AIAA Atmospheric Flight Mechanics Conference
13 - 16 August 2012, Minneapolis, Minnesota

AIAA 2012-4958

Continuous Differentiation of Complex Systems Applied to a Hypersonic Vehicle

Derek J. Dalle[*], James F. Driscoll[†]
University of Michigan, Ann Arbor, MI 48109

DOI: 10.2415/6.2012-4958

An implementation of a three-dimensional model for an air-breathing hypersonic vehicle using algorithmic differentiation is presented. Motivation for this work is given in terms of difficulties encountered while calculating the linearized flight dynamics of another hypersonic vehicle model. The new model automatically calculates derivatives with respect to flight condition variables, control variables, model parameters, and design variables. The differentiation approach used here is unique in that it calculates both left- and right-hand derivatives at first-order discontinuities. Other approaches including complex step and finite difference are compared with the present implementation. Difficulties encountered in the derivation include handling of `if` statements, iterative solution of systems, and sorting. Hypersonic flight for a simple climbing and accelerating trajectory is considered to demonstrate the use of the model, and the effect of ballast mass on the short-period and Dutch-roll modes is also shown.

Nomenclature

a	=	local speed of sound
$\mathbf{C}$	=	transformation matrix
e	=	eccentricity
$F, \mathbf{F}$	=	force or specific force
f	=	mass capture fraction
$\mathbf{f}$	=	vector equations of motion function
$\mathbf{g}$	=	gravity vector
H	=	height
h	=	altitude
$\mathbf{I}$	=	inertia tensor
J	=	Jacobian
L	=	length or geodetic latitude
M	=	Mach number
$\mathbf{M}$	=	moment (torque) or specific moment
m	=	number of external inlet shocks
$\dot{m}$	=	mass flow rate
n	=	number of internal inlet shocks
$\mathbf{n}$	=	normal vector
$\hat{\mathbf{n}}$	=	unit normal vector
P	=	roll rate
p	=	pressure
Q	=	pitch rate
q	=	dynamic pressure
R	=	yaw rate
$\mathbf{r}$	=	position vector
T	=	temperature
$\mathbf{u}$	=	vector of control variables
$V, v, \mathbf{v}$	=	velocity
w	=	dimensionless width
x, y, z	=	physical coordinates
$\mathbf{x}$	=	vector of state variables
α	=	angle of attack
β	=	sideslip angle
γ	=	ratio of specific heats or flight path angle
δ	=	deflection angle or control input
ε	=	pressure ratio
ζ, η, ξ	=	placeholder variables
θ	=	pitch angle or angle of flow
λ	=	length fraction
ρ	=	density
σ	=	velocity heading angle
ϕ	=	roll angle
ψ	=	yaw angle
$\omega, \boldsymbol{\omega}$	=	angular velocity

Subscripts/superscripts

$+$	=	right-hand derivative
$-$	=	left-hand derivative
$*$	=	reference value
0	=	stagnation value
$1, 2, \ldots$	=	station number
b	=	body frame
CE	=	collective (average) elevon value
CR	=	collective (average) rudder value
DE	=	differential (left minus right) elevon value
e	=	Earth-centered, Earth-fixed frame
ER	=	equivalence ratio
i	=	station index or intertial frame
N, E, D	=	north, east, down components
n	=	navigation frame

I. Introduction

Sensitivity analysis has many applications in engineering, for example in uncertainty analysis, design optimization, and inverse problems, but it has an additional level of importance to nonlinear systems and in particular flight dynamics. A very important and useful analysis of a complicated nonlinear system is to linearize about a certain condition [1]. Analysis of the resulting linearized dynamics gives insight into the stability and control characteristics of the system, and this is almost always a starting point for more sophisticated analysis. As a result, studying nonlinear systems often requires calculating derivatives even if optimization or uncertainty analysis is not considered.

The simplest way to calculate these derivatives is to use finite differences. This method is extremely easy to implement, but finite differences have well-known problems with accuracy and efficiency [2]. We have developed a tip-to-tail model of an X-43-like hypersonic vehicle [3–6] that has been useful in calculating vehicle performance but has had problems with estimating sensitivities [6–8]. In particular, that model, called MASIV (Michigan/AFRL Scramjet In Vehicle), includes complex effects such as shock interactions, fuel-air mixing, and finite-rate chemistry. While it has been very useful to study trim conditions, the model has given results for the stability of the short-period mode that are sometimes difficult to interpret. The problems arise from the finite-difference approximations used to calculate the sensitivities of the equations of motion to state variables and control variables.

Several strategies are available to address this problem. A particularly promising approach that has already been applied to sophisticated models is to use a complex-step approximation to the derivatives [2].

[*] Graduate Research Assistant, Department of Aerospace Engineering; `dalle@umich.edu`. Student Member AIAA

[†] Professor, Department of Aerospace Engineering; `jamesfd@umich.edu`. Fellow AIAA.

This approach is similar in use to the finite-difference step approximation in that each derivative (i.e. the derivative with respect to each variable) requires a separate evaluation of the function. However, the complex-step approximation is both more robust and more accurate. It is also fairly easy to implement assuming that every intermediate step of the model contains only real values when the inputs are real.

A second approach is called algorithmic differentiation (or sometimes automatic, manual, or continuous differentiation) [9]. Algorithmic differentiation produces a program that calculates both the value of a given output and its derivative each time it is evaluated. This works even for complex systems by calculating the derivatives at each intermediate step of the computation and repeated application of the chain rule. The technique of algorithmic differentiation has been applied successfully in a wide variety of applications [10, 11]. This style of differentiation seems to have very few insurmountable limitations and was selected for this study as a somewhat safe option.

Several models for air-breathing hypersonic vehicles with the level of complexity targeted for this study have already been proposed. Chavez and Schmidt [12], Frendreis and Cesnik [13], and Bolender and Doman [14], have created models to predict an unstable short-period mode for single flight conditions. Interestingly, the Chavez and Schmidt model contains a type of automatic differentiation as discussed in [12], but with several approximations and only with respect to the variables needed for the stability analysis. The model used here is very similar to [14] but applied to a three-dimensional vehicle.

This vehicle model is characterized by a series of equations (some of which require solving equations implicitly). In this discussion, a collection of “basis variables” are selected, and derivatives are taken with respect to these variables using repeated applications of the chain rule. In addition to the flight condition sensitivities and control variable sensitivities required for the stability analysis, the model also calculates derivatives with respect to a large set of vehicle design parameters and a smaller number of model parameters (for example the radius of the Earth). A small addition that was made to the model is that left-hand and right-hand derivatives are calculated separately. As a result, derivatives are handled correctly for functions such as `abs`, `max`, and `min`.

The bulk of this discussion focuses on the creation of this model with special attention to aspects of the code that are particularly difficult to differentiate. A brief description of the Matlab implementation is given, and the remaining description is split into sections of vehicle design, force and moment calculation, and equations of motion. A brief trajectory analysis provides a demonstration of the model in use. It consists of a practical climbing and accelerating trajectory with a constant dynamic pressure of one atmosphere and a constant acceleration of 2 m/s^2. The poles of the short-period and Dutch-roll modes are plotted along this trajectory, and the modes are also calculated for a range of ballast masses, which alter the center of gravity of the vehicle.

II. Vehicle Model

The vehicle used in this model is shown in Fig. 2, and it is roughly a scaled-up version of an X-43. For simplicity the sides of the vehicle are kept vertical. The design parameters include vehicle length, design Mach number, and about 38 others so that the vehicle can be completely constructed from the design parameters. Calculation of forces and moments on the vehicle is split into four parts: inlet, combustor, nozzle, and exterior surfaces.

The entire model has been implemented in Matlab. A proven and elegant method of algorithmic differentiation is to replace each numerical program variable with an object that stores both the values and derivatives [15, 16]. Then each arithmetic operator is overloaded with a function that calculates values and applies the chain rule. Since Matlab is an interpreted language, using complex data structures or objects can create efficiency problems due to the amount of overhead it creates. In particular, creation of new object types is particularly slow. In this implementation, the built-in variable type `struct` is used. This provides somewhat of a middle ground as it is possible to store both values and derivatives in the same variable, but it does not provide an easy way to directly overload operators such as addition and subtraction.

The coordinate system for this vehicle is the standard body-fixed frame for aerospace vehicles. The x-axis points forward with respect to the vehicle, the y-axis points toward the pilot’s right, and the z-axis points downward. The origin is chosen to be the middle of the leading edge.

A. Programming Environment

The model was implemented in Matlab using a variable type called `struct`. Instead of overloading operators, the chain rule is applied manually after each manipulation of variables. The following snippet of code demonstrates an example variable `a` that is considered to be a function of `x`. First we declare the variable `x`. Its derivative is 1 because `x` is also the basis variable.

```
x = struct(’v’, 3, ’L1’, 1, ’R1’);
```

Now we have stored a value of 3 in `x` and a first derivative of 1 (both the left-hand and right-hand derivatives are 1). A function that squares the value of `x` and calculates the derivative could be written like the following.

```
a = f(x);

function a = f(x)
% Function value
a.v = x.v ^ 2;
% Derivative
a.L1 = 2 * x.v * x.L1;
a.R1 = 2 * x.v * x.R1;
```

As a result, `a` will have the following value.

```
a =
     v: 9
    L1: 18
    R1: 18
```

The approach used here, which is perhaps less elegant but also has the property that exceptions can be handled easily, is to apply the chain rule manually. The previous section of code changes to the following.

```
a.v = f(x.v)
a = chain_rule(x, 2*x.v)

function A = f(X)
% Function value only
A = X ^ 2;
```

The `2*x.v` in `chain_rule(x, 2*x.v)` comes manually from calculating $\frac{\partial f}{\partial x}$. In the present model, the derivatives are actually vectors that keep track of the derivatives of the value with respect to all the basis variables.

B. MASIV Vehicle

The vehicle used for MASIV [4, 6] is shown in Fig. 1 while Fig. 2 shows the vehicle used for the present model. The two vehicle models are not identical, but the similarities are obvious. The flight dynamics of the two vehicles should not be extremely different.

C. Vehicle Design

Most of the vehicle design, including differentiation, is a straightforward process. For example, the length of the inlet is specified as a fraction of the total length of the vehicle, and thus involves two design variables.

$$L_{(inlet)} = \lambda_{(inlet)} L_{(vehicle)} \tag{1}$$

The Jacobian of the inlet length can be found quite simply using the chain rule.

$$J(L_{(inlet)}) = \lambda_{(inlet)} J(L_{(vehicle)}) + L_{(vehicle)} J(\lambda_{(inlet)}) \tag{2}$$

The length of the vehicle is distributed using this method among the inlet, isolator, combustor, and nozzle. This makes four design variables: $L_{(vehicle)}$, $\lambda_{(inlet)}$, $\lambda_{(isolator)}$, $\lambda_{(combustor)}$. The missing value, $\lambda_{(nozzle)}$ is chosen so that the λ values add up to 1.

