

Zona de Turbulência

Estudo da Estabilidade Estática e Dinâmica do
Cessna 182

SAA0184 - Dinâmica de Voo

Beatriz Figueirêdo de Oliveira (14560763)

Gabriel Nobuaki Nojima (14572227)

Helena Aoki Alves (14577618)

Lívia Máris Senosien Redígolo (13731091)

Renato de Carvalho Martins Filho (14605831)

Rodrigo Amancio Domingues (14606335)

Universidade de São Paulo



Zona de Turbulência

Estudo da Estabilidade Estática e Dinâmica do Cessna 182

por

Beatriz Figueirêdo de Oliveira (14560763)
Gabriel Nobuaki Nojima (14572227)
Helena Aoki Alves (14577618)
Livia Máris Senosien Redígolo (13731091)
Renato de Carvalho Martins Filho (14605831)
Rodrigo Amancio Domingues (14606335)

Professor:	Dr. Ricardo Afonso Angélico
Monitor:	Breno Araujo Valença
Desenvolvimento:	Mai. 2025 - Jun. 2025
Instituto:	Escola de Engenharia de São Carlos (EESC-USP)



Índice

1	Introdução	1
2	Desenvolvimento do Problema	2
2.1	Descrição do Problema	2
2.2	Metodologia	3
2.2.1	Análise de estabilidade estática	3
2.2.2	Análise de estabilidade dinâmica	4
3	Implementação em MATLAB	7
3.1	Modelo Matlab	7
3.2	Modelo Simulink	7
4	Discussão de Resultados	10
4.1	Estabilidade Estática	10
4.2	Estabilidade Dinâmica	11
4.2.1	Modelo Completo Longitudinal	11
4.2.2	Modelo Completo Látero-Direcional	11
4.2.3	Modelos Reduzidos	12
4.2.4	Resposta no Tempo	13
4.2.5	Comparação entre Modelos Linearizado e Não Linearizado	13
5	Conclusões	15
	References	16
A	Códigos Implementados	17
A.1	Definição da estrutura params (params.m)	17
A.2	Análise estática da aeronave (longstat.m)	19
A.3	Análise dinâmica da aeronave (dynamics.m)	20
A.4	Resposta da aeronave a uma entrada degrau unitário no profundor (stepresponse.m)	25
A.5	Preparação da função para ser resolvida pelo ODE45 (LongSimulator.m)	26
A.6	Ciclo principal da simulação (MainSimulator.m)	28
A.7	Extração de dados do Simulink (scopedata.m)	29

1

Introdução

Na formação de um engenheiro aeronáutico, é fundamental compreender o comportamento de uma aeronave diante de diferentes condições atmosféricas e comandos do piloto. Nesse contexto, a disciplina de Dinâmica de Voo apresenta noções fundamentais sobre estabilidade estática e dinâmica, além da resposta da aeronave à atuação dos controles.

Assim, com a finalidade de aplicar de maneira prática os conteúdos teóricos abordados ao longo do semestre, foi proposto o desenvolvimento de um simulador dinâmico de um Cessna 182 que permita observar as respostas a determinadas entradas de controle, como deflexões de profundor e variações nas condições de voo. A partir disso, é possível analisar aspectos estáticos, como a curva $\delta_{e_{trim}}$ vs. $C_{L_{trim}}$ e margem estática, além de aspectos dinâmicos, como a estabilidade longitudinal e látero-direcional do avião.

A elaboração de todo o trabalho tem como motivação a utilização de ferramentas computacionais que reproduzam com fidelidade os efeitos das equações de movimento sobre a trajetória e a atitude de uma aeronave, de maneira a familiarizar os estudantes com *softwares* utilizados na indústria aeronáutica. Ainda, o simulador tem potencial para ser ampliado e aplicado em diversos estudos futuros, como Estabilidade e Controle ou Desempenho.

A modelagem do simulador foi baseada nas equações matemáticas do sistema dinâmico que rege o movimento da aeronave, considerando forças e momentos aerodinâmicos, inerciais e gravitacionais. Além disso, usou-se como dados e *inputs* parâmetros geométricos e atmosféricos fornecidos pelo professor.

Além de ser um exercício fundamental para a consolidação dos conceitos teóricos, este projeto ainda permite o desenvolvimento de *soft skills*, como a cooperação em equipe e a capacidade comunicativa, extremamente valorizadas no mercado profissional.

Dessa forma, espera-se obter um simulador fidedigno da realidade capaz de representar de modo qualitativo e quantitativo o comportamento do Cessna 182. As simulações permitirão observar as características de estabilidade e resposta do avião, além de validar o modelo construído frente aos fenômenos da dinâmica de voo.

Dado que é uma aeronave certificada de uso civil, espera-se obter como resposta um comportamento dinâmico intrinsecamente **estável**, i.e. pequenas perturbações devem resultar em respostas amortecidas, retornando naturalmente ao estado de equilíbrio sem necessidade de atuação ou interferência do piloto.

Ainda, com a linearização do sistema por meio da teoria das pequenas perturbações, espera-se que os resultados obtidos com este modelo sejam suficientemente próximos à modelagem não linear, dado que as simulações são parecidas com as condições de voo de equilíbrio (reto e nivelado). Essa similaridade justifica o uso da linearização como ferramenta válida para análise de estabilidade e resposta de aeronaves operando em regimes normais.

2

Desenvolvimento do Problema

2.1. Descrição do Problema

O Cessna 182, de nome comercial *Skylane* [3] e mostrado na Figura 2.2, é uma aeronave de pequeno porte, amplamente utilizada na aviação geral. Monomotor de tipo *tractor*, asa alta e trem de pouso fixo, sua configuração convencional, vasta quantidade de estudos e dados disponíveis tornam-na ideal para estudos acadêmicos e validação de modelos, sendo, então, escolhida como objeto de estudo deste trabalho.



Figure 2.1: Cessna 182.

Nesse sentido, com base na bibliografia do Roskam [2], utilizou-se as informações do **Apêndice B: Airplane Data** para coletar os parâmetros aerodinâmicos, geométricos e atmosféricos da aeronave na situação de cruzeiro, condição escolhida pelo grupo para análise.

A aeronave, que é de pequeno porte e alta eficiência, tem empenagem tradicional, asa alta, e trem de pouso convencional; tem grande autonomia considerando seu tamanho, com incríveis 1694 km de alcance, segundo sua fabricante, e com a impressionante carga paga máxima de 503 kg.

O presente trabalho visa avaliar a estabilidade completa da aeronave selecionada por dois métodos distintos e sua posterior comparação. Nesse sentido, este objetivo geral pode ser dividido em objetivos específicos, os quais consistem em:

- Análise preliminar da estabilidade estática da aeronave e determinação dos parâmetros de estabilidade e de trimagem;
- Verificação do comportamento dinâmico da aeronave tanto para modelos completos quanto para os reduzidos linearizados;

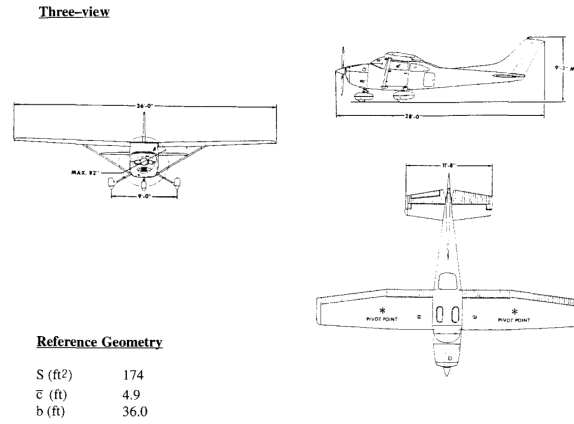


Figure 2.2: Dimensões típicas do Cessna 182 Skylane

- Desenvolvimento do simulador de três graus de liberdade a partir das equações de movimento não linearizadas;
- Comparação das saídas dos modelos adotados anteriormente para uma mesma entrada degrau unitário no profundor

2.2. Metodologia

A metodologia do processo pode ser dividida em três etapas para o cumprimento dos objetivos específicos.

2.2.1. Análise de estabilidade estática

Na primeira etapa, realizou-se a verificação da estabilidade estática da aeronave, i. e., sua tendência a retornar ao estado de equilíbrio após perturbações inerentes ao voo ou a entradas de controle. Para isso, determinou-se o valor da margem estática e a posição do ponto neutro a partir do *datum*.

A forma empregada para o cálculo da margem estática foi realizada através da relação entre as derivadas dos coeficientes de momento de arfagem e de sustentação com a variação do ângulo de ataque, conforme mostra a Equação 2.1.

$$SM = -\frac{C_{M_\alpha}}{C_{L_\alpha}} \quad (2.1)$$

Em seguida, o ponto neutro foi obtido segundo a Equação 2.2, de definição da margem estática.

$$SM = -\frac{(x_{cg} - x_N)}{\bar{c}} \quad (2.2)$$

Posteriormente, a análise das condições de trimagem da aeronave envolveu a determinação das deflexões do profundor e do ângulo de ataque necessários para satisfazer simultaneamente duas condições: um coeficiente de momento de arfagem nulo e os valores desejados de coeficiente de sustentação. Para isso, solucionou-se o sistema matricial apresentado na Eq. 2.3, com base na aplicação da condição de equilíbrio longitudinal de forças e de momentos, isto é, $C_L = C_{L_{trim}}$ e $C_{M_{cg}} = 0$.

$$\begin{bmatrix} C_{L_\alpha} & C_{L_{\delta_e}} \\ C_{M_\alpha} & C_{M_{\delta_e}} \end{bmatrix} \begin{bmatrix} \alpha_{trim} \\ \delta_{e_{trim}} \end{bmatrix} = \begin{bmatrix} C_{L_{trim}} - C_{L_0} \\ -C_{M_0} \end{bmatrix} \quad (2.3)$$

Os dados foram compilados em um gráfico de $\delta_{e_{trim}}$ vs. $C_{L_{trim}}$ para a visualização das deflexões necessárias para se atingir valores requeridos de coeficiente de sustentação.

