Accuracy and maneuverability of atmospheric reentry vehicles

Kai Van Brunt • kav@mit.edu

This work is built upon the existing open-source `FORTRAN` program `MC-NEW` [1]. That paper explains how the Monte Carlo simulation here works.

My modified version of `MC-NEW` to calculate aerodynamic coefficients, including the `MC-NEW` manual (useful for running the code and looking at output yourself): `https://github.com/tidalove/mc-new`

`rv_dynamics.py` to calculate time constants: `https://github.com/tidalove/rv-dynamics`

Contents

1 Problem scope

Our goal is to find a time constant governing the decay of an angular impulse during the flight of an atmospheric reentry vehicle. For a vehicle symmetric along pitch and yaw axes, a single time constant governing the change in time of a generalized angle of attack can be found. Specifically, the time constant we want to find governs the decay of the angle of attack to its trim angle through a decaying exponential dependence $e^{-t/\tau}$ where τ is the time constant. (Aerodynamic trim refers to an equilibrium state when the vehicle is not experiencing a net moment.)

This is relevant for the maneuverability of reentry vehicles: a reentry vehicle with a greater time constant takes longer to reach the trim condition. So if the control surfaces of such a craft deflect a certain amount, the craft is tilted by the torque exerted on the control surfaces, but it takes some amount of time - which depends on τ - for the velocity of the craft to reach the desired velocity.

We are interested in modeling the motion of SWERVE (Sandia Winged Energetic Reentry Vehicle Experiment), a hypersonic reentry vehicle built by Sandia National Laboratories (figure 2).

Following the conventions in `MC-NEW`, the Monte Carlo code I use to help determine this time constant, I take the x axis as the axis of symmetry of the vehicle (the axis along which the nose points; equivalently, a vehicle pinned along the x axis can only roll); the y axis as the direction the vehicle pitches in (equivalently, a vehicle pinned along the y axis can only yaw), and the z axis as the direction the vehicle yaws in (equivalently, a vehicle pinned along the z axis can only pitch).

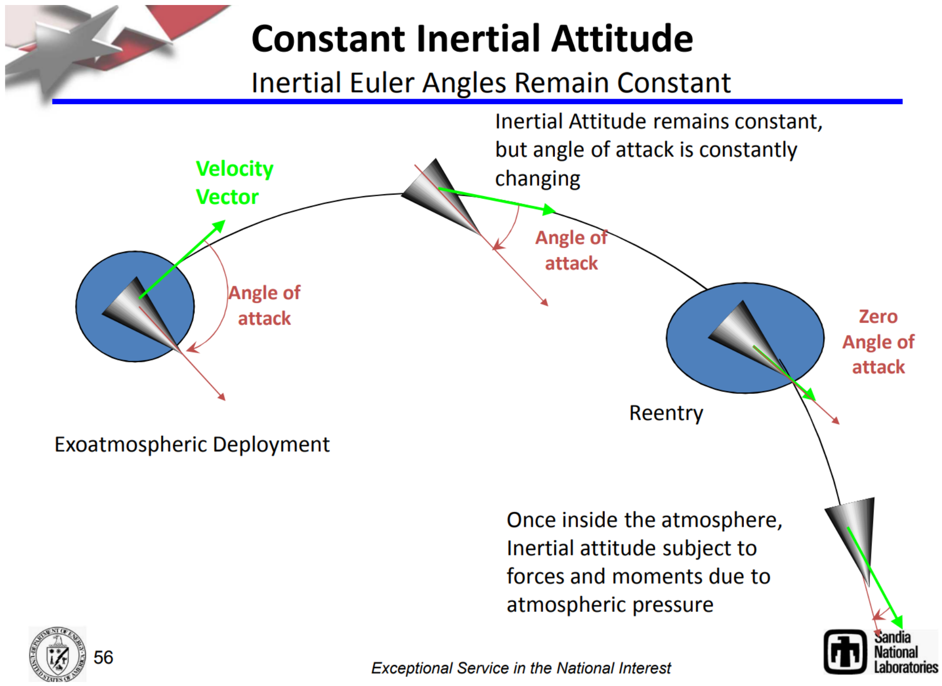


Figure 1: While outside the atmosphere, this vehicle's angle of attack is changing but it is not pitching; thus $\dot{\alpha} \neq q = 0$.

1.1 Aerodynamic coefficients

Modeling the aircraft as being pinned along the y axis through its center of gravity, we can derive a second-order differential equation for the angular (pitching) motion of the craft. The general form of this equation reads (assuming the xyz body axes of the vehicle are its principal axes, which is true for SWERVE):

$$I_y \ddot{\alpha} = M(\alpha, \dot{\alpha}, q, \delta_e) \approx M_\alpha \alpha + M_{\dot{\alpha}} \dot{\alpha} + M_q q + M_{\delta_e} \delta_e \tag{1}$$

where α is the angle of attack, q is the rate of pitching motion, and δ_e is the deflection of the control surfaces (for SWERVE, elevons).

$\dot{\alpha}$ and q can be different in situations where the nose of the vehicle does not point in the same direction as its velocity. One example is illustrated in figure 1.

For now we will work with an uncontrolled craft ($\delta_e = 0$). In situations such as a wind tunnel test, the assumption $\dot{\alpha} = q$ can be made, such that equation 1 becomes:

$$\ddot{\alpha} - \frac{1}{I_y}(M_q + M_{\dot{\alpha}})\dot{\alpha} - \frac{1}{I_y} M_\alpha \alpha = 0 \tag{2}$$

In general, although $q \neq \dot{\alpha}$, the difference between them is small. This difference can be calculated by considering C_{N_α} and goes as $\rho V S \alpha / m$; see Chapter 12 of Regan and Anandakrishnan (1993) for details. This quantity was small when I calculated it for SWERVE: including a dependence on C_{N_α} did not visibly affect the final results. When neglecting this difference, the final equation of motion is [5]:

$$\ddot{\alpha} - \frac{\rho V S c^2}{2I_y} C_{m_q} \dot{\alpha} - \frac{\rho V^2 S c}{2I_y} C_{m_\alpha} \alpha = 0 \tag{3}$$

Here, the coefficients M_q and M_α have been replaced by their respective non-dimensional forms which are dependent only on geometry (the dependences on vehicle velocity V and atmospheric density ρ are removed):

$$M_\alpha = \frac{\rho V^2 S c}{2} C_{M_\alpha} \tag{4}$$

$$M_q = \frac{\rho V S c^2}{4} C_{M_q} \tag{5}$$

Here, S is the reference area of the body and c is the reference length. The choice of those values makes no physical difference since they cancel out in the derivation of any physical quantity, including the time constant. I will take S to be the cross-sectional area of the craft and c to be the total length (tip-to-tail) of the craft. Most sources agree with this convention for S, but sometimes the "aerodynamic chord" is used for c instead, especially in situations involving airfoils. The exact choice is not important as long as it is consistent throughout a calculation, though it does affect comparisons between the C_i coefficients between different sources.