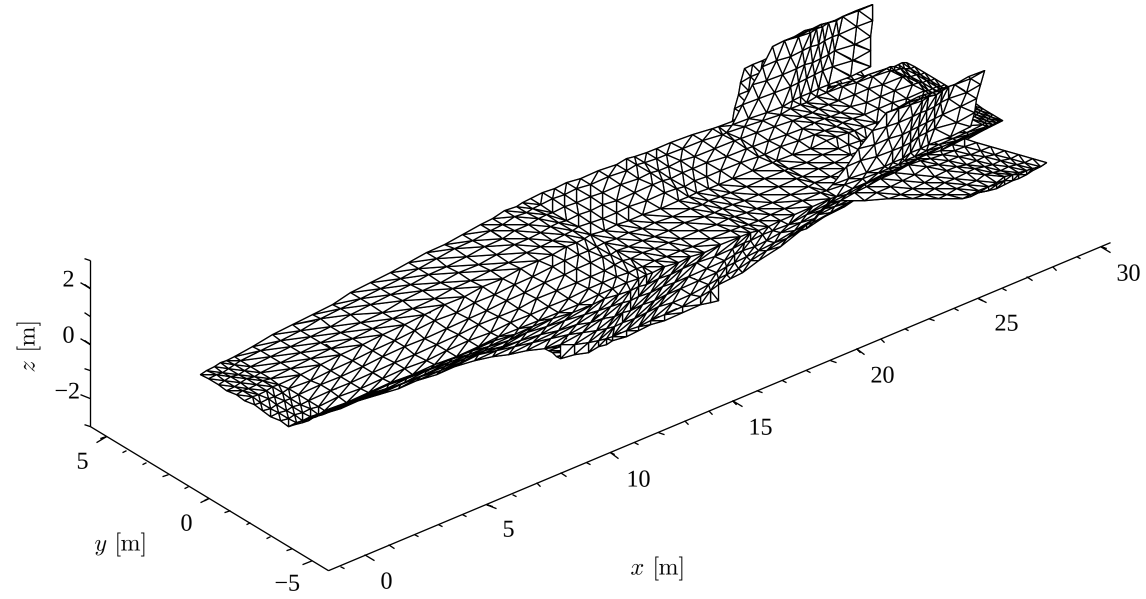


Figure 1. Isometric view of the MAX-1 vehicle used with MASIV.

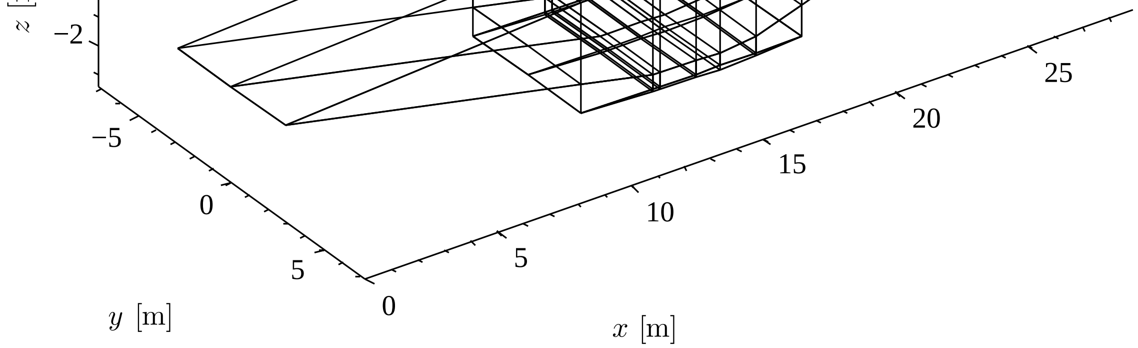


Figure 2. Vehicle used for present fundamental model.

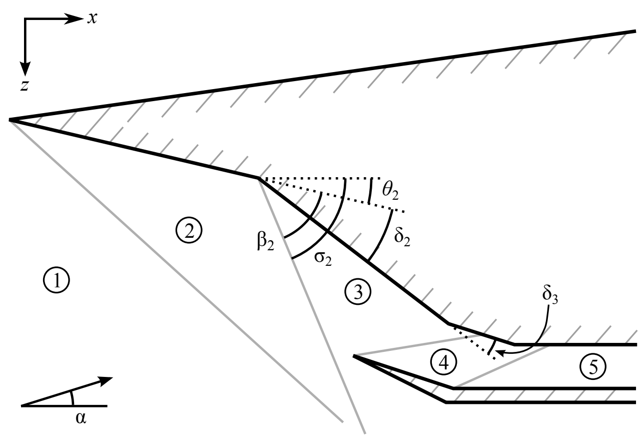


Figure 3. Definitions for inlet design methodology.

Similarly, the width of the vehicle is taken from a dimensionless parameter $w_{(vehicle)}$ so that the entire vehicle may be scaled up or down using only $L_{(vehicle)}$. The sides of the vehicle are taken to be completely flat, so that for any given vertex i, the y-coordinate is

$$y_i = \pm\frac{1}{2}w_{(vehicle)}L_{(vehicle)} \tag{3}$$

D. Inlet design

The full inlet design process from [17] is used here except that variable geometry is not considered. Despite the fact the inlet model used here is much simpler than the MASIV inlet model [3], a well-designed inlet has been shown to have the property that simple models predict their performance fairly accurately over a limited range of flight conditions [17].

The process begins by picking a certain flight condition–more specifically a Mach number and angle of attack–at which the inlet's performance should be optimized. Performance at other flight conditions will also be taken into account in the design process, but we start with a single flight condition using the design variables $M_{(design)}$ and $\alpha_{(design)}$. The inlet will consist of a sequence of m external shocks and n internal shocks. Figure 3 shows a visualization of these shocks and the regions they create for an inlet with $m = 2$ and $n = 2$. To optimize the pressure recovery factor (a measure of entropy generation) at the design flight condition, the external shocks all have the compression ratio $\varepsilon_{(ext)}$, and the internal shocks all have the compression ratio $\varepsilon_{(int)}$. Furthermore, the inlet will be constrained to have a certain overall compression ratio, $\varepsilon_{(inlet)}$, at the design flight condition. So we have a simple constraint of

$$\varepsilon_{(inlet)} = \varepsilon^m_{(ext)}\varepsilon^n_{(int)} \tag{4}$$

Now we have two variables to solve for: $\varepsilon_{(ext)}$ and $\varepsilon_{(int)}$. The second constraint is that the flow after the last shock must be parallel to the x-axis [18]. Each of the $m+n$ shocks deflects the flow by a certain angle δ_i, so the flowpath angles are

$$\theta_1 = \alpha_{(design)} \qquad \theta_{i+1} = \theta_i + \delta_i \qquad \theta_{m+n+1} = 0 \tag{5}$$

It is possible to write functions $\delta_i = \delta(\varepsilon_i, M_i)$ and $M_{i+1} = M(\varepsilon_i, M_i)$, so it is relatively straightforward to calculate the partial derivatives

$$\left.\frac{\partial\theta_{m+n+1}}{\partial\varepsilon_{(ext)}}\right|_1, \quad \left.\frac{\partial\theta_{m+n+1}}{\partial\varepsilon_{(int)}}\right|_1, \quad \left.\frac{\partial\theta_{m+n+1}}{\partial M_{(design)}}\right|_1, \quad \text{and} \quad \left.\frac{\partial\theta_{m+n+1}}{\partial\alpha_{(design)}}\right|_1;$$
$$\left.\frac{\partial\varepsilon_{(inlet)}}{\partial\varepsilon_{(ext)}}\right|_1, \quad \left.\frac{\partial\varepsilon_{(inlet)}}{\partial\varepsilon_{(int)}}\right|_1, \quad \left.\frac{\partial\varepsilon_{(inlet)}}{\partial M_{(design)}}\right|_1, \quad \text{and} \quad \left.\frac{\partial\varepsilon_{(inlet)}}{\partial\alpha_{(design)}}\right|_1.$$

where the subscript 1 is shorthand for the first basis, which is $\varepsilon_{(ext)}$, $\varepsilon_{(int)}$, $M_{(design)}$, and $\alpha_{(design)}$. Since the system of equations from (4) and (5) can only be solved iteratively, we calculate the partial derivatives above for the values of $\varepsilon_{(ext)}$ and $\varepsilon_{(int)}$ that satisfy the two constraints. The problem now is that the set of partial derivatives we have calculated is different from the set of partial derivatives we need. The needed derivatives are

$$\left.\frac{\partial\varepsilon_{(ext)}}{\partial\varepsilon_{(inlet)}}\right|_2, \quad \left.\frac{\partial\varepsilon_{(int)}}{\partial\varepsilon_{(inlet)}}\right|_2, \quad \left.\frac{\partial\varepsilon_{(ext)}}{\partial M_{(design)}}\right|_2,$$
$$\left.\frac{\partial\varepsilon_{(int)}}{\partial M_{(design)}}\right|_2, \quad \left.\frac{\partial\varepsilon_{(ext)}}{\partial\alpha_{(design)}}\right|_2, \quad \left.\frac{\partial\varepsilon_{(int)}}{\partial\alpha_{(design)}}\right|_2.$$

where the second basis is $\varepsilon_{(inlet)}$, $M_{(design)}$, and $\alpha_{(design)}$.

