Universidade Federal do Espírito Santo
Centro Tecnológico
Programa de Pós-Graduação em Engenharia Elétrica

Marcelo Souza Fassarella

Treinamento de Redes Perceptron Utilizando Janela Dinâmica

Vitória
2009

Marcelo Souza Fassarella

Treinamento de Redes Perceptron Utilizando Janela Dinâmica

Dissertação submetida ao Programa de Pós - Graduação em Engenharia Elétrica, da Universidade Federal do Espírito Santo, como requisito parcial para a obtenção do grau de Mestre em Engenharia Elétrica.
Orientador: *Prof. Dr. Sc. Evandro Ottoni Teatini Salles.*
Co-Orientador: *Prof. Dr. rer. nat. Hans-Jörg Andreas Schneebeli.*

Vitória
2009

Dados Internacionais de Catalogação-na-publicação(CIP)
(Biblioteca Central da Universidade Federal do Espírito Santo, ES, Brasil)

Fassarella, Marcelo Souza, 1978-
F249t Treinamento de redes perceptron utilizando janela dinâmica / Marcelo Souza Fassarella. - 2009.
101 f.:il.

Orientador: Evandro Ottoni Teatini Salles.
Co-Orientador: Hans-Jörg Andreas Schneebeli.
Dissertação (mestrado): Universidade Federal do Espírito Santo, Centro Tecnológico.

1. Redes Neurais (Computação). 2. Controle de ruído. 3. Aprendizado do computador. 4. Bioinformática. 5. Filtros elétricos digitais. I. Salles, Evandro Ottoni Teatini. II. Schneebeli, Hans-Jörg Andreas. III. Universidade Federal do Espírito Santo. Centro Tecnológico. IV. Título.

CDU: 621.3

Marcelo Souza Fassarella

Treinamento de Redes Perceptron Utilizando Janela Dinâmica

Dissertação submetida ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Espírito Santo, como requisito parcial para a obtenção do grau de Mestre em Engenharia Elétrica.

COMISSÃO EXAMINADORA

Prof. Dr. Sc. Evandro Ottoni Teatini Salles.
Universidade Federal do Espírito Santo
Orientador

Prof. Dr. rer. nat. Hans-Jörg Andreas Schneebeli.
Universidade Federal do Espírito Santo
Co-Orientador

Prof. Dr. José Leandro Félix Salles.
Universidade Federal do Espírito Santo

Dr. -Ing. Renato Antonio Krohling
Universidade Federal do Espírito Santo

Vitória, ES - BRASIL
Dezembro de 2009

"Perder tempo em aprender coisas que não interessam, priva-nos de descobrir coisas interessantes.".

Carlos Drummond de Andrade

A Beatriz, mulher inteligente. Amo você!

A Elisa, meu pequeno e grande amor. Uma artista de futuro.

Aos meus pais, por me incentivarem a estudar.

Agradecimentos

Agradeço à Beatriz e à Elisa, por privarem-se da minha companhia em muitos momentos, para que eu me dedicasse com afinco a esta obra. Beatriz também ajudou a corrigir a ortografia dos Capítulos 1 e 2.

Agradeço o professor Hans, por dedicar parte do seu tempo a este projeto. Ao professor Evandro, meu muito obrigado, pela paciência, pela grande ajuda que prestou, pelo conhecimento que passou e por sempre acreditar em mim.

Agradeço a Geocontrol, por ceder os dados do PIR usados nos testes de filtro. Ao Cleoner e José, responsáveis pela captura dos sinais. Ao Luiz, por sempre me incentivar e fornecer tempo para finalização deste trabalho.

Resumo

Neste trabalho apresentamos as redes neurais e o problema envolvendo o dilema bias-variância. Propomos o método da Janela a ser inserido no treinamento de redes supervisionadas com conjuntos de dados ruidosos. O método possui uma característica intrínseca de função regularizadora, já que procura eliminar ruídos durante a etapa de treinamento, reduzindo a influência destes no ajuste dos pesos da rede. Implementamos e analisamos o método nas redes lógicas adaptivas (ALN) e nas redes perceptrons de múltiplas camadas (MLP). Por último, testamos a rede em aplicações de aproximação de funções, filtragem adaptiva e previsão de séries temporais.

Abstract

In this work we discuss neural networks and the bias-variance dilemma. We propose the Window method to be inserted into supervisioned neural training with noise data. The method has an intrinsic caracteristic of regularization, because it tries to eliminate noise while the network is beeing trained, reducing its influence of the adjustment of network weights. We implement and analize the method in adaptive logic networks (ALN) and at multilayer perceptrons (MLP). Finally, we test the network in aplications as function aproximation, adaptive filters and time series prediction.

Sumário

Lista de Figuras

Lista de Tabelas

Nomenclatura

Termos

folhas — Neurônios que calculam uma saída relativa às entradas e seus respectivos pesos sinápticos na rede ALN

época — Passagem de todo o conjunto de dados de treinamento pela rede neural.

nós — Neurônios de decisão da rede ALN.

wnoise — Função do Matlab utilizada para gerar dados com descontinuidades.

Siglas

A/D — Analógico / Digital.

ALN — Rede lógica adaptiva (*Adaptive Logic Network*).

ALNGD — Rede lógica adaptiva com Algoritmo do Gradiente Descendente.

Flood — Biblioteca C++ de redes neurais artificiais em código aberto.

JG — Janela gradual.

LM — Levenberg-Marquardt.

LMS — Algoritmo do mínimo quadrado médio (*Least Mean Square Algorithm*).

MATLAB — Laboratório de matriz (*Matrix Laboratory*).

Max — Operador de máximo.

Min — Operador de mínimo.

MLP — Rede perceptron de múltiplas camadas (*Multilayer Perceptron*).

MSE	Erro quadrático médio (*Mean Square Error*).
NRMSE	Raiz quadrada do erro quadrático médio normalizado (*Normalized Root Mean Square Error*).
PIR	Sensor infravermelho passivo (*Passive Infrared Sensor*).
RF	Rádio frequência.
RMSE	Raiz quadrada do erro quadrático médio (*Root Mean Square Error*).
RNA	Rede neural artificial.
TRAINBR	Treinamento com regularização bayesiana.
TRAINLM	Treinamento com o algoritmo de Levenberg-Marquardt.
UPC	Universidade Politécnica da Catalunha (*Universitat Politécnica de Catalunya*).

Símbolos

η	Taxa de aprendizado.
λ	Coeficiente de regularização.
$\phi(\cdot)$	Função base.
ψ	Valor de abertura da Janela em um instante qualquer.
ψ_f	Valor final de abertura da Janela.
ψ_o	Valor inicial de abertura da Janela.
$\Re$	Conjunto dos números reais.
υ_ψ	Situação do erro quadrático em relação à Janela.
$\varphi'(t)$	Derivada da função de ativação.
$\xi\,(t)$	Valor instatâneo da energia total do erro.
$E(\cdot)$	Esperança matemática.
z^{-1}	Operador atraso unitário.

Capítulo 1

Introdução

1.1 Introdução

As redes neurais artificiais (RNAs) são estruturas computacionais que foram concebidas a partir de estudos do cérebro humano [14]. Elas são construídas por unidades básicas de processamento de informação chamadas de neurônios, os quais são organizados em camadas, de modo que o processamento das informações ocorra em paralelo e seja não-linear[1]. No projeto de RNAs existem elementos como a taxa de aprendizado, o número de épocas, a função de ativação de cada neurônio, o número de camadas ocultas da rede, o número de neurônios ocultos, entre outros, chamados de parâmetros livres da rede, que devem ser definidos para uma tarefa em particular. Após a definição dos parâmetros livres é necessário ajustar os pesos das conexões sinápticas da rede. Esse ajuste é feito por um algoritmo capaz de extrair as informações contidas em um conjunto de dados e repassá-las a estes pesos.

Um método muito comum de inserir o conhecimento do ambiente na rede é através do uso de exemplos rotulados, que nada mais é do que um conjunto de dados dos quais se conhece, para cada entrada, a saída desejada. A rede aprende procurando minimizar a diferença entre a resposta desejada e a resposta de saída produzida por um estímulo em particular. Esse processo é conhecido como aprendizagem supervisionada e é muito útil em aplicações envolvendo aproximação de funções, previsão de séries temporais e filtragem neural [36] [26] [5].

Em um contexto neural, mesmo no humano, aprender não é reproduzir

[1] Em acordo com o modelo biológico.

com exatidão os dados que nos foram ensinados[2], mas sim, utilizar o conhecimento adquirido para gerar bons resultados mesmo para dados nunca vistos. Esta propriedade é chamada de capacidade de generalização e leva-nos ao seguinte questionamento: *Quando podemos considerar que uma rede neural aprendeu?* A resposta a esta pergunta normalmente é obtida utilizando-se um conjunto de dados de teste, desconhecidos pela rede e não utilizados no processo de aprendizado. O resultado de saída, gerado por esse conjunto de dados, é utilizado para fornecer uma estimativa do quanto a rede está apta a lidar com variações nos dados de entrada.

Para o projetista de uma rede neural encontrar a rede que minimize a medida do erro durante o processo de treinamento, e que ainda assim seja capaz generalizar, é um problema conhecido como dilema bias-variância, e é um processo complexo e trabalhoso. Complexo porque os resultados dependem principalmente do tamanho da rede, do tempo e da velocidade de aprendizado e do conjunto de dados de treinamento e de teste. Trabalhoso pois normalmente são necessários inúmeros testes antes de definir a rede com os parâmetros de aprendizado que mais se adequam à solução. Entretanto, mesmo com estes cuidados, corre-se o risco de que a escolha forneça algum resultado imprevisto para um dado que não estava contido no processo ajuste e validação do sistema.

Para lidar com o dilema *bias*-variância muitos métodos têm sido desenvolvidos e podem ser encontrados na literatura, dentre os quais podemos citar:

- Técnicas de poda da rede [33].
- Técnicas de crescimento da rede [17].
- Decaimento dos pesos [16].
- Técnicas de parada antecipada [9].
- Treinamento com injeção de ruído [15].
- Regularização Bayesiana [11]

Nenhuma das técnicas anteriores por si só é suficiente para resolver todos os tipos de problemas existentes. Em geral, elas são polarizadas para determinadas classes de problemas, muitas vezes necessitando de um conhecimento antecipado das estatísticas envolvendo o conjunto de dados e de uma quantidade mínima de amostras nesse conjunto.

[2] Isso é considerado decorar.

Dentre as técnicas destacaremos as heurísticas de parada antecipada, que por sua, em geral, simplicidade, são utilizados em muitos algoritmos de treinamento. A ideia básica desses métodos é finalizar o aprendizado antes que o conjunto de dados de treinamento seja decorado. Para fazer isso, alguma métrica é usada durante o processo de ajuste dos pesos, como, por exemplo, pode-se medir o erro quadrático médio gerado por um conjunto de validação e quando este erro aumentar de uma medição para outra, o treino é finalizado.

1.2 Definição do problema

O problema a ser tratado é o do ajuste excessivo das redes neurais aos dados de treinamento ruidosos, que produz, como consequência, uma redução da habilidade de generalizar. Diz-se que uma rede generaliza bem quando o mapeamento de entrada-saída computado for correto (ou aproximadamente correto) para dados de teste não utilizados no treinamento da rede. Visto que o processo de aprendizagem pode ser entendido como um mapeamento não-linear dos exemplos de entrada-saída, a generalização pode ser considerada como o efeito de uma boa interpolação não-linear sobre os dados de entrada [41]. Chama-se de dilema *bias*-variância ao compromisso a ser estabelecido entre a minimização do erro durante o treinamento (*bias*) e a capacidade de generalização obtida na etapa de teste (*variância*).

A generalização é influenciada pelo tamanho do conjunto de treinamento (e o quão representativo do ambiente de interesse ele é), pela arquitetura da rede e pelo nível/tipo de ruído presente no conjunto de dados. Considerando uma rede com arquitetura fixa, o interesse é determinar o melhor resultado que esta produzirá, em virtude das características dos dados utilizados.

As abordagens tradicionais de regularização produzem resultados bons para dados contaminados com ruídos cuja distribuição é gaussiana. Quando nestes dados existem ruídos com chegada aleatória, de forma que a distribuição assuma uma outra forma qualquer, é necessário, em uma etapa a priori, retirar esses elementos contaminantes, por meio de alguma estimativa do ruído existente. Portanto, é de interesse que exista algum meio de treinar a rede para um conjunto de dados qualquer, sem necessariamente ter de assumir características específicas para o ruído contido nesses dados, deixando para o que o próprio algoritmo de treinamento julge quando um par de entrada-saída é inadequado para ser usado no processo de ajuste.

Nesta dissertação propõem-se o uso de um método de regularização baseado no bloqueio do ajuste da rede para dados considerados "inadequados", onde, o termo "inadequados", refere-se aos dados cujo erro absoluto (ou erro quadrático) produzido pela rede seja maior do que um limite máximo permi-

tido, formando o que pode-se chamar de Janela (ou região) de permissão. A abertura da Janela é alterada a cada época, ou seja, à medida que a rede se ajusta ao conjunto de dados, o que lhe dá um certo dinamismo. Por isso esse processo pode ser chamado de Janela Dinâmica, ou mais especificamente de Janela Gradual, quando a abertura é reduzida passo-a-passo.

1.3 Metodologia

Como forma de avaliar se um novo método de regularização é genérico e analisar os efeitos que ele provoca na solução dos diversos tipos de problemas relacionados às redes neurais, é importante mostrar os resultados para uma rede neural configurada como:

- um aproximador de funções,
- um previsor de séries temporais,
- um filtro.

Para comparar o desempenho de cada rede, pode-se utilizar como medida a raiz quadrada do erro quadrático médio (RMSE), sua variante normalizada (NRMSE) e, no caso de previsões de séries temporais, a medida do U de Theil [1]. Sempre que possível, também é de grande interesse que os resultados gráficos sejam gerados.

No caso dos testes teóricos, feitos com o aproximador de funções, utilizando conjuntos de dados ruidosos, pode-se medir o RMSE e o NRMSE em relação aos dados de treinamento, sinal contaminado, e em relação aos dados gerados diretamente pelas funções alvo, sinal limpo. Isso mostrará o quão próximo o resultado ficou em relação aos dois conjuntos de dados.

Para mostrar a facilidade de adaptação de um método regularizador em qualquer tipo de rede neural, é interessante, se possível, que ele seja implementado em redes com características distintas.

Como forma de avaliar se uma nova abordagem produz ganhos em relação às abordagens tradicionais, é importante que, para cada novo conjunto de dados, as mesmas análises de desempenho sejam feitas para estas últimas, considerando sempre redes de mesmas características.

1.4 Organização da dissertação

Este trabalho está organizado da seguinte forma:

O capítulo 2 estabelece os conceitos básicos das redes neurais, discute a problemática envolvendo o dilema bias-variância e analisa alguns métodos de regularização já disponíveis na literatura.

O capítulo 3 detalha a ideia geral do método proposto, como ele afeta o algoritmo de retropropagação e as modificações necessárias a serem feitas no software para sua implementação.

O capítulo 4 discute e analisa diversos resultados de teste, comparando os dados de saída produzidos por redes com e sem o método da Janela.

No capítulo 5 concluímos o trabalho e apresentamos propostas para trabalhos futuros.

Capítulo 2

Descrição do Problema

2.1 Introdução

Uma rede neural artificial é um modelo computacional criado a partir do estudo do funcionamento do cérebro humano. Ela surgiu em 1943 com o artigo de McCulloch e Pitts entitulado "*A logical calculus of ideas imminent in nervous activity*"[23]. Motivado por este artigo a investigar como olho e o cérebro humano processam computacionalmente a informação que chega do ambiente, Frank Rosenblatt, em seu trabalho sobre o *perceptron*, publicou um método inovador de aprendizagem supervisionada [34], que foi um dos mais importantes trabalhos feitos na fase inicial do desenvolvimento das modernas redes neurais artificiais.

2.1.1 Modelo de um neurônio

Um neurônio é um elemento base para o processamento de informação de uma rede neural. O modelo mais largamente utilizado é o representado pela Figura 2.1, onde cada neurônio contém um conjunto de pesos sinápticos e um bias[1] combinados de acordo com a equação[2] 2.1. O resultado de saída do neurônio é obtido passando-se a saída do combinador por uma função de ativação.

$$y = f\left(\sum_{i=1}^{N} w_i x_i + b\right) \tag{2.1}$$

[1]Normalmente o bias é considerado como um sinal de entrada constante e igual a um, e sua conexão possui um peso sináptico w_b

[2]Outros tipos de combinadores, como um combinador de segunda ordem, também podem ser utilizados, mas são menos comuns.

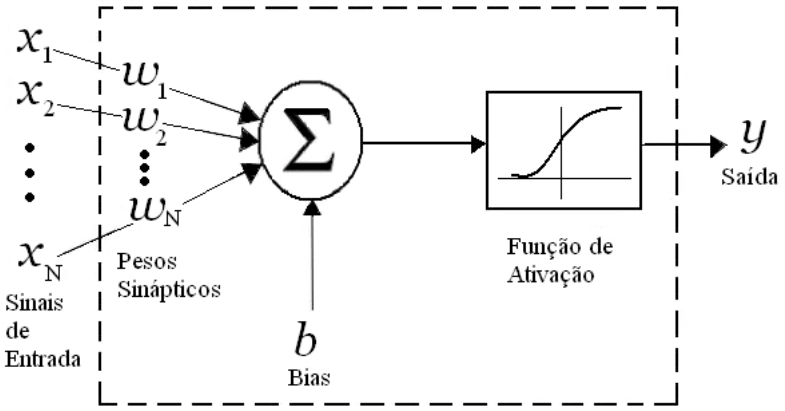

Figura 2.1: Modelo genérico de um neurônio

onde:

- w_i são os pesos sinápticos,
- x_i são os sinais de entrada,
- b é o *bias*.

2.1.2 Aprendizagem supervisionada

O principal objetivo da aprendizagem supervisionada é encontrar uma função que melhor descreve o comportamento entrada-saída de um conjunto finito de exemplos de dados do problema. Neste caso, para cada vetor de entrada existe um vetor alvo que será utilizado para passar o conhecimento que se tem do ambiente à rede. Isso pode ser feito através do ajuste dos pesos sinápticos do sistema por meio de um sinal de erro, que é a diferença entre a resposta desejada e a resposta realizada pela rede.

Muitos problemas importantes podem ser resolvidos com esta metodologia, mas neste trabalho estaremos focados nos problemas de regressão, onde o vetor alvo representa alguma medida relativa ao vetor de entrada. Nos casos reais, para este tipo de problema, o vetor alvo é contaminado com ruído devido a alguma influência externa, como por exemplo:

- Ruído devido a erros de quantização dos dados, devido a conversores A/D,
- ruído branco ou colorido de fundo, com baixa densidade espectral de potência,

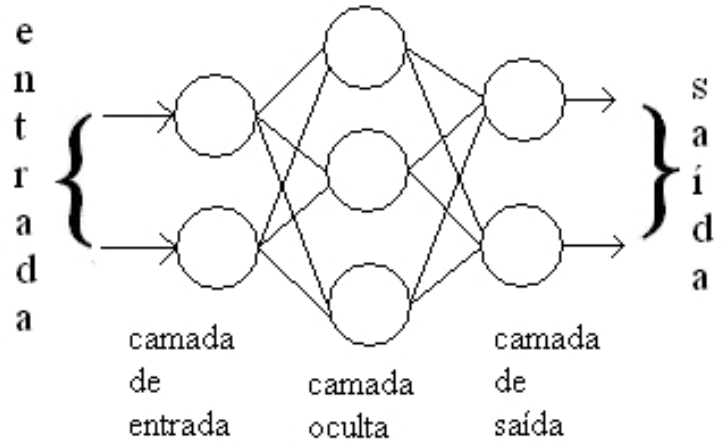

Figura 2.2: Modelo genérico de um perceptron de múltiplas camadas.

- ruído ou *outliers* devido a erros de medida,
- ruído ou *outliers* devido a distúrbios periódicos e/ou aleatórios (comuns em redes elétricas, ao se ligar ou desligar motores, por exemplo).
- ruído devido à captura de sinal com baixa relação sinal/ruído,
- ruído devido à resposta de componentes às variações térmicas.

2.1.3 Perceptrons de múltiplas camadas - MLP

Os perceptrons de múltiplas camadas são capazes de realizar um mapeamento não-linear de entrada-saída de natureza geral, ou seja, são aproximadores universais de funções. Eles têm sido aplicados com sucesso na resolução de diversos problemas como em [37] [38] [40], através do uso de treinamento supervisionado e com o algoritmo de retropropagação do erro. Em um dos seus modelos construtivos, os seus neurônios, que estão distribuídos em camadas, são interligados de forma a propagar o sinal aplicado da entrada para a saída. A Figura 2.2 representa uma arquitetura genérica do perceptron de múltiplas camadas.

2.1.4 Redes lógicas adaptivas - ALN

As redes lógicas adaptativas surgiram em 1974, como estruturas binárias com elementos lógicos de decisão (E, OU e Não), dispostos em forma de árvore [7] [27]. Elas evoluiram para poderem lidar diretamente com os valores numéricos dos problemas [2], ao invés de convertê-los para uma representação binária correspondente. Para isso ela conta com dois tipos distintos de neurônios: as folhas e os nós [3] [10].

As folhas implementam um neurônio conforme mostrado pela figura 2.1, mas com função de ativação constante e igual a um, ou seja, a equação final da folha é idêntica à Eq. 2.1. Já os nós utilizam um operador não linear de máximo ou de mínimo, conforme abaixo:

$$Maximo(R1, R2) = \begin{cases} R1 & \text{se } R1 \geq R2 \\ R2 & \text{se } R1 < R2 \end{cases}$$

$$Minimo(R1, R2) = \begin{cases} R1 & \text{se } R1 \leq R2 \\ R2 & \text{se } R1 > R2 \end{cases}$$

Podemos ver na Figura 2.3a um exemplo genérico de uma árvore ALN e ao lado, um possível cenário do resultado gerado por ela.

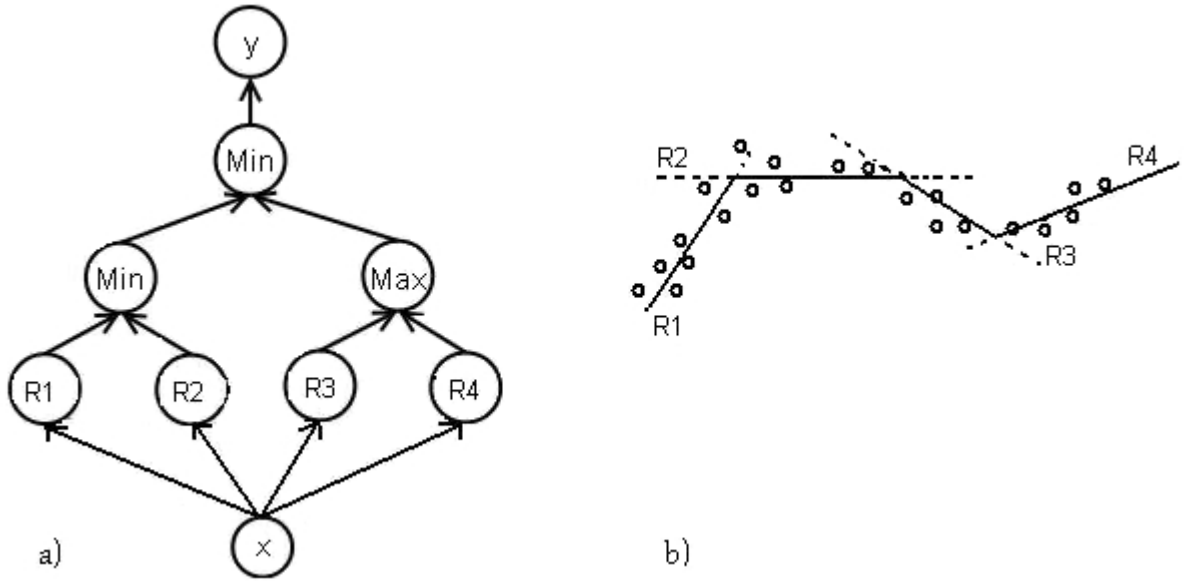

Figura 2.3: Figura exemplo de uma rede neural lógica adaptiva (ALN). a) Árvore ALN b) Resultado gerado.

