POLITECNICO DI TORINO

Master Degree in Aerospace Engineering
Department of Mechanical and Aerospace Engineering

Master Degree Dissertation

All weather, GPS free and data-driven synthetic sensor for angle of attack and sideslip estimation

Student:
Giuseppe Donofrio

Academic Advisors:
Ing. Angelo Lerro
Prof. Piero Gili
Leonardo Advisor:
Dott. Massimo Cifaldi

Academic Year 2019-2020

Acknowledgments

The present work has been performed thanks a close cooperation with Leonardo S.p.A. Aircraft Division and through the use of Information and Data, property of Leonardo S.p.A. Aircraft Division, which remains the sole owner of all such relevant IP rights. The results of the present work shall be, therefore, property of Leonardo S.p.A Aircraft Division.
For confidentiality reasons all data in this thesis are normalized in the range $[-1, 1]$.

Contents

Abstract

Despite very uncommon, the sequential failures of all aircraft Pitot tube, with the consequent loss for all dynamical parameters from the Air Data System, have been found to be the cause of a number of catastrophic accidents in aviation hystory. Malfunctioning pitot tubes contributed, for example, to the infamous disappearance of Air France Flight 447 over the Atlantic in 2009: temporary inconsistencies between the airspeed measurements caused the autopilot to disconnect, after which the crew reacted incorrectly and ultimately caused the aircraft to enter an aerodynamic stall, from which it did not recover[21]. This work proposes an all weather, GPS-free, data-driven and synthetic sensor for angle of attack and angle of sideslip estimation. This approach consists in an appropriate selection of input signals and maneuvers aimed to develop a Neural Network-based estimator to be used online for failure detection. The proposed approach has been carried out and validated exploiting real flight data provided by Leonardo S.p.A. Aircraft Division from one of their prototype's simulator. In particular, for the purpose of this works, only maneuvers at Mach lower than 0.6 have been taken into account so that transonic effects can be neglected. The performance of virtual sensor are evaluated for different wind conditions (asbsence of wind, constant wind, gusts and sinusoidal wind) and several center of gravity positions in order to preliminarly asses the feasibility of this kind of neural system in different scenarios. A basic sensitivity analysis is eventually carried out to evaluate how many and which parameters are really essential for the Neural Network training. The results confirms the robustness of a Neural Network-based approach in absence of external wind or with constant wind, but suggest a deeper analysis to explore the influence of the center of mass and variable wind on the virtual sensor's performances.

1 Introduction

The Air Data System (ADS) [15] is a critical component of the conventional suite of sensors for both manned and unmanned aircraft. The ADS provides direct measurements of critical flight data as airspeed, altitude, outside air temperature and aerodynamic angles (known as angle of attack and angle of sideslip) without whom, the remote piloting and automatic flight could not be possible. Currently, aerodynamic angles are generally derived from Multi function probes and vanes that are installed on the fuselage and/or wings. Though these devices are very robust to even extreme weather conditions, under particular circumstances, some peculiar ice crystals could obstruct the tiny conducts of the ADS, thus inducing faulty sensor measurements. Incorrect measures of critical flight data could eventually lead to unrecoverable flight conditions, such as the case of Air France Flight 447 [21]. Additional causes of crashes could be related to erroneus coverage of the Pitot tube (AeroPerù 757 [45]), presence of insects inside the static taps [27] and icing of angle of attack vanes (XL Airways Germany Flight 888T [14]). Currently, the conventional approach adopted to provide fault tolerance for flight sensors is based on Hardware Redundacy (HR) [19], that basically consists in the installation of multiple sensors, measuring the same parameter, which are continuosly monitored to check their status and compared to highlight

discrepancies and isolate the faulty sensor. The increasing need of both modern unmanned both manned aerial vehicles (UAVs and MAVs) to reduce the cost and the complexity of on-board systems, has encouraged the development of executable software codes which could, whenever possibile, replace the heavy, expensive and cumbersome physical sensors. This alternative approach is know as Analytical Redundancy (AnR) [43] and basically involves the use of synthetic sensors, such as conventional state estimators, Neural Networks, Kalman Filters and/or other predictive models which provide alternative estimates of the paramenter detected by the real sensor that needs to be redounded. Recently, an AnR scheme based on machine learning technique has been proposed in [53] for the failure detection and correction of the airspeed sensor of an aircraft. Exploiting data coming from other sensors (functionally related to the parameter associated with the sensor that need to be redounded) as inputs, the synthetic sensor can estimate the target measurement of the parameter and compare it with that provided by that sensor under verification. This comparison returns a residual signal that, if exceeds a detection threshold, declares an alarm status. It is important to remark as this approach is completly data-driven and does not require any specific type of aircraft-dependent dynamic mathematical model. More generally, AnR identifies with functional redundancy of the system. The idea of exploiting software algorithms to replace hardware redundancy was introduced as soon as digital computers started to be used in the 1970's to perform redundancy management. Approaches developed to detect and isolate faulty sensors were ultimately to become important part of later control reconfiguration schemes. An example is provided by the Sequential Probability Ratio Tests that were flight-tested on the F-8 Fly-By-Wire demonstrator in the late 1970's[55]. Throghout the 1970s and 1980s many papers appeared describing various algorithms for managing redundant systems and redundant sensors. Nowdays, the very challenge is to verify if these methods, developed within the academic community, are ready to be applied to existing industrially-developed UAVs and if they comply with current certification process.

The all weater, GPS-free synthetic sensor for angle of attack and angle of sideslip estimation, performed thanks a close cooperation with Leonardo S.p.A. Aircraft Division and through the use of Information and Data, property of Leonardo S.p.A. Aircraft Division, is based on Neural Networks (NN) to overcome those shortcomings related to model-based methods. In the aerospace field, NNs are already used as system identification devices to indirectly estimate aerodynamic coefficients [47], angle of attack [48] and sideslip [18] exploiting data not deriving from vanes, differential pressure sensors and modern multifunctional probes.

Neural Networks, as an emerging discipline, studies or emulates the information processing capabilities of neurons of the human brain. It uses a distributed representation of the information stored in the network, and thus resulting in robustness against damage and corresponding fault tolerance. Usually, a Neural Network model takes an input vector X and produces an output vector Y in a deterministic way. The relationship between X and Y is determined by the network architecture starting through observation of phisical phenomena (training stage). There are many forms of network architecture inspired by the neural architecture of the human brain. A major advantage of Neural Networks is their ability to provide flexible mapping between inputs and outputs. Having a general map between the input and

output vectors eliminates the need for unjustified priori restrictions that are needed in conventional statistical and econometric modeling. Therefore, a Neural Network is often viewed as a "universal approximator" i.e. a flexible functional form that can well approximate any arbitrary function, given sufficient middle-layer units and properly adjusted weights [58].
In order to evaluate the response of the neural system in different scenarios, the performance of the hereafter developed virtual sensor are evaluated firstly at varying of wind conditions in asbsence of wind (Sect. 11.1), with constant wind and gusts(Sect. 11.3 and 12.1) and with sinusoidal wind(Sect. 11.4 and Sect. 12.2)); secondly at shifting of the center of gravity position (Sect. 11.5 and Sect. 12.3). For certification assessment, the virtual sensor should mantain an accuracy within the threshold of $\pm 1.5°$. A basic sensitivity analysis is eventually carried out to evaluate how many and which parameters are really essential as inputs for the Neural Network training(Sect. 11.6 and Sect. 12.4).

2 Aircraft reference system

In this section will be presented an overview of the all possible aircraft reference system that are adopted throughout the whole work.

2.1 Earth-Centered Inertial Frame, ECI

The concept of an inertial frame is of fundamental philosophical importance in the history of science, evolving from the combined studies carried out by Galileo Galilei (Italian, 1564-1642), Isaac Newton (English, 1642- 1727), Ernst Mach (Austrian, 1838-1916) and Albert Einstein (German, 1879-1955). Newton conceived of an "absolute spac" to which is referred the acceleration in his second law. Whenever the equations of motion are applied, such as the force equations or moment equations, the acceleration must be measured relative to a Newtonian or Inertial reference frame. Difficulties in distinguishing between absolute and relative rotation led Mach to conclude that rotation could only be thought as occurring relative to the matter of the universe. He further defined inertial frames as those which are unaccelerated relative to the "fixed stars". Einstein eventually synthesized the observation by Galileo that a body's acceleration in a gravitational field is independent of its mass with the theory of Newton and Mach to arrive at the so-called principle of equivalence. According this principle, it is impossible to distinguish instantaneously between gravitational and inertial force since intertial forces that are measured in a non-intertial frame of reference are, in fact, gravitational forces exerted by the stars.
The question arises to how one can refer measured forces and motions to an inertial frame that has physical significance to the problem of navigation in the vicinity of the Earth.
In fact, the Earth is characterized by a complex motion with respect to each inertial reference system: the same solar system moves with respect to fixed star, the Earth has a revolution motion around the Sun and the Earth's center describes an elliptical

trajectory in a one year period. In addition, the Earth rotates around its own axis with a period equal to one day and other slower motions, like precession and nutation, contribute to add complexity to Earth's motion.
By the way, in athmospheric navigation problems, a reference frame with its origin in the Earth's center and invariable orientation with respect to fixed stars, can be reasonably assumed as inertial. This approximation is true when the duration of the phenomenon that is observed is little enough that can be assumed that Earth's center has constant speed and motions on large-scale can be negleted .
Since (when applying the equations of motion) acceleration must be measured relative to an inertial reference frame, it follows that angular velocity and angular acceleration (such as for a rigid body) must also be measured relative to this frame, as these quantities directly affect the acceleration.
This reference, known as Earth-Centered Inertial Frame (ECI), is called $\tau_{ECI} = \{O, x, y, z\}$ and has its z_{ECI} axis oriented along the north pole direction as shown in Fig. 1.

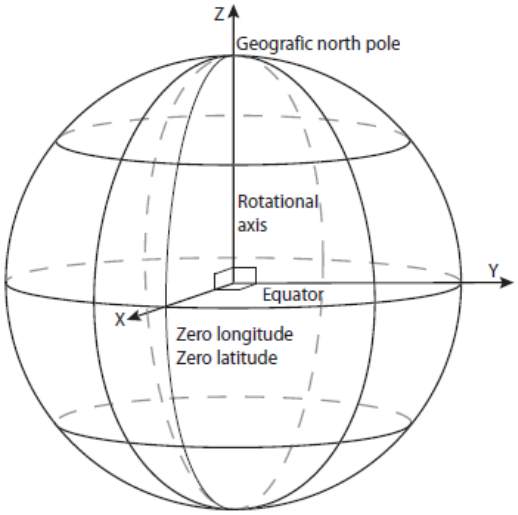


Figure 1: Earth-Centered Inertial Frame. From "The Different Frames and the Keplerian Elements",[54]

2.2 Geografic Frame, n, or North-East-Down, NED

The geographic frame (Fig. 2) is a local navigational frame which has its origin jointed to the aircraft's center of mass and its axes always aligned with the three standard geographic directions: North, East and Down. The down axis, D, is defined to be the normal to the reference ellipsoid, an analytically defined surface which is an approximation of the mean sea level gravity equipotential surface, the geoid. The north axis, N, is in the direction of the projection of the Earth's inertial angular velocity vector into the local horizontal plane (the plane which is perpendicular to Down direction). The east direction, E, eventually completes the right-handed orthogonal set.

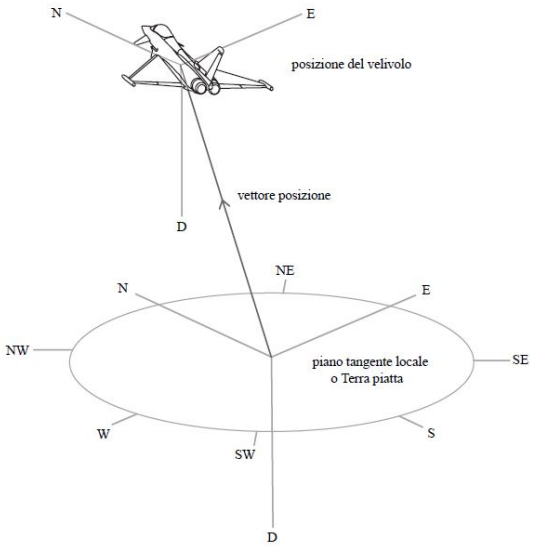
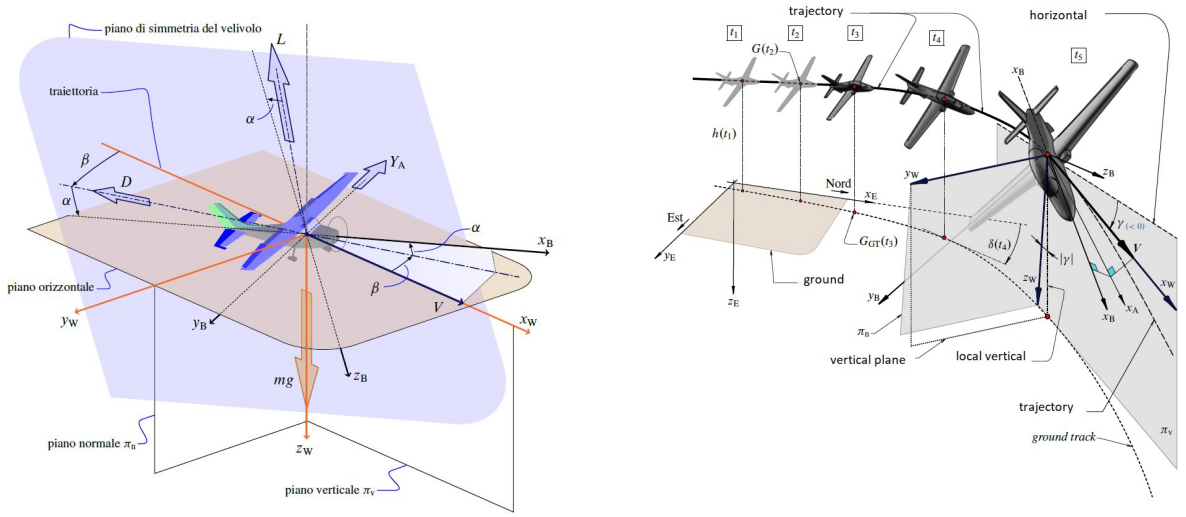


Figure 2: North-East-Down Frame. From De Marco et al.,"Elementi di Dinamica e simulazione di volo", [17].

