24th COBEM - 2017

24th ABCM International Congress of Mechanical Engineering
December 3-8, 2017, Curitiba, PR, Brazil

COBEM-2017-2396
A TOOL FOR LOCAL BUCKLING OPTIMIZATION OF COMPOSITE PANELS USING NASTRAN® SOL 200

Guilherme Pedroso Pires Abreu
Saullo Giovani Pereira Castro
Embraer, Brazilian Aerospace Company, São José dos Campos, SP, Brazil
guilherme.abreu@embraer.com.br
saullo.castro@embraer.com.br

Jose Antonio Hernandes
ITA, Technological Institute of Aeronautics, Division of Aeronautical Engineering, São José dos Campos, SP, Brazil
hernandes@ita.br

Abstract. *This work performs a local buckling optimization of composite panels with advanced constraints via multidisciplinary optimization solution of NASTRAN® (SOL 200) proposing a sizing methodology of local buckling with a semi-analytical approach, alternatively to the buckling eigenvalue solution at finite element solvers. An external FORTRAN® subroutine calculates at optimization runs the buckling eigenvalue for each panel. A Python-built management tool of the structure and the finite element model was extended, to control and create, automatically, the optimization cards. The optimization was performed considering two types of loading, one simulating a limit up-bending condition with load factor of 2.5g and another one simulating a limit down-bending condition with load factor of -1.0g. Results present that this methodology has a significant potential of utilization in the sizing of aeronautical structures.*

Keywords: *Structural Optimization, Composite Materials, Buckling Analysis, Structural Analysis, Finite Element Method.*

1. INTRODUCTION

After more than a century, that humankind was able to design, size and build an aircraft, aeronautical industry and, as a consequence, aeronautical engineering developed many new technologies which put this industry where it is now. However, even with this huge development, many areas of engineering lack new concepts and new definitions. Researches, development and new work must continue this quest for knowledge.

Composite materials have many advantages in comparison with metallic materials, like structural weight, stiffness and orthotropic properties, preventive maintenance. Nevertheless, metallic materials have also advantages against composite materials in other characteristics, like impact resistance, electric conductivity and repairing. That is why other authors stand that composite materials and metals can live together, using both at their best way.

The purpose of this work is to perform a local buckling optimization of composite panels. Such an objective can be achieved by extending a powerful tool for creating optimization cards in an automated way from a FE model manager in Python and a semi-analytical method of panel buckling calculation in a FORTRAN® external subroutine in the optimization (MSC NASTRAN® SOL 200) run.

2. FINITE ELEMENT MODEL

Wing box geometry was built in Catia (Dassault Systèmes, 2015). The finite element model of wing structure used a mesh refinement compatible with typical of preliminary design phase. The wing structural layout is shown in Fig. 1.

The modeling strategy was:

- Spars' webs have different properties from bay to bay;
- Each wing panel has its own property;
- Bar elements that represent panel stringers or spar caps have their own properties and orientations;
- Spars and skins were modeled as laminate plate elements and ribs as metallic plate elements;

- The wing main box is clamped at the root and simply supported at spar 3 to fuselage attachment;
- The considered FE model layup type in this work is the NASTRAN® smeared laminate (symmetrical and balanced laminate, where the solver does not take into account the stacking sequence);
- Initial composite layup is shown in Tab. 1.

Figure 1. Wing box structural layout

Table 1. Initial composite layup of FE model

Ply ID	Thickness [mm]	Orientation [°]
1	1.25	0
2	1.25	45
3	1.25	-45
4	1.25	90

The applied loading was defined by limit condition:

- Lift distributed on wing panels (Fig. 2) assumed as aircraft MTOW (32 ton – 16 ton per semi-wing):
  - Up-bending load case: load factor of 2.5g;
  - Down-bending load case: load factor of -1.0g.

The applied loading is shown in Fig. 2.

