7 Optimisation-Based Clearance

Andras Varga

German Aerospace Center, DLR - Oberpfaffenhofen
Institute of Robotics and Mechatronics D-82234 Wessling, Germany.
Andras.Varga@dlr.de

Summary. The basic feature of the optimisation-based clearance approach is to reformulate the clearance problems as equivalent minimum distance problems for which "anti"-optimisation is performed to determine the worst-case parameter combination/flight condition leading to worst performance. The basic requirements for the applicability of the optimisation-based approach are the availability of suitable parametric models describing the overall nonlinear dynamics of the augmented aircraft and of accompanying efficient and reliable trimming, linearisation and optimisation software tools. The optimisation-based approach has no limitations with respect to clearance criteria, being able to address all kind of clearance requirements which are expressible as mathematical criteria.

7.1 Classical Versus Optimisation-Based Approach

Let $c(p, FC)$ be a given clearance criterion, depending on the uncertain parameters grouped in a q-dimensional vector p and flight condition vector FC usually having up to three components (e.g., Mach-number M, altitude h, angle of attack α). p is generally unknown, but it is assumed that all its components lie in known intervals, defining a hyper-box $\mathcal{P}$ in the q-dimensional Euclidean space. The variation of flight condition FC is determined by the defined flight envelope where the aircraft is required to operate.

We can easily formulate the clearance problem for a given performance criterion $c(p, FC)$ as a distance minimisation problem. Let c_0 be the limiting acceptable value of $c(p, FC)$, as defined in the clearance documents. Then, the difference

$$d(p, FC) = c(p, FC) - c_0 \tag{7.1}$$

can be interpreted as a *signed* distance function to the limiting acceptable performance c_0. If for a fixed FC, $d(p, FC)$ is positive for all parameter values $p \in \mathcal{P}$, then the clearance requirement is fulfilled in FC and the point FC is *cleared*. The minimum distance

$$\underline{d}(FC) = \min_{p \in \mathcal{P}} d(p, FC)$$

can be interpreted as the robustness measure of how far the system is from the limiting acceptable performance c_0. A negative value of $\underline{d}(FC)$ can be

interpreted as a measure of the lack of robustness and the corresponding point FC is *not cleared*.

The current industrial clearance approach relies on an exhaustive search on a grid for both flight conditions and uncertain parameters. Typically, one chooses N flight conditions FC_i, $i = 1, \ldots, N$ and in each flight condition FC_i, $c(p, FC_i)$ is evaluated only in $\nu(\mathcal{P})$, the set of vertex points of the q-dimensional hyper-box $\mathcal{P}$. Thus,

$$\widetilde{d}(FC_i) := \min_{p \in \nu(\mathcal{P})} d(p, FC) \geq \underline{d}(FC_i)$$

and $\widetilde{d}(FC_i)$ is only an approximation (upper bound) of the true minimum distance $\underline{d}(FC_i)$. The value of $\widetilde{d}(FC_i)$ is used to decide if FC_i is cleared or not. Clearing N flight conditions, each in 2^q vertex points, requires $N \cdot 2^q$ evaluations of $c(p, FC)$. Thus, the required computation time increases exponentially with the dimension q. To have a feeling what exponential computational complexity means, assume that 1 second is necessary for one function evaluation. Then for $q = 5$, 9 and 15, the time needed to check *only one* flight condition in the 2^q vertices is 32 seconds, 512 seconds, and 9.1 hours, respectively. Note that typical values of N are of order 5000[1].

Two main difficulties of the classical approach are evident. First, there are tremendous costs involved when simultaneously checking the robustness for many uncertain parameters. Since the evaluation of each robustness measure increases exponentially with the number of parameters q, obviously the computational costs for large problems are too high to be affordable in industrial practice. Second, there is no guarantee that the "cleared" flight conditions should have been cleared, since for each parameter only the extreme points (maximum and minimum) are checked. Thus, if the minimum occurs in an intermediate point which is not cleared, then the clearance results could be false. The same applies when considering the finite set of flight conditions $\{FC_1, \ldots, FC_N\}$ which certainly can not cover all points of the physical flight envelope.