2.3 Wind Axes

The wind axes tern has its origin coincident with the aircraft's center of mass (**G**) and its longitudinal axis x_w oriented positive towards the aeroplane's velocity direction (**V**). The wind axis z_w is defined by the intersection between the vertical plane π_v (which contains **V** and **G**) and the normal plane π_n (which is perpendicular to the trajectory in **G**) towards positive downward.
The transversal axis is defined in such a way that completes the tern $\{G, x_w, y_w, z_w\}$.
It is remarkable that the transversal axis y_w is always horizontal. In fact, it is normal to the plane $\{G, x_w, z_w\}$ which is, by definition, constantly vertical.
In Fig. 3(a) is reported a particular case in which the trajectory of the center of mass is horizontal. It can be noticed that also in a not symmetrical orientation of the aircraft with respect to the vertical plane x_w-z_w, the wind axis z_w is vertical aligned with the weight force mg.



(a) (b)

Figure 3: Wind axes tern. From De Marco et al.,"Elementi di Dinamica e simulazione di volo", [17].

In Fig. 3(b) is eventually reported an evolution in which the trajectory described by the center of mass is curved and the aircraft's orientation is not symmetrical with respect to π_n. The wind axis x_w is, by definition, tangent to the trajectory and not horizontal whereas the z_w axis is not vertical. On the other hand, as by definition, the wind axis y_w is always horizontal.
In conclusion, the wind axes are a mobile tern with origin in **G**, orientation that can be reconstructed only by means of its trajectory and independent of aircraft's orientation in the space.

2.4 Body Frame

The body frame (Fig. 4) constitutes the familiar airplane axes of roll, pitch and yaw and has its origin at the vehicle center of mass. The longitudinal axis x_B is contained in the aircraft's plane of symmetry and is oriented positive towards the bow. The body axis z_B is perpendicular to x_B, it belongs to the plane of simmetry and is oriented positive in the head-feet's pilot direction. The body axis y_B completes the left-handed tern such that results oriented positive towards the pilot's right.
Note that the origin of the body frame does not, in general, coincide with the location of the navigation system.

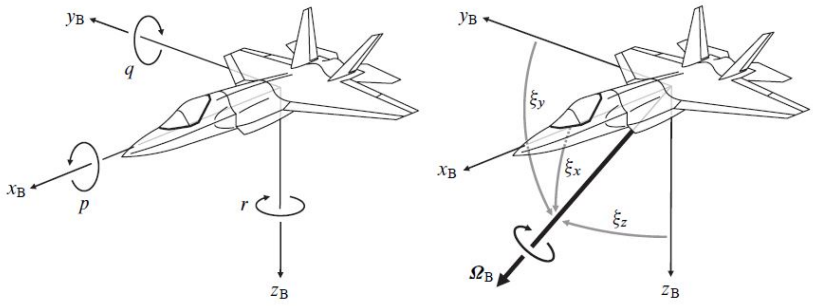


Figure 4: Body reference frame. From De Marco et al.,"Elementi di Dinamica e simulazione di volo", [17].

2.5 Body-to-NED matrix of rotation

In order to switch from NED to Body reference frame, three fundamental rotations are required:

1. $\psi > 0$ clockwise rotation about $z_{NED} = D$: $\{N, E, D\}$ turns into $\{x', y', D\}$;
2. $\theta > 0$ clockwise rotation about y': $\{x', y', D\}$ turns into $\{x'', y', z''\}$;
3. $\phi > 0$ clockwise rotation about $x_B == x''$: $\{x'', y', z''\}$ turn into x_B, y_B, z_B;

where ψ, θ and ϕ are known as Euler's angles.

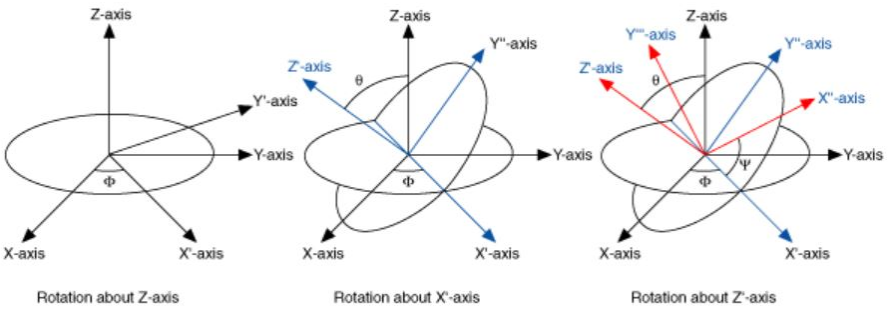


Figure 5: Euler's angles fundamental rotation. "From 3D Cartesian Coordinate Rotation (Euler) VI" [2].

Each elementar rotation can be explicited in matrix form respectively as:

1.
$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_1 = \begin{bmatrix} cos\psi & sin\psi & 0 \\ -sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{NED} = C_\psi \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{NED} \tag{1}$$

2.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_1 = \begin{bmatrix} cos\theta & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & cos\theta \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{NED} = C_\theta \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{NED} \tag{2}$$

3.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\phi & sin\phi \\ 0 & -sin\phi & cos\phi \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{NED} = C_\phi \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{NED} \tag{3}$$

Thus the gobal rotation matrix C_B^N necessary to pass from NED to Body reference frame can be expressed as:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\phi & sin\phi \\ 0 & -sin\phi & cos\phi \end{bmatrix} \cdot \begin{bmatrix} cos\theta & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & cos\theta \end{bmatrix} \cdot \begin{bmatrix} cos\psi & sin\psi & 0 \\ -sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{NED}$$

$$C_\phi C_\theta C_\psi \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{NED} = C_B^N \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{NED}$$

2.6 Body-to-Wind matrix of rotation

In this subsection it will be determined the matrix of rotation C_w^B that define the orientation of τ_B with respect to τ_w.

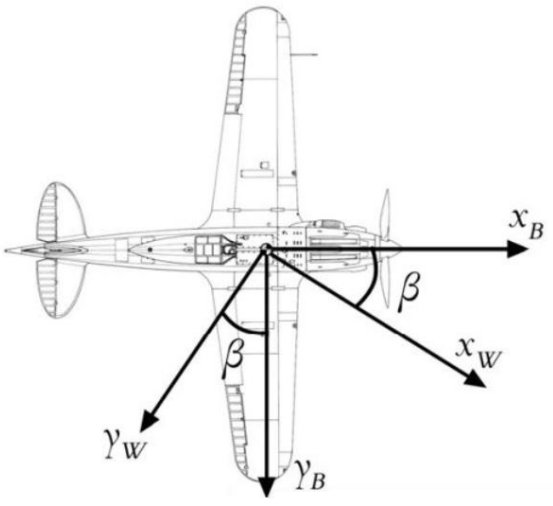


Figure 6: Angle of Sideslip (AoS). From Leonardo di Lena et al.,"Meccanica del volo - parte 1 di 2",[32].

Starting from the situation in Fig. 6, only two fundamental rotations are necessary to make Wind reference frame be coincident with Body reference frame:

1. $\beta < 0$ counterclockwise rotation about z_w axis that turns $\{x_w, y_w, z_w\}$ into $\{x', y_B, z_w\}$;
2. $\alpha > 0$ clockwise rotation about $y_w = y_B$ axis that turns $\{x', y_B, z_w\}$ into $\{x_B, y_B, z_B\}$.

The trasformation matrix that represents the first elementar rotation is

$$C_\beta = \begin{bmatrix} cos(-\beta) & sin(-\beta) & 0 \\ -sin(-\beta) & cos(-\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} cos\,\beta & -sin\,\beta & 0 \\ sin\,\beta & cos\,\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

The situation in the plane of simmetry after the first rotation is then depicted in Fig. 7.

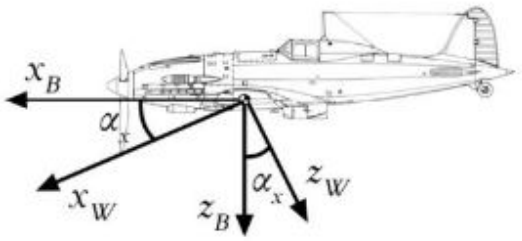


Figure 7: Angle of attack (AoA). From Leonardo di Lena et al.,"Meccanica del volo - parte 1 di 2",[32].

In order to make the two frames exactly co-incident it is necessary a rotation about y_B by an angle α. The second elementar matrix of rotation is hence:

$$C_\alpha = \begin{bmatrix} cos\,\alpha & 0 & -sin\,\alpha \\ 0 & 1 & 0 \\ sin\,\alpha & 0 & cos\,\alpha \end{bmatrix} \tag{5}$$

Thus, the rotation matrix C_w^B is:

$$C_w^B = C_\alpha \cdot C_\beta = \begin{bmatrix} cos\,\alpha\,cos\,\beta & -cos\,\alpha\,sin\,\beta & -sin\,\alpha \\ sin\,\beta & cos\,\beta & 0 \\ sin\,\alpha\,cos\,\beta & -sin\,\alpha\,sin\beta & cos\,\alpha \end{bmatrix} = C_w^B(\alpha, \beta) \tag{6}$$

Once introduced α and β is now important to verify how these aerodynamic angles are related to the variables of motion. Starting from

$$\vec{v_{c_B}} = C_w^B \vec{v_{c_W}} \;\Rightarrow\; \begin{bmatrix} u \\ v \\ w \end{bmatrix} = C_w^B \begin{bmatrix} V \\ 0 \\ 0 \end{bmatrix} \tag{7}$$

where V is the absolute value of $\vec{v_{c_W}}$ and can be expressed, from Eq.7, as $V = \sqrt{u^2 + v^2 + w^2}$.

Solving Eq.7, components of $\vec{v_{cB}}$ can be thus related to aerodynamic angles as:

$$\begin{cases} u = V\, cos\, \alpha\, cos\, \beta \\ v = V\, sin\, \beta \\ w = V\, sin\, \alpha\, cos\, \beta \end{cases} \tag{8}$$

Dividing the first with the last equation of system (8) it is obtained:

$$tan\, \alpha = \frac{w}{u}$$

$$\alpha = arctan\, \frac{w}{u} \tag{9}$$

The analytic expression for β can be directly found from the second equation of system (8) as

$$\beta = arcsen \frac{v}{V} \tag{10}$$

Hence, known the velocity components of the centre of mass in τ_B, using relations (9) and (10) is possible to compute directly the aerodynamic angles α and β.

2.7 Computation of V_{NED} components using GPS data

The most natural and convenient way to specify the inertially determined position relatively to the Earth is in terms of system geocentric position vector, $\mathbf{r}$. The use of the geocentric position vector is particularly appropriate as the gravitational field compensated accelerometer outputs are proportional to the second time derivative of the inertially referenced geocentric position vector. As can be seen in Fig. 8, the inertially referenced geocentric position vector is given by:

$$\mathbf{r^I} = \{r\, cos L_c\, cos\lambda, r\, cos L_c\, sin\lambda, r\, sin L_c\,\}$$

where r is the geocentric position vector, L_c the geocentric latitude and λ the celestial longitude.

From the same figure it can be seen that the expression of the geocentric position vector in the geographic frame is

$$\mathbf{r^n} = \{-r sin\, D, 0, r cos\, D\}$$

where $D = L - L_c$ is known as deviation of the normal and L is the geographic latitude. The geocentric position vector can be written in terms of the geocentric Earth radius $\mathbf{r_0}$ and the altitude above the reference ellipsoid as

$$\mathbf{r} = \mathbf{r_0} + \mathbf{h}$$

Since

$$\mathbf{r_0^N} = \{-r_0 sin\, D_0, 0, r_0 cos\, D_0\}$$

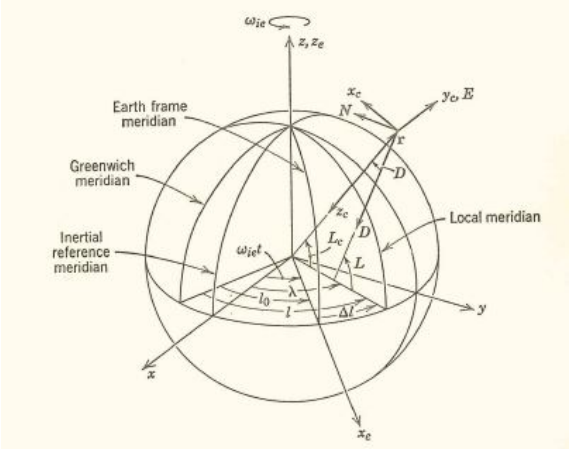


Figure 8: Coordinate frame geometry. From Kenneth R. Britting,"Inertial Navigation Systems Analysis",[12].

and

$$\mathbf{h^N} = \{0, 0, -h\}$$

Then, the geocentric position vector can be also given by:

$$\mathbf{r^n} = \{-r_0\, sinD_0,\ 0,\ -r_0\, cosD_0\ - h\}$$

The magnitude of the geocentric position vector is found to be

$$r = \sqrt{(r_0 + h)^2 - 2hr_0(1 - cosD_0)}$$

Factorization of $(r_0 + h)$ yields to

$$r = \sqrt{(r_0 + h)\left[1 - \frac{2hr_0(1 - cosD_0)}{(r_0 + h)^2}\right]}$$

On the other hand, as the quantity $(1 - cos\, D_0) \approx D_0^2/2$ to an accuracy greater than 1 part in 10^9, the expression above can be expanded in series so that

$$r = r_0 + h - \frac{hr_0D_0^2}{2(r_0 + h)} - \frac{h^2r_0^2D_0^4}{8(r_0 + h)^3}$$

For aircraft altitudes, it can be showed that the error involved in evaluating the geocentric radius magnitude with the expression

$$r = r_0 + h$$

is less than 1 ft.
The deviation of the normal is defined as the angle between the geocentric and geographic verticals, that is,

$$D = L - L_c$$

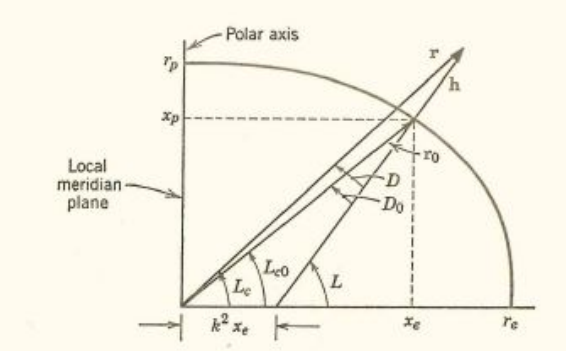


Figure 9: Earth radius altitude. From Kenneth R. Britting,"Inertial Navigation Systems Analysis",[12].

