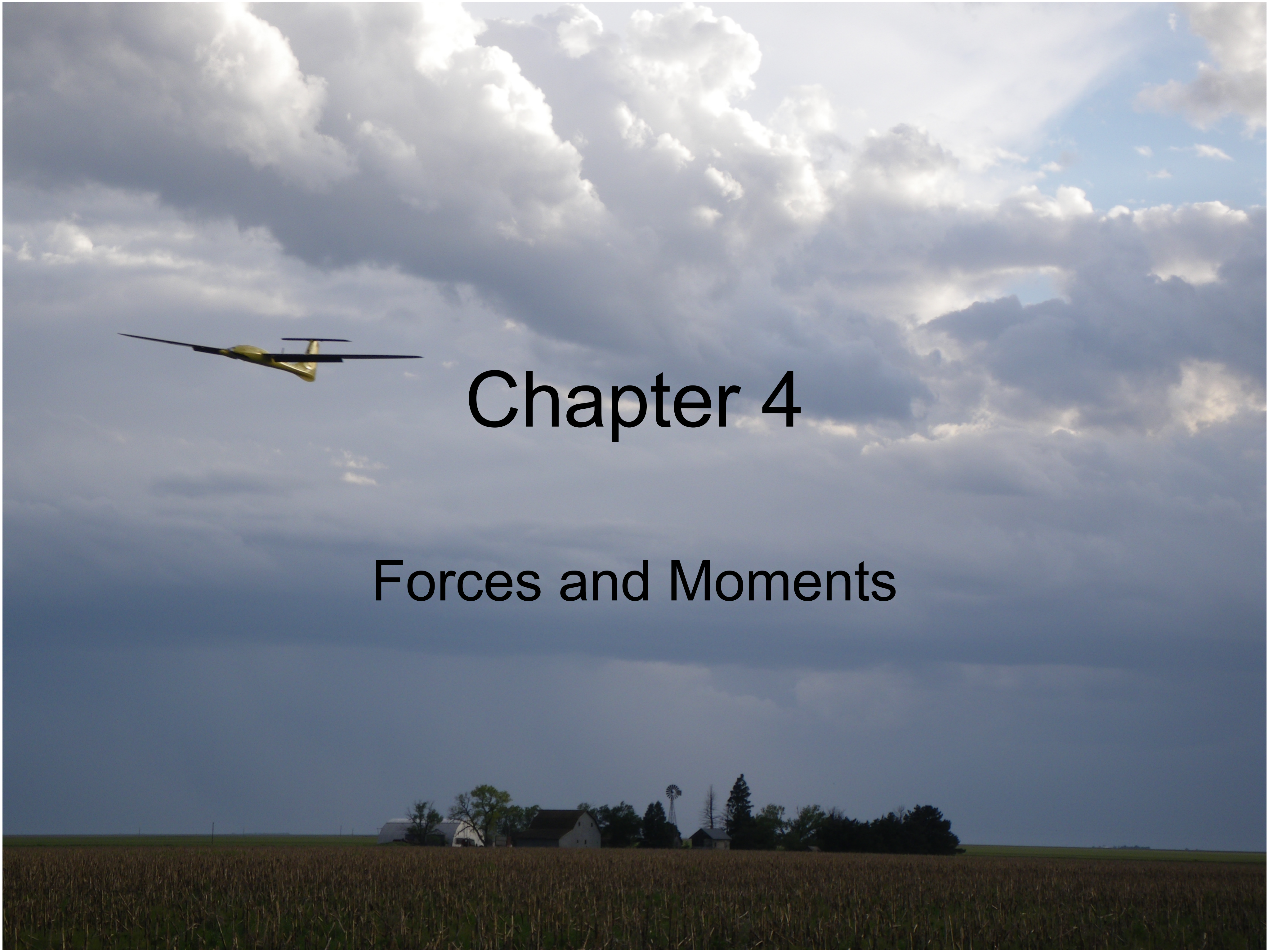


Chapter 4

Forces and Moments

Equations of Motion from Chap 3

The combined equations of motion from Chapter 3 are

$$
\begin{pmatrix} \dot{p}_n \\ \dot{p}_e \\ \dot{p}_d \end{pmatrix} = \begin{pmatrix} c_\theta c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi & c_\phi s_\theta c_\psi + s_\phi s_\psi \\ c_\theta s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{pmatrix} \begin{pmatrix} u \\ v \\ w \end{pmatrix}
$$

$$
\begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{pmatrix} = \begin{pmatrix} rv - qw \\ pw - ru \\ qu - pv \end{pmatrix} + \frac{1}{\mathsf{m}} \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix},
$$

$$
\begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{pmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{pmatrix} \begin{pmatrix} p \\ q \\ r \end{pmatrix}
$$

$$
\begin{pmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{pmatrix} = \begin{pmatrix} \Gamma_1 pq - \Gamma_2 qr \\ \Gamma_5 pr - \Gamma_6 (p^2 - r^2) \\ \Gamma_7 pq - \Gamma_1 qr \end{pmatrix} + \begin{pmatrix} \Gamma_3 l + \Gamma_4 n \\ \frac{1}{J_y} m \\ \Gamma_4 l + \Gamma_8 n \end{pmatrix}
$$

The objective of this chapter is to show how to compute the force vector

$$
\mathbf{f}^b = \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix}
$$

and the moment vector

$$
\mathbf{m}^b = \begin{pmatrix} \ell \\ m \\ n \end{pmatrix}.
$$

External Forces and Moments

The external forces are a combination of gravitational, aerodynamic, and propulsion:

$$\mathbf{f} = \mathbf{f}_g + \mathbf{f}_a + \mathbf{f}_p.$$

The external moments are a combination of aerodynamic, and propulsion:

$$\mathbf{m} = \mathbf{m}_a + \mathbf{m}_p.$$

Gravity Force

The gravity vector expressed in the vehicle frame is

$$\mathbf{f}_g^v = \begin{pmatrix} 0 \\ 0 \\ \mathsf{m}g \end{pmatrix}.$$

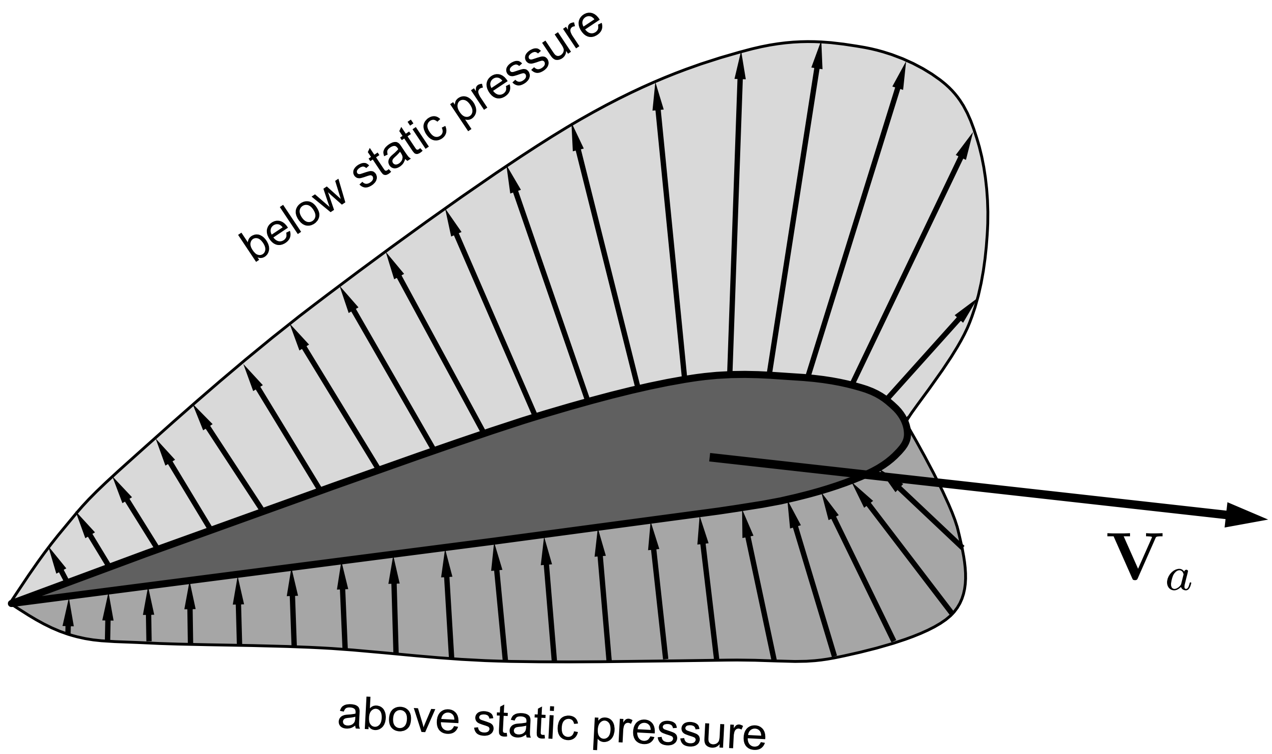
Expressed in the body frame we have

$$\mathbf{f}_g^b = \mathcal{R}_v^b \begin{pmatrix} 0 \\ 0 \\ \mathsf{m}g \end{pmatrix} = \begin{pmatrix} c_\theta c_\psi & c_\theta s_\psi & -s_\theta \\ s_\phi s_\theta c_\psi - c_\phi s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & s_\phi c_\theta \\ c_\phi s_\theta c_\psi + s_\phi s_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi & c_\phi c_\theta \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ \mathsf{m}g \end{pmatrix} = \mathsf{m}g \begin{pmatrix} -\sin\theta \\ \cos\theta\sin\phi \\ \cos\theta\cos\phi \end{pmatrix}$$

