UNIVERSIDADE ESTADUAL DO MARANHÃO – UEMA
CENTRO DE CIÊNCIAS TECNOLÓGICAS – CCT
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO E SISTEMAS – PECS

LUIZ OTÁVIO CORDEIRO FONTENELLE GRAÇA

REDES NEURAIS APLICADAS AO CÁLCULO DE AJUSTE DO LANÇADOR DE FOGUETES NÃO-GUIADOS

São Luís
2019

LUIZ OTÁVIO CORDEIRO FONTENELLE GRAÇA

REDES NEURAIS APLICADAS AO CÁLCULO DE AJUSTE DO LANÇADOR DE FOGUETES NÃO-GUIADOS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia da Computação e Sistemas da Universidade Estadual do Maranhão como requisito para obtenção do título de Mestre em Engenharia da Computação.

Orientador: Prof. Dr. Areolino de Almeida Neto

Coorientador: Prof. Dr. Alexandre Garcia

São Luís

2019

Graça, Luiz Otávio Cordeiro Fontenelle.
Redes neurais aplicadas ao cálculo de ajuste do lançador de foguetes não-guiados / Luiz Otávio Cordeiro Fontenelle Graça. – São Luís, 2019.

84f

Dissertação (Mestrado) - Curso de Engenharia de Computação e Sistemas, Universidade Estadual do Maranhão, 2019.

Orientador: Prof. Dr. Areolino de Almeida Neto
Coorientador: Prof. Dr. Alexandre Garcia

1.Foguetes não-guiados. 2.Ajuste do lançador. 3.Simulação de trajetórias. 4.Redes neurais. I.Título

CDU: 004.032.26:629.76

Elaborado por Giselle Frazão Tavares - CRB 13/665

LUIZ OTÁVIO CORDEIRO FONTENELLE GRAÇA

REDES NEURAIS APLICADAS AO CÁLCULO DE AJUSTE DO LANÇADOR DE FOGUETES NÃO-GUIADOS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia da Computação e Sistemas da Universidade Estadual do Maranhão como requisito para obtenção do título de Mestre em Engenharia da Computação.
Orientador: Prof. Dr. Areolino de Almeida Neto
Coorientador: Prof. Dr. Alexandre Garcia

Aprovado em: 26/04/2019

Prof. Dr. Areolino de Almeida Neto
Universidade Federal do Maranhão

Prof. Dr. Luis Carlos Costa Fonseca
Universidade Estadual do Maranhão

Prof. Me. Guilherme da Silveira
Instituto de Aeronáutica e Espaço

Prof. Dr. Ivanildo Silva Abreu
Universidade Estadual do Maranhão

Dedico este trabalho aos meus pais, que nunca mediram
esforços para contribuir com a minha
formação como ser humano.

Agradecimentos

Aos meus pais, por terem me concedido vida, saúde e exemplo de superação de adversidades.
À minha namorada e amigos, pelo incentivo e paciência nos momentos em que estive ausente.
Aos amigos e colegas de mestrado, pelo companheirismo e ajuda nesta jornada.
Ao meu orientador, prof. dr. Areolino de Almeida Neto, pela sabedoria e pelas valiosas lições, que levarei para o resto da vida.
Ao meu coorientador, dr. Alexandre Garcia, pelo tempo prestado e ajuda indispensável.
À Universidade Estadual do Maranhão e aos docentes do Programa de Pós-Graduação em Engenharia da Computação e Sistemas, por abrirem as portas para um horizonte profissional rico em oportunidades.
Ao Instituto Tecnológico de Aeronáutica, por inspirar a busca de uma carreira acadêmica de sucesso e orgulho.
À FAPEMA, pelo apoio financeiro.
A todos os profissionais do CLA, IAE e outras instituições ligadas ao DCTA, que, de alguma forma, contribuíram para a conclusão desta pesquisa.

" Existem muitas hipóteses em ciência que estão erradas.
Isso é perfeitamente aceitável, elas são a abertura para achar as que estão certas".
(Carl Sagan)

Resumo

Esta pesquisa objetivou o projeto, a implementação e a avaliação de uma rede neural artificial que calcula os ângulos do lançador de foguetes não-guiados do Centro de Lançamento de Alcântara (CLA). Para isso, contou-se com o auxílio de um simulador de lançamentos que usa modelos dinâmicos com seis graus de liberdade, em um método de compensação antissimétrica iterativa. A rede foi treinada a partir de dados de trajetórias simuladas e o desempenho do sistema proposto foi comparado com o método do *software* Guará. A comparação foi realizada por meio de diversos testes que consideraram condições aleatórias hipotéticas de vento e ângulos variados de azimute e elevação nominais. Para a geração e comparação das trajetórias compensadas pelos dois métodos, foi utilizado o mesmo simulador, que foi validado em uma comparação com *software* ROSI – *Rocket Simulation*, amplamente utilizado no Instituto de Aeronáutica e Espaço em campanhas de lançamento. O projeto da rede neural, o simulador e as equações utilizadas pelo programa Guará foram implementados no *software* Matlab. O CLA precisa garantir a segurança nas operações de lançamento de foguetes, evitando quaisquer acidentes, perdas ou imprevistos. Para isso, o setor de Segurança de Voo visa a manter o veículo em uma trajetória que resulte em um ponto de impacto próximo ao desejado. O vento é um elemento que influencia o atingimento desse ponto de impacto. Para tentar anular sua influência, o método utilizado atualmente pelo Setor de Segurança de Voo do CLA é o ajuste dos ângulos de azimute e elevação do lançador por meio do *software* Guará. Como resultado, nos testes realizados, a aplicação do método proposto produziu ajustes que levaram o veículo a um ponto de impacto mais próximo do nominal.

Palavras-chave: Foguetes não-guiados, ajuste do lançador, simulação de trajetórias, redes neurais.

Abstract

This research aimed at designing, implementing and evaluating an artificial neural network that calculates the angles of the unguided rocket launcher in Alcântara Launch Center (CLA). For this purpose, a launch simulator that uses dynamical models with six degrees of freedom was used, in an iterative antisymmetric compensation method. The network was trained from simulated trajectory data and the performance of the proposed system was compared with the Guará software method. The comparison was performed through several tests which considered hypothetical random wind conditions and varied nominal azimute and elevation angles. For the generation and comparison of the trajectories compensated by the two methods, the same simulator was used, which was validated in a comparison with the software ROSI - Rocket Simulation, widely used in the Aeronautics and Space Institute in launching campaigns. The neural network, the simulator and the equations used by Guará software were implemented in Matlab software. CLA needs to ensure the safety of rocket launch operations, avoiding any accidents, losses or unforeseen events. For this end, the Flight Safety sector aims to keep the vehicle on a trajectory that results in an impact point close to the desired one. The wind is an element that influences the attainment of that impact point. To try to override its influence, the method currently used by the CLA Flight Safety Sector is to adjust the azimuth and elevation angles of the launcher through Guará software. As a result, in the tests performed, the application of the proposed method produced adjustments that brought the vehicle to an impact point closer to the nominal one.

Keywords: Unguided rockets, launcher adjustment, trajectory simulation, neural networks.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AEB	Agência Espacial Brasileira
VLS	Veículo Lançador de Satélites
VLM	Veículo Lançador de Microssatélites
CLA	Centro de Lançamento de Alcântara
CLBI	Centro de Lançamento da Barreira do Inferno
SVO	Setor de Segurança de Voo
SARA	Satélite de Reentrada Atmosférica
DVO	Setor de Dinâmica de Voo
IAE	Instituto de Aeronáutica e Espaço
PI	Ponto de Impacto
TTE	*Tower Tilt Effect*
RNA	Rede Neural Artificial
MLP	Multilayer Perceptron
VSB-30	Veículo de Sondagem Brasileiro
UROCKS	*Unguided Rocket Simulation*
ROSI	*Rocket Simulation*
RTS	*Rocket Trajectory Simulator*
PIP	Ponto de Impacto Perturbado
PIC	Ponto de Impacto Compensado
PIN	Ponto de Impacto Nominal
PIR	Ponto de impacto resultante
URLANN	*Unguided Rocket Launcher Adjustment via Neural Networks*

Lista de Símbolos

$DPI_{L,F,C}$	Desvio do ponto de Impacto lateral, frontal ou de cauda
$VB_{L,F,C}$	Vento balístico lateral, frontal ou de cauda
$\delta_{L,F,C}$	Desvio de vento unitário lateral, frontal ou de cauda
C	Número de camadas de vento
c	Índice da camada
$f(c)$	Peso da camada c
$F(h)$	Fator de pesagem
h	Altitude
h_c	Altitude-limite da camada c
DPI_h	Desvio do ponto de impacto causado por um vento constante até a altitude h
DPI_{20}	Desvio do ponto de impacto causado por um vento constante até 20 km
$v(c)$	Média das medidas de velocidade do vento internas à camada c
$v_{LO}(c)$	Componente leste-oeste da média de velocidades do vento na camada c
$v_{NS}(c)$	Componente norte-sul da média de velocidades do vento na camada c
α	Direção do vento em relação ao norte
VB_{LO}	Componente leste-oeste do vento balístico
VB_{NS}	Componente norte-sul do vento balístico
VB	Intensidade do vento balístico
θ	Arco cuja tangente é a relação entre a componente leste-oeste e a componente Norte-sul do vento balístico
φ	Ângulo entre o vento balístico e o ângulo de azimute nominal
α_0	Ângulo de azimute nominal
α_b	Direção do vento balístico em relação ao norte
D_x	Distância entre o lançador e a projeção do PI compensado na trajetória nominal
D_n	Distância entre o lançador e o PI nominal, ou alcance
$\Delta\alpha$	Variação em azimute
TTE	Valor do *Tower Tilt Effect*
E_0	Elevação nominal
D_i	Distância do lançador até o PI perturbado

$\boldsymbol{x}$	Vetor de entradas em um neurônio ou uma rede neural
x_i	i-ésima componente de um vetor de entradas
$\boldsymbol{w_j}$	Vetor de pesos aliados a um neurônio j
v_j	Entrada em um neurônio *j*
f_j	Função de ativação de um neurônio *j*
y_j	Saída do neurônio *j*
$\boldsymbol{d}$	Vetor de saídas desejadas em uma rede neural
N	Número de exemplos de treinamento
d_k	Saída desejada no k-ésimo neurônio da camada de saída
Δw	Variação em um determinado peso
t	Época de treinamento
η	Taxa de aprendizagem
e	Erro em um determinado neurônio
J	Função custo
W	Número total de pesos na rede
ε	Erro máximo admitido para o treinamento
∇J	Gradiente da função custo
$\nabla^2 J$	Matriz Hessiana da função custo
μ	Termo de regularização do algoritmo Levenberg-Marquardt
I	Matriz identidade
R	Coeficiente de correlação
$\bar{d}$	Média dos elementos de um padrão de saída desejado
$\bar{y}$	Média dos elementos de um padrão de saída calculado
s_d	Desvio padrão dos elementos de saída desejados
s_y	Desvio padrão dos elementos de saída calculados
NSi	Coordenada norte-sul de um ponto de impacto
LOi	Coordenada leste-oeste de um ponto de impacto
Azi	Direção de impacto
D	Distância de impacto
M_e	Matriz que contém os padrões de entrada da rede neural em colunas
M_s	Matriz que contém os padrões de saída da rede neural em colunas
dNS_1	Desvio causado pelo vento na direção norte-sul, na trajetória nominal
dLO_1	Desvio causado pelo vento na direção leste-oeste, na trajetória nominal

NS_p	Coordenada norte-sul do ponto de impacto perturbado
LO_p	Coordenada leste-oeste do ponto de impacto perturbado
NS_n	Coordenada norte-sul do ponto de impacto nominal
LO_n	Coordenada leste-oeste do ponto de impacto nominal
NSc_1	Coordenada norte-sul do primeiro ponto de impacto compensado
LOc_1	Coordenada leste-oeste do primeiro ponto de impacto compensado
dNS_2	Distância no sentido norte-sul do ponto de impacto nominal até o resultante do primeiro ajuste
dLO_2	Distância no sentido leste-oeste do ponto de impacto nominal até o resultante do primeiro ajuste
NSr_1	Coordenada norte-sul do ponto de impacto resultante do primeiro ajuste
LOr_1	Coordenada leste-oeste do ponto de impacto resultante do primeiro ajuste
NSc_2	Coordenada norte-sul do primeiro ponto de impacto compensado corrigido
LOc_2	Coordenada leste-oeste do primeiro ponto de impacto compensado corrigido

Sumário

1 Introdução

O desenvolvimento e a autonomia tecnológicos são estratégicos para a soberania e o desenvolvimento econômico de um país. Nesse contexto, a conquista do espaço, por demandar alto desenvolvimento tecnológico, gera retornos para a solução de problemas de um país como o Brasil, que possui vasta extensão territorial, extensas florestas tropicais, fronteiras e costas marítimas e um grande volume de recursos naturais. Os benefícios derivados do investimento nesse setor vão desde a expansão das telecomunicações, acompanhamento de alterações no meio ambiente, meteorologia, até vigilância nas fronteiras do território (AEB, 2018).

Com vistas à conquista de um bom patamar tecnológico aeroespacial, a Agência Espacial Brasileira (AEB) formula e coordena projetos da Política Espacial Brasileira, como o desenvolvimento do veículo lançador de microssatélites (VLM). Em paralelo, é de grande importância o desenvolvimento e o lançamento de foguetes de sondagem e treinamento, pois estes possibilitam a prática de ideias com risco e investimento baixos, servindo de base tecnológica para projetos maiores, como o próprio VLM. Estes foguetes vêm desempenhando um papel importante na prática aeroespacial brasileira, pois permitiram a ampliação da infraestrutura de fabricação mecânica e testes de componentes, desenvolvimento de motores e propelentes, produção de equipamentos elétricos, pirotécnicos e computacionais, treinamento das equipes envolvidas, além da sua primordial função, que é o lançamento de cargas úteis científicas.

O Centro de Lançamento de Alcântara (CLA) tem capacidade de lançar não somente veículos lançadores de satélites/microssatélites (controlados), como também foguetes de sondagem e treinamento (não-guiados ou não-controlados). O setor de Segurança de Voo (SVO) do CLA é responsável por garantir que o lançamento seja seguro em todas as suas fases. No caso dos foguetes não-controlados, é importante, em termos de segurança, fazer com que o impacto do veículo ocorra dentro de uma área de dispersão calculada antes do lançamento. Para isso, a SVO estabelece e aplica procedimentos que asseguram que os riscos estejam dentro do esperado.

Algumas missões de lançamento envolvem a recuperação da carga útil, o que naturalmente aumenta o interesse em um impacto que seja o mais próximo possível do ponto nominal ou desejado. O vento é um dos fatores que dificultam o atingimento deste objetivo. Para compensar a sua influência, a SVO aplica um procedimento de ajuste dos ângulos de lançamento por meio do *software* Guará. Inconsistências e imprecisões desta metodologia demonstram que

existem oportunidades de melhoria. É desse tema que deriva o objetivo deste trabalho.

A realização de atividades espaciais requer o desenvolvimento de tecnologias de ponta e costuma exigir de cientistas conhecimento aprofundado acerca de matérias como dinâmica de voo de foguetes, mecânica dos fluidos, aerodinâmica, além de combustão, telecomunicações, meteorologia, entre outros. O aprendizado de máquina, de certa forma, pode ajudar a reduzir a necessidade do domínio profundo de matérias específicas para realização de algumas tarefas isoladas. Técnicas como redes neurais artificiais vêm auxiliando engenheiros e cientistas há várias décadas a modelar, compreender e resolver problemas diversos, com a ajuda de computadores. Com o contínuo e acelerado desenvolvimento da computação, atualmente é possível processar grandes quantidades de informações, mapear dados e identificar padrões não triviais para a percepção humana.

Este trabalho apresenta a integração entre uma rede neural e um simulador de lançamentos de foguetes não-guiados, para a obtenção dos ângulos de lançamento necessários para compensar os efeitos do vento no Centro de Lançamento de Alcântara.

1.1 Objetivos

1.1.1 Gerais

Desenvolver e aplicar uma rede neural artificial destinada ao ajuste iterativo do lançador de foguetes não guiados, com o auxílio de um simulador de lançamentos de seis graus de liberdade e avaliar suas vantagens e desvantagens em relação ao método utilizado atualmente pelo *software* Guará.

1.1.2 Específicos

- Verificar se a rede neural é capaz de fazer abstrações do comportamento do foguete a partir dos dados de simulações de lançamento;
- Avaliar, a partir de testes propostos, o desempenho do sistema com aplicação da rede neural na tarefa de calcular o ajuste do lançador;
- Comparar, em cada um dos testes, os desempenhos do método proposto e do método utilizado pelo programa Guará na tarefa de levar o foguete para um ponto de impacto mais próximo do nominal;

- Analisar as vantagens e desvantagens do método proposto.

1.2 Justificativa

Fazem parte das tarefas do operador da Segurança de Voo do CLA estabelecer e aplicar procedimentos específicos que minimizem os riscos de lançamentos ou que assegurem que estes estejam em conformidade com as regras de segurança estabelecidas (AEB, 2007). Uma tentativa de aproximar o ponto de impacto real do nominal contribui para estas finalidades.

Por vezes, em operações de lançamento de experimentos de microgravidade, é necessário realizar o resgate da carga útil, como no projeto SARA – Satélite de Reentrada Atmosférica (AEB, 2012) e como cita Da Mata (2017), em relação aos objetivos de uma missão de lançamento. Portanto, torna-se importante o desenvolvimento de práticas e melhorias que visem o aumento da proximidade entre o ponto de impacto real do veículo no oceano e o ponto de impacto nominal (planejado), estabelecido previamente, para que seja mais viável a recuperação da carga útil.

Atualmente, o *software* Guará é utilizado pela Segurança de Voo para o ajuste do lançador de foguetes não-guiados. No entanto, a compensação antissimétrica (KRAMER, 1973), teoria na qual a metodologia deste programa baseia-se, apresenta equívocos fundamentais em suas premissas. Paralelamente, os cálculos desempenhados pelo programa Guará para a previsão dos desvios causados pelo vento apresentam divergências em relação a dados provenientes do mesmo simulador empregado na pesagem de vento que o programa utiliza. Isto sugere que as simplificações adotadas por Lewis (1949), que são utilizadas pelo programa, culminam em inconsistências nos resultados que o *software* oferece. Além destes fatos, o cálculo do ângulo de elevação do lançador, na fase final da metodologia do programa, é fruto de uma aproximação imprecisa entre o alcance atingido e o valor desse ângulo. Portanto, o método do *software* Guará demonstra que existe abertura para melhorias na compensação dos efeitos do vento.

Em contrapartida, o algoritmo apresentado neste trabalho toma por base o desvio do ponto de impacto de foguetes obtido por simulações, sem a necessidade de realizar a pesagem do vento. Assim, o sucesso deste método em situações reais dependeria, em grande parte, do nível de fidelidade à realidade fornecido pelos dados do simulador utilizado. Além desta modificação, é proposta a aplicação de uma rede neural treinada a partir de dados provenientes de simulações para determinar os ângulos do lançador. Dessa forma, outro fator contribuinte para o sucesso do método proposto é o aprendizado efetivo da rede neural em relação ao

comportamento do foguete, sem a necessidade do uso de simplificações.

Uma vantagem do método proposto é a agilidade promovida pelo uso de um simulador de seis graus de liberdade para mensurar os efeitos do vento. Este fato torna possível a realização do ajuste iterativo dos ângulos do lançador para um mesmo perfil de vento medido, o que permite reduzir o erro da compensação antissimétrica rapidamente.

1.3 Trabalhos Relacionados

Esta pesquisa utilizou, entre outras fontes, trabalhos que explanam metodologias de cálculo de ajuste de lançadores de foguetes não-guiados, para compensação dos efeitos do vento. Buscou-se entender as teorias que embasam esse cálculo e procedimentos correlatos. Foi realizado um levantamento de trabalhos que analisam e corrigem métodos estabelecidos ou que propõem procedimentos alternativos para tal.

O principal meio de ajuste do lançador encontrado é o método da compensação antissimétrica, abordado por Kramer (1973). Este é executado atualmente pela Segurança de Voo do CLA, por meio do *software* Guará (YAMANAKA; GOMES, 2001). A primeira etapa da metodologia deste programa consiste em calcular o desvio que o vento causa no ponto de impacto desejado. Para isso, utiliza-se a técnica de pesagem de vento, de Lewis (1949). A segunda etapa é o cálculo dos ângulos necessários para atingir o ponto de impacto de deslocamento equivalente, mas no sentido contrário, ou seja, o ponto antissimétrico, referido por Da Mata (2017) como “compensado”. Em teoria, este ajuste deveria compensar os desvios calculados.

A pesagem de vento, apresentada por Lewis (1949), consiste em ponderar os efeitos das camadas de vento na trajetória de um veículo sem controle de atitude. Atualmente, este procedimento é realizado no IAE com o emprego de um simulador de trajetórias. O objetivo da pesagem é adquirir meios para calcular o desvio que um perfil de vento causaria em trajetórias de foguetes que serão lançados.

James e Harris (1961) analisam algumas hipóteses simplificadoras do método de Lewis, que resultam em erros na metodologia. No entanto, faltam na literatura disponível trabalhos que avaliem a precisão deste procedimento em comparações com o próprio simulador de lançamentos utilizado na pesagem.

Da Mata (2017) fez uma descrição detalhada do método utilizado pelo *software* Guará e, posteriormente, demonstrou as imprecisões no princípio da compensação antissimétrica executada pelo programa. Devido a isso, o autor propôs, por meio do método Mosca, três

modificações no algoritmo que compensa os efeitos do vento: considerar não-linear a relação entre o alcance de impacto e o ângulo de elevação do lançador, atualizar constantemente o desvio de vento unitário e realizar a compensação antissimétrica de forma iterativa. Estas contribuições aplicadas simultaneamente, no entanto, demonstraram ausência de resultados positivos expressivos, quando comparados à metodologia original do programa Guará. Por isso, o autor desenvolve o Método Baseado em Voos, em que são utilizados dados reais de lançamento para a obtenção dos parâmetros de segurança: desvios de vento unitário e curva de pesagem de vento.

Uma extensa busca na literatura de trabalhos acerca do ajuste de lançadores de foguetes não-guiados demonstrou que o acesso a esse tipo de informação é de alta restrição. Esse controle ocorre devido ao fato desses documentos terem, usualmente, caráter de segurança nacional, por serem relatórios técnicos produzidos por centros espaciais ou instituições de natureza semelhante. Por isso, houve dificuldade na obtenção de descrições técnicas aprofundadas a respeito dos procedimentos utilizados por outros países para o ajuste de lançadores. Dessa forma, os trabalhos relacionados limitaram-se aos descritos nesta seção.

