53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference
20th AI
23 - 26 April 2012, Honolulu, Hawaii

AIAA 2012-1708

Toward an Unsteady Aerodynamic ROM for Multiple Mach Regimes

Torstens Skujins* and Carlos E. S. Cesnik†
University of Michigan, Ann Arbor, Michigan 48109

The accurate prediction of unsteady aerodynamic loads is of utmost importance in an aeroelastic simulation framework, which often times will be used to simulate a vehicle passing through multiple Mach regimes on a single flight path. In terms of computational efficiency, it is beneficial for an unsteady aerodynamic model to have a single mathematical form for the aerodynamic loads through the different Mach regimes for an entire simulation. An unsteady aerodynamic reduced-order modeling methodology based on linear convolution combined with a nonlinear correction factor is developed to fulfill this need. Though each Mach regime presents unique modeling challenges, the transonic regime is particularly challenging in this regard due to moving shocks and other nonlinear flow field effects. The purpose of this paper is to characterize both the error of the reduced-order model within the transonic regime as well the applicability of a simplified calculation procedure for the nonlinear correction which can replace, at times, full CFD simulations within the modeling framework. The reduced-order model results show generally good agreement with computational simulations of the AGARD 445.6 wing undergoing oscillation of multiple elastic modes. Also, at higher oscillation frequencies, a distinct phase shift which develops between the model and full-order simulation results has been quantified. Finally, one factor which appears to adversely affect the model's accuracy is the increasing memory length of the system being modeled.

I. Introduction

Accurate calculation of unsteady aerodynamic loads due to vehicle rigid body pitch and plunge motion as well as elastic deformations is vital to both the determination of the flutter boundary and development of control algorithms. Often times, vehicles will encounter several different Mach regimes throughout the duration of a single flight; for example, hypersonic vehicles must pass through the subsonic, transonic, and supersonic regimes on the way to hypersonic flight. The different Mach regimes offer distinct challenges when modeling the unsteady aerodynamic loads. In the transonic regime, moving shocks and other effects result in a nonlinear flow field, thus complicating the prediction of aerodynamic forces. In the hypersonic regime, strong shocks, shock/boundary layer interactions, and the relatively large magnitude of the aerodynamic loads are among the factors which present further modeling challenges. When modeling the unsteady aerodynamic loads on a single vehicle throughout several different Mach regimes, a single mathematical form representing the loads in all regimes is important to have when coupling with a full aerothermoelastic framework. This increases the efficiency since the same aeroelastic equations can be used without modification regardless of the Mach regime in question.

Computational fluid dynamics (CFD)-based reduced-order models (ROMs) provide an effective way to model unsteady aerodynamic loads. Once constructed, the models run orders of magnitude faster than full CFD solutions while preserving a high level of accuracy seen by the original computational simulations. Silva[1] developed a method using first- and second-order Volterra kernels to predict the unsteady loads on a transonic airfoil; similar methodology was then extended to create a state-space system useful for aeroservoelastic analysis.[2] Raveh[3] used step responses to find the generalized aerodynamic forces for aeroelastic analysis in

*Graduate Research Assistant, Department of Aerospace Engineering, University of Michigan, Email: tskujins@umich.edu, Member AIAA.

†Professor, Department of Aerospace Engineering, University of Michigan, Email: cesnik@umich.edu, 1320 Beal Avenue, 3024 FXB, Ann Arbor, MI 48109-2140, Ph. (734) 764-3397, Fax: (734) 764-0578, AIAA Fellow.

the transonic regime. One major drawback of many reduced-order models is that they are only valid for flight conditions immediately around those from which the model was constructed. Efforts have been made to make the ROMs valid over a range of parameters. Silva[4] used a convolution-type methodology to construct a state-space ROM which is then used over a range of velocities in the transonic regime by modifying the time step of the numerical integration. Glaz and co-workers used a surrogate-based recurrence framework ROM to model the unsteady aerodynamics on a 2-D rotating airfoil[5] as well as the dynamic stall effects of a helicopter rotor blade.[6,7] Crowell and McNamara[8] combined steady CFD modeling with an unsteady piston theory correction term in modeling unsteady hypersonic aerodynamics. Other efforts for parameter-independent ROMs have focused on the analysis of flight test data. Lind et al.[9] created velocity-independent kernels by using curve fits of flight test data gathered at different conditions. Baldelli et al.[10] created a model valid over a range of dynamic pressures by combining linear and nonlinear operators for model construction. Prazenica et al.[11] extrapolated kernels found at different flight conditions to create one model valid over a range of conditions. Omran and Newman[12] used Volterra series submodels in different domains, such as pre-stall and post-stall, to construct an overall global piecewise Volterra series model.

The above ROM efforts have focused on a single regime and have not established their applicability across multiple Mach regimes. However, the most efficient way to model the aerodynamics of a vehicle designed to fly in multiple Mach regimes is to have a single ROM method applicable to all of the regimes while also minimizing the computational cost of model construction. To this end, Ref. 13 lays out the basic framework for a CFD-based ROM technique for the calculation of unsteady aerodynamic loads applied to the hypersonic regime, while Ref. 14 shows its potential application to the transonic regime. However, much of the transonic testing up to this point has focused on single-modal oscillations and the development of a transonic simplified model to aid in ROM construction. Still needed are both a rigorous, quantitative assessment of the model generalized to multiple modes of oscillation in the transonic regime as well as an exploration of the method's applicability boundaries and limitations, which are presented here. These items provide key contributions to the overall understanding of how the ROM can be applied across a range of Mach numbers.

II. ROM Methodology

The reduced-order modeling methodology presented here combines linear convolution with a nonlinear correction factor. The unsteadiness of the flow is captured through the use of linear convolution, which has been shown to be an effective modeling tool for linear unsteady aerodynamics.[1,15] However, the main drawbacks of a pure convolution ROM stem from the fact that unsteady aerodynamics are in general nonlinear in nature. As described subsequently, the convolution ROM is based upon the aerodynamic response to a step input of a certain magnitude applied to the geometric configuration under consideration. When the actual input magnitude in a particular simulation increases beyond that used in the ROM development, the ROM accuracy begins to erode due to various nonlinearities. Moreover, the convolution ROM may not be valid for flight conditions away from those around which the model is constructed. To address these issues, a nonlinear correction factor has been introduced to this convolution ROM by calculating the response of the system at larger amplitudes and a range of Mach conditions. Thus, rather than being geometry, amplitude, or Mach number-dependent, the general mathematical form of the model does not place any inherent limitations on configurations, input size, or flow conditions for which it is applicable.

In general, the nonlinear corrected ROM response y_{corr} can be written as

$$y_{corr} = f_c y_{conv} \tag{1}$$

where f_c is the correction factor and y_{conv} is the linear convolution response.

Figure 1 shows the schematic of the overall ROM framework. To begin, the inputs to the model are the structural mode shapes of the geometry as well as the modal deformations at each time step throughout the simulation. The structural mode shapes are used in CFD simulations for both modal step inputs and correction factor calculation; the details of these runs will be described subsequently. However, since these structural mode shapes are known *a priori*, and the CFD runs only require knowledge of these structural mode shapes and not the per-iteration modal amplitudes, all CFD runs can be conducted up front, prior to model construction. Thus, once completed, the CFD calculations are taken out of the ROM loop, leaving linear convolution and nonlinear correction factor application as the two in-the-loop ROM features. The calculation of these two items takes orders of magnitude less time than the full CFD simulations, thus making the ROM much more computationally efficient than the calculation of the full-order solution. Finally, the outputs are

the time-accurate force and moment coefficients or the generalized forces.

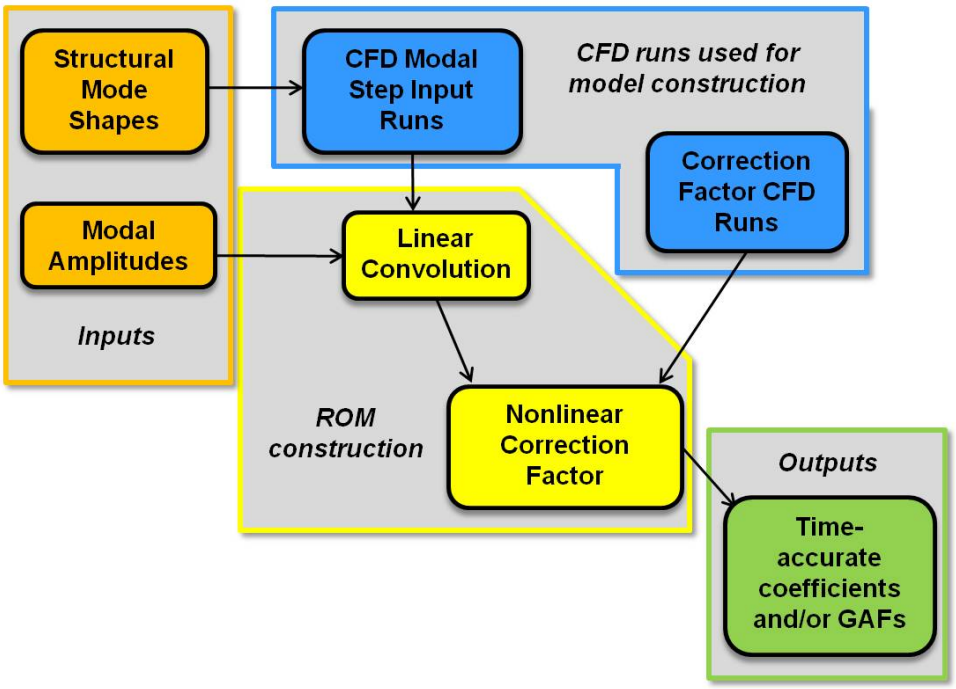


Figure 1. Overall ROM framework

A. Linear Convolution

The response of a linear system to an arbitrary input at time t can be found if the response of the system to a unit step ($H(t)$) or unit impulse ($h(t)$) function is known. In the continuous time form, the impulse and step input functions are written as functions of time as follows, for a case in which the step/impulse is applied at time t_0:

$$\begin{array}{ll} \textit{Impulse} & \textit{Unit Step} \\ u_{imp.}(t-t_0)=\infty,\ t=t_0 & u_{step}(t-t_0)=0,\ t<t_0 \\ u_{imp.}(t-t_0)=0,\ t\neq t_0 & u_{step}(t-t_0)=1,\ t\geq t_0 \end{array} \tag{2}$$

The response $y(t)$ due to an arbitrary input $f(t)$ is found through the use of convolution:[1,16]

$$y(t)=f(0)H(t)+\int_0^t \frac{df}{dt}(\tau)H(t-\tau)\,d\tau \tag{3}$$

The unit impulse is the derivative of the unit step, so integration by parts yields

$$y(t)=f(t)H(0)+\int_0^t f(\tau)h(t-\tau)\,d\tau \tag{4}$$

Equations 3 and 4 are the two forms of Duhamel's integral. In general, convolution can be thought of as a summation of scaled and shifted step/impulse responses.