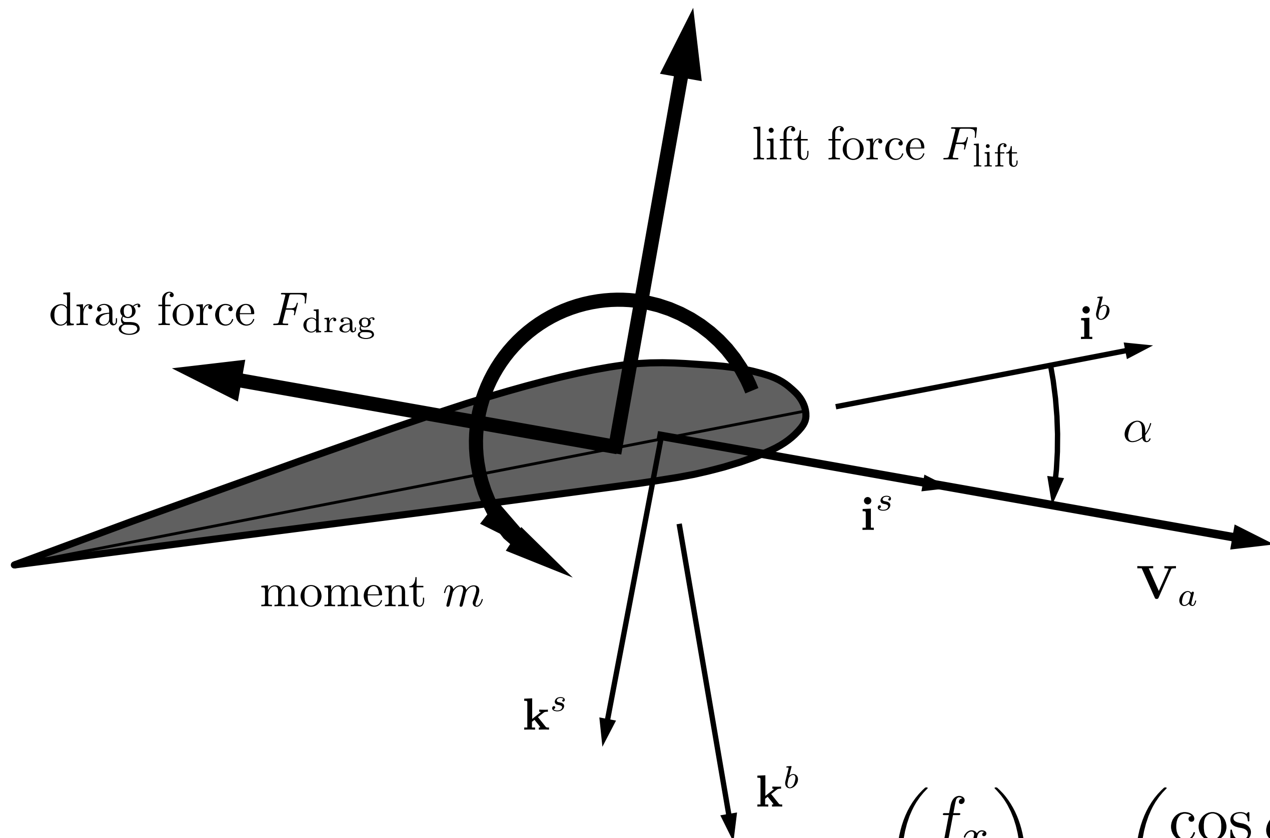
For quaternions, we have

$$\mathbf{f}_g^b = \begin{pmatrix} e_0^2 + e_x^2 - e_y^2 - e_z^2 & 2(e_x e_y - e_0 e_z) & 2(e_x e_z + e_0 e_y) \\ 2(e_x e_y + e_0 e_z) & e_0^2 - e_x^2 + e_y^2 - e_z^2 & 2(e_y e_z - e_0 e_x) \\ 2(e_x e_z - e_0 e_y) & 2(e_y e_z + e_0 e_x) & e_0^2 - e_x^2 - e_y^2 + e_z^2 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ \mathsf{m}g \end{pmatrix} = \mathsf{m}g \begin{pmatrix} 2(e_x e_z - e_y e_0) \\ 2(e_y e_z + e_x e_0) \\ e_z^2 + e_0^2 - e_x^2 - e_y^2 \end{pmatrix}.$$

Airfoil Pressure Distribution



Aerodynamic Approximation



$$F_{\text{lift}} = \frac{1}{2}\rho V_a^2 S C_L$$

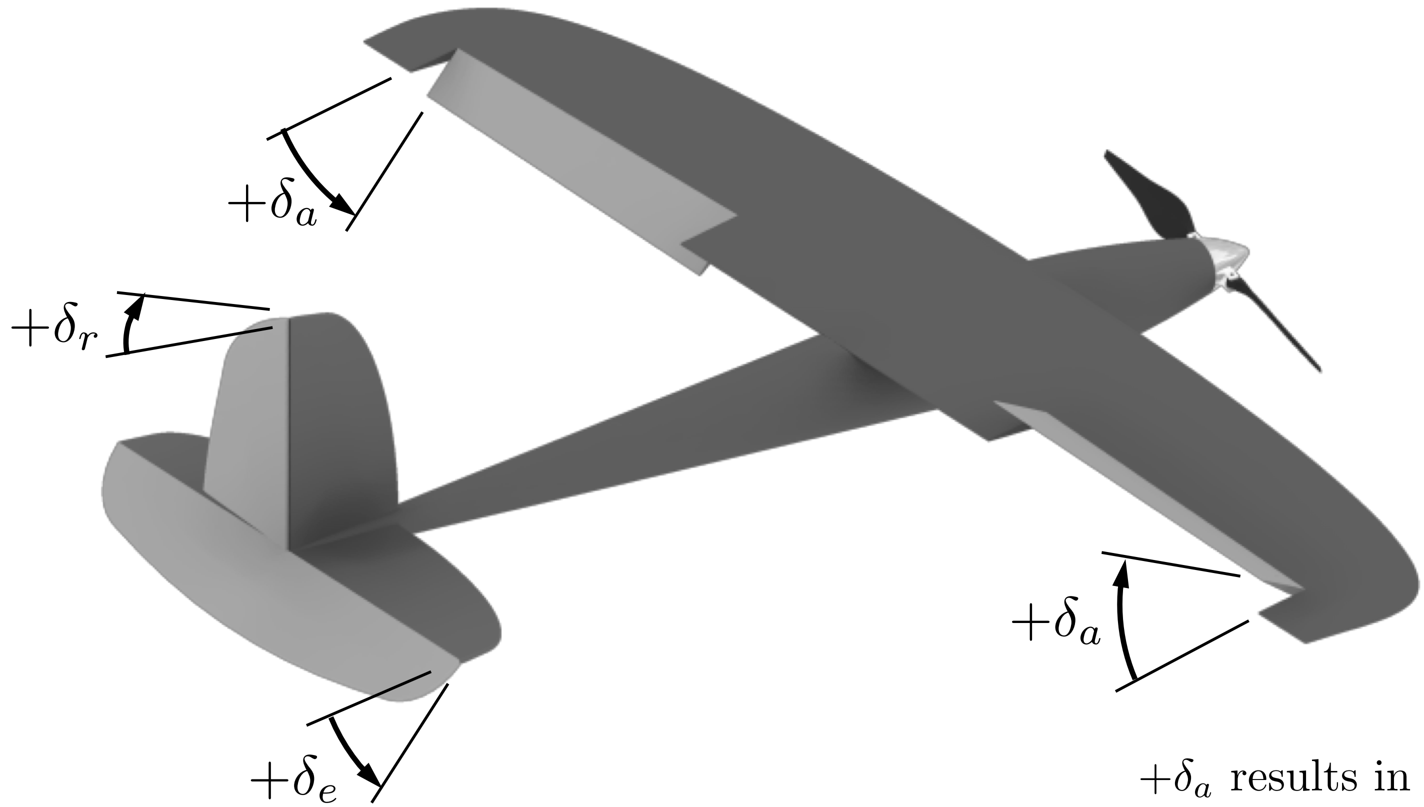
$$F_{\text{drag}} = \frac{1}{2}\rho V_a^2 S C_D$$

$$m = \frac{1}{2}\rho V_a^2 S c C_m$$

$$\begin{pmatrix} f_x \\ f_z \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} -F_{\text{drag}} \\ -F_{\text{lift}} \end{pmatrix}$$

Control Surfaces - Conventional

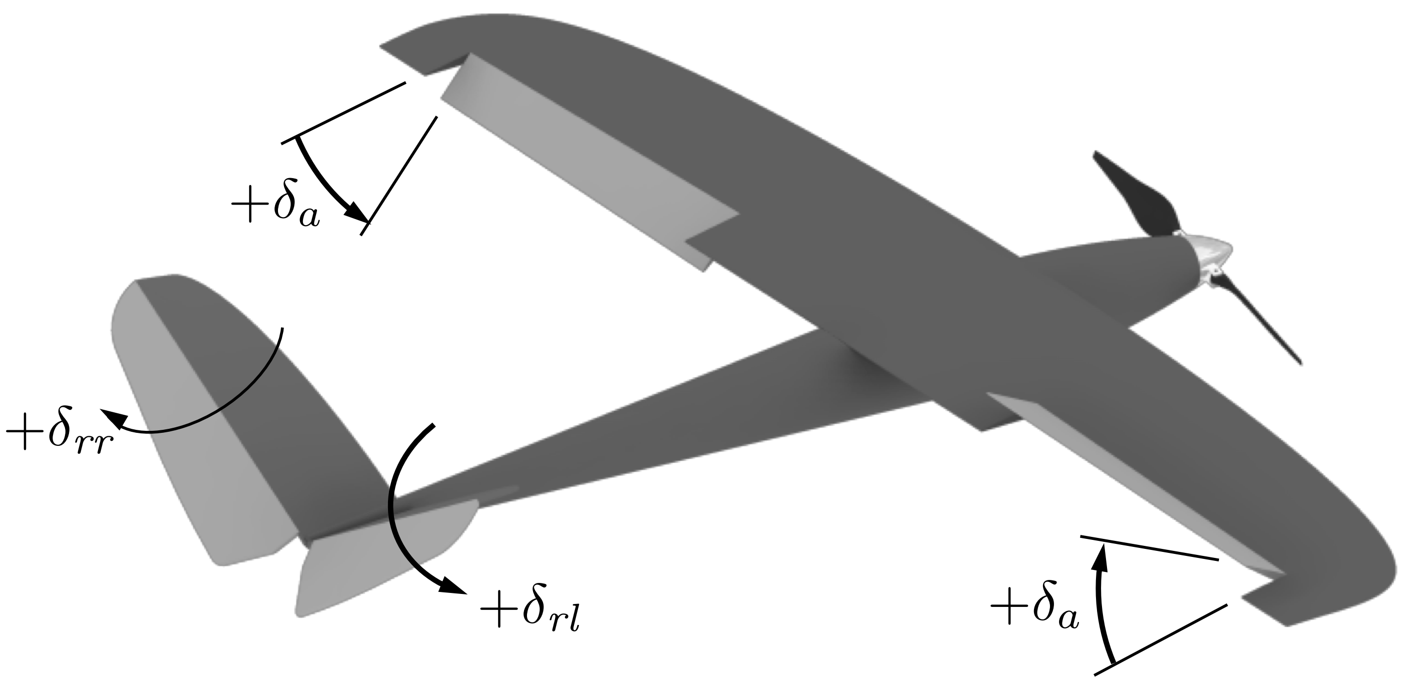
$$\delta_a = \frac{1}{2}\left(\delta_{a\text{-left}} - \delta_{a\text{-right}}\right)$$



