View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by NASA Technical Reports Server

Uncertainty Modeling for Robustness Analysis of Aircraft Control Upset Prevention and Recovery Systems

Christine M. Belcastro* and Thuan H. Khong†
NASA Langley Research Center, Hampton, Virginia, 23661

Jong-Yeob Shin‡
National Institute of Aerospace, Hampton, Virginia, 23666

Harry Kwatny§ and Bor-Chin Chang**
Drexel University, Philadelphia, Pennsylvania, 19104

and

Gary J. Balas††
University of Minnesota, Minneapolis, Minnesota, 55455

Formal robustness analysis of aircraft control upset prevention and recovery systems could play an important role in their validation and ultimate certification. Such systems (developed for failure detection, identification, and reconfiguration, as well as upset recovery) need to be evaluated over broad regions of the flight envelope and under extreme flight conditions, and should include various sources of uncertainty. However, formulation of linear fractional transformation (LFT) models for representing system uncertainty can be very difficult for complex parameter-dependent systems. This paper describes a preliminary LFT modeling software tool which uses a matrix-based computational approach that can be directly applied to parametric uncertainty problems involving multivariate matrix polynomial dependencies. Several examples are presented (including an F-16 at an extreme flight condition, a missile model, and a generic example with numerous cross-product terms), and comparisons are given with other LFT modeling tools that are currently available. The LFT modeling method and preliminary software tool presented in this paper are shown to compare favorably with these methods.

1. Introduction

Aircraft loss-of-control accidents comprise the largest and most fatal aircraft accident category across all civil transport classes, and can result from a large array of causal and contributing factors (e.g., system and component failures, control system impairment or damage, inclement weather, inappropriate pilot inputs, etc.) occurring either individually or in combination. Research into the characterization of the aircraft loss-of-control phenomenon as well as loss-of-control prevention and recovery system technologies is being conducted by NASA as part of its Aviation Safety and Security Program (AvSSP). As shown in Figure 1, loss-of-control events can involve flight beyond normal operating conditions. Moreover, these conditions are not well modeled in current transport simulations. Validation of both the mathematical models and the systems technologies for loss-of-control conditions is therefore highly nontrivial.

* Senior Research Engineer, Dynamic Systems and Control Branch, MS308, AIAA Senior Member.
† Research Engineer, Dynamic Systems and Control Branch, MS308, AIAA Member.
‡ Staff Scientist, Robust Control, MS 930, AIAA Member.
§ Professor, Department of Mechanical Engineering & Mechanics, AIAA Member.
** Professor, Department of Mechanical Engineering & Mechanics, AIAA Member.
†† Professor, Department of Aerospace Engineering & Mechanics, AIAA Associate Fellow.

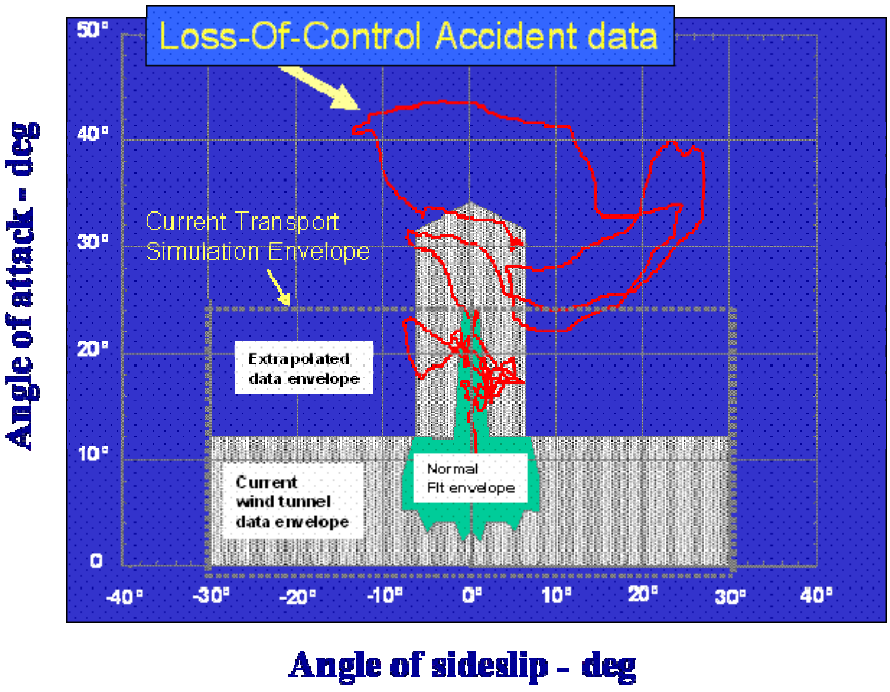

Figure 1. Current Transport Simulation and Loss-of-Control Accident Characteristics Relative to Angles of Attack and Sideslip

Certification of loss-of-control prevention and recovery systems (including failure detection, identification, and reconfiguration as well as upset recovery subsystems) for aircraft will require a comprehensive validation process (integrating analysis, simulation, and experimental methods) to ensure the safety and reliability of these systems. Robustness analysis for systems with structured uncertainty could play an important role in this process. Robustness is a key issue in the performance of failure detection and accommodation systems. Failure detection schemes can experience performance difficulties (such as false alarms) due to system uncertainties. Robustness of the control system can mask faults and failures and make the detection problem more difficult. It is fairly common for integration of failure detection and accommodation systems to be problematic if they're designed separately. Robustness analysis can also identify worst-case combinations of uncertainties, faults and failures for use in guided Monte Carlo simulation and/or experimental studies, and could provide risk mitigation for high-risk flight testing. Such testing will be conducted utilizing a dynamically scaled transport aircraft that has been developed at the NASA Langley Research Center as part of the Airborne Subscale Transport Aircraft Research (AirSTAR) Testbed (see References [1] and [2]). Robustness to nonlinear parameter variations over the flight envelope and at extreme flight conditions must also be considered. Reference [3] provides an excellent treatment of applying robustness analysis methods to the clearance of flight control laws, and reference [4] provides a robustness analysis framework for failure detection and accommodation systems.

Analytical robust control methods, such as the structured singular value (μ, see References [3] – [4]), require the formulation of a linear fractional transformation (LFT) model of the uncertain system, as shown in Figure 2.

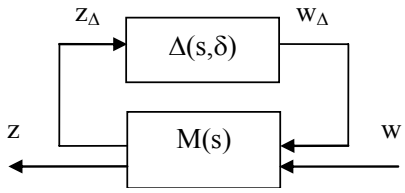

Figure 2. Block Diagram of LFT Model

Formulation of the LFT model can be extremely difficult and time consuming, especially for aircraft problems involving parametric uncertainties (see References [3] – [4] and [7] – [20]). In fact, the difficulty in formulating the

uncertainty model in LFT form has been a key impediment to performing robustness analyses for these systems. This paper presents a numerical matrix-based modeling method and preliminary software tool for computing LFT models from a linear parameter varying (LPV) model of the system. Section 2 summarizes the matrix-based computational approach and a preliminary software tool that have been developed for obtaining LFT models for complex systems involving parametric uncertainties, as presented in Reference [4]. Section 3 discusses the development of LPV and LFT models for an aircraft under extreme flight conditions and provides a detailed aircraft example as well as several additional examples. Section 4, provides a comparison to other LFT modeling tools that are currently available. Section 5 presents some concluding remarks.

2. Numerical Parametric LFT Modeling Approach

2.1 Numerical Matrix-Based LFT Modeling Method

The LFT model to be solved in this section is depicted in Figure 3.

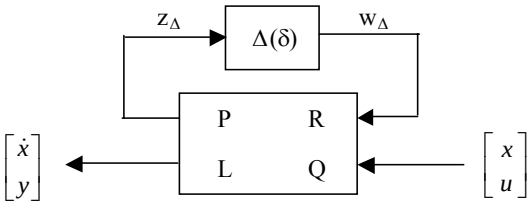

Figure 3. Block Diagram of LFT Modeling Problem

The matrix $\Delta(\delta)$ contains the system uncertainties, and can be represented as follows for parametric uncertainties.

$$\Delta(\delta) = \text{diag}\,[\delta_1 I_{n1}, \delta_2 I_{n2}, \ldots, \delta_m I_{nm}] \tag{1a}$$

$$\dim[\Delta(\delta)] = n_\Delta = \sum_{i=1}^{m} n_i \tag{1b}$$

$$\delta = [\delta_1, \delta_2, \cdots, \delta_m] \in R^m \tag{2}$$

The LFT equation associated with Figure 3 is given below.

$$S(\delta) = L(I - \Delta P)^{-1} \Delta R + Q \tag{3}$$

$$\Rightarrow \quad S(\delta) = S_\Delta(\delta) + S_o \tag{4}$$

The matrix $S(\delta)$ is a compact representation of the system model. The matrix Q represents the nominal system model. The interconnection matrices P, R, and L are to be determined for the uncertain component of S using the following equation.

$$S_\Delta(\delta) = L(I - \Delta P)^{-1} \Delta R \tag{5}$$

Note that $S_\Delta(\delta)$ contains given system matrices which are functionally dependent on the parameters δ. A solution for equation (5) is summarized below for $S_\Delta(\delta)$ formulated as a multivariate polynomial matrix function of δ. However, it should be noted that multivariate rational functions can also be formulated and solved using this approach (see Ref. [11]).

Equation (5) can be solved for multivariate polynomial problems by replacing the matrix inversion with a finite series expansion and a nilpotency condition,

$$S_\Delta(\delta) = L\Delta R + L[\Delta P + (\Delta P)^2 + \ldots + (\Delta P)^r]\,\Delta R \tag{6}$$

$$(\Delta P)^{r+1} = 0 \tag{7}$$

where r is determined by the degree of the largest nonzero term in $S_\Delta(\delta)$. An expanded definition of P, R, and L containing matrix partitions associated with the $\delta_i I_{n_i}$ blocks of Δ given in Eqn. (2) are defined for i, j = 1, 2,..., m.

$$\mathbf{L}=\begin{bmatrix}\mathbf{L}_1 & \mathbf{L}_2 & \cdots & \mathbf{L}_m\end{bmatrix} \tag{8a}$$

$$\mathbf{R}=\begin{bmatrix}\mathbf{R}_1\\ \mathbf{R}_2\\ \vdots\\ \mathbf{R}_m\end{bmatrix}\ ,\quad \mathbf{P}=\begin{bmatrix}\mathbf{P}_{11} & \mathbf{P}_{12} & \cdots & \mathbf{P}_{1m}\\ 0 & \mathbf{P}_{22} & \cdots & \mathbf{P}_{2m}\\ 0 & 0 & \ddots & \vdots\\ 0 & 0 & 0 & \mathbf{P}_{mm}\end{bmatrix} \tag{8b}$$

where:

$$P_{ij}\in R^{n_i\times n_j},\quad L_i\in R^{n_{rows}\times n_i},\quad R_i\in R^{n_i\times n_{cols}} \tag{9}$$

and each P main-diagonal block is nilpotent of index η_i:

$$(P_{ii})^{\eta_i}=0,\quad \eta_i\le n_i,\ i=1,2,\ldots,m \tag{10a}$$

$$\eta_i = \text{maximum degree of } \delta_i \text{ in } S_\Delta(\delta) \tag{10b}$$

The block-triangular structure of P is sufficient but not necessary for nilpotency, and other special structures can also be found. Solution of Eqn. (6) for the matrices L, P, R and $\Delta(\delta)$ can then be reduced to solving the following set of equations.

Linear δ_i Terms:

$$L_i R_{i_i}=S_{\Delta_{\delta_i}},\ i=1,2,\ldots,m \tag{11}$$

ξ^{th}-Degree δ_i Terms:

$$L_i(P_{ii})^{\xi-1}R_i=S_{\Delta_{(\delta_i)^\xi}},\ i=1,2,\ldots,m;\ \xi=1,2,\ldots,\eta_I \tag{12}$$

Crossterms:

$$L_{i_1}(P_{i_1i_1})^{\xi_{i_1}-1}P_{i_1i_2}(P_{i_2i_2})^{\xi_{i_2}-1}\cdots P_{i_{n_T-1}i_{n_T}}(P_{i_{n_T}i_{n_T}})^{\xi_{i_{n_T}}-1}R_{n_T}=S_{\Delta_{(\delta_{i_1})^{\xi_{i_1}}(\delta_{i_2})^{\xi_{i_2}}\cdots(\delta_{i_{n_T}})^{\xi_{i_{n_T}}}}} \tag{13}$$

where:

$$\xi=\xi_{i_1}+\xi_{i_2}+\ldots+\xi_{i_{n_T}}$$

$$i_1=1,2,\ldots,m-(n_T-1)$$

$$i_2=i_1+1,i_1+2,\ldots,m-(n_T-2)$$

$$\vdots$$

$$i_{n_T}=i_1+(n_T-1),\ldots,m$$

$$n_T = \text{number of parameters in the crossterm} \le m$$

Note that the S_Δ terms on the right-hand side of Eqns. (11) through (13) are the known constant matrix coefficients associated with the indicated parameter terms in $S_\Delta(\delta)$. Moreover, depending on the number of parameters and the degree of each appearing in $S_\Delta(\delta)$, there can be literally hundreds of S_Δ coefficient terms and coupled matrix equations to be solved (or more). Moreover, satisfying these equations as simultaneously as possible to take advantage of any common structure (and reduce the resulting model dimension) while satisfying the nilpotency condition of Eqn. (7) is highly nontrivial.