Fortunately we can apply the chain rule to obtain a system of equations for the desired derivatives.

$$\left.\frac{\partial\varepsilon_{(inlet)}}{\partial\varepsilon_{(inlet)}}\right|_2 = 1 = \left.\frac{\partial\varepsilon_{(inlet)}}{\partial\varepsilon_{(ext)}}\right|_1 \left.\frac{\partial\varepsilon_{(ext)}}{\partial\varepsilon_{(inlet)}}\right|_2 + \left.\frac{\partial\varepsilon_{(inlet)}}{\partial\varepsilon_{(int)}}\right|_1 \left.\frac{\partial\varepsilon_{(int)}}{\partial\varepsilon_{(inlet)}}\right|_2 \tag{6}$$
$$\left.\frac{\partial\theta_{m+n+1}}{\partial\varepsilon_{(inlet)}}\right|_2 = 0 = \left.\frac{\partial\theta_{m+n+1}}{\partial\varepsilon_{(ext)}}\right|_1 \left.\frac{\partial\varepsilon_{(ext)}}{\partial\varepsilon_{(inlet)}}\right|_2 + \left.\frac{\partial\theta_{m+n+1}}{\partial\varepsilon_{(int)}}\right|_1 \left.\frac{\partial\varepsilon_{(int)}}{\partial\varepsilon_{(inlet)}}\right|_2 \tag{7}$$

Actually, the chain rule should give more terms than are listed here, but terms such as $\partial M_{(design)}/\partial\varepsilon_{(inlet)}|_2$ are zero because both $M_{(design)}$ and $\varepsilon_{(inlet)}$ are basis variables in the second basis. The next pair of derivatives can be solved using the system

$$\begin{aligned}\left.\frac{\partial\varepsilon_{(inlet)}}{\partial M_{(design)}}\right|_2 = 0 = \left.\frac{\partial\varepsilon_{(inlet)}}{\partial M_{(design)}}\right|_1 &+ \left.\frac{\partial\varepsilon_{(inlet)}}{\partial\varepsilon_{(ext)}}\right|_1 \left.\frac{\partial\varepsilon_{(ext)}}{\partial M_{(design)}}\right|_2 \\ &+ \left.\frac{\partial\varepsilon_{(inlet)}}{\partial\varepsilon_{(int)}}\right|_1 \left.\frac{\partial\varepsilon_{(int)}}{\partial M_{(design)}}\right|_2\end{aligned} \tag{8}$$
$$\begin{aligned}\left.\frac{\partial\theta_{m+n+1}}{\partial M_{(design)}}\right|_2 = 0 = \left.\frac{\partial\theta_{m+n+1}}{\partial M_{(design)}}\right|_1 &+ \left.\frac{\partial\theta_{m+n+1}}{\partial\varepsilon_{(ext)}}\right|_1 \left.\frac{\partial\varepsilon_{(ext)}}{\partial M_{(design)}}\right|_2 \\ &+ \left.\frac{\partial\theta_{m+n+1}}{\partial\varepsilon_{(int)}}\right|_1 \left.\frac{\partial\varepsilon_{(int)}}{\partial M_{(design)}}\right|_2\end{aligned} \tag{9}$$

This system is very similar to the one used to calculate the derivatives with respect to $\alpha_{(design)}$.

Once the values and derivatives of $\varepsilon_{(ext)}$ and $\varepsilon_{(int)}$, the actual design of the geometry proceeds as it is presented in [17], and the differentiation is straightforward. Worth noting here is that the functions δ and M from earlier in this subsection are first used to calculate the deflection angles $\delta_1, \ldots, \delta_{m+n}$. At this point we introduce four more design variables $M_{(min)}$, $M_{(max)}$, $\alpha_{(min)}$, and $\alpha_{(max)}$, which specify a range of flight conditions for which the inlet should not have any bad shock interactions. Selecting a wide range for these design variables (i.e. large difference between $M_{(max)}$ and $M_{(min)}$) has been shown to improve the robustness of the inlet [17,19]. Then the function $\beta_i = \beta(M_i, \delta_i)$ is used to calculate the wave angles for each extreme pair of flight conditions, i.e. $(M_{(min)}, \alpha_{(min)})$, $(M_{(min)}, \alpha_{(max)})$, $(M_{(max)}, \alpha_{(max)})$, and $(M_{(max)}, \alpha_{(min)})$.

E. Minimum, Maximum, Absolute Value, and Sorting

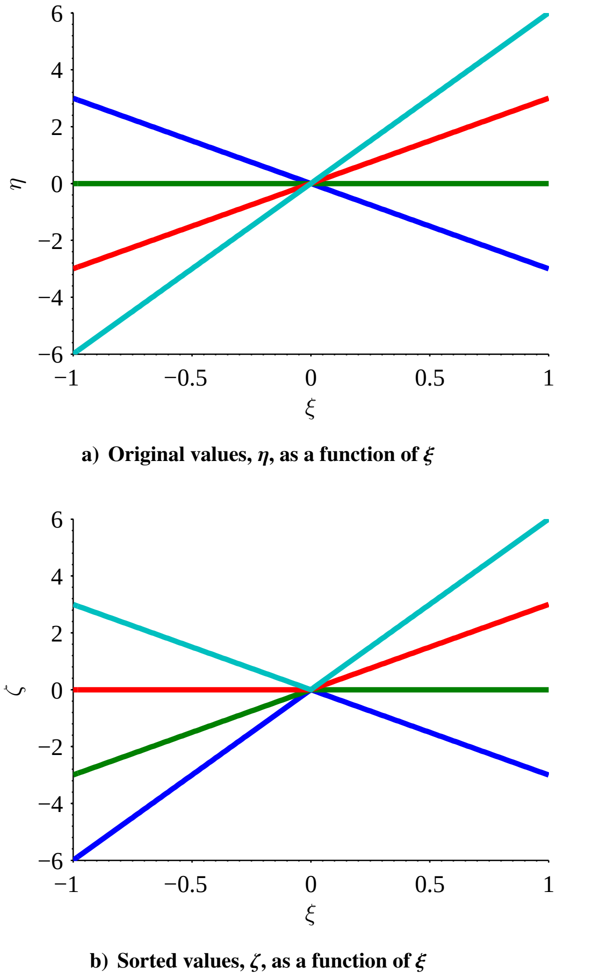
The next step in the inlet design process involves taking the minimum and maximum of several shock angles to ensure that the shocks placement is acceptable for the specified range of flight conditions [17]. This brings about a special type of problem when calculating derivatives because it introduces a discontinuity in the first derivative. While many techniques involve picking either the left-hand or right-hand derivative at such points, here we keep track of both derivatives. This section handles the three basic types of problems that involve comparing function values and keeping track of the derivatives.

The simplest continuous function that creates problems for derivatives is probably the absolute value.

$$\eta = \text{abs}(\xi) = \begin{cases} \xi, & \xi \geq 0 \\ -\xi, & \xi < 0 \end{cases} \tag{10}$$

However, we can write the left-hand and right-hand derivatives easily enough.

$$\frac{\partial\eta}{\partial\xi^-} = \begin{cases} 1, & \xi > 0 \\ -1, & \xi \leq 0 \end{cases} \qquad \frac{\partial\eta}{\partial\xi^+} = \begin{cases} 1, & \xi \geq 0 \\ -1, & \xi < 0 \end{cases} \tag{11}$$



a) Original values, η, as a function of ξ

b) Sorted values, ζ, as a function of ξ

Figure 4. Example of sorting four linear functions of one variable

Using Eq. (11) and the classic formulas

$$\max(\xi_1, \xi_2) = \frac{\xi_1 + \xi_2}{2} + \frac{|\xi_1 - \xi_2|}{2} \tag{12}$$

$$\min(\xi_1, \xi_2) = \frac{\xi_1 + \xi_2}{2} - \frac{|\xi_1 - \xi_2|}{2} \tag{13}$$

it is also easy to write the derivatives of the min and max functions.

While it is possible to apply min and max repeatedly to sort a list of values while keeping track of the left-hand and right-hand derivatives, the case of sorting the derivatives when several values are the same is worth more careful consideration. Consider the following relatively simple example of several lines intersecting at the origin.

$$\eta_i = c_i \xi \qquad i = 1, \ldots, N \tag{14}$$

$$[\zeta_1, \ldots, \zeta_N] = \mathrm{sort}[\eta_1, \ldots, \eta_N] \tag{15}$$

It is clear that $\zeta_1 = \ldots = \zeta_N = 0$ when $\xi = 0$, but it is less clear what the derivatives of each ζ_i should be.

Figure 4 shows an instance of the linear example with c_i-values of $-6, -3, 0, 3$. Each η_i line has a constant color in Fig. 4a, while each ζ_i has a constant color in Fig. 4b. At the intersection point, the lines are essentially broken apart so that the sorted values correspond to one value of η when $\xi < 0$ and a different η-value when $\xi > 0$. Looking at the kinked functions in Fig. 4b demonstrates that the derivatives of ζ_i at $\xi = 0$ are

$$\left[\frac{\partial \zeta_1}{\partial \xi^-}, \ldots, \frac{\partial \zeta_N}{\partial \xi^-}\right] = -\mathrm{sort}\left[-\frac{\partial \eta_1}{\partial \xi^-}, \ldots, -\frac{\partial \eta_N}{\partial \xi^-}\right] \tag{16}$$

$$\left[\frac{\partial \zeta_1}{\partial \xi^+}, \ldots, \frac{\partial \zeta_N}{\partial \xi^+}\right] = \mathrm{sort}\left[\frac{\partial \eta_1}{\partial \xi^+}, \ldots, \frac{\partial \eta_N}{\partial \xi^+}\right] \tag{17}$$

In other words, the left-hand derivatives are sorted in descending order, and the right-hand derivatives are sorted in ascending order.

F. Isolator, Combustor, and Nozzle Design

The geometry of the thrust-generating portion of the model is very simple. As shown in Fig. 5, the isolator is a simple constant-area section, while the combustor consists of a constant-area section followed by a diverging section. Because the lengths of the components are set by $\lambda_{(isolator)}$ and $\lambda_{(combustor)}$, this introduces only two new design variables: the length of the constant-area section, $\lambda_{(constant\text{-}area)}$, and the angle of the combustor divergence, $\theta_{(divergence)}$. We define $\lambda_{(constant\text{-}area)}$ to be the fraction of the combustor that is constant-area so that $0 \le \lambda_{(constant\text{-}area)} \le 1$.

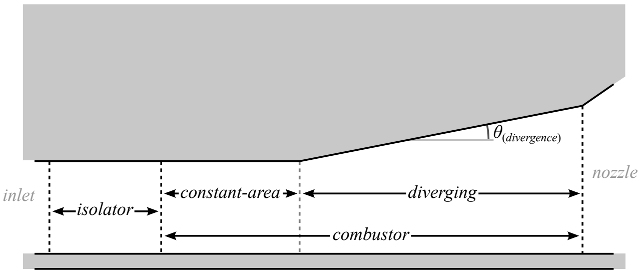


Figure 5. Sketch of isolator and combustor design.

The nozzle has a similarly simple design consisting of only a single ramp. Since the length of the nozzle is already set by the previous design parameters, the body portion of the nozzle requires a single further design parameter, which is the vertical offset of the vehicle's trailing edge compared to the leading edge. The nozzle then connects top of the end of the combustor to the trailing edge. The additional aspect of the design is the length of the cowl, as shown in Fig. 6, which is controlled by another design variable, $\lambda_{(cowl)}$.