1.4 Organização do Trabalho

Este capítulo introduziu e contextualizou o tema desta pesquisa, apresentando seus objetivos e justificativa, além de uma breve passagem pelos trabalhos relacionados ao tema.

O Capítulo 2 apresenta uma revisão teórica sobre lançamento de foguetes, com foco nos conceitos relacionados ao ajuste do lançador de foguetes não-guiados e outras atividades de segurança de voo no Centro de Lançamento de Alcântara. A metodologia de cálculo do programa Guará é descrita em detalhes e são expostas as possibilidades de melhoria.

No terceiro capítulo, são introduzidos os conceitos principais relacionados às redes neurais. Esse capítulo também destaca os aspectos técnicos de projeto acerca dessa ferramenta.

Em seguida, o Capítulo 4 descreve com detalhes a metodologia proposta. Inicialmente, são apresentados os materiais utilizados. Nesta etapa, também são descritos todos os passos do desenvolvimento do trabalho com detalhes e são descritos os códigos produzidos, tanto para a reprodução do modelo atual de cálculo do ajuste do lançador, quanto o modelo neural desenvolvido para a mesma finalidade. Nessa etapa, a rede é treinada a fim de realizar abstrações a respeito do comportamento do veículo, sem considerar a perturbação do vento, para que ela seja capaz de retornar ajustes necessários para que o foguete possa atingir um ponto de impacto determinado, dadas condições específicas de vento. Ainda nesse capítulo, são apresentados os

esquemas de execução dos códigos.

No Capítulo 5, são relatados os resultados dos testes aplicados, que levam em consideração situações hipotéticas de lançamento de foguetes não-guiados.

O Capítulo 6 evidencia as vantagens, contribuições e limitações da metodologia proposta. Ainda nesse capítulo, são propostos trabalhos futuros.

2 Lançamento de Foguetes Não-Guiados

2.1 Contextualização

Veículos não controlados (estabilizados aerodinamicamente) têm como predominante finalidade a realização de experimentos de microgravidade com cargas úteis científicas (foguetes de sondagem) e o treinamento de equipes envolvidas em operações de lançamento (PALMÉRIO, 2017). São, geralmente, de menor porte em relação aos veículos controlados e não dispõem de meios para correção automática da sua trajetória em voo, diferente de lançadores de satélites, como o VLS e o VLM. Assim, sua estabilização é aerodinâmica e o seu voo realizado em dois regimes: propulsado e balístico (PALMÉRIO, 2017).

Os veículos suborbitais são aqueles capazes de cruzar o limite da atmosfera terrestre (100 km) por alguns minutos, possibilitando a execução de experimentos de microgravidade (DA MATA, 2017). Com a sua reentrada na atmosfera, atingem o fim de seu voo em um ponto de impacto no oceano. A Figura 1 apresenta alguns exemplos de foguetes brasileiros de sondagem.

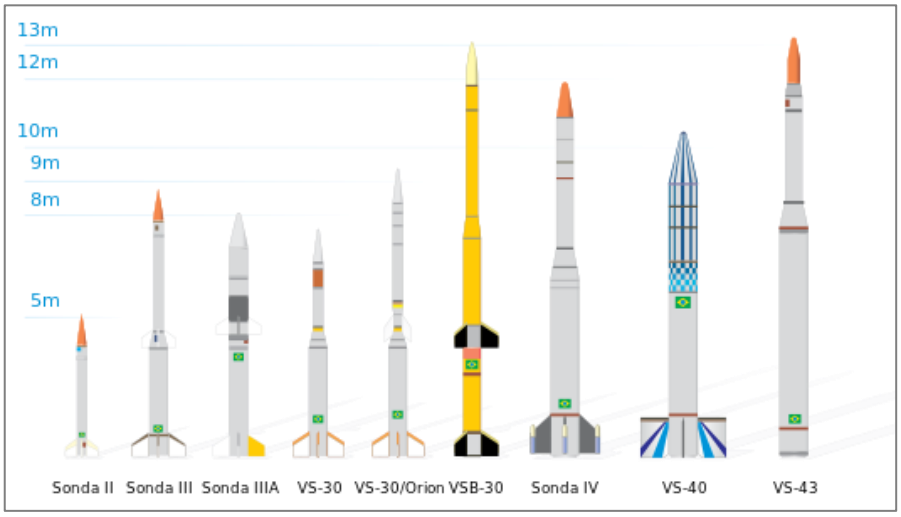

Figura 1 - Exemplos de foguetes de sondagem (AEB, 2012)

O Centro de Lançamento de Alcântara – CLA (Figura 2) é um dos dois centros de lançamento de foguetes existentes no Brasil. O outro é o CLBI - Centro de Lançamento da Barreira do Inferno. O CLA possui estrutura para realizar lançamentos tanto de veículos de grande porte

e controlados, quanto de veículos não-controlados.

Figura 2 - Centro de Lançamento de Alcântara (AEB, 2018)

Em operações de lançamento de veículos não-guiados, o setor de Segurança de Voo (SVO) do CLA é responsável por fazer com que o comportamento do foguete lançado descreva uma trajetória parecida à trajetória previamente estabelecida (trajetória nominal) e atinja um ponto de impacto (PI) próximo ao previsto. (DA MATA; FALCÃO, 2016).

Setor de Segurança de Voo

A cada operação de lançamento, é realizado um conjunto de atividades essenciais, como tratamento de dados, meteorologia, funcionamento da carga útil, localização do foguete, ajuste do lançador, entre outras. Parte delas fica sob a responsabilidade do setor de Segurança de Voo do CLA (GARCIA, 2007). Dentre as responsabilidades deste setor e seus operadores, estão a identificação, a análise e a avaliação de riscos decorrentes do lançamento de um veículo, no que diz respeito à integridade física de pessoas, das propriedades e do meio ambiente. É também tarefa da SVO a eliminação dos perigos ou redução destes a níveis aceitáveis por meio de procedimentos operacionais específicos (AEB, 2007).

Quando ocorrem preparações para a realização de missões de lançamento (campanhas) de foguetes não-controlados, o setor de Dinâmica de Voo (DVO) do Instituto de Aeronáutica e Espaço (IAE) realiza simulações de lançamento em função das características do foguete que será utilizado e dos requisitos da missão, para gerar trajetórias chamadas nominais, que desconsideram a ação do vento. A SVO deve obter os dados da DVO para eleger a melhor trajetória nominal, para ser seguida pelo foguete, com base nos regimentos de segurança estabelecidos pela Agência Espacial Brasileira e pelo próprio CLA. Devido à posição geográfica privilegiada do município de Alcântara, as chances de haver impacto de foguetes no continente são reduzidas, pois os veículos são lançados em direção ao mar (Figura 3).

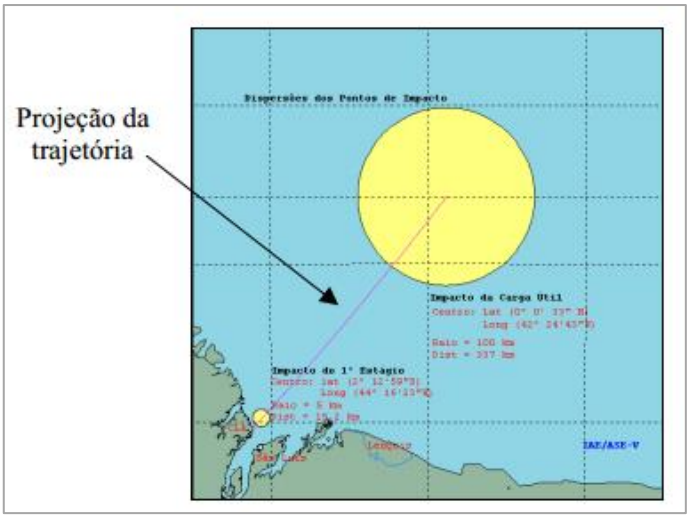

Figura 3 - Exemplo de trajetória e região de possíveis impactos (GARCIA, 2007)

Cabe à SVO, entretanto, interditar a região no oceano, bem como o espaço aéreo acima dela, dentro da qual é previsto o impacto, para evitar acidentes com embarcações e aeronaves (DA MATA, 2017). Um dos principais fatores contribuintes para a dispersão da trajetória e do ponto de impacto do veículo lançado são os efeitos provocados pelo vento no momento do lançamento (JAMES; HARRIS, 1961). O vento pode ser mensurado e ter seu efeito compensado através do ajuste do lançador universal (Figura 4), que se move com dois graus de liberdade (DA MATA; FALCÃO, 2016).

Figura 4 - Lançador Universal (PALMÉRIO, 2007)

A posição inicial do foguete no lançador é baseada nos estudos da trajetória nominal (GARCIA, 2007). De posse dos dados relativos ao vento em diversas camadas de altitude, os operadores da Segurança de Voo são capazes de realizar procedimentos para corrigir os ângulos relativos à posição do foguete no lançador, para compensação da influência do vento. Estes ângulos são chamados de azimute e elevação (Figura 5).

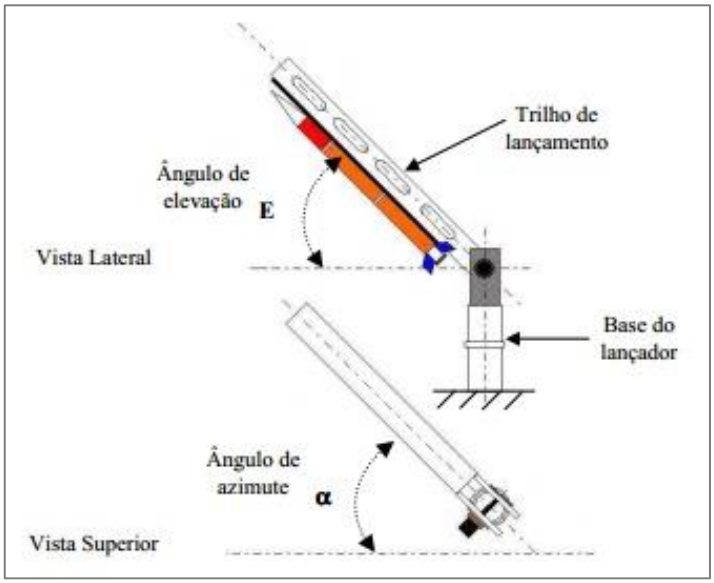

Figura 5 - Ângulos de Azimute e Elevação (GARCIA, 2007)

A correção é realizada baseada nos cálculos executados pelo *software* Guará (YAMA-NAKA; GOMES, 2001), operado na SVO antes do lançamento.

2.2 Ajuste do Lançador: *Software* Guará

O *software* Guará possui uma metodologia dividida em três etapas: 1 – obtenção do perfil de vento (medidas de velocidade e direção distribuídas verticalmente), 2 – previsão do ponto de impacto e 3 – compensação do efeito do vento (DA MATA, 2017). A previsão do PI baseia-se no método de Lewis (1949), que utiliza os conceitos de vento balístico (VB) e desvio de vento unitário (δ). Vento balístico refere-se a um vento constante que causa o mesmo efeito de todo um perfil de ventos medidos até uma determinada altitude. E o desvio de vento unitário é o efeito no ponto de impacto (distância medida em km) de um foguete causado por um perfil constante e uniforme de ventos de 1 m/s em uma determinada direção. Os deslocamentos do PI são calculados pela Eq. 1.

$$DPI_{L,F,C} = VB_{L,F,C} \times \delta_{L,F,C} \quad (1)$$

onde DPI refere-se ao desvio do ponto de impacto (km), o índice L refere-se à direção lateral, perpendicular à trajetória, e os índices F e C referem-se à mesma direção da trajetória nominal (azimute nominal), porém em sentidos opostos: o vento frontal (F) age contra o sentido do deslocamento horizontal do foguete, enquanto que o vento de cauda (C) atua a favor.

2.2.1 Recepção do Perfil de Vento

Atualmente, o CLA utiliza dois meios para medição dos dados relacionados ao vento, para as operações de ajuste do lançador: uma torre anemométrica de $72\ m$ de altura e balões meteorológicos, chamados de radiossonda (DA MATA, 2017). A torre coleta os dados do vento de superfície (até $100\ m$ de altitude) por meio de anemômetros distribuídos em seis níveis: $48, 52, 58, 70, 85$ e $112\ m$ (Figura 6).

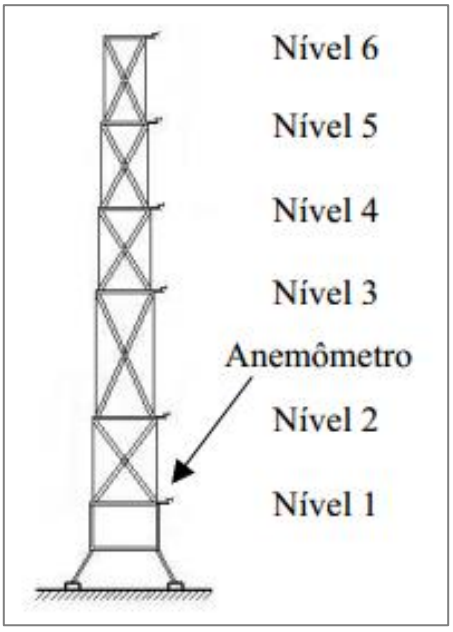

Figura 6 - Torre anemométrica (GARCIA, 2007)

A torre anemométrica é capaz de fornecer suas medições a cada 20 s para a Segurança de Voo. A importância dessas medidas deve-se pelo fato de ventos mais baixos serem mais influentes no comportamento do foguete, devido à menor velocidade na fase inicial do voo do veículo. Segundo Fisch (1999), ventos de até $1000\ m$ são responsáveis por 88% da influência na trajetória dos foguetes, sendo que, acima de $5000\ m$, os ventos realizam cerca de 3% do

desvio.

O balão meteorológico (Figura 7) é solto na atmosfera e chega a uma altitude entre 20.000 e 30.000 m. Ao longo do seu voo, são captadas informações de altitude, velocidade e direção do vento, entre outros. Essas informações são enviadas a cada 2 s para a SVO.

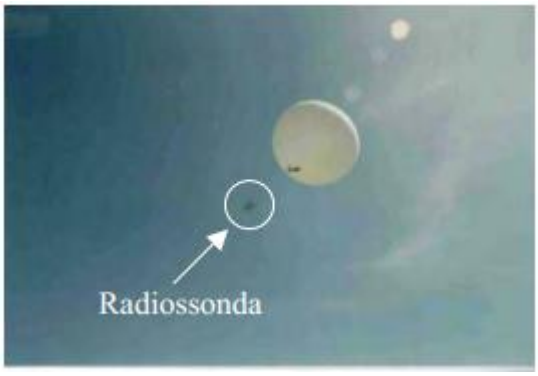

Figura 7 - Balão Meteorológico (GARCIA, 2007)

De posse do perfil vertical de vento, das características do veículo a ser lançado e das informações da trajetória nominal, o operador da Segurança de Voo executa o programa Guará, que realiza as rotinas matemáticas (seções 2.2.2 e 2.2.3) necessárias para o cálculo dos valores corrigidos dos ângulos de azimute e elevação do lançador. Os arquivos com as medições da torre anemométrica e das radiossondas ficam salvos posteriormente nos computadores do CLA em arquivos de texto, contendo, cada um, milhares de linhas com valores de velocidade, direção, altitude, temperatura, pressão atmosférica, entre outros.

2.2.2 Desvio do Ponto de Impacto

O método de Lewis (1949) divide um dado perfil de vento em C camadas e atribui a cada camada c um peso $f(c)$, utilizado para calcular o vento balístico VB. Os pesos são obtidos por meio dos fatores de pesagem de vento $F(h)$, que são calculados em função de uma determinada altitude h, de acordo com a Eq. 2, onde h_c e h_{c-1} são as altitudes-limite das camadas c e $c-1$, respectivamente. O fator de pesagem de vento $F(h)$ é obtido por meio da Eq. 3.

$$f(c) = F(h_c) - F(h_{c-1}) \tag{2}$$

$$F(h) = \frac{DPI_h}{DPI_{20}} \tag{3}$$

onde DPI_h é o desvio do ponto de impacto gerado por um perfil de vento constante até a altitude h e DPI_{20} é o desvio devido à influência de um perfil de vento uniforme até a altitude de $20.000\ m$. O vento uniforme utilizado para encontrar estes desvios são as médias mensais de velocidade e direção, proveniente de estudos feitos no CLA (GARCIA, 2007). Pereira (2002) apresenta estes valores na Tabela 1.

Tabela 1 - Médias mensais de velocidade e direção do vento no CLA. (PEREIRA, 2002)

Mês	**Velocidade (m/s)**	**Direção (graus) em relação ao norte**
Janeiro	7,2	90
Fevereiro	8,0	90
Março	7,2	103
Abril	8,0	95
Maio	8,3	93
Junho	8,4	103
Julho	8,4	97
Agosto	8,2	93
Setembro	8,6	95
Outubro	8,4	94
Novembro	7,2	92
Dezembro	7,4	94

O conjunto de todos os fatores $F(h)$, para todas as altitudes referentes aos limites das C camadas, é chamado de curva de pesagem de vento. Os desvios, atualmente, são calculados em função de dados obtidos por meio de simulação computacional das trajetórias (DA MATA, 2017) e a pesagem de vento é realizada por um programa a parte.

Para cada camada do perfil, é calculada a média $v(c)$ das medidas internas de velocidade do vento. Cada média possui uma componente no eixo norte-sul e outra no eixo leste-oeste, de acordo com a convenção meteorológica. Esta indica a forma de determinar a direção do vento, como ilustra a Figura 8. Garcia (2007) apresenta os cálculos para determinar a velocidade e a direção do vento balístico. São apresentadas, ainda, as equações que determinam os deslocamentos causados pelo vento unitário e o desvio do PI. Primeiramente, as componentes do vento médio de uma determinada camada na convenção meteorológica são calculadas de acordo com

as Eq. 4 e 5.

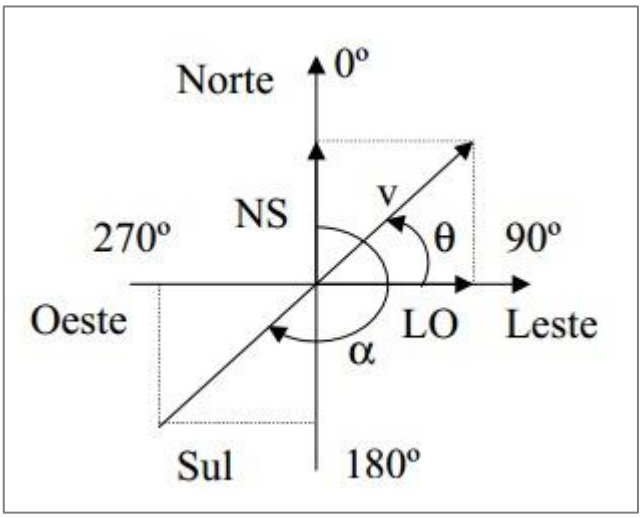

Figura 8 - Convenção meteorológica (GARCIA, 2007)

$$v_{LO}(c) = -v(c)sen(\alpha) \quad (4)$$

$$v_{NS}(c) = -v(c)\cos(\alpha) \quad (5)$$

em que $v_{LO}(c)$ refere-se à direção leste-oeste, $v_{NS}(c)$ refere-se à direção norte-sul e α é o ângulo de direção do vento em relação ao norte. A soma das componentes de mesma direção, de todas as camadas, ponderadas pelos respectivos pesos, resulta nas componentes do vento balístico, das Eq. 6 e 7, cuja intensidade é calculada pela Eq. 8.

$$VB_{LO} = \sum_{N} v_{LO}(c)f(c) \quad (6)$$

$$VB_{NS} = \sum_{N} v_{NS}(c)f(c) \quad (7)$$

$$VB = \sqrt{VB_{LO}^2 + VB_{NS}^2} \quad (8)$$

onde VB_{LO} é a componente do vento balístico na direção leste-oeste e VB_{NS} é a componente na direção norte-sul. O cálculo do ângulo α_b, de direção do vento balístico, está em função das suas componentes, como mostrado na Figura 9 e nas Eq. 9 a 14:

$$(a) \quad VB_{LO} < 0, \qquad VB_{NS} < 0 \quad \rightarrow \quad \alpha_b = \theta \quad (9)$$

$$(b)\ VB_{LO} < 0, \quad VB_{NS} = 0 \quad \rightarrow \quad \alpha_b = 90° \tag{10}$$

$$(c)\ VB_{NS} > 0 \quad \rightarrow \quad \alpha_b = 180° - \theta \tag{11}$$

$$(d)\ VB_{LO} > 0, \quad VB_{NS} = 0 \quad \rightarrow \quad \alpha_b = 270° \tag{12}$$

$$(e)\ VB_{LO} \geq 0, \quad VB_{NS} < 0 \quad \rightarrow \quad \alpha_b = 360° - \theta \tag{13}$$

$$\theta = tg^{-1}\left(\frac{|VB_{LO}|}{|VB_{NS}|}\right) \tag{14}$$

onde a Eq. 11 é aplicada em qualquer caso em que a componente VB_{NS} é positiva. Ou seja, sendo VB_{LO} negativo, positivo ou nulo, o ângulo α_b é calculado da mesma forma.

