Thierry Miquel, Antoine Drouin

2016

IENAC 1ére année
Représentation et analyse des systèmes dynamiques
Etude de synthèse

Représentation et analyse des systèmes dynamiques: Étude de synthèse

Antoine Drouin et Thierry Miquel
Département Transport Aérien - ENAC

17 mai 2016

1 Introduction **2**
1.1 Objectifs . . . 2
1.2 Pré-requis . . . 2
1.3 Travail à effectuer . . . 2

2 Modélisation de la dynamique longitudinale d'un avion **3**
2.1 Hypothèses et notations . . . 3
2.2 Modèle atmosphérique . . . 3
2.3 Poussée . . . 4
2.4 Expression de la portance, de la trainée et du moment de tangage . . . 4
2.4.1 Portance . . . 5
2.4.2 Trainée . . . 5
2.4.3 Expression du moment de tangage . . . 6
2.5 Modèle d'état . . . 7
2.6 Paramètres . . . 8

3 Séance 1 : Analyse des coefficients aérodynamiques **9**
3.1 Travail hors séances encadrées . . . 9
3.2 Séance encadrée . . . 9

4 Séance 2 : Linéarisation de la dynamique **9**
4.1 Travail hors séances encadrées . . . 9
4.2 Séance encadrée . . . 9

5 Séance 3 : Réponse longitudinale **10**
5.1 Travail hors séances encadrées . . . 10
5.2 Séance encadrée . . . 10

6 Annexes **11**
6.1 Annexe 1 : dynamic.py . . . 11
6.2 Annexe 1 : utils.py . . . 14

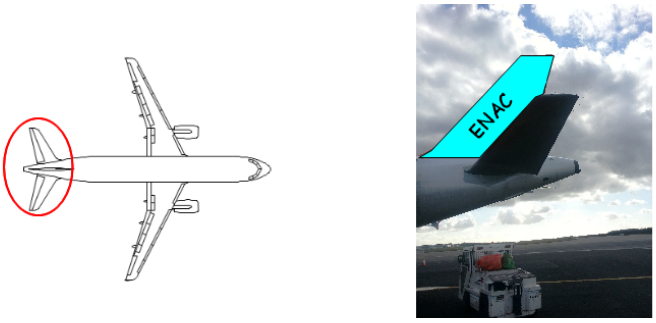

FIGURE 1 – Gouverne de profondeur et Plan Horizontal Réglable (PHR)

1 Introduction

Pour piloter un avion dans le plan vertical le pilote actionne le manche qui provoque un mouvement de tangage grâce à la gouverne de profondeur qui est située à l'arrière de l'empennage horizontal (cf. figure 1). En plus de la gouverne de profondeur les avions de transport commercial possèdent le plus souvent un Plan Horizontal Réglable (PHR ou en anglais THS : Trimmable Horizontal Stabiliser) qui reprend en permanence les efforts en tangage de la gouverne de profondeur de manière à toujours laisser au neutre la position du manche lors des phases équilibrées du vol. Le PHR est déplacé par l'intermédiaire d'une vis sans fin et est actionné soit manuellement par le pilote soit par le calculateur de commandes du vol.

Sur la figure 1 sont entourés la gouverne de profondeur et le Plan Horizontal Réglable (PHR) d'un avion de transport commercial.

1.1 Objectifs

L'objectif de ce projet est de simuler et d'analyser les effets de la masse, de l'altitude et du centrage sur les principales caractéristiques de la dynamique longitudinale d'un avion de transport commercial équilibré (*trimé*) grâce au PHR.

1.2 Pré-requis

Ce projet s'appelle *étude de synthèse* car les pré-requis pour mener à bien ce projet sont multidisciplinaires : vous mettrez en application vos connaissances acquises durant les cours de mécanique du vol, de représentation et analyse des systèmes dynamiques, d'analyse numérique ainsi que le cours de programmation en Python.

1.3 Travail à effectuer

Le travail à effectuer consiste à répondre aux questions en complétant les fichiers Python qui sont fournis (ou que vous pouvez éventuellement créer vous même) et à préparer une présentation orale sur l'ensemble de cette étude. En fin de projet vous présenterez pendant 15 minutes par équipe une partie de l'étude et serez interrogés sur son ensemble. Chaque groupe de TD sera divisé en équipes de trinôme (équipe Alpha, Bravo, Charlie, Delta, Echo, Foxtrot) qui étudiera un avion spécifique dont les caractéristiques sont données dans la section 2.6. Pour un même groupe de TD les avions choisis devront être différents.

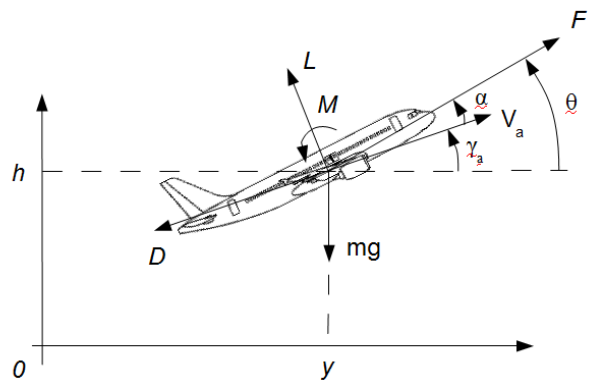

FIGURE 2 – Avion évoluant dans le plan vertical

2 Modélisation de la dynamique longitudinale d'un avion

2.1 Hypothèses et notations

Nous considérons un avion évoluant selon le plan vertical comme représenté sur la figure 2. Nous noterons par la suite :

