de dados, quanto simplesmente distinguir um conjunto de outros sem haver relação temporal. É importante notar que ao vetor de entrada é adicionado uma componente $x_0(i)$, cujo valor é sempre um, e o peso sináptico associado é o $w_0(i)$, conhecido como *bias* ou viés. A saída $y(i)$ é obtida mediante uma função $\varphi(\cdot)$ aplicada sobre $v(i)$, conhecida como função de ativação, a qual pode ser linear, ou como é de costume se usar na área, uma sigmoide. Essa saída tem a seguinte forma:

$$y = \varphi(v(i)). \tag{3.2}$$

Na fase de treinamento, é fornecido ao sistema uma saída desejada $d(i)$, a qual serve para medir o erro $e(i)$, da saída $y(i)$ do neurônio em relação a essa:

$$e(i) = d(i) - y(i). \tag{3.3}$$

Há duas maneiras distintas de se conectar esses neurônios: a rede neural de alimentação para a frente e a rede neural recorrente [7].

As FNNs são dispostas em camadas, podendo ser somente as de entrada e de saída (figura 2), ou conter camadas de unidades ocultas (figura 3). Contudo, as conexões ocorrem somente na direção entrada-saída, nunca entre neurônios de uma mesma camada, ou seja, não há estado interno a não ser os próprios pesos.

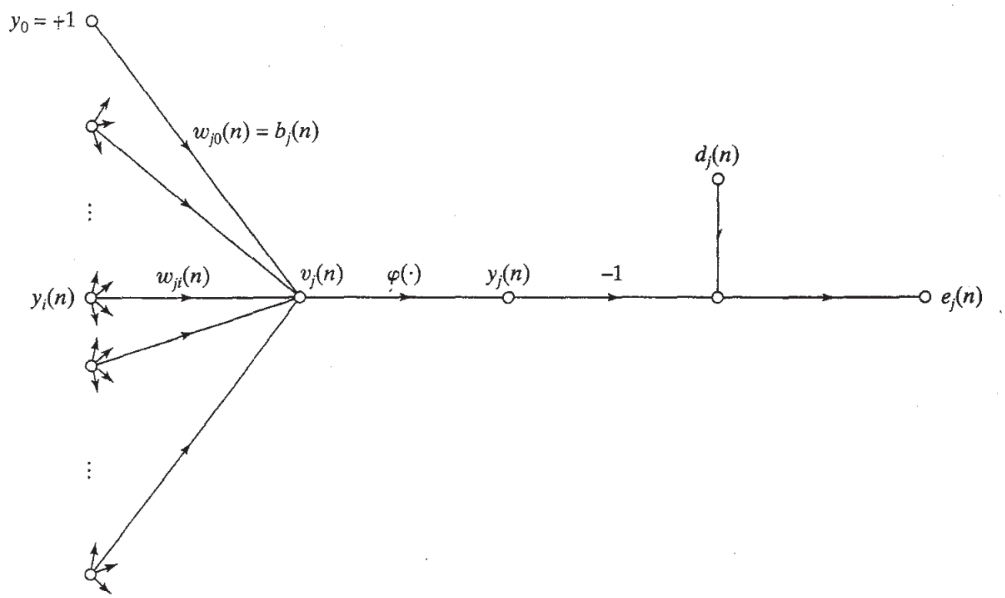


Figura 2 – Grafo do fluxo de sinal ressaltando os detalhes do neurônio de saída j conectado ao de entrada i. Figura retirada de [7].

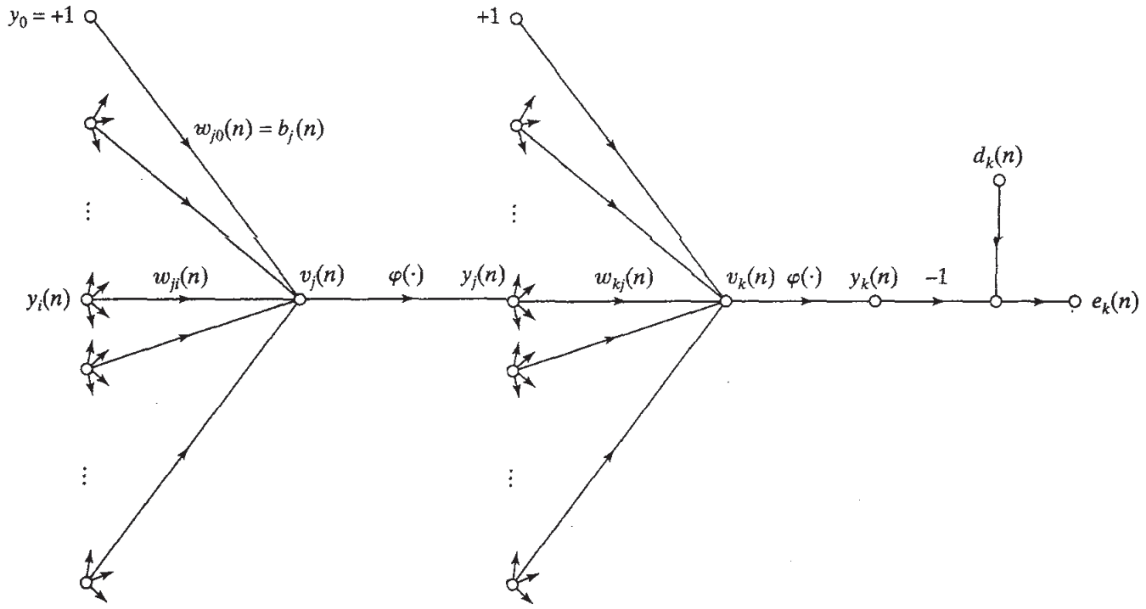


Figura 3 – Grafo do fluxo de sinal ressaltando os detalhes do neurônio de saída k conectado ao oculto j, e deste conectado ao de entrada i. Figura retirada de [7].

Além disso, podemos escolher obter somente uma saída $y(i)$ para a rede, ou múltiplas saídas representadas pelo vetor $\vec{y}(i)$, de acordo com a saída $d(i)$ ou as saídas $\vec{d}(i)$ desejadas, respectivamente. No caso em que não há camadas ocultas, a atualização dos pesos sinápticos costuma ser dada a partir de uma função custo dependente do erro $\vec{e}(i)$, mediante o método gradiente descendente, por exemplo. Já na presença de unidades

ocultas, caracterizando uma rede de múltiplas camadas, um método de aprendizagem amplamente utilizado é o de retro propagação [82], que se baseia no gradiente descendente, pois avalia-se o quanto cada peso da rede contribuiu para o erro total, porém, não é possível fazer esse cálculo de maneira direta. Nesse método, para cada conjunto de dados $\vec{x}(i)$ e $\vec{d}(i)$, a computação para a frente é realizada, obtendo as saídas $\vec{y}(i)$, e em seguida a computação para trás, na qual se calcula, camada a camada, a contribuição de cada peso sináptico para a função custo de toda a rede, e a partir disso, os ajusta. Não há uma regra específica para a escolha do número de neurônios e camadas na rede. Um número maior destas e daqueles pode permitir que o programa execute tarefas mais complexas, porém, há um maior custo computacional e aumenta o risco de acontecer uma superadaptação. Uma maneira de escolher essas quantidades é separar os dados em treinamento, validação cruzada e teste. O treinamento serve para ajuste dos pesos, a validação cruzada para ajuste dessas quantidades e a seleção da configuração que apresentar menor erro, e o teste para avaliar se o programa é eficiente na realização da tarefa [8].

As RNNs alimentam suas entradas com suas próprias saídas, isso é, a topologia das conexões de uma RNN possui ciclos. Uma RNN desenvolve uma dinâmica autossustentada ao longo do caminho das conexões até mesmo na ausência de uma entrada externa, o que leva a interpretarmos uma RNN como um sistema dinâmico, enquanto as FNNs são consideradas funções. Além disso, os dados de entrada comumente são tomados como um sinal externo ao sistema. Uma vez que a RNN preserva em seu estado interno uma transformação não linear do histórico dessa entrada, então apresenta memória dinâmica, portanto, se torna capaz de tratar dados dinâmicos [83]. Uma RNN costuma ter a dinâmica representada, quando se toma o tempo como uma variável discreta, da seguinte maneira:

$$\vec{x}(i) = \varphi(W_{in}\vec{u}(i) + W\vec{x}(i-1)), \tag{3.4}$$

onde $\vec{x}(i)$ e $\vec{x}(i-1)$ são os vetores que representam os estados dos neurônios do sistema, $\vec{u}(i)$ é o sinal externo de entrada, $\varphi(\cdot)$ simboliza a função de ativação do neurônio e i é a unidade de tempo discreta de evolução do sistema. As matrizes W e W_{in} representam as conexões entre os neurônios entre si, e os neurônios com as entradas, respectivamente. A dinâmica da rede poderia ser representada também com o tempo como variável contínua, mas para essa explicação o manteremos como variável discreta. A saída da RNN, por sua vez, é obtida da seguinte maneira:

$$\vec{y}(i) = \varphi_{out}(W_{out}\vec{x}(i)), \tag{3.5}$$

onde as componentes de $\vec{y}(i)$ são as saídas da RNN, $\varphi_{out}(\cdot)$ é a função de ativação de saída, e W_{out} é a matriz responsável, pelos pesos dos estados dos neurônios, na soma

ponderada da saída. Essa classe de redes costuma ser usada no contexto da neurociência computacional, com foco em modelar cérebros biológicos com a maior fidelidade às suas propriedades quanto possível, e com propósitos mais mecanicistas, as utilizando como ferramenta para a realização de tarefas. Nesse segundo enfoque, há uma abordagem mais voltada ao estudo de sistemas dinâmicos não lineares, portanto será com esse que lidaremos daqui em diante. Apesar de se mostrarem promissoras, as RNNs apresentam dificuldade na fase de aprendizagem com métodos baseados no gradiente descendente, por exemplo, devido à sua arquitetura. Dentre os motivos para essa dificuldade, destaca-se o custo computacional que até mesmo a atualização de um único parâmetro possa apresentar, devido aos ciclos presentes na rede, tornando a atualização de muitos parâmetros inviável [84].

Uma solução para esse problema foi o desenvolvimento de uma nova abordagem de RNNs denominada computação de reservatório. Nessa, somente os pesos W_{out} da combinação linear que relaciona os sinais dos neurônios com as componentes da saída são atualizados, enquanto os pesos W_{in}, que correspondem ao sinal de entrada com os neurônios da rede, assim como os pesos W das conexões dos neurônios entre si, permanecem os mesmos durante todo o treinamento. Além disso, em vez de a saída ser obtida a partir de uma função de ativação, no RC a saída costuma ser simplesmente uma combinação linear dos componentes do vetor de estado:

$$\vec{y}(i) = W_{out}\vec{x}(i). \tag{3.6}$$

Dessa maneira, não há necessidade de utilizar algoritmos com maior complexidade para treinar a rede. As categorias das variáveis de entrada e de saída dependem da tarefa a ser realizada em cada aplicação, como classificação de padrões, predição de séries temporais, e assim por diante [83, 85, 86, 87]. Devido ao fato de RC possuir um treinamento mais simples, com baixo custo computacional e processamento em tempo real, o método é atrativo também para pesquisadores que não são especificamente da área de redes neurais, tais quais medicina, engenharia, comunicação, segurança digital e financeira [84], além de apresentar, assim como programas semelhantes, características parecidas em relação a cérebros reais na realização de algumas tarefas [88, 89, 90, 91, 92, 93]. Se destaca também sua capacidade na manutenção da estrutura do modelo para a representação de novas saídas. Uma vez que os pesos de saída são os únicos a serem atualizados, o reservatório pode ser preservado de uma computação para a outra. Esses fatores favorecem a escolha da utilização do RC. Porém, diferentemente das propostas iniciais de se criar o reservatório com propriedades escolhidas aleatoriamente, se observou que essa escolha deve ser feita de acordo com a tarefa a ser realizada e o custo computacional estimado. Dentre os principais métodos de RC, dois merecem destaque.

Redes de estados de eco são compostas por neurônios artificiais cuja dependência temporal é discreta. Esse método se baseia na observação de que, se uma RNN possuir certas propriedades, o treinamento de uma saída provinda de uma combinação linear é suficiente para se obter um desempenho excelente em aplicações práticas. O reservatório é a parte responsável pela dinâmica do sistema, de onde se obtém os vetores de estado chamados de ecos, nesse caso, devido ao comportamento da rede apresentar a propriedade de estado de eco. Isso é, os estados do sistema e as entradas anteriores têm cada vez menos influência, ao decorrer do tempo, sobre os estados posteriores. Comumente, para que essa propriedade aconteça, o raio espectral da matriz peso de conexões entre os neurônios, ou seja, o maior autovalor absoluto dessa matriz, deve ser menor do que um. Observou-se também que para obter bons resultados com esse tipo de programa, o reservatório deve conter muitas unidades, as quais devem ser conectadas de maneiras esparsa e aleatória. A função de ativação não linear para cada neurônio costuma ser sigmoide, enquanto a saída, uma combinação linear dos componentes do vetor de estado. O método de treinamento mais utilizado para os pesos de saída é o gradiente descendente [84, 94].

Máquinas de estado líquido são compostas por neurônios artificiais cuja dependência temporal é contínua, e foi desenvolvido buscando maior compreensão das propriedades dos microcircuitos neurais do cérebro [95, 96, 97]. Dessa maneira, o modelo do reservatório nesse método se baseia nos de Redes Neurais do tipo *Spiking* e conexões sinápticas dinâmicas. A topologia e a conectividade da RNN, no LSM, seguem características de redes neurais biológicas. Por exemplo, a dependência da probabilidade de conexão entre dois neurônios com a distância entre estes. Nesse caso a operação é denominada computação líquida, devido a sua dinâmica apresentar ondulações perante estímulo externo, recebido pelo sinal de entrada. Apesar de simular de maneira mais fiel as propriedades de redes neurais biológicas e ser capaz de processar informações mais complicadas, LSM costuma apresentar um custo computacional muito maior que o ESN, sendo assim menos utilizado, quando comparados os dois métodos [84, 94].

Têm sido propostas variações de modelos de RC, seja no método de treinamento, na associação a outros modelos de aprendizagem, ou na arquitetura do reservatório. Uma variação que nos interessou foi o uso de outros sistemas dinâmicos baseados em fenômenos reais, na composição do reservatório, em vez das RNNs citadas. Observou-se que a eficiência de um sistema em realizar tarefas computacionais, nesse contexto, depende de alguns fatores. A alta dimensionalidade do reservatório e a distinção entre as dinâmicas das suas unidades, por exemplo, facilitam a separação da variáveis de entrada, quando o objetivo é classificá-las, e permite que se estabeleça dependências, entre essas, em tarefas de predição de uma maior variedade de dinâmicas. A não linearidade do sistema também é importante para que seja possível a separação linear das entradas, inicialmente não linearmente separáveis, no caso de classificação, e para extrair dependências não lineares dessas, em predições.

O reservatório pode ser composto por sistemas físicos reais, cujos valores de estado são lidos diretamente do experimento, ou por modelos matemáticos desses sistemas evoluídos computacionalmente. Esses sistemas físicos podem ser baseados em fenômenos dos campos mecânico, eletrônico, fotônico, químico ou biológico [84], por exemplo, cuja dinâmica pode ser representada por mapas ou fluxos. Na próxima seção examinaremos o que são mapas e fluxos e quais são suas propriedades. Alguns desses modelos de computação de reservatório físico têm se mostrado eficientes, rápidos e com baixo custo computacional, especialmente no processamento de dados que são dinâmicos e produzidos e transmitidos em grandes quantidades. Já outros têm chamado atenção devido às suas capacidades de processar informações sob limitações realísticas em química, engenharia, física, e em especial em biologia, na qual se espera que os PRC possam trazer novas noções sobre o mecanismo de processamento de informação, em tempo real, em diversas regiões do cérebro [98, 99, 100, 101, 102, 103].

Um desafio agora é explorar diferentes modos de implementação para vários tipos de reservatórios, investigando aspectos como eficiência, custo computacional, aplicabilidade, tecnologias para aplicação, e obter maior compreensão em relação à dinâmica do sistema durante a realização das tarefas [84, 94]. Neste trabalho, queremos contribuir para esse desenvolvimento, com ênfase no último aspecto mencionado.

3.2 Sistemas Dinâmicos

Em meados do Século XVII, Isaac Newton desenvolveu as equações diferenciais, descobriu as leis do movimento e a gravitação universal, e as combinou para explicar as leis de Kepler. Mais especificamente, Newton resolveu o problema dos dois corpos, como o da gravitação de cada planeta em torno do Sol, ou da Lua em torno da Terra, por exemplo.

Vários cientistas posteriores tentaram estender os métodos analíticos de Newton para 3 corpos, porém esse problema foi considerado impossível de se resolver até os trabalhos de Poincaré, no final do Século XIX.

Poincaré propôs uma abordagem mais qualitativa e menos quantitativa. Por exemplo, em vez de tentar prever a posição exata dos planetas a cada instante, o cientista fez perguntas como: “será que o sistema solar é estável para sempre, ou eventualmente algum planeta escapará para o infinito?”. Poincaré desenvolveu uma abordagem geométrica poderosa para analisar essas questões. Foi pioneiro também ao considerar a possibilidade da existência de caos, no qual um sistema determinístico exibe comportamento aperiódico que depende das condições iniciais de maneira sensível, dessa maneira tendo sua dinâmica imprevisível a tempos mais longos.

Durante a primeira metade do Século XX, diversos cientistas utilizaram dos métodos de Poincaré para expandir o entendimento sobre mecânica clássica, porém foi o avanço

tecnológico dos computadores, a partir de 1950, que possibilitou um desenvolvimento mais significativo dos estudos em sistemas dinâmicos não-lineares.

Em 1963, o matemático Edward Lorenz estava estudando sobre a imprevisibilidade das mudanças climáticas, quando observou que as soluções das suas equações tendiam a oscilar de maneira irregular e aperiódica. Além disso, Lorenz observou também que, mesmo se suas simulações partissem de condições iniciais muito próximas, logo as dinâmicas resultantes logo divergiriam, demonstrando assim que os sistemas utilizados apresentavam sensibilidade às condições iniciais.

A partir de 1970, dinâmicas caóticas passaram a ser estudadas em diversos campos do conhecimento. Feigenbaum descobriu que há certas leis que governam a transição de comportamento regulares para caóticos, isto é, que sistemas dinâmicos completamente distintos podem se tornar caóticos de maneiras semelhantes. Seus trabalhos estabeleceram, portanto, uma relação entre caos e transições de fase, o que atraiu muitos cientistas ao estudo em Sistemas Dinâmicos, área que continua a crescer até os dias atuais.

Um sistema pode ser definido como um conjunto de elementos agrupados por alguma interação, de modo que existam relações de causa e efeito nos fenômenos que ocorrem com esses. Para que este sistema seja denominado dinâmico, algumas grandezas que caracterizam seus objetos constituintes devem variar no tempo [104, 105].

Quando a evolução temporal ocorre de maneira discreta, ou seja, a variável tempo n assume somente valores inteiros, o sistema denomina-se mapa e é representado por um conjunto de equações de diferenças, o qual pode ser representado da seguinte maneira:

$$\begin{aligned} x_{n+1}^{(1)} &= M_1(x_n^{(1)}, x_n^{(2)}, \ldots, x_n^{(N)}), \\ x_{n+1}^{(2)} &= M_2(x_n^{(1)}, x_n^{(2)}, \ldots, x_n^{(N)}), \\ &\vdots \\ x_{n+1}^{(N)} &= M_N(x_n^{(1)}, x_n^{(2)}, \ldots, x_n^{(N)}), \end{aligned} \tag{3.7}$$

Ou, de forma compacta, na notação vetorial:

$$\vec{x}_{n+1} = \vec{M}[\vec{x}_n], \tag{3.8}$$

onde $\vec{x}$ é um vetor N-dimensional e $\vec{M}$ uma função vetorial desta variável [104, 105].

O sistema cujo tempo varia de maneira contínua recebe o nome de fluxo, e tem sua dinâmica descrita por equações diferenciais ordinárias (EDO's). Um fluxo é caótico

somente se satisfizer as seguintes condições: conter pelo menos três variáveis independentes que evoluam no tempo; as suas EDO's possuírem ao menos uma não-linearidade; duas condições iniciais muito próximas divergirem exponencialmente no tempo [104, 105]. Um fluxo N-dimensional, não linear e autônomo pode ser representado da seguinte maneira:

$$\begin{aligned}
\frac{dx^{(1)}}{dt} = \dot{x}^{(1)} &= F_1(x^{(1)},x^{(2)},...,x^{(N)}),\\
\frac{dx^{(2)}}{dt} = \dot{x}^{(2)} &= F_2(x^{(1)},x^{(2)},...,x^{(N)}),\\
&\vdots\\
\frac{dx^{(N)}}{dt} = \dot{x}^{(N)} &= F_N(x^{(1)},x^{(2)},...,x^{(N)}),
\end{aligned} \tag{3.9}$$

Ou, vetorialmente:

$$\frac{d\vec{x}}{dt} = \vec{F}[\vec{x}(t)], \tag{3.10}$$

onde $\vec{x}$ é um vetor N-dimensional e $\vec{F}$ uma função vetorial desta variável [105]. Esse conjunto de equações representa um sistema dinâmico, pois dado um estado inicial $\vec{x}(0)$, mediante a solução das equações (3.9), obtém-se o estado final $\vec{x}(t)$ para qualquer $t > 0$. Existem três maneiras de estudar o comportamento de um fluxo: (i) em casos raros, pode-se integrá-lo analiticamente, aplicando alguma metodologia matemática para resolução de EDOs, para encontrar as soluções $\vec{x}(t)$; (ii) estudá-lo qualitativamente, procurando descobrir quais famílias de soluções podem ser associadas a determinados valores dos parâmetros e/ou das condições iniciais do sistema dinâmico em questão; (iii) resolvê-lo numericamente, mediante o uso de um computador para integrar as EDOs do sistema e obter as séries temporais de $\vec{x}(t)$ para um determinado tempo de integração [104, 105].

No caso dos mapas, se for invertível, ou seja, assim como podemos obter:

$$\vec{x}_{n+1} = \vec{M}[\vec{x}_n], \tag{3.11}$$

também é possível obtermos:

$$\vec{x}_n = \vec{M}^{-1}[\vec{x}_{n+1}], \tag{3.12}$$

a condição para caoticidade, além da presença de não-linearidade, é que $N \geq 2$. Caso seja não invertível, o comportamento caótico pode acontecer até para mapas de uma dimensão.

É possível reduzir um fluxo N-dimensional a um mapa $(N-1)$-dimensional mediante a técnica conhecida por seção de Poincaré. Nesse método, em um espaço de N variáveis correspondentes ao fluxo, escolhe-se uma superfície de dimensão $N-1$, e observa-se os pontos em que a trajetória do sistema atravessa essa superfície. O mapa de Poincaré corresponde ao mapa que leva o sistema de um ponto ao outro em um tempo futuro, e vice e versa, quando houver invertibilidade. Na subseção seguinte, abordaremos de modo mais detalhado o espaço das variáveis, também denominado espaço de fases, e as trajetórias [105].

Outro método de se obter um mapa a partir de um fluxo é discretizar o tempo $t_n = t_0 + nT$ em intervalos T, escolhido de acordo com o sistema em questão. Desta maneira, o vetor de estado $\vec{x}(t)$ se torna um vetor de estado discreto $\vec{x}_n = \vec{x}(t_n)$. Sendo assim, tomando $\vec{x}_n$ como condição inicial das nas equações (3.9), e as integrando para frente por um tempo T, determina-se $\vec{x}_{n+1}$. Esse mapa $\vec{x}_{n+1} = \vec{M}(\vec{x}_n)$ é invertível se for possível integrar o fluxo para trás no tempo. Além disso, diferentemente do mapa de Poincaré, a dimensionalidade deste mapa é a mesma do fluxo que a originou [105].

3.2.1 Espaço de fases e atratores

Os sistemas dinâmicos podem ser divididos entre conservativos e dissipativos. Para compreender as diferenças entre esses, é necessária a definição de um espaço de fases, também conhecido como espaço de estados ou espaço das variáveis.

O espaço de fases é um espaço N-dimensional, cujos eixos coordenados são as variáveis dinâmicas do sistema. Um estado é representado como um ponto nesse espaço, o qual possui coordenadas $x^1(t), x^2(t),..., x^N(t)$, e tem sua dinâmica regida pelas equações (3.9) [105, 104].

A figura 4 mostra o caminho seguido pelo estado de um sistema hipotético, com três dimensões, à medida que este evolui temporalmente. O espaço gerado pelas variáveis x, y e z, equivalente às coordenadas $x^1(t)$, $x^2(t)$ e $x^3(t)$, de acordo com a notação apresentada no parágrafo anterior, é denominado espaço de fases. Os vetores $\vec{r}(0)$ e $\vec{r}(t)$ representam o conjunto de condições iniciais e a solução do sistema após passado um tempo t, respectivamente. O caminho percorrido nesse espaço é conhecido como trajetória ou órbita [105].

Tomando um conjunto de condições iniciais, de maneira que forme um volume no espaço de fases, pode-se definir o sistema como dissipativo caso esse volume se contraia com o decorrer do tempo, ou conservativo, quando este for preservado. Quando o sistema possuir mais do que 3 dimensões, costuma-se usar o termo hipervolume.

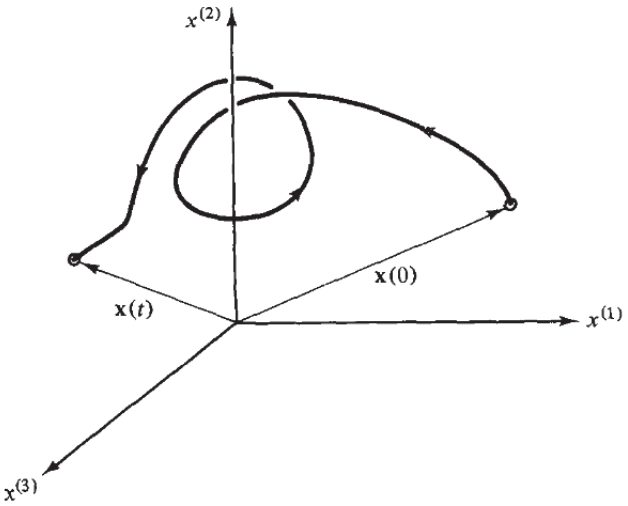


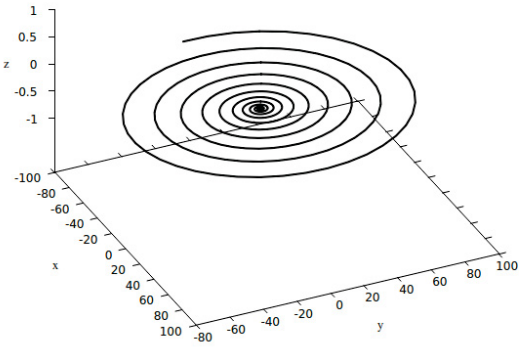
Figura 4 – Esboço de uma trajetória (fluxo) em um sistema dinâmico contínuo tridimensional. Figura retirada de [105].

Existe um conjunto invariante de pontos no espaço de fase em sistemas dissipativos, denominado atrator, para o qual as órbitas próximas convergem após um tempo suficientemente longo [104, 105]. Em sistemas autônomos contínuos tridimensionais, o atrator pode ser dos seguintes tipos:

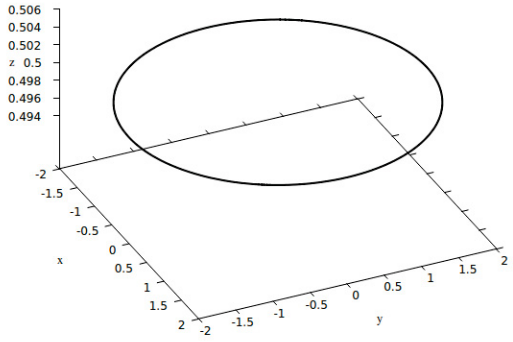
- **ponto de equilíbrio**: é uma solução do sistema, ou seja, um ponto do espaço de fases, para o qual o comportamento do sistema converge e independe do tempo;
- **ciclo limite ou periódico**: representa um comportamento periódico no tempo, com amplitude e período determinados pelos valores dos parâmetros e pela forma das equações;
- **quase-periódico**: representa um regime quase-periódico, com duas frequências fundamentais independentes, cuja razão entre estas é um número irracional. Nesse caso, as trajetórias nunca fecham sobre si mesmas;
- **caótico**: apresenta um comportamento aperiódico e dependência sensível às condições iniciais. Isto é, tomadas duas trajetórias vizinhas no espaço de fases, e evoluído o sistema no tempo, a distância entre essas divergirá exponencialmente.

A partir de um estudo com o cálculo dos expoentes de Lyapunov, podemos classificar os atratores, conforme abordado na próxima subseção.

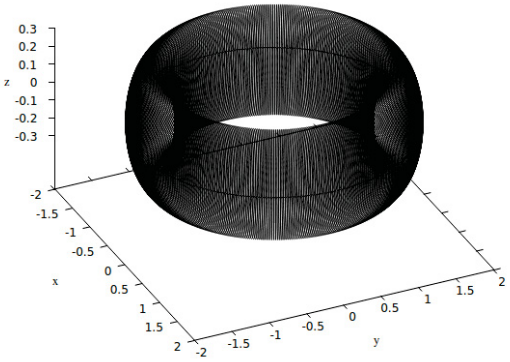
Quando trata-se de tempos discretos, um mapa é considerado conservativo se o volume no espaço de fases é mantido a cada iteração. Isso acontece quando o módulo do determinante da matriz Jacobiana, das derivadas parciais, é igual a 1:



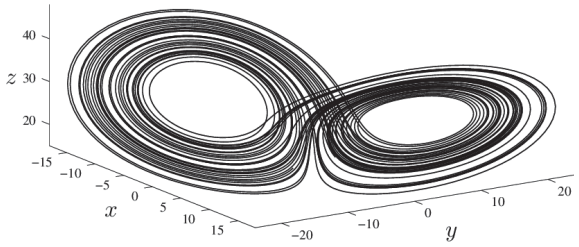
(a) Ponto de equilíbrio.



(b) Periódico.



(c) Quase-periódico.



(d) Atrator caótico.

Figura 5 – Possíveis atratores em sistemas dinâmicos contínuos dissipativos tridimensionais. Os painéis (a), (b) e (c) representam, respectivamente, os atratores ponto de equilíbrio, periódico e quase-periódico. Essas painéis foram gerados pelo autor, utilizando Fortran e Gnuplot, com funções trigonométricas que simulam os atratores representados. O painel (d) apresenta o atrator caótico do sistema de Lorenz [106].

$$|det[J(\vec{x})]| = \left| det\left[\frac{\partial \vec{M}(\vec{x})}{\partial \vec{x}}\right]\right| = 1, \tag{3.13}$$

No caso de $|det[J(\vec{x})]| < 1$ o mapa é dissipativo e também pode apresentar atratores, assim como no caso de sistemas contínuos.

3.2.2 Expoentes de Lyapunov

Para que um sistema apresente comportamento caótico, é necessária uma dependência sensível às condições iniciais. Considerando duas condições iniciais, $\vec{x_1}(0)$ e $\vec{x_2}(0)$, separadas por uma distância muito pequena, a rigor infinitesimal, $\Delta(0)$, se supõe que essas evoluam no tempo por um sistema contínuo gerando as trajetórias $\vec{x_1}(t)$ e $\vec{x_2}(t)$, respectivamente, de acordo com a representação da figura 6 [105].

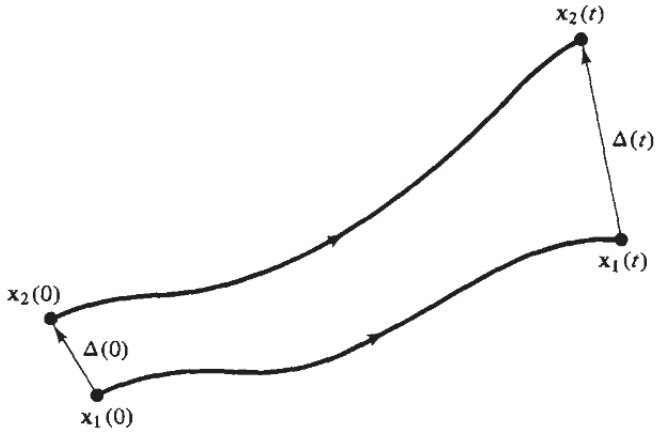


Figura 6 – Órbitas esquemáticas $\vec{x}_1(t)$ e $\vec{x}_2(t)$ geradas evoluindo temporalmente o sistema a partir de duas condições iniciais próximas $\vec{x}_1(0)$ e $\vec{x}_2(0)$. Figura retirada de [105].

