UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE PRODUÇÃO

APLICAÇÃO DE SÉRIES TEMPORAIS E REDES NEURAIS EM UM AMBIENTE DE COMPUTAÇÃO EM NUVEM

DISSERTAÇÃO DE MESTRADO

Tatiana Fernanda Mousquer dos Santos

Santa Maria, RS, Brasil
2014

APLICAÇÃO DE SÉRIES TEMPORAIS E REDES NEURAIS EM UM AMBIENTE DE COMPUTAÇÃO EM NUVEM

Tatiana Fernanda Mousquer dos Santos

Dissertação apresentada ao Curso de Mestrado do Programa de Pós-Graduação em Engenharia de Produção, Área de Concentração em Gerência, da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do grau de

Mestre em Engenharia de Produção

Orientador: Prof. Dr. Adriano Mendonça Souza

Santa Maria, RS, Brasil
2014

Universidade Federal de Santa Maria
Centro de Tecnologia
Programa de Pós-Graduação em Engenharia de Produção

A Comissão Examinadora, abaixo assinada,
aprova a Dissertação de Mestrado

APLICAÇÃO DE SÉRIES TEMPORAIS E REDES NEURAIS EM UM AMBIENTE DE COMPUTAÇÃO EM NUVEM

elaborada por
Tatiana Fernanda Mousquer dos Santos

Como requisito parcial para obtenção do grau de
Mestre em Engenharia de Produção

COMISSÃO EXAMINADORA:

Adriano Mendonça Souza, Dr.
(Presidente/Orientador)

Roselaine Ruviaro Zanini, Dra. (UFSM)

Rozelaine de Fatima Franzin, Dra. (URI – Santo Ângelo)

Santa Maria, 6 de Março de 2014.

AGRADECIMENTOS

Primeiramente a Deus, pelas oportunidades que me concedeu e por me fazer acreditar que somos capazes de superar qualquer obstáculo, mesmo aqueles que muitas vezes consideramos difíceis e até mesmo impossíveis.

Ao meu Orientador Adriano Mendonça Souza pelos conhecimentos transmitidos, pela paciência e pelo incentivo ao meu trabalho.

Agradeço à professora Roselaine Ruviaro Zanini pela disponibilidade, dedicação e paciência.

Aos meus pais Neusa e Odil pelo cuidado, apoio, amor e carinho que sempre me demonstraram ao longo da vida.

Ao meu querido Ober, pelo companheirismo, incentivo, amor e principalmente pelas palavras confortantes que me deram força para que eu continuasse esta longa caminhada.

Agradeço a minha amiga, Regiane Klidzio que me ajudou a dar o primeiro passo em busca desta conquista.

Por fim com imensa gratidão, agradeço as minhas amigas que fiz durante o mestrado, Meire Mezzomo e Lizandra Salau, por me acolher com carinho, pelas risadas e por compartilhar as mesmas angústias desta etapa da vida.

RESUMO

Dissertação de Mestrado
Programa de Pós-Graduação em Engenharia de Produção
Universidade Federal de Santa Maria

APLICAÇÃO DE SÉRIES TEMPORAIS E REDES NEURAIS EM UM AMBIENTE DE COMPUTAÇÃO EM NUVEM

AUTORA: TATIANA FERNANDA MOUSQUER DOS SANTOS
ORIENTADOR: ADRIANO MENDONÇA SOUZA

Data e Local da Defesa: Santa Maria, 6 de Março de 2014.

A computação em nuvem surgiu para mudar a forma como a computação é oferecida e utilizada. Ao invés de possuir todo o hardware e software necessários para manipular e armazenar seus dados, os usuários apenas necessitam de um mecanismo que acesse a Internet. Com isso, o provisionamento eficiente de recursos computacionais sob demanda é um desafio para atender as necessidades dos usuários. Dessa forma, percebe-se que existe um problema relacionado à necessidade de mecanismos que auxiliem um sistema de gerenciamento de nuvem a manter níveis adequados de qualidade de serviço (QoS) de forma pro-ativa. Nesse contexto, este trabalho faz uma análise comparativa da capacidade de predição de diferentes modelos estatísticos com vistas a definir o mais adequado ao provisionamento de recursos em um ambiente de nuvem. Para isso, foram aplicadas técnicas de séries temporais lineares ARIMA e ARMAX e não lineares baseadas em redes neurais MLP e NARX em um *dataset* de um cluster de computadores da Google. Os resultados de predição de uso de cpu, memória e disco demonstraram que a rede neural NARX consegue predizer com baixo erro os valores esperados, sendo viável a sua aplicação em um mecanismo de provisionamento de servidores em ambientes de nuvem computacional

Palavras-chave: Séries temporais. Redes neurais. ARIMA. ARMAX. MLP. NARX Computação em nuvem.

ABSTRACT

Mastership Dissertation
Post-graduation Program in Production Engineering
Federal University of Santa Maria

APPLICATION OF TIME SERIES AND NEURAL NETWORKS IN AN CLOUD COMPUTING ENVIRONMENT

AUTHOR: TATIANA FERNANDA MOUSQUER DOS SANTOS
ADVISOR: ADRIANO MENDONÇA SOUZA
Date and place of defense: Santa Maria, March 6th, 2014.

Cloud computing has emerged to change the way computing is offered and used. Instead of having all the necessary hardware and software to manipulate and to store their data, users just need a mechanism to access the Internet. So, the efficient provisioning on demand of computational resources is a challenge to comply with the needs of users. Thus, there is a problem related to the lack of an underlying mechanism to assist a cloud management system to maintain acceptable levels of Quality of Service (QoS) pro-actively. In this context, this work makes a comparative analysis of the predictive ability of different statistical models in seeking to define the most suitable for resource provisioning in a cloud environment. In this way, linear time series techniques namely ARIMA and ARMAX and nonlinear ones based on neural networks so-called MLP and NARX were applied on a dataset of a cluster from Google. The prediction results of usage of cpu, disk and memory shown that the NARX neural network can predict with low error the expected values, being feasible for application in a provisioning mechanism of servers in cloud computing environments.

Keywords: Time series. Neurais network. ARIMA. ARMAX. MLP. NARX. Cloud computer.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ADF – Dickey-Fuller Aumentado (*Augmented Dickey-Fuller test statistic*)
AIC – Critério de Informação Akaike *(Akaike Information Criterion)*
AR – Modelo Autorregressivo
ARI – Modelo Autorregressivo Integrado
ARMA – Modelo Autorregressivo e de Médias Móveis
ARMAX – Modelo Autorregressivo com Média Móvel e entradas externas
ARIMA – Autorregressivo Integrado de Médias Móveis *(Auto Regressive Integrated Moving Averages)*
BIC – Critério de Schwarz *(Bayesian Information Criterion)*
CPU – Unidade central de processamento *(Central Processing Unit)*
d – Número de diferenças
DF – Teste de Raiz Unitária *Dickey-Fuller*
F – Estatística "F" de Snedecor
FAC – Função de Autocorrelação
FACP – Função de Autocorrelação Parcial
IC – Intervalo de Confiança
i.i.d. – Independentes e Identicamente Distribuídas
IMA – Modelos de Médias Móveis Integradas
KPSS – Teste de Raiz Unitária proposto por *Kwiatkowski-Phillips-Schimidt-Schin*
LIC – Limite Inferior de Controle LSC – Limite Superior de Controle MA – Modelo de Médias Móveis
MLP – *Multilayer Perceptron*
NARX - Nonlinear AutoRegressive model with eXogenous inputs
p – Número de parâmetros autorregressivo
q – Número de parâmetros de médias móveis
Q(k) – Estatística Q de Ljung-Box
RB – Ruído Branco
RNAs – Redes Neurais Artificiais
$\chi 2$ – Distribuição Qui-quadrado

SUMÁRIO

1. INTRODUÇÃO

A partir do avanço da sociedade, surgiram novas tecnologias como computadores portáteis, *tablets* e celulares. Com isso serviços como e-mail, acessos à internet e vendas on-line tornaram-se essenciais na vida diária das pessoas. Para que se tenham esses produtos à nossa disposição existe uma grande infraestrutura envolvida por parte de empresas de tecnologia. Um exemplo é a empresa *Google*, conhecida mundialmente por seus diversos serviços e produtos como *Gmail*, *Google Docs*, *Google Maps*, etc.

Esse tipo de organização tem a tecnologia da informação *(hardware e software)* como sua base principal para gerir os negócios. Conforme Alecrim (2011), "a tecnologia da informação (TI) pode ser definida como o conjunto de todas as atividades e soluções providas por recursos computacionais que visam permitir a obtenção, o armazenamento, o acesso, o gerenciamento e o uso das informações".

Para obter um custo menor, as empresas de Internet utilizam a computação em nuvem. Nesse paradigma computacional, os dados dos usuários podem ser acessados a partir de qualquer dispositivo (celular, computador, *tablet*), sem ter que se preocupar com instalação e configuração de *software* ou ainda sem a necessidade de saber o local em que estão armazenados os seus dados.

Conforme afirma Taurion (2009), a computação em nuvem é um termo para descrever um ambiente de computação baseado em uma imensa rede de servidores. Na maior parte das vezes tais servidores são virtuais, porém também podem ser físicos. Entende-se por servidor um ou vários computadores conectados à rede (internet) que oferece aos clientes uma grande variedade de recursos. De acordo com Coutinho *et al.*, (2013), "nuvem é uma metáfora para a internet ou infraestrutura de comunicação entre os componentes arquiteturais, baseada em uma abstração que oculta a complexidade da infraestrutura".

Um problema que existe nesse tipo de arquitetura está relacionado à necessidade de escalonamento de novos servidores virtuais. Para Chirigati (2009), as aplicações

desenvolvidas para uma nuvem precisam ser escaláveis, de forma que os recursos utilizados possam ser ampliados ou reduzidos de acordo com a demanda.

Com o aumento cada vez mais crescente da procura dos serviços disponibilizados por essas empresas de tecnologia, o que ocorre é uma sobrecarga nos servidores ocasionando um baixo desempenho em recursos computacionais como armazenamento em disco, uso de processamento de cpu e consumo de memória.

Assim, se forem alocados poucos servidores irão ocorrer gargalos, mas, por outro lado, se forem alocados demais, recursos computacionais que poderiam ser utilizados de forma mais efetiva por outras aplicações serão desperdiçados. Além disso, existe ainda a necessidade que essa alocação de recursos seja efetuada de forma pró-ativa, visando assim, garantir os níveis de qualidade de serviços (QoS – do inglês *Quality of Service*) adequados aos usuários.

Neste contexto, o presente estudo busca demonstrar a viabilidade de um método estatístico que seja capaz de apontar de forma preditiva o escalonamento de recursos computacionais utilizados por servidores em um ambiente de nuvem computacional.

Dessa forma, na busca de tal método, foi aplicada a metodologia de *Box e Jenkins* por meio do modelo ARIMA, um modelo autorregressivo integrado de médias móveis, bem como a aplicação do modelo ARMAX, que é um modelo autorregressivo com média móvel e entradas externas. Além disso, foram utilizadas redes neurais por meio da modelagem de uma rede chamada *multilayer perceptron* – MLP e a outra chamada NARX. Para a aplicação das técnicas envolvidas neste estudo, foram utilizadas três variáveis: cpu, disco e memória, cujo os dados foram retirados de um servidor de internet, que fazia parte de um cluster da empresa *Google*.

As principais contribuições dessa pesquisa são:

- A apresentação de uma metodologia comparativa do uso de séries temporais e redes neurais como métodos preditivos para o escalonamento de recursos computacionais em um ambiente de nuvem;

- A definição de qual modelo estatístico é o mais adequado ao domínio do problema em questão;

1.1. Tema da pesquisa

Pode-se perceber que existe ainda o desafio de como definir, baseado em métodos preditivos, a melhor maneira de utilizar recursos computacionais de forma a evitar gargalos e desperdício de consumo de recursos como memória, processamento de cpu e disco.

Sendo assim, o tema da presente pesquisa é utilizar modelos de séries temporais e redes neurais, para verificar se algum destes métodos é capaz de gerar uma boa previsão de provisionamento destes recursos de forma a possibilitar escalonamento dinâmico de novos servidores, ou seja, reação automática a problemas de desempenho e o gerenciamento automático dos sistemas em nuvem.

1.2. Justificativa e importância da pesquisa

A computação em nuvem ocorreu a partir da necessidade de se reduzir custos e ter maior escalabilidade na infraestrutura. Com isso surgiu a técnica de virtualização, que mudou o modelo de computação para uma consolidação de recursos virtuais em um único recurso físico mais robusto.

Hoje em dia, há casos em que *data centers* inteiros são virtualizados, disponibilizando centenas de máquinas virtuais para permitir uma flexibilidade de gerenciamento, escalabilidade, manutenção, multiplicação de clientes, entre outros.

Dessa maneira, com a expansão dos serviços de computação em nuvem, verifica-se a necessidade de se otimizar os recursos computacionais como o consumo de cpu, disco e memória.

Assim, ao utilizar mecanismos como técnicas estatísticas por meio de séries temporais e redes neurais, pretende-se chegar a um método de predição que possa ser pró-ativo, podendo fazer escalonamento de recursos computacionais, de modo a evitar problemas no desempenho de serviços de computação em nuvem e dessa forma evitar transtornos aos usuários.

1.3. Objetivos

1.3.1. Objetivo geral

Fazer uma análise comparativa de técnicas estatísticas utilizando séries temporais e redes neurais, com vistas a definir qual das técnicas utilizadas possui a melhor capacidade preditiva no provisionamento de recursos computacionais utilizados por um servidor em um ambiente de computação em nuvem.

1.3.2. Objetivos específicos

- Definir qual é o melhor modelo de previsão para as metodologias ARIMA e ARMAX;
- Analisar a capacidade de predição das séries temporais, das redes neurais MLP e NARX.
- Identificar qual dos modelos aplicados no presente estudo possui maior acurácia na predição.

1.4. Delimitação da pesquisa

Desde já se define que o presente trabalho possui como escopo somente aspectos estatísticos envolvidos no provisionamento de recursos computacionais como cpu, disco e memória, em um ambiente de nuvem. Dessa forma será analisada qual das técnicas de predição utilizadas obteve a melhor acurácia.

Assim pretende-se apontar qual método de predição (ARIMA, ARMAX, MLP ou NARX), utilizados na pesquisa pode ser o mais adequado para auxiliar em um melhor desempenho de recursos computacionais em ambiente de computação em nuvem.

1.5. Estrutura do trabalho

Para atingir aos objetivos propostos, a pesquisa foi estruturada em cinco capítulos, descritos a seguir:

O primeiro capítulo apresenta a introdução do estudo, o tema, a justificativa e a importância, os objetivos geral e específicos, a delimitação e a estrutura da pesquisa.

O segundo capítulo, apresenta o referencial teórico que descreve a análise de séries temporais e seus modelos de AR, MA, ARMA, ARIMA e ARMAX. Além disso, serão apresentados conceitos de redes neurais, como a *multiplayer perceptron*, NARX e o algoritmo de retropropagação de erro, o *back-propagation*, assim como apresenta as características da empresa *Google* e conceitos de computação em nuvem.

O terceiro capítulo aborda a metodologia do trabalho e as técnicas utilizadas para seu desenvolvimento.

O quarto capítulo aborda a análise e discussão dos resultados obtidos.

O quinto e último capítulo apresenta as conclusões desse trabalho, além de sugestões para futuras pesquisas.

2. REVISÃO DE LITERATURA

Este capítulo apresenta o referencial teórico que descreve a análise de séries temporais e seus modelos de AR, MA, ARMA, ARIMA e ARMAX. Também apresentará o conceito das redes neurais artificiais MLP e NARX. E por fim mostrará conceitos de computação em nuvem e suas características.

2.1. Análises de séries temporais

Conforme enfoque de Morettin e Toloi (2004), uma série temporal é qualquer conjunto de observações ordenadas no tempo, que devido a essa ordenação cronológica, surge o efeito da autocorrelação entre as observações.

A análise de séries temporais é feita com dois objetivos: analisar o passado, tentando retirar conhecimento útil do mesmo; predizer o futuro, tentando através da análise dos dados, construir um modelo que permita antever a evolução futura da série temporal, segundo Oliveira (2007).

Uma metodologia bastante utilizada na análise de uma série temporal é *Box e Jenkins*. Conforme (Tápia, 2000), a utilização de séries temporais pelo método *Box e Jenkins* é representada pelo conjunto de processos estocásticos ARIMA (do inglês *Autoregressive Integrated Moving Average*) representado pelas letras *(p, d, q)*, em que *p* indica o número de parâmetros autorregressivo, *d* representa o número de diferenças efetuadas na série para que se possa tornar estacionária e *q* indica o número de parâmetros de médias móveis.

De acordo com Gujarati (2000), um processo é estocástico ou aleatório quando se tem um conjunto de variáveis aleatórias ordenadas no tempo. Denomina-se um processo estocástico estacionário aquele no qual sua média e variância são constantes ao longo do

tempo e quando o valor de sua covariância entre dois períodos de tempo depende apenas da distância da defasagem, ou seja:

• Média → $E(Z_t) = \mu$

• Variância → $Var\ (Z_t) = E(Z_t - \mu)^2 = \sigma^2$

• Covariância → $\gamma_k = E[(Z_t - \mu)\ (Z_{t+k} - \mu)]$

Onde, γ_k é a covariância ou autocovariância na defasagem k (covariância entre os valores de Z_t e Z_{t+k}).

Assim uma série temporal é estacionária e sua média, variância e covariância permanecem constantes, independente do período que foram analisadas, ou seja, elas não variam no decorrer do tempo.

Um processo de grande importância das séries temporais estocásticas discretas é a presença de ruído branco. De acordo com Bueno (2008, p.19), "uma sequência {at} é definido como um ruído branco, se cada valor apresentar média zero e variância constante, além de ser independente de qualquer realização da própria série, ou seja, autocorrelação igual a zero".

Os modelos autorregressivos integrados e de médias móveis (ARIMA) são lineares e univariados. Estes modelos podem ser autorregressivos (AR), de médias móveis (MA), autorregressivos e de médias móveis (ARMA), ou ainda processos integrados mistos (ARIMA), dependo do comportamento da série analisada, Andrade (2009).

Segundo Morettin e Toloi (2004), um processo linear, pressupõe que uma série temporal é gerada a partir de um filtro linear (ou sistema linear), cuja entrada é um ruído branco conforme ilustra Figura 1.

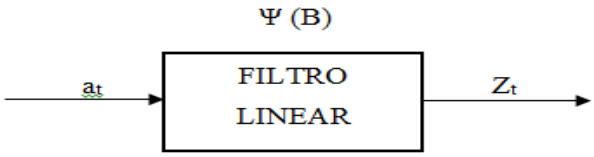


Figura 1 - Filtro linear com entrada at, saída Zt.
Fonte: Morettin e Toloi (2004).