2.2.2. Análise de estabilidade dinâmica

Após a verificação da estabilidade estática, prosseguiu-se com a análise da estabilidade dinâmica, a qual avalia o comportamento da aeronave ao longo do tempo após ser perturbada de sua condição de equilíbrio. Esse estudo verifica se os movimentos oscilatórios resultantes são amortecidos, mantêm-se constantes ou se são amplificados com o tempo. A estabilidade dinâmica abrange dois tipos de movimento: o longitudinal e o látero-direcional.

Modelo longitudinal

A estabilidade dinâmica longitudinal refere-se à resposta da aeronave em torno do seu eixo transversal, ou seja, relacionada ao movimento de arfagem. Essa análise é caracterizada por dois modos principais: o de curto-período, modo oscilatório de alta frequência e rápida amortização, e o de fugóide, modo oscilatório de baixa frequência e alta duração.

Para a modelagem desses comportamentos, utilizou-se as variáveis do espaço de estados $x_{lg} = [u \ w \ q \ \theta]^T$, representando, respectivamente, a velocidade longitudinal, a velocidade vertical, a razão de arfagem e o ângulo de arfagem. Além disso, determinaram-se as derivadas longitudinais para o cálculo das matrizes do modelo linearizado em espaço de estados, com base no material auxiliar [1] fornecido pelo professor.

$$\begin{aligned}
 X_u &= \frac{-qS(C_{D_u} + 2C_{D_1})}{mU_1} & X_{Tu} &= \frac{qS(C_{Txu} + 2C_{Tx1})}{mU_1} & X_\alpha &= \frac{-qS(C_{D_\alpha} - C_{L_1})}{m} \\
 Z_u &= \frac{-qS(C_{L_u} + 2C_{L_1})}{mU_1} & Z_\alpha &= \frac{-qS(C_{L_\alpha} + 2C_{D_1})}{m} & Z_{\dot{\alpha}} &= \frac{-qS\bar{c}C_{L\dot{\alpha}}}{2mU_1} & Z_q &= \frac{-qS\bar{c}C_{Lq}}{2mU_1} \\
 M_u &= \frac{qS\bar{c}C_{mu}}{I_{yy}U_1} & M_{Tu} &= \frac{qS\bar{c}C_{mTu}}{I_{yy}U_1} & M_\alpha &= \frac{qS\bar{c}C_{m\alpha}}{I_{yy}} \\
 M_{T\alpha} &= \frac{qS\bar{c}C_{mT\alpha}}{I_{yy}} & M_{\dot{\alpha}} &= \frac{qS\bar{c}^2C_{m\dot{\alpha}}}{2I_{yy}U_1} & M_q &= \frac{qS\bar{c}^2C_{mq}}{2I_{yy}U_1} \\
 X_{\delta_e} &= \frac{-qSC_{D\delta_e}}{m} & Z_{\delta_e} &= \frac{-qSC_{L\delta_e}}{m} & M_{\delta_e} &= \frac{qS\bar{c}C_{m\delta_e}}{I_{yy}}
 \end{aligned}$$

Após a determinação das derivadas longitudinais, foi possível escrever as equações de espaço de estado a partir das matrizes representadas pelas Eqs. 2.4 e 2.5.

$$A_{lg} = \begin{bmatrix} X_u + X_{Tu} & \frac{X_\alpha}{U_1} & 0 & -g \cos \theta_1 \\ \frac{U_1 Z_u}{U_1 - Z_{\dot{\alpha}}} & \frac{Z_\alpha}{U_1 - Z_{\dot{\alpha}}} & \frac{U_1(U_1 + Z_q)}{U_1 - Z_{\dot{\alpha}}} & \frac{-U_1 g \sin \theta_1}{U_1 - Z_{\dot{\alpha}}} \\ M_u + M_{Tu} & \frac{M_\alpha + M_{T\alpha}}{U_1} + \frac{M_{\dot{\alpha}} Z_\alpha}{U_1(U_1 - Z_{\dot{\alpha}})} & \frac{-M_{\dot{\alpha}}(Z_q + U_1)}{U_1 - Z_{\dot{\alpha}}} + M_q & \frac{M_{\dot{\alpha}} g \sin \theta_1}{U_1 - Z_{\dot{\alpha}}} \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (2.4)$$

$$B_{lg} = \begin{bmatrix} X_{\delta_e} \\ \frac{Z_{\delta_e}}{U_1 - Z_{\dot{\alpha}}} \\ M_{\delta_e} + \frac{M_{\dot{\alpha}} Z_{\delta_e}}{U_1 - Z_{\dot{\alpha}}} \\ 0 \end{bmatrix} \quad (2.5)$$

Desse modo, a representação em espaço de estados do modelo completo longitudinal, tendo como entrada uma deflexão no profundor, é apresentada por:

$$\begin{cases} \dot{x}_{lg} = A_{lg} \cdot x_{lg} + B_{lg} \cdot \delta_e \\ y_{lg} = x_{lg} \end{cases} \quad (2.6)$$

Modelo látero-direcional

A estabilidade dinâmica látero-direcional está relacionada à resposta da aeronave em torno dos eixos vertical e longitudinal, ou seja, aos movimentos de guinada e rolagem. Essa análise compreende modos característicos como o Dutch-roll, uma oscilação acoplada entre guinada e rolagem, um modo de amortecimento puro da rolagem e o modo espiral, uma guinada lenta.

Para a modelagem desses comportamentos, foram utilizadas as variáveis do espaço de estados $x_{ld} = [v \ p \ r \ \phi]^T$, representando, respectivamente, a velocidade lateral, a razão de rolagem, a razão de guinada e o ângulo de rolagem. Além disso, determinaram-se as derivadas látero-direcionais, as quais foram utilizadas no cálculo das matrizes do modelo linearizado em espaço de estados.

$$\begin{aligned} Y_\beta &= \frac{qSC_{Y\beta}}{m} & Y_p &= \frac{qSbC_{Yp}}{2mU_1} & Y_r &= \frac{qSbC_{Yr}}{2mU_1} \\ L_\beta &= \frac{qSbC_{l\beta}}{I_{xx}} & L_p &= \frac{qSb^2C_{lp}}{2I_{xx}U_1} & L_r &= \frac{qSb^2C_{lr}}{2I_{xx}U_1} \\ N_\beta &= \frac{qSbC_{n\beta}}{I_{zz}} & N_{T\beta} &= \frac{qSbC_{nT\beta}}{I_{zz}} & N_p &= \frac{qSb^2C_{np}}{2I_{zz}U_1} & N_r &= \frac{qSb^2C_{nr}}{2I_{zz}U_1} \\ Y_{\delta_a} &= \frac{qSC_{Y\delta_a}}{m} & Y_{\delta_r} &= \frac{qSC_{Y\delta_r}}{m} \\ L_{\delta_a} &= \frac{qSbC_{l\delta_a}}{I_{xx}} & L_{\delta_r} &= \frac{qSbC_{l\delta_r}}{I_{xx}} \\ N_{\delta_a} &= \frac{qSbC_{n\delta_a}}{I_{zz}} & N_{\delta_r} &= \frac{qSbC_{n\delta_r}}{I_{zz}} \end{aligned}$$

A partir das derivadas, foi possível determinar as matrizes látero-direcionais, representadas pelas Eqs. 2.7 e 2.8, utilizadas nas equações de espaço de estado.

$$A_{ld} = \begin{bmatrix} \frac{Y_\beta}{U_1} & Y_p & Y_r - U_1 & g \cos \theta_1 \\ \frac{L_\beta + i_x N_\beta + i_x N_{T\beta}}{U_1(1 - i_x i_z)} & \frac{L_p + i_x N_p}{1 - i_x i_z} & \frac{L_r + i_x N_r}{1 - i_x i_z} & 0 \\ \frac{i_z L_\beta + N_\beta + N_{T\beta}}{U_1(1 - i_x i_z)} & \frac{i_z L_p + N_p}{1 - i_x i_z} & \frac{i_z L_r + N_r}{1 - i_x i_z} & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \quad (2.7)$$

$$B_{ld} = \begin{bmatrix} Y_{\delta_a} & Y_{\delta_r} \\ \frac{L_{\delta_a} + i_x N_{\delta_a}}{1 - i_x i_z} & \frac{L_{\delta_r} + i_x N_{\delta_r}}{1 - i_x i_z} \\ \frac{i_z L_{\delta_a} + N_{\delta_a}}{1 - i_x i_z} & \frac{i_z L_{\delta_r} + N_{\delta_r}}{1 - i_x i_z} \\ 0 & 0 \end{bmatrix} \quad (2.8)$$

Sendo i_x e i_z termos combinados, tais que:

$$i_x = \frac{I_{xz}}{I_{xx}} \quad i_z = \frac{I_{xz}}{I_{zz}}$$

A representação em espaço de estados do modelo completo látero-direcional pode ser escrita de forma análoga ao longitudinal (Eq. 2.6), efetuando-se a substituição das matrizes e vetores pelos respectivos da dinâmica látero-direcional. As entradas serão, dessa vez, deflexões de aileron e de leme.