The optimisation-based approach offers immediate improvements for both of these aspects. The first improvement is a reasonable computational cost of clearance in case of many parameters. This occurs because the number of function evaluations necessary to compute the worst-case parameter combination is usually much lower than that corresponding to evaluating the function in all vertex points, even in the case when only two values for each parameter are used. The second improvement is achieved by allowing a continuous variation of parameters within the given parameter space $\mathcal{P}$. In this way, the clearance results cover all points of $\mathcal{P}$ and therefore are more reliable.

A straightforward way of enhancing the classical approach is to perform, in each of the selected N points FC_i, $i = 1, \ldots, N$ of the flight envelope, an optimisation-driven worst-case search to determine the minimum distance

[1] U. Korte, Private communication

$\underline{d}(p, FC_i)$. This approach can be seen as a combination of the gridding-based classical search in a discrete set of flight conditions with the optimisation-based continuous search for worst-case parameter combinations in the complete parameter space. Note that this approach can be very effective, even when discontinuities of derivatives are present in the mathematical model due to the use of linear interpolation formulas to evaluate aerodynamic coefficients defined by look-up tables. By "freezing" the flight condition during optimisation, the variations within these tables no longer play any role, and therefore, optimisation methods based on gradient search techniques can be readily employed to locate worst-case parameter combinations in a very effective way. The mathematical optimisation problems to be solved belong to the class of nonlinear programming problems (NLPs) with simple bound constraints on variables for which both gradient-based and gradient-free techniques can be employed (see Section 7.3).

An enhancement of this approach can be achieved by explicitly addressing the continuous variation for the flight condition. A possible approach is to define a coarse set of flight conditions $\mathcal{FC} = \{FC_1, \ldots, FC_K\}$, where $K \ll N$, and associate to each flight condition FC_i a box $FC_i \pm \Delta FC_i$ centered around FC_i. The sizes of these boxes are chosen such that their union covers the whole physical flight envelope. Then, by including FC among the optimisation variables, solve for all FC_i, $i = 1, \ldots, K$ the distance minimisation problems

$$\underline{d}(FC_i) = \min_{\substack{p \in \mathcal{P} \\ FC \in FC_i \pm \Delta FC_i}} d(p, FC)$$

If $\underline{d}(FC_i) > 0$, then each flight condition in the hyperbox around FC_i is cleared. Otherwise, a locally finer grid can be considered if necessary and the clearance can be repeated on this finer grid. The main advantage of this approach is the complete and continuous coverage of both the flight envelope and the parameter space, and thus a higher confidence in the clearance results. Another advantage is the potentially lower total costs, by using a reduced set of only $K \ll N$ flight conditions. The mathematical optimisation problems to be solved is a NLP with only simple bounds on variables and linear constraints (see Section 7.3).

It is important to note that robustness analysis problems are essentially global optimisation problems. When qualifying flight conditions as *cleared* or *not cleared* on basis of a local search, only the *not cleared* points are guaranteed. For a rigourous analysis, only the computationally very expensive global search approaches with guaranteed convergence are able to assess *cleared* points. Often, a restricted preliminary sensitivity analysis with only a few parameters can indicate the probable lack of multiple local minima. In such cases, the cheaper and more efficient local search methods can be used for solving clearance problems practically without any loss of reliability of the results.

7.2 Description of the Analysis Cycle

Let $c(p, FC)$ be a given clearance criterion depending on the uncertain parameters grouped in a parameter vector p and the flight condition vector FC. The analysis cycle used for the clearance of a control configuration for the given clearance criterion $c(p, FC)$ is illustrated by the flow diagram in Figure 7.1. Here we assume that a continuous search is performed only in the parameter space $\mathcal{P}$ for a finite set of flight conditions FC_i, $i = 1, \ldots, N$.