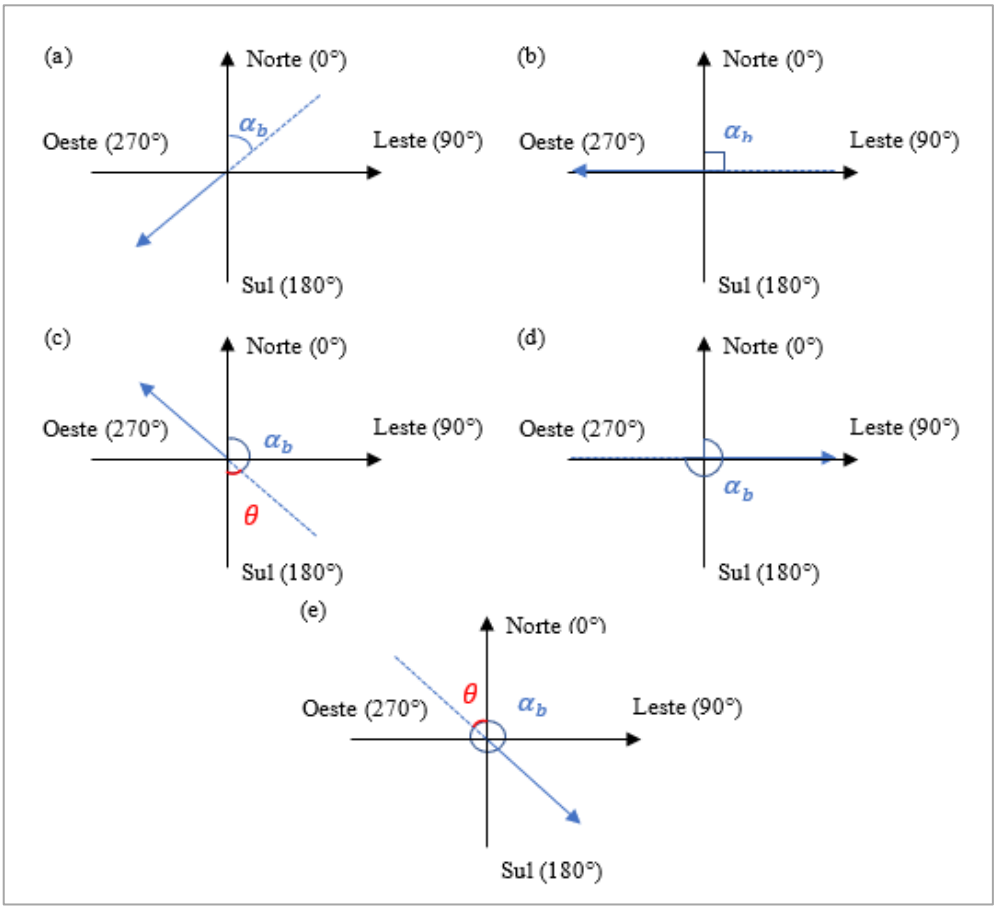

Figura 9 - Direção do vento balístico na convenção meteorológica (GARCIA, 2007)

Prosseguindo com o método de Lewis (1949) para a previsão do desvio do ponto de impacto, são considerados três perfis de vento unitário (velocidade de 1 m/s ao longo de toda a sua extensão), que atuam, cada um, exclusivamente em um sentido: lateral, frontal e de cauda. Seus ângulos de direção são calculados em função do azimute nominal de lançamento α_0 de acordo com as Eq. 15 a 17.

$$\alpha_{ul} = \alpha_0 + 90° \quad (15)$$

$$\alpha_{uf} = \alpha_0 \quad (16)$$

$$\alpha_{uc} = \alpha_0 + 180° \quad (17)$$

onde α_{ul} é a direção do vento lateral; α_{uf} é a direção do frontal e α_{uc} é a direção do vento de cauda. É importante notar que o vento lateral é perpendicular ao deslocamento nominal. Verifica-se também que, seguindo a convenção meteorológica, o vento de cauda é orientado no mesmo sentido do deslocamento horizontal, em oposição ao vento frontal, que, apesar de situar-se na mesma direção, segue o sentido contrário. Os ventos unitários efetuam desvios, cada um em sua direção: lateral (δ_L), frontal (δ_F) ou de cauda (δ_C). Estes desvios representam o quanto um vento constante de 1 m/s é capaz de deslocar o PI na direção e no sentido em que age.

Multiplicando os efeitos de cada vento unitário pelas componentes do vento balístico que atuam nas suas respectivas direções, são encontrados os desvios: lateral (DPI_L) e frontal (DPI_F) ou de cauda (DPI_C). Para o cálculo das componentes do vento pelas Eq. 18 a 20, é utilizado o ângulo φ, que é obtido pela Eq. 21 e encontra-se entre o vento balístico e a trajetória nominal, como ilustrado na Figura 10.

$$VB_L = VB \cdot sen(\varphi) \quad (18)$$

$$VB_F = VB \cdot cos(\varphi) \quad (19)$$

$$VB_C = VB \cdot cos(\varphi) \quad (20)$$

$$\varphi = \alpha_0 - \alpha_b \quad (21)$$

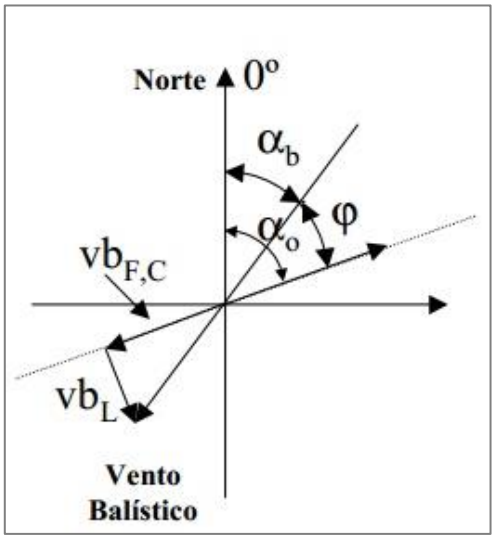

Figura 10 - Direção do vento balístico (GARCIA, 2007)

Os deslocamentos que as componentes do vento causam no PI são calculados pelas Eq. 22 a 24.

$$DPI_L = VB_L \cdot \delta_L \quad (22)$$

$$DPI_F = VB_F \cdot \delta_F \quad (23)$$

$$DPI_C = VB_C \cdot \delta_C \quad (24)$$

Nas Eq. 25 e 26, são estabelecidas as condições que determinam se o efeito do vento é frontal ou de cauda, respectivamente (GARCIA, 2007).

$$DPI_F > 0; \cos(\varphi) > 0 \quad (25)$$

$$DPI_C < 0; \cos(\varphi) < 0 \quad (26)$$

2.2.3 Compensação do Efeito do Vento

Da Mata (2017) aborda o conceito de compensação antissimétrica (KRAMER, 1973), utilizado pelo *software* Guará para o ajuste do lançador. Os novos ângulos a serem utilizados apontam para uma direção diferente daquela que culminaria no ponto de impacto nominal PI_n. Neste caso, o veículo lançado tenderia a cair em um ponto de impacto compensado PI_c, cujo deslocamento em relação a PI_n deveria ser antissimétrico ao ponto PI_i, que é resultante do efeito do vento. Ou seja, as distâncias lateral e frontal do PI_c a partir do PI_n seriam iguais em módulo e opostos em sinais (sentidos diferentes), como mostra a Figura 11.

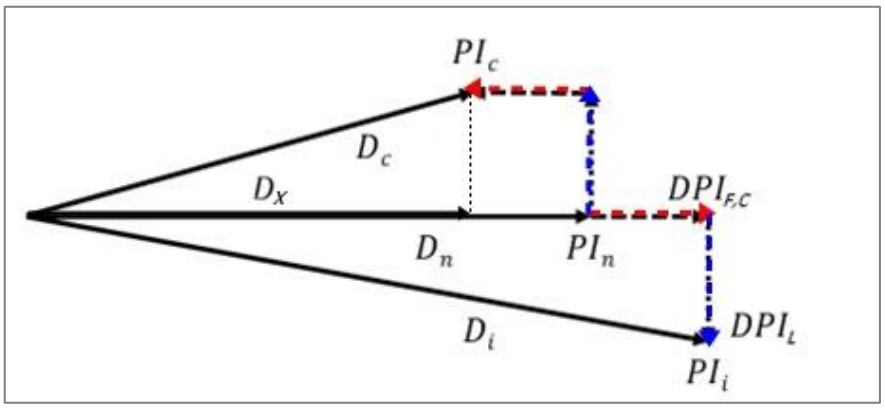

Figura 11 - Compensação Antissimétrica (DA MATA, 2017)

Assim, em teoria, os desvios deveriam ser anulados e então, o ponto de impacto

compensado seria aproximado do nominal. O próximo passo é encontrar os valores de azimute e elevação que originam o ponto de impacto PI_c. Para isso, é necessário calcular a distância D_x entre o lançador e a projeção do PI_c no eixo que se encontra no plano horizontal e que passa pelo lançador e pelo PI_n. Caso a condição da Eq. 25 seja verdadeira, a distância do lançador até a projeção do PI_c é calculada pela Eq. 27. No caso em que a Eq. 26 é verdadeira, essa distância é obtida pela Eq. 28.

$$D_x = D_n - DPI_F \tag{27}$$

$$D_x = D_n + DPI_c \tag{28}$$

em que D_n é a distância ou alcance de impacto da trajetória nominal.

Segundo Garcia (2007), o azimute compensado é calculado pela soma entre o azimute nominal e a variação $\Delta\alpha$, como indicado nas Eq. 29 e 30.

$$\alpha_c = \alpha_0 + \Delta\alpha \tag{29}$$

$$\Delta\alpha = tan^{-1}\left(\frac{DPI_L}{D_x}\right) \tag{30}$$

em que α_c é o azimute compensado.

Para o ajuste do ângulo de elevação, é introduzido o conceito de *tower tilt effect* (TTE), determinado pela Eq. 31, que significa a variação do alcance para cada grau de variação de elevação. Este efeito é, portanto, medido em km/graus e calculado em função de E_0 (elevação nominal) e D_n.

$$TTE = \frac{Dn}{90 - E_0} \tag{31}$$

Assim, a elevação corrigida, dada pela Eq. 33, é calculada em função do TTE e da distância D_c até o ponto de impacto compensado. Esta última é definida pela Eq. 32.

$$D_c = \sqrt{DPI_L^2 + D_x^2} \tag{32}$$

$$E_c = 90 - \frac{D_c}{TTE} \tag{33}$$

Limites de Ajuste

Os limites dos ângulos de azimute (em relação ao norte) e elevação e suas correções para o Centro de Lançamento de Alcântara são definidos como segue:

- Azimute: entre $-20°$ e $120°$, com correção máxima de $30°$ (LINDO;VIANA, 2017);
- Elevação: até $84°$, para veículos sem sistema de terminação de voo (CLA, 2017) e até $86°$, para veículos com sistema de terminação de voo, com correção máxima de $4°$ (LINDO, 2017).

2.2.4 Inconsistências da Metodologia do Programa

Da Mata (2017) expõe algumas hipóteses simplificadoras utilizadas por Lewis (1949) para a determinação do desvio do ponto de impacto:

- A proporcionalidade entre a média de vento de uma determinada camada e o efeito que ela produz;
- O efeito que uma camada produz no ponto de impacto não depende do vento em outras camadas;
- O efeito final de um perfil de vento é equivalente à soma dos efeitos que cada camada produz separadamente;
- Consideração de que o vento balístico produz o mesmo efeito que um perfil completo de ventos no desvio do ponto de impacto;
- Consideração do TTE constante.

Em relação ao cálculo dos ângulos de azimute e elevação compensados ou ajustados, uma simplificação é considerar que a variação do alcance do PI possui uma relação linear com a elevação. Ou seja, se, por exemplo, um determinado ajuste de elevação ΔE produz um desvio frontal $DPI_{frontal}$, um ajuste de $2\Delta E$ produziria um desvio de $2DPI_{frontal}$.

Algumas observações podem ser feitas no que diz respeito às hipóteses mencionadas. Segundo James e Harris (1961), um vento de cauda tende a fazer o foguete apontar para cima e um vento frontal tende a fazer o foguete apontar para baixo. Por isso, é natural concluir que a atitude de um foguete em uma determinada camada dependeria da influência que o vento exerce em uma camada anterior. Assim, o veículo estaria em diferentes atitudes e posições, devido à influência prévia das camadas por onde passou, oferecendo áreas de contato diferentes para os ventos que atuam sobre sua superfície, o que influenciaria as forças aerodinâmicas atuantes no foguete. Portanto, torna-se razoável considerar a possibilidade de uma camada anterior

influenciar nos efeitos de uma camada posterior.

Da Mata (2017) também identifica os três principais problemas com a metodologia do programa Guará: 1 – O deslocamento de vento unitário varia em função da elevação; 2 – a correção do azimute modifica os efeitos do vento e 3 – o *tower tilt effect* é considerado linear, mas essa consideração só é válida em torno do ponto de impacto nominal. Estes problemas acarretam erros consideráveis no resultado do ajuste do lançador. Por exemplo, a mudança dos efeitos do vento com a variação em azimute resulta em um ponto de impacto diferente do esperado, pois as componentes do vento na trajetória compensada são diferentes das componentes que incidem na trajetória nominal, como demonstra a Figura 12. Este resultado contradiz o princípio da compensação antissimétrica.

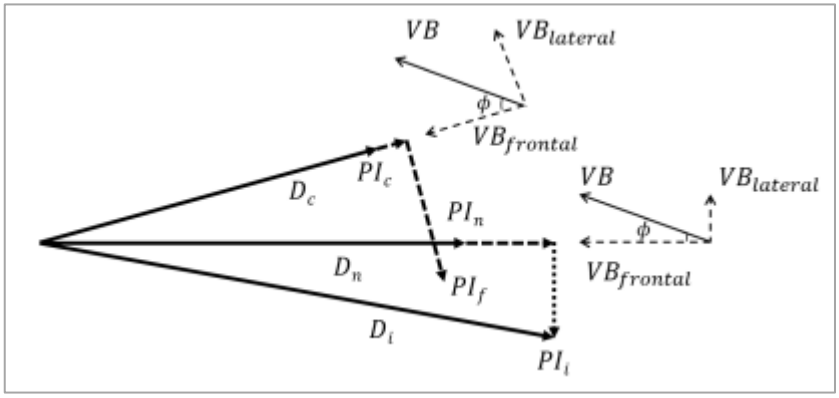

Figura 12 - Vento balístico na trajetória compensada (DA MATA, 2017)

Em relação ao TTE, sua linearidade significa que também é considerada linear a relação entre o alcance de um impacto qualquer e o ângulo de elevação necessário para atingi-lo. Todavia, essa aproximação só não resulta em grandes erros quando é utilizada para calcular a elevação associada a um ponto de impacto cujo alcance é semelhante ao alcance nominal D_n. Ou seja, para pontos de impacto de alcance muito maior ou muito menor que esse valor, esta aproximação é menos eficiente. Neste caso, para que o método utilizado pelo *software* Guará funcione de forma adequada, a distância D_x deve ter um valor próximo a D_n.

Por estes motivos, Da Mata (2017) propôs o método Mosca, que visa eliminar estes erros com as seguintes modificações: 1 – calcular os desvios de vento unitário lateral, frontal e de cauda para a trajetória compensada; 2 – realizar a compensação antissimétrica de forma iterativa e 3 – utilizar uma interpolação do tipo $[a \cdot sin(2x + \phi)] + b$ para aproximar a relação entre o alcance de impacto e o ângulo de elevação. A segunda mudança visa corrigir o ponto de impacto compensado repetidas vezes para um mesmo perfil de vento medido, a fim de reduzir

a distância entre o impacto após o ajuste e o PI nominal. Utilizando um estudo de caso, o método original do programa Guará e o método Mosca foram testados e comparados. Foi verificado que os ajustes produzidos pelos dois assemelhavam-se por valores decimais e que as imprecisões resultantes do ajuste não foram significativamente reduzidas. Portanto, o resultado das modificações, segundo o Da Mata (2017), não foi satisfatório.

Um ponto que não é abordado nessa proposta, entretanto, é que a pesagem de vento, necessária para o cálculo dos desvios causados pelo vento, depende da trajetória. Quando os novos ângulos do lançador são obtidos, o efeito das camadas de vento é diferente. Por isso, o mais adequado seria calcular novamente os pesos das camadas e, consequentemente, o vento balístico, além dos desvios de vento unitário. No entanto, a pesagem do vento é precedida de dezenas de simulações (número de camadas consideradas somado de três). Assim, na proposta do método Mosca, a cada iteração, uma nova pesagem deveria ser realizada. Entretanto, isso poderia resultar na necessidade de serem realizadas centenas de simulações em um intervalo de poucos segundos, entre as atualizações do perfil de vento recebido na SVO.

3 Redes Neurais

3.1 Conceitos Básicos

Uma rede neural artificial (RNA) é uma "máquina adaptativa" projetada para funcionar de forma semelhante ao cérebro humano, quando realiza tarefas do tipo: reconhecimento de padrões, representação do ambiente por imagens, coordenação motora, entre outras (HAYKIN, 2001). Uma RNA é formada por unidades de processamento chamadas de neurônios artificiais, interconectadas por ligações chamadas sinapses, cuja intensidade é indicada numericamente por parâmetros chamados "pesos". A expressão "adaptativa" refere-se à capacidade dos pesos da rede de serem ajustados em um processo de aprendizagem, realizado por algoritmos de treinamento. O objetivo da rede é produzir respostas ou saídas adequadas a estímulos de entrada após o aprendizado.

O neurônio é a unidade fundamental para a operação da rede. Ele consiste em um processador de informações, que recebe um vetor $\boldsymbol{x} = (x_0, x_1, x_2, \dots, x_n)$, multiplica-o de forma escalar (Eq. 34) por um vetor de pesos $\boldsymbol{w_j} = (w_{j0}, w_{j1}, w_{j2}, \dots, w_{jn})$ e aplica uma função de ativação $f(\cdot)$ no resultado (Eq. 35), produzindo uma saída y_j (HAYKIN, 2001). A Figura 13 apresenta uma representação de um neurônio artificial em comparação com um biológico.

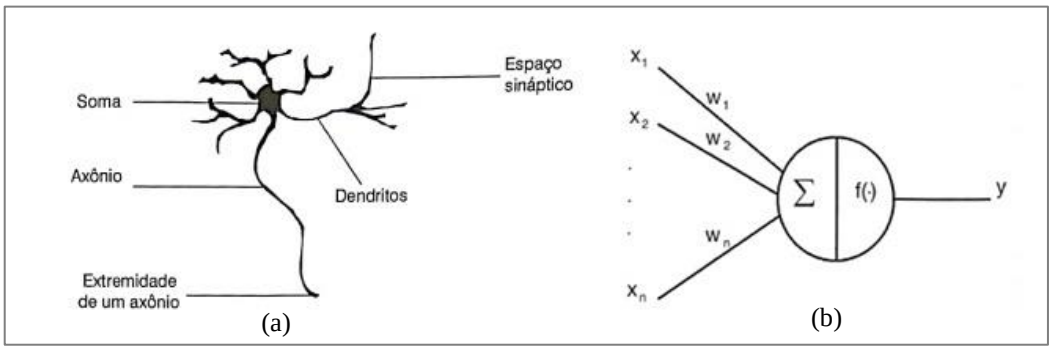

Figura 13 - Neurônio (a) biológico e (b) artificial (BRAGA;CARVALHO;LUDEMIR, 2007)

$$v_j = \boldsymbol{x} \cdot \boldsymbol{w_j} = \sum_{i=0}^{n} x_i w_{ji} \tag{34}$$

$$y_j = f(v_j) \tag{35}$$

onde o índice *j* refere-se ao neurônio, o índice *i* refere-se à conexão e v_j é a entrada do neurônio.

Um conjunto de neurônios interconectados (Figura 14), recebendo e repassando estímulos, caracteriza o funcionamento básico das diversas arquiteturas de redes neurais.

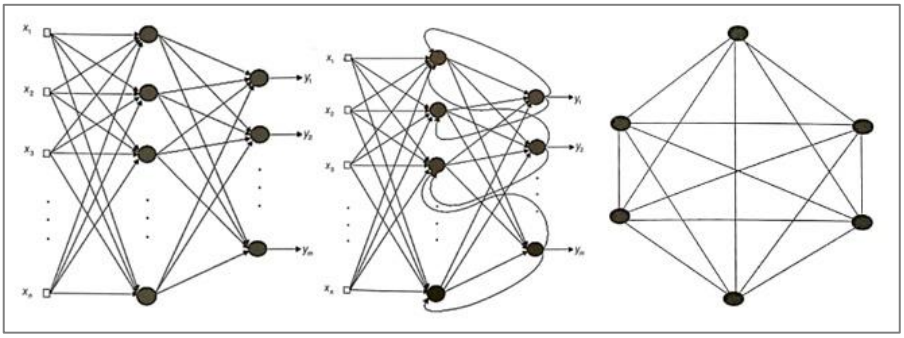

Figura 14 - Exemplos de arquiteturas de redes neurais (BRAGA;CARVALHO;LUDEMIR, 2007)

Redes *Feedforward*

Na rede do tipo *feedforward*, ou "alimentada adiante", os neurônios são distribuídos em camadas (Figura 15), nas quais as informações seguem apenas um sentido de fluxo. As redes são constituídas de uma camada de entrada, uma camada de saída e, possivelmente, camadas ocultas ou intermediárias. Redes desse tipo são estruturas capazes de resolver problemas multivariáveis, dependendo da sua topologia (número de camadas e neurônios). Cybenko (1989) afirma que redes com uma camada intermediária não-linear podem aproximar qualquer função contínua e, com duas camadas, podem aproximar qualquer função.

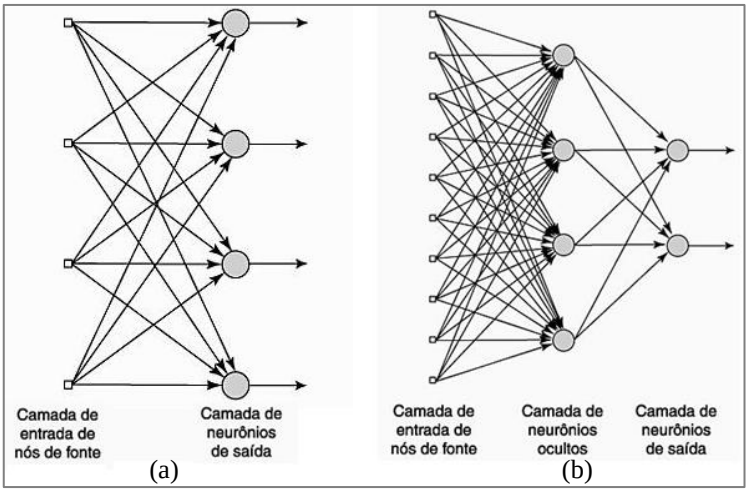

Figura 15 - Exemplos de topologias de redes *feedforward*: (a) de uma camada e (b) de duas camadas (HAYKIN, 2001)

Segundo Braga, Carvalho e Ludemir (2007), o funcionamento de uma rede *feedforward* de múltiplas camadas (por vezes denominada *Multilayer Perceptron*, ou MLP) consiste em sucessivas transformações não-lineares do vetor de entradas, ocorridas nos neurônios das camadas ocultas, usualmente por meio de funções de ativação $f(\cdot)$ sigmoides, como a tangente hiperbólica (Eq. 36), sendo que em aplicações de mapeamento entrada-saída, os neurônios da camada de saída geralmente aplicam transformações lineares (Eq. 37).

$$f_j(v_j) = \frac{e^{v_j} - e^{-v_j}}{e^{v_j} + e^{-v_j}} \tag{36}$$

$$f_k(v_k) = v_k \tag{37}$$

em que j é o índice correspondente a um neurônio de uma camada oculta e k é o índice correspondente a um neurônio da camada de saída.

Redes profundas ou Redes *Deep Feedforward*

O número de camadas de uma rede neural indica a sua "profundidade". Essa é a razão para o nome desse tipo de rede, que possui um número alto de camadas em sua arquitetura e por isso é dita "profunda" (GOODFELLOW; BENGIO; COURVILLE, 2016), como ilustra a Figura 16. Em geral, a adição de camadas na estrutura torna a rede capaz de extrair informações de ordem superior de um conjunto de dados, o que é importante em problemas onde a camada de entrada possui um número de nós de entrada (dimensão) elevado (HAYKIN, 2001).