Applying the law of sines to the triangle bounded by the geographic and geocentric radii (Fig. 9) it is obtained:

$$\frac{sin\, D}{k^2 x_e} = \frac{sin(\pi - L)}{r}$$

where

- $k = \sqrt{1 - \frac{r_p^2}{r_e^2}}$ is the eccentricity of the reference ellipsoid
- r_e is the the equatorial Earth radius (semimajor axis)
- r_p is the polar Earth radius (semiminor axis)
- x_e is the equatorial projection of the Earth radius vector

Given the approximation $r \cong r_0 + h$ and that the equatorial projection of the Earth radius vector is

$$x_e = r_0 cos\, L_{c0}$$

using $L_{c0} = L - D_0$, it can be written:

$$x_e = r_0(cos\, L\, cos\, D_0 + sin\, L \sin\, D_0)$$

Finally the eccentricity is related to the ellipticity by the relationship

$$k^2 = 2e\left(1 - \frac{e}{2}\right)$$

where the ellipticity is defined as

$$e = \frac{r_e - r_p}{r_e}$$

Substituting one expression into other yields to

$$D = e\frac{r_0}{r_0 + h}\left(1 - \frac{e}{2}\right) sin\, 2L\, cos\, D_0 + 2e\left(1 - \frac{e}{2}\right)\frac{r_0}{r_0 + h} sin^2 L\, sin\, D_0$$

Evaluating this expression at $h = 0$:

$$D_0 = e\, sin\, 2L + \epsilon$$

where

$$D_0 = -(e^2/2)sin\,2L + 2e^2 sin\,2L sin^2\,L + ... \leq 1.6arc - sec$$

The Earth radius vector, for the purpose of the inertial navigation computations, is defined as the vector extending from the center of the Earth to the surface of the refernce ellipsoid. Since the reference ellipsoid is a solid of revolution, it is only necessary to work with the meridian plane equation given by:

$$\frac{x_e^2}{r_e^2} + \frac{x_p^2}{r_0^2} = 1$$

where $x_e^2 = r_0\,cos^2\,L_{c0}$ and $x_p^2 = r_0^2\,sin^2\,L_{c0}$, so that it yields to

$$r_0^2 = \frac{r_p^2}{1 - [1 - (r_p/r_e)^2]\,cos^2\,L_{c0}}$$

As the bracketed quantity in the expression above is the square of the eccentricity of the ellipse, this latter can be expanded in series, yielding to

$$r_0 = r_p \left(1 + \frac{k^2}{2} cos^2\,L_{c0} + \frac{3}{8} k^4 cos^4\,L_{c0} + \frac{5}{16} k^6 cos^6\,L_{c0} + ...\right)$$

Noting that $k^2/2 = e(1 - e/2)$ and $L_{c0} = (L - D)$, the L_{c0} terms can be expanded so that the expression for r_0 becomes

$$r_0 = r_p \left[1 + \frac{e}{2}(1 + cos(2L) + \frac{e^2}{4}\left(\frac{13}{4} + 2\,cos(2L) - \frac{5}{4}\,cos(4L)\right) + ...\right]$$

Remembering that the relationship between the polar and the equatorial radii is given by

$$r_p = r_e(1 - e)$$

it is finally obtained

$$r_0 = r_e \left[1 - \frac{e}{2}(1 - cos(2L)) + \frac{5}{16} e^2 (1 - cos(4L)) - ...\right]$$

where r_0 can be approximated as

$$r_0 = r_e \left(1 - e\,sin^2\,L\right) \tag{11}$$

making an error on the order of 150 ft, assuming that r_e can be precisely specified.
The primary measurements on which the navigational computations are based are the accelerometers measurements. These measurements are either assumed to be coordinatized in the desired computation frame, which is the case for the local vertical and space stabilized platform systems.
The geographically referred Earth velocity $\mathbf{v^N} = \{v_N, v_E, v_D\}$ is, by definition:

$$\mathbf{v^N} \triangleq \mathbf{C_E^N \dot{r}^E}$$

It is observed, from the matrix form of the theorem of Coriolis, that:

$$\dot{\mathbf{r}}^{\mathbf{E}} = \mathbf{C}_{\mathbf{I}}^{\mathbf{E}} \left[\dot{\mathbf{r}}^{\mathbf{I}} - \left(\mathbf{\Omega}_{\mathbf{IE}}^{\mathbf{I}} \mathbf{r}^{\mathbf{I}} \right) \right]$$

So that, in geographic coordinates the equation above becomes:

$$\mathbf{v}^{\mathbf{N}} = \dot{\mathbf{r}}^{\mathbf{N}} + \mathbf{\Omega}_{\mathbf{EN}}^{\mathbf{N}} \mathbf{r}^{\mathbf{N}}$$

or, in component form:

$$\mathbf{v}^{\mathbf{N}} = \{ \dot{r_N} - r_D \dot{L}, -(r_D \cos L + r_N \sin L)\dot{l}, \dot{r_D} + r_N \dot{L} \}$$

Now, remembering that the geocentric position vector is given by

$$\mathbf{r}^{\mathbf{N}} = \{ -r_0 \sin D_0, 0, -r_0 \cos D_0 - h \}$$

differentiating $\mathbf{r}^{\mathbf{N}}$ and substituting into equation of Coriolis' theorem, the NED velocity vector components are finally obtained:

$$\begin{cases} v_N = (r_0 \cos D_0 + h)\, \dot{L} - \dot{r_0} \sin D_0 - r_0 \dot{D}_0 \cos D_0 \\ v_E = (r_0 \cos L_{c0} + h \cos L)\dot{l} \\ v_D = -\dot{h} - \dot{r_0} \cos D_0 + r_0 \dot{D}_0 \sin D_0 - r_0 \dot{L} \sin D_0 \end{cases}$$

2.7.1 The World Geodetic System

The World Geodetic System (Fig. 10) is a standard for use in cartography, geodesy, and satellite navigation which includes the definition of fundamental and derived constants of the coordinate system, the ellipsoidal (normal) Earth Gravitational Model (EGM), a description of the associated World Magnetic Model (WMM) and a current list of local datum transformations [1]. Its latest revision is WGS84 established in 1984 and last revised in 2004.
The coordinate origin of WGS84 is meant to be located at the Earth's center of mass with an uncertainty that is assumed to be less than 2 cm.
The WGS84 meridian of zero longitude is the IERS Reference Meridian, 5.3 arc seconds or 102 metres (335 ft) east of the Greenwich meridian at the latitude of the Royal Observatory.
The WGS84 datum surface is an oblate spheroid with equatorial radius $r_e = 6378137\, m$ at the equator, flattening $f = 1/298.257223563$, gravitational constant (mass of Earth's atmosphere included) $\mu = 3986004.418 \cdot 10^8 m^3/s$ and angular velocity of the Earth $\omega = 72.92115 \cdot 10^6\, rad/s$.
This leads to several computed parameters such as the polar semi-minor axis $r_p = a(1 - f) = 6356752.3142\, m$, and the first eccentricity squared $e^2 = 6.69437999014 \cdot 10^{-3}$.
Since 2000, the use of WGS84 is compulsory as standard for aircraft navigation.

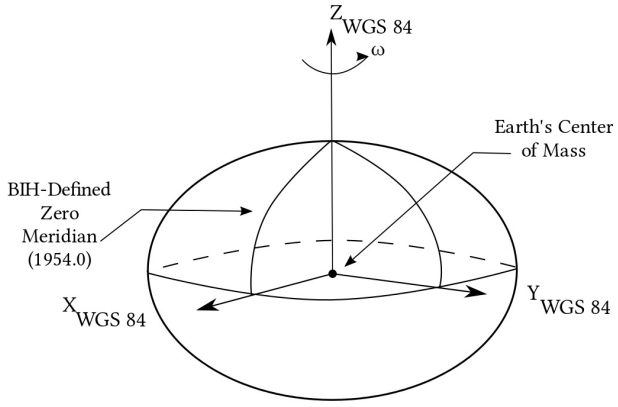


Figure 10: WGS84. From "World Geodetic System" [1]

.

2.8 Strategy for computation of aerodynamic angles in presence of external wind

From the mechanic of flight, inertial (GPS), body and wind velocities can be related as:

$$\vec{V_{GPS}} = \vec{V_B} + \vec{V_w} \tag{12}$$

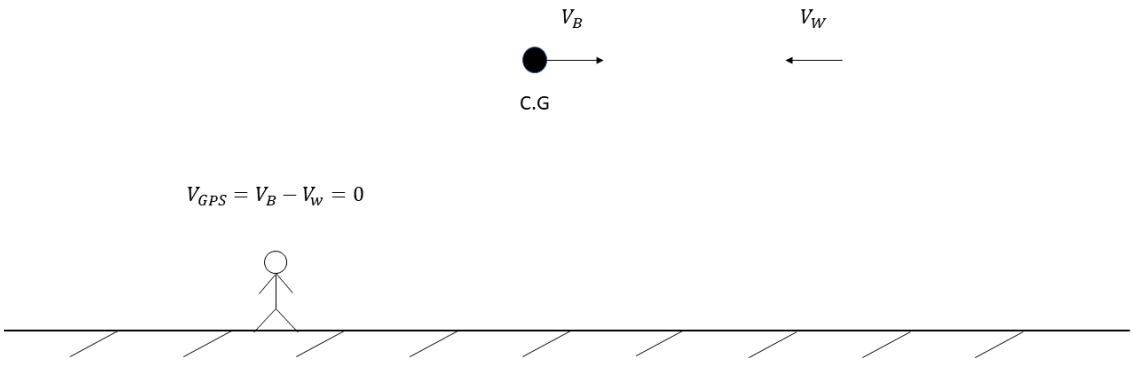


Figure 11: Graphical representation of Eq.12

Intuitively, it can be imagined an aircraft that moves with a certain velocity $\vec{V_B}$ which encounters an adverse wind equal in its absolute value to V_B: the velocity of the vehicle with respect to an observator located on the Earth is equal to zero but is not null if considered in its aerodynamic frame where the wing develops lift.
Rewriting the equation above in geographic coordinate system yields to:

$$\vec{V_{NED}} = \left(\vec{V_B}\right)_{NED} + \left(\vec{V_w}\right)_{NED}$$

where:

$$\vec{V_{NED}} = [\, V_N,\, V_E,\, V_D \,]^T$$

$$\vec{V_B} = [\, u,\, v,\, w \,]^T$$

$$\vec{V_w} = [\, u_w,\, v_w,\, w_w \,]^T$$

$$\left(\vec{V_B}\right)_{NED} = C_B^N \vec{V_B}$$

Taking into account their absolute value yields to:

$$|\vec{V_B}| = V$$

$$|\vec{V_{GPS}}| = |\vec{V_{NED}}| = \sqrt{V_N^2 + V_E^2 + V_D^2}$$

$$|\vec{V_w}| = |\vec{V_{NED}}| - V$$

where V is obtained from the Air Data System and V_{NED} is provided by GPS.
If $|\vec{V_w}| \leq 1\, m/s \Rightarrow \vec{V_B} \cong C_B^N \vec{V_{NED}}$.
Aerodynamic angles can be approximated as seen in Eq.9 and Eq.10 respectively:

$$\alpha = arctg\, \frac{w_B}{u_B}$$

$$\beta = arcsin\, \frac{v_B}{V}$$

Otherwise, if $|V_w| \geq 1\, m/s$ the approximation above is no longer valid as the inertial axis are no more coincident with aerodynamic axis: α and β must be then computed as Neural Network outputs.
The process is summarised in Fig. 12.

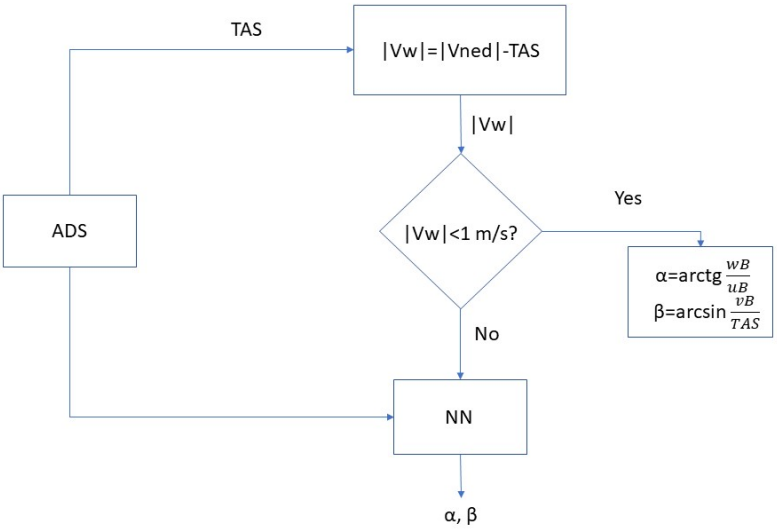


Figure 12: Block diagram for computation of aerodynamic angles

3 Six degrees of freedom model

Six degrees of freedom (6 DoF [49]) refers to the freedom of movement of a rigid body in the three-dimensional space (Fig. 13). Specifically, the body is free to translate (forward/backward, up/down, left/right) along three perpendicular axes (often named as normal axis, transverse axis, longitudinal axis) and change its orientation through rotation about them.

According to 6DoF model, there are three basic rotations an aircraft can make:

- *Roll*= Rotation about x-axis

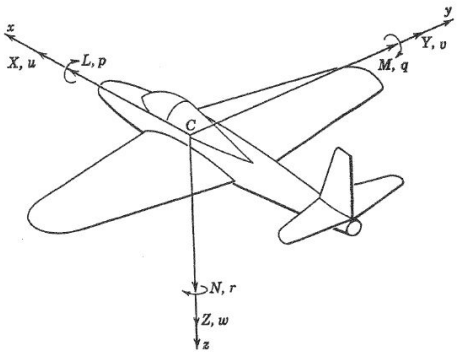


Figure 13: Six Degree of Freedom model for a generic A/C. From Matthew M. Peet et al.,"Spacecraft and Aircraft Dynamics",[49].

