EPSILON V2.0 - USER'S MANUAL

Miguel Angel AGUIRRE and Sébastien DUPLAA

March 4, 2020

SUMMARY

NOMENCLATURE

$\dot{\mathcal{A}}$	total anergy rate, W
$\dot{\mathcal{A}}_{\Phi}$	viscous anergy rate, W
$\dot{\mathcal{A}}_{\nabla T}$	thermal anergy rate, W
$\dot{\mathcal{A}}_{w}$	shockwave anergy rate, W
a	speed of sound, m.s^{-1}
c	airfoil chord, m
C_L	lift coefficient
C_D	drag coefficient
$C_{D\varepsilon}$	exergy-based drag coefficient
C_{Dv}	vortex (induced) drag coefficient
$C_{D\,visc}$	viscous drag coefficient
C_{Dp}	profile drag coefficient
C_{Dw}	wave drag coefficient
C_M	moment coefficient
C_p	pressure coefficient
C_Y	sideforce coefficient
c_p, c_v	mass specific heat at constant pressure and volume respectively, J.kg^{-1}.K^{-1}
D	drag force, N
e	mass specific internal energy, J.kg-1
$\dot{E}_u$	axial kinetic exergy rate, W
$\dot{E}_v$	transverse kinetic exergy rate, W
$\dot{E}_p$	boundary-pressure work rate, W
F_y	Sideforce, N
G	Green function
h_s, h_t	mass specific static and total enthalpy respectively, J.kg^{-1}
$\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$	unit vectors along the aerodynamic x-, y- and z-axes
I,J,K	unit vectors along the mesh X-, Y- and Z-axes
L	Lift force, N
$\vec{\boldsymbol{l}}$	Lamb vector
M	Mach number (= u_0/a_0)
$\vec{\boldsymbol{n}}$	n_x **i**, n_y **j**, n_z **k**, local surface normal
$\vec{\boldsymbol{P}}$	x**i**, y**j**, z**k**, point coordinates in the aerodynamic reference frame, m
$\vec{\boldsymbol{P}}_{\boldsymbol{mesh}}$	X **I**, Y **J**, Z **K**, point coordinates in the mesh reference frame, m
P_s, P_t	static and total pressure, Pa
Pr	Prandtl number (= $c_p\,\mu\,/\,k$)
q	dynamic pressure, Pa
R	gas constant, J.kg^{-1}.K^{-1}
Re	Reynolds number (= $\rho_0\,u_0\,c\,/\,\mu_0$)
R_x, R_y, R_z	rotation matrix about mesh X-, Y- and Z-axes
S	reference surface, m^2

s	mass specific entropy, J.kg^{-1}.K^{-1}
$\bar{\bar{\mathbf{T}}}$	transformation matrix from the mesh XYZ axes to the aerodynamic xyz axes
T_s, T_t	static and total temperatures, K
$\mathcal{V}$	domain volume
$\vec{\boldsymbol{V}}$	u**i**, v**j**, w**k**, local velocity vector in the aerodynamic reference frame, m.s^{-1}
$\vec{\boldsymbol{V}}_{\boldsymbol{mesh}}$	V_x**I**, V_y**J**, V_z**K**, local velocity vector in the mesh reference frame, m.s^{-1}

Greek letters

α	angle of attack, deg
β	angle of sideslip, deg
$\delta(\ )$	$(\ ) - (\ )_0$, local variation of a parameter respect to the upstream reference value
$\Delta(\ )$	$\delta(\ )/(\ )_0$, non-dimensional local variation of a parameter respect to the upstream value
$\dot{\varepsilon}_m$	mechanical exergy rate, W
$\dot{\varepsilon}_{th}$	thermal exergy rate, W
ϕ	potential velocity function, m^2.s^{-1}
Φ_{eff}	effective dissipation, W.m^{-3}
Γ	circulation, m^2.s^{-1}
γ	ratio of specific heats
k_{eff}	effective thermal conductivity, W.m^{-1}.K^{-1}
μ, μ_t	laminar and turbulent dynamic viscosities, kg.m.s^{-1}
ρ	air density, kg.m^{-3}
σ	flow source/sink, s^{-1}
$\bar{\bar{\boldsymbol{\tau}}}$	viscous stress tensor, Pa
ν	kinematic viscosity, m^2.s^{-1}
ψ	stream function, m^2.s^{-1}
$\vec{\boldsymbol{\omega}}$	ξ**i**, η**j**, ζ**k**, local vorticity vector in the aerodynamic reference frame, s^{-1}
$\vec{\boldsymbol{\omega}}_{\boldsymbol{mesh}}$	ω_x**I**, ω_y**J**, ω_z**K**, local vorticity vector in the mesh reference frame, s^{-1}

Subscripts

0	upstream reference values
b	body
eff	effective
out	outlet section (survey plane)
ref	reference
w	wake

Acronyms

CFD	Computational Fluid Dynamics
WTT	Wind Tunnel Testing

DISCLAIMER

⚠	This plugin was developed for research purposes and released under **GNU GPL V3** license **without any guarantee of any kind**, either express or implied. Its use for teaching, research and industrial applications is allowed under the strict responsibility of the user. The authors of this plugin are not responsible for any damage or error related to the use of the plugin.

SUMMARY

The **first chapter** of this user's manual deals with the theoretical background about the far-field and the exergy methods, as wells as its related formulae. Then, the **second chapter** presents the architecture and scope of the set of plugins. **Chapter 3** explains the installation procedure and the **Chapters 4 and 5** gives a detailed user's guide in order to fully exploit the capabilities of the plugin set.

EPSILON

Epsilon (*) is an open source aerodynamic analysis tool suited for CFD and experimental data. It allows the user to perform an aerodynamic analysis by using several methods: near field, far-field, Lamb vector and exergy methods.

Epsilon is a set of plugins for Paraview created to provide an easy to use tool for the post processing of 2D and 3D data as shown in Fig.1.

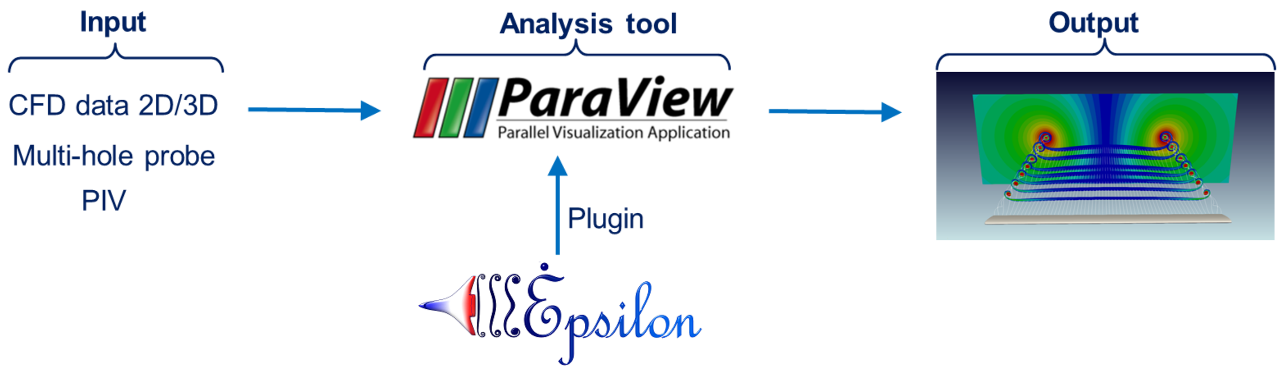


Figure 1: How to perform an aerodynamic analysis of CFD or experimental data with Epsilon

The code was developed by ISAE for research purposes (Development of new exergy formulations) and now is released to the public domain under **GNU GPL V3 license**. It can be downloaded from this internet site:

https://Epsilon-Exergy.isae-supaero.fr

The site also contains plenty of information complementary to this user manual (i.e., open data and theory). In particular, it is advised to watch the video tutorials that will help the user to quickly exploit all the capabilities offered by Epsilon:

https://websites.isae-supaero.fr/epsilon-exergy-analysis-tool/tutorials/

Please, cite the following paper in any published research or other work utilizing Epsilon:

Aguirre, M., Duplaa, S., "**Epsilon: An Open Source Tool for Exergy-Based Aerodynamic Analysis**", Aerospace Europe Conference, 25-28 February 2020, Bordeaux, France.

(*) Since this software was originally developed in order to perform an exergy analysis (whose symbol is "ε"), it has been named "Epsilon".

CHAPTER 1: BACKGROUND THEORY REVIEW

This chapter gives an introduction to the theory needed to understand the near-field, far-field, Lamb vector and exergy methods. The equations and methodology presented here are used later to explain the architecture and functioning of the related plugins.

1 CONVENTIONAL REFERENCE FRAME

All the formulations discussed below use a standard reference frame as described in the Figure 2. It considers a reference system where the x-axis is aligned with the upstream flow direction and pointing rearwards, the y-axis points towards the RHS of the body and the z-axis points upwards. The x-, y-, z-axes will be called "aerodynamic axes" (or "wind axes"). Moreover, when control volume formulations are used, it is supposed that the outlet section of the control volume is a plane (called "survey plane") and it is placed normal to the x-axis (i.e., normal to the upstream flow direction). Also, the lateral surfaces are considered parallels to the upstream direction.

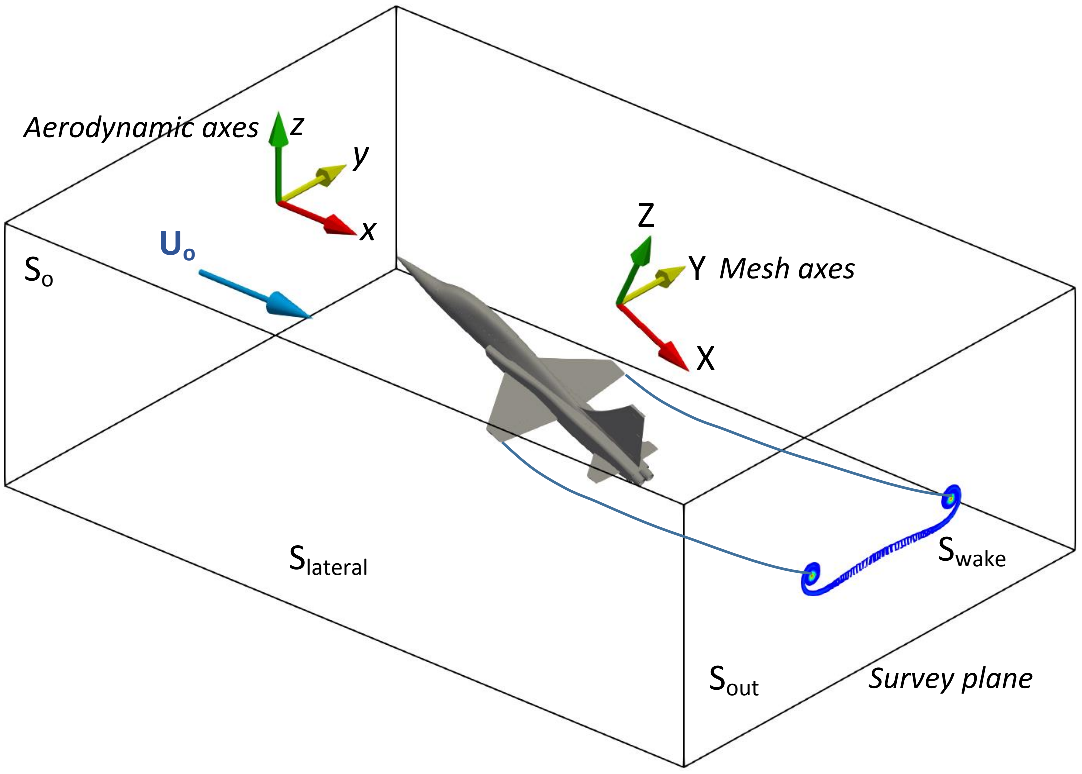


Figure 2: conventional reference frame

1.1 Angle of attack transformation

In a wind tunnel testing the aerodynamic axes are coincident with the tunnel axes. However, in a CFD case the mesh axes (X, Y, Z) can be different from the aerodynamic axes (x, y, z). Moreover, there is no standard practice to assign the axis pointing upwards: it can be Y or Z (usually, X is reserved for the longitudinal axis). Hence, in the most general case a transformation matrix "$\overline{\overline{\mathbf{T}}}$" is needed in order to express all the mesh vector fields in the aerodynamic reference frame. This transformation matrix is composed of three rotational transformation matrices "$\overline{\overline{\boldsymbol{R}}}_{\boldsymbol{x}}$", "$\overline{\overline{\boldsymbol{R}}}_{\boldsymbol{y}}$" and "$\overline{\overline{\boldsymbol{R}}}_{\boldsymbol{z}}$" as follows:

$$\overline{\overline{\boldsymbol{R}}}_{\boldsymbol{x}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(angle_x) & -\sin(angle_x) \\ 0 & \sin(angle_x) & \cos(angle_x) \end{bmatrix} \quad (1)$$

$$\overline{\overline{\boldsymbol{R}}}_{\boldsymbol{y}} = \begin{bmatrix} \cos(angle_y) & 0 & \sin(angle_y) \\ 0 & 1 & 0 \\ -\sin(angle_y) & 0 & \cos(angle_y) \end{bmatrix} \quad (2)$$

$$\overline{\overline{\boldsymbol{R}}}_{\boldsymbol{z}} = \begin{bmatrix} \cos(angle_z) & -\sin(angle_z) & 0 \\ \sin(angle_z) & \cos(angle_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3)$$

These are the rotational matrices about the mesh X-, Y- and Z-axes ("angle$_x$", "angle$_y$" and "angle$_z$" are the rotation angles about those axes). If the Z-mesh axis points upwards, then: angle$_x$=0, angle$_y$=- α, angle$_z$ =β.

The rotational matrices can be used to obtain the transformation matrix "**T**" from the mesh reference system (XYZ) to the aerodynamic reference system (xyz):

$$\overline{\overline{\boldsymbol{T}}}_{\boldsymbol{x}} = \overline{\overline{\boldsymbol{R}}}_{\boldsymbol{x}}{}^{T}.\overline{\overline{\boldsymbol{R}}}_{\boldsymbol{x}}.\overline{\overline{\boldsymbol{R}}}_{\boldsymbol{x}}{}^{T} \quad (4)$$

$$\overline{\overline{\boldsymbol{T}}}_{\boldsymbol{y}} = \overline{\overline{\boldsymbol{R}}}_{\boldsymbol{y}}{}^{T}.\overline{\overline{\boldsymbol{R}}}_{\boldsymbol{y}}.\overline{\overline{\boldsymbol{R}}}_{\boldsymbol{y}}{}^{T} \quad (5)$$

$$\overline{\overline{\boldsymbol{T}}}_{\boldsymbol{z}} = \overline{\overline{\boldsymbol{R}}}_{\boldsymbol{z}}{}^{T}.\overline{\overline{\boldsymbol{R}}}_{\boldsymbol{z}}.\overline{\overline{\boldsymbol{R}}}_{\boldsymbol{z}}{}^{T} \quad (6)$$

$$\overline{\overline{\boldsymbol{T}}} = \overline{\overline{\boldsymbol{T}}}_{\boldsymbol{x}}.\overline{\overline{\boldsymbol{T}}}_{\boldsymbol{y}}.\overline{\overline{\boldsymbol{T}}}_{\boldsymbol{z}} \quad (7)$$

The transformation matrix "**T**" can be used to transform the velocity vector in the mesh reference system to the aerodynamic reference frame as follows:

$$\vec{\boldsymbol{V}} = \overline{\overline{\boldsymbol{T}}}.\vec{\boldsymbol{V}}_{\boldsymbol{mesh}} \quad (8)$$

With:

$$\vec{\boldsymbol{V}}_{\boldsymbol{mesh}} = Vx\,\vec{\boldsymbol{I}} + Vy\,\vec{\boldsymbol{J}} + Vz\,\vec{\boldsymbol{K}} \quad (9)$$

$$\vec{\boldsymbol{V}} = u\,\vec{\boldsymbol{\imath}} + v\,\vec{\boldsymbol{\jmath}} + w\,\vec{\boldsymbol{k}} \quad (10)$$

Where "$\vec{\boldsymbol{V}}$" is the velocity vector, "Vx, Vy, Vz" are the X-, Y- and Z-velocity components in the mesh reference frame and "u, v, w" are the x-, y- and z-velocity components in the aerodynamic reference frame.

By following the same approach, the mesh vorticity vector can be referred to the aerodynamic axes:

$$\vec{\boldsymbol{\omega}} = \overline{\overline{\boldsymbol{T}}}.\vec{\boldsymbol{\omega}}_{\boldsymbol{mesh}} \quad (11)$$

With:

$$\vec{\boldsymbol{\omega}}_{\boldsymbol{mesh}} = \omega_x\,\vec{\boldsymbol{I}} + \omega_y\,\vec{\boldsymbol{J}} + \omega_z\,\vec{\boldsymbol{K}} \quad (12)$$

$$\vec{\boldsymbol{\omega}} = \xi\,\vec{\boldsymbol{\imath}} + \eta\,\vec{\boldsymbol{\jmath}} + \zeta\,\vec{\boldsymbol{k}} \quad (13)$$

The same is also valid for the points coordinates:

$$\vec{\boldsymbol{P}} = \bar{\bar{\boldsymbol{T}}}.\vec{\boldsymbol{P}}_{\boldsymbol{mesh}} \tag{14}$$

With:

$$\vec{\boldsymbol{P}}_{\boldsymbol{mesh}} = \mathrm{X}\,\vec{\boldsymbol{I}} + \mathrm{Y}\,\vec{\boldsymbol{J}} + \mathrm{Z}\,\vec{\boldsymbol{K}} \tag{15}$$

$$\vec{\boldsymbol{P}} = x\,\vec{\boldsymbol{\imath}} + \mathrm{y}\,\vec{\boldsymbol{\jmath}} + z\,\vec{\boldsymbol{k}} \tag{16}$$

Any other vector is transformed by using the same approach. Moreover, the mesh tensors (like the viscous stress tensor) can also be referred to the aerodynamic axes:

$$\bar{\bar{\boldsymbol{\tau}}} = \bar{\bar{\boldsymbol{T}}}.\bar{\bar{\boldsymbol{\tau}}}_{\boldsymbol{mesh}}.\bar{\bar{\boldsymbol{T}}}^{T} \tag{17}$$

2 UPSTREAM AERODYNAMIC DATA

All the aerodynamic analysis formulations discussed later require knowing in advance the upstream reference aerodynamic and thermodynamic parameters. The user of Epsilon usually knows some parameters there:

- Upstream total pressure: P_{t_0}
- Upstream total temperature: T_{t_0}
- Upstream speed: U_0

These are the only imposed upstream data. The remaining upstream parameters may be obtained directly by considering a locally isentropic-isoenergetic flow:

- Upstream static temperature: $T_{s_0} = T_{t_0} - \frac{{U_0}^2}{2\,c_p}$ (18)
- Upstream speed of sound: $a_0 = \sqrt{\gamma\ R\,T_{s_0}}$ (19)
- Upstream Mach number: $M_0 = \frac{U_0}{a_0}$ (20)
- Upstream static pressure: $P_{s_0} = P_{t_0} / \left(1 + \frac{\gamma-1}{2}{M_0}^2\right)^{\frac{\gamma}{\gamma-1}}$ (21)
- Upstream density: $\rho_0 = \frac{P_{s_0}}{R\,T_{s_0}}$ (22)
- Upstream static enthalpy: $h_{s_0} = c_p\,T_{s_0}$ (23)
- Upstream total enthalpy: $h_{t_0} = c_p T_{t_0}$ (24)
- Upstream internal energy: $e_0 = c_v T_{s_0}$ (25)
- Upstream dynamic pressure: $q_0 = \frac{1}{2}\,\rho_0\,{U_0}^2$ (26)

Where:

- Mass specific heat capacity at constant pressure: $c_p = \gamma * R/(\gamma-1)$ (27)
- Mass specific heat capacity at constant volume: $c_v = c_p/\gamma$ (28)

Sometimes the upstream static parameters are known instead of the total parameters. Also, the upstream Mach number may be known instead of the upstream velocity. In any case, the equations presented before can be used to compute the remaining upstream data.

The values used in Epsilon for the gas constants parameters were extracted from the properties table of the Fluent solver (using these values, a high accuracy of results is guaranteed):

Fluent Air Data		
R	8.314462618	J/(K mol)
m/M	28.966	g/Mol
r	287.04213969	J/Kg K
c_p	1006.43	J/Kg K
c_v	719.38786031	J/Kg K
γ	1.399008873407	

Table 1: reference values

3 DRAG BOOKKEEPING

Epsilon is a software capable of predicting lift and drag based on CFD and experimental data. It also allows calculating these values by several methods and to decompose it as well. The drag breakdown is performed by using the standard drag bookkeeping listed below, and depicted in Fig. 3, 4 and 5:

- Pressure drag: drag force related to the pressure distribution acting upon a body
- Friction drag: drag force related to the skin friction distribution acting upon a body
- Total near-field drag: Pressure drag + Friction drag

Near-field method

- Viscous drag: drag force due to the losses inside the viscous boundary layer
- Wave drag: drag force related to the losses occurred across the shockwave
- Profile drag: defined as Viscous drag+ Wave drag
- Induced drag: drag force related to the axial vorticity created by the body
- Total far-field drag: Profile drag + Induced drag

Far-field method
Lamb Vector method
Exergy method

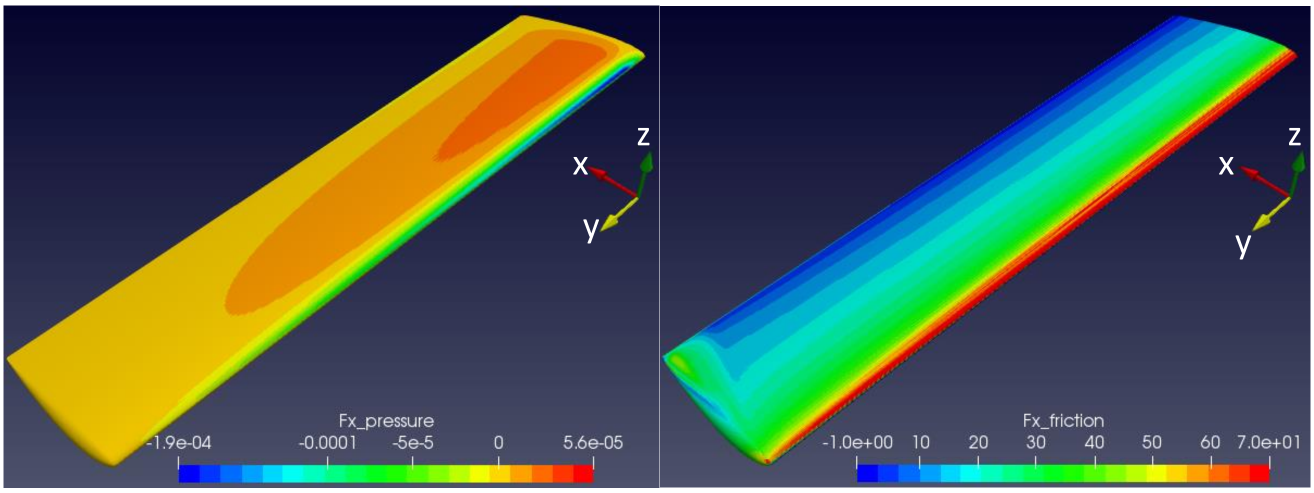


Figure 3: pressure drag (left) and friction drag (right)

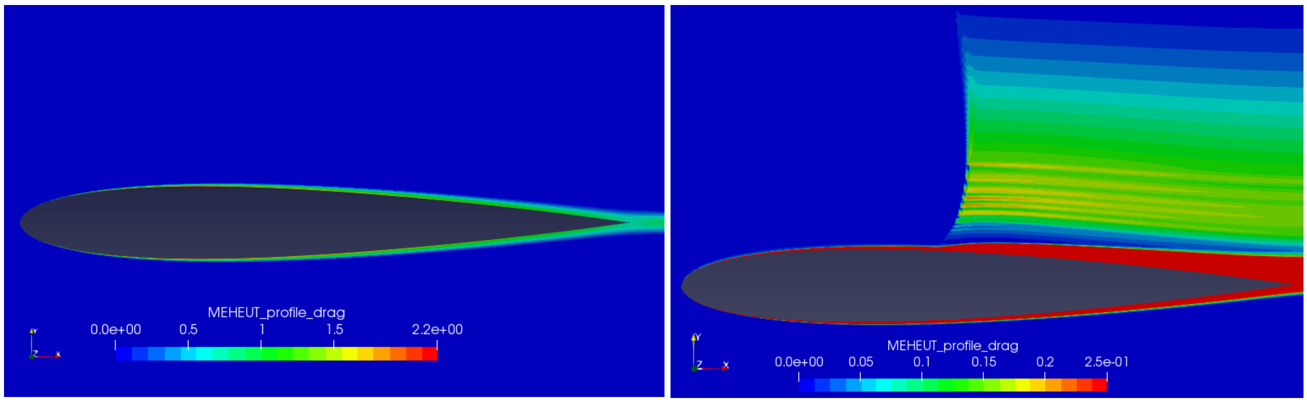


Figure 4: viscous drag (left) and wave drag (right)

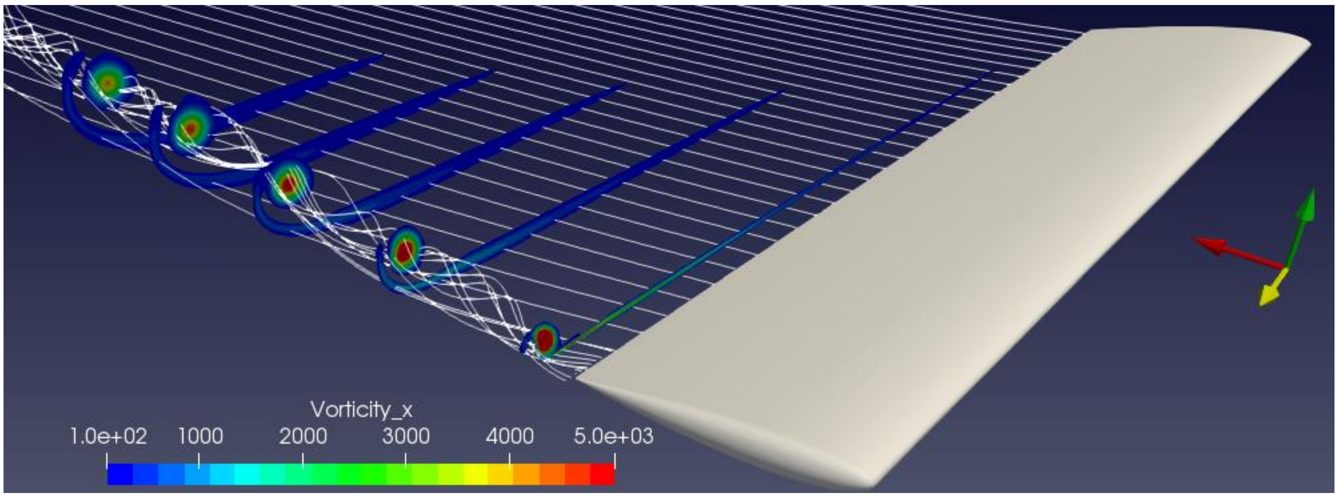


Figure 5: induced drag (also known as "vortex drag")

4 NEAR-FIELD METHOD

The most widely used method for aerodynamic assessment is the so-called near-field method, which allows calculating lift, drag and moments by making a surface integral [2,3]:

$$\overrightarrow{F_b} = \int_{S_b} (Ps\,\vec{n} - \bar{\bar{\tau}}\,.\vec{n})\,dS_b \qquad (29)$$

Where "$\overrightarrow{F_b}$" is the vector force acting on the body, "Ps" is the static pressure, "$\bar{\bar{\tau}}$" is the viscous stress tensor, "$\vec{n}$" is the vector normal to the surface, "S_b" the surface of the body.

This approach has the advantage to be very simple and easy to understand. Moreover, its implementation in a CFD code is straightforward. However it does not allows a phenomenological decomposition of the drag (other than pressure and friction drag) and it does not provide a complete understanding of the physics, which highlights its drawbacks for design purposes.

5 FAR-FIELD METHODS

A more powerful aerodynamic assessment can be made by using the so-called far-field methods. All of them use the momentum conservation approach to define a set of equations allowing making a phenomenological decomposition of drag, while giving at the same time a good insight into the physics. The methods implemented in Epsilon are the followings:

- Classical momentum conservation method
- Betz method
- Jones method
- Oswatitsch method
- Maskell method
- Van Der Vooren method
- Giles method
- Kusunose method
- Meheut method
- Onorato

All these approaches are valid only for stationary flows. No unsteady formulations are considered in Epsilon. Moreover, several of these formulations were developed for wind tunnel wake analysis (they require data inside the wake only) but their use for the analysis of CFD data is highly recommended since they provide a deeper physical understanding of the flow field.

5.1 Momentum conservation method

This approach [4] applies the momentum conservation equation to a control volume surrounding the body:

$$\int_{S_b+S_0+S_{lat}+S_{out}} \left[\rho \vec{V} (\vec{V} . \vec{n}) + P_s \vec{n} - \bar{\bar{\tau}} . \vec{n}\right] dS = \vec{0} \tag{30}$$

Where "S$_o$", "S$_{lat}$" and "S$_{out}$" are the upstream, lateral and outlet faces of the control volume respectively (external faces of the control volume), "S$_b$" is the body surface (internal face of the control volume) and "ρ" the local density. This equation can be rearranged in the following way, where the first term is equal to the near-field forces acting upon the body (Eq. 29):

$$\underbrace{\int_{S_b} [P_s \vec{n} - \bar{\bar{\tau}} . \vec{n}]\, dS}_{\text{Near-field forces}} = \underbrace{- \int_{S_0+S_{lat}+S_{out}} \left[\rho \vec{V} (\vec{V} . \vec{n}) + P_s \vec{n} - \bar{\bar{\tau}} . \vec{n}\right] dS}_{\text{Far-field forces}} \tag{31}$$

$$\vec{F_b} = - \int_{S_0+S_{lat}+S_{out}} \left[\rho \vec{V} (\vec{V} . \vec{n}) + P_s \vec{n} - \bar{\bar{\tau}} . \vec{n}\right] dS \tag{32}$$

This highlights the fact that the near-field forces can be obtained by analyzing the flow parameters on the external surfaces of the control volume (i.e., near-field forces = far-field forces). This equation can be further modified in order to explicitly take into account the upstream parameters:

$$\vec{F_b} = -\int_{S_0+S_{lat}+S_{out}} \left[\rho\left(\vec{V}-\vec{V_0}\right)\left(\vec{V}\,.\vec{n}\right)+\left(P_s-P_{s_0}\right)\vec{n}-\bar{\bar{\tau}}\,.\vec{n}\right] dS \tag{33}$$

Where "$\vec{V_0}$" is the upstream vector, given by:

$$\vec{V_0} = U_0\,\vec{\imath} + 0\,\vec{\jmath} + 0\,\vec{k}$$

This is an exact equation that allows obtaining the lift and total drag of a body. It is valid for compressible and incompressible flows, for flows with or without lifting bodies and/or power. Moreover, it does not impose limitations concerning the position of the outlet plane: the survey plane can be placed anywhere downstream of the body. However it requires integrating the entire outlet section of the control volume (its integral cannot be reduced to the wake only). Another major drawback is its inability to decompose drag.

In most applications, the viscous tensor term is usually neglected. Also, the upstream and lateral surfaces are considered to be far away. This leads to the simplified momentum conservations equations for lift "L", sideforce "Fy" and total drag "D_{Tot}", that will be widely used by all the other far-field methods as starting point for its equation development:

$$L = -\int_{S_{out}} \rho\, w\, u\;\, dS \tag{34}$$

$$Fy = -\int_{S_{out}} \rho\, v\, u\;\, dS \tag{35}$$

$$D_{Tot} = -\int_{S_{out}} \left[\rho\,(u-U_0)\,(u)+\left(P_s-P_{s_0}\right)\vec{n}\right] dS \tag{36}$$

An assessment of the stress tensor influence on the resulting force can be made by taking into account its effect on the drag:

$$D_{stress} = -\int_{S_{out}} \tau_{xx}\, dS \tag{37}$$

With:

$$\tau_{xx} = \mu\left[2\frac{\partial u}{\partial x}-\frac{2}{3}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right)\right]+R_{xx} \tag{38}$$

Where "R_{xx}" is the Reynolds stress term, given by the time average of velocities and densities fluctuations:

$$R_{xx} = -\rho\overline{u'^2} - 2u\overline{\rho' u'} - \overline{\rho' u'^2} \tag{39}$$

The previous equations are implemented in Epsilon. The integrand of each equation is evaluated at any point of the CFD domain in order to perform flow visualizations. The user can then perform a surface integration in order to obtain the desired drag or lift values. The chapters 2, 4 and 5 will explain the data analysis procedures in detail. Also, in the Appendix 1 there is a list of the variables names calculated by Epsilon at each point. (It can be an entire equation or a single term of it). For the momentum conservation method, all the Epsilon variables are called "MOMENTUM_" + the variable name.