— y la position horizontale de l'avion exprimée en m ;
— h l'altitude pression exprimée en m ;
— m la masse de l'avion exprimée en kg et que nous supposerons constante ;
— g l'accélération de la pesanteur : $g = 9.80665\ m/s^2$;
— V_a la vitesse aérodynamique de l'avion exprimée en m/s ;
— L (lift) la force de portance, exprimée en N et perpendiculaire à la vitesse V_a ;
— D (drag) la force de trainée, exprimée en N et orientée à l'opposé de la vitesse V_a ;
— F la force de poussée issue des réacteurs, exprimée en N ;
— M le moment de tangage, exprimé en Nm
— α l'incidence aérodynamique de l'avion exprimée en rad ;
— θ l'assiette de l'avion exprimée en rad ;
— q la vitesse de tangage de l'avion exprimée en rad/sec :

$$q = \dot{\theta} \tag{1}$$

— γ_a la pente de l'avion exprimée en rad :

$$\gamma_a = \theta - \alpha \tag{2}$$

— $\bar{c}$ la corde de référence de l'aile (corde aérodynamique moyenne) exprimée en m ;

2.2 Modèle atmosphérique

Nous utiliserons le modèle d'atmosphère standard (ISA, International Standard Atmosphere) pour lequel la température varie linéairement (loi de Toussaint) avec l'altitude jusqu'à 11 km d'altitude (c'est à dire dans la troposphère). Les notations sont les suivantes :

— T est la température qui dépend de l'altitude h.

$$T = T_0 + T_h h \quad \begin{cases} T_0 = 288.15\ K \\ T_h = -0.0065\ K/m \end{cases} \tag{3}$$

— R_s est la constante spécifique de l'air qui dépend de son humidité. Pour un air sec nous avons

$$R_s = 287.05\ m^2/(Ks^2) \tag{4}$$

— ρ_0 est la masse volumique de l'air au sol exprimée en kg/m^3

$$\rho_0 = 1.225\ kg/m^3 \tag{5}$$

— ρ la masse volumique de l'air exprimée en kg/m^3. La masse volumique de l'air sec est donnée par la relation suivante

$$\rho = \rho_0 \left(\frac{T}{T_0}\right)^{-\frac{g}{R_s T_h}-1} = \rho_0 \left(1 + \frac{T_h}{T_0} h\right)^{-\frac{g}{R_s T_h}-1} \tag{6}$$

Notez la dépendance de la masse volumique ρ avec la constante spécifique de l'air R_s, et donc avec son humidité.

— κ est le coefficient de compressibilité de l'air

$$\kappa = 1.4 \tag{7}$$

— M_a est le Mach de vol ($M_a < 0.9$). Il est relié à la vitesse aérodynamique V_a et à la température T par la relation suivante :

$$M_a = \frac{V_a}{\sqrt{\kappa R_s T}} \tag{8}$$

2.3 Poussée

La force de poussée F des moteurs est exprimée en N et est supposée parallèle à l'axe $\vec{i}_b$ du trièdre avion. Nous utiliserons le modèle proposé par J.Mattingly (*Jack D. Mattingly, William H. Heiser, Daniel H. Daley, Aircraft Engine Design. AIAA Education series, 1987. ISBN : 0-930403-23-1*) pour un réacteur double flux (turbofan) ayant un fort taux de dilution :

$$F = F_0 \left(\frac{\rho}{\rho_0}\right)^{0.6} \left(0.568 + 0.25\,(1.2 - M_a)^3\right) \delta_{th} \tag{9}$$

Où :

— F_0 est la poussée maximale au sol et à l'arrêt, exprimée en N

— δ_{th} est la position de la manette des gaz : $0 \leq \delta_{th} \leq 1$

2.4 Expression de la portance, de la trainée et du moment de tangage

Les valeurs numériques qui sont fournies ci-après sont données à titre indicatif et ne représentent en rien des données constructeurs.

2.4.1 Portance

Nous considérerons que la force de portance L, exprimée en N et orientée perpendiculairement à la vitesse, a l'expression suivante :

$$\left\{\begin{array}{l} L = \frac{1}{2}\rho V_a^2 S C_L \\ C_L = C_{L0} + C_{L\alpha}\alpha + C_{L\delta_{PHR}}\delta_{PHR} + C_{Lq}\frac{q}{V_a} \end{array}\right. \tag{10}$$

Avec :

$$\left\{\begin{array}{l} C_{L0} = \left(\frac{S_t}{S}\frac{d\epsilon}{d\alpha}C_{Lt\alpha} - C_{Lwb\alpha}\right)\alpha_0 \\ C_{L\alpha} = C_{Lwb\alpha} + \frac{S_t}{S}C_{Lt\alpha}\left(1 - \frac{d\epsilon}{d\alpha}\right) \\ C_{L\delta_{PHR}} = \frac{S_t}{S}C_{Lt\alpha} \\ C_{Lq} = l_t\frac{S_t}{S}C_{Lt\alpha}C_{Ltq} \\ \frac{d\epsilon}{d\alpha} = 0.25 \\ C_{Ltq} = 1.3 \end{array}\right. \tag{11}$$

Où :

— δ_{PHR} est l'angle de braquage de l'empennage horizontal exprimé en rad ;

— α_0 est l'incidence pour laquelle la portance de l'ensemble {aile (wing), fuselage, dérive, moteur (body)} est nulle. Nous prendrons par la suite :

$$\alpha_0 = -2\frac{\pi}{180}\ rad \tag{12}$$

— S est surface portante de référence exprimée en m^2 ;

— S_t la surface portante de l'empennage horizontal exprimée en m^2.