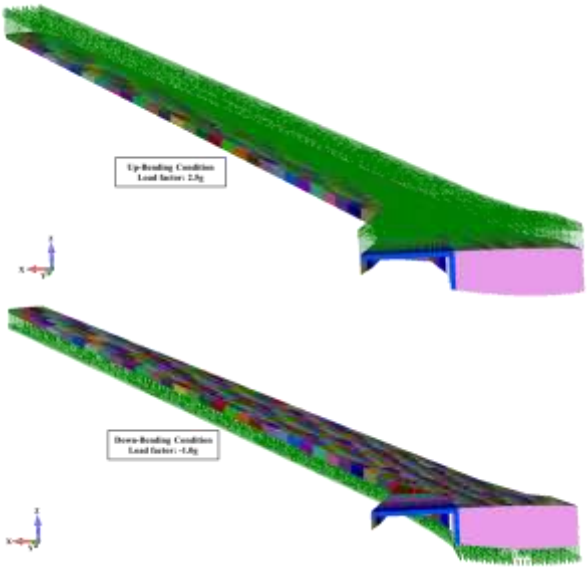

Figure 2. Applied loading in FE model

3. BUCKLING CONSTRAINT

For the buckling analysis, the governing equation is the eigenvalue problem of Eq. (1):

$$[K_0]\{\Phi\}-\lambda[K_{G0}]\{\Phi\}=\{0\} \tag{1}$$

Where $[K_0]$ is the constitutive stiffness matrix, $[K_{G0}]$ is the geometrical stiffness matrix, $\{\Phi\}$ is the global displacement vector of buckling mode and λ is the critical buckling load factor (Castro, 2014).

Equation (1) is the core of the FORTRAN® external subroutine called at NASTRAN® SOL 200. This subroutine solves recursively this eigenvalue problem to each wing panel by a semi-analytical approach based on the Ritz method, as formulated and implemented by (Castro, 2014).

4. OPTIMIZATION CRITERIA AND METHODOLOGY

The optimization cards were built by a suite of Python compilations, a structure manager library called structManager (Castro and Abreu, 2016), where the FE model is organized and seen in a different configuration, with structural elements and structural assemblies (groups of structural elements).

4.1 Structural Elements and Assemblies

For optimization purposes, the FE model was organized in a cascade structure of major elements according to their structural function. This criterion is explained below:

- *Structural elements*: First level of primary structure as panels, stringers, webs and flanges. They form a group of finite elements that plays the same role in the model. For example, the Panel 1.1 is formed by a group of co-located plate elements and nodes that has the same properties and play the same structural function. Figure 3 shows some examples of this organization;
- *Structural assemblies*: Second (or higher) level of primary structure that consists of groups of structural elements that play, together, the same structural function. For example, reinforced panels (panels + stringers), spars (web + flanges) and ribs (web + flanges – in this work, a simplified rib, as a structural element, was considered, with no caps). Figure 3 shows some examples of structural assemblies.

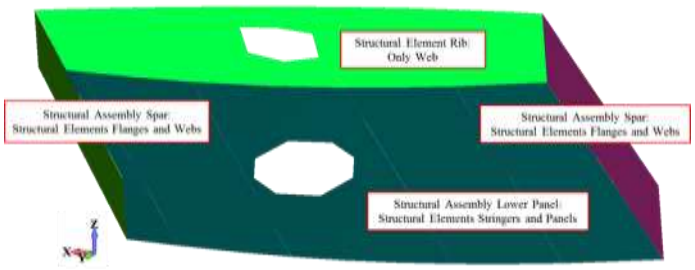

Figure 3. Examples of structural elements and assemblies at one wing box bay

4.2 Assumptions

To perform this optimization, some assumptions and initial conditions were set to simplify and/or to promote a feasible result according to the calculation subroutine. These parameters are listed below:

- NASTRAN® solution of optimization: SOL 200;
- NASTRAN® solution called every optimization: SOL 101 – linear static analysis;
- Laminate type: SMEAR (NASTRAN®). All plies are specified, but stacking sequence is ignored;
- Load idealization (Fig. 4) – transferring loads from the finite element model to the semi-analytical model:
  - Longitudinal, transversal and shear membrane stresses from the central element (to avoid disturbed zones next to interfaces);
- Minimum thickness of each ply angle at laminates: 10% of total laminate thickness. For 90° oriented plies, this ratio was fixed in 10% ($p_{90}=0.1$). To other orientations, only the lower boundary of ply thickness is fixed at 10%;
- Optimization constraint: local buckling of wing panels;
- Boundary conditions of semi-analytical method of buckling calculation: simply supported at panel ends.

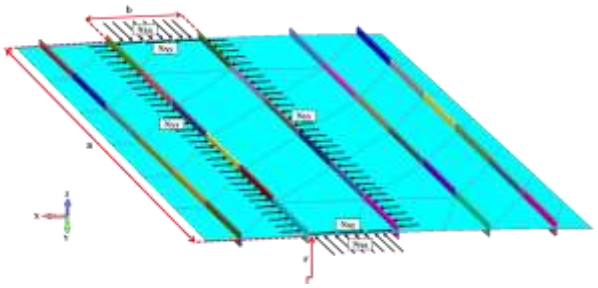

Figure 4. Loading representation in a generic wing panel

4.3 Design Variables and Parameters

The design variables (*DESVARs*) of this work were set in a certain way that was possible to change plies thicknesses of layups:

- Laminate thickness (t) – according to smeared laminate approach:

$$t = t_0 + t_{45} + t_{90} \tag{2}$$

- Thickness ratio of 45° angled plies (p45) – as a smeared laminate approach:

$$p_{45} = {}^{t_{45}}/_{t}; \tag{3}$$

Figure 5 shows the Python code in the structManager ambient and the optimization card generated – in NASTRAN® syntax – for SOL 200 run:

```
if ptype == 'PCOMP':
    dvar_t = DESVAR('PCt', self.t, self.t_lb, self.t_ub)
    self.add_dvar(dvar_t)
    dvar_p45 = DESVAR('PCp45', self.p45, self.p45_lb, self.p45_ub)
    self.add_dvar(dvar_p45)
```

Generated optimization card:

```
$
$ DESIGN VARIABLES
$
DESVAR   9000000     PCt     5.0     0.1    20.0
DESVAR   9000001   PCp45     0.5     0.1     0.5
```

Figure 5. Python code for design variables configuration and an example of an optimization card

Since the 90° angled plies thickness ratio was set as 0.1 and the considered laminate is a smeared type (symmetric and balanced), the thicknesses at 0°, 45°and 90° can be easily calculated by Eq. (4) and Eq. (5):

$$t_0 = (1 - p_{45} - p_{90}) * t \tag{4}$$

$$t_{90} = p_{90} * t \tag{5}$$

To lock and connect this thickness calculation to panel's properties (NASTRAN® *PCOMP*), to promote the properties updating at each optimization step, the NASTRAN® SOL 200 parameter *DVPREL2* – design variable to property relation – was set to each ply thickness property of wing panels laminates. Each *DVPREL2* has its own *DEQATN* (design equation definition) entry to calculate the properties parameters. Figure 6 shows this arrangement.

```
deqatn0 = DEQATN('T0(t,p45,p90) = (1.-p45-p90)*t')
self.add_deqatn(deqatn0)
dvprel2 = DVPREL2('PCOMP', pid, 'T1', deqatn0.id)
dvprel2.add_dvar(dvar_t.id)
dvprel2.add_dvar(dvar_p45.id)
dvprel2.add_dtable(self.dtables['T90'][10])
self.add_dvprel(dvprel2)
```

