Application of Response Surface Methodology to Stiffened Panel Optimization.

Antoine Merval*
SOGETI High Tech and ONERA-DTIM, Toulouse, France

Manuel Samuelides†
ENSAE & ONERA-DTIM, Toulouse, France

and

Stéphane Grihon‡
Airbus France, Toulouse, France

In a multilevel optimization frame, the use of surrogate models to approximate optimization constraints allows great time saving. Among available metamodelling techniques we chose to use Neural Networks to perform regression of static mechanical criteria, namely buckling and collapse reserve factors of a stiffened panel, which are constraints of our subsystem optimization problem. Due to the highly non linear behaviour of these functions with respect to loading and design variables, we encountered some difficulties to obtain an approximation of sufficient quality on the whole design space. In particular, variations of the approximated function can be very different according to the value of loading variables. We show how a prior knowledge of the influence of the variables allows us to build an efficient Mixture of Expert model, leading to a good approximation of constraints. Optimization benchmark processes are computed to measure time saving, effects on optimum feasibility and objective value due to the use of the surrogate models as constraints. Finally we see that, while efficient, this mixture of expert model could be still improved by some additional learning techniques.

I. Position of the problem in the multilevel optimization chain

Aeronautical structures are mainly made of stiffened panels, i.e. thin shells (also called skin) enforced with stiffeners (respectively called frames and stringers) in both orbital and longitudinal directions (see figure 1). For the sake of the study the whole structure is divided into elementary parts called Super Stiffeners, consisting in the theoretically union of a stringer and two half panels. These basic structures are subject to highly non linear phenomena such as buckling, collapse and damage tolerance.

In order to determine the optimal size of these super stiffeners, static mechanical criteria must be computed using dedicated software that is based on non linear calculation. Thus, the analysis and the dimension estimation of the whole structure is currently computed by running a two-level study: at a global level, a Finite Element (FE) analysis run on the whole FE model provides internal loads – applied to each S-Stiffener; at a local level, these loads are used to compute static mechanical criteria. Most of these criteria are formulated using Reserve Factors (*RF*): a structure is validated provided all its *RF*s are greater than 1.

So, detailed design of a aircraft fuselage requires a two-level loop: first, numerous local optimizations are run on isolated super stiffeners in order to size them respecting mechanical criteria, depending on current internal loads. But changes in local geometry from initial to optimal design involve a new load distribution in the whole structure; an update step must then be performed to take these changes into account. This bilevel optimization process is then repeated until convergence of load distribution in the whole structure. Nowadays, this loose coupling process depicted in figure 2 is practically implemented in real industrial case. Other multi level optimization strategies can be found in Ref 1, 2.

---

* PhD Student, SOGETI/ Ecole Nationale de l'Aéronautique et de l'Espace, Student Member AIAA
† Professor of Mathematics, Ecole Nationale de l'Aéronautique et de l'Espace
‡ Engineer, ESAN Department, Airbus France

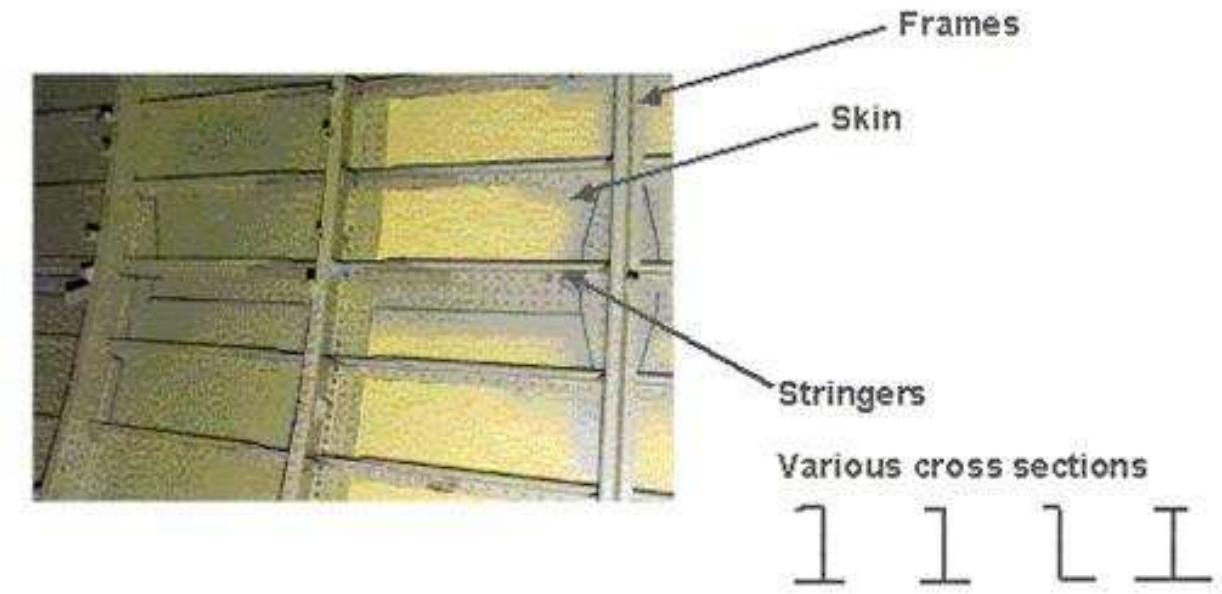

Figure 1: View of the stiffened panel structure of an airplane.

Local optimizations are computed using local methods that require gradients of the constraint functions - which can only be obtained by finite differences. Values of mechanical stability constraints are computed using dedicated software, through analytical calculation. A call to this software takes up to a second; as a consequence the need for finite difference calculations in each numerous local optimization greatly increases the time between two update steps.

