Surrogate Models for Aircraft Flight Control: Some Off-Line and Embedded Applications

J.M. Biannic, G. Hardier, C. Roos, C. Seren, L. Verdier

J.-M. Biannic, G. Hardier,
C. Roos, C. Seren,
(ONERA)

L. Verdier
(AIRBUS Operations SAS, EGYA)

E-mail: Georges.Hardier@onera.fr

DOI: 10.12762/2016.AL12-14

Surrogate Models for Aircraft Flight Control: Some Off-Line and Embedded Applications

The use of surrogate models is now very widespread in engineering activities to replace high-fidelity models, computation codes or simulators that are too complex or time consuming. This is especially the case in the aerospace field: although physical models are available, derived from aerodynamics, structural dynamics or flight mechanics, quite often they cannot be used just as they are. Hence, simplified representations need to be developed in order to achieve some tasks involving for instance optimization, parameter identification, embedded implementations, and so on. This paper illustrates some of these aspects in the field of aircraft flight control systems. After outlining the characteristics of the preferred surrogate models, the techniques that have been developed for constructing these models efficiently are presented. Then, three relevant off-line and on-line applications are described, with the surrogate models being respectively used: to create parsimonious representations useful for building Linear Fractional Representations for analysis and design of control laws, as intermediate models in the modeling of rigid aircraft to facilitate the identification process of aerodynamic nonlinearities from flight tests, and to obtain embeddable models used for the virtual sensing of some flight parameters required to schedule the control/protection laws.

Introduction

A number of activities in aeronautical engineering rely on the availability of models to represent the real behavior of the aircraft. For example, let us quote the analysis and design of flight control laws, the study of the handling qualities, the fault monitoring process, the prediction of hazardous behaviors, or the implementation of simulators used to train the pilots and to validate hardware and software systems. One feature of the aeronautical field is that many physical models are available, derived from aerodynamics, structural dynamics or flight mechanics. Accordingly, the development of Flight Control Systems (FCS) often makes use of model-based techniques (Figure 1), but these models cannot always be implemented just as they are because of high orders, strongly nonlinear behaviors or, more generally, because they have become increasingly complex in recent years. On the other hand, surrogate models are very useful to replace

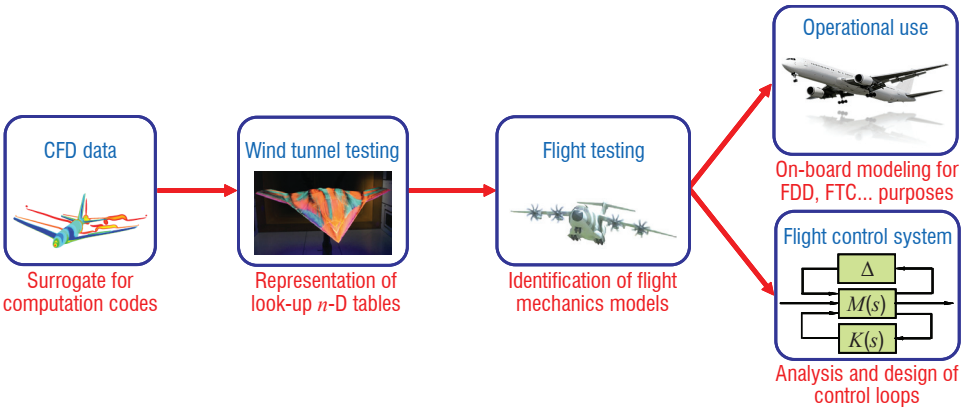

Figure 1 – Some development stages likely to benefit from surrogate models

the system or the reference model when the latter is too complex or time consuming for achieving some tasks (optimization, parameter identification, embedded implementation, etc.).

Consequently, a wide range of models and associated methods has been developed for building surrogate models efficiently, both accurately and parsimoniously [1][7][8][34][54]. Within this range, Neural Networks (NN) are recognized nowadays as an efficient alternative for representing complex nonlinear systems, including dynamical ones [14][21][49]. As a result, one can also benefit from efficient tools permitting the static nonlinearities to be modeled, such as those involved in the representation of aerodynamic coefficients resulting from CFD computations, or from wind tunnel or flight testing [26][40][47][52][55]. Usually, these data are only available in the form of look-up tables and hence are not very convenient for on-board implementation [4][22]; that is why analytical and differentiable approximations are contemplated, with lower memory requirements also. For instance, special types of NN can be advantageously used to design grey-box neural models arranged to accurately represent the aircraft aerodynamic coefficients appearing in the flight nonlinear equations of motion [5]. Such NN permit the physical readiness and the structure of aerodynamics to be preserved in the final surrogate model (as opposed to black-box approaches often promoted in the field), as illustrated below for the pitching moment coefficient Cm in clean configuration:

$$Cm = SlPd\left[\underbrace{Cm_0^{NN}}_{f(M)} + \left(x_{CG} - \underbrace{x_{AC}^{NN}}_{f(M)}\right)\underbrace{Cz_\alpha^{NN}}_{f(M)}\alpha + \underbrace{\kappa_{NL}^{NN}}_{f(Pd,M)}\underbrace{\Delta Cm_{NL}^{NN}}_{f(\alpha,M)} + \ldots\right] \quad (1)$$

In (1), the parameters α, M, S, Pd and l refer respectively to the aerodynamic Angle of Attack (AoA), Mach number, reference area, dynamic pressure and mean aerodynamic chord. x_{CG} and x_{AC} correspond to the longitudinal abscissa of the Center of Gravity and Aerodynamic Center. $\left[Cm_0\ x_{AC}\ Cz_\alpha\ \Delta Cm_{NL}\right]$ and $\left[\kappa_{NL}\right]$ are respectively rigid-body aerodynamic and static aeroelastic neural approximations of nonlinear effects (as indicated by the NN exponent) which contribute to the global model of Cm. From this simple example, it can be stated that if such surrogate models can be created from the initial data, several model-based techniques will be facilitated, both for off-line and on-line applications.

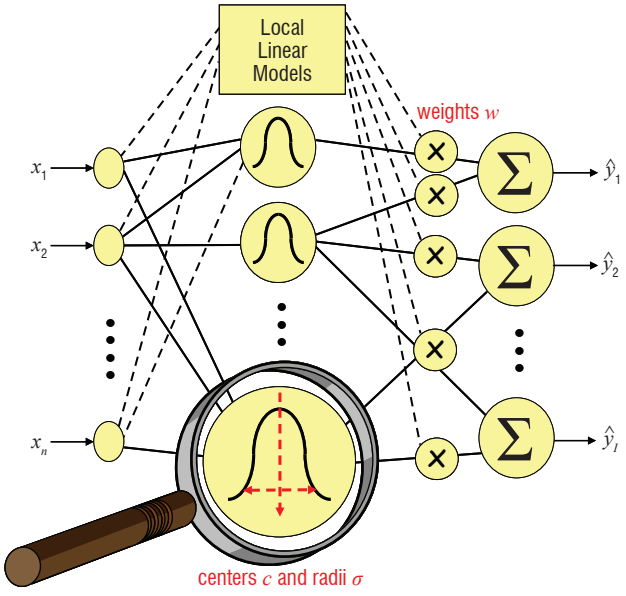

Figure 2 – RBF/LLM-type surrogate models

To cover these aspects, the paper is organized as follows. The first section outlines the characteristics of the surrogate models preferred for meeting the previous needs, and gives an overview of the techniques that have been developed for constructing these models efficiently. The three following sections are devoted to a short description of some relevant applications in the field of flight control systems:

- to derive simplified parsimonious representations useful for building Linear Fractional Representations (LFR) of lower complexity for analysis and design of control laws,
- as intermediate models to facilitate the identification process of aerodynamic nonlinearities from flight tests,
- to obtain embeddable models used for the virtual sensing of some flight parameters required to schedule the control/protection laws (tolerance to sensor faults). Concluding remarks and perspectives end the paper.

Surrogate models for aircraft modeling

The equations of motion of a rigid body A/C are derived from the fundamental principles of mechanics. Thus, the external forces involved in these equations arise from propulsion, aerodynamics and gravitational attraction. The major source of complexity in this model mostly comes from the highly nonlinear aerodynamic effects, and to a lesser extent from the propulsion components. Consequently, the modeling of the aerodynamic part within the whole flight envelope is the main challenge of many model-based applications. The following subsections outline the characteristics of the nonlinear surrogate models that were selected, and which play a pivotal role in the practical applications described afterwards, and focus on different aspects of flight control systems.

Nonlinear regression for Linear-in-their-Parameters models

At first, it is noteworthy that a nonlinear model can be either linear or nonlinear with regard to its internal parameters. Within the framework of NN, the latter case corresponds for example to Multi-Layered Perceptrons (MLP) [8][14] as well as to Radial Basis Function (RBF) networks [7][17][37][53] when the nonlinear parameters of the radial units (centers c and radii σ in Figure 2) are also optimized in addition to the linear ones (weights w in Figure 2). Clearly, this is the most general formulation since Linear-in-their-Parameters (LP) models are nothing but a special case, and it is the origin of the NN theoretical properties as parsimonious approximators. However, the joint optimization of the whole set of model parameters (linear plus nonlinear) practically results in ill-posed problems, which are likely to converge only by enforcing strong regularization constraints in the optimization process. This issue can be overcome by splitting the parameters into linear and nonlinear subsets [16], but LP models are always quite common practice anyway, because more simple and robust algorithms can be adopted, derived from the classical methods in use for estimating the parameters of linear regressions. Some advantages for preferring RBF versus MLP are:

- a possible integration of physical constraints,
- they make it easier to perform a local adjustment from heterogeneous data distribution or from sparse partial data relative only to portions of the flight domain,
- their grey-box architecture is better suited to integrate some initial knowledge (keeping the aerodynamic model readable) and to manage the optimization process (w.r.t. pure black-box networks),

- they develop local approximations instead of global ones making constructive algorithms possible (for determining the number of units),
- the various parameters (linear and nonlinear) can be sequentially optimized during the learning process.

On the other hand, most of the existing tools related to those local models are based on some initial user-defined architecture. For RBF, this amounts to setting *a priori* the number of kernels as well as their distribution in the input space. Although this distribution can be more or less automatically defined from the data repartition, and although some pruning is possible after learning, this is not a real structural optimization. Most often, only input data are used for choosing the kernel distribution without considering the output values, and hence the match between the model and the problem complexity cannot be guaranteed. Actually, by taking advantage of the features of LP models, both parametric and structural identification can be jointly proceeded to for extracting a suitable set of regressors from the available data. The procedure should determine the whole set of parameters involved in the nonlinear regression, and neither the number of kernels nor their distribution would have to be known in advance. The algorithm described in the sequel of this section makes use of the best techniques in the fields of Orthogonal Least-Squares (OLS) [9], local regularization [39], Separable Nonlinear Least-Squares (SNLS) [16], as well as a global evolutionary optimization that hybridizes local and global search [12].

The approach relies on a regularized constructive process that adds new regressors progressively, in terms of both a performance cost and the generalization errors. The LP models that we consider hereafter comply with the generic form:

$$\hat{y}_k = f(x_k) = \sum_{j=1}^{m} w_j r_j(x_k) \qquad (2)$$

where x_k is a vector gathering the n explanatory variables (α, M, Pd in the example of Eq. (1)), and y_k is assumed to be scalar to alleviate the developments. The functions r_j are the nonlinear regressors to be defined, while w_j are the regression parameters also to be determined during the optimization process, as well as the number m of regressors (unknown *a priori*). At this time, the regressors r_j can represent either the monomials of a polynomial expansion, the radial functions of RBF or Local Linear Models (LLM), etc. [34], but we will focus in the sequel on the use of local models (Figure 2). From N available data samples (x_k, y_k) representing the tabulated coefficient to be modeled, the LS minimization of the approximation errors yields to consider the criterion:

$$C = \sum_{k=1}^{N} \left(y_k - f(x_k)\right)^2 \quad \left[+\sum_{j=1}^{m} \lambda_j w_j^2\right] \qquad (3)$$

where adding an [optional] regularization term enables large values of the w_j parameters to be penalized, and thus the conditioning of the problem to be improved in case of model overfitting (ridge regression). By grouping the regularization parameters into the diagonal matrix $\Lambda = diag\{\lambda_j, j = 1,\ldots,m\}$ (also to be adapted), the well-known solution to this LS problem is given by:

$$\hat{w} = (R^T R + \Lambda)^{-1} R^T y = H^{-1} R^T y \qquad (4)$$

by denoting R_j as the j^{th} column of the regression matrix R, written as $R_j^T = \left[r_j(x_1) \quad r_j(x_2) \quad \cdots \quad r_j(x_N)\right]$, and H^{-1} as the covariance matrix of the estimation errors. It is noteworthy that, if $\Lambda = \lambda I_m$, this results in global regularization, whereas local ridge regression is expected when all of the λ_j are different from each other, every parameter w_j being weighted with its own penalty. When radial functions are used as regressors for instance, this form of regularization permits the resulting smoothness to be controlled in a local fashion [37]. If the individual λ_j are optimized, this smoothness can then be adapted in the different regions of the input space where the regressors are located. From (4), the output prediction error can be simply expressed as:

$$\varepsilon = y - \hat{y} = y - R\hat{w} = \left(I_N - RH^{-1}R^T\right) y = P y \qquad (5)$$

A matrix P appears in (5), which is the projection matrix in the simplest case (no penalty). It projects the vector y defined in a N-dimensional space in the m-dimensional subspace spanned by the model. This matrix plays a crucial role in the regression properties, and is involved in most of the relationships, *e.g.* the cost function $C = y^T P y$. To choose the regressors r_j, we will focus on methods based on forward selection, as opposed to another class of methods that consists in first selecting a full set of candidates, and then removing the less relevant ones one by one (backward elimination). Forward selection starts with an empty subset, and the regressors are added one at a time in order to gradually improve the results. Therefore, the final number of regressors is not known in advance, and the computational cost is reduced since the regression size will become large only if it is required to reduce the modeling error. We will examine in the sequel how the candidates can be chosen or generated from scratch. Let us first consider the question of how and when to stop this forward selection process. The available data being limited in number and sometimes noisy, the model that we seek should correspond to the best compromise between the performances evaluated on the training samples and the extrapolated performances evaluated when the model is used with other inputs absent from the initial data base. This is referred to as validation or generalization error. With a few not very restrictive assumptions, it can also be proved that the expected cost can be expanded as $E(C) \approx \mathrm{var}(\xi) + \mathrm{biais}(m) + \mathrm{var}(\xi)\tilde{m}/N$ (over all possible data sets), where $\mathrm{var}(\xi)$ is the variance of the noise ξ, and the effective number of parameters $\tilde{m}$ (usually less than m) depends on the regularization parameters.