As ALNs também são aproximadores não-lineares e universais de funções. Porém, diferentemente dos MLPs, nas ALNs é possível encontrar o neurônio

folha (perceptron linear) culpado por uma determinada saída. Essa é uma grande vantagem nestas implementações, já que é possível melhorar a resposta devido a uma entrada específica, corrigindo ou expandindo apenas um único neurônio. Esta característica também acaba por acelerar o algoritmo de treinamento, já que, para cada vetor de entrada apenas uma folha será corrigida por vez.

2.1.5 Problemas típicos para o uso de redes neurais

Aproximação de Funções

As redes neurais tem sido utilizadas para desempenhar tarefas de aproximação de funções [32] [19] [18]. Neste tipo de problema, deseja-se que a rede seja capaz de efetuar o mapeamento de entrada-saída de uma função $f(x)$ desconhecida. Para isso, são apresentados um conjunto de pontos $\{x_i, f(x_i)\}_{i=1}^{N}$ para treinar a rede. Os vetores x_i são utilizados como vetores de entrada e ao passar pela rede, geram o sinal de saída y_i. O erro e_i, que será utilizado para corrigir os pesos da rede, é obtido subtraindo-se a resposta desejada $f(x_i)$ da saída y_i, conforme a Figura 2.4.

Apesar do modelo apresentado acima ser genérico, para aplicações do mundo real faz-se necessário acrescentar mais um termo à equação: o ruído. Contaminação por ruído está presente em qualquer sistema não idealizado, aparecendo no conjunto de dados por diversas razões, como erros de arredondamento e quantização, defeitos nos equipamentos de medição, ruídos térmicos e até mesmo erro humano [6].

A tarefa de treinar uma rede neural envolve a minimização de alguma função de custo, sendo a mais usual a técnica dos quadrados mínimos. Esta técnica provê ótimos resultados se a distribuição do erro for gaussiana. Contudo, em aplicações reais, a distribuição do erro é desconhecida e podem surgir pertubações discordantes das demais, chamadas de *outliers* [13] [8]. Os *outliers* podem surgir devido a erros grosseiros de medição ou devido a intervenções exógenas, por exemplo, greves, alterações súbitas do mercado, alterações inesperadas em certas condições do sistema físico ou outras [28].

Previsão e Filtragem

O tempo constitui um ingrediente essencial do processo de aprendizagem e é através da sua incorporação na operação de uma rede neural que esta é capacitada a seguir as variações estatísticas de um sinal.

O processamento temporal é obtido através do reconhecimento de padrões que evoluem no tempo, de forma que a resposta em um instante particular

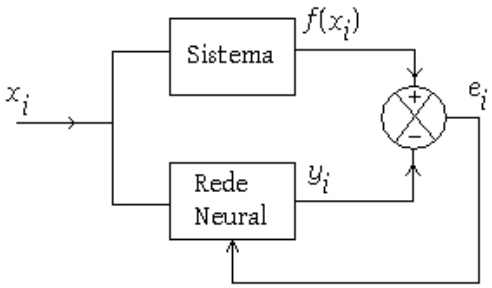

Figura 2.4: Diagrama de blocos de um sistema aproximador de funções.

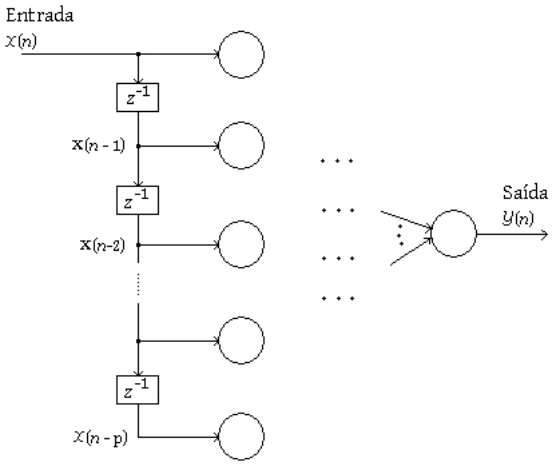

Figura 2.5: Rede neural atrasada no tempo.

dependa não apenas do valor atual do dado, mas também de seus valores passados. A Figura 2.5 representa genericamente uma rede neural esquematizada para processar informações temporais. O simbolo z^{-1} é o operador atraso unitário, isto é, z^{-1} opera sobre *x(n)* produzindo sua versão atrasada *x(n - 1)*.

Nos problemas de previsão e filtragem é necessário utilizar redes com capacidade de processamento temporal. Apesar da estrutura básica da rede para ambos os problemas ser a mesma, conforme mostrado na Figura 2.5, existem particularidades construtivas e conceituais que as diferem.

Particularidades dos Filtros: O filtro é um elemento utilizado para extrair informação de interesse de um conjunto de dados com ruído. O ruído pode surgir de uma variedade de fontes como por exemplo, através de medições com sensores ruins, equipamentos defeituosos e interferências eletromag-

néticas. O filtro diz-se linear se sua saída é uma função linear das observações apresentadas à sua entrada. Este tipo de filtro necessita de um conhecimento sobre as estatísticas do sinal a processar. O filtro diz-se adaptativo quando seus pesos são ajustados por algum algoritmo recursivo. Em um ambiente não estacionário, a recursividade do algoritmo possibilita ao filtro acompanhar as variações estatísticas dos dados ao longo do tempo.

O modelo básico para construção do filtro adaptativo é mostrado na Figura 2.6.

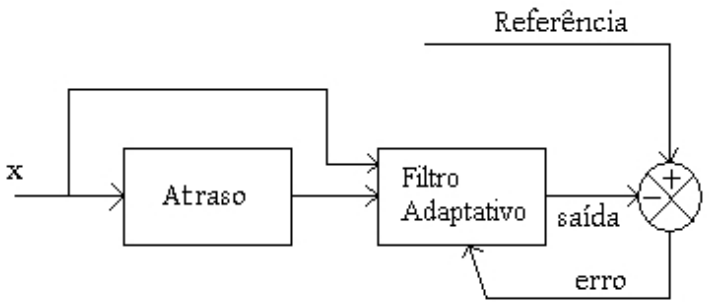

Figura 2.6: Diagrama de blocos básico de um filtro neural.

Particularidades dos Previsores: Um sistema previsor é uma ferramenta importante para auxiliar tomadas de decisão em engenharia, nos negócios e em qualquer área científica. Para entender como funciona um previsor, primeiro considere um conjunto de amostras $y(t_1)$, $y(t_2)$, ... , $y(t_n)$ em uma sequência temporal t_1, t_2, ..., t_n, e segundo, estabeleça o objetivo, que é determinar com base nos valores passados qual o valor $y(t_m)$, sendo $m > n$.

Em uma rede neural supervisionada, as amostras conhecidas são aplicadas como sinais de entrada e o valor desejado no futuro[3] é utilizado para formação do erro e ajuste dos pesos.

O modelo básico para construção do previsor é mostrado pela Figura 2.7.

[3]Durante a fase de treinamento os valores futuros devem ser valores conhecidos.

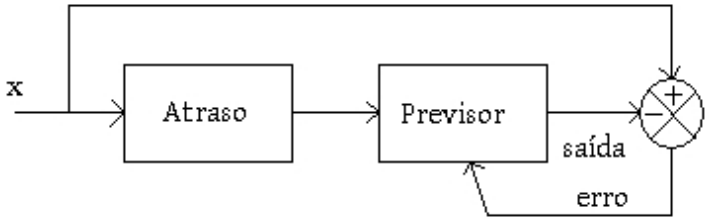

Figura 2.7: Diagrama de blocos básico de um previsor neural.

2.2 Dilema *Bias* - Variância

Uma das dificuldades em se trabalhar com uma rede neural é saber definir o número de neurônios ocultos da rede para um determinado problema, uma vez que, se forem escolhidos modelos neurais complexos[4], disponibilizar-se-á flexibilidade suficiente para que a rede seja treinada em excesso. Isto é, no treinamento o erro tende a valores muito pequenos, próximos a zero, mas a função gerada torna-se complexa, indicando uma incapacidade de generalizar[5], vide Figura 2.8 . Por outro lado, se limitarmos muito o número de neurônios ocultos, de forma a evitarmos o super treinamento, a rede poderá perder sua flexibilidade, compromentendo até mesmo a capacidade de modelar importantes tendências nos dados, como mostrado na Figura 2.9. Este problema é conhecido como dilema bias-variância.

Segundo Haykin [14], o vetor de pesos w que minimiza a distância quadrática entre a função de regressão f(x) e a função aproximativa $F(x, w)$ é dada pela expressão:

$$L_{med}\left(f\left(x\right), F(x, w)\right) = E_{\tau}\left(\left(f\left(x\right) - F(x, \tau)\right)^2\right), \tag{2.2}$$

onde:

- $f(x)$ é a média condicional da saída do modelo D quando a entrada X $= x$ e f(x) $= E(D \mid X = x)$.,
- $F(x, w)$ é a saída da rede neural produzida em resposta ao vetor de entrada x,
- w representa os pesos da rede,

[4]O termo complexo é usado no sentido da escolha de muitos neurônios ocultos.

[5]O termo generalizar refere-se a capacidade da rede de apresentar durante a fase de testes, bons resultados para dados desconhecidos

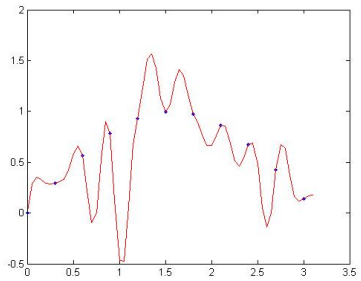

Figura 2.8: Resultado para o treinamento com poucos pontos e muitos neurônios. A função de saída é complexa.

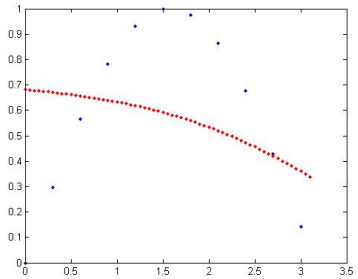

Figura 2.9: Resultado para o treinamento com poucos pontos e poucos neurônios. A função de saída não possui flexibilidade.

- $E(\cdot)$ é o operador de esperança matemática,
- τ representa todo o conjunto de amostra do treinamento.

Com as devidas manipulações matemáticas[6] a Equação (2.2) torna-se:

$$L_{med}\left(f\left(x\right), F\left(x,\tau\right)\right) = B^2\left(w\right) + V\left(w\right), \tag{2.3}$$

Agora fazemos duas observações importantes:

1. O termo B(w) é o *bias* do valor médio da função aproximativa $F(x,\tau)$, medido em relação à função de regressão $f(x) = E(D \mid X = x)$. Este termo representa a incapacidade da rede neural definida pela função $F(x,w)$ de aproximar com precisão a função de regressão $f(x) = E(D|X = x)$. Deste modo, podemos ver o bias como um erro aproximativo.

2. O termo V(w) é a variância da função aproximativa $F(x,w)$, medida sobre toda a amostra de treinamento τ. Este termo representa a não-adequação da informação contida na amostra de treinamento τ acerca da função de regressão $f(x)$ ou simplesmente como um erro *estimado*.

Um outro caminho que nos leva a lidar com o problema do dilema bias-variância é através da utilização de dados ruidosos. Neste caso, mesmo conjuntos de tamanho relativamente grande não garantem uma bom casamento entre aproximação e generalização. Uma vez que, se reduzirmos excessivamente o bias, estaremos, consequentemente, aprendendo o ruído inerente dos dados e assim, obteremos um resultado pobre em generalização, conforme Figura 2.10.

[6]Para maiores detalhes consulte Haykin [14] p.110 - 114.

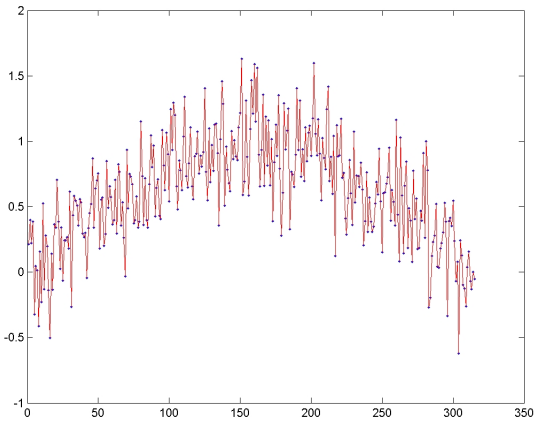

Figura 2.10: Resultado obtido após o treinamento com um conjunto de dados ruidoso. Percebemos que há uma baixa capacidade de generalização.

Infelizmente, constatamos que em uma rede neural que aprende por exemplos utilizando para isso um conjunto de treinamento de tamanho reduzido ou dotado de ruídos, o preço para se obter um *bias* pequeno é uma variância grande. Somente quando o tamanho da amostra de treinamento se torna infinitamente grande e a potência de ruído muito pequena, é que podemos esperar eliminar tanto o bias como a variância, ao mesmo tempo.

2.3 Abordagens para solução do dilema *bias* - variância

As técnicas de regularização são empregadas para evitar o supertreinamento de sistemas baseados em aprendizagem por exemplos rotulados (dependente dos dados). Assuma que a função de erro total que se deseja minimizar [6][7] possa ser escrita como

$$E_D(w) + \lambda E_W(w), \tag{2.4}$$

onde λ é o coeficiente de regularização que controla a importância relativa entre a componente de erro dependente dos dados $E_D(w)$ e o termo regularizador $E_W(w)$. Se a operação de regularização for dada pela soma de quadrados de w, podemos escrever a função de erro total como

[7] Para maiores detalhes consulte o Capítulo 3 de Bishop [6].

$$\frac{1}{2}\sum_{n=1}^{N} t_n - w^T\phi(x_n)^2 + \frac{\lambda}{2}w^T w. \tag{2.5}$$

onde:

- t_n é a enésima variável alvo,
- N é o número total de amostras,
- $\phi(x_n)$ é a função base.

O controle da complexidade ou regularização em redes neurais visa adequar o nível de não-linearidade disponibilizado pela estrutura à regularidade da superfície que deseja-se modelar, evitando a incorporação desnecessária do ruído e a consequente redução da capacidade de generalização do modelo. Dentre os métodos utilizados para essa finalidade, podemos citar a técnica de validação cruzada e as técnicas de poda e crescimento da rede.

Poda e Crescimento da Rede

Conforme vimos na seção 2.2, redes muito pequenas são incapazes de aproximar o conjunto de dados e que redes muito grandes são pobres em generalização. Considerando este fato podemos afirmar que: é possível encontrar uma de rede com tamanho adequado, que seja capaz de estimar a função desejada, com determinado grau de precisão, e também consiga generalizar. Podemos alcançar este objetivo de projeto de duas formas:

- Pela *poda da rede*, começa-se com uma rede grande o suficiente para modelar a função desejada e então, poda-se a rede através da eliminação de alguns pesos sinápticos ou neurônios, num processo seletivo e ordenado. O procedimento de poda adiciona o termo $E_W(w)$ à função de erro, de acordo com a técnica empregada,
- pelo *crescimento da rede*, começa-se com uma rede pequena o suficiente para ser incapaz de estimar a função desejada e então, adiciona-se um novo neurônio ou uma nova camada oculta enquanto as especificações de projeto não forem atendidas[8].

[8] As implementações mais recentes das redes ALN usam a técnica de crescimento para criar os neurônios lineares (ltus), chamados de folhas, e ao mesmo tempo os neurônios lógicos ou de decisão, chamados de nós [4].

Decaimento dos Pesos. No Procedimento de decaimento dos pesos, o termo de punição da complexidade é definido como a norma quadrada do vetor de pesos w conforme

$$E_W(w) = ||w||^2. \tag{2.6}$$

Esta técnica é conhecida em estatística como encolhimento de parâmetros e opera de modo a forçar os pesos sinápticos com pouca influência sobre o modelo a assumirem valores próximos de zero, com o objetivo de eliminar a participação dos elementos que poderiam causar ajuste excessivo dos dados e um resultado pobre em generalização.

Eliminação de Pesos. Neste segundo procedimento, a punição da complexidade é definida por

$$E_W(w) = \sum_{i \in \wp_{total}} \frac{(w_i/w_o)^2}{1 + (w_i/w_o)^2}. \tag{2.7}$$

onde:

- w_o é um parâmetro preestabelecido,
- w_i se refere ao peso de uma sinapse i,
- e $\wp_{total}$ se refere a todas as conexões sinápticas da rede.

Quando $|w_i| << w_o$, a punição da complexidade para aquele peso se aproxima de zero. A implicação desta condição é que o i-ésimo peso sináptico não é confiável e deveria ser eliminado da rede.

Validação Cruzada

A validação cruzada é uma técnica de parada antecipada simples de entender e fácil de usar. Primeiro, deve-se dividir aleatoriamente o conjunto de treinamento em dois subconjuntos[9]:

- *Subconjunto de estimação*, usado para ajustar os pesos da rede durante o treinamento.
- *Subconjunto de validação*, usado para verificar a capacidade de generalização da rede.

[9]Existem outras variantes da validação cruzada, como a validação cruzada múltipla, em que os exemplos são divididos em mais de dois subconjuntos

Depois, deve-se modificar a etapa de treinamento da seguinte forma: treina-se a rede com o subconjunto de estimação durante um período T de épocas. Em seguida, usa-se os dados do subconjunto de validação para estabelecer o erro de generalização. Este processo é repetido até que se possa identificar que o erro de generalização de um determinado momento K seja maior do que seu antecessor K-1, conforme Figura 2.11.

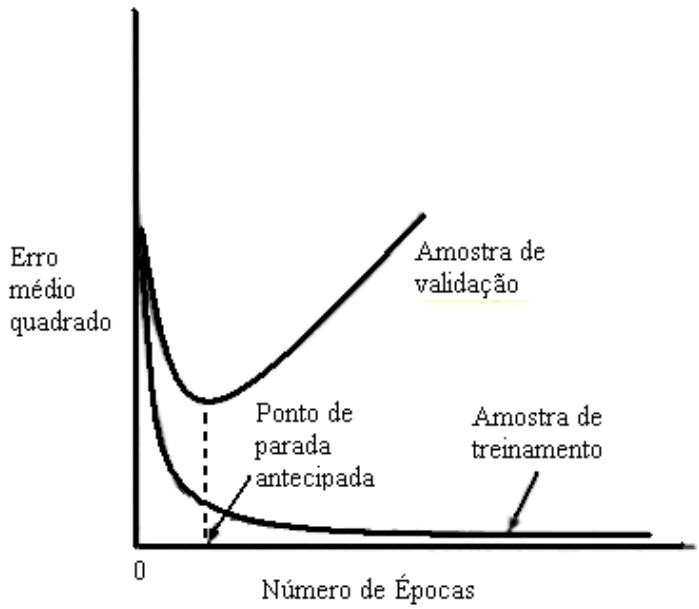

Figura 2.11: Ilustração da regra de parada antecipada por validação cruzada.

Dentre os inconvenientes que podem ser associados a esta técnica podemos citar:

- A redução do conjunto de treinamento, que é um ponto crítico para conjuntos pequenos,
- o ponto de parada ocorre sempre após o ponto de mínimo do erro de validação, caso contrário, o custo computacional torna-se alto,
- o processo funciona bem somente para o treinamento em lote[10], pois o erro quadrático de uma época para outra é mais estável, pois, não depende da ordem em que os dados são apresentados à rede durante o treinamento. No processo de treinamento por ponto, também conhecido como processo *on-line*, a redução do erro quadrático percorre um

[10]No treinamento em lote o ajuste dos pesos é feito com o erro gerado por todos os dados de treinamento, obtidos somente após uma época.

caminho mais ruidoso e instável, principalmente se embaralharmos os dados de uma época para outra.

2.4 Janela dinâmica

A técnica da Janela Dinâmica, ou simplesmente Janela, proposta nesta dissertação, procura regularizar uma rede neural treinada com dados ruidosos, através da eliminação de parte desses dados do processo de ajuste dos pesos, conforme Figura 2.12. A identificação dos dados indesejados é feita durante o processo de treinamento, através de uma métrica envolvendo o erro (ou o erro quadrático) de saída da rede, o qual, espera-se que esteja contido em uma região (a Janela) de interesse.

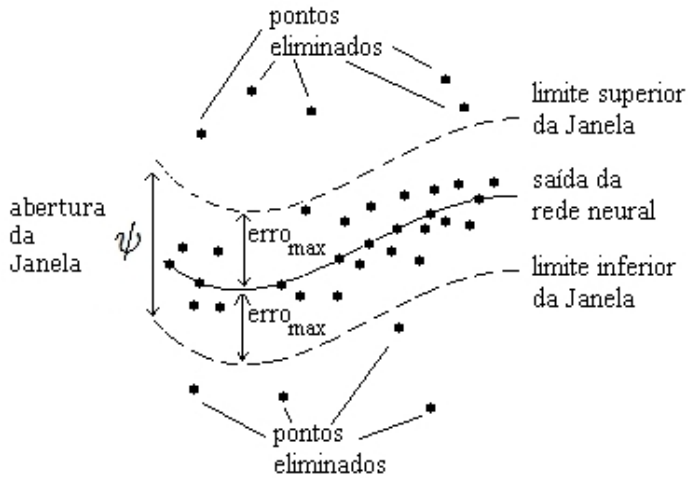

Figura 2.12: Ilustração do comportamento do método da Janela.

Dentre as vantagens deste método podemos citar:

- a não necessidade de se reduzir o conjunto de treinamento,
- a facilidade de interpretação e implementação da técnica em qualquer tipo de rede supervisionada,
- o baixo custo computacional,
- pode ser utilizado em algoritmos de treinamento sequencial,

Consideramos como aspectos inconvenientes do método:

- a não existência de uma medida para o ponto ótimo relacionando ajuste e generalização,
- o tempo de treinamento não é reduzido,
- a estrutura da rede não é minimizada automaticamente.

Podemos dizer que a Janela é um método capaz de regularizar o treinamento de uma rede neural, uma vez que, o ruído contido nos dados não utilizados no ajuste dos pesos sinápticos, não terá como ser aprendido.

2.5 Resumo

Neste capítulo fizemos uma breve revisão sobre as principais características das redes neurais. Abordamos as redes perceptrons de múltiplas camadas (MLP) e das redes lógicas adaptivas (ALN). Vimos alguns dos problemas onde elas podem ser utilizadas. Depois discutimos sobre a relação entre o ajuste excessivo dos dados de treinamento e a capacidade de generalização da rede, que é um problema conhecido como dilema *bias*-variância. Por último, mostramos algumas técnicas que foram desenvolvidas para tentar minimizar o problema relativo a este dilema.