Para qualquer instante $t > 0$, a separação entre as trajetórias é $\Delta(t) = \vec{x_2}(t) - \vec{x_1}(t)$. O sistema apresenta dependência sensível às condições iniciais se, em média, o módulo da diferença entre as trajetórias cresce exponencialmente com o tempo, ou seja, se $|\Delta(t)| = |\Delta(0)|e^{\lambda t}$. Esse comportamento decorre da presença das não-linearidades no sistema. Uma maneira de quantificar a taxa de divergência das trajetórias, e definir a caoticidade do sistema, é analisar os expoentes de Lyapunov (LE, do inglês: Lyapunov Exponents) [105].

Seja um sistema de n equações ordinárias, e uma hiperesfera de condições iniciais centrada em $\vec{x}(t_0)$. Conforme o tempo evolui, esta deve sofrer deformações. Tomando uma direção j, considerando que o raio inicial $d_j(t_0)$ varie exponencialmente no tempo, podemos escrever a relação entre $d_j(t_0)$ e o raio $d_j(t)$, em um tempo posterior t, da seguinte maneira:

$$d_j(t) = d_j(t_0)e^{\lambda_j(t-t_0)}, \tag{3.14}$$

com $j = 1,2,...,n$. Rearranjando temos:

$$\lambda_j = \lim_{d_j(t_0)\to 0} \lim_{t\to\infty} \frac{1}{(t-t_0)} \ln\left(\frac{d_j(t)}{d_j(t_0)}\right), \tag{3.15}$$

sendo os λ_j os LE. Apesar de os números de dimensões e LE coincidirem, esses e os eixos coordenados não estão diretamente relacionados.

A hiperesfera de condições iniciais é formada por um conjunto de vetores ortonormais, no espaço tangente, cujas dimensões correspondem à dimensão do sistema dinâmico. Ao longo do tempo, essa hiperesfera vai sofrendo deformações. Porém, a cada passo, nós mensuramos o quanto esses vetores que a compõem se deformaram, e os ortonormalizamos, para evitarmos que esses se tornem linearmente dependentes. Como apresentado na equação (3.15), os LE estão ligados às direções desses vetores, indicando o quanto essas

perturbações divergem ou convergem em relação à trajetória. Nas direções onde há a convergência da perturbação, o LE associado é negativo, enquanto onde há divergência, o LE é positivo. Os LE nulos indicam que não há convergência e nem divergência na direção em questão.

O volume da hiperesfera num instante $t > t_0$ deve ser proporcional ao produto das distâncias $d_j(t)$ que o caracterizam, isto é:

$$V(t) \propto \prod_{j=1}^{n} d_j(t) = V(t_0) e^{(t-t_0)\sum_{j=1}^{n} \lambda_j}, \tag{3.16}$$

onde $V(t_0)$ é o volume no instante t_0. Em um sistema conservativo, o volume no espaço de fases é conservado, ou seja, $V(t) = V(t_0)$. Isso implica que:

$$\sum_{j=1}^{n} \lambda_j = 0. \tag{3.17}$$

Em sistemas dissipativos, o volume no espaço de fase deve sofrer uma contração, portanto, $V(t) < V(t_0)$ para qualquer $t > t_0$, o que equivale a:

$$\sum_{j=1}^{n} \lambda_j < 0. \tag{3.18}$$

Com o objetivo de classificar os tipos de atratores, pode-se calcular o espectro de Lyapunov. Em um sistema quadridimensional, por exemplo, existem quatro LE. Nesse caso, podem existir os quatro tipo de atratores mencionados na subseção anterior, e mais um, com características semelhantes ao caótico. Ordenando os LE de modo decrescente, podemos relacioná-los com a forma do atrator da seguinte maneira:

- **ponto de equilíbrio**: $\lambda_{1,2,3,4} < 0$, ou seja, haverá uma contração em todas as direções, levando o sistema a convergir em um ponto no espaço de fases.
- **ciclo limite ou periódico**: $\lambda_1 = 0$ e $\lambda_{2,3,4} < 0$, com o expoente nulo correspondendo à direção ao longo da órbita fechada.
- **quase-periódico**: $\lambda_{1,2} = 0$ e $\lambda_{3,4} < 0$, ou $\lambda_{1,2,3} = 0$ e $\lambda_4 < 0$, onde as trajetórias atratoras se encontram sobre uma superfície.
- **caótico**: $\lambda_1 > 0$, $\lambda_2 = 0$ e $\lambda_{3,4} < 0$, ou $\lambda_1 > 0$, $\lambda_{2,3} = 0$ e $\lambda_4 < 0$, dos quais o expoente positivo está associado à direção na qual trajetórias vizinhas divergem

exponencialmente no espaço de fase, após o sistema evoluir no tempo. Por se tratar de um sistema dissipativo, a soma dos expoentes deve ser negativa, portanto, os expoentes negativos somados, em módulo, devem sempre ser maiores do que o expoente positivo.

- **hipercaótico**: $\lambda_{1,2} > 0$, $\lambda_3 = 0$ e $\lambda_4 < 0$. Apesar de distinto no espectro dos expoentes de Lyapunov, não apresenta diferenças notáveis na forma do atrator, se comparado ao caótico. Nesse caso, a soma dos valores dos maiores expoentes não deve ultrapassar o módulo do menor.

A tabela 1 mostra o comportamento acima, representado para os respectivos valores dos LE:

Tabela 1 – Valores dos expoentes de Lyapunov e seus respectivos atratores para um sistema quadridimensional [105].

λ_1	λ_2	λ_3	λ_4	Atratores
−	−	−	−	1 - Ponto de Equilíbrio
0	−	−	−	2 - Periódico
0	0	−	−	3 - Quase-periódico
0	0	0	−	4 - Quase-periódico
+	0	−	−	5 - Caótico
+	0	0	−	6 - Caótico
+	+	0	−	7 - Hipercaótico

Em sistemas de tempo discreto, é possível fazer uma análise das trajetórias, mediante LE de maneira similar. Digamos que, inicialmente, duas condições iniciais x_0 e $x_0 + \delta_0$ em um sistema unidimensional estão separadas por uma distância pequena δ_0, e após N iterações do mapa $x_{j+1} = M(x_j)$, com $N \to \infty$, a distância entre esses é δ_N. A relação entre δ_N e δ_0 pode ser dada da seguinte maneira:

$$|\delta_N| = |\delta_0| e^{\lambda N}, \tag{3.19}$$

onde λ é o . Podemos obtê-lo isolando-o na expressão anterior:

$$\lambda = \lim_{\delta_0 \to 0} \lim_{N \to \infty} \frac{1}{N} \ln \left(\left| \frac{\delta_N}{\delta_0} \right| \right), \tag{3.20}$$

Como a distância δ_N é a distância entre os dois pontos após N iterações, temos que:

$$\delta_N = M^{(N)}(x_0 + \delta_0) - M^{(N)}(x_0). \tag{3.21}$$

Desta maneira, temos que λ é dado por:

$$\lambda = \lim_{\delta_0 \to 0} \lim_{N \to \infty} \frac{1}{N} \ln \left(\left| \frac{M^{(N)}(x_0 + \delta_0) - M^{(N)}(x_0)}{\delta_0} \right| \right). \tag{3.22}$$

Como a distância inicial entre os pontos era muito pequena, a rigor infinitesimal, podemos escrever o argumento do logaritmo natural como a derivada de $M^{(N)}$, calculada em x_0:

$$\lambda = \lim_{N \to \infty} \frac{1}{N} \ln \left| \frac{dM^{(N)}(x)}{dx} \right|_{x=x_0}. \tag{3.23}$$

Para calcular essa derivada, podemos aplicar a regra da cadeia:

$$\frac{dM^{(N)}(x)}{dx}\Big|_{x_0} = \frac{dM(x)}{dx}\Big|_{x_{N-1}} \frac{dM(x)}{dx}\Big|_{x_{N-2}} \dots \frac{dM(x)}{dx}\Big|_{x_0}, \tag{3.24}$$

sendo que $x_k = M^k(x_0)$. Portanto, λ pode ser obtido por:

$$\lambda = \lim_{N \to \infty} \frac{1}{N} \ln \prod_{j=0}^{N-1} \left| \frac{dM(x)}{dx} \right|_{x=x_j}. \tag{3.25}$$

Finalmente, utilizando as propriedades de logaritmo, temos:

$$\lambda = \lim_{N \to \infty} \frac{1}{N} \sum_{j=0}^{N-1} \ln \left| \frac{dM(x)}{dx} \right|_{x=x_j}. \tag{3.26}$$

Se houver N dimensões, haverá N expoentes de Lyapunov associados a direções do espaço tangente.

4. O reservatório, a aprendizagem e os métodos de análise

4.1 O sistema de Kuramoto

Como mencionado no capítulo introdutório, escolhemos os osciladores de Kuramoto para compôr as unidades do reservatório do nosso modelo de PRC.

O estudo sobre sincronização de osciladores acoplados recebeu notoriedade, inicialmente, com os trabalhos de Winfree [107, 108] e Kuramoto [109, 110], apresentando a partir de então aplicabilidade em diversas áreas, tais quais física, química, biologia e medicina [111, 112, 113, 114, 115]. Escolhemos o modelo proposto por Kuramoto devido aos mesmos motivos que, provavelmente, o tornaram tão popular no meio acadêmico: simples o suficiente para ser matematicamente tratável, rico o suficiente para descrever diversos padrões de sincronização, e apresenta flexibilidade que possibilita ser adaptado em diferentes contextos. O modelo original consiste em uma população de N osciladores, acoplados todos com todos, cujas dinâmicas são descritas da seguinte maneira:

$$\dot{\theta}_i = \omega_i + \frac{\kappa}{N}\sum_{j=1}^{N} sin(\theta_j - \theta_i),\ i = 1,2,...,N \tag{4.1}$$

onde θ_i é a fase do $i-ésimo$ oscilador, ω_i é a frequência natural de oscilação do $i-ésimo$ oscilador, κ é a força de acoplamento e j é o índice que indica os osciladores aos quais o $i-ésimo$ se acopla. Desde o trabalho original foram propostas diversas variações, seja na inclusão de ruídos e forçamentos, na distinção das forças de acoplamento para cada oscilador, na arquitetura das conexões da rede, dentre outras. Com o objetivo de examinar a eficiência de diferentes configurações de rede, nós optamos inicialmente pelo acoplamento conhecido por "vizinhos mais próximos", ou seja, limitamos as conexões até K osciladores próximos considerando ambas as direções. Vale esclarecermos que apesar de o nome ser semelhante, estamos nos referindo a um tipo de ligações entre redes neurais, e não do método k-NN mencionado no capítulo 3. Dessa maneira, a dinâmica pode ser representada do seguinte modo:

$$\dot{\theta}_i = \omega_i + \frac{\kappa}{2K}\sum_{j=1}^{N} A_{ij} sin(\theta_j - \theta_i),\ i = 1,2,...,N \tag{4.2}$$

onde a matriz A_{ij} representa a matriz de adjacência, a qual contém as informações sobre

as conexões, de maneira análoga à matriz W no caso discreto exemplificado na equação (3.4). No nosso estudo, preliminarmente, o valor na matriz é 1 onde $|i-j| \leq K$, e 0, caso contrário.

Com o intuito de aumentar a diversidade e a variabilidade dos valores de saída, adicionamos à equação (4.2) um termo periódico do tipo cosseno, cujo argumento é a diferença entre as fases dos osciladores e suas médias, individualmente, da seguinte maneira:

$$\dot{\theta}_i = \omega_i + \frac{\kappa}{2K}\sum_{j=1}^{N} A_{ij} sin(\theta_j - \theta_i) + \beta \cos\left[\zeta\left(\theta_i - \sum_{j=1}^{N}\frac{\theta_j}{N}\right) - \gamma\right],\ i = 1,2,...,N \qquad (4.3)$$

onde β, ζ e γ são parâmetros do sistema.

Para melhor compreensão do modelo, na figura 7 apresentamos um grafo do fluxo de sinal, representando as ligações entre os nós do reservatório com a saída y, e dos nós entre si. A ligação entre as N unidades da rede no reservatório é a de K vizinhos mais próximos, e nessa representação $K = 2$.

Recapitulando o que foi explicado no capítulo 4, as FNN's possuem entradas externas, as quais alimentam a(s) camada(s) oculta(s) (caso estejam presentes no modelo), e resultam em um ou mais valores de saída. No caso das RNN's, há uma retroalimentação das unidades das camadas situadas entre as entradas e a(s) saída(s), onde os valores dessas unidades dependem das entradas e dos seus próprios valores anteriores, podendo assim serem caracterizadas como sistemas dinâmicos. Para aumentar a eficiência e reduzir o custo de aprendizagem, foram propostos os RC's, onde somente os pesos de saída são atualizados.

Neste trabalho utilizamos um modelo de PRC com osciladores de Kuramoto em suas unidades. Porém, como comentamos no capítulo introdutório, optamos por não incluir entradas externas às computações, o que nos permite utilizar o mesmo reservatório para a realização de diversas tarefas, alterando somente os parâmetros e os pesos entre os nós do reservatório e a saída. Isso é possível devido ao fato de os valores das unidades do reservatório serem alimentados pelos seus próprios valores em tempos anteriores, relação determinada pelo modelo de conexões internas, não havendo necessidade de entradas externas.

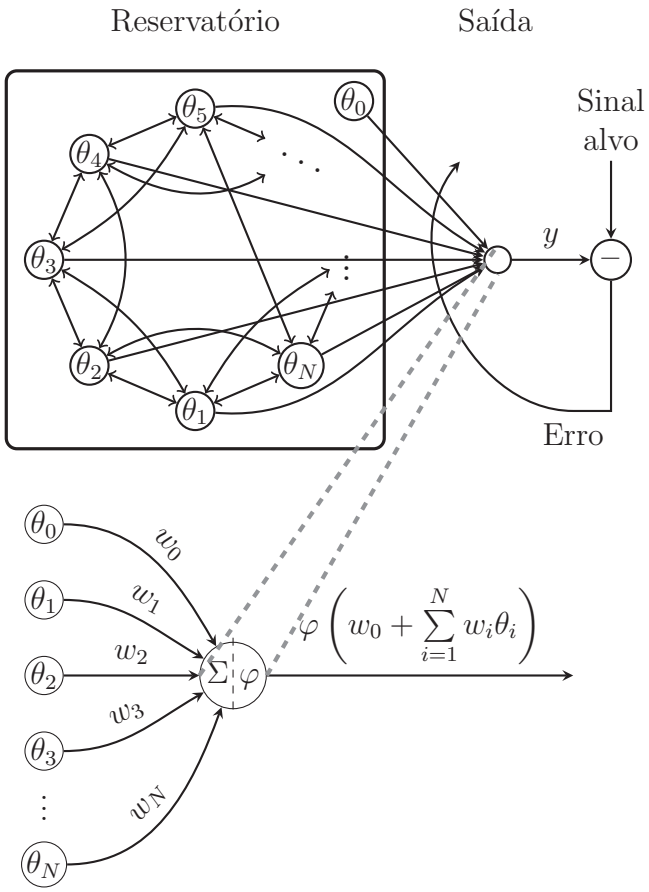


Figura 7 – Grafo do fluxo de sinal entre os nós das unidades do reservatório, cujos valores de suas unidades somados de maneira ponderada resultam no valor de saída y, o qual é comparado ao sinal alvo para que sejam realizadas as atualizações dos pesos de saída $w_0, ..., w_N$. A ligação entre as N unidades da rede no reservatório é K vizinhos mais próximos, onde $K = 2$. O bias é sempre $\theta_0 = 1$.

4.2 Sobre a aprendizagem

Utilizamos o método Runge-Kutta de Quarta Ordem para integrar o sistema, com passos de tamanho 10^{-2} e transiente 10^4, e extraímos as fases θ_i dos osciladores a cada 8 passos, as quais representam os estados dos neurônios da rede, novamente em analogia aos componentes dos vetores $\vec{x}$ na equação (3.4). Sobre esses valores é realizada a soma ponderada que resulta na saída, pois se fossem tomadas as fases a cada passo de integração haveria pouca variação nesses valores, tornando mais difícil a aprendizagem. Portanto, ao decorrer deste trabalho, quando mencionamos iterações estamos nos referindo a cada vez que extraímos as fases, e não ao passo dado pelo integrador. Escolhemos a distribuição Lorentziana para as frequências naturais, como é comumente utilizada na literatura que aborda osciladores de Kuramoto. Além disso, diferentemente do que é comum nos estudos da área de redes neurais, aqui não consideramos necessárias as funções de ativação para cada unidade do reservatório, devido ao fato de os estados do sistema serem representados pelas fases dos osciladores, os quais estão sempre limitados entre 0 e 2π.

Para a saída, optamos pela função de ativação do tipo degrau unitário:

$$y(\tau) = \varphi(W_{out}\vec{\theta}(\tau)), \tag{4.4}$$

tal que:

$$\varphi(W_{out}\vec{\theta}(\tau)) = \begin{cases} 1, & \text{se } W_{out}\vec{\theta}(\tau) \geq 0 \\ 0, & \text{caso contrário.} \end{cases} \tag{4.5}$$

onde W_{out} é a matriz que contém os pesos de conexões entre as unidades do reservatório, cujos estados são representados por $\vec{\theta}(\tau)$ e a saída y.

Apesar dos osciladores terem a dinâmica descrita por um fluxo, cujo tempo t varia continuamente, representamos o tempo τ = 1,2,... de maneira discreta, onde seu valor indica a ordem que foram tomados os valores das fases, ou seja, as iterações, como definimos anteriormente.

Nos casos em que a série de referência não apresenta somente dois valores nós dividimos o seu intervalo em seções, e após representá-las em números binários, realizamos o aprendizado bit por bit. Para a atualização dos pesos $w_0, w_1, w_2, ..., w_N$, componentes da matriz W_{out}, nos baseamos no método do gradiente descendente. Nesse método, mediante a utilização do operador matemático gradiente, se busca a minimização de uma função custo, a qual é uma medida de desempenho de aprendizagem. No nosso caso, a cada iteração, calculamos a função custo da seguinte maneira:

$$J(\tau, w_0, w_1, \ldots, w_N) = \frac{1}{2}(d(\tau) - y(\tau))^2, \tag{4.6}$$

onde $d(\tau)$ é o valor desejado, obtido da série de referência, e $y(\tau)$ a saída gerada pelo modelo. Os pesos foram atualizados mediante acréscimo da derivada da função custo, em relação a si mesmos, da seguinte maneira:

$$w_i(\tau + 1) = w_i(\tau) + \alpha(d(\tau) - y(\tau))\theta_i(\tau), \tag{4.7}$$

onde:

$$\frac{\partial}{\partial w_i} J(\tau, W_{out}) = (d(\tau) - y(\tau))\theta_i(\tau), \tag{4.8}$$

e α é a constante de convergência do gradiente, a qual mantivemos $\alpha = 1$ para todas as nossas simulações.

A partir da série de referência, tomamos T amostras de treinamento para que os pesos de saída sejam atualizados, sendo permitida a repetição do uso dessas para a

continuidade da aprendizagem. Cada vez que repetimos esse procedimento com o mesmo conjunto de amostras, denominamos que se passou uma época. Após a fase de aprendizagem, utilizamos V dados de validação para verificar se, a partir dos pesos aprendidos, há uma coerência entre a extrapolação do nosso programa e a série a ser aprendida. Essa mensuração pode ser feita mediante comparação entre os valores esperados e os obtidos, ou a partir de outros aspectos, como a manutenção da distribuição das seções, isso é, do percentual de valores pertencentes a cada seção. Neste trabalho, escolhemos $T = 200$, no intervalo de 1 a 200, e $V = 200$, no intervalo de 201 a 400. Da amostra 401 em diante consideramos intervalo de teste.

Testamos o desempenho do nosso modelo na emulação e predição, ou manutenção de distribuição de séries de diferentes formas. O primeiro tipo trata-se de uma série binária, gerada aleatoriamente, utilizando a função *RANDOM_NUMBER* do Fortran. Essa função gera uma sequência de números pseudoaleatórios dentro do intervalo [0,1[. Para valores maiores ou iguais a um número limite que escolhemos entre 0 e 1, atribuíamos 1, e 0, caso contrário. Essa mesma técnica utilizamos em uma série cujos valores, em vez de serem gerados aleatoriamente, são provenientes dos estados de um mapa logístico, visto que também pertencem ao intervalo [0,1[, desde que haja uma escolha paramétrica adequada. Testamos também o aprendizado de uma série dividida em 16 seções, a qual segue, aproximadamente, uma distribuição gaussiana. Dividimos, também em 16 seções, séries geradas por funções trigonométricas. Por fim, geramos uma distribuição exponencial negativa que simula a distribuição de uma série experimental gerada por saltos quânticos.

Escolhemos dois métodos distintos para avaliar o desempenho. Um deles, leva em consideração a capacidade do modelo em emulação e predição da série de referência na etapa de validação, o qual aplicamos para a série binária logística e as séries trigonométricas. Para o caso da série binária gerada aleatoriamente e da série gaussiana, devido ao fato de não seguirem uma equação bem definida, avaliamos a manutenção da distribuição de cada seção em relação à série de referência. Aplicamos esse último método, também, para a aprendizagem da simulação da série experimental gerada por saltos quânticos.

O primeiro método mencionado avalia dois aspectos: o número mínimo de épocas utilizado para emular completamente a série em questão, no intervalo de T amostras, e a quantidade de erros na predição, bit por bit, V valores após as T amostras de treinamento. No caso da série binária logística, basta verificar se a saída errou ou acertou o valor. Para exemplificar essa etapa, na tabela 2 apresentamos uma amostra com 5 valores fictícios, os quais supomos pertencerem ao intervalo de validação V, e a saída preditiva gerada pelo PRC. A terceira coluna da tabela contém o erro associado a essa predição, o qual é 1, caso haja discrepância entre a saída do PRC e a série de referência, e 0, caso contrário.

No caso das séries trigonométricas, as quais são divididas em mais seções, comparamos a saída e a série de referência, bit por bit, representadas em números binários,

Tabela 2 – Contagem de bits preditos incorretamente pela saída do PRC. Nesta tabela utilizamos 5 amostras ilustrativas para exemplo.

Referência	**Saída**	**Contagem**
0	0	0
1	0	1
1	1	0
0	1	1
1	0	0

e contamos quantos bits foram previstos incorretamente. Na tabela 3, assim como na tabela 2, propomos amostras de uma série de referência fictícia pertencentes ao intervalo de validação, porém com essa série dividida em 8 seções. Apresentamos, nas duas primeiras colunas, a série de referência representada em números inteiro e binário, respectivamente, na terceira coluna, a saída gerada pelo PRC (em representação binária), e na quarta coluna, a contagem de bits discrepantes entre a série de referência e a saída preditiva. Nesse caso, comparamos os bits individualmente, de modo que sua posição importa. Como podemos observar na quarta linha, se a ordem dos bits não importasse, a contagem do erro deveria ser 0, porém, como podemos constatar na quarta coluna, essa contagem marca o valor 2, indicando que o primeiro e o terceiro bit foram preditos incorretamente.

Tabela 3 – Contagem de bits preditos incorretamente pela saída do PRC, a qual, assim como a série de referência, está dividida em 8 seções. Nesta tabela utilizamos 5 amostras ilustrativas para exemplo.

Ref. Inteiro	**Ref. Binário**	**Saída**	**Contagem**
5	101	101	0
7	111	011	1
0	000	000	0
4	100	001	2
5	101	010	3

Finalmente, realizamos uma soma ponderada e normalizada de ambos os aspectos como uma forma associar a uma medida de erro, atribuindo 10% ao primeiro e 90% ao segundo. A equação para esse cálculo, portanto, tem a seguinte forma:

$$\text{Erro} = \frac{\sqrt{\left(Peso_{Ap} \cdot \frac{Ep_{Ap}}{Ep_{Máx}}\right)^2 + \left(Peso_{Pre} \cdot Er_{Pre}\right)^2}}{\sqrt{Peso_{Ap}^2 + Peso_{Pre}^2}}, \tag{4.9}$$

onde $Peso_{Ap} = 0{,}1$ e $Peso_{Pre} = 0{,}9$ são os pesos para o primeiro e o segundo termo na soma ponderada, relacionados às etapas de treinamento e predição, respectivamente. O número de épocas que o PRC utiliza para o aprendizado completo do intervalo de

treinamento, da série de referência, é dado por Ep_{Ap}, enquanto o número máximo de épocas que estipulamos para essa etapa é $Ep_{Máx} = 200$. O termo Er_{Pre} é a fração de bits preditos incorretamente, no intervalo de validação, da série de referência.

Já o segundo método avalia três aspectos: o primeiro é o mesmo do número de épocas para a emulação completa, como no método anterior; o segundo é a manutenção da quantidade de valores dentro de cada seção da série de referência (comparamos o intervalo de treinamento da série de referência com o intervalo de validação da saída do modelo); o terceiro olha para as sequências de dois valores consecutivos apresentadas na série de referência (mesmos intervalos do aspecto anterior). Esse último evita, por exemplo, que o modelo apresente uma série binária com metade dos valores consecutivos 0 e a outra metade 1, sendo que a série de referência contém essa mesma quantidade de valores, mas dispostos de maneira alternada, e obtenha um bom desempenho. Semelhantemente ao método anterior, realizamos a soma ponderada e normalizada do erro associado aos três aspectos, atribuindo 10% ao primeiro, 60% ao segundo e 30% ao terceiro. Realizamos esses cálculos da seguinte maneira:

$$\text{Erro} = \frac{\sqrt{\left(Peso_{Ap} \cdot \frac{Ep_{Ap}}{Ep_{Máx}}\right)^2 + (Peso_{PreS} \cdot Er_{PreS})^2 + (Peso_{PreD} \cdot Er_{PreD})^2}}{\sqrt{Peso_{Ap}^2 + Peso_{PreS}^2 + Peso_{PreD}^2}}, \quad (4.10)$$

onde $Peso_{Ap} = 0{,}1$, $Peso_{PreS} = 0{,}6$ e $Peso_{PreD} = 0{,}3$ são os pesos para o primeiro, o segundo e o terceiro termo na soma ponderada. Os termos Ep_{Ap} e Er_{Pre} possuem significados equivalentes ao método anterior, entretanto, dividimos a avaliação de erro da etapa de predição em duas partes. Para definir Er_{PreS}, contamos a quantidade de elementos em cada classe, tanto na série de referência (intervalo de 0 a 200) quanto na saída do PRC (intervalo de 201 a 400), tomamos a diferença entre essas contagens, e dividimos pelo número de elementos comparados. De maneira análoga, definimos o termo Er_{PreD}, entretanto, em vez de contarmos a quantidades de elementos em cada classe, contamos as sequências de dois valores consecutivos. As letras S e D ao fim de Er_{PreS} e Er_{PreD} remetem a simples e dupla, respectivamente.

4.3 Algoritmo Genético

Para selecionarmos um conjunto de parâmetros que leve o sistema a apresentar um bom desempenho, dentro de intervalos de parâmetros pré estabelecidos, utilizamos o que é conhecido como algoritmo genético, o qual é técnica inspirada na biologia evolutiva que apresenta aspectos como hereditariedade, mutação, seleção natural e recombinação [24, 25, 26]. No nosso modelo, utilizamos esse método na escolha de conjuntos para os

parâmetros κ, β, ζ e γ. Inicialmente, geramos dez conjuntos de parâmetros de maneira aleatória, evoluímos o sistema e realizamos o processo de aprendizagem. A cada exemplar desses, denominamos uma espécime. Para gerar outras 40 espécimes, sorteamos pares de valores entre um e dez, com o intuito de determinar os genitores de cada uma dessas. Comparamos então os desempenhos das 50 espécimes, selecionamos as 10 melhores e repetimos esse procedimento por G gerações. O processo de recombinação foi realizado para cada um dos quatro parâmetros. Para isso, convertemos cada parâmetro de cada um dos genitores, de um número real, com precisão dupla, para inteiro, dentro do intervalo de 0 a $2^{15} - 1$, representado em números binários ($2^{15} - 1$ é o valor máximo representado, no Fortran, por um número inteiro do tipo 2). Em seguida, sorteamos um valor entre 2 e 15, que seria o bit onde ocorre o corte, isso é, até esse bit um genitor doa os seus valores, e a partir desse, o outro genitor é o doador. Após isso, para evitar possíveis mínimos locais, para cada parâmetro, há uma probabilidade de 50% de que ocorra uma mutação, isso é, um dos bits que representa determinado parâmetro tenha seu valor invertido (se era 0, passa a ser 1, e vice e versa). Por fim, convertemos esses valores novamente para precisão dupla, e repetimos esse procedimento em todas as gerações posteriores.

Utilizamos o GA também para selecionar as condições iniciais e as frequências naturais de oscilação[1] para as novas espécimes, porém com o processo de recombinação genética um pouco diferente. Neste caso, optamos por um sorteio entre 2 a N, o qual representa o número de osciladores, para acontecer o corte. Até o corte um dos genitores doa seus valores, e o outro genitor, o restante.

Tendo em vista que não especificamos a semente, cada vez que inicializado novamente o *RANDOM_NUMBER* – presente em algumas partes do código – gerou novos valores baseados no relógio do sistema do computador. Nós rodamos o mesmo código 50 vezes para cada tarefa. No capítulo 5, analisamos a dinâmica do reservatório com os conjuntos de parâmetros que levaram o nosso modelo de PRC a obter os melhores desempenhos.

Na seção de apêndices, apresentamos um pseudocódigo explicando de maneira mais detalhada as etapas do GA.

4.4 Investigando a dinâmica

Por fim, escolhemos os métodos para investigação da dinâmica da rede neural que compõe o reservatório, e de quais aspectos exercem influência sobre o desempenho do PRC na realização das tarefas.

Para observar a sincronização da rede, evidenciamos a evolução temporal dos

[1] Mantivemos a distribuição aproximadamente Lonrentziana, porém, podendo variar os limites inferior e superior

osciladores, a cada iteração, durante 100 iterações, como apresentado na figura 12, por exemplo. Com esse mesmo objetivo, geramos outra figura com as fases de todos os osciladores, em uma iteração específica, com uma representação polar em um círculo unitário, incluindo o parâmetro de ordem [116, 117, 118], como pode-se notar na figura 13 e nas semelhantes a essa. O parâmetro de ordem é um modo de quantificar o grau de sincronização da rede, o qual possui a seguinte forma:

$$re^{i\Phi} = \frac{1}{N}\sum_{m=1}^{N} e^{i\theta_m}, \tag{4.11}$$

onde Φ é a fase média, θ_m é a fase do $m-ésimo$ oscilador, e r, conhecido como parâmetro de ordem, representa a uniformidade, ou a divergência, das fases da rede. Quando os osciladores estão alinhados, o valor de r tende a 1, e a 0, quando as fases estão distribuídas uniformemente. Neste trabalho, calculamos o parâmetro de ordem a cada passo de integração, por 10^5 passos, e tomamos sua média. Além disso, observamos também o espectro dos expoentes de Lyapunov, o qual foi abordado no capítulo anterior, com o intuito de caracterizar a dinâmica do nosso modelo quanto à sua caoticidade. Os três painéis da figura 14 são exemplos de espectros de expoentes de Lyapunov, ou simplesmente espectros de Lyapunov.

5. Resultados e discussão

Neste capítulo, apresentamos e discutimos os resultados da aplicação dos métodos descritos no capítulo anterior.

Como mencionado anteriormente, escolhemos séries de referência divididas em duas ou mais seções, cujas naturezas determinaram qual método de avaliação de erros aplicamos. Para as séries que foram geradas a partir de equações sem termos de aleatoriedade, como no caso da binária logística e das trigonométricas, nós avaliamos a capacidade do PRC de reproduzir a série de referência, bit por bit, no trecho de validação. Contudo, quando há aleatoriedade na geração das séries como nos casos das binárias aleatórias e da gaussiana, nosso interesse é saber o quão bem o PRC consegue produzir uma saída que mantenha, em seu trecho de validação, a distribuição (quantidade de elementos em cada classe) do trecho de treinamento da série de referência.

Na figura 8, apresentamos à esquerda um resumo da análise realizada para cada série de referência, e à direita um fluxograma contendo o procedimento adotado pelo algoritmo genético. Essa figura resume o que foi descrito no capítulo 4. Os retângulos vermelhos com bordas arredondadas correspondem ao início e ao fim da análise para cada série de referência, os trapézios azuis indicam as informações iniciais e os resultados finais de cada ciclo, os retângulos laranjas são os processos intermediários e os losangos verdes são as decisões que nos guiam através da sequência de processos. Os blocos à direita apresentam tons mais claros para diferenciá-los dos demais.