To tackle this well-known bias/variance dilemma, the most common technique consists in splitting the available data set into two groups, the 1st one being used for estimating the parameters w and the 2nd one for the validation stage. Although irrelevant in practice, this trick leads to a more interesting concept, the virtual "Leave-One-Out" method (LOO) and its variants [29]. This technique generalizes the previous split by suggesting the virtual use of N subsets to be made, each of them including only N–1 samples, prior to computing the model performance by averaging the N estimations. The idea is to achieve a theoretical prediction (*i.e.*, without really performing the corresponding optimizations) of the generalization errors resulting from the withdrawal of every sample included in the data base. All of the available data are thus effectively used for the optimization process, but the validation stage takes also into account the consequences of discarding any of them. Moreover, the interest of this technique is reinforced in the case of LP models, since a very simple analytical expression of the generalization error can then be derived. By denoting as $f_k(x_k)$ the predicted output obtained for the k^{th} sample when the model is optimized from the N–1 remaining

samples, the LOO variance of the generalization error (also known as PRESS for Predicted REsidual Sum of Squares) can be effectively written as [37]:

$$\sigma_{LOO}^2 = \frac{1}{N}\sum_{k=1}^{N}\left(y_k - f_k\left(x_k\right)\right)^2 = \frac{1}{N}\sum_{k=1}^{N}\left[\varepsilon^{N-1}(k)\right]^2 = \frac{1}{N}y^T P diag(P)^{-2} P y \quad (6)$$

Forward selection methods

Orthogonal Least Squares

Thanks to this estimation of the generalization error, it is possible to rule on the benefits from adding a new regressor to the model, at every step of the forward selection process. On the other part, when using LP models, it is straightforward to determine by advance the result of elementary operations, like adding or subtracting any regressor. Again, quite simple analytical formulae can be established to cover these situations [37], which avoid retraining the model from scratch. At every step of a constructive approach, the chosen new regressor will thus be the one reducing the most the value of the cost function. The process will be pursued that way until: either a user-defined minimum value of the criterion C_m is reached (knowing that this criterion will continue to decrease as long as m increases), or the generalization error (*e.g.* the LOO one estimated by σ_{LOO}^2) stops decreasing which reveals model overfitting. Otherwise, determining the pool of regressors within which the selection will operate is of course a key point for these approaches. There are several alternatives, which can modify the final results and the model parsimony. This aspect is discussed afterwards.

Forward selection is computationally efficient, but constructive algorithms can be sped up even further thanks to a preliminary orthogonalization process, making use of the famous Gram-Schmidt technique. Moreover, this procedure permits the successive regressors to be decoupled from each other, and hence their individual contribution to be evaluated regardless of those already recruited for the modeling. The principle of the method is based on factoring the regression matrix as $R_m = \tilde{R}_m U_m$, where U_m is upper triangular, and where the columns of the orthogonalized regression matrix $\tilde{R}_m = [\tilde{r}_1 \;\; \tilde{r}_2 \;\; \dots \;\; \tilde{r}_m]$ are such that $\tilde{r}_i^T \tilde{r}_j = 0$ for $i \neq j$. Consequently, when adding a new regressor r_{m+1}, corresponding to the $(m+1)^{th}$ column of the regression matrix R, only its projection perpendicular to the space already spanned by the m first regressors needs to be considered, and can contribute to a further reduction of the criterion [37]. This projection yields the recurrence:

$$\tilde{r}_{m+1} = r_{m+1} - \sum_{j=1}^{m}\frac{\tilde{r}_j^T r_{m+1}}{\tilde{r}_j^T \tilde{r}_j}\tilde{r}_j \quad (7)$$

From P, assuming no weight penalty ($\Lambda = 0$), an expression can be derived for the expected reduction of the cost, depending only on the output vector y and the orthogonal regressor $\tilde{r}_{m+1}$ (decoupling). The regression parameters (orthogonal $\tilde{w}_m$ and ordinary ones $\hat{w}_m = U_m^{-1}\tilde{w}_m$) are then also easily derived [37]:

$$\begin{cases} P_m = I_N - \sum_{j=1}^{m}\frac{\tilde{r}_j \tilde{r}_j^T}{\tilde{r}_j^T \tilde{r}_j} \Rightarrow \; C_m - C_{m+1} = y^T(P_m - P_{m+1})y = \frac{(\tilde{r}_{m+1}^T y)^2}{\tilde{r}_{m+1}^T \tilde{r}_{m+1}} \\ \tilde{w}_m^T = [\tilde{r}_1^T y \;\; \cdots \;\; \tilde{r}_m^T y]\; diag\{(\tilde{r}_j^T \tilde{r}_j)^{-1},\, j = 1,\dots,m\} \end{cases} \quad (8)$$

Local regularization and Separable Nonlinear Least Squares

Fortunately, the advantages of forward selection and regularization techniques (whether local or global) can be combined without too much complexity in the resulting algorithms. Furthermore, when radial functions are used as regressors (*e.g.* in the case of RBF or LLM), an analytical solution can usually be found [37], allowing the optimal value of each λ_j (for a local regularization) to be determined. However, this value is related to those of the $(m-1)$ other parameters λ_j, and an iterative process is required *a priori*. Another approach is explained below, resulting in a joint or alternate procedure to optimize both regressors and regularization parameters. It can also help to reduce the modeling size by removing a few regressors that have become irrelevant. Effectively, if a parameter value λ_j converges towards $+\infty$, the j^{th} regressor can be eliminated without any damage to the performances. Thus, introducing a local regularization into the forward selection incidentally enables useless regressors to be pruned without having to implement backward elimination. By combining OLS and local regularization, the LOO/PRESS validation error becomes $\varepsilon^{N-1}(k) = \varepsilon_m^N(k)/\eta_m^N(k)$, where the modeling error $\varepsilon_m^N(k)$ appears weighted by a coefficient $\eta_m^N(k)$ which contributes to the computation of the variance of the validation error [9]. A major advantage of this expression is to facilitate the implementation of a recurrent formulation, resulting in a very effective algorithm [9]:

$$\begin{cases} \varepsilon_m^N(k) = y_k - \sum_{j=1}^{m}\tilde{r}_j(k)\tilde{w}_j = \varepsilon_{m-1}^N(k) - \tilde{r}_m(k)\tilde{w}_m \\ \eta_m^N(k) = \eta_{m-1}^N(k) - \frac{\tilde{r}_m^2(k)}{\tilde{r}_m^T \tilde{r}_m + \lambda_m} \end{cases} \quad (9)$$

initialized only from the data outputs (no regressor to start the process): $C_0 = (y^T y)/N$, $\varepsilon_0(k) = y_k$ and $\eta_0(k) = 1$ (for $1 \le k \le N$). Regarding the joint search for optimal values of the λ_j (in addition to those of the model parameters w), it amounts to minimizing the cost $C(w,\Lambda) = \varepsilon^T\varepsilon + w^T\Lambda w$ and can be solved within the framework of Bayesian learning [9]. Finally, this iterative process involves the evaluation of the $\tilde{w}_j$ parameters in terms of the current λ_j, followed by an adaptation of the λ_j and so on. On the other hand, despite the unquestionable pros of the previous forward selection, the choice of each regressor is optimal iff it is considered by itself, *i.e.* for a given pre-selection of the pool of its predecessors. Disregarding this matter results in sub-optimality, which is the cost to pay for avoiding having to deal with a nonlinear complex optimization, involving both the regression parameters and the parameters of the regressor kernels. Above all, it has to do with the major issue that the number of regressors is not known in advance, and has to be inferred at the same time. However, it is possible to win on both counts, and thus to improve the forward selection, by adding some extra optimization stages to the previous constructive algorithm. The idea is to periodically call the global positioning of the selected regressors into question, in order to minimize the criterion value for a given model size, before continuing with the addition of new terms.

To improve the course of these optimization stages (time and conditioning), it is wise to benefit from particular LS techniques, referred to as SNLS, and developed for LP models [16]. Denoting by u the vector gathering all of the internal parameters defining a regressor (*e.g.*, the centers and radii of a RBF node), the sum-squared error (3) is expressed without loss of generality as: $C(u,w) = \varepsilon^T\varepsilon$ with $\varepsilon = \hat{y} - y = R(u)w - y$. It is worth noting that a regularization

term is no longer needed since the number of regressors is fixed for the time being. From this expression, the trick of the SNLS methods comes from the fact that the optimal value of the parameters w results directly from the value of the parameters u. In other words, for a given u, w is available by solving an ordinary LS problem: $w(u) = \left[R^T(u)R(u)\right]^{-1} R^T(u)y = H^{-1}R^T y$. Therefore, it would be useless and even detrimental to process both sets of parameters at the same time. By using the previous expression of $w(u)$ every time the cost function needs to be evaluated, it appears that we can consider C as a function of u only. The optimization stages can thus be restricted to this vector, reducing the dimension of the parameter space. It is also noteworthy that they will converge nicely and quickly even if 2nd order standard algorithms (Gauss-Newton type) are used, which are known for their sensitivity to the initial conditions. There are two reasons for this: the conditioning is greatly improved by the SNLS formulation, and the nonlinear optimization starts under good conditions thanks to the initial parameters u available from the constructive algorithm. Finally, after every optimization stage, a new orthogonalization of the regressors obtained from the resulting u_{opt} is also necessary before starting the forward selection again.

Determining the set of regressors

To implement the previous forward selection, two options are available: ① to first define an initial pool of candidate regressors from which the most relevant ones will be selected, ② to determine each regressor individually as the process goes on, which generally amounts to optimizing the kernel functions in the input space. Within class ① is the whole range of classical and direct methods that locate the regressor kernels quite arbitrarily: in a subset of the data samples, on the knots of a lattice derived from a gridding of the input space, by using data clustering or self-organization techniques. These approaches have the advantage of being very simple to implement and of providing a pool of candidate regressors almost instantly. On the other hand, they do not provide any guarantee regarding the balance between the distribution of the regressors and the real complexity of the modeling problem, since they do not use all available information (only input data are used). More elaborate methods attempt to fill this gap by building regression trees that do use the values of data outputs [38]. They recursively partition the input space, approximating the output values by their average value computed from the samples included in each partition. The main weakness of these methods is to be very greedy, the bias/variance dilemma resulting in the following interrogation: when should the branching out of the tree stop, and how should it then be pruned?

Class ② is related to optimization techniques, but to avoid the problems inherent to classical methods (convergence, sensitivity to initial values) global optimization is preferred, among which evolutionary algorithms have done particularly well for some years. New techniques have thus appeared such as Genetic Algorithms (GA), boosting search, or simulated annealing [10][53]. In return for their conceptual simplicity, these classical evolutionary algorithms can be computationally very expensive by requiring a number of cost function evaluations, and they often require the tuning of several internal parameters. More recently in the 90's, a new metaheuristics also arising from biological inspiration (bird flocking or fish schooling) was imagined, known as Particle Swarm Optimization (PSO). The collective behavior of the particles looks like a swarm of living beings, and the most relevant metaphor certainly concerns the bees because an individual having discovered a good spot passes on the information to the others, and is used to direct their next moves. Therefore, the swarm represents a set of autonomous and interacting agents, cooperating to solve a problem. The members of a group benefit from the accidental discoveries as well as the experience acquired by other individuals. Similarly to the evolutionary case, the method is based on an iterative and stochastic process [12]. At iteration i, the position of the particles $\vec{u}_i$ (which includes all of the parameters to be optimized) and their velocity $\vec{v}_i$ are updated as follows:

$$\begin{cases} \vec{v}_{i+1} = c_1\vec{v}_i + c_2U(0,1)\otimes\left[\vec{p}_i - \vec{u}_i\right] + c_3U(0,1)\otimes\left[\vec{g}_i - \vec{u}_i\right] \\ \vec{u}_{i+1} = \vec{u}_i + \vec{v}_{i+1} \end{cases} \quad (10)$$

where $\vec{p}_i$ corresponds to the best position ever reached by particle i, and $\vec{g}_i$ to the best position ever reached by the pool of its informants. $U(0,1)$ is a random number between 0 and 1 chosen following a uniform distribution, and $\otimes$ is a symbolic operator for the element-by-element product of two vectors. Thus, it appears that a particle updates its velocity through a weighted linear combination of three behaviors [12]: an *adventurous* behavior for preserving the acquired velocity (inertial component of the motion), a *conservative* behavior consisting of getting closer to its best position (*cognitive* component of the motion), and a *sheeplike* behavior for getting closer to its best informant (*social* component). It is also worth noting that the required number of particles remains quite low as opposed to usual GA populations, and that PSO algorithms are very robust regarding their tuning: coefficients c_i and swarm size T [13].

From this historical and basic version, a number of variants were studied during the last decade to improve the performances. They are based on various strategies for determining the informants (fixed or random topology), population splits (memory vs explorer swarms), the proximity distributions used to guide the bounded motion of the particles, etc. Adaptive versions were also developed to save the user from having to set these elements *a priori*, by self-adjusting them during the run. Hence, a recent extension is aimed at having several tribes of particles cooperating, with the idea that many swarms should be more efficient to explore the different regions of the search space than only one [12]. The most promising techniques have been selected and implemented in the PSO code developed by ONERA to optimize the regressor positioning (which is part of the *koala* tool described below). A detailed description of this software is beyond the scope of this paper. Thus, only a brief survey of the main functionalities offered is given hereafter, with some references for readers interested in obtaining more details:

- *fixed and adaptive topologies* → from static (star, ring, Von Neumann) to dynamic ones (*e.g.* Delaunay neighboring) [24],
- *particle displacement* → standard, with constriction factor, FIPS and weighted FIPS versions of the velocity update laws [28],
- *hybrid local/global method* → to speed up the convergence with direct search (improved Nelder-Mead, Delaunay tessellation for the initial simplex),
- *multiswarm strategies* → for competing swarms or for partitioning the search domain into several subregions [51],
- *diversity analysis* → to provide information about the swarm dispersion and to refine the convergence tests [35],
- *swarm initialization* → from random to low discrepancy sequences (Hammersley, centroidal Voronoï diagram) [13],
- *competitive multirun* → to benefit from several topologies, algorithm variants and tuning,
- *charged vs neutral particles* → cooperation of particles with different physical properties [3].

A Kernel Optimization Algorithm for Local Approximation (KOALA)

The coupling of this PSO algorithm with the constructive approach based on forward selection allows structural and parametric optimizations to be proceeded to jointly for various types of regressors with local basis (as opposed to [10] where a basic version of PSO is used). In the *koala* software developed by ONERA for that purpose, this approach is applied to various kernel-based NN such as RBF and LLM, provided some adjustments of the orthogonalization method. LLM generalize RBF [34], by replacing the linear weights (scalar w for RBF networks) by an affine expression depending on the model inputs (see the dotted connections in Figure 2). They gave rise to the famous *LOLIMOT* algorithm (LOcal LInear MOdel Tree). It is thus expected that fewer radial units will be required to achieve the same accuracy in most applications. The generic formulation (2) used to represent LP models remains thus suitable, but needs only to be adapted to this case by using an extended set of regressors $r_l^{\#}$, because a new kernel now breeds a subset of regressors and not only one as it was for RBF:

$$\hat{y}_k = f(x_k) = \sum_{j=1}^{m}\left(\sum_{i=0}^{n} w_{ji} x_k^i\right) r_j(x_k) = \sum_{l=1}^{m(n+1)} w_l\, r_l^{\#}(x_k) \quad (11)$$

denoting by x_k^i (for $i = 1$ to n) the value of the i^{th} input variable for the k^{th} data sample, and setting $x_k^0 = 1$ to include the constant terms of the local affine modeling in the 2nd sum. Hence, it will remain possible to use the constructive algorithms developed for any type of regression, but some slight adaptations are required to take the peculiarities of the kernel functions $r_l^{\#}$ into account. When adding or subtracting terms, the group of regressors sharing the same kernel r_j needs to be considered as a whole, and no longer separately as was the case for RBF, polynomials, etc. In any case, the algorithm is aimed at gradually selecting a series of regressors by optimizing their kernel parameters, *i.e.* the ellipsoid center c and radius σ related to each radial unit in Figure 2 (both vectors). Considering the n explanatory variables x^i, the components of the vector u (particle position) can be ordered arbitrarily for instance as $u_i = c_i$ if $1 \le i \le n$, and $u_i = \sigma_{i-n}$ if $n+1 \le i \le 2n$. To sum up, the improved performance of the *koala* tool results from two complementary aspects: on the one hand, applying efficient OLS-based forward selection and SNLS optimization to a more powerful modeling (LLM) and, on the other hand, implementing a new PSO algorithm that outperforms the standard one.

