Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Engenharia Aeroespacial

Estimação de parâmetros aerodinâmicos do caça Mirage III utilizando o filtro de Kalman estendido

Autor: Mateus Marocolo Alves de Freitas
Orientador: Prof. Dr. William Reis Silva

Brasília, DF
2024

Mateus Marocolo Alves de Freitas

Estimação de parâmetros aerodinâmicos do caça Mirage III utilizando o filtro de Kalman estendido

Monografia submetida ao curso de graduação em Engenharia Aeroespacial da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia Aeroespacial.

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Prof. Dr. William Reis Silva

Brasília, DF

2024

Mateus Marocolo Alves de Freitas
Estimação de parâmetros aerodinâmicos do caça Mirage III utilizando o filtro de Kalman estendido/ Mateus Marocolo Alves de Freitas. – Brasília, DF, 2024-
165 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. William Reis Silva

Trabalho de Conclusão de Curso – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2024.

1. Estimação de parâmetros. 2. Filtro de Kalman estendido. I. Prof. Dr. William Reis Silva. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Estimação de parâmetros aerodinâmicos do caça Mirage III utilizando o filtro de Kalman estendido

CDU

Mateus Marocolo Alves de Freitas

Estimação de parâmetros aerodinâmicos do caça Mirage III utilizando o filtro de Kalman estendido

Monografia submetida ao curso de graduação em Engenharia Aeroespacial da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia Aeroespacial.

Trabalho aprovado. Brasília, DF, 20 de setembro de 2024:

Prof. Dr. William Reis Silva
Orientador

Prof. Dra. Polliana Cândida Oliveira Martins
Convidado 1

Prof. Dr. Mateus Rodrigues Miranda
Convidado 2

Brasília, DF
2024

Resumo

A estimação de parâmetros e o filtro de Kalman estendido desempenham papéis essenciais na análise e controle de aeronaves. Em conjunto, essas ferramentas têm a capacidade de identificar aspectos cruciais do comportamento dinâmico de uma aeronave, permitindo a melhor compreensão de seu desempenho em voo. Este trabalho aborda a estimação de trinta e quatro estados (doze parâmetros de movimento e vinte e dois parâmetros aerodinâmicos) do caça Mirage III utilizando o filtro de Kalman estendido em condição de voo reto e nivelado e curva coordenada. A implementação foi feita em MATLAB® e Simulink® que visa comparar as estimativas dos estados com valores reais obtidos de um modelo dinâmico não linear e dados medidos por sensores. A metodologia envolve a análise dos erros absolutos das estimativas e a verificação da estabilização e convergência do filtro com base em intervalos de confiança. Os resultados mostraram que o FKE foi eficaz na estimação das posições lineares e angulares, com erro dentro dos limites de confiança estabelecidos. No entanto, o filtro apresentou dificuldades na estabilização das velocidades lineares e em alguns parâmetros aerodinâmicos de força e de momento. Apesar dessas limitações, o FKE demonstrou um desempenho satisfatório na maioria dos parâmetros, comparando a estimativa com o valor real, validando assim, a técnica para aplicações práticas.

Palavras-chaves: Estimação de parâmetros. Filtro de Kalman estendido. Dinâmica de voo. Coeficientes aerodinâmicos.

Abstract

Parameter estimation and the extended Kalman filter play essential roles in aircraft analysis and control. Together, these tools have the ability to identify crucial aspects of an aircraft's dynamic behavior, allowing a better understanding of its in-flight performance. This work addresses the estimation of thirty-four states (twelve motion parameters and twenty-two aerodynamic parameters) of the Mirage III fighter using the extended Kalman filter in straight and level flight and coordinated turn conditions. The implementation was done in MATLAB® and Simulink® which aims to compare the state estimates with real values obtained from a nonlinear dynamic model and data measured by sensors. The methodology involves the analysis of the absolute errors of the estimates and the verification of the stabilization and convergence of the filter based on confidence intervals. The results showed that the FKE was effective in estimating linear and angular positions, with error within the established confidence limits. However, the filter presented difficulties in stabilizing linear velocities and in some aerodynamic parameters of force and moment. Despite these limitations, FKE demonstrated satisfactory performance in most parameters, comparing the estimate with the real value, thus validating the technique for practical applications.

Key-words: Parameter estimation. Extended Kalman filter. Flight dynamics. Aerodynamic coefficients.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

FK	Filtro de Kalman
ARC	*Ames Research Center*
NASA	*National Aeronautics and Space Administration*
FKE	Filtro de Kalman estendido
CG	Centro de gravidade
VA	Variável aleatória
GPS	*Global Position System*

Lista de símbolos

F_B	Sistema de referência do corpo da aeronave com direção x, y e z
F_E	Sistema de referência da Terra com direção x_E, y_E e z_E
C	Origem de F_B
O	Origem de F_E
X	Força resultante na direção x de F_B
Y	Força resultante na direção y de F_B
Z	Força resultante na direção z de F_B
$\bar{L}$	Momento resultante de rolagem
M	Momento resultante de arfagem
N	Momento resultante de quinada
u	Velocidade linear na direção x de F_B
v	Velocidade linear na direção y de F_B
w	Velocidade linear na direção z de F_B
p	Velocidade de rolagem
q	Velocidade de arfagem
r	Velocidade de guinada
$\mathbf{f}$	Vetor de forças externas resultantes
m	Massa total da aeronave
$\mathbf{V}$	Vetor velocidade linear do CG
$\mathbf{G}$	Vetor de momentos externas resultantes em relação ao CG
$\mathbf{h}$	Vetor momento angular
ϕ	Ângulo de rolagem
θ	Ângulo de arfagem

ψ	Ângulo de guinada
$\boldsymbol{\omega}$	Vetor velocidade angular
$\mathbf{L}_{EB}$	Matriz de rotação de F_E para F_B
$\mathbf{T}_{EB}$	Matriz que representa a relação entre as velocidades angulares em F_E e F_B
$\mathbf{g}$	Vetor aceleração da gravidade.
g	Módulo da aceleração da gravidade
$\mathbf{I}_B$	Matriz de inércia da aeronave composto pelos momentos de inércia I_x, I_y e I_z e pelos produtos de inércia I_{xy}, I_{xz}, I_{yx}, I_{yz}, I_{zx}, I_{zy},
F_A	Sistema de referência aerodinâmico com direção x_a, y_a e z_a
α	Ângulo de ataque
β	Ângulo de derrapagem
γ	Ângulo de subida
V	Módulo do vetor velocidade $\mathbf{V}_B$
D	Força aerodinâmica de arrasto
F_Y	Força aerodinâmica lateral
L	Força aerodinâmica de sustentação
T	Tração do motor
ρ	Densidade do ar
S	Área da asa
C_D	Coeficiente de força de arrasto
C_Y	Coeficiente de força lateral
C_L	Coeficiente de força de sustentação
δ_r	Deflexão angular do leme
δ_a	Deflexão angular dos ailerons
δ_e	Deflexão angular do profundor
κ	Constante da função parabólica de $C_L \times C_D$

$\bar{c}$	Corda média aerodinâmica da asa
C_{D_0}	Coeficiente de força de arrasto para $C_L = 0$
C_{Y_β}	Derivada parcial aerodinâmica adimensional de força lateral devido à derrapagem
$C_{Y_{\delta_r}}$	Derivada parcial aerodinâmica adimensional de força lateral devido à deflexão angular do leme
$C_{Y_{\delta_a}}$	Derivada parcial aerodinâmica adimensional de força lateral devido à deflexão angular dos ailerons
C_{L_0}	Coeficiente de força de sustentação para $\alpha = 0$
C_{L_α}	Derivada parcial aerodinâmica adimensional de força de sustentação devido ao ângulo de ataque
$C_{L_{\delta_e}}$	Derivada parcial aerodinâmica adimensional de força de sustentação devido à deflexão angular do profundor
C_{L_q}	Derivada parcial aerodinâmica adimensional de força sustentação devido à velocidade de arfagem
b	Envergadura da asa
C_l	Coeficiente de momento de rolagem
C_m	Coeficiente de momento de arfagem
C_n	Coeficiente de momento de guinada
C_{l_β}	Derivada parcial aerodinâmica adimensional de momento de rolamento devido à derrapagem
$C_{l_{\delta_r}}$	Derivada parcial aerodinâmica adimensional de momento de rolamento devido à deflexão angular do leme
$C_{l_{\delta_a}}$	Derivada parcial aerodinâmica adimensional de momento de rolamento devido à deflexão angular dos ailerons
C_{l_p}	Derivada parcial aerodinâmica adimensional de momento de rolamento devido à velocidade de rolagem
C_{l_r}	Derivada parcial aerodinâmica adimensional de momento de rolamento devido à velocidade de guinada
C_{m_0}	Coeficiente de momento de arfagem para $\alpha = 0$

C_{m_α}	Derivada parcial aerodinâmica adimensional de momento de arfagem devido ao ângulo de ataque
$C_{m_{\delta_e}}$	Derivada parcial aerodinâmica adimensional de momento de arfagem devido à deflexão angular do profundor
C_{m_q}	Derivada parcial aerodinâmica adimensional de momento de arfagem devido à velocidade de guinada
C_{n_β}	Derivada parcial aerodinâmica adimensional de momento de guinada devido à derrapagem
$C_{n_{\delta_r}}$	Derivada parcial aerodinâmica adimensional de momento de guinada devido à deflexão angular do leme
$C_{n_{\delta_a}}$	Derivada parcial aerodinâmica adimensional de momento de guinada devido à deflexão angular dos ailerons
C_{n_p}	Derivada parcial aerodinâmica adimensional de momento de guinada devido à velocidade de rolagem
C_{n_r}	Derivada parcial aerodinâmica adimensional de momento de guinada devido à velocidade de guinada
$\mathbf{u}$	Vetor de entrada\controle de tamanho $s \times 1$
$\tilde{\mathbf{y}}$	Vetor de saída\medida do sistema de tamanho $m \times 1$
$\mathbf{x}$	Vetor de estado do sistema de tamanho $n \times 1$
$\boldsymbol{f}(.)$	Função vetorial não linear que representa a dinâmica do sistema de tamanho $n \times 1$
$\boldsymbol{h}(.)$	Função vetorial não linear que representa a saída do sistema de tamanho $m \times 1$
k	Índice de tempo discreto
$\mathbf{A}$	Matriz jacobiana de estado de um sistema genérico de tamanho $n \times n$
$\mathbf{B}$	Matriz jacobiana de entrada de um sistema genérico de tamanho $n \times s$
$\mathbf{C}$	Matriz jacobiana de saída de um sistema genérico de tamanho $m \times n$
$\boldsymbol{X}$	VA discreta e vetorial $\boldsymbol{X}$ de tamanho $N \times 1$
$\boldsymbol{Y}$	VA discreta e vetorial $\boldsymbol{Y}$ de tamanho $M \times 1$

$E[.]$	Valor esperado
$\bar{\boldsymbol{X}}$	Média da VA $\boldsymbol{X}$
$\bar{\boldsymbol{Y}}$	Média da VA $\boldsymbol{Y}$
$p(.)$	Função de densidade de probabilidade
$g(.)$	Função de VA
$C_{\boldsymbol{XY}}$	Covariância entre $\boldsymbol{X}$ e $\boldsymbol{Y}$
$C_{\boldsymbol{X}}$	Autocovariância de $\boldsymbol{X}$
σ_i^2	Variância de X_i
σ_{ij}^2	Covariância de X_i e X_j
σ	Desvio padrão
$\hat{\boldsymbol{\theta}}$	Estimador genérico
$\boldsymbol{\theta}$	Vetor de estimativas genérico de tamanho $K \times 1$
$\boldsymbol{s}$	Função genérica do vetor de amostras genérico
$\tilde{\boldsymbol{x}}$	Vetor de amostras genérico de tamanho $\eta \times 1$
μ	Média amostral
$\boldsymbol{F}$	Matriz de estado de um sistema para o FK
$\boldsymbol{G}$	Matriz de entrada de um sistema para o FK
$\boldsymbol{H}$	Matriz de saída de um sistema para o FK
$\mathbf{w}$	Ruído de modelo
$\mathbf{v}$	Ruído de medição
$\boldsymbol{Q}$	Matriz de variância de $\mathbf{w}$
$\boldsymbol{R}$	Matriz de variância de $\mathbf{v}$
$\hat{\mathbf{x}}^+$	Estimativa a *posteriori*
$\hat{\mathbf{x}}^-$	Estimativa a *priori*
$\boldsymbol{P}^-$	Matriz de covariância do erro da estimativa a *priori*
$\boldsymbol{P}^+$	Matriz de covariância do erro da estimativa a *posteriori*

$\boldsymbol{I}$	Matriz identidade
$\boldsymbol{K}$	Ganho de Kalman
$\boldsymbol{J}$	Função custo
z^{-1}	Atraso de uma amostra
$\boldsymbol{F}^*$	Matriz de estado do modelo linearizado para FKE
$\mathbf{u}^*$	Vetor de entrada\controle do modelo linearizado para FKE
$\mathbf{w}^*$	Ruído de modelo linearizado para FKE
$\boldsymbol{H}^*$	Matriz de saída do modelo linearizado para FKE
$\mathbf{v}^*$	Ruído de medição linearizado para FKE
$\boldsymbol{L}$	Jacobiana do ruído $\mathbf{w}$
$\boldsymbol{M}$	Jacobiana do ruído $\mathbf{v}$
dt	Passo de simulação

Sumário

1 Introdução

A estimação de parâmetros é um campo da estatística que tem como objetivo determinar valores desconhecidos ou não observáveis, chamados de parâmetros, em um modelo estatístico com base em dados observados disponíveis. Sua evolução ao longo da história é marcada por uma busca contínua por métodos mais precisos e eficientes para extrair informações significativas dos dados observados, permitindo tomada de decisões mais assertivas em sistemas complexos de diversas áreas do conhecimento. (MCCABE; A.; S., 2009)

No final do século XVIII e início do século XIX, Gauss deu as primeiras contribuições significativas para o desenvolvimento da teoria da estimação de parâmetros. Ele introduziu o método dos mínimos quadrados, o primeiro método para obter estimativas ótimas a partir de dados ruidosos. Tem o objetivo de estimar parâmetros de um modelo matemático ajustando-o aos dados experimentais de forma a minimizar a soma dos quadrados dos resíduos. (GREWAL; ANDREWS, 2001)

Com o passar dos anos, métodos mais robustos foram desenvolvidos e teorias probabilísticas começaram a ser empregadas. Como fala Eliasson (1993), entre em 1910 e 1920, Ronald Fisher desenvolveu e popularizou o método da máxima verossimilhança, que busca encontrar os valores dos parâmetros que tornam os dados observados mais prováveis dentro de um modelo estatístico.

No final de 1950 Rudolf Emil Kálmán procurava unir noções de variáveis de estado ao problema de filtragem de Wiener e a partir dai surgiu o filtro de Kalman (FK). Conforme Grewal e Andrews (2010), sua primeira aplicação em um caso real foi no programa Apollo no problema de estimação de trajetória e controle para enviar astronautas à lua.

O grupo de pesquisa ARC (*Ames Research Center*) da NASA (*National Aeronautics and Space Administration*) alcançou etapas essenciais no desenvolvimento do FK como um método prático para navegação em tempo real a bordo da missão Apollo. Passos adicionais incluíram o desenvolvimento do que agora é chamado de filtro de Kalman estendido (FKE) e o uso de análise de Monte Carlo para mostrar que as não linearidades do modelo de trajetória não comprometiam a precisão do FKE para as faixas de erros esperadas durante as missões. (GREWAL; ANDREWS, 2010)

Dada a sua atuação bem sucedida, o FK e FKE foram rapidamente disseminados na comunidade aeroespacial e em 1966 foi escolhido para compor o sistema de navegação da aeronave cargueira *Lockheed* C-5A (Fig. 1). O filtro combinava dados inerciais e informações de vários sistemas de navegação para produzir a estimativa do estado da aeronave. (MCGEE; SCHMIDT, 1985)

Figura 1 – *Lockheed* C-5A. (ALLISON, 2018)

O uso dos filtros não se limitou somente na estimativa da posição da aeronave, Brown (1976) estima coeficientes aerodinâmicos usando o FKE. Além disso, ele modelou a dinâmica não linear de um projétil com base em um corpo rígido de 6 graus de liberdade e incorporou medidas de posições linear e angulares.

A estimação de parâmetros aerodinâmicos se tornou uma prática comum na literatura. Chowdhary e Jategaonkar (2010) modelam a dinâmica de voo da aeronave de pesquisa HFB-320 e do veículo aéreo não tripulado ARTIS e utilizam dados de voos reais para implementar o FKE e o filtro de kalman *unscendted*. De forma semelhante, Kokolios (1994) utiliza o FKE a partir de dados experimentais do modelo em escala da aeronave X-31. Nesse trabalho, também são feitos testes em túnel de vento para colocar o FKE a prova em aplicações de tempo real.

Focado em estimar derivadas aerodinâmicas de uma aeronave, Curvo (2000) utiliza o FKE em seu trabalho e chega em conclusões interessantes. O filtro se mostrou adequado para aplicações em sistemas de controle adaptativo oferecendo boas estimativas de parâmetros para um conjunto de leis de controle. Ele conclui que o algoritmo do FKE tem uma capacidade atrativa por poder ser facilmente embarcado no computador de controle de voo, de modo que a estimativa possa ser feita quase em tempo real durante campanhas de teste.

1.1 Objetivo geral

O objetivo geral desse trabalho é estimar parâmetros aerodinâmicos do caça Mirage III através do filtro de Kalman estendido. Tal abordagem visa não apenas a obtenção desses parâmetros, mas também aperfeiçoar a compreensão sobre o comportamento e características aerodinâmicas desta aeronave, beneficiando assim a engenharia aeroespacial e a otimização de sistemas de controle e navegação.

1.2 Objetivos específicos

- Desenvolver um modelo da dinâmico de voo de aeronaves e representá-lo em espaço de estados.
- Estudar os filtros de Kalman e Kalman estendido como métodos de estimação de parâmetros.
- Aplicar o filtro de Kalman estendido na estimativa dos parâmetros aerodinâmicos do caça Mirage III em condição de voo reto e nivelado e curva coordenada.
- Modelar sensores para alimentar as medidas do FKE.

1.3 Justificativa

Nesse contexto, o presente trabalho objetiva aplicar o filtro de Kalman estendido para estimação de parâmetros aerodinâmicos do caça Mirage III. Desse modo, o motivo desse trabalho está aprimorar a precisão do sistema de navegação e controle, além de minimizar erros e ruídos inerentes ao sistema de medição.

1.4 Organização do trabalho

- Capítulo 2: apresenta os fundamentos teóricos de dinâmica de voo e sua representação em espaço de estados.
- Capítulo 3: expõe conceitos fundamentais de probabilidade.
- Capítulo 4: explana métodos de estimação de parâmetros, bem como os filtro de Kalman e Kalman estendido.
- Capítulo 5: mostra a metodologia do trabalho, expondo a solução proposta.
- Capítulo 6: exibe os resultados obtidos a partir das análises do filtro de Kalman estendido.
- Capítulo 7: exibe as conclusões finais e sugestões para trabalhos futuros.

2 Fundamentação teórica

Nesse capítulo serão apresentados os conceitos de dinâmica de voo e probabilidade e métodos de estimação de parâmetros, englobando os filtros de Kalman e Kalman estendido. Esses conceitos serão combinados para analisar a estimação de estados do caça Mirage III, tema central do capítulo de metodologia.

2.1 Dinâmica de voo

Conforme Etkin e Reid (1996), a dinâmica de voo é influenciada por diversos fatores, incluindo efeitos elásticos, dinâmica do piloto, etc. Dito isso, este estudo se restringe à modelagem das áreas de aerodinâmica e mecânica de corpos rígidos. O campo da aerodinâmica é responsável por descrever como o ar flui em torno da aeronave e como forças e momentos são gerados. Já a mecânica de corpos rígidos, foca em retratar como as forças e momentos externos impactam no movimento de objetos que não sofrem deformação.

Para considerar efeitos de movimentos relativos entre uma aeronave e a Terra, são necessários sistemas de referências para representar e analisar como vetores mudam de um sistema para o outro. Para estudar o movimento relativo, são usados o sistema de referência do corpo da aeronave F_B e o sistema de referência da Terra F_E. F_B se move com a aeronave e possui origem C no centro de gravidade (CG) enquanto que F_E é fixo na Terra e possui origem O no centro da Terra, como pode ser visto na Fig. 2 abaixo.

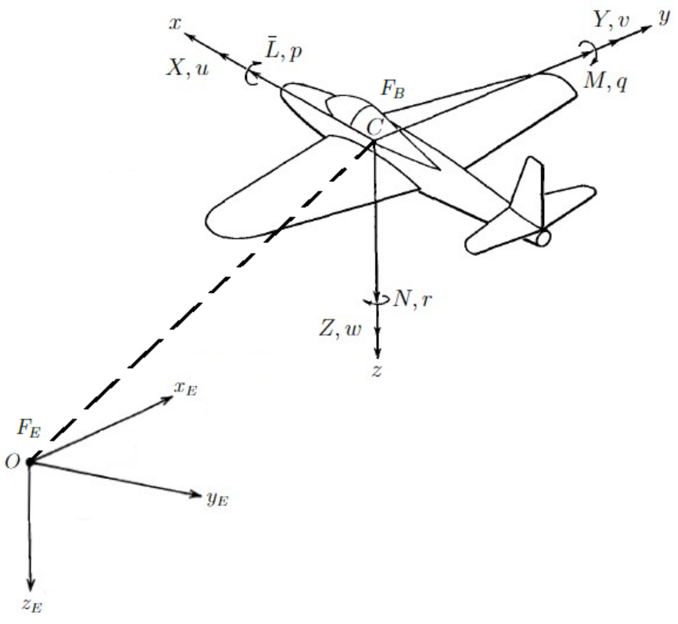


Figura 2 – F_B e F_E. Adaptado de Etkin e Reid (1996).

Onde, tem-se em F_B, as componentes das forças resultantes X, Y e Z nas componentes vetoriais x, y e z respectivamente, as componentes dos momentos resultantes de rolagem $\bar{L}$, de arfagem M e de guinada N, as componentes das velocidades lineares u, v, w nas componentes vetoriais x, y e z e as componentes das velocidades angulares de rolagem p, de arfagem q e guinada r.

Como a Terra é um referencial inercial, as leis de Newton se aplicam e as equações de força e momento podem ser descritas no formato vetorial de acordo com Etkin e Reid (1996), sem considerar a velocidade de vento, pelas Eqs. 2.1 e 2.2.

$$\mathbf{f}_E = \frac{d(m\mathbf{V}_E)}{dt} \tag{2.1}$$

$$\mathbf{G}_E = \dot{\mathbf{h}}_E \tag{2.2}$$

Onde o subscrito E simboliza que o vetor é representado em F_E. Além disso, $\mathbf{f}$ é o vetor de forças externas resultantes agindo sobre a aeronave, m é a massa total da aeronave, $\mathbf{V}$ é o vetor velocidade linear do CG em relação ao ar, $\mathbf{G}$ é o vetor de momentos externas resultantes em relação ao CG e $\mathbf{h}$ é o vetor momento angular.

Para que as Eqs. 2.1 e 2.2 sejam descritas em F_B, uma transformação de sistemas de referência é necessária e os ângulos de Euler são usados para que isso ocorra. Na Fig. 3 a seguir, pode ser visto a definição dos ângulos de Euler e a relação deles com os sistemas de referência da Fig. 2.

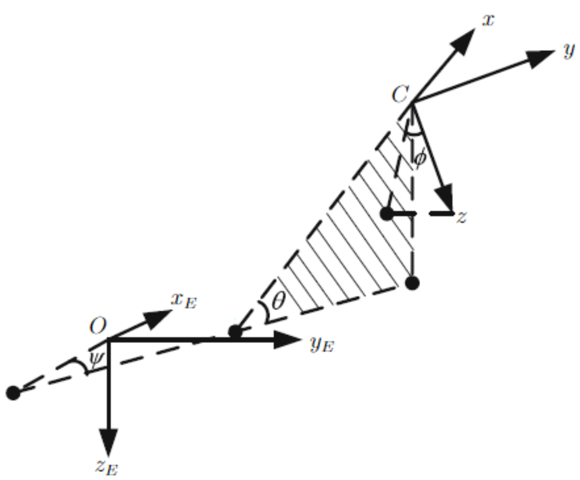


Figura 3 – Ângulos de Euler. Adaptado de Quan (2017).

Os movimentos angulares de uma aeronave são realizados por meio das superfícies de controle e na Fig. 3, para alterar do ângulo de rolagem ϕ, o profundor age, os ailerons modificam o ângulo de arfagem θ, e o leme muda o ângulo de guinada ψ.

Por meio dos ângulos de Euler, é possível construir duas matrizes de que transformam as velocidades linear e angular da aeronave de F_E para F_B. De acordo com a Fig.

2, seja $\mathbf{V}_B = [u\ v\ w]^T$ e $\boldsymbol{\omega}_B = [p\ q\ r]^T$ o vetor velocidade angular da aeronave, a relação entre esses vetores em F_B e F_E é dada pelas Eqs. 2.3 e 2.4 (ETKIN; REID, 1996).

$$\mathbf{V}_E = \mathbf{L}_{EB}\mathbf{V}_B \tag{2.3}$$

$$\boldsymbol{\omega}_E = \mathbf{T}_{EB}\boldsymbol{\omega}_B \tag{2.4}$$

Onde o subscrito B simboliza que o vetor é representado em F_B, $\mathbf{V}_E = [\dot{x}_E\ \dot{y}_E\ \dot{z}_E]^T$ e $\boldsymbol{\omega}_E = [\dot{\phi}\ \dot{\theta}\ \dot{\psi}]^T$. $\mathbf{L}_{EB}$ é a matriz de rotação de F_E para F_B, $\mathbf{T}_{EB}$ é a matriz que representa a relação entre as velocidades angulares em F_E e F_B, dadas pelas Eqs. 2.5 e 2.6 (ETKIN; REID, 1996).

$$\mathbf{L}_{EB} = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \tag{2.5}$$

$$\mathbf{T}_{EB} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \tag{2.6}$$

Com as seguintes limitações:

$$-\pi \leq \phi < \pi \qquad -\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2} \qquad -\pi \leq \psi < \pi$$

Aplicando a matriz de rotação da Eq. 2.5 nas Eq. 2.1 e 2.2 e considerando que não há varião de massa em relação ao tempo, tem-se as equações do movimento representadas em F_B pelas Eqs. 2.7 e 2.8 (ETKIN; REID, 1996).

$$\mathbf{f}_B = m(\dot{\mathbf{V}}_B + \boldsymbol{\omega}_B \times \mathbf{V}_B) \tag{2.7}$$

$$\mathbf{G}_B = \dot{\mathbf{h}}_B + \boldsymbol{\omega}_B \times \mathbf{h}_B \tag{2.8}$$

Em que $\mathbf{f}_B = m\mathbf{g}_B + [X\ Y\ Z]^T$ ($\mathbf{g}_B = \mathbf{L}_{BE}\mathbf{g}_E, \mathbf{g}_E = [0\ 0\ g]^T$ sendo g, o módulo da aceleração da gravidade) e $\mathbf{G}_B = [\bar{L}\ M\ N]^T$. Além disso, o momento angular $\mathbf{h}_B = \mathbf{I}_B\boldsymbol{\omega}_B$, onde $\mathbf{I}_B$ é a matriz de inércia da aeronave dada pela Eq. 2.9.

$$\mathbf{I}_B = \begin{bmatrix} I_x & -I_{xy} & -I_{xz} \\ -I_{yx} & I_y & -I_{yz} \\ -I_{zx} & -I_{zy} & I_z \end{bmatrix} \tag{2.9}$$

Segundo Stevens e Lewis (1992), as forças e os momentos que estão presentes no movimento de uma aeronave são as de sustentação, peso, arrasto e tração e compõem as forças e os momentos resultantes. A força de sustentação é produzida majoritariamente pelas asas e é responsável por manter a aeronave no ar e fazer com que seja manobrável. A força peso é devido a atração gravitacional gerada pela Terra. A força de arrasto age de forma oposta ao movimento, devido a resistência do ar e por fim a força de tração, gerada pelos motores, propulsiona a aeronave.

Essas forças dependem dos ângulos de inclinação da aeronave em relação ao sistema de referência aerodinâmico F_A (de origem C), os ditos ângulo de ataque α e ângulo de derrapagem β (STEVENS; LEWIS, 1992), mostrados na Fig. 4.

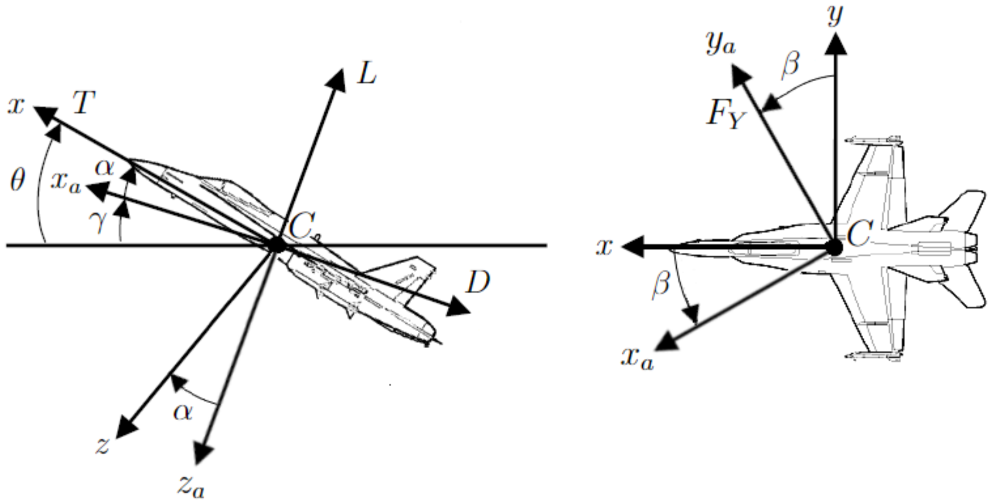


Figura 4 – F_B, F_A, α e β. Adaptado de Crassidis e Junkins (2012).

De acordo com a Fig. 4, $\theta = \alpha + \gamma$ sendo γ o ângulo de subida e conforme afirma Etkin e Reid (1996), os ângulos de ataque e derrapagem podem ser calculados pelas Eqs. 2.10 e 2.11, onde V é o módulo de $\mathbf{V}_B$ ($V = \sqrt{u^2 + v^2 + w^2}$):

$$\alpha = \arctan\left(\frac{w}{u}\right) \tag{2.10}$$

$$\beta = \arcsin\left(\frac{v}{V}\right) \tag{2.11}$$

Com essas ângulos, é possível obter as forças aerodinâmicas resultantes X, Y e Z decompondo as forças aerodinâmicas (definidas em F_A) em F_B. As forças aerodinâmicas são as de arrasto D, lateral F_Y e de sustentação L, nos sentidos, $-x_a$, y_a e $-z_a$ da Fig. 4 respectivamente. Como a tração do motor T já está em F_B no sentido x, ela não precisa ser decomposta. Dessa maneira, tem-se as Eqs. 2.12 (PAGLIONE; ZANARDI, 1990).

$$X = T - D\cos\alpha\cos\beta - F_Y\cos\alpha\sin\beta + L\sin\alpha \tag{2.12a}$$

$$Y = -D\sin\beta + F_Y\cos\beta \tag{2.12b}$$

$$Z = -D\sin\alpha\cos\beta - F_Y\sin\alpha\sin\beta - L\cos\alpha \tag{2.12c}$$

D, F_Y e L são calculados pelas Eqs. 2.13.

$$D = \frac{1}{2}\rho S V^2 C_D \tag{2.13a}$$
$$F_Y = \frac{1}{2}\rho S V^2 C_Y \tag{2.13b}$$
$$L = \frac{1}{2}\rho S V^2 C_L \tag{2.13c}$$

Onde ρ é a densidade do ar, S é a área da asa e C_D, C_Y e C_L são os coeficientes de força de arrasto, lateral e de sustentação, respectivamente. Em diferentes aplicações, é possível considerar diversas modelagens, levando em conta fatores como temperatura, número de Mach e outras variáveis relevantes para cada situação.

Paglione e Zanardi (1990) considera os efeitos de α, β, q, das deflexões angulares do leme δ_r, dos ailerons δ_a e do profundor δ_e. Sendo κ uma constate da função parabólica de C_D e $\bar{c}$ a corda média aerodinâmica da asa, os coeficientes de força são escritos de acordo com as Eqs. 2.14.

$$C_D = C_D(C_L) = C_{D_0} + \kappa C_L^2 \tag{2.14a}$$
$$C_Y = C_Y(\beta, \delta_r, \delta_a) = C_{Y_\beta}\beta + C_{Y_{\delta_r}}\delta_r + C_{Y_{\delta_a}}\delta_a \tag{2.14b}$$
$$C_L = C_L(\alpha, \delta_e, q) = C_{L_0} + C_{L_\alpha}\alpha + C_{L_{\delta_e}}\delta_e + C_{L_q}\frac{\bar{c}q}{2V} \tag{2.14c}$$

Em que:

- C_{D_0}: coeficiente de força de arrasto para $C_L = 0$;
- C_{Y_β}: derivada parcial aerodinâmica adimensional de força lateral devido à derrapagem;
- $C_{Y_{\delta_r}}$: derivada parcial aerodinâmica adimensional de força lateral devido à deflexão angular do leme;
- $C_{Y_{\delta_a}}$: derivada parcial aerodinâmica adimensional de força lateral devido à deflexão angular dos ailerons;
- C_{L_0}: coeficiente de força de sustentação para $\alpha = 0$;
- C_{L_α}: derivada parcial aerodinâmica adimensional de força de sustentação devido ao ângulo de ataque;
- $C_{L_{\delta_e}}$: derivada parcial aerodinâmica adimensional de força de sustentação devido à deflexão angular do profundor;

- C_{L_q}: derivada parcial aerodinâmica adimensional de de força sustentação devido à velocidade de arfagem.

De forma semelhante, pode-se obter os momentos aerodinâmicos resultantes $\bar{L}$, M e N pelas Eq. 2.15 onde b é a envergadura da asa.

$$\bar{L} = \frac{1}{2}\rho S b V^2 C_l \tag{2.15a}$$

$$M = \frac{1}{2}\rho S \bar{c} V^2 C_m \tag{2.15b}$$

$$N = \frac{1}{2}\rho S b V^2 C_n \tag{2.15c}$$

C_l, C_m e C_n são os coeficientes de momento de rolagem, arfagem e guinada respectivamente. Eles dependem de α, β, p, q, r, δ_r, δ_a e δ_e e são encontrados pelas Eqs. 2.16 (PAGLIONE; ZANARDI, 1990).

$$C_l = C_l(\beta, \delta_r, \delta_a, p, r) = C_{l_\beta}\beta + C_{l_{\delta_r}}\delta_r + C_{l_{\delta_a}}\delta_a + C_{l_p}\frac{bp}{2V} + C_{l_r}\frac{br}{2V} \tag{2.16a}$$

$$C_m = C_m(\alpha, \delta_e, q) = C_{m_0} + C_{m_\alpha}\alpha + C_{m_{\delta_e}}\delta_e + C_{m_q}\frac{\bar{c}q}{2V} \tag{2.16b}$$

$$C_n = C_n(\beta, \delta_r, \delta_a, p, r) = C_{n_\beta}\beta + C_{n_{\delta_r}}\delta_r + C_{n_{\delta_a}}\delta_a + C_{n_p}\frac{bp}{2V} + C_{n_r}\frac{br}{2V} \tag{2.16c}$$

Em que:

- C_{l_β}: derivada parcial aerodinâmica adimensional de momento de rolamento devido à derrapagem;
- $C_{l_{\delta_r}}$: derivada parcial aerodinâmica adimensional de momento de rolamento devido à deflexão angular do leme;
- $C_{l_{\delta_a}}$: derivada parcial aerodinâmica adimensional de momento de rolamento devido à deflexão angular dos ailerons;
- C_{l_p}: derivada parcial aerodinâmica adimensional de momento de rolamento devido à velocidade de rolagem;
- C_{l_r}: derivada parcial aerodinâmica adimensional de momento de rolamento devido à velocidade de guinada;
- C_{m_0}: coeficiente de momento de arfagem para $\alpha = 0$;
- C_{m_α}: derivada parcial aerodinâmica adimensional de momento de arfagem devido ao ângulo de ataque;

- $C_{m_{\delta_e}}$: derivada parcial aerodinâmica adimensional de momento de arfagem devido à deflexão angular do profundor;
- C_{m_q}: derivada parcial aerodinâmica adimensional de momento de arfagem devido à velocidade de guinada;
- C_{n_β}: derivada parcial aerodinâmica adimensional de momento de guinada devido à derrapagem;
- $C_{n_{\delta_r}}$: derivada parcial aerodinâmica adimensional de momento de guinada devido à deflexão angular do leme;
- $C_{n_{\delta_a}}$: derivada parcial aerodinâmica adimensional de momento de guinada devido à deflexão angular dos ailerons;
- C_{n_p}: derivada parcial aerodinâmica adimensional de momento de guinada devido à velocidade de rolagem;
- C_{n_r}: derivada parcial aerodinâmica adimensional de momento de guinada devido à velocidade de guinada.

Por fim, organizando todas as equações, tem-se o modelo dinâmico completo da aeronave, que é separado em equações cinemáticas e dinâmicas. Colocando na forma matricial, pelas Eqs. 2.3, 2.4, 2.5 e 2.6 tem-se as equações cinemáticas dadas pelas Eqs. 2.17 e 2.18. Pelas Eqs. 2.7, 2.8 e considerando que a aeronave é massivamente simétrica em torno do plano xz (na Eq. 2.9 $I_{xy} = I_{yx} = I_{yz} = I_{zy} = 0$ e $I_{zx} = I_{xz}$), tem-se as equações dinâmicas dadas pelas Eqs. 2.19 e 2.20. O subscrito -1 no caso de matrizes representa a função inversa.

- Equações cinemáticas:

$$\begin{bmatrix} \dot{x}_E \\ \dot{y}_E \\ \dot{z}_E \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{2.17}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{2.18}$$

- Equações dinâmicas:

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \frac{1}{m} \begin{bmatrix} X - mg\sin\theta \\ Y + mg\cos\theta\sin\phi \\ Z + mg\cos\theta\cos\phi \end{bmatrix} + \begin{bmatrix} rv - qw \\ pw - ru \\ qu - pv \end{bmatrix} \tag{2.19}$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} I_x & 0 & -I_{xz} \\ 0 & I_y & 0 \\ -I_{xz} & 0 & I_z \end{bmatrix}^{-1} \begin{bmatrix} \bar{L} + (I_y - I_z)qr + I_{xz}pq \\ M + (I_z - I_x)pr + I_{xz}(r^2 - p^2) \\ N + (I_x - I_y)pq - I_{xz}qr \end{bmatrix} \tag{2.20}$$

Pelas Eqs. 2.12 se encontram X, Y e Z, que dependem de α, β, D, F_Y e L, calculadas pelas Eqs. 2.10, 2.11 e 2.13 que por sua vez, dependem de C_D, C_Y e C_L calculados pelas Eqs. 2.14. Em resumo, X, Y e Z dependem das variáreis T, α, β, δ_r, δ_a, δ_e e q.

Pelas Eqs. 2.15 se encontram $\bar{L}$, M e N que dependem de C_l, C_m e C_n encontrados pelas Eqs. 2.16. Em resumo, $\bar{L}$, M e N dependem das variáreis α, β, δ_r, δ_a, δ_e, p, q e r.

2.1.1 Representação em espaço de estados: modelo dinâmico e de medidas

Na engenharia, a representação da dinâmica de um sistema pode ser feita em espaço de estados, sendo muito útil e compacta para sistemas de múltiplas entradas e múltiplas saídas. Ele é um modelo matemático que descreve um sistema físico através de um conjunto de equações diferenciais (tempo contínuo) ou de diferenças (tempo discreto) de primeira ordem. (NISE, 2012)

Essas equações relacionam as variáveis de entrada\controle, de saída\medida e a de estado do sistema, representadas respectivamente por $\mathbf{u} = [u_1\ u_2\ ...\ u_s]^T$, $\tilde{\mathbf{y}} = [\tilde{y}_1\ \tilde{y}_2\ ...\ \tilde{y}_m]^T$ e $\mathbf{x} = [x_1\ x_2\ ...\ x_n]^T$. Segundo Nise (2012), as variáveis de estado evoluem ao longo do tempo, dependendo não apenas dos valores atuais, mas também das entradas externas, enquanto que as variáveis de saída são funções somente das variáveis de estado.

Essas equações podem ser vistas a seguir conforme Stevens e Lewis (1992) pelas Eqs. 2.21, onde as funções vetoriais não lineares $\boldsymbol{f}(.) = [f_1(.)\ f_2(.)\ ...\ f_n(.)]^T$ e $\boldsymbol{h}(.) = [h_1(.)\ h_2(.)\ ...\ h_m(.)]^T$ representam a dinâmica do sistema no tempo contínuo e pelas Eqs. 2.22 no tempo discreto, com $\boldsymbol{f}_k(.) = [f_{k_1}(.)\ f_{k_2}(.)\ ...\ f_{k_n}(.)]^T$ e $\boldsymbol{h}_k(.) = [h_{k_1}(.)\ h_{k_2}(.)\ ...\ h_{k_m}(.)]^T$.

$$\dot{\mathbf{x}} = \boldsymbol{f}(\mathbf{x}, \mathbf{u}) \tag{2.21a}$$
$$\tilde{\mathbf{y}} = \mathbf{h}(\mathbf{x}) \tag{2.21b}$$

$$\mathbf{x}_k = \boldsymbol{f}_{k-1}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) \tag{2.22a}$$
$$\tilde{\mathbf{y}}_k = \boldsymbol{h}_k(\mathbf{x}_k) \tag{2.22b}$$

As equações do modelo dinâmico da aeronave mostradas anteriormente podem ser representadas em espaço de estados. Devido a sua não linearidade, um procedimento comum citado por Nise (2012), Crassidis e Junkins (2012) e Stevens e Lewis (1992) é a linearização, que trata de aproximar uma equação diferencial não linear por uma equação diferencial linear, válida somente para pequenas variações em torno de um ponto de equilíbrio.

Uma maneira de fazer essa linearização é utilizando a expansão em série de Taylor (BOYCE; DIPRIMA, 2010). Nessa expansão, uma função pode ser aproximada por uma soma de polinômio calculados a partir da derivada dessa função em torno de um ponto desejado. No caso de funções vetoriais utiliza-se a matriz jacobina.

Dessa maneira, as Eqs. 2.21 e 2.22 podem ser escritas na sua forma linear pelas Eqs. 2.23 e Eqs. 2.24.

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \tag{2.23a}$$

$$\tilde{\mathbf{y}} = \mathbf{C}\mathbf{x} \tag{2.23b}$$

$$\mathbf{x}_k = \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{B}_{k-1}\mathbf{u}_{k-1} \tag{2.24a}$$

$$\tilde{\mathbf{y}}_k = \mathbf{C}_k\mathbf{x}_k \tag{2.24b}$$

Em que $\mathbf{A}$ $(n \times n)$ é a matriz jacobiana de estado do sistema, $\mathbf{B}$ $(n \times s)$, a matriz jacobiana de entrada do sistema e $\mathbf{C}$ $(m \times n)$, a matriz jacobiana de saída do sistema, que podem ser calculadas, respectivamente, pelas Eqs. 2.25, 2.26 e 2.27:

$$\mathbf{A} = \frac{\partial \boldsymbol{f}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \tag{2.25}$$

$$\mathbf{B} = \frac{\partial \boldsymbol{f}}{\partial \mathbf{u}} = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \cdots & \cdots & \frac{\partial f_1}{\partial u_s} \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \frac{\partial f_n}{\partial u_1} & \cdots & \cdots & \frac{\partial f_n}{\partial u_s} \end{bmatrix} \tag{2.26}$$

$$\mathbf{C} = \frac{\partial \boldsymbol{h}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \cdots & \cdots & \frac{\partial h_1}{\partial x_n} \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \frac{\partial h_n m}{\partial x_1} & \cdots & \cdots & \frac{\partial h_m}{\partial x_n} \end{bmatrix} \tag{2.27}$$

Dessa forma, para o modelo dinâmico da aeronave, o vetor de estado é composto por 12 parâmetros de movimento: x, y, z, ϕ, θ, ψ, u, v, w, p, q, e r e 22 parâmetros aerodinâmicos: C_{D_0}, C_{Y_β}, $C_{Y_{\delta_r}}$, $C_{Y_{\delta_a}}$, C_{L0}, $C_{L\alpha}$, $C_{L_{\delta_e}}$, C_{L_q}, C_{l_β}, $C_{l_{\delta_r}}$, $C_{l_{\delta_a}}$, C_{l_p}, C_{l_r}, C_{m_0}, C_{m_α}, $C_{m_{\delta_e}}$, C_{m_q}, C_{n_β}, $C_{n_{\delta_r}}$, $C_{n_{\delta_a}}$, C_{n_p}, e C_{n_r}, totalizando $n = 34$ estados como pode ser visto na Eq 2.28.

Para o vetor de entrada, 4 parâmetros ($s = 4$) de controle afetam o estado do sistema: T, δ_r, δ_a e δ_e, dado pela Eq. 2.30 e por fim o vetor de saída/medida possui 12 parâmetros ($m = 12$) que podem ser medidos por sensores: α, β, V, p, q, q, r, x, y, z, ϕ, θ, ψ e encontrado pela Eq. 2.30.

$$\begin{aligned}\mathbf{x} = \; & [x_E \quad y_E \quad z_E \quad \phi \quad \theta \quad \psi \quad u \quad v \quad w \quad p \quad q \quad r \\ & C_{D_0} \quad C_{Y_\beta} \quad C_{Y_{\delta_r}} \quad C_{Y_{\delta_a}} \quad C_{L_0} \quad C_{L_\alpha} \quad C_{L_{\delta_e}} \quad C_{L_q} \quad C_{l_\beta} \quad C_{l_{\delta_r}} \quad C_{l_{\delta_a}} \quad C_{l_p} \\ & C_{l_r} \quad C_{m_0} \quad C_{m_\alpha} \quad C_{m_{\delta_e}} \quad C_{m_q} \quad C_{n_\beta} \quad C_{n_{\delta_r}} \quad C_{n_{\delta_a}} \quad C_{n_p} \quad C_{n_r}]^T\end{aligned} \tag{2.28}$$

$$\mathbf{u} = [T \quad \delta_r \quad \delta_a \quad \delta_e]^T \tag{2.29}$$

$$\tilde{\mathbf{y}} = [\alpha \quad \beta \quad V \quad \phi \quad \theta \quad \psi \quad p \quad q \quad r \quad x_E \quad y_E \quad z_E]^T \tag{2.30}$$

3 Conceitos de probabilidade

Durante a realização de um experimento, é crucial extrair dados relevantes para chegar a conclusões significativas. Diversas ferramentas são empregadas nesse processo, e de acordo com Devore (2006), a probabilidade é uma delas, concentrando-se no estudo da aleatoriedade e da incerteza.

