UNIVERSIDADE DE SÃO PAULO
FACULDADE DE ECONOMIA, ADMINISTRAÇÃO E CONTABILIDADE
DEPARTAMENTO DE ADMINISTRAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM ADMINISTRAÇÃO

PREVISÃO DE SÉRIES TEMPORAIS NO VAREJO BRASILEIRO: UMA
INVESTIGAÇÃO COMPARATIVA DA APLICAÇÃO DE REDES NEURAIIS
RECORRENTES DE ELMAN

Jorge Luis Durgante Pasquotto

Orientador: Prof. Dr. Ronaldo Zwicker

SÃO PAULO

2010

Prof. Dr. João Grandino Rodas
Reitor da Universidade de São Paulo

Prof. Dr. Reinaldo Guerreiro
Diretor da Faculdade de Economia, Administração e Contabilidade

Prof. Dr. Adalberto Américo Fischmann
Chefe do Departamento de Administração

Prof. Dr. Lindolfo Galvão de Albuquerque
Coordenador do Programa de Pós-Graduação em Administração

JORGE LUIS DURGANTE PASQUOTTO

**PREVISÃO DE SÉRIES TEMPORAIS NO VAREJO BRASILEIRO: UMA
INVESTIGAÇÃO COMPARATIVA DA APLICAÇÃO DE REDES NEURAIS
RECORRENTES DE ELMAN**

Dissertação apresentada ao Departamento de
Administração na Faculdade de Economia,
Administração e Contabilidade da
Universidade de São Paulo, como requisito
parcial para a obtenção do título de Mestre em
Administração.

Orientador: Prof. Dr. Ronaldo Zwicker

Versão corrigida
(a versão original encontra-se disponível na FEA/USP)

SÃO PAULO

2010

FICHA CATALOGRÁFICA

Elaborada pela Seção de Processamento Técnico do SBD/FEA/USP

Pasquotto, Jorge Luís Durgante

Previsão de séries temporais no varejo brasileiro : uma investigação comparativa da aplicação de redes neurais recorrentes de Elman / Jorge Luís Durgante Pasquotto. – São Paulo, 2010.
198 p.

Dissertação (Mestrado) – Universidade de São Paulo, 2010.
Orientador: Ronaldo Zwicker.

1. Redes neurais 2. Econometria 3. Varejo 4. Previsão (Análise de séries temporais) I. Universidade de São Paulo. Faculdade de Economia, Administração e Contabilidade. II. Título.

CDD – 006.32

À minha filha Bruna.

AGRADECIMENTOS

Ao final de um trabalho como este, ninguém pode alimentar a ilusão de que se trata de uma realização individual. É imprescindível agradecer a participação de muitas pessoas neste processo, mesmo que não seja possível nomeá-las integralmente nas linhas abaixo. A todas elas, expresso meu profundo reconhecimento, e com destaque especial:

Ao meu orientador, Prof. Dr. Ronaldo Zwicker, por acolher-me na FEA e guiar meus passos continuamente nesta jornada, sempre com muita sensatez e disponibilidade.

Ao Prof. Dr. Mauri A. Oliveira, por tantas horas cedidas incondicionalmente, compartilhando de forma generosa e entusiástica todo seu conhecimento no tema deste trabalho, tanto na parte conceitual quanto nas orientações práticas com os *softwares*.

Ao Prof. Dr. Henrique Castro, por suas valiosas observações na banca de qualificação.

Ao Prof. Dr. Fabiano Guasti de Lima, pela disponibilidade em sanar minhas dúvidas.

Ao Prof. Dr. Luis Paulo Favero, ao Prof. Dr. Nuno Fouto, e demais professores do Programa de Administração do Varejo - PROVAR/FIA, pela inspiração no processo de admissão na FEA. Da mesma forma, ao amigo e Prof. Dr. Denis Schiozer da UNICAMP.

Ao Prof. Dr. Abraham Yu, Prof. Dra. Alessandra Montini e Prof. Dra. Maria Aparecida Gouvea, por ministrarem suas aulas com vocação e apreço pela didática.

Aos funcionários da Biblioteca da FEA, especialmente Sra. Lucy Kawahara, e às funcionárias do setor do Programa de Comutação Bibliográfica (COMUT), por tornarem tudo mais fácil no meu acesso à literatura.

Aos meus colegas da FEA, especialmente Cláudia Mendes, Eduardo Contani, José Renato Kitahara e Ricardo Serra, pela contínua disposição na troca de experiências.

A Marina da Costa Manso Vasconcellos e Antonio Carlos Alves de Araújo, pelo incentivo, e apoio emocional quando minha própria rede neural deu sinais de saturação.

À empresa MAM Babyartikel pela disponibilidade, confiança e apoio financeiro.

Aos meus pais Nyrse e Telma, por me ensinarem desde cedo a valorizar a educação como forma de crescimento e desenvolvimento pessoal.

A minha esposa Edilusa, pela infinita paciência e ilimitada tolerância.

**“Eu vejo o futuro repetir o passado
Eu vejo um museu de grandes novidades
O tempo não para...”**

***Cazuza / Arnaldo Brandão
Letra da música “O Tempo não para”***

RESUMO

Neste trabalho foi explorada a aplicação de redes neurais recorrentes simples, também conhecidas como Redes de Elman, na previsão de três séries temporais mensais do varejo de bens e serviços no Brasil. As variáveis destas séries estão relacionadas com a demanda de produtos farmacêuticos, adubos, e tráfego aéreo. Em todos os casos a qualidade de previsão foi avaliada mediante cálculo do erro médio quadrático e do coeficiente de Theil no horizonte de previsão de 12 meses. As previsões seguiram o procedimento estático e recursivo (dinâmico). As previsões com Redes de Elman foram comparadas com as realizadas por modelos lineares sazonais obtidos através da metodologia de Box-Jenkins. A exatidão das previsões com as redes foi bem inferior, ou no máximo equivalente, às aquelas geradas pelos modelos lineares. Concluiu-se pela não superioridade das redes de Elman como método de previsão das três séries estudadas especificamente. A presença de sazonalidade e tendência nas séries motivou a observação de seu impacto na qualidade de previsão com as redes de Elman. Concluiu-se que a remoção da tendência e sazonalidade por meio de diferenças melhorou consideravelmente a qualidade de previsão comparada com aquelas obtidas com dados originalmente só padronizados no intervalo $[-1,1]$. As redes de Elman foram treinadas com o algoritmo de retropropagação no tempo com gradiente descendente e implementadas com o *software* MATLAB R2008b, mediante validação cruzada. Os modelos lineares foram obtidos com EVIEWS 6.0.

ABSTRACT

In this work we explored the application of simple recurrent neural networks, also known as Elman networks, in the prediction of three series of retail goods and services in Brazil. The series are formed by variables related to the monthly demand for pharmaceuticals, fertilizers and domestic air traffic. In all cases the quality of prediction was evaluated by calculating the mean square error and Theil coefficient in the forecast horizon of 12 months. The predictions followed the procedure static and recursive (dynamic). The forecast with Elman networks were compared with those performed by seasonal linear models obtained by Box-Jenkins methodology. The accuracy of predictions in the networks was much lower, or at most equivalent to those generated by linear models. Showed that there is no superiority of Elman networks as a forecasting method specifically addressed the three series. The presence of seasonality and trend in the series led to the observation of their impact on quality of forecast with Elman networks. It was concluded that the removal of trend and seasonality by differences improved the quality of prediction compared with those obtained with the original data only standardized in the interval $[-1,1]$. Elman networks were trained with the backpropagation through time algorithm with gradient descent and implemented with the MATLAB R2008b. Adopted early-stopping as cross-validation procedure. Linear models were obtained by Eviews 6.0.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS	04
LISTA DE SÍMBOLOS	05
LISTA DE TABELAS	06
LISTA DE GRÁFICOS	07
LISTA DE FIGURAS	08
1 INTRODUÇÃO E FORMULAÇÃO DO PROBLEMA DE PESQUISA	09
1.1 Motivação	09
1.2 Formulação da situação problema	09
1.2.1 Previsão de séries temporais	09
1.2.2 Métodos de previsão	10
1.2.3 Modelos de séries temporais	11
1.2.4 Redes Neurais Artificiais	13
1.3 Objetivos	15
1.4 Justificativa	16
1.5 Delimitação	17
1.6 Questões da pesquisa	18
1.7 Contribuições da pesquisa	19
1.8 Descrição dos capítulos	19
2 SÉRIES TEMPORAIS	21
2.1 Conceitos iniciais e notação	21
2.2 Comparação com dados em seção transversal	22
2.3 Linearidade	22
2.4 Componentes de uma série temporal	23
2.5 Estocasticidade em séries temporais	24
2.6 Estacionariedade	25
2.7 Ruído branco	25
2.8 Função de autocorrelação (FAC)	26
2.9 Função de autocorrelação parcial (FACP)	27
2.10 Previsão de séries temporais	28
2.10.1 Horizonte de previsão	29
2.10.1.1 Procedimento estático	29
2.10.1.2 Procedimento dinâmico	30
2.11 Qualidade da previsão	30
2.12 Domínios: tempo e frequência	32
2.13 Econometria de séries temporais	33
3 MODELOS LINEARES E METODOLOGIA DE BOX-JENKINS	35
3.1 Transformações	35
3.2 Modelos autoregressivos – AR(p)	37
3.3 Modelos de média móvel – MA(q)	38
3.4 Modelos mistos – ARMA(p,q)	39
3.5 Modelos integrados – ARIMA(p,d,q)	39
3.6 Modelos sazonais – SARIMA(p,d,q) x (P,D,Q)s	40
3.7 A metodologia de Box-Jenkins	41
3.8 Testes estatísticos úteis no procedimento de Box-Jenkins	45
3.8.1 Estacionariedade	45
3.8.2 Distribuição normal	47
3.8.3 Autocorrelação	48

3.8.4	Linearidade.....	48
4	REDES NEURAIS ARTIFICIAIS.....	51
4.1	O que são redes neurais artificiais?.....	51
4.1.1	O modelo de neurônio artificial	53
4.1.2	As funções de ativação	54
4.1.3	Arquiteturas de redes neurais artificiais	55
4.1.4	Tipos de treinamento	57
4.1.4.1	Algoritmo básico de retropropagação (BP – <i>backpropagation</i>)	58
4.1.5	Redes neurais e previsão de séries temporais.....	61
4.1.6	Aspectos gerais.....	61
4.2	A representação do tempo: memórias	62
4.2.1	Redes com atraso alimentadas adiante (TLFN)	66
4.2.2	Redes recorrentes dirigidas dinamicamente (<i>dynamically-driven</i>)	68
4.3	Redes de Elman.....	70
4.4	Algoritmos para treinamento de redes recorrentes.....	72
4.4.1	Retropropagação no tempo (BPTT – <i>backpropagation through time</i>)	72
5	METODOLOGIA DA PESQUISA	79
5.1	Aspectos teóricos.....	79
5.2	Desenvolvimento da pesquisa empírica	80
5.2.1	Previsão com modelos de séries temporais	80
5.2.2	Previsão com redes de Elman.....	81
5.2.2.1	Arquitetura da rede	81
5.2.2.2	Treinamento das redes.....	82
5.2.2.2.1	Validação cruzada	85
5.2.2.2.2	Dados originais e pré-processamento.....	86
5.2.2.2.3	Aspectos gerais.....	86
6	RESULTADOS EXPERIMENTAIS	91
6.1	Série mensal de tráfego aéreo – mercado doméstico	91
6.1.1	Metodologia de Box-Jenkins: AEREO	92
6.1.1.1	Transformações e testes de raízes unitárias.....	92
6.1.1.2	Modelo selecionado.....	94
6.1.2	Rede de Elman: AEREO	97
6.1.2.1	Dados sem pré-processamento de tendência e sazonalidade	97
6.1.2.2	Dados com pré-processamento.....	99
6.1.3	Comparação de previsões - AEREO	102
6.2	Série mensal do consumo de ADUBOS no Brasil	104
6.2.1	Metodologia de Box-Jenkins.....	105
6.2.1.1	Transformações e testes de raízes unitárias.....	105
6.2.1.2	Modelo selecionado.....	107
6.2.2	Rede de Elman: ADUBOS	110
6.2.2.1	Dados sem pré-processamento de tendência e sazonalidade	110
6.2.2.2	Dados com pré-processamento.....	112
6.2.2.3	Comparação de previsões - ADUBOS	115
6.3	Série da PMC: artigos farmacêuticos, médicos e outros artigos	116
6.3.1	Metodologia de Box-Jenkins.....	117
6.3.1.1	Transformações e testes de raízes unitárias.....	117
6.3.1.2	Modelo selecionado.....	119
6.3.2	Rede de Elman: FARMA	121
6.3.2.1	Dados sem pré-processamento de tendência e sazonalidade	121
6.3.2.2	Dados com pré-processamento.....	124

6.3.3	Comparação de previsões - FARMA	126
7	CONCLUSÃO	129
	REFERÊNCIAS	133
	APÊNDICES	145
	APÊNDICE 1	146
	APÊNDICE 2	147
	APÊNDICE 3	150
	APÊNDICE 4	153
	APÊNDICE 5	156
	APÊNDICE 6	159
	APÊNDICE 7	162
	APÊNDICE 8	165
	APÊNDICE 9	168
	APÊNDICE 10	171
	APÊNDICE 11	175
	APÊNDICE 12	178
	APÊNDICE 13	182
	APÊNDICE 14	185
	APÊNDICE 15	189
	APÊNDICE 16	190
	ANEXO A	191

LISTA DE ABREVIATURAS E SIGLAS

ADF: Augmented Dickey-Fuller – Dickey Fuller Aumentado
 AIC: Akaike Information Criteria – Critério de Informação de Akaike
 ANAC: Agência Nacional de Aviação Civil
 ANDA: Associação Nacional para Difusão de Adubos
 AR: Autorregressivo
 ARCH: Autoregressive Conditional Heterocedasticity – Autoregressivo com Heterocedasticidade Condicional
 ARIMA: Autoregressive Integrated Moving-average – Autorregressivo Integrado de Média Móvel
 ARMA: Autoregressive Moving-average – Autorregressivo de Média Móvel
 BP: Backpropagation – Retropropagação
 BPTT: Backpropagation Through Time – Retropropagação no Tempo
 DP: Dickey-Pantula
 EVIEWS® - marca registrada da empresa *Quantitative Micro Software (QMS)*.
 FAC: Função de Autocorrelação
 FACP: Função de Autocorrelação Parcial
 FF: Feedforward Network – Rede Alimentada à Frente
 GARCH: Generalized Autoregressive Conditional Heterocedasticity – Autoregressivo com Heterocedasticidade Condicional Generalizada
 IA: Inteligência Artificial
 IBGE: Instituto Brasileiro de Geografia e Estatística
 JB: Teste Jarque-Bera
 MA: Moving Average – Média Móvel
 MATLAB® - marca registrada da empresa *The Mathworks, Inc.*
 MLP: Multilayer Perceptron – Perceptron de Camadas Múltiplas
 MSE: Mean Squared Error – Erro Quadrático Médio
 NAR: Autorregressivo não Linear
 NARIMAx: Autoregressivo não Linear Integrado de Média Móvel com Entradas Exógenas.
 NARx: Autorregressivo com Entradas Exógenas
 Pax-km: unidade de medida, transporte de um passageiro na distância de um quilômetro.
 PP: Pré-processamento
p-value: menor nível de significância para o qual se rejeita a hipótese nula
 PMC: Pesquisa Mensal de Comércio
 RMLP: Perceptron Recorrente de Múltiplas Camadas
 RMSE: Root Mean Squared Error – Raiz Quadrada do Erro Quadrático médio
 RNA: Rede Neural Artificial
 RNE: Rede Neural de Elman
 RR: Rede recorrente
 RRDD: Rede Recorrente Dirigida Dinamicamente
 RTRL: Real Time Recurrent Learning – Algoritmo de Aprendizagem em Tempo Real
 RU: Raiz Unitária
 SARIMA: Seasonal Autoregressive Integrated Moving Average - ARIMA Sazonal
 SBC: Schwartz Bayesian Criteria – Critério de Informação de Schwartz
 SRN: Simple Recurrent Network– Rede Recorrente Simples
 TIC: Theil Inequality Coefficient – Coeficiente de Desigualdade de Theil
 TLFN: Time-lagged Feedforward Neural Network – Rede com Atraso Alimentada Adiante
 Vc: Valor Crítico de Testes Estatísticos

LISTA DE SÍMBOLOS

- $\mathbb{N}$: conjunto dos números naturais
 $\mathbb{N}^*$: conjunto dos números naturais, sem o zero
 $\mathbb{Z}$: conjunto dos números inteiros
 $\mathbb{R}$: conjunto dos números reais
 $E[X]$: valor esperado, ou esperança, de uma variável aleatória X
 ε_t : termo aleatório residual
 $N(\mu, \sigma^2)$: distribuição Normal de média μ , desvio-padrão σ , e variância σ^2
 ρ_k : função de autocorrelação na defasagem k
 γ_k : autocovariância de k -ésima ordem
 Δ : primeira diferença regular
 Δ^d : d -ésima diferença regular
 B^k : operador de atraso (*backshift*) de ordem k
 Δ_s^D : D -ésima diferença sazonal de período s
 $\chi^2(gl = m)$: distribuição Qui-quadrado com m graus de liberdade
 Q : estatística de Ljung-Box
 α : nível de significância estatística, equivalente à probabilidade de rejeitar a hipótese nula quando ela é verdadeira (erro do Tipo I)
 H_0 : hipótese nula
 H_1 : hipótese alternativa
 $\phi_k(.)$: função de ativação do k -ésimo neurônio
 $e_k(n)$: erro instantâneo na saída do neurônio k , no passo de tempo n
 $\phi'_k(.)$: derivada da função de ativação em relação ao seu argumento
 $w_{kj}(n)$: peso sináptico do neurônio k excitado pelo j -ésimo elemento de entrada, no instante n
 $\Delta w_{kj}(n)$: ajuste do peso sináptico no passo de tempo n
 η : taxa de aprendizado
 Δ^{-1} : atraso unitário

LISTA DE TABELAS

Tabela 5.1 – Número de pontos característicos de cada intervalo	84
Tabela 5.2 – Sequências de cada intervalo.....	84
Tabela 6.1 – Teste de Dickey-Pantula em Log (AEREO)	93
Tabela 6.2 – Procedimento estático de previsão de AEREO com SARIMA e redes de Elman	103
Tabela 6.3 – Procedimento dinâmico de previsão de AEREO com SARIMA e redes de Elman	104
Tabela 6.4 – Teste de Dickey-Pantula em Log (ADUBOS)	106
Tabela 6.5 – Procedimento estático de previsão de ADUBOS com SARIMA e redes de Elman	115
Tabela 6.6 – Procedimento dinâmico de previsão de ADUBOS com SARIMA e redes de Elman	116
Tabela 6.7 – Teste de Dickey-Pantula em Log (FARMA)	118
Tabela 6.8 – Procedimento estático de previsão de FARMA com SARIMA e redes de Elman	127
Tabela 6.9 – Procedimento dinâmico de previsão de FARMA com SARIMA e redes de Elman	128

LISTA DE GRÁFICOS

Gráfico 3.1 – Correlogramas característicos de processo representado por modelo AR(1).....	42
Gráfico 4.1 – Funções de ativação: a) limiar b) sigmoide c) tangente hiperbólica.....	54
Gráfico 6.1 – Partição da série AEREO: Box-Jenkins.....	91
Gráfico 6.2 – Partição da série AEREO: redes neurais.....	92
Gráfico 6.3 – Análise dos resíduos: histograma.....	95
Gráfico 6.4 – AEREO sem pré-processamento: melhores treinamentos de cada tamanho de rede	97
Gráfico 6.5 – AEREO sem pré-processamento: treinamento da rede selecionada.....	98
Gráfico 6.6 – Previsão estática sem pré-processamento de AEREO	99
Gráfico 6.7 – Previsão dinâmica sem pré-processamento de AEREO	99
Gráfico 6.8 – AEREO com pré-processamento: melhores treinamentos de cada tamanho de rede	100
Gráfico 6.9 – AEREO com processamento: treinamento da rede selecionada	101
Gráfico 6.10 – Previsão estática com pré-processamento de AEREO.....	102
Gráfico 6.11 – Previsão dinâmica com pré-processamento de AEREO	102
Gráfico 6.12 – Partição da série ADUBOS: Box-Jenkins.....	105
Gráfico 6.13 – Partição da série ADUBOS: redes neurais.....	105
Gráfico 6.14 – Análise de resíduos: histograma	109
Gráfico 6.15 – ADUBOS sem pré-processamento: melhores treinamentos de cada tamanho de rede	110
Gráfico 6.16 – ADUBOS sem processamento: treinamento da rede selecionada	111
Gráfico 6.17 – Previsão estática sem pré-processamento DE ADUBOS	111
Gráfico 6.18 – Previsão dinâmica sem pré-processamento de ADUBOS	112
Gráfico 6.19 – ADUBOS com pré-processamento: melhores treinamentos de cada tamanho de rede	113
Gráfico 6.20 – ADUBOS com processamento: treinamento da rede selecionada.....	113
Gráfico 6.21 – Previsão estática com pré-processamento de ADUBOS.....	114
Gráfico 6.22 – Previsão dinâmica com pré-processamento de ADUBOS	114
Gráfico 6.23 – Partição da série FARMA: Box-Jenkins.....	117
Gráfico 6.24 – Partição da série FARMA: redes neurais.....	117
Gráfico 6.25 – Análise de resíduos: histograma	120
Gráfico 6.26 – FARMA sem pré-processamento: melhores treinamentos	122
Gráfico 6.27 – FARMA sem processamento: treinamento da rede selecionada.....	122
Gráfico 6.28 – Previsão estática sem pré-processamento de FARMA	123
Gráfico 6.29 – Previsão dinâmica sem pré-processamento de FARMA.....	123
Gráfico 6.30 – FARMA com pré-processamento: melhores treinamentos.....	124
Gráfico 6.31 – FARMA com processamento: treinamento da rede selecionada	125
Gráfico 6.32 – Previsão estática com pré-processamento de FARMA.....	126
Gráfico 6.33 – Previsão dinâmica com pré-processamento de FARMA	126

LISTA DE FIGURAS

Figura 3.1 – Passos da metodologia de Box-Jenkins	42
Figura 4.1 – Representação de um neurônio biológico	51
Figura 4.2 – Representação de rede neural e neurônio artificial	51
Figura 4.3 – Neurônio artificial	53
Figura 4.4 – Exemplos de arquitetura de rede: a) camada única b) multicamada	56
Figura 4.5 – Algoritmo de retropropagação	59
Figura 4.6 – Neurônio artificial linear recorrente	64
Figura 4.7 – Rede <i>Focused Time-Lagged Feedforward</i> (FTLFN)	66
Figura 4.8 – Filtro de memória de curto prazo	67
Figura 4.9 – Janelamento de uma série temporal	68
Figura 4.10 – Redes recorrentes para previsão de séries temporais	69
Figura 4.11 – Rede de Elman: representação genérica	71
Figura 4.12 – Exemplo de uma rede de Elman	71
Figura 4.13 – Desdobramento de rede recorrente em camadas alimentadas à frente ao longo de 3 períodos de tempo	73
Figura 4.14 – Panorama de uma função de erro com mínimos: local e global	77
Figura 5.1 – Rede de Elman: modelo geral utilizado, com uma camada intermediária	82
Figura 5.2 – Sequência de entradas e de saídas desejadas na rede em cada intervalo de dados	84
Figura 5.3 – Fluxograma de treinamento de uma rede com o procedimento de validação cruzada	88
Figura 5.4 – Fluxograma para obter a melhor rede e comparar suas previsões com Box-Jenkins	89
Figura 6.1 – Correlogramas: a) D1LOGAER b) D1D6DLOG	94
Figura 6.2 – Análise dos resíduos: correlograma	96
Figura 6.3 – Correlogramas: a) DLOGADUB b) DLOGADSZA12	107
Figura 6.4 – Estimação do modelo SARIMA $(([2],[7]),1,[2]) \times (0,1,2)_{12}$	108
Figura 6.5 – a) correlograma dos resíduos, b) correlograma de resíduos quadrados	109
Figura 6.6 – a) correlograma dos resíduos, b) correlograma de resíduos quadrados	119
Figura 6.7 – a) correlograma dos resíduos, b) correlograma de resíduos quadrados	121

1 INTRODUÇÃO E FORMULAÇÃO DO PROBLEMA DE PESQUISA

1.1 Motivação

A maior motivação desta dissertação foi aplicar métodos quantitativos de previsão, usando dados de séries temporais reais do varejo brasileiro de bens e serviços.

A importância de fazer previsões está associada com tomar decisões. Empresas e governos fazem orçamentos, alocam recursos ou traçam políticas públicas, com base na expectativa de comportamento futuro de variáveis que afetam suas atividades, tais como consumo, inflação, arrecadação ou incidência de doenças, por exemplo.

Há vários métodos de se fazer previsões, alguns mais intuitivos e de natureza subjetiva; outros mais objetivos, com base matemática e estatística. O fato de não haver um método único e ideal de previsão, aplicável a todas as situações, deixa em aberto um amplo espaço para a pesquisa científica da aplicabilidade e eficiência de cada técnica.

Assim, outra motivação importante deste trabalho foi a curiosidade de saber quão adequada seria uma técnica em particular – baseada nas Redes Neurais Artificiais de Elman – quando usada na previsão das séries temporais nacionais citadas inicialmente.

1.2 Formulação da situação problema

1.2.1 Previsão de séries temporais

A habilidade de fazer boas previsões sempre foi apreciada ao longo da história e a importância deste tema se reflete na diversidade da sua aplicação em diferentes disciplinas, variando da engenharia à economia; ou mais amplamente, das ciências exatas às ciências sociais. A área de previsão é uma das que mais tem recebido investimentos em pesquisa, e assim deve permanecer, já que muitas decisões dependem de estimativas precisas a respeito do futuro (ZHANG, 2004, p. 1).

Tome-se de exemplo, a indústria do turismo em que o planejamento de atividades é particularmente importante em razão da influência dos fatores econômicos conjunturais, da complexidade operacional do setor e da natureza perecível de seus serviços. Assentos vazios na aviação, ingressos sobrando em espetáculos e quartos ociosos em hotéis são casos críticos desta perecibilidade e justificam boa parte do empenho em desenvolver técnicas de previsão nesta indústria (ZHANG, 2004, p. 4).

Uma série temporal é uma sequência ordenada de observações. Sua principal característica é a interdependência entre seus valores, o que exige a preservação de sua ordem no tempo. Neste trabalho usaremos série temporal como sinônimo de outras expressões equivalentes presentes na literatura: série de tempo, série histórica ou ainda sucessão cronológica. Alguns autores preferem o termo sucessão cronológica por respeitar mais a diferença que em matemática se estabelece entre sucessão e série (MURTEIRA *et al*, 1993, p. 1).

Dados de séries temporais estão presentes em todos os campos e são de especial interesse na administração e economia onde seu registro costuma ocorrer em intervalos regulares de tempo, igualmente espaçados. A lista de exemplos é extensa, incluindo taxas demográficas, índices econômicos ou as vendas de um produto, cuja divulgação pode ser anual, mensal ou de medição mais frequente. Entre os objetivos do estudo das séries temporais estão: o entendimento do mecanismo gerador dos valores da série e a sua previsão (WEI, 2006, p. 1).

1.2.2 Métodos de previsão

Em geral, os métodos de previsão se encaixam em dois grandes grupos: qualitativos e quantitativos. Os métodos qualitativos são subjetivos e intuitivos, o que os torna de difícil reprodução, mas isto não os impede de serem os mais adequados em certas situações. Já os métodos quantitativos são essencialmente racionais, baseados em modelos matemáticos e estatísticos, e sua objetividade permite sua reprodução por quaisquer pessoas (ABRAHAM; LEDOLTER, 2005, p. 2).

Em se tratando de previsão, a disponibilidade de dados quantitativos sugere quase que naturalmente o uso de métodos igualmente quantitativos, sempre e quando haja dados suficientes para isto. Dentre os métodos quantitativos de previsão, podemos considerar duas

categorias bem distintas: os métodos explanatórios e os métodos de séries temporais (MAKRIDAKIS; WHEELWRIGHT, 1985, p. 38).

Os métodos explanatórios buscam uma relação entre a variável que queremos prever e outras variáveis que expliquem a variação da primeira. A regressão é aqui a técnica principal. Nela se constrói uma equação que expressa a relação instantânea entre duas ou mais variáveis. Por exemplo, podemos estimar o consumo de biscoitos, com base no seu índice de preços e no respectivo investimento da indústria em propaganda. A regressão, porém, não capta a interdependência da variável prevista com os seus próprios valores anteriores, algo que, neste caso, corresponderia ao impacto no consumo presente de biscoitos causado pela evolução do próprio consumo em meses anteriores. Para avançar neste aspecto seria necessário combinar métodos explanatórios com modelos de séries temporais (ABRAHAM; LEDOLTER, 2005, p. 3).

Os métodos de séries temporais assumem que há um mecanismo gerador da série, que atua recorrentemente, e que a extração deste mecanismo torna-se possível tão somente com a própria série histórica de dados. Ao contrário do que ocorre nos modelos explanatórios, não há variáveis explicativas nos modelos de séries temporais. Apenas o tempo, em sentido muito restrito, poderia ser considerado a única variável explicativa de um modelo de série temporal (MAKRIDAKIS; WHEELWRIGHT, 1985, p. 39).

Este é um ponto chave no contexto desta dissertação. Abordaremos aqui a previsão de séries temporais utilizando unicamente os valores conhecidos de cada série, sem lançar mão de nenhuma teoria explicativa, seja de natureza econômica ou diversa. Tal como dito em Morettin e Tolo (2006, p. 7), deixaremos os “dados falarem por si”, apenas com procedimentos estatísticos, sem recorrer a uma possível teoria subjacente.

1.2.3 Modelos de séries temporais

A noção de tempo tem um papel fundamental na compreensão do mundo ao nosso redor. Com frequência, um fato ocorrido em determinado momento está relacionado ou depende de acontecimentos de outro instante. A análise de séries temporais estuda justamente esta relação de dependência entre os valores da série buscando uma forma prática, um modelo, que

expresse tal relação (PANKRATZ, 1991, p. 1).

Fava (2000, p. 199) observa que até o princípio dos anos 1970, a maior parte dos trabalhos envolvendo séries temporais procurava decompô-las em uma combinação de padrões: os componentes de nível horizontal, tendência, ciclo, sazonalidade e termo errático. Nesta abordagem, o mecanismo gerador das séries de tempo é de natureza determinista (exata e previsível, como as leis da física clássica) e apenas acrescido de uma mera distorção aleatória na prática.

Enders (2004, p. 2) menciona que na prática a tendência e a sazonalidade não têm natureza determinista. De acordo com Caldwell (2006, p. 5), o conceito determinista em séries temporais é oriundo das ciências exatas, porém gradualmente se mostrou pouco adequado nas ciências sociais em geral.

A partir de 1970, floresceu uma outra abordagem na qual as séries temporais são resultantes de um mecanismo estocástico (totalmente aleatório): existe uma família de valores que a série pode assumir, aos quais se atribui uma distribuição de probabilidade em cada instante de tempo. Em situações experimentais pode-se obter várias realizações do mesmo processo estocástico. No entanto, em ciências sociais geralmente contamos apenas com uma única trajetória para análise e construção do modelo a partir de suas observações passadas. A questão é então como fazer isto (FAVA, 2000, p. 201-202).

A maneira mais difundida de tratar esta questão é a metodologia de Box e Jenkins (1970) que trouxe consigo uma popularização dos modelos ARIMA (*autoregressive integrated moving-average*) a tal ponto de serem confundidos com a metodologia. Estes modelos baseiam-se em equações de diferenças contendo termos estocásticos (*stochastic linear difference equations*), uma classe de equações lineares muito conveniente para modelagem de processos econômicos (ENDERS, 2004, p. 1-3, 48-49, 52).

Apesar do grande destaque dado aos modelos ARIMA, dada sua ampla divulgação e utilização, e aparente superioridade em várias situações, eles encontram limitação ao assumir que as relações entre suas variáveis são descritas por equações lineares, o que nem sempre é adequado. Enders (2004, p. 387) cita, como exemplo, o caso das taxas de desemprego que

costumam crescer rapidamente em recessões, mas caem lentamente na recuperação e expansão da economia. Adicionalmente, muitas séries apresentam heterocedasticidade condicional, ou seja, variância condicionada ao longo do tempo.

A fim de lidar melhor com situações de não linearidade, muitas outras abordagens foram desenvolvidas, tais como: regressão não linear, modelo autorregressivo com “*threshold*” (TAR), o modelo autorregressivo de heterocedasticidade condicionada (ARCH) e a sua generalização (GARCH). Estes novos métodos trouxeram melhores resultados de previsão em muitos casos específicos, mas sua aplicação é mais complexa e com baixa capacidade de generalização (GHIASSI *et al*, 2005, p. 341-342).

Além da suposição de linearidade, na aplicação de modelos ARIMA quase sempre é necessário transformar os dados originais para obter estacionariedade. Estacionar a série é uma suposição básica na metodologia de Box-Jenkins e significa, grosso modo, que o valor médio e a variância da série transformada não se alteram sistematicamente no tempo. A estacionariedade é atingida geralmente por meio de diferenciações da série ou outras transformações, que geralmente demandam habilidade técnica e experiência do pesquisador (GUJARATI, 2000, p. 11).

1.2.4 Redes Neurais Artificiais

Todos os métodos quantitativos vistos até aqui são paramétricos: assumem uma ou mais hipóteses que condicionam a aplicação do método. Por exemplo, para estimar os parâmetros de um modelo ARIMA, assumimos haver uma relação entre os valores da série que pode ser explicitada em uma equação predefinida como linear.

Há, no entanto, um método quantitativo não paramétrico em que um capturador automático aproxima as equações ideais sem deduzi-las explicitamente, e ainda pode fazer isto por meio de funções não lineares. O termo não paramétrico significa que não são feitas suposições prévias sobre o modelo estatístico dos dados de entrada. Esta é a abordagem de redes neurais artificiais (DE OLIVEIRA, 2003, p.1-2; HAYKIN, 2001, p. 29).

Uma Rede Neural Artificial (RNA) é formada por neurônios artificiais inspirados e

associados em analogia com neurônios biológicos do sistema nervoso. As entradas que cada neurônio artificial recebe são processadas por meio de uma função, gerando uma saída. Esta poderá ser um ponto final ou ser reintroduzida como entrada intermediária. (CORRAR *et al*, 2007, p. 440; HAYKIN, 2001, p. 36; GATELY, 1996, p. 1-8).

As RNAs têm aspecto multidisciplinar e sua aplicação se estende por campos diversos em situações de séries temporais, classificação e reconhecimento de padrões. Tiveram um ressurgimento nos anos 1980 embora suas raízes tenham sido semeadas por volta de 1940, inspiradas em sistemas biológicos e nas conexões dos neurônios do sistema nervoso, tendo sido estudadas inicialmente com computadores analógicos. Um histórico mais detalhado pode ser encontrado em Haykin (2001, p. 63-69) e Corrar *et al* (2007, p. 433).

Sob o ponto de vista computacional, as RNAs são entendidas como um modelo de processamento de dados que atua de forma distribuída, isto é, alguns dos neurônios atuam ao mesmo tempo em rede, de forma muito mais rápida que no processamento exclusivamente serial. As RNAs são capazes de armazenar a informação captada em seu ambiente através de um processo de aprendizagem em que a conexão sináptica entre os neurônios é fortalecida ou enfraquecida ajustando-se iterativamente o peso dado às conexões, segundo Haykin (2001, p. 28) e Corrar *et al* (2007, p. 435-438).

Zhang *et al* (1998) e Crone (2005) apontam características que fazem as RNAs se distinguirem positivamente quando comparadas com modelos tradicionais de decomposição, com modelos ARIMA e com modelos paramétricos não lineares. Algumas destas características são:

- RNA é um método orientado para o banco de dados, isto é, não exige muitas premissas a respeito do modelo em estudo. Isto pode ser bem adequado quando não temos uma teoria clara a respeito das relações que regem os dados como variáveis;
- RNAs aprendem a partir de exemplos e capturam intrinsecamente as relações funcionais entre os dados, mesmo que tais relações estejam ocultas, sejam desconhecidas, imperceptíveis, ou de difícil descrição.

- RNAs são capazes de generalizar para uma parte desconhecida da população o que aprenderam com uma amostra. Por isto, sua aplicação é extremamente adequada para problemas de previsão, com a vantagem de que podem generalizar também padrões não lineares.

A arquitetura das redes neurais artificiais é determinada pela forma como os neurônios se combinam entre si e pela existência ou não de realimentação. Realimentação significa que de alguma forma a saída de um neurônio é reprocessada como nova entrada daquele mesmo neurônio ou da rede, caracterizando uma rede recorrente (RR). Redes do tipo recorrente são de particular interesse na previsão de séries temporais já que a realimentação de seus neurônios permite a elas reprocessar de forma dinâmica o erro destas saídas e com isto lidar melhor com certas características de padrões não lineares se comparadas a redes alimentadas adiante (ZHANG, 2004, p. 3).

Dentre as redes recorrentes, uma das mais conhecidas é a arquitetura proposta por Elman em que neurônios ocultos contêm conexões recorrentes para uma camada de unidades de contexto que consiste de atrasos unitários. Os neurônios ocultos têm assim um registro das suas ativações passadas, o que capacita a rede a realizar tarefas de aprendizagem que se estendem no tempo (HAYKIN, 2001, p. 790).

Algumas transformações feitas previamente nos dados podem afetar bastante a qualidade da previsão. Estas transformações que chamaremos de pré-processamento dos dados merecem atenção, conforme descrito em Zhang e Qi (2005, p. 501-514) que recomendam sempre eliminar a sazonalidade e a tendência dos dados originais. Em outro artigo, Zhang e Kline (2007) relatam o impacto do pré-processamento na previsão de 756 séries temporais sazonais trimestrais: as RNAs geraram melhores previsões quando foram treinadas com dados dessazonalizados e sem tendência. Os autores apontam, contudo, outros estudos em que este pré-processamento em nada contribuiu.

1.3 Objetivos

O objetivo geral desta pesquisa é investigar a utilização de Redes Neurais de Elman (RNEs)

na previsão de séries temporais univariadas de demanda de bens e serviços. Como objetivos mais específicos, podemos especificar:

- a) Investigar a viabilidade de fazer previsões de séries temporais nacionais de varejo usando RNEs treinadas com os dados originais da série, sem transformações adicionais, exceto a padronização intrínseca ao método de RNAs;
- b) Avaliar retroativamente a qualidade destas previsões no espaço de 12 meses à frente do último valor usado para treinamento da rede;
- c) Verificar o efeito final na qualidade das previsões quando as redes são treinadas com dados submetidos a pré-processamento (transformações para estabilizar a variância, tendência e sazonalidade antes da padronização da série original);
- d) Comparar as previsões das RNEs com aquelas obtidas por meio de modelos lineares sazonais, autorregressivos, de média móvel, obtidos por Box-Jenkins.

1.4 Justificativa

Redes neurais recorrentes são menos estudadas na literatura econômica ou estatística sobre previsão de séries temporais do que as redes alimentadas à frente (*feedforward*) e a eventual superioridade de uma arquitetura em relação à outra nunca foi estabelecida (GOMES, 2005, p. 58; ZHANG, 2004, p. 2, 3).

Em uma rápida pesquisa no motor de buscas Google acadêmico encontramos 12,5 mil resultados para a expressão “redes neurais artificiais” e apenas 1,4 mil resultados para a expressão “redes neurais artificiais recorrentes”. Nesta mesma consulta a expressão “*artificial neural networks*” revelou 693 mil saídas, uma frequência drasticamente maior do que a da expressão “*recurrent artificial neural networks*” que gerou apenas 133 mil resultados.

Além da concentração do uso de redes alimentadas à frente, uma boa parte da literatura combinando redes neurais artificiais e séries temporais dedica-se com frequência a dados de natureza financeira em administração e economia. Há muitos trabalhos abordando o retorno de investimentos na bolsa de valores, preço de ativos financeiros e taxas cambiais. É o caso, por exemplo, de Zhang e Berardi (2001), Thawornwong e Enke (2004), Santos (2005), e Kim

e Shin (2007) que utilizaram redes alimentadas à frente com dados financeiros, enquanto Li *et al* (2004) usaram redes recorrentes mas mantiveram o foco em finanças. Por sua vez, autores como Balaguer *et al* (2008), Tsai *et al* (2009), Ghiassi *et al* (2005), Law e Pine (2004) abordaram a previsão de demanda de varejo, porém mantiveram a arquitetura das redes alimentadas à frente.

Complementarmente, nas ciências exatas abundam aplicações de RNAs em previsão de séries temporais nas áreas de hidrologia, oceanografia, meteorologia e engenharia elétrica como, por exemplo, em Chiang *et al* (2004), More e Deo (2003), West e Dellana (2004), Leone Filho (2006), Paschoalino *et al* (2007) e Ballini (2000).

Tudo isto contribui para justificar uma dissertação que explore a utilização de algum tipo de rede recorrente, como as redes de Elman, em aplicações voltadas para o varejo nacional com dados eminentemente brasileiros. Espera-se com isto contribuir para a diferenciação no tema, ainda que sem alimentar qualquer pretensão de ineditismo.

1.5 Delimitação

Entende-se por série temporal do varejo brasileiro uma série temporal univariada, cuja variável seja uma quantidade relacionada ao varejo de bens e serviços no Brasil. Foram selecionadas três séries temporais conforme a definição acima, como segue:

- a) Transporte de passageiros no mercado aéreo doméstico brasileiro;
- b) Fertilizantes entregues ao consumidor final no mercado brasileiro de adubos;
- c) Índice de vendas de produtos farmacêuticos e outros produtos no Brasil.

