Master Thesis

Optimisation in aircraft pre-design with sizing-criteria

Author[s]: Weiss, Daniel

Publication Date: 2006

Permanent Link: https://doi.org/10.3929/ethz-a-005198049

Rights / License: In Copyright - Non-Commercial Use Permitted
Optimisation in Aircraft Pre-Design With Sizing-Criteria

Daniel Weiss
Mechanical Engineering

Diploma Thesis Winter Term 2005/06
Ref.-No: IMES-ST 06-019

Centre of Structure Technologies
ETH Zurich

Tutor: Christof Ledermann
Professor: Prof. Dr. Paolo Ermanni
Acknowledgments

I would like to thank my tutor, Christof Ledermann, for his support during my work and for taking the time to answer my questions and to provide the information needed to successfully complete the actual work.

In addition, I would like to express my thanks to Oliver König (EVEN) who contributed a considerable amount to the actual work. The possibility to build the algorithm into his predefined boost graph environment saved time and allowed a clean and professional implementation of the sizing algorithm.

Finally, I would like to thank Professor Ermanni for the possibility to carry out my work at his institute and for providing the necessary infrastructure.
Abstract

Presently, in aerospace engineering, aircraft pre-design is a time consuming and highly iterative process due to a high connectivity between structure development, mass evaluation, loads and fluid dynamics. Especially during this process a vast amount of FE-analyses is required e.g. for mass optimisation. To minimize the calculation time, tools are required for efficient estimations without detailed data being already available. To allow an accurate estimation, preliminary optimisation (e.g. of shell thickness) is required. The results of a previously conducted feasibility study\textsuperscript{1} showed that it is generally possible to achieve good estimations by optimising structures considering simple and basic mechanical laws - the sizing rules. The basic idea is to optimise and hence precondition shell and stringer thicknesses to minimize the amount of needed FE-analyses and therefore to create fast convergence for FE-calculations. Based on simple sizing criteria like plasticity, buckling etc. and data from FE-analyses, the thickness is adapted until the decisive criteria is just met. The resulting data can be read by any FE-software to recalculate the new stress distribution. The implemented sizing criteria are partly based on the findings of a preceding diploma thesis\textsuperscript{2} which showed that already a few basic rules lead to remarkably close estimations of the shell thicknesses.

\textsuperscript{1}Weiss D. (SS 2005), Semester Thesis: \textit{Implementierung von Sizing-Regeln}, Centre of Structure Technologies, ETH Zurich.


Daniel Weiss
\( A_{A_{\text{stringer}}} \) Stringer cross section area \( [m^2] \)

\( a \) Long edge \( [m] \)

\( b \) Loaded or shorter edge of shell element \( [m] \)

\( \alpha, \gamma \) Inclination angle of buckling pattern generated by \( \tau_{xy} \). \( 45^\circ \) were chosen for a first approximation. \( [^\circ] \)

\( D \) Plate stiffness for isotropic plates \( \frac{E t^3}{12(1-\nu^2)} \) \( [N \cdot m] \)

\( E \) E-modulus \( \left[ \frac{N}{m^2} \right] \)

\( G \) Shear modulus \( \left[ \frac{N}{m^2} \right] \)

\( g \) \( \sum(cuts + free\ edges) \) in stringer cross sections. Needed for calculations with Gerard and Needham. \([-]\)

\( h \) Distance from shell neutral plane to stringer neutral axis \( [m] \)

\( I \) Smaller of both moments of inertia for stringer \( [m^4] \)

\( k \) Factor for critical stringer buckling stress (depending on supports) \([-]\)

\( k_x, k_{cx}, k_y, k_{cy}, k_{cxy} \) Values for critical buckling stresses \( \sigma_{\text{crit},x}, \sigma_{\text{crit},y}, \tau_{\text{crit},xy} \) \([-]\)

\( k_w \) Value for effective width; depending on \( \frac{\text{platewidth}}{\text{platethickness}} \) \([-]\)

\( l \) Barlength \( [m] \)

\( \lambda_{\text{Str}}, \lambda_{\text{SkStr}} \) Gyration radius for stringer and skin-stringer-structure \( [m] \)

\( \nu \) Poisson number \([-]\)

\( N_x, N_y, N_{xy} \) In plane forces (non-linear plate equation) \( [N] \)

\( P_x, P_y, F_x, F_y \) External forces or loads \( [N] \)

\( r_i, r_a \) Inner and outer radius for tubular stringers \( [m] \)

\( R, R_{\text{criteria}}, i \) Reserve factors (of \( i \)-th element) \([-]\)

\( R_{\text{buc},i} \) Reserve factors for corresponding buckling force \([-]\)

\( \sigma_{\text{crit}}, \sigma_{\text{crit, Skin-Stringer}} \) Critical buckling stress for stringer and skin-stringer structure \( \left[ \frac{N}{m^2} \right] \)

\( \sigma_{\text{crit, Stringer}} \) Critical local buckling stress for stringer flanges (calculated according to Gerard or Needham) \( \left[ \frac{N}{m^2} \right] \)

\( \sigma_{x, \text{crit}}, \sigma_{y, \text{crit}}, \tau_{xy, \text{crit}} \) Critical buckling stress for shell element \( \left[ \frac{N}{m^2} \right] \)

\( \sigma_{x, \text{crit}, \text{t}, \text{initial}}, \sigma_{y, \text{crit}, \text{t}, \text{new}} \) Critical buckling stresses before and after reducing shell thickness \( \left[ \frac{N}{m^2} \right] \)

\( \sigma_{\text{load},x}, \sigma_{\text{load},y}, \tau_{\text{load},xy} \) Equivalent to \( \sigma_x, \sigma_y, \tau_{xy} \) \( \left[ \frac{N}{m^2} \right] \)

\( \sigma_{\text{newLoad}}, \sigma_{\text{oldLoad}} \) Old load and new load after thickness change (for skin -stringer failure criteria) \( \left[ \frac{N}{m^2} \right] \)

\( \sigma_{\text{shell},x}, \sigma_{\text{shell},y} \) New shell load after converting shear into normal stress components \( \left[ \frac{N}{m^2} \right] \)

\( \sigma_{\text{skin, Stringer}} \) Skin-stringer load \( \left[ \frac{N}{m^2} \right] \)

\( \sigma_{\text{stringer}} \) Stringer load \( \left[ \frac{N}{m^2} \right] \)

\( \sigma_{\text{tot},x}, \sigma_{\text{tot},y} \) Total stress in stringer-effective width structure after shell load redistribution due to buckling \( \left[ \frac{N}{m^2} \right] \)

\( \sigma_{x, \text{crit}}, \sigma_{y, \text{crit}}, \tau_{xy} \) Normal stress and shear stress of FE-element according to the local element coordinate system \( \left[ \frac{N}{m^2} \right] \)

\( \sigma_b \) Ultimate stress \( \left[ \frac{N}{m^2} \right] \)

\( \sigma_{s, \text{yield}} \) Yield stress \( \left[ \frac{N}{m^2} \right] \)

\( \sigma_v,i \) Von Mises stress of \( i \)-th element \( \left[ \frac{N}{m^2} \right] \)

\( t \) Shell thickness \((t_{\text{new}} = \text{optimised thickness}, t_{\text{initial}} = \text{initial thickness}) \) \( [m] \)
Width of current edge where $w_{eff,i}$ lies, i.e. where load is attacking. [m]

Plate deflection. [m]

Effective width on left and right side of plate. The effective width of the stringer is $2 \cdot w_{eff,i}$, as the stringer lies on the border of two shells. [m]

Ultimate Load: Highest allowed stress. The structure may be deformed plastically though must not fail.

Limit Load: Ultimate load reduced by a factor of 1.5. No plastic deformations are allowed.

Sizing-Rules: Dimensioning criteria (plasticity, buckling etc.)
## Contents

1 Introduction 1

2 Task Description 2
   2.1 FE-Sizing Loop ............................................. 2
   2.2 Graph Theory .................................................. 3
   2.3 Task Summary .................................................. 3

3 Sizing Process Overview 4

4 Graph Theory 6
   4.1 General Boost Graph Concept ................................. 6
      4.1.1 Searching the Graph .................................... 6
   4.2 Boost Spirit Actors ............................................ 8
   4.3 Boost Architecture in the Sizing Algorithm ................. 9

5 The Algorithm’s Structure 11

6 User Defined Settings 12

7 Building the Graph Structure 13
   7.1 Parsing FE-Data ............................................... 13
   7.2 Building Buckling Fields ...................................... 13
      7.2.1 Automatic Detection .................................... 14
      7.2.2 Predefined Thickness .................................... 18
      7.2.3 Zone File ................................................ 19
   7.3 Corner Detection ............................................... 19
   7.4 Single FE-Element Optimisation .............................. 22
   7.5 Stress Allocation .............................................. 22

8 Sizing Criteria for Isotropic Materials 25
   8.1 Plasticity Failure for Shells ................................ 25
   8.2 Shell Buckling .................................................. 26
   8.3 Ultimate Failure for Shells ................................... 26
   8.4 Skin-Stringer Failure for Ultimate Load ..................... 27
   8.5 Predefined Strain for Shells ................................ 28
   8.6 Plasticity Failure for Stringers .............................. 29
   8.7 Stringer Buckling .............................................. 29
   8.8 Predefined Strain for Stringers .............................. 29

9 Sizing Criteria for Composite Materials 31

10 Data Output 32

11 Simplifications and Restrictions 35
   11.1 FE-stresses ................................................... 35
   11.2 Bending-Moments .............................................. 35
   11.3 Properties of ‘Super-Elements’ ............................... 35
   11.4 Buckling of Triangular elements ............................. 36
   11.5 Curved Shells ................................................ 36
   11.6 Cutouts ...................................................... 36
List of Figures

1  Thickness optimisation .......................................................... 1
2  Sizing process overview .......................................................... 4
3  Single elements: thickness fringe for yield failure ......................... 5
4  Buckling fields: thickness fringe for yield failure ......................... 5
5  Graph structure: vertices and bi-directional edges ......................... 6
6  Breadth first search (BFS) .......................................................... 7
7  Checking for neighbours on element edges .................................... 7
8  Shortest path problem (source: [4]) .............................................. 8
9  Spirit actors (assign, & pushback) in target grammar ...................... 9
10 Graph structure in sizing algorithm ............................................. 10
11 Overview: steps of sizing algorithm ............................................ 11
12 Example of a graphical user interface for settings .......................... 12
13 Allocation of FE-data to containers .............................................. 13
14 Overview: zone detection and construction .................................... 14
15 Graph structure before and after zone detection ............................. 15
16 Checking neighbours of source element ....................................... 15
17 'Super-Stringers' do not run over several supports (ribs) .................. 16
18 Border criteria: bars (yellow) and element inclination ...................... 16
19 Winglet ....................................................................................... 17
20 Winglet, partly sized .................................................................... 18
21 Defining buckling fields by applying different thicknesses ............... 19
22 Determining wrong corner node by comparing node vectors ............... 20
23 Angle calculation using plane normal vectors .................................. 20
24 Front surface of wing nose .......................................................... 21
25 Automatic separation of heavily bent regions .................................. 21
26 Smooth round corners .................................................................. 21
27 Determining the angle between three shell zone border nodes .......... 21
28 bdf file after zone optimisation .................................................... 22
29 bdf file after single FE-element optimisation ................................ 22
30 Stress peaks in single nodes ....................................................... 22
31 Possibilities to choose stresses for criteria calculations ................... 23
32 Stress allocation from *.f06 file to FE-elements and from FE-elements to zones ......................................................... 23
33 Transforming non-uniform cross section thickness to uniform thickness .... 29
34 Extract of *.elementData.txt file .................................................. 32
35 Wing thickness distribution in Tecplot (for plasticity criteria) ......... 32
36 Extract of data saved in *.zon-file ............................................... 33
37 Nodal stresses showing tension and compression in the same element (in y-direction) ................................................................................................................................................................. 35
38 3-stringer method ([9]) ................................................................. 37
39 Load case plate: biaxial compression, all edges simply supported ....... 39
40 Manual calculation results vs. results of the sizing algorithm for vonMises and Buckling criteria ................................................................. 39
41 Manual calculation results vs. results of the sizing algorithm for predefined strain criteria ................................................................. 39
42 Load case plate for skin-stringer failure: biaxial compression, all edges simply supported, edges reinforced by stringers ................................................................. 40
43 Verification of sizing results for skin-stringer failure criteria .......... 40
44 Verification of sizing results for stringer sizing criteria (stringers applied on plate edges) ................................................................. 40
45 Security factor plot (ESA-COMP) ................................................. 41
46 Wing: FE-mesh .............................................................................. 42
Real load case: elliptical lift force distribution
Initial thickness and vonMises distribution
Optimised thicknesses and stress distribution for plasticity (zones)
Optimised thicknesses and stress distribution for plasticity (single FE-elements)
Optimised thicknesses and stress distribution for plasticity (zones, max. stress allocation to fields)
Optimised thickness distribution for buckling (zones)
FE-geometry vs. sizer geometry
Buckling reserve factor development for wing panel
Additional stringers on wing box surface (0401_Wings_V10.CATPart)
Smaller zones due to additional stringers
Load case and loads for fuselage aft section
Predefined buckling fields (by defining different thicknesses) and initial vonMises stresses
Thickness and vonMises distribution after shell optimisation for plasticity criteria (ultimate load)
Thickness and vonMises distribution after optimisation with 5 highest FE-stresses for each zone
Optimisation for plasticity (only shells, 1 loop)
Optimisation for plasticity (shells and stringers, 1 loop)
Reserve after 1 FE-sizing loop (failure if > 1)
Reserve after 10 FE-sizing loops (failure if > 1)
Accumulation and compensation of normal stresses in ply fibres (± 45°)
Calculation time for pre-processing without and with zone detection (feasibility study [2])
Sizer calculation time vs. FE-solver calculation time
Convergence of vonMises stresses for FE-sizing loop (plasticity criteria, $\sigma_{yield} = 3.6 \cdot 10^6 \frac{N}{m^2}$)
Thickness characteristics for FE-sizing loop (plasticity criteria, 10 loops)
Mass development over 10 FE-sizing loops (plasticity, Wing_Strings.CATAnalysis)
Gaussian curve to average over a range of differently weighted stresses
Graphical user interface (sheet 1)
Graphical user interface (sheet 2)
Graphical user interface (sheet 3)
Graphical user interface (sheet 4)
Digitalized $k_c$-values for $k_{c,x}$ and $k_{c,y}$
Digitalized $k_c$-values for $k_{c,xy}$
$k_c$-values for rectangular plates (source: HSS [3], A. Rectangular Plates)
Calculation of $k_c$-values for buckling of triangular equilateral elements (source: HSS [3], B. Polygonal Plates)
Scheme to the iterative calculation of optimised thicknesses
Curve for $w_{eff}$ according to Niu [12], p. 144
Curve approximation for $w_{eff}$: curve-fitting for a polynomial of degree 5 in MATLAB
Plate scheme with stringers and effective width
Load redistribution after shell buckling (source: Bruhn [5], C7.10)
Euler-Johnson-Curves for stringer buckling in Transition Region (source: Bruhn [5], C7.22)
Structure and sequence of bulk data file information
Local coordinate system for QUDAD-elements
Local coordinate system for CTRIA-elements
8 node stresses (2 values per node for upper and lower element surface)
Stress allocation in f06-NASTRAN-file for 2D shell elements
<table>
<thead>
<tr>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>93</td>
<td>Stress allocation in <em>f06-NASTRAN</em>-file for beam elements</td>
</tr>
<tr>
<td>94</td>
<td>Nested map structure for the stress property of an <code>ELEMENT_VERTEX</code> or <code>ZONE_VERTEX</code></td>
</tr>
<tr>
<td>95</td>
<td><em>ESA-COMP</em> full report for ±45° laminate (1 FE-sizing loop)</td>
</tr>
<tr>
<td>96</td>
<td><em>ESA-COMP</em> full report for ±45° laminate (10 FE-sizing loops)</td>
</tr>
</tbody>
</table>
1 Introduction

During the aircraft pre-design process a large amount of iterative calculations is required to achieve an optimised structure. The mass estimation and hence a preliminary optimisation of shell thicknesses with only scarce data available form part of these calculations. The structures mostly being large FE-models, need a considerable amount of calculation time for each FE-analysis. One possibility to avoid or at least minimize these rather expensive calculations, is to precondition the FE-model by adapting shell and stringer thicknesses. For this kind of computations, some knowledge about the restricting mechanical criteria is needed, the sizing criteria. Several basic criteria have been set up, validated and documented in a preceding diploma thesis [1]. In addition, a feasibility study [2] has been conducted to prove the effectiveness and the accuracy of this method of preconditioning structures. The results were remarkably good and showed that a structure could be optimised for further FE-analyses with only a few simple mechanical rules, needing little time. The aim of the actual work was now to implement these sizing rules and to create an algorithm that could be used for any shell and stringer thickness optimisation problem, working autonomously as well as being integrable into evolutionary algorithms to precondition newly generated topologies.

The report is a documentation of the implemented sizing algorithm. It indicates reflections and calculations needed for the implementation, shows the algorithm’s application areas as well as its restrictions and simplifications. In addition, test calculations and results are listed along with certain possible extensions for future versions. The appendix contains a list of all files generated, a short handling instruction for the sizing program, additional derivations for sizing criteria and further calculation results.

![Figure 1: Thickness optimisation](image)
2 Task Description

The aim of the actual work was to build a sizing algorithm based on the experiences gathered during the feasibility study [2]. In order to accelerate the calculation time, to generate a more robust and autonomous algorithm and to implement new sizing criteria, an entirely new concept for the algorithm’s structure was chosen. The basic requirements for the algorithm were:

- reading data from FE-calculations,
- implementing at least the six sizing criteria defined by O. Häfner [1],
- writing the optimised thickness data into a format that can be read by any FE-software,
- possibility to easily add new sizing criteria
- ability to optimise single FE-elements as well as entire buckling fields consisting of several FE-elements,
- ability to recognise such buckling fields.

The mentioned requirements have been basically met by the previous feasibility study [2] though some of them were implemented in a rather rough manner and were not appropriate for a robust and effectively working algorithm. As a result, some further requirements had been set up and mark, along with some extensions, the task for the actual diploma thesis:

- efficient data acquirement from FE-files,
- implementation of additional sizing criteria for isotropic materials (e.g. for stringers),
- implementation of a basic criteria for composite structures for studies on the applicability of sizing criteria for non-isotropic materials,
- writing the optimised thickness data into a bulk data file that can be directly read by NASTRAN,
- inserting criteria into the algorithm as independent libraries,
- efficient, reliable and robust recognition of buckling fields,
- automated loop between sizing and FE-calculation.