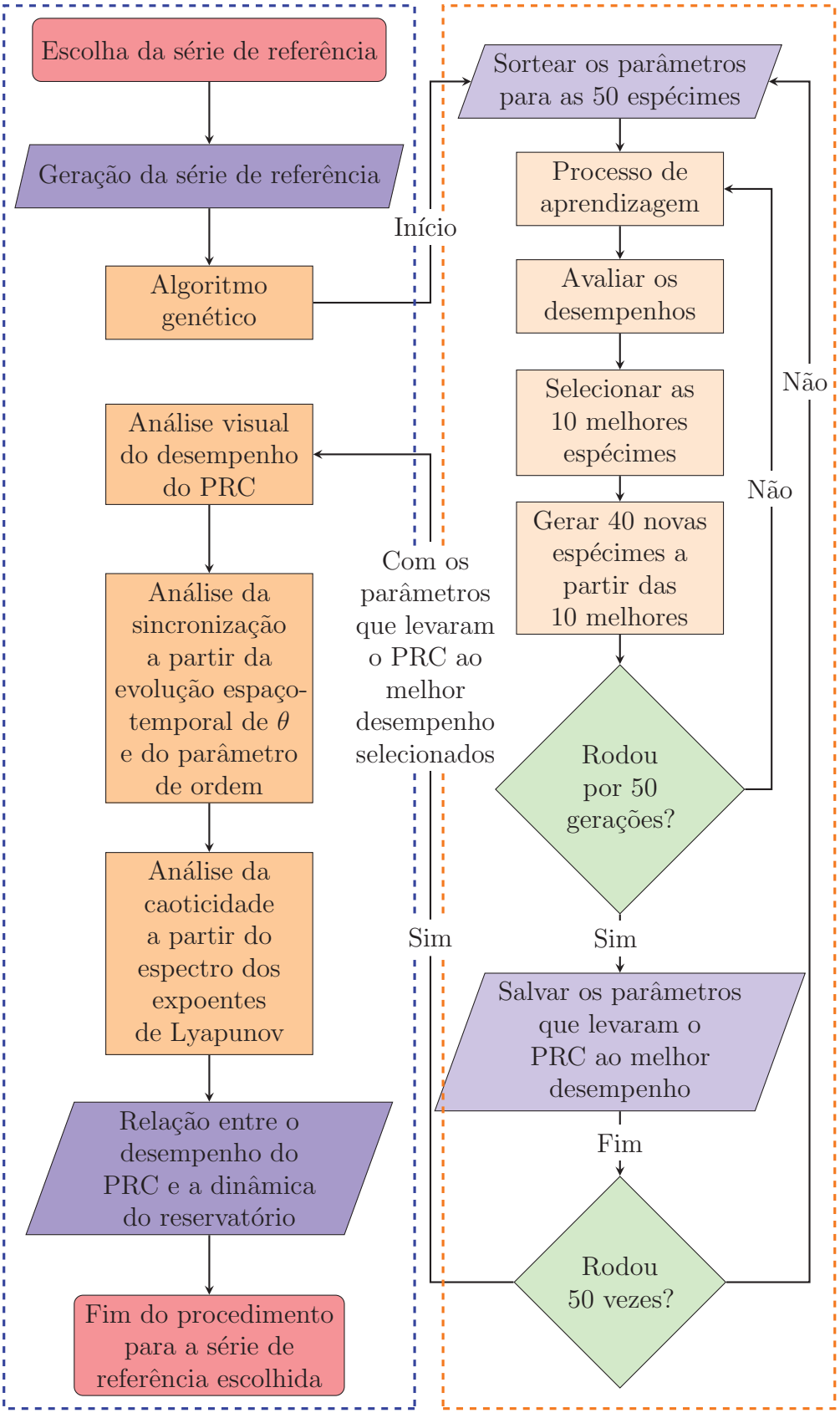


Figura 8 – À esquerda, dentro do retângulo tracejado azul, um resumo da análise realizada para cada série de referência. À direita, dentro do quadro laranja, um fluxograma do procedimento adotado pelo algoritmo genético.

5.1 Séries binárias aleatórias

Inicialmente, nós geramos séries binárias usando a função *RANDOM_NUMBER* do Fortran, sem especificar a semente. Nós fizemos de uma maneira que as séries S apresentam 1, caso o número sorteado R for maior ou igual a um valor limite l pré-determinado, e 0, caso contrário:

$$S(i) = \begin{cases} 1, & \text{se } R \geq l, \\ 0, & \text{caso contrário,} \end{cases} \tag{5.1}$$

onde i é o $i-ésimo$ termo da série de referência, e $\{R \in \mathbb{R} \mid 0 \leq R < 1\}$. A escolha do valor limite R foi realizada de acordo com a distribuição desejada, isso é, conforme a proporção de elementos que queremos em cada seção. Por exemplo, se desejamos que aproximadamente 20% dos elementos da série de referência sejam pertencentes à seção 1, devemos escolher o valor limite $l = 0{,}8$, para aproximadamente 50%, $l = 0{,}5$, para aproximadamente 80%, $l = 0{,}2$, e assim por diante.

Na figura 9, apresentamos as séries cujos valores limites são 0,2, 0,5 e 0,8. Assim como foi explicado, isso corresponde a distribuições de aproximadamente 20%, 50% e 80%, respetivamente. Em linhas vermelhas tracejadas observamos a série de referência, enquanto a saída gerada pelo PRC é representada em linhas cheias verdes. Uma vez que tratam-se de séries binárias, as seções são representadas somente por 0 e 1. Como podemos observar no painel 9(a), as séries se mantêm por mais iterações no 0 do que no 1, pois espera-se que sua distribuição (proporção de valores igual a 1) seja algo em torno de 20%. De maneira análoga, no painel 9(c) ocorre o inverso, e no painel 9(b) há um equilíbrio entre as duas seções.

Para cada um dos três casos, nós aplicamos os procedimentos do algoritmo genético, descritos no capítulo 4 e apresentados na parte direita da figura 8, para selecionarmos um conjunto de parâmetros que leve o sistema ao melhor desempenho dentre as variações ocorridas nesse processo. Na tabela 4, nós apresentamos o conjunto de parâmetros selecionado para cada uma das séries de referência, assim como os erros medidos.

Tabela 4 – Parâmetros selecionados pelo GA e os erros resultantes.

Distribuições	κ	β	γ	ζ	**Erro**
20%	0,8103	1,2552	4,7674	0,3588	0,008408
50%	0,3866	1,7340	5,0040	0,4485	0,015399
80%	1,4674	1,5198	4,3417	0,6502	0,006997

Para tornar o modo de medição de erro mais evidente no caso da distribuição de 20%, por exemplo, notemos que o número de épocas suficiente para a emulação completa da série, nas $T = 200$ amostras de treinamento, foi igual a 11. Quanto ao trecho de validação

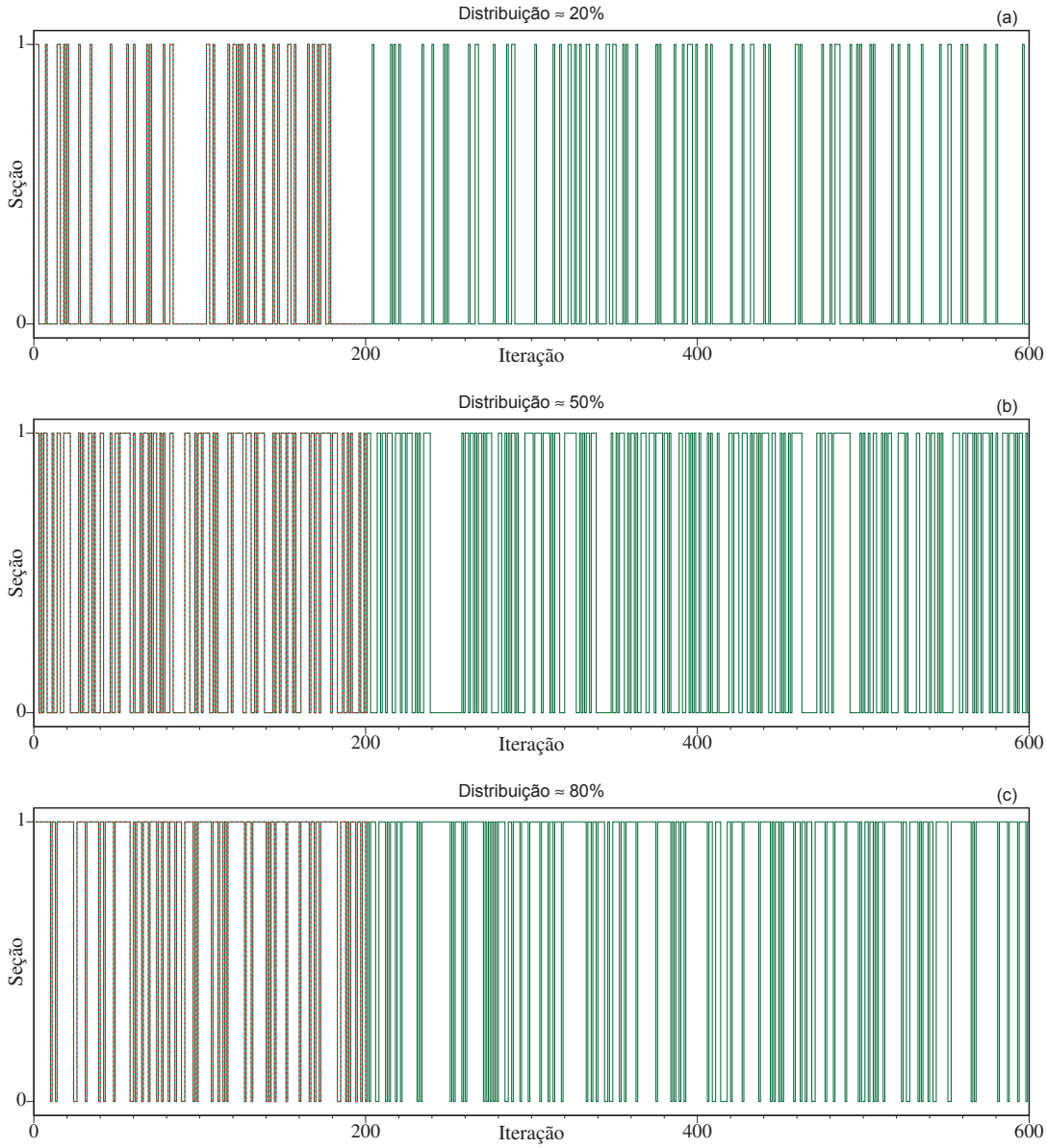


Figura 9 – Série temporal com distribuições aproximadas de 20%(a), 50%(b) and 80%(c). Linhas vermelhas tracejadas representam a série de referência, enquanto as linhas verdes, a saída do PRC.

(entre as amostras 200 e 400), o número de vezes que os valores 0 e 1 apareceram foi 160 e 40, respectivamente, relativos tanto à série de referência quanto ao sinal de saída do modelo, portanto não houve diferença entre ambas. Para as sequências de dois valores consecutivos, nós apresentamos na tabela 5 todas as possíveis sequências com o número de vezes que essas aparecem na série e na saída, assim como a diferença entre ambas. Esses três aspectos levaram nosso modelo a apresentar um erro de 0,008408.

A figura 10 apresenta a habilidade do nosso modelo em manter a distribuição do trecho de treinamento da série de referência. As linhas de baixo correspondem à distribuição aproximada (vezes que o valor 1 aparece) de 20%, as linhas do meio à de 50%, e as de

Tabela 5 – Comparação de quantas vezes as sequências de dois valores consecutivos aparecem na série de referência (segunda coluna) e no sinal de saída do modelo (terceira coluna).

Sequência	**Referência**	**Saída**	**Diferença**
00	128	127	1
01	31	32	1
10	32	32	0
11	8	8	0

cima à de 80%. As linhas tracejadas, em vermelho, correspondem à distribuição que a série de referência apresenta em suas primeiras 200 amostras. As linhas cheias verdes são relacionadas à distribuição da saída do PRC, do seu primeiro elemento até a iteração indicada na figura. De maneira semelhante à anterior, as linhas cheias roxas e azuis também são obtidas a partir do sinal de saída, mas iniciam, respectivamente, na iteração 201 – a primeira após a última amostra de treinamento, e considerando somente blocos de 200 elementos anteriores à iteração indicada na figura. As cores de fundo destacam as seções: salmão, referindo ao intervalo de validação, e ciano, relacionado ao intervalo de teste, isso é, ao trecho onde não há nem treinamento da rede e nem validação das escolhas dos parâmetros realizada pelo GA. Nós observamos que, em todos os casos, nosso modelo foi eficiente em manter a distribuição das séries de referência.

Para verificarmos a importância do termo de campo médio, testamos a eficiência do nosso modelo com os β obtidos pelo GA e com $\beta = 0{,}000000$ (correspondente a um termo de campo médio nulo), mantendo os demais parâmetros. Observamos que, para as três distribuições, o erro é muito maior sem a presença do termo de campo médio (tabela 6).

Tabela 6 – Erros apresentados pelo modelo ao utilizar o parâmetro β obtido pelo AG (segunda coluna), e com $\beta = 0{,}000000$ (terceira coluna).

Distribuições	β **selecionado pelo GA**	$\beta = 0{,}000000$
20%	0,008408	0,174876
50%	0,015400	0,124965
80%	0,006997	0,119073

Testamos, também, o desempenho do PRC com diferentes valores para a força de acoplamento κ (tabela 7), mantendo os demais parâmetros de acordo com a seleção do GA, no caso da distribuição aproximada de 20%. Notamos que o GA se mostrou eficiente na escolha dos parâmetros, mais especificamente falando do κ neste caso, pois é o que leva o PRC a apresentar um bom desempenho se comparado a outros κ.

Na figura 11, apresentamos o erro do nosso modelo para várias forças de aco-

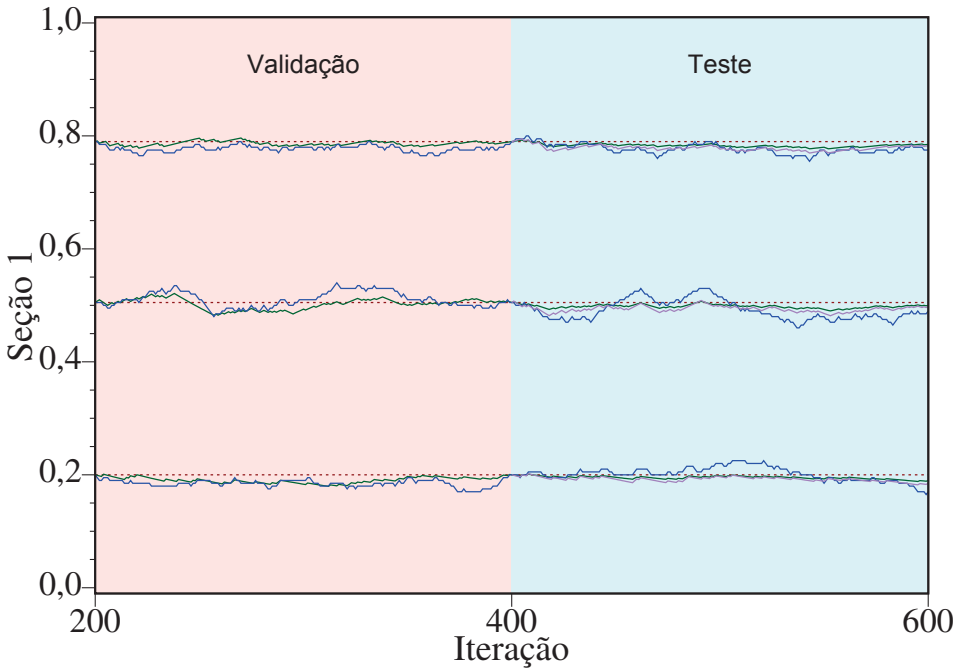


Figura 10 – Distribuições ao decorrer da série temporal. As linhas tracejadas vermelhas correspondem à distribuição da série de referência (vezes que o valor 1 aparece) no intervalo de treinamento (primeiras 200 amostras). As linhas cheias verdes estão relacionadas à distribuição da saída do PRC, do primeiro elemento até a iteração indicada. As linhas cheias roxas e azuis também são relativas à saída do PRC, mas as roxas começam a contagem na iteração 201 e as azuis consideram somente blocos dos 200 elementos anteriores a cada iteração.

Tabela 7 – Erros apresentados pelo modelo para diferentes forças de acoplamento.

κ	**Erro**
0,0000	0,020789
0,0500	0,031306
0,1000	0,020789
0,5000	0,020487
0,8103	0,008408
1,0000	0,062908
2,0000	0,032930
5,0000	0,104470

plamento, mantendo os outros parâmetros selecionados pelo GA para cada uma das distribuições. A interseção das linhas tracejadas vermelhas representa o κ selecionado pelo GA e o erro associado. Nos três painéis, o melhor desempenho é obtido pelo κ selecionado. Portanto, podemos observar que o GA é eficiente em escolher um bom valor de κ, isso é, que leve o PRC a apresentar um bom desempenho se comparado aos outros κ dentro do intervalo de variação proposto.

Em termos de sincronização, temos na figura 12 a evolução espaço-temporal das

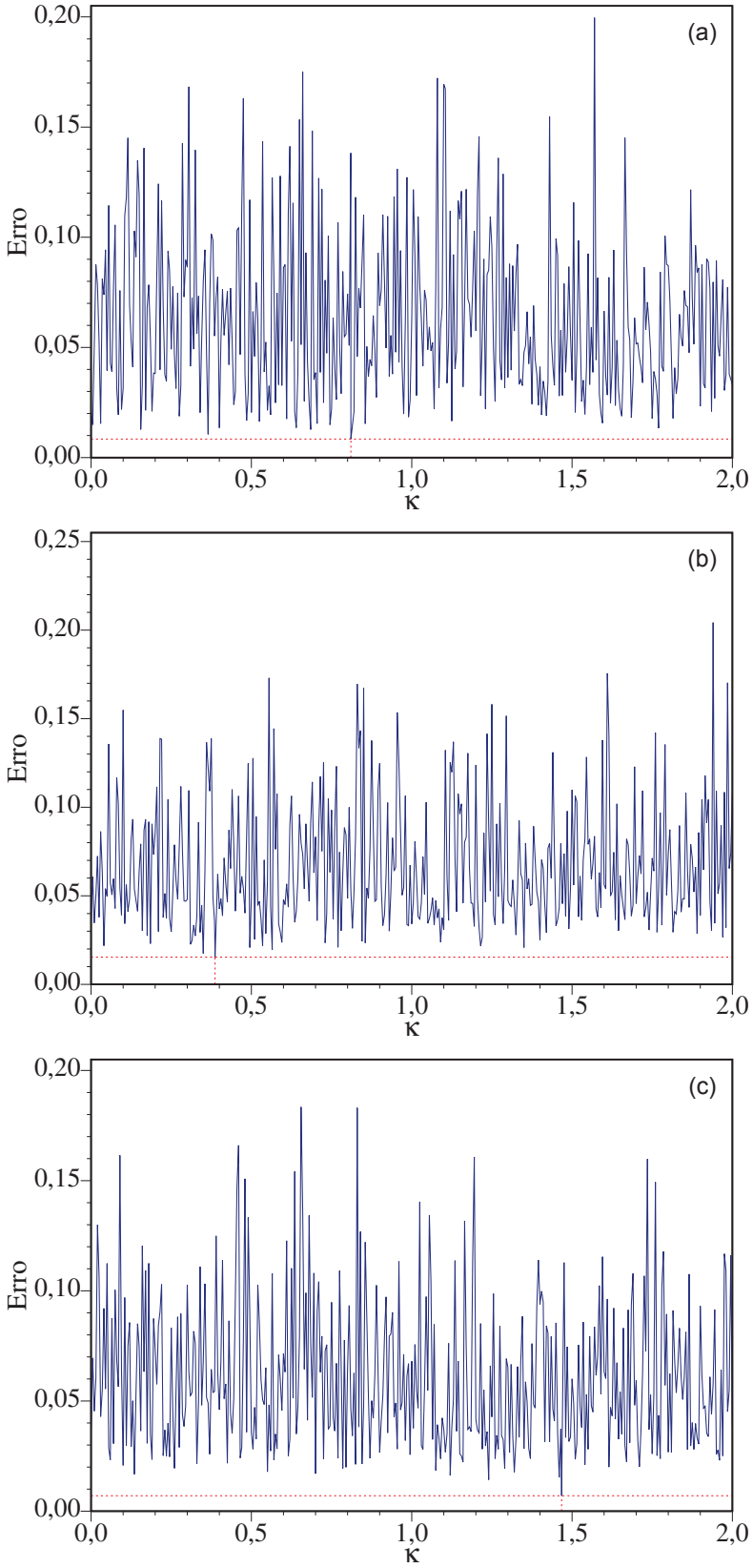


Figura 11 – Erros do PRC ao variarmos os valores da força de acoplamento κ, entre 0,000 e 2,000, com passos de 5×10^{-3}, para as distribuições de 20%(a), 50%(b) e 80%(c).

fases dos osciladores em 100 iterações. Os painéis 12(a) e 12(b) foram gerados com os parâmetros selecionados para a distribuição de 20%, 12(c) e 12(d) para a distribuição de 50%, e 12(e) e 12(f) para de 80%. Nos painéis à esquerda (12(a), 12(c) e 12(e)), ordenamos os osciladores de acordo com suas posições na rede, enquanto nos painéis da direita (12(b), 12(d) e 12(f)), de acordo com as suas frequências naturais de oscilação ω. As fases foram representadas pelas cores, em graus, para melhor visualização, porém foram manipuladas nos códigos em radianos. Tendo em vista que as frequências naturais foram sorteadas de acordo com uma distribuição Lorentziana centrada em zero, os osciladores centrais dos painéis da direita têm ω próximo a 0, enquanto os de baixo e os de cima representam as extremidades negativa e positiva, respectivamente. Nos três casos, notamos comportamentos semelhantes. Os osciladores que possuem ω próximo a 0 mantém suas fases próximas a 0°, aparentemente estando mais acoplados do que os que apresentam ω maiores, em módulo, os quais apresentam dinâmicas mais heterogêneas.

Para quantificar a sincronização entre os osciladores, obtivemos o parâmetro de ordem r apresentado pela rede, com os parâmetros selecionados para cada uma das distribuições. Na figura 13, podemos observar as fases dos osciladores em uma determinada iteração, representadas em pontos roxos sobre o círculo unitário externo, e em verde a média de todas as fases sobre o círculo interno cujo raio representa o parâmetro de ordem r. Como havíamos observado, a maioria dos osciladores encontram-se próximos a 0°. Os painéis 13(a), 13(b) e 13(c), respectivamente, estão associados aos parâmetros selecionados pelo GA para as três distribuições (20%, 50% e 80%). O parâmetro de ordem, para os três casos, são os seguintes: $r = 0{,}81$ (13(a)), $r = 0{,}84$ (13(b)) e $r = 0{,}85$ (13(c)). Isso demonstra que, apesar de os osciladores não estarem totalmente acoplados ($r = 1{,}00$), a rede apresenta alto grau de sincronização.

Quanto à caoticidade, apresentamos na figura 14 o espectro dos expoentes de Lyapunov (λ). Em todos os casos notamos que há mais de um LE positivo, caracterizando a dinâmica da rede de osciladores, para todos os três conjuntos de parâmetros, como hipercaótica. Podemos notar, também, que somente alguns LE estão próximos a zero, enquanto a maioria desses são negativos.

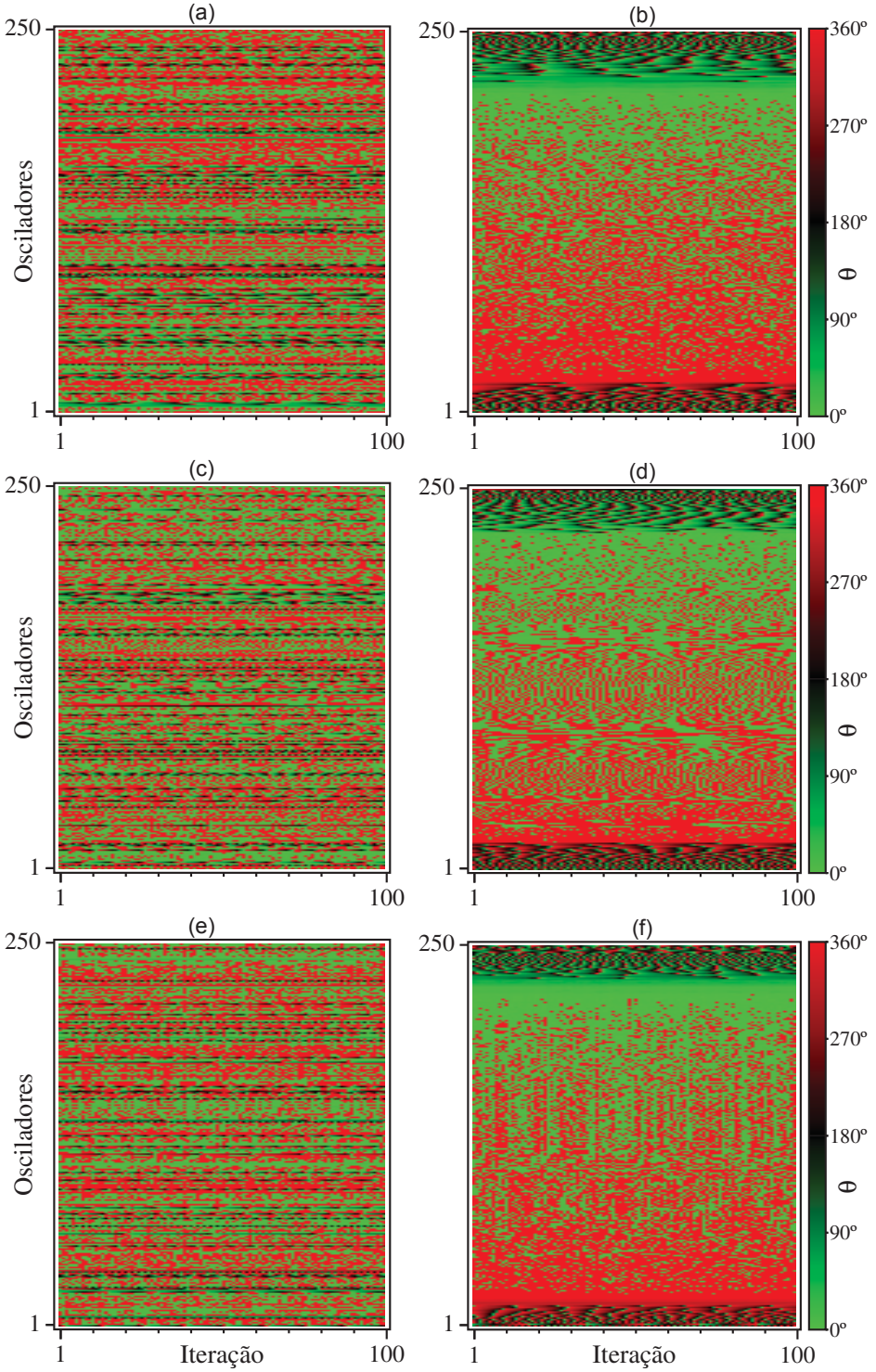


Figura 12 – Evolução espaço-temporal de θ para as distribuições de 20% [(a) e (b)], 50% [(c) e (d)] e 80% [(e) e (f)]. Nos painéis da esquerda, os osciladores estão ordenados de acordo com suas posições na rede, enquanto nos painéis da direita, de acordo com suas frequências naturais.

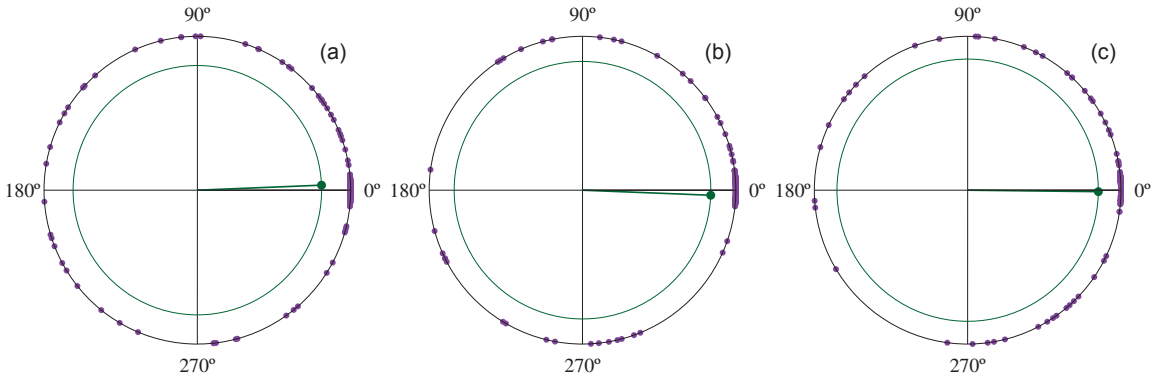


Figura 13 – Representação das fases dos osciladores em uma dada iteração, em roxo, cuja média é representada em verde. O círculo externo é unitário, enquanto o círculo interno tem raio igual ao parâmetro de ordem r. Os painéis (a), (b) e (c) estão relacionados às distribuições de 20%, 50% e 80%, respectivamente.

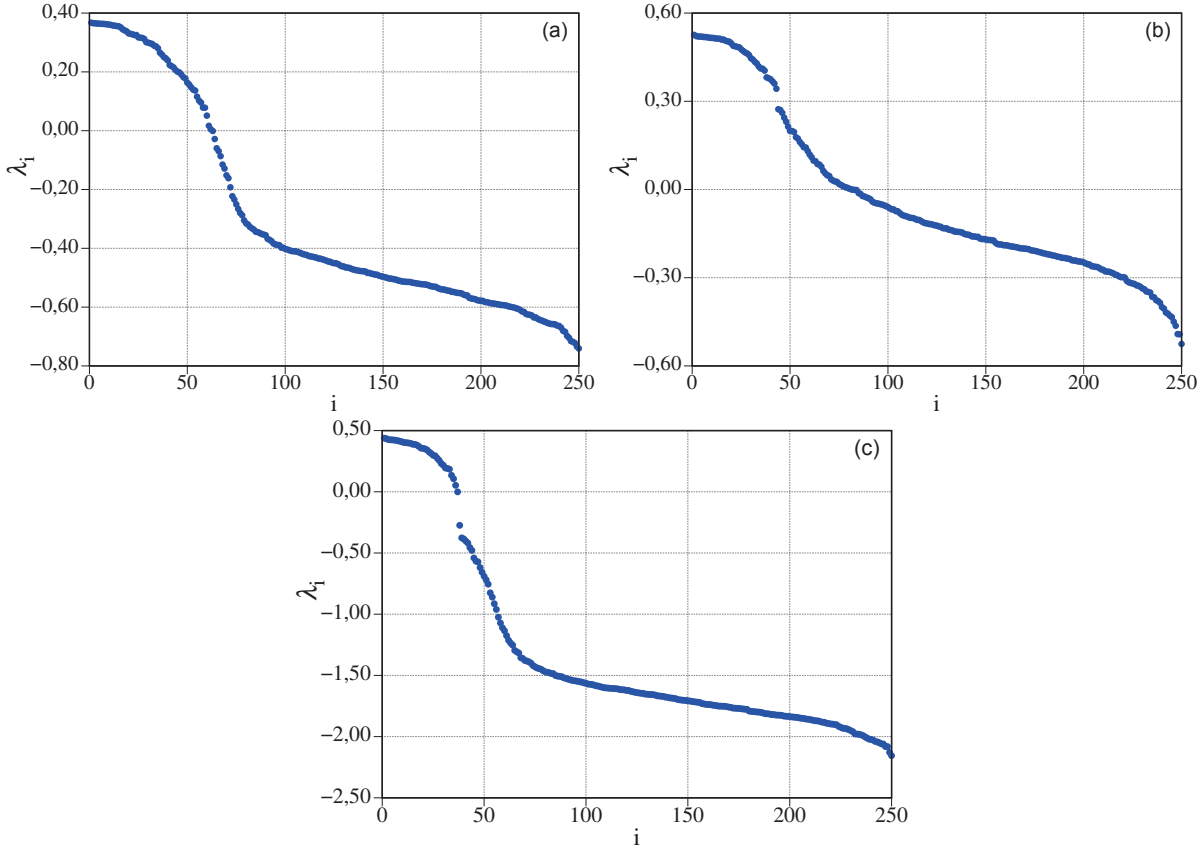


Figura 14 – Espectro de Lyapunov para o sistema de Kuramoto, em ordem decrescente, com os parâmetros selecionados para as séries de referência com distribuições de 20%(a), 50%(b) e 80%(c).