Application #1: building efficient LFR for control analysis and design

For this first application topic, the motivation is to use surrogate models to improve the creation process of Linear Fractional Representations (LFR) of reduced complexity. It happens that the final LFR object relies on rational functions which hence can be considered as simple surrogate models, but actually we will see in this section that more complex surrogate models will be used during the building process to obtain these rational functions *in fine*. Let us remember that a LFR is a model where all of the known and fixed dynamics of a given system are placed together in a linear time-invariant plant M, while the uncertain and varying parameters are stored in a perturbation matrix Δ (Figure 3). LFR modeling is now a widely spread and a very efficient tool in the fields of system analysis and control design. It notably allows the robustness properties of uncertain closed-loop plants to be evaluated (*e.g.* using μ-analysis or Lyapunov-based methods), and robust control laws (especially using H_∞ approaches) or gain-scheduled controllers to be designed [56]. However, the efficiency of the aforementioned analysis and synthesis techniques strongly depends on the complexity of the considered LFR, which is measured in terms of both the size of the matrix Δ and the order of the plant M. An increase in complexity is usually a source of conservatism, and can even lead to numerical intractability.

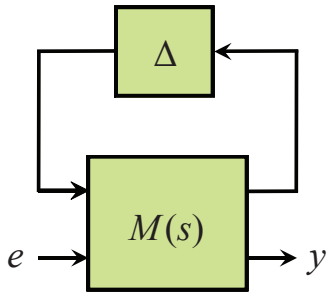

Figure 3 – LFR object

In most industrial applications, physical systems are described using a mix of nonlinear analytical expressions and tabulated data. Therefore, a two-step procedure has to be implemented to obtain a suitable LFR: a linear model with a rational dependence on the system parameters is first generated, and then converted into a linear fractional form. Several techniques exist, such as object-oriented realization, to perform the latter transformation. Although the minimality of the resulting LFR cannot be guaranteed, symbolic preprocessing techniques, as well as numerical reduction, usually enable complexity to be overcome. Efficient software such as the LFR Toolbox for Matlab© is also available (see [27] and references therein for an overview of LFR modeling).

On the other hand, the preliminary issue of converting the tabulated or irrational data into simple yet accurate rational expressions has been paid much less attention, although it is of significant practical importance. In the aeronautic field for example, most aircraft models include tabulated aerodynamic coefficients determined by CFD, wind tunnel experiments or flight tests, and several controller gains depend on the flight parameters in a tabulated fashion. The motivations for addressing the issue of tabulated data approximation are twofold. The first one is of a physical nature: computing parsimonious rational expressions, for which the number of terms in the numerator and denominator is as low as possible, is a natural way to prevent data overfitting and to ensure a smooth behavior of the model between the points used for approximation. On the other hand, building a LFR from a polynomial or a rational expression $f(x^1,\ldots,x^n)$ results in a block diagonal matrix $\Delta = \mathrm{diag}[x^1 I_{p_1},\ldots,x^n I_{p_n}]$. The number p_j of repetitions of each parameter x^j in Δ is strongly linked to the number of occurrences of x^j in f. Although this is not an exact rule, the trend is as follows: the fewer the occurrences of x^j in $f(x^1,\ldots,x^n)$ are, the smaller the size of Δ will be. In other words, no matter how efficient the LFR generation tools can be, they are of little help if the rational expressions to be converted are unnecessarily complex. Hence, the need to obtain tractable LFR for control analysis or design purposes is another strong motivation for generating sparse rational expressions.

For a given accuracy, an intuitive idea is to determine a rational function for which the numerator P and denominator Q are two polynomials of the lowest possible degrees. This fairly simple strategy is followed by most existing methods. A classical linear least-squares (LS) technique is notably implemented in the LFR Toolbox [27] in case the rational function is restricted to be polynomial. In the general case, a nonlinear LS technique, implemented for example in the

Curve Fitting Toolbox of Matlab©, strives to minimize the approximation error, whereas a Quadratic Programming problem solution [6] ensures that the resulting rational function intersects a set of intervals containing the data. However, all of these techniques suffer from the same drawback: all admissible monomials of P and Q are usually nonzero, regardless of their real ability to model the data. More generally, the question of which terms should be included in the model is often addressed by trial-and-error, or even ignored in practice. A way to deal with this question is to use orthogonal LS (OLS), which allows the ability of each monomial to efficiently model the data to be evaluated, and therefore only the most relevant ones to be selected leading to sparse expressions. This approach was applied in [32][33][41] to model aeronautical data with polynomials, but practical methods leading to rational expressions are still lacking. Yet, the additional degrees of freedom offered by such expressions are likely to lead to simpler expressions and thus to smaller LFR [43][44].

In this context, a new method has been developed to compute sparse rational approximants, *i.e.* as few monomials in P and Q as possible, by using an indirect approach that first builds a parsimonious model based on LLM networks, before translating the result into a fractional form. Note that a direct approach for computing a rational approximant in a single step thanks to a symbolic regression technique is proposed in [18], which uses another recent evolutionary algorithm (Genetic Programming) to select sparse monomials. A first idea for such an indirect approach capitalizing on the tool *koala* would be to convert (11) *a posteriori* into a rational form. By choosing Gaussian radial functions, this regression is expressed as the sum of m terms, the j^{th} one being for any x:

$$f_j(x) = \left(\sum_{i=0}^{n} w_{ji} x^i\right) r_j(x) = \left(\sum_{i=0}^{n} w_{ji} x^i\right) \exp\left[-\sum_{i=1}^{n} \frac{\left(x^i - c_{ji}\right)^2}{\sigma_{ji}^2}\right] \quad (12)$$

It is thus possible to use Pade approximants of the exponential function, so as to replace it by a rational function in reduced form $\exp_{[p,q]}$. The latter is expressed as the quotient of two polynomials of p^{th} and q^{th} degrees, and the corresponding approximant to $f_j(x)$ becomes a rational function of $(2p+1)^{\text{th}}$ and $2q^{\text{th}}$ degrees for every explanatory variable x^i. However, obtaining high-quality approximants (*e.g.*, decreasing rapidly to 0 as x^i increases) would require large values of q (with $q - p > 2$ or 3). Hence, the degree of the resulting rational function would be penalized, with no guarantee as regards the accuracy of the global regression $f(x)$. Consequently, a more relevant approach consists in replacing the exponential function straight away by such an approximant, and then using this new kernel form straightaway during the optimization of the regression. The simplest transform corresponds to the reduced form $\exp_{[0,1]}$, which yields a sum of m components like:

$$f_j(x) = \left(\sum_{i=0}^{n} w_{ji}\, x^i\right) \Bigg/ \left(1 + \sum_{i=1}^{n} \left(x^i - c_{ji}\right)^2 / \sigma_{ji}^2\right) \quad (13)$$

Accordingly, another class of models was added to the RBF/LLM kernels proposed by *koala*, based on the Pade approximant $\exp_{[0,1]}$. Thanks to this new form, it is also worth noting that the computational complexity of the surrogate modeling can be adjusted to limited coding requirements, if any, prior to on-board implementation. This remark does not apply to LFR models, but rather to other applications for which the exponential operator, for instance, is not available on aircraft computers (see Application #3 on embeddable models for virtual sensing of flight parameters).

Finally, it must also be mentioned that the post-processing of the resulting regression, prior to the derivation of the LFR, makes use of the Matlab© Symbolic Toolbox. Again, several options exist for gathering the m components $f_j(x)$ into a single rational function: global expansions of the numerator/denominator, factorization of the denominator, and sum of elementary rational terms. The latter appears to be the most relevant since it favors some simplifications when building the final LFR. A factor of two (in the scalar case) can usually be gained in the final LFR size. When the simultaneous modeling of several coefficients is considered, the benefit from using this approach is maximized with a LFR size usually reduced by a factor equal to the number of coefficients if the complexity of the nonlinear coefficients is like-for-like (and hence does not require a significant increase in the number of kernel functions).

This work takes place within the framework of a more general ONERA project aimed at developing a *Systems Modeling, Analysis and Control* (SMAC) toolbox [42]. This Matlab/Simulink© library is being developed to provide both researchers and control engineers with a complete set of tools for making the design, tuning and validation of control laws easier. More precisely, the purposes of the SMAC project are to control aeronautical vehicles throughout their whole flight domain in the presence of nonlinearities, uncertainties, external disturbances and imperfectly measured or estimated data, while obtaining strong guarantees w.r.t. the stability margins and the performance levels. A free version of SMAC can be downloaded from w3.onera.fr/smac, which includes three kinds of tools:

- **Modeling tools**, which allow the considered physical systems (usually represented in an industrial context by using a mix of nonlinear analytical expressions and tabulated data) to be described as a single parameterized model. Typically, they are aimed at creating accurate LFR with sizes that are as reduced as possible, in order to facilitate the subsequent use of the design and the analysis tools described in the next two items. The APRICOT library (*Approximation of Polynomial and Rational-type for Indeterminate Coefficients via Optimization Tools*) includes a set of optimization tools to convert numerical data into simple yet accurate polynomial or rational expressions [43][44], and notably implements the adaptation of the tool *koala* described in this section. A limited version is available at w3.onera.fr/smac/apricot, and can be applied to simple cases ($n \leq 2$, $N \leq 100$). The GSS library (*Generalized State Space*) then converts the resulting expressions into low-order LFR. Note that the GSS library replaces and extends the LFR toolbox [27], which is no longer maintained.

- **Control design tools.** The Convex Synthesis library is dedicated to convex synthesis of LTI and LFT feedback controllers using Youla parameterization. The SAW library is a collection of Matlab/Simulink© tools for *Saturated systems analysis and Anti-Windup design*, and the OISTeR library (*Output to Input Saturation Transformation extensions for Robustness*) allows output saturations to be transformed into input saturations, which can then be handled with the SAW library. An additional library will be available soon. It will combine robustified nonlinear dynamic inversion techniques, structured H_∞ synthesis and anti-windup compensation, so as to produce simple yet powerful controllers, which can be easily implemented but do not require any interpolation as is the case with classical gain-scheduling techniques.

- **Analysis and validation tools**, which permit the robustness properties of the resulting closed-loop systems to be evaluated. Let us mention the SMART library (*Skew-Mu Analysis based Robustness Tools*) [42], which implements most of the μ-analysis based algorithms developed by ONERA over the last 15 years, and allows systems with parametric uncertainties and unmodeled dynamics to be handled. A frequency-domain and a time-domain IQC-based techniques are also available in two dedicated libraries. They make it possible to consider time-varying parameters and hard nonlinearities such as saturations, deadzones and sector nonlinearities, in addition to model uncertainties.

To illustrate the use of some of the tools included into the APRICOT library, and relying on surrogate modeling to create LFR, a realistic example is processed (Figure 4) involving a set of 3 aerodynamic coefficients depending

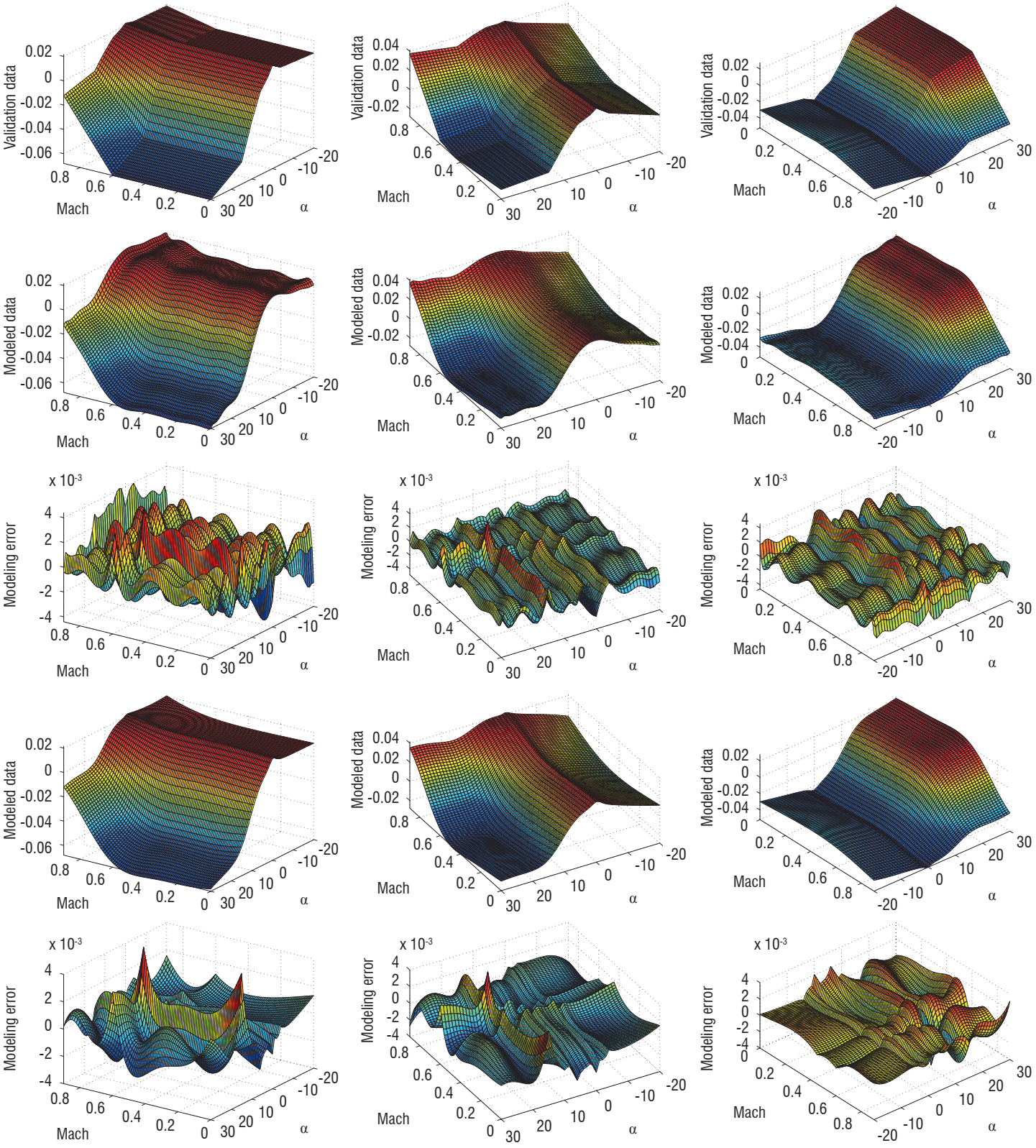

Figure 4 – Comparison of LFR results achieved by polynomial and rational models (1st row=reference data, 2nd and 3rd rows=polynomial approximations and modeling errors achieved by *olsapprox* 4th and 5th rows= rational approximations and modeling errors achieved by *koala*)

on 2 flight parameters (Angle of Attack α and Mach number). More details on this benchmark are available at w3.onera.fr/smac/?q=apricot_example1. These coefficients are depicted on a fine 50x90 mesh in the top row of Figure 4, a gridding which is used as a validation set to evaluate the approximation results achieved with a rougher 40x60 mesh of learning data. The middle rows of Figure 4 display the results achieved by using a polynomial surrogate model to create the LFR (routine *olsapprox* of the library which implements the same OLS-based forward selection to add relevant monomials one by one). A maximum degree of 12 is set for each explanatory variable and for the monomials, resulting in a size of 40 for the final LFR. On the other hand, the bottom rows of Figure 4 display the results achieved by using a rational surrogate model to create the LFR (routine *koala* described in the previous sections). A maximum degree of 12 is also set for a fair comparison, resulting in the creation of 6 kernels and in a size of 24 for the final LFR. In addition to providing a much smaller LFR size, the modeling accuracy is also better (improvements in the RMS errors of 33%, 15% and 25%, respectively). For this type of application, *i.e.* seeking a common model for a set of rather complex coefficients, using a powerful approach to construct a rational approximant presents a distinct advantage. Although the degrees of the polynomials would be globally the same if expanded, the conversion process of the rational expression into a LFR fully benefits from the factorized form of the rational function which is shared by all of the coefficients to be modeled.