- *Pitch*= Rotation about y-axis
- *Yaw*= Rotation about z-axis

Each rotation in a one-dimensional transformation and any two coordinate system can be related by a sequence of three rotations.
Forces and moments have standard labels too.
In particular, Forces are referred as:

- *X, Axial Force*= Net Force along the positive x-direction
- *Y, Axial Force*= Net Force along the positive y-direction
- *Z, Axial Force*= Net Force along the positive z-direction

On the other hand, Moments are intuitively called:

- *L, Rolling Moment*= Net Moment in the positive p-direction
- *M, Pitching Moment*= Net Moment in the positive q-direction
- *N, Yawing Moment*= Net Moment in the positive r-direction

Newton's Second Law tells us that for a particle that moves with inertial velocity $\vec{V_I}$, stands the relation:

$$\vec{F} = \sum_i \vec{F_i} = m\frac{d}{dt}\vec{V_I}$$

That is, if $\vec{F} = [\, F_x,\, F_y,\, F_z\,]$ and $\vec{V_I} = [\, u_I,\, v_I,\, w_I\,]$, then:

$$F_x = m\frac{du_I}{dt}$$

$$F_y = m\frac{dv_I}{dt}$$

$$F_z = m\frac{dw_I}{dt}$$

It is relevant to remark that Newton's Second Law is only valid if $\vec{F}$ and $\vec{V_I}$ are defined in an *inertial* coordinate system that is, as defined in the Sect. 2.1, a coordinate system which is not accelerating or rotating.
If a force $\vec{F}$ is exerted on a point mass m that is at a distance r from a pivot point, an analogous equation to Newton's Second Law which involves torque and rotational motion is obtained. Extending the analysis to rigid bodied by integrating over all particles leads to:

$$\vec{M} = \sum_i \vec{M_i} = \frac{d}{dt}\vec{H}$$

In which $\vec{H} = \int \left(\vec{r_c}\times\vec{v_c}\right) dm$ is the angular momentum.
Angular momentum of a rigid body can be found as

$$\vec{H} = I\vec{\omega_I}$$

Where $\vec{\omega_I} = [p_I,\, q_I,\, r_I]^T$ is the angular rotation vector of a body about its center of mass defined in a *Inertial* Frame. In particular:

- p_I is the rotation about the intertial x-axis
- q_I is the rotation about the intertial y-axis
- r_I is the rotation about the intertial z-axis

The matrix I is the *Moment of Inertia Matrix*, defined as:

$$I = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix}$$

In which, by definition:

$$\begin{aligned} I_{xy} &= I_{yx} = \int\int\int (xy)\, dm \\ I_{xz} &= I_{zx} = \int\int\int (xz)\, dm \\ I_{yz} &= I_{zy} = \int\int\int (yz)\, dm \end{aligned} \qquad \begin{aligned} I_{xx} &= \int\int\int (y^2 + z^2)\, dm \\ I_{yy} &= \int\int\int (x^2 + z^2)\, dm \\ I_{zz} &= \int\int\int (x^2 + y^2)\, dm \end{aligned}$$

So that:

$$\begin{bmatrix} H_x \\ H_y \\ H_z \end{bmatrix} = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix} \begin{bmatrix} p_I \\ q_I \\ r_I \end{bmatrix}$$

The moment of inertia matrix, I, is fixed in the body-fixed frame while the Newton's Second Law only applies for an inertial frame. If the body-fixed frame is rotating with the rotation vector $\vec{\omega} = [p,\, q,\, r]^T$ then, for any vector $\vec{k}$, its derivative $\frac{d}{dt}\vec{k}$ in the inertial frame can be expressed as:

$$\left.\frac{d\vec{k}}{dt}\right|_I = \left.\frac{d\vec{k}}{dt}\right|_B + \vec{\omega} \times \vec{k}$$

Specifically, the Newton's Second Law becomes:

$$\vec{F} = m \left.\frac{d\vec{V}}{dt}\right|_B + m\vec{\omega} \times \vec{V_I}$$

and similarly for rotational motion:

$$\vec{M} = \left.\frac{d\vec{H}}{dt}\right|_B + \vec{\omega} \times \vec{H}$$

Thus, combining the relations got so far:

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = m \begin{bmatrix} \dot{u_I} \\ \dot{v_I} \\ \dot{w_I} \end{bmatrix} + m\, det \begin{bmatrix} \hat{x} & \hat{y} & \hat{z} \\ p & q & r \\ u_I & v_I & w_I \end{bmatrix} = m \begin{bmatrix} \dot{u_I} + qw_I - rv_I \\ \dot{v_I} + ru_I - pw_I \\ \dot{w_I} + pv_I - qu_I \end{bmatrix} \tag{13}$$

and

$$\begin{bmatrix} L \\ M \\ N \end{bmatrix} = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} + \vec{\omega} \times \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

$$= \begin{bmatrix} I_{xx}\dot{p} - I_{xy}\dot{q} - I_{xz}\dot{r} + q\left(-pI_{xz} - qI_{yz} + rI_{zz}\right) - r\left(-pI_{xy} + qI_{yy} - rI_{yz}\right) \\ -I_{xy}\dot{p} + I_{yy}\dot{q} - I_{yz}\dot{r} - p\left(-pI_{xz} - qI_{yz} + rI_{zz}\right) + r\left(pI_{xx} - qI_{xy} - rI_{xz}\right) \\ -I_{xz}\dot{p} - I_{yz}\dot{q} + I_{zz}\dot{r} + p\left(-pI_{xy} + qI_{yy} - rI_{yz}\right) - q\left(pI_{xx} - qI_{xy} - rI_{xz}\right) \end{bmatrix} \tag{14}$$

As for an aircraft there is simmetry about the x-z plane, stands out the approximation $I_{xy} = I_{yz} = 0$. The final expression for Forces and Torques simplifies as follow:

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = m \begin{bmatrix} \dot{u_I} + qw_I - rv_I \\ \dot{v_I} + ru_I - pw_I \\ \dot{w_I} + pv_I - qu_I \end{bmatrix} \tag{15}$$

and

$$\begin{bmatrix} L \\ M \\ N \end{bmatrix} = \begin{bmatrix} I_{xx}\dot{p} - I_{xz}\dot{r} - qpI_{xz} + qrI_{zz} - rqI_{yy} \\ I_{yy}\dot{q} + p^2I_{xz} - prI_{zz} + rpI_{xx} - r^2I_{xz} \\ -I_{xz}\dot{p} + I_{zz}\dot{r} + pqI_{yy} - qpI_{xx} + qrI_{xz} \end{bmatrix} \tag{16}$$

Thus, it can be stated that

- Traslational variables (u_I, v_I, w_I) depend on rotational variables (p, q, r);
- Rotational variables (p, q, r) do not depend on traslational variables (u, v, w);

4 Air data system

ADS has to provide pilots, FCS's and other on-board systems with several quantities derived from external air flow measurement through signals from aerodynamic and thermodynamic probes. Exploiting these air data probes, several measures are derived from the air surrounding the aircraft and then, through dedicated trasducers that are integrated into the Air Data Computer (ADC), converted in electrical signals. Through correction algorithms inside the ADS, local sensor measurements are calibrated and converted into free stream data. Then, electrical signals coming from trasducers are calibrated and eventually processed to calculate all the required parameters. As ADS' are critical for aircraft, in order to comply with safety regulations for airworthiness certification, are generally redounded two or even three times for specific safety requirements. This is particularly true for modern UAVs intended to carry out aerial work over populated areas: in case like this airworthiness regulations can even be stricter than those applicable to manned aircraft.
Air data probes and sensors, that are largely discussed by Gracey [23] and Wuest [24], measure directly air data parameters from air surrounding the aircraft.
In particular, these quantities are the static pressure (p_s), the total pressure (p_0), the total or static temperature (T_0 or T_s respectively) and air flow angles with respect to aircraft body fixed reference axis. From the static pressure is possible to compute the barometric height, H.
The total pressure, in addition to the static pressure, is used to calculate the relative velocity between the aircraft and the external air flow, known as indicated, calibrated or true airspeed (IAS, CAS and TAS respectively). Through the dynamic pressure ($q_{\infty,m} = \frac{1}{2}\rho V^2$) is then possible to calculate the IAS referred to sea level conditions.

$$IAS = \sqrt{\frac{2q_{\infty,m}}{\rho_{SL}}}.$$

Once the dynamic pressure measure is corrected from calibration errors, under incompressible fluid hypothesis, the IAS can be converted into calibrated air speed as

$$CAS = \sqrt{\frac{2q_\infty}{\rho_{SL}}}.$$

Eventually, using a temperature sensor to measure the actual value of air density, the true air speed can be calculated as

$$TAS = \sqrt{\frac{2q_\infty}{\rho_\infty}}.$$

The CAS, or IAS, are fundamental for piloting and controlling the aircraft, while the TAS is important for navigation purposes.
The Mach number is obtained from static and total pressure.
Air flow angles are direct measurements, of the local angle of attack, α, and angle of sideslip, β. Typically, the angle of sideslip sensing is operated by vanes as for angle of attack α, or less used, exploiting differential static pressure measurements from the two aircraft sides and by means a calibration law which convert the differential pressure measurement in the angle of sideslip calculation. Since the aircraft is symmetric with respect to x-axis, the expression for β can be semplified as

$$\beta = K_\beta(p_{s,L} - p_{s,R}), \tag{17}$$

where K_β usually is function of angle of attack, Mach number and also sideslip itself to take into account the non linear aerodynamic effects of fuselage nose which occur at high attitudes and in presence of high lateral wind.
It's important to remark a particular issue related to angle of sideslip. In fact, on civil aircraft AoS is generally not calculated because it is a trivial information for pilots in order to control the vehicle. On the other hand, for some fast-dynamics aircraft, such as

Figure 14: An example of unmanned flying wing vehicle. The *UCAV Boeing X − 45*

military ones, the angle of sideslip (and also the angle of attack in some cases) is used by FCS for stability and augmentation purposes. The same issues are related to UAV, where, due to the absence of human pilots on board, a big amount of information are needed

by FCS to fly automatically and safely the aircraft. For unmanned intrinsically stable flight vehicles the angle of sideslip is especially required for landing and takeoff phase in cross-wind. For unstable UAVs, e.g. on flying wing models, the angle of sideslip is required by the control system, because of lack of stability on the vertical axis according to their peculiar aerodynamic configuration. Several unmanned unstable models exist such Unmanned Combat Air Vehicles (UCAVs) *Boeing* $X-45$ (see Fig. 14), *Dassault nEUROn* and *Northrop Grumman* $X-47$.

4.1 ADS state of the art

In this section, will be outlined typical ADS architectures for both civil and UAV applications and exposed their main differences. Initially, ADS intended for current civil aircraft will be presented, where parameters as IAS, TAS, barometric height and Mach number are strictly required for navigation and control. Though the angle of attack is always measured, this information is not usually displayed to pilots but it is only used by the stall prevention system to warn in case of incoming aircraft stall. Secondly, ADS for UAV applications will be presented, where the IAS, TAS, Mach number, barometric height, angles of sideslip and attack are required by the flight control system for automatic control.
In order to give an idea of how ADS' could be designed, several configurations will be presented using realistic (but not actual) architectures. Indeed, the purpose of these following representation is only to compare pro and cons of designed architectures using both off-the shelf and state-of-the-art sensors.
A simplex standard civil ADS configuration for stable aircraft, is illustrated in Fig. 15. The word *standard*, when referred to ADS, is used to indicate the use of off-the-shelf sensors and ADC. The Pitot tube measures the total pressure, the left and right flush ports sense the local static pressure and are pneumatically averaged together to minimize sideslip effect (commonly for $\beta < 10$). Tipically, the angle of attack, α, is measured using vanes, while the air temperature is obtained from dedicated Static Air Temperature (SAT) or Total Air Temperature (OAT) probes. Besides, even if the angle of sideslip, β, is not required by the majority of civil aircraft, it could be measured, for example, using an additional vane. All measured data are converted into electronic signals and then processed by the ADC in order to be converted into known quantities, such as IAS, TAS, barometric height, Mach number and angle of attack,α. In few cases only, such as military or unstable aircraft, the angle of sideslip, β, is provided as well.
Some simplifications can be introduced in the standard architecture of Fig. 15, if state-of-the-art, or *advanced*, probes are used instead of the standard ones, as depicted in Fig. 16. Basically, advanced probes are multi function probes able to measure three quantities using only one external sensor: total pressure, static pressure and one or two flow angles. Several models exist, such as the integrated *Goodrich SmartProbe*© used on some *Embraer* business regional aircraft and on *Airbus* $A400M$, the *MFP* (which is a self-orienting cone) manufactured by *Aerosonic Corp.* used on *Leonardo* $M346$ and the self-orienting probe manufactured by *Thales* (which is a flow angle vane with pressure ports for total and static pressure sensing) used on the *Dassault Rafale* and on $EF2000$ (lincesed to *Marconi Avionics* by Thales). All these probes are integrated with an electronic box (Air Data Unit, ADU) incorporating: pressure and position transducers,

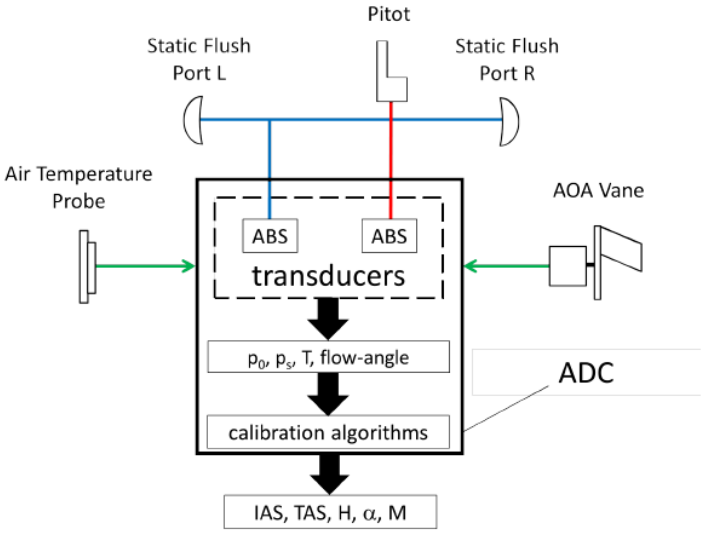


Figure 15: A realistic standard ADS architecture for civil aircraft. The red line represents the total pressure lane, the blue lines stand for the static pressures which are actually connected, the green represents electrical signal. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

computing and self-test capability thus eliminating the need for a central ADC and the need for pressure tubing running along the airplane, as described in Fig. 16, and it will be referred in this work as *advanced* ADS. In general, each aircraft is provided with a particular ADS architecture to respond to peculiar requirements. Actually, another state-of-the-art architecture, which exploits nonobtrusive sensors, is used by hypersonic, stealth, and some research aircraft for their specific purpose. This system, known as Flush Air Data System (FADS [5]), consists of multiple flush pressure ports distributed on aircraft surface,especially on the nose . An ADC, or a FCC, processes the pressures and returns a complete set of air data as depicted in Fig. 17. For this purpose, equations for potential flow around a sphere are used and then corrected, with calibration coefficients, for nonpotential and nonspherical flow. In general, FADS measures more pressure than the minimum required for the air data state: it is then referred as an overdetermined system. This allows to exclude by calculations any inaccurate pressure readings, therefore improving its robustness with respect to other ADS presented in this section. Furthermore, FADS is also used by some special aerial-space research vehicles, such as the *Lockheed Martin* $X-33$, which use to their own calibration algorithms to convert the raw pressure measurements into requested set of air data.

