SISTEMA PREDITIVO BASEADO EM REDES RECORRENTES PARA CONTROLE DE REATORES NUCLEARES DE PESQUISA

Paulo Caixeta de Oliveira

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Engenharia Nuclear, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Nuclear.

Orientador: Roberto Schirru

Rio de Janeiro

Março de 2019

SISTEMA PREDITIVO BASEADO EM REDES RECORRENTES PARA CONTROLE DE REATORES NUCLEARES DE PESQUISA

Paulo Caixeta de Oliveira

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA NUCLEAR.

Examinada por:

_____
Prof. Roberto Schirru, D.Sc.

_____
Prof.ª Andressa dos Santos Nicolau, D.Sc.

_____
Prof. Cesar Marques Salgado, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
MARÇO DE 2019

Oliveira, Paulo Caixeta de

Sistema Preditivo Baseado Em Redes Recorrentes Para Controle De Reatores Nucleares De Pesquisa/ Paulo Caixeta de Oliveira – Rio de Janeiro: UFRJ/COPPE, 2019.

XII, 83 pp.: il.: 29,7 cm

Orientador: Roberto Schirru

Dissertação (mestrado) – UFRJ/ COPPE/ Programa de Engenharia Nuclear, 2019.

Referências Bibliográficas: p. 78-81

1. Inteligência Artificial. 2. Redes Recorrentes. 3. Otimização. I. Schirru, Roberto. II Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Nuclear. III. Título.

AGRADECIMENTOS

Aos meus padrinhos, tios, meu irmão, minha avó, e toda família, que sempre me ofereceram todo o apoio necessário. Em especial, à minha mãe, que sempre fez o impossível por nós.

A todos os colegas pesquisadores, colegas de trabalho, professores e funcionários que viabilizam meu aprendizado e me apoiaram.

Ao meu orientador, professor Schirru, pela brilhante orientação, estímulo e apoio.

À Nathália, meu amor, pelo estímulo e compreensão, e que supera comigo todas as dificuldades do caminho.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

SISTEMA PREDITIVO BASEADO EM REDES RECORRENTES PARA CONTROLE DE REATORES NUCLEARES DE PESQUISA

Paulo Caixeta de Oliveira
Março/ 2019

Orientador: Roberto Schirru

Programa: Engenharia Nuclear

O controle de reatores nucleares é feito por sistemas que monitoram diversas variáveis simultaneamente, porém apenas apontam a ocorrência de um acidente, sem identificar seu tipo ou causa. O poder de previsão de que um acidente possa ocorrer baseado na evolução das variáveis atuais do reator pode ser uma ferramenta bastante útil para auxiliar o trabalho dos operadores do reator, aumentando não só sua confiança, mas também a disponibilidade geral da planta. As redes *Echo State Network* (ESN) são Redes Neurais Recorrentes (RNR) apropriadas para a formulação de sistemas preditivos, com a vantagem de utilizarem um algoritmo de treinamento muito mais simples e mais rápido que as RNR convencionais. Porém, para utilização das ESN em sistemas preditivos não é necessário especificar a estrutura do modelo, porém alguns parâmetros desta rede devem ser trabalhados e ajustados para se obter um bom desempenho. Este trabalho propõe a utilização de Inteligência Artificial para otimização dos parâmetros da ESN, afim de se obter o melhor desempenho possível. Foi utilizada a Otimização por Enxame de Partículas (PSO - *Particle Swarm Optimization*), método que utiliza a chamada inteligência de enxame, que evolui a população em busca dos melhores resultados. A rede ESN com seu treinamento otimizado pelo PSO foi utilizada para acompanhar o funcionamento de um reator nuclear, com dados reais de funcionamento, assim como identificar em tempo real se algum acidente está ocorrendo, bem como qual acidente está ocorrendo. Os resultados obtidos demonstram que a rede pode identificar e prever, o estado do reator, seja este em condição normal ou de acidente.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

PREDICTIVE SYSTEM BASED ON RECURRENT NETWORKS FOR RESEARCH NUCLEAR REACTORS CONTROL

Paulo Caixeta de Oliveira
March / 2019

Advisors: Roberto Schirru

Department: Nuclear Engineering

The control of nuclear reactors is done by systems that monitor several variables simultaneously, but only indicate the occurrence of an accident, without identifying its type or cause. The possibility of prediction of an accident based on the evolution of the current variables of the reactor can be a very useful tool to assist the work of the reactor operators, increasing their confidence, as well as the general availability of the plant. Echo State Networks (ESN) are Recurrent Neural Networks (RNNs) suitable for the formulation of predictive systems, with the advantage of using a training algorithm that is simpler and faster than conventional RNNs. To use Echo State Networks in predictive systems it is not necessary to specify the structure of the model, but some parameters of this network must be tuned for good performance. This work proposes the use of Artificial Intelligence to optimize ESN's parameters, to improve performance. Particle Swarm Optimization (PSO), a method that uses so-called swarm intelligence, was used. The ESN optimized by the PSO was used to monitor the operation of a nuclear reactor (with real operating data), as well as to identify in real time if an accident is occurring, as well which accident is occurring. The results show that the network can identify and predict the state of the reactor, be it in normal or accident condition.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Símbolos

ACO – *Ant Colony Optimization*
AE – Algoritmo Evolucionário
CR – Computação de Reservatório
DP – Desvio Padrão
EA – *Evolutionary Algorithm*
EMA – Erro Médio Absoluto
ESN – *Echo State Network*
FAL – *Final Actuation Logic*
FRPS – *First Shutdown System*
GA – *Genetic Algorith*
GB - *GigaByte*
IAEA – *Internetional Atomic Energy Agency*
LOCA – *Loss Of Coolant Accident*
LSM – *Liquid State Machine*
MI/MO – *Multiple Input/Multiple Output*
MSE – *Mean Squared Error*
Mv – Média dos Valores
MW – *MegaWatt*
PSO – *Particle Swarm Optimization*
RAM – *Random Access Memory*
RD – Reservatório Dinâmico
RMB – Reator Multipropósito Brasileiro
RMSE – *Root of Mean Squared Error*
RN – Reatores Nucleares
RNA – Rede Neural Artificial
RNN – *Recurrent Neural Network*
RPS – *Reactor Protection System*
SGTR – *Steam Generator Tube Rupture*
SICA – Sistema de Controle de Angra 1
SRPS – *Second Shutdown System*
TRIP – Desligamento de Emergência
XOR – Portão *"ou"* Exclusivo

1 - Introdução

Os sistemas de proteção dos Reatores Nucleares (RN) são bastante robustos e consistentes, mas foram planejados apenas para reagir quando as variáveis de segurança atingem seus limites, para evitar a ocorrência de acidentes e seu agravamento. Se tais sistemas tivessem um recurso de previsão, aumentariam a disponibilidade geral das usinas, reduzindo a ocorrência (atualmente imprevisível) de desligamentos de emergência (TRIP).

Previsão, porém, é um tema de pesquisa bastante desafiador, especialmente quando se trata de problemas complexos e não-lineares. Esses problemas estão presentes na mais ampla gama de aplicações relacionadas a muitos campos, como preços de mercado de ações, condições climáticas, nível de umidade do solo, posicionamento e rastreamento de robôs (Chouikhi *et al.*, 2017), bem como comportamento de RN (Martins *et al.*, 2015).

Redes Neurais Artificiais (RNA) têm sido utilizadas como sistemas de predição para problemas não lineares (Haykin, 1998), sendo as Redes Neurais Recorrentes (*Recurrent Neural Networks* - RNN) as mais apropriadas para previsão, por apresentarem um comportamento naturalmente dinâmico. Elas são, no entanto, difíceis de treinar e exigem uma profunda compreensão do problema que está sendo modelado. Além disto, os estudos que se tem sobre a utilização de RNN como sistemas preditivos, porém, estão concentrados basicamente em modelos off-line (que realizam uma previsão estática com dados existentes, e que posteriormente o modelo encontrado é utilizado para alguma previsão), enquanto o modelo desejado é um modelo online, que seja capaz de receber dados em tempo real e, juntamente com dados históricos, atualizar a previsão para dizer qual será o comportamento do sistema em períodos de tempo futuros.

As *Echo State Networks* (ESN) (Jaeger, 2001) são RNN com algoritmos de treinamento mais simples e rápidos, pois não exigem uma especificação detalhada da estrutura do problema. Isso requer, no entanto, ajustes de alguns parâmetros. Estes parâmetros são bastante sensíveis e requerem tratamento adequado para que o ESN forneça bons resultados (Chouikhi, *et al.*, 2017). Uma forma encontrada com frequência na literatura é a otimização destes parâmetros utilizando algoritmos de inteligência artificial (Goldberg & Holland, 1988; Jiang, Berry, & Schoenauer, 2008).

A Otimização por Enxame de Partículas (*Particle Swarm Optimization* - PSO) (Kennedy & Eberhart, 1995) é uma das aplicações mais populares dos algoritmos de inteligência artificiais para otimização, pela sua eficiência (Waintraub et al., 2009). Este processo de otimização baseia-se principalmente no comportamento de grupos de animais (como pássaros e peixes) e sua coletividade de ações em direção a um objetivo comum.

O PSO foi utilizado para essa tarefa pois é eficiente na resolução de problemas complexos (Nicolau & Schirru, 2011; Meneses *et. al*, 2007) e na otimização de parâmetros de RNNs (Chouikhi et al., 2017; Liu, Wang, & Wang, 2015; Jiang, Berry, & Schoenauer, 2008; Martins G. P., 2015; Rueda, 2014), seu tempo de convergência é reduzido e possui relativa facilidade de implementação (Carlisle & Dozier, 2010).

A proposta deste trabalho é a criação de um sistema preditivo que receba dados de funcionamento de um reator nuclear (em tempo real) e avalie se a evolução das variáveis levará a uma condição que requeira um desligamento de emergência (TRIP), por parte do sistema de proteção. Caso não seja possível evitar o TRIP, o sistema deve identificar o mais rápido possível qual acidente está ocorrendo (outra característica que os sistemas de proteção atuais não possuem).

Para isto, utilizou-se redes neurais do tipo ESN, com seus parâmetros otimizados por PSO, que foram treinadas com dados do simulador da usina nuclear de potência Angra 1, da Central Nuclear Almirante Álvaro Aberto. Apesar de ser um simulador, os dados gerados por ele reproduzem fielmente o comportamento das variáveis da planta, gerando os mesmos dados que seriam vistos durante a operação real do reator.

Para que a rede seja capaz de prever a ocorrência de um TRIP, primeiro ela tem que ser treinada para reconhecer desvios da operação normal que levem a estes acidentes. Este treinamento envolve fornecer à rede dados de funcionamento normal do reator, assim como dados de como cada variável se comporta durante cada acidente. A tabela 1 apresenta todas as variáveis obtidas pelo simulador, além dos acidentes simulados.

Devido ao alto custo computacional de se treinar a rede para todas as variáveis e acidentes, decidiu-se realizar este treinamento com 1 entrada e 1 saída, na qual a rede acompanha uma única variável ao longo do tempo. Tal método será melhor descrito no capítulo 3.

Tabela 1 Lista de Acidentes e Variáveis (Alvarenga, 1997)

Variáveis:	Acidentes (Sigla):
Vazão do núcleo (%)	Perda de alimentação elétrica externa (BLACKOUT)
Temperatura da perna quente (°C)	Perda de refrigerante do sistema primário (LOCA)
Temperatura da perna fria (°C)	Ruptura de tubos do gerador de vapor (SGTR)
Vazão do núcleo (Kg/s)	Ruptura da alimentação principal (MFWBR)
Nível do gerador de vapor - faixa larga (%)	Desligamento da turbina (TRIPTUR)
Nível do gerador de vapor - faixa estreita (%)	Isolamento da alimentação principal e auxiliar (MEFWISO)
Pressão no gerador de vapor (MPa)	Isolamento da alimentação principal (MFWISO)
Vazão de água de alimentação (Kg/s)	Isolamento da linha de vapor principal (MSTMISO)
Vazão de vapor (Kg/s)	Ruptura da linha de vapor principal (STMLIBR)
Vazão de ruptura (Kg/s)	Perda de alimentação elétrica sem desligamento do reator (BALCKSEM)
Vazão no circuito primário (Kg/s)	Ruptura de alimentação principal sem desligamento do reator (MFWBRSEM)
Tempo (s)	Isolamento da linha de vapor principal sem desligamento do reator (MSTMISEM)
Pressão no sistema primário (Mpa)	Isolamento da alimentação principal e auxiliar sem desligamento do reator (MEFWISEM)
Potência térmica (%)	Isolamento da alimentação principal sem desligamento do reator (MFWISEM)
Potência nuclear (%)	Desligamento da turbina sem desligamento do reator (TRIPTURSEM)
Margem de sub-resfriamento (°C)	Condição normal de potência (NORMAL)
Nível do pressurizador (%)	
Temperatura média no primário (°C)	

Uma vez a rede treinada, desenvolveu-se um algoritmo para simular o envio e recebimento de variáveis em tempo real do reator, no qual a parte que recebe os dados não sabe em qual estado o reator se encontra (funcionamento normal ou acidente), e deve identificá-lo e prever a evolução da variável ao longo do tempo.

1.1 Organização da Dissertação

Este trabalho propõe um sistema preditivo baseado em uma ESN otimizada por PSO, para o controle de reatores nucleares. Ele se encontra dividido em 5 partes, descritas a seguir.

Neste capítulo foi apresentada uma breve introdução ao tema e os sistemas preditivos para controle de reatores nucleares, assim como as metodologias aplicadas para sua formulação.

O capítulo 2 apresenta a fundamentação teórica e a revisão bibliográfica. Nele, serão descritas o que se pode ser encontrado na literatura sobre o tema, assim como linhas de estudo a serem desenvolvidas e a teoria e justificativa de escolha dos métodos empregados.

No capítulo 3, é apresentada a metodologia utilizada no trabalho, representando a aplicação do que foi visto até aqui em um método capaz de levar a alcançar o objetivo.

No capítulo 4, estão os resultados e discussões e, no capítulo 5, as conclusões.

2 – Fundamentação Teórica

2.1 Revisão Bibliográfica

2.1.1 Conceitos de Sistemas Preditivos

Um modelo preditivo, por definição (Guazzelli, 2018), é uma função matemática capaz de aprender o mapeamento entre um conjunto de variáveis de entrada de dados, geralmente agrupadas em um registro, e uma variável de resposta ou de destino. Quanto mais repetitivo, previsível e/ou conhecido for o sistema estudado, e quanto melhor for o treinamento do modelo, melhores serão seus resultados.

Um sistema preditivo é obtido por um processo chamado de análise preditiva (Nyce, 2017), e é um termo amplo que agrupa o uso de um conjunto de técnicas estatísticas e analíticas para desenvolver modelos capazes de prever comportamentos ou eventos. São muito usados, por exemplo, para modelos de mapeamento meteorológico, modelos econômicos etc (Chouikhi *et al.*, 2017).

Sistemas preditivos são encontrados com frequência em equipamentos mecânicos de alto valor agregado (como por exemplo, grandes motores, como os utilizados nos sitstemas de refrigeração principal das usinas), onde sua manutenção preventiva é muito mais barata e simples de se realizar do que sua substituição, ou também para peças (como válvulas, por exemplo) cuja falha repentina acarretaria em grandes prejuízos econômicos e/ou para a saúde e segurança dos trabalhadores. Estes sistemas ainda são bastante estudados (Tavassoli, 2018), além de serem produtos comercializados por diversas empresas (Asea Brown Boveri, 2018) - inclusive contando com opções contendo modelos baseados em redes neurais artificiais. Uma ilustração de um sistema preditivo se encontra na Figura 1.

No caso do funcionamento de um reator, todos os estados das suas variáveis podem ser calculados e/ou aproximados a partir de outras variáveis, e pode-se usar este mapeamento para detectar desvios durante a operação antes que levem a um desligamento de emergência (TRIP) do reator.

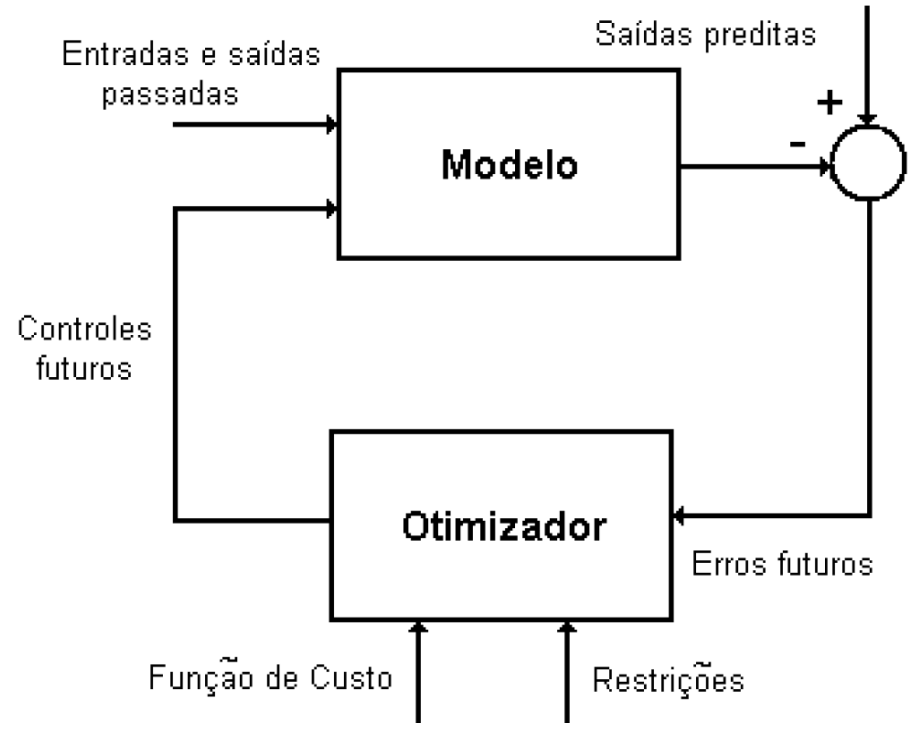


Figura 1 Ilustração de um Sistema Preditivo (Bravo & Normey-Rico, 2009)

2.1.1.1 Processo de Obtenção de um Sistema Preditivo

Embora o processo típico de obtenção de um sistema preditivo varie de acordo com o comportamento ou evento a ser descrito, como regra geral, ao se criar um sistema preditivo, dá-se um algum tipo de pontuação (por exemplo relação e interdependência entre dois fenômenos) aos parâmetros que estão sendo modelados e, no final as pontuações são somadas para que se possa determinar qual o evento/comportamento mais provável de ocorrer. Para este estudo, por exemplo, o evento que se deseja saber a probabilidade de ocorrer é o TRIP do reator.

Neste processo, são seguidas as seguintes etapas:

- Formulação do problema;
- Aquisição dos dados;
- Tratamento dos dados;
- Separação do grupo de treinamento e de teste;
- Desenvolvimento do modelo;
- Teste do modelo.

Na etapa de formulação do problema, deve-se selecionar estrutura matemática sob a qual o problema será abordado. Nesta fase deve ser conhecida a forma do problema (linear ou não-linear, contínua ou discreta etc.), se será tratado o problema de forma completa ou se o mesmo será subdividido, se o problema depende do tempo etc.

Na aquisição de dados, podem ser utilizados tanto dados reais quanto dados simulados/calculados, desde que os mesmos tenham alguma correlação comprovada com a realidade. Após a aquisição, devem-se avaliar os dados obtidos para verificar se não há vício nos dados, se os mesmos são suficientes em quantidade, se há ruídos e/ou erros, entre outros.

Caso seja possível, é ideal que se possa obter dados reais do sistema a ser estudado para que se tenha dados diferentes (porém em mesmo número) para treino e para teste. Se não for o caso, deve-se reservar uma parte dos dados coletados para teste (que pode ser maior, menor ou igual à parte reservada para treino, dependendo do problema); ou ainda, o mesmo grupo de dados pode ser utilizado tanto para treino quanto para teste.

Neste estudo, o problema é não-linear, contínuo e unidimensional e, para teste e treino um mesmo conjunto de 60 pontos para cada acidente, obtidos do simulador da usina de Angra 1 (Mól, 2002).

2.1.2 Sistemas de Controle de Reatores Nucleares

De acordo com a Agência Internacional de Energia Atômica (*International Atomic Energy Agency* – IAEA), os reatores de pesquisa são reatores nucleares utilizados para desenvolvimento, educação e treinamento (*International Atomic Energy Agency*, 2018). Eles produzem nêutrons para aplicações na indústria, medicina, agricultura e medicina forense, entre outros.

Como sua principal função não é a de geração de energia elétrica, reatores de pesquisa tem potência nominal muito inferior à dos reatores de potência. Dependendo da aplicação, a potência pode ser tão baixa quanto 100W (para, por exemplo, treinamento de operadores e estudo de materiais), podendo chegar a 30MW (por exemplo, para produção de radioisótopos e ativação de materiais).

Com a potência menor (reatores de potência chegam a 1500 MW), tem-se condições de trabalho para o fluido refrigerante muito mais brandas, como por exemplo a temperatura de água da piscina a menos de 40°C, e pressão atmosférica. Isto facilita o

controle da operação, diminui os requisitos de segurança e o tamanho do empreendimento como um todo.

Porém, para funções críticas de segurança (como o sistema de proteção do reator – RPS), ainda é necessária a utilização de redundância tripla de medidores, e votação 2 de 3 (2oo3) para a geração do sinal de TRIP. A votação 2oo3 consiste na comparação entre os resultados das três redundâncias, e se pelo menos duas estiverem solicitado o sinal de TRIP, este sinal será enviado e o desligamento de emergência ocorrerá. Este método busca um equilíbrio entre segurança e disponibilidade geral da planta.

A redundância não ocorre apenas na medição da variável (com 3 medidores para cada variável), mas cada sinal de cada transmissor é isolado e replicado para cada um dos 3 processadores do RPS, fazendo com que cada processador receba os 3 sinais originais (um de cada medidor).

Ao chegar no painel de processamento do RPS, cada um dos 3 sinais originais (tanto analógico, como por exemplo temperatura da água, quanto digital, como para detectar uma válvula aberta) é comparado com o limite pré-estabelecido e, caso seja maior, um sinal de ativação de TRIP interno é gerado. Caso 2 ou mais sinais internos sejam gerados, é então enviado um sinal de TRIP desta redundância para a Lógica Final de Atuação (FAL) – Figura 2. Se o FAL receber pelo menos dois pedidos de TRIP, esse pedido é validado e o reator é desligado.

No caso do Reator Multipropósito Brasileiro (RMB), o RPS será divido em dois: o Primeiro Sistema de Proteção do Reator (*First Reactor Protection System* - FRPS), que atuará nas barras de controle; e o Segundo Sistema de Proteção do Reator (*Second Reactor Protection System* - SRPS), que drena parcialmente a água pesada. Cada um medirá variáveis diferentes (ou limites diferentes da mesma variável), com sensores próprios, utilizando a lógica 2oo3. Além disso, uma das condições necessárias para o acionamento do SRPS é a falha do FRPS.

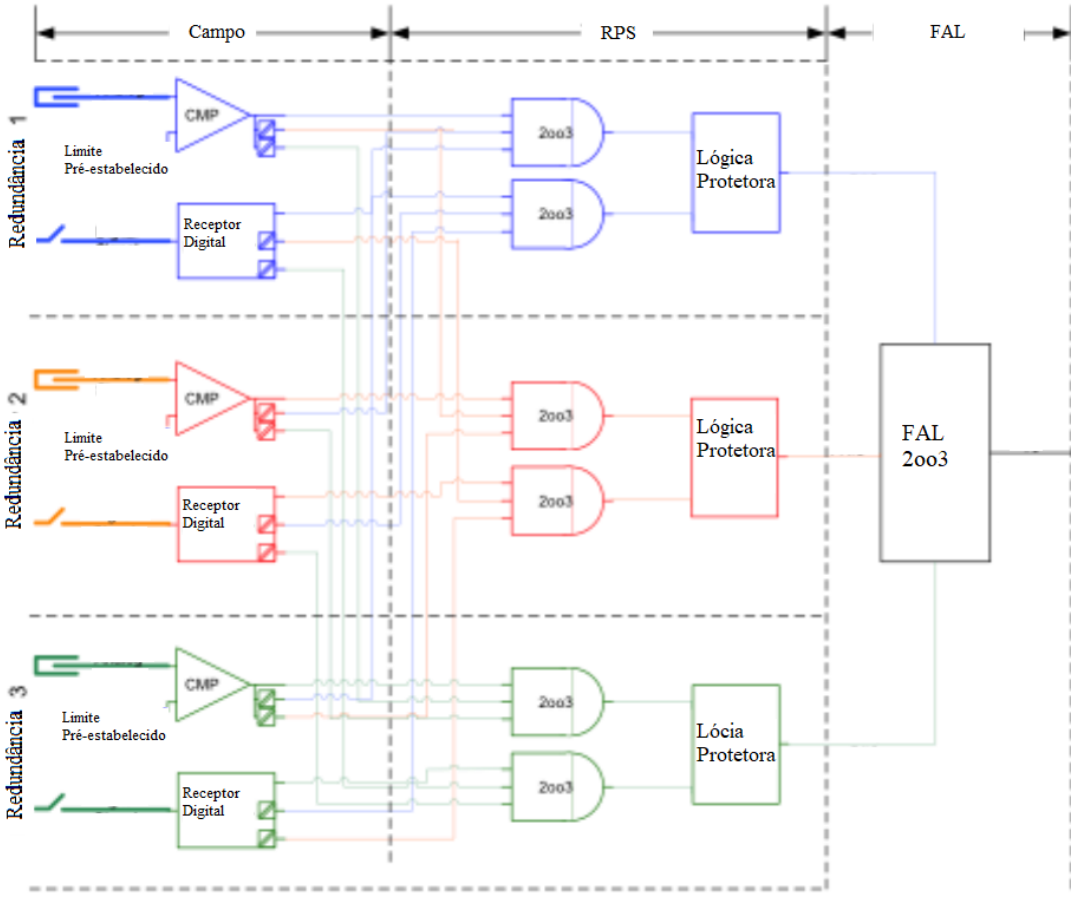


Figura 2 Diagrama de Atuação do FAL (Carvalho et al., 2014)

2.1.3 Redes Neurais Artificiais

Esta seção trata dos conceitos fundamentais das redes neurais recorrentes do tipo ESN, incluindo a descrição de sua estrutura, algoritmo de treinamento e aplicações, assim como o desenvolvimento das redes ao longo do tempo.

2.1.3.1 Introdução

O entendimento de inteligência pelo prisma neurológico pode ser entendido estudando, simplificadamente, o funcionamento e comportamento dos neurônios. O raciocínio humano é construído a partir de decisões em unidades binárias unitárias, organizadas em estruturas hierárquicas; e seu processo de decisão é o que é chamado de "processamento de limiares" (James, 2014). Uma rede é formada, portanto, conectando-se a saída de um neurônio à entrada do próximo, constituindo grafo ponderado.

A primeira referência que se tem de uma proposta de um sistema computacional que simule o raciocínio neurológico foi em 1943 (McCulloch & Pitts, 1943), onde foi

introduzido o conceito de "lógica de limiar" (*threshold logic*), baseado neste processamento de limiares, e utilizados para definir os portões de lógica de limiar (que implementam esta lógica de limiar). Posteriormente, em 1958 foi introduzido o conceito de *perceptrons* (Rosenblatt, 1958), que consiste num algoritmo para reconhecimento de padrões. Porém, apesar de a máquina na qual foi implementada esta teoria exibir resultados promissores, rapidamente se percebeu que a variedade de padrões que a mesma era capaz de identificar era reduzido, uma vez que por se tratar de uma rede de camada única, só conseguia reconhecer padrões lineares.