— $C_{Lwb\alpha}$ est le gradient de portance de l'ensemble {aile (wing), fuselage, dérive, moteur (body)}. Pour un profil mince l'approximation suivante est souvent utilisée (cf. cours de mécanique du vol) :

$$C_{Lwb\alpha} \approx \pi\frac{\lambda}{1 + \sqrt{1 + \left(\frac{\lambda}{2}\right)^2}} \tag{13}$$

— λ est l'allongement de l'aile (en anglais AR pour Aspect Ratio) ($\lambda = \frac{b^2}{S}$: c'est le carré de l'envergure b divisée par la surface portante S) ;

— $C_{Lt\alpha}$ est le gradient de portance de l'empennage horizontal. Nous prendrons pour son expression la même que celle de $C_{Lwb\alpha}$ où λ_t est l'allongement de l'empennage horizontal :

$$C_{Lt\alpha} \approx \pi\frac{\lambda_t}{1 + \sqrt{1 + \left(\frac{\lambda_t}{2}\right)^2}} \tag{14}$$

— l_t est la distance entre le foyer de l'empennage horizontal et le foyer de l'ensemble {aile (wing), fuselage, dérive, moteur (body)}. Nous approximerons très grossièrement l_t par un pourcentage de la longueur du fuselage, notée L_{fus} :

$$l_t \approx 0.5 L_{fus} \tag{15}$$

2.4.2 Trainée

Nous considérerons que la force D de trainée, exprimée en N et orientée à l'opposée de la vitesse, a l'expression suivante :

$$
\begin{cases}
D = \frac{1}{2}\rho V_a^2 S C_D \\
C_D \approx C_{D0} + k_i C_L^2 \\
k_i = \frac{1}{e}\frac{1}{\pi\lambda}
\end{cases}
\tag{16}
$$

Où :

— C_D est le coefficient de trainée

— C_{D0} est le coefficient de traînée de profil (traînée de forme du profil et traînée de frottement). Nous prendrons par la suite :

$$
C_{D0} = 0.025 \tag{17}
$$

— $k_i C_{Lwb}^2$ est le coefficient de traînée induite, conséquence de la portance. Le coefficient e qui intervient dans l'expression de k_i est le coefficient d'Oswald. Il est égal à 1 pour une aile de forme elliptique (cf. cours d'aérodynamique). Nous prendrons par la suite :

$$
e = 1 \tag{18}
$$

2.4.3 Expression du moment de tangage

Le moment de tangage M par rapport au centre de gravité de l'avion et exprimé en $N.m$ a l'expression suivante :

$$
\begin{cases}
M = \frac{1}{2}\rho V_a^2 S\bar{c} C_m \\
C_m = C_{m0} + C_{m\alpha}\left(\alpha - \alpha_0\right) + C_{m\delta}\delta_{PHR} + C_{mq}\frac{q l_t}{V_a}
\end{cases}
\tag{19}
$$

Où :

— C_m est le coefficient du moment de tangage

— C_{m0} est le coefficient de couple de tangage. Il est nul pour un profil symétrique, comme c'est le cas pour l'empennage horizontal. La valeur négative de ce coefficient indique un couple à piquer, comme c'est le cas pour un profil cambré. Nous prendrons par la suite :

$$
C_{m0} = -0.59 \tag{20}
$$

— $C_{m\alpha}$ est le gradient du moment de tangage

$$
C_{m\alpha} = -m_s C_{Lwb\alpha} \tag{21}
$$

— m_s désigne la marge statique de l'avion. Elle est définie par la relation suivante où x_F est la position du foyer de l'avion et x_G la position de son centre de gravité :

$$
m_s = \frac{x_G - x_F}{\bar{c}} \tag{22}
$$

Notons que la marge statique sera positive lorsque l'avion est statiquement stable et négative dans le cas contraire.

— V_t est *volume d'empennage* qui est défini par la relation :

$$
V_t = \frac{l_t S_t}{\bar{c} S} \tag{23}
$$

— Les coefficients $C_{m\delta}$ et C_{mq} sont définis comme suit à partir des coefficients $C_{Lt\alpha}$ et C_{Ltq} et du *volume d'empennage* V_t :

$$
\begin{cases}
C_{m\delta} = -V_t C_{Lt\alpha} \\
C_{mq} = -V_t C_{Lt\alpha} C_{Ltq}
\end{cases}
\tag{24}
$$

2.5 Modèle d'état

Les forces et les moments qui s'exercent sur l'avion s'expriment naturellement dans des repères spécifiques.

Nous considérerons que la Terre est plate et que c'est un référentiel inertiel. Nous y associons un trièdre direct, noté $\mathcal{R}_i = \{\vec{i}_i, \vec{j}_i, \vec{k}_i\}$, dans lequel le vecteur unitaire $\vec{k}_i$ est parallèle à la verticale ascendante du lieu. Dans ce repère, le poids de l'avion a pour expression :

$$m\vec{g} = -mg\vec{k}_i \tag{25}$$

Le trièdre aérodynamique, noté $\mathcal{R}_a = \{\vec{i}_a, \vec{j}_a, \vec{k}_a\}$, est le trièdre direct lié à la vitesse de l'avion dans lequel le vecteur unitaire $\vec{i}_a$ est parallèle à la vitesse aérodynamique de l'avion :

$$\vec{V}_a = V_a\vec{i}_a \tag{26}$$

Dans ce repère la portance L et la trainée D ont les expressions suivantes :

$$\begin{cases} \vec{D} = -D\vec{i}_a \\ \vec{L} = L\vec{j}_a \end{cases} \tag{27}$$

Enfin le trièdre avion, noté $\mathcal{R}_b = \{\vec{i}_b, \vec{j}_b, \vec{k}_b\}$, est le trièdre direct rigidement lié à l'avion dans lequel le vecteur unitaire $\vec{i}_b$ est parallèle à la poussée de l'avion :

$$\vec{F} = F\vec{i}_b \tag{28}$$

Les relations de passage entre ces trois trièdres sont des matrices de rotations.