$+\delta_a$ results in positive roll rate p.
$+\delta_e$ results in negative pitch rate q.
$+\delta_r$ results in negative yaw rate r

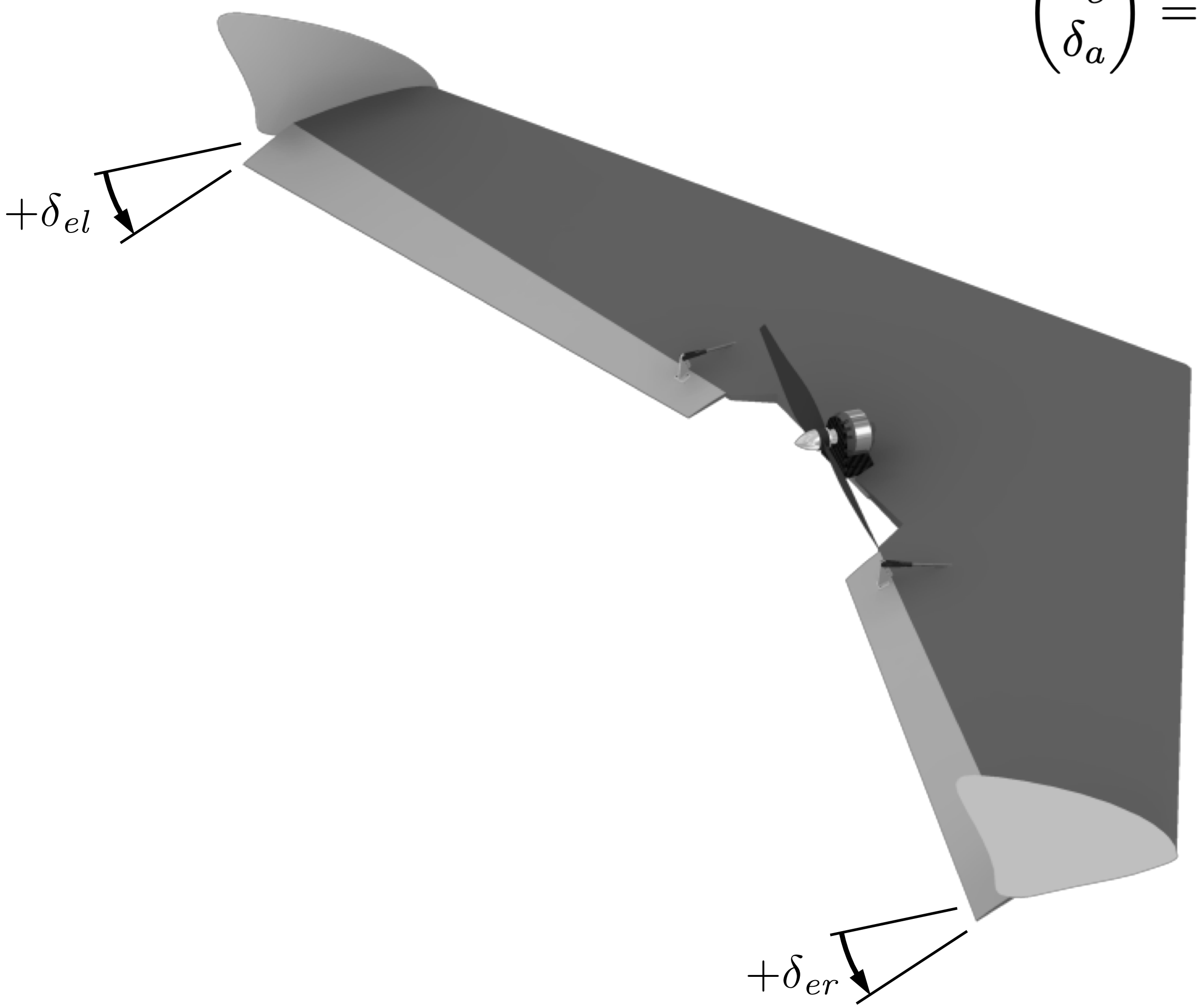
Control Surfaces – V-tail

$$\begin{pmatrix} \delta_e \\ \delta_r \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} \delta_{rr} \\ \delta_{rl} \end{pmatrix}$$



Control Surfaces – Flying Wing

$$\begin{pmatrix}\delta_e \\ \delta_a\end{pmatrix} = \frac{1}{2}\begin{pmatrix}1 & 1 \\ -1 & 1\end{pmatrix}\begin{pmatrix}\delta_{er} \\ \delta_{el}\end{pmatrix}$$



Aircraft Dynamics

- Aircraft dynamics and aerodynamics are commonly separated into two groups:
  - Longitudinal
    - Up-down, pitch plane, pitching motions
  - Lateral-directional
    - Side-to-side, turning motions (roll and yaw)

Longitudinal Aerodynamics

- Act in the $\mathbf{i}^b - \mathbf{k}^b$ plane, aka the pitch plane
- Heavily influenced by angle of attack
- Also influenced by pitch rate and elevator deflection

$$
\begin{aligned}
F_{\text{lift}} &\approx \frac{1}{2}\rho V_a^2 S C_L(\alpha, q, \delta_e) \\
F_{\text{drag}} &\approx \frac{1}{2}\rho V_a^2 S C_D(\alpha, q, \delta_e) \\
m &\approx \frac{1}{2}\rho V_a^2 S c C_m(\alpha, q, \delta_e)
\end{aligned}
$$

Aerodynamic Approximation

In the general nonlinear case we have

$$F_{\text{lift}} \approx \frac{1}{2}\rho V_a^2 S C_L(\alpha, q, \delta_e).$$

Expanding C_L as a Taylor series and keeping only the first order (linear) terms gives

$$\begin{aligned} F_{\text{lift}} &= \frac{1}{2}\rho V_a^2 S \left[C_{L_0} + \frac{\partial C_L}{\partial \alpha}\alpha + \frac{\partial C_L}{\partial q} q + \frac{\partial C_L}{\partial \delta_e}\delta_e \right] \\ &= \frac{1}{2}\rho V_a^2 S \left[C_{L_0} + C_{L_\alpha}\alpha + C_{L_q}\frac{c}{2V_a} q + C_{L_{\delta_e}}\delta_e \right] \end{aligned}$$

where the coefficients C_{L_0}, $C_{L_\alpha} \triangleq \frac{\partial C_L}{\partial \alpha}$, $C_{L_q} \triangleq \frac{\partial C_L}{\partial \frac{qc}{2V_a}}$, and $C_{L_{\delta_e}} \triangleq \frac{\partial C_L}{\partial \delta_e}$ are dimensionless quantities.

stability derivatives

control derivative

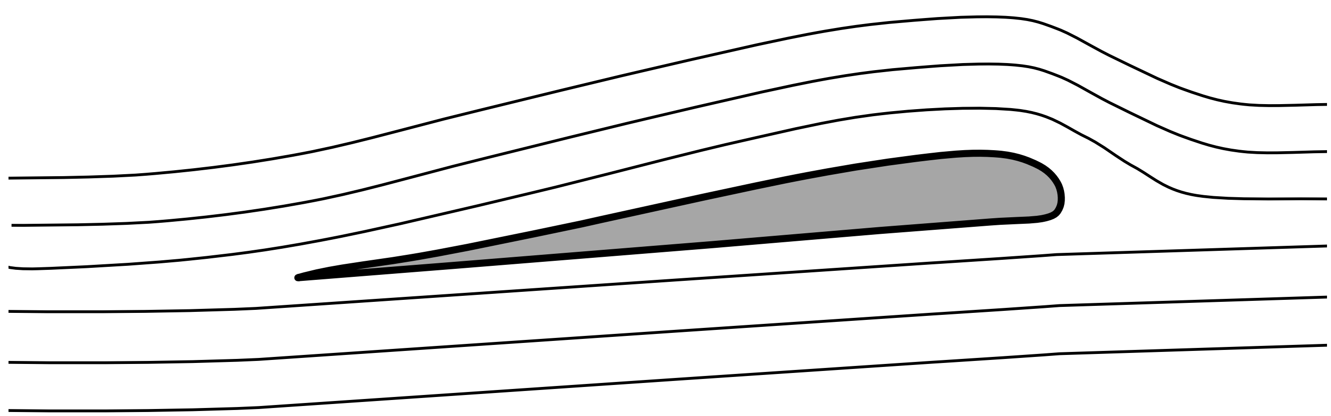
Linear Aerodynamic Model

$$F_{\text{lift}} = \frac{1}{2}\rho V_a^2 S\left[C_{L_0} + C_{L_\alpha}\alpha + C_{L_q}\frac{c}{2V_a}q + C_{L_{\delta_e}}\delta_e\right]$$