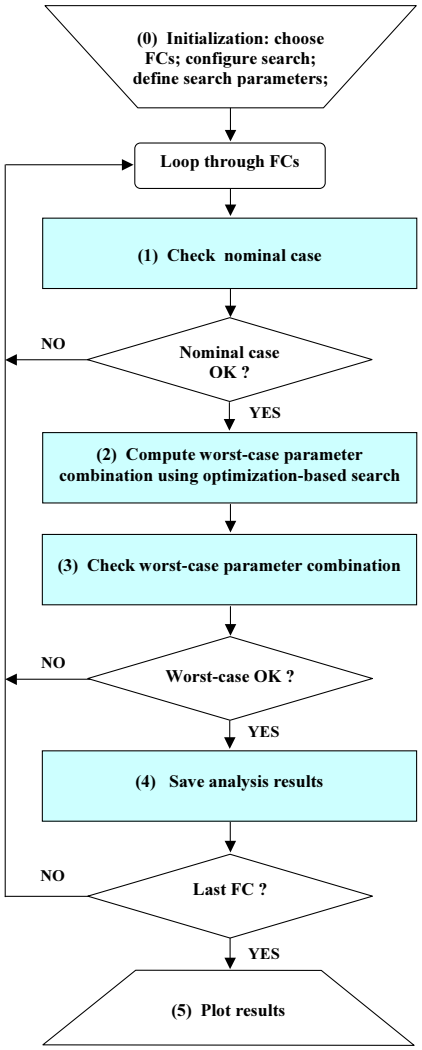

Fig. 7.1. Optimisation-based clearance analysis cycle

According to the flow diagram in Fig. 7.1, the following main steps have to be performed in an optimization-based clearance procedure:

Step 0 is the initialisation step for the optimisation-based clearance procedure and usually involves choosing the flight conditions FC_i, $i = 1, \ldots, N$ where the worst-case parameter combinations are to be determined, the definition of vector p (e.g., most relevant or full set, longitudinal or lateral), setting of appropriate options for trimming, setting the values of criterion specific variables (e.g., frequency-grid, time-grid), or choice of optimisation method (see Section 7.3) and corresponding options (e.g., stopping tolerances, maximum number of iterations etc.)

Step 1 is necessary to eliminate from the analysis those points where the clearance requirements are not fulfilled for the nominal values of parameters. Furthermore, here we can also check if the normal acceleration n_z is within an allowed range of values (e.g., $-3\,\mathrm{g} \leq \mathrm{n_z} \leq 7\,\mathrm{g}$ for HIRM+) or the control surface deflection saturation limits for δ_{TS}, δ_{TD}, and δ_R are reached. Points where such violations occur are not cleared and are automatically eliminated from the analysis. The neat effect of this check is a reduction of the overall computational effort.

Step 2 is the basic optimisation step performed for each selected flight condition FC_i. The results of this step are the worst-case parameter combination p_i^{worst} and the corresponding criterion value $c(p_i^{worst}, FC_i)$. The performed number of function evaluations is an indication of the efficiency of the optimisation-based search in comparison with the classical grid-based approach.

Step 3 is similar to Step 1 and the performed check is necessary because the worst-case parameter combination can lead to the same possible violations of some conditions as those occurring in the nominal case (e.g., violation of condition $-3\,\mathrm{g} \leq \mathrm{n_z} \leq 7\,\mathrm{g}$ or of the deflection saturation limits). Note however, that such points are found only incidentally by the optimiser, and may exist in a particular flight condition FC_i even in the case when the determined worst-case parameter combination does not violate the above conditions.

Step 4 is the outputting of computed data to a database. For each flight condition FC_i, the stored information contains typically the computed worst-case parameter combination p_i^{worst}, the corresponding minimum distance $d(p_i^{worst}, FC_i)$, the number of performed function evaluations, cleared/not cleared status information, etc.

Step 5 performs the graphical evaluation of obtained results by producing plots necessary for assessing and documenting the clearance results.

7.3 Optimisation Algorithms Suitable for Clearance

Each clearance analysis problem defined in Chapter 10 can be formulated as a standard *nonlinear programming problem* (NLP) of the form

$$\begin{aligned} &\min\ f(x) \\ &\text{subject to } c_j(x) \geq 0,\ j = 1, \ldots, m \\ &\qquad\qquad l_i \leq x_i \leq u_i,\ i = 1, \ldots, n \end{aligned} \tag{7.2}$$

to be solved for $x \in \mathbb{R}^n$. Here the components of x includes, in general, variables defining the flight condition (e.g., M, h, and/or α) and components of the vector p representing the uncertain parameters of the model. Each component x_i of x must lie between the corresponding lower bound l_i and upper bound u_i. The lower and upper bounds are defined by restricting the flight conditions to lie within the admissible region defined by the flight envelope, while the bounds on uncertain parameters are defined on basis of their physical significance. The scalar constraints $c_j(x)$ may correspond, for example, to restricting the search to a typical polygonal region, whose boundary is defined by several line segments. Thus, in the most general case, the NLP (7.2) corresponding to a particular clearance problem is still a particular NLP subject only to simple bounds on variables and linear constraints. If the flight condition (i.e., M, h and α) is not part of x, then the clearance problem can be formulated as an even simpler NLP with only simple bounds on variables.