If all parameters in equation 3 are known, then the motion can be characterized by two parameters ζ and ω_n. Take the following second-order differential equation:

$$\ddot{x} + 2\zeta\omega_n \dot{x} + \omega_n^2 x = 0 \tag{6}$$

If the system is underdamped ($\zeta < 1$), as SWERVE's parameters lend it to be in our calculations, then the motion is characterized by a solution of the form $\alpha(t) = c_1 e^{\lambda_1 t} + c_2 e^{\lambda_2 t}$, with:

$$\lambda_{1,2} = -\zeta\omega_n \pm \omega_n \sqrt{\zeta^2 - 1} \tag{7}$$

Thus the time constant τ we are after such that the envelope of the angle of attack over time is proportional to $e^{-t/\tau}$ is given by:

$$\tau = \frac{1}{\zeta\omega_n} = -\frac{4I_y}{\rho V S c^2 C_{M_q}} \tag{8}$$

The time constant governing decay of angle of attack only depends on C_{M_q} (referred to as the "pitch damping moment") and not C_{M_α} (referred to as the "pitching moment"). However, C_{M_α}

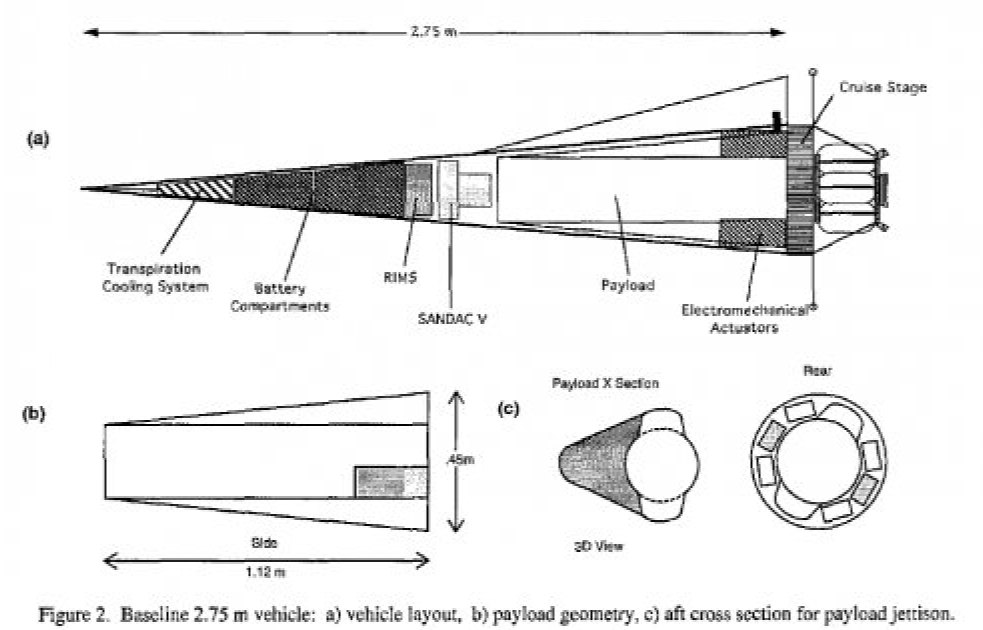


Figure 2: Diagram of SWERVE according to some guy on Secret Projects Forum

is still informative since it determines ω_n, the frequency of oscillation of the vehicle, and some assumptions are not valid at high frequencies.

1.2 SWERVE dimensions

SWERVE's dimensions are assumed to be as follows:

- Tip-to-tail length: 2.75m ([4], corroborated by [3])
- Nose half-angle: 5.25° ([3], corroborated by [4])
- Nose tip radius: 1.75cm (calculated from ratio to base radius as given in [3])
- Base radius: 0.277m (calculated from total length and half-angle)
- Wing base chord (projection measured along symmetry axis of vehicle): 1.24m (based on images)
- Wing length: 0.20m (based on images)
- Wing thickness: 5cm (just a guess, but see section 4 for a stability check on this value)

2 Monte Carlo simulation

To derive these aerodynamic coefficients, we use the `MC-NEW` code [1] which performs a Monte Carlo integration over the surface of the craft. Each data point in this report was acquired by running 100 trials with 100,000 particles per trial. Based on the number of trials, the error may be around $\pm 10\%$, though in practice it usually seems quite a bit lower than that since re-running the same code multiple times (on random seeds) reliably gives the same results to at least 2 significant figures.

`MC-NEW` usage summary

1. Modify or create an input file (file extension `.flap`) with desired geometry and run parameters. The reference area should be the cross-sectional area of the body (in this case, base area of the cone) and the reference length should be the total tip-to-base length of the body.
2. On Linux, run `./MC-NEW < input.flap > out`. All existing output files in the directory (including `out` and all `.dat` files) will be overwritten with the results from the new run.
3. To extract C_{M_α} coefficients for a certain angle of attack, see section 2.3.

2.1 Flap geometry

MC-NEW specifies all geometries using polar coordinates. Each possible component geometry in MC-NEW is parameterized by two quantities, typically $u = x/h$ which specifies the proportion along the length of the symmetry axis that has been traveled starting from the nose, and the angle θ in the $y-z$ plane. For any given u and θ, a point on the surface of the geometry is specified and returns:

- that point's xyz coordinates,
- the normal vector in xyz coordinates of the point, and
- the differential surface area at that point, which will be used to calculate an integral over area.