Les équations de la dynamique du solide sont établies dans le repère aérodynamique. En utilisant les lois de la cinématique et de la dynamique du solide les équations suivantes peuvent être établies :

$$\begin{cases} \dot{y} = V_a\cos(\theta-\alpha) \\ \dot{h} = V_a\sin(\theta-\alpha) \\ \dot{V}_a = \frac{F\cos(\alpha)-D}{m} - g\sin(\theta-\alpha) \\ \dot{\alpha} = q - \frac{L+F\sin(\alpha)}{mV_a} + \frac{g}{V_a}\cos(\theta-\alpha) \\ \dot{\theta} = q \\ \dot{q} = \frac{M}{I_{yy}} \end{cases} \tag{29}$$

I_{yy} est le moment d'inertie de l'avion autour de l'axe de tangage. Nous approximerons très grossièrement I_{yy} par la moitié (pour tenir compte du fait que la masse de l'avion est concentrée au niveau des ailes et des moteurs) du moment d'inertie d'une barre de section négligeable, de masse m et de longueur L_{fus} :

$$I_{yy} \approx \frac{1}{2}\left(\frac{1}{12}mL_{fus}^2\right) \tag{30}$$

Nous définissons le vecteur d'état $\underline{x}$ et le vecteur de commande $\underline{u}$ comme suit :

$$\underline{x} = \begin{bmatrix} y \\ h \\ V_a \\ \alpha \\ \theta \\ q \end{bmatrix} \text{ et } \underline{u} = \begin{bmatrix} \delta_{PHR} \\ \delta_{th} \end{bmatrix} \tag{31}$$

Les équations ci-dessus peuvent se ré-écrire à l'aide du champ de vecteur f :

$$\dot{\underline{x}} = f(\underline{x}, \underline{u}) \tag{32}$$

Équipe	Avion	Moteur	F_0 (N)
Alpha	Airbus A-320	CFM 56-5A1	2 x 111205
Bravo	Boeing 737-800	CFM 56-7B24	2 x 106757
Charlie	Airbus A-319	CFM 56-5B5	2 x 97860
Delta	Airbus A-321	CFM 56-5B1	2 x 133446
Echo	Boeing 737-700	CFM 56-7B20	2 x 91633
Foxtrot	Boeing 737-300	CFM 56-3B1	2 x 88694

TABLE 1 – Table des motorisations

Avion	λ	λ_t	S (m^2)	S_t (m^2)	$\bar{c}$ (m)	L_{fus} (m)
Airbus A-320	9.39	5	122.44	31	4.19	37.57
Boeing 737-800	9.45	6.28	124.6	32.8	4.17	38.02
Airbus A-319	9.39	5	122.44	31	4.19	33.84
Airbus A-321	9.13	5	126	31	4.34	44.51
Boeing 737-700	9.44	6.28	124.6	32.8	4.17	32.18
Boeing 737-300	9.16	5.15	91.04	31.31	3.73	32.18

TABLE 2 – Table des paramètres géométriques

Avion	$MTOW$ (kg)	OWE (kg)
Airbus A-320	73500	39733
Boeing 737-800	70534	41413
Airbus A-319	64000	39358
Airbus A-321	89000	47000
Boeing 737-700	60326	37648
Boeing 737-300	56473	31480

TABLE 3 – Table des masses

2.6 Paramètres

Les données des tables 1, 2 et 3 seront utilisées. Elles proviennent de l'ouvrage *Élodie Roux, Avions civils à réaction : plan 3 vues et données caractéristiques, 2007. ISBN : 978-2-9529380-2-0*. En ce qui concerne la table 3, MTOW (Maximum Takeoff Weight) désigne la masse maximale au décollage et OWE (Operating Empty Weight) la masse à vide en ordre d'exploitation (carburant non inclus !).

La masse m sera choisie entre la masse à vide en ordre d'exploitation et la masse maximale au décollage. Pour cela nous ferons varier le coefficient k_m de réglage de la masse entre 0.1 (on met quand même un peu de kérosène dans les réservoirs) et 1 :

$$m = (1 - k_m)\text{OWE} + k_m\text{MTOW où } 0.1 \leq k_m \leq 1 \tag{33}$$

3 Séance 1 : Analyse des coefficients aérodynamiques

3.1 Travail hors séances encadrées

Réaliser deux diapositives présentant l'avion que vous étudiez (année de mise en service, nombre d'exemplaires construits, compagnies exploitantes, capacités, performances, etc...)