Para que esse estudo seja feito e se possa lançar de artifícios matemáticos, algumas definições devem ser feitas. Conforme Assis et al. (2021) e Devore (2006):

- População: consiste em um conjunto de objetos bem definido, do qual se tem interesse estudar;
- Amostra: subconjunto da população que contém características dessa população;
- Variável aleatória (VA): uma função que mapeia um conjunto de resultados de um experimento (domínio) para um conjunto numérico (contradomínio), podendo ser contínua ou discreta e escalar ou vetorial.

O valor médio de uma VA retrata a quantidade média que se espera que a VA assuma durante várias repetições do experimento, o também chamado valor esperado. Para calcula-lo é preciso saber quais os valores possíveis que a VA pode ter e como sua probabilidade é distribuída.

Dito isso, seja $\boldsymbol{X}$ uma VA discreta e vetorial de tamanho $N \times 1$, com X_1, X_2, ..., X_N VA escalares, onde cada X_N tem m_i valores possíveis $x_i(j_i)$, com $j_i = 1, 2, ..., m_i$. Dessa forma, tem-se a definição do valor esperado de $\boldsymbol{X}$, denotado por $E[\boldsymbol{X}]$ e é calculado pela Eq. 3.1, (CRASSIDIS; JUNKINS, 2012) (Simon (2006) também retrata o valor esperado como a média, representada por $\bar{\boldsymbol{X}}$), onde $p(X_1, X_2, ..., X_N)$ é a função de densidade de probabilidade:

$$E[\boldsymbol{X}] = \bar{\boldsymbol{X}} = \sum_{j_1=1}^{m_1} ... \sum_{j_N=1}^{m_N} \begin{bmatrix} x_1(j_1) \\ \vdots \\ x_N(j_N) \end{bmatrix} p(X_1, X_2, ..., X_N) \tag{3.1}$$

O valor esperado pode ter algumas interpretações e segundo DeGroot e Schervish (2012), é uma medida que representa o "centro de massa" da função de densidade de probabilidade, ou seja, é onde os valores da VA se equilibram. Em muitas vezes na probabilidade e estatística pode ser usado como uma referência ou um ponto de partida para fazer análises e previsões.

A definição de valor esprado pode ser estendida para funções de VA. Dessa forma, a Eq. 3.1 pode ser generalizada para a Eq. 3.2, onde $g(\boldsymbol{X})$ é uma função da VA $\boldsymbol{X}$.

$$E[g(\boldsymbol{X})] = \sum_{j_1=1}^{m_1} ... \sum_{j_N=1}^{m_N} g\left(\begin{bmatrix} x_1(j_1) \\ \vdots \\ x_N(j_N) \end{bmatrix}\right) p(X_1, X_2, ..., X_N) \tag{3.2}$$

Quando o experimento trata de mais de uma VA, é necessário saber se há relação entre elas, como elas variam juntas e se são independentes. Para isso, a covariância trás um valor numérico para medir essa relação. Seja $\boldsymbol{Y}$ também uma VA discreta e vetorial de tamanho $M \times 1$ com média $\bar{\boldsymbol{Y}}$, a covariância entre $\boldsymbol{X}$ e $\boldsymbol{Y}$ pode ser encontrada a partir da Eq. 3.2, com $g(\boldsymbol{X}, \boldsymbol{Y}) = (\boldsymbol{X} - \bar{\boldsymbol{X}})(\boldsymbol{Y} - \bar{\boldsymbol{Y}})^T$ e é definida de pela Eq. 3.3 (SIMON, 2006).

$$\begin{aligned} C_{\boldsymbol{XY}} &= E[(\boldsymbol{X} - \bar{\boldsymbol{X}})(\boldsymbol{Y} - \bar{\boldsymbol{Y}})^T] \\ &= E[\boldsymbol{X}\boldsymbol{Y}^T] - \bar{\boldsymbol{X}}\bar{\boldsymbol{Y}}^T \end{aligned} \tag{3.3}$$

Quando $\boldsymbol{Y} = \boldsymbol{X}$, tem-se a autocovariância de $\boldsymbol{X}$ mostrada pela Eq. 3.4 abaixo.

$$\begin{aligned} C_{\boldsymbol{X}} &= E[(\boldsymbol{X} - \bar{\boldsymbol{X}})(\boldsymbol{X} - \bar{\boldsymbol{X}})^T] \\ &= \begin{bmatrix} E[(X_1 - \bar{X}_1)^2] & \cdots & E[(X_1 - \bar{X}_1)(X_N - \bar{X}_N)] \\ \vdots & \ddots & \vdots \\ E[(X_N - \bar{X}_N)(X_1 - \bar{X}_1)] & \cdots & E[(X_N - \bar{X}_N)^2] \end{bmatrix} \\ &= \begin{bmatrix} \sigma_1^2 & \cdots & \sigma_{1N} \\ \vdots & \ddots & \vdots \\ \sigma_{N1} & \cdots & \sigma_N^2 \end{bmatrix} \end{aligned} \tag{3.4}$$

Segundo Crassidis e Junkins (2012) as Eqs. 3.5 calculam a variância de X_i e a covariância de X_i e X_j, respectivamente.

$$\sigma_i^2 = E[(X_i - \bar{X}_i)^2] \tag{3.5a}$$

$$\sigma_{ij}^2 = E[(X_i - \bar{X}_i)(X_j - \bar{X}_j)] \tag{3.5b}$$

Através da variância σ^2 é possível calcular o desvio padrão σ fazendo $\sigma = \sqrt{\sigma^2}$. Caso X_i e X_j sejam independentes entre si, eles não possuem relação, logo $\sigma_{ij}^2 = E[(X_i - \bar{X}_i)(X_j - \bar{X}_j)] = 0$ e a Eq 3.4 se reduz a para a Eq. 3.6:

$$C_{\boldsymbol{X}} = \begin{bmatrix} \sigma_1^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_N^2 \end{bmatrix} \tag{3.6}$$

Para facilitar a notação, $X \sim (\bar{X}, \sigma^2)$ será usado para representar que a VA escalar X possui média $\bar{X}$ e variância σ^2. Quando uma VA é dita guassina ou normal, a notação se torna $X \sim N(\bar{X}, \sigma^2)$ e a função de densidade de probabilidade é dada pela Eq. 3.7.

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\bar{X})^2}{2\sigma^2}} \tag{3.7}$$

A distribuição guassiana apresenta formato de sino simétrico em torno de $\bar{X}$ como pode ser visto pela Fig. 5. A média $\bar{X}$ representa o centro da distribuição e sua mudança é responsável por uma translação em relação ao eixo horizontal. Já σ, representa a distância entre os valores e a média, ou seja, simboliza a dispersão das amostras em relação a média. (DEVORE, 2006)

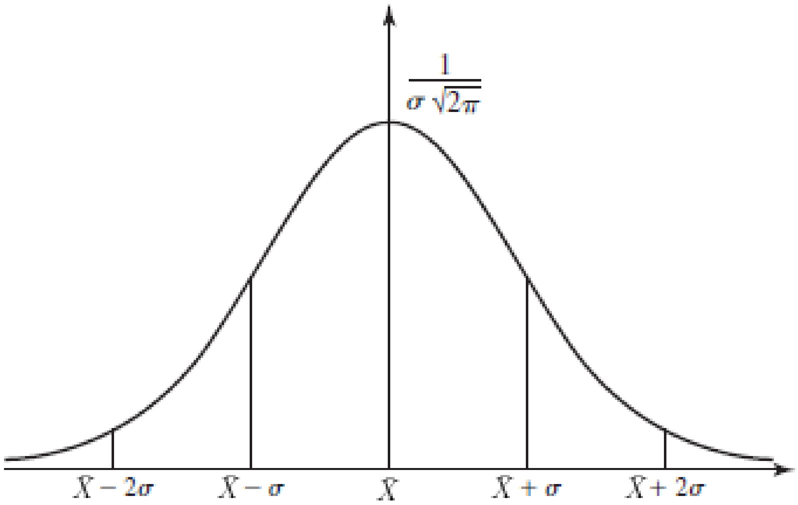


Figura 5 – Função de densidade de probabilidade gaussiana. Adaptado de DeGroot e Schervish (2012).

4 Estimação de parâmetros

No contexto da probabilidade e estatística, a inferência tem como objetivo tirar conclusões de uma população a partir de amostras. Dessa forma, com base nestas amostras, a inferência estatística propõe fazer extrapolações, expressas em probabilidade, sobre um conjunto de dados. Essa probabilidade demonstra quanta confiança pode ser depositada sobre essas conclusões. (MCCABE; A.; S., 2009)

Pela Fig. 6 pode-se visualizar o processo da inferência estatística, que funciona como uma função e leva um certo conjunto amostras para o conjunto população, onde são calculados parâmetros populacionais de interesse, como por exemplo a média, desvio padrão etc.

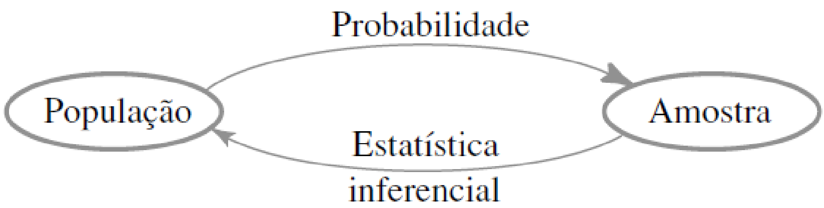


Figura 6 – Processo da inferência estatística. (DEVORE, 2006)

A estimação, assim como o teste de hipótese, é uma abordagem da inferência estatística que se utiliza dos dados amostrais para estimar os parâmetros populacionais desconhecidos, podendo ser determinísticos ou aleatórios, variantes ou invariantes no tempo e lineares ou não lineares.

De acordo com Crassidis e Junkins (2012) e seguindo sua notação, três grandezas são de interesse quando se fala em estimação de parâmetros: o valor verdadeiro (x), o valor medido ($\tilde{x}$) e o valor estimado ($\hat{x}$). O valor verdadeiro representa o valor real a ser estimado e na prática é desconhecido, pois em um experimento, o valor real sempre será afetado de alguma forma. (AGUIRRE, 2013) O valor medido é justamente o valor a ser determinado pelo instrumento/processo de medição. Por fim, o valor estimado é determinado pelo processo de estimação e faz uso do modelo do sistema e dos valores medidos. O grande objetivo da estimação de parâmetros é calcular o valor estimado, com base nas informações do sistema e no valor medido, para que se aproxime ao máximo do valor verdadeiro.

Para que esses parâmetros sejam estimados, Therrien (1992) define um estimador como uma combinação de amostras que estima algum parâmetro de interesse. De maneira matemática, pode-se escrever o estimador $\hat{\boldsymbol{\theta}}$ do vetor de estimativas $\boldsymbol{\theta} = [\theta_1\ \theta_2\ \ldots\ \theta_K]^T$

como sendo uma função $\boldsymbol{s}$ de um vetor de amostras $\tilde{\boldsymbol{x}}_\eta = [\tilde{x}_1 \ \tilde{x}_2 \ ... \ \tilde{x}_\eta]^T$. Desse modo, tem-se a Eq 4.1.

$$\hat{\boldsymbol{\theta}} = \boldsymbol{s}[\tilde{\boldsymbol{x}}_\eta] \tag{4.1}$$

E para parâmetros específicos, o valor numérico de $\hat{\theta}_i$ é dito como a estimativa do parâmetro θ_i. Logo pela Eq. 4.2:

$$\hat{\theta}_i = s_i[\tilde{\boldsymbol{x}}_\eta], \ i = 1, 2, ..., K \tag{4.2}$$

Como dito anteriormente, a média e o desvio padrão são geralmente parâmetros de interesse no momento de análise de dados. Seus estimadores são da forma mostrada na Eqs. 4.2 e são calculados de acordo com as Eqs. 3.1 e 3.5, considerando um VA escalar e uma função de densidade de probabilidade igualmente distribuída para todos os valores possíveis. Além disso, como se trata de um estimador, $\bar{\boldsymbol{X}} \rightarrow \hat{\mu}$, $\sigma \rightarrow \hat{\sigma}$ e $x \rightarrow \tilde{x}$ e tem-se as Eqs. 4.3 e 4.4.

$$\hat{\mu} = \frac{1}{\eta} \sum_{j=1}^{\eta} \tilde{x}_j \tag{4.3}$$

$$\hat{\sigma}^2 = \frac{1}{\eta} \sum_{j=1}^{\eta} [\tilde{x}_j - \hat{\mu}]^2 \tag{4.4}$$

De acordo com Crassidis e Junkins (2012), os métodos de estimação podem ser separados em dois grandes grupos, a estimação em lote (tradução livre de *batch estimation*) e a estimação sequencial. A estimação em lote pressupõe que todas as amostras estão disponíveis e a estimação sequencial recebe as amostras de forma sucessiva, conforme as amostras chegam a estimação é feita.

A interferência de entradas indesejadas deixa o sinal ruidoso e inviável para ser enviado aos demais sistema, ficando evidente a necessidade do uso de filtros. (AGUIRRE, 2013) Nesse ponto, os métodos de estimação sequencial podem ser usados para prever estados de um sistema dinâmico ao mesmo tempo que agem como um filtro.

4.1 Filtro de Kalman no tempo discreto

Além de ser um estimador sequencial, o filtro de Kalman também é recursivo, ou seja, ele utiliza as medidas $\tilde{\mathbf{y}}$ e o modelo do sistema dinâmico para calcular o valor estimado $\mathbf{x}$ e atualiza-lo conforme novas medidas sejam adquiridas. Para isso, ele propaga a média e covariância dos estados por meio do valor esperado (como visto no capítulo 3).

Dessa forma, a média do estado é a estimativa do estado e a covariância do estado é a covariância da do erro da estimativa. Em sua modelagem ainda é considerado processos ruidosos no modelo e na medição. (SIMON, 2006)

Semelhante às Eqs. 2.24, pode-se obter as Eqs. 4.5 do modelo dinâmico em tempo discreto, acrescentando-se os ruídos. Logo, com $k = 1, 2, ...$, $\mathbf{A} \rightarrow \boldsymbol{F}$, $\mathbf{B} \rightarrow \boldsymbol{G}$ e $\mathbf{C} \rightarrow \boldsymbol{H}$:

$$\mathbf{x}_k = \boldsymbol{F}_{k-1}\mathbf{x}_{k-1} + \boldsymbol{G}_{k-1}\mathbf{u}_{k-1} + \mathbf{w}_{k-1} \tag{4.5a}$$

$$\tilde{\mathbf{y}}_k = \boldsymbol{H}_k\mathbf{x}_k + \mathbf{v}_k \tag{4.5b}$$

$\mathbf{w}_k$ e $\mathbf{v}_k$ são ruídos de modelo e de medição, respectivamente. Simon (2006) e Crassidis e Junkins (2012) consideram esses ruídos como sendo guassianos brancos com matrizes de variância $\boldsymbol{Q}_k$ e $\boldsymbol{R}_k$ (assim como na Eq. 3.6). Além disso, para todo k, $\mathbf{w}_k$ e $\mathbf{v}_k$ são não correlacionados entre si para frente ou para trás no tempo. Todas essas relações podem ser vistas nas Eqs. 4.6, 4.6 e 4.8 a seguir.

$$\begin{gathered} \mathbf{w}_k \sim N(0, \boldsymbol{Q}_k) \\ E[\mathbf{w}_k\mathbf{w}_j^T] = \begin{cases} 0 & k \neq j \\ \boldsymbol{Q}_k & k = j \end{cases} \end{gathered} \tag{4.6}$$

$$\begin{gathered} \mathbf{v}_k \sim N(0, \boldsymbol{R}_k) \\ E[\mathbf{v}_k\mathbf{v}_j^T] = \begin{cases} 0 & k \neq j \\ \boldsymbol{R}_k & k = j \end{cases} \end{gathered} \tag{4.7}$$

$$E[\mathbf{v}_k\mathbf{w}_k^T] = 0 \tag{4.8}$$

Conforme Simon (2006), caso todas as medidas de $\tilde{\mathbf{y}}$ estejam á disposição até e incluindo o tempo k, é definido a chamada estimativa a *posteriori* $\hat{\mathbf{x}}_k^+$. Já, caso todas as medidas estejam disponíveis até mas não incluindo o tempo k, a estimativa a *priori* é definida e denotada por $\hat{\mathbf{x}}_k^-$. Em outras palavras, $\hat{\mathbf{x}}_k^+$ é a estimativa de $\mathbf{x}_k$ depois da medida de $\tilde{\mathbf{y}}_k$ ser computada, enquanto que $\hat{\mathbf{x}}_k^-$ é a estimativa de $\mathbf{x}_k$ antes da medida de $\tilde{\mathbf{y}}_k$ ser processada. Por isso, a estimativa a *posteriori* é melhor que a estimativa a *priori*, pois mais informação é considerada no seu cálculo.

As estimativas a *priori* e a *posteriori* pode ser encontras de acordo com Simon (2006) e Crassidis e Junkins (2012) aplicando o valor esperado em $\mathbf{x}_k$ e considerando as medidas $\tilde{\mathbf{y}}$. Dessa forma, tem-se a Eqs. 4.9a e 4.9b a seguir.

$$\hat{\mathbf{x}}_k^- = E[\mathbf{x}_k|\tilde{\mathbf{y}}_1, \tilde{\mathbf{y}}_2, ..., \tilde{\mathbf{y}}_{k-1}] = \boldsymbol{F}_{k-1}\hat{\mathbf{x}}_{k-1}^+ + \boldsymbol{G}_{k-1}\mathbf{u}_{k-1} \tag{4.9a}$$

$$\hat{\mathbf{x}}_k^+ = E[\mathbf{x}_k|\tilde{\mathbf{y}}_1, \tilde{\mathbf{y}}_2, ..., \tilde{\mathbf{y}}_k] = \hat{\mathbf{x}}_k^- + \boldsymbol{K}_k(\tilde{\mathbf{y}}_k - \boldsymbol{H}_k\hat{\mathbf{x}}_k^-) \tag{4.9b}$$

Em que $E[\mathbf{x}_k|\tilde{\mathbf{y}}_1, \tilde{\mathbf{y}}_2, ..., \tilde{\mathbf{y}}_k]$ é o valor esperado de $\mathbf{x}_k$ condicionado às medidas até e incluindo o tempo k e $E[\mathbf{x}_k|\tilde{\mathbf{y}}_1, \tilde{\mathbf{y}}_2, ..., \tilde{\mathbf{y}}_{k-1}]$, o valor esperado de $\mathbf{x}_k$ condicionado às medidas até e não incluindo o tempo k. Como visto na Eq. 3.1, o valor esperado também é dito como a média e dessa forma as Eqs. 4.9 ditam como a média do estado $\hat{\mathbf{x}}_k$ varia no tempo.

Para saber como a covariância muda no tempo, o valor esperado é aplicado ao erro da estimativa no tempo k $(\mathbf{x}_k - \hat{\mathbf{x}}_k)$, semelhante á Eq. 3.4, tanto para a estimativa a *posteriori* quanto para a *priori*. Simon (2006) e Crassidis e Junkins (2012) aplicam o valor esperado e obtêm as Eqs. 4.10a e 4.10b:

$$\boldsymbol{P}_k^- = E[(\mathbf{x}_k - \hat{\mathbf{x}}_k^-)(\mathbf{x}_k - \hat{\mathbf{x}}_k^-)^T] = \boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1}^+\boldsymbol{F}_{k-1}^T + \boldsymbol{Q}_{k-1} \tag{4.10a}$$

$$\boldsymbol{P}_k^+ = E[(\mathbf{x}_k - \hat{\mathbf{x}}_k^+)(\mathbf{x}_k - \hat{\mathbf{x}}_k^+)^T] = (\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H}_k)\boldsymbol{P}_k^- \tag{4.10b}$$

Em que $\boldsymbol{I}$ é a matriz identidade e $\boldsymbol{K}_k$ é o ganho de Kalman no tempo k, encontrado a partir da minimização da função custo $\boldsymbol{J}_k = Tr(\boldsymbol{P}_k^+)$ ($Tr()$ representa o traço da matriz). Desse modo, como mostrado por Simon (2006) e Crassidis e Junkins (2012) chegam na Eq. 4.11, derivando $\boldsymbol{J}_k$ em relação a $\boldsymbol{K}_k$ e igualando a 0.

$$\boldsymbol{K}_k = \boldsymbol{P}_k^-\boldsymbol{H}_k^T(\boldsymbol{H}_k\boldsymbol{P}_k^-\boldsymbol{H}_k^T + \boldsymbol{R}_k)^{-1} \tag{4.11}$$

Como visto pelas equações acima, a recursividade está presente e a linha do tempo mostrada na Fig. 7 abaixo ajuda a visualizar como as estimativas e as covariâncias a *posteriori* e a *priori* são atualizadas.

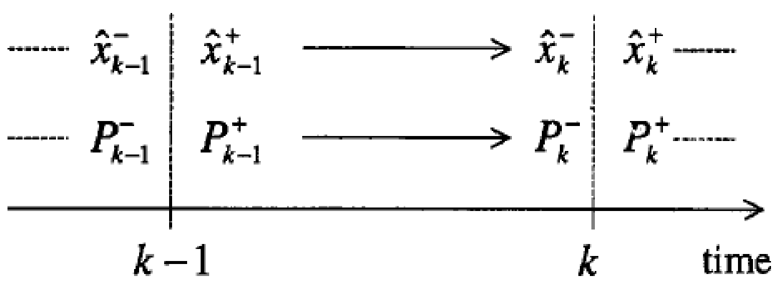


Figura 7 – Linha do tempo das estimativas e das covariâncias. (SIMON, 2006)

No tempo $k-1$ tem-se $\hat{\mathbf{x}}_{k-1}^-$ e $\boldsymbol{P}_{k-1}^-$ antes da informação de $k-1$ ser processada. Assim que a medida é feita, $\hat{\mathbf{x}}_{k-1}^+$ e $\boldsymbol{P}_{k-1}^+$ são calculados. Quando o tempo k chega, antes de computar a medida, $\hat{\mathbf{x}}_k^-$ e $\boldsymbol{P}_k^-$ são calculados. Após a medida em k ser processada, os termos $\hat{\mathbf{x}}_k^+$ e $\boldsymbol{P}_k^+$ são encontrados. Dessa maneira, de forma sequencial, o FK atualiza a estimativa do estado a cada instante.

Como visto nas Eqs. 4.5, o índice k tem início em 1, logo, para calcular $\hat{\mathbf{x}}_1$ é necessário saber $\hat{\mathbf{x}}_0$. Para isso, já que a medida $\tilde{\mathbf{y}}$ também começa em $k = 1$ é necessário fazer uma estimativa antes que as medidas de fato comecem, em $k = 0$. Usando as Eqs. 4.9b e 4.10b tem-se a inicialização do filtro dada pelas Eqs. 4.12a e 4.12b.

$$\hat{\mathbf{x}}_0^+ = E[\mathbf{x}_0] \tag{4.12a}$$

$$\boldsymbol{P}_0^+ = E[(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+)(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+)^T] \tag{4.12b}$$

Colocando todas as equações e conceitos apresentados, o FK pode ser resumido no algorítimo da Fig. 8 abaixo.

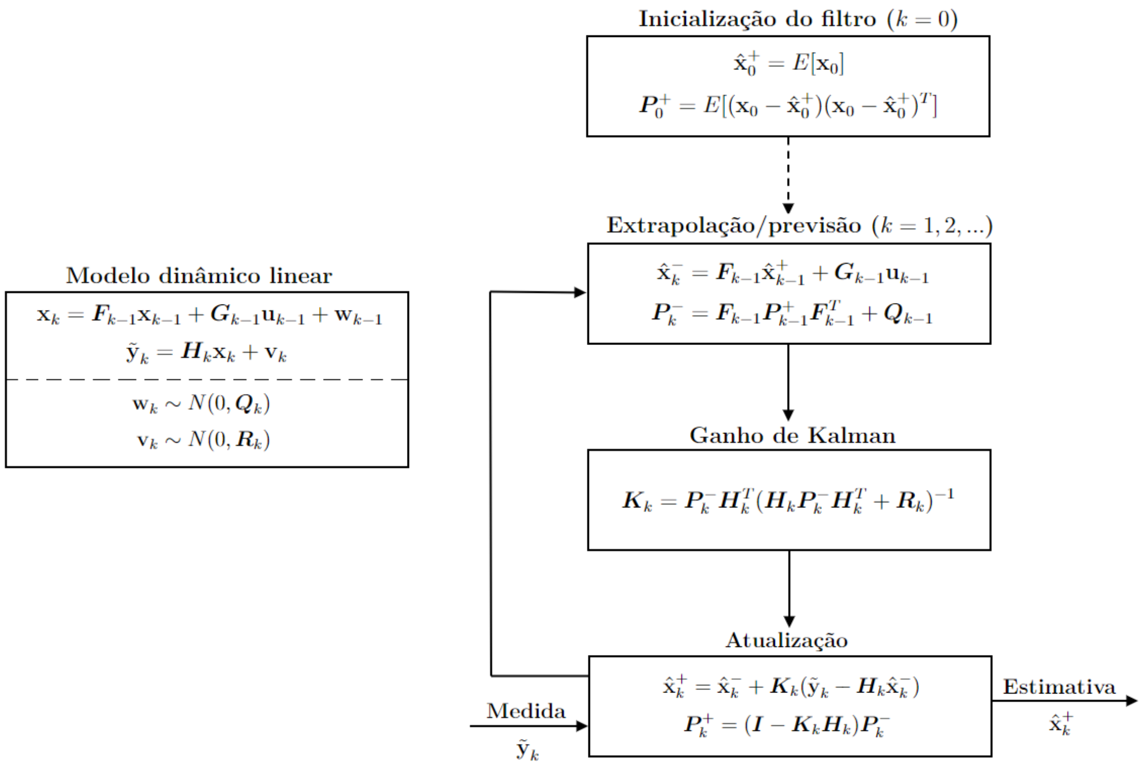


Figura 8 – Algoritmo do FK. Baseado em Busarello e Simões (2019).

O primeiro passo é a inicialização do filtro (Eqs. 4.12a e 4.12b), onde é feita uma estimativa inicial da média do estado e da covariância (após a inicialização do filtro, os parâmetros $\hat{\mathbf{x}}_0^+$ e $\boldsymbol{P}_0^+$ não são mais utilizados, por isso a representação da seta tracejada).

Após isso, o FK trabalha de forma recursiva. Na fase de extrapolação/previsão (Eqs. 4.9a e 4.10a), a estimativa a *priori* da média do estado e da covariância são feitas com base no modelo e na estimativa inicial. Com essas informações, o ganho de Kalman (Eq. 4.11) pode ser calculado e usado na fase de atualização. Nessa fase, além da medida entrar no cálculo, a estimativa a *posteriori* da média do estado e da covariância são calculadas (Eqs. 4.9b e 4.10b), e são realimentadas para a fase de extrapolação/previsão e o processo continua de forma recursiva e sequencial.

Outra maneira de visualizar o FK é por um diagrama de blocos, demonstrando como cada termo está conectado e quais cálculos são feitos para ter cada variável. Dessa maneira, a Fig. 9 mostra como o FK se conecta com o modelo, onde z^{-1} representa um atraso de uma amostra.

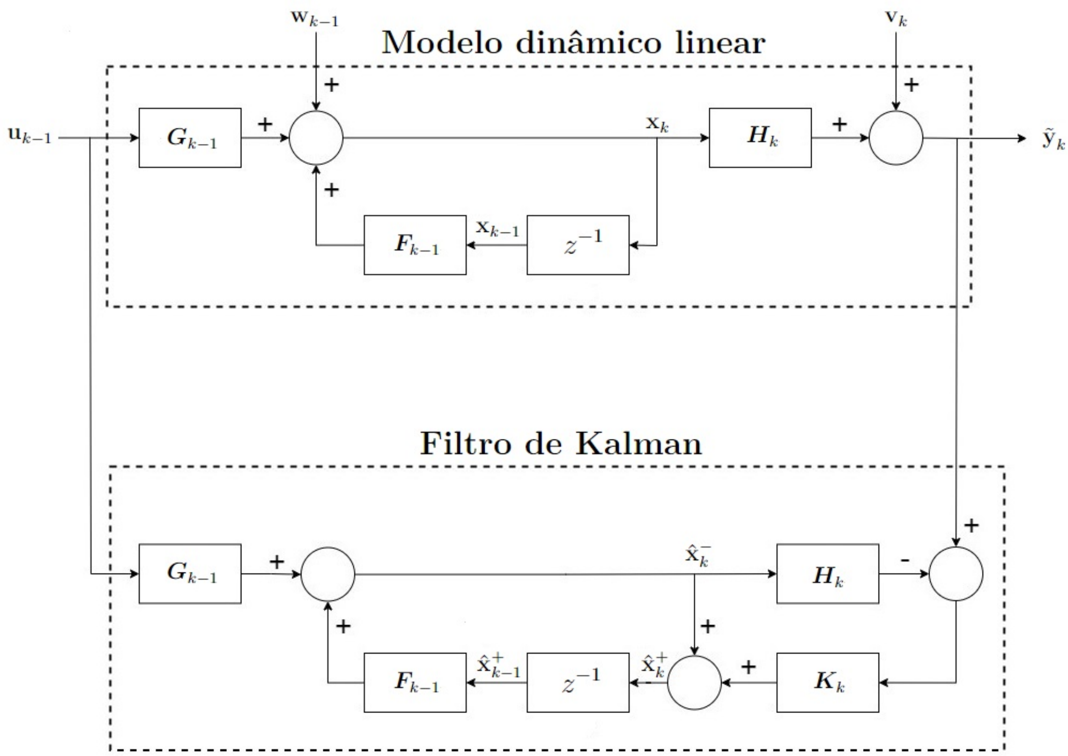


Figura 9 – FK e modelo em diagrama de blocos. Adaptado de Terejanu (2009).

4.2 Filtro de Kalman estendido no tempo discreto

O filtro de Kalman estendido é uma variação do FK que abrange modelos dinâmicos não lineares. Devido a complexidade desses modelos, um recurso comumente usado é realizar a linearização, como mostrado na seção 2.1.1 que conforme Simon (2006), é feita em torno da sua própria estimativa, refletindo a ideia de que o estado verdadeiro está razoavelmente próximo do estado estimado.

Adicionando os ruídos de processo e de medida no modelo das Eqs. 2.22, chega-se no modelo dinâmico não linear do FKE pelas Eqs. 4.13. Logo, com $k = 1, 2, ...$:

$$\mathbf{x}_k = \boldsymbol{f}_{k-1}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}) \tag{4.13a}$$
$$\tilde{\mathbf{y}}_k = \boldsymbol{h}_k(\mathbf{x}_k, \mathbf{v}_k) \tag{4.13b}$$

Simon (2006) aplica a expansão em série de Taylor no modelo das Eqs. 4.13 em torno dos pontos $\mathbf{x}_{k-1} = \hat{\mathbf{x}}^{*+}_{k-1}$, $\mathbf{u}_{k-1} = \mathbf{u}_{k-1}$ (aqui é considerado que a entrada de controle é perfeitamente conhecida), $\mathbf{w}_{k-1} = 0$, $\mathbf{x}_k = \hat{\mathbf{x}}^{*-}_k$ e $\mathbf{v}_k = 0$. Dessa forma, ele chega em uma linearização de primeira ordem dada pelas Eqs. 4.14:

$$\mathbf{x}^*_k = \boldsymbol{F}^*_{k-1}\mathbf{x}^*_{k-1} + \mathbf{u}^*_{k-1} + \mathbf{w}^*_{k-1} \tag{4.14a}$$
$$\tilde{\mathbf{y}}^*_k = \boldsymbol{H}^*_{k-1}\mathbf{x}^*_k + \mathbf{z}_k + \mathbf{v}^*_k \tag{4.14b}$$

Onde os termos com * simbolizam o modelo linearizado e são calculados pelas Eqs. 4.15 e 4.16 a seguir:

$$\boldsymbol{F}^*_{k-1} = \left.\frac{\partial \boldsymbol{f}_{k-1}}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}^{*+}_{k-1}} \tag{4.15a}$$
$$\mathbf{u}^*_{k-1} = \boldsymbol{f}_{k-1}(\hat{\mathbf{x}}^{*+}_{k-1}, \mathbf{u}_{k-1}, 0) - \boldsymbol{F}^*_{k-1}\hat{\mathbf{x}}^{*+}_{k-1} \tag{4.15b}$$
$$\mathbf{w}^*_{k-1} = \boldsymbol{L}_{k-1}\mathbf{w}_{k-1} \tag{4.15c}$$
$$\mathbf{w}^*_k \sim N(0, \boldsymbol{L}_k\boldsymbol{Q}_k\boldsymbol{L}^T_k)$$
$$\boldsymbol{L}_{k-1} = \left.\frac{\partial \boldsymbol{f}_{k-1}}{\partial \mathbf{w}}\right|_{\hat{\mathbf{x}}^{*+}_{k-1}} \tag{4.15d}$$

$$\boldsymbol{H}^*_k = \left.\frac{\partial \boldsymbol{h}_k}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}^{*-}_k} \tag{4.16a}$$
$$\mathbf{z}_k = \boldsymbol{h}_k(\hat{\mathbf{x}}^{*-}_k, 0) - \boldsymbol{H}^*_k\hat{\mathbf{x}}^{*-}_k \tag{4.16b}$$
$$\mathbf{v}^*_k = \boldsymbol{M}_k\mathbf{v}_k \tag{4.16c}$$
$$\mathbf{v}^*_k \sim N(0, \boldsymbol{M}_k\boldsymbol{R}_k\boldsymbol{M}^T_k)$$
$$\boldsymbol{M}_k = \left.\frac{\partial \boldsymbol{h}_k}{\partial \mathbf{v}}\right|_{\hat{\mathbf{x}}^{*-}_k} \tag{4.16d}$$

Onde $\boldsymbol{F}^*$ é a matriz de estado do calculada em $\hat{\mathbf{x}}^{*+}_{k-1}$ e $\boldsymbol{H}^*$ a matriz de saída calculada em $\hat{\mathbf{x}}^{*-}_k$, calculadas pelas jacobianas das Eqs. 2.25 e 2.27 respectivamente. Além disso, $\mathbf{u}^*$ é a entrada de controle e $\mathbf{w}^*$ e $\mathbf{v}^*$ são os ruídos de modelo e de medição, com variância $\boldsymbol{L}_k\boldsymbol{Q}_k\boldsymbol{L}^T_k$ e $\boldsymbol{M}_k\boldsymbol{R}_k\boldsymbol{M}^T_k$, respectivamente. Onde $\boldsymbol{L}_k$ e $\boldsymbol{M}_k$ são as jacobianas dos ruídos calculadas em $\hat{\mathbf{x}}^{*+}_{k-1}$ e $\hat{\mathbf{x}}^{*-}_k$.

Após a linearização, o modelo dinâmico não linear se tornou linear (é possível notar uma semelhança entre as Eqs. 4.5 e 4.14) e por causa a isso, o FKE faz uso das equações dos estimadores encontradas no FK (SIMON, 2006), vistas nas Eqs. 4.9, 4.10, 4.11 e 4.12. Então, tem-se, de forma semelhante, as Eqs. 4.17, 4.18, 4.19 e 4.20 a seguir.

$$\hat{\mathbf{x}}_k^{*-} = \boldsymbol{f}_{k-1}(\hat{\mathbf{x}}_{k-1}^{*+}, \mathbf{u}_{k-1}, 0) \tag{4.17a}$$

$$\hat{\mathbf{x}}_k^{*+} = \hat{\mathbf{x}}_k^{*-} + \boldsymbol{K}_k^*(\tilde{\mathbf{y}}_k^* - \boldsymbol{h}_k(\hat{\mathbf{x}}_k^{*-}, 0)) \tag{4.17b}$$

$$\boldsymbol{P}_k^{*-} = \boldsymbol{F}_{k-1}^* \boldsymbol{P}_{k-1}^{*+} \boldsymbol{F}_{k-1}^{*T} + \boldsymbol{L}_{k-1} \boldsymbol{Q}_{k-1} \boldsymbol{L}_{k-1}^T \tag{4.18a}$$

$$\boldsymbol{P}_k^{*+} = (\boldsymbol{I} - \boldsymbol{K}_k^* \boldsymbol{H}_k^*) \boldsymbol{P}_k^{*-} \tag{4.18b}$$

$$\boldsymbol{K}_k^* = \boldsymbol{P}_k^{*-} \boldsymbol{H}_k^{*T} (\boldsymbol{H}_k^* \boldsymbol{P}_k^{*-} \boldsymbol{H}_k^{*T} + \boldsymbol{M}_k \boldsymbol{R}_k \boldsymbol{M}_k^T)^{-1} \tag{4.19}$$

$$\hat{\mathbf{x}}_0^{*+} = E[\mathbf{x}_0^*] \tag{4.20a}$$

$$\boldsymbol{P}_0^{*+} = E[(\mathbf{x}_0^* - \hat{\mathbf{x}}_0^{*+})(\mathbf{x}_0^* - \hat{\mathbf{x}}_0^{*+})^T] \tag{4.20b}$$

Da mesma forma que as equações são adaptadas, o algoritmo da Fig. 8 pode ser modificado para incluir as jacobinas e obter o algoritmo de funcionamento do FKE dado pela Fig. 10 a seguir.

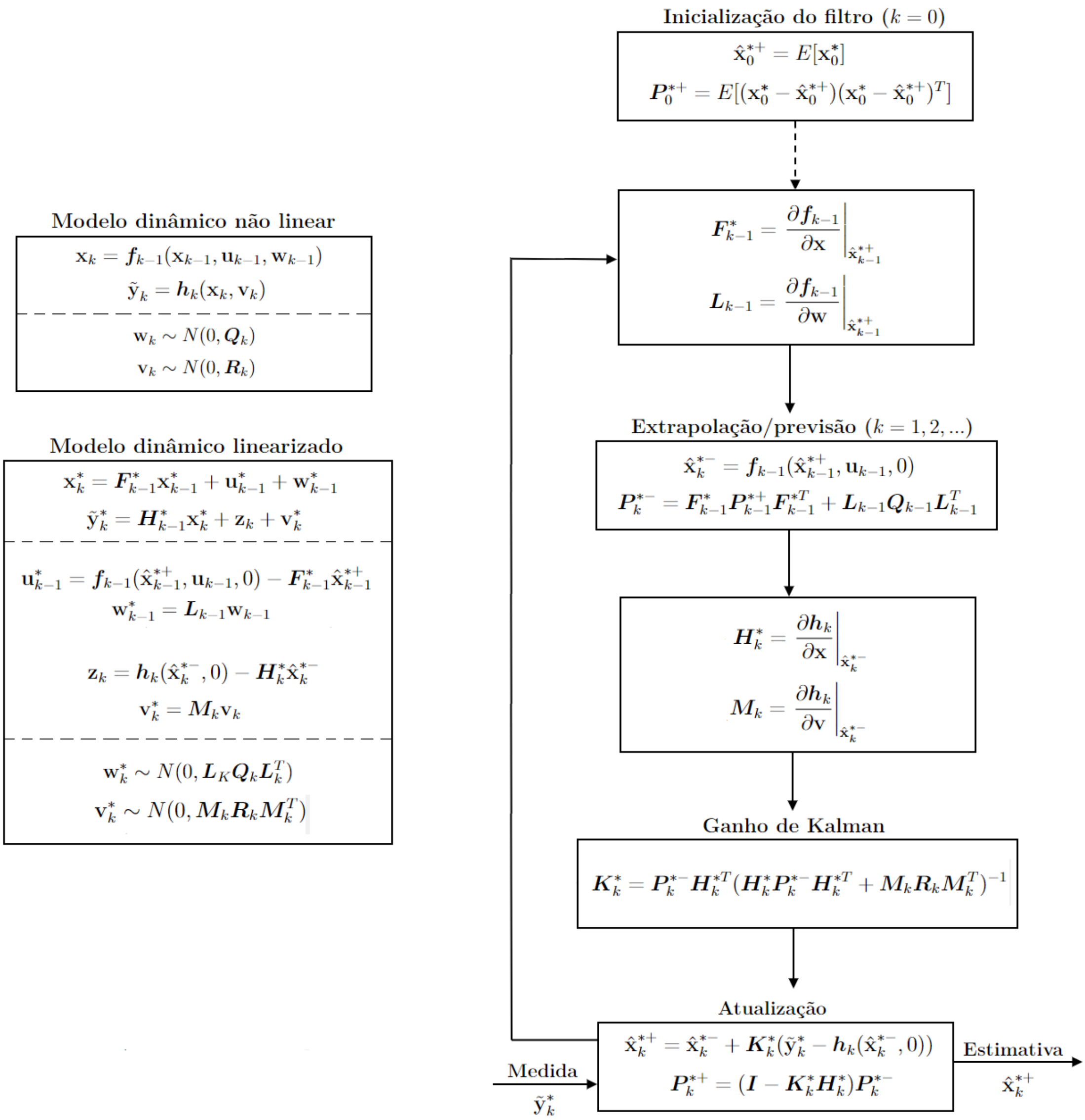


Figura 10 – Algoritmo do FKE.

5 Metodologia

A metodologia do trabalho apresenta o fluxograma da solução implementada para realizar a estimação dos parâmetros aerodinâmicos do caça Mirage III em condição de voo reto nivelado e curva coordenada. Para isso, os programas MATLAB® e Simulink® foram usados na aplicação das equações do modelo dinâmico da aeronave, do modelo de sensor e do FKE.

5.1 Objeto de estudo: Mirage III

O Mirage III (Fig. 11) é um caça de asa delta com capacidade para um tripulante, desenvolvido pela fabricante francesa *Dassault Aviation*. Ele é destinado às missões de interceptação, reconhecimento, combate aéreo, bombardeiro e ataque ao solo. Fez seu primeiro voo em 1956 e nas décadas de 70 e 80 se destacou por se tornar um dos primeiros aviões europeus a atingir velocidades supersônicas. A partir daí, se tornou um sucesso em diversas forças aéreas do mundo, sendo, por um longo período, o principal avião de defesa da Força Aérea Francesa.

Além disso, em 1970, a Força Aérea Brasileira adquiriu 17 modelos para serem concedidos ao 1º Grupo de Defesa Aérea, estabelecido na base de Anápolis, em Goiás. Ao longo do tempo, passou por programas de modernização e continuaram em operação até dezembro de 2005, acumulando mais de 67.000 horas de voo.

Figura 11 – Caça Mirage III. (BRASILEIRA, 2015)

Sua configuração padrão possui aproximadamente 15 m de comprimento, $7,5$ m de envergadura, 1 motor turbojato SNECMA Atar 09C capaz de gerar empuxo de até

$43,2\ KN$. A Tab. 1 a seguir, foi feita com base em Aviation (2016) e Paglione e Zanardi (1990) e mostra as suas características gerais.

Tabela 1 – Características gerais do Mirage III.

<table>
<tr><th>Característica</th><th colspan="2">Valor</th></tr>
<tr><td>Comprimento</td><td colspan="2">$15\ m$</td></tr>
<tr><td>Envergadura (b)</td><td colspan="2">$7,5\ m$</td></tr>
<tr><td>Corda média aerodinâmica ($\bar{c}$)</td><td colspan="2">$5,25\ m$</td></tr>
<tr><td>Área da asa (S)</td><td colspan="2">$36\ m^2$</td></tr>
<tr><td>Empuxo máximo</td><td colspan="2">$43,2\ KN$</td></tr>
<tr><td>Massa (m)</td><td colspan="2">$7400\ kg$</td></tr>
<tr><td>Momentos de inércia</td><td>$I_x = 9 \cdot 10^4\ kg \cdot m^2$
$I_y = 5,4 \cdot 10^4\ kg \cdot m^2$
$I_z = 6 \cdot 10^4\ kg \cdot m^2$</td><td>$I_{xy} = 0\ kg \cdot m^2$
$I_{xz} = 1,8 \cdot 10^3\ kg \cdot m^2$
$I_{yz} = 0\ kg \cdot m^2$</td></tr>
<tr><td>Constate da função parabólica de $C_L \times C_D$ (κ)</td><td colspan="2">$0,4$</td></tr>
</table>

Além disso, a Tab. 2 mostra seus parâmetros aerodinâmicos (derivadas parciais aerodinâmicas adimensionais de força e momento).

Tabela 2 – Características aerodinâmicas do Mirage III. (PAGLIONE; ZANARDI, 1990)

Parâmetro	**Valor (adimensional)**
C_{D_0}	$0,015$
C_{D_0}	$0,015$
C_{Y_β}	$-0,6$
$C_{Y_{\delta_r}}$	$0,075$
$C_{Y_{\delta_a}}$	$0,01$
C_{L_0}	0
C_{L_α}	$2,204$
$C_{L_{\delta_e}}$	$0,7$
C_{L_q}	0
C_{l_β}	$-0,05$
$C_{l_{\delta_r}}$	$0,018$
$C_{l_{\delta_a}}$	$-0,3$
C_{l_p}	$-0,25$
C_{l_r}	$0,06$
C_{m_0}	0
C_{m_α}	$-0,17$
$C_{m_{\delta_e}}$	$-0,45$
C_{m_q}	$-0,4$
C_{n_β}	$0,15$
$C_{n_{\delta_r}}$	$-0,085$
$C_{n_{\delta_a}}$	0
C_{n_p}	$0,055$
C_{n_r}	$-0,7$

5.2 Solução geral

A modelagem foi implementada no Simulink® seguindo a abordagem de diagrama de blocos, como pode ser visto na Fig. 12 abaixo. Ela é composta por três subsistemas, o MODELO NAO LINEAR - MIRAGE III, o SENSORES e o FILTRO DE KALMAN ESTENDIDO e utiliza um solucionador *ode4* e passo fixo de $dt = 0,01$.

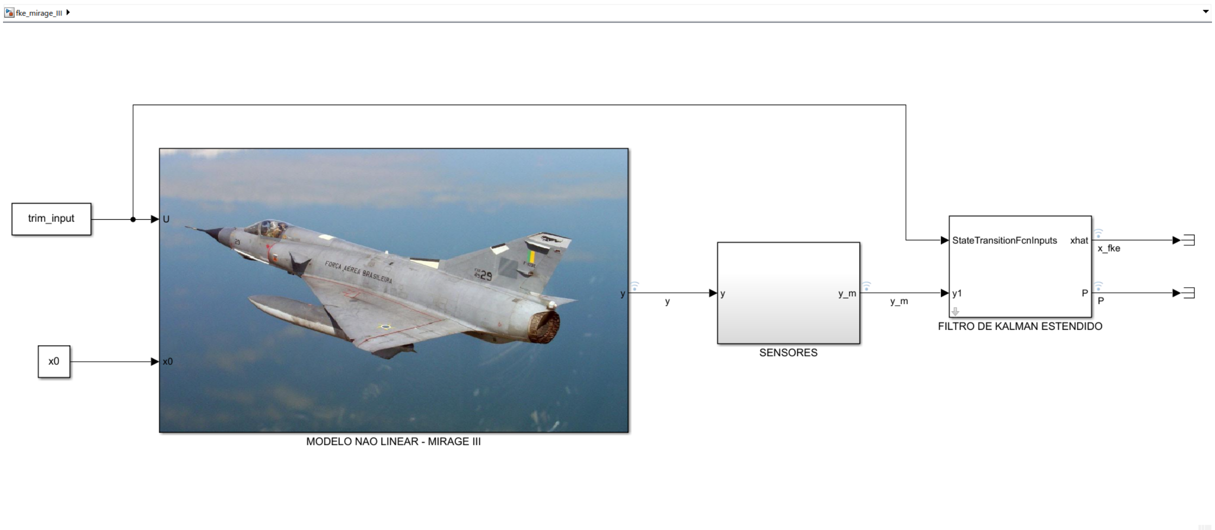


Figura 12 – Modelo completo.