Modelos reduzidos

O modelo reduzido de curto-período é construído com a matriz de estados A_{sp} composta pelos elementos centrais da matriz A_{lg} e considera como efeito dominante dos estados $x_{sp} = \begin{bmatrix} w & q \end{bmatrix}^T$. Desse modo, as matrizes A_{sp} e B_{sp} são mostradas em:

$$A_{sp} = \begin{bmatrix} \frac{Z_\alpha}{U_1 - Z_{\dot{\alpha}}} & \frac{U_1(U_1 + Z_q)}{U_1 - Z_{\dot{\alpha}}} \\ \frac{M_\alpha + M_{T\alpha}}{U_1} + \frac{M_{\dot{\alpha}}Z_\alpha}{U_1(U_1 - Z_{\dot{\alpha}})} & \frac{-M_{\dot{\alpha}}(Z_q + U_1)}{U_1 - Z_{\dot{\alpha}}} + M_q \end{bmatrix} \quad B_{sp} = \begin{bmatrix} \frac{Z_{\delta_e}}{U_1 - Z_{\dot{\alpha}}} \\ M_{\delta_e} + \frac{M_{\dot{\alpha}}Z_{\delta_e}}{U_1 - Z_{\dot{\alpha}}} \end{bmatrix} \quad (2.9)$$

O modelo reduzido de Dutch-roll é construído com a matriz de estados A_{dr} composta pelos elementos $[1, 1], [1, 3], [3, 1], [3, 3]$ da matriz A_{ld} e considera como efeito dominante dos estados $x_{dr} = \begin{bmatrix} v & r \end{bmatrix}^T$ e suas matrizes de estado e de entradas são:

$$A_{dr} = \begin{bmatrix} \frac{Y_\beta}{U_1} & Y_r - U_1 \\ \frac{i_z L_\beta + N_\beta + N_{T\beta}}{U_1(1 - i_x i_z)} & \frac{i_z L_r + N_r}{1 - i_x i_z} \end{bmatrix} \quad B_{dr} = \begin{bmatrix} Y_{\delta_a} & Y_{\delta_r} \\ \frac{i_z L_{\delta_a} + N_{\delta_a}}{1 - i_x i_z} & \frac{i_z L_{\delta_r} + N_{\delta_r}}{1 - i_x i_z} \end{bmatrix} \quad (2.10)$$

Por fim, o modelo reduzido de rolagem é construído com a matriz de estados A_{rl} composta pelo elemento $[2, 2]$ da matriz A_{ld} e considera como efeito dominante dos estados $x_{rl} = \begin{bmatrix} p \end{bmatrix}^T$. Suas matrizes de estados e de entrada estão dispostas em:

$$A_{rl} = \begin{bmatrix} \frac{L_p + i_x N_p}{1 - i_x i_z} \end{bmatrix} \quad B_{rl} = \begin{bmatrix} \frac{L_{\delta_a} + i_x N_{\delta_a}}{1 - i_x i_z} & \frac{L_{\delta_r} + i_x N_{\delta_r}}{1 - i_x i_z} \end{bmatrix} \quad (2.11)$$

Para os modelos reduzidos, a representação em espaço de estados segue o mesmo padrão adotado para os modelos completos, realizando-se as substituições necessárias.

3

Implementação em MATLAB

Neste trabalho, foram utilizadas duas abordagens distintas, o scripting simples para análise dos parâmetros da aeronave e montagem das equações para representação por espaço de estados da dinâmica látero-direcional e longitudinal da aeronave e a montagem do simulador de três graus de liberdade não linearizado.

3.1. Modelo Matlab

Primeiramente, foi criado o código `params.m`, que enumera os parâmetros do avião e das condições de voo obtidos no anexo B1 do Roskam[2], criando o arquivo `params.m`, que contém a estrutura `params.mat`, em cima da qual elaboraremos o resto dos códigos.

Feito isso, foi criado o código `longstat.m`, que faz a análise da estabilidade estática da aeronave, e de suas possíveis condições trimadas de voo.

Posteriormente, no código `dynamics.m`, é feita a análise da estabilidade dinâmica da aeronave através da montagem e análise de suas matrizes de estabilidade, tanto completas quanto reduzidas.

Por fim, o código `stepresponse.m` utiliza as matrizes elaboradas no código `dynamics.m` e aplica a elas uma função degrau unitário de deflexão de profundor, de modo que possa ser observada a resposta dinâmica do sistema a essa entrada.

3.2. Modelo Simulink

Para complementar a análise da estabilidade dinâmica, foi desenvolvido em MATLAB um simulador de três graus de liberdade para o movimento longitudinal, baseado em um modelo não linearizado. O simulador é dividido em três scripts principais: `params.m`, `MainSimulator.m` e `LongSimulator.m`.

O script `params.m` centraliza todos os parâmetros geométricos, aerodinâmicos e de voo necessários, salvando-os em um arquivo `aircraft.mat` para serem utilizados pelos outros módulos.

O script `MainSimulator.m` executa o ciclo principal da simulação. Primeiramente, carrega os parâmetros da aeronave a partir do arquivo `.mat`. Em seguida, realiza a trimagem por meio da função `trim_aircraft`, determinando o ângulo de ataque de trimagem (α_{trim}), a deflexão do profundor (δ_{trim}) e o empuxo de trimagem (T_{trim}). As condições iniciais são calculadas por:

$$u_0 = U_{inf} \cos(\alpha_{trim})$$

$$w_0 = U_{inf} \sin(\alpha_{trim})$$

$$q_0 = 0$$

$$\theta_0 = \alpha_{trim}$$

Assim, o vetor de estados iniciais é composto por $[u_0, w_0, q_0, \theta_0, x_0, z_0]$. Após isso, as entradas de controle são configuradas e as equações diferenciais são integradas numericamente pela função `ode45`, que utiliza `LongSimulator` para calcular as derivadas.

A função `LongSimulator.m` implementa o modelo dinâmico não linearizado da aeronave. Sua estrutura é organizada em três blocos lógicos. O primeiro bloco calcula os coeficientes aerodinâmicos por meio da subfunção `aerocoeff`:

$$C_L = C_{L0} + C_{L\alpha}\alpha + (C_{L\dot{\alpha}}\dot{\alpha} + C_{Lq}q)\frac{\bar{c}}{2U_\infty} + C_{L\delta_e}\delta_e$$

$$C_M = C_{M0} + C_{M\alpha}\alpha + (C_{M\dot{\alpha}}\dot{\alpha} + C_{Mq}q)\frac{\bar{c}}{2U_\infty} + C_{M\delta_e}\delta_e$$

$$C_D = C_{D0} + kC_L^2$$

Para obter $\dot{\alpha}$ de forma consistente, um algoritmo de ponto fixo é empregado.

O segundo bloco, `forcesmoments`, determina as forças e momentos aerodinâmicos e propulsivos:

$$D = \frac{1}{2}\rho U_\infty^2 S C_D$$

$$L = \frac{1}{2}\rho U_\infty^2 S C_L$$

$$M = \frac{1}{2}\rho U_\infty^2 S \bar{c} C_M$$

Para a condição de voo de cruzeiro, onde a aeronave está em voo retilíneo e nivelado, a tração de trimagem (T_{trim}) é aquela que equilibra o arrasto, sendo calculada como:

$$T = T_{trim} = -L \sin(\alpha_{trim}) + D \cos(\alpha_{trim})$$

$$X = T \cos(\epsilon_T) - D \cos(\alpha) + L \sin(\alpha)$$

$$Z = -T \sin(\epsilon_T) - D \sin(\alpha) - L \cos(\alpha)$$

Levando-se também em consideração que para o avião estudado, $\epsilon_T = 0$. Por fim, o bloco `motion` implementa as equações diferenciais do movimento:

$$\dot{u} = \frac{X}{m} - g \sin(\theta) - qw$$

$$\dot{w} = \frac{Z}{m} + g \cos(\theta) + qu$$

$$\dot{q} = \frac{M}{I_{yy}}$$

$$\dot{\theta} = q$$

$$\dot{x}_e = u \cos(\theta) + w \sin(\theta)$$

$$\dot{z}_e = -u \sin(\theta) + w \cos(\theta)$$

Um diagrama em Simulink complementa a visualização da arquitetura, ilustrando o fluxo de informações entre os blocos `aerocoef`, `forcesmoments` e `motion`, desde a entrada de parâmetros e comandos até a integração dos estados.

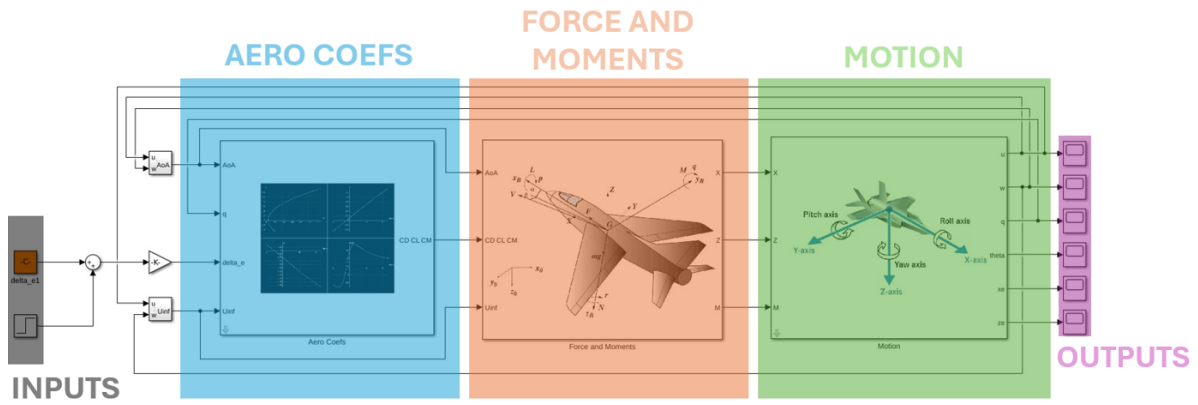


Figure 3.1: Representação gráfica da arquitetura do simulador longitudinal em Simulink.

Após a simulação, os resultados para as variáveis u , w , q e θ foram extraídos dos *scopes* do Simulink por meio do script `scope_data.m`. Esse script acessa os *timeseries* de cada elemento do Dataset de saída da simulação.