Therefore, the dimension estimation step in an aircraft development program is an repetitive, time-consuming process. Much time could be saved by using a reduced model instead of performing straightforward computing[3,4]. This time saving is the aim of the present study, and is of great importance to engineers working in Airbus Structural Analysis Framework.

In this study, we focus on the metamodelling of static instability phenomena which are the buckling and the collapse of a super stiffener. We present here results of two different application cases: the first case deals with a super stiffener made of composite material, whereas the second case deals with a metallic one.

Using a surrogate model, which is based on computer experiments, requires a building strategy of a response surface. First, we have to select a family of surrogate models. The classic quadratic polynomial response surfaces are not suited to imitate the highly non linear type of functions. Artificial Neural Networks (ANN) seemed to be the best suiting model. Indeed, ANN are known to provide good estimates of non linear response surfaces[5]. Moreover, as they provide an analytic formulation of the estimated model, ANN allow us rapid access to sensitivity by simply chain-differentiating the approximate function. However, despite the universal approximation property of the class of ANN as a non-linear model, we encountered some difficulties in performing global regression of our optimization constraint functions with sufficient quality. We solved this problem using Mixture of Experts (MoE) methodology to fuse local response surface models. Then we improved response surfaces quality.

In the following sections we outline the methods which are used in present research and then detail two application cases: composite and metal stiffened panels. We show how prior knowledge of mechanical constraints is used to divide the configuration domain in order to improve construction of surrogate models. Metamodels are embedded in local optimization benchmarks to check the validity of the approximation and its use as a constraint in an optimization process. Lastly, we end this article with some concluding remarks.

II. Response Surface Methodology and Neural Networks

A. Recall on ANN regression [4]

ANN are composed of several elementary nodes (or neurons) that are organized in separate layers. Each node i of layer k receives a linear combination of the outputs $y_{j,k-1}$ of the previous layer, and delivers output $y_{i,k}$. It is computed as a transformation of the linear combination through a smooth function H that is called the "*transfer"* or *"activation"* function:

$$y_{i,k} = H(\sum_j w_{ij,k}.y_{j,k-1} + b_{i,k})$$

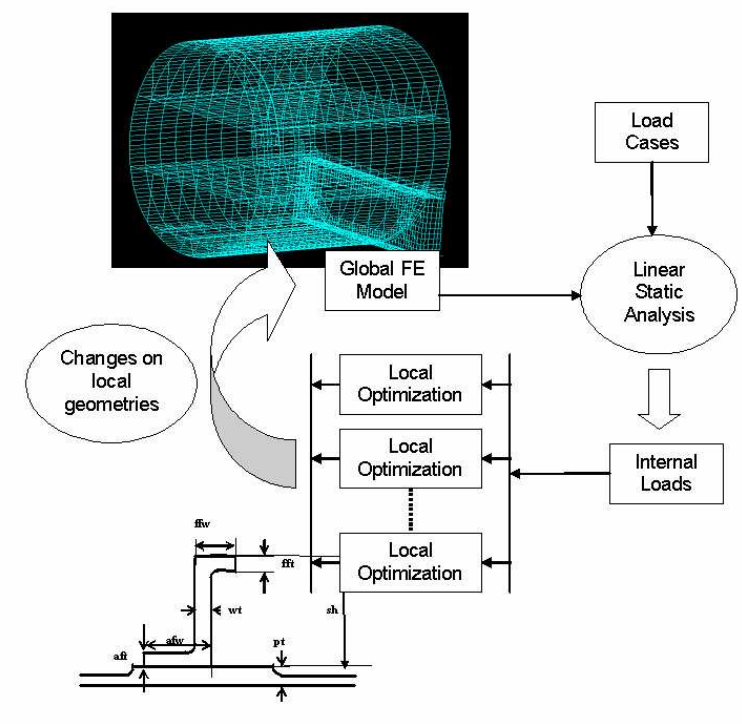

Figure 2: Bi level loose coupling optimization process

Coefficients of the linear combination $w_{ij,k}$ and $b_{i,k}$ are respectively called connection weights and bias.

The number of layers, the number of neurons in each layer and the nature of the activation function(s) -sigmoid, Gaussian or identity- are discrete hyperparameters of the architecture; they are set by the user. On the other hand, connection weights and biases are the configuration parameters. They vary continuously and are computed through an optimization algorithm so that the network outputs correspond with the training data.

This parameter estimation process is referred to as the training stage. It consists of changing the weight and the bias of the neurons in order to minimize a cost function that, in most cases, is the Mean Squared Learning Error (MSLE), i.e. the mean of the squared differences between network response to a given input and the desired target. The set of known input/target data (or I/O Pairs) is called the "training base" (TrB).

The number of hidden units (*Nhu*) determines the complexity of the response surface (RS) model: by adding neurons onto the hidden layers, one increases the network ability to fit exactly the data in the TrB. On the other hand, the prediction ability of the network, i.e. response to an unknown input, will be less efficient. This is called overfitting phenomenon. So, an optimal network architecture exists, depending on the size of the training base. Cross-validation criterion may be used to determine the optimal size of the architecture. Moreover, several methods allow us to avoid overfitting phenomenon, such as penalization of the cost function, early stopping of the training stage or bayes regularization[3]. We used Matlab© "Neural Network toolbox" to perform optimization and training.

We used three-layered feed-forward neural networks. Activation function of the hidden neurons is tangent-sigmoid or log-sigmoid, and output layer is linear.

We also used five-fold cross-validation criterion to acquire the most efficient architecture. It consists of randomly partitioning the training base into five equal parts. Then, in order to select the most suited architecture, for a given *Nhu* value, one performs 5 consecutive training steps of network using 4 out of 5 parts of the learning base gathered and computes the MSE on the last part. For any given architecture, the associated error is averaged over the 5 trials. The most efficient architecture is the one which results in the lowest error.

