Nonlinear Worst-Case Analysis of an LPV Controller for Approach-Phase of a Re-Entry Vehicle

Prathyush P Menon¹, Emmanuel Prempain², Ian Postlethwaite³ and Declan Bates⁴
Department of Engineering, University of Leicester, Leicester LE1 7RH, U. K.

Samir Bennani⁵
Guidance, Navigation and Control Group, ESA/ESTEC (TEC-ECN), Keplerlaan 1, 2201 AZ Noordwijk, The Netherlands

In this paper, a nonlinear robustness analysis of an LPV controller for the approach-phase of a re-entry vehicle is presented. The nonlinear longitudinal equations of motion of the NASA-HL-20 atmospheric re-entry vehicle, a benchmark provided by Deimos Space as a representative of future re-entry vehicles, constitute the open loop model. The analysis is carried out using the optimization-based worst-case analysis tools developed at University of Leicester for Phase I of the European Space Agency (ESA) project – “Robust LPV Gain Scheduling Techniques for Space Applications”. The tools make up an analysis framework using several optimization methods such as local gradient based algorithms, global evolutionary algorithms, dividing rectangles algorithm, hybrid local / global evolutionary algorithms and multi-objective algorithms. In this paper, the worst-case deviations from a predefined re-entry profile due to simultaneous variations of multiple uncertain parameters are determined by two optimization methods - hybrid differential evolution and hybrid dividing rectangles. The results demonstrate the flexibility, efficiency and reliability of the optimization-based worst-case analysis, and project it as a useful potential tool for complex controller validations in future space applications.

Nomenclature

\[
\begin{align*}
\text{AoA, } & \alpha = \text{Angle of attack [deg]} \\
\text{AoSS, } & \beta = \text{Angle of side slip [deg]} \\
\text{FPA, } & \gamma = \text{Flight path angle [deg]} \\
\text{IQC } & = \text{Integral quadratic constraints} \\
\text{LPV } & = \text{Linear Parameter Varying} \\
\text{LFR } & = \text{Linear fractional representation} \\
\text{LFT } & = \text{Linear fractional transformation} \\
\text{RLV } & = \text{Reusable Launch Vehicle} \\
\text{SOS } & = \text{Sum of squares} \\
\text{M } & = \text{Mach} \\
\text{he } & = \text{Altitude [Km]} \\
\text{q } & = \text{Pitch rate [deg/sec]} \\
\text{a}_z & = \text{Normal acceleration [m/sec}^2] \\
\end{align*}
\]

¹ Lecturer, Email: ppm6@le.ac.uk
² Lecturer, Email:ep14@le.ac.uk
³ Professor, Email: ixp@le.ac.uk
⁴ Senior Lecturer, Email: dgb3@le.ac.uk
⁵ Research Scientist,
\[ \| \cdot \|_{\infty} = \text{H-Infinity norm} \]
\[ \mu \] = Gravitational constant
\[ C_{D,L,M} \] = Coefficient of drag, lift and pitching moment respectively
\[ X_{CG}, Y_{CG}, Z_{CG} \] = Location of centre of gravity in the direction of x, y, and z-axis respectively [m]
\[ I_{YY} \] = Moment of inertia about y-axis [Kg m^2]

I. Introduction

Atmospheric re-entry is a critical and challenging phase of the RLV mission, during which the vehicle follows a predefined trajectory and approaches a landing point. During this mission phase, the vehicle travels through different atmospheric regions before finally entering the earth’s atmosphere. In each of these phases, the vehicle is subject to the effects of different extreme operating conditions and uncertainties. In many aerospace applications, extensive wind tunnel experiments can accurately reveal the vehicle’s aerodynamic characteristics during a flight condition, i.e., the uncertainties in the aerodynamic coefficients can be made relatively small. As of today, however, such experiments can’t correctly reveal the aerodynamic characteristics during a re-entry flight, while online flight identification experiments are enormously expensive. For design purposes, therefore, complex simulation models with large levels of uncertainty have to be relied upon. For example, during the approach phase of re-entry, the Mach number decreases (variation in operating condition) and the vehicle is exposed to rapid changes to aerodynamic flight parameters (uncertainties), whose values are known only within large uncertainty ranges. Hence, the control laws need to be extremely robust in tracking the re-entry profile in the presence of varying operating conditions, uncertainties and unmodelled dynamics. This is a demanding problem and robust gain scheduling methods are thus often employed to design such controllers. Traditional gain scheduling methods are inherently ad-hoc and the resulting controller does not provide any stability or performance guarantees for rapid changes in the scheduling parameters. The LPV approach, however, provides a systematic framework with which to compute gainscheduled controllers with rigorous stability and performance guarantees. The approach is capable of dealing rigorously with nonlinear systems having slow dynamic variations in their operating region. LPV systems are a class of linear systems, whose state-space descriptions are known functions of time-varying parameters. The time variation of each of the parameters is unknown a priori, but is assumed to be measurable in real-time.

In order to ensure the safety of the mission, the worst-case deviations from the defined re-entry profile must be evaluated for all expected levels of variations and uncertainties in parameters such as aerodynamic coefficients, mass, inertia and centre of gravity. Simultaneous perturbations of multiple uncertain parameters, many of which are not necessarily linear or time invariant, must be considered. The task of assessing the worst-case stability and performance by considering all possible combinations of multiple uncertain parameters is a very time consuming and expensive one. There exist several different classes of analysis techniques which may be used to conduct worst-case analysis, in particular for flight control laws. In Ref. 8, different clearance criteria and methods for worst-case assessment of flight control laws are discussed in detail. However, it is important to note that most of the analysis work reported in Ref. 8 is done from the linear systems perspective.