Discussão de Resultados

4.1. Estabilidade Estática

A partir da aplicação das Eqs. 2.1 e 2.2 sobre os dados da aeronave, foi possível a obtenção da margem estática, de $SM = 13.9\%$. Com referência no *datum* (o bordo de ataque da projeção de \bar{c} sobre o plano longitudinal) determinou-se também a posição do ponto neutro da aeronave ($x_N = 0.602 \text{ m}$). Conhecida a posição do centro de gravidade ($x_{cg} = 0.394 \text{ m}$), a estabilidade estática da aeronave pôde ser determinada, visto que a margem estática positiva indica que a derivada do momento de arfagem em relação ao ângulo de ataque (C_{M_α}) é negativa, isto é, a aeronave apresenta um momento restaurador uma vez submetida a uma variação positiva de ângulo de ataque. Tal comportamento é explicado pelo posicionamento do centro de gravidade à frente do ponto neutro da aeronave, onde o momento de arfagem é constante.

Através da análise da condição de equilíbrio, realizou-se a busca por deflexões do profundor, bem como de ângulos de ataque, que satisfizessem a ambos requisitos de trimagem: equilíbrio de forças e de momentos a partir da Eq. 2.3. As informações foram compilados em um gráfico $\delta_{e_{trim}}$ vs. $C_{L_{trim}}$, apresentado na Fig. 4.1.

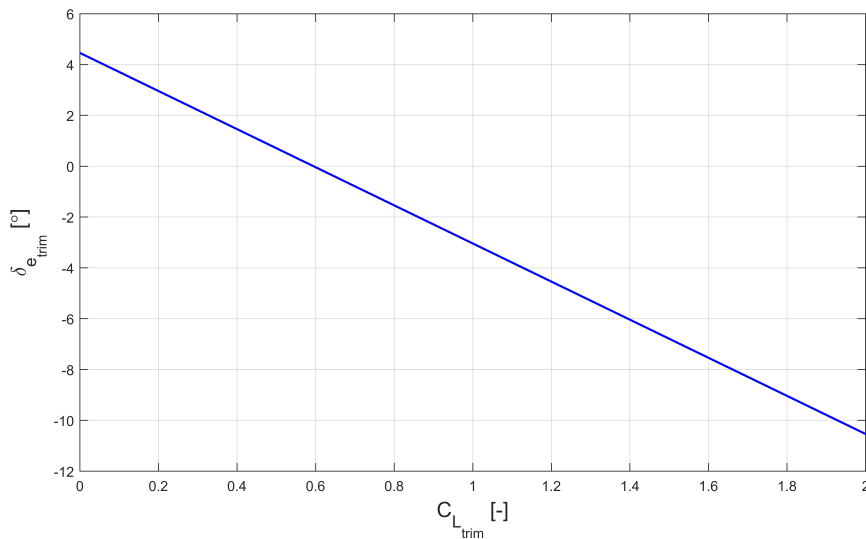


Figure 4.1: Variações de C_L trimado com deflexões do profundor.

O comportamento decrescente ilustrado na Fig. 4.1 é esperado, uma vez que deflexões

positivas do profundor geram força de sustentação na cauda para cima, cujo efeito, em relação ao c_g é de momento picador. A consequência direta da aplicação desse momento é a diminuição do ângulo de ataque da aeronave, o que provoca diminuição do C_L gerado pela asa. Nesse sentido, defletir o profundor para baixo gera uma diminuição do $C_{L_{trim}}$.

4.2. Estabilidade Dinâmica

4.2.1. Modelo Completo Longitudinal

O modelo completo da dinâmica longitudinal da aeronave foi obtida através da matriz de estados A_{lg} , apresentada na Eq. 2.4. O principal parâmetro de análise da estabilidade dinâmica longitudinal foi a verificação dos autovalores da matriz A_{lg} , cujos valores são dispostos na Tab. 4.1, junto à identificação dos modos correspondentes a cada autovalor.

Table 4.1: Autovalores associados aos modos dinâmicos longitudinais.

Modo dinâmico longitudinal	Autovalor (λ)
Curto-período	$-4.4579 \pm 2.8255i$
Fugóide	$-0.0226 \pm 0.1436i$

Uma análise preliminar permite concluir que a aeronave é estável para ambos os modos dinâmicos longitudinais, visto que possui parte real dos autovalores negativa. Isso expressa, na resposta no domínio do tempo, um decaimento exponencial, indicando uma tendência de retorno ao estado de equilíbrio após uma perturbação. Além disso, para ambos os modos, os autovalores são pares conjugados, cuja parte imaginária tem efeito oscilatório na resposta no tempo. A identificação dos autovalores correspondentes a cada modo dinâmico foi realizada a partir da comparação das partes imaginárias dos autovalores. Sendo o modo de curto-período caracterizado pela resposta rápida, sua parte imaginária será maior que a do modo fugóide, com resposta mais lenta. A confirmação dessa interpretação pode ser obtida pelos valores de fator de amortecimento e frequência natural dos modos, mostrada na Tab 4.2.

Table 4.2: Fator de amortecimento e frequências naturais dos modos dinâmicos.

Modo dinâmico longitudinal	Fator de amortecimento	Frequência Natural (rad/s)
Curto-período	0.8446	5.2779
Fugóide	0.1555	0.1454

O curto período apresenta, portanto, alta frequência de oscilação e um amortecimento maior que o modo fugóide.

4.2.2. Modelo Completo Látero-Direcional

A matriz de estados representativa da dinâmica látero-direcional A_{ld} é apresentada na Eq. 2.7. Analogamente à análise longitudinal, realizou-se a avaliação dos autovalores da matriz A_{ld} , fatores de amortecimento e frequências naturais dos modos dinâmicos. Os dados foram compilados na Tab 4.3.

A partir da Tab. 4.3, verifica-se que a aeronave é também estável para todos os modos dinâmicos látero-direcionais, uma vez que as partes reais dos autovalores são negativas. Observa-se, ainda, que apenas o modo de Dutch-roll é oscilatório, representado por um par conjugado de autovalores complexos; os demais modos apresentam apenas decaimento exponencial sem oscilação, sendo a rolagem mais rápida que a espiral, por apresentar autovalor mais distante da origem no plano complexo.

Table 4.3: Autovalores dos modos dinâmicos latero-direcionais.

Modo dinâmico latero-direcional	Autovalores
Dutch-roll	$-0.6734 \pm 3.1756i$
Rolagem	-13.0054
Espiral	-0.0179

Além disso, avaliou-se o fator de amortecimento e frequência natural do modo Dutch-roll para caracterizar sua estabilidade dinâmica:

$$\begin{aligned}\zeta_{dr} &= 0.8446 \\ \omega_{ndr} &= 5.2779 \text{ rad/s}\end{aligned}$$

4.2.3. Modelos Reduzidos

O modelo reduzido de curto-período é construído a partir das relações nas Eqs. 2.9. Nesse sentido, a Tab. 4.4 compila os dados dos autovalores, do fator de amortecimento e da frequência natural do modelo reduzido de curto-período.

Table 4.4: Autovalor, fator de amortecimento e frequência natural do curto-período reduzido.

Autovalor (λ)	Fator de amortecimento	Frequência Natural (rad/s)
$-4.4577 \pm 2.8243i$	0.8447	5.2771

É possível notar a semelhança entre os autovalores deste modelo e do completo e, consequentemente, o fator de amortecimento e frequência natural. Nesse sentido, a aproximação do modelo reduzido de curto-período é adequada para a representação da dinâmica da aeronave.

O modelo reduzido de Dutch-roll é construído com a matriz de estados A_{dr} , representada na Eq. 2.10. Assim, a Tab. 4.5 compila os dados dos autovalores, do fator de amortecimento e da frequência natural do modelo reduzido de Dutch-roll.

Table 4.5: Autovalor, fator de amortecimento e frequência natural do Dutch-roll reduzido.

Autovalor (λ)	Fator de amortecimento	Frequência Natural (rad/s)
$-0.6987 \pm 2.9888i$	0.2276	3.0694

É possível observar certa semelhança entre a parte real dos autovalores, apesar da leve diferença entre os valores de fator de amortecimento e de frequência natural obtidos entre o sistema reduzido e completo. Nesse sentido, a aproximação do modelo reduzido de Dutch-roll é adequada para a representação da dinâmica da aeronave, mas com certa ressalva, ainda mais quando comparado com o modelo reduzido de curto-período.

Já o modelo reduzido de rolagem é construído com a matriz de estados A_{rl} a partir das relações da Eq. 2.11. Dessa maneira, a Tab. 4.6 compila os dados do autovalor e da constante de tempo do modelo reduzido de rolagem.

Por fim, pode-se concluir que o modelo reduzido de rolagem se adequa muito bem ao seu papel, devido à grande semelhança entre os autovalores de rolagem obtidos.