B. Mixture of Experts.

Mixture of experts (MoE) methodology was introduced by Jacobs and Jordan[6,7]. It consists of applying the principle of "divide and conquer": indeed it appears that a single network may have difficulties performing regression of a function if its variations change too much within the whole design space; for example, in case of a target function defined as the minimum of several different functions. To overcome these difficulties, MoE methodology suggests dividing the whole training base into subsets that correspond to different parts of the output function and to train separate networks on each subset. This partition of the design space can be done either by applying prior knowledge of the function and variables, or by a training algorithm. The network responses to a given input are then melted using either a Gaussian mixture or a "gating" network that gives coefficients of the combination, or in some cases selects a single network response according to the given input.

In our study, we had some prior knowledge of the effect of variables on output functions (see III.B). We used this knowledge to divide loading variables design space into smaller subspaces, corresponding to different cases in output variations. On each subspace, networks were trained on several thousand examples sampled as described below.

Then, expert network outputs were fused using a Gaussian Mixture of Experts, so that the response to an input X is:

$$F(X) = \frac{\sum_k \alpha_k . f_k(X)}{\sum_k \alpha_k}$$

where $f_k(X)$ is expert network k response to input X,

α_k is a coefficient computed as:

$$\alpha_k(X) = \exp\left(\sum_{j \in J_l} \frac{(X_j - c_{j,k})^2}{b_{j,k}} \right), \quad k = 1..N_{\exp}$$

where J_l denotes indexes corresponding to components of input X whose design space is divided[§],

$c_{j,k}$ is the centre of subdomain k along jth axe

$b_{j,k}$ is the Gaussian width, computed so that coefficient α_k is equal to 0.5 at the subdomain frontier.

This Gaussian mixture leads to continuous response of the Mixture of Experts, contrary to a discrete mixture which would use a sole network according to subdomain where input X lies. Furthermore, it also leads to continuous gradients of the response.

Note that the response surface defined by this mixture of experts is to be used in a local optimization process. This optimization problem consists of minimizing area super stiffener under mechanical constraints. Only geometrical variables are optimization parameters, whereas loading variables are set constant as inputs of the optimization. Thus, for any j in J_g, set of indexes corresponding to geometrical variables, we have:

$$\frac{\partial \alpha_k}{\partial X_j} = 0, \quad k = 1..N_{\exp}$$

and finally:

$$\frac{\partial F}{\partial X_j}(X) = \sum_{k=1}^{N_{\exp}} \alpha_k . \frac{\partial f_k}{\partial X_j}(X), \quad \forall j \in J_g$$

Last, gradient of expert network output is easily obtained from analytic formulation of f_k.

C. Design of Experiment

In both application cases, the dimension of the input space is high (8 or 11). This induces a high-dimensional neural architecture, and then requires a large-sized learning base to perform training. Yet, a mesh-based training base with more than 4^8 examples cannot be considered. Therefore we build our training bases using regular meshing on loading variables crossed with a latin-hypercube sampling (LHS) on design variables. Finally, the training of networks is performed using several thousand examples.

LHS[8] was first introduced by McKay in 1979. It is well used because it can cope with high dimensional input space, and is easy and cheap to generate. Indeed, LHS will explore n levels for all input variables within only n experimental runs, instead of n^d with a regular meshing of the input space, d being the dimension of the space. Furthermore, LHS has superior space filling property than random sampling as it ensures a uniformly distributed sampling for all variables within its design space.

Note that the size of LHS is set by the designer. It corresponds to the number of stratified levels for each variable. Practically, LHS is built as follows. First, each variable design space is divided into n equal intervals, and d permutations of set $\Omega = \{1,..,n\}$ are chosen, namely $\pi_1,..,\pi_d$. Then, the jth realization of variable k $\in$ {1,..,d} is located in interval $\pi_k(j)$. The value of this variable can be chosen at the centre of this interval, or randomly selected within this interval with a uniform probability.

One must remark that generating a LHS remains a random process; some criteria exists that measures quality of a design, such as minimax, maximum entropy or orthogonal property. But the generation of an optimal design with respect to one of these criteria requires solving an optimization problem and consequently a greater computational time. In this study, we use simple LHS.

[§] Experts domains can be defined on all or a part of components of input X. In our study, only loading variables define expert domains (see III).

As seen above, we cross these simple LHS with a mesh-based design on loading variables. This sampling strategy provides us with a more accurate estimation of the effect of variables which are specific in our Mixture of Experts model (see III.D).

III. First application case: Composite Stiffened Panel.

A. Global surrogate model

As described in the introduction, the aim of the study is to perform approximation of several mechanical criteria that are constraints of our local optimization problem. These criteria are formulated as Reserve Factors (*RFs*) so that a structure is feasible if its *RFs* are greater than 1. In this first application case, we focus on a T-shaped composite stringer joint to composite panels. *RF*s of interest give information on

- Panel local buckling (RF_{PNL_LOC}),
- Panel global buckling (RF_{PNL_GLOB}), expressing the collapse of the whole super stiffener,
- Stringer buckling (RF_{STR}), which is the minimum of local buckling *RF*s among the two parts of the stringer (web and flange).