The classical approach to worst-case analysis usually employs analytical measures of (linear) robustness such as gain/phase margins and eigenvalue analysis. More modern approaches based on robust control theory begin by converting the given closed loop system to linear fractional transformation (LFT) based models and subsequently employ techniques such as \(\mu\)-analysis and the \(\nu\)-gap metric analysis to assess the robustness of the flight control laws to multiple sources of uncertainty (See chapter 17 and 18 of Ref. 8, Ref. 11 and references there in). The tools based on the LFT-\(\mu\) framework may be used to analyze LPV systems using the scaled small gain theorem. Nonlinear extensions of such approaches using multiplier theory, integral quadratic constraints (IQC’s) and Sum-Of-Squares (SOS) programming have also recently been developed as intermediate analysis tools. There exist many efforts to formulate as convex optimization problem, which can be solved effectively by LMI techniques. The dissipative systems theory along with parameter dependent Lyapunov functions are have also been used to analyze LPV systems. Parameter dependent Lyapunov functions that are quadratic or affine can be transformed into finite dimensional LMI problems. The conservatism resulting from such a transformation can be removed by the use of

American Institute of Aeronautics and Astronautics
092407
general classes of Lyapunov functions. However, the price to be paid for the reduction in conservatism will be the presence of computationally unattractive infinite-dimensional LMIs.

All of the above analysis methods provide only upper bounds on worst-case stability/performance and are prone to being highly conservative when applied to a closed-loop system in LPV form. In addition, all these methods require the development of simplified model representations, and cannot be applied to full high-fidelity nonlinear simulators. This is on contrast to the Monte Carlo simulation and Gridding based methods commonly used in the aerospace industry, which can be employed with linear time invariant, linear parameter varying and full nonlinear simulation models with a minimum of effort on the part of the designer. The drawback of these approaches, of course, is that the computational effort either increases exponentially with the number of considered uncertain parameters (gridding) or with the desired statistical confidence levels (Monte Carlo simulation). This could possibly become a stumbling block in case of complex nonlinear simulation models, in particular those with slow simulation speeds. Moreover, there is no actual guarantee in all of the results achieved by these methods that the worst-case uncertainty perturbation has really been found, since it is quite possible that the worst-case uncertain perturbation does not lie on the extreme points in the case of gridding, or in those random samples generated by Monte Carlo methods.

A promising approach to overcome the difficulties noted above is to leverage the power of advanced optimization algorithms. Optimization algorithms can search for worst-cases in the parameter space, which can be complex, nonlinear and nonconvex, in an intelligent, guided way. Global algorithms or more sophisticated, hybrid (local/global) optimization algorithms are necessary to identify the worst-case and avoiding getting trapped in locally optimal solutions. To search for the worst-case, a performance criterion must be defined. The optimization-based approach to worst-case analysis reformulates the problem as an equivalent norm maximization problem. Previous research work by the authors has explored the applicability of different optimization methods to the flight clearance problem for a high performance aircraft (ADMIRE). Most recently, authors have employed different optimization methods including hybrid and multi-objective optimization methods to analyze the robustness of full authority nonlinear dynamic inversion control law for a reusable launch vehicle in collaboration with Deimos Space and European Space Agency. Interval optimization method has been applied in case of flight clearance using worst-case linear eigenvalue and nonlinear Lyapunov criteria.

The optimization-based worst-case analysis tools developed at University of Leicester for Phase I of the European Space Agency (ESA) project—"Robust LPV Gain Scheduling Techniques for Space Applications"—are presented in this paper. The tools are based on the different methods proposed in Ref. 23 and 25. The tools consist of a framework using several optimization methods such as local gradient based algorithms, global evolutionary algorithms, dividing rectangles algorithm, hybrid local / global evolutionary algorithms and multi-objective algorithms. The nonlinear longitudinal equations of motion of the NASA-HL-20 atmospheric re-entry vehicle, a benchmark provided by Deimos Space as a representative of future re-entry vehicles, constitute the open loop re-entry vehicle. In contrast to the works reported previously, the reentry profile for NASA-HL-20 vehicle here is defined in terms of FPA, $he$ and $M$ and moreover, a loop shaping LPV output feedback controller is analyzed. The LPV controller is used with nonlinear simulation model. The contribution of the paper is the assessment of the flexibility, efficiency and reliability of the optimization-based worst-case approach for analyzing LPV controllers with LPV and full nonlinear simulation models, a task which most other currently available methods would not be able to handle directly.

The paper is organised as follows. Section 2 describes the re-entry vehicle simulation model, provides a summary of the associated mission requirements, and defines the LPV control law and associated analysis criteria used in the study. Section 3 provides details of the optimization-based worst-case analysis framework, and describes in detail two different optimization algorithms used in the analysis. Section 4 provides the results of the application of these methods to the worst-case analysis of a re-entry vehicle simulation model with LPV control law. The performance of the two different algorithms is compared with each other, and with standard Monte Carlo simulation techniques. Finally, some conclusions are presented in Section 5.
II. Re-entry vehicle closed loop simulation model