A construção do modelo usando séries temporais segue um ciclo iterativo da metodologia de *Box e Jenkins*, é composto pelas quatro etapas, conforme (Gujarati, 2000):

- Identificação: descobrir os valores apropriados para os parâmetros. Para determinar suas ordens e valores, a função de autocorrelação (FAC) e a função de autocorrelação parcial (FACP) auxiliam nessa tarefa;
- Estimação: estimar os parâmetros dos termos autorregressivos;
- Verificação de diagnóstico: nessa etapa procura-se atestar se o modelo identificado e estimado é adequado, ou seja, se ele descreve adequadamente a série de dados. A forma de verificação comumente utilizada é a realização da análise dos resíduos do modelo;
- Previsão: consiste em realizar a previsão, mas é importante verificar a potencialidade de previsão do modelo.

A Figura 2 representa o fluxograma de um modelo a partir da metodologia *Box e Jenkins.*

Figura 2 - Fluxograma construção modelo Box e Jenkins.
Fonte: (ANDRADE, 2009, p.23).

Ao se modelar uma série temporal se pressupõe a utilização de uma série estacionária, de modo que no período de estimação de seus parâmetros sejam representativos de toda a série que será estimada.

Assim há duas características para os modelos, uma é quando a série já se encontra estacionária, representada pelo modelo ARMA. A outra é quando se faz necessário estacionarizar a série para após aplicar a modelagem, este se denomina ARIMA. Ambos os modelos são designados genericamente por ARIMA *(p, d, q)*. A seguir serão apresentados alguns modelos de séries temporais.

2.1.1. Modelo autorregressivo (AR)

Um modelo ou processo autorregressivo, é caracterizado quando uma variável aleatória está relacionada com os próprios valores passados e com os erros aleatórios. A forma genérica de um modelo autorregressivo, sendo denotado por AR(*p*), onde p indica a ordem do modelo, é definida pela equação (1):

$$\tilde{Z}_t = \varphi_1 \tilde{Z}t-1 + \varphi_2 \tilde{Z}t-2 + \ldots + \varphi_p \tilde{Z}t-p + a_t \qquad (1)$$

Conforme Souza *et al.,* (2011), o modelo será considerado estacionário se $\phi(B)$ convergir para $|B| \leq 1$ como $\varphi(B) = 0$, condição chamada de condição de estacionariedade.

Para a identificação do modelo utiliza-se as funções de autocorrelação (FAC) e autocorrelação parcial (FACP), na qual indicam o tipo e a ordem do modelo respectivamente, de acordo com a Figura 3.

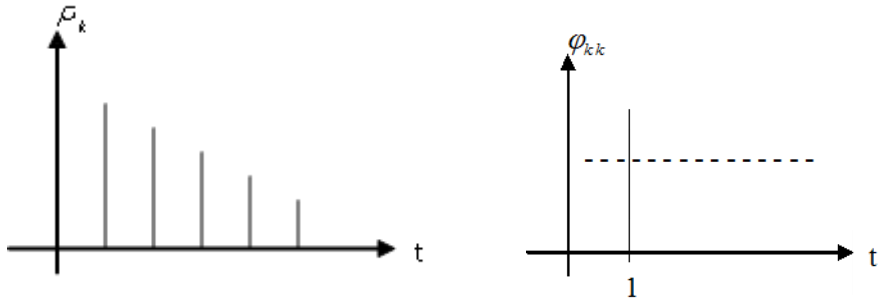


Figura 3 - Comportamento da FAC e FACP de um modelo AR (1).
Fonte: Moretin e Toloi (2004).

Quando o modelo das séries for autorregressivo de ordem 1, a FAC terá seus valores autocorrelacionados e irão decair exponencialmente alternando ou não de sinal, enquanto que a FACP apresenta um único coeficiente de autocorrelação parcial significativo, Junior (2005).

Um modelo autorregressivo nada mais é que uma regressão linear do valor corrente da série, sobre um ou mais valores anteriores da série. A seguir será apresentado o modelo de médias móveis.

2.1.2. Modelo de médias móveis (MA)

O modelo MA é a parte de médias móveis *(Moving Average)*. Este modelo é representado por MA(*q*), ou equivalentemente ao modelo ARIMA (*0,0,q*), sendo q a ordem do modelo, e assume a forma de acordo equação (2) :

$$\tilde{Z}_t = \theta_1 at_{t-1} + \theta_2 at_{t-2} + \ldots + \theta_p at_{t-q} + a_t \qquad (2)$$

A série Z_t é resultado da combinação dos ruídos brancos a_t do período atual e com os períodos anteriores. Segundo o enfoque de Moretin e Toloi (2004) a média móvel para um período escolhido consiste em uma série de médias aritméticas dos períodos passados. Nesses modelos a variável dependente Z_t é escrita como função linear de um número finito de

defasagens dos erros aleatórios não correlacionados. Supondo que *q* tem-se então um processo de médias móveis de ordem *q*.

A identificação do modelo é determinada por meio das funções de autocorrelação e autocorrelação parcial, sendo que a função de autocorrelação fornece a ordem do modelo, de acordo com Figura 4.

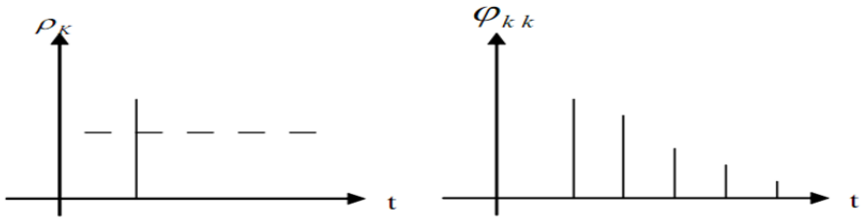


Figura 4 - Comportamento da (FAC) e (FACP) de um modelo MA (1).
Fonte: Moretin e Toloi (2004).

Na Figura 4, a autocorrelação apresenta um corte rápido no 1ag significativo, indicando a ordem do modelo, e a autocorrelação parcial decai exponencialmente.

O modelo MA é construído a partir de uma regressão linear do valor presente da série sobre as perturbações aleatórias de um ou mais valores anteriores da série. Admite-se que estas perturbações são geradas por uma mesma distribuição habitualmente normal de média e desvio-padrão constantes.

2.1.3. Modelo autorregressivo e de médias móveis (ARMA)

Assim como os modelos AR e MA, as aplicações do modelo ARMA, seguem as características de que a série seja estacionária, sem possuir influências como tendência, sazonalidade e ciclo. O modelo ARMA *(p, q)*, se dá pela equação (3):

$$Z_t = \xi + \phi_1 Z_{t-1} + \phi_2 Z_{t-2} + \phi_p Z_{t-p} + a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - \theta_q a_{t-q} \quad (3)$$

Onde: $\phi_1, \phi_2, \ldots \phi_p$ são parâmetros autorregressivos e $\theta_1, \theta_2, \theta_q$, são parâmetros de médias móveis, a_t considera-se um processo puramente aleatório com média zero e variância constante.

Segundo *Box e Jenkins* (1970), um processo ARMA *(p, q)* estacionário tem por caracterização um decaimento exponencial após a defasagem *q*, enquanto a FACP tem o mesmo comportamento após a defasagem *p* na Tabela 1.

Tabela 1 - Identificação dos modelos AR*(p)*, MA*(q)* e ARMA *(p, q)*.

Modelo	FAC	FACP
AR(p)	Decaimento exponencial	Truncada na defasagem p
MA(q)	Truncada na defasagem q	Decaimento exponencial
ARMA (p, q)	Decai exponencialmente se j > q	Decai exponencialmente se j > q

Fonte: (BUENO, 2008, p.43).

Quando existe a característica de estacionariedade, isso garante que um parâmetro estimado no modelo seja representativo para toda a série, o que possibilita a realização de previsão de forma mais assertiva.

2.1.4. Modelo autorregressivo integrado e de médias móveis (ARIMA)

Ao se utilizar séries temporais, se busca um conjunto de observações, as quais devem mostrar um comportamento estável ao longo do tempo, sendo assim busca-se encontrar um conjunto de observações com característica estacionária.

Um dos desafios encontrados por pesquisadores é que a maioria dos problemas encontrados é de não estacionariedade, pois muitas variáveis alteram o seu comportamento

através do tempo, ou seja, ocorre muita variabilidade, alterações bruscas e assim não há uma variância constante e isso faz com que a série observada não seja estável ou estacionária.

A construção da modelagem ARIMA, parte do pressuposto de que as séries temporais envolvidas na análise são geradas por um processo estocástico estacionário, sendo representada a partir de um modelo matemático.

Dessa forma se Z_t não é estacionária, aplica-se a primeira diferença *I(1)* para deixa-lá estacionária, conforme equação (4):

$$Z_t = \Delta Z_t = Z_t - Z_{t-1} \quad (4)$$

Caso seja necessário efetuar duas diferenças *I(2)* para tornar a série estacionária, então Z_t é denominada de ordem (2), representada equação (5):

$$Z_t = \Delta^2 Z_t = \Delta(\Delta Z_t) = \Delta(Z_t - Z_{t-1}) \quad (5)$$

Assim a metodologia proposta por *Box e Jenkins*, aplica-se a casos em que a série apresenta características não estacionárias, sendo necessário tomar uma ou mais diferenças para estacioná-la sendo nomeada *(d)*, a ordem de integração.

A seguir será apresentada a metodologia ARMAX.

2.1.5. Modelo autorregressivo com média móvel e entradas externas (ARMAX)

O modelo ARMAX (*Auto-Regressive with Moving Average and Exogenous inputs)* ou autorregressivo com média móvel e entradas externas, é um modelo similar ao modelo ARIMA, porém este modelo considera a possibilidade de incluir variáveis exógenas como regressores. Sua função é denotada pela equação (6):

$$A(q^{-1})y(k) = B(q^{-1})u(k) + C(q^{-1})a(t) \quad (6)$$

Onde q^{-n} é o operador de atraso, a(t) é o ruído branco e A (q), B (q), C (q) são polinômios, sendo A (q), B (q), C (q) de grande importância na definição do modelo que descreve o comportamento dinâmico desejado do sistema investigado. Os polinômios são definidos a seguir:

$$A(q^{-1}) = 1 + a_1\, q^{-1} + \ldots + a_{n_a}\, q^{-n_a} \quad (7)$$

$$B(q^{-1}) = b_1 + b_2\, q^{-1} + \ldots + b_{n_b}\, q^{-n_b} \quad (8)$$

$$C(q^{-1}) = 1 + c_1\, q^{-1} + \ldots + c_{n_c}\, q^{-n_c} \quad (9)$$

Segundo Moreira *et al.* (2002), o modelo ARMAX permite interpretar as perturbações como um erro de equação:

Em que:

$A(q^{-1})y(k)$ = parte autorregressiva

$B(q^{-1})u(k)$ = sinal de entrada u(k) variável exógena

$C(q^{-1})$ = parte média móvel

a (t) = erro de equação

Kohn (1979, *apud* SILVA, 2011, p.39) afirma que o modelo ARMAX, tem três pressupostos necessários para que seja aplicado:

- As variáveis que compõem o modelo são provenientes de uma população normal multivariada. Os resíduos aleatórios são distribuídos com média zero.
- As variáveis exógenas são estacionárias.
- A variável dependente é estacionária.

Ao contrário do modelo ARIMA, o modelo ARMAX é multivariável, uma vez que a série temporal é explicada pelos valores passados, mas também pelos valores passados de outras variáveis. A utilização deste modelo foi exemplificada por *Philip Hans Franses* em 1991, em um artigo, em que foi estudado o consumo de cerveja nos países baixos.

Ele utilizou para esse estudo como variáveis exógenas a temperatura destes países, os meios de publicidade do produto, os gastos dos consumidores e o preço da cerveja. A variável endógena utilizada para a pesquisa foi o consumo da cerveja.

O estudo deste artigo teve algumas conclusões como: a mudança na média de consumo possivelmente não é causada por alterações fiscais, mas pode ser o reflexo de outras variáveis como estilo de vida, ou problemas de saúde ou até substituições por outros tipos de bebidas. Ainda ficou evidenciado no estudo que os gastos referentes a publicidade de cerveja, não influenciaram a demanda do produto.

Para Franses (1991), o modelo ARMAX possibilita que todas as variáveis sejam consideradas simultaneamente e ainda que se pode haver eventuais inclusões de outras variáveis.

Ao efetuar as defasagens das variáveis regressoras no modelo ARMAX, utiliza-se o método *backward elimination,* este consiste em começar a análise do modelo com vários regressores das variáveis exógenas, ou seja, começa-se com o modelo saturado, assim se retira sucessivamente as interações de ordem mais elevada. Para Guimarães (2006, p.58) há três passos para se realizar o método *backward elimination,* conforme segue:

1ª passo: equacionar uma regressão que contenha todas as variáveis independentes a serem analisadas;

2° passo: calcular o valor de p para cada variável;

3° passo: comparar o maior *p-value* com o nível de confiança α do estudo da seguinte maneira:

- Se *p-value* > *α*, então se remove essa variável, dado que ela não é significativa e os valores de p são recalculados com as variáveis independentes restantes para que este passo se repita com a comparação do próximo maior valor de p encontrado após o recálculo.
- *Se p-value* < *α*, então esse é o modelo de regressão final a ser adotado.

Ainda, para se utilizar ARMAX, tem que haver correlação entre as variáveis exógenas e endógenas, o que possibilita a aplicação deste modelo no presente estudo, em que definiu-se a variável endógena (dependente) memória, sendo as variáveis exógenas (independentes) cpu e disco. A seguir serão comentados os testes estatísticos aplicados neste estudo.

2.2. Testes estatísticos

Existem inúmeros testes para determinar a condição de uma série temporal. Ao iniciar um estudo sobre séries temporais, a primeira condição a ser analisada é se a série possui estacionariedade, para isso a análise gráfica é relevante. De acordo com Bueno (2008, p.97), "a verificação visual de uma série, dificilmente permite distingui-la como sendo de tendência estocástica ou tendência determinística, com isso, pode-se cometer equívocos em relação a esta inspeção".

Por isso somente a análise gráfica não é o suficiente para analisar a série. Assim os testes estatísticos são utilizados para ajudar o pesquisador a estimar e validar o modelo analisado.

A seguir serão comentados alguns testes estatísticos como teste *Dickey-Fuller* (DF), *Dickey-Fuller* aumentado (ADF) e KPSS que auxiliam na verificação da estacionariedade das séries, função de autocorrelação (FAC) e função de autocorrelação parcial (FACP), que

auxiliam na determinação de um resíduo com característica de ruído branco, critérios de informação, dentre outros.

Função de autocorrelação e função de autocorrelação parcial

Uma das estatísticas mais releventes para se efetuar a identificação da ordem dos modelos, são os correlogramas da função de autocorrelação (FAC) e a função de autocorrelação parcial (FACP). Segundo Vasconcellos e Alves (2002, p.213), "estas funções consistem em determinar quais os filtros AR, I e MA compõem o processo gerador da série e também as suas respectivas ordens".

De acordo com Junior (2005), através da FAC e FACP é possível verificar o comportamento da série em relação à sua defasagem *(lag)*, sendo assim o primeiro valor é a correlação entre a observação atual e a observação anterior, ou seja, Z_t, Z_{t-1}, o segundo valor é a correlação entre a observação atual e a observação em 2 espaços de tempo, Z_t, Z_{t-2}, e assim sucessivamente. A FAC é definida por (10):

$$\rho_k = \frac{\gamma_k}{\gamma_0} \tag{10}$$

Onde:

γ_k é a covariância na defasagem (lag) k;

γ_0 é a variância;

E o coeficiente de correlação ρ_k, que varia entre -1 e 1 e o seu estimador é dado pela equação (11):

$$r_k = \frac{\hat{\gamma}_k}{\hat{\gamma}_0} = \frac{\frac{\sum_{t=1}^{n-k}(z_{t-}\bar{z})(z_{t-k} - \bar{z})}{n}}{\frac{\sum_{i=1}^{n}(z_i-\bar{z})^2}{n}} = \frac{\sum_{t=1}^{n-k}(z_t - \bar{z})(z_{t-k} - \bar{z})}{\sum_{i=1}^{n}(z_i - \bar{z})^2} \tag{11}$$

Onde:

z_i é o *i-ésimo* valor da variável aleatória Z,

$\bar{z}$ é a média estimada da variável Z.

Para constatar se ρ_k é diferente de zero e ainda testar a hipótese de que o valor de ρ_k é zero, deve-se calcular um intervalo com (1-α)% de confiança ρ_k, definido pela equação (12):

$$0 \pm z_{(1-\alpha)/2}\sqrt{1/n} \tag{12}$$

Em que:

n é o tamanho da amostra;

$z_{(1-\alpha)/2}$ é o valor da distribuição normal (0;1) para uma significância α.

Assim se ρ_k estiver dentro do intervalo, não se rejeita a hipótese de que seja igual a zero. Caso contrário rejeita-se a hipótese. Normalmente se utiliza um intervalo de confiança de 95%.

Teste de Ljung-Box

É um teste de Box-Pierce adaptado a amostras pequenas, para valores elevados da ordem de auto-correlação. Definido pela equação (13):

$$Q(K) = n(n+2)\sum_{k=1}^{K}\frac{r_k^2}{n-k} \tag{13}$$

Onde, Q(K) é a distribuição qui-quadrado $(x)^2$ com K graus de liberdade, *n* é o tamanho da amostra.

Sendo que as hipóteses são definidas como:

H_0: os resíduos são ruído branco

H_1: os resíduos não são ruído branco

De acordo com Gujarati (2000), caso seja rejeitada a hipótese nula, é aconselhável rejeitar o modelo, pois apresentará uma estrutura de correlação serial significativa nos resíduos.

Teste *Dickey-Fuller*

Esse teste é utilizado para verificar se a série temporal Z_t apresenta-se estacionária. O teste *Dickey-Fuller (DF)* tem as seguintes hipóteses a ser analisadas, na hipótese nula a série é não estacionária e em contrapartida, a hipótese alternativa diz que a série é estacionária.

H0: $\rho = 0$ (A série é não estacionária).

H1: $\rho \neq 0$ (A série é estacionária).

O teste é simples de ser aplicado, basta tomar as primeiras diferenças de Z_t e fazer a regressão em relação à Z_{t-1}, após verifica-se se coeficiente angular é zero ou não. Se o valor do coeficiente for igual à zero, então existe a presença de uma raiz unitária, ou seja, a série é não estacionária. Em contrapartida, se o coeficiente resultar em um valor negativo, a série é estacionária.