2.1 FE-Sizing Loop

As the sizing algorithm obviously optimises the shell thicknesses only locally due to the local stresses provided by NASTRAN for each FE-element, at least one FE-recalculation is needed to redistribute the stresses throughout the structure. Carrying out manually these recalculations is highly time consuming and laborious. Basically, a simple call from the sizing algorithm is needed to start NASTRAN (if optimised data is written directly into a bulk data file). Automating this process would save precious time and would provide even more accurate results after a few recalculations.

For results and comments on the convergence of such repeated loops, see chapter 12.10.
2.2 Graph Theory

The main asset and a highly promising concept of the work at hand marked the Graph Theory. In order to accelerate data acquirement and processing an entirely different concept from vector and map variables used during the feasibility study \cite{2} was chosen. With the support of EVEN\textsuperscript{3} the framework was drawn to base the new sizing algorithm entirely on working with the vertices and edges of a Graph structure. Further description of this concept can be found in chapter 4.

2.3 Task Summary

Summarizing the tasks for the actual work, a robust, fast and highly reliable sizing algorithm had to be implemented that would be able to optimise shells and stringers as a stand-alone software as well as being integrated into existing topology- or other optimisation tools. As a result it would have to be able to recognise predefined buckling fields as well as finding them according to a few simple border criteria. The sizing criteria should be extended by stringer optimisation criteria and, if possible, by some composite shell sizing conditions.

The following devices and components were available during the project:

- PC Intel Xeon 2.8Ghz, 2GB RAM; WindowsXP Professional Version 2002, Service Pack 2
- Laptop Intel Pentium 4, 3.33Ghz, 1GB RAM; Suse 10.0
- Kdevelop 3.2.2
- linCVS 1.4.3
- CATIA V5 Version 5.15, Service Pack 3, Build Number 15
- CATIA V5 Version 5.16
- SimDesigner for CATIA V5, 2005 r3 MOD 003 WINDOWS
- Wing\textunderscore Paper\textunderscore CATAnalysis and 0401\textunderscore Wing\textunderscore Paper\textunderscore CATPart
- Wing\textunderscore Stringers\textunderscore CATAnalysis
- Fuselage\textunderscore PredefZones\textunderscore CATAnalysis
- some simple plate models for validation: Plate\textunderscore Validation\textunderscore Crit1-5\textunderscore CATPart, Plate\textunderscore Validation\textunderscore Crit1-5\textunderscore smallElm\textunderscore CATAnalysis, Plate\textunderscore Crit6\textunderscore CATPart, Plate\textunderscore Crit6\textunderscore CATAnalysis, Plate\textunderscore Validation\textunderscore Crit1-5\textunderscore comp\textunderscore CATAnalysis.

\textsuperscript{3}EVEN - Evolutionary Engineering AG, Zurich; http://www.even-ag.ch
3 Sizing Process Overview

Based on simple analytical and mechanical failure criteria, the idea of sizing structures outside FE-software leads to the following procedure: Data received from a preliminary FE-calculation is read and assigned to each element. Data includes resulting stresses from the applied load case (read from an f06-file), material data (e.g. E-modulus, G-modulus etc.), thickness and geometry - all read from a bdf-file.

Having assigned the information to each element, for all activated sizing criteria the minimally allowed thickness is calculated for each element. After evaluating the highest thickness of all sizing criteria the value is written into a bulk data file and/or different txt-files according to the settings in the settings file *.ini.

The bdf-file can be read by any FE-software (e.g. CATIA V5) or used by NASTRAN to calculate the new stress distribution (i.e. generating a new f06-file for a new sizing procedure).

![Sizing process overview diagram](image)

With the sizing algorithm carrying an additional function to determine buckling fields consisting of several FE-elements, there is the possibility to not only optimise single FE-elements but also to apply the sizing criteria to larger fields either predefined by the user or detected automatically according to some simple border criteria. The result of optimising entire fields are zones of FE-elements having all the same thickness and therefore building panels of realistic size that are manufacturable. In addition, applying buckling criteria to single FE-elements would be of little practical use.

Pictures 3 and 4 show the difference in thickness distribution between optimising single FE-elements and zones for yield failure under ultimate load.
The following sizing criteria have been implemented and validated:

1 & 2 - No plasticity failure for shells (for limit and ultimate load)
3 & 4 - No shell buckling (for limit and ultimate load)
5 & 6 - No plasticity for stringers (for limit and ultimate load)
7 & 8 - No stringer buckling (for limit and ultimate load)
9 & 10 - Predefined maximum strain for shells (for limit and ultimate load)
11 & 12 - Predefined maximum strain for stringers (for limit and ultimate load)
13 - No ultimate failure for shells (for ultimate load)
14 - No ultimate failure for stringers (for ultimate load)
15 - No total failure of skin-stringer-structure (for ultimate load)
16 & 17 - First ply failure for composite shells according to Tsai-Hill (for limit and ultimate load)

Limit Load describes a load case where no plastic deformations must occur. Ultimate Load is a nonrecurring load case. In this case plastic deformations may occur, however the structure must still not fail.

For all criteria, except for skin-stringer failure and stringer buckling, the according thickness can be calculated analytically. The skin-stringer failure has to be calculated iteratively due to the effective width depending on the thickness and the buckling stiffness, which depend both again on the effective width. The iteration process is interrupted if:

- maximum number of iteration steps is reached (defined in settings file)
- reserve factor lies between upper and lower limit (defined in settings file)
- minimum shell thickness is reached (defined in settings file)

The user has also the possibility to set the thickness steps by which the thickness is increased or decreased after each step.

The reason to implement the stringer buckling criteria iteratively, lies in the calculation of the gyration radius (for more details see chapter 8.7).
4 Graph Theory

The *Graph Theory* is a mathematical concept to efficiently describe and calculate highly linked problems like e.g. determining the shortest connection between two geographical points. Instead of running through the entire database, only the surrounding points (i.e. the points of interest) are checked and therefore the calculation effort is reduced considerably. The *Boost Graph Library* provides a software architecture to efficiently implement the *Graph Theory* in C++.

To reduce calculation time in general, to generate a highly reliable automatic buckling field detector and to provide a more robust framework and data storage the *Boost Graph*-concept was chosen for the sizing algorithm. With the *Boost Graph Library* providing a vast collection of extremely efficient functions for such problems and data handling in general, a concept was found that facilitated the processing of element data considerably.

To give a basic idea of the *Boost Graph*-concept, some simple thoughts and examples will be explained in the following paragraphs. A comprehensive description and further examples can be found in the *User Guide and Reference Manual* [4].

4.1 General Boost Graph Concept

The *Graph* concept in general is based on the idea of describing relations between neighbour objects. The entities are basically vertices and edges linking the vertices. The idea is to describe relations between neighbour entities. To gather comprehensive information it is therefore necessary to run from vertex to vertex through the whole graph structure. The edges linking two vertices can be undirected, bi-directional or unidirectional. For some problems like e.g. loop search, unidirectional edges are more appropriate than undirected edges. The direction is given by the order of the vertices in the edge list. For the sizing algorithm undirected edges have been used as no distinction between 'in' and 'out' edges is required.

![Graph structure: vertices and bi-directional edges](image)

For the *Boost Graph* architecture, data stored in a graph can be accessed by pointers and especially designed functions like *breadth_first_search*, *depth_first_search* etc.

Saving data in a *Graph* structure brings the advantage of being able to access neighbour vertices extremely fast (with the corresponding *adjacency* function). Compared to lists where for each neighbour check the entire list has to be run through, this structure already reduces access time beyond any possible optimisation for list and vector algorithms.

4.1.1 Searching the Graph

In addition to the fast neighbour access, the *breadth_first_search* (BFS) allows to systematically check every vertex just once and not having to run through the list several times. Starting at
a vertex, all vertices in the graph reachable from the one starting vertex are checked. It can be compared with a 2D-wave spreading from a source vertex (User Guide and Reference Manual [4], p.61). Nearer vertices are checked first, then the next 'level' is checked etc. (fig. 6). At the same time a visitor can be included. A visitor is a function called at a specific event during the search (e.g. when discovering an ELEMENT_VERTEX). Such visitors allow activating further operations like reading information from the discovered vertex, adding edges etc.

![Figure 6: Breadth first search (BFS)](image)

A simplified example comparing the method used for the feasibility study [2] and the BFS-method reveals the reduction in calculation effort (fig. 7).

![Figure 7: Checking for neighbours on element edges](image)

The difference being approximately of $O(n)$ calculations for the graph concept vs. $O(n^2)$ for standard lists and vectors. For each check the entire list has to be examined whereas the neighbours in the graph can be directly found over the linking edges.

A second possibility for a graph search method offers the depth first search (DFS) concept. Similarly to the BFS, the DFS runs systematically through all reachable vertices from a starting vertex. Though, instead of expanding the search like a 2D-wave, it travels as 'deep' as possible in one direction, then returns to the source, takes a different direction etc. This search type can be used for tree searches where the search travels always to the last leave, then returns to the
root and so on, until the entire tree has been checked. For the sizing algorithm the BFS was used for obvious reasons. Rather than checking a tree shaped graph, a neighbour search in a highly interconnected graph for each element is needed.

Typical graph problems are e.g. routing problems, website linkages, flight plan management etc. All these problems can be reduced to a simple shortest path problem, which can be solved with the graph structure. Taking the routing problem as an example (User Guide and Reference Manual [4]), the Boost Graph algorithm supplies functions that sum up all weights stored on the edges between the routers and therefore determine the shortest path (fig. 8).

![Figure 8: Shortest path problem (source: [4])](image)

### 4.2 Boost Spirit Actors

In addition to the Boost Graph Library, the Boost Spirit parsing concept was used. Spirit enables the generation of arbitrary parsers with individually adapted grammar to match specific file information as well as entire text blocks. At first sight, the implementation is slightly more complex than a simple file reader but the result is a highly reliable and effective parser, needing a few fractions of a second to parse even large files. In addition, the grammar can be generalized for various similar entries whereas for a normal file reader each keyword has to be defined. This flexibility is especially useful for bdf-files as different FE-Software produces slightly different data output.

One Problem to be mentioned is the different interpretation of line breaks in Linux and Windows software. The problem encountered when transferring spirit code from one system to the other was that perfectly working grammar in Linux did not read the complete data when running under Windows. The difficulty may be lying in the different symbols used by the two systems to define a line break. The errors occurring are that more data is read than desired or that entire text blocks are not read at all.

Another very effective feature of spirit is the simultaneous storage of data read during the parsing process. The variables where the desired data will be stored can be inserted into the target grammar of the parser. Spirit actors like push\_back\_a and assign\_a, which can also be individually defined, save the acquired data directly into the specified variables.
The Boost-concept provides numerous additional functions to define program options, data output, even matrix constructors etc.

4.3 Boost Architecture in the Sizing Algorithm

The functions used in this project are only a fraction of the possibilities of the Boost architecture. The task is not a specific graph problem, though it still benefits from Boost Graph’s advantages. The adjacency function is used in several functions to determine nodes of FE-elements, neighbor elements during zone detection and zone vertices for element properties, to mention a few. The breadth_first_search function is used for the buckling field detection and construction.

A partly unresolved problem during the feasibility study [2] was to find a method for a complete and as efficient as possible detection of element neighbours. The need to run through large element lists made this process extremely slow and could take up to 35 minutes. With the BFS, the process could be accelerated about a 1000 times, taking 2 seconds now for the same amount of elements (for further information about calculation time reduction see chapter 12.9).

The visitor used during the BFS includes the entire neighbour detection algorithm for zones. Each time an ELEMENT_VERTEX is found, its adjacent element vertices are checked if they belong to the same zone or not. If they belong to the same zone, an edge is drawn between the neighbour ELEMENT_VERTEX and the corresponding ZONE_VERTEX.

As the visitor does not allow changes in the graph being searched (for obvious reasons of confusing the BFS), a temporary copy of the graph structure has to be made during the BFS. After concluding the search, the changes are written back into the original graph.

For parsing, the Boost Spirit-concept is used. The bdf-file, having several similar text blocks providing the needed information, would require different keywords to distinguish all the paragraphs slightly differing from each other. Using the spirit concept, specific grammar for each text block type can be written and stored in a rule variable. Using these rules during the parsing process, saves time and provides a reliable and robust method to gather complete information.

The sizing algorithm creates a new graph structure each time a bulk data file is read. The topology of this structure was firstly chosen under consideration of the information needed and therefore adapting it to the access sequences to be performed during the sizing process. Secondly, the structure represents in its actual form the data blocks written into the new bulk data file.
Each zone represents a property set. All FE-elements that belong to the same property set are linked to such a zone by edges. In addition, as the property set represents a zone (i.e. a buckling field) there are nodes linked directly to the ZONE_VERTEX. These nodes represent the zone’s corner nodes.

All nodes are connected to their FE-element. That way, they can be found by a simple and fast adjacency search.

Furthermore, there are nodes linked with other nodes. These connections mark borders of buckling fields or nodes of bar/beam elements. The idea of this linkage is to find the corner or end nodes of buckling fields and 'super-bars' by simply running along the linked nodes and checking the angles (for buckling fields). For a detailed description of zone building see chapter 7.2

The information stored in each vertex is:

<table>
<thead>
<tr>
<th>Vertex Type</th>
<th>Data Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZONE_VERTEX</td>
<td>Material data, Topology, Stresses, further auxiliary data</td>
</tr>
<tr>
<td>ELEMENT_VERTEX</td>
<td>Stresses, thickness (needed for buckling field detection)</td>
</tr>
<tr>
<td>NODE_VERTEX</td>
<td>Coordinates</td>
</tr>
</tbody>
</table>

After the graph construction, the sizer will optimise in any case (single FE-element or zone optimisation) all ZONE vertices and not the ELEMENT vertices. If single FE-elements are to be optimised, each element will have been linked to a separate ZONE_VERTEX. This structure coincides with a resulting bdf-file where each FE-element has its own property set. If buckling fields are to be optimised, there will be zone vertices with several FE-elements linked to it. The idea of optimising zone vertices in any case is to make the sizing algorithm indifferent against whether single FE-elements or buckling fields are optimised. The sizier always recognises just single elements either with larger dimensions (zones) or smaller dimensions (FE-elements).

For FE-sizing loops (see 2.1), the entire graph structure is deleted after each loop and a new graph is built after parsing the bdf-file.
5 The Algorithm’s Structure

The algorithm’s steps can be divided into four modules:

- Sizing: calculating the optimised shell and stringer thickness according to the activated sizing criteria.
- Output: writing optimised data into files (bdf-, different txt-, xls- and dat-file according to the settings in the settings file).
- NASTRAN: recalculation of newly written bdf-file by calling NASTRAN to generate a new f06-file (if the option is activated).

![Diagram of the algorithm's structure]

Figure 11: Overview: steps of sizing algorithm

The following paragraphs will describe the modules, special functions, the sizing criteria and the output possibilities.
6 User Defined Settings

The settings are read and stored by functions defined in `program_options.cpp`.

To allow a close interaction between user and sizing algorithm, a settings file (*.ini) was created. All relevant settings like input and output paths, sizing criteria to be considered, calculation methods, zone or FE-element sizing and several other parameters can be individually set in this file before starting the sizing process. A comprehensive description of all setting possibilities can be found in appendix C. To provide a cleaner overview over the applicable settings a graphical user interface (GUI) was generated in Visual Basic (VB). The parameters are split into four different sheets (fig. 12).

The settings file has to be copied into the same folder like `MainSizing.exe`. As the algorithm reads all information from this file, the existence of this file is mandatory to run the sizing process. Using the GUI, this step is carried out automatically.

Some calculation parameters are deliberately made available to offer the user the possibility to adapt the algorithm to certain specific problems. E.g. for some slightly bent shells, the `borderAngle` (to determine whether a neighbour shell element has an inclination that is too high to belong to the same zone) may be altered to build larger or smaller zones.

![Figure 12: Example of a graphical user interface for settings](image)

The *Boost* architecture offers a `program_options`-function to read and store the defined options and to access specific information during the sizing process.

Generally, few parameters have to be set for common calculations. Though, as the actual sizing software is an experimental tool the user might want to optimise calculation results or to evaluate different optimisation methods. In this case several parameters can be altered and adapted manually to meet the user’s requirements.
7 Building the Graph Structure

The core files needed to start the sizing process are the bdf- and f06-files containing all relevant FE-data. The first step consists of reading and allocating the information of these two files to the graph structure.

7.1 Parsing FE-Data

The functions to read and allocate the FE-data are defined in bdf_io.cpp and f06_io.cpp.

For this task the Boost architecture offers a parsing method using spirit actors to determine the information to be read and to save it simultaneously. There are several *.hpp-files defining the corresponding grammar for the property-, element-, stress- etc. text blocks in the two files. The information is saved in several containers named nodes, elements, properties, stresses. The information stored in these containers is allocated to the corresponding ZONE_VERTEX, ELEMENT_VERTEX and NODE_VERTEX. The graph structure is built simultaneously to the data allocation.

![Figure 13: Allocation of FE-data to containers](image)

7.2 Building Buckling Fields

All zone detection functions are located in zone_detection.cpp.

The detection and construction of buckling fields (i.e. zones consisting of several FE-elements) is a particularly important function supplied by the sizing algorithm. There are several reasons why rather zones than single FE-elements are required to be optimised:

1. thickness distributions of single FE-elements are rarely manufacturable and will not be needed for industrial purposes,
2. buckling criteria should be applied to the physical buckling fields and not to single FE-elements that are part of an entire field,
3. the sizing algorithm is a tool for rough optimisation, therefore an optimisation of single FE-elements could as well pretend an accuracy impossible to meet with the given rough data.

Presently, there are three possibilities implemented for detection and construction of buckling fields.
7.2.1 Automatic Detection

The most important function is the automatic zone detection. The algorithm runs through the entire graph structure (using the BFS, see 4.1.1) and checks for each ELEMENT_VERTEX (i.e. for each FE-element):

- whether the ZONE_VERTEX (i.e. the property set) the element’s property-id is pointing at has already assigned FE-elements
- whether one of the element’s neighbours belongs to the same ZONE_VERTEX.

The scheme in fig. 14 shows the different steps of the zone detection algorithm located in zone_detection.cpp.

The idea is not to actually construct a buckling field but to merely link the corresponding FE-elements and nodes to the actual property set vertex (ZONE_VERTEX). Hence, the only changes in the graph structure after the zone detection process will be a few more edges linking ZONE, ELEMENT and NODE vertices together.
From the initial *bdf*-file there are already some existing property sets (i.e. zone vertices). Though, during zone detection there will be much more zones needed than already available property sets. Therefore the first step for each FE-element before checking its neighbours is to check whether a new property set has to be generated or not. The functioning of this check depends entirely on the idea of the BFS. If the source element has already been assigned to a zone, it has to belong to the zone/property set its property id (pid) is pointing at (when assigning a neighbour element to a zone, its pid is adapted too). If the element has not yet been assigned, the zone its pid is pointing at is checked. Meaning that if the zone has already assigned FE-elements, a new zone has to be created (with the same properties) for the actual element. As mentioned, this procedure only works because the BFS extends from neighbour to neighbour and therefore each element of the same zone has a neighbour that has already been checked. If the element has not been assigned it has to belong to a different zone.