5.2 Betz method

This method was developed by Betz in 1925 [5, 6] and it allows calculating the profile drag of a body in a wind tunnel testing. It uses the simplified total drag momentum equation (Eq. 36) and introduces the concept of artificial velocity (known nowadays as "isentropic velocity" [30]) as an attempt to reduce the integral to the wake. This artificial velocity is a fictitious one which is equal to the actual speed outside of the wake, but it is larger inside of it. In fact it is intended to give the velocity profile if no loss of energy where present (i.e., the potential or inviscid component of the local velocity):

$$u^{*2} = u^2 + \frac{2}{\rho}\left(P_{t_0} - P_t\right) \tag{40}$$

Where "u*" is the artificial velocity and "Pt" the local total pressure. Betz also proposes to introduce the axial velocity perturbation concept, i.e., to express the local artificial velocity as a perturbation " u' " of the upstream velocity:

$$u^* = U_0 + u' \tag{41}$$

This allows converting the simplified total drag momentum equation into the Betz's profile drag equation:

$$\mathrm{D_p} = \int_{\mathrm{S_{wake}}} \left[\left(P_{t_0} - P_t\right) + \frac{\rho}{2}\,(\mathrm{u}^* - \mathrm{u})\,(\mathrm{u}^* + \mathrm{u} - 2\mathrm{U}_0)\right] \mathrm{dS} \tag{42}$$

Where "S_{wake}" is the cross-section surface of the wake at the survey station. This equation is valid only for incompressible flows (M_0<0.3) because it uses the Bernoulli equation. Moreover, it requires placing the outlet plane only in regions where the viscous effect can be neglected (it cannot be placed very close to the body) because it uses the simplified momentum equation (i.e., without the viscous term). Furthermore, this formulation performs well only with non-lifting bodies because it only takes into consideration the x-velocity component for drag prediction. Also, this profile drag expression is valid only if the static pressure is constant on the entire survey plane and equal to the upstream reference pressure (This plane is called Trefftz plane).

5.3 Jones method

Developed in 1936, the method proposed by Jones [7, 8] is a refinement of the Betz method in order to avoid the restriction of placing the survey plane coincident with the Trefftz plane. This is achieved by considering an isentropic flow between the survey plane and the Trefftz plane (Thus, the survey plane cannot be placed between two bodies). This hypothesis allows writing a simpler equation for the profile drag and allowing placing the survey plane everywhere downstream of a body:

$$D_p = 2\int_{S_{wake}} \left[\sqrt{P_t - P_s}\left(\sqrt{P_{t_0} - P_{s_0}} - \sqrt{P_t - P_{s_0}}\right)\right] dS \tag{43}$$

This formulation is subjected to the same other constraints as the Betz equation. Moreover, it is not valid when strong shockwaves are present: in those cases the local total pressure is smaller than the upstream static pressure, which leads to mathematical singularities (Negative root). Also, the assumption of isentropic flow along the wake is not accurate since entropy constantly increases along the wake (See exergy method).

5.4 Oswatitsch method

In 1956 Oswatitsch [9, 10] developed for the very first time an equation that relates field entropy variations with profile drag. It requires placing the survey plane in a position where the second-order terms can be neglected (i.e., far downstream of the body). Under such conditions the profile drag equation reduces to:

$$D_p = \frac{T_{s0}}{U_0} \int_{S_{wake}} \rho\, u\, \delta s\, dS \tag{44}$$

Where "δs" is the local specific entropy variation respect to the upstream condition.

5.5 Maskell method

In 1972, Maskell [11, 12, 13] proposed an improved far-field drag formulation for 3D lifting bodies. It provides a further decomposition of drag into profile and induced drag and he also proposed an equation to obtain lift.

5.5.1 Drag breakdown

A new drag breakdown is made by taking into account the induced drag (also called "vortex drag"). So far, the existing methods only take into consideration the axial velocity component, which leads only to a profile drag prediction. Maskell incorporates the transverse speed components which lead to the vortex drag formulation. This was achieved by reformulating the simplified total drag momentum conservation equation by considering a more general Bernoulli equation:

$$P_{t_0} = P_s + \frac{\rho}{2}\left(u^{*2} + v^2 + w^2\right) \tag{45}$$

This allows obtaining a new total drag expression:

$$D_{Tot} = \int_{S_{out}} \left[\left(P_{t_0} - P_t\right) + \frac{\rho}{2}\left(U_0{}^2 - u^2\right) + \frac{\rho}{2}\left(v^2 + w^2\right)\right] dS \tag{46}$$

Note that only the first term can be integrated in the wake area because outside of the wake the local total pressure equals the upstream total pressure. The other terms require a complete integration on the survey plane area. On the other hand, this equation can be decomposed into its profile and vortex drag. The profile drag itself is obtained by modifying this equation by using the same approach as Betz, allowing a wake integration:

$$D_p = \int_{S_{wake}} \left[\left(P_{t_0} - P_t\right) + \frac{\rho}{2}\left(u^* - u\right)\left(u^* + u - 2U_0\right)\right] dS \tag{47}$$

And the remaining term is the vortex drag:

$$D_v = \int_{S_{out}} \frac{\rho}{2}\left(v^2 + w^2\right) dS \tag{48}$$

This last equation was achieved by defining a blockage velocity in order to correct data because of the confinement effect (wind tunnel blockage).

$$u_{blockage} = \frac{1}{2\, S_{wt}} \int_{S_{wake}} \left(u^* - u\right) dS \tag{49}$$

Where "$u_{blockage}$" is the blockage velocity and "S_{wt}" is the wind tunnel cross section area. This correction is made by modifying the reference velocity:

$$\mathrm{u_{eff}} = \mathrm{U_0} + \mathrm{u_{blockage}} \tag{50}$$

Where "u_{eff}" is the effective upstream velocity (Note that u_{eff}=U_0 for free air condition because S_{wt}→∞).

Moreover, an alternative expression for the vortex drag can be obtained by introducing the scalar functions "ψ" (stream function) and "ϕ" (potential function) defined by Poisson equations as follows:

$$\frac{\partial^2\psi}{\partial y^2} + \frac{\partial^2\psi}{\partial z^2} = -\xi \tag{51}$$

$$\frac{\partial^2\phi}{\partial y^2} + \frac{\partial^2\phi}{\partial z^2} = \sigma \tag{52}$$

Where "ξ" is the axial vorticity and "σ" the in-plane flow source/sink, given by:

$$\xi = \frac{\partial w}{\partial y} - \frac{\partial v}{\partial z} \tag{53}$$

$$\sigma = \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = -\frac{\partial u}{\partial x} \tag{54}$$

By using these equations, the vortex drag can be re-expressed as:

$$D_v = \int_{S_{out}} \frac{\rho_0}{2} (\psi\,\xi - \Phi\,\sigma)\, dS \tag{55}$$

Note that only the first term can be integrated just in the wake region because the vorticity vanishes outside of the wake. The second term still requires a complete integration on the entire survey plane but when the survey plane is placed at least at 1 chord downstream, this term can be neglected [14]. Thus Epsilon allows the user to decide whether to include the "$\phi\sigma$" term or not. Moreover, Epsilon also allows the selection of the definition of "σ": it can be used the axial velocity derivative or its transverse derivative.

The stream and potential functions can be determined with an analytical approach, by using the Green functions as follows:

$$\begin{cases} \psi(y,z) = -\displaystyle\int_{S_{out}} G(y,z,y_w,z_w)\, \xi(y_w,z_w)\, dy_w\, dz_w & (56) \\ \Phi(y,z) = \displaystyle\int_{S_{out}} G(y,z,y_w,z_w)\, \sigma(y_w,z_w)\, dy_w\, dz_w & (57) \end{cases}$$

Where the Green functions varies depending on the type of survey data available:

$$\begin{cases} Free\ air - full\ model\ aircraft\colon G(y,z,y_w,z_w) = \dfrac{1}{4\pi}\, Log[(y-y_w)^2 + (z-z_w)^2] & (58) \\ Free\ air - half\ model\ aircraft\colon G(y,z,y_w,z_w) = \dfrac{1}{4\pi}\, \dfrac{Log[(y-y_w)^2 + (z-z_w)^2]_{model}}{Log[(y-y_w)^2 + (z-z_w)^2]_{mirror}} & (59) \end{cases}$$

The Green function method is not yet available for to bounded flows (wind tunnel data). This will be coded in the next version of Epsilon.

Finally, the induced velocity components in this plane are given by:

$$v_{induced} = \frac{\partial \psi}{\partial z} + \frac{\partial \Phi}{\partial y} \tag{60}$$

$$w_{induced} = -\frac{\partial \psi}{\partial y} + \frac{\partial \Phi}{\partial z} \tag{61}$$

The profile drag formulation is valid only for incompressible flows (like Betz), however the vortex drag is also valid for compressible regime. Moreover, the profile drag definition requires placing the survey plane in a zone of constant static pressure (Trefftz plane). However, the vortex drag does not impose any limitation to the plane position (when the complete equation is used along with a large survey plane).

5.5.2 Lift

Maskell also developed a formulation to calculate lift based on wake surveys. He started from the momentum conservation equation of lift. Then he introduces the general Bernoulli equation (which accounts for the traverse velocity components) and the lifting line relationship between vorticity and circulation "Γ":

$$\xi \, dS = -d\Gamma \tag{62}$$

The lift momentum equation then reduces to:

$$L = \int_{S_{wake}} \rho_0 \, U_0 \, y \, \xi \, dS + \int_{S_{out}} \rho_0 \, (U_0 - u) \, w \, dS \tag{63}$$

Where "S_{wake}" is the wake area at the survey plane. It can be seen that lift depends mainly on the axial vorticity created by the body (Usually the second term is neglected because it is very small compared to the first one). Epsilon allows calculating each term separately as well as the complete equation. The results are stored in 3 different variables, available in the Paraview's drop-down list (detailed in Chapter 4).

5.6 Van Der Vooren method

In 1990 Van Der Vooren [4, 15, 16] has proposed to improve the typical approach by making an insight into the physics of the flow. In fact, he decided to deeply analyze the velocity deficit "δu" measured on the survey plane:

$$\delta u = u - U_0 \tag{64}$$

This deficit is split into two components: a first one ($\delta\overline{u}$) associated with the viscous and wave phenomena (which is zero outside the wake), and other one (δu*) associated to an isentropic flow and the axial velocity variation due to the traverse flow field (a coupling), i.e.:

$$\delta u = \delta\overline{u} + \delta u^{*} \tag{65}$$

With:

$$\delta\overline{u} = U_0 \left(\sqrt{1 - \frac{2}{(\gamma-1)\,M^2}\left(e^{\frac{(\gamma-1)}{\gamma}\frac{\delta s}{R}} - 1\right) + \frac{2\,\delta h_t}{{U_0}^2}} - 1 \right) \tag{66}$$

Where "δh_t" is the specific total enthalpy variation respect to the upstream value and "M" the local Mach number. This approach allows writing the profile drag and the vortex drag as follows:

$$D_p = -\int_{S_{wake}} \rho\, u\, \delta\overline{u}\; dS \tag{67}$$

$$D_v = -\int_{S_{out}} \left[(P_s - P_{s_0}) + \rho\, u\; (u - U_0 - \delta\overline{u}) \right] dS \tag{68}$$

These equations are valid for incompressible and compressible regimes as well. It requires the existence of a Trefftz plane and that the flow remains isentropic and adiabatic between the survey plane and the Trefftz plane (So the survey plane cannot be placed between two bodies). Profile drag requires only a measurement on the wake region, nevertheless, the vortex drag requires data on the entire survey plane.

5.7 Giles method

In 1999 Giles [17] has developed a new formulation based on the simplified momentum conservation total drag equation, by using the definitions of stagnation enthalpy and entropy to replace the local axial velocity term of this equation, and by considering that the flow at the survey plane has no traverse component:

$$\vec{V} = u\,\vec{\imath} + 0\,\vec{\jmath} + 0\,\vec{k}$$

Then, after neglecting high-order terms he obtained:

$$D_p = \int_{S_{wake}} \left(P_{s_0}\,\frac{\delta s}{R} - \rho_0\,\delta h_t \right) dS \tag{69}$$

This expression is valid for compressible and incompressible flows as well, and it requires placing the survey plane on the Trefftz plane (i.e., far away from the body). It can be seen as an improvement of the Oswatitsch method because the "δh_t" term allows taking into account the power effects (i.e., thrust).

5.8 Kusunose method

In 2005 Kusunose [18] developed a new drag and lift formulations based on a small perturbations theory. He proposed to consider the perturbations of axial velocity, density and pressure:

$$\begin{cases} u = U_0 + \delta u \\ \rho = \rho_0 + \delta\rho \\ P_s = P_{s_0} + \delta P_s \end{cases} \tag{70}$$

5.8.1 Drag breakdown

After replacing the perturbation equations into the simplified total drag momentum equation and neglecting high-order terms, he obtained the expressions for the profile and vortex drag:

$$D_p = \int_{S_{wake}} \left[P_{s_0}\,\frac{\delta s}{R} - \rho_0\,\delta h_t - \frac{P_{s_0}}{2}\left(\frac{\delta s}{R}\right)^2 + \rho_0\,\delta h_t \left(\frac{\delta s}{R} - \frac{{M_0}^2}{2}\frac{\delta h_t}{{U_0}^2} \right) \right] dS \tag{71}$$

$$D_v = \int_{S_{wake}} \frac{\rho_0}{2} \psi\, \xi \; dS = \int_{S_{out}} \frac{\rho_0}{2} \left[(v^2 + w^2) - \left(1 - {M_0}^2\right) \delta u^2 \right] dS \qquad (72)$$

This is valid for compressible and incompressible regimes. Even though it is based on small perturbations theory, it is highly accurate even for high-lift and strong shockwave flow cases. Also, note that this drag formulation is an improvement of the Giles method since it allows placing the survey plane closer to the body.

5.8.2 Lift

Proceeding in a similar fashion but with the momentum conservation equation for lift, he obtains:

$$L = \int_{S_{wake}} \left[\rho_0\, U_0\, y\, \xi - \rho_0 {U_0}^2 \left(1 - {M_0}^2\right) \frac{w}{U_0} \frac{(u - U_0)}{U_0} + {M_0}^2\, \frac{\gamma\, P_{s_0}}{R} \frac{w}{U_0} \delta s - \rho_0 {M_0}^2 \frac{w}{U_0} \delta h_t \right] dS \qquad (73)$$

Note that this expression is an improvement of the Maskell method since it takes into account the compressibility and power effects.

5.9 Meheut method

In 2006 Meheut [4] developed a combined approach by using a small perturbations method (As Kusunose) and by decomposing the axial velocity deficit (As Van Der Vooren), which leads to the following equation:

$$\mathrm{D_p} = \frac{\rho_0\, {u_0}^2}{2} \int_{\mathrm{S_{wake}}} \left[-\frac{2}{\gamma\, {\mathrm{M_0}}^2} \Delta\mathrm{Pt} - \Delta\mathrm{Tt} + \left(1 - \frac{{\mathrm{M_0}}^2}{4}\right) \Delta\mathrm{Tt}^2 - \Delta\mathrm{Pt}\, \Delta\mathrm{Tt} - \left(1 - {\mathrm{M_0}}^2\right)\left(\Delta\bar{\mathrm{u}}^2 + 2\, \Delta\mathrm{u}^*\, \Delta\bar{\mathrm{u}}\right) \right] \mathrm{dS} \qquad (74)$$

With:

$$\begin{cases} \Delta P_t = \dfrac{P_t}{P_{t_0}} - 1 & (75) \\[2ex] \Delta T_t = \dfrac{T_t}{T_{t_0}} - 1 & (76) \\[2ex] \Delta u = \dfrac{u}{U_0} - 1 & (77) \\[2ex] \Delta u^* = \sqrt{1 - \dfrac{2}{(\gamma - 1)\, {M_0}^2} \left(\left(\dfrac{P_s}{P_{s_0}} \right)^{\frac{(\gamma - 1)}{\gamma}} - 1 \right) - \dfrac{v^2 + w^2}{{U_0}^2}} - 1 & (78) \\[2ex] \Delta\bar{u} = \Delta u - \Delta u^* & (79) \end{cases}$$

This is valid for compressible and incompressible regimes as well, and its related error is of third order (smaller than Kusunose's approach). In fact, it provides the most accurate profile drag value of all the far-field methods. It has also the advantage of reducing the measurement region to the wake. Nevertheless, its limitation is the survey plane position that cannot be placed closer than one chord to the body.

5.10 Profile drag breakdown

If there is no shockwave present in the flow field, the profile drag is equal to the viscous drag (the shockwave drag is zero). However, in the most general case, both components are present. Profile drag formulations already discussed do not allow separating the contribution of the wave drag and viscous drag. This can only be achieved by first calculating the profile drag. Then, a physical reasoning introduced by Kusunose [19] is used. It proposes that the profile drag survey can be divided in 2 regions (see Figure 6):

- **Pure shockwave region**: some streamlines traverse only the shockwave (outside of the boundary layer), so they have low or zero vorticity at the survey plane as well as low drag density value (because shockwave losses are always small compared to viscous losses). With this physical criterion in mind, it is possible to extract this component.
- **Viscous region**: this is the part of the wake where streamlines come from the boundary layer. If a shock was present, this shock has affected also the streamlines in the viscous region (Boundary layer thickness). This region is characterized by high drag density value and high vorticity. According to Kusunose, the viscous region can be further split into a pure viscous and a SBLI (Shockwave-boundary layer interaction) parts, as shown in Fig. 6. It has been proposed to first extract its wave component by doing a hypothesis: the strength of the shockwave is considered constant across the boundary layer on each side of the body. If a body has the upper and lower shocks with different strength, it is assumed that the wave drag varies linearly between both extremes of the viscous wake. The rest of the viscous is then attributed to the pure viscous component. Note that this approach is well suited for single bodies like wings, but it may lead to questionable decompositions for complex geometries.

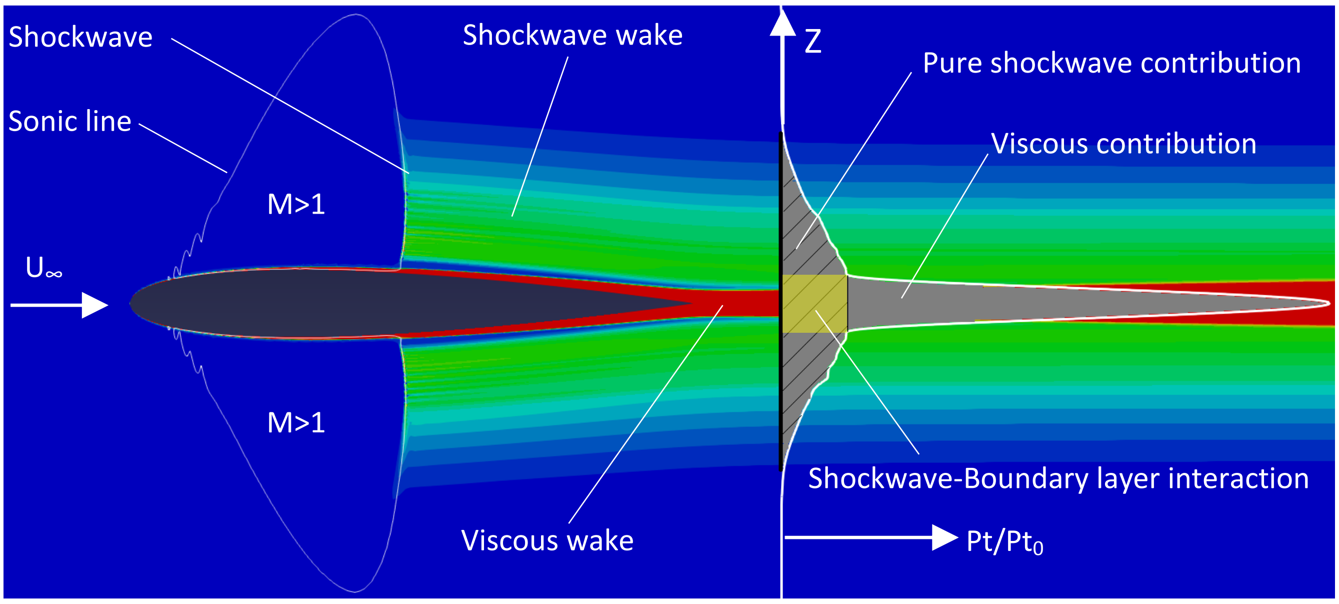


Figure 6: profile drag breakdown by Kusunose method

In Epsilon V2.0 the breakdown of the viscous region into its pure viscous and SBLI is not implemented because it has been shown to be impractical [20].

The far-field methods are well known for their limitation to analyze highly-coupled powered aircraft configurations. This is because those architectures do not allow a clean definition of drag and trust: the stream tube traversing the engine strongly interacts with the airframe. In order to provide an aerodynamic assessment for this kind of configurations, new methods have been developed (see the exergy section).

6 LAMB VECTOR METHODS

The Lamb vector method is an approach that generalizes the Kutta-Joukowsky theorem, which establishes a link between the lift force acting upon a body and the related circulation, i.e.:

$$L = \rho \, V \, \Gamma \tag{80}$$

Where "Γ" is the circulation, defined by:

$$\Gamma = \oint_l \boldsymbol{V} . d\boldsymbol{l} = \iint_S \boldsymbol{\omega} . d\boldsymbol{S} = \iint_S (\nabla \times \boldsymbol{V}) . d\boldsymbol{S} \tag{81}$$

With "$\boldsymbol{\omega}$" the vorticity vector (a measure of the local rotationality of the flow). Hence, the circulation is a measure of the total rotationality of the flow field. Then, in a 2D case, lift can be thought of as the domain integral of the cross product between the upstream flow vector and the local vorticity vector.

$$L = \rho \iint_S \boldsymbol{V} \times \boldsymbol{\omega} \, dS \tag{82}$$

This concept has been generalized in order to provide the total force acting upon a body either in 2D and 3D as follows:

6.1 Wu method

Wu was one the pioneers in the development of vorticity based formulations for the determination of the forces acting upon a body. In 2007 [21] he proposed using the Lamb vector to determine these forces:

$$\boldsymbol{l} = \boldsymbol{\omega} \times \boldsymbol{V} \tag{83}$$

Then, the lift, induced drag and profile drag are given by:

$$L = -\rho \iiint_v l_z dv \tag{84}$$

$$D_i = -\rho \iiint_v l_x dv \tag{85}$$

$$D_p = -\rho \int_{S_{out}} \boldsymbol{x} \times (\boldsymbol{n} \times \boldsymbol{l}) \, dS \tag{86}$$

This is a CFD-based formulation because it requires knowing the vorticity field inside the entire boundary layer (something impossible to achieve in a wind tunnel testing). However, profile drag can be applied in both, CFD and wind tunnel cases because the Lamb vector is limited to the wake region.

6.2 Mele method

More recently, Mele [22] has improved the Wu's formulation by including a compressibility correction term as follows, which leads to more accurate results:

$$L = -\hat{\boldsymbol{k}} \iiint_v \rho l_z + m_\rho \, dv \tag{87}$$

$$D_i = -\hat{\boldsymbol{\imath}} \iiint_v \rho l_x + m_\rho dv \tag{88}$$

$$D_p = -\rho \int_{S_{out}} \boldsymbol{x} \times (\boldsymbol{n} \times \boldsymbol{l})\, dS \tag{89}$$

With:

$$m_\rho = \boldsymbol{x} \times \left[\boldsymbol{\nabla}\rho \times \boldsymbol{\nabla}\left(\frac{V^2}{2}\right)\right] \tag{90}$$

Lamb Vector method is a powerful tool to understand physics from a very different point of view. Instead of regarding the surface tensions around the body surface (near-field) or the trace of the wake downstream of the body (far-field), it proposes to analyze the vorticity region surrounding the body (viscous region inside the boundary layer and wake). In fact, the vorticity contained inside the viscous region lead to the generation of forces that are perceived as lift, drag and sideforce.

7 EXERGY METHODS

This is a classical thermodynamic concept based on the 1st and 2nd laws of thermodynamics [23]. It decomposes the total energy of a system into two components: the exergy (the useful part of the energy) and the anergy (its useless part). The exergy concept states that any perturbation of the system (perturbation of speed, pressure, and so on) can be returned to its original (equilibrium) state by means of a reversible transformation (Carnot cycle). Hence, any perturbation has an inherent energetic potential (exergy) that can be converted to work by means of a reversible transformation. This can be expressed as follows:

$$\varepsilon = \delta h_t - T_\infty \delta s \tag{91}$$

Where "ε" is the exergy and " T_∞" is the equilibrium temperature. Its last term represents the anergy "$\mathcal{A}$":

$$\mathcal{A} = T_\infty \delta s_{created} \tag{92}$$

Several authors have used this concept to evaluate the external aerodynamic behavior of flight vehicles. One of most recent formulations is that proposed by Arntz at ONERA.

7.1 Arntz method

In 2014, Arntz [24] developed an exergy formulation well suited for the analysis of CFD simulations. It is given by:

$$\dot{\varepsilon}_{prop} + \dot{\varepsilon}_q = W\dot{\Gamma} + \dot{\varepsilon}_m + \dot{\varepsilon}_{th} + \dot{\mathcal{A}}_\Phi + \dot{\mathcal{A}}_{\nabla T} + \dot{\mathcal{A}}_w \tag{93}$$

In Epsilon V2.0 only unpowered adiabatic cases have been considered (i.e., $\dot{\varepsilon}_{prop}$=0 and $\dot{\varepsilon}_q$=0) and then this expression can be rewritten to obtain the exergy-based drag as follows:

$$D_\varepsilon = \dot{\varepsilon}_m + \dot{\varepsilon}_{th} + \dot{\mathcal{A}}_\Phi + \dot{\mathcal{A}}_{\nabla T} + \dot{\mathcal{A}}_w \tag{94}$$

This drag value is absolutely comparable with the drag prediction methods discussed before and it also allows a more comprehensive drag breakdown [20]. In this expression, "$\dot{\varepsilon}_m$" is the outflow of mechanical exergy across the survey plane, "$\dot{\varepsilon}_{th}$" is the thermal exergy outflow, "$\dot{\mathcal{A}}_{\Phi}$" the viscous anergy created inside the control volume, "$\dot{\mathcal{A}}_{\nabla T}$" the thermal anergy and « $\dot{\mathcal{A}}_w$ » the shockwave anergy. Each term represent an equation itself as indicated as follows:

$$\dot{\varepsilon}_m = \int_{Sout}\frac{1}{2}\rho\,\delta u^2(\vec{V}.\vec{n})dS + \int_{Sout}\frac{1}{2}\rho(v^2+w^2)(\vec{V}.\vec{n})dS + \int_{Sout}(P_s - P_{s0})[(\vec{V}-\vec{V}_0).\vec{n}]dS \tag{95}$$

$$(\dot{\varepsilon}_m = \dot{E}_u + \dot{E}_v + \dot{E}_p) \tag{96}$$

$$\dot{\varepsilon}_{th} = \int_{Sout}\rho\,\delta e\,(\vec{V}.\vec{n})\,dS + \int_{Sout}P_{s0}(\vec{V}.\vec{n})\,dS - Ts_0\int_{Sout}\rho\,\delta s\,(\vec{V}.\vec{n})\,dS \tag{97}$$

$$(\dot{\varepsilon}_{th} = \dot{E}_{th} + \dot{E}_w + \dot{\mathcal{A}}) \tag{98}$$

$$\dot{\mathcal{A}}_{\Phi} = \int_v \frac{T_{s0}}{T_s}\,\Phi_{eff}\,dv \tag{99}$$

$$\dot{\mathcal{A}}_{\nabla T} = \int_v \frac{T_{s0}}{{T_s}^2}\,k_{eff}\,(\nabla T)^2\,dv \tag{100}$$

$$\dot{\mathcal{A}}_w = T_{s0}\int_{S_{wave}}(\rho\,\delta s\,\vec{V}).\vec{n}\ \ dS_{wave} \tag{101}$$

Where"δs" is the specific total variation of entropy respect to the upstream condition (equilibrium condition), "k_{eff}" is the effective thermal conductivity and "Φ_{eff}" is the effective dissipation. These two terms are given by:

$$\kappa_{eff} = c_p * \left(\left(\frac{\mu}{0.7}\right) + \left(\frac{\mu_{turb}}{Pr}\right)\right) \tag{102}$$

$$\Phi = (\bar{\bar{\tau}}.\nabla).\vec{V} \tag{103}$$

$$\mu_{eff} = \mu + \mu_t \tag{104}$$

$$\Phi_{eff} = \mu_{eff}\left\{\left[2\left(\frac{\partial u}{\partial x}\right)^2 + 2\left(\frac{\partial v}{\partial y}\right)^2 + 2\left(\frac{\partial w}{\partial z}\right)^2 + \left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial z}+\frac{\partial w}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}\right)^2\right] - \frac{2}{3}\mu\left[\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right]^2\right\} \tag{105}$$

Where "μ" is the laminar viscosity, "μ_t" is the turbulent viscosity and "Pr" the Prandtl number.

One of the advantages of the exergy approach is that it takes into account the viscous term of the momentum equation during its formulation developing. So, it allows placing the survey plane everywhere. All the far-field formulations discussed before cannot accept a survey plane placed very close to the body, simply because the stress tensor influence has been ignored. Another advantage is its validity for both, incompressible and compressible flow. Also, this method provides a **very accurate drag prediction**.

7.1.1 Wave anergy

An alternative expression for the wave anergy can be obtained by using the Gauss theorem, in order to convert the surface integral into a volume integral:

$$\dot{\mathcal{A}}_w = \int_{Sw_{vol}}\nabla.(T_{s0}\,\rho\,\delta s\,\vec{V})\ \ dSw_{vol} \tag{106}$$

Where "Sw_{vol}" is the shockwave volume.

7.1.2 Total anergy

The total anergy is given by the sum of the anergy terms:

$$\Sigma\dot{\mathcal{A}} = \dot{\mathcal{A}}_{\Phi} + \dot{\mathcal{A}}_{\nabla T} + \dot{\mathcal{A}}_{w} \tag{107}$$

However, it was reported by Arntz that this sum does not give the real total anergy when the equations are implemented on a CFD post-treatment code because of numerical errors (due to the integration of high-order derivatives). Thus, an accurate way to obtain the total anergy "$\dot{\mathcal{A}}$" is by using the following equation:

$$\dot{\mathcal{A}} = T_{s_0} \int_{S_{out}} \rho\, \delta s\, (\vec{V}.\vec{n})\, dS \tag{108}$$

This requires a surface integration on the survey plane only, thus it gives a highly accurate total anergy value.

7.1.3 Thermal exergy

An alternative expression for $\dot{\varepsilon}_{th}$ is given by [24]:

$$\dot{\varepsilon}_{th} = \dot{\varepsilon}_{th_{temperature}} + \dot{\varepsilon}_{th_{pressure}} \tag{109}$$

With:

$$\dot{\varepsilon}_{th_{temperature}} = \int_{S_{out}} \rho\, c_v\, T_s\, (\vec{V}.\vec{n}) \left[1 - \frac{T_{s_0}}{T_s}\, ln\left(\frac{T_s}{T_{s_0}}\right)\right] dS - \int_{S_{out}} \rho\, c_v\, T_{s_0}\, (\vec{V}.\vec{n})\, dS \tag{110}$$

$$\dot{\varepsilon}_{th_{pressure}} = \int_{S_{out}} P_{s_0} \left[1 - \frac{\rho}{\rho_0}\, ln\left(\frac{\rho_0}{\rho}\right)\right] (\vec{V}.\vec{n})\, dS - \int_{S_{out}} \rho\, R\, T_{s_0}\, (\vec{V}.\vec{n})\, dS \tag{111}$$

This is the actual formulation used in Epsilon because it only requires data on the survey plane.

7.1.4 Exergy flow field interpretation

The physical interpretation of the exergy parameters is not provided in this manual, but it can be found in detail in [25, 26] as well as in the Epsilon website:

https://websites.isae-supaero.fr/epsilon-exergy-analysis-tool/exergy-crash-course-520/

7.2 Aguirre method

Since 2018, ISAE is working on the improvement of the original exergy method proposed by Arntz. The first stage was the modification of the exergy method in order to adapt it for wind tunnel testing. The new resulting formulations and methods are coded into Epsilon as described below:

7.2.1 Wave anergy (volume formulation)

The volumetric formulation of the wave anergy (Eq. 101) poses the challenge to properly limit the integral region. The original approach proposed by Arntz was to use the Lovely and Haimes shockwave detector (see next section) to identify the shockwave region. An alternative approach [31] is to use the Q-criterion (see next section) to take away the viscous region from the volume integral. This simplifies the integration task as it will be explained in Chapters 4 and 5.

7.2.2 Wave anergy (wake surface formulation)

The wave anergy extraction method proposed by Arntz requires performing a volume integral or a surface integral around the shockwave volume. This approach is not feasible for wind tunnel testing, thus an adaptation of the Kusunose drag breakdown was developed. The new method [20, 31] analyzes the total anergy distribution on a survey plane as shown in Figure 7.