Teste *Dickey-Fuller* aumentado

Bueno (2008, p.100) afirma que, "o problema do teste *Dickey-Fuller* (1979) é considerar o erro como um ruído branco. Entretanto, normalmente o erro é estimado de um processo estacionário qualquer. Assim, esse problema pode causar distorções no poder do teste".

Então, para solucionar o problema estima-se o modelo com variáveis autorregressivas, com o intuito de corrigir o desvio do valor correto da estatística, isto é, encontram-se os desvios de Z_t em relação a sua média, para deslocar a distribuição de δ em direção à zero, caso a hipótese nula seja verdadeira.

Na prática, é necessário introduzir tantas variáveis autorregressivas quantas forem necessárias, para que o teste de resíduos não rejeite a hipótese de que se trata de um ruído branco.

Com isso, o teste pode ser feito utilizando-se os mesmo valores críticos encontrados por *Dickey-Fuller*, desde que se realize a correção do modelo, de maneira a considerar as demais variáveis defasadas, que em nada modificam os valores críticos do teste e a interpretação do modelo.

Essa variação do teste é denominada de teste de *Dickey- Fuller* aumentado (ADF), sendo que a sua principal característica é considerar a presença de autocorrelação entre os resíduos. Dessa forma na hipótese nula a série é não estacionária e em contrapartida, a hipótese alternativa diz que a série é estacionária. Assim, as hipóteses são descritas na forma, Wolff (2011):

H0: $\rho = 0$ (A série é não estacionária)

H1: $\rho \neq 0$ (A série é estacionária).

O teste de ADF é expresso conforme (14):

$$\Delta Y_t = \alpha + \beta t + \eta y_{t-1} + \sum_{i=1}^{p-1} \lambda_i \Delta y_{t-1} + \mu \qquad (14)$$

Assim, para realizar a confirmação dos resultados expressos no teste *Dickey- Fuller* (DF), *Dickey-Fuller* aumentado (ADF), utiliza-se então outro teste chamado *Kwaiatkowski, Phillips, Schmidt and Shin (KPSS)*, comentado a seguir.

Teste de *Kwiatkwiski-Phillips-Schimidt-Shin (KPSS)*

Para Bueno (2008), existem testes alternativos para a verificação de raiz unitária já que o teste *ADF* e *DF* são de baixo poder, ou seja, há uma elevada probabilidade de se cometer o Erro Tipo II (não rejeitar a hipótese nula quando ela é falsa).

Para tentar aumentar o poder dos testes de estacionariedade surge então o KPSS, desenvolvido por Kwiatkowski, Phillips, Schmidt e Shin em 1992, tendo como hipótese nula a estacionariedade da série e a hipótese alternativa a não estacionariedade, a ideia dos autores é de utilizar este teste como um complemento ao teste ADF, DF ou de Phillips-Perron.

H0: $\rho = 0$ (A série é estacionária)

H1: $\rho \neq 0$ (A série é não estacionária).

Para entender o teste, se assume que o processo gerador dos dados é:

$$z_t = x_t + u_t \tag{15}$$

Em que x_t é um passeio aleatório, $z_t = z_{t-1} + u_t, u_t \sim$ *i.i.d.* $(0, \sigma^2)$ e u_té um processo estacionário, Bueno (2008, p.11). Assim, após analisada a etapa de verificação da estacionariedade da série é analisado o diagnóstico dos resíduos (erros).

Diagnóstico de resíduos

Um dos pressupostos de uma série temporal é assumir que os erros sejam um ruído branco, ou seja, a série tem que ter média zero, variância constante e não apresentar correlação serial:

$Z_{t} = a_{t}$

$E(Z_{t}) = 0$

$Var(Z_{t}) = \sigma^{2}$

$Cov(Z_{t,} Z_{t-k}) = (0)$

Para (BUENO, 2008, p. 68), "estimado o modelo, deve-se verificar o ajuste dos resíduos. Isso significa verificar se a FAC e FACP dos resíduos estimados revelam-se sem qualquer memória. Se a hipótese nula é rejeitada, implica que há informação ainda não captada pelo modelo e com isso obteremos previsões pobres".

Caso o modelo estimado não apresente a característica de ruído branco, então deve se descartar esse modelo e testar outras possibilidades até encontrar um modelo cujo os resíduos se comportem como tal.

Na próxima seção será comentado o critério de informação, este é utilizado para definir o número de parâmetros do modelo de série temporal.

Critério de informação

Várias áreas do conhecimento, entre elas a engenharia, utilizam constantemente a modelagem de dados. Porém um dos problemas encontrados ao se modelar um conjunto de dados é sobre qual a ordem que deve ser utilizada no modelo. Sempre se busca utilizar a menor ordem possível para que se tenha o modelo mais adequado aos dados.

Contudo, muitas vezes é difícil adequar os dados a algum modelo, pois esses podem apresentar alguns distúrbios ou ainda podem não se ajustar a algumas características dos modelos a serem escolhidos. Então uma das soluções para esse problema seria aumentar a ordem do modelo, com isso, poderia haver uma maior adequação aos dados.

"Assim é utilizado o critério de informação que é uma maneira de encontrar o número ideal de parâmetros de um modelo", Bueno (2008, p.46). O objetivo do critério de informação

é minimizar uma função baseado nos resíduos e penalizada pelo número de regressores incluídos no modelo.

Na prática é comum dois ou mais modelos gerarem resíduos cujos testes indicam ser ruído branco. Assim, a escolha do melhor modelo será o modelo mais parcimonioso, sendo que os resíduos sejam os menores possíveis.

Segundo Bueno (2008, p. 47), "há três principais critérios de informação. A estatística de *Schwarz*, denotada por BIC (*Bayesian Information Criterion*) e descrita pela equação (16)":

$$BIC(p,q) = \ln \hat{\sigma}^2 + n\frac{\ln T}{T} \qquad (16)$$

Em que:

n = *p* + *q*, se o modelo não apresenta constante;

n = *p* + *q* +*1*, se há constante no modelo e

T é o número de observações.

$$\sigma^2 = \frac{\sum_{t=1}^{T} \hat{a}_t^2}{T}$$

A estatística de *Akaike*, denotada por AIC (*Akaike Information Criterion*), é dada pela equação (17):

$$AIC(p,q) = \ln \hat{\sigma}^2 + n\frac{2}{T} \qquad (17)$$

Finalmente, a estatística de Hannan-Quinn, HQ, dada por (18):

$$HQ(p,q) = \ln \hat{\sigma}^2 + n\frac{2}{T}\ln\ln T \quad (18)$$

Assim, o modelo mais parcimonioso deverá gerar menos imprecisão de estimativas, justamente por apresentar um menor número de parâmetros do que um modelo com maior número de parâmetros.

2.3. Redes Neurais Artificiais

As Redes Neurais Artificiais, RNAs, representam uma subárea da Inteligência Artificial que se baseia na estrutura do cérebro humano utilizando funções matemáticas não-lineares e que possui a capacidade computacional de adquirir, armazenar e utilizar o conhecimento, Araújo *et. al.;* (2005).

O desenvolvimento de redes neurais pode parecer recente, porém este campo foi estabelecido antes da era dos computadores. O primeiro neurônio artificial foi produzido em 1943 pelo neurofisiologista *Warren Mc-Culloch* e o matemático *Walter Pits*, mas as tecnologias disponíveis na época não permitiram grandes avanços.

Porém, nas últimas décadas o campo ressurgiu e as RNAs têm sido aplicadas com sucesso em várias áreas como: finanças, medicina e engenharia. As redes neurais podem ser utilizadas em diferentes áreas como predição, classificação e controle.

As RNAs são compostas por unidades de processamento simples (neurônios) que computam as funções matemáticas. Estas unidades podem estar dispostas em estruturas interligadas por um grande número de conexões (sinapses). Geralmente estas conexões estão associadas à pesos, os quais armazenam o conhecimento representado no modelo, com isso todo o conhecimento adquirido por uma RNA está nos pesos de suas sinapses, Marangoni (2010).

Para o ajuste dos pesos é necessária uma etapa onde a rede aprende a lidar com os dados referentes ao problema em questão. Tal processo é definido como treinamento. Os dois tipos de treinamento mais conhecidos são: aprendizado supervisionado e não supervisionado. Ressalta-se que o uso de um ou outro tipo de aprendizagem é definido de acordo com o modelo de RNA usado.

Basicamente, pode-se descrever assim:

- Aprendizado supervisionado: o processo de aprendizagem ocorre com o auxílio de um "professor". "Em termos conceituais podemos considerar um professor como tendo conhecimento sobre o ambiente, com este conhecimento sendo representado por um conjunto de exemplos de entrada e saída" (HAYKIN, 2001, p.88). Em outras palavras, nesse tipo de aprendizagem são informados dois conjuntos de dados para a rede: um contendo as entradas e outro as saídas esperadas para tais entradas. É papel do algoritmo de treinamento fazer os ajustes internos dos pesos da rede para que ela seja capaz de fornecer saídas o mais próximo possível das saídas esperadas.
- Aprendizado não-supervisionado: neste tipo de aprendizagem não há um "professor" para supervisionar o processo de aprendizagem da rede. "Uma vez que a rede tenha se ajustado às regularidades estatísticas dos dados de entrada, ela desenvolve a habilidade de formar representações internas para codificar as características da entrada, desse modo, cria automaticamente novas classes", (HAYKIN, 2001, p.90).

Outro aspecto importante em uma rede neural está ligado a sua topologia. A topologia de uma rede é definida pela forma na qual os neurônios estão organizados e interconectados, ou seja, o número de camadas, número de neurônios que compõem cada camada, tipos de conexões entre os neurônios e a topologia da rede. Além disso, devem ser considerados o valor da taxa de aprendizagem (constante no intervalo [0,1] que indica a velocidade do aprendizado) e o uso ou não de variáveis de atraso.

2.3.1. Rede neural m*ultilayer perceptron* – MLP

Dentre os vários tipos de redes neurais, a rede multicamadas, também conhecida como *multilayer perceptron,* é uma das mais utilizada em diferentes domínios e áreas de aplicações (MIRJALILI, 2011; KARLIK e OLGAC, 2002). Uma MLP é constituída por uma camada de entrada, uma ou mais camadas ocultas e uma camada de saída. A Figura 5 mostra a estrutura de uma rede neural MLP *(Multi Layer Perceptron).*

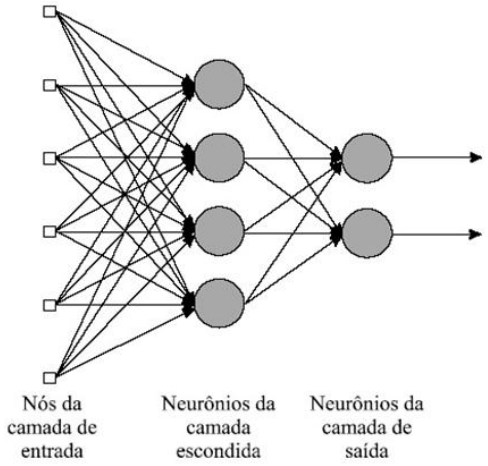


Figura 5 - Estrutura de um *Perceptron* de múltiplas camadas - MLP.
Fonte: HAYKIN (2001).

A rede MLP é composta por no mínimo três camadas, a de entrada, a(s) oculta(s) e a camada de saída:

- Camada de entrada: composta por neurônios que recebem os valores de entrada.
- Camada(s) Oculta(s): composta por neurônios que dividem o problema em outros problemas menores, sendo que estes são linearmente separáveis.
- Camada de saída: composta por neurônios computacionais faz a rotulação ou mapeamento dos dados.

Esta rede é caracterizada por utilizar o *back-propagation,* algoritmo de propagação retroativa de erro (que será apresentado na seção 2.3.3). O processo de aprendizado ocorre

através dos ajustes dos pesos das sinapses da rede de acordo com o erro existente entre o valor real e o predito.

A seguir será comentada a outra topologia de RNA utilizada neste estudo, NARX.

2.3.2. Rede neural *nonlinear autoregressive model with exogenous inputs* - NARX

A rede neural não linear NARX *(nonlinear autoregressive model with exogenous inputs),* é similar a uma MLP, porém a entrada da rede consiste na própria saída realimentada com atrasos de tempo e uma entrada exógena com atrasos.

A arquitetura da rede NARX se dá conforme (19):

$$y(n) = f[y(n-1) \ldots . y(n-dy), u(n-1) \ldots . u(n-du+1)] \quad (19)$$

Em que: "f(.) é uma função não linear, geralmente desconhecida e u(n) e y(n) correspondem à entrada e saída no tempo n, du > 0 e dy > 0, du $\leq$ dy são as ordens da memória de entrada e memória de saída. Quando esta função é aproximada por uma rede com múltiplas camadas, a topologia que resulta é chamada NARX", Campos (2013, p.3). Conforme ilustra Figura 6.

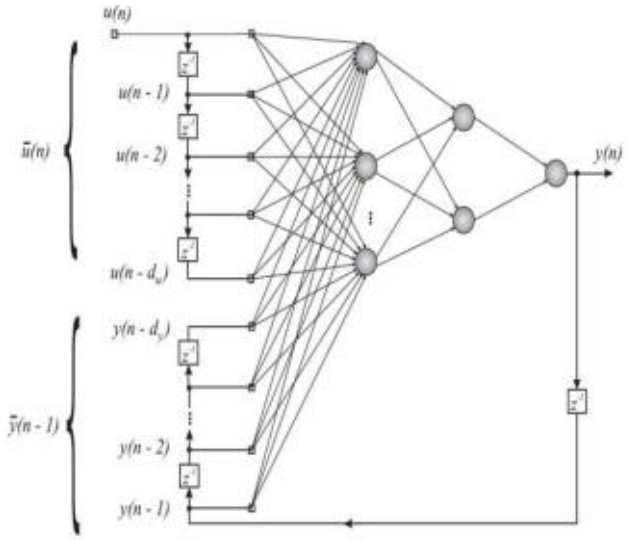


Figura 6 - Topologia da rede NARX.
Fonte: (Campos, 2013, p.3).

A rede neural NARX é utilizada para prever os valores futuros de uma série temporal y(t) a partir de valores passados desta série e os valores passados de outra série temporal x(t). Este tipo de rede pode ser aplicado em variáveis econômicas como a verificação do PIB, por exemplo, bem como pode ser utilizado em séries temporais de sistemas dinâmicos, motivo pelo qual foi utilizada esta topologia de RNA no presente estudo.

Na literatura houve alguns estudos com o intuito de verificar o comportamento de aplicações com um banco de dados, conforme Amani e Kihl (2012) “utilizou-se a topologia NARX para verificar a análise e desempenho, um passo à frente, de um servidor de banco de dados, os resultados foram promissores”.

2.3.3. Algoritmo *Back-propagation*

Os algoritmos de aprendizado variam de acordo com o tipo de rede utilizado. Um algoritmo basicamente difere do outro pela forma com que os pesos das sinapses da rede são ajustados. O mais popular destes algoritmos é o *back-propagation.*

O *back-propagation* é um algoritmo de treinamento supervisionado desenvolvido ao longo dos anos 70 e 80 por diversos pesquisadores, tornando-se o algoritmo mais popular no treinamento de redes neurais. Durante o treinamento de uma rede utilizando *back-propagation*, a rede opera utilizando sequência de dois passos, Carvalho (2013):

Primeiro: um padrão é apresentado à camada de entrada da rede. A atividade resultante flui através da rede, camada por camada, até que a resposta seja produzida pela camada de saída.

Segundo: o resultado da saída obtida é comparado à saída desejada. Se esta não estiver correta, o erro é calculado. O erro é propagado a partir da camada de saída até a camada de entrada, e os pesos das conexões das camadas internas vão sendo modificados conforme o erro é retropropagado. Conforme Artero (2009), o erro obtido na camada de saída é retro-propagado pela rede em direção à camada de entrada.

O cálculo do erro na saída da rede é dado pela soma dos quadrados dos erros instantâneos em cada nó de saída da rede sendo representado pela equação (20):

$$E(n) = \frac{1}{2}\sum_{k=1}^{M}(d_k\,(n) - \, y_k(n))^{\,2} \qquad (20)$$

As redes neurais artificiais MLP e NARX, utilizam em seu treinamento a aprendizagem supervisionada e o algoritmo *back-progation*.

2.3.4. Função de ativação

Para que as sinapses de uma RNA ocorram é necessário que em determinado momento um neurônio seja ativado ou desativado de acordo com uma função que defina o limiar de

ativação do mesmo. Tal função é chamada de função de ativação ou função de limiar (do inglês *threeshold function).*

- As funções de ativação se diferenciam uma das outras pela forma como ocorre a transição de estados (de uma forma mais suave ou mais brusca). Elas são assim descritas por Silva (2009):

- Função linear: repete o sinal de entrada do neurônio na sua saída. É bastante utilizada nos neurônios da camada de saída da rede. No software *Matlab* essa função é implementada através da função *Purelin.* Conforme equação (21) e Figura 7:

$$y = f(x) = x \tag{21}$$

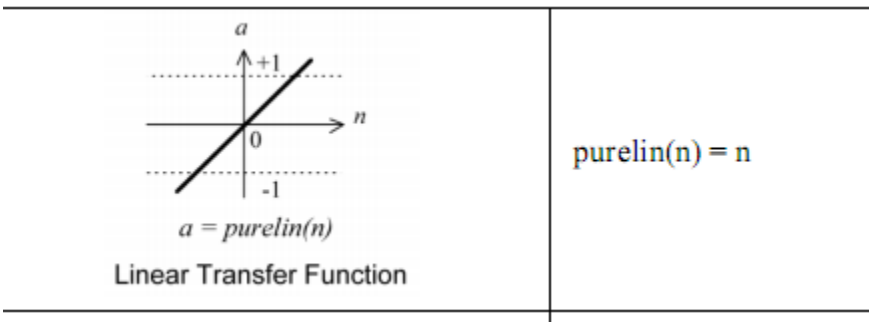


Figura 7 - Função de ativação *purelin.*
Fonte: (MARANGONI, 2010, p.73)

- Tangente *Sigmoidal:* gera as saídas entre -1 e 1, enquanto a entrada do neurônio vai de negativo até infinito. No *Matlab* ela é implementada pela função *Tansig*. Conforme equação (22) e Figura 8:

$$y = f(n) = \frac{e^n - e^n}{e^n + e^n} \tag{22}$$

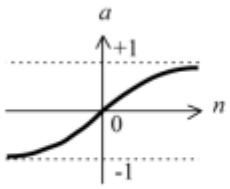
 *a = tansig(n)* Tan-Sigmoid Transfer Function	tansig(n) = 2/(1+exp(-2*n))-1

Figura 8 - Função de ativação *tansig.*
Fonte: (MARANGONI, 2010, p.73)

A MLP utilizada neste trabalho fará uso da função linear na sua camada oculta e da função tangente *sigmoidal em sua camada de saída*. Já a rede NARX usa em sua camada oculta a função de transferência sigmoide, e em sua camada de saída a função de transferência linear.

