JAVAFOIL User's Guide

Martin Hepperle
5-December-2011

Contents

JAVAFOIL

JAVAFOIL is a relatively simple program, which uses several traditional methods for the analysis of airfoils in subsonic flow.
The main purpose of JAVAFOIL is to determine the lift, drag and moment characteristics of airfoils. The program will first calculate the distribution of the velocity on the surface of the airfoil. For this purpose it uses a potential flow analysis module which is based on a higher order panel method (linear varying vorticity distribution). This local velocity and the local pressure are related by the Bernoulli equation. In order to find the lift and the pitching moment coefficient the distribution of the pressure can be integrated along the surface.

Next JAVAFOIL will calculate the behavior of the flow layer close to the airfoil surface (the boundary layer). The boundary layer analysis module (a so called integral method) steps along the upper and the lower surfaces of the airfoil, starting at the stagnation point. It solves a set of differential equations to find the various boundary layer parameters. The boundary layer data is then be used to calculate the drag of the airfoil from its properties at the trailing edge.
Both analysis steps are repeated for each angle of attack, which yields a complete polar of the airfoil for one fixed Reynolds number.

Additional tools for the creation and modification of airfoils have been added to fill the toolbox. These tools are wrapped in a Graphical User Interface (GUI) which was designed to be easy to use and not overly complicated. The GUI is organized into a stack of cards, which will be described later.

All calculations are performed by a computer code of my own. JAVAFOIL is neither a rewrite of Eppler' PROFIL nor of Drela's XFOIL program. The boundary layer module is based on the same equations which are also used in the initial version of the Eppler program. Additions include new stall and transition models. The panel method was developed with the help of the extensive survey of panel methods found in [14].
Compared with similar programs, JAVAFOIL can also handle multi-element airfoils and also simulate ground effect.

Limitations

As already noted, JAVAFOIL is a relatively simple program with some limitations. Like with all engineering computer codes, it is up to the user to judge and to decide how far he wants to trust a program.
As JAVAFOIL does not model laminar separation bubbles and flow separation, its results will become inaccurate if such effects occur. The boundary layer method does not include any feedback to the potential flow solution, which means that it is limited to mostly attached flows. Flow separation, as it occurs at stall, is modeled to some extent by empirical corrections, so that maximum lift can be estimated for conventional airfoils. If you analyze an airfoil beyond stall, the results will be quite inaccurate. On the other hand, it is somewhat questionable, whether any two-dimensional analysis method can be used at all in this regime, as the flow field beyond stall becomes fully three dimensional with spanwise flow and strong vortices developing.
In the case of multi element airfoils, one must be aware, that in the real world very complex flows can develop due to interaction of trailing wakes and the boundary layers of the individual elements or if the boundary layers separate locally. An accurate analysis would require a more sophisticated solver for the Navier-Stokes equations, which would also imply an increase in computer time in the order of 1000. Nevertheless a simple tool like JAVAFOIL can be helpful to estimate the main effects and to improve a design to avoid suction peaks and flow separation.

JAVAFOIL's Cards

The user interface of JAVAFOIL is divided into a stack of cards. Each card contains interface elements for a specific task. The content of some cards is also relevant for actions executed on other cards, for example the Mach number specified on the Options card affects the analyses on all other cards.

Geometry Card

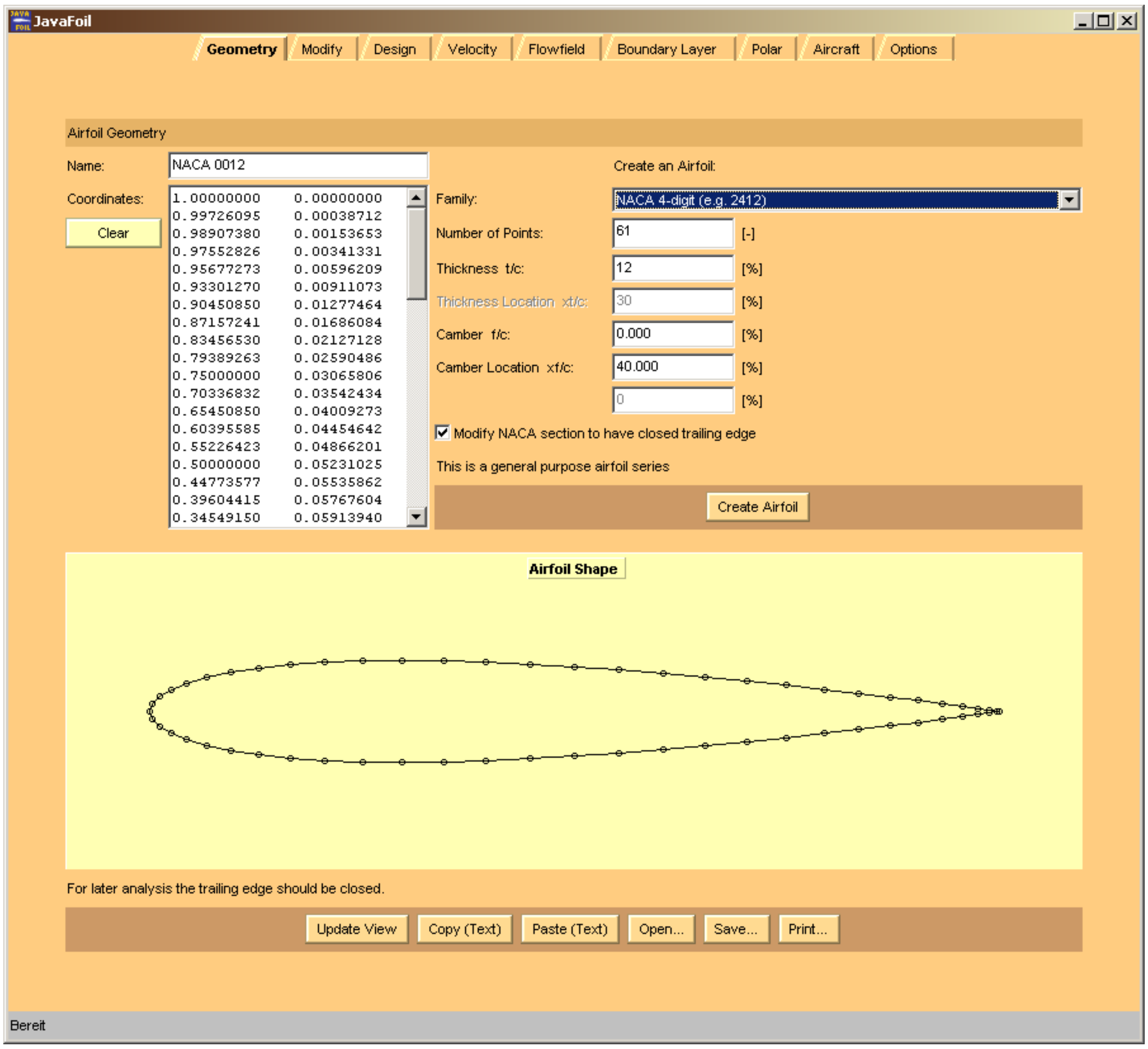
The Geometry card is used to store and prepare the geometry of your airfoil. It contains the “current” or “working” airfoil. The geometry is described by a set of coordinate points, each having an x and a y value. The working airfoil is used and modified by the actions you perform in JAVAFOIL.
The Geometry card shows a list of x- and y-coordinate pairs and a plot of the resulting airfoil shape. You can enter or paste arbitrary coordinates into this field and press the “Update View” button to copy the coordinates into the working airfoil.

The coordinates must be ordered so that they describe the shape in a continuous sequence. The order must be “trailing edge” → “upper surface” → “nose” → “lower surface” → “trailing edge”.

JAVAFOIL comes with a set of shape generators for a variety of airfoils which is accessible from this card. These airfoils represent classical airfoil sections for which analytical descriptions exist (e.g. NACA sections) or which can be constructed from geometrical constraints (e.g. wedge sections). Despite their age, many classical airfoil sections are still applicable to many problems or form a good starting point for new developments.

Today, modern airfoil sections are usually developed for specific purposes and their shapes are usually not published. More recent developments lead towards the direct design of three-dimensional wing shapes, eliminating the classical steps of two-dimensional airfoil design and three-dimensional wing lofting. In most cases, modern airfoil sections are not described anymore by analytical formulas, just by a set of points.

The row of buttons at the bottom allows for copying, saving, loading and printing of airfoil coordinate sets.



Name: NACA 0012

Coordinates:

```
1.00000000   0.00000000
0.99726095   0.00038712
0.98907380   0.00153653
0.97552826   0.00341331
0.95677273   0.00596209
0.93301270   0.00911073
0.90450850   0.01277464
0.87157241   0.01686084
0.83456530   0.02127128
0.79389263   0.02590486
0.75000000   0.03065806
0.70336832   0.03542434
0.65450850   0.04009273
0.60395585   0.04454642
0.55226423   0.04866201
0.50000000   0.05231025
0.44773577   0.05535862
0.39604415   0.05767604
0.34549150   0.05913940
```

Family: NACA 4-digit (e.g. 2412)

Number of Points: 61 [-]

Thickness t/c: 12 [%]

Thickness Location xt/c: 30 [%]

Camber f/c: 0.000 [%]

Camber Location xf/c: 40.000 [%]

0 [%]

☑ Modify NACA section to have closed trailing edge

This is a general purpose airfoil series

For later analysis the trailing edge should be closed.

Bereit

View of JAVAFOIL's Geometry card.

Exporting airfoil geometry

JAVAFOIL can write airfoil geometry from the following file types:

- `*.txt`
  multi-element airfoil geometry in form of simple two columnar x-y coordinate sets arranged in two columns. Multi-element airfoils must be separated by a pair of x and y-values larger than 999.
- `*.xml`
  multi-element airfoil geometry in JAVAFOIL's hierarchically structured `xml` format.
- `*.dxf`
  multi-element airfoil geometry in AutoCad drawing exchange format. Many CAD programs can read this file format, but the interpretation is not always perfect.
- `*.igs or *.iges`
  multi-element airfoil geometry in Initial Graphics Exchange Standard format. Many CAD programs can read this file format.

Note that JAVAFOIL selects the output file format according to the file name extension.

Importing airfoil geometry

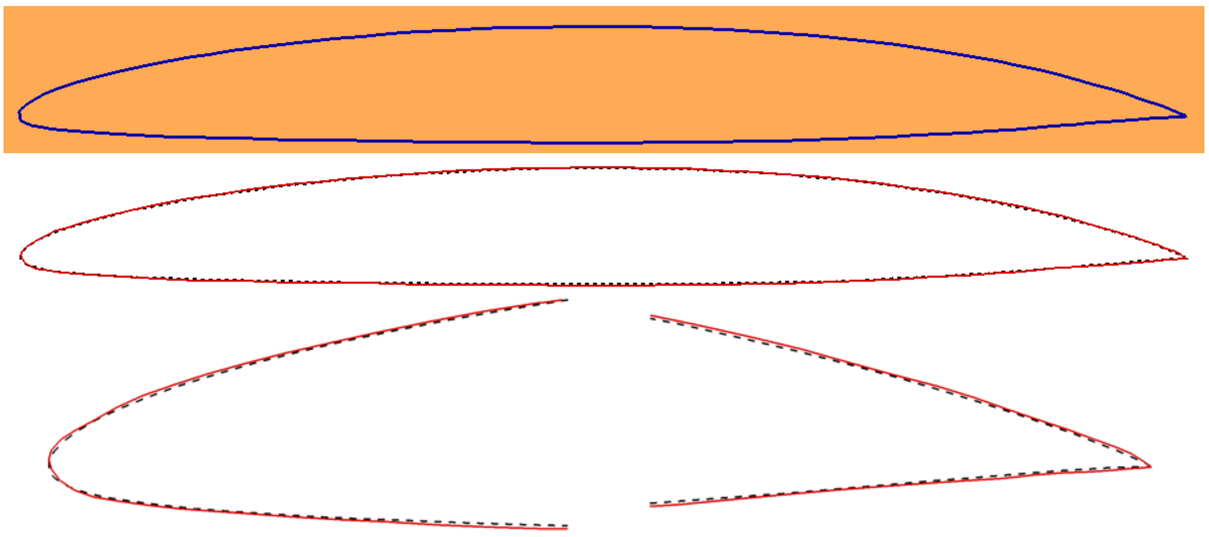
JAVAFOIL can read airfoil geometry from the following file types:

- `*.txt`
  multi-element airfoil geometry in form of simple two columnar x-y coordinate sets arranged in two columns. Multi-element airfoils must be separated by a pair of x and y-values larger than 999.
- `*.xml`
  multi-element airfoil geometry in hierarchically structured xml format.
- `*.png, *.gif, *.bmp, *.jpg`
  single element airfoil geometry from an image file.