Capítulo 3

Método da Janela

Neste capítulo apresentaremos a teoria por trás do método da Janela e como ela foi implementada nas redes MLP e ALN. O objetivo principal do método é o de capacitar as redes a serem robustas mesmo sob condições extremas de sinais ruidosos e sem necessariamente ter que considerar os ruídos como tendo distribuição Guassiana. A teoria será apresentada considerando-se um ambiente unidimensional, mas se necessário, poderá ser expandida facilmente para um ambiente multidimensional.

3.1 Introdução

Normalmente ao se iniciar os estudos em redes neurais são utilizados conjuntos de dados teóricos bem comportados: com pouca ou nenhuma transição abrupta, sem conjunto de dados faltantes e com pouco ou nenhum ruído. Quando há existência de ruídos a distribuição destes é considerada como sendo Guassiana. Resolvemos a partir daí investigar o comportamento neural frente a condições mais severas. Esse processo envolveu o uso de dados teóricos com ruídos não puramente Guassianos e dados capturados por sistemas de medição usados em aplicações práticas. Percebemos que, se o algoritmo de aprendizado neural é estável para um conjunto de pontos quaisquer $\{(x_1, f(x_1)), (x_2, f(x_2)), ..., (x_n, f(x_n))\}$, o erro quadrático médio inicial começa grande, com valor determinado pelas condições iniciais e então, decresce a uma taxa que depende do algoritmo utilizado e dos parâmetros envolvidos, como por exemplo, o valor da taxa de aprendizado.

Considere então os seguintes problemas:

Problema 1:

Seja $y = f(x)$ uma função qualquer e de ordem desconhecida, representada por um conjunto de pontos $\{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$, à qual é

adicionada um ruído gaussiano, formando um novo conjunto $\{(x_1, yr(x_1)), (x_2, yr(x_2)), \ldots, (x_n, yr(x_n))\}$, conforme Figura 3.1. Projete uma rede neural que gere uma boa estimativa ys $= \hat{f}(x)$ de $f(x)$, conhecendo-se apenas o conjunto ruidoso e não dispondo de nenhuma informação estatística da função $f(x)$ e nem do ruído.

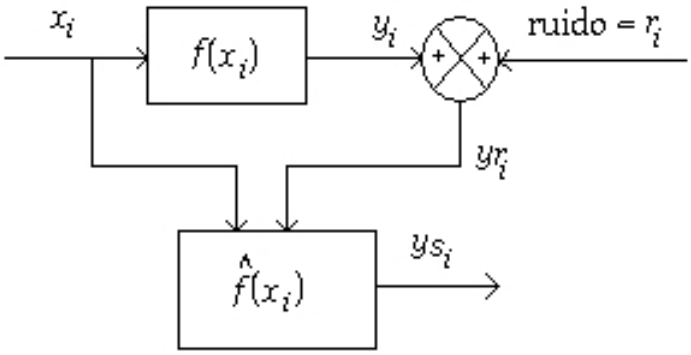

Figura 3.1: Diagrama de blocos com injeção de ruído, sendo $i = \{1, 2, \ldots, n\}$.

Este é um problema ao qual as redes neurais artificiais estão aptas a resolver, bastando para isso que alguns aspectos importantes para o treinamento sejam atendidos, como:

- O tamanho do conjunto disponível deve ser o mais representativo possível.
- A rede neural deve ser bem dimensionada, de forma que ela seja capaz de minimizar o erro e maximizar a capacidade de generalização.
- Determinação de uma taxa de aprendizado adequada.

Problema 2:

Considere agora que ao Problema 1 sejam injetados ruídos não gaussianos e com chegada aleatória e_j, com $j \in \aleph$, ao conjunto $\{(x_1, yr(x_1)),(x_2, yr(x_2)), \ldots ,(x_n, yr(x_n))\}$, conforme o diagrama da Figura 3.2. Para este problema, como podemos projetar uma rede neural que gere uma função $\hat{f}(x)$ que melhor aproxime a função $f(x)$?

Uma resposta alternativa a esta pergunta é: se pudermos identificar e eliminar do conjunto de treinamento os vetores que foram contaminados com

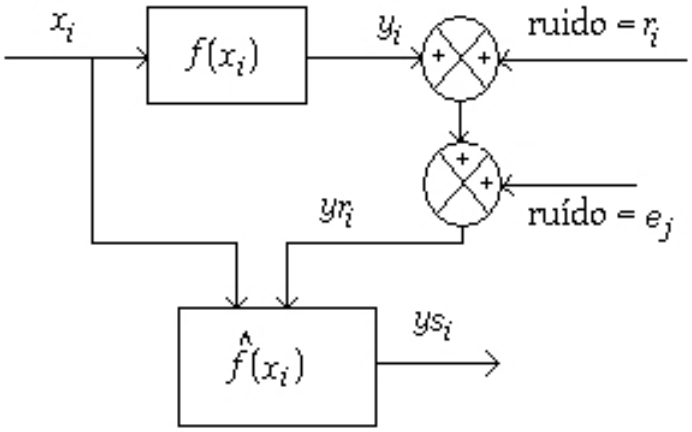

Figura 3.2: Diagrama de blocos com injeção de dois ruídos com funções de densidade de probabilidade distintas.

o ruído e_j, estaremos aproximando o Problema 2 do Problema 1 e desta forma, os dois problemas poderão ser tratados da mesma maneira.

Segundo propriedade de variáveis aleatórias, dada a soma entre duas variáveis aleatórias independentes $z = a + b$, sendo $h(a)$ a função de densidade de probabilidade (pdf) de a e $g(b)$ a pdf de b, podemos dizer que a pdf de z, $k(z)$, será dada pela convolução entre a pdf de a e de b, ou seja:

$$k(z) = h(a) * g(b). \tag{3.1}$$

Assim sendo, a soma de dois sinais de ruídos com pdf distintas, vai gerar uma nova variável aleatória [29], ou seja, a soma de um ruído com distribuição gaussiana com um ruído de distribuição aleatória produz um novo ruído, com uma nova distribuição.

O método da Janela Dinâmica

O método da Janela procura capacitar os algoritmos de treinamento supervisionados, como o do gradiente descendente, a lidarem com ambos os conjuntos de dados, o do Problema 1 e o do Problema 2, e tem como principais objetivos:

- Evitar que a rede aprenda ruídos com chegada aleatória,
- ser capaz de identificar durante o treinamento os dados que possam causar distúrbio excessivo no ajuste dos pesos,
- servir como elemento adicional para o processo de regularização das redes neurais, já que as informações dos espúrios não serão completamente absorvidas,

- poder ser aplicado em várias arquiteturas de rede,
- ser simples e de baixo custo computacional.

A proposta deste método é construir, durante o treinamento, um intervalo que avalie se o erro quadrático gerado a partir da saída da rede, como resposta a um vetor de entrada, é adequado para o treinamento. As saídas cujo erro quadrático não estiverem dentro deste intervalo não devem ser utilizadas para o ajuste dos pesos. O efeito que esperamos obter é uma aproximação maior dos pontos que estiverem dentro da faixa permitida.

Um aspecto interessante para a região de permissividade é que ela possa ser ajustada dinamicamente, de forma que, à medida em que o tempo passa (entenda como tempo a execução de uma época) e a rede neural for incorporando as características médias da função objetivo, o tamanho da Janela possa ser reduzido até um ponto de interesse.

3.2 Implementação

Para utilizarmos o método da Janela devemos considerar que, inicialmente a rede desconhece totalmente os dados e os pesos estão distribuídos aleatoriamente, formando uma superfície qualquer. Portanto, é necessário responder a alguns questionamentos:

- Como a Janela vai ser reduzida ao longo do tempo?
- Qual é o valor inicial da abertura da Janela?
- Qual o valor final da abertura da Janela?

Aspectos da Janela

Decaimento Gradual da Janela:

Para estabelecermos uma relação entre o valor inicial e final de abertura da Janela e a época atual e final do treinamento, podemos utilizar uma equação que relaciona esses quatro parâmetros ao mesmo tempo. Uma alternativa interessante é a equação do decaimento exponencial [12], inicialmente proposta para variação da taxa de aprendizado, foi adaptada aqui para o decaimento da Janela, conforme Equação (3.2), e possui a forma gráfica da Figura 3.3.

$$\psi = \psi_o \left(\frac{\psi_f}{\psi_o} \right)^{\frac{epoca}{epoca_{max}}}, \tag{3.2}$$

onde:
ψ_o = valor inicial da abertura da Janela,
ψ_f = valor final da abertura da Janela,
ψ = valor da Janela num instante qualquer,
epoca = época atual,
$epoca_{max}$ = número máximo de épocas permitido.

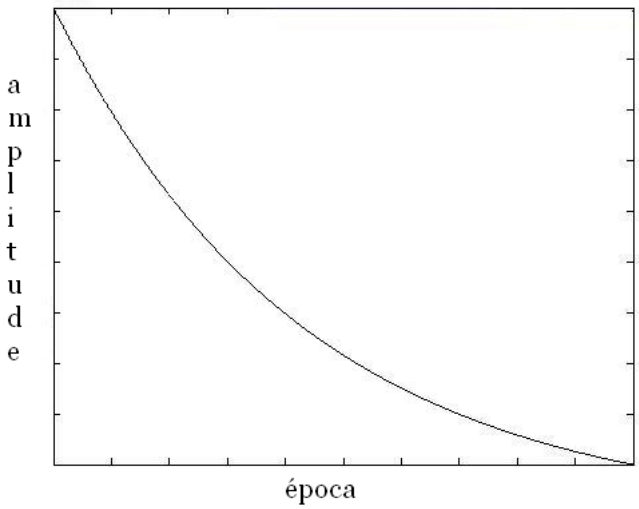

Figura 3.3: Forma do decaimento exponencial dado pela Equação 3.2

Determinando o valor de abertura inicial, ψ_o

Como queremos que a rede aprenda com a base de dados, o ψ_o deve ser tal que permita que todos os pontos possam ser utilizados durante as primeiras etapas de ajuste dos pesos. Para isso, antes do treinamento ser iniciado, devemos encontrar o maior erro quadrático, e^2_{max}, gerado pelo conjunto de treinamento e fazer

$$\psi_o \geq e^2_{max}. \tag{3.3}$$

Modo 1: Uma maneira de atender a inequação (3.3) é escolhendo um valor $N \geq 1$, sendo $N \in \Re$. Desta forma teremos:

$$\psi_o = Ne^2_{max}. \tag{3.4}$$

A principal vantagem deste modo é que ele é bem simples, necessitando apenas que determinemos o valor de N. A desvantagem é que não há uma escolha direta do momento em que ψ iguala-se ao e^2_{max} inicial. Esse momento é importante porque ele nos fornece uma expectativa de quanto tempo a rede terá para treinar, sem ser perturbada pela Janela. Podemos manipular a Equação (3.2) para obter:

$$\frac{\psi}{\psi_o} = \left(\frac{\psi_f}{\psi_o}\right)^{\frac{epoca}{epoca_{max}}}. \tag{3.5}$$

Aplicando o logaritmo neperiano em ambos os lados da Equação 3.5 teremos:

$$ln\left(\frac{\psi}{\psi_o}\right) = ln\left(\left(\frac{\psi_f}{\psi_o}\right)^{\frac{epoca}{epoca_{max}}}\right), \tag{3.6}$$

$$ln\left(\frac{\psi}{\psi_o}\right) = \frac{epoca}{epoca_{max}} ln\left(\frac{\psi_f}{\psi_o}\right). \tag{3.7}$$

Considerando que $\psi = e^2_{max}$ e substituindo o ψ_o e ψ na Equação (3.7) obtêm-se:

$$ln\left(\frac{e^2_{max}}{Ne^2_{max}}\right) = \frac{epoca}{epoca_{max}} ln\left(\frac{\psi_f}{Ne^2_{max}}\right). \tag{3.8}$$

Por fim, manipulando a Equação 3.8 obteremos a relação entre a época do encontro de ψ e e^2_{max} inicial com a época máxima de treinamento, conforme:

$$epoca = epoca_{max} \frac{-ln(N)}{ln(\psi_f) - ln(Ne^2_{max})}. \tag{3.9}$$

Podemos ver que na Equação (3.9) existe uma dependência relacionada ao e^2_{max} inicial, cujo valor é determinado pela inicialização dos parâmetros da rede e pelo próprio conjunto de dados, impedindo a determinação antecipada de N e da época desejada.

Modo 2: A segunda técnica utilizada procura explicitar melhor o momento do encontro entre ψ e o e^2_{max} inicial, através da Equação (3.2). Podemos dizer que a época escolhida é representada por uma relação $\frac{a}{b}$ do número total de épocas, com $a \in \aleph$, $b \in \aleph$ - 0, e $a < b$. Manipulando a Equação (3.2) temos:

$$\psi = \psi_o \left(\frac{\psi_f}{\psi_o}\right)^{\frac{aepoca_{max}}{bepoca_{max}}}, \tag{3.10}$$

$$\psi = \psi_o \left(\frac{\psi_f}{\psi_o}\right)^{\frac{a}{b}}, \tag{3.11}$$

$$\psi = \psi_o \left(\frac{\psi_f^{\frac{a}{b}}}{\psi_o^{\frac{a}{b}}}\right), \tag{3.12}$$

colocando em termos de ψ_o

$$\psi_o^{\frac{b-a}{b}} = \frac{\psi}{\psi_f^{\frac{a}{b}}}, \tag{3.13}$$

$$\psi_o = \sqrt[b-a]{\frac{\psi^b}{\psi_f^a}}. \tag{3.14}$$

Considerando agora que $\psi = e_{max}^2$, podemos determinar o valor inicial de abertura da Janela substituindo ψ por e_{max}^2 na Equação (3.14). A expressão final será:

$$\psi_o = \sqrt[b-a]{\frac{{e_{max}^2}^b}{\psi_f^a}}. \tag{3.15}$$

Determinando o ψ_f

Já o parâmetro ψ_f deve ser escolhido pelo usuário e corresponderá ao maior e_{max}^2 permitido como resposta da rede a um dado de entrada na última época de treinamento. Isso pode ser feito de acordo com algum conhecimento a priori que o projetista tenha dos dados, ou um valor que seja aceitável para uma aplicação específica, ou por meio de alguma heurística conveniente para o projetista.

A Influência da Janela no Algoritmo de Retropropagação

O algoritmo de retropropagação é uma generalização do algoritmo do mínimo quadrado médio (LMS). Este algoritmo é dividido em duas etapas: *o passo para frente* e *o passo para trás*. No passo para frente obtemos a resposta da rede a um vetor de entrada e no passo para trás, os pesos são ajustados de

acordo com alguma regra relacionada ao sinal de erro gerado pela diferença entre a saída e o valor desejado, conforme:

$$e(t) = d(t) - s(t), \tag{3.16}$$
$$e(t) = d(t) - X(t)F(w(t)). \tag{3.17}$$

onde para uma amostra t temos:
$e(t)$ = erro de saída,
$d(t)$ = valor desejado,
$s(t)$ = valor de saída da rede,
$X(t)$ = vetor de entrada,
$F(w(t))$ = função relacionando todos os pesos da rede.

Correspondentemente, o valor instantâneo da energia total do erro, $\xi(t)$, é obtido somando-se o erro de cada amostra, conforme a equação:

$$\xi(t) = \frac{1}{2}\sum e^2(t). \tag{3.18}$$

O algoritmo de retropropagação aplica uma correção Δw(t) nos pesos, proporcional à derivada parcial $\partial\xi(t)/\partial$w(t) definida pela regra delta:

$$\Delta w(t) = -\eta\frac{\partial\xi(t)}{\partial w(t)}, \tag{3.19}$$
$$= \eta e(t)\varphi'(t)s(t). \tag{3.20}$$

sendo:
η = taxa de aprendizado,
$\varphi'(t)$ = derivada da função de ativação.

A derivação completa das equações acima pode ser vista em [14][1].

Para aplicar o método da Janela, é necessário acrescentar o parâmetro υ_ψ na regra delta:

$$\Delta w(t) = \eta\upsilon_\psi e(t)\,\varphi'(t)\,s(t). \tag{3.21}$$

O parâmetro υ_ψ corresponde à situação do erro quadrático da amostra t em relação à Janela e é definido de acordo com a regra:

$$\upsilon_\psi = \begin{cases} 1 & \text{se } e^2(t) \leq \psi, \\ 0 & \text{caso contrário.} \end{cases}$$

[1]No capítulo 4, da página 183 à página 194.

É importante salientar que podemos incorporar o η ao υ_ψ, substituindo o um na regra anterior pelo η e eliminando este da regra delta. Porém preferimos manter os dois parâmetros em separado já que o η pode ser entendido como um elemento que ajusta o quanto a rede deve aprender numa determinada época, enquanto que o υ_ψ nos diz quem tem o direito de ajustar a rede durante a excecução da época.

Implementação em Software

Para implementar o método da Janela no treino de uma rede neural são necessários alguns poucos passos, conforme listado a seguir:

- Adicionar a variável ψ, correspondente *à* abertura da Janela.
- Adicionar a variável υ_ψ, que irá afetar a regra de correção dos pesos.
- Definir o valor de ψ_f.
- Definir o valor de N, se usando o Modo 1, ou o valor de a e b se usando o Modo 2.
- Passar os dados pela rede sem treinar, para estabelecer o valor de ψ_o.
- Implementar a equação (3.2).
- Acrescentar a variável υ_ψ na regra delta .
- Alterar, para cada amostra, o valor de υ_ψ de acordo com o e^2(t) e ψ.

3.3 Resumo

Neste capítulo foi apresentada a teoria que envolve o método da Janela, quais são seus principais parâmetros e como fazer para implementar em redes neurais baseadas no neurônio tipo perceptron.
Dentre as principais vantagens deste método citamos:

- Facilidade de implementação e interpretação.
- Não necessita de nenhum conhecimento da superfície a ser aprendida, nem do ruído que a envolve. O usuário define o quanto de ruído é aceitável como resposta de saída da rede para o problema.
- Custo computacional baixo.

- Funciona com regularizador, uma vez que os dados de treinamento estão limitados à uma determinada região em torno da superfície gerada pela rede, reduzindo a influência que as idiossincrazias de ruídos fora da região limite impões à rede.

Capítulo 4

Testes e resultados

4.1 Introdução

Neste capítulo apresentamos alguns dos casos de teste utilizados durante o processo de avaliação do método da Janela. Como forma de mostrar a facilidade de inserção do método em qualquer ambiente neural, o mesmo foi implementado em duas redes distintas: as redes lógicas adaptivas, ALN, e as redes perceptrons de múltiplas camadas, MLP. Ambas com a correção dos pesos baseado no algoritmo do gradiente descendente e procurando minimizar a função de custo do erro quadrático médio. No caso das redes MLP foram utilizadas apenas estruturas com uma única camada oculta.

Para os resultados mostraremos os dados de saída referentes a dois perceptrons de múltiplas camadas, MLP - E1 e MLP - E2, cuja diferença está no número de neurônios ocultos e os dados de saída da ALN usando o software ALNGD. Essas duas estruturas MLP procuram simular uma rede bem dimensionada, com o número de neurônios suficientes para o problema, e uma rede mal dimensionada, com excesso de neurônios. No caso da ALN isso não foi necessário, já que nela foi implementado um algoritmo de crescimento dinâmico de rede.

Quando as redes estiverem usando o método da Janela, serão acrescentados os termos JG ao final do nome, em referência a redução gradual da abertura da Janela.

Como uma das principais características de uma rede neural é a capacidade de generalização, o conjunto de dados foi separado em dados para treinamento e dados para teste. Apresentaremos os resultados gráficos e numéricos para ambos os casos, como forma de permitir uma melhor avaliação do comportamento geral da rede.

O treinamento de ambas as arquiteturas foi feito em softwares específicos

para cada uma delas, mais especificamente, o software ALNGD, implementado em Java por este autor e o software Flood, implementado em C++ como biblioteca livre por Roberto Lopez, da Universidade Politécnica da Catalunha, UPC [20]. Ao Flood foi incorporado o método proposto, conforme explicado no Capítulo 3. Por fim, os resultados gráficos e numéricos foram gerados no ambiente Matlab, do qual ainda foram aproveitadas as implementações de rede neural baseadas nos algoritmos de Levenberg-Marquardt [39], [21] e Regularização Bayesiana [22], implementada conforme [11]. Esta mudança de algoritmo é feita através da passagem dos parâmetros TRAINLM e TRAINBR durante a criação da rede[1].

4.1.1 Medidas usadas para avaliação dos resultados

Para avaliar o desempenhos das redes em cada problema existem várias formas já estabelecidas no meio acadêmico e profissional. Dentre eles optamos por utilizar:

- A visualização gráfica,
- a raiz quadrada do erro quadrático médio - RMSE,
- o RMSE normalizado - NRMSE,
- o coeficiente de U-Theil, para o caso de previsão de séries temporais.

A visualização gráfica

Nesta abordagem são gerados gráficos dos resultados. Normalmente estes gráficos são agrupados em uma mesma figura, mas optamos por colocá-los em figuras dispostas lado a lado, a fim de termos uma boa separação visual dos resultados obtidos em cada abordagem. Quando não explicitado, os dados após a rede neural são apresentados em vermelho.

A raiz quadrada do erro quadrático médio - RMSE

O erro quadrático médio, MSE, representa a variabilidade média absoluta dos valores obtidos em relação aos valores alvo. O RMSE nada mais é do que a raiz quadrada do MSE. O RMSE pode ser calculado da seguinte maneira:

[1] Nossa intenção inicial era comparar apenas os resultados da regularização gerada pelo TRAINBR com os resultados da Janela, mas como optamos por mostrar os dados sem e com Janela, ou seja, sem e com regularização, preferímos manter o padrão e apresentar também os resultados do TRAINLM, uma vez que o TRAINBR é um TRAINLM regularizado.

$$RMSE = \sqrt{\frac{\sum_{t=1}^{N} (yd_t - ys_t)^2}{N}}, \tag{4.1}$$

onde:
N = número de amostras,
yd_t = valor desejado para a amostra t
ys_t = valor de saída da rede para a amostra t

O RMSE normalizado - NRMSE

O NRMSE é uma medida de erro relativa entre dois conjuntos de dados, desenvolvida para ser independente da ordem de grandeza desses conjuntos e é muito útil quando se quer determinar com clareza, o quão grande ou pequeno é o resultado.

Se o valor do NRMSE for a unidade, podemos dizer que a resposta está dentro da média do conjunto desejado, por outro lado, se for zero, significa uma resposta perfeita dos valores alvo. O NRMSE pode ser calculado conforme a Equação (4.2):

$$NRMSE = \frac{\sqrt{\frac{\sum_{t=1}^{N}(yd_t - ys_t)^2}{N}}}{\sqrt{\frac{\sum_{t=1}^{n}\left(yd_t - \overline{yd_m}\right)^2}{N}}}, \tag{4.2}$$

onde:
N = número de amostras,
yd_t = valor desejado para a amostra t,
ys_t = valor de saída da rede para a amostra t,
$\overline{yd_m}$ = média do valor desejado das amostras.