Em 1959, Minsky e Papert (Minsky & Papert, 1969) escreveram um livro no qual ressaltaram as principais limitações das redes neurais artificiais da época, sendo a principal delas a sua incapacidade de implementação da função ou-exclusivo (XOR), além de o esforço computacional requerido pelas redes ser muito alto para a tecnologia da época. Este primeiro impedimento começou a ser solucionado em 1975 quando foi introduzido o algoritmo de retropropagação aplicado a redes neurais (Werbos, 1975), que consistia em distribuir o erro encontrado de volta através das múltiplas camadas da rede, modificando os pesos de cada nó. Os pesos, por sua vez (assim como as funções de ativação), são modificadas por um processo chamado de aprendizagem, que é governado pela regra de aprendizagem (Tutschku, 1995). Isto tornou possível a implementação da porta XOR de forma eficiente nos computadores existentes. A Figura 3 traz um comparativo entre um neurônio e uma RNA, mostrando como se implementa computacionalmente um neurônio natural.

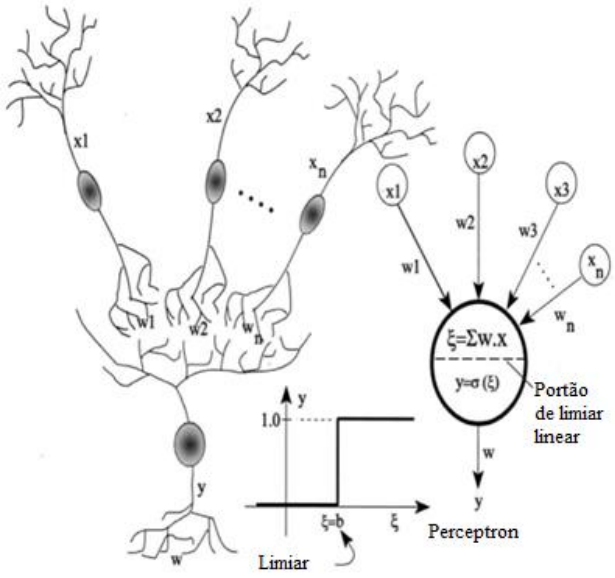


Figura 3 Esquematização de um neurônio (à esquerda) e de uma RNA (à direita) (Bryce, 2018)

2.1.3.1.1 Funcionamento e Componentes da Rede Neural Artificial

Cada dado de entrada, antes de ser inserido no neurônio de entrada, é multiplicado por um peso, que pode ser maior, menor ou igual a 1. Os dados de entrada, após ponderados, são somados em um neurônio de entrada por meio de um somatório simples. Este somatório, passa por uma função de ativação, que vai indicar se este neurônio irá enviar um sinal de saída.

A capacidade de processamento de uma RNA é determinada principalmente pelo tipo, quantidade e estrutura de seus neurônios. Tal estrutura é definida através dos pesos das conexões entre os neurônios. Esta informação a ser transmitida - também chamada de conhecimento - (Catto *et al.*, 2003), é adquirida durante a fase de treinamento, na qual os dados são introduzidos repetidamente na RNA para que, em cada rodada, os pesos sejam ajustados até que a saída seja a desejada.

A seguir, é fornecida uma breve explicação dos componentes de uma RNA citados acima.

2.1.3.1.1.1 Neurônios

Um neurônio *j*, recebendo uma entrada $P_j(t)$ no tempo *t* de um neurônio antecessor, compreende os seguintes passos (Zell, 1997):

- Ativação $A_j(t)$: estado do neurônio, dependente do tempo (discreto);
- Limiar θ_j (opcional): alterado pela função de aprendizagem;
- Função de Ativação *f*: calcula a nova ativação para t+1 baseado em $A_j(t)$, θj e $P_j(t)$, por meio da equação (1)

$$A_j(t+1) = f(A_j, \theta_J, P_j\ ) \qquad (1)$$

- Função de Saída f_{out}: Calcula o valor de saída *s* a partir da ativação (equação 2)

$$s_j(t) = fout(A_j) \qquad (2)$$

Um neurônio de entrada não possui antecessores, recebendo diretamente os dados de entrada. Analogamente, o neurônio de saída não possui sucessores, e sua saída é a saída final da rede.

2.1.3.1.1.2 Conexões, pesos e viés

Por definição, para que uma rede seja formada, neurônios devem estar conectados entre si, e cada conexão é responsável por transferir a saída de um neurônio *i* à entrada do neurônio *j* (Zell, 1997). Nesta configuração, o neurônio *i* é denominado predecessor de *j*, e o neurônio *j*, por sua vez, é denominado sucessor do *i*.

Estas conexões são ponderadas por pesos W_{ij}. Quando necessário, um viés pode ser introduzido e somado ao peso para que a entrada possa então atingir (ou deixar de atingir) o limiar, alterando o resultado da função de ativação (Lukoševičius, 2012).

2.1.3.1.1.3 Função de Propagação

A função de propagação (Zell, 1997) calcula a entrada $P_j(t)$ para o neurônio *j* a partir da saída $s_i(t)$ de seu(s) neurônio(s) predecessor(es) (equação 3),

$$P_j(t) = \sum_i s_i * W_{ij} + W_{0j} \quad (3)$$

Onde W_{0j} é o viés (quando aplicável).

2.1.3.1.1.4 Aprendizagem

A regra de aprendizagem é a função (ou algoritmo) utilizada para modificar as conexões, pesos e viés (Haykin, 1998). O resultado desta modificação é comparado com a função/comportamento que está sendo modelado para que se ajustem os parâmetros na próxima iteração. Há três principais formas de se modelar a regra de aprendizagem:

- Aprendizagem não-supervisionada: os dados de teste não foram classificados, categorizados ou marcados;

- Aprendizagem supervisionada: os dados de teste são classificados, e cada entrada é relacionada com um valor de saída desejado (sinal supervisório);
- Aprendizagem por reforço: os dados de teste são classificados, porém não são relacionados com uma saída desejada. Ao invés disso, uma função objetivo global é apresentada e os parâmetros são atualizados em prol deste objetivo.

2.1.3.2 Redes Neurais Recorrentes

Uma Rede Neural Recorrente (*Recurrent Neural Network* - RNN), analogamente ao cérebro humano, é uma rede neural cujos neurônios enviam sinais de resposta uns aos outros, formando assim um grafo (Grossberg, 2018). Esta comunicação interna dos neurônios faz com que a RNN exiba um comportamento dinâmico no quesito temporal, fazendo com que possua uma "memória" dos dados de que já recebeu. Com isto, diferentemente das redes *feed-forward* (onde o fluxo de informação dos neurônios ocorre apenas em um sentido), as RNN consigam processar múltiplos dados de entrada simultaneamente, o que as faz apropriadas para tarefas como reconhecimento de fala (Sak *et al.*, 2014) e escrita (Graves, Liwicki, Fernandez, Bertolami, & al., 2008). Na figura 4, é ilustrada uma RNN.

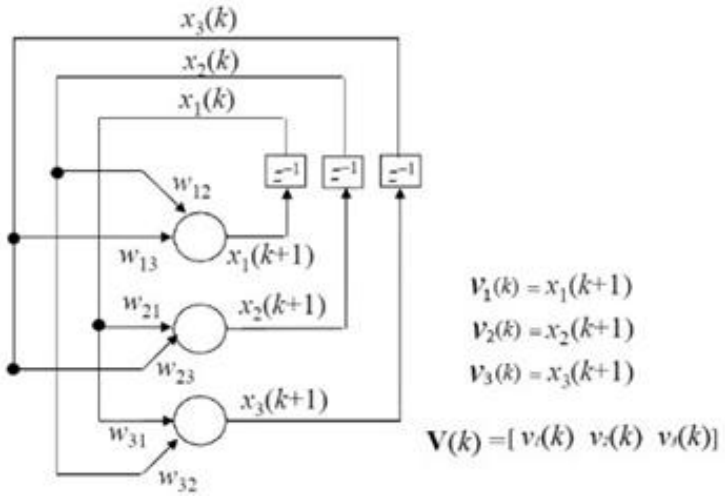


Figura 4 RNN de Hopfield (Martins, 2015)

2.1.3.3 Computação de Reservatório

As RNN, apesar de serem muito eficientes e, portanto, a referência em modelagem de sistemas, são relativamente difíceis de serem projetadas e treinadas. É necessário algum treinamento e estudo para entender seu funcionamento, algoritmos e a matemática envolvida. Além disso, o tempo gasto para treiná-las é elevado, fazendo com que seja necessário um bom entendimento da área na qual se realiza o estudo. Há também o que é chamado de gradiente de enfraquecimento, que se dá quando há a perda gradual de informação (por distorção) conforme o número de passos necessários para atingir a saída aumenta (Schrauwen *et al.*, 2007).

O conceito de Computação de Reservatório (CR) surgiu para solucionar este problema das RNN (Maass *et al.*, 2002). Começou-se (Buonomano & Merzenich, 1995) introduzindo o conceito de redes aleatórias de neurônios com elasticidades de curto prazo (chamadas sinapses dinâmicas), e que poderiam ser suportadas por uma topologia recorrente. Posteriormente, Maass (Maass *et. al*, 2002) propôs os conceitos da CR, que são a utilização de RNNs fixas, grandes e aleatórias (chamadas de reservatórios, que não recebem supervisão e não são alteradas durante o treinamento); e o treinamento apenas da interface do reservatório com as unidades de saída.

A implementação destas redes, para a obtenção deste efeito, não foi feita numa máquina de *Turing*, uma vez que esta requer transições sequenciais entre estados internos discretos, mas sim num modelo computacional chamado de Máquinas de Estados Sólidos (*Liquid State Machine* - LSM). As LSM, ao mesmo tempo em que não exigem este tipo de transição e permite processamento em tempo real de entradas variantes no tempo, continuam permitindo um poder computacional universal, sob condições ideais.

Por serem do tipo caixa-preta (*blackbox*), uma vez que a RNN começa a trabalhar para representar um problema, ela a faz autonomamente, sem interferência e/ou controle externo, como se assumisse vida própria. Isto faz com que as dificuldades e erros apresentados pela RNN (principalmente ao trabalhar com entradas com altos índices de variação e com sistemas de grandes dimensões) sejam de difícil identificação e resolução. Hertz (Hertz, Krogh & Palmer, 1991) propôs que fossem utilizados subgrupos de estados selecionados, chamados de atratores. Porém, isto elevou muito o poder computacional exigido, pois eram necessários muitos atratores para guardar pequenos pedaços de informação, fazendo assim com que esta técnica não permitisse trabalhos em tempo real.

Também foi teorizada (Maass *et al.*, 2002) uma abordagem na qual seriam gravadas as trajetórias dos estados internos, as quais seriam comparadas com as entradas e com as projeções de saída. Com isto, abre-se a possibilidade de treinar as leituras das saídas para extrair informações que anteriormente não eram interpretáveis das trajetórias gravadas.

2.1.3.4 *Echo State Networks*

Baseado nestes conceitos, H. Jaeger (Jaeger, 2001) publicou em 2001 (corrigida em 2010) sua abordagem que chamou de *Echo State Network* (ESN) – figura 5, e em 2012 foi classificada como sendo parte integrante da área de CR (Lukoševičius, 2012). Em seu artigo, Jaeger sugeriu reservatórios grandes (comparados os que são normalmente utilizados nas RNN) com poucos neurônios, e estes interligados por uma função de transferência sigmoidal. Estas conexões são assinaladas aleatoriamente e apenas uma vez (ou seja, não são treinadas ao longo do processo). Também são randômicos os pesos atribuídos às conexões entre os neurônios de entrada e o reservatório. Deste modo, os únicos pesos que são treinados são pesos de saída, que conectam o reservatório com os neurônios de saída.

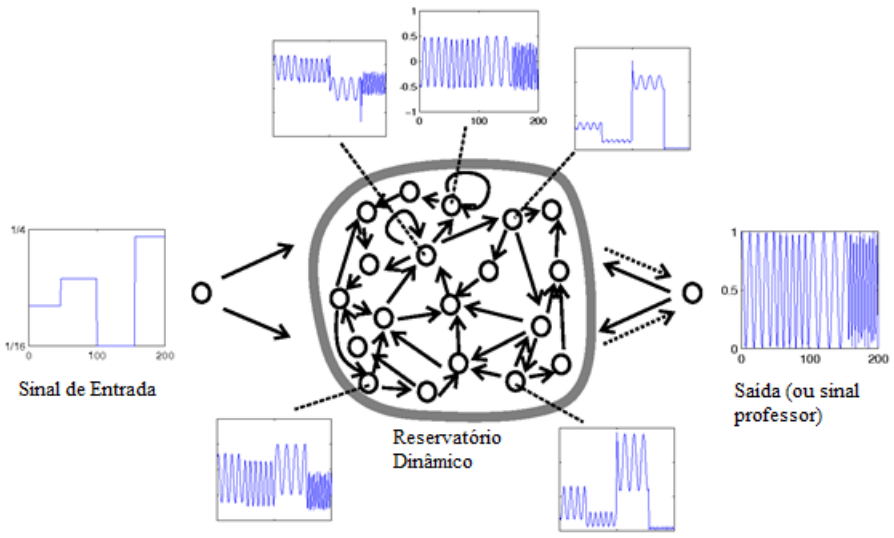


Figura 5 Representação Gráfica do Treinamento de uma ESN (http://www.scholar-pedia.org/article/Echo_state_network, acessado em 20/02/2019)

Durante o treinamento, as entradas serão inseridas no reservatório e uma saída "professora" é aplicada às unidades de saída. Os estados do reservatório ao longo do tempo são capturados e guardados. Uma vez que todos os inputs de treinamento forem

inseridos, usa-se uma regressão linear simples para relacionar os estados do reservatório com os outputs desejados.

A ideia é que, uma vez a rede treinada, ao serem inseridas entradas similares às que foram usadas no treinamento, a rede "ecoe" de volta os resultados obtidos anteriormente, fornecendo assim uma aproximação da evolução das entradas. A literatura mostra que as ESN são bastante eficientes e precisas em modelagem e previsão (Chouikhi *et al.*, 2017; Liu, Wang, & Wang, 2015; Prater, 2017; Jiang, Berry, & Schoenauer, 2008), e fornece métodos de treinamento (Jaeger, 2002; Lukoševičius, 2012).

2.1.4 *Particle Swarm Optimization*

Esta dissertação aborda a formação de sistemas preditivos a partir da geração de modelos ESN de maneira automática utilizando técnicas de otimização por algoritmos evolucionários. Na proposta apresentada na introdução, optou-se pela implementação de um algoritmo discreto evolucionário populacional de inteligência artificial para otimização de funções contínuas não-lineares, mais especificamente de inteligência de enxame: a Otimização por Enxame de Partículas (*Particle Swarm Optimization* - PSO) (Kennedy & Eberhart, 1995).

O PSO é utilizado nos mais diversos tipos de problemas gerais, como previsão de probabilidade de falência (Shin & Lee, 2002), demanda de energia elétrica (Ghanbari *et al.*, 2013) e otimização de redes (Chouikhi, 2017), além de problemas na área de engenharia nuclear, como o problema da recarga de reatores nucleares (Augusto, Nicolau & Schirru, 2015), modelo de dispersão atmosférica em acidentes nucelares (Serrão, 2018) e estudo da estrutura de componentes do elemento combustível (Leite, 2017).

O princípio básico de funcionamento é: após a inicialização aleatória da população inicial (posição e velocidade de cada partícula) e sua avaliação através da função *fitness* (função-alvo a ser otimizada), a cada geração a posição da partícula é atualizada baseada em sua velocidade; e sua velocidade é atualizada para a próxima geração seguindo uma fórmula que leva em conta a distância da partícula para a melhor partícula global e para a melhor partícula desta geração. Seu modelo esquemático de funcionamento se encontra na figura 6. Uma de suas vantagens é que pode ser implementado em poucas linhas de código (um pseudo-código será apresentado na

seção 2.3.4), com conceitos básicos de matemática e exige muito pouco computacionalmente.

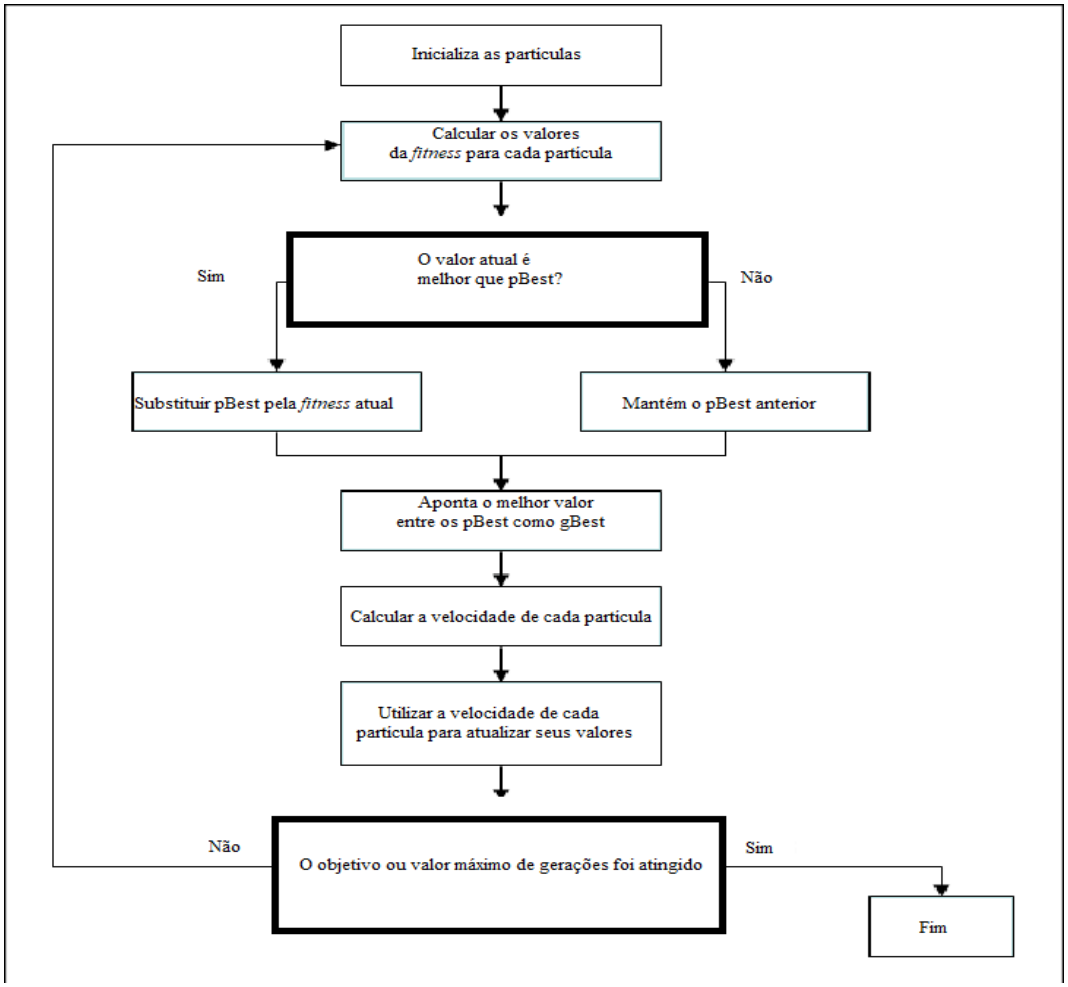


Figura 6 Estrutura do PSO (Oludare et al., 2014)

2.2 Arquitetura e Descrição Formal da *Echo State Network*

Como descrito anteriormente, uma ESN é uma RNN cuja camada *hidden* (interna) é um Reservatório Dinâmico (RD) grande em quantidade de neurônios - podendo ser da ordem de milhares – e pouco conectados (Rueda, 2014). Estas interconexões, assim como os pesos, são assinaladas aleatoriamente apenas uma vez. As interconexões, porém, não são alteradas, enquanto os pesos são alterados conforme o treinamento.

Uma formulação para o conceito de "eco" pode ser dada baseando-se na ideia de que um estado X(n) em um determinado temo *n* pode ser descrito como função de todo

o histórico de dados de entrada U(n), U(n-1), … , U(0). Isto é, existe uma função E tal que X(n) = E(U(n), U(n-1), … , U(0)) (Jaeger, 2001), o que tornaria X(n) um "eco" de todos os estados anteriores.

Para uma ESN com K unidades de entrada, N unidades internas e L unidades de saída, as ativações das unidades de entrada a um passo de tempo (discreto) de *n* são $U(n) = (U_1(n),..., U_k(n))$; as unidades internas (que compõem a memória) são $X(n) = (X_1(n),...X_N(n))$; e as unidades de saída são $Y(n) = (Y_1(n),...,Y_L(n))$. A única restrição que se impõe para os dados de entrada é que os mesmos venham de um espaço U compacto, ou seja, $U(n) \in U^J$, $N \in J$. Além disto, tem-se um grupo compacto de estados admissíveis $A \subset R^N$, o que implica em, para uma rede inicializada em um tempo *n*, o estado inicial X(n) estará contido em A. É necessário também que A seja fechado sob a atualização da rede, ou seja, para cada $U(n) \in U$, $X(n) \in A$, implica em $Y(X,U) \in A$. Estas são denominadas condições padrão de compactação.

A matriz $W^{in} = (w_{ij}^{in}) \in R$ de dimensão N x K guarda os pesos das conexões de entrada. Analogamente, a matriz $W = (w_{ij}) \in R$ de dimensão N x N guarda os pesos das conexões internas; a matriz $W^{out} = (w_{ij}^{out}) \in R$ de dimensão L x (K+N+L) guarda os pesos das conexões com as unidades de saída ; e a matriz $W^{back} = (w_{ij}^{back}) \in R$ de dimensão N x L guarda os pesos das conexões que fazem o *feedback* entre as unidades de saída e as internas.

O maior valor absoluto $|\lambda_{max}|$ de um autovetor de W é chamado de raio espectral. Durante o processo de construção da rede, este valor é utilizado para impedir que a rede apresente um comportamento caótico, perdendo assim sua característica de "eco". Para isto, divide-se o valor dos pesos pelo raio espectral, a fim de escaloná-los ou normalizá-los, dependendo do caso. Embora conste frequentemente na literatura (Jaeger, 2001; Jaeger, 2002; Martins, 2015; Venayagamoorthy & Shishir, 2009) que, para se garantir a propriedade de eco na ESN, o raio espectral tenha que ser um valor menor do que a unidade, não há restrições formais ou garantias que seja possível obter uma ESN com raios maiores do que 1 (Lukosevicius, 2012). Tal necessidade foi constatada não sendo necessária neste trabalho.

Neste ponto é importante ressaltar que não é necessária nenhuma outra restrição (seja ela de tipos de estrutura para as camadas ou proibições de conexões diretas

entrada-saída ou saída-saída), e todos os valores para dados de entrada, pesos e ativações utilizados neste trabalho são do conjunto dos números reais.

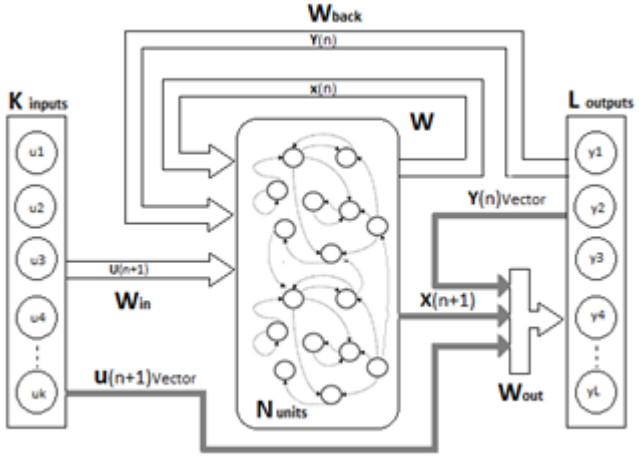


Figura 7 Esquemática de Funcionamento da ESN (Martins, 2015)

Na Figura 7 pode ser observada uma esquematização de uma ESN. As setas largas e as letras em caixa alta significam matrizes. As setas finas e as letras menores indicam vetores. A utilização da matriz W^{back} e do vetor U(n+1), ligado diretamente à camada de saída, também dependem da aplicação e devem ser avaliadas pelo usuário durante os testes para determinação da configuração mais adequada.

As funções de ativação estão descritas nas equações (4) e (5) abaixo,

$$X(n+1)= f\,(W^{in}\,u(n+1)+Wx(n)+W^{back}\,y(n)) \quad (4)$$

$$Y(n+1)=f_{out}\,(W^{out}\,(u(n+1),x(n+1),\,y(n)) \quad (5)$$

Onde. $f=(f_1,\ldots,f_n)$ são as funções internas de ativação, $f_{out}=(f_{out}1,\ldots,\,f_{out}n)$ são as funções de saída e $(u(n+1),x(n+1),\,y(n))$ é a concatenação dos vetores interno e de ativação da geração atual, e de saída da geração anterior.

2.2.1 Princípio de Funcionamento

De forma geral, uma abordagem inicial de RNN a um problema determinístico e discreto no tempo, utilizando-se de um sistema dinâmico, cuja função G aponta a próxima saída do sistema dado a entrada e o histórico das saídas, é (equação 6):

$$Y(n+1) = G(U(0),...U(n),U(n+1) ; Y(0),...Y(n),Y(n+1)). \quad (6)$$

Ao abordar o seguinte problema: “Construir uma RNN que retorne Y(n) que aproxima uma série temporal com entradas $U_{teach}(n)$, n0 ,..., n_{max} vindas de um grupo compacto U_{in} e saídas $Y_{teach}(n)$, n0 ,..., n_{max} professoras vindas de um grupo compacto U_{out}”, com uma ESN, segue-se os seguintes passos:

- Construir uma RNN que tenha a propriedade de eco em um grupo admissível A, em relação à entrada U_{teach} de U_{in} e um “pseudo-entrada” (*feedback*) Y_{teach} de U_{out}. Assim, obtém-se a matriz $W^{in\&back}$ N x (K+L) como a nova matriz de pesos; $\hat{U}_{teach}(n+1) = (U_{teach}(n+1),Y_{teach}(n))$ como o *feedback*.
- Inserir unidades de entrada à rede (com seus respectivos pesos de conexão). Neste passo, uma vez que a condição anterior foi atingida, há uma liberdade grande para escolher W^{in} e W^{out}.
- Rodar a rede com a entrada-professor e com a saída-professora, descartando-se o transiente inicial, escolhendo um estado inicial arbitrário X(0) e atualizando a rede utilizando os dados de entrada de n=0 até $n=n_{max}$, de acordo com a equação (7).

  $$X(n+1) = f(W^{in}*(U_{teach}(n+1)) + W^{back}*Y_{teach}(n) + W*x(n)) \quad (7)$$

  Escolhendo um transiente n_{min} para o qual, após passado o mesmo, o estado interno da rede seja determinado pelas entradas subsequentes, e com $Y_{teach}(n) = (Y_{Teach\ 1}(n),..., Y_{teach\ L}(n))$
- Determinar os pesos de saída que minimizem o erro de treinamento (equação 8),

$$E_j = \frac{1}{(n_{max} - n_{\min})} * \left(\sum_{n_{min}}^{n_{max}} G'_j - \sum_{1}^{n_{max}} (W_{i,j}^{out} * X_i(n)^2 \right) \quad (8)$$

Sendo $G'_j = (f_j^{out})^{-1} * Y_{teach\ j}(n)$

Nesta etapa pode ser utilizado qualquer algoritmo de regressão linear.

Uma vez que o erro foi minimizado ao objetivo, a rede está pronta para aproximar novas entradas.

2.2.2 Neurônios *Leaky Integrator*

Embora o reservatório como um todo possua uma memória das entradas, cada unidade neste modelo-padrão sigmóide, não. Seus valores num determinado instante de tempo *n* dependem apenas em parte e indiretamente dos valores anteriores, fazendo que possua um pouco de dificuldades, por exemplo, ao tentar aproximar dinâmicas que variam pouco de um instante de tempo para o outro e com alterações frequentes. Para estes casos, o ideal seria uma rede contínua.