2.1.1 Solution of L, R, and Main-Diagonal Blocks of P:

A solution for this part of the problem is given in Refs. [13] and [18], but is summarized here for completeness. The blocks of L and R, and the main-diagonal blocks of P are solved simultaneously for each uncertain parameter δ_i using the linear and ξ^{th}-degree δ_i terms. The solution is accomplished such that the resulting main-diagonal blocks

of P are nilpotent with the appropriate index of nilpotency equal to the maximum parameter degree in $S_\Delta(\delta)$. This solution is accomplished numerically with a matrix singular value decomposition (svd) by recognizing that this part of the problem is equivalent to a 1-D state-space (minimal) realization problem and by appropriately defining the equivalent block Hankel matrices. The solution is accomplished for each δ_i parameter as shown by the following theorem.

Theorem

Consider the linear and ζ^{th}-degree δ_i terms of $S_\Delta(\delta)$, which can be expanded as follows

$$S_{\Delta_{L,\zeta}}(\delta) = [S_{\Delta_{\delta_i}}]\,\delta_i + [S_{\Delta_{\delta_i^2}}]\,\delta_i^2 + \ldots + [S_{\Delta_{\delta_i^{\eta_i}}}]\,\delta_i^{\eta_i} \tag{14}$$

and use the constant coefficient matrices of Eqn. (14) to form the block Hankel matrices defined below

$$\bar{S}_{\Delta 0_{\delta_i}} = \text{Hankel}[S_{\Delta_{\delta_i}}\ S_{\Delta_{\delta_i^2}}\ \cdots S_{\Delta_{\delta_i^{\eta_i}}}] \tag{15}$$

$$\bar{S}_{\Delta 1_{\delta_i}} = \text{Hankel}[S_{\Delta_{\delta_i^2}}\ \cdots S_{\Delta_{\delta_i^{\eta_i}}}\ 0] \tag{16}$$

where:

$$\text{Hankel}\begin{bmatrix} H_1 & H_2 & \cdots & H_n \end{bmatrix} = \begin{bmatrix} H_1 & H_2 & \cdots & H_n \\ H_2 & \iddots & \iddots & 0 \\ \vdots & \iddots & \iddots & \vdots \\ H_n & 0 & \cdots & 0 \end{bmatrix} \tag{17}$$

Eqn. (16) can be constructed from (15) by shifting each block row up and filling in the bottom block row with zero blocks. Define the matrix svd of Eqn. (15) as follows.

$$\bar{S}_{\Delta 0_{\delta_i}} = U_{\delta_i}\Sigma_{\delta_i}V_{\delta_i}^T = (U_{\delta_i}\Sigma_{\delta_i}^{1/2})(\Sigma_{\delta_i}^{1/2}V_{\delta_i}^T) = \bar{L}_i\bar{R}_i \tag{18}$$

where:

$$\text{rank}(\bar{S}_{\Delta 0_{\delta_i}}) = \text{rank}(\bar{L}_i) = \text{rank}(\bar{R}_i)$$

Then, $\{P_{ii}, R_i\}$ is controllable and $\{L_i, P_{ii}\}$ is observable, and the matrices L_i, R_i, and P_{ii} form an irreducible realization of $S_{\Delta_{L,\zeta}}(\delta)$ as defined by Eqn. (14), where:

$$L_i = \begin{bmatrix} I_{n_{rows}} & 0 \end{bmatrix}\bar{L}_i \quad , \quad R_i = \bar{R}_i\begin{bmatrix} I_{n_{cols}} \\ 0 \end{bmatrix} \tag{19}$$

$$P_{ii} = (\bar{L}_i)^\dagger\,\bar{S}_{\Delta 1_{\delta_i}}\,(\bar{R}_i)^\dagger \tag{20}$$

and the notation $(A)^\dagger$ designates the pseudoinverse of matrix A. The P_{ii} matrix is nilpotent with index η_i. ¶

A proof of this theorem is provided in Reference [18].

2.1.2 Solution of P Off-Diagonal Blocks:

The P off-diagonal blocks are each solved using the appropriate crossterms of $S_\Delta(\delta)$, as defined by Eqn. (13). The equation to be solved for each off-diagonal block of P is a generalized linear matrix equation. The general form of the equation is given below for computing P_{nj}, where n = 1, 2, ..., m–1 and j = n+1, n+2, ... , m.

$$\bar{A}_n^{[n]}P_{nj}\,B_j = \bar{S}_{\Delta\delta_n\delta_j}^{[n]} \tag{21}$$

The matrices $\overline{A}_n^{[n]}$, B_j, and $\overline{S}_{\Delta\delta n\delta_j}^{[n]}$ in Eqn. (21) are comprised of known matrices as well as matrices that have already been computed at this point in the solution process. The detailed equations for Block Rows 1 and 2 are given below.

Block Row 1:

$$\overline{\mathbf{A}}_1^{[1]} P_{1j} B_j = \overline{S}_{\Delta\delta_1\delta_j}^{[1]} \tag{22a}$$

where:

$$\overline{\mathbf{A}}_1^{[1]} = \begin{bmatrix} \mathbf{L}_1 \\ \mathbf{L}_1\mathbf{P}_{11} \\ \vdots \\ \mathbf{L}_1\mathbf{P}_{11}^{\eta_1-1} \end{bmatrix} \tag{22b}$$

Block Row 2:

$$\overline{\mathbf{A}}_2^{[2]} P_{2j} B_j = \overline{S}_{\Delta\delta_2\delta_j}^{[2]} \tag{23a}$$

where:

$$\overline{\mathbf{A}}_2^{[2]} = \begin{bmatrix} \overline{\mathbf{A}}_2^{[1]} \\ \overline{\mathbf{A}}_1^{[1]}\mathbf{P}_{12} \\ \overline{\mathbf{A}}_1^{[1]}\mathbf{P}_{12}\mathbf{P}_{22} \\ \vdots \\ \overline{\mathbf{A}}_1^{[1]}\mathbf{P}_{12}\mathbf{P}_{22}^{\eta_2-1} \end{bmatrix}, \quad \overline{\mathbf{A}}_2^{[1]} = \begin{bmatrix} \mathbf{L}_2 \\ \mathbf{L}_2\mathbf{P}_{22} \\ \vdots \\ \mathbf{L}_2\mathbf{P}_{22}^{\eta_2-1} \end{bmatrix} \tag{23b}$$

B_j Matrix Structure for Each Block Row:

$$\mathbf{B}_j \equiv \overline{\mathbf{R}}_j = \begin{bmatrix} \mathbf{R}_j & \mathbf{P}_{jj}\mathbf{R}_j & \cdots & \mathbf{P}_{jj}^{\eta_j-1}\mathbf{R}_j \end{bmatrix} \tag{24}$$

The $\overline{S}_{\Delta\delta n\delta_j}^{[n]}$ matrices contain the constant coefficient matrices associated with the cross-product terms being solved.

$$\overline{S}_{\Delta\delta_i\delta_j}^{[1]} = \begin{bmatrix} \mathbf{S}_{\delta_i\delta_j} & \mathbf{S}_{\delta_i\delta_j^2} & \cdots & \mathbf{S}_{\delta_i\delta_j^{\eta_j}} \\ \mathbf{S}_{\delta_i^2\delta_j} & \mathbf{S}_{\delta_i^2\delta_j^2} & \cdots & \mathbf{S}_{\delta_i^2\delta_j^{\eta_j}} \\ \vdots & \vdots & \cdots & \vdots \\ \mathbf{S}_{\delta_i^{\eta_i}\delta_j} & \mathbf{S}_{\delta_i^{\eta_i}\delta_j^2} & \cdots & \mathbf{S}_{\delta_i^{\eta_i}\delta_j^{\eta_j}} \end{bmatrix} \tag{25a}$$

where: $i = 1, 2, \ldots, j\text{-}1$, $j = 2, 3, \ldots, m$ $(i \neq j)$

$$\overline{S}_{\Delta\delta_2\delta_j}^{[2]} = \begin{bmatrix} \overline{S}_{\Delta\delta_2\delta_j}^{[1]} \\ \overline{S}_{\Delta\delta_1\delta_2\delta_j} \end{bmatrix}, \quad \overline{S}_{\Delta\delta_1\delta_2\delta_j} = \begin{bmatrix} \overline{S}_{\Delta\delta_1\delta_2\delta_j}^{[1]} \\ \overline{S}_{\Delta\delta_1\delta_2\delta_j}^{[2]} \\ \vdots \\ \overline{S}_{\Delta\delta_1\delta_2\delta_j}^{[\eta_2]} \end{bmatrix} \tag{25b}$$

$$\bar{S}_{\Delta\delta_1\delta_2\delta_j}{}^{[k]} = \begin{bmatrix} \mathbf{S}_{\delta_1\delta_2{}^k\delta_j} & \mathbf{S}_{\delta_1\delta_2{}^k\delta_j{}^2} & \cdots & \mathbf{S}_{\delta_1\delta_2{}^k\delta_j{}^{\eta_j}} \\ \mathbf{S}_{\delta_1{}^2\delta_2{}^k\delta_j} & \mathbf{S}_{\delta_1{}^2\delta_2{}^k\delta_j{}^2} & \cdots & \mathbf{S}_{\delta_1{}^2\delta_2{}^k\delta_j{}^{\eta_j}} \\ \vdots & \vdots & \cdots & \vdots \\ \mathbf{S}_{\delta_1{}^{\eta_1}\delta_2{}^k\delta_j} & \mathbf{S}_{\delta_1{}^{\eta_1}\delta_2{}^k\delta_j{}^2} & \cdots & \mathbf{S}_{\delta_1{}^{\eta_1}\delta_2{}^k\delta_j{}^{\eta_j}} \end{bmatrix} \tag{25c}$$

where: $k = 1, 2, \ldots, \eta_2$

The off-diagonal block equations for Block Row 1, Eqns. (22), (24) and (25a) for i=1, solve all pair-wise cross-product terms associated with $\delta_1\delta_j$. The off-diagonal block equations for Block Row 2, Eqns. (23), (24), and (25) for i=2, solve all pair-wise cross-product terms associated with $\delta_2\delta_j$, plus all triple terms involving $\delta_1\delta_2\delta_j$. The third block row equations (not shown, but constructed similarly) solve all pair-wise crossterms associated with $\delta_3\delta_j$, plus all triples associated with $\delta_1\delta_3\delta_j$ and $\delta_2\delta_3\delta_j$, plus all quadruple terms associated with $\delta_1\delta_2\delta_3\delta_j$. Thus, the equations for the i^{th} block row solves the pair-wise crossterms associated with $\delta_i\delta_j$, plus all combinations of crossterms involving δ_1, δ_2, …, δ_i and δ_j. Note that the solutions include all n^{th}-degree terms. This is accomplished by the main-diagonal blocks of P (P_{ii}) raised to various powers up to $\eta_i - 1$, as defined by Eqn. (10b), appearing in the A and B matrices of Eqn. (21). The general expressions for Eqn. (21) are given in Ref. [18], but details of an approach for solving them are given below.

Eqn. (21) is a generalized linear matrix equation of the form

$$AXB = C \tag{26}$$

where A, B, and C are known constant matrices. Thus, solution of the off-diagonal blocks of P can be reduced to solving matrix equations of the form of Eqn. (26), which requires satisfaction of the following rank conditions.

$$\text{rank}[A \;\; C] = \text{rank}[\, A \,], \;\; \text{rank}[B^T \; C^T]^{\,T} = \text{rank}[\, B \,] \tag{27}$$

Satisfaction of the column rank condition (first) and row rank condition (second) of Eqn. (27) is accomplished through augmentation of the dimension of the appropriate parameters, δ_i, in Δ, which translates to an augmentation of the associated partitions of L, R, and P comprising the A and B matrices in Eqn. (26). This is a nontrivial task, because columns or rows of C cannot simply be appended to A and B. For the column rank condition, the structure of A becomes more complicated for successive block rows of P, for higher numbers of parameters, and for higher parameter degrees. For the row rank condition, the B matrix structure is fixed, but is more complicated for higher parameter degrees.