Todos os dados foram obtidos por meio de consulta a relatórios disponíveis nos *websites* da Agência Nacional de Aviação Civil (ANAC), Associação Nacional para Difusão de Adubos (ANDA) e Instituto Brasileiro de Geografia e Estatística (IBGE). As principais variáveis destas séries e suas respectivas definições operacionais foram:

AEREO = quantidade de passageiros transportada no mercado aéreo doméstico brasileiro e multiplicada pelos quilômetros voados na respectiva rota. A unidade desta variável é dada em passageiros-quilômetros (pax-km) conforme obtida nos relatórios anuais do transporte aéreo da ANAC (antigo Departamento de Aviação Civil (DAC)).

ADUBOS = quantidade de adubos e fertilizantes vendida e entregue pelos fabricantes aos consumidores finais destes produtos, isto é, agricultores e suas cooperativas. A unidade desta variável é medida em toneladas.

FARMA = índice de vendas no varejo de produtos farmacêuticos, médicos, ortopédicos, de perfumaria e de cosméticos divulgado na PMC/IBGE. Esta variável é um índice com base 100 correspondente à média de seus valores em 2003.

A PMC, Pesquisa Mensal de Comércio, produz indicadores que permitem acompanhar o comportamento conjuntural do comércio varejista no país, investigando a receita bruta de revenda nas empresas formalmente constituídas, com 20 ou mais pessoas ocupadas, e cuja atividade principal é o comércio varejista.

Em todas as três séries, os 12 meses do último ano com dados conhecidos foram tomados exclusivamente como horizonte de previsão e utilizados somente na avaliação da qualidade das previsões neste intervalo. A obtenção destas previsões se manteve restrita a dois modelos: redes neurais artificiais recorrentes de Elman e modelos lineares de séries temporais obtidos pela metodologia de Box-Jenkins.

1.6 Questões da pesquisa

Considerando a delimitação das três séries temporais já descritas anteriormente, surge a curiosidade de responder as perguntas formuladas a seguir.

Pergunta 1: Redes de Elman podem ser uma ferramenta eficiente de previsão para estas séries temporais?

Pergunta 2: A previsão com Redes de Elman pode melhorar com o uso de transformações que eliminem tendência e sazonalidade nestas séries?

Pergunta 3: A qualidade de previsão com Redes de Elman é melhor do que aquela feita com modelos lineares obtidos através da metodologia de Box-Jenkins, nestas séries?

1.7 Contribuições da pesquisa

Além da justificativa apresentada anteriormente, espera-se que esta dissertação possa contribuir como relato de experiência na aplicação comparativa de redes recorrentes de Elman e de modelos lineares. A literatura menciona fartamente a importância da habilidade do analista ao aplicar a metodologia de Box-Jenkins e a dificuldade de treinar redes neurais recorrentes, quer seja pela complexidade dos algoritmos, pelo amplo consumo de recursos computacionais, ou ainda pela carência de *softwares* com interface amigável (*user-friendly*) para um público não especialista, mas que anseia pela maior utilização de técnicas quantitativas no ambiente empresarial.

1.8 Descrição dos capítulos

Esta dissertação está dividida da seguinte maneira: neste primeiro capítulo introduzimos o tema, a motivação e a justificativa para um trabalho nesta área bem como a formulação do problema da pesquisa e perguntas a serem respondidas na investigação. No Capítulo 2, as características de séries temporais são abordadas, enquanto a metodologia Box-Jenkins é mostrada no Capítulo 3. No capítulo 4, as redes neurais artificiais são abordadas com destino às redes de Elman. No Capítulo 5, é descrita a metodologia da pesquisa e no Capítulo 6 são apresentados os resultados experimentais baseados nas três séries temporais de varejo. Fechamos com conclusões, debilidades e sugestões de continuidade no Capítulo 7, seguido de referências, apêndices e anexos.

2 SÉRIES TEMPORAIS

2.1 Conceitos iniciais e notação

Uma série temporal é uma sequência ordenada de observações colhidas ao longo do tempo. Em dados obtidos desta forma é comum notar, ou descobrir, que as observações em diferentes períodos estão relacionadas umas com as outras. Neste caso, valores de um período anterior influenciam ou ajudam a explicar os valores observados em um período posterior. Ehlers (2009) resume algumas particularidades deste tipo de dados:

- Observações correlacionadas: requerem técnicas específicas na sua análise;
- É imprescindível levar em conta a ordem, isto é, a sequência das observações;
- Devido à natureza sequencial, é mais difícil de lidar com observações perdidas ou discrepantes (técnicamente conhecidas como *outliers*);
- Aspectos adicionais, como tendência e sazonalidade, podem dificultar a análise;
- A seleção de modelos e sua interpretação podem ser tarefas bem complexas.

Embora menos usual, nada impede o uso do conceito básico de séries temporais em observações tomadas ao longo de outras dimensões, em vez do tempo. Por exemplo, em Gomes (2005) encontram-se dados obtidos de uma sequência de pontos ao longo de uma linha no solo de uma plantação agrícola, identificados pelo autor como *transects*.

Em teoria, uma série temporal pode ser uma função contínua ou discreta. Exemplos de cada caso são, respectivamente: o registro de um eletrocardiograma e de um índice periódico de inflação. Na prática, as séries são tratadas como discretas porque, mesmo no caso contínuo, sua análise estatística e seu tratamento computacional são feitos com valores digitais correspondentes a pontos ou intervalos discretos (MURTEIRA *et al*, p. 2).

As séries temporais são ditas multivariadas quando suas observações incluem duas ou mais variáveis. Em casos assim, Ehlers (2009) observa que várias séries correlacionadas devem ser analisadas em conjunto, ou seja, em cada instante tem-se um vetor de observações. Como exemplo, vide os históricos de vendas de um produto e de seu investimento promocional: caso

em que há duas variáveis observadas simultaneamente.

As séries trabalhadas nesta dissertação são formadas por n observações sucessivas de uma só variável (série univariada) denotadas por $y(t_1), y(t_2), \dots, y(t_n)$ coletadas nos instantes $t_1, t_2, \dots, t_n$, igualmente espaçados e discretos. Note-se que o fato do tempo ser tratado como variável discreta não obriga a variável da série temporal a ser discreta.

Alternativamente pode-se escrever $y_1, y_2, \dots, y_n$ para denotar a série de tamanho n cujas observações são tomadas sequencialmente em intervalos regulares de tempo $t_0+h, t_0+2h, \dots, t_0+nh$, onde $n \in \mathbb{N}^*$, $h \in \mathbb{Z}$, e $y_t \in \mathbb{R}$ para $1 \leq t \leq n$.

Em geral, ao escrever y_t estaremos nos referindo a um elemento qualquer da série y_t dada pela sequência $\{y_1, y_2, y_3, \dots, y_i, \dots, y_n\} = \{y_t\}_{t=1}^n$ ou mais compactamente $\{y_t\}$.

2.2 Comparação com dados em seção transversal

Para fins de análise estatística, é importante estabelecer a distinção entre dados de seção transversal (*cross-section*) e dados de séries temporais. Dados de seção transversal são coletados aproximadamente no mesmo momento, em condições relativamente constantes. Em *cross-sections* a ordem de coleta é irrelevante, pois a preocupação está em obter dados com observações independentes (FAVA, 2000, p. 199).

Os dados de uma série temporal, matéria-prima desta dissertação, resultam da observação ao longo do tempo de uma variável cujos valores são correlacionados serialmente, isto é, são dependentes. Por isso, é tão essencial preservar sua ordem.

2.3 Linearidade

Uma característica importante em séries temporais é o fato de que suas observações estão de alguma forma relacionadas com outras observações passadas. Quando a série é linear ela pode

ser adequadamente expressa por uma equação linear, por exemplo, algo do tipo $y_t = \alpha \cdot y_{t-1} + \beta \cdot y_{t-2} + \dots$ onde y_t é a série, e $\alpha, \beta, \dots$ são os parâmetros. Caso, contrário temos um série não linear. A relação linear é apenas a forma mais simples que pode descrever uma série (DE OLIVEIRA, 2003, p. 3; SAMARASINGHE, 2007, p. 437).

2.4 Componentes de uma série temporal

Dentro da abordagem mais tradicional, uma série temporal resulta da combinação de um subconjunto ou de todas as componentes a seguir:

- Tendência: uma componente que produz mudanças graduais e regulares no longo prazo, geralmente atribuídas ao crescimento prolongado da população ou da atividade econômica, ou a outros fatores sem impacto repentino. Pode ser linear ou assumir formas polinomiais: uma exponencial, por exemplo;
- Sazonalidade: movimento oscilatório recorrente, de periodicidade homogênea;
- Ciclo: movimento em torno da tendência, notadamente ligado a fatores econômicos mas que não é regular como a sazonalidade;
- Nível: corresponde à constante;
- Termo errático: movimento esporádico, irregular, que decorre de fatores puramente aleatórios.

Cabe destacar que, embora nem sempre todos os termos acima sejam incluídos no modelo da série, o termo errático deverá sempre estar contido no subconjunto tomado quando se tratar de série de natureza econômica.

A decomposição da série pode ser feita por meio do recurso de médias móveis, ou explicitando os componentes acima por meio de uma fórmula aditiva, multiplicativa ou ainda uma combinação mista. (FAVA, 2000, p. 200).

2.5 Estocasticidade em séries temporais

Quando a relação que há entre observações defasadas da série temporal é obtida matematicamente e de forma exata, temos uma série de natureza determinística. Por exemplo, os pagamentos mensais de um financiamento bancário feito com uma certa taxa de juros prefixada são obtidos por um cálculo puramente matemático no sentido de que não admite termos aleatórios em sua formulação.

No entanto, na maioria das séries reais de interesse social e econômico, a relação entre termos da série não é exatamente matemática e segue processos controlados por leis probabilísticas. Um processo deste tipo é sofisticadamente chamado de estocástico ou, mais simplesmente, um processo não determinístico.

Em vez de fazer uma definição mais formal de um processo estocástico, vamos nos concentrar aqui na aplicação do seu conceito: se uma série temporal for vista como fruto de um processo puramente estocástico, então ela será parte de uma trajetória dentre muitas que poderiam ser observadas para o mesmo processo gerador da série.

Morettin e Tolo (2006, p. 2) citam como exemplo as medidas de temperatura feitas por 24h em duas ocasiões diferentes no mesmo local: teríamos duas trajetórias distintas para a mesma variável temperatura, assumindo um processo gerador imutável.

Os valores de uma série estocástica são resultantes de uma função de distribuição de probabilidade (em cada instante) e a série registrada apenas é uma sequência das realizações possíveis instantaneamente do processo estocástico. No contexto, se as realizações instantâneas são correlacionadas ao longo do tempo, existe uma probabilidade conjunta daquele grupo seletivo de observações ocorrer formando uma das trajetórias possíveis. (ENDERS, 2004, p. 49).

Esta concepção de processo estocástico é muito diferente da abordagem de decomposição da série. Na decomposição, as séries resultam da combinação de funções deterministas do tempo, acrescidas do termo aleatório que apenas representa desvios pontuais da trajetória pré-determinada. No processo estocástico, a trajetória resulta de um processo 100% aleatório em

cada instante (FAVA, 2000, p. 201).

2.6 Estacionariedade

Uma série temporal y_t é dita fracamente estacionária quando atende todas as condições abaixo:

$$E[y_t] = \mu \quad \text{para todo } t = 1, 2, \dots, n$$

$$E[(y_t - \mu)^2] = \gamma_0 = \sigma^2 \quad \text{para todo } t = 1, 2, \dots, n$$

$$E[(y_t - \mu)(y_{t-k} - \mu)] = \gamma_k \quad \text{para todo } t = 1, 2, \dots, n \quad \text{com } t \neq k, \quad k \in \mathbb{Z}$$

Em síntese, tomadas em conjunto, as condições acima significam que a média μ , a variância σ^2 e a autocovariância γ_k da série são todos números finitos e não se alteram com o tempo.

As duas primeiras condições indicam que a média e a variância da função distribuição de probabilidade são invariantes no tempo. A terceira condição impõe que as autocovariâncias dependem apenas da defasagem k entre os pontos da série e independem do instante propriamente dito: a relação entre dois pontos da série, separados pela distância k , muda apenas com esta distância que separa as duas observações. Se tomarmos quaisquer outros dois pontos, separados pelo mesmo período de tempo, a covariância deles manter-se-á sempre a mesma (FAVA, 2000, p. 203; HAMILTON, 1994, p. 45-46; MATOS, 2000, p. 236).

De uma maneira geral, sempre que mencionarmos estacionariedade neste texto, estaremos nos referindo à sua modalidade fraca exposta acima. O conceito de estacionariedade forte afasta-se de nosso escopo.

2.7 Ruído branco

O ruído branco é um bloco construtivo básico de modelos estocásticos discretos de séries temporais que veremos mais adiante. Uma sequência $\{\varepsilon_t\}$ é um processo chamado de ruído

branco se cada valor da sequência tiver esperança nula, variância constante e não tiver correlação com nenhum dos demais valores da sequência.

Para ser um ruído branco, $\{\varepsilon_t\}$ deve atender todas as condições a seguir (DE OLIVEIRA, 2003, p. 10):

$$E[\varepsilon_t] = E[\varepsilon_{t-1}] = \dots = 0$$

$$E[(\varepsilon_t)^2] = E[(\varepsilon_{t-1})^2] = \dots = \sigma^2 < \infty$$

$$E[\varepsilon_t \cdot \varepsilon_{t-k}] = E[\varepsilon_{t-j} \cdot \varepsilon_{t-j-k}] = \dots = 0, \text{ para } t \neq k$$

Dito de outra forma, uma série é um ruído branco quando seus valores formam uma sequência de variáveis aleatórias independentes e identicamente distribuídas (i.i.d) com média e variância finitas. O ruído branco é dito gaussiano se a distribuição em questão for uma normal (curva de Gauss) tal que $\{\varepsilon_t\} \sim N(0, \sigma^2)$ (TSAY, 2001, p. 23).

2.8 Função de autocorrelação (FAC)

Uma importante característica de séries temporais é a autocorrelação, que implica na correlação do valor de uma observação, em um momento particular, com observações em instantes prévios. Em geral a autocorrelação é alta para eventos situados no passado imediato e diminui à medida que as observações vão se distanciando ao longo de vários passos de tempo. O número de passos que são relativamente significativos são conhecidos como *lags*. Se apenas o evento imediatamente anterior for importante para determinar a observação atual, temos $lag = 1$, e assim por diante. (SAMARASINGHE, 2007, p. 438).

A função de autocorrelação (FAC) de uma série y_t é dada por:

$$\rho_k = \frac{\sum_{t=k+1}^n (y_t - \bar{y})(y_{t-k} - \bar{y})}{\sum_{t=1}^n (y_t - \bar{y})^2}$$

ou

$$\rho_k = \text{Cov}[y_t, y_{t-k}] / \text{Var}[y_t] = \gamma_k / \gamma_0,$$

onde γ_k é a autocovariância de k-ésima ordem de y_t :

$$\gamma_k = E[(y_t - \mu)(y_{t-k} - \mu)], \quad k \in \mathbb{Z}$$

Observando as equações acima podemos verificar que:

$$\rho_0 = 1, \quad \rho_{-k} = \rho_k \text{ e } -1 < \rho_k < 1$$

A FAC é útil na caracterização de modelos lineares, cujo exemplo mais simples é a série ruído branco ε_t , para a qual $E[\varepsilon_t] = 0$ e $\rho_k = 0$ para todo $k \neq 0$.

2.9 Função de autocorrelação parcial (FACP)

A autocorrelação parcial expressa a correlação que existe entre dois pontos y_t e y_{t-k} de uma série temporal, desconsiderando a influência sobre y_t dos demais pontos, isto é, descontando o efeito dos elementos intermediários $y_{t-1}, y_{t-2}, \dots, y_{t-k+1}$ sobre y_t .

O coeficiente de autocorrelação parcial de ordem k, usualmente representado por ϕ_{kk} , é dado pelo último coeficiente β_{kk} de cada uma das autoregressões a seguir (DE OLIVEIRA; FÁVERO, 2003):

$$y_t = \beta_{11}y_{t-1} + \varepsilon_t$$

$$y_t = \beta_{11}y_{t-1} + \beta_{22}y_{t-2} + \varepsilon_t$$

...

$$y_t = \beta_{k1}y_{t-1} + \beta_{k2}y_{t-2} + \dots + \beta_{kk-1}y_{t-k+1} + \beta_{kk}y_{t-k} + \varepsilon_t$$

Assim:

$$\begin{aligned}\phi_{11} &= \beta_{11} \\ \phi_{22} &= \beta_{22} \\ &\dots \\ \phi_{kk-1} &= \beta_{kk-1} \\ \phi_{kk} &= \beta_{kk}\end{aligned}$$

De acordo com Matos (2000, p. 238), os coeficientes de autocorrelação simples e parciais seguem aproximadamente uma distribuição normal com média zero e variância $1/n$.

Portanto, para intervalos de confiança de 95%, os valores da FAC e da FACP estão situados entre os limites $\pm \frac{1.96}{\sqrt{n}}$, onde n é o tamanho da série.

2.10 Previsão de séries temporais

Para Enders (2004, p. 1-3), a tarefa da moderna econometria de séries temporais é desenvolver modelos relativamente simples capazes de prever, interpretar e testar hipóteses relativas a dados econômicos, pois em muitas situações é importante estimar a intensidade (ou pelo menos a direção) de certas variáveis, tendo por base suas observações conhecidas no passado. O processo tem início com a construção de um modelo que capture relações intrínsecas (o mecanismo gerador) entre os diversos eventos, no intervalo temporal conhecido. O modelo então é usado para prever observações através de extrapolação fora deste intervalo.

Se tomarmos o processo estocástico abordado acima, e tendo observado as primeiras t realizações do processo, poderemos pensar no valor esperado de y_{t+1} , y_{t+2} , ..., y_{t+k} condicionado aos valores de y_1 a y_t , sendo k o número de passos à frente ou horizonte de previsão. Este valor esperado, denotado por $E_t[y_{t+i} | y_t, y_{t-1}, \dots, y_1]$, ou $E_t y_{t+i}$, com i variando de 1 a k , também é chamado de média condicional, e este é o conceito que embasa a concepção mais moderna de previsão de séries temporais.

Métodos ingênuos (naïve)

O método mais simples de previsão, conhecido como método ingênuo (*naïve*), pressupõe que o passado deve se repetir no futuro. Assim, os efeitos de observações passadas são capturados

na observação atual da variável que passa a ser uma boa preditora dos próximos valores. Um modelo deste tipo, muito polêmico em finanças, é o chamado passeio aleatório ou *random walk model*: $y_t = y_{t-1} + \varepsilon_t$.

A previsão ingênua pode ser adequada quando a série possui um comportamento altamente imprevisível e, neste caso, a melhor previsão para o valor de amanhã é exatamente o valor de hoje. Contudo, na prática, é sabido que a inclusão de *lags* melhora a qualidade da previsão (SAMARASINGHE, 2007, p. 439; SARTORIS, 2003, p. 341).

2.10.1 Horizonte de previsão

Tomando-se as observações de uma série temporal y_t , podemos fazer sua previsão para o próximo instante de tempo ou para vários passos à frente. Se tomarmos um instante t , a previsão 1 passo à frente é denotada por $\hat{y}_{t+1}$, e no caso de vários passos à frente é dada por $\hat{y}_{t+h}$, com $h \in \mathbb{Z}$, onde h é o chamado horizonte de previsão.

Quando este horizonte é maior do que um passo à frente, e é usado sempre o mesmo modelo na previsão de cada passo, é possível seguir dois procedimentos distintos: o estático e o dinâmico

2.10.1.1 Procedimento estático

Previsão estática: a cada instante, considera-se conhecido o valor real da série no instante anterior. Assim, a cada instante a série é atualizada com o último valor conhecido, isto é, o valor real. Isto não significa que o modelo seja reconstruído com base na nova observação. Apenas reutiliza-se o mesmo modelo em uma nova sequência, atualizada com a informação real.

Como exemplo, em uma previsão estática 3 passos à frente, faríamos:

- Com a série $y_1, y_2, \dots, y_n$ obtém-se a previsão $\hat{y}_{n+1}$;
- A série torna-se $y_1, y_2, \dots, y_n, y_{n+1}$ com a adição do novo ponto conhecido, e com ela

obtem-se a previsão $\hat{y}_{n+2}$;

- Atualiza-se a série para $y_1, y_2, \dots, y_n, y_{n+1}, y_{n+2}$ e obtém-se a previsão $\hat{y}_{n+3}$.

2.10.1.2 Procedimento dinâmico

Previsão dinâmica: o modelo estima os passos futuros tomando por base suas próprias estimativas anteriores de cada passo à frente. Em outras palavras, as previsões vão sendo feitas recursivamente com a realimentação de previsões anteriores no mesmo modelo de previsão.

Repetindo o exemplo anterior, em uma previsão dinâmica, faríamos:

- Com a série $y_1, y_2, \dots, y_n$ obtém-se a previsão $\hat{y}_{n+1}$;
- Com $y_1, y_2, \dots, y_n, \hat{y}_{n+1}$ obtém-se a previsão $\hat{y}_{n+2}$;
- Com $y_1, y_2, \dots, y_n, \hat{y}_{n+1}, \hat{y}_{n+2}$ obtém-se a previsão $\hat{y}_{n+3}$.

Note-se que a previsão um passo à frente tem valores iguais tanto para o procedimento estático quanto para o procedimento dinâmico. Porém, a partir do segundo passo, a previsão estática passa a ter valor diferente da dinâmica, pois esta não se valerá da atualização real da série. A previsão dinâmica é menos exata, pois o erro de previsão se propaga em cada ponto previsto.

2.11 Qualidade da previsão

Tomando a série temporal y_t , no instante t , temos que o erro de previsão e_t , para cada um dos próximos instantes $t+1, t+2, \dots, t+h$, é dado pela diferença entre o valor estimado e o valor real da série, ou seja:

e_{t+1} é o erro de previsão 1 passo à frente.

$$e_{t+1} = \hat{y}_{t+1} - y_{t+1}$$

e_{t+h} é o erro de previsão h passos à frente.

$$e_{t+h} = \hat{y}_{t+h} - y_{t+h}$$

O erro total de previsão para todo o horizonte de previsão h é dado por E :

$$E = \sum_{i=1}^h e_{t+i}$$

Esta medida de erro E é pouco utilizada na prática porque a presença de sinais opostos nos erros e_{t+i} pode ser compensada e anular totalmente o somatório dos erros ao longo do intervalo de previsão. Para contornar este problema, é melhor elevar cada erro ao quadrado e obter o erro quadrático médio (MSE – *mean squared error*):

$$MSE = \sum_{i=1}^h e_{t+i}^2 / n$$

Baseando-se em Santos (2005, p. 52), De Oliveira (2007, p. 21) e De Lima (2004, p. 31) demos preferência nesta dissertação às duas medidas a seguir:

$$RMSE = \sqrt{\sum_{i=1}^h e_{t+i}^2 / n}$$

$$TIC = \frac{\sqrt{\sum_{i=1}^h (\hat{y}_{t+i} - y_{t+i})^2 / n}}{\sqrt{\sum_{i=1}^h \hat{y}_t^2 / n} + \sqrt{\sum_{i=1}^h y_t^2 / n}}$$

O RMSE (*root mean squared error*) é expresso na mesma unidade de medida da variável prevista, o que é uma vantagem do RMSE comparado com o MSE, já que ambos superam o problema da compensação dos sinais opostos dos erros.

O TIC (*Theil Inequality Coefficient*) é um índice de desigualdade que avalia o ajuste da série

prevista à série real. Notar que quanto menor é a divergência entre os valores previstos e reais da série, mais próximo de zero será o numerador da expressão do TIC e quanto maior for aquela divergência, o TIC aproximar-se-á de 1.

Desta forma, temos que $TIC = 0$ para um ajuste perfeito, com $0 \leq TIC < 1$.

Além disto, também é possível avaliar a qualidade da previsão através da análise de três componentes percentuais do MSE: a proporção de viés (*bias*) indica quanto a média da previsão se afasta da média dos valores reais; a proporção de variância nos diz quanto a variação da previsão se afasta da variação da série real no horizonte de previsão; e o percentual de covariância atua como um resíduo não sistemático da previsão. Desta forma, o que se espera de uma boa previsão são altas porcentagens de covariância e baixa participação de viés e de variância na composição do MSE.

2.12 Domínios: tempo e frequência

Há duas formas distintas de se analisar uma série temporal: no tempo e na frequência.

No domínio da frequência o interesse está na repetição, isto é, em aspectos periódicos que ocorrem na série decomposta em frequências. Esta forma de análise é mais comum em ciências exatas, mostra-se conveniente em processos determinísticos, e seus modelos trabalham com infinitos parâmetros (análise espectral).

Nesta dissertação, concentrar-nos-emos no domínio do tempo, cujo foco está na interdependência entre os valores posicionados em diferentes instantes do tempo, abordagem bastante conveniente em processos não determinísticos presentes em ciências sociais. Neste domínio usam-se modelos com um número finito de parâmetros, e sua ferramenta básica é a função de autocorrelação. (MORETTIN; TOLOI, 2006, p. 2; WEI, 2006, p. 3; MURTEIRA *et al*, 2000, p. 168).

2.13 Econometria de séries temporais

A Econometria é uma combinação de Matemática, Estatística e Economia, situando-se na interseção destas três disciplinas, e que trabalha com dados econômicos de vários tipos, inclusive os de séries temporais, cuja especificidade permite falar hoje em uma Econometria das Séries Temporais (FAVA, 2000, p. 199).

Há duas abordagens em econometria, uma delas mais tradicional dita abordagem clássica e a outra conhecida como abordagem inglesa por estar ligada ao economista inglês David Hendry. Na abordagem clássica, um modelo econométrico é uma simplificação da realidade econômica que é formulado sempre com base em uma Teoria Econômica, cabendo aos dados disponíveis apenas o papel de testar e validar o modelo fornecido pela teoria. Já na abordagem inglesa, faz-se o caminho contrário: o pesquisador parte somente dos dados disponíveis, muitas vezes sem uma teoria envolvida, e busca montar um modelo estatístico o mais adequado possível aos dados, acreditando que os dados falam por si (BRAGA, 2000, p. 13-17).

Neste contexto, a moderna econometria de séries temporais se identifica melhor com a abordagem inglesa ao usar a metodologia de Box-Jenkins, que, como veremos mais adiante, modela o mecanismo gerador da série a partir dos dados disponíveis, sem nenhum compromisso prévio com hipóteses econômicas (BRAGA, 2000, p. 17).

3 MODELOS LINEARES E METODOLOGIA DE BOX-JENKINS

A metodologia de Box-Jenkins foi concebida para selecionar modelos lineares de séries temporais que podem combinar termos autorregressivos e de média móvel, regulares ou sazonais, desde que a série analisada seja fracamente estacionária. Se a série não é estacionária, é preciso transformá-la em uma outra que o seja. Neste capítulo veremos em mais detalhes como se faz isto dentro da metodologia de Box-Jenkins.

Considerando um processo estocástico, na maioria das vezes, o que observamos é apenas uma única série temporal do processo, sobretudo nos fenômenos econômico-sociais. A impossibilidade de fazer a história repetir-se traz problemas para desvendar o mecanismo gerador da série: fica difícil determinar as características momentâneas da função densidade de probabilidade, usando apenas uma observação do processo.

Então, quando se quer utilizar modelos para descrever séries temporais, é necessário introduzir a hipótese simplificadora da estacionariedade fraca da série, o que permite assumir que a média e variância instantâneas equivalem, respectivamente, à média e variância da série completa. Caso contrário, não seria possível generalizar para outros períodos o que sabemos da série a partir de um período conhecido (MORETIN; TOLOI, 2006, p. 23; DE OLIVEIRA, 2003, p. 10; FAVA, 2000, p. 202).

3.1 Transformações

Algumas transformações podem ser úteis em uma série originalmente não estacionária.

Logaritmo

Se a não estacionariedade estiver associada de alguma forma com a variância da série, pode-se aplicar o logaritmo, ou uma raiz quadrada, à série original. A transformação logarítmica contribui também para linearizar tendências exponenciais.

Todavia, um cuidado deve ser tomado posteriormente: se $y_t = \ln x_t$, a transformação inversa

do logaritmo de x_t não é imediata. Embora matematicamente $x_t = e^{y_t}$, estatisticamente esta operação inversa conduz a previsões viesadas de x_t com base em $\hat{y}_t$. Para contornar isto, é necessário introduzir na equação $\hat{x}_t = e^{\hat{y}_t}$ uma correção que considera a variância do erro na previsão de y_t (MORETTIN; TOLOI, 2006, p. 232-234; DE LIMA, 2004, p. 104).

Diferenciação

Quando uma variável y_t não é estacionária, podemos definir uma nova variável correspondente à primeira diferença Δy_t . Se a estacionariedade não for atingida com a primeira diferença, podemos fazer novas diferenças das diferenças, sucessivamente.

A primeira diferença de uma série é dada por:

$$\Delta y_t = y_t - y_{t-1}$$

A d-ésima diferença de uma série y_t é dada por:

$$\Delta^d y_t = \Delta^{d-1} y_t - \Delta^{d-1} y_{t-1}$$

Por comodidade de notação, define-se o operador de atraso B (*backshift*), dado por:

$$By_t = y_{t-1}$$

Tal que:

$$B^2 y_t = BBy_t = By_{t-1} = y_{t-2}$$

$$B^n y_t = y_{t-n}$$

$$\Delta y_t = y_t - y_{t-1} = y_t - By_t = (1 - B)y_t$$

Diferenciação sazonal

As diferenciações sazonais de período s são definidas como segue:

$$\Delta_s^1 y_t = \Delta_s y_t = y_t - y_{t-s} \quad (\text{primeira diferença sazonal com período } s)$$

$$\Delta_s^2 y_t = \Delta_s y_t - \Delta_s y_{t-s} \quad (\text{segunda diferença sazonal com período } s)$$

...

$$\Delta_s^D y_t = \Delta_s^{D-1} y_t - \Delta_s^{D-1} y_{t-s} \quad (\text{D-ésima diferença sazonal com período } s)$$

E o operador *backshift* sazonal, B_s , é tal que:

$$B_s y_t = y_{t-s}$$

$$B_s^2 y_t = B_s B_s y_t = B_s y_{t-s} = y_{t-2s}$$

...

$$B_s^n y_t = y_{t-n s}$$

$$\Delta_s y_t = y_t - y_{t-s} = y_t - B_s y_t = (1 - B_s) y_t$$

3.2 Modelos autorregressivos – AR(p)

Um modelo AR(p) relaciona os valores futuros com suas observações passadas.

A especificação de um modelo Autorregressivo (AR) é dada pela equação:

$$y_t = \varphi_1 y_{t-1} + \varphi_2 y_{t-2} + \dots + \varphi_p y_{t-p} + \varepsilon_t$$

Onde:

y_t corresponde à observação da série temporal no tempo t ;

φ_p corresponde ao parâmetro do modelo AR de ordem p ;

ε_t representa o resíduo, ou seja, o que não pode ser explicado pelo modelo.

Também, podemos escrever o processo AR(p) usando o operador *backshift* B :

$$y_t = \phi_1 B y_t + \phi_2 B^2 y_t + \dots + \phi_p B^p y_t + \varepsilon_t$$

$$y_t - \phi_1 B y_t - \phi_2 B^2 y_t - \dots - \phi_p B^p y_t = \varepsilon_t$$

Embora o operador B não seja uma variável, ele pode ser tratado algebricamente, e colocando y_t em evidência, temos que:

$$\underbrace{(1 - \phi_1 B - \phi_2 B^2 - \dots - \phi_p B^p)}_{\phi_p(B)} y_t = \varepsilon_t$$

$$\phi_p(B) y_t = \varepsilon_t$$

3.3 Modelos de média móvel – MA(q)

Nos processos de médias móveis, os valores da série y_t resultam de uma soma ponderada, isto é, uma média dos valores mais recentes dos erros aleatórios. O processo é uma combinação de choques passados e o modelo é definido conforme a equação:

$$y_t = \varepsilon_t - \vartheta_1 \varepsilon_{t-1} - \vartheta_2 \varepsilon_{t-2} - \dots - \vartheta_q \varepsilon_{t-q}$$

Onde:

ϑ_q corresponde ao parâmetro do modelo MA de ordem q ;

ε_t representa os choques, isto é, as diferenças entre os valores reais e os previstos da série em cada instante t .

Da mesma forma como fizemos com o processo AR(p), um modelo MA(q) também pode ser expresso com auxílio do operador B :

$$y_t = \underbrace{(1 - \vartheta_1 B - \vartheta_2 B^2 - \dots - \vartheta_q B^q)}_{\theta_q(B)} \varepsilon_t$$

$$y_t = \theta_q(B) \varepsilon_t$$

3.4 Modelos mistos - ARMA(p,q)

Uma combinação de termos AR e MA, dada pela equação abaixo, é um ARMA(p,q):

$$y_t = \underbrace{\varphi_1 y_{t-1} + \varphi_2 y_{t-2} + \dots + \varphi_p y_{t-p}}_{\text{Autorregressivo}} + \underbrace{\varepsilon_t - \vartheta_1 \varepsilon_{t-1} - \vartheta_2 \varepsilon_{t-2} - \dots - \vartheta_q \varepsilon_{t-q}}_{\text{Média móvel}}$$

Que pode ser escrito na forma compacta (SARTORIS, 2003, p. 346-350):

$$\phi_p(B)y_t = \theta_q(B)\varepsilon_t$$

3.5 Modelos integrados – ARIMA(p,d,q)

Se y_t torna-se estacionária após passar por d diferenças, e a série z_t que resulta destas diferenciações for um modelo do tipo ARMA(p,q), diz-se que y_t é descrito por um modelo ARIMA(p,d,q), tal que:

$$z_t = \varphi_1 z_{t-1} + \varphi_2 z_{t-2} + \dots + \varphi_p z_{t-p} + \varepsilon_t - \vartheta_1 \varepsilon_{t-1} - \dots - \vartheta_q \varepsilon_{t-q}$$

$$z_t = \Delta^d y_t$$

Um modelo ARIMA(p,d,q) pode ser expresso mais compactamente como segue:

$$\phi_p(B)(1-B)^d y_t = \theta_q(B)\varepsilon_t$$

Visto desta forma, um ARIMA(p,d,q) nada mais é do que um modelo ARMA(p,q) que sofreu uma transformação tal que implicou na substituição de $\phi_p(B)y_t$ pelo componente $\phi_p(B)(1-B)^d y_t$, e portanto, ambos os modelos gozam das mesmas características (PACHECO, 2001, p. 26).

3.6 Modelos sazonais - SARIMA(p,d,q) x (P,D,Q)s

Muitas séries temporais também apresentam uma componente sazonal importante. Uma série y_t com um componente sazonal de período s pode depender de seus valores passados na mesma estação, ou seja, $y_{t-s}, y_{t-2s}, \dots$. O efeito sazonal pode ser visto como um fenômeno estocástico como alternativa à abordagem determinista em que variáveis fictícias (*dummy*) são geralmente associadas a cada estação.

Em um modelo estocástico autorregressivo puramente sazonal SAR(P), de ordem P, uma série estacionária y_t é regredida nos seus valores anteriores defasados em múltiplos de s :

$$y_t = \phi_1 y_{t-s} + \phi_2 y_{t-2s} + \dots + \phi_P y_{t-Ps} + \varepsilon_t$$

Um modelo SMA(Q), puramente sazonal de médias móveis e de ordem Q, é dado por:

$$y_t = \varepsilon_t - \theta_1 \varepsilon_{t-s} - \theta_2 \varepsilon_{t-2s} - \dots - \theta_Q \varepsilon_{t-Qs}$$

A combinação de termos sazonais autorregressivos e de médias móveis pode ser expresso em sua forma geral como um modelo SARMA (P,Q)s:

$$y_t = \phi_1 y_{t-s} + \phi_2 y_{t-2s} + \dots + \phi_P y_{t-Ps} + \varepsilon_t - \theta_1 \varepsilon_{t-s} - \theta_2 \varepsilon_{t-2s} - \dots - \theta_Q \varepsilon_{t-Qs}$$

Portanto, tal como nos modelos regulares, pode-se demonstrar que se:

$$\Phi_P(B_s) = 1 - \phi_1 B_s - \phi_2 B_{2s} - \dots - \phi_P B_{Ps}$$

$$\Theta_Q(B_s) = 1 - \theta_1 B_s - \theta_2 B_{2s} - \dots - \theta_Q B_{Qs}$$

Para um SAR(P)s, tem-se:

$$\Phi_P(B_s) y_t = \varepsilon_t$$

E para um SMA(Q)s:

$$y_t = \Theta_Q(B_s) \varepsilon_t$$

Por fim, para um SARMA(P,Q)s:

$$\Phi_P(B_s) y_t = \Theta_Q(B_s) \varepsilon_t$$

Os modelos SARMA também podem ser usados em séries não estacionárias quando a D-ésima diferença sazonal conduz à estacionaridade da série transformada. Se isto acontecer, o modelo é sazonalmente integrado de ordem D e denominado SARIMA.

A combinação multiplicativa de componentes estocásticas regulares com sazonais é denotada por um modelo SARIMA (p,d,q) x (P,D,Q)_s e pode ser escrita compactamente:

$$\phi_p(B)(1-B)^d \Phi_P(B_s)(1-B_s)^D y_t = \theta_q(B)\Theta_Q(B_s)\varepsilon_t$$

Onde:

$\phi_p(B) =$ operador dos coeficientes regulares de autoregressão, de ordem p.

$(1-B)^d =$ operador d-ésima diferença regular

$\Phi_P(B_s) =$ operador dos coeficientes sazonais de autoregressão, de ordem P.

$(1-B_s)^D =$ operador D-ésima diferença sazonal, com periodicidade s

$\theta_q(B) =$ operador dos coeficientes regulares de média móvel, de ordem q

$\Theta_Q(B_s) =$ operador dos coeficientes sazonais de média móvel, de ordem Q

Como será visto mais adiante, o procedimento de obtenção de modelos SARIMA segue os mesmos passos da metodologia de Box-Jenkins empregados para achar o modelo ARIMA não sazonal. Isto significa que, no SARIMA, faz-se também a observância do comportamento da FAC e FACP, entretanto, olha-se para as defasagens sazonais (em séries mensais, por exemplo, as defasagens 12, 24, 36,...).

3.7 A metodologia de Box-Jenkins

Esta metodologia consiste no ajuste sistemático de modelos ARIMA aos valores de uma série temporal. Busca-se um modelo no qual os resíduos (diferença entre os valores observados e os gerados pelo modelo) formem uma série do tipo ruído branco.

A metodologia consiste dos seguintes passos:

Figura 3.1 – Passos da metodologia de Box-Jenkins
 FONTE: DE OLIVEIRA; FÁVERO, 2003.

Identificação

Inicialmente é preciso descobrir qual modelo ARIMA(p,d,q) representaria melhor a série. A escolha é feita entre várias possibilidades de (p,d,q) e pode ser feita com base no gráfico da função FAC e FACP. Este gráfico é denominado correlograma.

De uma maneira geral, as funções FAC e FACP de uma série temporal apresentam um comportamento característico de cada modelo linear AR, MA, ou ARMA:

Quadro 3.1 – Aspecto das funções FAC/FACP nos correlogramas

Modelo	FAC	FACP
AR	Declina rapidamente	Truncada em p
MA	Truncada em q	Declina rapidamente
ARMA	Declina rapidamente	Declina rapidamente

No caso de um processo AR(p) a FAC é declinante rapidamente e a FACP truncada exatamente na ordem p do processo. A FAC de um processo MA(q) é truncada na ordem do modelo, e a FACP declinante. De Oliveira e Fávero (2003, p. 9) ilustram os padrões de comportamento da FAC/FACP para um AR(1), conforme Gráfico 3.1.

Gráfico 3.1 – Correlogramas característicos de processo representado por modelo AR(1)
 FONTE: Adaptado de DE OLIVEIRA; FÁVERO, 2003.

A inspeção visual da função de autocorrelação (FAC) da série pode dar uma boa indicação da não estacionariedade. Quando a FAC decai lentamente, indicando o prolongamento da

interdependência dos valores ao longo de muitos *lags*, adota-se o procedimento de diferenciar a série progressivamente. A cada diferenciação verifica-se novamente o correlograma da FAC e o processo avança enquanto for necessário.

A condição para um modelo AR(p) ser estacionário é que $|\varphi_p| < 1$ para todo *lag* p.

Se algum destes coeficientes é igual a 1 temos pelo menos uma raiz unitária no AR, o que pode ser entendido de maneira simplificada como segue:

[...] Do ponto de vista estatístico, diz-se que uma série y_t tem raiz unitária se, numa equação que relacione y_t como variável dependente e seus próprios valores relativos ao período anterior, y_{t-1} , como variável explicativa, o coeficiente estimado associado a y_{t-1} for estatisticamente igual à unidade.” (MATOS, 2000, p. 237).

Estimação

Uma vez escolhidos os valores mais apropriados para p , d e q , estima-se os parâmetros da equação do modelo por meio de método estatístico adequado. No caso de modelos AR pode-se realizar a estimação pelo método dos mínimos quadrados através da minimização de $\sum \hat{\varepsilon}_t^2$, ou por máxima verossimilhança. Um modelo de médias móveis (MA) não pode ser estimado pelo método dos mínimos quadrados ordinários usualmente adotado nas regressões lineares, porque são incorporados no modelo os valores não observados dos erros aleatórios. A estimação, neste caso, é feita por máxima verossimilhança, ou por tentativa: atribuindo-se valores aos parâmetros em um procedimento iterativo de estimativas convergentes.