G. Control Surfaces

The horizontal and vertical control surfaces are both considered to be flat plates with a moving section. The location of the horizontal control surface on the vehicle is determined by two variables, $x_{(elevator)}$ and $z_{(elevator)}$, which both range between 0 and 1. The position of the leading edge is selected so that the base of the horizontal control surface is entirely on the side of the fuselage, i.e. the wing does not go beyond the limits of the vehicle. For example, if $x_{(elevator)} = 1$, the trailing edge of the horizontal control surface will be at the trailing edge of the vehicle. Similarly, the vertical position of the control surface along the side of the vehicle is interpolated from the top of the vehicle fuselage to the bottom. Figure 7 shows how this is defined so that a value of $z_{(elevator)}$ between 0 and 1 ensures that the control surface is completely touching the fuselage.

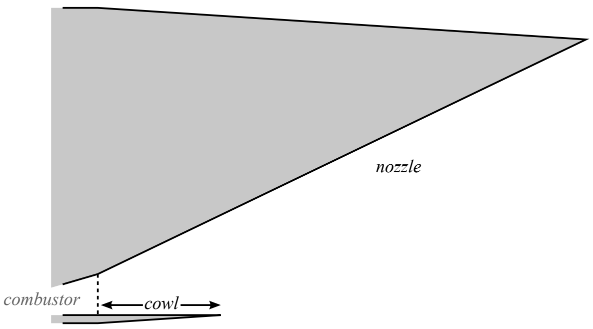


Figure 6. Sketch of nozzle design.

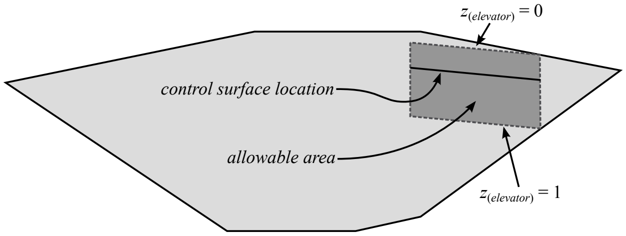


Figure 7. Sketch of side of vehicle model with vertical placement of horizontal control surface. The design variable is defined so that either the leading or trailing edge is touching the upper surface of the vehicle when $z_{(elevator)} = 0$.

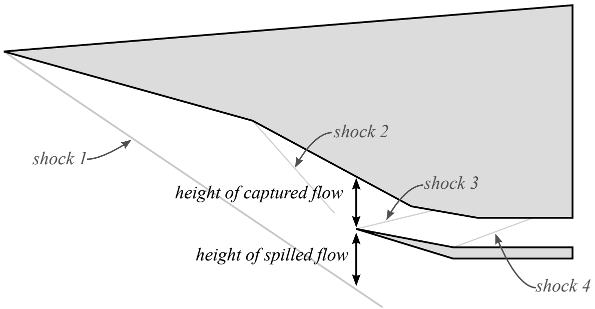


Figure 8. Sketch of mass flow spillage in inlet.

Design parameters used to describe the shape of horizontal control surface are chord length, sweep angle, dihedral angle, span, taper ratio, incidence angle, fraction of the chord that is a movable surface, and fraction of the span that is movable. The same parameters are in use for the tail (vertical control surface).

III. Force and Moment Calculation

Apart from the calculation of derivatives, the calculation of forces and moments is similar to that of Bolender and Doman [14] with a few modifications. As with previous hypersonic vehicle models, the analysis is split into four parts: the inlet, the isolator and combustor, the nozzle, and the external surfaces including control surfaces. The differentiation of the force and moment calculation is relatively straightforward, but several issues must carefully be considered.

A. Inlet Model

The inlet model has two phases; a sequence of shocks and a correction for mass spillage. There are m external shocks and n internal shocks, as described in Section II.D. The properties of the shock can actually be calculated analytically [20], so they cause no difficulties for the differentiation.

A simple analysis which considers the inlet flow to pass through one shock after another cannot account for the spilled flow that does not make it into the combustor. Following the shock calculations, we have values ρ_2^*, p_2^*, T_2^*, u_2^*, and M_2^* that describe the initial prediction of the conditions at the end of the inlet. If H_2 is the internal height at the end of the inlet (or beginning of the isolator), this means that the mass flow to the isolator is $\dot{m}_2^* = \rho_2^* u_2^* H_2$. We can see from Fig. 8 that the angle of the first shock provides a simple estimate of how much mass actually goes into the engine. If f is the fraction of air that is captured compared to the shock-on-lip condition, the actual mass flow rate is

$$\dot{m}_2 = \rho_2 u_2 H_2 = f \rho_2^* u_2^* H_2 \tag{18}$$

In this model, the excess mass is subtracted isentropically, so that

$$\frac{p_2}{p_2^*} = \left(\frac{1 + \frac{\gamma-1}{2}(M_2^*)^2}{1 + \frac{\gamma-1}{2}M_2^2} \right)^{\frac{\gamma}{\gamma-1}} \tag{19}$$

These two equations have the solution

$$f = \frac{M_2}{M_2^*} \left(\frac{1 + \frac{\gamma-1}{2}(M_2^*)^2}{1 + \frac{\gamma-1}{2}M_2^2} \right)^{\frac{1}{2}\frac{\gamma+1}{\gamma-1}} \tag{20}$$

which must be solved implicitly for M_2.

B. Combustor Model

The combustor model used here is the same as what is used in [14] and [12]. Rearranging the equations slightly and using a different station numbering, the equation for the heat addition is

$$\frac{M_4^2\left(1 + \frac{\gamma-1}{2}M_4^2\right)}{\left(1 + \gamma M_4^2\right)^2} = \frac{M_3^2\left(1 + \frac{\gamma-1}{2}M_3^2\right)}{\left(1 + \gamma M_3^2\right)^2} \frac{T_{04}}{T_{03}} \tag{21}$$

where the total temperature after heat addition, T_{04}, is a function of the equivalence ratio and T_{03}. A similar process is used for the diverging section of the combustor.

C. Nozzle Model

The nozzle model is also directly taken from [14] and [12] except that a nozzle cowl has been added. The pressure at the end of the combustor (and thus beginning of the nozzle) is p_5. The model assumes that the flow perfectly expands to atmospheric pressure, p_1, along the nozzle surface. The resulting force normal to a given panel of length L_i is

$$F_i = \frac{p_1 p_5}{p_5 - p_1} L_i \ln \frac{p_5}{p_1} \tag{22}$$

This formula is applied to both the lower surface of the nozzle body and the upper surface of the nozzle cowl shown in Fig. 6.

D. Exterior Surfaces

Surfaces that are not part of the inlet, isolator, combustor, or nozzle are split into panels that are analyzed using Newtonian aerodynamics. These panels are readily apparent in Fig. 2. One thing that is worth noting, however, is that panels with zero area cannot be ignored when calculating derivatives. For a given panel i, the force acting on the panel is $\mathbf{F}_i = -p_i A_i \hat{\mathbf{n}}_i$ where A_i is the area of the panel, and $\hat{\mathbf{n}}_i$ is the unit normal. This can also be written $\mathbf{F}_i = -p_i \mathbf{n}_i$ where $\|\mathbf{n}\|_i = A_i$. Then

$$\frac{\partial \mathbf{F}_i}{\partial \xi} = -p_i \frac{\partial \mathbf{n}_i}{\partial \xi} \tag{23}$$

when $A_i = 0$. An unfortunate problem here is that $\hat{\mathbf{n}}_i$ is not defined for a panel with no area, so we consider the unnormalized normal vector, $\mathbf{n}_i$ as a single entity that is a function of the vertex coordinates.

IV. Equations of Motion

For hypersonic vehicles, the velocities and altitude are high enough that the assumption a flat Earth is no longer valid. As a result, we employ a model that is consistent with a rotating WGS84 Earth [21, 22]. For example, an aircraft flying east at a constant altitude above the equator would require a lift force that is about 13% less than what would be predicted when using the flat-Earth equations of motion.

The output of the hypersonic vehicle model is a net specific force (i.e. net force divided by vehicle mass) and net specific moment in the body frame, which we write

$$\mathbf{F}^b = \mathbf{F}^b(\mathbf{x}, \mathbf{u}) \tag{24}$$

$$\mathbf{M}^b = \mathbf{M}^b(\mathbf{x}, \mathbf{u}) \tag{25}$$

Here $\mathbf{x}$ is a vector of state variables, and $\mathbf{u}$ is a vector of control variables. Since these forces and moments are calculated in a frame fixed to the body of the vehicle, we will write the equations of motion in this coordinate system. Since this frame is not an inertial frame, we cannot calculate the accelerations by just adding up all the forces in the body frame and dividing by the mass. Instead, we must apply the Coriolis transport theorem to account for the rotation of the body-fixed frame. Eventually this leads to an equation of the type

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \tag{26}$$

Two types of problem arise when calculating the derivatives of the equations of motion. From the previous section, we already have the derivatives of $\mathbf{F}^b$ and $\mathbf{M}^b$, but the function $\mathbf{f}$ introduced in (26) involves matrix multiplication and one matrix inversion.

A. Coordinate Systems

In order to apply Newton's second law directly, we need an inertial frame. In the current model, this is the Earth-centered inertial (ECI) frame, and we label it the *i*-frame.

$$\ddot{\mathbf{r}}^i_{ib} = \mathbf{f}^i + \mathbf{g}^i \qquad\qquad \mathbf{f}^i = \mathbf{C}^i_b\mathbf{f}^b \tag{27}$$

We also see the need to be able to transform a vector from one coordinate system to another, which requires a transformation matrix. In this notation, b denotes the body frame, which is attached to the vehicle. This notation is saying that multiplying a vector that is resolved in the b-frame by $\mathbf{C}^i_b$ on the left results in the same vector resolved in the i-frame.The chain rule for this example of matrix multiplication is

$$\frac{\partial \mathbf{f}^i}{\partial \xi} = \frac{\partial \mathbf{C}^i_b}{\partial \xi}\mathbf{f}^b + \mathbf{C}^i_b\frac{\partial \mathbf{f}^b}{\partial \xi} \tag{28}$$

but there is no vector form of this equation for the case that ξ is a vector. The options are to apply the chain rule for each variable of differentiation or use gradients for each element of the matrix.

The notation $\mathbf{r}^i_{ib}$ requires more explanation. The $\mathbf{r}_{ib}$ part refers to the vector from the origin of the i-frame (the center of the Earth) to the origin of the b-frame (the center of mass). The superscript i means that the vector is written as a vector in the i coordinate system.