Another complication to performing the augmentation is that in augmenting the underlying L, R, and P matrix partitions of A and B, the previously obtained solutions involving these partitions must be retained, as well as the nilpotency (with the correct nilpotency index) of the main-diagonal blocks of P. Moreover, the augmentation process must be general and implementable for any number of parameters and for any parameter degree.

The approach taken in this paper to solve the above augmentation problem is based on utilizing the structure of the A and B matrices (i.e., involving successive powers of nilpotent matrices), and allows an arbitrary augmentation to be performed independently of the coefficient matrices in C. This enables the satisfaction of the rank conditions of Eqn. (27) for any problem. The basic result is given in the following Lemma, which is an extension of a Theorem by Halmos in Ref. [21].

Lemma

Let $N \in R^{nxn}$ be a nilpotent matrix of index q, and $M \in R^{nxm}$ (n>m) be an arbitrary rank m matrix such that $\text{rank}(N^{q-1} M) = m$. Then, the columns of M, NM, … , $N^{q-1}M$ are linearly independent; i.e., for $n \geq qm$:

$$\text{Rank}[M, NM, \ldots, N^{q-1}M] = qm \tag{28}$$

where $[M, NM, \ldots, N^{q-1}M] \in R^{nxqm}$. ¶

This lemma directly applies to the B matrix given by Eqn. (24); its dual (obtained by taking transposes) can be directly applied to the A matrix for block row 1, as given by Eqn. (22b). However, it can be shown that the lemma

can also be applied to the off-diagonal block solution for any block row by reformulating the associated A matrix to be in the (transposed) form of Eqn. (28). Thus, arbitrary columns can be added to the columns of L_i, and arbitrary rows can be added to the rows of R_j during the augmentation process. Moreover, an arbitrary nilpotent augmentation of the correct nilpotency index can be added to the main diagonal block partitions. The augmented matrices become:

$$\mathbf{L}_{i_{aug}} = \begin{bmatrix} \mathbf{L}_i & \hat{\mathbf{L}}_i \end{bmatrix}, \ \mathbf{R}_{i_{aug}} = \begin{bmatrix} \mathbf{R}_i \\ \hat{\mathbf{R}}_i \end{bmatrix}, \ \mathbf{P}_{ii_{aug}} = \begin{bmatrix} \mathbf{P}_{ii} & 0 \\ 0 & \hat{\mathbf{P}}_{ii} \end{bmatrix} \tag{29}$$

where $\hat{\mathbf{P}}_{ii}$ is obtained using the general description for a nilpotent matrix given in Ref. [22].

The general process is to augment L_i and P_{ii} to satisfy the column rank condition for block row i, and to augment R_j and P_{jj} to satisfy the row rank condition for block column j. In order to retain previous solutions and to permit arbitrary augmentations to previously computed P_{ij} matrices, R_i (L_j) is augmented with zero rows (columns) when L_i and P_{ii} (R_j and P_{jj}) are augmented to satisfy the column (row) rank condition.

2.1.3 Full P-Δ Model Solution:

Once the L_i, R_i, P_{ii}, and P_{ij} matrices for each parameter have been determined, the full solution is assembled. This is a simple matter of collecting the matrix partitions together into the full L, R, and P matrices defined in Eqn. (8). The Δ matrix is also known and given by Eqn. (2), where the number of repetitions for each parameter, n_i, was determined in solving the L_i, R_i, P_{ii}, and P_{ij} matrices.

2.2 Description of Preliminary Software Tool

The parametric LFT modeling method described in Section 2.1 has been implemented as a preliminary software tool. The tool has been developed for general problems involving m parameters each raised to any maximum degree, plus all possible cross-product terms. The software was developed in Matlab, and requires Matlab 5 or above, the Control System Toolbox (for the current reduction algorithms described in Subsection 2.2.3), and the robust control toolbox. Subsection 2.2.1 describes the data structure and syntax for the function call, Subsection 2.2.2 describes the LFT model construction, and Subsection 2.2.3 describes a rudimentary reduction approach that is currently included in the software.

2.2.1 Data Structure and Function Call

The main function to solve for the LFT uncertainty model is "lft_model", which has the following syntax.

[L,P,R,Q,L_full,P_full,R_full,Q_full,delta,Snew,maxtest] = lft_model(S,tol)

The input S is an n-dimensional cell array containing the coefficient matrices, where n is the number of parameters. The index of cell array S relates to the term, and its content is the coefficient matrix associated with that term. Since the index of a Matlab cell array does not start with zero, the index of the cell array S is defined as the order of the term plus one. For example, consider the following:

$$S(\delta) = S_0 + \left[S_{\Delta_x}\right]x + \left[S_{\Delta_{y^3}}\right]y^3 + \left[S_{\Delta_{x^2y}}\right]x^2y$$

The highest degree of x is 2 from the term x^2y, and the highest degree of y is 3 from the term y^3. If x is considered as the first parameter and y the second one, the order of parameters is (x, y). The cell array S would then be entered as follows.

$$S\{1,1\} = S_0, \ S\{2,1\} = S_{\Delta_x}, \ S\{1,4\} = S_{\Delta_{y^3}} \text{ and } S\{3,2\} = S_{\Delta_{x^2y}}$$

Note that the index of a cell is equal to the order of the corresponding parameter plus one. If the order of parameters is (y, x), then $S\{1,2\} = S_{\Delta_x}$ and so on. Zero coefficient matrices, for example, S_{Δ_y}, $S_{\Delta_{xy}}$, $S_{\Delta_{x^2y^2}}$, …, can be left empty, and they will be filled with zeros by the software.

The input "tol" is optional, and specifies the tolerance on the frobenius norm of the difference between the LFT model and the LPV model. The default value of "tol" is 1e-10. The parameter "tol" will be explained more below when the output "maxtest" is discussed.

The outputs L, P, R are cell arrays. $L\{i\}$, $R\{i\}$ are associated with $\delta_i I_{n_i}$ (equation (8)), $P\{i,j\}$ with $\delta_i I_{n_i}$ and $\delta_j I_{n_j}$ (equation (8b)). The output Q is a matrix; $Q = S_0$. The outputs L_full, P_full, and R_full are matrices formed from the cells of L, P, R (equation 8), and Q_full = Q.

The output "delta" is a 1xn vector, where n is the number of parameters. The i[th] element of "delta" is n_i (see equation (1a)).

The output "Snew" is a cell array, which contains the cells of the input cell array S and the zero coefficient matrices filled out by the software. If the input cell array S has zero coefficient matrices associated with terms with higher degree than η_i, the maximum degree of δ_i (equation (10b)), the software will delete those cells and adjust the index of the cell array S. If the coefficient matrices associated with one or more parameters are zero (or empty) matrices, the software deletes all those matrices and adjusts the dimension of cell array S. The software displays messages that the maximum degree of a parameter is reduced and/or a parameter is deleted. The parameters subsequent to the deleted parameter are moved up to replace the deleted one. The adjusted cell array S is stored in the output "Snew". The input cell array S in the Matlab workspace is not altered by the software.

The output "maxtest" displays the maximum result of the comparison of the original coefficient matrices of the LPV model with the corresponding coefficient matrices of the LFT model and the results of the nilpotency of $P\{i,i\}$ using the frobenius norm. If the comparison of the LPV and LFT and/or the nilpotency of $P\{i,i\}$ is greater than "tol", messages will be displayed to notify the user.

2.2.2 LFT Model Construction

This section will refer to equations in this paper and in Reference [18]. Note that the notation in this paper and that in Reference [18] are related as follows. P_{11}, P_{12}, P_{21}, P_{22} in Reference [18] are equivalent to P, R, L, Q in this paper. $P_{21_{\delta_i}}$ in Reference [18] is equivalent to L_t in this paper, $P_{11_{\delta_i\delta_j}}$ in Reference [18] to P_{ij} in this paper, $P_{12_{\delta_j}}$ in Reference [18] to R_j in this paper.

Let n_x be the number of states, n_u be the number of inputs, and n_y be the number of outputs. The LFT model outputs are calculated in the following steps: (1.) Q, (2.) main-diagonal block P_{ii}, L_i, R_i associated with the parameter δ_i, and (3.), off-diagonal block P_{ij}, updating L_i, updating R_j. These steps are described below.

Step 1: Calculate Q

From equation (3), Q is the nominal system and is equal to S_0 or $Q = S(\underbrace{1,\ldots,1}_{n}) = S(1)$

Step 2: Calculate L_i, R_i and the main-diagonal blocks of P

The matrices L_i, R_i, and P_{ii} (i.e., the main-diagonal blocks of P) are computed using Matlab function "lft_svpoly" of the LFT uncertainty modeling software. The blocks P_{ii}, L_i, R_i associated with each parameter δ_i are calculated simultaneously. The appropriate coefficient matrices are used to form $\bar{S}_{\Delta_{0_{\delta_i}}}$ and $\bar{S}_{\Delta_{1_{\delta_i}}}$ in equations (15) and (16), respectively. If n_i, the rank of $\bar{S}_{\Delta_{0_{\delta_i}}}$, is not zero, equations (18), (19) and (20) yield the blocks $P_{ii} \in R^{n_i \times n_i}$, $L_i \in R^{(n_x+n_y)\times n_i}$, $R_i \in R^{n_i \times (n_x+n_u)}$. In the Matlab workspace, P_{ii}, L_i, R_i are stored as $P\{i,i\}$, $L\{i\}$, $R\{i\}$, respectively. If n_i is zero, $P\{i,i\}$, $L\{i\}$, $R\{i\}$ are left empty.

Step 3: Calculate the off-diagonal blocks of P

The off-diagonal blocks of P are computed using Matlab function "lft_mvcross" of the LFT uncertainty modeling software. The formation of $\overline{A}_n^{[n]}$, B_j, and $\overline{S}_{\Delta_{\delta_n\delta_j}}{}^{[n]}$ in equation (21) is nontrivial, and equation 3.42 in Reference [18] was used in the implementation. $\overline{A}_n^{[n]}$ in Reference [18] is split into two matrices, which are translated into the notation in this paper as $\overline{A}_n^{[n]} = \overline{L}_{\delta_n}{}^{[n]}\overline{P}_{\delta_n}{}^{[n]}$. The structures for $\overline{L}_{\delta_n}{}^{[n]}$ and $\overline{P}_{\delta_n}{}^{[n]}$ are discussed in detail in Reference [18]. Forming the generalized $\overline{S}_{\Delta_{\delta_i\delta_j}}{}^{[1]}$, $\overline{S}_{\Delta_{\delta_2\delta_j}}{}^{[2]}$, …, $\overline{S}_{\Delta_{\delta_1\delta_2\delta_j}}{}^{[k]}$, … matrix partitions (as shown in equations (25 a,b,c) for two parameters) for any number of parameters raised to an arbitrary order was also challenging. Since each cell of cell array S is associated with a term of the LPV model, allowing the number of parameters to change from problem to problem means that the dimension of S must change accordingly. In addition, since the order of the uncertain parameters change from problem to problem, the maximum index of S also changes. Placing the correct cell into the appropriate location in the matrix partitions of equations (25 a,b,c) was accomplished using the following observation in order to avoid the use of Matlab string operations. Any cell in a Matlab multidimensional cell array can be referred to by two methods: either by the full index of the cell or the equivalent one-number index of the cell. Cells in a cell array in the Matlab workspace are stacked 'column-wise'. Cell $\{k_1,\ldots,k_n\}$ of cell array $S\{m_1,\ldots,m_n\}$, $(k_i \le m_i)$ can be referred to as cell $\{K\}$ where K is calculated as follows.

$$K = k_1 + \sum_{i=2}^{n}\left((k_i - 1)\prod_{j=1}^{i-1} m_j \right)$$

The off-diagonal blocks of P are then calculated in the following order P_{12}, P_{13}, …, P_{1n}, P_{23}, …, P_{2n}, …, $P_{(n-1)(n)}$. A different order of calculation of the off-diagonal blocks of P can be followed, but equation (21) and related equations must be revised accordingly. Equation (21), which has the form of equation (26), will yield P_{ij}, $i = 1,\ldots,n-1$, $j = i+1,\ldots,n$ if the rank condition in equation (27) is satisfied. If the column (row) rank condition for P_{ij} is not satisfied, $\overline{A}_i^{[i]}$ (B_j) in equation (21) is augmented to satisfy the column (row) rank condition. As mentioned in section 2.1, the augmentation must retain previously obtained solutions as well as the nilpotency of the main-diagonal blocks of P. The augmentation is described in equation (29) and in the last paragraph of section 2.1. In this preliminary software implementation, random matrices are generated for the arbitrary matrix augmentations. A more sophisticated approach will be considered for future work.