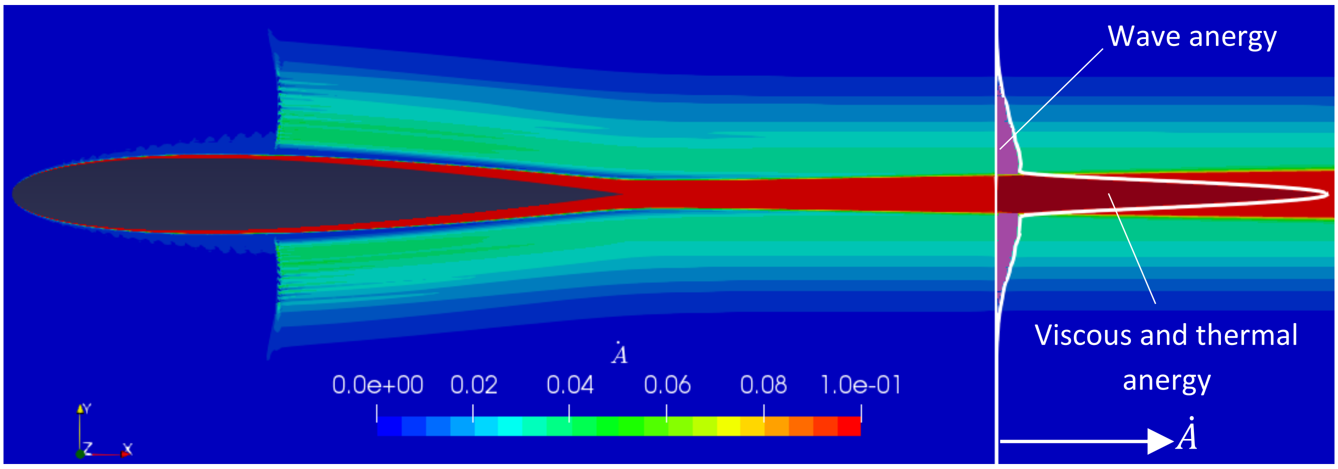


Figure 7: total anergy breakdown

Then, the part of its distribution lying outside the viscous wake corresponds to the wave anergy losses. This allows the extraction of the wave anergy from the total anergy distribution. The remaining part of the distribution is a mixing of thermal and viscous anergy created inside the boundary layer and the wake.

This total anergy breakdown method is included in the same Epsilon module as the Kusunose breakdown method (it will be detailed in Chapter 4).

7.2.3 Redefinition of the mechanical and thermal exergy (infinite surface formulation)

Some inconsistencies have been identified in the original exergy formulation as discussed in [29]. This led to a redefinition of the mechanical and thermal exergies through the reformulation of the boundary pressure work rate:

$$\dot{E}_{p\,new} = \dot{E}_{p\,Arntz} + \dot{\varepsilon}_{th_{pressure}} \tag{112}$$

$$\dot{\varepsilon}_{th\,new} = \dot{\varepsilon}_{th_{temperature}} \tag{113}$$

Hereafter, the Arntz equations with the modifications indicated here will be called "redefined Arntz equations". These are still infinite surface formulations (as the Artnz equations), hence, only suited for CFD analysis.

7.2.4 Profile drag

Recently, the redefined Arntz's equations have been used as the starting point to develop a new formulation suited for measuring exergy in a wind tunnel testing (WTT) [27-31]. The objective of the wind-tunnel method (also called "WTT-method") is to reduce the infinite surface integral required by the Arntz's method to the wake region. This is achieved by using the velocity decomposition technique [28, 30] that splits a velocity component into its isentropic $(\,)^*$ and non-isentropic $\overline{(\,)}$ parts:

$$\mathrm{u} = u^* + \bar{u} \tag{114}$$

$$\mathrm{v} = v^* + \bar{v} \tag{115}$$

$$\mathrm{w} = w^* + \bar{w} \tag{116}$$

The non-isentropic velocity is the part of the velocity field that is related to the losses (viscous or wave losses), whereas the isentropic velocity gives the equivalent of a potential flow field (where no losses occur). The magnitude of this isentropic velocity is given by:

$$\mathrm{V}^* = \mathrm{u}_0 \sqrt{1 - \frac{2}{(\gamma-1)\,\mathrm{M}_0{}^2}\left[\left(\frac{\mathrm{P_t}}{\mathrm{P_{t_0}}}\right)^{\frac{(\gamma-1)}{\gamma}} * \zeta - 1\right]} \tag{117}$$

With:

$$\zeta = 1 + \frac{\gamma-1}{2}\mathrm{M}_0{}^2\left(1 - \frac{\mathrm{V}^2}{\mathrm{u}_0{}^2}\frac{\mathrm{T_{t_0}}}{\mathrm{T_t}}\right) \tag{118}$$

In the same fashion, the temperature and density can be also splitted into its isentropic and non-isentropic components as follows:

$$\mathrm{T_s} = T_s{}^* + \bar{T_s} \tag{119}$$

$$\rho = \rho^* + \bar{\rho} \tag{120}$$

With:

$$\mathrm{T_s}^* = \mathrm{T_{s_0}}\left[1 + \frac{\gamma-1}{2}\,\mathrm{M}_0{}^2\left(1 - \frac{\mathrm{V}^{*2}}{\mathrm{u}_0{}^2}\right)\right] \tag{121}$$

$$\rho^* = \rho_0\left[1 + \frac{\gamma-1}{2}\,M_0{}^2\left(1 - \frac{V^{*2}}{u_0{}^2}\right)\right]^{\frac{1}{\gamma-1}} \tag{122}$$

Then, by replacing the "u, v, w, ρ and T_s" terms of the redefined Arntz equations by their isentropic version $(\,)^*$, the isentropic drag and exergies can be obtained (See Ref. [30] for further details). This finally leads to the determination of the non-isentropic drag and exergies as follows:

$$\overline{\mathrm{C}_{\mathrm{D}_\varepsilon}} = \mathrm{C}_{\mathrm{D}_\varepsilon} - \mathrm{C}_{\mathrm{D}_\varepsilon}^* \tag{123}$$

$$\overline{\dot{\varepsilon}_\mathrm{m}} = \dot{\varepsilon}_\mathrm{m} - \dot{\varepsilon}_\mathrm{m}^* \tag{124}$$

$$\overline{\dot{\mathrm{E}}_\mathrm{u}} = \dot{\mathrm{E}}_\mathrm{u} - \dot{\mathrm{E}}_\mathrm{u}^* \tag{125}$$

$$\overline{\dot{\mathrm{E}}_\mathrm{v}} = \dot{\mathrm{E}}_\mathrm{v} - \dot{\mathrm{E}}_\mathrm{v}^* \tag{126}$$

$$\overline{\dot{\mathrm{E}}_\mathrm{p}} = \dot{\mathrm{E}}_\mathrm{p} - \dot{\mathrm{E}}_\mathrm{p}^* \tag{127}$$

$$\overline{\dot{\varepsilon}_\mathrm{th}} = \dot{\varepsilon}_\mathrm{th} - \dot{\varepsilon}_\mathrm{th}^* \tag{128}$$

The idea behind this decomposition method is that the **non-isentropic** components are reduced to the wake region and physically related to the **profile drag generation** (viscous and wave losses). The **isentropic** components are equivalent to a **potential flow** (where no losses occur), thus, they do not intervene in the generation of drag. Hence, the determination of drag by the WTT-method only requires a wake integral of the non-isentropic terms (whereas the CFD-based method requires the knowledge of aerodynamic parameters on an infinite survey plane).

7.2.5 Profile drag correction

The WTT method presented before has proven to perform well for survey planes placed at least at 0.5 chords downstream of the trailing edge of a body [30]. If the survey plane gets closer to the body, a correction accounting for the isentropic coupling must be introduced [28] as follows:

$$\overline{C_{D}}_{\varepsilon\,corrected} = \int_{S_w} \left(\overline{C_{D}}_{\varepsilon} + \left| C_{D\,\varepsilon}^{*} \right|_{wake} \right) dS \tag{129}$$

$$\overline{\dot{\varepsilon}}_{m\,corrected} = \int_{S_w} (\overline{\dot{\varepsilon}_m} + |\dot{\varepsilon}_m^{*}|_{wake}) dS \tag{130}$$

7.2.6 Vortex exergy (induced drag)

The vortex exergy (transverse exergy) formulation proposed by Arntz requires performing an integration on the entire survey plane, which is not suited for wind tunnel testing. Thus a new formulation was provided in order to limit the integration to the wake region only [27] by combining the Maskell method with the velocity decomposition method as follows:

$$\dot{E}_v = \int_{Swake} \frac{1}{2} \rho \, (\psi\xi - \Phi\sigma) \, u^{*} \, dS \tag{131}$$

$$u^{*} = U_0 \sqrt{1 - \frac{2}{(\gamma-1){M_0}^2} \left[\left(\frac{P_t}{P_{t_0}} \right)^{\frac{\gamma-1}{\gamma}} \mathfrak{I} - 1 \right] - \frac{(\psi\xi - \Phi\sigma)}{{U_0}^2}} \tag{132}$$

$$\mathfrak{I} = 1 + \frac{\gamma-1}{2} {M_0}^2 \left(1 - \frac{u^2 + \psi\xi - \Phi\sigma}{{U_0}^2} * \frac{T_{t_0}}{T_t} \right) \tag{133}$$

This formulation has proven to provide **more accurate induced drag values** than the Maskell method.

8 CLASSICAL ANALYSIS METHODS

Epsilon also provides some useful parameters to perform classical aerodynamic analysis:

8.1 Shockwave detection

The shockwave surface can be detected by using the Lovely and Haimes criterion [32]:

$$ShockDetector = \frac{\vec{V}.\overrightarrow{\nabla Ps}}{a\,|\overrightarrow{\nabla Ps}|} \geq 0.95 \tag{134}$$

Where "$\overrightarrow{\nabla Ps}$" is the pressure gradient.

8.2 Vortex detection

A vortex can be detected by several methods. In Epsilon V2.0, there are two of the most used methods:

8.2.1 Q-criterion

The Q-criterion is based on the magnitudes of the vorticity "ω" and strain rate "S" to identify the vortex region as follows:

$$Q = \frac{1}{2}\left(\omega^2 - S^2\right) \geq 0 \tag{135}$$

8.2.2 Lambda 2 criterion

According to this criterion, a vortex exists in regions where is negative the second eigenvalue (λ_2) of the following equation, where "**J**" is the velocity gradient:

$$\boldsymbol{S}^2 + (\boldsymbol{J} - \boldsymbol{S})^2 \tag{136}$$

8.3 Wake detection

8.3.1 Total pressure ratio

The wake region can be detected by several methods. The most used is the total pressure ratio:

$$\frac{P_t}{P_{t_0}} < 1 \tag{137}$$

9 NONDIMENSIONALIZATION

The lift and drag equations previously discussed are dimensionalized. It is much more practical to work with lift and drag coefficients (C_L and C_D respectively), defined as follows:

$$C_L = \frac{L}{\frac{1}{2}\rho_0\, {U_0}^2\, S_{ref}} \tag{138}$$

$$C_y = \frac{Fy}{\frac{1}{2}\rho_0\, {U_0}^2\, S_{ref}} \tag{139}$$

$$C_D = \frac{D}{\frac{1}{2}\rho_0\, {U_0}^2\, S_{ref}} \tag{140}$$

Where "S_{ref}" is a reference surface (the airfoil chord for 2D cases or wing plan surface for 3D cases). In a similar fashion, it is defined the anergy and exergy coefficients (expression valid also for its components) and the exergy-based drag coefficient:

$$C_{\dot{\mathcal{A}}} = \frac{\dot{\mathcal{A}}}{\frac{1}{2}\rho_0\, {U_0}^3\, S_{ref}} \tag{141}$$

$$C_{\dot{\varepsilon}} = \frac{\dot{\varepsilon}}{\frac{1}{2}\rho_0\, {U_0}^3\, S_{ref}} \tag{142}$$

$$C_{D_\varepsilon} = \frac{\dot{\varepsilon}_m + \dot{\varepsilon}_{th} + \dot{\mathcal{A}}_\Phi + \dot{\mathcal{A}}_{\nabla T} + \dot{\mathcal{A}}_w}{\frac{1}{2}\rho_0\, {U_0}^3\, S_{ref}} \tag{143}$$

The denominator used to non-dimensionalize the previous equations will be available as a separate variable:

$$AdimF = \frac{1}{2}\rho_0\, {U_0}^2\, S_{ref} \tag{144}$$

$$AdimP = \frac{1}{2}\rho_0\, {U_0}^3\, S_{ref} \tag{146}$$

Moreover, the pressure will be non-dimensionalized in order to obtain the pressure coefficient:

$$C_\mathrm{p} = \frac{P_\mathrm{s} - P_{\mathrm{s}_0}}{\frac{1}{2}\rho_0\, {U_0}^3} \tag{146}$$

CHAPTER 2: THE ARCHITECTURE OF EPSILON

This chapter details the architecture of Epsilon and the functionalities of Paraview. The programmable sources and filters with a custom-made GUI are explained here. Chapters 3 and 4 will explain in detail how to use those tools in practice.

It is advised to have some background knowledge of Paraview before starting using Epsilon. A recommended lecture is available in references [33,34]. Hereafter we will consider that the Epsilon user is already familiarized with the CFD posttreatment in Paraview.

1 ARCHITECTURE

Epsilon is set of filters and plugins created for the analysis of 2D and 3D steady state CFD simulations and experimental data (WTT). Its architecture is very simple as it can be seen in Figure 8:

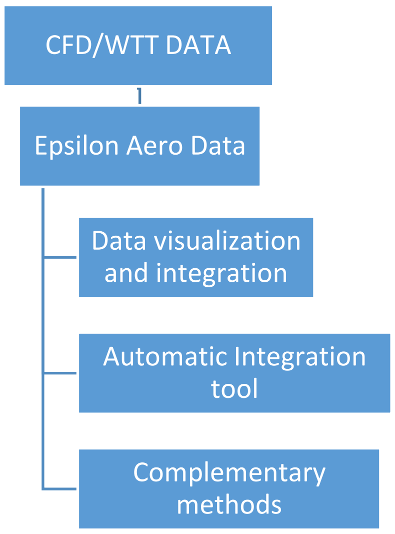

CFD/WTT DATA
Epsilon Aero Data
Data visualization and integration
Automatic Integration tool
Complementary methods

Figure 8: conceptual diagram of the code

It consists of 5 stages. Firstly, the user must load its CFD or experimental data into Paraview. Then all the near-field, far-field, Lamb vector and exergy equations are implemented in a Python code called "Epsilon_Aero_Data". This allows the visualization and analysis of these aerodynamics parameters on the entire field by using the standard Paraview tools. Moreover, a special integration tool was included in order to study several survey plane positions automatically. Finally, some particular near-field, far-field and exergetic methods are computed by separated modules due to the related complexity.

2 INPUT DATA

2.1 Data formats

Epsilon supports any type of aerodynamic data either computational or experimental data, **provided that these formats are readable by Paraview**.

2.1.1 CFD data

Several commercial CFD solvers are supported by Epsilon by default (non-exhaustive list):

• Fluent
• STAR CCM+
• elsA
• Open Foam

CFD data from other solvers can be easily read by writing a small Python script (see "Readers files" section). CFD data is simply loaded into Paraview and then it is possible to apply the Epsilon custom filters.

2.1.2 Wind Tunnel data

Experimental data can also be analyzed with Epsilon: PIV (2D2C, stereo and 3D) as well as directional probes (five-hole probe and seven-hole probe). This kind of data is loaded indirectly by using a custom source called "Epsilon Wind Tunnel". It is a Python programmable source that reads the measurement CSV file and creates a delaunay2D mesh plane containing all the measured variables (pressures and velocity components). The CSV file must respect the following **headers designation**: X, Y, Z, Vx, Vy, Vz, Pt, Ps (Pt and Ps are present only in multi-hole probe data). Moreover, the **XYZ-coordinates** of the CSV measurement data **must be coincident with the aerodynamic reference frame** (The user must arrange the headers if needed prior to open it with Epsilon). Details about this source will be covered by a specific video tutorial in the Epsilon website.

IMPORTANT: please, keep in mind that Epsilon Wind Tunnel is still in development/validation phase. Moreover, the actual version of Epsilon (V2.0) do not calculate the static pressure field from the PIV data. Thus, the user must ignore any Epsilon Aero Data parameter requiring Ps in its formulation (see below).

2.2 Readers files

These are auxiliary Python files executed by the main Epsilon python code (Epsilon Aero Data). The readers load the incoming CFD or experimental data arrays and calculate some missing parameters. Its output always contains the following variables: POINTS, Vx, Vy, Vz, Ps, Ts, Rho, MU_LAM, MU_TURB, MU_EFF, Keff. These are the only required variables to properly execute Epsilon Aero Data. Hence, the input CFD data must provide those variables or at least it should be possible to compute a missing variable from the existing data.

Epsilon V2.0 is released with readers of some CFD softwares. Other CFD codes may be used with Epsilon, provided that the user develops an appropriate reader file by replicating the examples already provided.

3 EPSILON AERO DATA

At this point we are ready to use the CFD/WTT data loaded into Paraview in order to implement the equations presented in the previous chapter. This will be made by using the so called "Programmable Filters".

3.1 Paraview's programmable filters

In Paraview, the programmable filters are a way to process data inside the Paraview Server by a Python code provided by the user, and then to display it on the Paraview Client. The user can also create a custom GUI to display and control the input parameters of the Python code. This is made by a Server Manager Configuration file (XML file).

3.1.1 The Python code

The python code can be either coded in a dedicated window on Paraview, or even better, executed from an external Python file. This Python file contains the code that reads the existing aerodynamic parameters at each point of the mesh in Paraview. Then it determines the new near-field, far-field, Lamb vector and exergy aerodynamic parameters at each point. This data is stored in several new arrays that are sent to the Paraview Server and displayed in the Paraview client.

The Python file is called "**Epsilon_Aero_Data.py**" and is fully commented to help the understanding of the code. By the way, it is worth to note that this file can be easily modified by users in order to implement some improvements (to add new formulations for example). For normal purposes, this file is ready to use and should not be modified.

3.1.2 The XML file

In order to create a custom-made GUI showing all the inputs of the Python code to play with, a XML file must be used. This file is called "**Epsilon_Aero_Data.xml**" and contains information about the type of filter to be created (programmable filter), the file path to the Python code to be executed, the input parameters, its default values and the way they are displayed in the GUI.

3.1.3 Paraview's plugin

When this XML file is loaded into Paraview, it creates a new filter icon called "Epsilon Aero Data". If the user clicks on it, this will execute the XML file which creates the GUI and executes the associated Python file. The resulting output arrays are displayed on the screen. So, for the final user, an easy to use aerodynamic calculator is available (it provides the near-field, far-field, Lamb vector and exergy parameters). An advanced user can also improve its capabilities by slightly modifying the XML or Python files.

In a general way, whenever a custom filter is needed, a Python code and a XML file with the same name are created and loaded into Paraview. This approach has been used extensively on Epsilon.

3.2 Epsilon Aero Data Code

This is the Python file mentioned above and is the brain of the entire Epsilon code. It is here where most of the aerodynamic equations are introduced. It is basically composed of 4 stages:

First of all, it reads the existing CFD/WTT data already loaded into Paraview. This is achieved by calling the reader file (Python codes) that suits the input format. This allows converting the input data arrays from the Paraview format into a Python readable format.

The second step is to rotate the vector data. As it was stated in Chapter 1, Epsilon uses the aerodynamic reference frame, which is not always coincident with the mesh reference frame. Hence, a transformation of some data arrays is required (i.e., the vectors and tensors) based on the angle of attack and sideslip angles.

The third step is to implement the formulations. **Since all formulations are integral, Epsilon Aero Data just implements the integrands of those equations**. The integration procedure is left to be applied later. Thus, Epsilon Aero Data will evaluate the integrand of the equations from Chapter 1 at each point of the mesh. Some equations have integrands composed by several terms. In some of those cases, an evaluation is made for each term at every mesh point. This decomposition is particularly useful while studying the influence of its terms in a given result.

The fourth and final step is to send all these new data arrays to the Paraview server in order to be shown on the Paraview Client. At the end of this process, Epsilon Aero Data creates several data arrays that will be available for the user. The list of arrays names and its corresponding equations are detailed in the Appendix 1.

4 DATA VISUALIZATION AND INTEGRATION

Once the Epsilon Aero Data filter has been applied, the user is ready to visualize and analyze the aerodynamic parameters (i.e., near-field, far-field, Lamb vector and exergy).

4.1 Epsilon Axes

Since the mesh axes are not always coincident with the aerodynamic axes, it may be confusing for the user to interpret the aerodynamic data (evaluated in the wind axes) by using the mesh reference axes. That's why a Programmable filter is provided ("Epsilon Axes".), which allows drawing the aerodynamic axes on the screen as shown in Figure 9. This is a practical tool for a beginner user (Once the user gets familiarized with Epsilon, it is no longer needed).

4.2 Field visualization

At this stage, the user can profit from the standard Paraview filters in order to visualize data as usual. The Epsilon Aero Data output do not allow to detect automatically the body geometry, so this surface must be extracted first (The step-by-step procedure is explained in Chapters 4 and 5). Once made, the surface can be colored by the typical parameters (i.e., pressure coefficient) but also by the new aerodynamic parameters. This color data visualization can also be made on slices across the data volume in order to see the field parameters. In particular, it is interesting to **use a plane normal to the upstream flow direction** because all the formulations consider such a survey plane. In fact, when the user executes Epsilon Aero Data, it displays in the Paraview messages window the components of the normal surface vector. This information must be used to set up the proper survey plane inclination.

Epsilon Aero Data also provides a velocity vector field, thus the user can directly employ the **SurfaceLIC** filter in order to analyze the surface flow pattern (Figure 9). This can be applied to the body surface and the slice plane as well. Other important tools are the streamlines tracer and iso-surface plots. They can be used as usual to analyze the new aerodynamic parameters. Details on these tools are available in the Paraview manual.

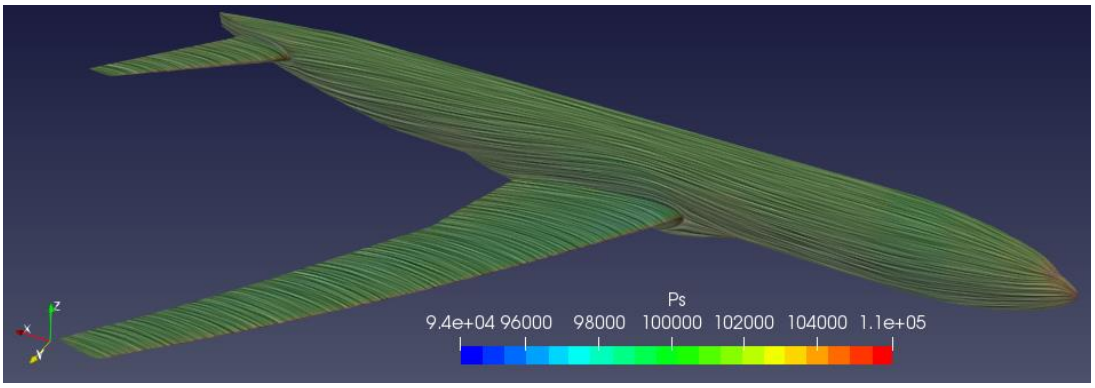


Figure 9: data visualization example

4.3 Integration of the equations

Epsilon Aero Data evaluates the integrand of the aerodynamic analysis methods at every mesh point. This allows visualizing the **density of drag**, density of lift, and so on. However the major goal is to obtain drag, lift, exergy and anergy values already integrated. This is achieved by following two approaches:

4.3.1 Surface integrals

The majority of the equations presented in the Chapter 1 require to be integrated on the survey plane (and sometimes this integral is reduced to the wake region of this survey plane). For those cases, it is just necessary to place a slice plane normal to the upstream flow direction, and to set the desired axial position as seen on Figure 10-left. This will allow obtaining the values of the equations integrands at the entire slice plane. Thus, it just requires applying the standard "Integrate variables" filter to the plane data. Its output is a table with one row and one column for each parameter. In each cell it is shown the integrated value of each parameter.

Important: Note that only the parameters requiring a surface integral must be taken into account in this output table. For example, the integral of the "BETZ_profile_drag" density gives the profile drag value. However, the integrals of some anergy parameters (demanding a volume integration in its formulae) are **meaningless**. Thus, the user MUST always be conscious of the equations discussed in Chapter 1.

4.3.2 Volume integrals

Some anergy and Lamb vector equations require integrating data inside the control volume (Figure 10-right). In this case the "Integrate variables" filter can be applied directly to the Epsilon Aero Data output.

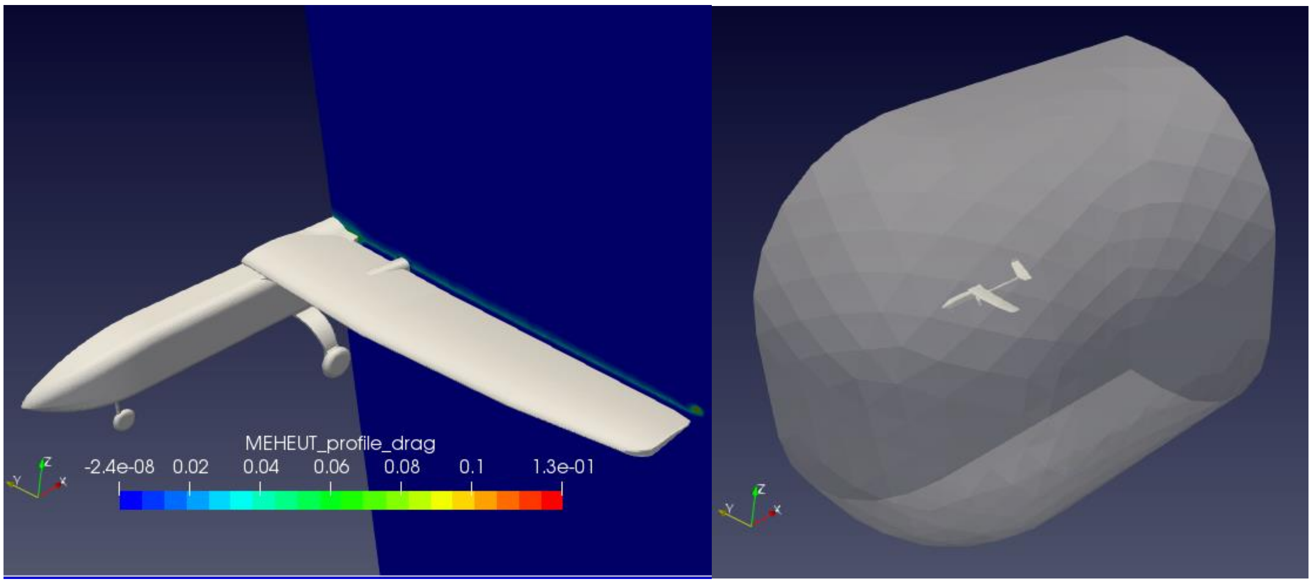


Figure 10: surface vs volume integrals

As before, the output table with the integrated data must be analyzed carefully. The user MUST only observe the anergy and Lamb vector terms requiring volume integration. All the remaining volume integrated parameters are **meaningless**. An intermediate solution to the problem is to apply the "Pass Arrays" filter to Epsilon Aero Data in order to retain only the formulations requiring a volume integral and then to integrate the output of this "Pass Arrays" filter. However, **the simplest solution is to use Epsilon Integration Tool** since all these interpretation issues are solved.

5 EPSILON INTEGRATION TOOL

5.1 The conceptual approach

The integration technique already presented allows the user to know the integrated parameters for a certain survey plane position. For most of applications this is enough. However, sometimes it is useful to sweep this survey plane and to integrate the parameter's values at each plane position in order to evaluate aerodynamic parameters dependency with plane position (For example, to evaluate the profile drag vs the plane position downstream of a body, or to obtain the lift and drag distribution along the wing span). This is achieved with Epsilon Integration Tool.

5.2 Line distribution

This tool option allows animating the survey plane position: the idea is to place the survey plane at several positions and to store the integrated values at each position in a **vtk polyline** output. This vtk polyline is in fact formed by several points as shown in Figure 11. These points are used to define the survey plane positions (this plane is always placed normal to the upstream flow direction). Then, by following an iterative approach, the plane is placed at each point and its data is integrated and stored as point data.

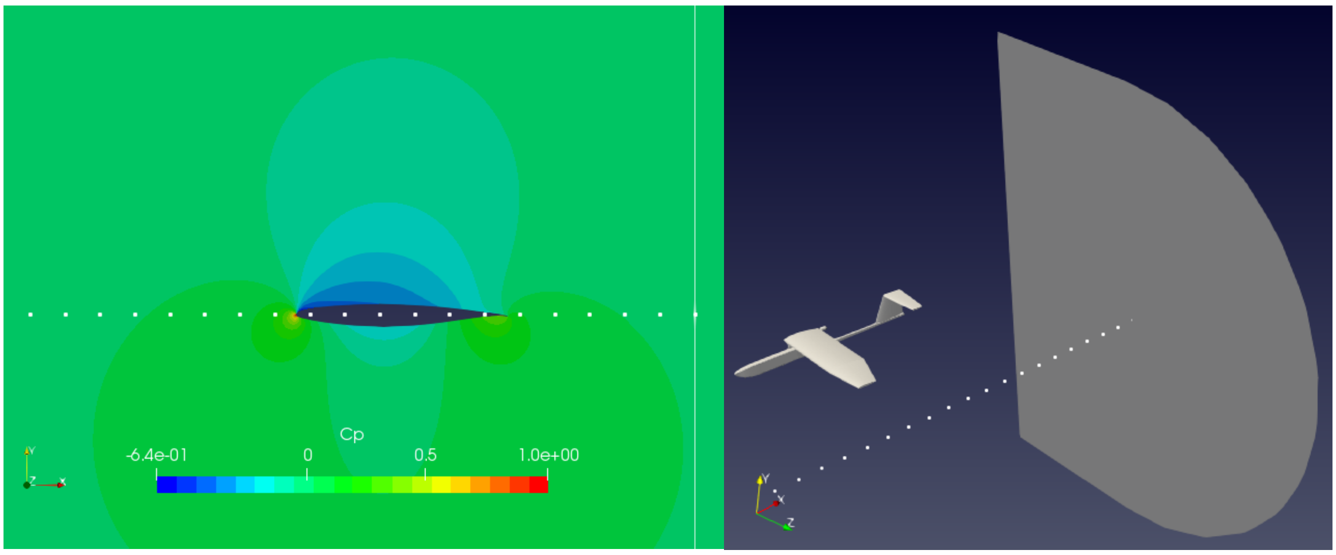


Figure 11: integrated x-, y-, z-coordinates for several 2D and 3D survey planes positions

The resulting polyline data can then be visualized in several ways. The simplest way is to WarpByScalar the integrated values as shown in Fig. 12:

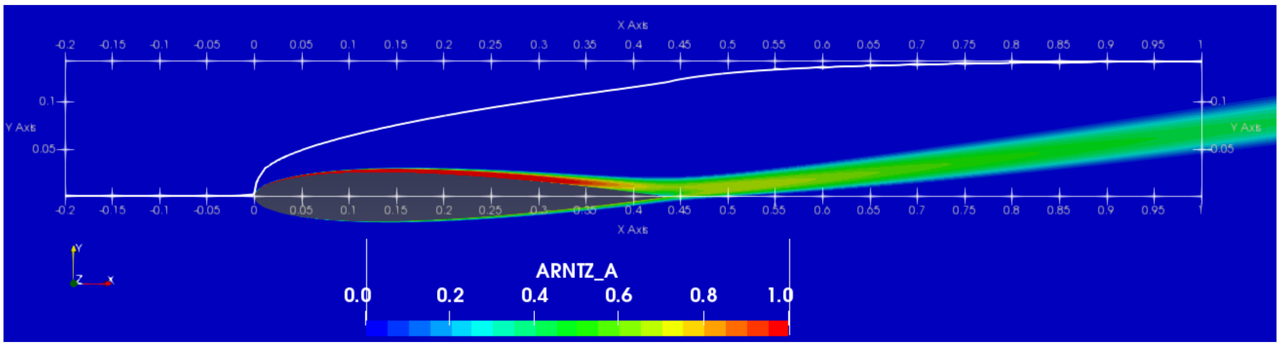


Figure 12: vtkPolyline data warped by ARNTZ_A along the mesh Y-axis

5.3 3D to 2D mapping

One of the major functionalities provided by Epsilon Integration Tool is its ability to integrate a 3D volume data into a plane as shown in Figure 13.

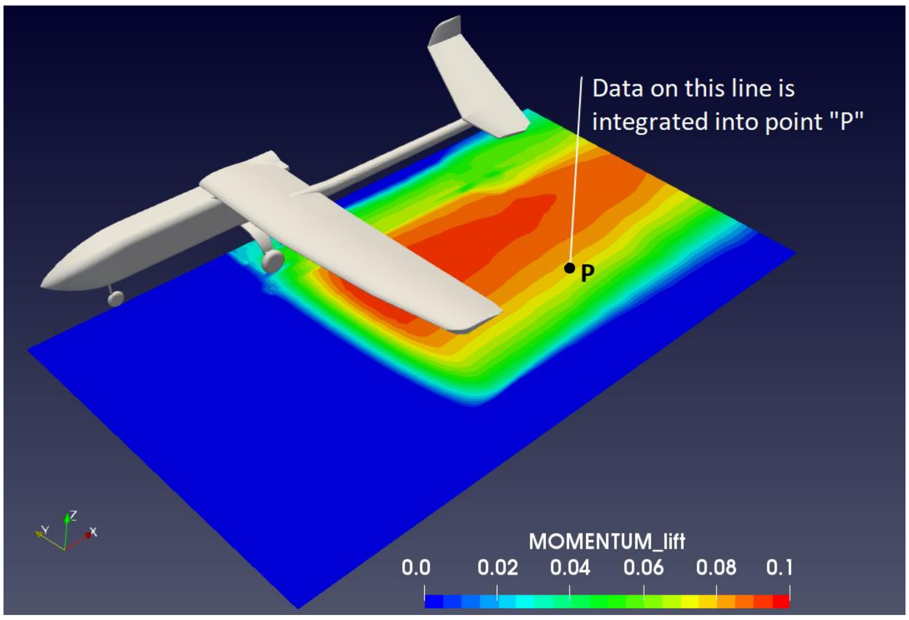


Figure 13: integration of 3D data into a plane

Thus, each point of this plane represents the local integral of the 3D domain along the survey line. This is achieved by creating a survey line traversing the entire 3D domain, placed normal to the upstream flow direction (because the integrated far-field and exergy values are only meaningful if the survey plane is normal to the upstream direction). This line is obtained by the intersection of 2 planes: the survey plane and a plane parallel to the x- and z- axes (called "intersection plane").