2.1.1 xyz coordinates

Currently there is no way to incorporate an explicit r-dependence in the geometry of a component that already has a u- and θ-dependence. Thus, the "flaps" (wings on SWERVE) are specified using polar coordinates such that a cross-section in the $y-z$ plane of one flap returns a thin ellipse, which should serve as a decent approximation to any reasonably thin and flat flap. The three parameters relevant to the flap geometry, two of which must be specified by the user in the `.flap` input file, are:

- a: flap length; corresponds to maximum semimajor axis of ellipse minus base cone radius
- l: flap base chord (along the x-axis)
- b: hard-coded as 2.5cm; corresponds to semiminor axis of ellipse

Given these parameters, the equations specifying the flaps pointing along the y- and z-directions, respectively, are:

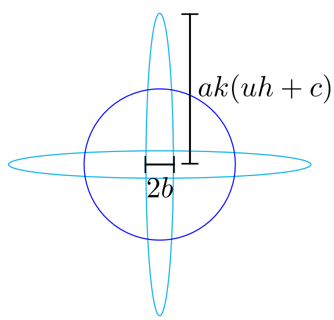
$$r = \frac{ak(uh + c) \cdot b}{\sqrt{(b \cos\theta)^2 + (ak(uh + c)\sin\theta)^2}} \tag{9}$$

where c and k are defined such that the semimajor axis $ak(uh+c)$ increases linearly from $x = h - l$ to $x = h$, resulting in triangular flaps:

$$k = \frac{r}{ah} + \frac{1}{l} \tag{10}$$

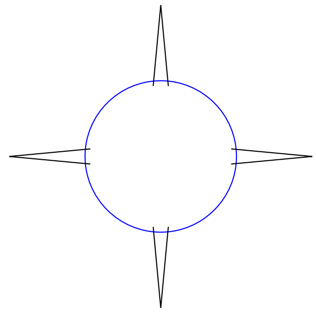
$$c = \frac{1 + r/a}{k} - h \tag{11}$$

A cross-sectional view of the craft at a certain x-value is shown.



2.1.2 Normal vectors

Finding the exact components of the normal vectors to the SWERVE flaps assuming an elliptical shape would require somewhat cumbersome calculations and would cause the program to run more slowly. Hence, I assume the normal vector to the surface is constant at every point along the surface of a given side of a flap. In particular, I now model the flap cross-section as a thin isosceles triangle:



So the normal vectors on each surface are given as some combination of $(\cos\theta, \sin\theta)$ where θ is the half-angle of the flap.

2.1.3 Differential area

Nearly every quantity output by `MC-NEW` relies on performing integration over surface area. To make this feasible, once again, I'll assume the cross-section of a flap can be modeled as a thin isosceles triangle. The Monte Carlo integration is parameterized by non-Cartesian coordinates u and θ; some function $\mathbf{f} : \mathbb{R}^2 \to \mathbb{R}^3$ maps these coordinates to the Cartesian 3D surface of the body. The formula for surface area is then given by:

$$A = \int_S \left\| \frac{\partial \mathbf{f}}{\partial u} \times \frac{\partial \mathbf{f}}{\partial \theta} \right\| du\, d\theta \tag{12}$$

This was already calculated and implemented for the conical body of the vehicle, so I will focus solely on the flaps. Take the flaps pointing in the y-direction as an example first. For thin enough flaps, we can approximate $\frac{\partial z}{\partial \theta} = \frac{\partial z}{\partial u} = 0$. By definition $\frac{\partial x}{\partial u} = h$ and from the triangular shape of the flaps we see that $\frac{\partial y}{\partial u} = akh$. Finally, after calculating the arc subtended by the base of the flap $\Delta\theta$, we have $\frac{\partial y}{\partial \theta} = \pm \frac{ak(uh+c)-r}{\Delta\theta/2}$, with the sign depending on which side of the flap we are on.

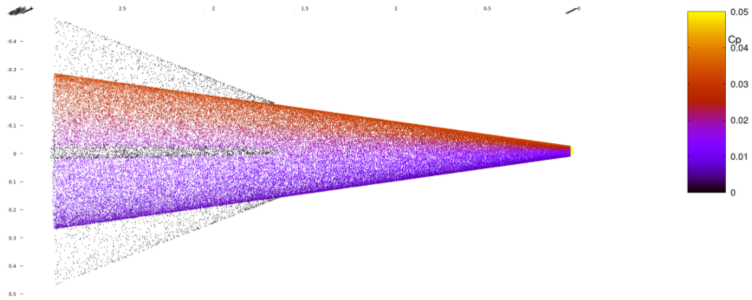
So:

$$A = \int_S \frac{akh(uh+c) - rh}{\Delta\theta/2}\, du\, d\theta \tag{13}$$

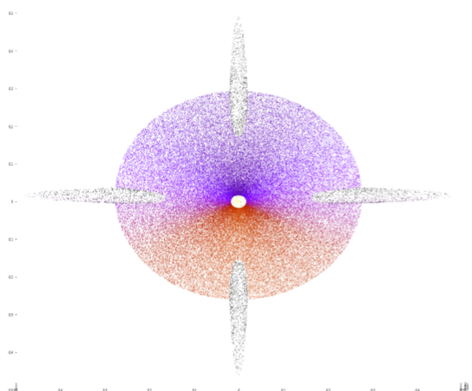
The resulting geometry is shown in figure 3.