Having assigned the source element to the right *ZONE_VERTEX*, its neighbours are checked. First, all neighbour elements lying on one of its edges (i.e. having two nodes in common) are searched. Possible elements are shells (*CQUAD, CTRIA*) or beams (*CBAR, CBEAM*).

If the source element is a beam, only beam neighbours are considered to be neighbours and checked for the following criteria:

- Same cross section type (for *PBEAML, BARL* property) or cross section area (for *PBEAM, PBAR* property)?
- Inclination angle between the two beams?
- Other perpendicular beams in the same node?

Therefore, if the inclination angle between the two beams is not too high, the cross section type is the same and no perpendicular bars lie in the same node, they belong to the same zone. In this case an *ELEMENT2ZONE* edge is built between the neighbour bar-*ELEMENT_VERTEX* and the corresponding *ZONE_VERTEX*. The additional check for any other bar elements lying in the same node
at a certain angle is carried out for the following reason. Wherever other bars cross the ‘path’ of a ‘super-stringer’ the stringer is supported in this node. As a result it is assumed that there will never be a stringer running over several supports without being actually supported (such a stringer’s critical buckling stress would be unnecessarily lowered). Following this assumption, the ‘super-stringer’-construction is interrupted as soon as a support is reached, ensuring that such stringer zones always lie between two supports. The same assumption then applies to the skin-stringer failure criteria (see 8.4).

Even though the buckling field’s edge runs from A to B (in fig. 18) the adjacent ‘super-stringer’, being supported by each rib’s stringers, is divided into smaller sections.

For shell source elements there are three border criteria to be checked:

- Same shell thickness?
- Inclination angle between the two shell elements?
- Beam element lying on the edge?

Because the presently implemented sizing criteria for shells do not consider thickness variations, the thickness in one zone has to be the same for all FE-elements. This criteria though could lead to zone merging if several FE-Sizing-loops are carried out as the thickness of two adjacent zones may suddenly be the same. For this reason, even if the options automatic zone detection or predefined zones are activated in the settings file, the zones will be detected automatically only the first time. At the same time, the zones are written into a file (*.zon) which will be used during all following loops. This ensures that always the same zones are optimised and prevents zone merging.

A high inclination angle marks a zone border too. This borderAngle can be altered in the settings file to adapt the sensibility in case of bent shells. The third border criteria considers beam FE-elements lying on the same edge and marking a zone border. The inclination and beam
border criteria can be turned off in the settings file. If a shell neighbour belongs to the same zone like the source element, it is assigned to the zone by building an \texttt{ELEMENT2ZONE} edge. If the actual edge marks a zone border, the shell element is not assigned, though the corresponding two nodes are linked by a \texttt{NODE2NODE} edge, which will be needed for corner detection later on. This procedure is repeated for each \texttt{ELEMENT}\_\texttt{VERTEX} in the graph structure (i.e. for each FE-element). The result is a graph structure with zone vertices marking the different buckling fields, element and corner node vertices linked to their corresponding \texttt{ZONE}\_\texttt{VERTEX} and nodes linked together if they lie on a border.

The automatic zone detection can be used if no zones have been previously defined or especially if the sizing algorithm is included in topology optimisation. During topology optimisation there will be a constant change of buckling zones as e.g. new ribs are created or shell areas are reduced. In this case a highly flexible, reliable and efficient zone detection algorithm has to be provided for fast buckling field construction and sizing.

The automatic detection has still some limitations concerning the recognition of complex shapes like closed surfaces and arbitrarily changing inclinations. This is linked to the fact that the current sizing criteria are not applicable to such structures. One example of such a rather complex shape would be the winglet shown in fig. 19.

![Figure 19: Winglet](image)

The algorithm will detect different fields that are more or less similar to a triangular or rectangular shape. These zones are treated and sized. Other regions having more than four corners are not sized (fig. 20).
Nevertheless, such a structure can be sized if the FE-model is adapted. The winglet shown in fig. 19 is a highly simplified FE-model and lacks any stringers or other reinforcements. If stringers are applied, the sizer will have the possibility to define buckling zones between them. Such zones can then be sized.

As soon as sizing criteria are implemented that consider closed cylindrical fields (i.e. fields without corners), additional buckling field detection criteria will have to be implemented to recognise such shapes.

### 7.2.2 Predefined Thickness

Considering the possibility to previously define buckling fields with FE-software, e.g. *CATIA V5*, the algorithm has to be able to recognise these marked fields and size them. Using the automatic zone detection algorithm as basis, an additional function offering this option was included into the sizing algorithm. The ‘numbering’ of zones is presently done by applying different thickness values to different element groups as there is currently no other useful possibility in *CATIA V5* to number the zones.

Having predefined property sets for each zone in the *bdf* file after the numbering process, the simplest way to assign the FE-elements to their corresponding zone/property vertex would require little effort as the elements could be assigned by simply checking their property-id. This method, being basically the method used during the feasibility study [2], has one major weak point. Using only the pid for FE-element allocation prevents the algorithm from distinguishing different zones with the same thickness. Taking into account that it might be easier for the user to reuse some thickness values for several (non-adjacent) zones, the automatic detection algorithm was used again for predefined zones. Working with the BFS-method, the algorithm is now able to distinguish different zones with the same thickness (as long as they are separated by zones having different thickness values). Because of the very efficient *Boost Graph* functions, the additionally needed fractions of a second for this detection do not affect the overall calculation time considerably.

The difference to the initial automatic zone detection lies only in the applied border detection criteria. Instead of using all three criteria, for predefined thicknesses, only the thickness criteria is used.
7.2.3 Zone File

After running the automatic or predefined zone detection it is possible to write the zones into a txt-file (by enabling the WriteZoneFile option in the settings file). Initially implemented during the feasibility study [2] to reduce the zone detection time from 35 minutes down to 2 minutes, the actual main purpose of this file is to prevent zone merging (see 10) and to merely offer the user a possibility to manually alter zones. The calculation speed reached with the Boost concept leaves almost no difference in time between building zones automatically and reading them from a file.

For each zone, the zone number, its FE-elements, the detected corner nodes and stringers lying on shell borders are written into the file (fig. 36). If there is a zone that has an unusual number of corners (i.e. more than 4 nodes) or that should be merged with another zone, the user can simply alter the corresponding data in the *.zon-file and enable the FileZones option in the settings file. The algorithm then reads the predefined zones and corners from the file.

7.3 Corner Detection

If zones are to be sized, after allocating all corresponding FE-elements the FE-nodes marking the zone’s corners have to be detected (except if zone information is read from an existing zone file). The definition of a corner node implies some assumptions that have to be checked before adding the node as a corner. The basic idea is that the angle between three nodes leading around a corner will be of approximately 90 degrees (the range of deviation can be set in the settings file). Using this criteria, the corners of all fields meeting the requirements for the presently implemented sizing criteria are assigned. The requirements are:

- rectangular or triangular shape
- no curvature (bent shells etc.)
Taking into account that in FE-models few zones do entirely meet the mentioned criteria, some features to broaden the acceptance of slightly distorted zones were implemented:

To check the angle between three nodes, a different approach from simply checking the angle between the edges was chosen to bypass the problem occurring with a 3D curvature. If the curvature is higher than the limit set for cornerAngle, the node would be set as corner node (angle $\beta$ in fig. 22) even though being perfectly in line with its neighbours (angle $\alpha$).

![Figure 22: Determining wrong corner node by comparing node vectors](image)

The solution was found by comparing rather planes (i.e. their normal vectors) than the vectors between the three nodes (fig. 23). Though the problem with distorted adjacent elements still remains. In this case, the normal vectors (n1, n2) too differ in their third direction and therefore inflict an additional inclination. Solving this problem by simply projecting the third node into the plane of the first element would require to know in which direction the plane normal is pointing. As the FE-elements can be arbitrarily turned upside down, the determination of the normal vector’s direction would require additional calculations.

![Figure 23: Angle calculation using plane normal vectors](image)

In this case effort and results are questionable. As the implemented sizing criteria are strictly spoken only applicable to perfectly plane rectangles and equilateral triangles there is no need to detect and size such distorted zones. In case of automatic zone detection such strongly bent regions are divided into separate zones (fig. 25).
The one still remaining problem lies in detecting zones that come close to rectangular or triangular shape but that have smooth enough corners to lie lower than the critical corner detection angle (fig. 26). This problem could be overcome by applying a ‘compass’ variable (14.4).

To find all FE-nodes lying on a zone border, the nodes on such a border and the nodes linking beam elements are connected by \texttt{NODE2NODE}-edges during the zone detection. Using these connections, the algorithm runs along the shell zone border calculating each angle between three nodes (fig. 27). For bars, the algorithm stops if no more edges bearing the actual zone number can be found. The node then marks one of the beam’s end nodes.

End nodes of ‘super-beams’ or corner nodes of shell zones are linked directly to the corresponding \texttt{ZONE VERTEX} by \texttt{ZONE2NODE}-edges.
7.4 Single FE-Element Optimisation

In case of single FE-element optimisation no buckling fields have to be determined. Though, considering the structure of a bdf-file where each FE-element has its own thickness and hence its own property set, a few changes similar to a zone detection have to be made. In this case, for each FE-element a separate **ZONE_VERTEX** is created. The **ELEMENT_VERTEX** is linked to its property set as well as its nodes. Basically, there will be as many zone vertices as there are FE-elements, the zones simply consisting of only one FE-element. Apart from coinciding entirely with the NASTRAN bdf format, this process has the advantage of treating the data equally; no matter if zones or single FE-elements are optimised. The algorithm therefore always optimises zones with the only difference that in one case each zone consists of only one, in the other case, of several FE-elements.

Figure 28: *bdf* file after zone optimisation

Figure 29: *bdf* file after single FE-element optimisation

7.5 Stress Allocation

During the feasibility study [2] the importance and fragility of a correct stress assignment to the elements were highlighted. The amount of stress data provided by the *f06*-file for each FE-element is rather large and in order to save memory, only part of this data is read by the parser. Generally, there are stresses calculated in each element node and the average of all nodes. This information is provided for the upper and the lower side of the element, leading to a total of 30 stress values for a **CQUAD**-element (4 nodes, mean stress, in x-, y-, xy-direction, for upper and lower side).

Currently, only the average stresses of all nodes are read (for upper and lower side) and stored in the **stresses** container. This already reduces the amount of information considerably but leads at the same time to the fact that local stress peaks in an FE-element will not be considered during the sizing process (fig. 30).

![Figure 30: Stress peaks in single nodes](image)

During the sizing process, the user has two possibilities to use the stress information:

- sizing with the mean stress of upper and lower side of each element
- sizing with the absolute maximum of the two stresses of upper and lower side of each element

\[
\begin{align*}
\sigma_{\text{u, load}} &= \frac{\sigma_{\text{u, max}} + \sigma_{\text{l, max}}}{2} \\
\tau_{\text{u, load}} &= \frac{\tau_{\text{u, max}} + \tau_{\text{l, max}}}{2} \\
\sigma_{\text{l, load}} &= \text{MAX} \left[ \sigma_{\text{u, max}}, \sigma_{\text{l, max}} \right] \\
\tau_{\text{l, load}} &= \text{MAX} \left[ \tau_{\text{u, max}}, \tau_{\text{l, max}} \right]
\end{align*}
\]

Figure 31: Possibilities to choose stresses for criteria calculations

Considering that, with bending load cases the upper stress may be positive and hence the lower negative, it would not be wise to activate the 'mean' stress option, which may result in almost zero element stress. Therefore the user is advised to choose 'max' stress whenever possible. In this case, the highest stress (being positive or negative) is taken to size the structure. Though, being an acceptable solution for plasticity criteria, this simplification may affect the results of buckling sizing criteria (see 11.2).

In addition to the stress mode choice for the sizing criteria, there are also two possibilities for the FE-element to zone stress allocation. While the zone with only one FE-element (in case of single element optimisation) directly receives the stresses saved in the stress container, the buckling field consisting of more than one FE-element receives a combination of several FE-stresses. In this case, the user has the additional possibility of either allocating the average stress calculated out of all corresponding FE-elements to the zone or to take the maximum stress of a defined range of elements and calculate the mean of these few maxima (fig. 32).

Some problems remaining especially when optimising zones are:

- taking the maximum of one element leads to over-sizing
- taking more than one element for maximum or average calculation may extinguish some internal stresses
- single node stress peaks will not be considered (neither for single element nor for zone optimisation)
- Internal bending in FE-elements is not considered (e.g. pressure on upper side, tension on lower side of same FE-element)

- Internal bending in zones is not considered due to internal stress extinction when calculating averages

Even though leaving the mentioned problems still unresolved, the results achieved with the currently implemented methods (see chapter 12) show a fast convergence and therefore improving this module of the algorithm has been postponed to a later version.

Having concluded the parsing, data assignment and - if required - the zone construction part, the graph structure now contains all information and connections needed for the sizing procedure.
8 Sizing Criteria for Isotropic Materials

After having built the graph structure and after allocating all data to the corresponding vertices and edges, the sizing criteria are called to calculate the optimised thicknesses. Each sizing criteria can be turned on or off in the settings file *.ini.

To achieve a modular structure where sizing criteria can arbitrarily be appended or removed, each criteria has been written into a separate *.hpp-library file. The libraries are called one by one from the main function MainSizing.cpp. Most criteria are designed to handle limit load as well as ultimate load cases. The distinction is simply made by handing over different conversion factors when calling the library from MainSizing.cpp (1.5 for limit and 1.0 for ultimate load).

The sizing criteria are all applicable to 2D-shell elements (CQUAD4, CQUAD8, CTRIA3, CTRIA6) or, with some restrictions, to 1D-bar/beam elements (CBAR, CBEAM) accordingly. Elements (especially entire buckling fields) with other than rectangular or triangular shape will not be sized. Neither will be any 3D-elements.

The currently implemented sizing criteria will be roughly explained in the following paragraphs. Further details can be found in appendix D.

Generally, in each sizing criteria, the optimised thicknesses for all load cases are calculated (the *f06-file can provide resulting FE-stresses for different load cases for each FE-element). The highest thickness is then stored for each element (i.e. zone) along with the corresponding load case. The reserve factor which has to be reached for each criteria is set by the Lowerlimit setting in the *.ini-file (or in the GUI: 'Lower Limit of Reserve Factor').

Each sizing criteria loops through the graph, sizing each element. This means that for e.g. three different criteria, the entire graph is searched three times instead of applying each sizing criteria to the actual element before sizing the next element (only one graph search in total). The reason to do multiple graph searches is to provide a certain independence between the different criteria. At some point it might be useful that the four adjacent stringers of a panel are sized before the panel itself is optimised. Therefore all stringers have to undergo the treatment of a stringer criteria before the actual criteria is applicable to the shell. The calculation effort is obviously increased with this method, though taking into account that few criteria will ever be activated at the same time (limit load criteria will have no effect if the same criteria for ultimate load is activated too), the additional time will scarcely be noticed.

8.1 Plasticity Failure for Shells

In this criteria, the thickness of each shell-element/zone is optimised to prevent plastic deformation under limit or ultimate load. The thickness is calculated analytically by simply solving the vonMises equation for ductile materials:

\[ \sigma_{v,i} = \sqrt{\sigma_x^2 + \sigma_y^2 + 3\tau_{xy}^2 - \sigma_x \sigma_y} \leq \frac{\sigma_{yield}}{R} \]

R is the reserve factor (Lowerlimit in the *.ini-file). Solving the equation for the thickness yields:

\[ t_{new} = \sqrt{\frac{t_{initial}^2 \cdot R^2}{\sigma_{yield}^2} \left( \left( \frac{\sigma_x}{factor} \right)^2 + \left( \frac{\sigma_y}{factor} \right)^2 + 3 \left( \frac{\tau_{xy}}{factor} \right)^2 - \left( \frac{\sigma_x}{factor} \right) \left( \frac{\sigma_y}{factor} \right) \right)} \]

With \( \sigma_{yield} \) being the maximally allowed yield stress and factor being the transformation factor (1.5 if transformation from Ultimate to Limit Load is needed). Both values can be set in the settings file *.ini.
8.2 Shell Buckling

Here, the shell thickness is optimised for buckling failure under limit or ultimate load. Any stringers attached to the element/zone are not considered, i.e. this criteria considers only the shell without any reinforcements.

According to Hafner [1], chapter 6 and HSS [3], combined loads can be divided into unidirectional load components and the reserve factors can be added up.

\[
\sigma_{x,\text{crit}} = k_{cx} \cdot \frac{\pi^2 E}{12(1-\nu^2)} \left( \frac{t}{b} \right)^2 \\
\sigma_{y,\text{crit}} = k_{cy} \cdot \frac{\pi^2 E}{12(1-\nu^2)} \left( \frac{t}{a} \right)^2 \\
\tau_{xy,\text{crit}} = k_{sxy} \cdot \frac{\pi^2 E}{12(1-\nu^2)} \left( \frac{t}{b} \right)^2
\]

\(k_{cx}, k_{cy}, k_{sxy}\) k-values for corresponding shell dimensions (according to HSS [3], see appendix D.1).

a, b edge where load is attacking - or with shear stress, shortest edge.

The total reserve factor then being:

\[
\frac{1}{R_{\text{tot},i}} = \frac{1}{R_{\text{buck},x,i}} + \frac{1}{R_{\text{buck},y,i}} + \frac{1}{R_{\text{buck},xy,i}}
\]

\[
\frac{1}{R_{\text{tot},i}} \geq \frac{-\sigma_{x,\text{load initial}} t_{\text{initial}}}{t_{\text{new}} \sigma_{x,\text{crit}} \cdot \text{factor}^2} + \frac{-\sigma_{y,\text{load initial}} t_{\text{initial}}}{t_{\text{new}} \sigma_{y,\text{crit}} \cdot \text{factor}^2} + \left( \frac{-\tau_{xy,\text{load initial}} t_{\text{initial}}}{t_{\text{new}} \tau_{xy,\text{crit}} \cdot \text{factor}^2} \right)^2
\]  \[1\]

\(\frac{1}{R_{\text{tot},i}}\) is \(\frac{1}{R}\) (Lower limit in *.ini-file).

To consider the stabilizing effect of tension loads, and to receive positive values for compression, \(-\sigma_{x,\text{load}} \sigma_{x,\text{crit}}\) etc. is inserted. Hence, with only tension loads attacking on the shell, \(\frac{1}{R_{\text{buck},x,i}}\) yields a negative value and therefore contributes to a lower total value of \(\frac{1}{R_{\text{tot},i}}\), which is equal to a higher reserve \(R_{\text{tot},i}\).