Rather than using the continuous-time form of Duhamel's integral, the application of the convolution integral to a CFD code requires its discrete form.[1] The definitions are slightly different for the discrete case, where the input values are only defined for specified points in time. Thus, the impulse/step inputs are given as follows, with the input occurring at time step 1:

$$\begin{array}{ll} \textit{Impulse} & \textit{Unit Step} \\ u_{imp.}[n] = \frac{1}{\Delta t},\ n = 1 & u_{step}[n] = 0,\ n < 0 \\ u_{imp.}[n] = 0,\ n \neq 1 & u_{step}[n] = 1,\ n \geq 1 \end{array} \tag{5}$$

where the square brackets denote a value at a specified integer time step n. Equation 5 leads to the two forms of the discrete convolution integral:[3]

$$\begin{aligned} &\textit{Impulse:} \quad y[n] = h[0] + \textstyle\sum_{k=0}^{n} f[n]\, h[n-k]\, \Delta t \\ &\textit{Step:} \quad\ \ y[n] = f[0]\, H[n] + \textstyle\sum_{k=0}^{n} (u[n] - u[n-1])\, H[n-k] \end{aligned} \tag{6}$$

In this work, the step input is chosen over the impulse input for use in the convolution integral due to both ease of implementation into the CFD code as well as the quality of the response found. This improvement of results using the step over impulse input was already noted by Raveh.[15] Also, note that the system inputs considered here are modal deformations. Thus, whatever modal deformation is used for the step input is considered to have a scaled value of 1, the unit value. Because of this, for the rest of this paper, all modal input values will be given in multiples of the step input.

The step response and arbitrary modal motion are combined in Eq. 6 to calculate the uncorrected linear convolution ROM response y_{conv}. The next step is to consider the nonlinear correction to this linear ROM.

B. Correction Factor

If left as described above, the ROM will only work for flight conditions and input amplitudes very near to those used for the step input, as the responses do not in general scale linearly with oscillation amplitude. Consider the example from Ref. 13 of a 2-D half-diamond airfoil in hypersonic flight undergoing oscillations of the first bending mode with a maximum amplitude 40 times larger than the one used for the original identification of the step response. Figure 2 shows a sample drag coefficient comparison between the linear ROM y_{conv} and direct CFD result for this case. The response y_{conv} in this case is a qualitatively very poor representation of the actual CFD results, demonstrating the necessity of a nonlinear correction to it.

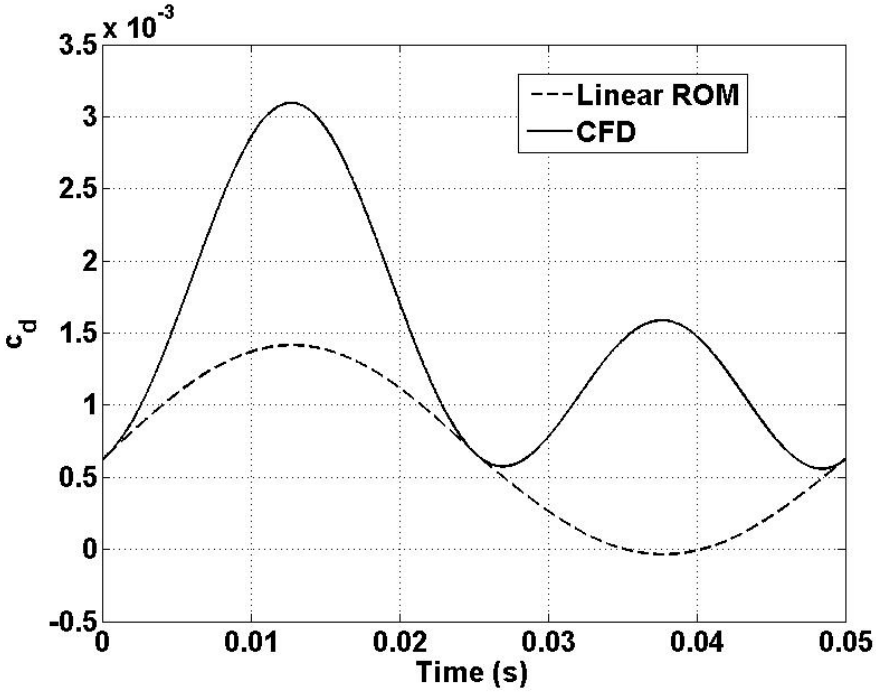


Figure 2. Linear ROM vs. CFD at large oscillation amplitude

To obtain the nonlinear corrected ROM response y_{corr}, a correction factor f_c is introduced. This quantity is defined as the ratio between the quasi-steady linear (y_{lin}) and nonlinear (y_{nonlin}) responses of a certain configuration due to the modal deformations and flight conditions at a particular instant in time, that is,

$$f_c \equiv \frac{y_{nonlin}}{y_{lin}} \tag{7}$$

In computing the correction factor value, the first challenge is to calculate y_{nonlin}. To do so, the input values of the modal amplitudes at a particular time step during a simulation are identified, and these inputs are applied to each mode shape simultaneously. Using CFD or some simplified model to run the simulation, the response is allowed to converge to some quasi-steady value, which is then taken to be y_{nonlin}. Next, y_{lin} is found by first determining the final, quasi-steady response value for each mode after a step input for that particular mode has been applied individually. Then, each one of these quasi-steady values is multiplied by the respective modal amplitude at that time step to find the individual linear modal responses. y_{lin} is then computed by summing these individual modal responses using superposition. These calculation procedures for y_{nonlin} and y_{lin} are illustrated in Figs. 3 and 4, respectively. y_{nonlin} and y_{lin} differ from y_{corr} (nonlinear ROM response) and y_{conv} (linear convolution ROM response), respectively, in that y_{corr} and y_{conv} involve the convolution integral in their calculation. Thus, the modal velocities, which enter the convolution integral as the time derivative of the arbitrary modal inputs, are included in the calculation of y_{conv} and y_{corr} but not in the calculation of y_{nonlin} and y_{lin}.

For a purely linear system, the correction factor value will be 1. In certain situations, the value for y_{lin} will be equal to or very close to zero, resulting in an f_c value approaching infinity. For these situations, the definition is modified by the addition of an offset term δ:

$$f_c = \frac{y_{nonlin} + \delta}{y_{lin} + \delta} \tag{8}$$

Note that δ is placed in the numerator as well as the denominator such that a linear system will still have a correction factor value of 1.

With this correction factor definition in place, the corrected ROM value y_{corr} is calculated by:

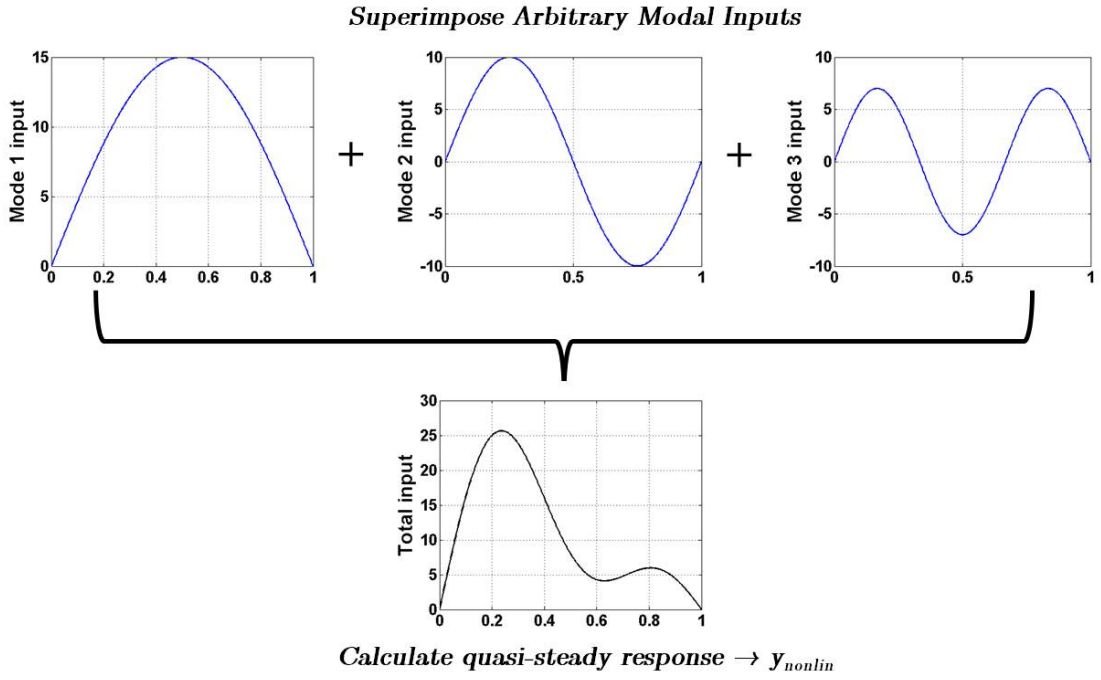


Figure 3. Superposition of inputs in the calculation of y_{nonlin}

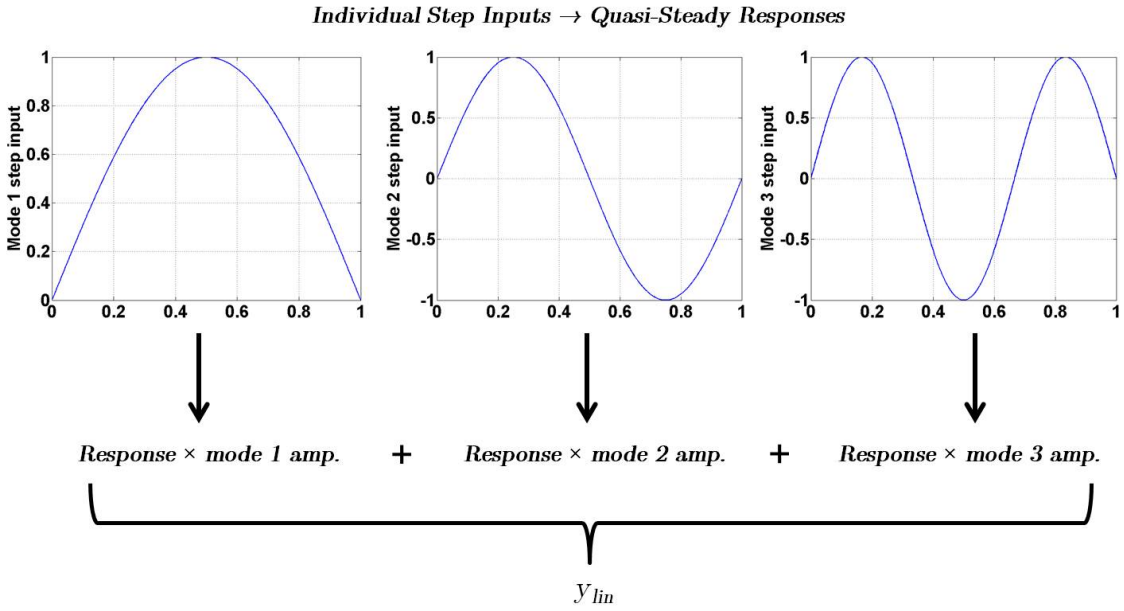


Figure 4. Superposition of responses in the calculation of y_{lin}

$$y_{corr} = (f_c)\, y_{conv} = \left(\frac{y_{nonlin}}{y_{lin}}\right) y_{conv} \tag{9}$$

This leads to the basic correction factor assumption, that the ratio of the quasi-steady response values at a particular time step t will be equal to the ratio of unsteady response values at that particular time step, namely:

$$\left.\frac{y_{nonlin}}{y_{lin}}\right|_t = \left.\frac{y_{corr}}{y_{conv}}\right|_t \tag{10}$$

Now that the correction factor has been defined, the challenge is to find its value over the entire parameter space being considered, which in this work consists of modal amplitudes and Mach number. Using CFD to directly calculate f_c at every point of interest would be prohibitive in terms of computational cost. To solve this problem, consider the difference between computer experiments and actual physical experiments. Each time a physical experiment is repeated, the result will not be exactly the same as the time before due to measurement and other inherent random errors. However, a computer experiment will give the exact same result each time it is completed, and thus each response value in a certain parameter space would be expected to be an exact value of the response quantity.

1. Kriging

Kriging[17] is a methodology that takes advantage of this lack of random error to create a representation of the response function based on the results at a certain number of sampling points. The kriging function predictor $\hat{y}(X)$ is a combination of a regression model and a random process $Z(X)$, which are the first and second terms, respectively, in the following equation:[18]

$$\hat{y}(X) = \sum_{j=1}^{k} \beta_j f_j(X) + Z(X) \tag{11}$$

For the regression model, $f_{j,j=1..k}(x)$ are the set of k regression functions, and β_j are the set of regression parameters. The random process has a mean zero and covariance of $\sigma^2\mathcal{R}$, where σ is the process variance and $\mathcal{R}$ is the correlation model. The goal of the kriging method is to minimize the mean squared error φ

of the predictor $\hat{y}$ over the parameter space, which is found by[18] the following equation, where $E[\,]$ denotes the covariance of a particular quantity:

$$\varphi(x) = E\left[\left(\hat{y}(x) - y(x)\right)^2\right] \tag{12}$$

If a linear predictor is assumed over the parameter space for the value of $\hat{y}(x)$, this quantity can be expressed as[18]

$$\hat{y}(x) = c^{\mathrm{T}} Y \tag{13}$$

where Y are the outputs from sampled values. The kriging methodology finds the best linear unbiased predictor c^{T} by minimizing the mean squared error prediction. The result of this minimization procedure is that the predictor can be written as[19]

$$\hat{\mathbf{y}}(\mathbf{x}) = \mathbf{f}^{\mathrm{T}}(\mathbf{x})\,\hat{\beta} + \mathbf{r}^{\mathrm{T}}(\mathbf{x})\,\mathbf{R}^{-1}\left(\mathbf{Y} - \mathbf{F}\hat{\beta}\right) \tag{14}$$

where $\mathbf{F}$ is the vector of f_j at the sampling points, $\mathbf{R}$ is the correlation function matrix, $\mathbf{r}(x)$ is the correlation between an unknown point x and the known sampling points, and $\hat{\beta}$ is the least squares predictor given by[19]

$$\hat{\beta} = \left(\mathbf{F}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\mathbf{F}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{Y} \tag{15}$$

In this research, the kriging methodology is implemented using the MATLAB Design and Analysis of Computer Experiments (DACE) toolbox's built-in functions.[18]

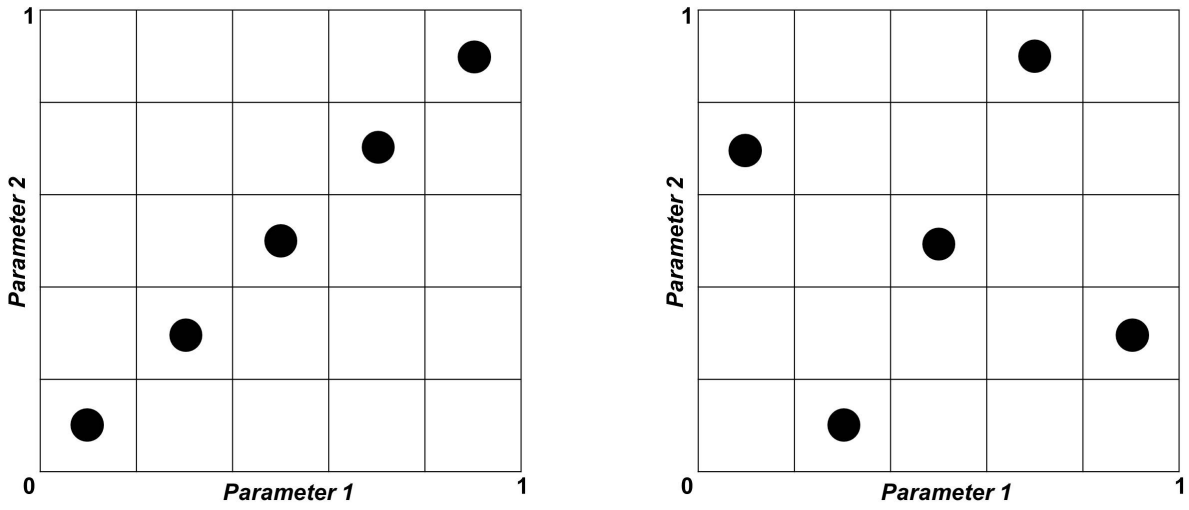
2. Latin Hypercube Sampling

In order to obtain the values to use for kriging surface construction, one must select appropriate sampling points within the parameter space. As the number of parameters increases in a particular problem, it becomes more difficult to conduct simulations pairing every value of one parameter with every value of all other parameters. For example, consider a problem with five separate parameters. Suppose that the range of each parameter is broken into ten intervals. In order to test each parameter value with all other parameter values, 10^5 trials would need to be conducted, which in many applications, including CFD simulations, is generally not feasible. Thus, it is important to be able to smartly sample the parameter space such that the behavior of the response function is known, but the overall number of trials to be run is not prohibitively high.

For this purpose, Latin hypercube sampling (LHS) is employed.[20] LHS works by first dividing up the range for each parameter into a user-defined number of intervals. Then, one sampling point is placed in each of the intervals for each parameter. Consider Fig. 5, which shows a sample parameter space consisting of two separate parameters divided into the intervals shown. Both Figs. 5(a) and 5(b) are examples of a potential Latin hypercube sampling configuration. However, in a strictly qualitative sense, it is obvious that Fig. 5(b) does a better job of "smearing" the sampling points more evenly throughout the parameter space. Thus, using any Latin hypercube design is not enough to guarantee a good sampling distribution, as it is important in many cases to sample points as evenly as possible. Because of this, LHS is furthered by the concept of orthogonal or nearly orthogonal Latin hypercube sampling,[21] which works to minimize the correlation among the various vectors of sampling point parameter values. In this research, the sampling point values are obtained through MATLAB's built-in `lhsdesign` command. Anywhere from $10,000$ to $100,000$ iterations of the command are run, and the sampling points used are selected from all the iterations using either the `maximin` option for the points with maximum minimum distance from each other in the parameter space or the `correlation` option for the vectors of input quantities to have the minimum correlation with each other.