Unfortunately, things are not that black-and-white and counter-examples prove that there is no definite answer for choosing either one or the other surrogate model. For example, when seeking LFR models in matrix form involving a set of low-complexity coefficients, the pros of rational expressions can be wasted during the final conversion stage into LFR form. In this case, accurate polynomial approximants can be achieved with low degrees, and can provide very satisfactory LFR with much smaller sizes. This is illustrated by the following aircraft benchmark, also available with the SMAC toolbox, corresponding to a civil transport A330-like aircraft. The goal is to obtain low-order but accurate open-loop LFT models covering a significant part of the operating domain (during the approach and landing phases), in order to perform control law validation and eventually to find worst cases for improving the design. To achieve this, a realistic nonlinear model of the aircraft is linearized at various values of the airspeed, altitude, mass, Center of Gravity location, and temperature. Accordingly, a family of 3000 linear models is obtained from which short-term longitudinal and lateral models are extracted. In this benchmark, the main challenge is not the complexity of the coefficients to be interpolated, but rather their high number (for longitudinal models, one has 26 varying coefficients distributed in a sparse 6x8 matrix), as well as the significant number of explanatory variables.

These two properties tend to generate high-order LFT models especially when using rational interpolation techniques. Consequently, the LFT-based representations are computed here by using polynomial interpolation only. A first and very accurate (global relative error = 0.5%) model is obtained by a standard LS technique (routine *lsapprox* of APRICOT) using polynomials of the third degree (resulting in full expressions with 55 monomials per coefficient). The global size of this first model is 109. Next, an enhanced OLS optimization technique is used (routine *olsapprox*)

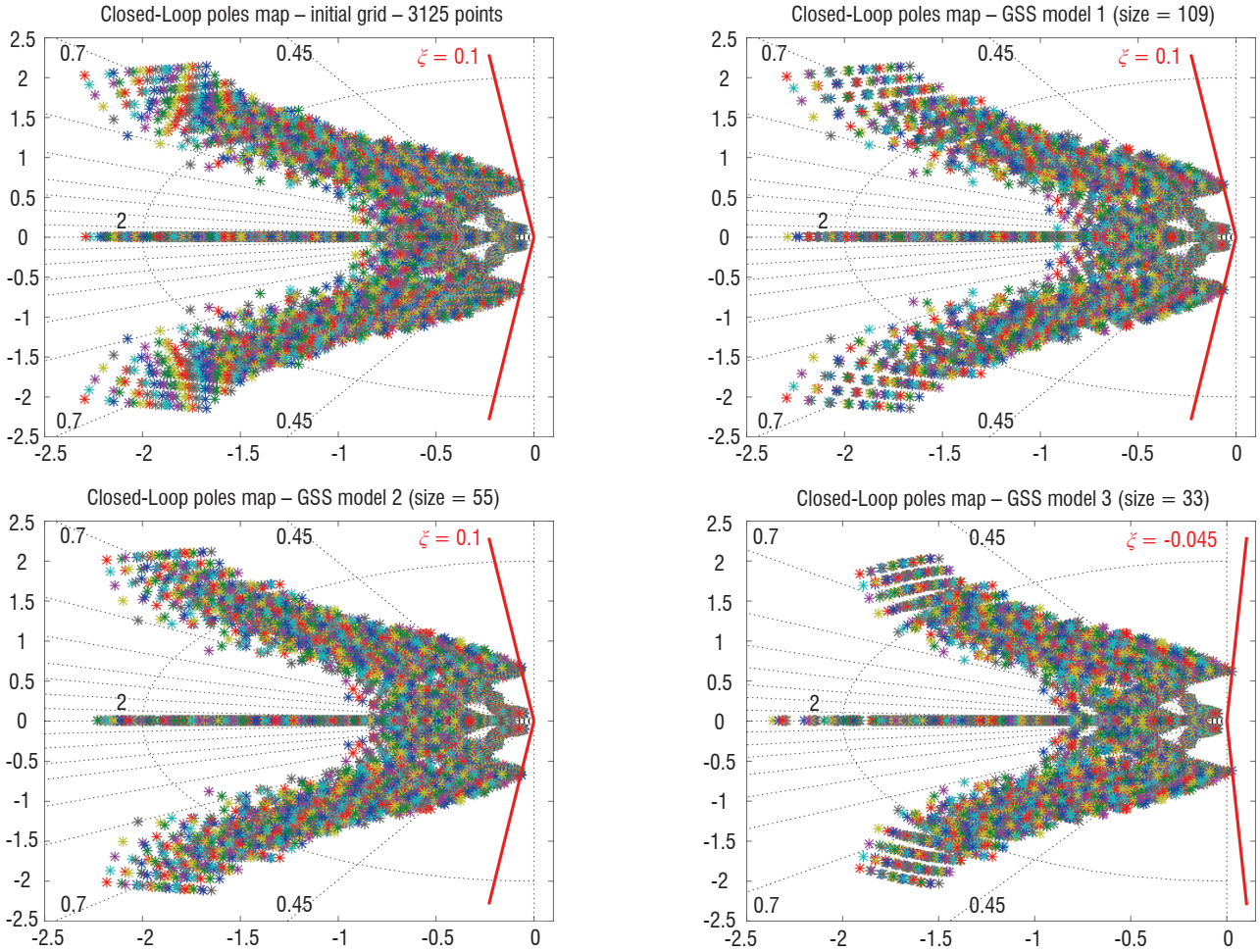

Figure 5 – Comparison of the longitudinal closed-loop poles of a civil aircraft over a large operating domain before and after LFT modeling (upper-left = reference data, upper-right = high-fidelity surrogate model, lower-left = medium-fidelity surrogate model, lower-right = low-fidelity surrogate model)

which enables the sparsity to be better controlled. The required accuracy is successively set to 1% and 5%, resulting in very sparse polynomials for the 26 coefficients with a number of monomials from 3 to 26 (10 on average) in the 1st case, and from 2 to 12 in the 2nd case (4 on average). Two reduced-order LFT models (referred to as GSS Models 2 and 3 respectively) are obtained. Their global sizes are significantly reduced to 55 and 33, which makes these models quite attractive for robustness analysis provided that the degraded accuracy does not compromise the analysis.

To evaluate the impact of the quality of the approximation, a preliminary validation of the models is easily performed by comparing closed-loop pole maps in the complex plane for a given basic control system designed to stabilize the nominal plant. The result of this validation is visible in Figure 5. The upper-left subplot shows the closed-loop poles directly obtained from the initial family of 3000 linear models. Next, the upper-right, lower-left and lower-right subplots respectively display the closed-loop poles corresponding to high, medium or low-order GSS models. Quite interestingly, one observes that the medium-order model (size 55) is still perfectly representative of the initial set. On the other hand, the lower-order model is not accurate enough in the whole operating domain; however, further investigations have shown that this model is still quite satisfactory in a reduced domain, and hence can be used for a more local validation. This benchmark shows that even simpler (polynomial) surrogate models can be obtained using the constructive OLS-based optimization techniques described in the previous sections, which are quite efficient to obtain low-order reliable LFR. The order reduction achieved by these advanced algorithms is essential for the robustness analysis tools. These will later provide stability and perfor-mance certificates more efficiently, or alternatively some worst cases useful to further improve the controller design.

Application #2: identification of aerodynamic non linearities

The concept of model identification refers to a set of tasks required to determine, and then to tune, a suitable modeling likely to explain the experimental behavior of the system. This involves choosing the type of mathematical relationships linking the i/o observed variables (often denoted as structural identification), as well as adjusting the unknown parameters of these equations (the so-called parametric identification). The early developments in system identification date back to the 70's, but this topic remains the subject of new developments nowadays, especially for aeronautics. They apply to the modeling of both rigid aircraft, described by the flight mechanics equations, and flexible aircraft where structural deformations are considered (see [5] and references therein). Owing to its mission, ONERA plays an intermediate role between academy and its main industrial partners. Within the framework of identification as in others domains, this role presupposes that new promising methods are investigated, adapted and transposed whenever necessary, in order to be evaluated through aeronautical applications. Accordingly, ONERA has been working with French aircraft manufacturers (Airbus and Dassault Aviation) on a large spectrum of themes and methods [5], applying these through a succession of industrial programs from the A320 to the A380, the Rafale, UAVs, etc. A close cooperation with the Flight Mechanics and Simulation Department of Airbus has been running for about 30 years, and has led to the implementation of several software codes in their identification toolbox for an operational use [25]. This continues nowadays through research programs aimed at improving the industrial process that allows the aerodynamic model to be fitted more rapidly into a larger part of the flight envelope.

Identification is aimed at producing accurate models to represent the behavior of the real aircraft. Given that pre-flight modeling obtained from CFD, wind tunnel, or ground tests is seldom faithful enough, this accuracy is achieved thanks to a final updating stage from a set of particular flight tests devoted to the ultimate model refinement. However, the aerodynamic models used for example for rigid aircraft, are also becoming drastically complex since from now on they integrate several effects that were disregarded before, or simply because the airplanes themselves have become much more complicated. Let us mention the A400M for the propeller blasts, or the A380 with an unprecedented proliferation of control surfaces. Hence, the identification becomes a much trickier task, but is also more crucial than it was in the past. Although physical models are available, derived from aerodynamics, structural dynamics or flight mechanics, quite often they cannot be implemented into the identification algorithms just as they are, because of high orders or strongly nonlinear behaviors. Hence, they require simplified grey-box representations to be developed, of linear or nonlinear types. These surrogate models facilitate the processing of the multivariate aerodynamic nonlinearities, usually complex and poorly structured. In addition, some constraints should be respected:

- the aircraft simulation requires continuous-time differential equations to be integrated,
- *a priori* knowledge about the predicted A/C behavior should be considered,
- a physical understanding and interpretation of the results is mandatory and introduces additional constraints into the optimization process.

The need to reduce the duration and the cost of the identification tests taking place during the first flights of a new airplane also requires specific techniques to be developed for designing, and then for processing this type of tests, without degrading the quality of the resulting models.

The variety of problems and models under consideration entails having a wide range of identification techniques available. Obviously, these include basic methods, such as LS or Maximum Likelihood and their variants, estimators based on Kalman filtering, etc. Practically, they result in 3 major categories of methods [5][23] for minimizing the Equation Errors (EE), Output Errors (OE) or Filter Errors (FE). Most of these methods are not directly usable as they are and need to be adapted to the peculiarities of aeronautical problems. For example, a very restrictive point comes from the requirement of coupling the identification tools with industrial simulators, in order to facilitate the implementation of the results and incremental updating of the aerodynamic data. This is all the more restrictive because these simulators also suffer from increasing complexity, involving a drift of the computational costs which is hardly compatible with the number of simulations required by the identification procedure. Another major concern arises from the will to automate and to systematize the test processing, to reduce its duration and to facilitate its progress. Actually, the whole set of available flight tests represents a huge amount of data, and hence some semi-manual steps in the process are especially tedious for the engineers responsible for sifting through the data. Automation also enables a global and joint processing of many tests, as well as the gradual introduction of new tests as they become available during the preliminary test campaigns.

Techniques based on linear or weakly nonlinear models are efficiently used as a first step of the identification process by processing only tests flown under close flight conditions [25]. However, when a global model is sought, valid over an extended area of the flight domain and including all aircraft specific nonlinearities, appropriate approaches and techniques must be applied. The task is all the more complex because the aerodynamic non-linearities are only available in the form of multivariate look-up tables (depending on configuration, Mach, AoA, sideslip, deflection angles, and dynamic pressure) which are not suited for identification algorithms. In industry, this global modeling is typically obtained after a long iterative process mainly based on EE approaches, and the result can be dependent on the skill of the performing engineers. For that matter, ONERA has been developing a so-called *hybrid identification* approach for several years, which might also achieve a more automated processing of the flight data. The ultimate goal would be to tune all of the model parameters in a single step (linear and nonlinear ones) using all of the available test data. The concept of *hybrid identification* refers to the hybridization between classical algorithms and specific techniques based on surrogate models intended to handle the aerodynamic nonlinearities. The RBF/LLM networks described in the previous sections are implemented for that purpose. They are particularly well suited for modeling complex and unstructured nonlinear systems, whether static or dynamic ones [11][47][52][55]. In the hybrid approach, they are typically used to replace the look-up tables describing the various coefficients, *e.g.* those appearing in equations similar to (1) [26]. This allows an algorithmically efficient identification to be performed and, additionally, this does not bias the results against *a priori* knowledge (*e.g.*, a predefined look-up index). This kind of implementation is grey-box type since it preserves the physical meaning as well as the structure of the aerodynamic developments used by industry [4][22].

The methods classically used (EE/OE/FE algorithms) have been implemented in a tool developed in cooperation between Airbus and ONERA (named *IdPy*, see Box 1), and this has been achieved through extensive algorithmic adaptations (Figure 7). Two complementary options are available for the identification: either time dependent coefficients are directly compared with their counterpart extracted from flight data throughout the sequence of tests available (EE), or they are integrated into the flight dynamic equations in order to minimize the errors between measured and simulated state variables (OE/FE). Both approaches benefit from the analytical and differentiable formulations of the surrogate models, which make it possible to perform quasi-exact parameter optimization (unlike purely numerical approaches using finite differences). Much CPU time is also saved for computing the derivative estimates required by the sensitivity equations, which is quite valuable especially for the greediest algorithms (OE/FE).