Note that JAVAFOIL selects the input file format according to the file name extension.

Importing scanned images

You can also load an airfoil from a bitmap image in GIF, PNG, BMP or JPG format. JAVAFOIL then tries to find an airfoil shape in this image by comparing the image points with the color found in the upper left corner of the image. Therefore the image should have no border, and a monochrome background. Before scanning the image, a smoothing filter is applied to remove spurious points from the image. To achieve acceptable results an image width of 1000 or more pixels is recommended. The interior of the airfoil shape can be empty or arbitrarily filled, because the algorithm searches from the top and bottom edges of the image and stops when it detects the border of the shape. The resulting points are filtered again to improve the smoothness of the shape. Nevertheless the results will not be perfect, but this method can be considered as a last resort to quickly determine airfoil coordinates if only a scanned image is available. It is recommended to inspect the resulting velocity distribution and to use the inverse design method for smoothing the airfoil shape.



Airfoil image (top) and comparison of the original (dashed) and the reconstructed airfoil shape (solid) using JAVAFOIL's bitmap import capability on the Geometry card.

JAVAFOIL's Geometry Generators

General Remarks on NACA airfoils

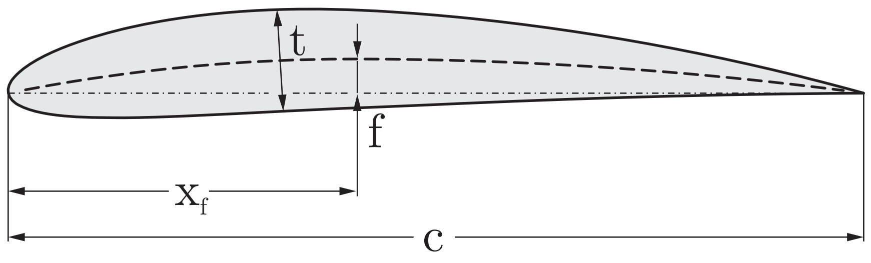
The construction of the cambered NACA airfoil sections requires that the thickness distribution is erected at right angles to the camber line. Some computer programs do not follow this construction principle and add the thickness just to the y-coordinates of the camber line. This leads to larger deviations from the true airfoil section when the camber line is

inclined, e.g. close to the leading edge or close to the trailing edge of airfoils with a high amount of aft camber.
The construction method may lead to points extending slightly into the negative x-range, when a large amount of camber is located close to the leading edge. This is a correct behavior and an expected result.
Note also that most NACA sections have a thick trailing edge by definition. In order to produce a thin, sharp trailing edge, JAVAFOIL has an option to close the airfoil shape by bending the upper und lower surfaces to close the trailing edge.

NACA 4-digit airfoils

The calculation of these classical airfoils is easy because their shape and the associated camber lines are defined by rather simple formulas. The maximum thickness is located at $x/c = 30\%$, whereas the maximum camber is typically located at $x/c = 40\%$. See [3] and [4] for more details.

The camber lines are composed of two parabolic arcs, which are joined with equal tangents, but a kink in the curvature. This kink can be seen in the velocity distributions, especially when the position of the maximum camber is different from the common 40% chord station.



Parameters:

- Free: t/c, f/c, x_f/c
- Fixed $x_t/c = 0.3$

Naming Scheme

The first two integers define the camber line, while the last two integers define the thickness.

- 1st digit: maximum ordinate of camber $100 \cdot f/c$
- 2nd digit: location of maximum camber $10 \cdot x_f/c$
- 3rd and 4th digit: maximum thickness $100 \cdot t/c$

Example:
NACA 2412: 2% camber at 40% chord, 12% thickness

The thickness distribution for the 10% thick section is given by the polynomial:

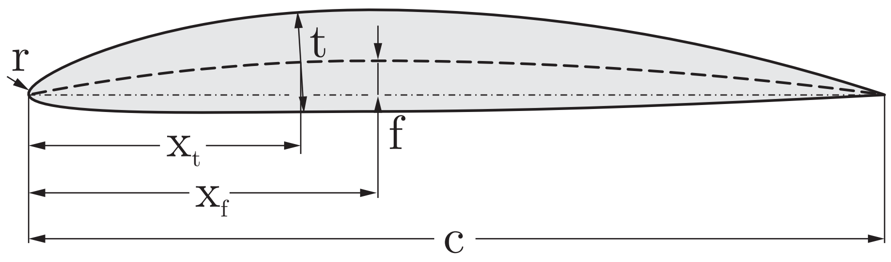
$$y = \pm\left(0.29690 \cdot \sqrt{x} - 0.12600 \cdot x - 0.35160 \cdot x^2 + 0.28530 \cdot x^3 - 0.10150 \cdot x^4\right)$$

The coefficients of this thickness distribution had been chosen according to the following constraints [4] (for a 10% thick section):

- maximum thickness at $x/c = 0.3 \rightarrow \frac{\partial y}{\partial x}(0.3) = 0$,
- finite thickness at trailing edge $\frac{y}{c} = 0.004$,
- finite trailing edge angle at $x/c = 1.0 \rightarrow \frac{\partial y}{\partial x}(1.0) = -0.234$,
- nose shape defined by $y/c = 0.078$ at $x/c = 0.1$.

Modified NACA 4-digit airfoils

The modification adds the position of the maximum thickness as well as the nose radius to the parameter set of the 4-digit series (see [3]).



Parameters:

- Free: t/c, f/c, x_t/c, x_f/c, r

Naming Scheme

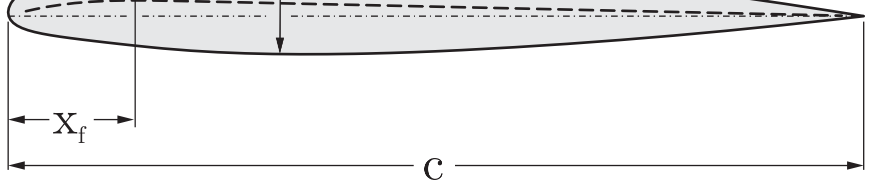
The name consists of a 4 digit prefix which is identical to the 4-digit series designation, followed by a dash and two additional digits.

- 1st digit: maximum ordinate of camber $100 \cdot f/c$
- 2nd digit: position of the maximum camber $10 \cdot x_f/c$
- 3rd and 4th digit: maximum thickness $100 \cdot t/c$
- dash
- 5th digit: indicates the leading edge radius and is usually one of 0, 3, 6, or 9:
  - 0: sharp leading edge
  - 3: the normal radius of the 4-digit series,
  - 6: ¼ normal radius,
  - 9: 3 times the normal radius.
- 6th digit: position of the maximum thickness $10 \cdot x_t/c$

Example:
NACA 1410-35: 1% camber at 40% chord, 10% thickness, reduced leading edge radius, maximum thickness at 50% x/c

NACA 5-digit airfoils

These sections use the same thickness distributions as the 4-series, but have new camber lines leading to lower pitching moments. The camber line is composed of a cubic in the forward part to which a straight line is attached towards the rear. Instead of the camber f/c, a design lift coefficient $C_{\ell\ \mathrm{design}}$ is now used to define the maximum height of the camber line. In practical applications, these airfoils are often used with a maximum camber at $x/c = 0.15$, i.e. relatively far forward.



Parameters:

- Free: t/c, x_f/c, $C_{\ell\ \mathrm{design}}$

- Fixed $x_t / c = 0.3$

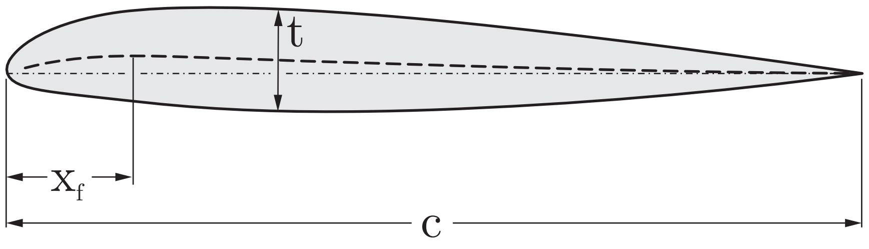
Naming Scheme

- 1st digit: design $10 \cdot 2/3 \cdot C_{\ell \text{ design}}$
- 2nd and 3rd digit: $2 \cdot 100 \cdot x_f / c$. Note that the 3rd digit is usually a zero, i.e. the position of the maximum camber is a multiple of 5%.
- 4th and 5th digit: maximum thickness $100 \cdot t / c$

Example:
NACA 23012: design lift coefficient 0.3, maximum camber at 15% chord, 12% thickness.

Modified NACA 5-digit airfoils

The rear part of the camber line of these sections has been modified to a cubic curve which provides some reflex. Therefore the pitching moments are reduced further or may become even positive.



Parameters:

- Free: t / c, x_f / c, $C_{\ell \text{ design}}$
- Fixed $x_t / c = 0.3$

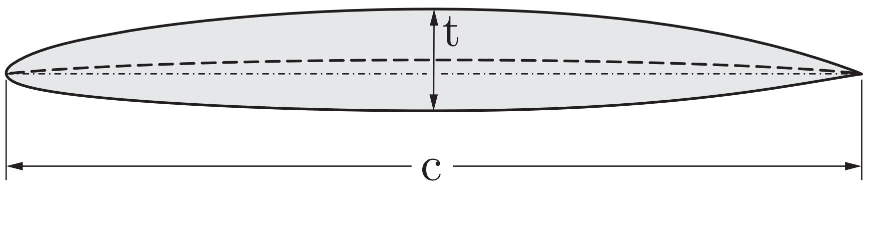
Naming Scheme

- 1st digit: design $10 \cdot 2/3 \cdot C_{\ell \text{ design}}$
- 2nd and 3rd digit: $2 \cdot 100 \cdot x_f / c$ plus 1. Assuming that the position of the maximum camber is a multiple of 5% the 3rd digit is always a 1.
- 4th and 5th digit: maximum thickness $100 \cdot t / c$

Example:
NACA 23112: like NACA 23012: design lift coefficient 0.3, maximum camber at 15% chord, 12% thickness, but with a reflexed aft camber line.

NACA 1-series airfoils

The development of these airfoils was aiming at high subsonic speed applications like propellers (see [5]). Their shape was designed with the help of the new (that is, in the 1930s) numerical design methods. JAVAFOIL can create airfoils of the NACA-16 type, which are the only members of the 1-series published by NACA. The maximum thickness and the maximum camber are located at 50% chord, whereas the minimum pressure is reached at 60% of the chord length.



Parameters:

- Free: t/c, $C_{\ell\ \mathrm{design}}$
- Fixed $x_f/c = 0.5$, $x_t/c = 0.5$

Naming Scheme

- 1st digit “1”: series designation
- 2nd digit: position of minimum pressure of the thickness distribution $10 \cdot x/c$
- a dash
- 3rd digit: $10 \cdot C_{\ell\ \mathrm{design}}$
- 4rd and 5th digit: maximum thickness $100 \cdot t/c$

  Example:
  16-212: 1-series, minimum pressure at 60% chord, design lift coefficient 0.2, 12% thickness.

While these airfoil shapes are not based on analytical expressions, the published coordinates have been approximated to produce an accurate representation of these airfoils. The camber lines used are of the uniform load type (a=1.0).

NACA 6- and 6A-series airfoils

These airfoils were the first NACA airfoils which had been systematically developed with the inverse design method by Theodorsen. The conformal mapping algorithm was able to deliver a shape for a given pressure distribution. This means that no closed form equations describing the thickness distributions exist.
Earlier JavaFoil versions used a very approximate algorithm which had been lifted from the "Digital Datcom" programs, but this produced rather inaccurate representations of the 6-series airfoils. Therefore, since version 2.09 (August 2009) JavaFoil uses a more elaborate algorithm, which is based on the work of Ladson [6]. This new method is using quite accurate tables of the stream function for most of the 6-series airfoils. JavaFoil can generate individual members of the 63, 64, 65, 66 and 67 as well as the 63A, 64A, and 65A families. The "A" modification leads to a less cusped trailing edge region.