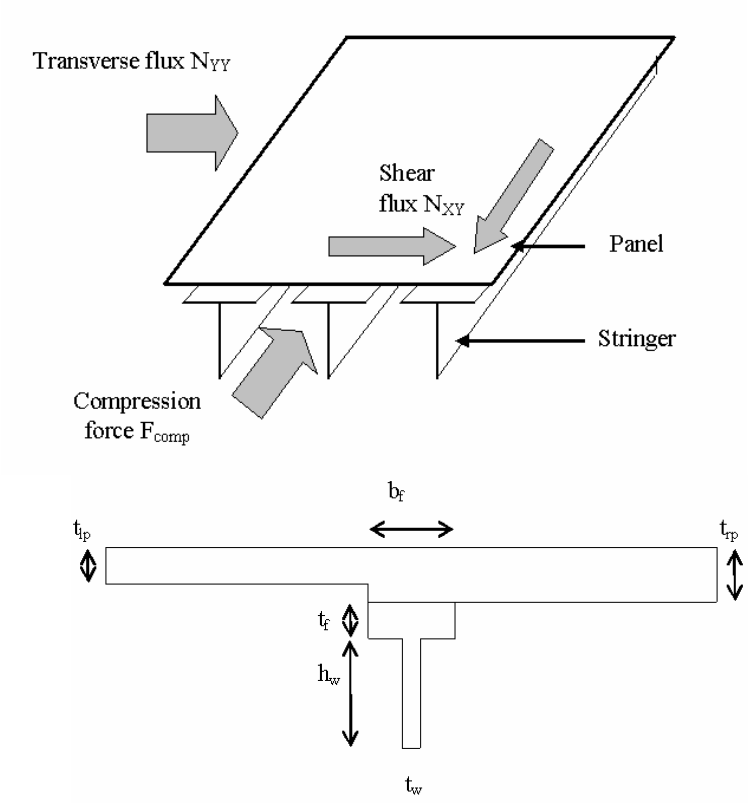

Figure 3: view of 2D loading on stiffened panel and detailed design of super stiffener

Values of these criteria according to design and loading variables are supplied by internal Airbus software to be used in a new project focused on optimization composite stiffened panels. This static calculation software assesses instability phenomena such as buckling, local buckling and post buckling solving an energy method (Rayleigh-Ritz method). It gives information on damage tolerance margins too.

Note that the software we used provides us with one *RF* for stringer buckling (RF_{STR}) and another one for panel buckling (RF_{PNL}), which is the lower among local and global *RF*. So, computed panel buckling *RF* is a continuous function with discontinuous gradient. But the software gives us the information of current panel buckling mode too.

These *RFs* depend on both the geometry of the structure and the applied loading. Here, super Stiffener's geometry is defined by:

- left and right panel thickness t_{lp} and t_{rp},
- stringer flange thickness and width t_f and t_f,
- stringer web thickness and height t_w and h_w,
- panel width and length.

In our study, panel width and length and flange width are constant, so that geometry is defined by 5 continuous variables.

The super stiffener is subject to two-dimensional loading defined by:

- compression force F_{comp} applied to the super stiffener (applying on both panel and stringer)
- transverse flux N_{YY} and shear flux N_{XY}, both applied to the panel only.

Thus, the input space is 8-dimensional in this first application case (5 geometrical variables and 3 loading ones).

First studies showed us that whereas ANN were able to approximate output RF_{STR} well they have more difficulty providing good results on RF_{PNL}: while MSLE is 0.025 on RF_{STR} (95% of examples on test base result

in under 0.05 absolute error), it increase to 0.158 on RF_{PNL} (and 75% of examples of test base give under 0.2 absolute error).

To have better insight of these problems, we performed approximation of RF_{PNL} on a reduced input space, for example by setting one loading variable out of 3 constant. In all cases, approximation was much more accurate, leading to an acceptable quality. So, difficulties in regression seemed to be caused by the cross effects of the three loading variables. Indeed, variations of buckling *RF* with respect to input variables could be totally different depending on values of loading variables (see Fig. 4).

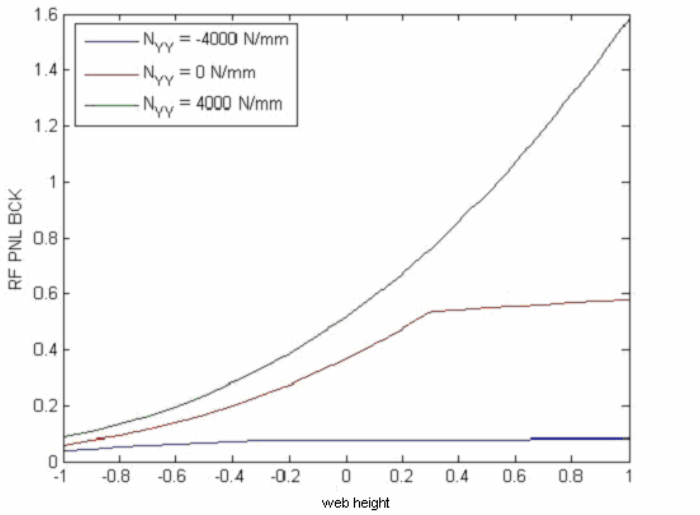

Figure 4: effect of web height for three transverse load conditions.

Actually, it appears that ANN have great difficulty capturing the switch between local and global buckling modes corresponding to a change in slope on response surface. This switch depends mostly on the value of loading variables. Moreover, Gibbs oscillations appeared around slope changes. These remarks lead to the building of local models that are governed by the dominant buckling mode and then to fuse them, using the MOE methodology which was described above.

B. Remarks on mechanical problem. Further study of buckling mode switching

Recall that a super stiffener's design is defined by 5 geometrical variables: two panel thicknesses and three stringer dimensions. But, these variables do have a crossed influence on panel and stringer buckling due to the stiffening ratio; indeed, compression force applied to super stiffener is divided onto the panel and the stringer with respect to this ratio. For instance, consider a constant load condition and assume that there is no local buckling mode, then an increase of one stringer dimension causes an increase of stringer area and a decrease of compression stress in the skin. So, this leads to an increase in RF_{PNL}.