Whatever the approach, it is crucial for the aerodynamic corrections to remain physically acceptable. Practically, the purpose of an

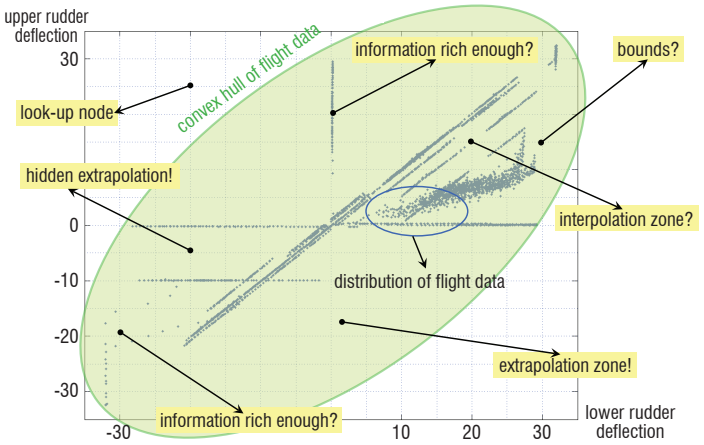

Figure 6 – Some traps and issues related to heterogeneous data distributions

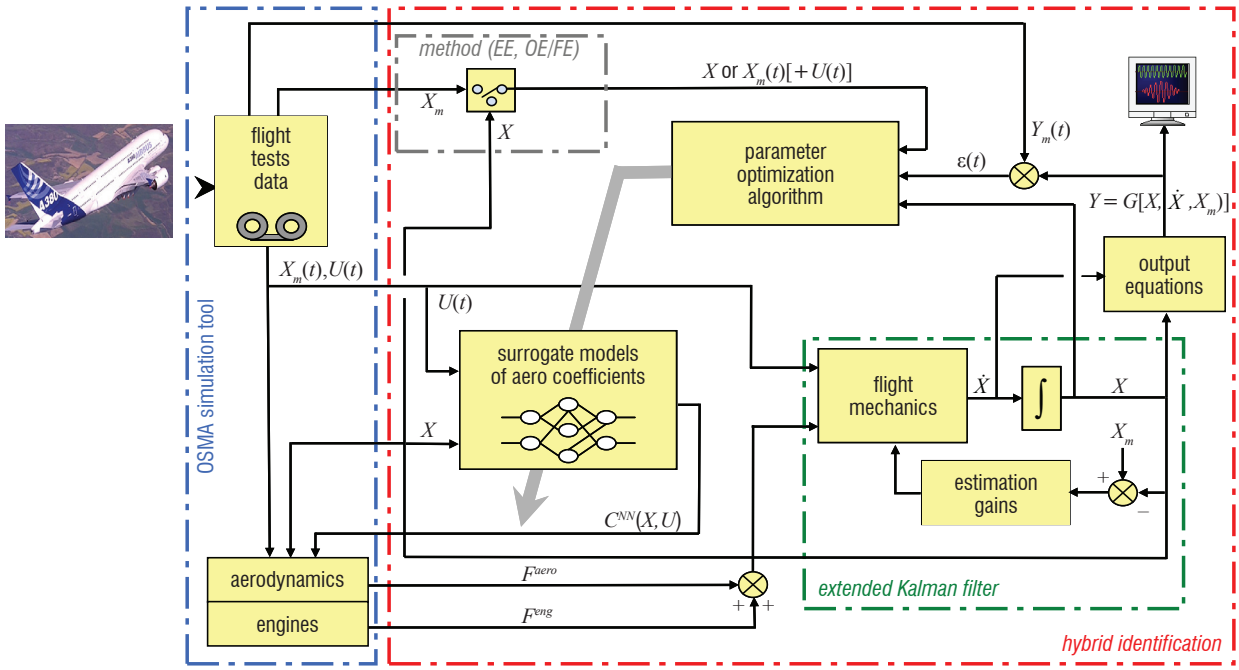

Figure 7 – Principles of the hybrid identification (X = simulated states, X_m = frozen states, Y = simulated outputs, Y_m = measured outputs, U = control deflections, F^{aero} = aerodynamic forces/moments, F^{eng} = engine forces/moments, G = measurement equations, C^{NN} = surrogate model of aerodynamic coefficients)

Box 1 - Identification of the A350 pitching moment in clean configuration

The following results have been achieved by using the tool *IdPy*, which stands for **Id**entification with **Py**thon, which is the most recent tool jointly developed by Airbus and ONERA. Compared to the previous tools, the novelty provided by *IdPy* comes from directly using OSMA, the Airbus simulation tool [4][22], thanks to a python interface that provides all of the functionalities required to match the simulation loop and the algorithm needs. In that way, the computations done by the identification tool are in line with the end user requirements. This common interface is a key point for spreading the results effectively to the operational context and for using other Airbus post-processing tools. *IdPy* follows an agile process: any new Airbus requirement to deal with a new issue can be implemented, tested by ONERA and validated by Airbus within a few weeks. *IdPy* is mostly generic since all Airbus models are operated through OSMA; hence, *IdPy* can be deployed very quickly for any aircraft, from the oldest A320 aircraft to the most recent A350 aerodynamic model. *IdPy* also offers new capabilities to operational end users with respect to the legacy Airbus tools: to run a complete identification process chaining EE, OE and FE minimizations within the same framework, and to process large flight data sets since the amount of flight test maneuvers has tripled during the last decade.

This user case corresponds to the pitching moment Cm, as expressed by (1). The 3 rigid coefficients to be estimated are the moment at zero lift $Cm_0 = f(M)$, the aerodynamic center position $x_{AC} = f(M)$ and the non-linear term $\Delta Cm_{NL} = f(\alpha, M)$. In the aerodynamic model, these coefficients are in the form of look-up tables; 15 parameters describe the linear coefficients and 868 parameters are required for the non-linear coefficient. For each coefficient, a surrogate model can be used as a grey-box to estimate an additional term w.r.t. the pre-flight model, following an EE approach to start with the identification process. Two look-up tables are set for the linear coefficients Cm_0 and x_{AC} with 13 parameters, and a LLM network is introduced for the non-linear coefficient ΔCm_{NL}. In the present case, the initialization of the network does not make use of a constructive approach, which would result in a more parsimonious model. This is a simple mapping of the explanatory variables done by hand, just considering the input space, the output, the flight domain achieved and the extrapolation objective. Figure B1-1 superimposes the initial kernel distribution with the flight test data in the input space of the non-linearity (AoA, Mach). The density of kernels is raised at high Mach number (green ellipsoids) to ensure enough flexibility during the updating process, whereas a few kernels are set up beyond the domain explored during the tests (blue ellipsoids) to facilitate a recovery of the pre-flight values (orange ellipsoids). Even with this straightforward parameterization, the number of parameters to be processed is 129, much lower than the 868 ones required by the original look-up table.

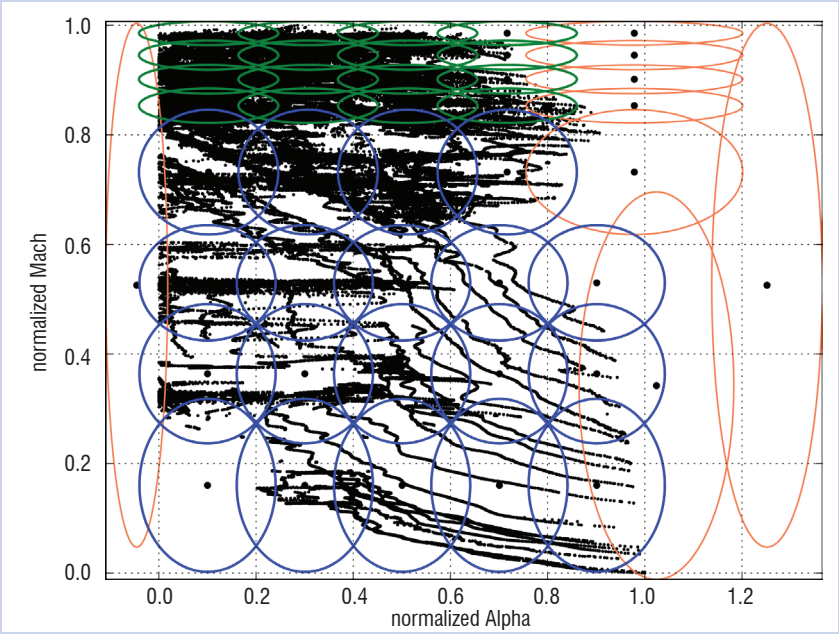

Figure B1-1 – Kernel and data distribution in the (α, M) plane

With regard to the flight test data, 374 flight test maneuvers have been selected for this identification step corresponding to over 2×10^5 data samples (about 6 flying hours). The identification is done in a single step for both linear and non-linear coefficients. Additive biases on the pitching moment are also estimated for each maneuver to account for shifts in the measurements. Several constraints are also added into the process, to get suitable updates and to regularize the optimization:

- to constrain the model output to zero in the linear part, assuming that the pre-flight linear domain was properly defined from CFD computations,
- to smooth the behavior of the surrogate models (look-up tables and LLM network),
- to extrapolate the estimation results in the boundary parts of the flight domain covered by the available data,
- to smoothly connect the updated and pre-flight models in the far-off regions.

The identification results obtained within about one hour of computation time, and involving 3 optimization iterations, are shown in Figure B1-2. The left part of the figure corresponds to the pre-flight "predicted" nonlinear coefficient ΔCm_{NL}, and the right part to the updated one.

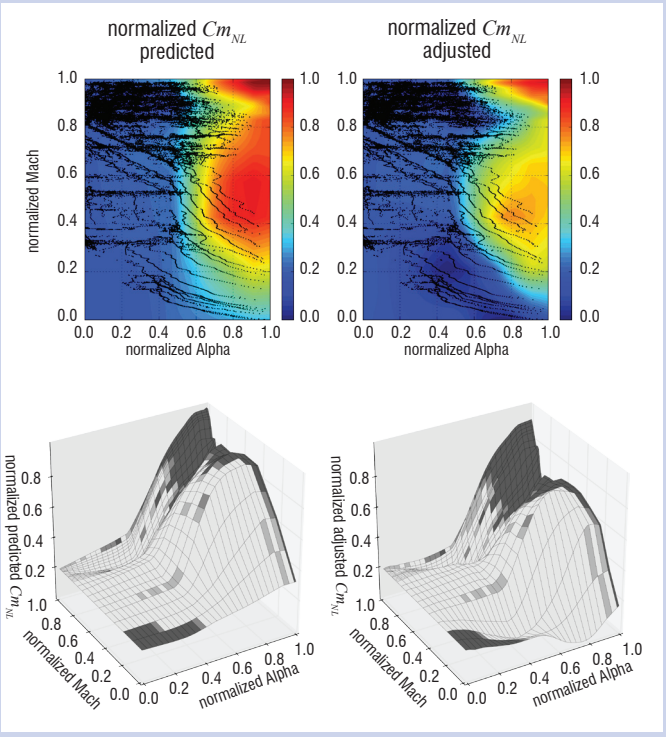

Figure B1-2 – Comparison of pre-flight and identified coefficients ΔCm_{NL}

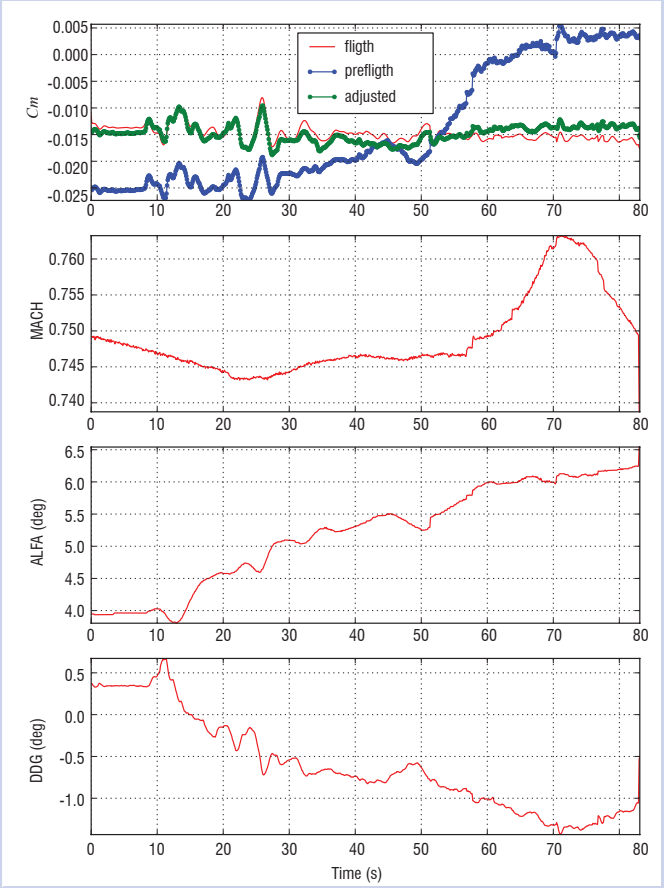

Figure B1-3 – Pitching moment during a turn maneuver with pitch-up at high Mach (from top to bottom: pitching moments, Mach, AoA, elevator deflection)

The heterogeneous data distribution is superimposed to the contour lines of the coefficient in the upper part of Figure B1-2, and is also displayed via a color code in the 3D plots (from white to black mesh depending on the proximity of the data samples). Finally, the plots of Figure B1-3 compare the pitching moment reconstructed from the flight data with the pre-flight and updated values, for one maneuver among the 374 flight tests. This user case also shows that the industrial requirement of coupling the identification tool with the Airbus simulator OSMA has been satisfied, and is not so detrimental to the industrial process since a large set of data can be managed within a reasonable time, which was not possible with the previous *standard* Airbus tools.

automated identification over large areas of the flight domain has raised a new need: specifying constraints to be followed by the non-linearities (*i.e.*, the outputs of the surrogate models). Constraints are a way to compensate for insufficient or sparse test data (Figure 6) or to introduce some kind of expertise into the problem. For instance, freezing the output levels may be required in some zones (*e.g.*, ΔCm_{NL} in (1) should remain zero at low AoA and low Mach, so that it does not interfere with the other terms of the development). It may also be desired to smooth the nonlinearities, or to connect identified and pre-flight models in areas where no flight data is available, and hence *a priori* knowledge from CFD or wind tunnel tests should be preserved [31]. These constraints can be enforced by robust regression mechanisms relying on various forms of penalties w.r.t. constrained values, smoothing, regularization, etc. This in turn raises the question of choosing and tuning these hyper-parameters, which should also be as automated as possible.

All of these issues, which are drawn from an identification stage of the A380 rudder efficiency, are illustrated by Figures 6-8. The coefficient represented in Figure 8 corresponds to a strongly nonlinear effect, contributing to the yawing moment gradient due to the rudders. For the A380 aircraft, this is a 3D coefficient depending on the sideslip angle and on the rudder deflections (lower and upper surfaces). Figure 6 corresponds to a cross-section view of the distribution of the available flight test data for a given value of the sideslip angle (*i.e.*, a thin slice of the 3D volume). The green ellipsoid is a 2D section of the convex hull of the flight data, and the background grid represents the look-up nodes of the pre-flight look-up table. To facilitate the industrial process, incremental updates of the aerodynamic data must be provided to be used by the OSMA simulator, and this requires the estimated corrections to be projected onto these look-up nodes. The issue highlighted by Figure 6 is that this projection should only be validated in some regions, due to the heterogeneous distribution of the input data in the domain.

