Università degli Studi di Napoli Federico II

Scuola Politecnica e delle Scienze di Base

DIPARTIMENTO DI INGEGNERIA INDUSTRIALE
Corso di Laurea in Ingegneria Aerospaziale

Tesi di Laurea Magistrale
in
Ingegneria Aerospaziale ed Astronautica

Development of a Java Framework for Parametric Aircraft Design

The Performance Analysis Module

Candidato:
Vittorio Trifari
Matricola M53/411

Relatori:
Prof. Fabrizio Nicolosi
Prof. Agostino De Marco

Anno Accademico 2014/2015

Abstract

This thesis has the main purpose of providing a comprehensive overview about the development, in Java, of a software dedicated to the preliminary design of an aircraft, focusing on the performance analysis module.

The point of view from which this subject will be observed expects first to define methodologies and theoretical aspects necessary for the examined performance calculation for then show, in more detail, the implementation of the latter within the software; this will be seen both from the point of view of the developer, through a detailed explanation of the architecture of the different calculation modules, both of a potential user developer, by showing some commented examples of use supplied with graphical and numerical results suitable, among other things, also to the validation of calculations performed.

Sommario

Questo lavoro di tesi ha lo scopo principale di fornire una panoramica esaustiva circa lo sviluppo, in ambiente Java, di un software dedicato al progetto preliminare di un aeromobile, concentrandosi sull'aspetto dell'analisi delle performance.

Il punto di vista con il quale verrà affrontato questo argomento prevede dapprima di inquadrare le metodologie e gli aspetti teorici necessari per la stima delle performance in esame per poi mostrare, più nel dettaglio, l'implementazione di questi ultimi all'interno del software; ciò sia dal punto di vista dello sviluppatore, attraverso la spiegazione dettagliata dell'architettura dei vari moduli di calcolo, sia dal punto di vista di un potenziale utente, tramite alcuni esempi d'uso commentati e corredati di risultati grafici e numerici atti, tra l'altro, anche alla validazione dei calcoli svolti.

Contents

Chapter 1

Software overview

Chapter 2

High lift devices effects

In the preliminary design of the wing a large number of requirements have to be fulfilled as exemplified in table 2.1; but, as many of them are in conflict, it is hardly ever possible to check them all, so that a compromise has to be found.

Wing design requirements
Low drag
High lift
Satisfactory maximum lift coefficient
Satisfactory stall quality
High value of the critical Mach number
Low weight
Ensure satisfactory performance in all flight phases

Table 2.1 Some wing design requirements

With respect of the last shown requirement, the wing is usually equipped with high-lift devices, which change it's shape, in order to make it performant both in cruise both in take-off and landing phases. In fact, as can be seen from table 2.2, these phases show conflicting objectives which can only be mediated through the introduction of these devices.

Cruise requirements
Small wing surface and high wing loading W/S
Small camber
Low drag
High speed
Lift generated using low c_L

Take-Off and Landing requirements
Big wing surface, or high c_{Lmax}, in order to have an high equivalent wing loading $m = W/(S \cdot c_{Lmax})$
High drag value in landing
Lift generated using high c_L due to the low speed

Table 2.2 Comparison between cruise and take-off/landing design requirements

2.1 Theoretical background

In this paragraph, a general overview of the different type of high-lift devices is provided as well as the semi-empirical steps used to predict their effects on aerodynamic perofmance of the wing.

The disigner may choose from a large collection of feasible high-lift systems, although in the case of a specific project this freedom will be limited, since incremental drag, mechanical complexity, development and maintenance costs and structural weight are all factors to be considered [30].

All high-lift devices can be divided in two main groups of which only the first one will be analyzed in this discussion:

- Systems for passive lift increase, such as *leading-edge devices* or *trailing-edge devices*, which modify the wing shape.
- Systems for active lift increase, such as *blown flaps* or *jet flaps*, which acts directly on the flow in order to control it.

Genrally speaking, *trailing-edge devices* are used to increase the wing maximum lift coefficient, while *leading-edge devices* are used to increase the stalling angle of attack.

A more in depth analysis shows that *trailing-edge devices* increase the camber and improve the flow at the trailing edge, but tend to promote leading edge stall on thin sections and may cause a reduction in the stalling angle of attack; on the other hand *leading-edge devices* postpone or eliminate leading edge stall, but they have little effects on the airfoil camber as a whole, although locally the camber is increased [30].

About *trailing-edge devices*, their effects can be resumed in:

- Higher c_L at a given angle of attack and higher c_{Lmax}
- Lower stalling angle of attack
- Lower zero-lift angle due to increasing camber

while *leading-edge devices* provide the followings:

- Extension of the linear trait of the lift curve with an increase of the maximum angle of attack and of the c_{Lmax}
- Higher zero-lift angle due to translation of the lift curve on the right caused by leading edge deflection which reduce the actual angle of attack
- Higher slope of the linear trait of the lift curve, for those devices which extend airfoils chords with the effect of increase the wing surface and, with constant wing span, the aspect ratio