A. Re-entry vehicle dynamics

The simulation model considered in the present study is an implementation\(^6\) of the nonlinear equations of motion for the longitudinal dynamics of the NASA-HL-20 atmospheric re-entry vehicle, which is intended to be representative of future re-entry vehicles\(^28\). NASA’s aerodynamic database for HL20 real simulation studies\(^29\) is used, which accounts for performance variations due to Mach \((M \in [4.0, 1.5])\), Altitude \((h \in [30, 15] \text{ km})\), AoA \((\alpha \in [-2, 30] \text{ deg})\) and AoSS \((\beta \in [0, 2] \text{ deg})\). The dynamics of the re-entry vehicle are described by a nonlinear model having 12 states, which are angular rates, angle of attack, angle of sideslip, bank angle, velocity, flight path angle, heading angle, distance to earth centre, longitude and geocentric latitude. The aero-elasticity of the vehicle is neglected and considered as a rigid body. Moreover, the lateral/directional couplings and Earth’s angular velocity terms are discarded and only the coupled translational and rotational motions are considered. For convenience, the translational motion dynamics are derived using spherical position and velocity coordinates. Notations are standard as given in the nomenclature. The Earth is assumed to be an ellipsoid with mass symmetry about the polar axis and gravitational constant \( \mu_E \) \((g = \mu_E / R^2 \approx 9.81 \text{ m/s}^2)\). In this case, the equations of longitudinal motion are

\[
\begin{align*}
\dot{V} &= \frac{F_{wx}}{m} - \frac{\mu_E}{R^2} \sin \gamma \quad (5.1) \\
\dot{\gamma} &= -\frac{F_{wz}}{V m} - \frac{\mu_E}{R^2} \cos \gamma + \frac{V}{R} \cos \gamma \quad (5.2) \\
\dot{R} &= V \sin \gamma \quad (5.3) \\
\dot{\alpha} &= q + \left( \frac{F_{wz}}{gm} + \cos \gamma \right) \frac{\mu_E}{R^2 V} \quad (5.4) \\
\dot{q} &= \frac{M_s}{I_{yy}} \quad (5.5)
\end{align*}
\]

See Ref. 28 for full details regarding the implementation of the vehicle model. The reference trajectory is defined in terms of FPA, altitude and Mach number. During approach phase, the Mach number decreases from 4.0 to 1.5 in 100 seconds and the altitude changes from 28 Km to 15 Km. The predefined approach reference trajectory, adapted from Ref. 29, is shown in Figure 1.

B. LPV controller details

The inputs and outputs that are available in the benchmark example for designing a 2 degree of freedom LPV control design are depicted in Figure 2. A full order \( L_2 \) optimal output feedback LPV controller, which robustly tracks the reference trajectory given in Figure 1, is considered. The controller synthesis is done using the LPV design tools developed at University of Leicester in the same project, which is based on an extended McFarlane Glover loop shaping methodology for polytopic LPV systems. See Ref. 27 for details on the controller synthesis procedure. The LPV controller is realized as linear fractional representation (LFR) model\(^11\). The reference trajectory is dependent on Mach number, and therefore, it is considered as the scheduling parameter while designing the LPV controllers. It means that the controller depends affinely on Mach number (scheduling parameter). The output measurements used for controller synthesis are FPA, AoA, and pitch rate. It is assumed that precise information on variation of Mach number is always available. The LPV controller outputs are elevator and speed brake control signals. Control surface mixer logic (control allocation) is implemented in the simulation model depending on the elevator and speed-brake signals from the LPV controller and the Mach number. The closed loop HL-20 simulation model architecture with the output feedback LPV controller is shown in Figure 3.a. The controller structure is shown in Figure 3.b, with 3 measurement signals (FPA, AoA and pitch rate) and 2 control output signals (elevator and

---

\(^6\) The implemented HL-20 system is more advanced and representative than the Mathworks publicly available model which use simplified aerodynamic data\(^10\) and is valid only for the Mach range \([0.08, 0.6]\).
speed-brake). The implementation is in Matlab 2007a. Moreover, LFRSIM block set is used for implementation of the LPV controller.

![Graphs showing Mach number, altitude, and FPA over time](image)

Figure 1 Reference approach trajectory

The available control surfaces are upper left and right flaps (DUL and DUR), lower left and right flaps (DLL and DLR), wing left and right flaps (DEL and DER) and rudder (DR). The control surface mixer calculates the deflection levels for each of these surfaces depending on elevator input, speed brake input and Mach number. Each actuator model consists of a second order transfer function, amplitude and rate limited and a time lag which is a 1st order Pade' approximation. The sensors consist of first order filters with specific corner frequency, bias and Gaussian white noise. See Ref. 28 for additional details on actuators and sensors.

![Diagram of the controller input/output template](image)

Figure 2 Required controller input/output template
Figure 3 HL-20 closed-loop nonlinear simulation model with LPV controller (a) closed-loop structure (b) LPV controller architecture
C. Uncertain parameters and robustness requirements

Table 1 shows the nominal values and the range of uncertain parameters considered in the model. The pitching moment uncertainty is dependent on the Mach and linearly interpolated based on Mach. The robustness criteria are given in terms of the required level of aerodynamic uncertainty that the controller must be capable to absorb. Performance levels HL1 and HL2 correspond to highly desirable and desirable, based on the satisfactory validation w.r.t ±100% and ±50% uncertainty respectively. The performance and safety criteria are given in terms of absolute errors on performance signals (Mach, altitude and flight path angle). These errors are provided as bounds formed from the five uncertainty percentages (-1, -0.5, 0, +0.5, +1) in the robustness criteria and measured with respect to the reference trajectory. The requirement is not to have errors larger than those shown for comparable levels of uncertainty. Figure 3 shows the Mach-based absolute errors allowed on the trajectory guidance parameters tracking for the three absolute uncertainty levels required. More details on the controller design objectives and some guidelines on practical rules for the design of the longitudinal approach controller are available in Ref. 28.