Como a ESN é discreta no tempo, propõe-se uma formulação híbrida (Jaeger, 2001), na qual se usa os *leaky integrators* (figura 8). A equação que descreve a evolução de uma rede com *leaky integrator* é a equação 9:

$$\dot{X} = C\,(-aX + f(\,W^{in}*U + W*X + W^{back}*Y)) \qquad (9)$$

Onde C é a constante de tempo e *a* é a taxa de decaimento.

Ao se resolver esta diferencial com um passo δ, temos a equação (10):

$$X(n+1) = (1- \delta C a)\,X(n) + \delta C\,(f\,(W^{in}*U(n+1) + W*X(n) + W^{back}*Y(n))) \qquad (10)$$

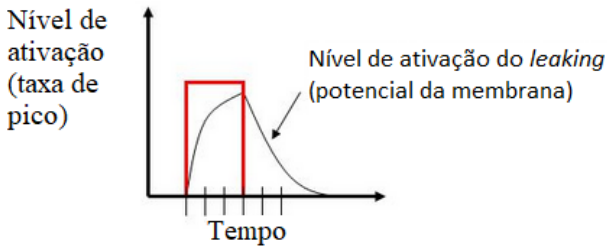


Figura 8 Ilustração da resposta do sistema com leaky integrator (em vermelho), e sem (em preto) (Stanley, 2006)

2.2.3 Treinamento

Como foi visto, uma das grandes vantagens da ESN. Além de ser mais simples e rápido do que as RNN, não há o escalonamento de esforço computacional com o aumento do número de neurônios e camadas. Além disto, o treinamento de uma ESN é tanto versátil (pois um mesmo modelo de rede pode ser treinado para várias tarefas diferentes) quanto precisa (Jaeger, 2001). Esta combinação faz possível, pois ao considerar a equação (11),

$$G' \sim \sum W_i e_i \qquad (11)$$

Onde $e_i=e_i$ (U(0),...,U(n); Y(0),...,Y(n-1)), percebe-se que uma parte integrante da função de atualização é o próprio sinal desejado. Além disto, como a rede possui a característica de "eco" dos estados passados, algumas propriedades do conjunto de dados de entrada e/ou de saída serão compartilhadas com a rede, porém sem que a rede perca a habilidade de introduzir variações. Por exemplo, ao treinar dados de entrada em formato de seno, e_i testará várias formas de variação, porém com o mesmo período. Com isto, a saída desejada será moldada desde o começo com a "forma" certa.

Foi mencionado anteriormente que, durante a explicação da construção da ESN, a fórmula comumente aplicada para o treinamento, assim como seu objetivo (encontrar os pesos da camada de saída que melhor aproximam a rede da fórmula real). Nesta seção, será descrito um passo a passo mais específico sobre o treinamento.

Cabe destacar que, além do método de treinamento baseado em minimizar o erro entre a saída desejada e a estimada por mínimos quadrados (nomeado de *off-line* em (Rueda, 2014)), existe também o denominado *online*. O treinamento *online* é feito iterativamente, com pares entrada-saída, no qual os pesos são atualizados após cada par atravessar a rede, visando diminuir o erro entre o estimado e o valor real. Este método conta com a ferramenta de *backpropagation* (ou alguma outra com o mesmo princípio) para tal ajuste dos pesos de saída. Porém, para a modelagem de sistemas dinâmicos, este é mais complexo que o *online*. O processo de treinamento a ser utilizado (*online*) se divide em duas partes: amostragem e cálculo de pesos (Jaeger, 2002).

Na fase de amostragem, acontece o chamado *teacher forcing*, que se configura ao se escrever o sinal professor diretamente nas unidades de saída para os *n* primeiros intervalos de tempo. Então, a rede é inicializada no tempo *n*=1 e em um estado arbitrário (pode-se inclusive utilizar o estado 0). Através das conexões de *backpropagation* W^{back}, o sinal professor d(n) é jogado no RD e excita a dinâmica de ativações dentro da mesma. Ao final deste processo, os padrões de ativação dentro do RD serão os mesmos do sinal professor d(n); e os padrões de ativação dentro do RD são diferentes entre si.

Ainda durante esta fase, os estados internos X(n) = (X_1(n),...,X_i(n)) para os tempos n=n_{min},...,n_{max} são coletados e alocados nas linhas da matriz de estados M (n_{max}-

n_{min}) X *i*. Concomitantemente, as saídas professoras d(n) são coletadas nas linhas de uma matriz T (n_{max}-n_{min}) X 1. Lembrando que, como citado, n_{min} é o número de estados descartados afim de se evitar um transiente.

Na fase de cálculo de pesos, os pesos de saída W_j^{out} para a unidade de saída linear Y(n), de forma que Y(n) e d(n) possam ser aproximadas como uma combinação linear da série temporal de ativação interna $X_i(n)$ da seguinte forma (equação 12):

$$d(n) \sim Y(n) = \sum_{i=1}^{i}(W_i^{out} * X_i(n)) \qquad (12)$$

Na implementação algorítmica da equação (12), tem-se que esta operação pode ser resumida calculando a pseudoinversa de M e multiplicando-a por T (equação 13), que é de fácil implementação computacional.

$$W^{out} = M^{-1}T \qquad (13)$$

Finalmente, integra-se os valores recém-calculados dos pesos de saída à rede, e a mesma está pronta para uso.

2.2.3.1 Parâmetros que Afetam o Desempenho da ESN

Ao se treinar uma ESN há, basicamente, 11 parâmetros a serem trabalhados durante o treinamento de uma ESN:

- Raio espectral;
- Tamanho do reservatório;
- Número de entradas;
- Número de saídas;
- Esparsidade;
- *Input shift*;
- *Input scaling;*
- *Teacher shift* (caso haja *teacher forcing*);
- *Teacher scaling* (caso haja *teacher forcing*);
- Ruído*;*
- Função de ativação.

O número de entradas e saídas depende do problema sendo tratado. No caso deste trabalho, como foi escolhido trabalhar com 1 variável de 1 acidente por vez, resulta em apenas 1 entrada e 1 saída, porém uma ESN pode ser utilizada para casos múltiplas entradas/múltiplas saídas (MI/MO) com a mesma eficiência (Martins, 2015). A função de ativação será aplicada à saída do neurônio. Sua forma mais comumente utilizada é função sigmóide tangente hiperbólica (*tanh*) (Jaeger, 2002), mas também pode aparecer como uma função identidade para redes lineares.

O *input shift* e o *teacher shift* (caso possua) é um escalar (ou vetor de escalares de dimensão do número de entradas, se for maior que 1), quer será adicionado a todos os valores de entrada (e de saída, caso haja *teacher forcing*), e sua variação causa uma variação no comportamento da ESN, fazendo com que o processo de treinamento seja ou mais ou menos rápido/preciso. O mesmo efeito ocorre com o *input scaling* e o *teacher scaling* (caso possua), que é um escalar (ou vetor de escalares de dimensão do número de entradas, se for maior que 1), que será utilizado para multiplicar todos os valores de entrada (e de saída, caso haja *teacher forcing*), e determina o quão não-linear serão as respostas do RD. Em geral, recomenda-se testar se a tarefa necessita do *teacher forcing*, pois há casos em que sua utilização atrapalhe o desempenho da rede (Lukosevicius, 2012). Para ambos os valores, não há uma regra geral para determiná-los, pois o valor ideal varia bastante para cada tarefa modelada.

A esparsidade (*sparsity*) é a porcentagem de conexões entre os neurônios do RD que serão zeradas. Ao propor as ESN, (Jaeger, 2001) escreveu que um reservatório "rico" é um reservatório cujas conexões esparsas (em alguns casos, chegando a 98,75% de conexões zeradas) e aleatoriamente conectadas, e que é essa esparsidade que permite que o reservatório se "desacople" em sub-redes, desenvolvendo assim as dinâmicas individuais. Este parâmetro, porém, não afeta muito significativamente o desempenho da ESN, embora maiores valores de conexões nulas geralmente forneçam erros menores (Lukosevicius, 2012).

O ruído, por sua vez, é o valor que determinará o intervalo do qual será escolhido um número aleatoriamente para ser adicionado (ou subtraído) a cada dado de entrada. Ou seja, um ruído = 0,001 significa que o intervalo de escolha do número a ser somado é [-0,001 ; 0,001]. Ela se faz necessária para que se garanta generalidade à rede, principalmente quando se tem dados que fiquem vários períodos de tempo sem variar ou dados que sigam fórmulas físicas.

Finalmente, chega-se nos dois principais parâmetros (juntamente com a taxa de *leaking*, que não é aplicável a este estudo) a se trabalhar na construção da ESN (Lukosevicius, 2012; Venayagamoorthy & Shishir, 2009; Jiang *et al.*, 2008; Chouikhi *et al.*, 2017): o raio espectral e o tamanho do reservatório.

Como mencionado, o maior valor absoluto $|\lambda_{max}|$ de um autovetor de W é chamado de raio espectral, e é responsável por escalonar a matriz W ou, em outras palavras, aumentar a largura da distribuição de seus elementos não nulos. Por via de regra, ele deve ser maior quanto mais longa for a memória da entrada necessária para o problema (Jaeger, 2012). Embora seja muito frequente na literatura a restrição para que o raio não seja >=1, isto não é matematicamente comprovado, e um argumento intuitivo que pode ser dado contra esta restrição é que o "esmagamento" das ativações das não-linearidades pela *tanh*, entradas fortes o suficiente consigam "espremer" atividades autônomas das ativações do RD. Além disto, um raio espectral <1 também não garante matematicamente para todas as entradas que a propriedade de eco será mantida. Outra propriedade descrita na literatura é que, de forma geral, raios espectrais maiores levam a redes melhores, ou seja, com menor erro em relação às entradas originais (Venayagamoorthy & Shishir, 2009).

O número de neurônios (tamanho do reservatório), apesar de na teorização da ESN ser tratado como preferencialmente grande, tem que ser tratado com bastante cuidado (Jaeger, 2010). Isto porque um número exageradamente grande de neurônios, embora possa levar a um erro baixo, causa um problema chamado de *overfitting*, que se caracteriza pela a perda de generalidade da ESN na busca por um erro menor, além de comprometer sua utilidade para sistemas cujas entradas variem. Como regra geral para selecionar um número de neurônios que minimize o erro e não cause *overfitting*, Jaeger sugeriu que este seja entre R/2 e R/10, sendo R o número de dados de entrada; ou então ser pelo menos igual ao número (estimado) de valores independentes que o RD tem que se lembrar (Lukosevicius, 2012).

Apesar de haver uma impressão que um reservatório maior leve necessariamente a um erro menor, há exemplos na literatura de que não há uma relação direta envolvendo o tamanho do reservatório e o erro (Chouikhi *et al.*, 2017; Prater, 2017; Jiang, 2008).

2.2.3.2 Métodos de Avaliação do Erro

Há diversas formas de se calcular o tamanho da diferença entre os valores que se espera de uma rede, e os valores que ela retorna. Nesta seção serão abordados os mais comumente utilizados.

2.2.3.2.1 Variância e Desvio Padrão

A Variância (Var), e sua raiz quadrada, o Desvio Padrão (DP), são medidas de dispersão estatística, que indicam o quão afastado da média os valores estão (Bussab & Morettin, 2017). Sua fórmula está indicada na equação 14:

$$\operatorname{Var}(X) = \mathrm{E}\left[(X - \mu)^2\right]. \qquad (14)$$

Onde E está descrito na equação (15) abaixo, X é a distribuição e μ é a média

$$E[X] = \sum_{i=1}^{k} x_i p_i \qquad (15)$$

O DP é uma ferramenta muito útil para se ter uma ideia, principalmente graficamente, de como estão se comportando os dados, pois quando se tem um DP pequeno, os valores estão concentrados perto da média. Outra propriedade útil é que o intervalo [MI-DP ; MI+DP] compreende 64,2% dos dados, enquanto o intervalo [MI-2*DP ; MI+2*DP] compreende 91,4% dos dados.

2.2.3.2.2 Erro Médio Quadrático

Embora a variância seja um bom indicador de espalhamento, ela ainda é proporcional à aos valores de entrada, fazendo com que a comparação entre variâncias de conjuntos de dados de grandezas diferentes não seja direta. Para normalizar-se a variância, introduziu-se o conceito de Erro Médio Quadrático (MSE - *Mean Squared Error*), apresentado na equação (16):

$$\mathrm{MSE} = 1/n * \mathrm{Var} \qquad (16)$$

Onde *n* é o número de dados e Var é a variância (equação 14).

O MSE foi o quesito utilizado por Jaeger (Jaeger, 2001) para analisar a eficiência das redes ESN.

2.2.3.2.3 Raiz do Erro Médio Quadrático

O MSE, por ter elevado as diferenças ao quadrado, não possui a mesma unidade dos valores de entrada, não sendo possível colocá-los no mesmo gráfico das variáveis de entrada, por exemplo. Para isto, retira-se a raiz quadrada do MSE, obtendo o RMSE (*Root Of The Mean Squared Error* – Raiz do Erro Médio Quadrático) (equação 17):

$$\text{RMSE} = \sqrt{MSE} \qquad (17)$$

O RMSE também é dito como sendo uma medida da precisão e, portanto, bastante utilizada para medir diferenças entre previsões de diferentes modelos feitas com os mesmos dados, que é o caso deste estudo. Além disto, como cada erro é proporcional a seu valor, erros grandes, mesmo que pouco frequentes, são bastante penalizados, fazendo com que eles sejam facilmente detectados. Como este estudo prioriza erros menores ponto a ponto, este método de avaliação é o mais indicado e foi o escolhido.

2.2.3.2.4 Erro Médio Absoluto

O Erro Médio Absoluto (EMA) é a medida da diferença entre duas variáveis contínuas. No caso de vetores, o EMA deles é a distância vertical média entre cada ponto e a linha identidade (equação 18):

$$MAE = {\sum_{i=1}^{n} |y_i - x_i|}/{n} \qquad (18)$$

Onde *y* e *x* são os vetores a serem comparados, *i* são seus elementos e *n* o número total de elementos.

Esta avaliação fornece informações que o RMSE não tem, como direção e magnitude dos erros, além de uma interpretação mais clara e direta do erro (Willmott & Matsuura, 2005). Porém, tem como desvantagem o fato de usar valores absolutos, o quê pode não ser desejado, dependendo da formulação matemática do problema.

2.2.4 Aplicações

As ESN têm sido aplicadas com sucesso em previsão de séries temporais (Rueda, 2014; Chouikhi *et al*., 2017), identificação de sistemas e identificação inversa de sistemas (Cao, Xu & Hu, 2015), telecomunicações (Bauduin *et al.*, 2015), previsões climáticas (Liu *et al*., 2015), identificação de sistemas (Martins, 2015), entre muitas outras.

Este trabalho concentra-se na utilização das ESN em formulação de sistemas preditivos.

2.3 Arquitetura e Descrição Formal do PSO

Para a implementação da mecânica que simule um cardume, Kennedy e Russell propuseram uma população inicial de tamanho N, dimensão *d* (representando o número de variáveis a ser otimizada), sendo inicializada aleatoriamente com posições $X_n(1,...,d)$, e velocidades $V_n(1,..,d)$, e tendo suas *fitness* avaliadas. Como é a primeira geração, cada *fitness* individual será a melhor até então, e serão todas guardadas numa lista *pbest* ($pbest_1,...,pbest_N$), que recolhe a melhor fitness de cada indivíduo; e suas posições serão guardadas numa lista *Xpbest*($Xpbest_1,...,Xpbest_N$), que recolhe a posição na qual se obteve a melhor *fitness* de cada partícula. Guarda-se também o *gbest* (melhor *fitness* global) e o *Xbest* (posição da partícula que obteve a *gbest*). Usualmente inicializa-se a velocidade como sendo 1% da posição, ou seja, $V_n=X_n*0,01$.

Na dinâmica de movimentação do cardume, foram identificados 3 fatores que influenciam no movimento que cada indivíduo realiza (Kennedy & Eberhart, 1995):

- A inércia de movimento, *w*, comum a todo cardume, que caracteriza o quanto o vetor velocidade será alterado;
- O fator de contribuição individual, C1, que indica o quanto a partícula será influenciada pela sua melhor posição; e
- O fator de contribuição global, C2, que indica o quanto a partícula será influenciada pela melhor posição do grupo.

A cada geração, após avaliar as fitness de cada partícula e atualizar (se for o caso), *pbest, Xpbest, gbest* e *Xbest*, cada partícula tem sua velocidade atualizada em todas as dimensões, de acordo com a seguinte fórmula (equação 19):

$$V_{n,d}(j+1) = w*V_{n,d}(j) + \text{cognitivo} + \text{social} \quad (19)$$

$$\text{Cognitivo} = C1*r1*(pbest_{n,d}-X_{n,d}(j)) \qquad (20)$$

$$\text{Social} = C2*r2*(gbest_{n}-X_{n,d}(j)) \qquad (21)$$

Onde $V_n(j)$ é o vetor velocidade atual da partícula, $V_n(j+1)$ é o vetor velocidade da partícula na iteração j+1, *w* é a inércia, cognitivo é representado pela equação (20), o social pela equação (21), C1 e C2 são, respectivamente, o fator de contribuição individual e global, r1 e r2 são dois números aleatórios no intervalo [0,1], gerados novamente a cada iteração.

Finalmente, cada dimensão de cada partícula será atualizada seguindo a equação (22), que implementa o movimento linear uniforme que cada partícula fará em cada instante de tempo para seguir o fluxo do cardume e se aproximar do melhor indivíduo.

$$X_{n,d}(j+1) = X_{n,d}(j) + V_{n,d}(j+1) \qquad (22)$$

Onde $X_n(j+1)$ é o vetor posição da partícula *n* na iteração j+1, $X_n(j)$ é o vetor posição atual da partícula e $V_n(j+1)$ é o vetor velocidade atualizado da partícula n, conforme equação (19).

2.3.1 Princípio de Funcionamento

Conforme indicado, o PSO, se inicializa com partículas espalhadas aleatoriamente pelo espaço, avalia qual possui melhores características e direciona, de maneira indireta e com componentes de aleatoriedade, as partículas em direção à melhor, melhorando-as a cada interação.

Como a convergência do PSO tende a ser rápida e seu fator de espalhamento é limitado, o PSO pode acabar ficando preso em mínimos locais. Portanto, muitas vezes, é mais produtivo realizar N tentativas com J interações, ao invés de uma tentativa com N*J interações. A figura 9 ilustra como é o algoritmo de funcionamento do PSO.

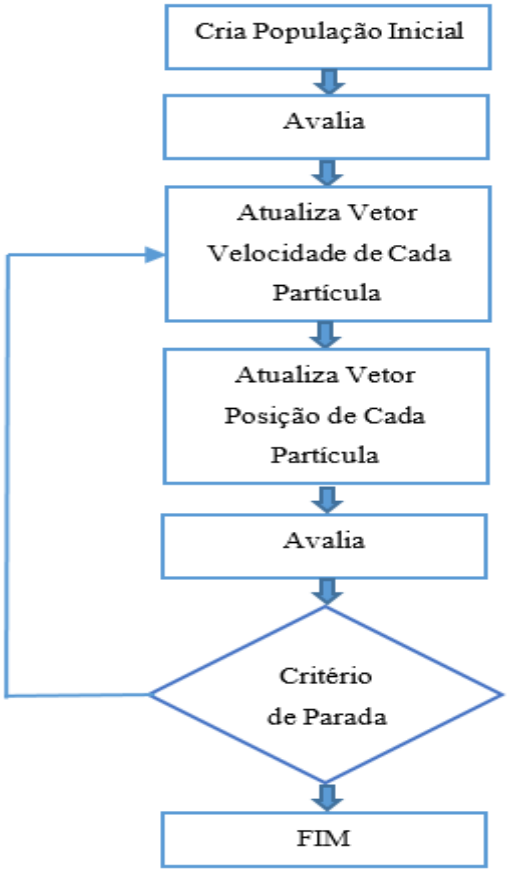


Figura 9 Fluxograma de Funcionamento do PSO (Meneses et al., 2007)

Como critério de parada, pode-se adotar como alvo um determinado valor de *fitness* que esteja se desejando otimizar ou um número máximo de interações. No PSO desenvolvido pelo autor para este estudo, foi utilizado como critério de parada um número máximo de iterações.

2.3.2 Parâmetros que Afetam o Desempenho

Outra vantagem do PSO é que são poucos parâmetros que afetam o seu desempenho e, por ser um conceito relativamente antigo e popular, há diversos estudos sobre a escolha dos mesmos (Carlisle & Dozier, 2001; Eberhart & Shi, 2000; Shi & Eberhart, 1998; Clerc & Kennedy, 2002; Trelea, 2003). São eles: *w*, C1, C2 e número de indivíduos.

A escolha do número de indivíduos se dá mais por uma questão de esforço computacional e uma relação custo-benefício que se faz entre aumento de tempo causado pelo maior número de partícula *versus* o ganho que se tem, pois em relação ao

seu funcionamento, o PSO é pouco sensível a este número (Kennedy & Eberhart, 1995). Este ajuste é relativamente simples de se fazer, e pode ser feito ao longo da utilização, conforme os dados forem sendo obtidos. Pode-se, por exemplo, começar com um valor relativamente grande (500 ou mais), rodar com poucas interações ($j<50$), e ir diminuindo em passos de 50 ou 10, para se encontrar um valor que apresente o melhor resultado. Porém, geralmente, valores pequenos (entre 30 e 50) oferecem o melhor custo-benefício (Carlisle & Dozier, 2010).

A mesma análise e observações podem ser feitas para o número de iterações: deve-se ser feita uma análise de custo-benefício entre o aumento do tempo causado pelo maior número de iteração versus o desempenho do PSO. Em geral, a convergência do PSO se dá para um mínimo local muito rapidamente, às vezes 20 iterações ou menos (Martins, 2015), fazendo com que geralmente seja mais eficiente rodar o PSO mais vezes com um número menor de iterações. Além disso, deve-se ter em mente que um número maior de partículas requer um número maior de iterações para atingir o mesmo nível de desempenho (Trelea, 2003).

Para a escolha do valor de inércia, valores menores que 0,8 fazem o PSO convergir muito rapidamente (Shi & Eberhart, 1998), porém são altas as chances de se ter chegado a um mínimo local. Caso a solução encontrada atenda a necessidade do problema (a *fitness* final for a desejada), pode-se utilizar valores baixos de w. Caso contrário, sugere-se realizar um estudo de convergência para valores medianos de w ($0,8<w<1,2$). Valores maiores que 1,2 tendem a demorar muito tempo para convergir, podendo não atender aos requisitos de desempenho.

Finalmente, há os fatores de contribuição C1 e C2. A “regra de ouro” para estes fatores é que ambos somem 4,1 (Carlisle & Dozier, 2010). Como a alteração na velocidade de convergência causada por estes fatores é muito fácil de ser percebida, o melhor jeito de ajustar estes parâmetros é realizar um estudo de convergência, com N e J pequenos (20 e 10, respectivamente) e variar ambos em passos de 0,1, fazendo C1 começar em 0 e ir até 4,1, enquanto C2 varia de 4,1 até 0.

Cabe também mencionar aqui o conceito de (Eberhart & Shi, 2000), o qual consiste em colocar restrições para os valores da posição e/ou velocidade. A introdução desta restrição pode melhorar significativamente a velocidade de convergência do PSO, uma vez que impede que os valores de dispersem muito, concentrando as buscas. Porém, uma restrição somente pode ser aplicada quando se conhece completamente o espaço de busca, para se ter certeza que a região de máximo global não está na região

restrita. Neste estudo foram utilizadas diferentes restrições em diferentes dimensões da partícula, como será explicado no capítulo 3.

2.3.4 Pseudo-Código

Abaixo, descreve-se o pseudo-código para implementação do PSO, no qual foi baseado para a implementação do código de autoria própria utilizado neste estudo, com N partículas de D dimensões cada, K iterações máximas e função *fitness* = f(X).

----------*inicialização da população*----------

```
Para k de 1 a K iterações:
        Para j de 1 a N partículas:
                Para i de 1 a D dimensões:
                        x[i][j]=random(0,1)
```

```
Guardar todas em pbest e xpbest;
Encontrar a melhor partícula e guardá-la em gbest e xbest;
```

----------*atualização:*----------

```
Para k de 1 a K iterações:
        Para j de 1 a N partículas:
                Para n de 1 a D dimensões:
                        x[i][j]+=v[i][j]
                        r1=random(0,1)
                        r2=random(0,1)
                        Social = c1 * r1 * (xbest[i] - x[j][i])
                        Cognitivo = c2 * r2 * (pbest[j][i] - x[j][i])
                        v[j][i] = (w * v[j][i]) + Social + Cognitivo
```

----------*avaliação das novas fitness*----------

```
        Para j de 1 a N partículas:
                fit= f( X[j])
                If fit < gbest:
                        Gbest = fit
                        Xbest = x[j]
                Elif fit<xpbest[j]:
                        pbest[j]=fit
                        xbest[j]=x[j]
```

3 - Metodologia

3.1 Introdução

Como ressaltado na introdução, ainda não se tem implementada comercialmente esse tipo de rede, e os estudos que se tem sobre a utilização de RNN como sistemas preditivos está concentrado basicamente em modelos off-line (que realizam uma previsão estática com dados existentes, e que posteriormente o modelo encontrado é utilizado para alguma previsão), enquanto o modelo desejado é um modelo online, que seja capaz de receber dados em tempo real e, juntamente com dados históricos, atualizar a previsão para dizer qual será o comportamento do sistema em períodos de tempo futuros.

Portanto, antes de se estabelecer este sistema online, há primeiro que se provar sua possibilidade. Para isto, o trabalho com a ESN foi dividido em 3 partes: primeiro, o sistema tem que ser capaz de acompanhar o funcionamento normal do reator, para formular um modelo. Depois, ele deve ser capaz de identificar em qual estado o reator se encontra, isto é, dizer se está em operação normal ou em acidente (e qual acidente está acontecendo), recebendo como entrada os dados em tempo real. Finalmente, após ter confirmado os dois passos anteriores, será montada uma rede que consiga prever se o comportamento atual dos dados é o que já se observou como um comportamento pré-acidente.

3.2 Dados de Treinamento

Durante todo o experimento, foi utilizado um mesmo conjunto de dados que consiste em 60 dados (coletados um a cada segundo, formando um monitoramento ao longo de 60 segundos) de cada uma das 18 variáveis medidas pelo Sistema de Monitoração de Processos de Angra (SICA), para cada um dos 16 acidentes simulados, além de 60 segundos de operação normal do reator de ANGRA 1. Estes dados são dados obtidos do simulador da usina, que fornece o comportamento de variáveis idêntico ao comportamento real da usina. Na tabela 1 estão discriminados as variáveis e os acidentes (Mól, 2002).

A escolha deste conjunto de dados se deu por ser um caso real, o que é essencial para fundamentar os resultados, uma vez que a rede já estará sendo treinada para seu uso final.

Como este conjunto de dados fixo e não há acesso a novos dados em tempo real, o mesmo grupo de dados que foi utilizado para treinamento, foi utilizado para teste. Isto não leva a perda de generalidade nem leva ao *overfitting*, pois há a introdução do ruído, além do controle no número de neurônios.

Outra consequência deste conjunto de dados é que não se tem as condições de operação que levaram ao acidente (apenas um dado do instante imediatamente anterior), além do fato de só haver 60 segundos. Isto não se configura um problema para o desenvolvimento da rede em si, porém algumas adaptações foram feitas, que serão descritas a seguir.

A primeira, devido ao alto custo computacional que seria necessário para treinar a rede com 18 entradas (variáveis) e 16 saídas (acidentes), e tendo em mente o objetivo primário que é obter uma rede que apenas acompanhe o funcionamento do reator, foi a de realizar esta primeira parte com 1 entrada e 1 saída, na qual a rede acompanharia uma única variável ao longo de 60 segundos de 1 acidente. Para se ter uma base comparativa e para fins estatísticos, foi utilizado este método para 3 acidentes diferentes (LOCA, SGTR e Blackout), além do funcionamento normal. A variável selecionada para acompanhamento foi a pressão do gerador de vapor.