This survey line then is iteratively placed at each point of a user defined plane mesh and then data is integrated along the line. Hence, the integrated data is stored at each point of the mesh.

6 COMPLEMENTARY METHODS

Some of the formulae discussed in Chapter 1 cannot be directly implemented in Epsilon Aero Data because of its complexity. That is why they are treated separately.

6.1 Epsilon Near-Field

The near-field method requires knowing some body surface parameters like the surface normal vectors. This requires first to extract the body surface data from the hole field data and then to execute Epsilon Near Field. This filter computes the normal vectors and wall shear stress at each surface point, leading to the calculation of the near field lift, drag and moments.

6.2 Epsilon Profile Drag Breakdown

This filter allows performing a profile drag breakdown into viscous and wave drag, based on Kusunose and Aguirre approaches. Both methods start from the drag density field of some parameter provided by Epsilon

Aero Data (e.g., MEHEUT_profile_drag). This field is then decomposed following different criteria to split the domain: the Kusunose method relies on the vorticity field whereas the Aguirre method relies on the viscous anergy generation field.

6.3 Epsilon Poisson Solver

The vortex drag calculation based on the Maskell and Aguirre methods requires the determination of the stream function and the potential function on the survey plane. This is provided by the Epsilon Poisson Solver custom filter. It first reads a planar wake data: it could be a slice of the 3D CFD flow field or the actual wake data obtained in a WTT. Then it solves the Poisson equation by using Green functions. Finally, it calculates the induced drag and vortex exergy parameters.

CHAPTER 3: INSTALLATION PROCEDURE

1 DISCLAIMER

Epsilon V2.0 was developed for **Paraview versions with Python 2.7**. It will not work in Paraview versions with Python 3.

2 EPSILON DOWNLOAD

Download the latest Epsilon version here:
https://websites.isae-supaero.fr/epsilon-exergy-analysis-tool/download/

3 EPSILON INSTALLATION

The installation procedure just requires 4 steps:

- Place XML files and Python files in the desired folder
- Update the Python file path indicated in the XML files
- Load the plugins in Paraview
- Restart Paraview

Those steps are detailed below.

3.1 Files organization

If you have administrator rights, it is advisable to place all the files (XML and Python) in the Paraview installation folder or directly on the disk C. If you don't have administrator rights, you can place the files elsewhere but, once the plugin is successfully installed, they must not be removed.

3.2 File path definition

If the installation folder is not "**C:/EPSILON/**", all the XML files must be slightly edited: they must be opened with a text editor and the Python **file path must be updated**. The modifications to do are detailed in Table 2.

The best and quick way to edit all the files at the same time in a single step is to use a **TXT editor** (e.g., Notepad++), which allows replacing the file paths very easily with his "**Search and replace**" tool.

It is also possible to modify this path by hand before loading the plugins, by using the GUI displayed on Paraview's screen when the filter is executed. So, there is an alternative way to set up the file path directly on Paraview but the first method is recommended.

FILE	ACTION
Epsilon_Aero_Data.py	edit line 58
Epsilon_Aero_Data.xml	edit line 55
Epsilon_Axes.xml	edit line 52
Epsilon_Integration_Tool.xml	edit line 53
Epsilon_Near_Field.xml	edit line 54
Epsilon_Profile_Drag_Breakdown.xml	edit line 46
Epsilon_Poisson_Solver.xml	edit line 46
Epsilon_Wind_Tunnel.xml	edit line 39

Table 2: file path edition

3.3 Loading the plugins

Open Paraview and follow these steps:

Tools → Manage Plugins… → Load New → Files of type: select XML type
→ Search and Open XML file
→ Activate "Auto Load"

A legend with the name of the plugin will appear in the plugins list name and its property status must be "loaded". Otherwise, an error message will pop-out and the status will be "Not Loaded".

If the user wants to automatically load this plugin every time Paraview is opened, the "**Auto Load**" option must be selected. Click on the triangle placed at the left of the plugin name. This will drop-down advanced options. Then activate the option "Auto Load" and close.

Besides the Epsilon plugins, the user must also activate the "**SurfaceLIC**" standard Paraview plugin.

3.4 Reset session

The final step is to reset the session. This allows to restart Paraview properly and avoiding any error in the installation procedure.

Edit → Reset session

Now just close Paraview and open it again: you are ready to perform an aerodynamic analysis with Epsilon.

CHAPTER 4: ANALYSIS OF 2D CFD DATA

In order to simplify the usage of Epsilon, its use can be divided in two cases: 2D and 3D applications. The main Epsilon functionalities will be explained for the 2D case but they are valid for 3D cases as well.

1 PROCEDURE

The list of recommended actions for the analysis of 2D cases is detailed in Figure 14:

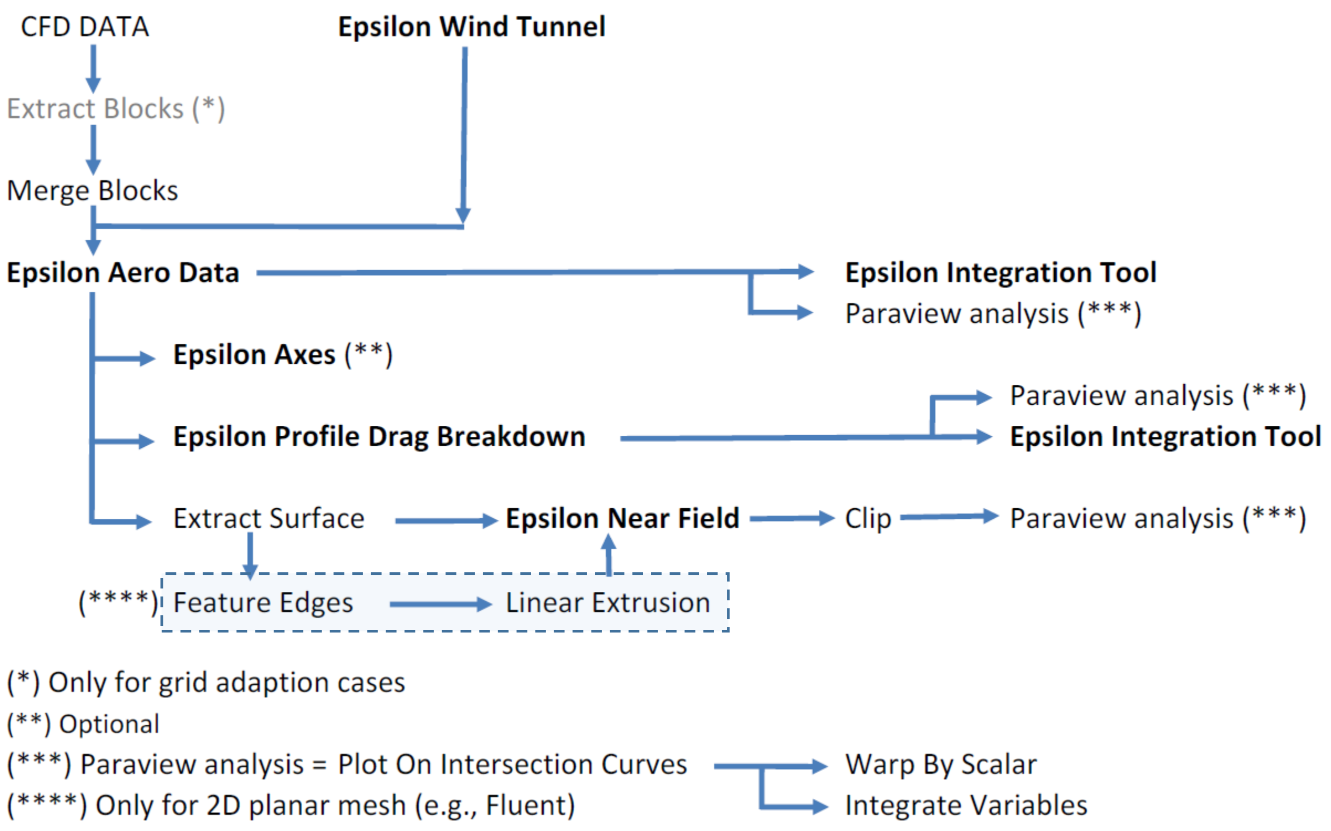


Figure 14: setup procedure for 2D cases

2 PREVIOUS CONSIDERATIONS

It is **NOT recommended** to activate the "**Auto Apply**" option because it may leads to unattended actions that requires a lot of time to be executed. Instead, it is recommended that the user confirms every action (The confirmation is made by clicking on the "Apply" button every time an action is demanded).

3 LOADING CFD/WTT DATA

The first stage is to load the data into Paraview. The procedure change slightly depending on the type of file to read:

3.1 Experimental data

Sources → EPSILON → Epsilon Wind Tunnel → Define CSV file path
→ Select instrument type
→ Select survey plane alignment
→Define origin position (if needed)
→Apply

The name of the variables contained in the file must be the followings: X, Y, Z, Vx, Vy, Vz, Ps and Pt

3.2 Fluent CAS files

File → Open → Select .cas Fluent file
→ Open data with →Fluent Case Files →Ok

The name of the variables contained in the file must be the followings: X_VELOCITY, Y_VELOCITY, Z_VELOCITY, PRESSURE, DENSITY, TEMPERATURE, MU_LAM and MU_TURB.

3.3 Fluent CGNS files

File → Open → Select .CGNS Fluent file
→ "Properties" tab → Select all the "Points Array"

The name of the variables contained in the file must be the followings: Velocity, Pressure, Density, Temperature, ViscosityMolecular and ViscosityEddy.

It is advisable to check if the CGNS file was well loaded into Paraview by plotting the "dV_dX" array. It must provide a reasonable variation around the body. If the CGNS was not well generated in Fluent, a weird distribution of "dV_dX" will displayed, especially towards the far field boundary. In this case, it is necessary to create again the CGNS file in Fluent.

3.4 Fluent Ensight Gold Case files

File → Open → Select .encas file
→ Open data with → Ensight Files →Ok

The name of the variables contained in the file must be the followings: x_velocity, y_velocity, z_velocity, pressure, density, temperature, viscosity_lam and viscosity_turb.

Note: the user may want to apply "Extract Block" filter in order to retain only the fluid region and discard the mesh lines (internal_fluid) that annoys the visualization.

3.5 Open Foam files

File → Open → Files of type → All files → Select .OpenFOAM file
→ Open data with → OpenFOAM →Ok

For single a block mesh, let the properties panel by default an Apply. However, for **multiblock** mesh cases, select:

Case type →Decomposed Case

The name of the variables contained in the file must be the followings: U, p, t and nut.

3.6 STAR CCM Ensight Gold Case files

File → Open → Select .case file → OK

The name of the variables contained in the file must be the followings: Velocityi, Velocityj, Velocityk, AbsolutePressure (or StaticPressure), Density, Temperature (or StaticTemperature), TurbulentViscosity and TurbulentViscosityRatio (or EffectiveViscosity).

3.7 elsA Tecplot files

File → Open → Files of type → All files → Select solutionXXXX file →OK
→ Open data with → Tecplot Binary Files (Visit) →OK

The name of the variables contained in the file must be the followings: ro, rovx, rovy, rovz, roE, rok and roeps.

3.8 elsA CGNS files

File → Open → Select .CGNS file
→ "Properties" tab → Select all the "Points Array"

The name of the variables contained in the file must be the followings: Density, Momentum, Pressure, Temperature and Viscosity_EddyMolecularRatio.

3.9 TAU Tecplot files

File → Open → Files of type → All files → Select .plt file →OK
→ Open data with → Tecplot Binary Files (Visit) →OK

The name of the variables contained in the file must be the followings: x_velocity, y_velocity, z_velocity, pressure, density, temperature, viscosity and eddy_viscosity.

3.10 Potential Flow Generator files

Epsilon can also be used to analyze data coming from the "Potential Flow Generator". It is a Paraview data source developed by ISAE, allowing determining the potential flow field around a cylinder or an airfoil (by using transformations):

Sources → Potential Flow Generator → Set up desired parameters → Apply

The name of the variables contained in the file must be the followings: X_VELOCITY, Y_VELOCITY, Z_VELOCITY, PRESSURE, DENSITY, TEMPERATURE, MU_LAM and MU_TURB.

4 EXTRACT BLOCK/MERGE BLOCKS

CFD data must be pre-treated before applying Epsilon Aero Data filter:

4.1 Extract Block

This step is only mandatory for **grid adaption** cases. CFD cases without an adapted mesh must skip this step.

In an adapted CFD case, extra blocks are created for the refined regions. If all those blocks are loaded (i.e., the original mesh blocks + the blocks of the refined regions) a visualization problem arises as shown in Figure 15:

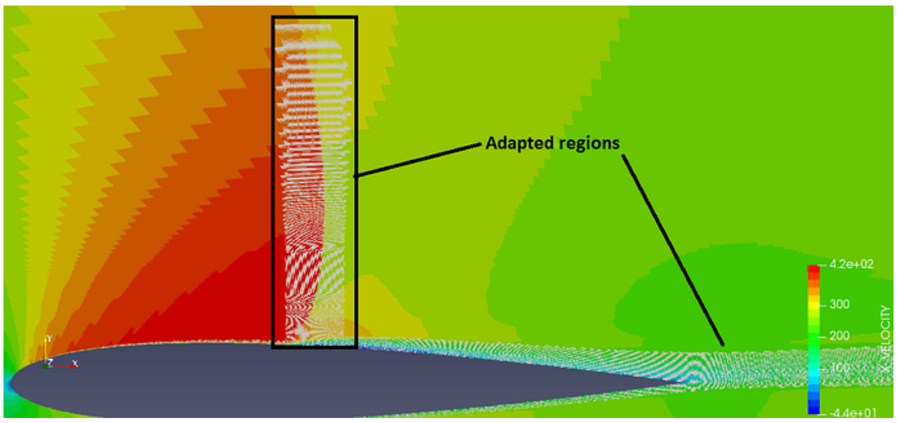


Figure 15: visualization problem

This problrm is not limited to the visualization but also to the data itself, because the aerodynamic variables are missing in these new blocks. This is solved by applying an **Extract Block filter** and by selecting all blocks except the blocks concerning the refined regions. The result is shown in Figure 16:

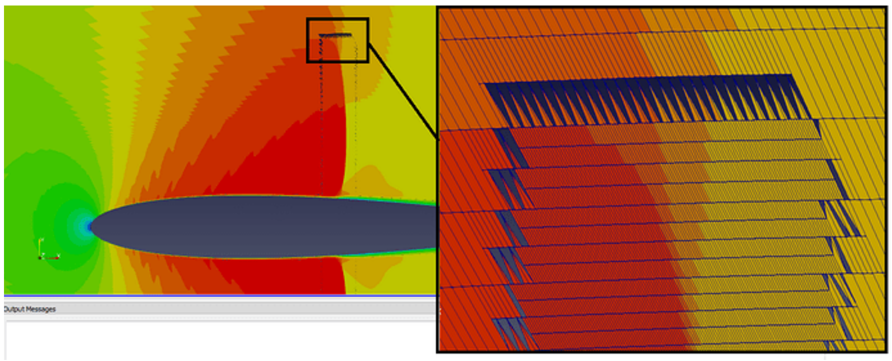


Figure 16: output from Extract Block filter

Even though some cells at the limits of the refinement region are now missing, it does not represents a problem for the aerodynamic analysis by Epsilon.

4.2 Merge Blocks

The current version of **Epsilon (V2.0) does not support multiblock data sets**, hence a Merge Blocks filter must be applied. A more efficient multiblock analysis will be supported in future Epsilon versions.

5 EPSILON AERO DATA

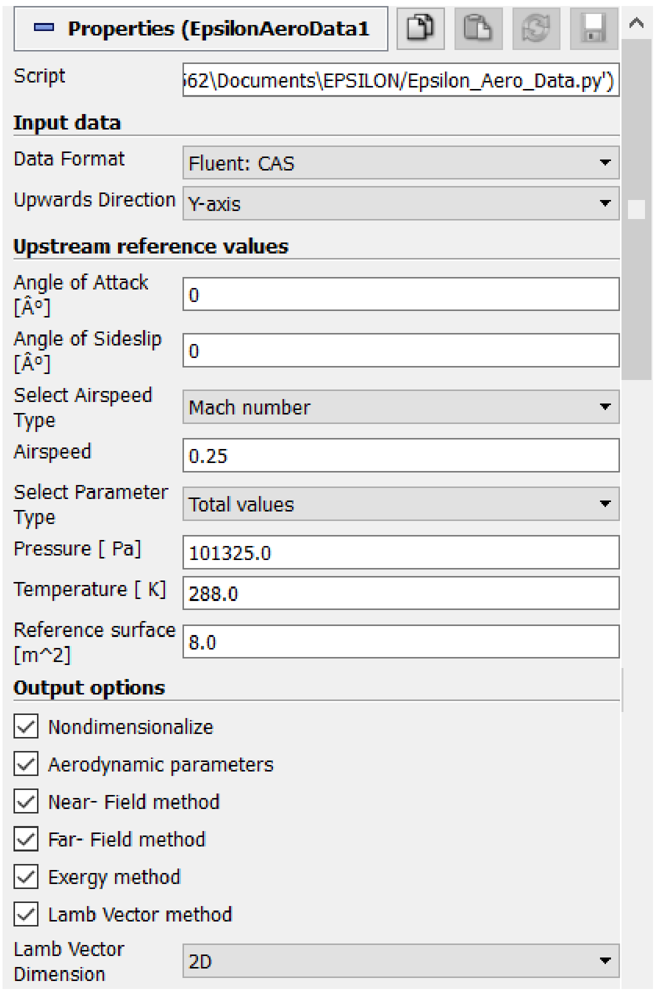
At this point we are ready to implement the aerodynamic analysis equations (i.e., near-field, far-field, Lamb vector and exergy). This is made by applying the "Epsilon Aero Data" filter.

First make sure that you have selected the input data (The Epsilon Wind Tunnel filter or the Merge Blocks filter) in the pipeline browser (it must be highlighted in blue), and then apply the Epsilon Aero Data filter as follows:

Filters → Epsilon → Epsilon Aero Data

5.1 Filter setup

Before clicking on the "Apply" button, it is necessary to setup the filter by using the properties panel as seen in the Figure 17.


Properties (EpsilonAeroData1

Script	;62\Documents\EPSILON/Epsilon_Aero_Data.py')
Input data	
Data Format	Fluent: CAS
Upwards Direction	Y-axis
Upstream reference values	
Angle of Attack [Â°]	0
Angle of Sideslip [Â°]	0
Select Airspeed Type	Mach number
Airspeed	0.25
Select Parameter Type	Total values
Pressure [Pa]	101325.0
Temperature [K]	288.0
Reference surface [m^2]	8.0
Output options	
✓ Nondimensionalize	
✓ Aerodynamic parameters	
✓ Near- Field method	
✓ Far- Field method	
✓ Exergy method	
✓ Lamb Vector method	
Lamb Vector Dimension	2D

Figure 17: Epsilon Aero Data properties panel

The option "Script" allows modifying the Path to the Python file. If the installation procedure recommended in the previous chapter was followed, the user must ignore this option.

The user must define which type of data has been loaded by using the "Input Format" drop down list. Then it must be specified the mesh upwards direction as well as the angles of attack and sideslip. This information allows defining the parameters of the transformation matrices discussed in Chapter 1.

Then it must be specified some upstream reference data, which allows calculating all the upstream parameters. First of all, the airspeed can be defined as a Mach number value or a Velocity value, by using a drop-down list. Then, its airspeed value must be indicated. The pressure and temperature are also necessary, so the user can define these values as static or total values by using another drop-down list and by indicating their values in a dedicated panel. The reference surface value must be specified: it is used to calculate the forces and exergy coefficients. This reference surface is equal to the **airfoil chord** in 2D cases and equal to the **wing surface** for 3D cases. The 2D cases consider that the depth is unitary, thus the 2D reference surface equals the chord times the unitary span length.

The parameters described above are the ones to modify each time Epsilon Aero Data filter is applied. The remaining parameters are optional.

The "Nondimensionalize" button (activated by default) allows showing drag, lift, exergy and anergy coefficients instead of its dimensional values.

The user can also select the desired output variables. All methods are activated by default which is suited for most of analyses. However, it must be kept in mind that the computation time and the **required memory to store the output data** could be demanding for 3D CFD cases with a large amount of cells.

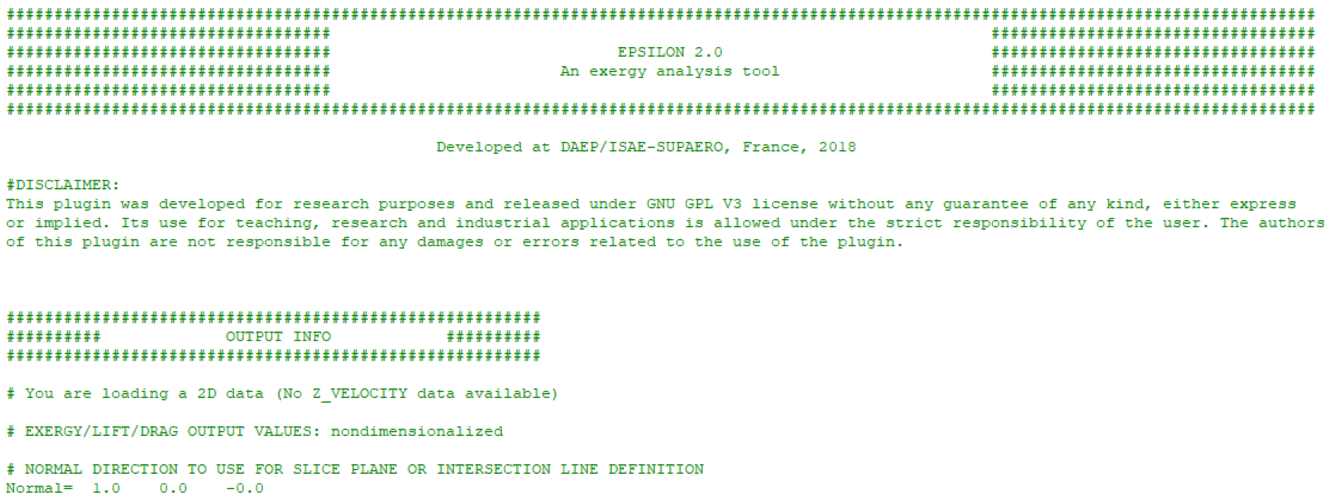
Lamb vector calculations also require to define the field dimension (2D or 3D field). This can be made by using a drop down list.

5.2 Output messages

After defining the previously discussed parameters the Epsilon Aero Data filter can be executed ("Apply" button). By doing so, the Paraview's output messages window will display some information as shown in Figure 18 (in order to see it properly, the user must active the option "**View Full Messages**" or "**Console View**" in the output messages window).

First of all, it informs the user that the Python code has recognized the input data as 2D or 3D data. It also indicates if the output data will be non-dimensionalized or not. These are only confirmation messages. No further action is required.

The last message shows the **survey plane normal values** to be used later by the user in order to setup the survey plane inclination. Thus, it is advised not to clear the Paraview's Output Messages window; otherwise, this information will be lost. Nevertheless, this information can be recovered by simply re-executing (Apply) the Epsilon Aero Data filter again.



```
#####################################################################################################
##################################                                  ##################################
##################################             EPSILON 2.0          ##################################
##################################         An exergy analysis tool  ##################################
##################################                                  ##################################
#####################################################################################################

                                     Developed at DAEP/ISAE-SUPAERO, France, 2018

#DISCLAIMER:
This plugin was developed for research purposes and released under GNU GPL V3 license without any guarantee of any kind, either express
or implied. Its use for teaching, research and industrial applications is allowed under the strict responsibility of the user. The authors
of this plugin are not responsible for any damages or errors related to the use of the plugin.


########################################################
##########          OUTPUT INFO            ##########
########################################################

# You are loading a 2D data (No Z_VELOCITY data available)

# EXERGY/LIFT/DRAG OUTPUT VALUES: nondimensionalized

# NORMAL DIRECTION TO USE FOR SLICE PLANE OR INTERSECTION LINE DEFINITION
Normal=  1.0    0.0    -0.0
```

Figure 18: printed info in the Output Messages window

5.3 Visualizing the aerodynamic data

After executing Epsilon Aero Data, we are ready to exploit the aerodynamic parameters calculated. This aerodynamic data can be quickly displayed by clicking on the drop-down list tagged as "Solid Color" in the menu bar as shown in Figure 19. This will show the list of data arrays available. By selecting any property, the mesh will be automatically colored. The meaning of each parameter of this drop-down list is indicated in the Appendix 1. The physical interpretation of most of these parameters is presented in [25, 26].

IMPORTANT: in order to easily manipulate the 2D object on the screen (i.e., zoom or pan-and-tilt) the "3D" button must be activated in advance (it is highlighted in red in the Figure 19).

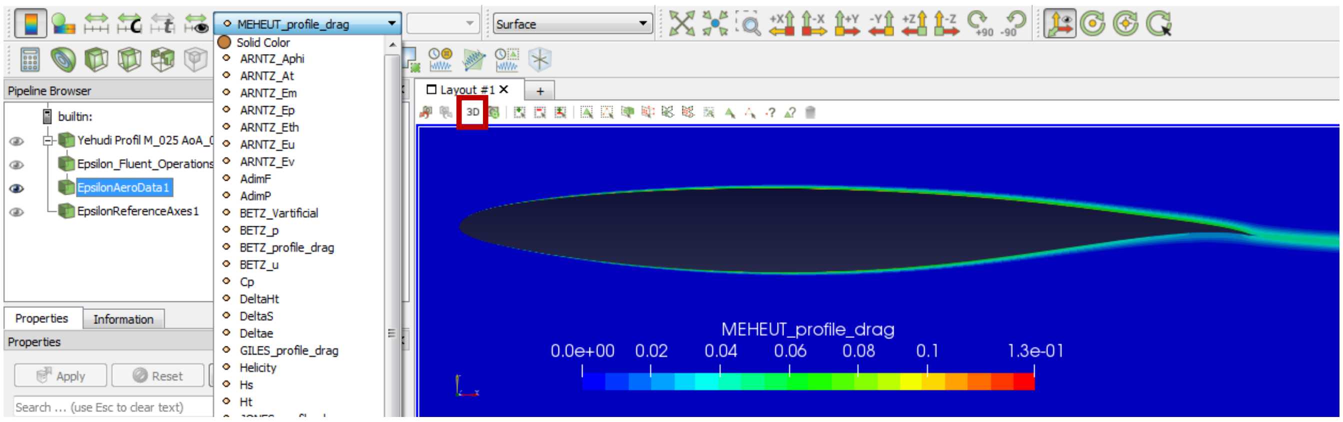


Figure 19: selection of visualization field

By default, color map is scaled to contain the maximum and minimum values. If a custom range is needed, the scale color map options marked in red in Figure 20 must be used.

In order to see the flow pattern in 2D cases, go to the menu bar and search the drop down list named "Surface", then select the "SurfaceLIC" option (See Figure 21). Note that this streamlines can be easily colored by some aerodynamic parameter by simply selecting the desired array in the menu bar. This approach has the drawback that no filter parameter can be changed to modify the visualization (line density and line thickness are already fixed).

IMPORTANT: it is forbidden to activate the full-screen mode (F11 keyboard button) while displaying the surfaceLIC flow pattern. This will make Paraview to crash and close (It is a bug).

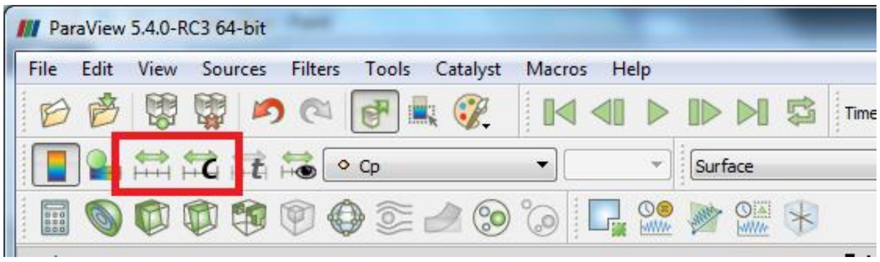

ParaView 5.4.0-RC3 64-bit
File Edit View Sources Filters Tools Catalyst Macros Help
Cp
Surface
Time

Figure 20: color map scale

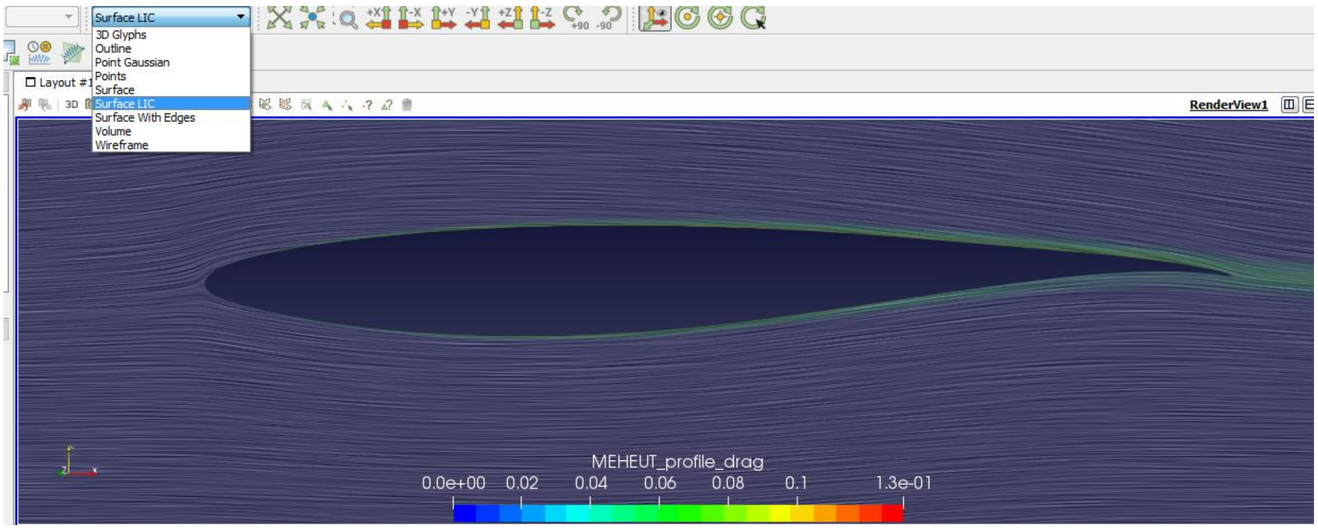


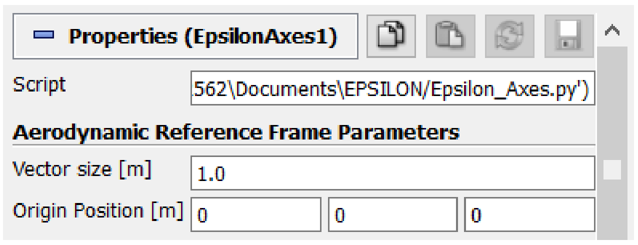
Figure 21: surface LIC display

6 EPSILON AXES (OPTIONAL)

This stage is optional but it is recommended for beginners because it helps to interpret the output data from Epsilon Aero Data. This filter creates a set of 3 vectors representing the aerodynamic axes:

Filters → Epsilon → Epsilon Axes

The user must only define the size of the vectors and the position of the origin of these vectors (Fig. 22)



Properties (EpsilonAxes1)

Script: 562\Documents\EPSILON/Epsilon_Axes.py')

Aerodynamic Reference Frame Parameters

Vector size [m]: 1.0

Origin Position [m]: 0 | 0 | 0

Figure 22: Properties panel for the Epsilon References Axes source

The vectors are not labelled like the mesh vectors. However they have the same color code: Red=x-axis, yellow=y-axis, green=z-axis (See Figure 23).