Table 1 HL-20 uncertain parameters: nominal value and uncertainty range

<table>
<thead>
<tr>
<th>Nominal</th>
<th>Δn range</th>
<th>range</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_{CG}$ [m]</td>
<td>0.55</td>
<td>±0.5</td>
</tr>
<tr>
<td>$Y_{CG}$ [m]</td>
<td>0</td>
<td>±0.1</td>
</tr>
<tr>
<td>$Z_{CG}$ [m]</td>
<td>0</td>
<td>±0.5</td>
</tr>
<tr>
<td>$I_{yy}$ [Kg m$^2$]</td>
<td>4.5547e+4</td>
<td>±5</td>
</tr>
<tr>
<td>Mass [Kg]</td>
<td>8660</td>
<td>±5</td>
</tr>
<tr>
<td>$C_D$</td>
<td>Calculated on-line</td>
<td>±10</td>
</tr>
<tr>
<td>$C_L$</td>
<td>Calculated on-line</td>
<td>±10</td>
</tr>
<tr>
<td>$C_M$</td>
<td>Calculated on-line</td>
<td>±7 (at $M = 1.5$)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>±5 (at $M = 5$)</td>
</tr>
</tbody>
</table>

Figure 3 Tolerable Mach based performance error with respect to uncertainty
D. Worst-case analysis criterion as cost function

Linear or nonlinear analysis criterion, which is a function of model outputs, can be formulated as a cost/fitness function for the optimization problems used in the proposed framework. In this study, the analysis criterion used is the deviation from the nominal reference trajectory, i.e., the framework is applied to identify the combination of multiple uncertain parameter values which provides the worst-case (global maximum) deviation for the vehicle from its desired reference trajectory. The analysis criterion for evaluating the worst-case FPA tracking performance over the approach phase in presence of varying operating conditions is thus defined as

\[
\text{Max } J_1 \text{ where } J_1 = \left| \gamma_{\text{ref}} - \gamma_{\Delta} \right| \quad (1)
\]

\[
\text{s.t. } \Delta \leq \Delta \leq \Delta \quad (2)
\]

The cost function \( J_1 \) is defined by (2), where \( \gamma_{\text{ref}} \) represents the reference FPA trajectory and \( \gamma_{\Delta} \) represents the actual FPA trajectory followed by the vehicle in simulation in presence of an uncertain parameter vector \( \Delta \) within the limits. The upper and lower bounds of the uncertain parameters, as defined as \( \% \) variations in Table 1, are given by \( \Delta \) and \( \Delta \). Such clearance criteria are widely used in European aerospace industry for the clearance of control laws\(^{21-23}\).

III. Optimization-based worst-case analysis framework

An optimization-based framework is proposed for worst-case analysis of the LPV controller for the approach phase of the HL-20 vehicle. While developing the framework, reusability of the tools was considered a key requirement. Figure 4 shows the block schematic of the framework. It consists of a closed loop simulation model and an optimization-based worst-case analysis tool (Matlab code is `worst_case.m`). In Figure 4, three different sub-blocks are provided with the tool: (1) Selection of optimization algorithm, (2) Selection of analysis criterion and (3) Definition of the uncertain parameters in the simulation model and their ranges. For the optimization-based worst-case analysis the user needs to provide only this information.

In particular, the use of global nonlinear optimization algorithms is necessary to ensure that the worst-case (considered as the global maximum) has been found. However, as shown in Refs. 23-24, both the reliability and efficiency of global optimization methods can be significantly improved by augmenting them with local gradient-based optimization algorithms in deterministic or probabilistic hybrid switching schemes. The performance of a given optimization algorithm is generally problem dependent, and there is no unique optimization algorithm for general classes of problems which will guarantee computation of the true global solution with reasonable computational complexity. In this tool, therefore, a number of different optimization algorithms are provided, in order that the most promising methods (as evaluated by the designer) for a specific problem of assessing the robust performance of LPV systems may be used.

The main objective of our analysis is to identify the simultaneous perturbation of multiple uncertain parameters which maximises a cost function corresponding to the defined analysis (performance) criterion. The performance criteria could be related to robust stability, robust performance, tracking performance or handling qualities of the space vehicle, either in the time domain or frequency domain. Whichever criterion is chosen, it is posed as an equivalent norm maximization problem. The optimization algorithm makes use of the output information to evaluate a defined cost function. A subsequent candidate uncertain parameter vector is generated by the optimization algorithm. The closed loop model is then simulated with the newly generated uncertain parameter vector. Over iterations, the objective is to maximise the value of cost function. Thus, the optimization algorithm identifies the uncertain parameter vector that maximise a specific cost function. The analysis steps are given in below:
Remark 1: The attraction of the global optimization based methods for nonlinear clearance problem is that it entails no specific requirements on the structure of the problem, in contrast to many other classical and modern linear and nonlinear analysis methods, such as those based on Lyapunov theory or perturbation theory. As and when a convenient structure for the model is available (such as in the case of LTI and simple models of LPV systems), it is certainly advisable to exploit this structure by, for example, converting the problem to a classical semi definite programming problem. However, it must be borne in mind that many realistic system representations will also need to include hard nonlinearities such as look up tables, conditional loops and switches, which significantly complicate the system structure and generally render such methods unsuitable.

E. Optimization based worst-case analysis procedure

Consider the closed loop system as shown in Figure 4. The plant can be either an LPV or a full nonlinear simulation model. The full order LPV controller is given by:

$$\begin{align*}
\begin{bmatrix}
\dot{x}_c \\
u
\end{bmatrix} &= \begin{bmatrix}
A_c(\theta) & B_c(\theta) \\
C_c(\theta) & D_c(\theta)
\end{bmatrix}
\begin{bmatrix}
x_c \\
y
\end{bmatrix}
\end{align*}$$

\tag{4}
The closed loop model can be simulated for a finite time period with a reference trajectory to be tracked, such as that given in Figure 2. The objective is to identify worst-case uncertain parameter combination and therefore the uncertain parameters, \( \delta \in \Delta \), which are to be perturbed while simulating the closed loop system need to be chosen. The minimum and maximum bounds of the uncertain parameters (side constraints for the optimization problem) are required by the optimization algorithm. It is always advisable to normalize the search space and scale it accordingly while assigning the perturbed values to the uncertain parameters. For an asymmetric range of uncertain parameters, the corresponding bias values must be calculated. It is important to have accessibility to the uncertain variables in the model. The chosen uncertain parameters become the optimization parameters.