Coeficiente U de Theil

É uma métrica que mede o quanto os resultados estão melhores que uma previsão trivial[2] que diz que: "a melhor estimativa para o preço de amanhã é o preço de hoje"[1].

A equação para o cálculo do U de Theil é:

[2]Também conhecida como previsão ingênua.

$$U = \frac{\sqrt{\sum_{t=1}^{n} (alvo_t - pred_t)^2}}{\sqrt{\sum_{t=1}^{n} (alvo_t - alvo_{t-1})^2}}, \tag{4.3}$$

onde:
$alvo_t$ = valor desejado no instante t,
$alvo_{t-1}$ = valor imediatamente anterior ao $alvo_t$,
$pred_t$ = valor de saída do sistema previsor no instante t.

A cerca do resultado de 4.3 podemos dizer que:

- Quando $U \geq 1$, temos um indicador de que o erro da previsão das redes é maior que o erro da previsão trivial, ou seja, os resultados são considerados ruins,
- quando $U < 1$, mesmo que por uma quantidade pequena[3], as previsões são boas, pois o erro do numerador foi menor que o do denominador, mostrando que o resultado da rede é melhor do que a previsão trivial.

4.1.2 Variação de parâmetros

Os principais parâmetros que foram alterados durante os testes foram:

- MLP
  - Taxa de Aprendizado,
  - número de Épocas,
  - número de Neurônios Ocultos,
  - valor Final da Janela.
- ALN
  - Taxa de Aprendizado,
  - número de Épocas,
  - valor Final da Janela.

A escolha dos parâmetros foi realizada de forma heurística e não envolveu nenhum método determinístico para isso. Os resultados que serão apresentados foram obtidos após comparação das saídas de vários casos de teste e serão mostrados somente os mais comuns e significativos.

[3]Segundo [36] somente valores menores ou iguais a 0,55 são considerados confiáveis.

4.2 Análise da influência da escolha do ψ_o

Estamos agora interessados em avaliar a influência da Janela e da escolha do ψ_o na resposta da rede neural, conforme a Equação (3.15). Para isso, manteremos fixos o número de neurônios ocultos, a taxa de aprendizado, o número de épocas e o valor final de abertura da Janela ψ_f, enquanto variamos o ψ_o através da modificação dos parâmetros a e b apresentados na Equação (3.15).

4.2.1 Chamadas internacionais na Bélgica

Para esta análise utilizaremos um conjunto de dados proveniente de uma pesquisa estatística feita na Bélgica, publicada pelo Ministério da Economia, contendo o número de chamadas internacionais, em dezenas de milhões, que foram feitas naquele país de 1950 a 1973 [35] e apresentados na Figura 4.1.

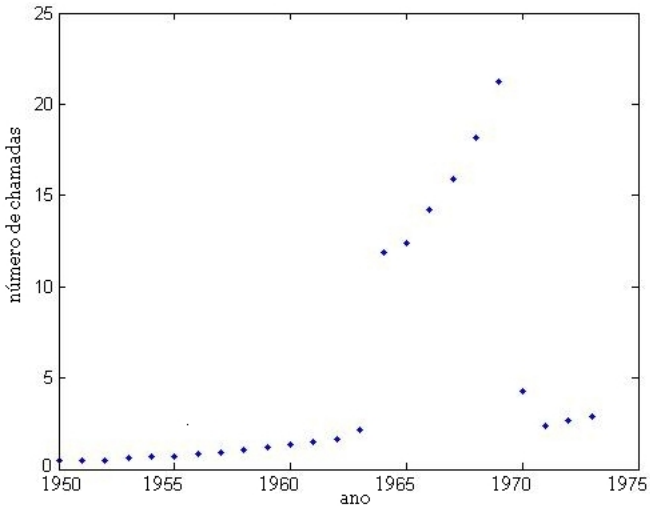

Figura 4.1: Gráfico do número de chamadas internacionais da Bélgica.

O gráfico mostra o crescimento do número de chamadas ao longo dos anos, contudo há um aumento exagerado no ano de 1964 e uma diminuição brusca em 1969. Esses dois acontecimentos revelam o erro de medição que ocorreu durante 6 anos seguidos, de 1964 a 1969, e que afetou também parte dos anos de 1963 e 1970. Esse erro na base de dados ocorreu após a mudança do sistema utilizado para gravação dos dados estatísticos. Descobriu-se, após inquérito, que o novo sistema devolvia o número total de minutos das chamadas ao invés do número de chamadas. Os anos de 1963 e 1970 só foram

parcialmente afetados porque a transição do sistema de gravação e a correção do equívoco não ocorreram exatamente no ano novo.

Os resultados apresentados na Figura 4.2 e nas Tabelas 4.1, 4.2, 4.3 e 4.4 representam quatro diferentes configurações de treinamento, nas quais foram alterados ora o número de neurônios ocultos, ora a taxa de treinamento e ora o número total de épocas. As medições do RMSE e o NRMSE foram feitas em relação ao conjunto de dados original e em relação a um conjunto de dados teórico, obtido por regressão linear de primeira ordem após a retirada dos *outliers*.

Podemos ver na Figura 4.2 que o aumento do valor de ψ_o produz resultados mais próximos daquele que é produzido pela rede tradicional sem Janela. Porém, esse efeito não é necessariamente linear. Conforme Tabelas 4.1 a 4.4 o valor do NRMSE relativo aos dados contaminados não é cada vez menor a medida que aumentamos o ψ_o e isso se deve a três fatores principais: o primeiro é devido às características peculiares do conjunto de dados, poucos pontos e com transições abruptas, e o segundo é o fato da rede se ajustar lentamente ao conjunto de dados e o terceiro é o valor baixo do ψ_f. Esses fatores combinados produzem um aumento no valor do NRMSE após sucessivas quedas. Esse efeito que pode ser visto por exemplo na Tabela 4.1 nas linhas de 75% e 83.3%. O acréscimo é ocasionado pelo maior tempo que a rede permanece utilizando os poucos pontos do final do conjunto de dados, que "puxam"os resultados para valores mais próximos da linha de regressão e afasta-os dos pontos ruidosos muito acima do alcance da Janela.

			Dados Contaminados		Dados da Regressão	
a	b	% $epoca_{max}$	RMSE	NRMSE	RMSE	NRMSE
0	7	0	6,8207	1.0632	0.3549	0,5066
1	7	14,3	6,7881	**1,0581**	0,3548	**0,5065**
1	4	25	3,8526	0,6005	4,3998	6,2803
1	2	50	3,0905	0,4818	5,3713	7,6669
3	4	75	3,2490	0,5064	5,3022	7,5683
5	6	83,3	3,3904	0,5285	5,2453	7,4871
sem Janela			1,9031	0,2967	6,7467	9,6301

Tabela 4.1: **Primeiro Caso:** Resultado para 15.000 épocas, taxa de aprendizado 0,0005 e estrutura 1 - 5 - 1.

			Dados Contaminados		Dados da Regressão	
a	b	% $epoca_{max}$	RMSE	NRMSE	RMSE	NRMSE
0	7	0	6,6932	1,0433	0,4081	0,5825
1	7	14,3	6,7189	**1,0473**	0,3845	**0,5488**
1	4	25	6,7461	1,0516	0,3822	0,5456
1	2	50	6,7090	1,0458	0,4132	0,5898
3	4	75	6,5894	1,0272	0,5554	0,7928
5	6	83,3	4,9448	0,7708	4,2320	6,0407
sem Janela			4,4895	0,6998	5,3001	7,5652

Tabela 4.2: **Segundo Caso:** Resultado para 15.000 épocas, taxa de aprendizado 0,0001 e estrutura 1 - 5 - 1.

			Dados Contaminados		Dados da Regressão	
a	b	% $epoca_{max}$	RMSE	NRMSE	RMSE	NRMSE
0	7	0	6,8475	**1,0674**	0,2103	**0,3002**
1	7	14,3	6,7756	1,0562	0,3089	0,4409
1	4	25	6,6967	1,0439	0,4041	0,5768
1	2	50	6,1483	0,9584	1,2524	1,7876
3	4	75	5,3097	0,8277	3,9174	5,5917
5	6	83,3	5,0888	0,7932	4,3377	6,1915
sem Janela			4,6266	0,7212	5,1816	7,3962

Tabela 4.3: **Terceiro Caso:** Resultado para 5.000 épocas, taxa de aprendizado 0,0001 e estrutura 1 - 5 - 1.

			Dados Contaminados		Dados da Regressão	
a	b	% $epoca_{max}$	RMSE	NRMSE	RMSE	NRMSE
0	7	0	6,6046	**1,0295**	0,4911	**0,7010**
1	7	14,3	6,0134	0,9374	1,1012	1,5718
1	4	25	4,4993	0,7014	4,7849	6,8298
1	2	50	4,5625	0,7112	4,4709	6,3816
3	4	75	3,6734	0,5726	5,2940	7,5566
5	6	83,3	3,7638	0,5867	4,5639	6,5144
sem Janela			2,3933	0,3731	6,5437	9,3404

Tabela 4.4: **Quarto Caso:** Resultado para 15.000 épocas, taxa de aprendizado 0,0001 e estrutura 1 - 15 - 1.

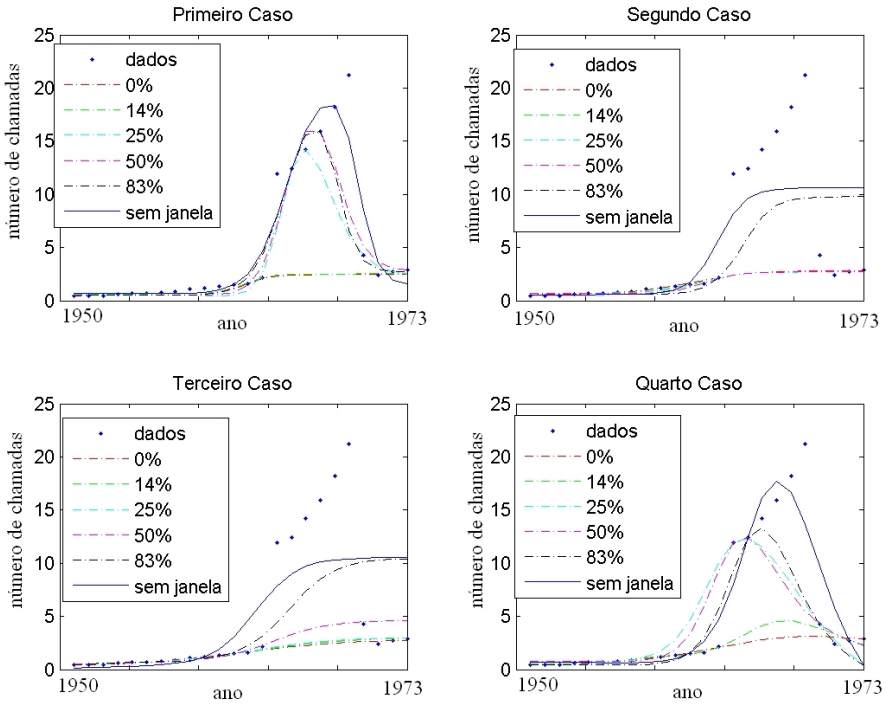

Figura 4.2: Visualização gráfica do efeito que a escolha dos parâmetros **a** e **b** provoca no resultado final da rede.

4.2.2 Filtragem de ruído

Neste experimento procuramos verificar a influência da Janela nos resultados de um filtro neural. O sinal limpo utilizado é fornecido pela Equação (4.4) e pode ser visualizado na Figura 4.3. Inicialmente, investigaremos o efeito produzida pela Janela em uma situação de ruído com distribuição normal somente, posteriormente, injetaremos espúrios no conjunto de dados.

$$x(n) = sen(n + sen(n^2)), \quad n \in \Re. \tag{4.4}$$

No início dos testes, a escolha do ψ_o foi baseada na determinação de um número N que atenda a Equação (3.4) e o ruído gaussiano injetado foi atenuado a 25% do seu valor normal, de forma a obtermos um resultado visual mais nítido. A Tabela 4.5 nos mostra as estatísticas do sinal limpo e contaminado associadas ao experimento.

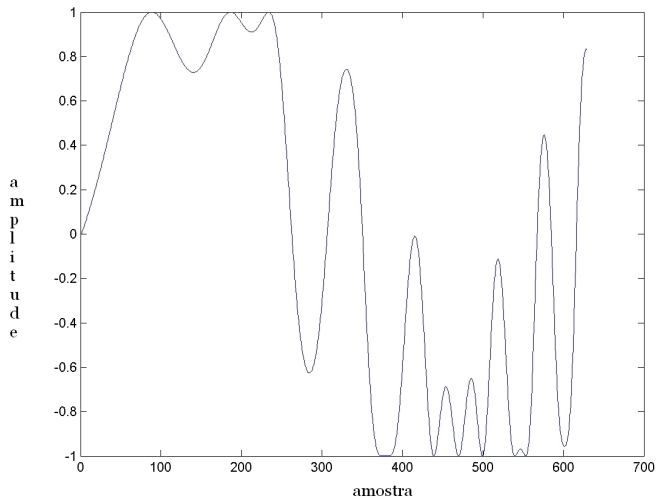

Figura 4.3: Forma de onda produzida pela Equação (4.4) no intervalo $[0, 2\pi]$.

Tipo de sinal	Média - μ	Variância - σ^2
Sinal Limpo	0,0684	0,5376
Sinal Contaminado	0,0848	0,5839

Tabela 4.5: Estatísticas de média e variância associadas ao experimento om ruído gerado pela função *wnoise* do Matlab.

Considere para este experimento os seguintes parâmetros:

- Número de unidades de atraso: 5.
- $\psi_o = 3e^2_{max}$.
- $\psi_f = 0{,}01$.
- Número de épocas MLP = 5000.
- Número de épocas ALN = 150.
- Taxa de aprendizado = 0,0001.

Pode-se verificar pela Tabela 4.6 que a Janela piora o resultado de todas as redes, fato visto pelo aumento no valor do NRMSE em relação ao sinal limpo. Apesar disso, no caso da rede ALN há um ganho a se considerar, que é a redução do número de neurônios utilizados. Podemos ver os resultados gráficos na Figura 4.4. Observa-se também que, a rede treinada com regularização bayesiana produz o mesmo comportamento, quando comparada com a implementação usando Levenberg-Marquadt.

		Sinal Contaminado		Sinal Limpo			
Rede	Estrutura	RMSE	NRMSE	RMSE	NRMSE	Média	Variância
ALN	54 folhas	0,1869	0,2448	0,0917	**0,1251**	0,0759	0,5018
ALN - JG	**10 folhas**	0,1827	0.2393	0,0956	0,1305	0,0743	0,5030
MLP - E1	1 - 5 - 1	0,1820	0.2384	0,0922	**0,1259**	0,0688	0,5293
MLP - E1JG	1 - 5 - 1	0,1817	0,2380	0,0931	0,1271	0,0689	0,5295
MLP - E2	1 - 15 - 1	0,1878	0,2459	0,0836	**0,1141**	0,0689	0,5310
MLP - E2JG	1 - 15 - 1	0,1876	0,2458	0,0843	0,1150	0,0685	0,5309
TRAINLM	1 - 15 - 1	0,1942	0,2544	0,0669	**0,0912**	0,0689	0,5336
TRAINBR	1 - 15 - 1	0,1860	0,2437	0,0876	0,1195	0,0689	0,5300

Tabela 4.6: Estatísticas após treinamento para o filtro.

Nos testes de generalização os resultados foram similares ao do treinamento, portanto, omitiremos as Tabelas e Figuras correspondentes a estes testes e passaremos a investigar o efeito provocado pela Janela.

Considere agora que o mesmo sinal está contaminado por um ruído gaussiano não escalonado, conforme Figura 4.5. O filtro será analisado considerando as seguintes características:

- Número de unidades de atraso: 5,

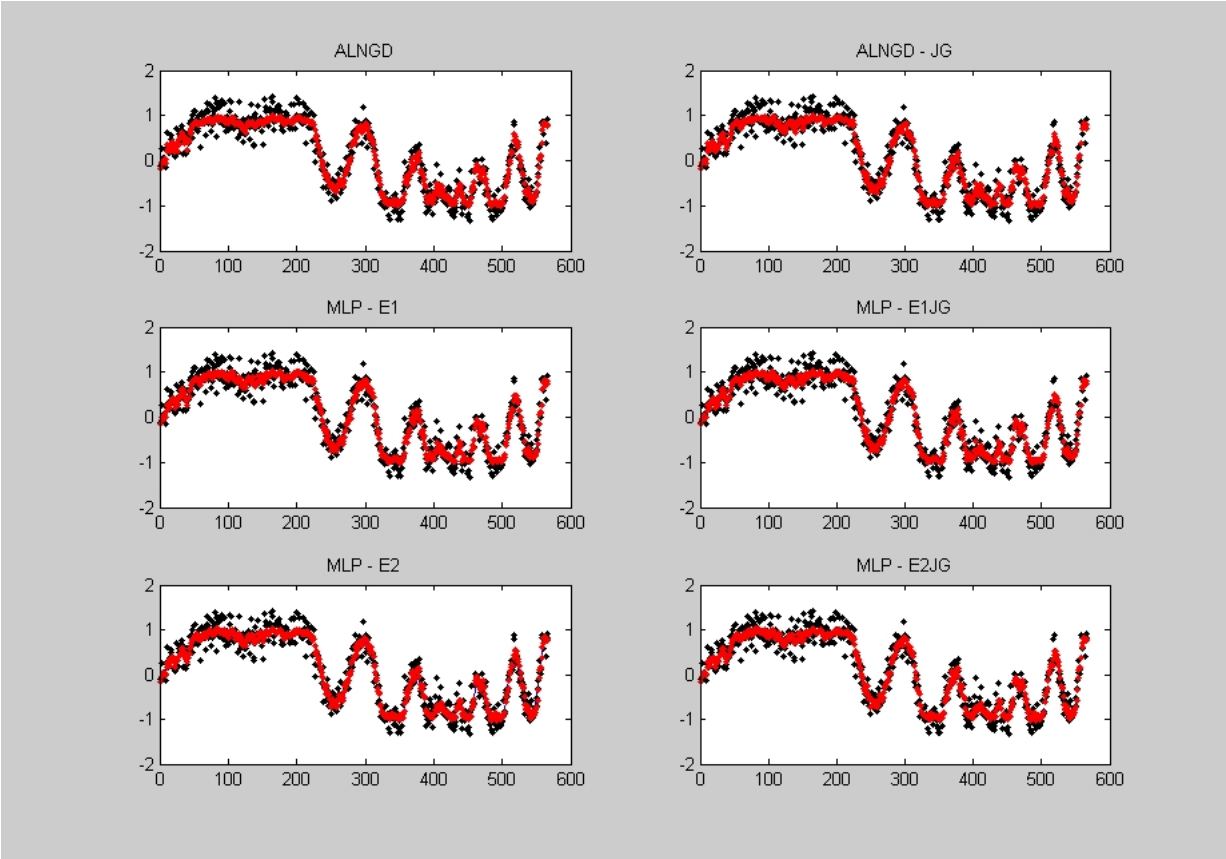

Figura 4.4: Resultado do treinamento. O eixo vertical representa a amplitude do sinal relativa a amostra dada no eixo horizontal.

- ψ_o determinado pela Equação (3.15),
- $\psi_f = 0{,}04$,
- número de épocas MLP $= 5000$,
- taxa de aprendizado $= 0{,}0005$,
- número de neurônios ocultos $= 5$.

Na Tabela 4.7 são apresentados os resultados do treinamento relativos ao sinal desejado, enquanto que, a Tabela 4.8 contém os resultados de teste. Podemos ver que, a redução na relação a/b produz um aumento no número de elementos com erro quadrático abaixo do 0,04 estabelecido no treinamento, o qual chamaremos de $N\psi$. Contudo, o NRMSE também aumenta, significando uma menor precisão. A primeira vista este parece ser um resultado ruim, mas

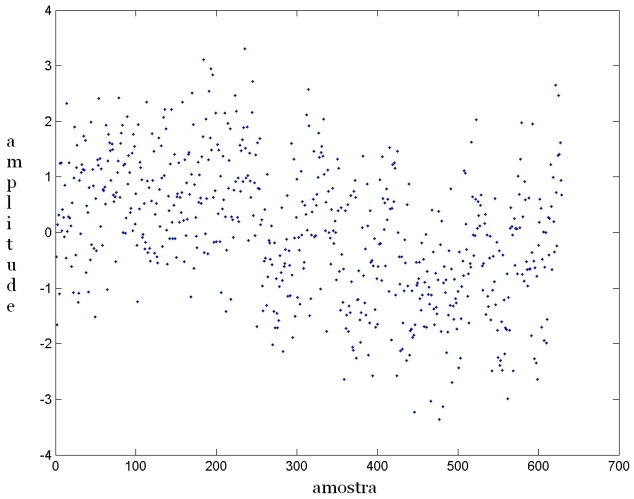

Figura 4.5: Sinal de entrada contaminado com ruído gaussiano.

esse é o efeito Janela esperado, já que nesse conjunto de dados não existem ruídos conforme descrito no Problema 2 do Capítulo 3, portanto, os dados eliminados pertencem ao conjunto gaussiano e a aproximação acontece para um conjunto menor de dados.

a/b	$N\psi$	Maior Erro	Menor Erro	RMSE	NRMSE
sem Janela	(360 pontos) 57,23 %	0,9405	-1,1695	0,3030	**0,4136**
20/21	**(387 pontos)** 61,53%	0,9754	-1,2383	0,3132	**0,4275**
5/6	(387 pontos) 61,53%	1,0404	-1,2528	0,3199	0,4367
1/2	**(397 pontos)** 63,12%	1,1566	-1,2787	0,3323	0,4536
0/1	(396 pontos) 62,96 %	1,2086	-1,2982	0,3446	0,4703

Tabela 4.7: Resultados do filtro neural, implementado para retirar um ruído gaussiano. O sinal encontra-se no intervalo $[0, 2\pi]$.

Na etapa de testes, percebeu-se novamente a influência da Janela no valor final do NRMSE, ficando a rede sem Janela com a melhor aproximação geral. Entretanto, percebemos novamente que a Janela produz um aumento no número de pontos com erro quadrático abaixo dos 0,04, contudo, diferentemente do treinamento, quanto maior a abertura inicial, maior é o número de pontos. Esse efeito ocorre devido a uma combinação das características do conjunto de dados, com uma melhor capacidade de generalização das redes com maior relação a/b.

a/b	$N\psi$	Maior Erro	Menor Erro	RMSE	NRMSE
sem Janela	(312 pontos) 49,60 %	1,2434	-1,1178	0,3814	0,5409
20/21	(343 pontos) 54,53%	1,3265	-1,2179	0,3956	0,5610
5/6	(337 pontos) 53,58%	1,3695	-1,2541	0,4055	0,5751
1/2	(318 pontos)50,56%	1,4157	-1,2757	0,4244	0,6018
0/1	(313 pontos) 49,76%	1,4423	-1,3225	0,4384	0,6217

Tabela 4.8: Resultados do teste para o filtro neural, implementado para retirar um ruído gaussiano. O sinal encontra-se no intervalo $[2\pi, 4\pi]$.

Vamos agora analisar um caso onde existem dois ruídos com distribuições diferentes: O primeiro terá distribuição normal e será injetado em todo o conjunto de dados, o segundo terá distribuição uniforme e afetará apenas 10% desse conjunto.