2.2.3 One-Dimensional Model Reduction

A modest approach to LFT model reduction is included in the preliminary software tool. Once the LFT model is formed, each sub-block of Δ is treated as a one-dimensional system and "uncontrollable" and "unobservable" modes are removed by performing the transformation provided in the Control System Toolbox. The one-dimensional system 'A', 'B', and 'C' matrices are formed as follows.

$$A_k = P\{k,k\},\ k = 1,\ldots,n$$
$$B_k = \begin{bmatrix} R\{k\} & P\{k,k+1\} & P\{k,k+2\} & \ldots & P\{k,n\} \end{bmatrix}$$
$$C_k = \begin{bmatrix} L\{k\}; & P\{1,k\}; & P\{2,k\}; & \vdots & P\{k-1,k\} \end{bmatrix}$$

The A_k, B_k, C_k matrices are reduced using the controllability and observability rank tests, and the reduced matrices are denoted as $A_{r,k}$, $B_{r,k}$, $C_{r,k}$. Let $P_{size} = \begin{bmatrix} rank_1 & rank_2 & \ldots & rank_n \end{bmatrix}$, where $rank_i$ is the result of the rank test for the i[th] parameter. Then the matrices L, P, R are updated as shown below.

$$P\{k,k\} = A_{r,k}$$

$L\{k\} = C_{r,k}(1:n_{rows},:)$ where n_{rows} is the number of rows of a coefficient matrix

$R\{k\} = B_{r,k}(:,1:n_{cols})$ where n_{cols} is the number of columns of a coefficient matrix

Let $C_{b,k} = C_{r,k}(n_{rows}+1:end,:)$ and $B_{b,k} = B_{r,k}(:,n_{cols}+1:end)$

$$P\{i,k\} = C_{b,k}\left(\sum P_{size}(1:i-1)+1:\sum P_{size}(1:i),1:rank_k\right),\ i = 1,2,\ldots,k-1$$

$$P\{k,j\} = B_{b,k}\left(1:rank_k,\sum P_{size}(k+1:j-1)+1:\sum P_{size}(k+1:j)\right),\ j = k+1,k+2,\ldots,n$$

This reduction process is described in more detail in Reference [11]. Other more sophisticated reduction methods (see References [15]-[17]), will be incorporated into the next version of the software.

2.2.4 Potential Software Improvements

The software tool described in this paper is preliminary, and a number of potential improvements in its sophistication can be made. Some of these are considered below.

- Numerical conditioning issues have not been fully considered. A more thorough evaluation and treatment of this should be performed.
- Generating a random matrix for the augmentation was an initial simplistic approach to generating the arbitrary augmentation matrices for solving the off-diagonal blocks of the P matrix. A more sophisticated approach should be considered – perhaps one that makes use of the minimality criteria defined in References [15] – [17].
- The current software tool does not include much logic for tailoring the set of equations to be solved to the specific problem. Specific potential areas of improvement include the following.
  - Simplification of the equations may be possible (and preferable relative to model dimension) for simpler problems.
  - The software currently uses an upper block triangular structure for P – which is sufficient but not necessary for nilpotency. Allowing more flexibility in this structure could reduce the resulting model order. For example, Reference [18] includes a problem for which an LFT model of dimension 18 can be achieved when the P matrix is not constrained to an upper block triangular structure (as compared to a dimension of 19 with the constraint).
- The structure of the equations being solved for the cross-product terms was formulated such that the matrix A in equation (26) absorbs most of the cross-product term complexity. It may be possible to reformulate the equations to balance the complexity between the A and B matrices of equation (26), which could possibly result in lowering the LFT model dimension.
- A more sophisticated model reduction method, such as that presented in References [15] – [17], should be incorporated into the software.
- Scaling is not currently included in the software to produce a normalized LFT. This function should be incorporated into the software.

3. LFT Modeling Examples

Several example problems are presented in this section to illustrate the development of LPV and LFT models using the matrix-based approach developed in this paper. Section 3.1 presents the development of these models for an aircraft at extreme flight conditions, Section 3.2 presents a missile model problem, and Section 3.3 provides a

generic example developed to be a difficult problem involving three uncertain parameters and all associated cross-product terms.

3.1 Uncertainty Modeling for Extreme Aircraft Flight Conditions

To illustrate uncertainty modeling for an extreme flight condition, LPV and LFT models were developed for an F-16 aircraft near a stall bifurcation. The following sub-sections describe the development of these models. Section 3.1.1 describes the general approach for developing LPV models for extreme flight conditions, and section 3.1.2 provides the F-16 example problem.

3.1.1 Formulation of LPV Models Near Bifurcation Points

An LPV model can be formulated for extreme flight conditions at (or very near) bifurcation points using the method proposed in Reference [24]. Consider the parameter dependent nonlinear system

$$\begin{aligned} \dot{x} &= f(x,u,\mu) \\ y &= g(x,u,\mu) \end{aligned} \tag{30}$$

Where $x \in R^n$ is the state, $u \in R^p$ is the control, $y \in R^m$ is the output, and $\mu \in R^k$ is a parameter vector. Parameters may include quantities that define the desired operating condition, like speed, flight path angle, altitude or physical parameters such as weight, center of mass location, etc. We are interested in how the vehicle behaves given different parameter values. To this end we wish to construct a family of linear time-invariant models corresponding to different values of the parameters, μ.

A linear time-invariant model is obtained from equation (30) by linearizing at a specified equilibrium point. Consequently, we first need to characterize the dependence of equilibrium points on the parameters. An equilibrium point for the system of equations (30) is a triple (x_0,u_0,μ_0) that satisfies the conditions $f(x_0,u_0,\mu_0)=0$ and $g(x_0,u_0,\mu_0)=0$. The set of equilibrium points is a k dimensional manifold in the space R^{n+m+k}. Formally, the equilibrium manifold is defined by

$$\mathcal{E} = \left\{ (x,u,\mu) \in R^{n+m+k} \,\middle|\, f(x,u,\mu)=0, g(x,u,\mu)=0 \right\} \tag{31}$$

The manifold will typically be quite complex and may not even be smooth. An example of a two parameter surface is shown in Figure 4. At any equilibrium point, it is possible to derive a linear approximation to the system (30) in the form

$$\begin{aligned} \delta\dot{x} &= A\delta x + Bu \\ \delta y &= C\delta x + Du \end{aligned} \tag{32}$$

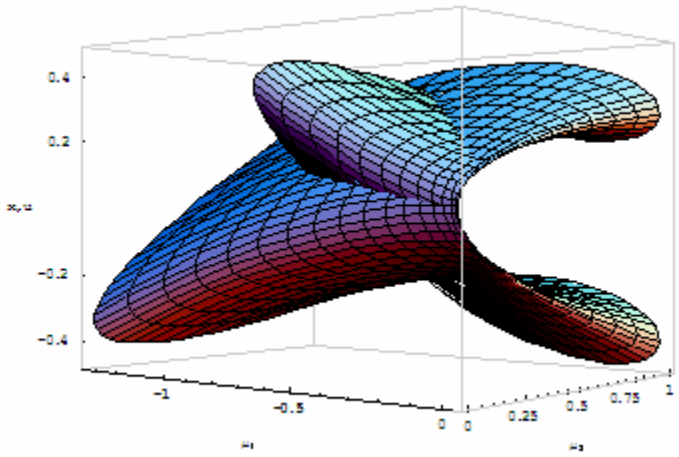

Figure 4. A two parameter surface illustrates the complex structure found in the equilibrium manifold.

The parameters A, B, C, D are obtained by evaluating the appropriate Jacobian matrices at a specified equilibrium point $p \in \mathcal{E}$.

It is evident from Figure 4 that any attempt to express equilibrium values for x, u as functions of the parameters μ is inherently limited by the folds and creases in the surface – i.e., the bifurcation points. An alternative approach can be defined in which the surface itself is parameterized so that it is possible to define functions in these new parameters. These functions may be globally valid, but even if local, the domain need not be constrained by bifurcation points. In Reference [24] it is shown how to obtain a local parameterization of $\mathcal{E}$ around any point $p_0 \in \mathcal{E}$. In this method, a parameter vector $s \in R^k$ is introduced so that points $p \in \mathcal{N} \subset \mathcal{E}$, where $\mathcal{N}$ is a neighborhood of p_0 in $\mathcal{E}$, are defined parametrically by $p = p(s)$ on a neighborhood of the origin in R^k. In this representation, the equilibrium state, control, and parameters are all given as functions of the new parameters s.

$$x_0 = x_0(s), \quad u_0 = u_0(s), \quad \mu_0 = \mu_0(s)$$

Once this parameterization of the equilibrium surface is obtained, an LPV model of the form

$$\begin{aligned} \delta\dot{x} &= A(s)\delta x + B(s)u \\ \delta y &= C(s)\delta x + D(s)u \end{aligned} \tag{33}$$

is easily constructed. A software tool has been developed in Mathematica® for accomplishing this.

3.1.2 F-16 Example

The LPV and LFT models for an F-16 example near a stall bifurcation are given in this section. Figures 5 and 6 show a portion of the equilibrium surface of an F-16 near the stall condition. Details about the dynamical behavior of the aircraft near the bifurcation point can be found in Reference [25].

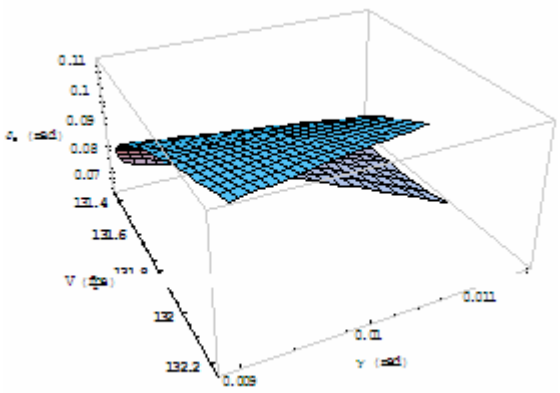

Figure 5. A portion of the equilibrium surface for an F-16 is shown. The parameters are speed V and flight path angle γ. Only one of the control variables, elevator deflection δ_e, is shown.

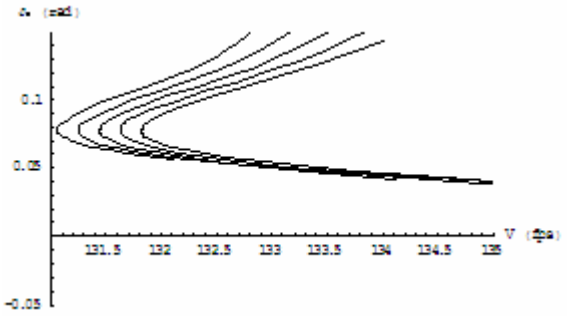

Figure 6. Another representation of the equilibrium surface. Slices through the surface at constant flight path angle are shown. From left to right $\gamma = -0.005, -0.0025.0, 0.0025, 0.005$. The surface clearly shows stall as speed is reduced. Notice that the stall speed increases with increasing flight path angle.

The functions defining the equilibrium surface, $x_0(s), u_0(s), \mu_0(s)$ as well as the matrices $A(s), B(s), C(s), D(s)$ are obtained as polynomials in the parameters s.