2.3.5. Critério de avaliação

Para verificar a acurácia dos modelos de previsão descritos nesta pesquisa, será utilizado como critério de avaliação a função da raiz quadrada do erro quadrático médio (RMSE). O motivo de escolha do RMSE ao invés do erro médio quadrático (MSE) deve-se ao fato do RMSE representar o erro na mesma unidade da variável de saída e não ser sucinto a interpretações errôneas (por exemplo, o dobro do RMSE significa que o erro é duas vezes maior, porém o dobro do MSE não significa que o erro é duas vezes maior). A equação do RMSE é representada equação (23):

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(S_i - O_i)^2} \quad (23)$$

A seguir serão comentados os conceitos e características referentes a computação em nuvem.

2.4. Computação em nuvem

A computação em nuvem é um paradigma, que surgiu a partir da necessidade de se reduzir custos e de se ter maior escalabilidade na infraestrutura. Com isso, empresas de tecnologia utilizam a chamada computação em nuvem, que proporciona a redução de custos com infraestrutura e assim se consegue ter despesas de capital reduzidas e consequentemente dispender esses recursos para aumentar a sua capacidade interna.

A infraestrutura do ambiente de computação em nuvem normalmente é composta por um grande número, centenas ou milhares de máquinas físicas ou nós físicos de baixo custo, conectadas por meio de uma rede, Coutinho et al.(2013). Na Figura 9 pode-se observar a arquitetura de computação em nuvem.

Figura 9 - Estrutura que envolve o conceito de computação em nuvem.
Fonte: MARY (2012).

Ambientes de computação em nuvem são compostos por servidores virtuais, solução que diminui os custos de recursos computacionais. Uma vez, que a enorme quantidade de informações geradas pelas empresas e usuários, torna necessária a procura de soluções que visam oferecer serviços de informação que tratem um grande volume de dados com um menor custo.

A virtualização é uma técnica para ocultar características físicas de recursos computacionais, da forma na qual os sistemas, aplicações e usuários interagem com esses recursos. Esta técnica utiliza um *hardware* para dividir os recursos desse computador em vários computadores virtuais.

De acordo com Carissimi (2008, p.173) "virtualização é a técnica que permite particionar um único sistema computacional em vários outros denominados de máquinas virtuais. Cada máquina virtual oferece um ambiente completo muito similar a uma máquina física. Com isso, cada máquina virtual pode ter seu próprio sistema operacional, aplicativos e serviços de rede (Internet)".

Esta metodologia tem proporcionado às empresas de tecnologia, o desenvolvimento de aplicações que compartilham a mesma infraestrutura de *hardware* e *software*, contribuindo assim com os serviços de computação em nuvem.

Mas para assegurar a funcionalidade dos serviços e a infraestrutura da computação em nuvem, sem que ocorram falhas nos serviços, existem os *data centers*.

Esses são ambientes planejados para garantir dentre outras coisas: que a temperatura do local em que se encontram os servidores fique bem refrigerada; que não falte energia elétrica para que os servidores não parem de funcionar; garantir monitoramento e segurança do local 24 horas por dia; assim como deve haver redundância nos serviços.

A Figura 10 demonstra estrutura física de um *data center*.

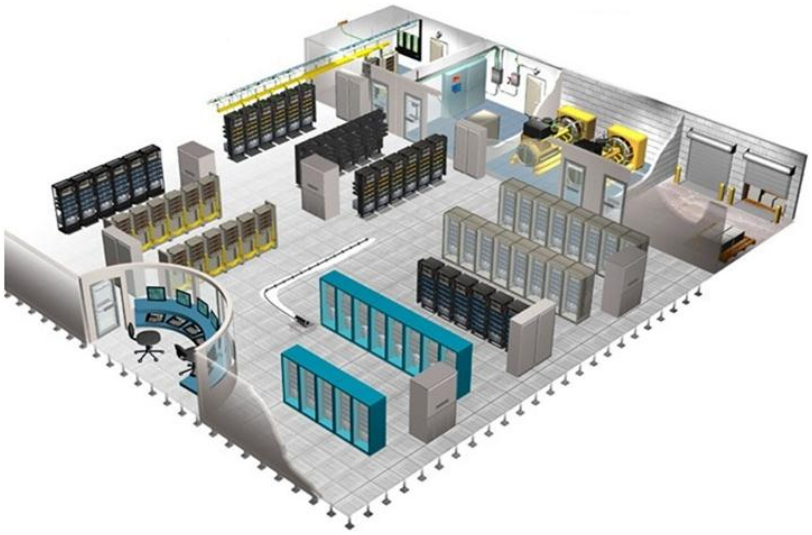


Figura 10 - Ambiente de um *data center* com vários servidores de internet.
Fonte: *Google*.

Segundo a literatura, a computação em nuvem é composta por três diferentes camadas: *SaaS, PaaS e IaaS,* as quais são representadas na Figura 11 e descritas mais abaixo:

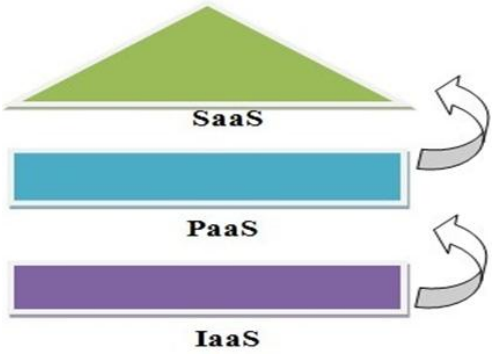


Figura 11 - Camadas computação em nuvem.
Fonte: elaborada pela autora.

SaaS (Software as a Service) – é a camada de mais alto nível, em que as aplicações são oferecidas aos usuários através de serviços ofertados por provedores de internet, sendo acessíveis em qualquer horário e local através de aplicações como o browser. Exemplos de

implementações SaaS são os serviços prestados pela *Google* (*Google Docs, Google* Agenda, *Google Mail*).

PaaS (Plataform as a Service) – é a camada intermediária, sendo que é a capacidade disponibilizada pelo provedor para que o usuário desenvolva suas aplicações que serão hospedadas e executadas em nuvem. Atualmente serviços como *Google AppEngine* e *Microsoft Azure* são exemplos de PasS.

IaaS (Infraestructure as a Service) - encontra-se em um nível mais baixo. Nesta camada o provedor fornece aos usuários uma infraestrutura de processamento e armazenamento, sendo que o cliente não tem acesso à estrutura física do serviço, mas através de ferramentas de virtualização consegue ter controle sobre os sistemas operacionais, aplicações e armazenamento. Exemplo da utilização deste serviço é *Amazon EC2*, *Google Cloud Storage*.

A camada *IaaS* é de extrema importância, pois esta serve de base para a prestação de serviços pelas outras camadas. Por este motivo o presente trabalho possui como objetivo investigar um método de predição que seja capaz de ser utilizado como técnica básica para um mecanismo de provisionamento de servidores, sendo que este fica na camada *IaaS.*

A computação em nuvem possui algumas características, dentre elas:

Elasticidade: oferece uma flexibilidade caso haja a necessidade de alterar a configuração de recursos computacionais, escalonando recursos tanto em nível de *hardware* quanto de *software* para atender a demanda dos usuários.

O objetivo da elasticidade é criar um ambiente que se adapte às diferentes cargas de trabalho que são impostas aos servidores. Taurion (2013), afirma que elasticidade é a capacidade do ambiente computacional da nuvem aumentar ou diminuir de forma automática os recursos computacionais demandados e provisionados para cada usuário.

Para Armbrust *et. al.,* (2009), a qualidade do serviço prestado na nuvem é um dos aspectos mais importantes, essa qualidade depende da capacidade da nuvem se auto gerenciar

para assim disponibilizar aos seus usuários recursos necessários de acordo com a demanda, para alguns autores dizem que esta característica chama-se elasticidade.

A escalabilidade de recursos, conforme Schubert (2013) pode ser facilmente efetuada por um operador humano, porém esta opção deixou de ser prática devido ao crescente tamanho das nuvens e o compartilhamento da infraestrutura por várias aplicações e consumidores.

No entanto uma das principais dificuldades encontradas na computação em nuvem é a falta de ferramentas para previsão de provisionamento dinâmico de recursos em servidores, como armazenamento em disco, uso de processamento de cpu e memória. Para Lori (2014, p.7) “objetivo do provisionamento dinâmico é utilizar os recursos disponíveis da forma mais eficiente possível”.

Assim o problema encontrado é que ao aumentar a demanda dos serviços, os servidores podem se sobrecarregar e causar transtornos aos usuários como lentidão no acesso, ou até mesmo deixar os serviços inoperantes, situações essas que impactam diretamente na qualidade de serviços oferecidos por empresas de Tecnologia.

Diante deste contexto existem alguns autores interessados nesta problemática como Coutinho *et al*.(2013), que sugerem diferentes maneiras de se trabalhar análise de desempenho em elasticidade em ambientes de computação em nuvem. Listando aspectos relacionados as formas de realizar análise de desempenho, ferramentas, *benchmarks*, cargas de trabalho, métricas e tendências de pesquisa em elasticidade em computação em nuvem, com aspectos de análise de desempenho.

Em Rego *et al.*, (2013) são descritas estratégias baseadas na criação de máquinas virtuais e alocação dinâmica de recursos, para atender os requisitos de SLA e garantir a qualidade de serviço. A solução aproveita a arquitetura *FairCPU* que tem o objetivo de ajustar dinamicamente a quantidade de cpu conforme limites pré-estabelecidos pelo usuário.

Conforme Sladescau (2012) é proposto um framework chamado de *Event Aware Prediction (EAP)* o qual a proposta é lidar com rajadas de sobrecarga de um ambiente de

nuvem de forma a manter os indicadores de Qualidade de níveis de serviços - QoS em níveis aceitáveis.

Já o trabalho de Khan (2012) apresenta um método baseado em Modelos Ocultos de *Markov* (HMM, do inglês *Hidden Markov Modeling*) para caracterizar correlações temporais em *clusters* de servidores e predizer variações nos padrões de carga do ambiente.

Como se pode imaginar, o provisionamento de recursos é um dos aspectos mais importantes da computação em nuvem, e um dos maiores desafios também.

Dessa forma se percebe que nenhuma dessas iniciativas é capaz de proporcionar de forma preditiva indicativos de provisionamentos de recursos em servidores através de configurações de recursos como cpu, disco e memória, necessários para se manter a qualidade dos produtos disponibilizados por empresas de Tecnologia.

Com isso verifica-se uma necessidade de uma técnica que seja capaz de ser utilizada como base para um mecanismo de provisionamento de recursos de servidores em nuvem. Assim, este trabalho aborda uma solução viável para alocar o consumo de cpu, disco e memória, a fim de que a qualidade dos serviços oferecidos aos clientes-usuários seja mantida.

2.5. Comentários gerais do capítulo

Neste capítulo inicialmente, abordou-se no referencial teórico as análises de séries temporais e seus modelos de AR, MA, ARMA, ARIMA e ARMAX. Após foram descritos alguns testes estatísticos como: os testes de raiz unitária, os critérios de ajuste e o diagnóstico dos resíduos.

Logo após apresentou-se o conceito de redes neurais, como a rede *multilayer perceptron* e seu algoritmo de retropropagação de erro, o *back-propagation*. E o critério de avaliação da rede, raiz quadrada do erro quadrático médio. A MLP será o tipo de rede utilizada no trabalho devido a sua capacidade já comprovada por pesquisas em diferentes

áreas. E a rede NARX que pode ser utilizado em séries temporais de sistemas dinâmicos. Por fim apresentou-se o conceito e a infraestrutura que envolve esse serviço de internet como as camadas *PaSS, SaSS, IaSS*. Bem como as características de uma arquitetura de computação em nuvem.

A metodologia empregada neste estudo é o assunto a ser abordado no próximo capítulo.

3. METODOLOGIA

Este capítulo irá apresentar a metodologia empregada na análise de um *dataset* disponibilizado pela *Google.* Tal *dataset* representa os dados de processamento de um cluster de servidores coletado ao longo de um período de tempo. Serão empregados diferentes tipos de técnicas estatísticas na busca de encontrar àquela capaz de predizer com maior acurácia valores que caracterizam estados futuros de consumo de cpu, disco e memória de um servidor da empresa *Google.*

3.1. Características da empresa em estudo

A empresa *Google* começou em 1996 como um projeto de pesquisa dos estudantes de doutorado, Larry Page e Sergey Brin. Sendo fundada em 1998, cuja missão é organizar as informações do mundo e torná-las mundialmente acessíveis e úteis *Google* (2013).

Originou-se com a ideia de se ter um motor de busca por informações que pudesse ajudar a organizar páginas na internet. Foi uma das empresas pioneiras nos serviços de internet a utilizar a arquitetura de computação e nuvem.

Hoje é uma empresa multinacional que hospeda e desenvolve vários serviços e produtos, todos baseados na internet como redes sociais, sistemas de anúncios e serviços online. Os seus serviços são executados através de mais de um milhão de servidores em *data centers* ao redor do mundo.

Por mês são efetuadas 100 bilhões de pesquisas no site da *Google* e o número de sites varridos diariamente pela empresa para atualização da base de dados usada nas buscas é de 20 bilhões. 60 trilhões de endereços da internet já foram catalogados pelo *Google* e 100 petabytes é o volume de dados nos servidores de busca da *Google*, Grego (2013).

O rápido crescimento da *Google* desde sua incorporação culminou em uma cadeia de produtos, aquisições e parcerias que vão além do núcleo inicial que era o motor de buscas. A empresa oferece *softwares* online, como o *software* de e-mail, através do *Gmail*, e ferramentas de redes sociais, incluindo *Google+* e também o *Google Drive*, serviço de armazenamento e sincronização de arquivos.

Dentre os produtos da *Google* existem os aplicativos como o navegador *Google* Chrome, o *Google Talk* que envia e recebe mensagens instantâneas. Também há o produto chamado *Google Maps*, que é um serviço de pesquisa e visualização de mapas e imagens de satélite de todos os lugares do planeta. O *You Tube*, um site que permite aos seus usuário carregar e compartilhar vídeos. Notavelmente, a *Google* também lidera o desenvolvimento do sistema operacional móvel para *smartphones Android*, usado em celulares, segue a logo da empresa e seus serviços Figura 12.

Figura 12 - Logomarca da Empresa *Google* e seus principais serviços: You Tube, motor de buscas *Google*, *Google+*, *Google Drive*, *Google Maps* e *Adroid.*
Fonte: FELIX (2013).

Atualmente a Google é o website mais visitado e sua marca está entre as mais poderosas no mundo. Alguns números expressivos referente a empresa Google, segundo Felix (2013), são listados a seguir:

- *Google Maps*: possui aproximadamente 1 bilhão de usuários ativos;

- *You Tube*: todo mês o You Tube recebe cerca de 1 bilhão de visitantes únicos;
- *Android:* seu sistema operacional para dispositivos móveis, já teve mais de 900 milhões de aparelhos ativados;
- *Google Chrome:* Mais de 750 milhões de pessoas usam o navegador Google Chrome;
- *Gmail:* utilizado por mais de 450 milhões de pessoas;
- *Blogger (blogspot)*:A plataforma para blogs recebe mais de 300 milhões de visitas por mês;
- *Google Translate:* possui cerca de 200 milhões de usuários ativos mensalmente;
- *Google+:* conta com 190 milhões de internautas que fazem uso frequente dele publicando em suas *timelines.*

A empresa conta com aproximadamente 30 mil colaboradores e possui mais de 70 escritórios, localizados em mais de 40 países ao redor do mundo. Sua sede fica na cidade de *Mountain View,* estado da Califórnia, nos Estados Unidos.

3.2. Dados utilizados

Os dados utilizados na presente pesquisa são disponibilizados publicamente pela empresa *Google*[1]. O objetivo da empresa ao disponibilizar esses dados é tornar visível a complexidade da carga de trabalho dos servidores, como a variedade de tipos de trabalho, os tipos de *hardwares* necessários para a qualidade dos serviços, bem como o alto consumo de recursos.

A base de dados utilizada possui um conjunto de variáveis representando a carga de trabalho *(workload)* de um cluster de 11 mil computadores da *empresa*. Essas variáveis são

[1] *Dataset* disponibilizado publicamente pela empresa *Google* (http://code.google.com/p/googleclusterdata/).

agrupadas em conjuntos de dados (*datasets*) que representam diferentes tipos de informações e originalmente possuem mais de 40GB compactados.

Os *datasets* disponibilizados estão divididos em diferentes arquivos no formato csv (com valores separados por vírgulas) que contém: dados de cada uma das máquinas do cluster *(Machine)*, dados que caracterizam as tarefas executadas pelas máquinas *(Jobs and Tasks)*, restrições que definem os recursos necessários para a execução de tarefas *(Task contraints)* e dados referentes aos recursos utilizados na execução das tarefas *(Resource usage).*

Cada um desses arquivos possui um conjunto de valores que representam diferentes variáveis. Para o desenvolvimento deste trabalho foi usado o arquivo de recursos utilizados para a execução de tarefas *(task resource usage)* que é composto pelas seguintes variáveis:

- *Timestamp:* início e fim do tempo de execução das tarefas.
- *Job ID:* número identificador do trabalho executado.
- *ID* da máquina: número identificador da máquina disponível no cluster.
- Uso de cpu: mínimo e máximo de recurso usado a cada 1segundo.
- Uso de memória: medição de uso de memória, o número de usuário por páginas acessadas, incluindo o cache da página.
- Memória cache de página: total de páginas em cache (memória de arquivo).
- Uso máximo de memória: valor máximo do uso da memória observada ao longo do intervalo de medição.
- Tempo disco I/O: o uso de medição do tempo disco I/O foi a soma de todos os discos da máquina, usando a unidade de tempo segundo.
- Uso espaço em disco: representa o uso da capacidade de disco local em execução.
- Ciclos por instrução: são dados recolhidos a partir de contadores de desempenho do processador, mas nem todas as máquinas contidas no cluster tiveram esses dados coletados.

Destas variáveis, foram selecionadas as seguintes para o desenvolvimento da pesquisa:

- Uso de cpu (medida normalizada em core-segundos);
- Uso de memória (medida normalizada em bytes);

- Uso espaço em disco (medida normalizada em bytes).

Optou-se pelo estudo destas três variáveis (cpu, disco e memória) pelo fato delas serem unidades de medida de recursos em servidores com flexibilidade para seu gerenciamento em um ambiente virtualizado, ou seja, caso ocorra uma sobrecarga em um servidor virtual pode se fazer um escalonamento dinâmico com base nos valores dessas variáveis.