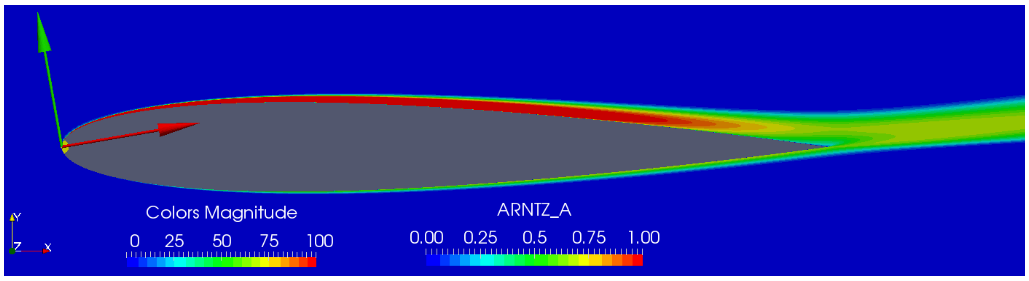


Figure 23: wind axes color code

This color code must be activated by selecting the variable "Colors" and its option "Magnitude" in the tool bar drop-down list as shown in Figure 24. If the user wants to hide the "Colors Magnitude" legend in the visualization, the button highlighted in red must be deactivated:

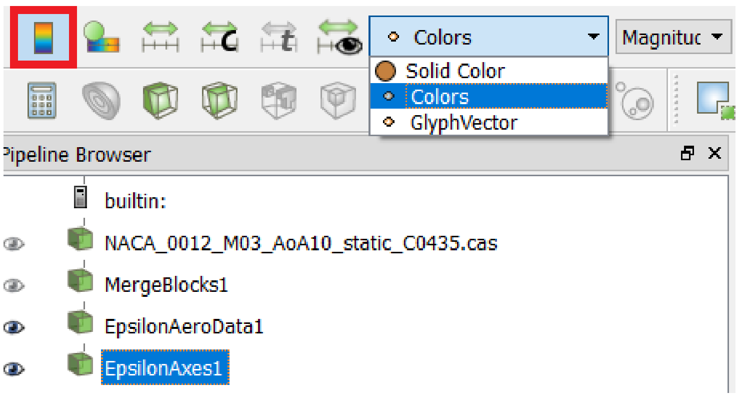


Figure 24: wind axes ID coloring

IMPORTANT: the default color map MUST be BGR (blue-green-red). Otherwise, the color code will be meaningless.

7 PARAVIEW ANALYSIS

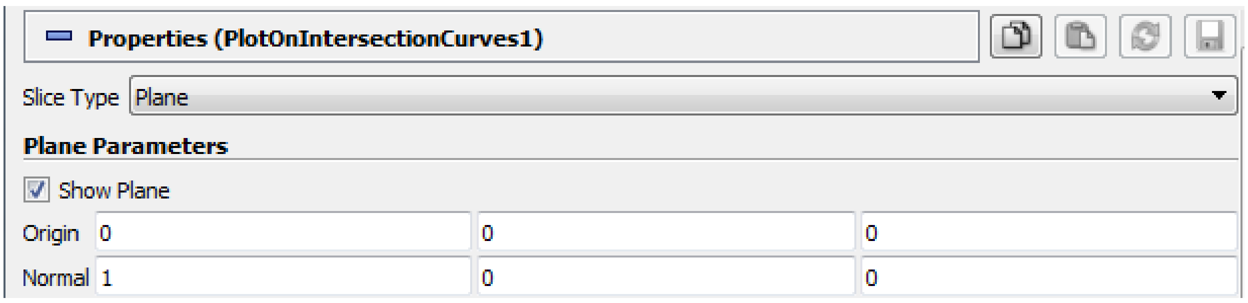
The output from Epsilon Aero Data can be further analyzed by using the standard ParaView tools. For 2D cases, there are three filters particularly useful: Plot On Intersection Cuves, Integrate variables and warp by scalar.

7.1 Plot On Intersection Curves

One of the most practical 2D data analysis is the observation of the aerodynamic parameters distribution along the **survey line** (Survey line = trace of the survey plane in 2D cases). In order to plot the aerodynamic parameters along a survey line, a "Plot On Intersection Curves" standard filter must be applied directly to the Epsilon Aero Data output:

Filters → Alphabetical → Plot On Intersection Curves

By default it defines an intersection plane whose origin and normal must be defined by the user (Figure 25):


Properties (PlotOnIntersectionCurves1)

Slice Type: Plane

Plane Parameters

☑ Show Plane

Origin	0	0	0
Normal	1	0	0

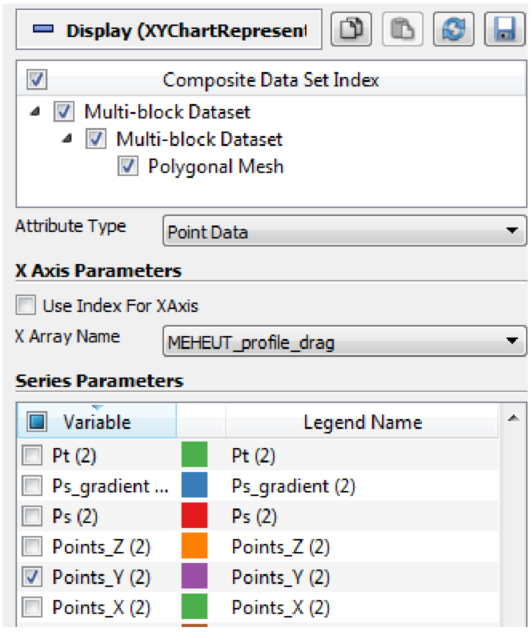
Figure 25: Plot On Intersection Curves parameters

The "origin" has 3 boxes to preset the x-, y- and z-coordinates. The user must use the x-coordinate to place the survey plane at the intended position (usually downstream of the body).

The normal direction of the plane must be settled by the user based on the output message of the Epsilon Aero Data filter ("`NORMAL DIRECTION TO USE FOR SLICE PLANE OR INTERSECTION LINE`").

Note: it is advised to deactivate the "Show Plane" checkbox because it may annoys during the manipulating of the render view object.

After applying the filter, a new window will be displayed ("Line Chart View" type). By clicking on this window, its properties will be accessible as shown in Figure 26. The user must ensure that all the boxes under "Composite Data Set Index" are checked. Then, it is possible to select the parameter to be plotted along the x-axis by choosing the desired parameter in the "X-Array Name" drop-down list. The parameter to be plotted along the Y-axis must be selected from the "Series Parameters" list. It is recommended to plot the Y-coordinates (Points_Y or Points_Z) along the Y-axis and the aerodynamic parameter along the X-axis as it is shown on the right hand side of the Figure 26.


Display (XYChartRepresent

☑ Composite Data Set Index
- ☑ Multi-block Dataset
  - ☑ Multi-block Dataset
    - ☑ Polygonal Mesh

Attribute Type: Point Data

X Axis Parameters

☐ Use Index For XAxis

X Array Name: MEHEUT_profile_drag

Series Parameters

Variable		Legend Name
☐ Pt (2)	🟩	Pt (2)
☐ Ps_gradient ...	🟦	Ps_gradient (2)
☐ Ps (2)	🟥	Ps (2)
☐ Points_Z (2)	🟧	Points_Z (2)
☑ Points_Y (2)	🟪	Points_Y (2)
☐ Points_X (2)	🟩	Points_X (2)

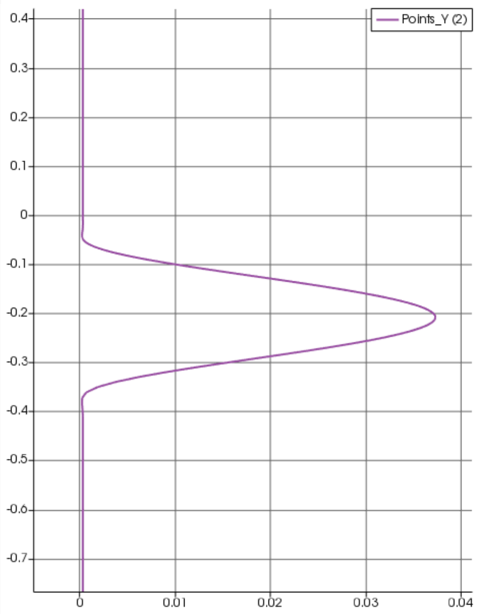


Figure 26: Plot On Intersection Curve settings

Note: the user may want to visualize the actual survey line overlaid on the 2D flow field in order to have an idea of its position. This can be made by clicking on the Render View in order to set this window as the active window. Then, go to the Pipeline Browser and click on the "visualize/hide icon" highlighted in red in Figure 27. When it is in gray color, it is hidden. When it is in black color, it is shown on the active window. This visualization tool works in the same fashion for any element on the Pipeline browser and it changes of course from one active window to the other.

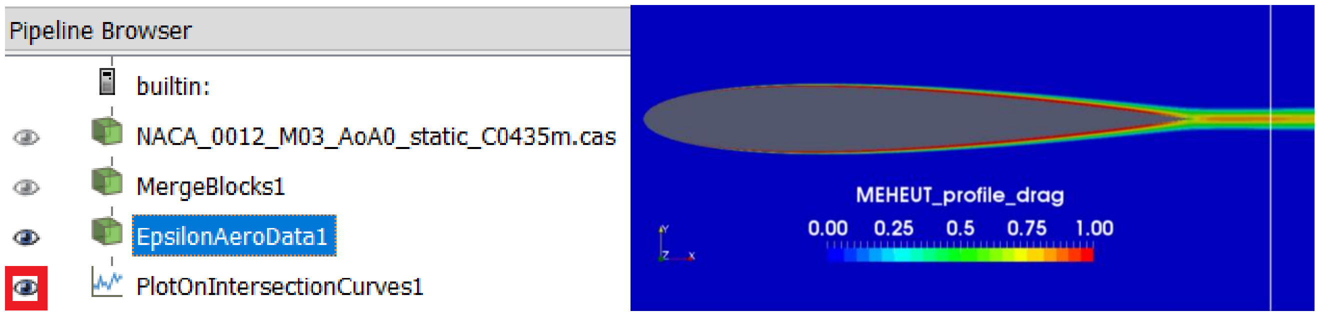


Figure 27: show or hide an element

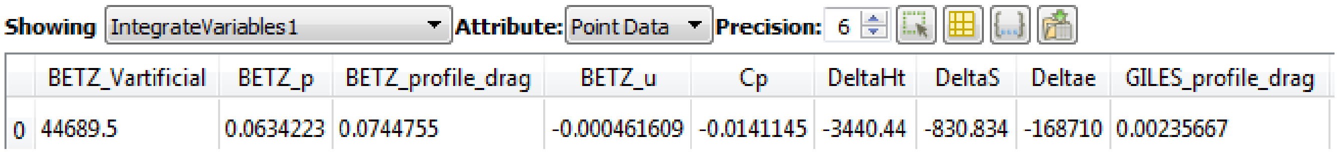
7.2 Integrate variables

So far we were concerned with data visualization. However the main objective is to obtain the value of drag, lift, exergies, and so on. This is achieved by integrating the data contained on the survey plane (survey line in 2D CFD cases). This is because the drag distributions shown before are in fact drag densities: any variable calculated by Epsilon represents just the integrand of the equations presented in Chapter 1 (The detail of each parameter is available in the Appendix 1). In order to obtain the integrated value of the equations, an integration of the density data must be carried out.

Select again the "PlotOnIntersectionCurves" filter on the pipeline browser and apply the "Integrate Variables" filter:

Filters → Alphabetical → Integrate Variables

A new window will appear in a table format (**Spreadsheet view**), showing the integrated value of each parameter contained in the survey line as shown in Figure 28.



	BETZ_Vartificial	BETZ_p	BETZ_profile_drag	BETZ_u	Cp	DeltaHt	DeltaS	Deltae	GILES_profile_drag
0	44689.5	0.0634223	0.0744755	-0.000461609	-0.0141145	-3440.44	-830.834	-168710	0.00235667

Figure 28: integrated values

Important: The user must keep in mind that the integrated values of the integrands give the total value of these parameters (For example, the integration of Betz Profile Drag density will give the actual value of profile drag). On the other hand, the integrated value of any other field data than the equation integrands is meaningless (like the "ARNTZ_Ep" integral). Also, the integration of the wave anergy and the viscous anergy densities (among others) are **meaningless**. This is because these integrands require a volume integration to give the viscous and wave anergy (so, its plane integration is meaningless, or at least it can be interpreted as the **local anergy generation rate** at the survey plane position).

7.3 Warp by scalar/Clip by scalar

As we have seen in Figure 26, the "Plot On Intersection Curves" filters allows plotting data in a separate window. However, it may be difficult to relate the data values with its spatial position. So, it would be nicer to have the same plot but superimposed to the survey line as shown in Figure 29, where the vertical white line is the survey plane and the red curve is the qualitative profile drag distribution along this line:

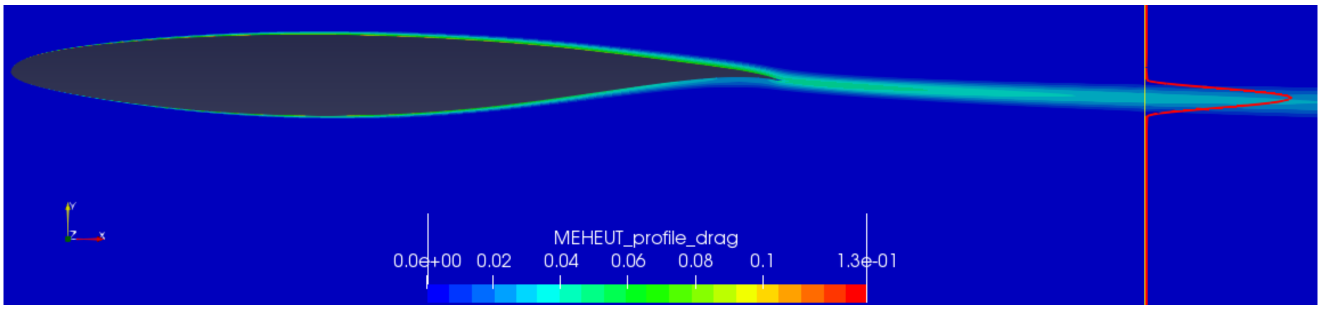
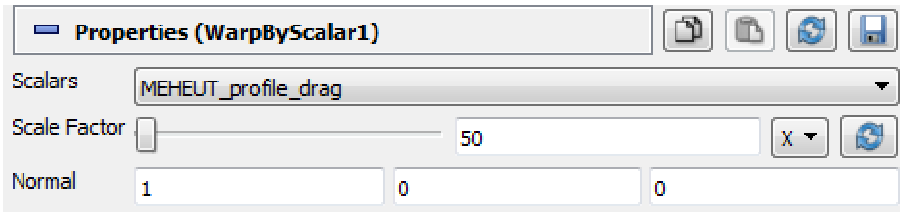


Figure 29: warp by scalar

This is achieved by applying a "Warp By Scalar" filter to the existing "Plot On Intersection Curve":

Filters → Alphabetical → Warp By Scalar

The user must select the aerodynamic parameter to do the warp of the survey line by using the "Scalars" drop-down list (Figure 30). Then he must select the proper warp direction (it should be normal to the survey plane: use the "normal direction" printed in the output messages window). If necessary, the warp magnitude can be enlarged or reduced by using the Scale Factor:



Properties (WarpByScalar1)

Scalars	MEHEUT_profile_drag		
Scale Factor	50	X	
Normal	1	0	0

Figure 30: warp by scalar parameters

This feature is also extremely useful to detect a possible drag density shift as it is shown in Figure 31. In fact, sometimes the small numerical errors gives a non-zero drag density outside the wake region. Instead, a slightly positive drag value is encountered in the zones outside the wake. Thus, when integrating the drag density all along the line, the resulting drag value will be wrong or biased. So, it is advisable to check if the density is shifted by using the "Warp By Scalar" filter.

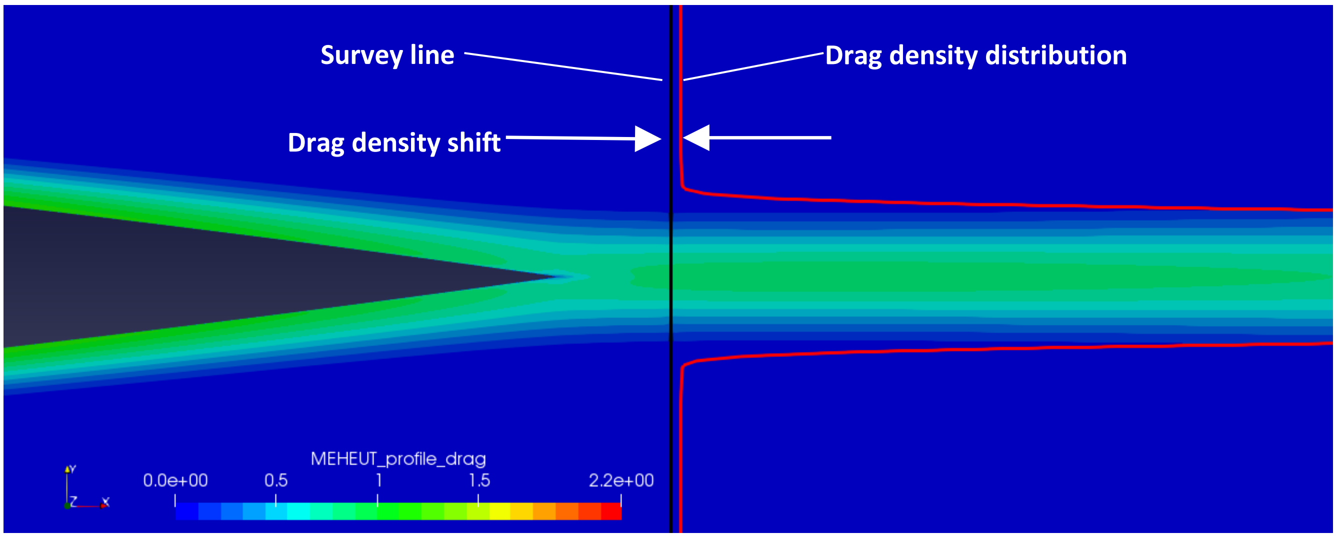


Figure 31: drag density shift

If a drag density exists, the simplest way to improve de accuracy of the drag prediction is to limit the integration zone to the wake region (in fact, this shift appears in WTT far-field methods that require a wake integral only). The user must select the "Epsilon Aero Data" filter in the pipeline and then to apply the Clip filter:

Filters → Alphabetical → Clip → Clip type → scalar
Scalars → Select the desired parameter
Value → indicate the threshold value

The parameter to be selected must be such that it varies only inside the wake (you can use P_t/P_{t0}, a drag density or the total anergy variable). The threshold value will depend on the selected variable (for P_t/P_{t0}, a good starting values is 0.9999). Hence, by applying this filter, only the zone inside the wake will be kept as shown in Figure 32. Then this output can be used to make a "Plot Over Line" followed by "Warp by Scalar".

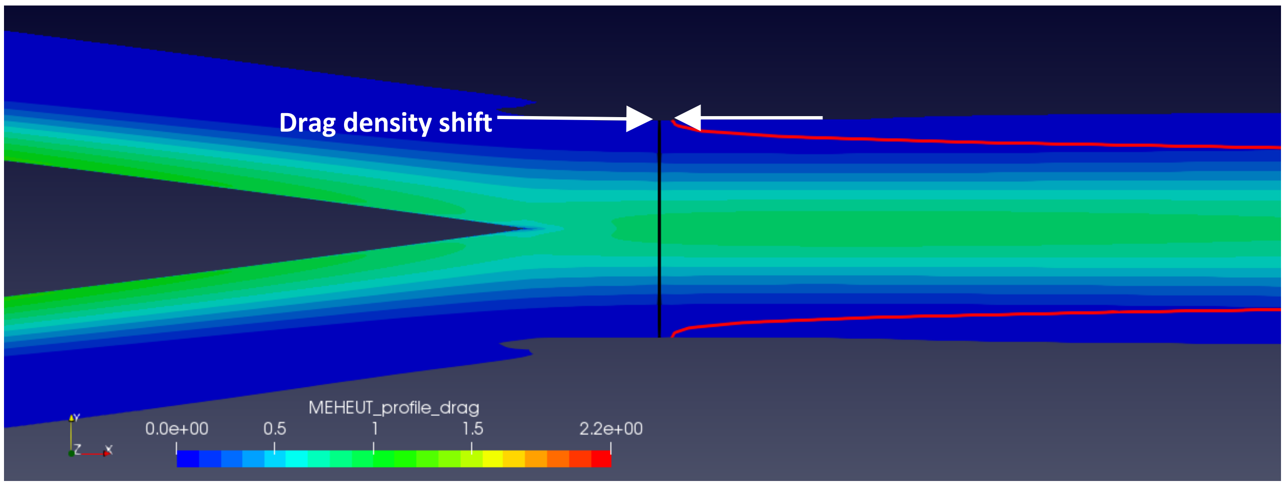


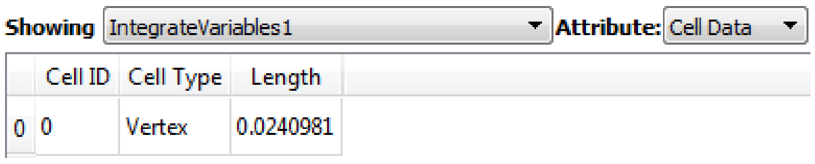
Figure 32: clipped domain

It can be seen that the survey line no longer contains the shifted non-zero values outside the wake. Thus, by integrating data along this new clipped line, an accurate drag value will be obtained. **This is the standard recommended procedure for drag extraction in Epsilon for any far-field method and for the WTT-exergy**

method (Aguirre). The only precaution to take care about this procedure is to avoid excessively cropping the wake by using an aggressive clipping value. Another possible issue is that sometimes the clipped survey line does not allow to be integrated (a very rare situation). In this case the integrated value shows a "NaN" legend in the spread sheet view. This means that Paraview was unable to integrate the array because of some missing data along the line. The simplest way to overcome this is by clipping a box around the desired wake region, then to apply the survey line on this box and to integrate values.

If the user is looking for highly accurate drag values, the remaining drag density shift still present inside the clipped wake must be manually removed (see Figure 32). This is made by inspecting two parameters on the Spreadsheet view: the length of the survey line and the shift drag value.

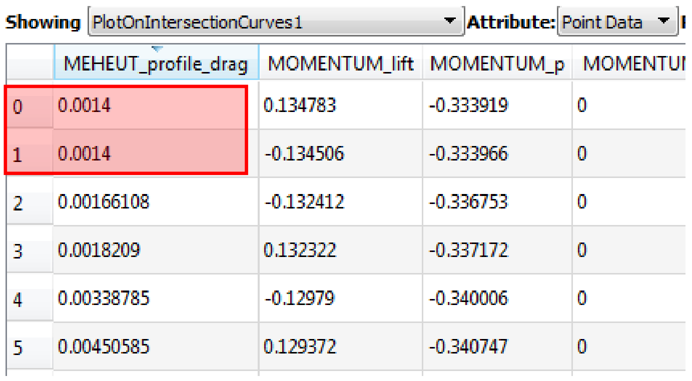
The length is visible by selecting the "Integrate Variables" and the by changing the attribute from "Point Data" to "Cell Data" (Figure 33). The field "Length" shows the line lenght.


Showing IntegrateVariables1 Attribute: Cell Data

	Cell ID	Cell Type	Length
0	0	Vertex	0.0240981

Figure 33: finding the length of the clipped survey line

On the other hand, the shift drag value is the value used for clipping the domain (Thus this value is still available in its related Property Panel). It can also be obtained by selecting on the spreadsheet view the "PlotOnIntersectionCurves" and "Point Data". Then, by inspecting the desired drag density column one can identify the minimum drag value (Figure 34).


Showing PlotOnIntersectionCurves1 Attribute: Point Data

	MEHEUT_profile_drag	MOMENTUM_lift	MOMENTUM_p	MOMENTUI
0	0.0014	0.134783	-0.333919	0
1	0.0014	-0.134506	-0.333966	0
2	0.00166108	-0.132412	-0.336753	0
3	0.0018209	0.132322	-0.337172	0
4	0.00338785	-0.12979	-0.340006	0
5	0.00450585	0.129372	-0.340747	0

Figure 34: finding the shift drag value

The product of both values (line length x shift drag density) will give the shift drag inside the wake: this value must be removed from the integrated one. **The order of magnitude of this shift drag goes from 0.1 to 1.5 drag counts**, thus it is a small error that can be absolutely **negligible** unless highly accurate drag values be desired.

The methods indicated in this section are valid for 2D cases and 3D cases as well. The user must consider the good practice of clipping by scalar the wake before attempting any wake integration procedure. This is valid for far-field methods and WTT exergy formulation (Aguirre) only: **DON'T DO THAT** for the CFD exergy formulation (Arntz).

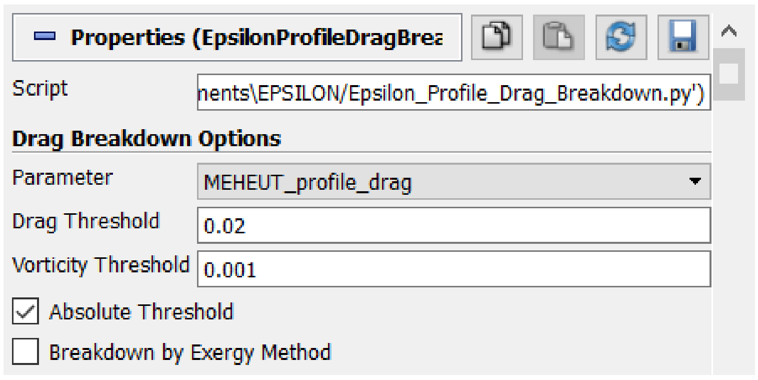
8 EPSILON PROFILE DRAG BREAKDOWN

Another interesting feature provided by Epsilon is the profile drag decomposition based on Kusunose's approach, i.e., profile drag = viscous + wave (or, equivalently, the exergy/anergy breakdown into viscous/wave components by the Aguirre's method).

Select again "Epsilon Aero Data" in the pipeline browser and apply the following filter:

Filters → Epsilon → Epsilon Profile Drag Breakdown

This will display the properties panel shown in Figure 35, which must be configured before clicking on "Apply". As usual, the "Script" window allows changing the path of the Python filter. It must not be modified if the standard installation procedure described in Chapter 3 was followed.



Script: nents\EPSILON/Epsilon_Profile_Drag_Breakdown.py')

Drag Breakdown Options

Parameter	MEHEUT_profile_drag
Drag Threshold	0.02
Vorticity Threshold	0.001

☑ Absolute Threshold

☐ Breakdown by Exergy Method

Figure 35: Epsilon Profile Drag Breakdown properties panel

This filter applies the breakdown method to a single variable at time: its output splits the field of the selected variable into three components called "wave", "viscous" and "jet". The latter identifies the regions with negative drag (i.e., jet plume).

Hence, the user must first select the desired variable to decompose by using the drop down list. Then, it must be setup two thresholds factors which are used to define a criterion to detect the wave drag. These threshold values can be defined as absolute values or relative to the maximum values. By default, "absolute values" option is activated in the checkbox. If unchecked, Epsilon will interpret this as "relative values".

If the "absolute values" definition is used, the wave drag region exists where: (Profile Drag<Drag Threshold) and (Vorticity < Vorticity Threshold).

If "relative values" are used, the wave drag region exists where: (Profile Drag<Drag Threshold*Maximum Drag) and (Vorticity < Vorticity Threshold*Maximum Vorticity).

The default absolute threshold values have been settled based on typical data but it may not work fine for all cases. Thus, the user must play with those values in order to obtain the correct decomposition. The quickest way to do so is to observe the output fields of the filter. In fact, the filter provides 5 output arrays: "Jet", "Profile_Drag", "Vorticity_Magnitude", "Viscous" and "Wave". "Profile_Drag" is the same array that has been selected to decompose (e.g., if you selected "MEHEUT_Profile_Drag", then the output called "Profile_Drag" is that same array -it has been repeated for convenience-). "Vorticity_Magnitude" is again a copy of that variable that comes from Epsilon_Aero_Data. Hence, "wave", "viscous" and "Jet" are the actual new outputs.

Then, the procedure to verify if the threshold values were correctly settled is the following. First, visualize the "Profile_Drag" field and find the correct range value limit that allows you to observe the shockwave as shown in Fig. 36-left. Then visualize the wave (Fig. 36-right) and viscous components (Fig. 37) by using the same data range. If the wave output does not contain any trace of the viscous region, then the breakdown was successful.

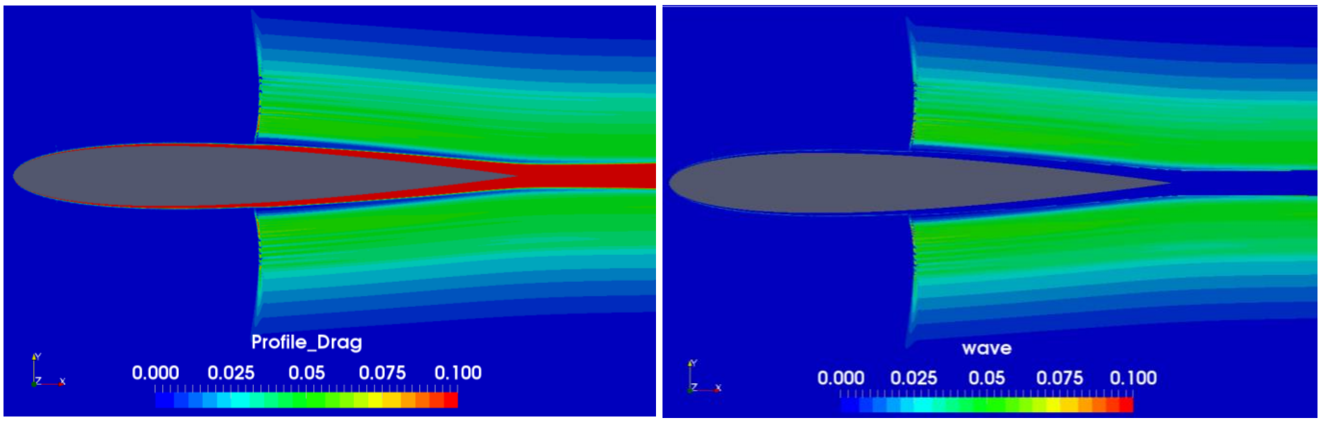


Figure 36: Profile Drag field (left) and wave drag (right)

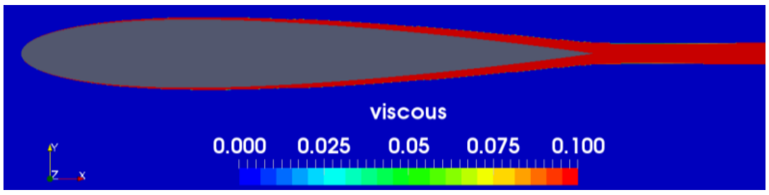


Figure 37: viscous drag

Another option of the filter GUI is "Breakdown by Exergy Method" (unchecked by default). If this option is selected, then, instead of using the Kusunose method, the Aguirre method is used. In this case, vorticity is no longer used to define the wave region, instead the viscous anergy rate parameter is used ("ARNTZ_Aphi" variable from the Epsilon Aero Data output). The advantage of this option is that the viscous anergy field is a more reliable parameter to define the threshold region [31].

If the "Breakdown by Exergy Method" option is selected, then, the viscous anergy threshold must be specified by using the same input box than the vorticity (The same box was kept in order to simplify the GUI).

9 EPSILON INTEGRATION TOOL

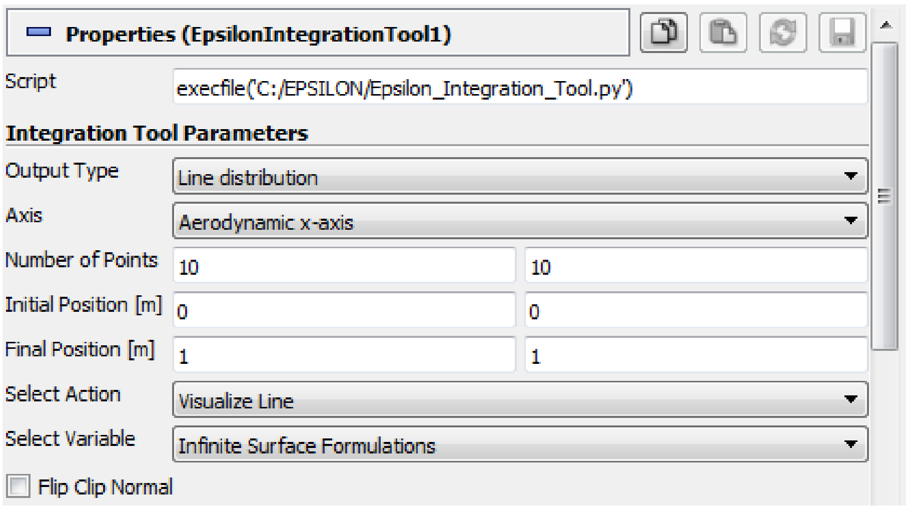
Epsilon Integration Tool is a versatile analysis tool that allows you to quantify and visualize the evolution of the integral of certain parameters along a specified direction. For 2D applications, the output is a line data. Select again "Epsilon Aero Data" in the pipeline browser and apply the following filter:

Filters → Epsilon → Epsilon Integration Tool

The related properties panel is shown in Figure 38 and it must be properly configured before clicking on "Apply" button. The "Output Type" must be "Line Distribution" for 2D cases. However, the rest of the parameters must be configured depending on the stage of the analysis. In fact, this filter must be used in a **three-step procedure**:

1) Creation and visualization of the line along which the survey plane integrals will be performed
2) Visualization of the survey plane
3) Integration of data at each survey station and storing this data along the line

The first step requires defining the integration line direction. This is achieved with the "Axis" drop down list: select the axis along which the integral will be performed. For 2D cases you can use either "Aerodynamic x-axis" or "Mesh X-axis". Then the position of this line and the amount of points along it must be defined. This is made by using the left column of the input data table (the right column will be used in 3D cases). Then select "Visualize Line" in the "Select Action" drop-down list and click on "Apply": this will create a line in the render view. In order to visualize the actual points of the line, change the visualization type from "surface" to "points". The result is shown in Fig. 39. The points are the positions of the survey plane used later to integrate data.