No MODELO NÃO LINEAR - MIRAGE III estão todas as equações expostas na seção 2.1 (para mais detalhes, ver Apêndice B). Possui como entradas, o vetor de entrada\controle $\mathbf{u} = [T\ \delta_r\ \delta_a\ \delta_e]^T$ e as condições iniciais do modelo dinâmico.

Já o subsistema SENSORES, engloba os sensores de ângulos de ataque, derrapagem, tubo de Pitot, giroscópio e GPS (*Global Position System*), tendo como saída o vetor $\tilde{\mathbf{y}} = [\alpha\ \beta\ V\ \phi\ \theta\ \psi\ p\ q\ r\ x_E\ y_E\ z_E]^T$. A escolha desse modelo foi baseada em Quan (2017) e Ascorti (2013), e leva em consideração que o sensor é composto por dois termos principais, um ruído (considerado como gaussiano branco) e uma amostragem\discretização igual a dt, como pode ser visto na Fig. 13 abaixo. (A configuração desses blocos pode ser vista no Apêndice C)

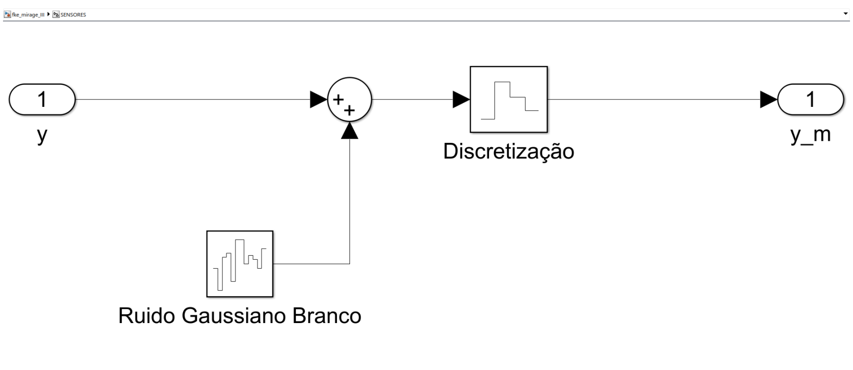


Figura 13 – Subsistema SENSORES.

Por fim, o subsistema FILTRO DE KALMAN ESTENDIDO implementa toda a lógica exposta na seção 4.2, com os passos de previsão, cálculo do ganho de Kalman e atualização, além de calcular, numericamente, as matrizes jacobianas de estado e saída.

Possui como entradas o vetor de entrada\controle e as medidas provindas dos sensores. Além do mais, leva em consideração uma inicialização (média e matriz de covariância inicial do estado), os ruídos de medição e modelo e as funções não lineares dos estados e das saídas (mais detalhes da configuração no Apêndice D). Como resultado, tem as saídas de estado estimado e a matriz de covariância.

5.3 Condições de voo

Para realizar uma condição de voo especificada, além de se conhecer as condições iniciais, é necessário saber quais combinações das entradas (tração do motor, deflexão angular do leme, ailerons e profundor) geram as saídas desejadas, ou em outras palavras, é necessário realizar uma trimagem.

5.3.1 Condição de voo reto e nivelado

Na condição de voo reto e nivelado, espera-se que a aeronave realize uma trajetória reta, sem aumentar ou diminuir sua altitude e permaneça paralela ao solo. Para que isso aconteça, a força de sustentação deve igualar a força peso, a tração do motor deve ser suficiente para manter a aeronave em velocidade constante e compensar a força de arrasto e as superfícies de controle devem ser ajustadas para manter o equilíbrio de momento e evitar alteração nos ângulos de arfagem, rolagem e guinada.

Em uma operação padrão de voo de cruzeiro, o Mirage III atinge 10.000 m ($\rho_{10.000} = 0,73\ kg/m^3$) de altitude e uma velocidade de 980 km/h ($272,22\ m/s$). Aqui vale frisar que, apesar da sua capacidade supersônica, esse trabalho não considera nenhum efeito desse tipo e a altitude e velocidade escolhidas mantém a modelagem dentro dos limites subsônicos.

Dado esse voo de cruzeiro, as entradas e condições iniciais (Tab. 3) para que aconteça um voo reto (na direção x_E) e nivelado foram calculadas com o auxílio do MATLAB® , onde o modelo Simulink® foi trimado de forma numérica ($trim\_input$ e $x0$ da Fig. 12 respectivamente (ver Apêndice A)). Além disso, a trajetória resultante pode ser vista na Fig. 14, com um tempo de simulação de 200 s.

Tabela 3 – Condições para voo reto e nivelado.

Parâmetro	**Valor**
Condições iniciais	
x_E	$0\ m$
y_E	$0\ m$
z_E	$10.000\ m$
ϕ	$0\ rad$
θ	$0,0381\ rad$
ψ	$0\ rad$
u	$272,02\ m/s$
v	$0\ m/s$
w	$10,36\ m/s$
p	$0\ rad/s$
q	$0\ rad/s$
r	$0\ rad/s$
C_{D_0}	$0,015$
C_{Y_β}	$-0,6$
$C_{Y_{\delta_r}}$	$0,075$
$C_{Y_{\delta_a}}$	$0,01$
C_{L_0}	0
C_{L_α}	$2,204$
$C_{L_{\delta_e}}$	$0,7$
C_{L_q}	0
C_{l_β}	$-0,05$
$C_{l_{\delta_r}}$	$0,018$
$C_{l_{\delta_a}}$	$-0,3$
C_{l_p}	$-0,25$
C_{l_r}	$0,06$
C_{m_0}	0
C_{m_α}	$-0,17$
$C_{m_{\delta_e}}$	$-0,45$
C_{m_q}	$-0,4$
C_{n_β}	$0,15$
$C_{n_{\delta_r}}$	$-0,085$
$C_{n_{\delta_a}}$	0
C_{n_p}	$0,055$
C_{n_r}	$-0,7$
Entradas de controle	
T	$16,74\ KN$
δ_r	$0\ rad$
δ_a	$0\ rad$
δ_e	$-0.014\ rad$

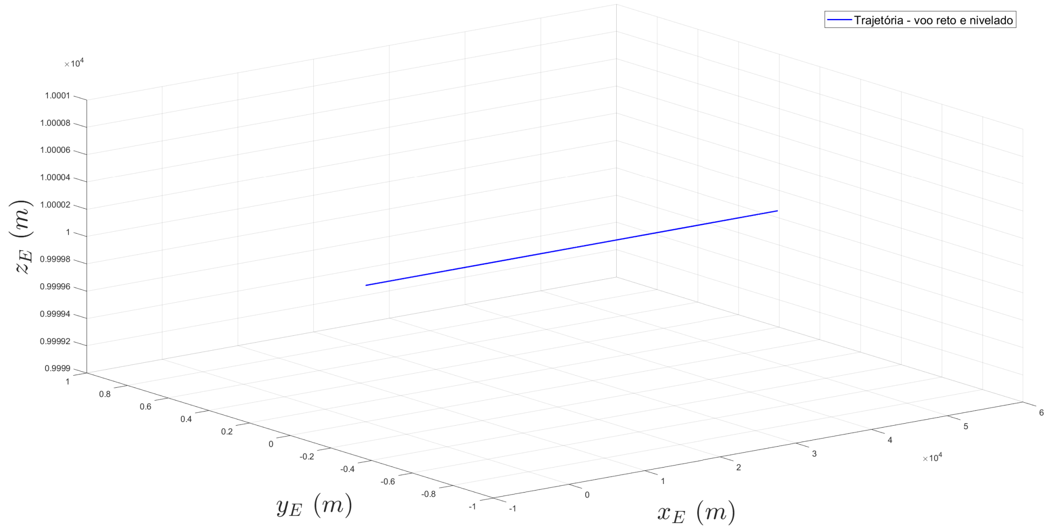


Figura 14 – Trajetória - condição de voo reto e nivelado.

5.3.2 Condição de curva coordenada

Já na condição de curva coordenada, espera-se que a aeronave se incline para o lado da curva, para que a sustentação seja usada tanto para manter a altitude constante quanto para gerar força centrípeta, dessa maneira, deve haver alteração no ângulo de rolagem através do comando de aileron. Para ser coordenada, ou seja, para não haver derrapagem, deve haver comando de leme para alinhar o eixo longitudinal com a direção do movimento. Além disso, para compensar a perda de sustentação, a aeronave deve aumentar seu ângulo de ataque por meio do comando de profundor.

Dado essa condição de voo, para que o Mirage III realize uma curva coordenada a uma altitude de 10.000 m, com velocidade de 360 km/h (100 m/s) e ângulo de rolagem igual a 30°, o modelo Simulink® também foi trimado de forma numérica ($trim\_input$ e $x0$ da Fig. 12 respectivamente (ver Apêndice A)). Após isso, as entradas e as condições iniciais (Tab. 4) foram calculadas e a trajetória resultante plotada de acordo com a Fig. 15, com o mesmo tempo de simulação de 200 s.

Tabela 4 – Condições para curva coordenada.

Parâmetro	**Valor**
Condições iniciais	
x_E	$0\ m$
y_E	$0\ m$
z_E	$10.000\ m$
ϕ	$0,5236\ rad$
θ	$0,2441\ rad$
ψ	$0\ rad$
u	$100\ m/s$
v	$0\ m/s$
w	$28,75\ m/s$
p	$-0,0126\ rad/s$
q	$0,0252\ rad/s$
r	$0,0437\ rad/s$
C_{D_0}	$0,015$
C_{Y_β}	$-0,6$
$C_{Y_{\delta_r}}$	$0,075$
$C_{Y_{\delta_a}}$	$0,01$
C_{L_0}	0
C_{L_α}	$2,204$
$C_{L_{\delta_e}}$	$0,7$
C_{L_q}	0
C_{l_β}	$-0,05$
$C_{l_{\delta_r}}$	$0,018$
$C_{l_{\delta_a}}$	$-0,3$
C_{l_p}	$-0,25$
C_{l_r}	$0,06$
C_{m_0}	0
C_{m_α}	$-0,17$
$C_{m_{\delta_e}}$	$-0,45$
C_{m_q}	$-0,4$
C_{n_β}	$0,15$
$C_{n_{\delta_r}}$	$-0,085$
$C_{n_{\delta_a}}$	0
C_{n_p}	$0,055$
C_{n_r}	$-0,7$
Entradas de controle	
T	$19,75\ KN$
δ_r	$-0,0186\ rad$
δ_a	$-0,0012\ rad$
δ_e	$-0.1042\ rad$

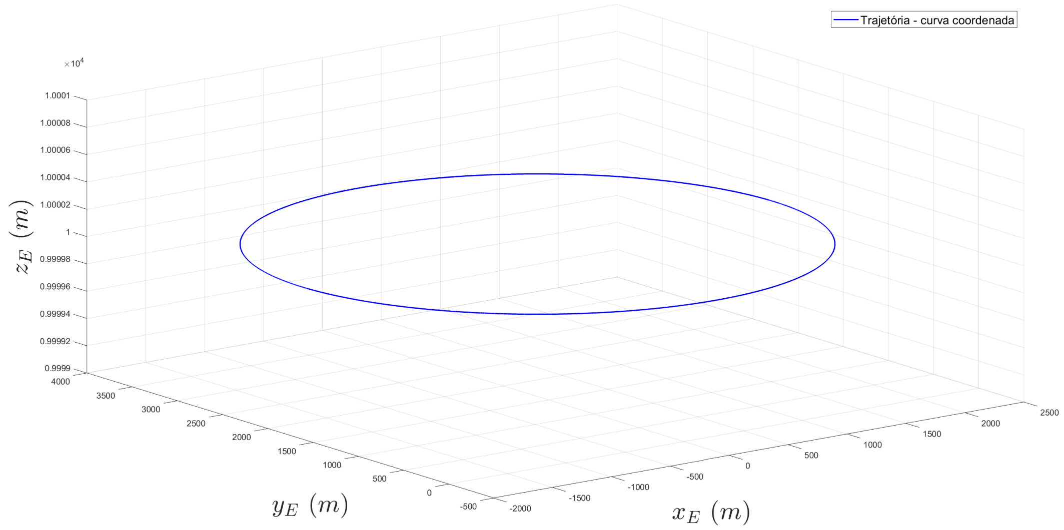


Figura 15 – Trajetória - condição de curva coordenada.

5.4 Inicialização do filtro de Kalman estendido

Como dito na seção 4.2, o FKE requer uma inicialização do estado $\mathbf{x}$ e matriz de covariância $\boldsymbol{P}$. Para obter uma estimação adequada, foi necessário ajustar esses parâmetros, além dos ruídos de modelo $\boldsymbol{Q}$ e de medida $\boldsymbol{R}$. Um ajuste iterativo e manual foi feito, executando o modelo em uma certa configuração inicial e analisando o resultado. Após verificar se a robustez e a convergência do filtro estavam aceitáveis para um parâmetro, o mesmo foi feito para os demais parâmetros e ambas as condições de voo.

Posteriormente, para analisar a convergência da filtragem, o erro de estado será calculado por meio de $\mathbf{x}_{real} - \mathbf{x}_{fke}$. Quanto menor e mais próximo de zero esse valor se aproximar no tempo, maior o indicativo que a inicialização foi escolhida de maneira adequada. Além disso, a covariância do erro $\boldsymbol{P}_{fke}$ será analisada e através dela, os desvios padrões σ_i dos estados serão calculados por meio de: $\sigma_i = \sqrt{\boldsymbol{P}_{fke}(i,i)}$.

Consolidando os tópicos falados acima, o fluxograma geral de funcionamento da simulação pode ser visto na Fig. 16 a seguir, onde mostra a trimagem, a inicialização dos parâmetros dos sensores, do FKE e por fim, a plotagem dos resultados.

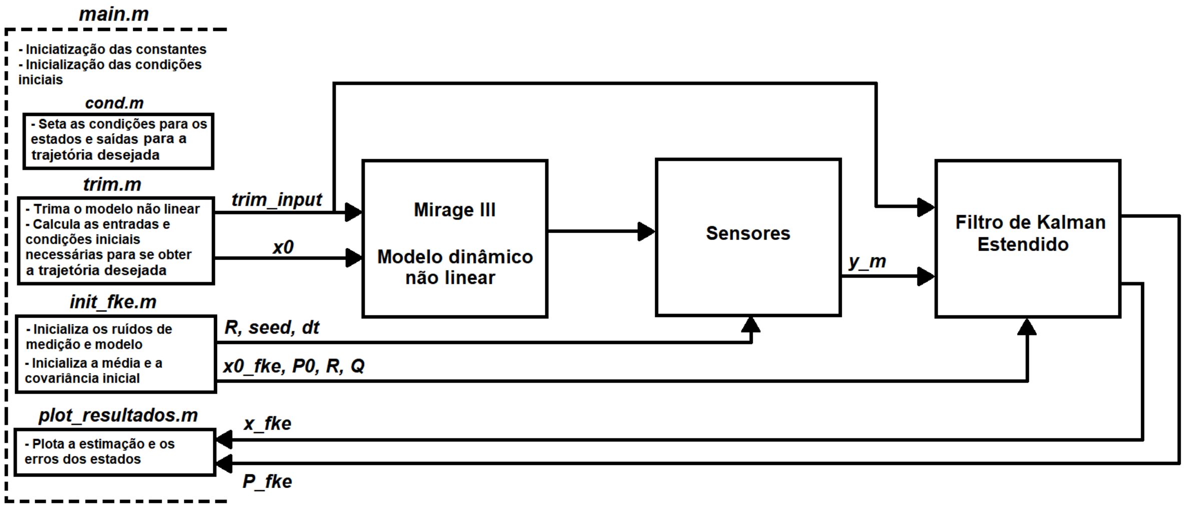


Figura 16 – Fluxograma da solução geral.

Onde, $\boldsymbol{Q}$ e $\boldsymbol{R}$ são dados pelas Eqs. 5.1 e 5.2 abaixo.

$$\begin{array}{lllllllll} \boldsymbol{Q} = diag(0,1 & 0,1 & 0,1 & 0,01 & 0,01 & 0,01 & 0,01 & 0,001 \\ 0,1 & 0,001 & 0,001 & 0,001 & 0,0001 & 0,001 & 100 & 0,00001 \\ 0,0001 & 0,0001 & 0,0001 & 0,0001 & 0,0001 & 0,0001 & 0,0001 & 0,0001 \\ 0,0001 & 0,0001 & 0,0001 & 0,0001 & 0,0001 & 0,0001 & 0,0001 & 0,0001 \\ 0,0001 & 0,0001) & & & & & & \end{array} \tag{5.1}$$

$$\begin{array}{lllllllll} \boldsymbol{R} = diag(0,25 & 0,25 & 0,01 & 0,01 & 0,01 & 0,01 & 0,01 & 0,01 & 0,01 \\ 100 & 100 & 100) & & & & & & \end{array} \tag{5.2}$$

5.4.1 Condição de voo reto e nivelado

Dessa forma, as Eqs. 5.3, 5.4 mostram os valores de inicialização do filtro para a condição de voo reto e nivelado.

$$\begin{array}{lllllllll} \mathbf{x}_0 = [0 & 0 & 10.000 & 0 & 0,0381 & 0 & 272,02 & 0 & 10,36 \\ 0 & 0 & 0 & 0,015 & -0,6 & 0,075 & 0,01 & 0 & 2,204 \\ 0,7 & 0 & -0,05 & 0,018 & -0,3 & -0,25 & 0,06 & 0 & -0,17 \\ -0,45 & -0,4 & 0,15 & -0,085 & 0 & 0,055 & -0,7]^T & & \end{array} \tag{5.3}$$

$$\begin{array}{lllllllll} \boldsymbol{P}_0 = diag(5 & 10 & 10 & 1 & 10 & 10 & 10 & 1 & 10 \\ 0,01 & 0,01 & 0,01 & 0,01 & 0,1 & 0,1 & 0,01 & 0,01 & 0,01 \\ 0,01 & 0,01 & 0,01 & 0,01 & 0,01 & 0,01 & 0,01 & 0,01 & 0,01 \\ 0,01 & 0,01 & 0,01 & 0,01 & 0,01 & 0,01 & 1) & & \end{array} \tag{5.4}$$

5.4.2 Condição de curva coordenada

Da mesma forma, as Eqs. 5.5, 5.6 mostram os valores de inicialização do filtro para a condição de curva coordenada.

$$
\begin{array}{ccccccccc}
\mathbf{x}_0 = [0 & 0 & 10.000 & 0,5236 & 0,2441 & 0 & 100 & 0 & 28,75 \\
-0,0126 & 0,0252 & 0,0437 & 0,015 & -0,6 & 0,075 & 0,01 & 0 & 2,204 \\
0,7 & 0 & -0,05 & 0,018 & -0,3 & -0,25 & 0,06 & 0 & -0,17 \\
-0,45 & -0,4 & 0,15 & -0,085 & 0 & 0,055 & -0,7]^T & &
\end{array}
\tag{5.5}
$$

$$
\begin{array}{ccccccccc}
\boldsymbol{P}_0 = diag(10 & 7,5 & 7,5 & 0,5 & 1 & 1 & 500 & 500 & 500 \\
0,01 & 0,01 & 0,01 & 10 & 0,001 & 0,1 & 0,5 & 10 & 0,0001 \\
100 & 0,001 & 100 & 50 & 50 & 0,0001 & 0,001 & 0,001 & 0,01 \\
100 & 0,001 & 100 & 100 & 5 & 0,01 & 0,01) & &
\end{array}
\tag{5.6}
$$

6 Resultados e Discussões

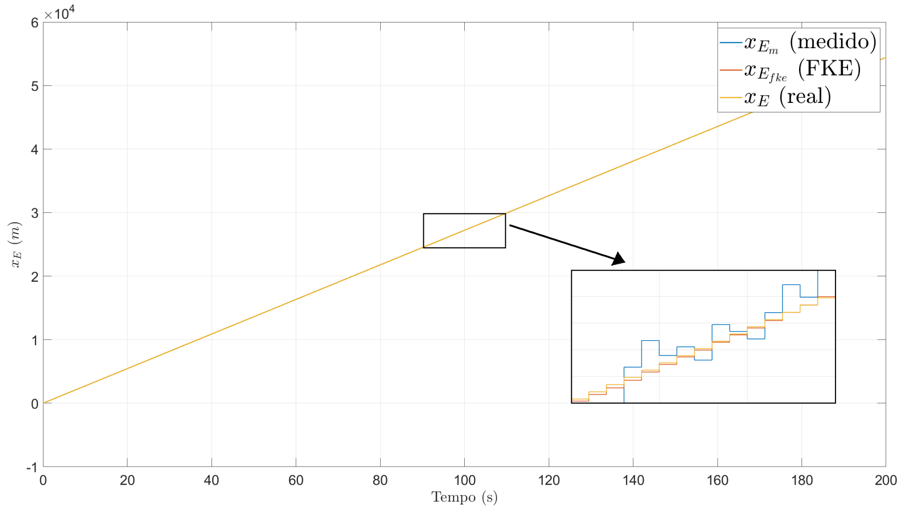
Neste capítulo serão mostrados os resultados da estimação do FKE para o caça Mirage III em voo reto e nivelado e curva coordenada. A estimação dos 34 estados serão comparadas com o valor real (gerado pelo modelo dinâmico não linear) e com o valor medido (gerado pelos sensores). Após isso, o erro absoluto do estado será apresentado juntamente com um intervalo de confiança de $3\sigma_i$ ($99,7$ % de probabilidade da função de densidade de probabilidade), a fim de analisar a estabilização e convergência do filtro.

6.1 Estimação e erros dos estados

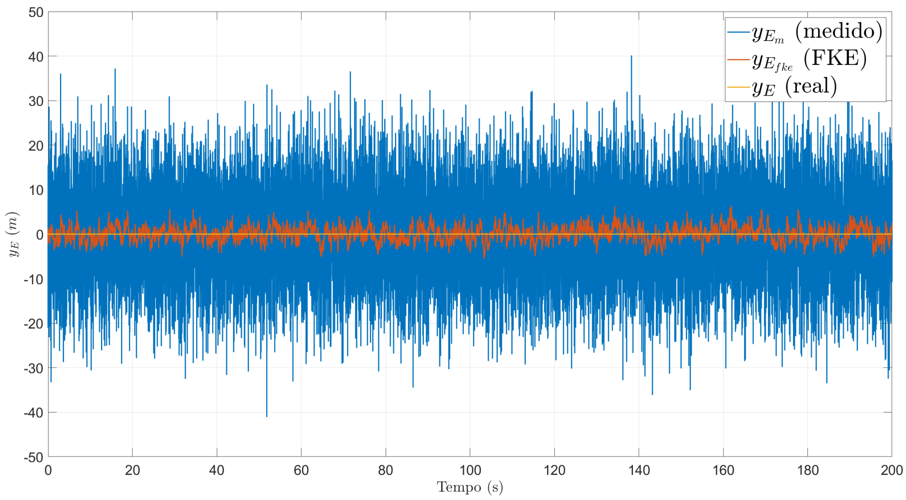
6.1.1 Condição de voo reto e nivelado

Na Fig. 17 estão as estimações das posições lineares x_E, y_E e z_E. Percebe-se que para esses parâmetros, a sintonização e inicialização do filtro foram satisfatórias, pois além de estimar os estados, filtrou de forma considerável a medida ruidosa proveniente do GPS. Outra comprovação da eficácia do filtro para esses parâmetros, está na Fig. 18, que mostra o erro entre os limites do intervalo de confiança estabelecido.

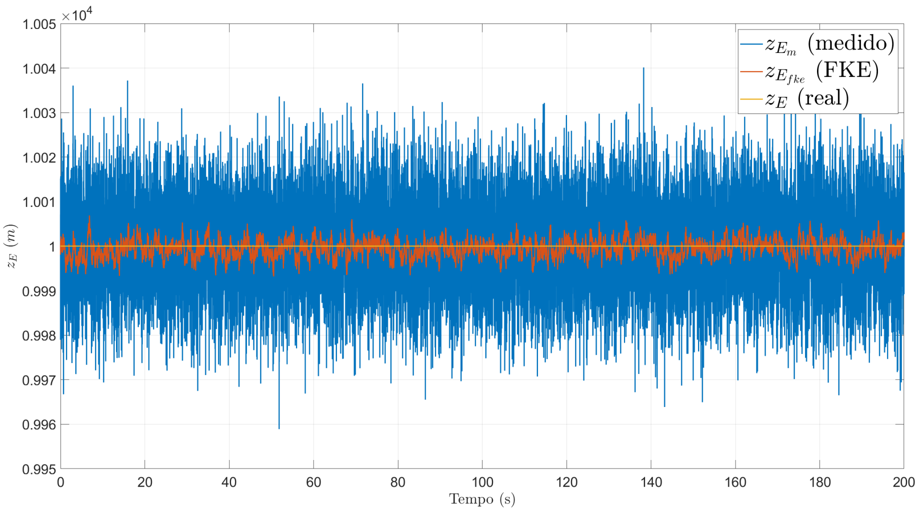
De forma semelhante, também ocorre para ϕ, θ e ψ nas Fig. 19 e 20, onde o FKE atingiu um resultado aceitável para estimar as posições angulares. Apesar de não conseguir filtrar tanto as medidas do giroscópio, cumpriu seu papel de estimação e estabilizou em torno do valor real.



(a) Posição linear x_E.

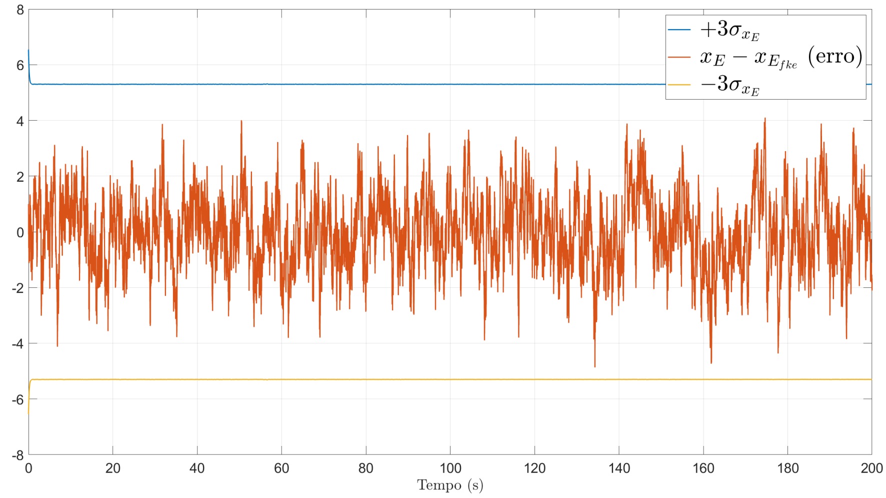


(b) Posição linear y_E.

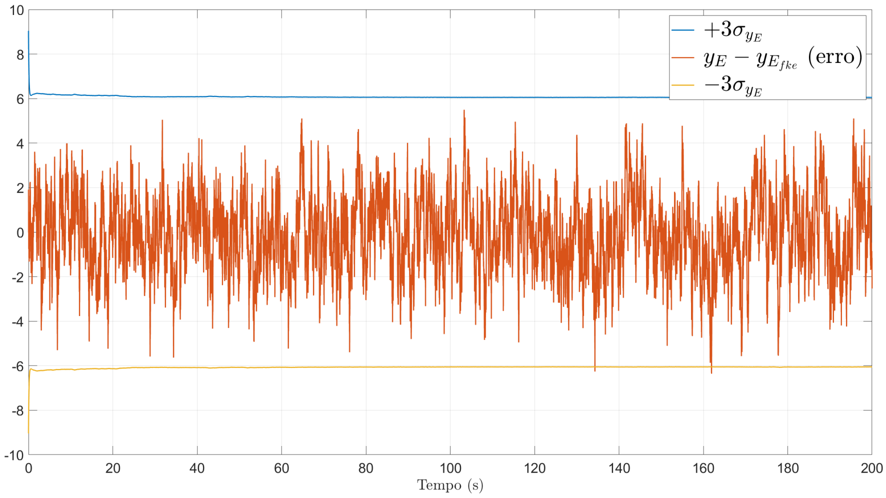


(c) Posição linear z_E.

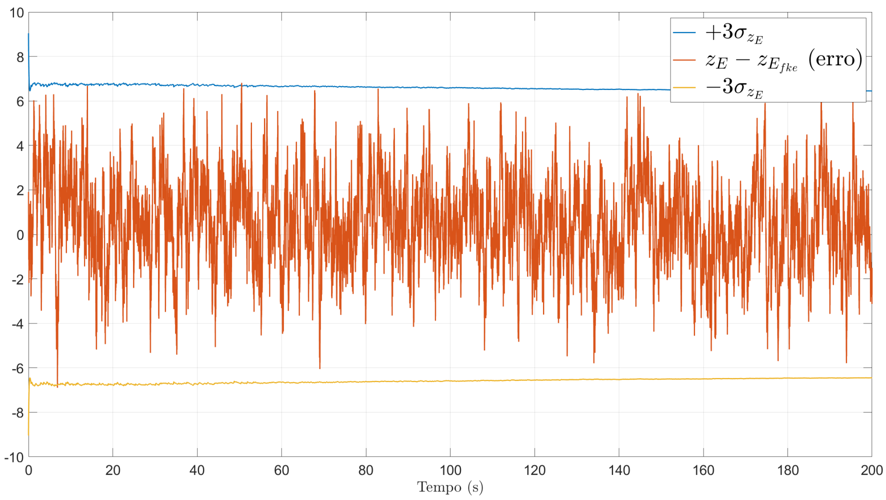
Figura 17 – Posições lineares medidas, estimadas e reais (reto e nivelado).



(a) Posição linear x_E.

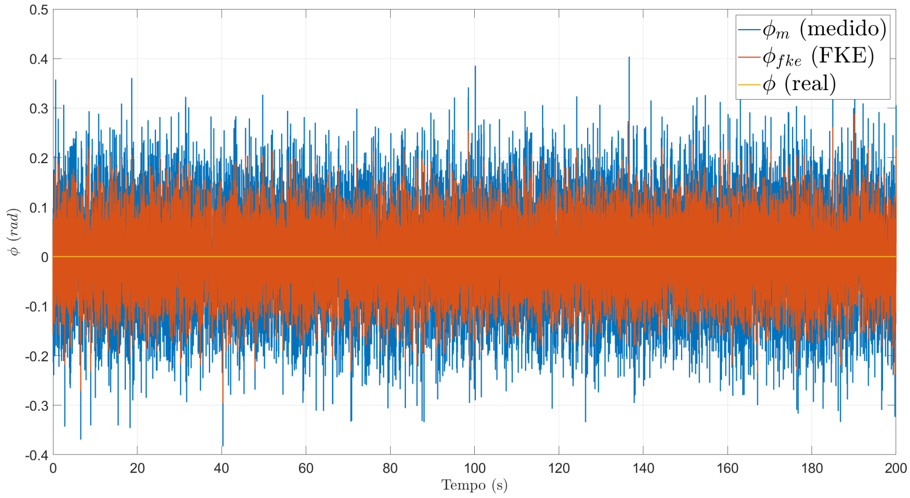


(b) Posição linear y_E.

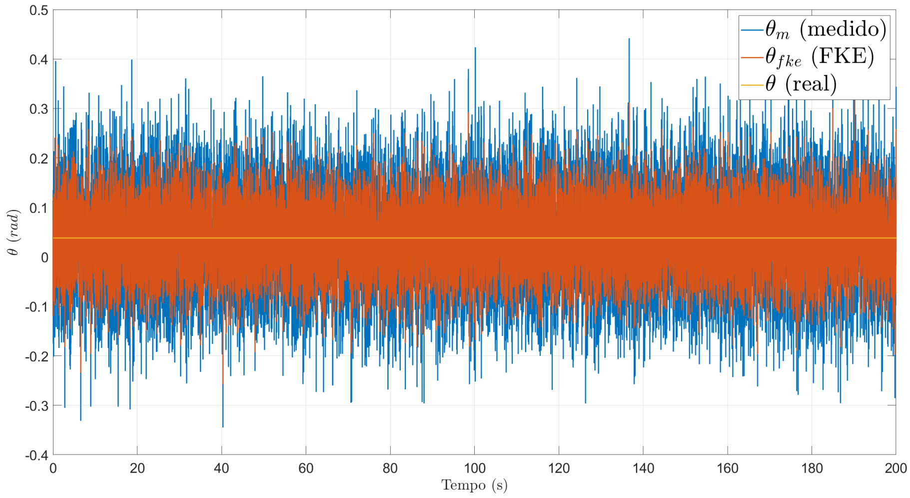


(c) Posição linear z_E.

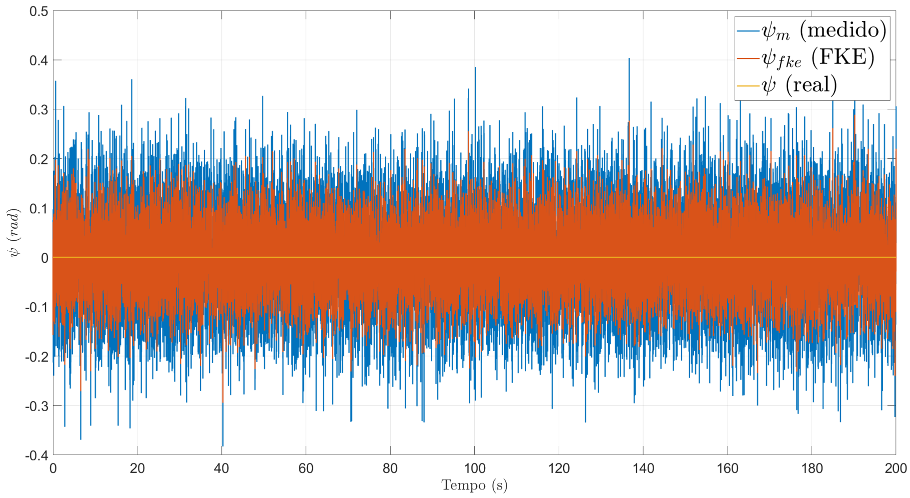
Figura 18 – Erros e intervalos de confiança para as posições lineares (reto e nivelado).



(a) Posição angular ϕ.

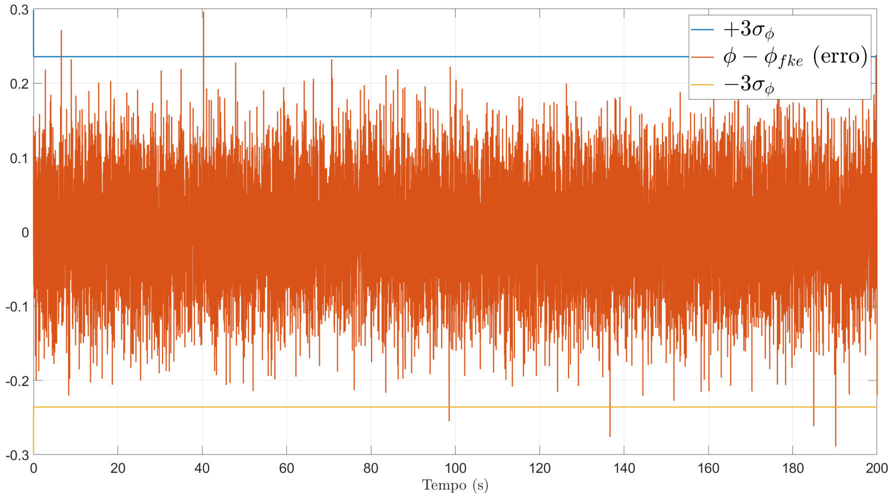


(b) Posição linear θ.

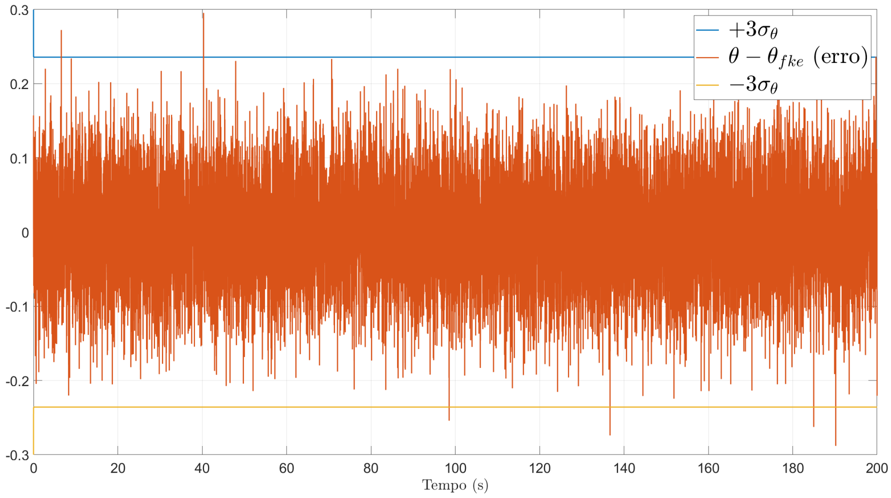


(c) Posição linear ψ.

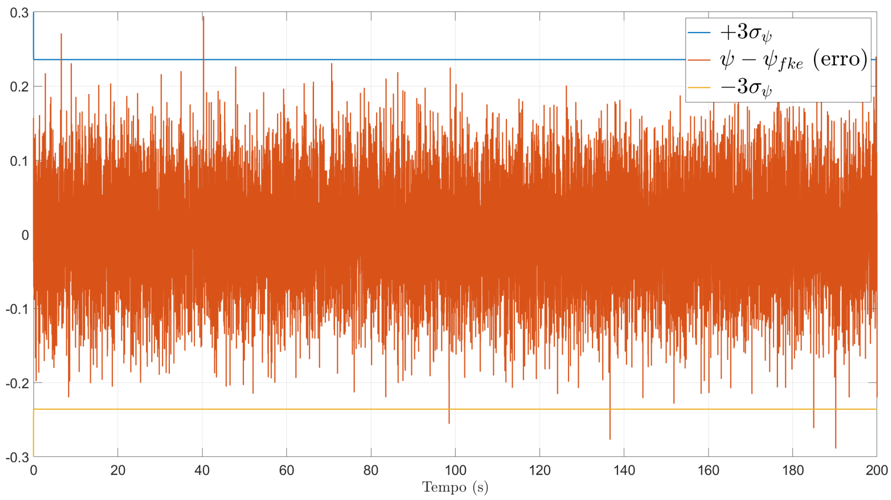
Figura 19 – Posições angulares medidas, estimadas e reais (reto e nivelado).



(a) Posição angular ϕ.



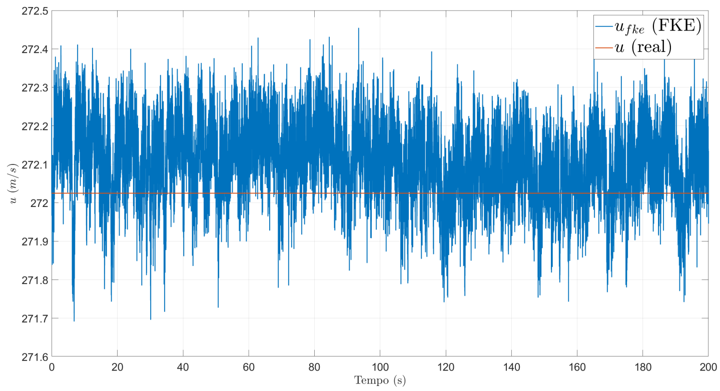
(b) Posição linear θ.



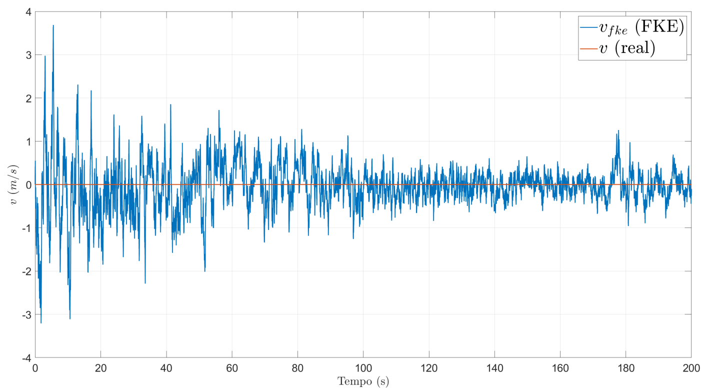
(c) Posição linear ψ.

Figura 20 – Erros e intervalos de confiança para as posições angulares (reto e nivelado).

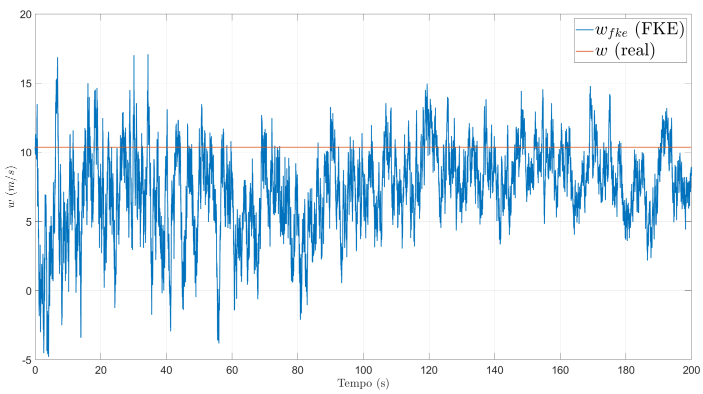
Para as velocidades lineares u e w da Fig. 21, o FKE sofreu um pouco mais de dificuldade para estabilizar, contudo, todas velocidades as lineares ficaram dentro do intervalo de confiança (Fig. 22). Vale lembrar que as velocidades lineares não são medidas diretamente, o que valida a sintonização do filtro para esses parâmetros. Já para as velocidades angulares p, q e r, o FKE se comportou melhor (Fig. 23), sendo capaz de filtrar de forma considerável o ruído do giroscópio e estabilizar dentro dos limites de 3σ (Fig. 24).



(a) Velocidade linear u.

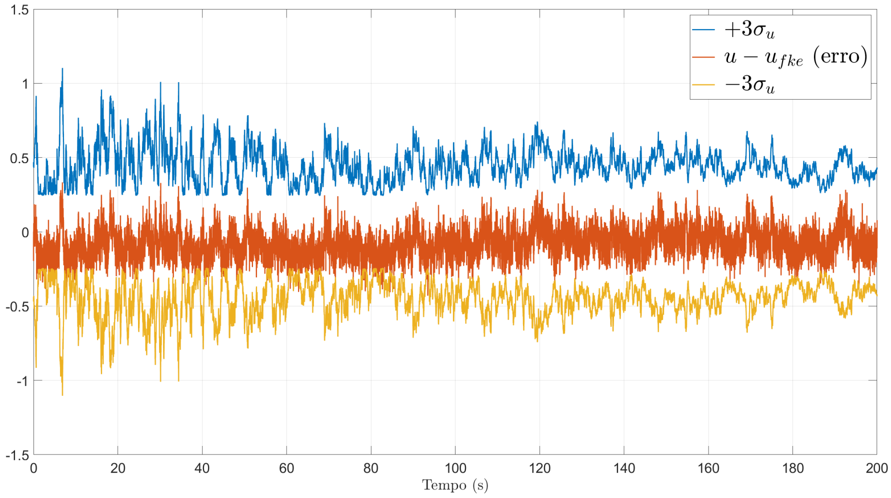


(b) Velocidade linear v.

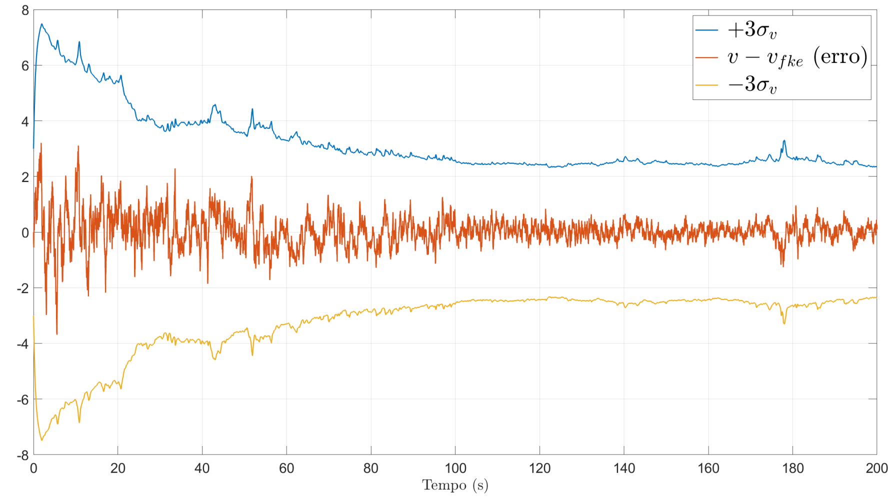


(c) Velocidade linear w.

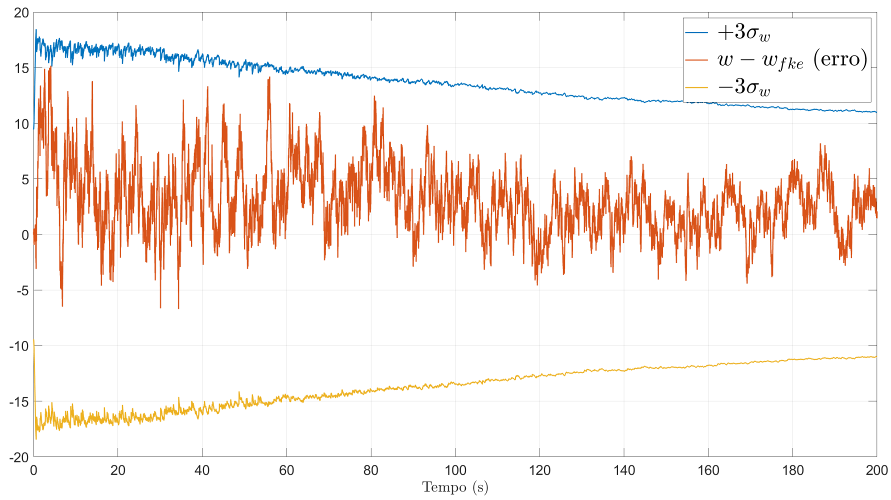
Figura 21 – Velocidades lineares estimadas e reais (reto e nivelado).