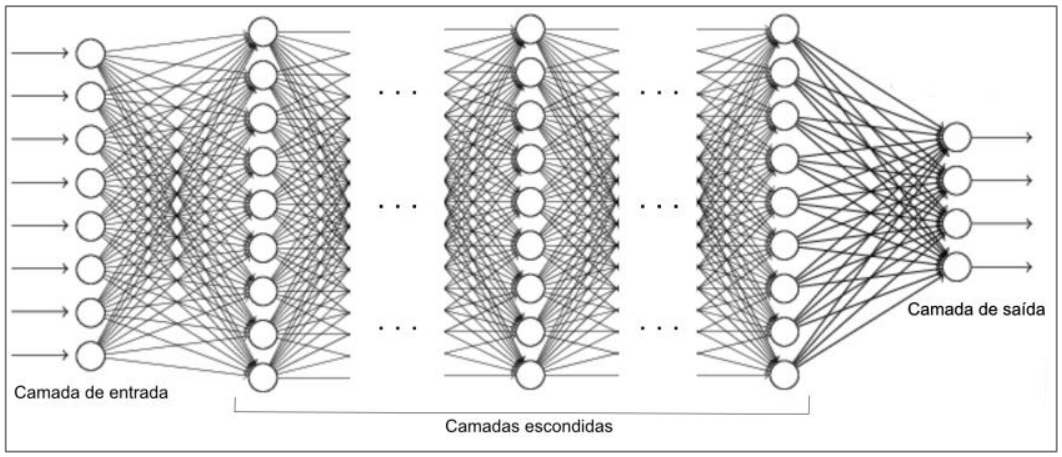

Figura 16 - Rede *deep feedforward* (DE ALMEIDA NETO, 2018)

Segundo Braga, Carvalho e Ludemir (2007), problemas de maior complexidade exigem a aplicação de redes de complexidade equivalente, significando a necessidade de um grande

número de neurônios nas camadas ocultas, o que torna a rede "larga". Entretanto, Daumé III (2016) ressalta que com o aumento do número de camadas, o número de neurônios necessários para a resolução do problema é reduzido.

Aplicações da RNA

Hagan et al. (2014) citam quatro exemplos de aplicações de RNAs, que são: o reconhecimento de padrões, o mapeamento entrada-saída, o *clustering* e a previsão. O reconhecimento de padrões, ou classificação, é uma tarefa em que as entradas da rede devem ser rotuladas em classes. Ou seja, as respostas da rede para cada entrada limitam-se a um número finito de possibilidades. Um exemplo de aplicação seria o diagnóstico de doenças por exames de imagens. O *clustering* é realizado quando se deseja separar dados por similaridade. A previsão, que possui nome autoexplicativo, é realizada quando se deseja estimar valores futuros de uma série temporal.

Quando a rede neural é, por exemplo, projetada para aplicações de mapeamento entrada-saída (também chamada de aproximação de funções, regressão ou modelamento), o aprendizado realizado é do tipo supervisionado (BISHOP, 2006), no qual são apresentados diversos exemplos (vetores de variáveis) de informações de entrada $\boldsymbol{x}$ e as respectivas saídas desejadas correspondentes $\boldsymbol{d}$ de um conjunto de treinamento $\{(\boldsymbol{x}(\boldsymbol{n}), \boldsymbol{d}(\boldsymbol{n}))\}_{n=1}^{N}$, em um modelo de aprendizagem por correção de erro. O objetivo é que a rede neural seja capaz de fazer a aproximação da função que mapeia relações de entrada e saída. Depois disso, a rede deve realizar a generalização, que é a capacidade de apresentar respostas ou saídas coerentes para entradas que não foram utilizadas durante o treinamento (HAYKIN, 2008). Para aplicações de classificação e aproximação de funções, o tipo de rede mais comum utilizado é a rede *feedforward.*

Treinamento

Haykin (2001) apresenta a sequência de eventos que caracteriza as iterações no algoritmo de aprendizagem supervisionada de uma rede neural.

1. Receber um estímulo do ambiente.
2. Sofrer modificações nos pesos de suas conexões como resultado do estímulo.
3. Responder de uma maneira diferente devido às modificações.

O algoritmo de aprendizado supervisionado mais usual em redes neurais, mais especificamente em redes *feedforward*, é o *backpropagation*, ou retropropagação do erro. Trata-se de um aprendizado por correção de erro. Um modo amplamente explorado na literatura é o *batch learning* ou modo *offline* de treinamento, em que todos os exemplos são apresentados à rede antes da atualização dos seus pesos, contabilizando uma época, ou uma iteração. As Eq. 38 a

(47) demonstram este processo de treinamento supervisionado, dividido em duas fases. Toma-se como exemplo as três camadas consecutivas da rede da Figura 17.

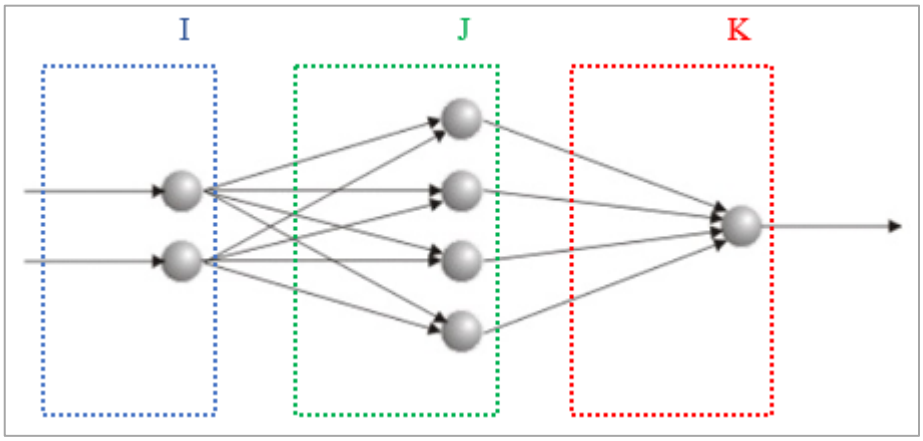

Figura 17 - Divisão da rede em camadas (DE ALMEIDA NETO, 2003)

Fase *Forward*

Nesta etapa, um vetor de entradas $\boldsymbol{x}$ é processado até a saída pelas Eq. 38 a 40, onde se calcula o erro (Eq. 41).

- Camada I:

$$y_i = f_i(x_i w_i) = f(v_i) \tag{38}$$

em que: x_i é o i-ésimo elemento do vetor $\boldsymbol{x}$, que entra no neurônio i da camada I; w_i é o peso da conexão anterior ao neurônio i; f_i é a função de ativação do neurônio i; y_i é a saída do neurônio i.

- Camada J:

$$y_j = f_j\left(\sum_{i=1}^{I} y_i w_{ij}\right) = f_j(v_j) \tag{39}$$

onde: w_{ij} é o peso que conecta o neurônio i ao neurônio j da camada J; f_j é a função de ativação do neurônio j; y_j é a saída do neurônio j.

- Camada K:

$$\hat{y}_k = f_k\left(\sum_{j=1}^{J} y_j w_{jk}\right) = f_k(v_k) \tag{40}$$

$$e_k = d_k - \hat{y}_k \tag{41}$$

Em que w_{jk} é o peso que conecta o neurônio j ao neurônio k da camada de saída K; f_k é a função de ativação do neurônio k; $\hat{y}_k$ é a saída calculada do neurônio k; d_k é a saída desejada no neurônio k e e_k é o erro no neurônio k.

Fase *Backward*

É a fase que calcula a atualização Δw dos pesos da rede, da camada de saída até a camada de entrada. Consiste, primeiramente, na estimação do erro nos neurônios, para posteriormente realizar o cálculo da atualização dos pesos (Eq. 42 a 46).

- Camada K:

$$\Delta w_{jk} = \eta \cdot \sum_{n=1}^{N} e_k(n) \cdot f_k'(v_k(n)) \cdot y_j(n) \tag{42}$$

- Camada J:

$$e_j(n) = -\sum_{k=1}^{K} e_k(n) \cdot f_j'\big(v_j(n)\big) \cdot w_{jk} \tag{43}$$

$$\Delta w_{ij} = -\eta \cdot \sum_{n=1}^{N} e_j(n) \cdot f_j'\left(v_j(n)\right) \cdot y_i(n) \tag{44}$$

- Camada I:

$$e_i(n) = -\sum_{j=1}^{J} e_j(n) \cdot f_i'(v_i(n)) \cdot w_{ij} \tag{45}$$

$$\Delta w_i = -\eta \cdot \sum_{n=1}^{N} e_i(n) \cdot f_i'\big(v_i(n)\big) \cdot x_i(n) \tag{46}$$

onde e_j é o erro estimado no neurônio j, e_i é o erro estimado no neurônio i , n é o índice dos

exemplos de treinamento e η é uma constante que funciona como a taxa de aprendizagem.

A atualização dos pesos das conexões nas camadas é realizada somando o valor atual do peso com as atualizações calculadas (Eq. 47).

$$w(t+1) = w(t) + \Delta w(t) \tag{47}$$

onde t refere-se à iteração ou época.

A resolução do problema concentra-se em encontrar um conjunto de pesos que minimizem uma função custo J. O valor dessa função é calculado a cada época e é obtido, usualmente, pelo erro quadrático médio total, definido pela Eq. 48.

$$J = \frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K} e_k(n)^2 \tag{48}$$

É importante que a função custo seja calculada não apenas para os erros causados pelos dados de treinamento, mas também para um conjunto de "validação". Este conjunto faz parte do grupo total de dados destinados para o projeto da rede, mas não é utilizado no treinamento. Sua finalidade é verificar, a cada iteração ou época, se o valor da função custo está decrescendo também para outros dados que não sejam aqueles utilizados na atualização dos pesos. Um outro conjunto, de "teste", deve ser utilizado para verificar se a rede consegue realizar uma boa generalização.

3.2 Projeto de uma Rede Neural

Carvalho (2011) cita as abordagens mais utilizadas no projeto de redes neurais:

- **Empírica**: Consiste em uma busca cega, baseada em testes e comparações de diversas arquiteturas aleatórias diferentes, apresentando um alto custo de tempo;
- **Meta-heurística**: Gera diversas variações de redes neurais e combina características das redes que apresentam melhor resultado, apresentando um elevado custo computacional;
- **Poda**: A rede é iniciada com uma estrutura complexa, com grande número de neurônios, é treinada e, em seguida, é utilizado um algoritmo para remover conexões e neurônios irrelevantes nas camadas ocultas, como forma de melhorar a capacidade de

generalização da rede;

- **Construtiva**: Nessa abordagem, a rede é iniciada com uma estrutura simples e são adicionados neurônios e conexões até que seja encontrado um desempenho satisfatório para o problema em questão.

Segundo Hagan et al. (2014), o projeto de uma rede neural pode ser dividido em três etapas, apresentadas nas subseções a seguir.

3.2.1 Pré-Treinamento

Nesta fase, ocorre a seleção dos dados, o pré-processamento e a escolha do tipo de rede.

Seleção dos dados

A seleção é feita com a separação dos dados em grupos: de treinamento (por volta de 70% dos exemplos), de validação (cerca de 15%) e de teste (também cerca de 15%). Uma questão pertinente nesta fase é a de quantos dados são necessários para o treinamento. A resposta é que depende da complexidade do problema. Assim, uma função "suave" a ser aproximada não exige uma quantidade grande de exemplos.

Pré-processamento

O pré-processamento tem o papel de facilitar o treinamento, por exemplo, com redução da dimensão da entrada da rede, ou eliminação de elementos redundantes por meio de extração de "características" dos vetores de entrada. Essas características, após serem obtidas, transformam-se em entradas, substituindo o vetor original.

Quando os exemplos de entrada e, consequentemente a camada de entrada, apresentam dimensão elevada, geralmente há dificuldade na resolução do problema, pois um grande número de nós de entrada implica uma grande quantidade de pesos na rede. Nesse tipo de situação, é comum aplicar métodos para reduzir a dimensão da camada de entrada, como a utilização de redes *autoencoders* (HINTON; SALAKHUTDINOV, 2006).

Escolha da rede

Para Carvalho (2011), a capacidade de uma rede neural em modelar a relação entre um conjunto de dados depende de uma boa escolha da arquitetura, que consiste nas funções de ativação e na topologia (número de neurônios e camadas). O número adequado de neurônios nas camadas ocultas de uma rede neural está em função do número de exemplos de treinamento, da quantidade de ruído presente nos exemplos, da complexidade da função a ser aprendida, da distribuição estatística dos dados de treinamento, entre outros fatores. Uma rede com número exageradamente alto de neurônios (mais complexa do que o problema) pode aprender detalhes

excessivos dos exemplos de treinamento, mas tende a fazer uma generalização pobre, não respondendo adequadamente a entradas diferentes daquelas utilizadas no seu processo de aprendizagem. A essa ocorrência, dá-se o nome de *overfitting.* Assim, de forma geral, ocorre uma boa generalização se for verdadeira a condição da Eq. 49 (LIMA, 2016):

$$N > \frac{W}{\varepsilon} \tag{49}$$

em que N é o número de exemplos de treinamento, W é a quantidade de pesos na rede e ε é o erro admitido na fase de treinamento. Braga, Carvalho e Ludemir (2007) atentam para o fato de que, se o objetivo do treinamento da rede é encontrar o conjunto de pesos que resolva o problema, um número superior de pesos em uma rede faz com que exista mais soluções possíveis. Entretanto, isso não significa que a "qualidade" das soluções será satisfatória. Quanto maior o número de neurônios e, consequentemente, de parâmetros, mais difícil é a busca por soluções que se aproximam de forma eficiente da função que gera os dados.

Goodfellow, Bengio e Courville (2016) demonstram que redes profundas apresentam melhor aprendizagem em relação a redes "rasas" (com poucas camadas) e "largas" (com grande número de neurônios e pesos), por permitirem uma quantidade maior de neurônios em sua estrutura antes de causar *overfitting.*

Hagan et al. (2014) recomendam que o número de camadas seja iniciado em um valor baixo e que seja aumentado à medida que for necessário. O número de neurônios, por sua vez, é comum que seja iniciado em valor alto e que depois sejam utilizados algoritmos como regularização Bayesiana ou métodos de *prunning* (poda), para reduzir essa quantidade. Os pesos da rede geralmente são iniciados em valores no intervalo entre $-0{,}5$ e $0{,}5$.

3.2.2 Treinamento da Rede

No treinamento, são realizadas a escolha e a aplicação do algoritmo que fará a atualização dos pesos, até que seja encontrado o conjunto adequado para a resolução do problema. O treinamento supervisionado de redes neurais é predominantemente baseado na minimização do erro na saída, o que constitui um problema de otimização. O algoritmo desta natureza mais usual em redes neurais é o *backpropagation.* Este tipo de treinamento apresenta uma alta dependência da inicialização dos pesos, que devem ser iniciados em valores aleatórios relativamente baixos (GOODFELLOW; BENGIO; COURVILLE, 2016). Segundo Lima (2014), o

critério de parada do treinamento *backpropagation* é constituído, geralmente, por uma combinação entre uma boa capacidade de generalização e a obtenção de um erro menor que um valor admissível, determinado previamente. Uma boa generalização está associada à prevenção contra o *overfitting* pelo uso de um teste de validação durante o treinamento.

Segundo Hagan et al. (2014), para redes com centenas de pesos, o algoritmo de otimização Levenberg-Marquardt é o método de treinamento mais rápido. No entanto, este é um algoritmo que demanda uma grande quantidade de memória, o que pode tornar sua utilização inviável no caso de redes muito grandes. Segundo Braga, Carvalho e Ludemir (2007), os pesos da rede são ajustados conforme a Eq. 50.

$$\Delta w_{ji}(t) = -\left[\nabla^2 J\left(w_{ji}(t)\right) + \mu I\right]^{-1} \nabla J\left(w_{ji}(t)\right) \tag{50}$$

Em que $\nabla^2 J\left(w_{ji}(t)\right)$ é a matriz Hessiana, $\nabla J\left(w_{ji}(t)\right)$ é o gradiente da função custo e μ é um termo de regularização que, quanto mais baixo, mais o algoritmo aproxima-se do método da Descida do Gradiente e, quanto mais alto, mais o algoritmo aproxima-se do método de Gauss-Newton (HAYKIN, 1999). Quando o número de neurônios chega à ordem de 10^3, o algoritmo Levenberg-Marquardt perde sua eficiência. Para redes com esta complexidade, o algoritmo *Scaled Conjugate Gradient* (MILLER, 1993) pode apresentar bom desempenho.

3.2.3 Pós-Treinamento

Hagan et al. (2014) demonstram uma forma de analisar o desempenho da rede em tarefas de regressão após o treinamento, por meio do o cálculo do coeficiente de correlação R, definido pela Eq. 51.

$$R = \frac{\sum_{n=1}^{N}\left(d_n - \bar{d}\right)(y_n - \bar{y})}{(N-1)s_d s_y} \tag{51}$$

onde $\bar{d}$ e $\bar{y}$ são as médias entre as saídas desejadas e as saídas calculadas pela rede, respectivamente, nos neurônios da camada de saída. Esses valores são obtidos pelas Eq. 52 e 53. Os fatores s_d e s_y são os desvios-padrão de d e y, respectivamente, calculados pelas Eq. 54 e 55.

$$\bar{d} = \frac{1}{N}\sum_{n=1}^{N} d_n \tag{52}$$

$$\bar{y} = \frac{1}{N}\sum_{n=1}^{N} y_n \tag{53}$$

$$s_d = \sqrt{\frac{1}{N-1}\sum_{n=1}^{N}\left(d_n - \bar{d}\right)^2} \tag{54}$$

$$s_y = \sqrt{\frac{1}{N-1}\sum_{n=1}^{N}(y_n - \bar{y})^2} \tag{55}$$

Quanto mais próximo de 1 for o coeficiente R, melhor é a aproximação da função-alvo. No caso em que $R = 0$, a rede não consegue fazer um mapeamento entrada-saída correto. Ressalta-se a importância do cálculo do coeficiente de correlação, não só para os dados de treinamento, como para os subconjuntos de validação e testes. A avaliação do coeficiente R e as recomendações após o treinamento são realizadas conforme os critérios da Tabela 2.

Tabela 2 - Avaliações conforme o coeficiente de correlação

Treinamento	**Validação**	**Teste**	**Conclusão e recomendação**
Ruim	Ruim	Ruim	É necessário reconfigurar a topologia da rede
Bom	Ruim	Ruim	Indica *overfitting.* Recomenda-se a diminuição do tamanho da rede e a repetição do treinamento.
Bom	Bom	Ruim	Chama-se extrapolação. O treinamento precisa de mais dados (exemplos).
Bom	Bom	Bom	Treinamento satisfatório.

4 Ajuste do Lançador via Redes Neurais

Diante do exposto em relação às desvantagens e simplificações do método utilizado pelo Guará, este trabalho propõe uma nova forma para o cálculo dos ângulos do lançador. Antes da exposição dos passos para o desenvolvimento do trabalho, é necessário deixar claro que o foco desta pesquisa está nas oportunidades de melhoria do cálculo do ajuste do lançador e não em outras funções que o programa Guará possui. Portanto, nesta etapa não são tratados, por exemplo, critérios para cancelamento de lançamento por segurança. Ressalta-se, ainda, que a metodologia deste trabalho não considera outros fatores que influenciam a dispersão do ponto de impacto além do vento. Portanto, o método ideal de ajuste do lançador seria aquele que levaria o foguete ao ponto de impacto nominal.

Este capítulo está dividido nos seguintes tópicos:

- Materiais utilizados
- Reprodução da metodologia do *software* Guará
- Projeto da rede neural e do ajuste antissimétrico iterativo
- Testes propostos

4.1 Materiais utilizados

O desenvolvimento do método proposto e os testes realizados para comparação com o desempenho do programa Guará utilizaram:

- *Software* Matlab;
- Simulador de lançamento de foguetes com dinâmica de até seis graus de liberdade;
- Código contendo a pesagem de vento e os cálculos realizados pelo programa Guará;
- Computador com as configurações: processador Intel Core i7-7700T 2.9 GHz, memória RAM de 16 GB de e sistema operacional Windows 10;
- Dados de sondagens de vento, realizadas por balões meteorológicos no CLA.
- Dados de um foguete fictício do tipo não-guiado, com características semelhantes ao foguete de sondagem brasileiro VSB-30 (Figura 18).

A rede neural, as equações do programa Guará e o simulador foram implementados no *software* Matlab. É importante salientar que não foi possível utilizar dados reais de um foguete de sondagem devido ao acesso restrito a esse tipo de informação. Os dados reais utilizados nesta

metodologia limitam-se aos arquivos de sondagens de vento.

Figura 18 - Foguete VSB-30 (LUCCA, 2014)

Todas as etapas desta metodologia necessitaram de um simulador de lançamento de veículos não-guiados para geração e validação de dados. A ferramenta utilizada nesta pesquisa foi o simulador UROCKS - *Unguided Rocket Simulator*. Trata-se de uma versão simplificada do simulador RTS, desenvolvido por Da Silveira (2014). Seu comportamento possui grande semelhança com o programa ROSI, que é utilizado atualmente pelo IAE em missões de lançamento.

O simulador UROCKS realiza simulações de trajetórias com um, três e seis graus de liberdade. A dinâmica de um grau de liberdade é utilizada para o movimento do trilho do lançador, onde o foguete movimenta-se apenas em uma direção. O modelo com seis graus de liberdade é utilizado durante todo o voo após a saída do trilho até a reentrada atmosférica. Já o modelo com três graus de liberdade é utilizado desde a fase de reentrada, até o impacto.

Os dados de entrada do simulador relativos ao veículo são propriedades de massa e inércia, características propulsivas, aerodinâmicas, entre outros. É possível realizar simulações considerando a ação do vento ou com vento nulo. Além disso, deve ser especificada a posição do lançador, por meio dos valores dos ângulos de azimute e elevação. Ao executar as simulações, o programa retorna dezenas de parâmetros do voo, dentre os quais destacam-se para este trabalho as coordenadas que indicam a posição do foguete ao longo da trajetória, em relação ao sistema com origem no lançador.

A Figura 19 mostra as semelhanças entre alguns parâmetros de saída dos programas UROCKS e ROSI. Os parâmetros foram obtidos com as simulações de lançamento do mesmo foguete fictício referido na seção 4.1, utilizando os mesmos ângulos do lançador e

desconsiderando a ação do vento. O procedimento detalhado de execução dos modelos utilizados pelo simulador foge do escopo deste trabalho e pode ser encontrado em Da Silveira (2014).