In the next future, another class of nonintrusive sensor may be used, the optical airdata sensors. These sensors use lasers to measure the speed, flow angles and temperature of the air. They only need a flush window in the airplane for the laser equipment.

Leaving behind ADS' for civil and military piloted airplanes, those for automatic control and navigation of any kind of aircraft, manned or unmanned, stable or unstable,

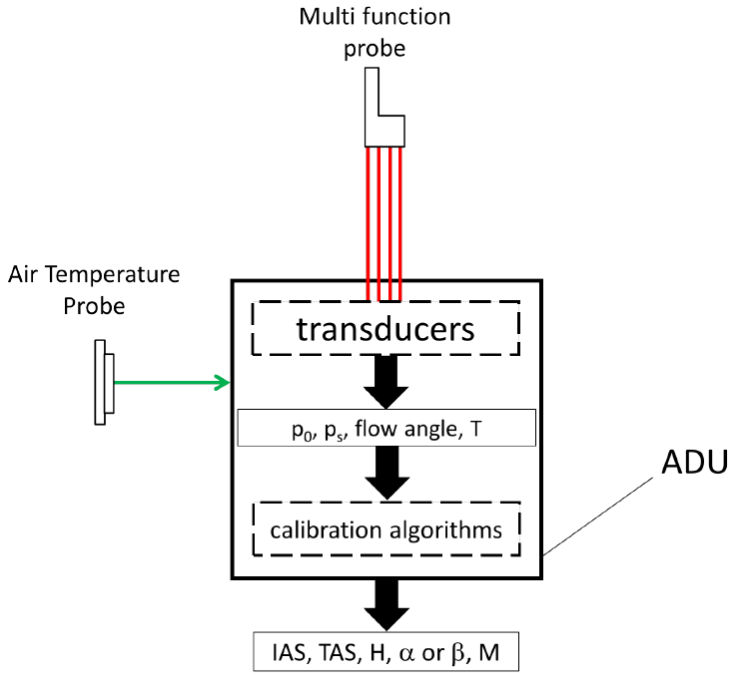


Figure 16: A possible *advanced* ADS architecture for civil aircraft. The red lines represents the electrical output signals. From Angelo Lerro, "Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

require the following *complete* suite of air data

$$\{p_0,\ p_s,\ T_s \text{ or } T_0,\ \alpha,\ \beta\}.$$

In order to safely fly along the entire flight envelope, from take-off to landing, unmanned aircrafts require the complete air data set. At present, UAVs usually are equipped with air data boom (represented in Fig. 18) which provides a complete set of air data. It can be built starting from a common nose booms or five-hole probes [31]. The air data booms usually represent on UAV the primary and the sole ADS for their ability of measuring free stream air data not requiring any calibration algorithms, and for its installation simplicity. For these two reasons this kind of ADS often operates as calibrating system and hence it is indicated as *flight test* air data system. Though the air data boom solution shows large advantages, on the other hand it has relevant interference with the camera field of view and make the air data boom not completely feasible to be redundant on UAV.
First of all, a possible ADS for unmanned flight vehicles, based on *standard* probe technology, is depicted in Fig. 19 very similar to that presented in Fig. 15 with the addition of another static pressure sensor which allows to calculate the angle of sideslip, β, using left and right static pressure difference in (17). Though this architecture (the group of probes, sensors and ADC) has the big advantage to be economic, it requires a lot of external sensors mounted on the fuselage, and therefore it is not very convenient to redound this system for modern UAVs.

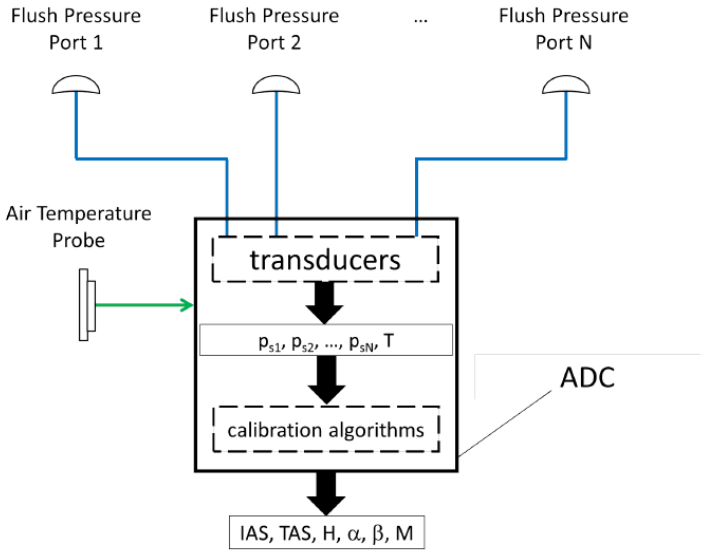


Figure 17: FADS architecture. The red line represents the electrical output signal, the blue lines represent static pressures lanes. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

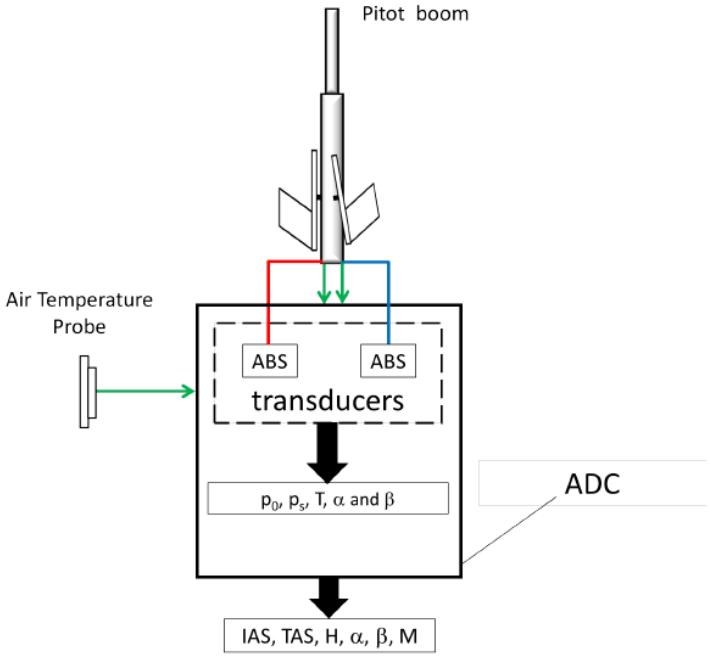


Figure 18: A standard ADS boom architecture for UAV aircraft. The red line represents the total pressure lane, the blue lines stand for the static pressures which are actually connected, the green represents electrical signal. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

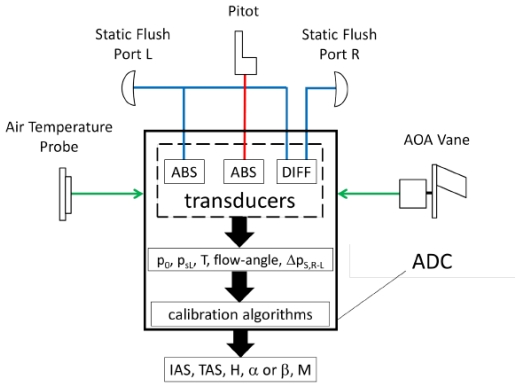


Figure 19: A possible ADS architecture for automatic control and navigation using *standard* probes. The red line represents the total pressure lane, the blue lines are for the static pressures, the green represents electrical signal. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

The last ADS architecture, presented in Fig. 19, can be simplified to a great extent using advanced probes. Indeed, a possible advanced ADS for UAVs, able to provide a complete set of air data, is represented in Fig. 20. In this latter case, the multi function probes are adequately mounted on the nose of the aircraft on opposite sides to sense correctly the angle of attack, in addition to left and right static pressures and two total pressures. Even if redundant measures of α and P_0 are not required for a simplex system, the two measures of static pressure are necessary to calculate correctly static pressure when $\beta \neq 0$ and fundamental to be used in (17) in order to calculate the angle of sideslip, β. Clearly, since one of the multi function probe works only as static pressure source, it could be easily replaced with a standard static port. Even if it is a possible solution, it will not be taken into account here since it only reduces the overall ADS costs that, in this work, are approached only in a qualitative way. Actually, the advanced system results always more expensive than the standard one of Fig. 19, on the other hand it allows to reduce the number of external probes from 5 to only 3 (two multi function integrated probes and and one temperature sensor). To outline all possible ADS' for UAV and better remark their relative differences, in Fig 21 are depicted three possible solutions for ADS' with as many different technologies: air data boom, classic or advanced ADS'. The temperature sensor is here not represented since it operates on each system. In order to comply with current safety requirements, one of ADS represented in Fig. 21, which is able to provide the complete set of air data, need to be physically duplicated, or even triplicated, to let UAVs fly in common airways to carry out aerial works. For this reason, the air data boom of Fig. 21(a) is not suitable to be redounded since it interferes with electro-optical sensors, as stated before. Although the ADS which exploits standard probes (Fig. 21(b)) can be easily duplicated, the large number of external probe required leads to mounting positions related issues. Since the ADS system is very complex, the use of advanced probes (Fig. 21(c)) permits to redound the system using less external probes than the standard ADS, but with higher costs. As usual in the engineering field, the best solution is that which represents

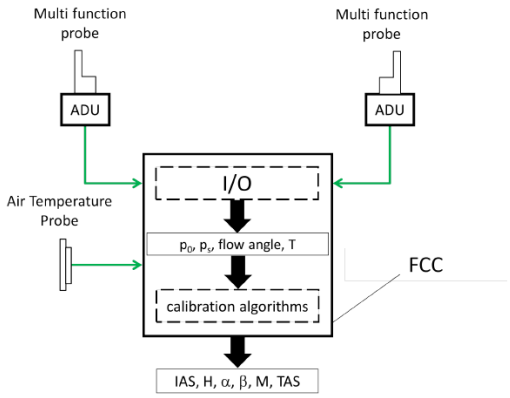
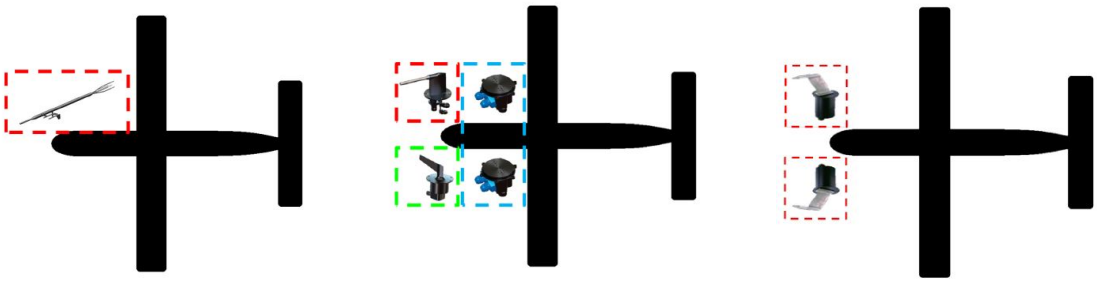


Figure 20: An advanced ADS architecture for automatic control and navigation. The red lines represents electric signals. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].



(a) Air data boom (b) *Standard* ADS (c) *Advanced* ADS

Figure 21: Possible ADS architecture according to current air data technologies. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

the best trade-off: the redundant ADS design is always a choice of compromise between technical requirements, performance and costs.
In this context, the introduction of a software-based synthetic sensor to redound physical sensors may represents a efficient way to save external sensors with all the related benefits that will be discussed in the following section.

5 Synthetic Sensor Design

In this Section the virtual sensor will be presented, from its first concept till its final version.
The virtual sensor concept was born after studying some ADS multi-probe architecures proposed for MALE (Medium Altitude Long Endurance) applications and later realizing that a fully suitable architecture was not available on the market for this kind of aircraft due to their intrinsic complexity when redundant systems are required. Indeed, some technological issues exist: for example, the need to keep costs low, which is an important driver in the UAV market, often opposites the need to make innovation; besides, the need to keep the aircraft weight down in order to increase the payload capability in terms of weight and volume, usually opposites the certification regulations, which request redundant systems for safety reasons. For what concerns ADS, many other issues exist. It's remarkable, for example, that most of ADS sensors need to be installed in the forward part of the UAV fuselage, which is already taken up by payload, avionic equipment, radomes, antennae and opto-electronic sensors specific for the aircraft mission. Another big issue is represented by the so called *bird strike* event. Indeed, since the reduced size of UAV, it is usually diffucult to assemble several external air data probes or sensors adequately spaced out to comply with *bird strike* certification. Therefore, in light of these, and other minor, considerations, the needs of reducing actual sensors emerged as a primary importance issue and the virtual sensor concept was conceived. Since both total and static pressure may be calculated using standard Pitot-static tubes, a big simplification (see Fig. 22(a)) to ADS architecture may come from the use of virtual sensor for aerodynamic angles estimation. Therefore, in this work, the virtual sensors are developed for aerodynamic angle estimation with the aim to be used both as primary source of data both as stand-by system to be used for voting and monitoring purposes, decreasing the level of redundancy.
At the beginning, very few design requirements were set to bound and drive the project, as follows

- it must have comparable performance with current actual sensors: maximum errors within ± 1.5 *deg*;
- it must be run in few milliseconds on real FCC, in order to work in real time;
- it may use all data available at the FCC as other inputs.

A tolerance band of ± 1.5 *deg* has been defined according to experience gained in Leonardo S.p.A. Aircraft Division and it is due to three main contributions: error of current angle vanes, which is within ± 0.4 *deg*, error due to calibration algorithm, which is within ± 0.3 *deg*, and error from the installation, which is within ± 0.3 *deg*. Each of

the three error contributions could be further expanded in sub-contributions, but it goes beyond the goal of this work.
The most important peculiarity of a synthetic sensor is that it provides indirect measures of aerodynamic angles whereas all other sensors built until today measures aerodynamic angles directly through some pressure measurements from the airflow surrounding the aircraft. Hence, an innovative way to estimate aerodynamic angles was required in order to replace physical sensors exploiting the flight data already measured on-board, such as inertial data. Eventually, a synthetic sensor could be based on an aircraft model or on neural network: our choice fell to the second option and the reasons of this decision are exposed in the following sections.