Since the Earth is rotating, and the atmosphere more or less rotates with it (neglecting weather patterns), we describe the motion of the vehicle in a coordinate system where each point on the surface of the Earth has constant coordinates. For this purpose, we use the Earth-centered, Earth-fixed frame (ECEF), in which the z^e-axis points from the center of the Earth toward the north pole, the x^e-axis points toward the intersection of the equator and the prime meridian, and the y^e-axis completes a right-handed system. The angular velocity between the ECEF frame and the ECI frame is

$$\boldsymbol{\omega}^e_{ie} = \begin{bmatrix} 0 \\ 0 \\ \omega_{ie} \end{bmatrix} \qquad\qquad \omega_{ie} \approx 7.292115 \times 10^{-5}\ \text{rad/s} \tag{29}$$

To make the equations more intuitive, a coordinate system is introduced in which the x-axis points toward the local north, the y-axis points toward the local east, and the z-axis points locally down. This is called the navigation frame, and the transformation matrix from the navigation frame to the ECEF frame is

$$\mathbf{C}^e_n = \begin{bmatrix} -\sin L\cos\lambda & -\sin\lambda & -\cos L\cos\lambda \\ -\sin L\sin\lambda & \cos\lambda & -\cos L\sin\lambda \\ \cos L & 0 & -\sin L \end{bmatrix} \tag{30}$$

The navigation frame is carried with the vehicle, so that the origin of the navigation frame is always located at the center of mass of the vehicle. However, the navigation frame does not rotate with the vehicle, so that, for example, the z^n-axis always points directly normal to the surface of the Earth. Figure 9 gives a visual demonstration of the relation between the ECEF and navigation coordinate frames for an example point. The components of the vehicle velocity in this frame are

$$\mathbf{v}^n_{eb} = \mathbf{C}^n_e\dot{\mathbf{r}}^e_{eb} = \begin{bmatrix} v_N \\ v_E \\ v_D \end{bmatrix} \tag{31}$$

This equation shows an implicit assumption made in this notation; derivatives that are shown with a dot are always taken in the resolving frame. This means, for example, that $\mathbf{v}^n_{eb} \neq \dot{\mathbf{r}}^n_{eb}$. Also, the vehicle velocity and position are both always measured from the ECEF origin to the body origin, so the two subscripts will be dropped in the following references.

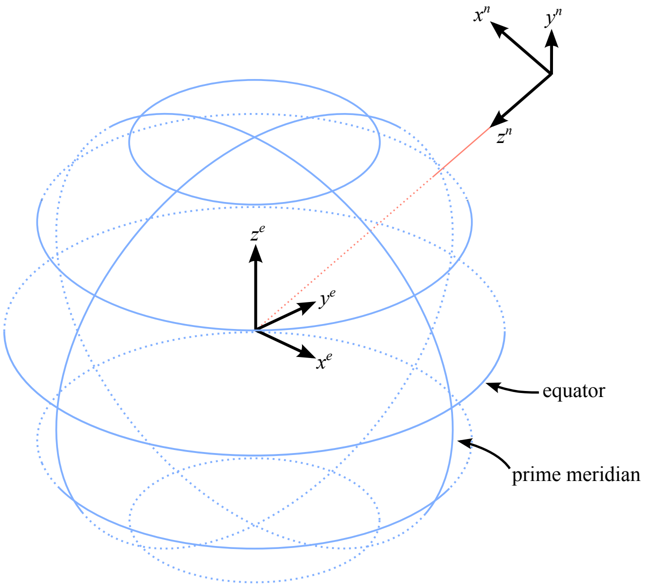


Figure 9. ECEF and navigation coordinate frames.

The final coordinate frame is the body frame, which was discussed previously and has the transformation matrix that carries the navigation frame to the body frame of

$$\mathbf{C}^b_n = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix}\begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}\begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{32}$$

The components of the velocity are $\mathbf{v}^b = (u, v, w)$.

B. The WGS84 Earth Model

Instead of using a spherical Earth, we have chosen to use the WGS84 ellipsoidal reference datum [21]. This is the same model used for the Global Positioning System (GPS) [22], and it thus provides an accurate reference for navigation. In this model, each curve of constant latitude is a circle, but each meridian is an ellipse with an eccentricity of

$$e = 0.0818191908426 \tag{33}$$

and a semi-major axis of

$$R_Q = 6378137\ \text{m} \tag{34}$$

One result of using an ellipsoidal Earth is the possibility of ambiguously defined latitude and altitude. In this paper, we use geodetic latitude, which has the property that moving normal to the reference ellipsoid does not change the latitude.

C. Net Accelerations

The equations of motion in the n-frame, which are often called the navigation equations [22] are

$$\dot{\mathbf{v}}^n = \mathbf{C}^n_b\mathbf{f}^b + \mathbf{g}^n - \boldsymbol{\omega}^n_{ie} \times (\boldsymbol{\omega}^n_{ie} \times \mathbf{C}^n_e\mathbf{r}^e) - (2\boldsymbol{\omega}^n_{ie} + \boldsymbol{\omega}^n_{en}) \times \mathbf{v}^n \tag{35}$$

and a simple application of the transport theorem provides the net accelerations in the body frame.

$$\dot{\mathbf{v}}^b = \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \mathbf{f}^b + \mathbf{C}^b_n(\mathbf{g}^n - \boldsymbol{\omega}^n_{ie} \times (\boldsymbol{\omega}^n_{ie} \times \mathbf{C}^n_e\mathbf{r}^e) - (2\boldsymbol{\omega}^n_{ie} + \boldsymbol{\omega}^n_{en} + \mathbf{C}^n_b\boldsymbol{\omega}^b_{nb}) \times \mathbf{v}^n) \tag{36}$$

The angular equations of motion in the body frame can be shown to be

$$\dot{\boldsymbol{\omega}}^b_{nb} = \left(\mathbf{I}^b\right)^{-1}\left(\mathbf{M}^b - \boldsymbol{\omega}^b_{ib} \times \mathbf{I}^b\boldsymbol{\omega}^b_{ib}\right) - \mathbf{C}^b_n\left(\dot{\boldsymbol{\omega}}^n_{en} + \boldsymbol{\omega}^n_{in} \times \mathbf{C}^n_b\boldsymbol{\omega}^b_{nb} + \boldsymbol{\omega}^n_{ie} \times \boldsymbol{\omega}^n_{en}\right) \tag{37}$$

where $\mathbf{I}^b$ is the inertia tensor divided by the total mass of the vehicle. In our model, we assume that changes in the mass of the vehicle simply change the uniform density of the entire vehicle volume, which is a first approximation used before the mass properties of the vehicle are realistically established. As a result, $\mathbf{I}^b$ is independent of the vehicle mass in this simple model. The components of the body angular velocity are given as $\boldsymbol{\omega}^b_{nb} = (P, Q, R)$.

Calculating the derivatives of $\dot{P}$, $\dot{Q}$, and $\dot{R}$ require knowing the derivatives of the matrix inverse $(\mathbf{I}^b)^{-1}$. A special case of implicit differentiation is useful here. Consider a system of equations

$$\mathbf{A}\boldsymbol{\eta} = \boldsymbol{\zeta}$$

where $\mathbf{A}$ and $\boldsymbol{\zeta}$ are both functions of a variable ξ, and the derivatives $\partial\boldsymbol{\eta}/\partial\xi$ are desired. Taking the derivative of both sides gives

$$\frac{\partial\mathbf{A}}{\partial\xi}\boldsymbol{\eta} + \mathbf{A}\frac{\partial\boldsymbol{\eta}}{\partial\xi} = \frac{\partial\boldsymbol{\zeta}}{\partial\xi}$$

which is a linear system that can easily be solved for $\partial\boldsymbol{\eta}/\partial\xi$ once the original linear system for $\boldsymbol{\eta}$ has been solved.

D. Trim

For our purposes, finding a trimmed flight condition entails selecting a value of $\dot{\mathbf{x}}$ and finding values of $\mathbf{x}$ and $\mathbf{u}$ that satisfy Eq. (26). The acceleration vector, $\dot{\mathbf{x}}$, consists of the velocity and angular velocity derivatives, $\dot{u}$, $\dot{v}$, $\dot{w}$, $\dot{P}$, $\dot{Q}$, and $\dot{R}$. Typically, we want to select values for some of the state variables (i.e., the entries in $\mathbf{x}$)) and find values all of the control values plus the remaining state variables. Using these concepts, we can rewrite Eq. (26) as

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{X}, \mathbf{Y}, \mathbf{u}) \tag{38}$$

where $\mathbf{X}$ is a vector of the independent state variables and $\mathbf{Y}$ contains the dependent state variables. In this discussion, we consider the independent state variables to be those describing the position, velocity, and angular velocity. The dependent variables include the orientation of the vehicle and the various control variables.

For this paper, the independent state variables are

$$\mathbf{X} = \begin{bmatrix} L & \lambda & h & M & \gamma & \sigma & P & Q & R \end{bmatrix}^T \tag{39}$$

and the dependent variables are

$$\mathbf{Y} = \begin{bmatrix} \alpha & \beta & \phi \end{bmatrix}^T \qquad \mathbf{u} = \begin{bmatrix} \delta_{ER} & \delta_{CE} & \delta_{DE} & \delta_{CR} \end{bmatrix}^T \tag{40}$$

where δ_{CE} is the average of the left and right elevator deflection angles (defined so that a positive deflection moves the trailing edge down) and δ_{DE} is the deflection angle of the right elevator minus the deflection angle of the left elevator. For simplicity we also make the assumption that the sideslip angle, β, is zero. This is convenient because it makes Eq. (38) a system of equations with six variables and six equations.

E. Ascending Trajectory

We consider trajectories with constant dynamic pressure. Using a standard atmospheric model [23], the assumption of constant dynamic pressure allows us to determine altitude as a function of Mach number. In other words there is a function such that $h = h(V)$. Since h and V are both basis variables, it is important to remember the relationship between h and V when we calculate derivatives of the performance. Differentiating this function with respect to time gives

$$\dot{h} = \frac{\mathrm{d}h}{\mathrm{d}V}\dot{V} \tag{41}$$

Since $\dot{h} = V\sin\gamma$, we can solve for the acceleration,

$$\dot{V} = V\frac{\mathrm{d}V}{\mathrm{d}h}\sin\gamma \tag{42}$$

The dynamic pressure is

$$q = \frac{1}{2}\rho V^2 \tag{43}$$

where ρ is given as a function of altitude, $\rho(h)$. Differentiating both sides with respect to altitude gives

$$\frac{\mathrm{d}V}{\mathrm{d}h} = -\frac{V}{2\rho}\rho'(h)$$

Substituting this result back into (42) gives

$$\dot{V} = -\frac{V^2}{2\rho}\rho'(h)\sin\gamma \tag{44}$$

This means that the acceleration, $\dot{V}$, can be determined from the flight-path angle, γ, or vice versa. The result is that the entire range of possible flight conditions with a given constant dynamic pressure can be expressed in terms of only two variables.