No caso de modelos MA(q) não há restrições de estacionariedade sobre os seus parâmetros, mas impõe-se a chamada condição de invertibilidade: $|\vartheta_q| < 1$. Por trás disto há o fato de que um processo MA finito sempre pode ser expresso por um modelo AR(∞) equivalente. Na estimativa dos parâmetros de um modelo MA, pode-se valer deste fato e aproximar um modelo AR finito. Como isto envolve recursividade nas estimativas, é necessário que o modelo AR seja estacionário para haver convergência (SARTORIS, 2003, p. 362; DE OLIVEIRA; FÁVERO, 2003; MATOS, 2000, p. 248-249).

Diagnóstico

Depois da etapa de estimação é necessário verificar a significância estatística dos parâmetros estimados e analisar os resíduos do modelo. Se o modelo estiver bem ajustado ao dados, ele explicará a maior parte da relação entre os valores da série deixando aos resíduos apenas a parcela não explicável.

Nesta etapa, o teste Q de Ljung-Box, que testa a hipótese nula de ausência de autocorrelação em uma série, é aplicado aos resíduos de modelos selecionados previamente: os resíduos de um modelo adequado devem ser desprovidos de autocorrelação, mantendo o aspecto de ruído branco (SARTORIS, 2003, p. 364-365).

Ao final desta etapa é possível que reste mais de um modelo candidato a representar o processo estocástico gerador da série. A comparação entre modelos de ordens diferentes só é válida mediante o uso dos chamados critérios de informação. Entre vários critérios existentes, seguem dois bastante utilizados:

Akaike information criterion (AIC)

$$AIC = \log_e \hat{\sigma}_\varepsilon^2 + \frac{2}{n}(p + q)$$

Schwartz Bayesian criterion (SBC)

$$SBC = \log_e \hat{\sigma}_\varepsilon^2 + \frac{\log_e n}{n}(p + q),$$

Onde: n = tamanho da amostra e $\hat{\sigma}_\varepsilon^2$ o estimador da variância dos resíduos.

Pelo princípio da parcimônia, escolhe-se o modelo que minimiza o critério adotado.

Previsão

Dada uma série y_t , se queremos prever os valores y_{t+1}, y_{t+2} até y_{t+h} , temos de encontrar o previsor ótimo $\hat{y}_{t+h}$, que minimiza o erro quadrático médio da previsão.

Ao fazer a previsão com um modelo ARMA(p,q), cada um dos termos MA de média móvel

vai sumindo gradualmente. O modelo vai se tornando passo a passo apenas um AR. Com isto, há perda de informações referentes aos choques aleatórios e redução no poder preditivo pelo fato da previsão cada vez se basear mais nas previsões anteriores.

3.8 Testes estatísticos úteis no procedimento de Box-Jenkins

3.8.1 Estacionariedade

A inspeção visual dos correlogramas nem sempre é conclusiva, pois, na prática, as funções amostrais não são tão perfeitas como as previstas teoricamente. Uma outra maneira possível de comprovar a estacionariedade é testar a presença de raízes unitárias na série. Esta comprovação não se dá de forma inequívoca, mas sim por meio de testes estatísticos associados a um nível de confiança β , equivalente a um nível de significância $\alpha = (1-\beta)$.

Teste Augmented Dickey-Fuller (ADF)

Dickey e Fuller (1981) criaram um procedimento básico para testar a presença de uma raiz unitária em y_t e que consiste em regredir a série contra seus valores defasados de um período, y_{t-1} . Em seguida, testa-se a significância estatística do parâmetro de y_{t-1} mediante comparação da estatística t da regressão com valores críticos obtidos da tabela de Dickey e Fuller (Apêndice 1) em três modelos de regressão linear:

- a) Equação sem intercepto e sem tendência;
- b) Equação com intercepto e sem tendência;
- c) Equação com intercepto e com tendência.

A versão aumentada deste teste (ADF – *Augmented Dickey-Fuller*) inclui as primeiras p diferenças de Δy_t na especificação da equação de regressão do teste, e no caso mais geral do item (c) acima, com constante de intercepto β_0 e tendência δ , é dada por:

$$\Delta y_t = \beta_0 + \delta t + \beta_1 y_{t-1} + \eta_1 \Delta y_{t-1} + \eta_2 \Delta y_{t-2} + \dots + \eta_p \Delta y_{t-p-1} + \varepsilon_t$$

Os valores críticos do teste ADF foram tabelados por Dickey e Fuller (1981) em simulações

de Monte Carlo, e posteriormente aperfeiçoados por Mackinnon (1991)(MATOS, 2000, p. 239-241; ENDERS, 2004, p. 190).

Teste de Dickey-Pantula (DP)

Dickey e Pantula (1987) apontaram que não é correto aplicar o teste ADF sucessivamente em várias diferenças da série. Isto porque a hipótese nula do teste ADF é de que há no máximo uma raiz unitária na série y_t , contra a hipótese alternativa de que não há raiz unitária. Inicialmente não se pode descartar a possibilidade de haver mais raízes na série que poderia assim ter múltiplas raízes unitárias.

Com o teste DP inicia-se testando a ordem máxima de integração presumida (ordem 2 ou 3, na prática) e havendo rejeição da presença do número máximo de raízes, testa-se a presença de uma raiz a menos, e assim sucessivamente usando a mesma estatística dos testes ADF. O teste Dickey-Pantula segue a sequência inversa daquela que, às vezes, é aplicada sem muito rigor com testes ADF em sucessivas diferenciações.

Tomando como exemplo o caso de se cogitar inicialmente 3 raízes unitárias, as equações do teste DP em cada passo estão listadas a seguir.

Passo 1: $H_0 = 3$ raízes unitárias, contra 2 raízes na hipótese alternativa.

- a) $\Delta^3 y_t = \beta_1 \Delta^2 y_{t-1} + \varepsilon_t$
- b) $\Delta^3 y_t = \beta_0 + \beta_1 \Delta^2 y_{t-1} + \varepsilon_t$
- c) $\Delta^3 y_t = \beta_0 + \delta t + \beta_1 \Delta^2 y_{t-1} + \varepsilon_t$

Passo 2: $H_0 = 2$ raízes unitárias, contra 1 raiz na hipótese alternativa.

- a) $\Delta^3 y_t = \beta_1' \Delta^2 y_{t-1} + \beta_2' \Delta y_{t-1} + \varepsilon_t$
- b) $\Delta^3 y_t = \beta_0' + \beta_1' \Delta^2 y_{t-1} + \beta_2' \Delta y_{t-1} + \varepsilon_t$
- c) $\Delta^3 y_t = \beta_0' + \delta' t + \beta_1' \Delta^2 y_{t-1} + \beta_2' \Delta y_{t-1} + \varepsilon_t$

Passo 3: $H_0 = 1$ raiz unitária, contra a hipótese alternativa de estacionariedade fraca.

- a) $\Delta^3 y_t = \beta_1 \Delta^2 y_{t-1} + \beta_2 \Delta y_{t-1} + \beta_3 y_{t-1} + \varepsilon_t$
- b) $\Delta^3 y_t = \beta_0 + \beta_1 \Delta^2 y_{t-1} + \beta_2 \Delta y_{t-1} + \beta_3 y_{t-1} + \varepsilon_t$
- c) $\Delta^3 y_t = \beta_0 + \delta t + \beta_1 \Delta^2 y_{t-1} + \beta_2 \Delta y_{t-1} + \beta_3 y_{t-1} + \varepsilon_t$

A ideia aqui é evitar o teste de uma única raiz unitária antes de fazer o teste de ordens de integração mais elevadas. O teste é encerrado quando se chega ao final da terceira etapa ou quando não é possível rejeitar a hipótese nula em algum passo anterior.

Em cada passo, a seleção das equações é dada inicialmente pela equação mais completa (c) que será substituída pelos modelos sem tendência (b), ou sem intercepto e sem tendência (a), caso não haja significância estatística dos coeficientes da tendência (δ, δ', δ) e do intercepto ($\beta_0, \beta_0', \beta_0$). Recomenda-se sempre incluir o intercepto no último passo porque séries de natureza econômica costumam ter média diferente de zero (DICKEY; PANTULA, 1987).

3.8.2 Distribuição normal

Teste Jarque-Bera (JB)

O teste JB testa a hipótese nula de um conjunto de dados ter distribuição normal de probabilidade. A distribuição normal possui assimetria zero, curtose 3, e equivale a uma distribuição χ^2 com 2 graus de liberdade. Se o valor da estatística do teste for maior do que o valor crítico (Vc) da distribuição χ^2 (gl=2), rejeita-se a hipótese nula de normalidade. A maneira equivalente de fazer isto é através do *p-value*: se ele for maior do que o nível de significância desejado, pode-se rejeitar a hipótese nula (SOARES; CASTELAR, 2003, p. 25).

A estatística JB do teste, aplicada a uma série y_t , é dada por:

$$JB = n \left[\frac{\hat{A}^2}{6} + \frac{(\hat{C} - 3)^2}{24} \right]$$

com:

$$\hat{A} = \frac{1}{n\hat{\sigma}^3} \sum_{t=1}^n (y_t - \bar{y})^3$$

$$\hat{C} = \frac{1}{n\hat{\sigma}^4} \sum_{t=1}^n (y_t - \bar{y})^4$$

O termo $\hat{A}$ é o coeficiente de assimetria, e $(\hat{C} - 3)$ é chamado excesso de curtose.

O teste JB é assintótico, isto é, tende para um certo valor à medida que a amostra cresce e, assim, é mais adequado para grandes amostras (DE LIMA, 2004, p. 39-40).

3.8.3 Autocorrelação

Teste Q de Ljung-Box

O teste de Ljung-Box testa a hipótese nula de ausência de autocorrelação mediante a comparação da estatística do teste com o Vc correspondente em uma distribuição χ^2 . A estatística Q do teste é dada por:

$$Q = n(n+2) \sum_{k=1}^m \frac{\rho_k^2}{n-k}$$

Se Q é menor do que o valor crítico tomado da distribuição χ^2 , pode-se assumir independência estatística nos dados, isto é, não se rejeita a hipótese nula ao nível de significância selecionado α . Visto de outra maneira, um *p-value* $< \alpha$ indica rejeição da hipótese nula e quando isto ocorre assume-se, na prática, a presença de autocorrelação nos dados (SARTORIS, 2003, p. 365).

3.8.4 Linearidade

Teste de McLeod-Li

A análise dos resíduos quadrados permite detectar se o modelo ARMA está bem ajustado aos dados. Sejam $(e_1, e_2, \dots, e_n)$ os resíduos de um modelo ARMA e $\hat{\rho}_{e^2}$ a autocorrelação do quadrado dos resíduos deste modelo. McLeod e Li (1983) propuseram a seguinte estatística

para detectar um modelo ARMA mal especificado:

$$Q = n(n+2) \sum_{k=1}^m \frac{[\hat{\rho}_{e^2}(k)]^2}{n-k}$$

Esta estatística equivale à aplicação do teste Q de Ljung-Box, aos quadrados dos resíduos do modelo ARMA, testando a hipótese nula de linearidade. A hipótese alternativa equivale a um modelo não linear do tipo ARCH (*autoregressive conditional heterocedasticity*) onde a variância é modelada por uma função não linear.

No teste de McLeod-Li, compara-se a estatística Q aplicada aos resíduos do modelo com o V_c extraído de uma distribuição χ^2 com m graus de liberdade. No contexto da metodologia de Box-Jenkins o que se busca é a não rejeição da hipótese nula para podermos assumir linearidade na prática. A não rejeição da hipótese nula se dá quando $Q < V_c$ para o nível de significância selecionado α , o que equivale a ter um $p\text{-value} > \alpha$ (MORETTIN; TOLOI, 2006, p. 502).

4 REDES NEURAIS ARTIFICIAIS

4.1 O que são redes neurais artificiais?

Redes neurais artificiais (RNAs) são estruturas computacionais que imitam parte do sistema nervoso biológico. A ideia foi reproduzir o processamento de informações no cérebro com um modelo artificial de neurônios que, tal como seus pares biológicos, se unem funcionalmente através de conexões e formam *redes neurais*.

Figura 4.1 – Representação de um neurônio biológico
FONTE: WASSERMAN, 1989.

Figura 4.2 – Representação de rede neural e neurônio artificial
FONTE: GARCIA, 2006, p. 19.

Cada neurônio artificial funciona como uma unidade autônoma de processamento, cuja tarefa individual é converter um sinal de entrada em outro sinal de saída. Com neurônios atuando em rede, a intensidade dos sinais é ampliada ou amortecida por meio de parâmetros atribuídos

às sinapses, chamados pesos sinápticos ou apenas pesos.

No contexto desta dissertação, uma rede neural artificial aprende a realizar uma tarefa através de um processo iterativo em que valores originais arbitrários dos pesos sinápticos são gradualmente ajustados até alcançar um objetivo. Este procedimento conhecido como algoritmo de aprendizagem costuma ser bastante trabalhoso na prática e costuma demandar uma simulação analógica ou uma programação em computadores eletrônicos digitais (HAYKIN, 2001, p. 28, 75).

As características das redes neurais artificiais podem ser então resumidas como segue:

- Processam a informação localmente em unidades (neurônios);
- Os pesos das conexões entre neurônios refletem a força de cada conexão;
- Extraem conhecimento dos dados disponíveis em um processo de aprendizagem cujo resultado final fica armazenado no conjunto de pesos;
- Seu processamento pode se dar em paralelo, ao invés de ser apenas sequencial;
- Capturada a essência de um processo, podem reproduzi-lo confiavelmente.

Entende-se por conhecimento os modelos utilizados por uma pessoa ou máquina para interpretar, prever e responder apropriadamente ao mundo exterior; ou ainda, a informação armazenada com estas características mencionadas (FISCHLER; FIRSCHEIN, 1987 *apud* HAYKIN, 2001, p. 49).

As redes neurais artificiais são expoentes da abordagem conexionista de Inteligência Artificial (IA) que se opõem aos Sistemas Especialistas (*Expert Systems*) criados por meio de técnicas simbólicas. Nestes sistemas constrói-se uma estrutura de regras quase linguística, que é aplicada nos dados posteriormente (*top-down*) enquanto os modelos conexionistas movem-se no sentido oposto: em vez de promover a representação simbólica de uma realidade externa preconcebida, extraem o conhecimento de forma adaptativa mediante entradas e saídas que interagem com o ambiente externo (ZIEMKE, 2001, p. 4-6; HAYKIN, 2001, p. 59-63).

Os pesos são os parâmetros das redes neurais artificiais e sua quantidade equivale aos graus de liberdade da rede. É esta estrutura flexível que torna as RNAs capazes de resolver vários

problemas do mundo real, pois elas reproduzem dois importantes atributos do sistema biológico: a capacidade de aprendizado e de generalização. Contudo, nunca é demais ressaltar que neurônios artificiais são sempre modelos simplistas se comparados com a complexidade da realidade biológica (SAMARASINGHE, 2007, p. 16; ABELÉM, 1994, p. 21).

4.1.1 O modelo de neurônio artificial

O modelo geral de um neurônio artificial é apresentado na figura a seguir, conforme extraído de Haykin (2001, p. 36-39):

Figura 4.3 – Neurônio artificial
 FONTE: Adaptado de HAYKIN. 2001, p. 36.

Onde:

m é o número de sinais de entrada do neurônio;

x_j é o j -ésimo sinal de entrada do neurônio;

w_{kj} é o peso associado com o j -ésimo sinal de entrada, no neurônio k ;

b é o limiar de cada neurônio, frequentemente chamado de *bias* na literatura de RNAs;

v_k é uma combinação ponderada dos sinais de entrada e do *bias*, no k -ésimo neurônio;

$\varphi(.)$ é a função de ativação, do k -ésimo neurônio.

O *bias* tem o efeito de aumentar ou diminuir a entrada líquida da função de ativação, conforme seu sinal muda de positivo para negativo. Com um pequeno ajuste ao modelo acima, é possível substituir o *bias* b_k por uma entrada fixa $x_0 = 1$, de forma que o valor do *bias* torna-se um novo peso sináptico $w_{k0} = b_k$. Assim, pode-se descrever matematicamente o

neurônio k através das equações:

$$v_k = \sum_{j=0}^m x_j \cdot w_{kj}$$

$$y_k = \varphi(v_k)$$

Onde v_k é chamado de campo local induzido, ou potencial de ativação, e a função de ativação define a saída $y_k(n)$ do k -ésimo neurônio em relação ao instante n como:

$$y_k(n) = \varphi_k(v_k(n))$$

4.1.2 As funções de ativação

A versatilidade das redes neurais se deve em boa parte ao fato da função de ativação poder ser linear ou não linear de acordo com sua devida conveniência. Uma rede que tenha apenas neurônios com função de ativação linear terá valor prático limitado pois suas saídas serão comparáveis a regressões lineares (ALMEIDA; NAKAO, 2007, p. 443).

Seguem algumas das funções não lineares mais conhecidas:

Gráfico 4.1 – Funções de ativação: a) limiar b) sigmóide c) tangente hiperbólica
 FONTE: LNCC (2010).

Função de limiar (*threshold*)

$$\varphi(v) = \begin{cases} 1 & \text{se } v \geq 0 \\ 0 & \text{se } v < 0 \end{cases}$$

É uma função descontínua e binária, cuja saída só pode ser 0 ou 1. Foi utilizada no modelo pioneiro de neurônio artificial criado por McCulloch e Pitts (1943) e tem a forma de um degrau.

Função sigmóide

$$\varphi(v) = 1 / (1 + e^{-v})$$

Onde: e representa a constante de Napier: 2,7182...

Também conhecida como função logística, é uma função contínua que tem a forma de um S que faz a transição gradual entre os dois extremos, variando de 0 a 1.

Função tangente hiperbólica

$$\varphi(v) = (1 - e^{-v}) / (1 + e^{-v})$$

É semelhante ao S da função sigmoide, mas varia no intervalo entre -1 e 1.

A função tangente hiperbólica apresenta características interessantes para as redes a serem utilizadas nesta dissertação: é uma função contínua, derivável em todos os seus pontos, o que é muito conveniente na aplicação de certos algoritmos de treinamento. Além disto, esta função admite valores negativos bastante comuns quando trabalhamos com séries temporais transformadas por diferenças da série original.

4.1.3 Arquiteturas de redes neurais artificiais

A arquitetura das RNAs varia bastante de acordo com sua finalidade. A maneira como os neurônios estão distribuídos em rede também está muito relacionada com o algoritmo de aprendizagem a ser utilizado. Em geral, a classificação de redes neurais artificiais considera a forma como o processamento flui na rede e o quanto seus neurônios estão dispostos em camadas. Estas classificações, detalhadas brevemente nas próximas linhas, são apenas algumas maneiras dentre muitas possibilidades (HAYKIN, 2001, p. 46-47).

Redes de camada única

São redes com neurônios dispostos em paralelo em uma camada única. Nas entradas destas redes há nós que não são neurônios (não existe computação do sinal) e o processamento prossegue em apenas uma camada de onde emergem as saídas da rede.

Redes multicamadas

São redes onde há uma ou mais camadas posicionadas entre os nós de entrada e a camada que gera as saídas finais da rede. Estas camadas escondidas, também chamadas de ocultas ou intermediárias, propagam os sinais até que eles cheguem à saída da rede em um certo momento. Rede multicamadas é a tradução de *multilayer network*.

Figura 4.4 –Exemplos de arquitetura de rede: a) camada única b) multicamadas

FONTE: FLORES, 2009, p. 47-48.

Redes alimentadas à frente (FF - *feedforward*)

Redes nas quais o processamento se dá sempre em sentido único, isto é, da entrada para a saída da rede, com ausência de qualquer realimentação. Tomando-se a representação gráfica horizontal mais usual, este processamento se dá partindo das entradas situadas à esquerda, movendo-se para as saídas à direita, e fundamentalmente sem retornos.

Estas redes também são referenciadas na literatura como redes alimentadas adiante, ou *feedforward networks* (HAYKIN, 2001, p. 46; DE OLIVEIRA, 2003, p. 50; SAMARASINGHE, 2007, p. 446).

Redes recorrentes

São redes onde há pelo menos um laço de realimentação (*feedback*), isto é, a saída de pelo menos um dos neurônios é reintroduzida em algum ponto anterior da rede configurando recorrência do processamento. Quando a realimentação se dá no próprio neurônio que a originou é chamada de autorrealimentação (HAYKIN, 2001, p. 48).

Tal como ocorre em redes alimentadas à frente, a rede recorrente pode ter uma única camada ou pode ter camadas ocultas (rede recorrente de múltiplas camadas). Algumas destas redes recorrentes levam o nome de seus criadores: Elman, Jordan e Hopfield.

4.1.4 Tipos de treinamento

O processo de treinamento, conforme já vimos, consiste em ajustar os parâmetros da rede interativamente e implica na seguinte sequência de eventos:

- a) A rede neural é estimulada pelo ambiente;
- b) Ela sofre modificações em seus pesos devido aos estímulos do passo anterior;
- c) A rede responde de um jeito novo ao ambiente por conta das modificações acima.

Há pelo menos dois paradigmas de aprendizado para redes neurais:

Aprendizado supervisionado

O aprendizado supervisionado é também conhecido como aprendizado com um professor porque ele funciona como se houvesse um professor na saída da rede que indica a resposta correta para cada exemplo: há um conjunto de dados de entrada que são apresentados à rede como exemplos e geram uma saída da rede que é comparada com a saída esperada (*target*). Obtém-se assim o erro correspondente a cada exemplo.

Considerando o neurônio k na saída de uma rede, temos que no passo de tempo n , o erro instantâneo $e_k(n)$ será dado por:

$$e_k(n) = d_k(n) - y_k(n)$$

Onde:

$d_k(n)$ = sinal de resposta desejada (*target*) do neurônio k , no instante n ;

$y_k(n)$ = sinal de saída obtido (*output*) do neurônio k , no instante n .

O conjunto de pares onde cada par é uma entrada associada à respectiva resposta correta é chamado amostra de treinamento.

O erro de saída é usado como parâmetro em um algoritmo iterativo de ajuste dos pesos, cuja intenção é diminuir progressivamente o erro na saída da rede até um mínimo aceitável. Algoritmo é um conjunto de regras bem definidas para achar a solução de um problema de maneira sistemática. O algoritmo de retropropagação (*backpropagation*) é um dos mais usados dentro deste paradigma supervisionado.

Aprendizado não supervisionado

É caracterizado pela ausência de indicação de uma saída correta desejada. Neste caso não cabe a metáfora do professor. Resumidamente, há duas maneiras de conduzir este tipo de aprendizado: por reforço e de forma auto-organizada (HAYKIN, 2001, p. 89).

4.1.4.1 Algoritmo básico de retropropagação (BP – *backpropagation*)

O algoritmo de retropropagação se aplica a redes neurais artificiais alimentadas adiante.

Este algoritmo de aprendizado supervisionado funciona percorrendo a função de erro, ou função de custo, na saída da rede, em busca de um ponto de mínimo. Os pesos sinápticos restam alterados após serem percorridas duas etapas:

- a) Propagação à frente: o sinal é propagado ao longo da rede iniciando pela primeira camada até gerar o erro de saída na última camada;
- b) Retropropagação: o erro é corrigido, camada a camada, alterando-se os pesos no sentido inverso.

Figura 4.5 – Algoritmo de retropropagação
 FONTE: LNCC (2010).

Apresentação dos exemplos à rede: ajuste incremental e batch

Com o algoritmo BP apresentam-se pares de entrada-saída para a rede, cada um deles é um padrão a ser aprendido, e há duas maneiras de se aplicar a correção dos pesos:

- a) Alteração incremental (por padrão, ou *on-line*): a alteração nos pesos é feita sempre que um novo par de entrada-saída é apresentado à rede gerando um erro de saída que é corrigido individualmente logo após cada par ser submetido à rede.

Neste caso, a função de erro instantâneo para o neurônio k , na saída da rede no instante

$$n, \text{ pode ser equacionada como } E = \frac{1}{2} e_k^2(n);$$

- b) Alteração por época (por lote, ou *batch*): todos os n pares da amostra de treinamento são apresentados para a rede, gerando-se um erro correspondente a todo o lote e só depois disto é feita a atualização dos pesos. Isto pode envolver a apresentação repetitiva do mesmo conjunto de treinamento por muitas vezes. Neste caso, a função de erro a ser

$$\text{minimizada com o ajuste dos pesos, usualmente é dada por } E = \frac{1}{2n} \sum_{i=1}^n e_k^2(n) \text{ de maneira}$$

análoga ao MSE visto no Capítulo 2.

A fração $\frac{1}{2}$ que aparece nas funções de erro é arbitrária e usada na comunidade estatística apenas por conveniência matemática, simplificando as expressões que resultam da aplicação de derivadas nestas funções (SAMARASINGHE, 2007, p. 48).

Épocas

O treinamento da rede, seja em lote, seja por padrão, prossegue com a repetição da apresentação dos padrões à rede. Cada vez que todos os exemplos de um lote são

apresentados à rede temos o que se chama de época de treinamento.

Assim, ao final de uma época, terminamos de apresentar todo o conjunto de treinamento e ao repetir a apresentação dos pares à rede estaremos percorrendo nova época de treinamento. Este processo repete-se enquanto as alterações nos pesos resultarem em redução significativa do erro de saída ou até que um critério de parada seja acionado como, por exemplo, ao atingir um número máximo prefixado de épocas.

No ajuste incremental os pesos podem oscilar devido ao fato do ajuste de pesos requerido por um exemplo de treinamento estar sujeito a ser cancelado por outro exemplo. A alternativa mais estável é esperar até que todos os exemplos, ou uma parte deles, sejam processados e só então ajustar os pesos de uma maneira global. Se todos os exemplos são apresentados o treinamento é dito por época e neste caso há um único ajuste de pesos após cada época (SAMARASINGHE, 2007, p. 50).

Regra delta

O algoritmo de retropropagação é um processo de aprendizado supervisionado, por correção de erro, no qual a alteração dos pesos pode ser feita de algumas formas distintas para minimizar a função de erro. A maneira mais simples de fazer isto é conhecida como regra delta: “O ajuste feito em um peso sináptico de um neurônio é proporcional ao produto do sinal de erro pelo sinal de entrada da sinapse em questão.” (HAYKIN, 2001, p. 78).

Para exemplificar a regra delta, tome-se, por exemplo, um neurônio k como o da Figura 4.3 situado na camada de saída de uma rede alimentada adiante. Supondo que $w_{kj}(n)$ represente o valor do peso sináptico do neurônio excitado por um elemento x_j do vetor de sinais de entrada no passo de tempo n . Pela regra delta, neste mesmo passo de tempo, o ajuste $\Delta w_{kj}(n)$ aplicado ao peso w_{kj} será dado por $\Delta w_{kj}(n) = \eta e_k(n) x_j(n)$ onde η é a taxa de aprendizado que determina a alteração para o novo peso, como segue:

$$w_{kj}(n+1) = w_{kj}(n) + \Delta w_{kj}(n)$$

A taxa de aprendizado é uma constante positiva que desempenha na prática um papel

importante para o desempenho da aprendizagem por correção de erro e influencia bastante a convergência do processo iterativo de treinamento (HAYKIN, 2001, p. 78).

4.1.5 Redes neurais e previsão de séries temporais

A utilização de RNAs em previsão de séries temporais pode ser entendida como uma resposta ao seguinte problema: dados os valores conhecidos de uma série temporal, obtenha um preditor capaz de estimar valores futuros da própria série a partir do comportamento da série temporal até o presente momento.

Para atuar como este preditor a rede precisa extrair durante o seu treinamento as autocorrelações presentes na série, conduzindo um mapeamento das entradas em cada instante com as respectivas saídas em instantes futuros. Esta ideia é “suficientemente genérica para permitir aplicar o procedimento na obtenção de preditores para séries temporais estacionárias, ou seja, em que a influência do passado no futuro da série temporal não varia com o tempo.” (VON ZUBEN, 2003, p. 72).

Com frequência, porém, o ambiente de interesse é não estacionário e os parâmetros estatísticos da série variam com o tempo. Com isto, os métodos tradicionais de aprendizagem supervisionada podem se mostrar inadequados, pois a rede estará tentando aprender com experiências passadas sem dispor de meios para acompanhar as variações estatísticas do ambiente onde opera. Nestas situações é desejável que a rede aprenda ao mesmo tempo em que processa o sinal em um sistema adaptativo, em uma forma de aprendizagem chamada de aprendizagem contínua, ou em tempo real (*on-the-fly*) em que o processo de aprendizagem nunca para (HAYKIN, 2001, p. 108).

4.1.6 Aspectos gerais

Em informática, o dicionário Michaelis (2010) registra a palavra topologia como o modo pelo qual os vários elementos de uma rede são interconectados. Ferneda (2006, p. 26), resume a forma mais usual de classificação de RNAs como segue:

As redes neurais artificiais se diferenciam pela sua arquitetura e pela forma como os pesos associados às conexões são ajustados durante o processo de aprendizado. A arquitetura de uma

rede neural restringe o tipo de problema no qual a rede poderá ser utilizada, e é definida pelo número de camadas (camada única ou múltiplas camadas), pelo número de nós em cada camada, pelo tipo de conexão entre os nós (feedforward ou feedback) e por sua topologia.

Um modelo de rede alimentada adiante de camada única passou a história conhecido como *perceptron de Rosenblatt*, ou simplesmente *perceptron*. O *perceptron* foi a base para a mais estudada arquitetura das redes alimentadas adiante, as redes do tipo *multilayer perceptron* (MLP) que são de longe as mais utilizadas em modelos de previsão. MLPs são constituídas de uma camada de entrada, seguidas de uma ou mais camadas ocultas, e uma camada de saída, todas sem realimentação. Durante o treinamento de um MLP, os neurônios das camadas ocultas têm a missão de mapear padrões intrínsecos que moldam as relações entre as entradas e as saídas da rede.

Nas aplicações de MLPs em previsão de séries temporais, as entradas da rede podem ser os valores conhecidos da própria série e/ou outras variáveis independentes e exógenas. Daí a versatilidade das RNAs como método de previsão: além de serem capazes de aproximar complexas relações funcionais entre os dados com bastante exatidão, ainda encontram correspondência tanto com modelos explanatórios, quanto com modelos puramente de séries temporais, ou mesmo uma combinação de ambos (ZHANG, 2004, p. 2-3).

4.2 A representação do tempo: memórias

Há muitos aspectos do comportamento humano que se desenvolvem ao longo do tempo e nele se baseiam. Seria tolice tentar entender estes comportamentos sem levar em conta a sua natureza temporal, e portanto, a representação do tempo em modelos conexionistas é fundamental. Uma maneira de fazer isto é implícita através dos efeitos do tempo no processamento da rede, como ocorre no caso de redes recorrentes. Outra forma é explícita através de uma representação espacial do tempo na arquitetura da rede, tal como se dá em redes com entradas defasadas no tempo. Ambas maneiras conferem capacidade de processamento temporal às redes neurais (ELMAN, 1990, p. 179, 206).

Adicionalmente, Samarasinghe (2007, p. 438) destaca a importância de extrair o mecanismo das autocorrelações em uma série temporal para obter previsões de boa qualidade. Por isto, muitas configurações de RNAs têm sido propostas com a finalidade de capturar aspectos

temporais nas entradas da rede. Na mais simples destas configurações, basta dotar a rede de memória por meio de atrasos (*lags* de tempo), que podem ser implementados a nível sináptico dentro da rede ou na camada de entrada. Outra configuração é através de redes recorrentes dirigidas dinamicamente.

A adição de memória transforma uma rede estática em dinâmica, conforme segue:

Redes estáticas

Em uma RNA estática, as saídas são funções (geralmente não lineares) de suas entradas, e fixado um vetor de entrada, a rede estática gera sempre o mesmo vetor de saída para aquele determinado vetor de entrada.

Redes estáticas são adequadas para o processamento de padrões espaciais, sem dependência temporal: a informação relevante é distribuída na rede, em coordenadas espaciais associadas com as variáveis que compõem a amostra de treinamento. Problemas tipicamente abordados com redes estáticas são: reconhecimento de padrões e aproximação de funções matemáticas. O MLP talvez seja o exemplo mais frequente de rede com aplicações estáticas.

Redes dinâmicas

São redes capazes de memorizar o efeito de entradas anteriores, o que dá a elas uma possibilidade de reter informação para ser usada mais tarde. Assim, uma rede dinâmica pode muito bem gerar saídas distintas para um mesmo vetor de entradas, porque sua resposta dependerá também do estado atual da memória desta rede.

Por sua característica inerente de memorizar informação passada, em curtos e longos períodos, as redes dinâmicas são boas candidatas a resolver problemas que envolvam processamento temporal ou espaço-temporal. É o caso da previsão de séries temporais.

As redes dinâmicas incluem, como exemplos, as redes recorrentes, onde o tempo é representado implicitamente (SANTOS; VON ZUBEN, 2001, p. 2).

A presença de realimentação causa profundo impacto no desempenho de uma rede e na sua capacidade de aprendizagem. Indo mais além, o processamento de atrasos unitários

(introduzidos pela realimentação) conferem um comportamento dinâmico à rede que a torna particularmente interessante, sobretudo se possuir funções de ativação não lineares (HAYKIN, 2001, p. 48-49).

A seguir, vamos ilustrar como uma memória pode se instalar dinamicamente em um neurônio recorrente que, por simplicidade, é dotado de função de ativação linear. O peso na entrada do neurônio é dado por a , enquanto ar é o peso da conexão recorrente.

Figura 4.6 – Neurônio artificial linear recorrente
 FONTE: Adaptada de SAMARASINGHE, 2007, p. 486.

Em um instante t o neurônio desta figura receba simultaneamente a atual entrada externa x_t junto com y_t que nada mais é do que a saída da rede no passo anterior. A saída y_{t+1} do neurônio no passo seguinte $t+1$, é expressa como segue:

$$y_{t+1} = ar \cdot y_t + a \cdot x_t$$

e avançando um instante a mais, teremos:

$$y_{t+2} = ar \cdot y_{t+1} + a \cdot x_{t+1}$$

ou seu equivalente, por substituição:

$$y_{t+2} = ar \cdot (\overbrace{ar \cdot y_t + a \cdot x_t}^{y_{t+1}}) + a \cdot x_{t+1}$$

$$y_{t+2} = ar^2 \cdot y_t + ar \cdot a \cdot x_t + a \cdot x_{t+1}$$

E assim da mesma forma:

$$y_{t+3} = ar.y_{t+2} + a.x_{t+2}$$

$$y_{t+3} = ar^3.y_t + ar^2.a.x_t + ar.a.x_{t+1} + a.x_{t+2}$$

Pode-se demonstrar que para h passos à frente, a saída será:

$$y_{t+h} = ar^h.y_t + ar^{h-1}.a.x_t + ar^{h-2}.a.x_{t+1} + \dots + a.x_{t+h-1}$$

Com este exemplo, dá para perceber que as entradas externas consecutivas, x_t e x_{t+1} , foram definitivamente incorporadas em todas as saídas do neurônio, a partir de $t+2$.

A expressão acima mostra que este neurônio recorrente é capaz de construir recursivamente uma memória ao longo do tempo, utilizando apenas a entrada externa x_t , disponível em cada instante t . Durante o treinamento, tanto o peso a de entrada quanto o peso recorrente ar são adaptados. O papel do peso recorrente é ponderar a influência das entradas mais recentes com a das mais antigas. Se $ar < 1$, a expressão acima nos permite ver que as entradas mais recentes terão um impacto maior na saída.

No caso de uma série temporal, a variável externa x_t nada mais é do que o valor atual da série, e a saída da rede y_{t+1} é o valor da série no próximo instante. A memória da série neste exemplo foi, portanto, encapsulada através do laço recorrente na forma de um modelo autorregressivo, cujos *lags* são determinados pelo próprio sistema.

Se nenhuma variável exógena é utilizada, então a saída deste neurônio depende apenas da sua própria memória (implícita) de valores anteriores da série e do valor atual da série que se apresenta como única entrada externa do neurônio. Nada impede, porém, a extensão deste exemplo linear para neurônios com função de ativação não linear, bem como o uso adicional de variáveis exógenas nas entradas, além da série temporal.

Para construir uma rede neural artificial recorrente, um ou mais neurônios recorrentes, como o visto acima, devem ser dispostos como neurônios ocultos em uma estrutura de rede

feedforward dotada de laços de realimentação saindo destes neurônios ocultos (SAMARASINGHE, 2007, p. 487).

Propriedades de memória: profundidade e resolução

A memória em uma rede neural pode ser categorizada por duas propriedades: profundidade e resolução. A profundidade revela quão longe a memória pode retroceder no armazenamento de informação passada: uma memória de longo prazo pode reter informação de muitos passos anteriores no tempo. A resolução da memória mede o quanto ela é capaz de preservar as características dos elementos de entrada. Uma memória de alta resolução pode reconstituir facilmente as propriedades individuais de cada elemento apresentado a ela (BRAX, 2000, p. 20).

4.2.1 Redes com atraso alimentadas adiante (TLFN)

Este é um modelo de rede fácil de entender conceitualmente, porque sua estrutura é similar a de um perceptron de uma ou mais camadas (MLP) conforme figura abaixo. A diferença reside nas entradas da rede TLFN (*time-lagged feedforward neural network*) que são os atrasos (*lags*) da própria variável a ser prevista: as entradas precisam ser submetidas sequencialmente à rede, ao contrário da apresentação aleatória que é usualmente adotada com redes alimentadas à frente.

Figura 4.7 - Rede *Focused Time-Lagged Feedforward* (FTLFN)

As entradas apresentadas à rede TLFN no instante de tempo t são apresentadas novamente, porém com uma unidade de atraso, no próximo instante $t+1$. O procedimento é repetido até

atingir o número p de atrasos desejados. Desta maneira introduz-se na rede uma memória que guarda no curto prazo a estrutura de autocorrelação dos dados externos (SAMARASINGHE, 2007, p. 446-448).

Figura 4.8 – Filtro de memória de curto prazo
 FONTE: Adaptada de SAMARASINGHE, 2007, p. 449.

Esta rede também é caracterizada como uma rede focada (*focused TLFN*), porque a estrutura inteira da memória é externamente localizada nos pesos de entrada dos neurônios da primeira camada. Cada um destes neurônios é considerado um filtro focado, pois embute a memória em seus próprios pesos (DE OLIVEIRA, 2007, p. 37).

Na figura acima, toma-se um neurônio qualquer da primeira camada da rede TLFN, a saída z_t em resposta à entrada y_t e seus valores passados $y_{t-1}, y_{t-2}, \dots, y_{t-p}$ é dada como segue, onde φ é a função de transferência:

$$z_t = \varphi\left(\sum_{i=0}^p a_i \cdot y_{t-i} + b\right)$$

A estrutura da TLFN exemplificada capta um trecho da série, de tamanho p , a cada instante. É como se uma janela móvel fosse deslizada ao longo da série para os dados serem apresentados em sequência à rede. Este conceito de janela também pode ser usado para extrair vários fragmentos, de mesmo tamanho (p), para gerar exemplos de treinamento não contíguos. O chamado janelamento da série ilustrado na Figura 4.9 é procedimento frequente em aplicações

de previsão de séries com RNAs alimentadas adiante, e está associado à redes que recorrem à representação explícita, basicamente uma metáfora espacial para o tempo da qual podem decorrer alguns inconvenientes não detalhados aqui (ELMAN, 1990, p. 181).

Figura 4.9 – Janelamento de uma série temporal
 FONTE: MUELLER, 1996.

4.2.2 Redes recorrentes dirigidas dinamicamente (*dynamically-driven*)

A noção de estado tem um papel crucial na concepção de RNAs baseadas em sistemas dinâmicos. O estado de um sistema dinâmico é definido pelo conjunto de quantidades que condensam toda a informação a respeito do seu comportamento passado necessária para descrever de maneira única o comportamento futuro do sistema, excluindo-se apenas os efeitos causados pelas entradas externas ao sistema (HAYKIN, 2001, p. 794).

As redes recorrentes podem ser categorizadas em duas classes: aquelas que recebem um sinal constante, ou um pulso de entrada, e são usadas para entrar em determinado estado, e outras que recebem entradas com valor variável ao longo do tempo e são usadas para gerar saídas interdependentes em momentos distintos. Estas últimas são chamadas redes recorrentes dinâmicas (KREMER, 2001, p. 1).

O conceito de redes recorrentes dirigidas dinamicamente (RRDD) é inteiramente distinto de uma rede TLFN, cuja memória é mantida externa à rede. As RRDDs são projetadas para absorver progressivamente a estrutura de autocorrelação da série, que é memorizada internamente na rede, por meio de conexões de realimentação. Apenas os valores externos atuais são usados pela RRDD para construir uma memória dinâmica em que não há indicação do número de *lags* necessários para captar a autocorrelação.

Estas redes estão apoiadas na ideia de capturar a dinâmica da série ao longo do tempo, a partir do estado atual da rede, realimentado recursivamente. A cada passo, junto com a entrada externa atual, é submetido internamente à rede um resquício de memória de todos os valores submetidos a ela anteriormente (SAMARASINGHE, 2007, p. 447, 485).

A arquitetura de redes recorrentes é muito diversificada. A figura a seguir ilustra três tipos de RRDDs que compartilham das seguintes características:

- Incorporam um MLP estático, ou parte dele;
- Exploram a capacidade de mapeamento não linear do MLP;
- Só há entradas do momento presente;
- A rede encontra a estrutura de autocorrelação por conta própria, dando importância maior às entradas mais recentes;
- A memória é interna e acumulada recursivamente a longo prazo;
- Não há autorrealimentação na rede.