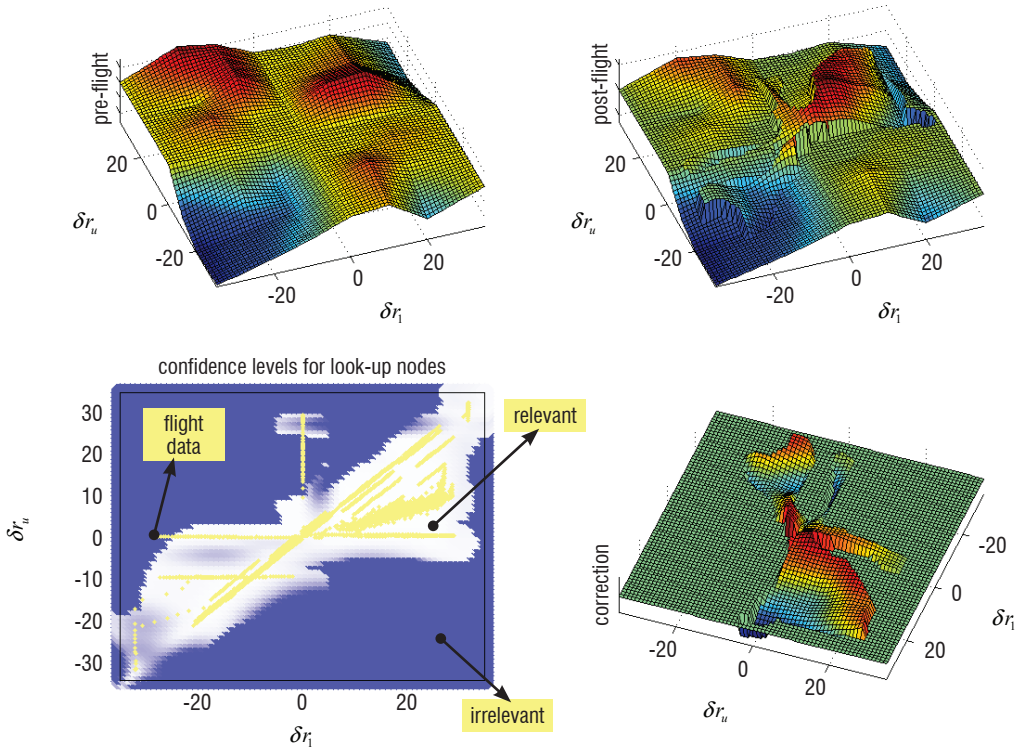

Figure 8 – Identification of the rudder efficiency on the yaw axis: sectional view of a 3D coefficient $C = f(\beta, \delta r_l, \delta r_u)$

To make this decision and to detect hidden extrapolation zones for instance, confidence levels can be associated with the identification results by computing useful indicators like the leverages (diagonal terms of the hat matrix [2][30]). Such confidence levels are displayed in the left bottom part of Figure 8, with a color code going from white to blue according to the local relevance of the results. Since this plot is only a sectional view of a more general 3D computation, the flight data displayed in yellow in this plane do not account on their own for these confidence levels. Finally, to smoothly connect the identified and pre-flight models, a robust regression procedure is used during a post-processing stage, involving a weighted LS optimization for which the penalties are inferred from the confidence levels, and an extended criterion is defined by introducing regularization constraints on the curvature of the outputs. Figure 8 illustrates the results achieved by this process with a surrogate LLM-type model (18 kernels) optimized using the *koala* tool described in the previous sections. From the right bottom plot of Figure 8, it can be seen that the pre-flight surface is reshaped by incremental updates only in the informative zones.

Application #3: embeddable models for virtual sensing

The development of electrical FCS, known as Fly-By-Wire, and the increasing level of automation have resulted in advanced capabilities for detecting, protecting and optimizing A/C flight guidance and control [50]. However, this higher level of automation requires the availability of some key flight parameters to be extended, particularly both aircraft AoA and calibrated airspeed, to keep the nominal flight control laws activated, thus avoiding any switch to degraded control modes (*e.g.*, *alternate* or *direct* laws as opposed to *normal* flight control laws for most of civil transport A/C). Hence, the monitoring and consolidation of those signals are significant issues, usually achieved via many functionally redundant sensors, allowing the way in which those flight parameters are measured to be directly enlarged (*hardware redundancy*). Unfortunately, this solution is detrimental to aircraft weight, power consumption and space requirements, and requires extra maintenance needs. Another alternative consists in benefiting from specific physical and dynamical relationships which link some A/C states. Advanced signal processing techniques can achieve real-time estimates of the critical flight parameters and yield dissimilar signals, by merging all or part of the available measurements through a model-based simulation of the aircraft flight mechanics (*analytical redundancy*). The resulting filtered and consolidated information is delivered under unfaulty conditions by estimating an extended state vector including wind components, measurements biases and modeling errors, and can replace failed signals under degraded conditions (*virtual probes*). To this aim, adaptive estimation schemes, which make use of integrated Fault Detection and Diagnosis (FDD) methods, can be developed in order to provide to the FCS with the ability to accommodate itself to potential sensor failures automatically.

Together with its industrial partners, ONERA has been working on this topic for a decade now and has exploited the potential and developed adaptive versions of several nonlinear state estimation techniques. This is the case of the well-proven Extended Kalman Filtering (EKF) [19][20][48], which permits basically all available sensor information to be merged while enhancing the measurements through a modeling that can describe the expected A/C behavior with more or less accuracy. The resulting estimates can be processed in parallel with the measured signals, thus contributing to extend the way the flight parameters are both monitored and consolidated. Several schemes allow that extra information to be introduced into the overall FDD architecture (see Figure 9): processing the virtual sensor just like the real ones, using it only in duplex or simplex modes (after one or several faults have already occurred), or using it to form an $(n+1)$-plex

sensing channel (*e.g.*, quadruplex in the case of 3 available sensors). The use of analytical redundancy for FCS is not new, as opposed to virtual sensors which have been studied more recently [36]. Practically, a major obstacle to an operational use of this kind of approach springs from their implementation in real time on embedded computers and especially from the computational burden of the associated algorithms. This issue is of primary importance when dealing with civil transport aircraft due to the certification purposes which impose stringent requirements to be fulfilled. That is why, to limit the complexity, surrogate models have been widely harnessed to derive simplified representations of the A/C aerodynamics (and possibly of the thrust forces). In addition, the selected analytical models, of an LLM type, offer other substantial benefits for encoding the EKF-based estimation algorithm, while remaining consistent with the implementation constraints related to the FCS for aviation industry.

Subsequently, this compliant adaptive estimation algorithm embeds a dedicated nonlinear state space representation, denoted as M_{NL} in the sequel, which describes the flight dynamics. This is derived from both kinematic relationships and longitudinal flight mechanics equations (see [19] for more details). In order to obtain an accurate and suitable model, the most relevant rigid-body aerodynamics, as well as static aeroelastic and possibly propulsion effects, are modeled using LLM networks. Such a dynamic modeling can be mathematically formulated as follows:

$$M_{NL}:\begin{cases}\dot{x}(t)=f\left(x(t),y_m(t)\right)+v(t)\\ y(t)=g_{LLM}\left(x(t),u(t),y_m(t)\right)+w(t)\end{cases} \tag{14}$$

In (14), $x(t)$ and $y(t)$ designate the state and output vectors of the nonlinear state space representation respectively; $u(t)$ refers to the input vector and gathers the deflection of all aircraft control surfaces (ailerons, spoilers, rudder, elevators and horizontal stabilizer); $y_m(t)$ (subscript m will refer to the measurements in what follows) is a vector of measurements, which includes also the time histories of several measured flight parameters, especially those of the longitudinal/lateral variables whose dynamics are not modeled in $\dot{x}(t)$. Subscript LLM indicates that the analytical expressions associated with the nonlinear output equations contained in function g_{LLM} exhibit the LLM approximations used to model at least the aerodynamic lift force, plus the total static mean gross thrust force and pitching moment, depending on the filter formulation. The process equations appearing in f correspond to a set of common kinematic relationships, augmented by slowly time-varying parameters (accounting for the accelerometer biases, the wind speed components and the modeling error in the lift force coefficient) whose dynamics are represented by random walks. Vectors $v(t)$ and $w(t)$ are process and observation Gaussian white noises. They are introduced to model errors in both the modeling M_{NL} and the measurements $y_m(t)$. It is assumed that they are characterized by zero-mean, uncorrelated and mutually independent processes with estimated covariance matrices denoted by Q and R respectively.

Based on previous nonlinear flight dynamics modeling introduced in (14), the proposed Adaptive EKF (AEKF) algorithm resumes, but also adapts, the main processing steps of the nonlinear Kalman filtering theory. Considering the standard filtering equation given in continuous-time (see [23]) with $p\in N^*$ measurements available every dt, and assuming that the correction term remains constant over $[t_k;t_{k+1}=t_k+dt]$, one can derive:

$$\dot{\hat{x}}\left(t_k|t_k\right)=f\left(\hat{x}\left(t_k|t_k\right),y_m\left(t_k\right)\right)+\left[\frac{K^i\left(t_{k+1}\right)}{dt}\times\begin{Bmatrix}z^i\left(t_{k+1}\right)\\ -g^i_{LLM}\begin{pmatrix}\hat{x}\left(t_{k+1}|t_k\right),\\ u\left(t_{k+1}\right),y_m\left(t_{k+1}\right)\end{pmatrix}\end{Bmatrix}\right]\begin{matrix}i\leftarrow 1\\ i\leftarrow 2\\ \vdots\\ i\leftarrow p\end{matrix} \tag{15}$$

with: $$\hat{x}\left(t_{k+1}|t_k\right)=\hat{x}\left(t_k|t_k\right)+\int_{t_k}^{t_{k+1}}f\left(x(t),y_m(t)\right)dt$$

and then: $$\hat{x}\left(t_{k+1}|t_{k+1}\right)=\hat{x}\left(t_k|t_k\right)+\int_{t_k}^{t_{k+1}}\dot{\hat{x}}\left(t_k|t_k\right)dt$$

In the differential equation (15), the gain matrix K is computed as in the discrete-time case. (15) also shows that the correction step is processed sequentially, measurement by measurement, which permits the computational complexity to be alleviated significantly. Indeed, this procedure avoids the matrix inversion traditionally required for the calculation of the correction gain matrix K, and replaces it by p scalar division(s). This sequential update is strictly equivalent to the standard computation of the EKF gain (which involves a matrix inversion each time new measurements become available) as both process and output noises are assumed to be uncorrelated (diagonal estimated covariance matrices Q,R). Then, given that sensor reliability can be periodically assessed by means of specific fault detection techniques (FDD is a component of the AEKF), the calculation of the K matrix in (15) can be adapted continuously over time, in case of detected and isolated faulty measurements, by simply omitting them

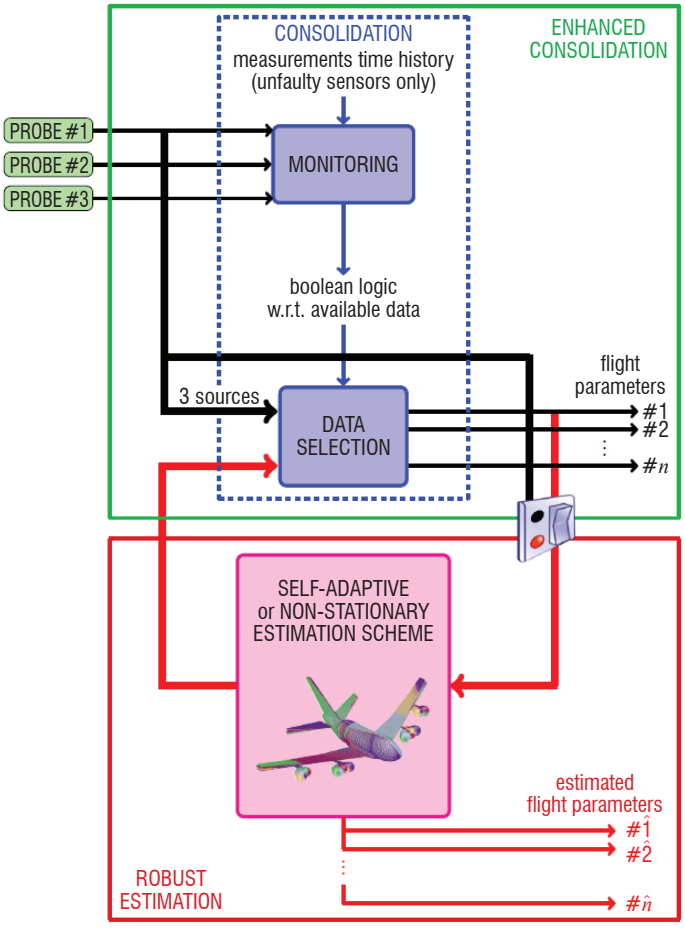

Figure 9 – Consolidation scheme using a virtual sensor

within the sequential update process. Usually, another difficulty for implementing such an estimation scheme on embedded computers arises from the double linearization that must be performed in real-time for both the system process (around $\hat{x}(t_k|t_k)$) and the output equations (around $\hat{x}(t_{k+1}|t_k)$). Resorting to numerical finite differences does not appear to be a viable solution w.r.t. the capabilities of current on-board computers since this implies multiple calls to the nonlinear flight dynamics modeling M_{NL}, as well as numerous multidimensional save/restore points. Fortunately, regarding the details of the equations which compose both nonlinear f and g functions, the linear tangent matrices $A = \partial f / \partial x$ and $C = \partial g_{LLM} / \partial x$ can be obtained analytically by means of:

- resorting to kinematic relationships to describe A/C dynamics (*i.e.*, in f). The resulting set of differential equations is known to be fully differentiable analytically w.r.t. the state vector $x(t)$ considered. Consequently, the analytical expressions of all A matrix coefficients can be established *a priori*, and their values can be calculated over time as the estimated state vector $\hat{x}(t_k|t_k)$ is continuously updated,

- using LLM networks which have the property of also being analytically differentiable w.r.t. their inputs (explanatory variables). Thus, given that the g_{LLM} function corresponds to a set of very simple static relations (based on standard trigonometric functions or vector norm calculations), except for the vertical load factor equation which relies on prior knowledge involving a LLM of the lift force coefficient, the nonlinear observation equations can again be differentiated once, in an analytical way, in order to derive the C matrix. The latter will be refreshed as the predicted state vector $\hat{x}(t_{k+1}|t_k)$ is periodically calculated.