The 63, 64, 65, 66 and 67 families can be combined with camber lines of the $a = 0$ to $a = 1$ type.

The 63A, 64A, and 65A sections use a modified $a = 0.8$ camber line which is straight aft of $x/c = 0.8$. The thickness distribution of these airfoils has also been modified to yield straight lines from $x/c = 0.8$ to the trailing edge.

The “a” camber line shapes are specified in terms of the design lift coefficient and the position x/c where the constant loading ends. This is indicated with an additional $a = \mathrm{x.y}$ label in the airfoil name.
If you specify $a \geq 1$ in JavaFoil’s input, the camber line has a constant loading from leading edge to trailing edge. The resulting airfoils do not carry the “a” label.

Note that officially no intermediate airfoils (e.g. a NACA64.5-012) exist.

Naming Scheme

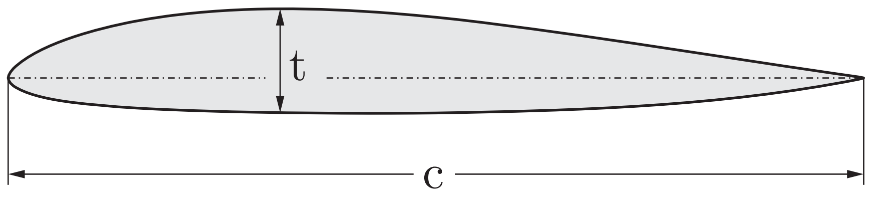
- 1st digit “6”: series designation

- 2nd digit: chordwise position of minimum pressure of the thickness distribution $10 \cdot x / c$
- single digit suffix following a comma, which is $10 \cdot \Delta C_\ell$. It represents the range ΔC_ℓ above and below $C_{\ell\,\mathrm{design}}$ where favorable (accelerating) pressure gradients for laminar flow exist (therefore ΔC_ℓ is approximately the semi-width of laminar bucket)
- a dash
- 3rd digit: design $10 \cdot C_{\ell\,\mathrm{design}}$
- 4rd and 5th digit: maximum thickness $100 \cdot t / c$

A camber line shape different from $a = 1.0$ is indicated by the additional designation $a = x.y$, where $x.y$ is replaced by the location x / c where the constant part of the loading ends and the linear drop towards the trailing edge starts..

TsAGI "B" airfoils

The TsAGI (also ZAGI, CAGI) was and is Russia's leading aeronautical research organization. Not much is known about early airfoil development, but the available literature [6], [9] shows that similar to other nations Russia has developed airfoil families based on analytical shape descriptions. The TsAGI series-B is just one such airfoil family. The very simple shape description is using just the maximum thickness. The resulting sections have a reflexed camber line and hence low pitching moment.



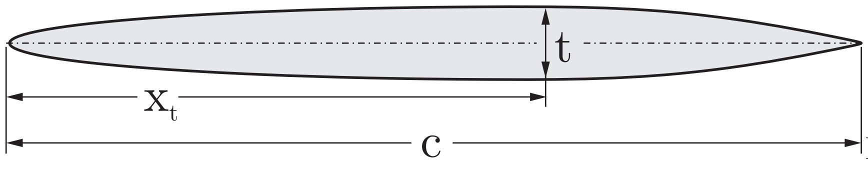
Parameters:

- Free: t / c
- Fixed $x_t / c = 0.3388$, maximum (positive) camber at $x_f / c = 0.3018$, minimum (negative) camber at $x_f / c = 0.9204$.
- The maximum camber is linked to the thickness by the expression $f / c = 0.168 \cdot t / c$.

!!! I am still looking for more information about Russian airfoil developments.

NPL-EC, ECH and EQH airfoils

These British symmetrical airfoil sections are composed of an elliptical forward portion (E) and a cubic (C) or quartic (Q) rear end. The tail closure is built from a hyperbolic curve (H series). The location of the maximum thickness can be varied between 30 and 70% of the chord length. A limited description is contained in [10], [13].



Parameters:

Parameters:

- Free: t/c, x_t/c

After some reverse engineering, I have used the following assumptions for these airfoils:

- the trailing edge thickness is 2% of the airfoil thickness,
- in case of the "C" and "Q" series the rear end is attached with C0, C1, C2 continuity (position, tangent, curvature) to the elliptic front part,
- in case of the "Q" series the second derivative at the trailing edge is set to -0.2, (this gave the best approximations for 1240 to 1260 airfoils),
- the "H" modification closes the thick trailing edge by a hyperbolic curve which is attached with C0, C1 continuity (position, tangent) to the thickness distribution at $x/c = 0.965$.

Camber lines are 3rd order polynomials which allow to place the location of the maximum camber approximately between 30 and 60% of the chord length.

Note: I am still looking for the "official" description of the airfoil geometry of the EQ and EQH aerofoils, especially how the quartic curve was defined and how the hyperbolic closure was attached to the quartic curve. It seems to be that the procedure to generate these shapes was not published.

Biconvex airfoils

These are symmetrical airfoils, formed by two arcs. They can be represented by the following formula:

$$y = a \cdot \left(x - x^b\right)$$

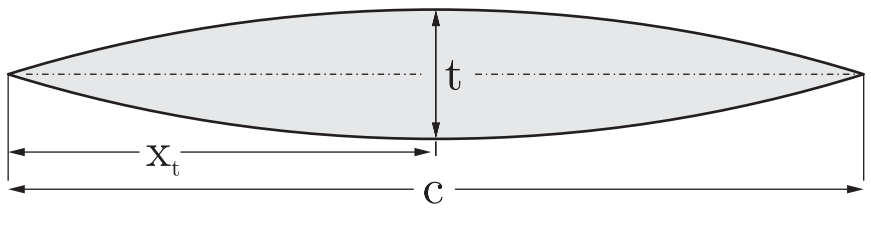
The exponent b can be found from the location of the maximum thickness, i.e. the point where $\partial y / \partial x = 0$

$$x_{t\,max} = \left(\frac{1}{b}\right)^{\frac{1}{b-1}},$$

while the factor a depends on the value of the maximum thickness:

$$t_{max} = 2 \cdot a \cdot \left(x_{t\,max} - x_{t\,max}^b\right)$$

If the maximum thickness is placed at $x/c = 0.5$, the airfoil is composed of two equal circular arcs. These airfoils are intended for supersonic flow.

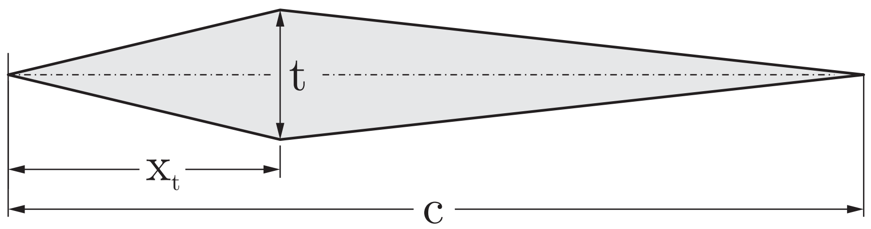


Parameters:

- Free: t/c, x_t/c

Double Wedge airfoils

These are symmetrical airfoils composed of straight lines. They are intended for supersonic flow.

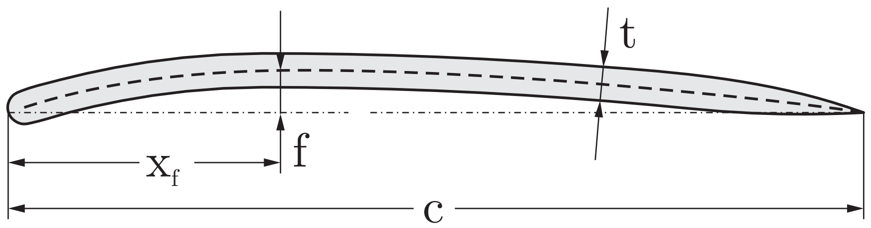


Parameters:

- Free: t/c, x_t/c

Plate airfoils

These sections are generated to represent flat plates with rounded noses and sharp trailing edges. The shape can be superimposed over a NACA 4-series camber line to produce a cambered plate.

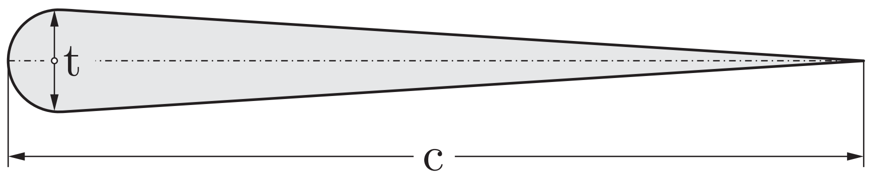


Parameters:

- Free: t/c, f/c, x_f/c
- Fixed: leading edge radius $r = 1/2 \cdot t/c$, trailing edge closure begins at $x/c = 0.8$

Newman airfoils

These sections consist of a circular nose to which straight tapered tail is attached. It can be manufactured easily, but has a curvature jump at the junction between the nose and the trailing wedge, leading to suction peaks and a risk of flow separation.



Parameters:

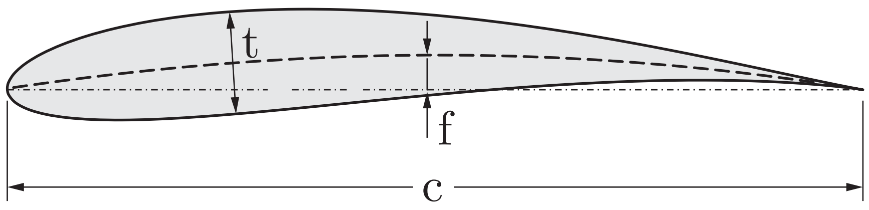
- Free: t/c

Joukowsky airfoils

These classical airfoil sections are generated by applying a conformal mapping procedure. They were the first practical airfoils developed on a theoretical model. Besides producing the airfoil shape, the mapping procedure was also used to find the flow field around the airfoil as well as the force and the moment acting on the wing section. The airfoils have very thin cusped trailing edges and are therefore difficult to analyze with panel methods and difficult to manufacture.
The conformal mapping is performed using the Joukowsky transformation of the complex points z_{circle} on a unit circle with is center at (x_0, y_0).

$$z_{\text{airfoil}} = z_{\text{circle}} + \frac{\lambda^2}{z_{\text{circle}}}, \text{ where } \lambda = -x_0 + \sqrt{1 - y_0^2} \,.$$

In order to match the prescribed airfoil thickness and camber, JAVAFOIL performs an iterative search for the center of the circle. As usual, the resulting coordinates are scaled to unit length.



Parameters:

- Free: t/c, f/c

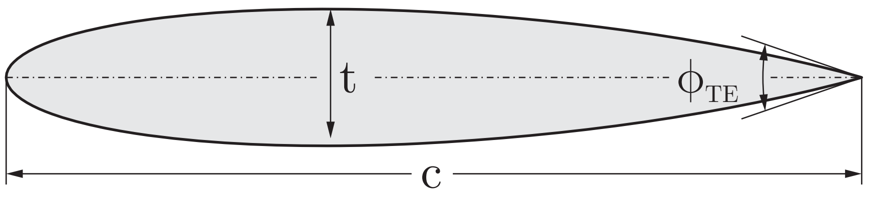
Van de Vooren airfoils

In contrast to the classical Joukowsky airfoils, these airfoils have a finite trailing edge angle. The transformation function is of the type

$$z_{\text{airfoil}} = \frac{\left(1 - z_{\text{circle}}\right)^k}{\left(\varepsilon - z_{\text{circle}}\right)^{k-1}} \,.$$

They can be used to create sections with thick trailing edge regions e.g. for fairings. A description of this shape can be found in [13].

Note that not all thicknesses can be achieved for all trailing edge angles; therefore the final maximum thickness may not be what was desired. Also only symmetrical sections are generated in JAVAFOIL.