Figura 4.10 – Redes recorrentes para previsão de séries temporais
 FONTE: SAMARASINGHE, 2007, p. 447.

A Figura 4.10a apresenta o modelo de entradas e saídas (E/S), no qual a saída da rede é realimentada com uma unidade de atraso na camada de entrada. Esta rede equivale a um modelo autorregressivo não linear (NAR) quando as entradas são apenas as saídas realimentadas ao longo do tempo. Se o modelo incluir como entrada outras variáveis, além

das geradas na saída, passa a ser autorregressivo com entradas exógenas (NARx).

No modelo de rede estilizado pela Figura 4.10b há uma computação do erro na saída da rede, e realimentação deste erro como entrada no próximo passo em conjunto com as saídas da rede do passo anterior. Isto equivale a adicionar médias móveis ao modelo da Figura 4.10a, ou seja, o NARx passa a ser um NARIMAx.

A Figura 4.10c mostra um percéptron recorrente de múltiplas camadas (RMLP) no qual as saídas, tanto da camada final quanto das camadas intermediárias, são realimentadas com uma unidade de atraso na camada de entrada e nas camadas intermediárias. Nestas condições a rede pode encapsular uma memória dinâmica de longo prazo. Muitas variantes deste modelo têm sido usadas em aplicações práticas. As redes de Elman são um caso particular deste tipo de rede recorrente (SAMARASINGHE, 2007, p. 447-448).

Note-se que nas redes acima não há defasagem de tempo (*lags*) nas entradas, portanto não se aplica o procedimento de janela deslizando no treinamento de redes recorrentes.

4.3 Redes de Elman

Em 1990, Jeffrey Elman, psicolinguista da Universidade da Califórnia, treinou uma rede recorrente para processar palavras dispostas em sequências de sentenças simples. Essa disposição de palavras refletia informações gramaticais descritas em termos sintáticos (como sujeito, verbo e objeto) e semânticos (como agente, paciente e tema) (GARCIA, 2006, p. iv).

A rede recorrente simples (SRN – *simple recurrent network*) descrita em Elman (1990), representada na Figura 4.11 e exemplificada na Figura 4.12, agrega uma camada de unidades de contexto a um percéptron de uma camada oculta. As unidades de contexto armazenam as saídas dos neurônios da camada oculta por um passo de tempo (atraso unitário) e seu conteúdo é realimentado na camada de entrada (HAYKIN, 2001, p. 790).

Figura 4.11 – Rede de Elman: representação genérica

FONTE: Adaptado de ELMAN, 1990, p. 182.

Figura 4.12 – Exemplo de uma rede de Elman

FONTE: GARCIA, 2006, p. 27.

Nas redes de Elman o estado interno da rede (contexto) em um determinado instante depende de duas coisas: da entrada da rede no momento considerado e do próprio estado da rede no instante anterior. O papel das unidades de contexto é fornecer memória dinâmica à rede, pois a camada de contexto tem a capacidade de codificar toda a informação de entrada desde o começo da sequência apresentada à rede (STOIANOV, 2001, p. 6, 12).

Em uma rede de Elman a realimentação (*positive feedback*) é usada para introduzir na rede uma memória de grande profundidade e pouca resolução: as unidades de contexto conservam um registro exponencialmente decrescente do valor anterior de saída dos neurônios da camada oculta da rede. Em relação a redes do tipo com atraso de tempo (*time-lagged*), as redes de Elman não impõem limites rígidos ao número de *lags* de memória e a informação apresentada

mais recentemente vai perdendo resolução à medida que vai ficando mais distante no passado (KOSKELA *et al*, 1996, p. 493).

As redes de Jordan são semelhantes às redes de Elman, porém diferem quanto ao posicionamento da realimentação. Enquanto nas redes de Elman a realimentação se dá a partir da camada intermediária, nas redes de Jordan ela provém da última camada.

Redes recorrentes podem ser entendidas como uma sobreposição de várias redes alimentadas à frente, o que equivale a dizer que uma rede de Elman pode ser desdobrada em várias camadas de redes alimentadas à frente, conforme veremos a seguir (SAMARASINGHE, 2007, p. 489).

4.4 Algoritmos para treinamento de redes recorrentes

Existe uma grande quantidade de algoritmos disponíveis para os ajustes dos pesos de redes recorrentes e alguns dos mais utilizados são o da aprendizagem em tempo real (RTRL – *real time recurrent learning*), a retropropagação através do tempo (BPTT – *backpropagation through time*) e o filtro de Kalman (DE OLIVEIRA, 2003, p. 73).

Os três algoritmos citados valem-se do cálculo do gradiente da função de erro para atualizar os pesos, distinguindo-se pelo método específico com que o fazem. O BPTT e RTRL usam o método do gradiente descendente, que será visto mais adiante, enquanto o terceiro usa o cálculo do gradiente na linearização do sistema a fim de estender a aplicação do filtro de Kalman (HAGNER *et al*, 2001, p. 10). O gradiente de uma função é um vetor que aponta no sentido e direção da taxa máxima de variação da função (BRAGA *et al*, 2000, p. 65).

4.4.1 Retropropagação no tempo (BPTT – *backpropagation through time*)

O algoritmo de retropropagação no tempo (BPTT) para treinamento de redes recorrentes, incluindo as redes de Elman, é uma extensão do algoritmo básico de retropropagação visto anteriormente. Ele opera com a premissa de que a operação temporal de uma rede recorrente pode ser desdobrada em uma rede alimentada à frente, cuja arquitetura é acrescida de uma

camada a cada passo de tempo.

Considerando N uma rede recorrente que deve ser treinada em uma tarefa temporal iniciando no tempo n_0 e se estendendo até o tempo n , e que N^* represente a rede *feedforward* que resulta do desdobramento de N , temos as seguintes implicações:

- A cada passo de tempo dentro do intervalo $(n_0, n]$, a rede N^* tem uma camada contendo K neurônios, onde K é o número de neurônios integrantes da rede N ;
- Existe uma cópia de cada neurônio da rede N em cada camada da rede N^* ;
- Para cada passo de tempo $l \in [n_0, n]$, a conexão do neurônio i na camada l para o neurônio j na camada $l+1$ da rede N^* é uma cópia da conexão sináptica do neurônio i para o neurônio j na rede N (HAYKIN, 2001, p. 808).

Para ilustrar este conceito de desdobramento em camadas ao longo do tempo, vejamos a Figura 4.13 na qual ilustramos uma rede de Elman que, por simplicidade, tem apenas um neurônio recorrente na camada intermediária e um neurônio na camada de saída. No exemplo, ao longo de t , $t+1$ e $t+2$, esta rede recebe como entrada os valores de uma série temporal y_t, y_{t+1} e y_{t+2} , e gera como saída os valores $\hat{y}_{t+1}, \hat{y}_{t+2}$ e $\hat{y}_{t+3}$.

Figura 4.13 – Desdobramento de rede recorrente em camadas alimentadas à frente ao longo de 3 períodos de tempo

FONTE: Adaptado de SAMARASINGHE, 2007, p. 489.

O algoritmo BPTT pode ser implementado de duas formas: por época e truncado.

BPTT por época (*epochwise*)

Considerando que n_0 represente o início de uma época e n_1 o seu tempo final, temos que a função de erro é dada por:

$$E_{total}(n_0, n_1) = \frac{1}{2} \sum_{n=n_0}^{n_1} \sum_{j \in A} e_j^2(n)$$

Em que A é o conjunto de índices j relativos aos neurônios da rede para os quais as respostas corretas são especificadas, e $e_j(n)$ corresponde ao erro na saída de um destes neurônios confrontada com a resposta correta desejada. O objetivo é avaliar a sensibilidade parcial da função de erro, o que significa computar o gradiente de $E_{total}(n_0, n_1)$, ou seja, as derivadas parciais desta função em relação aos seus pesos sinápticos. O BPTT por época segue os seguintes passos:

- Executar um passo único à frente através da rede para o intervalo $(n_0, n_1]$. Salvar o registro completo dos dados de entrada, as respostas desejadas e os pesos sinápticos (estado da rede);
- Executar um único passo atrás sobre o registro salvo, para calcular os gradientes locais $\delta_j(n)$ como segue, onde $v_j(n)$ é o campo local induzido do neurônio j :

$$\delta_j(n) = -\frac{\partial E_{total}(n_0, n_1)}{\partial v_j(n)}$$

Para todo $j \in A$ e $n_0 < n \leq n_1$ tem-se a computação realizada através da expressão:

$$\delta_j(n) = \begin{cases} \phi'(v_j(n))e_j(n) & \text{para } n = n_1 \\ \phi'(v_j(n))[e_j(n) + \sum_{K \in A} w_{jk} \delta_k(n+1)] & \text{para } n_0 < n < n_1 \end{cases}$$

Nesta expressão assume-se que todos os neurônios tenham a mesma função de ativação ϕ . A derivada da função em relação ao seu argumento é $\phi'(\cdot)$. O uso da equação acima, em que $n_0 < n < n_1$, é repetido começando no tempo n_1 e voltando passo a passo até o tempo n_0 .

O número de passos de tempo contidos na época determinará o número de passos envolvidos até aqui.

Tendo voltado ao tempo n_0+1 , após executar a retropropagação, é aplicado o seguinte ajuste aos pesos sinápticos:

$$\begin{aligned}\Delta w_{ji} &= -\eta \frac{\partial E_{total}(n_o, n_i)}{\partial w_{ji}} \\ &= \eta \sum_{n=n_o+1}^{n_i} \delta_j(n) x_i(n-1)\end{aligned}$$

Onde, η é a taxa de aprendizagem e $x_i(n-1)$ é a entrada na i -ésima sinapse do neurônio j no tempo $n-1$.

O BPTT por época é similar ao modo de apresentação dos dados em lotes (*batch*) do algoritmo padrão de retropropagação, porém no BPTT por época as respostas desejadas são especificadas aos neurônios em várias camadas da rede porque a camada de saída real é replicada várias vezes ao desdobrarmos a operação temporal da rede recorrente em várias camadas alimentadas à frente (HAYKIN, 2001, p. 810).

BPTT(h) truncado (*online*):

No algoritmo de retropropagação através do tempo truncado (BPPT(h)) usa-se o valor instantâneo da soma dos erros quadrados, para definir a função de erro dada por:

$$E(n) = \frac{1}{2} \sum_{j \in A} e_j^2(n)$$

O gradiente negativo da função de erro $E(n)$ é aplicado para calcular o ajuste dos pesos sinápticos da rede em cada instante n , em um procedimento similar ao modo incremental de ajuste dos pesos no algoritmo básico de retropropagação.

Os ajustes são feitos continuamente à medida que a rede conduz o processamento e, para viabilizar isto computacionalmente, é necessário limitar um número fixo de passos de tempo,

chamado profundidade de truncamento (h), salvando apenas a história relevante dos dados de entrada. Se a computação não é truncada, ela retroage ao tempo inicial e o esforço computacional cresce linearmente conforme a rede continua o processamento, podendo ficar impraticável. Desta forma, tem-se o gradiente local do neurônio j dado agora por:

$$\delta_j(l) = -\frac{\partial E(l)}{\partial v_j(l)} \text{ para todo } j \in A \text{ e } n-h < l \leq n$$

Que resulta na seguinte expressão:

$$\delta_j(l) = \begin{cases} \phi'(v_j(l))e_j(l) & \text{para } l = n \\ \phi'(v_j(l))[e_j(l) + \sum_{K \in A} w_{jk} \delta_k(l+1)] & \text{para } n-h < l < n \end{cases}$$

Após retornar ao tempo $n-h+1$, através da retropropagação, o seguinte ajuste sináptico é aplicado ao peso w_{ji} do neurônio j :

$$\Delta w_{ji}(n) = -\eta \sum_{l=n-h+1}^n \delta_j(l)x_i(l-1)$$

Ao contrário do BPTT por época, no BPTT(h) o sinal de erro é injetado na computação somente no tempo corrente n e por isto não há necessidade de manter um registro dos valores passados das respostas desejadas (HAYKIN, 2001, p. 811).

Gradiente descendente

Os dois modos do BPTT baseiam-se no algoritmo de retropropagação padrão com atualização dos pesos no sentido oposto ao gradiente da função de erro, em uma generalização da regra delta. Desta forma, os pesos da rede mudam no sentido da descida mais acentuada da superfície multidimensional da função de erro, o que leva a redução gradual do erro (BRAGA *et al*, 2000, p. 65).

Esta é a heurística do gradiente descendente, cuja ideia pode ser entendida em analogia a uma pessoa que esteja em um terreno montanhoso e que tenha o objetivo de mover-se para o ponto mais baixo do terreno. O terreno montanhoso simboliza a função de erro, com seus picos e vales, e no contexto de otimização o que se busca é o ponto de menor elevação, movendo-se

na direção de maior inclinação de descida, isto é, do gradiente descendente (SUNDARESHAN, 2001).

Um dos problemas desta abordagem, contudo, é a possibilidade do algoritmo ficar preso a um vale que não seja o mínimo absoluto da função, e que corresponde ao chamado mínimo local conforme ilustrado pela Figura 4.14. Superfícies de erros convexas, com um único mínimo, permitem que este método atinja o mínimo global. Porém, em superfícies de erros não convexas e altamente convolutas, habituais em problemas práticos, a solução alcançada pode não ser a ótima (MUELLER, 1996).

Figura 4.14 – Panorama de uma função de erro com mínimos: local e global
FONTE: KOVÁCS, 1996, p. 85.

5 METODOLOGIA DA PESQUISA

5.1 Aspectos teóricos

Esta pesquisa foi conduzida em duas grandes etapas: uma de fundamentação teórica e outra empírica. A fundamentação teórica compreendeu inicialmente uma revisão de literatura fortemente direcionada aos métodos quantitativos de análise e previsão de séries temporais com a metodologia de Box-Jenkins e com redes neurais artificiais.

Há várias maneiras de classificar uma pesquisa científica. Com relação aos dados e métodos empregados, este é um trabalho quantitativo, começando pelo tipo de dados analisados e pelas questões formuladas a respeito deles. As respostas para tais perguntas vieram através de mensuração objetiva da qualidade de previsões geradas por estruturas computacionais (redes neurais artificiais) e por equações lineares, em uma abordagem baseada na matemática e estatística, com modelos de séries temporais, isto é, modelos nos quais uma variável é prevista apenas com base nela mesmo.

Por se tratar de pesquisa de natureza quantitativa, os dados foram quantificados por meio de variáveis operacionais e os resultados avaliados por métricas de erros de previsão adequadas para comparação. Buscou-se com isto a fundamentação teórica para levar adiante a parte empírica descrita no próximo tópico.

Com respeito à sua função, este trabalho pode ser enquadrado como uma pesquisa exploratória já que um tipo específico de rede neural (Rede de Elman) foi aplicado na previsão de séries temporais do varejo, de bens e serviços, caracterizando uma combinação pouco usual, cujos resultados podem vir a contribuir para outros trabalhos. De acordo com Zikmund (1997, p. 51) e Pinsky (2009, p. 35) uma pesquisa é considerada exploratória quando trabalha para encontrar a dimensão de um problema, coletando informação para que futuras pesquisas cheguem a evidências mais conclusivas.

5.2 Desenvolvimento da pesquisa empírica

Os dados utilizados neste trabalho, preexistentes em relatórios disponíveis na *internet*, podem ser considerados de natureza secundária já que foram originalmente coletados por outras pessoas com distintas finalidades e propósitos (AAKER *et al*, 1995, p. 77; KITAHARA, 2007, p. 33).

Desta forma, foram constituídas três séries temporais discretas univariadas e unidimensionais, ou seja, três conjuntos distintos de observações ordenadas ao longo do tempo, de uma só variável discreta, como segue:

AEREO: de janeiro de 2003 até dezembro de 2008, com 72 pontos mensais.

ADUBOS: de janeiro de 1991 até dezembro de 2009, com 228 pontos mensais.

FARMA: de janeiro de 2003 até dezembro de 2009, com 84 pontos mensais.

De posse dos dados, partiu-se para a previsão de cada uma destas séries com redes neurais artificiais. Como parte do trabalho foi direcionada a comparar resultados entre redes neurais e modelos lineares, logo constatou-se a necessidade destas duas abordagens andarem em conjunto a fim de gerar uma comparação mais equilibrada dos dois métodos de previsão. As transformações que foram aplicadas nos dados como pré-processamento (logaritmo e diferenças) para previsão com RNAs foram as mesmas usadas para tornar estacionárias as séries originais na metodologia de Box-Jenkins.

Em todos os casos fizemos previsões dinâmicas e estáticas para os últimos 12 meses de cada série, simulando a condição de que os valores reais desses últimos meses eram desconhecidos *a priori*. Para o treinamento das redes e geração das respectivas previsões foram desenvolvidos códigos com comandos e funções do *software* MATLAB 2008b, porém sem a preocupação de otimizar a eficiência de programação.

5.2.1 Previsão com modelos de séries temporais

Com recursos do *software* Eviews 6.0, cada uma das séries foi trabalhada como segue:

- a) Transformação da série original até atingir indicação visual de estacionariedade no correlograma da FAC e FACP da série transformada por logaritmo e diferenças;
- b) Comprovação da estacionariedade: testes de raízes unitárias de Dickey-Pantula;
- c) Identificação de modelos SARIMA candidatos a representar a série;
- d) Estimação de parâmetros significativos dos modelos candidatos;
- e) Diagnóstico de modelos que tiveram parâmetros significativos: teste de Jarque-Bera (JB), teste Q de Ljung-Box e teste Q dos resíduos quadrados. Se ainda restar mais de um modelo finalista, escolher aquele que minimize os critérios AIC / SIC;
- f) Previsão estática e dinâmica para 12 meses, com o modelo finalista;
- g) Análise de indicadores de desempenho: RMSE, TIC e proporção de covariância.

5.2.2 Previsão com redes de Elman

A previsão baseou-se na ideia de apresentar, um por um, os valores y_t de cada série, à redes de Elman descritas a seguir, para obter a cada instante a previsão $\hat{y}_{t+1}$ do valor seguinte, no procedimento estático. Para o procedimento dinâmico valeu o mesmo raciocínio, porém com realimentação de previsões, conforme explicado no Capítulo 2.

5.2.2.1 Arquitetura da rede

A arquitetura geral das redes recorrentes de Elman, com uma entrada e uma saída, utilizadas neste trabalho é descrita na Figura 5.1 a seguir:

Figura 5.1 – Rede de Elman: modelo geral utilizado, com uma camada intermediária

Cada rede foi implementada através do *software* MATLAB 2008b, cuja função *newelm* cria automaticamente, e por *default*, uma rede com 1 camada intermediária recorrente com neurônios ativados pela função tangente hiperbólica, e 1 camada de saída linear. As conexões entre a saída da camada intermediária e as unidades de contexto foram fixadas com peso 1, para configurar o atraso unitário indicado por Δ^{-1} .

Várias redes semelhantes foram criadas por meio do parâmetro (*SI*) que determina o tamanho da camada intermediária na função *newelm*. Neste caso, *SI* assumiu valores entre 1 e 20 neurônios ocultos. A entrada da rede foi sempre de 1 nó e a camada de saída foi fixada em um único neurônio para todas as redes criadas. As redes podem então ser identificadas pelo número *SI* de neurônios em sua camada intermediária e foram treinadas como descrito mais adiante.

5.2.2.2 Treinamento das redes

Cada uma das séries temporais AEREO, ADUBOS e FARMA foi sistematicamente submetida como entrada de redes distintas, a fim de capacitar estas redes na tarefa de fazer

previsões. Os últimos 12 meses de cada série não foram incluídos no treinamento, ficando reservados para avaliação de suas previsões via redes neurais. Excluído o horizonte de previsão $h=12$, o restante dos dados foi utilizado no treinamento da rede subdivididos em 75% para aprendizado e 25% para validação.

Os dados de aprendizado foram utilizados para efetivamente ajustar os pesos da rede, segundo o algoritmo de retropropagação no tempo, por época, na direção do gradiente descendente da função de erro, conforme descrito no Capítulo 4.

Inicialmente a rede foi criada sem estar treinada por meio do comando do MATLAB: `net = newelm(Pa,Ta,S1)`, onde:

`net` = atributo de identificação da rede de Elman com S1 neurônios na camada oculta;

`Pa` = sequência de entrada formada pelos valores da série no intervalo de aprendizado;

`Ta` = sequência desejada (*targets*) na saída da rede no intervalo de aprendizado.

Assim, se $Pa = \{y_1, y_2, \dots, y_t, \dots, y_{ta}\}$, então $Ta = \{y_2, y_3, \dots, y_{t+1}, \dots, y_{ta+1}\}$. O intervalo de aprendizado `Pa` configura uma época que se inicia no primeiro valor da série ($n_0=1$) e finaliza no tempo $n_1 = ta + 1$. Do mesmo jeito que `Pa` e `Ta` correspondem à sequência de entrada e de valores desejados na saída da rede no intervalo de aprendizado, temos de forma equivalente `Pv` e `Tv` no intervalo de validação.

A Figura 5.2 ilustra como foi feita esta programação no MATLAB: os valores foram indexados em relação ao último ponto (*end*) da série de tal forma que $ta = end - h - v - 1$, onde v é igual ao número de pontos de validação.

Figura 5.2 – Sequência de entradas e de saídas desejadas na rede em cada intervalo de dados

De acordo com Samarasinghe (2007, p. 446, 456), o uso do conjunto completo dos dados nesta forma contribui para um melhor resultado da previsão com redes neurais.

Considerando as três séries deste trabalho e seus respectivos tamanhos, as redes foram treinadas com os dados organizados conforme as tabelas 5.1 e 5.2 abaixo:

Tabela 5.1 – Número de pontos característicos de cada intervalo

Série	Aprendizado	Validação (v)	Previsão (h)	Série completa
AEREO	45	15	12	72
ADUBOS	162	54	12	228
FARMA	54	18	12	84

Tabela 5.2 – Sequências de cada intervalo

Série	Pa	Ta	Pv	Tv
AEREO	1,2,...,44	2,3,...,45	1,2,...,59	2,3,...,60
ADUBOS	1,2,...,161	2,3,...,162	1,2,...,215	2,3,...,216
FARMA	1,2,...,53	2,3,...,54	1,2,...,71	2,3,...,72

Os pesos a ajustar em cada rede durante o treinamento são as ligações sinápticas entre:

- O nó de entrada e os neurônios da camada intermediária;

- b) Neurônios da camada intermediária e o neurônio da camada de saída;
- c) Unidades de contexto e neurônios da camada intermediária.

Adicionalmente, há um peso associado ao limiar (*bias*) de cada neurônio que por simplicidade está omitido na figura da rede.

5.2.2.2.1 Validação cruzada

Os dados de validação foram utilizados no procedimento de validação cruzada (*cross-validation*) a fim de evitar ajuste excessivo (*overfitting*) nos pesos da rede que sem isto poderia memorizar o conjunto de aprendizado e perder capacidade de generalização fora deste intervalo. A validação cruzada foi programada da seguinte maneira:

- a) Ao fim de cada época os pesos da rede são alterados pelo algoritmo escolhido, reduzindo-se gradativamente o MSE para a previsão no intervalo de aprendizado;
- b) O MSE também é calculado na previsão feita pela rede no intervalo de validação;
- c) O MSE de validação é comparado com o seu valor no final da época anterior;
- d) Se o MSE de validação diminuir (ou ficar igual) em relação ao seu valor na época anterior, o treinamento prossegue com ressubmissão do conjunto de aprendizado em uma nova época. Se o MSE crescer, é considerada uma falha de validação;
- e) Se houver falha de validação, ela é um indício de excesso de treinamento. Estas falhas foram admitidas até um limite máximo de 15 épocas consecutivas visando ultrapassar pontos de mínimo local na função do MSE de validação;
- f) O processo de treinamento foi interrompido quando atingido o máximo de 15 épocas consecutivas com MSE de validação crescente, ou após atingir-se o limite máximo total de 1000 épocas de treinamento.

O efeito de *overfitting* ocorre quando a partir de determinada época a rede invés de melhorar começa a piorar sua taxa de acerto para padrões diferentes daqueles utilizados para o ajuste dos pesos. Uma das alternativas mais utilizadas para evitar este efeito é encerrar o treinamento mais cedo (*early-stopping*), isto é, quando o erro de validação começa a subir (BRAGA *et al*, 2000, p. 68).

Apesar do método de validação cruzada reduzir o conjunto de dados disponíveis para aprendizado, o benefício de encontrar o ponto ótimo de parada tipicamente supera esta desvantagem. Este é o método recomendado como critério de parada em aplicações do mundo real (PRINCIPE *et al*, 2000, p. 197).

5.2.2.2.2 Dados originais e pré-processamento

As previsões com redes neurais foram obtidas com redes treinadas de duas maneiras: sem pré-processamento dos dados e com pré-processamento permitindo observar o efeito do tratamento prévio da tendência e da sazonalidade na qualidade da previsão.

No caso de dados originais, eles sofreram apenas uma transformação inerente ao método de redes neurais: os valores da série passaram por uma padronização no intervalo $[-1,1]$ através da função *mapminmax* do MATLAB. A padronização dos dados faz com que os valores apresentados na entrada da rede não se afastem dos limites da função de ativação tangente sigmoide, e aproxima a escala dos dados de entrada com a dos pesos iniciais, o que contribui para a convergência do algoritmo (FARAWAY; CHATFIELD, 1998, p. 237; ZHANG *et al*, 1998; KOVÁCS, 2002, p. 83).

No caso de dados pré-processados, foi aplicada a transformação logarítmica e as mesmas diferenças regulares e sazonais aplicadas na metodologia de Box-Jenkins, e ao fim, os dados também passaram pela padronização inerente ao método de RNAs. Este tratamento prévio dos dados replica o procedimento de estudos como o de Kline (2004, p. 6) que também utilizou séries relativamente curtas para previsões de vários passos adiante, e o de Zhang e Kline (2007, p. 1.803) para séries sazonais.

5.2.2.2.3 Aspectos gerais

Foram treinadas várias redes com a variação do tamanho da camada intermediária entre 1 e 20 neurônios. Para cada rede foram feitas pelo menos 10 tentativas de treinamento, com diferentes inicializações atribuídas pelo MATLAB aos pesos de cada camada. Para isto o MATLAB usa o método de Nguyen e Widrow (1990) no qual há uma distribuição equilibrada dos valores iniciais dos pesos, considerando a faixa de valores da função de ativação, em cada

neurônio.

Estes valores iniciais contêm uma certa aleatoriedade, e mudam cada vez que a rede passa por uma nova tentativa de treinamento. É possível obter resultados de treinamento totalmente diferentes, mesmo com a repetição das entradas. As reinicializações são necessárias para aumentar as chances de se atingir o mínimo global da função de erro, pois não há garantia de atingi-lo, mesmo partindo várias vezes de diferentes combinações iniciais (FARAWAY; CHATFIELD, 1998, p. 237).

O treinamento foi levado até o limite máximo estipulado de 1.000 épocas quando não houve interrupção por falha de validação em 15 épocas consecutivas. A convergência dos pesos para um valor final de MSE se deu geralmente após 200 épocas.

A taxa de aprendizado mais utilizada foi de 10%, estipulada empiricamente por tentativa e erro no intervalo 2,5% a 30%. A taxa de aprendizado determina o quanto os pesos mudam a cada iteração. Taxas muito grandes tornam o algoritmo instável, sem convergência, e taxas muito pequenas acarretam treinamento demorado e ineficiente (BALLINI, 2000, p. 70).

Os fluxogramas nas Figuras 5.3 e 5.4, ilustram os procedimentos adotados no treinamento das redes e para a obtenção de previsões via redes de Elman a serem comparadas com aquelas obtidas com modelos lineares seguindo Box-Jenkins.

Figura 5.3 – Fluxograma de treinamento de uma rede com o procedimento de validação cruzada

Figura 5.4 – Fluxograma para obter a melhor rede e comparar suas previsões com Box-Jenkins

6 RESULTADOS EXPERIMENTAIS

6.1 Série mensal de tráfego aéreo – mercado doméstico

A série de tráfego aéreo de passageiros no mercado doméstico brasileiro é apresentada no Gráfico 6.1. No eixo vertical, a variável AEREO registra a quantidade mensal de passageiros x km voados, segundo dados disponíveis no *site* da ANAC (Agência Nacional de Aviação Civil). A série contém 72 pontos mensais: jan. 2003 – dez. 2008 dos quais tomamos os primeiros 60 para ajustar um modelo SARIMA aos dados, conforme Gráfico 6.1. Para treinar as redes neurais usamos os mesmos 60 pontos fracionados em uma amostra de aprendizado (45) e outra de validação (15), conforme ilustrado pela Gráfico 6.2.

Gráfico 6.1 – Partição da série AEREO: Box-Jenkins

Gráfico 6.2 – Partição da série AEREO: redes neurais

A inspeção visual do gráfico da variável AEREO revela uma tendência de crescimento, o que já denuncia a não estacionariedade da série. Nota-se também um aumento da variabilidade da série ao longo do tempo, o que justifica uma transformação logarítmica inicial nos dados. Por fim, há claro indício de sazonalidade com picos e vales que se sucedem em um padrão repetitivo ao longo do tempo.

A previsão foi sempre para os últimos 12 meses, isto é, o período de janeiro a dezembro de 2008. Os valores da série nos últimos 12 meses são mantidos apenas para comparação com a previsão em um processo simulado de desconhecimento no horizonte de previsão com SARIMA e redes neurais artificiais recorrentes de Elman.

6.1.1 Metodologia de Box-Jenkins: AEREO

6.1.1.1 Transformações e testes de raízes unitárias

A variável AEREO foi transformada na nova variável D1LOGAER através da transformação primeira diferença no logaritmo de AEREO, eliminando-se a tendência. A ordem de integração foi testada através do teste de Dickey-Pantula partindo da hipótese nula inicial de 3 raízes unitárias (RU). No primeiro passo do teste rejeitamos a hipótese nula de 3 raízes unitárias, no segundo rejeitamos a presença de 2 raízes unitárias. Nestes 2 passos iniciais

ocorreram coeficientes significativos apenas na equação sem intercepto e sem tendência.

Tabela 6.1 – Teste de Dickey-Pantula em Log (AEREO)

Dickey-Pantula	Intercepto	Tendência	Estatísticas do teste	Vc a 1%	Vc a 5%
<u>Passo 1</u> H_0 : 3 RU's H_1 : 2 RU's	Não significativo	Não significativa	-14.33	-2.60	-1.95
<u>Passo 2</u> H_0 : 2 RU's H_1 : 1 RU	Não significativo	Não significativa	-5.02 -9.44	-2.60	-1.95
<u>Passo 3</u> H_0 : 1 RU H_1 : 0 RU	Significativo	Significativa	-6.34 -4.46 -3.50	-4.12	-3.49

No passo final, os coeficientes foram significativos na equação completa. A estatística -3,50 do terceiro coeficiente, maior do que o valor crítico a 1% e praticamente igual ao de 5%, não permite rejeitar a existência de uma raiz unitária (hipótese nula). A detecção de uma raiz unitária somente no terceiro passo confirma que a primeira diferença é suficiente neste caso.

A observação do correlograma de D1LOGAER mostra a presença de picos sazonais a cada 6 meses e que decrescem muito lentamente. A estacionariedade é obtida mediante aplicação adicional de uma diferença sazonal de 6 meses na variável D1LOGAER: obtém-se nova variável transformada, D1D6LOG, cujo correlograma declina rapidamente na FAC e é truncado na FACP a partir do 6° lag.

Figura 6.1 – Correlogramas: a) D1LOGAER b) D1D6LOG

Em resumo, fizemos a seguinte sequência de transformações:

AEREO = série no nível original

LOGAEREO = $\log(\text{AEREO})$

DLOGAEREO = $\Delta \log(\text{AEREO})$

D1D6LOG = $\Delta_6 \Delta \log(\text{AEREO})$

Tem-se, enfim, uma variável estacionária D1D6LOG com valores autocorrelacionados, conforme indicam os $p\text{-values} < 5\%$ no teste Q de Ljung-Box da Figura 6.1a. Com isto a série D1D6LOG pode ser modelada por um processo SARMA(p,q) x (P,Q)₆ que será estimado através da metodologia de Box-Jenkins. Consequentemente, o logaritmo da série original será um processo integrado de 1ª ordem: SARIMA(p,1,q) x (P,1,Q)₆.

6.1.1.2 Modelo selecionado

Foi selecionado um modelo SARIMA (1,1,0) x (1,1,0)₆ que minimizou os critérios AIC e SIC, entre várias possibilidades com coeficientes significativos testadas. Este modelo possui um termo autorregressivo de primeira ordem AR(1) com coeficiente -0,295198 em

combinação com um termo autorregressivo sazonal semestral SAR(6), de coeficiente - 0,743045, conforme saída abaixo gerada pelo *software* Eviews 6.0.

Dependent Variable: DLOG(AEREO,1,6)				
Method: Least Squares				
Date: 09/20/10 Time: 10:11				
Sample (adjusted): 2004M03 2007M12				
Included observations: 46 after adjustments				
Convergence achieved after 5 iterations				
Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	0.001022	0.003889	0.262873	0.7939
AR(1)	-0.295198	0.150253	-1.964677	0.0559
SAR(6)	-0.743045	0.096928	-7.665923	0.0000
R-squared	0.685169	Mean dependent var		0.004167
Adjusted R-squared	0.670526	S.D. dependent var		0.103623
S.E. of regression	0.059479	Akaike info criterion		-2.743388
Sum squared resid	0.152124	Schwarz criterion		-2.624129
Log likelihood	66.09792	Hannan-Quinn criter.		-2.698713
F-statistic	46.79067	Durbin-Watson stat		2.016224
Prob(F-statistic)	0.000000			
Inverted AR Roots	.82-.48i -.30	.82+.48i -.82-.48i	.00+.95i -.82+.48i	-.00-.95i

Quadro 6.1 – Estimação do modelo SARIMA (1,1,0) x (1,1,0)₆

O bom ajuste do modelo é confirmado pelo aspecto fortemente gaussiano (Jarque-Bera com *p-value* 94%) conforme o histograma dos resíduos (Gráfico 6.3), e pela ausência de autocorrelações significativas no respectivo correlograma dos resíduos na Figura 6.2.

Gráfico 6.3 – Análise dos resíduos: histograma

Correlogram of Residuals

Figura 6.2 – Análise dos resíduos: correlograma

A adequação de um modelo linear aos dados é confirmada neste caso pelo teste de McLeod-Li, que corresponde ao teste Q de Ljung-Box aplicado no quadrado dos resíduos do modelo selecionado. Os $p\text{-values} > 5\%$ nos lags da Figura 6.2 não permitem rejeição da hipótese nula (H_0 : ausência de efeito não linear do tipo ARCH) nos resíduos.

Portanto, tendo um modelo SARIMA adequadamente selecionado pela metodologia de Box-Jenkins, foi possível obter a previsão da série com este modelo no procedimento estático e dinâmico, cujos resultados estão nas Tabelas 6.2 e 6.3.

6.1.2 Rede de Elman: AEREO

6.1.2.1 Dados sem pré-processamento de tendência e sazonalidade

Inicialmente a série original AEREO foi apenas padronizada entre -1 e 1. Com estes dados padronizados foram treinadas redes de 1 a 20 neurônios, em pelo menos 10 tentativas de reinicialização dos pesos. Ao todo foram obtidas 200 redes treinadas desta maneira e uma delas, com 8 neurônios, foi selecionada porque apresentou o menor MSE de validação (MSE = 0,0326 na série padronizada). O melhor resultado de cada tamanho de rede pode ser visualizado no Gráfico 6.4 juntamente com o treinamento e validação cruzada da rede com 8 neurônios selecionada, visualizado no Gráfico 6.5.

Gráfico 6.4 – AEREO sem pré-processamento: melhores treinamentos de cada tamanho de rede

Gráfico 6.5 – AEREO sem pré-processamento: treinamento da rede selecionada

Para evitar um ajuste excessivo (*overfitting*) da rede aos dados, o treinamento foi interrompido com 929 épocas, porque o MSE de validação havia crescido 15 vezes consecutivamente. Neste caso, os pesos finais foram retroagidos em 15 épocas para serem fixados em seu melhor momento com respeito ao erro de validação.

Os Gráficos 6.6 e 6.7 mostram os resultados gerados pela rede selecionada com 8 neurônios, no procedimento de previsão estática e dinâmica. Nota-se visualmente o grande deslocamento da média das previsões em relação à média da série real no horizonte de previsão de ambos os procedimentos. Isto é comprovado pela alta proporção de viés na composição do MSE destas previsões, conforme Tabelas 6.2 e 6.3.

Gráfico 6.6 – Previsão estática sem pré-processamento de AEREO

Gráfico 6.7 – Previsão dinâmica sem pré-processamento de AEREO

6.1.2.2 Dados com pré-processamento

A partição dos dados em conjuntos de previsão, validação e aprendizagem é a mesma usada no treinamento sem pré-processamento. Entretanto, neste caso, o treinamento da rede foi realizado com dados transformados da seguinte forma:

- a) Aplicação de logaritmo natural nos valores da série original;
- b) Diferenciação regular de ordem 1;
- c) Diferenciação sazonal de ordem 6;
- d) Padronização no intervalo $[-1,1]$.

Os dados de toda a série modificada, conforme os quatro passos acima, foram submetidos à rede treinada para obter inicialmente uma previsão da variável transformada. A lista acima foi percorrida no sentido contrário, em uma transformação inversa, para obtenção das previsões na mesma unidade de medida da série original. Para estas previsões e transformações foi escrito um código MATLAB específico.

A rede com 10 neurônios selecionada gerou $MSE = 0,0585$ no intervalo de validação para a série padronizada. Este foi o menor valor obtido nas várias tentativas de treinamento, conforme Gráfico 6.8, a seguir.

Gráfico 6.8 – AEREO com pré-processamento: melhores treinamentos de cada tamanho de rede

Gráfico 6.9 – AEREO com processamento: treinamento da rede selecionada

A curva de validação cruzada da rede selecionada com 10 neurônios mostra no Gráfico 6.9 que após 200 épocas os pesos da rede já haviam convergido para seus valores finais. O treinamento foi interrompido após atingir o limite máximo de 1.000 épocas, sem necessidade de fazer parada prematura (*early-stopping*) na validação cruzada.

As figuras a seguir ilustram as previsões obtidas com a rede de 10 neurônios selecionada usando dados pré-processados. Visualmente as previsões com dados pré-processados aparentam ser mais próximas dos valores reais, tanto na média quanto nas suas oscilações, quando comparadas com previsões obtidas anteriormente sem tratamento prévio da tendência e sazonalidade nos dados originais. Isto será comprovado de forma objetiva com a predominância da participação de covariância no MSE das previsões com pré-processamento, conforme Tabelas 6.2 e 6.3.

Gráfico 6.10 – Previsão estática com pré-processamento de AEREO

Gráfico 6.11 – Previsão dinâmica com pré-processamento de AEREO

6.1.3 Comparação de previsões – AEREO

Com base na previsão estática (Tabela 6.2), as redes neurais não conseguiram superar a qualidade de previsão do modelo SARIMA que apresentou o menor TIC (5,3%) e o menor RMSE (439.398) no horizonte de previsão da série AEREO. Cabe lembrar que uma previsão

perfeita teria $TIC = 0$. Além disto, com o modelo SARIMA a parcela de covariância corresponde a 95,2% do MSE da previsão estática: uma predominância que é altamente desejável, com pouca proporção de viés e de variância, em uma boa previsão.

Analizando o impacto do pré-processamento, nota-se que a qualidade da previsão estática com as redes neurais foi melhorada significativamente, pois o TIC e o RMSE foram reduzidos com a adoção de dados pré-processados. A participação do viés no MSE da previsão estática sem pré-processamento foi relativamente alta (58%) e isto foi corrigido na previsão com dados pré-processados cujo MSE passou a ser dominado pela covariância (87%).

Tabela 6.2 – Procedimento estático de previsão de AEREO com SARIMA e redes de Elman

MÊS	Previsão Estática			
	AEREO	SARIMA	RNA08 (sem PP)	RNA10 (com PP)
jan/08	4.494.055	4.642.285	3.950.942	4.348.300
fev/08	3.765.600	3.647.467	3.282.834	3.501.100
mar/08	3.904.922	3.911.639	3.645.377	4.128.000
abr/08	3.988.844	4.355.694	3.577.088	4.711.100
mai/08	4.334.282	4.025.556	3.265.570	4.254.400
jun/08	3.997.441	4.298.520	3.989.261	4.196.300
jul/08	4.299.330	4.438.748	3.904.040	4.348.500
ago/08	4.385.300	3.449.132	3.259.982	3.508.200
set/08	3.664.170	4.425.195	3.765.504	4.462.300
out/08	3.832.816	4.454.635	3.873.175	3.668.500
nov/08	3.854.419	3.990.028	3.282.467	3.210.400
dez/08	4.395.544	4.112.204	3.329.067	4.181.700
	RMSE	439.398	632.860	465.880
	TIC	5,3%	8,2%	5,7%
	VIÉS	2,5%	58,2%	0,5%
	VAR	2,3%	0,0%	12,5%
	COV	95,2%	41,8%	87,0%

Com base na previsão dinâmica (Tabela 6.3), apenas a rede neural treinada com dados pré-processados superou levemente a qualidade de previsão do modelo SARIMA ao resultar no menor TIC(5,4%) e no menor RMSE (439.990) no horizonte de previsão. Sob o ponto de vista da composição do MSE de previsão, a predominância da covariância (que indica uma boa previsão) foi muito semelhante no caso da rede neural com pré-processamento (63,7%) e do modelo SARIMA (57,9%).