O tempo de medição dos dados foi efetuado pela Google em intervalos de 5 minutos (300 segundos). A coleta teve seu início às 19h00min do dia 01 de Maio de 2011 e seu término foi às 19h05min do dia 30 de Maio de 2011. Com isso o arquivo que foi usado na pesquisa *(task resource usage)*, possui dados referentes a um total de 1.232.799.308 tarefas (observações) processadas durante esse período.

A máquina que mais processou tarefas foi a com ID 2497958300 (total de 456.758 tarefas) e a que menos processou foi à máquina com ID 32007818 (total de 1.074 tarefas). Assim foi selecionada aleatoriamente a máquina com o ID 4155527081. Essa máquina processou um total de 92021 tarefas (observações).

Pelo fato de serem muitos dados para uma série temporal, foi aumentado o tempo de medição em três horas. Com isso diminuiu o número de observações, totalizando 233 tarefas durante 29 dias. Assim, esse foi o número de observações utilizadas para as modelagens ARIMA, ARMAX, MLP e NARX. A seguir seguem as etapas utilizadas na metodologia.

3.3. Etapas da metodologia

A seguir serão descritas as etapas metodológicas efetuadas para o desenvolvimento do trabalho.

Para a modelagem ARIMA e ARMAX foi utilizado o software *Eviews 6.0*, sendo utilizado o método dos Mínimos Quadrados Ordinários. Para efetuar a comparação dos

modelos concorrentes, na metodologia de *Box e Jenkins* por meio dos modelos ARIMA e ARMAX foram utilizados os critérios de menor AIC e BIC. Para comparar qual a melhor rede neural treinada foi utilizado como critério o RMSE, raiz quadrada do erro quadrático médio, e o software *Matlab* R2013b.

Para efetuar a estimação dos modelos *Box e Jenkins* e redes neurais artificiais encontrados neste trabalho, foram efetuados os seguintes passos:

1º) Coleta e tratamento dos dados utilizados, disponíveis na página (http://code.google.com/p/googleclusterdata/), no período de 01 à 30 de Maio de 2011. Sendo utilizadas um total de 233 observações;

2º) Investigação do comportamento das variáveis originais por meio de estatísticas descritivas;

3º) Verificação dos correlogramas da função de autocorrelação (FAC) e função de autocorrelação parcial (FACP) para efetuar a modelagem dos resíduos por meio de modelos *Box e Jenkins*.

4°) Testes de estacionariedade: teste de raiz unitária *Dickey-Fuller Aumentado (ADF)* e o teste *KPSS*, sendo que após efetuadas as diferenciações, se a série se tornar estacionária, então os modelos ARIMA são aplicáveis.

5º) Identificação do melhor modelo matemático que represente o processo gerador de cada série, sendo que foram ajustados diversos modelos ARIMA denominados de concorrentes. A escolha do modelo mais adequado para cada série foi verificado, por meio da presença de ruído branco nos resíduos estimados, isto é, obtendo-se uma série de resíduos aleatórios, independentes e identicamente distribuídos (i.i.d.) com média zero e variância constante e não correlacionados.

6°) Utilização dos critérios penalizadores *Akaike (AIC)* e *Schwarz (BIC)*. Os modelos que apresentaram os menores valores desses critérios, seguindo o princípio da parcimônia e a significância dos parâmetros estimados foram os modelos selecionados.

7º) Verificação da correlação existente entre as variáveis. Após, confirmada esta correlação, então pode-se aplicar a modelagem ARMAX aos dados.

8°) Utilização do método *backward* para testar as variáveis independentes e alcançar o modelo que utilize somente variáveis preditoras consideradas em 8 defasagens.

9º) Para a realização de todas as etapas descritas anteriormente, foi utilizado o nível de significância $\alpha = 5\%$.

10°) Encontrar a melhor configuração das redes neurais por meio das etapas de treinamento, teste e validação. O programa computacional utilizado para efetuar estas etapas foi *Matlab R2013b.*

11º) Na etapa de treinamento são fornecidos dois conjuntos de dados: os valores de entrada e os valores que representam a saída esperada para tais entradas. Define-se uma configuração (número de camadas ocultas, quantidade de neurônios nessas camadas, funções de transferência e ativação e taxa de aprendizagem) e então a rede é treinada utilizando-se o algoritmo *back-propagation*. Ao final dessa etapa é comparada a saída gerada pela rede com a saída esperada. O erro dessa fase é dado com base na diferença desses valores (gerado – esperado).

12°) Na etapa de teste é fornecida somente a entrada para a rede e então o erro é calculado novamente com base nos valores gerados - esperados. Com base no erro destas duas etapas é selecionada a melhor configuração para a rede. Passa-se então para a etapa de validação.

13°) Na etapa de validação é verificado o poder de generalização da rede. Usando a melhor configuração obtida através dos menores erros, a rede é novamente treinada com valores de entrada e saída esperados diferentes dos utilizados nas fases anteriores. Após esse novo treinamento é feita a execução da rede somente com os valores de entrada e então a saída obtida é comparada com a saída esperada. Com base nisso é gerado o erro de validação e então pode-se analisar se a rede é capaz de ser aplicada no contexto do problema.

14º) Por fim foi calculada a medida de acurácia através do critério de avaliação RMSE, com intuito de comparar e descobrir qual é a melhor técnica de predição entre os modelos analisados ARIMA, ARMAX, MLP e NARX.

No capítulo 4, apresenta-se a análise e a discussão dos resultados obtidos nessa pesquisa.

4. RESULTADOS E DISCUSSÃO

Nesse capítulo são apresentados os resultados e discussão referentes à aplicação da metodologia desenvolvida no capítulo 3 e as técnicas estatísticas explicitadas no capítulo 2.

4.1. Análise exploratória dos dados

Com o objetivo de realizar um estudo sobre o comportamento de consumo das variáveis cpu, disco e memória da máquina selecionada foi efetuada a análise descritiva de cada uma das variáveis, conforme Tabela 2:

Tabela 2 - Análise descritiva das variáveis originais cpu, disco e memória dos servidores, (n = 233).

Variáveis	Média	Mediana	Máximo	Mínimo	Desvio Padrão	Assimetria
Cpu	0.015655	0.001369	0.234100	0.000562	0.035406	3.218845
Disco	0.002400	0.000188	0.37190	0.000146	0.008696	3.703145
Memória	0.055433	0.06750	0.068360	0.032040	0.016204	-0.646362

Fonte: elaborada pela autora.

Ao analisar a assimetria das variáveis da Tabela 2, nota-se que somente o coeficiente da variável memória foi negativo, apresentando assim uma distribuição assimétrica à esquerda (ou negativa), ao contrário das outras variáveis (cpu e disco) que apresentaram uma distribuição assimétrica à direita (positiva).

4.2. Aplicabilidade modelagem ARIMA - identificação dos modelos

A primeira etapa realizada para a modelagem das séries cpu, disco e memória foi a verificação da estacionariedade. Essa etapa consiste em verificar se as séries são produzidas por um processo estacionário, ou seja, se a média e a variância são constantes ao longo do tempo.

Assim foi realizada a inspeção visual da variável cpu ao longo do tempo. Conforme Figura 13:

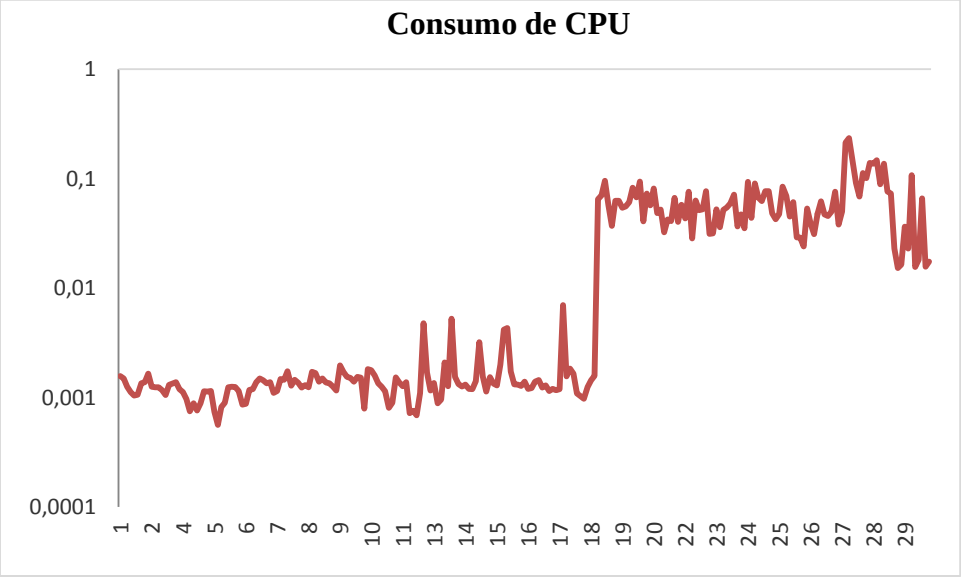


Figura 13 - Gráfico da série original consumo de cpu.
Fonte: elaborada pela autora.

Na Figura 13, no eixo vertical está o consumo da variável cpu, e no eixo horizontal está o tempo em dias. Ao visualizar o gráfico, percebe-se que há uma tendência de crescimento de consumo de cpu ao longo dos dias. Pode-se observar ainda que a partir do dia 18/05/11, há um consumo maior do recurso de processamento de cpu do servidor de internet em estudo.

A seguir segue o gráfico da série original variável disco, Figura 14:

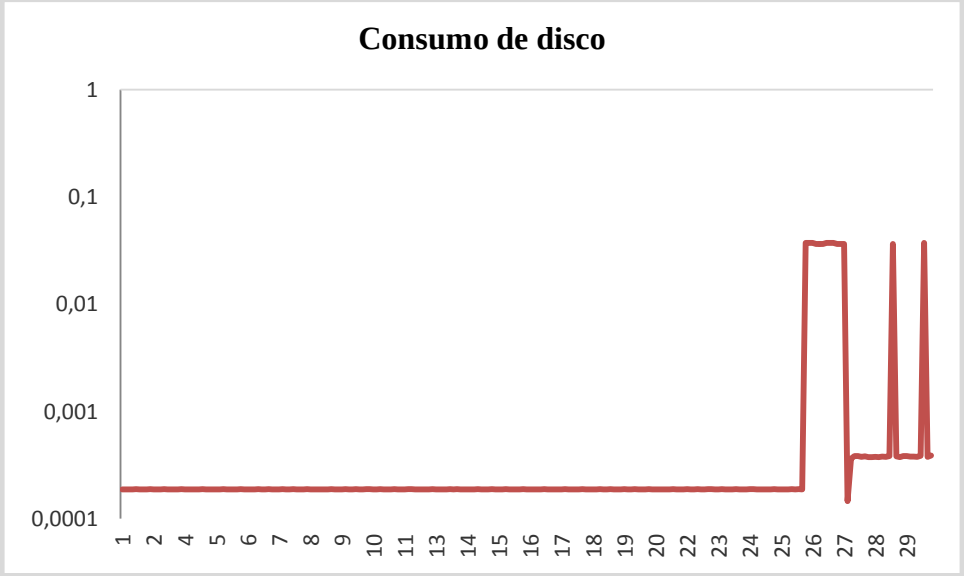


Figura 14 - Gráfico da série original consumo de disco.
Fonte: elaborada pela autora.

Na *Figura 14*, nota-se que o consumo de disco, apresentado no eixo vertical é constante até o dia 26/05/11. Em seguida ocorreu um crescimento acentuado de consumo no processamento do disco nesse servidor, diminuindo no dia 27/05/11.

Uma possível hipótese para esse aumento crescente em um curto intervalo de tempo pode ser uma sobrecarga em consequência de um alto número acessos simultâneos aos serviços que rodam nesse servidor.

Na Figura 15, apresenta o comportamento da variável memória.

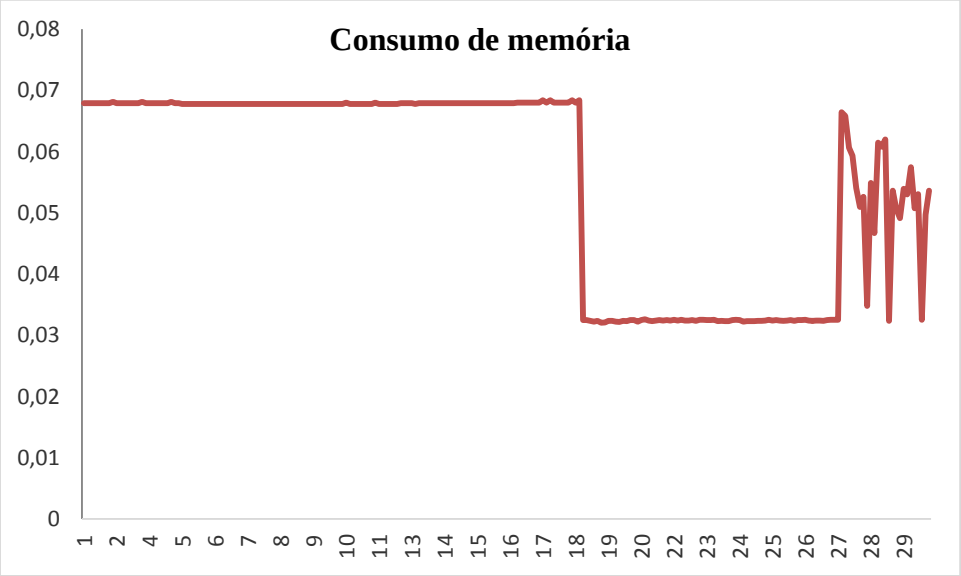


Figura 15 - Gráfico da série original consumo de memória.
Fonte: elaborada pela autora.

No eixo vertical, a variável memória registra a quantidade diária de gigabytes e no eixo horizontal aparece a data e hora de consumo de memória ao longo de um mês.

A inspeção visual do gráfico revela que não há uma tendência de crescimento de consumo de memória. A série mostra que o recurso consumido de memória é constante até o dia 18/05/11. Após esse dia o consumo cai bruscamente, havendo picos de consumo a partir do dia 27/05/11.

Para verificar se as variáveis são estacionárias além da inspeção visual, é necessário aplicarmos outros testes de estacionariedade. Dessa forma foram utilizados os testes

Augmented Dickey-Fuller (ADF) e o teste proposto por *Kwiatkowski, Phillips, Schimidt, Schin (KPSS)*, para verificar a presença ou não da estacionariedade nas variáveis.

Conforme já exposto no capítulo 2, a hipótese H_0 do teste ADF é que a série possui raiz unitária, isto é, a série não é estacionária em nível, sem aplicar nenhuma diferença. Logo, observa-se na Tabela 3 que não se pode rejeitar a hipótese nula, ou seja, de que as séries possuam raiz unitária.

Ao verificar os valores na Tabela 3, verifica-se que as séries disco e memória são não estacionárias em nível, pois o valor crítico do teste é menor que o valor das estatísticas calculadas para os níveis de significância, bem como o *p*-valor não foi significativo. Mas quando as séries das variáveis memória e disco são analisadas em primeira diferença, não se pode rejeitar a hipótese alternativa de ausência de raiz unitária, assim pode-se dizer que as séries são estacionárias.

Em relação a série cpu observa-se que a hipótese nula foi rejeitada, pois não há presença de raiz unitária, ou seja, a série da variável cpu é estacionária em nível *I(0)*, pois o *p*-valor apresentou um valor inferior ao nível de significância $\alpha = 5\%$, tanto em nível, quanto em primeira diferença, conforme se verifica na Tabela 3.

Tabela 3 - Teste da raiz unitária Augmented Dickey-Fuller (ADF)

Nível de confiança	Em nível			Em primeira diferença		
	CPU	DISCO	MEMÓRIA	CPU	DISCO	MEMÓRIA
Valor crítico do ADF	-5,2574	-2,5538	-2,2528	-4,0522	-6,5665	-20,9878
5%	-3,4293	-3,4305	-3,4293	-3,4302	-3,4305	-3,4293
p-valor	0,0001*	0,3021	0,4576	0,0085*	0,0000*	0,0000*

Fonte: elaborada pela autora. *Significância a 5%.

Ainda com o propósito de confirmar os resultados obtidos pelo teste ADF, empregou-se o teste KPSS, cuja hipótese nula corresponde a estacionariedade da série - *I(0).* Na Tabela 4, encontram-se os resultados do teste KPSS em nível e em primeira diferença das séries analisadas.

Tabela 4- Teste de estacionariedade Kwaiatkowski, Phillips, Schmidt and Shin (KPSS).

Nível de confiança	Em nível			Em primeira diferença		
	CPU	DISCO	MEMÓRIA	CPU	DISCO	MEMÓRIA
Valor crítico do KPSS	0,2775	0,1061	0,1840	0,0800	0,257	0,0884
5%	0,1460	0,1460	0,1460	0,1460	0,1460	0,1460
p-valor	0,0000	0,0000	0,0000	0,9289	0,9424	0,7605

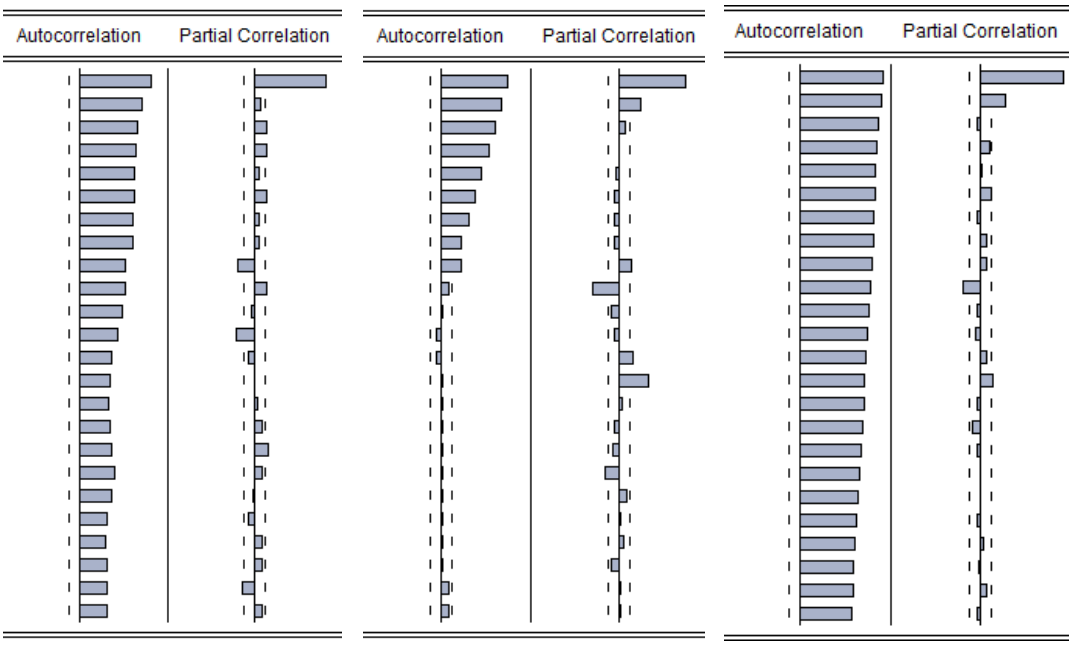
Fonte: elaborada pela autora.