3.2 Séance encadrée

1. Tracer l'évolution de la poussée maximale en fonction du nombre de Mach (compris entre 0.5 et 0.8) lorsque l'altitude h vaut 3000 m puis 11000 m. Comment évolue la poussée maximale avec l'altitude et le nombre de mach ?
2. Tracer le coefficient de portance C_L en fonction de l'incidence α (comprise entre $-10\frac{\pi}{180}\ rad$ à $+20\frac{\pi}{180}\ rad$) lorsque δ_{PHR} vaut $-30\frac{\pi}{180}\ rad$ puis $+20\frac{\pi}{180}\ rad$. Quel est l'effet de δ_{PHR} sur le coefficient de portance ? Le modèle proposé simule t-il le décrochage de l'avion ?
3. Tracer le coefficient C_m du moment de tangage en fonction de α pour quatre valeurs de la marge statique m_s : -0.1, 0, 0.2 et 1 lorsque $\delta_{PHR} = 0$. Quel est la réaction de l'avion en cas d'augmentation intempestive de l'incidence α ?
4. Calculer et tracer en fonction de l'incidence α la valeur δ_{PHRe} de δ_{PHR} pour laquelle le moment de tangage est nul (i.e. $C_m = 0$) et le vol stabilisé (i.e. $q = 0$). Comment varie δ_{PHRe} avec la marge statique m_s et le volume d'empennage V_t ? Comment varie δ_{PHRe} en fonction de l'incidence d'équilibre que l'on notera α_e ?
5. Tracer en fonction de α_e le coefficient de portance équilibrée C_{Le}, c'est à dire le coefficient C_L lorsque $\delta_{PHR} = \delta_{PHRe}$. Tracer C_{Le} pour deux valeurs de la marge statique : $m_s = 0.2$ et $m_s = 1$. Conclusions ?
6. Tracer la polaire équilibrée pour les deux valeurs précédentes de la marge statique. La polaire équilibrée dépend-elle de la marge statique ? Quelle est la valeur de la finesse maximale ? Retrouver par le calcul ce résultat.

4 Séance 2 : Linéarisation de la dynamique

4.1 Travail hors séances encadrées

Réaliser deux diapositives présentant la biographie scientifique de Otto Lilienthal (équipe Alpha), Ludwig Prandtl (équipe Bravo), Octave Chanute (équipe Charlie), Nikolaï Joukovski (équipe Delta), Alberto Santos-Dumont (équipe Echo) et André Borschberg (équipe Foxtrot).

4.2 Séance encadrée

Pour cette séance, vous donnerez successivement à l'altitude h, au nombre de Mach M_a, à la marge statique ms et au coefficient k_m de réglage de la masse de l'avion les deux valeurs suivantes :

$$\begin{cases} h \in \{3000, 11000\}m \\ M_a \in \{0.5, 0.8\} \\ ms \in \{0.2, 1\} \\ k_m \in \{0.1, 0.9\} \end{cases}$$

L'ensemble des points de trim est donc un ensemble constitué de $2^4 = 16$ points.

1. Utiliser la méthode numérique fournie dans le code du modèle de dynamique pour déterminer les valeurs de α, δ_{PHR} et δ_{th} correspondant à un vol en palier (point de trim). Vous pourrez par exemple présenter les différentes valeurs de trim en fonction de l'altitude, paramétrées par les différentes valeurs du nombre de Mach et vous pourrez associer une figure pour chaque valeur du couple {marges statique m_s, coefficient de réglage de la masse k_m}.
2. Choisir arbitrairement un point de trim parmi dans l'ensemble des points de trim proposés. Identifier l'équation de sustentation dans l'équation d'état. En faisant l'hypothèse que l'incidence α est *petite* utiliser cette équation pour calculer la valeur du coefficient de portance C_L en fonction de la vitesse et de la masse. Conclure quant à la technique de réglage de la vitesse de l'avion. Utiliser la valeur de C_L obtenue ainsi que les graphiques tracés lors de la première séance pour déterminer (approximativement) les valeurs de trim α_e et δ_{PHRe}. Comment obtenir la valeur de trim de la manette des gaz δ_{th} à partir de la polaire équilibrée ?
3. Pour le point de trim précédemment étudié vérifier que le calcul numérique conduit aux mêmes résultats que la méthode graphique et simuler la trajectoire de l'avion sur $100s$.
4. Linéariser numériquement le modèle d'état pour toutes les conditions de trim. Extraire des représentations d'état linéarisées $\{\mathbf{A}, \mathbf{B}\}$ de dimension 6 (qui est la dimension du vecteur d'état $\begin{bmatrix} y & h & V_a & \alpha & \theta & q \end{bmatrix}^T$) les matrices $\mathbf{A}_4$ et $\mathbf{B}_4$ associées aux composantes $\begin{bmatrix} V_a & \alpha & \theta & q \end{bmatrix}^T$ du vecteur d'état et calculer numériquement les valeurs propres de la matrice $\mathbf{A}_4$. Vous tracerez les valeurs propres pour toutes les conditions de trim en faisant varier successivement l'altitude h, le nombre de Mach M_a, la marge statique ms et la masse m (au travers les différentes valeurs du coefficient k_m de réglage de la masse). Comment varient les valeurs propres en fonction de ces différents paramètres ?

5 Séance 3 : Réponse longitudinale

5.1 Travail hors séances encadrées

Réaliser deux diapositives présentant la biographie scientifique de James Clerk Maxwell (équipe Alpha), Adolf Hurwitz (équipe Bravo), Edward Routh (équipe Charlie), Alexandre Liapounov (équipe Delta), Rudolf Kalman (équipe Echo) et Henri Poincaré (équipe Foxtrot).

5.2 Séance encadrée

Vous choisirez un point de trim pour laquelle toutes les valeurs propres de la matrice $\mathbf{A}_4$ sont à partie réelle strictement négative.

Nous supposerons que l'avion rencontre à l'instant initial un cisaillement de vent. Ce cisaillement de vent d'expression $W_h\delta(t)$ où $\delta(t)$ est l'impulsion de Dirac a pour vitesse verticale $W_h = 2\ m/s$. En termes de simulation cela se traduit par un changement de l'incidence initiale : en notant α_e l'incidence de trim et V_{ae} la vitesse de trim, la nouvelle incidence initiale vaut maintenant $\alpha_e + \arctan\left(\frac{W_h}{V_{ae}}\right)$

1. Simuler et tracer la vitesse V_a, l'incidence α, l'assiette θ et la vitesse de tangage q de l'avion sur $240s$ en utilisant le modèle non linéaire puis le modèle linéarisé. Le modèle linéarisé reflète t-il correctement le comportement du modèle non linéaire ? Quelle est la relation qui relie la période d'oscillation du modèle linéarisé et les valeurs propres de la matrice $\mathbf{A}_4$? Quel est le nom donné à ces oscillations à *grande* échelle de temps ?