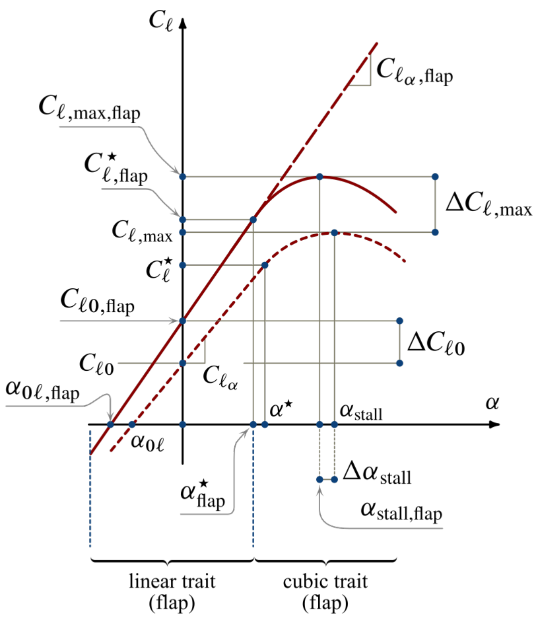

Figure 2.1 Trailing-edge devices effects

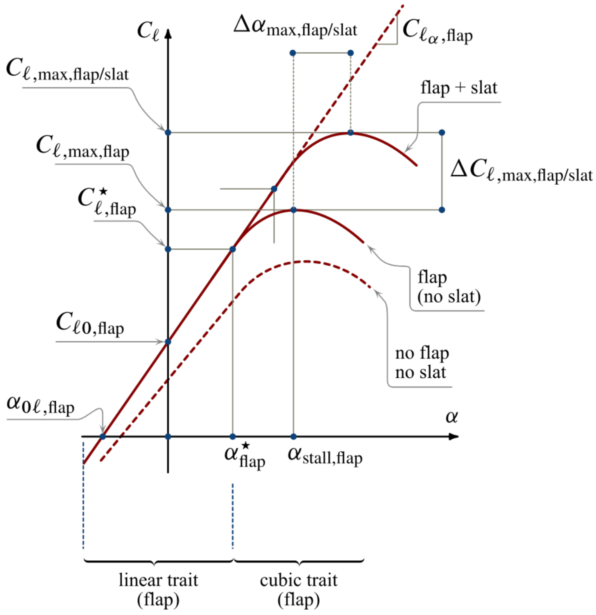

Figure 2.2 Leading-edge devices effects

Among the variety of different devices, only the followings will be taken into account as they represents the most used ones.

Plain flap This device is most used on small aircraft or ones equipped of a thin wing because it doesn't support a complex mechanism of retraction. Typical deflections are about 20° for take-off and 60° for landing.

Single slotted flap It can be seen as a plain flap with a gap between the two elements composing the airfoil. The single slotted flap has very little flap overlap with the fixed trailing edge and hence develops only little Fowler motion, that is the aft travel of the flap that increases the section chord. Its typical deflections are about 20° for take-off and 40° for landing.

The effects of a single slotted flap show an increment in all the aerodynamic coefficients, but it must be said that the increment in drag is lower than that for plain flaps. The slotted flap chord usually ranges from the 25% up to the 30% of the section chord. Moreover the slot influences boundary layer control, in fact it introduces a blowing that energizes the boundary layer delaying separation, so an increase in lift is generated.

Double slotted flap This device is superior to the previous type at large deflections, because separation of the flow over the flap is postponed by the more favourable pressure distribution. Its typical deflections are about 20° for take-off and 50° for landing; in particular, in order to avoid an increasing twisting moment due the deflection, this devices are usually combined with leading edge slats deflected of the same quantity.

Triple slotted flap This device is used on several transport aircraft with very high wing loadings. In combination with leading edge devices, this system represents almost the ultimate achievement in passive high-lift technology, but its shape shows that complicated flap supports and controls are required. Its typical deflections are about 20° for take-off and 40° for landing.

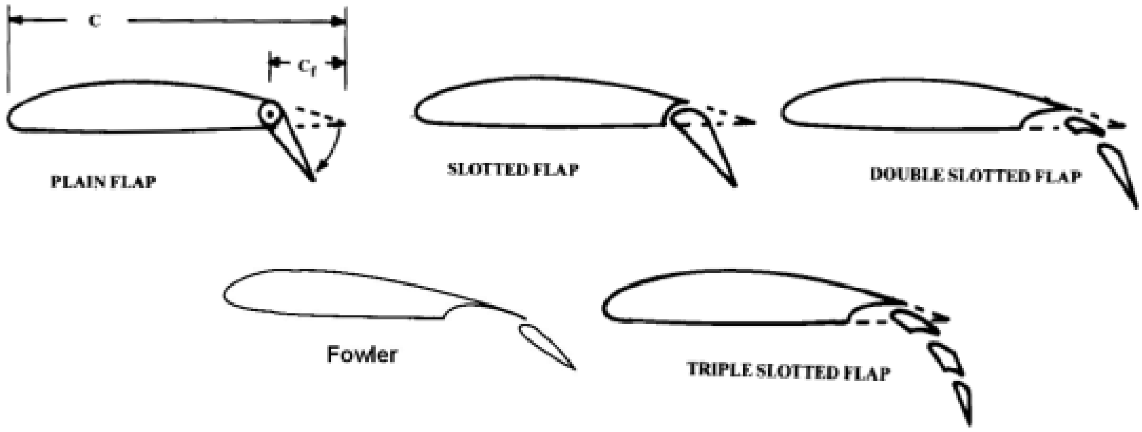

Figure 2.3 Analyzed trailing edge devices types

Fowler flap It is theoretically a single slotted flap that adds to the downward deflection also a backward motion that allows the increment of the effective chord and camber.

Due to the necessity of keeping the rear part of the wing section extended out the main element its implementation system is usually more complicated than the single slotted flaps but its weight and costs are largely justified by its high lift effectiveness. Its typical deflection are about 40° for landing and 15° for take-off, smaller than other types because the chord extension provides a bigger lift increment due to the bigger wing surface; this also reduces the drag in take-off wth benefit an the required field length.

Slat It's the most efficient leading edge device; thanks to combined deflection and forward motion, it acts in order to increase airfoil camber, and so the maximum lift coefficient, as well as increase the airfoil chord with the result of a bigger surface which provides a bigger aspect ratio with the effect of increase the lift curve slope of the linear trait. Furthermore, thanks to the slot which provides a boundary layer energization, it also increases the stalling angle of attack.

Kruger flap It acts in the same way as the slat, but it is thinner and more suitable for installation on thin wings. Krueger flaps are very common because of their simple architecture.

Plain leading edge flap Is less effective than slat since it has no slot, it is mechanically simple and rigid and particularly suitable for thin airfoil sections. The leading edge can be hinged in order to move it backward (droop nose) or it has a mechanism inside that changes the curvature of the nose (variable camber flap).