No entanto, a rede treinada sem pré-processamento ficou bem aquém do desempenho do

modelo SARIMA levando-se em conta o TIC(10,6%) e RMSE(788.150) da previsão com a rede. Note-se que o viés foi muito alto nas previsões sem pré-processamento, tanto estática (58,2%) quanto dinâmica (85,3%), e que esta característica indesejável do viés elevado foi substituída pela predominância da covariância no MSE das previsões com pré-processamento: estática (87%) e dinâmica (63,7%). O impacto do pré-processamento é, portanto, muito forte neste caso, melhorando significativamente a qualidade da previsão com as redes neurais.

Tabela 6.3 – Procedimento dinâmico de previsão de AEREO com SARIMA e redes de Elman

MÊS	Previsão Dinâmica			
	AEREO	SARIMA	RNA08 (sem PP)	RNA10 (com PP)
jan/08	4.494.055	4.642.285	3.950.942	4.348.300
fev/08	3.765.600	3.731.852	3.332.994	3.542.800
mar/08	3.904.922	3.924.311	3.376.907	3.129.800
abr/08	3.988.844	4.359.325	3.073.849	3.839.900
mai/08	4.334.282	4.291.871	3.230.782	4.199.200
jun/08	3.997.441	4.382.234	3.469.789	4.624.400
jul/08	4.299.330	4.761.593	3.122.856	4.331.800
ago/08	4.385.300	3.777.411	3.247.089	3.805.800
set/08	3.664.170	4.107.162	3.483.945	3.314.800
out/08	3.832.816	4.725.355	3.166.119	4.303.800
nov/08	3.854.419	4.692.885	3.245.299	3.985.300
dez/08	4.395.544	5.119.926	3.483.269	5.116.700
	RMSE	509.448	788.150	439.990
	TIC	6,0%	10,6%	5,4%
	VIÉS	34,7%	85,3%	0,5%
	VAR	7,4%	0,4%	35,8%
	COV	57,9%	14,4%	63,7%

6.2 Série mensal do consumo de adubos no Brasil

A série ADUBOS registra a venda mensal no mercado doméstico brasileiro de fertilizantes entregues ao consumidor final, isto é, agricultores e suas cooperativas, medida em toneladas, obtida no *website* da ANDA (Associação Nacional para Difusão de Adubos). A série tem 228 pontos, sendo os primeiros 216 correspondentes ao período de estimação entre janeiro de 1991 e dezembro de 2008, e os 12 últimos tomados para previsão entre janeiro e dezembro de 2009, conforme Gráfico 6.12.

No caso das redes neurais, a amostra de treinamento foi fracionada em 162 pontos para aprendizado (75%) e 54 pontos para validação (25%), como mostrado no Gráfico 6.13.

Gráfico 6.12 – Partição da série ADUBOS: Box-Jenkins

Gráfico 6.13 – Partição da série ADUBOS: redes neurais

6.2.1 Metodologia de Box-Jenkins

6.2.1.1 Transformações e testes de raízes unitárias

Aplicada inicialmente uma transformação logarítmica, seguida da primeira diferença do logaritmo da série ADUBOS, obteve-se nova variável DLOGADUB. A aplicação do teste de

Dickey-Pantula confirma a ordem de integração (1), com a não rejeição de hipótese nula no último passo do teste onde a estatística -3,38 está à direita, ou seja, é maior do que os respectivos valores críticos à 1% e 5%.

Tabela 6.4 – Teste de Dickey-Pantula em Log (ADUBOS)

Dickey-Pantula	Intercepto	Tendência	Estatísticas do teste	Vc a 1%	Vc a 5%
<u>Passo 1</u>					
H ₀ : 3 RU's H ₁ : 2 RU's	Não significativo	Não significativa	-17,30	-2,58	-1,95
<u>Passo 2</u>					
H ₀ : 2 RU's H ₁ : 1 RU	Não significativo	Não significativa	-12,45 -8,81	-2,58	-1,95
<u>Passo 3</u>					
H ₀ : 1 RU H ₁ : 0 RU	Significativo	Significativa	-17,62 -3,38 -9,53	-4.00	-3.43

O correlograma de DLOGADUB permanece com picos sazonais a cada 12 *lags*, e por isto, optou-se por uma diferenciação sazonal de 12 meses e a criação de uma nova variável DLOGADSAZ12.

Figura 6.3 – Correlogramas: a) DLOGADUB

b) DLOGADSAZ12

Em resumo, a busca de uma série estacionária levou às seguintes transformações:

ADUBOS = série no nível original

LOGADUB = $\log(\text{ADUBOS})$

DLOGADUB = $\Delta \log(\text{ADUBOS})$

DLOGADSAZ12 = $\Delta_{12} \Delta \log(\text{ADUBOS})$

O modelo a ser adotado desta forma será expresso como um SARIMA $(p,1,q) \times (P,1,Q)_{12}$, cujos parâmetros devem ser estimados.

6.2.1.2 Modelo selecionado

Foi selecionado um modelo SARIMA $(([2],[7]),1,[2]) \times (0,1,2)_{12}$ obtido somente após várias tentativas. O modelo selecionado minimizou os critérios AIC e SIC, entre algumas possibilidades com coeficientes significativos testadas. A notação inclui colchetes nos termos cuja ordem não obedece a uma sequência completa. Os coeficientes dos termos AR([2]), AR([7]), MA([2]), SMA(12) e SMA(24) são apresentados na Figura 6.4, gerada pelo *software*

Eviews 6.0.

Dependent Variable: DLOG(ADUBOS,1,12)				
Method: Least Squares				
Date: 09/14/10 Time: 19:48				
Sample (adjusted): 1992M09 2008M12				
Included observations: 196 after adjustments				
Convergence achieved after 14 iterations				
MA Backcast: OFF				
Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	0.000231	0.000410	0.564478	0.5731
AR(2)	0.540237	0.031435	17.18608	0.0000
AR(7)	-0.155408	0.063762	-2.437300	0.0157
MA(2)	-0.936523	0.034457	-27.17912	0.0000
SMA(12)	-0.529705	0.078868	-6.716358	0.0000
SMA(24)	-0.206600	0.077791	-2.655817	0.0086
R-squared	0.368175	Mean dependent var		-0.002476
Adjusted R-squared	0.351548	S.D. dependent var		0.162070
S.E. of regression	0.130509	Akaike info criterion		-1.204611
Sum squared resid	3.236206	Schwarz criterion		-1.104261
Log likelihood	124.0519	Hannan-Quinn criter.		-1.163985
F-statistic	22.14323	Durbin-Watson stat		1.871357
Prob(F-statistic)	0.000000			
Inverted AR Roots	.78+.26i	.78-.26i	.18+.67i	.18-.67i
	-.52-.51i	-.52+.51i	-.90	
Inverted MA Roots	.98	.97	.86-.23i	.86+.23i
	.85+.49i	.85-.49i	.63-.63i	.63-.63i
	.49+.85i	.49-.85i	.23-.86i	.23+.86i
	.00+.98i	-.00-.98i	-.23+.86i	-.23-.86i
	-.49-.85i	-.49+.85i	-.63-.63i	-.63+.63i
	-.85+.49i	-.85-.49i	-.86+.23i	-.86-.23i
	-.97	-.98		

Figura 6.4 – Estimação do modelo SARIMA $(([2],[7]),1,[2]) \times (0,1,2)_{12}$

O correlograma dos resíduos (Figura 6.5a) indica ausência de correlação serial, com *p-values* > 5% no teste Q de Ljung-Box. No entanto, a análise dos resíduos quadrados (teste de McLeod-Li) rejeita a hipótese nula de ausência de não linearidade, registrando *p-value* < 5% no lag 6, e *p-values* < 10% nos lags 7 e 8, conforme Figura 6.5b.

Figura 6.5 – a) correlograma dos resíduos, b) correlograma de resíduos quadrados

Adicionalmente, a hipótese de normalidade é rejeitada no teste de Jarque-Bera: os resíduos do modelo apresentam perfil leptocúrtico, isto é, elevada curtose em relação à Distribuição Normal, conforme Gráfico 6.14. De acordo com De Oliveira (2003, p.103) a presença de não normalidade com o perfil leptocúrtico sugere a necessidade de um modelo não linear do tipo ARCH. Porém, neste caso, o teste LM-ARCH descrito por Enders (2004, p. 119) e De Lima (2004, p. 44), aplicado no *Eviews 6.0*, contraditoriamente não rejeitou a hipótese nula (ausência de efeito ARCH).

Gráfico 6.14 – Análise de resíduos: histograma

Diante de resultados contraditórios, ou pouco significativos, conclui-se que o ajuste deste modelo linear não é perfeito e deixa resíduos que, apesar de serem considerados um ruído branco, não são gaussianos. Optou-se, porém, em não fugir ao escopo da utilização de modelos lineares, e na ausência de um modelo linear melhor, seguir utilizando este modelo para fazer a previsão da série ADUBOS.

Constatou-se neste caso que a metodologia de Box-Jenkins nem sempre é um procedimento fácil, podendo enfrentar dificuldades práticas. Entre outras coisas, há o fato das funções FAC e FACP serem amostrais e que não observamos as funções teóricas. Naturalmente só se adquire habilidade nesta metodologia, mediante a prática (GUJARATI, 2006, p. 678).

6.2.2 Rede de Elman: ADUBOS

6.2.2.1 Dados sem pré-processamento de tendência e sazonalidade

Inicialmente a série original ADUBOS foi apenas padronizada entre -1 e 1. Com estes dados padronizados foram treinadas redes de 1 a 20 neurônios, em pelo menos 10 tentativas de reinicialização dos pesos. Ao todo foram obtidas 200 redes treinadas desta maneira e uma delas, com 13 neurônios, foi selecionada porque apresentou o menor MSE de validação (MSE = 0,0173 na série padronizada). O melhor resultado de cada tamanho de rede pode ser visualizado no Gráfico 6.15 juntamente com o treinamento e validação cruzada da rede com 13 neurônios selecionada, visualizado no Gráfico 6.16.

Gráfico 6.15 – ADUBOS sem pré-processamento: melhores treinamentos de cada tamanho de rede

Gráfico 6.16 – ADUBOS sem processamento: treinamento da rede selecionada

O treinamento foi finalizado depois de 1000 épocas no procedimento de validação cruzada sem necessidade de parada prematura (*early-stopping*). Os pesos convergiram para seu valor final na rede selecionada, praticamente depois de 400 épocas.

Os Gráficos 6.17 e 6.18 mostram os resultados gerados pela rede selecionada com 13 neurônios, no procedimento de previsão estática e dinâmica. Visualmente há indicação de que as previsões em ambos os procedimentos se aproximaram bem da média e da variabilidade da série real. Isto será comprovado objetivamente pela alta proporção de covariância no MSE destas previsões conforme será visto nas Tabelas 6.5 e 6.6.

Gráfico 6.17 – Previsão estática sem pré-processamento de ADUBOS

Gráfico 6.18 – Previsão dinâmica sem pré-processamento de ADUBOS

6.2.2.2 Dados com pré-processamento

A partição dos dados em conjuntos de previsão, validação e aprendizagem é a mesma usada no treinamento sem pré-processamento. Entretanto, neste caso, o treinamento de outras 200 redes foi realizado com dados transformados da seguinte forma:

- a) Aplicação de logaritmo natural nos valores da série original;
- b) Diferenciação regular de ordem 1;
- c) Diferenciação sazonal de ordem 12;
- d) Padronização no intervalo $[-1,1]$.

Uma rede com 10 neurônios foi selecionada, pois gerou $MSE = 0,0506$ no intervalo de validação para a série padronizada. Este foi o menor valor obtido nas várias tentativas de treinamento, conforme Gráfico 6.19, a seguir.

Gráfico 6.19 – ADUBOS com pré-processamento: melhores treinamentos de cada tamanho de rede

Gráfico 6.20 – ADUBOS com processamento: treinamento da rede selecionada

Para evitar um ajuste excessivo (*overfitting*) da rede aos dados, o treinamento foi interrompido com 969 épocas, porque o MSE de validação havia crescido 15 vezes consecutivamente no Gráfico 6.20. Neste caso, os pesos finais foram retroagidos em 15 épocas para serem fixados em seu melhor momento com respeito ao erro de validação.

Os dados da série modificada, conforme os passos acima, foram submetidos à rede treinada para obter inicialmente uma previsão da variável transformada. A lista dos quatro passos foi percorrida no sentido contrário, em uma transformação inversa, para obtenção das previsões na mesma unidade da variável da série original. Para estas previsões e transformações foi escrito um código MATLAB específico.

O Gráfico 6.21 mostra os resultados gerados pela rede selecionada com 10 neurônios, no procedimento de previsão estática e dinâmica. Visualmente há indicação de que as previsões em ambos os procedimentos acompanharam relativamente bem o sentido das oscilações da série real mas tiveram alguns pontos muito divergentes. Isto será comprovado objetivamente pela baixa participação da componente de variância, e por alta proporção de viés, no MSE das previsões, conforme será visto nas Tabelas 6.5 e 6.6.

Gráfico 6.21 – Previsão estática com pré-processamento de ADUBOS

Gráfico 6.22 – Previsão dinâmica com pré-processamento de ADUBOS

6.2.2.3 Comparação de previsões – ADUBOS

Com base nos resultados do procedimento estático, descritos na Tabela 6.5, as redes neurais não superaram a qualidade da previsão com o modelo linear SARIMA adotado, pois este apresentou o menor TIC (3,6%) e o menor RMSE (142.425) no horizonte de previsão. Além disto, a participação da covariância no MSE da previsão com o modelo SARIMA é muito mais alta (97,2%) do que nas previsões com redes de Elman, cujo máximo de participação da covariância foi de 76,9%.

A introdução do pré-processamento nos dados trouxe previsões relativamente melhores nas redes neurais quanto ao TIC e RMSE, porém houve uma diminuição da componente de covariância que se limitou a 56,5% do MSE do horizonte de previsão estática.

Tabela 6.5 – Procedimento estático de previsão de ADUBOS com SARIMA e redes de Elman

MÊS	Previsão Estática			
	ADUBOS	SARIMA	RNA13 (sem PP)	RNA10 (com PP)
jan/09	1.313.453	1.172.298	1.175.500	1.191.300
fev/09	1.362.398	1.335.994	1.282.100	1.313.600
mar/09	1.381.814	1.415.115	1.908.700	1.335.300
abr/09	1.053.783	1.287.420	1.937.800	1.252.700
mai/09	1.358.762	1.449.770	1.478.300	1.148.000
jun/09	1.803.792	1.943.528	1.892.800	1.536.700
jul/09	2.419.063	2.171.802	2.231.500	1.784.200
ago/09	2.573.790	2.769.174	2.575.300	2.268.100
set/09	2.704.962	2.733.313	2.440.900	2.722.000
out/09	2.883.244	2.834.832	2.488.600	2.770.300
nov/09	2.173.906	1.990.202	2.097.100	1.842.800
dez/09	1.441.854	1.497.390	1.468.600	1.393.800
	RMSE	142.425	338.390	257.610
	TIC	3,6%	8,6%	6,8%
	VIÉS	0,6%	1,6%	38,3%
	VAR	2,2%	21,5%	5,3%
	COV	97,2%	76,9%	56,5%

Tomando em conta o procedimento dinâmico de previsão com as redes de Elman, cujos resultados estão na Tabela 6.6, novamente as redes neurais ficaram atrás do modelo linear SARIMA na qualidade de previsão. O TIC e o RMSE do modelo SARIMA foram menores do que com as redes e, curiosamente, o pré-processamento piorou os resultados nesta série, causando impacto negativo na qualidade da previsão medida pelo aumento do TIC (de 16,5% para 22,4%), aumento do RMSE (de 650.310 para 752.860) e deterioração da participação da covariância (de 98,6% para 39,1%) no MSE do horizonte de previsão dinâmica.

Tabela 6.6 – Procedimento dinâmico de previsão de ADUBOS com SARIMA e redes de Elman

Previsão Dinâmica				
MÊS	ADUBOS	SARIMA	RNA13 (sem PP)	RNA10 (com PP)
jan/09	1.313.453	1.172.298	1.175.500	1.191.300
fev/09	1.362.398	1.192.417	1.191.800	1.328.200
mar/09	1.381.814	1.295.636	1.756.400	1.614.100
abr/09	1.053.783	1.216.528	2.050.300	1.042.900
mai/09	1.358.762	1.698.801	2.381.500	1.269.100
jun/09	1.803.792	2.253.953	2.472.500	1.188.900
jul/09	2.419.063	2.666.353	2.585.600	1.753.200
ago/09	2.573.790	2.896.121	2.710.100	2.146.100
set/09	2.704.962	3.189.129	2.193.300	1.425.700
out/09	2.883.244	3.122.322	1.677.400	1.450.100
nov/09	2.173.906	2.119.153	1.446.500	1.003.700
dez/09	1.441.854	1.399.006	1.216.900	629.820
	RMSE	268.089	650.310	752.860
	TIC	6,5%	16,5%	22,4%
	VIÉS	29,6%	0,2%	50,6%
	VAR	28,6%	1,2%	10,3%
	COV	41,8%	98,6%	39,1%

6.3 Série da PMC: artigos farmacêuticos, médicos e outros artigos

A Pesquisa Mensal de Comércio (PMC), realizada pelo IBGE, tem como objetivo produzir indicadores que permitam acompanhar a evolução conjuntural do comércio varejista e de seus principais segmentos. As vendas de artigos farmacêuticos, médicos, ortopédicos, produtos de perfumaria e cosméticos integram a PMC na forma de um índice que é transcrito aqui pela variável FARMA.

A série FARMA tem 84 pontos plotados no Gráfico 6.23, sendo os primeiros 72 correspondentes ao período de estimação entre janeiro de 2003 e dezembro de 2008, e os 12 últimos tomados para previsão entre janeiro e dezembro de 2009. No caso das redes neurais a amostra de estimação foi fracionada em 54 pontos para aprendizado (75%) e 18 pontos para validação (25%).

Vendas reais - varejo - artigos farma., méd., ortoped. e perfum.

Gráfico 6.23 – Partição da série FARMA: Box-Jenkins

Vendas reais - varejo - artigos farma., méd., ortoped. e perfum.

Gráfico 6.24 – Partição da série FARMA: redes neurais

6.3.1 Metodologia de Box-Jenkins

6.3.1.1 Transformações e testes de raízes unitárias

A exemplo das séries anteriores, foi aplicada a primeira diferença no logaritmo na variável

original (FARMA), obtendo-se nova variável transformada DLOGFARMA.

A ordem de integração foi testada através do teste de Dickey-Pantula partindo da hipótese nula inicial de 3 raízes unitárias (RU). A detecção de uma raiz unitária somente no terceiro passo confirma neste caso que basta uma diferenciação regular para estacionar a série.

Tabela 6.7 – Teste de Dickey-Pantula em Log (FARMA)

Dickey-Pantula	Intercepto	Tendência	Estatísticas do teste	Vc a 1%	Vc a 5%
Passo 1 H_0 : 3 RU's H_1 : 2 RU's	Não significativo	Não significativa	-17,01	-2,60	-1,95
Passo 2 H_0 : 2 RU's H_1 : 1 RU	Significativo	Não significativa	-4,83 -11,33	-3,55	-2,91
Passo 3 H_0 : 1 RU H_1 : 0 RU	Significativo	Significativa	-5,85 -5,30 -2,97	-4.12	-3.49

No passo final, os coeficientes foram significativos na equação do teste na sua forma completa. A estatística -2,97 do terceiro coeficiente, maior do que os respectivos valores críticos (1% e de 5%), não permite rejeitar a existência de uma raiz unitária (hipótese nula).

Adicionalmente, a variável DLOGFARMA foi diferenciada sazonalmente em 12 meses para eliminação dos picos sazonais visíveis no seu correlograma (Figura 6.6a).

Figura 6.6 – a) correlograma dos resíduos, b) correlograma de resíduos quadrados

O correlograma da série transformada, D1D12LFARMA, já não apresenta picos sazonais e está truncado, conforme indicado pela Figura 6.6b, com indício da presença de possíveis termos ARMA de baixa ordem no modelo.

Em resumo, temos:

FARMA = série no nível original

LOGFARMA = $\log(\text{FARMA})$

DLOGFARMA = $\Delta \log(\text{FARMA})$

D1D12LFARMA = $\Delta_{12} \Delta \log(\text{FARMA})$

Neste momento, podemos dizer que a série transformada D1D12LFARMA será modelada por um processo SARMA(p,q) x (P,Q)₁₂ que será estimado através da metodologia de Box-Jenkins e, portanto, a série do logaritmo da variável original FARMA será expressa por um SARIMA(p,1,q) x (P,1,Q)₁₂.

6.3.1.2 Modelo selecionado

Foi selecionado um modelo SARIMA (2,1,0) x (1,1,0)₁₂ que minimizou os critérios AIC e SIC, entre algumas possibilidades obtidas com coeficientes significativos. Segue no Quadro

6.2 a respectiva saída da estimação do modelo gerada no Eviews 6.0.

Dependent Variable: DLOG(FARMA,1,12)				
Method: Least Squares				
Date: 09/14/10 Time: 15:44				
Sample (adjusted): 2005M04 2008M12				
Included observations: 45 after adjustments				
Convergence achieved after 8 iterations				
Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	0.001838	0.001289	1.425827	0.1615
AR(1)	-0.689562	0.137190	-5.026337	0.0000
AR(2)	-0.312530	0.142726	-2.189721	0.0343
SAR(12)	-0.362142	0.147337	-2.457918	0.0183
R-squared	0.418002	Mean dependent var		0.002580
Adjusted R-squared	0.375417	S.D. dependent var		0.029270
S.E. of regression	0.023132	Akaike info criterion		-4.610490
Sum squared resid	0.021939	Schwarz criterion		-4.449898
Log likelihood	107.7360	Hannan-Quinn criter.		-4.550623
F-statistic	9.815658	Durbin-Watson stat		2.055539
Prob(F-statistic)	0.000053			
Inverted AR Roots	.89+.24i	.89-.24i	.65-.65i	.65-.65i
	.24+.89i	.24-.89i	-.24-.89i	-.24+.89i
	-.34-.44i	-.34+.44i	-.65-.65i	-.65+.65i
	-.89-.24i	-.89+.24i		

Quadro 6.2 - Estimação do modelo SARIMA (2,1,0) x (1,1,0)₁₂.

O bom ajuste do modelo é confirmado fortemente (*p-value* 94%) pelo teste de Jarque-Bera, conforme histograma dos resíduos apresentado no Gráfico 6.25, e pela ausência de autocorrelações significativas no respectivo correlograma dos resíduos (Figura 6.7a).

Gráfico 6.25 – Análise de resíduos: histograma

Figura 6.7 – a) correlograma dos resíduos, b) correlograma de resíduos quadrados

A adequação de um modelo linear aos dados é confirmada neste caso pelo teste de McLeod-Li, que corresponde ao teste Q de Ljung-Box aplicado no quadrado dos resíduos do modelo selecionado, conforme Figura 6.7b. Os *p-values* > 5% em todos os lags dos resíduos quadrados não indicam presença de heterocedasticidade condicionada. Com isto validamos o modelo selecionado, dentro da metodologia de Box e Jenkins, ao constatar ausência de autocorrelação serial nos resíduos do modelo nos quais há também características de ruído branco gaussiano. A previsão da série FARMA com este modelo é apresentada nas Tabelas 6.8 e 6.9.

6.3.2 Rede de Elman: FARMA

6.3.2.1 Dados sem pré-processamento de tendência e sazonalidade

Inicialmente a série original FARMA foi apenas padronizada entre -1 e 1. Com estes dados padronizados foram treinadas redes de 1 a 20 neurônios, em pelo menos 10 tentativas de reinicialização dos pesos. Ao todo foram obtidas 200 redes treinadas desta maneira e uma delas, com 8 neurônios, foi selecionada porque apresentou o menor MSE de validação (MSE = 0,0223 na série padronizada). O melhor resultado de cada tamanho de rede pode ser visualizado no Gráfico 6.26 juntamente com o treinamento e validação cruzada da rede com 8 neurônios selecionada, visualizado no Gráfico 6.27.

Gráfico 6.26 – FARMA sem pré-processamento: melhores treinamentos

Gráfico 6.27 – FARMA sem processamento: treinamento da rede selecionada

O ajuste excessivo (*overfitting*) da rede aos dados foi evitado com o treinamento interrompido após 277 épocas, quando o MSE de validação havia crescido 15 vezes consecutivamente. Os pesos finais foram então retroagidos em 15 épocas para serem fixados em seu melhor momento com respeito ao erro de validação.

Os Gráficos 6.28 e 6.29 mostram os resultados gerados pela rede selecionada com 8 neurônios, no procedimento de previsão estática e dinâmica. Nota-se visualmente que a rede não é capaz de fazer boas previsões com os dados sem pré-processamento, havendo grande

distanciamento das previsões em relação à média da série real, no horizonte de previsão de ambos os procedimentos. Isto é comprovado nas Tabelas 6.8 e 6.9 com a elevada proporção de viés na composição do MSE destas previsões.

Gráfico 6.28 – Previsão estática sem pré-processamento de FARMA

Gráfico 6.29 – Previsão dinâmica sem pré-processamento de FARMA

6.3.2.2 Dados com pré-processamento

A partição dos dados em três conjuntos (aprendizagem, validação, e previsão) é a mesma usada no treinamento sem pré-processamento. Entretanto, neste caso, outras 200 redes foram treinadas com dados transformados da seguinte forma:

- a) Aplicação de logaritmo natural nos valores da série original;
- b) Diferenciação regular de ordem 1;
- c) Diferenciação sazonal de ordem 12;
- d) Padronização no intervalo $[-1,1]$.

Uma rede com 13 neurônios foi selecionada porque gerou o menor MSE no intervalo de validação, com valor de $MSE = 0,0221$ na série transformada. Este foi o menor valor obtido nas 200 tentativas de treinamento, cujos melhores resultados em cada tamanho de rede estão mostrados no gráfico 6.30. As trajetórias de validação cruzada do MSE, de aprendizado e de validação, da rede selecionada estão no gráfico 6.31 e mostram que o treinamento seguiu até o limite máximo de 1.000 épocas sem necessidade de parada prematura (*early-stopping*).

Gráfico 6.30 – FARMA com pré-processamento: melhores treinamentos

Gráfico 6.31 – FARMA com processamento: treinamento da rede selecionada

O percurso no sentido contrário, em uma transformação inversa, dos quatro passos acima, permitiu gerar a previsão estática e dinâmica na mesma unidade de medida da série original. Para estas transformações e respectivas previsões foi escrito um código MATLAB específico, cuja saída gráfica para a rede selecionada está nos Gráficos 6.32 e 6.33.

É possível observar visualmente que, no procedimento estático, a rede selecionada de 13 neurônios foi capaz de gerar previsões bem próximas dos valores reais. Isto é comprovado objetivamente na Tabela 6.8 com alta participação da covariância no MSE de previsão estática. Porém, na previsão dinâmica nota-se visualmente que a média dos valores previstos certamente está bem acima da média dos valores reais no horizonte de previsão. Por isto, obteve-se na Tabela 6.9 uma previsão dinâmica com alta proporção de viés (83,6%) na composição do MSE neste horizonte de previsão.

Gráfico 6.32 – Previsão estática com pré-processamento de FARMA

Gráfico 6.33 – Previsão dinâmica com pré-processamento de FARMA

6.3.3 Comparação de previsões – FARMA

Levando em conta resultados da previsão estática (Tabela 6.8) para a série FARMA, as redes neurais não conseguiram superar a qualidade de previsão do modelo SARIMA. Este apresentou o menor TIC (1%) e o menor RMSE (3) no horizonte de previsão da série FARMA. Além disto, com o modelo SARIMA, a parcela relativa da covariância segue

predominante (83,9%) entre as componentes do MSE desta previsão estática.

Sob o ponto de vista de pré-processamento, nota-se que a qualidade da previsão estática com redes neurais foi melhorada significativamente nesta série FARMA com o uso de dados dessazonalizados e sem tendência. O TIC e o RMSE das previsões com redes foram reduzidos com a adoção de dados pré-processados no treinamento. A participação de viés no MSE da previsão sem pré-processamento foi bem elevada (74,1%) e foi eliminada na previsão com dados pré-processados, dando lugar a uma maciça participação da covariância (99,6%) na composição do MSE de previsão estática. Isto revela o benefício do tratamento prévio dos dados quanto à eliminação da sazonalidade e tendência.

Tabela 6.8 – Procedimento estático de previsão de FARMA com SARIMA e redes de Elman

MÊS	Previsão Estática			
	FARMA	SARIMA	RNA08 (sem PP)	RNA13 (com PP)
jan/09	148	153	141	158
fev/09	138	139	122	135
mar/09	160	157	132	148
abr/09	155	156	141	152
mai/09	161	166	141	168
jun/09	159	161	142	160
jul/09	170	166	135	161
ago/09	168	168	129	165
set/09	164	169	117	175
out/09	173	173	113	177
nov/09	165	166	115	164
dez/09	192	193	115	186
	RMSE	3	40	7
	TIC	1,0%	13,6%	2,2%
	VIÉS	14,9%	74,1%	0,1%
	VAR	1,3%	0,2%	0,3%
	COV	83,9%	25,7%	99,6%

Considerando a previsão dinâmica (Tabela 6.9), manteve-se a incapacidade das redes neurais em superar as previsões com o modelo linear que apresentou o menor TIC (1,9%) e o menor RMSE (6) em um ajuste bem melhor do que o obtido com as redes, neste caso.

Sob o ponto de vista da composição do MSE de previsão, todas as previsões dinâmicas foram

fortemente impactadas pela componente de viés, deixando pouquíssimo espaço para a proporção de covariância. Mesmo assim, o modelo linear foi o que contou com a maior participação da covariância (apenas 15,5%) dentro do MSE da previsão.

Levando em conta apenas as redes neurais, percebe-se que neste caso dinâmico o efeito do pré-processamento também foi benéfico, reduzindo o TIC de 8,6% para 4,9%, e reduzindo o RMSE de 26 para 7, sem alterar significativamente as proporções de viés, variância e covariância na composição do MSE no horizonte de previsão, conforme a Tabela 6.9:

Tabela 6.9 – Procedimento dinâmico de previsão de FARMA com SARIMA e redes de Elman

MÊS	Previsão Dinâmica			
	FARMA	SARIMA	RNA08 (sem PP)	RNA13 (com PP)
jan/09	148	153	141	158
fev/09	138	141	123	146
mar/09	160	160	139	173
abr/09	155	159	142	163
mai/09	161	169	143	176
jun/09	159	165	142	182
jul/09	170	172	140	177
ago/09	168	174	140	186
set/09	164	174	140	183
out/09	173	179	140	188
nov/09	165	173	140	181
dez/09	192	202	140	223
	RMSE	6	26	17
	TIC	1,9%	8,6%	4,9%
	VIÉS	77,2%	82,0%	83,6%
	VAR	7,3%	8,7%	10,2%
	COV	15,5%	9,3%	6,2%

7 CONCLUSÃO

As principais conclusões deste trabalho são apresentadas a seguir, de forma objetiva, em comparação direta com os objetivos do trabalho expostos no primeiro capítulo:

- a) A viabilidade de fazer previsões de séries temporais nacionais de varejo com redes neurais artificiais de Elman foi investigada e os resultados deixaram a desejar nos casos em que não houve tratamento prévio, quanto à sazonalidade e à tendência, nos dados originais das três séries analisadas: AEREO, ADUBOS e FARMA;
- b) A avaliação da qualidade das previsões para 12 meses, em condição retrospectiva, foi realizada mediante o coeficiente de Theil (TIC), pelo RMSE e pela proporção de covariância no MSE. Além destes critérios objetivos e quantitativos, também foi possível constatar visualmente o afastamento das previsões em relação aos valores reais quando não houve pré-processamento dos dados;
- c) O pré-processamento, que eliminou a tendência e sazonalidade por meio de diferenças, tendeu a surtir um efeito muito benéfico na qualidade das previsões com as redes de Elman. Este benefício, contudo, não foi unânime e encontrou exceção no caso da previsão dinâmica da série ADUBOS;
- d) De uma maneira geral, as previsões das três séries com redes de Elman não superaram previsões obtidas por meio de modelo lineares via metodologia de Box-Jenkins. A previsão com a rede de Elman foi apenas ligeiramente superior em um único caso (AEREO, previsão dinâmica), com pouca vantagem comparativa.

Tomando por base as questões de pesquisa, também enunciadas no primeiro capítulo, cabe responder como segue:

- As redes de Elman foram capazes de gerar previsões das três séries temporais analisadas, porém a eficiência destas redes como ferramenta de previsão não pôde ser estabelecida em termos absolutos. A mera submissão à rede dos dados, sem um tratamento prévio, trouxe resultados insatisfatórios, bastando a simples observação visual para constatar uma forte discrepância entre as previsões sem pré-processamento e os respectivos valores reais de cada série;

- As transformações adotadas, e que eliminaram a tendência e a sazonalidade das séries originais, contribuíram na maioria das situações para melhorar significativamente a qualidade das previsões via redes de Elman;
- A superioridade da previsão através das redes de Elman não foi constatada em relação aos modelos lineares. Pelo contrário, os modelos lineares geraram previsões melhores, ou muito semelhantes, em relação às redes neurais.

Os resultados desta pesquisa não são de todo inesperados, apesar de desfavoráveis às redes de Elman nas condições estudadas. Gomes (2005) obteve resultados mistos, com redes recorrentes, redes alimentadas à frente e modelos lineares se alternando como melhores modelos preditivos para séries temporais de inflação, dependendo do país analisado, e do horizonte de previsão. Anteriormente, Thawornwong e Enke (2004) compilaram 45 periódicos a respeito de redes neurais no período 1996-2002, concluindo que o grau de sucesso com redes neurais oscilava muito em relação a outros métodos de previsão. Assim, por exemplo, tanto More e Deo (2003) que usaram redes recorrentes de Jordan com dados financeiros, quanto Cholewo e Zurada (1997), que usaram redes de Elman com séries extraídas da natureza, chegaram a resultados superiores em comparação a modelos lineares. Por outro lado, nosso resultado se alinha com Brax (2000) que também não encontrou nenhuma evidência de vantagem comparativa a favor de redes de recorrentes (Elman/Jordan) em relação a redes alimentadas à frente e a modelos lineares, na previsão de vendas de um varejista sueco.

Feitas estas ponderações sobre os resultados específicos deste trabalho, pode-se tecer algumas considerações adicionais sobre aspectos relacionados ao seu desenvolvimento, dificuldades, limitações e possíveis variações ou extensões do mesmo, como segue.

Apesar da concepção inicial do estudo ter sido comparar o desempenho de modelos distintos e de comprovar a superioridade de um em relação ao outro, rapidamente chegou-se à constatação de que uma análise comparativa com base nas três séries estudadas não poderia ser motivo de generalização dos resultados obtidos: tanto no caso das redes, quanto nos modelos lineares, há muitas decisões baseadas em heurísticas para se chegar a um modelo selecionado. A interpretação dos resultados só pode se dar, portanto, dentro das condições específicas deste estudo.

No caso das redes, há a inicialização dos pesos através de tentativas e de vários outros parâmetros de ampla escolha, tais como o número de camadas, o número de neurônios por camada, a seleção de entradas da rede, a escolha das funções de ativação, do algoritmo de treinamento e da taxa de aprendizado. No caso dos modelos estocásticos, há a decisão sobre diferenciações nos dados que pode tomar rumos distintos, inclusive com outras transformações possíveis em busca da estacionariedade da série.

Com respeito às ferramentas de computação, os algoritmos de treinamento de redes neurais recorrentes ainda não estão facilmente disponíveis em *softwares* de redes neurais de uso amigável e a um custo acessível. Isto foi uma barreira para a utilização das redes recorrentes neste trabalho e, possivelmente, o seja para a utilização mais frequente destas redes. A necessidade de se programar o procedimento de validação cruzada no MATLAB foi um exemplo desta dificuldade, pois contava-se inicialmente com a utilização bastante simples da interface gráfica da ferramenta *Neural Networks Toolbox* do MATLAB, algo que ao fim não foi possível. Outros *softwares* também foram descartados por não respaldarem suficientemente o treinamento de redes recorrentes ou por não apresentarem consistência na disponibilidade de recursos. No caso do *Emergent*, por exemplo, o seu *website* <http://grey.colorado.edu/emergent/index.php/Backpropagation> acessado pela última vez em 01/11/2010, revelava indisponibilidade do algoritmo de retropropagação para redes recorrentes, ao contrário do divulgado em Aisa *et al* (2008).

A obtenção de séries temporais com dados de boa qualidade oriundos do varejo foi outra dificuldade encontrada ao longo deste trabalho. Fouto (2008, p. 104) considera que o período pós-plano Real, aliado a uma melhora nas bases de dados nacionais, deveria contribuir para se realizar mais análises de séries temporais brasileiras. No caso de séries sazonais, este período de carência de dados deve se estender ainda um pouco mais até que se juntem suficientes períodos sazonais para análise. O maior problema não está em obter as séries, visto que órgãos como o IBGE possuem muitos dados, mas no fato de que elas devem ser suficientemente longas para análise estatística. Apesar disto, as séries utilizadas resultaram em um bom ajuste do modelo linear das séries AEREO e FARMA, em que pesem as dificuldades relatadas com a série ADUBOS.

Neste trabalho foi utilizada uma arquitetura muito simples de rede recorrente com apenas uma

entrada alimentada por valores da própria variável a ser prevista, gerando uma saída única. Segundo Walczak (2004), a inclusão de variáveis exógenas, além da própria série, pode contribuir significativamente para melhorar a qualidade de previsão.

As redes foram treinadas neste trabalho pelo algoritmo de retropropagação no tempo (BPTT). Sabendo que o algoritmo de treinamento desempenha um papel fundamental nos resultados e, ressalvada a dificuldade já mencionada a respeito dos *softwares* de redes neurais, outros algoritmos poderiam ser objeto de pesquisa. Por exemplo, o filtro de Kalman estendido, considerado mais eficiente, pode convergir para uma boa solução quando o BPTT falha, conforme Hagner *et al* (2001, p. 10) e De Oliveira (2007, p. 80).

A ideia de análise de sensibilidade com respeito à escolha das redes deixa em aberto uma investigação sobre a conclusão a que se chegaria caso outras redes tivessem sido escolhidas no processo de tentativa e erro do treinamento das redes. De forma similar, deve-se apontar que neste trabalho os resultados obtidos via redes neurais foram apresentados sem menção de intervalos de confiança na predição dos valores das séries. Isto levanta a oportunidade de se estender a aplicação da teoria estatística de intervalos de confiança em previsões para redes recorrentes, tal como já demonstrado em De Oliveira (2007, p. 46-52) no caso de redes alimentadas à frente.

Por último, não se deveria descartar a comparação do resultado via redes neurais com as previsões de modelos tradicionais, do tipo média móvel e alisamento exponencial, que talvez possam fazer bem o papel de referência comparativa em substituição aos modelos de Box-Jenkins nas séries analisadas, certamente com mais parcimônia.

REFERÊNCIAS

- AAKER, David A. *et al.* **Marketing Research**. 5th ed. USA: John Wiley & Sons, 1995.
- ABELÉM, Antônio Jorge Gomes. **Redes Neurais Artificiais na previsão de séries temporais**. Rio de Janeiro, 1994. Dissertação (Mestrado em Engenharia Elétrica) – Pontifícia Universidade Católica do Rio de Janeiro. Disponível em: <http://www.uepa.br/prof/cmvserra/series-temporais/tes_0012.pdf>. Acesso em: 14/03/2010.
- ABRAHAM, Bovas; LEDOLTER, Johannes. **Statistical methods for forecasting**. Hoboken, New Jersey: John Wiley & Sons, 2005.
- AISA Brad *et al.* *The Emergent neural modeling system*. **Neural Networks**, 21, 2008, p. 1.146-1.152.
- ANDERSON, David R. *et al.* **Estatística aplicada à administração e economia**. 2. ed. São Paulo: Thomson Learning, 2007.
- ARTIFICIAL NEURAL NETWORK AND COMPUTATIONAL INTELLIGENCE FORECASTING COMPETITION. NNGC1. **Time series forecasting competition for neural networks and computational intelligence**, 2009. Disponível em: <<http://www.neural-forecasting-competition.com/index.htm>>. Acesso em: 25/07/2009.
- BALAGUER, Emili *et al.* *Predicting Service request in support centers based on nonlinear dynamics, ARMA modeling and neural networks*. **Expert Systems with Applications**, 34, 2008, p. 665-672.
- BALLINI, Rosangela. **Análise e previsão de vazões utilizando modelos de séries temporais, redes neurais e redes neurais nebulosas**. Campinas, 2000. Tese (Doutorado em Engenharia Elétrica) – Faculdade de Engenharia Elétrica e Computação da Universidade Estadual de Campinas.
- BIONDI NETO, Luiz *et al.* Previsão da vazão afluente mensal do rio São Francisco usando-se redes de Elman. **Engevista**, v. 7, n. 1, p. 63-72, 04/2005. Disponível em: <http://www.uff.br/engevista/1_7Engevista5.pdf>. Acesso em: 06/04/2010.
- BOX, George. E.; JENKINS, G. M. **Time series analysis: forecasting and control**. San Francisco: Holden Day, 1970.
- BRAGA, Antonio de Pádua *et al.* **Redes neurais artificiais: teoria e aplicações**. Rio de Janeiro: LTC, 2000.

BRAGA, Márcio Bobik. In: VASCONCELLOS, Marco Antonio Sandoval; ALVES, Denisard. (Coord.). **Manual de Econometria**. São Paulo: Atlas, 2000.

BRAX, Christofer. *Recurrent neural networks for time-series prediction*. University of Skövde, Suécia, Departamento de ciências da computação, 2000.