By inserting the equations for critical buckling load \(\sigma_{x,\text{crit}}\) etc. into \([1]\), the following equation scheme is obtained for the optimised thickness:

\[
t_{\text{new}} = \frac{2 \pi}{A \cdot \sqrt{A^2 + 4B}}
\]

With:

\[
A = \frac{12(1-\nu^2) \cdot R}{\pi^2 E} \left[ \frac{-\sigma_{x,\text{load initial}} t_{\text{initial}} b^2}{\text{factor} \cdot k_{cx}} + \frac{-\sigma_{y,\text{load initial}} t_{\text{initial}} a^2}{\text{factor} \cdot k_{cy}} \right]
\]

\[
B = \frac{12^2(1-\nu^2)^2 \cdot R}{\pi^4 E^2} \left[ \frac{\tau_{xy,\text{load initial}} t_{\text{initial}} b^4}{\text{factor}^2 \cdot k_{sxy}^2} \right]
\]

The \(k_{\cdot}\) values are calculated directly over a help function included in the MainSizing.h header file. Details especially concerning triangular elements can be found in the appendix D.1.

8.3 Ultimate Failure for Shells

The only difference for this criteria from plasticity failure being the maximally allowed stress, the plasticity failure library can be reused. This time with the \(\text{factor}\) being 1.0 (Ultimate Load) and \(\sigma_{\text{ultimate}}\) instead of \(\sigma_{\text{yield}}\). \(\sigma_{\text{ultimate}}\) can be set in the settings file. The altered equation is:

\[
t_{\text{new}} = \sqrt{\frac{t_{\text{initial}}^2 \cdot R^2}{\sigma_{\text{ultimate}}^2} \left( \left( \frac{\sigma_x}{\text{factor}} \right)^2 + \left( \frac{\sigma_y}{\text{factor}} \right)^2 + 3 \left( \frac{\tau_{xy}}{\text{factor}} \right)^2 - \left( \frac{\sigma_x}{\text{factor}} \right) \left( \frac{\sigma_y}{\text{factor}} \right) \right)}
\]
8.4 Skin-Stringer Failure for Ultimate Load

This criteria defined by Häfner [1] calculates the thickness considering the entire skin-stringer structure. The previous criteria have considered only the shell itself, disregarding any stringer reinforcements. This criteria now takes into account that stringers lying on the shell borders may still prevent a total failure even though the shell is already buckling. Currently only rectangular elements are optimised. The optimisation of triangular elements is more complicated as stringers lying on two edges will not be parallel to each other and will therefore lead to more complex critical buckling stress distributions.

The thickness is adapted iteratively due to the effective width depending on the thickness which again depends on the effective width and the critical buckling stress for the entire structure (which also depends on \( w_{eff} \) and \( t \)).

According to Häfner [1] and Bruhn [5] it is assumed that the shell itself is able to bear loads only up to \( \sigma_{crit,buck} \). The rest of the load will be redistributed onto the stringers and the effective width (see fig. 86 in D.2.4). Note: for combined load cases, \( \sigma_{crit,buck} \) is not \( k_c \cdot \frac{\pi^2 E}{12(1-\nu^2)} \left( \frac{1}{t} \right)^2 \) but a lower stress that does not exceed the buckling reserve factor \( \frac{1}{R_{tot,i}} = \frac{1}{R_{buckx,i}} + \frac{1}{R_{bucky,i}} + \frac{1}{R_{buckxy,i}} \) in combination with the other loads. Therefore, the thickness for simple shell buckling is calculated and the applied loads are updated for the new thickness before starting the skin-stringer optimisation. The adapted loads represent the critical buckling stresses to start the process. After each iteration step, these critical stresses and the loads have to be adapted again. For a scheme describing the entire iteration process, see appendix D.2.1, fig. 82.

The following paragraphs will give a rough overview over the calculations carried out during one iteration step. Further details regarding formulas and points to be considered can be found in the appendix D.2.

The effective width can be calculated by: \( w_{eff} = t \sqrt{\frac{k_c E}{\sigma_{crit,\text{stringer}}}} \) (Häfner [1]). Hence, the stringer’s critical buckling stress as well as the \( k \)-values for the effective width are needed. \( \sigma_{crit,\text{stringer}} \) is being calculated with Euler-Johnson (Häfner [1] and Bruhn [5]):

\[
\sigma_{crit} = \sigma_{cripp} - \sigma_{cripp}^2 \left( \frac{(\frac{\lambda}{\lambda_{Str}})^2 k}{4\pi^2 E} \right) \tag{2}
\]

- \( l \) length of the stringer
- \( \lambda \) gyration radius of the stringer \( \lambda_{Str} = \sqrt{\frac{I}{A_{\text{Stringer}}}} \)
- \( I \) smallest moment of inertia of the stringer
- \( k \) geometry factor depending on the supports of the stringer (here \( k=1 \), as stringer is assumed to be simply supported; see Häfner [1], chap.6)

An important point to mention is that for the stringer length, not the edge length of the buckling field is taken but the length of an adjacent ‘super-stringer’. The ‘super-stringer’ may be shorter than the actual edge length due to intermediate supports (see chapter 7.2.1, fig. 17). To calculate \( \sigma_{cripp} \), Gerard’s method is applied (for further details see appendix D.2.6).

Considering the convergence of the Johnson-curve width the Euler-curve (appendix D.2.5), it can be observed that the two curves coincide at \( \sigma_{crit} = \frac{1}{2} \sigma_{cripp} \) (after Bruhn [5]). Therefore, for \( \sigma_{crit} \leq \frac{1}{2} \sigma_{cripp} \) the critical Euler-stress instead of \( \sigma_{crit,Eul-John} \) is taken to calculate the effective width.

Now, the effective width has been determined and the critical failure stresses as well as the
redistributed stresses can be calculated. A detailed calculation of the stresses to be applied to the new effective width ($\sigma_{x,i}$ and $\sigma_{y,i}$) can be found in appendix D.2.4.

The final redistributed load in the effective width and stringer structure amounts to:

$$\sigma_{\text{SkinStringer}} = \frac{\sigma_{x,i} \cdot w_{\text{eff}} + \sigma_{\text{Stringer}} \cdot A_{\text{Stringer}}}{A_{\text{Stringer}} + t \cdot w_{\text{eff}}}$$

- $\sigma_{x,i}$: new load after redistribution because of buckling; to be carried by the effective width
- $w_{\text{eff}}$: $2 \cdot w_{\text{eff},i}$, as stringer lies between two plates and therefore receives the effective width $w_{\text{eff},i}$ from each plate.
- $A_{\text{Stringer}}$: stringer cross section area

The calculation of the new critical failure stress for the combined structure (effective width and stringer) after shell buckling is identical to equation [2], though the stringer gyration radius $\lambda_{\text{Stringer}} = \sqrt{\frac{l}{A_{\text{Stringer}}}}$ is now substituted by the gyration radius of the combined structure. Bruhn [5], C7.26 suggests the following approach:

$$(\frac{\lambda_{\text{SkinStringer}}}{\lambda_{\text{Stringer}}})^2 = 1 + \left(1 + \left(1 + \left(\frac{h}{A_{\text{Stringer}}}ight)^2\right)^2\frac{w_{\text{eff}}t_{\text{skin}}}{A_{\text{Stringer}}^2}\right)^2$$

- $w_{\text{eff}}$: $2 \cdot w_{\text{eff},i}$, as stringer lies between two plates and therefore receives the effective width $w_{\text{eff},i}$ from each plate.
- $h$: distance from shell neutral plane to the stringer’s neutral axis

The reserve factor is therefore given by:

$$R = \frac{\sigma_{\text{crit,SkinStringer}}}{\sigma_{\text{SkinStringer}}} \geq 1$$

To apply the actual criteria, the following restriction has to be considered: There is no sense in calculating the optimised shell thickness if the stringer is too weak to even support the actual load. In such a case, the plate thickness would be increased until the stress is sufficiently low to be taken up by the skin-stringer structure. Though, as it normally should be the stringer’s function to stiffen the plate and not vice versa, the algorithm will increase the plate’s thickness only up to the optimised thickness for ‘buckling under ultimate load’.

### 8.5 Predefined Strain for Shells

A further criteria apparently needed for structure evaluation is the predefinition of a maximum strain. The user sets the maximum allowed strain in the settings file and the algorithm optimises the shell thickness accordingly. The equation is simply derived from Hooke’s law for plane stress:

$$
\varepsilon_x = \frac{1}{E} \left( \frac{\sigma_x}{\text{factor}} - \nu \frac{\sigma_y}{\text{factor}} \right) \quad \varepsilon_y = \frac{1}{E} \left( -\nu \frac{\sigma_x}{\text{factor}} + \frac{\sigma_y}{\text{factor}} \right)
$$

With the new stress being $\sigma_{\text{new}} = \sigma_{\text{old},\text{factors}}$, the equation can be solved for $t_{\text{new}}$:

$$
t_{\text{new},\varepsilon_x} = \frac{t_{\text{old}} \cdot R}{E \cdot \varepsilon_x} \left( \frac{\sigma_{x,\text{old}}}{\text{factor}} - \nu \frac{\sigma_{y,\text{old}}}{\text{factor}} \right) \quad t_{\text{new},\varepsilon_y} = \frac{t_{\text{old}} \cdot R}{E \cdot \varepsilon_y} \left( \frac{\sigma_{y,\text{old}}}{\text{factor}} - \nu \frac{\sigma_{x,\text{old}}}{\text{factor}} \right)
$$

For shells, obviously both, $\varepsilon_x$ and $\varepsilon_y$ are of interest. The algorithm therefore calculates both values and chooses the thickness resulting from the more critical direction (i.e. the higher thickness). Again, the criteria is applicable for limit load as well as for ultimate load, depending on the factor handed over from MainSizing.cpp when calling the function.
8.6 Plasticity Failure for Stringers

To allow a complete optimisation of the entire FE-structure for yield stress, this criteria was additionally appended to the sizing algorithm. The equation is similar to the formula mentioned in 8.1 though with the difference of stringers being one-dimensional elements and therefore approximately yielding a uniaxial stress condition. The equation then being:

\[
t_{\text{new}} = \frac{|\sigma_x| \cdot t_{\text{initial}} \cdot R}{\text{factor} \cdot \sigma_{\text{yield}}}
\]

8.7 Stringer Buckling

For critical stringer buckling stress optimisation the approach of Euler-Johnson [5] is used:

\[
\sigma_{\text{crit}} = \sigma_{\text{cripp}} - \sigma_{\text{cripp}}^2 \left( \frac{\lambda_{\text{Str}}}{4\pi^2E} \right)^2
\]

The crippling stress is calculated according to Gerard (see appendix D.2.6), where the thickness is included.

Similar to the skin-stringer buckling criteria, if \( \sigma_{\text{crit}} \leq \frac{1}{2}\sigma_{\text{cripp}} \), \( \sigma_{\text{crit}} \) is calculated with the Euler-equation.

With NASTRAN providing several different cross section types (see E.3), the difficulty to determine the thickness of a stringer arises. The problem was solved by simply calculating a mean thickness out of the provided cross section dimensions, using this thickness to optimise the stringer (i.e. inserting it into Gerard’s formula) and to redistribute the optimised thickness value to the single dimensions. This simplification leads to stringer cross sections with the same thickness in all its sections (fig. 33).

![Figure 33: Transforming non-uniform cross section thickness to uniform thickness](image)

In the Euler-Johnson-formula, the gyration radius \( \lambda_{\text{gyr}} = \sqrt{\frac{I}{A}} \) is included. \( I \) is the smaller of both moments of inertia. Because for complex cross section types \( I_1 \) can one time be higher than \( I_2 \) and with other thicknesses it could be vice versa, the two moments of inertia have to be compared after each thickness change. Therefore, the stringer buckling optimisation is carried out iteratively and not analytically.

8.8 Predefined Strain for Stringers

The predefined stress criteria for stringers can be reduced to a uniaxial stress condition:

\[
\varepsilon_x = \frac{\sigma_x}{E \cdot \text{factor}}
\]
The new thickness then being:

\[ t_{\text{new}} = \frac{t_{\text{old}} \cdot R \cdot \sigma_{x,\text{old}}}{E \cdot \varepsilon_{x} \cdot \text{factor}} \]

The included factor allows again optimisation for limit load and ultimate load:
9 Sizing Criteria for Composite Materials

To consider the rising interest in applying composite materials to lightweight structures, a first attempt to size such components was made by adding a further criteria especially designed for shells with PCOMP-properties.

NASTRAN distinguishes different PCOMP entry possibilities in the bulk data file. The mode is described by an entry in the PCOMP property text block (BLANK, SYM, MEM, BEND, SMEAR or SMCORE; see NASTRAN’s user’s guide [16] for further description).

Currently, only the BLANK-option is implemented, meaning that each single ply of the laminate has to be specified in the PCOMP section of the bulk file. To consider the remaining possibilities, simply an additional function, which converts the information into a fully described composite shell would have to be added (e.g. that the data given in the bulk file describes only one half of a symmetric ply etc.).

For the criteria, the first thought would probably be an optimisation considering the global stiffness matrix for stress redistribution after adapting the plies, hence leading to an iterative calculation (adapting ply thickness, calculating new stiffness, recalculating local stresses etc.).

Though there is a simpler approach, leaving the stress redistribution entirely to the FE-analysis which has to be carried out anyway to redistribute stresses on the global structure after sizing (see 12.10). This reduces the complexity of the composite sizing criteria considerably and enables a completely analytical optimisation of each ply.

As NASTRAN already provides all stresses for each single ply, the calculation of the optimised thickness is carried out similarly to the plasticity failure for isotropic shells, using the Tsai-Hill criteria. Apparently, Tsai-Hill (or a conservative form of the Puck criteria) is widely used in aerospace construction to include enough security in the calculations.

\[
\left(\frac{\sigma_x}{X \cdot \text{factor}}\right)^2 + \left(\frac{\sigma_y}{Y \cdot \text{factor}}\right)^2 - \left(\frac{\sigma_x \sigma_y}{X^2 \cdot \text{factor}^2}\right) + \left(\frac{\tau_{xy}}{St \cdot \text{factor}}\right)^2 \leq \frac{1}{R}
\]

The stress for the new thickness is again \(\sigma_{\text{new,ply}} = \sigma_{\text{old,ply}} \cdot \frac{t_{\text{old,ply}}}{t_{\text{new,ply}}}\). The equation can therefore be solved for \(t_{\text{new,ply}}\):

\[
t_{\text{new,ply}} = \sqrt{t_{\text{old,ply}}^2 \cdot R \cdot \left(\left(\frac{\sigma_x}{X \cdot \text{factor}}\right)^2 + \left(\frac{\sigma_y}{Y \cdot \text{factor}}\right)^2 - \left(\frac{\sigma_x \sigma_y}{X^2 \cdot \text{factor}^2}\right) + \left(\frac{\tau_{xy}}{St \cdot \text{factor}}\right)^2\right)}
\]

\(X, Y\) if \(\sigma_x\) is negative, \(X\) corresponds to the critical compression stress \(X_c\)
\(\) if \(\sigma_x\) is positive, \(X\) corresponds to the critical tension stress \(X_t\).
\(St\) critical shear stress.
\(R\) reserve factor (Lowerlimit in *.ini).

Considering that normally only specific ply thicknesses are available on the market, the sizer runs an additional check after having optimised all plies of the actual zone. The user can define a plyStep in the *.ini-file to force the sizer to only apply the available thickness steps.

\[
\text{remainder} = \frac{t_{\text{optimised}}}{\text{plyStep}}
\]

If the remainder is 0.5 or higher, the optimised thickness is increased to the next higher available value.
10 Data Output

The algorithm offers different output possibilities for the calculated thicknesses and other data. The following formats can be individually chosen by setting the according parameters in the settings file *.ini (*elementData.txt, *tec.dat and *catia.txt are not available if composite sizing criteria are activated):

- ***_elementData.txt**: This file serves mainly to check if all data information has been read and allocated to the zones/element. It lists all properties belonging to the according zone/element that will be optimised.

```
ZONE   1  TYPE    THICKNESS   PROP 0.005  1  7.6e-010  9.5e-007  0.746  2710  0.594  3193  -2.319e+007  6.7737e+007  -6.3159e+007  1e+007
```

Figure 34: Extract of *_elementData.txt file

- ***_tec.dat**: To allow a fast and simple display, a dat-file can be created that allows thickness fringe display in Tecplot. Thicknesses are not mapped onto elements but onto their nodes. This means that for each node, the mean thickness of its adjacent elements is calculated.

Figure 35: Wing thickness distribution in Tecplot (for plasticity criteria)

- ***_catia.txt**: As a remainder of the feasibility study [2] this file has been kept as a further output possibility. The format is adapted to CATIA V5 for individual thickness mapping for each zone (i.e. in the end, for each single FE-element). This rather time consuming method is now obsolete as the optimised thicknesses are directly written into the bdf-file.
• *thickPlot.xls: During loop calculations (FE-sizing loop) the user may want to monitor thickness changes of zones, e.g. to check convergence etc. This file offers the possibility to create simple Excel graph diagrams by listing the zone numbers and their thicknesses after each iteration step. For composite shells, the thickness of each ply is written out along with the zone number.

• *stressPlot.xls: For the same reason a stress plot file can be written to create convergence plots of the zone stresses. For each zone the vonMises stress is written along with the zone number. For composite shells again, the stress data is written for each single ply of the actual zone.

• *.bdf: Designing the actual algorithm especially for FE-sizing loops, a faster method to recalculate the FE-model with the new thicknesses had to be found. As only a NASTRAN recalculation is needed to provide a new f06-file for the next sizing loop, the obvious approach was to simply alter the current bdf-file. Therefore, after each sizing loop, the new thicknesses are written into the property set section. After the first iteration step, possibly new zones and hence new property sets will have been created. Therefore, the algorithm rewrites the element text block as well (re-allocating new property-ids and property headers) and adds new property sets to the property text block. It should be considered that the new data is written into the old bdf-file. The user may therefore be advised to keep a copy of the original bdf-file by disabling the deletebdf option in the settings file. If a copy of the previous file is left, the numbering follows NASTRAN’s numbering of the *.f06-files: the newest file bears no index, the oldest is named *.f06.1 (hence *.f06.2 being newer than *.f06.1).

• *.zon: Initially, this file was created to save time during zone creation (instead of repeating every time the zone detection process, the zones found during the first run were written into the file and could be read during the following runs, see 7.2.3). The algorithm now being fast enough, saving time with this method is not the primary aspect anymore. The function was still kept though to offer the user the possibility to change, add or delete zones manually. The user can remove single FE-elements, merge zones and allocate different zone corner nodes from the already assigned ones. It has to be considered though that only rectangular or triangular shaped zones with the corresponding number of nodes can be sized. An extract of the data saved in *.zon is shown in fig. 36.