Since we have introduced a term involving $\mathrm{d}\rho/\mathrm{d}h$ into the analysis, our derivative analysis will require the value of $\mathrm{d}^2\rho/\mathrm{d}h^2$. For example,

$$\frac{\partial\dot{V}}{\partial h} = -\frac{V^2}{2\rho}\rho''(h)\sin\gamma$$

A very similar result occurs if we use $\dot{M}$ in our linearization analysis. Since M depends on the temperature, which is a function of altitude, we need the second derivative of temperature with respect to altitude.

$$\dot{M} = \frac{\dot{V}}{a} - \frac{V\dot{a}}{a^2} \qquad\qquad \dot{a} = \frac{1}{2}\frac{T'(h)}{T}a\dot{h} \tag{45}$$

In other words, we need to keep an extra derivative of the atmospheric model with respect to the altitude h.

V. Results

One of the main motivations for this work is to simulate the open-loop linearized dynamics using a technique with little or no computational instability. Calculating the linearized dynamics requires the derivatives of the equations of motion. Because the model used in this paper calculates the derivatives automatically, it is particularly well-suited to these linearization calculations.

Section V.C quantifies the numerical errors associated with using finite differences to calculate the linearization derivatives. Although the model is relatively simple for a tip-to-tail vehicle model, it is complicated enough to make roundoff errors in the finite differences to occur at relatively large step sizes. Finally, section V.D contains a discussion of the relative advantages and disadvantages of various techniques to calculate derivatives.

A. Trajectory Simulation

A model that can calculate the linearized dynamics is useful to compare against other models. The authors have developed another model, called MASIV [3, 5, 6, 24, 25], which contains analysis of effects such as shock interactions, finite-rate chemistry, fuel-air mixing, and a ram-mode solver. Like the model developed in this paper, it is a tip-to-tail vehicle model that contains trim analysis. The MASIV model has been successful in many respects, but performance values such as thrust are not always smooth functions of the input variables [7]. As a result, calculating the derivatives required for linearization is difficult [6, 8].

The model presented in this paper, however, calculates the derivatives exactly. Despite the fact that the two models are very different, and indeed consider slightly different vehicles, it is useful to see if the two models predict qualitatively similar linearized dynamics. This is especially informative in the case of the Dutch-roll mode because there is so little information available about the lateral-directional dynamics of hypersonic vehicles. Although comparing MASIV with a model of lesser fidelity does not necessarily provide information on the accuracy of the original model, it can provide a check for consistency. It also provides information on what aspects of the model can be responsible for changes in the flight dynamics. This alludes to the fact that two models could have significant differences in the intermediate values and still have very similar linearized flight dynamics. The same could be said about two different designs. As an example, consider two engines that have different combustion efficiencies, and everything else is the same. At a given flight condition, the short-period mode would probably be almost identical for both vehicles because the one with the lower efficiency would simply inject more fuel to get the same acceleration.

Figure 10 shows the locus of poles for modes of two linearized flight dynamic modes using both models. The short-period mode is a purely longitudinal mode (i.e. it is a two-dimensional process and does not involve roll or yaw of the vehicle) that primarily involves the angle of attack (α) and the pitch rate (Q). It is shown for MASIV in Fig. 10a and the model from this paper inf Fig. 10c. The Dutch-roll mode is primarily a trade between sideslip angle (β) and yaw rate (R), and is shown in Figs. 10b and 10d.

The vehicles used for the trajectory simulations are the ones from Figs. 1 and 2. For the low-order vehicle from Fig. 2, the design Mach number range is 7 to 9, the inlet compression ratio is 70, and the ballast mass in the nose is 20% of the weight of the vehicle. This hight ballast weight was used to shift the center of gravity forward to qualitatively match the short-period stability of the MASIV model. The MASIV vehicle design has been described in previous works, particularly [6] and [17]. The inlet compression ratio used here is 70, and the design Mach number range is 7 to 11.

The more interesting results are probably in the short-period plots. The MASIV results in Fig. 10a appear to show a stable, oscillatory short-period mode, and as the Mach number increases, both the time-to-half and the oscillation frequency decrease. However, the jaggedness of the plot seems to cast doubt on at least some of these statements. Discomfort with this plot in particular was one of the reasons to investigate exact differentiation techniques. In particular, the fine-scale variations mean that there is some amount of computational noise, and it raises doubt about the bulk properties of the short-period mode calculation.

However, the comparison between Figs. 10a and 10c show the same overall trends. This is not a confirmation that either model is physically correct, since there is no comparison to flight data, but it does indicate that the overall trend of the MASIV short-period pole locus is not a computational aberration. Instead, it appears that the uncertainty introduced by the non-smooth MASIV model is on the order of the fluctuations in Fig. 10a.

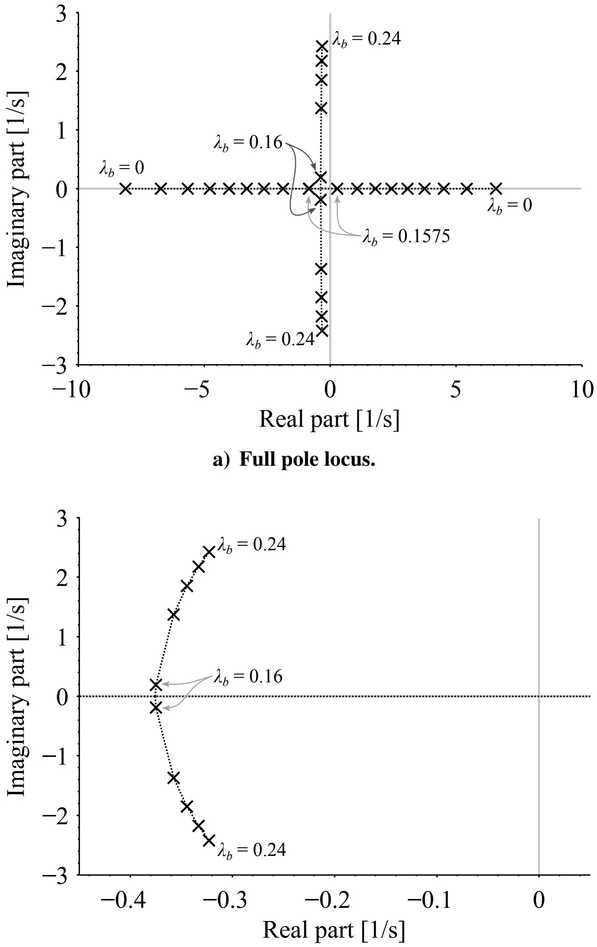
The results for the Dutch-roll mode are somewhat different. Figure 10b does not show large fluctuations, which is mostly due to the relative simplicity of the MASIV lateral force model compared to the MASIV engine model. Overall, the Dutch-roll mode shows a similar trend to that of the short-period mode except that it is much more lightly damped. The results from the exact differentiation model in Fig. 10d actually differ substantially. For one thing, the damping is much greater, and the frequency increases rather than decreases with Mach number. However, this is mostly due to the difference in vehicle design. In particular, the large, flat sides of the vehicle used in the present model produce very large lateral forces while they offer little or no damping for roll perturbations.

B. Center of Mass Location

Figures 11 and 12 investigate the effects of the center of mass location on the stability of the linearized flight dynamics modes. The results from Section V.A showed stable short-period modes, which differs from the results in [12] and [14]. However, shifting the center of mass far enough forward will cause almost any air vehicle to be stable, so the difference of results should not be alarming.

In this model, we consider the vehicle to have a uniform density except for an extra ballast mass placed at the nose. These two assumptions lead to both a center of mass location and an inertia tensor. With no ballast mass, both modes are unstable with no oscillations. As ballast mass, the short-period mode becomes less unstable until at a certain point it switches to a stable oscillatory mode, as shown in Fig. 11a. From this point, Fig. 11b shows that adding more ballast causes both the time-to-half and the frequency to actually increase.

The Dutch-roll poles plotted in Fig. 12 show a more complicated trend. With zero ballast, the mode is unstable and non-oscillatory, but adding ballast actually makes the mode more unstable. But at a certain amount of ballast, the Dutch-roll abruptly changes to a stable, oscillatory mode. From there, each of the two poles seems to be making a half ellipse back towards the origin.



a) Full pole locus.

b) Close-up of stable poles.

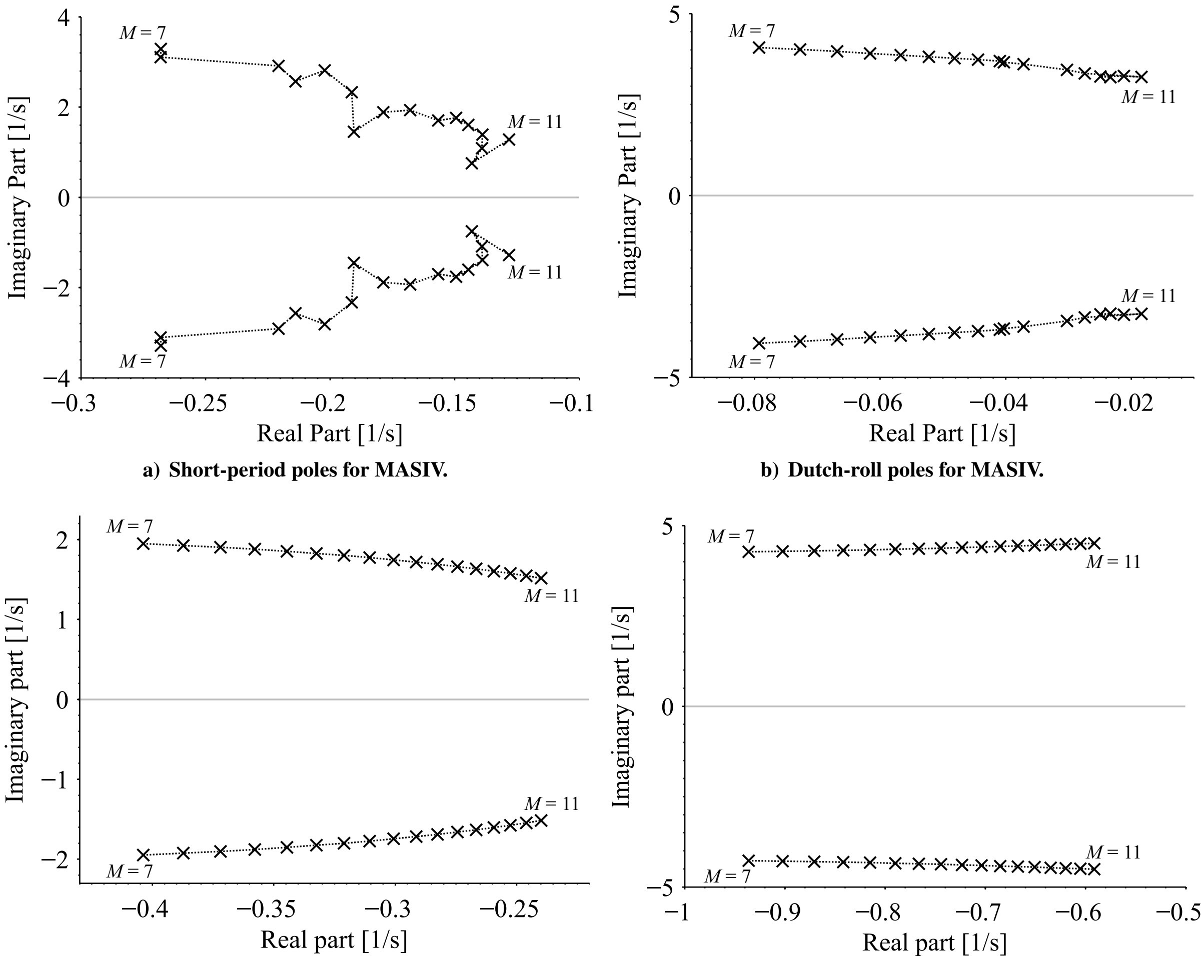
Figure 11. Short-period poles for various ballast masses, $\lambda_b = m_{(ballast)}/m_{(vehicle)}$ at Mach 8 and one atmosphere dynamic pressure.