2.2 Center of pressure

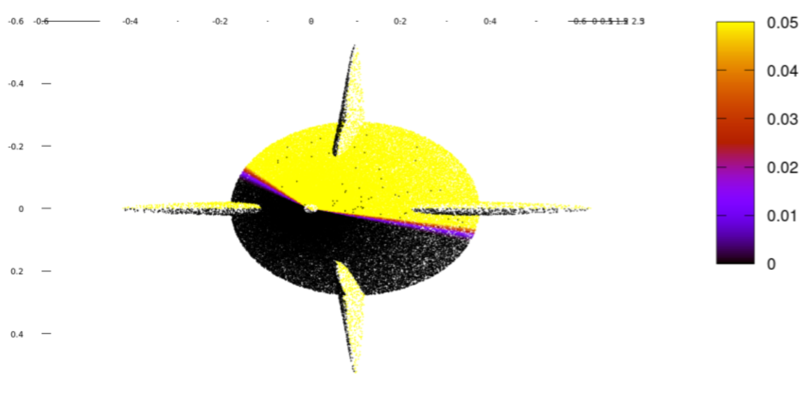
To find the position vector of the center of pressure of the craft (useful later for validation), we want to perform the integral:



(a) Side view at 8° angle of attack



(b) Front view at 8° angle of attack



(c) Angled view at more extreme angle of attack, such that 3D structure of flaps is visible

Figure 3: SWERVE visualizations, colored by coefficient of pressure at each point on surface

$$\vec{r}_{cp} = \frac{\int \vec{r}(\vec{p}(x,y)\cdot\hat{n})n_z\,dA}{\int (\vec{p}(x,y)\cdot\hat{n})n_z\,dA} \tag{14}$$

Since our vehicle is axisymmetric, this should result in a center of pressure along the x-axis. I implemented this (results appear as `CP_x` and `CP_y` in output files) but it doesn't seem to be very accurate for cones with small half-angle at small angles of attack, I suspect because the integrated force in the body z-direction over area in those cases is so small, so a small random deviation can cause a significant error in the results. (It gives expected analytical results for "easier" configurations, e.g. a sphere.) Thus I won't actually draw conclusions from its output, but I also didn't want to un-implement it in case someone thinks of a way to make it better in the future, so here it is.

2.3 Pitching coefficient

The moment coefficients about each axis are calculated as:

$$\vec{C}_M = \frac{1}{\frac{1}{2}\rho V^2 Sc}\int_S (\vec{r}-\vec{q})\times(p\vec{n})\,dA \tag{15}$$

where $\vec{q}$ is the location of the center of gravity, and $\vec{r}$ is the position vector of a certain point on the surface.

By default, `MC-NEW` outputs C_M at the angle of attack declared in the `.flap` input file; to find $C_{M_\alpha} = \frac{\partial C_M}{\partial \alpha}$, we need to run many Monte Carlo simulations at different angles of attack and fit a line to the resulting C_M vs. α curve. The slope of this line is the desired derivative.

In practice, this is done by running `run.sh` in the `Examples/SwerveTemplate` directory (or a copy of it). To run properly, these scripts require the presence of `MC-NEW`, an input `.flap` file, a directory to store run results in, and several `.py` setup scripts inside that directory. The easiest thing to do is probably to keep the original `SwerveTemplate` directory untouched and just make a copy of the whole directory when you want to test something new.

`setup.sh` takes 6 parameters: `i`, the name of the input file to be used as a template; `d`, the name of the directory to store runs in; `p`, the name of the parameter to be varied; `s`, the initial value of the parameter; `e`, the final value of the parameter; and `n`, the total number of steps to take between these values. `p` is implemented by searching for the line in the `.flap` file that contains the string passed to `p`, so writing the entire name is not necessary, but writing enough such that there is exactly one line containing that substring is necessary.

As an example, the following sequence of commands would result in a graph of C_M vs α for a series of runs with all parameters the same as `input.flap` except for the pitch angle of attack which varies from -5.0 to 5.0. A line will be automatically fitted to the curve, and the slope of this line, also given, is C_{M_α}. `cmy_alpha.plot` comes with the code, but if you want to investigate changes in other parameter, you can read from `runs/all_aero_coefs.dat`, in which all results from the iteration will be recorded.

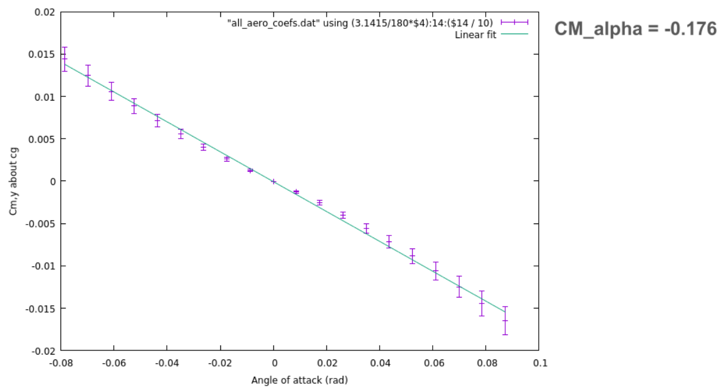


Figure 4: C_M for different small angles of attack.

```
> bash run.sh -i input.flap -d runs -p pitch -s -5.0 -e 5.0 -n 21
> cd runs
> gnuplot
> load "cmy_alpha.plot"
```

Note that `bash` only understands integer inputs - it would not correctly interpret 5.5 as an argument, for example. The script that actually modifies the input files is a Python script and thus deals with the intermediate decimal values between the start and end values fine, but you just can't start or end the iteration on a non-integer. I don't actually know what `bash` does if you write something like 5.5, but it won't throw an error; instead it'll just mess up your range of values.

The results around $\alpha = 0$ are shown in figure 4. This corresponds to $C_{M_\alpha} = -0.176$.

2.4 Calculation of pitch damping coefficient

The time constant we are interested in depends solely on C_{M_q}, the pitch damping coefficient. Most studies find this coefficient by inducing forced oscillations or other predetermined patterns of motion in a CFD code, since the derivative with respect to q is an inherently dynamic property. Our Monte Carlo approach approximates this via Newtonian impact theory. A brief derivation of the expression I use for the pitch damping coefficient is given here; see [2] for the full derivation.