O teste de KPSS inverte a hipótese nula (estacionariedade) contra a hipótese alternativa de existência de raiz unitária. De acordo com *Kwiatkowski et al.* (1992, p. 176), o teste KPSS tende a complementar o teste de raiz unitária de *Dickey-Fuller*. Observa-se que nas séries em nível não se pode rejeitar a hipótese alternativa de não estacionariedade, enquanto que, depois de aplicada uma diferença, pode-se dizer que as séries são estacionárias.

Assim a aplicação dos testes de estacionariedade mostrou que nem todas as séries se encontravam estacionárias em nível, após a aplicação de uma diferenciação, tornaram-se estacionárias, sendo este o pressuposto básico para a aplicação da metodologia de *Box e Jenkins.*

Depois de verificada a estacionariedade das variáveis cpu, disco e memória por meio dos testes ADF e KPSS em nível e primeira diferença, foi efetuada a análise da função de

autocorrelação (FAC) e autocorrelação parcial (FACP) das variáveis em estudo, apresentadas na Figura 16.



a) cpu b) disco c) memória

Figura 16 - FAC e FACP – a) variável cpu, b) variável disco, c) variável memória.
Fonte: elaborada pela autora.

Na Figura 16, observa-se que a FAC da variável cpu diminui lentamente, as desafagens estão fora do limite de confiança de 95%, ou seja, os valores são significativamente diferentes de zero. A FACP após a primeira defasagem diminui sendo insignificante até o *lag* 24.

A FAC da variável disco, decai exponencialmente até a defasagem 9, a partir daí encontram-se dentro do limite de confiança de 95%, ou seja, a partir da defasagem 9 a média cai em torno de zero. Na FACP a partir da segunda defasagem apresenta-se um nível significativo, o que pode indicar um possível modelo AR*(p).*

O correlograma da variável memória mostra que os coeficientes de autocorrelação (FAC) decrescem lentamente conforme aumenta o número de defasagens, e a FACP decai exponencialmente para zero.

4.2.1. Estimativa dos modelos

Nessa fase foram estimados os diversos modelos concorrentes para as variáveis cpu, disco e memória. Como o modelo ARIMA é univariado, cada variável foi modelada separadamente.

Na Tabela 5 são apresentados os dez melhores modelos ARIMA, encontrados para a variável cpu e suas respectivas equações matemáticas.

Tabela 5 - Modelos ARIMA selecionados para a variável cpu, (n = 233).

Variável	**Modelos ARIMA**	**Equação matemática**	**AIC**	**BIC**
CPU	ARIMA (1,1,1)	$\Delta Z_t = +0{,}5715z_{t-1} - 0{,}8630a_{t-1}$	-5,139	-5,110
CPU	ARIMA (1,1,2)	$\Delta Z_t = -0{,}2625z_{t-1} - 0{,}2799a_{t-2}$	-5,106	-5,076
CPU	ARIMA (1,1,3)	$\Delta Z_t = -0{,}1877z_{t-1} - 0{,}2014a_{t-3}$	-5,073	-5,043
CPU	ARIMA (2,1,1)	$\Delta Z_t = -0{,}2000z_{t-2} - 0{,}2453a_{t-1}$	-5,096	-5,067
CPU	ARIMA (2,1,3)	$\Delta Z_t = -0{,}2105z_{t-2} - 0{,}1841a_{t-3}$	-5,077	-5,047
CPU	ARIMA (3,1,1)	$\Delta Z_t = -0{,}1975z_{t-3} - 0{,}3333a_{t-1}$	-5,090	-5,060
CPU	ARIMA (3,1,2)	$\Delta Z_t = -0{,}2059z_{t-3} - 0{,}2392a_{t-2}$	-5,078	-5,048
CPU	ARIMA (3,1,3)	$\Delta Z_t = +0{,}7203z_{t-3} - 0{,}9585a_{t-3}$	-5,094	-5,064
CPU	ARIMA (5,1,2)	$\Delta Z_t = -0{,}1585z_{t-5} - 0{,}2178a_{t-2}$	-5,052	-5,022
CPU	ARIMA (5,1,4)	$\Delta Z_t = +0{,}8324z_{t-5} - 0{,}9186a_{t-4}$	-5,023	-4,992

Fonte: elaborada pela autora.

Observando os resultados, Tabela 5, verifica-se que o melhor modelo para a variável cpu foi um ARIMA (1,1,1), sendo que o pior modelo foi ARIMA (5,1,4), pois além de possuir os maiores valores de critérios de informação AIC e BIC, o pior modelo ainda apresentou o maior número de parâmetros.

A seguir a Tabela 6, descreve os dez melhores modelos ARIMA, encontrados para a variável disco.

Tabela 6 - Modelos ARIMA selecionados para a variável disco, (n = 233).

Variável	**Modelos ARIMA**	**Equação matemática**	**AIC**	**BIC**
DISCO	ARIMA (1,1,2)	$\Delta Z_t = -0{,}3847 z_{t-1} - 0{,}1614 a_{t-2}$	-7,558	-7,529
DISCO	ARIMA (1,1,9)	$\Delta Z_t = -0{,}2692 z_{t-1} + 0{,}3520 a_{t-9}$	-7,654	-7,624
DISCO	ARIMA (1,1,11)	$\Delta Z_t = -0{,}4289 z_{t-1} - 0{,}3596 a_{t-11}$	-7,594	-7,564
DISCO	ARIMA (2,1,2)	$\Delta Z_t = +0{,}9069 z_{t-2} - 1{,}0860 a_{t-2}$	-7,568	-7,538
DISCO	ARIMA (2,1,10)	$\Delta Z_t = -0{,}1568 z_{t-2} - 0{,}3325 a_{t-10}$	-7,473	-7,443
DISCO	ARIMA (5,1,5)	$\Delta Z_t = -0{,}7061 z_{t-5} + 0{,}8357 a_{t-5}$	-7,424	-7,394
DISCO	ARIMA (6,1,4)	$\Delta Z_t = +0{,}4610 z_{t-6} + 0{,}1554 a_{t-4}$	-7,498	-7,468
DISCO	ARIMA (6,1,6)	$\Delta Z_t = -0{,}6752 z_{t-6} + 0{,}9104 a_{t-6}$	-7,434	-7,404
DISCO	ARIMA (8,1,2)	$\Delta Z_t = -0{,}2680 z_{t-8} - 0{,}0627 a_{t-2}$	-7,435	-7,404
DISCO	ARIMA (8,1,9)	$\Delta Z_t = -0{,}1697 z_{t-8} + 0{,}4025 a_{t-9}$	-7,577	-7,547

Fonte: elaborada pela autora.

Observando-se os resultados da Tabela 6, verifica-se que o melhor modelo de acordo com AIC e BIC, foi um ARIMA (1,1,9), que apresentou 1 parâmetro autorregressivo (*p*) e 9

parâmetros de médias móveis (*q*) sendo que o pior modelo foi (6,1,6), que apresentou 6 parâmetros autorregressivos (*p*) e 6 de médias móveis (*q*).

Na Tabela 7, descreve-se os dez melhores modelos ARIMA, encontrados para a variável memória:

Tabela 7 - Modelos ARIMA selecionados para a variável memória, (n = 233).

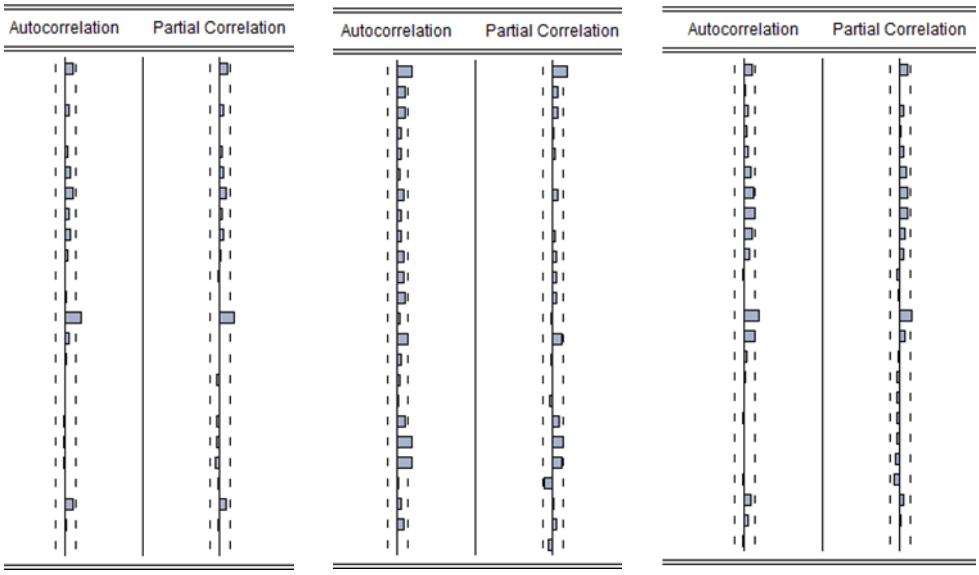
Variável	Modelos ARIMA	Equação matemática	AIC	BIC
MEMÓRIA	ARIMA (3,1,1)	ΔZ_t=-0,1481z_{t-3} – 0,3095a_{t-1}	-7,872	-7,842
MEMÓRIA	ARIMA (3,1,5)	ΔZ_t= -0,1601z_{t-3} – 0,1723a_{t-5}	-7,798	-7,768
MEMÓRIA	ARIMA (3,1,6)	ΔZ_t= -0,1844z_{t-3} + 0,2044a_{t-6}	-7,803	-7,773
MEMÓRIA	ARIMA (3,1,7)	ΔZ_t = -0,1747z_{t-3} – 0,1400a_{t-7}	-7,793	-7,763
MEMÓRIA	ARIMA (5,1,1)	ΔZ_t= -0,1528z_{t-5} – 0,3275a_{t-1}	-7,865	-7,835
MEMÓRIA	ARIMA (5,1,7)	ΔZ_t = -0,1793z_{t-5} – 0,1523a_{t-7}	-7,785	-7,755
MEMÓRIA	ARIMA (6,1,3)	ΔZ_t= +0,1823z_{t-6} – 0,1786a_{t-3}	-7,781	-7,751
MEMÓRIA	ARIMA (6,1,9)	ΔZ_t = +0,1425z_{t-6}+ 0,1876a_{t-9}	-7,786	-7,756
MEMÓRIA	ARIMA (7,1,3)	ΔZ_t= -0,1562z_{t-7}– 0,1445a_{t-3}	-7,772	-7,742
MEMÓRIA	ARIMA (7,1,5)	ΔZ_t = -0,1630z_{t-7}– 0,1970a_{t-5}	-7,780	-7,750

Fonte: elaborada pela autora.

Observando-se os resultados da Tabela 7, verifica-se que o melhor modelo de previsão para a variável memória foi ARIMA (3,1,1), que apresentou 7 parâmetros autorregressivos (*p*) e 5 parâmetros de médias móveis (*q*).

4.2.2. Verificação dos modelos

Posteriormente foi realizada a escolha do melhor modelo que representasse o processo gerador da série, sendo que a verificação dos modelos ocorre por meio da análise dos resíduos e ordem do modelo. Os resíduos devem apresentar o comportamento de ruído branco, ou seja, as correlações do modelo devem ser não significativas apresentando uma ausência de autocorrelação serial. Figura 17.



a) cpu b) disco c) memória

Figura 17 - Correlograma dos resíduos: a) cpu, b) disco, c) memória.
Fonte: elaborada pela autora.

A Figura 17 revela que os resíduos, efetivamente, apresentam comportamento aleatório, ou seja, ruído branco. Se houvesse uma dependência temporal dos valores sucessivos dos resíduos, os valores iriam apresentar correlação entre si e estariam fora do

limite de confiança de 95%. Com isso haveria uma piora na estimativa dos parâmetros dos modelos encontrados.

Mas este não é o caso, logo os modelos das variáveis cpu, disco e memória são adequados no que se refere à análise dos resíduos. Os valores *p* das séries em estudo ficaram abaixo de 5% de significância estatística indicando um forte grau de ajuste do modelo escolhido.

Ordem do modelo: A ordem dos modelos se dá através dos critérios de informação AIC e BIC. Assim, após efetuadas todas as análises descritas anteriormente, segue os melhores modelos selecionados para cada variável representados na

Tabela *8* de acordo com os critérios AIC e BIC:

Tabela 8 - Melhores modelos ARIMA selecionados para a variável cpu, disco e memória, (n = 233).

Variável	**Modelos ARIMA**	**Equação matemática**	**AIC**	**BIC**
CPU	ARIMA (1,1,1)	$\Delta Z_t = +0{,}5715z_{t-1} - 0{,}8630a_{t-1}$	-5,139	-5,110
DISCO	ARIMA (1,1,9)	$\Delta Z_t = -0{,}2692z_{t-1} + 0{,}3520a_{t-9}$	-7,654	-7,624
MEMÓRIA	ARIMA (3,1,1)	$\Delta Z_t = -0{,}1481z_{t-3} - 0{,}3095a_{t-1}$	-7,872	-7,842

Fonte: elaborada pela autora.

Como se pode verificar a Tabela 8 mostra os modelos mais adequados para a análise de previsão para cada série analisada.

4.2.3. Previsão

Um dos objetivos ao se analisar uma série temporal é a possibilidade de utilizar técnicas apropriadas a fim de encontrar bons modelos preditivos. Isso se dá pelo fato de que as séries temporais se baseiam na ideia de que observações passadas de um conjunto de dados possuem informações sobre o padrão do comportamento destes dados no futuro.

Além disso, é possível analisar o comportamento das séries. Assim foi possível prever o consumo dos recursos computacionais referentes ao servidor de internet. Conforme Tabela 9 observa-se a previsão de consumo um passo à frente, das variáveis cpu, disco e memória, o valor real das variáveis referem-se ao dia 30 de Maio de 2011.

Tabela 9- Previsão de consumo cpu, disco e memória.

Variável	**Valor real**	**Valor previsto**
CPU	0,01736	0,02049
DISCO	0,00039	0,00026
MEMÓRIA	0,05359	0,05398

Fonte: elaborada pela autora.

As estimativas de previsões para as variáveis ocorreram a partir das estimativas dos modelos ajustados. Após se efetuar todas as etapas da metodologia *Box e Jenkins*, também foi realizada a análise do modelo ARMAX, outra metodologia de séries temporais.

4.3. Aplicabilidade modelagem ARMAX - Identificação dos modelos

De acordo cos os pressuposto da modelagem ARMAX, Capítulo 2, as variáveis tem que ser estacionárias. Conforme testes realizados na seção anterior, as variáveis foram estacionadas em primeira diferença *I(1)*. Para a elaboração do modelo ARMAX, foram utilizadas as variáveis exógenas cpu e disco, como variável endógena a variável memória.

Verificou-se também a correlação entre as variáveis, (Tabela 10), uma vez que para modelar ARMAX a variável endógena tem que ser correlacionada com as variáveis exógenas.

Tabela 10 - Correlação entre as variáveis

	CPU	**DISCO**	**MEMÓRIA**
CPU	1,00	0,50	0,77
DISCO	0,50	1,00	0,71
MEMÓRIA	0,77	0,71	1,00

Fonte: elaborada pela autora.

Na Tabela 10, há uma relação positiva entre o consumo de cpu, disco e memória, ou seja, quando aumenta o consumo de recurso de memória no servidor de internet, aumenta também o consumo de processamento de cpu e consumo de disco.

4.3.1. Estimativa dos modelos

Nessa fase, aplicou-se o método *backward* para seleção de regressores das variáveis exógenas. Sendo que as defasagens das variáveis cpu e disco foi em até oito tempos (t_{-1}, t_{-2}, t_{-3}, t_{-4}, t_{-5}, t_{-6}, t_{-7}, t_{-8}). Foi considerado um nível descritivo de 5% para exclusão dos regressores dessas variáveis. Utilizou-se o software *Eviews 6.0* para execução da parte computacional dos cálculos, sendo adotado o método de estimação dos mínimos quadrados ordinários. A seguir, na Tabela 11 observa-se os resultados encontrados.

Tabela 11 - Coeficientes, erro padrão, estatística t e nível descritivo (*p*-valor) para o modelo para o consumo de memória – método *backward.*

Variável	**Coeficientes**	**Desvio-Padrão**	**t-student**	**p-valor**
CPU(-1)	0,0662	0,01715	3,8632	0,0001
CPU(-3)	0,0506	0,0176	2,8609	0,0046
CPU(-8)	0,0478	0,0193	2,4664	0,0144
DISCO(-1)	0,2452	0,0571	4,2904	0,0000
DISCO(-2)	0,1656	0,0640	2,5853	0,0104
DISCO(-8)	0,1608	0,0722	2,2271	0,0270

Fonte: elaborada pela autora.

Na Tabela 12, são apresentados os resultados para os dez modelos de previsão e respectivos critérios de informação AIC e BIC, mais adequados a série.

Tabela 12 - Dez melhores modelos de previsão para a variável memória.

MODELOS ARMAX	AIC	BIC
AR(1), I(1), MA(1), X (DISCO $_{1:2}$, DISCO $_{8,}$ CPU_1, CPU_3,CPU_8)	-7,875	-7,753
AR(1), I(1), MA(3), X (DISCO $_{1:2}$, DISCO $_{8,}$ CPU_1, CPU_3, CPU_8)	-7,888	-7,766
AR(1), I(1), MA(6), X (DISCO $_{1:2}$, DISCO $_{8,}$ CPU_1, CPU_3, CPU_8)	-7,882	-7,760
AR(1), I(1), MA(9), X (DISCO $_{1:2}$, DISCO $_{8,}$ CPU_1, CPU_3, CPU_8)	-7,878	-7,755
AR(2), I(1), MA(2), X (DISCO $_{1:2}$, DISCO $_{8,}$ CPU_1, CPU_3 ,CPU_8)	-7,834	-7,711
AR(3), I(1), MA(1), X (DISCO $_{1:2}$, DISCO $_{8,}$ CPU_1, CPU_3, CPU_8)	-7,890	-7,767
AR(4), I(1), MA(4), X (DISCO $_{1:2}$, DISCO $_{8,}$ CPU_1, CPU_3, CPU_8)	-7,842	-7,719
AR(6), I(1), MA(1), X (DISCO $_{1:2}$, DISCO $_{8,}$ CPU_1, CPU_3, CPU_8)	-7,852	-7,728
AR(6), I(1), MA(3), X (DISCO $_{1:2}$, DISCO $_{8,}$ CPU_1, CPU_3, CPU_8)	-7,881	-7,757
AR(9), I(1), MA(1), X (DISCO $_{1:2}$, DISCO $_{8,}$ CPU_1, CPU_3, CPU_8)	-7,838	-7,712

Fonte: elaborada pela autora.