5.2 Série binária logística

Além das séries aleatórias, utilizamos um mapa logístico [119], cujos valores também estão contidos no intervalo [0,1[. O mapa logístico é uma função matemática discreta utilizada para modelar sistemas dinâmicos não-lineares. Essa função é definida como

uma equação que descreve a evolução de uma população, e apresenta a seguinte forma matemática:

$$x_{n+1} = \rho \cdot x_n \cdot (1 - x_n), \tag{5.2}$$

onde x_n é a densidade populacional na geração n, e ρ é um parâmetro que representa a taxa de crescimento e a capacidade de suporte do ambiente.

Para gerar uma série binária, escolhemos $\rho = 4.0$ (valor que leva o sistema a apresentar comportamento caótico), e seguimos a mesma regra que anteriormente, atribuindo 1 para valores maiores ou igual a um número limite (0,5, neste caso), e 0, caso contrário. Diferentemente das séries aleatórias, cujas medidas de erro avaliavam a capacidade do PRC de manter a distribuição, aqui o objetivo é emular a série de referência no trecho de validação (entre as iterações 201 e 400). Lembrando que, como explicado no capítulo anterior, comparamos os valores da série de referência, bit a bit, com o sinal de saída do PRC, ambos representados em números binários.

Na figura 15 apresentamos, em vermelho tracejado, a série de referência, e em linhas cheias verdes, o sinal de saída do nosso modelo. Observa-se que a série de referência apresenta uma certa alternância de valores, não permanecendo na mesma seção por muitas iterações consecutivas. Essa característica é mantida pelo sinal de saída nos intervalos seguintes ao de treinamento.

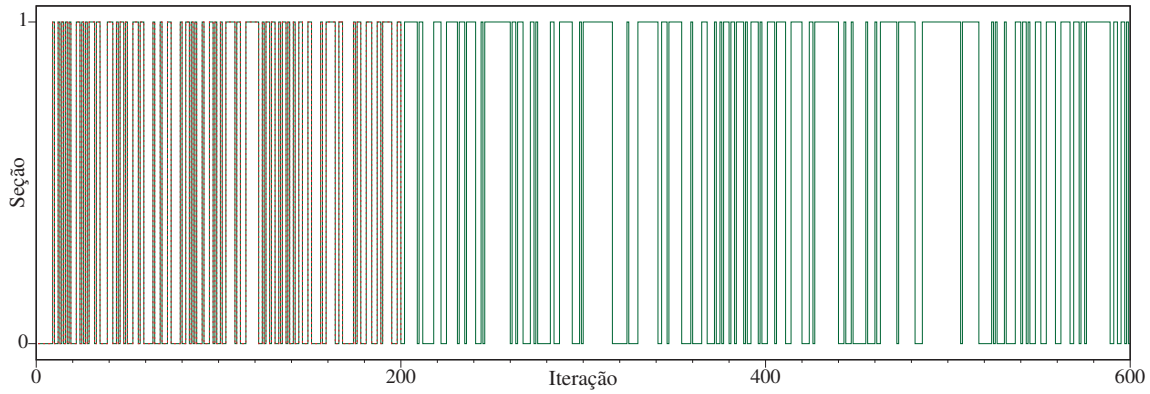


Figura 15 – Série temporal gerada por um mapa logístico. Quando o mapa logístico apresenta um valor igual ou maior que 0,5, a série de referência é 1, e 0, caso contrário. As linhas vermelhas tracejadas representam a série de referência, enquanto as linhas verdes, a saída do PRC.

A tabela 8 contém os parâmetros selecionados pelo AG, assim como o erro associado à saída do modelo com esses parâmetros. Para tornar mais evidente a medida de erro, explicarei seus dois termos. O primeiro diz respeito ao número mínimo de épocas suficiente para a aprendizagem completa, que nesse caso foram 65, das $T = 200$ primeiras amostras de treinamento, dentro do máximo de 200 épocas que estipulamos. O segundo termo

mensura o quanto o modelo foi capaz de aprender, da série de referência, no intervalo de validação (da amostra 201 até a 400). Nesse caso, o sinal de saída gerado pelo PRC acertou 73,5% dos bits do sinal de referência nesse trecho avaliado. Com essas informações, o erro resultante foi o valor de 0,265813, lembrando que 0 representa o erro mínimo, e 1, o máximo.

Tabela 8 – Parâmetros do modelo e o erro resultante.

κ	β	γ	ζ	**Erro**
0,2927	0,008	0,4253	2,6587	0,265813

Novamente, testamos o desempenho do nosso modelo com todos os parâmetros obtidos pelo GA, e sem o termo de campo médio ($\beta = 0{,}000000$). Como é possível observar na tabela 9, ao retirarmos o termo de campo médio, o valor do erro aumenta de 0,265813 para 0,369762.

Tabela 9 – Erros apresentados pelo modelo ao utilizar o parâmetro β obtido pelo GA (primeira coluna), e com $\beta = 0{,}000000$ (segunda coluna).

β **selecionado pelo GA**	$\beta = 0{,}000000$
0,265813	0,369762

Testamos também o desempenho do PRC com vários valores para a força de acoplamento κ (tabela 10), mantendo os demais parâmetros de acordo com a seleção do GA. Notamos que o GA é uma ferramenta poderosa na escolha de parâmetros que levam o sistema a apresentar um desempenho dentro da variação paramétrica proposta, visto que, para outros valores de κ, o erro tende a aumentar consideravelmente. Vale a pena destacar que, para $\kappa = 2{,}0000$ e $\kappa = 5{,}0000$, 200 épocas não foram suficientes para que houvesse aprendizado completo na fase de treinamento.

Tabela 10 – Erros apresentados pelo modelo para diferentes forças de acoplamento.

κ	**Erro**
0,0000	0,492943
0,0500	0,418343
0,1000	0,513266
0,2927	0,265813
0,5000	0,463590
1,0000	0,510706
2,0000	0,533461
5,0000	0,509184

A eficiência do GA torna evidente ao observarmos a figura 16, onde apresentamos o erro do nosso modelo associado a várias forças de acoplamento, mantendo os outros

parâmetros selecionados pelo GA. A interseção das linhas tracejadas vermelhas representa o κ selecionado pelo GA e o erro associado. Como podemos notar, o κ selecionado, assim como alguns valores próximos, apresentam os melhores desempenhos quando comparados ao restante das forças de acoplamento.

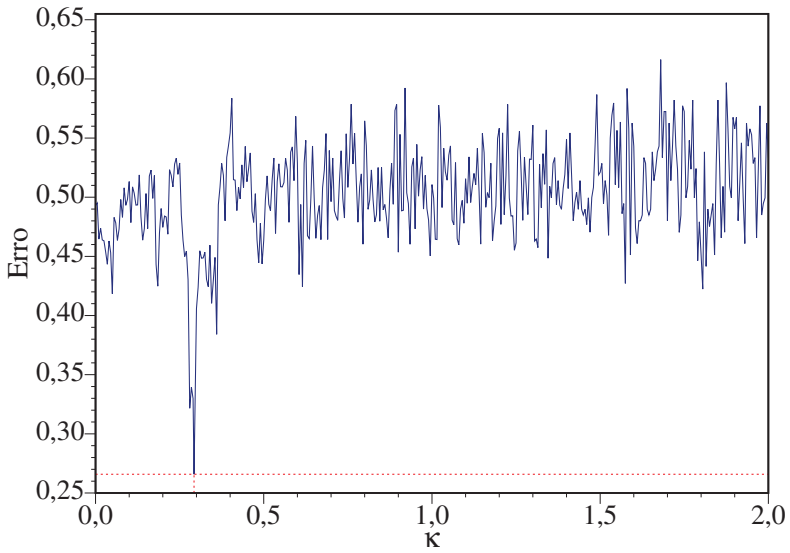


Figura 16 – Erros do PRC ao variarmos os valores da força de acoplamento κ, entre 0,000 e 2,000, com passos de 5×10^{-3}.

Quanto à sincronização, temos na figura 17 a evolução espaço-temporal das fases dos osciladores em 100 iterações, semelhantemente à figura 12. Novamente, à esquerda [17(a)] os osciladores são ordenados de acordo com suas posições na rede, enquanto, à direita [17(b)], de acordo com suas frequências naturais de oscilação ω. As cores representam as fases em gruas. Diferentemente do caso das séries binárias geradas aleatoriamente, aqui os parâmetros selecionados levam o reservatório a apresentar, em geral, uma dinâmica mais heterogênea.

Podemos confirmar essa heterogeneidade na figura 18, a qual mostra as fases dos osciladores em uma determinada iteração em pontos roxos no círculo unitário externo, e a média desses em verde, cujo raio representa o parâmetro de ordem r. Nesse caso, obtivemos $r = 0{,}07$, demonstrando que há um baixo grau de sincronização entre os osciladores da rede que compõem o reservatório.

Quanto à caoticidade, apresentamos na figura 19 o espectro dos expoentes de Lyapunov (λ). Apesar de o maior LE ser menor desta vez ($\lambda_{max} = 4{,}48 \times 10^{-3}$), em comparação ao que foi apresentado na figura 14, novamente há mais de um LE positivo, demonstrando que o reservatório continua apresentando comportamento hipercaótico. Entretanto, desta vez, muitos dos LE estão próximos a zero.

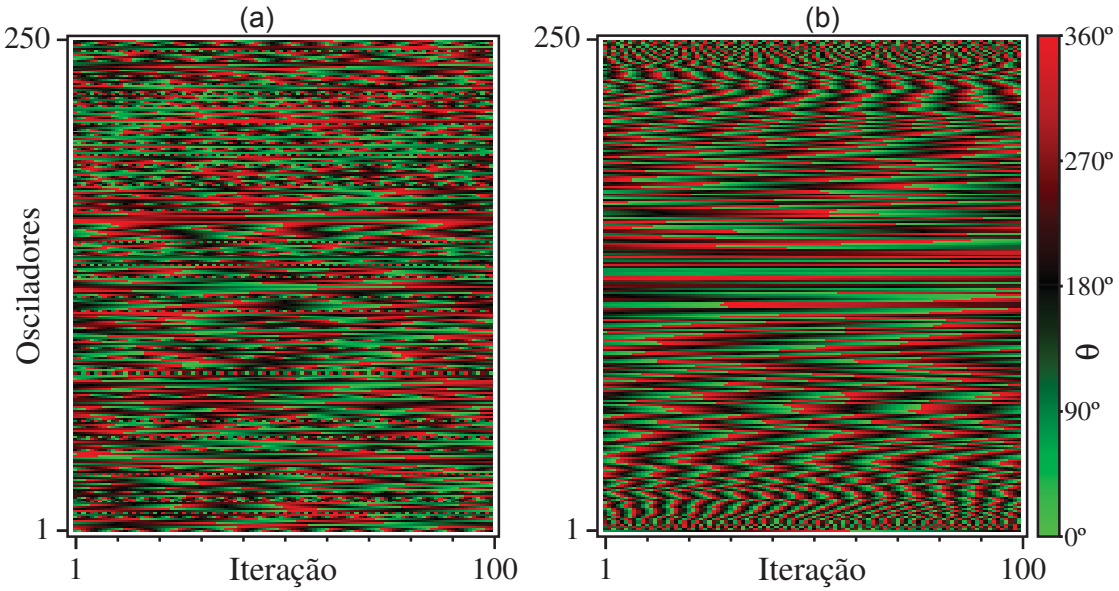


Figura 17 – Evolução espaço-temporal de θ. O painel (a) apresenta os osciladores ordenados de acordo com suas posições na rede, enquanto o painel (b), de acordo com suas frequências naturais.

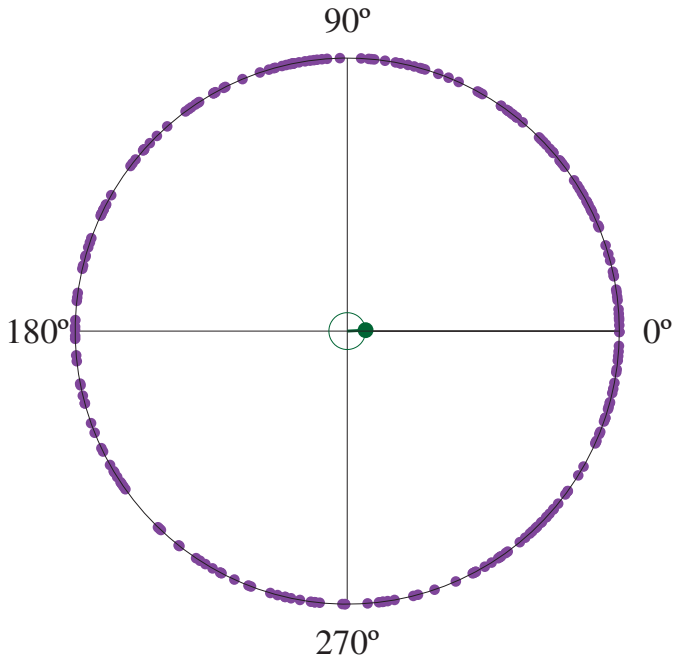


Figura 18 – Representação das fases dos osciladores em uma dada iteração, em roxo, cuja média é representada em verde. O círculo externo é unitário, enquanto o círculo interno tem raio igual ao parâmetro de ordem r.

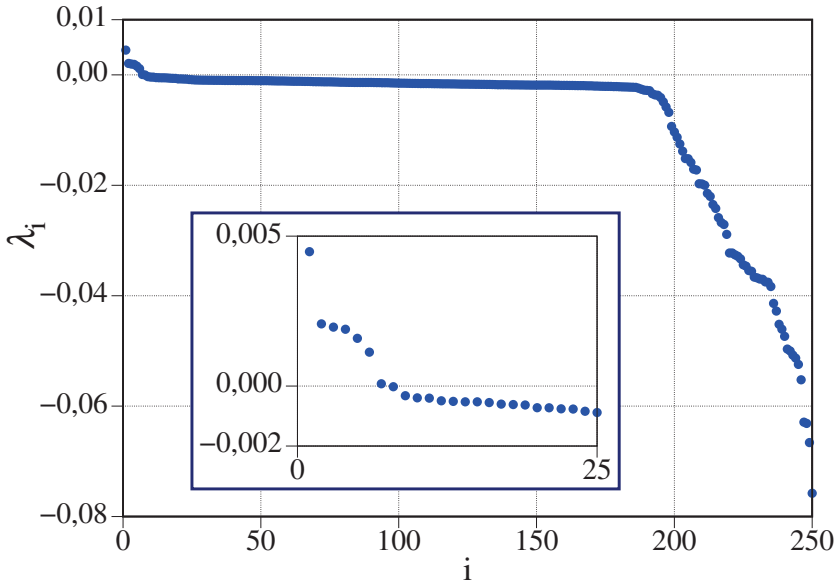


Figura 19 – Espectro de Lyapunov para o sistema de Kuramoto, em ordem decrescente, com os parâmetros selecionados pelo GA.

5.3 Série gaussiana

Utilizamos novamente a função *RANDOM_NUMBER*, dessa vez manipulada dentro da função Box-Muller [120], a qual gera valores aproximadamente de acordo com uma distribuição gaussiana. Após gerarmos a série, convertemos os seus valores de modo a pertencerem ao intervalo [0,1[e a dividimos em 16 seções. Nesse caso, a saída não possui somente dois valores. Portanto, como mencionado anteriormente, representamos a saída em números binários, e realizamos o aprendizado bit a bit como se fossem b saídas distintas para cada valor da série, onde b é o número mínimo suficiente de bits para representar a quantidade de seções em questão.

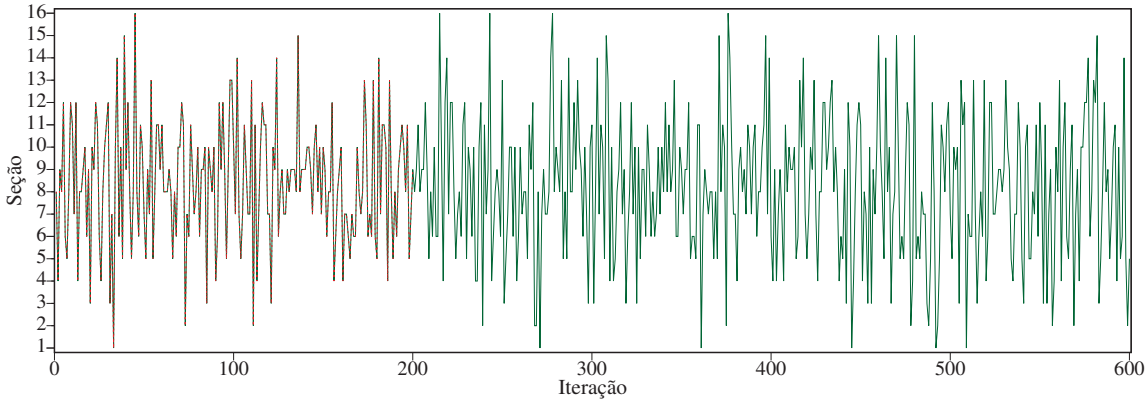


Figura 20 – Série temporal gerada aleatoriamente, de acordo com uma distribuição gaussiana, dividida em 16 seções. As linhas vermelhas tracejadas representam a série de referência, enquanto as linhas verdes, a saída do PRC.

A figura 20 apresenta a série de referência, em linhas vermelhas tracejadas, e a saída do PRC em linhas cheias verdes. A diferença em relação às figuras 9 e 15 é o número

de seções, as quais eram 2 e passam a ser 16. Observamos que a série de referência têm seus valores concentrados em torno das seções centrais (como esperávamos, por se tratar de uma série com distribuição aproximadamente gaussiana), característica aprendida e mantida pela saída do PRC durante todas as próximas iterações.

Na figura 21 evidenciamos, em um histograma, a proporção de elementos de cada seção na série de referência, representada pela coluna roxa, e na saída do PRC até as iterações 200, 400 e 600, representadas pelas colunas azul, verde e laranja, respectivamente. De fato, notamos que todas as colunas se aproximam a uma gaussiana.

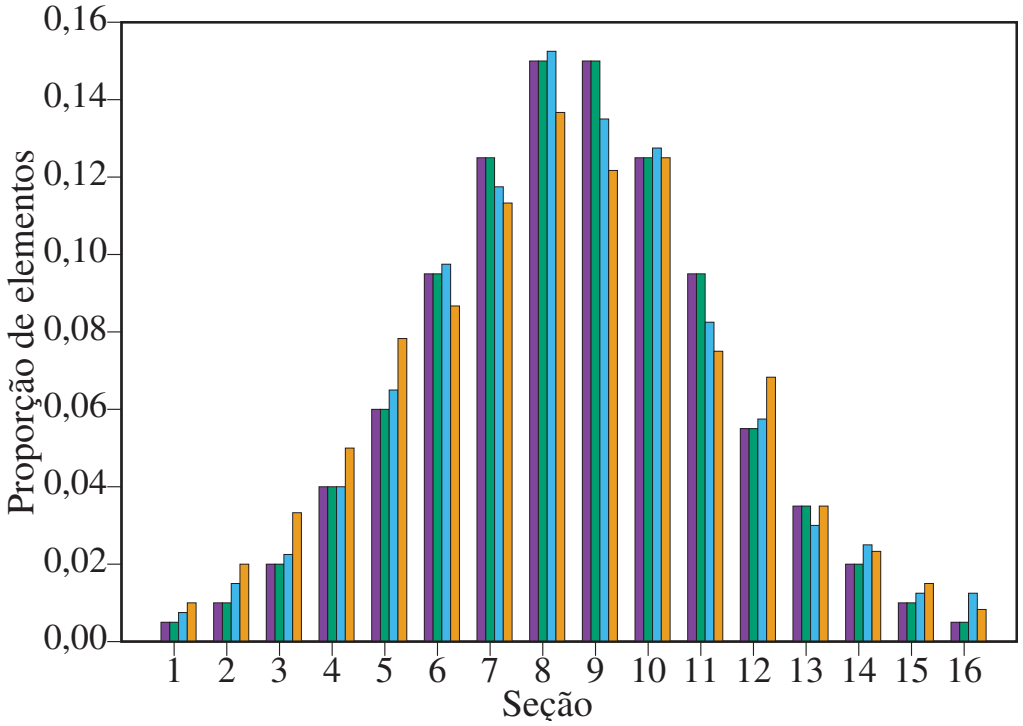


Figura 21 – Histograma com o número de elementos em cada seção da série de referência (coluna roxa) e da saída do PRC nos intervalos de treinamento (coluna verde), validação (coluna azul) e teste (coluna laranja).

Na tabela 11 apresentamos o conjunto de parâmetros selecionado pelo GA. Essa tabela contém também o valor do erro, avaliado pela capacidade do modelo em manter a distribuição da série de referência, assim como descrito na seção 5.1.

Tabela 11 – Parâmetros do modelo e o erro resultante.

κ	β	γ	ζ	**Erro**
0,1526	1,3421	4,2427	0,6387	0,179962

Novamente, testamos o desempenho do nosso modelo com todos os parâmetros obtidos pelo GA e sem o termo de campo médio ($\beta = 0{,}000000$). Como é possível observar na tabela 12, ao retirarmos o termo de campo médio o valor do erro aumenta de 0.179962 para 0.278290.

Tabela 12 – Erros apresentados pelo modelo ao utilizar o parâmetro β obtido pelo AG (primeira coluna), e com $\beta = 0{,}000000$ (segunda coluna).

β **selecionado pelo GA**	$\beta = 0{,}000000$
0,179962	0,278290

Na figura 22, apresentamos o erro para várias forças de acoplamento, e assim como para as outras séries de referência o melhor desempenho é obtido com o κ selecionado pelo GA. A interseção das linhas tracejadas vermelhas representa o κ selecionado pelo GA e o erro associado.

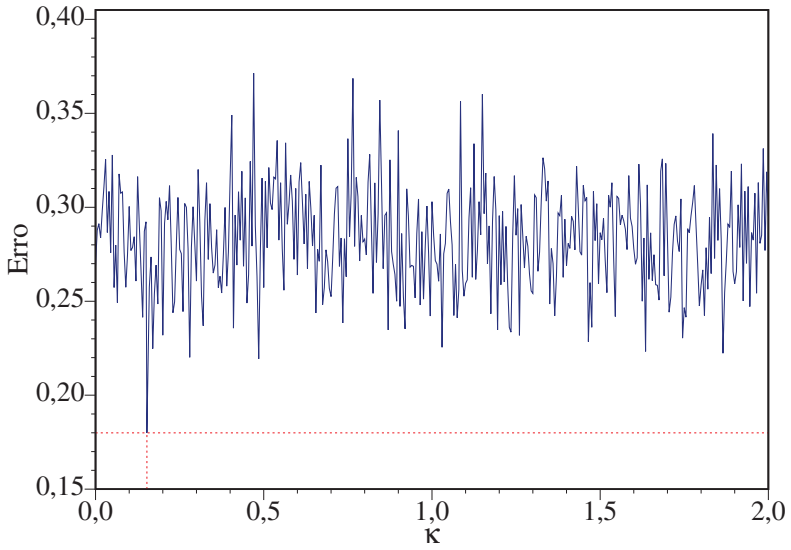


Figura 22 – Erros do PRC ao variarmos os valores da força de acoplamento κ, entre 0,000 e 2,000, com passos de 5×10^{-3}.

Na sincronização da rede, notamos semelhanças em relação ao caso descrito na seção 5.1, onde o desempenho era medido da mesma maneira. Como podemos observar no painel 23(b), onde a ordenação é feita de acordo as frequências naturais, os osciladores cujo ω é próximo a 0 (parte central da do painel) mantêm suas fases próximas a 0°, enquanto aqueles próximos às extremidades apresentam dinâmicas mais heterogêneas.

Essa semelhança foi demonstrada quantitativamente com o parâmetros de ordem $r = 0{,}80$, valor muito próximo aos apresentados na seção 5.1. Na figura 24, r é atribuído ao raio do ponto verde posicionado na média das fases dos osciladores, as quais estão representadas individualmente pelos pontos roxos no círculo unitário externo. Além disso, como trata-se de uma iteração específica, confirmamos na figura 24 o que observamos no painel 23(b), sobre as fases estarem concentradas, em sua maioria, próximas a 0°.

Mais uma vez, analisando o espectro dos LE (figura 25), nota-se que a rede que compõe o reservatório apresenta comportamento hipercaótico, visto que há mais de um LE positivo ($\lambda_{max} = 0{,}39$). Assim como na figura 14, somente um LE é zero, enquanto a maioria desses são negativos.

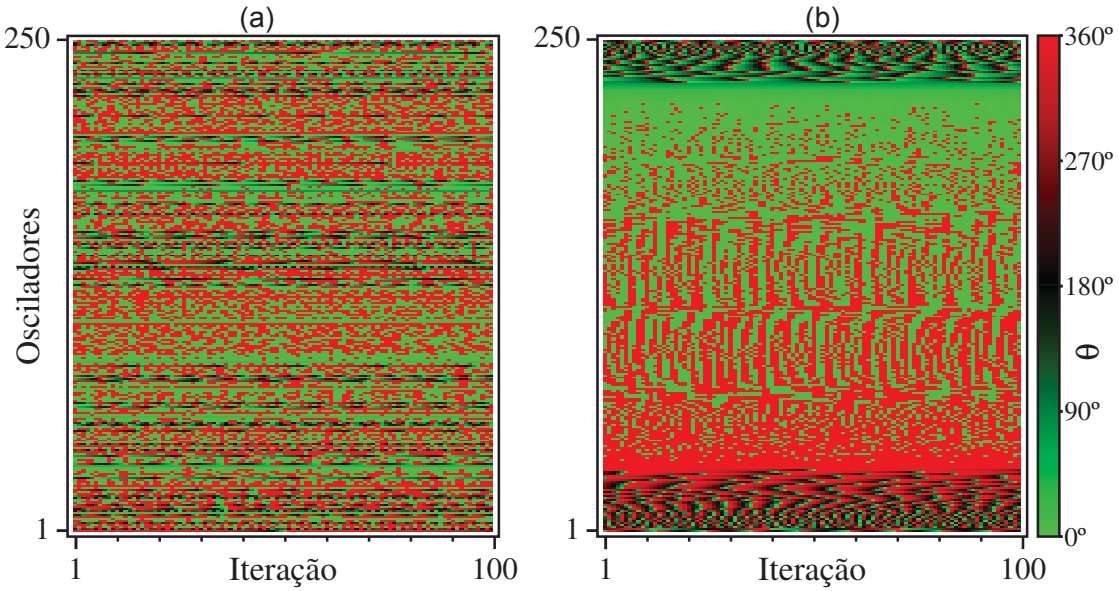


Figura 23 – Evolução espaço-temporal de θ. O painel (a) apresenta os osciladores ordenados de acordo com suas posições na rede, enquanto o painel (b), de acordo com suas frequências naturais.

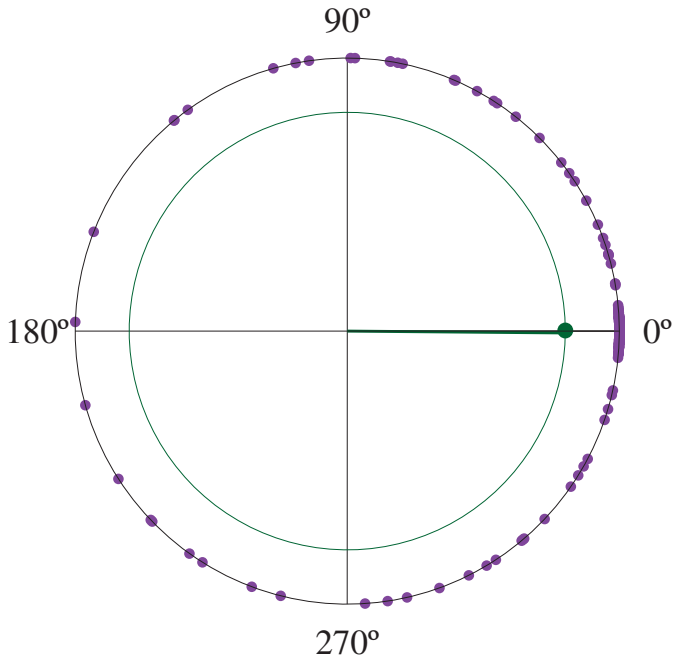


Figura 24 – Representação das fases dos osciladores em uma dada iteração, em roxo, cuja média é representada em verde. O círculo externo é unitário, enquanto o círculo interno tem raio igual ao parâmetro de ordem r.

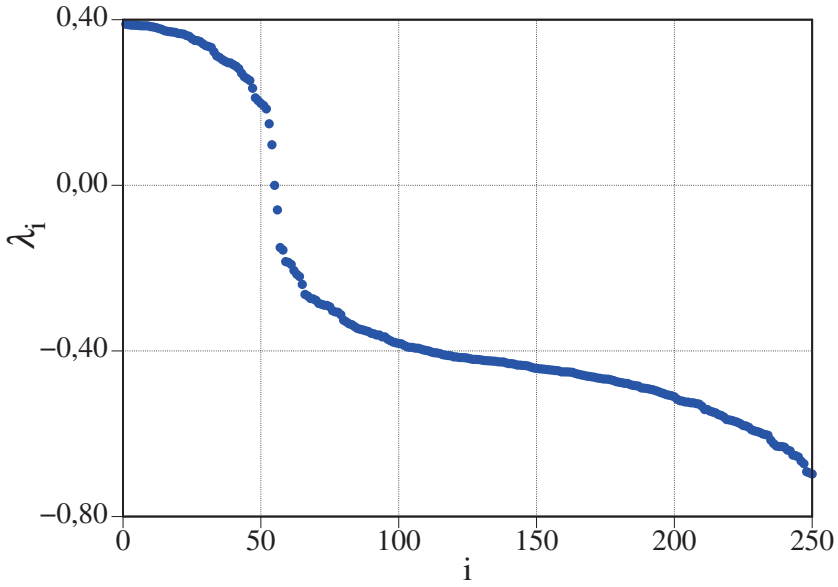


Figura 25 – Espectro de Lyapunov para o sistema de Kuramoto, dispostos em ordem decrescente.

5.4 Séries trigonométricas

Por fim, repetimos o procedimento de dividir a saída em seções, porém agora com séries geradas pelas seguintes funções trigonométricas: $\sin x$, $\sin^2 x \cdot \cos x$ e $\sin 2x \cdot \cos^2 x$. As três séries foram representadas, respectivamente, em linhas tracejadas vermelhas na figura 26, com as linhas verdes provenientes da saída do modelo. Dessa vez, foi avaliada a capacidade do PRC em emular a série de referência no trecho de validação, tal qual descrito na seção 5.2. Na figura 26 podemos notar que a saída, aparentemente, mantém o comportamento da série de referência tanto no trecho avaliado, de validação, quanto no intervalo de teste, da iteração 400 em diante.

Na tabela 13 apresentamos os conjuntos de parâmetros selecionados pelo GA, para cada uma das séries, e também a avaliação dos desempenhos. Observamos que para os três casos os valore de κ e as medidas de erro foram semelhantes.

Tabela 13 – Parâmetros do modelo e o erro resultante.

Funções	κ	β	γ	ζ	**Erro**
$\sin x$	1,2521	0,0692	4,7208	0,9104	0,191092
$\sin^2 x \cdot \cos x$	1,3343	0,0174	5,3486	4,4078	0,237656
$\sin 2x \cdot \cos^2 x$	1,5892	0,5951	0,8123	0,0186	0,186013

Novamente, temos o desempenho do nosso modelo com todos os parâmetros obtidos pelo GA e sem o termo de campo médio ($\beta = 0{,}000000$). Como é possível observar na tabela 14, ao retirarmos o termo de campo médio o valor do erro aumenta consideravelmente em todos os casos, mais uma vez indicando que esse termo auxilia no processo de aprendizagem.