A escolha destes acidentes se deu, pois, para testar e comprovar a eficiência das redes, foram elegidos os dois casos mais difíceis (SGTR e LOCA), que são dois acidentes que apresentam, para esta variável, curvas muito semelhantes (o que torna bastante difícil para sistemas computacionais sua diferenciação); enquanto o Blackout é o acidente mais comum.

A escolha da variável se deu devido ao comportamento da mesma em acidentes ser de caráter fortemente não-linear (como pode ser observado na figura 10), além de esta ser uma das mais importantes para a segurança da planta em caso de acidentes (Martins, 2015).

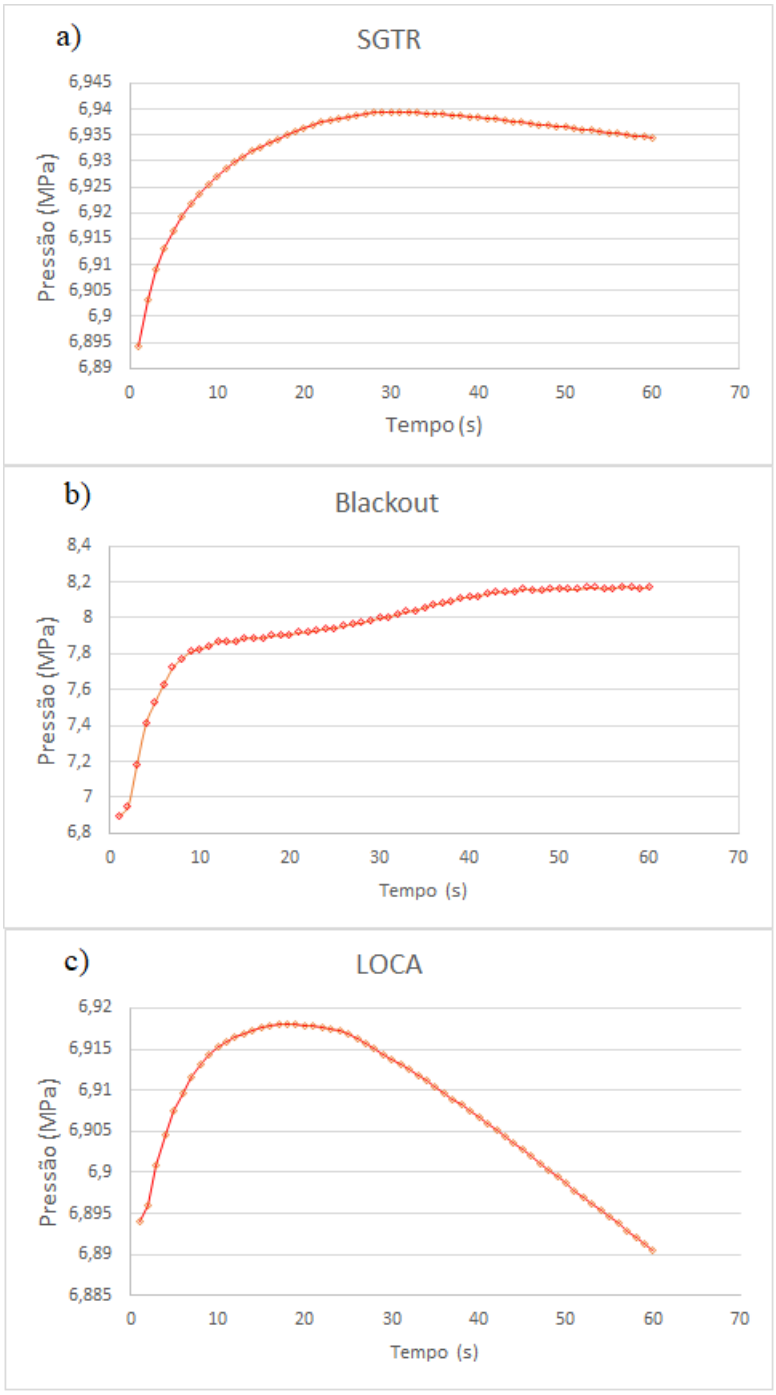


Figura 10 Gráfico da Pressão nos Diferentes Acidentes: a) SGTR, b) Blackout e c) LOCA

3.2.1 Separação dos Dados de Treinamento

Como uma rede precisa de um grupo de dados para seu treinamento e outro grupo para testar, inicialmente foi feita a divisão de 30 segundos iniciais para treinamento e os 30 finais para teste (30/30) e, posteriormente, a divisão 50/10, conforme os resultados indicavam a necessidade de mais dados de treinamento.

Rapidamente, porém, estas ideias se mostraram pouco eficazes, uma vez que apenas 30 ou 50 dados de treinamento são quantidades muito pequenas para que a rede possa reconhecer o comportamento dos dados e os reproduzir de forma satisfatória. Além disto, pela característica da ESN de "ecoar" o comportamento dos dados treinados, é necessário que os dados de entrada cubram todo o espectro a ser reproduzido. Com isto, a solução encontrada foi utilizar todos os 60 segundos como treinamento, e os mesmos 60 para teste da rede.

Houve uma melhora do RMSE ao trocar para este procedimento, porém os resultados ainda não estavam satisfatórios (conforme pode ser visto na figura 11). Vale ressaltar que, apesar de numericamente o erro ser pequeno, ainda não permite uma simulação confiável do comportamento dos dados.

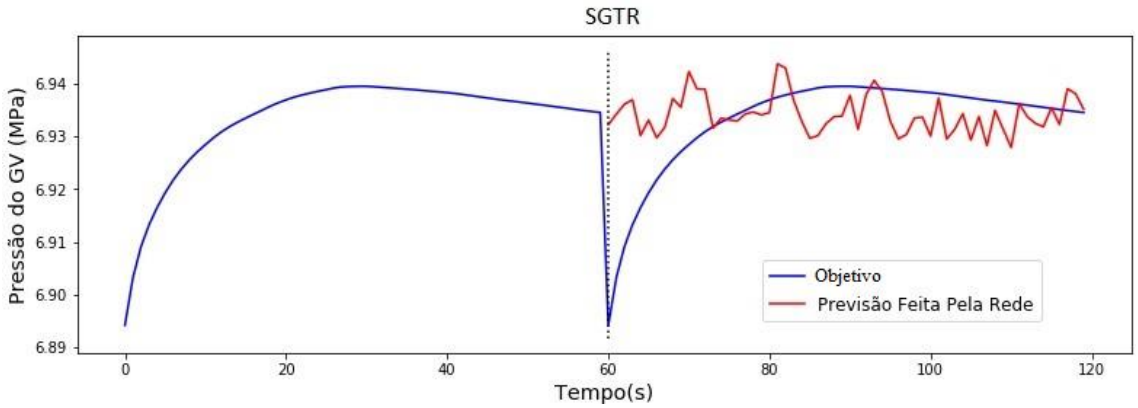


Figura 11 Gráfico da rede com apenas 60 dados de treinamento, para o acidente SGTR, utilizando reservatório=30, raio=0,95, input shift=1, input scaling=1, teacher scaling=1, teacher shift=1, sparsity=0, ruído=0,001. RMSE = 0,00998

Há uma ideia intuitiva de que, em geral, um maior grupo de dados está diretamente ligado a uma redução no RMSE da rede. Como só há 60 dados disponíveis, cogitou-se utilizar funções de regressão linear para encontrar uma função que aproxime o comportamento dos dados e, a partir dela, gerar diversos novos dados para melhorar o treinamento da ESN. Esta ideia, porém, foi descartada, pois além de haver um erro intrínseco no processo, que seria carregado ao longo de todo estudo, estes dados novos gerados não seriam dados reais de funcionamento.

Finalmente, a solução encontrada foi replicar os 60 dados de entrada cinco vezes, formando assim um conjunto de 300 dados para treinamento (vale lembrar que o tratamento dos parâmetros da ESN previne o *overfitting*, mesmo com a repetição dos dados de entrada). Na figura 12 pode ser observada a melhora na previsão da rede (em relação à figura 11), e apesar de a mesma ainda apresentar uma oscilação grande ponto a

ponto, o comportamento geral da curva começa a ser replicado (principalmente a primeira metade).

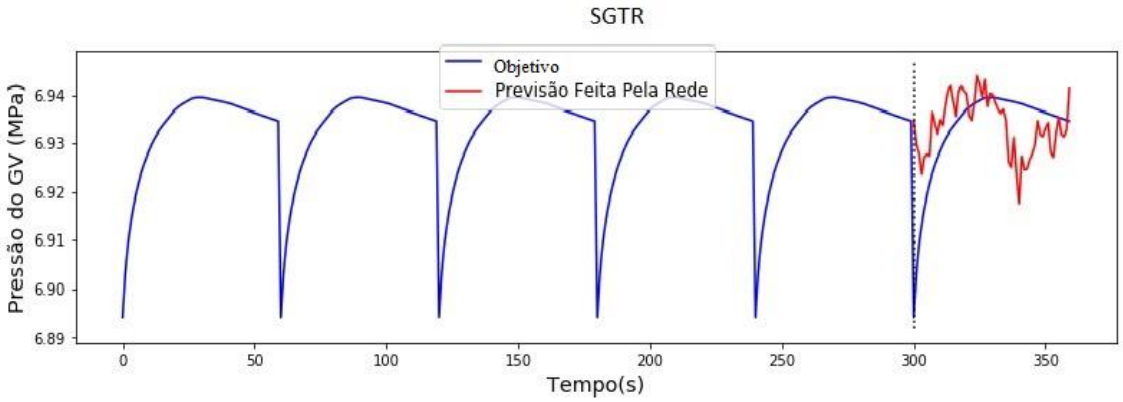


Figura 12 Gráfico da rede com 300 dados de treinamento, para o acidente SGTR, utilizando reservatório=120, raio=0,95, input shift=1, input scaling=1, teacher scaling=1, teacher shift=1, sparsity=0, ruído=0,001. RMSE = 0,00987

3.2.2 Dados do Funcionamento Normal

Ao analisar o gráfico dos dados de funcionamento normal do reator (figura 13, tabela 2), percebeu-se que o mesmo não apresentou o comportamento linear esperado (uma vez que durante a operação normal, o reator apresenta-se bastante estável, com poucas ou nenhuma alteração nas variáveis medidas pelos sistemas de controle), o que influencia no treinamento da rede, uma vez que a ESN tentaria imitar este comportamento. Os motivos para tal são desconhecidos, uma vez que não há como verificar como a geração dos dados foi feita.

Tabela 2 Dados originais do funcionamento normal

Pressão (Mpa)	Tempo (s)	Pressão (Mpa)	Tempo (s)	Pressão (Mpa)	Tempo (s)	Pressão (Mpa)	Tempo (s)
6,89416	1	6,889	16	6,88305	31	6,87811	46
6,89442	2	6,88858	17	6,88269	32	6,87782	47
6,89441	3	6,88817	18	6,88233	33	6,87754	48
6,89429	4	6,88775	19	6,88197	34	6,87726	49

6,894	5	6,88734	20	6,88162	35	6,87698	50
6,8936	6	6,88694	21	6,88128	36	6,87671	51
6,89313	7	6,88653	22	6,88094	37	6,87644	52
6,89264	8	6,88613	23	6,8806	38	6,87618	53
6,89215	9	6,88573	24	6,88027	39	6,87592	54
6,89166	10	6,88534	25	6,87995	40	6,87567	55
6,89119	11	6,88494	26	6,87963	41	6,87542	56
6,89073	12	6,88456	27	6,87932	42	6,87517	57
6,89029	13	6,88417	28	6,87901	43	6,87493	58
6,88985	14	6,88379	29	6,8787	44	6,87469	59
6,88942	15	6,88342	30	6,8784	45	6,87445	60

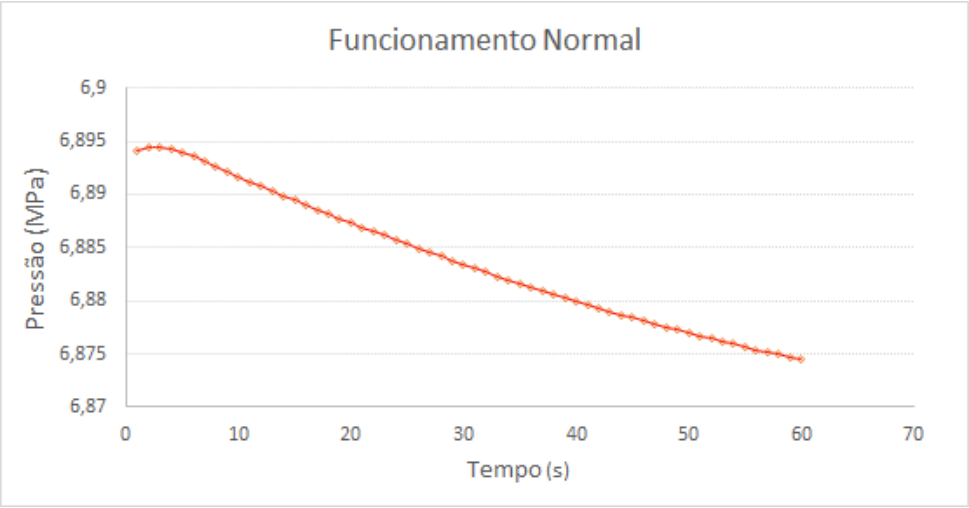


Figura 13 Gráfico dos dados de funcionamento normal

Para solucionar tal problema, tomou-se a média dos valores (Mv) e fez-se um novo grupo de 60 dados (apresentados da figura 14 e na tabela 3), onde cada valor é obtido somando um valor aleatório no intervalo de +- 0,001% do Mv, [-0,001%Mv , +0,001%Mv], ao Mv.

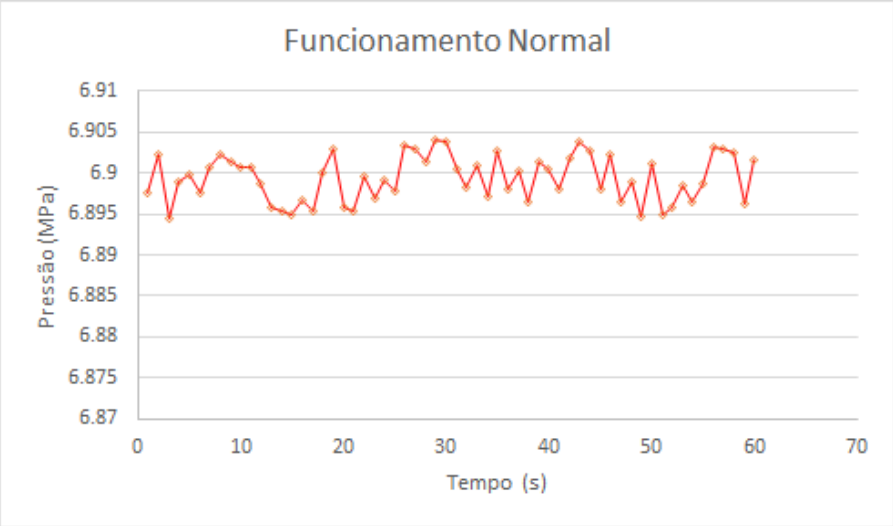


Figura 14 Gráfico dos novos dados de funcionamento normal

Tabela 3 Novos dados de funcionamento normal

Pressão (Mpa)	**Tempo (s)**	**Pressão (Mpa)**	**Tempo (s)**	**Pressão (Mpa)**	**Tempo (s)**	**Pressão (Mpa)**	**Tempo (s)**
6,8976229	1	6,8966539	16	6,90058124	31	6,90219113	46
6,9022161	2	6,8954360	17	6,89832749	32	6,89655817	47
6,8945667	3	6,9000661	18	6,90087017	33	6,89899825	48
6,8988924	4	6,9030349	19	6,89722378	34	6,89475971	49
6,8997799	5	6,8958213	20	6,90276909	35	6,90124908	50
6,8976444	6	6,8953974	21	6,89797234	36	6,89487366	51
6,9006112	7	6,8995513	22	6,90021086	37	6,89571513	52
6,9023315	8	6,8968775	23	6,89648273	38	6,89857184	53
6,9014461	9	6,8991929	24	6,90140849	39	6,89648710	54
6,9006849	10	6,8977507	25	6,90053729	40	6,89863010	55
6,9007714	11	6,9034951	26	6,89792738	41	6,90306887	56

6,8985968	12	6,9029401	27	6,90182394	42	6,90299986	57
6,8958680	13	6,9014560	28	6,90377100	43	6,90255894	58
6,8952571	14	6,9039599	29	6,90270491	44	6,89632782	59
6,8950144	15	6,9039313	30	6,89808984	45	6,90152955	60

3.3 Treinamento

Nesta seção, será discutido como foi feito o treinamento da ESN, considerando o que foi discutido nas seções anteriores, assim como as mudanças e adaptações que foram feitas ao longo do trabalho conforme os resultados foram sendo obtidos.

3.3.1 ESN

O formato e código da rede utilizada para este trabalho foi o proposto por Jaeger, e disponibilizado pelo próprio em (http://www.scholarpedia.org/article/Echo_state_network), sendo apenas traduzido de *MatLab* para *Python* 3.6.

Conforme apresentado na seção 2.1.7.1, os parâmetros da ESN que foram mantidos fixos foram: número de entradas e saídas, ruído e a função de ativação; enquanto os escolhidos para serem otimizados foram o raio espectral, esparsidade, tamanho do reservatório, *input shift*, *input scaling*, *teacher shift* e *teacher scaling.*

Conforme discutido, não foi imposta a restrição para que o raio espectral seja menor ou igual a 1, assim como não foi imposta nenhuma restrição para os valores de *input shift*, *input scaling*, *teacher shift* e *teacher scaling*. Para o tamanho do reservatório, observou-se a restrição de $T/10 \leq R \leq T/2$ (Jaeger, 2001), e para a esparsidade, por se tratar de uma porcentagem, a mesma foi mantida no intervalo $]0,1[$.

Após algumas simulações, percebendo a alta sensibilidade da rede mesmo a pequenas alterações em qualquer um dos parâmetros e conforme visto também em (Jiang, Berry & Schoenauer, 2008) e (Chouikhi *et al.*, 2017), decidiu-se centrar a inicialização dos parâmetros em torno dos melhores resultados obtidos até então, para acelerar o algoritmo, reduzindo o universo de exploração original. Com isto, o raio espectral foi inicializado com uma função geradora de números aleatórios (*random)*

centrado em 3; o *input shift* e o *teacher shift*, em 1; e o *input scaling* e o *teacher scaling*, em 10^{-2}. A esparsidade não foi alterada. Estas configurações foram mantidas ao longo de todo o experimento.

3.3.2 PSO

O pseudo-código para o PSO utilizado neste trabalho foi descrito em 3.2.4. Para os seus parâmetros ajustáveis (*w*, C1 e C2), durante a maior parte do experimento foi utilizado os valores de 0,729844, 1,49618 e 1,49618, respectivamente. Posteriormente foi feito um teste de convergência para C1 e C2.

Inicialmente, o PSO estava sendo utilizado para aprimorar os 7 parâmetros (raio, tamanho do reservatório, esparsidade, *input shift*, *input scaling*, *teacher shift* e *teacher scaling*). Porém, como o PSO foi uma ferramenta desenvolvida para a otimização de parâmetros contínuos (conforme destacado no capítulo anterior), estavam sendo encontrados problemas ao otimizar o número de neurônios do reservatório, uma vez que este é necessariamente discreto (isto é, não são aceitos números decimais, apenas números inteiros). Como este é o único parâmetro discreto a ser otimizado e o tempo para rodar cada otimização de treinamento era relativamente baixo (menos de 1 dia para 1000 gerações de PSO), optou-se por manter um valor de número de neurônios fixo e otimizar os outros 5 parâmetros. Após o fim de cada rodada do PSO, trocava-se o valor do número de neurônios e se repetia o processo. No final, comparava-se o valor do RMSE de cada rodada e tomou-se o menor erro para os parâmetros ideias de cada rede. Os valores do número de neurônios testados foram todos no intervalo [R/10 ; R/2].

Esta abordagem, justamente por produzir um número grande de resultados, levou a diversas descobertas inesperadas, que não eram o alvo principal deste estudo, porém foram de grande valor para o entendimento do funcionamento das ESN. Tais descobertas serão melhor discutidas no capítulo 4, porém a primeira delas foi que, conforme apontado por Martins (Martins, 2015), o PSO na otimização de ESN possui uma convergência muito rápida, chegando rapidamente ao mínimo local e dificilmente saindo dele. Além disto, as rodadas que apresentaram melhorias em quase todas as gerações foram as que apresentaram o pior resultado final, enquanto as que convergiram para o mínimo local logo nas primeiras gerações (algumas tão cedo quanto na quarta geração) foram as que apresentaram os melhores resultados. Com isto, reduziu-se o número de geração de cada rodada do PSO para 100, em um primeiro momento, e posteriormente para 30.

Após a primeira rodada de otimização (depois de se obter uma configuração ótima de rede para cada um dos casos: 3 acidentes e a operação normal), dedicou-se a solução do "problema" de continuidade do PSO, para que se pudesse comparar estes resultados com os de uma otimização completa do problema. A solução encontrada foi a de utilizar o PSO com o número de neurônios do reservatório continuamente (ou seja, deixá-los serem transformados em reais para que os cálculos fossem feitos) porém, na hora de calcular a função *fitness* (treinar a ESN), arredondar o número obtido. Ou seja, caso o primeiro dígito decimal do número fosse 5 ou maior, a função *fitness* seria calculada com a parte inteira do valor de N, acrescido de 1. Caso contrário, (o primeiro dígito decimal do número fosse menor do que 5) seria calculado apenas com a parte inteira, sem alterações.

Finalmente, uma última alteração feita para efeitos de comparação foi a aplicação da restrição do raio espectral estar no intervalo]0,1].

3.3.3 Recebimento e Análise de Dados em Tempo Real

Como a conexão com algum sistema de um reator nuclear não foi possível (uma vez que tais conexões, por motivo de segurança, são bastante restritas), foi criado um sistema *server-client* para que pudesse ser simulado e testado o envio e recebimento de dados em tempo real, e consequente teste da capacidade da rede em reconhecer o estado do reator.

3.3.3.1 Envio e Recebimento de Dados

Para o envio e recebimento de dados, foi escrito um programa em *python*, de autoria própria, específico para este estudo. Seu princípio de funcionamento consta em enviar uma lista d com 60 dados a cada segundo, $d(0)=[d_1,d_2,...,d_{60}]$, no tempo inicial t=0, para que fossem testados pela rede. A cada novo segundo, o valor mais antigo é substituído por um novo. Ou seja, no tempo t=1, a lista enviada é $d(1)=[d_2,d_3,...,d_{61}]$, e assim por diante.

Foram disponibilizadas 4 opções de teste, com dados diferentes a serem enviados: um para cada acidente (LOCA, SGTR e Blackout) e mais um para funcionamento normal. A seleção era feita apenas no servidor de envio, ou seja, o servidor que estava recebendo os dados não tinha informação de qual tipo de teste havia

sido selecionado. Com isto, pode-se colocar à prova a capacidade da rede de identificar o estado atual do reator.

Cada um dos testes possui uma lista pré-carregada de 120 dados, correspondente a 60 dados de operação normal mais 60 dados específicos do acidente (no caso da opção de operação normal, os 120 dados são de operação normal), conforme figura 15. No início do teste, a primeira lista enviada em todos os casos são os 60 dados de operação normal, e a cada segundo subsequente um dado de operação normal era substituído por um do acidente, até que, 60 segundos depois de iniciado o teste, seja enviada uma lista contendo apenas dados de acidente, encerrando-se assim um ciclo de teste.

```
In [55]: envio=[6.899779996996082, 6.8976444532993115, 6.90061122584567, 6.902331524149424, 6.901446193018727, 6.90068499803737,
6.90077146269844, 6.898596853024812, 6.8958680118685525, 6.895257121523339, 6.895014495204795, 6.89665398058184,
6.895436083154133, 6.900066184822864, 6.903034978960509, 6.895821376376537, 6.895397493034179, 6.899551352016308,
6.896877518951276, 6.899192928079877, 6.897750704008085, 6.903495193835388, 6.9029401562607005, 6.901456040194434,
6.903959976406766, 6.903931314114493, 6.900581242604314, 6.898327494027394, 6.900870173639479, 6.897223789278369,
6.902769097057846, 6.897972345333851, 6.90021086350707, 6.896482734795506, 6.901408491404248, 6.900537297135435,
6.897927382120521, 6.901823944400938, 6.903771000977892, 6.902704915656927, 6.898089845838607, 6.902191130646763,
6.89655817031335, 6.898998254662329, 6.894759711946095, 6.901249088608959, 6.894873666985928, 6.895715136570018,
6.898571849089868, 6.896487101511636, 6.898630100737143, 6.903068876233201, 6.902999866942624, 6.902558940243312,
6.896327829051072, 6.901529549779121, 6.897622950171866, 6.902216128797056, 6.894566739623984,
6.898892412444011,6.89413,6.89596,6.90081,6.90465,6.90742,6.90968,6.91156,6.91309,6.91433,6.91525,6.91597,6.91652,6.91691,6.91724,
6.91761,6.91784,6.91797,6.91804,6.918,6.91793,6.91781,6.91767,6.91752,6.91721,6.91681,6.91625,6.91563,6.91502,6.91441,6.91381,6.91
323,6.91258,6.91186,6.91113,6.91043,6.90971,6.90896,6.90823,6.90746,6.9067,6.90594,6.90515,6.90438,6.90358,6.90277,6.90196,6.90114
,6.90033,6.89954,6.8987,6.89783,6.89698,6.89619,6.89539,6.89456,6.89377,6.89294,6.89215,6.89133,6.89052]
```

Figura 15 Exemplo de lista de dados de envio para o acidente LOCA, com o separador em azul indicando onde acabam os dados de operação normal e onde começam os de acidente

3.3.3.2 Teste dos Dados

Ao receber a lista de dados mencionada acima, o servidor de recebimento, possuindo os parâmetros otimizados para cada um dos 4 tipos de teste, constrói uma rede para cada caso, com estes parâmetros e a lista recebida, e compara os RMSE obtidos para determinar qual o estado do reator, e imprime o resultado na tela. A cada segundo, com uma nova lista, o processo se repete, até que o estado do reator seja corretamente identificado.

Na figura 16, é mostrado um exemplo dos resultados obtidos para o teste com o acidente SGTR.