Generated optimization card:

```
$
$ DESIGN VARIABLE-TO-PROPERTY RELATIONS
$
DVPREL2  9000000   PCOMP   83601      T1              9000000
+       DESVAR  9000000 9000001
+       DTABLE      T90
$
$ DESIGN EQUATIONS
$
DEQATN   9000000  T0(t,p45,p90) = (1.-p45-p90)*t
```

Figure 6. Python fragment of design variable to property relation configuration and an example of DVPREL2 + DEQATN optimization cards

As shown in Fig. 4, the loading came from the membrane stresses at panels. To get this output from a NASTRAN® run, were defined design sensitivity responses (NASTRAN® SOL 200 *DRESP1* parameter) to store the N_{xx}, N_{yy} and N_{xy} values from each run to variables used as input parameters. The *DRESP1* parameter gets, from the linear static analysis – SOL 101 – the membrane stresses from the central element of each wing panel. Figure 7 shows the Python code of *DRESP1* structure to membrane stress storing parameter. These membrane stresses shown in Fig. 4 and Fig. 7 are the input loads at panel buckling calculation in the external FORTRAN® subroutine.

Figure 7. Python code to DRESP1 configuration to retrieve loading data from each SOL 101 called at SOL 200 optimization steps

All other parameters that are set as constants before each optimization are stored in a design table of constant parameters (NASTRAN® SOL 200 *DTABLE*) – Figure 8 shows the following structure:

- Panel width (*a*);
- Panel length (*b*);
- Panel radius (*r*);
- Elastic modulus at longitudinal direction (E_1);
- Elastic modulus at transversal direction (E_2);
- Shear modulus (G_{12});
- Poisson's ratio between longitudinal and transversal directions (v_{12});
- Poisson's ratio between transversal and longitudinal directions (v_{21}).

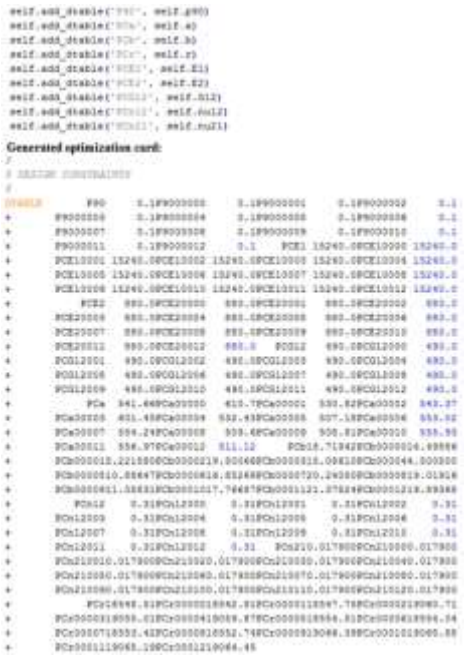

Figure 8. Python code to DTABLE configuration and an example of optimization card

4.4 Monitored Outputs

During the optimization, some output(s) must be monitored to promote feedback to the optimizer in order to control and set the design variables for the following run.

With all the parameters set, the monitored output for panel optimization is the buckling eigenvalue from the FORTRAN® external subroutine (Castro and Abreu, 2016) called by NASTRAN® SOL 200 optimization. This subroutine uses, as input data, t, a, b, r, E_1, E_2, G_{12}, v_{12}, v_{21}, N_{xx}, N_{yy} and N_{xy}.

