Steps

1). Assumptions & Definitions — need to make sure we are careful about ft vs m units when creating controller

- No thrust, mass = const.
- $F_D = \frac{1}{2}\rho(h)C_D(u)A(u)v|v|$
    - $u \in [0,1]$ is airbrake deployment fraction
    - use lookup table for $\rho(h) \rightarrow \rho(h) = \text{interp1}(h_{table}, \rho_{table}, h)$ → can use 1976 Standard Atmosphere
- $g(h) = g_0\left(\frac{R_e}{R_e+h}\right)^2$
- Avionics provide current h, v, α at predetermined control update times

2). Physics Model

Variables:

- $h(t)$ = altitude
- $v(t)$ = vertical velocity
- m = rocket dry mass
- $g(h)$ = gravity at altitude h
- $\rho(h)$ = air density at altitude h
- $C_D(\alpha, u)$ = drag coeff. as a function of a.o.a. α and airbrake deployment u } can choose either C_D or A to vary as a fxn of α, u depending on what data we have
- A = reference area
- u = airbrake deployment (0→1)

Equations of Motion:

$$\dot{h} = v$$
$$\dot{v} = -g(h) + \frac{1}{2m}\rho(h)\, v|v|\, C_D(\alpha,u)A$$

?

Drag Coefficient Model

$$C_D(\alpha,u) = C_{D_0} + k_\alpha \alpha^2 + (C_{D,brake}(u) - C_{D_0})$$

- C_{D_0} = base drag coefficient
- k_α models increase w/ a.o.a.
- $C_{D,brake}(u)$ from lookup table

3). Apogee Prediction

- Simulate until $v(t) = 0$
- Use numerical integration (ode45 in MATLAB, scipy.integrate.solve_ivp in Python) of above dynamics

4). MPC Objective

At each timestep (in Hz?):

1. Measure h, v, α
2. Predict apogee $h_{apogee}(u)$ using current model & possible future u values
3. Compute target apogee logic:

$$h_{\text{target}} = \begin{cases} h_{\text{desired}}, & h_{\text{apogee}}(0) > h_{\text{desired}} \\ \max(\text{floor}(h_{\text{apogee}}(0)/2500) \times 2500, \text{min altitude}), & \text{otherwise} \end{cases}$$

4. Optimize u to minimize error cost function:

$$J(u) = |h_{pred}(u) - h_{target}| + R_u(u - u_{prev})^2$$ → R_u penalizes large deployment changes

5. Apply optimal u^* for next control cycle