(a) Velocidade linear u.

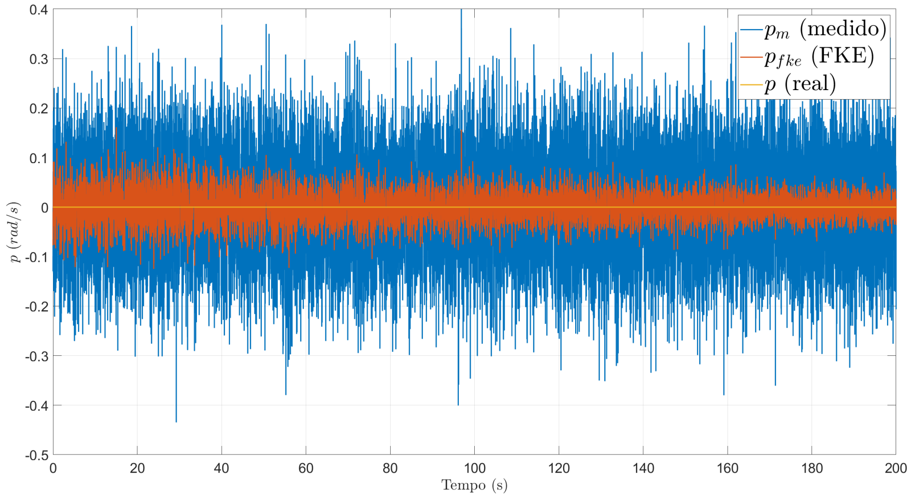


(b) Velocidade linear v.

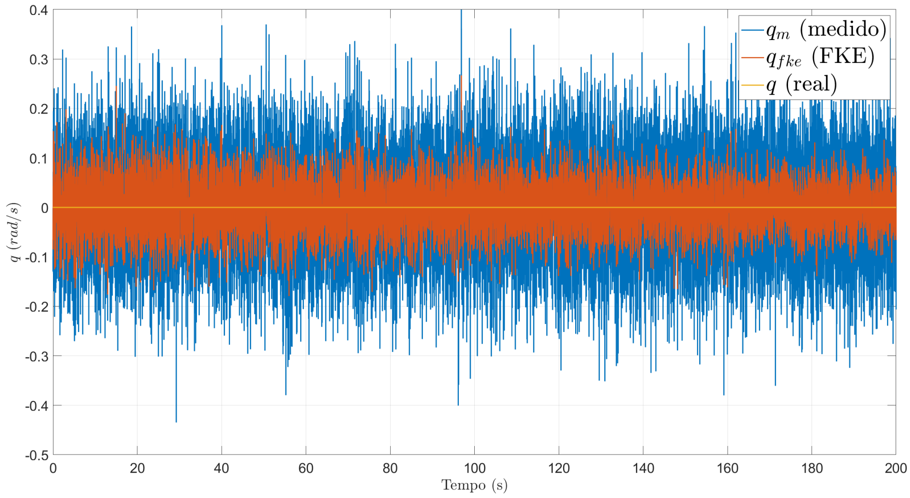


(c) Velocidade linear w.

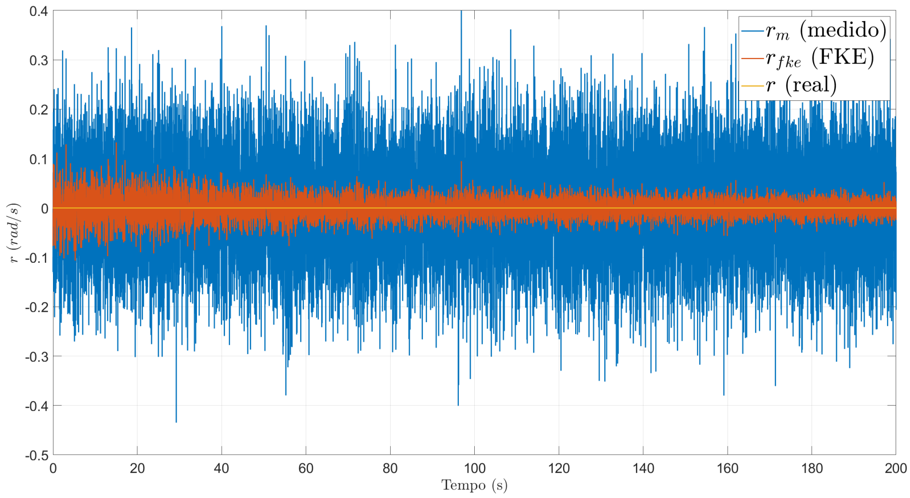
Figura 22 – Erros e intervalos de confiança para as velocidades lineares (reto e nivelado).



(a) Velocidade angular p.

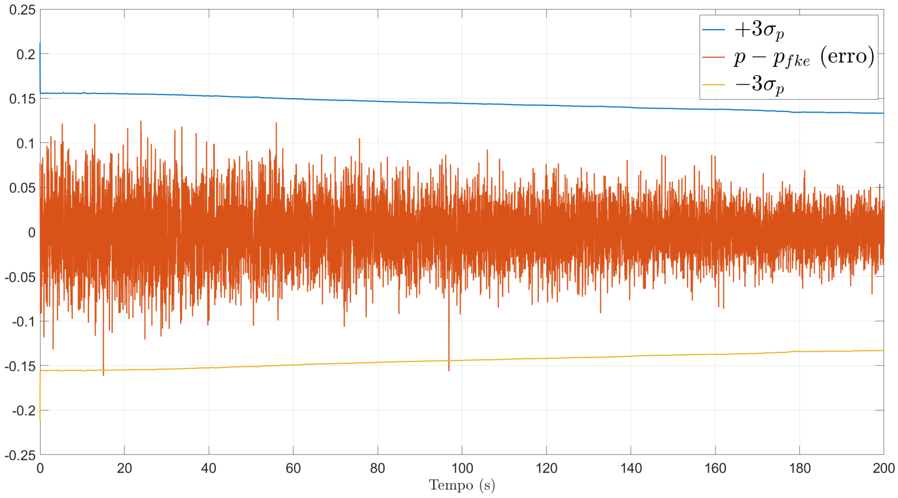


(b) Velocidade angular q.

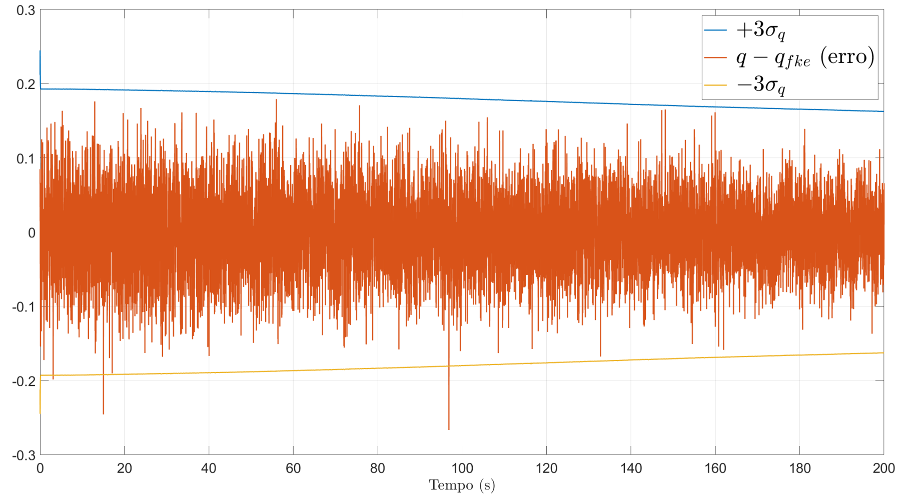


(c) Velocidade angular r.

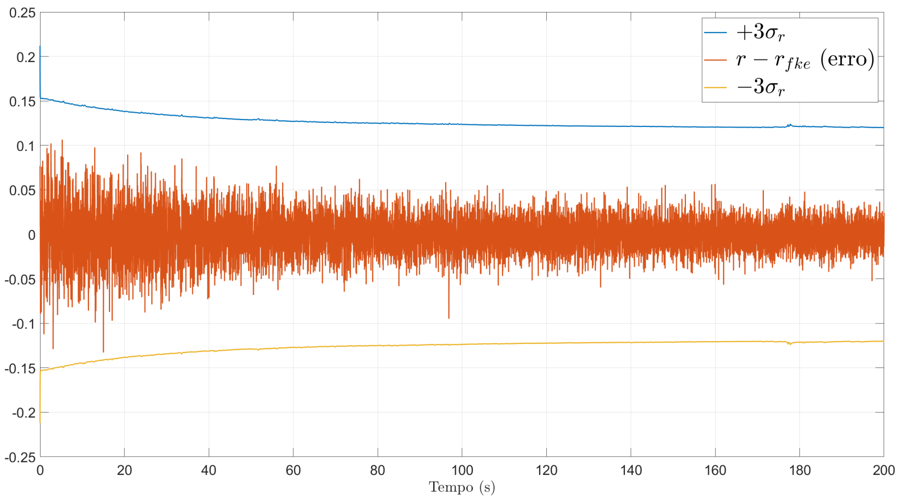
Figura 23 – Velocidades angulares medidas, estimadas e reais (reto e nivelado).



(a) Velocidade angular p.



(b) Velocidade angular q.



(c) Velocidade angular r.

Figura 24 – Erros e intervalos de confiança para as velocidades angulares (reto e nivelado).

Iniciando com a análise dos parâmetros aerodinâmicos, o FKE fez um bom trabalho em estimar C_{D_0} (Fig. 25), com o erro absoluto dentro do intervalo de confiança, como pode ser visto na Fig. 26.

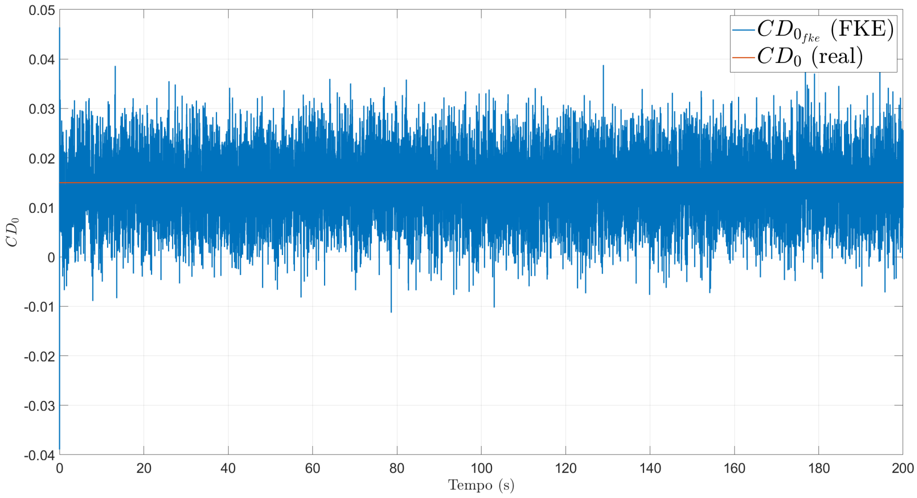


Figura 25 – C_{D_0} estimado e real (reto e nivelado).

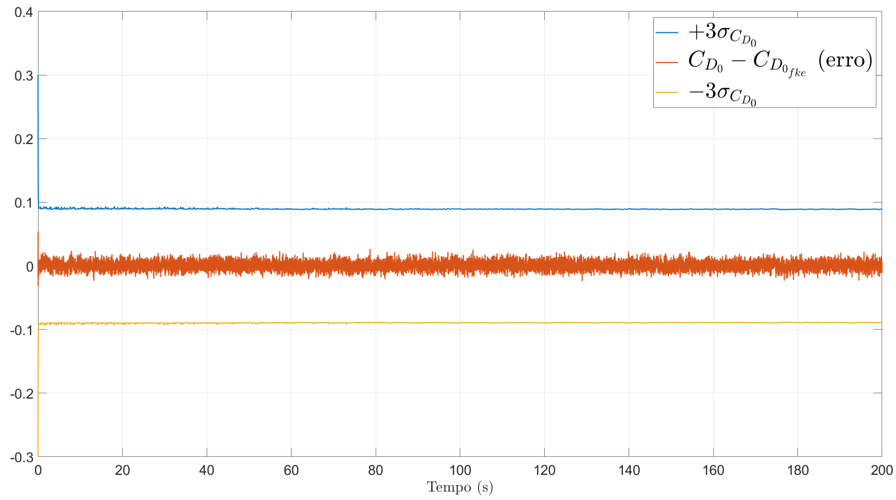
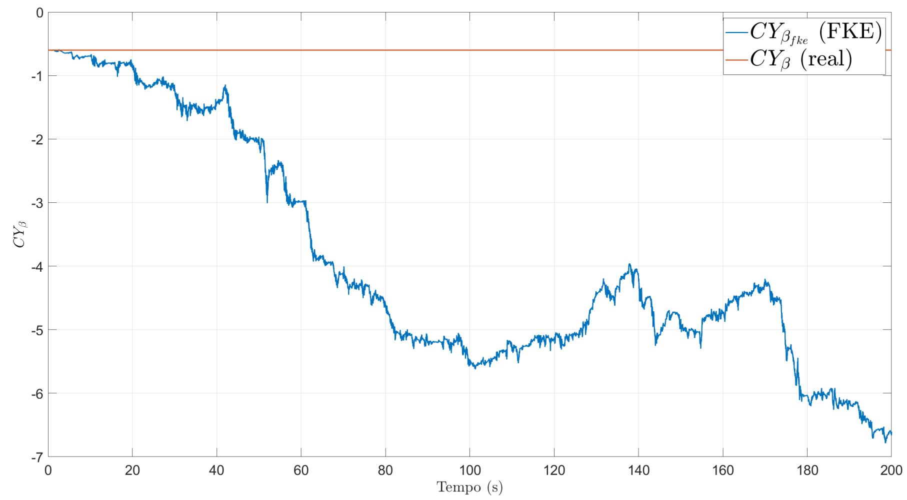
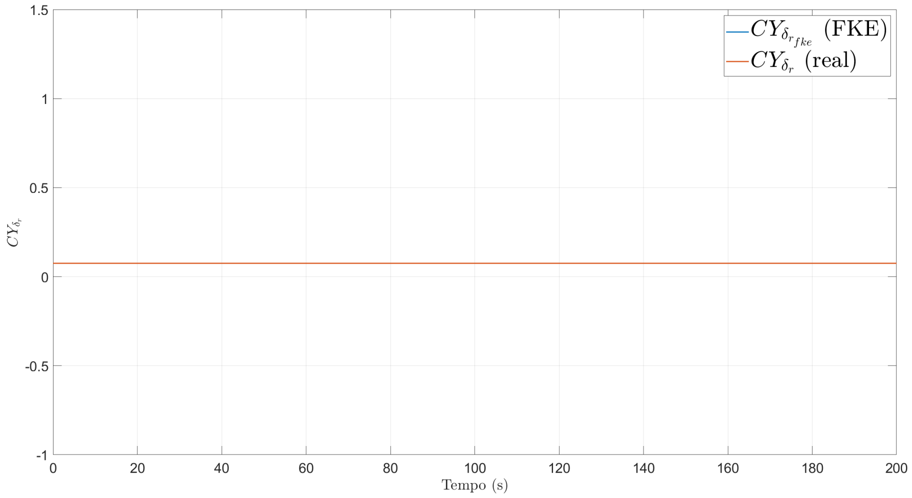


Figura 26 – Erro e intervalo de confiança para C_{D_0} (reto e nivelado).

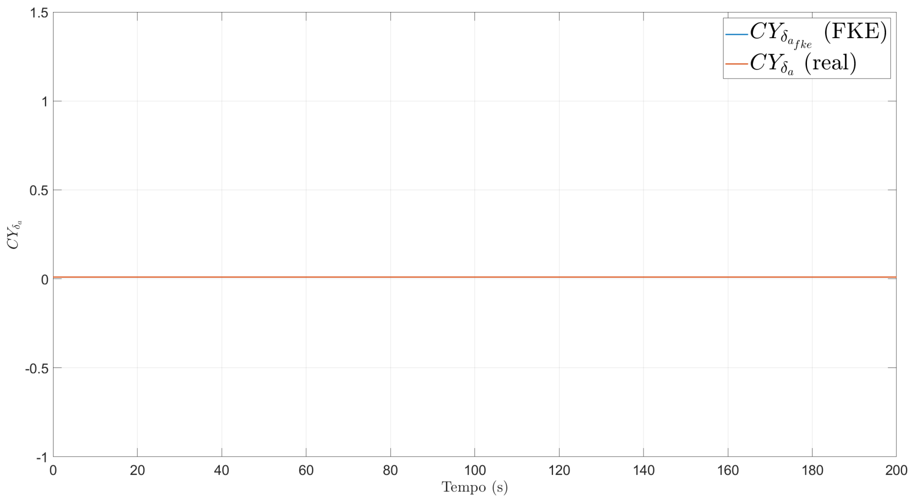
Para os parâmetros de força lateral, somente C_{Y_β} não apresentou um resultado esperado (Figs. 27 e 28), tendo seu valor diminuindo com o tempo e não estabilizando a depender da inicialização. Diversas sintonias foram feitas para se obter uma curva que se aproximasse do valor real, porém, nenhuma foi encontrada.



(a) Devido à derrapagem β.

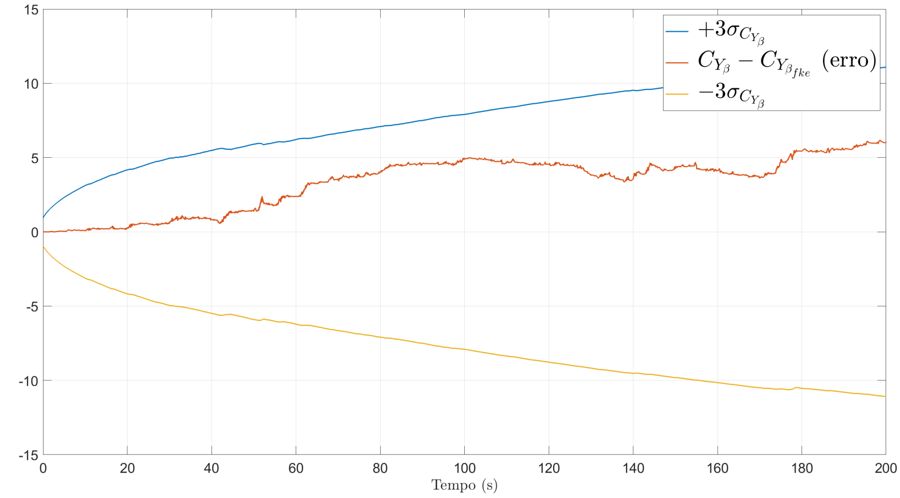


(b) Devido à deflexão angular do leme δ_r.

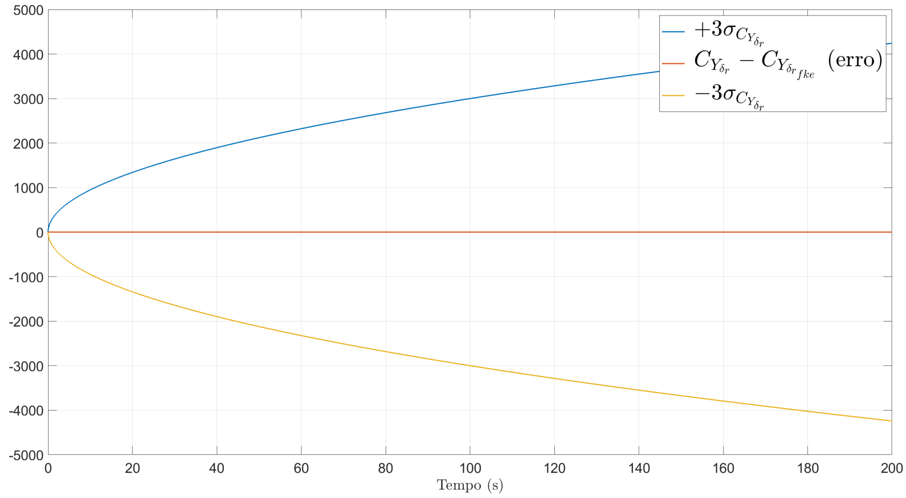


(c) Devido à deflexão angular dos ailerons δ_a.

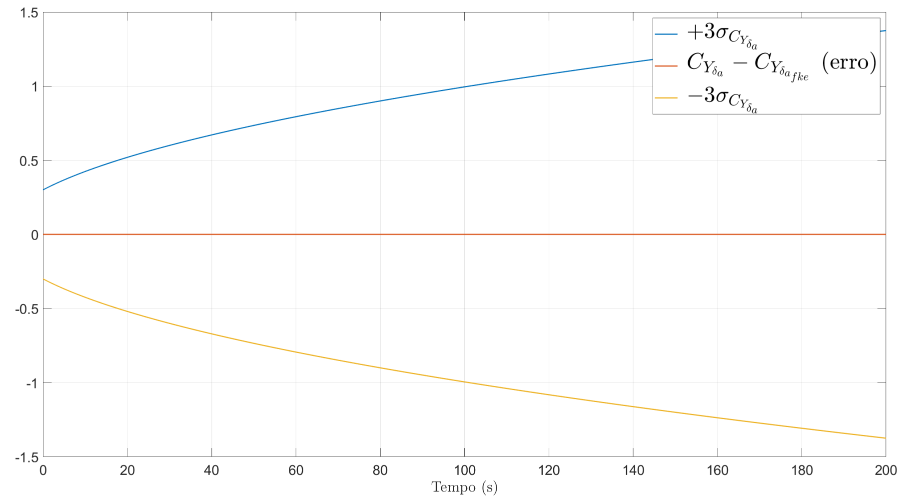
Figura 27 – C_Y estimados e reais (reto e nivelado).



(a) Devido à derrapagem β.



(b) Devido à deflexão angular do leme δ_r.



(c) Devido à deflexão angular dos ailerons δ_a.

Figura 28 – Erros e intervalos de confiança para C_Y (reto e nivelado).

O comportamento do FKE para C_{L_0} foi um pouco semelhante ao do C_{D_0}, porém, na Fig. 29 a curva está com tendências à divergir, somente sendo possível verificar sua convergência com mais passos de simulação. Contudo, dentro do tempo de 200 s simulado, o erro (Fig. 30) ficou dentro dos batentes estabelecidos pelo desvio padrão.

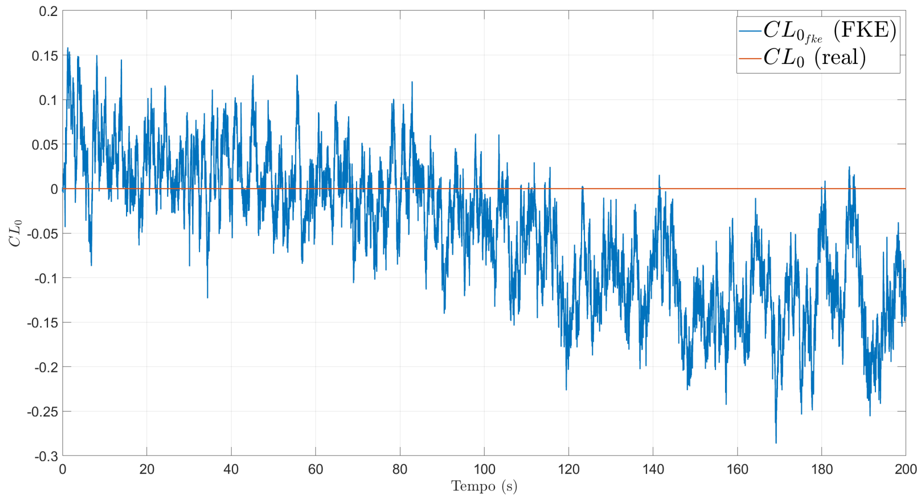


Figura 29 – C_{L_0} estimado e real (reto e nivelado).

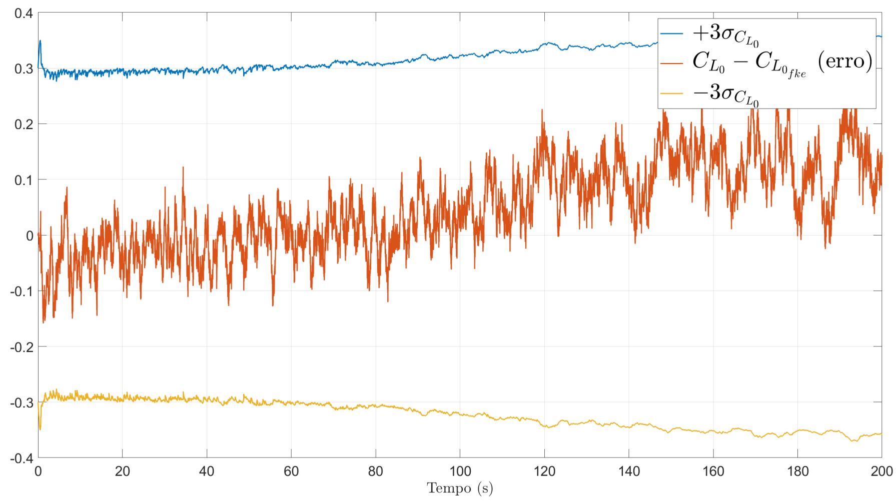
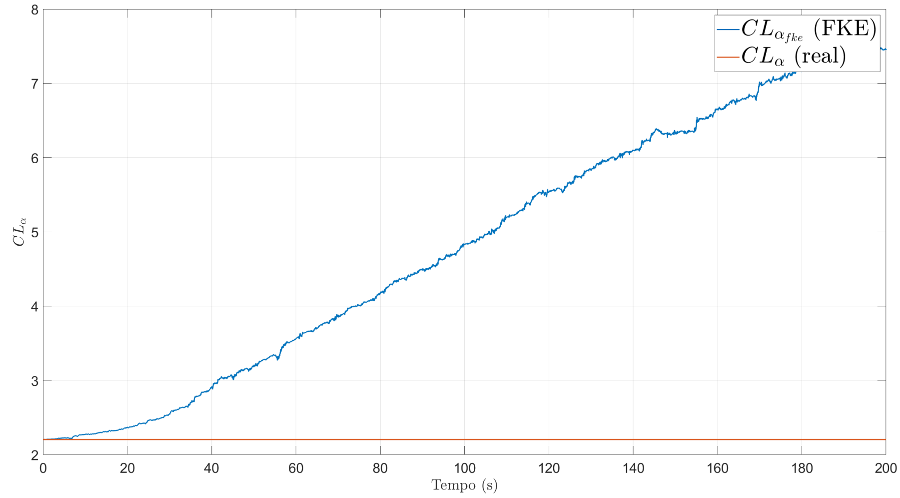
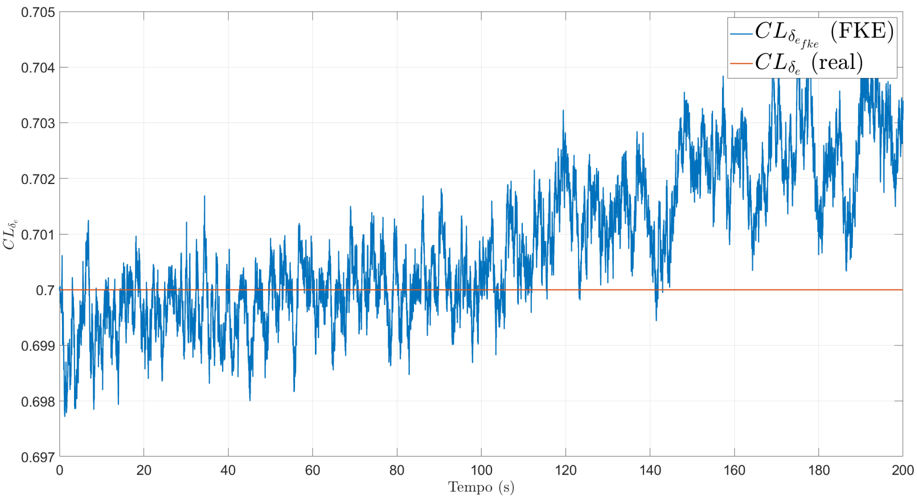


Figura 30 – Erro e intervalo de confiança para C_{L_0} (reto e nivelado).

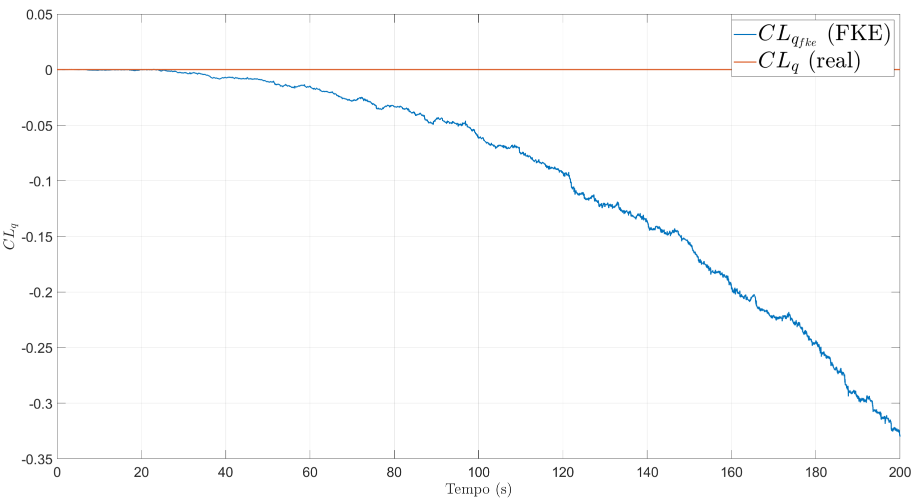
Um comportamento semelhante foi observado para os coeficientes C_{L_α}, $C_{L_{\delta_e}}$ e C_{L_q}, resultando em divergências, com apenas o coeficiente C_{L_α} ficando fora dos limites do intervalo de confiança, como pode ser visto nas Figs. 31 e 32 a seguir.



(a) Devido ao ângulo de ataque α.

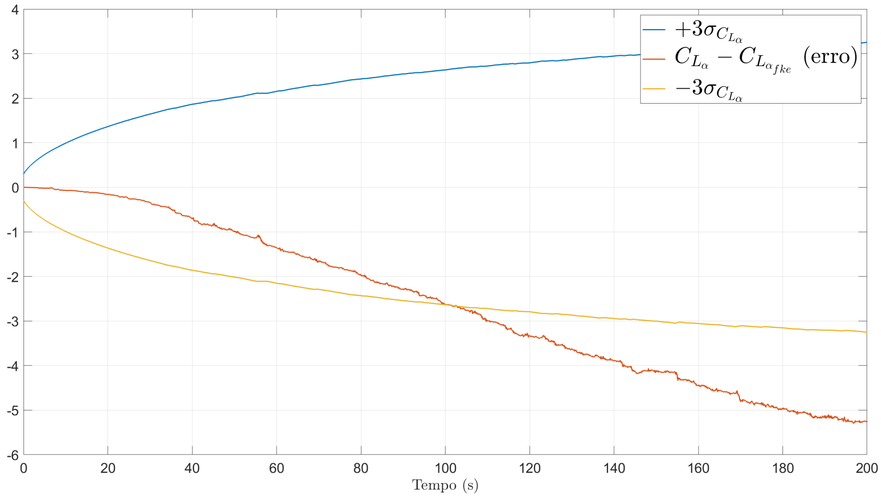


(b) Devido à deflexão angular do profundor δ_e.

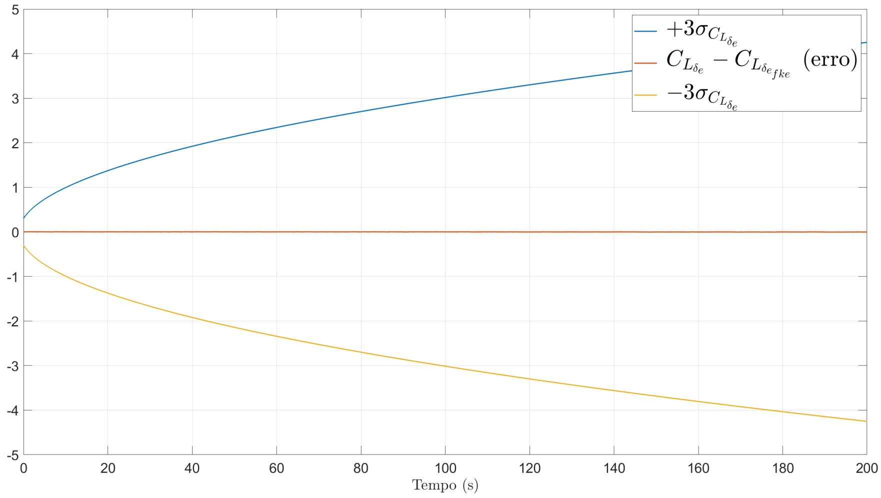


(c) Devido à velocidade de arfagem q.

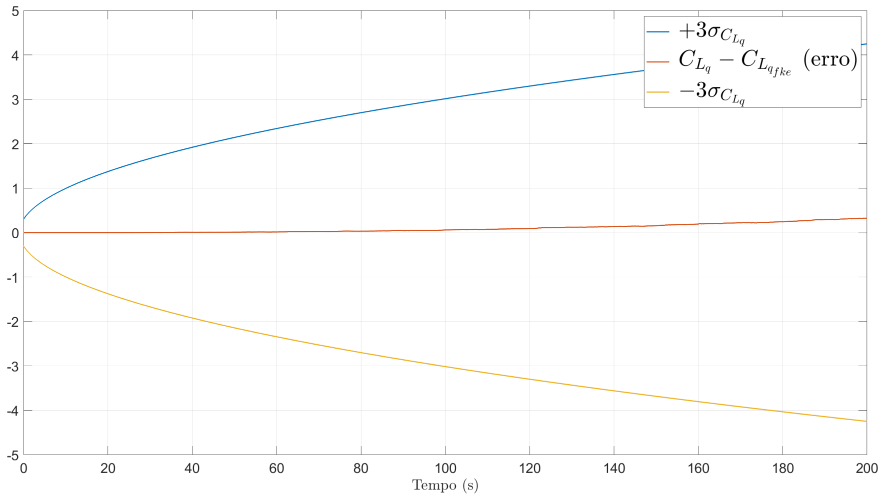
Figura 31 – C_L estimados e reais (reto e nivelado).



(a) Devido ao ângulo de ataque α.



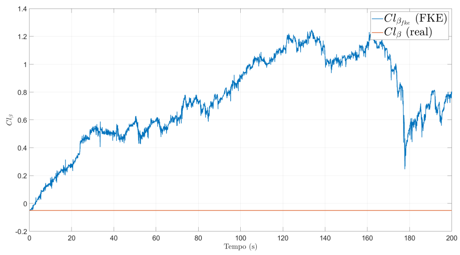
(b) Devido à deflexão angular do profundor δ_e.



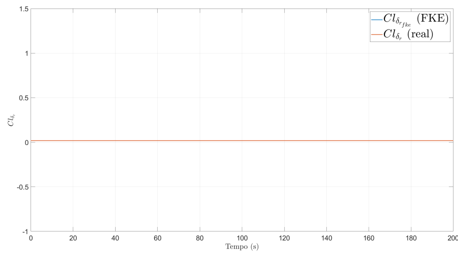
(c) Devido à velocidade de arfagem q.

Figura 32 – Erros e intervalos de confiança para C_L (reto e nivelado).

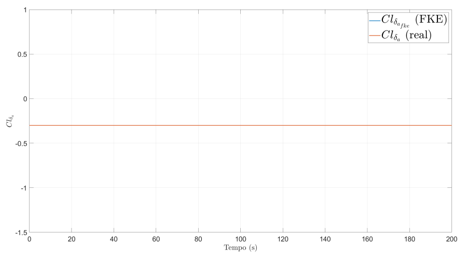
Já para os parâmetros aerodinâmicos de momento, o comportamento foi bem distribuído, com alguns parâmetros sendo perfeitamente estimados e outros saindo dos limites de confiança, como pode ser visto pelas Figs. 33 e 34 abaixo.



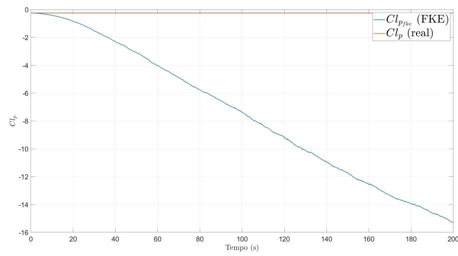
(a) Devido à derrapagem β.



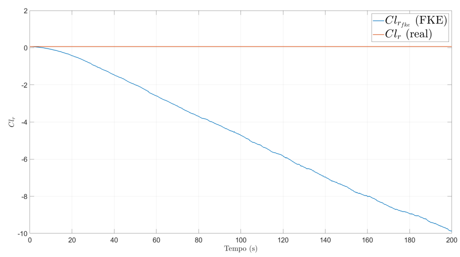
(b) Devido à deflexão angular do leme δ_r.



(c) Devido à deflexão angular dos ailerons δ_a.

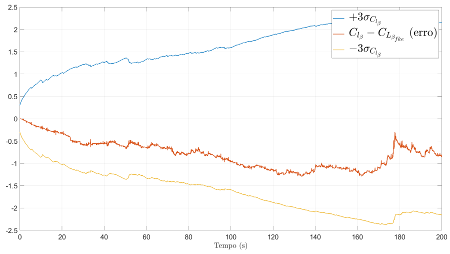


(d) Devido à velocidade de rolagem p.

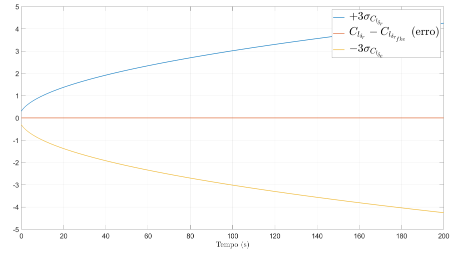


(e) Devido à velocidade de guinada r.

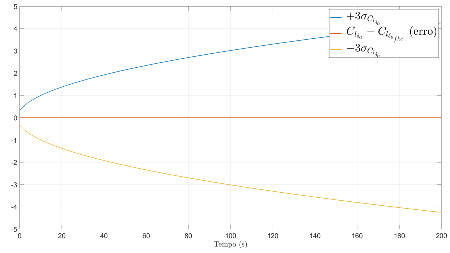
Figura 33 – C_l estimados e reais (reto e nivelado).



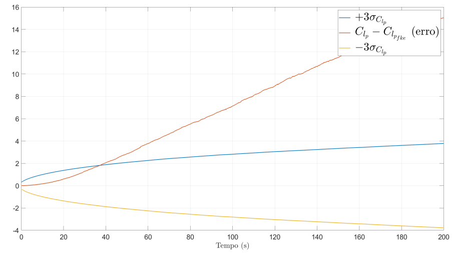
(a) Devido à derrapagem β.



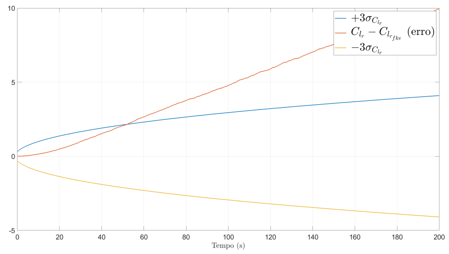
(b) Devido à deflexão angular do leme δ_r.



(c) Devido à deflexão angular dos ailerons δ_a.



(d) Devido à velocidade de rolagem p.



(e) Devido à velocidade de guinada r.

Figura 34 – Erros e intervalos de confiança para C_l (reto e nivelado).

O comportamento da estimação para C_{m_0} foi semelhante ao do C_{L_0}, com a curva da Fig. 35, tendendo a divergir, só sendo possível verificar sua convergência com mais passos de simulação. Contudo, dentro do tempo simulado, o erro ficou dentro dos limites a maior parte do tempo (Fig. 36).

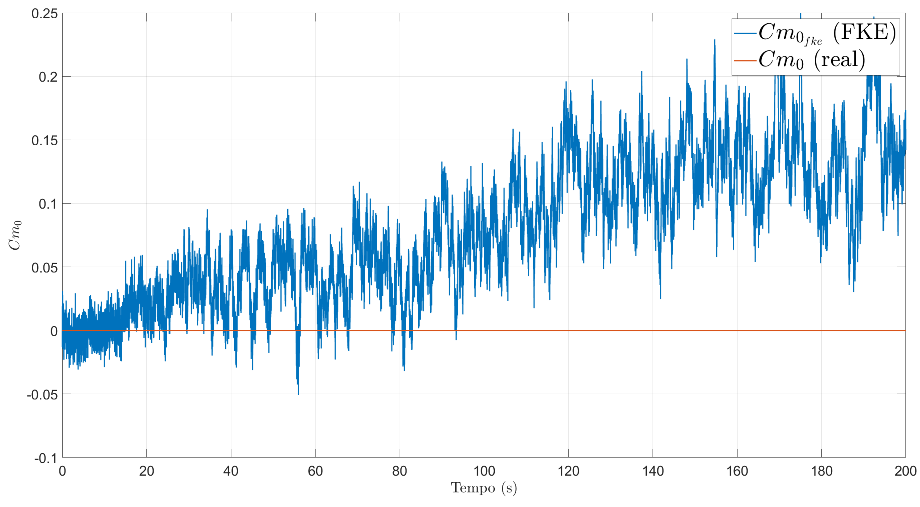


Figura 35 – C_{m_0} estimado e real (reto e nivelado).

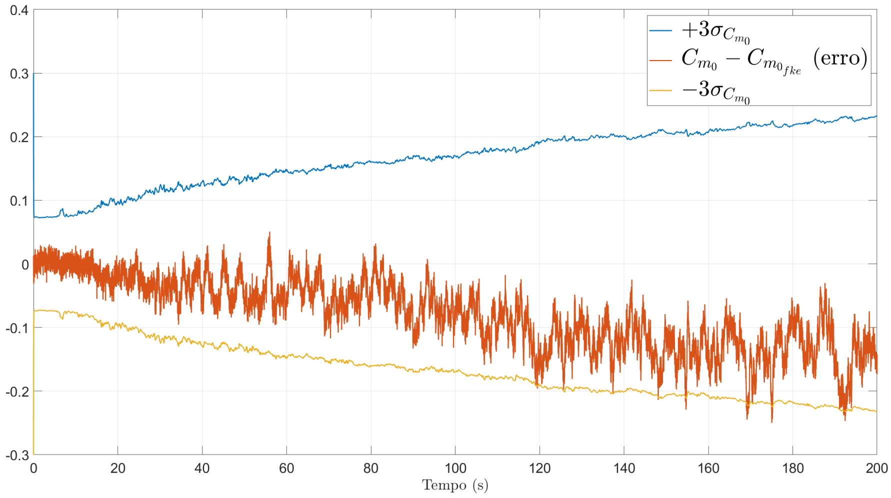
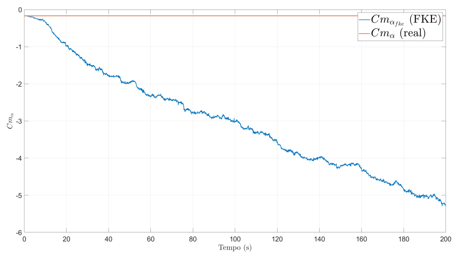
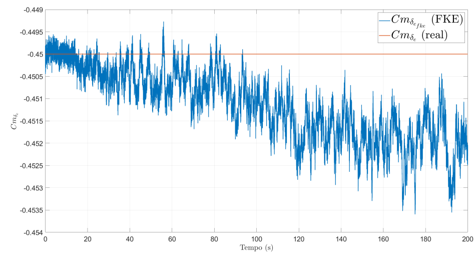


Figura 36 – Erro e intervalo de confiança para C_{m_0} (reto e nivelado).

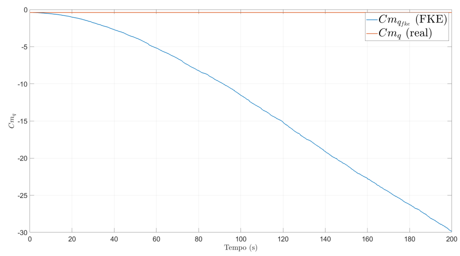
Assim como anteriormente, os parâmetros C_{m_α} e C_{m_q} não convergiram e saíram dos batentes de intervalo de confiança, como mostra as Figs. 37 e 38. Apesar disso, o erro de $C_{m_{\delta_e}}$ se manteve próximo de zero.



(a) Devido ao ângulo de ataque α.

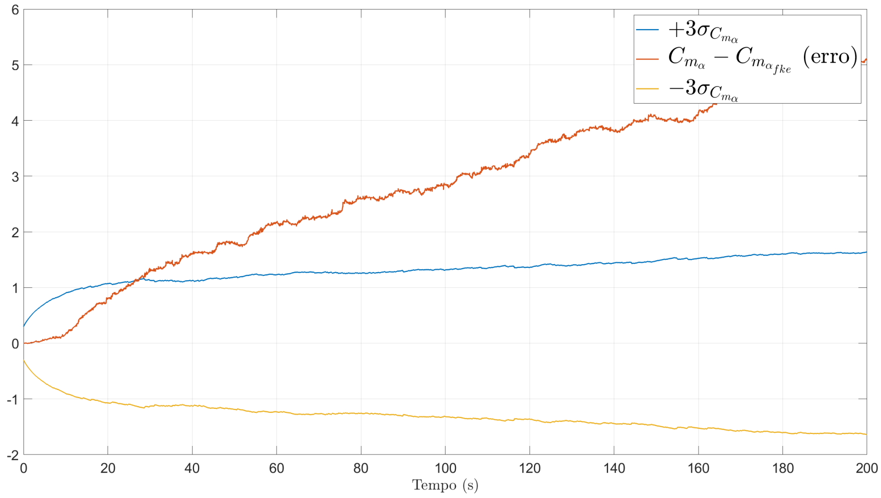


(b) Devido à deflexão angular do profundor δ_e.

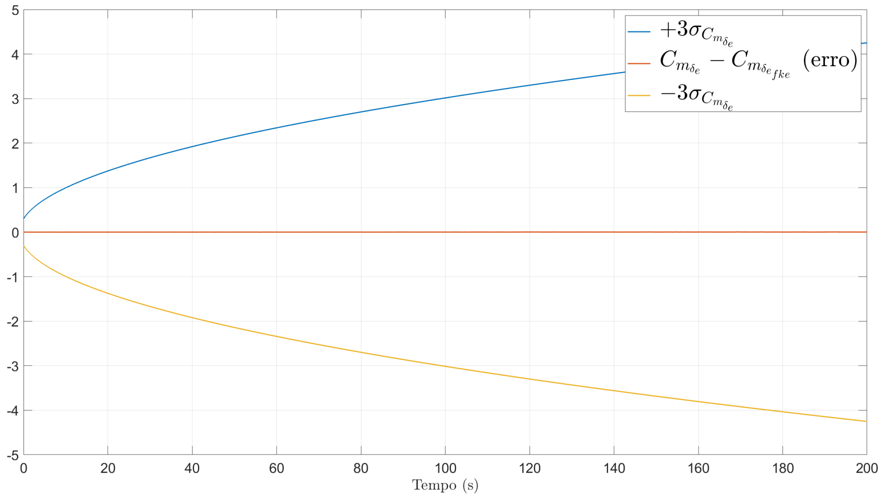


(c) Devido à velocidade de guinada q.