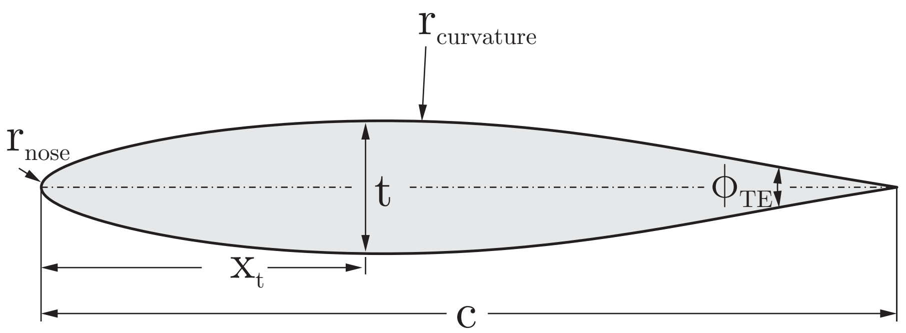


Parameters:

- Free: t/c, ϕ_{TE}

Helmbold-Keune airfoils

In the 1940s many attempts were made to extend the then classical NACA airfoil section methodology to more general airfoil shapes. Helmbold and Keune [15] developed elaborate methods to characterize and parameterize airfoil sections. While the mathematical approach allowed for representation of a wide range of shapes, the methodology was not really successful in these years of manual calculation. Later in the age of numerical shape optimization similar methods have been developed, e.g. the Parsec shape functions.
The parameters of the symmetrical airfoil must be carefully chosen to generate a realistic airfoil shape. The center curvature must be large enough to avoid self-crossing of the outline.

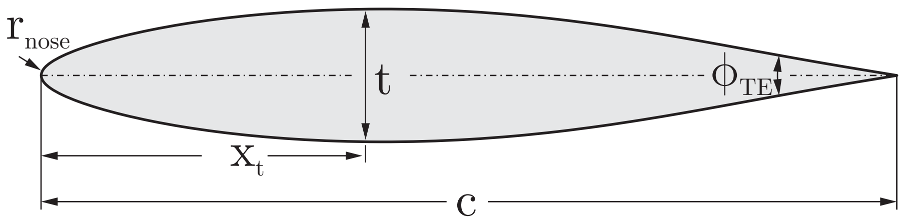


Parameters:

Free: t/c, x_t/c, trailing edge angle, curvature radius at middle, nose radius.

Roßner airfoils

Another algorithm to generate analytical airfoil shapes based on conformal mapping was published by Roßner [16]. Like all methods using conformal mapping, his solution also allowed for the exact analytical determination of the corresponding pressure distributions.



Parameters:

Free: t/c, x_t/c, trailing edge angle, nose radius.

Parsec airfoils

The Parsec geometry parameterization was developed by H. Sobietzky in the 1990s. It tries to model airfoil shapes by superposition of selected polynomial terms. The parameters resemble the Helmbold-Keune approach and are mainly intended to be used for numerical shape optimization.
The parameters of the symmetrical airfoil must be carefully chosen to generate a realistic airfoil shape. The center radius, the nose radius as well as the trailing edge wedge angle must be carefully adjusted to avoid self-crossing of the outline.

Due to the limited number of text entry fields in JAVAFOIL user interface the parameters of a Parsec-11 formulation have reduced so that they model symmetrical sections only.

Parameters:

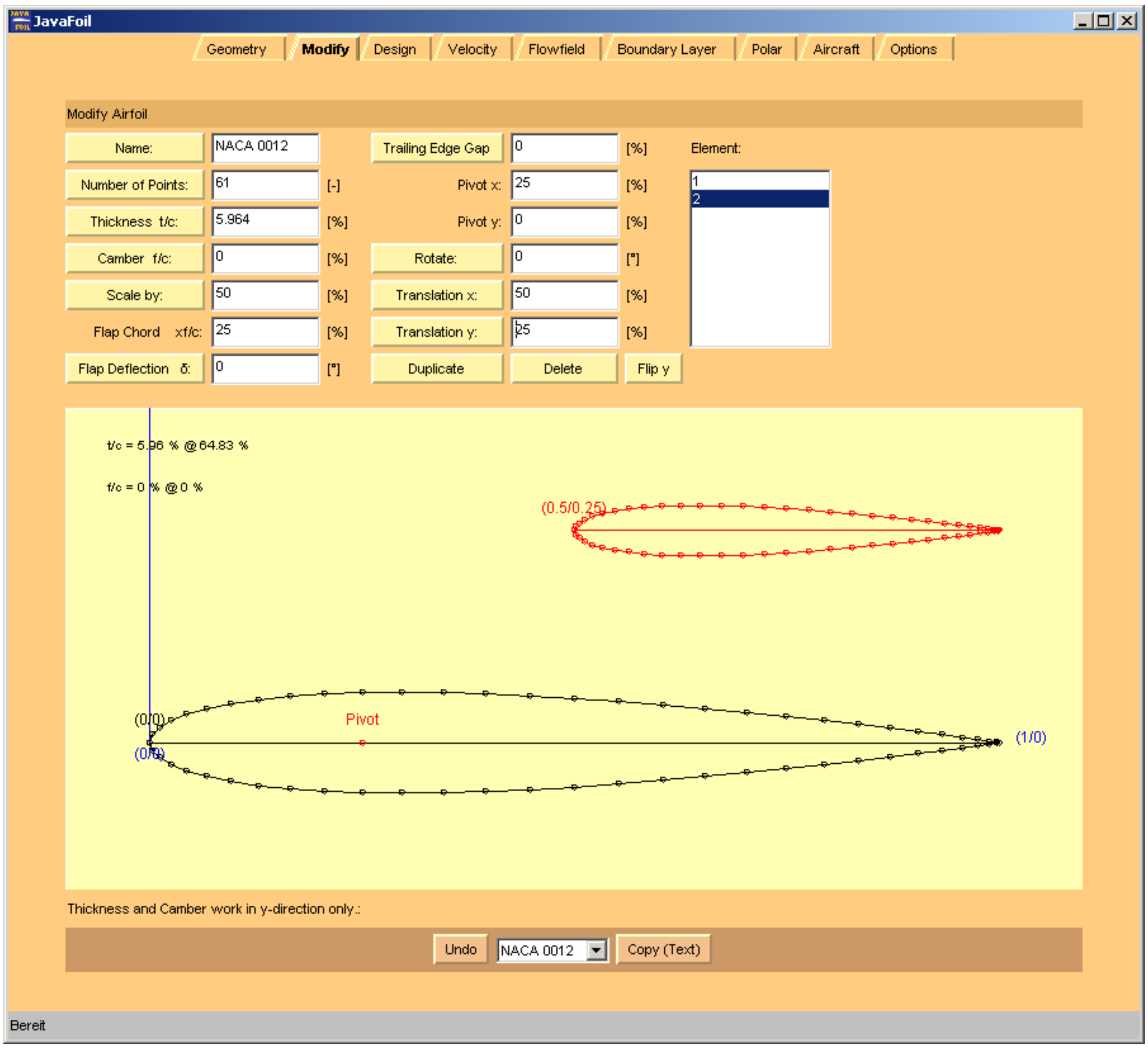
Free: t/c, x_t/c, trailing edge angle, curvature at x_t/c, nose radius.

Modify Card

This card can be used to perform various modifications to the airfoil geometry. It consists of an input and action area and a geometry view below.

The modification of parameters is performed by entering new values into a text field and then pressing the button at the left of the text field. Thus it is easy to apply certain operations several times. A modification will be applied to the airfoil elements which are currently selected in the “Element” list box.

The geometry view is automatically scaled to fit all airfoil elements. The currently selected elements are highlighted in red.



Name:	NACA 0012		Trailing Edge Gap	0	[%]	Element:
Number of Points:	61	[-]	Pivot x:	25	[%]	1
Thickness t/c:	5.964	[%]	Pivot y:	0	[%]	2
Camber f/c:	0	[%]	Rotate:	0	[°]	
Scale by:	50	[%]	Translation x:	50	[%]	
Flap Chord xf/c:	25	[%]	Translation y:	25	[%]	
Flap Deflection δ:	0	[°]				

Thickness and Camber work in y-direction only.:

NACA 0012

View of the Modify card showing a two-element airfoil with element #2 selected.

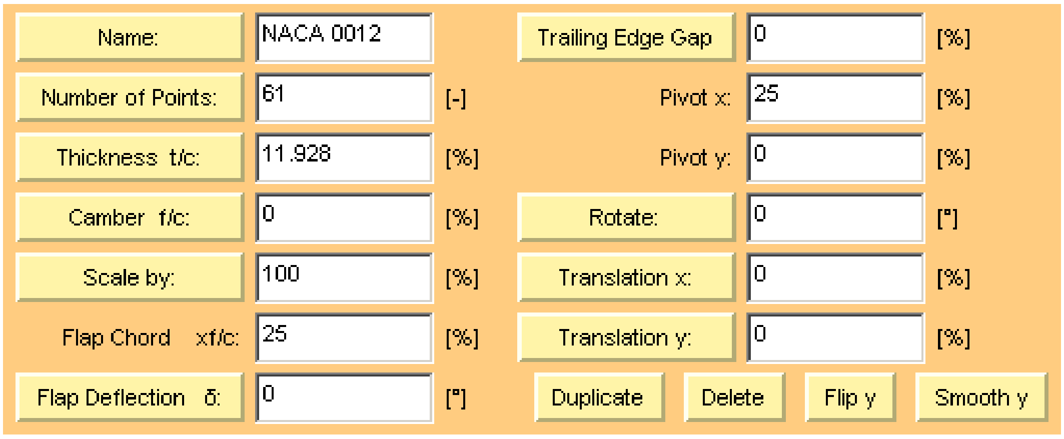
The following modifications can be performed:

- NAME
  Changes the name of the airfoil
- NUMBER OF POINTS
  Changes the number of coordinate points of the selected element(s).
- THICKNESS
  Scales the thickness of the selected element(s) by decomposing the shape into a thickness distribution and a camber line. Only the thickness distribution is scaled, so that the camber line is maintained. Note that small changes to the camber may occur due to numerical errors.

- CAMBER
  Scales the camber line to a new height. This works only if the airfoil is already cambered. Scaling the camber line of a symmetrical airfoil accomplishes nothing.
- SCALE BY
  Scales the airfoil shape by multiplying the coordinates with the given scaling factor.
- FLAP DEFLECTION
  Modifies the coordinates by deflecting a plain flap of the given chord length. The axis of rotation is always the middle between upper and lower surface.
- TRAILING EDGE GAP
  Modifies the shape so that the prescribed trailing edge gap is produced. Generally it is recommended to use closed trailing edges for analysis, except if the airfoil is extremely thin towards the trailing edge. This function can also be applied before exporting airfoil shapes suitable for manufacturing.
- ROTATE
  Rotates the selected airfoil element(s) around the specified Pivot point.
- TRANSLATE X
  Moves the selected airfoil element(s) by the given distance horizontally.
- TRANSLATE Y
  Moves the selected airfoil element(s) by the given distance vertically.
- DUPLICATE
  Creates a copy of the currently selected element(s). Note that you have to move the new element from its initial location so that it is not overlapping with other elements.
- DELETE
  Deletes the selected element(s)
- FLIP Y
  Reflects the selected elements across a horizontal line passing through the pivot point.
- SMOOTH Y
  This command currently supports two smoothing variants:
  If the smoothing factor is <u>positive</u>, the coordinates are approximated by a smoothing spline curve. Only this first variant is available via the graphical user interface. There, it uses a hard-wired smoothing factor of 0.1. Other factors can be used when the scripting interface is used.
  If the smoothing factor is <u>negative</u>, a filter is applied to the y-coordinates to reduce waviness. This filter applies a weighted average to each point and its two neighbor points. If for example the smoothing factor is 0.1, the y coordinate of the smoothed point is 90% of its initial value and 10% of the linear interpolation between the two neighboring points according to:

  $$y_i \leftarrow (1-f)\cdot y_i + f\cdot\left(y_{i-1} + \left(y_{i+1} - y_{i-1}\right)\cdot\frac{s_i - s_{i-1}}{s_{i+1} - s_{i-1}}\right)$$

  This filter can be applied several times, but subsequent application will also smooth out details like a pointed airfoil nose. This option is only available via the scripting language.