5.1 Project Requirements and Objectives

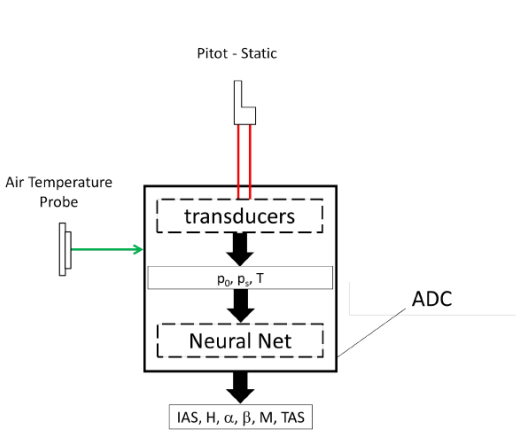
A neural network based processing system has been proposed because it's an easier to implement and more stable alternative with respect to model based techniques.
As far as mathematical model based technique concerns, an aircraft model is required in order to define the state vector, its derivative, and hence its ODEs to be solved to obtain the desired measures of aerodynamic angles. Modeling a very complex system, like an aircraft is, will always introduce some discrepancies with the real-world system. Indeed, building a reliable aircraft mathematical model is quite intricate: generally the aircraft model and its subsystems, are approximated and it may require several tuning iterations after comparison with actual flight test data. In view of these considerations, neural networks can be more useful than model based technique. Indeed, neural networks allow to model a real-world system without any information about the dynamic model, just exploiting observed input and output patterns. On the other hand, it is worth to be remarked that a knowledge of actual system analytical equations, which describe its working and evolution, extensively helps engineers to better design the neural network.
Besides, even if a very accurate aircraft mathematical model is available, the virtual sensors is needed to be run in real time mode: indirect measures of aerodynamic angles need the aircraft model to run in very short time, of the order of magnitude of milliseconds, on actual FCC, which are much slower than modern personal computers. This issue represented a big problem to be solved at once, using model based technique. In addition, programming a mathematical model on aircraft FCC is not a simple task due to time consuming recursive methods to implement, and for safety reason which always suggest to avoid sub-iterations to converge at each time step, when it is possible. On the other hand, whatever complicated neural networks could be, they simply reduce to matrix calculation using particular activation functions. Hence, neural nets overcome the drawback of *time consuming software* and the presence of inner loop.
The main drawback of virtual sensor with respect to model based technique is represented by the meaningless of network weights, whilst coefficients of aircraft mathematical models have always physical meaning. This is a crucial aspect of virtual sensor in real life operations. For instance, consider the changing of the aircraft center of gravity (CG) by simply relocating the payload. In order to maintain the same virtual

sensor performance, the neural network should be retrained with the new weight and balance configuration. On the other hand, if a mathematical model is used, changing the CG is a very simple operation. Therefore, it clearly comes out as very slight change in the aircraft configuration could degrade the virtual sensor performance and, at least, it must be investigated, also using a simulator, to avoid un-expected errors.
The correct construction of a neural network needs that the training set accurately and adequately represents the function which is meant to be learned, as described in 6.5. For this task, several maneuvers at different speed are required in all possible flight scenarios (flap extension, still or turbulent air, and so on) in order to cover the entire flight envelope. Obviously, this is not realistic, and the number of sample maneuvers, and thus training data points, will be reduced to be managed by a common workstation, as will be described later. Moreover, the optimum architecture and degree of training has to still be determined heuristically during the training process and not a priori.
In conclusion, the characteristics of neural networks match initial requirements better than model based technique and this is the reason why they were selected to develop the virtual sensors.

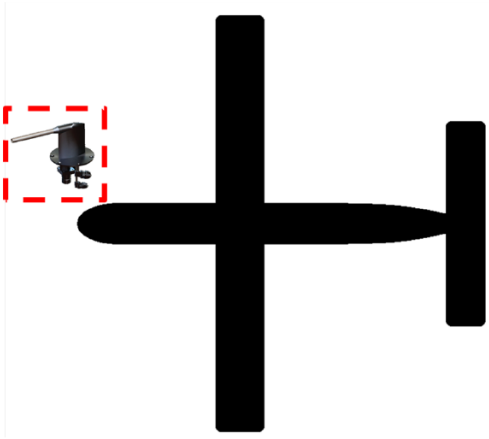
5.2 Trade-off: Advantages and Drawbacks

All presented ADS solutions in section 4.1 are realistic layout considering sensors available on the market. Since the basic set of air data is required in any aircraft equipped with autonomous control system, the issue is how to design the ADS according to the desired type of sensors. In this section an analysis on each system, presented in section 4.1, will be carried out in order to highlight some advantages and drawbacks of each one, and better compare the virtual sensor with respect to the actual ones.
The system depicted in Fig. 19 has the enormous advantage to be made up of very consolidated technology and available on the market by several manufacturers. The main drawback is the number of external devices with consequent complexity due to wirings, piping and mechanical constraints.
An advanced ADS presented in Fig. 16 let engineers to save two external sensors, but the most of these sensors need to be installed in the front part of the fuselage that could be a problem for some UAVs. Even if the cost evaluation goes beyond this work, it is clear that such a system could be more expensive than the first one.
Since FADS' (described in Fig. 17) are more complicated than other architecture mentioned here, they are suitable for very particular application where a flush system is the only technological solution that can be followed. The virtual sensor can be used in a realistic architecture as depicted in Fig. 22(a), carrying the following advantages with respect to other modern architectures:

- save external actual sensors and related costs;
- save weight;
- save space onboard;
- cut emissions (mainly for anti-icing purposes).



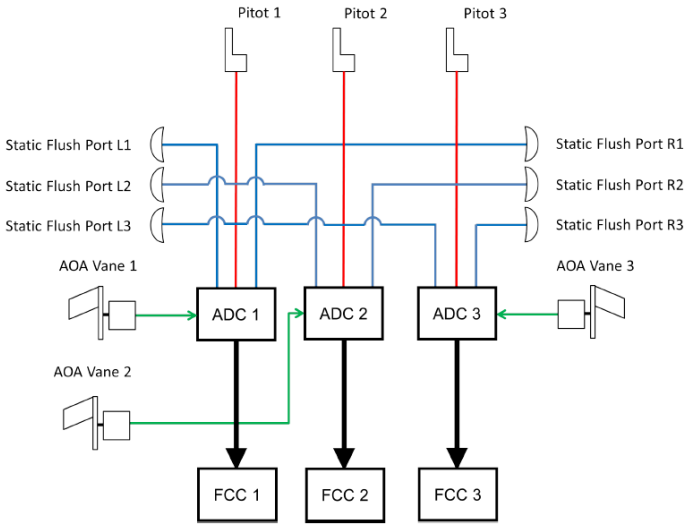
(a) ADS with neural networks



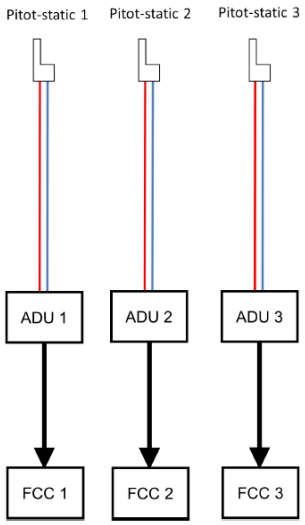
(b) Possible ADS architecture when using NNs

Figure 22: A possible ADS architecture using virtual sensors for aerodynamic angles estimation based on neural network. From Angelo Lerro, "Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

Moreover, it is clear that comparing ADS of Fig. 22(b) with all other possible current ADS represented in Fig. 21, can be easily redounded. In fact, as Fig. 23 shows, a realistic triplex ADS architecture can be simplified a lot introducing virtual sensors for α and β, saving at least nine external off-the-shelf probes: six static flush ports and three α vanes.



(a) Triplex ADS using off-the-shelf air data probes



(b) Possible triplex ADS architecture exploiting NNs

Figure 23: Realistic triplex ADS architecture using, or not, virtual sensors for aerodynamic angles estimation based on neural network. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

Moreover, another advantage when using virtual sensors is the lack of need for dedicated ADC. Indeed, In Fig. 23(b) the ADC are substituted by ADU, which are cost saving with respect to more complex ADC because do not have calculation capabilities but only a suite of transducers.
Overall, as far as analytical redundancy concerns, a virtual sensor, based on neural network theory, has been identified as the best strategy to reduce complexity due to current redundant ADS sensors.

6 Neural Network

6.1 Historical Background

In 1955 McCarthy et al. coined the term artificial intelligence (AI) to identify that branch of computer science where some techniques have in common the ability of human mind to reason and learn in an environment of uncertainty and imprecision. The aim of AI is to create intelligent machines mimicking the human intelligent behavior by expressing it in language forms or symbolic rules . Within computer science, *Soft*

computing belongs to artificial intelligence branch. According to Zadeh , soft computing aims to adapt to the pervasive imprecision of the real world, unlike traditional, or hard methods of calculation. Its guiding principle can be expressed as: "exploit the tolerance for imprecision, uncertainty and partial truth in order to obtain tractability, robustness and low cost solutions". Soft computing includes three human inspired techniques: artificial neural network (ANN), fuzzy logic (FL) and genetic algorithm (GA). GA is search heuristic that mimics the process of natural evolution. The fuzzy logic is based on calculation by using linguistic labels stipulated by such functions, called membership functions. Indeed, a selection of if-then rules are the core of the fuzzy inference system that can model human expertise in some specific application. Inspired to human brain structure, neural networks are expected to mimic brain mechanism to simulate intelligent behavior, simply using conventional matrix calculations. In 1904, Cajal introduced neurons as basic component of human brain; later many researchers were involved to reproduce mathematically the complex, nonlinear and parallel computer which is human brain. After few decades several artificial intelligence techniques have been completed and fully demonstrated, but due to high calculation power required they did not spread very much. The first mathematical neuron model was was proposed by Rosenblatt in 1958 [50] , called perceptron model, which is still widely used in neural network field. Anyway, rigorous demonstration had a very long course which lasts about forty years and were proposed only after successful neural network applications. The fundamental equation 5(18) of feed-forward neural network units, the neurons, is very similar to expression proposed by McCulloch et al. already in 1943 to mimic nervous activity. Only starting from 1987 several authors, Hecht-Nielsen , Lippmann and Spreecher , suggested that Kolmogorov's theorem (1957) provides theoretical support for neural networks that concerns the realization of arbitrary multivariate functions.

Kolmogorov's Theorem 1. *Any continuous real-valued functions*

$f(x_1, x_2, ..., x_n)$ *defined on* $[0, 1]n$*, with* $n \geq 2$ *can be represented in the form*

$$f(x_1, x_2, ..., x_n) = \sum_{j=1}^{2n+1} g_j \left(\sum_{i=1}^{n} \Phi_{ji} x_j \right),$$

where the g_j*'s are properly chosen continuous functions of one variable, and the* ij*'s are continuous monotonically increasing functions independent of* f*.*

(Kolmogorov's Theorem) Any continuous real-valued functions
$f(x_1, x_2, ..., x_n)$ defined on $[0, 1]n$, with $n \geq 2$ can be represented in the form

$$f(x_1, x_2, ..., x_n) = \sum_{j=1}^{2n+1} g_j \left(\sum_{i=1}^{n} \Phi_{ji} x_j \right),$$

where the g_j's are properly chosen continuous functions of one variable, and the ij's are continuous monotonically increasing functions independent of f.

In contrast, other authors, such as Girosi and Poggio , have criticized this interpretation of Kolmogorov's theorem as irrelevant to neural networks by stating that the Φ_{ij}

functions are highly non-smooth. As this debate continues, the importance of Kolmogorov's theorem is pointing the feasibility of using parallel and layered network structures for multivariate function mappings, and not proving the universality of neural nets as function approximators. Few years later, other authors independently, Cybenko , Hornik et al. and Funahashi , proved analytically that one hidden layer feed-forward neural network is capable of approximating uniformly any continuous multivariate function, employing continuous sigmoid type (instead of non-smooth $\Phi_j i$), as well as other more general activation functions. From the 1980's, thanks to computing progress neural networks can be exploited also using common computers, even if the human brain is still a mirage. Indeed, although silicon gates are six order of magnitude faster than human neurons, human brain is able to process a big amount of information much faster than modern computers.

6.2 Theoretical Background

According to Arbib , the nervous system can be described as in Fig. 24, where the human brain continuously receives information from environment (stimulus) and make decisions (response). The two sets of contrary arrows indicates that the brain can communicate with the receptors and effectors, it is common to say that the arrows pointing right indicate the forward transmission, the others indicate the feedback transmission. In other words, the response could be sensed by brain and also used as an input. Continuing the

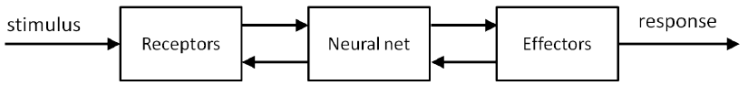


Figure 24: A schematic representation of nervous system. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

parallelism between human brain and neural network, the neurons under stimulation send out electrical pulses (or spikes) to communicate with other neurons connected with itself using a particular connection (synaptic conections). The engineering neuron model (see Fig. 25) has the same characteristics of human ones: stimulated by input signals (x), it elaborates by means a mathematical function (or activation function f) an output signal (y) which is sent by dedicated links (w) to other neurons. According to perceptron neuron model of Fig. 25, the $j-th$ neuron is mathematically described by Eq. 18

$$y_j = f_j(v_j) = f_j\left(\sum_{i=1}^{n} w_{ji}x_i + b_j\right). \tag{18}$$

In order to highlight the artificial neural network working and the importance of bias role, consider, for example, to model a thermocouple , the governing equation of which can be approximated as in Eq. 19 for certain range of temperatures,

$$V_{out} = K_T \Delta T + b \tag{19}$$

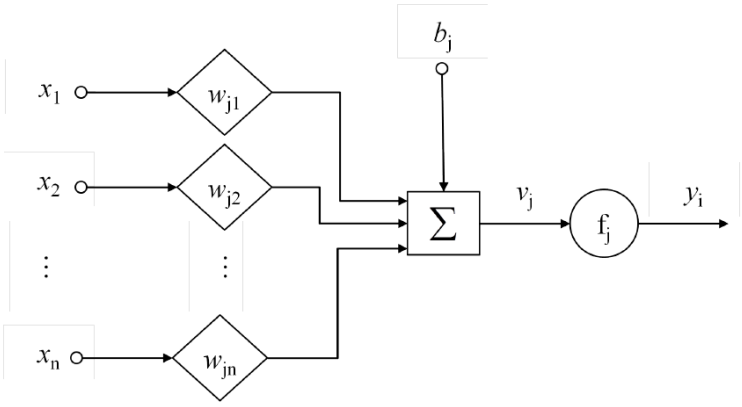


Figure 25: Perceptron neuron model with non-linear activation function f_i and bias b_i. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

using a neural network with only one neuron and a linear activation function ($f = 1$). The neural network approximation will be

$$\hat{y} = w_{11}x_1 + b_1. \tag{20}$$

It is clear from Eq. 20 the importance of the bias for any single neuron and the weight, w. After the learning process, obviously the neural network will be as much accurate as the weight w_{11} is closer to K_T.
Once proved that neural networks can approximate continuous multi variable functions, the issues regards training methods to be used in order to optimize free parameters of neural networks. This process is known as the *learning process*. Indeed, the neural network training is very often viewed as one of key-to-success in using NNs. The important roles of the neuron model in Fig. 25 is played by link weights, w, optimized using one of numerous learning algorithms, and by the activation functions. These two topics will be treated in next two sections. Generally speaking, a neural network is made up of several neurons, organized in different layers, as depicted in Fig. 26. This particular neural network estimates two outputs, $\hat{y}_1$ and $\hat{y}_n$, has n inputs in the input layer, r neurons in the h hidden layers and two neurons in the output layer. From Fig. 26 is quite easy to compare the mathematical to the human nervous system of Fig. 24. The sensory unit are the source node that constitute the input layer, the hidden layers represents the neural net and the effectors are the computation node in the output layer. This kind of neural network is commonly referred to as multilayer perceptrons (MLPs). Each neuron of Fig. 26, n_{hr}, in turn contains the scheme of 25, therefore the outputs are calculated as in Eq. 21

$$\hat{y}_{1,2} = f_{o1,2} = \left(\sum_{l=1}^{r} f_{hl} \ldots \left(f_{1i} \left(\sum_{k=1}^{n} w_{ik}x_k + b_{1i} \right) \right) \right). \tag{21}$$