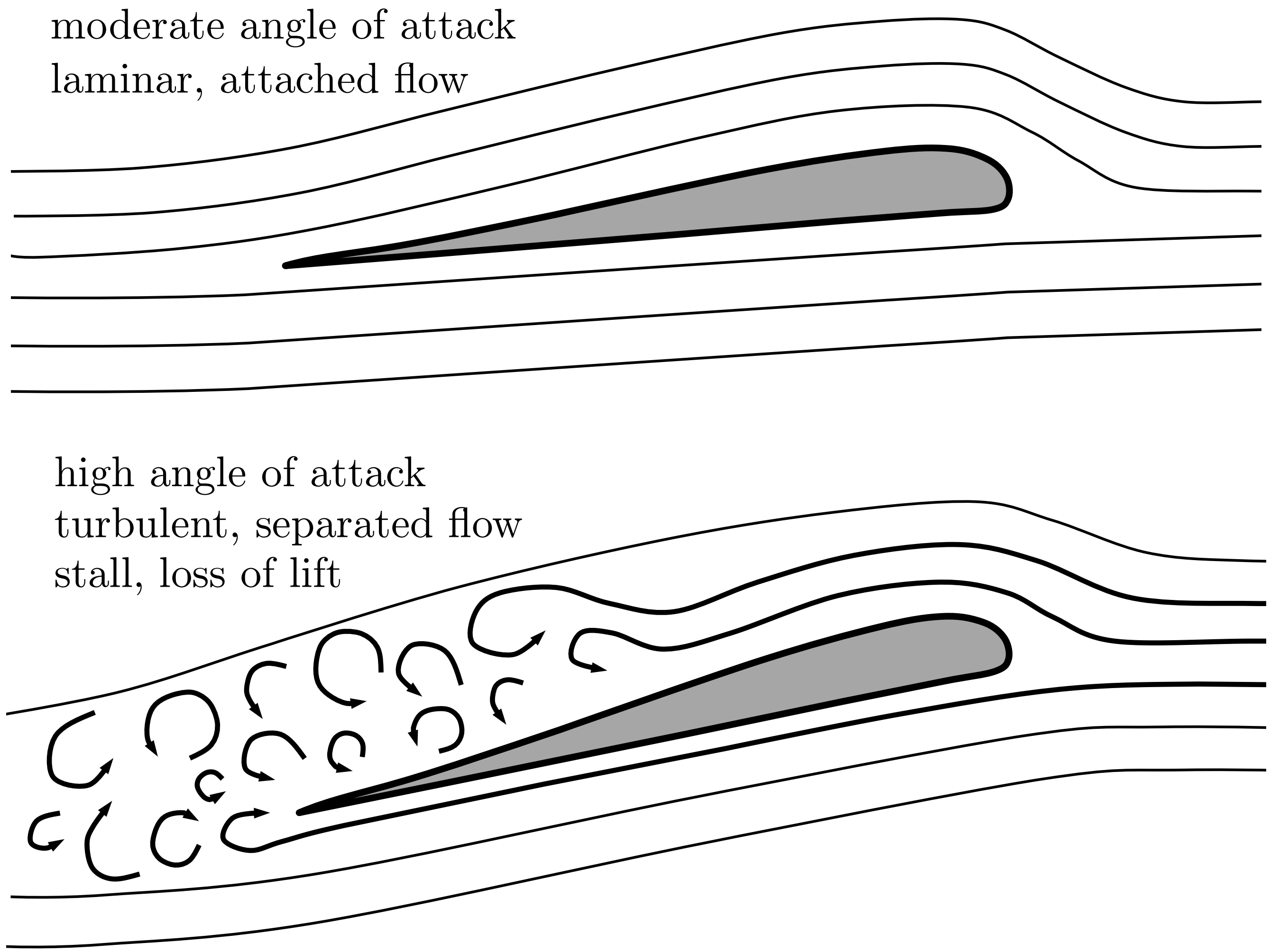
$$F_{\text{drag}} = \frac{1}{2}\rho V_a^2 S\left[C_{D_0} + C_{D_\alpha}\alpha + C_{D_q}\frac{c}{2V_a}q + C_{D_{\delta_e}}\delta_e\right]$$

$$m = \frac{1}{2}\rho V_a^2 Sc\left[C_{m_0} + C_{m_\alpha}\alpha + C_{m_q}\frac{c}{2V_a}q + C_{m_{\delta_e}}\delta_e\right]$$

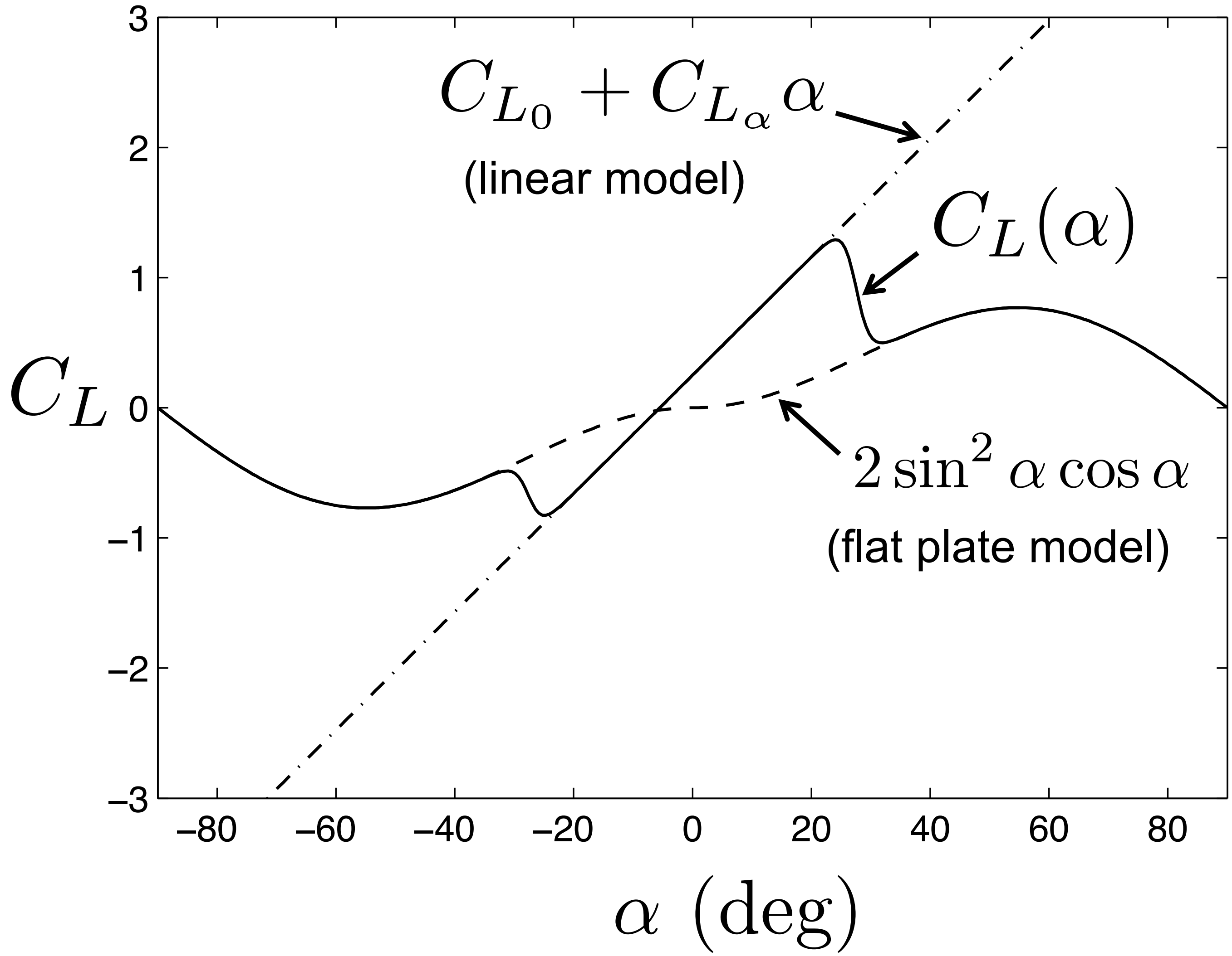
Linear aerodynamic model is valid for small angles of attack – flow remains attached over wing



Nonlinear Aerodynamics – Stall



Nonlinear Lift Model



Nonlinear Aerodynamic Model

$$F_{\text{lift}} = \frac{1}{2}\rho V_a^2 S\left[\underline{C_L(\alpha)} + C_{L_q}\frac{c}{2V_a}q + C_{L_{\delta_e}}\delta_e\right]$$