2. Réaliser les mêmes simulations sur $10s$. Le modèle linéarisé reflète t-il correctement le comportement du modèle non linéaire ? Quel est le nom donné à ces perturbations à *petite* échelle de temps ?
3. Comparer les trajectoires obtenues sur $240s$ en utilisant le même modèle linéaire mais en choisissant un autre point de trim et en simulant le modèle non linéaire. Quelle conclusion tirez vous quant à la précision du modèle linéaire autour d'un point de trim différent de celui autour duquel il a été obtenu ?

   Par la suite, vous utiliserez la représentation d'état linéarisée $\{\mathbf{A}_4, \mathbf{B}_4\}$ associée au vecteur d'état de composantes $\left[\begin{array}{cccc} V_a & \alpha & \theta & q \end{array}\right]^T$
4. Donner la représentation d'état en base modale. Étudier la stabilité et la commandabilité du modèle linéarisé
5. Calculer de la fonction de transfert $F(p) = \frac{\theta(p)}{\delta_{PHR}(p)}$. Réduire la fonction de transfert $F(p)$ en utilisant l'approximation de Padé ; vous noterez $F_r(p)$ la fonction de transfert réduite. Comparer les réponses indicielles de $F(p)$ et $F_r(p)$
6. Proposer des solutions pour améliorer les qualités de vol de l'avion naturel quelles que soient les conditions de trim.

6 Annexes

6.1 Annexe 1 : dynamic.py

```python
#-*- coding: utf-8 -*-
'''
Dynamic model for a 3 Degrees Of Freedom longitudinal aircraft
'''
import math, numpy as np, scipy.optimize
import matplotlib.pyplot as plt

import utils as ut

''' naming of state and input components '''
s_y, s_h, s_va, s_a, s_th, s_q, s_size = range(0, 7)
i_dm, i_dth, i_wy, i_wz, i_size =  range(0, 5)

def get_mach(va, T, k=1.4, Rs=287.05): return va/math.sqrt(k*Rs*T)

def va_of_mach(m, h, k=1.4, Rs=287.05):
    p, rho, T = ut.isa(h)
    return m*math.sqrt(k*Rs*T)

def propulsion_model(X, U, P):
    p, rho, T = ut.isa(X[s_h])
    rho0 = 1.225
    mach = get_mach(X[s_va], T)
    return P.F0*math.pow(rho/rho0, 0.6)*(0.568+0.25*math.pow(1.2-mach, 3))*U[i_dth]

def get_aero_ceofs(h, va, alpha, q, dphr, P):
    St_over_S = P.St/P.S
    CL0 = (St_over_S*0.25*P.CLat - P.CLa)*P.a0
    CLa = P.CLa + St_over_S*P.CLat*(1-0.25)
    CLq = P.lt * St_over_S * P.CLat * P.CLq
    CLdphr = St_over_S*P.CLat
    CL = CL0 + CLa*alpha + CLq*q/va + CLdphr * dphr
    CD = P.CD0 + P.ki*CL**2
    Cm = P.Cm0 - P.ms*P.CLa + P.Cmq*P.lt/va*q + P.Cmd*dphr
    return CL, CD, Cm

def get_aero_forces_and_moments(X, U, P):
    p, rho, T = ut.isa(X[s_h])
    pdyn = 0.5*rho*X[s_va]**2
```

```
    CL, CD, Cm = get_aero_ceofs(X[s_h], X[s_va], X[s_a], X[s_q], U[i_dm], P)
    L, D, M = pdyn*P.S*np.array([CL, CD, P.cbar*Cm])
    return L, D, M

def dyn(X, t, U, P):
    '''  Dynamic model '''
    Xdot = np.zeros(s_size)
    gamma_a = X[s_th] - X[s_a]   # air path angle
    cg, sg = math.cos(gamma_a), math.sin(gamma_a)
    ca, sa = math.cos(X[s_a]), math.sin(X[s_a])
    L, D, M = get_aero_forces_and_moments(X, U, P)
    F = propulsion_model(X, U, P)
    Xdot[s_y] = X[s_va] * cg - U[i_wy]
    Xdot[s_h] = X[s_va] * sg - U[i_wz]
    Xdot[s_va] = (F*ca-D)/P.m-P.g*sg
    Xdot[s_a] = X[s_q] - (L+F*sa)/P.m/X[s_va]  + P.g/X[s_va]*cg
    Xdot[s_th] = X[s_q]
    Xdot[s_q] = M/P.Iyy
    return Xdot

def trim(P, args=None):
    va = args.get('va', 100.)
    gamma = args.get('gamma', 0.)
    h = args.get('h', 5000.)
    wy, wz = 0, 0
    def err_func(arg):
        (dm, dth, alpha) = arg
        theta = gamma + alpha
        U = np.array([dm, dth, wy, wz])
        X = np.array([0., h, va, alpha, theta, 0])
        L, D, M = get_aero_forces_and_moments(X, U, P)
        F = propulsion_model(X, U, P)
        cg, sg = math.cos(gamma), math.sin(gamma)
        ca, sa = math.cos(alpha), math.sin(alpha)
        return [(F*ca-D)/P.m-P.g*sg, -(L+F*sa)/P.m + P.g*cg, M]
    p0 = [ut.rad_of_deg(0.), 0.5, ut.rad_of_deg(1.)]
    sol = scipy.optimize.root(err_func, p0, method='hybr')
    dm, dth, alpha = sol.x
    X, U = [0, h, va, alpha, gamma+alpha, 0], [dm, dth, 0, 0]
    return X, U


class Param:
    def __init__(self):
        self.g = 9.81
        self.m_k = 0.5
        self.ms = 0.3                    # static margin
        # aero
        self.a0 = ut.rad_of_deg(-2.)     # zero lift angle
        self.CD0 = 0.025                 # zero drag coefficient
        self.Cm0 = -0.59                 # zero moment coefficient
        self.CLq = 1.3

    def set_mass_and_static_margin(self, km, sm):
        self.m_k = km
        self.ms = sm
        self.compute_auxiliary()

    def compute_auxiliary(self):
        self.m = (1-self.m_k)*self.m_OWE + self.m_k*self.m_MTOW
        self.Iyy = 0.5*(1./12.*self.m*self.l_fus**2)

        self.lt = 0.5 * self.l_fus            # CG to tail distance

        self.CLa = math.pi*self._lambda/(1.+math.sqrt(1+(0.5*self._lambda)**2))
        self.ki = 1./(math.pi*self._lambda)

        self.CLat = math.pi*self._lambdat/(1.+math.sqrt(1+(0.5*self._lambdat)**2))

        Vt = self.lt*self.St/self.cbar/self.S # tail volume
        self.Cmd = -Vt*self.CLat
        self.Cmq = self.Cmd * self.CLq