Every iteration, the optimization algorithm provides a perturbation vector, (see Figure 4) which can be assigned to the respective uncertain parameters in the closed loop simulation model. Each candidate uncertain parameter vector must be assigned with a cost. A function evaluation is essentially the finite time simulation of the closed loop system. In the model, we must have access to the performance output associated with the performance criteria (clearance criteria, robustness criteria, etc.) to be analyzed. In fact, one can consider any relevant state or any performance output signal for analyzing the worst-case performance. Often, these are time response criteria. To identify the worst-case, in essence we maximize the chosen performance output, and thereby eventually try to make the closed loop system become unstable (states diverge to infinity). Assume \( z(t) \) is chosen as the performance output. We can define an \( L_1 \) type cost function as

\[
\max_{\delta \in \Delta} \left| z(t) \right| \tag{5}
\]

where \( \delta \in \Delta \) implies the candidate uncertain parameter vector of a bounded set. An \( L_2 \) type cost function with the same performance output is

\[
\max_{\delta \in \Delta} \| z(t) \|_2 \tag{6}
\]

In Ref. 8, 21, 23 and 24, a specific AoA analysis criterion, which is widely used in Aerospace industry to assess the limits and performance of an aircraft, is formulated as an optimization-based worst-case analysis problem. In Ref. 25, a specific tracking performance criterion was defined for a re-usable launch vehicle control law analysis. In fact, any specific mathematical criteria can be defined as the cost function.

The worst-case analysis tool has a suit of optimization algorithms: local optimization options such as i) Sequential Quadratic Programming with random initial point and ii) Sequential Quadratic Programming with user defined initial point. In the case of single objective analysis, global optimization options such as i) Differential Evolution, a class of evolutionary optimization and ii) Dividing RECTangles, a class of Lipschitzian optimization are available. The analysis framework also offers algorithms for hybrid optimization, which include i) Hybrid Differential Evolution, DE augmented with a local SQP method and ii) Hybrid Dividing RECTangles, DIRECT augmented with local SQP method. For multi-objective analysis, Non-dominated sorting genetic algorithm-II and a multi-algorithm may be selected. The tool includes an option to select a default set of values for the tuning parameters of each algorithm. The optimization progresses until the convergence criterion is met. See Ref. 23 and 25 and references therein for additional details of the individual algorithms.

Different optimization algorithms were developed and implemented in the form of a reusable Matlab toolbox18. A user interface with structure given in Figure 4 (worst_case.m) is available18 to select the cost function for worst-case analysis, to define and access the uncertain parameters.

For completeness, two optimization algorithms that are compared in the present study are explained in the sequel.

**F. Hybrid differential evolution**

The Differential Evolution (DE) method, introduced by Storn and Price32, starts the optimization from a randomly generated set of candidate solutions. In DE, a new search point is generated by adding a weighted vector difference between two randomly selected candidate points in the present population, with yet another third randomly chosen point. The vector difference determines the search direction and a weighting factor decides the step size in that particular search direction. The DE methodology consists of the following four main steps 1) Random initialization, 2) Mutation 3) Crossover 4) Evaluation and Selection. There are different schemes of DE available based on the
various operators that are employed. The one preferred in general and the one which is employed in the present studies is referred as “DE/rand/1/bin”.

(1) Random initialization: Like many other evolutionary algorithms, DE works with a fixed number, \( N_p \), of potential solution vectors, initially generated at random according to:

\[
x_i = x^L + \rho \cdot (x^U - x^L), \quad i = 1, 2, ..., N_p
\]

(7)

where \( x^U \) and \( x^L \) are the upper and lower bounds of the parameters of the solution vector and \( \rho \) is a vector of random numbers in the range \([0, 1]\). Each \( x_i \) consists of elements \( (x_{i1}, x_{i2}, ..., x_{id}) \) which correspond to the uncertain parameters listed in Table 1. The dimension of the optimization problem considered is determined by the number of uncertain parameters involved in the worst-case analysis problem, here 8. The fitness of each of these \( N_p \) solution vectors is evaluated using the defined performance criterion. The analysis tool provides a set of default optimization parameters as well as an option for varying the optimization algorithm parameters, for example in this step the tool will be provided with the option to select \( N_p \).

(2) Mutation: The scaled difference vector \( F_m D_{ij} \) between two random solution vectors \( x_i \) and \( x_j \) is added to another randomly selected solution vector \( x_k \) to generate the new mutated solution vector \( x_{n+1}^{G} \), i.e.,

\[
x_{n+1}^{G} = x_k^G + F_m D_{ij}, \quad D_{ij} = x_i^G - x_j^G
\]

(8)

where \( F_m \) is the mutation scale factor, a real valued number in the range \([0, 1]\), (fixed at 0.8 in this study), and \( G \) represents the iteration number. Figure 5 shows a simple two dimensional example of the mutation operation used in the DE scheme. The difference vector \( D_{ij} \) determines the search direction and \( F_m \) determines the step size in that direction from the point \( x_k^G \).