De acordo com a Tabela 12, todos os melhores modelos de previsão para a variável memória, utilizam o consumo de cpu e de disco de oito tempo anteriores para obter as melhores previsões. Com o teste de *backward* que exclui os regressores, fica evidenciado que o consumo de cpu e disco até 24 horas anteriores são importantes para a previsão de consumo da variável memória.

Ainda, conforme Tabela 12, e de acordo com os critérios de informação AIC e BIC, o melhor modelo de previsão modelo ARMAX foi AR (3), I(1), MA (1), X (DISCO $_{1:2}$, DISCO $_{8}$, CPU_2, CPU_3, CPU_8).

4.3.2. Verificação dos modelos

Assim como a modelagem ARIMA, para a verificação do modelo ARMAX tem que se fazer uma análise do melhor modelo que seja capaz de representar o processo gerador da série por meio da análise dos resíduos e ordem do modelo. Conforme Figura 18.

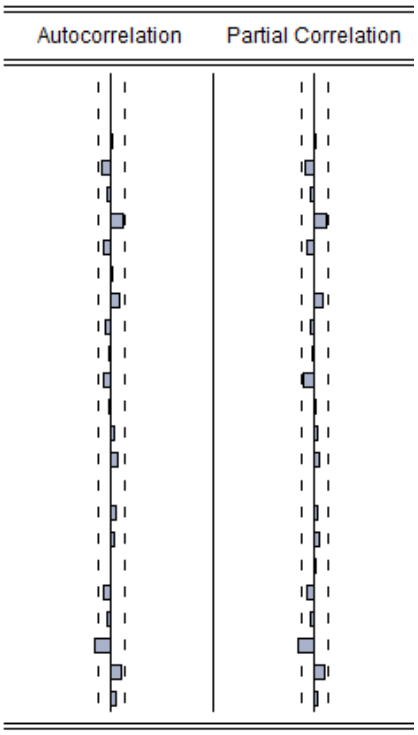


Figura 18 - Correlograma dos resíduos, modelo ARMAX - memória.
Fonte: elaborada pela autora.

A análise do correlograma dos resíduos indica ausência de autocorrelação serial, pois todos os valores residuais encontram-se dentro de intervalo de confiança de 95%. Assim o melhor modelo de previsão para o consumo de memória é o modelo final apresentado na Tabela 13.

Tabela 13 - Coeficientes, erro padrão, estatística t e p-valor para o modelo final variável memória.

Variáveis	**Coeficientes**	**Desvio-Padrão**	**t-student**	**p-valor**
CPU(-1)	0,0583	0,0178	3,2624	0,0013
CPU(-3)	0,0563	0,0191	2,9432	0,0036
CPU(-8)	0,0523	0,0205	2,5475	0,011
DISCO(-1)	0,1674	0,0677	2,4736	0,0142
DISCO(-2)	0,1608	0,0611	2,6303	0,0092
DISCO(-8)	0,1444	0,0712	2,0331	0,0433

AR(3)	-0,2330	0,0786	-2,9620	0,0034
MA(1)	-0,2004	0,0868	-2,3068	0,0220

Fonte: elaborada pela autora.

Seguindo a análise dos resíduos e a significância dos parâmetros estimados, na Tabela 13 é apresentado o modelo mais adequado para a análise de previsão para o consumo de memória, sendo utilizas as variáveis cpu e disco como externas.

Com isso, após a verificação do modelo mais adequado de acordo com o modelo ARMAX, o próximo passo é realizar a previsão.

4.3.3. Previsão

De acordo com o gráfico de previsão de consumo de memória, Figura 19, visualmente os valores previstos acompanharam relativamente às oscilações de consumo dos valores reais da série.

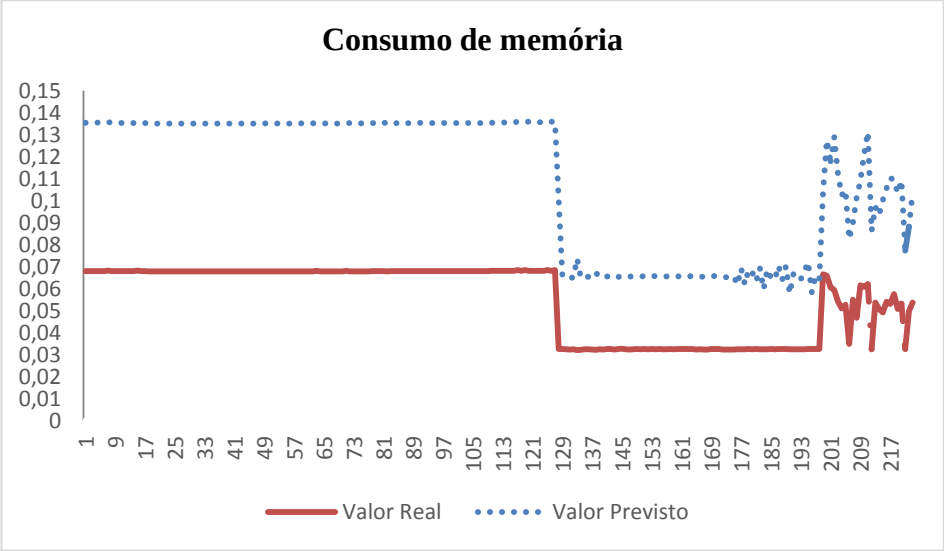


Figura 19 - Gráfico de previsão para o modelo ARMAX.
Fonte: Elaborada pela autora.

No gráfico, observa-se que os valores de previsão são próximos aos valores reais. Isso indica que o modelo escolhido é adequado, uma vez que os erros são baixos devido aos valores previstos e reais serem próximos.

Foi realizada a previsão de consumo da variável memória em um horizonte de previsão $\hat{Z}_{T+1}$, um passo à frente. Dessa forma, o valor previsto para o consumo de memória no servidor de internet em estudo, no período três horas à frente, é de 0,0509.

Assim depois de realizados os quatro ciclos iterativos da metodologia *Box e Jenkins* (identificação, estimação, verificação e previsão), o próximo passo proposto neste estudo é efetuar o treinamento e aprendizado de uma RNA MLP.

4.4. Rede neural multilayer perceptron – MLP

Após definir as variáveis de entrada *(inputs)* e a variável de saída *(output)*, obtém-se uma rede que permite minimizar o RMSE (raiz quadrada do erro quadrático médio).

Para isso, deve-se escolher um modelo de rede que tenha o melhor valor de previsão, por meio de tentativa e erro, sendo simuladas inúmeras redes para assim alcançar um alto grau de precisão.

Um problema encontrado neste estudo referente a MLP foi o fato de que a rede não obteve um bom modelo com apenas 233 observações. Um dos prováveis motivos para isso ter ocorrido é chamado de *overfitting,* que se dá durante o processo de treinamento da rede.

> Tal problema é caracterizado como a perda da capacidade de generalização, ou seja, a diminuição da capacidade de fornecer respostas coerentes para padrões que não foram utilizados para treiná-la. O *overfitting* ocorre quando a rede passa a "decorar" os padrões de treinamento incorporando até eventuais ruídos presentes nos dados. (SEVERO, 2010, p.16)

Na Figura 20, observa-se o melhor desempenho encontrado para a MLP com 233 observações:

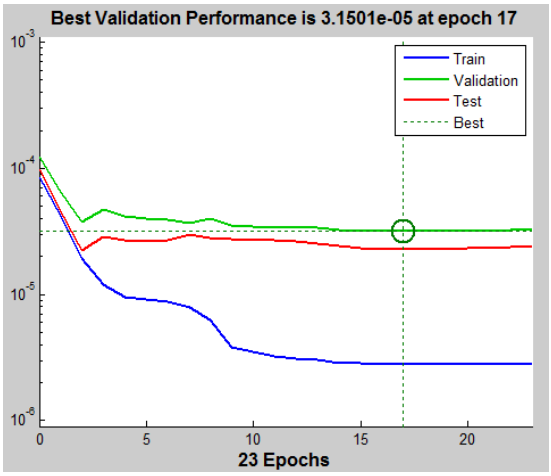


Figura 20 - Melhor performance da RNA MLP com 233 observações.
Fonte: gerado no *Matlab* R20013b.

Ao verificar, encontrou-se o erro foi 3,1501 x 10^{-5}, isso fez com que essa configuração fosse descartada, uma vez que este erro é considerado alto para a predição. Esse fato ocorreu devido ao pouco número de observações utilizadas para o treinamento da rede.

Portanto foi necessário fazer uma nova definição referente aos dados de treinamento da rede. Para isso, foram utilizadas 600 observações do conjunto de dados originais associados ao servidor escolhido, sendo que desse total 70% (420 amostras) foram utilizadas para treinamento, 15% (90 amostras) para teste e 15% (90 amostras) para validação.

Dessa maneira, após várias simulações, chegou-se ao modelo mais adequado para a nova rede. A Figura 21 representa a melhor arquitetura encontrada para a MLP durante as simulações.

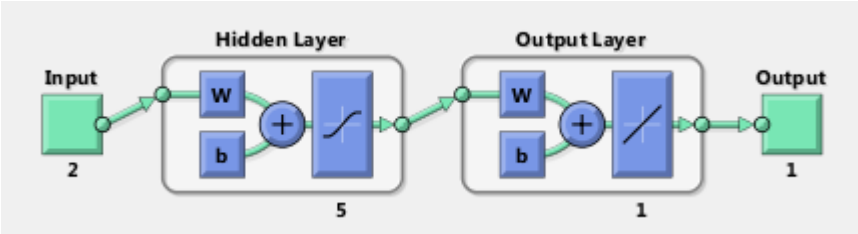


Figura 21 - Estrutura da rede neural MLP com 600 observações.
Fonte: Fonte: gerado no *Matlab* R20013b.

Conforme ilustra Figura 21, a topologia de rede MLP que apresentou menor RMSE foi definida com 2 neurônios na camada de entrada, 2 camadas ocultas e por fim 1 neurônio na camada de saída. Após encontrar a melhor topologia da rede, verificou-se a performance da mesma, conforme Figura 22.

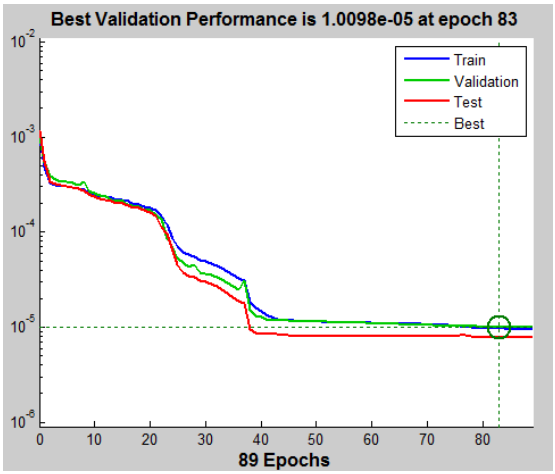


Figura 22 - Melhor performance da RNA MLP.
Fonte: gerado no *Matlab* R20013b.

A taxa de convergência do treinamento, teste e validação da rede ocorreu a partir de 40 épocas, ou seja, a partir de 40 épocas o erro da rede se tornou constante. A melhor

performance da MLP, foi obtida com erro quadrático médio (mse) de 1,0098 x 10^{-5} e com 83 épocas. O RMSE, critério de avaliação deste estudo, foi 0,003177.

Após foi verificado o coeficiente de determinação de validação do modelo, conforme Figura 23. Este mostra o nível de correlação que há entre as variáveis preditas x esperadas para a validação da rede.

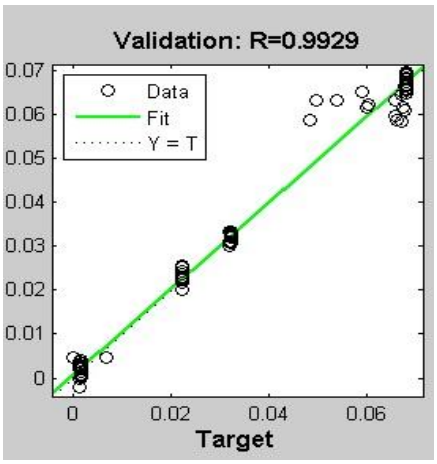


Figura 23 - Correlação entre as variáveis preditas X esperadas.
Fonte: Fonte: gerado no *Matlab* R20013b.

O valor R^2 é referente ao coeficiente de determinação do ajuste do modelo entre os valores observados e os valores estimados, quanto mais próximo de 1, melhor é o ajuste do modelo. Neste caso o valor de R^2 é de 0,9929, que representa um ajuste bom da rede.

Pela análise dos valores obtidos por meio dos gráficos pode-se constatar que a rede foi capaz de aprender com os valores fornecidos na entrada (consumo de disco e consumo de cpu) e predizer com pouco erro os valores esperados na saída (consumo de memória).

Na Tabela 14 são apresentados os parâmetros do modelo de RNA desenvolvido.

Tabela 14 - Parâmetros da rede neural MLP.

Parâmetros	**Valor**
Neurônios	9
Camadas ocultas	2
Épocas de treinamento (iterações)	83
RMSE	0,003177
R^2 estimado	0,9929

Fonte: elaborada pela autora.

Dessa forma, a partir dos resultados encontrados na construção da MLP, pode-se constatar que ao se utilizar poucas amostras no treinamento da rede, não foi possível construir uma arquitetura MLP adequada para a predição.

Contudo ao aumentar o número de observações para treinamento da rede, percebeu-se a melhora da performance da topologia da MLP e isso possibilitou achar o modelo mais adequado da rede MLP para se efetuar a previsão do consumo de memória. A seguir seguem os resultados analisados a partir da construção da RNA chamada NARX.

4.5. Rede neural NARX

Assim como a MLP, como entrada para rede no modelo NARX foram utilizados o consumo de disco e cpu e, como saída esperada, foi utilizado o consumo de memória.

Para a configuração desta foi utilizado o conjunto de 233 observações, sendo usados 70% para o treinamento (163 amostras), 15% foram usados para a teste da rede (35 amostras) e 15% foi utilizado para validação (35 amostras).

A Figura 24 mostra a topologia da RNA NARX.

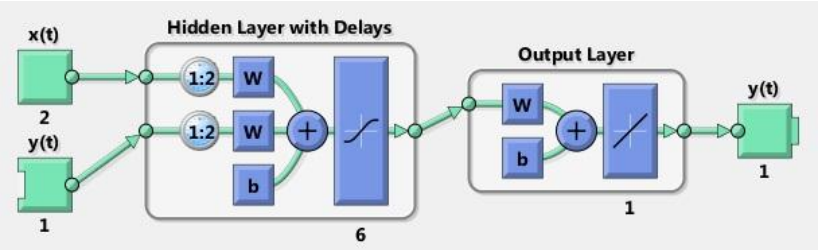


Figura 24 - Estrutura da rede neural NARX.
Fonte: gerado no *Matlab* R20013b.

A topologia NARX, ilustrada pela Figura 24, é composta por duas camadas de entrada x(t) e y(t), 1 camada oculta com 6 neurônios e uma camada de saída y(t). Esta rede teve duas defasagens de atraso $x(t_{-1})$, $x(t_{-2})$. Os valores da série x(t) correspondem as entradas exógenas das variáveis cpu e disco, os valores referentes a série y(t), correspondem aos dados da variável memória. A partir daí foi analisada a performance da NARX (Figura 25).

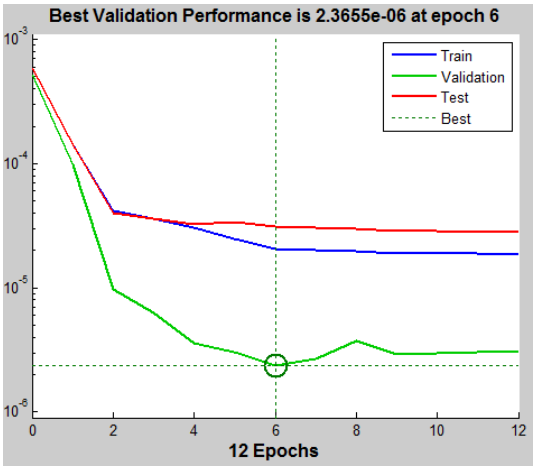


Figura 25 - Melhor performance da RNA NARX.
Fonte: gerado no *Matlab* R20013b.

Na Figura 25 é mostrada a interface que representa a evolução do erro obtido na etapa de validação, sendo que após a convergência da rede, o decaimento se torna constante ao

longo das iterações. A melhor performance na etapa de validação foi um erro de 2,3655 x $10^{-6,}$ sendo que o RMSE foi 0,001538.

Na Figura 26 observa-se o nível de correlação que há entre as variáveis preditas x esperadas para a validação da rede.

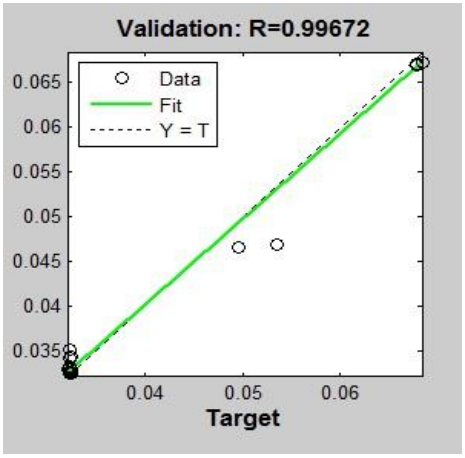


Figura 26 - Correlação entre as variáveis preditas X esperadas rede NARX.
Fonte: Fonte: gerado no *Matlab* R20013b.

O valor do coeficiente de determinação da rede neural NARX foi de 0,99672, o que representa que está topologia é adequada para a previsão do consumo de memória.

A Figura 27 mostra autocorrelação dos resíduos (erros) encontrado para esta RNA.