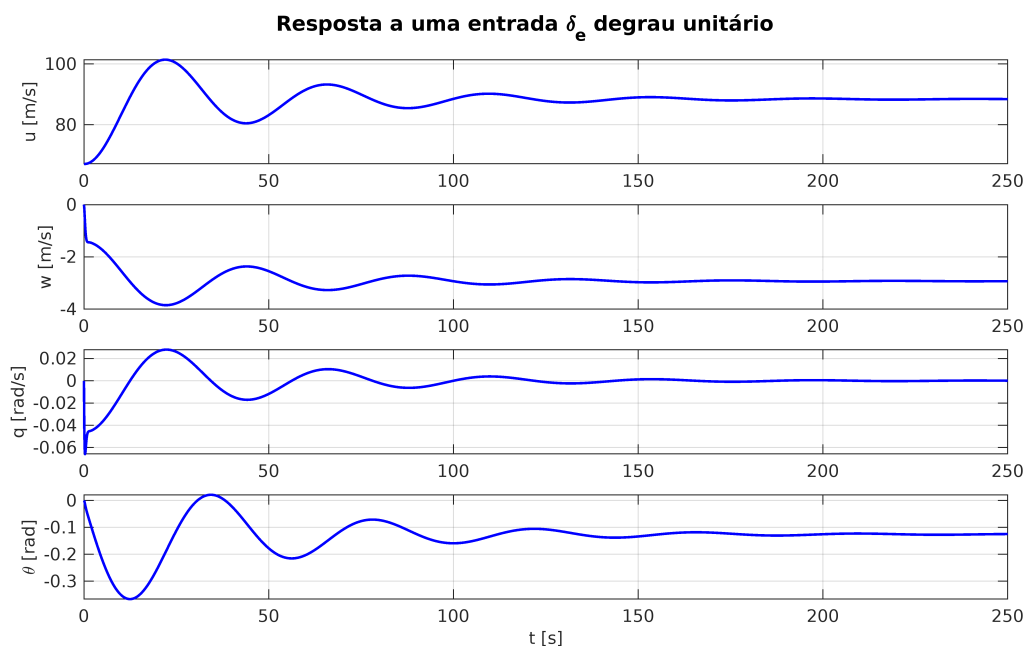
Table 4.6: Autovalor, e constante de tempo da rolagem reduzida.

Autovalor (λ)	Constante de tempo (rad/s)
-12.9726	0.077

4.2.4. Resposta no Tempo

A resposta no tempo a uma entrada degrau unitário no profundor foi obtida através da construção do espaço de estados completo segundo a Eq. 2.6, a partir das matrizes A_{lg} e B_{lg} previamente determinadas. Dado que os parâmetros das matrizes são expressos em radianos, a amplitude da entrada foi transformada de forma que ela correspondesse a 1° , pois foi considerado que uma deflexão de $1\ rad$ não seria condizente com a realidade de operação da aeronave.

Na Figura 4.2, são dispostos os resultados de cada um dos parâmetros no tempo, decorrentes da entrada aplicada sobre o modelo. Por meio dela, é possível atestar a estabilidade longitudinal da aeronave previamente aferida por meio dos autovalores da matriz A_{lg} . Observa-se a atenuação das amplitudes das oscilações de todos os estados ao longo do tempo até o atingimento da nova condição de equilíbrio, diferente da condição inicial.

**Figure 4.2:** Resposta dos estados no tempo, submetidos a δ_e degrau unitário.

4.2.5. Comparação entre Modelos Linearizado e Não Linearizado

Para complementar a análise da estabilidade dinâmica, a resposta no tempo da aeronave também foi obtida através do simulador de três graus de liberdade para o movimento longitudinal, baseado em um modelo não linearizado, desenvolvido em MATLAB & Simulink.

A Figura 4.3 apresenta a comparação da resposta dos estados longitudinais no tempo para uma entrada degrau unitário no profundor, obtida tanto pelo modelo linearizado quanto pelo simulador não linearizado.

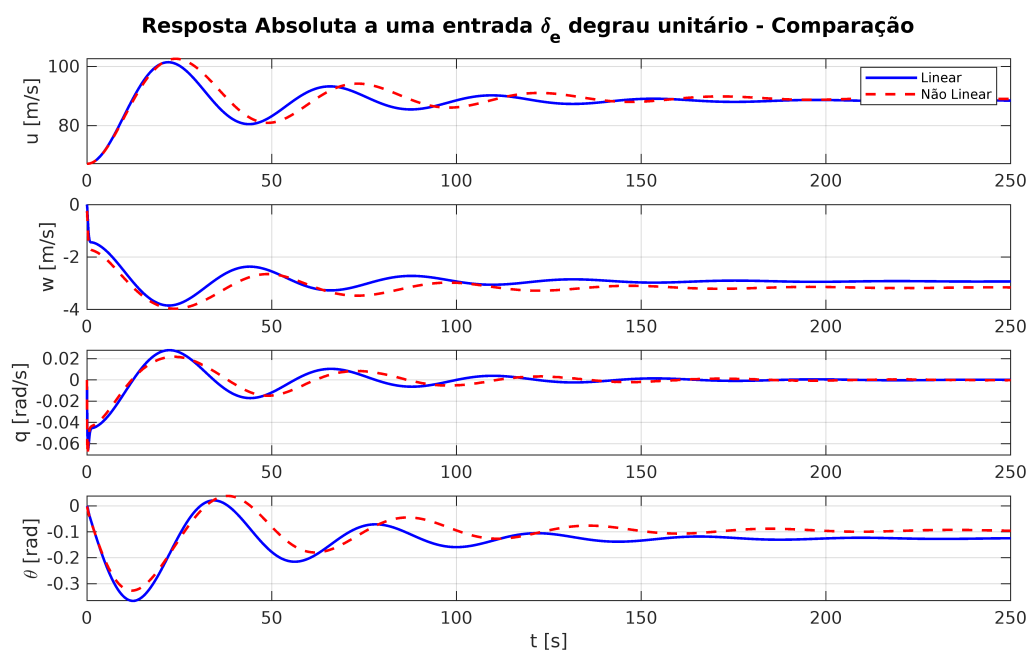


Figure 4.3: Resposta Absoluta a uma entrada δ_e degrau unitário - Comparação entre Modelos Linearizado e Não Linearizado.

Observa-se na Figura 4.3 uma relevante concordância entre as respostas dos dois modelos. Tanto as oscilações iniciais quanto o comportamento de decaimento para a nova condição de equilíbrio são muito similares para todas as variáveis de estado (u , w , q , e θ). Essa proximidade corrobora a validade da linearização realizada em torno do ponto de trimagem, indicando que, para as condições de perturbação e operação consideradas, o modelo linearizado é uma representação precisa da dinâmica longitudinal da aeronave. Pequenas diferenças podem ser atribuídas à natureza não linear do segundo modelo e aos efeitos que são desprezados na linearização, mas a consistência geral reforça a robustez da análise de estabilidade dinâmica apresentada.

5

Conclusões

Este trabalho teve como intuito a análise da estabilidade estática e dinâmica do avião Cessna 182, retirando os dados necessários do apêndice do Roskam, desenvolvendo uma rotina em MATLAB e um simulador projetado em Simulink para observar o comportamento da aeronave em determinadas situações.

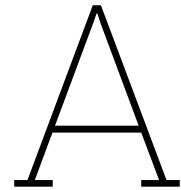
Na parte de estabilidade estática, foram construídos os diagramas de trimagem e, determinada a margem estática, localizou-se o ponto neutro da aeronave em relação ao *datum*. Já para dinâmica, a partir das derivadas de estabilidade fornecidas, montaram-se as matrizes que descrevem a dinâmica longitudinal e látero-direcional no espaço de estados. Calculando os autovalores de ambas as matrizes, classificou-se os modos dinâmicos quanto à estabilidade.

Finalmente, utilizando as mesmas derivadas de estabilidade no simulador de voo, testou-se a resposta do sistema a uma entrada degrau de 1° no profundor. Comparando as respostas obtidas a partir do simulador com a da matriz de estados, os resultados apresentaram boa concordância, confirmando a consistência da modelagem realizada. Todos os códigos utilizados para essas análises estão reunidos nos apêndices anexados ao final deste relatório.

Com isso, pode-se dizer que os resultados esperados no início do projeto foram atendidos, tendo em vista que a aeronave apresentou um comportamento estável, como esperado, e o sistema dinâmico e o simulador apresentaram respostas condizentes com a realidade.

References

- [1] Ricardo Afonso Angélico. *Notas sobre as notações utilizadas em dinâmica de voo*. Portuguese. Material didático interno. 2023.
- [2] Jan Roskam. *Airplane Flight Dynamics and Automatic Flight Controls – Part I*. 2nd ed. See Appendix B for detailed aerodynamic data and modeling references. Lawrence, Kansas: DARcorporation, 2007.
- [3] Textron Aviation. *Cessna Skylane*. Acesso em: 1 jul. 2025. 2024. URL: <https://cessna.txtav.com/en/piston/cessna-skylane>.