Leading edge fixed slot It has a fixed slot at the leading edge that, at high angle of attack, allows the airflow to pass through energizing the boundary layer; this helps to increase the stalling angle of attack. During the criuse phase, in which the angle of attack is small, the gap is usually seald.

In order to predict, from the preliminary design phase, the aerodynamic characteristics of the high-lift devices, some useful semi-empirical methods are available; in this particular case the followings formulas and charts are taken from [30] and [26].

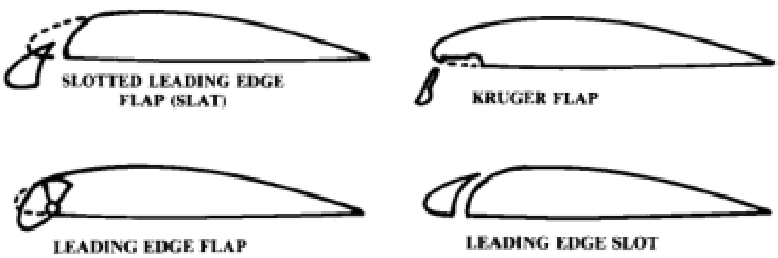

Figure 2.4 Analyzed leading edge devices types

The guideline that will be followed provides to analyze separately the trailing edge and the leading edge devices effects; moreover it will start by evaluating the changes in aerodynamic characteristics of airfoils for then extends these to entire wing.

From figures 2.1 and 2.2, it's possible to understand that the main changes introduced by trailing edge, or leading edge, devices are related to the evaluation of four quantities:

- Δc_{l0}
- $\Delta c_{l\max}$
- $c_{l\alpha,\text{flap}}$
- $\Delta\alpha_{\text{stall}}$

2.1.1 Δc_{l0} and Δc_{L0} calculation

An empirical method for predicting airfoil lift increments at zero angle of attack for high-lift systems (Δc_{l0}) comes from the Glauert's linearized theory for thin airfoils with flaps. A result obtained from this theory for the lift due to flap deflection is the following.

$$\alpha_\delta = 1 - \frac{\theta_f - \sin(\theta_f)}{\pi} \tag{2.1}$$

where

$$\theta_f = \cos^{-1}\left(2\,\frac{c_f}{c} - 1\right) \tag{2.2}$$

Known this value, it's possible to evaluate the theoretical Δc_{l0} which can be calculated as proposed in (2.3).

$$\Delta c_{l0} = \alpha_\delta\; c_{l\alpha}\; \delta_f \tag{2.3}$$

with $c_{l\alpha}$ equals to the linear slope of the llift curve of the airfoil, and δ_f the flap angular deflection.

For large flap deflections and for the separation at large flap angles due to viscosity, linear theory is in error when compared with exact one, for this reason we assume the effectiveness factor η_δ, so the formulation becomes the following.

$$\Delta c_{l0} = \alpha_\delta\; c_{l\alpha}\; \delta_f\; \eta_\delta \tag{2.4}$$

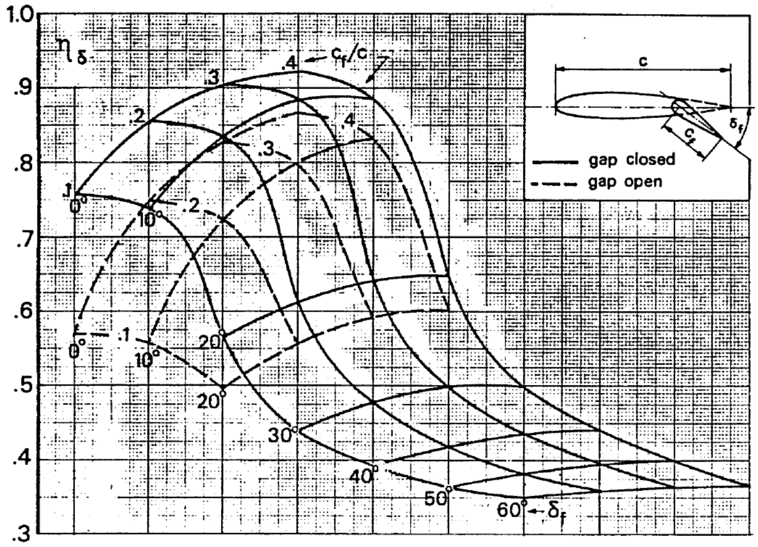

Figure 2.5 η_δ for plain flap

where η_δ can be evaluated from the charts provided in figures 2.5 and 2.6.
In case of flaps which extend the airfoil chord, this effects also contributes to the lift increase and can be taken into account by referring the section lift to the extended chord and then converting the result to the original chord. As a results of this, the section lift coefficient can be evaluated as shown in (2.5).

$$c_l = \left(c'_{l0} + \Delta c'_{l0}\right) \frac{c'}{c} \tag{2.5}$$

where the variables with superscript are referred to the extended chord. Assuming that for the basic section $c'_{l0} = c_{l0}$, it's possible to derive the Δc_{l0} as folllows.

$$\Delta c_{l0} = \Delta c'_{l0} \frac{c'}{c} + c_{l0} \left(\frac{c'}{c} - 1\right) \tag{2.6}$$