C. Correction Factor CFD Runs

Much of the potential difficulty in calculating the correction factor at a certain point lies in the ability to calculate the quantity y_{nonlin} at that point. The quantity y_{lin} is calculated from the various individual modal step inputs, so nothing new needs to be computed for this term at each individual sampling point. However the quantity y_{nonlin} at a certain sampling point is calculated by inputting all modal input values simultaneously, requiring an individual CFD run at an individual sampling points. For some situations, these modal deformations are significantly larger than those used for the step inputs. This becomes a problem



(a) Non-evenly spread LHS points

(b) More evenly-spread LHS points

Figure 5. Example LHS sampling points

when the modal inputs become so large that the CFD code cannot input the deformations as step inputs without numerical issues arising in the obtained solution due to the resulting very large grid velocities (in many cases, the code will crash due to the large inputs). This problem can be solved by considering the fact that y_{nonlin} only relies on the final, quasi-steady response value after the desired inputs have been given. Thus, it does not matter if the inputs are given as steps or by gradually increasing the amplitude up to the final value. Because of this, to find y_{nonlin} for large amplitudes, the modal amplitudes are sinusoidally increased up to the final value and then leveled off. Once the response has reached a quasi-steady state, that value is used for the quantity y_{nonlin}.

D. Method of Segments

In certain instances, such as when one is trying to determine the correction factor sampling points to use for model construction or when a large number of sampling points is necessary, the use of simplified models to calculate correction factor values is warranted over full CFD simulations. In the transonic regime, the Method of Segments (MoS) has been developed in Ref. 14 to efficiently calculate correction factor values at locations throughout the parameter space without requiring a separate CFD run at each of the locations. The lift and drag coefficients calculated using this method were compared with the coefficients calculated using direct CFD simulation for 78 sampling points, and errors were found to be around 5% for both c_l and c_d, demonstrating the applicability of the method.[14] The basic idea of MoS is that, when the wing is in an elastically-deformed position, it can be approximated as a series of chordwise-rigid segments along the span which are at different angles of attack, as illustrated in Fig. 6.

Since the correction factor methodology relies on the quasi-steady coefficients after a certain modal deformation has been input, the lift, drag, and pitching moment at each of the chordwise segments along the span are found utilizing steady rigid CFD simulations conducted at varying angles of attack and Mach numbers. While the individual segments will also undergo a plunge motion in addition to pitching motion during elastic deformations, these plunge motions are neglected here due to the quasi-steadiness of the CFD solutions being found. The specific steps to the method, shown graphically in Fig. 7, are as follows:

1. Divide the wing into chordwise segments along the span, which are assumed to be rigid in the chordwise direction.

2. Conduct steady, rigid CFD runs throughout the parameter space, which consists of Mach number and angle of attack. The parameter space dimensionality will remain at two regardless of how many modes

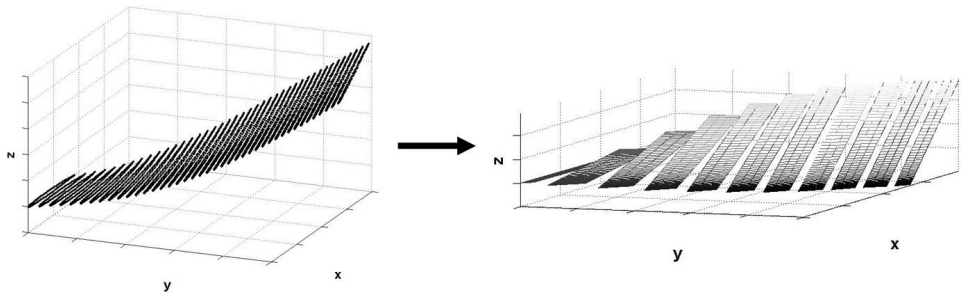


Figure 6. AGARD wing divided into segments

are being considered. Thus, the total number of runs remains relatively low, and steady runs are computationally cheaper than unsteady ones.

3. Track the lift and drag forces on each of the chordwise segments, taking into consideration the spanwise width of each segment. Construct separate kriging surfaces for the lift and drag forces at each of the segments.

4. For a certain wing deformation at a particular time step in a simulation, calculate the local angle of attack at each wing segment.

5. Pick the lift and drag forces off the kriging surfaces for each segment corresponding to the specific Mach number and local angle of attack; sum them together to find the lift and drag for the entire wing.

6. Calculate the coefficients for the wing. These values can then be used to formulate the correction factor for that particular set of parameters.

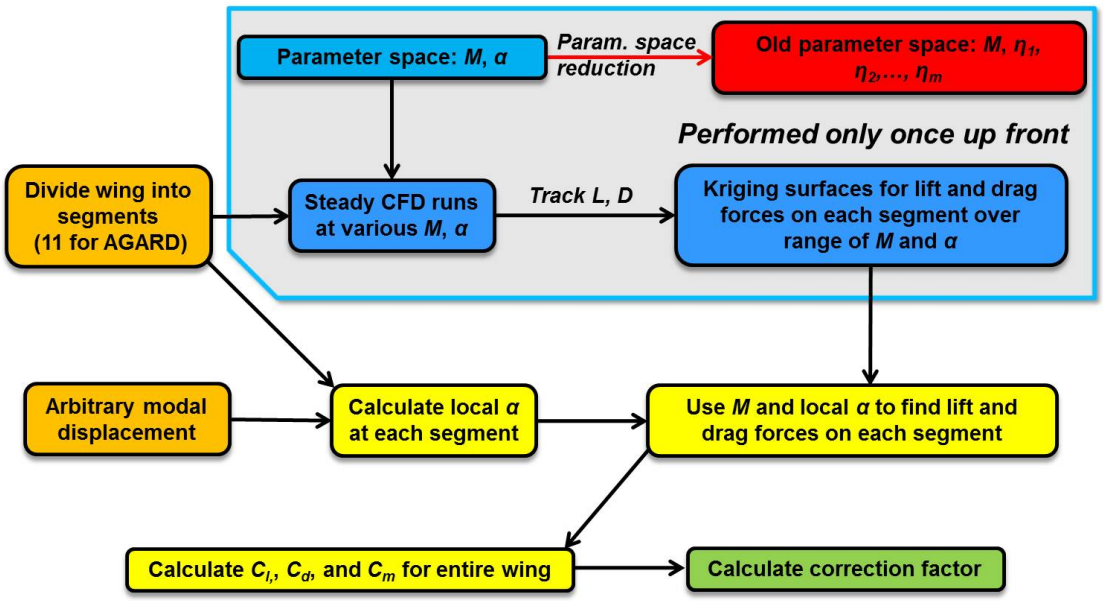


Figure 7. Method of segments process

III. Basic Numerical Example Definition and Setup

The CFD code used in this study is CFL3Dv6, developed at NASA Langley.[22] The code is capable of solving the Euler/Navier-Stokes equations for both steady and unsteady flows on two and three-dimensional structured grids and has mesh deformation capability, through which the modal inputs are given to the geometry. Response quantities tracked are the lift, drag, and moment coefficients, which are output by the code. All solutions presented here are Euler ones.

A. Geometry

For the simulations themselves, a CFD grid of the AGARD 445.6 wing[23] has been obtained from NASA Langley, where wind tunnel aeroelastic tests have been conducted on the wing. The wing has also been widely used for computational aeroelastic studies.[3,15] The computational model is a structured $65 \times 193 \times 41$ grid with the i direction being along the span, j direction along the chord, and k direction normal to the wing surface. Figure 8 shows the grid as well as a zoomed-in figure of the wing itself.

Oscillations of the first three elastic mode shapes of the wing are considered and are reproduced in Fig. 9. Note that, for each mode shape, the unit step input corresponds to a maximum wing deflection of just around 0.1% of the span. For the Method of Segments, this wing has been divided into 11 equally-sized chordwise-rigid segments.

B. Error Metrics

Two separate error metrics are used to judge the accuracy of the ROM compared with the CFD. The first metric, the L_1 error, is characterized by finding the mean absolute difference between the ROM and CFD results at each time step; it is normalized by the range spanned by the CFD results. For a simulation over N time steps, it yields:

$$L_1 \text{ error} = \frac{\frac{1}{N}\sum_{i=1}^{N}\left(\left|y_{ROM,i} - y_{CFD,i}\right|\right)}{\max\left(y_{CFD}\right) - \min\left(y_{CFD}\right)} \times 100\% \tag{16}$$

where $y_{ROM,i}$ and $y_{CFD,i}$ are the respective ROM and CFD response values found at time step i, and the denominator represents the difference between the maximum and minimum values found over all time steps of the ROM response.

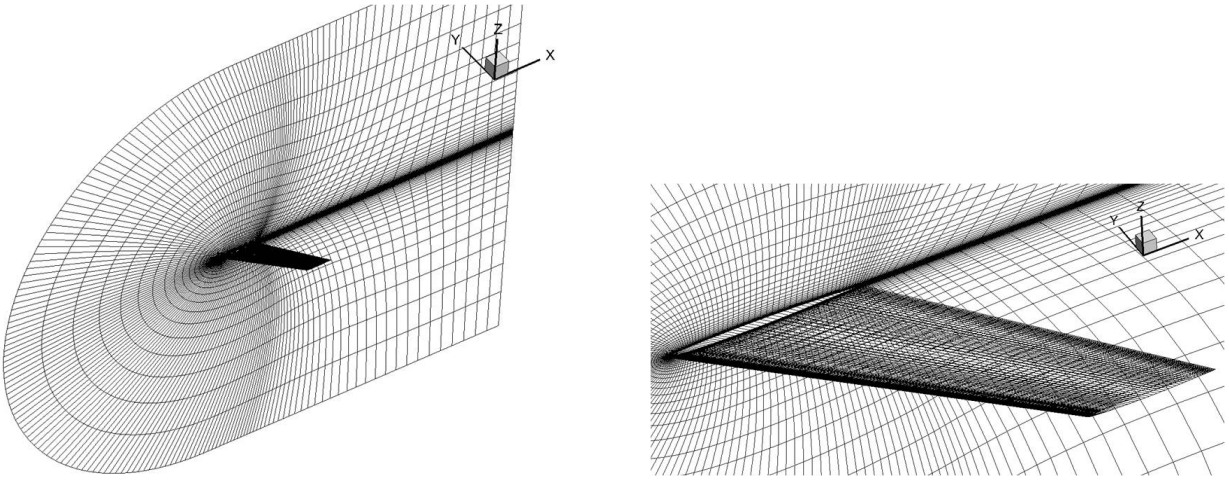
The second error metric is the L_∞ error, defined as

$$L_\infty \text{ error} = \frac{\max\left(\left|y_{ROM} - y_{CFD}\right|\right)}{\max\left(y_{CFD}\right) - \min\left(y_{CFD}\right)} \times 100\% \tag{17}$$

Rather than the mean value of the difference over all time steps, the L_∞ error finds the maximum ROM-CFD difference over all time steps and normalizes this quantity by the same range as in the L_1 error.