Properties (EpsilonIntegrationTool1)

Script	execfile('C:/EPSILON/Epsilon_Integration_Tool.py')	
Integration Tool Parameters		
Output Type	Line distribution	
Axis	Aerodynamic x-axis	
Number of Points	10	10
Initial Position [m]	0	0
Final Position [m]	1	1
Select Action	Visualize Line	
Select Variable	Infinite Surface Formulations	
☐ Flip Clip Normal		

Figure 38: Epsilon Integration Tool properties panel

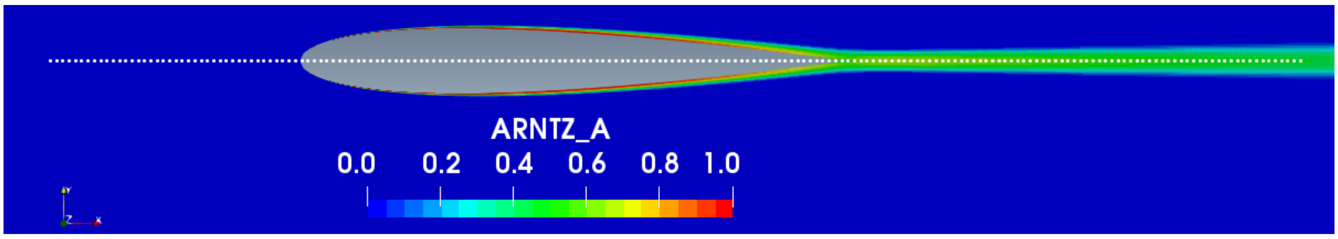


Figure 39: Points along the integration line

This is also the best moment to correct the position of the line ends as well as the point density. Just modify those parameters in the properties panel and click again on "Apply". This will update the visualization.

Now you can go to the second step: the visualization of the survey plane that will be used for the integration of the aerodynamic parameters. This is achieved by changing "Select Action" from "Visualize Line" to "Visualize Plane". Then click on "Apply" and the survey plane will be displayed roughly at the center point along the integration line as shown in Fig. 40 (make sure you select the visualization mode back to "surface" again). This step is required to visually confirm that the survey plane is normal to the upstream flow direction.

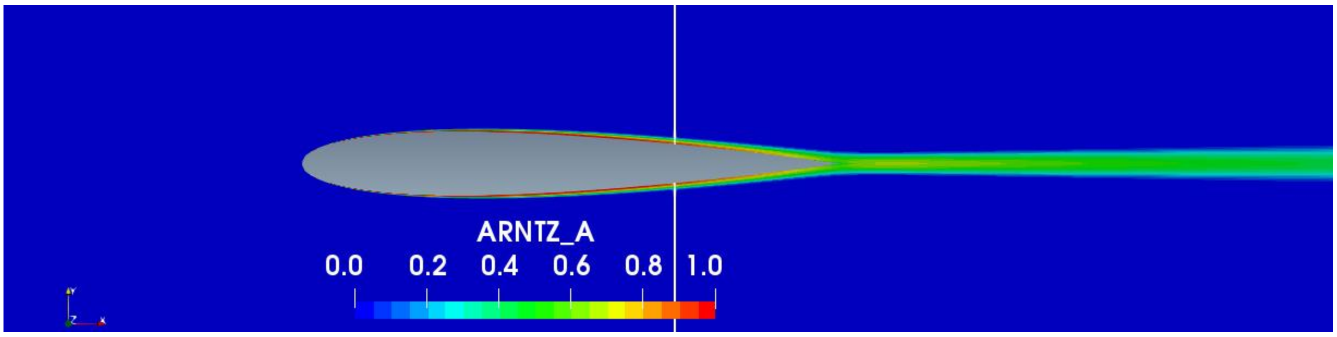


Figure 40: Survey plane visualization

The final step is the actual integration of the parameters on the survey plane for each plane position defined by the points of the integration line: change "Set Action" to "Perform Automatic Integration" and click on "Apply". Few seconds later an output line will be created which contains the integrated values at each point. The time required to integrate the values depends on the amount of plane positions (line points) and the CFD grid density. It is advised to start with 100 points for 2D cases in order to have a first idea of the output, then increase it to the desired value.

There are three ways to visualize the output of this integrated data in 2D cases. The first-one is to visualize the distribution of a given integrated parameter on the top of the field as shown in Fig. 41. This is achieved by using "Warp By Scalar" filter. This method is suited for a qualitative visualization.

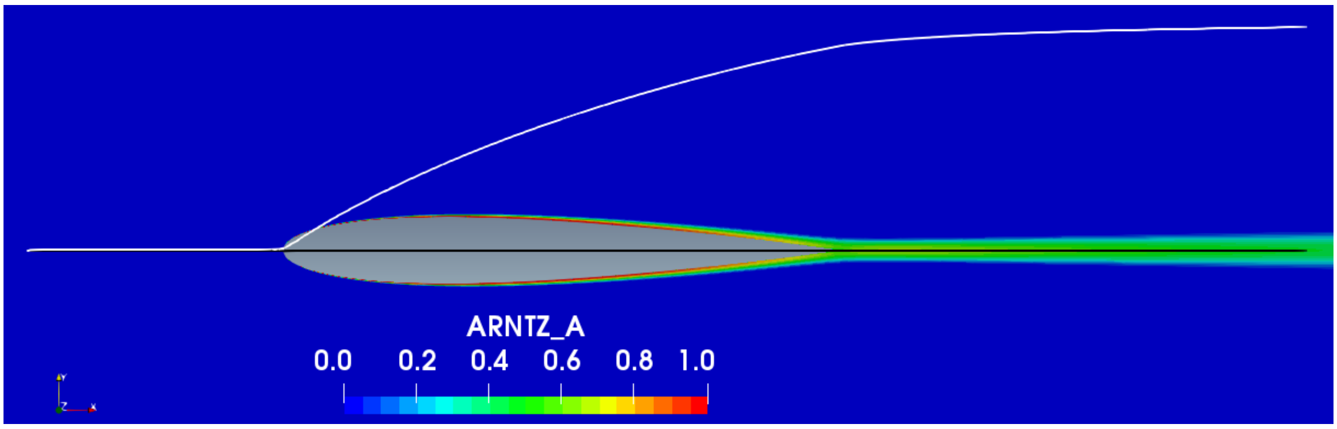


Figure 41: Visualization of the integrated "ARNTZ_A" field (white line) along the integration line (black)

If a quantitative visualization is required, select the "Epsilon Integration Tool" in the pipeline browser and apply the "Plot Data" filter:

Filters → Alphabetical → Plot Data

The setting of the filter is identical to "Plot On Intersection Curves" filter. The result is shown in Fig. 42.

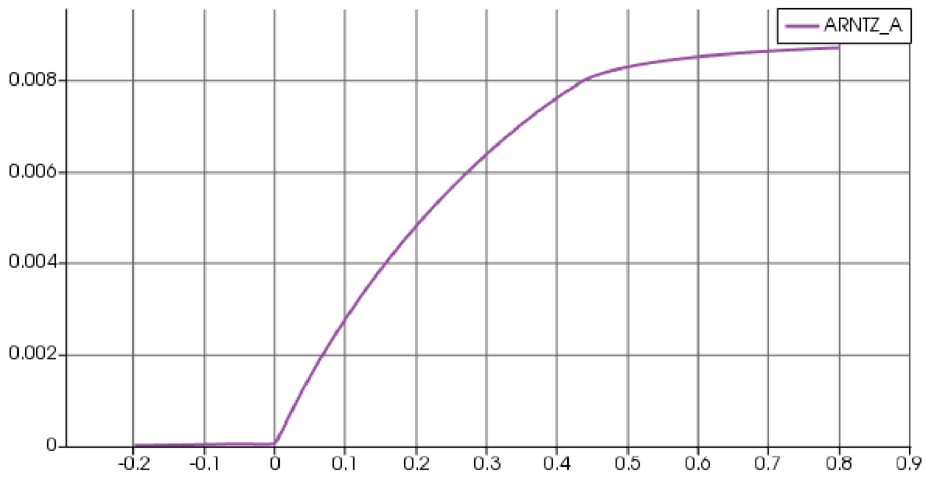
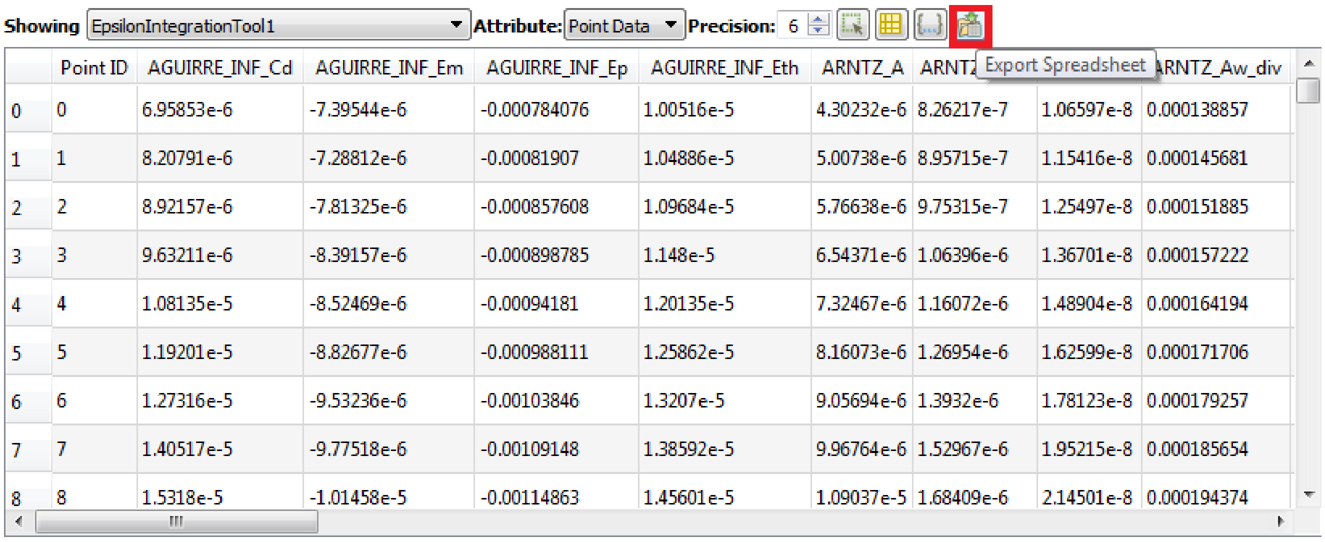


Figure 42: Plot of the integrated "ARNTZ_A" field along the integration line

The third way to visualize data is to export it to your favorite code (Python, Matlab). This is achieved by using the "Spread Sheet View": first select "Epsilon Integration Tool" in the pipeline browser. Then create a new layout of type "Spread Sheet View". This gives you a table with all the integrated data that can be exported into a CSV file by clicking on the button highlighted in red in Figure 43.



Showing EpsilonIntegrationTool1 **Attribute:** Point Data **Precision:** 6

	Point ID	AGUIRRE_INF_Cd	AGUIRRE_INF_Em	AGUIRRE_INF_Ep	AGUIRRE_INF_Eth	ARNTZ_A	ARNTZ		ARNTZ_Aw_div
0	0	6.95853e-6	-7.39544e-6	-0.000784076	1.00516e-5	4.30232e-6	8.26217e-7	1.06597e-8	0.000138857
1	1	8.20791e-6	-7.28812e-6	-0.00081907	1.04886e-5	5.00738e-6	8.95715e-7	1.15416e-8	0.000145681
2	2	8.92157e-6	-7.81325e-6	-0.000857608	1.09684e-5	5.76638e-6	9.75315e-7	1.25497e-8	0.000151885
3	3	9.63211e-6	-8.39157e-6	-0.000898785	1.148e-5	6.54371e-6	1.06396e-6	1.36701e-8	0.000157222
4	4	1.08135e-5	-8.52469e-6	-0.00094181	1.20135e-5	7.32467e-6	1.16072e-6	1.48904e-8	0.000164194
5	5	1.19201e-5	-8.82677e-6	-0.000988111	1.25862e-5	8.16073e-6	1.26954e-6	1.62599e-8	0.000171706
6	6	1.27316e-5	-9.53236e-6	-0.00103846	1.3207e-5	9.05694e-6	1.3932e-6	1.78123e-8	0.000179257
7	7	1.40517e-5	-9.77518e-6	-0.00109148	1.38592e-5	9.96764e-6	1.52967e-6	1.95215e-8	0.000185654
8	8	1.5318e-5	-1.01458e-5	-0.00114863	1.45601e-5	1.09037e-5	1.68409e-6	2.14501e-8	0.000194374

Figure 43: Output table from Epsilon Integration Tool

There are two options of the "Epsilon integration Tool" that haven't been described yet: "Flip Clip Normal" and "Select Variable". "Flip Clip Normal" is reserved for 3D applications, however, "Select variable" is used in both, 2D and 3D cases.

"Select variable" is a drop down list with seven options. The user must select the proper option depending on the intended application as follows:

• **Infinite surface formulations**: it can only be used when the survey plane slices the entire CFD domain (as shown in Figure 40).

• **Wake surface formulations**: it can only be used if the CFD domain has been clipped prior to the application of the Epsilon Integration Tool, in order to integrate data only inside the boundary layer and wake regions as shown in Fig. 44.
• **Volume formulations**: it must be selected to integrate only the volume formulations (e.g., the local viscous anergy rate). In this case, instead of using a survey plane, a clip filter is used to integrate the volume data upstream of the survey plane.
• **Body surface formulations**: this option is reserved for 3D cases.
• **Profile drag breakdown**: it must be used when the integration tool is applied to the Epsilon Profile Drag Breakdown filter.
• **Maskell lift/sideforce**: this option is reserved for 3D cases.
• **AERO parameters**: it integrates the classical aerodynamic parameters from Epsilon Aero Data.

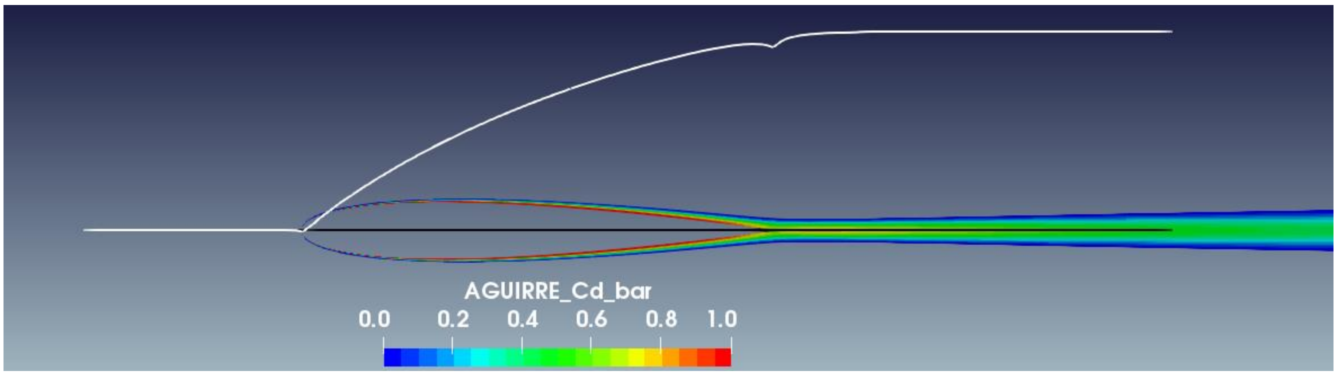


Figure 44: Output visualization example for the "Wake surface formulations" case with clipped domain

10 EPSILON NEAR FIELD

This tool allows calculating the near-field forces and moments acting upon the body surface. **This filter must be applied to an actual surface**, but not to edges or volume data. This means that if the 2D CFD mesh is already extruded (as the Open Foam cases), it is enough to extract the surface of the domain:

Filters → Alphabetical → Extract Surface

However, most of 2D CFD grids are planar (e.g., Fluent files). In these cases, the 2D body surface must be reconstructed by adding the following steps to the extracted surface:

Filters → Alphabetical → Feature Edges

Filters → Alphabetical → Linear Extrusion

"Feature Edges" filter detects the edges of the grid (Figure 45-left). This includes the actual body (e.g., an airfoil contour) but also the limits of the CFD domain. "Linear Extrusion" filter takes these contour lines and extrudes them in order to create an extruded surface perpendicular to the mesh plane (Figure 45-right).

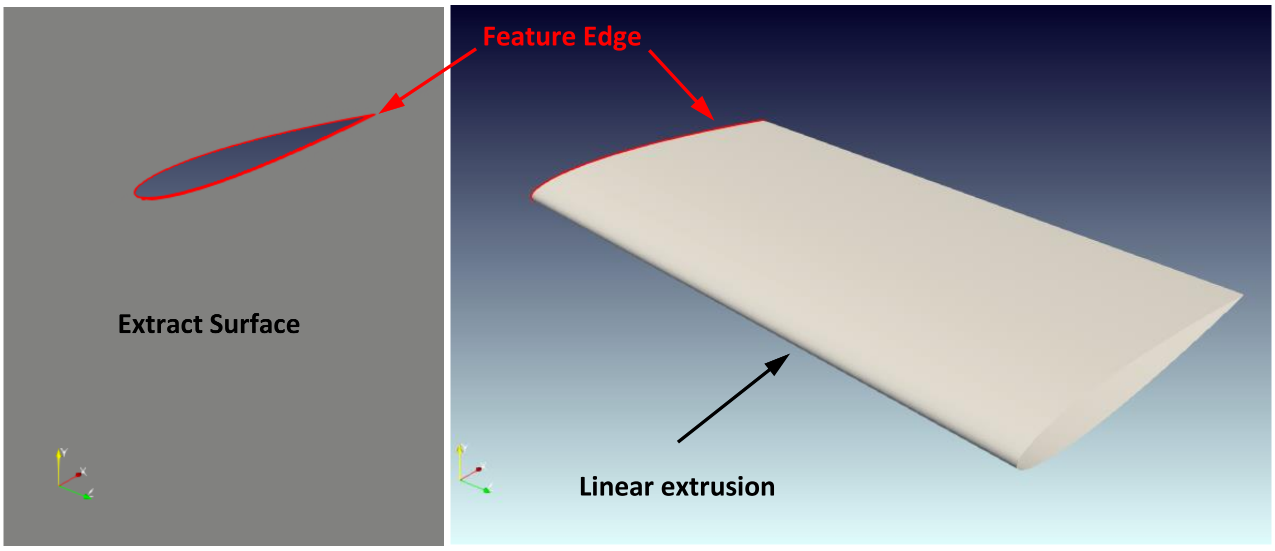


Figure 45: Extract surface and feature edge (left), followed by a linear extrusion (right)

Now we have the surface required to apply the "Epsilon Near Field" filter:

Filters → Epsilon → Epsilon Near Field → Apply

The parameters of the filter will be discussed later. First, an intermediate step is required. The fact is that the Linear Extrusion filter also extrudes the boundary domain contour (Fig. 46-left) and this must be taken away from the data. This is achieved by using a "Clip" filter of "Box" type (Fig. 46-center) that only retains the airfoil geometry (Fig. 46-right). The steps are the followings (See property panel details in Figure 47):

Filters → Alphabetical → Clip → Clip Type →Box
→ Show Box → Activate
→ Scale → Set a value smaller than 1 (the box must fit inside the boundaries)
→ Invert → Activate
→ Apply
→ Show Box → Deactivate (because this box annoys the visualization)

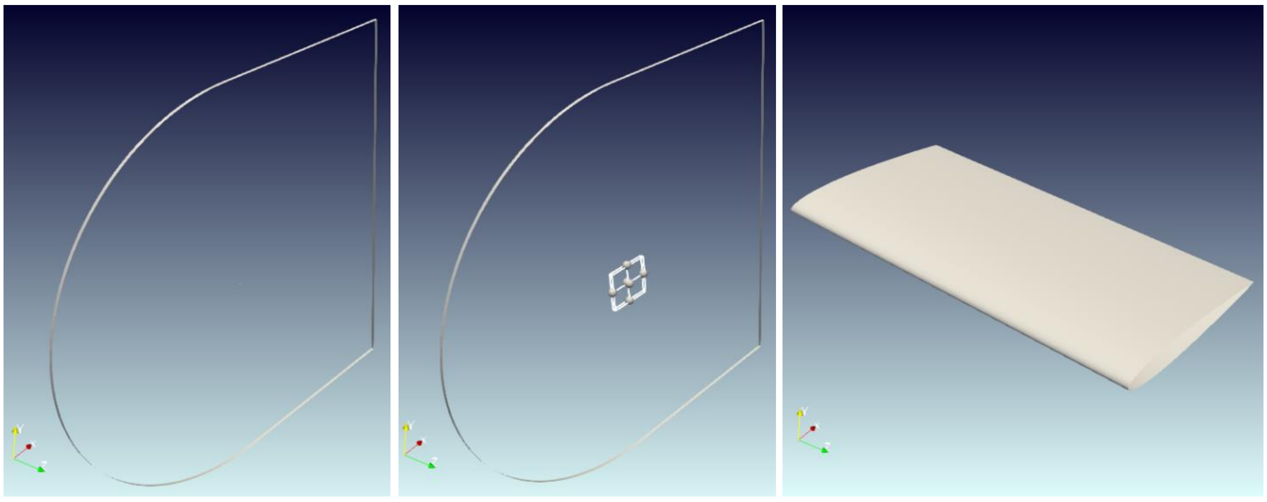
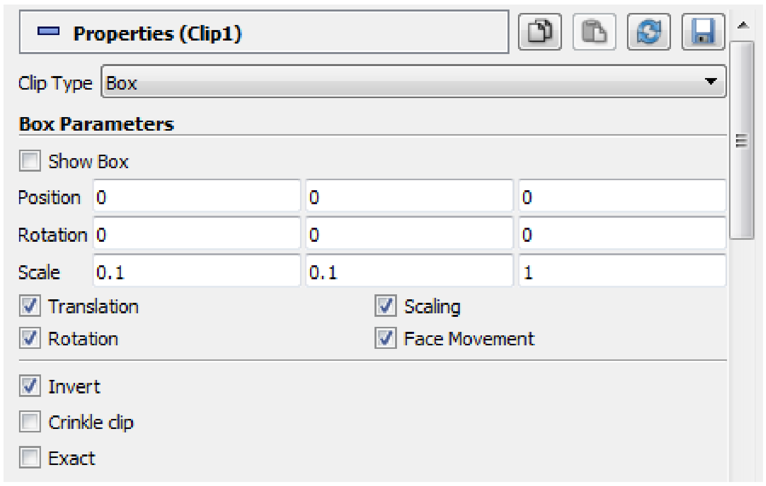


Figure 46: Extruded surface (left), clip domain (center) and surface retained (right)


Properties (Clip1)

Clip Type: Box

Box Parameters

[ ] Show Box

Position	0	0	0
Rotation	0	0	0
Scale	0.1	0.1	1

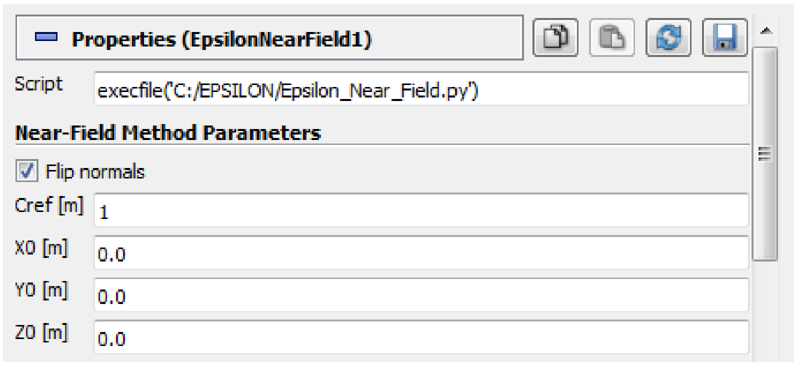
[✓] Translation [✓] Scaling
[✓] Rotation [✓] Face Movement

[✓] Invert
[ ] Crinkle clip
[ ] Exact

Figure 47: Clip filter properties panel

Now that the surface of the body is only present we can go back to the Epsilon Near Field filter and verify its setting as shown in Figure 48. The most important feature is to verify the normal direction. In fact, the proper calculation of lift, drag and moment values relies on **outward pointing normal**. Hence, the user must visualize the "Normals" variable along the aerodynamic Z-axis (Normals is a vector variable). This must display the normal colors as shown in Figure 49. If not, the option "Flip normal" must be activated.

On the other hand, "Cref [m]", "X0 [m]", "Y0 [m]" and "Z0 [m]" are the reference cord and reference center coordinates used for the moment calculation.


Properties (EpsilonNearField1)

Script: execfile('C:/EPSILON/Epsilon_Near_Field.py')

Near-Field Method Parameters

[✓] Flip normals

Cref [m]	1
X0 [m]	0.0
Y0 [m]	0.0
Z0 [m]	0.0

Figure 48: Epsilon Near Field *properties panel*

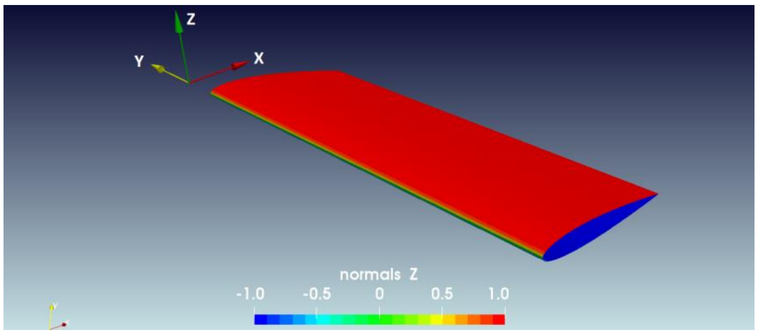


Figure 49: Correct normal direction

Finally, it is possible to integrate the output from this filter in order to obtain the lift, drag and moment acting on the 2D body.

11 PARAVIEW TIPS

11.1 Improving legend labels

The scale range numbers in the label of any Parameter are displayed by default with scientific notation (Figure 50-left). If you want to use standard numbers instead (Figure 50-right), you have to open the "Edit Color Legend Properties", then click on the advanced properties button and then change the range label format: replace the "e" by an "f" (Figure 51). Also, the number of decimal can be changed by modifying the number in front of "e" (i.e., change "6.1" to "6.3" if 3 decimals are desired). Moreover, the same control panel allows changing the name of the variable displayed (it accepts **Latex** format).

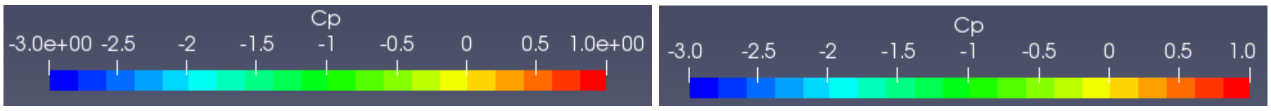
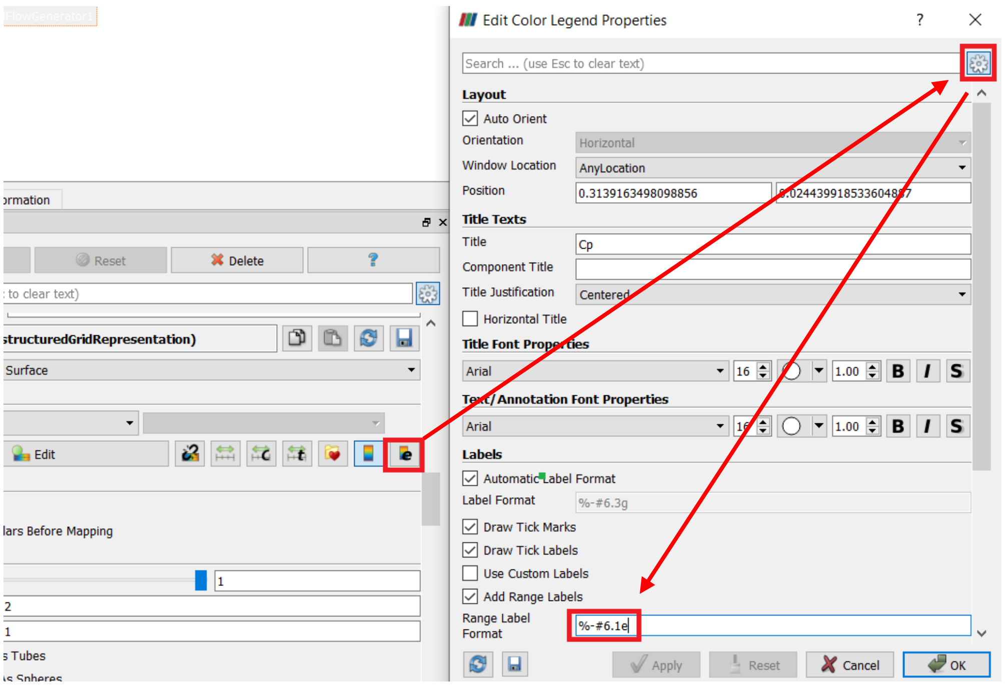


Figure 50: standard label (left) and new label type (right)


Reset
Delete
Edit
Surface
Edit Color Legend Properties
Search ... (use Esc to clear text)
Apply
Reset
Cancel
OK

Layout
Auto Orient ✓
Orientation: Horizontal
Window Location: AnyLocation
Position: 0.3139163498098856 | 0.02443991853360487
Title Texts
Title: Cp
Component Title:
Title Justification: Centered
Horizontal Title ☐
Title Font Properties: Arial 16 1.00
Text/Annotation Font Properties: Arial 16 1.00
Labels
Automatic Label Format ✓
Label Format: %-#6.3g
Draw Tick Marks ✓
Draw Tick Labels ✓
Use Custom Labels ☐
Add Range Labels ✓
Range Label Format: %-#6.1e

Figure 51: edit color legend properties

11.2 Paraview state file

Once the series of filters have been applied, it is possible to save the entire setup by using the state files:

File → Save State → Chose name of file

It can be loaded back into Paraview in this way:

File → Load State… → Load "FileName.pvsm" file

CHAPTER 5: ANALYSIS OF 3D CFD DATA

The step-by-step procedure for a complete 3D analysis is shown in Figure 52.

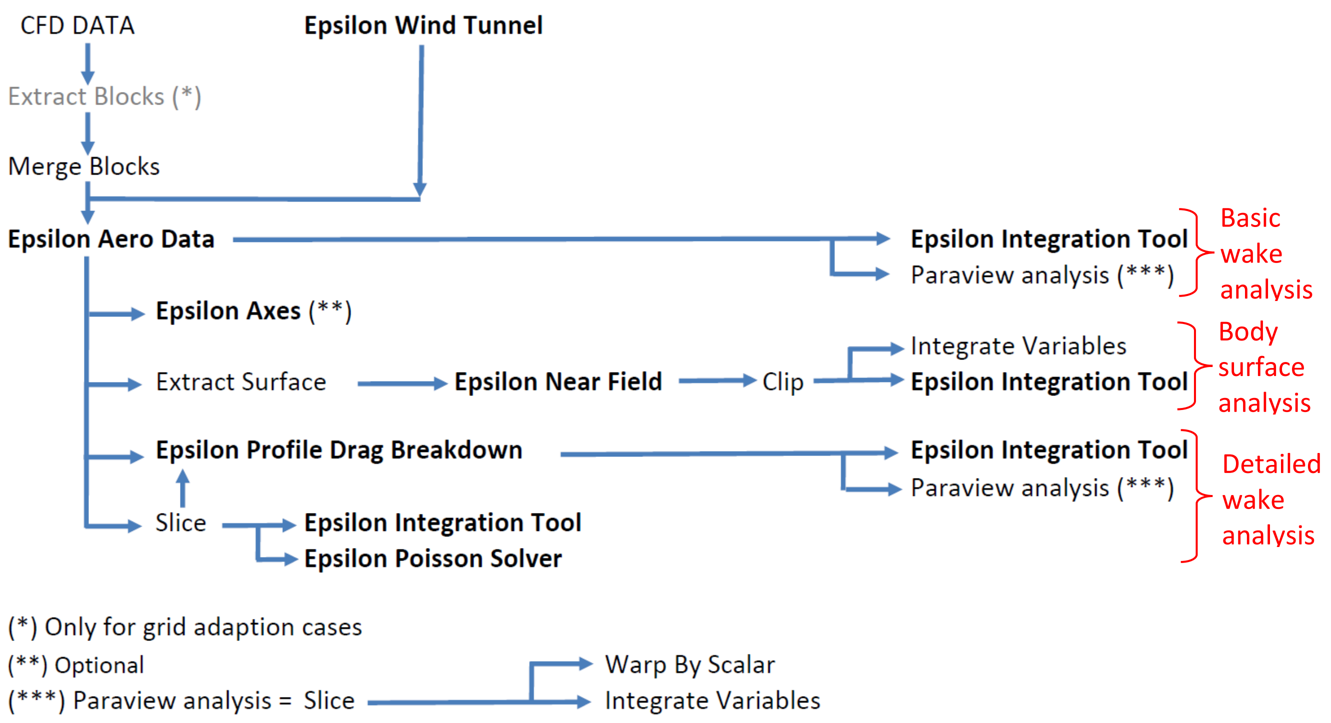


Figure 52: setup procedure for 3D cases

Since we already discussed most of the analysis tools in the 2D case, here we will present only the remaining tools as wells the idea behind this new setup procedure for 3D cases. In general, a complete 3D analysis is made in three-stages:

• Body surface analysis.
• Basic wake analysis.
• Detailed wake analysis.