For both modes, the lowest ballast that results in a stable mode actually gives the stablest mode for any of the cases. This seems a little unusual, but it is important to note that the center of mass is not changed in isolation. Moving the center of mass also affects the trim condition, and thus the changes to the linearized flight dynamics are also affected by the changes to trim angle of attack, equivalence ratio, and elevon angle.

C. Derivative Convergence

Although finite-difference approximations to derivatives have well-known and well-documented potential pitfalls [2], they are extremely simple to apply, and they can serve as a test for more challenging approximation methods. Here the simple forward finite-difference approximation has been applied to several derivatives using a range of step sizes from 10^{-16} to 10^{-1}. Because the differentiation was written manually for this code, such testing is particularly important to verify accuracy of the analytical derivatives. However, the finite-difference approximation does not fully converge when implemented on a computer, so the comparison only assures that the analytical derivatives are accurate to a relative error of about 10^{-8}. Using the complex-step approximation would give a higher degree of verification.

Since the analysis has been done using double-precision numbers, the computer will say that $a - b$ is zero if $|a - b|/a < 10^{-16}$. Because



a) Short-period poles for MASIV.

b) Dutch-roll poles for MASIV.

c) Short-period poles for present model.

d) Dutch-roll poles for present model.

Figure 10. Poles of the open-loop linearized dynamics for two vehicle models on a constant dynamic pressure trajectory of 1 atm accelerating from Mach 7 to Mach 11.

the forward finite difference,

$$f'(\xi) = \frac{f(\xi + \Delta\xi) - f(\xi)}{\Delta\xi} + O(\Delta\xi) \tag{46}$$

requires subtracting very small numbers as the step size decreases, it predicts a sensitivity of exactly 0 if the relative change in f is less than 10^{-16}, and in most cases problems occur even for much larger step sizes.

Figure 13 shows the comparison between the analytical derivative of the forward acceleration, $\dot{u}$ to a flight condition variable (angle of attack, α) and a design variable (design Mach number, $M_{(design)}$) using the low-order model. The equations of motion are applied to a vehicle flying at Mach 8 and an altitude of 28000 m. The angle of attack is 0, and the equivalence ratio (i.e. the throttle setting) is 0.4. For these conditions, the value of $\dot{u}$ is $-0.2163\ \mathrm{m/s^2}$.

Similarly, Fig. 14 shows the sensitivity comparison for two derivatives of T_5, which is the static temperature immediately after the constant-area heat addition. To show the copious amount of derivatives that the model calculates, the figure shows the accuracy of the derivatives of an intermediate computation, T_5, to a flight condition variable (angle of attack, α) and a model parameter (combustion efficiency, $\eta_{(combustion)}$). The actual values associated with these figures are

$$\begin{array}{lll} \dot{u} = -0.2163\ \mathrm{m/s^2} & \dfrac{\partial \dot{u}}{\partial \alpha} = -52.83\ \dfrac{\mathrm{m}}{\mathrm{s^2 \cdot rad}} & \dfrac{\partial \dot{u}}{\partial M_{(design)}} = 1.544\ \mathrm{m/s^2} \\ T_5 = 2851\ \mathrm{K} & \dfrac{\partial T_5}{\partial \alpha} = 1740\ \mathrm{K/rad} & \dfrac{\partial T_5}{\partial \eta_{(combustion)}} = 2699\ \mathrm{K} \end{array}$$

The plots in Figs. 13 and 14 illustrate the subtraction errors that plague the finite-difference approximation. For step sizes lower than approximately 10^{-6}, the approximation shows the expected first-order convergence. For smaller step sizes, the approximation becomes less and less accurate, and the graph is not smooth. The non-smooth nature of the left portion of each curve is probably due to the fact that more and more intermediate computations experience loss of significance as the step size decreases. Eventually all of the computations have loss of significance, and the derivative approximation goes to zero, although this is only visible for the very smallest step sizes in Fig. 13b here.

D. Discussion

Although there are several remaining technical issues with algorithmic differentiation and the complex-step approximation, the theoretical problems rarely show up in practice [2]. Using the proper tools, both of those methods can be relatively easy to implement. In this model, differentiation was done manually as part of the code, so the theoretical problems are not a concern, but it required considerably more work to implement.

Compared to the complex-step approximation, this approach toward calculating derivatives has both advantages and disadvantages. Using Matlab, implementing the complex step requires very little work even for complicated programs, while any type of algorithmic differentiation almost requires that the program use a dedicated syntax. The implementation used here is even more difficult because the coder is also responsible for keeping track of the derivatives, and a special syntax was created to ease this process. However, this method keeps track

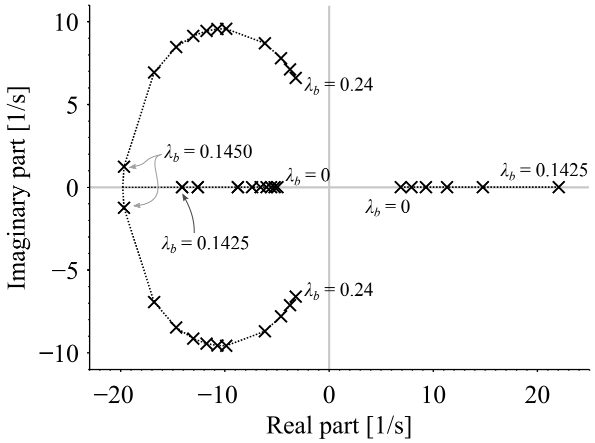
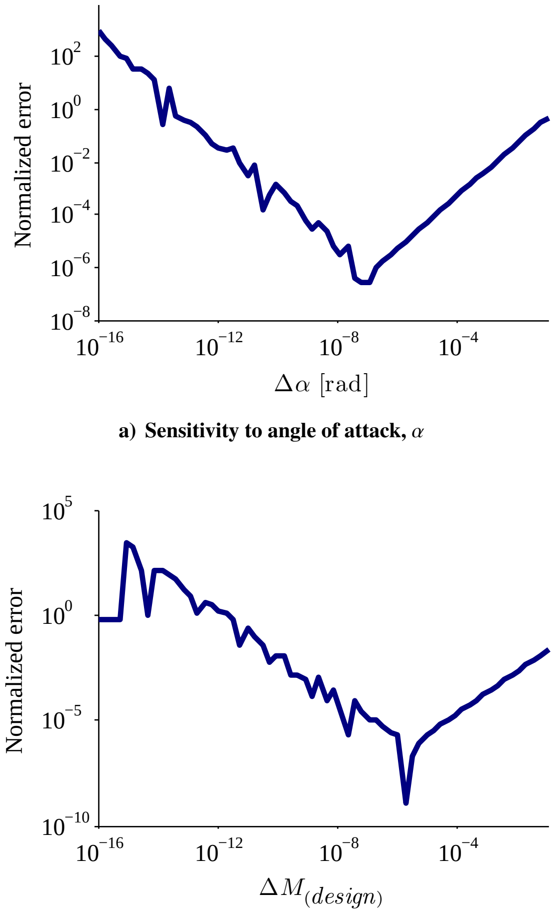


Figure 12. Dutch-roll poles for various ballast masses, $\lambda_b = m_{(ballast)}/m_{(vehicle)}$ at Mach 8 and one atmosphere dynamic pressure.

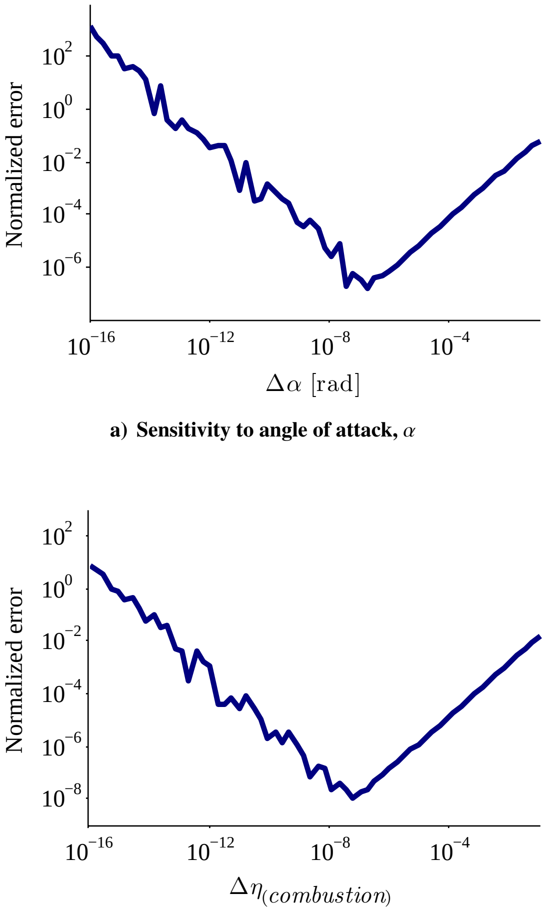


a) Sensitivity to angle of attack, α

b) Sensitivity to design Mach number, $M_{(design)}$

Figure 13. Relative error in the sensitivity estimates given by the forward finite-difference method of forward acceleration ($\dot{u}$) using the analytic result as reference. The errors are normalized by the absolute value of the analytic result.

of left-hand and right-hand derivatives separately, which can alert to otherwise undetectable behavior of the system. For example, in the standard atmosphere the temperature is defined as a piecewise linear function of geopotential altitude [23]. If the vehicle is flying at a geopotential altitude of exactly 11 km, the derivative of temperature with respect to an increase in altitude will differ from the derivative with respect to a decrease in altitude. In this model, these differences were found to be extremely rare in practice.



a) Sensitivity to angle of attack, α

b) Sensitivity to design combustion efficiency, $\eta_{(combustion)}$

Figure 14. Relative error in the sensitivity estimates given by the forward finite-difference method of static temperature after constant-area heat addition (T_5) using the analytic result as reference. The errors are normalized by the absolute value of the analytic result.