Suppose the vehicle's center of mass travels at a linear velocity $\vec{v}_c$ and the vehicle has some angular velocity $\vec{\omega} = (p, q, r)$ around its COM. Then the velocity relative to the air of some surface element at position $\vec{r}$ is:

$$\vec{v}(\vec{r}) = \vec{v}_c + \vec{\omega} \times \vec{r} \tag{16}$$

The Newtonian pressure coefficient at $\vec{r}$, which depends on the velocity of air $\vec{v}_\infty(\vec{r})$ with respect to the surface at that point, becomes:

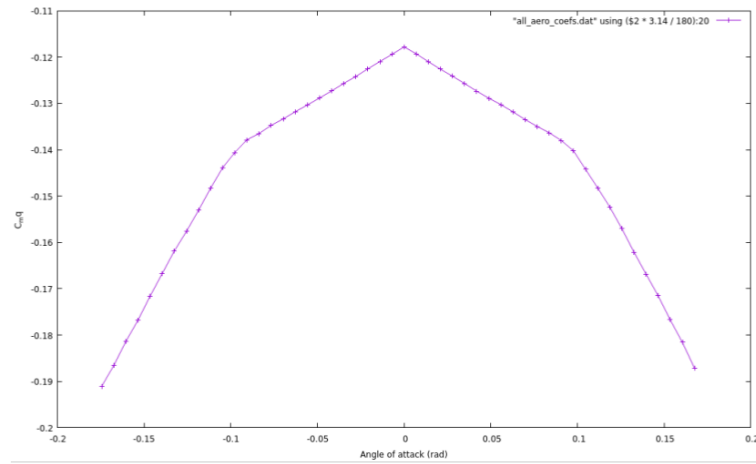


Figure 5: C_{M_q} vs α at small angles of attack; here, $x_{cg}/l = 0.64$

$$c_p(\vec{r}) = 2\,(\hat{v}_\infty(\vec{r}) \cdot \vec{n})^2 \approx 2\,(\hat{v}_\infty(0) \cdot \vec{n} - \frac{(\vec{\omega} \times \vec{r}) \cdot \vec{n}}{v_\infty(0)})^2 \tag{17}$$

For $r\omega << v_\infty$, the change in c_p due to ω is then expressed as:

$$\delta c_p(\vec{r}) = -4\,\frac{\hat{v}_\infty(0) \cdot \vec{n}\,(\vec{r} \times \vec{n}) \cdot \vec{\omega}}{v_\infty(0)} \tag{18}$$

Then with an infinitesimal moment element expressed as $d\vec{M} = qc_p(\vec{r} \times \vec{n}) \cdot dS$,

$$C_{M_q} = -\frac{4}{S} \int_{\hat{v}_\infty(0) \cdot \vec{n} \geq 0} \frac{\hat{v}_\infty(0) \cdot \vec{n}\,(\vec{r} \times \vec{n})^2}{v_\infty(0) \cdot c^2} dS \tag{19}$$

The results are shown in figure 5.

2.5 Validation of aerodynamic coefficients

The various coefficients can be compared to existing values in the literature and to analytical calculations to assess their accuracy.

2.5.1 Comparison to analytical expressions

To check our implementation of the C_N, C_M, and C_{M_q} calculations, we can compare to analytical expressions for those values for a right circular cone. A table with our outputs for a right circular cone compared to Gallais's values [2] is shown below. For regimes in which Gallais's expressions hold (specifically, his C_{M_q} expression about the center of gravity holds only in the limit of a very sharp cone), the maximum error is 3%, in C_{M_q}.

α	θ	C_M (G)	C_M (M)	C_{M_q} (G)	C_{M_q} (M)	C_N (G)	C_N (M)
0°	5.25°	0	0	-0.132	-0.136	0	0
5°	12°	-0.116	-0.115			0.166	0.165
10°	12°	-0.228	-0.227			0.327	0.326

Table 1: Comparisons between our results and analytical results from Gallais. α is the angle of attack and θ is the half-angle of the cone. Those from Gallais are labeled (G); those from MC-NEW are labeled (M). The C_M values are measured about the apex of the cone, because those are the expressions Gallais gave, whereas everywhere else in this project I use C_M values about the center of gravity.

2.5.2 Comparison to SWERVE PNS calculations

SWERVE underwent several flight tests between 1979 and 1985. Though the numerical results from these tests were not released, the results from a preliminary PNS (parabolized Navier-Stokes) analysis were. The results, and comparison to results from our code, are shown below. (The results alone are shown in more detail in figure 10.

α	C_N (PNS)	C_N (MC-NEW)	C_M (PNS)	C_M (MC-NEW)
10°	0.476	0.401	-0.00529	-0.0223
2°	0.093	0.067	-0.00291	-0.00217

Table 2: Comparison to Sandia PNS results for $x_{cg}/l = 0.64$. [3]

Note that the case with $\alpha = 10°$ was simulated in PNS at Mach 12 with a turbulent boundary layer, which we have not considered in the Monte Carlo simulation. Our results are generally of the correct order of magnitude, but their agreement with the PNS numbers is pretty low overall. There are errors of 20-30% in C_N. Given our uncertainty in the dimensions and especially mass distribution of the vehicle, the disagreement in C_M, which is measured about the center of gravity, is not unexpected. Finally, the study states that the PNS values only predicted flight values to within 20-30% themselves, and had worse accuracy than this for some periods of flight; thus, agreement of our results with PNS values beyond 20-30% may not be meaningful.

3 Finding time constants

With coefficients calculated, finding the time constant at each timestep requires the vehicle velocity and atmospheric density; this entails numerically integrating the vehicle's acceleration over time on its trajectory through the atmosphere. We assume the vehicle undergoes free-fall in a perfectly vertical trajectory (nose pointing straight at ground; area subject to drag equal to its cross-sectional area). With gravity and drag as the only two forces acting on the vehicle, and using an exponential model for atmospheric density, we can find its velocity at each timestep and calculate the corresponding time constant. To calculate drag we take $C_D = 0.2$ throughout the flight.

3.1 Moment of inertia

The expression for time constant in equation 8 depends on the moment of inertia of the body about the pitch axis. We approximate the body as a cone for this step, assuming the mass of the flaps and the size of the spherical nose cap are negligible.

However, the vehicle is not stable ($C_{M_\alpha} > 0$) for a uniform mass distribution where the center of gravity of the cone lies 3/4 of the body length towards the back, measured from the nose. The center of gravity must be moved forward to make the vehicle stable. The location of the center of gravity can be specified in both an input file for `MC-NEW` and in the first section (which specifies vehicle characteristics) of `rv_dynamics.py`.