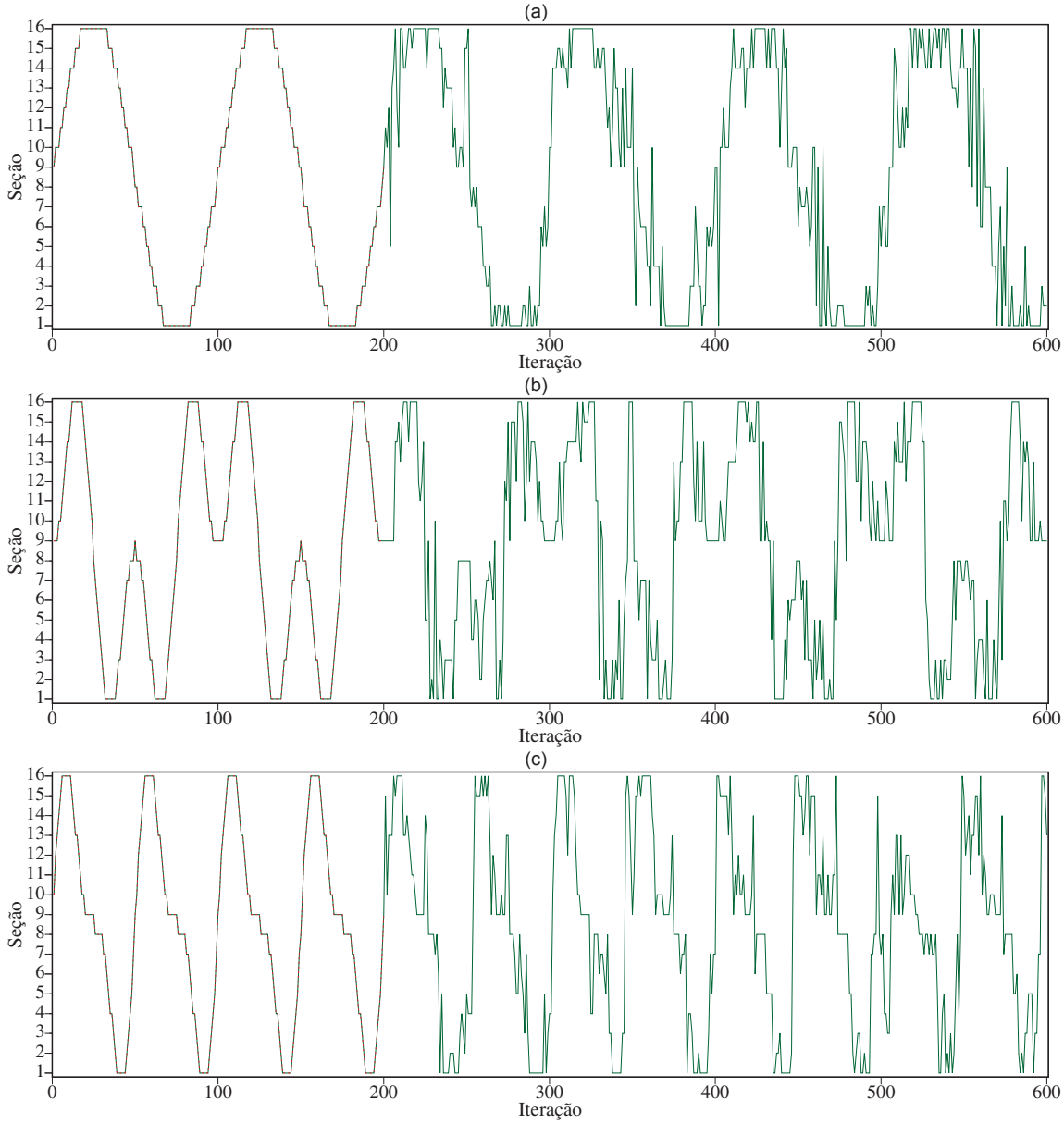


Figura 26 – Séries temporais geradas pelas seguintes funções trigonométricas: $\sin x$ (a), $\sin^2 x \cdot \cos x$ (b) and $\sin 2x \cdot \cos^2 x$ (c). As linhas vermelhas tracejadas representam a série de referência, enquanto as linhas verdes, a saída do PRC.

Tabela 14 – Erros apresentados pelo modelo ao utilizar o parâmetro β obtido pelo GA (segunda coluna), e com $\beta = 0{,}000000$ (terceira coluna).

Funções	β **selecionado pelo GA**	$\beta = 0{,}000000$
$\sin x$	0,191092	0,485894
$\sin^2 x \cdot \cos x$	0,237656	0,307974
$\sin 2x \cdot \cos^2 x$	0,186013	0,544403

Na figura 27 avaliamos o desempenho para várias forças de acoplamento. A interseção das linhas tracejadas vermelhas representa o κ selecionado pelo GA e o erro associado. Para as duas primeiras funções o melhor desempenho é obtido com o κ selecionado pelo GA, enquanto que para a terceira, esse valor está próximo ao erro mínimo,

demonstrando novamente o poder computacional do GA.

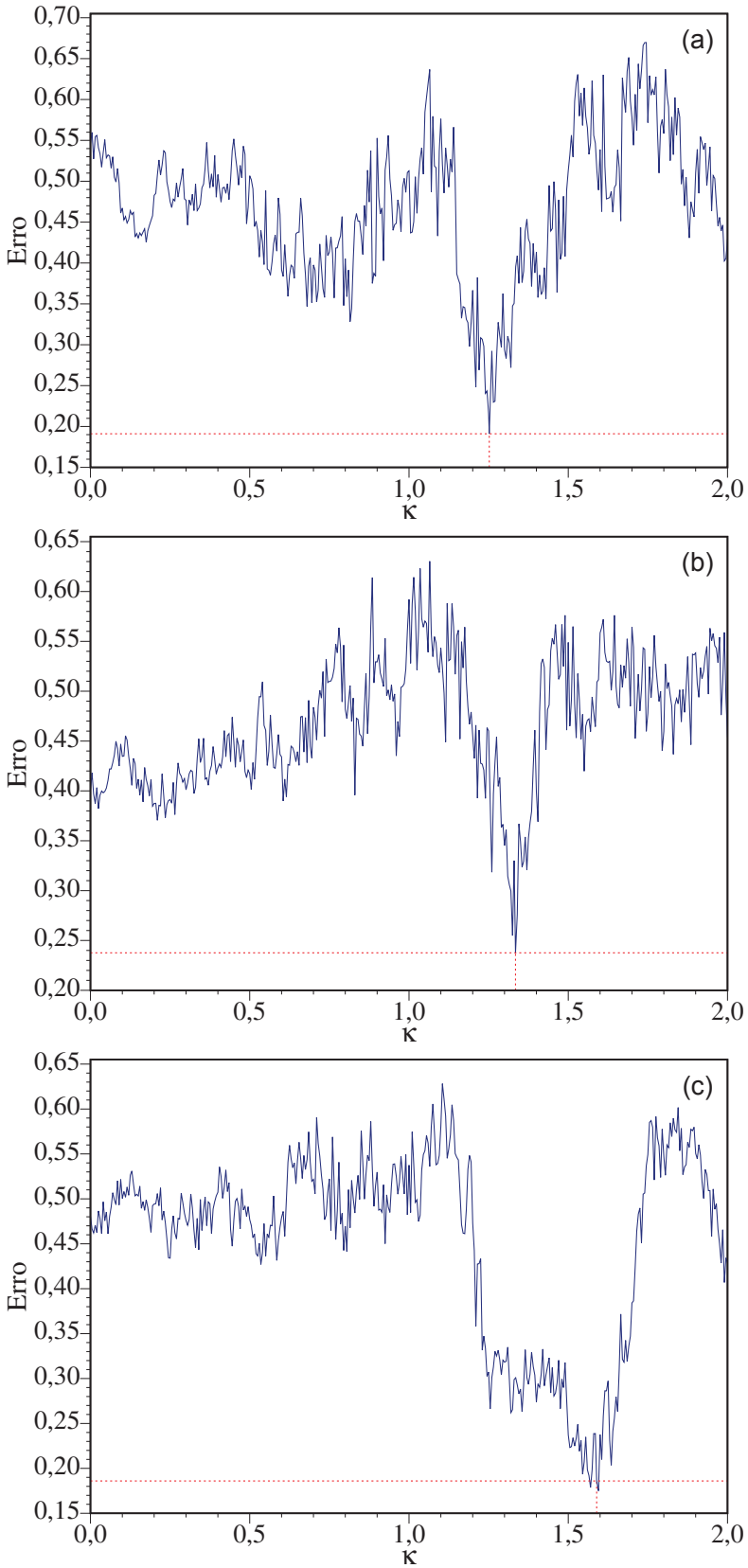


Figura 27 – Erros do PRC ao variarmos os valores da força de acoplamento κ, entre 0,000 e 2,000, com passos de 5×10^{-3}, tendo como séries de referência as funções trigonométricas: $\sin x$ (a), $\sin^2 x \cdot \cos x$ (b) e $\sin 2x \cdot \cos^2 x$ (c).

Em termos de sincronização, os três casos apresentam dinâmicas semelhantes. Assim como na figura 12 temos, na figura 28, as fases dos osciladores representadas em cores, com esses ordenados de acordo com suas posições na rede (28(a), 28(c) e 28(e)), e com suas frequências naturais ω (28(b), 28(d) e 28(f)). Observamos que há uma homogeneidade nas dinâmicas dos osciladores com seus vizinhos, o que é mais evidente na figura 28(e), porém nota-se também em 28(a) e 28(c). Notamos ainda que, apesar de os osciladores com valores absolutos de ω maiores apresentarem comportamentos mais anárquicos, a maioria desses demonstram que há certo grau de acoplamento na rede.

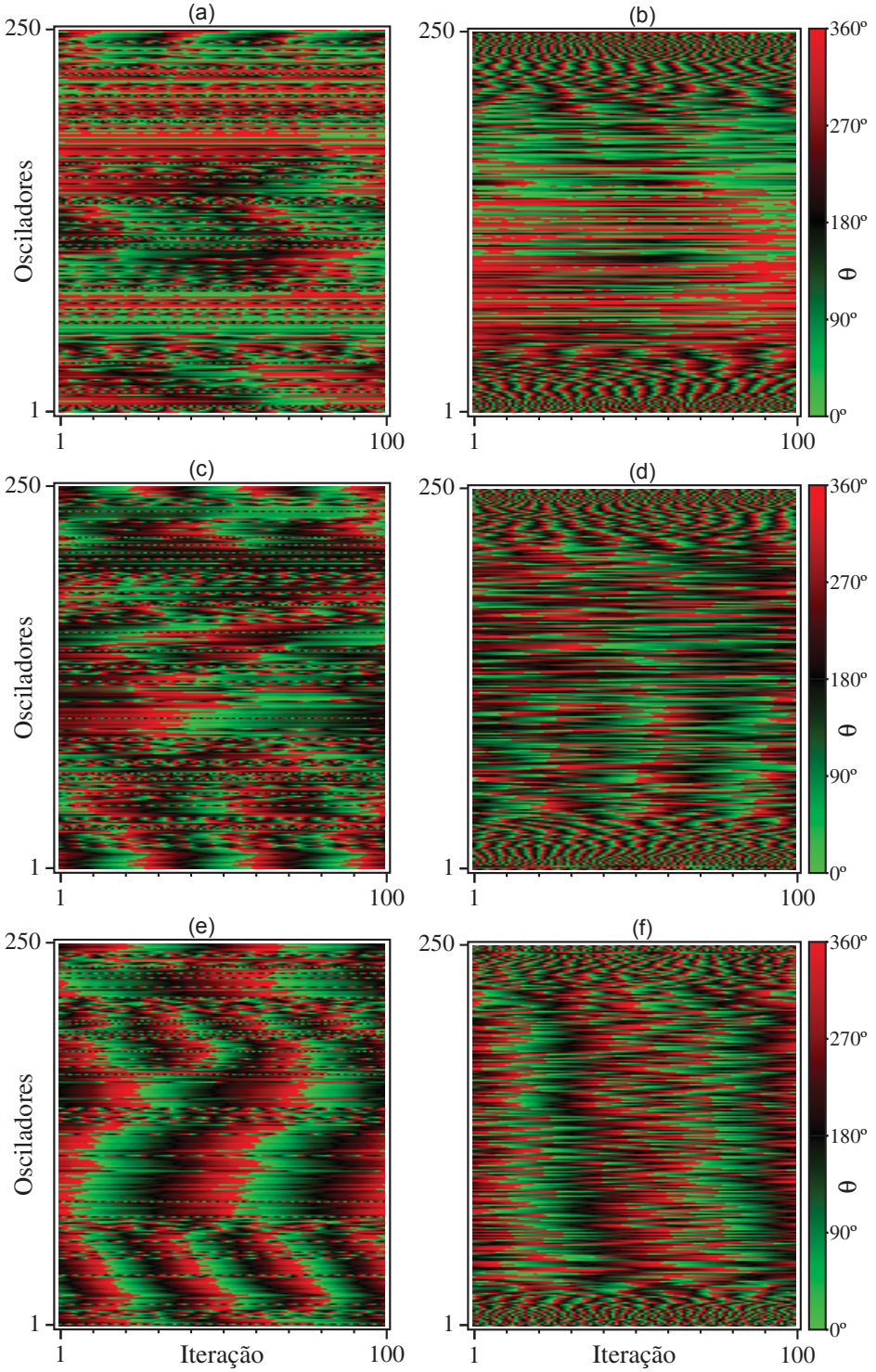


Figura 28 – Evolução espaço-temporal de θ com os parâmetros selecionados para a aprendizagem as séries trigonométricas: $\sin x$ [(a) e (b)], $\sin^2 x \cdot \cos x$ [(c) e (d)] e $\sin 2x \cdot \cos^2 x$ [(e) e (f)]. Nos painéis da esquerda, os osciladores estão ordenados de acordo com suas posições na rede, enquanto nos painéis da direita, de acordo com suas frequências naturais.

Quantificamos essa medida a partir do parâmetro de ordem. Na figura 29 apresentamos as fases em uma dada iteração, suas médias, e o parâmetro de ordem, da mesma maneira descrita nas figuras 13, 18 e 24. Para as três funções, os parâmetros de ordem

foram os seguintes: $r = 0{,}23$, $r = 0{,}16$ e $r = 0{,}20$. Essa é mais uma evidência da semelhança das dinâmicas nos três casos.

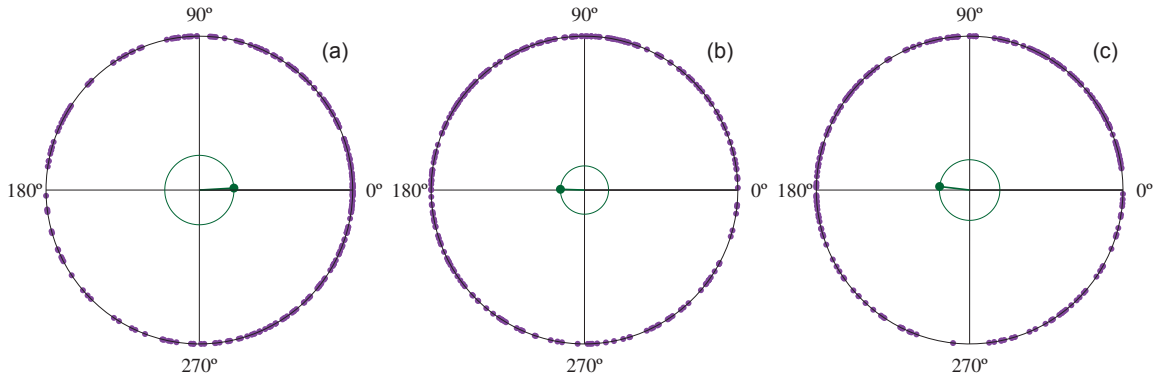


Figura 29 – Representação das fases dos osciladores em uma dada iteração, em roxo, cuja média é representada em verde. O círculo externo é unitário, enquanto o círculo interno tem raio igual ao parâmetro de ordem r. Os painéis (a), (b) e (c) estão relacionados, respectivamente, às funções trigonométricas: $\sin x$ (a), $\sin^2 x \cdot \cos x$ (b) e $\sin 2x \cdot \cos^2 x$ (c).

Ao analisarmos o espectro de LE, percebemos que há mais de um LE positivo em todos os painéis da figura 30. Assim como para as outras séries de referência, o conjunto de parâmetros que leva o PRC a apresentar o melhor desempenho nas tarefas, seja de emulação, predição, ou de manutenção da distribuição, também o leva a ter uma dinâmica hipercaótica. Neste caso, alguns dos LE estão próximos a zero, e a maioria desse são negativos.

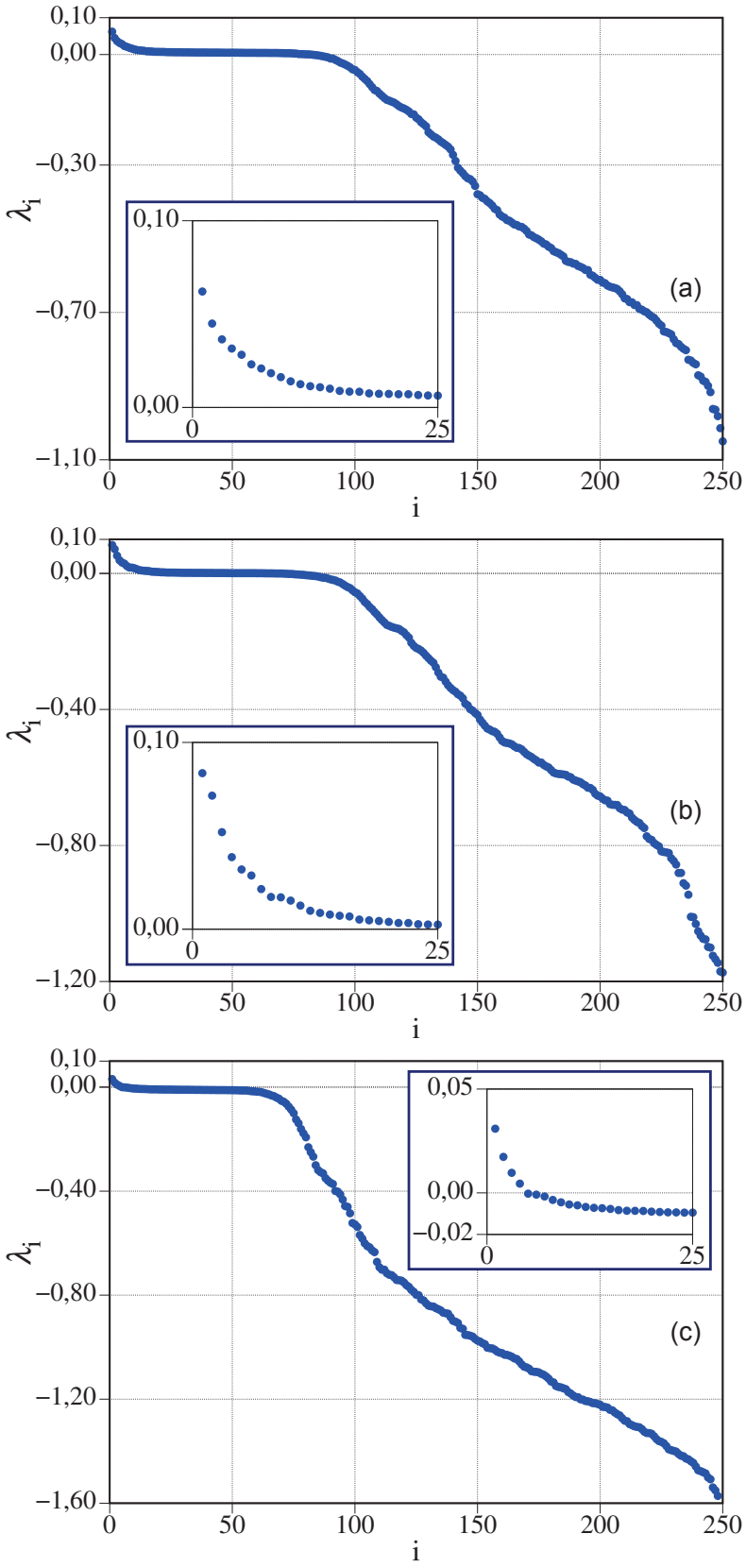


Figura 30 – Espectro de Lyapunov para o sistema de Kuramoto, em ordem decrescente, com os parâmetros selecionados para as séries trigonométricas: $\sin x$ (a), $\sin^2 x \cdot \cos x$ (b) e $\sin 2x \cdot \cos^2 x$ (c).

5.5 Série experimental gerada por saltos quânticos

Na referência [121], os autores demonstraram uma observação direta de saltos quânticos entre as camadas $6^2S_{\frac{1}{2}}$ e $5^2D_{\frac{5}{2}}$, de um íon individual de Ba^+, contido em uma armadilha de radio-frequência.

Nesse experimento, um íon individual (Ba^+) com dois estados excitados é aprisionado em uma armadilha de radio-frequência. Dois lasers são usados para conduzir o íon a um desses estados excitados ($6^2P_{\frac{1}{2}}$), no qual o íon tende a permanecer por 1 segundo em média. Cada vez que o íon retornava, emitia fótons suficientes para serem detectados pelo aparato experimental a cada, aproximadamente, 1 segundo, tornando o sinal detectado aproximadamente constante. Para levar o íon ao outro estado excitado ($6^2P_{\frac{3}{2}}$), foi utilizado uma lâmpada de cátodo oco com um filtro para a frequência desejada. Ao retornar ao estado fundamental, no caso desse segundo estado excitado, por vezes o íon ia para um estado metaestável $5^2D_{\frac{5}{2}}$, o qual tinha a peculiaridade de ter o tempo de permanência maior (em média 30 segundos) do que os outros estados envolvidos no experimento. Enquanto o íon estava nesse estado não ocorria a emissão de fótons, sendo essa uma observação direta do fenômeno. Um modelo visual dessas estruturas de níveis do Ba^+ é apresentado na figura 31 (figura 1 da referência [121]).

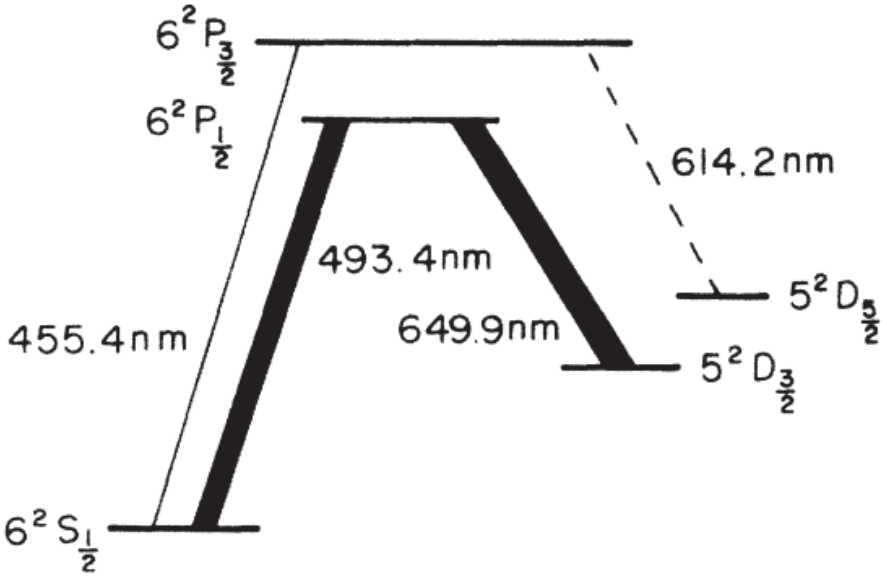


Figura 31 – Estrutura de níveis do Ba^+. O nível metaestável é o $5^2D_{\frac{5}{2}}$. As excitações pelos lasers são mostradas pelas linhas em negrito. A linha sólida fina representa a excitação causada pela lâmpada de cátodo oco, enquanto o subsequente decaimento para o nível $5^2D_{\frac{5}{2}}$ é indicado pela linha tracejada. Figura 1 da referência [121].

Na figura 32 (figura 2 da referência [121]), podemos observar que há emissão de fótons quando a linha está na parte alta da figura, e na parte baixa o íon se encontra no estado metaestável $5^2D_{\frac{5}{2}}$.

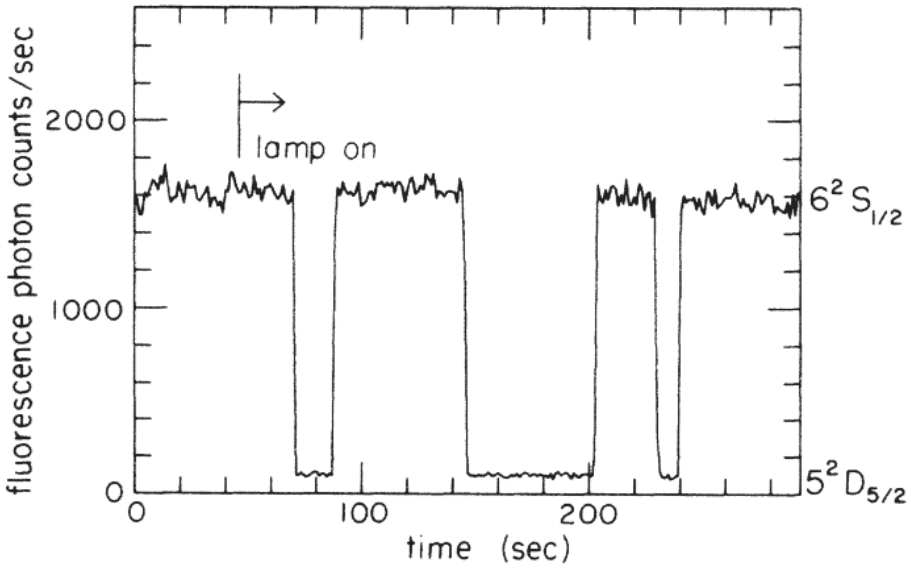


Figura 32 – Emissão de fótons pelo íon Ba^+, medidos a cada 1 segundo, aproximadamente. Após o ligamento da lâmpada de cátodo oco, os períodos de baixa fluorescência indicam que o íon está no estado metaestável $5^2D_{\frac{5}{2}}$. Figura 2 da referência [121].

Já na figura 33 (figura 3 do referência [121]), os autores contaram os tempos de permanência no estado metaestável por 203 vezes consecutivas e os apresentaram em um histograma, assim como um modelo teórico exponencial. Foi observado, portanto, que a distribuição desses tempos de permanência tende a ser uma exponencial negativa.

Baseado nesse estudo geramos um conjunto de 400 elementos, entre 0 e 80, representando os tempos de permanência consecutivos do íon no estado metaestável, já que no artigo é o tempo de interesse, uma vez que os intervalos em que o íon não está nesse estado (intervalos de emissão de fótons) não são abordados. Assim como no artigo, a distribuição desses valores gerados por nós se aproxima a uma exponencial negativa, tendo mais elementos próximos a 0 do que a 80. Dividimos esse conjunto de valores em 80 seções, isso é, se o elemento gerado estivesse no intervalo]0,10], esse pertenceria à seção 1, entre]10,20], à seção 2, e assim por diante. Essa foi a série de referência para a realização do aprendizado.

Na figura 34 apresentamos a série de referência, em linhas vermelhas tracejadas, e a saída do PRC em linhas cheias verdes. No eixo das ordenadas estão representados os tempos de permanência consecutivos no estado metaestável, e no eixo das abcissas a ordem em que esses tempos aparecem. Observamos que, como esperado, tanto na série de referência quanto na saída produzida pelo PRC há uma concentração maior de elementos nas seções menores (tempo de permanência mais próximos a 0).

A figura 35 apresenta a contagem dos tempos de permanência consecutivos no estado metaestável, dividido em 8 seções, como descrito anteriormente, tanto da série de

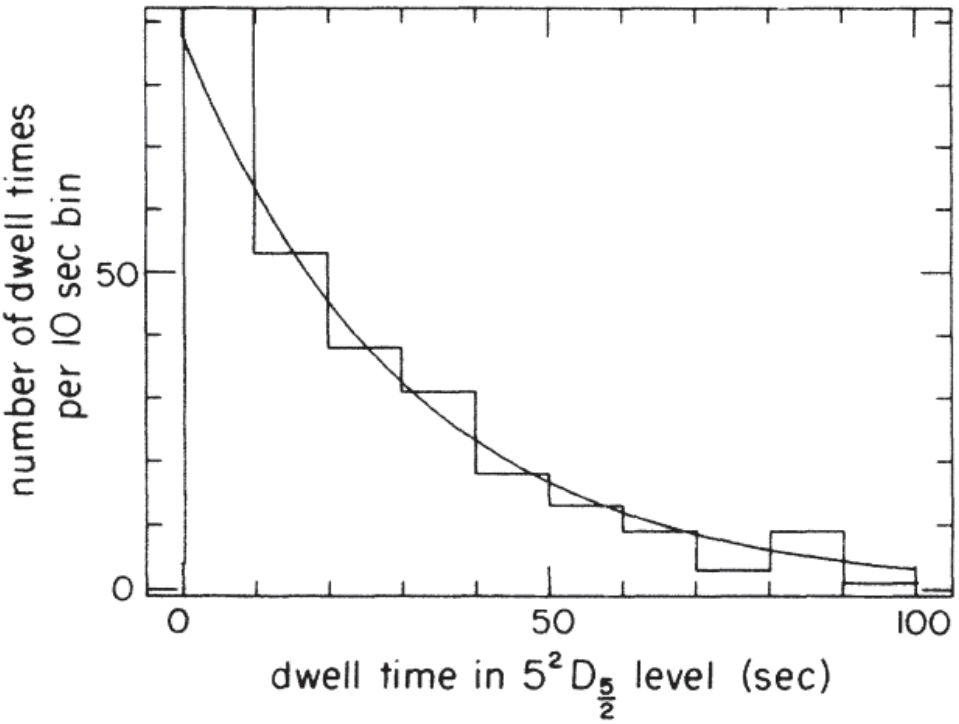


Figura 33 – Histograma apresentando a distribuição de tempos de permanência, no estado metaestável $5^2D_{\frac{5}{2}}$, mediante a contagem de 203 períodos de baixa luminescência consecutivos. Uma curva exponencial teórica foi gerada e sobreposta ao histograma experimental. Figura 3 da referência [121].

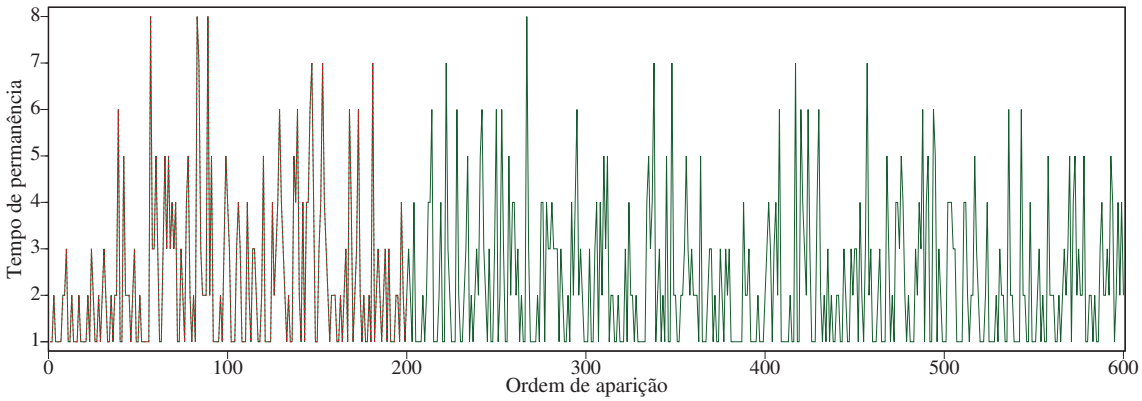


Figura 34 – Série gerada aleatoriamente, de acordo com uma distribuição exponencial, dividida em 8 seções. As linhas vermelhas tracejadas representam a série de referência, enquanto as linhas verdes, a saída do PRC.

referência no intervalo de treinamento (coluna roxa) quanto da saída do PRC nesse mesmo intervalo (coluna verde), até a iteração 400 (coluna azul) e até a iteração 600 (coluna laranja). Observamos que o PRC foi capaz de gerar uma saída cuja distribuição de cada seção se aproxima muito da série de referência.

Da mesma maneira que com as séries binárias aleatórias (5.1) e gaussiana (5.3), o desempenho, aqui, é avaliado a partir da capacidade do PRC de manter a distribuição que a série de referência apresenta no trecho de treinamento. Tanto o erro quanto os

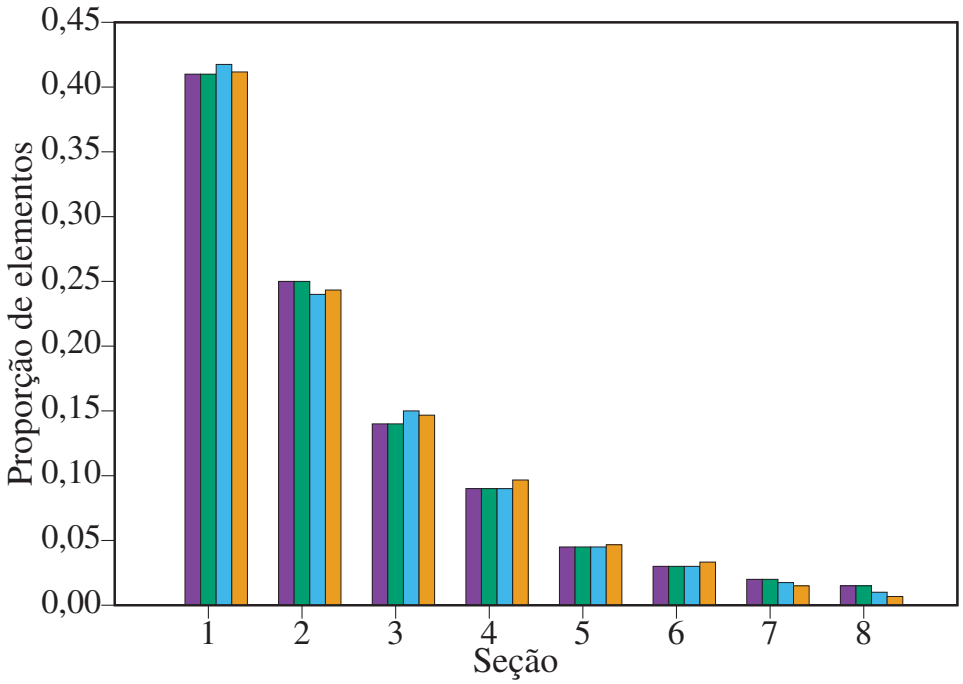


Figura 35 – Histograma com o número de elementos em cada seção da série de referência (coluna roxa) e da saída do PRC nos intervalos de treinamento (coluna verde), validação (coluna azul) e teste (coluna laranja).

parâmetros selecionados pelo GA são apresentados na tabela 15.