As tabelas 4.9 e 4.10 resumem as medidas feitas após o treinamento e teste das redes. Por estas tabelas podemos ver que as redes com a Janela produzem os melhores resultados, tanto no valor do NRMSE quanto na quantidade de elementos cujo erro ficou dentro da região desejada. Além disso, podemos perceber que o resultado melhora à medida que a relação $\frac{a}{b}$ cresce. O mesmo comportamento pode ser visto no teste. Isso evidencia que é necessário permitir que as informações contidas nos dados sejam aprendidas e, somente depois, é que os ruídos poderão ser eliminados.

a/b	$N\psi$	Maior Erro	Menor Erro	RMSE	NRMSE
sem Janela	(550 pontos) 87,44%	0,4244	-0,3893	0,1241	0,1694
20/21	**(577 pontos) 91,73%**	0,4240	-0,3353	0,1109	**0,1514**
5/6	(577 pontos) 91,77%	0,4262	-0,3267	0,1110	0,1515
1/2	(576 pontos)91,57%	0,4240	-0,3354	0,1113	0,1519
0/1	(575 pontos) 91,41%	0,4220	-0,3389	0,1114	0,1521

Tabela 4.9: Resultados do treinamento para o filtro neural, cujo sinal está contaminado por um ruído com distribuição normal e outro com distribuição uniforme.

a/b	$N\psi$	Maior Erro	Menor Erro	RMSE	NRMSE
sem Janela	(421 pontos) 66,93%	0,5940	-0,6403	0,2103	0,2982
20/21	**(430 pontos) 68,36%**	0,5947	-0,5842	0,1982	**0,2811**
5/6	(425 pontos) 67,57%	0,6034	-0,5866	0,2016	0,2859
1/2	(423 pontos)67,25%	0,6058	-0,5907	0,2049	0,2905
0/1	(419 pontos) 66,61%	0,6043	-0,5907	0,2058	0,2919

Tabela 4.10: Resultados do teste para o filtro neural, cujo sinal está contaminado por um ruído com distribuição normal e outro com distribuição uniforme.

4.2.3 Considerações

Podemos concluir que o valor escolhido para ψ_o é importante e capaz de alterar o resultado da rede quanto mais próximo for o ψ_o do e^2_{max} inicial.

Durante o processo de avaliação das redes com Janela esse valor foi fixado para gerar uma relação $epoca/epoca_{max}$ de 95% ou mais. Nos casos em que optamos pelo Modo 1 de escolha do ψ_o, pudemos determinar essa relação por meio da Equação (3.9).

4.3 Casos de teste

Os casos de teste utilizados para avaliação do método foram separados em:

- Aproximador de função,
- previsão,
- filtragem.

Nos casos de aproximador de função mostraremos os resultados obtidos sobre conjuntos de dados teóricos. Já para a previsão e filtragem, apresentaremos os resultados de conjuntos de dados reais, obtidos por meio de medições feitas em sensores.

4.3.1 Aproximador de função

Vejamos agora os resultados comparativos entre as redes com e sem o método da Janela, para uma rede neural configurada como aproximadora de função, conforme descrito no Capítulo 2.

Sem Ruído: Função Valor Absoluto do Seno

Para este caso, considere que a rede precisa aprender a função definida pela Equação 4.5. O conjunto de dados pertence ao intervalo [0, 2π] e as estatísticas associadas ao sinal desejado são apresentadas na Tabela 4.11.

$$y = |sin(\theta)|. \tag{4.5}$$

Tipo de sinal	Média - μ	Variância - σ^2
Sinal Limpo	0,6360	0,0952

Tabela 4.11: Estatísticas de média e variância associadas ao experimento y = |sin(θ)|.

Consideremos para este exemplo que a rede MLP seja configurada com os seguintes parâmetros:

- taxa de aprendizado = 0,0001,
- número de épocas = 35.000,
- ψ_f = 0,01,
- a/b = 20/21.

Na Tabela 4.12 podemos ver que a Janela melhorou os resultados medidos pelo NRMSE em 11,51% no caso da ALN, além da redução no número de neurônios, 4,39% no caso do MLP com 8 neurônios ocultos e 3,46% no caso do MLP com 15 neurônios ocultos. Esta melhora também pode ser observada nos resultados de teste, apresentados na Tabela 4.13. Na Figura 4.6 vemos que a diferença entre as redes está mais visível nas regiões em torno dos pontos $\pi/2$, π e $3\pi/2$. A ALNGD - JG procura suavizar mais as curvas nestes pontos em relação a ALNGD. Já a MLP - E1 consegue uma melhor aproximação do ponto π, mas acaba por se afastar da função nas laterais, o que não acontece com sua versão similar usando Janela. No caso da MLP - E2 e MLP - E2JG, o ponto $3\pi/2$ é o que mais se destaca e novamente, a versão com Janela produz um erro menor.

Rede	Estrutura	RMSE	NRMSE	Média	Variância
ALNGD	32 folhas	0,0086	0,0278	0,6361	0,0950
ALNGD - JG	**28 folhas**	0,0076	**0,0246**	0,6361	0,0951
MLP - E1	1 - 8 - 1	0,0302	0,0979	0,6262	0,0952
MLP - E1JG	**1 - 8 - 1**	0,0288	**0,0936**	0,6388	0,0918
MLP - E2	1 - 15 - 1	0,0330	0,1069	0,6453	0,0931
MLP - E2JG	1 - 15 - 1	0,0318	0,1032	0,6387	0,0920

Tabela 4.12: Estatísticas do treinamento do sistema y = $|\sin(\theta)|$.

Rede	Estrutura	RMSE	NRMSE	Média	Variância
ALNGD	32 folhas	0,0091	0,0296	0,6353	0,0964
ALNGD - JG	**28 folhas**	0,0090	**0,0292**	0,6354	0,0969
MLP - E1	1 - 8 - 1	0,0298	0,0963	0,6247	0,0976
MLP - E1JG	**1 - 8 - 1**	0,0282	**0,0913**	0,6372	0,0944
MLP - E2	1 - 15- 1	0,0323	0,1043	0,6440	0,0952
MLP - E2JG	1 - 15 - 1	0,0312	0,1010	0,6375	0,0939

Tabela 4.13: Estatísticas de teste do sistema y = $|\sin(\theta)|$.

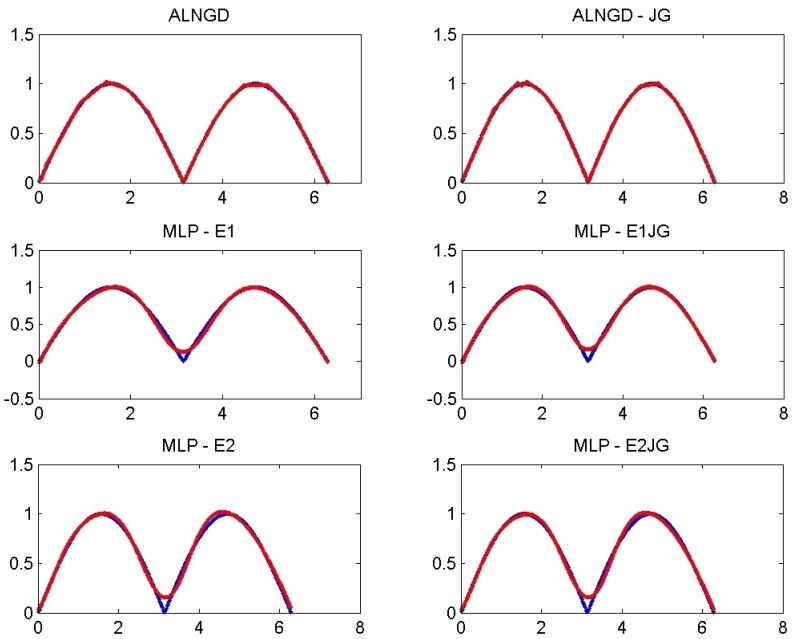

Figura 4.6: Comparação gráfica dos resultados de saída para o experimento y = $|\sin(\theta)|$. O eixo vertical representa o y e o eixo horizontal o θ em radianos.

Com Ruído: Equação de Reta Contaminada

Neste caso utilizaremos um sistema que gera um sinal de saída correspondente à equação y = 3x limitado ao intervalo]-5, 5[e que depois passa por um agente contaminante até tomar a forma da Figura 4.7 e possuir estatísticas de média e variância conforme apresentadas pela tabela 4.14.

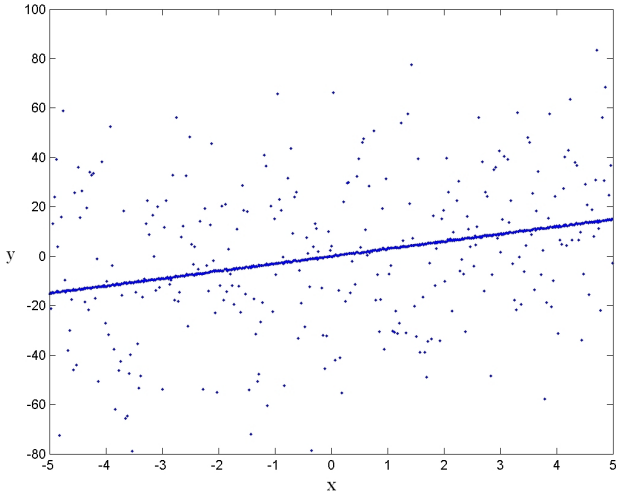

Figura 4.7: Gráfico do sinal y = 3x após contaminação.

Tipo de sinal	Média - μ	Variância - σ^2
Sinal Limpo	0	75,0819
Sinal Contaminado	0,1890	368,5605

Tabela 4.14: Estatísticas de média e variância associadas ao experimento y = 3x.

O objetivo é que a rede neural possa encontrar o sinal limpo, com base na apresentação dos exemplos contaminados. Primeiramente discutiremos os resultados sobre os dados de treinamento e depois sobre o conjunto de teste. Para avaliarmos o desempenho do método da Janela iremos comparar as estruturas duas a duas, separadas por tamanho, caso das MLPs, ou por tipo de rede, caso das ALNs. No caso das redes MLP, foram escolhidos para apresentação somente dois tamanhos diferentes: um com pouca oscilação do sinal de saída e outro com muita, ou seja, no primeiro caso o tamanho da rede é suficiente para resolver o problema, e no segundo, a rede foi superdimensionada.

Para este exemplo consideramos os seguintes parâmetros:
$\psi_o = 3e^2_{max}$,
$\psi_f = 0{,}01$,
Número de épocas = 5000,
Taxa de aprendizado = 0,001.

A tabela 4.15 mostra as medições estatísticas feitas sobre os resultados gerados pelas redes após o treinamento, obtidos com os dados de treino. Observamos que em relação ao sinal contaminado não há grandes diferenças entre os RMSEs e NRMSEs das redes com e sem Janela[4], mas quando comparamos os resultados em relação ao sinal limpo, percebemos que o método da Janela proporcionou às redes um ganho significativo, com NRMSEs próximos de zero. Agora, quando comparamos o método de regularização Bayesiana com a rede MLP - E2, podemos ver pelo NRMSE relativo ao sinal limpo, que o método Bayesiano melhorou o resultado, mas se a comparação for feita com a rede MLP - E2JG, observamos que a Janela aproxima melhor da equação desejada (y = 3x). Vemos também que, as médias e variâncias das redes com Janela ficaram próximas dos valores obtidos para o sinal limpo.

		Sinal Contaminado		Sinal Limpo			
Rede	Estrutura	RMSE	NRMSE	RMSE	NRMSE	Média	Variância
ALN	13 folhas	17,4399	0,9089	2,6371	0,3045	-1,6592	43,7468
ALN - JG	12 folhas	17,2517	0,8991	0,0262	**0,0030**	0,0205	75,3647
MLP - E1	1 - 5- 1	17,1079	0,8916	2,5144	0,2903	-0,0960	70,6838
MLP - E1JG	1 - 5 - 1	17,2602	0,8996	0,2534	**0,0293**	-0,0150	74,0034
MLP - E2	1 - 15- 1	16,7734	0,8742	3,9028	0,4507	0,2773	75,8197
MLP - E2JG	1 - 15 - 1	17,2325	0,8981	0,1624	**0,0188**	0,0186	74,7321
TRAINLM	1 - 15 - 1	16,5371	0,8619	4,9344	0,5698	0,1890	94,7657
TRAINBR	1 - 15 - 1	16,9602	0,8839	2,3376	0,2699	0.1709	75,8271

Tabela 4.15: Estatísticas obtidas com dados de treinamento do sistema y = 3x contaminado.

Mostramos na Figura 4.8 os resultados gráficos relativos ao treinamento comparando as redes com e sem a Janela. Podemos perceber que as saídas geradas pelas redes com Janela, lado direito da Figura, praticamente sobrepõe-se aos dados limpos. No caso específico das MLPs, percebe-se também que as oscilações apresentadas nos gráficos à esquerda, que são resultados das

[4]No caso das MLPs, existe até uma pequena piora nos resultados com Janela. No caso da ALN, devido a um critério de parada específico da implementação da rede, obtivemos um resultado final pior sem do que com Janela.

implementações tradicionais, desaparecem quando nelas é inserido o método da Janela.

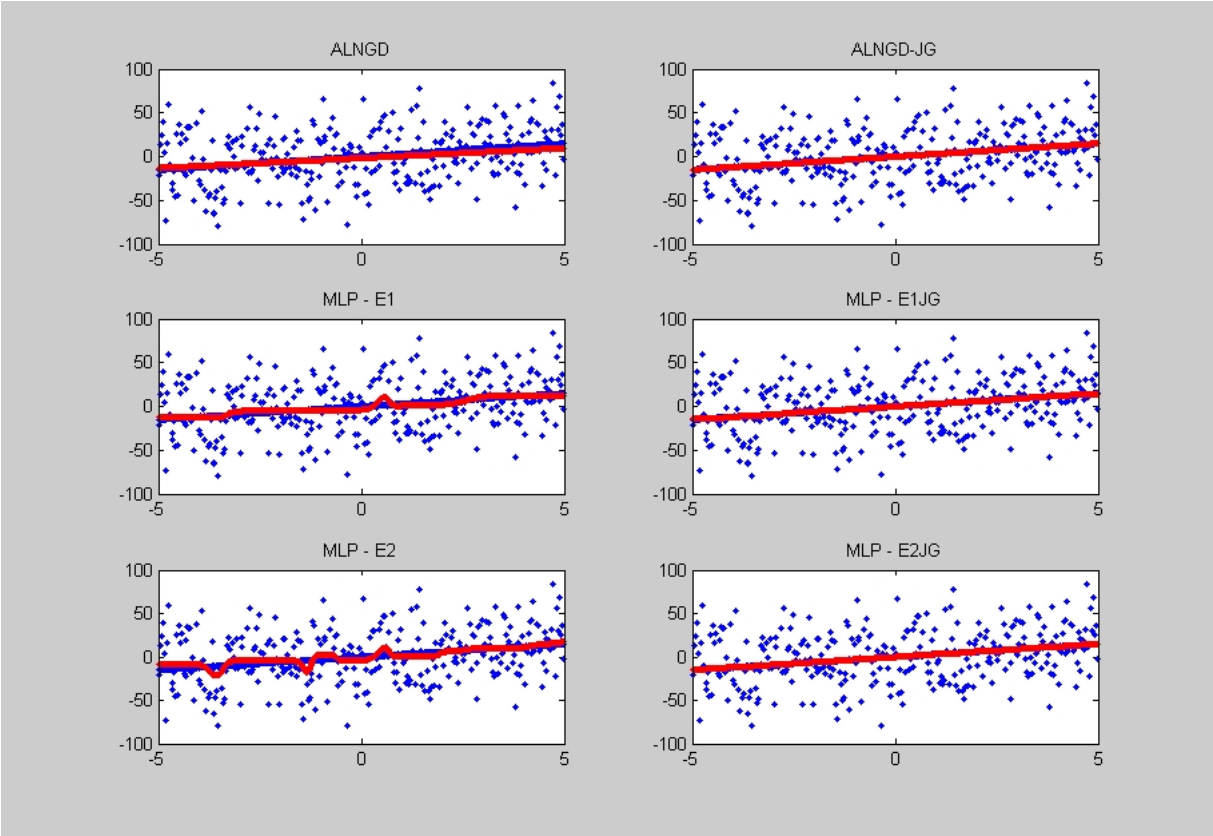

Figura 4.8: No eixo vertical é apresentada a amplitude dos dados e no eixo horizontal o número da amostra. As Figuras à esquerda contém as saídas das redes sem a implementação do método da Janela, enquanto que as Figuras à direita, apresentam os resultados das que utilizam o método.

Já a Figura 4.9 mostra os resultados de saída do TRAINLM e TRAINBR. Vemos que a rede regularizada por Bayes, apesar de melhorar o resultado da implementação Levenberg-Marquardt e deixá-la mais suave, ainda oscila em torno da equação desejada, enquanto que, comparativamente, a saída regularizada da rede com Janela é isenta dessas oscilações.

Analisando agora o comportamento das redes, sem e com Janela, frente aos dados de teste, podemos notar pelos NRMSEs na Tabela 4.16, que a Janela não altera significativamente os resultados das redes em relação ao sinal contaminado, o que significa dizer que frente a este conjunto de dados, as redes são equivalentes, com dispersão dos valores abaixo de 2%, mas quando

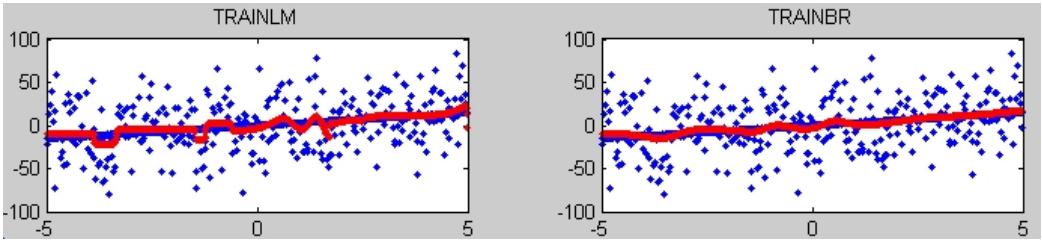

Figura 4.9: Comparação gráfica entre os resultados das redes neurais geradas no Matlab sem e com regularização, TRAINLM e TRAINBR respectivamente. O eixo vertical representa a amplitude e o eixo horizontal o número da amostra.

comparamos com o sinal limpo, vemos que os resultados são mais precisos, uma vez que os NRMSEs são de 10 a 100 vezes menores.

Graficamente, conforme mostrado pela Figura 4.10, podemos observar que a rede ALN tradicional sofre um desvio em relação aos pontos da reta desejada, o mesmo não ocorre para a ALN com Janela. As redes MLP tradicionais oscilam em amplitude, eixo vertical, ao longo do conjunto de dados, enquanto que, com Janela, os resultados são lineares, conforme o sinal desejado.

Por último, comparando o TRAINBR com a Janela, podemos notar que, o método da Janela aqui proposto, apresenta resultado bem superior a regularização Bayesiana, não só na etapa de treinamento, mas também na etapa de teste de generalização, fato que pode ser observado ao comparmos os resultados gráficos das Figuras 4.10 e 4.11.

		Sinal Contaminado		Sinal Limpo			
Rede	Estrutura	RMSE	NRMSE	RMSE	NRMSE	Média	Variância
ALN	13 folhas	16,9402	0,8904	2,6530	0,3033	-1,6592	45,0188
ALN - JG	12 folhas	16,6076	0,8730	0,0263	**0,0030**	0,0205	77,5560
MLP - E1	1 - 5- 1	16,3612	0,8600	2,5149	0,2875	-0,0939	72,1734
MLP - E1JG	1 - 5 - 1	16,5883	0,8719	0,2715	**0,0310**	-0,0159	75,8946
MLP - E2	1 - 15- 1	16,5588	0,8704	3,9170	0,4478	0,3196	77,6666
MLP - E2JG	1 - 15 - 1	16,5993	0,8720	0,1726	**0,0197**	0,0218	76,7673
TRAINLM	1 - 15 - 1	17,0676	0,8971	6,2469	0,7142	-0.1659	98.5086
TRAINBR	1 - 15 - 1	16,5007	0,8673	2,3703	0,2710	0,2091	77,5496

Tabela 4.16: Estatísticas de teste do sistema y = 3x contaminado.

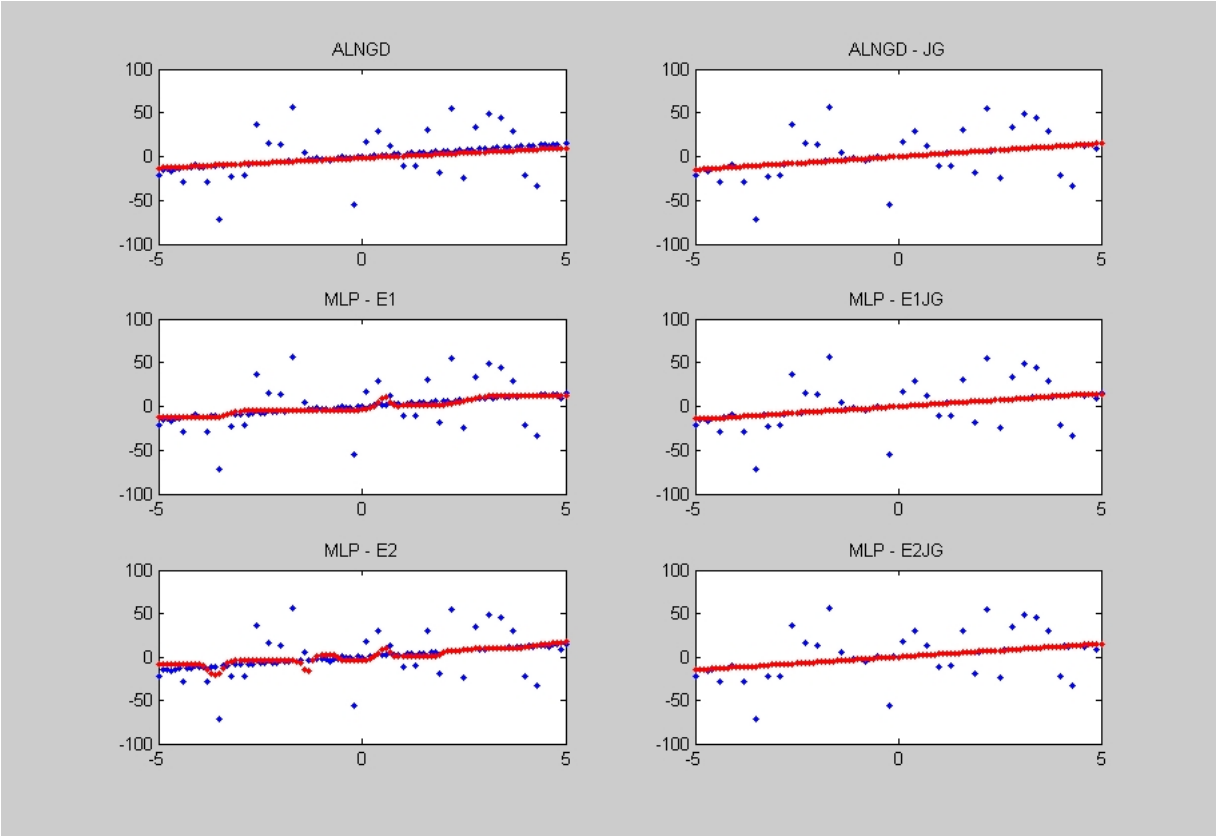

Figura 4.10: Resultados de teste de generalização. O eixo vertical representa a amplitude e o eixo horizontal a amostra.