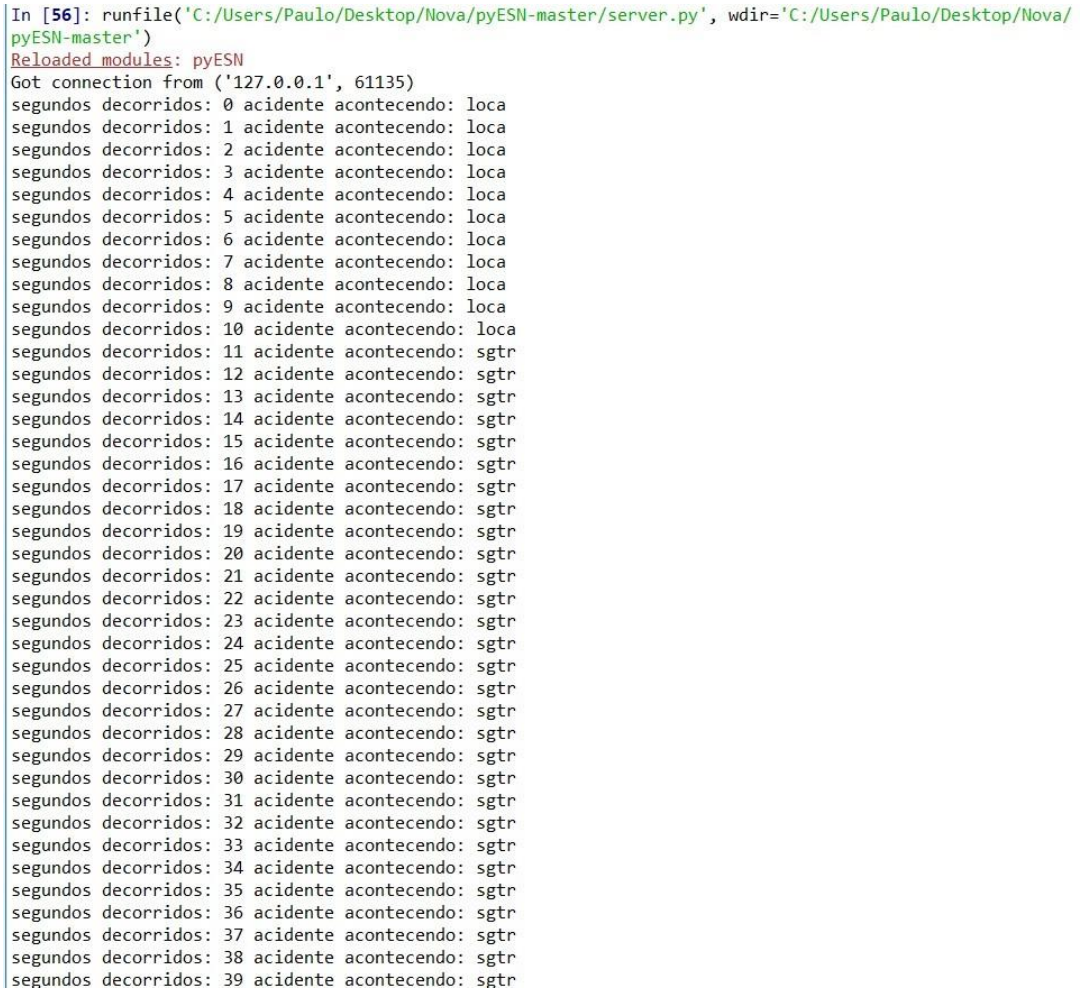


Figura 16 Exemplo da tela do teste para o acidente SGTR

3.3.4 Mudanças e Atualizações na Metodologia

Nesta seção serão apresentadas as mudanças que foram feitas na filosofia e condições iniciais do trabalho, em virtude das descobertas e resultados que foram sendo obtidos ao longo dos experimentos.

Na primeira delas, para se executar a fase de testes (envio/recebimento de dados), otimizou-se as redes para identificar o comportamento dos 60 segundos de dados do acidente específico. Porém, na hora de comparar com a lista de 60 dados que começam na operação normal e gradualmente vão tomando a forma do acidente, como a rede só estava treinada para reconhecer a forma final, ela demorava muito para reconhecer o acidente que estava ocorrendo. Para solucionar este problema, decidiu-se por alterar a lista de envio/recebimento de dados para um total de 120 dados, que

inicialmente conta com 120 dados de operação normal, e que vai progredindo ao longo de 60 segundos até finalizar com 60 dados de operação normal e 60 dados específicos do acidente. Com isto, as redes tiveram que ser treinadas e otimizadas novamente, desta vez com 600 dados de treinamento (120*5).

Esta mudança, porém, ainda não foi suficiente para que o sistema preditivo identificasse o estado do reator com a rapidez desejada. A identificação ocorria, em média, apenas 40 segundos após o começo do acidente e, para o caso do LOCA e SGTR, esta decisão não era estável, fazendo com que a rede ficasse "trocando" de decisão, conforme pode ser visto na figura 17.

Isto estava ocorrendo, pois, a comparação entre os erros para determinar o estado atual do reator estava sendo feita de forma absoluta, descrito pela equação (23).

$$RMSE_{esperado} - RMSE_{obtido}. \quad (23)$$

O erro encontrado para a operação normal é uma ordem de grandeza inferior aos outros, fazendo com que o sistema preditivo apenas assumisse que um determinado acidente estava ocorrendo após metade desta diferença ser tirada, o que acontecia apenas nos segundos finais do teste. Além disto, o erro do Blackout é de ordens de grandeza maior do que os outros, fazendo com que este fosse detectado apenas no último segundo.

Ao se perceber isto, a fórmula de análise de erro no servidor de recebimento (cliente) foi trocada para o erro percentual, conforme equação 24:

$$|(RMSE_{esperado} - RMSE_{obtido})| / RMSE_{obtido} \quad (24)$$

Com esta mudança, o sistema preditivo passou a identificar corretamente o estado do reator, porém ainda com alguma demora (28 segundos).

```
In [5]: runfile('C:/Users/Paulo/Desktop/Nova/pyESN-master/server.py'
Got connection from ('127.0.0.1', 59642)
segundos decorridos: 0 acidente acontecendo: nenhum
segundos decorridos: 1 acidente acontecendo: nenhum
segundos decorridos: 2 acidente acontecendo: nenhum
segundos decorridos: 3 acidente acontecendo: nenhum
segundos decorridos: 4 acidente acontecendo: nenhum
segundos decorridos: 5 acidente acontecendo: nenhum
segundos decorridos: 6 acidente acontecendo: nenhum
segundos decorridos: 7 acidente acontecendo: nenhum
segundos decorridos: 8 acidente acontecendo: nenhum
segundos decorridos: 9 acidente acontecendo: nenhum
segundos decorridos: 10 acidente acontecendo: nenhum
segundos decorridos: 11 acidente acontecendo: nenhum
segundos decorridos: 12 acidente acontecendo: nenhum
segundos decorridos: 13 acidente acontecendo: nenhum
segundos decorridos: 14 acidente acontecendo: loca
segundos decorridos: 15 acidente acontecendo: nenhum
segundos decorridos: 16 acidente acontecendo: nenhum
segundos decorridos: 17 acidente acontecendo: nenhum
segundos decorridos: 18 acidente acontecendo: nenhum
segundos decorridos: 19 acidente acontecendo: loca
segundos decorridos: 20 acidente acontecendo: loca
segundos decorridos: 21 acidente acontecendo: loca
segundos decorridos: 22 acidente acontecendo: loca
segundos decorridos: 23 acidente acontecendo: loca
segundos decorridos: 24 acidente acontecendo: loca
segundos decorridos: 25 acidente acontecendo: loca
segundos decorridos: 26 acidente acontecendo: loca
segundos decorridos: 27 acidente acontecendo: loca
segundos decorridos: 28 acidente acontecendo: sgtr
segundos decorridos: 29 acidente acontecendo: sgtr
segundos decorridos: 30 acidente acontecendo: sgtr
segundos decorridos: 31 acidente acontecendo: sgtr
segundos decorridos: 32 acidente acontecendo: sgtr
segundos decorridos: 33 acidente acontecendo: sgtr
segundos decorridos: 34 acidente acontecendo: sgtr
segundos decorridos: 35 acidente acontecendo: sgtr
segundos decorridos: 36 acidente acontecendo: sgtr
segundos decorridos: 37 acidente acontecendo: sgtr
segundos decorridos: 38 acidente acontecendo: sgtr
segundos decorridos: 39 acidente acontecendo: sgtr
segundos decorridos: 40 acidente acontecendo: sgtr
segundos decorridos: 41 acidente acontecendo: sgtr
```

Figura 17 Tela com a "indecisão" da rede

4 – Resultados e Discussões

4.1 Introdução

Uma vez que o sistema preditivo estava funcionando corretamente e produzindo resultados aceitáveis, foram propostos alguns outros testes, para verificação e validação dos resultados obtidos.

Nesta seção, serão apresentados os testes realizados e os resultados obtidos em cada teste, assim como uma discussão sobre o quê este resultado representa para o teste, suas causas e consequências, tabelas e gráficos ilustrativos.

Para os testes de comparação, cada caso foi rodado 10 vezes. Os parâmetros e erros mostrados aqui foram truncados para melhor visualização, e os resultados completos são mostrados no Anexo I.

Serão tratados os seguintes casos: aumento no número de dados de treino; correlação tamanho do reservatório *versus* erro; comparação entre *gbest* inicial e final; alteração de um dos 6 valores após a rede otimizada; fixação do objetivo de erro ao invés de número máximo de gerações; limitação da variação do raio espectral; testes de convergência de C1 e C2; fixação de 5 parâmetros para trabalhar apenas com 1; remoção do *ruído*; correlação entre erro e raio espectral; além dos resultados do objetivo principal do trabalho.

4.2 Aumento no Número de Dados de Treinamento

Conforme discutido na seção 2.2.3.2, há indícios que apontam uma dependência entre o número de dados de treinamento e o desempenho da ESN. Tal hipótese foi colocada a prova.

Ao se deparar com apenas 60 dados totais, decidiu-se utilizar 30 para treinamento e 30 para teste, antes mesmo da etapa de otimização, apenas para se ter uma ideia do comportamento. A figura 18 mostra o resultado, no qual pode ser observado que o comportamento dos dados simulados está longe do ideal; e a tabela 4 apresenta os parâmetros utilizados.

Tabela 4 Parâmetros para as redes das figuras 18 a 20

Reservatório	Raio	*Input Shift*	*Input Scaling*	*Teacher Shift*	*Teacher Scaling*	*Sparsity*	Ruído
15	0,95	1	1	1	1	0,85	0,001

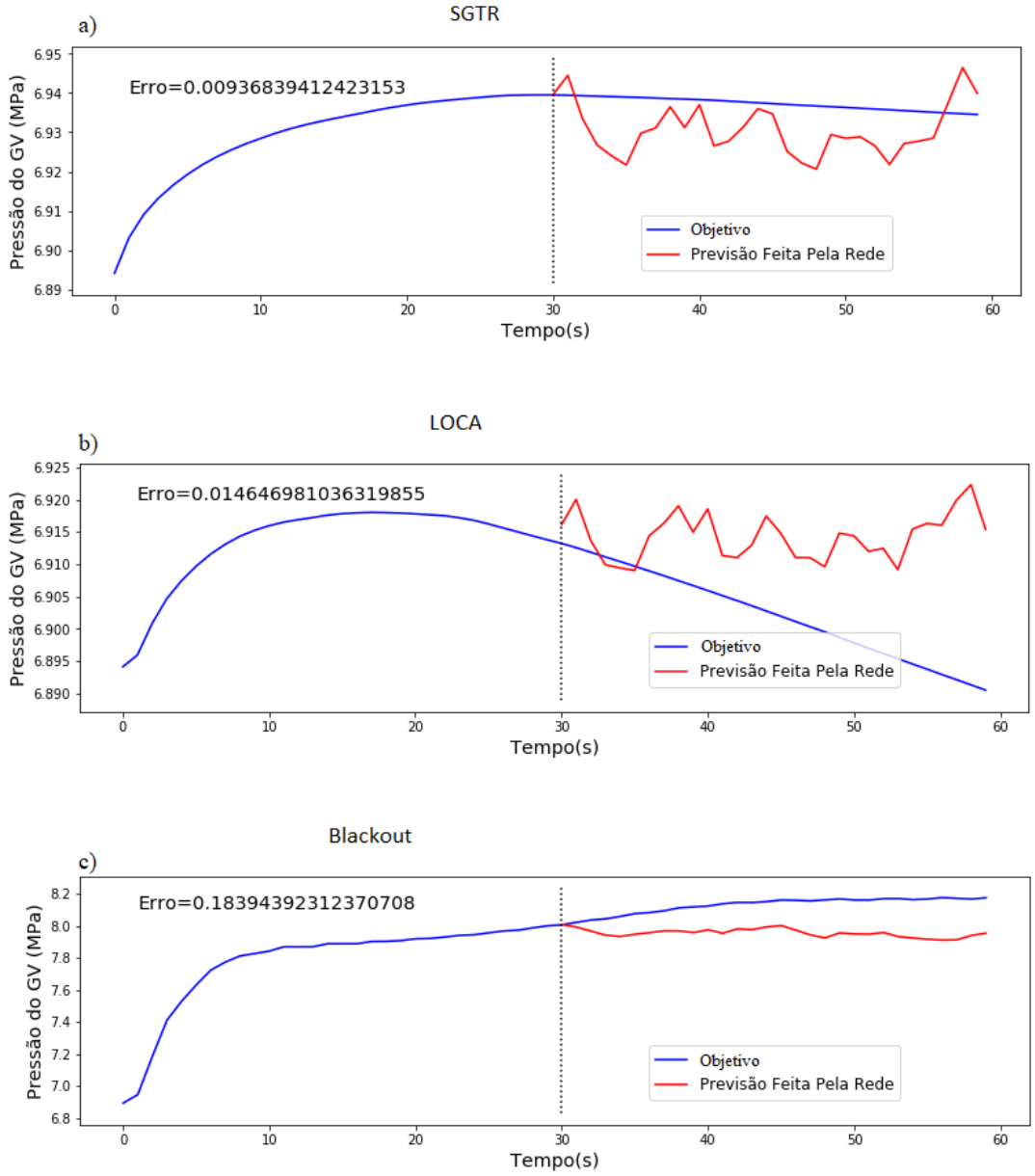


Figura 18 Resultado da rede com 30 dados de treinamento e 30 de teste, para cada um dos acidentes: a) SGTR ; b) LOCA ; c) Blackout

Como este erro estava bastante elevado (apesar de ser numericamente baixo, graficamente ainda apresenta uma discrepância grande), decidiu-se aumentar a quantidade de dados utilizados para treinamento para 50, deixando 10 dados para teste. O resultado encontra-se na Figura 19 e mostra que, apesar de haver uma melhora gráfica no comportamento da rede, sua atuação fica muito restrita. Os parâmetros são os mesmos da tabela 4.

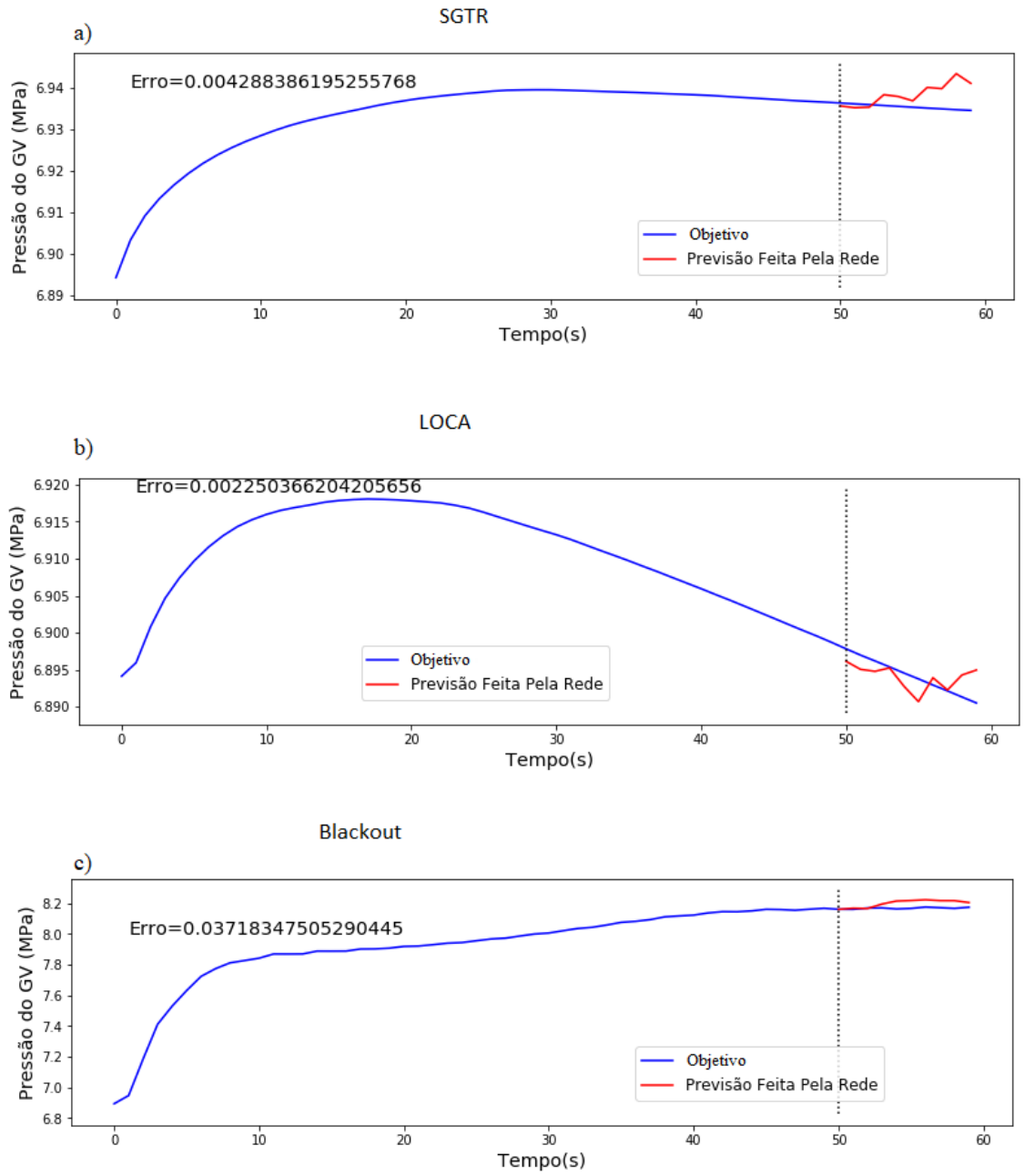


Figura 19 Resultado da rede com 50 dados de treinamento e 10 de teste, para cada um dos acidentes: a) SGTR ; b) LOCA ; c) Blackout

Com o erro permanecendo alto, chegou-se à conclusão, neste caso, que o maior fator influenciando era o fato de que os dados de treino e de teste não eram os mesmos. Isto se dá pelo fraco desempenho da ESN para extrapolação, uma vez que a concepção da mesma é a de replicar o comportamento dos dados de treinamento. Além disto, estas configurações não permitiam que a rede fosse treinada em todo o espectro do trabalho, sendo restrita apenas a metade (30/60) ou um sexto (10/60) dos dados totais.

Cabe mencionar que o resultado para o acidente Blackout aparentemente está satisfatório. Isto se deu, pois, a rede estava restrita à parte do acidente que se comporta de forma linear (os 10 segundos finais). Este comportamento (linear) não é, porém, o comportamento de todo o espectro dos dados deste acidente.

Seguiu-se então, a utilizar todos os 60 dados como treinamento, e os mesmos 60 para teste. O primeiro resultado obtido, com os parâmetros da tabela 4, encontra-se na figura 20, onde pode ser constatado que a rede ainda não está ótima.

Nesta etapa do trabalho notou-se a importância de se otimizar os parâmetros da ESN, juntamente com o descrito por (Chouikhi *et al.*, 2017; Jiang, Berry & Schoenauer, 2008), e que realizar testes rápidos, com parâmetros aleatórios, gera resultados aleatórios e inconclusivos. Por este motivo, deste ponto em diante, fez-se uma otimização rápida com o PSO (com apenas 35 iterações) dos parâmetros da ESN sempre, antes de um novo teste.

Após, como descrito na seção 3.2.1, criou-se um novo grupo de dados para treinamento com cinco vezes o tamanho original, composto por 5 réplicas dos 60 dados iniciais, compondo 300 dados de treinamento. Conforme pode ser observado pela figura 21, a melhora no erro ocorreu de forma geral (porém ainda longe do ótimo). Os parâmetros estão na tabela 5.

Cabe ressaltar que, para a maioria dos testes, além da melhora gráfica, o erro também diminuiu bastante, e este é o objetivo final do trabalho.

Com esta melhora, também foi testado um tamanho de dado de testes de 600 (10 vezes maior que o original). Os resultados desse teste se encontram na figura 22, e os parâmetros, tabela 6.

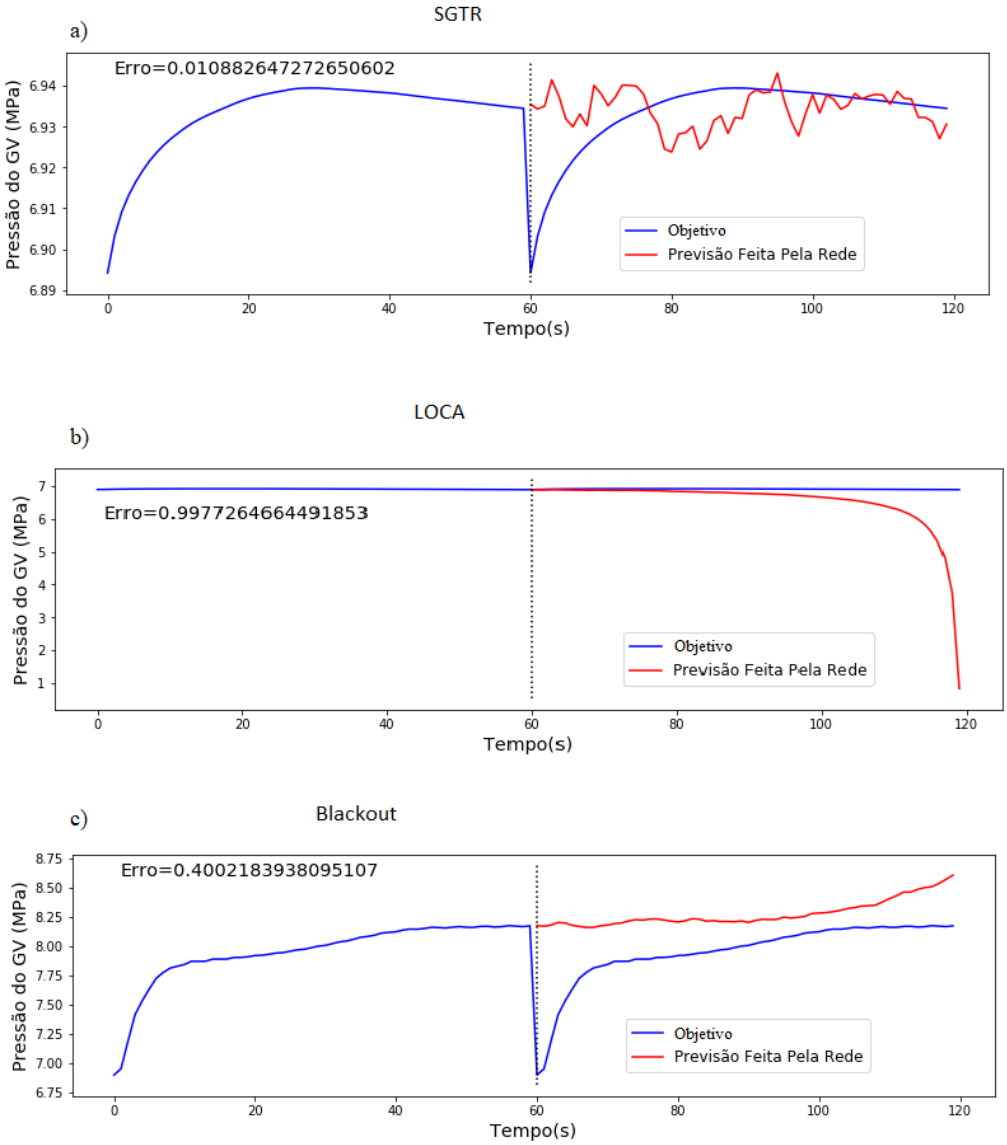


Figura 20 Resultado da rede com 60 dados de treinamento e 60 de teste, para cada um dos acidentes: a) SGTR ; b) LOCA ; c) Blackout

Tabela 5 Parâmetros encontrados após otimização, para a figura 21

Acidente	Reserva-tório	Raio	*Input Shift*	*Input Scaling*	*Teacher Shift*	*Teacher Scaling*	*Sparsity*	Ruído
Blackout	123	3,362	0,5151	0,00668	-2,817	26,11	0,9408	0,001
LOCA	55	-2,76	3,1	0,075	-0,596	2,119	0,162	0,001
SGTR	90	2,843	0,0552	-0,0204	0,0011	3,076	0,757	0,001

Tabela 6 Parâmetros encontrados após otimização, para a figura 22

Acidente	Reserva-tório	Raio	*Input Shift*	*Input Scaling*	*Teacher Shift*	*Teacher Scaling*	*Sparsity*	Ruído
Blackout	139	3,138	2,0491	0,0408	-0,837	0,843	0,689	0,001
LOCA	88	2,784	8,1377	0,0248	-23,977	171,412	0,922	0,001
SGTR	176	2,462	2,2136	-0,010	-0,0829	0,725	0,757	0,001

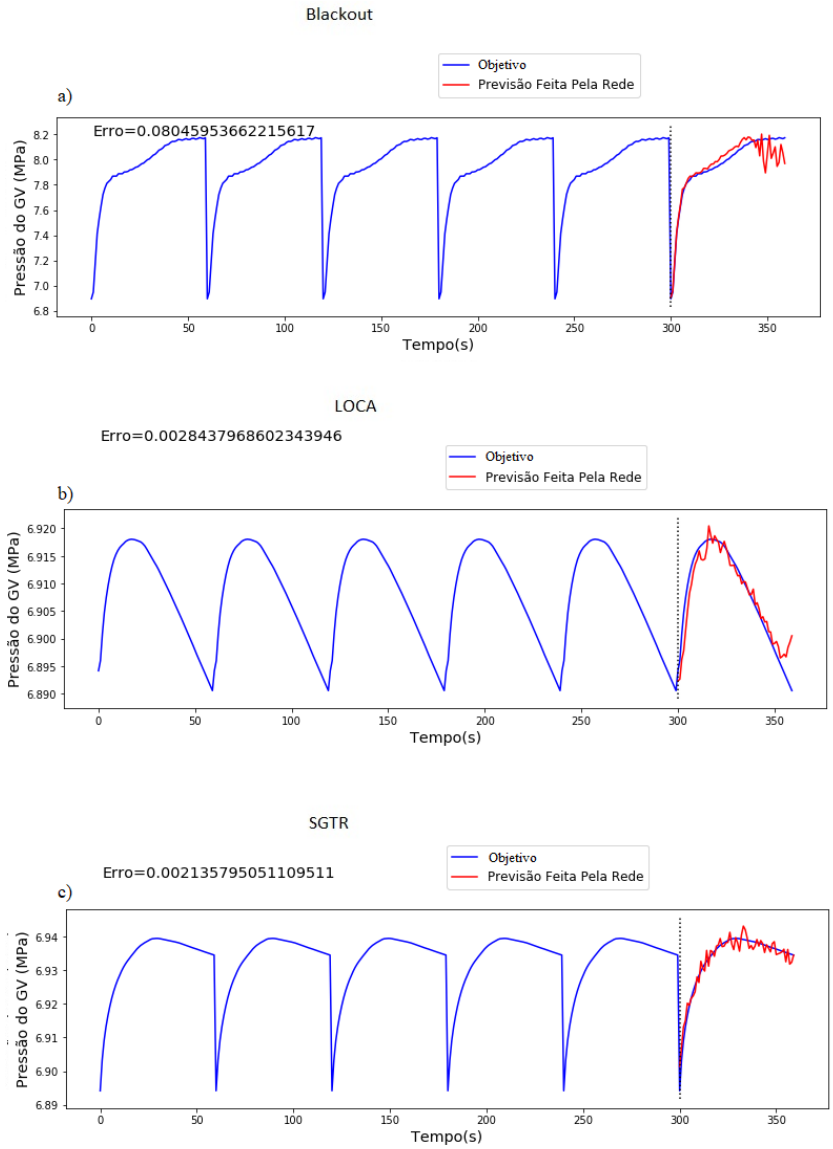


Figura 21 Resultado da rede com 300 dados de treinamento e 60 de teste, para cada um dos acidentes: a) SGTR ; b) LOCA ; c) Blackout

Comparando estes resultados com os do teste com 300 *inputs*, pode-se perceber uma melhora considerável nos acidentes LOCA e SGTR (tanto numérica quanto graficamente), porém uma piora (em menor grandeza) para o acidente Blackout. Os motivos para isto não ficaram muito claros a princípio. Porém considerando que o comportamento da variável durante o acidente de Blackout é bastante não linear e diferente do comportamento dos outros dois acidentes, isto provavelmente teve influência no resultado. Isto sugere que, devido a esta diferença, o tratamento da ESN para os casos tenha que ser diferente.