BRUNELLI, U. *et al. Three hours ahead prevision of SO₂ pollutant concentration using an Elman neural based forecaster*. **Building and Environment**, v. 43, p. 309-314, 2008.

CALDWELL, Joseph George. *The Box-Jenkins Forecasting Technique*. January 1971 (Reformatted Sept. 2006). Disponível em: <<http://www.foundationwebsite.org/BoxJenkins.pdf>>. Acesso em: 24/06/2010.

CARMO, Heron Carlos E. do *et al.* Modelo de Regressão Linear Geral. Cap. 2. In: VASCONCELLOS, Marco Antonio Sandoval; ALVES, Denisard. (Coord.). **Manual de Econometria**. São Paulo: Atlas, 2000.

CHIANG, Yen-Ming *et al.* *Comparison of static-feedforward and dynamic-feedback neural networks for rainfall-runoff modeling*. **Journal of Hydrology**, 290, p. 297-311, 2004.

CHOLEWO, Tomasz J.; ZURADA, Jacek M. *Sequential network construction for time series prediction*. In: **Proceedings of the IEEE International Joint Conference on Neural Networks**, p. 2.034-2.039, Houston, Texas, USA, June-9-12, 1997.

CHOPRA, S.; MEINDL, P. **Gerenciamento da Cadeia de Distribuição**. São Paulo: Prentice Hall, 2003.

CLEMENTS, Michael P.; HENDRY, David F. *Forecasting non-stationary economic time series*. USA: MIT, 1999.

CORRAR, Luiz. J. *et al* (Org.). **Análise multivariada**. São Paulo: Atlas, 2007.

CRONE, S. F. *IEEE Summer School in Computational Intelligence EVIC'05. Forecasting with artificial networks: Tutorial EVIC*. Santiago do Chile. 2005. Disponível em: <www.neural-forecasting.com>. Acesso em: 30/09/2009.

DE ALMEIDA, Fernando Carvalho; NAKAO, Silvio Hiroshi. In: CORRAR, Luiz. J. *et al* (Org.). **Análise multivariada**. São Paulo: Atlas, 2007.

DE LIMA, Fabiano Guasti. **Um método de análise e previsão de sucessões cronológicas unidimensionais lineares e não-lineares**. São Paulo, 2004. Tese (Doutorado em Administração) – Faculdade de Economia, Administração e Contabilidade da Universidade de São Paulo.

DE OLIVEIRA, Mauri Aparecido. **Previsão de sucessões cronológicas econômico-financeiras por meio de redes neurais artificiais recorrentes de tempo real e de processos ARMA-GARCH: um estudo comparativo quanto à eficiência de previsão.** São Paulo, 2003. Dissertação (Mestrado em Administração) – Faculdade de Economia, Administração e Contabilidade da Universidade de São Paulo.

_____. **Aplicação de redes neurais artificiais na análise de séries temporais econômico-financeiras.** São Paulo, 2007. Tese (Doutorado em Administração) – Faculdade de Economia, Administração e Contabilidade da Universidade de São Paulo.

DE OLIVEIRA, Mauri Aparecido; FÁVERO, Luiz Paulo Lopes. **Uma breve descrição de algumas técnicas para análise de séries temporais: séries de Fourier, wavelets, ARIMA, modelos estruturais para séries de tempo e redes neurais.** VI Semead: Ensaio de MQI. 2003. Disponível em: <<http://www.ead.fea.usp.br/Semead/6semead/MQI.htm>>. Acesso em: 28/10/2010.

DICIONÁRIO Michaelis da língua portuguesa. Disponível em: <www.michaelis.uol.com.br>. Acesso em: 20/10/2010.

DICKEY, David A.; FULLER, Wayne A. *Likelihood ratio statistics for autoregressive time series with a unit root.* *Econometric.* v. 49 (4), p. 1.057-1.073, 1981.

DICKEY, David A.; PANTULA, Sastry G. *Determining the order of differencing in autoregressive processes.* *Journal of business & economic statistics*, October 1987, v. 5, n. 4, p.18-24.

DORFFNER, Georg. *Neural networks for time series processing.* *Neural Network World*, 4 (6), v. 6, n. 4, 1996, p. 447- 468.

EHLERS, R. S. **Análise de séries temporais.** 5. ed. 2009. Disponível em: <<http://www.icmc.usp.br/~ehlers/notas/stemp.pdf>>. Acesso em: 17/09/2010.

ELMAN, Jeffrey L. *Finding structure in time.* *Cognitive Science*, 14, p. 179-211, 1990. Disponível em: <<http://crl.ucsd.edu/~elman/Papers/fsit.pdf>>. Acesso em: 23/08/2009.

ENDERS, Walter. *Applied Econometric Time Series.* 2nd ed. USA: John Willey & Sons, 2004.

FARAWAY, Julian; CHATFIELD, Chris. *Time Series Forecasting with Neural Networks: A Case Study.* *Applied Statistics*, 47, 1998, p. 231-250. Disponível em: <<http://www.maths.bath.ac.uk/~jjf23/papers/nnts.pdf>>. Acesso em: 02/02/2010.

FAVA, Vera Lúcia. Análise de Séries de Tempo. Cap. 9. In: VASCONCELLOS, Marco Antonio Sandoval; ALVES, Denisard. (Coord.). **Manual de Econometria**. São Paulo: Atlas, 2000.

FERNEDA, Edberto. Redes neurais e sua aplicação em sistemas de recuperação de informação. **Ci. Inf.**, Brasília, v. 35, n. 1, abr. 2006. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0100-19652006000100003&lng=en&nrm=iso>. doi: 10.1590/S0100-19652006000100003. Acesso em: 19/09/2010.

FLORES, João Henrique Ferreira. **Comparação de modelos MLP/RNA e modelos Box-Jenkins em séries temporais não lineares**. Rio Grande do Sul, 2009. Dissertação (Mestrado em Engenharia de Produção – Programa de Pós-Graduação em Engenharia de Produção (PPGEP) na Escola de Engenharia da Universidade Federal do Rio Grande do Sul (UFRGS).

FOUTO, Nuno Manoel Martins Dias. **Mercado de consumo brasileiro: evolução e determinantes do volume de vendas. Uma análise do período pós-real**. São Paulo, 2008. Tese (Doutorado em Administração) – Faculdade de Economia, Administração e Contabilidade da Universidade de São Paulo.

GARCIA, Cesar Alexandre Fernandes Mendes. **Análise de séries temporais com recurso a técnicas de base de dados**. Porto, out. 2000. Dissertação (Mestrado em Inteligência Artificial e Computação) – Faculdade de engenharia da Universidade do Porto.

GARCIA, Ricardo Basso. **Conhecimento sintático-semântico e processamento de sentenças em rede neural recorrente simples**. Campinas, jul. 2006. Dissertação (Mestrado em Linguística) – Programa de Pós-Graduação do Departamento de Linguística do Instituto de Estudos da Linguagem.

GATELY, Edward J. *Neural networks for financial forecasting*. USA: John Wiley & Sons, 1996.

GHIASSI, M. *et al.* A dynamic artificial neural network model for forecasting time series events. **International Journal of Forecasting**, v. 21, n. Elsevier B.V., p. 341-362, Abr./Jun. 2005.

GOEDEL, Alessandro *et al.* Uma abordagem neural para estimação de conjugado em motores de indução. **Sba Controle & Automação**, Campinas, v. 17, n. 3, set. 2006. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0103-17592006000300010&lng=en&nrm=iso>. doi: 10.1590/S0103-17592006000300010. Acesso em: 19/09/2010.

GOMES, Daniel Takata. **Redes neurais recorrentes para previsão de séries temporais de memórias curta e longa**. Campinas, 2005. Dissertação (Mestrado em Estatística) – Instituto de Matemática, Estatística e Computação Científica da Universidade Estadual de Campinas.

GRANJER, Clive William John; NEWBOLD, Paul. *Forecasting economic series*. New York: Academic Press, 1977.

GRANJER, Clive William John; TERÄSVIRTA, Timo. *Modelling nonlinear economic relationships*. Oxford: Oxford University Press, 1993.

GUJARATI, Damodar N. *Econometria Básica*. 4. ed. São Paulo: Campus, 2006.

_____. *Econometria Básica*. 3. ed. São Paulo: Pearson Education do Brasil, 2000.

GUTEMBERG, H. *et al.* **Modelos de previsão para séries de produção e preços: metodologia bayesiana e Box-Jenkins para séries temporais**. IPEA – Instituto de Planejamento Econômico e Social, textos para discussão interna, n. 92, Rio de Janeiro, 1986.

HAGNER, David G. *et al.* *Comparison of recurrent neural networks for trajectory generation*. In: MEDSKER, Larry; JAIN, Lakhmi C. *Recurrent neural networks: design and applications*. Boca Raton: CRC Press, 2001. Disponível em: <ftp://58.192.112.18/Pub2/EBooks/Books_from_EngnetBase/pdf/7181/CH-10.pdf>. Acesso em: 29/10/2010.

HAMILTON, James D. *Time Series Analysis*. New Jersey: Princeton University Press, 1994.

HAYKIN, Simon. **Redes neurais: princípios e prática**. 2. ed. Porto Alegre: Bookman Companhia, 2001.

IBGE / PMC – **Explicação da variável e da pesquisa**. Disponível em: <<http://www.ibge.gov.br/home/estatistica/indicadores/comercio/pmc/default.shtm>>. Acesso em: 23/03/2010.

KIM, Hyun-jung; SHIN, Kyung-shik. *A hybrid approach based on neural networks and genetic algorithms for detecting temporal patterns in stock markets*. **Applied Soft Computing**, 7, 2007, p. 569-576.

KITAHARA, José Renato. **Responsabilidade social e desempenho financeiro das empresas: um estudo empírico utilizando o balanço social padrão ibase**. São Paulo, 2007. Dissertação (Mestrado em Administração) – Faculdade de Economia, Administração e Contabilidade da Universidade de São Paulo.

KLINE, Douglas M. *Methods for multi-step time series forecasting with neural networks*. Cap. 12. In: ZHANG, G. Peter. **Neural networks in business forecasting**. USA: Idea Group, 2004.

KOSKELA, Timo *et al.* *Times series prediction with multilayer perceptron, FIR and Elman neural networks*. In: *Proceedings of the World Congress on Neural Networks*, INNS Press, 1996, p. 491-496.

KOVÁCS, Zsolt László. **Redes neurais artificiais: fundamentos e aplicações**. 3. ed. revista. São Paulo: Livraria da Física, 1996.

KREMER, Stefan C. *Lessons from language learning*. Cap. 8. 2001. In: MEDSKER, Larry; JAIN, Lakhmi C. *Recurrent neural networks: design and applications*. Boca Raton: CRC Press, 2001. Disponível em: ftp://58.192.112.18/Pub2/EBooks/Books_from_EngnetBase/pdf/7181/CH-08.pdf. Acesso em: 11/2009.

KRIESEL, David. *A brief introduction to neural networks*. 2005. Disponível em: www.dkriesel.com/en/science/neural_networks. Acesso em: 14/09/2009.

LAKATOS, Eva Maria; MARCONI, Marina de Andrade. **Fundamentos de Metodologia Científica**. 5. ed. São Paulo: Atlas, 2003.

LAW, Robert; PINE, Ray. *Tourism demand forecasting for the tourism industry: a neural network approach*. Cap. 6. In: ZHANG, G. Peter. *Neural networks in business forecasting*. USA: Idea Group, 2004.

LEONE FILHO, Marcos de Almeida. **Previsão de carga de curto prazo usando ensembles de previsores selecionados e evoluídos por algoritmos genéticos**. Campinas, 2006. Dissertação (Mestrado em Engenharia Elétrica) – Faculdade de Engenharia Elétrica e Computação da Universidade de Campinas.

LI, Leong-Kwan *et al.* *Forecasting short-term exchange rates: a recurrent neural network approach*. Cap. 10. In: ZHANG, G. Peter. *Neural networks in business forecasting*. USA: Idea Group, 2004.

LNCC. Laboratório Nacional de Computação Científica. Petrópolis, Rio de Janeiro. Disponível em: http://www.lncc.br/~labinfo/tutorialRN/frn4_backpropagation.htm. Acesso em: 20/09/2010.

MACKINNON, James. *Critical values for cointegration tests*. In: ENGLE, R. F.; GRANGER, C. W. J. (Ed.). *Long-run economic relationships: readings in cointegration*, Oxford: Oxford University Press, 1991.

MAKRIDAKIS, Spyros; WHEELWRIGHT, Steven C. *The handbook of forecasting*. 2nd ed. USA: John Wiley & Sons, 1987.

MAKRIDAKIS, Spyros; WHEELWRIGHT, Steven C. *Forecasting methods for management*. 4th ed. USA: John Wiley & Sons, 1985.

_____. *Interactive Forecasting*. 2nd ed. San Francisco: Holden-Day, 1978.

MATOS, Orlando Carneiro de. *Econometria básica: teoria e aplicações*. 3. ed. São Paulo: Atlas, 2000.

MC CULLOCH, Warren Sturgis; PITTS, Walter. *A logical calculus of the ideas immanent in nervous activity*, 1943 *apud* HAYKIN, S. **Redes neurais: princípios e prática**. 2. ed. Porto Alegre: Bookman Companhia, 2001. p. 39.

MC LEOD, A. Ian; LI, Wai Keung. *Diagnostic checking ARMA time series models using squared-residuals autocorrelations*. **Journal of Time Series Analysis**, 4, p. 269-273, 1983.

MC NELIS, Paul D. *Neural networks in finance: gaining predictive edge in the market*. USA: Elsevier, 2005.

MEDSKER, Larry; JAIN, Lakhmi C. *Recurrent neural networks: design and applications*. **CRC Press**. 2001. Disponível em: <ftp://58.192.112.18/Pub2/EBooks/Books_from_EngnetBase/pdf/7181/toc.pdf>. Acesso em: 11/2009.

MORE, Anurag; DEO, M. C. *Forecasting wind with neural networks*. *Marine Structures* 16, 2003, p. 35-39.

MORETTIN, Pedro; TOLOI, Clélia M. C. **Previsão de séries temporais**. 2. ed. São Paulo: Atual, 1985.

_____. **Análise de séries temporais**. 2. ed. São Paulo: Edgard Blücher, 2006.

MUELLER, Alessandro. **Uma Aplicação De Redes Neurais Artificiais. na previsão do mercado acionário**. Santa Catarina, 1996. Dissertação (Mestrado em Engenharia de Produção) – Faculdade de Engenharia de Produção da Universidade Federal de Santa Catarina. Disponível em: <<http://www.eps.ufsc.br/disserta96/mueller/index/index.htm>>. Acesso em: 02/01/2010.

MURTEIRA, José Bento Ferreira *et al.* **Análise de Sucessões Cronológicas**. Amadora: McGraw-Hill de Portugal, 1993.

NEUROSOLUTIONS. *What is a Neural Network?* Disponível em: <www.neurosolutions.com/resources/videotour.html>. Acesso em: 12/10/2009.

NGUYEN, Derrick; WIDROW, Bernard. *Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights*. *Proceedings of the International Joint Conference on Neural Networks*, v. 3, 1990, p. 21–26.

PACHECO, Antonio Guilherme Fonseca. **Estudo da influência de variáveis meteorológicas no aparecimento de casos graves de leptospirose em Salvador – BA via modelos de séries temporais**. Ministério da Saúde, Fundação Oswaldo Cruz, Escola Nacional de Saúde Pública, 2001. Disponível em: <<http://teses.iciet.fiocruz.br/pdf/pachecoagfm.pdf>>. Acesso em: 03/06/2010.

PALMER, Alfonso *et al.* *Designing an artificial neural network for forecasting tourism time series*. **Tourism Management**, v. 27, p. 781-790, 2006.

PANKRATZ, Alan. *Forecasting with dynamic regression models*. **Wiley-Interscience Publication**. New York: John Wiley & Sons, 1991.

PASCHOALINO, F. F. *et al.* **Previsão de demanda de energia elétrica no Brasil utilizando redes neurais de Elman**. **SPOLM 2007 RJ**. Disponível em: <http://www.uff.br/decisao/spolm2007_133.pdf>. Acesso em: 08/12/2009.

PINO, F. A. *et al.* Sazonalidade em séries temporais econômicas: um levantamento sobre o estado da arte. **Agricultura em São Paulo**, v. 41, n. 3, p. 103-133, 1994.

PINSKY, Daniel. **O uso do livro eletrônico no ensino superior sob a ótica dos professores universitários**. São Paulo, 2009. Dissertação (Mestrado em Administração) – Faculdade de Economia, Administração e Contabilidade da Universidade de São Paulo.

PRINCIPE, Jose C. *et al.* **Neural and Adaptive Systems: Fundamentals Through Simulations**. New York: John Wiley & Sons, 2000.

REDES NEURAIAS ARTIFICIAIS. Disponível em: <<http://www.google.com>>. Acesso em: 30/06/2010.

SAMARASINGHE, Sandhya. **Neural networks for applied sciences and engineering**. USA: Auerbach, 2007.

SANTOS, Eduardo Alves Portela. **Previsão não-linear da taxa de câmbio real/dólar utilizando redes neurais e sistemas nebulosos**. Florianópolis, mar. 2005. Dissertação (Mestrado em Economia) – Universidade Federal de Santa Catarina.

SANTOS, Eurípedes P. dos; VON ZUBEN, Fernando J. *Efficient second-order learning algorithms for discrete-time recurrent neural networks*. Cap. 3. In: MEDSKER, Larry; JAIN, Lakhmi C. **Recurrent neural networks: design and applications**. Boca Raton: CRC Press, 2001. Disponível em:

<ftp://58.192.112.18/Pub2/EBooks/Books_from_EngnetBase/pdf/7181/CH-03.pdf>. Acesso em: 28/11/2009.

SARTORIS, Alexandre. **Estatística e introdução à econometria**. São Paulo: Saraiva, 2003.

_____. In: VASCONCELLOS, Marco Antonio Sandoval; ALVES, Denisard. (Coord.). **Manual de Econometria**. São Paulo: Atlas, 2000.

SOARES, Ilton G.; CASTELAR, Ivan. **Econometria aplicada com o uso do Eviews**. Fortaleza: UFC/CAEN, 2003.

STAGGE, Peter; SENDHOFF, Bernhard. *An extended Elman net for modeling time series*. **International Conference on Artificial Networks - LNCS**, n. Springer-Verlag, p. 427-432, 1997.

STOIANOV, Ivelin. **Recurrent autoassociative networks: developing distributed representations of structured sequences by autoassociation**. Cap. 9. In: MEDSKER, Larry; JAIN, Lakhmi C. **Recurrent neural networks: design and applications**. Boca Raton: CRC Press, 2001. Disponível em: <ftp://58.192.112.18/Pub2/EBooks/Books_from_EngnetBase/pdf/7181/CH-09.pdf>. Acesso em: 11/2009.

SUNDARESHAN, Malur K. *et al. Training algorithms for recurrent neural nets that eliminate the need for computation of error gradients with application to trajectory production problem*. Cap. 11. In: MEDSKER, Larry; JAIN, Lakhmi C. **Recurrent neural networks: design and applications**. Boca Raton: CRC Press, 2001. Disponível em: <ftp://58.192.112.18/Pub2/EBooks/Books_from_EngnetBase/pdf/7181/CH-11.pdf>. Acesso em: 28/11/2009.

THAWORNWONG, Suraphan; ENKE, David. *Forecasting stock returns with artificial neural networks*. Cap.3. In: ZHANG, G. Peter. **Neural networks in business forecasting**. USA: Idea Group, 2004.

TSAI, Tsung-Hsien *et al Neural network based temporal feature models for short-term railway passenger demand forecasting*. Expert Systems with Applications 36, 2009, p. 3.728-3.736.

TSAY, Ruey S. **Analisis of financial time series**. USA: John Wiley & Sons, 2001.

UNISUL. **Sobre Redes neurais na previsão de séries temporais**. Disponível em: <inf.unisul.br/~ines/workcomp/cd/pdfs/2878.pdf>. Acesso em: 14/09/2009.

VASCONCELLOS, Marco Antonio Sandoval; ALVES, Denisard. (Coord.). **Manual de Econometria**. São Paulo: Atlas, 2000.

VON ZUBEN, Fernando J. Uma caricatura funcional de redes neurais artificiais. *Learning and Nonlinear Models*. **Revista da Sociedade Brasileira de Redes Neurais**, v. 1, n. 2, p. 66-76, 2003.

WALCZAK, Steven. *Forecasting Emerging Market indexes with neural networks*. Cap. 4. In: ZHANG, G. Peter. **Neural networks in business forecasting**. USA: Idea Group, 2004.

WEI, William. W. S. *Time series analysis: univariate and multivariate methods*. 2nd ed. USA: Pearson Education, 2006.

WEST, David; DELLANA, Scott. *Predicting wastewater BOD levels with neural network time series models*. Cap. 5. In: ZHANG, G. Peter. **Neural networks in business forecasting**. [S.l.]: Idea Group, 2004.

WIDROW, B.; HOFF JR, M.E. *Adaptive switching circuits*. IRE WESCON Convention Record, p. 96-104, 1960 *apud* HAYKIN, S. **Redes neurais: princípios e prática**. 2. ed. Porto Alegre: Bookman Companhia, p. 77, 2001.

WOLD, Herman. **A Study in the Analysis of Stationary Time Series**. Stockholm: Almqvist and Wiksell (first edition 1938), 1954.

ZHANG, G. Peter. *A neural network ensemble method with jittered training data for time series forecasting*. **Information Sciences**, 177, 2007, p. 5.329-5.346.

_____. *A combined ARIMA and neural network approach for time series forecasting*. Cap 11. In: _____. **Neural networks in business forecasting**. USA: Idea Group, 2004.

_____. *Business forecasting with artificial neural networks: an overview*. Cap. 1. In: _____. **Neural networks in business forecasting**. USA: Idea Group, 2004.

ZHANG, G. Peter; BERARDI, V. L. *Time series forecasting with neural network ensembles: an application for exchange rates prediction*. **Journal of the Operational Research Society**, v. 52 (6), p. 652-664, 2001.

ZHANG, G. Peter *et al.* *Forecasting with neural networks: the state of art*. **International Journal of Forecasting**, v. 14, p. 35-62, Mar. 1998.

ZHANG, G. Peter; KLINE, Douglas M. *Quarterly time series forecasting with neural networks*. **IEEE transactions on neural networks**, v. 18, n. 6, Nov. 2007.

ZHANG, G. Peter; QI, M. *Neural network forecasting for seasonal and trend time series. European Journal of Operational Research and Information Technology*, 160, 2005, p. 501-514.

ZHAO, Lin *et al.* *The problem of neural networks in business forecasting: an attempt to reproduce the Hill, O'Connor and Remus study. Sprouts: Working Papers on Information Systems*, v. 3, p. 233-243, 2003. ISSN 1535-6078. Disponível em: <<http://sprouts.aisnet.org/3-18>>. Acesso em: 14/10/2009.

ZIEMKE, Tom. *Remembering how to behave: recurrent neural networks for adaptive robot behavior. In: MEDSKER, Larry; JAIN, Lakhmi C. Recurrent neural networks: design and applications.* Boca Raton: CRC Press, 2001.

ZIKMUND, William G. *Exploring marketing research*. 6. ed. Orlando: The Dryden Press, 1997.

APÊNDICES

APÊNDICE 1: TABELA DE DICKY- FULLER

APÊNDICE 2: ALGUNS RESULTADOS ADICIONAIS INTERESSANTES DE TREINAMENTOS COM VALIDAÇÃO CRUZADA.

APÊNDICE 3: CÓDIGO MATLAB PARA TREINAMENTO DA SÉRIE AEREO, SEM PRÉ-PROCESSAMENTO

APÊNDICE 4: CÓDIGO MATLAB PARA PREVISÃO DA SÉRIE AEREO, SEM PRÉ-PROCESSAMENTO

APÊNDICE 5: CÓDIGO MATLAB PARA TREINAMENTO DA SÉRIE ADUBOS, SEM PRÉ-PROCESSAMENTO

APÊNDICE 6: CÓDIGO MATLAB PARA PREVISÃO DA SÉRIE ADUBOS, SEM PRÉ-PROCESSAMENTO

APÊNDICE 7: CÓDIGO MATLAB PARA TREINAMENTO DA SÉRIE FARMA, SEM PRÉ-PROCESSAMENTO

APÊNDICE 8: CÓDIGO MATLAB PARA PREVISÃO DA SÉRIE FARMA, SEM PRÉ-PROCESSAMENTO

APÊNDICE 9: CÓDIGO MATLAB PARA TREINAMENTO DA SÉRIE AEREO, COM PRÉ-PROCESSAMENTO

APÊNDICE 10: CÓDIGO MATLAB PARA PREVISÃO DA SÉRIE AEREO, COM PRÉ-PROCESSAMENTO

APÊNDICE 11: CÓDIGO MATLAB PARA TREINAMENTO DA SÉRIE ADUBOS, COM PRÉ-PROCESSAMENTO

APÊNDICE 12: CÓDIGO MATLAB PARA PREVISÃO DA SÉRIE ADUBOS, COM PRÉ-PROCESSAMENTO

APÊNDICE 13: CÓDIGO MATLAB PARA TREINAMENTO DA SÉRIE FARMA, COM PRÉ-PROCESSAMENTO

APÊNDICE 14: CÓDIGO MATLAB PARA PREVISÃO DA SÉRIE FARMA, COM PRÉ-PROCESSAMENTO

APÊNDICE 15: SÉRIES PADRONIZADAS

APÊNDICE 16: TRANSFORMAÇÃO *MAPMINMAX*

APÊNDICE 1: TABELA DE DICKEY- FULLER

Valores críticos para a estatística do teste ADF

Tamanho da série n	Nível de significância (α)				
	1%	2,5%	5%	10%	50%
a) Sem intercepto e sem tendência					
25	-2,65	-2,26	-1,95	-1,60	-0,47
50	-2,62	-2,25	-1,95	-1,61	-0,49
100	-2,60	-2,24	-1,95	-1,61	-0,50
250	-2,58	-2,24	-1,95	-1,62	-0,50
500	-2,58	-2,23	-1,95	-1,62	-0,50
∞	-2,58	-2,23	-1,95	-1,62	-0,51
b) Com intercepto e sem tendência					
25	-3,75	-3,33	-2,99	-2,64	-1,53
50	-3,59	-3,23	-2,93	-2,60	-1,55
100	-3,50	-3,17	-2,90	-2,59	-1,56
250	-3,45	-3,14	-2,88	-2,58	-1,56
500	-3,44	-3,13	-2,87	-2,87	-1,57
∞	-3,42	-3,12	-2,86	-2,86	-1,57
c) Com intercepto e com tendência					
25	-4,38	-3,95	-3,60	-3,24	-2,14
50	-4,16	-3,80	-3,50	-3,18	-2,16
100	-4,05	-3,73	-3,45	-3,15	-2,17
250	-3,98	-3,69	-3,42	-3,13	-2,18
500	-3,97	-3,67	-3,42	-3,13	-2,18
∞	-3,96	-3,67	-3,41	-3,13	-2,18

FONTE: WEI (2006, p. 593)

APÊNDICE 2: ALGUNS RESULTADOS ADICIONAIS INTERESSANTES DE TREINAMENTOS COM VALIDAÇÃO CRUZADA

Em todos os gráficos:

Curva azul: trajetória do MSE com dados no intervalo de treinamento (aprendizado).

Curva vermelha: idem, MSE com dados no intervalo de validação.

Gráfico 1 – Série AEREO, rede de Elman com 2 neurônios ocultos, treinamento com *early-stopping* após fazer alteração de pesos iniciais durante apenas 21 épocas

Gráfico 2 – Série AEREO, rede de Elman com 5 neurônios ocultos, treinamento interrompido com *early-stopping* após 350 épocas

Gráfico 3 – Série AEREO, rede de Elman com 7 neurônios ocultos, treinamento com *early-stopping* após 205 épocas tendo superado alguns mínimos locais da função de erro MSE de validação

Gráfico 4 – Série AEREO, rede de Elman com 20 neurônios ocultos: neste caso não houve convergência na alteração dos pesos iniciais que se mantiveram intactos até o limite máximo de 1000 épocas

Gráfico 5 – Série AEREO, rede de Elman com 6 neurônios ocultos: neste caso a alteração dos pesos impactou drasticamente a função de erro MSE após 500 épocas aproximadamente

Gráfico 6 – Série FARMA, rede de Elman com 3 neurônios ocultos: instabilidade no treinamento ocorrida em torno de 50 épocas. A rede parece ter desaprendido a partir daquele ponto

Gráfico 7 – Série FARMA, rede de Elman com 8 neurônios ocultos: treinamento interrompido por *early-stopping* após ter superado alguns mínimos locais na função de erro MSE de validação

Gráfico 8 – Série FARMA, rede de Elman com 1 neurônio oculto: treinamento interrompido por *early-stopping* após terem decorrido apenas 30 épocas

APÊNDICE 3: CÓDIGO MATLAB PARA TREINAMENTO DA SÉRIE AEREO, SEM PRÉ-PROCESSAMENTO

```
%-- 20/04/10 11:11 --%
% TREINAMENTO de Rede Elman com Matlab                                DADOS ORIGINAIS
% SÉRIE AEREO: Passageiros transportados via aérea no Brasil (pax-km/mês)
%=====
% Para interromper processamento tecle "CTRL + C"
%=====
clear all; %limpar area de trabalho
%*****
% carregar arquivo de dados originais
load anacpax72
% serie inicial preservada para uso futuro
STreal = ST;
%=====
% especificacao da rede e demais dados solicitados ao usuario:
% h = tamanho do intervalo em que será feita previsão
% v = numero de periodos usados para validacao (25%Valid+75%Aprendiz)
% S1 = numero de neuronios na camada oculta
% emax = maximo de epocas, limitado pelo "early stopping"
% vmax = maximo de vezes em que se permite aumentar erro na validacao
% alfa = taxa de aprendizado
%
h = 12;
v = 15;
S1 = 19 ;
emax = 1000;
vmax = 15;
alfa = 0.025;
%
%*****
% inicia contagem de tempo de processamento
tic;
%*****
% padronizacao entre -1 e 1
[STpad,PS] = mapminmax(ST);
ST = STpad;
%*****
% carrega targets e inputs para treinamento
% Pv = matriz de pontos destinados a validacao da rede (inputs)
% Tv = matriz de pontos destinados a validacao da rede (targets)
% Pa = matriz de pontos destinados ao aprendizado da rede (inputs)
% Ta = matriz de pontos destinados ao aprendizado da rede (targets)
% TR = training record
%*****
Pa = ST(1:(end-h-v-1));
Ta = ST(2:(end-h-v));
Pv = ST(1:(end-h-1));
Tv = ST(2:(end-h));
% entradas e alvos convertidos em sequencias
Paseq = con2seq(Pa);
Taseq = con2seq(Ta);
Pvseq = con2seq(Pv);
Tvseq = con2seq(Tv);
%*****
% criar rede de Elman (1-1-1)com 1 camada de entrada e
% 1 camada oculta de S1 neuronios com função tangente sigmoide (tansig)
% 1 camada de saída com função linear (purelin)
```

```

net = newelm(Pa,Ta,S1);
% algoritmo de ajuste de pesos com gradiente
net.trainFcn = 'traingdm';
% algoritmo de treinamento: BPTT
net.gradientFcn = 'calcbtt';
%=====
% define taxa de aprendizado
net.trainParam.lr = alfa ;
%=====
% treinar rede de epoca em epoca
net.trainParam.epochs = 1;
% enquanto nao atinge maximo de épocas e o erro de validacao não cresce
% por mais de um numero vmax de epocas sucessivas.
epoca = 1;
validfail = 0;
y1(1) = 0;
y2(1) = 0;
while (epoca <= emax) && (validfail < vmax) ;
    [net,TR] = train(net,Paseq,Taseq); % treina rede/aprendizado
    % inibe display da janela de aprendizado
    net.trainParam.showWindow = false;
    Y = sim(net,Pvseq); % simulacao no intervalo de validacao
    Z = seq2con(Y); % conversao de formato sequencial para matriz
    errovalid = Tv - Z{1,1}; % calcula erro de validacao
    msev = mse(errovalid); % erro mse de validacao
    if (epoca > 1) && (msev > msevant),
        validfail = validfail + 1 ; % contador de falhas de validacao
        if (validfail == 1)
            stopesoI = pesoI; % guarda pesos do early stopping
            stopesoL = pesoL;
            stopesoB = pesoB;
        end
    else
        validfail = 0; % zera o contador cada vez que o erro diminui
    end
    pesoI = net.IW;
    pesoL = net.LW;
    pesoB = net.b;
    msea = TR.perf(2); % erro mse de aprendizado esta registrado com TR
    % TR.perf(i) é uma matriz que cresce uma coluna a cada época
    % a primeira coluna é o erro inicial e coluna adicional é o erro
    % no final da época.
    % TR.perf(1) = erro inicial e TR.perf(2)=erro no final da epoca
    y1(epoca) = msea;
    y2(epoca) = msev;
    epoca = epoca + 1 % comandos sem ; para exibir variáveis
    validfail % idem
    msea
    msev
    elapsed_time_parcial = (toc)/60
    % armazena mse do intervalo de validacao para comparar na prox epoca
    msevant = msev;
end
% Se ocorreu early stopping, a rede deve retroagir e recuperar pesos
% de vmax epocas atrás.
if (validfail == vmax)
    net.IW = stopesoI;
    net.LW = stopesoL;
    net.b = stopesoB;
else
end
end

```



```

% grafico de validacao cruzada
t=(1:epoca-1) ;
%plot(t,y1(t),'b',t,y2(t),'r--o','linewidth',2);
plot(t,y2(t),'r--o',t,y1(t),'b','linewidth',2);
title('Validação Cruzada:      Treinamento (-)      Validação (o)');
xlabel('época');
ylabel('mse');
%
% Fim do treinamento da rede
% Lembrar que saída da rede é padronizada

```

APÊNDICE 4: CÓDIGO MATLAB PARA PREVISÃO DA SÉRIE AEREO, SEM PRÉ-PROCESSAMENTO

```
%-- 20/04/10 11:11 --%
% Previsao de Serie Temporal com Rede Elman no Matlab
% Previsão com SERIE AEREO ORIGINAL
%*****
% carregar arquivo de dados com a rede treinada
clear all

                                load RedeANAC_N08_msev0326.mat
%=====
% Previsao ESTÁTICA para o intervalo de teste
%=====
% rede simulada da origem até ultimo ponto de teste
Pf = ST(1:end-1);
Pfseq = con2seq(Pf);
Yf = sim(net,Pfseq);
Zf = seq2con(Yf);
Sf = cell2mat(Zf);
% neste ponto Sf contem toda a serie estim ponto a ponto,por simulacao.
%
% reverter a transformacao mapminmax
Sf = mapminmax('reverse',Sf,PS);
Sfd = Sf;
% plotar as series real e estimada
figure;
% serie original (targets) em azul
plot(STreal,'b','linewidth',1.5);
title('Dados reais (- azul)                                Previsão ESTÁTICA da rede (--
vermelho) ');
grid
hold on;
% ajuste de plotagem ao primeiro ponto perdido no forecast
STgraf = [STreal(1) Sfd];
% estimativas (previsoes) em vermelho
plot(STgraf,'r--','linewidth',1.5);
%hold on;
i=1:h;
%*****
% Qualidade da previsao
Prev(i) = Sfd(end-h+i);
% calculo do erros individuais no intervalo i=1:h
% erro = individual
% sumerroquad = soma dos "erros ao quadrado"
sumerroquad = 0;
% sumprev = soma das previsões ao quadrado
sumprevquad = 0;
% sumactual = soma dos valores conhecidos (ex-post) ao quadrado
sumactualquad = 0;
for i=1:h
    erro(i) = Prev(i)-STreal(end-h+i);
    sumerroquad = erro(i).^2 + sumerroquad;
    sumprevquad = Prev(i).^2 + sumprevquad;
    sumactualquad = STreal(end-h+i).^2 + sumactualquad;
end
%figure
%i=1:h;
%plot(i,erro(i))
% mean squared error = erro quadrático médio = MSE do forecast Estático
```

```

EMSE = sumerroquad/h;
% root mean squared error = RMSE do forecast Estático
ERMSE = sqrt(EMSE);
% ETIC = coef de ineq de Theil do forecast Estático
ETIC = ERMSE / ( sqrt(sumprevquad/h) + sqrt(sumactualquad/h) );
% EBIAS = bias proportion do MSE do forecast Estático
% YBar = média dos valores conhecidos ex-post
Ybar = mean( STreal(end-h+1:end) );
EBIAS = ((mean(Prev) - Ybar).^2) / EMSE;
% EVAR = variance proportion
% SYbar = desvio padrao "viesado" das previsoes
% SY = desvio padrao "viesado" dos valores ex-post
% => usar flag 1 no argumento de std
SYbar = std(Prev,1);
SY = std( STreal(end-h+1:end),1);
EVAR = ((SYbar - SY).^2) / EMSE;
% ECOV = covariance proportion
% ER = correlacao entre as previsoes e os valores ex-post
% Transpor matrizes para usar argumentos corretos na funcao "corr"
ER = corr (Prev', STreal(end-h+1:end)');
ECOV = (2 * (1 - ER)* SYbar * SY ) / EMSE;
% Prop = variavel teste = tem de ser igual a 1.0000 = 100%
Prop = EBIAS + EVAR + ECOV;
%=====
% Previsao DINÂMICA para o intervalo de teste
%=====
%
% reconstituir ST tal como na etapa de treinamento
ST = STpad;
% primeiro ponto previsto dinamicamente = previsão estática
Pfdin = ST(1:(end-h));
Pfseqdin = con2seq(Pfdin);
Yfdin = sim(net,Pfseqdin);
Zfdin = seq2con(Yfdin);
Sfdin = cell2mat(Zfdin);
%
% previsao para demais pontos, calculada recorrentemente
for i=2:h
    Pfdin = [Pfdin Sfdin(end)];
    Pfseqdin = con2seq(Pfdin);
    Yfdin = sim(net,Pfseqdin);
    Zfdin = seq2con(Yfdin);
    Sfdin = cell2mat(Zfdin);
end
% Desfazer a transformacao mapminmax para
% obter a estimativa da série (Sfdin) na unidade da variavel original.
Sfdin = mapminmax('reverse',Sfdin,PS);
% plotar as series real e estimada
figure
% serie original (targets) em azul
plot(STreal,'b','linewidth',1.5);
title('Dados reais (- azul)                                Previsão DINÂMICA da rede
(-- magenta) ');
grid
hold on;
% ajuste de plotagem ao primeiro ponto perdido no forecast
SNgraf = [STreal(1) Sfdin];
% estimativas (previsoes dinamicas) em verde
plot(SNgraf,'m--','linewidth',1.5);
% *****
% Qualidade da previsão

```

```

% previsoes dinamicas
i=1:h;
Prevdin(i) = Sfdin(end-h+i);
% calculo do erros individuais no intervalo i=1:h
% errodin = erro individual dinamico
% sumerroquaddin = soma dos "erros dinamicos ao quadrado"
sumerroquaddin= 0;
% sumprevdin = soma das previsões ao quadrado
sumprevquaddin = 0;
% sumactualdin = soma dos valores conhecidos (ex-post) ao quadrado
sumactualquaddin = 0;
for i=1:h
    errodin = Prevdin(i)-STreal(end-h+i);
    sumerroquaddin = errodin.^2 + sumerroquaddin;
    sumprevquaddin = Prevdin(i).^2 + sumprevquaddin;
    sumactualquaddin = STreal(end-h+i).^2 + sumactualquaddin;
end
% mean squared error = erro quadrático médio = MSE
EMSEdin = sumerroquaddin/h;
% root mean squared error, tem a vantagem de estar na unidade da variável
ERMSEdin = sqrt(EMSEdin);
% ETIC = coef de ineq de Theil do forecast Dinamico
ETICdin = ERMSEdin / ( sqrt(sumprevquaddin/h) + sqrt(sumactualquaddin/h) );
% EBIASdin = bias proportion do MSE do forecast Dinamico
% Ybardin = média dos valores conhecidos ex-post
Ybardin = mean( STreal(end-h+1:end) );
EBIASdin = ((mean(Prevdin) - Ybardin ).^2) / EMSEdin;
% EVARdin = variance proportion
% SYbardin = desvio padrao "viesado" das previsoes
% SYdin = desvio padrao "viesado" dos valores ex-post
% => usar flag 1 no argumento de std
SYbardin = std(Prevdin,1);
SYdin = std( STreal(end-h+1:end),1);
EVARDin = ((SYbardin - SYdin).^2) / EMSEdin;
% ECOVdin = covariance proportion
% ERdin = correlacao entre as previsoes e os valores ex-post
% Transpor matrizes para usar argumentos corretos na funcao "corr"
ERdin = corr (Prevdin', STreal(end-h+1:end)');
ECOVdin = (2 * (1 - ERdin)* SYbardin * SYdin ) / EMSEdin;
% Prop = variavel teste = tem de ser igual a 1.0000 = 100%
Propdin = EBIASdin + EVARDin + ECOVdin;