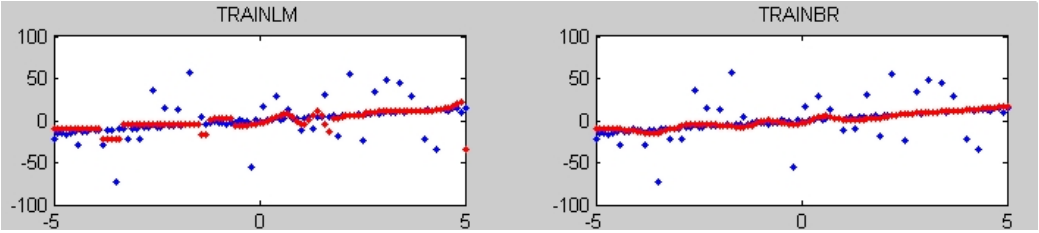

Figura 4.11: Comparação gráfica entre a regularização Bayesiana e o método da Janela frente aos dados de teste. O eixo vertical representa a amplitude e o eixo horizontal a amostra.

Com Ruído: *wnoise* Contaminado por Ruído Aditivo

wnoise é uma função do Matlab utilizada para gerar dados de teste com descontinuidades e com possibilidade de escolha de diferentes formatos para a forma de onda, como por exemplo, um formato que simula o efeito doppler, ou blocos de nível de diferentes tamanhos e amplitudes, ou uma função senoidal com pequenas transições abruptas. Para este teste utilizamos como sinal limpo a forma de onda do seno representada pela Figura 4.12, principalmente com o intuito de mostrar que a rede ALN, também é capaz de lidar com curvas e transições abruptas.

Para dificultar ainda mais o problema e demonstrar a capacidade do método da Janela de lidar com sinais que possuam nível em parte dos elementos, contaminamos o sinal limpo com um ruído gaussiano injetado em todo o conjunto de dados e em 25% dos dados foi adicionado um nível de sinal constante e positivo, resultando na forma de onda representada pela Figura 4.13. As medidas de média e variância dessa forma de onda podem ser vistas na na Tabela 4.17.

Para este exemplo, o sinal de entrada foi pré-processado e limitado ao intervalo [0, 1], para evitarmos que os valores grandes na entrada acabem por saturar as saídas das funções de ativação dos MLPs.

Tipo de sinal	Média - μ	Variância - σ^2
Sinal Limpo	-0,8359	8,8224
Sinal Contaminado	-0,1551	10,1600

Tabela 4.17: Estatísticas de média e variância associadas ao experimento *wnoise*.

As redes neurais foram treinadas com a seguinte configuração:
$\psi_o = 3e^2_{max}$, se estiver usando o método da Janela,
$\psi_f = 0{,}01$, se estiver usando o método da Janela,

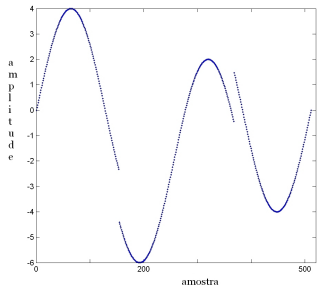

Figura 4.12: Gráfico do *wnoise* para função senoidal obtido no Matlab.

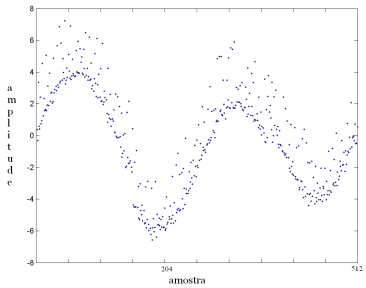

Figura 4.13: Gráfico do sinal *wnoise* senoidal após contaminação por ruído.

Número de épocas = 5000,
Taxa de aprendizado = 0,001.

A Tabela 4.18 mostra os resultados obtidos após o treinamento. Podemos notar que, no caso do sinal contaminado, as redes ALN-JG, MLP-E1JG e o MLP - E2JG aumentaram o NRMSE em relação às implementações que não utilizaram o método da Janela em respectivamente 13,9%, 18,8% e 17,9%, indicando uma menor aproximação dos dados contaminados. Agora, quando comparamos os mesmos resultados em relação ao sinal limpo, vemos que há uma redução mais acentuada do NRMSE, que em termos relativos nos dá 54,7%, 56,0% e 49,2%, respectivamente.

Ao observarmos o comportamento gráfico dos resultados, conforme a Figura 4.14, podemos notar que o efeito produzido pela Janela foi o de eliminar o *offset* contido em parte dos dados. Foi esse *offset* que forneceu a grande vantagem das redes com Janela quando medido em relação ao sinal limpo. Por último, podemos observar que o TRAINBR não apresentou vantagens em relação ao TRAINLM, e nenhuma das duas implementações, conseguiu eliminar o *offset* existente, conforme Figura 4.15. Porém, nesta Figura, podemos observar o efeito de suavização que a regularização Bayesiana produz no algoritmo Levenberg-Marquardt, ao olharmos para as regiões compreendidas entre os intervalos [0, 0,2] e [0,6, 0,8].

		Sinal Contaminado		Sinal Limpo			
Rede	Estrutura	RMSE	NRMSE	RMSE	NRMSE	Média	Variância
ALN	47 folhas	1,1566	0,3632	0,8437	0,2841	-0,0647	9,0659
ALN - JG	55 folhas	1,3172	0,4137	0,3826	**0,1288**	-0,7744	8,4844
MLP - E1	1 - 5- 1	1,1830	0,3716	0,9374	0,3157	0,0435	8,5132
MLP - E1JG	1 - 5 - 1	1,4055	0,4414	0,4128	**0,1390**	-0,0837	9,0012
MLP - E2	1 - 15- 1	1,1888	0,3734	0,9480	0,3192	0,0484	8,4926
MLP - E2JG	1 - 15 - 1	1,4019	0,4403	0,4819	**0,1623**	-0,8323	9,2628
TRAINLM	1 - 15 - 1	1,1149	0,3501	0,7281	0,2452	-0,1551	8,9144
TRAINBR	1 - 15 - 1	1,1393	0,3578	0,7282	0,2452	-0,1551	8,8353

Tabela 4.18: Estatísticas de treino do sistema *wnoise*.

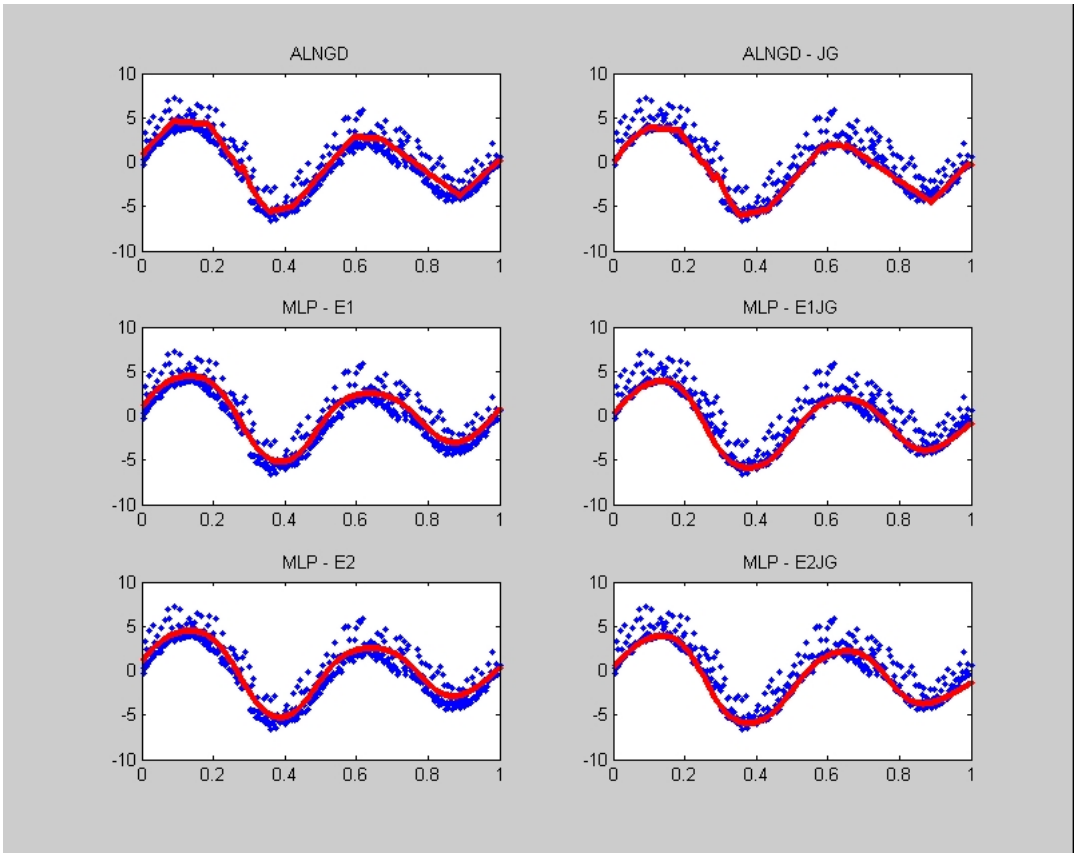

Figura 4.14: Resultado gráfico do treinamento para a função *wnoise*, comparando a saída das redes MLP e ALN com e sem o método da Janela. O eixo vertical representa a amplitude e o eixo horizontal normalizado, o número da amostra.

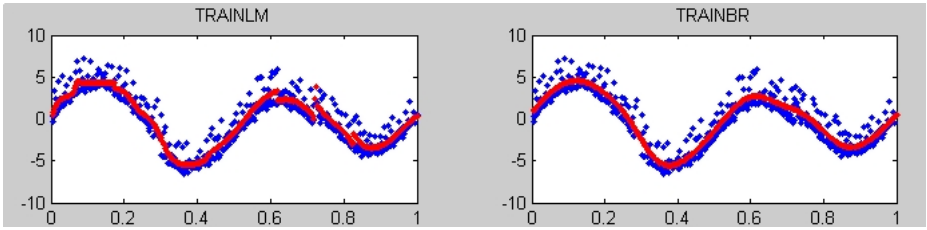

Figura 4.15: Resultado gráfico do *wnoise* para as MLPs treinadas com TRAINBR e TRAINLM. O eixo vertical representa a amplitude e o eixo horizontal normalizado, o número da amostra.

Com os resultados dos testes de generalização, conforme Tabela 4.19, Figura 4.16 e Figura 4.17, é possível fazer as mesmas conclusões anteriores, ou seja, a Janela produz os melhores resultados, pois é capaz de eliminar o *offset* de parte dos dados. No caso do TRAINBR o mesmo não ocorre e, se observarmos na Tabela 4.19 o NRMSE relativo ao sinal limpo , podemos ver que a regularização Bayesiana afasta-se mais desse sinal do que o seu parceiro TRAINLM.

Por último, podemos concluir que os algoritmos tradicionais são sensíveis a níveis DC existentes no conjunto de dados, mesmo que, se o *offset* contamina somente uma parte desse conjunto. A Janela, por outro lado, é capaz de identificar a tendência, mesmo no nível DC, da maioria do conjunto e com isso, o efeito do *offset* é retirado.

		Sinal Contaminado		Sinal Limpo			
Rede	Estrutura	RMSE	NRMSE	RMSE	NRMSE	Média	Variância
ALN	47 folhas	1,3958	0,4337	0,8137	0,2763	-0,0659	9,0544
ALN - JG	55 folhas	1,6546	0,5141	0,3736	**0,1268**	-0,7687	8,4850
MLP - E1	1 - 5- 1	1,3518	0,4200	0,9174	0,3115	0,0570	8,5232
MLP - E1JG	1 - 5 - 1	1,7233	0,5354	0,4179	**0,1419**	-0,8734	9,0107
MLP - E2	1 - 15- 1	1,3605	0,4227	0,9276	0,3149	0,0602	8,5017
MLP - E2JG	1 - 15 - 1	1,7258	0,5362	0,4888	**0,1660**	-0,8249	9,2767
TRAINLM	1 - 15 - 1	1,4612	0,4540	0,7015	0,2382	-0,1636	8,8273
TRAINBR	1 -15 - 1	1,3840	0,4300	0,7102	0,2411	-0,1414	8,8458

Tabela 4.19: Estatísticas de teste para os dados do *wnoise*.

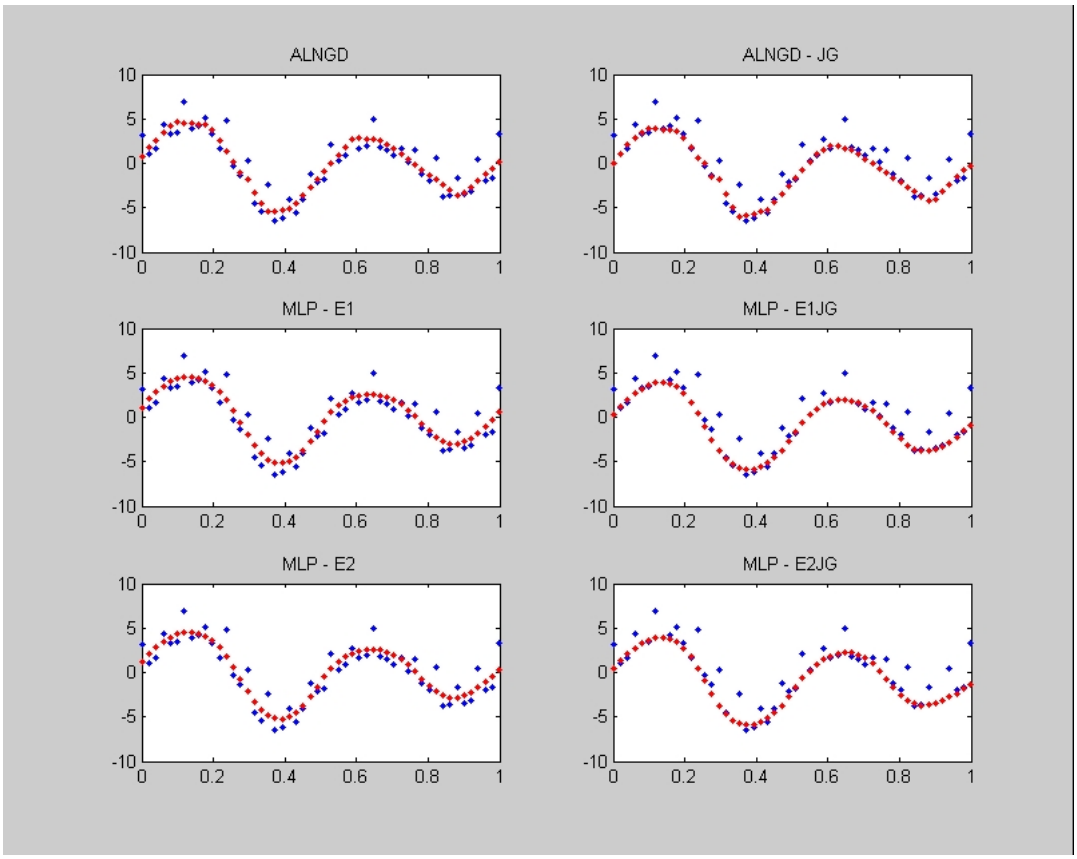

Figura 4.16: Resultado gráfico do teste de generalização para a função *wnoise*, comparando as saídas das redes MLP e ALN com e sem o método da Janela. O eixo vertical representa a amplitude e o eixo horizontal normalizado, o número da amostra.

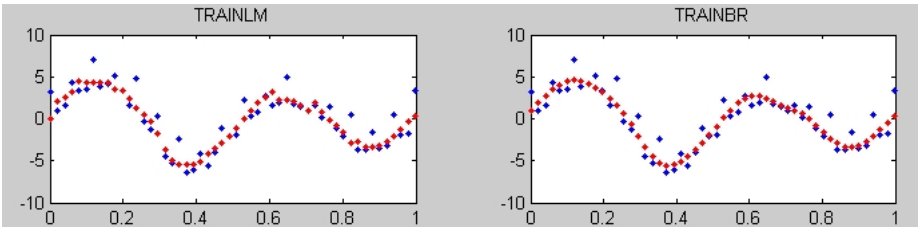

Figura 4.17: Resultado gráfico do teste de generalização para a função *wnoise*, comparando as saídas das rede MLP treinadas com os algoritmos TRAINLM e TRAINBR. O eixo vertical representa a amplitude e o eixo horizontal normalizado, o número da amostra.

4.3.2 Previsão de séries temporais: Ozônio

Para análise da influência do método da Janela em um sistema previsor, foram utilizadas bases de dados de séries históricas, apresentadas em [24], como por exemplo:

- Medições de temperatura médias mensais, em graus centígrados, de janeiro de 1976 a dezembro de 1985, na cidade de Ubatuba.
- Número de manchas solares de Wölfer; observações anuais de 1749 a 1924.
- Precipitação atmosférica em Lavras, MG; observações mensais de janeiro de 1966 a dezembro de 1997.
- Valores mensais de concentração de ozônio em Azuza, Califórnia, EUA, de janeiro de 1956 a dezembro de 1970.
- Valores mensais do consumo de energia elétrica no estado do Espírito Santo, de janeiro de 1968 a setembro de 1979.

Apesar dos inúmeros testes, optou-se aqui por mostrar somente os resultados de um deles, mais precisamente o das observações das concentrações de ozônio em Azuza, Figura 4.18, já que não houve grandes diferenças nas características dos resultados deste conjunto com relação aos demais.

Os parâmetros utilizados no treinamento e teste foram:

- tamanho total dos dados: 180 amostras,
- quantidade dos dados utilizados para treinamento: 90% das amostras,
- quantidade dos dados utilizados para teste: 10% das amostras,
- número de unidades de atraso: 5,
- número de passos à frente: 1,
- $\psi_o = 3e^2_{max}$,
- $\psi_f = 0{,}01$,
- número de épocas = 10000,
- taxa de aprendizado inicial = 0,0001.

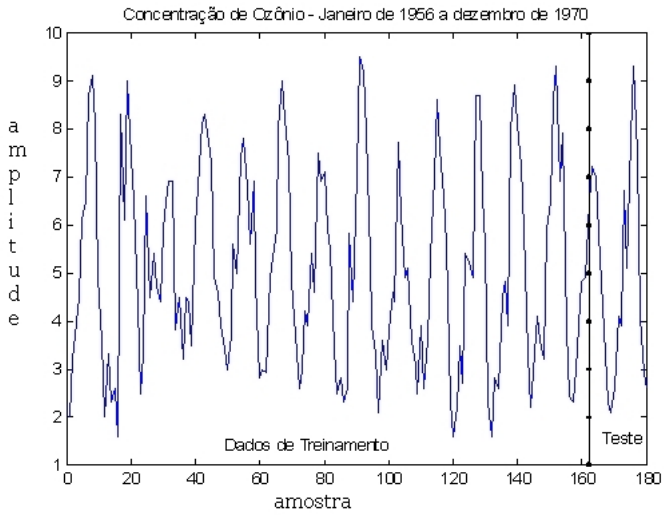

Figura 4.18: Série histórica da concentração de ozônio em Azuza, de 1956 a 1970.

Dos resultados apresentados na Tabela 4.20, podemos verificar que todas as redes, exceto a MLP com TRAINLM, possuem o indice U de Theil abaixo de 1, o que indica uma previsão melhor do que a previsão trivial. Apesar disso, ao compararmos as redes entre si, vemos que o método da Janela aplicado às redes ALN, MLP-E1 e MLP-E2 proporcionou um ganho de 3,1%, 14,8% e 2,7% na previsão e na precisão (NRMSE), enquanto que, o TRAINBR produziu resultados aproximadamente 52% melhores do que o TRAINLM.

Comparando os resultados do TRAINBR com o MLP - E2JG, vemos que o primeiro apresenta um ganho de 10,3%, mas ainda assim é pior do que os valores gerados pela rede MLP - E1JG, o que nos leva a questionar se a previsão do TRAINBR com uma estrutura menor, com apenas 5 neurônios ocultos, produzirá previsões melhores. Podemos ver na Tabela 4.21 que não é isso que ocorre. O TRAINBR, apesar de continuar melhorando o resultado do TRAINLM, se manteve estável em relação a rede com 15 neurônios ocultos, e com isso, a rede MLP - E1JG apresentou um resultado favorável de 1,5% em relação a rede de mesmo tamanho e com regularização Bayesiana.

Nas Tabelas 4.20 e 4.21 podemos ver o efeito que uma quantidade excessiva de neurônios produz nos resultados da rede[5], ao compararmos os resultados das estruturas com 5 e 15 neurônios das redes TRAINLM. Vemos que a estrutura menor consegue melhorar, e muito, o resultado da previsão.

[5]Dilema *bias*-variância

Investigando esse fato, percebemos que, após o processo de treinamento, a estrutura com mais neurônios produz um menor valor de NRMSE e de U de Theil em relação ao conjunto de treinamento, o que, com a ajuda dos resultados de teste, acaba por mostrar que o alto grau de liberdade dessa estrutura, produz uma memorização do conjunto de treinamento e, consequentemente, a perda da capacidade de generalização.

Rede	Tamanho da Estrutura	RMSE	NRMSE	U de Theil
ALNGD	40 folhas	0,1204	0,5420	0,7061
ALNGD - JG	40 folhas	0,1167	0,5254	**0,6845**
MLP - E1	1 - 5 - 1	0,1281	0,5768	0,7514
MLP - E1JG	1 - 5 - 1	0,1091	0,4913	**0,6401**
MLP - E2	1 - 15 - 1	0,1269	0,5716	0,7446
MLP - E2JG	1 - 15 - 1	0,1235	0,5563	0,7247
TRAINLM	1 - 15 - 1	0,2347	1,0571	1,3772
TRAINBR	1 - 15 - 1	0,1108	0,4988	**0,6498**

Tabela 4.20: Resultados da previsão dos níveis de concentração da camada de ozônio sobre os dados de teste.

Rede	Tamanho da Estrutura	RMSE	NRMSE	U de Theil
TRAINLM	1 - 5 - 1	0,1322	0,5951	0,7753
TRAINBR	1 - 5 - 1	0,1108	0,4988	0,6498

Tabela 4.21: Resultados da previsão dos níveis de concentração da camada de ozônio sobre os dados de teste.

Graficamente podemos ver pela Figura 4.19 que no intervalo [0, 5] a aplicação da Janela às redes permite uma maior aproximação dos resultados de saída em relação aos dados desejados. Além disso, o pico no intervalo [10, 15] está mais bem definido nessas redes. No caso do TRAINBR, percebemos que o resultado de saída da rede é muito mais suave do que o fornecido pelo TRAINLM, conforme Figura 4.20, e comparando-o com o MLP - E2JG, no intervalo [0, 5] vemos que há uma menor aproximação da curva desejada, mas nos intervalos [10, 12[e [15, 20] ele produz os melhores resultados.