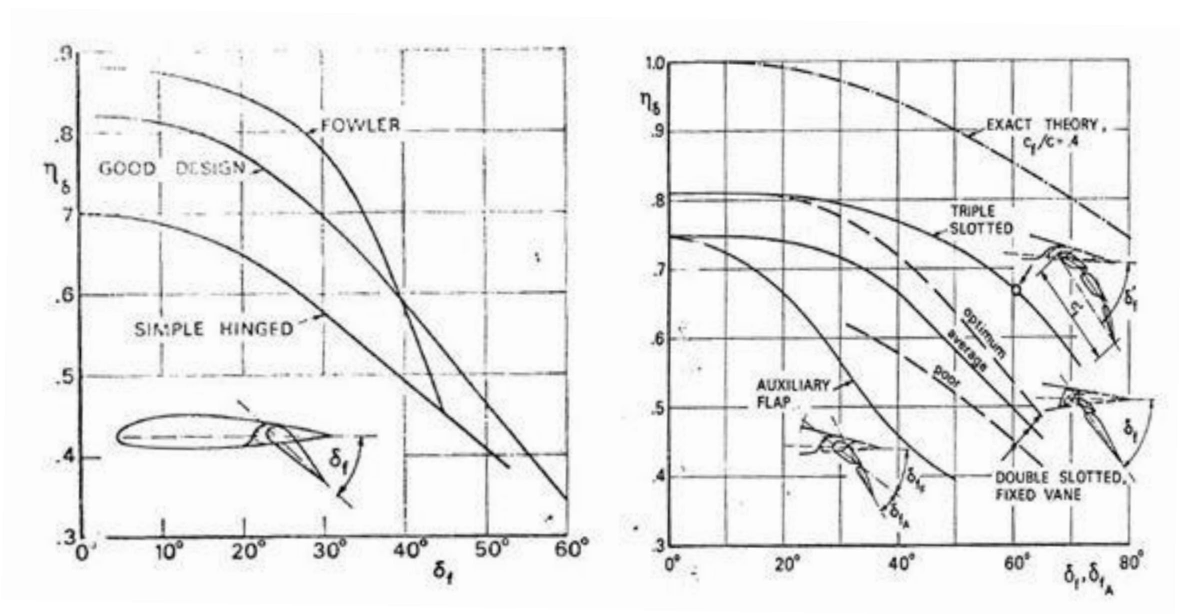

Figure 2.6 η_δ for other type of flaps

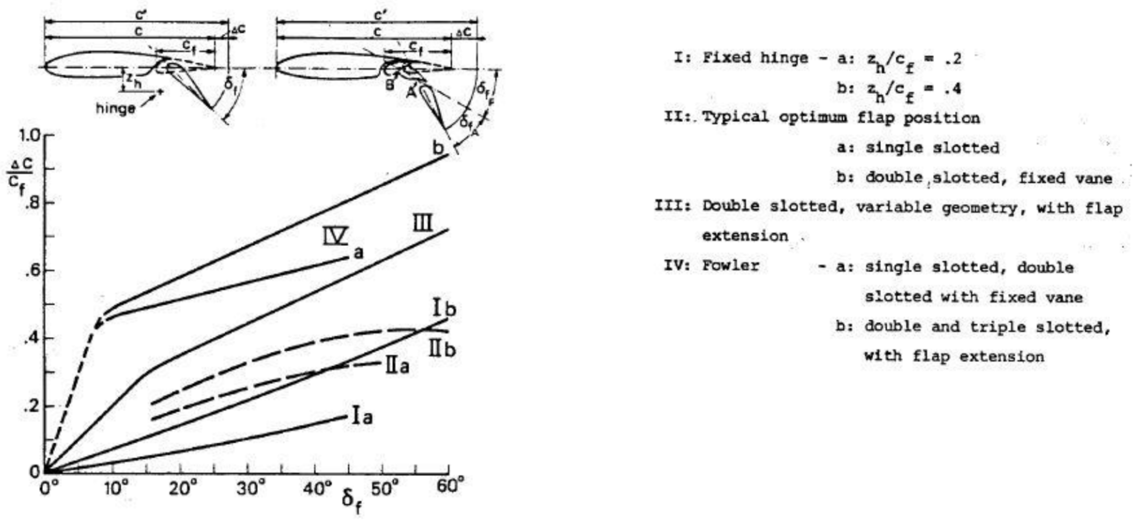

Figure 2.7 $\frac{\Delta c}{c_f}$ for different type of flaps as function of flap deflection

in which c_{l0} is known, $\Delta c'_{l0}$ is calculated as in (2.4) and $\frac{c'}{c}$ is equal to the following expression.

$$\frac{c'}{c} = 1 + \frac{\Delta c}{c_f}\frac{c_f}{c} \tag{2.7}$$

where $\frac{\Delta c}{c_f}$ can be derived from the charts of figure 2.7.
The Δc_{l0} so calculated has now to be extended to the entire wing; this can be through the following forumla.

$$\Delta c_{L0} = \Delta c_{l0}\left(\frac{c_{L\alpha}}{c_{l\alpha}}\right)\left[\frac{(\alpha_\delta)_{c_L}}{(\alpha_\delta)_{c_l}}\right] K_b \tag{2.8}$$

where $c_{L\alpha}$ and $c_{l\alpha}$ are respectively the lift curve slopes of the wing and the airfoil, $\left[\frac{(\alpha_\delta)_{c_L}}{(\alpha_\delta)_{c_l}}\right] = K_c$ is the ratio of the three-dimensional flap effectiveness parameter to the two dimensional flap effectiveness one, which can be derived from the figure 2.8, and K_b is a flap span effectiveness factor, which is function of the flap span-wise extension, that can be read from 2.9.

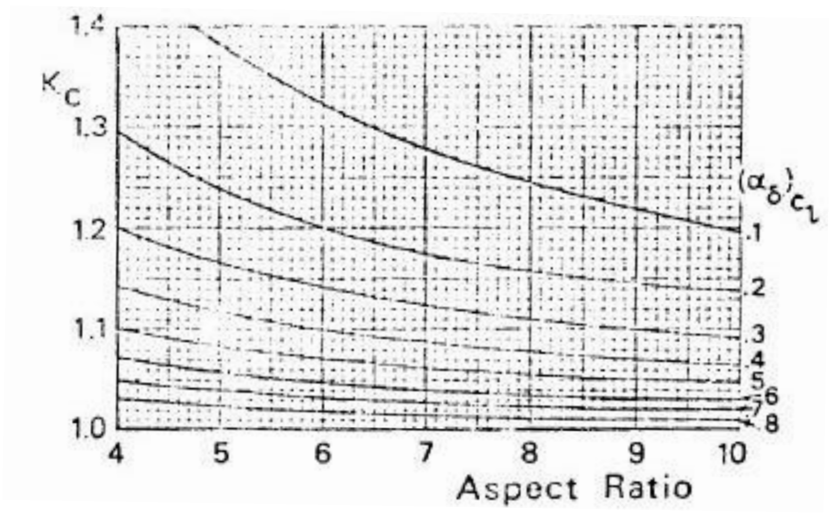

Figure 2.8 K_c for different $(\alpha_\delta)_{c_L}$ as function of wing aspect ratio