Moreover, increase in stringer dimensions implies increase in stringer inertia and modifies support condition of skin on stringer. Thus a weak inertia stringer does not provide support to the skin, and buckling mode will be global, leading to super stiffener collapse. On the other hand, a high inertia stringer provides a support to the skin, avoids the super stiffener collapse, and finally makes local buckling modes dominant.

A contrario, when buckling of the skin is due to transverse compression or shear, the design of the stringer no longer influences RF_{PNL}, because the stringer does not bear part of these loads.

These changes in effects of the dimensions are illustrated in figure 4. Variations of RF_{PNL} with respect to stringer web height for three different values of transverse compression N_{YY} are represented. Other variables are set constant.

Note that:

- transverse tension ($N_{YY}>0$) has a stabilizing effect on the structure; there is no local skin buckling mode and therefore the increase in web height h_w involves an increase in RF_{PNL} following the above remark;
- the curve corresponding to null transverse effort case shows a switch between local and global buckling mode;
- Under a high transverse compression ($N_{YY}<0$), skin buckling mode is local, and stringer design has almost no effect on RF_{PNL}.

Lastly, we can remark that although loading variables have a cross effect on RF_{PNL}, the high value of one of these variables can make RF_{PNL} almost insensitive to the other two. For example, figures 5 a/ and b/ show that:

- under a high transverse compression, the value of compression force F_{comp} does not influence RF_{PNL};
- under a high compression force, shear flux does not modify RF_{PNL}.

According to these remarks, it appears that a relevant choice for the input space division in our case is to divide the loading space into subdomains in order to bound variables variation and so to reduce differences in the effect of the variables.

Please note that on figures given below, the values of the variables are rescaled between -1 and 1. For all design variables, -1 stands for the lowest value and 1 for the highest one. This is different for loading variables: F_{comp} = -1 stands for the highest compression force and F_{comp} = 1 stands for a small compression force. On the other hand, value -1 of variable N_{YY} stands for a high compression flux, +1 stands for a high tension flux and 0 stands for a null flux. Last, value -1 of variable N_{XY} stands for the lowest shear flux, say 0N/mm. Value +1 stands for the highest shear flux N_{YY}= 4000 N/mm.

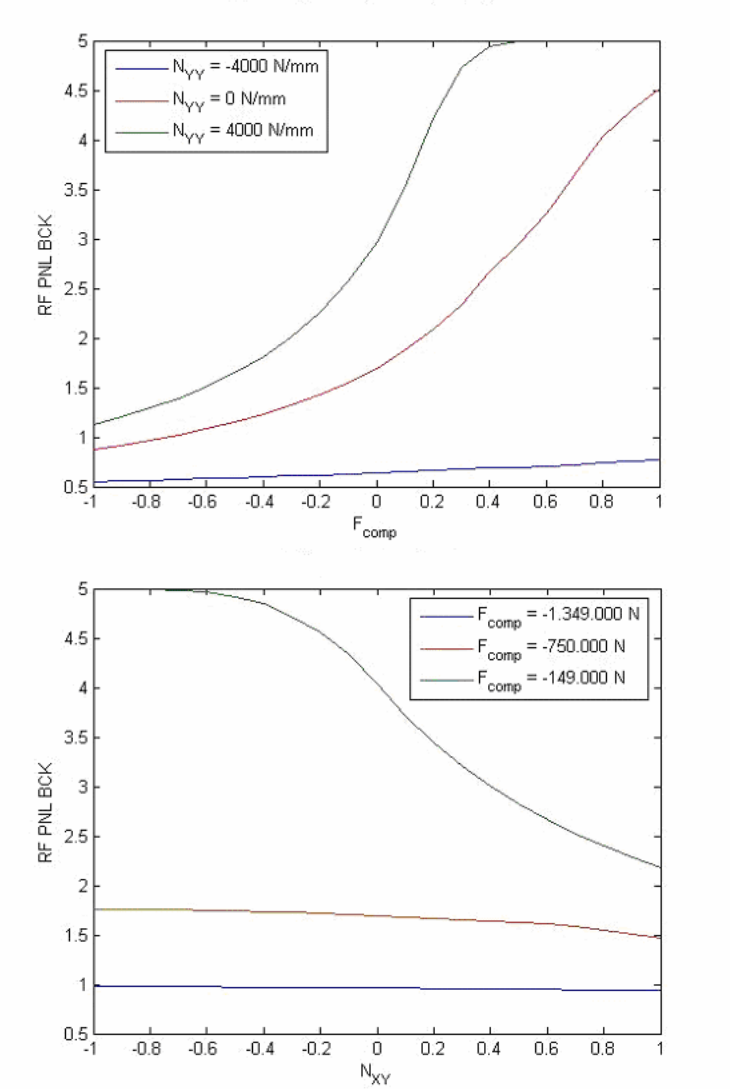

Figure 5: effect of compression force according to transverse flux (top) and effect of shear flux according to compression force (bottom)

D. Local surrogate models

The aim of the study was to perform regression of function RF_{PNL} and RF_{STR} on the following domains:

- t_{lp} and $t_{rp} \in$ [2 20];
- t_f and $t_w \in$ [2 30];
- $h_w \in$ [30 80];
- $F_{comp} \in$ [-1.5E6 -1000];
- $N_{YY} \in$ [-5000 5000];
- $N_{XY} \in$ [0 5000].

Length unity is millimetre, forces are expressed in Newton and fluxes in N/mm. Due to the large differences between variables values, inputs are rescaled on [-1 1] before training. From now on, we shall use reduced variables values.