F-16 LPV Model

The LPV model consists of matrices $A(s_1,s_2), B(s_1,s_2), C(s_1,s_2), D(s_1,s_2)$. The parameters s_1, s_2 are related to speed, V, and flight path angle, γ, through well-defined formulas, as shown below.

$$V = f_V(s_1,s_2), \quad \gamma = f_\gamma(s_1,s_2)$$

$$\begin{aligned}\alpha &= 0.873652 + 0.124836\, s_1 - 0.0607641\, s_1^2 + 0.0320992\, s_1^3 - 9.60292\, s_2 \\ &\quad + 10.371\, s_1 s_2 - 8.14459\, s_1^2 s_2 - 438.22\, s_2^2 + 690.549\, s_1 s_2^2 - 19536.8\, s_2^3\end{aligned}$$

$$\begin{aligned}\mathrm{Th} &= 15064.284 + 2256.19\, s_1 - 1040.09\, s_1^2 + 745.335\, s_1^3 - 187438\, s_2 \\ &\quad + 180989\, s_1 s_2 - 186906\, s_1^2 s_2 - 7.8323\times10^6\, s_2^2 + 1.56862\times10^7\, s_1 s_2^2 - 4.4026\times10^8\, s_2^3\end{aligned}$$

$$\begin{aligned}\mathrm{dele} &= 0.0827526 + 0.0717976\, s_1 + 0.0226667\, s_1^2 + 0.0239419\, s_1^3 - 5.52297\, s_2 \\ &\quad - 2.89913\, s_1 s_2 - 5.00515\, s_1^2 s_2 + 88.8865\, s_2^2 + 349.239\, s_1 s_2^2 - 8145.83\, s_2^3\end{aligned}$$

$$\begin{aligned}\theta &= 0.863652 + 0.124836\, s_1 - 0.0638262\, s_1^2 + 0.0381331\, s_1^3 - 10.6029\, s_2 \\ &\quad + 10.8421\, s_1 s_2 - 9.71647\, s_1^2 s_2 - 460.435\, s_2^2 + 825.264\, s_1 s_2^2 - 23364.5\, s_2^3\end{aligned}$$

$$\begin{aligned}V &= 131.475 + 1.0\, s_1 + 6.38958\, s_1^2 - 5.1431\, s_1^3 + 0.0\, s_2 \\ &\quad - 954.805\, s_1 s_2 + 1300.36\, s_1^2 s_2 + 35898.1\, s_2^2 - 108465\, s_1 s_2^2 + 2.99105\times10^6\, s_2^3\end{aligned}$$

$$\begin{aligned}\gamma &= 0.01 + 0.0\, s_1 + 0.00306217\, s_1^2 - 0.00603397\, s_1^3 + 1.0\, s_2 \\ &\quad - 0.47111\, s_1 s_2 + 1.57187\, s_1^2 s_2 + 22.2155\, s_2^2 - 134.715\, s_1 s_2^2 + 3827.73\, s_2^3\end{aligned}$$

Thus, each pair (s_1,s_2) corresponds to a unique (V,γ). The origin $(s_1,s_2)=(0,0)$ corresponds to a flight condition close to the stall bifurcation point. In particular, $(s_1,s_2)=(0,0)\rightarrow(V,\gamma)=(131.45\ \text{fps}, 0.01\ \text{rad})$. Functions for V and γ are plotted in Figures 7 and 8.

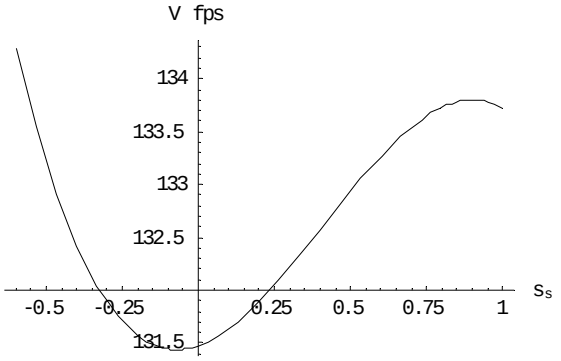

Figure 7. V **as a function of** s_1 **with** $s_2 = 0$**.**

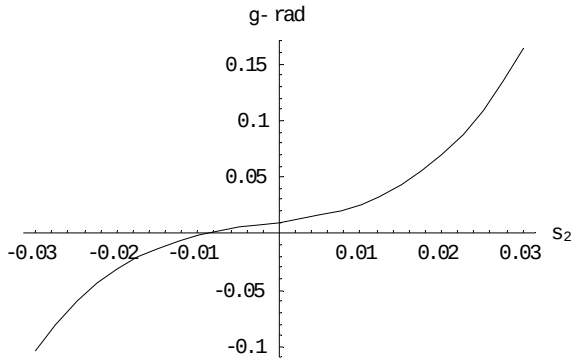

Figure 8. γ **as a function of** s_2 **with** $s_1 = 0$**.**

The LPV model for this example is given as:

$$\begin{aligned}\dot{x} &= A(s_1,s_2)x + B(s_1,s_2)u \\ y &= C(s_1,s_2)x + D(s_1,s_2)u\end{aligned}$$

where the parameter dependent matrices are built as follows:

$$A(s_1,s_2) = AA00 + AA10 s_1 + AA01 s_2 + AA11 s_1 s_2 + AA20 s_1^2 + \cdots$$

and similarly for the others.

The LPV model has 9 states: $x = \{\phi, \theta, \psi, p, q, r, V, \alpha, \beta\}$, 5 controls $u = \{\delta_a, \delta_{el}, \delta_{er}, \delta_r, Th\}$, and 10 outputs $y = \{x, \gamma\}$, and it does capture the bifurcation behavior shown in Figures 5 and 6. The detailed F-16 LPV model is provided in the Appendix.

F-16 LFT Model

The LFT can be obtained using the LFT modeling tool described in Section 2 by assigning the S matrix cell array as follows:

S{1,1} = [AA00 BB00], S{2,1} = [AA10 BB10], S{1,2} = [AA01 BB01]
S{2,2} = [AA11 BB11], S{3,1} = [AA20 BB20], S{1,3} = [AA02 BB02]
S{3,2} = [AA21 BB21], S{2,3} = [AA12 BB12], S{4,1} = [AA30 BB30]
S{1,4} = [AA03 BB03]

Note that matrices C and D need not be included in S, since they do not contain uncertain components for this example. The value of tol was set to 1×10^{-8}. The resulting LFT model dimension was 67, with 27 occurrences for s_1 and 40 occurrences for s_2. These results can be further reduced using more sophisticated reduction methods, as shown in Section 4.2.

3.2 Missile Model Problem

The short period dynamics of a missile are considered in the example from Reference [27], with states angle of attack (α) in radians and pitch rate (q) in radians/sec, and control input (fin deflection) (δ) in radians. The nonlinear equations of motion are given as follows.

$$\dot{\alpha} = \frac{0.7 p_0 C_n M^2 S \cos^2(\alpha)}{(mass)U} + q$$

$$\dot{q} = \frac{0.7 p_0 C_n M^2 S d}{I_y}$$

where the aerodynamic coefficients are given by:

$$C_n = a_n\alpha^3 + b_n\alpha^2 + c_n\left(2 + \frac{M}{3}\right)\alpha + d_n\delta$$

$$C_m = a_m\alpha^3 + b_m\alpha^2 - c_m\left(7 - \frac{8}{3}M\right)\alpha + d_m\delta$$

$$U = M * ss * \cos(\alpha)$$

The properties of the missile are defined as follows.

$p_0 = 973.3 lb/ft^2$	Pressure at 20000 ft
$ss = 1037.1 ft/s$	Speed of sound at 20000 ft
$S = 0.44 ft^2$	Reference area
$d = 0.75 ft$	Diameter
$mass = 13.98 slug$	Mass
$M = 2$	Mach number

$$I_y = 182.5\, slug \cdot ft^2 \qquad \text{Pitch moment of inertia}$$

The coefficients are given by:

$$a_n = 0.000103\text{deg}^{-3},\ b_n = -0.00945\text{deg}^{-2},\ c_n = -0.1696\text{deg}^{-1},\ d_n = -0.034\text{deg}^{-1}$$
$$a_m = 0.000215\text{deg}^{-3},\ b_m = -0.0195\text{deg}^{-2},\ c_m = 0.051\text{deg}^{-1},\ d_m = -0.206\,\text{deg}^{-1}$$

The nonlinear equations are written in quasi-LPV form as follows.

$$\begin{bmatrix}\dot{\alpha}\\ \dot{q}\end{bmatrix} = \begin{bmatrix} 0.0207\left(a_n\alpha^2 + b_n\alpha + c_n\left(2+\frac{M}{3}\right)\right)M\left(1-\frac{\alpha^2}{2}\right) & 1 \\ 1.232\left(a_m\alpha^2 + b_m\alpha - c_m\left(7-\frac{8}{3}M\right)\right)M^2 & 0 \end{bmatrix}\begin{Bmatrix}\alpha\\ q\end{Bmatrix} + \begin{bmatrix} 0.0207 d_n M\left(1-\frac{\alpha^2}{2}\right) \\ 1.232 d_m M^2 \end{bmatrix}\delta$$

The nominal values of M and α are 2 and 0 radians, respectively.

The LFT model obtained for this example has a total dimension of 10 (with 4 occurrences for α and 6 occurrences for M). These results are compared to other software tools in Section 4.2.

3.3 Generic Example

In this section, an extension of a physics-based model is considered. The LPV model for this example involves: three uncertain parameters, one parameter with maximum degree 3 and the other two parameters with maximum degree 2, and all associated cross-product terms. The LPV model is given below.

$$S(x,y,z) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} x^2 + \begin{bmatrix} 0 & 0 & 2 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -2 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} y^2$$

$$+ \begin{bmatrix} 0 & 0 & 0 & -4 & 0 & 0 \\ 0 & 0 & 0 & 0 & -3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} z^2 + \begin{bmatrix} 0 & 0 & 0 & 0 & -3 & 0 \\ 0 & 0 & -2 & -4 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \end{bmatrix} xy + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 \end{bmatrix} xz + \begin{bmatrix} 0 & -2 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -3 & 0 \end{bmatrix} yz$$

$$+ \begin{bmatrix} 0 & 2 & 1 & 0 & 4 & 0 \\ -1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 3 & 0 & -1 & 0 & 6 \end{bmatrix} x^3 + \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 4 & 1 & 0 \\ 0 & 3 & 0 & 0 & 0 & 1 \end{bmatrix} x^2y + \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 3 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 2 & 0 & 0 \end{bmatrix} x^2z + \begin{bmatrix} 0 & 1 & 1 & -1 & 0 & 1 \\ 1 & 0 & -1 & 1 & 1 & 0 \\ 1 & 4 & 0 & 0 & 3 & 0 \end{bmatrix} xy^2$$

$$+ \begin{bmatrix} 0 & 3 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix} y^2z + \begin{bmatrix} 1 & 1 & 0 & 0 & 2 & 2 \\ 1 & 0 & 1 & 0 & 3 & 0 \\ 0 & -1 & 0 & 2 & 4 & 0 \end{bmatrix} xz^2 + \begin{bmatrix} 2 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 2 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} yz^2 + \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ 1 & 2 & 0 & 1 & 1 & 0 \end{bmatrix} xyz$$

$$+ \begin{bmatrix} 0 & 0 & 0 & 2 & 0 & 1 \\ 1 & 0 & 0 & 3 & 1 & -1 \\ -1 & 0 & 5 & 0 & 0 & 0 \end{bmatrix} y^2z^2 + \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & -1 & 0 & 2 & 1 & 5 \\ 0 & 1 & 0 & 0 & 0 & 2 \end{bmatrix} x^2yz + \begin{bmatrix} 1 & -1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 \\ 5 & 0 & 0 & -1 & 0 & 1 \end{bmatrix} xy^2z + \begin{bmatrix} 0 & 0 & 2 & 1 & 0 & 1 \\ 2 & 1 & 4 & -1 & 1 & 0 \\ 1 & 0 & 1 & 0 & -1 & -1 \end{bmatrix} xyz^2$$

$$+ \begin{bmatrix} 0 & 1 & 2 & 0 & 1 & 1 \\ 1 & 0 & 3 & 3 & 1 & -1 \\ 0 & -1 & 0 & 4 & 1 & 1 \end{bmatrix} x^3y + \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & -1 & 0 & -1 \end{bmatrix} x^3z + \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & -1 & -1 & 1 & 0 \\ 0 & 1 & 0 & 1 & -1 & 0 \end{bmatrix} x^2y^2z + \begin{bmatrix} 0 & -1 & 0 & -1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} x^2yz^2$$

$$+\begin{bmatrix} 0 & 0 & -1 & 2 & 1 & 0 \\ 1 & 1 & 1 & 3 & 0 & 0 \\ 1 & 0 & 0 & 4 & 0 & 1 \end{bmatrix} xy^2z^2 + \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 2 & -1 & 0 & 1 & 1 & 0 \\ 3 & 1 & 0 & -1 & 0 & 0 \end{bmatrix} x^3yz + \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 2 & 1 & 1 & 0 & 0 & 1 \\ 3 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} x^3y^2 + \begin{bmatrix} 1 & 0 & 2 & 1 & 2 & 1 \\ 0 & 4 & 1 & -1 & 1 & 1 \\ -1 & 0 & 1 & 2 & -1 & 1 \end{bmatrix} x^3z^2$$

$$+\begin{bmatrix} 1 & 2 & 0 & 1 & 0 & 0 \\ 0 & 3 & 0 & -1 & 1 & 5 \\ 1 & 0 & 4 & 0 & -1 & 0 \end{bmatrix} x^2y^2z^2 + \begin{bmatrix} 0 & 1 & 3 & 1 & 0 & 0 \\ 2 & 0 & 0 & -1 & 1 & 0 \\ 0 & 4 & 0 & 1 & 0 & 1 \end{bmatrix} x^3y^2z + \begin{bmatrix} 3 & 0 & 1 & -1 & 5 & 1 \\ 0 & 4 & 1 & 1 & 1 & 1 \\ 1 & 0 & -1 & 2 & 1 & 5 \end{bmatrix} x^3yz^2 + \begin{bmatrix} 1 & 2 & 0 & 0 & 1 & -1 \\ 5 & 0 & 3 & 0 & -1 & 2 \\ 1 & 0 & 0 & 4 & 1 & -1 \end{bmatrix} x^3y^2z^2$$

The LFT model obtained for this example has a dimension of 45 (with 9 occurrences for x, 24 occurrences for y, and 12 occurrences for z). These results are compared to other software tools in the following sub-section.