![Figure 36: Extract of data saved in *.zon-file](image-url)

The second, more important use is preventing zone merging during FE-sizing loops. The thicknesses of two neighbouring zones (being a zone detection criteria) may become the same over several FE-sizing loops. Especially for detection of predefined zones the algorithm will not be able to longer distinguish the zones and will therefore merge them to one larger...
buckling field. To prevent such mergings, the zones found during the first sizing run are written into the *.zon-file, which is then used during the following loops.

- **_*massPlot.xls:* To keep track of each zone's mass development, an additional output possibility was created. The file contains the number and mass of each zone (stringer or shell). Starting with the initial mass, for each sizing loop the new value is written into a new column. At the bottom of each column the total mass of the entire structure is displayed.

- **_*reserve.xls:* To evaluate the accuracy of the resulting thicknesses, at the end of the set number of sizing loops the reserve factor of each zone is calculated according to the newly distributed stresses.

A close look at the resulting factors reveals that even if the stress and thickness convergence may be rather fast (see 12.10), reserve factors of individual zones may not always converge with the same efficiency. This is due to several different factors. First, even small changes of neighbour element thicknesses may cause considerable stress flow changes. A second cause may simply be that the minimally allowed shell thickness was set too high. Therefore the reserve factor will not further decrease as the thickness remains constant.

Especially for buckling criteria, there is an additional fact to be considered. As the calculation of the reserve factor is done by combining all normal and shear stresses, it may occur that before sizing, the components erase each other, e.g. resulting in a slightly negative reserve (i.e. tension, no buckling). In this case, the sizer reduces the thickness until the reserve becomes positive (mostly due to increasing shear stress). After stress redistribution it may occur though that there is now a high compression load which will yield an extremely low reserve factor considering the previously calculated thickness for tension load. It may therefore take a few more steps to equalize these fluctuations. Generally, when optimizing shells and stringers the necessary number of FE-sizing loops to reach a reasonable reserve factor will be higher than in the case of only optimizing shells (due to higher stress flow changes).

It has also to be considered that *NASTRAN* may redistribute the stresses differently between shells and stringers. The skin-stringer-buckling criteria assumes that the remaining stress after shell buckling is divided equally between the effective width and the stringer cross section. *NASTRAN* seems to use a different distribution method, which yields higher deviations from the originally calculated reserve factor. The effect appears to increase with more complex structures. For a single plate, the reserve factor is exactly reached after no more than four sizing loops.

Instead of writing the optimised thicknesses into the different file formats, the user has also the possibility to:

- **Output zone numbers:** if all sizing criteria are turned off, the algorithm assigns zone numbers instead of thicknesses. This function may be used to simply receive a color map of the different zones when loading the new *bdf*-file into an FE-program.

- **Map decisive criteria:** the user may be interested in the criteria leading to the optimised thickness of a particular zone. In this case, additionally to the desired criteria, the feature *CriteriaMapping* can be activated and instead of the thicknesses, the number of the decisive criteria will be mapped. The number corresponds to the numbering in the settings file and the graphical user interface.
11 Simplifications and Restrictions

The following paragraphs will give an overview over the simplifications that have been made to reduce calculation time and to simplify the sizing process in general. The result of these reductions and of the restricted amount of implemented sizing rules are some restrictions to the application of the actual sizing algorithm.

11.1 FE-stresses

As already mentioned in 7.5, to reduce the amount of data to be read from the *f06*-file, only the average stresses from all corners are read. This simplification leads to some peak stresses in single nodes left unconsidered during the sizing process. The result is a less conservative optimisation that yields some stresses above the initially allowed yield or buckling stress. Though considering that in any case, one optimisation run does not lead to the ideal result as stress redistribution through the whole model is not taken into account during a sizing run, this effect can be neglected. During several FE-sizing loops, all stresses are more and more equally distributed and hence the peaks in single nodes will decrease too.

11.2 Bending-Moments

Examining the data provided by the *f06*-file, it can be observed that even bending-moments inside single FE-elements can occur. An example is given in fig. 37: in y-direction, there is a compression stress (< 0) on the upper side of the element and a tension stress on the lower side, which is clearly the result of a bending moment attacking on the FE-shell.

For yield stress criteria, this distinction does not matter as the absolute values are used. A problem occurs though for buckling criteria. A bending-moment obviously increases the danger of buckling. Currently, this additional bending moment is not considered in the implemented buckling criteria, i.e. just the maximal load is taken. An improvement would be to adapt the formulas used in the buckling criteria to such combined loads.

11.3 Properties of 'Super-Elements'

Apart from the simplified stress allocation for 'super-elements' (chapter 7.5), some generalizations are made during property assignment. The algorithm assumes that along a zone/element edge only one type of beams is attached and that the 'super-stringer' runs from one corner to the next. For the skin-stringer failure criteria therefore, the properties of the adjacent 'super-stringer' are taken and considered to apply for the entire stringer on the actual edge. The same applies to 'super-shells': the properties of the first FE-element are assigned to the zone and all other neighbour FE-elements belonging to the same zone (by adapting their property-id). This simplification was made due to the currently implemented sizing criteria. They are all applicable only to elements with uniform properties (thickness, cross section area etc.). The idea behind this restriction comes from the basic field of application for the sizing algorithm: the algorithm was designed to be used during the earliest stages of aircraft design, where only rough data is available and such details as variable cross section areas are rarely modelled.
11.4 Buckling of Triangular elements

The currently implemented buckling criteria apply only to \texttt{CQUAD}- and \texttt{CTRIA}-shaped elements. Though, there are further restrictions for \texttt{CTRIA}-elements. In HSS [3], \textit{B. Polygonal Plates}, the corresponding formulas for simply supported triangles under combined pressure and shear load are given to calculate $\sigma_{\text{crit}}$ (see appendix D.1). Theoretically, these equations apply only to exactly equilateral triangles.

For single FE-element optimisation this would not pose a considerable problem as in most carefully built meshes the \texttt{CTRIA}-elements differ little from this ideal form. The larger difference will occur when optimising triangular-shaped zones consisting of several FE-elements and therefore seldom matching the shape of an exactly equilateral triangle.

At the moment, there is no data describing the deviation from theoretical results when sizing such deformed triangles. Though, considering the rough data available at this early design stage, these differences may possibly be neglected as long as there are no extremely distorted elements. Obviously the same reflections apply to deformed rectangular zones.

11.5 Curved Shells

In addition to strictly rectangular and triangular shapes, the actual sizing criteria assume that the shells are not curved. Generally, even small curvatures have a considerable effect especially on shell buckling. This effect has been partly investigated by Hänner [1] regarding the implementation in sizing criteria. It was found that the (de-)stabilizing effect of a curvature in one direction was in fact considerable, though that a second curvature in the other direction would nearly compensate the effect of the first curvature. For this reason, curvature is still not considered in the actual shell buckling criteria.

An even higher impact of curvature in structures will be found in 1D-stringers, especially spars in fuselage structures. These bendings are not considered in the stringer buckling criteria. To minimize the error, the user can adapt the \texttt{BorderElmAngle} in the settings file. If the value is near to zero, zones with almost no curvature will be created (as soon as the curvature is too high, a new zone will be created). This angle applies to stringer zones (‘super-stringers’) as well as to shell zones. In case of the fuselage spars, they will be divided in smaller stringers when lowering the \texttt{BorderElmAngle}.

11.6 Cutouts

Even though cutouts being present in many aircraft structures, panels with cutouts are still not considered in the actual sizing criteria. One reason is that in the early stages of aircraft design few cutouts are already included in the structure. Mostly, zones with such spaces are modelled with different mean thicknesses. This fact leads to the second point: even if cutouts were modelled, there would be some thicker regions around the borders making the shell highly anisotropic.

There will have to be designed special sizing criteria to handle such anisotropic cases. A method explained in \textit{Leichtbau II} [9] is the 3-stringer-method where the stringers lying on the panel are also included in the calculation (fig. 38).
An alternative approach may be using empirical values or estimations based on such values. Including such data in the sizing process could already lead to good estimations, though some further analysis may still be required.
11.7 Round Corners

The already mentioned problem in 7.3 of zones having round corners is still unresolved i.e. the required additional routines have not yet been implemented. The corner detection, being based on the fact that a corner node lies in an angle of nearly 90°, does not recognise corners that are built by very small FE-elements, each one including an angle less than the required corner angle. The solution is, as mentioned before, to accumulate these small angles e.g. until an amount of 45°. If 45° are reached then the node is marked as corner node.

11.8 Stiffened Plates

The skin-stringer failure criteria assumes that stringers lie only on shell borders and that the shell element itself is not stiffened by further stringers lying between the border stringers. If such orthotropic plates are to be considered, a new sizing criteria will have to be added considering the additional stiffening effects. HSS [3] gives some suggestions for particular constellations and stringer cross section forms.

In addition to a new sizing criteria it will be necessary that the algorithm checks every border node (of buckling fields consisting of several FE-elements) if there are stiffening stringers. For the time being such rather complex structures are not considered.

11.9 Stringer Thickness

Due to Gerard operating with uniform thicknesses in the entire stringer cross section, stringers with different thicknesses in their cross section are currently treated as sections with uniform thickness. This means that from the initially read dimensions, a mean thickness is calculated and used to size the stringer. This generalization of the thickness may lead to different moments of inertia and hence a different critical buckling stress from the data read from the *.bdf-file. Furthermore, to rewrite the PBEAM, PBAR, etc. property sets into the *.bdf-file, the optimised thickness has to be converted back into the single dimension values (dim1, dim2 etc., see E.3). The thickness value is equally distributed onto the dimensions, generating a stringer with uniform thickness.

These conversions may seem very rough. Though, considering that in this early stage of structural design few stringers will already have exact cross section characteristics, this simplification might be acceptable. Nevertheless, for future versions it could be useful to add criteria that consider such structures.
12 Validation and Results

The sizing criteria have been validated using a simple plate (files Plate\_Validation\_Crit1-5\_smallElm.CATAnalysis and Plate\_Crit6.CATAnalysis).

To check the reliability of the different zone detection functions, simplified models of a wing and a fuselage (dimensions similar to an Airbus-A330 aircraft) have been used (files Wing\_Paper.CATAnalysis, Wing\_Stringers.CATAnalysis and Fuselage\_PredefZones.CATAnalysis).

12.1 Validation: Isotropic Plate Model

- Plate\_Validation\_Crit1-5.CATPart, Plate\_Validation\_Crit1-5\_smallElm.CATAnalysis.

- Plate\_Crit6.CATPart, Plate\_Crit6.CATAnalysis.

To check the optimised thicknesses, a simple plate model was generated in CATIA V5. The simple geometry (fig. 39) allows manual calculation of the vonMises, buckling and predefined strain criteria.

<table>
<thead>
<tr>
<th>Material (Aluminum)</th>
<th>Dimensions</th>
<th>Load</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E = 7 \cdot 10^{10} \frac{N}{m^2}$</td>
<td>$a = 0.1m$</td>
<td>$F_x = 10000N$</td>
</tr>
<tr>
<td>$\nu = 0.346$</td>
<td>$b = 0.2m$</td>
<td>$F_y = 10000N$</td>
</tr>
<tr>
<td>$G = 9.5 \cdot 10^7 \frac{N}{m^2}$</td>
<td>$t = 0.002m$</td>
<td>$\varepsilon_{max} = 0.3$</td>
</tr>
</tbody>
</table>

![Figure 39: Load case plate: biaxial compression, all edges simply supported](image)

As the sizing process initially optimises every single FE-element and hence the buckling criteria would be applied to single FE-elements, all elements were assigned to one plate zone with the zone detection algorithm. As a result, the plate was sized as one single element. The results yielded the expected accuracy, confirming the correct implementation of the sizing criteria.

![Figure 40: Manual calculation results vs. results of the sizing algorithm for vonMises and Buckling criteria](image)
To verify the stringer and the skin-stringer structure criteria, stringers were applied to the plate. All FE-shell-elements were again assigned to a single plate zone.

### Table 1: Material Properties

<table>
<thead>
<tr>
<th>Material (Aluminum)</th>
<th>Dimensions</th>
<th>Load</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 2: Stringer and Skin-Stringer Failure Criteria

<table>
<thead>
<tr>
<th>Fig. 43: Verification of sizing results for skin-stringer failure criteria</th>
</tr>
</thead>
</table>

### Table 3: Stringer Buckling Criteria

<table>
<thead>
<tr>
<th>Fig. 44: Verification of sizing results for stringer sizing criteria (stringers applied on plate edges)</th>
</tr>
</thead>
</table>

The small deviations are due to NASTRAN’s stress distribution between shells and stringers and the calculation method of the thickness for skin-stringer failure and stringer buckling. While the optimisation for vonMises and buckling criteria is carried out analytically, the results for these criteria are achieved by iteratively reducing the thickness. As the algorithm stops as soon as it lies between upper and lower limit of the reserve (defined in the settings file), the algorithm will rarely reach an exact factor of 1. In the current case, a reserve factor slightly higher than 1 was reached, which explains the higher thickness of the sizing algorithm calculation. The deviations...
for the analytically calculated criteria result from NASTRAN’s stress allocation for shells and stringers. Obviously NASTRAN does not equally distribute the loads onto stringers and shells.

12.2 Validation: Composite Plate Model

To validate the Tsai-Hill composite criteria, a composite property set was applied to the plate used for the vonMises and buckling criteria (file Plate\_Validation\_Crit1-5\_comp.CATAnalysis). The laminate’s data are as follows:

<table>
<thead>
<tr>
<th>Material (T800 woven)</th>
<th>$X_1, Y_1 = 8.34 \cdot 10^8 \frac{N}{m^2}$</th>
<th>$X_2, Y_2 = 6.3 \cdot 10^8 \frac{N}{m^2}$</th>
<th>$St = 1.23 \cdot 10^8 \frac{N}{m^2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E_1, E_2 = 6.2 \cdot 10^{10} \frac{N}{m^2}$</td>
<td>$\nu = 0.04$</td>
<td>$G_{12} = 4.39 \cdot 10^9 \frac{N}{m^2}$</td>
<td></td>
</tr>
<tr>
<td>$a = 0.1m$</td>
<td>$b = 0.2m$</td>
<td>$t_{ply1} = 0.23mm$</td>
<td>$t_{ply2} = 0.23mm$</td>
</tr>
<tr>
<td>$F_x = 10000N$</td>
<td>$F_y = 10000N$</td>
<td>angles: $\pm 45^\circ$</td>
<td>$-$</td>
</tr>
</tbody>
</table>

To calculate the resulting security factor for the optimised ply thicknesses, the plate was built in ESA-COMP and the corresponding failure analysis was run. The applied loads are distributed unequally to the two plies by the FE-solver, leading to different thickness distributions:

<table>
<thead>
<tr>
<th>Ply</th>
<th>Angle</th>
<th>Thickness</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$-45^\circ$</td>
<td>0.135mm</td>
</tr>
<tr>
<td>2</td>
<td>$45^\circ$</td>
<td>0.11mm</td>
</tr>
</tbody>
</table>

The security factor calculated by ESA-COMP is:

![Security factor plot (ESA-COMP)](image)

The full report is attached in the appendix G.

12.3 Application: FE-Wing Model

- Wing\_Paper\_CATAnalysis and 0401\_Wing\_Paper\_CATPart
- Wing\_Stringers\_CATAnalysis and 0401\_Wings\_V10\_CATPart

To verify the algorithm’s functions with more complex geometries, the calculations were carried out with an FE-wing model. In particular, the reliability of the zone detection algorithm could be tested with this geometry. The first model (0401\_Wing\_Paper\_CATPart) is strongly simplified and yields too high thickness distributions for the buckling criteria in its initial form as stringers between the ribs are not modelled (see 12.5).
To receive more realistic thickness values, the model \(0401\_Wings\_V10.CATPart\) was created later on where now stringers between the ribs were modelled to cut the initial buckling zones into smaller fields.

The applied load case corresponds to half the lift force distribution needed for the compensation of two times the aircraft weight (2g). In this case: 250'000kg distributed on 72m\(^2\) (3.472 \(\cdot\) \(10^4\) \(\frac{kg}{m^2}\)). For simplification, instead of the elliptical distribution (fig. 47, real load case), a linear approximation was chosen (fig. 48). The wing is fixed at its base.

The initial thickness distribution and the resulting vonMises stresses are shown in fig. 49.

The following paragraphs will present and discuss results for the plasticity and buckling criteria under \textit{Ultimate Load}. Further pictures of thickness and vonMises distributions are displayed in appendix H.
12.4 Application: Plasticity Under Ultimate Load

After sizing the wing model with the plasticity criteria for shells and stringers under ultimate load, the thickness distribution is similar to a common beam under bending load. Close to the fixed support the moment reaches its maximum, hence the highest thickness values are applied here. The vonMises stresses are already more uniformly distributed. In fig. 50 the results for the optimisation of buckling fields are displayed.

![Figure 50: Optimised thicknesses and stress distribution for plasticity (zones)](image)

The vonMises stresses are already more uniformly distributed though they are generally too high. This is due to the fact that the sizer optimises locally and does not consider global stress redistribution during the process. Therefore after the first loop, the thicknesses are generally too thin. Some regions near the winglet with particularly high stress values can still be distinguished. The reason leading to these peak values is the simplified stress allocation. The difficulties for stress assignment to FE-elements and to buckling fields mentioned in 7.5 lead to single nodes bearing peak stress values. The mean element/zone stresses may lie below or near the allowed yield stress ($3.6 \cdot 10^8 \frac{N}{mm^2}$) though, single FE-elements or rather single nodes may carry higher values. This effect may be amplified in this region by the winglet which is only partly optimised due to its rather complex geometry (see fig. 20).

In addition, on the large surface behind the wing box some non-uniform stress distributions can be spotted. The reason being the fact that the whole surface is considered to be one single zone (due to the strongly simplified FE-model). Hence, during the calculation of the mean stress for the zone out of the single FE-element stresses, a large region is taken for the average stress, which leaves highly deferring peak values unconsidered. Additionally, the optimised thickness is applied to all FE-elements of the zone and therefore will be inappropriate for all FE-elements bearing stresses different from the calculated mean value.

Considering the optimisation with single FE-elements instead of zones, the mentioned regions show already lower values. In this case only the simplified stress allocation from the $f06$-file leads to the still appearing nodal stress peaks. The simplification being to leave aside single node stresses provided by the $f06$-file and only reading the mean element stresses.
Running the sizing criteria with the mean value of the 5 maximum FE-element stresses of each zone leads to higher thicknesses and the recalculated vonMises stresses show lower values. The peak stresses near the winglet vanish completely as can be seen in fig. 52.

12.5 Application: Buckling Under Ultimate Load

An interesting detail can be observed in the narrow buckling field located at the wing bend. As the ribs below the wing box surface are placed closer together in this region the risk of buckling is diminished. Even if the sharp bend may yield higher stresses, the optimised thickness lies below the adjacent buckling fields.
To receive an idea of the accuracy of the buckling criteria, a single panel was analysed manually by calculating the buckling reserve with the new thickness after redistributing the FE-stresses on the optimised model. The chosen panel is bent and has a trapezoidal shape. In such a case the algorithm calculates the mean edge lengths and sizes a plane rectangle whose dimensions lie between the largest and the shortest corresponding FE-edge (fig. 54).