Figure 5 DE mutation strategy

(3) Crossover: During crossover, each element of the \( n^{th} \) solution vector of the new iteration, \( x_{n+1}^{G} \), is reproduced from the mutant vector \( x_{n+1}^{G} \) and a chosen parent individual \( x_i^G \) as given below,

\[
x_{ji}^{G+1} = \begin{cases} x_{ji}^G & \text{if generated number > } \rho_c \\ x_{ji}^{G+1} & \text{otherwise} \end{cases}
\]

(9)
where \( j = 1,2,...,d \) and \( i = 1,2,...,N_p \). Note \( \vec{x}_n^{G+1} \) has elements \((\bar{x}_{1n}^{G+1}, \bar{x}_{2n}^{G+1}, \ldots, \bar{x}_{dn}^{G+1})\) and \( x_n^G \) has elements \((x_{1n}^G, x_{2n}^G, \ldots, x_{dn}^G)\) and \( \rho_c \in [0,1] \) is the crossover factor, which can be selected using the worst case analysis tool.

(4) Evaluation and selection: The fitness of the new candidate \( x_n^{G+1} \) is evaluated using the defined cost function. An elitist selection method is used. If the new candidate has better fitness than its parent candidate, then the new candidate becomes part of the new iteration. Otherwise, the parent candidate itself is selected.

(5) Termination criterion: An adaptive termination criterion is considered. This criterion is dependent on improvement in the solution accuracy over a finite number of successive generations. The algorithm terminates the search if there is no improvement on the best solution achieved (above a defined accuracy level, chosen as \( 10^{-6} \)) for a defined successive number of generations. This number of generations can be fixed according to the user’s choice. The worst case analysis tool also has a termination criterion depending on the computational budget provided by the user. The tool has the option to select the suitable termination criterion and related parameters.

(6) Hybridization of optimization algorithms: The hybrid DE scheme employed in the worst-case analysis tool applies gradient-based local optimization, again using “fmincon”, to a solution vector randomly selected from a current generation set. When the local scheme is chosen, the optimization starts from the given initial condition and continues until it either converges or reaches a defined maximum number of cost function evaluations. The algorithm is simple, and tries to search for the global optimum in a “greedy” way, demanding improvement in the achieved optimum value in every iteration. A pseudo-code for the hybrid DE algorithm is given in Table 2. The worst-case tool allows the user to select the options for the hybrid differential evolution algorithm.

Table 2 Hybrid Differential Evolution\(^{25}\)

1) Initialize random candidate solutions in search space
2) Evaluate the fitness of each solution and choose the best fitness
3) Apply DE for few initial iterations (say 20); update the best fitness value in each iteration
4) While the termination criteria not satisfied, calculate the improvement in best fitness
   a. If improvement in best fitness, then continue DE
   b. Else
      i. Choose a random solution from the current set, say \( x_0 \) and apply the local optimization scheme “fmincon” with \( x_0 \) as initial point.
      ii. If improvement in best fitness, then replace \( x_0 \) in the set with the new solution
      iii. Else keep \( x_0 \) in the set
5) End of While

G. Dividing RECTangle (DIRECT)

The DIRECT optimization/search method, originally proposed in\(^{33}\), does not require any derivative information to be supplied, and uses a center point sampling strategy. Without loss of generality, in the DIRECT algorithm, it is assumed that every variable has a lower bound of zero and an upper bound of one, since in our formulation normalization of the variables to this interval can always be done. Thus, the search space is an \( n \)-dimensional hypercube or box. It can be defined as \( D = \{ x \in \mathbb{R}^n : 0 \leq x_i \leq 1 \} \). The algorithm works in the normalized parametric space, transforming to the actual search space as and when the cost function has to be evaluated. The main idea of the algorithm is as follows: As the algorithm proceeds, the search space will be partitioned into smaller hypercubes or boxes and each will be sampled at the center point of the interval. Over iterations, the algorithm tries to find all the potentially optimal hypercubes or boxes in the search space and then partition them, thereby obtaining the global solution.
(1) Center point sampling and dividing strategy: The algorithm begins with the evaluation of the cost function about the center point, say \( c \), of the normalized search space. The subsequent step is to divide the hyper box. We sample the points \( c \pm \delta e_i \), where \( \delta \) equals one-third of the side length of the cube \( (\delta = \frac{1}{3} \varepsilon) \) and \( e_i \) is the \( i \)th Euclidean base vector. \( w_i \) is defined as the \( \min \{ f(c - \delta e_i), f(c + \delta e_i) \}, 1 \leq i \leq N \), and the division is done in the order given by \( w_i \), starting with the lowest \( w_i \). Therefore the hyper box is first partitioned along the direction of lowest \( w_i \) and then the remaining field is divided along the direction of the second lowest \( w_i \) and so on until the hyper box is partitioned in all directions. From this point onwards, the algorithm starts identifying the potentially optimal hyper boxes, dividing these hyper boxes further and sampling at their center points until the termination criteria is satisfied.

(2) Potentially optimal hyperboxes: Suppose the unit hyper box is divided into \( m \) smaller hyper boxes. Let \( c_i \) denote the center point of the \( i \)th hyperbox and \( \varepsilon_i \) the distance from the center point to the vertices. One box among these \( m \) hyper boxes must be selected for further sampling. The definition of the potentially optimal hyper box is as follows: Let \( \xi \) be a positive constant and \( f_{\min} \) be the current lowest function value. A hyperbox \( j \) is said to be potentially optimal if there exists some rate of change constant \( K > 0 \) such that

\[
\frac{f(c_i) - K \varepsilon_{ij}}{f(c_i) - K \varepsilon_i} \leq f_{\min}, \quad \text{for any } i=1, \ldots, m
\]

\[
|f(c_j) - K \varepsilon_j| \leq f_{\min} - \xi |f_{\min}|
\]

Figure 6 illustrates the above definition further. The horizontal co-ordinate is the size of the hyperbox, which is the distance from the center to the vertices of the box. This captures the goodness based on the amount of unexplored region in the search space. The vertical coordinate is the value of the cost function at the center point of the particular hyperbox. This captures the goodness of the interval with respect to the local search that is the goodness based on the known function values. Each point on the graph represents hyperboxes.