Posteriormente, com a mudança na quantidade de dados a ser enviada (afim de que a ESN imite o comportamento ao longo de 120 segundos), obteve-se a configuração final de 600 dados de entrada (5*(60+60)), e 120 (60+60) dados de teste. Os resultados estão na figura 22 e na tabela 7.

Vale notar que neste caso, diferentemente do caso apresentado na figura 22 (na qual os 600 dados de entrada eram os dados do comportamento do acidente), neste caso apenas 300 dos 600 eram do comportamento estudado, enquanto os outros 300 eram dados de funcionamento, fazendo com que a curva a ser acompanhada e prevista pela rede mude drasticamente. Isto é a principal explicação para que o erro dos 3 casos aumentou quando se passou para esta abordagem.

Além disto, novamente, os resultados para o Blackout foram piores que os primeiros. Cabe ressaltar que, de forma geral, a rede tem mais problemas em imitar o comportamento do LOCA e do SGTR na fase de comportamento normal do que na curva do acidente propriamente dito.

Tabela 7 Parâmetros encontrados após otimização, para a figura 23

Acidente	Reserva-tório	Raio	Input Shift	Input Scaling	Teacher Shift	Teacher Scaling	Sparsity	Ruído
Blackout	230	1,43	0,92	0,0106	0,0099	-0,3098	0,78	0,001
LOCA	272	5,186	-22,927	-0,126	-2,42	0,9772	0,6776	0,001
SGTR	202	-2,84	3,9771	0,018	0,477	0,9162	0,5965	0,001

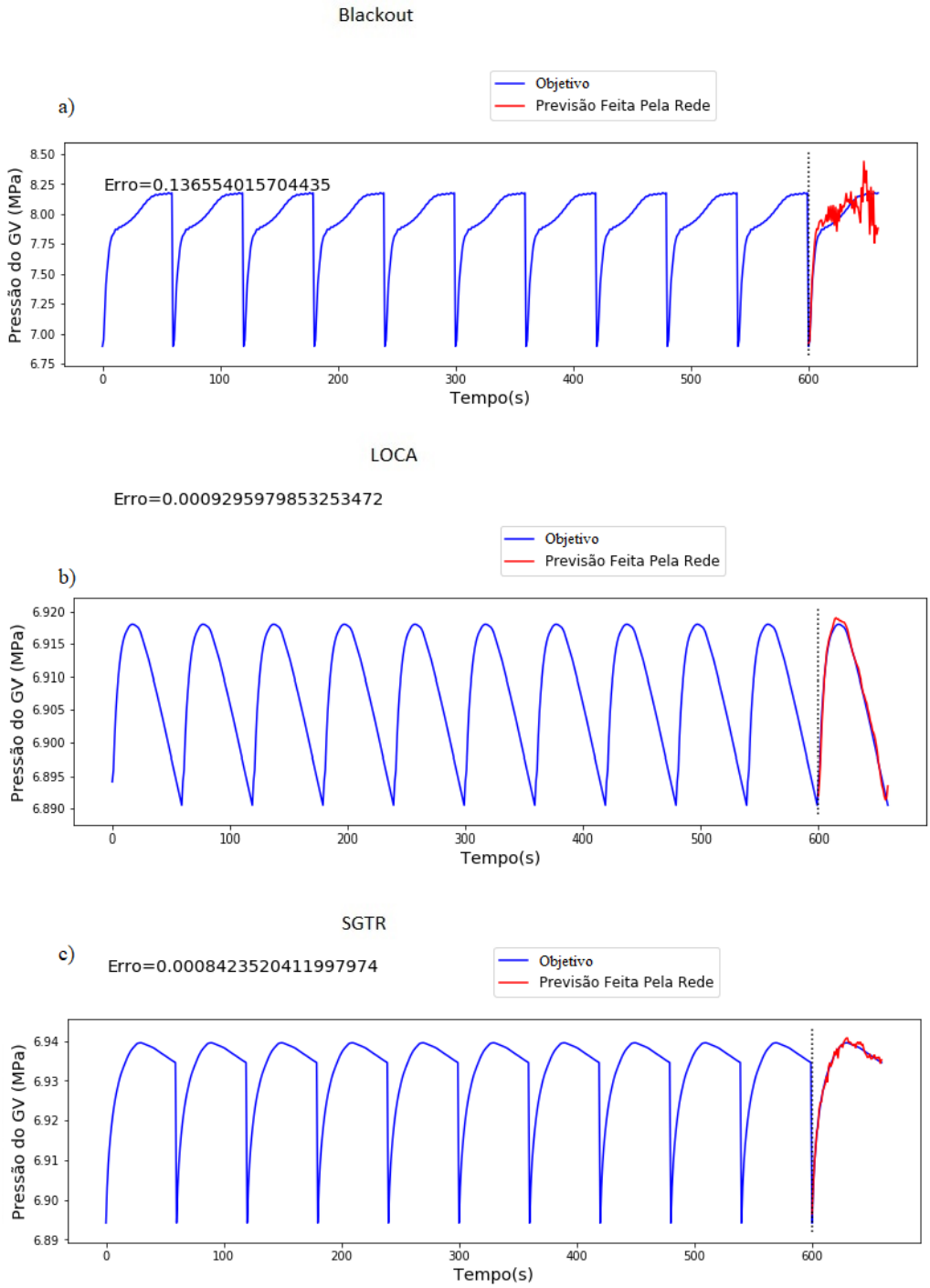


Figura 22 Resultado da rede com 600 dados de treinamento e 60 de teste, para cada um dos acidentes: a) Blackout ; b)LOCA ; c) SGTR

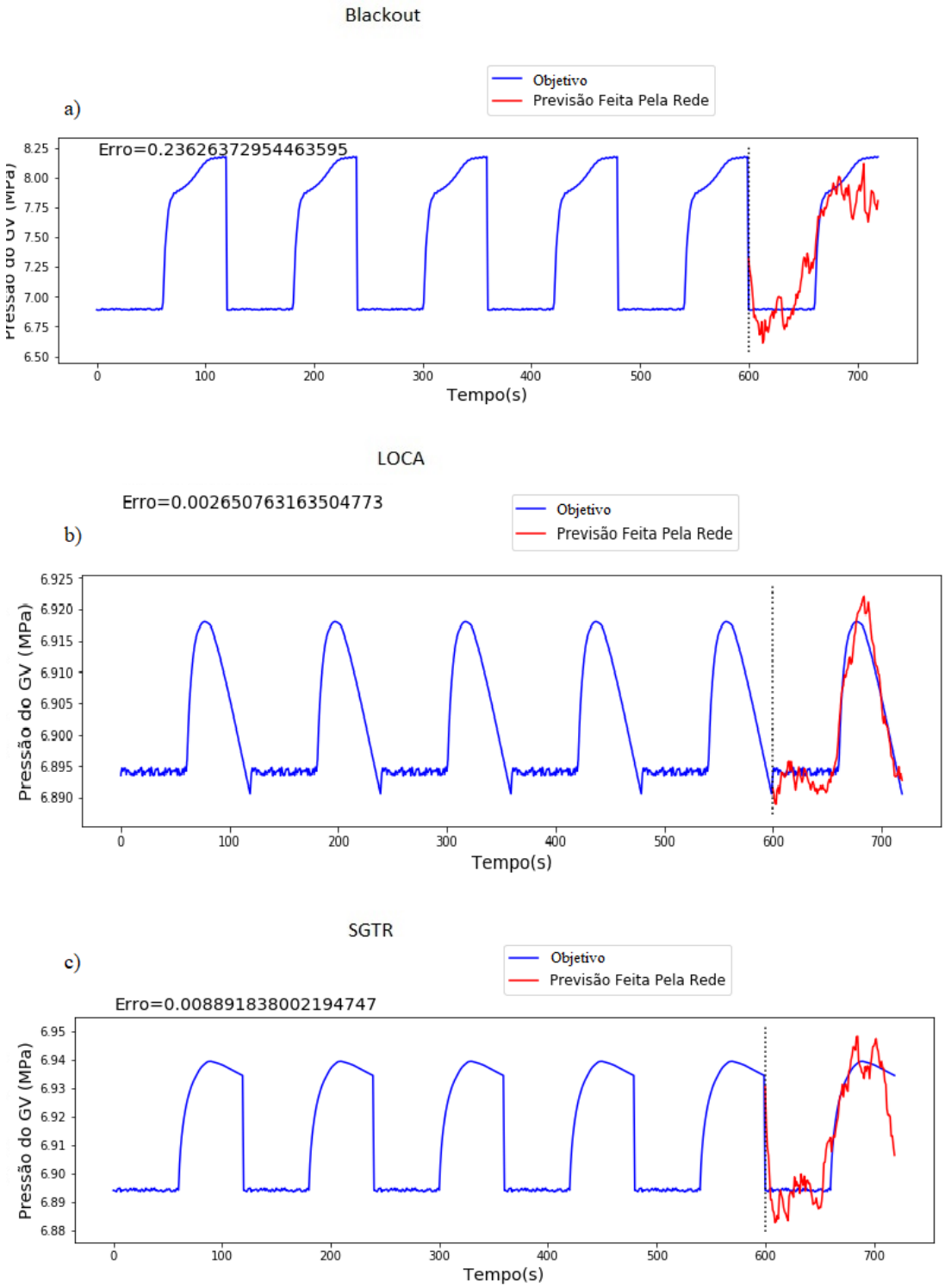


Figura 23 Resultado da rede com 600 dados de treinamento e 120 de teste, para cada um dos acidentes: a) Blackout ; b)LOCA ; c) SGTR

Embora estes resultados sozinhos não sejam capazes de provar definitivamente que um aumento puro no número de dados, os casos do LOCA e SGTR sugerem que, dependendo da forma do problema, esta hipótese tem fundamento.

4.3 Correlação Entre Tamanho do Reservatório e Erro

Por falta de unanimidade na literatura analisada sobre a influência do tamanho do reservatório no erro, e com a obtenção do resultado da otimização de todo o intervalo de variação do reservatório, decidiu-se então analisar a variação do erro conforme o número de neurônios variava. Os resultados obtidos para os casos com 600 dados de entrada estão nas tabelas 8 e 9 e na figura 24. Os com 300 estão no Anexo I.

Tabela 8 Análise sobre o Erro: Média, desvio padrão e diferença de valores de cada acidente

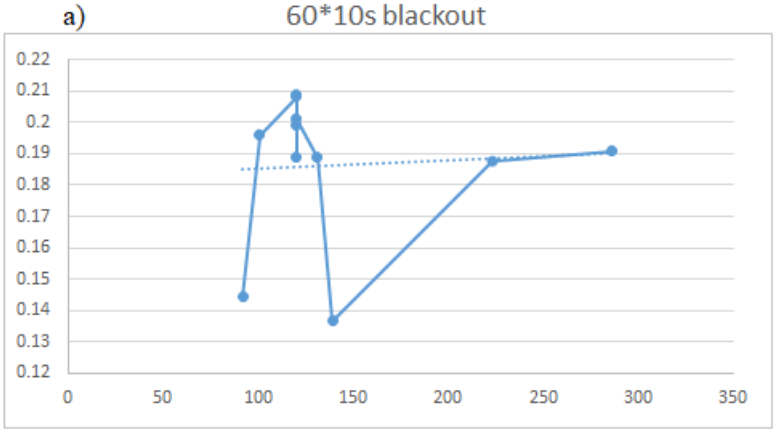
Erro					
Acidente	Média	Desvio Padrão	% do desvio padrão em relação ao erro	Diferença entre o maior e o menor	Diferença entre o maior e o menor (%)
Blackout 60*10	0,186	0,022	12,2	0,072	52,7
LOCA 60*10	0,003	0,0015	51,07	0,004	500,3
SGTR 60*10	0,005	0,0017	31,2	0,006	755,9
Blackout 120*5	0,315	0,0409	12,9	0,127	50,8
LOCA 120*5	0,004	0,0003	7,7	0,001	34,5
SGTR 120*5	0,0119	0,0012	10,4	0,004	41,6

Tabela 9 Análise sobre o Erro: Média, desvio padrão e diferença de valores de cada acidente

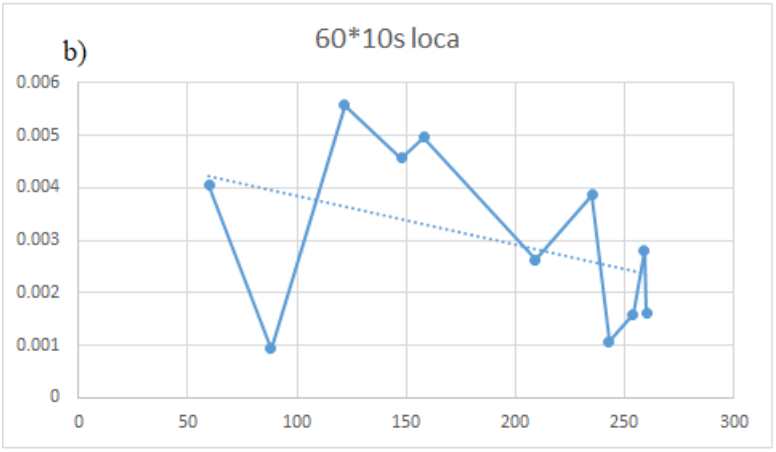
Reservatório					
Acidente	Média	Desvio Padrão	% do desvio padrão em relação ao	Diferença entre o maior e o menor	Diferença entre o maior e o menor (%)

erro					
Blackout 60*10	142,9	55,6	38,9	194	210,8
LOCA 60*10	185	69,6	37,6	200	333,3
SGTR 60*10	172	57,5	33,4	177	210,7
Blackout 120*5	174	56,7	28,9	174	179,3
LOCA 120*5	225	78,1	39,7	225	300
SGTR 120*5	201	62,9	30,6	201	213,8

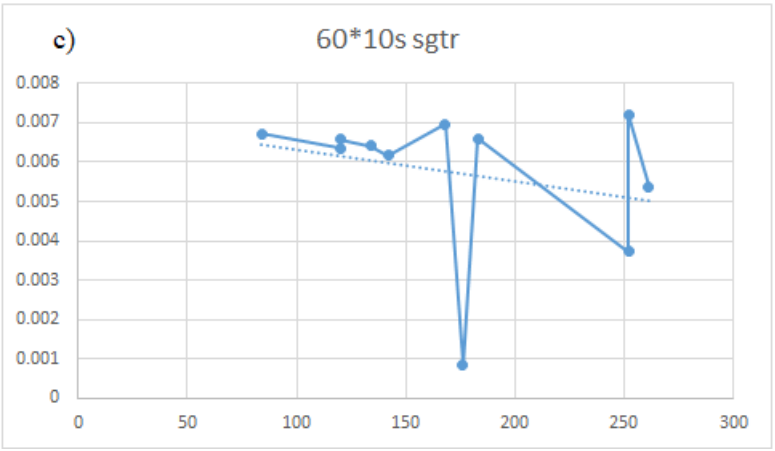
a)


60*10s blackout
0.22
0.21
0.2
0.19
0.18
0.17
0.16
0.15
0.14
0.13
0.12
0
50
100
150
200
250
300
350

b)


60*10s loca
0.006
0.005
0.004
0.003
0.002
0.001
0
0
50
100
150
200
250
300

c)


60*10s sgtr
0.008
0.007
0.006
0.005
0.004
0.003
0.002
0.001
0
0
50
100
150
200
250
300

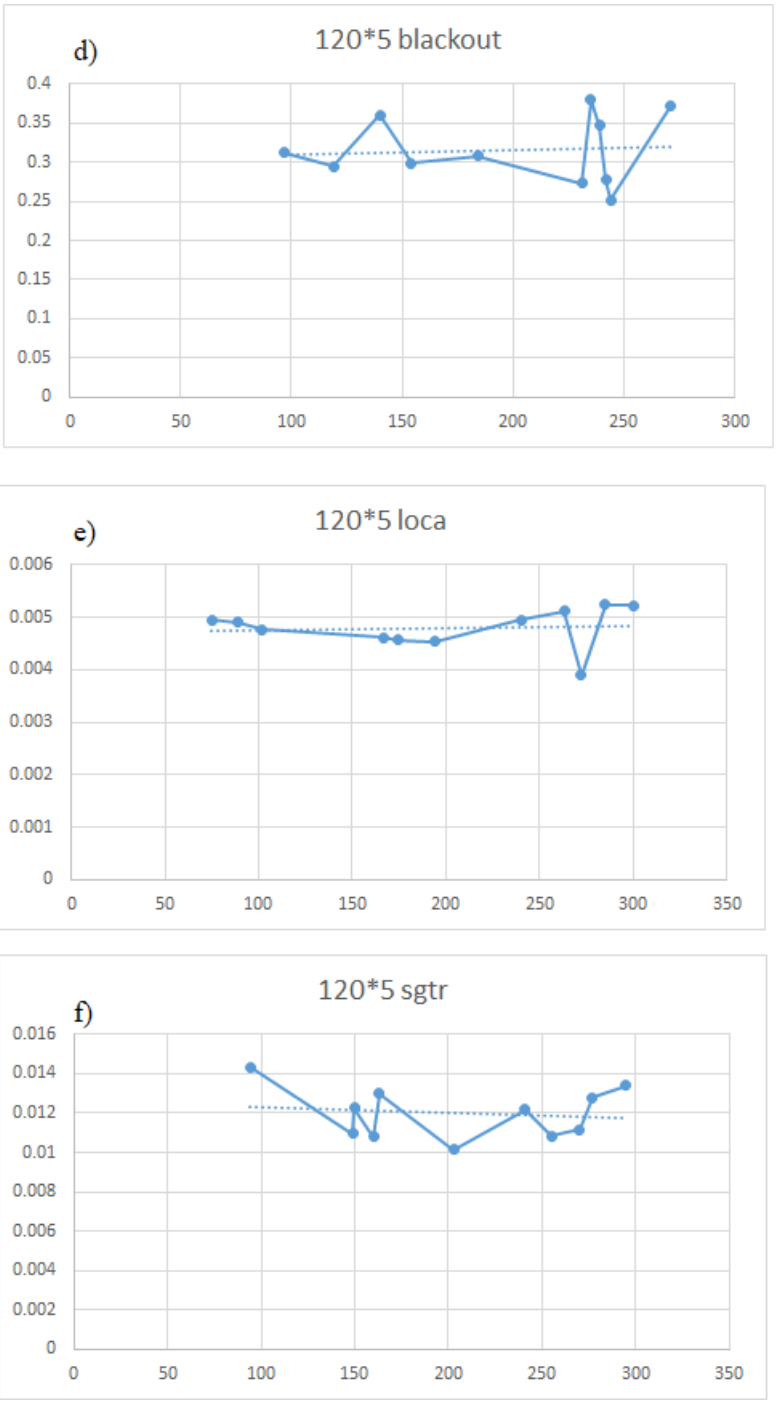


*Figura 24 Gráficos da dispersão Erro versus Tamanho do Reservatório, para cada dos um dos casos: a) Blackout com 60*10s ; b)LOCA com 60*10s ; c) SGTR com 60*10s; d) Blackout com 120*5s ; e)LOCA com 120*5s ; f) SGTR com 120*5s. A linha tracejada representa a linha de tendência*

Como pode-se observar na figura 24, as linhas de tendência não são unânimes, e os gráficos de distribuição são bastante dispersos. Além disto, a variação dos resultados é muito grande, denotado pelo desvio padrão bastante elevado - entre 7,7% e 51,07% do valor da média (tabela 8), assim como é elevada a diferença entre o maior e o menor

valor (chegando até a mais de 755,9%). Ou seja, não há uma definição concreta sobre se o aumento no tamanho do reservatório leva a um aumento ou diminuição do erro.

O mesmo pode ser observado com o tamanho do reservatório (tabela 9). Tanto a porcentagem do desvio padrão em relação à média (próximo a 30% nos diversos casos) quanto a porcentagem da variação entre o maior e o menor valor obtido nas 10 rodadas para cada acidente (variando de 179,3% a 333,3%) são bem elevadas, e não apontam a uma direção clara a se seguir para um valor universalmente ótimo.

Isto denota o fato de que a otimização se faz bastante eficiente, sendo capaz de encontrar um resultado satisfatório para cada condição imposta (neste caso, tamanho do reservatório), o que torna a escolha deste parâmetro bastante flexível, podendo ser escolhidos valores que melhor se adequem ao problema, para que não se tenha *overfitting* (Jaeger, 2010), ou um esforço computacional grande, podendo-se inclusive fixar um valor de tamanho de reservatório e otimizar os demais parâmetros em torno deste, como descrito na seção 4.9.

4.4 Comparação Entre o *gbest* Inicial e Final

Procurando fundamentar a hipótese de que a otimização dos parâmetros da ESN se faz necessária (Chouikhi *et al.*, 2017; Jiang, Berry, & Schoenauer, 2008), foi feita uma comparação entre o *gbest* inicial (ou seja, o melhor valor de erro encontrado dentre todos os indivíduos após seus parâmetros serem inicializados de forma aleatória) e o *gbest* final, para se ter a dimensão da melhora. Os resultados são representados na tabela 10.

Tabela 10 Valores gbest inicial versus *final para o melhor resultado de cada acidente*

Acidente	*gbest* inicial	*gbest* final	% de melhora
Blackout 60*10	0,25	0,136	45,5
LOCA 60*10	0,0075	0,00092	87,6
SGTR 60*10	0,0085	0,00084	90,1
Blackout 120*5	0,472	0,2362	50
LOCA	0,0083	0,0026	68,3

120*5			
SGTR 120*5	0,0187	0,00889	52,6

Com estes resultados, pode-se observar que, mesmo com uma quantidade grande de partículas iniciais (300), os valores iniciais do erro são bem grandes. Além disto, é notável a melhora que se obtém com o PSO, mesmo quando se utiliza poucas iterações. Por exemplo, 35 iterações (que duram menos de 15 minutos para acontecer, no pior dos casos, em um notebook com 8GB de RAM e processador *intel* i5) oferecem uma melhora bastante significativa (entre 50 e 90%).

Este é mais um argumento bastante forte para a teoria de que, para utilizar a ESN, se faz necessária sua otimização prévia (mesmo que com poucas iterações).

4.5 Alteração de Valores Após a Otimização

Para testar e confirmar a sensibilidade da ESN a alterações nos parâmetros após a otimização (conforme visto na seção 3.3.4), rodou-se a rede com a mesma configuração, alterando apenas 1 parâmetro cada vez, conforme a tabela 12. Aqui serão exibidos os resultados para o acidente LOCA, com 120*5 dados de treinamento, porém o mesmo fenômeno ocorre em todos os casos estudados neste trabalho. Os dados estão exibidos truncados, para maior clareza.

Tabela 11 Tabela apresentando a alteração nos valores e suas correspondentes alterações no RMSE

Mudança	+1 reservatório	-1 reservatório	-1% raio	+1% raio	+10% raio	-10% raio
Novo erro	0,0178	0,6136	0,0031358	0,004613	0,007007	2,013
% de diferença do original	571,5047291	23048,05066	18,29800727	74,02535479	164,3389684	75840,394
Mudança	-1% sparsity	+1% sparsity	-1% input shift	+1% input shift	-1% input scaling	
Novo erro	0,008217	0,005399	0,005132	0,003268	0,0026527	
% de diferença do original	209,9861999	103,6771928	93,60462189	23,28525026	0,073067127	
Mudança	+1% input scaling	-1% teacher shift	+1% teacher shift	-1% teacher scaling	+1% teacher scaling	
Novo erro	0,0026975	0,00517	0,02148	0,00269	0,0026649	
% de diferença do original	1,763146445	95,03817132	710,3326731	1,48020906	0,533311942	

Como pode ser observado na tabela 11, a variação é muito grande (entre 1,5 e 23.048% no RMSE), mesmo com alterações mínimas (1% do valor do parâmetro que levou ao melhor resultado). Vale ressaltar que, como o tamanho do reservatório é um número discreto, a menor alteração que se pode fazer é de uma unidade, que neste caso representa 0,37% do tamanho original do reservatório e, mesmo sendo a menor alteração percentual, promove o maior aumento do erro, mostrando que este é um dos parâmetros mais sensíveis da rede, seguido pela *sparsity* e pelo raio (confirmando o que é comumente encontrado na literatura).

Isto denota o quão eficiente é a otimização do PSO, além da forte característica especialista e de "eco" da ESN, fazendo com que ele execute perfeitamente sua função de "ecoar" situações às quais já foi previamente exposta, porém falhe bastante para situações novas (para as quais a rede não foi treinada). Isto também aponta que a ESN não seja eficiente para ser utilizada para extrapolar dados.

A figura 25 apresenta o gráfico do maior erro obtido (redução de 1 unidade no tamanho do reservatório).

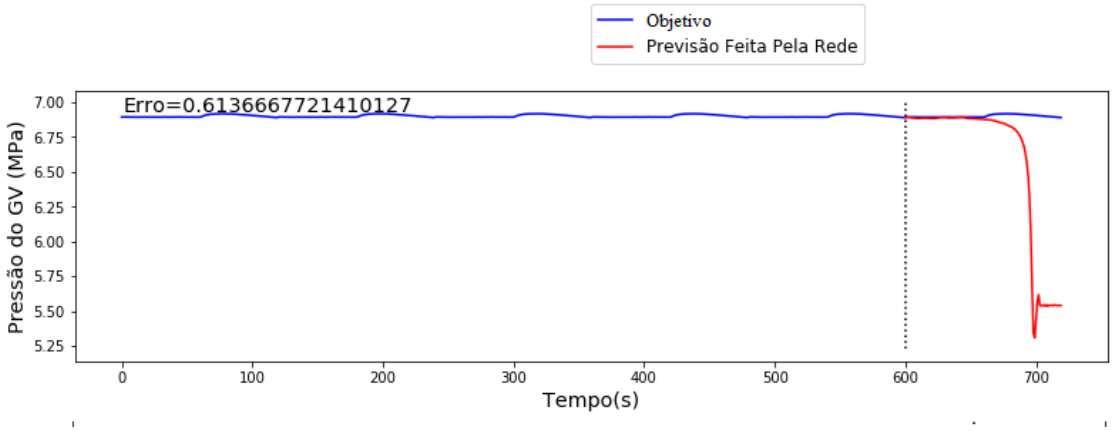


Figura 25 Gráfico com a rede que apresentou o maior erro ao mudar um parâmetro

Estas restrições não constituem, por si só, um defeito ou um problema da ESN. Apenas indica que para o trabalho dentro do seu domínio os resultados são bastante eficientes, porém se faz necessário certo cuidado com o domínio na hora de aplicar a rede. Além disto, este caráter especialista, faz com que a rede possa ser utilizada como um classificador, identificando rapidamente se um novo grupo de dados aleatórios recebidos se comporta como os dados de treinamento.

4.6 Fixação do Valor do Erro como Objetivo

Uma vez que o treinamento foi concluído e a otimização realizada, a fim de validar o resultado e testar sua repetitividade, recomeçou-se o processo de otimização das redes, só que desta vez o critério de parada para a rede é o valor do RMSE, configurado para ser igual ao melhor resultado dos testes anteriores, e não mais o número de iterações como foi feito anteriormente.

O teste foi feito para o acidente de LOCA, o PSO rodou por 11 horas, executando 2100 gerações. O valor do erro desejado (0,00092, como consta na tabela 19) não foi encontrado, ficando em 0,00346.

Este resultado não invalida o que foi obtido, pois denota uma característica deste problema: há muitos mínimos locais, e o PSO converge muito rapidamente para um deles (conforme destacado no item 4.2). Isto significa que, ao invés de deixar o PSO rodando indefinidamente com uma mesma configuração, é melhor realizar diversas buscas menores, para se aumentar as chances de localizar o mínimo global (ou então o melhor mínimo local).