In `rv_dynamics.py`, this manifests in the moment of inertia calculation by overlaying a 1D rod of a calculated mass along the cone's axis of symmetry such that the overall center of gravity lies at the specified position. This only works for conical vehicles so far (i.e. if you model a vehicle with some cylindrical extent, as the code allows for, it will fail; at the moment this can only be done by allowing the code to calculate the COM itself for a uniformly dense body). It is assumed that the center of gravity cannot lie more than halfway up the cone, i.e. it cannot be closer to the tip than the base. This is a very reasonable assumption as far as reentry vehicles are concerned.

3.2 Outputs

Currently, by default, the code outputs:

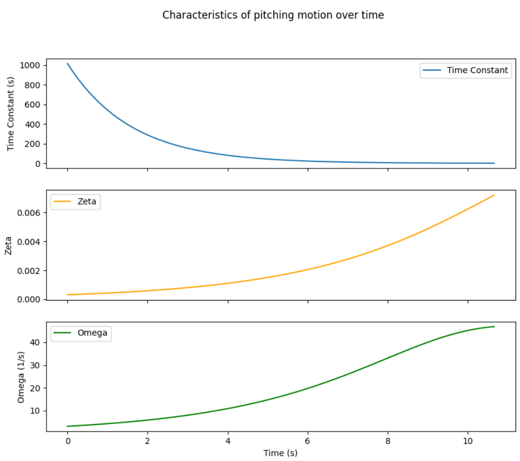
- A plot of ζ vs. time.
- A plot of ω_n vs. time.
- A plot of time constant vs. time.
- A plot of time constants left vs. altitude.

This last plot takes into account the fact that the time constant is changing over time. In other words, it calculates:

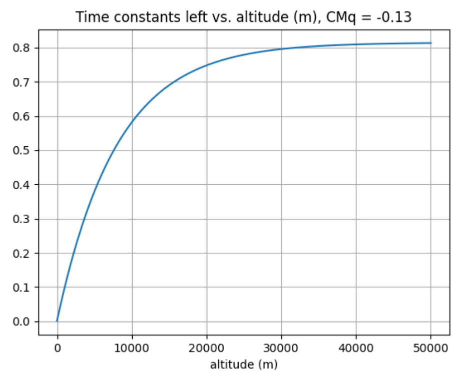
$$n_{\text{remaining}}(t) = \int_t^{t_f} \frac{dt'}{\tau(t')} \tag{20}$$

where t_f is the total time of flight.

The values of altitude and velocity are also recorded at each point in the flight, since they are needed to calculate the time constant, and can be plotted easily if desired.

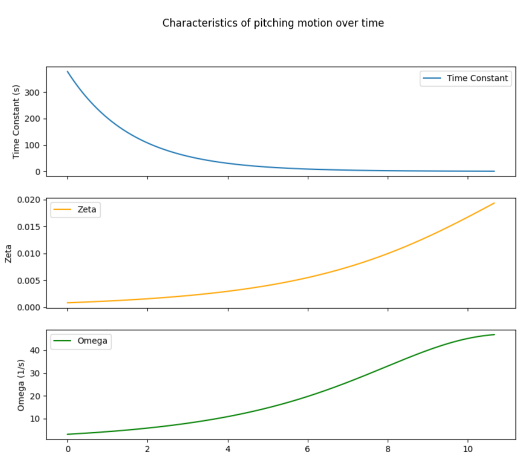


(a) τ, ζ, and ω_n over time

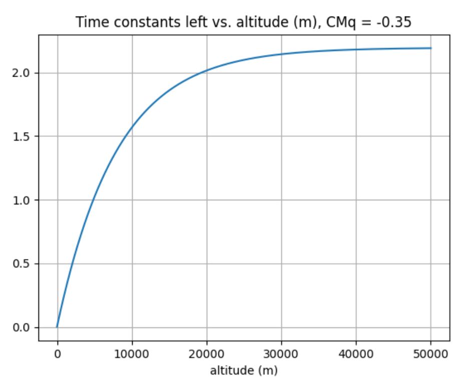


(b) Number of time constants left over time

Figure 6: Results for $C_{M_q} = -0.13$, corresponding to approximately zero angle of attack

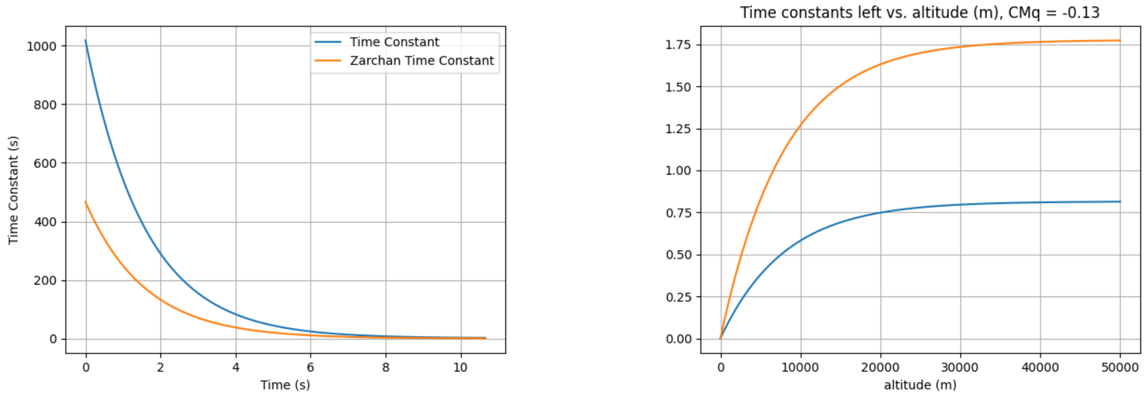


(a) τ, ζ, and ω_n over time



(b) Number of time constants left over time

Figure 7: Results for $C_{M_q} = -0.35$, corresponding to angle of attack of 20°, which is around the maximum achievable α (and hence maneuverability)



(a) Time constant over time

(b) Time constants left over time

Figure 8: At $\alpha = 0°$, Zarchan's results for the time constant (orange) vs our results (blue)

3.3 Comparison to Zarchan

Zarchan's tactical missile guidance textbook also contains a treatment of the time constant. In Zarchan's book, this depends on C_{M_α}, $C_{M_{\delta_e}}$, C_{N_α}, and $C_{N_{\delta_e}}$. Zarchan's method differs from ours in terms of the approximations he makes, including some which may reflect an assumption that the missile is basically a long cylinder. His time constant is different from ours by approximately a factor of 2, apparent in figure 8, wherein the time constants resulting from the method in this report are compared to the time constants resulting from applying Zarchan's methodology to the same geometry.