First, only compression force F_{comp} domain is divided into 10 subdomains. Each subdomain length is 0.2. On each subdomain, two networks are trained: one performs regression of RF_{STR}, the other performs regression of RF_{PNL}. The learning base is composed of 2000 examples, with F_{comp} uniformly distributed within its subdomain and other variables are sampled using Latin Hypercube Design. These 2000 examples are used to select the optimal network architecture according to the five fold cross validation criterion, so that only 1600 examples are used in each training stage. In order to check networks generalization capabilities, an additional test base is made with 500 randomly chosen examples.

Figures 6 a/ and b/ show the evolution of root mean squared error (RMSE) measured on test base vs. compression force F_{comp}. It appears that RMSE increases as loading decreases. Actually, as explains above, when F_{comp} is great, it has a predominant effect on *RF* values. Other loading variables produce little effect to the values, so that within a subdomain the effective input space is almost 6-dimensionnal (instead of 8) and regression is greatly eased.

Moreover, one can see that neural approximation is better on output RF_{STR}. Regression is accurate enough on domain $F_{comp} \in$ [-1 0]. But error is still unacceptable on the rest of the domain, and on output RF_{PNL} on [-0.4 1].

To improve the quality of regression we divide the input space according to values of transverse flux N_{YY} too, where this is useful. Moreover, to ease regression of output RF_{PNL}, we perform training of three networks to model RF_{PNL}: one is dedicated to local buckling mode RF_{PNL_LOC}, another is dedicated to RF_{PNL_GLOB}, and the last one is a classifier trained to predict active panel buckling mode.

Thus, we now need examples corresponding to both panel buckling modes. But, as N_{YY} becomes high (and positive), there are fewer and fewer examples of local buckling. Finally, we build the learning base and perform network training according to the following process:

- a 3000 examples learning base is computed the same way as described above: F_{comp} and N_{YY} are uniformly distributed and other variables are set on LHD;
- two networks are trained on output RF_{STR} and on panel buckling mode;
- new examples are added using this buckling mode classifier until 2000 examples of each buckling mode are reached. Training of networks on output RF_{PNL_LOC} and RF_{PNL_GLOB} are then performed.

Tests on several partitions of loading space reveal that:

- at high compression force ($F_{comp} < -0.4$), there is no need to divide N_{YY} space;
- for F_{comp} in [-0.4 0], it is sufficient to divide N_{YY} space into two subdomains: N_{YY} in [-1 0] (compression) and N_{YY} in [0 1] (tension);
- for weaker compression force ($F_{comp} > 0$) N_{YY} space has to be divided into six subdomains: N_{YY} in [-1 -0.6], N_{YY} in [-0.7 -0.3], N_{YY} in [-0.4 0] and N_{YY} in [0 0.4], N_{YY} in [0.3 0.7] and N_{YY} in [0.6 1].

To understand the reason for this partition, one must consider the above remarks on effects of loading variables. As F_{comp} is high, it has a predominant effect on *RF*s. When it decreases, buckling mode is more and more due to transverse flux N_{YY}, and we have to get a sharper idea of the effect of N_{YY} on *RF* values.

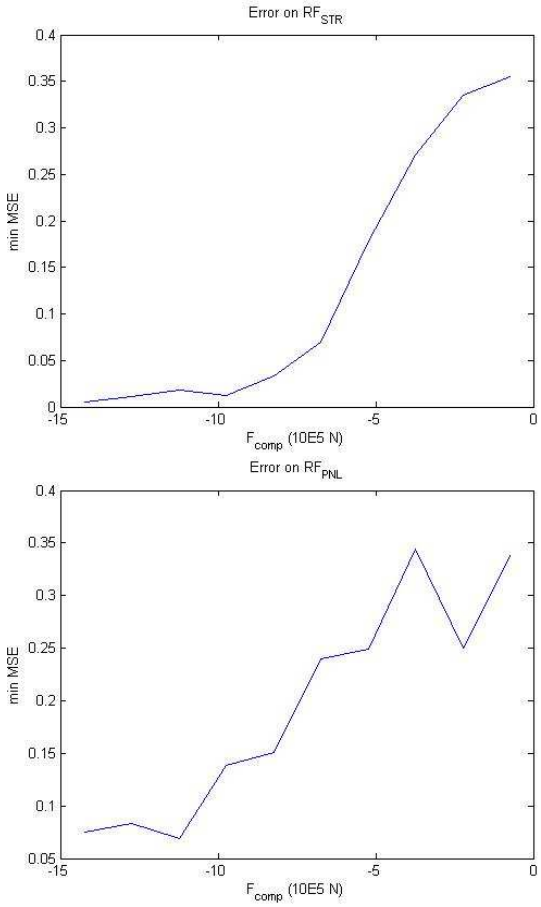

Figure 6: evolution of MSE vs. compression force a/ MSE on RF_{STR} –b/ MSE on RF_{PNL}

After training, the goal of 90% of test base examples under 0.03 absolute error is reached on each subdomain.

Finally, experts outputs are fused following a Gaussian Mixture as described above (see II.B).

E. Using surrogate model in optimization process

The aim of the study is to use neural approximating model in the following optimization problem:

$$Min \quad A_{SSt}(t_f, t_w, h_w, t_{lp}, t_{rp})$$

$$s.t. \quad \begin{cases} RF_{PNL}(t_f, t_w, h_w, t_{lp}, t_{rp}, F_{comp}, N_{YY}, N_{XY}) \geq 1 \\ RF_{STR}(t_f, t_w, h_w, t_{lp}, t_{rp}, F_{comp}, N_{YY}, N_{XY}) \geq 1 \\ 3 \leq b_f / t_f \leq 20 \\ 3 \leq h_w / t_w \leq 20 \end{cases}$$

where A_{SSt} stands for Super Stiffener Area.