```

APÊNDICE 5: CÓDIGO MATLAB PARA TREINAMENTO DA SÉRIE ADUBOS, SEM PRÉ-PROCESSAMENTO

```
%-- 20/04/10 11:11 --%
% TREINAMENTO de Rede Elman com Matlab
%
%===== SÉRIE ADUBOS ORIGINAL =====
% Para interromper processamento tecle "CTRL + C"
%=====
clear all; %limpar area de trabalho
%*****
% carregar arquivo de dados originais
load Anda228
% serie inicial preservada para uso futuro
STreal = ST;
%=====
% especificacao da rede e demais dados solicitados ao usuario:
% h = tamanho do intervalo em que será feita previsão
% v = numero de periodos usados para validacao (25%Valid + 75%Aprendiz)
% S1 = numero de neuronios na camada oculta
% emax = maximo de epocas, limitado pelo "early stopping"
% vmax = maximo de vezes em que se permite aumentar erro na validacao
%
h = 12;
v = 54;
S1 = 5;
emax = 1000;
vmax = 15;
%
%*****
% inicia contagem de tempo de processamento
tic;
%*****
% padronizacao entre -1 e 1
[STpad,PS] = mapminmax(ST);
ST = STpad;
%*****
% carrega targets e inputs para treinamento
% Pv = matriz de pontos destinados a validacao da rede (inputs)
% Tv = matriz de pontos destinados a validacao da rede (targets)
% Pa = matriz de pontos destinados ao aprendizado da rede (inputs)
% Ta = matriz de pontos destinados ao aprendizado da rede (targets)
% TR = training record
%*****
Pa = ST(1:(end-h-v-1));
Ta = ST(2:(end-h-v));
Pv = ST(1:(end-h-1));
Tv = ST(2:(end-h));
% entradas e alvos convertidos em sequencias
Paseq = con2seq(Pa);
Taseq = con2seq(Ta);
Pvseq = con2seq(Pv);
Tvseq = con2seq(Tv);
%*****
% criar rede de Elman (1-1-1) com 1 camada de entrada e
% 1 camada oculta de S1 neuronios com função tangente sigmoide (tansig)
% 1 camada de saída com função linear (purelin)
net = newelm(Pa,Ta,S1);
% algoritmo de ajuste de pesos
```

```

net.trainFcn = 'traingdm';
% algoritmo de treinamento: BPTT
net.gradientFcn = 'calcgbt';
%=====
% define taxa de aprendizado
net.trainParam.lr = 0.1;
%=====
% treinar rede de epoca em epoca
net.trainParam.epochs = 1;
% enquanto nao atinge maximo de épocas e o erro de validacao não cresce
% por mais de um numero vmax de epocas sucessivas.
epoca = 1;
validfail = 0;
y1(1) = 0;
y2(1) = 0;
while (epoca <= emax) && (validfail < vmax) ;
    [net,TR] = train(net,Paseq,Taseq); % treina rede/aprendizado
    % inibe display da janela de aprendizado
    net.trainParam.showWindow = false;
    Y = sim(net,Pvseq); % simulacao no intervalo de validacao
    Z = seq2con(Y); % conversao de formato sequencial para matriz
    errovalid = Tv - Z{1,1}; % calcula erro de validacao
    msev = mse(errovalid); % erro mse de validacao
    if (epoca > 1) && (msev > msevant),
        validfail = validfail + 1 ; % contador de falhas de validacao
        if (validfail == 1)
            stopesoI = pesoI; % guarda pesos do early stopping
            stopesoL = pesoL;
            stopesoB = pesoB;
        end
    else
        validfail = 0; % zera o contador cada vez que o erro diminui
    end
    pesoI = net.IW;
    pesoL = net.LW;
    pesoB = net.b;
    msea = TR.perf(2); % erro mse de aprendizado esta registrado com TR
    % TR.perf(i) é uma matriz que cresce uma coluna a cada época
    % a primeira coluna é o erro inicial e coluna adicional é o erro
    % no final da época.
    % TR.perf(1) = erro inicial e TR.perf(2)= erro no final da epoca
    y1(epoca) = msea;
    y2(epoca) = msev;
    epoca = epoca + 1 % deixar sem ; para acompanhar processamento
    validfail % idem
    msea
    elapsed_time_parcial = (toc)/60
    % armazena mse do intervalo de validacao para comparar na prox epoca
    msevant = msev;
end
% Se ocorreu early stopping, a rede deve retroagir e recuperar pesos
% de vmax epocas atrás.
if (validfail == vmax)
    net.IW = stopesoI;
    net.LW = stopesoL;
    net.b = stopesoB;
else
end
% grafico de validacao cruzada
t=(1:epoca-1) ;
%plot(t,y1(t),'b',t,y2(t),'r--o','linewidth',2);

```

```
plot(t,y2(t),'r--o',t,y1(t),'b','linewidth',2);
title('Validação Cruzada:      Treinamento (-)      Validação (o)');
xlabel('época');
ylabel('mse');
%
% Fim do treinamento da rede
% lembrar que dados de saída são padronizados
```

APÊNDICE 6: CÓDIGO MATLAB PARA PREVISÃO DA SÉRIE ADUBOS, SEM PRÉ-PROCESSAMENTO

```
%-- 20/04/10 11:11 --%
% Previsao de Serie Temporal com Rede Elman no Matlab
% Previsão com SÉRIE ADUBOS ORIGINAL
%*****
% carregar arquivo de dados com a rede treinada
clear all

                                load RedeAnda_N13_msev0173.mat
%=====
% Previsao ESTÁTICA para o intervalo de teste
%=====
% rede simulada da origem até ultimo ponto de teste
Pf = ST(1:end-1);
Pfseq = con2seq(Pf);
Yf = sim(net,Pfseq);
Zf = seq2con(Yf);
Sf = cell2mat(Zf);
% Sf contem toda a serie estimada ponto a ponto,por simulacao.
% com intervalo de entrada equivalente a toda ST = ST(1) até ST(end)
% Quando plotarmos os resultados usaremos a serie real STreal
% reverter a transformacao mapminmax
Sf = mapminmax('reverse',Sf,PS);
Sfd = Sf;
% plotar as series real e estimada
figure;
% serie original (targets) em azul
plot(STreal,'b','linewidth',1.5);
title('Dados reais (- azul) Previsão ESTÁTICA da rede (--
vermelho) ');
grid
hold on;
% estimativas (previsoes) em vermelho
STgraf = [STreal(1) Sfd];
plot(STgraf,'r--','linewidth',1.5);
%hold on;
i=1:h;
Prev(i) = Sfd(end-h+i);
% calculo do erros individuais no intervalo i=1:h
% erro = individual
% sumerroquad = soma dos "erros ao quadrado"
sumerroquad = 0;
% sumprev = soma das previsões ao quadrado
sumprevquad = 0;
% sumactual = soma dos valores conhecidos (ex-post) ao quadrado
sumactualquad = 0;
for i=1:h
    erro(i) = Prev(i)-STreal(end-h+i);
    sumerroquad = erro(i).^2 + sumerroquad;
    sumprevquad = Prev(i).^2 + sumprevquad;
    sumactualquad = STreal(end-h+i).^2 + sumactualquad;
end
%figure
%i=1:h;
%plot(i,erro(i))
% mean squared error = erro quadrático médio = MSE do forecast Estático
EMSE = sumerroquad/h;
% root mean squared error = RMSE do forecast Estático
```



```

ERMSE = sqrt(EMSE);
% ETIC = coef de ineq de Theil do forecast Estático
ETIC = ERMSE / ( sqrt(sumprevquad/h) + sqrt(sumactualquad/h) );
% EBIAS = bias proportion do MSE do forecast Estático
% YBar = média dos valores conhecidos reais
Ybar = mean( STreal(end-h+1:end) );
EBIAS = ((mean(Prev) - Ybar).^2) / EMSE;
% EVAR = variance proportion
% SYbar = desvio padrao "viesado" das previsoes
% SY = desvio padrao "viesado" dos valores reais
% => usar flag 1 no argumento de std
SYbar = std(Prev,1);
SY = std( STreal(end-h+1:end),1);
EVAR = ((SYbar - SY).^2) / EMSE;
% ECOV = covariance proportion
% ER = correlacao entre as previsoes e os valores reais
% Transpor matrizes para usar argumentos corretos na funcao "corr"
ER = corr (Prev', STreal(end-h+1:end)');
ECOV = (2 * (1 - ER)* SYbar * SY ) / EMSE;
% Prop = variavel teste = tem de ser igual a 1.0000 = 100%
Prop = EBIAS + EVAR + ECOV;
%=====
% Previsao DINÂMICA para o intervalo de teste
%=====
%
% reconstituir ST tal como na etapa de treinamento
ST = STpad;
% primeiro ponto previsto dinamicamente = previsão estática
Pfdin = ST(1:(end-h));
Pfseqdin = con2seq(Pfdin);
Yfdin = sim(net,Pfseqdin);
Zfdin = seq2con(Yfdin);
Sfdin = cell2mat(Zfdin);
%
% previsao para demais pontos, calculada recorrentemente
for i=2:h
    Pfdin = [Pfdin Sfdin(end)];
    Pfseqdin = con2seq(Pfdin);
    Yfdin = sim(net,Pfseqdin);
    Zfdin = seq2con(Yfdin);
    Sfdin = cell2mat(Zfdin);
end
% Desfazer a transformacao mapminmax
% para obter a estimativa da série (Sfdin) na unidade da variavel original.
Sfdin = mapminmax('reverse',Sfdin,PS);
% plotar as series real e estimada
figure
% serie original (targets) em azul
plot(STreal,'b','linewidth',1.5);
title('Dados reais (- azul)                               Previsão DINÂMICA da rede
(-- magenta) ');
grid
hold on;
% ajuste de plotagem ao primeiro ponto perdido no forecast
SNgraf = [STreal(1) Sfdin];
% estimativas (previsoes dinamicas) em verde
plot(SNgraf,'m--','linewidth',1.5);
% previsoes dinamicas
i=1:h;
Prevdin(i) = Sfdin(end-h+i);
% calculo do erros individuais no intervalo i=1:h

```

```

% errodin = erro individual dinamico
% sumerroquaddin = soma dos "erros dinamicos ao quadrado"
sumerroquaddin= 0;
% sumprevdin = soma das previsões ao quadrado
sumprevquaddin = 0;
% sumactualdin = soma dos valores conhecidos reais ao quadrado
sumactualquaddin = 0;
for i=1:h
    errodin = Prevdin(i)-STreal(end-h+i);
    sumerroquaddin = errodin.^2 + sumerroquaddin;
    sumprevquaddin = Prevdin(i).^2 + sumprevquaddin;
    sumactualquaddin = STreal(end-h+i).^2 + sumactualquaddin;
end
% mean squared error = erro quadrático médio = MSE
EMSEdin = sumerroquaddin/h;
% root mean squared error, tem a vantagem de estar na unidade da variável
ERMSEdin = sqrt(EMSEdin);
% ETIC = coef de ineq de Theil do forecast Dinamico
ETICdin = ERMSEdin / ( sqrt(sumprevquaddin/h) + sqrt(sumactualquaddin/h) );
% EBIASdin = bias proportion do MSE do forecast Dinamico
% Ybardin = média dos valores conhecidos reais
Ybardin = mean( STreal(end-h+1:end) );
EBIASdin = ((mean(Prevdin) - Ybardin ).^2) / EMSEdin;
% EVARdin = variance proportion
% SYbardin = desvio padrao "viesado" das previsoes
% SYdin = desvio padrao "viesado" dos valores reais
% => usar flag 1 no argumento de std
SYbardin = std(Prevdin,1);
SYdin = std( STreal(end-h+1:end),1);
EVARdin = ((SYbardin - SYdin).^2) / EMSEdin;
% ECOVdin = covariance proportion
% ERdin = correlacao entre as previsoes e os valores reais
% Transpor matrizes para usar argumentos corretos na funcao "corr"
ERdin = corr (Prevdin', STreal(end-h+1:end)');
ECOVdin = (2 * (1 - ERdin)* SYbardin * SYdin ) / EMSEdin;
% Prop = variavel teste = tem de ser igual a 1.0000 = 100%
Propdin = EBIASdin + EVARdin + ECOVdin;

```

APÊNDICE 7: CÓDIGO MATLAB PARA TREINAMENTO DA SÉRIE FARMA, SEM PRÉ-PROCESSAMENTO

```
%-- 20/04/10 11:11 --%
% TREINAMENTO de Rede Elman com Matlab                                DADOS ORIGINAIS
% SÉRIE FARMA
%=====
% Para interromper processamento tecle "CTRL + C"
%=====
clear all; %limpar area de trabalho
%*****
% carregar arquivo de dados originais
load farma84
% serie inicial preservada para uso futuro
STreal = ST;
%=====
% especificacao da rede e demais dados solicitados ao usuario:
    % h = tamanho do intervalo em que será feita previsão
    % v = numero de periodos usados para validacao (25%Valid+75%Aprendiz)
    % S1 = numero de neuronios na camada oculta
    % emax = maximo de epocas, limitado pelo "early stopping"
    % vmax = maximo de vezes em que se permite aumentar erro na validacao
    % alfa = taxa de aprendizado
%
h = 12;
    v = 18;
        S1 = 7;
            emax = 1000;
                vmax = 15;
                    alfa = 0.1;
%
%*****
% inicia contagem de tempo de processamento
tic;
%*****
% padronizacao entre -1 e 1
[STpad,PS] = mapminmax(ST);
ST = STpad;
%*****
% carrega targets e inputs para treinamento
% Pv = matriz de pontos destinados a validacao da rede (inputs)
% Tv = matriz de pontos destinados a validacao da rede (targets)
% Pa = matriz de pontos destinados ao aprendizado da rede (inputs)
% Ta = matriz de pontos destinados ao aprendizado da rede (targets)
% TR = training record
%*****
Pa = ST(1:(end-h-v-1));
Ta = ST(2:(end-h-v));
Pv = ST(1:(end-h-1));
Tv = ST(2:(end-h));
% entradas e alvos convertidos em sequencias
Paseq = con2seq(Pa);
Taseq = con2seq(Ta);
Pvseq = con2seq(Pv);
Tvseq = con2seq(Tv);
%*****
% criar rede de Elman (1-1-1)com 1 camada de entrada e
% 1 camada oculta de S1 neuronios com funcao tangente sigmoide (tansig)
% 1 camada de saída com função linear (purelin)
```

```

net = newelm(Pa,Ta,S1);
% algoritmo de ajuste de pesos
net.trainFcn = 'traingdm';
% algoritmo de treinamento: BPTT
net.gradientFcn = 'calcbtt';
%=====
% define taxa de aprendizado
net.trainParam.lr = alfa ;
%=====
% treinar rede de epoca em epoca
net.trainParam.epochs = 1;
% enquanto nao atinge maximo de épocas e o erro de validacao não cresce
% por mais de um numero vmax de epocas sucessivas.
epoca = 1;
validfail = 0;
y1(1) = 0;
y2(1) = 0;
while (epoca <= emax) && (validfail < vmax) ;
    [net,TR] = train(net,Paseq,Taseq); % treina rede/aprendizado
    % inibe display da janela de aprendizado
    net.trainParam.showWindow = false;
    Y = sim(net,Pvseq); % simulacao no intervalo de validacao
    Z = seq2con(Y); % conversao de formato sequencial para matriz
    errovalid = Tv - Z{1,1}; % calcula erro de validacao
    msev = mse(errovalid); % erro mse de validacao
    if (epoca > 1) && (msev > msevant),
        validfail = validfail + 1 ; % contador de falhas de validacao
        if (validfail == 1)
            stopesoI = pesoI; % guarda pesos do early stopping
            stopesoL = pesoL;
            stopesoB = pesoB;
        end
    else
        validfail = 0; % zera o contador cada vez que o erro diminui
    end
    pesoI = net.IW;
    pesoL = net.LW;
    pesoB = net.b;
    msea = TR.perf(2); % erro mse de aprendizado esta registrado com TR
    % TR.perf(i) é uma matriz que cresce uma coluna a cada época
    % a primeira coluna é o erro inicial e coluna adicional é o erro
    % no final da época.
    % TR.perf(1) = erro inicial e TR.perf(2)=erro no final da epoca
    y1(epoca) = msea;
    y2(epoca) = msev;
    epoca = epoca + 1 % comando sem ; para acompanhar processamento
    validfail % idem
    msea
    elapsed_time_parcial = (toc)/60
    % armazena mse do intervalo de validacao para comparar na prox epoca
    msevant = msev;
end
% Se ocorreu early stopping, a rede deve retroagir e recuperar pesos
% de vmax epocas atrás.
if (validfail == vmax)
    net.IW = stopesoI;
    net.LW = stopesoL;
    net.b = stopesoB;
else
end
% grafico de validacao cruzada

```

```

t=(1:epoca-1) ;
%plot(t,y1(t),'b',t,y2(t),'r--o','linewidth',2);
plot(t,y2(t),'r--o',t,y1(t),'b','linewidth',2);
title('Validação Cruzada:      Treinamento (-)      Validação (o)');
xlabel('época');
ylabel('mse');
%
% Fim do treinamento da rede
% Lembrar que saída da rede é padronizada

```

APÊNDICE 8: CÓDIGO MATLAB PARA PREVISÃO DA SÉRIE FARMA, SEM PRÉ-PROCESSAMENTO

```
%-- 20/04/10 11:11 --%
% Previsao de Serie Temporal com Rede Elman no Matlab
% Previsão com SERIE FARMA ORIGINAL
%*****
% carregar arquivo de dados com a rede treinada
clear all

                                load RedeFarma_N08_msev0223.mat
%=====
% Previsao ESTÁTICA para o intervalo de teste
%=====
% rede simulada da origem até ultimo ponto de teste
Pf = ST(1:end-1);
Pfseq = con2seq(Pf);
Yf = sim(net,Pfseq);
Zf = seq2con(Yf);
Sf = cell2mat(Zf);
% Sf contem toda a serie estimada ponto a ponto, por simulacao.
% % reverter a transformacao mapminmax
Sf = mapminmax('reverse',Sf,PS);
Sfd = Sf;
% plotar as series real e estimada
figure;
% serie original (targets) em azul
plot(STreal,'b','linewidth',1.5);
title('Dados reais (- azul)                                Previsão ESTÁTICA da rede (--
vermelho) ');
grid
hold on;
% ajuste de plotagem ao primeiro ponto perdido no forecast
STgraf = [STreal(1) Sfd];
% estimativas (previsoes) em vermelho
plot(STgraf,'r--','linewidth',1.5);
%hold on;
i=1:h;
%*****
% Qualidade da previsao
Prev(i) = Sfd(end-h+i);
% calculo do erros individuais no intervalo i=1:h
% erro = individual
% sumerroquad = soma dos "erros ao quadrado"
sumerroquad = 0;
% sumprev = soma das previsões ao quadrado
sumprevquad = 0;
% sumactual = soma dos valores conhecidos reais, ao quadrado
sumactualquad = 0;
for i=1:h
    erro(i) = Prev(i)-STreal(end-h+i);
    sumerroquad = erro(i).^2 + sumerroquad;
    sumprevquad = Prev(i).^2 + sumprevquad;
    sumactualquad = STreal(end-h+i).^2 + sumactualquad;
end
%figure
%i=1:h;
%plot(i,erro(i))
% mean squared error = erro quadrático médio = MSE do forecast Estático
EMSE = sumerroquad/h;
```

```

% root mean squared error = RMSE do forecast Estático
ERMSE = sqrt(EMSE);
% ETIC = coef de ineq de Theil do forecast Estático
ETIC = ERMSE / ( sqrt(sumprevquad/h) + sqrt(sumactualquad/h) );
% EBIAS = bias proportion do MSE do forecast Estático
% YBar = média dos valores conhecidos ex-post
Ybar = mean( STreal(end-h+1:end) );
EBIAS = ((mean(Prev) - Ybar).^2) / EMSE;
% EVAR = variance proportion
% SYbar = desvio padrao "viesado" das previsoes
% SY = desvio padrao "viesado" dos valores reais
% => usar flag 1 no argumento de std
SYbar = std(Prev,1);
SY = std( STreal(end-h+1:end),1);
EVAR = ((SYbar - SY).^2) / EMSE;
% ECOV = covariance proportion
% ER = correlacao entre as previsoes e os valores reais
% Transpor matrizes para usar argumentos corretos na funcao "corr"
ER = corr (Prev', STreal(end-h+1:end)');
ECOV = (2 * (1 - ER)* SYbar * SY ) / EMSE;
% Prop = variavel teste = tem de ser igual a 1.0000 = 100%
Prop = EBIAS + EVAR + ECOV;
%=====
% Previsao DINÂMICA para o intervalo de teste
%=====
%
% reconstituir ST tal como na etapa de treinamento
ST = STpad;
% primeiro ponto previsto dinamicamente = previsão estática
Pfdin = ST(1:(end-h));
Pfseqdin = con2seq(Pfdin);
Yfdin = sim(net,Pfseqdin);
Zfdin = seq2con(Yfdin);
Sfdin = cell2mat(Zfdin);
%
% previsao para demais pontos, calculada recorrentemente
for i=2:h
    Pfdin = [Pfdin Sfdin(end)];
    Pfseqdin = con2seq(Pfdin);
    Yfdin = sim(net,Pfseqdin);
    Zfdin = seq2con(Yfdin);
    Sfdin = cell2mat(Zfdin);
end
% Desfazer a transformacao mapminmax
% para obter a estimativa da série (Sfdin) na unidade da variavel original.
Sfdin = mapminmax('reverse',Sfdin,PS);
% plotar as series real e estimada
figure
% serie original (targets) em azul
plot(STreal,'b','linewidth',1.5);
title('Dados reais (- azul)                               Previsão DINÂMICA da rede
(-- magenta) ');
grid
hold on;
% ajuste de plotagem ao primeiro ponto perdido no forecast
SNgraf = [STreal(1) Sfdin];
% estimativas (previsoes dinamicas) em verde
plot(SNgraf,'m--','linewidth',1.5);
% *****
% Qualidade da previsão
% previsoes dinamicas

```

```

i=1:h;
Prevdin(i) = Sfdin(end-h+i);
% calculo do erros individuais no intervalo i=1:h
% errodin = erro individual dinamico
% sumerroquaddin = soma dos "erros dinamicos ao quadrado"
sumerroquaddin= 0;
% sumprevdin = soma das previsões ao quadrado
sumprevquaddin = 0;
% sumactualdin = soma dos valores conhecidos reais, ao quadrado
sumactualquaddin = 0;
for i=1:h
    errodin = Prevdin(i)-STreal(end-h+i);
    sumerroquaddin = errodin.^2 + sumerroquaddin;
    sumprevquaddin = Prevdin(i).^2 + sumprevquaddin;
    sumactualquaddin = STreal(end-h+i).^2 + sumactualquaddin;
end
% mean squared error = erro quadrático médio = MSE
EMSEdin = sumerroquaddin/h;
% root mean squared error, tem a vantagem de estar na unidade da variável
ERMSEdin = sqrt(EMSEdin);
% ETIC = coef de ineq de Theil do forecast Dinamico
ETICdin = ERMSEdin / ( sqrt(sumprevquaddin/h) + sqrt(sumactualquaddin/h) );
% EBIASdin = bias proportion do MSE do forecast Dinamico
% Ybardin = média dos valores conhecidos reais
Ybardin = mean( STreal(end-h+1:end) );
EBIASdin = ((mean(Prevdin) - Ybardin ).^2) / EMSEdin;
% EVARDin = variance proportion
% SYbardin = desvio padrao "viesado" das previsoes
% SYdin = desvio padrao "viesado" dos valores reais
% => usar flag 1 no argumento de std
SYbardin = std(Prevdin,1);
SYdin = std( STreal(end-h+1:end),1);
EVARDin = ((SYbardin - SYdin).^2) / EMSEdin;
% ECOVDin = covariance proportion
% ERdin = correlacao entre as previsoes e os valores reais
% Transpor matrizes para usar argumentos corretos na funcao "corr"
ERdin = corr (Prevdin', STreal(end-h+1:end)');
ECOVDin = (2 * (1 - ERdin)* SYbardin * SYdin ) / EMSEdin;
% Prop = variavel teste = tem de ser igual a 1.0000 = 100%
Propdin = EBIASdin + EVARDin + ECOVDin;

```


APÊNDICE 9: CÓDIGO MATLAB PARA TREINAMENTO DA SÉRIE AEREO, COM PRÉ-PROCESSAMENTO

```
%-- 20/04/10 11:11 --%
% TREINAMENTO de Rede Elman com Matlab
% SÉRIE AEREO DADOS PRÉ-PROCESSADOS
%=====
% Para interromper processamento tecle "CTRL + C"
%=====
clear all; %limpar area de trabalho
%*****
% carregar arquivo de dados originais com 72 pontos,
load anacpax72
% serie inicial preservada para uso futuro
STreal = ST;
ST01 = ST(1);
ST02 = ST(2);
ST03 = ST(3);
ST04 = ST(4);
ST05 = ST(5);
ST06 = ST(6);
ST07 = ST(7);
ST08 = ST(8);
%=====
% especificacao da rede e demais dados solicitados ao usuario:
% h = tamanho do intervalo em que será feita previsão ex-post
% v = numero de periodos usados para validacao (25%Valid+75%Aprendiz)
% S1 = numero de neuronios na camada oculta
% emax = maximo de epocas, limitado pelo "early stopping"
% vmax = maximo de vezes em que se permite aumentar erro na validacao
% alfa = taxa de aprendizado
%
h = 12;
v = 15;
S1 = 13 ;
emax = 1000;
vmax = 15;
alfa = 0.1;
%
%*****
% inicia contagem de tempo de processamento
tic;
%*****
% PRÉ-PROCESSAMENTO
% logaritmo
ST = log(ST);
% primeira diferença regular (serie encolhe 1 ponto à esquerda)
ST = diff(ST);
% 1a diferença sazonal de 6 meses(a serie encolhe em mais 6 pontos)
i=7:71;
%STdsaz(i-12)=ST(i)-ST(i-12);
%STdsaz(i-6)=ST(i)-ST(i-6);
ST = STdsaz;
% padronizacao entre -1 e 1
[STpad,PS] = mapminmax(ST);
ST = STpad;
%*****
% carrega targets e inputs para treinamento
% Pv = matriz de pontos destinados a validacao da rede (inputs)
```

```

% Tv = matriz de pontos destinados a validacao da rede (targets)
% Pa = matriz de pontos destinados ao aprendizado da rede (inputs)
% Ta = matriz de pontos destinados ao aprendizado da rede (targets)
% TR = training record
%*****
Pa = ST(1:(end-h-v-1));
Ta = ST(2:(end-h-v));
Pv = ST(1:(end-h-1));
Tv = ST(2:(end-h));
% entradas e alvos convertidos em sequencias
Paseq = con2seq(Pa);
Taseq = con2seq(Ta);
Pvseq = con2seq(Pv);
Tvseq = con2seq(Tv);
%*****
% cria rede de Elman (1-1-1) com 1 camada de entrada e
% 1 camada oculta de S1 neuronios com função tangente sigmoide (tansig)
% 1 camada de saída com função linear (purelin)
net = newelm(Pa,Ta,S1);
% algoritmo de ajuste de pesos
net.trainFcn = 'traingdm';
% algoritmo de treinamento: BPTT
net.gradientFcn = 'calcgbtt';
%=====
% define taxa de aprendizado
net.trainParam.lr = alfa;
%=====
% treina rede de epoca em epoca
net.trainParam.epochs = 1;
% enquanto nao atinge maximo de épocas e o erro de validacao não cresce
% por mais de um numero vmax de epocas sucessivas.
epoca = 1;
validfail = 0;
y1(1) = 0;
y2(1) = 0;
while (epoca <= emax) && (validfail < vmax) ;
    [net,TR] = train(net,Paseq,Taseq); % treina rede/aprendizado
    % inibe display da janela de aprendizado
    net.trainParam.showWindow = false;
    Y = sim(net,Pvseq); % simulacao no intervalo de validacao
    Z = seq2con(Y); % conversao de formato sequencial para matriz
    errovalid = Tv - Z{1,1}; % calcula erro de validacao
    msev = mse(errovalid); % erro mse de validacao
    if (epoca > 1) && (msev > msevant),
        validfail = validfail + 1 ; % contador de falhas de validacao
        if (validfail == 1)
            stopesoI = pesoI; % guarda pesos do early stopping
            stopesoL = pesoL;
            stopesoB = pesoB;
        end
    else
        validfail = 0; % zera o contador cada vez que o erro diminui
    end
    pesoI = net.IW;
    pesoL = net.LW;
    pesoB = net.b;
    msea = TR.perf(2); % erro mse de aprendizado esta registrado com TR
    % TR.perf(i) é uma matriz que cresce uma coluna a cada época
    % a primeira coluna é o erro inicial e coluna adicional é o erro
    % no final da época. Como estamos treinando de 1 em 1 epoca, teremos
    % TR.perf(1) = erro inicial e TR.perf(2)=erro no final da (unica) epoca

```

```

    y1(epoca) = msea;
    y2(epoca) = msev;
    epoca = epoca + 1 % deixar sem ; para acompanhar processamento
    validfail % idem
    msea
    msev
    elapsed_time_parcial = (toc)/60
    % armazena mse do intervalo de validacao para comparar na prox epoca
    msevant = msev;
end
% Se ocorreu early stopping, a rede deve retroagir e recuperar pesos
% de vmax epocas atrás.
if (validfail == vmax)
    net.IW = stopesoI;
    net.LW = stopesoL;
    net.b = stopesoB;
else
end
% grafico de validacao cruzada
t=(1:epoca-1) ;
%plot(t,y1(t),'b',t,y2(t),'r--o','linewidth',2);
plot(t,y2(t),'r--o',t,y1(t),'b','linewidth',2);
title('Validação Cruzada:      Treinamento (-)      Validação (o)');
xlabel('época');
ylabel('mse');
%
% Fim do treinamento da rede, lembrar que a saida da rede é transformada
% pelos pré-processamentos

```

APÊNDICE 10: CÓDIGO MATLAB PARA PREVISÃO DA SÉRIE AEREO, COM PRÉ-PROCESSAMENTO

```
%-- 20/04/10 11:11 --%
% Previsao de Serie Temporal com Rede Elman no Matlab
% Previsão para SÉRIE AEREO COM PRÉ-PROCESSAMENTO
%*****
% carregar arquivo de dados com a rede treinada
clear all

load N10_MSEV_0585.mat
% informar tamanho da serie incluindo intervalo ex-post
T = 72;
%=====
% Previsao ESTÁTICA para o intervalo de teste
%=====
% rede simulada com entrada da origem até o penultimo ponto
Pf = ST(1:end-1);
Pfseq = con2seq(Pf);
Yf = sim(net,Pfseq);
Zf = seq2con(Yf);
Sf = cell2mat(Zf);
% A saída padronizada (Sf) é a previsao estática das Diferenças Sazonais.
% PÓS-PROCESSAMENTO
% reverter a transformacao mapminmax
Sf = mapminmax('reverse',Sf,PS);
F = Sf; % F = forecast da diferença sazonal D sem padronizacao
% Se D é diferença sazonal
% entao D(LYt,1,6) = LYt - LYt(-1) - [ LYt(-6) - LYt(-7) ]
% reconstitui série com valores iniciais e previsoes
LYt(1) = log(ST01);
LYt(2) = log(ST02);
LYt(3) = log(ST03);
LYt(4) = log(ST04);
LYt(5) = log(ST05);
LYt(6) = log(ST06);
LYt(7) = log(ST07);
LYt(8) = log(ST08);
% Cada ponto é estimado a partir dos anteriores conhecidos
Aux = log(STreal);
for i = (1:T-8); % a serie encurtou 8 pontos
LYt(i+8) = F(i) + Aux(i+7)+ Aux(i+2) - Aux(i+1);
end;
% desfazer log
Yt = exp(LYt);
% plotar as series real e estimada
figure;
% serie original (targets) em azul
plot(STreal,'b','linewidth',1.5);
title('Dados reais (- azul) Previsão ESTÁTICA da rede (..
vermelho) ');
grid
hold on;
% estimativas (previsoes) em vermelho
STgraf = Yt;
plot(STgraf,'r--','linewidth',1.5);
%hold on;
%*****
% Qualidade da previsao
%
```

```

i=1:h;
Prev(i) = Yt(end-h+i);
% calculo do erros individuais no intervalo i=1:h
% erro = individual
% sumerroquad = soma dos "erros ao quadrado"
sumerroquad = 0;
% sumprev = soma das previsões ao quadrado
sumprevquad = 0;
% sumactual = soma dos valores reais ao quadrado
sumactualquad = 0;
for i=1:h
    erro(i) = Prev(i)-STreal(end-h+i);
    sumerroquad = erro(i).^2 + sumerroquad;
    sumprevquad = Prev(i).^2 + sumprevquad;
    sumactualquad = STreal(end-h+i).^2 + sumactualquad;
end
%
% mean squared error = erro quadrático médio = MSE do forecast Estático
EMSE = sumerroquad/h;
% root mean squared error = RMSE do forecast Estático
ERMSE = sqrt(EMSE);
% ETIC = coef de ineq de Theil do forecast Estático
ETIC = ERMSE / ( sqrt(sumprevquad/h) + sqrt(sumactualquad/h) );
% EBIAS= bias proportion do MSE do forecast Estático
% YBar = média dos valores conhecidos reais
Ybar = mean( STreal(end-h+1:end) );
EBIAS = ((mean(Prev) - Ybar ).^2) / EMSE;
% EVAR = variance proportion
% SYbar = desvio padrao "viesado" das previsoes
% SY = desvio padrao "viesado" dos valores reais
% => usar flag 1 no argumento de std
SYbar = std(Prev,1);
SY = std( STreal(end-h+1:end),1);
EVAR = ((SYbar - SY).^2) / EMSE;
% ECOV = covariance proportion
% ER = correlacao entre as previsoes e os valores reais
% Transpor matrizes para usar argumentos corretos na funcao "corr"
ER = corr (Prev', STreal(end-h+1:end)');
ECOV = (2 * (1 - ER)* SYbar * SY ) / EMSE;
% Prop = variavel teste = tem de ser igual a 1.0000 = 100%
Prop = EBIAS + EVAR + ECOV;
%
%=====
% Previsao DINÂMICA para o intervalo de teste
%=====
%
% reconstituir ST tal como na etapa de treinamento
ST = STpad;
% primeiro ponto previsto dinamicamente = previsão estática
Pfdin = ST(1:(end-h));
Pfseqdin = con2seq(Pfdin);
Yfdin = sim(net,Pfseqdin);
Zfdin = seq2con(Yfdin);
Sfdin = cell2mat(Zfdin);
%
% A previsao para demais pontos é calculada recorrentemente
for i=2:h
    Pfdin = [Pfdin Sfdin(end)];
    Pfseqdin = con2seq(Pfdin);
    Yfdin = sim(net,Pfseqdin);
    Zfdin = seq2con(Yfdin);

```

```

    Sfdin = cell2mat(Zfdin);
end
% Sfdin contem toda a serie estimada, desde a origem até o fim
% PÓS_PROCESSAMENTO
    % desfazer a transformacao mapminmax
    Sfdin = mapminmax('reverse',Sfdin,PS);
    Fdin = Sfdin;
    %
    LYtdin(1) = log(ST01);
    LYtdin(2) = log(ST02);
    LYtdin(3) = log(ST03);
    LYtdin(4) = log(ST04);
    LYtdin(5) = log(ST05);
    LYtdin(6) = log(ST06);
    LYtdin(7) = log(ST07);
    LYtdin(8) = log(ST08);
    % Cada ponto é estimado a partir dos anteriores conhecidos
    Auxdin = log(STreal);
    for i = 1:(T-8-h+1);
        LYtdin(i+8) = Fdin(i) + Auxdin(i+7)+ Auxdin(i+2) - Auxdin(i+1);
    end;
    %...exceto nos 11 pontos finais que sao calculados recursivamente
    for i = (T-8-h+2):(T-8);
        LYtdin(i+8) = Fdin(i) + LYtdin(i+7)+ LYtdin(i+2) - LYtdin(i+1);
    end;
    % desfazer log
    Ytdin = exp(LYtdin);
%*****
%
%plotar as series real e estimada
figure
% serie original (targets) em azul
plot(STreal,'b','linewidth',1.5);
title('Dados reais (- azul)                                Previsão DINÂMICA da rede
(-- magenta) ');
grid
hold on;
SNgraf = Ytdin;
% estimativas (previsoes dinamicas) em magenta
plot(SNgraf,'m--','linewidth',1.5);
%
%*****
% Qualidade da previsão dinâmica
%
i=1:h;
Prevdin(i) = Ytdin(end-h+i);
% calculo do erros individuais no intervalo i=1:h
% errodin = erro individual dinamico
% sumerroquaddin = soma dos "erros dinamicos ao quadrado"
sumerroquaddin= 0;
% sumprevdin = soma das previsões ao quadrado
sumprevquaddin = 0;
% sumactualdin = soma dos valores conhecidos reais, ao quadrado
sumactualquaddin = 0;
for i=1:h
    errodin = Prevdin(i)-STreal(end-h+i);
    sumerroquaddin = errodin.^2 + sumerroquaddin;
    sumprevquaddin = Prevdin(i).^2 + sumprevquaddin;
    sumactualquaddin = STreal(end-h+i).^2 + sumactualquaddin;
end
% mean squared error = erro quadrático médio = MSE

```

```

EMSEdin = sumerroquaddin/h;
% root mean squared error, tem a vantagem de estar na unidade da variável
ERMSEdin = sqrt(EMSEdin);
% ETIC = coef de ineq de Theil do forecast Dinamico
ETICdin = ERMSEdin / ( sqrt(sumprevquaddin/h)+ sqrt(sumactualquaddin/h) );
% EBIASdin = bias proportion do MSE do forecast Dinamico
% Ybardin = média dos valores conhecidos reais
Ybardin = mean( STreal(end-h+1:end) );
EBIASdin = ((mean(Prevdin) - Ybardin).^2) / EMSEdin;
% EVARDin = variance proportion
% SYbardin = desvio padrao "viesado" das previsoes
% SYdin = desvio padrao "viesado" dos valores reais
% => usar flag 1 no argumento de std
SYbardin = std(Prevdin,1);
SYdin = std( STreal(end-h+1:end),1);
EVARDin = ((SYbardin - SYdin).^2) / EMSEdin;
% ECOVDin = covariance proportion
% ERdin = correlacao entre as previsoes e os valores reais
% Transpor matrizes para usar argumentos corretos na funcao "corr"
ERdin = corr (Prevdin', STreal(end-h+1:end)');
ECOVDin = (2 * (1 - ERdin)* SYbardin * SYdin ) / EMSEdin;
% Prop = variavel teste = tem de ser igual a 1.0000 = 100%
Propdin = EBIASdin + EVARDin + ECOVDin;

```

APÊNDICE 11: CÓDIGO MATLAB PARA TREINAMENTO DA SÉRIE ADUBOS, COM PRÉ-PROCESSAMENTO

```
%-- 20/04/10 11:11 --%
% TREINAMENTO de Rede Elman com Matlab
% SÉRIE ADUBOS
%===== DADOS PRÉ-PROCESSADOS =====
% Para interromper processamento tecle "CTRL + C"
%=====
clear all; %limpar area de trabalho
%*****
% carregar arquivo de dados originais com 228 pontos, incluindo 12 ex-post
load Anda228
% serie inicial preservada para uso futuro
STreal = ST;
ST01 = ST(1);
ST02 = ST(2);
ST03 = ST(3);
ST04 = ST(4);
ST05 = ST(5);
ST06 = ST(6);
ST07 = ST(7);
ST08 = ST(8);
ST09 = ST(9);
ST10 = ST(10);
ST11 = ST(11);
ST12 = ST(12);
ST13 = ST(13);
ST14 = ST(14);
%=====
% especificacao da rede e demais dados solicitados ao usuario:
% h = tamanho do intervalo em que será feita previsão ex-post
% v = numero de periodos usados para validacao (25%Valid + 75%Aprendiz)
% S1 = numero de neuronios na camada oculta
% emax = maximo de epocas, limitado pelo "early stopping"
% vmax = maximo de vezes em que se permite aumentar erro na validacao
% alfa = taxa de aprendizado
%
h = 12;
v = 54;
S1 = 19;
emax = 1000;
vmax = 15;
alfa = 0.025;
%
%*****
% inicia contagem de tempo de processamento
tic;
%*****
% PRÉ-PROCESSAMENTO
% logaritmo
ST = log(ST);
% primeira diferença regular (serie encolhe 1 ponto à esquerda)
ST = diff(ST);
% 1a diferença sazonal de 12 meses(a serie encolhe em mais 12 pontos)
i=13:227;
STdsaz(i-12)=ST(i)-ST(i-12);
ST = STdsaz;
% padronizacao entre -1 e 1
```



```

[STpad,PS] = mapminmax(ST);
    ST = STpad;
%*****
% carrega targets e inputs para treinamento
% Pv = matriz de pontos destinados a validacao da rede (inputs)
% Tv = matriz de pontos destinados a validacao da rede (targets)
% Pa = matriz de pontos destinados ao aprendizado da rede (inputs)
% Ta = matriz de pontos destinados ao aprendizado da rede (targets)
% TR = training record
%*****
Pa = ST(1:(end-h-v-1));
Ta = ST(2:(end-h-v));
Pv = ST(1:(end-h-1));
Tv = ST(2:(end-h));
% entradas e alvos convertidos em sequencias
Paseq = con2seq(Pa);
Taseq = con2seq(Ta);
Pvseq = con2seq(Pv);
Tvseq = con2seq(Tv);
%*****
% cria rede de Elman (1-1-1) com 1 camada de entrada e
% 1 camada oculta de S1 neuronios com função tangente sigmoide (tansig)
% 1 camada de saída com função linear (purelin)
net = newelm(Pa,Ta,S1);
% algoritmo de ajuste de pesos
net.trainFcn = 'traingdm';
% algoritmo de treinamento: BPTT
net.gradientFcn = 'calcgbtt';
%=====
% define taxa de aprendizado
net.trainParam.lr = alfa;
%=====
% treina rede de epoca em epoca
net.trainParam.epochs = 1;
% enquanto nao atinge maximo de épocas e o erro de validacao não cresce
% por mais de um numero vmax de epocas sucessivas.
epoca = 1;
validfail = 0;
y1(1) = 0;
y2(1) = 0;
while (epoca <= emax) && (validfail < vmax) ;
    [net,TR] = train(net,Paseq,Taseq); % treina rede/aprendizado
    % inibe display da janela de aprendizado
    net.trainParam.showWindow = false;
    Y = sim(net,Pvseq); % simulacao no intervalo de validacao
    Z = seq2con(Y); % conversao de formato sequencial para matriz
    errovalid = Tv - Z{1,1}; % calcula erro de validacao
    msev = mse(errovalid); % erro mse de validacao
    if (epoca > 1) && (msev > msevant),
        validfail = validfail + 1 ; % contador de falhas de validacao
        if (validfail == 1)
            stopesoI = pesoI; % guarda pesos do early stopping
            stopesoL = pesoL;
            stopesoB = pesoB;
        end
    else
        validfail = 0; % zera o contador cada vez que o erro diminui
    end
    pesoI = net.IW;
    pesoL = net.LW;
    pesoB = net.b;
end