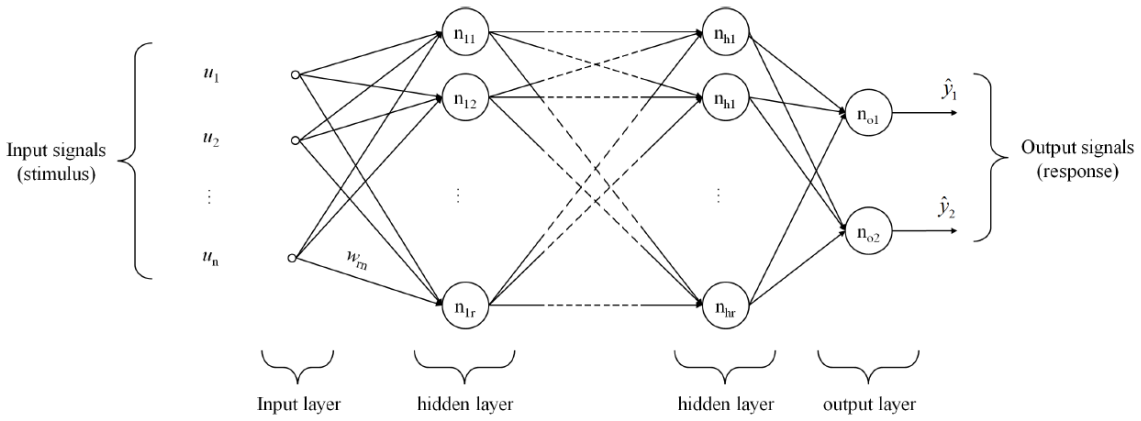
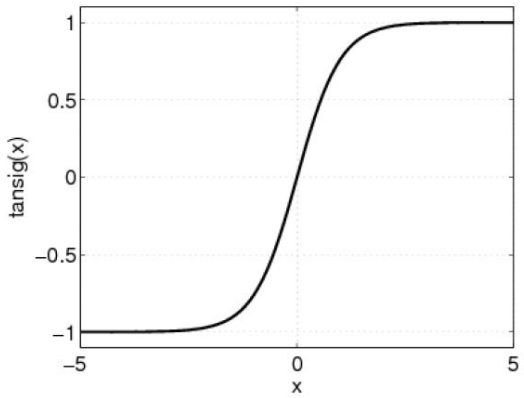


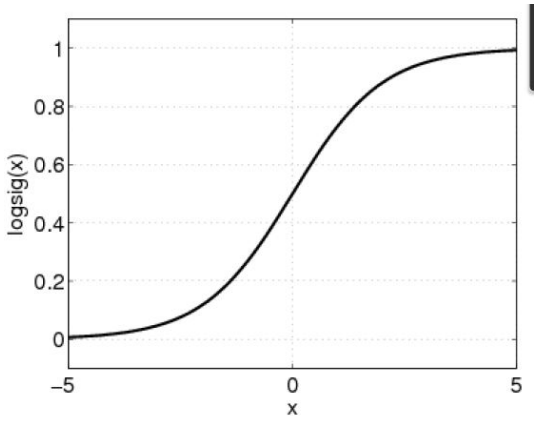
Figure 26: General architecture of neural network. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

6.3 Activation Functions

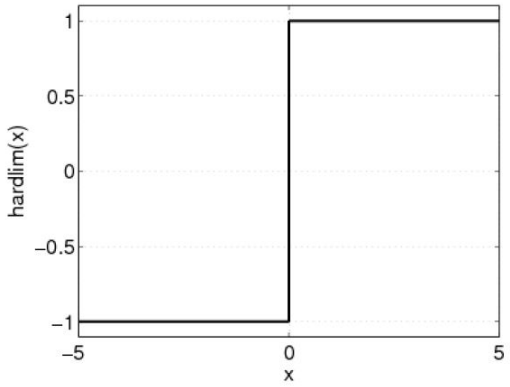
Any kind of mathematical activation function can be used, paying attention that it is defined in the input domain. Anyway, there are some special function, as well as sigmoid function, which are widely used because of their demonstrated ability to approximate continuous systems in addition to the great advantage of limited output. Consider the model described by Eq. (19) and using the linear neuron (21) to approximate the system. If the input increases for such a reason, such as noise or external disturbances, beyond the linearity range, the Eq. (19) will not be longer valid to model the system, but the neural network will continue to work as a linear function. Hence, the output will increase as input increases magnified by the factor w_{11}. This issue is better known as neural network extrapolation: if the training data set does not contain that maximum possible output value, an unmodified network will be unable to recognize this peak value. The example is quite bold, but can give the idea of the problems encountered with real-life application of neural network. Therefore, in order to avoid that the neuron output could increase very much, limited activation functions are preferred as activation functions for hidden neurons. Here, a list of nonlinear activation function considered for this work is provided in Fig. 27. The most used neuron function in this work is the sigmoid function. For the same reason, the linear function (Fig. 27(d)) are commonly used as activation function of the output neurons, otherwise, using a limited function, the neural network would not be able to extrapolate beyond the training limits. There are several non-linear activation function used as hidden neuron function, such as the hard limitier (Fig. 27(c)), that has the drawback to have non-continuous derivatives. The logarithmic sigmoid function (27(b)), positive defined, has been demonstrated less suitable because of worse performance for the present application.



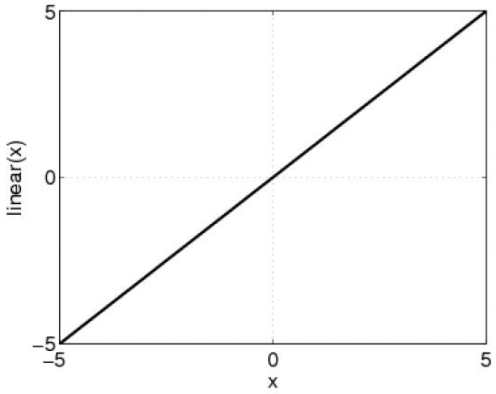
(a) sigmoid function



(b) log-sigmoid function



(c) hard-limiter function



(d) linear function

Figure 27: Examples of activation functions. From Angelo Lerro, "Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

6.4 System identification Methods

In order to study dynamic behavior, design control systems and improve performance of real systems, mathematical model are used instead of actual test to reduce cost and time. For the development of an aircraft model a substantial amount of work is allocated to build the mathematical model starting from known dynamic equations and experimental parameters, as well as wind tunnel data, using several approximation degrees in agreement with the level of accuracy required to simulations. The observed data from the real systems are used to tune some of the uncertain parameters related to approximations made. Consider the sample aircraft system of Fig. 28, where a general case is depicted, clearly the main block is made up of several sub-block, each of them mathematically reproduce a real sub system, as well as the engine block. The signals manipulated by an observer, such as pilot commands, are called inputs, the observable signals, as well as vertical acceleration, are called outputs, the external disturbances that can be measured, such as compressor rotating velocity, or cannot be measured, such as actual air turbulence, are indicated as well. Real-life actual systems are always very complex to reproduce, for the numerous parts and sub systems that are very complicated to model mathematically in particular. Therefore, engineers must often direct towards other techniques different from conventional mathematical model approach. System identification methods are widely used when the complexity of the system and the processes involved in itself are so high, because the model can be built even if governing equations are completely unknown, indeed the model building is only based on observed data from the actual system: the input and output signals are recorded and subjected to data analysis to infer a model. The key of this route is the evaluation of the so-called regression vectors, which are able to predict the future output of the system. Now, a simple linear system which satisfies the Eq. (22) is described in order to introduce some quantities and notations that will be used in this work.

$$y(t) + a_1 y(t-1) + \ldots + a_n y(t-n) = b_0 u(t) + b_1 u(t-1) + \ldots + b_m u(t-m). \quad (22)$$

It has been chosen to represent the system in discrete time because the observed data are always collected by sampling. The same system can be written in a deterministic way, see Eq. (23), where the output at the present time is expressed as function of previous observations

$$y(t) = b_0 u(t) + b_1 u(t-1) + \ldots + b_m u(t-m) - a_1 y(t-1) - \ldots - a_n y(t-n). \quad (23)$$

Eq. (23) can be rearranged in a more compact notation introducing vectors

$$y(t) = \phi^T(t)\theta. \quad (24)$$

where

$$\phi(t) = [u(t), \ldots, u(t-m), -y(t-1), \ldots, -y(t-n)]^T, \quad (25)$$

is the regression vector and its components are the regressors, and

$$\theta = [b_0, \ldots, b_m, a_1, \ldots, a_n]^T. \quad (26)$$

Such a model described in Eq. (22) calculates the output "regressing" or going back to the regression vector, $\phi(t)$. It is also partly "Auto-Regressive" since the regression vector

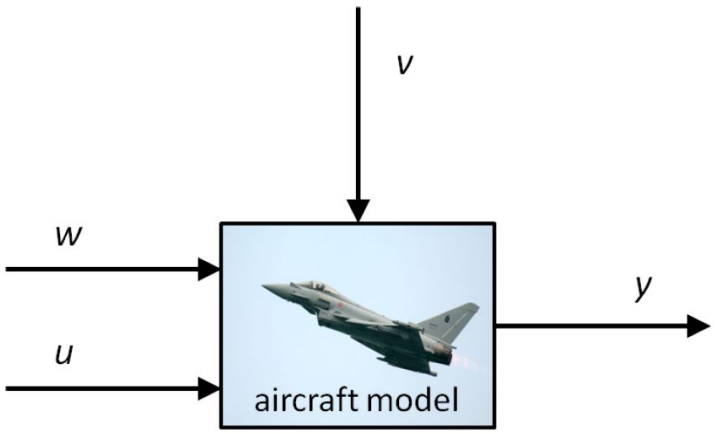


Figure 28: A system with input u, output y, measured disturbances w and unmeasured disturbances v. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

$\phi(t)$ contains old values of the variable to be explained, $y(t)$. The model structure of Eq. (22) has the standard name of ARX-model: Auto-Regression with eXtra inputs (or eXogenous variables). Once the regression vector structure has been defined, how many old outputs and inputs using, the ARX-model performance lays on the definition of vector θ. The ARX-model often cannot be used in real life application because the exact old output, y, is not available, so the ARX-model can be modified in the Output-Error model (OE), for which the regression vector has the estimated output as regressors

$$\phi(t) = [u(t), \ldots, u(t-m), -\hat{y}(t-1), \ldots, -\hat{y}(t-n)]^T, \tag{27}$$

The regression vector (27) of OE-model was used in this work because of the lack of exact output available. However, the use of output in the regression vector is almost tricky because at any time step some errors are inserted by means of old estimated output. If the model is supposed to work for a long time, some expedients are necessary to overcome the uncontrolled growth of errors.

6.4.1 Multilayer Perceptron

The single perceptron model was introduced and demonstrated by Rosenblatt [50], and it remains the simplest form of neural network. As foresaid, the neural network depicted in Fig. 26 represents a common multilayer perceptron, which has been successfully applied in the past to solve several problems by training with a very popular algorithm known as the error back-propagation algorithm, first proposed by Werbos [57]. This algorithm belongs to the error correction learning process, and it is characterized by two ways of propagation through the layers, the forward pass and the backward pass. In the forward pass, the input vector is applied to sensory nodes of input layers, and its effects propagates layer by layer to the output node. Here a set of output are calculated and compared with the actual, or desired, target. During the backward pass, the error flows from the

output node until to the first hidden layer, hence the name error back-propagation (see Fig. 29). The synaptic weights and bias levels are adjusted in order to minimize the distance between neural network response and desired target by an error correction rule. Anyway, whatever is the learning algorithm, a very common learning loop is followed, as depicted in Fig. 30. In the first step synaptic weights and bias levels (neural network free parameters) are initialized, then the neural network response is calculated using the available learning, or training, data set. The estimation error is calculated with respect to a desired target d. The error is used to update the free parameters according to chosen learning algorithm until the convergence criterion is satisfied.

6.4.2 Neural Network Structure Selection

The purpose of this section is to introduce the link between the system identification and the neural network. Sometimes, neural networks are referred to as black-box identification technique for nonlinear dynamic systems, because of the non physical sense of synaptic weight. Designing neural networks is subjected to the same rules of system identification presented at section 3.2. Therefore, the neural network structure can be chosen with the following two steps:

- selecting the inputs to the network
- selecting an internal network architecture.