Name:	NACA 0012		Trailing Edge Gap	0	[%]
Number of Points:	61	[-]	Pivot x:	25	[%]
Thickness t/c:	11.928	[%]	Pivot y:	0	[%]
Camber f/c:	0	[%]	Rotate:	0	[°]
Scale by:	100	[%]	Translation x:	0	[%]
Flap Chord xf/c:	25	[%]	Translation y:	0	[%]
Flap Deflection δ:	0	[°]			

View of the controls on the Modify card.

You can also modify individual points by dragging them up or down with the left mouse button depressed. This modification method is restricted to movements in the y-direction. If you need more freedom, you have to modify the numerical coordinate values on the Geometry card.

Note concerning multi-element airfoils
Modifications are applied only to the airfoil element(s) selected in the “Elements” list box. The selection is also used by other cards. Only selected elements are taken into account when total force, moment and drag coefficients are determined.

The Panel Method

JAVAFOIL implements a classical panel method to determine the linear potential flow field around single and multi-element airfoils. In JAVAFOIL the airfoil surfaces carry a linearly varying vorticity distribution. This is the same type of distribution as used in *XFOIL* but simpler than the higher order (parabolic) distribution as used in Eppler’s *PROFIL* code. The resulting equation system consists therefore of a (# of panels +1)^2 sized matrix and two right hand sides. These are for 0° and 90° angle of attack and can be solved efficiently at the same time for the two corresponding vorticity distributions. The vorticity distribution for any arbitrary angle of attack is then derived from these two solutions (remember that potential theory is linear and allows for superposition). There is no interaction with the boundary layer, as in *XFOIL*, though.
For a shape discretization by N panels, the equation system of this classical panel method consists of the matrix of influence coefficients, the unknown circulation strength at each panel corner point and the two right hand side vectors. These represent the “no flow through the surface” conditions for 0° and 90° angle of attack. Each coefficient $C_{i,j}$ reflects the effect the influence of the triangular vorticity distribution due to the vortex strength γ_i at each corner point on the center point of each panel j. The last row contains the tangential flow condition at the trailing edge (AKA “Kutta-condition”).

$$\begin{bmatrix} C_{1,1} & \cdots & C_{N+1,1} \\ \vdots & \ddots & \vdots \\ C_{1,N} & \cdots & C_{N+1,N} \\ 1 & \cdots 0 \cdots & 1 \end{bmatrix} \cdot \begin{bmatrix} \gamma_{1,0°} & \gamma_{1,90°} \\ \gamma_{2,0°} & \gamma_{2,90°} \\ \vdots & \vdots \\ \gamma_{N+1,0°} & \gamma_{N+1,90°} \end{bmatrix} = \begin{bmatrix} RHS_{1,0°} & RHS_{1,90°} \\ RHS_{2,0°} & RHS_{2,90°} \\ \vdots & \vdots \\ RHS_{N+1,0°} & RHS_{N+1,90°} \end{bmatrix}$$

Boundary Layer Analysis

The boundary layer analysis module implements an integral boundary layer integration scheme following publications by Prof. Richard Eppler. Such integral methods are based on differential equations describing the growth of boundary layer parameters depending on the external local flow velocity. These equations are then integrated starting at the stagnation point. While accurate analytical formulations are available for laminar boundary layers, some empirical correlations are needed to model the turbulent part.

Note: the local skin friction coefficient as given on the Boundary Layer card is twice the value as used by Eppler to follow the more common convention $C_f = \tau_0 / \left(\frac{\rho}{2} \cdot v_\infty^2\right)$.

In JAVAFOIL there is no interaction between the boundary layer and the external flow, as in *XFOIL*, though. Therefore largely separated flows cannot be analyzed – a short flow separation ($s_{separated} / c < 10\%$) at the trailing edge does not affect the results very much. Also laminar separation bubbles are not modeled; when laminar separation is detected the code switches to turbulent flow.

Transition Criteria

Methods to predict transition from laminar to turbulent flow have been developed by many authors since the early days of Prandtl's boundary layer theory. While it is possible to analyze the stability of a boundary layer numerically, all methods which are practical and fast are more or less approximate and rely on empirical relations (usually derived from experiments). Because the local boundary layer parameters at a station s are the result of an integration process starting at the stagnation point, they contain a "history" of the flow.

Local Criteria

Many methods predict transition by applying a criterion based on local boundary layer parameters. These criteria are based on relations, which can be evaluated at any station along the surface. They do not need an extra integration of some instability parameter, but of course are affected by the "history" of the flow. Most of these criteria are relating Re_{δ_2} to the shape of the boundary layer profile.

Eppler

Transition is assumed to occur when $Re_{\delta_2} \geq e^{18.4 \cdot H_{32} - 21.74 - 0.36 \cdot r}$.

Eppler enhanced

Transition is assumed to occur when $Re_{\delta_2} \geq e^{18.4 \cdot H_{32} - 21.74 + 125 \cdot (H_{32} - 1.573)^2 - 0.36 \cdot r}$.

Michel (1)

This simple criterion assumes transition to occur when $Re_{\delta_2} \geq 1.535 \cdot Re_s^{0.444}$.

Michel (2)

Transition is assumed to occur when $\mathrm{Re}_{\delta_2} \geq 1.174 \cdot \left(1 + 22400 / \mathrm{Re}_s\right) \cdot \mathrm{Re}_s^{0.46}$. See [22].

H_{12}-Re_s

Transition is assumed to occur when $2.1 < \mathrm{H}_{12} < 2.8$ and
$\log_{10}\left(\mathrm{Re}_s\right) \geq -40.4557 + 64.8066 \cdot \mathrm{H}_{12} - 26.7538 \cdot \mathrm{H}_{12}^{\,2} + 3.3819 \cdot \mathrm{H}_{12}^{\,3}$. See [23].

Criteria based on a region of instability, $\mathrm{n} - \mathrm{Re}_{\delta_2}$ envelopes

These methods first determine a local point of instability and then begin at this point to integrate a measure for the amplification of instability.

Drela approximates the envelopes of the amplification rate n versus Re_{δ_2} by straight lines of the form $\tilde{\mathrm{n}} = \mathrm{f}\left(\mathrm{Re}_{\delta_2}, \mathrm{H}_{12}\right)$. Two versions of this approximation were used in his codes of the *XFOIL* and *MSES/ISES* family.

The approximation is expressed by

$$\tilde{\mathrm{n}} = \frac{\partial \tilde{\mathrm{n}}}{\partial \mathrm{Re}_{\delta_2}} \cdot \left(\mathrm{Re}_{\delta_2} - \mathrm{Re}_{\delta_2,\mathrm{crit}}\right).$$

Transition can occur when $\mathrm{Re}_{\delta_2} > \mathrm{Re}_{\delta_2,\mathrm{crit}}$ and $\tilde{\mathrm{n}} > \mathrm{n}_{\mathrm{crit}}$. In JAVAFOIL transition is assumed to occur when the value $\mathrm{n}_{\mathrm{crit}} = 9 - \mathrm{r}$ is exceeded.

Drela, XFOIL 1.1 and 5.4

$$\frac{\partial \tilde{\mathrm{n}}}{\partial \mathrm{Re}_{\delta_2}} = 0.01 \cdot \sqrt{\left(2.4 \cdot \mathrm{H}_{12} + 2.5 \cdot \tanh\left(1.5 \cdot \mathrm{H}_{12} - 4.65\right) - 3.7\right)^2 + 0.25}$$

$$\log_{10}\left(\mathrm{Re}_{\delta_2,\mathrm{crit}}\right) = \left(\frac{1.415}{\mathrm{H}_{12} - 1} - 0.489\right) \cdot \tanh\left(\frac{20}{\mathrm{H}_{12} - 1} - 12.9\right) + \frac{3.295}{\mathrm{H}_{12} - 1} + 0.44$$

These approximations can be found in [1] and [2].

Drela, XFOIL 5.7

Modification in 1991

$$\frac{\partial \tilde{\mathrm{n}}}{\partial \mathrm{Re}_{\delta_2}} = 0.028 \cdot \left(\mathrm{H}_{12} - 1\right) - \frac{0.0345}{e^{-\left(\frac{3.87}{\mathrm{H}_{12} - 1} - 2.52\right)^2}}$$

$$\log_{10}\left(\mathrm{Re}_{\delta_2,\mathrm{crit}}\right) = 0.7 \cdot \tanh\left(\frac{14}{\mathrm{H}_{12} - 1} - 9.24\right) + 2.492 \cdot \left(\frac{1}{\mathrm{H}_{12} - 1}\right)^{0.43} + 0.66$$

Drela, XFOIL 6.8

only a tiny modification (term 0.66 → 0.62)

$$\frac{\partial\tilde{n}}{\partial \mathrm{Re}_{\delta_2}} = 0.028 \cdot \left(H_{12} - 1\right) - \frac{0.0345}{e^{-\left(\frac{3.87}{H_{12}-1}-2.52\right)^2}}$$

$$\log_{10}\left(\mathrm{Re}_{\delta_2,crit}\right) = 0.7 \cdot \tanh\left(\frac{14}{H_{12} - 1} - 9.24\right) + 2.492 \cdot \left(\frac{1}{H_{12} - 1}\right)^{0.43} + 0.62$$

Method of Arnal:

A set of tables produced by D. Arnal has been approximated by W. Würz with polynomials:

$$\frac{\partial\tilde{n}}{\partial \mathrm{Re}_{\delta_2}} = a_1 + a_2 \cdot H_{12} + a_3 \cdot H_{12}^2$$

$$\log_{10}\left(\mathrm{Re}_{\delta_2,crit}\right) = b_1 + b_2 \cdot H_{12} + b_3 \cdot H_{12}^2$$

Here the envelope is not a straight line as in Drela's method. For details see [19].
In JAVAFOIL transition is assumed to occur when the value $n_{crit} = 9 - r$ is exceeded.

Method of Granville

This method is not described here. It also works by integrating a stability parameter starting from a point of instability.

Abbreviations:

approximation of *n*	$\tilde{n}$
roughness factor (0 = smooth)	r
displacement thickness	δ_1
momentum thickness	$\delta_2 = \theta$
shape factor displacement thickness / momentum thickness	$H_{12} = \frac{\delta_1}{\delta_2}$
Reynolds number based on local momentum thickness	$\mathrm{Re}_{\delta_2} = \mathrm{Re}_\theta$
Reynolds number based on local arc length	Re_s

Effect of Roughness

The effect of roughness on transition and drag is complex and cannot be simulated accurately. Even modern resource hungry direct numerical simulation methods have difficulties to simulate the effect.
In JAVAFOIL two effects of surface roughness are modeled:

- laminar flow on a rough surface will be destabilized leading to premature transition,
- laminar as well as turbulent flow on rough surfaces produce a higher skin friction drag.

The effect on toughness is modeled in the following transition models

Eppler Standard	Transition is assumed to occur when $\mathrm{Re}_{\delta_2} \geq e^{18.4 \cdot H_{32} - 21.74 - 0.36 \cdot r}$.
Eppler enhanced	Transition is assumed to occur when $\mathrm{Re}_{\delta_2} \geq e^{18.4 \cdot H_{32} - 21.74 + 125 \cdot (H_{32} - 1.573)^2 - 0.36 \cdot r}$.
Drela,	Transition is assumed to occur when the value $n_{crit} = 9 - r$ is exceeded.

e^n approx.	
Arnal (Würz)	Transition is assumed to occur when the value $n_{crit} = 9 - r$ is exceeded.

The global effect on drag is taken into account by a simple scaling of the total drag coefficient

$$C_d = C_d \cdot (1 + r/10)$$

The roughness factor r is meant to represent the following surface conditions

$r = 0$	perfect smooth surface as for example on a composite material sailplane wing
$r = 1$	smooth, but slightly rough surface as for example a painted cloth surface
$r = 2$	similar to the NACA standard roughness
$r = 3$	dirty surface with spots of dirt, bugs and flies

Note that the NACA standard roughness is usually applied to the leading edge only. It consists of a sparse (5-10% of the area) leading edge coating up to 8% x/c. The grain size is about 0.45‰ of the chord length. Thus for a wing chord length of 1m the grain size would be 0.45mm.