A repetitividade do resultado pôde ser garantida reutilizando as mesmas *seeds* para a função *random* do *pyton*.

4.7 Limitação da Variação do Raio Espectral

Para o experimento principal (construção da rede preditiva) não foi aplicada a restrição para que o raio espectral fosse entre 0 e 1. Após a obtenção de resultados inciais, foi implementada tal restrição para título de comparação. Os resultados estão na tabela 12.

O caso testado foi o do LOCA com 300 dados de entrada, com as mesmas configurações do teste original e por 15 vezes, para que se tenha a mesma base de comparação.

Tabela 12 Resultados das otimizações com e sem restrição no raio

Caso	*Média (erro)*	*Média (raio)*	Menor raio	Maior raio	Desvio padrão (raio)	Desvio padrão (erro)
Sem restrição	0,0041	0,658	0,213	0,993	0,2139 (32,5%)	0,00061(14,9%)

Com restrição	0,0039	2,466	-62,92	125,77	44,8 (1816,61%)	0,001 (26,2%)

Como pode ser observado, a adição da restrição não melhorou os resultados obtidos, apesar das médias do erro terem ficado bem parecidas. Além disto, o caso sem restrição encontrou resultados otimizados espalhados por quase 80% do universo de busca (entre 0,213 e 0,993, num intervalo entre 0 e 1), o que reforça a ideia de que não haja uma correlação direta entre o valor do raio espectral e o erro, nem uma região onde os resultados sejam melhores.

Além dos resultados experimentais suportarem a hipótese da não obrigatoriedade desta restrição, há também o fato de que não há comprovação matemática que obrigue a implementação da restrição (Jaeger, 2001). Há casos de algumas otimizações sem restrição nas quais o valor do raio fica entre 0 e 1, porém não foi o melhor resultado.

O fato de espectro de valores de raios encontrados ser bem grande aponta mais uma vez para a necessidade de se otimizar a rede antes de utilizá-la.

4.8 Teste de Convergência de C1 e C2

Conforme mencionado em 4.3.2 e em 4.3.4, foi realizado um teste de convergência de C1 e C2, para identificar qual mais se adequa ao problema em estudo, fazendo com que convirja mais rápida e consistentemente para o menor erro. Este teste foi realizado variando C1 de 0 a 4,1 em intervalos de 0,1, enquanto C2 foi variado de 4,1 a 0, também em intervalos de 0,1. Os resultados deste teste estão na tabela 13 abaixo.

O caso testado foi o LOCA com 300 dados de entrada, 200 indivíduos no PSO e 35 iterações, todos com a mesma *seed* para ter a mesma base de comparação. Os dados estão apresentados truncados, e o menor valor foi marcado de verde (tabela 13).

O melhor resultado de conversão foi obtido com C1= 0,3 e C2 = 3,8. Como este estudo foi feito apenas ao final do experimento (uma vez que inicialmente os resultados vinham se mostrando satisfatórios), este não foi o valor aplicado às simulações principais. O fato do menor valor encontrado nesta seção ser menor que a do valor previamente encontrado sugere que ainda há bastantes melhorias a serem feitas no método de otimização, e que esta avaliação de convergência deve ser feita para todos os

casos ao se utilizar o PSO. Vale ressaltar também que a distribuição (figura 26) mostra que não há uma correspondência direta entre valores de C1 e C2 e o erro.

Tabela 13 Resultados do teste de convergência

C1	C2	Erro após 35 iterações (x10-3)	C1	C2	Erro após 35 iterações (x10-3)
0	4.1	5.8	2.1	2	4.6
0.1	4	5.2	2.2	1.9	3.1
0.2	3.9	4.8	2.3	1.8	4.1
0.3	3.8	1.4	2.4	1.7	6.4
0.4	3.7	4.3	2.5	1.6	4.2
0.5	3.6	3.8	2.6	1.5	6.4
0.6	3.5	4.5	2.7	1.4	4.4
0.7	3.4	6.1	2.8	1.3	1.9
0.8	3.3	4.6	2.9	1.2	6.1
0.9	3.2	3.8	3	1.1	4.6
1	3.1	3	3.1	1	3.4
1.1	3	2.1	3.2	0.9	1.6
1.2	2.9	4.4	3.3	0.8	4.8
1.3	2.8	5.6	3.4	0.7	4.3
1.4	2.7	5.5	3.5	0.6	3.5
1.5	2.6	5.4	3.6	0.5	5.8
1.6	2.5	5.4	3.7	0.4	5.2
1.7	2.4	5.8	3.8	0.3	4.5
1.8	2.3	4.4	3.9	0.2	5.4
1.9	2.2	2.9	4	0.1	2.3
2	2.1	6.2	4.1	0	5

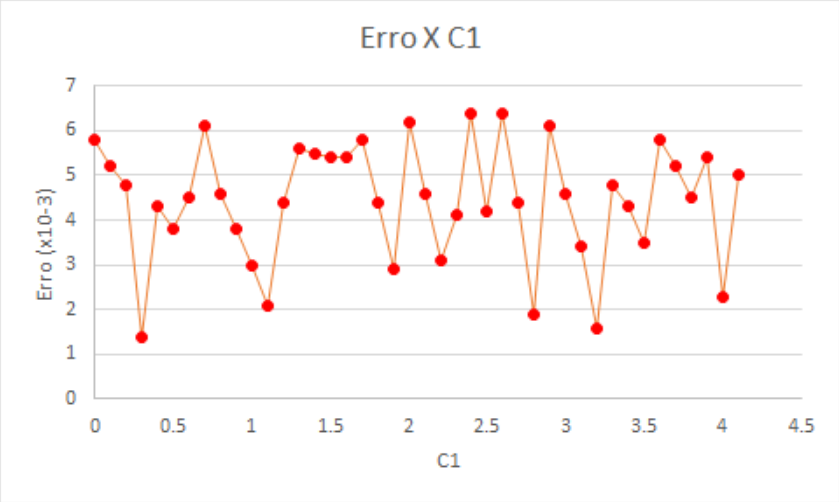


Figura 26 Distribuição dos erros versus *C1*

4.9 Fixação de Cinco Variáveis e Otimização da Sexta

Novamente, ao fim do experimento principal, a fim de validar o resultado e testar sua repetitividade, recomeçou-se o processo de otimização das redes para cada acidente, só que desta vez 5 valores foram fixados (com os melhores valores encontrados no experimento principal) e o sexto foi otimizado pelo PSO, para testar se o valor desta variável e o erro da rede encontrados foram os mesmos. O valor escolhido para ser variado foi o tamanho do reservatório, enquanto os outros se mantiveram fixos.

Neste teste, o valor da *sparsity* e do ruído foram zerados. Apesar de os resultados deste experimento serem diferentes do principal, este não foi comprometido, pois os resultados serão comparados com a otimização feita com as mesmas condições.

O acidente de estudo foi o Blackout com 120*5 dados de entrada, com as mesmas configurações dos demais, e cada otimização de PSO foi executada 50 vezes, com valores de *seed* diferente, para que se obtivesse uma boa base de comparação.

Dos cinquenta testes realizados, trinta e cinco encontraram como melhor solução o valor de 299; sete, valor de 125; quatro, o de 267; e um para cada um dos seguintes valores: 265, 179 e 275. Os valores dos erros (truncados) destes casos são, respectivamente, 0,0267, 0,271, 0,319, 0,36091, 0,36108 e 0,324.

Os resultados foram bem consistentes, pois em 70% das vezes o PSO foi capaz de encontrar o tamanho do reservatório que otimiza o conjunto dos 5 parâmetros fixos, e consequentemente o menor erro. Este resultado aponta novamente para e consistência e eficiência de se utilizar o PSO para otimizar a ESN, assim como reforça que a tese de que realizar mais tentativas com um menor número de iterações ao invés de uma única com mais iterações é a mais correta, pois apesar de na maioria das vezes o PSO encontrar o valor ótimo, 30% das vezes ele encontra um mínimo local (não ótimo) que não que consegue ser melhorado.

4.10 Remoção de Ruído

Foi realizado também otimizações sem a adição do ruído de 0,001. Para isto, cada dado de entrada foi somado a um valor aleatório no intervalo de [-0,001 ; 0,001]. Os melhores resultados obtidos estão na tabela 14, e os resultados completos estão no Anexo I. Estão apresentados também a comparação com os melhores resultados com a adição de erro, e as células apresentadas em verde denotam o melhor valor entre os 2 casos.

Tabela 14 Resultados com e sem a adição do ruído

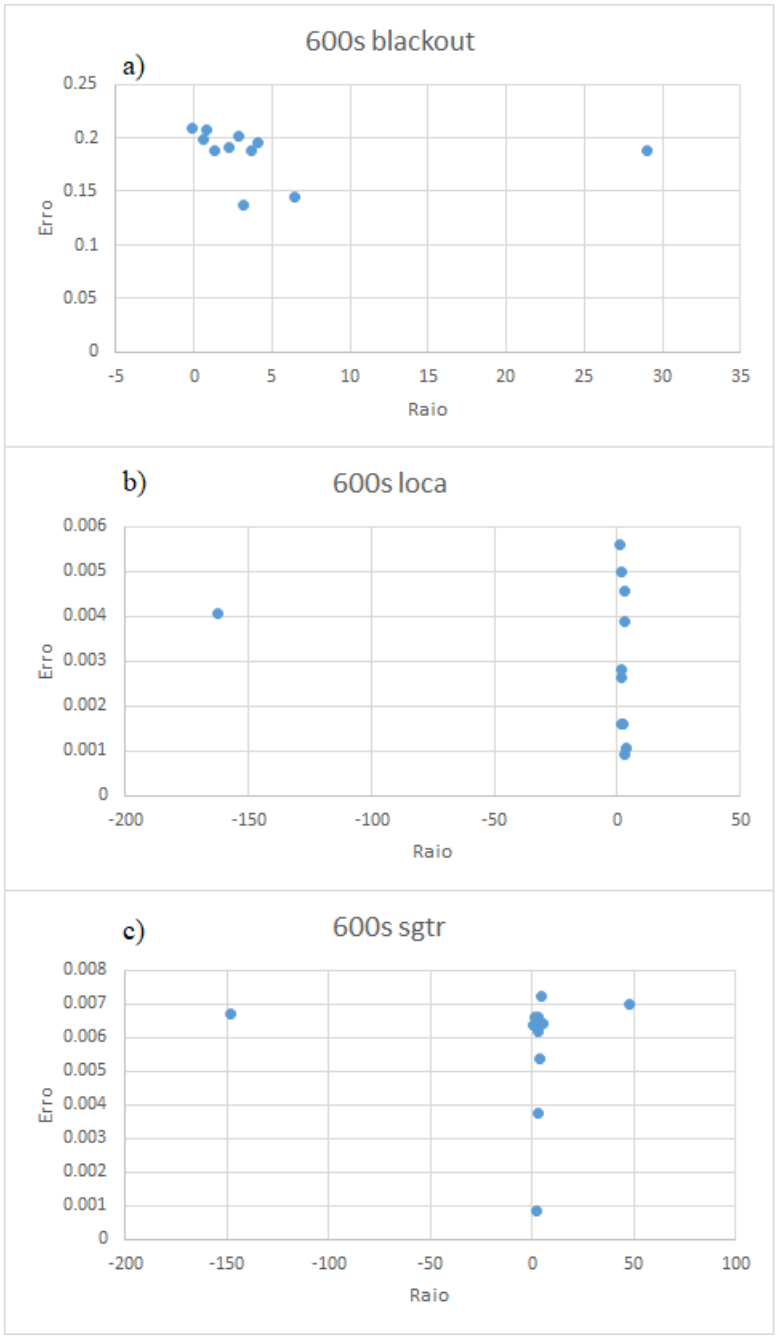
Acidente	Quantidade de dados		
Blackout	300	600	120*5
Erro com noise	0.0804	0.1365	0.2362
Erro sem noise	0.0116	0.003	0.1891
loca	300	600	120*5
Erro com noise	0.0028	0.000929	0.00265
Erro sem noise	0.0016	0.0008576	0.00294
sgtr	300	600	120*5
Erro com noise	0.00213	0.000842	0.00889
Erro sem noise	0.00439	0.00211	0.005077

Pode-se perceber que a influência da adição do erro no resultado final depende do problema a ser tratado. Isto pode ser explicado (Jaeger, 2001) pelo fato de que, quanto mais lineares e menos dispersos forem os dados, menos graus de liberdade são retirados da função na hora da formulação da rede. Isto pode ser uma característica boa ou ruim, dependendo da configuração do problema. Por exemplo, problemas com alto grau de recorrência ou ciclos tendem a se beneficiar de dados de entrada mais exatos (sem o erro), enquanto problemas altamente não-lineares e sem recorrência tendem a se beneficiar de dados mais dispersos (com erro), pois estes dados obrigam a rede a ser mais abrangente.

4.11 Correlação entre o Raio Espectral e o Erro

Como é encontrado frequentemente na literatura que um dos fatores que mais influem no desempenho da rede é o raio espectral e pode ser constatado ao longo do estudo, foi realizado o mesmo tipo estudo de variação que foi feito na seção 4.3. As tabelas 15 e 16, e a figura 27, mostram os resultados obtidos.

Vale lembrar que os valores do raio espectral foram inicializados centrados em 3, o que explica o fato de, em alguns acidentes, haver uma concentração de resultados em torno deste valor.


600s blackout
a)
0.25
0.2
0.15
0.1
0.05
0
Erro
-5
0
5
10
15
20
25
30
35
Raio
b)
600s loca
0.006
0.005
0.004
0.003
0.002
0.001
0
Erro
-200
-150
-100
-50
0
50
Raio
c)
600s sgtr
0.008
0.007
0.006
0.005
0.004
0.003
0.002
0.001
0
Erro
-200
-150
-100
-50
0
50
100
Raio

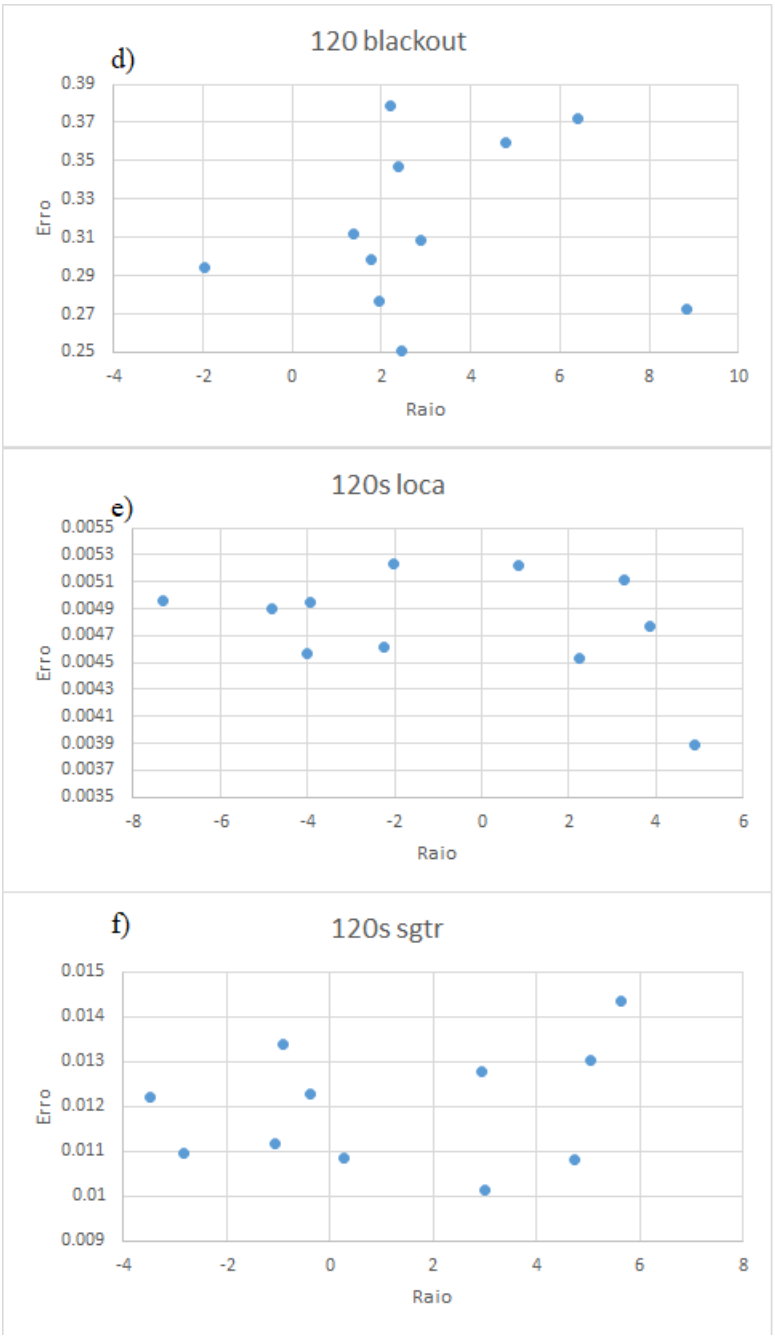


*Figura 27 Comparação entre Erro e Raio Espectral, para cada acidente: a) Blackout com 60*10s ; b)LOCA com 60*10s ; c) SGTR com 60*10s; d) Blackout com 120*5s ; e)LOCA com 120*5s ; f) SGTR com 120*5s*

Tabela 15 Valores da variação do erro para cada experimento

Erro					
Acidente	**Média**	**Desvio Padrão**	**% do desvio padrão em relação ao erro**	**Diferença entre o maior e o menor**	**Diferença entre o maior e o menor (%)**
Blackout 60*10	0,186	0,022	12,2	0,072	52,7
LOCA 60*10	0,003	0,0015	51,07	0,004	500,3
SGTR 60*10	0,005	0,0017	31,2	0,006	755,9
Blackout 120*5	0,315	0,0409	12,9	0,127	50,8
LOCA 120*5	0,004	0,0003	7,7	0,001	34,5
SGTR 120*5	0,0119	0,0012	10,4	0,004	41,6

Tabela 16 Valores da variação do raio espectral para cada experimento

Raio Espectral					
Acidente	**Média**	**Desvio Padrão**	**% do desvio padrão em relação ao erro**	**Diferença entre o maior e o menor**	**Diferença entre o maior e o menor (%)**
Blackout 60*10	4,92	7,81	158,6	29,07	33246,5
LOCA 60*10	-12,7	47,3	370,6	165,79	102,1
SGTR 60*10	-6,75	46,54	688,6	196,01	132,3
Blackout 120*5	3,005	2,69	89,6	10,79	547,8
LOCA 120*5	-0,84	3,87	-457,7	12,2	166,6

SGTR 120*5	1,17	3,08	261,8	9,14	261,3

Como pode ser observado, os valores dos resultados otimizados dos raios são bastante espalhados (a diferença entre o maior e o menor é grande, assim como o desvio padrão é maior do que a média, em alguns casos por várias ordens de grandeza). Pode-se perceber também que a média dos melhores valores varia bastante de acidente para acidente, indicando que não há uma regra geral que possa ser aplicado a todos os casos, e reforçando a ideia de que é necessária uma otimização prévia dos parâmetros da ESN. Isto também vai ao encontro com o que foi constatado na seção 4.7.

4.12 Resultados Principais

4.12.1 Treinamento

Depois de otimizar a ESN para cada caso, os melhores parâmetros encontrados estão na tabela 17, assim como o desvio padrão por meio dos diferentes ciclos de otimização (10 no total), assim como qual fração dos valores médios esse desvio padrão representa (em percentagem). O RMSE correspondente está na tabela 18, e a figura 28 mostra os gráficos comparando a previsão feita pelo ESN e a distribuição real, para cada acidente.

Tabela 17 Valores dos parâmetros da ESN otimizada para cada acidente

	LOCA		SGTR		Blackout	
Parâmetro	**Valor**	**Desvio Padrão (%)**	**Valor**	**Desvio Padrão (%)**	**Valor**	**Desvio Padrão (%)**
Raio Espectral	2,78	47,31 (370%)	2,46	46,54 (688%)	3,13	7,81 (158%)
Tamanho do Reservatório	89	70 (37%)	176	57 (33%)	139	56 (39%)
Sparsity	0,39	0,25 (73%)	0,92	0,33 (57%)	0,69	0,28 (57%)
Input Shift	8,13	3,7 (178%)	2,21	3,92 (64%)	2,04	45,18 (316%)

Input Scaling	0,02	0,058 (524%)	-0,0105	0,07 (268%)	0,0408	1,11 (933%)
Teacher Shift	-23,97	6,92 (317%)	-0,082	1,51 (4691%)	-0,83	4,14 (329%)
Teacher Shift	171,41	50 (348%)	0,726	12,89 (182%)	0,84	63,8 (2059%)

Tabela 18 Valores dos RMSE para cada acidente

Acidente	RMSE
Blackout	0,136554015704435
LOCA	0,000929597985325
SGTR	0,000842352041199
Operação Normal	0,000116085919158

O primeiro resultado que ressalta é que o RMSE para a simulação de operação normal é uma ordem de grandeza menor que os acidentes, o que pode ser explicado pelo fato de que durante a operação normal, o comportamento dessa variável é linear. Isso torna mais fácil para o sistema identificar quando o reator não está mais operando sob condições normais, e tornando a detecção de condição de acidente pelo sistema preditivo bastante rápida.

Outro ponto a notar é que o RMSE para o acidente de Blackout é algumas ordens de grandeza maior do que os outros dois acidentes. Isso é explicado pelo fato de que o comportamento da variável para esse acidente é muito diferente dos demais, e a configuração atual da ESN favorece o comportamento dos outros acidentes. Uma maneira de melhorar o RMSE para este caso seria alterar a função de ativação, utilizando uma para cada acidente.

Além disso, ao analisar os parâmetros na tabela 17, pode-se notar que o intervalo de valores para cada parâmetro é muito amplo (conforme indicado pelo desvio padrão). Isso sugere que não há uma regra geral que possa ser aplicada à seleção de parâmetros que levaria a um bom resultado sem otimização.

Um resultado que realmente se destacou foi a *sparsity*. Jaeger (Jaeger, 2001) afirma que é desejável que um ESN tenha altos valores de dispersão para que ele tenha um bom desempenho, e como pode ser visto na tabela 17, há valores bem baixos para a

sparsity (0,39, em oposição ao mencionado por Jaeger de 0,95). Este resultado ainda não tem uma explicação formal, e mais estudos são necessários sobre este assunto.

A figura 28 mostra os gráficos comparando a previsão feita pelo ESN e a distribuição real, para cada acidente. Nela, pode-se perceber que a previsão para o acidente blackout ainda requer refinamentos, enquanto a previsão do LOCA e do SGTR estão bastante próximas ao objetivo. Isto pode ser explicado principalmente pelo fato de a função de ativação ser a mesma para todos os acidentes, enquanto seus comportamentos são distintos entre si. Além disto, em todos os casos, a primeira parte da curva apresenta uma precisão maior.

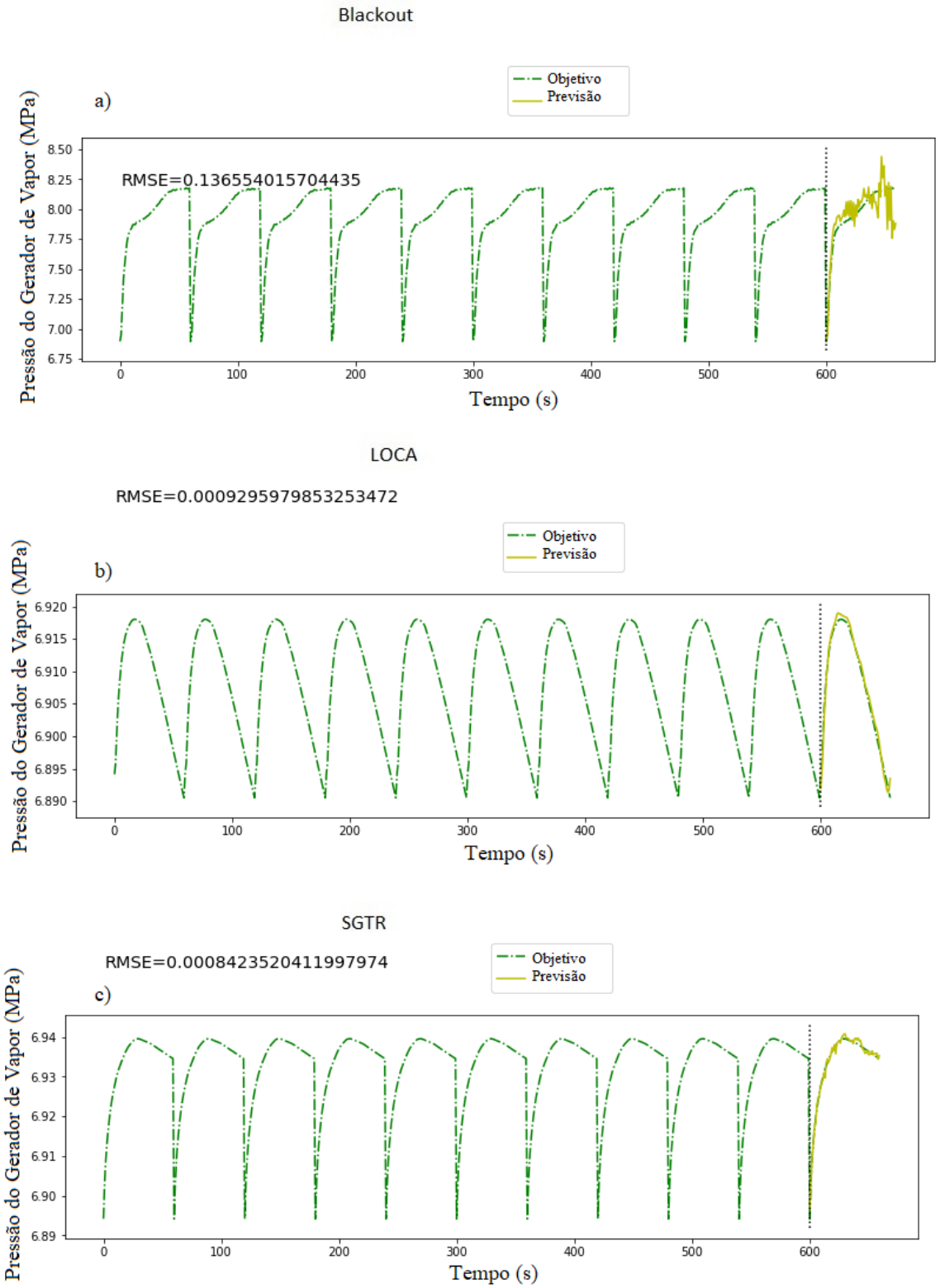


Figura 28 Previsões da ESN versus *objetivo, para cada acidente: a) Blackout ; b)LOCA ; c) SGTR*

4.12.2 Teste

Uma vez que as redes foram otimizadas, as simulações com recebimento e processamento de dados em tempo real foram realizadas.

Cada lista contendo as entradas recebidas pelo cliente foi usada para construir quatro ESN (um para cada condição operacional), sempre com os mesmos parâmetros da tabela 17, e os RMSE dessas ESN foram comparados com os da tabela 18. Os menores RMSE da ESN recém-treinada indicam a condição operacional atual (se é normal ou acidente, e qual acidente está ocorrendo).

A tabela 19 mostra quanto tempo demorou o sistema preditivo a identificar qual condição operacional estava ocorrendo.