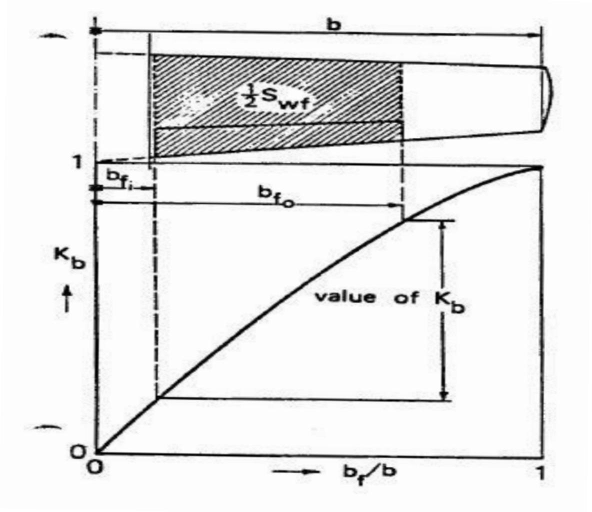

Figure 2.9 K_b as function of flap span to wing span ratio ($\frac{b_f}{b}$)

2.1.2 $\Delta c_{l\mathbf{max}}$ and $\Delta c_{L\mathbf{max}}$ calculation for flaps and slats

2.1.3 $c_{l\alpha,\mathbf{flap}}$ and $c_{L\alpha,\mathbf{flap}}$ calculation

2.1.4 $\Delta\alpha_{\mathbf{stall}}$ calculation for flaps and slats

2.1.5 Further effects calculations

2.2 Java class architecture

2.3 Case study: ATR-72 and B747-100B

Chapter 3

Take-off performance

Appendices

Appendix A

HDF DATABSE CREATION AND READING

A.1 Creation of a database using MATLAB

Creation and mangment of an HDF Dataset are very important to handle because they allow to generate resources which are required by a lot of analysis; for example by using this datasets it's possible to implement a new engine type allowing the analysis of the preformance of a new aircraft. This feature has not been implemented inside JPAD with the purpose of being able to generate the required resources independently.

First of all it's necessary to have curves of the database that has to be digitalized; then, with the use of sotware like *PlotDigitizer*, it's possible to acquire them using a finite number of points chosen by the user. The output of this procedure is a *.csv* file containing all the copule of points which have been used to digitalize the specific curve.

Now the matlab code comes in play to manage these data and to generate the digitalized curves and the HDF dataset. In the example reported there are four curves defined by points through *PlotDigitizer* which have, firstly, been imported in MATLAB generating four *.mat* files; at this point the code interpolates curves points with cubic splines in order to have more points to plot for each curve.

Finally curves are plotted and the HDF Dataset is populated by using *h5create* and *h5write*; in particular curves points, abscissas and parameterization values are attached to the h5 file through these commands.