The NLPs arising in clearance problems have several particular features:

Low order. Since the optimisation variables are the uncertain parameters and possibly some components of the flight condition vector, the dimension of the optimisation problem is relatively small, satisfying $n \leq 25$.

Multiple local minima. The functions expressing clearance criteria exhibit very complex dependencies of parameters. It follows, that we can always expect that these functions have several local minima.

Expensive function evaluation. The evaluation of criteria based on linearised models, involves trimming, linearisation and frequency response or eigenvalue computation of relatively high order systems (up to 60 state vector components). The evaluation of criteria based on nonlinear models usually involves simulations, preceded by trimming. Thus typically, the evaluation of clearance criteria is very time consuming. Fast and reliable trimming (e.g. via inverse models) is a prerequisite to increase the efficiency of function evaluations. Model reduction techniques can be efficiently used to reduce the order of linear models used to evaluate frequency-response based criteria.

Discontinuous derivatives. Discontinuities in derivatives of functions arise from several sources. Naive implementation of criteria by defining distance functions to regions with polygonal boundaries will certainly lead to functions with discontinuous derivatives. By approximating boundaries

by polynomials (e.g., spline functions) we can get rid of such discontinuities, but the clearance problem can be falsified by such an approximation. Other sources of discontinuities can lie in the model itself, if table-driven linear interpolations are present. Finally, failures to accurately evaluate the function (e.g., because of inaccurate trimming) lead to discontinuities even in the functions themselves. Handling trim failures, for example by setting the function values to a very large number, can rise severe problems for certain solvers.

Noisy function. Noise in function values originates from various truncation errors made in intermediary computations such as trimming, linearisation, order reduction, numerical evaluation of gradients, simulation, as well as from the round-off errors associated with difficult numerical computations like eigenvalue computation. To handle such functions, the usage of more robust, derivative-free optimisation methods could be necessary (e.g., pattern search) or enhancements of gradient-search techniques are necessary (e.g., usage of central difference approximation of gradients, usage of gradually increased accuracy in gradient computations, etc.). For additional aspects of optimisation with noisy function see [1].

In the following paragraphs we present brief information on several optimisation algorithms which are suitable for solving the NLPs appearing in the clearance problems. For most algorithms software implementations are freely available on the Internet [2].

7.3.1 Gradient-based local search methods

Gradient-based minimisation methods use local information on the function through its gradient to achieve fast convergence rates. This is why, when applicable, many gradient-based search methods allow highest computational performance in solving general or particular NLPs. For the usage of most gradient-based techniques a basic requirement is the continuity of gradient with respect to the optimisation variables. Furthermore, for a satisfactory performance, the availability of an analytic expression of gradient is highly desirable. However, for complex functions like those typically arising in clearance problems, usually no analytic gradients are available. Therefore, numerical approximations of gradients have to be computed resultsing in a slower and less reliable execution, especially when function evaluations are noisy.

The **sequential quadratic programming** (SQP) method to solve the general NLP with equality and inequality constraints can be used to solve the particular NLP of the form (7.2) which arises in clearance problems. The SQP method can be seen as a generalisation of Newton's method for unconstrained optimisation in that it finds a step away from the current point by minimising a quadratic approximation of the problem function $f(x)$. Under mild conditions this method has a fast, so-called *superlinear* convergence [3]. An alternative approach for problems with only simple bounds constraints on the vari-

ables is the **limited memory BFGS with bound constraints** (L-BFGS-B) described in [4] together with accompanying Fortran 77 software. This approach extends the standard Broyden-Fletcher-Goldfarb-Shanno (BFGS) quasi-Newton method to handle NLPs with simple bounds, by using a gradient projection approach. The algorithm has a superlinear convergence and no violations of bound constraints on variables occur during optimisation. For a more detailed discussion of both approaches see [3].