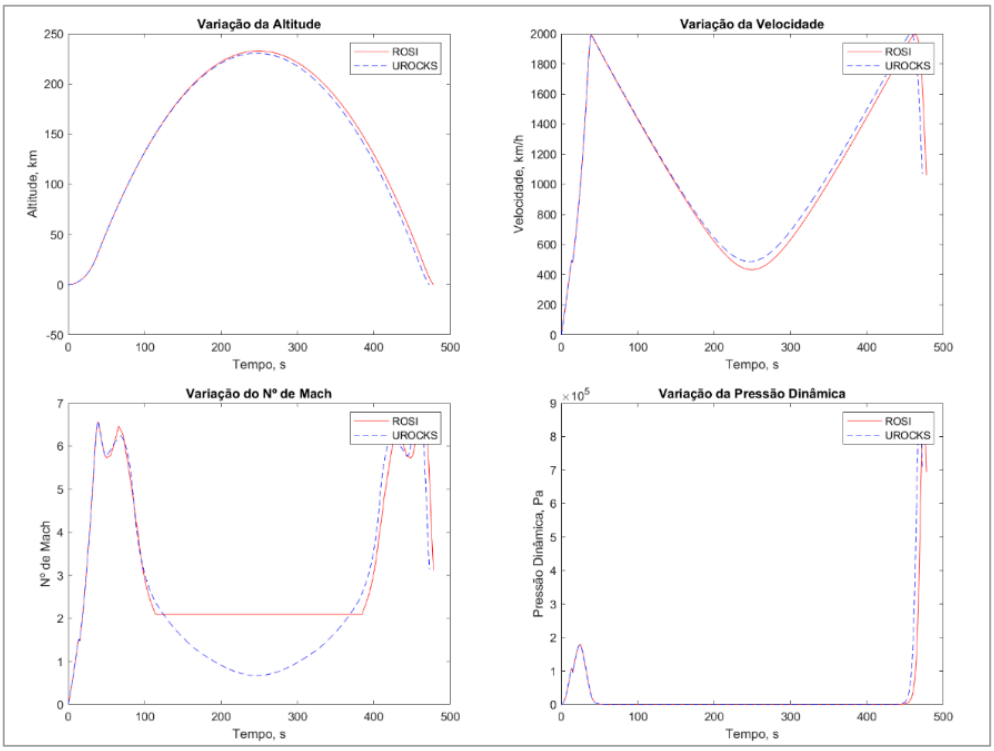

Figura 19 - Comparação dos parâmetros de saída dos simuladores UROCKS e ROSI

Foi criada uma rotina que recebe os dados de vento provenientes da sondagem com balões meteorológicos e os transforma em entradas, que contém informações de vento para o simulador. Essas entradas constituem uma tabela de ventos de quatro colunas. A primeira é referente à altitude da medição e as outras três são as componentes norte-sul, leste-oeste e vertical do vento medido, nessa ordem. A componente vertical é sempre igual a zero, pois essa medição não é realizada. As duas outras componentes são calculadas conforme as Eq. 4 e 5.

O simulador UROCKS recebe os dados de entrada, seleciona a dinâmica (um, três ou seis graus de liberdade) com base nas fases de voo e executa as equações diferenciais dos modelos utilizados. São eles, os modelos: terrestre, atmosférico, gravitacional, de massa, de momento, propulsivo, de vento e de abertura do sistema ioiô, que cessa o rolamento do foguete em um determinado instante do voo. O modelo terrestre utilizado por este programa considera a terra plana, devido ao alcance relativamente curto que veículos não-guiados costumam atingir. Após a execução dos modelos, as equações são resolvidas de forma numérica com o método de Runge-Kutta de 4ª e 5ª ordens, pela rotina *ode45* do Matlab, para a obtenção dos parâmetros de saída. O fluxograma da Figura 20 mostra as etapas de funcionamento do simulador.

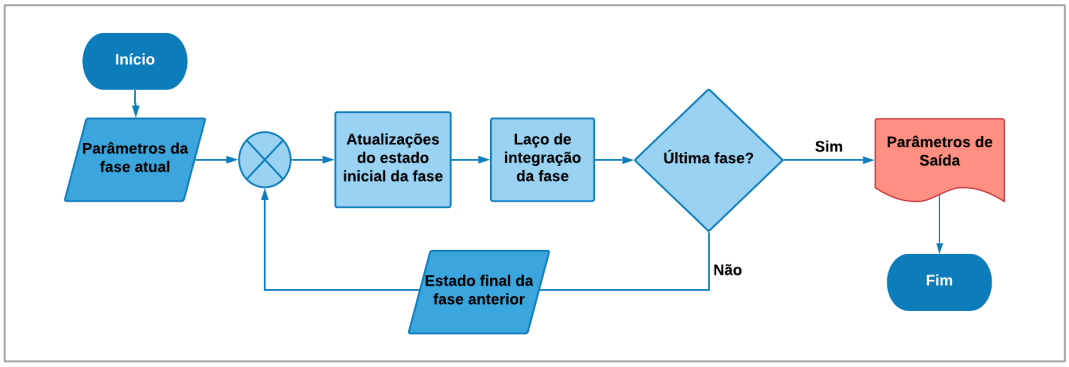

Figura 20 - Fluxograma do simulador UROCKS (DA SILVEIRA, 2014)

No computador empregado neste trabalho, uma simulação que desconsidera a ação do vento com o programa UROCKS tem duração de cerca de 1,0 s. No caso de simulações que utilizam dados de vento, cada execução tem duração de, aproximadamente, 1,3 s.

4.2 Reprodução do Método de Cálculo do *Software* Guará

Nesta seção, será abordado o funcionamento do código utilizado para reproduzir os cálculos do *software* Guará. Além disso, será demonstrado o desempenho do programa por meio de exemplos que simulam situações de lançamentos.

Como apresentada no Capítulo 2, a metodologia de cálculo do programa Guará consiste em três passos: 1 - recepção dos dados de vento, 2 - cálculo dos desvios causados na trajetória nominal e 3 - determinação dos ângulos de azimute e elevação necessários para compensar esse efeito. O cálculo dos desvios é precedido pela pesagem das camadas de vento. Este procedimento não é realizado pelo Guará, todavia, também será reproduzido nesta metodologia. Por isso, o código que contém as equações executadas pelo programa é dividido em dois arquivos com duas rotinas diferentes: uma para a pesagem e a outra para o cálculo dos desvios e dos ângulos que compensam o vento.

A rotina de pesagem das camadas recebe os ângulos de azimute e elevação nominais, o arquivo com os parâmetros do foguete, o vetor contendo os níveis em que a atmosfera foi dividida e os valores das médias de velocidade e de direção do vento do mês considerado. Os arquivos com os perfis de vento coletados no CLA foram agrupados considerando-se os meses em que foram gerados. Sendo assim, a rotina identifica o mês do arquivo e seleciona as médias de velocidade e de direção do vento correspondentes. Estes valores, apresentados na Tabela 1, são utilizados nas simulações executadas durante o procedimento de pesagem. A ferramenta

utilizada para as simulações foi o simulador UROCKS. Prosseguindo para a rotina do cálculo de ajuste, os pesos encontrados são utilizados no cálculo do vento balístico, que, ao multiplicar os desvios de vento unitário lateral e frontal ou de cauda, resulta nos desvios causados pelo perfil de vento completo. Estes desvios são utilizados para obter o ponto de impacto perturbado (PIP) e, em seguida, o ponto de impacto compensado (PIC).

Para demonstrar o funcionamento dos cálculos executados pelo *software* Guará, Figura 21 apresenta exemplos simulados de trajetórias nominais perturbadas. Foram plotadas a trajetória nominal (em vermelho) e a trajetória perturbada, simulada pelo programa UROCKS (em verde). Além disso, foi utilizado o editor de códigos do *software* Matlab para inserir nas figuras geradas pelo simulador o PI perturbado previsto pelo *software* Guará (em azul) e os PIs compensados. As imagens mostram a vista superior das trajetórias projetadas no plano horizontal. O eixo vertical aponta para o norte e o horizontal aponta para o leste.

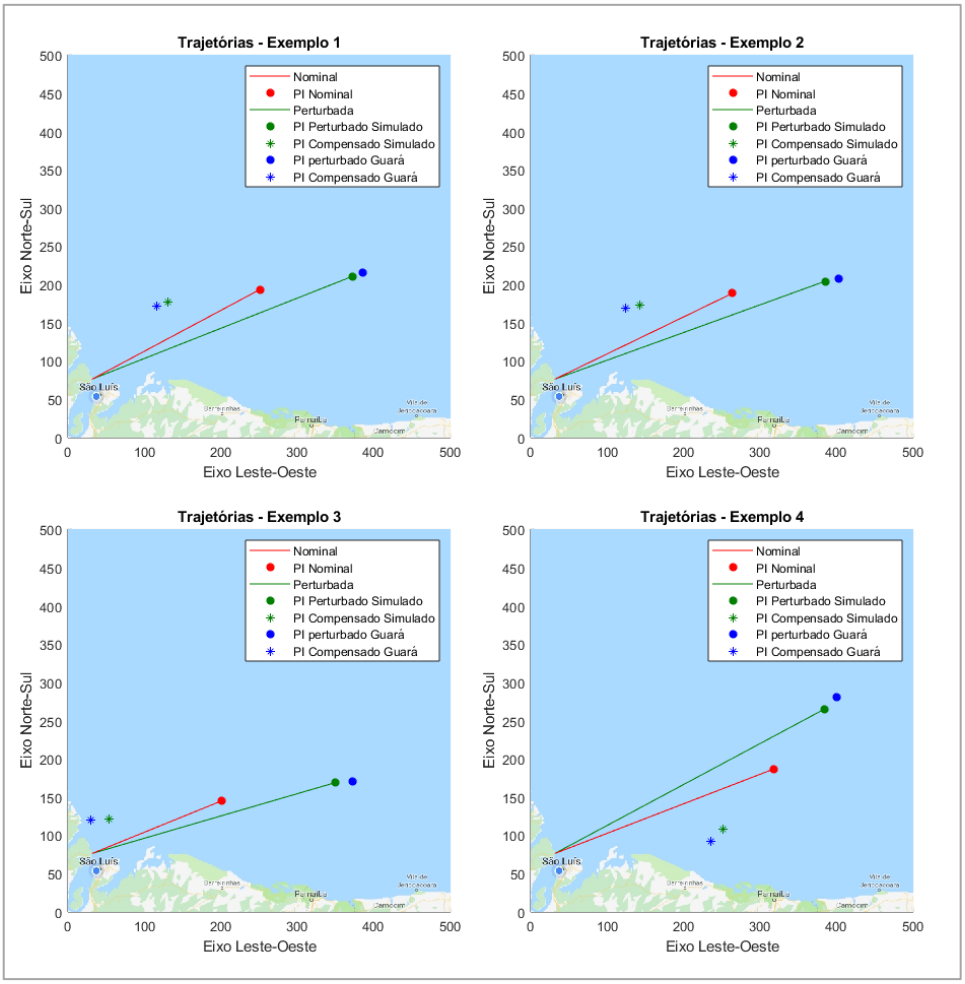

Figura 21 - Exemplos de trajetórias nominais perturbadas

Em cada exemplo, para gerar as trajetórias, foram definidos um perfil de vento e ângulos

de azimute e elevação aleatórios. Estes exemplos foram retirados dos primeiros quatro testes executados e demonstrados na seção 4.4. O foguete utilizado foi o mesmo referido na seção 4.1. Uma observação importante é que, sabendo-se que simulador UROCKS foi a ferramenta utilizada para realizar as simulações necessárias para a pesagem das camadas de vento, seria natural esperar que o PI perturbado calculado pelo programa Guará coincidisse com o simulado. Entretanto, pode-se notar que existe um erro considerável na estimativa realizada pelo programa Guará, como exposto na Tabela 3.

Tabela 3 - Erro de ponto de impacto perturbado previsto

Exemplo	**Distância entre os dois PIs perturbados: simulado via UROCKS e calculado via Guará (km)**
1	14,5
2	18,7
3	23,1
4	22,2

A Tabela 4 descreve a distância entre o ponto de impacto perturbado simulado e o nominal em cada exemplo.

Tabela 4 - Distâncias entre os pontos de impacto perturbado e nominal

Exemplos	**Coordenadas do PI Nominal (km)**		**Coordenadas do PI Perturbado Simulado (km)**		**Distância entre os PIs (km)**
	Norte-Sul	**Leste-Oeste**	**Norte-Sul**	**Leste-Oeste**	
1	117,3	219,8	131,6	354,5	135,45
2	112,4	232,7	123,8	372,5	140,2
3	69,0	170,1	85,6	341,3	172,0
4	110,6	286,9	196,7	369,2	246,6

Finalmente, seguindo os procedimentos descritos na subseção 2.2.3, são calculados os ângulos que levariam o foguete ao PI compensado sem a ação do vento. Pela teoria da compensação antissimétrica, o efeito do vento deveria a anular as distâncias do PIC em relação ao PI nominal (PIN), aproximando o PI resultante do ajuste (PIR) do ponto ideal de impacto. A Figura 22 ilustra as trajetórias compensadas após os cálculos de ajuste nos exemplos demonstrados.

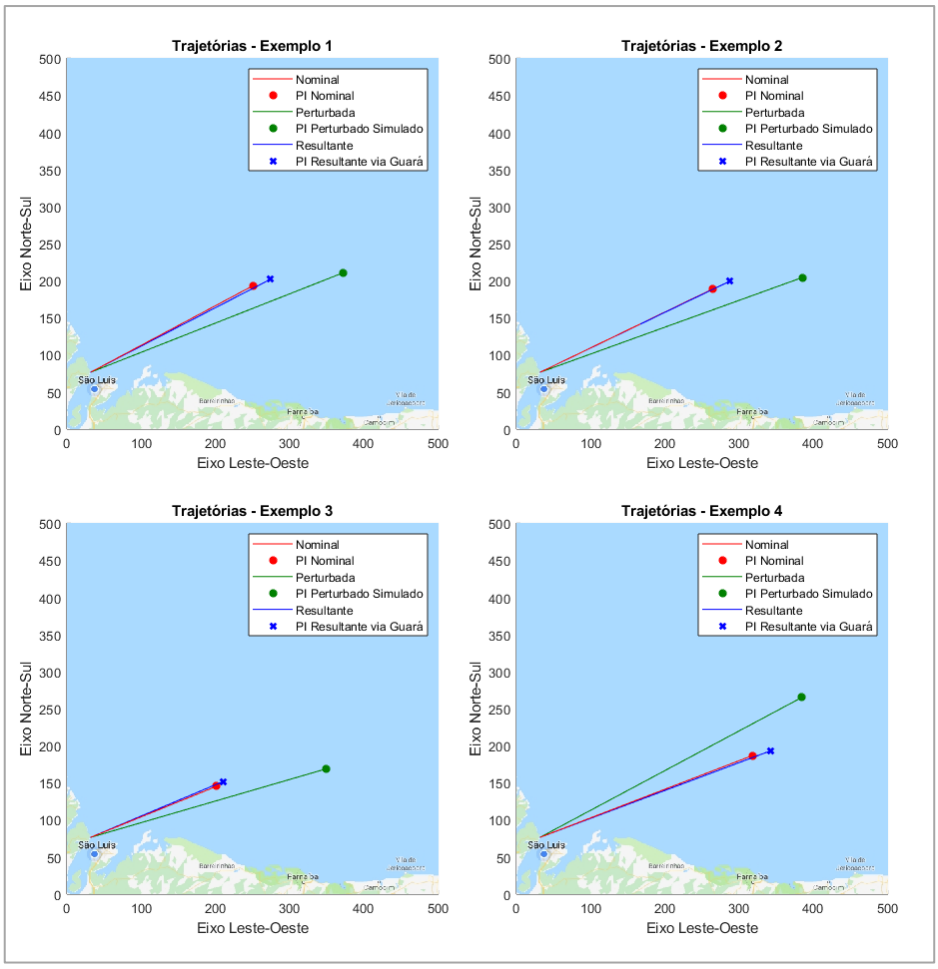

Figura 22 - Exemplos de trajetórias após ajuste com o programa Guará

Pode-se perceber que, como explicado na subseção 2.2.4, o efeito que o vento exerce na trajetória compensada é diferente do efeito produzido na trajetória nominal. Portanto, a compensação antissimétrica realizada apenas uma vez para cada perfil de vento não é suficiente para fazer o PIR coincidir com o nominal, conforme pode ser observado na Tabela 5. A Figura 23 apresenta o fluxograma que resume as rotinas descritas.

Tabela 5 - Ajustes obtidos pelo programa Guará nos exemplos

Exemplos	**Coordenadas do PI Nominal (km)**		**Coordenadas do PI Compensado - Guará (km)**		**Distância entre os PIs (km)**
	Norte-Sul	**Leste-Oeste**	**Norte-Sul**	**Leste-Oeste**	
1	117,3	219,8	125,7	242,9	24,5
2	112,4	232,7	122,9	255,8	25,3
3	69,0	170,1	75,7	179,4	11,4
4	110,6	286,9	117,1	311,5	25,4

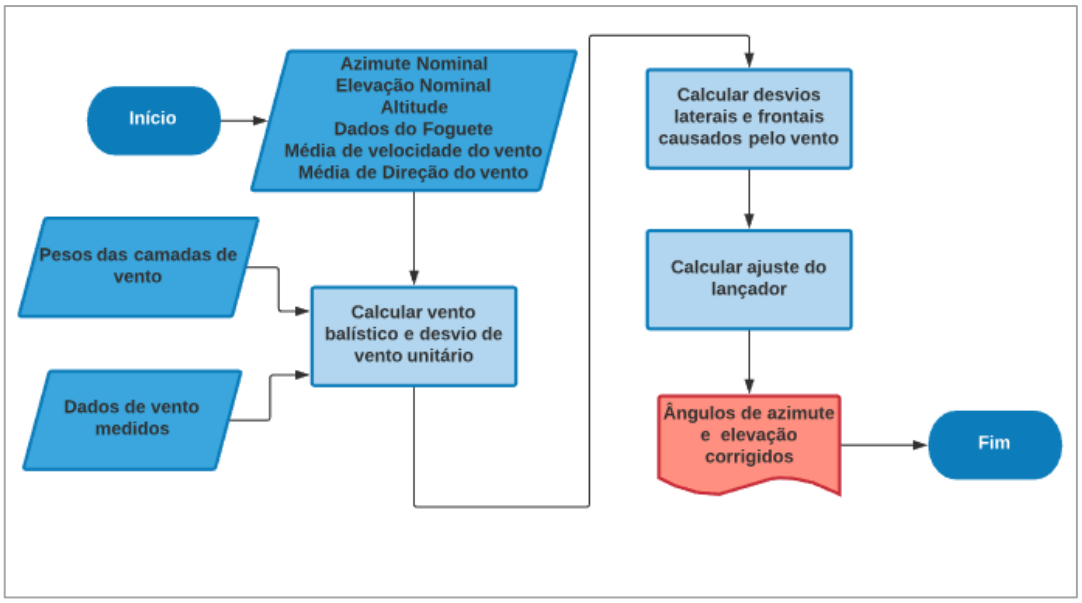

Figura 23 - Fluxograma das etapas necessárias ao ajuste via *software* Guará

4.3 Ajuste do Lançador via Redes Neurais - URLANN

Nesta seção, são descritos o método proposto e o desenvolvimento da rede neural empregada no método de ajuste proposto. Tendo em vista as limitações do programa Guará e a ausência de resultados expressivos na tentativa de redução das suas simplificações pelo Método Mosca, apresentado por Da Mata (2017), esta pesquisa propõe e avalia um novo método de ajuste do lançador, batizado de URLANN (*Unguided Rocket Launcher Adjustment via Neural Networks*). Os passos nos quais este procedimento é dividido buscam, essencialmente, concluir as mesmas etapas: 1 - receber dados do vento medido, 2 - prever o desvio que será causado e 3 - calcular o ajuste do lançador. No entanto, os dois últimos passos deste método são realizados de forma diferente. Para isso, três mudanças são propostas e são explicadas nos tópicos a seguir:

- **Determinação do ponto de impacto perturbado:** Atualmente, o *software* Guará faz o cálculo dos desvios a serem causados pelo vento utilizando os conceitos de pesagem das camadas de vento e vento balístico. Entretanto, como demonstrado nos exemplos da seção 4.2, a previsão dos desvios causados apresenta erros consideráveis em relação aos resultados do próprio simulador utilizado nas pesagens do vento. Por isso, nesta metodologia, a ferramenta utilizada para a determinação do ponto de impacto perturbado foi o simulador UROCKS.
- **Cálculo dos ângulos de azimute e elevação:** O cálculo do ângulo de elevação por meio do *software* Guará considera linear a relação entre a inclinação do lançador e o alcance

atingido. Uma das medidas de melhoria adotadas por Da Mata (2017), apresentadas na subseção 2.2.4, visava fazer uma aproximação não-linear desta relação, em uma tentativa de aumentar a precisão do cálculo. Devido ao relato do autor sobre a ausência de resultados positivos expressivos, nesta metodologia, é proposta uma nova forma de calcular o ajuste dos ângulos do lançador, utilizando os conceitos de redes neurais artificiais. Este procedimento é explicado com detalhes na subseção 4.3.1.

- **Compensação antissimétrica iterativa:** Na metodologia do programa Guará, após a determinação dos desvios causados pelo vento, é realizada a compensação antissimétrica. Porém, este ajuste é realizado apenas uma vez para cada atualização do vento medido, o que resulta em um ponto de impacto com uma distância ainda relativamente alta do PI nominal. No Método Mosca (DA MATA, 2017), uma medida corretiva foi realizar essa compensação de forma iterativa. Como já relatado, as modificações promovidas pelo método Mosca não obtiveram resultados satisfatórios. Entretanto, em uma análise mais profunda, ainda que essas correções tivessem apresentado uma melhoria considerável, seu custo computacional seria elevado. Este fato seria devido às constantes pesagens das camadas de vento e, consequentemente, dezenas de simulações necessárias a cada iteração, para encontrar o ponto de impacto compensado, já que o efeito que as camadas de vento realizam nas trajetórias está em função dos ângulos de azimute e elevação do lançamento. A quantidade de simulações necessárias seria inadequada para as atividades do Setor de Segurança de Voo (SVO) nos instantes que antecedem o lançamento do foguete, pois o Setor de Meteorologia envia dados da torre anemométrica à SVO a cada 20 s. Nesse contexto, no método proposto por este trabalho, foi estabelecido que seria utilizada a compensação antissimétrica de forma iterativa por dois motivos. Primeiramente, porque bastaria uma simulação para que sejam conhecidos os pontos de impacto perturbado ou resultante, o que é conveniente para o cálculo do ajuste de forma iterativa. A outra razão é que um novo método de cálculo dos ângulos do lançador foi proposto, o que poderia gerar resultados diferentes do Método Mosca.