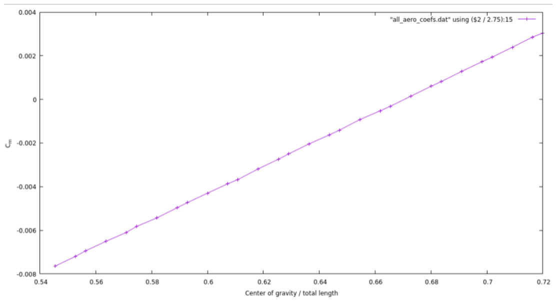
4 Sources of uncertainty

Aside from the variance due to having a limited number of sample points in our Monte Carlo simulation, we have two sources of uncertainty. Some quantities are unable to be simulated in the current Monte Carlo framework: the Reynolds number (and thus turbulent/laminar behavior) of the air, for instance, or deflection of control surfaces. On the other hand, some quantities are implemented in the simulation, but we are uncertain about them for reasons of lack of detailed information about the craft. These include the location of the center of gravity, the average density of the craft, and the detailed mass distribution (which affects moment of inertia).

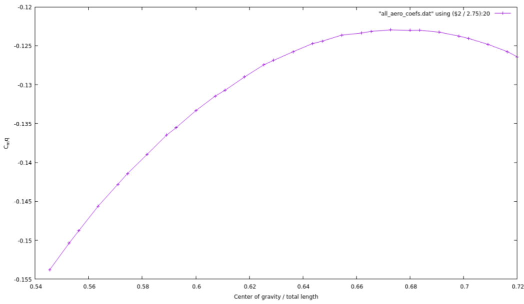
4.1 Center of gravity

Based on equation 15, one expects C_M to have a linear dependence on $\vec{q}$. Indeed, that is what we see in figure 9.

As for the pitch damping coefficient, we would expect a quadratic dependence on $\vec{r} - \vec{q}$, based on 19. The parabola displayed in figure 9 peaks (attains its least negative value) when the center of gravity



(a) C_M vs $\frac{x_{cg}}{l}$



(b) C_{M_q} vs $\frac{x_{cg}}{l}$

Figure 9: At an angle of attack of 2°, simulated moment coefficients vs. distance of center of gravity to the nose normalized by total length

is closest to the simulated center of pressure (in this case, the simulated center of pressure lies at $x_{cp} = 0.67 \cdot l$). This makes sense, since the motion of the craft should be damped more quickly (corresponding to a smaller time constant, or more negative C_{M_q}) when the center of pressure is farther from the center of gravity, since the air then exerts a greater torque on the vehicle. So as long as the center of gravity is reasonably close to the center of pressure, the time constant of the vehicle, which only depends on C_{M_q}, should not vary much with the *precise* location of the center of gravity.

4.2 Density

Since the time constant depends linearly on the vehicle's moment of inertia, its total mass may be important. Unfortunately we don't really know what it is. However, we can assume that the vehicle should be more dense than water (i.e. $\rho > 1000$ kg/m^3), and set an upper bound on reasonable density for a reentry vehicle of 2000 kg/m^3 (which is the value I use throughout this project). This only changes the time constant by a factor of two. There would need to be a much larger difference in mass to significantly affect our conclusions. We can also conclude from this that, for a fixed center of gravity, the detailed mass distribution probably will not differ enough to be *qualitatively* important.

4.3 Wing thickness

The thickness of the wing is set at 5cm (corresponding to a semiminor axis $b = 0.025$ in the code), but we should check to see whether a small difference in wing thickness affects the results. We'll look at the relevant moment coefficients (C_{M_α} and C_{M_q}) for a 1cm thick wing and a 10cm thick wing, at $\alpha = 0°$ and $\alpha = 10°$. This assumes $x_{cg}/l = 0.6$.

From the results, shown in tables 3-5, it seems that as long as the wings are thin enough, their exact thickness does not matter too much. Since I assume that the wings are very thin throughout this project (including e.g. making small-angle approximations in the calculations of normal vectors and differential area on the wings), their thickness relative to the base radius of the vehicle should

Table 1. PNS results for Mach 12 with turbulent boundary layer.

Parabolized Navier-Stokes results (Mach 12)				
α, deg	δ, deg	C_N	X_{CP}/LB	C_m
10	0	0.476	0.701	-0.00529
10	5	0.452	0.687	0.00114
10	10	0.437	0.6878	0.00514
$M = 12$			$\delta_{\mathrm{TRIM}} = 4.11°$	
Turbulent			$T_W = 4000\,°\mathrm{R}$	

Table 2. PNS results for Mach 8 with laminar boundary layer.

Parabolized Navier-Stokes results (Mach 8)				
α, deg	δ, deg	C_N	X_{CP}/LB	C_m
2	0	0.0930	0.721	-0.00291
2	2	0.0861	0.702	-0.00107
2	5	0.0756	0.667	0.00175
$M = 8$			$\delta_{\mathrm{TRIM}} = 3.1°$	
Laminar			$T_W = 2000\,°\mathrm{R}$	

Figure 10: SWERVE PNS simulation showing variance in C_M and C_N with control surface deflection in various conditions. [3]

be kept relatively small anyway, about 10% or less.

Table 3: 1cm thick

α	$C_{\mathrm{M}\alpha}$	$C_{\mathrm{M}q}$
0°	-0.18	-0.14
10°	-0.36	-0.23

Table 4: 5cm thick

α	$C_{\mathrm{M}\alpha}$	$C_{\mathrm{M}q}$
0°	-0.18	-0.13
10°	-0.33	-0.22

Table 5: 10cm thick

α	$C_{\mathrm{M}\alpha}$	$C_{\mathrm{M}q}$
0°	-0.13	-0.12
10°	-0.28	-0.20

5 Limitations

These calculations do not account for the effects of turbulence, ablation of the heat shield, or high temperatures during reentry. While turbulent fluid effects may not become important until the last part of the flight, I haven't found any information about how much of an effect this has on the final CEP.