IV. ROM Testing

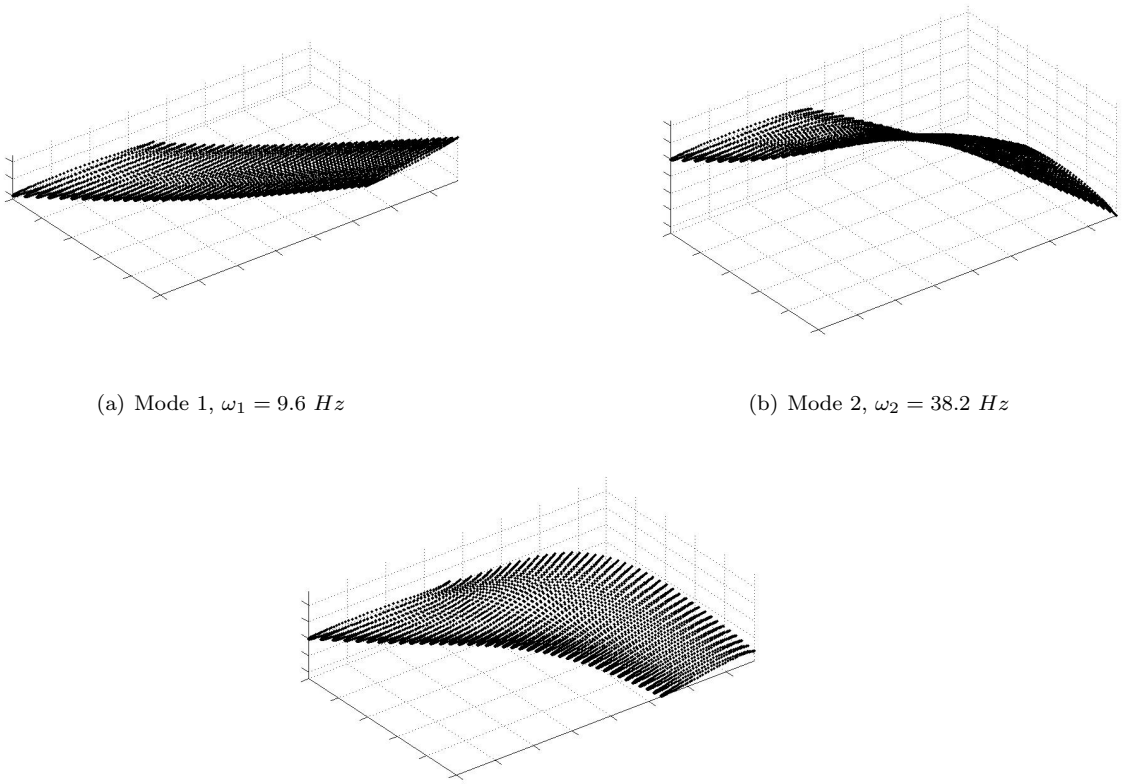
Ref. 14 gives some initial results of the ROM methodology's applicability to transonic regime by comparing ROM results, obtained through the use of the Method of Segments, to a few sample CFD simulations. However, to gain a fuller understanding of the ROM's performance, further testing is necessary and presented here. The first consideration is delve deeper into the accuracy of the Method of Segments by considering two separate ROMs, each with correction factor values calculated through the use of MoS, albeit with different philosophies. Then, the results of the two MoS ROMs are compared with a ROM in which the correction factor is calculated by way of a kriging surface constructed using individual sampling point values obtained from CFD simulations and not MoS. These comparisons are made using a total of 50 full-order CFD simulations with sinusoidal oscillations of the first three modes of the AGARD 445.6 wing described previously. For this work, the chosen Mach range to focus on is 0.9-1.0. Table 1 highlights the ranges of the various parameters used in these tests, with d_i being the amplitude of the i^{th} mode given in multiples of the step amplitude. Note that the frequency range was chosen such that the minimum value corresponds to $\omega_1 = 60.3\ rad/s$, and the maximum value corresponds to slightly over $5\omega_1$. All simulations are conducted at a rigid-body angle of attack of zero.



(a) AGARD grid

(b) AGARD wing close-up

Figure 8. AGARD 445.6 wing



(a) Mode 1, $\omega_1 = 9.6\ Hz$

(b) Mode 2, $\omega_2 = 38.2\ Hz$

(c) Mode 3, $\omega_3 = 48.2\ Hz$

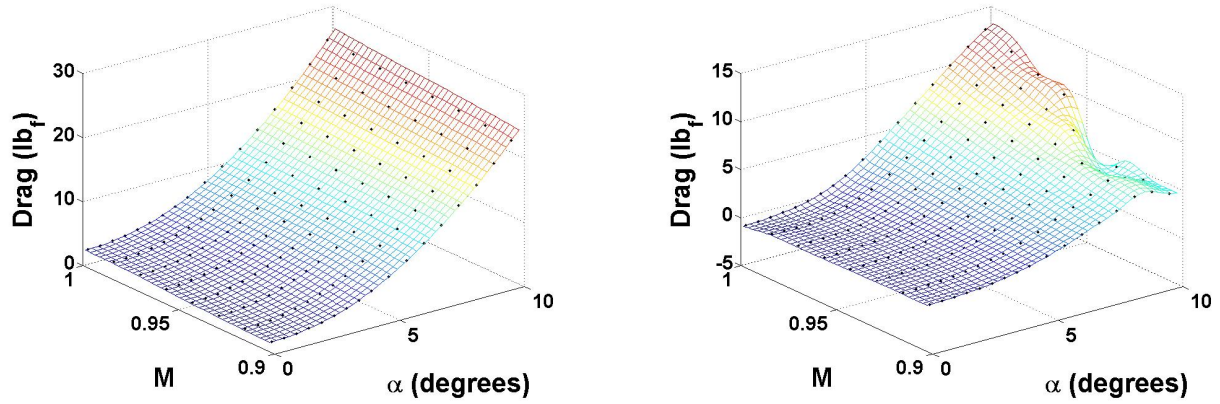
Figure 9. AGARD 445.6 wing mode shapes

Table 1. Parameter ranges for sinusoidal test cases

Parameter	Min	Max
M	0.9	1.0
d_1	-50	50
d_2	-35	35
d_3	-30	30
ω (rad/s)	60.3	350

A. Method of Segments ROM Construction

To construct the Method of Segments ROMs, steady CFD simulations are first conducted over the Mach number-α parameter space of the problem. A total of 152 points in a lattice-type of pattern is used for this purpose. Figure 10 shows the drag force kriging surfaces for the root (Fig. 10(a)) and tip (Fig. 10(b)) strips of the wing. For the tip segment, the surface is not smooth at angles of attack in the vicinity of 10°; this may potentially be due to computational issues with the CFD code at these higher angles of attack. However, this region does not play a significant role in the test results due to the fact that, of all instances of all strips at all time steps in the 50 test case simulations, under 0.1% of them occur with a given strip above 10° angle of attack. Also, in viscous flow, flow separation and other effects may be expected to be seen at these relatively high angles, which are not modeled in the inviscid simulations.



(a) Root strip

(b) Tip strip

Figure 10. Drag strip kriging surfaces

Next, two separate MoS-based ROMs are constructed. The first, denoted from here on as ROM A, takes advantage of the relative computational efficiency of the Method of Segments by directly calculating the MoS correction factor value at each time step throughout the simulation without the computation of any correction factor kriging surfaces. Thus, this ROM eliminates any ROM uncertainties based on the kriging surface fit of the data. The second MoS-based ROM, denoted ROM B, is created by first generating an initial number of sampling points in the Mach number-modal amplitude parameter space via Latin hypercube sampling. Using MoS, the correction factor at each of these points is found, and a kriging surface is computed. Next, the location of surface's maximum error is calculated through the use of the built-in MATLAB `predictor` function.[18] Then, an additional point is placed at this location, the surface is re-computed, and the maximum error is re-calculated. The whole process is repeated until a desired stopping criterion is met. For this ROM, a total of 1,000 sampling points are used in this process. Additionally, 11 more sampling points are placed at zero amplitudes and varying Mach numbers to reduce errors seen at those locations. ROM A can be thought of as the limit of the construction process of ROM B if infinite sampling points are used. Table 2 summarizes

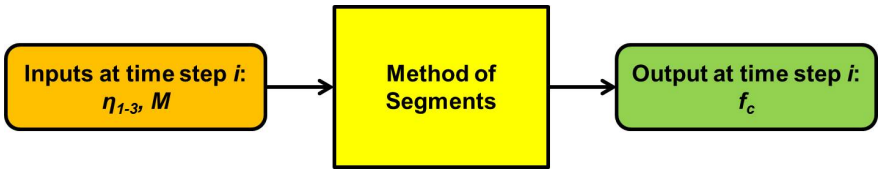


Figure 11. Schematic for ROM A

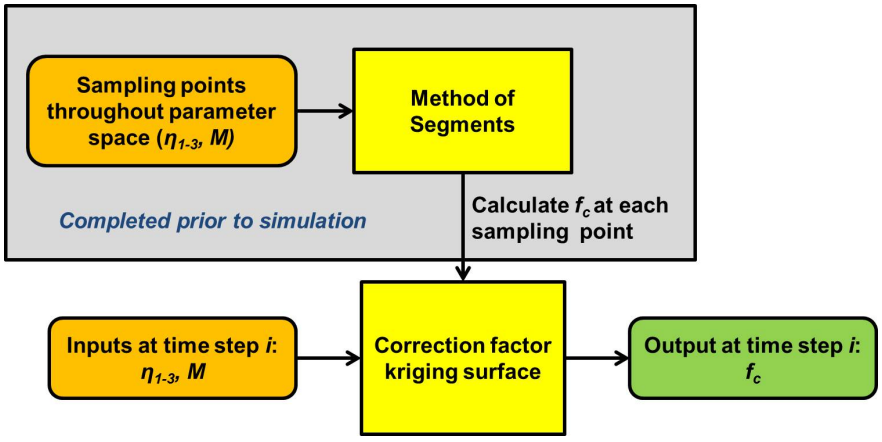


Figure 12. Schematic for ROM B

the key aspects of these ROMs as well as the CFD-based ROM described subsequently, and Figs. 11and 12 show schematics highlighting the differences between ROMs A and B.