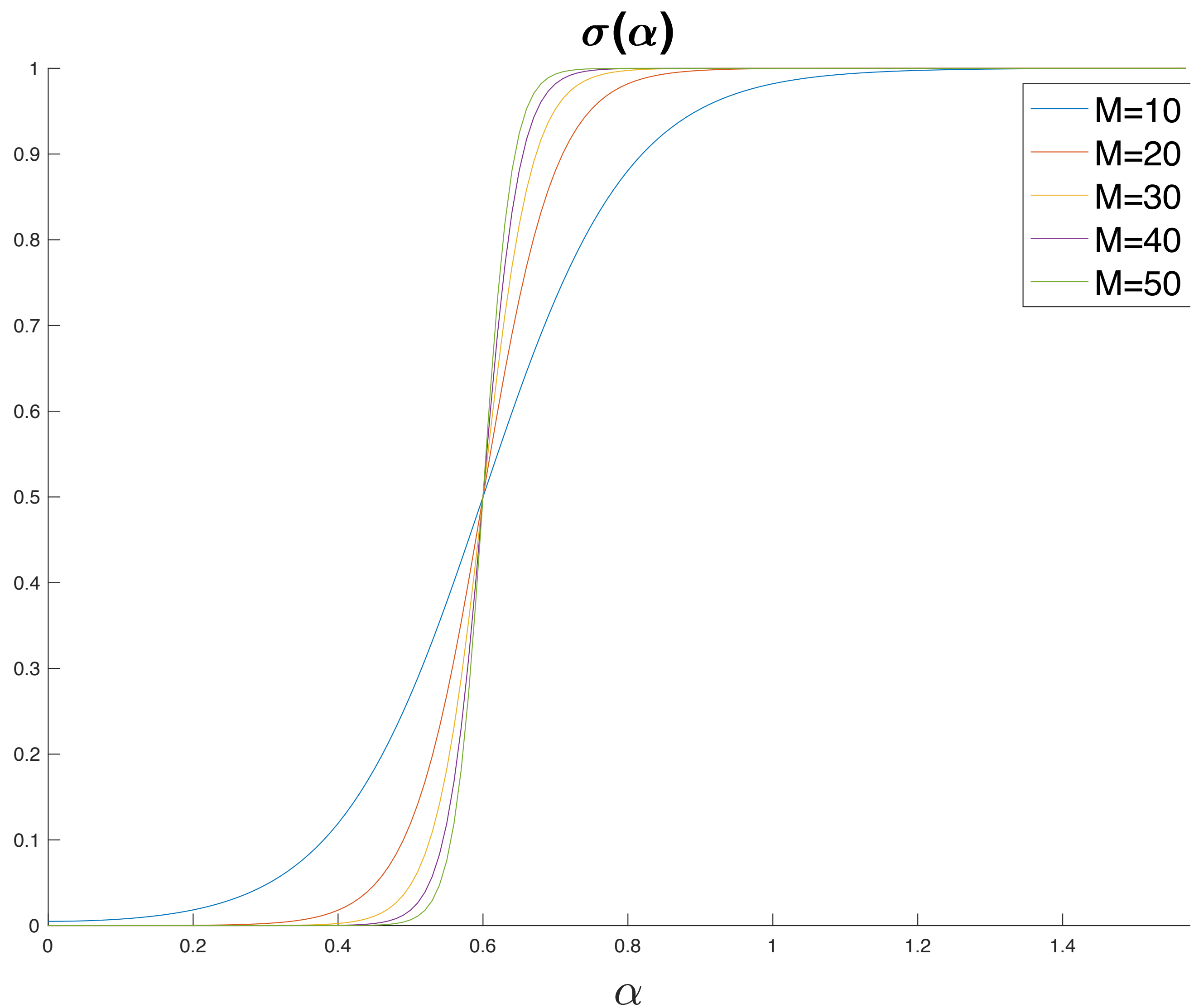
$$F_{\text{drag}} = \frac{1}{2}\rho V_a^2 S\left[\underline{C_D(\alpha)} + C_{D_q}\frac{c}{2V_a}q + C_{D_{\delta_e}}\delta_e\right]$$

$$C_L(\alpha) = (1-\sigma(\alpha))\left[\underbrace{C_{L_0} + C_{L_\alpha}\alpha}_{\text{linear model}}\right] + \sigma(\alpha)\left[\underbrace{2\,\mathrm{sign}(\alpha)\sin^2\alpha\cos\alpha}_{\text{flat-plate model}}\right]$$

$$\sigma(\alpha) = \frac{1 + e^{-M(\alpha-\alpha_0)} + e^{M(\alpha+\alpha_0)}}{\left(1 + e^{-M(\alpha-\alpha_0)}\right)\left(1 + e^{M(\alpha+\alpha_0)}\right)}$$

blending function

Blending Function



Nonlinear Aerodynamic Model

The stability derivative

$$C_{L_\alpha} = \frac{\pi AR}{1 + \sqrt{1 + (AR/2)^2}}$$

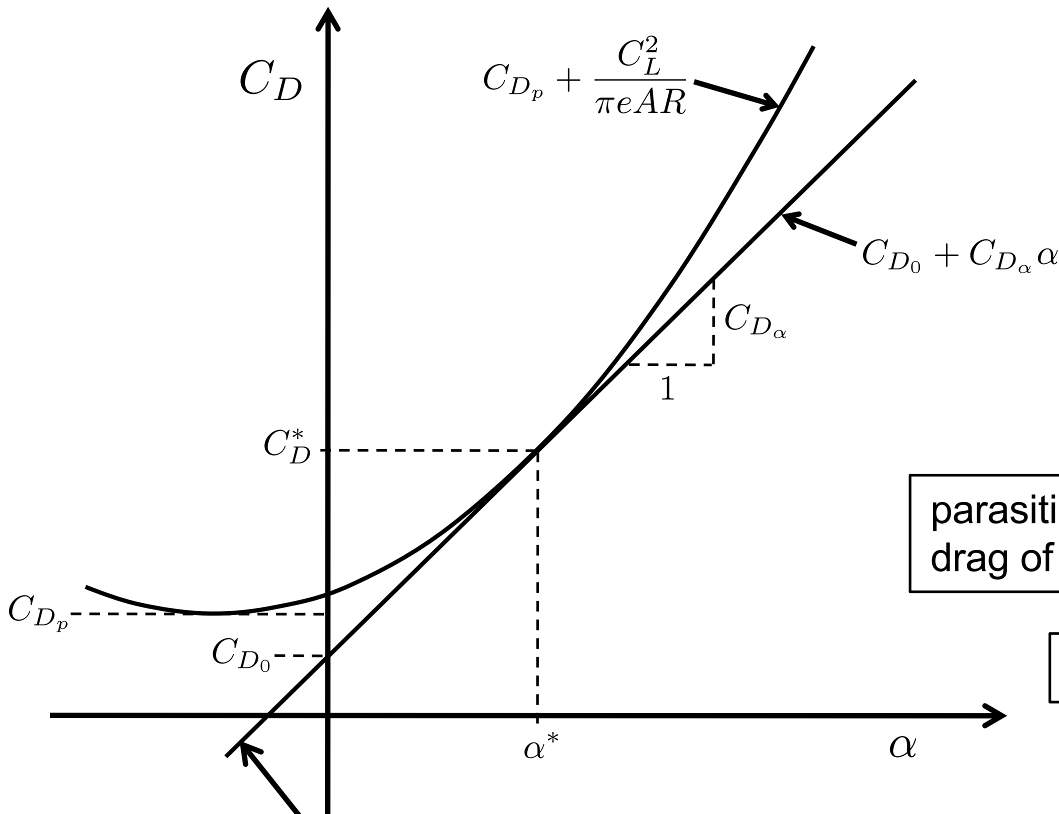
represents the sensitivity of lift to the angle of attack, and can be approximated by physical dimensions of the airfoil, where

$AR \stackrel{\triangle}{=} b^2/S$ is the wing aspect ratio

b is the wingspan

S is the wing area

Drag vs. Angle of Attack



linear model incorrect for negative angles of attack

$$C_D(\alpha) = C_{D_p} + \frac{(C_{L_0} + C_{L_\alpha}\alpha)^2}{\pi e AR}$$

parasitic drag

induced drag

e is the Oswald efficiency factor

parasitic drag: profile drag of wing, friction and pressure drag of tail surfaces, fuselage, landing gear, etc.

induced drag: "drag due to lift"

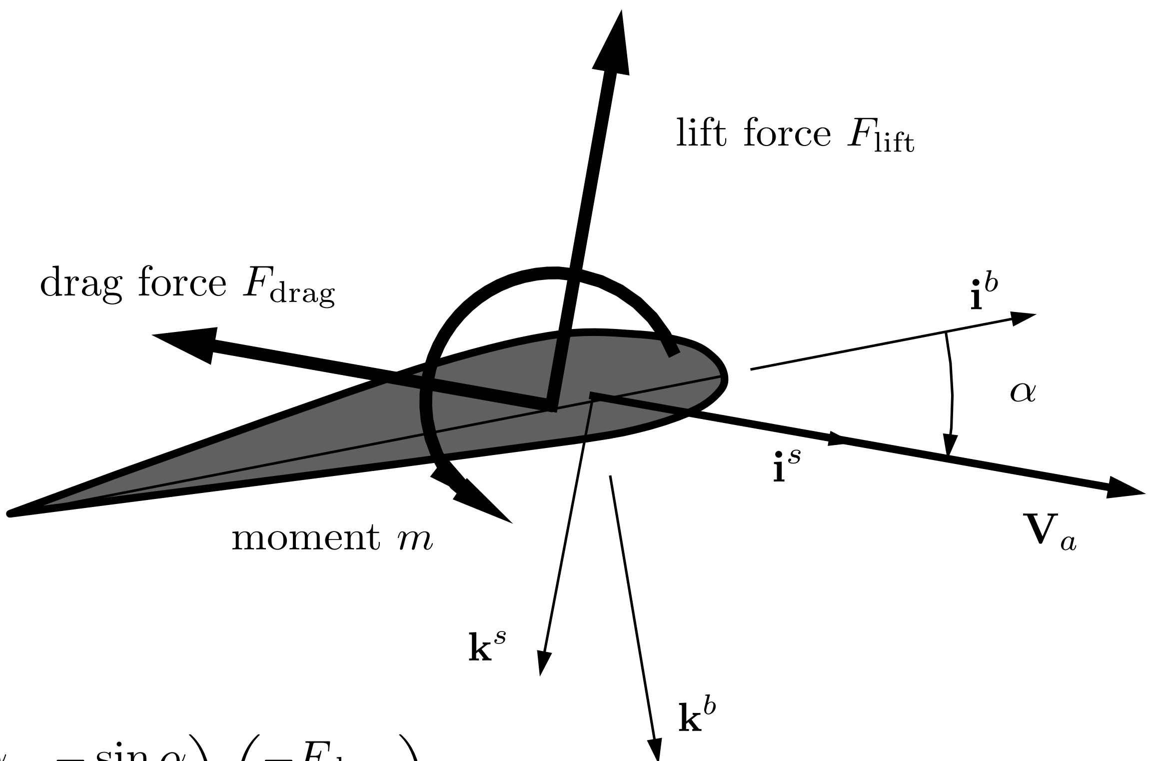
Linear Lift and Drag Models

The linear lift and drag models

$$\begin{aligned} C_L(\alpha) &= C_{L_0} + C_{L_\alpha}\alpha \\ C_D(\alpha) &= C_{D_0} + C_{D_\alpha}\alpha \end{aligned}$$

is valid for small deviations of angle of attack from trim.