Loading variables F_{comp}, N_{YY} and N_{XY} are optimization inputs; geometrical variables t_f, t_w, h_w, t_{lp} and t_{rp} are optimization variables.

The last two constraints determine the stringer's aspect ratio; the two first constraints are mechanical constraints described above. They are replaced by reduced model in our implementation.

In order to measure savings from the use of a reduced model instead of analytic constraints, we compare the results of several optimization runs on loading space meshing defined by:

- $F_{comp} \in$ {-150.000, -400.000, -600.000, -1.000.000, -1.400.000},
- $N_{YY} \in$ {-4000, -2000, 0, 2000, 4000},
- $N_{XY} \in$ {0, 2000, 5000},

leading to 75 optimization runs.

Optimizations are run on BOSS Quattro© software using Globally Convergent Method (GCM) –a second order algorithm using moving asymptotes as proposed by Svanberg).

First of all, the objective of time reduction is greatly reached: computational time for 75 optimization processes falls from more than 10 hours when using analytical software to 2 hours with MoE approximation. The reduction is merely due to computation of analytical gradients instead of using finite difference.

The quality of optimization results depends on the domain where input lies:

- Optimization examples under high transverse loading (N_{YY} = {-4000; -2000}) give very accurate results; maximum error on *RF* is 0.117 on RF_{STR} and 0.09 on RF_{PNL}. Only 4 (resp. 1) examples out of 30 lead to an absolute error on RF_{PNL} (resp. RF_{STR}) greater than 0.05. Moreover, optimal super stiffeners' areas given by optimization run on approximated constraints all are between 2 and 9% less than those obtained with original model constraints.
- Under null transverse flux (N_{YY} = 0), approximation of RF_{STR} is good, with one sole example out of 15 leading to a high absolute error (0.165), other are below 0.04 absolute error. But values of RF_{PNL} are not well estimated under low loading. Only 7 examples out of 15 give error under 0.06. Moreover, examples for which applied compression force is -150.000 N all give error higher than 0.15. Optimal areas are between 11% higher and 8.7% less than those obtained with original model constraints.
- Under transverse tension (N_{YY} > 0) errors on *RF*s are even higher. Expert networks on RF_{PNL} lead to an absolute error on optimal *RF* over 0.1 on 10 out of 30 examples. Experts on RF_{STR} give a better approximation: with N_{YY}=2000N/mm only 2 examples (out of 15) lead to an error greater than 0.05. But with N_{YY}=4000N/mm, 6 examples (out of 15) lead to an error greater than 0.1, others are under 0.05 error. Moreover, optimal areas obtained by optimization processes using approximated *RF*s are greater than those obtained with analytical constraints.

We see that, although expert networks all give good approximation results in their own subdomain, we still get unacceptable errors on low loading domain. Note that most of these great error values are due to a mistake in predicted buckling mode. Model improvement should therefore be focused on networks which predict panel buckling mode.

IV. Second application case: Metallic Super Stiffener.

In this second application case, we still aim to perform approximation of the mechanical criteria used in local optimization process. This time, the study focuses on metallic stiffeners with Z-shaped stringer. Input and output spaces in this study have changed; indeed, super stiffener is now defined by:

- 2 external geometrical variables: panel width (also referred as stringer pitch) and curvature radius
- 7 local geometrical variables, defining super stiffener detailed design: panel thickness and thickness and width for each stringer element (skin side flange, web and foot)
- 2 loading variables: compression force and shear flux.

Furthermore, outputs to be approximated are now:

- panel buckling reserve factor RF_{BCK} (corresponding to panel local buckling mode)
- super stiffener collapse reserve factor RF_{COLL}, which is the minimum *RF* among general buckling (or collapse), and stringer elements buckling reserve factors.

Note that in this study, the cross effects of loading variables will be less important because there is no orbital compression or tension. On the other hand, there are more geometrical variables.

Function approximation is to be performed on the following domain:

- panel width and curvature radius are discrete variables; the couple of variables (Pitch, R_{curve}) takes 13 values. Note that these physical variables do have a continuous effect on mechanical criteria.
- Detailed geometry variables vary continuously in their domains, which are [1.6 4] for thicknesses variables, [10 60] for flange width and [25 65] for web height.
- Loading variables vary from 10,000 to 300,000 N for compression force F_{comp} and from 5 to 130 MPa for shear stress τ_{XY}.

Previous studies showed that high errors in function approximation are mainly located at domain boundaries. To cope with this problem, variables variation domains were increased to obtain a more accurate approximation on the domain of interest. For instance, thicknesses vary from 1 to 6 mm, flange varies from 8 to 65 and web height varies from 20 to 60.

Variables are now rescaled in [0 1] before the training stage.

The training base is made of 1600 examples computed on the following design:

- loading variables form a regular meshing with 17 values for F_{comp} and 11 values for τ_{XY};
- detailed geometry variables are set on a ten levels LHS;
- external geometry variables couple (Pitch, R_{curve}) is randomly chosen in its allowable values.

Note that this initial design leads to 2470 calculation but only 900 effective *RF* values. Other cases are out of software validity domain. Thus, examples have to be added in the training base in order to obtain enough I/O pairs for training.

The test base is made of 200 randomly chosen examples.

Training performed on this learning base shows that output RF_{BCK} is well estimated with a sole three-layered network with 15 neurons on both hidden layers: network response on test base leads to 80% examples below 0.02 absolute error, and 90% under 0.035.

Approximation of RF_{COLL} seems to be more difficult: a sole network produces an unacceptable error with only 14% examples in test base under 0.02 absolute error and 90% examples under 0.17.

One can explain these differences by considering that:

- RF_{BCK} is now easier to approximate because it only represents skin local buckling, there is no switch in buckling mode. Also, there is no transverse loading.
- On the other hand, RF_{COLL} now gather two different buckling modes that are super stiffener collapse and stringer element local bucking.