B. CFD ROM Construction

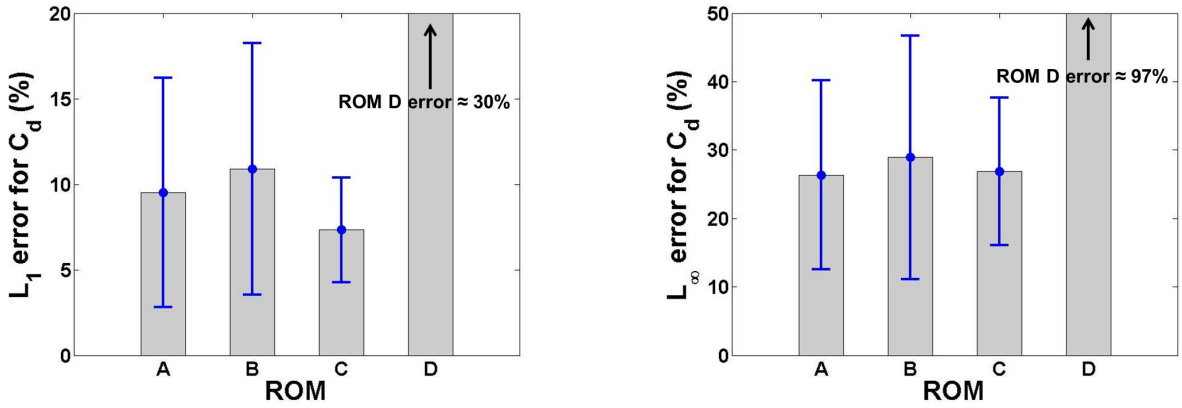
The next ROM considered here, ROM C, is calculated using the same sampling points as in ROM B, but the correction factor values at each point are computed using individual direct CFD simulations rather than the Method of Segments. Note that, rather than the 1,011 points used in ROM B, ROM C consists of only 990 sampling points. This is due to CFD code limitations, as some of the runs at higher modal deformations ran into numerical issues and thus did not produce results. Finally, ROM D consists of the linear ROM y_{conv} found using linear convolution and superposition.

Table 2. ROM variations

ROM	Sampling points	f_c calc.	Comment
A	N/A	MoS	Direct calculation of f_c at each time step using MoS
B	1,011	MoS/kriging	Points placed at location of maximum surface error
C	990	CFD/kriging	Mostly the same sampling points as ROM B
D	N/A	N/A	Linear ROM y_{conv}

C. Multi-modal Testing Results

Figure 13 shows both the L_1 and L_∞ drag coefficient error results for each of the four ROMs listed in Table 2. The bars show the mean value of each of the error metrics over all 50 runs, while the error bars show the standard deviations.



(a) L_1 (b) L_∞

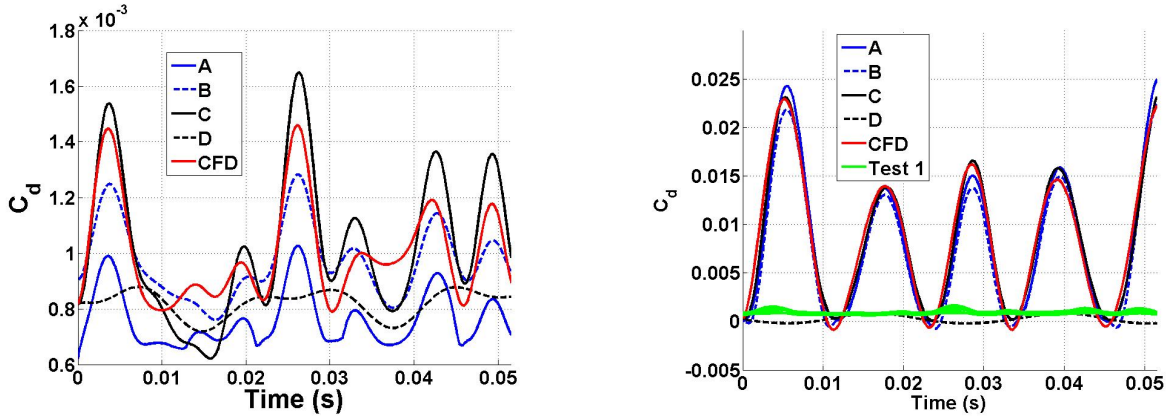
Figure 13. C_d **error results over 50 test cases**

For the two Method of Segments ROMs, ROM A shows a slight improvement over ROM B for each of the error metrics, decreasing the mean L_1 error from around 11% to 9.5% and the L_∞ error from around 29% to 26%. This is to be expected due to the fact that ROM A calculates the MoS correction factor value at each time step, while ROM B obtains the correction factor value from a previously-constructed kriging surface. Next, ROM D clearly performs the worst, with L_1 and L_∞ errors of around 30% and 97%, respectively. This demonstrates that the linear ROM is not suitable to model the drag coefficient. Finally, ROM C performs the best in terms of the L_1 metric and is comparable to the two MoS ROMs in terms of the L_∞ metric, showing slightly less error than ROM B but slightly higher error than ROM A. For a better illustration of these comparisons, the ROM and CFD results for a number of specific test cases are shown in Figs. 14 and 15; the parameters and errors for these runs are listed in Table 3.

Table 3. Test Case Parameters and Errors

Test	M	d_1	d_2	d_3	ω_1	ω_2	ω_3	ROM A	ROM B	ROM C
						(rad/s)		C_d errors: L_1/L_∞		
1	0.98	-11.1	3.30	-4.20	109	289	541	36.2/68.4	12.7/30.3	15.9/35.0
2	0.91	25.2	-29.2	5.80	158	274	492	5.11/17.3	6.26/22.9	3.63/12.1
3	0.94	28.4	-9.00	-22.4	137	269	557	9.90/29.1	12.0/30.9	6.14/25.8
4	0.96	13.4	-23.7	21.3	180	245	379	3.77/14.0	4.58/16.1	6.92/29.3

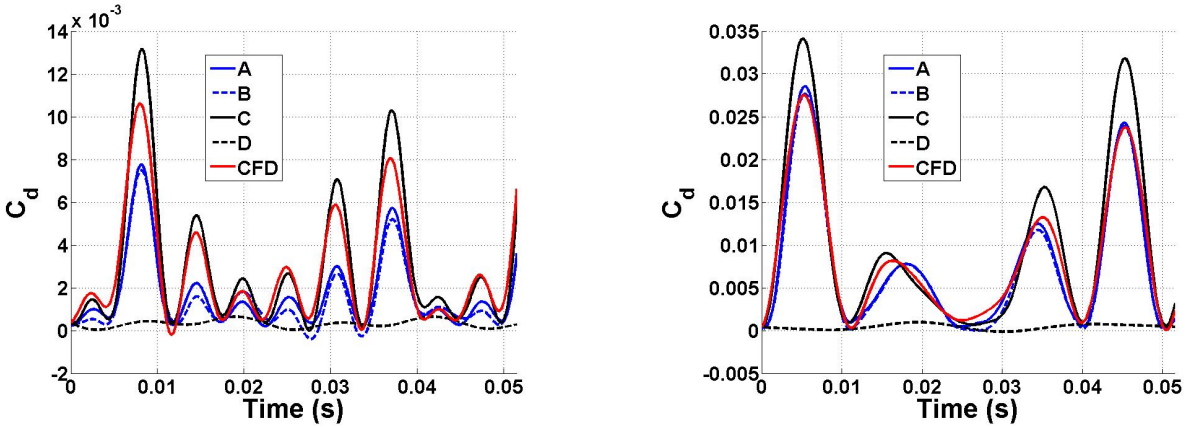
In general, graphically speaking, two main sources of error can be seen. Figure 14(a) displays the results for Test 1, which has the highest L_1 error for ROM C over all runs. Qualitatively, relatively large discrepancies can be seen between each of the ROMs and the CFD results. However, when looking at the total range spanned by the drag coefficient response value, it is relatively small. This is further illustrated in Fig. 14(b), which shows the ROM-CFD comparisons for Test 2, which has some of the smallest error values out of all test cases. In addition to the ROMs for that test case, the values from Test 1 are superimposed on the plot with the green lines. As can be seen, though the errors for Test 1 are larger than those for Test 2, the range spanned by the response of Test 2 is much larger. This shows that some of the large error values are due to small ranges spanned by the response quantity, resulting in small denominators for the error metric equations (Eqs. 16 and 17) and hence larger error values. Note that, for Test 1, the unexpected result of ROM A having a larger error than ROM B is observed. This appears to be the result of a slight DC-type of offset introduced by ROM A for this particular case which has been magnified due to the small ranges of coefficient values spanned in the simulation and is the subject of ongoing investigation.