Códigos Implementados

A.1. Definição da estrutura params (params.m)

```
1 %% esse arquivo cria o arquivo aircraft.mat, que usaremos para montar as matizes
  de estabilidade longitudinal e laterodirecional,
2 %% e os dados nele presentes foram retirados da tabela B1do apendice B do roskam,
  e sua nomenclatura leva como padrão o adotado no apendice
3 %%
4 % declarando as condições de voo
5
6 % %cruise
7 params.flycond.altitude = 5000;% altitude, ft
8 params.flycond.mach = 0.201; %número de mach,M adm
9 params.flycond.airspeed = 220.1;% TAS, velocidade do ar verdadeira, ft/s
10 params.flycond.dinamicpressure = 49.6; %pressão dinamica qbarra, lbs/ft^2
11 params.flycond.AOAi = 0;% angulo de ataque alfa em graus
12 params.flycond.gravity = 32.174; %ft/s^2, gravidade
13
14
15 %corrigindo as unidades da condição de voo
16 params.flycond.altitudesi =params.flycond.altitude/3.28084 ;% altitude, m
17 params.flycond.machsi = 0.120; %número de mach,M adm
18 params.flycond.airspeedsi = params.flycond.airspeed/3.28084;% TAS, velocidade do
  ar verdadeira, m/s
19 params.flycond.dinamicpressuresi =params.flycond.dinamicpressure * 47.88 ; %
  pressão dinamica qbarra, Pa
20 params.flycond.AOAisi = params.flycond.AOAi*pi/180;% angulo de ataque alfa em rad
21 params.flycond.gravitysi = 9.81; %m/s^2, gravidade
22 params.flycond.rho = 2*params.flycond.dinamicpresuresi/(params.flycond.airspeedsi
  *params.flycond.airspeedsi); % (Kg/m^3) Densidade a 5000 fts
23
24 %%
25
26 %geometria de referência
27
28 params.area = 174;%ft^2, área alar
29 params.chord = 4.9; % ft, corda média aerodinâmica da asa , cbarra
30 params.wingspan = 36.0;%ft, envergadura da asa, b
31
32 %corrigindo as unidades da geometria
33
34 params.areas = params.area/10.7639 ;%m^2, area alar
35 params.chordsi = params.chord/3.28084 ; % m, corda média aerodinamica da asa ,
  cbarra
```

```

36 params.wingspan_si = params.wingspan/3.28084;%m, envergadura da asa, b
37
38 %%
39 %parâmetros de massa
40 params.mass = 2650; % lb, massa
41 params.Ixx = 948;% (slug*ft^2)
42 params.Iyy = 1346;% (slug*ft^2)
43 params.Izz = 1967;% (slug*ft^2)
44 params.Ixz = 0;% (slug*ft^2)
45 params.Xcg_adim = 26.4;% posição do cg, em fração da corda cbarr
46
47 %corrigindo as unidades dos parâmetros de massa
48
49 params.mass_si = params.mass/2.20462; % Kg, massa
50 params.Ixx_si = params.Ixx*1.3558;% (Kg*m^2)
51 params.Iyy_si = params.Iyy*1.3558;% (Kg*m^2)
52 params.Izz_si = params.Izz*1.3558;% (Kg*m^2)
53 params.Ixz_si = params.Ixz*1.3558;% ((Kg*m^2)
54 params.Xcg_adim_si = 26.4;% posição do cg, em fração da corda cbarr
55 %%
56 %derivadas adimensionalizadas na trimagem para essa condição de voo
57
58 params.aero.CD1 = 0.032;
59 params.aero.CL1 = 0.307 ;
60 params.aero.Cm1 = 0;
61 params.aero.CmT1 = 0;
62 params.engine.CTx1 = 0.032;
63
64
65 %derivadas adimensionalizadas tanto longitudinais quanto latero-direcionais, nessa
    ordem, dividido entre das superfícies de controle e aerodin
66
67 params.aero.CD0 = 0.0270 ;
68 params.aero.CDu = 0;
69 params.aero.CDalpha = 0.121 ;
70 params.aero.k = 0.032;% constante do arrasto induzido calculada a partir do cd e do
    cl.
71
72 params.aero.CL0 = 0.307 ;
73 params.aero.CLu = 0;
74 params.aero.CLalpha = 4.41 ;
75 params.aero.CLq = 3.9;
76 params.aero.CLalphadot = 1.7;
77
78 params.aero.Cm0 = 0.04;
79 params.aero.Cmu = 0;
80 params.aero.CmTu = 0;
81 params.aero.Cmalpha = -0.613;
82 params.aero.CmTalpha = 0;
83 params.aero.Cmq = -12.4;
84 params.aero.Cmalphadot = -7.27;
85
86 params.aero.CDde = 0;%1/rad
87 params.aero.CLde = 0.43;%1/rad
88 params.aero.Cmde = -1.122;%1/rad
89 params.aero.Chalpha = -0.0584;%1/rad
90 params.aero.Chde = -0.585;%1/rad
91
92 params.aero.Clbeta = -0.0923;
93 params.aero.Clp = -0.484 ;
94 params.aero.Clr = 0.0798;

```

```

95
96 params.aero.CYbeta = -0.393;
97 params.aero.CYp = -0.075 ;
98 params.aero.CYr = 0.214;
99
100 params.aero.Cnbeta = 0.0587;
101 params.aero.CnTbeta = 0;
102 params.aero.Cnp = -0.0278 ;
103 params.aero.Cnr = -0.0937 ;
104
105 params.aero.Clda = 0.229; %1/rad
106 params.aero.Cldr = 0.0147; %1/rad
107
108 params.aero.CYda = 0; %1/rad
109 params.aero.CYdr = 0.187; %1/rad
110
111 params.aero.Cnda = -0.0216; %1/rad
112 params.aero.Cndr = -0.0645; %1/rad
113
114 params.aero.Chda = -0.363; %1/rad
115 params.aero.Chbetar = 0.0819 ; %1/rad
116 params.aero.Chda = -0.567; %1/rad
117
118 %%
119 %parâmetros do motor
120
121 params.engine.CTxu = -0.096;
122 params.engine.epsilon_T = 0.0; % Assumido ângulo nulo entre o eixo do avião e o
    motor
123
124 %%
125 save('aircraft.mat', 'params') %cria o arquivo 'myaircraft_023.mat' onde as
    variáveis são salvas para simulação posterior

```

A.2. Análise estática da aeronave (longstat.m)

```

1 clear ~params
2 close all
3
4 %% 1. ANALYSIS OF deltae_trim AS A FUNCTION OF CL_trim
5
6 % 1.1 Interval of the analysis (CL_trim varies from a to b)
7 CL_interval = 0:.1:2;
8
9 % 1.2 Initialization of arrays where the results will be allocated
10 alpha_trim = zeros(length(CL_interval), 1);
11 deltae_trim = zeros(length(CL_interval), 1);
12
13 % 1.3 Initialization of a counter to be used as index in step 1.4
14 counter = 1;
15
16 % 1.4 Equationing and solutioning
17 for CL_trim = CL_interval
18     % System of equations in matrix form A * trim_condition = B, in which:
19     % A = [CLalpha CLde; Cmalpha Cmde]
20     % B = [CLtrim - CL0, -Cm0]
21     % trim_condition = [alpha_trim, deltae_trim], both in radians
22
23     A = [params.aero.CLalpha params.aero.CLde;
24         params.aero.Cmalpha params.aero.Cmde];

```

```

25
26     B = [CL_trim - params.aero.CL0;
27          -params.aero.Cm0];
28
29     trim_condition = A\B; % Solution (trim_condition = A^-1 * B)
30
31     % Assigns each solution to its respective array in position (counter),
32     % in degrees
33     alpha_trim(counter) = rad2deg(trim_condition(1));
34     deltae_trim(counter) = rad2deg(trim_condition(2));
35
36     % Increases the index count
37     counter = counter + 1;
38 end
39
40 % 1.5 Results plot (deltae_trim * CL_trim)
41 figure(Name='Análise da trimagem da aeronave',Position=[100 100 960 540])
42 plot(CL_interval, deltae_trim,...
43      'LineWidth', 1.5,...
44      'Color', 'blue')
45
46 grid on
47 xlabel('C_{L_{trim}} [-]', 'fontsize', 15)
48 ylabel('\delta_{e_{trim}} [\circ]', 'fontsize', 15)
49
50 % 1.6 Save plot
51 print('trimcond', '-dpng', '-r600')
52
53 %% 2. DETERMINATION OF THE AIRCRAFT'S STATIC MARGIN
54
55 % 2.1 Calculation of SM as -Cmalpha/CLalpha
56 SM = -params.aero.Cmalpha/params.aero.CLalpha * 100; % [%]
57
58 % 2.2 Determination of the neutral point's x coordinate
59 xCG = params.Xcg_adim/100 * params.chord; % [m]
60 xNP = xCG + SM/100 * params.chord; % [m]

```

A.3. Análise dinâmica da aeronave (dynamics.m)

```

1 load('aircraft.mat')
2
3 %% Load data
4
5 % Flight conditions
6 g = params.flycond.gravitysi;
7 dq = params.flycond.dinamicpressuresi;
8 U1 = params.flycond.airspeedsi;
9 theta1 = params.flycond.AOAisi;
10
11 % Physiscal Parameters
12 S = params.areas;
13 c = params.chordsi;
14 b = params.wingspansi;
15 m = params.masssi;
16 Ixx = params.Ixxsi;
17 Ixz = params.Ixzsi;
18 Iyy = params.Iyyi;
19 Izz = params.Izzsi;
20
21 % Adimensional derivatives

```

```

22 CD1 = params.aero.CD1;
23 CL1 = params.aero.CL1;
24 Cm1 = params.aero.Cm1;
25 CmT1 = params.aero.CmT1;
26
27 CTx1 = params.engine.CTx1;
28 CTxu = params.engine.CTxu;
29
30 CD0 = params.aero.CD0;
31 CDu = params.aero.CDu;
32 CDa = params.aero.CDalpha;
33
34 CL0 = params.aero.CL0;
35 CLu = params.aero.CLu;
36 CLa = params.aero.CLalpha;
37 CLq = params.aero.CLq;
38 CLadot = params.aero.CLalphadot;
39
40 Cm0 = params.aero.Cm0;
41 Cmu = params.aero.Cmu;
42 CmTu = params.aero.CmTu;
43 Cma = params.aero.Cmalpha;
44 CmTa = params.aero.CmTalpha;
45 CmQ = params.aero.CmQ;
46 Cmadot = params.aero.Cmalphadot;
47
48 CDde = params.aero.CDde;
49 CLde = params.aero.CLde;
50 Cmde = params.aero.Cmde;
51
52 CYb = params.aero.CYbeta;
53 CYP = params.aero.CYP;
54 CYr = params.aero.CYr;
55
56 Clb = params.aero.Clbeta;
57 Clp = params.aero.Clp;
58 Clr = params.aero.Clr;
59
60 Cnb = params.aero.Cnbeta ;
61 CnTb = params.aero.CnTbeta;
62 Cnp = params.aero.Cnp;
63 Cnr = params.aero.Cnr;
64
65 Clda = params.aero.Clda;
66 Clldr = params.aero.Clldr;
67
68 CYda = params.aero.CYda;
69 CYdr = params.aero.CYdr;
70
71 Cnda = params.aero.Cnda;
72 Cndr = params.aero.Cndr;
73
74 %% Longitudinal stability derivatives
75
76 % With respect to u
77
78 Xu = - dq * S * (CDu + 2 * CD1) / (m * U1);
79 XTu = dq * S * (CTxu + 2 * CTx1) / (m * U1);
80 Zu = - dq * S * (CLu + 2 * CL1) / (m * U1);
81 Mu = dq * S * c * Cmu / (Iyy * U1);
82 MTu = dq * S * c * CmTu / (Iyy * U1);