Tabela 15 – Parâmetros do modelo e o erro resultante.

κ	β	γ	ζ	**Erro**
0,1540	1,0115	4,1584	0,4901	0,090026

Assim como fizemos para as outras séries de referências, comparamos o desempenho do PRC com e sem a presença do termo de campo médio, cujo resultado foi apresentado na tabela 16. Observamos que, quando tirado o termo de campo médio ($\beta = 0{,}000000$), o erro aumenta de 0,090026 para 0.174798.

Tabela 16 – Erros apresentados pelo modelo ao utilizar o parâmetro β obtido pelo AG (primeira coluna), e com $\beta = 0{,}000000$ (segunda coluna).

β **selecionado pelo GA**	$\beta = 0{,}000000$
0,090026	0,174798

Na figura 36, apresentamos o erro para várias forças de acoplamento, e assim como para as outras séries de referência, o melhor desempenho é obtido com o κ selecionado pelo GA. A interseção das linhas tracejadas vermelhas representa o κ selecionado pelo GA e o erro associado.

Notamos, na sincronização da rede, semelhanças em relação aos casos descritos nas seções 5.1 e 5.3, onde o desempenho era medido da mesma maneira. Como podemos

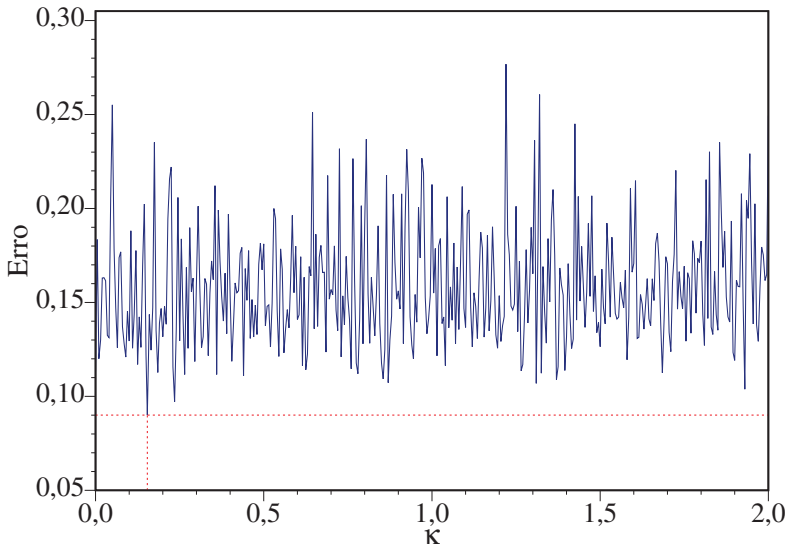


Figura 36 – Erros do PRC ao variarmos os valores da força de acoplamento κ, entre 0,000 e 2,000, com passos de 5×10^{-3}.

observar no painel 37(b), onde a ordenação é feita de acordo as frequências naturais, os osciladores cujo ω é próximo a 0 (parte central do painel) mantêm suas fases próximas a 0°, enquanto aqueles próximos às extremidades apresentam dinâmicas mais heterogêneas.

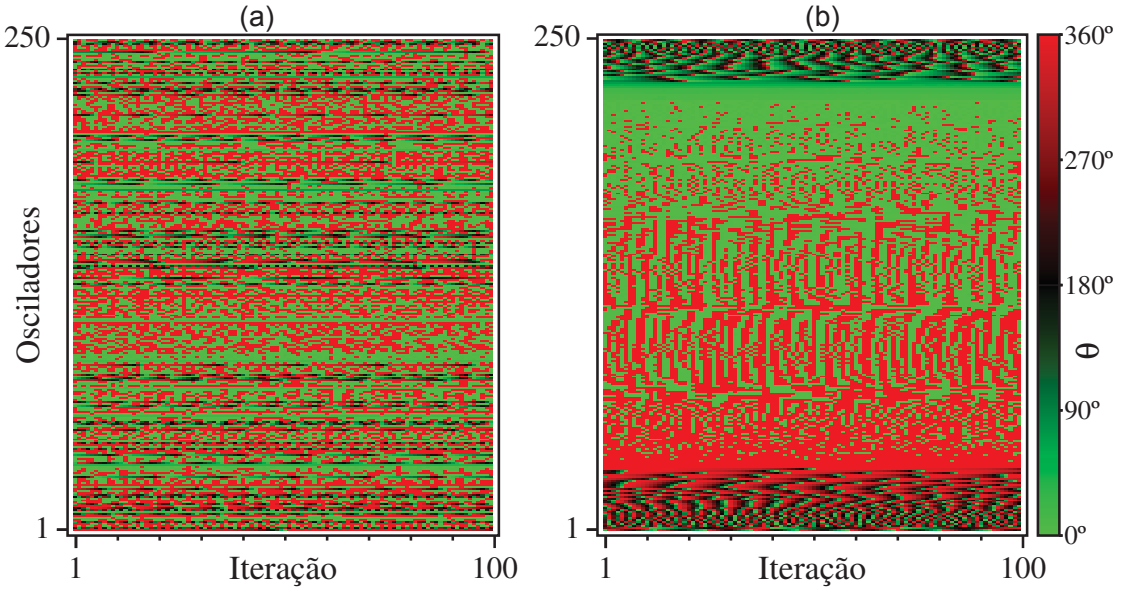


Figura 37 – Evolução espaço-temporal de θ. O painel (a) apresenta os osciladores ordenados de acordo com suas posições na rede, enquanto o painel (b), de acordo com suas frequências naturais.

Novamente, quantificamos essa semelhança com o parâmetros de ordem $r = 0{,}78$, valor muito próximo aos apresentados nas seções 5.1 e 5.3. Na figura 38, r é atribuído ao raio do ponto verde posicionado na média das fases dos osciladores, representadas individualmente pelos pontos roxos no círculo unitário externo. Além disso, como trata-se de uma iteração específica, confirmamos na figura 24 o que observamos no painel 37(b),

sobre a maioria das fases estarem concentradas próximas a 0°.

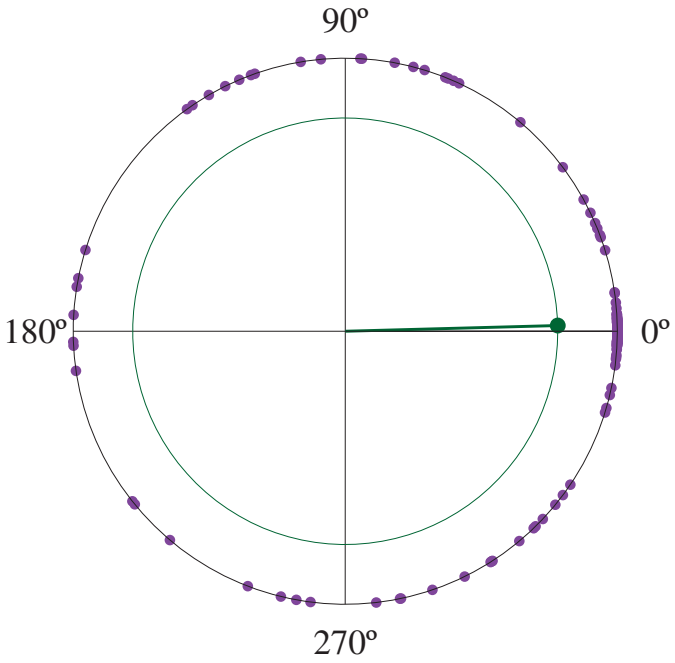


Figura 38 – Representação das fases dos osciladores em uma dada iteração, em roxo, cuja média é representada em verde. O círculo externo é unitário, enquanto o círculo interno tem raio igual ao parâmetro de ordem r.

Mais uma vez, analisando o espectro dos LE (figura 39), nota-se que a rede que compõe o reservatório apresenta comportamento hipercaótico, visto que há mais de um LE positivo ($\lambda_{max} = 0{,}32$). Assim como nas figuras 14 e 25, poucos LE são zero, vários são positivos, e a maioria desses são negativos.

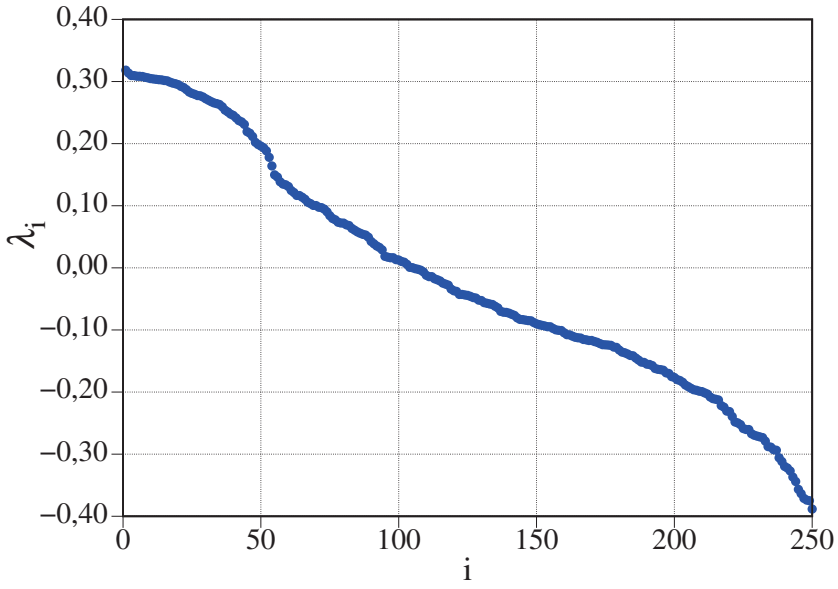


Figura 39 – Espectro de Lyapunov para o sistema de Kuramoto, dispostos em ordem decrescente.

5.6 Desempenho e dinâmica do reservatório

Nas seções anteriores deste capítulo, para diferentes séries de referência, estudamos as características da dinâmica do reservatório que levaram o PRC aos melhores desempenhos dentro dos intervalos paramétricos propostos. O objetivo, nesta seção, é variar alguns dos parâmetros selecionados pelo GA, para algumas das tarefas apresentadas anteriormente, e estudar quais os efeitos disso na dinâmica do reservatório e no desempenho do PRC.

O desempenho foi medido a partir da capacidade do PRC de emulação e predição, ou do quanto o PRC é capaz de manter a distribuição de elementos em cada seção das séries de referência. Observamos que, dentre as séries que foram utilizadas como referência para a primeira dessas tarefas citadas, os parâmetros selecionados pelo GA levaram o reservatório a apresentar dinâmicas semelhantes, ocorrendo o mesmo para a segunda tarefa. Portanto, selecionamos uma série de referência de cada tipo de tarefa para realizar a análise proposta para esta seção da tese.

Dentre as utilizadas para a tarefa de emulação e predição, escolhemos a primeira das séries trigonométricas, a qual é uma função seno dividida em 16 seções. Os parâmetros selecionados pelo GA foram apresentados na tabela 13. A figura 40 é um espaço de parâmetros cujos eixos contêm a força de acoplamento κ, e β, o coeficiente do termo de campo médio, onde as cores representam o erro apresentado pelo PRC, cuja escala é apresentada na caixa de cores à direita da figura. Observamos que o par de parâmetros escolhido pelo GA está na parte superior esquerda da figura, onde $\kappa = 1{,}2521$ e $\beta = 0{,}0692$. A região em torno desse ponto contém pares de parâmetros que levam o PRC a apresentar os melhores desempenhos dentro das variações propostas. Além disso, notamos que em torno de $\beta = 0{,}0692$, vários valores de κ são melhores escolhas do que o restante da figura à direita, onde o coeficiente do campo médio começa a aumentar.

É interessante destacar que, dentre todos os pares de parâmetros testados dentro do intervalo proposto, o selecionado pelo GA é o que leva o PRC ao melhor desempenho. Evidenciamos isso na figura 41, onde variamos o parâmetro κ entre 0,000 e 2,000, com passos de 5×10^{-3}, mantendo o β selecionado pelo GA (painel 41(a)), e fizemos o mesmo procedimento, porém invertendo os parâmetros κ e β (painel 41(b)). Os dois paineis são correspondentes às linhas tracejadas vertical e horizontal, respectivamente, da figura 40.

Conforme citado anteriormente, o objetivo aqui é estudar a relação entre o desempenho do PRC e a dinâmica do reservatório. Para isso, escolhemos dois outros pontos do espaço de parâmetros (figura 40), avaliamos o desempenho do PRC com esses pares de parâmetros, e investigamos a dinâmica do reservatório. No ponto $P1$ mantivemos o β selecionado pelo GA e tornamos κ nulo, ou seja, tiramos o termo de acoplamento característico dos osciladores de Kuramoto. No ponto $P2$, com κ nulo, tomamos $\beta = 1{,}5000$. A tabela 17 apresenta κ, β e o erro resultante para cada um dos pontos evidenciados

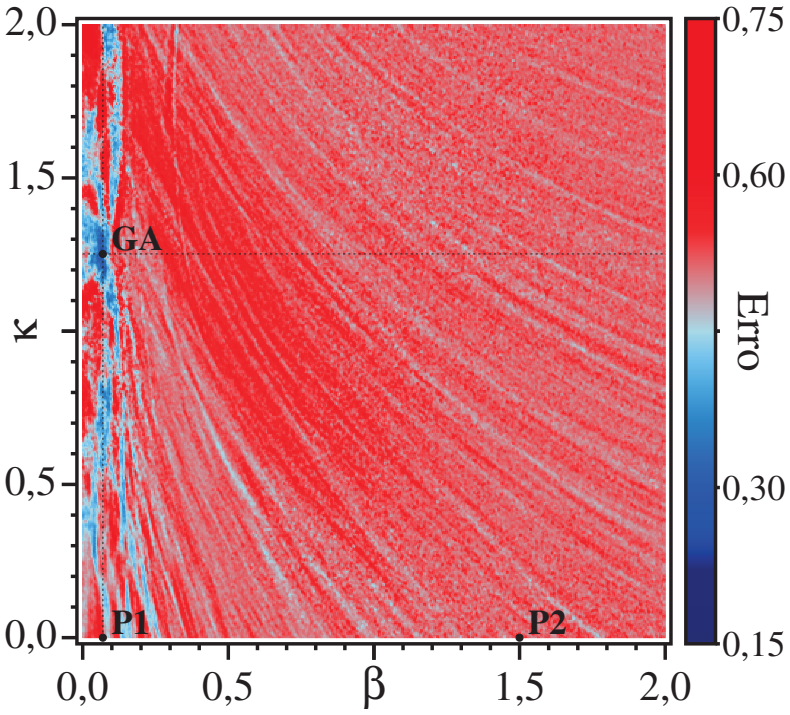


Figura 40 – Espaço de parâmetros com os erros do PRC, representados pelas cores, ao variarmos os valores da força de acoplamento (κ), e do coeficiente do termo de campo médio (β), entre 0,000 e 2,000, com passos de 5×10^{-3}.

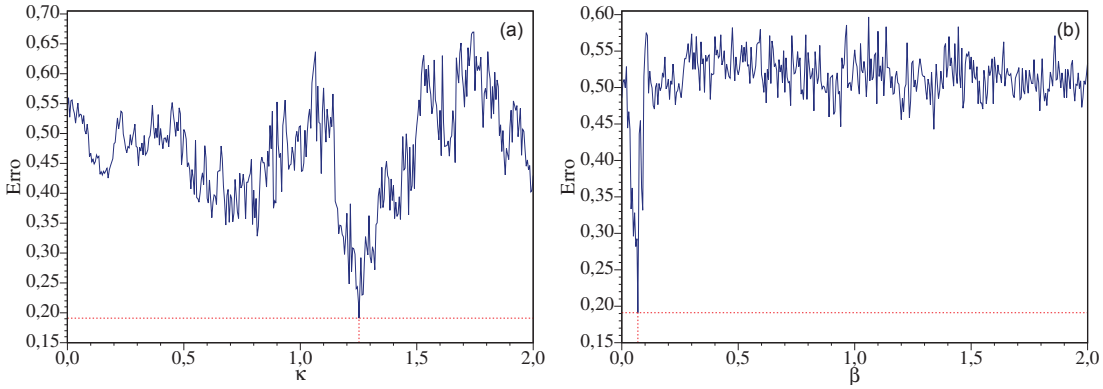


Figura 41 – Erros do PRC ao variarmos os valores da força de acoplamento (κ)(a), e do coeficiente do termo de campo médio (β)(b), ambos entre 0,000 e 2,000, com passos de 5×10^{-3}.

na figura 40, sendo que o restante dos parâmetros foram mantidos os selecionados pelo GA. Notamos que os pares de parâmetros destacados em $P1$ e $P2$ levam o PRC a um desempenho muito pior do que o par selecionado pelo GA.

Tabela 17 – Parâmetros do modelo e o erro resultante.

Ponto	κ	β	**Erro**
GA	1,2521	0,0692	0,191092
$P1$	0,0000	0,0692	0,546064
$P2$	0,0000	1,5000	0,541971

Para tornar mais evidente a perda de desempenho devido à variação dos parâmetros, apresentamos, na figura 42, a série de referência $\sin x$ em linhas tracejadas vermelhas, assim como a saída do PRC em linhas verdes, para as três configurações paramétricas descritas na tabela 17, respectivamente. No painel 42(a) o PRC aprendeu a série no intervalo de treinamento, dentro das 200 épocas pré-estabelecidas, e manteve o comportamento da curva de referência após a aprendizagem. O painel 42(b) apresenta o primeiro intervalo idêntico ao anterior, porém a partir do intervalo de treinamento houve uma deformação mais evidente em relação à série de série de referência. No painel 42(c) observamos que, além de uma perda ainda mais acentuada das características da série de referência, não houve aprendizado completo do intervalo de treinamento dentro das 200 épocas estipuladas.

Iniciando a investigação da dinâmica do reservatório para cada uma das três variações de parâmetros, descritas na tabela 17, apresentamos na figura 43, em roxo, as fases dos osciladores de Kuramoto em uma determinada iteração, com a média dessas representada pelo ponto verde, o qual está sobre o raio cuja magnitude é dada pelo valor do parâmetro de ordem r. De acordo com o que foi apresentado na seção 5.4, com os parâmetros selecionados pelo GA, o parâmetro de ordem calculado foi $r = 0{,}23$. Notamos que ao tornar nula a força de acoplamento κ reduzimos o grau de sincronização do sistema, levando o parâmetro de ordem a $r = 0{,}09$ e prejudicando o desempenho do PRC, como vimos na tabela 17 e no painel 42(b). Ao mantermos $\kappa = 0{,}0000$ e tomarmos $\beta = 1{,}5000$, aumentamos a homogeneidade da rede de osciladores, o que pode ser quantificado com o parâmetro de ordem $r = 0{,}68$. Isso levou o PRC a apresentar um desempenho semelhante ao do par de parâmetros do ponto $P1$ (figura 40), se considerarmos o valor medido do erro (tabela 17), porém, como observamos no painel 42(c), com essa configuração o PRC não foi nem capaz de emular a série de referência no intervalo de treinamento, dentro das 200 épocas estipuladas, e deformou ainda mais curva senoide nos intervalos seguintes.

Além da sincronicidade, podemos investigar a dinâmica do reservatório a partir do espectro dos LE para cada uma das variações de parâmetros, apresentados na figura 44. O painel 44(a) é referente à dinâmica do sistema com os parâmetros selecionados pelo GA, $\kappa = 1{,}2521$ e $\beta = 0{,}0692$, na qual há poucos LE positivos, alguns nulos, e a maior parte, negativos. A partir do painel 44(b) observamos que, na configuração do ponto $P1$ (figura 40), o sistema não apresenta nenhum LE positivo, sendo na maioria próximos a zero, e alguns negativos. Quando tomamos $\beta = 1{,}5000$, mantendo κ nulo, notamos (painel 44(c)) que o espectro se dispõe de uma forma característica de quando β assume valores maiores, demonstrado também nas figuras 14 e 25. No entanto, relacionado a essas figuras recém citadas, o método de avaliação de desempenho era baseado na capacidade do PRC em manter a distribuição das séries de referência, diferentemente do método adotado para as séries trigonométricas. Isso explica o porquê configurações de parâmetros semelhantes, que levam o sistema a espectros de LE semelhantes, podem tanto levar o PRC a um bom desempenho (seções 5.1 e 5.3), quanto a prejudicar a aprendizagem, que é o observado

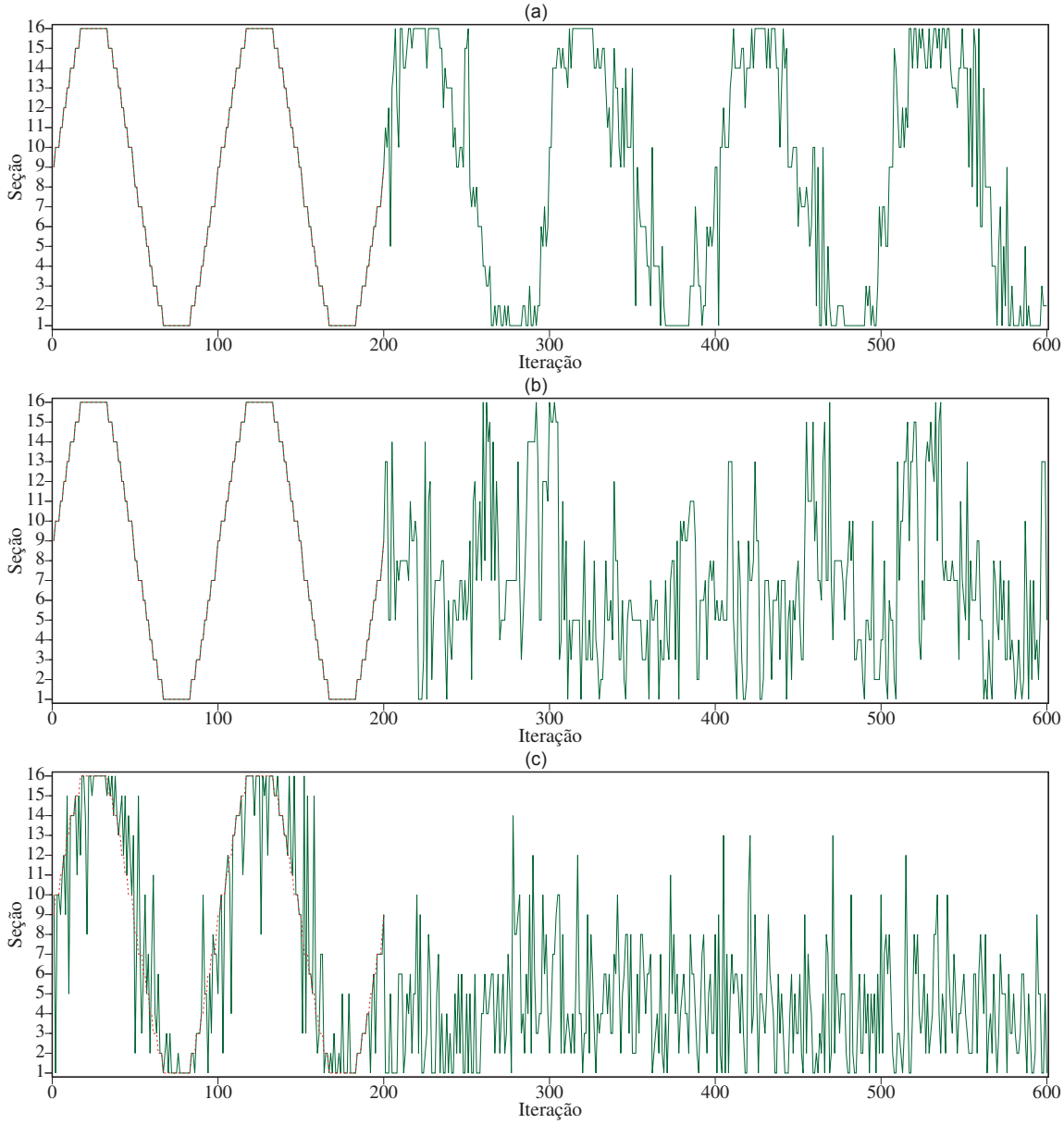


Figura 42 – Séries temporais geradas pela função trigonométrica $\sin x$. As linhas vermelhas tracejadas representam a série de referência, enquanto as linhas verdes, a saída do PRC. Em (a), os parâmetros do sistema que descreve a dinâmica da rede de osciladores de Kuramoto foram selecionados pelo GA, em (b), κ foi zerado, e em (c), κ permaneceu nulo, enquanto tomamos $\beta = 1{,}5000$.

neste caso da série de referência $\sin x$, como foi demonstrado na tabela 17 e na figura 42.

Dentre as séries de referência utilizadas para as tarefas de manutenção da distribuição dos elementos, escolhemos a série gaussiana dividida em 16 seções. Os parâmetros selecionados pelo GA foram apresentados na tabela 11. A investigação da relação entre a dinâmica do reservatório e o desempenho do PRC foi conduzida de maneira semelhante ao caso da série trigonométrica, portanto, iniciamos com o espaço de parâmetros da figura 45. Nessa, assim como na figura 40, os eixos contém κ e β e as cores representam o erro apresentado pelo PRC. Novamente, destacamos o par de parâmetros selecionado pelo GA

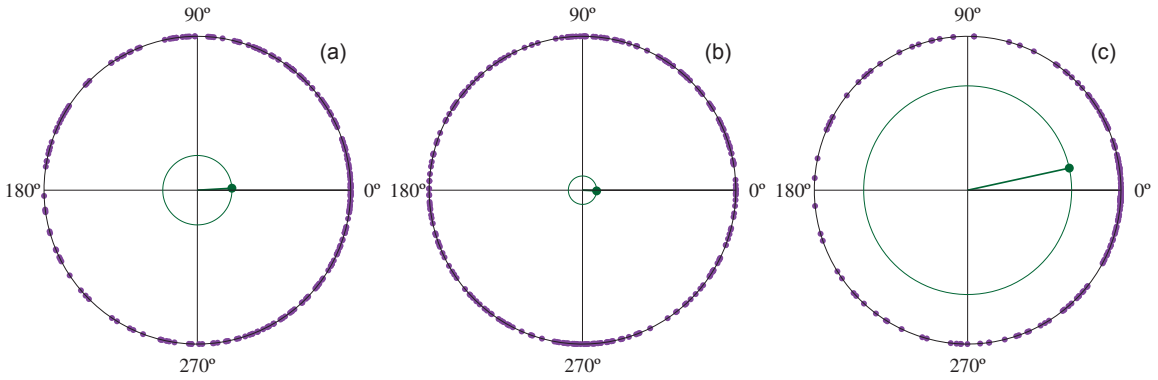


Figura 43 – Representação das fases dos osciladores em uma dada iteração, em roxo, cuja média é representada em verde. O círculo externo é unitário, enquanto o círculo interno tem raio igual ao parâmetro de ordem r. Em (a), os parâmetros do sistema que descreve a dinâmica da rede de osciladores de Kuramoto foram selecionados pelo GA, em (b), κ foi zerado, e em (c), κ permaneceu nulo, enquanto tomamos $\beta = 1{,}5000$.

($\kappa = 0{,}1526$ e $\beta = 1{,}3421$) e escolhemos outros dois pontos, $P1$ e $P2$, para investigarmos a dinâmica do reservatório e avaliarmos o desempenho do PRC.

Notamos que diferentemente da figura 40, onde havia uma região em torno do ponto GA que apresentava os melhores desempenhos, na figura 45 não há uma região tão bem definida, mesmo que, novamente, o par de parâmetros do ponto GA[1] tenha sido aquele que levou o PRC ao melhor desempenho dentre todos os pontos do espaço de parâmetros. Além disso, notamos que o valor do erro, na maior parte dessa figura, é menor do que apresentado na figura 40. Isso se deve às diferenças no modo de avaliação do desempenho. No caso anterior, eram comparadas a saída do PRC e a série de referência, bit a bit, e se contava os bits distintos. Neste caso são realizadas contagens de quantos elementos há em cada seção da série de referência no trecho de treinamento, e da saída do PRC entre as iterações 200 e 400, e ambas as contagens são comparadas, sem haver a necessidade de haver congruência de valores bit a bit (há também a contagem das sequências de dois valores consecutivos e a avaliação do número de épocas para a emulação do intervalo de treinamento da série de referência, como foi explicado no capítulo 4, mas o objetivo dessa comparação é explicar o motivo de os erros apresentarem valores menores, em geral, na figura 45 em comparação à figura 40). Notamos, também, que os melhores desempenhos são apresentados pelos pares de parâmetros nos quais há uma predominância do valor de β em relação a κ. Isso é, conforme se aumenta κ, o erro também tende a aumentar, como é possível observar no canto superior esquerdo do espaço de parâmetros. Com o aumento de β, no entanto, o erro tende a diminuir novamente.

Na figura 46 apresentamos, individualmente, as variações de κ (painel 46(a)) e β

[1] Onde GA está escrito em itálico, refirmo-me ao ponto GA, e não somente à sigla referente a Algoritmo Genético

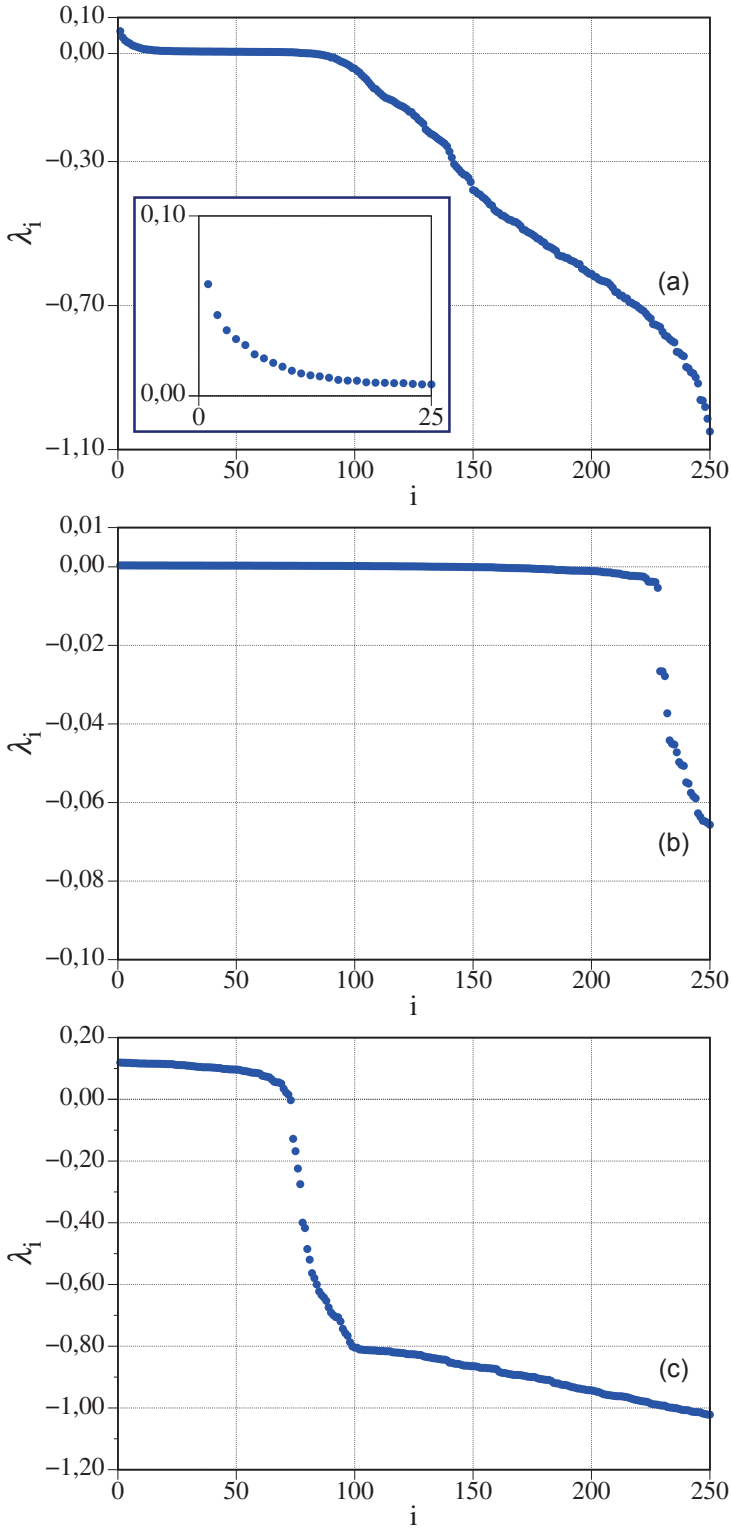


Figura 44 – Espectro de Lyapunov para o sistema de Kuramoto, em ordem decrescente. Em (a), os parâmetros do sistema que descreve a dinâmica da rede de osciladores de Kuramoto foram selecionados pelo GA, em (b), κ foi zerado, e em (c), κ permaneceu nulo, enquanto tomamos $\beta = 1{,}5000$.