(a) Test 1 (b) Test 2

Figure 14. Example test cases, large and small C_d errors

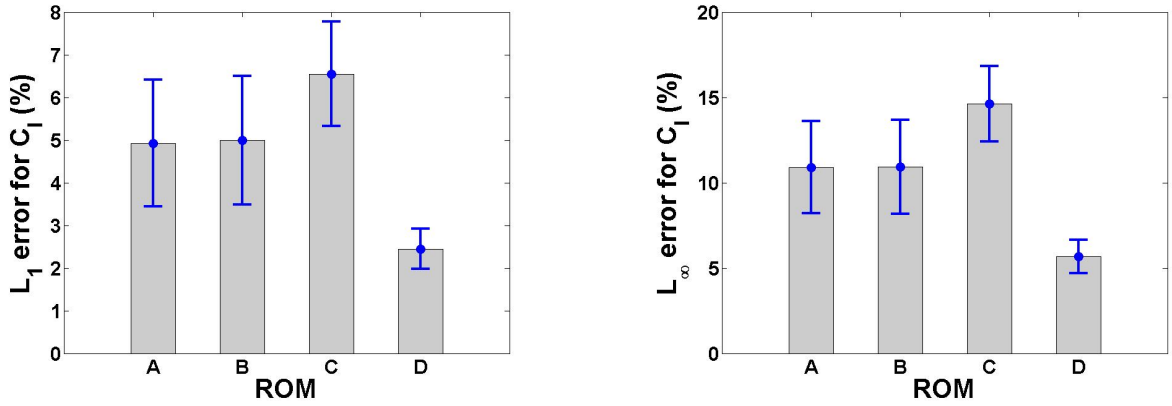
For the second source of error, consider Fig. 15, which shows test cases having error values right around the mean. In these cases, the largest source of error appears to be amplitude discrepancies between the ROM and CFD results. In many situations, the predictions from ROM C tend to over-predict peak drag coefficient values, resulting in some error, while the peak comparisons for the MoS-based ROMs vary.



(a) Test 3 (b) Test 4

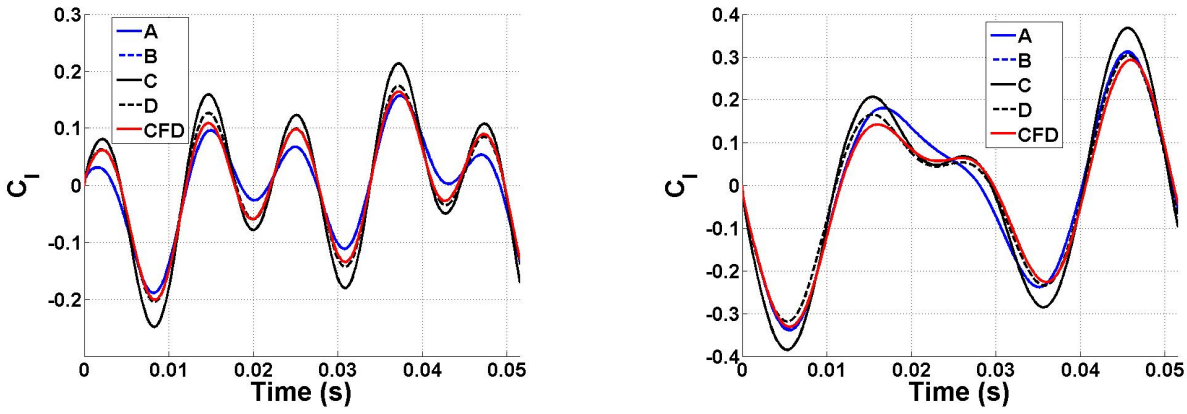
Figure 15. Example test cases, mean C_d errors

In addition to the drag coefficient, lift coefficient results for these same test cases are found as well, and the mean errors and standard deviations can be found in Fig. 16. These results are strikingly different than those found for the drag coefficient. The best agreement is found between ROM D (linear) and the CFD results, and each of the correction factor ROMs give slightly higher errors. This suggests that, for these test cases, the linear ROM is sufficient to model the lift response. Figure 17 shows the ROM-CFD comparisons for test cases 3 and 4 from Table 3. One interesting item to note is the improvement of results with the purely linear ROM over the nonlinear ROMs. For ROM C, the introduction of the correction factor generally results in a slight over-prediction of peak lift coefficient values, resulting in larger error values; the reason for this is the subject of an ongoing investigation. The results for the pitching moment are found to be similar in nature to the lift coefficient results in that ROM D is a good predictor of the response.



(a) L_1 (b) L_∞

Figure 16. C_l **error results over 50 test cases**



(a) Test 3 (b) Test 4

Figure 17. Example test cases, C_l **errors**

V. ROM Error Investigations

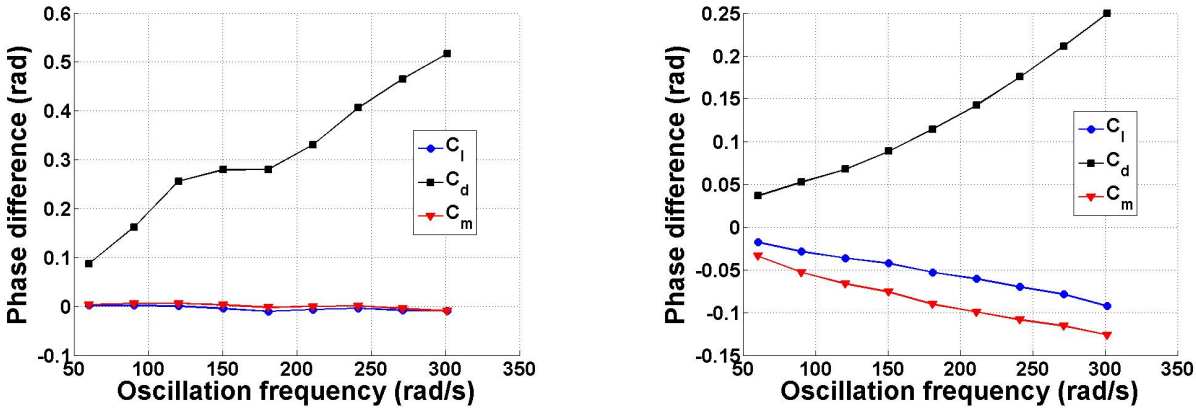
Two separate studies are conducted and presented here in order to help characterize the applicability and error boundaries of the ROM methodology. First, a phase shift between the ROM and CFD results seen especially at higher oscillation frequencies is quantified. Second, the effect of system memory on ROM error is looked into by considering the ROM's application to the nonlinear Riccati differential equation.

A. Phase Shift Study

Previous studies[14,24] have shown that a phase shift is often seen between the ROM and CFD results as the oscillation frequency increases. The study presented here looks to quantify this phase shift. To do so, CFD simulations are conducted with constant oscillation amplitude (first mode only) but increasing oscillation frequency; the parameters for these runs are summarized in Table 4. For each of these runs, the Fast Fourier Transform[25] is computed for both the ROM and CFD results, and the phase difference between the two responses is calculated. Figure 18 shows the lift, drag, and moment coefficient results for two separate series of results conducted at Mach 0.9 and Mach 1.1. Note that, for these tests, a single-mode ROM is used in which the correction factor is calculated through the use of a kriging surface with sampling points computed using direct CFD simulations.

Table 4. ROM Phase Shift Test Parameters

Test	M	d_1	ω_{min} (rad/s)	ω_{max} (rad/s)
1	0.9	20	60.3	$5\omega_{min}$=301
2	1.1	20	60.3	$5\omega_{min}$=301



(a) Mach 0.9

(b) Mach 1.1

Figure 18. ROM-CFD phase shift

For each of the two sets of results, the drag coefficient shows a greater phase difference over the range of frequencies than the lift and moment coefficients, though this increase is more pronounced for the Mach 0.9 tests. Because of this overall phase difference increase with frequency, when looking at a specific test case, it is recommended to run a sample simulation at the highest frequency expected to be encountered and evaluate how the error fits in with the error tolerances for the problem.

B. System Memory Study

Rather than being valid for only CFD-based problems, this ROM methodology can be applied to other sets of nonlinear differential equations as well. In doing so, one potential limiting factor is the length of the

memory of the system being modeled. The system's memory is characterized by looking at how many time steps after some sort of perturbation has been given that the effects of that perturbation will be felt. To investigate this, the Riccati equation is chosen as the model equation to be utilized, and it is given by:

$$\dot{y} + \alpha y + \epsilon y^2 = x\,(t) \tag{18}$$

where the input is the circuit voltage $x\,(t)$, and the output is the current $y\,(t)$; α and ϵ are resistance parameters. This equation describes a nonlinear circuit and has been used previously as a model equation for Volterra series-type of ROM analyses.[1]

In this case, the system's memory is measured by how long it takes the step response to reach 95% of its final, quasi-steady value given a set of parameters. This quantity, denoted T_{95}, is shown graphically in Fig. 19 and is measured in terms of the number of time steps to reach the 95% value. In order to alter the system's memory, the quantity α is changed for each run. For the ROM results, the correction factor is directly calculated at each time step throughout the course of the run, and the truth model used is the direct solution of the differential equation. For each test run, the input voltage $x\,(t)$ is given as follows,

$$\begin{aligned} x\,(t) &= A\,(1 - cos 2\pi f t) \\ A &= 25 \\ f &= 0.1 \text{ Hz} \end{aligned} \tag{19}$$

where A is the input amplitude given in multiples of the step input. $\epsilon = 0.01$ is used for each run. Figure 20 shows the ROM errors change both with increasing system memory (Fig. 20(a)) and α (Fig. 20(b)). As can be seen, an increase of system memory length adversely affects the accuracy of the ROM, with L_1 errors increasing to over 80% for the largest values of T_{95}. These results show that, when applying the ROM methodology to other Mach regimes or flight conditions, it may be important to investigate the memory length of the new system in comparison with other CFD systems for which the ROM applicability has previously been determined.