The reserve factor after the first sizing loop lies far below 1 (fig. 55), again for the same reason of not considering global stress redistribution during the sizing process. Though as can be seen after the second loop, the reserve factor is already close to 1. Generally, the sizer seems to yield reserve factors slightly lower than 1, being therefore less conservative.

The buckling criteria works with rough simplifications. As shown in fig. 54, a perfectly rectangular shell is created by taking the mean geometry values of a distorted field. The resulting thickness distribution will therefore be different from the required value for the actual FE-model.
The rather unrealistic thickness values may attract some attention as well. To understand this result, it has to be considered that the simplified FE-model 0401_Wing_Paper.CATPart lacks the stringers in the upper and lower wing box surface. In reality, the currently rather large buckling fields are divided into smaller fields by several small stringers running on the surface (fig. 56).

![Figure 56: Additional stringers on wing box surface (0401_Wings_V10.CATPart)](image1)

The resulting thicknesses with the adapted FE-model (0401_Wings_V10.CATPart) yield more realistic values.

![Figure 57: Smaller zones due to additional stringers](image2)

### 12.6 Application: Predefined Buckling Fields

The FE-model used in the following calculations is fuselage_stringers.CATAnalysis. To prove the applicability of the predefined zone detection algorithm and to evaluate the sizing criteria on more complex structures, the FE-model of a fuselage aft section was optimised. The load case was chosen as:
Figure 58: Load case and loads for fuselage aft section

Figure 59: Fuselage geometry

The passenger weight was calculated according to: Passenger Weight: 85kg, Area/Passenger: 0.64m², resulting in roughly 1300 N/m². The load case does not include security against load factors higher than 1, i.e. it is a simple static load case without consideration of any flight and landing manouvers leading to higher loads. The rather low loads result in very low shell thickness values for the plasticity criteria.

The front section is fixed and the back section is supported by a linearly distributed force on the circumference, equalling \( \frac{2}{3} \) of the total passenger load in the aft section (simulating the lift of the elevator unit). The defined zones are:

Figure 60: Predefined buckling fields (by defining different thicknesses) and initial vonMises stresses
The comparison of the vonMises distribution before and after one optimisation step (plasticity criteria for shells) already shows vonMises stresses that are nearer to the set plasticity limit of $3.6 \cdot 10^8 \frac{N}{m^2}$. Though, as expected after only one loop, the vonMises stresses are not uniformly distributed because the sizer does not consider global stress redistribution during a sizing step. It has also to be considered that the zones are rather large and therefore the allocation of the mean of all corresponding FE-stresses leaves several stress peaks unconsidered during optimisation (see fig. 61 pressure dome). The results are peak values that lie above the yield stress and a less uniform vonMises stress distribution as all elements of the same zone have the same thickness. Optimising single FE-elements would yield better results though the thickness distributions would hardly be manufacturable.

Figure 61: Thickness and vonMises distribution after shell optimisation for plasticity criteria (*ultimate load*)

If the calculation is carried out taking into account only the five highest stresses of each zone and assigning the mean of these values to the field, the resulting stresses are much lower as can be seen in fig. 62. Obviously, the thicknesses increase in this case.

Figure 62: Thickness and vonMises distribution after optimisation with 5 highest FE-stresses for each zone

Experimental calculations with the fuselage model showed the importance of choosing realistic dimensions and geometries for the FE-model to be sized. A fuselage model without any longitudinal stringers resulted in unrealistically high shell thicknesses. Applying longitudinal stringers and varying their cross sectional area lead to thinner fuselage shells. The results shown above
were obtained by optimising only the shells (no stringers and frames were adapted).
Sizing the structure with buckling criteria yields slightly higher shell thicknesses, though as the stringers take up the main normal stresses, the shells still remain rather thin.

12.7 Application: Optimising Stringers and Shells

Compared to the sizing of only shells, a further weight reduction can be achieved. It has to be mentioned though that the algorithm still optimises locally and therefore does not design the stringers to carry most of the normal stresses in order to relieve the shells. This means that the entire structure is still not optimised according to minimum weight but simply to support the local FE-stresses read from the *.f06-file. Still, examining the results obtained after 5 FE-sizing loops (see 12.10), optimising the stringers tends to lead to lighter structures.

<table>
<thead>
<tr>
<th></th>
<th>Mass</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only shells optim.</td>
<td>7440kg</td>
</tr>
<tr>
<td>Shells &amp; stringers optim.</td>
<td>4963kg</td>
</tr>
</tbody>
</table>

As can clearly be seen in fig. 64, the stress distribution after shell and stringer optimisation is more uniform than after optimising only shells. The additional sizing of stringers obviously enables a better adaptation to the applied load case.
12.8 Application: Composite Structure Sizing

During the validation an unequal stress distribution between the two plies could be observed. For one FE-sizing loop, the sizer adapts the thicknesses correctly to produce a total ply reserve factor higher than 1. The question arising is now, how further loops would change the two thicknesses. A total of 10 FE-sizing loops were performed on the mentioned composite plate, which produced an interesting and rather unexpected result. The ply having the higher thickness after the first loop is further thickened whereas the second thickness is further decreased. The ESA-COMP plots still show an acceptable reserve. It is even reduced towards the optimum reserve factor of 1 (fig. 66).

![Figure 65: Reserve after 1 FE-sizing loop (failure if > 1)](image1)

![Figure 66: Reserve after 10 FE-sizing loops (failure if > 1)](image2)

Though, it is obviously not the aim to finally substitute one ply by the other, as it seems currently to be done by the sizer. There are several possible factors that lead to this unusual result. Firstly, according to the load case applied, the resin of one ply is less strained by shear stresses than the other (see fig. 67). This results in different reserve factors as in Tsai-Hill the shear stress is considered too.

![Figure 67: Accumulation and compensation of normal stresses in ply fibres (∓ 45°)](image3)

A second reason for the sizer to continuously thicken one ply is that it only considers normal stresses and neither shear deformations nor bending deformations due to the fact that the laminate is not symmetric and that it becomes more and more unbalanced. Meaning that it does not notice that the thickness distribution is indeed correct considering the stresses but that at the same time the laminate will be heavily deformed.

The conclusion to these tests is that the Tsai-Hill criteria leads indeed to optimised results
considering the stresses alone, though that it should be combined with additional restraints like e.g. maximum deformations.

12.9 Calculation Time

The main aim of developing and implementing a sizing algorithm working autonomously and with as few FE-calculations as possible was to reduce the currently needed large amount of time for calculations during aircraft pre-design. To receive therefore a rough estimation of the time saved or the time needed for an optimisation run, several time checks were run. The algorithm used during the feasibility study [2], showed already acceptable results. The time needed for the sizing itself was and still is negligible (analytically calculated thicknesses take fractions of a second, even for as large structures as 7800 elements).

The major time consumption during the feasibility study [2] appeared to be the reading of the bdf- and f06-files and especially the automatic zone detection.

The results achieved are impressive: the zone detection, taking up to 35 minutes for 7800 FE-elements could be reduced by a factor of 1000, down to 2 seconds. The whole sizing process previously needing 37 minutes is now run in about 2.5 seconds. Currently therefore, the FE-analyses are the decisive time factor.

![Figure 68: Calculation time for pre-processing without and with zone detection (feasibility study [2])](image)

As a result, one of the major tasks during the actual work was to further reduce pre-processing time, especially for zone detection. In order to achieve the required speed, the Boost Graph concept was introduced as framework of the sizing algorithm (see chap. 4). Now, all data is stored in a single graph structure which provides extremely fast access to any required information in contrast to the rather slow and bulky vectors and maps that have been used before.
Having achieved such a significant acceleration, the algorithm can now be included into other optimisation routines to fastly run some optimisation loops e.g. for preconditioning new topologies during an over all topology optimisation with evolutionary algorithms.

12.10 Convergence in FE-Sizing Loop

An important feature included in the algorithm is the possibility to automatically run recalculations of the FE-model using the optimised thicknesses. These recalculations are needed because the elements are sized locally and therefore no global stress redistributions are taken into account while running one sizing step.

The allowed yield stress was set to $3.6 \cdot 10^8 \frac{N}{m^2}$, the model examined is the wing box introduced in 12.5.
The convergence to the yield stress for most of the curves can clearly be seen. The reason for some curves remaining at a lower stress level can be found in the thickness plot (fig. 71). Zone 1611 reaches the minimum thickness and is therefore not made thinner anymore. Zone 344 has almost reached the minimum thickness and will probably not reach the maximum yield stress.

![Figure 71: Thickness characteristics for FE-sizing loop (plasticity criteria, 10 loops)](image)

Convergence of the shell thickness itself could not be clearly proven for all zones, though the curves are generally flattening for increasing loop numbers. In the actual case, no break outs are occurring (i.e. the curves are well damped). Should this not be the case for a different model, the user has the possibility to insert a damping factor in order to reduce the thickness change after each sizing step. The factor can be set in the settings file and ranges from 0 to 1, with 0 triggering no damping.

The formula considering the damping factor is: \( t_{\text{new}} = t_{\text{initial}} - (t_{\text{initial}} - t_{\text{new}}) \cdot (1 - \text{DampingFactor}) \)

All values in fig. 70 and fig. 71 have been calculated with DampingFactor=1.

The plots were generated using the output data saved in *thickPlot.xls and *stressPlot.xls.

Examining the mass plot over 10 FE-sizing loops proves that even though single thickness curves may not continuously decrease, the entire structural weight converges rapidly.

![Figure 72: Mass development over 10 FE-sizing loops (plasticity, Wing_Stringers.CATAnalysis)](image)
13 Conclusion

Considering the data received from the validation during the feasibility study [2] and the additional results acquired with more complex structures (wing and fuselage) the operability of the implemented algorithm could clearly be proven and demonstrated.

The results of the actual work can be summarized as follows: A complete sizing process has been implemented that operates autonomously from the parsing of the FE-data, along the detection of zones and up to the calculation and output of the optimised thickness into a bdf-file that can be directly used for further NASTRAN calculations.

The algorithm is extremely efficient, needing fractions of the time used when carrying out a complete FE-analysis. In addition, as it is based on simple and established analytical sizing criteria, the sizer can be used for any FE-model consisting of 2D-shells and 1D-stringers, not only aircraft structures.

The vonMises stress convergence showed to be rather fast, needing generally no more than four or five FE-sizing loops to achieve the yield stress set in the settings file.

Providing a semi-automatic or automatic zone detection function, thickness distributions can be achieved which are actually manufacturable. Furthermore, the algorithm can be included in topology optimisation tools (such as evolutionary algorithm topology optimisation) where the topology undergoes constant changes and therefore zones disappear or are created. The function easily handles these modifications by automatically detecting the newly created zones, using no more than fractions of seconds ( 12.9 ).

Aware that to guarantee a high flexibility, the user has to be able to adapt certain parameters to the actual problem, a settings file *.ini was created. Before running the sizing process, the user can individually set parameters, enable or disable sizing criteria, chose zone detection methods etc.

With support of the Boost Graph framework, an extremely fast and reliable data processing method was found to accelerate the sizing process up to 1000 times compared to the approach working with lists and vectors.

Furthermore, with additional sizing criteria such as stringer optimisation and adaptation of simple composite structures, the sizer is able to handle a vast range of different FE-structures and FE-elements with still a relatively small amount of basic mechanical rules.

As the algorithm calculates all optimised thicknesses considering the provided local FE-element stresses, it is not possible to find a global minimum of thickness distributions (i.e. a thickness distribution to achieve the lowest structural weight). This means that if e.g. the stringers had had a different thickness at the beginning of the process, the resulting shell thicknesses would have possibly resulted in lower values. The sizer only adapts each element to match the provided local stresses and therefore does not consider subsequent stress redistribution during the process.

The only way to consider redistribution is to run the FE-sizing loop, though there are still only local minima detected.

The sizer itself is therefore rather designed to precondition an already existing structure with rough thickness distributions for further FE-Analyses than to globally optimize and output a structure ready for fabrication.
14 Outlook

According to the results, the algorithm works properly and produces optimised thicknesses needing only seconds even for larger FE-models. Though, ready for first professional calculations, there are still some possible improvements to complete the algorithm. The following paragraphs will explain some of the extensions and modifications considered to be useful for further application of the sizing algorithm.

14.1 Considering Bending-Moments

Any bending-moments occurring in FE-elements (and hence possibly in buckling fields too), are not taken into account for buckling criteria. However, according to Farshad [11] these effects should not be neglected as they considerably lower the critical buckling stress. It is strongly recommended that in future versions, these effects are considered by implementing according sizing rules.

14.2 Stress Allocation for Zones

Providing the possibility to calculate the mean value of a defined number of highest FE-stress values for each zone, the algorithm already offers a more flexible method for stress allocation than during the feasibility study [2]. The user simply defines the number of highest values to be taken for the calculation of the average stress in the settings file *.ini.

There is still a different possibility already described in the feasibility study [2]. The described Gauss-method would allow to take all elements into account, though weighting them differently. Higher values would therefore be weighted higher, lower values would have a lower impact. The Gauss-method is still not implemented in the actual algorithm because the improved stress allocation already yields usable results. Nevertheless, the Gauss-method may be included in a later version for a more complete consideration of the FE-stresses.

![Gaussian curve to average over a range of differently weighted stresses](image)

Figure 73: Gaussian curve to average over a range of differently weighted stresses

14.3 Non-Isotropic Materials

Currently, for non-isotropic materials, i.e. composite structures, a simple Tsai-Hill-criteria is implemented. The structures that can be calculated are rather basic structures, which will be enough for shell optimisation. Composite stringers and beam elements are generally more complex to design and hence sizing criteria applying to these structures have still to be created. A basic introduction to methods for composite skin-stringer structure optimisation is given by
In Addition, as shown in 12.8, the Tsai-Hill criteria alone will not provide usable results. The laminates will result in highly asymmetric and unbalanced structures because the criteria considers only the stresses and is not concerned by any deformation. Therefore it might be useful to add a restrictive criteria for deformations to receive more realistic ply thickness distributions.

14.4 Corner Detection for Zones

As mentioned in 7.3 the problem of detecting rectangular or triangular zones with smooth edges still remains. If the angle between three nodes is lower than the CornerAngle set in the settings file, there will never be detected a corner even if the edge describes a $90^\circ$ direction change. The solution to this problem is simply inserting a 'compass' variable into the corner detection algorithm in zone_detection.cpp which cumulates the angles during the direction changes and marks the current node as zone corner when a certain angle is reached (e.g. $45^\circ$).

14.5 Cutouts

In a later version, it may also be necessary to size panels with cutouts. This would require new sizing criteria as well as the sizer’s ability to detect such zones. Currently the algorithm will be confused by zones bearing holes because it will detect an inner and an outer border (the zone will then be marked as 'weird' and will not be optimised).

14.6 Stiffened Plates

Generally, the zones defined on wing surfaces and fuselage structures may stretch over several stringer rows and will therefore not only have stringers attached to their edges but also in between them. Sizing criteria for such stiffened plates can be found in HSS [3] and Bruhn [5] for some specific stringer cross section types. In addition to the implementation of a new criteria it will be necessary to implement an additional function to enable the zone detection algorithm to detect stringers lying on the plate, i.e. not on the edges.

14.7 Considering Exact Geometry

The presently implemented criteria are strictly spoken only applicable to exactly rectangular shapes and exactly equilateral triangles. The algorithm generally recognises all shapes having three or four corner nodes and assumes that these elements are either rectangles or triangles. Obviously, there are several four- and three-corner forms that are not equal to the mentioned geometries (trapezoids, isosceles triangles etc.). Applying the sizing criteria to such elements results in minor or even major errors, depending on the deviation of the element from the ideal form. Especially buckling criteria are extremely sensible to the geometry. Therefore, to add criteria taking such forms into account would be an improvement in the accuracy of the sizer’s results.