The first condition in the definition forces the hyper box to be on the lower right of the convex hull of the dots. Hyperboxes having low objective function values are inclined to fall on the convex hull of the set, as are (relatively) large hyperboxes. One of the largest hyperboxes is chosen for division. The second condition insists that the lower bound for the interval, based on the rate of change constant \( K \), exceed the current best solution by a nontrivial amount. This condition prevents the algorithm from becoming too local in its orientation, in terms of Figure 6, it implies that some of the smaller intervals might not be selected. In this way, the groups of hyperboxes are larger.
and consequently the iteration places a stronger emphasis on the value of the objective function at the center point of the hyperbox, which biases the search locally. The parameter $\xi$ was introduced to balance the local and global search. In Figure 6, the point $(0, f_{\min} - \xi | f_{\min})$ changes the convex hull so that the hyperbox with smallest objective function value need not be potentially optimal. By this approach, more sampling is done in larger, unexplored hyperboxes. If $\xi = 0.01$, then the lower bound for the hyper box would have to exceed the current best solution by more than 1%. Previous studies indicate that a choice of value for $\xi$ ranging from $10^{-3}$ to $10^{-7}$ generally provides the best results, and a value of $\xi$ equal to $10^{-4}$ was used in this study.

Termination criterion: The worst-case analysis tool has the option for selecting the termination criterion between the fixed computational budget and the adaptive termination depending on the improvement in objective cost.

Hybrid- DIRECT: There is the option to select the hybridization of the DIRECT algorithm with a local optimization scheme. Once the option is selected, the DIRECT algorithm is modified using a simple hybridisation strategy. A local optimization method based on Sequential Quadratic Programming (SQP) is incorporated into the DIRECT algorithm using the MATLAB function “fmincon”. The solution obtained from the DIRECT algorithm is considered as the initial solution for the local optimization method. The DIRECT algorithm including the local optimization is referred to as H-DIRECT. The hybridisation attempts to overcome one of the main disadvantages of the DIRECT algorithm, namely its lack of fast convergence to solutions that are on the bounds of the uncertain parameter space due to the center point sampling strategy. The pseudo code for the DIRECT algorithm is given in Table 3.

Table 3 DIRECT and H-DIRECT algorithm

| 1) | Normalise the search space to unit hyperbox. |
| 2) | Sample the center point $c_1$ of the hyperbox; Evaluate $f(c_1)$. Set $f_{\min} = f(c_1)$, $m = 1$, $t = 0$ (iteration counter), and TC = 0 (Termination counter) |
| 3) | While Termination criterion not satisfied; say $(TC \leq 300)$ do |
| | a. Identify the set 'S' of potentially optimal hyperboxes |
| | b. Select any rectangle/box $j \in S$ |
| | c. Divide the box $j$ as follows: |
| | i. Identify the set $I$ of dimensions with the maximum side length $\varepsilon$. Let $\delta$ equal one third of this maximum side length ($\delta = \frac{1}{3}\varepsilon$) |
| | ii. Sample the function at the points $c \pm \delta e_i$, $\forall i \in I$, where $c$ is the center of the box and $e_i$ is the $i^{th}$ unit vector. |
| | iii. Divide the box $j$ containing $c$ into thirds along the dimension $I$, starting with the dimension with the lowest value of $w_i = \min\{ f(c \pm \delta e_i) \}$, and continuing to the dimension with the largest $w_i$. Update $f_{\min}$, $x_{\min}$, $m$. |
| | d. Set $S = S - j$. If $S \neq \{\}$ Go to step (b) |
| | e. Set $t = t + 1$. Calculate the improvement in $f_{\min}$ obtained from the previous iteration. |
| | TC = TC + 1 if the improvement $\leq 1e^{-6}$ in subsequent iterations, if not TC = 0. |
| 4) | End of While End of DIRECT algorithm. |

Begin Hybridisation

5) Choose the solution $x_{\min}$ from STEP4, set $x_{\text{initial}} = x_{\min}$ and execute “fmincon” algorithm to refine the global solution. (This is particularly effective when some or all $x_{\min}$ are on their bounds.)

End of H-DIRECT algorithm.
IV. Worst-case analysis results and discussion

The objective of the optimization problem is to identify the combination of uncertain parameters giving the maximum value of the defined analysis criterion, i.e. the worst-case value. The analysis criterion is defined in Eq. (1). The optimization scheme provides the values of the uncertain parameters used when simulating the re-entry model. The uncertainties given in Table 1 are considered. In the model, a multiplicative uncertainty model is used based on the percentage uncertainty range $\Delta \%,$ nominal values and a normalized random number.

The nonlinear closed loop model of the re-entry vehicle is simulated about the predefined approach flight condition as in Figure 1, but with the new set of uncertain parameter combinations provided by the optimization algorithm. A cost must be assigned to each candidate uncertain parameter vector. At the end of each simulation, the optimization algorithm evaluates the cost/fitness function in Eq.(1) for the specific combination of uncertain parameters, based on the simulation output. The optimization is continued until the defined termination criterion is satisfied. The framework is implemented in MATLAB Version 7.4.0.287 (R2007a) and Simulink Version 6.6 (R2007a). The optimization algorithms make use of the cost function value to generate a search direction in the parameter space in an ‘intelligent’ manner. Certain algorithms make use of the gradient information and certain evolutionary algorithms try to mimic the evolution inspired from nature. Ultimate aim of all these algorithms is to obtain the worst-case uncertain parameter vector over sufficient number of iterations.