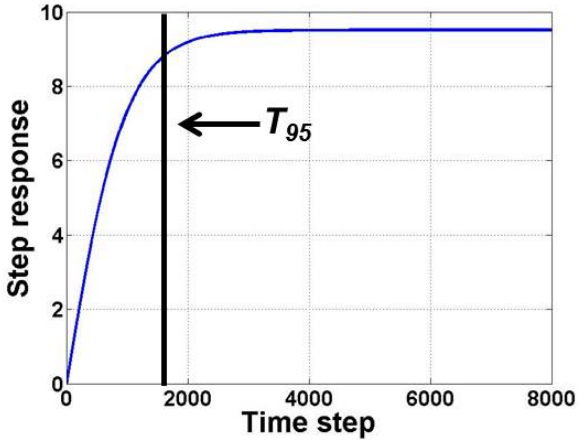
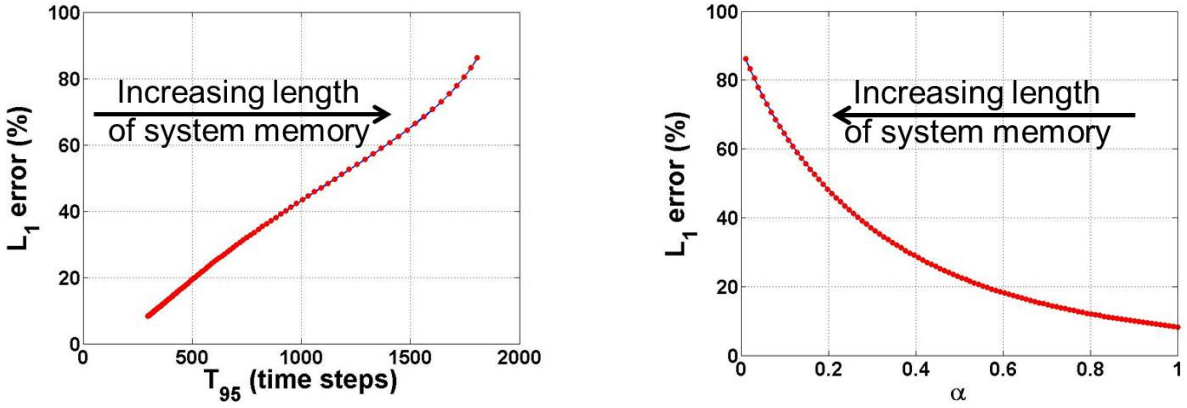


Figure 19. T_{95} **calculation**

VI. Concluding Remarks

The application of a reduced-order modeling methodology combining linear convolution with a nonlinear correction factor to the transonic regime has been investigated through the use of a multitude of full-order CFD test cases. These test cases are compared to ROMs with correction factors computed using both full CFD simulations as well as the simplified Method of Segments model. Further investigations have looked into a phase shift seen at higher oscillation frequencies as well as the effect of a system's memory to the ROM applicability. The major conclusions are as follows:



(a) Errors as function of T_{95}

(b) Errors as function of α

Figure 20. Effect of system memory on ROM errors

- The same general mathematical ROM form utilizing linear convolution along with a nonlinear correction factor, namely

$$y_{corr} = f_c y_{conv}$$

  has now been applied across multiple Mach regimes. To calculate the correction factor, direct CFD simulations can be used throughout all Mach regimes, or various regime-dependent simplified models may be used as well.

- Of the three ROMs constructed (two based on the Method of Segments for correction factor calculation and one on direct CFD results), each agreed relatively well with the CFD results for most cases. For drag coefficient results, the calculation of correction factor values using direct CFD simulations did show an improvement of the L_1 error over the MoS ROMs, though the reduction is small enough to allow MoS to be a viable alternative to a high number of CFD runs for ROM construction.

- The lift coefficient results demonstrated linear behavior, thus the linear ROM is able to accurately predict the response.

- The ROM-CFD phase shift does increase with oscillation frequency for single mode oscillation test cases. The reduction of these errors is an area of ongoing study.

- The accuracy of the ROM methodology seems to degrade with increased system memory, as shown by the application of the ROM to the solution of the nonlinear Riccati equation.

Acknowledgments

Funds for the Michigan-AFRL Collaborative Center in Control Science (MACCCS) were made available from the Air Force Research Laboratory/Air Vehicles Directorate grant number FA 8650-07-2-3744 (program manager Michael Bolender). Funds were also made available through NASA award NNX08AB32A (program managers Don Soloway and Jorge Bardina) along with computer resources from the NASA Advanced Supercomputing Division. The authors would like to thank Walt Silva from NASA Langley for providing the computational grid as well as his assistance in the computational studies.

References

[1]Silva, W., "Discrete-Time Linear and Nonlinear Aerodynamic Impulse Responses for Efficient CFD Analyses," Ph.D. dissertation, College of William & Mary, December 1997.

[2]Silva, W., "Simultaneous Excitation of Multiple-Input/Multiple-Output CFD-Based Unsteady Aerodynamic Systems," *Journal of Aircraft*, Vol. 45, No. 4, July-August 2008, pp. 1267–1274.

[3]Raveh, D., Levy, Y., and Karpel, M., "Efficient Aeroelastic Analysis Using Computational Unsteady Aerodynamics," *Journal of Aircraft*, Vol. 38, No. 3, May-June 2001, pp. 547–556.

[4]Silva, W., "Recent Enhancements to the Development of CFD-Based Aeroelastic Reduced-Order Models," *Proceedings of the 48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, AIAA Paper No. 2007-2051, April 2007.

[5]Glaz, B., Liu, L., and Friedmann, P., "Reduced-Order Nonlinear Unsteady Aerodynamic Modeling Using a Surrogate-Based Recurrance Framework," *AIAA Journal*, Vol. 48, No. 10, 2010, pp. 2418–2429.

[6]Glaz, B., Liu, L., Friedmann, P., Bain, J., and Sankar, L., "A Surrogate Based Approach to Reduced-Order Dynamic Stall Modeling," *Proceedings of the 51st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, AIAA Paper No. 2010-3042, April 2010.

[7]Glaz, B., Liu, L., Friedmann, P., Cajigas, J., Bain, J., and Sankar, L., "Reduced-Order Dynamic Stall Modeling With Swept Flow Effects Using a Surrogate-Based Recurrence Framework," *Proceedings of the 2011 International Forum of Aeroelasticity and Structural Dynamics*, IFASD-2011-033, June 2011.

[8]Crowell, A. and McNamara, J., "Model Reduction of Computational Aerothermodynamics for Hypersonic Aerothermoelasticity," *AIAA Journal*, Vol. 50, No. 1, January 2012, pp. 74–84.

[9]Lind, R., Prazenica, R., Brenner, M., and Baldelli, D., "Identifying Parameter-Dependent Volterra Kernels to Predict Aeroelastic Instabilities," *AIAA Journal*, Vol. 43, No. 12, December 2005, pp. 2496–2502.

[10]Baldelli, D., Zeng, J., Lind, R., and Harris, C., "Flutter-Prediction Tool for Flight-Test-Based Aeroelastic Parameter-Varying Models," *Journal of Guidance, Control, and Dynamics*, Vol. 32, No. 1, January-February 2009, pp. 158–171.

[11]Prazenica, R., Reisenthel, P., Kurdila, A., and Brenner, M., "Volterra Kernel Extrapolation for Modeling Nonlinear Aeroelastic Systems at Novel Flight Conditions," *Journal of Aircraft*, Vol. 44, No. 1, January-February 2007, pp. 149–162.

[12]Omran, A. and Newman, B., "Piecewise Global Volterra Nonlinear Modeling and Characterization for Aircraft Dynamics," *Journal of Guidance, Control, and Dynamics*, Vol. 32, No. 3, May-June 2009, pp. 749–759.

[13]Skujins, T. and Cesnik, C., "Reduced-Order Modeling of Hypersonic Vehicle Unsteady Aerodynamics," *Proceedings of the AIAA Atmospheric Flight Mechanics Conference*, AIAA Paper No. 2010-8127, August 2010.

[14]Skujins, T. and Cesnik, C., "On the Applicability of an Unsteady Aerodynamic ROM to the Transonic Regime," *Proceedings of the AIAA Atmospheric Flight Mechanics Conference*, AIAA Paper No. 2011-6525, August 2011.

[15]Raveh, D., "Reduced-Order Models for Nonlinear Unsteady Aerodynamics," *AIAA Journal*, Vol. 39, No. 8, August 2001, pp. 1417–1429.

[16]Fung, Y., *An Introduction to the Theory of Aeroelasticity*, Dover Publications, Mineola, New York, 1993.

[17]Matheron, G., "Principles of Geostatistics," *Economic Geology*, Vol. 58, 1963, pp. 1246–1266.

[18]Lophaven, S., Nielsen, H., and Sondergaard, J., "DACE: A MATLAB Kriging Toolbox, Version 2.0," Tech. Rep. IMM-TR-2002-12, Informatics and Mathematical Modelling, Denmark, August 1, 2002.

[19]Martin, J. and Simpson, T., "Use of Kriging Models to Approximate Deterministic Computer Models," *AIAA Journal*, Vol. 43, No. 4, April 2005, pp. 853–863.

[20]McKay, M., Beckman, R., and Conover, W., "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics*, Vol. 21, No. 2, May 1979, pp. 239–245.

[21]Cioppa, T. and Lucas, T., "Efficient Nearly Orthogonal and Space-Filling Latin Hypercubes," *Technometrics*, Vol. 49, No. 1, February 2007, pp. 45–55.

[22]Rumsey, C. and Biedron, R., "CFL3D User's Manual, Version 5.0, Second Edition," Tech. Rep. Hampton, VA, NASA Langley Research Center, September 1997 (last updated August 2009).

[23]Yates, E., "AGARD Standard Aeroelastic Configurations for Dynamic Response. Candidate Configuration 1.-wing 445.6," Tech. Rep. NASA TM-100492, NASA Langley Research Center, 1987.

[24]Skujins, T. and Cesnik, C., "Reduced-Order Modeling of Hypersonic Unsteady Aerodynamics Due to Multi-Modal Oscillations," *Proceedings of the 17th AIAA International Space Planes and Hypersonic Systems and Technologies Conference*, AIAA Paper No. 2011-2341, April 2011.

[25]Brigham, E., *The Fast Fourier Transform and Its Applications*, Prentice Hall, Englewood Cliffs, New Jersey, 1988.