Tabela 19 Tempo para detecção de cada acidente

Acidente	Tempo até a detecção (s)
Blackout	50
LOCA	41
SGTR	32

Este tempo aparentemente longo para identificar qual acidente está ocorrendo pode ser explicado pelo fato de que o RMSE para operação normal é muito menor que os outros. Há espaço para melhorias no RMSE dos acidentes, com um estudo mais aprofundado do comportamento das ESN para cada caso, junto com maiores conjuntos de dados e melhor otimização (com a ajuda da programação da GPU e paralelização, por exemplo). Quando o RMSE de todos os casos estiver na mesma magnitude, esse tempo de detecção será reduzido.

Cabe ressaltar que atualmente não há nenhum sistema implementado em sistemas de controles de reatores nucleares que execute esta função de detectar em tempo real qual acidente está ocorrendo. O diagnóstico de qual acidente ocorreu é realizado após a normalização da situação, utilizando o histórico de alarmes e realizando uma *fault tree analysis* (Larsen, 1974).

5 – Conclusões

As primeiras conclusões do estudo foram sobre os parâmetros do ESN. Ficou provado que não há necessidade de uma restrição no raio espectral. Um breve argumento para isso é que, durante o "apertar" das não-linearidades pela função de ativação, uma entrada suficientemente forte é capaz de "extrair" ativações autônomas do reservatório (Venayagamoorthy & Shishir, 2009). Não há, tampouco, correlação direta entre o aumento ou a diminuição de um dado parâmetro e seu efeito sobre o desempenho (RMSE), confirmando as indicações encontradas na literatura (Chouikhi *et al*., 2017).

Além disso as características elitistas da ESN a tornam um excelente classificador binário. Como ele não funciona bem com dados extrapolados (dados de fora do intervalo com o qual foi treinado), ao receber dados que não têm o mesmo comportamento que os dados usados durante o treinamento, o RMSE da previsão da ESN neste caso será muito alto, indicando instantaneamente se os novos dados e os dados de treinamento são semelhantes ou não.

A conclusão mais importante sobre o comportamento das ESN é a necessidade de se realizar uma otimização em seus parâmetros antes de qualquer treinamento. Como não há correlação entre um parâmetro e sua influência no RMSE final, uma ESN treinada com parâmetros aleatórios ou escolhidos ao acaso tende a apresentar previsões muito erráticas.

Finalmente, o principal objetivo do estudo foi alcançado. Foi mostrado que uma ESN otimizada é adequada para ser usada como um sistema preditivo para indicar uma condição operacional de um núcleo de reator em tempo real.

5.1 Trabalhos Futuros

Como mencionado no capítulo 1, ainda não há nenhum estudo ou aplicação atual deste tipo, portanto este trabalho envolveu o desenvolvimento, prova e validação de conceitos a serem aplicados para que se tenha uma rede deste tipo. Ainda há, portanto, algumas frentes a serem desenvolvidas, dentre elas:

- Diminuir o RMSE, com estudos como: melhorar a forma da rede para cada caso (como visto na seção 4.2); treinar a rede para outros dados de operação; treinar a rede com mais acidentes e variáveis (para conferir mais realismo à simulação da rede); utilizar outros algoritmos de inteligência artificial (formiga, cuco, algoritmos quânticos, entre outros);
- Diminuir o tempo de resposta da rede, com estudos como: implementar cálculos paralelizados via GPU; implementar atualização das configurações da rede em tempo real, conforme novos dados vão sendo recebidos; realizar um estudo de ampliação e escolha das variáveis a serem medidas (além das de segurança), a fim de procurar possíveis relações ainda não documentadas entre fatores externos e o TRIP; verificar se há a possibilidade/necessidade de conferir pesos às variáveis, para que se chegue a um grupo mínimo de variáveis a ser monitoradas;
- Explorar a capacidade de se utilizar a ESN como um classificador, devido à sua natureza altamente especialista.

Referências Bibliográficas

Alvarenga, M. A. (Maio de 1997). Diagnóstico do Desligamaneto de um Reator Nuclear através de Técnicas Avançadas de Inteligência Artificial - tese. Rio de Janeiro: COPPE/UFRJ.

Asea Brown Boveri. (21 de 11 de 2018). *ABB*. Fonte: https://new.abb.com/control-systems/features/advanced-process-control-and-analytics-in-industrial-automation

Augusto, J. P., Nicolau, A. d., & Schirru, R. (Agosto de 2015). PSO with Dynamic Topology and Random Keys method applied to Nuclear Reactor Reload. *Progress in Nuclear Energy*, pp. 191-196.

Bauduin, M., Smerieri, A., Massar, S., & Horlin, F. (2015). Equalization of the Non-Linear Satellite Communication Channel with an Echo State Network. *IEEE 81st Vehicular Technology Conference.* Glasgow: IEEE.

Bravo, C. O., & Normey-Rico, J. E. (Outubro-Dezembro de 2009). Controle de plantas não lineares utilizando control preditivo linear baseado em modelos locais. *Sba: Controle & Automação*, pp. 465-481.

Bryce, S. (27 de Novembro de 2018). *TekHabit*. Fonte: https://tekhabit.wordpress.com/2013/04/15/artificial-neural-network/

Buonomano, D. V., & Merzenich, M. M. (Março de 1995). Temporal information trans-formed into a spatial code by a neural network with real-istic properties. *Science*, pp. 1028-1030.

Bussab, W. O., & Morettin, P. A. (2017). *Estatística Básica.* Saraiva.

Cao, S., Xu, W., & Hu, X. (2015). Dual adaptive control of nonlinear stochastic systems based on echo state network. *The 27th Chinese Control and Decision Conference.* Qingdao: IEEE.

Carlisle, A., & Dozier, G. (2010). An off-the-shelf PSO. *Proceeding of Workshop on Particle Swarm Optimization* (pp. 1-6). Indianapolis: IUPUI.

Carvalho, P., Santos, I., Santana, M., Soares, A., & al., e. (2014). Instrumentation & Control Systems for the Brazilian Multipurpose Reactor - RMB. *IGORR 2014 / IAEA Technical Meeting.* Bariloche: IAEA.

Catto, J., Linkens, D., Abbod, M., Chen, M., & al., e. (2003). Artificial Intelligence in Predicting Bladder Cancer Outcome A Comparison of Neuro-Fuzzy Modeling and Artificial Neural Networks. *Clinical cancer research : an official journal of the American Association for Cancer Research*, pp. 4172-4177.

Chouikhi, N., Ammar, B., Rokbani, N., & Alilmi, A. M. (2017). PSO-based analysis of Echo State Network parameters for time series forecasting. *Applied Soft Computing*, pp. 211-225.

Clerc, M., & Kennedy, J. (Fevereiro de 2002). The Particle Swarm—Explosion, Stability, and Convergence in a Multidimensional Complex Space. *IEEE transactions on Evolutionary Computation, 2002*.

Dorigo, M. (1992). Optimization, Learning and Natural Algorithms (tese). Milão: Politecnico di Milano.

Eberhart, R., & Shi, Y. (2000). Comparing inertia weights and constriction factors in particle swarm optimization. *Congress on Evolutionary Computation.* La Jolla: IEEE.

Ghanbari, A., Kazemi, S., Mehmanpazir, F., & Nakhostin, M. M. (Fevereiro de 2013). A Cooperative Ant Colony Optimization-Genetic Algorithm approach for construction of energy demand forecasting knowledge-based expert systems. *Knowledge-Based Systems*, pp. 194-206.

Goldberg, D. E., & Holland, J. H. (Outubro de 1988). Genetic Algorithms and Machine Learning. *Machine Learning*, pp. 95-99.

Grossberg, S. (22 de Novembro de 2018). *ScholarPedia.* Fonte: http://www.scholarpedia.org/article/Recurrent_neural_networks

Guazzelli, A. (12 de 03 de 2018). *IBM*. Fonte: https://www.ibm.com/developerworks/br/industry/library/ba-predictive-analytics1/index.html

Haykin, O. S. (1998). *Neural Networks: A Comprehensive Foundation, 2nd Edition.* Ontario: Pearson.

Hertz, J. A., Krogh, A. S., & Palmer, R. G. (1991). *Introduction To The Theory Of Neural Computation.* Santa Fe: Westview Press.

International Atomic Energy Agency. (12 de Março de 2018). *IAEA*. Fonte: https://www.iaea.org/topics/research-reactors

Jaeger, H. (2001). The" echo state" approach to analysing and training recurrent neural networks-with an erratum note. *GMD Technical Report, 148*, pp. 1-44.

Jaeger, H. (2002). *Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the echo state network approach.* Bonn: German National Research Center for Information Technology.

James, A. P. (Janeiro de 2014). Threshnomics: An Introduction to Threshold Logic in Algorithms and Circuits. *Journal of Computer Science & Systems Biology*, pp. 235-240.

Jiang, F., Berry, H., & Schoenauer, M. (2008). Supervised and Evolutionary Learning of Echo State Networks. Em G. Rudolph, T. Jansen, N. Beume, S. Lucas, & C. Poloni, *Lecture Notes in Computer Science* (pp. 215-224). Berlin: Springer.

Kachitvichyanukul, V. (Setembro de 2012). Comparison of Three Evolutionary Algorithms: GA, PSO, and DE. *Industrial Engineering & Management Systems*, pp. 215-223.

Kennedy, J., & Eberhart, R. (1995). Particle Swarm Optimization. *IEEE International Conference on Neural Networks* (pp. 1941-1948). Perth: IEEE.

Larsen, W. F. (1974). *Fault Tree Analysis.* Dover: U.S. Army.

Leite, V. C. (Abril de 2017). Otimização Por Enxame De Partículas Aplicado À Estrutura Mecânica Da Mola Da Grade Espaçadora Do Elemento Combustível De Um Reator Nuclear. Rio de Janeiro: COPPE/UFRJ.

Liu, D., Wang, J., & Wang, H. (Junho de 2015). Short-term wind speed forecasting based on spectral clustering and optimised echo state networks. *Renewable Energy*, pp. 599-608.

Lukoševičius, M. (2012). A Practical Guide to Applying Echo State Networks. Em G. Montavon, G. Orr, & K. Müller, *Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science* (pp. 659-689). Berlin: Springer.

Maass, W., Natschläger, T., & Markram, H. (Novembro de 2002). Real-Time Computing Without Sta-ble States: A New Framework for Neural Computation Based on Perturbations. *Neural Computing*, pp. 2531-2560.

Martins, G. P. (2015). Modelo Híbrido Baseado Em Redes Neurais Echo State Network Otimizadas Por Algoritmos Evolucionários Para Identificação Automática De Sistemas Dinâmicos E A Sua Aplicação À Identificação Do Nível Do Gerador De Vapor De Uma Usina Nuclear PWR (dissertação). Rio de Janeiro: Pontifícia Universidade Católica.

Martins, G., Vellasco, M., & Schirru, R. (2015). Identificação em malha fechada do nível de água de um gerador de vapor de usina nuclear utili-zando echo state network sintonizada por algoritmos gen-éticos. *Simpósio Brasileiro de Automação Inteligente* (p. 1). Natal: Sociedade Brasileira de Automação Inteligente.

McCulloch, W. S., & Pitts, W. (Dezembro de 1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, pp. 115-133.

Meneses, A. A., Macae, F. E., Machado, M. D., Medeiros, J. A., & Schirru, R. (2007). Particle swarm optimization with random keys applied to the nuclear reactor reload problem. *INAC 2007 International nuclear atlantic conference* (p. 7). Santos: INAC.

Minsky, M., & Papert, S. (1969). *Perceptrons. An Introduction to Computational Geometry.* Cambridge: MIT Press.

Mól, A. C. (2002). Um sistema de identificação de transientes com inclusão de ruídos e indicação de eventos desconhecidos (tese). Rio de Janeiro: COPPE/UFRJ.

Nicolau, A. d., & Schirru, R. (Janeiro de 2011). Study of the Quantum Evolutionary Algorithm Parameters Applied to Transient Identifica-tion. *Progress in Nuclear Energy*, pp. 86-91.

Nyce, C. (2017). *Predictive Analytics White Paper.* Malvern: AICPCU.

Oludare, O., Stephen, O., Ayodele, O., & Temitayo, F. (2014). An Optimized Feature Selection Technique For Email Classification. *International journal of scientific & technology research*, p. 7.

Prater, A. A. (2017). Comparison of echo state network output layer classification methods on noisy data. *International Joint Conference on Neural Networks.* Anchorage: IEEE.

Rosenblatt, F. (Novembro de 1958). The perceptron: A probabilistic model for information storage and organiza-tion in the brain. *Psychological Review*, pp. 386-408.

Rueda, C. V. (Abril de 2014). Ferramenta de Previsão de Séries Temporais Baseada em Echo State Networks Otimizadas por Algoritmos Genéticos e Particle Swarm Optimization (dissertação). Rio de Janeiro: Pontifícia Universidade Católica.

Sak, H., Senior, A., & Beaufays, F. (2014). *Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition.* Google.

Schrauwen, B., Verstraeten, D., & Campenhout, J. V. (2007). An overview of reservoir computing: theory, applications and implementations . *Proceedings of the 15th European Symposium on Artificial Neural Networks* (pp. 471-482). Bruges: Department of Electronics and information systems.

Serrão, B. P. (Fevereiro de 2018). Modelo De Dispersão Atmosférica Para Acidente Nuclear Na Região Da Base Naval De Itaguaí Com Otimização Do Campo De Vento Utilizando Inteligência De Enxames. Rio de Janeiro: COPPE/UFRJ.

Shi, Y., & Eberhart, R. (1998). A modified particle swarm optimizer. *IEEE World Congress on Computational Intelligence.* Anchorage: IEEE.

Shin, K.-S., & Lee, Y.-J. (Outubro de 2002). A genetic algorithm application in bankruptcy prediction modeling. *Expert Systems with Applications*, pp. 321-328.

Stanley, K. (25 de Outubro de 2006). *Slide Share.* Fonte: https://slideplayer.com/slide/4642416/

Tavassoli, B. (09 de Junho de 2018). *On Model Predictive Control of Hybrid Systems.* Fonte: ArXiv: https://arxiv.org/pdf/1806.03459.pdf

Trelea, I. C. (Março de 2003). The particle swarm optimization algorithm: convergence analysis and parameter selection. *Information Processing Letters*, pp. 317-325.

Tutschku, K. (1995). *Recurrent multilayer perceptrons for identification and control: The road to applications.* Würzburg: University of Würzburg.

Venayagamoorthy, G. K., & Shishir, B. (Maio de 2009). Effects of spectral radius and settling time in the performance of echo state networks. *Neural networks: the official journal of the Inter-national Neural Network Society*, pp. 861-863.

Waintraub, M., Schirru, R., & Pereira, C. M. (Agosto-Setembro de 2009). Multiprocessor modeling of parallel Particle Swarm Optimization applied to nuclear engineering problems. *Progress in Nuclear Energy*, pp. 680-688.

Werbos, P. J. (1975). Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences (Tese de Doutorado). Cambridge: Harvard University.

Willmott, C. J., & Matsuura, K. (Dezembro de 2005). Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climatic Reserach*, pp. 79-82.

Zell, A. (1997). *Simulation Neuronaler Netze.* Tübingen: De Gruyter Oldenbourg.

Zitzler, E., Deb, K., & Thiele, L. (Junho de 2000). Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, pp. 173-195.

Apêndice I – Dados dos Experimentos

COM NOISE								
erro								
300s_blackout	300s_loca	300s_sgtr	600.s_blackout	600.s_loca	600.s_sgtr	GV_120blackout	GV_120loca	GV_120sgtr
0.16603201	0.003187925	0.006397974	0.187859174	0.001077213	0.006169326	0.251104822	0.004770854	0.010848252
0.209622728	0.002953384	0.006138284	0.144385531	0.002634955	0.006593169	0.308232077	0.005118671	0.012275964
0.210207661	0.005434067	0.00683356	0.208107732	0.003869814	0.000842352	0.378877518	0.005217907	0.010126226
0.137572508	0.00358044	0.006357436	0.199179785	0.000929598	0.006359341	0.298582577	0.004565525	0.010814845
0.182503433	0.004689574	0.006630358	0.190700274	0.001596463	0.006713793	0.276888354	0.004619903	0.013384582
0.205447298	0.002843797	0.006168084	0.195797476	0.004566039	0.006416718	0.359860193	0.004943546	0.012777124
0.212812238	0.005677044	0.006999714	0.188796704	0.004970541	0.006582332	0.311838784	0.003889508	0.011152115
0.225548509	0.005196686	0.00661796	0.208587136	0.002806023	0.005371584	0.346624553	0.005234518	0.013004452
0.208135973	0.002950823	0.006769775	0.201021029	0.005581057	0.003731865	0.272773496	0.004954168	0.010942914
0.219076413	0.003139653	0.006415734	0.136554016	0.004048441	0.006959085	0.294156542	0.004536405	0.012192627
0.094143777	0.003970462	0.006803452	0.188726244	0.001598756	0.007209842	0.372024196	0.004897294	0.014340279
sparsity								
300s_blackout	300s_loca	300s_sgtr	600.s_blackout	600.s_loca	600.s_sgtr	GV_120blackout	GV_120loca	GV_120sgtr
0.264056318	0.692274974	0.858001212	0.45348258	0.178973268	0.17855198	0.730534299	0.827997843	0.927297922
0.848351664	0.782018331	0.774268566	0.104111277	0.566982264	0.629031306	0.234972096	0.248209078	0.553566974
0.203179049	0.157963533	0.195725689	0.657569951	0.236907614	0.922741364	0.869374243	0.077248527	0.80360996
0.831746173	0.221557017	0.900010659	0.368865629	0.390261217	0.921428339	0.778378875	0.084525578	0.277967342
0.875845933	0.77047545	0.571373606	0.05027222	0.646759393	0.936822317	0.704448513	0.718977145	0.113263799
0.03893536	0.162540689	0.050265601	0.816417452	0.084766255	0.016784933	0.040119095	0.000916516	0.230474581
0.009066814	0.89410447	0.827508517	0.142616637	0.231554141	0.872150652	0.449489014	0.671528587	0.017138503
0.641637761	0.077150273	0.097599261	0.892934113	0.166098087	0.422500184	0.021169131	0.72940417	0.062158678
0.26050254	0.004296948	0.709482937	0.805645466	0.227052964	0.636457604	0.944346088	0.697746519	0.712048799
0.944005969	0.90993736	0.495872537	0.689700691	0.942570973	0.088643483	0.805151947	0.177677723	0.008586713
0.692014588	0.252453914	0.896783397	0.54153561	0.1270859	0.823341957	0.275520177	0.14724945	0.102371891
reservatório								
300s_blackout	300s_loca	300s_sgtr	600.s_blackout	600.s_loca	600.s_sgtr	GV_120blackout	GV_120loca	GV_120sgtr
59	110	36	223	243	142	244	102	255
136	97	31	92	209	183	184	263	150
129	82	47	120	235	176	235	300	203
34	128	123	120	88	120	154	174	160
143	135	56	286	254	84	242	167	295
60	55	87	101	148	134	140	75	277
38	33	133	120	158	120	97	272	270
68	41	34	120	259	261	239	285	163
107	114	35	120	122	252	231	240	149
53	44	84	139	60	168	119	194	241
49	120	45	131	260	252	271	89	94
raio								
300s_blackout	300s_loca	300s_sgtr	600.s_blackout	600.s_loca	600.s_sgtr	GV_120blackout	GV_120loca	GV_120sgtr
9.98412534	3.171816025	14.28057218	3.670728503	3.431216397	2.806032115	2.445313386	3.85323833	0.277418258
2.464562951	1.735972506	-4.609614025	6.436240714	1.80086661	2.695526368	2.888326834	3.252576716	-0.379559908
-2.042452135	4.89969882	3.43102438	0.787089503	2.704431346	2.462571428	2.213215345	0.830115045	3.006986834
62.26878929	2.393090121	3.0588825	0.582539631	2.784166853	0.526779125	1.771623906	-4.029174451	4.733728151
16.11854384	0.634638002	3.961119395	2.274516471	1.742164325	-148.1587701	1.946171391	-2.236226678	-0.918010571
3.090408464	-2.766671952	2.376580883	4.121356063	3.208349105	5.063631642	4.774351931	-3.947842878	2.936755342
3.357435151	-47.95216697	-83.61819788	1.34934869	1.628318395	1.443202442	1.377605214	4.882470357	-1.06809252
5.107700873	125.7722046	3.719058904	-0.087438541	1.70888426	3.34053124	2.389057364	-2.031074508	5.053302623
3.490146932	4.093882669	5.699003466	2.915001634	0.872039243	3.012611402	8.827284191	-7.326761924	-2.829168868
2.911305456	-62.9286882	-4.717654631	3.138387801	-162.3593782	47.8582327	-1.970980884	2.255785413	-3.499164748
3.444255584	-1.923343727	3.253494924	28.98287506	2.064887614	4.600659046	6.393692694	-4.806807691	5.644287303
			7.812754532	47.31154081	46.54373551			
10.01771107	2.466402901	-4.833248173	4.924604139	-12.76491401	-6.758999324	3.005060125	-0.845791115	1.178043809
62.26878929	125.7722046	14.28057218	28.98287506	3.431216397	47.8582327	8.827284191	4.882470357	5.644287303
-2.042452135	-62.9286882	-83.61819788	-0.087438541	-162.3593782	-148.1587701	-1.970980884	-7.326761924	-3.499164748

input shift								
300s_blackout	300s_loca	300s_sgtr	600.s_blackout	600.s_loca	600.s_sgtr	GV_120blackout	GV_120loca	GV_120sgtr
-79.14291516	2.748342883	2.377421898	-7.406473977	6.950702739	-1.463725051	2.99432853	0.830650506	13.94849008
1.280059371	0.748258223	2.335734365	-3.043135132	0.149302916	3.577528242	5.636689144	8.222607944	-12.91312373
0.941998447	-11.51873728	1.768693682	-0.938608562	3.065993102	2.213629414	2.695632604	2.849126525	4.734452161
-7.115294177	2.652272042	4.030120408	152.9748621	8.137790602	-2.553928671	0.79659251	-32.57675039	12.32728932
1.684817642	1.201257708	-6.07328642	2.136151923	1.081304468	-1.383892982	1.203277711	-2.770226057	0.506392913
1.885006957	3.100223119	1.680910739	14.6858723	-7.056758219	10.55098739	1.973716075	-19.28336016	20.12829829
-1.931610848	14.28790185	-4.997599354	-27.60543839	1.888671256	-1.503136312	-0.111397479	34.77240915	1.211726697
0.97028124	-21.59563541	1.847615374	1.9810684	0.869798931	-2.153846907	1.14676165	-2.849224376	-123.9973924
6.982786947	16.01512297	-1.286397542	11.26868925	2.482660348	-4.433525834	-9.974204145	-13.04168593	0.754158945
0.507995114	10.84273278	2.078657198	2.049190921	3.251241202	-0.445073971	-0.455260694	1.419334525	-5.065050352
-0.581755036	0.699772142	1.406267554	10.89961966	2.094015975	-1.736951434	-17.856839	5.743046397	-6.542231512
input scaling								
300s_blackout	300s_loca	300s_sgtr	600.s_blackout	600.s_loca	600.s_sgtr	GV_120blackout	GV_120loca	GV_120sgtr
0.001076823	-0.142401041	0.003450925	0.028662802	0.099721681	0.021736604	-0.030582406	-0.081320004	-0.202297788
-0.019777981	0.006903095	0.024361565	2.767766142	0.006279115	0.019572185	0.079372164	0.345037222	-0.075975442
0.094860831	-0.01052831	0.007776784	-0.109620054	-0.006038675	-0.010556672	0.016532768	0.027622567	-3.041024059
0.006761832	0.00338612	0.009465106	-2.297173319	0.024871151	0.031607815	0.03753599	0.6786461	0.003726205
0.000950572	0.04041448	0.625066056	0.946008967	0.012921426	-0.241789627	-0.006610044	-0.631150713	-0.031227208
0.003830859	0.075585311	0.001533305	-0.058002702	-0.005661141	-0.017875839	-0.003963023	-1.040785355	0.015832949
-0.005969961	0.128643998	-1.52222374	-0.016818661	-0.031344689	-0.017291683	0.003344329	0.262112764	0.010749332
0.010616393	5.947328884	0.007914444	0.007719287	-0.000704945	-0.001012066	0.053451211	0.090601463	-0.719993689
-0.123417439	4.129449569	-0.05002499	-0.005575442	0.128121041	-0.108183557	-0.069564087	-0.591753308	0.188824963
-0.003060519	-0.544837441	-0.010602867	0.040864075	-0.105814999	0.005866032	-0.201604878	-0.010024418	-0.06582271
-0.407580653	-0.025831027	-0.004120846	0.01108326	0.001106489	0.005691994	-0.021954299	0.248855355	0.399041746
teacher scaling								
300s_blackout	300s_loca	300s_sgtr	600.s_blackout	600.s_loca	600.s_sgtr	GV_120blackout	GV_120loca	GV_120sgtr
6.156051282	0.000338668	0.061697638	0.386332734	-1.071655282	0.097317863	0.011743802	-3.354611672	-0.037955755
0.001825955	0.010806858	-0.031916901	-5.992343144	0.022107176	0.00093105	0.039084004	-0.586768629	0.043874861
0.093986067	-0.000614035	0.007704989	0.059200582	1.802091328	-0.082979318	-0.001976468	-0.153590249	22.71062271
-0.091703323	0.002288842	-0.000374654	5.487800949	-23.97752131	0.121433596	0.008710248	0.312729442	0.105102694
1.038221776	0.024303843	-0.370233098	0.12098741	-0.010556907	1.964163325	0.026709215	-2.081047206	1.214806348
0.005489611	-0.596855963	0.009346998	2.169431301	-0.098804258	-0.867915435	-0.669628833	-1.03376401	-0.088533886
-0.173406492	10.44037203	23.41051962	0.055438005	-0.047560379	2.812288853	0.004837237	3.778261441	-0.016298055
0.002352964	-0.415721917	0.001841632	0.125433131	-0.024122772	0.022000482	-0.005608557	1.008811048	-0.263803497
0.073068956	30.62626941	0.061025816	0.657353615	0.038116079	-0.004112153	-1.390533434	5.656339144	0.569540811
0.013047477	0.00966491	0.317729308	-0.837248232	-0.621544441	-0.165229994	-0.063792016	0.008081716	0.157993927
-0.084202145	-0.018833949	0.002368758	11.58363971	0.02368205	-3.54356396	-0.064360437	-2.978126152	0.007879828
teacher shift								
300s_blackout	300s_loca	300s_sgtr	600.s_blackout	600.s_loca	600.s_sgtr	GV_120blackout	GV_120loca	GV_120sgtr
-7.234529895	5.036765811	1.377921359	0.846171151	5.554574504	-4.481723179	2.349656773	2.979111388	0.546138683
1.462572163	-1.625392193	2.36743082	98.65981066	0.68972825	0.107199771	2.81150701	-0.282233494	-1.153525111
1.514735548	0.248158321	0.347537765	10.60064517	-17.48366915	0.725651277	0.592142532	-2.769350599	0.524319455
18.50811185	-0.293108411	-1.967862253	-0.105416338	171.4129298	7.364116581	0.546175933	2.022351438	-0.95539933
32.70709764	0.730568455	0.32060636	0.855452435	0.707520582	-0.103338621	1.743138348	2.575509814	-10.36234474
0.886677452	2.119836437	0.928252249	12.16216581	-2.708995917	10.19922631	19.57055988	4.18121505	-1.27532229
2.620446471	-2.657485886	25.16479729	-7.587701102	4.345023331	8.290839679	-0.156600318	-27.86475977	0.51143214
0.896911331	-194.3848403	0.62055219	0.708023869	-0.775971787	-5.097662492	-3.329335132	5.587277123	4.760254429
2.483020024	-210.9072421	2.044772423	32.69233717	-2.607042365	0.946908992	35.00348282	0.289374303	-34.54568531
0.275976628	3.631829195	0.255799678	0.843065062	-2.446365312	17.14774621	2.026799693	3.327563656	3.33027134
4.724838003	0.608341162	0.517785505	-183.7599494	0.982286986	42.52608259	58.87762277	-8.038403897	16.86863752