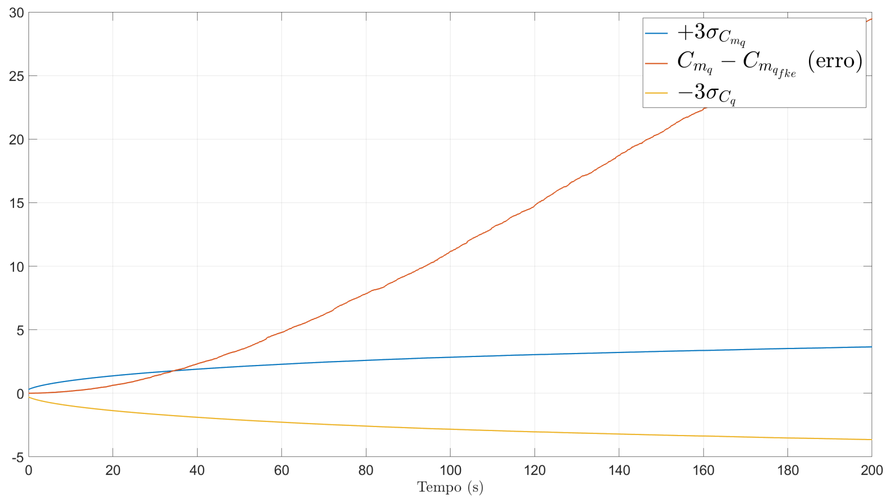
Figura 37 – C_m estimados e reais (reto e nivelado).



(a) Devido ao ângulo de ataque α.



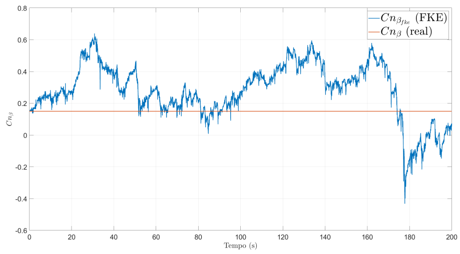
(b) Devido à deflexão angular do profundor δ_e.



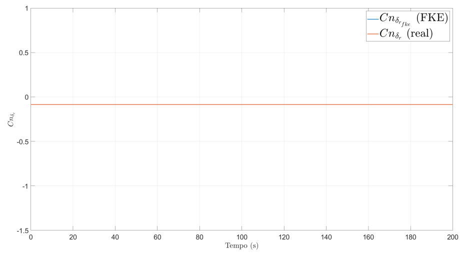
(c) Devido à velocidade de guinada q.

Figura 38 – Erros e intervalos de confiança para C_m (reto e nivelado).

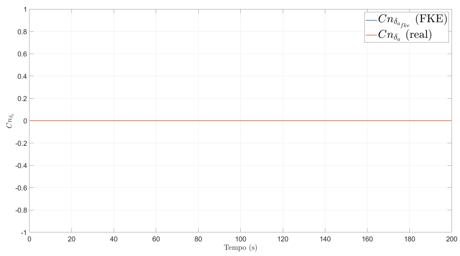
Da mesma maneira que na Fig. 34, os parâmetros de momento de guinada, se comportaram de forma mista e em apenas 2 deles ($C_{n_{\delta_r}}$ e $C_{n_{\delta_a}}$), a estimação apresentou exatamente o valor real. Já para os demais parâmetros, a convergência não ocorreu e novamente os limites de desvio padrão foram superados, como visto nas Figs. 39 e 40.



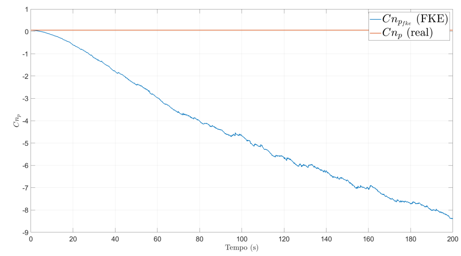
(a) Devido à derrapagem β.



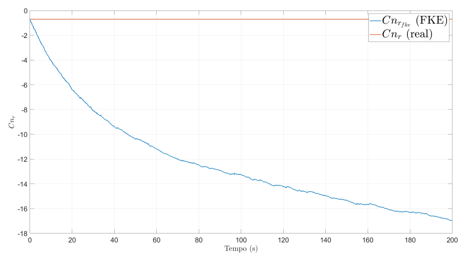
(b) Devido à deflexão angular do leme δ_r.



(c) Devido à deflexão angular dos ailerons δ_a.

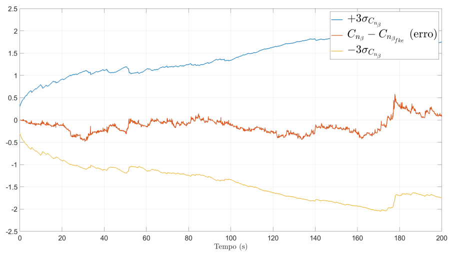


(d) Devido à velocidade de rolagem p.

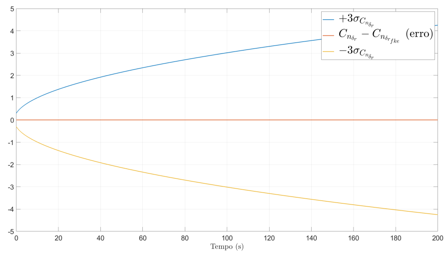


(e) Devido à velocidade de guinada r.

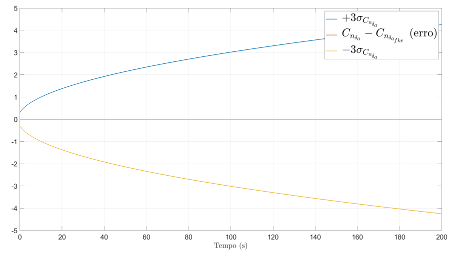
Figura 39 – C_n estimados e reais (reto e nivelado).



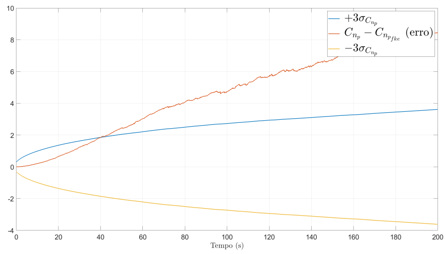
(a) Devido à derrapagem β.



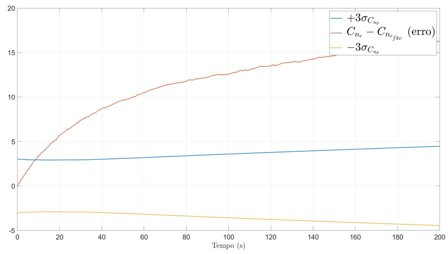
(b) Devido à deflexão angular do leme δ_r.



(c) Devido à deflexão angular dos ailerons δ_a.



(d) Devido à velocidade de rolagem p.

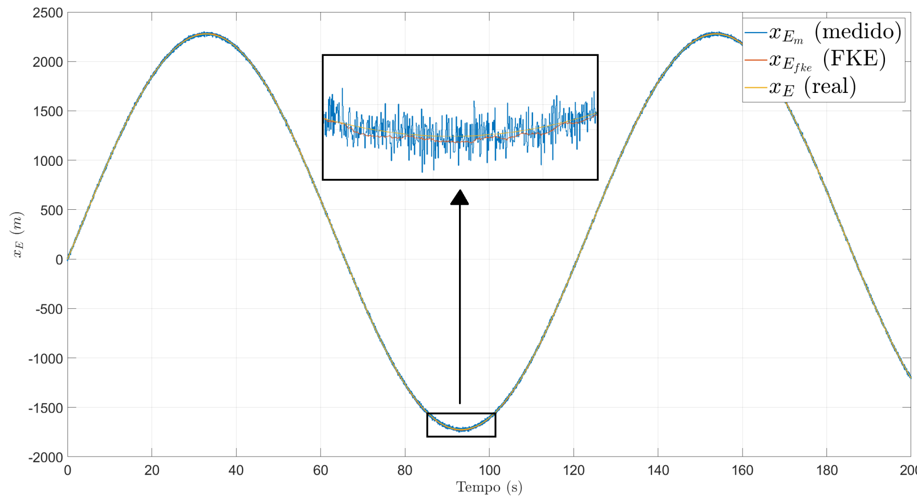


(e) Devido à velocidade de guinada r.

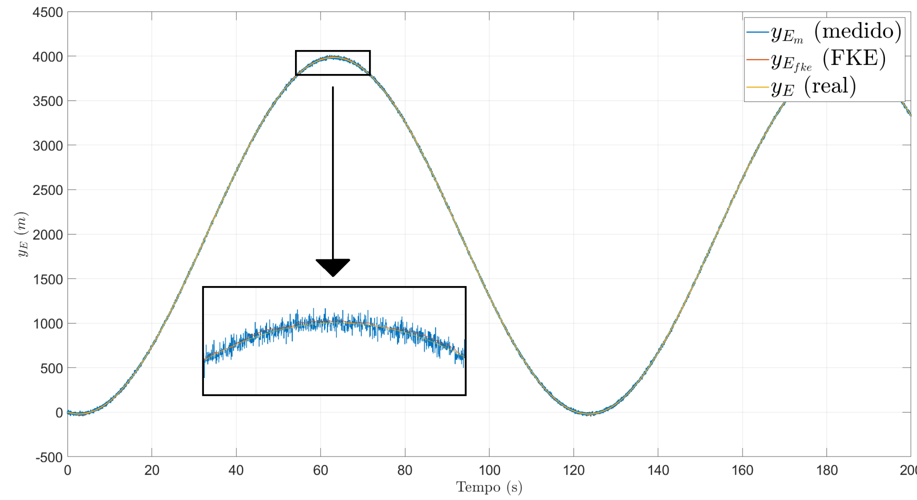
Figura 40 – Erros e intervalos de confiança para C_n (reto e nivelado).

6.1.2 Condição de curva coordenada

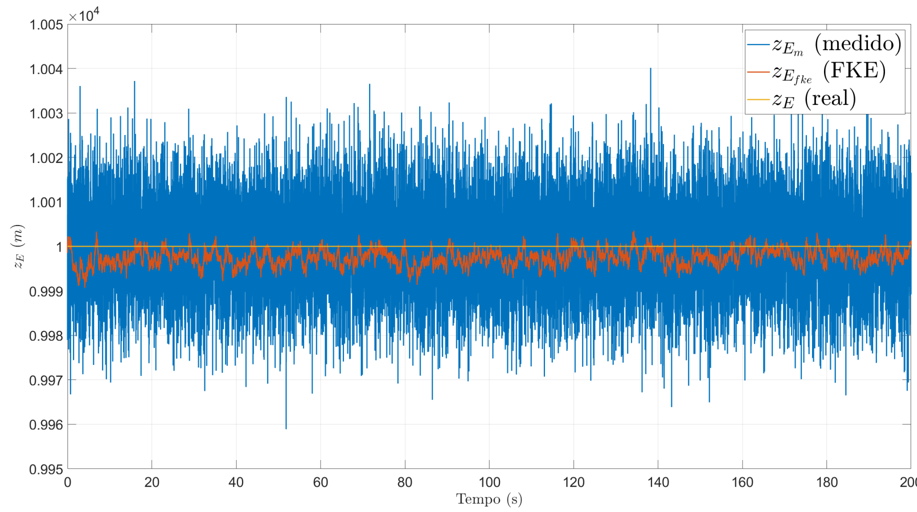
Na Fig. 41 estão as estimações das posições lineares x_E, y_E e z_E para a curva coordenada. Percebe-se o mesmo comportamento visto na condição de voo reto e nivelado, a sintonização e inicialização do filtro foram satisfatórias e o erro (Fig. 42), que apesar de estar em alguns momentos fora dos limites do intervalo de confiança, se estabilizou na maioria do tempo.



(a) Posição linear x_E.

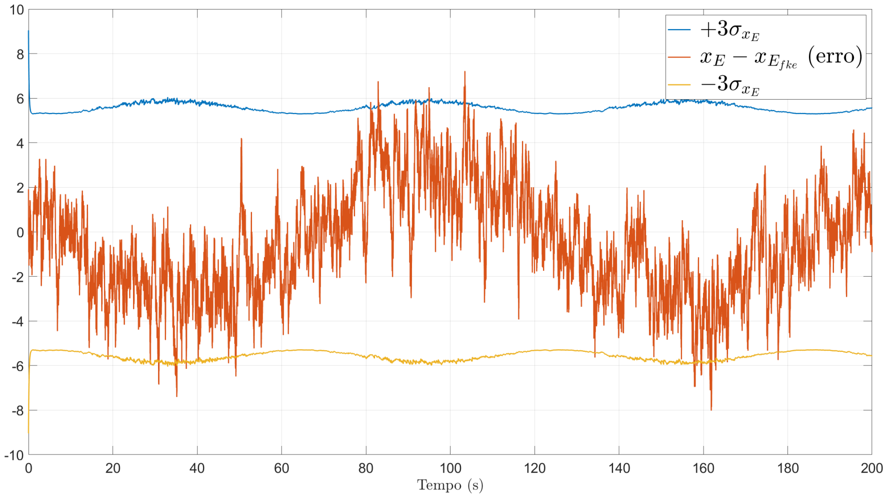


(b) Posição linear y_E.

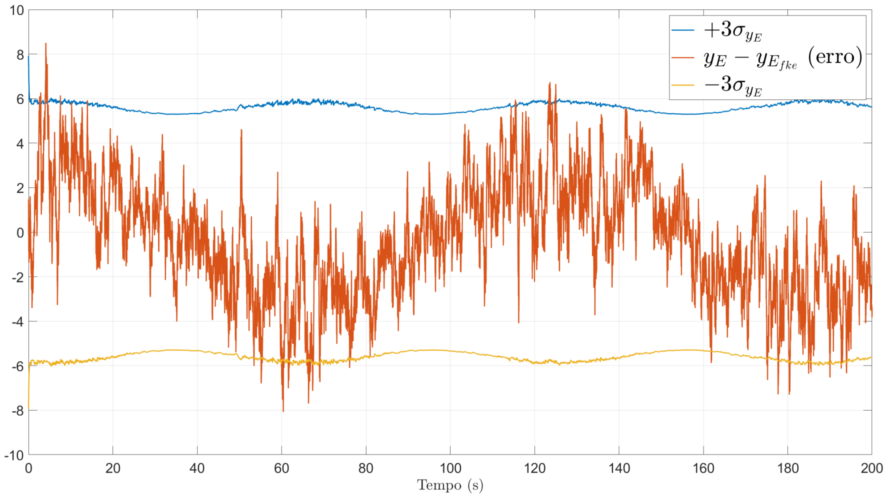


(c) Posição linear z_E.

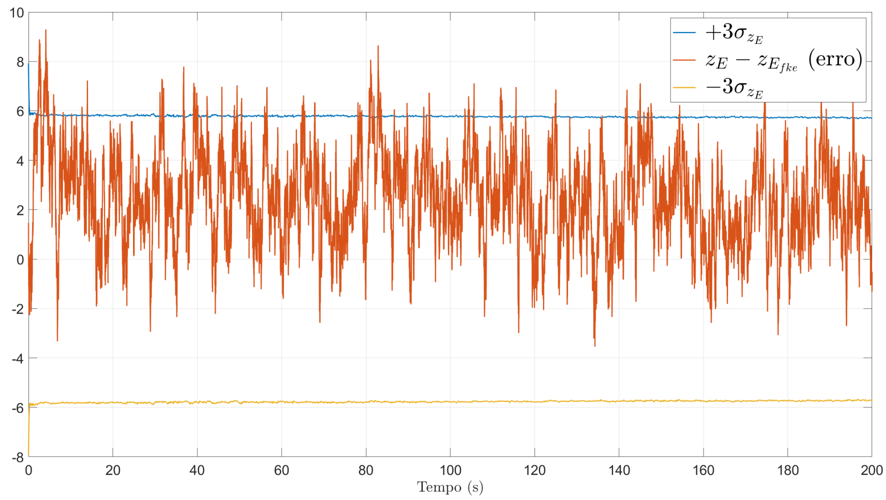
Figura 41 – Posições lineares medidas, estimadas e reais (curva coordenada).



(a) Posição linear x_E.



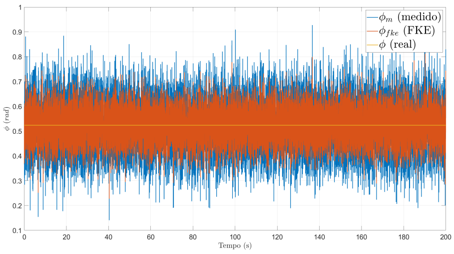
(b) Posição linear y_E.



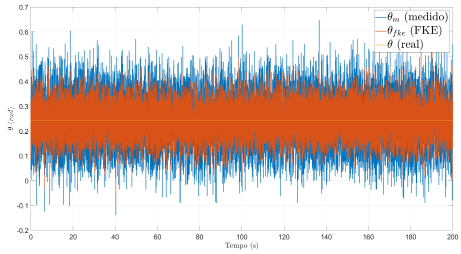
(c) Posição linear z_E.

Figura 42 – Erros e intervalos de confiança para as posições lineares (curva coordenada).

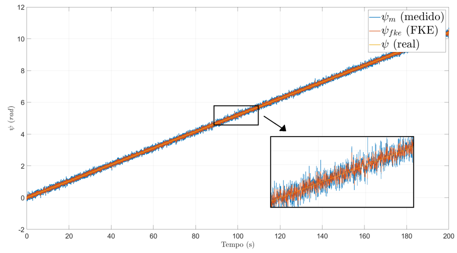
O mesmo padrão visto no voo reto e nivelado para as posições angulares ϕ, θ e ψ (Fig. 43 e 44) é verificado para a curva coordenada, onde o FKE cumpre seu trabalho de filtrar o ruido vindo do giroscópio.



(a) Posição angular ϕ.

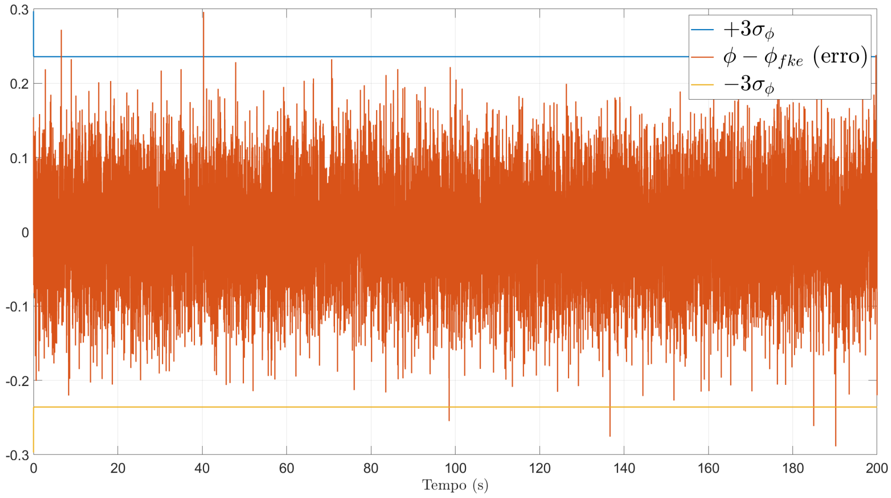


(b) Posição linear θ.

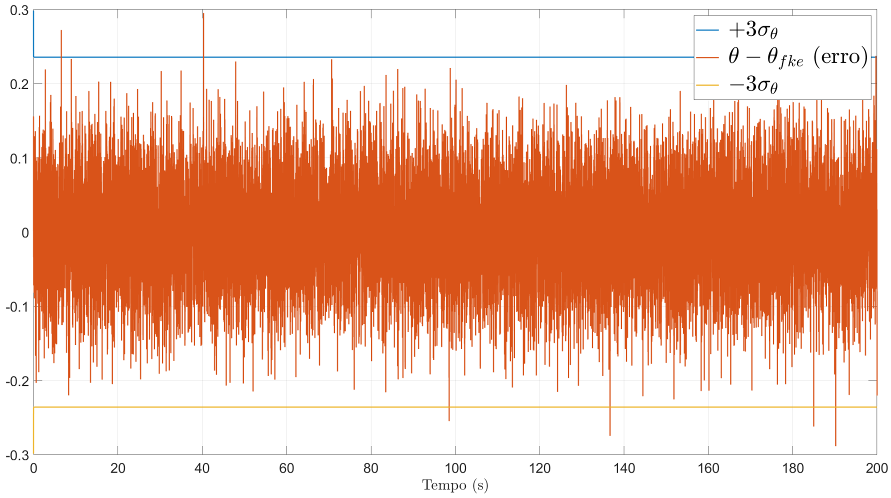


(c) Posição linear ψ.

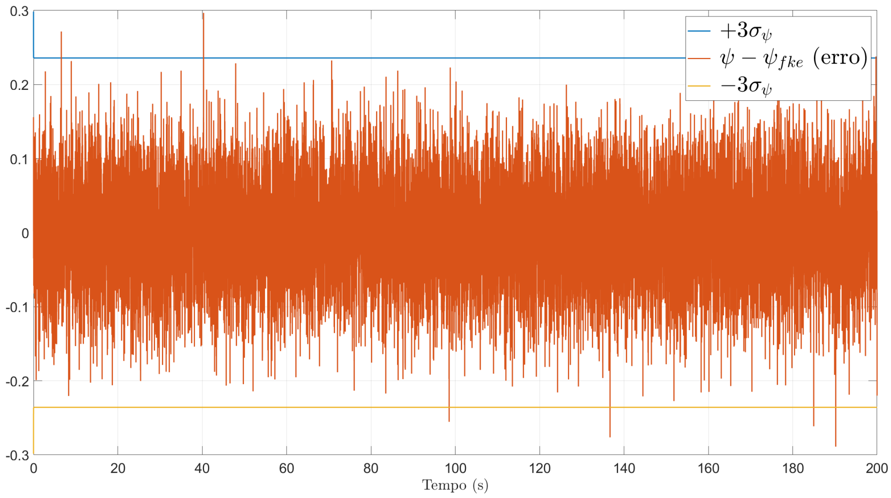
Figura 43 – Posições angulares medidas, estimadas e reais (curva coordenada).



(a) Posição angular ϕ.



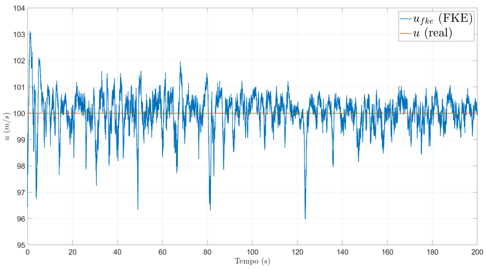
(b) Posição linear θ.



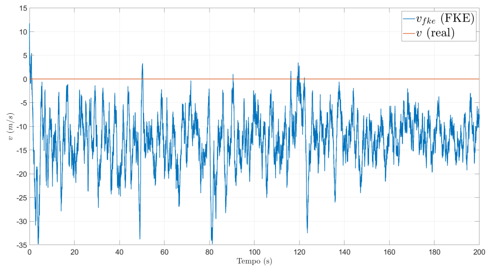
(c) Posição linear ψ.

Figura 44 – Erros e intervalos de confiança para as posições angulares (curva coordenada).

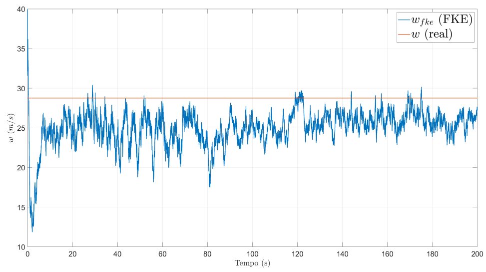
Um padrão semelhante visto no voo reto e nivelado também é observado para as velocidades lineares u e w da Fig. 45. O FKE sofreu um pouco mais de dificuldade para estabilizar, e somente a velocidade v ficou fora do intervalo de confiança (Fig. 46). Vale lembrar novamente que as velocidades lineares não são medidas diretamente, fazendo o FKE ter um pouco mais de dificuldade de estimar esses parâmetros. Já para as velocidades angulares p, q e r, o FKE se comportou melhor (Figs. 47 e 48) e obteve o mesmo padrão do visto no voo reto e nivelado.



(a) Velocidade linear u.

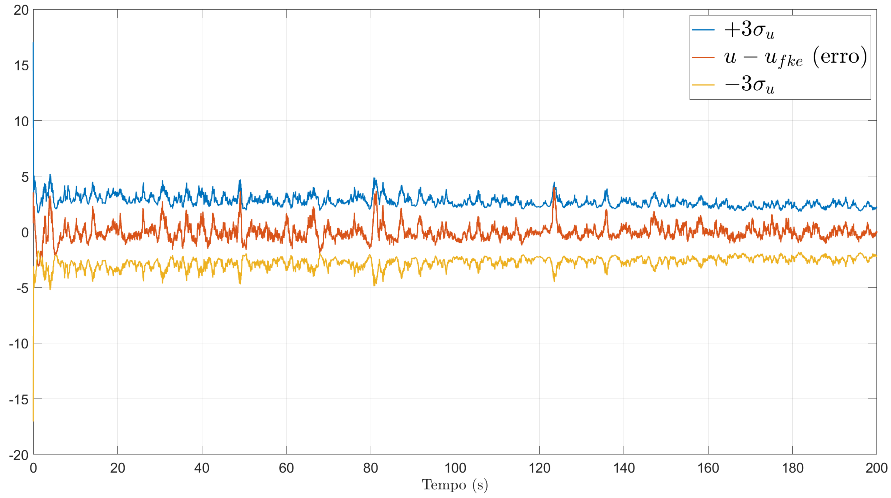


(b) Velocidade linear v.

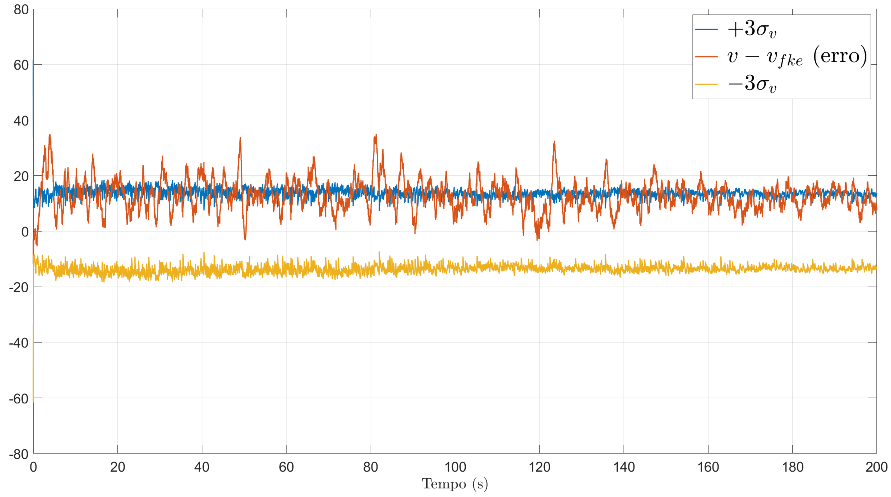


(c) Velocidade linear w.

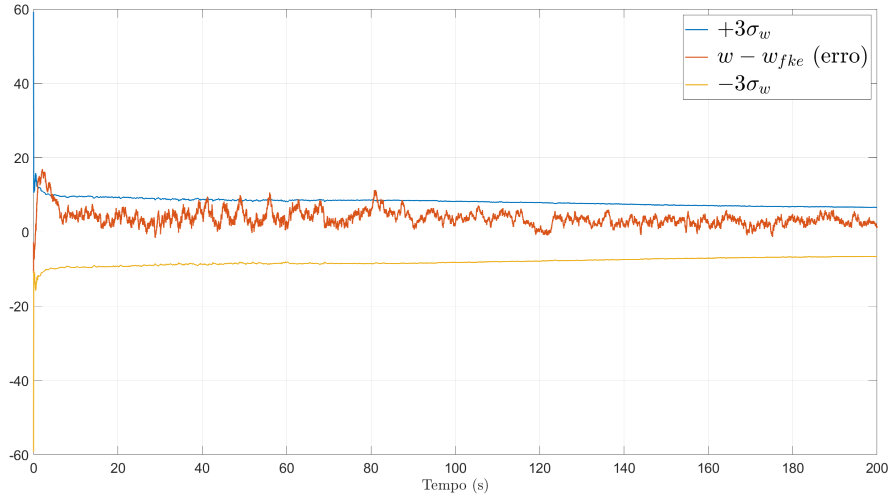
Figura 45 – Velocidades lineares estimadas e reais (curva coordenada).



(a) Velocidade linear u.

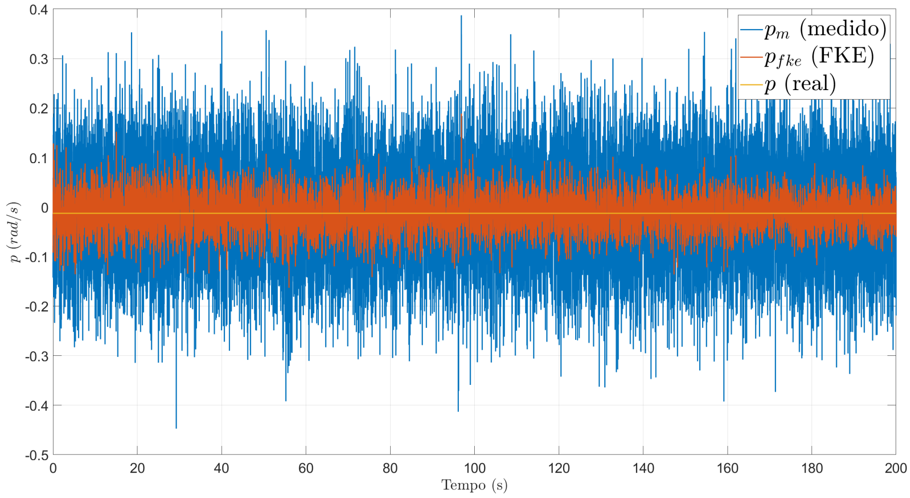


(b) Velocidade linear v.

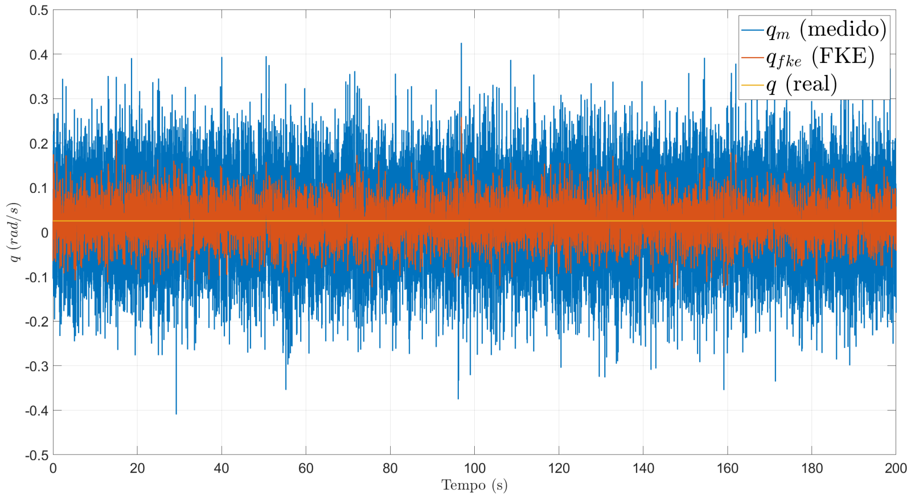


(c) Velocidade linear w.

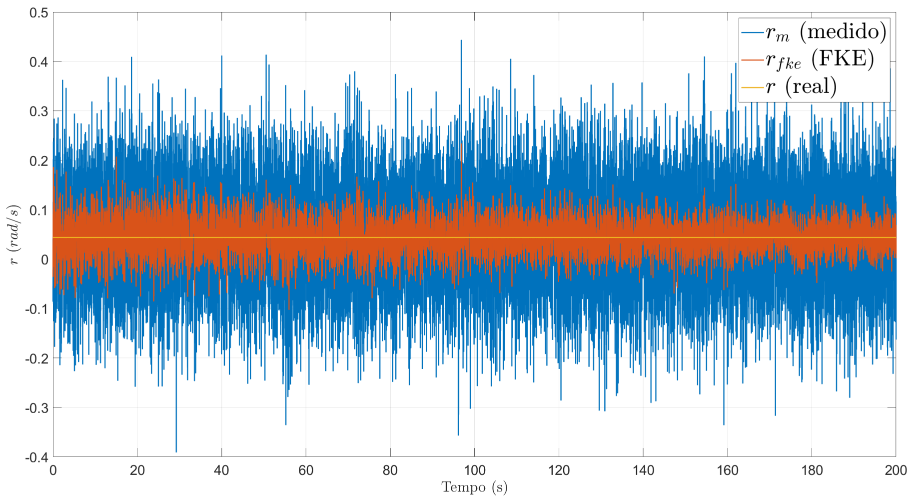
Figura 46 – Erros e intervalos de confiança para as velocidades lineares (curva coordenada).



(a) Velocidade angular p.

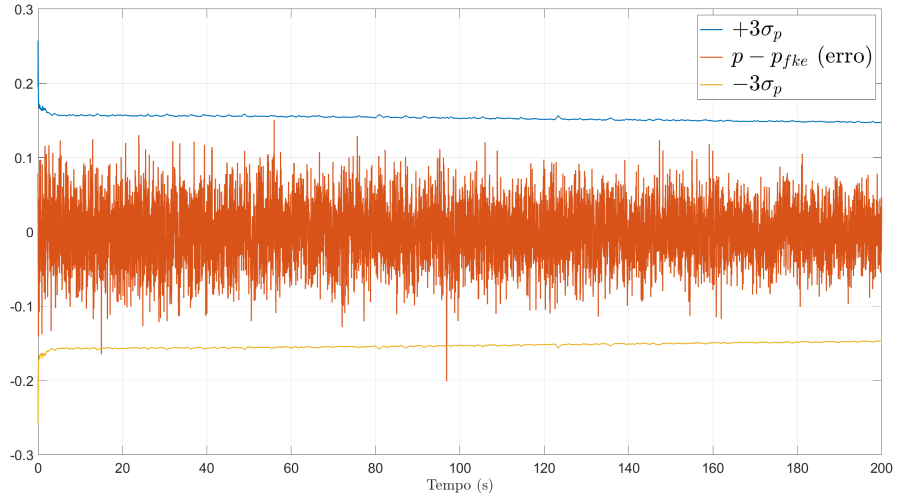


(b) Velocidade angular q.

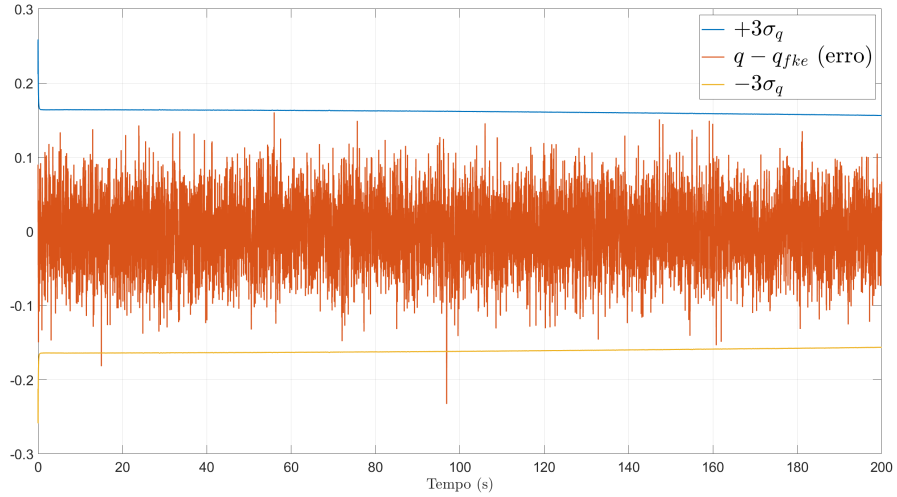


(c) Velocidade angular r.

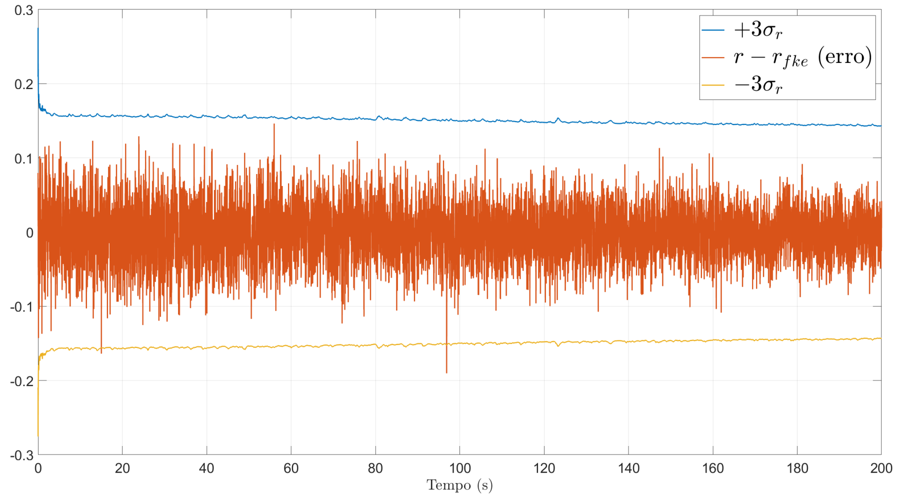
Figura 47 – Velocidades angulares medidas, estimadas e reais (curva coordenada).



(a) Velocidade angular p.



(b) Velocidade angular q.



(c) Velocidade angular r.

Figura 48 – Erros e intervalos de confiança para as velocidades angulares (curva coordenada).

Partindo para a análise dos parâmetros aerodinâmicos, o FKE também fez um bom trabalho estimando C_{D_0} para a condição de curva coordenada (Fig. 49), com o erro absoluto dentro do intervalo de confiança (Fig. 50).

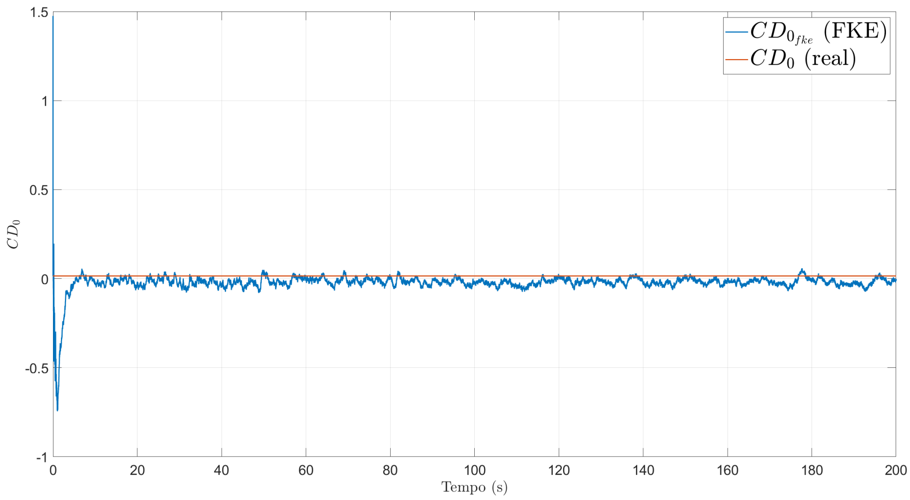


Figura 49 – C_{D_0} estimado e real (curva coordenada).

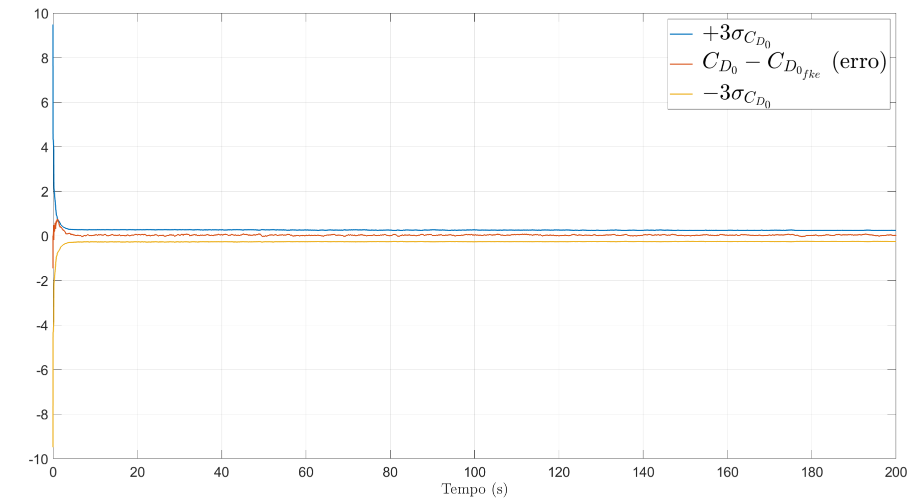
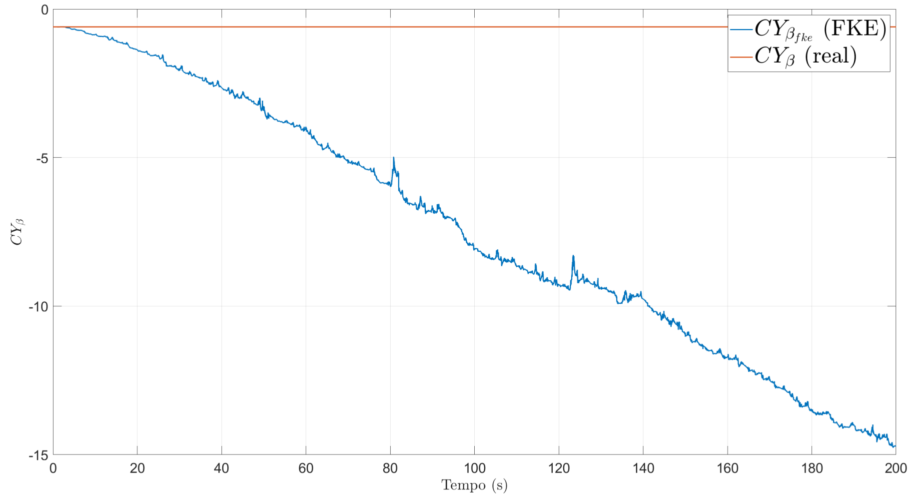
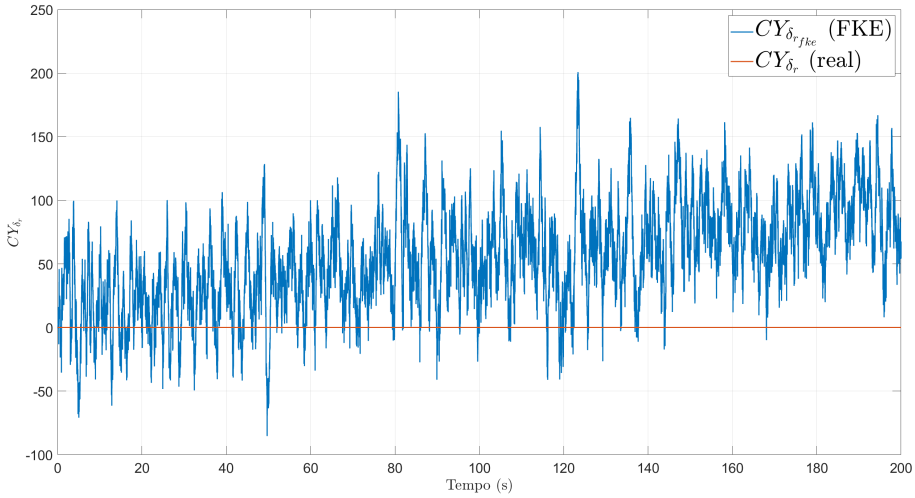


Figura 50 – Erro e intervalo de confiança para C_{D_0} (curva coordenada).

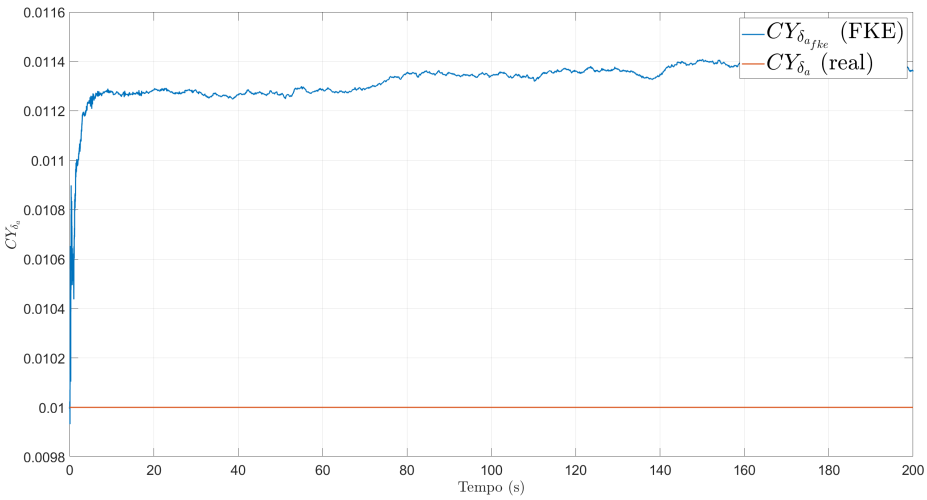
Para os parâmetros de força lateral, somente C_{Y_β} e $C_{Y_{\delta_r}}$ não apresentaram um resultado esperado (Figs. 51 e 52), tendo seus valores e erros divergindo com o tempo e não estabilizando a depender da inicialização. Como falado na condição de voo e nivelado, diversas sintonias também foram feitas para se obter uma curva que se aproximasse do valor real, porém, nenhuma foi encontrada.



(a) Devido à derrapagem β.