With laminate thicknesses set as design variables, material, properties and geometric parameters as model constants at design table and membrane stresses as sensitivity responses, the semi-analytic method implemented in the external FORTRAN® subroutine called at NASTRAN® SOL 200 run calculates wing panels buckling eigenvalues. The design sensitivity response type used in this case is the *DRESP3*, which defines an external response using user-supplied routine. Figure 9 shows the Python code of structManager (Castro and Abreu, 2016) section of *DRESP3* creation and an example of an optimization card.

```
# calculating buckling eigenvalue using an external subroutine
# all parameters (desvars, dtables, dresp's) that this DRESP3 needs to run are listed below
# creating DRESP3
dresp = DRESP3('PCBUCK1', 'PCBUCK', 'BUCK_PC')
dresp.add_dvar(self.dvars['PCt'].id)
#dresp.add_dvar(dvar_t.id)
dresp.add_dtable(self.dtables['PCa'][0])
dresp.add_dtable(self.dtables['PCb'][0])
dresp.add_dtable(self.dtables['PCr'][0])
dresp.add_dtable(self.dtables['PCE1'][0])
dresp.add_dtable(self.dtables['PCE2'][0])
dresp.add_dtable(self.dtables['PCG12'][0])
dresp.add_dtable(self.dtables['PCn12'][0])
dresp.add_dtable(self.dtables['PCn21'][0])
dresp.add_drespl(dresp_Nxx.id)
dresp.add_drespl(dresp_Nyy.id)
dresp.add_drespl(dresp_Nxy.id)
self.add_dresp(dresp)
```

Generated optimization card:

```
DRESP3   9000003 PCBUCK1  PCBUCK BUCK_PC
+        DESVAR   9000000
+        DTABLE       PCa      PCb      PCr     PCE1     PCE2    PCG12    PCn12
+                   PCn21
+        DRESP1   9000000 9000001 9000002
```

Figure 9. Python to DRESP3 configuration to calculate buckling eigenvalue of wing panels and a fragment of an optimization card

4.5 Constraints

The constraints applied at this optimization act on the monitored outputs of wing panels buckling eigenvalues at DRESP3 level:

$$\lambda = {P_{allow}}/{P_{app}} = 1.0 \quad \Rightarrow \quad M.S._{lim} = \lambda - 1; \tag{6}$$

The application of this constraint is done with the NASTRAN® SOL 200 *DCONSTR* parameter, which is the design constraints parameter. Figure 10 shows the Python code and a fragment of an optimization card:

Figure 10. Python code to set response constraints at structManager and an example of an optimization card

4.6 Optimization Data

This section presents the optimization data for all SOL 200 runs. Table 2 shows the detailed amount of optimization parameters.

Table 2. Number of optimization parameters per run

RUN		DESIGN VARIABLES	CONSTANT PARAMETERS	VARIABLES TO PROPs RELATION	DESIGN RESPONSES (Nx, Ny, Nxy)	DESIGN RESPONSES (FORTRAN EIGV λ)	CONSTRAINTS
OPT1	Bays 01-05	234	1053	390	351	117	117
OPT2	Bays 06-10	198	891	330	297	99	99
OPT3	Bays 11-15	180	810	300	270	90	90
OPT4	Bays 16-20	148	666	247	222	74	74
OPT5	Bays 21-25	126	567	210	189	63	63
OPT6	Bays 26-30	106	477	177	159	53	53
OPT7	Bays 31-35	72	324	120	108	36	36
OPT8	Bays 36-37	28	126	47	42	14	14
OPT9	All Bays	1092	4914	1820	1638	546	546

5. BUCKLING OPTIMIZATION

In this work, the optimization solution used is the SOL 200. This one performs many rounds of a selected basic analysis solution (MSC Software, 2010). To save run time, the buckling calculation philosophy adopted was not to use the NASTRAN® buckling eigenvalue solution (SOL 105), but to calculate the buckling eigenvalue in an external FORTRAN® subroutine (Castro, 2016) with the membrane forces of panels' elements after runs of static analysis solution (SOL 101). Figure 11 shows a schematic diagram of NASTRAN® SOL 200 optimization structure. Figure 12 shows the flux of work using structManager to NASTRAN® SOL 200.