These remarks make possible an off-line derivation of all of the analytical expressions associated with the terms composing both A and C matrices. Once all of these mathematical expressions have been determined, their valuation is carried out naturally over time through a single call to the internal model of the estimator. Besides, in most aircraft applications, these matrices will also appear to be sparse, thereby reducing the numerical complexity of the proposed approach slightly more. Deriving the Jacobian matrix $C = \partial g_{LLM} / \partial x$ is not as straightforward as for A. Indeed, the nonlinear output equations used for the prediction step of the AEKF require prior knowledge of the A/C aerodynamics. However, the latter can be limited to the aircraft lift force coefficient, which is generally represented by a complex nonlinear modeling in order to be accurate enough. Although surrogate models offer an efficient and powerful solution for representing such complex physics, there is also another benefit of using such analytical modeling, related to differential calculation. Let us consider the general LLM representation with $N_i \in N^*$ explanatory variables, $N_o \in N^*$ outputs and $N_c \in N^*$ kernels:

$$\forall k \in [1, N_o] \qquad y_k(e) = \sum_{i=1}^{N_c} \left(\sum_{j=0}^{N_i} w_{ij} e_j \Big/ \left(1 + \sum_{k=1}^{N_i} \frac{(e_k - c_{ik})^2}{\sigma_{ik}^2} \right) \right) \qquad (16)$$

and: $y = [y_1 \;\; y_2 \;\; \dots \;\; y_{N_o}]^T$

In (16), the set of parameters $(w_{ij}, c_{ik}, \sigma_{ik})$ must be optimized so that the output vector y match some reference data for given input vector values $e = [e_1 \;\; e_2 \;\; \dots \;\; e_{N_i}]^T$. Given that the exponential operator is not available in most dedicated versions of the SCADE© formalism used in the aviation industry, the LLM kernels adopted in this application are based on the Pade approximant $\exp_{[0,l]}$ (leading to a set of rational function(s)). This illustrates that the computational complexity associated with any modeling involved in new potential solutions can be adjusted to limited encoding capabilities, despite their algorithmic sophistication. Subsequently, it is noteworthy that the partial derivative of any given LLM output $y_k, k \in [1, N_o]$ with respect to the vector e of explanatory variables, the so-called sensitivity, can be easily computed from (16) s.t.:

$$\forall k \in [1, N_o] \qquad \frac{\partial y_k}{\partial e} = \sum_{i=1}^{N_c} \frac{\partial}{\partial e} \left(\frac{\sum_{j=0}^{N_i} w_{ij} e_j}{\left(1 + \sum_{k=1}^{N_i} \frac{(e_k - c_{ik})^2}{\sigma_{ik}^2} \right)} \right) \in M_{1 \times N_i}(R) \qquad (17)$$

It is of primary interest to note that the differentiation of the Pade approximant in (17) leads to an analytical expression which again depends on parameterized rational functions. Therefore, the calculation of $C = \partial g_{\mathrm{LLM}} / \partial x$, which combines standard differentiation developments (analytically known thanks to flight mechanics equations) and LLM sensitivities as in (17), can also be carried out analytically as for $A = \partial f / \partial x$. Consequently, this adaptive EKF-based estimation algorithm can be implemented on-board with a low complexity, and used to reconstruct any faulty information so as to maintain nominal flight control laws.

As was aforementioned, the adaptation of the nonlinear state estimation is managed by a dedicated FDD technique since the EKF solely becomes unsuitable on its own in faulty cases, given that it assumes noisy, but healthy, measured signals. This FDD technique can take on several forms and can result from any signal- or model-based approach provided that it ensures both fault detection and isolation capabilities. To say a word about the detection and adaptive part of the method, a candidate EKF reconfiguration method has been experienced in [20] to monitor the measured signals from sensors and reduce the estimation errors induced by those potential incorrect measurements to recover acceptable performances. This method accomplishes both detection and isolation of specific single or multiple abrupt faults (constant bias, stuck value, strong drift, etc.) before adjusting the sequential measurement update process of the EKF.

The FDD relies on Fisher-Snedecor statistical hypothesis tests, recursively processed in order to periodically assess the reliability of the sensors. In the case of invalid test results, this pure signal-based technique will instantaneously detect and isolate unexpected sensor faults characterized by High-Frequency (HF) signatures beyond the A/C dynamics bandwidth. All declared and identified faulty measurements will then be denied for data fusion, thus achieving fault-free estimates of the key flight parameters, namely the AoA and the Calibrated AirSpeed (CAS). In a few words (see [46] for more details), these stages are accomplished using several signal processing operations which consisting, firstly, in extracting the HF components from the considered time-varying measurements and, secondly, in applying an appropriate inverse Auto-Regressive whitening filter. Then, a statistical analysis of the resulting HF residues compares a given reference variance (characteristic of a healthy signal) with a time-varying one (calculated over a sliding window), and permits the AoA and CAS states to be determined (healthy or faulty measurements).

The AEKF methodology was evaluated with data drawn from real flight tests performed on a civil transport aircraft (A340-600). These data correspond to an operational flight path profile, comprising

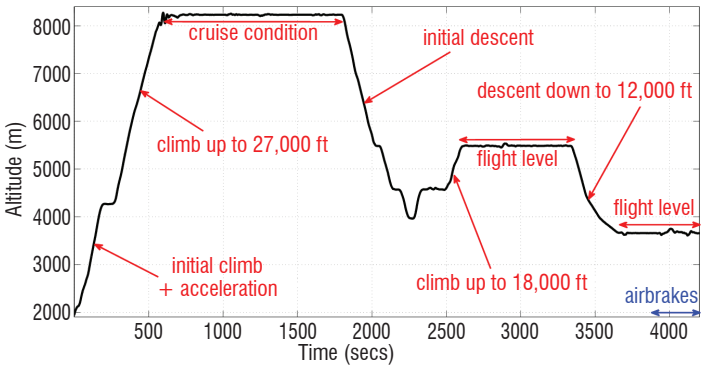

Figure 10 – Flight profile of the real test data

climb, cruise and descent flight conditions with heading changes (see Figure 10) which span more than 1 hour. The results gathered after processing these real flight data are presented in Figure 11. The simulated additive AoA and CAS faults (constant biases) are both applied at time t = 20 sec, and are quickly detected and isolated by the FDD mechanism used for sensor monitoring (detection time less than 1 sec). The results show that the mixed predicted/estimated flight parameters reconstructed afterwards remain valid over a long time horizon (more than 1 hour) under realistic changing flight conditions after the sensor reliability is affected by a double failure. With respect to these real data, the residual errors associated with the parameters to be reconstructed reach maximum peak values equal to 1° for AoA and 15 knots for CAS in case of a double fault. It is worth noting that these results also prove the robustness of the developed approach w.r.t. the modeling errors. Actually, the internal model used by the estimator corresponds to a simplified one resulting from a LLM representation of only a subset of the aerodynamic coefficients. Moreover, several effects (*e.g.*, the lateral ones) were ignored (ailerons) or simply approximated (spoilers). Besides, the thrust dynamics appear also roughly modeled, resulting in errors on forces/moments varying from 50% to 100% w.r.t. the theoretical ones. Hence, the internal model approximates the best reference model, which itself only approximates the real (unknown) aircraft.

Conclusion and prospects

This paper highlights the potential value of introducing surrogate models in some stages of the development of FCS for aircraft. The special types of neural networks, which have been selected for their attractive properties, were firstly described, as well as the constructive procedure implemented to optimize their internal parameters. Then, three specific applications related to modeling, identification and control aspects were considered and illustrated by processing real aircraft data and real aerodynamic models. These applications extend from the exploitation of the first flight tests of a new airplane to in-service monitoring of the flight parameters, through the intermediate steps required for designing the control laws. To complete this picture, it is worth noting that other stages of aircraft development are likely to benefit from surrogate-based approaches. For instance, at both ends of the process, the Multidisciplinary Design Optimization (MDO) during the preliminary design of a new airplane, and the clearance of the flight control laws involving worst-case analysis prior to aircraft certification, are two candidates for which surrogate models are commonly investigated. These applications, although slightly different, involve a costly multiobjective optimization process, and hence share the need to limit the number of high-fidelity computations by means of low-fidelity models, which can replace a significant proportion of the computationally demanding simulations.

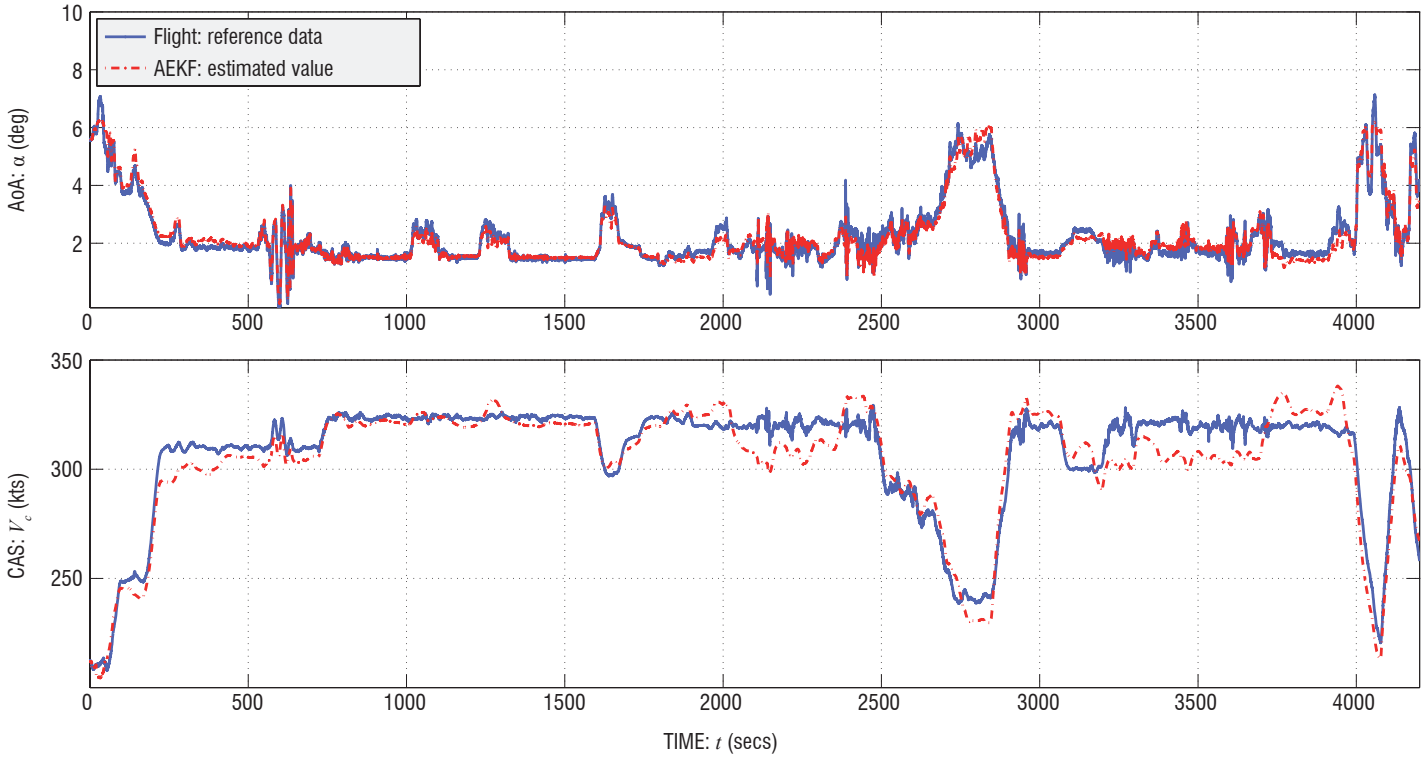

Figure 11 – AEKF results obtained by processing the flight data and mimicking a complete loss of both AoA and airspeed information at t = 20 sec

Concerning the first application topic (building efficient LFR for control analysis and design), room for improvement still exists and would be based on a better sparsity of the surrogate-based indirect approach by pruning useless explanatory variables in the final rational expressions. This would contribute to enlarging the practical user cases for which rational approximants are superior to polynomial ones and can provide smaller sized LFR. Though basically suited to off-line modeling applications, it should be noted that the *koala* tool can also be used for on-line implementation of LFT controllers. Actually, the usual way of computing scheduled LFT gains, for instance, is not relevant for embedded controllers because its computational burden is generally far too high. Rather than pre-computing tabulated values of the control gains, which can be a tricky option for high dimensional functions, very accurate and parsimonious models can be achieved thanks to the APRICOT library. The surrogate model will fully benefit the practical implementation since the (factorized) rational expressions are consistent with the stringent coding requirements for aircraft computers (see [15] for example).

The second application topic (identification of aerodynamic non linearities) illustrates that the transfer of new tools to industry is quite a long term process. The end users need to gain confidence in the results provided by new tools and also need time to appropriate these tools and their functionalities. In the long run, surrogate models might not only be intermediate models introduced into the identification process to facilitate the optimization stages, but may also represent a new way of representing aerodynamic nonlinearities in a wider part of the industrial process. This would avoid the present upstream and downstream conversions: from look-up tables to neural networks, and then backward reprojection of the updated results in the tables used *in fine* by the simulation environment. Regarding prospects, the development of multicriteria optimization is a very promising topic for the identification process. As mentioned in the section devoted to the second application, this process is still tedious today, involving a series of semi-manual steps which are time-consuming and would benefit from being more automated. Some of these steps are typically based on dealing with various criteria: tracking the time domain histories of the measured flight parameters by means of both EE and OE fitting criteria, considering also the values of modal frequencies and dampings in the frequency domain, and offering a compromise between the raw solution of the optimization problem and the discrepancies w.r.t. *a priori* knowledge encoded in the pre-flight data. Multiobjective optimization is a way of coping with these criteria of various types, and also of providing the end users not only with a unique "optimal" solution, but with a set of solutions (the so-called Pareto surface), some of which are likely to offer more satisfactory trade-offs for those conflicting goals. As the whole process should be achieved over large areas of the flight domain and should include all of the aerodynamic nonlinearities, surrogate modeling should be an integral part of this multicriteria identification, which is a challenging topic for future developments.

The third application topic (embeddable models for virtual sensing) is a good example of the aforementioned remark: the industrial process could benefit from a common way of representing aerodynamic nonlinearities by means of surrogate models. If this were the case, the updated models resulting from the identification stage could be directly used as internal models in the estimation scheme. Besides, it is worth noting that the same simplified aircraft surrogate modeling can be shared by the AEKF and by any model-based FDD techniques (as for instance the one used in [45]). In addition to being useful for civil aircraft, this virtual sensing strategy can also be beneficial for under-equipped vehicles (UAV, small airplanes). In all cases, the grey-box representation selected is essential for preserving only the most relevant aerodynamic terms in the embedded surrogate model, and for proceeding to a sensitivity analysis as regards the performances of the estimator. Another possibility offered by this modular and grey-box architecture, combined with the local properties of the RBF/LLM-type networks, would be to adjust the internal model of the estimator in a local and incremental way. Such an adjustment could be achieved in terms of the real performances under operational flight conditions, with standard flight profiles. Hence, the model would be finally fitted to the behavior of the estimator, resulting in a surrogate model that may be slightly different from a pure knowledge-based model. Finally, on-going works related to this topic are aimed at directly dealing with the anemometric measurements available on-board (pressures and temperature) instead of the CAS information, in order to improve the detection capability of the combined FDD and estimation schemes ■

References

[1] R. BABUSKA, H. VERBRUGGEN - *Neuro-Fuzzy Methods for Nonlinear System Identification*. Annual Reviews in Control, Vol. 27, p. 73-85, 2003.