The LPV controller is designed to provide good tracking in flight path angle and hence it was decided to check the worst-case performance w.r.t. flight path angle. For this purpose, the absolute error between the actual flight path angle and the reference flight path angle, $J_1 = |\gamma - \gamma_{ref}|$ in Eq. (2) is considered and maximized. Note that the optimization algorithms try to find the global minimum and hence the maximisation problem is posed as a minimisation problem. The cost function for the worst-case analysis must be provided by the user as a MATLAB function.

There are different optimization algorithms available to perform the worst-case analysis and hence an interface function is provided to select the optimization algorithm and the parameters related to each algorithm. The MATLAB interface function allows the user to choose the optimization function and is available in an M-file, ‘worst_case.m’. The Hybrid Differential Evolution algorithm was chosen. The Matlab screen shot given below, demonstrates how we choose this specific optimization algorithm and parameters associated with this algorithm. Even though the default option exists, a case of user-defined selection of the parameters is demonstrated.

---------------------------------------------------------------MATLAB SCRIPT---------------------------------------------------------------

>> worst_case

Worst case analysis set up:

Analysis [ 0:=single / 1:= multi / 2:monte ] ? 0

Hybrid Differential Evolution Selected

Option [ 0:=Default / 1:= user-defined ] ? 1
Termination [Fixed Termination / Adaptive Termination] ? 'Adaptive Termination'
Solution accuracy [ 1e-3 is good ] ? 1e-2
Adaptive iteration [ 300 ] ? 100
Population size [ Integer ] ? 40
DE stepsize [ value between 0-1 ] ? 0.8
DE crossover [ value between 0-1 ] ? 0.8
Local iterations [50 default]? 50

Optimization Proceeding

--- END OF SCRIPT ---
The above MATLAB script describes the values selected for various optimization parameters. Note that an adaptive termination is selected, implying that the algorithm terminates the search if there is no improvement on the best solution achieved (above a defined accuracy level, chosen as 10−2) for a defined successive number of generations chosen as 100 in this case.

Figure 7 shows a set of worst-case analysis results. The green line in Figure 7 corresponds to the nominal case. The red lines in Figure 7 are responses due to the various uncertainty combinations, which were supplied to the closed loop simulation by the optimization algorithm during all the iterations. Figure 7 provides the worst-case envelope for the considered level of uncertainties for the chosen performance objective. No uncertain parameter combination within the considered range destabilizes the nonlinear closed loop simulation model with the LPV control law \( 27 \). The maximum \( J_I = |\gamma_{ref} - \gamma_\Delta| \) obtained was 0.887 degrees, which occurs when the Mach number is around 3.5 (See Figure 7).

Figure 8 shows the result from the H-DIRECT optimization method with the default settings for the optimization parameters. The screenshot below shows the method for selection of the H=DIRECT optimization. The maximum value of \( J_I = |\gamma_{ref} - \gamma_\Delta| \) obtained is 0.887 degrees, exactly the same level of worst-case performance found by the HDE algorithm.

```
>> worst_case

Worst case analysis set up:

Analysis [ 0:=single / 1:= multi / 2:=monte ] ? 0

Option [ 0:=Default / 1:= user-defined ] ? 0

Optimization Proceeding
```

The normalized worst-case uncertain parameter combination vector found by both the HDE and H=DIRECT algorithms was \([+1, -1, 1, -1, 1, 0, 0, 0]\). A comparison of nominal vs. worst-case tracking error is given in Figure 9. The nominal tracking error is in green and the worst-case tracking error is in red. The optimization based analysis in both case were completed within 2 hours of simulation time. In general the computational times are much better than that of the Monte Carlo simulations currently used in industry. Moreover, the ability of optimization algorithms in determining the global solution (or a solution very near the true global solution) provides certain guarantees of identifying the worst-case and makes the approach attractive.
Figure 7 Mach dependent tracking performance error with respect to all uncertain parameter combinations supplied by the HDE optimization algorithm (in red) including the worst-case; reference trajectory (in green)

\[ \gamma_{\text{err}} = |\gamma_{\text{ref}} - \gamma_{\Delta}| \]

Figure 8 Mach dependent tracking performance error with respect to all uncertain parameter combinations supplied by the HDIRECT optimization algorithm (in red) including the worst-case
A novel optimization-based framework is presented in this paper for analyzing the robustness of the LPV controllers with a nonlinear simulation model of NASA HL-20 vehicle. The analysis framework incorporates the evolutionary as well as deterministic optimization algorithms and their hybrid local/global optimization algorithms. The worst-case deviations from a predefined re-entry profile due to simultaneous variations of multiple uncertain parameters are determined by two optimization methods - hybrid differential evolution and hybrid dividing rectangles.

The main lessons learnt so far from this application are:

- The optimization-based worst-case analysis tool is simple, highly flexible and easy to use.
- It allows direct comparison of closed loop time-domain performance on the LPV and full nonlinear models.
- Computation times compare favourably with Monte Carlo simulations. It took approximately 2hrs and 45 minutes for a full worst-case analysis.
- The results project the proposed framework as a useful, potential tool for complex controller validations in future space applications.

In the future, the development of tool will focus on:

- The performance and the computational feasibility of the tools need to be validated on a detailed and complex industrial benchmark.
- The usability and advantages, if any, of the tools in frequency-domain performance criterion need to be investigated.
- Efforts will be made to refine the tools to integrate with current validation processes.

**Acknowledgments**

This work is part of the ESA funded "Robust LPV Gain-Scheduling Techniques for Space Applications" LPVMAD project. The authors would like to acknowledge DEIMOS Space S. L. (Spain) and in particular Dr. Andres Marcos for his valuable comments and support. We also would like to express our acknowledgements to the other team members involved in this project.
References

14. Scherer
15. Parrilo, P., Structure semidefinite programs and semi algebraic geometry methods in robustness and optimization, in California Institute of Technology. 2000, California Institute of Technology: Pasadena, California