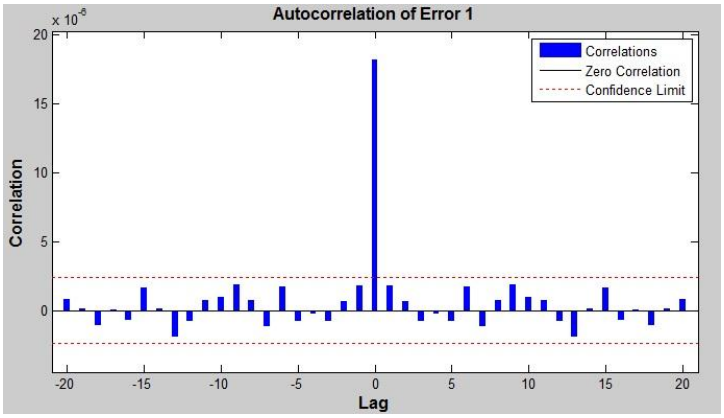


Figura 27 - Autocorrelação do erro rede NARX.
Fonte: gerado no *Matlab* R20013b.

Na Figura 27, verifica-se os erros de previsão em relação ao tempo. Só há um valor fora do intervalo de confiança que ocorre no *lag* zero. Isso significa que não há correlação significativa nos erros de previsão do consumo de memória. Neste caso, todos os valores do erro estão dentro dos limites de confiança de 95%, em torno de zero, de modo que o modelo parece ser adequado.

Na Tabela 15 observa-se os parâmetros da topologia do modelo de previsão encontrado para a rede neural NARX.

Tabela 15 - Parâmetros da rede neural NARX.

Parâmetros	**Valor**
Neurônios	9
Camada oculta	1
Épocas de treinamento (iterações)	6
RMSE	0,001538
R^2 estimado	0,99772

Fonte: elaborada pela autora.

Dessa forma, a partir dos resultados encontrados na construção da rede neural NARX, pode-se constatar que ao contrário da MLP, foi possível construir uma arquitetura NARX adequada para a predição com 233 observações.

Assim encerram-se as análises referentes aos modelos lineares de séries temporais ARIMA e ARMAX, bem como os modelos não lineares através das RNAs MLP e NARX. Na Tabela 16, apresenta-se os resultados do RMSE referente a cada modelo.

Tabela 16 - RMSE dos modelos ARIMA, ARMAX, MLP e NARX.

	ARIMA (RMSE)	ARMAX (RMSE)	MLP (RMSE)	NARX (RMSE)
CPU	0,019468	-	-	-
DISCO	0,05869	-	-	-
MEMÓRIA	0,004979	0,004713	0,003177	0,001538

Fonte: elaborada pela autora

Observando os resultados, nota-se que os modelos preditivos de redes neurais artificiais foram os que tiveram o desempenho mais acurado, segundo o critério da raiz quadrada do erro quadrático médio.

Percebe-se, a partir dos resultados gerados neste estudo, que a rede neural NARX possui condições de ser utilizada como técnica base para um mecanismo de provisionamento de recursos em ambientes de nuvem.

4.6. Comentários gerais do capítulo

Com o objetivo de se efetuar uma análise comparativa de técnicas estatísticas utilizando séries temporais e redes neurais, foram encontrados vários modelos de previsão ARIMA e ARMAX, e também foram realizados inúmeros treinamentos para a RNAs MLP e NARX.

Primeiramente aplicou-se nos dados, a metodologia de *Box e Jenkins* ARIMA, como essa modelagem é univarida, teve que se efetuar os testes estatísticos ADF, KPSS, FAC FACP, análise dos resíduos, etc, em cada uma das variáveis separadamente, e assim a partir disto encontrar o melhor modelo de previsão ARIMA.

Para se aplicar o modelo multivariável ARMAX, foi necessário efetuar a correlação entre as variáveis exógenas e a variável endógena, além disso foi necessário aplicar o método *backward* que consiste em começar a análise do modelo com vários regressores das variáveis exógenas e assim se retirar sucessivamente as interações de ordem mais elevada.

Para fazer o treinamento das RNAs MLP e NARX, efetuou-se várias simulações a fim de encontrar a topologia com melhor acurácia.

Todas as análises estatísticas ocorridas na seção anterior visavam definir qual das técnicas utilizadas possui a melhor capacidade preditiva no escalonamento de recursos

computacionais utilizados por um servidor de internet em um ambiente de nuvem computacional.

Por fim, os resultados obtidos revelam que a previsão de melhor qualidade, foi obtida através da topologia de uma rede neural NARX, conforme os resultados da Tabela 16.

No próximo capítulo serão descritas as considerações finais desse trabalho, além de sugestões para futuras pesquisas.

5. CONSIDERAÇÕES FINAIS E RECOMENDAÇÕES

O presente capítulo apresenta uma síntese das principais conclusões que foram sendo inferidas ao longo dos capítulos que compõem o decorrente estudo, tanto em nível de investigação teórica e científica, como em nível de aplicação prática.

A seguir serão descritas, de forma sucinta, algumas ações a serem desenvolvidas em consequência dos resultados obtidos e serão efetuadas recomendações que podem sustentar o desenvolvimento de futuros trabalhos científicos.

5.1. Considerações Finais

Por meio da computação em nuvem, tanto usuários como as organizações, acessam os serviços sob demanda, sem ter a necessidade de saber a localização destes, ou seja, os usuários movem suas informações e aplicações para a nuvem, acessando-as de forma simples, sem haver um custo de investimento ou instalação de uma estrutura física local.

Conforme já mencionado no início deste trabalho, um dos problemas existentes na arquitetura de computação em nuvem refere-se ao provisionamento de recursos em servidores. Assim, este teve por objetivo demonstrar a viabilidade da aplicação técnicas estatísticas, como séries temporais e redes neurais, como base para um mecanismo de alocação de recursos em ambientes de nuvem.

No presente estudo foi modelado o consumo de cpu, disco e memória de um servidor de internet da empresa *Google*, no período de Maio de 2011 a partir de metodologias distintas: ARIMA, ARMAX, MLP e NARX.

Os modelos efetuados com base na metodologia de redes neurais artificiais apresentaram resultados mais acurados em termos preditivos do que resultados obtidos com modelos lineares, isso se deve possivelmente a sofisticação destes modelos.

Uma das principais dificuldades na utilização de redes neurais artificiais para a previsão da variável memória foi à determinação da arquitetura ótima para a rede MLP. Uma vez que a literatura encontrada é subjetiva para a escolha do número de camadas e do número de neurônios em cada camada. Pesquisadores como Hecht-Nielsen (1989), afirmam que com apenas uma camada oculta já é possível modelar uma função qualquer a partir de dados fornecidos. Para esses autores, a camada oculta deve ter por volta de 2i+1 neurônios, onde i é o número de variáveis de entrada. Para Cybenko (1989), a rede necessita duas camadas ocultas.

Outra dificuldade encontrada foi conseguir achar a melhor topologia de uma rede neural multicamadas (MLP) com 233 amostras. Como não houve um modelo com boa acurácia, partiu-se para uma nova topologia com um total de 600 observações, com isso chegou-se a um bom modelo a partir da arquitetura de RNA MLP.

Os resultados obtidos revelam que a previsão de melhor qualidade ocorreu através da topologia de uma rede neural NARX, composta por 2 neurônios na camada de entrada, 1 camada oculta, sendo esta composta por 6 neurônios e 2 defasagens e por fim 1 neurônio na camada de saída. Assim o modelo NARX, através do algoritmo de treinamento b*ack-propagation*, aparentou atender com maior eficiência a necessidade deste trabalho.

Com isso a proposta abordada neste estudo mostrou-se válida. Uma vez que a utilização de uma rede neural não linear autorregressiva com entrada exógena - NARX, pode ser uma alternativa ao provisionamento em ambientes dinâmicos, ou seja, em ambientes computacionais em que há uma constante variabilidade no sistema.

Com base nos resultados da capacidade preditiva desta técnica estatística, consegue-se fornecer indicativos necessários ao provimento da elasticidade de recursos computacionais, a fim de que o ambiente de computação em nuvem se adapte de acordo com as suas necessidades.

Dessa maneira, caso ocorra algum tipo de situação atípica em uma máquina (servidor), através do modelo NARX, pode-se trabalhar de forma pró-ativa para que os serviços não

sejam afetados ou ainda que os impactos sejam amenizados ao máximo com o intuito de que os clientes não sintam reflexos de perda de desempenho no nível de qualidade dos serviços.

Com isso chega-se à conclusão de que a computação em nuvem pode se beneficiar da utilização de redes neurais NARX, pois esta metodologia possibilitou efetuar a análise das informações contidas no *dataset*, por meio de previsões sobre o padrão do comportamento destes dados no futuro.

5.2. Recomendações para futuros estudos

Durante o desenvolvimento deste trabalho, foram identificadas algumas possibilidades que poderão ser desenvolvidas. Assim, acredita-se que trabalhos futuros poderão ser desenvolvidos em continuidade a este, com intuito de melhorar a predição de recursos computacionais para um melhor provisionamento dinâmico em servidores de internet.

A seguir serão apresentadas sugestões futuras, com base no tema desenvolvido.

- Utilizar o vetor autorregressivo – VAR/ VAR-VEC;
- Utilizar outras topologias de redes neurais;
- Modelar novas topologias de redes que façam o uso de outras variáveis existentes no *dataset* da *Google* e não utilizadas nessa pesquisa;
- Efetuar a implementação de provisionamento dinâmico de recursos computacionais em ambiente de produção;

6. REFERÊNCIAS BIBLIOGRÁFICAS

ALECRIM, E. **O que é Tecnologia da Informação?** Publicado em 24/02/2011. <http://www.infowester.com/ti.php> Acesso em Agosto 2013.

AMANI, Payam; KIHL, Maria; **NARX-based Multi-step Ahead Response Time Prediction for Database Servers.** Department of Electrical and Information Technology Lund University, Sweden. 2012.

ANDRADE, Wany Leydiane Souza**. Estimação de modelos ARIMA/ARIMAX e aplicação em inferência de perdas de propano.** Universidade Federal do Rio Grande do Norte. Centro de Tecnologia. Natal, RN. 2009

ARAÚJO, A.; SANTANA, D.; BRANDÃO, R**. Sistema de Detecção de Intrusos Baseado em redes Neurais Artificiais de Kohonen.** Monografia de Final de Curso, Faculdade Ruy Barbosa, 2005.

ARMBRUST, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. H., Konwinski, A., Lee, G., Patterson, D. A., Rabkin, A., Stoica, I., and Zaharia, M. **Above the clouds: A berkeley view of cloud computing.** Technical report, EECS Department, University of California, Berkeley. 2009.

ARTERO, Almir Olivette. **Inteligência Artificial: teórica e prática.** São Paulo: Editora Livraria da Física. 2009.

BOX, G. E. O.; JENKINS, G.M. **Time series Analysis: Forecasting and Control**. San Francisco: Holden-Day, 1970.

BUENO, R. L. S. **Econometria de Séries Temporais.** São Paulo: Cengage Learning, 2008.

CAMPOS, L. M. L.; **Modelos não lineares recorrentes com capacidade de extrapolação – um estudo para predição do índice de preços ao consumidor (IPC-BR).** IX Simpósio de Excelência em Gestão e Tecnologia. 2012.

CARISSIMI, Alexandre. **Virtualização: da teoria a soluções.** 26º Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos. Rio de Janeiro, RJ. 2008.

CARVALHO, André Ponce de Leon F. de; **Rede Neurais Artificiais.** Departamento de Ciência da Computação. Universidade de São Paulo – USP. < http://www.icmc.usp.br/pessoas/andre/research/neural/>. Acesso em Janeiro 2013.

CHIRIGATI, Fernando. **Computação em Nuvem.** 2009.<http://www.gta.ufrj.br/ensino/eel879/trabalhos_vf_2009_2/seabra/arquitetura.html>. Acesso em Setembro 2013.

COUTINHO, Emanuel F.; FLÁVIO R. C. Sousa; DANIELO G. Gomes; JOSÉ, N. de Souza. **Elasticidade em Computação na Nuvem: Uma Abordagem Sistemática.** 31º Simpósio

Brasileiro de Redes de Computadores e Sistemas Distribuídos – SBRC 2013. Brasília-DF, 2013.

CYBENKO, G. **Mathematical Problems in Neural Computing.** Center for Supercomputing Research and Development, University of Illinois at Urbana Champaign, CSRD Report No. 905, 1989.

FELIX, Alexsandro. **Alguns números expressivos do Google.** Disponível em http://asnews.com.br/alguns-numeros-expressivos-do-google/. Acesso em 20 Dezembro, 2013.

FRANSES, P. H. **Primary Demand for Beer in The Netherlands: An Application of ARMAX Model Specification.** Journal of Marketing Research (JMR), v. XXVIII, n. 2, p. 240–245, 1991.

GOOGLE, Empresa Google Inc. http://www.google.com/intl/pt-BR/about/. Acesso em 20 Dezembro, 2013.

GREGO, Maurício. **Google faz 15 anos. Veja números impressionantes do buscador.** Disponível em http://exame.abril.com.br/tecnologia/album-de-fotos/o-google-faz-15-anos-nesta-sexta-veja-numeros-do-buscador. Acesso em 9 Dezembro, 2013.

GUIMARÃES, Roberta Valente. **Uso de Regressão Logística para previsão de fechamento de operações financeiras: termo de moedas.** Escola Politécnica da Universidade de São Paulo. São Paulo, 2006.

GUJARATI, D. N. **Econometria Básica.** 3. ed. São Paulo: Makron Books, 2000.

HAYKIN, Simon. **Redes neurais: príncipios e práticas.** Trad. Paulo Martins Engel. 2° ed. - Porto Alegre : Bookman. 2001.

HECHT-NIELSEN R.; KUDRICKY A. **Theory of Backpropagation Neural Network**. IEEE International Joint Conference On Neural Networks, Washington D.C., June 18-22, IEEE, Volume I, pp.593-606, 1989.

JUNIOR, Fernando de Jesus Moreira. **Proposta de um método para o controle estatístico de processo para observações autocorrelacionadas.** Dissertação Mestrado em Engenharia da Produção. Universidade Federal do Rio Grande do Sul, UFRGS. Porto Alegre, 2005.

KHAN, Arijit; Xifeng Yan; Shu Tao; Nikos Anerousis. **Workload Characterization and Prediction in the Cloud: A Multiple Time Series Approach**. IFIP/IEEE International Workshop on Cloud Management (CloudMan), 2012.

KARLIK, Bekir; OLGAC, Vehbi. **Performance Analysis of various activation functions in generalized MLP Architectures of Neural Networks.** International Journal of Artificial Intelligence and Expert Systems (IJAE), volume (1). 2002.

KWIATKOWSKI, D., P. C. B. PHILLIPS, P. SCHMIDT; SHIN, Y. **Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root?**, Journal of Econometrics, 54, 159-178, 1992.

LEWICKI, P.; HILL, T. **Statistics methods and applications.** In. [S.l.]: Statsoft, 2008. cap. Neural Networks.

LORI MacVittie. **Automatizando o data center.** http://www.f5networks.com.br/pdf/white-papers/dynamic-provisioning-wp.pdf. Acesso em Fevereiro 2014.

MARANGONI, Pedro Henrique. **Redes Neurais Artificiais para Previsão de Séries Temporais no Mercado Acionário.** 2010. 80 fl. Monografia. Universidade Federal de Santa Catarina, 2010.

MARY, C. Onbile. **Cloud Computing Concerns and Issues.** Publicado em 13/ 10/ 2012. <www.onbile.com/info/what-cloud-computing-means> - Acesso em Setembro 2013.

MIRJALILI, S.; Johor Bahru. **Magnetic Optimizacion Algorithm for training Multi Layer Perceptron.** IEEE 3rd International Conference Communication Software and Networks, p. 42-46, 2011.

MOREIRA, A. P. G. M.; COSTA, P. J. G.; SANTOS, P. J. L. **Introdução à identificação de modelos discretos para sistemas dinâmicos.** Faculdade de Engenharia de Universidade do Porto, Portugal, 2002.

MORETTIN, P. A.; TOLOI, C. M. C. **Análise de Séries Temporais.** 1ª edição. Editora Edgard Blucher. 2004.

OLIVEIRA, P. C. **Séries Temporais: Analisar o Passado, Predizer o Futuro.** Departamento de Engenharia Informática, Universidade de Coimbra, Portugal, 2007.

REGO, Paulo. A. L.; COUTINHO, Emanuel.; SOUSA, José N.; **Estratégias para Alocação Dinâmica de Recursos em um Ambiente Híbrido de Computação em Nuvem**. XI Workshop de Computação em Clouds e Aplicações, 2013.

SCHUBERT, F.; Mendes, R.; WESTPHALL, C. Becker. **Redes Bayesianas para a Detecção de Violação de SLA em Infraestrutura como Serviço.** XI Workshop de Computação em Clouds e Aplicações, 2013.

SEVERO, Diogo da Silva. **Otimização Global em Redes Neurais Artificiais.** Universidade Federal de Pernambuco. Centro de Informática. Recife, 2010.

SLADESCAU, Matthew; ALAN Fekete, Kevin Lee; ANNA Liu WISE, volume 7651 of Lecture Notes in Computer Science, page 368-381. Springer, 2012.

SILVA, A. M. DA. **Utilização de Redes neurais Artificiais para classificação de Spam.** [s.l.] Centro Federal de Educação Tecnológica de Minas Gerais, 2009.

SILVA, Letícia da Costa. **Utilizando ARMAX para estimar a influência do PIB, SELIC e INFLAÇÃO no faturamento de empresas do setor siderúrgico com as ações da bolsa.** Universidade de Brasília. Departamento de Administração. Brasília – DF. 2011.

SOUZA, Francisca Mendonça; ALMEIDA, Silvana Gonçalves; SOUZA, Adriano Mendonça; LOPES, Luis Felipe Dias; ZANINI, Roselaine Ruviaro .; **Previsão do preço da gasolina para a região sul do Brasil.** IJIE – Iberoamerican Journal of industrial Engineering. v.3, n.1, pg 234-248, julho, 2011.

TÁPIA, Milena, Redes Neurais Artificiais: **Uma Aplicação na Previsão de Preços de Ovos,** Dissertação de mestrado, UFSC, 2000.

TAURION, Cezar. **Cloud Computing: computação em nuvem: transformando o mundo da tecnologia da informação.** Rio de janeiro: Brasport, 2009.

_____.**O que é elasticidade em cloud computing? (Software, Open Source, Soa, Innovation, Open Standards, Trends)**. <http://goo.gl/qidBs> Acesso em Julho 2013.

VASCONCELLOS, M. A. S.; ALVES, D. **Manual de econometria.** São Paulo. Atlas, 2002.

WOLFF, L. **Relação entre as dez principais bolsas de valores do mundo.** Dissertação de mestrado. Santa Maria, RS, 2011.