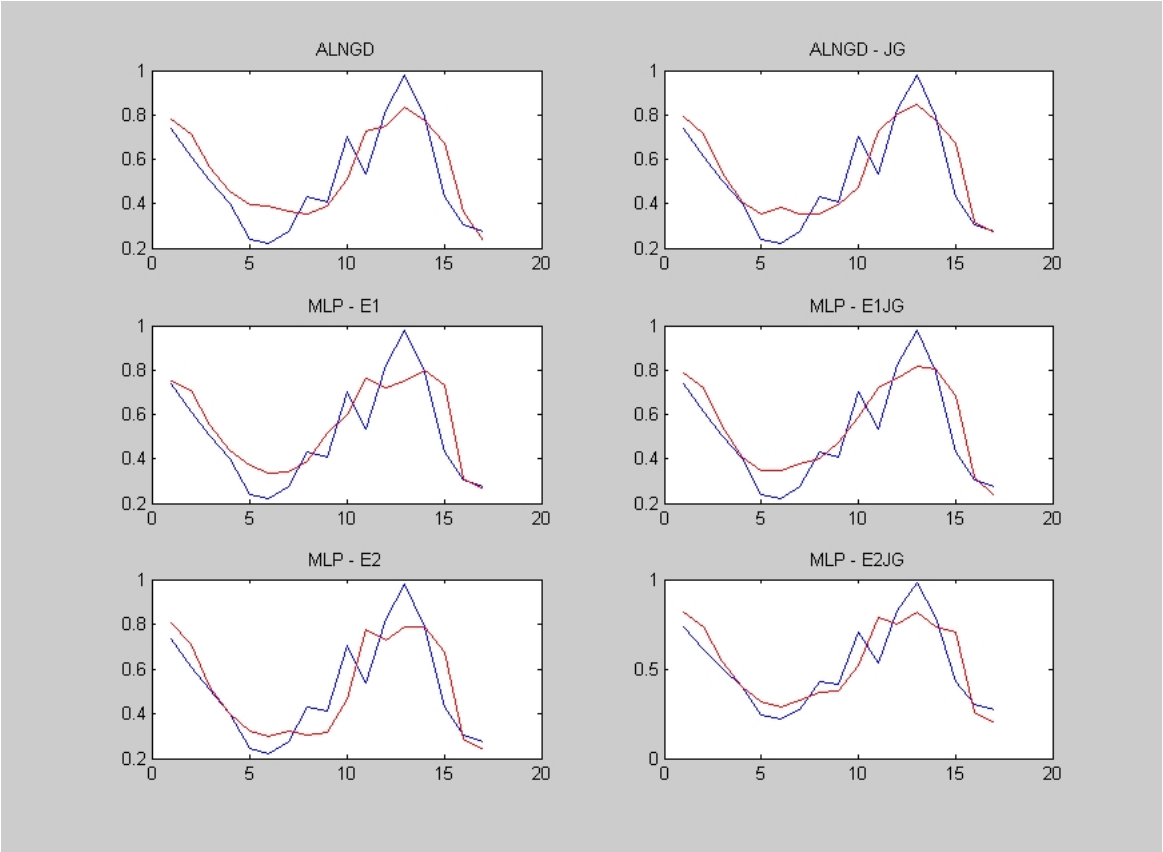

Figura 4.19: Resultado gráfico da previsão dos níveis de concentração da camada de ozônio. O eixo vertical representa a amplitude normalizada e o eixo horizontal o número da amostra de teste.

4.3.3 Filtragem neural: PIR

Como aplicação prática de um filtro neural, utilizamos os sinais provenientes de um sensor de infravermelho passivo(PIR). Infelizmente, não foi possível simular este problema com o Flood. Devido a quantidade de dados, mais de 80.000 vetores, o Flood apresentou erro de aproximação numérica[6].

O PIR

O sensor PIR é capaz de detectar mudanças no padrão de luz infravermelha emitida pelos objetos que estão na sua vizinhaça e, dependendo de como são

[6]Após contato com o autor da biblioteca, descobrimos que a parte que usamos (classe SumSquaredError.cpp) está sujeita a este tipo de problema. Ele recomenda usar a classe NormalizedSquaredError.cpp.

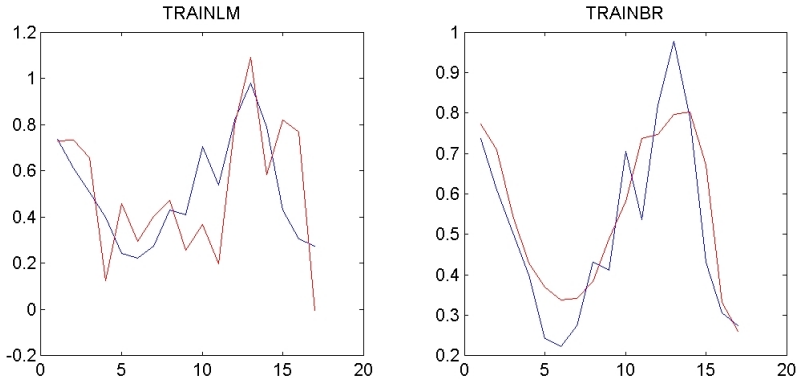

Figura 4.20: Resultado gráfico da previsão dos níveis de concentração da camada de ozônio com TRAINLM e TRAINBR. O eixo vertical representa a amplitude normalizada e o eixo horizontal o número da amostra de teste.

construídos, é possível detectar o movimento de um objeto no ambiente e a direção para onde ele se desloca. Para isso, o sensor deve ser construído com dois elementos PIR dispostos lado a lado, conforme a Figura 4.21. A saída resultante é uma forma de onda que depende do sentido no qual o objeto passa. Podemos ver na Figura 4.22 como esse sinal é gerado e sua forma característica.

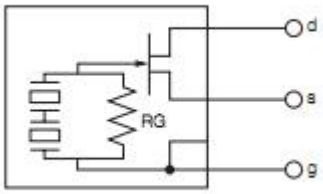

Figura 4.21: Sensor PIR construído para detecção de direção de movimento.

Estes sensores são muito utilizados em aplicações como sistemas de alarme, chaveamento de luz ativada por movimento e em robótica. Porém, recentemente têm surgido a necessidade de automatizar os sistemas que informam a lotação de passageiros dos veículos de transporte coletivo [25]. Várias alternativas existem para tentar resolver este problema como: barreiras de sensores infravermelhos ativos, câmeras com processamento digital de imagens, sensores capacitivos e os sensores infravermelhos passivos. Alguns pontos

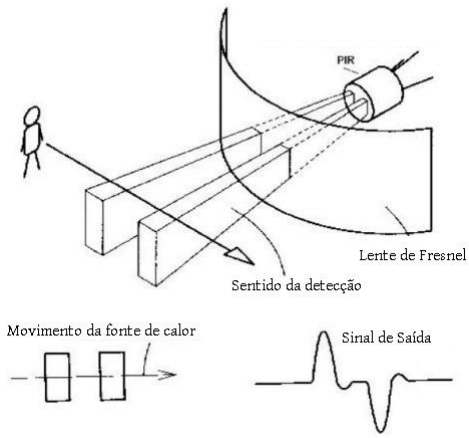

Figura 4.22: Utilização do sensor PIR para detecção de direção do movimento da fonte de calor e forma de onda típica.

importantes como o custo para implementação em toda a frota existente, a eficiência de cada uma e o seu grau de invisibilidade[7] devem ser usados na hora de se escolher qual método usar.

Neste contexto, o PIR é um sensor que atende muito bem a todos estes requisitos, pois ele é de construção simples e barata, pode ser instalado no teto acima de onde as pessoas passam e é bem eficiente. De forma que, para contar o número de pessoas que entram ou saem de um recinto com um PIR, basta olhar a variação do sinal da sequência de derivadas da resposta do sensor. Entretanto, conforme mostrado na Figura 4.23, os sinais provenientes destes sensores não está isento de ruídos provocados por variações térmicas no ambiente, nos componentes do sistema sensor e interferências de rádio frequência (RF) e essa contaminação produz diversas sequências de variação do sinal da derivada, o que dificulta a análise dos resultados.

O tratamento dos sinais ruídosos é feito através do uso de filtros, cujo objetivo é reduzir o efeito das impurezas contida nos dados. No caso do sensoriamento com o PIR, o uso de um filtro adaptativo é de extremo interesse já que a saída deste sensor é sensível a variações estatísticas do ambiente e a sua resposta normalmente está contaminada com ruído e com um nível de sinal que depende, entre outras coisas, da claridade do ambiente, da temperatura e da quantidade dos objetos em redor.

Como não dispunhamos de um sinal de referência para a implementação do filtro neural, foi necessário processar os dados em duas etapas, conforme

[7] A invisibilidade refere-se a capacidade do sistema integrar-se ao ambiente alterando-o o menos possível.

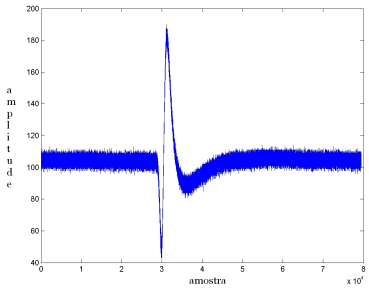

Figura 4.23: Sinal capturado correspondente à passagem de uma pessoa da esquerda para a direita.

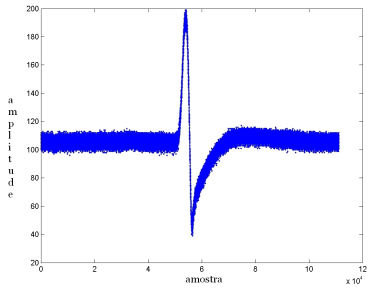

Figura 4.24: Sinal capturado correspondente à passagem de uma pessoa da direita para a esquerda.

Figura 4.25. Na primeira contruímos um aproximador de função para gerar o sinal desejado (sinal limpo) e na segunda implementamos o filtro utilizando o sinal de saída do aproximador como o sinal de referência.

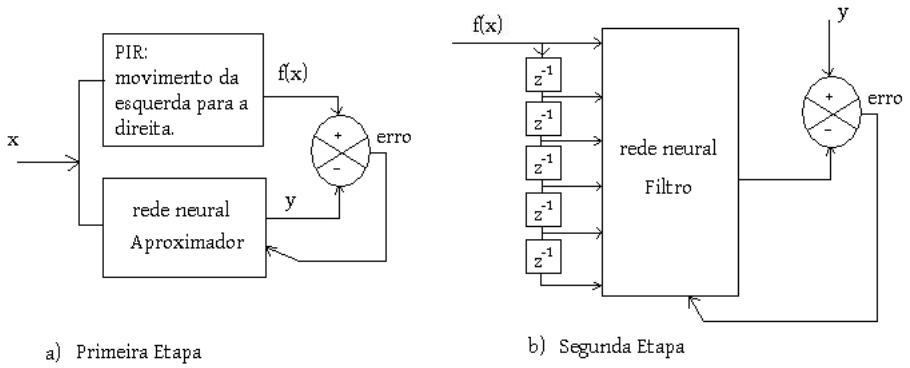

Figura 4.25: Metodologia utilizada para implementação do filtro neural para o sensor PIR.

Primeira Etapa: Aproximador de Função

Nesta etapa utilizamos o sinal obtido após o deslocamento de uma pessoa na região de alcance do sensor, com sentido do movimento ocorrendo da esquerda para a direita, Figura 4.23. Esse conjunto de dados foi utilizado para treinar a rede neural esquematizada como aproximador de função.

As Figuras 4.26e 4.27 mostram os gráficos resultantes após o treinamento das redes neurais como aproximador de função. Podemos notar que elas são

capazes de gerar uma boa estimativa para função alvo, mas, no caso da ALN, devido à característica linear e a critérios de parada específicos, vemos que na região arredondada houve uma linearização do conjunto de pontos. No caso do TRAINBR, percebemos que há uma certa instabilidade no intervalo [2x10^4, 3x10^4]. Apesar disso, o resultado final de ambas as redes nos fornece os valores alvo adequados o suficiente, para serem utilizados como referência no processo de treinamento do filtro. Porém, optamos por usar somente o resultado da ALN - JG.

Devemos destacar que apesar desse resultado ter a aparência de um filtro, ele executa somente um mapeamento entrada-saída e é adequado apenas para dados contidos na região de treinamento e com o formato dos dados para o qual foi realizado o ajuste. Se tentarmos extrapolar esse conjunto de dados obteremos respostas inadequadas. Além disso, para gerar estes dados, precisamos de um treinamento recursivo (aprendizado), que gasta um bom tempo para processar os dados, o que dificulta e provavelmente impede o seu uso em sistemas com restrição de tempo.

A configuração de treinamento da ALN é:

- taxa de aprendizado = 0,02 ,
- número de épocas = 150,
- $\psi_f = 300$,
- relação a/b = 3/4.

A Tabela 4.22 apresenta os resultados do RMSE e NRMSE obtidas após o treinamento. Vemos que o TRAINBR aproxima mais do conjunto de dados de treinamento do que a ALN.

Rede	Tamanho da Estrutura	RMSE	NRMSE
ALNGD - JG	28 folhas	3,7468	0,3256
TRAINBR	1 - 15 - 1	2,9516	0,2565

Tabela 4.22: Tabela de resultados do Aproximador de Função para o PIR.

Segunda Etapa: Construção do Filtro

Agora estamos interessados em incorporar informações temporais na rede neural. Para isso, utilizaremos como sinais de entrada o sinal atual e um conjunto de sinais atrasados no tempo. Mostraremos os resultados usando-se um vetor de entrada com 5 elementos atrasadores. Consideraremos como

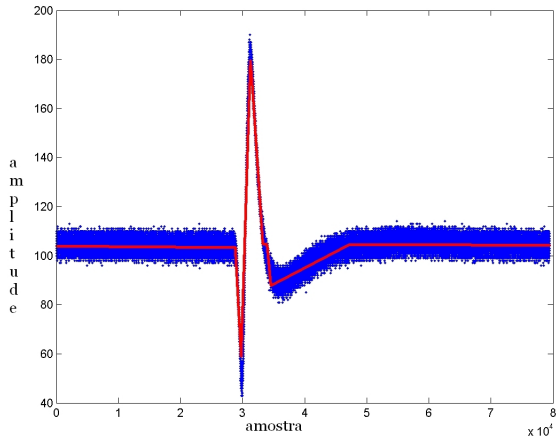

Figura 4.26: Em vermelho temos o resultado de saída da ALN - JG, treinada como um aproximador de função para o problema do PIR.

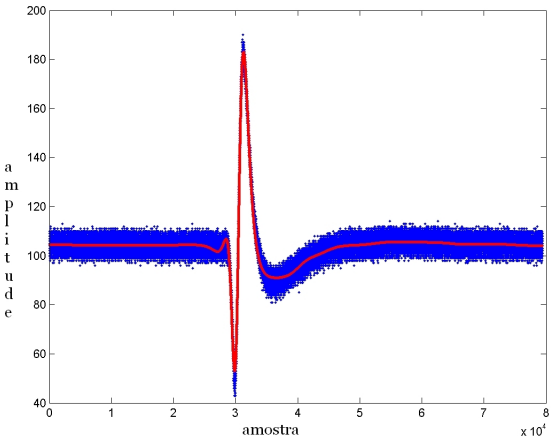

Figura 4.27: Em vermelho temos o resultado de saída gerado pelo TRAINBR, treinado como um aproximador de função para o problema do PIR.

sinal desejado o conjunto de dados gerado pela rede ALN da etapa anterior (fase de aproximação). As configurações utilizadas durante o treino são:

- taxa de aprendizado = 0,00001,
- número de épocas = 150,
- $\psi_f = 300$,
- relação a/b = 3/4,
- número de entradas = 6.

Podemos notar pelas Figuras 4.28 e 4.29 que o nível de ruído, durante os momentos sem o sinal de pessoa passando, foi reduzido, mas não completamente eliminado. A Tabela 4.23 mostra pelo NRMSE, medido em relação ao sinal desejado (limpo), que a rede com TRAINBR consegue, no geral, ser melhor do que a ALNGD - JG. Porém, devemos considerar que, para treinar a MLP foi necessário pré-processar os dados de entrada e pós-processar os dados de saída, procedimento que não foi utilizado na ALN. Pelas Figuras podemos observar também que, o resultado do sinal para baixo na ALN - JG, compreendido no intervalo [2x10^4, 3x10^4], é mais forte do que no TRAINBR,

e no sinal para cima, contido no intervalo [3x10^4, 3,5x10^4] eles são equivalentes. Outro ponto a destacar é que a curva do sinal não foi linearizada pelos filtros, como o que ocorreu com o aproximador ALN - JG, e manteve sua forma arredondada. Isso ocorreu devido às informações provenientes dos dados passados (z^{-1}), contidas no sinal de entrada na forma de memória.

Rede	Tamanho da Estrutura	RMSE	NRMSE
ALNGD-JG	40 folhas	2,3948	0,2395
TRAINBR	1 - 15 - 1	1,8638	0,1864

Tabela 4.23: Resultados do filtro neural para o sinal do PIR com deslocamento no sentido da esquerda para a direita, medido em relação ao sinal desejado.

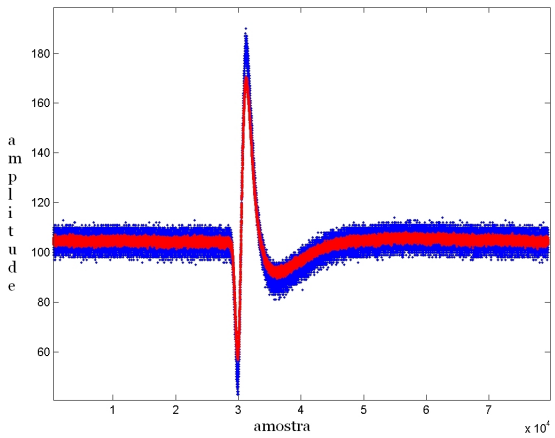

Figura 4.28: Resultado gráfico do filtro neural ALNGD - JG para o PIR. Deslocamento ocorrendo no sentido da esquerda para direita.

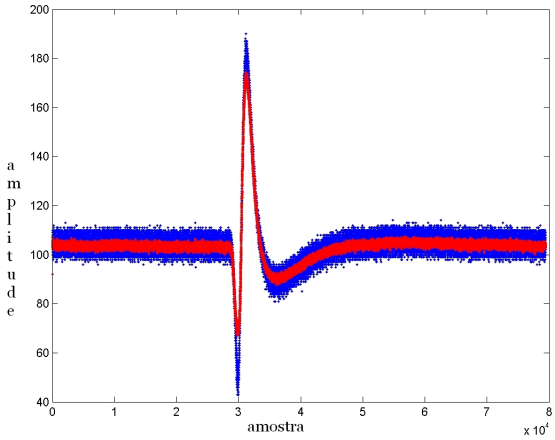

Figura 4.29: Resultado gráfico do filtro neural TRAINBR para o PIR. Deslocamento ocorrendo no sentido da esquerda para direita.

Teste Final

Neste teste final, usaremos somente o filtro gerado pela rede ALNGD - JG, uma vez que não precisaremos pré-processar os dados de entrada e nem pós-processar os dados de saída. Além disso, estamos interessados em demonstrar apenas a capacidade de generalização do filtro. Assim sendo, vamos passar

pelo mesmo, em uma fase somente de teste, outro sinal proveniente do PIR, mas capturado sob condições diferentes e com outras características.

O sinal que utilizaremos contém a informação do deslocamento de duas pessoas que, inicialmente passam pelo sensor da esquerda para à direta, uma após a outra, e logo em seguida retornam, ou seja, o segundo movimento ocorre da direita para à esquerda, conforme a Figura 4.30.

O resultado desse conjunto de dados após a passagem pelo filtro pode ser visto na Figura 4.31. Podemos notar que as principais características do sinal, que são os picos e as transições, se mantiveram e houve, conforme ocorreu durante o treinamento, uma redução no nível do ruído durante os momentos sem a informação de pessoas passando.

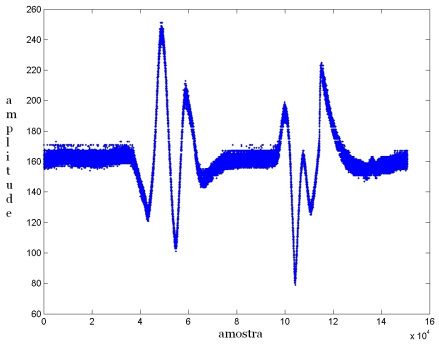

Figura 4.30: Resultado de saída do PIR após a passagem de duas pessoas, ida e volta .

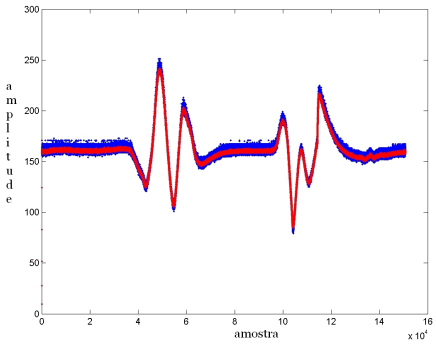

Figura 4.31: Resultado do filtro, em vermelho, sobreposto aos dados originais.

Considerações Finais

Apesar do uso do filtro ter reduzido o nível de ruído do sinal que chega do sensor, o sinal de saída da rede ainda é inadequado para ser usado diretamente no processo de contar pessoas que passam, caso deseje-se usar a análise da variação do sinal da derivada. Neste caso, será necessário o uso de algum outro filtro, por exemplo um filtro de média, para linearizar mais o sinal.

4.4 Resumo

Neste capítulo apresentamos alguns dos casos de teste utilizados para validar o método da Janela. Inicialmente, procuramos avaliar as variações nos resultados da rede frente à mudança da abertura inicial da Janela ψ_o. Depois,

comparamos os resultados entre redes similares com e sem a Janela para os casos de: aproximação de funções, predição e filtragem.

Capítulo 5

Conclusões

Ao empregar uma rede neural artificial como ferramenta para a resolução de um determinado problema, estamos interessados em construir um modelo não-linear do fenômeno físico responsável pela geração dos exemplos de entrada - saída. Porém, como os dados são representações limitadas da superfície que se deseja modelar e normalmente estão contaminados com ruídos, precisamos assumir um compromisso adequado entre a qualidade/quantidade de ajuste dos dados usados no treinamento e uma boa estimação daqueles exemplos que não estejam presentes no processo de aprendizado(i.e., capacidade de generalização dos dados desconhecidos). Esse é um problema conhecido como dilema *bias* - variância e sua solução requer o uso de algoritmos capazes de evitar que os dados de treinamento sejam decorados, em um processo conhecido como regularização.

Neste trabalho propusemos um novo método, chamado de método da Janela, para regularizar as redes neurais supervisionadas, treinadas a partir de dados ruidosos. Esse método não assume que o ruído contido nos dados é puramente Guassiano, na realidade opera melhor quando não é. Para usá-lo não é necessário reduzir o conjunto de dados de treinamento ou interromper o processo de ajuste para avaliar o comportamento da rede. Além disso, ele possui baixo custo computacional e é adequado para o uso em algoritmos que ajustam os pesos a cada amostra apresentada.

O método da Janela baseia-se na medição do erro quadrático gerado por cada vetor de entrada, cujo valor não pode ultrapassar um limite máximo estabelecido. Assim sendo, o erro deverá estar contido dentro de uma região permitida, caso contrário, a amostra não é utilizada para o ajuste dos pesos. Para um único processo de treinamento, o valor inicial da Janela deve ser grande o suficiente para permitir o uso integral de todas as amostras e depois, à medida em que a rede for incorporando as características da função objetivo, a abertura da Janela irá sendo reduzida até chegar ao valor final

desejado.