```

```

msea = TR.perf(2);% erro mse de aprendizado esta registrado com TR
% TR.perf(i) é uma matriz que cresce uma coluna a cada época
% a primeira coluna é o erro inicial e coluna adicional é o erro
% no final da época. Como estamos treinando de 1 em 1 época, teremos
% TR.perf(1) = erro inicial e TR.perf(2)=erro no final da (única) época
y1(epoca) = msea;
y2(epoca) = msev;
    epoca = epoca + 1 % deixar sem ; para acompanhar processamento
validfail % idem
msea
elapsed_time_parcial = (toc)/60
% armazena mse do intervalo de validacao para comparar na prox época
msevant = msev;
end
% Se ocorreu early stopping, a rede deve retroagir e recuperar pesos
% de vmax épocas atrás.
if (validfail == vmax)
    net.IW = stopesoI;
    net.LW = stopesoL;
    net.b = stopesoB;
else
end
% grafico de validacao cruzada
t=(1:epoca-1) ;
%plot(t,y1(t),'b',t,y2(t),'r--o','linewidth',2);
plot(t,y2(t),'r--o',t,y1(t),'b','linewidth',2);
title('Validação Cruzada:          Treinamento (-)          Validação (o)');
xlabel('época');
ylabel('mse');
%
% Fim do treinamento da rede, lembrar que a saída da rede é transformada
% pelos pré-processamentos
%
```

APÊNDICE 12: CÓDIGO MATLAB PARA PREVISÃO DA SÉRIE ADUBOS, COM PRÉ-PROCESSAMENTO

```
%-- 20/04/10 11:11 --%
% Previsao de Serie Temporal com Rede Elman no Matlab
% Previsão para SÉRIE ADUBOS COM PROCESSAMENTO
%*****
% carregar arquivo de dados com a rede treinada
clear all

                                load RedeAnda_N10_msev0506.mat
% informar tamanho da serie incluindo intervalo ex-post
T = 228;
%=====
% Previsao ESTÁTICA para o intervalo de teste
%=====
% rede simulada com entrada da origem até o penultimo ponto
Pf = ST(1:end-1);
Pfseq = con2seq(Pf);
Yf = sim(net,Pfseq);
Zf = seq2con(Yf);
Sf = cell2mat(Zf);
% A saída padronizada (Sf) é a previsao estática das Diferenças Sazonais.
% PÓS-PROCESSAMENTO
% reverter a transformacao mapminmax
Sf = mapminmax('reverse',Sf,PS);
F = Sf; % F = forecast da diferenca sazonal D sem padronizacao
% Se D é diferenca sazonal
% entao D(LYt,1,12) = LYt - LYt(-1) - [ LYt(-12) - LYt(-13) ]
% reconstitui série com valores iniciais e previsoes
LYt(1) = log(ST01);
LYt(2) = log(ST02);
LYt(3) = log(ST03);
LYt(4) = log(ST04);
LYt(5) = log(ST05);
LYt(6) = log(ST06);
LYt(7) = log(ST07);
LYt(8) = log(ST08);
LYt(9) = log(ST09);
LYt(10) = log(ST10);
LYt(11) = log(ST11);
LYt(12) = log(ST12);
LYt(13) = log(ST13);
LYt(14) = log(ST14);
% Cada ponto é estimado a partir dos anteriores conhecidos
Aux = log(STreal);
    for i = (1:T-14); % a serie encurtou 14 pontos
        LYt(i+14) = F(i) + Aux(i+13)+ Aux(i+2) - Aux(i+1);
    end;
% desfazer log
Yt = exp(LYt);
% plotar as series real e estimada
figure;
% serie original (targets) em azul
plot(STreal,'b','linewidth',1.5);
title('Dados reais (- azul)                               Previsão ESTÁTICA da rede (...
vermelho) ');
grid
hold on;
% estimativas (previsoes) em vermelho
```

```

STgraf = Yt;
plot(STgraf,'r--','linewidth',1.5);
%hold on;
%*****
% Qualidade da previsao
%
i=1:h;
Prev(i) = Yt(end-h+i);
% calculo do erros individuais no intervalo i=1:h
% erro = individual
% sumerroquad = soma dos "erros ao quadrado"
sumerroquad = 0;
% sumprev = soma das previsões ao quadrado
sumprevquad = 0;
% sumactual = soma dos valores conhecidos (ex-post) ao quadrado
sumactualquad = 0;
for i=1:h
    erro(i) = Prev(i)-STreal(end-h+i);
    sumerroquad = erro(i).^2 + sumerroquad;
    sumprevquad = Prev(i).^2 + sumprevquad;
    sumactualquad = STreal(end-h+i).^2 + sumactualquad;
end
%
% mean squared error = erro quadrático médio = MSE do forecast Estático
EMSE = sumerroquad/h;
% root mean squared error = RMSE do forecast Estático
ERMSE = sqrt(EMSE);
% ETIC = coef de ineq de Theil do forecast Estático
ETIC = ERMSE / ( sqrt(sumprevquad/h) + sqrt(sumactualquad/h) );
% EBIAS = bias proportion do MSE do forecast Estático
% Ybar = média dos valores conhecidos
Ybar = mean( STreal(end-h+1:end) );
EBIAS = ((mean(Prev) - Ybar ).^2) / EMSE;
% EVAR = variance proportion
% SYbar = desvio padrao "viesado" das previsoes
% SY = desvio padrao "viesado" dos valores reais
% => usar flag 1 no argumento de std
SYbar = std(Prev,1);
SY = std( STreal(end-h+1:end),1);
EVAR = ((SYbar - SY).^2) / EMSE;
% ECOV = covariance proportion
% ER = correlacao entre as previsoes e os valores reais
% Transpor matrizes para usar argumentos corretos na funcao "corr"
ER = corr (Prev', STreal(end-h+1:end)');
ECOV = (2 * (1 - ER)* SYbar * SY ) / EMSE;
% Prop = variavel teste = tem de ser igual a 1.0000 = 100%
Prop = EBIAS + EVAR + ECOV;
%
%=====
% Previsao DINÂMICA para o intervalo de teste
%=====
%
% reconstituir ST tal como na etapa de treinamento
ST = STpad;
% primeiro ponto previsto dinamicamente = previsão estática
Pfdin = ST(1:(end-h));
Pfseqdin = con2seq(Pfdin);
Yfdin = sim(net,Pfseqdin);
Zfdin = seq2con(Yfdin);
Sfdin = cell2mat(Zfdin);
%
```

```

% A previsao para demais pontos é calculada recorrentemente
for i=2:h
    Pfdin = [Pfdin Sfdin(end)];
    Pfseqdin = con2seq(Pfdin);
    Yfdin = sim(net,Pfseqdin);
    Zfdin = seq2con(Yfdin);
    Sfdin = cell2mat(Zfdin);
end
% Sfdin contem toda a serie estimada, desde a origem até o fim
% PÓS_PROCESSAMENTO
% desfazer a transformacao mapminmax
Sfdin = mapminmax('reverse',Sfdin,PS);
Fdin = Sfdin;
%
LYtdin(1) = log(ST01);
LYtdin(2) = log(ST02);
LYtdin(3) = log(ST03);
LYtdin(4) = log(ST04);
LYtdin(5) = log(ST05);
LYtdin(6) = log(ST06);
LYtdin(7) = log(ST07);
LYtdin(8) = log(ST08);
LYtdin(9) = log(ST09);
LYtdin(10) = log(ST10);
LYtdin(11) = log(ST11);
LYtdin(12) = log(ST12);
LYtdin(13) = log(ST13);
LYtdin(14) = log(ST14);
% Cada ponto é estimado a partir dos anteriores conhecidos...
Auxdin = log(STreal);
for i = 1:(T-14-h+1);
    LYtdin(i+14) = Fdin(i) + Auxdin(i+13)+ Auxdin(i+2) - Auxdin(i+1);
end;
%...exceto nos 11 pontos finais que sao calculados recursivamente
for i = (T-14-h+2):(T-14);
    LYtdin(i+14) = Fdin(i) + LYtdin(i+13)+ LYtdin(i+2) - LYtdin(i+1);
end;
% desfazer log
Ytdin = exp(LYtdin);
%*****
%
%plotar as series real e estimada
figure
% serie original (targets) em azul
plot(STreal,'b','linewidth',1.5);
title('Dados reais (- azul)
Previsão DINÂMICA da rede
(-- magenta) ');
grid
hold on;
SNgraf = Ytdin;
% estimativas (previsoes dinamicas) em magenta
plot(SNgraf,'m--','linewidth',1.5);
%
%*****
% Qualidade da previsão dinâmica
%
i=1:h;
Prevdin(i) = Ytdin(end-h+i);
% calculo do erros individuais no intervalo i=1:h
% errodin = erro individual dinamico
% sumerroquaddin = soma dos "erros dinamicos ao quadrado"

```

```

sumerroquaddin= 0;
% sumprevdin = soma das previsões ao quadrado
sumprevquaddin = 0;
% sumactualdin = soma dos valores conhecidos, ao quadrado
sumactualquaddin = 0;
for i=1:h
    errodin = Prevdin(i)-STreal(end-h+i);
    sumerroquaddin = errodin.^2 + sumerroquaddin;
    sumprevquaddin = Prevdin(i).^2 + sumprevquaddin;
    sumactualquaddin = STreal(end-h+i).^2 + sumactualquaddin;
end
% mean squared error = erro quadrático médio = MSE
EMSEdin = sumerroquaddin/h;
% root mean squared error, tem a vantagem de estar na unidade da variável
ERMSEdin = sqrt(EMSEdin);
% ETIC = coef de ineq de Theil do forecast Dinamico
ETICdin = ERMSEdin /(sqrt(sumprevquaddin/h) + sqrt(sumactualquaddin/h) );
% EBIASdin = bias proportion do MSE do forecast Dinamico
% Ybardin = média dos valores conhecidos reais
Ybardin = mean( STreal(end-h+1:end) );
EBIASdin = ((mean(Prevdin) - Ybardin ).^2) / EMSEdin;
% EVARdin = variance proportion
% SYbardin = desvio padrao "viesado" das previsoes
% SYdin = desvio padrao "viesado" dos valores reais
% => usar flag 1 no argumento de std
SYbardin = std(Prevdin,1);
SYdin = std( STreal(end-h+1:end),1);
EVARdin = ((SYbardin - SYdin).^2) / EMSEdin;
% ECOVdin = covariance proportion
% ERdin = correlacao entre as previsoes e os valores reais
% Transpor matrizes para usar argumentos corretos na funcao "corr"
ERdin = corr (Prevdin', STreal(end-h+1:end)');
ECOVdin = (2 * (1 - ERdin)* SYbardin * SYdin ) / EMSEdin;
% Prop = variavel teste = tem de ser igual a 1.0000 = 100%
Propdin = EBIASdin + EVARdin + ECOVdin;

```

APÊNDICE 13: CÓDIGO MATLAB PARA TREINAMENTO DA SÉRIE FARMA, COM PRÉ-PROCESSAMENTO

```
%-- 20/04/10 11:11 --%
% TREINAMENTO de Rede Elman com Matlab
% SÉRIE FARMA DADOS PRÉ-PROCESSADOS
%=====
% Para interromper processamento tecle "CTRL + C"
%=====
clear all; %limpar area de trabalho
%*****
% carregar arquivo de dados originais incluindo os 12 pontos ex-post
load farma84
%
% serie inicial preservada para uso futuro
STreal = ST;
ST01 = ST(1);
ST02 = ST(2);
ST03 = ST(3);
ST04 = ST(4);
ST05 = ST(5);
ST06 = ST(6);
ST07 = ST(7);
ST08 = ST(8);
ST09 = ST(9);
ST10 = ST(10);
ST11 = ST(11);
ST12 = ST(12);
ST13 = ST(13);
ST14 = ST(14);
%=====
% especificacao da rede e demais dados solicitados ao usuario:
% h = tamanho do intervalo em que será feita previsão ex-post
% v = numero de periodos usados para validacao (25%Valid+75%Aprendiz)
% S1 = numero de neuronios na camada oculta
% emax = maximo de epocas, limitado pelo "early stopping"
% vmax = maximo de vezes em que se permite aumentar erro na validacao
% alfa = taxa de aprendizado
% T = tamanho da serie incluindo h
%
h = 12;
v = 18;
S1 = 11;
emax = 1000;
vmax = 15;
alfa = 0.025;
T = 84;
%
%*****
% inicia contagem de tempo de processamento
tic;
%*****
% PRÉ-PROCESSAMENTO
% logaritmo
ST = log(ST);
% primeira diferença regular (serie encolhe 1 ponto à esquerda)
ST = diff(ST);
% 1a diferença sazonal de 12 meses(a serie encolhe em mais 12 pontos)
i=(13:T-1);
```

```

        STdsaz(i-12)=ST(i)-ST(i-12);
        ST = STdsaz;
        % padronizacao entre -1 e 1
        [STpad,PS] = mapminmax(ST);
        ST = STpad;
%*****
% carrega targets e inputs para treinamento
% Pv = matriz de pontos destinados a validacao da rede (inputs)
% Tv = matriz de pontos destinados a validacao da rede (targets)
% Pa = matriz de pontos destinados ao aprendizado da rede (inputs)
% Ta = matriz de pontos destinados ao aprendizado da rede (targets)
% TR = training record
%*****
Pa = ST(1:(end-h-v-1));
Ta = ST(2:(end-h-v));
Pv = ST(1:(end-h-1));
Tv = ST(2:(end-h));
% entradas e alvos convertidos em sequencias
Paseq = con2seq(Pa);
Taseq = con2seq(Ta);
Pvseq = con2seq(Pv);
Tvseq = con2seq(Tv);
%*****
% cria rede de Elman (1-1-1) com 1 camada de entrada e
% 1 camada oculta de S1 neuronios com função tangente sigmoide (tansig)
% 1 camada de saída com função linear (purelin)
net = newelm(Pa,Ta,S1);
% algoritmo de ajuste de pesos
net.trainFcn = 'traingdm';
% algoritmo de treinamento: BPTT
net.gradientFcn = 'calcgbtt';
%=====
% define taxa de aprendizado
net.trainParam.lr = alfa;
%=====
% treina rede de epoca em epoca
net.trainParam.epochs = 1;
% enquanto nao atinge maximo de épocas e o erro de validacao não cresce
% por mais de um numero vmax de epocas sucessivas.
epoca = 1;
validfail = 0;
y1(1) = 0;
y2(1) = 0;
while (epoca <= emax) && (validfail < vmax) ;
    [net,TR] = train(net,Paseq,Taseq); % treina rede/aprendizado
    % inibe display da janela de aprendizado
    net.trainParam.showWindow = false;
    Y = sim(net,Pvseq); % simulacao no intervalo de validacao
    Z = seq2con(Y); % conversao de formato sequencial para matriz
    errovalid = Tv - Z{1,1}; % calcula erro de validacao
    msev = mse(errovalid); % erro mse de validacao
    if (epoca > 1) && (msev > msevant),
        validfail = validfail + 1 ; % contador de falhas de validacao
        if (validfail == 1)
            stopesoI = pesoI; % guarda pesos do early stopping
            stopesoL = pesoL;
            stopesoB = pesoB;
        end
    else
        validfail = 0; % zera o contador cada vez que o erro diminui
    end
end
end

```



```

pesoI = net.IW;
pesoL = net.LW;
pesoB = net.b;
msea = TR.perf(2); % erro mse de aprendizado esta registrado com TR
% TR.perf(i) é uma matriz que cresce uma coluna a cada época
% a primeira coluna é o erro inicial e coluna adicional é o erro
% no final da época. Como estamos treinando de 1 em 1 época, teremos
% TR.perf(1) = erro inicial e TR.perf(2)=erro no final da (única) época
y1(epoca) = msea;
y2(epoca) = msev;
    epoca = epoca + 1 % deixar sem ; para acompanhar processamento
validfail % idem
msea
elapsed_time_parcial = (toc)/60
% armazena mse do intervalo de validacao para comparar na prox época
msevant = msev;
end
% Se ocorreu early stopping, a rede deve retroagir e recuperar pesos
% de vmax épocas atrás.
if (validfail == vmax)
    net.IW = stopesoI;
    net.LW = stopesoL;
    net.b = stopesoB;
else
end
% grafico de validacao cruzada
t=(1:epoca-1) ;
%plot(t,y1(t),'b',t,y2(t),'r--o','linewidth',2);
plot(t,y2(t),'r--o',t,y1(t),'b','linewidth',2);
title('Validação Cruzada:          Treinamento (-)          Validação (o)');
xlabel('época');
ylabel('mse');
%
% Fim do treinamento da rede, lembrar que a saída da rede é transformada
% pelos pré-processamentos
%
```

APÊNDICE 14: CÓDIGO MATLAB PARA PREVISÃO DA SÉRIE FARMA, COM PRÉ-PROCESSAMENTO

```
%-- 20/04/10 11:11 --%
% Previsao de Serie Temporal com Rede Elman no Matlab
% Previsão para SÉRIE FARMA COM PRÉ- PROCESSAMENTO
%*****
% carregar arquivo de dados com a rede treinada
clear all

                                load RedeFarma_N13_msev0221.mat
% informar tamanho da serie incluindo horizonte de previsão
T = 84;
%=====
% Previsao ESTÁTICA para o intervalo de teste
%=====
% rede simulada com entrada da origem até o penultimo ponto ex-post
Pf = ST(1:end-1);
Pfseq = con2seq(Pf);
Yf = sim(net,Pfseq);
Zf = seq2con(Yf);
Sf = cell2mat(Zf);
% A saída padronizada (Sf) é a previsao estática das Diferenças Sazonais.
% PÓS-PROCESSAMENTO
% reverter a transformacao mapminmax
Sf = mapminmax('reverse',Sf,PS);
F = Sf; % F = forecast da diferença sazonal D sem padronizacao
% Se D é diferença sazonal
% entao D(LYt,1,12) = LYt - LYt(-1) - [ LYt(-12) - LYt(-13) ]
% reconstitui série com valores iniciais e previsoes
LYt(1) = log(ST01);
LYt(2) = log(ST02);
LYt(3) = log(ST03);
LYt(4) = log(ST04);
LYt(5) = log(ST05);
LYt(6) = log(ST06);
LYt(7) = log(ST07);
LYt(8) = log(ST08);
LYt(9) = log(ST09);
LYt(10) = log(ST10);
LYt(11) = log(ST11);
LYt(12) = log(ST12);
LYt(13) = log(ST13);
LYt(14) = log(ST14);
% Cada ponto é estimado a partir dos anteriores conhecidos
Aux = log(STreal);
for i = (1:T-14);
    LYt(i+14) = F(i) + Aux(i+13)+ Aux(i+2) - Aux(i+1);
end;
% desfazer log
Yt = exp(LYt);
% plotar as series real e estimada
figure;
% serie original (targets) em azul
plot(STreal,'b','linewidth',1.5);
title('Dados reais (- azul)                               Previsão ESTÁTICA da rede (...
vermelho) ');
grid
hold on;
% estimativas (previsoes) em vermelho
```

```

STgraf = Yt;
plot(STgraf, 'r--', 'linewidth', 1.5);
%hold on;
%*****
% Qualidade da previsao
%
i=1:h;
Prev(i) = Yt(end-h+i);
% calculo do erros individuais no intervalo i=1:h
% erro = individual
% sumerroquad = soma dos "erros ao quadrado"
sumerroquad = 0;
% sumprev = soma das previsões ao quadrado
sumprevquad = 0;
% sumactual = soma dos valores reais ao quadrado
sumactualquad = 0;
for i=1:h
    erro(i) = Prev(i)-STreal(end-h+i);
    sumerroquad = erro(i).^2 + sumerroquad;
    sumprevquad = Prev(i).^2 + sumprevquad;
    sumactualquad = STreal(end-h+i).^2 + sumactualquad;
end
%
% mean squared error = erro quadrático médio = MSE do forecast Estático
EMSE = sumerroquad/h;
% root mean squared error = RMSE do forecast Estático
ERMSE = sqrt(EMSE);
% ETIC = coef de ineq de Theil do forecast Estático
ETIC = ERMSE / ( sqrt(sumprevquad/h) + sqrt(sumactualquad/h) );
% EBIAS = bias proportion do MSE do forecast Estático
% YBar = média dos valores reais
Ybar = mean( STreal(end-h+1:end) );
EBIAS = ((mean(Prev) - Ybar ).^2) / EMSE;
% EVAR = variance proportion
% SYbar = desvio padrao "viesado" das previsoes
% SY = desvio padrao "viesado" dos valores reais
% => usar flag 1 no argumento de std
SYbar = std(Prev,1);
SY = std( STreal(end-h+1:end),1);
EVAR = ((SYbar - SY).^2) / EMSE;
% ECOV = covariance proportion
% ER = correlacao entre as previsoes e os valores reais
% Transpor matrizes para usar argumentos corretos na funcao "corr"
ER = corr (Prev', STreal(end-h+1:end)');
ECOV = (2 * (1 - ER)* SYbar * SY ) / EMSE;
% Prop = variavel teste = tem de ser igual a 1.0000 = 100%
Prop = EBIAS + EVAR + ECOV;
%
%=====
% Previsao DINÂMICA para o intervalo de teste
%=====
%
% reconstituir ST tal como na etapa de treinamento
ST = STpad;
% primeiro ponto previsto dinamicamente = previsão estática
Pfdin = ST(1:(end-h));
Pfseqdin = con2seq(Pfdin);
Yfdin = sim(net,Pfseqdin);
Zfdin = seq2con(Yfdin);
Sfdin = cell2mat(Zfdin);
%
```

```

% A previsao para demais pontos é calculada recorrentemente
for i=2:h
    Pfdin = [Pfdin Sfdin(end)];
    Pfseqdin = con2seq(Pfdin);
    Yfdin = sim(net,Pfseqdin);
    Zfdin = seq2con(Yfdin);
    Sfdin = cell2mat(Zfdin);
end
% Sfdin contem toda a serie estimada, desde a origem até o fim
% PÓS_PROCESSAMENTO
    % desfazer a transformacao mapminmax
    Sfdin = mapminmax('reverse',Sfdin,PS);
    Fdin = Sfdin;
    %
    LYtdin(1) = log(ST01);
    LYtdin(2) = log(ST02);
    LYtdin(3) = log(ST03);
    LYtdin(4) = log(ST04);
    LYtdin(5) = log(ST05);
    LYtdin(6) = log(ST06);
    LYtdin(7) = log(ST07);
    LYtdin(8) = log(ST08);
    LYtdin(9) = log(ST09);
    LYtdin(10) = log(ST10);
    LYtdin(11) = log(ST11);
    LYtdin(12) = log(ST12);
    LYtdin(13) = log(ST13);
    LYtdin(14) = log(ST14);
    % Cada ponto é estimado a partir dos anteriores conhecidos...
    Auxdin = log(STreal);
    for i = (1:T-14-h+1);
        LYtdin(i+14) = Fdin(i) + Auxdin(i+13)+ Auxdin(i+2) - Auxdin(i+1);
    end;
    %...exceto nos 11 pontos finais que sao calculados recursivamente
    for i = (T-14-h+2:T-14);
        LYtdin(i+14) = Fdin(i) + LYtdin(i+13)+ LYtdin(i+2) - LYtdin(i+1);
    end;
    % desfazer log
    Ytdin = exp(LYtdin);
%*****
%
%plotar as series real e estimada
figure
% serie original (targets) em azul
plot(STreal,'b','linewidth',1.5);
title('Dados reais (- azul)
Previsão DINÂMICA da rede
(-- magenta) ');
grid
hold on;
SNgraf = Ytdin;
% estimativas (previsoes dinamicas) em magenta
plot(SNgraf,'m--','linewidth',1.5);
%
%*****
% Qualidade da previsão dinâmica
%
i=1:h;
Prevdin(i) = Ytdin(end-h+i);
% calculo do erros individuais no intervalo i=1:h
% errodin = erro individual dinamico
% sumerroquaddin = soma dos "erros dinamicos ao quadrado"

```

```

sumerroquaddin= 0;
% sumprevdin = soma das previsões ao quadrado
sumprevquaddin = 0;
% sumactualdin = soma dos valores reais) ao quadrado
sumactualquaddin = 0;
for i=1:h
    errodin = Prevdin(i)-STreal(end-h+i);
    sumerroquaddin = errodin.^2 + sumerroquaddin;
    sumprevquaddin = Prevdin(i).^2 + sumprevquaddin;
    sumactualquaddin = STreal(end-h+i).^2 + sumactualquaddin;
end
% mean squared error = erro quadrático médio = MSE
EMSEdin = sumerroquaddin/h;
% root mean squared error, tem a vantagem de estar na unidade da variável
ERMSEdin = sqrt(EMSEdin);
% ETIC = coef de ineq de Theil do forecast Dinamico
ETICdin = ERMSEdin / ( sqrt(sumprevquaddin/h) + sqrt(sumactualquaddin/h) );
% EBIASdin = bias proportion do MSE do forecast Dinamico
% Ybardin = média dos valores reais
Ybardin = mean( STreal(end-h+1:end) );
EBIASdin = ((mean(Prevdin) - Ybardin ).^2) / EMSEdin;
% EVARDin = variance proportion
% SYbardin = desvio padrao "viesado" das previsoes
% SYdin = desvio padrao "viesado" dos valores reais
% => usar flag 1 no argumento de std
SYbardin = std(Prevdin,1);
SYdin = std( STreal(end-h+1:end),1);
EVARDin = ((SYbardin - SYdin).^2) / EMSEdin;
% ECOVDin = covariance proportion
% ERdin = correlacao entre as previsoes e os valores reais
% Transpor matrizes para usar argumentos corretos na funcao "corr"
ERdin = corr (Prevdin', STreal(end-h+1:end)');
ECOVDin = (2 * (1 - ERdin)* SYbardin * SYdin ) / EMSEdin;
% Prop = variavel teste = tem de ser igual a 1.0000 = 100%
Propdin = EBIASdin + EVARDin + ECOVDin;

```

APÊNDICE 15: SÉRIES PADRONIZADAS

Série AEREO: padronizada no intervalo [-1,1]

Pax-km	2003	2004	2005	2006	2007	2008
JAN	-0,6321	-0,5003	-0,2092	0,4710	0,6259	1
FEV	-1	-0,8356	-0,6908	-0,2346	0,0198	0,4273
MAR	-0,8589	-0,8592	-0,5304	-0,0038	0,1054	0,5368
ABR	-0,8382	-0,8158	-0,5619	-0,1058	0,3460	0,6028
MAI	-0,9960	-0,7254	-0,4835	0,0140	0,2856	0,8744
JUN	-0,9932	-0,7527	-0,4868	0,0117	0,2531	0,6096
JUL	-0,5960	-0,3533	0,2309	0,3296	0,4860	0,8469
AGO	-0,8256	-0,6019	-0,1412	0,0523	-0,1547	0,9145
SET	-0,9163	-0,7185	-0,1446	0,0430	0,0767	0,3475
OUT	-0,7948	-0,5574	0,0657	0,1612	0,4990	0,4801
NOV	-0,8387	-0,6496	-0,1620	-0,0357	0,4800	0,4971
DEZ	-0,6784	-0,5171	0,1990	0,3812	0,8223	0,9225

Série ADUBOS: padronizada no intervalo [-1,1]

Ton	Jan	Fev	Mar	Abr	Mai	Jun	Jul	Ago	Set	Out	Nov	Dez
1991	-0,9593	-1	-0,9305	-0,8826	-0,7838	-0,8107	-0,7575	-0,5781	-0,4943	-0,3068	-0,5463	-0,8452
1992	-0,9391	-0,9214	-0,8735	-0,8795	-0,8682	-0,7804	-0,6406	-0,5727	-0,4103	-0,2572	-0,4771	-0,7748
1993	-0,9098	-0,9170	-0,8164	-0,8222	-0,7454	-0,6652	-0,5149	-0,3874	-0,2783	-0,2506	-0,4726	-0,8090
1994	-0,8805	-0,8433	-0,7563	-0,8138	-0,6774	-0,6387	-0,6435	-0,3864	-0,1808	-0,1163	-0,1545	-0,6026
1995	-0,7912	-0,8297	-0,7475	-0,8886	-0,8610	-0,8861	-0,7649	-0,3938	-0,2467	-0,0538	-0,2277	-0,7078
1996	-0,8144	-0,8013	-0,8140	-0,8268	-0,7257	-0,7328	-0,5553	-0,3468	-0,1613	0,0682	-0,1342	-0,6563
1997	-0,7710	-0,7564	-0,7384	-0,7401	-0,6590	-0,5874	-0,3568	-0,2188	0,0379	0,1488	-0,2499	-0,5978
1998	-0,6772	-0,6928	-0,6869	-0,8099	-0,7228	-0,6114	-0,4228	-0,1879	0,1017	0,3131	-0,0361	-0,5237
1999	-0,7328	-0,7223	-0,7340	-0,9396	-0,8725	-0,8049	-0,6241	-0,2500	0,1592	0,3522	0,0819	-0,4942
2000	-0,6171	-0,6184	-0,6405	-0,7996	-0,6369	-0,4246	-0,1840	0,1187	0,2122	0,2817	-0,0266	-0,5224
2001	-0,5545	-0,6675	-0,7256	-0,8083	-0,6347	-0,2911	0,0181	0,3259	0,1982	0,4053	-0,1228	-0,5688
2002	-0,4943	-0,5844	-0,7075	-0,6057	-0,2817	0,0124	0,2345	0,2191	0,3450	0,2863	-0,0520	-0,4933
2003	-0,4846	-0,3613	-0,4859	-0,5501	-0,2849	-0,0069	0,3063	0,5599	0,6915	0,7878	0,2970	-0,2424
2004	-0,1823	-0,6172	-0,6172	-0,5211	-0,1735	0,1110	0,1493	0,4885	0,7614	0,8168	0,3402	-0,3476
2005	-0,5436	-0,5171	-0,6276	-0,7467	-0,5655	-0,3778	-0,0499	0,4576	0,7579	0,8186	0,3438	-0,3823
2006	-0,4759	-0,4844	-0,5868	-0,7480	-0,6642	-0,4965	-0,1298	0,4359	0,7917	1	0,6013	-0,1741
2007	-0,1905	-0,2731	-0,1802	-0,3589	-0,1373	-0,0260	0,2224	0,6804	0,6882	0,8165	0,3298	-0,1889
2008	-0,0080	-0,0113	-0,0877	-0,1208	0,0656	0,3403	0,3712	0,1361	0,2082	0,0977	-0,4296	-0,5693
2009	-0,3550	-0,3238	-0,3114	-0,5206	-0,3261	-0,0423	0,3501	0,4488	0,5324	0,6461	0,1937	-0,2731

Série FARMA: padronizada no intervalo [-1,1]

	2003	2004	2005	2006	2007	2008	2009
JAN	-0,88	-0,78	-0,78	-0,60	-0,49	-0,11	0,13
FEV	-1	-0,88	-0,87	-0,77	-0,66	-0,36	-0,06
MAR	-0,99	-0,58	-0,54	-0,45	-0,31	-0,07	0,36
ABR	-0,94	-0,75	-0,64	-0,60	-0,42	-0,05	0,26
MAI	-0,81	-0,63	-0,53	-0,42	-0,24	0,10	0,39
JUN	-0,90	-0,67	-0,53	-0,50	-0,24	-0,01	0,34
JUL	-0,79	-0,64	-0,54	-0,47	-0,23	0,15	0,57
AGO	-0,83	-0,69	-0,44	-0,39	-0,12	0,12	0,53
SET	-0,83	-0,76	-0,58	-0,47	-0,25	0,19	0,44
OUT	-0,72	-0,71	-0,53	-0,42	-0,13	0,25	0,62
NOV	-0,80	-0,77	-0,53	-0,46	-0,23	0,13	0,47
DEZ	-0,47	-0,38	-0,18	-0,13	0,19	0,63	1

APÊNDICE 16: TRANSFORMAÇÃO *MAPMINMAX*

PADRONIZAÇÃO DA ESCALA ORIGINAL DAS SÉRIES NO INTERVALO [-1,1]
EXEMPLO COM VARIÁVEL AEREO

Valores originais decrecentes	Valores padronizados decrecentes
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10
11	11
12	12
13	13
14	14
15	15
16	16
17	17
18	18
19	19
20	20
21	21
22	22
23	23
24	24
25	25
26	26
27	27
28	28
29	29
30	30
31	31
32	32
33	33
34	34
35	35
36	36
37	37
38	38
39	39
40	40
41	41
42	42
43	43
44	44
45	45
46	46
47	47
48	48
49	49
50	50
51	51
52	52
53	53
54	54
55	55
56	56
57	57
58	58
59	59
60	60
61	61
62	62
63	63
64	64
65	65
66	66
67	67
68	68
69	69
70	70
71	71
72	72
73	73
74	74
75	75
76	76
77	77
78	78
79	79
80	80
81	81
82	82
83	83
84	84
85	85
86	86
87	87
88	88
89	89
90	90
91	91
92	92
93	93
94	94
95	95
96	96
97	97
98	98
99	99
100	100

[illegible]

$$\frac{(AEREO - AEREO_{\min})}{(AEREO_p - (-1))} = \frac{(AEREO_{\max} - AEREO_{\min})}{2}$$

$$(AEREO_p - (-1)) = \frac{(AEREO - AEREO_{\min}) \times 2}{(AEREO_{\max} - AEREO_{\min})}$$

$$AEREO_p = \frac{(AEREO - AEREO_{\min}) \times 2}{(AEREO_{\max} - AEREO_{\min})} - 1$$

$$AEREO_p = \frac{(AEREO - AEREO_{\min}) \times 2}{(AEREO_{\max} - AEREO_{\min})} - 1$$

$$AEREOp = \frac{(AEREO - AEREO_{\min}) \times 2}{(AEREO_{\max} - AEREO_{\min})} - 1$$

$$\text{AEREOp} = \frac{(\text{AEREO} - 1.950.216) \times 2}{(4.494.055 - 1.950.216)} - 1$$

AEREOp	=	$\frac{(AERE0 - 1.950.216) \times 2}{2.543.839}$	- 1
--------	---	--	-----

4.494.055	↖	1
4.395.544		0,9225
...		
AEREO		
	↗	AEREOp
		-0,9960
		-1
...		
1.955.288		
1.950.216	↙	

$$\text{AEREOp} = \frac{(4.395.544 - 1.950.216) \times 2}{2.543.839} - 1 = 0,9225$$

$$\text{AEREOp} = \frac{(1.955.288 - 1.950.216) \times 2}{2.543.839} - 1 = -0,9960$$

A função `mapminmax` do MATLAB generaliza esta padronização $[-1,1]$, para qualquer escala original, como segue:

$$\text{valor padronizado} = \frac{(\text{valor original} - \text{valor original mínimo}) \times 2}{(\text{valor original máximo} - \text{valor original mínimo})} - 1$$

ANEXO A: SÉRIES TEMPORAIS

Série AEREO: evolução mensal da utilização de assentos na aviação comercial doméstica

Pax-km	2003	2004	2005	2006	2007	2008
JAN	2.418.126	2.585.754	2.956.113	3.821.247	4.018.237	4.494.055
FEV	1.950.216	2.159.358	2.343.476	2.923.702	3.247.305	3.765.600
MAR	2.129.704	2.129.243	2.547.510	3.217.297	3.356.140	3.904.922
ABR	2.156.028	2.184.543	2.507.416	3.087.621	3.662.263	3.988.844
MAI	1.955.288	2.299.444	2.607.190	3.239.975	3.585.334	4.334.282
JUN	1.958.866	2.264.725	2.602.957	3.237.004	3.544.114	3.997.441
JUL	2.464.120	2.772.747	3.515.861	3.641.400	3.840.320	4.299.330
AGO	2.172.006	2.456.582	3.042.493	3.288.666	3.025.377	4.385.300
SET	2.056.642	2.308.222	3.038.274	3.276.878	3.319.732	3.664.170
OUT	2.211.180	2.513.176	3.305.744	3.427.131	3.856.766	3.832.816
NOV	2.155.412	2.395.927	3.016.042	3.176.791	3.832.603	3.854.419
DEZ	2.359.239	2.564.455	3.475.189	3.706.971	4.268.026	4.395.544

Fonte: ANAC - Agência Nacional de Aviação Civil

Série ADUBOS: Fertilizantes entregues ao consumidor final

Ton	Jan	Fev	Mar	Abr	Mai	Jun	Jul	Ago	Set	Out	Nov	Dez
1991	365.765	302.016	411.058	486.090	640.992	598.862	682.282	963.590	1.095.017	1.389.079	1.013.418	544.799
1992	397.444	425.216	500.421	491.041	508.629	646.349	865.538	972.088	1.226.769	1.466.813	1.121.987	655.168
1993	443.419	432.141	589.905	580.819	701.192	827.067	1.062.682	1.262.606	1.433.703	1.477.193	1.129.062	601.545
1994	489.429	547.790	684.202	594.024	807.930	868.538	861.013	1.264.196	1.586.610	1.687.712	1.627.866	925.169
1995	629.416	568.989	697.965	476.657	519.905	480.664	670.714	1.252.656	1.483.305	1.785.730	1.513.101	760.269
1996	593.109	613.585	593.611	573.534	732.115	721.050	999.348	1.326.360	1.617.139	1.977.111	1.659.657	840.981
1997	661.039	683.938	712.218	709.515	836.746	949.072	1.310.648	1.527.036	1.929.563	2.103.462	1.478.166	932.661
1998	808.248	783.732	793.043	600.047	736.615	911.410	1.207.152	1.575.375	2.029.610	2.361.048	1.813.412	1.048.878
1999	721.039	737.447	719.145	396.718	501.985	607.874	891.494	1.478.046	2.119.669	2.422.354	1.998.518	1.095.193
2000	902.466	900.438	865.707	616.220	871.349	1.204.278	1.581.570	2.056.287	2.202.774	2.311.834	1.828.423	1.050.870
2001	1.000.598	823.453	732.239	602.675	874.778	1.413.698	1.898.419	2.381.052	2.180.921	2.505.701	1.677.580	978.100
2002	1.094.933	953.664	760.647	920.308	1.428.405	1.889.534	2.237.878	2.213.613	2.411.093	2.319.041	1.788.616	1.096.536
2003	1.110.226	1.303.585	1.108.111	1.007.443	1.423.388	1.859.329	2.350.398	2.748.122	2.954.363	3.105.386	2.335.816	1.490.065
2004	1.584.158	902.266	902.225	1.052.960	1.598.064	2.044.184	2.104.156	2.636.088	3.063.968	3.150.843	2.403.515	1.325.062
2005	1.017.697	1.059.181	885.927	699.183	983.388	1.277.729	1.791.856	2.587.642	3.058.556	3.153.798	2.409.177	1.270.597
2006	1.123.916	1.110.565	950.019	697.242	828.593	1.091.476	1.666.483	2.553.623	3.111.515	3.438.171	2.812.978	1.597.153
2007	1.571.409	1.441.857	1.587.530	1.307.289	1.654.836	1.829.387	2.218.896	2.936.944	2.949.204	3.150.471	2.387.268	1.573.902
2008	1.857.471	1.852.384	1.732.624	1.680.678	1.972.888	2.403.717	2.452.191	2.083.541	2.196.593	2.023.363	1.196.385	977.397
2009	1.313.453	1.362.398	1.381.814	1.053.783	1.358.762	1.803.792	2.419.063	2.573.790	2.704.962	2.883.244	2.173.906	1.441.854

Fonte: ANDA - Associação Nacional para Difusão de Adubos

Série FARMA: Índice de vendas no varejo de produtos farmacêuticos, médicos, ortopédicos, de perfumaria e de cosméticos

Índice	2003	2004	2005	2006	2007	2008	2009
JAN	97,30	102,34	102,66	111,56	116,98	135,95	148,09
FEV	91,47	97,35	98,02	103,01	108,34	123,53	138,40
MAR	91,90	112,43	114,50	119,13	126,01	138,18	159,88
ABR	94,72	103,80	109,40	111,57	120,60	138,97	154,61
MAI	100,77	110,03	114,84	120,75	129,43	146,43	161,29
JUN	96,24	108,27	115,04	116,74	129,46	140,92	158,66
JUL	102,02	109,78	114,59	118,23	129,93	149,26	170,42
AGO	99,81	106,84	119,37	122,26	135,78	147,70	168,32
SET	100,21	103,64	112,70	117,89	129,04	151,09	163,51
OUT	105,67	105,85	115,17	120,60	135,16	153,96	172,98
NOV	101,61	103,26	114,85	118,77	129,94	148,40	165,16
DEZ	118,28	122,77	132,76	134,90	151,38	173,15	191,84

Fonte: IBGE - PMC www.sidra.ibge.gov.br/bda/comerc/

Nota: valores máximo e mínimo de cada série indicados em negrito nas tabelas acima.