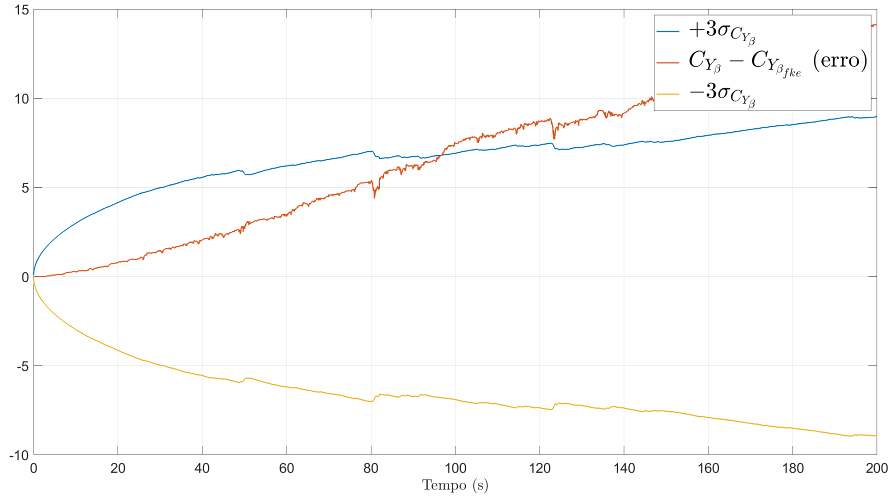


(b) Devido à deflexão angular do leme δ_r.

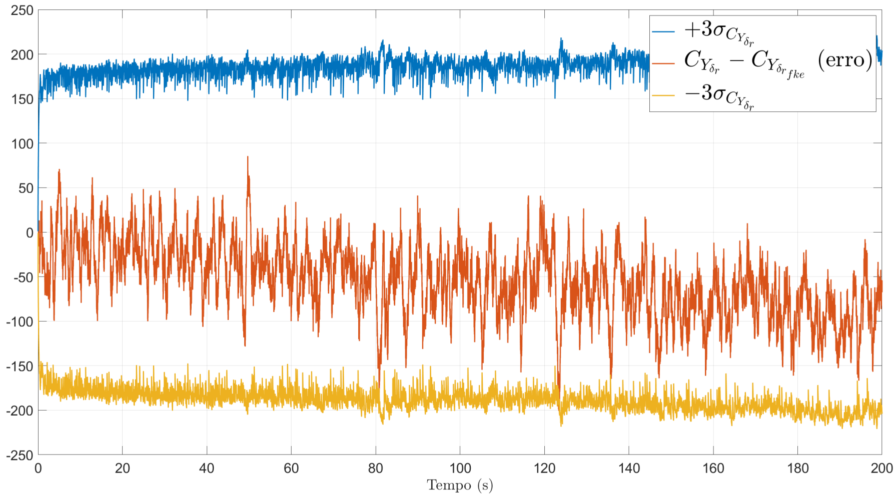


(c) Devido à deflexão angular dos ailerons δ_a.

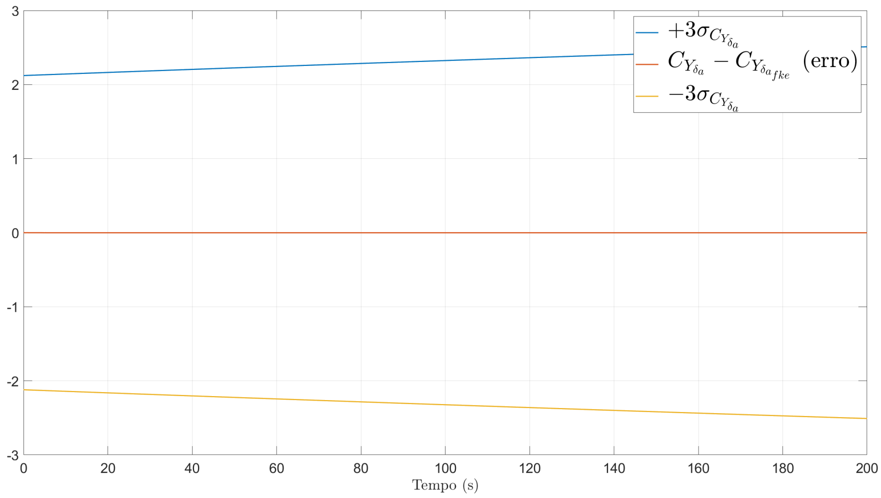
Figura 51 – C_Y estimados e reais (curva coordenada).



(a) Devido à derrapagem β.



(b) Devido à deflexão angular do leme δ_r.



(c) Devido à deflexão angular dos ailerons δ_a.

Figura 52 – Erros e intervalos de confiança para C_Y (curva coordenada).

Assim como na condição de voo reto e nivelado, o comportamento de C_{L_0}, tendeu a divergir com o tempo (Fig. 53). Diversas sintonizações foram feitas, mas nenhuma foi capaz de fazer a curva convergir para o valor real. Contudo, dentro do tempo de 200 s simulado, o erro (Fig. 54) ficou dentro dos limites estabelecidos pelo desvio padrão.

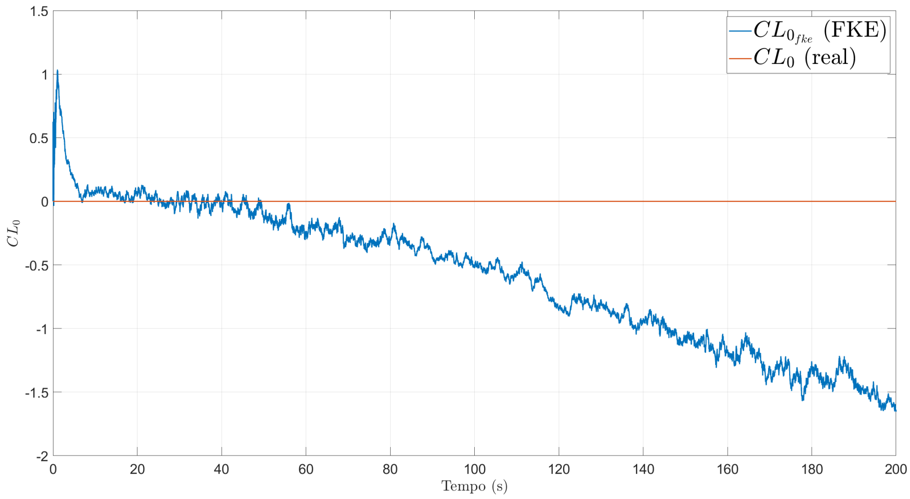


Figura 53 – C_{L_0} estimado e real (curva coordenada).

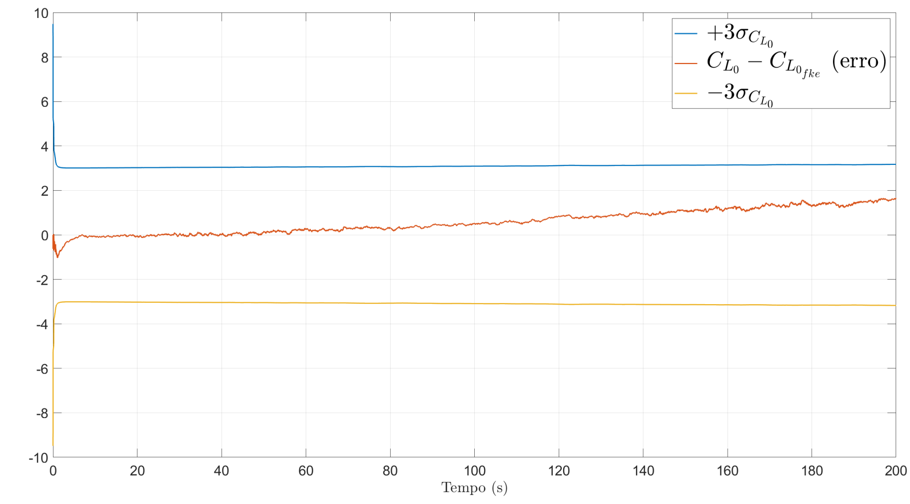
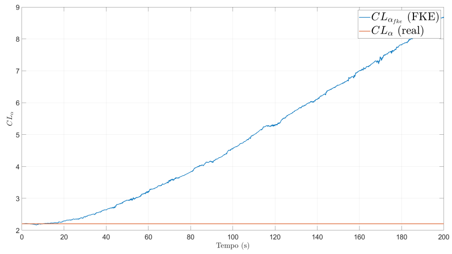
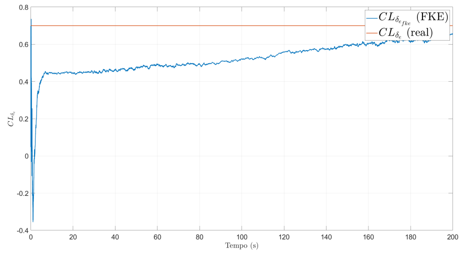


Figura 54 – Erro e intervalo de confiança para C_{L_0} (curva coordenada).

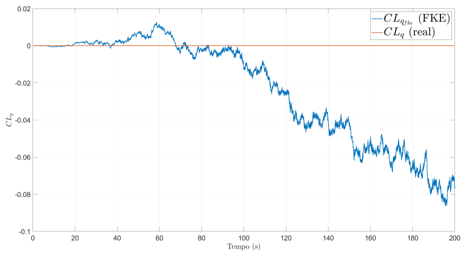
Um comportamento diferente do observado na condição de voo reto e nivelado foi apresentado para os coeficientes C_{L_α} e C_{L_q}, que tiveram tendências divergentes, enquanto que $C_{L_{\delta_e}}$ convergiu para o valor real (Fig. 55). O mesmo acontece com os erros (Fig. 56), com apenas C_{L_α} ficando fora dos limites de desvio padrão.



(a) Devido ao ângulo de ataque α.

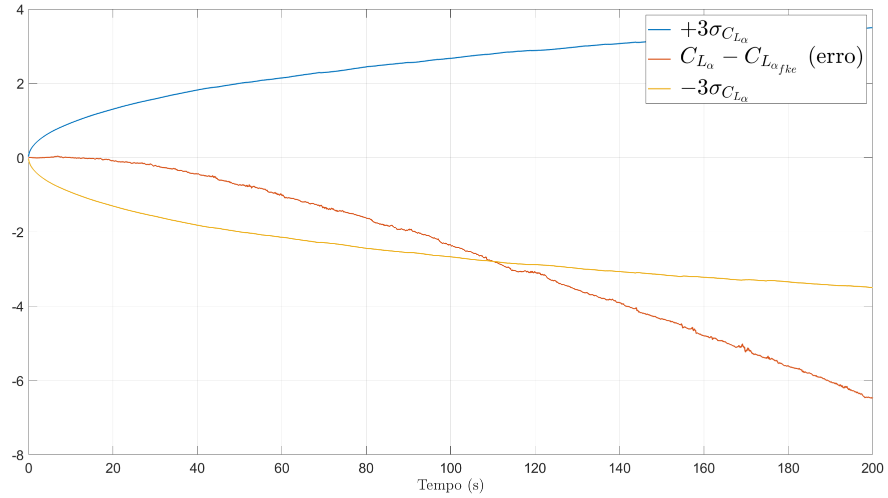


(b) Devido à deflexão angular do profundor δ_e.

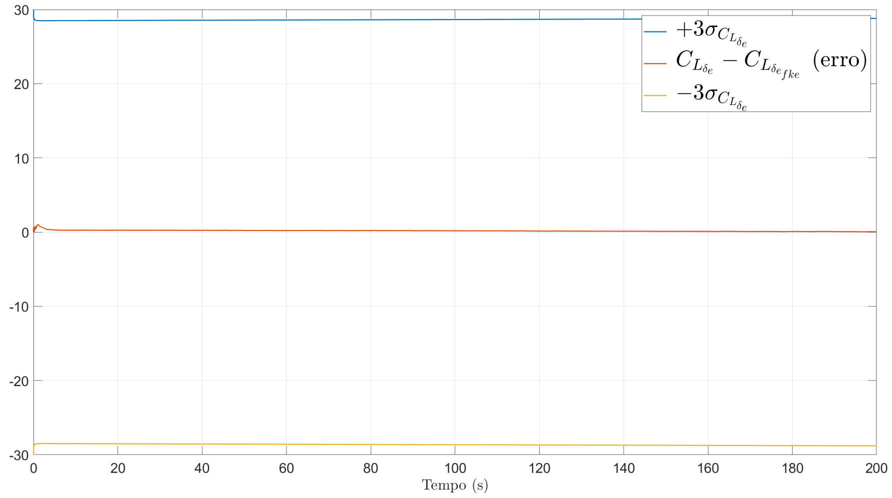


(c) Devido à velocidade de arfagem q.

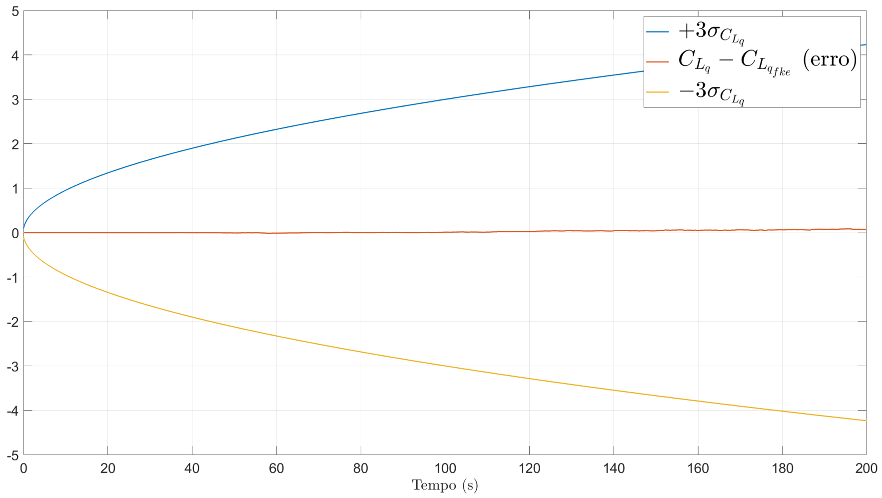
Figura 55 – C_L estimados e reais (curva coordenada).



(a) Devido ao ângulo de ataque α.



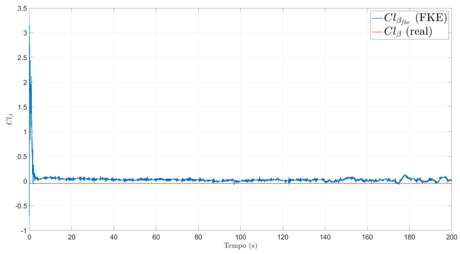
(b) Devido à deflexão angular do profundor δ_e.



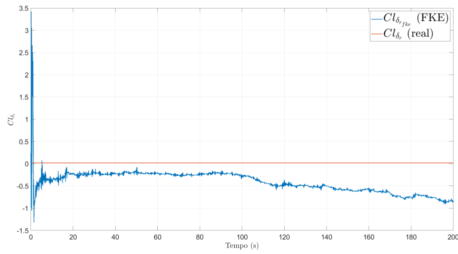
(c) Devido à velocidade de arfagem q.

Figura 56 – Erros e intervalos de confiança para C_L (curva coordenada).

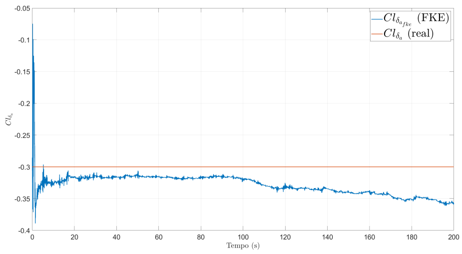
Já para os parâmetros aerodinâmicos de momento, o comportamento foi semelhante ao da condição de voo reto e nivelado, sendo bem distribuído, com alguns parâmetros sendo bem estimados (C_{l_β}, $C_{l_{\delta_r}}$ e $C_{l_{\delta_a}}$) e outros (C_{l_p} e C_{l_r}) saindo dos limites de confiança, como pode ser visto pelas Figs. 57 e 58 abaixo.



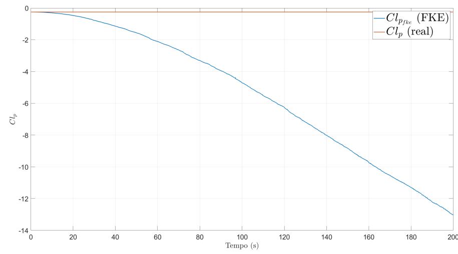
(a) Devido à derrapagem β.



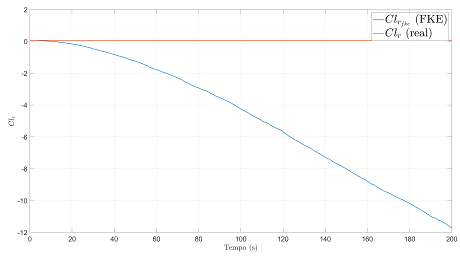
(b) Devido à deflexão angular do leme δ_r.



(c) Devido à deflexão angular dos ailerons δ_a.

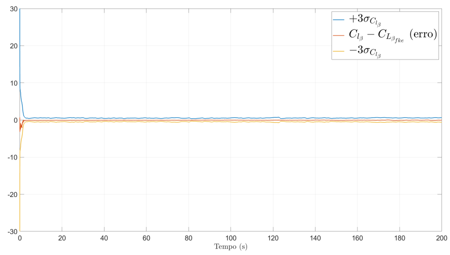


(d) Devido à velocidade de rolagem p.

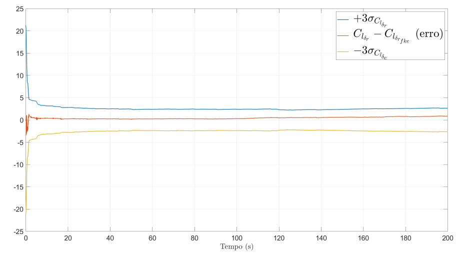


(e) Devido à velocidade de guinada r.

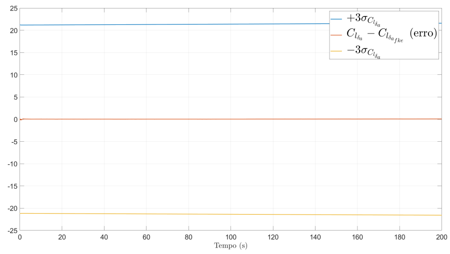
Figura 57 – C_l estimados e reais (curva coordenada).



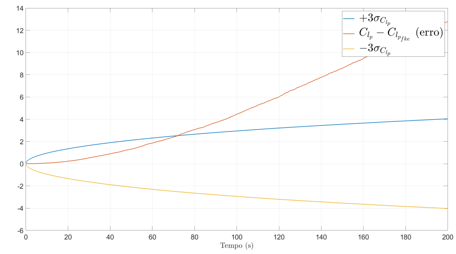
(a) Devido à derrapagem β.



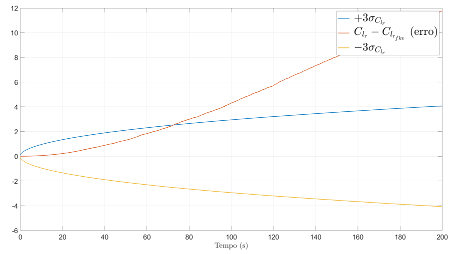
(b) Devido à deflexão angular do leme δ_r.



(c) Devido à deflexão angular dos ailerons δ_a.



(d) Devido à velocidade de rolagem p.



(e) Devido à velocidade de guinada r.

Figura 58 – Erros e intervalos de confiança para C_l (curva coordenada).

O comportamento da estimação para C_{m_0} para a condição de curva coordenada foi semelhante visto na condição de voo reto e nivelado, com a curva da Fig. 59 tendendo a divergir. Contudo, dentro do tempo simulado, o erro ficou dentro dos limites a maior parte do tempo (Fig. 60).

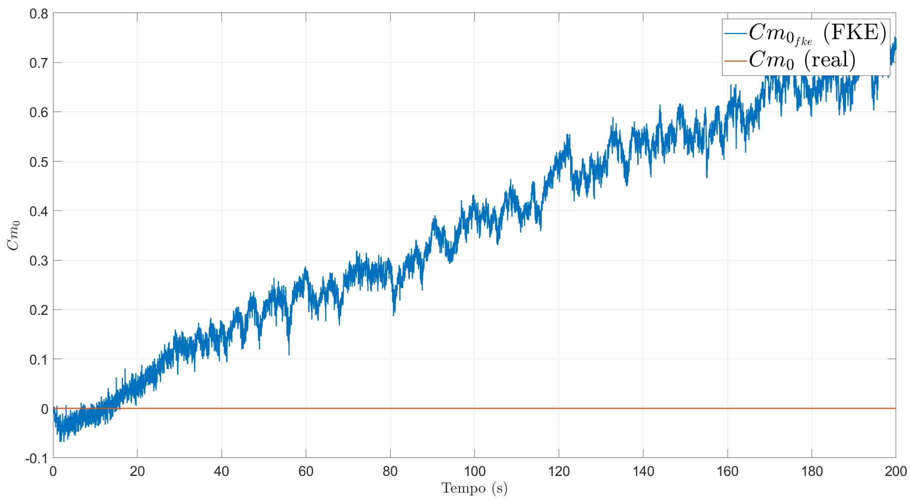


Figura 59 – C_{m_0} estimado e real (curva coordenada).

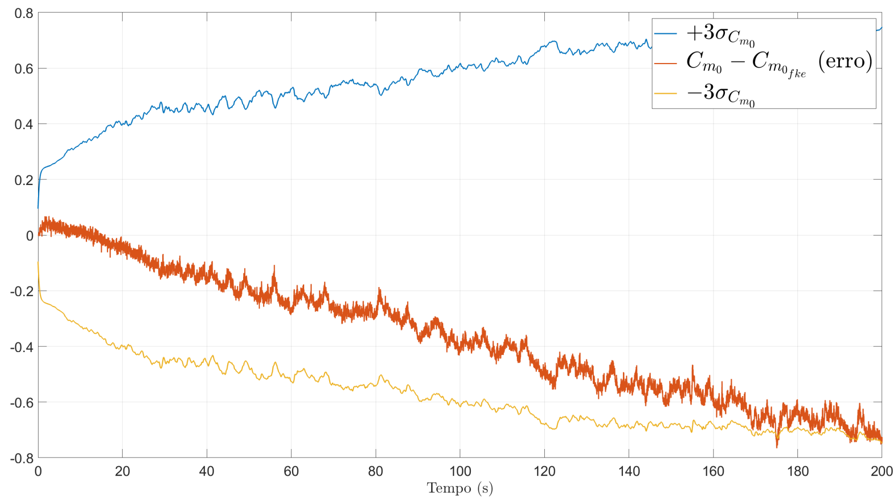
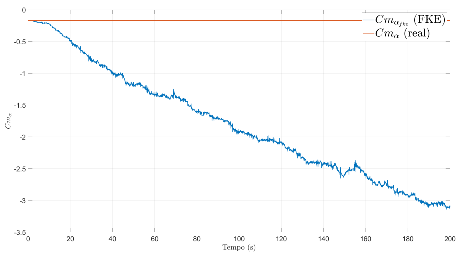
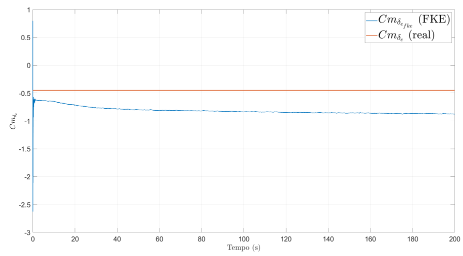


Figura 60 – Erro e intervalo de confiança para C_{m_0} (curva coordenada).

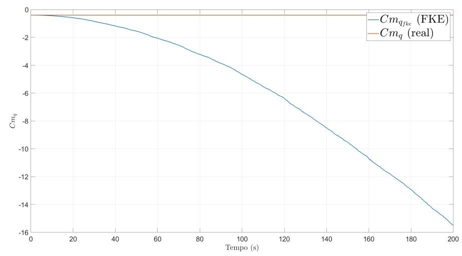
Assim como na condição de voo reto e nivelado, os parâmetros C_{m_α} e C_{m_q} não convergiram e saíram dos batentes de intervalo de confiança, como mostra as Figs. 61 e 62. Além disso, o valor e erro de $C_{m_{\delta_e}}$ ficaram dentro dos limites.



(a) Devido ao ângulo de ataque α.

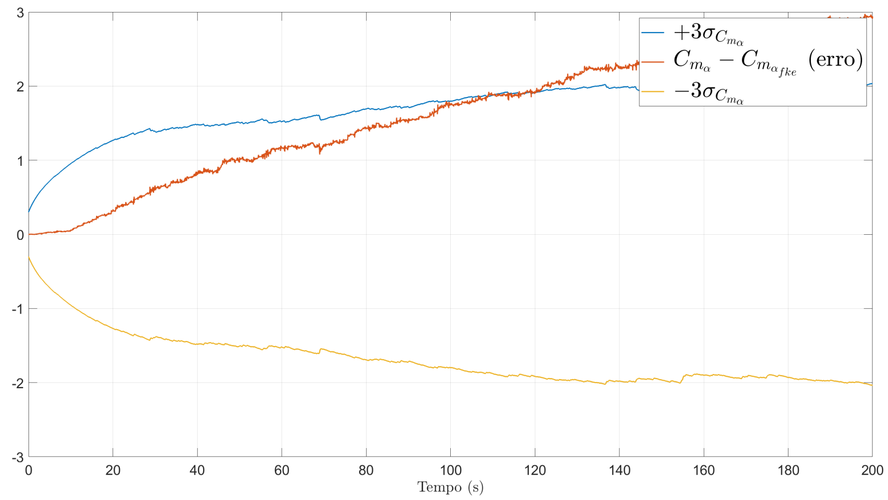


(b) Devido à deflexão angular do profundor δ_e.

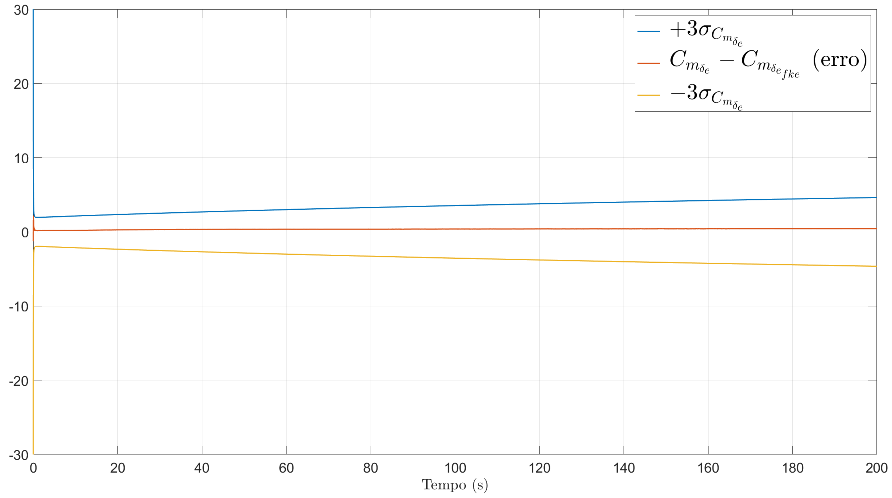


(c) Devido à velocidade de guinada q.

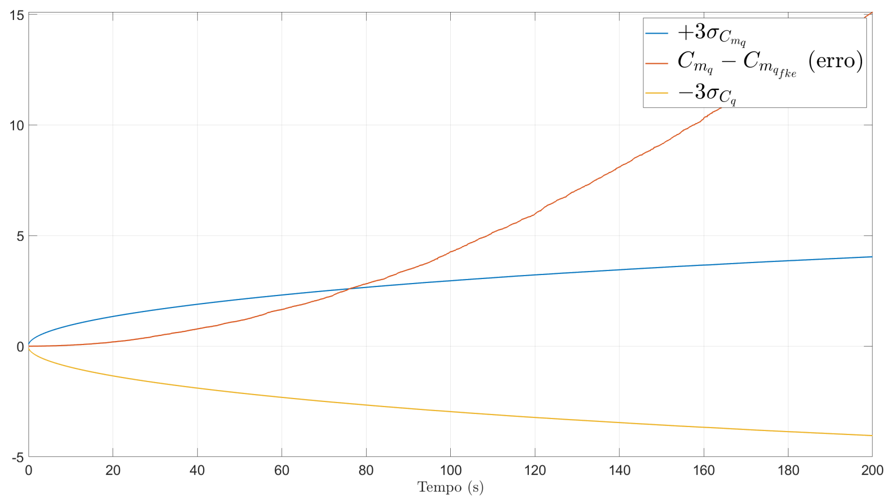
Figura 61 – C_m estimados e reais (curva coordenada).



(a) Devido ao ângulo de ataque α.



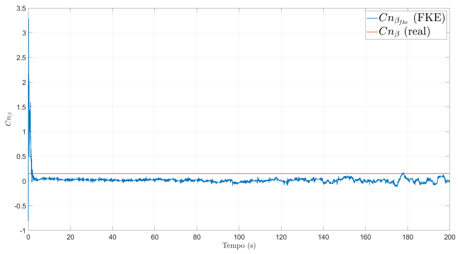
(b) Devido à deflexão angular do profundor δ_e.



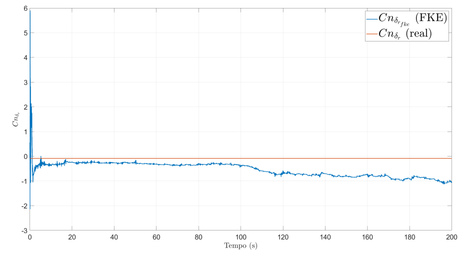
(c) Devido à velocidade de guinada q.

Figura 62 – Erros e intervalos de confiança para C_m (curva coordenada).

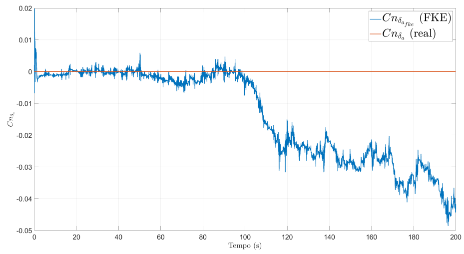
Os parâmetros de momento de guinada se comportaram de forma mista e em apenas 3 deles (C_{n_β}, $C_{n_{\delta_r}}$ e $C_{n_{\delta_a}}$), a estimação ficou próxima ao valor real. Já para os demais parâmetros (C_{n_p} e C_{n_r}), a convergência não ocorreu e novamente os limites de desvio padrão foram superados, como visto nas Figs. 63 e 64.



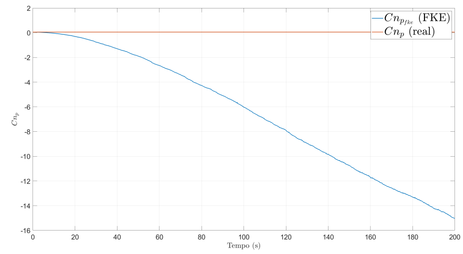
(a) Devido à derrapagem β.



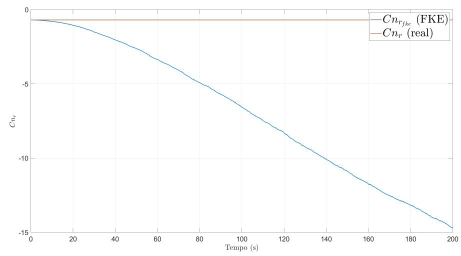
(b) Devido à deflexão angular do leme δ_r.



(c) Devido à deflexão angular dos ailerons δ_a.

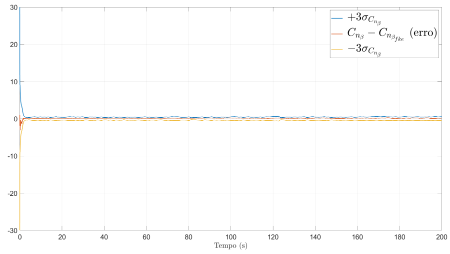


(d) Devido à velocidade de rolagem p.

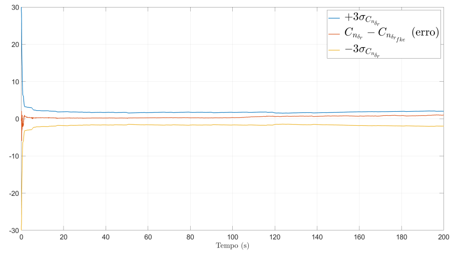


(e) Devido à velocidade de guinada r.

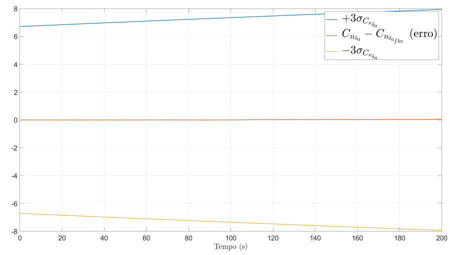
Figura 63 – C_n estimados e reais (curva coordenada).



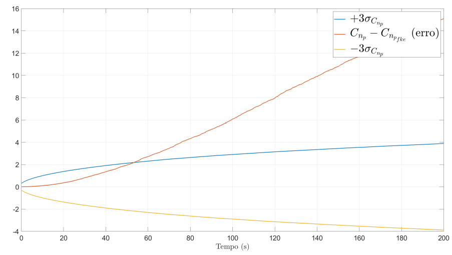
(a) Devido à derrapagem β.



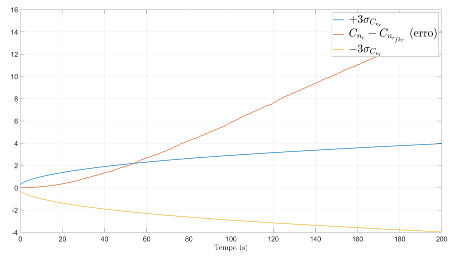
(b) Devido à deflexão angular do leme δ_r.



(c) Devido à deflexão angular dos ailerons δ_a.



(d) Devido à velocidade de rolagem p.



(e) Devido à velocidade de guinada r.

Figura 64 – Erros e intervalos de confiança para C_n (curva coordenada).

7 Conclusões

Neste estudo foi realizada a estimação dos estados do caça Mirage III em condição de voo reto e nivelado e curva coordenada utilizando um filtro de Kalman estendido. A implementação envolveu modelagem em MATLAB® e Simulink® e a análise abrangeu a comparação entre os estados estimados pelo filtro e os valores reais, gerados por um modelo dinâmico não linear, bem como com as medidas dos sensores. O desempenho do FKE foi avaliado através do erro absoluto e do intervalo de confiança estabelecido para cada parâmetro estimado.

O FKE demonstrou um bom desempenho na estimação das posições lineares (x_E, y_E, z_E), posições angulares (ϕ, θ, ψ) e velocidades angulares (p, q, r) para ambas as condições, com erros dentro dos intervalos de confiança estabelecidos. A eficácia do filtro foi notável na redução do ruído nas medidas do GPS e giroscópio. No entanto, o filtro apresentou uma maior dificuldade na estabilização das velocidades lineares u e w, embora essas estimativas tenham permanecido dentro dos intervalos de confiança.

A estimação dos parâmetros aerodinâmicos apresentou um comportamento variado. O coeficiente de arrasto (C_{D_0}) foi bem estimado em ambas as condições, mantendo-se consistentemente dentro dos limites de confiança. No entanto, os parâmetros relacionados à força lateral mostraram dificuldades significativas, especialmente os coeficientes C_{Y_β}, C_{m_α}, C_{m_q}, C_{l_p}, C_{l_r}, C_{n_p} e C_{n_r}, cujas estimativas não estabilizaram adequadamente ao longo do tempo.

Quanto aos coeficientes de força de sustentação, como C_{L_0}, C_{L_α}, $C_{L_{\delta_e}}$ e C_{L_q}, houveram divergências notáveis na condição de voo e reto e nivelado, com alguns parâmetros saindo dos limites de confiança e demandando mais simulações para se verificar uma possível convergência. Já na condição de curva coordenada, o coeficiente $C_{L_{\delta_e}}$ foi melhor estimado pelo FKE. Por fim, a estimação dos parâmetros de momento exibiu um comportamento misto: alguns coeficientes foram estimados com precisão, enquanto outros não conseguiram convergir e excederam os limites de desvio padrão.

As dificuldades na estimação de certos parâmetros, como o coeficiente C_{Y_β} e alguns coeficientes de momento, indica a necessidade de aprimoramentos na sintonização do filtro, como o uso de sintonização automática. Além disso, prolongar o tempo de simulação pode proporcionar uma avaliação mais completa da convergência dos parâmetros que apresentaram divergência, permitindo ajustes mais refinados nos parâmetros do filtro.

Sendo assim, o filtro de Kalman estendido provou ser uma ferramenta valiosa para a estimação dos estados do caça Mirage III, demonstrando boa performance na maioria dos parâmetros analisados. Apesar das dificuldades encontradas, os resultados obtidos oferecem uma base sólida para futuros aprimoramentos e aplicações práticas do filtro em sistemas de navegação e controle de aeronaves. Esse trabalho contribui para o avanço do conhecimento na área de estimação de estados e abre novas possibilidades para pesquisas e melhorias na tecnologia de filtros.

7.1 Sugestões para trabalhos futuros

Para próximos trabalhos, sugere-se:

- Experimentar diferentes estratégias de sintonização para o FKE, visando melhorar a estimação de parâmetros que apresentaram dificuldades;
- Implementar técnicas de ajuste adaptativo para otimizar o desempenho do filtro em diferentes condições de voo;
- Realizar simulações mais longas para observar a convergência completa dos parâmetros;
- Explorar a aplicação de sensores adicionais na modelagem para uma melhor representação;
- Validar as estimativas e o desempenho do FKE com dados experimentais reais obtidos em voo, para verificar a precisão e a aplicabilidade das simulações;
- Investigar o uso de técnicas alternativas de filtragem, como o filtro de Kalman *unscented* e/ou o filtro de partículas, para lidar com não-linearidades e melhorar a estimação de parâmetros complexos.

Referências

AGUIRRE, L. A. *Fundamentos de instrumentação*. São Paulo: Pearson Education do Brasil, 2013. Citado 2 vezes nas páginas 35 e 36.

ALLISON, G. *Lockheed Martin delivers 52nd C-5M Super Galaxy*. 2018. <https://ukdefencejournal.org.uk/lockheed-martin-delivers-52nd-c-5m-super-galaxy/>. Acesso em 23/11/2023. Citado 2 vezes nas páginas 6 e 20.

ASCORTI, L. *An application of the extended kalman filter to the attitude control of a quadrotor*. Milão, Itália: [s.n.], 2013. Citado na página 46.

ASSIS, J. P. et al. *Estimação estatística*. Mato Grosso: Pantanal Editora, 2021. Citado na página 32.

AVIATION, D. *MIRAGE III*. 2016. <https://www.dassault-aviation.com/en/passion/aircraft/military-dassault-aircraft/mirage-iii/>. Acesso em 16/10/2023. Citado na página 45.

BOYCE, W. E.; DIPRIMA, R. C. *Equações diferenciais elementares e problemas de valores de contorno*. 9. ed. Rio de Janeiro: LTC - Livros Tecnicos e Cientificos Editora Ltda., 2010. Citado na página 30.

BRASILEIRA, F. A. *Mirage - DASSAULT MIRAGE III-EBR (F-103E) | Avions Marcel Dassault*. 2015. <https://www2.fab.mil.br/musal/index.php/aeronaves-em-exposicao/55-avioes/333-f-103e>. Acesso em 16/10/2023. Citado 2 vezes nas páginas 6 e 44.

BROWN, C. M. An extended kalman filter for estimating aerodynamic coefficients. 1976. Citado na página 20.

BUSARELLO, T. D. C.; SIMõES, M. G. A tutorial on implementing kalman filters with commonly used blocks. In: *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*. [S.l.: s.n.], 2019. v. 1, p. 60–67. Citado 2 vezes nas páginas 6 e 39.

CHOWDHARY, D.; JATEGAONKAR, R. Aerodynamic parameter estimation from flight data applying extended and unscented kalman filter. *Aerospace Science and Technology*, v. 14, n. 2, p. 106–117, 2010. ISSN 1270-9638. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1270963809000650>. Citado na página 20.

CRASSIDIS, J. L.; JUNKINS, J. L. *Optimal Estimation of Dynamic Systems*. 2. ed. Flórida: Taylor & Francis Group, 2012. Citado 9 vezes nas páginas 6, 25, 30, 32, 33, 35, 36, 37 e 38.

CURVO, M. Estimation of aircraft aerodynamic derivatives using extended kalman filter. *Journal of the Brazilian Society of Mechanical Sciences*, The Brazilian Society of Mechanical Sciences, v. 22, n. 2, p. 133–148, 2000. ISSN 0100-7386. Disponível em: <https://doi.org/10.1590/S0100-73862000000200001>. Citado na página 20.

DEGROOT, M. H.; SCHERVISH, M. J. *Probability and Statistics*. 4. ed. Boston: Pearson Education, Inc., 2012. Citado 3 vezes nas páginas 6, 32 e 34.

DEVORE, J. L. *Prbabilidade e Estatística para Engenharia e Ciências*. 6. ed. São Paulo: Cengage Learning, 2006. Citado 4 vezes nas páginas 6, 32, 34 e 35.

ELIASSON, S. R. *Maximum likehood estimation: logic and practice*. Parque Newbury: SAGE Publications, Inc, 1993. Citado na página 19.

ETKIN, B.; REID, L. D. *Dynamics of flight: stability and control*. 3. ed. Nova Iorque: John Wiley & Sons, Inc, 1996. Citado 5 vezes nas páginas 6, 22, 23, 24 e 25.

GREWAL, M. S.; ANDREWS, A. P. *Kalman Filtering: Theory and Practice Using MATLAB*. 2. ed. Nova Iorque: John Wiley & Sons, Inc, 2001. Citado na página 19.

GREWAL, M. S.; ANDREWS, A. P. Applications of kalman filtering in aerospace 1960 to the present [historical perspectives]. *IEEE Control Systems Magazine*, v. 30, n. 3, p. 69–78, 2010. Citado na página 19.

KOKOLIOS, A. Use of a kalman filter for the determination of aircraft aerodynamic characteristics from flight test data. In: . [s.n.], 1994. Disponível em: <https://api.semanticscholar.org/CorpusID:61347215>. Citado na página 20.

MCCABE, G. P.; A., C. B.; S., M. D. *Introduction to the Practice of Statistics*. 6. ed. Nova Iorque: W.H. Freeman and Company, 2009. Citado 2 vezes nas páginas 19 e 35.

MCGEE, L. A.; SCHMIDT, S. F. Discovery of the kalman filter as a practical tool for aerospace and industry. *NASA-TM-86847, N86-13311*, 1985. Citado na página 19.

NISE, N. S. *Engenharia de sistesmas de controle*. 6. ed. Rio de Janeiro: LTC - Livros Técnicos e Científicos Eitora Ltda., 2012. Citado 2 vezes nas páginas 29 e 30.

PAGLIONE, P.; ZANARDI, M. C. *Estabilidade e Controle de Aeronaves - Curso MV0-03*. São José dos campos: Centro Técnico Aeroespacial - Instituto Técnológico da Aeronáutica, 1990. Citado 5 vezes nas páginas 9, 25, 26, 27 e 45.

QUAN, Q. *Introduction to Multicopter Design and Control*. Singapura: Springer Nature Singapore Pte Ltd, 2017. Citado 3 vezes nas páginas 6, 23 e 46.

SIMON, D. *Optimal State Estimation*. Nova Jersey: John Wiley & Sons, Inc, 2006. Citado 8 vezes nas páginas 6, 32, 33, 37, 38, 40, 41 e 42.

STEVENS, B. L.; LEWIS, F. L. *Aircraft Control and Simulation*. 2. ed. Nova Iorque: John Wiley & Sons, Inc, 1992. Citado 4 vezes nas páginas 24, 25, 29 e 30.

TEREJANU, G. Discrete kalman filter tutorial. In: . [s.n.], 2009. Disponível em: <https://api.semanticscholar.org/CorpusID:36003090>. Citado 2 vezes nas páginas 6 e 40.

THERRIEN, C. W. *Discrete random signals and statistical signal processing*. Nova Jersey: Prentice Hall, 1992. Citado na página 35.