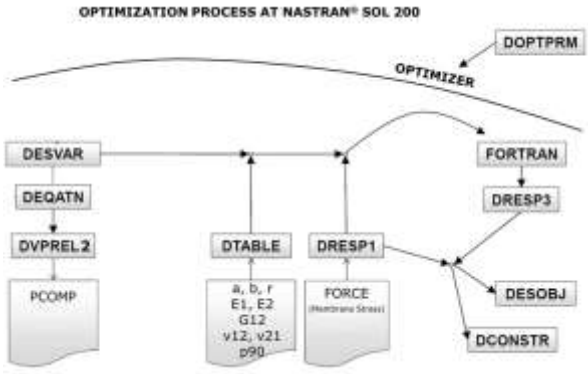

Figure 11. Wing composite panels optimization process at NASTRAN® SOL 200

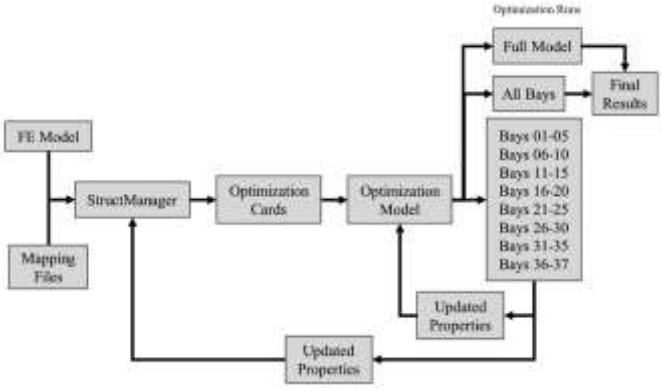

Figure 12. Optimization process at NASTRAN® SOL 200 using structManager

After all optimization runs, the evolution of laminates' thicknesses are shown in Fig. 13 and Fig. 14:

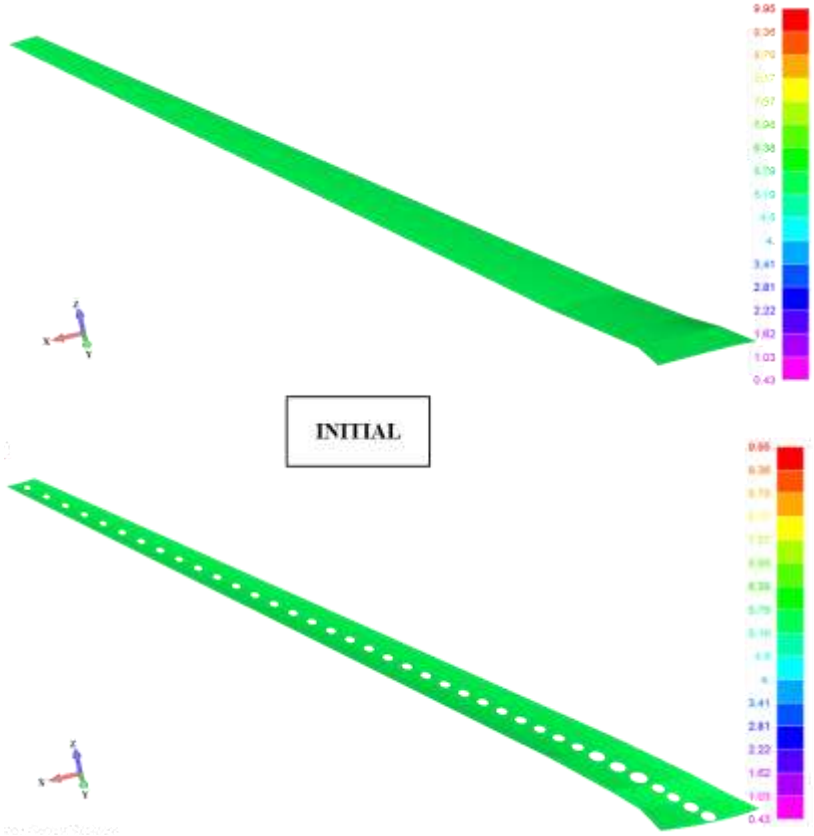

Figure 13. Initial thickness map at upper and lower panels

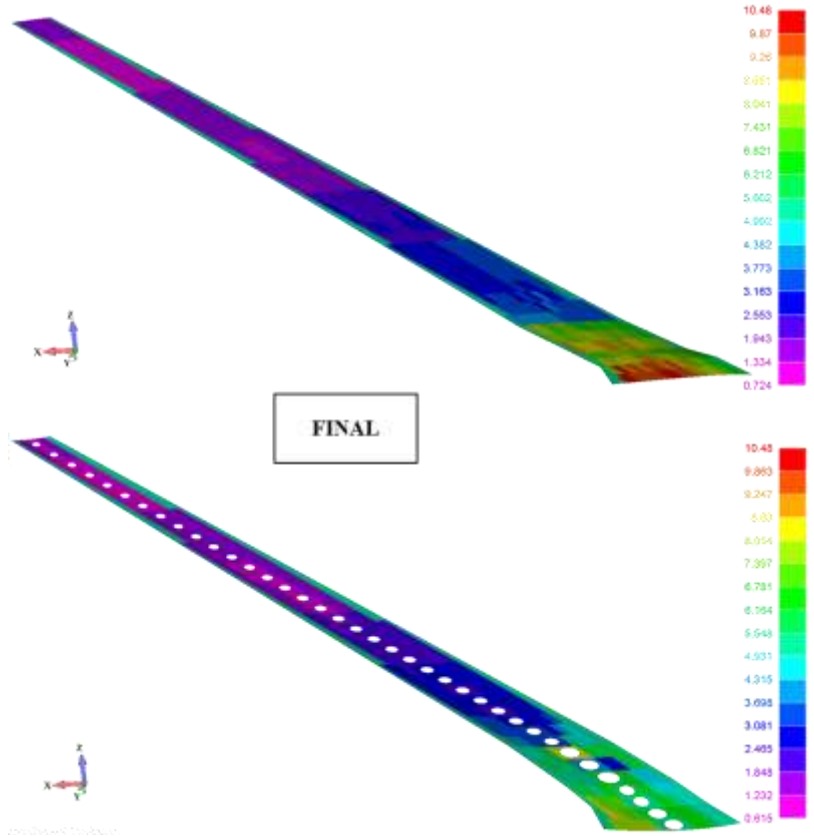

Figure 14. Final thickness map at upper and lower panels

Table 3 and Fig. 15 show the weight evolution after local buckling optimization at composite panels.

Table 3. Weight reduction after wing box panels buckling optimization

		WEIGHT EVOLUTION			
RUN	**DESVARs**	**REL BY OPT**	**REL BY INITIAL**	**ACCUMULATED**	**WALL TIME**
INITIAL	-	-	-	-	-
OPT 1	Bays 01-05	4.26%	4.26%	4.26%	01:09:01
OPT 2	Bays 06-10	-0.33%	-0.34%	3.92%	01:04:41
OPT 3	Bays 11-15	-3.24%	-3.37%	0.55%	00:05:32
OPT 4	Bays 16-20	-3.65%	-3.67%	-3.12%	00:04:32
OPT 5	Bays 21-25	-4.26%	-4.13%	-7.25%	00:06:45
OPT 6	Bays 26-30	-3.71%	-3.44%	-10.69%	00:01:41
OPT 7	Bays 31-35	-3.20%	-2.86%	-13.55%	00:01:35
OPT 8	Bays 36-37	-1.11%	-0.96%	-14.51%	00:00:46
OPT 9	All Bays	-1.00%	-0.85%	-15.37%	00:54:56