4. LFT Modeling Tool Comparison

In this section, LFT models are obtained for the examples of Section 3 using two available software tools (developed by ONERA and MuSyn, Inc.) and the results are compared to those obtained using the matrix-based numerical software tool presented in this paper. The LFT modeling tool developed by ONERA (see Reference [23]) and that developed by MuSyn, Inc. (see Reference [6]) are considered, and a brief description of each is provided in subsections 4.1 and 4.2, respectively. The comparison of results is presented in subsection 4.3.

4.1 LFT Modeling Tool Developed by ONERA

The Linear Fractional Representation (LFR) toolbox was developed by J-F Magni (see Reference [23]) based on an object-oriented realization technique (see Reference [28]). The description provided herein is taken from Reference [23]. In the toolbox, the following LFT modeling techniques have been implemented: Morton's method (see Reference [8]), Horner factorization (see Reference [29]), and tree decomposition (see Reference [30]). The tree decomposition method is recommended in Ref. [23] as being the most efficient. The software also utilizes a symbolic method (based on Maple, or the symbolic tool within MATLAB) for algebraic manipulation. Morton's method can represent linear parameter dependent systems as an LFT model using singular value decomposition. This method is applied to a polynomial parameter dependent system, including rational functional forms of the parameter, which results in the Δ matrix containing rational functions of the uncertain parameters (instead of the parameters themselves). The Redheffer's star product is then used to obtain the usual form for Δ. Horner factorization concerns single variable polynomials, and its objective consists of avoiding calculation of all the powers of each uncertain parameter. The tree decomposition is a generalization of the idea consisting of factorizing parameters so that they appear a minimum number of times as is possible before proceeding to the realization. For a detailed description of these methods, the reader is referred to Ref.[23] and its references. For the examples presented in this paper, the tree decomposition method was used by invoking the "symtreed" command to construct the LFT models.

The LFR toolbox also contains three different order reduction methods (see Reference [23]). The first method is a 1-D reduction approach, which consists of considering that each uncertain parameter plays the role of 1/s. For each parameter, a balanced realization is applied to reduce its size. The second approach is an n-D reduction method developed by Carolyn Beck et. al. (see References [15] – [17]), which consists of considering controllability and observability of the uncertain system. When the system matrices (A, B, C, D) have an uncontrollable or unobservable space, the algorithm can calculate a transformation to produce a null block on the B matrix (uncontrollable) or a null block on the C matrix (unobservable). Generally, the n-D approach is less conservative than the 1-D approach, since it treats simultaneously all the parameters of the uncertain block [17]. The third reduction method is the generalized Gramian approach, which consists of considering a generalized Gramian defined in an LMI system as follows.

$$\begin{aligned} &X > 0, \\ &M_{11}^T X M_{11} - X + M_{21}^T M_{21} < 0 \\ &Y > 0 \\ &M_{11} Y M_{11}^T - Y + M_{12} M_{12}^T < 0 \end{aligned}$$

After using the similarity transformation, singular values less than the given tolerance are truncated and the order of the model is thereby reduced. A detailed description is provided in Ref.[23] and its references.

For practical use of each reduction method, note that the n-D reduction method is computationally fast and the user should define a tolerance level of uncontrollability and unobservability. When the tolerance level is high (e.g., 1e-4 instead of 1e-12), the reduced LFT model may not capture the original dynamics. The user should therefore define the appropriate tolerance level for each problem. Using the Gramian approach, the user can approximate the system based on singular values of the Gramian (X and Y). Note, however, that the computation of the Gramian can have a high computational cost (calculation time) to solve the LMI optimization (see Reference [23]).

In this paper, the n-D reduction method is applied to the LFT models obtained for the examples using the three modeling tools being compared (i.e., the NASA, ONERA, and MuSyn tools). The 1-D reduction approach is also applied to the ONERA results for comparison to the 1-D results obtained using the method presented in this paper.

4.2 LFT Modeling Tool Developed by MuSYN, Inc. (and Included in the Matlab 7.0.4 Robust Tool)

The following description about building uncertainty models using the Robust Control Toolbox in MATLAB 7.0.4 is taken from Chapter 6 of Ref.[6]. In this paper, we used the Robust Control Toolbox to generate LFT models of the examples, using the function "ureal". The function generates uncertain atoms for uncertain real parameters. Uncertain atoms are used to form uncertain matrix objects and system objects. Note that each uncertain atom is written in an LFT block. An LFT model of a matrix object is built up from uncertain atoms, depending on the sequence of operations in their construction. Note that different ways of matrix construction in terms of the uncertain atoms may generate different sizes of uncertainty blocks in the LFT models.

In the Robust Control Toolbox, there are several reduction methods for LFT models. The command "simplify" or "AutoSimplify", as an option to the function "ureal", can reduce the uncertainty block size. The AutoSimplify parameter can be set to "off", "basic", or "full". In the "off" case, no simplification is attempted. In this paper, the models obtained for the "off" case are referred to as having "no reduction". In the "basic" case, fairly simple schemes to detect and eliminate non-minimal representations are used (such as removal of zero rows and columns), and in the "full" case, numerical-based methods similar to truncated balanced realizations are used, with a very tight tolerance to minimize error. The AutoSimplify property of each uncertain atom dictates the types of computations that are performed to generate an LFT model of the uncertain matrix or system.

4.3 Comparison of LFT Modeling Results

The LFT results obtained using the three software tools for the example problems presented in Section 3 are summarized in Table 1. It should be noted that each of the above methods produces an excellent representation of the given LPV model both before and after model reduction[‡‡]. As indicated in the Table, the numerical matrix-based method and software tool presented in this paper produced an LFT model (before reduction or after 1-D reduction) that was comparable or lower in dimension for each of the examples than the other two tools. Models of substantially lower order (prior to reduction) were obtained using the NASA and ONERA tools, as compared to the MuSYN tool, for the F-16 and Generic examples – which involved the most complex LPV models. Using the n-D reduction method developed by Carolyn Beck (see References [15] – [17]) and included in the ONERA tool, the LFT models can be reduced to about the same low order despite what modeling tools were used. The exceptions are that the MUSYN "full" reduction option produced a lower-order LFT for the F-16 example (by 2 parameter repetitions) and a higher-order model for the Missile example (by 5 parameter repetitions).

[‡‡] The H-infinity norm was computed for each model at corner points associated with the uncertain parameters and over a large frequency range (0.001 – 100,000 rad/sec). The maximum deviation of the LFT and LPV models was on the order of $1x10^{-5}$.

Table 1. Comparison of LFT Modeling Results

Example: / Tool:	F-16 (V, γ)			Missile (α, M)			Generic (x, y, z)			
	n_Δ	n_V	n_γ	n_Δ	n_α	n_M	n_Δ	n_x	n_y	n_z
NASA ("lft_model")										
No Reduction	86	31	55	12	4	8	94	9	64	21
1-D Reduction (NASA)	67	27	40	10	4	6	45	9	24	12
n-D Reduction (ONERA)	55[§§]	22	33	9	4	5	45	9	24	12
ONERA ("symtreed")										
No Reduction	72	24	48	11	6	5	110	9	24	77
1-D Reduction (ONERA)	67	23	44	9	4	5	109	9	24	76
n-D Reduction (ONERA)	56	23	33	9	4	5	46	9	24	13
MUSYN ("ureal")										
No Reduction	560	280	280	16	8	8	738	318	210	210
"Basic" Reduction (MUSYN)	235	106	129	16	8	8	579	268	173	138
"Full" Reduction (MUSYN)	53	22	31	14	8	6	45	9	24	12
n-D Reduction (ONERA)	53	22	31	14	8	6	45	9	24	12

[§§] Note: Reversing the order of V and γ parameters resulted in a dimension of 54 (with $n_\gamma = 22$ and $n_V = 32$) after n-D reduction (using the ONERA tool).

The results reported in Table 1 represent the best results that we obtained using the MuSyn and ONERA tools. For the ONERA tool, the tree-decomposition method resulted in the LFT model of lowest dimension, whereas the "lfrs" command produced the same LFT model dimension as the MuSyn tool with the "off" option. Moreover, both methods (i.e., the "lfrs" in the ONERA tool and "ureal" in the MuSyn tool) yield LFT models whose reduction depends highly on how the LPV model is represented in terms of the uncertain parameters. That is, the 1-D and n-D reduction results using the ONERA tool and the "full" and "basic" reduction options in the MuSyn tool appear to be highly dependent on the equation representation of the LPV model. To illustrate this, the Generic Example was equivalently implemented using the following three representations.

LPV Representation 1: Using a For-loop statement, the equations were written in low to high order with the parameter multiplications occurring in alphabetical order, as follows.

```
LPV1 = zeros(3,6);
for k= 1:3
 for j= 1:3
  for i = 1:4
   if ~isempty(S{i,j,k})
    LPV1 = LPV1 + S{i,j,k}*x^(i-1)*y^(j-1)*z^(k-1);
   end
  end
 end
end
```

LPV Representation 2: Without using a For-loop, the equations were written in ascending order with alphabetical parameter order, such as x,y,z, as follows.

```
LPV2 =  S{3,1,1}*x^2    +  S{1,3,1}*y^2      + S{1,1,3}*z^2     +  S{2,2,1}*x*y  + S{2,1,2}*x*z +...
        S{1,2,2}*y*z     + S{4,1,1}*x^3      + S{3,2,1}*x^2*y   + S{3,1,2}*x^2*z + ...
        S{2,3,1}*x*y^2    + S{1,3,2}*y^2*z    + S{2,1,3}*x*z^2   + S{1,2,3}*y*z^2 +...
        S{2,2,2}*x*y*z    + S{3,3,1}*x^2*y^2  + S{3,1,3}*x^2*z^2 + ...
        S{1,3,3}*y^2*z^2  + S{3,2,2}*x^2*y*z  + S{2,3,2}*x*y^2*z  + ...
        S{2,2,3}*x*y*z^2  + S{4,2,1}*x^3*y    + S{4,1,2}*x^3*z   + S{3,3,2}*x^2*y^2*z +...
        S{3,2,3}*x^2*y*z^2 + S{2,3,3}*x*y^2*z^2 + S{4,2,2}*x^3*y*z + ...
        S{4,3,1}*x^3*y^2  + S{4,1,3}*x^3*z^2  + S{3,3,3}*x^2*z^2*y^2 + ...
        S{4,3,2}*x^3*y^2*z + S{4,2,3}*x^3*y*z^2 + S{4,3,3}*x^3*y^2*z^2 ;
```

LPV Representation 3: Without using a For-loop, the equations were written without any particular order.

```
LPV3 =  S{3,1,1}*x^2   + S{1,3,1}*y^2      + S{1,1,3}*z^2     + S{2,2,1}*x*y  + S{2,1,2}*x*z +...
        S{1,2,2}*y*z     + S{4,1,1}*x^3      + S{3,2,1}*y*x^2   + S{3,1,2}*z*x^2 + ...
        S{2,3,1}*x*y^2    + S{1,3,2}*z*y^2    + S{2,1,3}*x*z^2   + S{1,2,3}*y*z^2 +...
        S{2,2,2}*x*y*z    + S{3,3,1}*(x^2)*y^2 + S{3,1,3}*(x^2)*z^2 + ...
        S{1,3,3}*(y^2)*z^2 + S{3,2,2}*(x^2)*y*z + S{2,3,2}*x*y^2*z + ...
        S{2,2,3}*x*y*z^2  + S{4,2,1}*y*x^3    + S{4,1,2}*z*x^3   + S{3,3,2}*z*(x^2)*y^2 +...
        S{3,2,3}*y*(x^2)*z^2 + S{2,3,3}*x*(y^2)*z^2 + S{4,2,2}*y*z*x^3 + ...
        S{4,3,1}*(x^3)*y^2 + S{4,1,3}*(x^3)*z^2 + S{3,3,3}*(x^2)*z^2*y^2 + ...
        S{4,3,2}*z*(x^3)*y^2 + S{4,2,3}*y*(x^3)*z^2 + ...
        S{4,3,3}*(y^2)*(x^3)*z^2 ;
```

Although the LPV1, LPV2, and LPV3 equations are exactly the same in terms of x, y, and z, the resulting LFT models (after reduction) were very different, as shown in Table 2. Based on the results for this example, LPV Representation 1 produced the LFT models of lowest dimension (and were therefore included in Table 1). Note that the ONERA tree-decomposition produced the same LFT model for each representation. The NASA method is also impervious to the equation format, because the LPV model does not need to be written in symbolic form – although re-defining the parameter order in the S matrix can have a minor impact on LFT model dimension for some problems.