Listing A.1 MATLAB script for creating the HDF Database

```
clc; close all; clear all;

%% Import data
DeltaAlphaCLmax_vs_LambdaLE_dy1p2 =
    importdata('DeltaAlphaCLmax_vs_LambdaLE_dy1p2.mat');
DeltaAlphaCLmax_vs_LambdaLE_dy2p0 =
    importdata('DeltaAlphaCLmax_vs_LambdaLE_dy2p0.mat');
```

```
DeltaAlphaCLmax_vs_LambdaLE_dy3p0 =
    importdata('DeltaAlphaCLmax_vs_LambdaLE_dy3p0.mat');
DeltaAlphaCLmax_vs_LambdaLE_dy4p0 =
    importdata('DeltaAlphaCLmax_vs_LambdaLE_dy4p0.mat');

nPoints = 30;
lambdaLEVector_deg = transpose(linspace(0, 40, nPoints));

%% dy/c = 1.2
smoothingParameter = 0.999999;
DAlphaVsLambdaLESplineStatic_Dy1p2 = csaps( ...
    DeltaAlphaCLmax_vs_LambdaLE_dy1p2(:,1), ...
    DeltaAlphaCLmax_vs_LambdaLE_dy1p2(:,2), ...
    smoothingParameter);
DAlphaVsLambdaLEStatic_Dy1p2 = ppval( ...
    DAlphaVsLambdaLESplineStatic_Dy1p2, ...
    lambdaLEVector_deg);

%% dy/c = 2.0
smoothingParameter = 0.999999;
DAlphaVsLambdaLESplineStatic_Dy2p0 = csaps( ...
    DeltaAlphaCLmax_vs_LambdaLE_dy2p0(:,1), ...
    DeltaAlphaCLmax_vs_LambdaLE_dy2p0(:,2), ...
    smoothingParameter);
DAlphaVsLambdaLEStatic_Dy2p0 = ppval( ...
    DAlphaVsLambdaLESplineStatic_Dy2p0, ...
    lambdaLEVector_deg);

%% dy/c = 3.0
smoothingParameter =0.999999;
DAlphaVsLambdaLESplineStatic_Dy3p0 = csaps( ...
    DeltaAlphaCLmax_vs_LambdaLE_dy3p0(:,1), ...
    DeltaAlphaCLmax_vs_LambdaLE_dy3p0(:,2), ...
    smoothingParameter);
DAlphaVsLambdaLEStatic_Dy3p0 = ppval( ...
    DAlphaVsLambdaLESplineStatic_Dy3p0, ...
    lambdaLEVector_deg);

%% dy/c = 4.0
smoothingParameter = 0.999999;
DAlphaVsLambdaLESplineStatic_Dy4p0 = csaps( ...
    DeltaAlphaCLmax_vs_LambdaLE_dy4p0(:,1), ...
    DeltaAlphaCLmax_vs_LambdaLE_dy4p0(:,2), ...
    smoothingParameter);
DAlphaVsLambdaLEStatic_Dy4p0 = ppval( ...
    DAlphaVsLambdaLESplineStatic_Dy4p0, ...
    lambdaLEVector_deg);

%% Plots
figure(1)
plot (lambdaLEVector_deg, DAlphaVsLambdaLEStatic_Dy1p2, '-*b' ... , ...);
hold on;
plot (lambdaLEVector_deg, DAlphaVsLambdaLEStatic_Dy2p0, '-b' ... , ...);
```

```
plot (lambdaLEVector_deg, DAlphaVsLambdaLEStatic_Dy3p0, '*b' ... , ...);
plot (lambdaLEVector_deg, DAlphaVsLambdaLEStatic_Dy4p0, 'b' ... , ...);
xlabel('\Lambda_{le} (deg)'); ylabel('\Delta\alpha_{C_{L,max}}');
title('Angle of attack increment for wing maximum lift in subsonic flight');
legend('\Delta y/c = 1.2', '\Delta y/c = 2.0', '\Delta y/c = 3.0','\Delta y/c =
    4.0');
axis([0 50 0 9]);
grid on;

%% preparing output to HDF
% dy/c
dyVector = [1.2;2.0;3.0;4.0];
%columns --> curves
myData = [ ...
        DAlphaVsLambdaLEStatic_Dy1p2, ...
        DAlphaVsLambdaLEStatic_Dy2p0, ...
        DAlphaVsLambdaLEStatic_Dy3p0, ...
        DAlphaVsLambdaLEStatic_Dy4p0];

hdfFileName = 'DAlphaVsLambdaLEVsDy.h5';
if ( exist(hdfFileName, 'file') )
    fprintf('file %s exists, deleting and creating a new one\n', hdfFileName);
    delete(hdfFileName)
else
    fprintf('Creating new file %s\n', hdfFileName);
end
% Dataset: data
h5create(hdfFileName, '/DAlphaVsLambdaLEVsDy/data', size(myData'));
h5write(hdfFileName, '/DAlphaVsLambdaLEVsDy/data', myData');
% Dataset: var_0
h5create(hdfFileName, '/DAlphaVsLambdaLEVsDy/var_0', size(dyVector'));
h5write(hdfFileName, '/DAlphaVsLambdaLEVsDy/var_0', dyVector');
% Dataset: var_1
h5create(hdfFileName, '/DAlphaVsLambdaLEVsDy/var_1', size(lambdaLEVector_deg'));
h5write(hdfFileName, '/DAlphaVsLambdaLEVsDy/var_1', lambdaLEVector_deg');
```