Stall Corrections

Empirical Stall Correction #1 („CalcFoil“)

```
if ($\alpha > 0$)
{
    // handle separation on upper surface
    // drag increment
    $C_{d,\,upper} = C_{d,\,upper} + \left|\sin^2\alpha \cdot \left(x_{TE} - x_{sep,\,upper}\right)^2 + 0.025 \cdot \cos\alpha \cdot \left(x_{TE} - x_{sep,\,upper}\right)^2\right|$
    // lift multiplier reduces lift linearly with length of separated length
    $C_\ell = C_\ell \cdot \left(1 - 0.2 \cdot \left(x_{TE} - x_{sep,\,upper}\right)\right)$
}
else if ($\alpha < 0$)
{
    // handle separation on lower surface
    // drag increment
    $C_{d,\,lower} = C_{d,\,lower} + \left|\sin^2\alpha \cdot \left(x_{TE} - x_{sep,\,lower}\right)^2 + 0.025 \cdot \cos\alpha \cdot \left(x_{TE} - x_{sep,\,lower}\right)^2\right|$
    // lift multiplier reduces lift linearly with length of separated length
    $C_\ell = C_\ell \cdot \left(1 - 0.2 \cdot \left(x_{TE} - x_{sep,\,lower}\right)\right)$
}

// moment multiplier
$C_{m,\,corrected} = C_{m,\,panel\ method} \cdot 0.9 \cdot x^2_{sep,\,lower} \cdot x^2_{sep,\,upper}$

// lift multiplier due to suction peak criterion
```

$C_\ell = C_\ell \cdot \frac{1}{\left(\frac{\Delta C_{P,\,max}}{20}\right)^2 + 1}$, where $\Delta C_{P,\,max}$ is the difference between the minimum pressure coefficient close to the nose of the airfoil and the pressure close to the trailing edge.

Empirical Stall Correction #2 („Eppler“)

```
if ($\alpha > 0$)
{
        // handle separation on upper surface
        if ( $x_{sep,\,upper} < x_{TE}$ )
        {
                // trailing edge angle of upper surface
                $\theta_{TE} = \arctan\left(-\frac{y_{sep,\,upper} - y_{TE}}{x_{sep,\,upper} - x_{TE}}\right)$
        }
        else
        {
                $\theta_{TE} = 0$
        }
        // drag increment
        $C_{d,\,upper} = C_{d,\,upper} + 0.2 \cdot \sin(\alpha + \theta_{TE}) \cdot \left(x_{TE} - x_{sep,\,upper}\right)^2$

        $\Delta C_\ell = C_{l,\,max,\,fudge} \cdot (\alpha + \theta_{TE}) \cdot \pi \cdot \left(x_{TE} - x_{sep,\,upper}\right)$

        if ( $\Delta C_\ell > 0$ )
        {
                // lift reduction
                $C_\ell = C_\ell - \Delta C_\ell$
        }
        else
        {
                // lift multiplier
                $C_\ell = C_\ell \cdot \left(1 - \sin\alpha \cdot \left(x_{TE} - x_{sep,\,upper}\right)\right)$
        }

        // moment increment
        $C_m = C_m - \sin\alpha \cdot \left(x_{TE} - x_{sep,\,upper}\right) \cdot \left(0.5 \cdot \left(1 + x_{sep,\,upper}\right) - 0.25\right)$
}
else if ($\alpha < 0$ )
{
        // handle separation on lower surface
        if ( $x_{sep,\,lower} < x_{TE}$ )
        {
```

```
        // trailing edge angle of lower surface
        $\theta_{TE} = \arctan\left(-\frac{y_{sep,\ lower} - y_{TE}}{x_{sep,\ lower} - x_{TE}}\right)$
    }
    else
    {
        $\theta_{TE} = 0$
    }
    // drag increment
    $C_{d,\ lower} = C_{d,\ lower} - 0.2 \cdot \sin(\alpha + \theta_{TE}) \cdot \left(x_{TE} - x_{sep,\ lower}\right)^2$

    $\Delta C_\ell = C_{l,\ max,\ fudge} \cdot (\alpha + \theta_{TE}) \cdot \pi \cdot \left(x_{TE} - x_{sep,\ lower}\right)$

    if ( $\Delta C_\ell < 0$ )
    {
        // lift reduction
        $C_\ell = C_\ell - \Delta C_\ell$
    }
    else
    {
        // lift multiplier
        $C_\ell = C_\ell \cdot \left(1 - \sin\alpha \cdot \left(x_{TE} - x_{sep,\ lower}\right)\right)$
    }

    // moment increment
    $C_m = C_m - \sin\alpha \cdot \left(x_{TE} - x_{sep,\ lower}\right) \cdot \left(0.5 \cdot \left(1 + x_{sep,\ lower}\right) - 0.25\right)$
}
```

// lift multiplier due to modified suction peak criterion
$C_\ell = C_\ell \cdot \frac{1}{\left(\frac{\Delta C_{P,\ max}}{30}\right)^2 + 1}$, where $\Delta C_{P,\ max}$ is the difference between the minimum pressure coefficient close to the nose of the airfoil and the pressure close to the trailing edge.

Compressible Flow

JAVAFOIL analyzes airfoils in incompressible flow, which means low Mach numbers as they are common in model aircraft of general aviation airplanes. In practical application this means Mach numbers below M = 0.25. It is possible however to extend the Mach number range somewhat by applying compressibility corrections to the incompressible results. This is only possible, as long as the flow speed is subsonic all over the surface of the airfoil and compressibility effects are small.

Critical Pressure Coefficient

The character of the flow changes dramatically when sonic speed is exceeded anywhere on the surface. The pressure coefficient associated with sonic speed is called "critical pressure" coefficient ($C_{p,crit}$). In most cases pressure recovery from supersonic to subsonic speeds (from $C_p < C_{p,crit}$ to $C_p > C_{p,crit}$) is leading to an abrupt recompression with a shock. The analysis of such flows requires more complex methods than implemented in JAVAFOIL. Such methods must be capable of handling compressible flows (for example by solving the full, compressible potential equations or by solving the Euler equations).

In order to indicate how close the local flow is to supersonic speeds, JAVAFOIL calculates the critical pressure coefficient if a Mach number is specified on the Options card. The critical limit is drawn as a wavy line in the graph on the Velocity card. Additionally, a compressibility correction is applied to the incompressible solution to model first order compressibility effects. Note however, that the theory becomes invalid, when flow reaches or exceeds sonic speed.

In JAVAFOIL, the critical pressure coefficient is calculated from the relation

$$C_{p,\,crit} = \frac{2}{\kappa \cdot M_\infty^2} \cdot \left(\left(\frac{2}{\kappa+1} \cdot \left(1 + \frac{\kappa-1}{2} \cdot M_\infty^2\right)\right)^{\frac{\kappa}{\kappa-1}} - 1 \right)$$

(Küchemann, „The Aerodynamic Design of Aircraft", p.115).

In terms of the velocity ratio the critical limit is found from

$$\left(\frac{v}{v_\infty}\right)_{crit} = \sqrt{1 + 2 \cdot \frac{1+M_\infty^2}{(\kappa+1) \cdot M_\infty^2}}$$

(Küchemann, „The Aerodynamic Design of Aircraft", p.114).

Compressibility Corrections

There are different ways to correct incompressible flow results for compressibility effects. One should keep in mind that these are only corrections – they can never produce the correct physical effects. Therefore the applicability of all compressibility corrections is limited to cases where the local flow velocity (which can be much higher than the onset flow velocity) is well beyond the speed of sound. In practical application one can use such corrections well up to about Mach = 0.5, the error grows very rapidly when Mach exceeds 0.7.

In JAVAFOIL, the panel analysis is always running on the given airfoil – the shape is never geometrically distorted. Compressibility corrections are applied later to the local surface velocities according to the Kàrmàn-Tsien approximation (usually written for C_p)

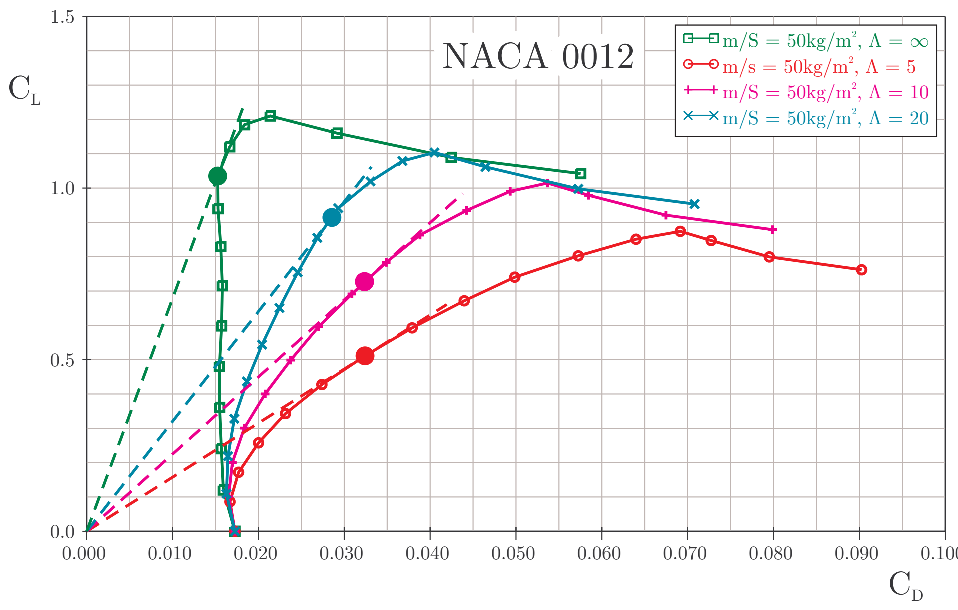
$$\left(\frac{v}{v_\infty}\right)_{comp.} = \left(\frac{v}{v_\infty}\right)_{inc.} \cdot \frac{1 - \frac{M_\infty^2}{2 - M_\infty^2}}{1 - \frac{M_\infty^2}{2 - M_\infty^2} \cdot \left(\frac{v}{v_\infty}\right)_{inc.}^2} \, .$$

(Cebeci, "An Engineering Approach to the Calculation of Aerodynamic Flows", p. 32).

Finite Wings in *JAVAFOIL*

In the 1920s it has been found by Prandtl and also by Lanchester that the finite span of wings affects their aerodynamic performance. They found that the effects could be expressed as a function of the aspect ratio (a.k.a. “slenderness” or “finesse”) of the wing. Prandtl’s “Lifting Line” theory was developed and successfully applied to design wings up to the 1940s and even today it is useful for unswept wings of relatively high aspect ratio ($\Lambda > 5$). The aspect ratio can be determined from $\Lambda = b / \ell = b^2 / S$ (span b divided by the mean chord length ℓ or span squared divided by wing area S). The main result of this theory is that the airfoil drag is increased by an additional drag force (“induced drag” a.k.a. “vortex drag”) which is caused by the finite wing span. The vortex drag coefficient of a wing can be expressed by $C_{D,\text{ induced}} = k \cdot C_L^2 / (\pi \cdot \Lambda)$, where C_L is the lift coefficient of the whole wing and k is a factor to account for the shape of the lift force distribution along the span (for good wing designs $k \approx 1$).

Now, JAVAFOIL is a program for the analysis of two dimensional airfoils. Nevertheless it supports a very simple model of finite wings to allow for a more realistic comparison of airfoils. When the user supplies a value for the aspect ratio on the Options card classical wing theory formulas are used to determine an approximation of the 3D effects.
These effects can applied to the polars produced by JAVAFOIL and make it possible to get a first impression of the relations between induced drag and airfoil drag. For example the importance of the airfoil drag is diminishing for higher lift coefficients and lower aspect ratios.
The three dimensional corrections can be applied to the results for constant Reynolds number (Polar card) as well as more realistically for the results associated with a constant wing loading (Aircraft card).



Lift versus drag coefficient polars for a NACA 0012 airfoil and wings of different aspect ratio.