4.3.1 Projeto da Rede Neural

O simulador UROCKS utiliza como entradas as condições iniciais de lançamento e retorna parâmetros de trajetória. Essas condições iniciais, desconsiderando o vento, são o ângulo de azimute α e o ângulo de elevação E do lançador. Os parâmetros de trajetória considerados importantes foram as coordenadas do ponto de impacto NSi e LOi (considerando o sistema de

referência com o lançador na origem e os eixos horizontais apontando para o norte e para o leste), a direção de impacto Azi e a distância do impacto D em relação ao lançador. Estes parâmetros são calculados em função das condições iniciais, como é ilustrado na Figura 24 e descrito pelas Eq. 56 a 59.

Figura 24 - Esquema de funcionamento do simulador

$$NSi = f_1(\alpha, E) \tag{56}$$

$$LOi = f_2(\alpha, E) \tag{57}$$

$$Azi = f_3(\alpha, E) \tag{58}$$

$$D = f_4(\alpha, E) \tag{59}$$

Paralelamente, o ajuste do lançador é realizado em função do ponto de impacto compensado, para o qual o foguete deve ser lançado. Portanto, buscou-se determinar uma forma de processar parâmetros de trajetórias não perturbadas pelo vento, cujo ponto de impacto deseja-se atingir, para obter os ângulos de azimute e de elevação necessários para alcança-lo. Dessa forma, foi projetada uma rede neural que funcionaria de forma invertida em relação ao simulador, como demonstra a Figura 25.

Figura 25 - Esquema de funcionamento da rede neural

Por possuir características de um aproximador universal, a rede neural proposta teve como tarefa aproximar as funções descritas pelas Eq. 60 e 61.

$$\alpha = \mathcal{F}_1(NSi, LOi, Azi, D) \tag{60}$$

$$E = \mathcal{F}_2(NSi, LOi, Azi, D) \tag{61}$$

Para isso, foi estabelecido que a rede deveria realizar um aprendizado supervisionado a partir de dados gerados em simulações de lançamento, semelhante ao estilo de modelagem caixa preta. O foguete utilizado nas simulações foi o mesmo descrito na seção 4.1. Para a criação, configuração e treinamento da rede neural, foi utilizado o *toolbox* de redes neurais do Matlab.

Geração dos dados

Foram executadas simulações com variação do ângulo de azimute entre -20 e $120°$, visto que estes são os limites desse ângulo utilizados em lançamentos no CLA. O ângulo de elevação do lançador foi variado entre os valores de 80 e $86°$, por estes serem próximos aos limites estabelecidos para elevação nominal e corrigida, para diferentes veículos. Os azimutes foram variados com incremento de $0{,}3°$ e os ângulos de elevação, com incremento de $0{,}2°$. No total, foram realizadas 14477 simulações, que geraram exemplos para o treinamento da rede neural. A cada simulação, a combinação de ângulos de lançamento utilizados foi armazenada em uma coluna da matriz M_s, criada para conter os padrões de saída, enquanto os parâmetros das trajetórias geradas foram inseridos nas colunas da matriz M_e, destinada a guardar os padrões de entrada, como mostram as Eq. 62 e 63.

$$M_e = \begin{bmatrix} NSi(1) & NSi(2) & NSi(3) & & NSi(14477) \\ LOi(1) & LOi(2) & LOi(3) & \cdots & LOi(14477) \\ Azi(1) & Azi(2) & Azi(3) & & Azi(14477) \\ D(1) & D(2) & D(3) & & D(14477) \end{bmatrix} \tag{62}$$

$$M_s = \begin{bmatrix} \alpha(1) & \alpha(2) & \alpha(3) & \cdots & \alpha(14477) \\ E(1) & E(2) & E(3) & & E(14477) \end{bmatrix} \tag{63}$$

Topologia da rede

Por meio do *toolbox* utilizado, foi possível configurar características da rede, como número de neurônios ocultos, algoritmo de treinamento e critérios de parada. Foi gerada uma rede neural do tipo *feedforward* com uma única camada oculta de 15 neurônios. Por receber padrões com quatro variáveis e retornar dois valores, trata-se de uma rede de configuração $4 - 15 - 2$. Aos neurônios ocultos foi atribuída a função de ativação tangente hiperbólica e foram utilizados neurônios lineares na camada de saída, devido à rede neural ser destinada a uma aplicação de regressão ou aproximação de funções.

Pré-Treinamento

Os dados utilizados no treinamento foram divididos em: 70% para treinamento, 15%

para validação e 15% para testes realizados durante o processo de aprendizagem. A intenção de separar estes três grupos foi realizar o monitoramento da curva da função custo para cada conjunto e verificar se durante a aprendizagem ocorreriam divergências. Foi estabelecido um pré-processamento na camada de entrada e na camada de saída, por meio da função nativa *mapminmax*, que normaliza as entradas e saídas em valores entre -1 e 1.

Treinamento

O método de treinamento escolhido foi o algoritmo Levenberg-Marquardt, por tratar-se de uma rede com pouco mais de 100 pesos ou conexões e por este método geralmente apresentar uma alta taxa de redução do valor da função custo ao longo das iterações, o que significa que o treinamento demora relativamente pouco para ser concluído. A função custo utilizada foi o erro quadrático médio, abordado na seção 3.1. Ela será referida deste ponto do texto em diante apenas como "erro". Foi estabelecido como critério de parada a satisfação de uma das seguintes condições: redução do erro a 0; conclusão de 5000 iterações (épocas de treinamento) ou a divergência entre as curvas de erro de validação e de treinamento pelo número máximo de 500 iterações seguidas, como visto na Figura 26. Foi utilizada a técnica de *early stopping*, que recupera os pesos obtidos no treinamento que resultou no menor erro da curva de validação.

Figura 26 - Tela de treinamento da rede neural

Pós-Treinamento

O treinamento foi finalizado pelo critério de conclusão de 5000 iterações. O erro de treinamento foi reduzido a $8{,}74 \cdot 10^{-8}$. As três curvas de erro (Figura 27) permaneceram semelhantes ao longo de todas as iterações, confirmando que não houve *overfitting*. O menor erro de validação foi de $9{,}17 \cdot 10^{-7}$, obtido ao final das 5000 épocas.

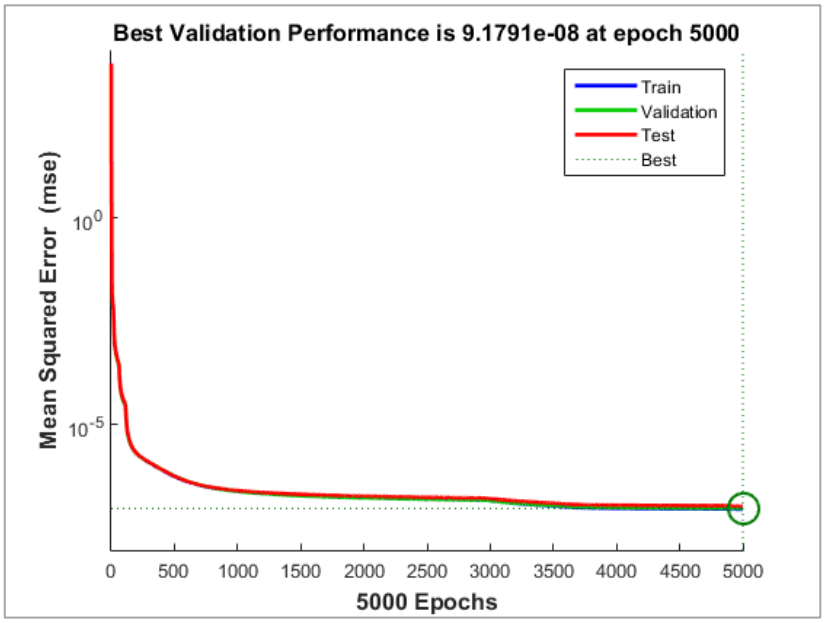

Figura 27 - Curvas de erro de treinamento, validação e teste

Foi verificado, ainda, como mostra a Figura 28, que o coeficiente de correlação *R* para os três conjuntos de dados utilizados aproximou-se de 1, o que significa que o treinamento foi satisfatório, de acordo com os critérios da Tabela 2.

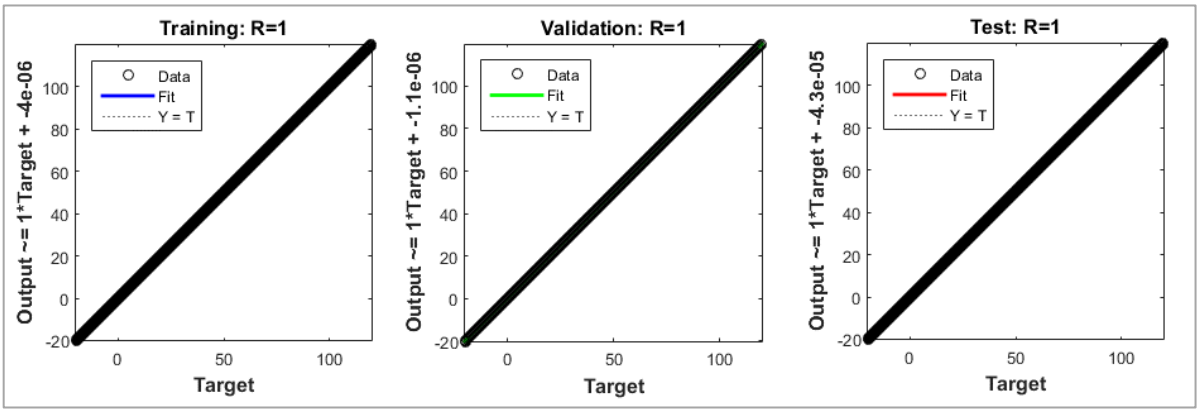

Figura 28 - Coeficientes de correlação obtidos com o treinamento

4.3.2 Compensação Antissimétrica Iterativa

Devido ao ponto de impacto resultante do ajuste geralmente apresentar um desvio considerável em relação ao PI nominal por motivos já descritos, foi decidido utilizar no método URLANN a compensação antissimétrica de forma iterativa. Este método consiste em, a cada iteração, realizar os seguintes passos: 1 - obter o PI resultante (PIR), 2 - calcular os seus desvios em relação ao PI nominal (PIN) e 3 – utilizar esses desvios para corrigir o PI compensado (PIC). Após a correção, o ajuste do lançador deve ser realizado em função do PI compensado corrigido. Então, o processo é repetido, até que um critério de parada seja satisfeito. Os passos descritos são realizados após a primeira correção do lançador, para um mesmo perfil de vento. Com a atualização do vento medido, deve ser realizada uma nova simulação para obter o PI perturbado (PIP) a partir do nominal, dando início à repetição dos passos descritos. A Figura 29 ilustra as constantes correções do PI compensado e a aproximação do PIR até o PIN.

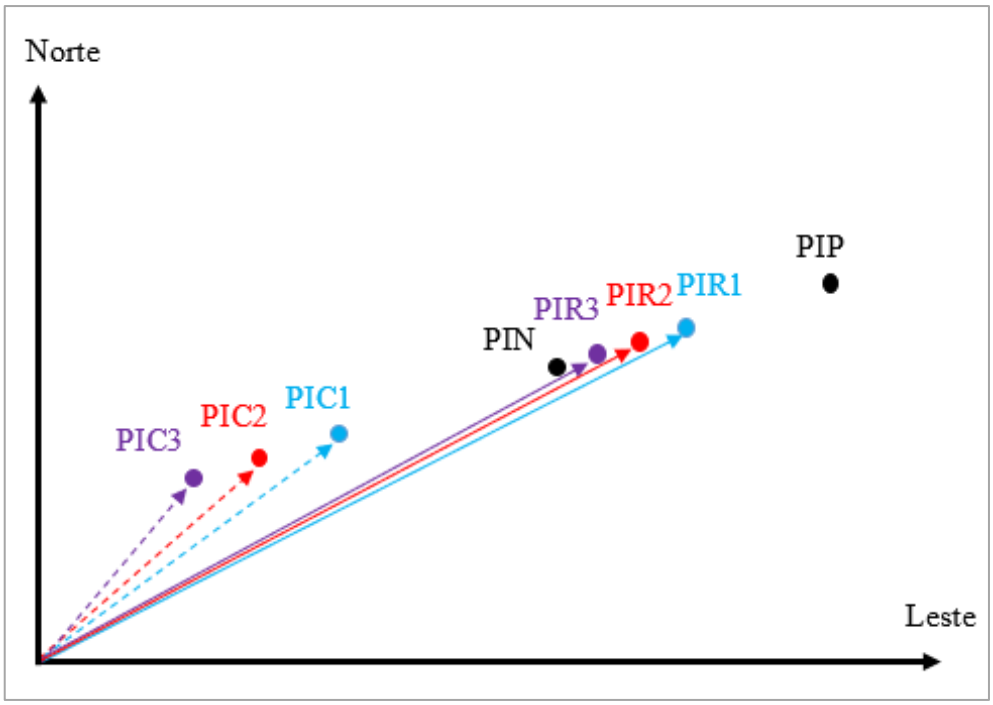

Figura 29 - Correções do PI compensado

O PI compensado PIC1 é calculado com a utilização das diferenças entre as coordenadas do PI perturbado e do PI nominal, que são calculadas conforme as Eq. 64 e 65.

$$dNS_1 = NSp - NSn \tag{64}$$

$$dLO_1 = LOp - LOn \tag{65}$$

em que dNS_1 é a diferença entre as coordenadas norte-sul: NSp (perturbado) e NSn (nominal) e dLO_1 é a diferença entre as coordenadas leste-oeste: LOp (perturbado) e LOn (nominal). Estas distâncias são utilizadas para que sejam obtidas as coordenadas NSc_1 e LOc_1 do PIC1, como mostram as Eq. 66 e 67.

$$NSc_1 = NSn - dNS_1 \tag{66}$$
$$LOc_1 = LOn - dLO_1 \tag{67}$$

De posse das coordenadas do PIC1, o próximo passo é obter o PI resultante PIR1. A atualização do ponto de impacto compensado é realizada a partir das diferenças entre o PIR1 e o nominal PIN, conforme as Eq. 68 e 69.

$$dNS_2 = NSr_1 - NSn \tag{68}$$
$$dLO_2 = LOr_1 - LOn \tag{69}$$

em que que dNS_2 é a diferença entre as coordenadas norte-sul: NSr_1 (resultante 1) e NSn (nominal) e dLO_2 é a diferença entre as coordenadas leste-oeste: LOr_1 (resultante 1) e LOn (nominal). A partir destes valores, podem ser obtidas as coordenadas NSc_2 e LOc_2 do primeiro PI compensado corrigido PIC2, por meio das Eq. 70 e 71. A partir de então, o processo descrito continua, até que seja satisfeito um critério de parada.

$$NSc_2 = NSc_1 - dNS_2 \tag{70}$$
$$LOc_2 = LOc_1 - dLO_2 \tag{71}$$

No método URLANN, os PIs perturbado e resultantes devem ser obtidos por meio de simulações executadas com o simulador UROCKS, após os cálculos do ajuste do lançador, realizados pela rede neural projetada. Foram adotados dois critérios de parada no método proposto. Um deles é a obtenção de um PI resultante que esteja dentro de um determinado raio a partir do PI nominal. O outro é a determinação de um número máximo de iterações. Uma iteração equivale a ajustar o PI compensado e calcular novamente os ângulos do lançador. O fluxograma da Figura 30 ilustra o procedimento.

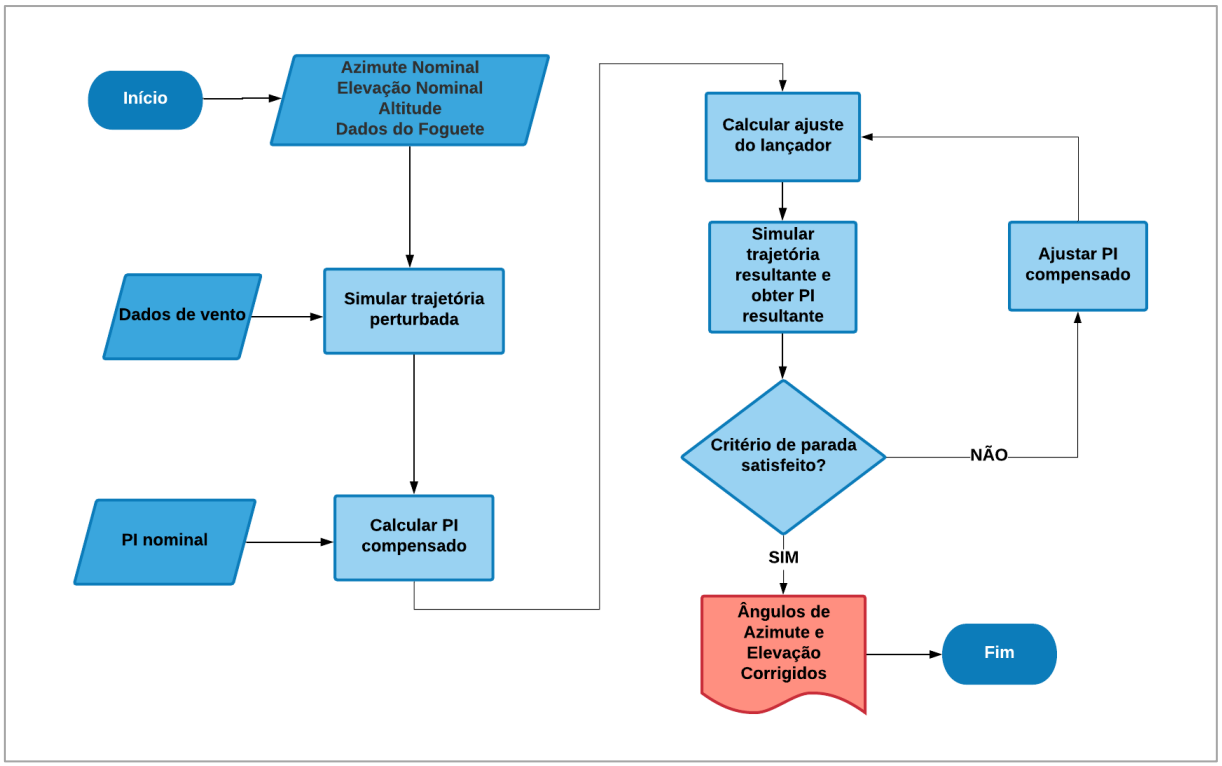

Figura 30 - Fluxograma do método URLANN

4.4 Testes Propostos

A verificação da efetividade do sistema proposto nesse trabalho se deu por meio de testes comparativos. Essa análise foi dividida em duas etapas. A primeira teve como objetivo a avaliação da capacidade da rede neural em calcular ângulos do lançador, necessários para atingir pontos de impacto determinados, sem a ação do vento. A segunda teve como finalidade analisar o desempenho da integração entre a rede neural e o simulador, no ajuste iterativo pelo método URLANN.

À rede neural, foram propostos dez pontos de impacto aleatórios, cujos ângulos necessários para atingi-los deveriam ser calculados. Nesse sentido, foi feita uma comparação com o método que o programa Guará utiliza para esses cálculos. Esses testes não consideraram a ação do vento, pois o objetivo dessa comparação foi analisar a capacidade da rede neural e do programa Guará em calcular os ângulos do lançador que levariam o foguete ao ponto de impacto compensado (PIC), na última etapa do ajuste antissimétrico.

Já para a verificação do funcionamento do método URLANN, foi feita uma comparação do seu desempenho com o programa Guará, na tarefa de compensar a ação do vento. Para isso, foram determinadas 50 condições iniciais aleatórias de lançamento, com a intenção de simular situações reais, nas quais seria necessário ajustar o lançador. Para cada um dos 50 testes

propostos, foi especificada uma combinação de ângulos de azimute e elevação nominais aleatórios. Além disso, foi utilizado em cada teste um perfil de vento diferente, obtido de medidas realizadas no CLA com balões meteorológicos. As medidas são provenientes de arquivos armazenados nos computadores da Seção de Meteorologia. Para o cálculo de ajuste realizado com o programa Guará, foram consideradas 40 camadas de vento para o procedimento de pesagem e obtenção do vento balístico.

Para explorar o funcionamento do método proposto neste trabalho, foram realizadas quatro rodadas dos 50 testes propostos. Nelas, foi estabelecido como um critério de parada do método URLANN a conclusão de uma, duas, três e quatro iterações, respectivamente. O segundo critério foi a obtenção do PI resultante dentro do raio de $100\ m$ a partir do PI nominal. Buscou-se analisar a precisão do PIR obtido após o ajuste e o tempo de execução do método URLANN até a obtenção dos ângulos ajustados do lançador. Portanto, as rodadas de testes também tiveram a finalidade de analisar a eficiência do método proposto em cada caso. O tempo foi contado desde a simulação da trajetória perturbada até o último cálculo de ajuste, ou seja, até a última iteração do método.

5 Resultados

5.1 Desempenho da Rede Neural

Para demonstrar o funcionamento da rede neural projetada para a tarefa de calcular os ângulos do lançador, foram propostos testes em que foram determinados dez pontos de impacto desejados. A rede recebeu, de cada um dos pontos propostos, as coordenadas norte-sul e leste-oeste, a direção de impacto e a distância de impacto a partir da posição de lançamento. Como resultado, ela retornou os ângulos do lançador necessários para atingir os PIs desejados, desconsiderando a ação do vento. Posteriormente, foram realizadas as simulações de lançamento com os ângulos apresentados pela rede e os resultados foram comparados com os pontos de impacto desejados. A Tabela 6 expõe os ângulos do lançador utilizados para obter os pontos de impacto propostos nos testes e as coordenadas destes PIs.

Tabela 6 - Pontos de Impacto propostos para teste da rede neural

Teste	**Ângulos**		**Coordenadas de impacto**	
	$\boldsymbol{\alpha}$ (°)	$\boldsymbol{E}$ (°)	**NS (km)**	**LO (km)**
1	10	81	303,0	52,9
2	25	83,5	211,7	98,3
3	35	80	273,5	190,9
4	40	82,5	202,9	169,8
5	45	84	153,8	153,4
6	50	81	198,1	235,3
7	55	81,5	168,9	240,4
8	60	82	140,1	241,8
9	75	83	64,9	240,7
10	80	84,5	35,1	197,5

Durante os testes, foi realizada uma comparação entre os resultados da rede neural e o método do programa Guará para o cálculo dos ângulos do lançador. Para simular uma situação real, foi utilizado um par de ângulos aleatórios de azimute e elevação, que atuaram como ângulos nominais fictícios. De posse desses ângulos, foram obtidas a trajetória nominal e a distância de impacto, para que fosse possível calcular o TTE (*tower tilt effect*). Os ângulos nominais utilizados foram de $65°$ para azimute e $84°$ para elevação. O TTE calculado foi de $36{,}2045\ km/°$. A Figura 31 e a Tabela 7 mostram os resultados obtidos pelos dois métodos.