Table 2. LFT Modeling Results Using ONERA and MuSyn Tools for Three Equivalent LPV Representations of the Generic Example

LPV Rep. / Tool	**LPV1**				**LPV2**				**LPV3**			
	n_Δ	n_x	n_y	n_z	n_Δ	n_x	n_y	n_z	n_Δ	n_x	n_y	n_z
ONERA ("symtreed")												
No Reduction	110	9	24	77	110	9	24	77	110	9	24	77
1-D Reduction	109	9	24	76	109	9	24	76	109	9	24	76
n-D Reduction	46	9	24	13	46	9	24	13	46	9	24	13
ONERA ("lfrs")												
No Reduction	738	318	210	210	738	318	210	210	738	318	210	210
1-D Reduction	445	238	148	14	448	285	135	28	471	290	135	46
n-D Reduction	45	9	24	12	57	9	30	18	96	45	27	24
MuSyn ("ureal")												
No Reduction	738	318	210	210	738	318	210	210	738	318	210	210
Basic	579	268	173	138	632	268	182	182	608	270	169	169
Full	45	9	24	12	57	9	30	18	96	45	27	24

5. Conclusion

Methods and software tools for developing linear fractional transformation (LFT) models for uncertain systems were considered in this paper, as a precursor to applying formal robustness analysis methods to control upset prevention and recovery systems as part of a validation process being developed for potential use in their ultimate certification. Such systems (developed for failure detection, identification, and reconfiguration, as well as upset recovery) need to be evaluated over broad regions of the flight envelope and under extreme flight conditions, and should include various sources of uncertainty. However, formulation of LFT models for representing system uncertainty can be very difficult for complex parameter-dependent systems. A numerical matrix-based LFT modeling method and preliminary software tool were presented and evaluated in this paper for several example problems and in comparison to other available methods and software tools. The examples included an F-16 aircraft at an extreme flight condition (i.e., near a stall bifurcation), a missile model problem, and a generic example designed to be a difficult modeling problem (particularly relative to cross-product terms). The numerical modeling method and preliminary software tool presented in this paper compared favorably for each of the example problems relative to the other methods considered. The matrix-based modeling approach therefore appears to be promising. Several areas for further refinement of the preliminary software tool were also discussed. Further research will focus on these refinements, as well as applying the tool to robustness analysis studies for control upset prevention and recovery technologies. These studies will provide risk mitigation for high-risk flight testing under extreme and/or loss-of-control flight regimes, aircraft failure and damage, and other adverse or upset conditions. These kinds of high-risk tests will be performed at NASA Langley using a dynamically scaled transport aircraft model, as part of the Airborne Subscale Transport Aircraft Research (AirSTAR) Testbed. Open-loop tests will be performed to validate and further investigate vehicle dynamics under extreme/upset conditions, and closed-loop tests will be performed for failure accommodation, upset recovery, and damage mitigation.

A possible advantage of the numerical LFT modeling method presented in this paper is its potential future use as part of an online robustness analysis tool for risk mitigation during high-risk flight tests or for onboard aircraft applications. An on-line robustness analysis tool is currently being developed to provide risk mitigation during flight tests involving the AirSTAR Testbed, and a future extension could possibly utilize an online uncertainty modeling capability to update the system model being used for analysis. Onboard modeling and robustness analysis methods for future transport aircraft applications may also be feasible.

Appendix

The LPV model for the F-16 example presented in Section 3 is given as follows.

$$A(s_1, s_2) = AA00 + AA10s_1 + AA01s_2 + AA11s_1s_2 + AA20s_1^2 + AA02s_2^2 + AA21s_1^2s_2 + AA12s_1s_2^2 + AA30s_1^3 + AA03s_2^3$$

$$B(s_1, s_2) = BB00 + BB10s_1 + BB01s_2 + BB11s_1s_2 + BB20s_1^2 + BB02s_2^2 + BB21s_1^2s_2 + BB12s_1s_2^2 + BB30s_1^3 + BB03s_2^3$$

$$C(s_1, s_2) = CC00 + CC10s_1 + CC01s_2 + CC11s_1s_2 + CC20s_1^2 + CC02s_2^2 + CC21s_1^2s_2 + CC12s_1s_2^2 + CC30s_1^3 + CC03s_2^3$$

$$D(s_1, s_2) = DD00 + DD10s_1 + DD01s_2 + DD11s_1s_2 + DD20s_1^2 + DD02s_2^2 + DD21s_1^2s_2 + DD12s_1s_2^2 + DD30s_1^3 + DD03s_2^3$$

The nonzero coefficient matrices are given below.

$$AA00 = \begin{bmatrix} 0 & 0 & 0 & 1.0000 & 0 & 1.1702 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.5393 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.0042 & 0 & -0.0849 & 0 & 0 & 0.0157 \\ 0 & 0 & 0 & 0 & -0.0152 & 0 & 0 & 0.0127 & 0 \\ 0 & 0 & 0 & -0.0001 & 0 & -0.0131 & 0 & 0 & -0.0668 \\ 0 & -4.1662 & 0 & 0 & -10.2531 & 0 & -0.2358 & -9.4733 & 0 \\ 0 & 0.0003 & 0 & 0 & 0.9241 & 0 & 0 & 0.1824 & 0 \\ 0.0206 & 0 & 0 & 0.7607 & 0 & -0.6589 & 0 & 0 & -0.0822 \end{bmatrix}$$

$$AA01 = 10^3 \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -0.0251 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -0.0191 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.0002 & 0 & 0.0088 & 0 & 0 & 0.0016 \\ 0 & 0 & 0 & 0 & -0.0023 & 0 & 0 & -0.0006 & 0 \\ 0 & 0 & 0 & 0.0003 & 0 & 0.0003 & 0 & 0 & 0.0016 \\ 0 & 0 & 0 & 0 & -0.1232 & 0 & -0.0003 & -2.3492 & 0 \\ 0 & 0 & 0 & 0 & -0.0013 & 0 & 0 & -0.0151 & 0 \\ 0.0003 & 0 & 0 & -0.0056 & 0 & -0.0063 & 0 & 0 & 0.0002 \end{bmatrix}$$

$$AA02 = 10^4 \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -0.0779 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -0.0506 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.0018 & 0 & 0.0159 & 0 & 0 & -0.0151 \\ 0 & 0 & 0 & 0 & 0.0009 & 0 & 0 & 0.0019 & 0 \\ 0 & 0 & 0 & 0.0002 & 0 & 0.0006 & 0 & 0 & 0.0034 \\ 0 & 0.0003 & 0 & 0 & 0.6992 & 0 & 0.0096 & -3.9688 & 0 \\ 0 & 0.0001 & 0 & 0 & -0.0019 & 0 & -0.0002 & -0.0543 & 0 \\ 0.0004 & 0 & 0 & -0.0300 & 0 & -0.0272 & 0 & 0 & -0.0019 \end{bmatrix}$$

$$AA03 = 10^5 \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -0.3310 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -0.1872 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.0110 & 0 & 0.0055 & 0 & 0 & -0.1278 \\ 0 & 0 & 0 & 0 & 0.0256 & 0 & 0 & -0.0059 & 0 \\ 0 & 0 & 0 & -0.0016 & 0 & 0.0003 & 0 & 0 & 0.0083 \\ 0 & 0.0025 & 0 & 0 & 5.8716 & 0 & 0.0869 & -4.0137 & 0 \\ 0 & 0.0011 & 0 & 0 & 0.0115 & 0 & -0.0006 & -0.1937 & 0 \\ -0.0008 & 0 & 0 & -0.1537 & 0 & -0.1127 & 0 & 0 & -0.0121 \end{bmatrix}$$

$$AA10 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0.2958 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.2249 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0030 & 0 & -0.1147 & 0 & 0 & -0.0212 \\ 0 & 0 & 0 & 0 & 0.0304 & 0 & 0 & 0.0082 & 0 \\ 0 & 0 & 0 & -0.0035 & 0 & -0.0039 & 0 & 0 & -0.0218 \\ 0 & 0 & 0 & 0 & 1.5240 & 0 & 0.0016 & 30.8244 & 0 \\ 0 & 0 & 0 & 0 & 0.0173 & 0 & 0.0005 & 0.2009 & 0 \\ -0.0032 & 0 & 0 & 0.0722 & 0 & 0.0815 & 0 & 0 & -0.0007 \end{bmatrix}$$

$$AA11 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 18.3487 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 11.9117 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.4309 & 0 & -3.1205 & 0 & 0 & 4.1293 \\ 0 & 0 & 0 & 0 & -0.4923 & 0 & 0 & -0.5664 & 0 \\ 0 & 0 & 0 & -0.0330 & 0 & -0.1094 & 0 & 0 & -0.6622 \\ 0 & -0.0196 & 0 & 0 & -194.1753 & 0 & -2.4965 & 740.3101 & 0 \\ 0 & -0.0129 & 0 & 0 & 0.3508 & 0 & 0.0413 & 12.3560 & 0 \\ -0.0864 & 0 & 0 & 7.2006 & 0 & 6.4003 & 0 & 0 & 0.5259 \end{bmatrix}$$

$$AA12 = 10^4 \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0.1169 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.0661 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0040 & 0 & -0.0008 & 0 & 0 & 0.0460 \\ 0 & 0 & 0 & 0 & -0.0092 & 0 & 0 & 0.0029 & 0 \\ 0 & 0 & 0 & 0.0006 & 0 & -0.0001 & 0 & 0 & -0.0029 \\ 0 & -0.0008 & 0 & 0 & -2.1076 & 0 & -0.0317 & 1.2821 & 0 \\ 0 & -0.0004 & 0 & 0 & -0.0046 & 0 & 0.0002 & 0.0675 & 0 \\ 0.0004 & 0 & 0 & 0.0548 & 0 & 0.0396 & 0 & 0 & 0.0044 \end{bmatrix}$$

$$AA20 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -0.1080 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -0.0701 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.0026 & 0 & 0.0137 & 0 & 0 & -0.0281 \\ 0 & 0 & 0 & 0 & 0.0049 & 0 & 0 & 0.0042 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.0005 & 0 & 0 & 0.0028 \\ 0 & 0.0001 & 0 & 0 & 1.3392 & 0 & 0.0161 & -2.9008 & 0 \\ 0 & 0.0001 & 0 & 0 & -0.0014 & 0 & -0.0002 & -0.0687 & 0 \\ 0.0004 & 0 & 0 & -0.0430 & 0 & -0.0373 & 0 & 0 & -0.0035 \end{bmatrix}$$

$$AA21 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -13.7634 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -7.7846 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.4737 & 0 & 0.0278 & 0 & 0 & -5.4545 \\ 0 & 0 & 0 & 0 & 1.0739 & 0 & 0 & -0.4604 & 0 \\ 0 & 0 & 0 & -0.0765 & 0 & 0.0050 & 0 & 0 & 0.3573 \\ 0 & 0.0782 & 0 & 0 & 249.1658 & 0 & 3.8505 & -159.6116 & 0 \\ 0 & 0.0452 & 0 & 0 & 0.5933 & 0 & -0.0243 & -7.8976 & 0 \\ -0.0481 & 0 & 0 & -6.5172 & 0 & -4.6449 & 0 & 0 & -0.5303 \end{bmatrix}$$