The graph above shows the effect of the wing aspect ratio on lift over drag coefficient. Starting with infinite aspect ratio (aspect ratio = 0 on the Options card) three wings with increasing aspect ratio have been analyzed. For each curve the maximum of the lift over drag (L/D) ratio is indicated by a filled circle. It can be clearly seen, that depending on the aspect ratio the additional induced drag distorts the polar so that the optimum L/D ratio is shifted to lower lift coefficients. While the two dimensional airfoil achieves its maximum of L/D at slightly above $C_\ell = 1.0$, the low aspect ratio wing of $\Lambda = 5$ requires to operate the airfoil at $C_\ell = 0.5$ because this is the optimum C_L of the whole wing. If we compare with another airfoil we would better compare the airfoils at the lift coefficients corresponding to the wing aspect ratios.

Note that the results as shown above are accurate for a wing having an elliptical lift distribution and an elliptical, untwisted planform. Due to the spanwise lift distribution on a generic wing, the airfoils along the span of the wing will operate somewhat above and below the total lift coefficient of the wing. To study this effect requires using a more sophisticated three dimensional wing analysis code (e.g. for subsonic flow lifting line, vortex lattice or panel methods). Also no additional wing effects (like Reynolds number variation due to taper) are taken into account.

Polars for Constant Wing Loading

Airfoil data has traditionally been presented in form of graphs and tables for constant Reynolds numbers. This form results from the typical way wind tunnel experiments and numerical analyses are conducted. In a wind tunnel it is relatively easy to maintain a constant wind speed and Reynolds number.
Now the lift coefficient of a real airplane depends on the speed because the wing loading is usually fixed during flight – flying at low lift coefficients results in high speeds (and high Reynolds numbers) and vice versa. Therefore the operating points during flight would slice through a set of polars having constant Reynolds numbers.
It is possible to create polars more closely related to the conditions during flight. This would require adjusting the wind speed to each lift coefficient, which is cumbersome and expensive in a wind tunnel, but feasible in a numerical tool like JAVAFOIL.

Abbreviations:

mass of aircraft	m	kg
gravity constant	g	m/s^2
density of medium	ρ_∞	m/s^2
kinematic viscosity	ν	m^2/s
flight speed	v_∞	m/s
wing area	S	m^2
chord length	ℓ	m
Reynolds number	Re	-

Basic Equations

The definition of the lift coefficient is $C_L = \frac{m \cdot g}{\frac{\rho_\infty}{2} \cdot v_\infty^2 \cdot S}$. Solving the definition of the Reynolds number $Re = \frac{v_\infty \cdot \ell}{\nu}$ for the velocity v_∞ yields $v_\infty = \frac{Re \cdot \nu}{\ell}$. Inserting this result into the definition of the lift coefficient produces

$$C_L = \frac{m \cdot g \cdot \ell^2}{\frac{\rho_\infty}{2} \cdot Re^2 \cdot \nu^2 \cdot S} .$$

Solving for the Reynolds number yields

$$\mathrm{Re} = \frac{\ell}{\nu} \cdot \sqrt{\frac{2 \cdot g}{\rho_\infty \cdot C_L} \cdot \frac{m}{S}} \ .$$

Note that this equation can also be written $\mathrm{Re} \cdot \sqrt{C_L} = \frac{\ell}{\nu} \cdot \sqrt{\frac{2 \cdot g}{\rho_\infty} \cdot \frac{m}{S}}$, which means that we can also calculate polars of constant $\mathrm{Re} \cdot \sqrt{C_L}$ to match a given aircraft.

Using these results one can derive an aircraft oriented airfoil polar for a given wing loading $\frac{m}{S}$ and given mean chord length ℓ. Due to the dependency between lift coefficient and Reynolds number an iterative calculation procedure is used:

- prescribe the environmental condition density ρ_∞ and kinematic viscosity ν.
- prescribe a wing loading $\frac{m}{S}$ and a reference chord length ℓ.
- perform the following calculation sequence:

```
inital value
$\mathrm{Re}^* = 10^6$
for ($\alpha = \alpha_0$ to $\alpha_1$ step $\Delta\alpha$)
{
    iterate
    {
        $\mathrm{Re} = \mathrm{Re}^*$
        $C_L = f(\alpha, \mathrm{Re})$
        $\mathrm{Re}^* = \sqrt{\frac{g}{\frac{\rho_\infty}{2} \cdot C_L \cdot \nu^2} \cdot \frac{m}{S} \cdot \ell^2}$
    }
    while($\frac{|\mathrm{Re}^* - \mathrm{Re}|}{\mathrm{Re}} > \varepsilon$)
}
```

Note that the result still is an airfoil polar, even if wing loading and chord length are involved. Only when you additionally specify an aspect ratio on the options card, the polars include the induced drag and approximate a finite wing.

A precaution must be undertaken to handle cases where $C_L \to 0$. Here JAVAFOIL limits the Reynolds number to a value corresponding to a small lift coefficient, e.g. $C_L = 0.02$.

Note: One can also derive the Reynolds number for a constant ratio $\frac{m}{\Lambda}$, eliminating the chord length ℓ. This has not been implemented in JAVAFOIL as it was considered more abstract to think in terms of $\frac{m}{\Lambda}$ instead of the aircraft design parameters $\frac{m}{S}$ and ℓ. But as the relation is $\frac{m}{S} \cdot \ell^2 = \frac{m}{\Lambda}$ it would be sufficient to use $\frac{m}{\Lambda}$ instead of $\frac{m}{S}$ in JAVAFOIL while setting $\ell = 1$.

Correction of Lift for given Aspect Ratio and Mach number

For a given angle of attack, a 3D wing of finite aspect ratio produces less lift than the 2D airfoil section, which corresponds to an infinite aspect ratio. Another correction has to be applied when the Mach number is larger than zero. In subsonic flight more lift is produced when the Mach numbers is increased.
The 3D wing correction is applied only if you specify a value for the aspect ratio of the wing $\Lambda = b^2 / S$ (span b b and wing area S) on the Options card.
The following correction is applied to the lift coefficient of a 2D airfoil C_ℓ in order to approximate the lift coefficient C_L of the 3D wing in compressible flow. The correction is divided into two regimes of aspect ratios.

For small aspect ratios ($\Lambda < 4$) the following formula is used:

$$C_L = \frac{C_\ell}{\sqrt{1 - M_\infty^2 + \left(\frac{2 \cdot \pi}{\Lambda \cdot \pi}\right)^2} + \frac{2 \cdot \pi}{\Lambda \cdot \pi}}$$

If the aspect ratio is larger, $\Lambda \geq 4$, the simplified approximation is applied:

$$C_L = \frac{C_\ell}{\sqrt{1 - M_\infty^2} + \frac{2 \cdot \pi}{\Lambda \cdot \pi}}$$

Implementation in *JavaFoil*

```
public final static double LiftForAspectRatio(double dCl,
                                              double dAspectRatio,
                                              double dMachNumber)
    {
        double dReturn = dCl;

        // correction for finite wings
        if (dAspectRatio > 0.1)
        {
            // Source: Anderson, "Aircraft Performance and Design"
            // lift gradient reduction factor
            // a_0 / (pi*AR)
            double dGradientRatio = 2.0 * Math.PI / (Math.PI *
                                    dAspectRatio);
            if (dAspectRatio < 4.0)
            {
                // low aspect ratio, compressible (Anderson [2.18b])
                dReturn /=
                        (Math.sqrt(1.0 - Math.pow(dMachNumber, 2.0) +
                                   Math.pow(dGradientRatio, 2.0)) +
                         dGradientRatio);
            }
            else
            {
                // high aspect ratio, compressible (Anderson [2.16])
                dReturn /=
                        (Math.sqrt(1.0 - Math.pow(dMachNumber, 2.0)) +
                         dGradientRatio);
            }
        }
        return (dReturn);
    }
```

Determination of Drag for given Aspect Ratio and Mach number

After the lift coefficient of the 2D airfoil for a given angle of attack α has been corrected to the effect of the 3D wing, an approximation of the induced drag is added to the airfoil drag (for the same angle of attack α). This correction is also only applied if you specify a value for the aspect ratio of the wing $\Lambda = b^2 / S$ (span b and wing area S) on the Options card.
As no information about the real wing shape is available, the assumption of having a "good" wing planform is assumed.
Therefore the induced drag component is calculated by using the classical formula derived by lifting line theory (Prandtl).

$$C_{D,i} = k \cdot \frac{C_L^2}{\pi \cdot \Lambda}$$

In JAVAFOIL the "k-Factor" is assumed to be 1.0 (planar wing with elliptical lift distribution).

Note that the idea of these simple corrections is to give you a feeling for the relative importance of the induced drag in relation to the airfoil drag only. For real wing design you should use a more appropriate 3D aerodynamic analysis tool, e.g. a vortex lattice or panel method.

Implementation in *JAVAFOIL*

```
    public final static double DragForAspectRatio(double dCd, double dCl,
                                                  double dAspectRatio,
                                                  double dMachNumber)
    {
        double dReturn = dCd;

        if (dAspectRatio > 0.1)
        {
            // add the induced drag of finite wing according to Prandtl
            dReturn += dCl * dCl / (Math.PI * dAspectRatio);
        }

        return (dReturn);
    }
```

Note that all finite wing results are only approximations. If you need more accurate results, you must use a 3D wing analysis code, which ideally can also handle friction effects.

The Aerodynamic Center

The output of the "Polars" and "Aircraft" cards contains a column with the position of the aerodynamic center (A.C.). The aerodynamic center is a point on the airfoil at which the pitching moment is constant (not necessarily zero) for all angles of attack.
It can be calculated from the gradient of the pitching moment over lift coefficient curve:

$$x_{A.C.} = 0.25 - \frac{\partial C_{m\,0.25}}{\partial C_\ell}.$$

According to thin airfoil theory the aerodynamic center is located at 25% of the chord length and does not move when the angle of attack is changed. In real life airfoils are thick and the location typically can vary about ±2 % around this location.

The aerodynamic center is not to be confused with the center of pressure (C.P.), which is the point at which the total aerodynamic force acts. This total force produces the same effect as the lift and pitching moment. The location of the center of pressure changes with angle of attack and can even move in front or behind the airfoil shape. The center of pressure can be calculated from lift and pitching moment coefficients:

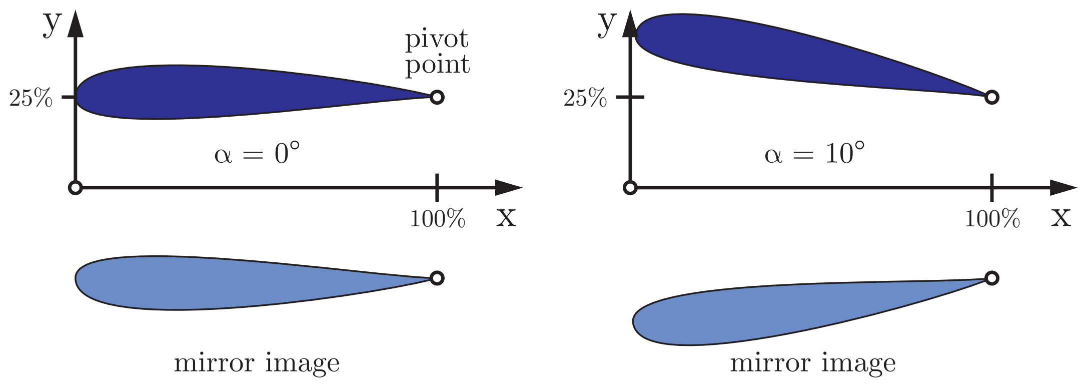
$$x_{C.P.} = 0.25 - \frac{C_{m\,0.25}}{C_\ell}.$$

Note that both, center of pressure as well as the aerodynamic center, are for the airfoil only, not for the complete aircraft with tailplanes.