(painel 46(b)), entre 0,0000 e 2,0000, com passos de 5×10^{-3}, assim como os erros associados. Nessas, notamos que, de fato, o menor erro é proveniente do par de parâmetros selecionado

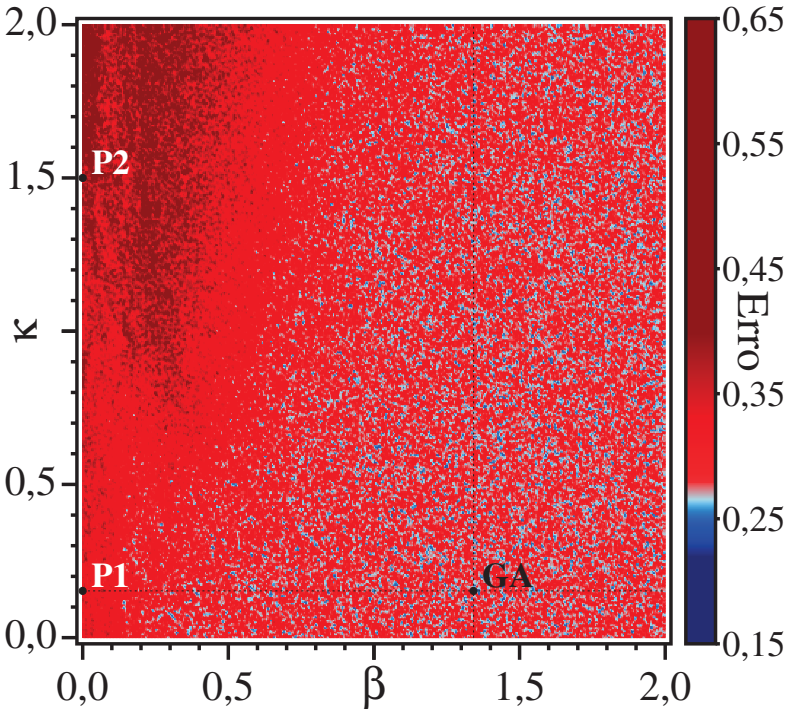


Figura 45 – Espaço de parâmetros com os erros do PRC, representados pelas cores, ao variarmos os valores da força de acoplamento (κ), e do coeficiente do termo de campo médio (β), entre 0,000 e 2,000, com passos de 5×10^{-3}.

pelo GA. Outra característica, observada também na figura 45, mas que se torna mais clara aqui, é a menor variação entre o erro apresentado pelo par de parâmetros do GA e o restante. Comparando com a figura 41, onde os parâmetros próximos aos selecionados apresentavam também desempenhos melhores que os demais, aqui isso não ocorre de maneira tão evidente. Os painéis da figura 46 são, respectivamente, correspondentes às linhas tracejadas vertical e horizontal da figura 45.

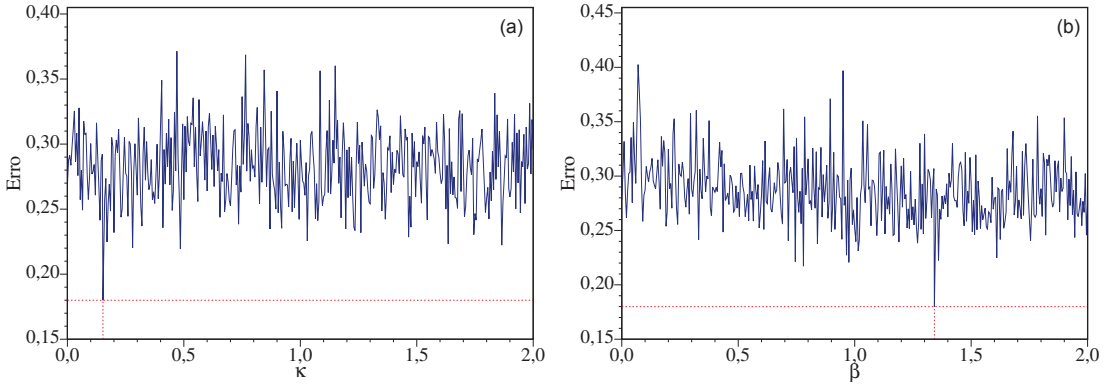


Figura 46 – Erros do PRC ao variarmos os valores da força de acoplamento (κ)(a), e do coeficiente do termo de campo médio (β)(b), ambos entre 0,000 e 2,000, com passos de 5×10^{-3}.

Conforme pode ser verificado na figura 45, o par de parâmetros selecionado pelo GA está na parte inferior direita da figura, onde $\beta = 1{,}3421$ e $\kappa = 0{,}1526$. Como observado

no caso da série periódica, as condições que favorecem a aprendizagem são opostas, com valores de κ mais altos do que de β. Para investigar essa relação do desempenho com os parâmetros, e consequentemente com a dinâmica do reservatório, escolhemos outros dois pontos da figura 45: $P1$, onde mantivemos $\kappa = 0{,}1526$ e zeramos β; e $P2$, no qual, com β zerado, aumentamos a força de acoplamento para $\kappa = 1{,}5000$. A tabela 18 contém os pares de parâmetros GA, $P1$ e $P2$, assim como os erros associados ao desempenho do PRC com essas configurações. Comparando, notamos que nesse caso o melhor desempenho é obtido com um par de parâmetros com β predominante, característica que levava a aprendizagem da série senoide ao pior desempenho (dentre os pontos escolhidos da figura 40). Em contrapartida, o pior desempenho, aqui, ocorre quando κ predomina em relação a β, região do espaço de parâmetros onde, para a série senoide, o PRC apresentava melhores desempenhos.

Tabela 18 – Parâmetros do modelo e o erro resultante.

Ponto	κ	β	**Erro**
GA	0,1526	1,3421	0,179962
$P1$	0,1526	0,0000	0,278290
$P2$	1,5000	0,0000	0,321908

Como já foi mencionado, o fator determinante para a avaliação do desempenho, neste caso, em vez de a emulação da série de referência, iteração a iteração, é a manutenção da sua distribuição. Portanto, no lugar das séries temporais, apresentamos na figura 47 as proporções de elementos em cada seção. As colunas roxas estão relacionada à série de referência, enquanto as distribuições das seções da saída do PRC, até as iterações 200, 400 e 600, são representadas pelas colunas azuis, verdes e laranjas, respectivamente. A perda de desempenho nos painéis 47(b) e 47(c) não é tão evidente quanto nas séries temporais da figura 42. Porém, ao olharmos com atenção para as colunas verdes e laranjas, as quais correspondem às iterações posteriores ao intervalo de aprendizado, é possível notar que há uma maior deformação da distribuição da série de referência (colunas roxas), principalmente das colunas das extremidades, onde há menos elementos e essas diferenças se tornam mais visíveis.

Em concordância com o que já foi observado, se compararmos as figuras 40 e 45, notamos que a região do espaço de parâmetros que favorece o bom desempeho do PRC, para o caso da série periódica, é ruim para a aprendizagem da série gaussiana, enquanto o oposto também é verdadeiro. Já que os três pontos de cada espaço de parâmetros estão próximos (se compararmos as duas figuras), observamos que as figuras 43 e 48 apresentam parâmetros de ordem semelhantes. Entretanto, o painel 43(a), o qual é referente ao par de parâmetros selecionado pelo GA, é semelhante ao painel 48(c), que por sua vez está relacionado ao ponto $P2$, o qual apresenta o pior desempenho entre os pontos escolhidos. Já o painel 43(c), que é referente ao ponto $P2$ (pior desempenho dentre os pontos propostos

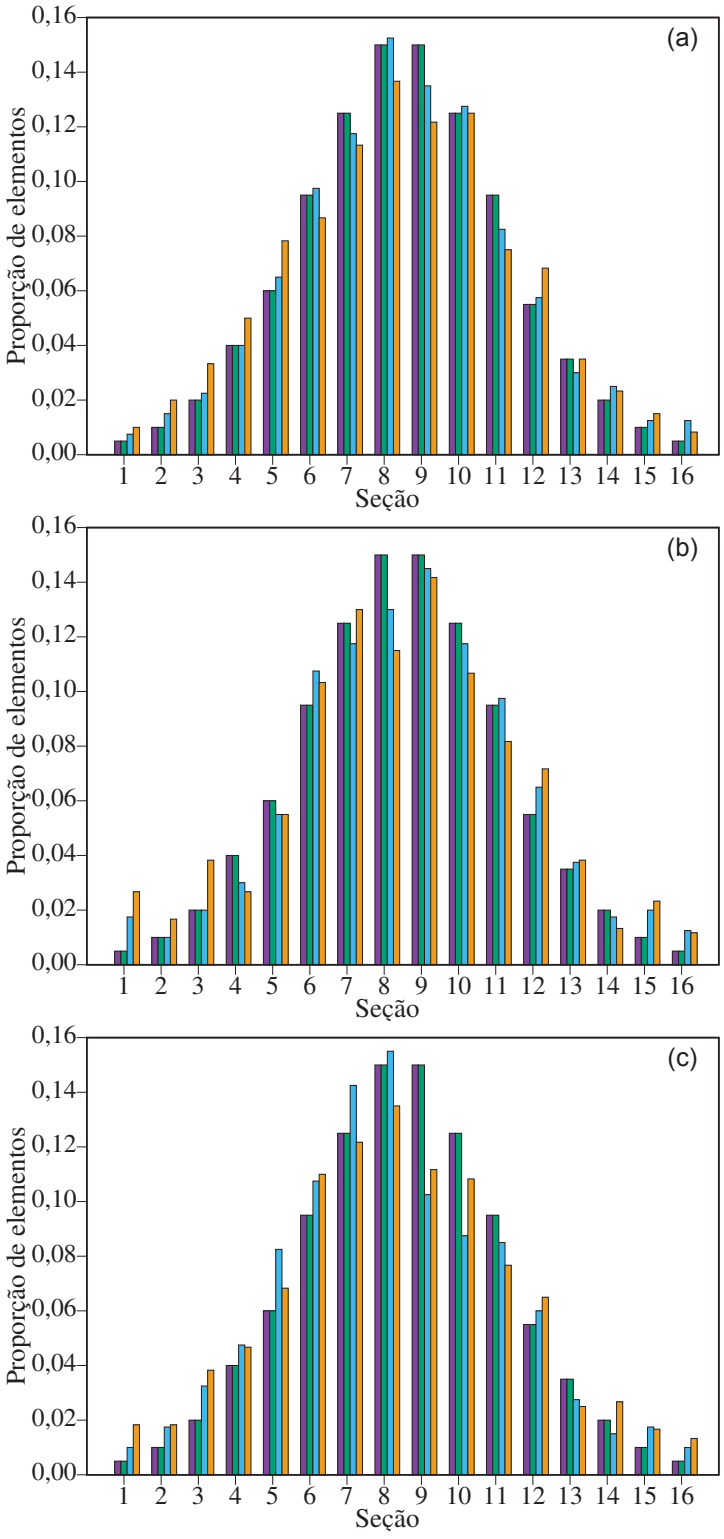


Figura 47 – Histogramas com o número de elementos em cada seção da série de referência (coluna roxa) e da saída do PRC nos intervalos de treinamento (coluna verde), validação (coluna azul) e teste (coluna laranja). Os painéis estão associados aos pares de parâmetros GA(a), $P1$(b) e $P2$(c).

para essa tarefa), é semelhante ao painel 48(a) gerado com o ponto GA. Os painéis 43(b) e 48(b), que estão relacionados aos pontos $P1$ de ambos os espaços de parâmetros, são

semelhantes.

Com os parâmetros selecionados pelo GA, conforme apresentado na seção 4, o parâmetro de ordem da rede é $r = 0{,}80$. Com o coeficiente do termo de campo médio (β) nulo, reduzimos também o grau de sincronização, tornando $r = 0{,}06$. Nessa configuração, como pode ser observado na tabela 18 e no painel 47(b), há um perda de desempenho do PRC. Mantendo $\beta = 0{,}0000$ e tomando $\kappa = 1{,}5000$, aumentamos um pouco, novamente, a homogeneidade na rede, quantificada pelo parâmetro de ordem $r = 0{,}18$. Esse par de parâmetros levou o PRC a apresentar o pior desempenho dentre os pontos investigados (tabela 18 e painel 47(c)).

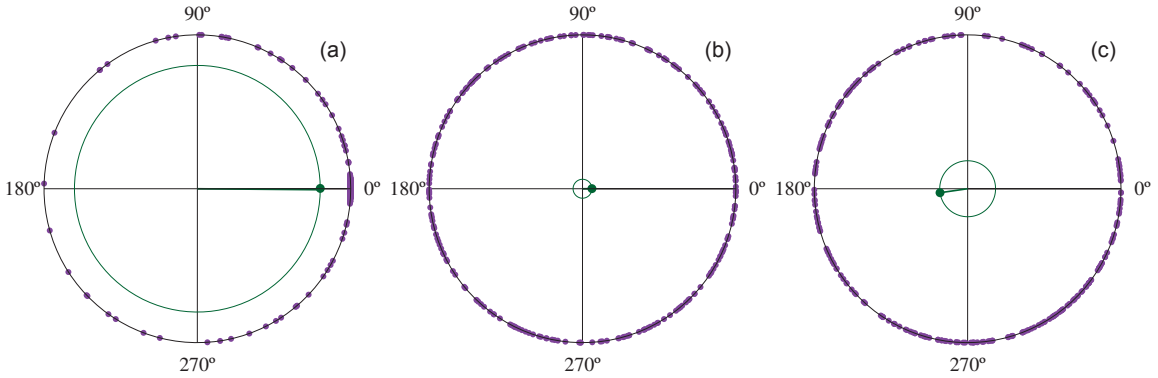


Figura 48 – Representação das fases dos osciladores em uma dada iteração, em roxo, cuja média é representada em verde. O círculo externo é unitário, enquanto o círculo interno tem raio igual ao parâmetro de ordem r. Em (a), os parâmetros do sistema que descreve a dinâmica da rede de osciladores de Kuramoto foram selecionados pelo GA, em (b), β foi zerado, e em (c), β permaneceu nulo, enquanto tomamos $\kappa = 1{,}5000$.

Analogamente às análises da sincronização, os espectros dos LE apresentam similaridades, porém com as sequências invertidas (painéis 44(a), (b) e (c), semelhantes aos painéis 49(c), (b) e (a), respectivamente). O painel 49(a) é refente à dinâmica do sistema com os parâmetros selecionados pelo GA, $\kappa = 0{,}1526$ e $\beta = 1{,}3421$, onde há alguns LE positivos, um é nulo, e a maioria negativos, como notamos, ao longo deste capítulo, ser característico conforme o valor de β aumenta. O painel 49(b) foi gerado com o mesmo $\kappa = 0{,}1526$, porém devido ao β nulo, nota-se que há só LE nulos e negativos, pois praticamente não há acoplamento entre os osciladores da rede. Na figura 49, obtivemos os LE distribuídos da maneira que normalmente se encontra na literatura (pois é equivalente a tirarmos o termo de campo médio, o qual foi proposto por nós neste trabalho), com poucos LE positivos, alguns nulos, e a maior parte negativos.

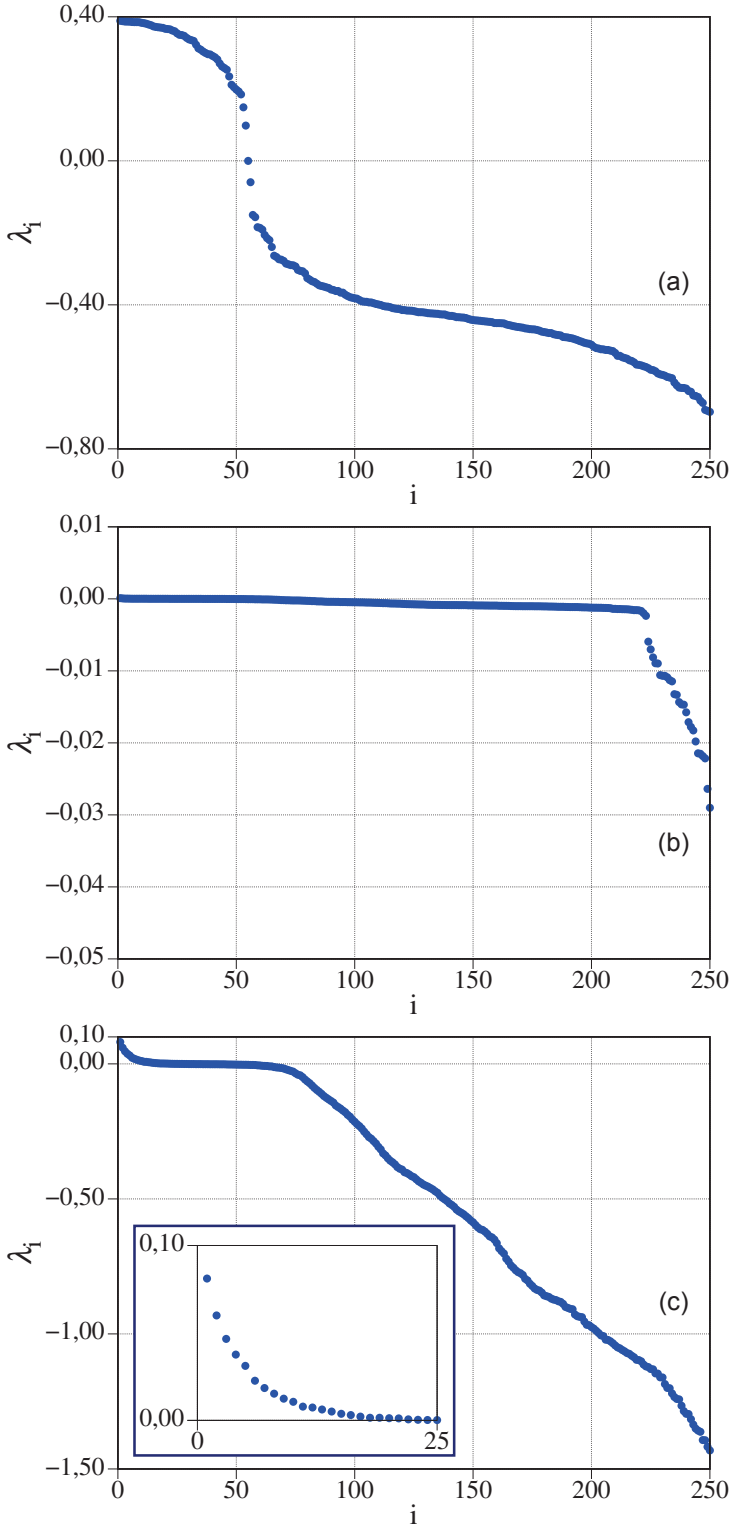


Figura 49 – Espectro de Lyapunov para o sistema de Kuramoto, em ordem decrescente. Em (a), os parâmetros do sistema que descreve a dinâmica da rede de osciladores de Kuramoto foram selecionados pelo GA, em (b), β foi zerado, e em (c), β permaneceu nulo, enquanto tomamos $\kappa = 1{,}5000$.

6. Conclusão

Nesta tese tivemos como objetivo ampliar a compreensão dos processos intermediários do aprendizado de máquina, em vez de buscar obter a melhor performance em determinadas tarefas. Para isso, propusemos um modelo de computação de reservatório físico, no qual as unidades da rede neural que compõem o reservatório são osciladores de Kuramoto, e suas fases os valores que, somados, resultam na saída. Investigamos, assim, o comportamento do reservatório com os parâmetros que levaram o PRC a apresentar os melhores desempenhos na aprendizagem de séries de referência, dentro das variações propostas. Além disso, alteramos os parâmetros e testamos novamente o desempenho do PRC, para o aprendizado de algumas das séries de referência utilizadas durante o trabalho, e investigamos a dinâmica do reservatório sob essas condições, com o intuito de reconhecer características que relacionem a dinâmica ao desempenho.

É comum, nas publicações da área de computação de reservatório, a escolha de séries de referência que sejam semelhantes às entradas externas. Nesta tese, entretanto, escolhemos não propôr entradas externas, mantendo o reservatório com a mesma configuração para todas as tarefas. Dessa maneira, ajustamos somente os pesos de conexão entre as unidades do reservatório e a saída, e os valores dos parâmetros, que foram selecionados para cada tarefa com o recurso do algoritmo genético.

Utilizamos dois métodos para avaliar o desempenho: o primeiro avalia a capacidade do PRC na emulação e na predição das série de referência (no trecho de validação), iteração por iteração; enquanto o segundo, o quanto o PRC consegue manter a distribuição, isso é, o número de elementos em cada classe da série de referência (no trecho de treinamento).

Para as tarefas cujo método de avaliação de desempenho é do primeiro tipo, observamos o nosso modelo de PRC demonstrou maior habilidade computacional quando o parâmetro de ordem é pequeno ($r \approx 0{,}20$, ou menor), demonstrando baixos graus de sincronização na rede. Nesse caso, há poucos LE positivos, alguns nulos, e a maior partes desses é negativa, indicando que há hipercaoticidade na rede. O reservatório apresentou dinâmicas com essas propriedades quando o coeficiente do termo de acoplamento (κ), característico dos sistemas de Kuramoto, foi predominante em relação ao coeficiente do termo de campo médio (β).

Para as demais tarefas, entretanto, demonstramos que um reservatório com alto grau de sincronização é favorável ao aprendizado ($r \approx 0{,}80$). Nesse caso, vários LE são positivos, apenas um é nulo, e a maior parte desses é negativa. Isso demonstra que, assim como no caso anterior, há hipercaoticidade na rede de osciladores, porém há ainda maior

instabilidade na dinâmica, pois há um número maior de direções tangenciais à trajetória, a cada ponto, nas quais duas condições iniciais próximas se afastam ao longo do tempo de maneira exponencial. Desta vez, essas características foram observadas quando houve predominância do coeficiente β em relação ao κ.

Não é intuitivo um reservatório com comportamento caótico favorecer o aprendizado de uma série de referência como a trigonométrica, por exemplo. Contudo, provavelmente isso possa ser explicado pelo fato de que, ao haver caos, deve haver maior diversidade e variabilidade nas fases dos osciladores ao longo do tempo. Tendo em vista que essas são os nós do reservatório, as quais somadas de maneira ponderada compõem a saída, uma maior diversidade nos valores das fases do osciladores corresponde a uma saída capaz de aprender séries de referência mais complexas. Essa hipótese é reforçada ao notarmos que, nas tarefas onde o desempenho do PRC é medido de acordo com sua capacidade de manter a distribuição, as séries de referência são geradas por funções aleatórias, e com os parâmetros selecionados pelo GA, como um conjunto que leva o PRC a um bom desempenho, a dinâmica da rede apresenta vários LE positivos. Enquanto isso, quando o aprendizado é mensurado a partir da capacidade do PRC em emulação e predição, as séries de referência são regidas pelo mapa logístico e por séries trigonométricas, as quais são menos complexas que aquelas geradas aleatoriamente, e ao mesmo tempo, os parâmetros selecionados pelo GA levam o reservatório a apresentar menos LE positivos que o caso anterior.

Outro ponto que merece destaque é a relação direta e contraintuitiva entre a sincronização da rede e a estabilidade do comportamento do reservatório. Ao decorrer deste trabalho, observamos que os conjuntos de parâmetros que levam a rede a apresentar altos graus de sincronização, apresentam também mais LE positivos do que quando há menor coerência entre as fases dos osciladores. Quando há a predominância do termo de acoplamento, a forma de distribuição dos LE é semelhante ao que encontramos na literatura que aborda sistemas de Kuramoto. Entretanto, um maior número de LE positivos e maiores parâmetros de ordem são observados quando há a predominância do termo de campo médio, o qual foi proposto por nós para este trabalho. Uma hipótese, que pode explicar esse fenômeno, é que grande parte dos osciladores possam estar sincronizados, devido à forma matemática que esse termo foi escrito, estando relacionados aos LE negativos, enquanto outros, os quais não estão fortemente ligados ao comportamento coletivo (tendo em vista que são 250 osciladores ao todo), possam contribuir para o surgimento de hipercaoticidade na rede, levando o reservatório a apresentar, simultaneamente, alto grau de sincronização e alguns LE positivos. Porém, como é esperado, a maior parte dos LE, assim como a soma total dos LE, são negativas.

O algoritmo genético provou ser um método eficiente para a seleção de bons conjuntos de parâmetros em todas as tarefas propostas durante esta tese, demonstrando,

também, baixo custo computacional.

Por fim, resultados sugerem que é possível construir um reservatório sem a presença de entradas externas. Entretanto, há a necessidade de resultados mais robustos para a confirmação dessa hipótese. Para a continuação deste trabalho, é interessante que sejam feitos testes com outras séries de referência mais complexas, sejam utilizadas mais amostras para treinar a rede e avaliar o desempenho, e sejam propostos outros sistemas físicos para compôr o reservatório. Além disso, é importante que se busque obter desempenhos ainda melhores, tanto mediante as alterações citadas, quanto adaptações no algoritmo genético para seleção de melhores parâmetros, ou ainda, a partir de outras técnicas. Um método que pode ser testado é o de multiplexação de sinal [122], por exemplo, onde um mesmo sinal pode ser dividido em intervalos iguais, semelhante ao que chamamos de iteração. Entretanto, a cada iteração durante este trabalho, tomamos apenas um valor de fase por oscilador para servir como nós para as unidades do reservatório. Uma proposta é que dividamos esse intervalo em sub-intervalos igualmente espaçados, e a cada um desses tomemos os valores das fases. Desta maneira, pode-se obter um número maior do que apenas um nó por oscilador a cada iteração (considerando o modo que definimos o termo iteração). Finalmente, para demonstração da aplicabilidade dos métodos e das técnicas apresentados, recomendamos que sejam feitos testes com PRCs com reservatórios que sejam compostos por sistemas físicos reais, cujas medidas dos valores de estados possam ser tomadas diretamente a partir do experimento.

Referências

1 BALLARD, D. H.; BROWN, C. M. *Computer Vision*. Prentice-Hall, 1981. Disponível em: ⟨https://books.google.com.br/books?id=EfRRAAAAMAAJ⟩.

2 JOSHI, A. K. Natural language processing. *Science*, v. 253, n. 5025, 1991. ISSN 00368075. Disponível em: ⟨https://www.jstor.org/stable/2879169⟩.

3 JORDAN, M. I.; MITCHELL, T. M. Machine learning: Trends, perspectives, and prospects. *Science*, v. 349, n. 6245, 7 2015. ISSN 0036-8075. Disponível em: ⟨https://www.science.org/doi/10.1126/science.aaa8415⟩.

4 SAMUEL, A. L. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, v. 3, n. 3, p. 210–229, 1959. Disponível em: ⟨https://doi.org/10.1147/rd.33.0210⟩.

5 TURING, A. M. Computing Machinery and Intelligence. *Mind*, LIX, n. 236, 10 1950. ISSN 1460-2113. Disponível em: ⟨https://academic.oup.com/mind/article/LIX/236/433/986238⟩.

6 MITCHELL, T. M. *Machine Learning*. 1. ed. USA: McGraw-Hill, Inc., 1997. ISBN 0070428077. Disponível em: ⟨https://books.google.com.br/books?id=EoYBngEACAAJ⟩.

7 HAYKIN, S. *Redes Neurais: Princípios e Prática*. Bookman Editora, 2001. ISBN 9788577800865. Disponível em: ⟨https://books.google.com.br/books?id=bhMwDwAAQBAJ⟩.

8 RUSSEL, S.; NORVIG, P. *Artificial intelligence: A modern approach*. [s.n.], 2012. ISSN 0269-8889. Disponível em: ⟨https://books.google.com.br/books?id=8jZBksh-bUMC⟩.

9 RUINEIHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. *Learning Internal Representations Error Propagation*. [S.l.], 1986. v. 1, n. V. Disponível em: ⟨https://ieeexplore.ieee.org/document/6302929⟩.

10 JORDAN, M. I. Chapter 25 Serial order: A parallel distributed processing approach. In: *Advances in Psychology*. [s.n.], 1997. v. 121, n. C. Disponível em: ⟨https://www.sciencedirect.com/science/article/abs/pii/S0166411597801112⟩.

11 LUKOŠEVIČIUS, M.; JAEGER, H. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, v. 3, n. 3, 2009. ISSN 15740137. Disponível em: ⟨https://www.sciencedirect.com/science/article/abs/pii/S1574013709000173⟩.

12 JAEGER, H. Tutorial on training recurrent neural networks, covering bppt, rtrl, ekf and the echo state network approach. *GMD - Forschungszentrum Informations technik, 2002.*, v. 5, 01 2002. Disponível em: ⟨https://www.ai.rug.nl/minds/uploads/ESNTutorialRev.pdf⟩.

13 MAASS, W.; NATSCHLÄGER, T.; MARKRAM, H. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, v. 14, n. 11, 2002. ISSN 08997667. Disponível em: ⟨https://pubmed.ncbi.nlm.nih.gov/12433288/⟩.

14 COULOMBE, J. C.; YORK, M. C.; SYLVESTRE, J. Computing with networks of nonlinear mechanical oscillators. *PLoS ONE*, v. 12, n. 6, 2017. ISSN 19326203. Disponível em: ⟨https://journals.plos.org/plosone/articleid=10.1371/journal.pone.0178663⟩.

15 URBAIN, G. et al. Morphological properties of mass-spring networks for optimal locomotion learning. *Frontiers in Neurorobotics*, v. 11, n. MAR, 2017. ISSN 16625218. Disponível em: ⟨https://www.frontiersin.org/articles/10.3389/fnbot.2017.00016/full⟩.

16 NIKOLIĆ, D. et al. Distributed fading memory for stimulus properties in the primary visual cortex. *PLoS Biology*, v. 7, n. 12, 2009. ISSN 15449173. Disponível em: ⟨https://journals.plos.org/plosbiology/article?id=10.1371/journal.pbio.1000260⟩.

17 DRANIAS, M. R. et al. Short-term memory in networks of dissociated cortical neurons. *Journal of Neuroscience*, v. 33, n. 5, 2013. ISSN 02706474. Disponível em: ⟨https://www.jneurosci.org/content/33/5/1940⟩.

18 NAKAJIMA, K. et al. Boosting Computational Power through Spatial Multiplexing in Quantum Reservoir Computing. *Physical Review Applied*, v. 11, n. 3, 2019. ISSN 23317019. Disponível em: ⟨https://journals.aps.org/prapplied/abstract/10.1103/PhysRevApplied.11.034021⟩.

19 GHOSH, S. et al. Quantum reservoir processing. *npj Quantum Information*, v. 5, n. 1, 2019. ISSN 20566387. Disponível em: ⟨https://www.nature.com/articles/s41534-019-0149-8⟩.

20 APPELTANT, L. et al. Information processing using a single dynamical node as complex system. *Nature Communications*, v. 2, n. 1, 2011. ISSN 20411723. Disponível em: ⟨https://www.nature.com/articles/ncomms1476⟩.

21 ALOMAR, M. L. et al. Digital Implementation of a Single Dynamical Node Reservoir Computer. *IEEE Transactions on Circuits and Systems II: Express Briefs*, v. 62, n. 10, 2015. ISSN 15583791. Disponível em: ⟨https://ieeexplore.ieee.org/document/7161321⟩.

22 KURAMOTO, Y. Self-entrainment of a population of coupled non-linear oscillators. *Lecture Notes in Physics*, Springer, v. 39, p. 420–422, 1975. Disponível em: ⟨https://ui.adsabs.harvard.edu/abs/1975LNP....39..420K/abstract⟩.