7.3.2 Gradient-free local search methods

Derivative-free methods using only function evaluations are a real alternative to gradient-based methods, especially when function evaluations are noisy and/or discontinuities in the gradient are present. Two classes of derivative-free methods are known: *direct-search* methods which include the popular simplex and pattern search methods, and *trust-region* methods relying on linear or quadratic interpolation models. Derivative-free methods are useful when the function $f(x)$ is not smooth (e.g., "noisy" function) or when accurate derivatives are difficult to determine numerically. For more details on derivative-free methods see [5] and for performance comparisons see [6].

Pattern search (PS) algorithms are a class of *direct search* methods initially proposed for unconstrained minimisation which has a rigourous global convergence theory. The PS techniques has been recently extended to solve NLPs with simple bounds [7]. PS methods use a *simple* decrease criterion to accept a step as opposed to the *sufficient* decrease criterion used by gradient-based search. This is why, PS methods usually have a slower convergence rate than a gradient-based search. On the other hand, PS methods are often numerically more robust than gradient-based methods in avoiding local minima as well as tackling with noisy functions. PS methods may require a relatively large number of function evaluations, hence they tend to be effective primarily for problems of relatively small dimensions and low accuracy situations.

Model-based trust region methods exploit the smoothness of the objective function and attempt to preserve the convergence properties of their gradient-based counterparts. The **constrained optimisation by linear approximations** (COBYLA) approach employs linear approximations to the objective and constraint functions [8]. The approximations are formed by linear interpolation at $n + 1$ points in the space of the variables (regarded as vertices of a simplex) and the size of the simplex is reduced as the optimisation advances. The main advantage of COBYLA over many of its competitors, is that it treats each constraint individually when calculating a change to the variables, instead of lumping the constraints together into a single penalty function. Therefore, COBYLA usually has better convergence than the pattern search method. One disadvantage of the COBYLA software, is that it does not address simple bounds explicitly and these must be transformed to $2n$ general constraints in the NLP (7.2) of the form $c_j = x_j - u_j$,

$c_{j+n} = l_j - x_j$, for $i = 1, \ldots, n$. Unfortunately, this leads to frequent violations of bound constraints during the computation.

The **derivative-free optimisation** (DFO) trust-region method uses a quadratic approximation of the objective function (see [6] and references therein). The quadratic model approximates the function well within a certain "trust"-region of a given radius and serves to determine new points by minimising the current approximation instead of the function itself. The new points generated by the algorithm are used both to advance the optimisation and to update the approximation. Since the DFO algorithm needs only a relatively few function evaluations, this method is well-suited to minimise expensive functions which depend on few (some hundred at most) variables.

7.3.3 Global search methods

For functions with many minima, the use of global optimisation techniques is the only alternative for successful computations. In this section we discuss three global optimisation approaches which can be employed for solving optimisation-based clearance problems with simple bounds on the parameters. Typically, these methods require a very large number of function evaluations and therefore they are primarily intended either to determine good starting points for local search based methods, or to address difficult clearance problems with many local minima.

The **simulated annealing** (SA) algorithm is essentially an iterative random search procedure with adaptive moves along the coordinate directions [9]. It permits uphill moves under the control of a probabilistic criterion, thus tending to avoid the first local minima encountered. It has been proved that the sequence of points sampled by the SA algorithm form a Boltzmann distribution and converges to a global minimum with a probability of one as the annealing "temperature" goes to zero.

The **genetic algorithm** (GA) is a global optimisation approach based on evolution strategies which guarantee the survival of the fittest individual in each population [10]. The GA can easily handle problems with simple bounds on the variables, and even general constraints by using penalty function techniques. There are several selection schemes which can be combined with a shuffling technique for choosing random pairs for mating. The GAs based on binary coding, use mutations (e.g., jump or creep mutations), crossover (single-point, uniform, etc.), niching and various other strategies to produce successive populations. The use of GA for function optimisation is quite costly in terms of the required number of function evaluations, but usually its cost can be predicted in advance by choosing the population size and the number of successive generations. To find the global extremum with high accuracy, this method typically requires a very large number of function evaluations.