Tabela 7 - Resultado do cálculo dos ângulos via Guará e via Rede Neural

Teste	Guará					Rede Neural				
	α (°)	E (°)	NS (km)	LO (km)	Distância (km)	α (°)	E (°)	NS (km)	LO (km)	Distância (km)
1	9,9	81,5	289,3	50,1	14,0	10	80,9	303,0	52,9	0,008
2	24,9	83,5	210,4	97,3	1,7	25	83,4	211,7	98,3	0,004
3	34,9	80,7	257,2	178,9	20,3	35	80	273,5	190,9	0,005
4	39,9	82,6	198,7	165,7	5,8	40	82,5	202,9	169,7	0,007
5	44,9	84	154,1	153,1	0,3	45	84	153,8	153,4	0,005
6	49,9	81,5	189,4	224,4	14,0	49,9	80,9	198,1	235,3	0,003
7	54,9	81,8	162,9	231,1	11,0	54,9	81,5	168,9	240,4	0,003
8	59,9	82,2	136,3	234,4	8,3	59,9	81,9	140,1	241,8	0,001
9	74,9	83,1	64,3	237,1	3,6	75	82,9	64,9	240,7	0,002
10	79,9	84,4	35,7	198,8	1,4	80	84,4	35,1	197,5	0,009

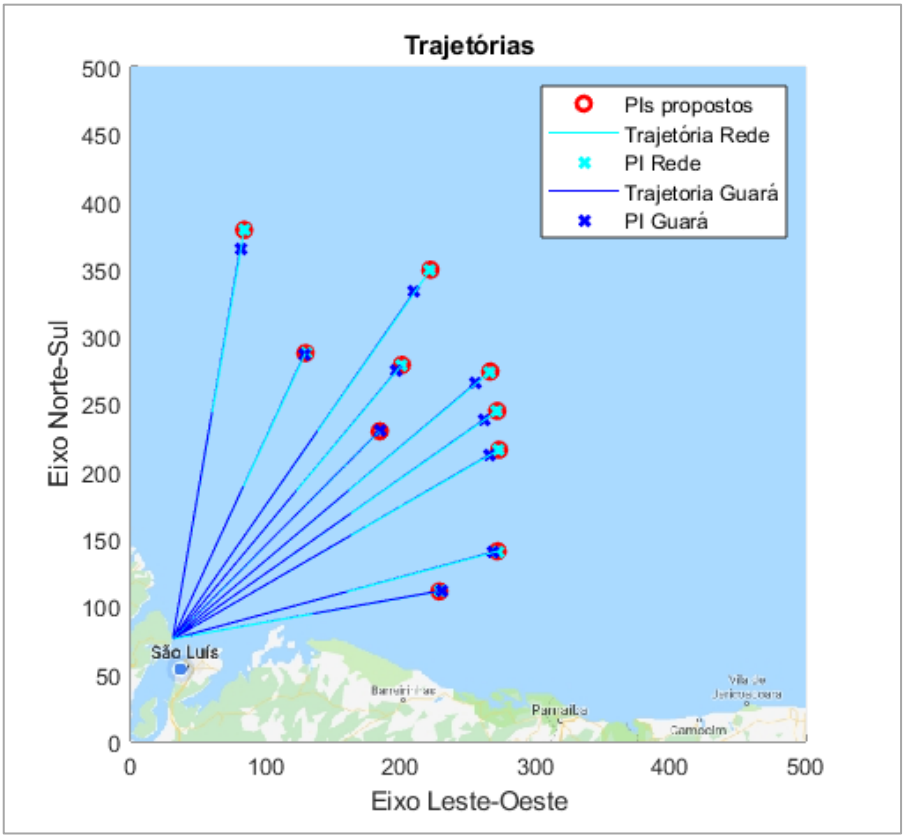

Figura 31 - PIs após cálculo dos ângulos via rede neural e via *software* Guará

Com os resultados dos testes, em uma situação ideal de ausência de vento, ficou claro que a rede neural apresenta vantagem em fornecer ângulos que resultam em um ponto de impacto mais próximo do desejado. Foi notado que os melhores resultados do programa Guará

ocorreram nos casos dos PIs obtidos com elevações próximas à nominal, como nos casos dos testes 2, 5 e 10, por exemplo. Isto demonstra que quanto maior for a diferença entre a distância D_x (do lançador até o PI compensado) e a distância D_n (do lançador até o PI nominal), menor é a capacidade do programa Guará em fornecer o ângulo de elevação adequado para a compensação do vento. Ou seja, para perturbações de vento que gerem PIs perturbados com alcances muito diferentes em relação ao PI nominal, os cálculos do programa Guará apresentarão menor precisão. Em contrapartida, a rede neural não apresentou essa limitação. Para os diferentes casos de elevação, foram obtidos ângulos que levaram o foguete a um impacto com erro quase desprezível. Esse resultado demonstra clara superioridade da rede neural para uso no método de compensação antissimétrica.

5.2 Testes Comparativos entre o *Software* Guará e o Método URLANN

Nesta subseção, são apresentados os resultados dos testes comparativos do método URLAN em relação ao programa Guará. Um dos parâmetros utilizados para comparação foi a média das distâncias, em km, do PI nominal (PIN) até o PI resultante (PIR), após o ajuste com cada método. Outro parâmetro foi a média das reduções da distância entre o PI perturbado (PIP) e o nominal, em porcentagem. Esses parâmetros são apresentados no início da análise de cada uma das rodadas de testes descritas na seção 4.4.

O programa Guará obteve PIRs, em média, 21,3 km distantes do PIN, o que resultou em uma média de 80% de redução da distância do PIN até o PIP. Estes valores foram comparados com os valores obtidos pelo método URLANN em cada uma das rodadas de testes. Os cálculos executados pelo programa Guará foram concluídos de forma instantânea, com duração desprezível. Nessa contagem, não foi considerada a pesagem do vento, que, em situações reais, ocorre com antecedência à operação de lançamento. Portanto, não foram comparados os tempos de execução dos dois métodos.

Para cada rodada de testes, este capítulo apresenta uma tabela comparativa entre os dois ajustes, com detalhes do desempenho de cada um. Para compreendê-la, é necessário estabelecer as seguintes legendas:

- Desvio PIN – PIP (km): desvio, em km, do ponto de impacto, causado pela ação do vento na trajetória nominal;
- Distância PIR (km): distância, em km, entre o PI nominal e o PI resultante do ajuste;
- Redução PIP – PIN (%): redução, em porcentagem, da distância entre o PI nominal e o

PI perturbado, causada pelo ajuste. Ou seja, indica o efeito do ajuste realizado;

- Melhoria (%): representa o quanto a distância entre o PI nominal e o PI resultante, obtida após o ajuste com o método URLANN, foi melhor que a obtida pelo método do *software* Guará.

URLANN (1 iteração)

Após o ajuste realizado com o método URLANN, executado com uma iteração, foram obtidos os seguintes parâmetros comparativos:

- Média das distâncias entre o PIN e o PIR: 4,9 km;
- Média das reduções da distância entre o PIN e o PIP: 95,96%.

A Tabela 8 apresenta os resultados dos testes comparativos nesta rodada. Para entender os dados apresentados nas tabelas desta seção, é realizada a seguir uma breve explicação, utilizando o Teste 1 como exemplo. A segunda coluna da Tabela 8 mostra os desvios causados pelo vento na trajetória nominal. No Teste 1, esse desvio foi de 121,919 km. Nas duas colunas seguintes, são apresentados os resultados obtidos com o método do programa Guará. A coluna esquerda mostra os efeitos dos ajustes do lançador, com as distâncias entre o PI nominal e o PI resultante. No primeiro teste, a distância obtida foi de 24,523 km. A coluna à direita mostra a redução provocada por esse ajuste, que, nesse caso, foi de 79,89 %.

As duas colunas seguintes da Tabela 8 apresentam os resultados obtidos pelo método URLANN. A coluna esquerda indica que, no Teste 1, a distância entre o PI nominal e o PI resultante do ajuste foi de 5,790 km. A coluna à direita mostra que esse resultado correspondeu a uma redução de 95,25 % do desvio que o vento causaria. A última coluna dessa tabela apresenta a melhoria resultante da aplicação do método URLANN, em relação aos resultados do programa Guará. No primeiro teste, o resultado do ajuste via URLANN (5,790 km) foi 76,39 % melhor que o resultado do programa Guará (24,523 km). Esta análise pode ser aplicada para todas as linhas dessa tabela e para todas as tabelas apresentadas nesta seção.

Tabela 8 - Testes comparativos: Guará - URLANN com uma iteração

Teste	Desvio PIN – PIP (km)	Guará		URLANN		
		Distância PIR (km)	Redução PIP – PIN (%)	Distância PIR (km)	Redução PIP – PIN (%)	Melhoria (%)
1	121,919	24,523	79,89	5,790	95,25	76,39
2	122,182	25,350	79,25	7,071	94,21	72,11
3	150,011	11,501	92,33	6,089	95,94	47,06
4	103,039	25,396	75,35	8,239	92,00	67,56

5	98,375	30,719	68,77	9,981	89,85	67,51
6	98,562	7,385	92,51	0,646	99,34	91,25
7	124,076	25,854	79,16	6,093	95,09	76,43
8	116,946	14,798	87,35	1,272	98,91	91,40
9	100,461	21,150	78,95	4,255	95,76	79,88
10	138,867	21,164	84,76	4,840	96,51	77,13
11	97,982	21,496	78,06	4,844	95,06	77,47
12	80,625	20,841	74,15	3,063	96,20	85,30
13	99,334	22,456	77,39	3,408	96,57	84,82
14	154,147	101,783	33,97	7,570	95,09	92,56
15	139,058	18,083	87,00	2,256	98,38	87,52
16	141,779	19,313	86,38	0,754	99,47	96,10
17	139,126	23,115	83,39	7,113	94,89	69,23
18	85,151	19,132	77,53	2,637	96,90	86,22
19	144,376	15,865	89,01	6,037	95,82	61,94
20	134,895	20,643	84,70	3,919	97,09	81,01
21	145,364	16,329	88,77	4,652	96,80	71,51
22	114,168	26,883	76,45	9,843	91,38	63,39
23	138,813	12,075	91,30	3,274	97,64	72,89
24	108,547	28,580	73,67	8,529	92,14	70,16
25	96,488	8,439	91,25	0,563	99,42	93,33
26	112,523	29,799	73,52	8,869	92,12	70,24
27	87,001	24,934	71,34	6,008	93,09	75,90
28	92,491	23,564	74,52	5,160	94,42	78,10
29	134,056	22,736	83,04	4,948	96,31	78,24
30	110,558	17,598	84,08	3,258	97,05	81,49
31	82,437	19,977	75,77	2,887	96,50	85,55
32	113,836	17,275	84,82	1,447	98,73	91,62
33	129,707	22,949	82,31	4,928	96,20	78,53
34	121,275	25,085	79,32	5,014	95,87	80,01
35	104,170	11,096	89,35	0,797	99,24	92,82
36	122,284	28,510	76,69	10,183	91,67	64,28
37	97,333	15,014	84,57	0,821	99,16	94,53
38	136,349	19,193	85,92	3,561	97,39	81,44
39	115,919	27,668	76,13	8,205	92,92	70,34
40	126,721	22,864	81,96	4,992	96,06	78,17
41	93,919	32,857	65,02	11,399	87,86	65,31
42	101,906	30,449	70,12	11,752	88,47	61,41
43	122,223	18,801	84,62	5,138	95,80	72,67
44	140,604	17,256	87,73	3,486	97,52	79,80
45	95,863	9,272	90,33	0,434	99,55	95,32
46	99,202	33,328	66,40	10,911	89,00	67,26
47	107,694	18,541	82,78	3,167	97,06	82,92
48	101,471	20,221	80,07	4,075	95,98	79,85
49	89,852	33,991	62,17	12,984	85,55	61,80
50	112,423	27,428	75,60	7,462	93,36	72,80

A Figura 32 ilustra as diferenças de desempenho entre os dois métodos, em cada teste. O parâmetro comparativo utilizado foi a distância do PIN até o PIR.

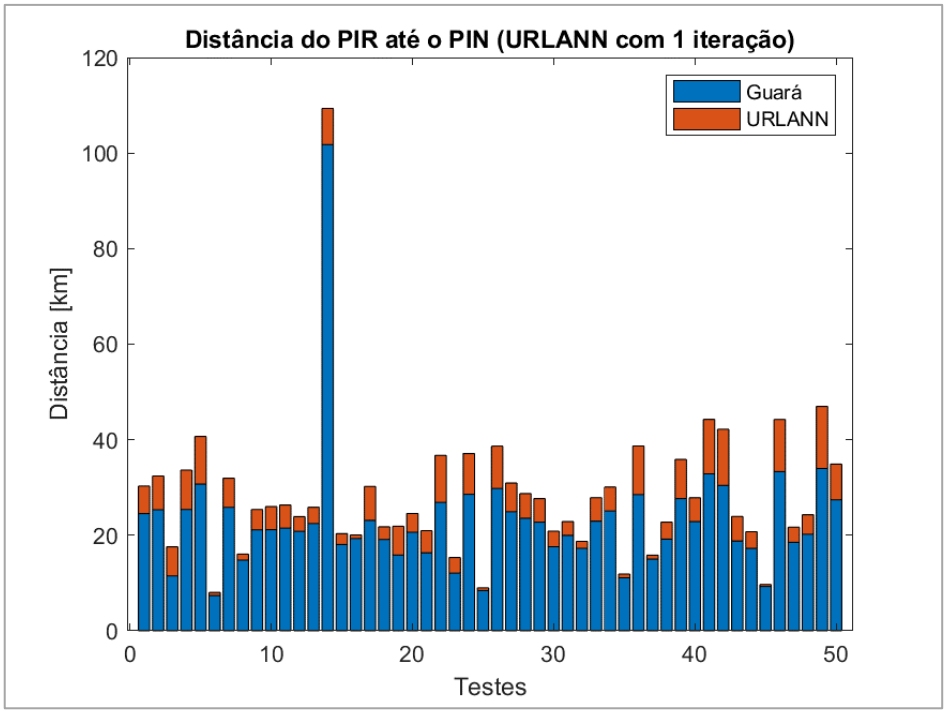

Figura 32 - Comparação Guará - URLANN com uma iteração

A distância entre o PIN e o PIR após o ajuste com o método URLANN foi, em média, 78,13% menor que a distância obtida após o ajuste com o programa Guará. A execução do programa teve duração máxima de 3,8 s. A primeira iteração, na maioria dos casos, foi responsável por reduzir a distância até o PI resultante para um valor abaixo de 10 km, que corresponde a um valor 53% menor que a média das distâncias resultantes do ajuste fornecido pelo programa Guará.

URLANN (2 iterações)

Nesta rodada, os parâmetros comparativos obtidos pelo método URLANN foram os seguintes:

- Média das distâncias entre o PIN e o PIR: 0,7 km;
- Média das reduções da distância entre o PIP e o PIN: 99,4 %.

Neste caso, o PIR obtido pelo método URLANN foi, em média, 96,74% mais próximo do PIN que o PIR obtido pelo ajuste com os cálculos do programa Guará. O tempo máximo de

execução foi de 5,0 s. A Tabela 9 apresenta os resultados dos testes comparativos nesta rodada.

Tabela 9 - Testes comparativos: Guará - URLANN com duas iterações

Teste	Desvio PIN – PIP (km)	Guará		URLANN		
		Distância PIR (km)	Redução PIP – PIN (%)	Distância PIR (km)	Redução PIP – PIN (%)	Melhoria (%)
1	121,919	24,523	79,89	0,904	99,26	96,31
2	122,182	25,350	79,25	1,233	98,99	95,13
3	150,011	11,501	92,33	1,642	98,91	85,72
4	103,039	25,396	75,35	1,918	98,14	92,45
5	98,375	30,719	68,77	2,640	97,32	91,41
6	98,562	7,385	92,51	0,067	99,93	99,09
7	124,076	25,854	79,16	0,762	99,39	97,05
8	116,946	14,798	87,35	0,116	99,90	99,21
9	100,461	21,150	78,95	0,717	99,29	96,61
10	138,867	21,164	84,76	0,828	99,40	96,09
11	97,982	21,496	78,06	0,943	99,04	95,62
12	80,625	20,841	74,15	0,496	99,39	97,62
13	99,334	22,456	77,39	0,527	99,47	97,66
14	154,147	101,783	33,97	1,967	98,72	98,07
15	139,058	18,083	87,00	0,491	99,65	97,28
16	141,779	19,313	86,38	0,132	99,91	99,32
17	139,126	23,115	83,39	1,193	99,14	94,84
18	85,151	19,132	77,53	0,405	99,52	97,88
19	144,376	15,865	89,01	1,773	98,77	88,82
20	134,895	20,643	84,70	0,579	99,57	97,20
21	145,364	16,329	88,77	0,881	99,39	94,60
22	114,168	26,883	76,45	2,276	98,01	91,53
23	138,813	12,075	91,30	0,467	99,66	96,14
24	108,547	28,580	73,67	1,937	98,22	93,22
25	96,488	8,439	91,25	0,050	99,95	99,41
26	112,523	29,799	73,52	1,841	98,36	93,82
27	87,001	24,934	71,34	1,310	98,49	94,75
28	92,491	23,564	74,52	1,042	98,87	95,58
29	134,056	22,736	83,04	0,688	99,49	96,97
30	110,558	17,598	84,08	0,474	99,57	97,31
31	82,437	19,977	75,77	0,476	99,42	97,62
32	113,836	17,275	84,82	0,134	99,88	99,23
33	129,707	22,949	82,31	0,622	99,52	97,29
34	121,275	25,085	79,32	0,545	99,55	97,83
35	104,170	11,096	89,35	0,072	99,93	99,35
36	122,284	28,510	76,69	2,310	98,11	91,90
37	97,333	15,014	84,57	0,040	99,96	99,74
38	136,349	19,193	85,92	0,514	99,62	97,32
39	115,919	27,668	76,13	1,642	98,58	94,07

40	126,721	22,864	81,96	0,715	99,44	96,87
41	93,919	32,857	65,02	3,198	96,60	90,27
42	101,906	30,449	70,12	3,257	96,80	89,30
43	122,223	18,801	84,62	0,766	99,37	95,92
44	140,604	17,256	87,73	0,455	99,68	97,36
45	95,863	9,272	90,33	0,034	99,96	99,63
46	99,202	33,328	66,40	2,914	97,06	91,26
47	107,694	18,541	82,78	0,367	99,66	98,02
48	101,471	20,221	80,07	0,631	99,38	96,88
49	89,852	33,991	62,17	3,950	95,60	88,38
50	112,423	27,428	75,60	1,544	98,63	94,37

A Figura 33 ilustra as diferenças de desempenho entre os dois métodos, em cada teste.

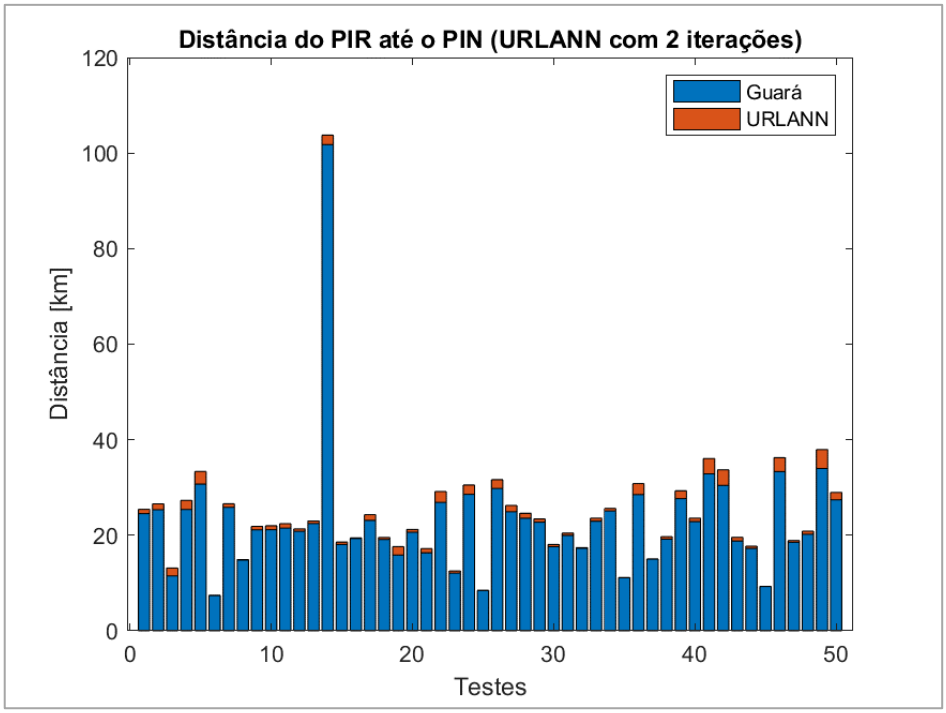

Figura 33 - Comparação Guará - URLANN com duas iterações

URLANN (3 iterações)

Na terceira rodada de testes, os parâmetros comparativos obtidos pelo método URLANN foram:

- média das distâncias entre o PIN e o PIR: 0,1 km
- média das reduções das distâncias entre o PIP e o PIN: 99,9%

Como observado, a distância média do PI nominal até o resultante obtido pelo método URLANN foi equivalente ao valor do raio utilizado como segundo critério de parada. Este fato

explica a ocorrência de cinco casos em que a execução do programa foi finalizada na segunda iteração. Em média, a distância entre o PIN e o PIR foi 99,4% menor que a obtida pelo código que executa as equações do Guará. O tempo de execução teve duração máxima de 7,1 s. A Tabela 10 apresenta os resultados dos testes comparativos nesta rodada.