$$AA30 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0.0540 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.0306 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0019 & 0 & -0.0002 & 0 & 0 & 0.0213 \\ 0 & 0 & 0 & 0 & -0.0040 & 0 & 0 & 0.0024 & 0 \\ 0 & 0 & 0 & 0.0003 & 0 & 0 & 0 & 0 & -0.0016 \\ 0 & -0.0003 & 0 & 0 & -0.9675 & 0 & -0.0155 & 0.7910 & 0 \\ 0 & -0.0002 & 0 & 0 & -0.0025 & 0 & 0.0001 & 0.0312 & 0 \\ 0.0002 & 0 & 0 & 0.0258 & 0 & 0.0182 & 0 & 0 & 0.0021 \end{bmatrix}$$

$$BB00 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -0.0167 & -0.3934 & 0.3934 & 0.0045 & 0 \\ 0 & -0.0111 & -0.0111 & 0 & 0 \\ 0.0009 & -0.0061 & 0.0061 & -0.0014 & 0 \\ 0 & -2.3330 & -2.3330 & 0 & 0.0010 \\ 0 & -0.0038 & -0.0038 & 0 & 0 \\ 0.0044 & 0 & 0 & 0.0121 & 0 \end{bmatrix}$$

$$BB01 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -0.4317 & 0 & 0 & 0.1210 & 0 \\ 0 & -0.8317 & -0.8317 & 0 & 0 \\ 0.1155 & 0 & 0 & -0.1519 & 0 \\ 0 & 20.8448 & 20.8448 & 0 & 0.0116 \\ 0 & -0.3167 & -0.3167 & 0 & 0.0001 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$BB02 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -50.6223 & -214.8311 & 214.8311 & 8.0884 & 0 \\ 0 & -12.7580 & -12.7580 & 0 & 0 \\ 1.5743 & -3.3433 & 3.3433 & -6.7643 & 0 \\ 0 & -645.2558 & -645.2558 & 0 & 0.4809 \\ 0 & -8.5432 & -8.5432 & 0 & 0.0063 \\ 1.2085 & 0 & 0 & 3.2993 & 0 \end{bmatrix}$$

$$BB03 = 10^4 \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -0.3677 & -1.7900 & 1.7900 & 0.0527 & 0 \\ 0 & -0.2239 & -0.2239 & 0 & 0 \\ -0.0019 & -0.0279 & 0.0279 & -0.0371 & 0 \\ 0 & -3.9126 & -3.9126 & 0 & 0.0019 \\ 0 & -0.0678 & -0.0678 & 0 & 0 \\ 0.0101 & 0 & 0 & 0.0275 & 0 \end{bmatrix}$$

$$BB10 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0.0054 & -0.0060 & 0.0060 & -0.0015 & 0 \\ 0 & 0.0106 & 0.0106 & 0 & 0 \\ -0.0015 & -0.0001 & 0.0001 & 0.0020 & 0 \\ 0 & -0.3065 & -0.3065 & 0 & -0.0002 \\ 0 & 0.0041 & 0.0041 & 0 & -0 \\ 0 & 0 & 0 & 0.0001 & 0 \end{bmatrix}$$

$$BB11 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1.2691 & 5.7140 & -5.7140 & -0.1971 & 0 \\ 0 & 0.2341 & 0.2341 & 0 & 0 \\ -0.0272 & 0.0889 & -0.0889 & 0.1578 & 0 \\ 0 & 20.0750 & 20.0750 & 0 & -0.0113 \\ 0 & 0.1866 & 0.1866 & 0 & -0.0002 \\ -0.0321 & 0 & 0 & -0.0878 & 0 \end{bmatrix}$$

$$BB12 = 10^3 \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0.1329 & 0.6475 & -0.6475 & -0.0190 & 0 \\ 0 & 0.0865 & 0.0865 & 0 & 0 \\ 0.0008 & 0.0101 & -0.0101 & 0.0133 & 0 \\ 0 & 1.3147 & 1.3147 & 0 & -0.0007 \\ 0 & 0.0253 & 0.0253 & 0 & 0 \\ -0.0037 & 0 & 0 & -0.0100 & 0 \end{bmatrix}$$

$$BB20 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -0.0080 & -0.0383 & 0.0383 & 0.0012 & 0 \\ 0 & -0.0009 & -0.0009 & 0 & 0 \\ 0.0001 & -0.0006 & 0.0006 & -0.0009 & 0 \\ 0 & -0.1536 & -0.1536 & 0 & 0.0001 \\ 0 & -0.0010 & -0.0010 & 0 & 0 \\ 0.0002 & 0 & 0 & 0.0006 & 0 \end{bmatrix}$$

$$BB21 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -1.5956 & -7.7385 & 7.7385 & 0.2281 & 0 \\ 0 & -1.1246 & -1.1246 & 0 & 0 \\ -0.0097 & -0.1204 & 0.1204 & -0.1599 & 0 \\ 0 & -13.9688 & -13.9688 & 0 & 0.0078 \\ 0 & -0.3181 & -0.3181 & 0 & 0.0002 \\ 0.0438 & 0 & 0 & 0.1195 & 0 \end{bmatrix}$$

$$BB30 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0.0064 & 0.0305 & -0.0305 & -0.0009 & 0 \\ 0 & 0.0049 & 0.0049 & 0 & 0 \\ 0 & 0.0005 & -0.0005 & 0.0006 & 0 \\ 0 & 0.0453 & 0.0453 & 0 & 0 \\ 0 & 0.0013 & 0.0013 & 0 & 0 \\ -0.0002 & 0 & 0 & -0.0005 & 0 \end{bmatrix}$$

$$CC00 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Acknowledgments

The multivariate polynomial modeling method (using orthogonal functions) developed by Gene Morelli (see References [31] – [32]) was utilized in developing the LPV model for the F-16 aircraft example, and is a primary component to the formulation of LPV models used at NASA Langley.

References

[1] Jordan, Thomas L., Langford, William M., and Hill, Jeffrey S.: "Airborne Subscale Transport Aircraft Research Testbed - Aircraft Model Development". *Proceedings of the Guidance, Navigation, and Control Conference*, August, 2005.

[2] Bailey, Roger M., Hostetler, Robert W., Barnes, Kevin N., Belcastro, Celeste M., and Belcastro, Christine M.: "Experimental Validation: Subscale Aircraft Ground Facilities and Integrated Test Capability**".** *Proceedings of the Guidance, Navigation, and Control Conference*, August, 2005.

[3] Fielding, Christopher; Varga, Andras; Bennani, Samir; and Selier, Michiel (Eds.), *Advanced Techniques for the Clearance of Flight Control Laws*. Springer, 2002.

[4] Belcastro, Christine M. & Chang, B-C, "Uncertainty Modeling for Robustness Analysis of Failure Detection & Accommodation Systems". *Proceedings of the IEEE American Control Conference*, May 2002.

[5] Packard, A. (Univ of California); Doyle, J. Source: "Complex structured singular value". Automatica, v 29, n 1, Jan, 1993, p 71-109, ISSN: 0005-1098 CODEN: ATCAA9

[6] Balas, G., Chiang, R., Packard, A., and Safonov, M. "Robust Control Toolbox User's Guide V.3" , The MathWorks, 2005.

[7] Morton, Blaise G., R. M. McAfoos, "A Mu-Test for Robustness Analysis of a Real-Parameter Variation Problem". *Proceedings of the American Control Conference*, pp. 135-138, 1985.

[8] Morton, B., " New Application of mu to real-parameter Variations problems", 24th IEEE Conference on Decision and Control, Fort Lauderdale, Florida, Dec. 1985, pp233-238.

[9] Steinbuch, Maarten, et. al.: "Robustness Analysis for Real and Complex Perturbations Applied to an Electro-Mechanical System". *Proceedings of the American Control Conference*, 1991.

[10] Lambrechts, Paul, et. al.: "Parametric Uncertainty Modeling using LFT's". *Proceedings of the American Control Conference*, Vol. 1, pp. 267-272, 1993.

[11] Belcastro, Christine M.: "Uncertainty Modeling of Real Parameter Variations for Robust Control Applications". Ph.D. Dissertation, Drexel University, 1994.

[12] Belcastro, Christine M.: "Parametric Uncertainty Modeling: An Overview". *Proceedings of the American Control Conference*, 1998.

[13] Belcastro, Christine M. and Chang, B.-C.: "LFT Formulation for Multivariate Polynomial Problems". *Proceedings of the American Control Conference*, 1998.

[14] Cockburn, Juan C: "Linear Fractional Representation of Systems with Rational Uncertainty". *Proceedings of the American Control Conference*, 1998.

[15] Beck, Carolyn and D'Andrea, Raffaello, "Minimality, Controllability and Observability for Uncertain Systems", Proceedings of the American Control Conference, June 1997, pp 3130-3135.

[16] Beck, Carolyn and D'Andrea, Raffaello: "Computational Study and Comparisons of LFT Reducibility Methods". *Proceedings of the American Control Conference*, 1998.

[17] Beck, Carolyn and Doyle, John, "A Necessary and Sufficient Minimality Condition for Uncertain Systems", IEEE Transactions on Automatic Control, Vol. 44, No. 10, October 1999, pp. 1802-1813.

[18] Belcastro, Christine M.: "On the Numerical Formulation of Parametric Linear Fractional Transformation (LFT) Uncertainty Models for Multivariate Matrix Polynomial Problems". NASA TM-1998-206939, November 1998.

[19] Belcastro, Christine M., Lim, Kyong B. and Morelli, Eugene A.: "Computer-Aided Uncertainty Modeling of Nonlinear Parameter-Dependent Systems, Part I: Theoretical Overview". *Proceedings of the Computer Aided Control System Design Conference*, August 1999.

[20] Belcastro, Christine M., Lim, Kyong B. and Morelli, Eugene A.: "Computer-Aided Uncertainty Modeling of Nonlinear Parameter-Dependent Systems, Part II: F-16 Example". *Proceedings of the Computer Aided Control System Design*, August 1999.

[21] Halmos, Paul R.: *Finite-Dimensional Vector Spaces*. Springer-Verlag New York, Inc., 1974

[22] Gantmacher, F. R.: *The Theory of Matrices, Vol. I*. Chelsea Publishing Company, New York, NY, 1959.

[23] Magni, J.-F., "Linear Fractional Representation Toolbox - Modeling, Order Reduction, and Gain Scheduling", ONERA Technical Report TR 6/08162 DSCD, ONERA, Systems Control and Flight Dynamics Department, July 2004.

[24] H. G. Kwatny and B.-C. Chang, "Constructing Linear Families from Parameter-Dependent Nonlinear Dynamics," *IEEE Transactions on Automatic Control*, vol. 43, pp. 1143-1147, 1998.

[25] S. Thomas, H. G. Kwatny, and B. C. Chang, "Bifurcation Analysis of Flight Control Systems," presented at 16th IFAC World Congress, Prague, 2005.

[26] E. A. Morelli, Global Nonlinear Parametric Modeling with Application to F-16 Aerodynamics. Proceedings American Control Conference, Philadelphia, pp. 997-1001, 1998.

[27] Bennani, S., Willemsen, D.M.C., and Scherer, C.W. "Robust Control of Linear Parametrically Varying Systems with Bounded Rates", Journal of Guidance, Control, and Dynamics, Vol 21, No. 6, Nov.-Dec., 1998, pp.916- 922.

[28] Terlouw, J.C, and Lambrechts, P.F., " A MATLAB Toolbox for Parameter Uncertainty Modelling", Technical Report ,CR-93455-L, National Aerospace Laboratory NLR, Amsterdam, 1993.

[29] Varga, V. and Looye, G., "Symbolic and numerical software tools for LFT-based low order Uncertainty modeling", the IEEE International Symposium on computed Aided control System Design, Kohala Coast, Hawaii, U.S.A. Aug. 1999, pp176-181.

[30] Barmish, B.R., Ackermann, A., and Hu, H.Z., "The tree structured decomposition", Conference on Information Sciences and Systems, Baltimore, MD. U.S.A. 1989

[31] Morelli, E.A. "System IDentification Programs for AirCraft (*SIDPAC)*," AIAA Paper 2002-4704, *AIAA Atmospheric Flight Mechanics Conference*, Monterey, CA, August 2002.

[32] Morelli, E.A. and DeLoach, R., "Wind Tunnel Database Development using Modern Experiment Design and Multivariate Orthogonal Functions," AIAA Paper 2003-0653, *41st AIAA Aerospace Sciences Meeting and Exhibit*, Reno, NV, January 2003.