There is also no explicit consideration of a control surface (i.e. elevons to maneuver the craft). In the case of SWERVE, control surface deflection significantly affects C_M at least (see figure 10), though its effects on the derivatives C_{M_α} and C_{M_q} are unknown. If these derivatives remain constant, then one could in theory simulate a controlled craft using the SWERVE control surface deflection data for δ_e and approximating $C_{M_{\delta_e}}$ by a finite difference method. This may at least give an order-of-magnitude estimate for how the deflection of a control surface could affect flight.

Appendix A Supplementary Data

These runs all assume $x_{cg}/l = 0.64$. Also note that C_D is calculated with respect to the base area of the conical body in all runs, i.e. the cross-sectional area considered is constant even as the angle of attack changes. Some small inconsistencies may occur between the data listed here and the data listed throughout the report because these data use pressure coefficients which are modified by multiplying by a correction factor that considers the pressure behind the shock wave for vehicles traveling above the speed of sound. A Mach number of 12 is assumed for all data points that follow.

α (°)	C_D	C_L	C_N	C_m	C_{m_q}
0.0	0.0180	0.000	0.000	0.0000	-0.118
0.2	0.0181	0.006	0.006	-0.0002	-0.119
0.4	0.0182	0.012	0.012	-0.0003	-0.120
0.6	0.0183	0.018	0.018	-0.0005	-0.121
0.8	0.0185	0.024	0.024	-0.0008	-0.121
1.0	0.0188	0.030	0.030	-0.0010	-0.122
1.2	0.0192	0.036	0.036	-0.0012	-0.123
1.4	0.0196	0.042	0.043	-0.0014	-0.124
1.6	0.0201	0.048	0.049	-0.0016	-0.125
1.8	0.0205	0.055	0.055	-0.0018	-0.125
2.0	0.0212	0.061	0.061	-0.0021	-0.126
2.2	0.0219	0.067	0.068	-0.0023	-0.127
2.4	0.0226	0.073	0.074	-0.0026	-0.128
2.6	0.0234	0.080	0.081	-0.0029	-0.129
2.8	0.0243	0.086	0.087	-0.0032	-0.129
3.0	0.0252	0.092	0.094	-0.0035	-0.130
3.2	0.0262	0.099	0.100	-0.0038	-0.131
3.4	0.0273	0.105	0.107	-0.0041	-0.131
3.6	0.0284	0.112	0.113	-0.0043	-0.132
3.8	0.0296	0.118	0.120	-0.0047	-0.133
4.0	0.0309	0.125	0.126	-0.0050	-0.134
4.2	0.0322	0.131	0.133	-0.0054	-0.135
4.4	0.0337	0.138	0.140	-0.0057	-0.136
4.6	0.0351	0.144	0.147	-0.0061	-0.136
4.8	0.0367	0.151	0.153	-0.0065	-0.137
5.0	0.0383	0.157	0.160	-0.0069	-0.138
5.2	0.0399	0.164	0.167	-0.0072	-0.138
5.4	0.0417	0.171	0.174	-0.0076	-0.140
5.6	0.0436	0.177	0.181	-0.0079	-0.141
5.8	0.0455	0.184	0.188	-0.0084	-0.143
6.0	0.0474	0.191	0.195	-0.0088	-0.145

Continued on next page

α (°)	C_D	C_L	C_N	C_m	C_{m_q}
6.2	0.0494	0.198	0.202	-0.0093	-0.147
6.4	0.0515	0.205	0.209	-0.0098	-0.149
6.6	0.0538	0.212	0.217	-0.0101	-0.151
6.8	0.0559	0.219	0.224	-0.0106	-0.153
7.0	0.0583	0.226	0.232	-0.0110	-0.155
7.2	0.0607	0.234	0.239	-0.0115	-0.158
7.4	0.0632	0.241	0.247	-0.0120	-0.160
7.6	0.0657	0.249	0.255	-0.0126	-0.163
7.8	0.0684	0.256	0.263	-0.0130	-0.165
8.0	0.0711	0.264	0.271	-0.0135	-0.167
8.2	0.0740	0.272	0.280	-0.0140	-0.170
8.4	0.0769	0.280	0.288	-0.0146	-0.172
8.6	0.0797	0.288	0.296	-0.0152	-0.175
8.8	0.0828	0.296	0.305	-0.0157	-0.177
9.0	0.0861	0.304	0.314	-0.0164	-0.180
9.2	0.0893	0.313	0.323	-0.0169	-0.182
9.4	0.0925	0.321	0.331	-0.0175	-0.185
9.6	0.0960	0.329	0.341	-0.0181	-0.187
9.8	0.0994	0.338	0.350	-0.0187	-0.190
10.0	0.1031	0.347	0.360	-0.0194	-0.193

References

[1] M. A. Furudate. "MC-New: A Program to Calculate Newtonian Aerodynamic Coefficients Based on Monte-Carlo Integration". In: *Aerospace* 9 (2022), p. 330. DOI: `10.3390/aerospace9060330`. URL: `https://doi.org/10.3390/aerospace9060330`.

[2] Patrick Gallais. "Atmospheric re-entry vehicle mechanics". en. In: Berlin: Springer, 2007. Chap. 4.3, pp. 40–52. ISBN: 978-3-540-73646-2.

[3] Kenneth W. Iliff and Mary F. Shafer. *A Comparison of Hypersonic Flight and Prediction Results*. Tech. rep. P.O. Box 273. NASA Dryden Flight Research Facility. Edwards, California: NASA Dryden Flight Research Facility, 1993.

[4] Marcus S. Murbach, David L. Keese, and Jack Farmer. "AEOLUS: A System Concept for Precise Delivery of Mars Scientific Instrumentation". In: *Proceedings of the 11th AIAA/USU Conference on Small Satellites*. AIAA/USU. 1997.

[5] Frank J Regan and Satya M Anandakrishnan. "Angular motion during re-entry". In: *Dynamics of Atmospheric Re-Entry*. American Institute of Aeronautics and Astronautics, 1993, p. 409.