Para o processo de fechamento da Janela adaptamos uma equação de decaimento de pesos tradicional, que garante um decaimento monotônico, não linear e limitado a um valor final pré-estabelecido. Para o valor inicial de abertura da Janela, ψ_o, utilizamos uma passagem do conjunto de dados pela rede não-treinada, para encontrar o maior erro quadrático gerado e a partir daí, estabelecer um valor de abertura maior ou igual a esse. Numa atitude mais conservadora, recomendamos que a escolha do ψ_o seja tal que, o ψ iguale-se ao e^2_{max} inicial, faltando 5% ou menos do tempo de treinamento.

Para adequar um algoritmo de treinamento à Janela, basta efetuar pequenas modificações em software como a inserção de: algumas variáveis, da equação do decaimento e de um teste após a passagem da amostra pela rede. Particularmente, implementamos o método nas redes ALN e MLP, ambos baseados no algoritmo do gradiente descendente.

Para validação do método, analisamos a influência da Janela em vários casos de teste, separados de acordo com a sua aplicação, mais especificamente, em aproximação de funções, previsão de série temporais e filtragem de ruídos. Dos casos de teste, uma parte dos dados utilizados são provenientes de conjuntos teóricos e outra parte, são amostras de medições de sensoriamento em aplicações práticas.

Para comparação dos resultados foram utilizadas as métricas da raiz quadrada do erro quadrático médio, RMSE, e de sua variante normalizada, o NRMSE, medidos sobre o conjunto de dados de treinamento, dados ruidosos, e nos casos teóricos, extendemos estas medições sobre o conjunto de dados gerados pelas funções primárias, dados limpos. No caso dos sistemas previsores, foi utilizado também o medidor U de Theil, que gera resultados comparativos entre a previsão efetuada e a previsão trivial.

Os resultados obtidos, mostraram que o método da Janela funciona como elemento regularizador, sendo capaz de garantir um bom compromisso entre o ajuste dos dados e a capacidade de generalização da rede. Além disso, quando esses mesmos resultados são comparados com processos tradicionais de regularização, como a regularização bayesiana e o crescimento de rede, podemos ver que eles são equivalentes ou superiores, principalmente se o ruído não é gaussiano.

Concluímos que, o uso de elementos regularizadores procura garantir uma boa relação entre o ajuste e a capacidade de generalização de uma rede neural para um problema em particular, através de um ajuste dos parâmetros livres à tendência dos dados ao invés de aos dados. Porém, isso não miniminiza a importância de um bom conjunto de exemplos de entrada - saída da função desejada e nem elimina por completo a influência gerada pelo projetista na escolha dos parâmetros de treinamento da rede.

Para continuação deste trabalho sugerimos as seguintes linhas de pesquisa:

- Testar o método da Janela em redes com múltiplas saídas.
- Estabelecer um critério para escolha do valor final de abertura da Janela, ψ_f.
- Testar o método em um processo de treinamento por lote.
- Estabeler de critérios para interromper o fechamento da Janela, quando for detectado que há um número excessivo de amostras sendo descartadas.
- Adaptar o método para redes com treinamento baseados nos métodos de Newton.

Referências Bibliográficas

[1] Abelém, A. J. G.. **Redes Neurais Artificiais na Previsão de Séries Temporais**, *Dissertação de Mestrado*, Pontifícia Universidade Católica do Rio de Janeiro, 1994.

[2] Armstrong, W.. **Hardware requirements for fast evaluation of functions learned by Adaptive Logic Networks**, *Evolvable Systems: From Biology to Hardware*, vol. 1259, 1997.

[3] Armstrong, W.; Gorodnich, D.. **Breaking Hyperplanes to fit Data with Applications to 3D World Modeling and Oil Sand Data Analysis**, *Proc. ICSC Symposium on Neural Computation*, CD-ROM publ. by ICSC Academic Press, May 23 - 26, 2000.

[4] Armstrong, W.. **Piecewise linear data fitting with ALNfit Pro**, Disponível em: *<http://www.dendronic.com/tutoriais/Tutorial on ALNfit Pro.zip>*. Acesso em: 06 de Janeiro de 2008.

[5] Bakucz, P; Yacoot, A.; Cziomba, T.; Koenders, L.; Kruger-Shem, R.. **Neural network approximation of tip-abrasion effects in AFM imaging**, *Measurement Science and Technology*, 6, 2008

[6] Bishop, C.. **Pattern Recognition and Machine Learning**, *Springer Science+Business Media, LLC*, 2006.

[7] Bochman, G. V.; Armstrong, W.. **Properties of Boolean Functions with a Tree Decomposition***BIT, Numerical Mathematics*, vol. 14, p. 1-13, 1974.

[8] Castro, R.; Moreira, R.; Eposito, E.; Lucca, E.. **Avaliação do ruído em sensores eletroópticos: abordagem da imagem**

escura no HSS, *Anais XII Simpósio Brasileiro de Sensoriamento Remoto*, INPE, p. 355-362, 2005.

[9] Dias, F.; Antunes, A.; Mota, A.. **Regularization Versus Early Stopping: A case Study With a Real System**, *2nd IFAC Conference Control Systems Design*, Bratislava, 2003.

[10] Esp, D.. **Adaptive Logic Networks in Bank Customer Modeling**, , 2001.

[11] Foresee, F. D.; Hagan, M. T.. **Gauss-Newton approximation to Bayesian regularization**, *Proc. IJCNN 1997*, p. 1930-1935, 1997

[12] Fritzke, B.. **Some Competitive Learning Methods**, tech. rep. *Institute For Neural Computation Ruhr-Universitat Bochum*, 1997.

[13] Ganguli, R.. **Noise and Outlier Removal From Jet Engine Health Signals Using Weithed FIR Median Hybrid Filters**, *Mechanical Systems and Signal Processing*, p. 967-978, 2002.

[14] Haykin, S.. **Redes Neurais Princípios e Prática**, *Bookman*, 2ª edição, 2004.

[15] Ho, K.; Leung, C.; Sum, J.. **On Weight-Noise-Injection Training**, *Advances in Neuro-Information Processing*, p. 324-331, Springer Berlin / Heidelberg, 2009.

[16] Krogh, A.; Hertz, J.. **A Simple Weight Decay Can Improve Generalization**, *In Advances in Neural Information In Processing Systems*, vol. 4, p. 950-957, 1992.

[17] Kwok, T. ; Yeung, D.. **Constructive Algorithms For Structure Learning In Feedforward Neural Networks For Regression Problems**, *IEEE Transactions on Neural Networks*, p. 630-645, 1997.

[18] Li, S.; Chen, S.. **Function Approximation Using Robust Wavelet Neural Networks**, *14th IEEE International Conference on Tools with Artificial Intelligence*, ICTAI, p. 483, 2002.

[19] Li, S.; Leiss, E.. **On Noise-immune RBF Networks**, *Radial basis function networks 1: recent developments in theory and applications*, 2001.

[20] Lopez, R.. **Flood: An Open Source Neural Network C++ Library**. Disponível em: *<http://www.cimne.com/flood/download.asp>*. Acesso em: 3 de Julho de 2009.

[21] Lourakis, M. I. A.. **A Brief Description of the Levenberg-Marquardt Algorithm Implemented by levmar**. *Institute of Computer Science Foundation for Research and Technology - Hellas (FORTH)*, 2005

[22] Mackay, D.J.C. **Bayesian interpolation**. *Neural Computation*, v. 4, n. 3, p. 415-47, May 1992.

[23] McCulloch, W.; Pitts, W.. **A logical calculus of ideas imminent in nervous activity**, *Bulletin of Mathematical Biophysics*, vol. 5, p. 115-133, 1943.

[24] Morettin, P. A., Toloi, C. M.. **Análise de Séries Temporais**, *Edgard Blücher*, 2 edição rev. e ampl. São Paulo, 2006.

[25] Moro, J. Z.. **Sistema com Sensor Infravermelho para Monitoramento de Usuários de Transporte Público**, *Projeto de Graduação*, Universidade Federal do Espirito Santo - UFES, Vitória, 2009.

[26] Oliveira, M.; Ebecken, N.; Santos, I.; Neves, C.; Caloba, L.; Oliveira, J.. **Modelagem da Maré Meteorológica utilizando Redes Neurais Artificiais: Uma Aplicação para a Baía de Paranaguá - PR. Parte 2: Dados Meteorológicos de Reanálise do NCEP/NCAR**, *Revista Brasileira de Meteorologia*, v. 22, n.1, 53-62, 2007.

[27] Oliver, J.; Fonseca, A.; Perez, J.; Vega, R.; Canetti, R.. **Implementation of Adaptive Logic Networks on a FPGA Board**, *Proc. SPIE*, vol. 3526, p. 264-273, Configurable Computing: Technology and Applications, 1997.

[28] Palma, J. A.. **Outliers em Séries temporais - Uma Abordagem no Domínio dos Modelos ARMA**, *Tese de Mestrado*, FC-UL, 1998.

[29] Papoulis, A.. **Probability, Random Variables, and Stochastic Processes**, *McGraw-Hill*, 1991.

[30] Pimentel, W. R. O.. **Aplicação de Redes Neurais Artificiais e de Quimiometria na Modelagem do Processo de Craqueamento Catalítico Fluído**, *Tese de Doutorado*, Campinas, SP:[s.n], 2005.

[31] Pipa, D.; Silva, E.; Plagiari, C.. **Correção de Não-Uniformidade em Vídeo Infravermelho por Gradiente Descendente**, *XXVI Simpósio Brasileiro de Telecomunicações*, p. 2-5, Rio de Janeiro, 2008.

[32] Ramesh, J.; Chen, D.. **A Robust Back Propagation Learning Algorithm for Function Approximation**, *IEEE Trans. Neural Networks*, Vol 5, 1994.

[33] Reed, R.. **Pruning Algorithms - A Survey**, *IEEE Transactions on Neural Networks*, p. 740-747, 1993.

[34] Rosenblatt, F.. **The perceptron: a Probabilistic Model for Information Storage and Organization in the Brain**, *Psychological Review*, vol. 65, p. 386-408, 1958.

[35] Rousseeuw, P. J.; Leroy, A. M.. **Robust Regression And Outlier Detection**, *Wiley-Interscience*, 1ª edição, p. 25-27,2003.

[36] Ribeiro, C. V.. **Um ambiente para previsão de séries temporais utilizando comitês de aprendizado**, *Dissertação (mestrado) - Instituto Militar de Engenharia*, Rio de Janeiro, 2009.

[37] Ruas, G.; Bragatto, T.; Lamar, M.; Aoki, A.; Rocco, S.. **Previsão de Demanda de Energia Elétrica Utilizando Redes Neurais Artificiais e Support Vector Regression**, 2005.

[38] Silva, J.; Duarte, R.; Machado, M.; Medeiros, T.. **Aplicação de Modelos Neurais para Predição do Mapa de Energia de uma Rede de Sensores Sem Fio**, *XXV Congresso da Sociedade Brasileira de Computação*, 2005.

[39] Soares, P. P.; Nadal, J.. **Aplicação de uma Rede Neural Feedforward com Algoritmo de Levenberg-Marquardt para Classificação de Alterações do Segmento ST do Eletrocardiograma**, *IV Congresso Brasileiro de Redes Neurais* pp. 888-999, ITA, São José dos Campos - SP, 1999.

[40] Veiga, N.; Souza, C.; Gasparetto, D.; Barbosa, F.; Barreiros, A.; Soffiatti, N.. **Classificação de Dados Botânicos e Geomorfológicos, Utilizando Redes Neurais Artificiais, Aplicados a Análise Ecoepidemiológica da Doença de Chagas em Abaetetuba, Barcarena e Bragança, no estado do Pará no período de 2000 a 2006**, *XXVIII Congresso da Sociedade Brasileira de Computação*, 2008.

[41] Wieland, A., Leighton, R.. **Geometric Analysis of Neural Network capabilities**, *First IEEE International Conference on Neural Networks*, vol. III, p. 385-392, 1987.

Apêndice A

Levenberg - Marquardt

O algoritmo de Levenberg-Marquardt (LM) é uma técnica interativa que localiza o mínimo de uma função multivariada, que é expressa como dos quadrados de uma função não-linear. Na computação matemática, este algoritmo provê uma solução numérica para o problema de minimização de função.

O algoritmo LM é uma combinação do algoritmo de Gauss-Newton e do método do gradiente descendente. Quando a solução corrente está longe da solução desejada, o algoritmo se comporta como um método de gradiente descendente: lento, mas com garantia de convergência. Quando a solução corrente encontra-se próxima da solução desejada, o algoritmo comporta-se como o algoritmo de Gauss-Newton.

A.0.1 O algoritmo de LM

Deixe f ser uma relação funcional que mapeia o vetor de parâmetros $p \in \Re^m$ para um vetor estimado $\hat{x} = f(p)$. O algoritmo procura minimizar a distância quadrática $\epsilon^T\epsilon$, $\epsilon = x - \hat{x}$. A base do algoritmo LM é uma aproximação de f nas vizinhaças de p. Para pequenos valores $||\delta_p||$, uma expansão da série de Taylor leva à aproximação:

$$f(p + \delta_p) \approx f(p) + \mathbf{J}\delta_p,$$

onde $\mathbf{J}$ é a matrix Jacobiana $\delta f(p)/\delta p$. O algoritmo LM é iterativo: Iniciando no ponto p_0, o método produz uma série de vetores $p_1, p_2, ...$, que converge em direção ao minimizador local p por f. Então, a cada passo é necessário encontrar o δ_p que minimiza a quantidade $||x - f(p + \delta_p)|| \approx ||x - f(p) - \mathbf{J}\delta_p|| = ||\epsilon - \mathbf{J}\delta_p||$. O procurado δ_p é, portanto, a solução de um problema de mínimos quadrados: o mínimo é alcançado quando $\mathbf{J}\delta_p - \epsilon$ é ortogonal à coluna do espaço de $\mathbf{J}$:

$$\mathbf{J}^T\mathbf{J}\delta_p = \mathbf{J}^T\epsilon.$$

A matriz $\mathbf{J}^T\mathbf{J}\delta_p$ é a aproximação da Hessiana, ou seja, é uma aproximação de uma matriz de derivadas de segunda ordem. O método LM na realidade soluciona uma pequena variação da Equação $\mathbf{J}^T\mathbf{J}\delta_p = \mathbf{J}^T\epsilon$, conhecida como equação normal aumentada:

$$\mathbf{N}\delta_p = \mathbf{J}^T\epsilon,$$

onde os termos fora da diagonal de $\mathbf{N}$ são idênticos aos correspondentes elementos de $\mathbf{J}^T\mathbf{J}\delta_p$ e os elementos da diagonal são fornecidos por $N_{ii} = \mu + [\mathbf{J}^T\mathbf{J}]_{ii}$ para algum $\mu > 0$. A estratégia de alterar os elementos da diagonal é conhecida como *amortecimento* e μ como um termo de *amortecimento*. Se a atualização dos parâmetros $\mathbf{p} + \delta_p$ produz uma redução no erro ϵ, a atualização é aceita e o processo é repetido com decréscimo no termo de *amortecimento*. Caso contrário, o o termo de *amortecimento* é incrementado, a equação normal aumentada é resolvida novamente e o processo repete-se até que o valor de δ_p que decrementa o erro seja encontrado. O processo de repetição de procura do δ_p que reduz o erro ϵ, corresponde a apenas uma iteração do algoritmo LM.

Em LM, o termo de *amortecimento* é ajustado em cada iteração de forma a assegurar a redução do erro ϵ. Se o termo de amortecimento é definido por valores grandes, a matriz $\mathbf{N}$ é quase diagonal e o passo de atualização do LM é próxima da direção da descida mais íngrime. Se o termo de *amortecimento* é pequeno, o passo LM aproxima-se do exato passo quadrático de um problema totalmente linear. O LM é adaptativo porque controla o próprio *amortecimento*: ele incrementa o *amortecimento* se o passo falha em reduzir o ϵ, caso contrário, ele reduz o *amortecimento*. Desta forma o LM é capaz de alternar entre a aproximação lenta de uma descída íngrime quando a solução encontra-se longe do mínimo e produz uma convergência rápida quando a solução encontra-se nas vizinhanças do mínimo.

Apêndice B

Probabilidade Bayesiana

A maneira clássica de se usar probabilidades é em termos da frequência de repetição de eventos randômicos. Porém, em termos Bayesianos, a probabilidade provê uma quantificação da incerteza. Considere como um evento incerto, por exemplo, se a camada de gelo terá desaparecido do Ártico no final do século. Este não é um problema que pode ser repetido inúmeras vezes, com o objetivo de permitir a definição de probabilidade em termos clássicos. Contudo, é possível ter-se uma ideia, por exemplo, o quão rápido imaginamos que a camada de gelo esteja derretendo. Se agora, obtermos uma evidência atualizada, como uma imagem de satélite provendo novas informações para diagnóstico, poderemos revisar nossa opnião sobre a taxa de derretimento do gelo. Nossa avaliação de tal questão afetará as ações que iremos tomar, por exemplo, aumentar o esforço para reduzir a emissão de gases que produzem o efeito estufa. Nestas cirscunstâncias, gostaríamos de quantificar nossa expressão de incerteza e fazer uma revisão precisa dessa incerteza sob a luz de novas evidências.

O teorema de Bayes, sob uma nova ótica, pode ser usado para converter uma probabilidade anterior (*a priori*) em uma probabilidade posterior, através da incorporação de uma evidência fornecida por uma observação. Podemos fazer uma aproximação similar quando fazemos inferências sobre quantidades como o parâmetro w em um ajuste de curvas polinomial, por exemplo. Nós fazemos suposições sobre w antes de observarmos os dados, na forma de uma probabilidade a priori $p(w)$. O efeito sobre w de um conjunto de dados observados D é expresso em termos da probabilidade condicionada $p(D/w)$, desta forma o teorema de Bayes toma a forma

$$p(w|D) = \frac{p(D|w)p(w)}{p(D)},$$

que permite-nos avaliar a incerteza em w depois de termos observado D

na forma de uma probabilidade posterior $p(w|D)$.

A quantidade $p(D|w)$ pode ser vista como uma função do vetor w e neste caso é chamada de função de vizinhança, considerando isso, podemos enunciar o teorema de Bayes nas palavras

$$\textit{posterior} \ \alpha \ \textit{vizinhança} \text{ x } \textit{prior},$$

onde todas as quantidades podem ser vistas como função de w, .

Uma vantagem do ponto de vista Bayesiano é que a inclusão do conhecimento prévio surge naturalmente. Porém, uma crítica comum a aproximação Bayesiana é que ela é frequentemente selecionada com base em alguma conveniência matemática, ao invés de um reflexo de qualquer conhecimento a priori.

Apêndice C

Regularização Bayesiana

Na técnica de Regularização Bayesiana, a função de custo F é definida como [30]:

$$F_{reg} = \alpha F + \beta F_w,$$

onde:

- F é a função típica para treinar redes neurais do tipo MLP, que é a soma dos erro quadráticos,
- F_w é a soma dos quadrados dos parâmetros (pesos e *bias*),
- α e β são parâmetros da função objetivo.

Na estrutura Bayesiana, os pesos são considerados variáveis aleatórias, portanto podem ser atribuídos à uma função de distribuição. Depois, os dados são tomados e a função densidade para os pesos pode ser antecipada de acordo com a regra de Bayes:

$$P(w|D, \alpha, \beta, M)\frac{P(D|w, \beta, M)P(w|\alpha, M)}{P(D|\alpha, \beta, M)},$$

onde:

- D representa o conjunto de dados,
- M é o modelo de rede neural,
- w é o vetor de pesos da rede,
- $P(D|w, \beta, M)$ é a função de probabilidade dos dados, dados os pesos w,

- $P(D|\alpha, \beta, M)$ é o fator de normalização, que garante que a probabilidade total seja igual a 1.

Se for assumido que o ruído no conjunto de treinamento é Gaussiano e a distribuição anterior para os pesos é Gaussiana, a densidade de probabilidade pode ser escrita como:

$$P(D|w, \beta, M) = \frac{1}{Z_D(\alpha)} exp(-\beta E_D), P(w|D, \alpha, \beta, M) = \frac{1}{Z_D(\alpha)} exp(-\alpha E_w),$$

onde:

- $Z_D(\beta) = (\pi/\beta)^{n/2}$,
- $Z_w(\alpha) = (\pi/\alpha)^{N/2}$.

Se substituirmos estas probabilidades na Equação de $P(w|D, \alpha, \beta, M)$, obtemos após algumas manipulações:

$$P(w|D, \alpha, \beta, M) = \frac{1}{Z_F(\alpha, \beta)} exp(-F(w)).$$

Nesta estrutura Bayesiana, os pesos ótimos deverão maximizar a probabilidade posterior $P(w|D, \alpha, \beta, M)$. Maximizando a probabilidade posterior é equivalente a minimizar a função objetiva regularizada $F = \alpha E_D + \beta E_w$.

Para otimizar os parâmetros da função objetivo α e β, agora vamos considerar a aplicação da regra de Bayes:

$$P(\alpha, \beta|D, M) = \frac{P(D|\alpha, \beta, M)P(\alpha, \beta|M)}{P(D|M)}.$$

Se assumirmos como uniforme a densidade anterior $P(\alpha, \beta|D, M)$ para os parâmetros α e β, então a maximização da posterior é realizada pela maximização da função de probabilidade $P(D|\alpha, \beta, M)$. Desde que todas as probabilidades tenham a forma Gaussiana, pode-se conhecer a forma da densidade posterior. Agora pode-se resolver a equação de $P(D|\alpha, \beta, M)$ como:

$$P(D|\alpha, \beta, M) = \frac{Z_F(\alpha, \beta)}{Z_D(\beta) Z_w(\alpha)}.$$

Note que se conhece as constantes $Z_D(\beta)$ e $Z_w(\alpha)$. Porém, o $Z_F(\alpha, \beta)$ deve ser estimado por expansão com série de Taylor. Desde que a função objetivo tenha a forma quadrática em uma pequena área ao redos do ponto mínimo, podemos expandir $F(w)$ em volta do ponto mínimo da densidade

posterior w^{MP}, onde o gradiente é zero. Resolvendo para a constante de normalização obtemos:

$$Z_F = (2\pi)^{N/2}(det((H^{MP})^{-1}))1/2exp(-F(w^{MP})),$$

onde,
$H = \beta\nabla^2 E_D + \alpha\nabla^2 E_w$ é a matriz Hessiana da função objetivo.

Colocando este resultado, pode-se resolver para os valores de α e β no ponto de mínimo. Fazemos isto pegando a derivada em relação a cada logarítmo e igualamos a zero:

$$\alpha^{MP} = \frac{\gamma}{2E_w(w^{MP})} e \beta^{MP} = \frac{n-\gamma}{2E_D(w^{MP})}$$

onde:

- $\gamma = N - 2\alpha^{MP} tr(H^{MP})$ é chamado de número efetivo de parâmetros,
- N é o número total de parâmetros,
- γ é a medida de quantos parâmetros da rede são efetivamente usados na redução da função erro. Este parâmetro pode variar de 0 até N.

Em [11] foi proposto aplicar a aproximação de Gauss-Newton à matriz Hessiana, a qual pode ser convenientemente implementada se o algoritmo de otimização Levenberg-Marquadt é usado para localização do ponto mínimo. Este minimiza a computação adicional requerida para a regularização.