```

```
class Param_A320(Param):
    def __init__(self, m_k=0.5):
        Param.__init__(self)
        self.name = 'Airbus A-320'
        self.m_OWE  = 39733.
        self.m_MTOW = 73500.
        self.m_k = m_k

        self.l_fus     =  37.57         # length of fuselage
        self.cbar      =   4.19         # wing reference chord
        self.St        =  31.           # tail lifting surface
        self.S         = 122.44         # wing surface
        self._lambdat =   5.            # tail aspect ratio
        self._lambda  =   9.39          # wing aspect ratio

        self.F0        = 2.*111205      # engine max thrust
        self.eng_name = 'CFM 56-5A1'

        self.compute_auxiliary()


class Param_737_800(Param):
    def __init__(self, ):
        Param.__init__(self)
        self.name = 'Boeing 737-800'
        self.m_OWE  = 41413.
        self.m_MTOW = 70534.

        self.l_fus     =  38.02         # length of fuselage
        self.cbar      =   4.17         # wing reference chord
        self.St        =  32.8          # tail lifting surface
        self.S         = 124.6          # wing surface
        self._lambdat =   6.28          # tail aspect ratio
        self._lambda  =   9.45          # wing aspect ratio

        self.F0        = 2.*106757      # engine max thrust
        self.eng_name = 'CFM 56-7B24'

        self.compute_auxiliary()


class Param_A319(Param):
    def __init__(self):
        Param.__init__(self)
        self.name = 'Airbus A-319'
        self.m_OWE  = 39358.
        self.m_MTOW = 64000.

        self.l_fus     =  33.84         # length of fuselage
        self.cbar      =   4.19         # wing reference chord
        self.St        =  31.0          # tail lifting surface
        self.S         = 122.44         # wing surface
        self._lambdat =   5.            # tail aspect ratio
        self._lambda  =   9.39          # wing aspect ratio

        self.F0        = 2.*97860       # engine max thrust
        self.eng_name = 'CFM 56-5B5'

        self.compute_auxiliary()


class Param_A321(Param):
    def __init__(self):
        Param.__init__(self)
        self.name = 'Airbus A-321'
        self.m_OWE  = 47000.
        self.m_MTOW = 89000.

        self.l_fus     =  44.51         # length of fuselage
        self.cbar      =   4.34         # wing reference chord
        self.St        =  31.0          # tail lifting surface
        self.S         = 126.00         # wing surface
```

```
        self._lambdat =   5.            # tail aspect ratio
        self._lambda  =   9.13          # wing aspect ratio

        self.F0       = 2.*133446       # engine max thrust
        self.eng_name = 'CFM_56-5B1'

        self.compute_auxiliary()


class Param_737_700(Param):
    def __init__(self):
        Param.__init__(self)
        self.name = 'Boeing_737-700'
        self.m_OWE  = 37648.
        self.m_MTOW = 60326.

        self.l_fus     =  32.18         # length of fuselage
        self.cbar      =   4.17         # wing reference chord
        self.St        =  32.8          # tail lifting surface
        self.S         = 124.60         # wing surface
        self._lambdat  =   6.28         # tail aspect ratio
        self._lambda   =   9.44         # wing aspect ratio

        self.F0       = 2.*61633        # engine max thrust
        self.eng_name = 'CFM_56-7B20'

        self.compute_auxiliary()


class Param_737_300(Param):
    def __init__(self):
        Param.__init__(self)
        self.name = 'Boeing_737-300'
        self.m_OWE  = 31480.
        self.m_MTOW = 56473.

        self.l_fus     =  32.18         # length of fuselage
        self.cbar      =   3.73         # wing reference chord
        self.St        =  31.31         # tail lifting surface
        self.S         =  91.04         # wing surface
        self._lambdat  =   5.15         # tail aspect ratio
        self._lambda   =   9.16         # wing aspect ratio

        self.F0       = 2.*88694        # engine max thrust
        self.eng_name = 'CFM_56-3B1'

        self.compute_auxiliary()

all_ac_types = [Param_A320, Param_737_800, Param_A319, Param_A321, Param_737_700, Param_737_300]

def plot(time, X, U=None, figure=None, window_title="Trajectory"):
    figure = ut.prepare_fig(figure, window_title, (20.48, 10.24))
    plots = [("$y$",         "m",     X[:,s_y], None),
             ("$h$",         "m",     X[:,s_h], 2.),
             ("$v_a$",       "m/s",   X[:,s_va], 1.),
             ("$\\alpha$", "deg",   ut.deg_of_rad(X[:,s_a]), 2.),
             ("$\\theta$", "deg",   ut.deg_of_rad(X[:,s_th]), 2.),
             ("$q$",         "deg/s", ut.deg_of_rad(X[:,s_q]), 2.)]
    for i, (title, ylab, data, min_yspan) in enumerate(plots):
        ax = plt.subplot(3, 2, i+1)
        plt.plot(time, data)
        ut.decorate(ax, title=title, ylab=ylab, min_yspan=min_yspan)
    return figure
```