14.8 Shell Curvature

Häfner [1] has already partly investigated the impact of shell curvature on buckling criteria for fuselage shells. The conclusion that the curvature in one direction does lower the critical buckling stress but that the curvature in the second direction will compensate this effect was drawn. The question now arising concerns structures with only one bending direction. How deep is the impact of a single curvature? For further accuracy of the buckling criteria, this problem should be investigated and eventually be integrated in the criteria as curvature is a major weakening factor for shells.
14.9 Integration in EA

The main aim to improve the automatic zone detection was to create a robust algorithm that is able to cope with a constantly changing topology. Such changes are found in topology optimisation with evolutionary algorithms where entire ribs and zones are newly created or deleted. Therefore the next step would be to include the sizing algorithm into such an optimisation tool and, after testing its reliability, to use the sizer to further accelerate the topology optimisation tool.
A File List

C++-Program Files:

Main-files:

1) basic_types.hpp: Definition of some general types
2) beambuckling_criteria.hpp: Buckling criteria for beams, for isotropic materials
3) beampredefStrain_criteria.hpp: Criteria for predefined maximum strain for beams, for isotropic materials
4) beamvonMises_criteria.hpp: Plasticity criteria for beams, for isotropic materials
5) edge_properties.hpp: Properties applicable to graph edges
6) exception.h: Messages thrown in cases of algorithm failures
7) graph_filter.hpp: Predefined filter functions for graph properties
8) graph_types.hpp: Definition of basic graph variables and iterators
9) log.cpp: Functions for different log outputs during the sizing process
10) log.h: Log header file
11) MainSizing.cpp: Main sizing algorithm calling all criteria and subfunctions
12) MainSizing.h: Header file for MainSizing.cpp
13) program_options.cpp: Function to read information defined in the settings file
14) program_options.h: Header file for program_options
15) shellbuckling_criteria.hpp: Buckling criteria for shells, for isotropic materials
16) shellcomp_tsaiHill_criteria.hpp: Tsai-Hill criteria for shells, for composite materials
17) shellpredefStrain_criteria.hpp: Criteria for predefined maximum strain for shells, for isotropic materials
18) shellvonMises_criteria.hpp: Plasticity criteria for shells, for isotropic materials
19) skinstringerbuck_criteria.hpp: Skin-stringer failure criteria, for isotropic materials
20) StdAfx.cpp: Auxiliary file
21) StdAfx.h: Header for StdAfx.cpp
22) vec_geo.cpp: Auxiliary functions for vector calculations
23) vec_geo.h: Header for vec_geo.cpp
24) vertex_properties.hpp: Properties and functions applicable to graph vertices
25) zone_detection.cpp: Functions for different zone detection methods
26) zone_detection.h: Header containing BFS-visitor
IO-files:
27) bdf_basic_grammar.hpp: Basic grammar library for bdf-file parsing
28) bdf_element_grammar.hpp: Grammar library to read element text blocks
29) bdf_io.cpp: Main parsing functions for bdf-file
30) bdf_io.h: Header for bdf-parsing
31) bdf_material_grammar.hpp: Grammar library to read material text blocks
32) bdf_node_grammar.hpp: Grammar library to read node text blocks
33) bdf_parser.hpp: Parser library needed before rewriting bdf-file
34) bdf_property_grammar.hpp: Grammar library to read property text blocks
35) bdf_writer.hpp: Library to rewrite optimised data into bdf-file
36) csm_element.cpp: Functions for elements-container
37) csm_element.h: Types and functions for elements-container
38) csm_element_types.cpp: Functions for different element types (CQUAD4, CQUAD8 etc.)
39) csm_element_types.h: Definition of different element types (CQUAD4, CQUAD8 etc.)
40) csm_material.cpp: Functions for materials-container
41) csm_material.h: Types and functions for materials-container
42) csm_node.cpp: Functions for nodes-container
43) csm_node.h: Types and functions for nodes-container
44) csm_property.cpp: Functions for properties-container
45) csm_property.h: Types and functions for properties-container
46) csm_simple.cpp: Functions for simple-container (used to parse and rewrite *thickPlot.xls and *stressPlot.xls-files)
47) csm_simple.h: Types and functions for simple-container (for *thickPlot.xls- and *stressPlot.xls-files)
48) csm_spirit_actors.hpp: Spirit actors library used for parsing
49) csm_stress.cpp: Functions for stress-container (for f06-data)
50) csm_stress.h: Types and functions for stress-container
51) csm_zone.cpp: Functions for zones-container (for *.zon-file)
52) csm_zone.h: Types and functions for zones-container
53) f06_io.cpp: Main parsing functions for f06-file
54) f06_io.h: Header for f06-parsing
<table>
<thead>
<tr>
<th>No.</th>
<th>File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>55</td>
<td>f06_stress_grammar.hpp</td>
<td>Grammar library to read stress text blocks</td>
</tr>
<tr>
<td>56</td>
<td>io.cpp</td>
<td>General function for data input</td>
</tr>
<tr>
<td>57</td>
<td>io.h</td>
<td>Header containing variable definitions for bdf- and f06-parsing</td>
</tr>
<tr>
<td>58</td>
<td>load_file.cpp</td>
<td>Function to load files</td>
</tr>
<tr>
<td>59</td>
<td>load_file.h</td>
<td>Header for file load function</td>
</tr>
<tr>
<td>60</td>
<td>ReadElmData.cpp</td>
<td>Function to call bdf- and f06-parsing functions</td>
</tr>
<tr>
<td>61</td>
<td>ReadElmData.h</td>
<td>Header for parsing call functions</td>
</tr>
<tr>
<td>62</td>
<td>text_blocks.cpp</td>
<td>Functions for text block handling (during file parsing)</td>
</tr>
<tr>
<td>63</td>
<td>text_blocks.h</td>
<td>Header for text block handling functions</td>
</tr>
<tr>
<td>64</td>
<td>txt_writer.hpp</td>
<td>Functions to write data into files other than the bdf-file</td>
</tr>
<tr>
<td>65</td>
<td>zone_grammar.hpp</td>
<td>Grammar library to read zones from *.zon-file</td>
</tr>
</tbody>
</table>

**Output Files:**

<table>
<thead>
<tr>
<th>No.</th>
<th>File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>66</td>
<td>*.log_siz</td>
<td>Log file containing comments and information about each step of the sizing process</td>
</tr>
<tr>
<td>67</td>
<td>*.bdf</td>
<td>bulk data file providing property and topology information</td>
</tr>
<tr>
<td>68</td>
<td>*.tec.dat</td>
<td>Optimised thicknesses output in Tecplot-format</td>
</tr>
<tr>
<td>69</td>
<td>*.elementData.txt</td>
<td>Data output for each element/zone; allocated properties, stresses, nodes etc.</td>
</tr>
<tr>
<td>70</td>
<td>*.f06</td>
<td>NASTRAN result file providing stress values for each FE-element</td>
</tr>
<tr>
<td>71</td>
<td>*.catia.txt</td>
<td>Optimised thicknesses output in CATIA V5-format for single thickness fringe assignment</td>
</tr>
<tr>
<td>72</td>
<td>*.thickPlot.xls</td>
<td>Optimised thicknesses output (zone number &amp; thickness)</td>
</tr>
<tr>
<td>73</td>
<td>*.stressPlot.xls</td>
<td>Stress output (zone number &amp; vonMises stress)</td>
</tr>
<tr>
<td>74</td>
<td>*.massPlot.xls</td>
<td>Mass output (zone number &amp; mass)</td>
</tr>
<tr>
<td>75</td>
<td>*.reserve.xls</td>
<td>Reserve output (zone number &amp; reserve factor for each activated criteria)</td>
</tr>
<tr>
<td>76</td>
<td>*.ini</td>
<td>Settings file containing all parameters and paths used for the sizing process</td>
</tr>
<tr>
<td>77</td>
<td>*.zon</td>
<td>Detected zones output in text file (zone number, FE-elements, corner nodes, adjacent stringers)</td>
</tr>
</tbody>
</table>
B Windows: Compiling and Running the Sizer

B.1 Compiling with Dev-C++

The Windows version of the sizing algorithm was compiled with Dev-C++ (version 4.9.9.2). The development environment uses the GNU compiler collection (GCC) and facilitates therefore the conversion from C++-code written in LINUX to Windows applicable source code. There are though some characteristics which could make the conversion rather intricate. Here follows a general description to convert and compile code imported from LINUX.

1) Download and install Dev-C++ (the software can be found in [18]).
2) Download and install the Boost Graph Library from [19].
3) Create a new project in Dev-C++ and add all *.cpp files to the project (*.h- and *.hpp-files do not have to be necessarily included).
4) Under Project - Project Options - Directories - Include Directories include the general boost path and all paths where the source and header files are located.
5) To optimise the execution time of the *.exe-file created later on, activate ‘Best Optimisation’ under Project - Project Options - Compiler - Optimisation.
6) To create the *.exe-file, compile the code.

When compiling the code error messages may be displayed. The Dev-C++ environment seems to be more sensible to minor flaws that were overlooked by LINUX developers (e.g. string a='f' instead of string a="f"). Such errors will be displayed and commented by Dev-C++.

A rather intricate error could occur when linking the header files after a successful compilation. The message then being:
[Linker error] undefined reference to ‘boost::filesystem::path....’ This is the case if there is e.g. a Boost Graph header defining functions in a source file. The problem encountered by the linker is that the header is actually found but the according source file is not. The easiest solution is to simply add the corresponding source file to the actual project. It may also be possible to just indicate explicitly the path of the missing file.

In addition, it has to be considered that the Windows and LINUX systems mark line ends differently. This may cause problems especially for parsing grammar. In Windows, the parser may continue reading symbols at the file end or it may not be able to detect a line end and the following data will not be completely read.

The problem was solved in the actual grammar descriptions by either writing a specific string to mark the end of a line or by checking the containers for unsuitable data after parsing.

B.2 Running Win_Sizer.exe

The settings file *.ini has to be located in the same folder like Win_Sizer.exe.

To facilitate the handling of the various settings possibilities, a graphical user interface (GUI) was created.
Figure 74: Graphical user interface (sheet 1)

Figure 75: Graphical user interface (sheet 2)
Figure 76: Graphical user interface (sheet 3)

Figure 77: Graphical user interface (sheet 4)
An online help tool is enclosed to the program providing more detailed information about each setting point. The graphical user interface simply writes all settings into an *.ini file named after the *.bdf-file used for the optimisation. The user has therefore the possibility to manually change settings directly in the file itself.

The sizer receives the name of the *.ini file either from the graphical interface or from the console input (e.g. as console command in the corresponding directory, where WinSizer.exe is located: WinSizer.exe wing_ifasd3.ini).

In LINUX no graphical user interface has been implemented so far, therefore the sizer has to be started by console command.

To locate possible errors occurring during sizing, the user may consult the log file (*.log_siz) created during the entire process. For each step and loop, different information and results are given to enable the user to determine where the error occurred.
C Settings File: *.ini

General Options:
- log-file: Define log-file path
- txt-files: Define path for *.elementData.txt, *.tec.dat, *.catia.txt
- plot-files: Define path for *.thickPlot.xls and *.stressPlot.xls
- zone-file: Define path for *.zon

Model Options:
- bulk-file: Path for *.bdf
- f06-file: Path for *.f06
- authorize: Authorisation sequence for NASTRAN call
- bdf-scale-mode: Scale of input *.bdf-file

Sizing Options
- WriteTECFile: Activating/Deactivating file output for *.tec.dat
- WriteTXTFile: Activating/Deactivating file output for *.catia.txt
- WriteZoneFile: Activating/Deactivating file output for *.zon
- WriteBDFFile: Activating/Deactivating file output for *.bdf
- WriteThicknessPlotFile: Activating/Deactivating file output for *.thickPlot.xls
- WriteStressPlotFile: Activating/Deactivating file output for *.stressPlot.xls
- WriteDataPlotFile: Activating/Deactivating file output for *.elementData.txt
- WriteReservePlotFile: Activating/Deactivating file output for *.reserve.xls
- WriteMassPlotFile: Activating/Deactivating file output for *.massPlot.xls
- TotalStringFailure: Defining calculation method for stringer failure (presently only the Gerard-method is implemented)
- Limitfactor: Conversion factor from ultimate to limit load: \( \sigma_{\text{LimitLoad}} = \frac{\sigma_{\text{UltimateLoad}}}{\text{Factor}} \)
- MinimalShellThickness: Lowest allowed shell thickness
- MinimalPlyThickness: Lowest allowed ply thickness
- Skinstep: Skin step during iterations for iteratively calculated thickness (in [m])
- Plystep: Ply step during iterations for iteratively calculated ply thickness (in [m])
- mass_precision: floating point precision for mass output (0 = only kg, 1 = down to 100g etc.)
- SigmaYieldMax: Allowed maximum yield stress
- **SigmaultMax**: Allowed maximum failure stress
- **MaxStrain**: Allowed maximum strain
- **LimitIterations**: Maximum number of iterations for iteratively calculated thickness
- **Upperlimit**: Upper limit for reserve factor (for iterative calculations)
- **Lowerlimit**: Lower limit for reserve factor (for iterative calculations)
- **stressMode**: Use 'mean' or 'max' stress for sizing criteria
- **stressModeZone**: Assign 'max' or 'mean' stress of FE-elements to the corresponding zone
- **maxStressElms**: Number of FE-elements to be taken into account to calculate maximum stress for zone (i.e. 3 = the mean of the three highest occurring FE-element stresses in this zone is calculated and allocated to the zone)

**Sizing Criteria:**

- **PlasticityLimitLoad**: Activating/Deactivating criteria for plasticity with *limit load*, for shells
- **PlasticityUltimateLoad**: Activating/Deactivating criteria for plasticity with *ultimate load*, for shells
- **BucklingLimitLoad**: Activating/Deactivating criteria for buckling with *limit load*, for shells
- **BucklingUltimateLoad**: Activating/Deactivating criteria for buckling with *ultimate load*, for shells
- **BeamPlastLimitLoad**: Activating/Deactivating criteria for plasticity with *limit load*, for stringers
- **BeamPlastUltimateLoad**: Activating/Deactivating criteria for plasticity with *ultimate load*, for stringers
- **BeamBuckLimitLoad**: Activating/Deactivating criteria for buckling with *limit load*, for stringers
- **BeamBuckUltimateLoad**: Activating/Deactivating criteria for buckling with *ultimate load*, for stringers
- **PredefStrainLimitLoad**: Activating/Deactivating criteria for predefined strain with *limit load*, for shells
- **PredefStrainUltimateLoad**: Activating/Deactivating criteria for predefined strain with *ultimate load*, for shells
- **BeamStrainLimitLoad**: Activating/Deactivating criteria for predefined strain with *limit load*, for stringers
- **BeamStrainUltimateLoad**: Activating/Deactivating criteria for predefined strain with *ultimate load*, for stringers
- **UltimateFailureUltimateLoad**: Activating/Deactivating criteria for ultimate failure with *ultimate load*, for shells
- **BeamUltimFailUltimateLoad**: Activating/Deactivating criteria for ultimate failure with ultimate load, for stringers
- **SkinStringerFailure**: Activating/Deactivating criteria for skin-stringer failure with ultimate load
- **CompShellTsaiHillLimit**: Activating/Deactivating criteria for Tsai-Hill with limit load, for composite shells
- **CompShellTsaiHillUltimate**: Activating/Deactivating criteria for Tsai-Hill with ultimate load, for composite shells
- **CriteriaMapping**: Activating/Deactivating the output of the criteria numbers instead of the optimised thickness

**Zones:**
- **excl_zones**: List of zones to be excluded from sizing. The numbers correspond to the numbering in the *.zon-file. NOTE: the format has to be: 1,2,3... (no spaces are allowed)
- **FileZones**: Reading predefined zones from zone file *.zon
- **PredefThickZones**: Determining zones by predefined thickness distributions
- **AutomZoneDetect**: Building zones automatically, considering different thicknesses, stringers and inclination angle as border criteria
- **StringerBorders**: Activating/Deactivating stringer border criteria
- **AngleBorders**: Activating/Deactivating angle border criteria
- **BorderAngle**: Setting the inclination angle above which neighbour elements are considered to belong to a different zone
- **CornerAngle**: Setting the corner angle above which a node is considered to be a zone’s corner node

**FE-Sizing Loop:**
- **number_of_felops**: Setting the number of FE-sizing loops to be run
- **keepZonesAfterFirstLoop**: If enabled, the algorithm detects the zones automatically only the first time and reads the zones from the *.zon-file during the following loops (to ensure that always the same zones are optimised)
- **nastCall**: Call NASTRAN to recalculate new stresses
- **nastranPath**: Define the path for the NASTRAN call
- **deletef06**: Activating/Deactivating deletion of old *.f06-file before calling NASTRAN
- **deletebdf**: Activating/Deactivating deletion of old *.bdf-file
- **DampingFactor**: Setting the damping factor to reduce breakouts during FE-sizing loops

\[
t_{\text{new}} = t_{\text{initial}} - (t_{\text{initial}} - t_{\text{new}}) \cdot (1 - \text{dampfac})
\]

\[
t_{\text{initial}} \quad \text{initial thickness read from *.bdf-file}
\]

\[
t_{\text{new}} \quad \text{optimised thickness}
\]
Additional Details to Sizing Criteria Formulas

In this paragraph detailed explanations of formulas used and assumptions made in the currently implemented sizing criteria can be found. It is a summary of the documented equations in Häfner [1], chap. 6 for sizing rules, including some extensions, corrections and comments.

D.1 Buckling Criteria

For calculation of critical buckling stresses \( \sigma_{\text{crit}} = k_c \cdot \frac{\pi^2 E}{12(1-\nu^2)} \left( \frac{1}{b} \right)^2 \) different values for \( k_c \) are needed for \( \sigma_{\text{crit},x} \), \( \sigma_{\text{crit},y} \) and \( \tau_{\text{crit},xy} \).

The derivation of these values is drawn from the basic equation for plates (according to Ermanni [9] and Farshad [11]):

\[
\frac{E t^3}{12(1-\nu^2)} \left[ \frac{\partial^4 w_b}{\partial x^4} + 2 \frac{\partial^4 w_b}{\partial x^2 \partial y^2} + \frac{\partial^4 w_b}{\partial y^4} \right] - N_x \frac{\partial^2 w_b}{\partial x^2} - 2 N_{xy} \frac{\partial^2 w_b}{\partial x \partial y} - N_y \frac{\partial^2 w_b}{\partial y^2} = 0
\]

For unidirectional compression \( N_x = -\frac{P_x}{b}, \) \( N_y = N_{xy} = 0 \) or \( N_y = -\frac{P_y}{a}, \) \( N_x = N_{xy} = 0 \) respectively, the equation is being reduced to:

\[
D \nabla^4 w_b + \frac{P_x}{b} \frac{\partial^2 w_b}{\partial x^2} = 0 \quad \text{and} \quad D \nabla^4 w_b + \frac{P_y}{a} \frac{\partial^2 w_b}{\partial y^2} = 0
\]

with \( D = \frac{E t^3}{12(1-\nu^2)} \).

The corresponding constraints and the approach of \( w_b = c_{mn} \sin \frac{m \pi x}{a} \sin \frac{n \pi y}{b} \) lead to:

\[
N_{\text{crit},x} = \frac{P_{\text{crit}}}{b} = k_{c,x} \frac{\pi^2 D}{b^2} \quad \text{and} \quad N_{\text{crit},x} = \frac{P_{\text{crit}}}{a} = k_{c,y} \frac{\pi^2 D}{a^2}
\]

with \( k_{c,x} = \left( \frac{ma}{b} + \frac{b}{mb} \right) \) and \( k_{c,y} = \left( \frac{ma}{b} + \frac{b}{mb} \right) \), if \( n \) is set to \( n = 1 \).

The same formulas can be found in HSS [3] (for \( k_{c,y} \) the reciprocal ratio of \( \frac{a}{b} \), which was used for \( k_{c,x} \), is taken). For \( k_{c,xy} \) too, HSS [3] provides an approximate formula:

\[
\begin{align*}
a \frac{a}{b} & \leq 1 \quad & k_{c,xy} = 4.00 + \frac{5.34}{\beta^2} & \quad \beta = \frac{a}{b} \\
a \frac{a}{b} & > 1 \quad & k_{c,xy} = 5.34 + \frac{4.00}{\beta^2} & \quad \beta = \frac{a}{b}
\end{align*}
\]

The calculation methods have been implemented in C++ and provide the curves shown in fig. 78 and fig. 79.
The $k_c$-values for triangular elements are calculated with similar formulas, also indicated in HSS [3]. Assuming that the triangles are more or less equilateral, the corresponding equations were included in the algorithm.
1.2 Compression and shear

Equilateral triangular plate
All edges simply supported.

\[ P = \alpha \sigma_{x} \sigma_{y}, \quad Q = \rho \sigma_{x} \sigma_{y}, \quad R = \tau \sigma_{x}, \quad S = P + Q, \quad T = P - Q \]

\[ \sigma_{c} = \frac{E \pi^2}{12(1-\nu^2)} \left( \frac{l}{a} \right)^2 \]

<table>
<thead>
<tr>
<th>No.</th>
<th>Loading</th>
<th>Results</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( P=0 ) ( Q=R=0 )</td>
<td>( S=T=P )</td>
<td>( P=7.316 )</td>
</tr>
<tr>
<td>2</td>
<td>( Q=0 ) ( P=R=0 )</td>
<td>( S=-T=Q )</td>
<td>( Q=7.419 )</td>
</tr>
<tr>
<td>3</td>
<td>( P=-Q ) ( R=0 )</td>
<td>( T=2P ) ( S=0 )</td>
<td>( P=13.43 ) ( R=-15.86 )</td>
</tr>
<tr>
<td>4</td>
<td>( R=0 ) ( P=Q=0 )</td>
<td>( S=T=0 )</td>
<td>( R=\pm 14.37 )</td>
</tr>
</tbody>
</table>

Figure 81: Calculation of \( k_{c} \)-values for buckling of triangular equilateral elements (source: HSS [3], B. Polygonal Plates)
D.2 Skin-Stringer Failure Under *Ultimate Load*

D.2.1 Scheme

The skin-stringer failure criteria performs several intermediate and new calculations in each iteration step. The following figure gives an overview over the calculation sequence.