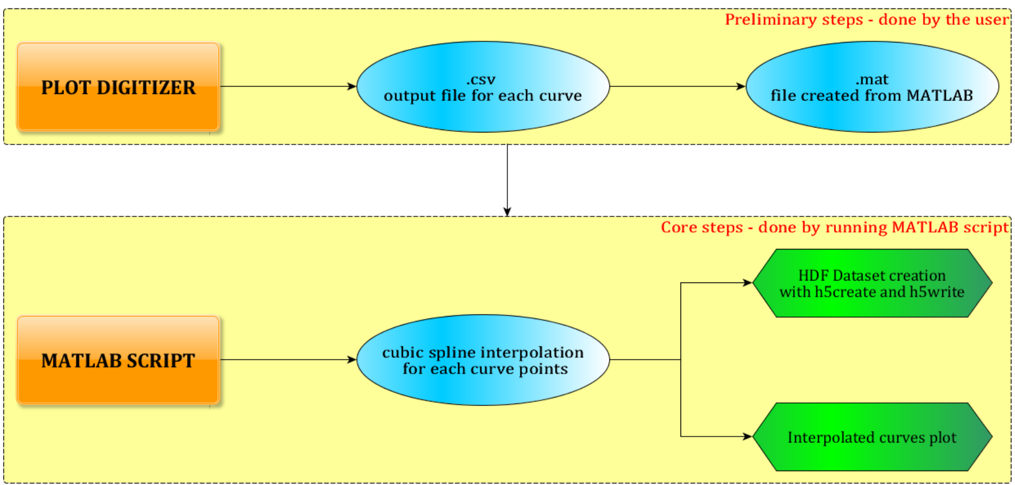

Figure A.1 Flowchart of an HDF Database creation

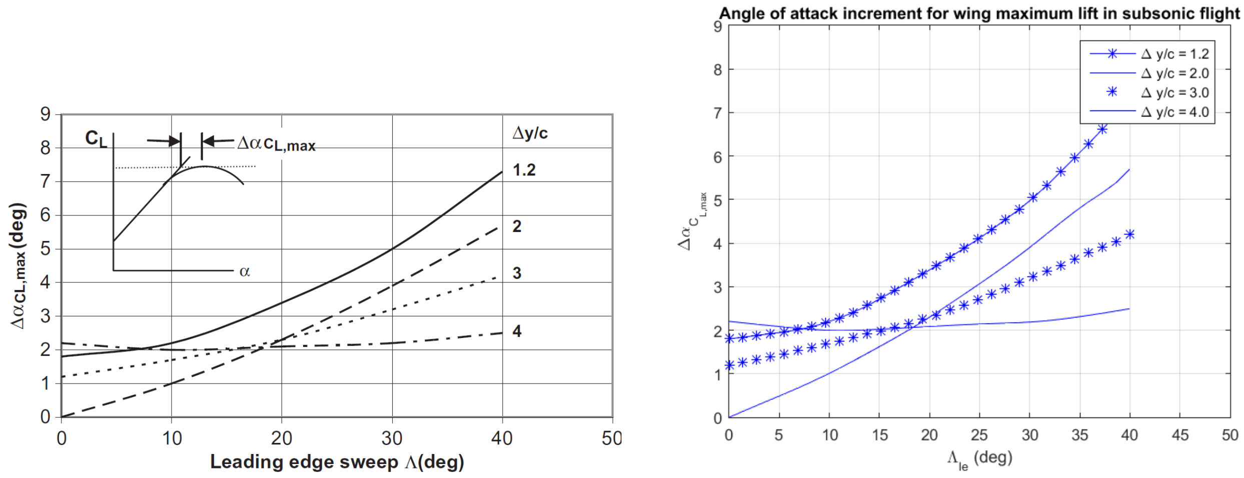

Figure A.2 Comparison between the initial graph and the digitized one

A.2 Reading data from an HDF database in JPAD

After creating the database, this has to be read in order to obtain the required data; inside JPAD this operation can be done defining a specific class, which extends the abstract class `DatabaseReader` that is designed for HDF dataset reading.

This son class has a specific structure which main key points can be summarized in the following ones:

1. Creation of variables, in number equal to the function to be interpolated, using the type `MyInterpolatingFunction`
2. Creation of variables for all values that are wanted to be read from the interpolating functions
3. Creation of a builder that accepts the folder path string and the file name string of the database. This builder has to launch the interpolating method for all functions contained into the database by using `MyInterpolatingFunction` methods.
4. Creation of a getter method for each of the variables allocated at point 2 in order to obtain values from interpolated functions by giving in input the required parameters

In particular the class `MyInterpolatingFunction` implements methods for a spline, bicubic and tricubic data interpolation as well as three methods for extracting a specific value from each of the previou interpolated curve.

The following listing describes, with an example upon the aerodynamic database, how the reader class should be build following the previous steps.

Listing A.2 DatabaseReader son class creation

```
public class AerodynamicDatabaseReader extends DatabaseReader {
// STEP 1:
  private MyInterpolatingFunction
          c_m0_b_k2_minus_k1_vs_FFR,
          ar_v_eff_c2_vs_Z_h_over_b_v_x_ac_h_v_over_c_bar_v;
```

```java
// STEP 2:
  double cM0_b_k2_minus_k1, ar_v_eff_c2;
// STEP 3:
  public AerodynamicDatabaseReader(String databaseFolderPath, String databaseFileName) {
    super(databaseFolderPath, databaseFileName);

    c_m0_b_k2_minus_k1_vs_FFR =
            database.interpolate1DFromDatasetFunction(
                    "(C_m0_b)_k2_minus_k1_vs_FFR"
            );
    ar_v_eff_c2_vs_Z_h_over_b_v_x_ac_h_v_over_c_bar_v =
            database.interpolate2DFromDatasetFunction(
                    "(AR_v_eff)_c2_vs_Z_h_over_b_v_(x_ac_h--v_over_c_bar_v)"
                    );
  }
// STEP 4:
  public double get_C_m0_b_k2_minus_k1_vs_FFR(double length, double diameter) {
    return c_m0_b_k2_minus_k1_vs_FFR.value(length/diameter);
  }
// STEP 4:
  public double get_AR_v_eff_c2_vs_Z_h_over_b_v_x_ac_h_v_over_c_bar_v(double zH, double bV, double xACHV, double cV) {
    return ar_v_eff_c2_vs_Z_h_over_b_v_x_ac_h_v_over_c_bar_v.value(zH/bV, xACHV/cV);
  }
}
```

Bibliography

[1] L. Attanasio. «Development of a Java Application for Parametric Aircraft Design». Master thesis. University of Naples "Federico II", 2014.

[2] Various Authors. *ATR 72*. URL: https://en.wikipedia.org/wiki/ATR_72.

[3] Various Authors. *Boeing 747*. URL: https://en.wikipedia.org/wiki/Boeing_747.

[4] Various Authors. *eXtensible Markup Language (XML)*. URL: https://en.wikipedia.org/wiki/XML.

[5] S. Ciornei. *Mach number, relative thickness, sweep and lift coefficient of the wing - An empirical investigation of parameters and equations*. Paper. 31.05.2005.

[6] Oracle Corporation. *List*. URL: https://docs.oracle.com/javase/8/docs/api/java/util/List.html.

[7] Jean-Marie Dautelle. *Jscience*. URL: http://jscience.org/.

[8] E. De Bono. «Prestazioni di decollo e atterraggio. Calcolo approssimato e verifica sperimentale.» Thesis. University of Naples "Federico II", 2011.

[9] D. et al Gilbert. *JFreeChart*. URL: http://www.jfree.org/jfreechart/.

[10] The HDF Group. *HDF*. URL: https://www.hdfgroup.org/products/java/hdf-java-html/javadocs/ncsa/hdf/hdf5lib/H5.html.

[11] W.F. Hilton. *High-speed Aerodynamics*. Longmans, Green, 1951.

[12] D. Howe. *Aircraft conceptual design synthesis*. Aerospace Series. Professional Engineering Publishing, 2000.

[13] J. Huwaldt. *1976 Standard Atmosphere Program*. URL: http://thehuwaldtfamily.org/java/Applications/StdAtmosphere/StdAtmosphere.html.

[14] P. Jackson and FRAeS. *Jane's All the World's Aircraft*. Jane's All the World's Aircraft. Jane's Information Group, 2004-2005.

[15] L.R. Jenkinson, D. Rhodes, and P. Simpkin. *Civil jet aircraft design*. AIAA education series. Arnold, 1999.

[16] K. Kawaguchi. *Args4j*. URL: http://args4j.kohsuke.org/.

[17] I. Kroo and R. Shevell. *Aircraft Design: Synthesis and Analysis*. 1999.

[18] A. Lausetti. *Decollo e atterramento di aeroplani, idrovolanti trasportati*. Levrotto & Bella Editrice S.a.s., 1992.