Apêndices

APÊNDICE A – *Scripts* MATLAB®

A.1 *main.m*

```
addpath('.\FKE');
addpath('.\Modelo_dinamico');
close all; clear; clc;

%% Definicao da trajetoria
% Se traj = 1 -> voo reto nivelado, caso contrario -> curva
    coordenada
traj = 1;

% Altitude (em m) e velocidade (em m/s) para as condicoes
h_rn = 10000; v_rn = 980/3.6;
h_cc = 10000; v_cc = 360/3.6;

%% Caracteristicas do Mirage III
comp = 15;        % Comprimento (m)
b = 7.5;          % Envergadura (m)
c_bar = 5.25;     % Corda media aerodinamica (m)
S = 36;           % Area da asa (m^2)
m = 7400;         % Massa (kg)

% Matriz momento de inercia (kg*m^2) - PAGLIONE PAGINA 366 DO
     PDF
% Obs: aeronave massivamente simetrica em torno do plano xz
I_B = [9*10^4, 0, -1.8*10^3;
        0, 5.4*10^4, 0;
       -1.8*10^3, 0, 6*10^4];

% Constante da polar CL x CD
k = 0.4;

% Coeficientes aerodinamicos adimentionais
CD_0 = 0.015;
CY_beta = -0.6; CY_delta_r = 0.075; CY_delta_a = 0.01;
```

```
CL_0 = 0; CL_alpha = 2.204; CL_delta_e = 0.7; CL_q = 0;
Cl_beta = -0.05; Cl_delta_r = 0.018;  Cl_delta_a = -0.3; Cl_p = -0.25; Cl_r = 0.06;
Cm_0 = 0; Cm_alpha = -0.17; Cm_delta_e = -0.45; Cm_q = -0.4;
Cn_beta = 0.15; Cn_delta_r = -0.085; Cn_delta_a = 0; Cn_p = 0.055; Cn_r = -0.7;

%% Caracterestica do ambiente
g = 9.80665;       % Aceleracao da gravidade (m/s^2)
rho = 0.73;        % Densidade do ar a 10000 m (kg/m^3)

%% Condicao de voo
cond;

%% Encontrando o ponto de trimagem para o modelo nao linear
dt = 0.01;          % Passo de simulacao
t_final = 200;      % Tempo final de simulacao
modelo = 'mirage_III';
trim;

%% Incializa e simula o modelo com Filtro de Kalman Estendido
uiopen('.\FKE\fke_mirage_III.slx', 1)
init_fke;
sim_out_fke = sim("fke_mirage_III");

%% Plota e salva a trajetoria e os resultados
plot_resultados;
```

A.2 *cond.m*

```
if traj == 1
%% Condicao de voo reto e nivelado
    % Condicoes iniciais para a trimagem
    x0 = [0; 0; h_rn;                                  % x_E(0); y_E(0); z_E(0)
          0; 0; 0;                                     % phi(0); theta(0); psi(0)
          v_rn; 0; 0;                                  % u(0); v(0); w(0)
```

```
         0; 0; 0;                                   % p(0); q(0); r(0)
         CD_0;                                      % CD_0(0)
         CY_beta; CY_delta_r; CY_delta_a;           % CY_beta(0); CY_delta_r(0); CY_delta_a(0)
         CL_0; CL_alpha; CL_delta_e; CL_q;          % CL_0(0); CL_alpha(0); CL_delta_e(0); CL_q(0)
         Cl_beta; Cl_delta_r; Cl_delta_a; Cl_p; Cl_r;   % Cl_beta(0); Cl_delta_r(0); Cl_delta_a(0); Cl_p(0); Cl_r(0)
         Cm_0; Cm_alpha; Cm_delta_e; Cm_q;          % Cm_0(0); Cm_alpha(0); Cm_delta_e(0); Cm_q(0)
         Cn_beta; Cn_delta_r; Cn_delta_a; Cn_p; Cn_r];  % Cn_beta(0); Cn_delta_r(0); Cn_delta_a(0); Cn_p(0); Cn_r(0)

   % Condicao para os estados
   % Os estados x(1:12) nao sao conhecidos e a funcao calculara para a condicao
   % Obs: 0 -> estado nao conhecido. 1 -> estado conhecido
   estados_knwon = [0; 0; 0;                        % x_E; y_E; z_E
                    0; 0; 0;                        % phi; theta; psi
                    0; 0; 0;                        % u; v; w
                    0; 0; 0;                        % p; q; r
                    1;                              % CD_0
                    1; 1; 1;                        % CY_beta; CY_delta_r; CY_delta_a
                    1; 1; 1; 1;                     % CL_0; CL_alpha; CL_delta_e; CL_q
                    1; 1; 1; 1; 1;                  % Cl_beta; Cl_delta_r; Cl_delta_a; Cl_p; Cl_r
```

```
                     1; 1; 1; 1;                        % Cm_0; Cm_alpha; Cm_delta_e; Cm_q
                     1; 1; 1; 1; 1];                    % Cn_beta; Cn_delta_r; Cn_delta_a; Cn_p; Cn_r

    % Na condicao de voo reto e nivelado na direcao x com velocidade constante,
    % o unico estado livre para variar no tempo e x_E, logo x_E_dot =~ 0.
    % Para o restante dos estados a variacao no tempo deve ser igual a 0, logo:
    % Obs: 0 -> estado_dot =~ 0. 1 -> estado_dot = 0
    estados_estaveis = [0; 1; 1;                        % x_E; y_E; z_E
                        1; 1; 1;                        % phi; theta; psi
                        1; 1; 1;                        % u; v; w
                        1; 1; 1;                        % p; q; r
                        1;                              % CD_0
                        1; 1; 1;                        % CY_beta; CY_delta_r; CY_delta_a
                        1; 1; 1; 1;                     % CL_0; CL_alpha; CL_delta_e; CL_q
                        1; 1; 1; 1; 1;                  % Cl_beta; Cl_delta_r; Cl_delta_a; Cl_p; Cl_r
                        1; 1; 1; 1;                     % Cm_0; Cm_alpha; Cm_delta_e; Cm_q
                        1; 1; 1; 1; 1];                 % Cn_beta; Cn_delta_r; Cn_delta_a; Cn_p; Cn_r

    % Condicao para as saidas
    % As saidas alpha, theta e x nao sao conhecidas e a funcao calculara para a condicao
```

```
    % Obs: 0 -> estado nao conhecido. 1 -> estado conhecido
    saidas_knwon = [0; 1; 1;                                    % alpha; beta; V
                    1; 0; 1;                                    % phi; theta; psi
                    1; 1; 1;                                    % p; q; r
                    0; 1; 1];                                   % x_E; y_E; z_E

    % Palpite inicial para as saidas
    saidas = [0; 0; v_rn;                                       % alpha; beta; V
              0; 0; 0;                                          % phi; theta; psi
              0; 0; 0;                                          % p; q; r
              0; 0; h_rn];                                      % x_E; y_E; z_E

else
%% Condicao de curva coordenada
    % Condicoes iniciais para a trimagem
    omega = 0.0566;
    phi_cc = atan(v_cc*omega/g);
    x0 = [0; 0; h_cc;                                           % x_E(0); y_E(0); z_E(0)
          phi_cc; 0; 0;                                         % phi(0); theta(0); psi(0)
          v_cc; 0; 0;                                           % u(0); v(0); w(0)
          0; 0; 0;                                              % p(0); q(0); r(0)
          CD_0;                                                 % CD_0(0)
          CY_beta; CY_delta_r; CY_delta_a;                      % CY_beta(0); CY_delta_r(0); CY_delta_a(0)
          CL_0; CL_alpha; CL_delta_e; CL_q;                     % CL_0(0); CL_alpha(0); CL_delta_e(0); CL_q(0)
```

```
        Cl_beta; Cl_delta_r; Cl_delta_a; Cl_p; Cl_r;   % Cl_beta(0); Cl_delta_r(0); Cl_delta_a(0); Cl_p(0); Cl_r(0)
        Cm_0; Cm_alpha; Cm_delta_e; Cm_q;              % Cm_0(0); Cm_alpha(0); Cm_delta_e(0); Cm_q(0)
        Cn_beta; Cn_delta_r; Cn_delta_a; Cn_p; Cn_r];  % Cn_beta(0); Cn_delta_r(0); Cn_delta_a(0); Cn_p(0); Cn_r(0)

    % Condicao para os estados
    % Os estados x(1:2), x(4:6) e x(8:12) nao sao conhecidos e a funcao calculara para a condicao
    % Obs: 0 -> estado nao conhecido. 1 -> estado conhecido
    estados_knwon = [0; 0; 1;                          % x_E; y_E; z_E
                     1; 0; 0;                          % phi; theta; psi
                     1; 1; 0;                          % u; v; w
                     0; 0; 0;                          % p; q; r
                     1;                                % CD_0
                     1; 1; 1;                          % CY_beta; CY_delta_r; CY_delta_a
                     1; 1; 1; 1;                       % CL_0; CL_alpha; CL_delta_e; CL_q
                     1; 1; 1; 1; 1;                    % Cl_beta; Cl_delta_r; Cl_delta_a; Cl_p; Cl_r
                     1; 1; 1; 1;                       % Cm_0; Cm_alpha; Cm_delta_e; Cm_q
                     1; 1; 1; 1; 1];                   % Cn_beta; Cn_delta_r; Cn_delta_a; Cn_p; Cn_r

    % Na condicao de curva coordenada, os unicos estados livres para variar no tempo sao x_E, y_E e
    % phi, logo x_E_dot =~ 0, y_E_dot =~ 0 e phi_dot =~ 0
```

```
    % Para o restante dos estados a variacao no tempo deve ser igual a 0, logo:
    % Obs: 0 -> estado_dot =~ 0. 1 -> estado_dot = 0
    estados_estaveis = [0; 0; 1;                              % x_E; y_E; z_E
                        1; 1; 0;                              % phi; theta; psi
                        1; 1; 1;                              % u; v; w
                        1; 1; 1;                              % p; q; r
                        1;                                    % CD_0
                        1; 1; 1;                              % CY_beta; CY_delta_r; CY_delta_a
                        1; 1; 1; 1;                           % CL_0; CL_alpha; CL_delta_e; CL_q
                        1; 1; 1; 1; 1;                        % Cl_beta; Cl_delta_r; Cl_delta_a; Cl_p; Cl_r
                        1; 1; 1; 1;                           % Cm_0; Cm_alpha; Cm_delta_e; Cm_q
                        1; 1; 1; 1; 1];                       % Cn_beta; Cn_delta_r; Cn_delta_a; Cn_p; Cn_r

    % Condicao para as saidas
    % As saidas y(1), y(2:11) nao sao conhecidas e a funcao calculara para a condicao
    % Obs: 0 -> estado nao conhecido. 1 -> estado conhecido
    saidas_knwon = [0; 1; 0;                                  % alpha; beta; V
                    1; 0; 0;                                  % phi; theta; psi
                    0; 0; 0;                                  % p; q; r
                    0; 0; 1];                                 % x_E; y_E; z_E

```

```
    % Palpite inicial para as saidas
    saidas = [0; 0; v_cc;                                   % alpha; beta; V
              phi_cc; 0; 0;                                 % phi; theta; psi
              0; 0; 0;                                      % p; q; r
              0; 0; h_cc];                                  % x_E; y_E; z_E
end
```

A.3 *trim.m*

```
% Criando as especificacoes do ponto de operacao para o modelo Simulink
aux = operspec(modelo);

% Quais estados sao conhecidos
aux.States.Known = estados_knwon;

% Quais estados sao estaveis (quais estao em steady state)
aux.States.SteadyState = estados_estaveis;

% Quais saidas sao conhecidas
aux.Outputs.Known = saidas_knwon;

% Qual o palpite inicial das saidas
aux.Outputs.y = saidas;

% Encontra os pontos de operacao (condicoes de trimagem) de acordo com
% as especificacoes
[rn , rn_report] = findop(modelo, aux);

% Entradas calculadas a partir da trimagem
trim_input = rn.Inputs.u';

% Setado as condicoes iniciais como sendo os estados calculados pela
```

```
% trimagem
x0 = rn.States.x;
```

A.4 *init_fke*

```
%% Inicializacao do Filtro de Kalman Estendido
x0_fke = x0;

if traj == 1
    P0(1,1) = 5;
    P0(2,2) = 10;
    P0(3,3) = 10;
    P0(4,4) = 1;
    P0(5,5) = 10;
    P0(6,6) = 10;
    P0(7,7) = 10;
    P0(8,8) = 1;
    P0(9,9) = 10;
    P0(10,10) = 0.01;
    P0(11,11) = 0.01;
    P0(12,12) = 0.01;
    P0(13,13) = 0.01;
    P0(14,14) = 0.1;
    P0(15,15) = 0.1;
    P0(16,16) = 0.01;
    P0(17,17) = 0.01;
    P0(18,18) = 0.01;
    P0(19,19) = 0.01;
    P0(20,20) = 0.01;
    P0(21,21) = 0.01;
    P0(22,22) = 0.01;
    P0(23,23) = 0.01;
    P0(24,24) = 0.01;
    P0(25,25) = 0.01;
    P0(26,26) = 0.01;
    P0(27,27) = 0.01;
    P0(28,28) = 0.01;
    P0(29,29) = 0.01;
    P0(30,30) = 0.01;
```

```
    P0(31,31) = 0.01;
    P0(32,32) = 0.01;
    P0(33,33) = 0.01;
    P0(34,34) = 1;
else

    P0(1,1) = 10;
    P0(2,2) = 7.5;
    P0(3,3) = 7.5;
    P0(4,4) = 0.5;
    P0(5,5) = 1;
    P0(6,6) = 1;
    P0(7,7) = 500;
    P0(8,8) = 500;
    P0(9,9) = 500;
    P0(10,10) = 0.01;
    P0(11,11) = 0.01;
    P0(12,12) = 0.01;
    P0(13,13) = 10;
    P0(14,14) = 0.001;
    P0(15,15) = 0.1;
    P0(16,16) = 0.5;
    P0(17,17) = 10;
    P0(18,18) = 0.0001;
    P0(19,19) = 100;
    P0(20,20) = 0.001;
    P0(21,21) = 100;
    P0(22,22) = 50;
    P0(23,23) = 50;
    P0(24,24) = 0.0001;
    P0(25,25) = 0.001;
    P0(26,26) = 0.001;
    P0(27,27) = 0.01;
    P0(28,28) = 100;
    P0(29,29) = 0.001;
    P0(30,30) = 100;
    P0(31,31) = 100;
    P0(32,32) = 5;
    P0(33,33) = 0.01;
```

```
    P0(34,34) = 0.01;
end

%% Sensores
sig_alpha = 0.5; sig_beta = 0.5; sig_V = 0.1;
sig_phi = 0.1; sig_theta = 0.1; sig_psi = 0.1;
sig_p = 0.1; sig_q = 0.1; sig_r = 0.1;
sig_x_E = 10; sig_y_E = 10; sig_z_E = 10;

% Ruido de medicao
R = diag([sig_alpha^2, sig_beta^2, sig_V^2, ...
          sig_phi^2, sig_theta^2, sig_psi^2, ...
          sig_p^2, sig_q^2, sig_r^2, ...
          sig_x_E^2, sig_y_E^2, sig_z_E^2]);

% Seed das medidas (idendentifica quao randomico e cada
   medida)
seed_alpha = 23341; seed_beta = 23341; seed_V = 23341;
seed_phi = 23342; seed_theta = 23342; seed_psi = 23342;
seed_p = 23343; seed_q = 23343; seed_r = 23343;
seed_x_E = 23344; seed_y_E = 23344; seed_z_E = 23344;

seed = [seed_alpha; seed_beta; seed_V; ...
        seed_phi; seed_theta; seed_psi; ...
        seed_p; seed_q; seed_r; ...
        seed_x_E; seed_y_E; seed_z_E];

%% Ruido de modelo
Q(1,1) = 0.1;        % x_E
Q(2,2) = 0.1;        % y_E
Q(3,3) = 0.1;        % z_E
Q(4,4) = 0.01;       % phi
Q(5,5) = 0.01;       % theta
Q(6,6) = 0.01;       % psi
Q(7,7) = 0.01;       % u
Q(8,8) = 0.001;      % v
Q(9,9) = 0.1;        % w
Q(10,10) = 0.001;    % p
Q(11,11) = 0.001;    % q
```

```
Q(12,12) = 0.001;    % r
Q(13,13) = 0.0001;   % CD_0

cY = 0.00001;
Q(14,14) = 0.001;   % CY_beta
Q(15,15) = 100;   % CY_delta_r
Q(16,16) = cY;  % CY_delta_a

cL = 0.0001;
Q(17,17) = cL;  % CL_0
Q(18,18) = cL;  % CL_alpha
Q(19,19) = cL;  % CL_delta_e
Q(20,20) = cL;  % CL_q

cl = 0.0001;
Q(21,21) = cl;  % Cl_beta
Q(22,22) = cl;  % Cl_delta_r
Q(23,23) = cl;  % Cl_delta_a
Q(24,24) = cl;  % Cl_p
Q(25,25) = cl;  % Cl_r

cm = 0.0001;
Q(26,26) = cm;  % Cm_0
Q(27,27) = cm;  % Cm_alpha
Q(28,28) = cm;  % Cm_delta_e
Q(29,29) = cm;  % Cm_q

cn = 0.0001;
Q(30,30) = cn;  % Cn_beta
Q(31,31) = cn;  % Cn_delta_r
Q(32,32) = cn;  % Cn_delta_a
Q(33,33) = cn;  % Cn_p
Q(34,34) = cn;  % Cn_r
```

A.5 *plot_resultados.m*

```
time = sim_out_fke.tout;
P_fke = sim_out_fke.logsout{1}.Values.Data;
x_fke = sim_out_fke.logsout{2}.Values.Data;
```

```
x = zeros(length(time), 34);
y_m = zeros(length(time), 12);
sigma = zeros(length(time), 34);

for i = 1:34
    aux_x = sim_out_fke.logsout{4}.Values.Data;
    for j = 1:length(time)
        x(j,i) = aux_x(i,:,j);
    end
end

for i = 1:12
    aux_y_m = sim_out_fke.logsout{5}.Values.Data;
    for j = 1:length(time)
        y_m(j,i) = aux_y_m(i,:,j);
    end
end

for i = 1:34
    for j = 1:length(time)
        aux_sigma = P_fke(i,i,j);
        sigma(j,i) = sqrt(aux_sigma);
    end
end

%% Estados estimados
x_E_fke = x_fke(:,1); y_E_fke = x_fke(:,2); z_E_fke = x_fke
   (:,3);
phi_fke = x_fke(:,4); theta_fke = x_fke(:,5); psi_fke = x_fke
   (:,6);
u_fke = x_fke(:,7); v_fke = x_fke(:,8); w_fke = x_fke(:,9);
p_fke = x_fke(:,10); q_fke = x_fke(:,11); r_fke = x_fke(:,12)
   ;
CD_0_fke = x_fke(:,13);
CY_beta_fke = x_fke(:,14); CY_delta_r_fke = x_fke(:,15);
   CY_delta_a_fke = x_fke(:,16);
CL_0_fke = x_fke(:,17); CL_alpha_fke = x_fke(:,18);
   CL_delta_e_fke = x_fke(:,19); CL_q_fke = x_fke(:,20);
```

```
Cl_beta_fke = x_fke(:,21); Cl_delta_r_fke = x_fke(:,22); Cl_delta_a_fke = x_fke(:,23); Cl_p_fke = x_fke(:,24); Cl_r_fke = x_fke(:,25);
Cm_0_fke = x_fke(:,26); Cm_alpha_fke = x_fke(:,27); Cm_delta_e_fke = x_fke(:,28); Cm_q_fke = x_fke(:,29);
Cn_beta_fke = x_fke(:,30); Cn_delta_r_fke = x_fke(:,31); Cn_delta_a_fke = x_fke(:,32); Cn_p_fke = x_fke(:,33); Cn_r_fke = x_fke(:,34);

% Vetor de estados
x_E = x(:,1); y_E = x(:,2); z_E = x(:,3);
phi = x(:,4); theta = x(:,5); psi = x(:,6);
u = x(:,7); v = x(:,8); w = x(:,9);
p = x(:,10); q = x(:,11); r = x(:,12);
CD_0 = x(:,13);
CY_beta = x(:,14); CY_delta_r = x(:,15); CY_delta_a = x(:,16);
CL_0 = x(:,17); CL_alpha = x(:,18); CL_delta_e = x(:,19); CL_q = x(:,20);
Cl_beta = x(:,21); Cl_delta_r = x(:,22); Cl_delta_a = x(:,23); Cl_p = x(:,24); Cl_r = x(:,25);
Cm_0 = x(:,26); Cm_alpha = x(:,27); Cm_delta_e = x(:,28); Cm_q = x(:,29);
Cn_beta = x(:,30); Cn_delta_r = x(:,31); Cn_delta_a = x(:,32); Cn_p = x(:,33); Cn_r = x(:,34);

% Saidas dos sensores
alpha_m = y_m(:,1); beta_m = y_m(:,2); V_m = y_m(:,3);
phi_m = y_m(:,4); theta_m = y_m(:,5); psi_m = y_m(:,6);
p_m = y_m(:,7); q_m = y_m(:,8); r_m = y_m(:,9);
x_E_m = y_m(:,10); y_E_m = y_m(:,11); z_E_m = y_m(:,12);

% Desvio padrao
sigma_x_E = sigma(:,1); sigma_y_E = sigma(:,2); sigma_z_E = sigma(:,3);
sigma_phi = sigma(:,4); sigma_theta = sigma(:,5); sigma_psi = sigma(:,6);
sigma_u = sigma(:,7); sigma_v = sigma(:,8); sigma_w = sigma(:,9);
```

```
sigma_p = sigma(:,10); sigma_q = sigma(:,11); sigma_r = sigma(:,12);
sigma_CD_0 = sigma(:,13);
sigma_CY_beta = sigma(:,14); sigma_CY_delta_r = sigma(:,15); sigma_CY_delta_a = sigma(:,16);
sigma_CL_0 = sigma(:,17); sigma_CL_alpha = sigma(:,18); sigma_CL_delta_e = sigma(:,19); sigma_CL_q = sigma(:,20);
sigma_Cl_beta = sigma(:,21); sigma_Cl_delta_r = sigma(:,22); sigma_Cl_delta_a = sigma(:,23); sigma_Cl_p = sigma(:,24); sigma_Cl_r = sigma(:,25);
sigma_Cm_0 = sigma(:,26); sigma_Cm_alpha = sigma(:,27); sigma_Cm_delta_e = sigma(:,28); sigma_Cm_q = sigma(:,29);
sigma_Cn_beta = sigma(:,30); sigma_Cn_delta_r = sigma(:,31); sigma_Cn_delta_a = sigma(:,32); sigma_Cn_p = sigma(:,33); sigma_Cn_r = sigma(:,34);

%% Plotagem

% Trajetoria
f(1) = figure;
f.WindowState = 'maximized';
pl = plot3(x_E, round(y_E), round(z_E), 'b');
pl.LineWidth = 1.5;
grid on;
xlabel('$x_E\ (m)$', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$y_E\ (m)$', 'Interpreter', 'latex', 'FontSize', 30);
zlabel('$z_E\ (m)$', 'Interpreter', 'latex', 'FontSize', 30);
if traj == 1
    legend('Trajetoria - voo reto e nivelado', 'FontSize', 14, 'Location', 'northeast');
    saveas(f, '.\Plots\Reto nivelado\trajetoria_rn.png');
else
    legend('Trajetoria - curva coordenada', 'FontSize', 14, 'Location', 'northeast');
    saveas(f, '.\Plots\Reto nivelado\trajetoria_cc.png');
end
ax = gca; ax.FontSize = 18;

%% Estado vs estado estimado vs medida dos sensores
```

```

%% x_E
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [x_E_m, x_E_fke, x_E]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$x_E \ (m)$', 'Interpreter', 'latex', 'FontSize', 30);
legend('$x_{E_{m}}$ (medido)', '$x_{E_{fke}}$ (FKE)', '$x_E$ (real)','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\x_E_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\x_E_FKE_cc.png');
end

%% y_E
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [y_E_m, y_E_fke, y_E]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$y_E \ (m)$', 'Interpreter', 'latex', 'FontSize', 30);
legend('$y_{E_{m}}$ (medido)', '$y_{E_{fke}}$ (FKE)', '$y_E$ (real)','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\y_E_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\y_E_FKE_cc.png');
```

```
end

%% z_E
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [z_E_m, z_E_fke, z_E]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$z_E \ (m)$', 'Interpreter', 'latex', 'FontSize', 30)
    ;
legend('$z_{E_{m}}$ (medido)', '$z_{E_{fke}}$ (FKE)', '$z_E$
    (real)','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\z_E_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\z_E_FKE_cc.png');
end

%% phi
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [phi_m, phi_fke, phi]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$\phi \ (rad)$', 'Interpreter', 'latex', 'FontSize',
    30);
legend('$\phi_{m}$ (medido)', '$\phi_{fke}$ (FKE)', '$\phi$ (
    real)','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\phi_FKE_rn.png');
else
```

```
    saveas(f, '.\Plots\Curva coordenada\phi_FKE_cc.png');
end

%% theta
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [theta_m, theta_fke, theta]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$\theta \ (rad)$', 'Interpreter', 'latex', 'FontSize'
    , 30);
legend('$\theta_{m}$ (medido)', '$\theta_{fke}$ (FKE)', '$\
    theta$ (real)','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\theta_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\theta_FKE_cc.png');
end

%% psi
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [psi_m, psi_fke, psi]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$\psi \ (rad)$', 'Interpreter', 'latex', 'FontSize',
    30);
legend('$\psi_{m}$ (medido)', '$\psi_{fke}$ (FKE)', '$\psi$ (
    real)','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\psi_FKE_rn.png');
```

```
else
    saveas(f, '.\Plots\Curva coordenada\psi_FKE_cc.png');
end

%% u
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [u_fke, u]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$u \ (m/s)$', 'Interpreter', 'latex', 'FontSize', 30);
legend('$u_{fke}$ (FKE)', '$u$ (real)','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\u_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\u_FKE_cc.png');
end

%% v
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [v_fke, v]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$v \ (m/s)$', 'Interpreter', 'latex', 'FontSize', 30);
legend('$v_{fke}$ (FKE)', '$v$ (real)','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\v_FKE_rn.png');
else
```

```
    saveas(f, '.\Plots\Curva coordenada\v_FKE_cc.png');
end

%% w
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [w_fke, w]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$w \ (m/s)$', 'Interpreter', 'latex', 'FontSize', 30)
    ;
legend('$w_{fke}$ (FKE)', '$w$ (real)','Interpreter', 'latex'
    , 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\w_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\w_FKE_cc.png');
end

%% p
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [p_m, p_fke, p]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$p \ (rad/s)$', 'Interpreter', 'latex', 'FontSize',
    30);
legend('$p_{m}$ (medido)', '$p_{fke}$ (FKE)', '$p$ (real)','
    Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\p_FKE_rn.png');
else
```

```
    saveas(f, '.\Plots\Curva coordenada\p_FKE_cc.png');
end

%% q
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [q_m, q_fke, q]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$q \ (rad/s)$', 'Interpreter', 'latex', 'FontSize',
   30);
legend('$q_{m}$ (medido)', '$q_{fke}$ (FKE)', '$q$ (real)','
   Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\q_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\q_FKE_cc.png');
end

%% r
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [r_m, r_fke, r]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$r \ (rad/s)$', 'Interpreter', 'latex', 'FontSize',
   30);
legend('$r_{m}$ (medido)', '$r_{fke}$ (FKE)', '$r$ (real)','
   Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\r_FKE_rn.png');
```

```
else
    saveas(f, '.\Plots\Curva coordenada\r_FKE_cc.png');
end

%% CD_0
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [CD_0_fke, CD_0]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$CD_0$', 'Interpreter', 'latex', 'FontSize', 30);
legend('$CD_{0_{fke}}$ (FKE)', '$CD_0$ (real)','Interpreter',
    'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\CD_0_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\CD_0_FKE_cc.png');
end

%% CY_beta
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [CY_beta_fke, CY_beta]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$CY_{\beta}$', 'Interpreter', 'latex', 'FontSize',
   30);
legend('$CY_{\beta_{fke}}$ (FKE)', '$CY_{\beta}$ (real)','
   Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\CY_beta_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\CY_beta_FKE_cc.png');
```

```
end

%% CY_delta_r
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [CY_delta_r_fke, CY_delta_r]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$CY_{\delta_{r}}$', 'Interpreter', 'latex', 'FontSize
    ', 30);
legend('$CY_{\delta_{r_{fke}}}$ (FKE)', '$CY_{\delta_{r}}$ (
    real)','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\CY_delta_r_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\CY_delta_r_FKE_cc.png
       ');
end

%% CY_delta_a
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [CY_delta_a_fke, CY_delta_a]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$CY_{\delta_{a}}$', 'Interpreter', 'latex', 'FontSize
    ', 30);
legend('$CY_{\delta_{a_{fke}}}$ (FKE)', '$CY_{\delta_{a}}$ (
    real)','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\CY_delta_a_FKE_rn.png');
else
```

```
    saveas(f, '.\Plots\Curva coordenada\CY_delta_a_FKE_cc.png');
end

%% CL_0
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [CL_0_fke, CL_0]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$CL_0$', 'Interpreter', 'latex', 'FontSize', 30);
legend('$CL_{0_{fke}}$ (FKE)', '$CL_0$ (real)','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\CL_0_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\CL_0_FKE_cc.png');
end

%% CL_alpha
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [CL_alpha_fke, CL_alpha]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$CL_{\alpha}$', 'Interpreter', 'latex', 'FontSize', 30);
legend('$CL_{\alpha_{fke}}$ (FKE)', '$CL_{\alpha}$ (real)','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\CL_alpha_FKE_rn.png');
else
```

```
    saveas(f, '.\Plots\Curva coordenada\CL_alpha_FKE_cc.png')
        ;
end

%% CL_delta_e
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [CL_delta_e_fke, CL_delta_e]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$CL_{\delta_e}$', 'Interpreter', 'latex', 'FontSize',
    30);
legend('$CL_{\delta_{e_{fke}}}$ (FKE)', '$CL_{\delta_e}$ (
   real)','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\CL_delta_e_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\CL_delta_e_FKE_cc.png
        ');
end

%% CL_q
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [CL_q_fke, CL_q]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$CL_{q}$', 'Interpreter', 'latex', 'FontSize', 30);
legend('$CL_{q_{fke}}$ (FKE)', '$CL_q$ (real)','Interpreter',
    'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\CL_q_FKE_rn.png');
else
```

```
    saveas(f, '.\Plots\Curva coordenada\CL_q_FKE_cc.png');
end

%% Cl_beta
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [Cl_beta_fke, Cl_beta]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$Cl_{\beta}$', 'Interpreter', 'latex', 'FontSize',
   30);
legend('$Cl_{\beta_{fke}}$ (FKE)', '$Cl_{\beta}$ (real)','
   Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\Cl_beta_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\Cl_beta_FKE_cc.png');
end

%% Cl_delta_r
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [Cl_delta_r_fke, Cl_delta_r]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$Cl_{\delta_r}$', 'Interpreter', 'latex', 'FontSize',
    30);
legend('$Cl_{\delta_{r_{fke}}}$ (FKE)', '$Cl_{\delta_r}$ (
   real)','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\Cl_delta_r_FKE_rn.png');
else
```

```
    saveas(f, '.\Plots\Curva coordenada\Cl_delta_r_FKE_cc.png
        ');
end

%% Cl_delta_a
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [Cl_delta_a_fke, Cl_delta_a]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$Cl_{\delta_a}$', 'Interpreter', 'latex', 'FontSize',
    30);
legend('$Cl_{\delta_{a_{fke}}}$ (FKE)', '$Cl_{\delta_a}$ (
   real)','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\Cl_delta_a_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\Cl_delta_a_FKE_cc.png
        ');
end

%% Cl_p
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [Cl_p_fke, Cl_p]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$Cl_p$', 'Interpreter', 'latex', 'FontSize', 30);
legend('$Cl_{p_{fke}}$ (FKE)', '$Cl_p$ (real)','Interpreter',
    'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\Cl_p_FKE_rn.png');
else
```

```
    saveas(f, '.\Plots\Curva coordenada\Cl_p_FKE_cc.png');
end

%% Cl_r
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [Cl_r_fke, Cl_r]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$Cl_r$', 'Interpreter', 'latex', 'FontSize', 30);
legend('$Cl_{r_{fke}}$ (FKE)', '$Cl_r$ (real)','Interpreter',
    'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\Cl_r_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\Cl_r_FKE_cc.png');
end

%% Cm_0
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [Cm_0_fke, Cm_0]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$Cm_0$', 'Interpreter', 'latex', 'FontSize', 30);
legend('$Cm_{0_{fke}}$ (FKE)', '$Cm_0$ (real)','Interpreter',
    'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\Cm_0_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\Cm_0_FKE_cc.png');
end

```

```
%% Cm_alpha
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [Cm_alpha_fke, Cm_alpha]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$Cm_{\alpha}$', 'Interpreter', 'latex', 'FontSize',
    30);
legend('$Cm_{\alpha_{fke}}$ (FKE)', '$Cm_{\alpha}$ (real)','
    Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\Cm_alpha_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\Cm_alpha_FKE_cc.png')
        ;
end

%% Cm_delta_e
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [Cm_delta_e_fke, Cm_delta_e]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$Cm_{\delta_e}$', 'Interpreter', 'latex', 'FontSize',
     30);
legend('$Cm_{\delta_{e_{fke}}}$ (FKE)', '$Cm_{\delta_e}$ (
    real)','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\Cm_delta_e_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\Cm_delta_e_FKE_cc.png
        ');
end
```

```

%% Cm_q
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [Cm_q_fke, Cm_q]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$Cm_q$', 'Interpreter', 'latex', 'FontSize', 30);
legend('$Cm_{q_{fke}}$ (FKE)', '$Cm_q$ (real)','Interpreter',
    'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\Cm_q_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\Cm_q_FKE_cc.png');
end

%% Cn_beta
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [Cn_beta_fke, Cn_beta]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$Cn_{\beta}$', 'Interpreter', 'latex', 'FontSize',
   30);
legend('$Cn_{\beta_{fke}}$ (FKE)', '$Cn_{\beta}$ (real)','
   Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\Cn_beta_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\Cn_beta_FKE_cc.png');
end

%% Cn_delta_r
```

```
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [Cn_delta_r_fke, Cn_delta_r]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$Cn_{\delta_r}$', 'Interpreter', 'latex', 'FontSize',
    30);
legend('$Cn_{\delta_{r_{fke}}}$ (FKE)', '$Cn_{\delta_r}$ (
   real)','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\Cn_delta_r_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\Cn_delta_r_FKE_cc.png
       ');
end

%% Cn_delta_a
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [Cn_delta_a_fke, Cn_delta_a]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$Cn_{\delta_a}$', 'Interpreter', 'latex', 'FontSize',
    30);
legend('$Cn_{\delta_{a_{fke}}}$ (FKE)', '$Cn_{\delta_a}$ (
   real)','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\Cn_delta_a_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\Cn_delta_a_FKE_cc.png
       ');
end

```

```
%% Cn_p
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [Cn_p_fke, Cn_p]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$Cn_p$', 'Interpreter', 'latex', 'FontSize', 30);
legend('$Cn_{p_{fke}}$ (FKE)', '$Cn_p$ (real)','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\Cn_p_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\Cn_p_FKE_cc.png');
end

%% Cn_r
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [Cn_r_fke, Cn_r]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
ylabel('$Cn_r$', 'Interpreter', 'latex', 'FontSize', 30);
legend('$Cn_{r_{fke}}$ (FKE)', '$Cn_r$ (real)','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\Cn_r_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\Cn_r_FKE_cc.png');
end

%% Erro e desvio padrao

%% x_E
```

```
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_x_E, x_E - x_E_fke, -3*sigma_x_E])
    ;
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{x_E}$', '$\ x_E - x_{E_{fke}}$ (erro)',
    '$\ -3\sigma_{x_E}$','Interpreter', 'latex',  'FontSize',
    30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_x_E_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_x_E_FKE_cc.png')
        ;
end

%% y_E
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_y_E, y_E - y_E_fke, -3*sigma_y_E])
    ;
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{y_E}$', '$\ y_E - y_{E_{fke}}$ (erro)',
    '$\ -3\sigma_{y_E}$','Interpreter', 'latex',  'FontSize',
    30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_y_E_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_y_E_FKE_cc.png')
        ;
```

```
end

%% z_E
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_z_E, z_E - z_E_fke, -3*sigma_z_E]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{z_E}$', '$\ z_E - z_{E_{fke}}$ (erro)', '$\ -3\sigma_{z_E}$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_z_E_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_z_E_FKE_cc.png');
end

%% phi
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_phi, phi - phi_fke, -3*sigma_phi]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{\phi}$', '$\ \phi - \phi_{fke}$ (erro)', '$\ -3\sigma_{\phi}$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_phi_FKE_rn.png');
```

```
else
    saveas(f, '.\Plots\Curva coordenada\erro_phi_FKE_cc.png');
end

%% theta
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_theta, theta - theta_fke, -3*sigma_theta]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{\theta}$', '$\ \theta - \theta_{fke}$ (erro)', '$\ -3\sigma_{\theta}$','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_theta_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_theta_FKE_cc.png');
end

%% psi
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_psi, psi - psi_fke, -3*sigma_psi]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{\psi}$', '$\ \psi - \psi_{fke}$ (erro)', '$\ -3\sigma_{\psi}$','Interpreter', 'latex', 'FontSize', 30);
```

```
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_psi_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_psi_FKE_cc.png')
        ;
end

%% u
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_u, u - u_fke, -3*sigma_u]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_u$', '$\ u - u_{fke}$ (erro)', '$\ -3\
   sigma_u$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_u_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_u_FKE_cc.png');
end

%% v
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_v, v - v_fke, -3*sigma_v]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_v$', '$\ v - v_{fke}$ (erro)', '$\ -3\
   sigma_v$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
```

```
    saveas(f, '.\Plots\Reto nivelado\erro_v_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_v_FKE_cc.png');
end

%% w
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_w, w - w_fke, -3*sigma_w]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_w$', '$\ w - w_{fke}$ (erro)', '$\ -3\
    sigma_w$','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_w_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_w_FKE_cc.png');
end

%% p
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_p, p - p_fke, -3*sigma_p]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_p$', '$\ p - p_{fke}$ (erro)', '$\ -3\
    sigma_p$','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_p_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_p_FKE_cc.png');
```

```
end

%% q
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_q, q - q_fke, -3*sigma_q]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_q$', '$\ q - q_{fke}$ (erro)', '$\ -3\
    sigma_q$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_q_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_q_FKE_cc.png');
end

%% r
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_r, r - r_fke, -3*sigma_r]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_r$', '$\ r - r_{fke}$ (erro)', '$\ -3\
    sigma_r$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_r_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_r_FKE_cc.png');
end

%% CD_0
```

```
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_CD_0, CD_0 - CD_0_fke, -3*sigma_CD_0]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{D_0}}$', '$\ C_{D_0} - C_{D_{0_{fke}}}$ (erro)', '$\ -3\sigma_{C_{D_0}}$','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_CD_0_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_CD_0_FKE_cc.png');
end

%% CY_beta
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_CY_beta, CY_beta - CY_beta_fke, -3*sigma_CY_beta]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{Y_\beta}}$', '$\ C_{Y_\beta} - C_{Y_{\beta_{fke}}}$ (erro)', '$\ -3\sigma_{C_{Y_\beta}}$','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_CY_beta_FKE_rn.png');
else
```

```
    saveas(f, '.\Plots\Curva coordenada\erro_CY_beta_FKE_cc.
        png');
end

%% CY_delta_r
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_CY_delta_r, CY_delta_r -
    CY_delta_r_fke, -3*sigma_CY_delta_r]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{Y_{\delta_r}}}$', '$\ C_{Y_{\delta_r
    }} - C_{Y_{\delta_{r_{fke}}}}$ (erro)', '$\ -3\sigma_{C_{
    Y_{\delta_r}}}$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_CY_delta_r_FKE_rn.
        png');
else
    saveas(f, '.\Plots\Curva coordenada\
        erro_CY_delta_r_FKE_cc.png');
end

%% CY_delta_a
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_CY_delta_a, CY_delta_a -
    CY_delta_a_fke, -3*sigma_CY_delta_a]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{Y_{\delta_a}}}$', '$\ C_{Y_{\delta_a
    }} - C_{Y_{\delta_{a_{fke}}}}$ (erro)', '$\ -3\sigma_{C_{
    Y_{\delta_a}}}$','Interpreter', 'latex',  'FontSize', 30);
```

```
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_CY_delta_a_FKE_rn.
       png');
else
    saveas(f, '.\Plots\Curva coordenada\
       erro_CY_delta_a_FKE_cc.png');
end

%% CL_0
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_CL_0, CL_0 - CL_0_fke, -3*
   sigma_CL_0]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{L_0}}$', '$\ C_{L_0} - C_{L_{0_{fke
   }}}$ (erro)', '$\ -3\sigma_{C_{L_0}}$','Interpreter', '
   latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_CL_0_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_CL_0_FKE_cc.png'
       );
end

%% CL_alpha
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_CL_alpha, CL_alpha - CL_alpha_fke,
    -3*sigma_CL_alpha]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
```

```
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{L_\alpha}}$', '$\ C_{L_\alpha} - C_{L_{\alpha_{fke}}}$ (erro)', '$\ -3\sigma_{C_{L_\alpha}}$', 'Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_CL_alpha_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_CL_alpha_FKE_cc.png');
end

%% CL_delta_e
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_CL_delta_e, CL_delta_e - CL_delta_e_fke, -3*sigma_CL_delta_e]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{L_{\delta_e}}}$', '$\ C_{L_{\delta_e}} - C_{L_{\delta_{e_{fke}}}}$ (erro)', '$\ -3\sigma_{C_{L_{\delta_e}}}$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_CL_delta_e_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_CL_delta_e_FKE_cc.png');
end

%% CL_q
f(1) = figure;
f.WindowState = 'maximized';
```

```
pl = stairs(time, [3*sigma_CL_q, CL_q - CL_q_fke, -3*sigma_CL_q]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{L_q}}$', '$\ C_{L_q} - C_{L_{q_{fke}}}$ (erro)', '$\ -3\sigma_{C_{L_q}}$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_CL_q_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_CL_q_FKE_cc.png');
end

%% Cl_beta
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_Cl_beta, Cl_beta - Cl_beta_fke, -3*sigma_Cl_beta]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{l_\beta}}$', '$\ C_{l_\beta} - C_{L_{\beta_{fke}}}$ (erro)', '$\ -3\sigma_{C_{l_\beta}}$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_Cl_beta_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_Cl_beta_FKE_cc.png');
end
```

```

%% Cl_delta_r
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_Cl_delta_r, Cl_delta_r -
    Cl_delta_r_fke, -3*sigma_Cl_delta_r]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{l_{\delta_r}}}$', '$\ C_{l_{\delta_r
    }} - C_{l_{\delta_{r_{fke}}}}$ (erro)', '$\ -3\sigma_{C_{
    l_{\delta_e}}}$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_Cl_delta_r_FKE_rn.
        png');
else
    saveas(f, '.\Plots\Curva coordenada\
        erro_Cl_delta_r_FKE_cc.png');
end

%% Cl_delta_a
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_Cl_delta_a, Cl_delta_a -
    Cl_delta_a_fke, -3*sigma_Cl_delta_a]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{l_{\delta_a}}}$', '$\ C_{l_{\delta_a
    }} - C_{l_{\delta_{a_{fke}}}}$ (erro)', '$\ -3\sigma_{C_{
    l_{\delta_a}}}$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
```

```
    saveas(f, '.\Plots\Reto nivelado\erro_Cl_delta_a_FKE_rn.
       png');
else
    saveas(f, '.\Plots\Curva coordenada\
       erro_Cl_delta_a_FKE_cc.png');
end

%% Cl_p
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_Cl_p, Cl_p - Cl_p_fke, -3*
   sigma_Cl_p]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{l_p}}$', '$\ C_{l_p} - C_{l_{p_{fke
   }}}$ (erro)', '$\ -3\sigma_{C_{l_p}}$','Interpreter', '
   latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_Cl_p_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_Cl_p_FKE_cc.png'
       );
end

%% Cl_r
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_Cl_r, Cl_r - Cl_r_fke, -3*
   sigma_Cl_r]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
```

```
legend('$\ +3\sigma_{C_{l_r}}$', '$\ C_{l_r} - C_{l_{r_{fke}}}$ (erro)', '$\ -3\sigma_{C_{l_r}}$','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_Cl_r_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_Cl_r_FKE_cc.png');
end

%% Cm_0
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_Cm_0, Cm_0 - Cm_0_fke, -3*sigma_Cm_0]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{m_0}}$', '$\ C_{m_0} - C_{m_{0_{fke}}}$ (erro)', '$\ -3\sigma_{C_{m_0}}$','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_Cm_0_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_Cm_0_FKE_cc.png');
end

%% Cm_alpha
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_Cm_alpha, Cm_alpha - Cm_alpha_fke, -3*sigma_Cm_alpha]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
```

```
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{m_\alpha}}$', '$\ C_{m_\alpha} - C_{
    m_{\alpha_{fke}}}$ (erro)', '$\ -3\sigma_{C_{m_\alpha}}$',
    'Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_Cm_alpha_FKE_rn.png
        ');
else
    saveas(f, '.\Plots\Curva coordenada\erro_Cm_alpha_FKE_cc.
        png');
end

%% Cm_delta_e
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_Cm_delta_e, Cm_delta_e -
    Cm_delta_e_fke, -3*sigma_Cm_delta_e]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{m_{\delta_e}}}$', '$\ C_{m_{\delta_e
    }} - C_{m_{\delta_{e_{fke}}}}$ (erro)', '$\ -3\sigma_{C_{
    m_{\delta_e}}}$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_Cm_delta_e_FKE_rn.
        png');
else
    saveas(f, '.\Plots\Curva coordenada\
        erro_Cm_delta_e_FKE_cc.png');
end

%% Cm_q
f(1) = figure;
```

```
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_Cm_q, Cm_q - Cm_q_fke, -3*sigma_Cm_q]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{m_q}}$', '$\ C_{m_q} - C_{m_{q_{fke}}}$ (erro)', '$\ -3\sigma_{C_q}$','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_Cm_q_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_Cm_q_FKE_cc.png');
end

%% Cn_beta
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_Cn_beta, Cn_beta - Cn_beta_fke, -3*sigma_Cn_beta]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{n_\beta}}$', '$\ C_{n_\beta} - C_{n_{\beta_{fke}}}$ (erro)', '$\ -3\sigma_{C_{n_\beta}}$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_Cn_beta_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_Cn_beta_FKE_cc.png');
```

```
end

%% Cn_delta_r
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_Cn_delta_r, Cn_delta_r -
    Cn_delta_r_fke, -3*sigma_Cn_delta_r]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{n_{\delta_r}}}$', '$\ C_{n_{\delta_r
    }} - C_{n_{\delta_{r_{fke}}}}$ (erro)', '$\ -3\sigma_{C_{
    n_{\delta_r}}}$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_Cn_delta_r_FKE_rn.
        png');
else
    saveas(f, '.\Plots\Curva coordenada\
        erro_Cn_delta_r_FKE_cc.png');
end

%% Cn_delta_a
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_Cn_delta_a, Cn_delta_a -
    Cn_delta_a_fke, -3*sigma_Cn_delta_a]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{n_{\delta_a}}}$', '$\ C_{n_{\delta_a
    }} - C_{n_{\delta_{a_{fke}}}}$ (erro)', '$\ -3\sigma_{C_{
    n_{\delta_a}}}$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
```

```
    saveas(f, '.\Plots\Reto nivelado\erro_Cn_Delta_a_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_Cn_delta_a_FKE_cc.png');
end

%% Cn_p
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_Cn_p, Cn_p - Cn_p_fke, -3*sigma_Cn_p]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
legend('$\ +3\sigma_{C_{n_p}}$', '$\ C_{n_p} - C_{n_{p_{fke}}}$ (erro)', '$\ -3\sigma_{C_{n_p}}$','Interpreter', 'latex', 'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_Cn_p_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_Cn_p_FKE_cc.png');
end

%% Cn_r
f(1) = figure;
f.WindowState = 'maximized';
pl = stairs(time, [3*sigma_Cn_r, Cn_r - Cn_r_fke, -3*sigma_Cn_r]);
pl(1).LineWidth = 1.5;
pl(2).LineWidth = 1.5;
pl(3).LineWidth = 1.5;
grid on;
xlabel('Tempo (s)', 'Interpreter', 'latex', 'FontSize', 30);
```

```
legend('$\ +3\sigma_{C_{n_r}}$', '$\ C_{n_r} - C_{n_{r_{fke}}}$ (erro)', '$\ -3\sigma_{C_{n_r}}$','Interpreter', 'latex',  'FontSize', 30);
ax = gca; ax.FontSize = 18;
if traj == 1
    saveas(f, '.\Plots\Reto nivelado\erro_Cn_r_FKE_rn.png');
else
    saveas(f, '.\Plots\Curva coordenada\erro_Cn_r_FKE_cc.png');
end

%% Fecha todas as figuras
close all;
```

APÊNDICE B – Modelos Simulink®

O subsistema **MODELO NAO LINEAR - MIRAGE III** da Fig. 12 é dado por:

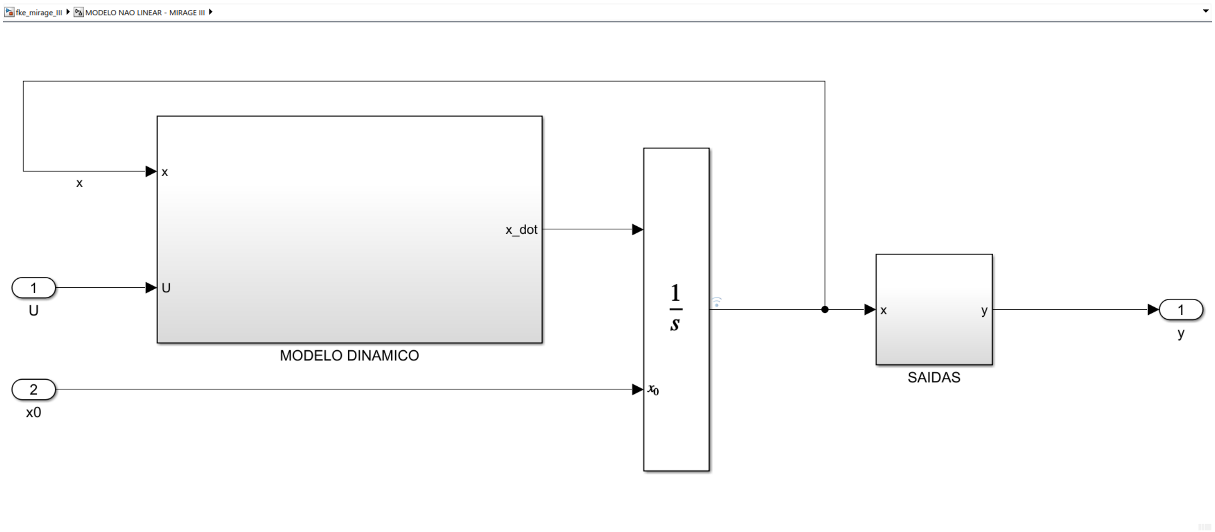


Figura 65 – Subsistemas MODELO NAO LINEAR - MIRAGE III.