Tabela 10 - Testes comparativos: Guará - URLANN com três iterações

Teste	Distância PIN – PIP (km)	Guará		URLANN		
		Distância PIR (km)	Redução PIP – PIN (%)	Distância PIR (km)	Redução PIP – PIN (%)	Melhoria (%)
1	121,919	24,523	79,89	0,144	99,88	99,41
2	122,182	25,350	79,25	0,206	99,83	99,19
3	150,011	11,501	92,33	0,324	99,78	97,18
4	103,039	25,396	75,35	0,438	99,57	98,27
5	98,375	30,719	68,77	0,681	99,31	97,78
6	98,562	7,385	92,51	0,067	99,93	99,09
7	124,076	25,854	79,16	0,090	99,93	99,65
8	116,946	14,798	87,35	0,014	99,99	99,90
9	100,461	21,150	78,95	0,119	99,88	99,44
10	138,867	21,164	84,76	0,135	99,90	99,36
11	97,982	21,496	78,06	0,167	99,83	99,22
12	80,625	20,841	74,15	0,082	99,90	99,61
13	99,334	22,456	77,39	0,080	99,92	99,64
14	154,147	101,783	33,97	0,154	99,90	99,85
15	139,058	18,083	87,00	0,170	99,88	99,06
16	141,779	19,313	86,38	0,040	99,97	99,79
17	139,126	23,115	83,39	0,202	99,85	99,13
18	85,151	19,132	77,53	0,061	99,93	99,68
19	144,376	15,865	89,01	0,259	99,82	98,37
20	134,895	20,643	84,70	0,083	99,94	99,60
21	145,364	16,329	88,77	0,138	99,90	99,15
22	114,168	26,883	76,45	0,511	99,55	98,10
23	138,813	12,075	91,30	0,054	99,96	99,55
24	108,547	28,580	73,67	0,421	99,61	98,53
25	96,488	8,439	91,25	0,050	99,95	99,41
26	112,523	29,799	73,52	0,356	99,68	98,81
27	87,001	24,934	71,34	0,280	99,68	98,88
28	92,491	23,564	74,52	0,209	99,77	99,11
29	134,056	22,736	83,04	0,103	99,92	99,55
30	110,558	17,598	84,08	0,074	99,93	99,58
31	82,437	19,977	75,77	0,073	99,91	99,64
32	113,836	17,275	84,82	0,012	99,99	99,93
33	129,707	22,949	82,31	0,084	99,94	99,64
34	121,275	25,085	79,32	0,054	99,96	99,79
35	104,170	11,096	89,35	0,072	99,93	99,35

36	122,284	28,510	76,69	0,513	99,58	98,20
37	97,333	15,014	84,57	0,040	99,96	99,74
38	136,349	19,193	85,92	0,067	99,95	99,65
39	115,919	27,668	76,13	0,326	99,72	98,82
40	126,721	22,864	81,96	0,098	99,92	99,57
41	93,919	32,857	65,02	0,886	99,06	97,30
42	101,906	30,449	70,12	0,882	99,13	97,10
43	122,223	18,801	84,62	0,125	99,90	99,34
44	140,604	17,256	87,73	0,055	99,96	99,68
45	95,863	9,272	90,33	0,034	99,96	99,63
46	99,202	33,328	66,40	0,749	99,24	97,75
47	107,694	18,541	82,78	0,035	99,97	99,81
48	101,471	20,221	80,07	0,092	99,91	99,54
49	89,852	33,991	62,17	1,163	98,71	96,58
50	112,423	27,428	75,60	0,310	99,72	98,87

A Figura 34 ilustra as diferenças de desempenho entre os dois métodos, em cada teste. É notável que a distância obtida após ajuste via URLANN praticamente não é percebida no gráfico, devido à aproximação do PIR até o PIN.

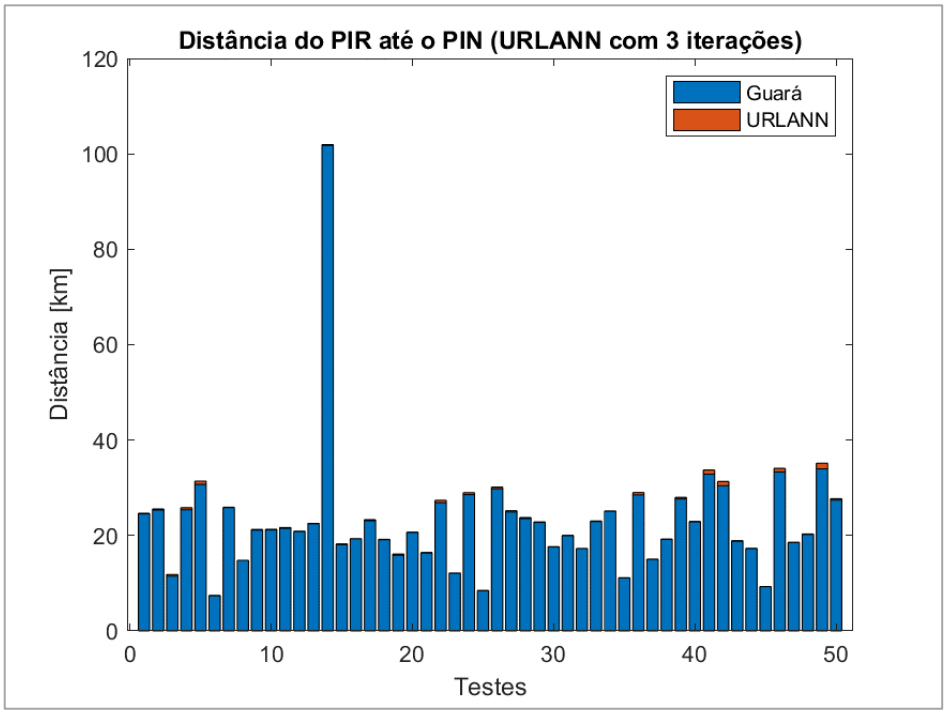

Figura 34 - Comparação Guará - URLANN com três iterações

URLANN (4 iterações)

A seguir, são apresentados os parâmetros comparativos obtidos pelo método URLANN com a execução de 4 iterações.

- média das distâncias entre o PIN e o PIR: 0,1 km
- média das reduções da distância entre o PIP e o PIN: 99,9%

Os resultados nesta rodada mostraram-se praticamente iguais à rodada anterior. Do total de 50 testes, 23 foram finalizados antes da conclusão das quatro iterações. Foi obtida uma distância entre o PIN e o PIR, em média, 99,7% menor que a resultante com o programa Guará. O URLANN apresentou o tempo máximo de execução de 7,4 s. A Tabela 11 apresenta os resultados dos testes comparativos nesta rodada.

Tabela 11 - Testes comparativos: Guará - URLANN com quatro iterações

Teste	Distância PIN – PIP (km)	Guará		URLANN		
		Distância PIR (km)	Redução PIP – PIN (%)	Distância PIR (km)	Redução PIP – PIN (%)	Melhoria (%)
1	121,919	24,523	79,89	0,022	99,98	99,91
2	122,182	25,350	79,25	0,035	99,97	99,86
3	150,011	11,501	92,33	0,028	99,98	99,75
4	103,039	25,396	75,35	0,107	99,90	99,58
5	98,375	30,719	68,77	0,181	99,82	99,41
6	98,562	7,385	92,51	0,067	99,93	99,09
7	124,076	25,854	79,16	0,090	99,93	99,65
8	116,946	14,798	87,35	0,014	99,99	99,90
9	100,461	21,150	78,95	0,018	99,98	99,91
10	138,867	21,164	84,76	0,022	99,98	99,90
11	97,982	21,496	78,06	0,035	99,96	99,84
12	80,625	20,841	74,15	0,082	99,90	99,61
13	99,334	22,456	77,39	0,080	99,92	99,64
14	154,147	101,783	33,97	0,037	99,98	99,96
15	139,058	18,083	87,00	0,085	99,94	99,53
16	141,779	19,313	86,38	0,040	99,97	99,79
17	139,126	23,115	83,39	0,031	99,98	99,87
18	85,151	19,132	77,53	0,061	99,93	99,68
19	144,376	15,865	89,01	0,013	99,99	99,92
20	134,895	20,643	84,70	0,083	99,94	99,60
21	145,364	16,329	88,77	0,020	99,99	99,88
22	114,168	26,883	76,45	0,118	99,90	99,56
23	138,813	12,075	91,30	0,054	99,96	99,55
24	108,547	28,580	73,67	0,095	99,91	99,67
25	96,488	8,439	91,25	0,050	99,95	99,41
26	112,523	29,799	73,52	0,073	99,93	99,75
27	87,001	24,934	71,34	0,064	99,93	99,74
28	92,491	23,564	74,52	0,039	99,96	99,83
29	134,056	22,736	83,04	0,015	99,99	99,93
30	110,558	17,598	84,08	0,074	99,93	99,58
31	82,437	19,977	75,77	0,073	99,91	99,64

32	113,836	17,275	84,82	0,012	99,99	99,93
33	129,707	22,949	82,31	0,084	99,94	99,64
34	121,275	25,085	79,32	0,054	99,96	99,79
35	104,170	11,096	89,35	0,072	99,93	99,35
36	122,284	28,510	76,69	0,118	99,90	99,58
37	97,333	15,014	84,57	0,040	99,96	99,74
38	136,349	19,193	85,92	0,067	99,95	99,65
39	115,919	27,668	76,13	0,060	99,95	99,78
40	126,721	22,864	81,96	0,098	99,92	99,57
41	93,919	32,857	65,02	0,239	99,75	99,27
42	101,906	30,449	70,12	0,235	99,77	99,23
43	122,223	18,801	84,62	0,029	99,98	99,84
44	140,604	17,256	87,73	0,055	99,96	99,68
45	95,863	9,272	90,33	0,034	99,96	99,63
46	99,202	33,328	66,40	0,197	99,80	99,41
47	107,694	18,541	82,78	0,035	99,97	99,81
48	101,471	20,221	80,07	0,092	99,91	99,54
49	89,852	33,991	62,17	0,347	99,61	98,98
50	112,423	27,428	75,60	0,058	99,95	99,79

A Figura 35 ilustra as diferenças de desempenho entre os dois métodos, em cada teste. Neste caso, as distâncias obtidas após o ajuste com o método URLANN são imperceptíveis, pois, na maioria dos testes, a distância obtida ficou abaixo de 100 m.

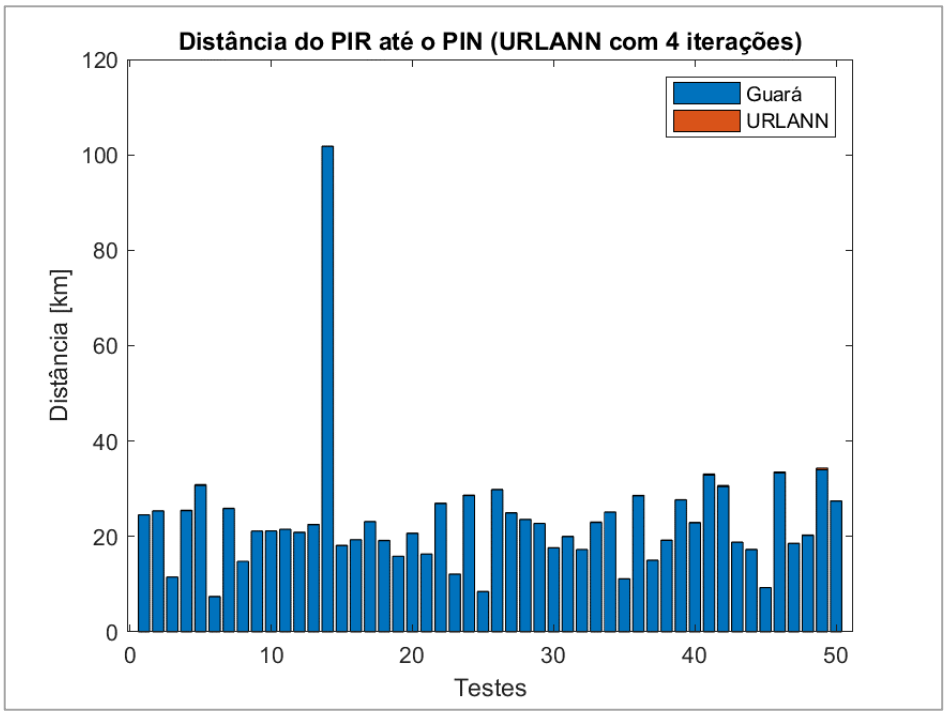

Figura 35 - Comparação Guará - URLANN com quatro iterações

6 Conclusão

Neste trabalho, foi demonstrado que os cálculos executados pelo programa Guará apresentam imprecisões e erros consideráveis. Nesse contexto, foi desenvolvido o sistema batizado como “URLANN”, que calcula o ajuste do lançador e que promove três grandes diferenças: 1 – o uso de um simulador com dinâmica de seis graus de liberdade para previsão do ponto de impacto perturbado; 2 – a utilização de uma rede neural para o cálculo dos ângulos de lançamento e 3 – a realização do ajuste de forma iterativa. A aplicação de testes comparativos mostrou que o método proposto resultou em melhorias consideráveis em relação ao programa Guará.

O fator determinante para o êxito do método URLANN foi o atingimento dos objetivos específicos. Primeiramente, a rede neural criada teve a aprendizagem concluída com erro próximo de zero, da ordem de 10^{-8}. Após o treinamento, foi observado que o mapeamento realizado entre os ângulos do lançador e os parâmetros de trajetória resultou em uma forma mais precisa para o cálculo desses ângulos. Esse fato é resultante do sucesso da rede em realizar uma aproximação efetiva da relação inversa entre as entradas e as saídas do simulador. Posteriormente, a integração entre rede neural e o simulador UROCKS, no método de compensação antissimétrica iterativa, foi capaz de fornecer o ajuste do lançador de forma eficaz. Esse método reduziu drasticamente a distância do PI nominal até o PI resultante, em comparação ao programa Guará. Portanto, foi demonstrado que o método proposto é superior para o ajuste do lançador.

Em relação ao *software* Guará, os pesos das camadas de vento, calculados para os ângulos nominais de lançamento, só permitem prever os desvios efetuados na trajetória nominal. Não é possível calcular os desvios precisos em trajetórias corrigidas utilizando os mesmos pesos das camadas. Essa operação exigiria que uma nova pesagem fosse realizada. Além disso, a pesagem exige que dezenas de simulações sejam executadas, o que impossibilita que a compensação antissimétrica seja realizada de forma iterativa pelo método do programa Guará, momentos antes do lançamento. Por isso, o ajuste é realizado apenas uma vez, para cada atualização do perfil de vento. Como demonstrado em testes, esse cálculo é pouco preciso, pois o efeito do vento na trajetória compensada é diferente, culminando em resultados que contradizem o princípio da compensação antissimétrica. Além deste fato, o ponto de impacto perturbado previsto pelo programa Guará não coincide com aquele da trajetória perturbada simulada, obtida

pelo mesmo simulador empregado na pesagem das camadas vento. Foi observado que o erro entre esses dois pontos chega a dezenas de quilômetros. Este fato contribuiu para a decisão de empregar o simulador para a previsão do ponto de impacto perturbado, no método URLANN.

Outra constatação de imprecisão em relação ao método do programa Guará foi em relação ao cálculo dos ângulos do lançador para corrigir a trajetória. A relação entre a variação da distância de impacto e a variação em elevação (*tower tilt effect*) pode ser considerada constante apenas em torno do ponto de impacto nominal. Isso significa que, para atingir um PI com alcance relativamente parecido ao alcance do PI nominal, desconsiderando a ação do vento, a utilização do TTE constante não implica grandes imprecisões. No entanto, para pontos de impacto associados a grandes variações de elevação, ou seja, cuja distância até o lançador é muito diferente do alcance do PI nominal, a utilização do *TTE* não é satisfatória, pois foi observado que a relação entre o alcance e a elevação é não-linear.

Por outro lado, a rede neural criada apresentou resultados superiores ao fazer a aproximação dessa relação. Em testes propostos, foi demonstrado maior versatilidade nesse cálculo, em comparação com as equações do programa Guará. Ou seja, não foram apresentadas as mesmas restrições descritas no parágrafo anterior. Sendo assim, a rede demonstrou maior capacidade de fornecer os ângulos necessários para atingir pontos de impacto desejados.

O método URLANN tem tempo de execução variável. A sua duração muda conforme o número de iterações determinadas pelo usuário e as configurações do computador que o executa. Nas rodadas de testes propostos, foi observado que, no computador de configurações descritas na seção 4.1, a duração dos cálculos com quatro iterações chegou ao valor máximo de 7,4 s, estando dentro da janela de atualização do perfil de vento no Setor de Segurança de Voo do CLA. Por outro lado, a execução de uma única iteração, ou seja, uma repetição do cálculo de ajuste do lançador, demonstrou uma melhoria considerável na redução da distância entre o PI resultante e o PI nominal. Tal fato indica que, em computadores cujas configurações só permitiriam realizar a primeira iteração dentro do intervalo de atualização do perfil de vento, o método já poderia apresentar resultados superiores àqueles do programa Guará. Contudo, ainda são necessários testes práticos para esta constatação.

Trabalhos futuros

Para trabalhos futuros, recomendam-se os tópicos a seguir.

- Comparações entre o método URLANN e o MBV – método baseado em voos de Da Mata (2017) em um estudo de caso;
- Teste do método URLANN com foguetes diferentes;
- Verificação de resultados do método URLANN para ajuste do lançador em

lançamentos reais;

- Uso de outros simuladores para integrar o método, por exemplo, simuladores que adotam o modelo de terra esferoidal.

Referências

AEB – Agência Espacial Brasileira. Disponível em: < www.aeb.gov.br/imagens/#!gallery-31-22 >. Acesso em 08 ago. 2018

AEB – Agência Espacial Brasileira. **PNAE – Programa Nacional de Atividades Espaciais**. 2012. Disponível em < http://www.aeb.gov.br/wp-content/uploads/2012/11/pnae_web.pdf >. Acesso em: 16 ago. 2018

AEB, Agência Espacial Brasileira. **Regulamento Técnico da Segurança para Lançamento e Voo**. 2007. Disponível em: < http://www.aeb.gov.br/wp-content/uploads/2018/04/Parte_3_Regulamento_Tecnico_da_Seguranca_para_Lancamento_e_Voo.pdf >. Acesso em: 08 mar. 2018.

BISHOP, C. M. **Pattern Recognition and Machine Learning**. Cambridge: Springer, 2006.

BRAGA, A. P., CARVALHO, A. P. L., e LUDEMIR, T. B. (2007). **Redes neurais artificiais: teoria e aplicações**. LTC, Livros Técnicos e Científicos.

CARVALHO, André et al. **Inteligência Artificial–uma abordagem de aprendizado de máquina.** Rio de Janeiro: LTC, 2011.

CLA – Centro de Lançamento de Alcântara. **MAN-SGO-001**: Manual de Segurança Operacional do CLA. Alcântara, 2017.

CYBENKO, G. **Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals, and Systems**, v. 2, n. 4, p. 303–314, dez. 1989.

DA MATA, Henrique Oliveira. **Modelo de Cálculo de Parâmetros de Segurança de Veículos Suborbitais Baseado em Voos do Centro de Lançamento de Alcântara**. 2017. 110f. Dissertação de Mestrado – Instituto Tecnológico de Aeronáutica, São José dos Campos

DA MATA, H. O.; FALCÃO, J. B. P. **COMPENSAÇÃO DO EFEITO DO VENTO EM TRAJETÓRIAS VEÍCULOS SUBORBITAIS**. IX Congresso Nacional de Engenharia Mecânica. Anais...Fortaleza: Agência Brasileira de Engenharia e Ciências Mecânicas, 2016

DA SILVEIRA, Guilherme**. Desenvolvimento de Uma Ferramenta Computacional para Simulação de Voo de Veículos Lançadores**. 2014. 120 p. Dissertação de Mestrado (Curso de Pós-Graduação em Engenharia e Tecnologia Espaciais/Mecânica Espacial e Controle) - Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2014.

DAUMÉ III, Hal. A course in machine learning. **Publisher, ciml. info**, p. 5-73, 2012.

DE ALMEIDA NETO, Areolino. **Redes Neurais – Aspectos Avançados: Deep Learning**. 10 jul. 2018. 19 slides. Material apresentado para a disciplina de Redes Neurais no curso de Mestrado em Ciência da Computação da UFMA.

DE ALMEIDA NETO, A. **Aplicações de Múltiplas Redes Neurais em Sistemas**

Mecatrônicos. 2003. 154 p. Tese de Doutorado (Curso de Engenharia Aeronáutica e Mecânica) - Instituto Tecnológico de Aeronáutica, São José dos Campos, 2003.

FISCH, G. Características do perfil vertical do vento no Centro de Lançamento de Foguetes de Alcântara (CLA). **Revista Brasileira de Meteorologia**, v. 14, n. 1, p. 11-21, 1999.

GARCIA, A. **Automatização Aplicada a Lançadores de Foguetes de Sondagem para compensação da Influência dos Ventos**. [s.l.] Universidade Estadual Paulista, Faculdade de Engenharia de Guaratinguetá, 2007.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. Cambridge, MA, USA: The Mit Press, 2016. 800 p.

HAGAN, M. T.; DEMUTH, H. B.; BEALE, M. H. **Neural Network Design**. 2nd. ed.

HAYKIN, Simon. **Redes Neurais**: Princípios e Prática. 2ª. ed. Porto Alegre: Bookman, 2001. 900 p.

HINTON, Geoffrey E.; SALAKHUTDINOV, Ruslan R. Reducing the dimensionality of data with neural networks. **science**, v. 313, n. 5786, p. 504-507, 2006.

JAMES, J. R. L.; HARRIS, R. J. NASA Technical Note D-645, **Calculation of Wind Compensation for Launching of Unguided Rockets**. Langley Field, 1961.

KRAMER, H.J. **Wind Weighting Analysis for Unguided Rockets**. (Alemanha) Techisches Memorandum Nr. 119. DLR, 1973

KRAMER, H.J., CRAUBNER, A., ZIEGLTRUM, W. (Alemanha), **Rocket Simulation – ROSI: Analysis and Specification of Program ROSI**, DFVLR TN 12/76, 1976.

LIMA, Isaías; PINHEIRO, Carlos AM; SANTOS, Flávia A. Oliveira. **Inteligência artificial**. Elsevier Brasil, 2016.

LINDO, N. C.; VIANA, M. A. M. **Análise das Restrições de Lançamento no CLA**. In: VII FÓRUM DE PESQUISA E INOVAÇÃO DO CLBI, Paranamirim, 2017

LUCCA, E. V. D. **The Brazilian Sounding Rocket VSB-30: meeting the Brazilian Space Program and COPUOS objectives**. Departamento de Ciência e Tecnologia Aeroespacial, 2014. Disponível em: < http://www.unoosa.org/pdf/pres/stsc2014/tech-44E.pdf >.

MILLER, M. F. A scaled conjugate gradient algorithm for fast supervised learning. **Neural Networks**, v. 6, n. 4, p. 525–533, 1 jan. 1993.

PALMERIO, A. F. **Introdução à Tecnologia de Foguetes**. São José dos Campos: SindCT, 2017.

PEREIRA, Eduardo Iorio. **Levantamento dos Perfis Médios de Ar Superior para Composição do Atlas Climatológico do Centro de Lançamento de Alcântara.** São José dos Campos, SP: Relatório Técnico Aca/RT-01/01 GDO-000000/B0047.

Ministério da Aeronáutica. Departamento de Pesquisas e Desenvolvimento, 2002. 42p.

YAMANAKA, S. S. C.; GOMES, R. M. Technical Report 024/ASE-V/01, **Launch Pad Setting Calculation – GUARÁ**. São José dos Campos – SP: Instituto de Aeronáutica e Espaço, 2001