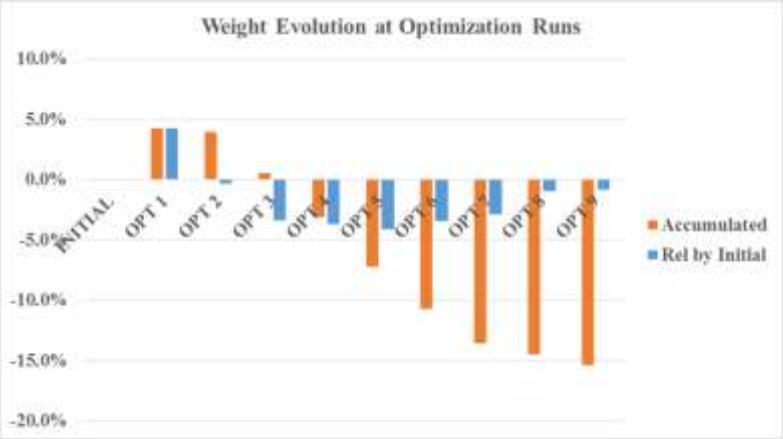

Figure 15. Weight evolution at Optimization Runs

6. CONCLUSION

The proposed semi-analytical criterion of local buckling calculation of composite panels presented a potential of weight reduction. However, this optimization analysis must be taken as a methodology evaluation, not as a sizing analysis restricted to the final reduction values. The load cases selection was made in an arbitrary way, with a simply simulation of limit up-bending and down-bending conditions.

In the same thinking, the results of panels' thicknesses at lower skin cutouts (at inspection windows) must be treated specially. Since this methodology takes the central element of the panel to extract the membrane stresses, to cutout regions this may be incorrect, due to non-consistent boundary condition in these regions and to the intrinsic stress concentration at windows edges which are not being considered correctly.

An important indirect result of this work is the method sensitivity. At the beginning, before all optimization runs, all wing panels were set with the same initial plies thicknesses (t_{ply}*=1.25mm*) and, by consequence, the same laminate thicknesses (t_{lam}*=5.00mm*). After all optimization runs, the solver led panels thicknesses to values much smaller than initial (from middle to tip bays) and to values considerably higher than initials (from root to middle bays). This behavior shows robustness characteristic, because even with different paths to course among wing box regions, the optimization was able to find a feasible design.

The structure management tool (structManager) is essential to keep the model manageable and to create optimization cards for complex models. The amount of necessary design information to perform an optimization run is impracticable of being generated manually.

The smeared laminate is a good initial approach to pre-optimize a composite structure. A plausible optimization sequence could be a pre-optimization using smeared laminate and this semi-analytical buckling criterion, because it presents less computational cost than NASTRAN® eigenvalue solution. With the whole structure pre-sized and optimized in a first level, a new type of detailed analysis and optimization could be applied to obtain an optimized structure to different types of analysis.

This is an embryonic suite of composite materials optimization and analysis tools, with several gaps to be filled by new blocks of methodologies of analysis and optimization criteria.

7. REFERENCES

Castro, S. G. P. "Semi-Analytical Tools for the Analysis of Laminated Composite Cylindrical and Conical Imperfect Shells under Various Loading and Boundary Conditions", 2014. 201f. Thesis (Doctorate of Sciences) – Faculty of Natural and Material Science, Clausthal University of Technology.

Castro, S. G. P. Computational mechanics (compmech)**.** [2016]. Version 0.5.5. Available at <http://compmech.github.io/compmech/>. Access in: February-10-2016.

Castro, S. G. P.; Abreu, G. P. P. StructManager. Version 0.2.0. Available at <http://saullocastro.github.io/structManager/>. Access in: February-29-2016.

Dassault Systèmes. **Catia V5-6R2015.**

MSC Software. "MD/MSC NASTRAN 2010 design sensitivity and optimization user's guide". Santa Ana, CA, 2010. **(**MD/MSC NASTRAN 2010)

8. RESPONSIBILITY NOTICE

The author is the only responsible for the printed material included in this paper.