1 BODY SURFACE ANALYSIS

This analysis is performed when the **near-field forces and moments** are required and also when the near-field lift and drag **distributions** along the span are needed.

1.1 Extract surface/ Clip

The output data from Epsilon Aero Data do not allow separating the surface by name ID. Thus, in order to separate the surface of the body from the symmetry plane and the domain boundary surfaces it is necessary to manipulate a little bit the geometry data.

Firstly it is necessary to extract the surface from the entire data set (which contains the volume data as well as the surface data). Select the "Epsilon Aero Data" filter on the Pipeline Browser, and then apply this filter:

Filters → Alphabetical → Extract Surface

At this stage, the 3D visualization will not change as it is shown in the Figure 53-left (it just deletes the internal points data). So, we need to find a way to hide the surfaces surrounding the body. This can be easily achieved by using a clip filter to cut away the unwanted surfaces. Thus, select the "Extract Surface" filter from the Pipeline browser and then apply the "Clip" filter:

Filters → Alphabetical → Clip

The "Clip Type" must be a box: we will use this box to surround only the body and to clip out the external geometry. By default, this box encloses the entire geometry (Figure 53-center). Thus, this box must be scaled down in order to only enclose the body (Figure 53-right). This is made by setting a value smaller than 1 in the "Scale" option (Figure 55). Also, a very small translation of the box is required in order to exclude the symmetry plane from the selection. Finally, by selecting the option "Inside Out" we retain only the geometry enclosed by this box, as it is shown in the Figure 54–left. Then, by deactivating the "Show Box" option we get only the body (Figure 54 – right). The resulting geometry can be colored by all the Epsilon Aero Data parameters and even the surface flow patter can be seen by using the surface LIC as mentioned in the 2D case.

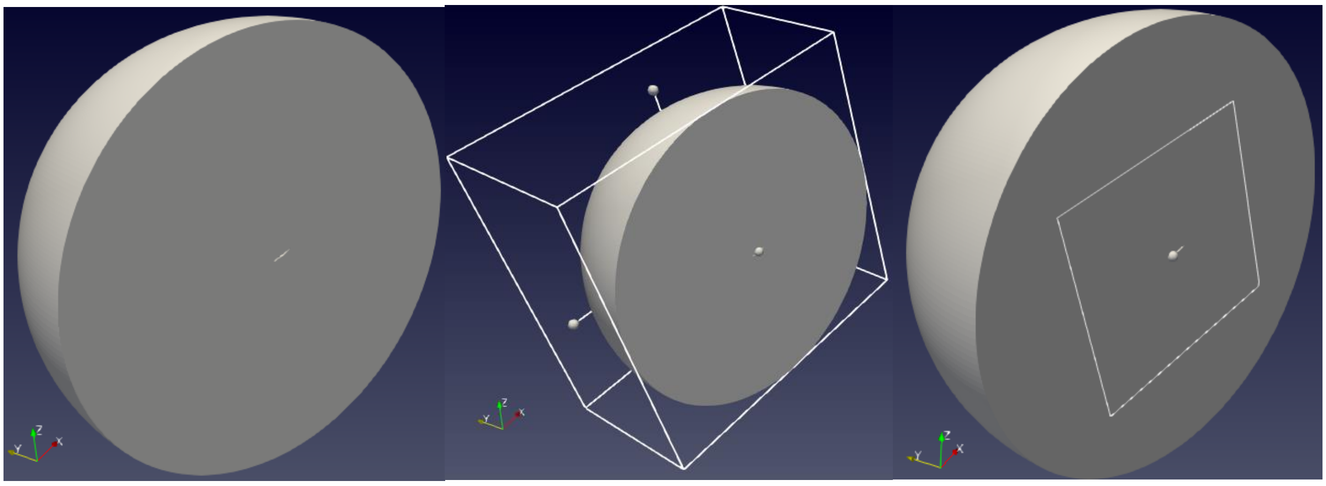


Figure 53: clip procedure

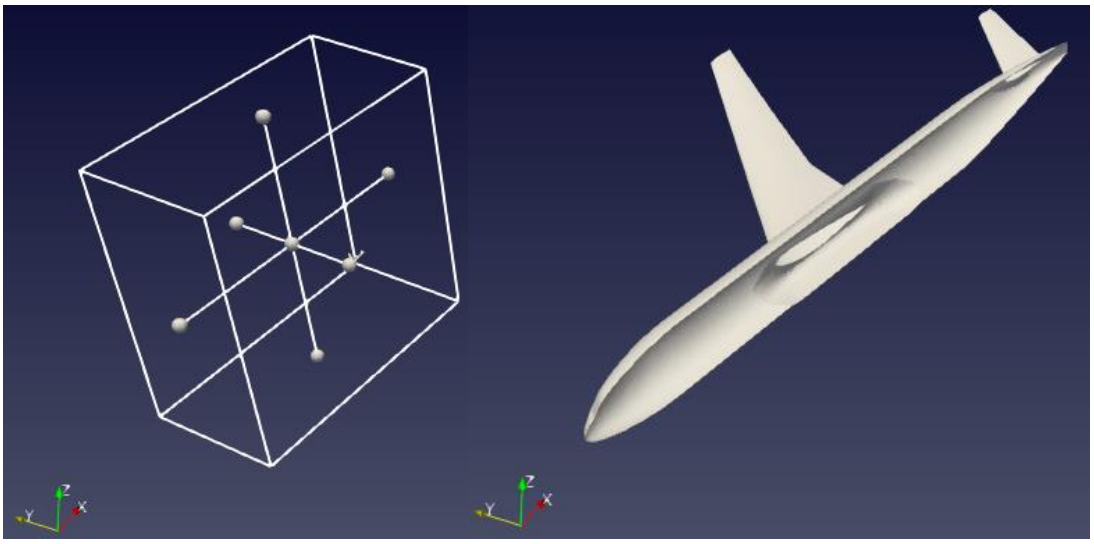
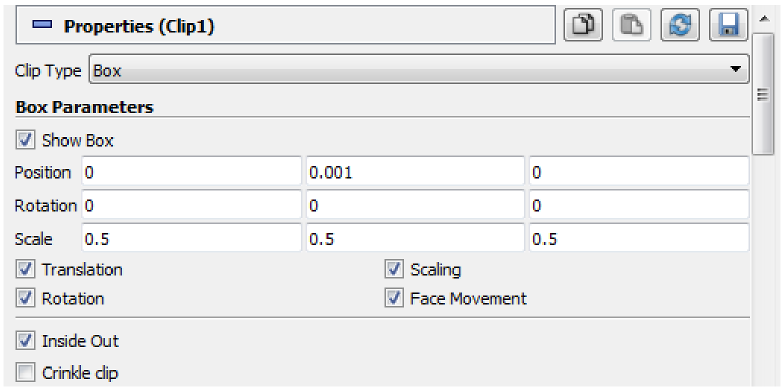


Figure 54: clip procedure (continuation)


Properties (Clip1)

Clip Type: Box

Box Parameters

- [✓] Show Box

Position	0	0.001	0
Rotation	0	0	0
Scale	0.5	0.5	0.5

- [✓] Translation
- [✓] Scaling
- [✓] Rotation
- [✓] Face Movement
- [✓] Inside Out
- [ ] Crinkle clip

Figure 55: clip properties panel

1.2 Epsilon Near Field

This filter is placed between the "Extract Surface" and "Clip" filters as it was explained in the 2D example. Once the domain is clipped by retaining only the geometry, it is possible to obtain lift, drag and moments by integrating data on the entire body surface (use "Integrate variables" filter). The resulting integrated values will be displayed in a "Spread Sheet" window.

1.3 Epsilon Integration Tool

This filter can be applied to the output of the "Extract Surface + Epsilon Near Field + Clip" sequence, in order to obtain the spanwise lift or drag distribution by the near-field method. The setup is identical to the 2D case. The only exception is that the variable "**Body Surface Formulations**" must be selected for the integral. The output can be visualized with "warp by scalars" or "plot data" filters as usual.

2 BASIC WAKE ANALYSIS

This analysis provides a quick assessment of the **far-field, Lamb vector and exergy** parameters by analyzing data on the survey plane.

2.1 Slice CFD domain

The survey plane can be created by using the "Slice" filter. It is a standard Paraview tool to create a plane crossing the entire CFD domain into which the CFD data is interpolated. Firstly, select "Epsilon Aero Data" from the Pipeline browser and then apply the following filter:

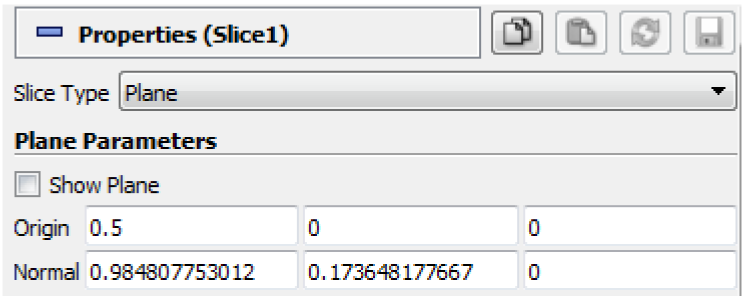
Filters → Alphabetical → Slice

It is recommended to deactivate the "Show Plane" option on the properties panel (Figure 57) because one may change inadvertently the plane position with the mouse. The plane axial position along the X-axis must be defined by the user but its inclination must be settled up by using the output messages from Epsilon Aero Data:

```
# NORMAL DIRECTION TO USE FOR SLICE PLANE OR INTERSECTION LINE DEFINITION
Normal=  0.984807753012    0.173648177667    -0.0
```

Figure 56: Epsilon Aero Data output message

These printed normal vector directions must be introduced on the plane's normal definition on the properties panel (Figure 57).



Properties (Slice1)

Slice Type: Plane

Plane Parameters

Show Plane

Origin	0.5	0	0
Normal	0.984807753012	0.173648177667	0

Figure 57: Slice properties panel

The resulting slice plane (survey plane) can be colored by any Epsilon Aero Data parameter (Figure 58), and even it allows showing the surface LIC flow pattern.

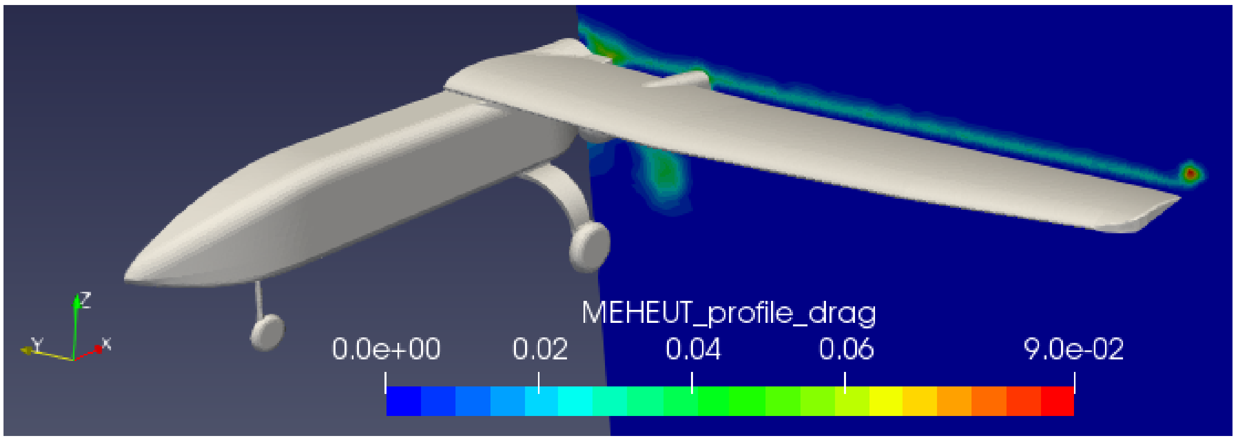


Figure 58: survey plane obtained by "Slice" filter

2.2 Clip wake region

WTT formulations require data on the wake region only. Thus, the survey plane must be clipped before integrating its data (Figure 59). This visualization of the data on the clipped plane can be improved by using "Warp By Scalar" filter or "Plot On Intersection Curves" (Figure 60).

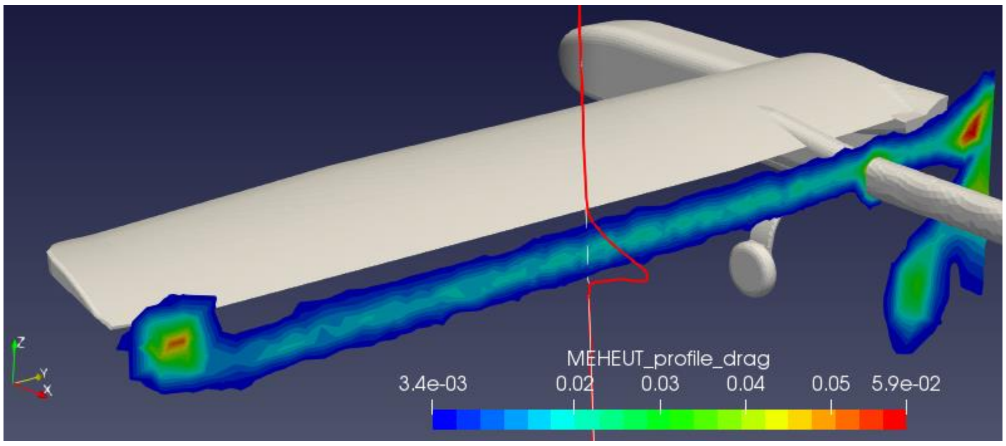


Figure 59: drag density distribution at certain station

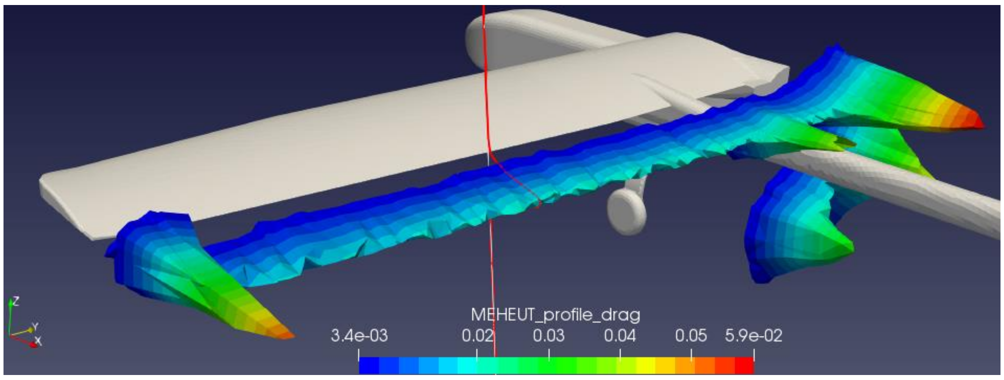


Figure 60: drag density at the survey plane warped by its magnitude

2.3 Integrate variables

For 3D cases, the far-field and exergy parameters displayed on the slice plane are the corresponding "densities" (i.e., the integrand of the far-field and exergy equations), so this plane data can be directly integrated by using the "Integrate Variables" standard filter.

Important: The user must keep in mind that the integrated values of the integrands give the total value of these parameters (For example, the integration of Betz Profile Drag density will give the actual value of profile drag). However, the integration of wave anergy density, thermal anergy density or any other volume-based formulation is **meaningless**. This is because these integrands require a volume integration to give the thermal and wave anergy (so, its plane integration is meaningless or at least, it can be interpreted as the local rate of generation of anergy at the plane position). It is important that the user keep always in mind the equations already presented in Chapter 1.

2.4 Epsilon integration tool

This integration can also be made automatically for several plane positions by using "Epsilon Integration Tool" already described in the analysis of 2D CFD data. The parametrization for 3D cases is identical to the 2D cases and the sweep axis must be the mesh or aerodynamic x-axis as shown in Figure 61.

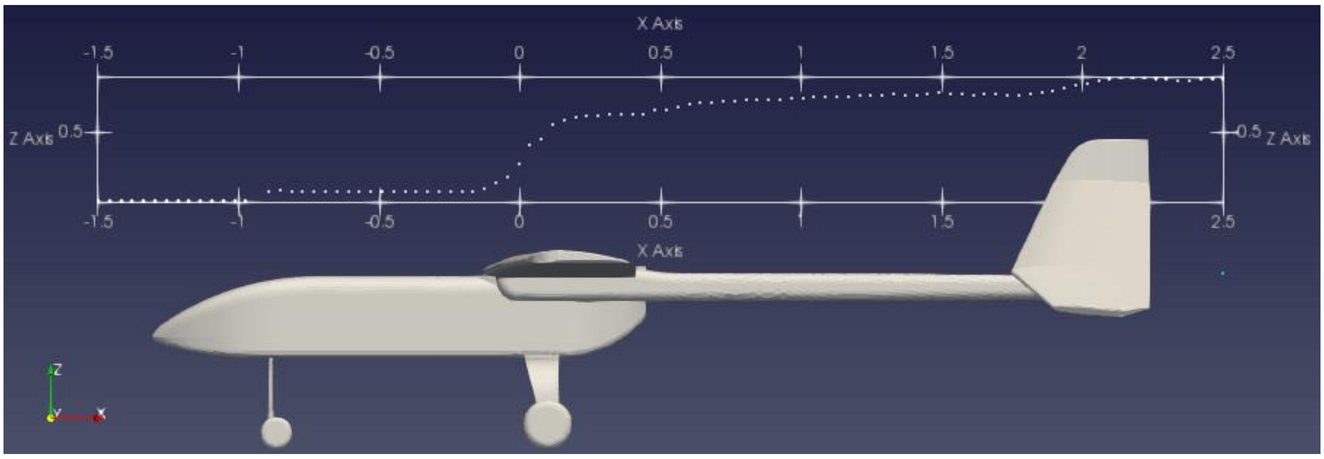


Figure 61: automatic integration tool output example showing drag value along the body

3D cases allow creating a different type of visualization: the 3D to 2D mapping (Figure 62).

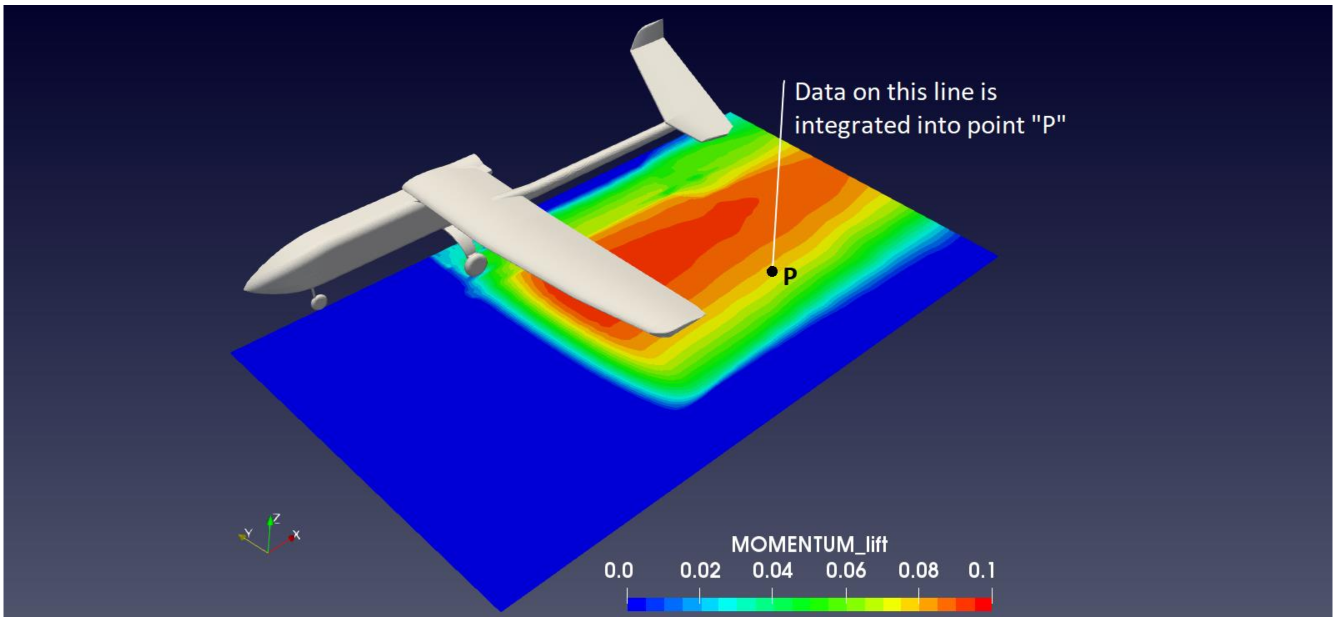


Figure 62: example of 3D volume integration into a plane

This is achieved by using "Epsilon integration tool" with the "Output type" as "**3D to 2D mapping**". Note that this time the "Axis" option to be selected is the normal direction of the integration plane mesh shown in figure 62. Moreover, for this option it is required to define number of points and start/end of the surface mesh along its two directions: this is made by filling both columns of the input table. Also, in "Select action" the "Visualize line" option is no longer needed. Instead, you should visualize the mesh plane with "Visualize plane" option. Finally, the selected variable to integrate will depend on the formulation you want to visualize.

3 DETAILED WAKE ANALYSIS

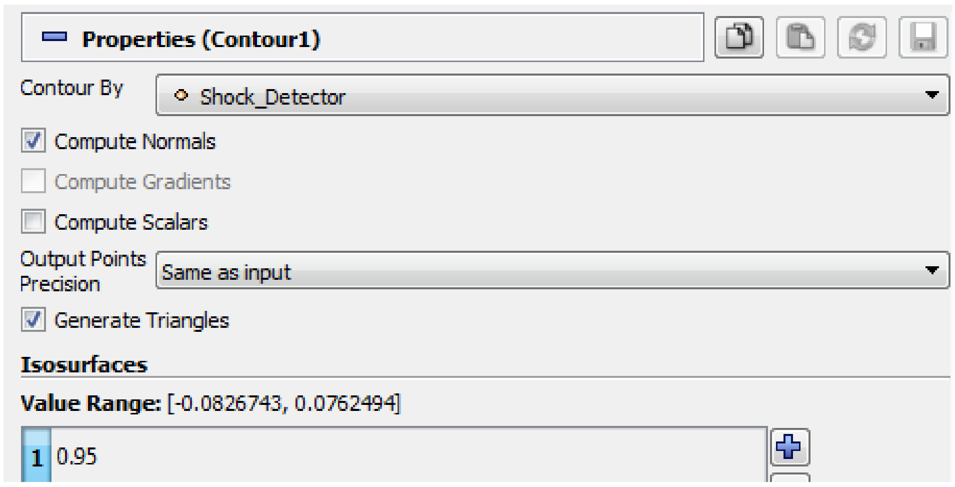
For most of applications, the basic wake analysis is enough. However, a detailed wake analysis will provide a deeper insight into the physics.

3.1 Profile drag breakdown

The first step is to visualize the shockwave: a Contour filter must be applied directly to the Epsilon Aero Data output:

Filters → Alphabetical → Contour

In its properties panel (Figure 63), the user must select the "**Shock Detector**" parameter on the "Contour By" drop-down list. Then, it is necessary to indicate its contour value in the "Value Range" option. This number must be settled to 0.95. The resulting visualization can be seen on the Figure 64:


Properties (Contour1)

Contour By: Shock_Detector

☑ Compute Normals

☐ Compute Gradients

☐ Compute Scalars

Output Points Precision: Same as input

☑ Generate Triangles

Isosurfaces

Value Range: [-0.0826743, 0.0762494]

1 0.95

Figure 63: contour properties panel

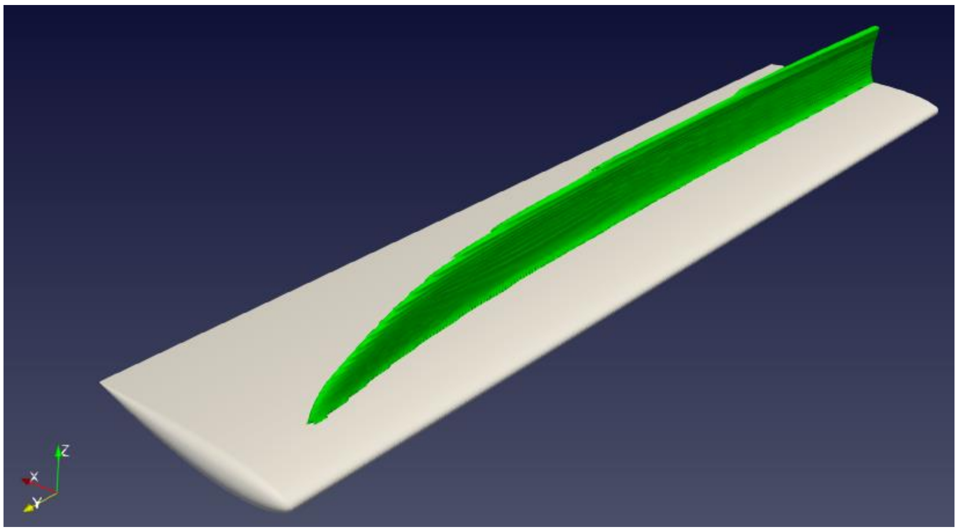


Figure 64: shockwave visualization

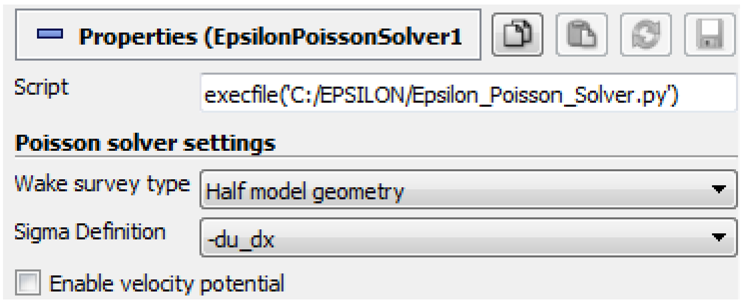
Then, the "Epsilon Profile Drag Breakdown" filter must be applied either to the entire 3D data (i.e., the output from Epsilon Aero Data) or a slice (i.e., the survey plane). It is strongly **recommended** to start with the survey plane because it is faster to properly setup the drag breakdown parameters as it was explained in the 2D case. Once these parameters are OK, one can think about breaking down the entire CFD domain.

3.2 Induced drag with Epsilon Poisson Solver

The induced drag by WTT formulation is obtained by solving the Poisson equation:

Filters → Alphabetical → Epsilon Poisson Solver

Before clicking on the "Apply" button, some parameters must be settled up. As usual, the "Script" file path must not be modified if the recommended installation procedure has been followed (Figure 65).



Properties (EpsilonPoissonSolver1

Script: execfile('C:/EPSILON/Epsilon_Poisson_Solver.py')

Poisson solver settings

Wake survey type: Half model geometry

Sigma Definition: -du_dx

☐ Enable velocity potential

Figure 65: Epsilon Poisson Solver properties panel

The "wake survey type" drop down list is a parameter only valid for the case of Green functions: it allows the specification of the characteristics of data being analyzed. By default it considers that the CFD domain corresponds to a half-body ("Half-model geometry"). However a full-body is also available. This will affect the type of equation to be used (See the theoretical guide in Chapter 1). In fact, the "Upwards Direction" parameter explained before, allows the specification of the upwards mesh direction in order to know how to do the data reflection for the case of half-model geometry.

The "Sigma definition" drop down list allows choosing the way the "σ" parameter is calculated. The default approach takes into consideration the axial velocity derivative, while the alternative is to consider the transverse velocity gradients. Theoretically, both methods provide the same result. However, the use of the transverse velocity gradients is prone to some local numerical errors (spots of unphysical "σ" values and/or distorted regions) which are mesh refinement sensitive. It can be solved by simply refining the survey mesh.

The "Enable velocity potential" checkbox allows the user to decide whether or not to solve the Poisson equation for the velocity potential. Poisson equation for the stream function is always solved by default. However, the user must indicate if he wants to solve the Poisson equation for the potential velocity function: if this checkbox is not activated, then the Maskell equation terms containing the "Φ" symbol will be discarded.

After clicking on the Apply button, the user must wait some time while the Python file solves the Poisson equation. The resulting stream function will show a circle-like pattern around the high vorticity regions as shown in Figure 66.

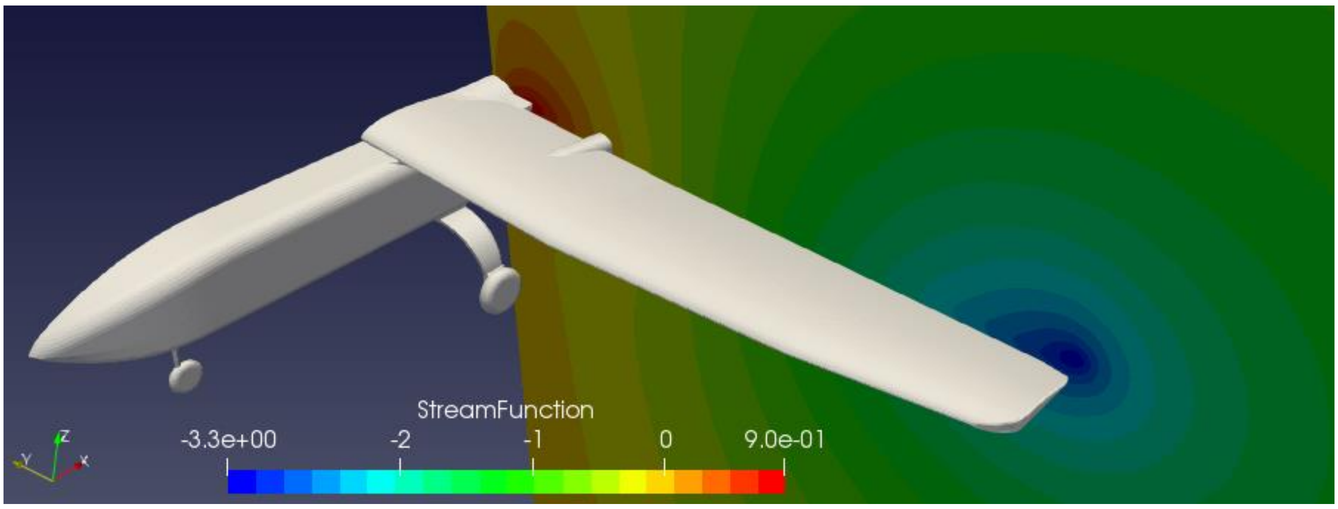


Figure 66: stream function example

The required time to solve the Poisson equation depends on the number of points at the survey plane mesh (this is the dominant factor) and the amount of equations to be solved (If the velocity potential is activated). An order of magnitude of this required time is shown in Figure 67 for a case without the velocity potential calculations.

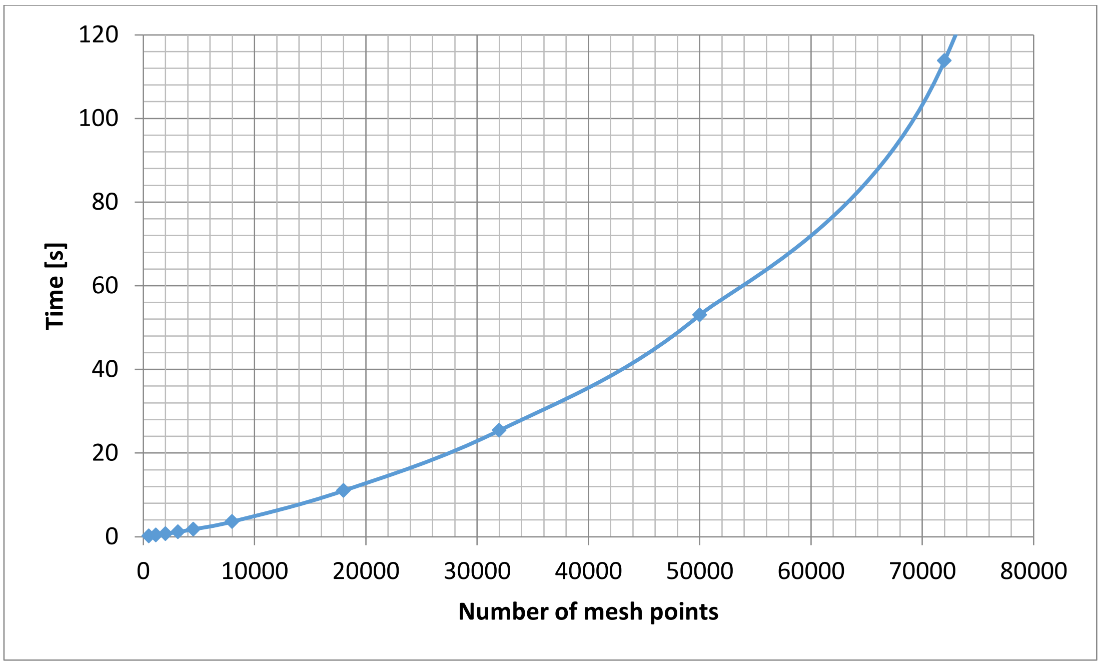


Figure 67: time required to solve the Poisson equation

In order to know if the calculation has finished, an output message indicates the total calculation time required (This information is printed for any method).

```
Starting Poisson solver…
Calculation time 67.9409999847 Seconds ( 1.13234999975 minutes)
```

Once executed the Epsilon Poisson Solver, the stream function, potential function, vortex drag and other related parameters are accessible at the output. If needed, this data can be integrated by using the "Integrate Variables" filter, in order to have the total value of the vortex drag.