23 BREAKSPEAR, M.; HEITMANN, S.; DAFFERTSHOFER, A. Generative models of cortical oscillations: Neurobiological implications of the Kuramoto model. *Frontiers in Human Neuroscience*, v. 4, 2010. ISSN 16625161. Disponível em: ⟨https://www.frontiersin.org/articles/10.3389/fnhum.2010.00190/full⟩.

24 HOLLAND, J. H. *Adaptation in natural and artificial systems*. University of Michigan Press, 1975. Disponível em: ⟨https://mitpress.mit.edu/9780262581110/adaptation-in-natural-and-artificial-systems/⟩.

25 JONG, K. A. D. *An analysis of the behavior of a class of genetic adaptive systems*. Tese (Doutorado) — University of Michigan, 1975. Disponível em: ⟨https://deepblue.lib.umich.edu/handle/2027.42/4507⟩.

26 GOLDBERG, D. E. *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley Longman Publishing Co., Inc., 1989. Disponível em: ⟨https://books.google.com.br/books/about/Genetic_Algorithms_in_Search_Optimizatio.html?id=2IIJAAAACAAJ&redir_esc=y⟩.

27 KATOCH, S.; CHAUHAN, S. S.; KUMAR, V. A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications*, v. 80, n. 5, 2021. ISSN 15737721. Disponível em: ⟨https://link.springer.com/article/10.1007/s11042-020-10139-6⟩.

28 SHOUGAT, M. R. E. et al. A Hopf physical reservoir computer. *Scientific Reports*, v. 11, n. 1, 2021. ISSN 20452322. Disponível em: ⟨https://www.nature.com/articles/s41598-021-98982-x⟩.

29 SHOUGAT, M. R. E. U.; LI, X.; PERKINS, E. Dynamic effects on reservoir computing with a Hopf oscillator. *Physical Review E*, American Physical Society, v. 105, n. 4, p. 044212, 4 2022. ISSN 2470-0045. Disponível em: ⟨https://journals.aps.org/pre/abstract/10.1103/PhysRevE.105.044212⟩.

30 GADELHA, J. A evolução dos computadores. 2001. Disponível em: ⟨http://www.ic.uff.br/~aconci/evolucao.html⟩.

31 FREETH, T. et al. Decoding the ancient Greek astronomical calculator known as the Antikythera Mechanism. *Nature*, v. 444, n. 7119, 11 2006. ISSN 0028-0836. Disponível em: ⟨https://www.nature.com/articles/nature05357⟩.

32 TANONAKA, E. M. *Régua de cálculo: uma contribuição de William Oughtred para a matemática*. Tese (Doutorado) — Pontifícia Universidade Católica de São Paulo, São Paulo, 10 2008. Disponível em: ⟨https://tede2.pucsp.br/handle/handle/13397⟩.

33 GOLDSTINE, H. H. *The Computer from Pascal to Von Neumann*. USA: Princeton University Press, 1972. ISBN 0691081042. Disponível em: ⟨https://books.google.com.br/books?id=3FvELn2KiUYC⟩.

34 COSTA, E. B. L. d. *Charles Babbage (1791-1871) e a mecanização do cálculo: das engrenagens à "máquina de pensar"*. Tese (Doutorado) — Pontifícia Universidade Católica de São Paulo, São Paulo, 10 2012. Disponível em: ⟨https://tede2.pucsp.br/bitstream/handle/13273/1/Eli\%20Banks\%20Liberato\%20da\%20Costa.pdf⟩.

35 FILHO, C. *História da computação: O Caminho do Pensamento e da Tecnologia*. Edipucrs. ISBN 9788574306919. Disponível em: ⟨https://books.google.com.br/books?id=3FvELn2KiUYC⟩.

36 ROJAS, R. Konrad zuse's legacy: the architecture of the z1 and z3. *IEEE Annals of the History of Computing*, v. 19, n. 2, p. 5–16, 1997. Disponível em: ⟨https://ieeexplore.ieee.org/document/586067⟩.

37 COHEN, I. B. *Mark I, Harvard*. GBR: John Wiley and Sons Ltd., 2003. 1078–1080 p. ISBN 0470864125. Disponível em: ⟨https://books.google.com.br/books?id=yQ9LAQAAIAAJ⟩.

38 FILHO, G. de S.; ALEXANDRE, E. de S. M. *Introdução a Computação*. Editora da UFPB, 2014. ISBN 9788523708924. Disponível em: ⟨https://docplayer.com.br/143123955-Introducao-a-computacao.html⟩.

39 OGAWA, S. et al. Brain magnetic resonance imaging with contrast dependent on blood oxygenation. *Proceedings of the National Academy of Sciences*, National Academy of Sciences, v. 87, n. 24, p. 9868–9872, 1990. Disponível em: ⟨https://www.ncbi.nlm.nih.gov/pmc/articles/PMC55275/⟩.

40 CABEZA, R.; NYBERG, L. Imaging cognition ii: An empirical review of 275 pet and fmri studies. *Journal of cognitive neuroscience*, MIT Press, v. 12, n. 1, p. 1–47, 2000. Disponível em: ⟨https://pubmed.ncbi.nlm.nih.gov/10769304/⟩.

41 CRAIK, K. *The Nature of Explanation*. Cambridge University Press, 1952. (Philosophy: science CAM). ISBN 9780521094450. Disponível em: ⟨https://books.google.com.br/books?id=wT04AAAAIAAJ⟩.

42 KELEMEN, J. From artificial neural networks to emotion machines with marvin minsky. *Acta Polytechnica Hungarica*, v. 4, 12 2007. Disponível em: ⟨http://acta.uni-obuda.hu/Kelemen_12.pdf⟩.

43 KOVÁCS, Z. *Redes Neurais Artificiais*. LIVRARIA DA FISICA, 2002. ISBN 9788588325142. Disponível em: ⟨https://books.google.com.br/books?id=O0nLxR67wmUC⟩.

44 MINSKY, M.; PAPERT, S. *Perceptrons: an Introduction to Computational Geometry*. MIT Press, 1969. ISBN 9780262630221. Disponível em: ⟨https://books.google.com.br/books?id=Ow1OAQAAIAAJ⟩.

45 HORVITZ, E. J.; BREESE, J. S.; HENRION, M. Decision theory in expert systems and artificial intelligence. *International Journal of Approximate Reasoning*, v. 2, n. 3, 7 1988. ISSN 0888613X. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/0888613X8890120X⟩.

46 BANKO, M.; BRILL, E. Scaling to very very large corpora for natural language disambiguation. In: *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics - ACL '01*. Morristown, NJ, USA: Association for Computational Linguistics, 2001. Disponível em: ⟨https://aclanthology.org/P01-1005/⟩.

47 HAYS, J.; EFROS, A. A. Scene completion using millions of photographs. *Communications of the ACM*, v. 51, n. 10, 10 2008. ISSN 0001-0782. Disponível em: ⟨http://graphics.cs.cmu.edu/projects/scene-completion/⟩.

48 HALEVY, A.; NORVIG, P.; PEREIRA, F. The unreasonable effectiveness of data. *IEEE Intelligent Systems*, v. 24, n. 2, 2009. ISSN 15411672. Disponível em: ⟨https://s3-us-west-2.amazonaws.com/ieeeshutpages/xplore/xplore-shut-page.html⟩.

49 THRUN, S. et al. Stanley: The robot that won the DARPA Grand Challenge. *Journal of Field Robotics*, v. 23, n. 9, 2006. ISSN 15564959. Disponível em: ⟨http://robots.stanford.edu/papers/thrun.stanley05.pdf⟩.

50 GRIGORESCU, S. et al. A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics*, v. 37, n. 3, 2020. ISSN 15564967. Disponível em: ⟨https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21918⟩.

51 JONSSON, A. K. et al. Planning in Interplanetary Space: Theory and practice. In: *AIPS*. [s.n.], 2000. Disponível em: ⟨http://www.ai.mit.edu/courses/6.834J-f01/rax-planner.pdf⟩.

52 BRESINA, J. L.; MORRIS, P. H. Mixed-initiative planning in space mission operations. *AI Magazine*, v. 28, n. 2, 2007. ISSN 07384602. Disponível em: ⟨https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/2041⟩.

53 GOODMAN, D.; KEENE, R. Man Versus Machine: Kasparov Versus Deep Blue. *ICGA Journal*, v. 20, n. 3, 2018. ISSN 13896911. Disponível em: ⟨https://content.iospress.com/articles/icga-journal/icg20-3-08⟩.

54 GOODMAN, J.; HECKERMAN, D. Fighting spam with statistics. *Significance*, v. 1, n. 2, 2004. ISSN 17409713. Disponível em: ⟨https://www.pnas.org/doi/10.1073/pnas.1906831117⟩.

55 ZWEBEN, M. et al. Intelligent Scheduling. *Technology*, v. 6, n. 2, 1994. Disponível em: ⟨https://catalogue.nla.gov.au/Record/364723⟩.

56 BRANTS, T. et al. Large language models in machine translation. In: *EMNLP-CoNLL 2007 - Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. [s.n.], 2007. Disponível em: ⟨https://aclanthology.org/D07-1090/⟩.

57 MAHATA, S. K.; DAS, D.; BANDYOPADHYAY, S. MTIL2017: Machine translation using recurrent neural network on statistical machine translation. *Journal of Intelligent Systems*, v. 28, n. 3, 2019. ISSN 03341860. Disponível em: ⟨https://www.degruyter.com/document/doi/10.1515/jisys-2018-0016/html⟩.

58 LOBO, L. C. Inteligência Artificial e Medicina. *Revista Brasileira de Educação Médica*, v. 41, n. 2, 2017. ISSN 0100-5502. Disponível em: ⟨https://www.scielo.br/j/rbem/a/f3kqKJjVQJxB4985fDMVb8b/abstract/?lang=pt⟩.

59 SIAU, K.; WANG, W. *Building trust in artificial intelligence, machine learning, and robotics*. 2018. Disponível em: ⟨https://www.cutter.com/article/building-trust-artificial-intelligence-machine-learning-and-robotics-498981⟩.

60 BEJGER, S.; ELSTER, S. Artificial Intelligence in economic decision making: how to assure a trust? *Ekonomia i Prawo*, v. 19, n. 3, 2020. ISSN 1898-2255. Disponível em: ⟨https://apcz.umk.pl/EiP/article/view/EiP.2020.028⟩.

61 COSGRIFF, C. V. et al. The clinical artificial intelligence department: A prerequisite for success. *BMJ Health and Care Informatics*, v. 27, n. 1, 2020. ISSN 26321009. Disponível em: ⟨https://informatics.bmj.com/content/27/1/e100183⟩.

62 BAILLIE, C. A. et al. The readmission risk flag: Using the electronic health record to automatically identify patients at risk for 30-day readmission. *Journal of Hospital Medicine*, v. 8, n. 12, 2013. ISSN 15535592. Disponível em: ⟨https://pubmed.ncbi.nlm.nih.gov/24227707/⟩.

63 DAVIS, S. E. et al. Calibration Drift Among Regression and Machine Learning Models for Hospital Mortality. *AMIA ... Annual Symposium proceedings. AMIA Symposium*, v. 2017, 2017. ISSN 1942597X. Disponível em: ⟨https://pubmed.ncbi.nlm.nih.gov/29854127/⟩.

64 SHAH, N. D.; STEYERBERG, E. W.; KENT, D. M. *Big data and predictive analytics: Recalibrating expectations*. 2018. Disponível em: ⟨https://jamanetwork.com/journals/jama/article-abstract/2683125⟩.

65 GENNATAS, E. D. et al. Expert-augmented machine learning. *Proceedings of the National Academy of Sciences of the United States of America*, v. 117, n. 9, 2020. ISSN 10916490. Disponível em: ⟨https://www.pnas.org/doi/10.1073/pnas.1906831117⟩.

66 GLESS, S.; SILVERMAN, E.; WEIGEND, T. If robots cause harm, who is to blame? Self-driving cars and criminal liability. *New Criminal Law Review*, v. 19, n. 3, 2016. ISSN 19334206. Disponível em: ⟨http://euro.ecom.cmu.edu/program/law/08-732/AI/Gless.pdf⟩.

67 ZAVRŠNIK, A. Criminal justice, artificial intelligence systems, and human rights. *ERA Forum*, v. 20, n. 4, 2020. ISSN 18639038. Disponível em: ⟨https://link.springer.com/article/10.1007/s12027-020-00602-0⟩.

68 SINGER, P. *Wired for War: The Robotics Revolution and Conflict in the Twenty-first Century*. Penguin Press, 2009. (A Penguin Book. Technology/Military Science). ISBN 9781594201981. Disponível em: ⟨https://books.google.com.br/books?id=AJuowQmtbU4C⟩.

69 BOYLE, M. J. The legal and ethical implications of drone warfare. *International Journal of Human Rights*, v. 19, n. 2, 2015. ISSN 1744053X. Disponível em: ⟨https://www.routledge.com/Legal-and-Ethical-Implications-of-Drone-Warfare/Boyle/p/book/9780367139100⟩.

70 ETZIONI, A. Are New Technologies the Enemy of Privacy? *Knowledge, Technology & Policy*, v. 20, n. 2, 2007. ISSN 0897-1986. Disponível em: ⟨https://link.springer.com/article/10.1007/s12130-007-9012-x⟩.

71 LATHAM, R.; VINGE, V. The coming technological singularity: How to survive in a post-human era. In: *Science Fiction Criticism*. [s.n.], 2020. Disponível em: ⟨https://ntrs.nasa.gov/citations/19940022856⟩.

72 GOOD, I. J. Speculations Concerning the First Ultraintelligent Machine. In: . [s.n.], 1966. Disponível em: ⟨https://www.sciencedirect.com/science/article/abs/pii/S0065245808604180⟩.

73 PORTER, A. Bioethics and transhumanism. *Journal of Medicine and Philosophy (United Kingdom)*, v. 42, n. 3, 2017. ISSN 17445019. Disponível em: ⟨https://academic.oup.com/jmp/article/42/3/237/3817401⟩.

74 MEAD, W. R.; KURZWEIL, R. The Singularity Is near: When Humans Transcend Biology. *Foreign Affairs*, v. 85, n. 3, 2006. ISSN 00157120. Disponível em: ⟨https://www.foreignaffairs.com/reviews/capsule-review/2006-05-01/singularity-near-when-humans-transcend-biology⟩.

75 MOOR, J. H. *The nature, importance, and difficulty of machine ethics*. 2006. Disponível em: ⟨https://ieeexplore.ieee.org/document/1667948⟩.

76 MOOR, J. H. Are there decisions computers should never make? In: *Computer Ethics*. [s.n.], 2017. Disponível em: ⟨https://dl.acm.org/doi/abs/10.5555/2569.2676⟩.

77 CHOWDHURY, G. G. Natural language processing. *Annual Review of Information Science and Technology*, v. 37, 2003. ISSN 00664200. Disponível em: ⟨https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/aris.1440370103⟩.

78 LIDDY, E. D. *Natural Language Processing. In Encyclopedia of Library and Information Science*. 2001. Disponível em: ⟨https://surface.syr.edu/istpub/63/⟩.

79 POGGIO, T.; TORRE, V.; KOCH, C. Computational vision and regularization theory. *Nature*, v. 317, n. 6035, 1985. ISSN 00280836. Disponível em: ⟨https://www.nature.com/articles/317314a0⟩.

80 YAN, X. *Linear Regression Analysis: Theory and Computing*. World Scientific Publishing Company Pte Limited, 2009. ISBN 9789812834119. Disponível em: ⟨https://books.google.com.br/books?id=MjNv6rGv8NIC⟩.

81 Data Science Academy. *Deep Learning Book*. Data Science Academy, 2022. Disponível em: ⟨https://www.deeplearningbook.com.br/⟩.

82 RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. *Nature*, v. 323, n. 6088, 1986. ISSN 00280836. Disponível em: ⟨https://www.nature.com/articles/323533a0⟩.

83 JAEGER, H.; HAAS, H. Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication. *Science*, v. 304, n. 5667, 2004. ISSN 00368075. Disponível em: ⟨https://www.science.org/doi/10.1126/science.1091277⟩.

84 LUKOŠEVIČIUS, M.; JAEGER, H. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, v. 3, n. 3, 2009. ISSN 15740137. Disponível em: ⟨https://www.sciencedirect.com/science/article/abs/pii/S1574013709000173⟩.

85 JAEGER, H. The "echo state" approach to analysing and training recurrent neural networks. *GMD Report*, v. 148, 2001. Disponível em: ⟨https://www.ai.rug.nl/minds/uploads/EchoStatesTechRep.pdf⟩.

86 BRUGNAGO, E. L.; GALLAS, J. A.; BEIMS, M. W. Predicting regime changes and durations in Lorenz's atmospheric convection model. *Chaos*, v. 30, n. 10, 2020. ISSN 10897682. Disponível em: ⟨https://aip.scitation.org/doi/10.1063/5.0013253⟩.

87 BRUGNAGO, E. L. et al. Classification strategies in machine learning techniques predicting regime changes and durations in the Lorenz system. *Chaos*, v. 30, n. 5, 2020. ISSN 10897682. Disponível em: ⟨http://fisica.ufpr.br/mbeims/MLTonyEduardo.pdf⟩.

88 BUONOMANO, D. V.; MERZENICH, M. M. Temporal information transformed into a spatial code by a neural network with realistic properties. *Science*, v. 267, n. 5200, 1995. ISSN 00368075. Disponível em: ⟨https://www.science.org/doi/10.1126/science.7863330⟩.

89 HAEUSLER, S.; MAASS, W. A statistical analysis of information-processing properties of lamina-specific cortical microcircuit models. *Cerebral Cortex*, v. 17, n. 1, 2007. ISSN 10473211. Disponível em: ⟨https://pubmed.ncbi.nlm.nih.gov/16481565/⟩.

90 KARMARKAR, U. R.; BUONOMANO, D. V. Timing in the Absence of Clocks: Encoding Time in Neural Network States. *Neuron*, v. 53, n. 3, 2007. ISSN 08966273. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0896627307000256⟩.

91 STANLEY, G. B.; LI, F. F.; DAN, Y. Reconstruction of natural scenes from ensemble responses in the lateral geniculate nucleus. *Journal of Neuroscience*, v. 19, n. 18, 1999. ISSN 02706474. Disponível em: ⟨https://www.jneurosci.org/content/19/18/8036⟩.

92 KISTLER, W. M.; ZEEUW, C. I. D. Dynamical working memory and timed responses: The role of reverberating loops in the olivo-cerebellar system. *Neural Computation*, v. 14, n. 11, 2002. ISSN 08997667. Disponível em: ⟨https://direct.mit.edu/neco/article-abstract/14/11/2597/6648/Dynamical-Working-Memory-and-Timed-Responses-The?redirectedFrom=fulltext⟩.

93 YAMAZAKI, T.; TANAKA, S. The cerebellum as a liquid state machine. *Neural Networks*, v. 20, n. 3, 2007. ISSN 08936080. Disponível em: ⟨https://www.sciencedirect.com/science/article/abs/pii/S0893608007000366⟩.

94 TANAKA, G. et al. Recent advances in physical reservoir computing: A review. *Neural Networks*, v. 115, 2019. ISSN 18792782. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0893608019300784⟩.

95 NATSCHLÄGER, T.; MARKRAM, H.; MAASS, W. Computer Models and Analysis Tools for Neural Microcircuits. In: *Neuroscience Databases*. [s.n.], 2003. Disponível em: ⟨https://link.springer.com/chapter/10.1007/978-1-4615-1079-6_9⟩.

96 MAASS, W.; NATSCHL, T.; MARKRAM, H. Computational Models for Generic Cortical Microcircuits A Conceptual Framework for Real-Time Neural Computation. *Computational Neuroscience: A Comprehensive Approach*, v. 18, n. 6009, 2004. ISSN 10959203. Disponível em: ⟨https://igi-web.tugraz.at/people/maass/psfiles/149-v05.pdf⟩.

97 VERSTRAETEN, D. et al. Isolated word recognition with the Liquid State Machine: A case study. *Information Processing Letters*, v. 95, n. 6 SPEC. ISS., 2005. ISSN 00200190. Disponível em: ⟨https://www.sciencedirect.com/science/article/abs/pii/S0020019005001523⟩.

98 VINCENT-LAMARRE, P.; LAJOIE, G.; THIVIERGE, J. P. Driving reservoir models with oscillations: a solution to the extreme structural sensitivity of chaotic networks. *Journal of Computational Neuroscience*, v. 41, n. 3, 2016. ISSN 15736873. Disponível em: ⟨https://link.springer.com/article/10.1007/s10827-016-0619-3⟩.

99 ENEL, P. et al. Reservoir Computing Properties of Neural Dynamics in Prefrontal Cortex. *PLoS Computational Biology*, v. 12, n. 6, 2016. ISSN 15537358. Disponível em: ⟨https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1004967⟩.

100 BARAK, O. et al. From fixed points to chaos: Three models of delayed discrimination. *Progress in Neurobiology*, v. 103, 2013. ISSN 03010082. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0301008213000129⟩.

101 DOMINEY, P. F. Complex sensory-motor sequence learning based on recurrent state representation and reinforcement learning. *Biological Cybernetics*, v. 73, n. 3, 1995. ISSN 03401200. Disponível em: ⟨https://link.springer.com/article/10.1007/BF00201428⟩.

102 DOMINEY, P. F. Recurrent temporal networks and language acquisition—from corticostriatal neurophysiology to reservoir computing. *Frontiers in Psychology*, v. 4, 2013. ISSN 1664-1078. Disponível em: ⟨https://www.frontiersin.org/articles/10.3389/fpsyg.2013.00500/full⟩.

103 HINAUT, X.; DOMINEY, P. F. Real-Time Parallel Processing of Grammatical Structure in the Fronto-Striatal System: A Recurrent Network Simulation Study Using

Reservoir Computing. *PLoS ONE*, v. 8, n. 2, 2013. ISSN 19326203. Disponível em: ⟨https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0052946⟩.

104 STROGATZ, S. *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering*. Levant Books, 2007. (Studies in nonlinearity). ISBN 9788187169857. Disponível em: ⟨https://books.google.com.br/books?id=PHmED2xxrE8C⟩.

105 OTT, E. *Chaos in Dynamical Systems*. Cambridge University Press, 2002. ISBN 9780521010849. Disponível em: ⟨https://books.google.com.br/books?id=nOLx--zzHSgC⟩.

106 CLEMSON, P. T.; STEFANOVSKA, A. Discerning non-autonomous dynamics. *Physics Reports*, v. 542, n. 4, 2014. ISSN 03701573. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0370157314001318⟩.

107 WINFREE, A. T. Biological rhythms and the behavior of populations of coupled oscillators. *Journal of Theoretical Biology*, v. 16, n. 1, 7 1967. ISSN 00225193. Disponível em: ⟨https://www.sciencedirect.com/science/article/abs/pii/0022519367900513⟩.

108 WINFREE, A. *The Geometry of Biological Time*. Springer New York, 1980. ISBN 9780387989921. Disponível em: ⟨https://books.google.com.br/books?id=FGnyCAAAQBAJ⟩.

109 KURAMOTO, Y. Self-entrainment of a population of coupled non-linear oscillators. In: *International Symposium on Mathematical Problems in Theoretical Physics*. [s.n.], 2005. Disponível em: ⟨https://link.springer.com/chapter/10.1007/BFb0013365⟩.

110 KURAMOTO, Y. *Chemical Oscillations, Waves, and Turbulence*. Dover Publications, 1980. ISBN 9780486428819. Disponível em: ⟨https://books.google.com.br/books?id=tcTyCAAAQBAJ⟩.

111 WIESENFELD, K.; COLET, P.; STROGATZ, S. H. Frequency locking in Josephson arrays: Connection with the Kuramoto model. *Physical Review E - Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, v. 57, n. 2, 1998. ISSN 1063651X. Disponível em: ⟨https://journals.aps.org/pre/abstract/10.1103/PhysRevE.57.1563⟩.

112 KOZYREFF, G.; VLADIMIROV, A. G.; MANDEL, P. Global coupling with time delay in an array of semiconductor lasers. *Physical Review Letters*, v. 85, n. 18, 2000. ISSN 00319007. Disponível em: ⟨https://www.wias-berlin.de/people/vladimir/papers/Vladimirov29.pdf⟩.

113 WANG, W.; KISS, I. Z.; HUDSON, J. L. Experiments on arrays of globally coupled chaotic electrochemical oscillators: Synchronization and clustering. *Chaos*, v. 10, n. 1, 2000. ISSN 10541500. Disponível em: ⟨https://aip.scitation.org/doi/10.1063/1.166470⟩.

114 WANG, W.; KISS, I. Z.; HUDSON, J. L. Clustering of arrays of chaotic chemical oscillators by feedback and forcing. *Physical Review Letters*, v. 86, n. 21, 2001. ISSN 00319007. Disponível em: ⟨https://www.semanticscholar.org/paper/Clustering-of-arrays-of-chaotic-chemical-by-and-Wang-Kiss/e4449e81a4ad3831e29224cbc342d2234da1b529⟩.

115 BUCK, J. Synchronous rhythmic flashing of fireflies. II. *Quarterly Review of Biology*, v. 63, n. 3, 1988. ISSN 00335770. Disponível em: ⟨https://www.jstor.org/stable/2830425⟩.

116 MARDIA, K. V. *Statistics of Directional Data*. Academic Press, 1972. (Probability and Mathematical Statistics a Series of Monographs and Textbooks). ISBN 9780124711501. Disponível em: ⟨https://books.google.com.br/books?id=fG7c-yvRs38C⟩.

117 HAKEN, H. *Synergetics: An Introduction : Nonequilibrium Phase Transitions and Self-organization in Physics, Chemistry, and Biology*. Springer, 1983. (Recent Results in Cancer Research, v. 3). ISBN 9783540088660. Disponível em: ⟨https://books.google.com.br/books?id=09PuAAAAMAAJ⟩.

118 TASS, P. A. *Phase Resetting in Medicine and Biology: Stochastic Modelling and Data Analysis*. Springer Berlin Heidelberg, 2007. (Springer Series in Synergetics). ISBN 9783540381617. Disponível em: ⟨https://books.google.com.br/books?id=OKeAbImYmRsC⟩.

119 MAY, R. M. Simple mathematical models with very complicated dynamics. *Nature*, Nature Publishing Group, v. 261, n. 5560, p. 459–467, 1976. Disponível em: ⟨https://www.nature.com/articles/261459a0⟩.

120 BOX, G. E. P.; MULLER, M. E. A Note on the Generation of Random Normal Deviates. *The Annals of Mathematical Statistics*, Institute of Mathematical Statistics, v. 29, n. 2, p. 610 – 611, 1958. Disponível em: ⟨https://doi.org/10.1214/aoms/1177706645⟩.

121 NAGOURNEY, W.; SANDBERG, J.; DEHMELT, H. Shelved optical electron amplifier: Observation of quantum jumps. *Physical Review Letters*, v. 56, n. 26, 1986. ISSN 00319007.

122 NAKAJIMA, K. et al. Boosting computational power through spatial multiplexing in quantum reservoir computing. *Phys. Rev. Appl.*, American Physical Society, v. 11, p. 034021, Mar 2019. Disponível em: ⟨https://link.aps.org/doi/10.1103/PhysRevApplied.11.034021⟩.

Apêndices

A. Pseudocódigo do modelo de PRC

- Abaixo, o modelo de PRC com osciladores de Kuramoto desenvolvido por nós. As sub-rotinas que possuem as letras "S_A" no início de seus nomes estão relacionadas à aprendizagem, enquanto "S_D" aos dados. Todas são seguidas de comentários explicando suas funções. Somente algumas das sub-rotinas são apresentadas em detalhes, uma vez que o foco deste apêndice é tornar mais claras as etapas do algoritmo genético.

Início PRC

Parâmetros:
seleção = 10 ▷ Espécimes escolhidas para serem genitoras da próxima geração.
espécimes = 50 ▷ Número total de espécimes a cada geração.
geração = 50 ▷ Número de vezes que são geradas novas espécimes.

Variáveis:
ii

CHAMAR S_D_CarregaSerie ▷ Carrega a série de referência.
CHAMAR AtribuiIntervaloCromossomos ▷ Atribui limites inferior e superior aos cromossomos (parâmetros ou variáveis a serem selecionados pelo GA).
CHAMAR S_A_IniciaPopulacao ▷ Sorteia valores iniciais para todos os cromossomos de cada espécime dentro dos intervalos estipulados na sub-rotina "AtribuiIntervaloCromossomos".

CHAMAR S_A_GerenciaEtapaAprendizado(1, seleção) ▷ Sub-rotina descrita abaixo deste pseudocódigo.
CHAMAR S_A_GerenciaEtapaAjuste(1, seleção) ▷ Sub-rotina descrita abaixo deste pseudocódigo.
CHAMAR S_A_CalculaErroNormalizado(1, seleção) ▷ Detalhes acerca do modo de cálculo de erro na seção 4.2.
CHAMAR S_A_OrdenaSelecao ▷ Ordena as espécimes de maneira decrescente de acordo com seus desempenhos.

PARA ii = 1 **ATÉ** geracoes **FAÇA**
  CHAMAR S_A_GerenciaEtapaAprendizado(seleção + 1, espécimes)

22: **CHAMAR** S_A_GerenciaEtapaAjuste(seleção + 1, espécimes)
23: **CHAMAR** S_A_CalculaErroNormalizado(seleção + 1, espécimes)
24: **CHAMAR** S_A_SelecionaEspecime ▷ Reordena as espécimes de maneira decrescente de acordo com seus desempenhos e seleciona as "seleção" melhores.
25: **CHAMAR** S_A_NovaGeracao ▷ Cada uma das espécimes restantes são geradas a partir
26: **FIM PARA**
27:
28: **CHAMAR** S_A_ProlongaSerie ▷ Sub-rotina descrita abaixo deste pseudocódigo.
29: **CHAMAR** S_A_SalvaSerieSelecao ▷ Salva em um arquivo os valores saída gerados pela melhor espécime após 50 gerações para os três intervalos (treinamento, validação, teste).
30: **Fim PRC**

- Sub-rotina "S_A_GerenciaEtapaAprendizado":

1: **SUB-ROTINA** S_A_GerenciaEtapaAprendizado(inicioEspecime, fimEspecime)
2:
3: **Parâmetros (entrada):** inicioEspecime, fimEspecime
4: **Variáveis:** i
5:
6: **PARA** i **DE** inicioEspecime **ATÉ** fimEspecime **FAÇA**
7: **CHAMAR** S_A_GeraSerieOsciladores(i) ▷ Evolui o sistema de Kuramoto para obter as fases dos osciladores, as quais serão utilizadas como unidades do reservatório. Recomenda-se indicar que aqui a quantidade de fases geradas para cada oscilador deve ser correspondente ao tamanho da amostra de treinamento.
8: **CHAMAR** S_A_AprendizadoPerceptron(i) ▷ Treinamento/aprendizado do PRC (atualização dos pesos de saída).
9: **FIM PARA**

- Sub-rotina "S_A_GerenciaEtapaAjuste":

1: **SUB-ROTINA** S_A_GerenciaEtapaAjuste(inicioEspecime, fimEspecime)
2:
3: **Parâmetros (entrada):** inicioEspecime, fimEspecime
4: **Variáveis:** i
5:
6: **PARA** i **DE** inicioEspecime **ATÉ** fimEspecime **FAÇA**
7: **CHAMAR** S_A_GeraSerieOsciladores(i) ▷ Evolui o sistema de Kuramoto para obter as fases dos osciladores, as quais serão utilizadas como unidades

do reservatório. Recomenda-se indicar que aqui a quantidade de fases geradas para cada oscilador deve ser correspondente ao tamanho da amostra de validação.

8: **CHAMAR** S_A_AmpliaSerieSaida(i) ▷ Com os pesos aprendidos, gera os valores de saída. Recomenda-se indicar que aqui a quantidade de valores de saída deve ser equivalente ao tamanho da amostra de validação.

9: **FIM PARA**

- Sub-rotina "S_A_ProlongaSerie":

1: **SUB-ROTINA** S_A_ProlongaSerie

2:

3: **Variáveis:** i

4:

5: i = 1 ▷ Seleciona a melhor espécime para gerar os valores de saída no intervalo de teste.

6: **CHAMAR** S_A_GeraSerieOsciladores(i) ▷ Evolui o sistema de Kuramoto para obter as fases dos osciladores, as quais serão utilizadas como unidades do reservatório. Recomenda-se indicar que aqui a quantidade de fases geradas para cada oscilador deve ser correspondente ao tamanho da amostra de teste.

7: **CHAMAR** S_A_AmpliaSerieSaida(i) ▷ Com os pesos aprendidos, gera os valores de saída. Recomenda-se indicar que aqui a quantidade de valores de saída deve ser equivalente ao tamanho da amostra de teste.