Ground Effect

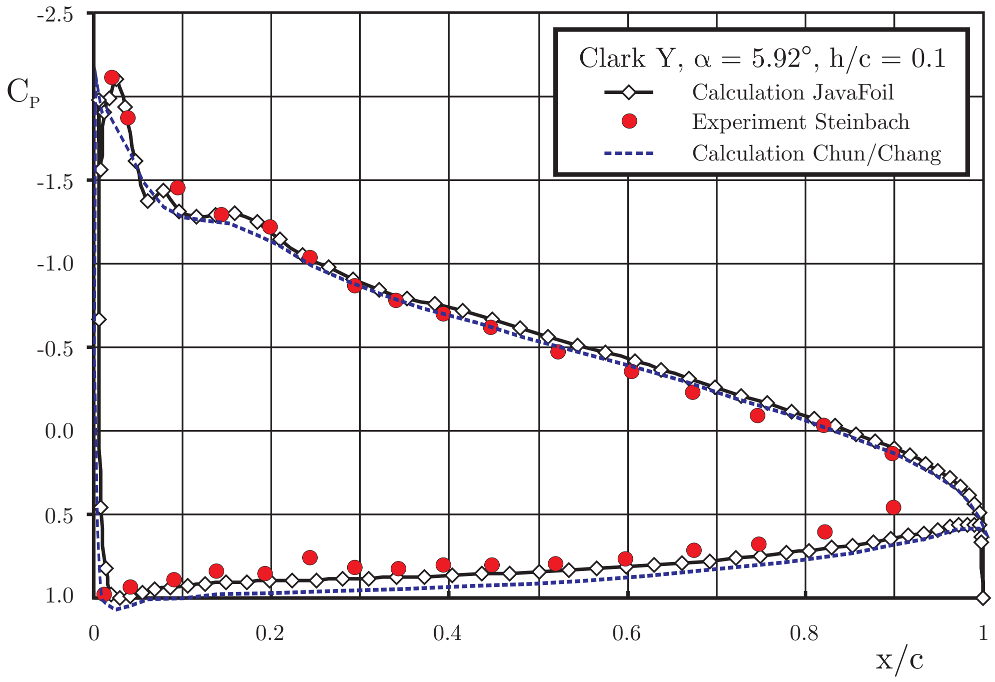
When a wing is brought close to the ground, its characteristics are changed considerably. First the pressure distribution around the two dimensional airfoil shape (a wing of infinite span) is affected by the presence of the ground. Second, the lift and the induced drag of a wing of finite span are affected also.

JAVAFOIL simulates the ground effect on the flow around the two dimensional airfoil by using a mirror image of the airfoil section. The mirror plan is always located at y = 0. Note that for a proper simulation, the baseline airfoil must be translated into the positive y-direction so that it does not intersect the horizontal line y = 0. This translation can be performed using the Modify card.
In contrast to the flow around a free airfoil, where the flow field can be constructed from a superposition of the solutions for zero and 90 degrees angle of attack, the ground effect case cannot be created by superposition. Any change of angle of attack also changes the geometry of the airfoil and mirror airfoil pair. Therefore a new panel solution is required for each angle of attack, which slows down the calculation of a polar somewhat.



Changing the angle of attack in ground effect rotates around the pivot point.

The angle of attack of the airfoil is always changed by rotating the section around the pivot point specified on the Modify card. If you want to analyze an airfoil at a height of 25% of the chord length and want to maintain the trailing edge point, you would first translate the airfoil in y-direction by 25% and then set the pivot point to x=100%,y=25%. Then any subsequent change of angle of attack would maintain the trailing edge point and elevate the nose of the airfoil above the y=25% line. Note that the airfoil is rotated and thus its projection on the x/c axis becomes shorter, but pressure, velocity or Mach number distributions on the Velocity card are still plotted over x/c.



Distributions of the pressure coefficient on a Clark Y airfoil in ground proximity.

Results of the numerical solution of the Navier-Stokes equations have been taken from [17], the experimental results have been reproduced from [18]. The experiments were carried out with a fixed ground board, equipped with a suction system. The results of JAVAFOIL match the experimental results quite well. The Navier-Stokes solutions should model boundary layer displacement effects more accurately.

The ground effect on a wing of a finite span is approximated by applying a modified calculation of the induced drag. If you specify the aspect ratio of the wing $\Lambda = b^2 / S$ (span b and wing area S), and the height of the wing above ground h/b (height h over wing span b) on the Options card, these values are used to calculate an approximation of the induced drag using

$$C_{D,i} = \frac{C_L^2}{\pi \cdot \Lambda} \cdot \left(1 - \frac{33 \cdot (h/b)^{1.5}}{1 + 33 \cdot (h/b)^{1.5}}\right).$$

Multi-Element Airfoils

The maximum lift of single element airfoils is limited by onset of flow separation. The achievable limit for single element airfoils seems to be at lift coefficients between 2 and 3. For maximum lift it can be beneficial to split an airfoil unto several elements, arranged to form a slotted cascade. Each element then develops its own fresh boundary layer and positive interference effects between the elements allow for a higher lift loading per element. JAVAFOIL can handle such multi-element airfoils to a limited extent. Limitations are imposed by the fact that boundary layer effects are not modeled. Therefore inaccurate results must be expected when slots are very narrow (less than twice the displacement thickness of the boundary layer and when the wake of a leading element interacts with a following element. Nevertheless JAVAFOIL should be useful to produce a reasonable first design for a slotted airfoil with appropriate gap, overlap and element angle settings. Also one can design such sections so that suction peaks and too steep pressure gradients are avoided.

The following script shows how a two element downforce wing section can be generated starting with a basic NACA 4-digit section.

```
//
// A simple JavaFoil example which creates a two element
// airfoil for downforce generation.
//
// switch to US country settings
Options.Country(0)
// create a cambered NACA airfoil for starting
Geometry.CreateAirfoil(0;61;15;30;6;40,000;0;0;1)
//
// create a copy of this first airfoil element to be used later
Modify.Select(1)
Modify.Duplicate()
// now select the first element again
Modify.Select(1)
// ... flip it upside down
Modify.Flip(25;0)
// ... scale it to 75%
Modify.Scale(75)
// ... rotate it 5 degrees trailing edge up around its nose (0;0)
Modify.Rotate(0;0;-5)
//
// now select the second element
Modify.Select(2)
// ... flip it upside down
Modify.Flip(25;0)
// ... scale this copy to 30%
Modify.Scale(30)
// ... move it back so that there is 5% overlap
Modify.Move(70.0;12)
// ...rotate it by 30 degrees around a point at (70%/12%)
Modify.Rotate(70;12;-30)
// finally: (THIS IS OMPORTANT!)
// select both elements again for all further analyses
// if only one element is selected on the modify card, only this element
// will be considered during the calculation of polars etc.!
Modify.Select(1;2)
// and move both elements up by 25% for ground clearance
// (note that the airfoil may not cross the y=0 line which is the ground
plane)
Modify.Move(0.0;25)
//
// switch ground effect simulation ON
Options.GroundEffect(1)
//
// prepare for analysis
Options.MachNumber(0)
Options.StallModel(0)
Options.TransitionModel(1)
Options.AspectRatio(0)
// velocity versus x/c should show no strong suction peaks in the
// nose region of 2nd element
Velocity.Analyze(0;0;1;0)
//
// polar for Re=500'000, alfa=-15 to +10 degrees
// with ground present a string suction force is generated
// (even for single element airfoils)
// therefore a Cl_min in the order of -5 to -6 can be seen.
Polar.Analyze(500000;500000;500000;-15;10;1;100;100;0;0)
//
// finally: export coordinates in XML format
Geometry.Save("Z:\groundforce-example.xml")
```

Automating JAVAFOIL with a Script

Sometimes it is useful to have JAVAFOIL execute a command sequence in a script automatically and then terminate. This allows running JAVAFOIL inside a parameter sweep or as part of an optimization loop. For this purpose you prepare the script file and start JavaFoil with the name of the script file. Then JAVAFOIL runs invisible, without opening a window, and executes the commands in the script file.

The name of the script file can be transferred to JAVAFOIL in two ways. First you can define a "system property" using the "-D" command line option of the java command, like so:

```
java.exe -DScript="Path\Script" -cp "Path\mhclasses.jar" -jar "Path\javafoil.jar"
```

secondly, you can specify the script file using as a command line argument to JAVAFOIL like this:

```
java.exe -cp "Path\mhclasses.jar" -jar "Path\javafoil.jar" Script="Path\Script"
```

Both ways methods are equivalent and produce the same result.

Note that this example in Windows style uses the backslash as file separator, for other Unix-like systems you have to use the appropriate separator, usually a forward slash.

Note:
As JAVAFOIL is running without showing a window, you must make sure that the script ends with an Exit() command to terminate the JAVAFOIL run properly. Otherwise, the JAVAFOIL process will continue to run. In Windows you can check for running JAVAFOIL processes using the Task Manager window. In Unix-like operating systems you can use the "ps" command to list all processes running under your user account.

References

[1] Mark Drela, Michael B. Giles, "Viscous-Inviscid Analysis of Transonic and Low Reynolds Number Airfoils", AIAA-86-1786-CP, 1986

[2] Xiao-liang Wang, Xue-xiong Shan, "Shape Optimization of Stratosphere Airship", Journal of Aircraft V43N1, 2006.

[3] Ira Abbott and Albert Von Doenhoff, "Theory of Wing Sections".

[4] Eastman N. Jacobs, Kenneth E. Ward, Robert M. Pinkerton, "The Characteristics of 78 Related Airfoil Sections from Tests in the Variable-Density Wind Tunnel", NACA Rep. No. 460, 1933.

[5] John Stack, "Tests of Airfoils Designed to Delay the Compressibility Burble", NACA Rep. No. 763, 1943.

[6] Charles L. Ladson and Cuyler W. Brooks, Jr., "Development of a Computer Program To Obtain Ordinates for NACA 6- and 6A-Series Airfoils", NASA Technical Memorandum TM X-3069, September, 1974.

[7] Charles L. Ladson, Cuyler W. Brooks, Jr. and Acquila S. Hill, "Computer Program To Obtain Ordinates for NACA Airfoils", NASA Technical Memorandum 4741, December, 1996.

[8] A. K. Martynov, "Practical Aerodynamics", Pergamon Press 1965.

[9] A. C. Kravets, "Characteristics of Aircraft Profiles", Moscow 1939.

[10] Anonymous, "Measurement of Force Coefficients on the Aerofoils EC 1240 and EC 1240/0640 in the High-Speed Tunnel at the National Physical Laboratory", British ARC R&M 2246.
[11] S. Goldstein, "Approximate Two-Dimensional Aerofoil Theory. Part I. Velocity Distributions for Symmetrical Aerofoils", British A.R.C. Technical Report C.P. No. 68, 1952.
[12] British A.R.C. R&M 4726.
[13] British A.R.C. R&M 4978.
[14] Katz, Plotkin, "Low-Speed Aerodynamics", McGraw-Hill, 1991.
[15] H. B. Helmbold, F. Keune, "Beiträge zur Profilforschung, II. Geometrie der Profilsystematik", Luftfahrt-Forschung, Band XX, 1943, pp 81 ff.
[16] G. Roßner, "Über eine Klasse von theoretischen Profilen mit vier frei wählbaren geometrischen Parametern", Jahrbuch 1942 der Deutschen Luftfahrtforschung, p.I/141-I/159, 1942.
[17] H. H. Chun, R. H. Chang, "Turbulence Flow Simulation for Wings in Ground Effect with two Ground Conditions: fixed and moving Ground", International Journal of Maritime Engineering, Royal Institution of Naval Architects, 2003.
[18] D. Steinbach, "Comment on Aerodynamic Characteristics of a Two-Dimensional Airfoil with Ground Effect", Journal of Aircraft, Col. 34, No. 3, pp. 455-456, 1996.
[19] W. Würz, "Hitzdrahtmessungen zum laminar-turbulenten Strömungsumschlag in anliegenden Grenzschichten und Ablöseblasen sowie Vergleich mit der linearen Stabilitätstheorie und empirischen Umschlagskriterien", Dissertation, Institut für Aero- und Gasdynamik, Universität Stuttgart, 1995.
[20] R. Eppler, D. Somers, „A Computer Program fort he Design and Analysis of Low-Speed Airfoils", NASA TM-80210, 1980.
[21] R. Eppler, „Airfoil Design and Data", Springer-Verlag, 1990.
[22] T. Cebeci, P. Bradshaw, "Momentum Transfer in Boundary Layers", Hemisphere Publishing Corporation, Washington, London, 1977
[23] A. R. Wazzan, C. Gazley, A. M. O. Smith, "H-Rx method for Predicting Transition", AIAA Journal, Vol. 19, No. 6, June 1981, pp.810-811.