```



```

83
84 % With respect to angle of attack
85
86 Xa = - dq * S * (CDa - CL1) / m;
87 Za = - dq * S * (CLa + 2 * CD1) / m;
88 Ma = dq * S * c * Cma / Iyy;
89 MTa = dq * S * c * CmTa / Iyy;
90
91 % With respect to adot
92
93 Zadot = - dq * S * c * CLadot / (2 * m * U1);
94 Madot = dq * S * c * c * Cmadot / (2 * Iyy * U1);
95
96 % With respect to q
97
98 Zq = - dq * S * c * CLq / (2 * m * U1);
99 Mq = dq * S * c * c * Cm q / (2 * Iyy * U1);
100
101 %% Longitudinal control derivatives
102
103 Xde = - dq * S * CDde / m;
104 Zde = - dq * S * CLde / m;
105 Mde = dq * S * c * Cmde / Iyy;
106
107 %% Lateral-Directional stability derivatives
108
109 % With respect to b
110
111 Yb = dq * S * CYb / m;
112 Lb = dq * S * b * Clb / Ixx;
113 Nb = dq * S * b * Cnb / Izz;
114 NTb = dq * S * b * CnTb / Izz;
115
116 % With respect to p
117
118 Yp = dq * S * CYp / (2 * m * U1);
119 Lp = dq * S * b * b * Clp / (2 * Ixx * U1);
120 Np = dq * S * b * b * Cnp / (2 * Izz * U1);
121
122 % With respect to r
123
124 Yr = dq * S * b * CYr / (2 * m * U1);
125 Lr = dq * S * b * b * Clr / (2 * Ixx * U1);
126 Nr = dq * S * b * b * Cnr / (2 * Izz * U1);
127
128 %% Lateral-Directional control derivatives
129
130 % With respect to da
131
132 Yda = dq * S * CYda / m;
133 Lda = dq * S * b * b * Cl da / Ixx;
134 Nda = dq * S * b * b * Cn da / Izz;
135
136 % With respect to dr
137
138 Ydr = dq * S * CYdr / m;
139 Ldr = dq * S * b * b * Cl dr / Ixx;
140 Ndr = dq * S * b * b * Cn dr / Izz;
141
142 %% Complete Model Longitudinal Dynamics
143

```

```

144 Alg = [Xu + XTu,          Xa / U1,          0,
          - g * cos(theta1);
145      U1 * Zu / (U1 - Zdot),  Za / (U1 - Zdot),
          U1 * (U1 + Zq) / (U1 - Zdot),          - U1 * g * sin(theta1) / (
          U1 - Zdot);
146      Mu + MTu,          ((Ma + MTa) / U1) + (Madot * Za / (U1 * (U1 -
          Zdot))), Mq + (Madot * (Zq + U1) / (U1 - Zdot)),          - Madot * g * sin
          (theta1) / (U1 - Zdot);
147      0,          0,
          1,
          0];

148
149 Blg = [Xde;
150      Zde / (U1 - Zdot);
151      Mde + (Madot * Zde / (U1 - Zdot));
152      0];
153
154 % Complete Model Longitudinal dynamic stability analysis
155
156 lambda_lg = eig(Alg)
157
158 if real(lambda_lg) < 0
159     disp('The aircraft is stable')
160 else
161     disp('The aircraft is unstable')
162 end
163
164 % Identify dynamic modes eigvalues
165 [~, idx_ph] = min(abs(imag(lambda_lg)));
166 [~, idx_sp] = max(abs(imag(lambda_lg)));
167
168 lambda_ph = lambda_lg(idx_ph);
169 lambda_sp = lambda_lg(idx_sp);
170
171 % Natural frequencies and damping factor
172 omegan_ph = sqrt(real(lambda_ph)^2 + imag(lambda_ph)^2);
173 zeta_ph = - real(lambda_ph) / omegan_ph;
174
175 omegan_sp = sqrt(real(lambda_sp)^2 + imag(lambda_sp)^2);
176 zeta_sp = - real(lambda_sp) / omegan_sp;
177
178 %% Complete Model Lateral-Directional Dynamics
179
180 % Inertial coupling parameters
181
182 ix = Ixz / Ixx;
183 iz = Ixz / Izz;
184
185 Ald = [Yb / U1,          Yr - U1,          Yp,          g * cos(theta1);
186      (Lb + ix * Nb + ix * NTb) / (U1 * (1 - ix * iz)), (Lp + ix * Np) / (1 - ix
          * iz), (Lr + ix * Nr) / (1 - ix * iz), 0;
187      (iz * Lb + Nb + NTb) / (U1 * (1 - ix * iz)), (iz * Lp + Np) / (1 - ix
          * iz), (iz * Lr + Nr) / (1 - ix * iz), 0;
188      0,          0,          1,
          0];

189
190 Bld = [Yda,          Ydr;
191      (Lda + ix * Nda) / (1 - ix * iz), (Ldr + ix * Ndr) / (1 - ix * iz);
192      (iz * Lda + Nda) / (1 - ix * iz), (iz * Ldr + Ndr) / (1 - ix * iz);

```

```

193         0,                                0 ];
194
195 % Complete Model Lateral-Directional dynamic stability analysis
196
197 lambda_ld = eig(Ald)
198
199 if real(lambda_ld) < 0
200     disp('The aircraft is stable')
201 else
202     disp('The aircraft is unstable')
203 end
204
205 % Identify dynamic modes eigvalues
206 [~, idx_dr] = max(abs(imag(lambda_ld)));
207 [~, idx_rl] = max(abs(real(lambda_ld)));
208 [~, idx_s] = min(abs(real(lambda_ld)));
209
210 lambda_dr = lambda_lg(idx_dr);
211 lambda_rl = lambda_lg(idx_rl);
212 lambda_s = lambda_lg(idx_s);
213
214 % Natural frequencies and damping factor
215 omegan_dr = sqrt(real(lambda_dr)^2 + imag(lambda_dr)^2);
216 zeta_dr = - real(lambda_dr) / omegan_dr;
217
218 %% Short-period Reduced Model Dynamics
219
220 Asp = [Za / (U1 - Zdot),                                U1 * (U1 + Zq) / (
221         U1 - Zdot);
222        ((Ma + MTa) / U1) + (Madot * Za / (U1 * (U1 - Zdot))), Mq + (Madot * (Zq +
223         U1) / (U1 - Zdot))];
224
225 Bsp = [Zde/(U1 - Zdot);
226        Mde + (Madot * Zde / (U1 - Zdot))];
227
228 lambda_sp_red = eig(Asp)
229
230 % Natural frequencies and damping factor
231 omegan_sp_red = sqrt(real(lambda_sp_red(1))^2 + imag(lambda_sp_red(1))^2);
232 zeta_sp_red = - real(lambda_sp_red(1)) / omegan_sp_red;
233
234 %% Dutch-Roll Reduced Model Dynamics
235
236 Adr = [Yb / U1,                                Yr-U1;
237        (iz * Lb + Nb + NTb) / (U1 * (1 - iz * iz)), (iz * Lr + Nr) / (1 - iz * iz
238        )];
239
240 Bdr = [Yda,                                Ydr;
241        (iz * Lda + Nda) / (1 - iz * iz), (iz * Ldr + Ndr) / (1 - iz * iz)];
242
243 lambda_dr_red = eig(Adr)
244
245 % Natural frequencies and damping factor
246 omegan_dr_red = sqrt(real(lambda_dr_red(1))^2 + imag(lambda_dr_red(1))^2);
247 zeta_dr_red = - real(lambda_dr_red(1)) / omegan_dr_red;
248
249 %% Roll Reduced Model Dynamics
250
251 Arl = (Lp+iz*Np)/(1-iz*iz);
252
253 Brl = [(Lda + iz * Nda) / (1 - iz * iz), (Ldr + iz * Ndr) / (1 - iz * iz)];