Longitudinal Forces – Body Frame



$$\begin{pmatrix} f_x \\ f_z \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} -F_{\text{drag}} \\ -F_{\text{lift}} \end{pmatrix}$$

$$= \frac{1}{2}\rho V_a^2 S \begin{pmatrix} [-C_D(\alpha)\cos\alpha + C_L(\alpha)\sin\alpha] \\ + \left[-C_{D_q}\cos\alpha + C_{L_q}\sin\alpha\right] \frac{c}{2V_a} q + \left[-C_{D_{\delta_e}}\cos\alpha + C_{L_{\delta_e}}\sin\alpha\right]\delta_e \\ --- \\ [-C_D(\alpha)\sin\alpha - C_L(\alpha)\cos\alpha] \\ + \left[-C_{D_q}\sin\alpha - C_{L_q}\cos\alpha\right] \frac{c}{2V_a} q + \left[-C_{D_{\delta_e}}\sin\alpha - C_{L_{\delta_e}}\cos\alpha\right]\delta_e \end{pmatrix}$$

Pitching Moment

The linear pitching moment is given by

$$m = \frac{1}{2}\rho V_a^2 Sc \left[C_{m_0} + C_{m_\alpha}\alpha + C_{m_q}\frac{c}{2V_a}q + C_{m_{\delta_e}}\delta_e \right]$$

where no rotational transformation is necessary.

Lateral Aerodynamics

$$
\begin{aligned}
f_y &= \frac{1}{2}\rho V_a^2 S C_Y(\beta, p, r, \delta_a, \delta_r) \\
l &= \frac{1}{2}\rho V_a^2 S b C_l(\beta, p, r, \delta_a, \delta_r) \\
n &= \frac{1}{2}\rho V_a^2 S b C_n(\beta, p, r, \delta_a, \delta_r)
\end{aligned}
$$

Lateral Aerodynamics

$$f_y \approx \frac{1}{2}\rho V_a^2 S\left[C_{Y_0} + C_{Y_\beta}\beta + C_{Y_p}\frac{b}{2V_a}p + C_{Y_r}\frac{b}{2V_a}r + C_{Y_{\delta_a}}\delta_a + C_{Y_{\delta_r}}\delta_r\right]$$

$$l \approx \frac{1}{2}\rho V_a^2 Sb\left[C_{l_0} + C_{l_\beta}\beta + C_{l_p}\frac{b}{2V_a}p + C_{l_r}\frac{b}{2V_a}r + C_{l_{\delta_a}}\delta_a + C_{l_{\delta_r}}\delta_r\right]$$

$$n \approx \frac{1}{2}\rho V_a^2 Sb\left[C_{n_0} + C_{n_\beta}\beta + C_{n_p}\frac{b}{2V_a}p + C_{n_r}\frac{b}{2V_a}r + C_{n_{\delta_a}}\delta_a + C_{n_{\delta_r}}\delta_r\right]$$

where for symmetric aircraft, $C_{Y_0} = C_{l_0} = C_{n_0} = 0$.

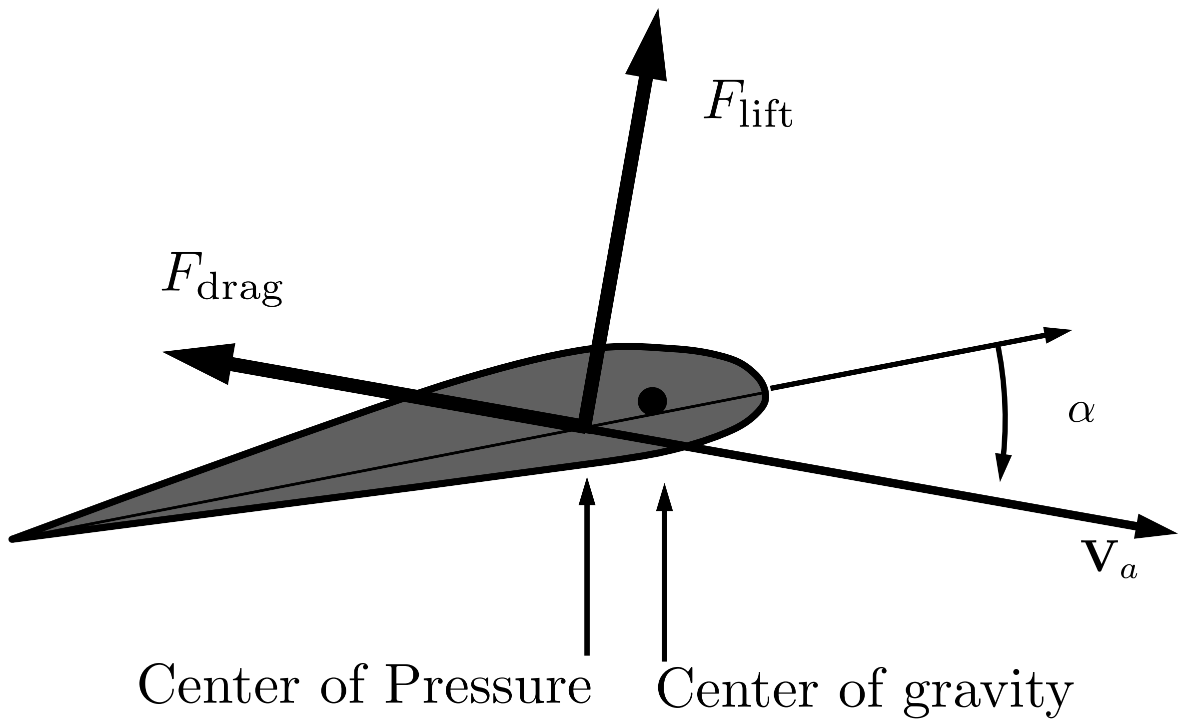
Aerodynamic Coefficients

C_{m_α}, C_{ℓ_β}, C_{n_β}, C_{m_q}, C_{ℓ_p}, C_{n_r} are called the *stability derivatives* because their values determine the stability of the aircraft.

Static Stability Derivatives

- C_{m_α} - longitudinal static stability derivative. Must be ≤ 0 for stability: increase in α causes and downward pitching moment.
- C_{ℓ_β} - roll static stability derivative. Associated with dihedral in wings. Must be ≤ 0 for stability: positive roll ϕ causes a restoring moment.
- C_{n_β} - yaw static stability derivative. Weathercock stability derivative. Influenced by design of tail. Causes airframe to align with the wind vector. Must be ≥ 0 for stability: cocks airframe into wind driving β to zero.

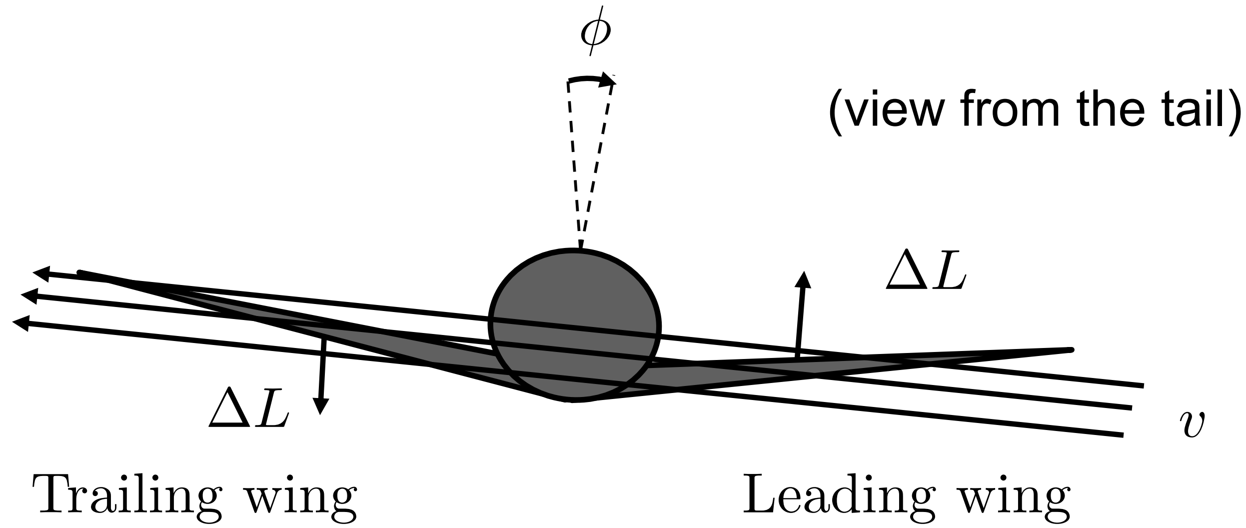
Longitudinal Static Stability Derivative



Center of Pressure is the point where there is no moment due to aerodynamic forces. The lift and drag forces act at this point.

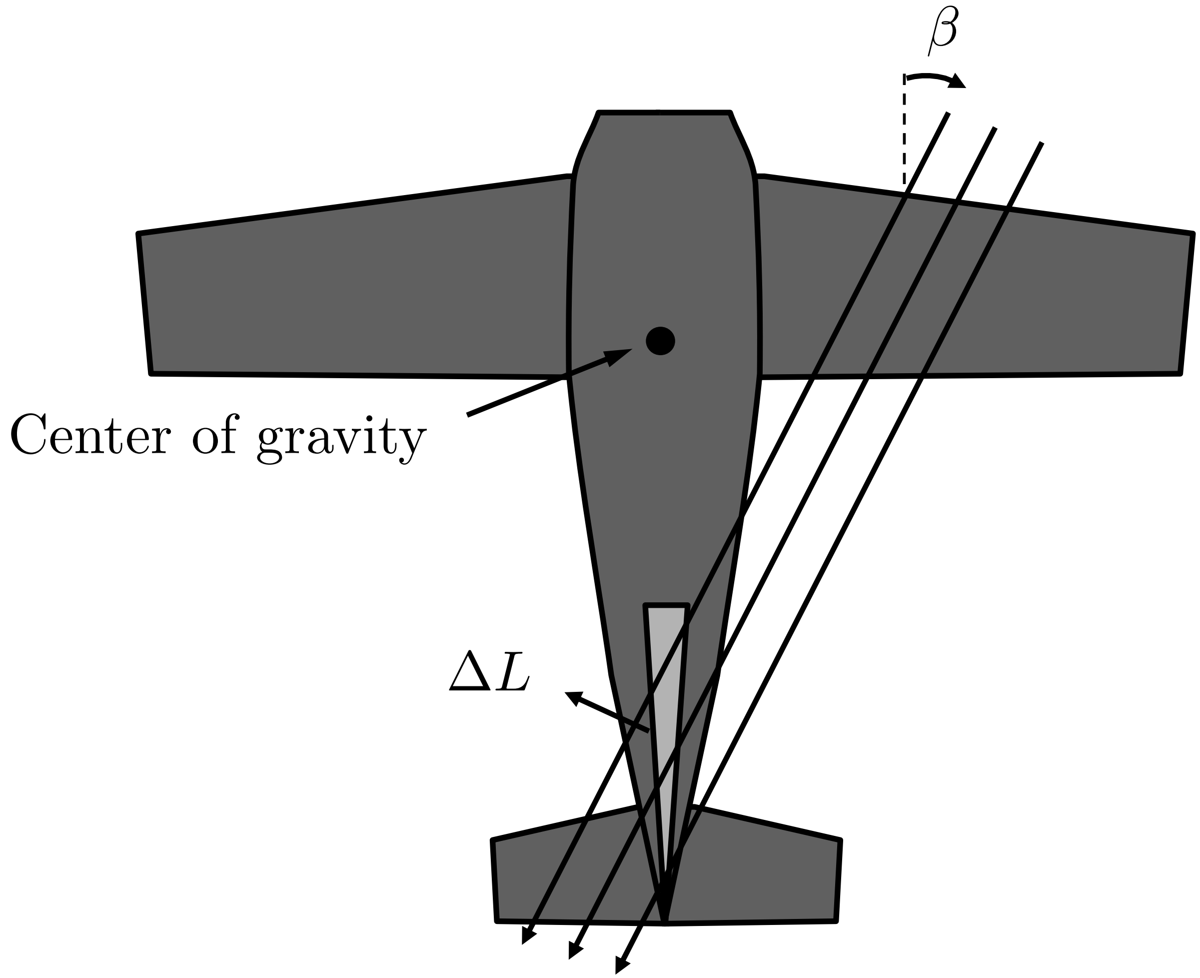
If the center of pressure is behind the center of gravity, then when $\alpha \neq 0$, the lift force will tend to push α back to zero. Otherwise, the lift force will tend to increase $|\alpha|$. Therefore $C_{m_\alpha} < 0$.