The global optimisation using **multilevel coordinate search** (MCS) attempts to find the global minimiser of the bound constrained optimisation problem using function values only, based on a multilevel coordinate search

that balances global and local search [11]. The local search is done via SQP. The search is not exhaustive, so occasionally the global minimum may be missed. However, a comparison to other global optimisation algorithms shows excellent performance of the MCS method in many cases, especially in low dimensions.

7.4 Conclusions

The main benefits of the optimisation-based search are the lower costs in the case of many simultaneous parameters and an increased reliability of the results because of the continuous exploration of parameter space. Prerequisites for the applicability of this approach are appropriate parameterised models, fast and reliable trimming and linearisation procedures (necessary for efficient function evaluation) and robust optimisation software capable of addressing the challenge of solving NLP problems with possibly non-smooth, expensive to evaluate and noisy functions. Taking into account all these aspects, the best suited approach appears to be the trust-region DFO method. For functions with only a few variables, DFO typically requires relatively few evaluations of the problem function. For more difficult problems with many local minima, the MCS method combining local and global search appears to be a viable alternative to more expensive GA and SA methods.

The acceptance of the optimisation-based clearance approach by the industry depends on several aspects. Since the optimisation-based clearance can be seen just as a straightforward (more powerful) extension of the classical approach, the **effort to learn** this method is almost negligible. In fact, the classical gridding-based approach can always be used as a standard option even in an optimisation-based clearance methodology. This is why, the **first time setting up** of the method is not much different than for the grid based approach. However, the usage of sophisticated optimisation tools requires special care when defining suitable smooth distance functions on basis of standard clearance criteria. Further, the implementation of fast and reliable procedures to evaluate these functions is of crucial importance for the success of the optimisation-based worst-case search.

The **reusability** of software to cope with new aircraft models and control laws can be enforced by performing the optimisation-based clearance within a dedicated software environment which supports the interchange of different models and criteria. Within such an environment, the effort for a **new analysis setup** with different models and control laws is expected to be easily affordable. For maximum flexibility, such an environment has to provide additional facilities for experimenting with various optimisation techniques, different parameter sets, different criteria, different optimisation options etc. A clearance software environment satisfying all above requirements will be implemented as an add-on to the optimization based design environment MOPS of DLR (Multi-Objective Parameter Synthesis) [12].

References

1. C. T. Kelley. *Iterative Methods for Optimisation.* SIAM, Philadelphia, 1999.
2. H.D. Mittelmann and P. Spellucci. Decision tree for optimization software. World Wide Web, `http://plato.la.asu.edu/guide.html`, 2001.
3. J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer Series in Operations Research. Springer-Verlag, New York, 1999.
4. C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal. Algorithm 778. L-BFGS-B: Fortran subroutines for Large-Scale bound constrained optimization. *ACM Transactions on Mathematical Software*, 23:550–560, 1997.
5. M.J.D. Powell. Direct search algorithms for optimization calculations. In A. Iserles, editor, *Acta Numerica, Vol. 7*, pages 287–336. Cambridge University Press, 1998.
6. A. R. Conn, K. Scheinberg, and Ph. L. Toint. A derivative free optimization algorithm in practice. In *Proc. of AIAA St Louis Conference*, 1998. (`http://www.fundp.ac.be/~phtoint/pht/publications.html`).
7. R. M. Lewis and V. Torczon. Pattern search methods for bound constrained minimization. *SIAM Journal on Optimization*, 9:1082–1099, 1999.
8. M.J.D. Powell. A direct search optimization method that models the objective and constraint functions by linear interpolation. In S. Gomez and J.P. Hennart, editor, *Advances in optimization and numerical analysis*, pages 51–677. Kluwer Academic Publishers, 1994.
9. P. Laarhoven and E. Aarts. *Simulated Annealing: Theory and Applications.* D. Reidel Publishing, 1987.
10. D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, Reading, MA, 1989.
11. W. Huyer and A. Neumaier. Global optimization by multilevel coordinate search. *J. Global Optimizationn*, 14:331–355, 1999.
12. H.-D. Joos, J. Bals, G. Looye, K. Schnepper, and A. Varga. A multi-objective optimisation-based software environment for control systems design. In *Proc. IEEE CADCS Symposium, Glasgow, UK*, 2002.