6.2 Annexe 1 : utils.py

```
#-*- coding: utf-8 -*-
'''

Utilities
```

```
'''
import math, numpy as np
import matplotlib.pyplot as plt
import pdb

"""
Unit convertions
"""
def rad_of_deg(d): return d/180.*math.pi
def deg_of_rad(r): return r*180./math.pi
# http://en.wikipedia.org/wiki/Nautical_mile
def m_of_NM(nm): return nm*1852.
def NM_of_m(m): return m/1852.
# http://en.wikipedia.org/wiki/Knot_(speed)
def mps_of_kt(kt): return kt*0.514444
def kt_of_mps(mps): return mps/0.514444


"""
International Standard Atmosphere Model
see: http://en.wikipedia.org/wiki/International_Standard_Atmosphere
"""
_name, _h0, _z0, _a, _T0, _p0 = np.arange(6)
#      name              h(m)     z(km)     a(K/m)     T0(K)     p0(Pa)
isa_param = \
[[ 'Troposphere',          0,      0.0,    -6.5e-3,    288.15,    101325],
 [ 'Tropopause',       11000,   11.019,     0.0e-3,    216.65,     22632],
 [ 'Stratosphere',     20000,   20.063,     1.0e-3,    216.65,     5474.9],
 [ 'Stratosphere',     32000,   32.162,     2.8e-3,    228.65,     868.02],
 [ 'Stratopause',      47000,   47.350,     0.0e-3,    270.65,     110.91],
 [ 'Mesosphere',       51000,   51.413,    -2.8e-3,    270.65,     66.939],
 [ 'Mesosphere',       71000,   71.802,    -2.0e-3,    214.65,     3.9564],
 [ 'Mesopause',        84852,   86.000,     0.,        186.87,     0.3734]]

def isa(h):
    layer = 0
    while isa_param[layer][_h0] < h: layer+=1
    if layer == 0: layer = 1 # in case h<= 0
    name, h0, z0, a, T0, p0 = isa_param[layer-1]
    dh = h - h0
    T = T0 + a*dh
    g, R = 9.81, 287.05
    if a != 0.:
        p = p0*math.pow(T/T0, -g/a/R)
    else:
        p = p0*math.exp(-g/R/T0*dh)
    rho = p/R/T
    return p, rho, T


"""
Compute numerical jacobian
"""
def num_jacobian(X, U, P, dyn):
    s_size = len(X)
    i_size = len(U)
    epsilonX = (0.1*np.ones(s_size)).tolist()
    dX = np.diag(epsilonX)
    A = np.zeros((s_size, s_size))
    for i in range(0, s_size):
        dx = dX[i,:]
        delta_f = dyn(X+dx/2, 0, U, P) - dyn(X-dx/2, 0, U, P)
        delta_f = delta_f / dx[i]
        A[:,i] = delta_f

    epsilonU = (0.1*np.ones(i_size)).tolist()
    dU = np.diag(epsilonU)
    B = np.zeros((s_size,i_size))
    for i in range(0, i_size):
        du = dU[i,:]
        delta_f = dyn(X, 0, U+du/2, P) - dyn(X, 0, U-du/2, P)
        delta_f = delta_f / du[i]
```

```
        B[:,i] = delta_f

    return A,B


"""
Plotting
"""
def decorate(ax, title=None, xlab=None, ylab=None, legend=None, xlim=None, ylim=None, min_yspan=None):
    ax.xaxis.grid(color='k', linestyle='-', linewidth=0.2)
    ax.yaxis.grid(color='k', linestyle='-', linewidth=0.2)
    if xlab: ax.xaxis.set_label_text(xlab)
    if ylab: ax.yaxis.set_label_text(ylab)
    if title: ax.set_title(title, {'fontsize': 20 })
    if legend != None: ax.legend(legend, loc='best')
    if xlim != None: ax.set_xlim(xlim[0], xlim[1])
    if ylim != None: ax.set_ylim(ylim[0], ylim[1])
    if min_yspan != None: ensure_yspan(ax, min_yspan)

def ensure_yspan(ax, yspan):
    ymin, ymax = ax.get_ylim()
    if ymax-ymin < yspan:
        ym = (ymin+ymax)/2
        ax.set_ylim(ym-yspan/2, ym+yspan/2)

def prepare_fig(fig=None, window_title=None, figsize=(20.48, 10.24), margins=None):
    if fig == None:
        fig = plt.figure(figsize=figsize)
    else:
        plt.figure(fig.number)
    if margins:
        left, bottom, right, top, wspace, hspace = margins
        fig.subplots_adjust(left=left, right=right, bottom=bottom, top=top,
                            hspace=hspace, wspace=wspace)
    if window_title:
        fig.canvas.set_window_title(window_title)
    return fig
```