B.1 Subsistema - Modelo Dinâmico

O subsistema **MODELO DINAMICO** da Fig. 65 é dado por:

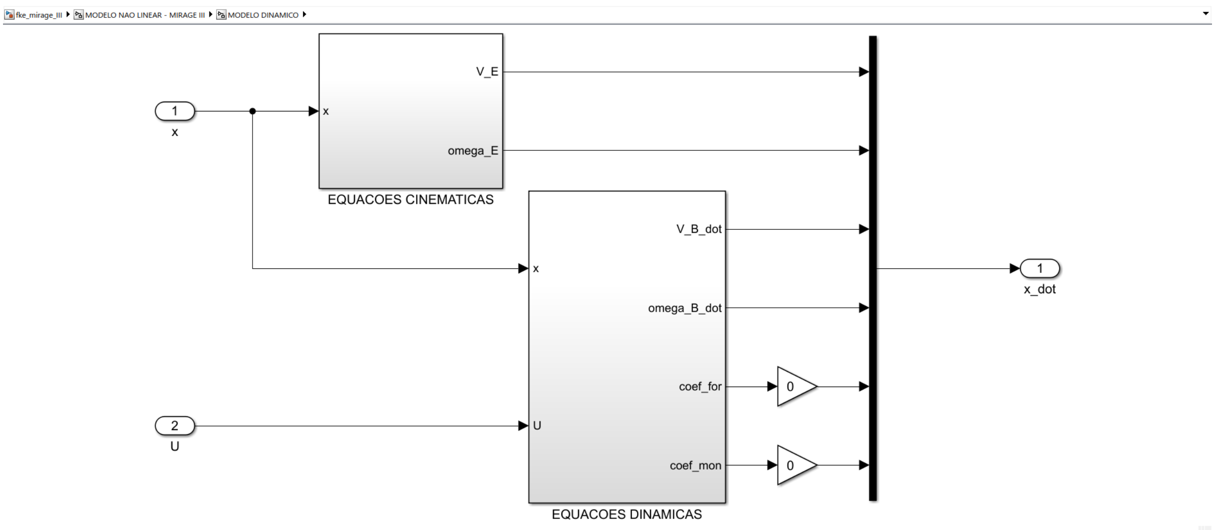


Figura 66 – Subsistema MODELO DINAMICO.

O subsistema **EQUACOES CINEMATICAS** é dado por:

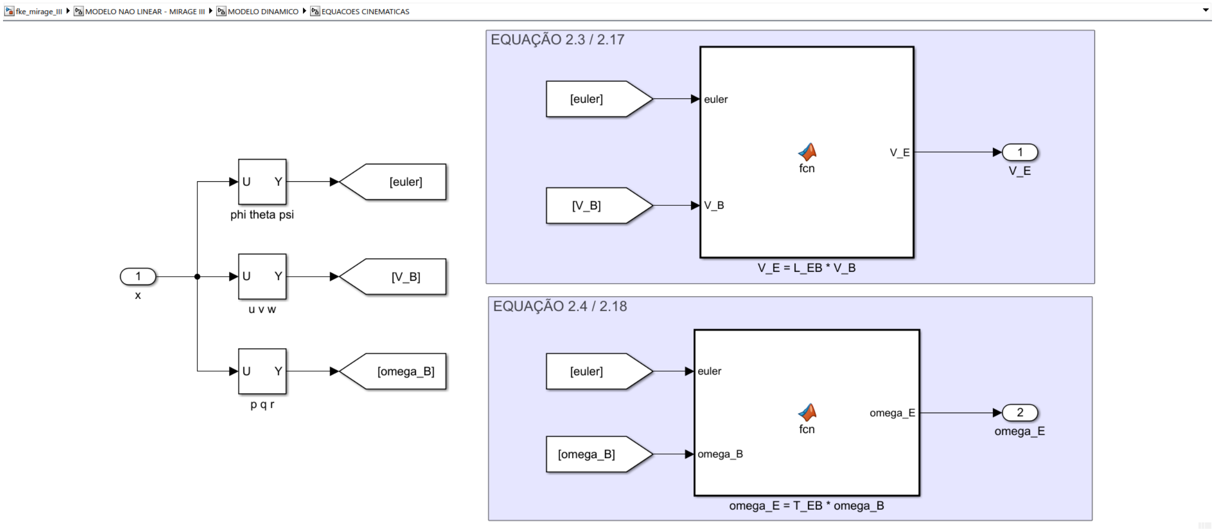


Figura 67 – Subsistema EQUACOES CINEMATICAS.

Onde a função $V_E = L_{EB} * V_B$ é dada por:

```
function V_E = fcn(euler, V_B)

phi = euler(1);
theta = euler(2);
psi = euler(3);

 % EQUACAO 2.5
 L_EB = [cos(theta)*cos(psi), sin(phi)*sin(theta)*cos(psi) -
    cos(phi)*sin(psi), cos(phi)*sin(theta)*cos(psi) + sin(phi
    )*sin(psi);
         cos(theta)*sin(psi), sin(phi)*sin(theta)*sin(psi) +
            cos(phi)*cos(psi), cos(phi)*sin(theta)*sin(psi) -
            sin(phi)*cos(psi)
         -sin(theta), sin(phi)*cos(theta), cos(phi)*cos(theta)
            ];

 V_E = L_EB * V_B;
```

E a função $omega_E = T_{EB} * omega_B$ é dada por:

```
function omega_E = fcn(euler, omega_B)

phi = euler(1);
theta = euler(2);
```

```
% EQUACAO 2.6
T_EB = [1, sin(phi)*tan(theta), cos(phi)*tan(theta);
        0, cos(phi), -sin(phi);
        0, sin(phi)*sec(theta), cos(phi)*sec(theta)];

omega_E = T_EB * omega_B;
```

O subsistema **EQUACOES DINAMICAS** da Fig. 66 é dado por:

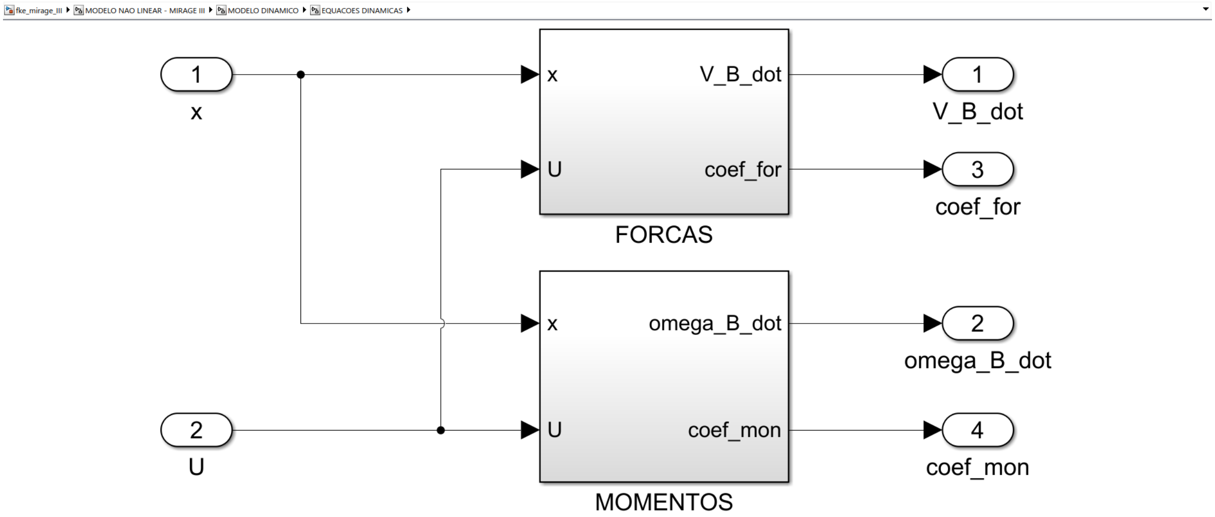


Figura 68 – Subsistema EQUACOES DINAMICAS.

Onde o subsistema **FORCAS** é dado por:

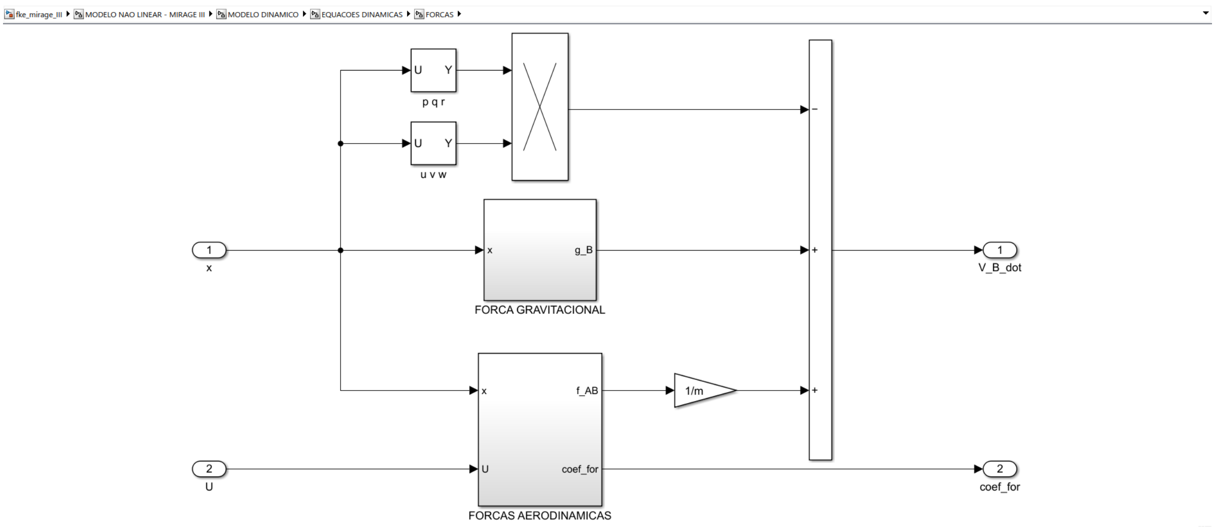


Figura 69 – Subsistema FORCAS.

Onde o subsistema **FORCA GRAVITACIONAL** é dado por:

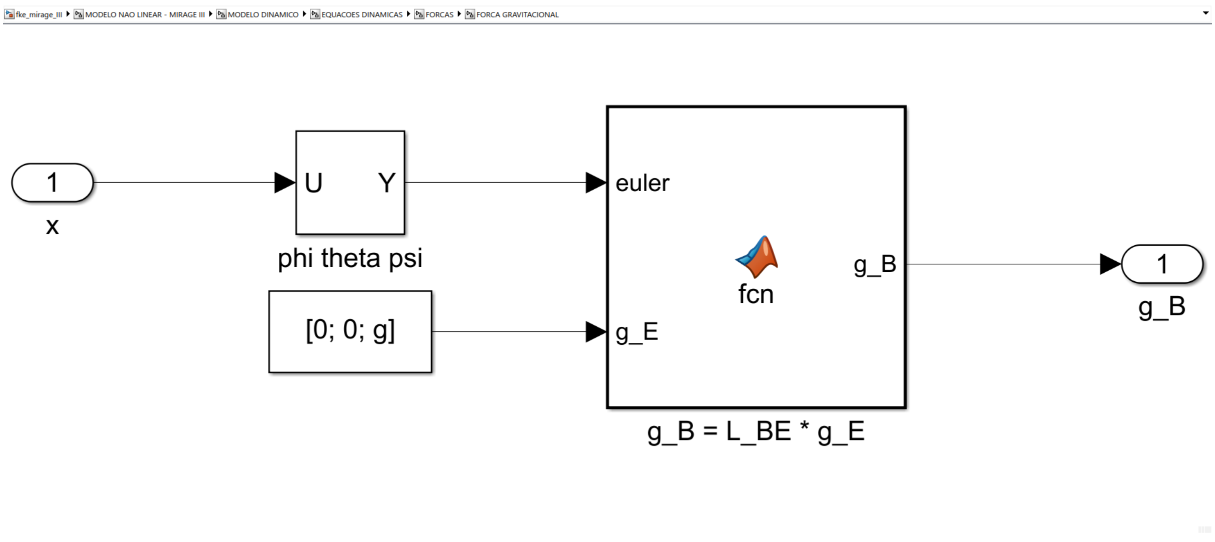


Figura 70 – Subsistema FORCA GRAVITACIONAL.

E a função $g_B = L_{BE} * g_E$ é dada por:

```
 function g_B = fcn(euler, g_E)

phi = euler(1);
theta = euler(2);
psi = euler(3);

% EQUACAO 2.5
L_EB = [cos(theta)*cos(psi), sin(phi)*sin(theta)*cos(psi) - cos(phi)*sin(psi), cos(phi)*sin(theta)*cos(psi) + sin(phi)*sin(psi);
        cos(theta)*sin(psi), sin(phi)*sin(theta)*sin(psi) + cos(phi)*cos(psi), cos(phi)*sin(theta)*sin(psi) - sin(phi)*cos(psi)
        -sin(theta), sin(phi)*cos(theta), cos(phi)*cos(theta)];

g_B = transpose(L_EB) * g_E;
```

O subsistema **FORCAS AERODINAMICAS** da Fig. 69 é dado por:

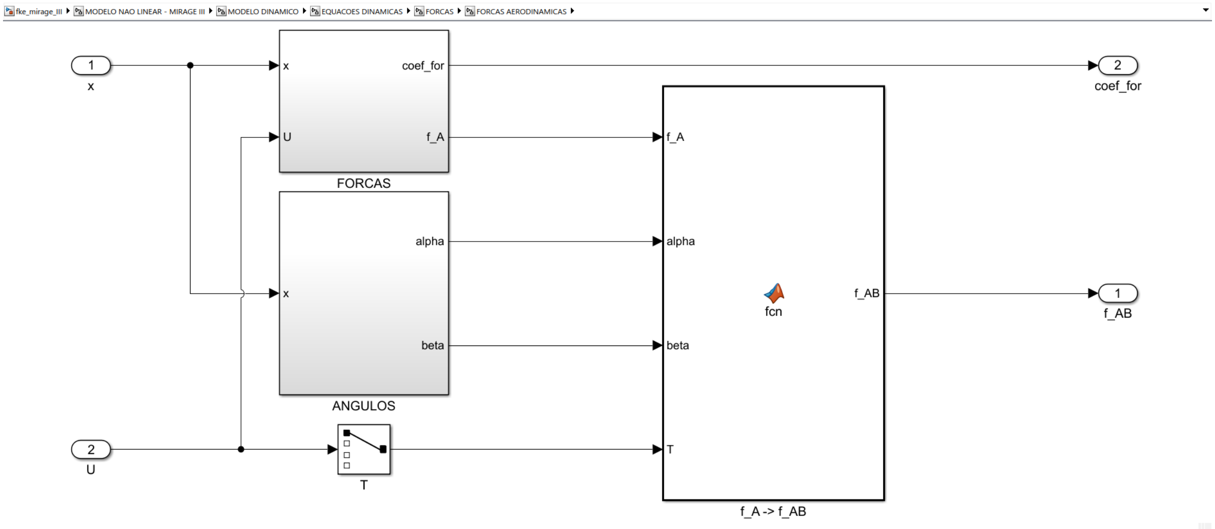


Figura 71 – Subsistema FORCAS AERODINAMICAS.

Onde o subsistema **ANGULOS** é dado pela Fig. 78 e a função $f_A -> f_{AB}$ é dada por:

```
function f_AB = fcn(f_A, alpha, beta, T)

L_AB = [-cos(alpha)*cos(beta), -cos(alpha)*sin(beta), sin(
    alpha);
        -sin(beta), cos(beta), 0;
        -sin(alpha)*cos(beta), -sin(alpha)*sin(beta), -cos(
           alpha)];

f_AB = L_AB * f_A + [T; 0; 0];
```

E o subsistema **FORCAS** é dado por:

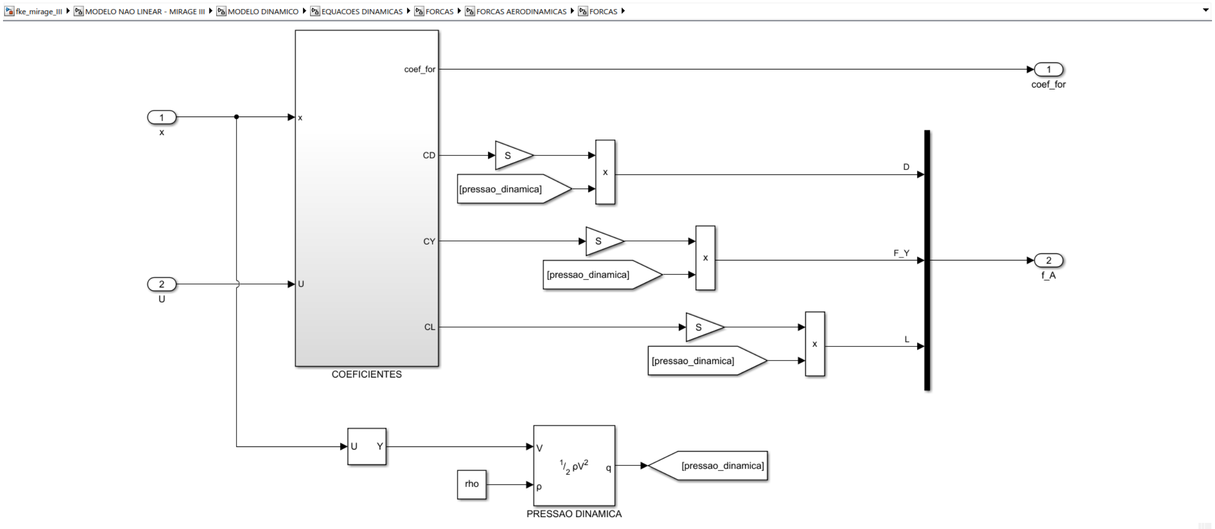


Figura 72 – Subsistema FORCAS AERODINAMICAS/FORCAS.

Onde o subistema **COEFICIENTES** é dado por:

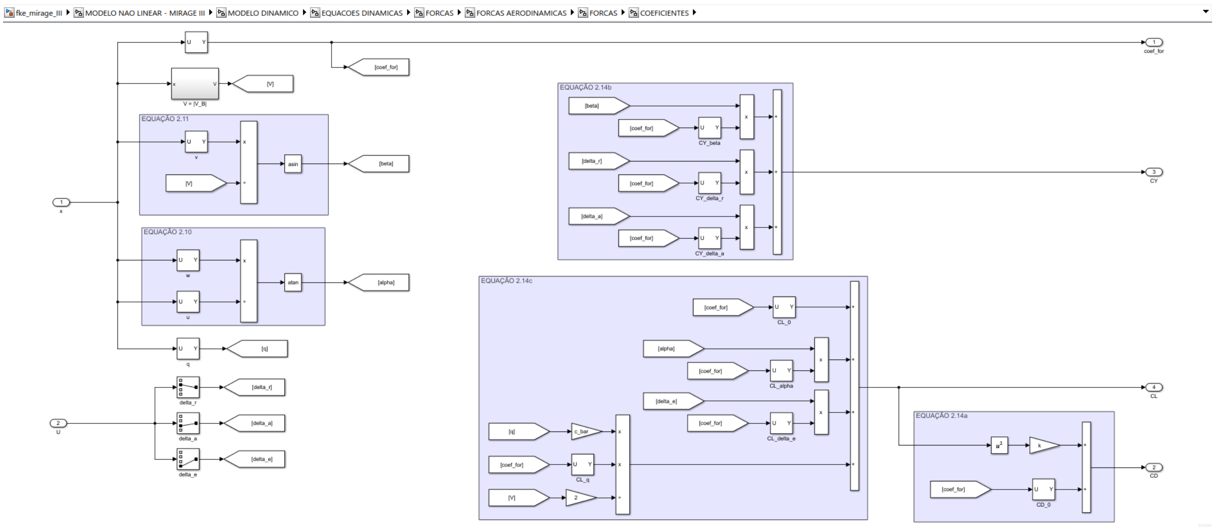


Figura 73 – Subsistema FORCAS/COEFICIENTES.

O subsistema **MOMENTOS** da Fig. 68 é dado por:

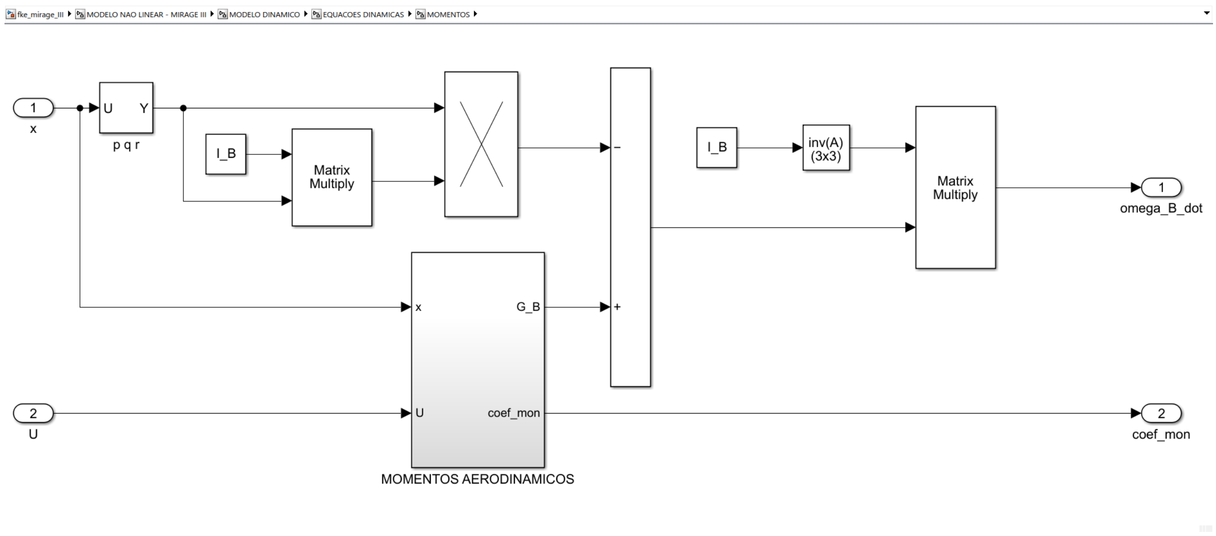


Figura 74 – Subsistema MOMENTOS.

Onde o subsistema **MOMENTOS AERODINAMICOS** é dado por:

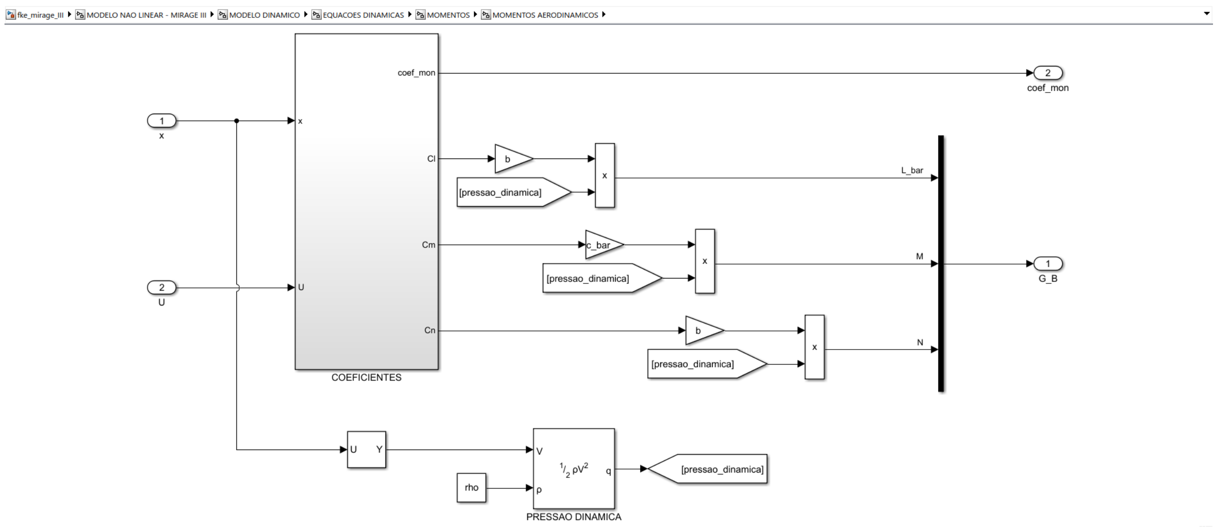


Figura 75 – Subisistema MOMENTOS AERODINAMICOS.

E o subsistema **COEFICIENTES** é dado por:

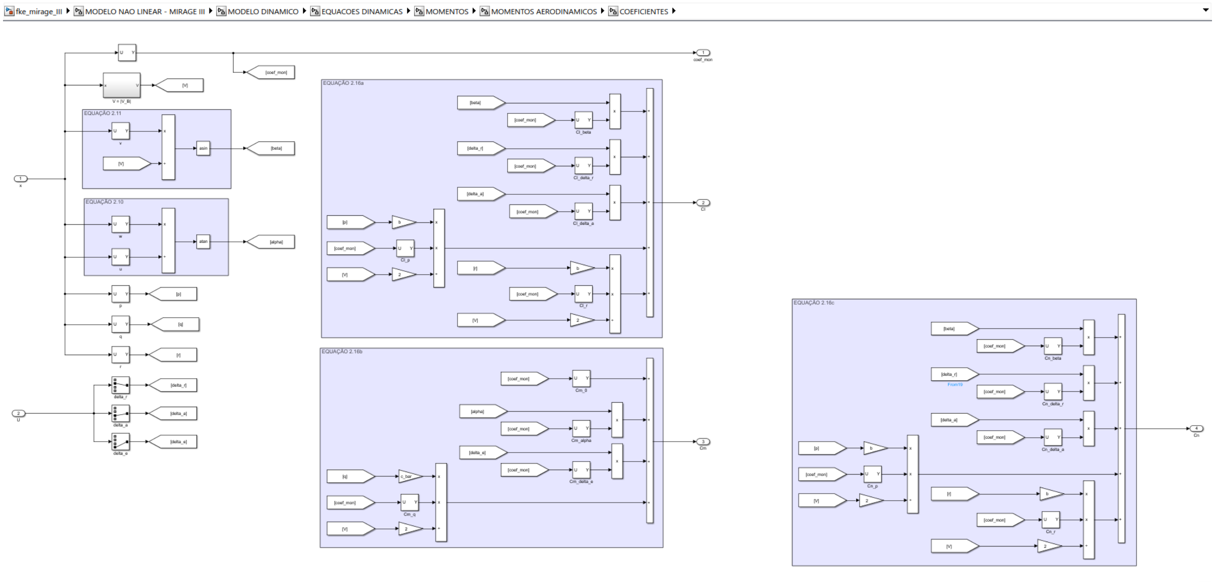


Figura 76 – Subsistema MOMENTOS AERODINAMICOS/COEFICIENTES.

B.2 Subsistema - Saídas

O subsistema **SAIDAS** da Fig. 65 é dado por:

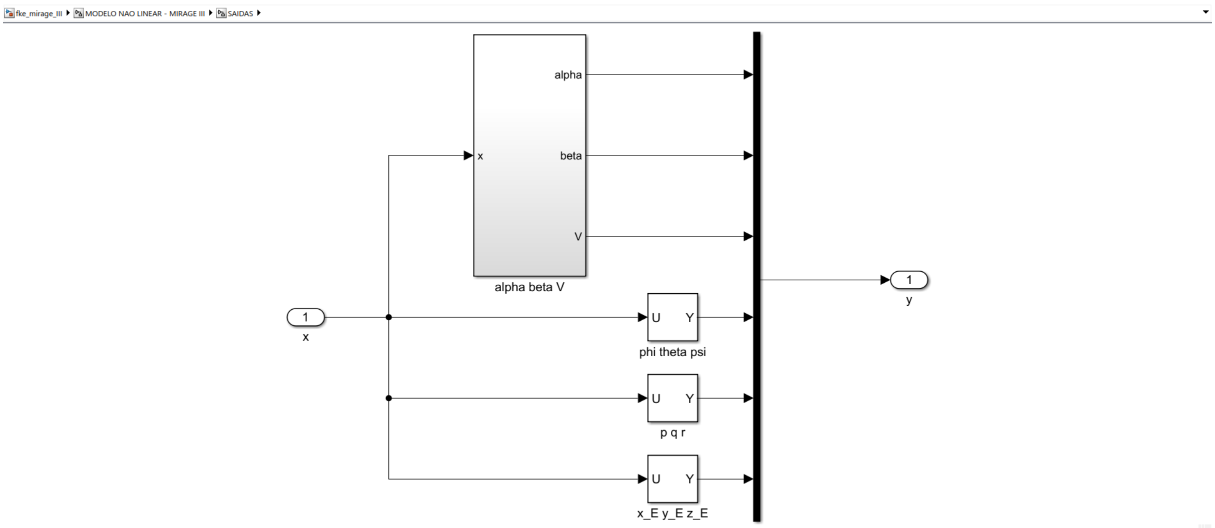


Figura 77 – Subsistema SAIDAS.

Onde o subsistema **alpha beta V** é dado por:

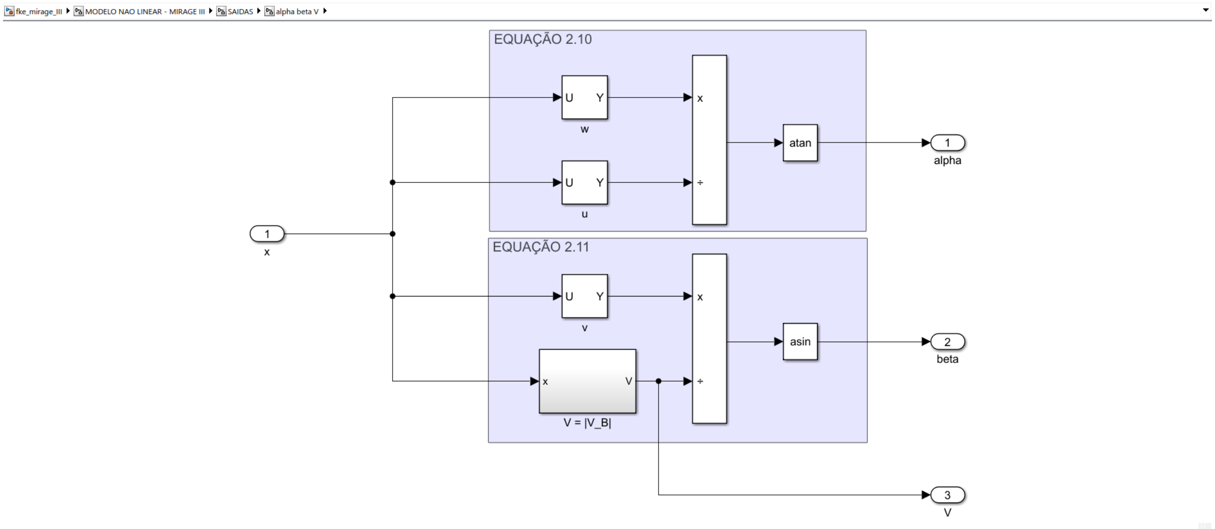


Figura 78 – Subsistema alpha beta V.

E o subsistema $V = |V_B|$ é dado por:

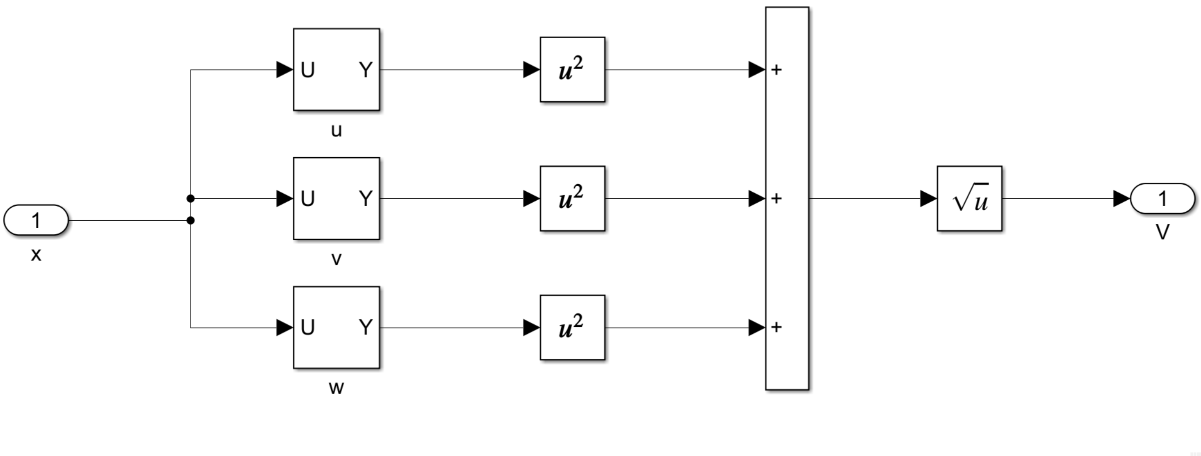


Figura 79 – Cálculo $V = |V_B|$.

APÊNDICE C – Simulink® - Configuração sensor

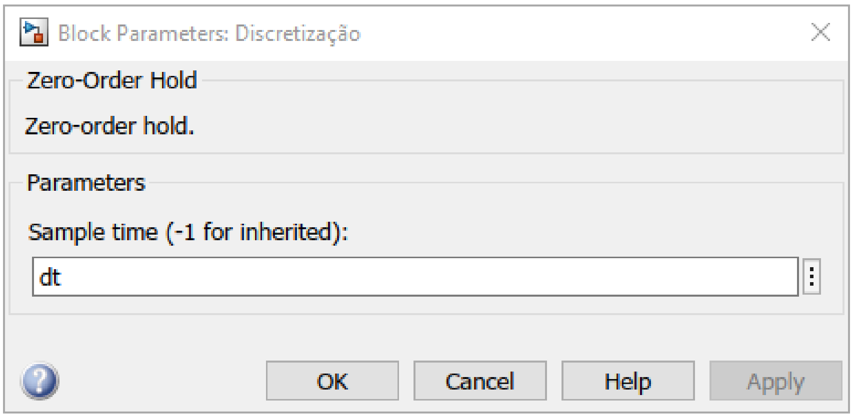


Figura 80 – Configuração do bloco Discretização.

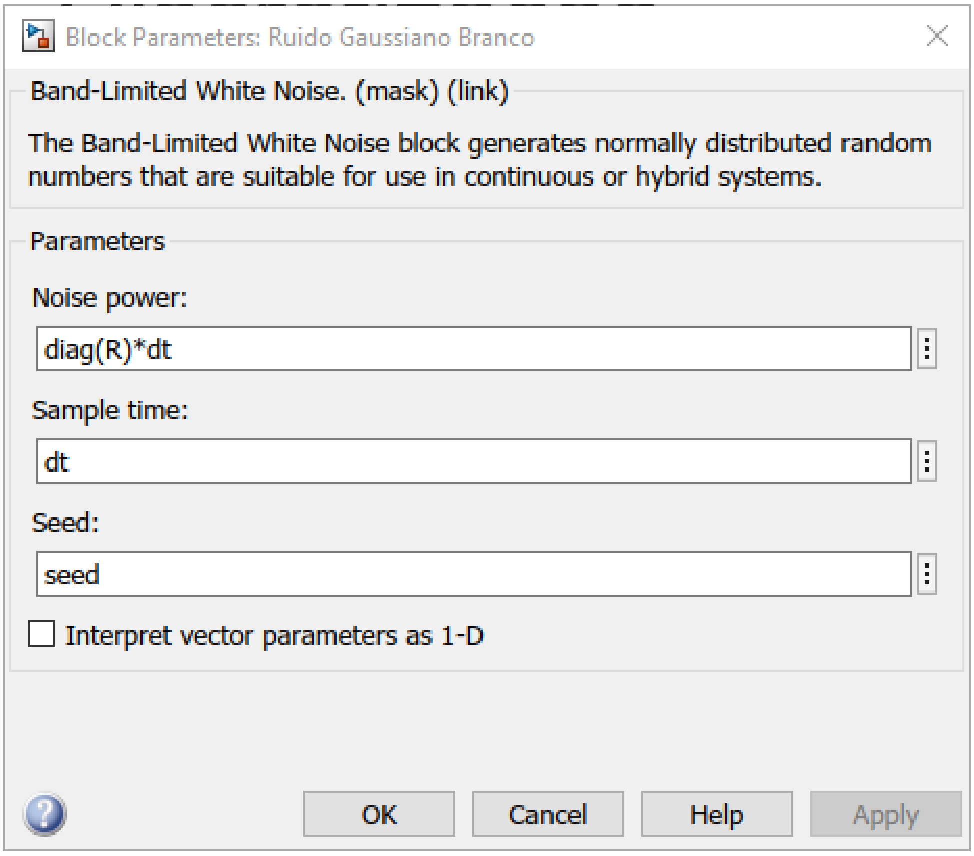


Figura 81 – Configuração do bloco Ruido Gaussiano Branco.

APÊNDICE D – Simulink® - Configuração FKE

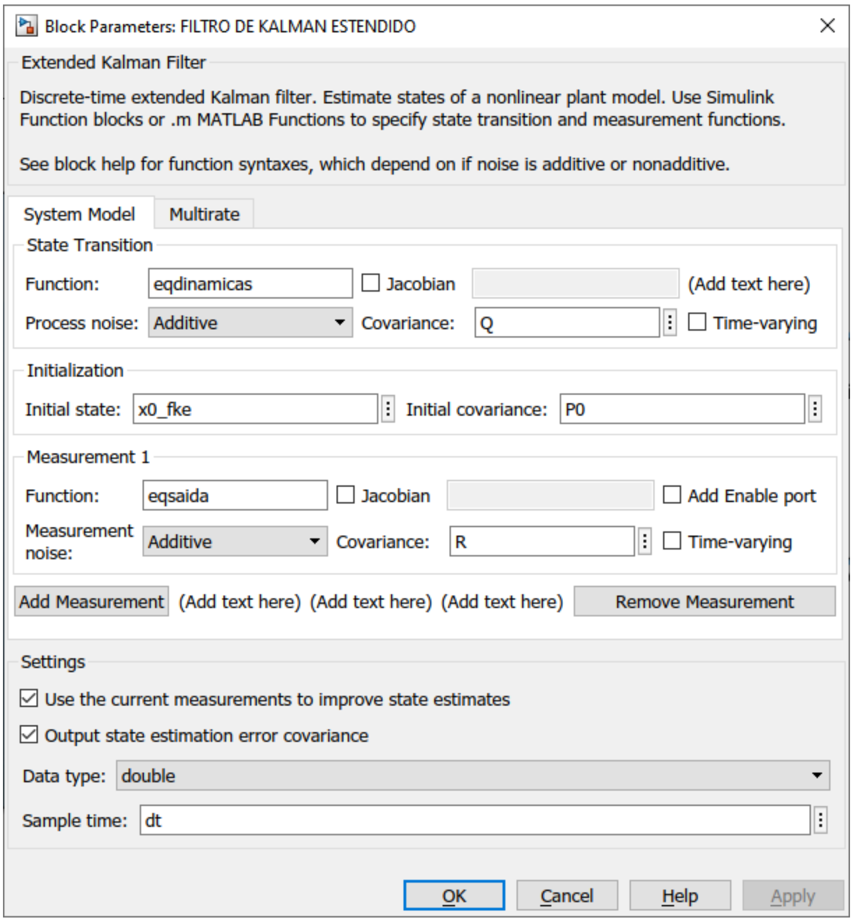


Figura 82 – Configuração do bloco FILTRO DE KALMAN ESTENDIDO.

Em que Q, $x0\_fke$, $P0$, R e dt, são gerados pelos *scripts* do Apêndice A e as funções *eqdinamicas* e *eqsaidas* são dadas por:

```
function x = eqdinamicas(x, U)

dt = 0.01;
f = zeros(34,1);

%% Vetor de estado x

x_E = x(1); y_E = x(2); z_E = x(3);
phi = x(4); theta = x(5); psi = x(6);
u = x(7); v = x(8); w = x(9);
```

```
p = x(10); q = x(11); r = x(12);
CD_0 = x(13);
CY_beta = x(14); CY_delta_r = x(15); CY_delta_a = x(16);
CL_0 = x(17); CL_alpha = x(18); CL_delta_e = x(19); CL_q = x
    (20);
Cl_beta = x(21); Cl_delta_r = x(22); Cl_delta_a = x(23); Cl_p
     = x(24); Cl_r = x(25);
Cm_0 = x(26); Cm_alpha = x(27); Cm_delta_e = x(28); Cm_q = x
    (29);
Cn_beta = x(30); Cn_delta_r = x(31); Cn_delta_a = x(32); Cn_p
     = x(33); Cn_r = x(34);

%% Vetor de entradas u

T = U(1); delta_r = U(2); delta_a = U(3); delta_e = U(4);

%% Caracteristicas do Mirage III
comp = 15;       % Comprimeito (m)
b = 7.5;         % Envergadura (m)
c_bar = 5.25;    % Corda media aerodinamica (m)
S = 36;          % Area da asa (m^2)
m = 7400;        % Massa (kg)

% Matriz momento de inercia (kg*m^2) - PAGLIONE PAGINA 366 DO
     PDF
% Obs: aeronave massivamente simetrica em torno do plano xz
I_B = [9*10^4, 0, -1.8*10^3;
        0, 5.4*10^4, 0;
       -1.8*10^3, 0, 6*10^4];

% Constante da polar CL x CD
k = 0.4;

%% Caracteristicas do ambiente
g = 9.80665;       % Vetor aceleracao da gravidade (m/s^2)
rho = 0.73;        % Densidade do ar a 10000 m (kg/m^3)

%% Equacoes
V = sqrt(u^2 + v^2 + w^2);
```

```
% EQUACAO 2.10
alpha = atan(w/u);

% EQUACAO 2.11
beta = asin(v/V);

% EQUACOES 2.14
CL = CL_0 + CL_alpha*alpha + CL_delta_e*delta_e + CL_q*((
   c_bar*q)/(2*V));
CD = CD_0 + k*CL^2;
CY = CY_beta*beta + CY_delta_r*delta_r + CY_delta_a*delta_a;

% EQUACAO 2.13
D = 0.5*rho*S*V^2*CD;
F_Y = 0.5*rho*S*V^2*CY;
L_F = 0.5*rho*S*V^2*CL;

% EQUACOES 2.12
X = T - D*cos(alpha)*cos(beta) - F_Y*cos(alpha)*sin(beta) +
   L_F*sin(alpha);
Y = -D*sin(beta) + F_Y*cos(beta);
Z = -D*sin(alpha)*cos(beta) - F_Y*sin(alpha)*sin(beta) - L_F*
   cos(alpha);

% EQUACAO 2.16
Cl = Cl_beta*beta + Cl_delta_r*delta_r + Cl_delta_a*delta_a +
    Cl_p*((b*p)/(2*V)) + Cl_r*((b*r)/(2*V));
Cm = Cm_0 + Cm_alpha*alpha + Cm_delta_e*delta_e + Cm_q*((
   c_bar*q)/(2*V));
Cn = Cn_beta*beta + Cn_delta_r*delta_r + Cn_delta_a*delta_a +
    Cn_p*((b*p)/(2*V)) + Cn_r*((b*r)/(2*V));

% EQUACAO 2.15
L = 0.5*rho*S*b*V^2*Cl;
M = 0.5*rho*S*c_bar*V^2*Cm;
N = 0.5*rho*S*b*V^2*Cn;

%% Equacoes cinematicas
```

```
% EQUACAO 2.17
f(1) = u*cos(theta)*cos(psi) + v*(sin(phi)*sin(theta)*cos(psi
   ) - cos(phi)*sin(psi)) + w*(cos(phi)*sin(theta)*cos(psi) +
    sin(phi)*sin(psi));
f(2) = u*cos(theta)*sin(psi) + v*(sin(phi)*sin(theta)*sin(psi
   ) + cos(phi)*cos(psi)) + w*(cos(phi)*sin(theta)*sin(psi) -
    sin(phi)*cos(psi));
f(3) = -u*sin(theta) + v*sin(phi)*cos(theta) + w*cos(phi)*cos
   (theta);

% EQUACAO 2.18
f(4) = p + q*sin(phi)*tan(theta) + r*cos(phi)*tan(theta);
f(5) = q*cos(phi) - r*sin(phi);
f(6) = q*sin(phi)*sec(theta) + r*cos(phi)*sec(theta);

%% Equacoes dinamicas

% EQUACAO 2.19
f(7) = X/m - g*sin(theta) + r*v - q*w;
f(8) = Y/m + g*cos(theta)*sin(phi) + p*w - r*u;
f(9) = Z/m + g*cos(theta)*cos(phi) + q*u - p*v;

% EQUACAO 2.20
f(10:12) = I_B\[L + (I_B(2,2) - I_B(3,3))*q*r + I_B(1,3)*p*q;
                M + (I_B(3,3) - I_B(1,1))*p*r + I_B(1,3)*(r
                   ^2-p^2);
                N + (I_B(1,1) - I_B(2,2))*p*q - I_B(1,3)*q*r
                   ];

% Os coeficientes adimensionais nao variam ao longo do tempo
f(13:34) = 0;

x = x + f*dt;
end
```

```
function y = eqsaida(x)

x_E = x(1); y_E = x(2); z_E = x(3);
```

```
phi = x(4); theta = x(5); psi = x(6);
u = x(7); v = x(8); w = x(9);
p = x(10); q = x(11); r = x(12);

V = sqrt(u^2+v^2+w^2);

% EQUACAO 2.10
alpha = atan(w/u);

% EQUACAO 2.11
beta = asin(v/V);

y = [alpha; beta; V;
     phi; theta; psi;
     p; q; r;
     x_E; y_E; z_E];
end
```