Roll Static Stability Derivative



Given the wind dihedral, a roll angle of ϕ will cause a side slip angle β, which induces a side velocity v, which increases the lift on the leading wing, and decreases the lift on the trailing wing, causing a negative rolling moment. Hence the dihedral angle causes $C_{\ell_\beta} < 0$.

Yaw Static Stability Derivative



For a positive side slip angle β, the change in lift on the tail creates a moment arm about the center of gravity, that pushes the nose toward the direction of the wind, or in other words, creates a positive yawing moment n. Hence $C_{n_\beta} > 0$.

Aerodynamic Coefficients

Dynamic Stability Derivatives

- $C_{m_q}, C_{\ell_p}, C_{n_r}$ are known as the pitch damping derivative, roll damping derivative, and yaw damping derivative, respectively. They quantify the level of damping associated with angular motion of the airframe.

Control Derivatives

- $C_{m_{\delta_e}}$, $C_{\ell_{\delta_a}}$, and $C_{n_{\delta_r}}$ are the primary control derivatives and quantify the effect on the control surfaces on their primary intended axes of influence.
- $C_{\ell_{\delta_r}}$ and $C_{n_{\delta_a}}$ are the cross-control derivatives.

Propeller Thrust and Torque

The thrust T_p and the torque Q_p along the propeller axis can be modeled as

$$T_p = \rho n^2 D^4 C_T$$
$$Q_p = \rho n^2 D^5 C_Q,$$

where

$$J \stackrel{\triangle}{=} \frac{V_a}{nD} = \frac{2\pi V_a}{\Omega D}, \text{ advance ratio (unitless)}$$
$$D \stackrel{\triangle}{=} \text{ propeller diameter (m)}$$
$$n \stackrel{\triangle}{=} \text{ propeller speed (revolutions/sec)}$$
$$\Omega \stackrel{\triangle}{=} 2\pi n \text{ propeller speed (rad/sec)}$$

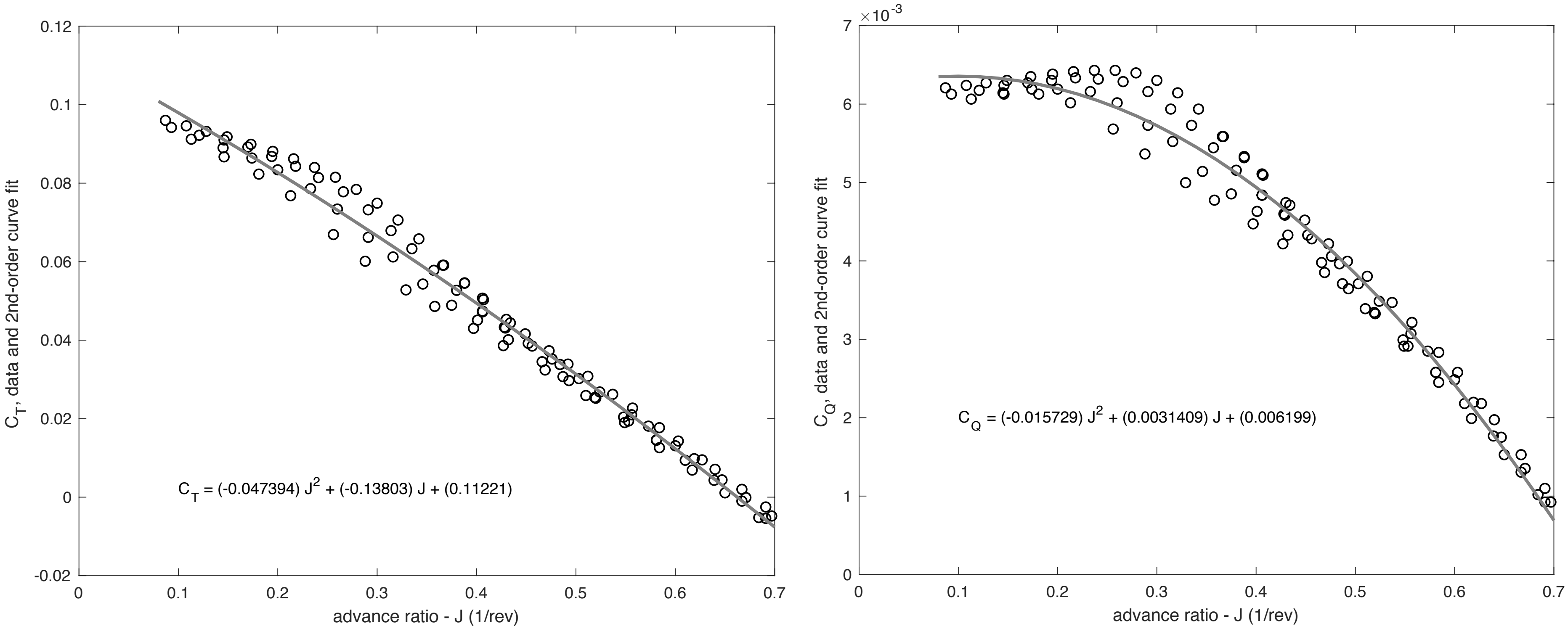
Propeller Thrust and Torque

If the propeller shaft is aligned along the body frame $\mathbf{i}^b$ axis, then the resulting force and torque are given by

$$\mathbf{f}^b = \begin{pmatrix} T_p \\ 0 \\ 0 \end{pmatrix}$$

$$\mathbf{m}^b = \begin{pmatrix} Q_p \\ 0 \\ 0 \end{pmatrix}.$$

Propeller Thrust and Torque



Experimental data indicates that

$$C_Q(J) \approx C_{Q2}J^2 + C_{Q1}J + C_{Q0}$$
$$C_T(J) \approx C_{T2}J^2 + C_{T1}J + C_{T0}$$

Propeller Thrust and Torque

For a DC motor, the steady-state torque generated for a given input voltage V_{in} is given by

$$Q_m = K_Q \left[\frac{1}{R} (V_{in} - K_V \Omega) - i_0 \right],$$

where

$$\begin{aligned} R &\stackrel{\triangle}{=} \text{resistance of the motor windings} \\ K_Q &\stackrel{\triangle}{=} \text{motor torque constant} \\ K_V &\stackrel{\triangle}{=} \text{back-emf voltage constant} \\ i_0 &\stackrel{\triangle}{=} \text{zero-torque or no-load current} \end{aligned}$$

Propeller Thrust and Torque

Setting $Q_p = Q_m$ gives

$$\left(\frac{\rho D^5}{(2\pi)^2}C_{Q0}\right)\Omega^2 + \left(\frac{\rho D^4}{2\pi}C_{Q1}V_a + \frac{K_Q K_V}{R}\right)\Omega + \left(\rho D^3 C_{Q2}V_a^2 - \frac{K_Q}{R}V_{in} + K_Q i_0\right) = 0.$$

Solving for the positive root gives the operating rotor speed

$$\Omega_{\text{op}} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

where

$$\begin{aligned}
a &= \frac{\rho D^5}{(2\pi)^2}C_{Q0} \\
b &= \frac{\rho D^4}{2\pi}C_{Q1}V_a + \frac{K_Q K_V}{R} \\
c &= \rho D^3 C_{Q2}V_a^2 - \frac{K_Q}{R}V_{in} + K_Q i_0.
\end{aligned}$$

The associated advance ratio is

$$J_{\text{op}} = \frac{2\pi V_a}{\Omega_{\text{op}} D}.$$

Propeller Thrust and Torque

```
# compute thrust and torque due to propeller  (See addendum by McLain)
# map delta_t throttle command(0 to 1) into motor input voltage
V_in = MAV.V_max * delta_t
# Quadratic formula to solve for motor speed
a = MAV.C_Q0 * MAV.rho * np.power(MAV.D_prop, 5) \
    / ((2.*np.pi)**2)
b = (MAV.C_Q1 * MAV.rho * np.power(MAV.D_prop, 4)
     / (2.*np.pi)) * self._Va + KQ**2/MAV.R_motor
c = MAV.C_Q2 * MAV.rho * np.power(MAV.D_prop, 3) \
    * self._Va**2 - (KQ / MAV.R_motor) * Volts + KQ * MAV.i0
# Consider only positive root
Omega_op = (-b + np.sqrt(b**2 - 4*a*c)) / (2.*a)
# compute advance ratio
J_op = 2 * np.pi * self._Va / (Omega_op * MAV.D_prop)
# compute non-dimensionalized coefficients of thrust and torque
C_T = MAV.C_T2 * J_op**2 + MAV.C_T1 * J_op + MAV.C_T0
C_Q = MAV.C_Q2 * J_op**2 + MAV.C_Q1 * J_op + MAV.C_Q0
# add thrust and torque due to propeller
n = Omega_op / (2 * np.pi)
fx += MAV.rho * n**2 * np.power(MAV.D_prop, 4) * C_T
Mx += -MAV.rho * n**2 * np.power(MAV.D_prop, 5) * C_Q
```