[19] B. W. McCormick. *Aerodynamics, Aeronautics, and Flight Mechanics*. John Wiley & Sons, 1979.

[20] L.M. Nicolai and G. Carichner. *Fundamentals of Aircraft and Airship Design*. AIAA education series v. 1. American Institute of Aeronautics and Astronautics, 2010.

[21] F. Nicolosi and G. Paduano. *Behind ADAS*. 2011.

[22] E. Obert. *Aerodynamic Design of Transport Aircraft*. IOS Press, 2009.

[23] D.P. Raymer. *Aircraft Design: A Conceptual Approach*. 1992.

[24] D.P. Raymer. *Aircraft Design / RDS-Student: A Conceptual Approach*. AIAA Education Series. Amer Inst of Aeronautics &, 2013.

[25] M.H. Sadraey. *Aircraft Design: A Systems Engineering Approach*. Aerospace Series. Wiley, 2012.

[26] P.M. Sforza. *Commercial Airplane Design Principles*. Elsevier Science, 2014.

[27] E. Torenbeek. *Advanced Aircraft Design: Conceptual Design, Technology and Optimization of Subsonic Civil Airplanes*. Aerospace Series. Wiley, 2013.

[28] E. Torenbeek. *Optimum Cruise Performance of Subsonic Transport Aircraft*. Delft University Press, 1998.

[29] E. Torenbeek. *Synthesis of Subsonic Aircraft Design*. 1976.

[30] E. Torenbeek. *Synthesis of Subsonic Airplane Design: An Introduction to the Preliminary Design of Subsonic General Aviation and Transport Aircraft, with Emphasis on Layout, Aerodynamic Design, Propulsion and Performance*. Springer, 1982.

[31] A.D. Young. *The Aerodynamic Characteristics of Flaps*. Technical Report. 1947.

Glossary

Autonomy Factor Is the combination of three main efficiency: the propulsive efficiciency represented by SFC, the propeller efficiency η_p or the jet efficiency represented by V and, finally, the aerodynamic efficinecy $\frac{L}{D}$.

Direct Operative Cost The totality of aircraft costs directly connected to the aircraft flight. It can be seen as the amount of money necessary to carry 1 ton of payload upon 1 km.

Hierarchical Data Format A set of file formats (HDF4, HDF5) designed to store and organize large amounts of data.

Java Program toolchain for Aircraft Design Collection of libraries and classes with the aim of providing complete aircraft preliminary design analyses through the use of several semi-empirical formulas tested against experimental data.

List The java.util.List interface is a subtype of the java.util.Collection interface. It represents an ordered list of objects, meaning you can access the elements of a List in a specific order, and by an index too. You can also add the same element more than once to a List.

Map The java.util.Map interface represents a mapping between a key and a value. The Map interface is not a subtype of the Collection interface. Therefore it behaves a bit different from the rest of the collection types.

Parsing Parsing or syntactic analysis is the process of analysing a string of symbols, either in natural language or in computer languages, conforming to the rules of a formal grammar.

Portable Network Graphics A raster graphics file format that supports lossless data compression. PNG was created as an improved, non-patented replacement for Graphics Interchange Format (GIF), and is the most used lossless image compression format on the Internet.

Static method The term static means that the method is available at the Class level, and so does not require that an object is instantiated before it's called.

Table Is a collection that associates an ordered pair of keys, called a row key and a column key, with a single value. A table may be sparse, with only a small fraction of row key / column key pairs possessing a corresponding value.

TikZ Is a set of higher-level macros that use PGF.

True AirSpeed Is the speed of the aircraft relative to the airmass in which it is flying; TAS is the true measure of aircraft performance in cruise, thus it is the speed listed in aircraft specifications, manuals, performance comparisons, pilot reports, and every situation when cruise or endurance performance needs to be measured. It is the speed normally listed on the flight plan, also used in flight planning, before considering the effects of wind.

User developer The term refers to the developer which will use a method without being interested in how the method performs the required action. This is the case of a utility method: the developer is the one who writes the method, while the user developer is who uses that method to accomplish some action which requires the functionality provided by the utility method. It has to be noticed that the user developer and the developer can be the same person.

Acronyms

A.F. Autonomy Factor.

DOC Direct Operative Cost.

HDF Hierarchical Data Format.

JPAD Java Program toolchain for Aircraft Design.

MZFW Maximum Zero Fuel Weight.

PNG Portable Network Graphics.

SFC Specific Fuel Consumption.

SFCJ Jet Specific Fuel Consumption.

TAS True AirSpeed.

TikZ TikZ.

UAV Unmanned Aerial Vehicle.

List of symbols

$\mathcal{R}$ aspect ratio.

$()_{cruise}$ quantity related to cruise condition.

η_p propeller efficiency.

$()_{\mathbf{f}}$ quantity related to flaps.

$()_{\mathbf{LG}}$ quantity related to the landing gear.

b span.

$C_{\mathbf{D}}$ aerodynamic drag coefficient.

$C_{\mathbf{L}}$ aerodynamic lift coefficient.

$C_{\mathbf{D0}}$ aerodynamic parasite drag coefficient.

D aerodynamic drag.

$\boldsymbol{g}$ gravitational acceleration.

$i_{\mathbf{W}}$ the angle between the wing root chord and the ACRF x-axis.

L aerodynamic lift.

M Mach number.

$M_{\mathbf{ff}}$ Fuel fraction over entire mission.

c chord.

Λ sweep.

λ taper ratio.

t thickness.

n load factor.

R range in nmi or km.

S surface.

T thrust.

V scalar velocity.

$W_{\mathbf{OE}}$ Operating Empty Weight.

$W_{\mathbf{TO}}$ Take Off Weight.

$W_{\mathbf{fuel}}$ Fuel Weight.

$W_{\mathbf{Payload}}$ Payload Weight.

W weight, in N or lbf.

$\alpha_{\mathbf{W}}$ angle of attack referred to wing root chord.

ρ air density.