Another advantage of the present method is that it works without modification for complex-valued functions or for systems with complex-valued inputs. Although this is a problem that can be overcome with the complex-step approximation, for example by keeping track of the real part and imaginary part of each function separately, it would make the two methods for differentiation about equally difficult to implement. In the current hypersonic vehicle model, there are no complex-valued functions.

The present implementation calculates the derivatives with respect to a large set of basis variables each time the vehicle performance is calculated. This provides convenience because the derivatives are always present, but naturally that comes at a price of efficiency if not all the derivatives will be used later. Perhaps more importantly, this process almost requires establishing a set of basis variables before the computation. Calculating a derivative with respect to a different variable can be difficult or impossible.

By comparison, the complex-step approximation requires one function evaluation for each derivative. In other words, to calculate the derivatives with respect to ξ_1, ξ_2, ξ_3 would require three function evaluations. On the other hand, the user can pick an input with respect to which to differentiate at any point. Consider an example where

$$\zeta = f_2(\eta) \qquad \eta = f_1(\xi) \qquad (47)$$

Supposing that η is a basis variable, we can easily (and automatically) calculate $\partial\zeta/\partial\eta$, and we can calculate $\partial\xi/\partial\eta$ implicitly. However, they may not be any way to represent $\partial\eta/\partial\xi$ in the syntax because ξ is not a basis variable. Provided that f_1 is actually a function of several variables, getting $\partial\eta/\partial\xi$ from the derivatives that we would have may require an application of the inverse function theorem. In contrast, the complex-step method can provide an approximation by simply evaluating $f_2(\xi + i\Delta\xi)$. This is relevant to the hypersonic vehicle example because there are multiple conventions to specify the flight condition. For example, it is sometimes useful to specify the pitch angle and the angle of attack, while other times it is more useful to specify the angle of attack and velocity flight path angle.

Another approach similar in many ways to complex-step differentiation is automatic differentiation. With such a technique, numeric variables are replaced with a special class of variable that stores both the value of the variable and its derivative (or sometimes Jacobian or even Hessian). Then each operation, such as addition, multiplication, trigonometric functions, exponentiation, calculation of eigenvalues, etc., is rewritten to apply the chain rule while it calculates the results. The authors have implemented a version of this technique in Matlab, but the results are not presented here.

Although the various differentiation methods have various advantages and disadvantages, they can also be used to work together. Testing and verification are two obvious applications, but they can also be useful for calculating approximations to higher-order derivatives. For one thing, the complex-step approximation loses many of its advantages even for the second derivative [26], so that method will need help if second derivatives or mixed derivatives are important. Finite differences can always be used as a last resort, but an interesting idea is to combine algorithmic differentiation and the complex-step approximation. If a program has been set up to calculate n derivatives, then using the complex step and taking the imaginary part of the nth derivative gives an easy-to-implement approximation to the $(n + 1)$st derivative.

VI. Conclusions

A simple hypersonic vehicle model has been constructed that automatically calculates derivatives along with its performance. These derivatives allow a user to easily calculate the linearized dynamics of the vehicle without any further computation, and the derivatives can also be useful for design optimization. Left-hand and right-hand derivatives are tracked separately in the case that the vehicle flight condition or design is at a discontinuity in any of the first derivatives.

The model provides both a useful tool for analysis of hypersonic flight and an example of exact differentiation applied to a complex system. For example, calculation of sensitivities to model parameters provides valuable information for model uncertainty analysis.

The short-period and Dutch-roll modes were compared to those calculated for a higher-fidelity model. The comparison showed that both models produce qualitatively similar trends for the linearized open-loop dynamics, which provides an indication that the two models are at least consistent. The effects of ballast mass on the stability of the vehicle are also investigated. Unsurprisingly, adding ballast in order to move the center of gravity forward can make an unstable vehicle stable. A more interesting result is that adding further ballast to a stable vehicle can reduce the stability margin for the short-period and Dutch-roll modes.

Acknowledgements

This research is funded by the Air Force Research Laboratory/Air Vehicles Directorate grant FA 8650-07-2-3744 for the Michigan/AFRL Collaborative Center in Control Sciences. Dr. Michael Bolender as technical monitor.

References

[1] Khalil, H. K., *Nonlinear Systems*, Prentice Hall, 2002.

[2] Martins, J. R. R. A., Sturdza, P., and Alonso, J. J., "The Complex-Step Derivative Approximation," *ACM Transactions on Mathematical Software*, Vol. 29, No. 3, 2003.

[3] Dalle, D. J., Fotia, M. L., and Driscoll, J. F., "Reduced-Order Modeling of Two-Dimensional Supersonic Flows with Applications to Scramjet Inlets," *Journal of Propulsion and Power*, Vol. 26, No. 3, 2010, pp. 545–555.

[4] Torrez, S. M., Dalle, D. J., and Driscoll, J. F., "Multidisciplinary Optimization of the Fuel Consumption of a Dual Mode Scramjet-Ramjet," *47th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, 2011, AIAA Paper 2011-5757.

[5] Dalle, D. J., Torrez, S. M., and Driscoll, J. F., "Rapid Analysis of Scramjet and Linear Plug Nozzles," *Journal of Propulsion and Power*, Vol. 28, No. 3, 2012, pp. 545–555.

[6] Dalle, D. J., Torrez, S. M., Driscoll, J. F., and Bolender, M. A., "Flight Envelope Calculation of a Hypersonic Vehicle Usinga First Principles-Derived Model," *17th AIAA International Space Planes and Hypersonic Systems and Technologies Conference*, 2011.

[7] Torrez, S. M., Driscoll, J. F., Dalle, D. J., Bolender, M. A., and Doman, D. B., "Hypersonic Vehicle Thrust Sensitivity to Angle of Attack and Mach Number," *AIAA Atmospheric Flight Mechanics Conference*, 2009, AIAA Paper 2009-6152.

[8] Dalle, D. J., Torrez, S. M., and Driscoll, J. F., "Turn Performance of an Air-Breathing Hypersonic Vehicle," *AIAA Atmospheric Flight Mechanics Conference*, 2011.

[9] Griewank, A., *Evaluation Derivatives*, SIAM, 2000.

[10] Green, L. L., Newman, P. A., and Haigler, K. J., "Sensitivity Derivatives for Advanced CFD Algorithm and Viscous Modeling Parameters vio Automatic Differentiation," *Journal of Computational Physics*, Vol. 125, 1996, pp. 313–324.

[11] Slawig, T. and Zickfeld, K., "Parameter optimization using algorithmic differentiation in a reduced-form model of the Atlantic thermohaline circulation," *Nonlinear Analysis: Real World Applications*, Vol. 5, No. 3, 2004, pp. 501–518.

[12] Chavez, F. R. and Schmidt, D. K., "Analytical Aeropropulsive/Aeroelastic Hypersonic-Vehicle Model with Dynamic Analysis," *Journal of Guidance, Control, and Dynamics*, Vol. 17, No. 6, 1994, pp. 1308–1319.

[13] Frendreis, S. G. V. and Cesnik, C. E. S., "3D Simulation of a Flexible Hypersonic Vehicle," *Atmospheric Flight Mechanics Conference & Exhibit*, 2010, AIAA Paper 2010-8229.

[14] Bolender, M. A. and Doman, D. B., "Nonlinear Longitudinal Dynamical Model of an Air-Breathing Hypersonic Vehicle," *Journal of Spacecraft and Rockets*, Vol. 44, No. 2, 2007, pp. 374–387.

[15] Bischof, C., Carle, A., Corliss, G., Griewank, A., and Hoveland, P., "ADIFOR: Generating derivative codes from Fortran programs," *Sci. Prog.*, Vol. 1, No. 1, 1992, pp. 11–29.

[16] Bischof, C. H., Roh, L., and Mauer-Oats, A. J., "ADIC; An extensible automatic differentiation tool for ANSI-C." *Softw. – Prac. Exp.*, Vol. 27, No. 12, 1997, pp. 1427–1456.

[17] Dalle, D. J., Torrez, S. M., and Driscoll, J. F., "Performance Analysis of Variable-Geometry Scramjet Inlets Using a Fundamental Model," *47th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, 2011.

[18] Smart, M. K., "Optimization of Two-Dimensional Scramjet Inlets," *Journal of Aircraft*, Vol. 36, No. 2, 1999, pp. 430–433.

[19] Torrez, S. M., Driscoll, J. F., Dalle, D. J., and Fotia, M. L., "Preliminary Design Methodology for Hypersonic Engine Flowpaths," *16th AIAA/DLR/DGLR International Space Planes and Hypersonic Systems and Technologies Conference*, 2009, AIAA Paper 2009-7289.

[20] Thompson, M. J., "A Note on the Calculation of Oblique Shock Wave Characteristics," *Journal of Aerospace Sciences*, Vol. 17, 1950, pp. 741–744.

[21] Anon., "Department of Defense World Geodetic System 1984," Tech. Rep. TR8350.2, 3rd ed., National Imagery and Mapping Agency (now NGA), 1997.

[22] Groves, P. D., *Principles of GNSS, Inertial, and Multisensor Integrated Navigations Systems*, Artec House, 2008.

[23] "U.S. Standard Atmosphere," U.S. Government Printing Office, 1976, Washington, D.C.

[24] Torrez, S. M., Driscoll, J. F., Ihme, M., and Fotia, M. L., "Reduced Order Modeling of Turbulent Reacting Flows With Application to Scramjets," *Journal of Propulsion and Power*, Vol. 27, No. 2, March-April 2011, pp. 371–382.

[25] Torrez, S. M., Dalle, D. J., and Driscoll, J. F., "Design of Dual-Mode Hypersonic Vehicle Flow Paths Using a Reduced-Order Model," *17th AIAA International Space Planes and Hypersonic Systems and Technologies Conference*, April 2011.

[26] Lai, K.-L. and Crassidis, J. L., “Extensions of the first and second complex-step derivative approximations,” *Journal of Computational and Applied Mathematics*, Vol. 219, No. 1, 2008, pp. 276–293.