![Scheme to the iterative calculation of optimised thicknesses](image)

Figure 82: Scheme to the iterative calculation of optimised thicknesses

D.2.2 Effective Width

The $k_{w}$-values for $w_{eff}$ are obtained from a corresponding diagram (Niu [12], p. 144). The curve was digitalized with *MATLAB*, using the *polynomial curve-fitting* function (fig. 83 and fig. 84). The coefficients for a polynomial of degree 5 were determined in *MATLAB* and implemented in C++.
D.2.3 Shear to Normal Stress Conversion

To determine the resulting load for the shell, Hafner [1], chap. 6 suggests the shear stress to be redistributed to the normal stresses. In Leichtbau I [9] for the redistribution of shear stresses for tension fields, the following calculation is suggested:

\[ \sigma_{x,\text{tau}} = R(\tau_{xy}\cot\alpha - \tau_{xy,\text{crit}}) \]
\[ \sigma_{y,\text{tau}} = \tau_{xy}\tan\alpha + (R - 1)\tau_{xy}\cot\alpha - R\tau_{xy,\text{crit}} \approx \sigma_{y,\text{tau}} \]

with \( R = \frac{\sigma_x}{\sigma_{x,\text{max}}} \approx 1 \):

\[ \sigma_{x,\text{tau}} = \tau_{xy}\cot\alpha - \tau_{xy,\text{crit}} \]
\[ \sigma_{y,\text{tau}} = \tau_{xy}\tan\alpha - \tau_{xy,\text{crit}} \]

For the actual problem this results in:

\[ \sigma_{\text{load},x} = \sigma_{\text{load},x} + \tau_{xy}\cot\gamma - \tau_{xy,\text{crit}} \]
\[ \sigma_{\text{load},y} = \sigma_{\text{load},y} + \tau_{xy}\tan\gamma - \tau_{xy,\text{crit}} \]

\( \gamma \) is set to approximately 45°.

As compression loads are treated, the following transformations were made:

\[ \sigma_{\text{load},x} = \sigma_{\text{load},x} + \tau_{xy}\cot\gamma + \tau_{xy,\text{crit}} \]
\[ \sigma_{\text{load},y} = \sigma_{\text{load},y} - \tau_{xy}\tan\gamma + \tau_{xy,\text{crit}} \]

D.2.4 Total Load on Effective Width After Shell Buckling

In D.2.3 the total load acting in the shell plane was determined. If the stresses are higher than the allowed critical buckling stress, the shell itself will carry only loads up to the corresponding critical buckling stress. The remaining load will be redistributed to the (non-buckling) effective width.

The new load on the effective width is calculated according to Hafner [1], chap. 6 for each side of the x- and y-direction (i.e. left and right zone edge):

\[ \sigma_{x,1} = \frac{(\sigma_{\text{shell},x})w + \sigma_{x,\text{crit}}(w - w_{\text{eff},1} - w_{\text{eff},2})}{w_{\text{eff},1}} \frac{w - w_{\text{eff},2}}{2w - w_{\text{eff},1} - w_{\text{eff},2}} \]
\[ \sigma_{x,2} = \frac{(\sigma_{\text{shell},x})w + \sigma_{x,\text{crit}}(w - w_{\text{eff},1} - w_{\text{eff},2})}{w_{\text{eff},2}} \frac{w - w_{\text{eff},2}}{2w - w_{\text{eff},1} - w_{\text{eff},2}} \]
And analogous for the y-direction. $\sigma_{x,\text{crit}}$ is added, as $\sigma_{\text{shell},x}$ etc. are all negative values (compression loads).

**Figure 85:** Plate scheme with stringers and effective width

![Sheet stress distribution before buckling](image1)

**Figure 86:** Sheet stress distribution after buckling (source: Bruhn [5], C7.10)

**Figure 86:** Load redistribution after shell buckling (source: Bruhn [5], C7.10)

It is essential that the critical buckling values $\sigma_{x,\text{crit}}$ etc. do not correspond to the values calcu-
lated after $\sigma_{x,crit} = k_c \cdot \frac{E}{(1 - \nu^2)^2}$ but that they correspond to the load stresses $\sigma_x, \sigma_y, \tau_{xy}$ found as soon as the plate buckles as it is a combined load case ($\sigma_{x,crit}$ etc. apply to unidirectional load cases). For combined load cases, the following equation has to be considered:

$$\frac{1}{R_{buckx,i}} + \frac{1}{R_{bucky,i}} + \frac{1}{R_{buckxy,i}} = \frac{-\sigma_{load,x,initial}}{t_{new}\sigma_{x,crit}} + \frac{-\sigma_{load,y,initial}}{t_{new}\sigma_{y,crit}} + \left(\frac{-\tau_{load,xy,initial}}{t_{new}\tau_{xy,crit}}\right)^2 \leq 1$$

Therefore the plate already buckles before $\sigma_x$ etc. meet the critical unidirectional buckling stress $\sigma_{x,crit}$. Hence, $\sigma_{x,crit}$ in $\sigma_{x,i}$ is recalculated after each thickness iteration step in the skin-stringer criteria.

To meet the required minimal reserve of $R_{buckling,i} = \left(\frac{1}{R_{buckx,i}} + \frac{1}{R_{bucky,i}} + \frac{1}{R_{buckxy,i}}\right)^{-1} \geq 1$ the following equation has to be met:

$$\frac{\sigma_{x,oldLoad}}{\sigma_{x,crit,t_{prev}}} = \frac{\sigma_{x,newLoad}}{\sigma_{x,crit,t_{new}}}$$

The new critical buckling stresses for combined loads are therefore calculated after:

$$\sigma_{x,newLoad} = \frac{\sigma_{x,oldLoad} \cdot \sigma_{x,crit,t_{new}}}{\sigma_{x,crit,t_{prev}}} = \sigma_{x,crit}$$

$$\sigma_{y,newLoad} = \frac{\sigma_{y,oldLoad} \cdot \sigma_{y,crit,t_{new}}}{\sigma_{y,crit,t_{prev}}} = \sigma_{y,crit}$$

$$\tau_{xy,newLoad} = \frac{\tau_{xy,oldLoad} \cdot \tau_{xy,crit,t_{new}}}{\tau_{xy,crit,t_{prev}}} = \tau_{xy,crit}$$

The ratios between the three components may certainly change but to keep the recalculation simple the ratios are assumed to be constantly the same.
D.2.5 Euler-Johnson-Curves

Figure 87: Euler-Johnson-Curves for stringer buckling in Transition Region (source: Bruhn [5], C7.22)

D.2.6 Gerard

To calculate \( \sigma_{cripp} \), Bruhn [5] suggests different semi empirical calculation methods for Gerard considering different cross section types:

- For angles, tubes, V groove plates, multi-corner sections and stiffened plates:
  \[
  \sigma_{cripp} = \sigma_s \cdot 0.56 \left( \frac{\alpha^2}{A} \right) \left( \frac{F}{\sigma_s} \right)^{0.85}
  \]

- For plates, T, cruciform and H sections:
  \[
  \sigma_{cripp} = \sigma_s \cdot 0.67 \left( \frac{\alpha^2}{A} \right) \left( \frac{F}{\sigma_s} \right)^{0.40}
  \]

- For 2-corner sections, Z, J and channel sections:
  \[
  \sigma_{cripp} = \sigma_s \cdot 3.2 \left( \frac{\alpha^2}{A} \right) \left( \frac{F}{\sigma_s} \right)^{0.75}
  \]

The values for parameter \( g \) were implemented for a number of different NASTRAN cross section types (‘T’, ‘T1’, ‘T2’, ‘CROSS’, ‘H’, ‘TUBE’, ‘L’, ‘BOX’, ‘BOX1’, ‘BAR’, ‘ROD’, ‘I’, ‘II’, ‘CHAN’, ‘CHAN2’, ‘HEXA’, ‘HAT’). NOTE: some of these cross sections have non-uniform thicknesses. Gerard considers only cross sections with uniform thickness. Therefore, the listed forms are identified though the thickness handed over to Gerard is a mean value of the cross section dimensions (see 11.9). The corresponding formulas for the Needham-method have not yet been implemented.
E NASTRAN-Conventions

During the feasibility study [2] a considerable part of the work was dedicated to the analysis of NASTRAN-output formats, coordinate system orientation and node numbering. The resulting conventions are summarized below to serve as reference for future work with such formats. The content of a typical bulk data file is shown in fig. 88. The sections used by the sizing algorithm are marked grey.

<table>
<thead>
<tr>
<th>Bulk-File *.bdf</th>
</tr>
</thead>
<tbody>
<tr>
<td>Displacements</td>
</tr>
<tr>
<td>Loads: PLOAD4...</td>
</tr>
<tr>
<td>Properties: PSHELL, PBEAML...</td>
</tr>
<tr>
<td>Elements: CQUAD, CTRIA, CBEAM...</td>
</tr>
<tr>
<td>Materials: MAT1, MAT2...</td>
</tr>
<tr>
<td>Nodes: GRID</td>
</tr>
</tbody>
</table>

Figure 88: Structure and sequence of bulk data file information

E.1 FE-Element Coordinate System and Node Numbering

The correct definition of the local FE-coordinate system is particularly important for the sizing algorithm. The complete stress allocation from FE-elements to buckling fields is based on the correct identification of the $\sigma_x$- and $\sigma_y$-directions of FE-elements and zones. In NASTRAN (Quick Reference Guide [16]) the local FE-coordinate system is defined as follows. The node numbering sequence is based on the sequence the nodes appear in the $bdf$-file for each FE-element.

The conventions apply to CQUAD8- and CTRIA6-elements as well as the first four/three nodes are located on the element’s corners like for simple CQUAD4- and CTRIA3-elements. The remaining nodes are distributed in between the corners. The algorithm only needs the first four/three nodes.
E.2 $f06$-File

E.2.1 Stress Allocation

In fig. 92 the stress output in the $f06$-NASTRAN-file for node and element stresses is shown. Even though there are only 2D-shells, NASTRAN distinguishes between nodes lying on the upper and lower surface of each FE-element, showing compression and tension stresses in case of bending moments (fig. E.2.1).

```
E.2.1 Stress Allocation

In fig. 92 the stress output in the $f06$-NASTRAN-file for node and element stresses is shown. Even though there are only 2D-shells, NASTRAN distinguishes between nodes lying on the upper and lower surface of each FE-element, showing compression and tension stresses in case of bending moments (fig. E.2.1).
```

Figure 91: 8 node stresses (2 values per node for upper and lower element surface)

Generally, the more complex the load case in a single FE-element, the more difficulties arise in calculating a reasonable average stress from the four nodes (i.e. from the eight provided values), as stresses on upper and lower surface can highly differ in direction and amount (fig. 92, gray mark).

Figure 92: Stress allocation in $f06$-NASTRAN-file for 2D shell elements
The row marked by \( CEN/4 \) contains the mean values of all eight values for the corresponding FE-element (row 1: \( \sigma_{x,elm,upper\,surface} = \frac{1}{4}\sum_1^4 \sigma_{x,nodeupper} \), row 2: \( \sigma_{x,elm,lower\,surface} = \frac{1}{4}\sum_1^4 \sigma_{x,nodelower} \)).

For \textit{C}TRIA-elements a similar format can be found.

The sizing algorithm only reads the average node values for the FE-element (red mark). All six values are stored and allocated to the \texttt{ELEMENT\_VERTEX} in the graph structure. During the sizing process either the maximum or mean value of upper and lower surface are taken, according to the method set in the settings file (see 7.5).

For beam elements, the four node stresses (\( \textit{SXC}, \textit{SXD}, \text{etc.}; \) marked gray) and the cross section maximum/minimum values at each end of the element are displayed (marked red). The sizing algorithm reads only the maximum and minimum cross section values of both ends.

![Figure 93: Stress allocation in \textit{f06-NASTRAN} file for beam elements](image)

Taking only the mentioned averaged stresses (especially for shell elements) leads to the mentioned problems of not considering single node peak values (see 11.1).

### E.3 NASTRAN Stringer Cross Sections

\textit{NASTRAN} provides several default stringer crosssection types for the \texttt{PBARL} and \texttt{PBEAML} properties. The sizer recognises the following types:
F  C++-Algorithm

This paragraph gives some explanations about implemented routines to facilitate the familiarization with the algorithm for any required extensions or changes. In addition, it may also provide some starting points for improvements. There are further comments written into the code on the respective lines in order to keep track of the algorithm’s steps.

F.1 Detecting and Saving Optimised Thicknesses

For each criteria the optimised thickness and the corresponding load case are stored in a map where the criteria number serves as key. The initial thickness is therefore not overwritten as it may be used in other sizing criteria. For stringers, the thickness is saved in the same structure and before writing the optimised data back into the bulk data file it is converted to the stringer dimensions (\texttt{dim1}, \texttt{dim2},...).

F.2 Reading FE-Information

To acquire the FE-information provided by the \texttt{bdf}- and \texttt{f06}-files a parser has been generated. The parser works with predefined grammar, using \textit{Boost Spirit Actors}. The grammar is defined for the different sections accordingly (nodes, elements, properties, materials). Currently, for elements the grammar is designed to read \texttt{CQUAD}-, \texttt{CTRIA}-, \texttt{CBAR}- and \texttt{CBEAM}-elements. Should any additional type be required, the corresponding grammar has to be added into the according \texttt{*_grammar.hpp}-file. Note: the bulk data file may display the values in 8-digit-format or 16-digit-format. The algorithm is able to recognise both formats and to parse the file accordingly.

The \texttt{f06}-file may contain several different load cases for the same FE-structure. The algorithm is able to allocate different stress values for each load case to each FE-element (and later to each zone). The sizing part then determines the load case yielding the highest thickness and writes the according thickness into the \texttt{bdf}-file.

To save the different stresses, a slightly more complex stress property structure had to be created...
to consider different load cases and different plies (the same property is used for isotropic and composite materials):

<table>
<thead>
<tr>
<th>nx</th>
<th>ply</th>
<th>loadcase</th>
<th>nx_1, nx_2,...</th>
</tr>
</thead>
<tbody>
<tr>
<td>ny</td>
<td>ply</td>
<td>loadcase</td>
<td>ny_1, ny_2,...</td>
</tr>
<tr>
<td>sxy</td>
<td>ply</td>
<td>loadcase</td>
<td>sxy_1, sxy_2,...</td>
</tr>
</tbody>
</table>

Figure 94: Nested map structure for the stress property of an ELEMENT_VERTEX or ZONE_VERTEX

A specific stress value of a zone vertex is then called by e.g. `graph[*vit].nx.ply[1].loadcase[1][0]`: first stress value in x-direction of load case 1 in ply 1 of the actual ZONE_VERTEX. For isotropic materials the ply number is always '1'.

F.3 Cross Section Area for Stringers

*NASTRAN* provides several different cross section types for CBAR- and CBEAM-elements (see *Quick Reference Guide* [16] and E.3). Therefore calculations of moments of inertia and cross section area have to be individually implemented for each type. The currently implemented forms are: 'ROD', 'TUBE', 'L', 'T', 'CHAN', 'T', 'BOX', 'BAR', 'CROSS', 'H', 'TH', '11', 'CHANI', 'Z', 'CHAN2', 'T2', 'BOX1', 'HEXA', 'HAT', 'BAR'. Being part of the vertex property class, the calculations are located in the `vertex_properties.hpp`-file.

F.4 `bdf_parser.hpp` and `bdf_writer.hpp`

The output into the `bdf`-file requires some additional parsing and ordering steps as *NASTRAN* and FE-software such as *CATIA V5* appear to be quite sensible to deviations from the original format. Therefore, before writing the data back to the file, the entire file is parsed again (taking a few microns, this step does not affect calculation time considerably) and saved in a single string variable. The altered data then is inserted in the following steps:

- Erasing all elements in the file string (starting point and ending point in the string are localised by examining the `elements-container` provided by the parser).
- Inserting all elements and their corresponding nodes. The elements have to be adapted because their property-id may have changed due to zone allocation.
- Erasing all property sets (again, positions are localised by examining the `properties-container` provided by the parser).
- Calling auxiliary function to reorder property sets. There may be some property sets that are not used anymore and will therefore not be written into the new `bdf`-file.
- Inserting property sets and their corresponding data.
- The `bdf`-file may include line breaks marked with '*` or '*A..'. As these line numberings continue in the node section of the file, they have to be adapted if the number of previously inserted properties is different from the initially existing number of property sets. Therefore, the algorithm runs a renumbering routine after inserting all properties.
- Finally, the writer has for some reason the tendency to add additional symbols at the end of the file. These symbols have to be removed.
Figure 95: ESA-COMP full report for ±45° laminate (1 FE-sizing loop)

Figure 96: ESA-COMP full report for ±45° laminate (10 FE-sizing loops)
Further Results

On the following pages additional calculation results can be found. Results for plasticity criteria under *limit load* for the wing model are presented on the following page.

For the fuselage model, criteria mapping and thickness distribution for plasticity, buckling and predefined strain criteria are shown. Only the shells were sized, the stringer dimensions were left as defined in the initial *bdf*-file. As the FE-elements correspond to buckling fields (surrounded by four stringers/frames on their edges), single FE-element optimisation was conducted in this case. In addition, an example for single FE-element optimisation for plasticity criteria under *ultimate load* is displayed with thickness and vonMises stress distribution as a further example.
<table>
<thead>
<tr>
<th>Fuselage: Criteria Mapping (Ultimate Load, 1 Loop)</th>
<th>Fuselage: Thickness Distrib. Shell Plasticity Crit., Ultimate Load, 1 Loop</th>
</tr>
</thead>
<tbody>
<tr>
<td>2: shell plasticity</td>
<td></td>
</tr>
<tr>
<td>4: shell buckling</td>
<td></td>
</tr>
<tr>
<td>10: shell predefined strain (0.36%)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fuselage: Thickness Distrib. Criteria 2, 4, 10 (Ultimate Load, 1 Loop)</th>
<th>Fuselage: vonMises Distrib. Shell Plasticity Crit., Ultimate Load, 1 Loop</th>
</tr>
</thead>
</table>

2: shell plasticity
table 4: shell buckling
table 10: shell predefined strain (0.36%)
I References


[19] Boost Graph Library: http://www.boost.org/more/getting_started.html