[2] D. A. BELSLEY, E. KUH, R. E. WELSCH - *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*. John Wiley & sons, New York, 1980.

[3] T. M. BLACKWELL, P. J. BENTLEY - *Dynamic Search With Charged Swarms*. GECCO'02, New York, USA, July 2002.

[4] C. BOUCHEREAU, D. LIOT - *Aerodynamic Specification for A340/A330/A321 Flight Training Simulators*. Airbus TR 456.006/90, Issue 8, April 1993.

[5] A. BUCHARLES *et al.* - *An Overview of Relevant Issues for Aircraft Model Identification*. AerospaceLab, Issue 4, http://www.aerospacelab-journal.org/al4, 2012.

[6] O. S. CELIS, A. CUYT, B. VERDONK - *Rational Approximation of Vertical Segments*. Numerical Algorithms, 45, p. 375-388, 2007.

[7] S. CHEN, S. A. BILLINGS, C. F. N. COWAN, P. M. GRANT - *Nonlinear System Identification Using Radial Basis Functions*. International Journal of Systems Science, 21, p. 2513-2539, 1990.

[8] S. CHEN, S. A. BILLINGS - *Neural Networks for Nonlinear Dynamic System Modelling and Identification*. International Journal of Control, 56, p. 319-346, 1992.

[9] S. CHEN, X. HONG, C. J. HARRIS, P. M. SHARKEY - *Sparse Modelling Using Orthogonal Forward Regression with PRESS Statistic and Regularization*. IEEE Transactions on Systems, Man and Cybernetics, Part B, 34 (2), p. 898-911, 2004.

[10] S. CHEN, X. HONG, B. L. LUK, C. J. HARRIS - *Nonlinear System Identification Using Particle Swarm Optimization Tuned Radial Basis Function Models*. International Journal of Bio-Inspired Computation, 1 (4), p. 246-258, 2009.

[11] J. CHO, J. LAN, G. K. THAMPI, J. C. PRINCIPE, M. A. MOTTER - *Identification of Aircraft Dynamics Using a SOM and Local Linear Models*. 45th Midwest Symposium on Circuits and Systems, Tulsa, USA, August 2002.

[12] M. CLERC - *Particle Swarm Optimization*, ISTE, London, 2006.

[13] M. CLERC - *Why Does it Work ?* International Journal of Computational Intelligence Research, 4 (2), p. 79-91, 2008.

[14] G. DREYFUS *et al.* - *Neural Networks*. Methodology and applications, Springer, 2005.

[15] G. FERRERES, G. HARDIER, C. SEREN - *Adaptive Control of a Civil Aircraft Through On-Line Parameter Estimation*. SysTol'16, Barcelona, Spain, September 2016.

[16] G. GOLUB, V. PEREYRA - *Separable Nonlinear Least Squares: the Variable Projection Method and its Applications*. Inverse Problems, 19 (2), p. R1-R26, April 2003.

[17] G. HARDIER - *Recurrent RBF Networks for Suspension System Modeling and Wear Diagnosis of a Damper.* IEEE World Congress on Computational Intelligence, Vol. 3, Anchorage, USA, May 1998.

[18] G. HARDIER, C. ROOS, C. SEREN - *Creating Sparse Rational Approximations for LFR Modeling Using Genetic Programming.* 3rd IFAC Intal Conf on Intelligent Control and Automation Science, Chengdu, China, 2013.

[19] G. HARDIER, C. SEREN, P. EZERZERE - *On-Line Estimation of Longitudinal Flight Parameters*. SAE AeroTech Congress and Exhibition, Toulouse, October 2011.

[20] G. HARDIER, C. SEREN, P. EZERZERE - *Model-Based Techniques for Virtual Sensing of Longitudinal Flight Parameters*. International Journal of Applied Mathematics and Computer Science, 25(1), March 2015.

[21] S. HAYKIN - *Neural Networks: a Comprehensive Foundation*. IEEE Press, MacMillan Ed., New York, 1994.

[22] S. HUTASSE, T. DUCHAMP, B. BARRIETY - *Model Specification AER A350*. Airbus TR V00EDD081441, issue 19.0, March 2016.

[23] R. V. JATEGAONKAR - *Flight Vehicle System Identification - A Time Domain Methodology.* AIAA Progress in Astronautics and Aeronautics, Frank K. Lu Ed., Vol. 216, Reston, 2006.

[24] J. LANE, A. P. ENGELBRECHT, J. GAIN - *Particle Swarm Optimization with Spatially Meaningful Neighbours*. IEEE Swarm Intelligence Symposium, St Louis, USA, September 2008.

[25] D. LIOT, A. BUCHARLES - *Outils d'identification de la mécanique du vol latérale des Airbus*. Symposium RTO System Concepts and Integration Panel, Madrid, 1998.

[26] C. F. LO, J. L. ZHAO, R. DELOACH - *Application of Neural Networks to Wind Tunnel Data Response Surface Methods*. 21st AIAA Aerodynamic Measurement Technology and Ground Testing Conference, Denver, USA, June 2000.

[27] J. F. MAGNI - *Linear Fractional Representation Toolbox for Use with Matlab*. Available at http://w3.onera.fr/smac/lfrt with the SMAC Toolbox, 2006.

[28] R. MENDES, J. KENNEDY - *The Fully Informed Particle Swarm: Simpler, maybe Better.* IEEE Transactions on Evolutionary Computation, 8 (3), June 2004.

[29] G. MONARI, G. DREYFUS - *Withdrawing an Example from the Training Set : an Analytic Estimation of its Effect on a Nonlinear Parametrised Model*. Neurocomputing, Vol. 35, p. 195-201, 2000.

[30] G. MONARI, G. DREYFUS - *Local Overfitting Control via Leverages*. Neural Computation, Vol. 14, p. 1481-1506, 2002.

[31] E. A. MORELLI, D. G. WARD - *Automated Simulation Updates Based on Flight Data*. AIAA AFM, Hilton Head, USA, August 2007.

[32] E. A. MORELLI, R. DeLoach - *Wind Tunnel Database Development Using Modern Experiment Design and Multivariate Orthogonal Functions*. 41st AIAA Aerospace Sciences Meeting and Exhibit, Reno, USA, 2003.

[33] E. A. MORELLI - *Global Nonlinear Aerodynamic Modeling Using Multivariate Orthogonal Functions*. Journal of Aircraft, 32(2), 1995.

[34] O. NELLES, R. ISERMANN - *Basis Function Networks for Interpolation of Local Linear Models.* 35th IEEE Conference on Decision and Control, 1, Kobe, Japan, December 1996.

[35] O. OLORUNDA, A. P. ENGELBRECHT - *Measuring Exploration/Exploitation in Particle Swarms Using Swarm Diversity*. IEEE Congress on Evolutionary Computation, Hong Kong, China, June 2008.

[36] M. OOSTEROM, R. BABUSKA - *Virtual Sensor for Fault Detection and Isolation in Flight Control Systems*. Fuzzy Modeling Approach. Proc. of the 39th IEEE Conference on Decision and Control, Sydney, December 2000.

[37] M. J. L. ORR - *Introduction to Radial Basis Function Networks*. Technical Report of the Center for Cognitive Science, Edimburgh University, Scotland, April 1996.

[38] M. J. L. ORR, J. HALLAM, A. MURRAY, T. LEONARD - *Combining Regression Trees and Radial Basis Function Networks*. International Journal of Neural Systems, 10 (6), December 2000.

[39] M. J. L. ORR - *Local Smoothing of Radial Basis Function Networks*. Intal Symp. on Artificial Neural Networks, Hsinchu, Taiwan, 1995.

[40] J. R. RAOL, R. V. JATEGAONKAR - *Artificial Neural Networks for Aerodynamic Modeling*. TR IB 111-94/41, DLR Braunschweig, Germany, October 1994.

[41] C. ROOS - *Optimization Based Clearance of Flight Control Laws*. In Varga-Hansson-Puyou, Generation of LFRs for a flexible aircraft model, §4. Lecture Notes in Control and Information Sciences, Springer-Verlag, 2011.

[42] C. ROOS - *Systems Modeling, Analysis and Control (SMAC) Toolbox: an Insight into the Robustness Analysis Library*. IEEE Multiconference on Systems and Control, Hyderabad, India, 2013.

[43] C. ROOS, G. HARDIER, C. DOLL - *A Comparison of Techniques to get Sparse Rational Approximations for Linear Fractional Representations.* 29th Congress of the International Council of the Aeronautical Sciences, St Petersburg, Russia, September 2014.

[44] C. ROOS, G. HARDIER, J. M. BIANNIC - *Polynomial and Rational Approximation with the APRICOT Library of the SMAC Toolbox*. IEEE Conference on Control Applications, IEEE Multi-conference on Systems and Control, Antibes, France, October 2014.

[45] S. SAMAR, D. GORINEVSKY, S. BOYD - *Embedded Estimation of Fault Parameters in an Unmanned Aerial Vehicle*. IEEE Conference on Control Applications, Munich, Germany, October 2006.

[46] P. A. SAMARA, G. N. FOUSKITAKIS, J. S. SAKELLARIOU, S. D. FASSOIS - *A Statistical Method for the Detection of Sensor Abrupt Faults in Aircraft Control Systems*. IEEE Transactions on Control Systems Technology, 16(4), p. 789-798, 2008.

[47] S. SEHER-WEISS - *Identification of Nonlinear Aerodynamic Derivatives Using Classical and Extended Local Model Networks.* AST 15(1):33-44, February 2011.

[48] C. SEREN, G. HARDIER - *Adaptive Extended Kalman Filtering for Virtual Sensing of Longitudinal Flight Parameters*. 2nd IEEE International Conference on Control and Fault Tolerant Systems, Nice, France, October 2013.

[49] J. SJÖBERG, H. HJALMARSSON, L. LJUNG - *Neural Networks in System Identification*. IFAC SYSID, Copenhagen, July 1994.

[50] P. TRAVERSE, I. LACAZE, J. SOUYRIS - *Airbus Fly-by-Wire: a Total Approach to Dependability*. 18th IFIP World Computer Congress, p. 191-212, Toulouse, 2004.

[51] K. TROJANOWSKI - *Multi-Swarm that Learns, Intelligent Information Systems*. Vol. XVI, p. 121-130, 2008.

[52] C. C. DE VISSER - *Global Nonlinear Model Identification with Multivariate Splines – Application to Aerodynamic Model Identification of the Cessna Citation II*. Ph.D. Thesis, TU Delft, 2011.

[53] X. X. WANG, S. CHEN, C. J. HARRIS - *Using the Correlation Criterion to Position and Shape RBF Units for Incremental Modelling*. International Journal of Automation and Computing, 3 (4), p. 392-403, 2006.

[54] D. WEDGE, D. INGRAM, D. McLEAN, C. MINGHAM, Z. BANDAR - *On Global-Local Artificial Neural Networks for Function Approximation*. IEEE Transactions on Neural Networks, 17 (4), p. 942-952, July 2006.

[55] E. DE WEERDT, Q. P. CHU, J. A. MULDER - *Neural Network Aerodynamic Model Identification for Aerospace Reconfiguration.* AIAA GNC, San Francisco, August 2005.

[56] K. ZHOU, J. C. DOYLE, K. GLOVER - *Robust and Optimal Control*. Prentice-Hall, Upper Saddle River, 1996.

Acronyms

A/C	(Aircraft)
AEKF	(Adaptive Extended Kalman Filtering)
AoA	(Angle of Attack)
CAS	(Calibrated AirSpeed)
CFD	(Computational Fluid Dynamics)
EE	(Equation Error)
FCS	(Flight Control System)
FDD	(Fault Detection and Diagnosis)
FE	(Filter Error)
GA	(Genetic Algorithm)
HF	(High-Frequency)
LFR	(Linear Fractional Representation)
LLM	(Local Linear Model)
LOO	(Leave-One-Out)
LP	(Linear-in-their-Parameters)
LS	(Least Squares)
MLP	(Multi-Layered Perceptron)
NN	(Neural Network)
OE	(Output Error)
OLS	(Orthogonal Least Squares)
OSMA	(Outil de Simulation des Mouvements d'un Avion)
PRESS	(Predicted REsidual Sum of Squares)
PSO	(Particle Swarm Optimization)
RBF	(Radial Basis Function network)
SMAC	(Systems Modeling, Analysis and Control toolbox)
SNLS	(Separable Nonlinear Least Squares)

Jean-Marc Biannic graduated from SUPAERO in 1992, and received his PhD degree in Robust Control Theory. He joined ONERA as a research scientist in 1997, and has been accredited as a PhD supervisor since 2010. He is the author or co-author of 20 journal papers, more than 50 conference papers, many book chapters, teaching documents, a tutorial book on multivariable control, and Matlab toolboxes. He has participated in several European projects and Garteur Groups (on PIO and nonlinear control). From 2012 to 2016, he led a research project involving 10 research scientists for the development of the SMAC toolbox for Systems Modeling, Analysis and Control.

Georges Hardier holds a PhD in Automatic Control from SUPAERO. After joining ONERA, he was in charge of the development of autopilots and control laws for a series of French warships for two decades. He now has over 30 years of experience in parametric estimation, modeling and identification techniques, applied to the aeronautics industry. Recently, he implemented on-line estimation methods for FDD/FTC during the European project RECONFIGURE. He is also interested in surrogate modeling and evolutionary techniques for optimization, for which he has developed a series of algorithms and tools (*e.g.*, the APRICOT library of the SMAC toolbox).

Clément Roos graduated from SUPAERO in 2004 and holds a PhD in Automatic Control. He joined ONERA as a research scientist in 2007. He often takes part in industrial projects with Airbus and Dassault, and was notably involved in the European projects GARTEUR-AG17 and COFCLUO. His research interests focus on aircraft modeling, robustness analysis and control laws validation, as well as nonlinear design based on robustified dynamic inversion schemes and anti-windup synthesis. He is the author or co-author of several papers, book chapters, teaching documents and Matlab toolboxes.

Cédric Seren is a research scientist at ONERA. He graduated in 2003 from SUPAERO and received his Ph.D. degree in 2007. During his Ph.D., he worked on A/C flight tests protocol optimization for flight dynamics identification, using new evolutionary algorithms. Since 2007, his activity has been essentially focused on both nonlinear modeling and estimation for aircraft fault detection, isolation and recovery, as well as on mathematical optimization. Recently, he has implemented adaptive estimation techniques robust to sensors faults during the EU project RECONFIGURE.

Laurent Verdier, graduated from ENSICA in 1999 before joining Airbus in 2000 in the Flight Dynamics Simulation department. He acts as a technical competence leader for free-air aerodynamic identification and is the focal point for R&T activities. For 16 years, he has been involved in the flight dynamics identification of several Airbus programs (A340-600, A318, A380, Single Aisle Sharklet, A350) from the design of the flight test maneuvers to the validation of the final simulation tool. His research interests focus on flight test protocol optimization, identification methods, and flight test measurement processing, and he has also co-supervised several PhD works.