```

```
251
252 lambda_rl_reduced = Arl;
```

A.4. Resposta da aeronave a uma entrada degrau unitário no profundor (stepresponse.m)

```
1 close all
2
3 %% 1. STEP RESPONSE - LINEAR MODEL
4
5 % 1.1 Construction of the state-space system (xdot = A*x + B*u; y = x)
6 sys = ss(Alg, Blg, eye(4), 0);
7
8 % 1.2 Configuration of the unitary step (amplitude = 1 degree)
9 Config = RespConfig(Amplitude=deg2rad(1));
10
11 % 1.3 Application of the step as input to the system
12 [resp, tOut] = step(sys, 0:.1:250, Config);
13
14 % 1.4 Extract variations (deltas) from linear model
15 du = resp(:,1); % [m/s]
16 dw = resp(:,2); % [m/s]
17 dq = resp(:,3); % [rad/s]
18 dtheta = resp(:,4); % [rad]
19
20 % 1.5 Plot of the delta variations (only linear)
21 yld = {'\Delta_u[m/s]', '\Delta_w[m/s]', ...
22        '\Delta_q[rad/s]', '\Delta_theta[rad]'};
23 plot_step_results(tOut, du, dw, dq, dtheta, 'unitstepdelta', yld)
24
25 %% 2. ABSOLUTE VALUES - COMPARISON LINEAR VS NONLINEAR
26
27 % 2.1 Initial conditions (linear model)
28 u0 = params.flycond.airspeedsi * cosd(params.flycond.AOAi);
29 w0 = params.flycond.airspeedsi * sind(params.flycond.AOAi);
30 q0 = 0;
31 theta0 = params.flycond.AOAi;
32
33 % 2.2 Absolute response - linear
34 u_lin = u0 + du;
35 w_lin = w0 + dw;
36 q_lin = q0 + dq;
37 theta_lin = theta0 + dtheta;
38
39 % 2.3 Load nonlinear data
40 load('dados_simulacao.mat'); % carrega t, u, w, q, theta (valores absolutos)
41
42 % 2.4 Plot comparison
43 figure('Name','Resposta_Absoluta_Linear_vs_Não_Linear','Position',[100 100 960
44    540])
45 sgtitle('Resposta_Absoluta_a_uma_entrada_delta_degrau_unitário_Comparação','
46    Fontsize',12,'FontWeight','bold')
47
48 subplot(4,1,1)
49 plot(tOut, u_lin, 'b', 'LineWidth', 1.5); hold on
50 plot(t, u, '--r', 'LineWidth', 1.5)
51 ylabel('u[m/s]'); grid on
52 legend('Linear', 'Não_Linear')
```

```

52 subplot(4,1,2)
53 plot(tOut, w_lin, 'b', 'LineWidth', 1.5); hold on
54 plot(t, w, '--r', 'LineWidth', 1.5)
55 ylabel('w[m/s]'); grid on
56
57 subplot(4,1,3)
58 plot(tOut, q_lin, 'b', 'LineWidth', 1.5); hold on
59 plot(t, q, '--r', 'LineWidth', 1.5)
60 ylabel('q[rad/s]'); grid on
61
62 subplot(4,1,4)
63 plot(tOut, theta_lin, 'b', 'LineWidth', 1.5); hold on
64 plot(t, theta, '--r', 'LineWidth', 1.5)
65 ylabel('\theta[rad]'); xlabel('t[s]'); grid on
66
67 print('comparacao_modelos_abs','-dpng','-r600')
68
69 %% FUNCTION - DELTA PLOT ONLY (LINEAR)
70 function plot_step_results(tf, uf, wf, qf, thetfa, filenamef, y_label)
71
72     figure(Name='Resposta_no_tempo_ao_degrau', Position=[100 100 960 540])
73
74     sgtitle('Resposta_ao_degrau-Modelo_Linearizado(variações)',...
75         'FontSize', 12, 'FontWeight', 'bold')
76
77     subplot(4,1,1)
78     plot(tf, uf, 'b', 'LineWidth', 1.5)
79     xlim([0 max(tf)]); grid on
80     ylabel(y_label{1}, 'fontsize', 10)
81
82     subplot(4,1,2)
83     plot(tf, wf, 'b', 'LineWidth', 1.5)
84     xlim([0 max(tf)]); grid on
85     ylabel(y_label{2}, 'fontsize', 10)
86
87     subplot(4,1,3)
88     plot(tf, qf, 'b', 'LineWidth', 1.5)
89     xlim([0 max(tf)]); grid on
90     ylabel(y_label{3}, 'fontsize', 10)
91
92     subplot(4,1,4)
93     plot(tf, thetfa, 'b', 'LineWidth', 1.5)
94     xlim([0 max(tf)]); grid on
95     ylabel(y_label{4}, 'fontsize', 10)
96     xlabel('t[s]', 'fontsize', 10)
97
98     print(filenamef, '-dpng', '-r600')
99 end

```

A.5. Preparação da função para ser resolvida pelo ODE45 (LongSimulator.m)

```

1 %% Preparando a função pra passar pro ODE45
2 function xdot = LongSimulator(t, x, params, input)
3     u = x(1);
4     w = x(2);
5     q = x(3);
6     theta = x(4);
7     delta_e = input(1);
8     Uinf = sqrt(u^2 + w^2);
9     alpha = atan2(w, u);

```

```

10     alphad_new = 0;
11     alphad = 1;
12     tol_alphad = 1e-6;
13     while abs(alphad_new - alphad) > tol_alphad
14         alphad = alphad_new;
15         [CD,CL,CM] = aerocoeff(params, alpha, alphad, Uinf, q, delta_e);
16
17         coefs = [CD,CL,CM];
18
19         [X,Z,M] = forcesmoments(params, alpha, coefs, Uinf, params.engine.
20             T_trim_value);
21
22         f_m = [X,Z,M];
23
24         xdot = motion(params, f_m, x);
25
26         alphad_new = (xdot(2) * u - w * xdot(1)) / (u * u);
27     end
28 %% Módulo A (coeficientes aerodinamicos)
29 function [CD, CL, CM] = aerocoeff(params, alpha, alphad, Uinf, q, delta_e)
30     c = params.chordsi;
31     AoA = params.flycond.AOAisi;
32     CL0 = params.aero.CL0;
33     CLa = params.aero.CLalpha;
34     CLad = params.aero.CLalphadot;
35     CLq = params.aero.CLq;
36     CLde = params.aero.CLde;
37     CL = CL0 + CLa * alpha + (CLad * alphad + CLq * q) * ( c / (2*Uinf)) + CLde *
38         delta_e ;
39     CM0 = params.aero.Cm0;
40     CMa = params.aero.Cmalpha;
41     CMad = params.aero.Cmalphadot;
42     CMq = params.aero.Cmq;
43     CMde = params.aero.Cmde;
44     CM = CM0 + CMa * alpha + (CMad * alphad + CMq * q) * ( c / (2*Uinf)) + CMde *
45         delta_e ;
46     k = params.aero.k;
47     CD0 = params.aero.CD0;
48     CD = CD0 + k * CL * CL ;
49 end
50 %% Módulo B (forças e momentos)
51 function [X, Z, M] = forcesmoments(params, alpha, coefs, Uinf, T_applied_value)
52     S = params.areasi;
53     c = params.chordsi;
54     rho = params.flycond.rho;
55     eps_T = params.engine.epsilon_T;
56     D = 0.5 * rho * Uinf * Uinf * S * coefs(1);
57     L = 0.5 * rho * Uinf * Uinf * S * coefs(2);
58     M = 0.5 * rho * Uinf * Uinf * S * coefs(3) * c;
59     T = T_applied_value;
60     X = T * cos(eps_T + alpha) - D * cos(alpha) + L * sin(alpha);
61     Z = -T * sin(eps_T + alpha) - D * sin(alpha) - L * cos(alpha);
62 end
63 %% Módulo C (movimento)
64 function xdot = motion(params, f_m, x)
65     g = params.flycond.gravitysi;
66     m = params.masssi;
67     Iyy = params.Iyysi;
68     X = f_m(1);
69     Z = f_m(2);

```

```

68     M = f_m(3);
69     u = x(1);
70     w = x(2);
71     q = x(3);
72     theta = x(4);
73     xe = x(5);
74     ze = x(6);
75     udot = X / m - g * sin(theta) - q * w ;
76     wdot = Z / m + g * cos(theta) + q * u ;
77     qdot = M / Iyy ;
78     thetadot = q ;
79     xedot = u * cos(theta) + w * sin(theta);
80     zedot = -u * sin(theta) + w * cos(theta);
81     xdot = [udot, wdot, qdot, thetadot, xedot, zedot];
82 end

```

A.6. Ciclo principal da simulação (MainSimulator.m)

```

1 %% Função principal
2 function [params, response] = MainSimulator
3
4     clear all
5     close all
6     clc
7     load('aircraft.mat');
8
9     [alpha_trim, delta_trim, T_calculated_trim] = trim_aircraft(params);
10
11     params.engine.T_trim_value = T_calculated_trim;
12
13     u0 = params.flycond.airspeedsi * cos(alpha_trim)
14     w0 = params.flycond.airspeedsi * sin(alpha_trim)
15     q0 = 0.0
16     theta0 = alpha_trim
17     x0 = 0;
18     z0 = -params.flycond.altitudesi;
19
20     x = [u0, w0, q0, theta0, x0, z0];
21
22     input = [0.0*delta_trim];
23
24     xdot = @(t,x)transpose(LongSimulator(t,x,params,input));
25
26     [t_sim, x_sim] = ode45(xdot, [0 250], x);
27
28 end
29
30 %% Trimagem
31 function [alpha_trim, delta_trim, T_trim] = trim_aircraft(params)
32     c = params.chordsi;
33     S = params.areasi;
34     rho = params.flycond.rho;
35     Uinf = params.flycond.airspeedsi;
36     m = params.masssi;
37     g = params.flycond.gravitysi;
38     CL0 = params.aero.CL0;
39     CLa = params.aero.CLalpha;
40     CLde = params.aero.CLde;
41     CM0 = params.aero.Cm0;
42     CMa = params.aero.Cmalpha;

```

```
43 CMde = params.aero.Cmde;
44
45 CLtrim = (m * g) / (0.5 * rho * Uinf^2 * S)
46 M_matrix = [Cma, CMde; CLa, CLde];
47 f_vector = [-CM0; CLtrim - CL0];
48 resp = M_matrix \ f_vector;
49 alpha_trim = resp(1);
50 delta_trim = resp(2);
51 CDtrim = params.aero.CD0 + params.aero.k * CLtrim^2;
52 D = 0.5 * rho * Uinf^2 * S * CDtrim;
53 L = 0.5 * rho * Uinf^2 * S * CLtrim;
54 T_trim = -L * sin(alpha_trim) + D * cos(alpha_trim)
55 end
```

A.7. Extração de dados do Simulink (scopedata.m)

```
1 % Extrair timeseries de cada elemento do Dataset
2 q_data = out.qscope.getElement(1).Values;
3 theta_data = out.thetascope.getElement(1).Values;
4 u_data = out.uscope.getElement(1).Values;
5 w_data = out.wscope.getElement(1).Values;
6
7 % Tempo global da simulação, que é o mesmo para todos os dados
8 t = out.tout;
9
10 q = q_data.Data;
11 theta = theta_data.Data;
12 u = u_data.Data;
13 w = w_data.Data;
14
15 save('dados_simulacao.mat', 'q', 'theta', 'u', 'w', 't');
```