Again, a mixture of experts methodology is used to improve approximation quality. F_{comp} variation domain is greatly reduced compared to the former study. We therefore require fewer expert networks to cover the whole domain. Finally, loading space is divided into four subdomains defined by: (F_{comp}, τ_{XY}) in [0 0.35]x[0 0.5], (F_{comp}, τ_{XY}) in [0 0.35]x[0.5 1], (F_{comp}, τ_{XY}) in [0.325 0.675]x[0 1], (F_{comp}, τ_{XY}) in [0.65 1]x[0 1].

With this partition, regression error is efficiently decreased. A mixture of experts methodology leads to an approximation of good quality with 78% examples in test base under 0.03 absolute error. Again, difficulties in regression are located in low loading domain (F_{comp} < 0.35 in reduced value).

This mixture of expert networks is now tested on 20 local optimization cases run on the regular meshing of loading space defined by

$$(F_{comp}, \tau_{XY}) \text{ in } \{50{,}000;\ 100{,}000;\ 200{,}000;\ 250{,}000\} \times \{0;\ 20;\ 50;\ 100;\ 130\}.$$

Each optimization process is run three times, with three different initial designs to avoid local minima phenomena. Thus, comparison is made of 60 optimization results.

First of all, one must note that optimization processes under high loading (F_{comp} = 250,000N) have stopped before convergence because the path followed during optimization went out of software validity domain. This problem of course did not occur when using neural regression of constraint functions and this does feature an advantage in use of a response surface model instead of dedicated software. Consequently a comparison can only be made on 45 optimization results.

Optimization processes run with approximated constraints lead to the following results:

- only 3 cases out of 45 lead to an error of order 0.1 on RF_{BCK}. All other cases lead to an error under 0.04.
- error on RF_{COLL} is more important. 12 cases lead to unacceptable error, greater than 0.1. These cases are all located in the low-loading domains (F_{comp} = 50,000N or F_{comp} = 100,000N and τ_{XY} = 0MPa).Three more cases give error between 0.06 and 0.09. All other cases are below 0.05 absolute error.
- On cases leading to low error (<0.05), optimal super stiffener's areas, i.e. the lowest area among three tries with different initial designs, is lower when obtained using approximated constraints, with mean 4.3%.
- Last but not least, computation time is divided by two, with 45 minutes per optimization with approximated constraints vs. 1h30 when using dedicated software.

Thus, one can see two advantages in using response surface model in optimization processes: first, computation time is greatly decreased, without damages in optimization results. And moreover, use of approximated constraints allows avoiding problems in convergence during optimization process due to software limitations in validity domain.

V. Conclusions

The use of surrogate models to perform regression of constraints in an optimization problem requires a high quality approximation: whereas approximation of the objective function just has to give an accurate estimate of the function near its optimum, the use of approximated constraints in an optimization process does require a great precision of approximation on the whole critical region of the input space.

We showed how division of input space into subdomains based on a prior knowledge of the design space and use of Gaussian mixture of local experts has allowed us ease of regression and provided us with an approximation of constraints functions that were sufficient. The use of such a mixture of experts in optimization has given good results here. In most optimization benchmarks, computational times were greatly reduced without damaging results. Another great advantage of the use of the surrogate model is that there is no crash during optimization processes due to a restricted validity domain of the replaced software. But metamodels still has to be improved on low loading domains. One may consider two ways of improvement: before or after training of the experts.

After training stages, the mixture of expert could be improved by optimizing the centres of width of Gaussian function used in mixture. This may move the boundaries of defined subdomains according to each expert quality.

Additional training examples could also be chosen using optimal search strategy based on the statistical frame introduced by Cohn[9].
On the other hand, the training of expert networks could be performed along with the partitioning of the design space. This is what is done in EM algorithm for example. Lastly, one may consider a simple idea that is to divide the design space more sharply. However this last idea requires a longer training time and a greater computational cost that may outweigh the time saving of the use of surrogate models.

References

[1] Schmit Jr., L. A., Ramanathan, R.K., "Multilevel Approach to Minimum Weight Design including Buckling Constraints", in *AIAA Journal* 16 (2), 1978

[2] Balling, R.J., Sobieszczanski-Sobieski, J., "An Algorithm for solving the System-level Problem in Multilevel Optimization", NASA Contractor Report 195015, 1994

[3] Simpson, T. W., Mauery, T.M., Korte, J.J., Mistree, F., "Comparison of Response Surface and Kriging Models for Multidisciplinary Design Optimization", AIAA paper 98-4755, 1998

[4] Sellar, R.S., Batill, S.M., Renaud, J.E., "Response Surface Based, Concurrent Subspace Optimization for Multidisciplinary System Design"AIAA paper, 1996.

[5] Dreyfus, G., Samuelides, M., Martinez, M., Gordon, M., *et al.*, "Réseaux de Neurones : Méthodologie et applications", Eyrolles ed., Paris, 2002

[6] Jordan, M.I., Jacobs, R.A., Nowlan, S.J., Hinton, G.E. "Adaptative Mixture of Local Experts", in *Neural Computation*, 3, 79-87 1991

[7] Jordan, M.I., Jacobs, R.A., "Hierarchical Mixtures of Experts and the EM Algorithm", in *Neural Computation*, 6, 181-214, 1994

[8] Wang, G.G., "Adaptative Response Surface Method Using Inherited Latin Hypercube Design Points", Transaction of the ASME, Journal of Mechanical Design, Vol. 125, pp. 210-220, June 2003.

[9] Cohn, D.A., "Neural Network Exploration using Optimal Experiment Design", MIT CBCL paper No. 99, 1994