Wind Model

From the wind triangle we have

$$\mathbf{V}_g = \mathbf{V}_a + \mathbf{V}_w.$$

The wind vector can be decomposed into steady state and gust components:

$$\mathbf{V}_w = \mathbf{V}_{w_s} + \mathbf{V}_{w_g}.$$

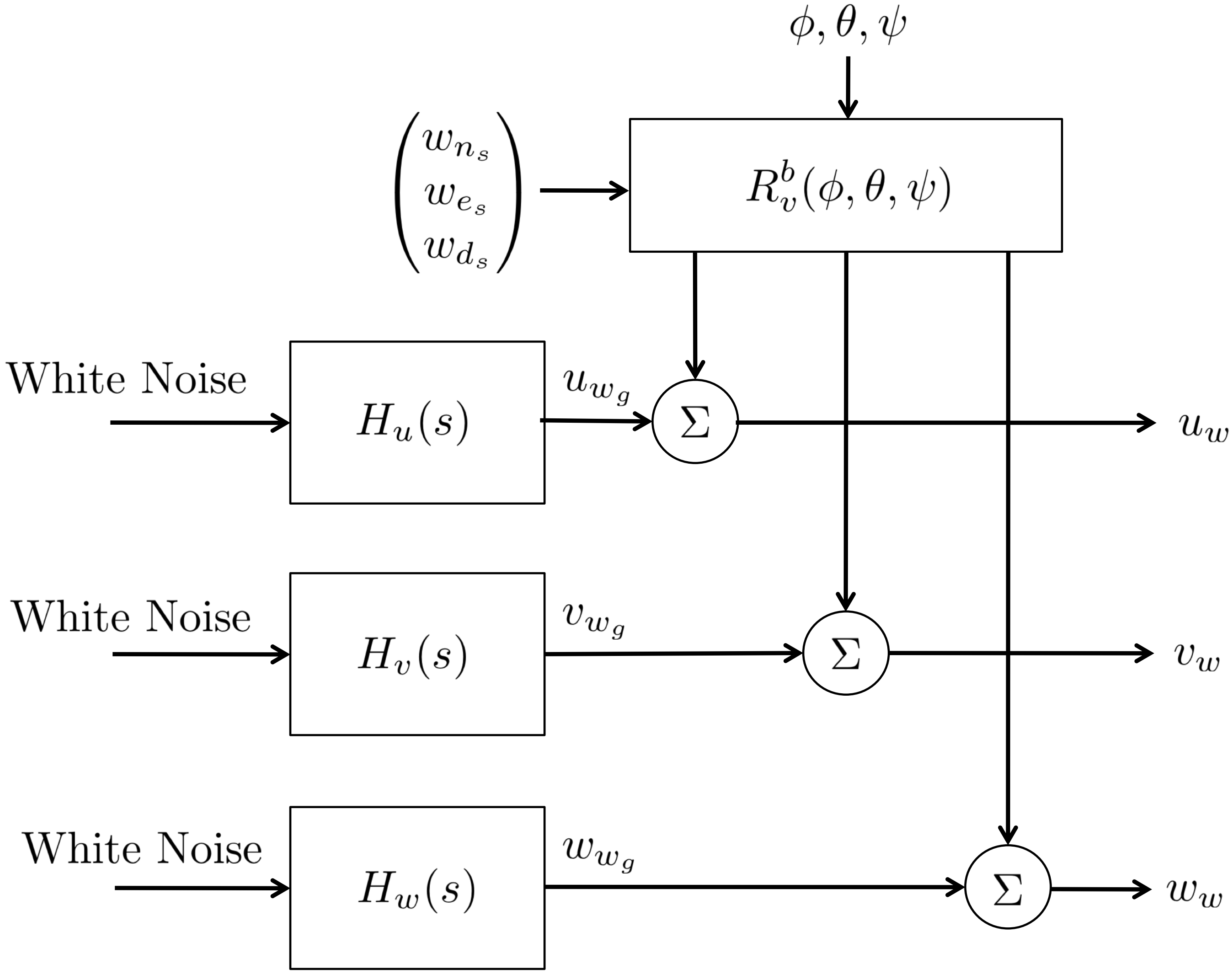
The steady state component is typically expressed in the NED frame:

$$\mathbf{V}_{w_s}^i = \begin{pmatrix} w_{n_s} \\ w_{e_s} \\ w_{d_s} \end{pmatrix}.$$

The gust component is typically expressed in the body frame:

$$\mathbf{V}_{w_g}^b = \begin{pmatrix} u_{w_g} \\ v_{w_g} \\ w_{w_g} \end{pmatrix}.$$

Wind Model



ϕ, θ, ψ
$\begin{pmatrix} w_{n_s} \\ w_{e_s} \\ w_{d_s} \end{pmatrix}$
$R_v^b(\phi, \theta, \psi)$
White Noise
$H_u(s)$
u_{w_g}
Σ
u_w
White Noise
$H_v(s)$
v_{w_g}
Σ
v_w
White Noise
$H_w(s)$
w_{w_g}
Σ
w_w

Dryden Gust Model

$$H_u(s) = \sigma_u \sqrt{\frac{2V_a}{L_u}} \frac{1}{s + \frac{V_a}{L_u}}$$

$$H_v(s) = \sigma_v \sqrt{\frac{3V_a}{L_v}} \frac{\left(s + \frac{V_a}{\sqrt{3}L_v}\right)}{\left(s + \frac{V_a}{L_v}\right)^2}$$

$$H_w(s) = \sigma_w \sqrt{\frac{3V_a}{L_w}} \frac{\left(s + \frac{V_a}{\sqrt{3}L_w}\right)}{\left(s + \frac{V_a}{L_w}\right)^2}$$

Table 1: Dryden gust model parameters

gust description	altitude (m)	$L_u = L_v$ (m)	L_w (m)	$\sigma_u = \sigma_v$ (m/s)	σ_w (m/s)
low altitude, light turbulence	50	200	50	1.06	0.7
low altitude, moderate turbulence	50	200	50	2.12	1.4
medium altitude, light turbulence	600	533	533	1.5	1.5
medium altitude, moderate turbulence	600	533	533	3.0	3.0

Transfer Function Implementation

To implement the system

$$Y(s) = \frac{as+b}{s^2+cs+d}U(s),$$

first put the system into control canonical form

$$\dot{x} = \begin{pmatrix} -c & -d \\ 1 & 0 \end{pmatrix} x + \begin{pmatrix} 1 \\ 0 \end{pmatrix} u$$
$$y = \begin{pmatrix} a & b \end{pmatrix} x,$$

and then convert to discrete time using

$$x_{k+1} = x_k + T_s(Ax_k + Bu_k)$$
$$y_k = Cx_k$$

where T_s is the sample rate, to get

$$x_{k+1} = \begin{pmatrix} 1 - T_s c & -T_s d \\ T_s & 1 \end{pmatrix} x_k + \begin{pmatrix} T_s \\ 0 \end{pmatrix} u_k$$
$$y_k = \begin{pmatrix} a & b \end{pmatrix} x_k.$$

For the Dryden model gust models, the input u_k will be zero mean Gaussian noise with unity variance.

Propeller Thrust and Torque

```
class transfer_function:
    def __init__(self, Ts):
        self.ts = Ts
        # set initial conditions
        self._state = np.array([[0.0], [0.0]])
        # define state space model
        self._A = np.array([[1-Ts*c, -Ts*d], [Ts, 1]])
        self._B = np.array([[Ts], [0]])
        self._C = np.array([[a, b]])

    def update(self, u):
        '''Update state space model'''
        self._state = self._A @ self._state + self._B * u
        y = self._C @ self._state
        return y

# initialize the system
Ts = 0.01  # simulation step size
system = transfer_function(Ts)

# main simulation loop
sim_time = 0.0
while sim_time < 10.0:
    u=np.random.randn()  # (white noise)
    y = system.update(u)  # update based on current input
    sim_time += Ts   # increment the simulation time
```

Adding in the Effects of Wind

The wind vector is given by

$$\mathbf{V}_w^b = \begin{pmatrix} u_w \\ v_w \\ w_w \end{pmatrix} = \mathcal{R}_v^b(\phi, \theta, \psi) \begin{pmatrix} w_{n_s} \\ w_{e_s} \\ w_{d_s} \end{pmatrix} + \begin{pmatrix} u_{w_g} \\ v_{w_g} \\ w_{w_g} \end{pmatrix}.$$

The airspeed vector is adjusted to be

$$\mathbf{V}_a^b = \begin{pmatrix} u_r \\ v_r \\ w_r \end{pmatrix} = \begin{pmatrix} u - u_w \\ v - v_w \\ w - w_w \end{pmatrix}.$$

The airspeed, angle of attack, and side slip angle are given by

$$V_a = \sqrt{u_r^2 + v_r^2 + w_r^2}$$

$$\alpha = \tan^{-1}\left(\frac{w_r}{u_r}\right)$$

$$\beta = \sin^{-1}\left(\frac{v_r}{\sqrt{u_r^2 + v_r^2 + w_r^2}}\right)$$

Project 4

1. Add simulation of the wind to the mavsim simulator. The wind element should produce wind gust along the body axes, and steady state wind along the NED inertial axes.

2. Add forces and moments to the dynamics of the MAV. The inputs to the MAV should now be elevator, throttle, aileron, and rudder. The aerodynamic coefficients are given in Appendix E.

3. Verify your simulation by setting the control surface deflections to different values. Observe the response of the MAV. Does it behave as you think it should?