3.3 Spanwise distributions

Epsilon Integration Tool can be used to produce a typical aerodynamic analysis plot: spanwise distribution of different variables.

3.3.1 Distribution of local variables

The first step is to create a slice plane either full size (infinite survey plane) or clipped (wake survey plane). Then, "Epsilon Integration Tool" is applied to this plane in order to integrate data along the span direction by following the same procedure as explained for 2D cases. The typical result is shown in Figure 68.

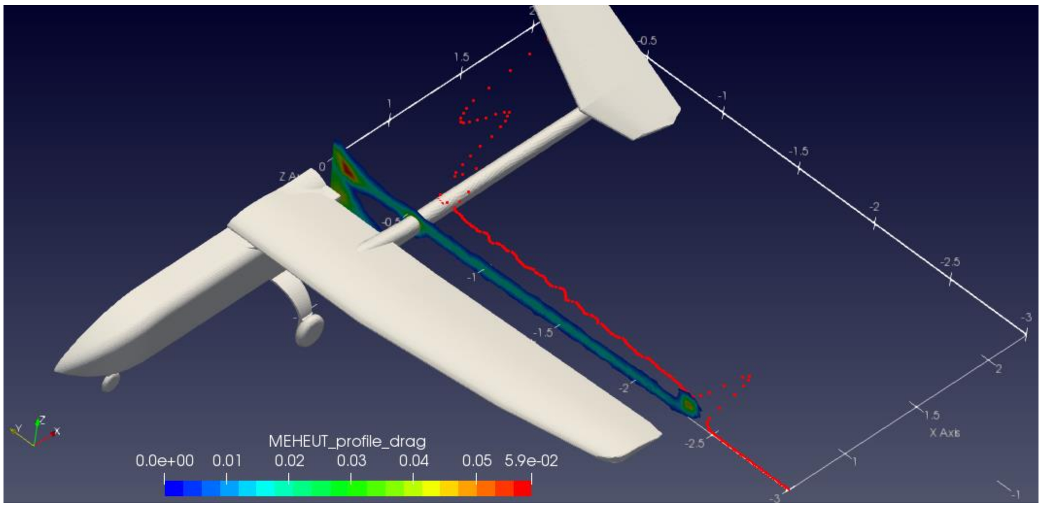


Figure 68: spanwise drag distribution (red) along with its drag density on the survey plane

3.3.2 Lift distribution

In order to obtain the spanwise lift distribution by the Maskell method a similar procedure must be followed. The only exception is that the user must select "**Maskell lift/sideforce**" in the "Select Variable" drop down list. If required, the option "**Flip Clip Normal**" must be activated. In order to understand how this particular plot is created inside the filter and when is necessary to invert the clip normal, the internal functioning of the filter is explained as follows.

Internally, the filter takes the entire survey plane data as an input. Then it defines a certain spanwise station and finally it only integrates the survey plane data lying on the outside part of this station (i.e., from the desired station towards the wing tip). In fact, since the Maskell lift density distribution contains the axial vorticity, its related field must be interpreted as the local change in lift distribution (Figure 69).

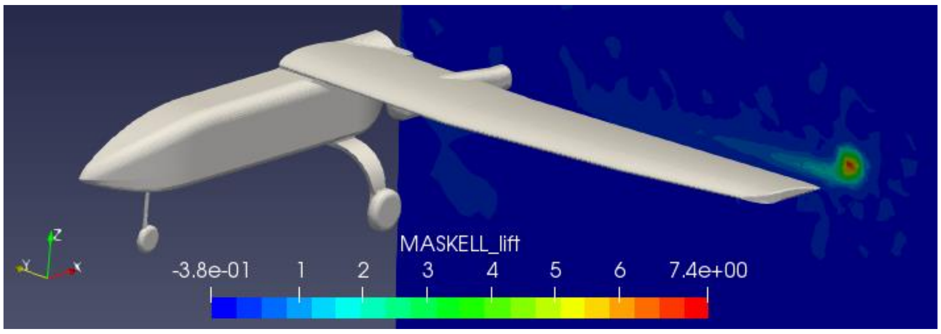


Figure 69: Maskell lift density

Thus, in order to know the actual lift value at certain station, the integration of this lift density must be carried out on the area outside of this station (Figure 70). That's why the survey plane data is clipped by the integration tool. This also explains why sometimes will be necessary to activate the "Flip Clip Normal" option: in some cases the normal may be pointing in the wrong direction which leads to the integration of the wrong side of the clipped plane.

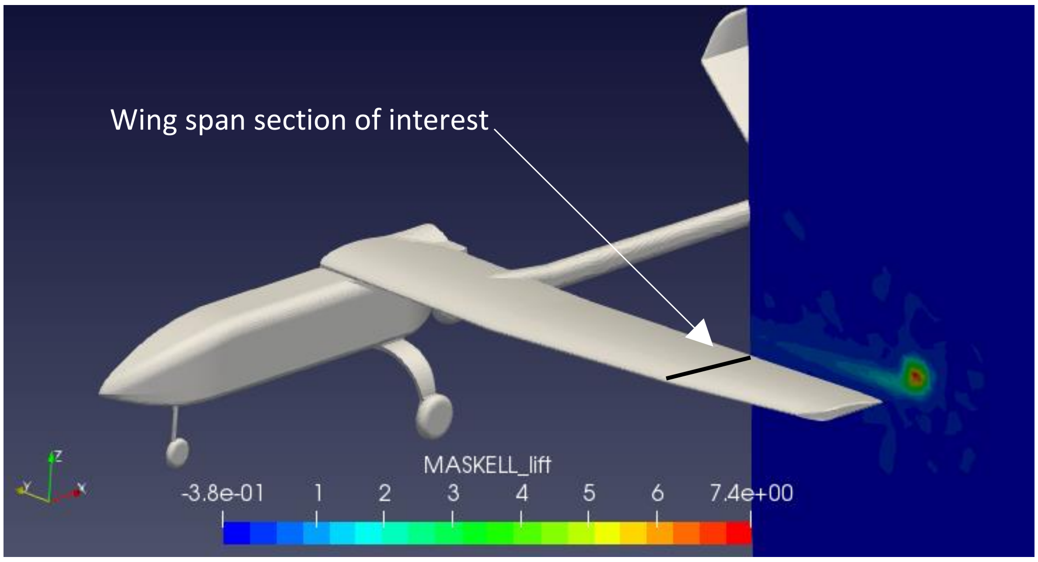


Figure 70: Maskell lift density clipped

Finally, this clip filter is animated in order to sweep it along the entire span of the wing. At each position, data is integrated and stored into a vtk polyline output. The resulting spanwise lift distribution can be obtained by using "Warp By Scalar" filter as it is shown in the Figure 71:

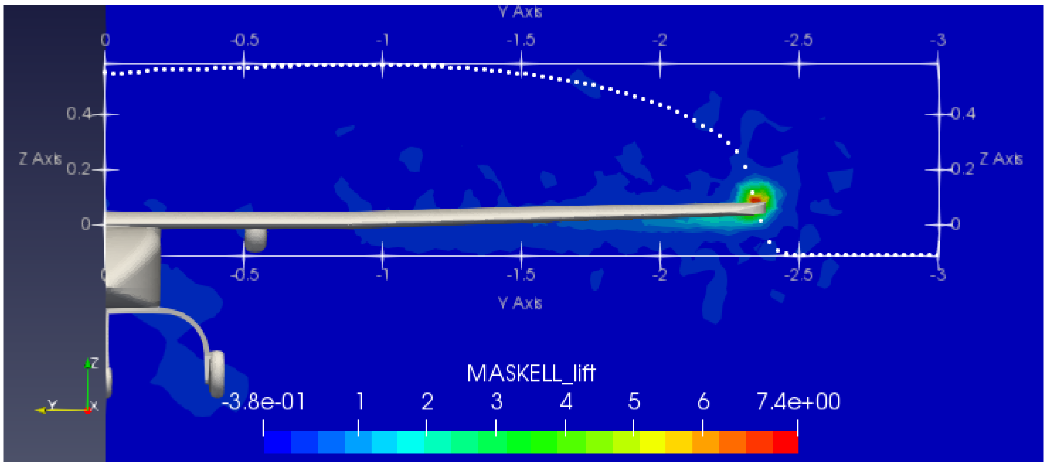


Figure 71: spanwise lift distribution example, along with the Maskell lift density

APPENDIX 1

In this appendix there is a detail of all the output arrays for any Epsilon filter.

1 EPSILON AERO DATA

The list below follows an alphabetical order and it is also the same order shown in the Paraview's GUI. For each variable, an ID is provided, which correspond to the number of equation presented on the Chapter 1. Moreover, a detail of the equation (or a term of its equation) coded into each variable is also provided. This allows the user to know explicitly the meaning of each variable. It must be noticed that all the variables represents the INTEGRAND of the equations of the Chapter 1. They DO NOT provide the integrated value of those equations.

VARIABLE NAME	ID	EQUATION CODED IN THE VARIABLE
AERO_Cp	146	$\frac{P_s - P_{s_0}}{\frac{1}{2}\rho_0 U_0{}^2}$
AERO_Cp_gradient	---	∇C_p
AERO_Helicity	---	$\frac{\vec{V}.\vec{\omega}}{\lvert\vec{V}.\vec{\omega}\rvert}$
AERO_Hs	23	$c_p T s$
AERO_Ht	24	$h_s + \frac{1}{2} V^2$
AERO_Ht-Ht0	---	$h_t - h_{t_0}$
AERO_Keff	102	$c_p * \left(\left(\frac{\mu}{0.7}\right) + \left(\frac{\mu_{turb}}{Pr}\right)\right)$
AERO_Lambda_2	136	eigenvalue($\boldsymbol{S}^2 + (\boldsymbol{J} - \boldsymbol{S})^2$)[:,1]
AERO_M	20	$\frac{V}{a}$
AERO_MU_EFF	104	MU_LAM + MU_TURB
AERO_MU_LAM	---	μ (CFD data)
AERO_MU_TURB	---	μ_t (CFD data)
AERO_Phi_eff	105	$\mu_{eff}\left[2\left(\frac{\partial u}{\partial x}\right)^2 + 2\left(\frac{\partial v}{\partial y}\right)^2 + 2\left(\frac{\partial w}{\partial z}\right)^2 + \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right)^2\right] - \frac{2}{3}\mu_{eff}\left[\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right]^2$
AERO_Ps	---	P_s (CFD data)
AERO_Pt	21	$P_s\left(1 + \frac{\gamma - 1}{2} M^2\right)^{\frac{\gamma}{\gamma - 1}}$
AERO_Pt/Pt0	---	$\frac{P_t}{P_{t_0}}$
AERO_Q_criterion	135	$\frac{1}{2}(\omega^2 - S^2)$
AERO_Rho	---	ρ(CFD data)

AERO_Shock_Detector	134	$\frac{\vec{V}.\overrightarrow{\nabla P_s}}{a\,\lvert\overrightarrow{\nabla P_s}\rvert}$
AERO_Strain	---	strain(AERO_Vector_Speed)
AERO_Stress(tau)	---	MU_EFF*strain
AERO_Ts	---	T_s (CFD data)
AERO_Tt	---	$T_s\left(1+\frac{\gamma-1}{2}M^2\right)$
AERO_V_gradient	---	$\nabla\vec{V}$
AERO_Vabs	---	$\sqrt{Vx^2+Vx^2+Vz^2}$
AERO_Vector_Speed	9	$Vx\,\vec{I}+Vy\,\vec{J}+\,Vz\,\vec{K}$ $\rightarrow$ *Mesh reference frame!*
AERO_Vorticity	13	$\xi\,\vec{\imath}+\eta\,\vec{\jmath}+\,\zeta\,\vec{k}$
AERO_e	25	c_vT_s
AERO_q	26	$\frac{1}{2}\,\rho\,V^2$
AERO_s-s0	---	$s-s_0$
AERO_u	8	$\bar{\bar{\boldsymbol{T}}}.\vec{\boldsymbol{V}}_{\boldsymbol{mesh}}[\,:,0]$
AERO_udelta	---	$u-U_0$
AERO_v	8	$\bar{\bar{\boldsymbol{T}}}.\vec{\boldsymbol{V}}_{\boldsymbol{mesh}}[\,:,1]$
AERO_w	8	$\bar{\bar{\boldsymbol{T}}}.\vec{\boldsymbol{V}}_{\boldsymbol{mesh}}[\,:,2]$
AGUIRRE_AERO_Hs_bar	119	c_pTs - $c_p\mathrm{T_s}^*$
AGUIRRE_AERO_Hs_star	119	$c_p\mathrm{T_s}^*$
AGUIRRE_AERO_Rho_bar	120	$\rho-\rho^*$
AGUIRRE_AERO_Rho_star	122	$\rho_0\left[1+\frac{\gamma-1}{2}\,{M_0}^2\left(1-\frac{V^{*2}}{{u_0}^2}\right)\right]^{\frac{1}{\gamma-1}}$
AGUIRRE_AERO_Ts_bar	119	$\mathrm{T_s}-T_s{}^*$
AGUIRRE_AERO_Ts_star	121	$\mathrm{T_{s0}}\left[1+\frac{\gamma-1}{2}\,\mathrm{M_0}^2\left(1-\frac{\mathrm{V}^{*2}}{\mathrm{u_0}^2}\right)\right]$
AGUIRRE_AERO_Vabs_star	117	$\mathrm{u_0}\sqrt{1-\frac{2}{(\gamma-1)\,\mathrm{M_0}^2}\left[\left(\frac{\mathrm{P_t}}{\mathrm{P_{t0}}}\right)^{\frac{(\gamma-1)}{\gamma}}*\zeta-1\right]}$
AGUIRRE_AERO_e_bar	---	$c_v\mathrm{T_s}-c_v\mathrm{T_s}^*$
AGUIRRE_AERO_e_star	---	$c_v\mathrm{T_s}^*$
AGUIRRE_AERO_u_bar	114	$\mathrm{u}-u^*$
AGUIRRE_AERO_u_star	---	AGUIRRE_Vabs_star*algs.cos(FlowTita)*algs.cos(FlowBeta)
AGUIRRE_AERO_v_bar	115	$\mathrm{v}-v^*$
AGUIRRE_AERO_v_star	---	AGUIRRE_Vabs_star*algs.cos(FlowTita)*algs.sin(FlowBeta)
AGUIRRE_AERO_w_bar	116	$\mathrm{w}-w^*$
AGUIRRE_AERO_w_star	---	AGUIRRE_Vabs_star*algs.sin(FlowTita)
AGUIRRE_Aw_div_MASKED		$\nabla.(T_{s_0}\,\rho\,\delta s\,\vec{V})$ + Masking by negative Q_criterion
AGUIRRE_Cd_bar	---	ARNTZ_A + AGUIRRE_Em_bar + AGUIRRE_Eth_bar
AGUIRRE_Cd_bar_corrected	129	$\overline{C_{D_\varepsilon}}+\lvert C^*_{D\varepsilon}\rvert_{wake}$
AGUIRRE_Cd_star	123	AGUIRRE_Em_star + AGUIRRE_Eth_star

AGUIRRE_Em_bar	---	AGUIRRE_Eu_bar + AGUIRRE_Ev_bar + AGUIRRE_Ep_bar
AGUIRRE_Em_bar_corrected	130	$\overline{\dot{\varepsilon}_m} + \lvert\dot{\varepsilon}_m^*\rvert_{wake}$
AGUIRRE_Em_star	---	AGUIRRE_Eu_star + AGUIRRE_Ev_star + AGUIRRE_Ep_star
AGUIRRE_Ep_bar	127	$\dot{\mathrm{E}}_{\mathrm{p}} - \dot{\mathrm{E}}_{\mathrm{p}}^*$
AGUIRRE_Ep_star	---	(Ps-Ps0)*(AGUIRRE_u_star-V0)+ AGUIRRE_Eth_p_star
AGUIRRE_Eth_bar	128	$\dot{\varepsilon}_{\mathrm{th}} - \dot{\varepsilon}_{\mathrm{th}}^*$
AGUIRRE_Eth_star	---	AGUIRRE_Eth_t_star
AGUIRRE_Eu_bar	125	$\dot{\mathrm{E}}_{\mathrm{u}} - \dot{\mathrm{E}}_{\mathrm{u}}^*$
AGUIRRE_Eu_star	---	(0.5*Rho*(AGUIRRE_u_star-V0)**2*AGUIRRE_u_star)
AGUIRRE_Ev_bar	126	$\dot{\mathrm{E}}_{\mathrm{v}} - \dot{\mathrm{E}}_{\mathrm{v}}^*$
AGUIRRE_Ev_star	---	0.5*Rho*(AGUIRRE_v_star**2+AGUIRRE_w_star**2)*AGUIRRE_u_star
AGUIRRE_INF_Cd	---	ARNTZ_A + AGUIRRE_INF_Em + AGUIRRE_INF_Eth
AGUIRRE_INF_Em	---	ARNTZ_Eu + ARNTZ_Ev + AGUIRRE_INF_Ep
AGUIRRE_INF_Ep	112	$\dot{E}_{p\ Arntz} + \dot{\varepsilon}_{th_{pressure}}$
AGUIRRE_INF_Eth	113	$\dot{\varepsilon}_{th_{temperature}}$
ARNTZ_A	108	$T_{s_0}\,\rho\,\delta s\,\mathrm{u}$
ARNTZ_Aphi	99	$\frac{T_{s_0}}{T_s}\Phi_{eff}$
ARNTZ_At	100	$\frac{T_{s_0}}{{T_s}^2}\,k_{eff}\,(\nabla T)^2$
ARNTZ_Aw_div	106	$\nabla.(T_{s_0}\,\rho\,\delta s\,\vec{V})$
ARNTZ_Aw_div_MASKED	---	$\nabla.(T_{s_0}\,\rho\,\delta s\,\vec{V})$ + Masking by Shock_Detector
ARNTZ_Cd	94	ARNTZ_A + ARNTZ_Em + ARNTZ_Eth
ARNTZ_Em	95	$\frac{1}{2}\rho\,\delta u^2 u + \frac{1}{2}\rho(v^2+w^2)u + (P_s - P_{s_0})(u-U_0)$
ARNTZ_Ep	95	$(P_s - P_{s_0})(u-U_0)$
ARNTZ_Eth	109	ARNTZ_Eth_p + ARNTZ_Eth_t
ARNTZ_Eth_p	111	$P_{s_0}\left[1-\frac{\rho}{\rho_0}\ ln\left(\frac{\rho_0}{\rho}\right)\right](\vec{V}.\vec{n}) - \rho\,R\,T_{s_0}\,(\vec{V}.\vec{n})$
ARNTZ_Eth_t	110	$\rho\,c_v\,T_s\,(\vec{V}.\vec{n})\left[1-\frac{T_{s_0}}{T_s}\ ln\left(\frac{T_s}{T_{s_0}}\right)\right] - \rho\,c_v\,T_{s_0}\,(\vec{V}.\vec{n})$
ARNTZ_Eu	95	$\frac{1}{2}\rho\,\delta u^2 u$
ARNTZ_Ev	95	$\frac{1}{2}\rho(v^2+w^2)u$
BETZ_Vartificial	40	$\sqrt{u^2+\frac{2}{\rho}(P_{t_0}-P_t)}$
BETZ_p	42	$P_{t_0} - P_t$
BETZ_profile_drag	42	$(P_{t_0}-P_t) + \frac{\rho}{2}\,(u^*-u)\,(u^*+u-2U_0)$
BETZ_u	42	$\frac{\rho}{2}\,(u^*-u)\,(u^*+u-2U_0)$
GILES_profile_drag	69	$P_{s_0}\frac{\delta s}{R} - \rho_0\,\delta h_t$

JONES_profile_drag	43	$2\sqrt{P_t - P_s}\left(\sqrt{P_{t0} - P_{s0}} - \sqrt{P_t - P_{s0}}\right)$
KUSUNOSE_lift	73	$\rho_0\, U_0\, y\, \xi - \rho_0 {U_0}^2 (1 - {M_0}^2)\frac{w}{U_0}\frac{(u - U_0)}{U_0} + {M_0}^2\, \frac{\gamma\, P_{s0}}{R}\frac{w}{U_0}\delta s - \rho_0 {M_0}^2 \frac{w}{U_0}\delta h_t$
KUSUNOSE_profile_drag	71	$P_{s0}\, \frac{\delta s}{R} - \rho_0\, \delta h_t - \frac{P_{s0}}{2}\left(\frac{\delta s}{R}\right)^2 + \rho_0\, \delta h_t \left(\frac{\delta s}{R} - \frac{{M_0}^2}{2}\frac{\delta h_t}{{U_0}^2}\right)$
KUSUNOSE_total_drag	---	KUSU_profile_drag + KUSU_vortex_drag
KUSUNOSE_vortex_drag	72	$\frac{\rho_0}{2}\left[(v^2 + w^2) - (1 - {M_0}^2)\, \delta u^2\right]$
Lamb_Vector	83	$-\boldsymbol{\omega} \times \boldsymbol{V}$
MASKELL_lift	63	$\rho_0\, U_0\, y\, \xi + \rho_0\, (U_0 - u)\, w$
MASKELL_sideforce	---	$\rho_0\, U_0\, z\, \xi + \rho_0\, (U_0 - u)\, v$
MASKELL_vortex_drag	48	$\frac{\rho}{2}(v^2 + w^2)$
MEHEUT_profile_drag	74	$\frac{1}{S_{ref}}\left[-\frac{2}{\gamma\, {M_0}^2}\Delta P_t - \Delta T_t + \left(1 - \frac{{M_0}^2}{4}\right)\Delta {T_t}^2 - \Delta P_t\, \Delta T_t - (1 - {M_0}^2)(\Delta\bar{u}^2 + 2\, \Delta u^*\, \Delta\bar{u})\right]$
MELE_induced_drag	88	$\rho l_x - m_\rho$
MELE_lift	87	$\rho l_z - m_\rho$
MELE_profile_drag	89	$-\rho\boldsymbol{x}\ \times (\boldsymbol{n} \times \boldsymbol{l})$
MOMENTUM_lift	34	$-\rho\, w\, u$
MOMENTUM_p	36	$-(P_s - P_{s0})$
MOMENTUM_sideforce	35	$-\rho\, v\, u$
MOMENTUM_stress	38	$-\mu\left[2\frac{\partial u}{\partial x} - \frac{2}{3}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)\right]$ (*See note 1*)
MOMENTUM_total_drag	36	$-\left[\rho\, (u - U_0)\, (u) + (P_s - P_{s0})\right]$
MOMENTUM_u	36	$-\rho\, (u - U_0)\, (u)$
ONORATO_Pt	---	Pt0-Pt
ONORATO_drag	---	ONORATO_Pt + ONORATO_u + ONORATO_vw
ONORATO_u	---	0.5*Rho*(u-V0)**2
ONORATO_vw	---	0.5*Rho*(v**2+w**2)
OSWATITSCH_profile_drag	44	$\frac{T_{s0}}{U_0}\rho\, u\, \delta s$
VDV_DeltaUbar	66	$U_0\left(\sqrt{1 - \frac{2}{(\gamma - 1)\, M^2}\left(e^{\frac{(\gamma-1)}{\gamma}\frac{\delta s}{R}} - 1\right) + \frac{2\, \delta h_t}{{U_0}^2}} - 1\right)$
VDV_DeltaUstar	65	$\delta u - \delta\bar{u}$
VDV_profile_drag	67	$-\rho\, u\, \delta\bar{u}$
VDV_total_drag	--	VDV_profile_drag + VDV_vortex_drag

VDV_vortex_drag	68	$-[(P_s - P_{s0}) + \rho\, u\ (u - U_0 - \delta\overline{u})\,]$
WU_induced_drag	85	$-\rho l_x$
WU_lift	84	$-\rho l_z$
WU_profile_drag	86	$-\rho \boldsymbol{x} \times (\boldsymbol{n} \times \boldsymbol{l})$
AdimF	144	$\frac{1}{2}\rho_0\, U_0^{\,2}\, S_{ref}$
AdimP	145	$\frac{1}{2}\rho_0\, U_0^{\,3}\, S_{ref}$
Alfa	---	α [input data]
Beta	---	β [input data]
CpGas	---	1006.43
CvGas	---	719.38786031
DimF	---	0.5*Rho0*Sref*V0**2
Gamma	---	1.399008873407
M0	20	$\frac{U_0}{a_0}$
Ps0	---	P_{s0} [input data]
Pt0		$P_{s0}\left(1 + \frac{\gamma - 1}{2} M_0^{\,2}\right)^{\frac{\gamma}{\gamma-1}}$
R	---	287.04213969
Rho0		$\frac{P_{s0}}{R\, T_{s0}}$
Sref	---	input data
TRANSFORMATION	7	$\overline{\overline{\boldsymbol{T}}}_x.\overline{\overline{\boldsymbol{T}}}_y.\overline{\overline{\boldsymbol{T}}}_z$
Ts0	18	$T_{t0} - \frac{U_0^{\,2}}{2c_p}$
Tt0	18	$T_{s0} + \frac{U_0^{\,2}}{2c_p}$
UpwardAxis	---	input data
V0	---	input data

Notes:

• *Note: The equation for the shear stress drag ignores the Reynolds stress term because Epsilon is mainly intended for the analysis of CFD RANS solutions with the Spalart Allmaras turbulence model, where the fluctuation terms are not available. The user must keep in mind that this neglected term is important especially for detached flows.*

2 EPSILON NEAR FIELD

VARIABLE NAME	ID	EQUATION CODED IN THE VARIABLE
F_friction	29	$-\overline{\overline{\tau}}\,.\vec{n}$
F_pressure	29	$Ps\,\vec{n}$
F_total	29	F_pressure+F_friction

M_friction	---	algs.cross(coords,F_friction)
M_pressure	---	algs.cross(coords,F_pressure)
M_total	---	M_pressure+M_friction

3 EPSILON POISSON SOLVER

VARIABLE NAME	ID	EQUATION CODED IN THE VARIABLE
AGUIRRE_vortex_exergy	131	$\frac{1}{2}\rho\,(\psi\xi - \Phi\sigma)\,u^*$
MASKELL_vortex_drag	55	$\frac{\rho_0}{2}(\psi\,\xi - \Phi\,\sigma)$
Phi*Sigma	55	$\Phi\,\sigma$
Potential_Function	57	$\frac{1}{4\pi}\frac{Log[(y-y_w)^2+(z-z_w)^2]_{model}}{Log[(y-y_w)^2+(z-z_w)^2]_{mirror}} * \sigma(y_w, z_w)$ *(See note 1)*
Sigma	54	$-\frac{\partial u}{\partial x}$
Stream_Function	56	$-\frac{1}{4\pi}\frac{Log[(y-y_w)^2+(z-z_w)^2]_{model}}{Log[(y-y_w)^2+(z-z_w)^2]_{mirror}} * \xi(y_w, z_w)$ *(See note 1)*
Vector_Speed	9	$Vx\,\vec{I} + Vy\,\vec{J} + Vz\,\vec{K}$
Vorticity_x	---	ξ

REFERENCES

[1] Aguirre, M., Duplaa, S., "Epsilon: An Open Source Tool for Exergy-Based Aerodynamic Analysis", Aerospace Europe Conference, 25-28 February 2020, Bordeaux, France.

[2] M. Drela, "Flight Vehicle Aerodynamics", The MIT Press, Cambridge, MA, 2014 ISBN: 9780262526449.

[3] Anderson, J., "Fundamentals of Aerodynamics", McGraw-Hill Education, Boston, 2010.

[4] M. Meheut, "Evaluation des Composantes Phénoménologiques de la Trainée d'un Avion à Partir des Résultats Expérimentaux", Thèse ONERA, 2006.

[5] A. Betz, "A Method For The Direct Determination of Wing-Section Drag", NACA Technical Report 337, 1925.

[6] G. Brune, "Quantitative Low speed Wake Surveys", Journal of Aircraft, Vol. 31 No 2, 1994.

[7] B. Jones, "Measurement of Profile Drag by the Pitot-Traverse Method", Aeronautical Research Council R&M Rept. 1688, 1936.

[8] H. Goett, "Experimental Investigation Of The Momentum Method For Determining Profile Drag", NACA report No 660, 1939.

[9] K. Oswatitsch, "Gas Dynamics", Academic Press, New York, 1956.

[10] E. Thoubin, "Prediction and Phenomenological Breakdown of Drag for Unsteady Flows", Thèse ONERA, 2015.

[11] E. Maskell, "Progress Towards a Method of Measurement of the Components of the Drag of a Wing of Finite Span", Royal Aircraft Establishment TR 72232, Jan. 1973.

[12] J. Diebold, "Aerodynamics Of A Swept Wing With Leading-Edge Ice At Low Reynolds Number", Master of Science Thesis. University of Illinois at Urbana-Champaign, 2012.

[13] C. Van Dam, "Drag Calculations Using Euler Methods", AIAA-91-0338, 29th Aerospace Sciences meeting, Nevada, 1991.

[14] M. Aguirre, S. Duplaa and X. Carbonneau, " Vortex Exergy Prediction", 54th 3AF International Conference on Applied Aerodynamics, 25 – 27 March 2019, Paris - France. Submitted for publication in October 2018.

[15] J. Van der Vooren and D. Destarac , "Drag/Thrust Analysis of Jet-Propelled Transonic Transport Aircraft; Definition of Physical Drag Components", Aerospace Science and Technology, Vol. 8, No. 6, 2004.

[16] J. Van der Vooren, "On Drag And Lift Analysis Of Transport Aircraft From Wind Tunnel Measurements", Aerospace Science and Technology, Vol. 12, pp. 337–345, 2008.

[17] Giles, "Wake Integration for Three-Dimensional Flowfield Computations Theoretical Development", Journal of Aircraft, Vol 36 No 2, 1999.

[18] A. Kusunose, "A Wake Integration Method for Airplane Drag Prediction", Tohoku Univ. Press, Sendai, Japan, 2005.

[19] A. Kusunose, "Wave Drag Extraction from Profile Drag Based on a Wake-Integral Method", AIAA 99-0275, 1999.

[20] Aguirre, Duplaa, "Exergetic Drag Characteristic Curves," AIAA Journal, Vol. 57, No. 7, 2019.

[21] Wu, J., Lu, X.-Y., and Zhuang, L.-X., "Integral Force Acting on a Body Due to Local Flow Structures," Journal of Fluid Mechanics, Vol. 576, April 2007, pp. 265–286.

[22] B. Mele, M. Ostieri and R. Tognaccini, "Vorticity Based Breakdown of the Aerodynamic Force in Three-Dimensional Compressible Flows", AIAA Journal, Vol. 54, No. 4, April 2016

[23] Y. Cengel, "Thermodynamics : An Engineering Approach"*, Eighth edition, Mc Graw Hill Education, 2015*.

[24] A. Arntz, "Civil Aircraft Aero-thermo-propulsive Performance Assessment by an Exergy Analysis of High-delity CFD-RANS Flow Solutions", Fluids mechanics. Université de Lille 1, 2014.

[25] M. Aguirre, S. Duplaa and X. Carbonneau, "2D Flow Field Analysis by the Exergetic Method", AIAA Applied Aerodynamics Conference, 17-21 June 2019, Dallas, Texas, United States. Submitted for publication in October 2018.

[26] M. Aguirre, S. Duplaa, X. Carbonneau and A. Turnbull, "Exergy: an alternative approach for flow field analysis", AIAA Journal. (Submitted).

[27] Aguirre, Duplaa, Carbonneau, "Vortex Exergy Prediction", 54th 3AF Int. Conference on Applied Aerodynamics, 25 – 27 March, France, 2019.

[28] Aguirre, Duplaa, Carbonneau, Turnbull, "A Systematic Analysis of the Mechanical Exergy of an Airfoil by Using Potential Flow, Euler & RANS", 24ème CFM, France, 2019.

[29] Aguirre, Duplaa, Carbonneau, Turnbull, Arntz, "A Better Assessment of the Recoverable Energy Behind a Body by the Exergy Method", Aerospace Europe Conference, 25-28 February, France, 2020

[30] Aguirre, M., Duplaa, S., Turnbull, A. and Carbonneau, X., "A velocity decomposition method for exergy-based drag prediction," AIAA Journal, (Submitted).

[31] Aguirre, M., Duplaa, S., Turnbull, A. and Carbonneau, X., "Wave drag assessment in transonic flows by the exergy method," AIAA Journal, (Submitted).

[32] D.Lovely, R.Haimes, "Shock Detection From Computational Fluid Dynamics Results", AIAA 99-3285, 1999.

[33] Utkarsh Ayachit, "The Paraview guide: Community Edition", Kitware inc., 2016.

[34] K. Moreland, "The ParaView Tutorial, Version 5.4.1", Sandia National Laboratories, 2017.