Where, the first step is also equivalent to selecting the model, such as ARX or OE model, or better NNARX and NNOE model if referred to neural network. Indeed, firstly, there is the choice of inputs to the network among the available signal, and secondly, how many old, or past, use as input data. Consider, for example, the single input single output neural networks of Fig.s 6.4.2, 6.4.2. Since there is only one input, the choice of input signal is forced, but in general it requires a detailed analysis of the system at hand to establish which are the necessary inputs. Here the system analysis refers to a mathematical description of the real-world to be modeled.. The same is applicable for the output signal. Anyway, the numbers of old inputs and outputs is chosen by the user according to his previous experience or real-world system analysis. Moreover, generally speaking, each input, and output, may have its own number of past inputs, even if this practice is not very common. Once the first step is complete, the regression vector is defined as in (25) for NNARX model and (27) for NNOE model. The neural network architecture, such as the choice of neuron and hidden layer numbers, is defined in order to meet the network performance specification and best generalization behavior. Performance is defined in the next section, generalization will be discussed in section 3.3.6. Anyway, once the architecture is identified, the vector θ is also defined and made up of neural network free parameters. As described in this section, neural network problems can be considered in the same way of system identification ones. Therefore, all the known optimization techniques suitable for system identification can be used to optimize, or train, neural networks.

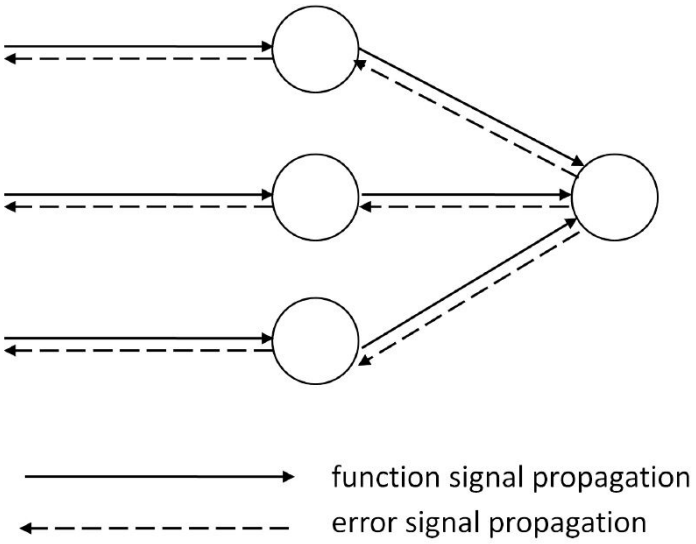


Figure 29: Illustration of the two propagation ways for signals and errors in the back propagation algorithm. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

6.5 Neural network training

In 1970 Mendel and McClaren [46] defined the learning process as "the process by which the free parameters of a neural network are adapted through a process of stimulation by the environment in which the network is embedded. The type of learning is determined by the manner in which the parameter changes take place". Definitely, the training process is a set of well defined rules that allows the neural network to learn from an input-output data pattern and hence to adapt itself to its environment. In other words, during the learning process the synaptic weights are optimized with the aim to improve so far the neural network performance which usually is the prediction error about approximation of given continuous multivariate non-linear functions, such as complex real-world systems.
Consider, for simplicity, a neural network with the sole $i-th$ neuron depicted in Eq. (25) in the output layer. The neuron is driven by a signal vector, $x(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^T$, produced by several hidden layers, which are themselves driven by an input vector applied upstream in the input layer. The output signal of i-th neuron is the only output of the neural network, it is compared to the actual value, or desired target d_j, and the final goal of this process is to minimize the output error of Eq. (28), or match the user requirements.

$$e_j(t) = d_j(t) - y_j(t). \tag{28}$$

The method used to to minimize Eq. (28) define the learning method and its algorithm. Several learning methods exist : supervised, reinforcement and unsupervised.
In supervised learning, the learning process incorporates an external teacher and/or performance information through a training set (input-output) of desirable network

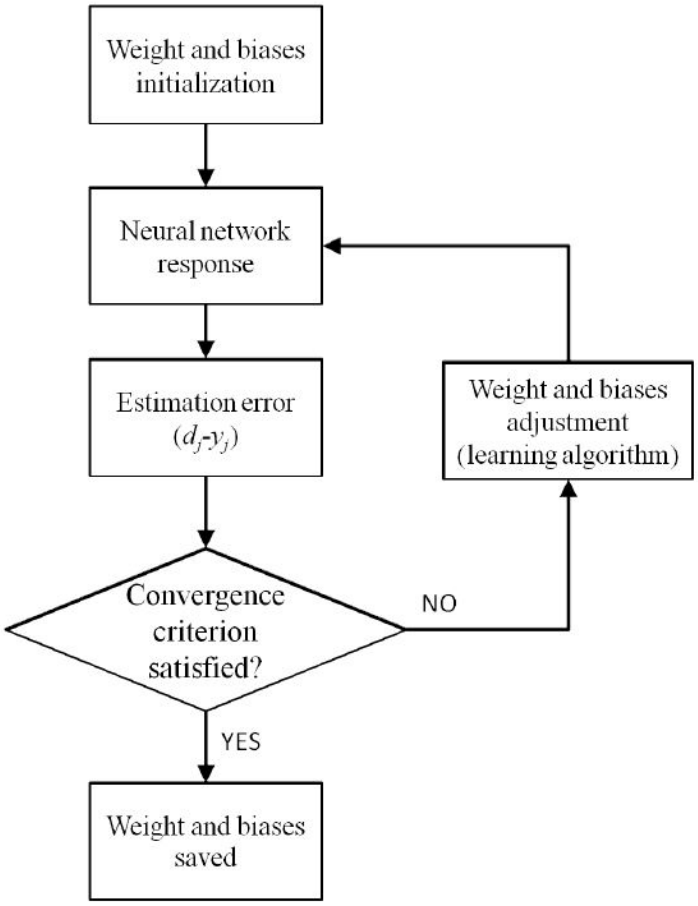
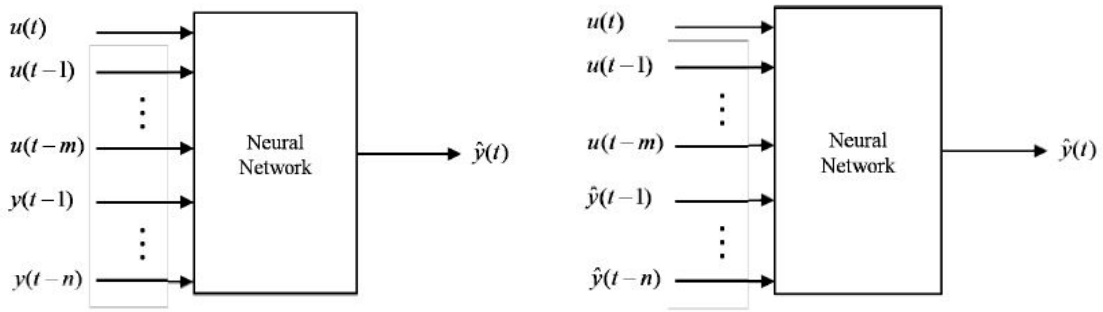


Figure 30: Flow chart of standard learning process, From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].



(a) The NNARX model structure

(b) The NNOE model structure

Figure 31: Neural network models. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

behavior. In supervised training, both the inputs and the outputs are provided. The network then processes the inputs and compares its resulting outputs against the desired outputs. Errors are then propagated back through the system, causing the system to adjust the weights which control the network.
It has been assumed that correct 'target' output values are known for each input pattern. But is some situations there is less detailed information available. In the extreme case there is only a single bit of information, saying whether the output is right or wrong. In this situation supervised learning is not feasible even if reinforcement learning can be view as a form of supervised learning because the network does, after all, get some feedback from its environment.
In unsupervised learning there is no teacher at all. A network with both inputs and outputs is still considered, but there is no feedback from the environment to say what those output should be or whether they are correct. The network must discover for itself patterns, features, correlations or categories in the input data and code for them in the output. The units and connections must thus display some degree of self-organization. Currently, in fact, this learning method is limited to networks known as self-organizing maps. These kinds of networks are not in widespread use.
Overall, the best learning method for this research project is the supervised. Within of supervised learning, several algorithms exist, gradient methods will be considered in particular. Among the descent methods, the first order and second order learning rules will be considered.
As shown in next two sections, two classical algorithms are used for training in this work: the back propagation (BP) and the Levenberg-Marquardt (LM) algorithms. The BP method is one of the oldest and most used training technique, conversely the LM is a second order training method and it is reveled much faster than the BP.

6.5.1 Error Back Propagation Algorithm

The backward propagation of error method was published first in 1969 by Bryson and Ho [13]. Few years later in the early 1970's several independent authors developed the back-propagation architecture. In 1974 Werbos [57] applied it to behavioural sciences, and only in 1986 it was applied to the training of multi-layer networks and called *back-propagation* by Rumelhart, Hinton and Williams [28].
Consider, for simplicity, a neural network with two hidden layers having the same activation function, f, and one output layer, as illustrated in Fig. 32. For each time step the error, as defined in (28), can be calculated at the output node, and the error energy for j-th output neuron at time step t can be written as

$$\mathrm{E}\,(t) = \frac{1}{2}\sum_{j\in C} e_j^2(t) = \frac{1}{2}\sum_{j\in C}\left(d_j(t) - y_j(t)\right), \tag{29}$$

where C is the output signal set, (29) simplifies to $\mathrm{E}\,(t) = e^2(t)$ in the case of single output neuron of Fig. 32. The (29) represents the cost function of the learning performance to be minimized adjusting the free parameters of the neural network. The back propagation algorithm applies a correction, Δw_{ji}, to the synaptic weight, w_{ji}, proportional to the gradient $\frac{\partial \mathcal{E}(t)}{\partial w_{ji}}$. According to the chain rule it may be written as

$$\frac{\partial \mathcal{E}(t)}{\partial w_{ji}(t)} = \frac{\partial \mathcal{E}(t)}{\partial e_j(t)}\frac{\partial e_j(t)}{\partial y_j(t)}\frac{\partial y_j(t)}{\partial v_j(t)}\frac{\partial y_j(t)}{\partial w_{ji}(t)}. \tag{30}$$

Considering the neuron of Fig. 25 to be an output neuron, hence the i-th input, x_i, represents the output of previous neuron, and, for conformity of notation, it will be indicated later in this work as y_i (see Fig. 33). Eq. (18) may be rewritten as

$$y_j = f_j(v_j) = f_j\left(\sum_{i=1}^{n} w_{ji}y_i + b_j\right) = f_j\left(\sum_{i=0}^{n} w_{ji}y_i\right), \tag{31}$$

where the synaptic weight w_{j0} (corresponding to the fixed input $y_0 = 1$) equals to bias b_j. Differentiating Eq. (18) with respect to $v_j(t)$ and $w_{ji}(t)$, the following equations can be obtained

$$\frac{\partial y_j(t)}{\partial v_j(t)} = f_j'(v_j(t)) \tag{32}$$

and

$$\frac{\partial v_j(t)}{\partial w_{ji}} = y_i(t) \tag{33}$$

Differentiating both sides of Eq.(28) with respect to $y_j(t)$ and Eq. (29) with respect to $e_j(t)$, leads respectively to

$$\frac{\partial e_j(t)}{\partial y_j(t)} = -1 \tag{34}$$

end

$$\frac{\partial \mathcal{E}(t)}{\partial e_j(t)} = e_j(t). \tag{35}$$

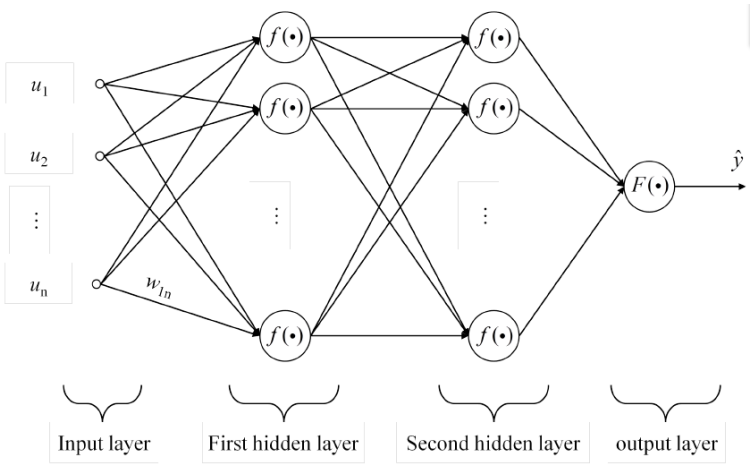


Figure 32: Multi layer perceptron neural network with two hidden layers with the same activation function and one output layer. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

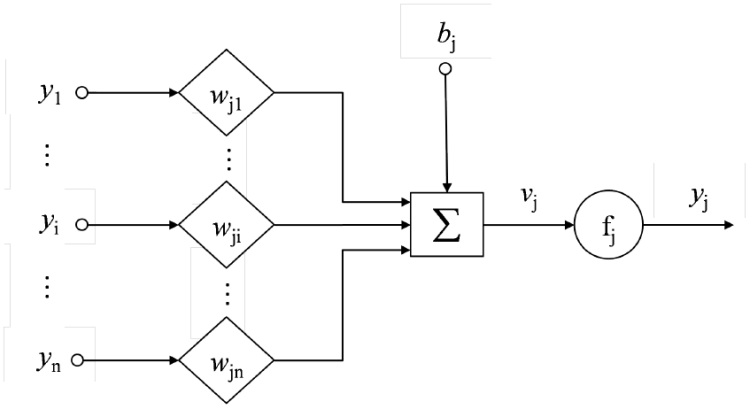


Figure 33: Signal flow scheme of the j-th output neuron. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

Therefore, the use of equations (32) to (35) in (30), yields

$$\frac{\partial \mathcal{E}(t)}{\partial w_{ji}(t)} = -e_j(t) f_j'\left(v_j(t)\right) y_i(t) = -\delta_j(t) y_i(t), \tag{36}$$

where $\delta_j(t)$ is commonly defined as the local gradient. The correction to the synaptic weight $w_{ji}(t)$ is established using the delta rule

$$\Delta w_{ji}(t) = -\eta \frac{\partial \mathcal{E}(t)}{\partial w_{ji}(t)} = \eta \delta_j(t) y_i(t), \tag{37}$$

where η is defined as the learning rate. Therefore, the updated weight is

$$w_{ji}(t+1) = w_{ji}(t) + \Delta w_{ji}(t) = w_{ji}(t) - \eta \frac{\partial \mathcal{E}(t)}{\partial w_{ji}(t)} = w_{ji}(t) + \eta \delta_j(t) y_i(t). \tag{38}$$

As shown by several authors [29, 25], there is not an optimum learning rate but according to the particular problem there is a η that assures fast and stable convergence. There are some algorithm provided with a variable learning algorithm rate according to the local gradient, or other parameters, to speed up the convergence [16]. If the j-th neuron is part of an hidden layer, the error flow backward through another layer. In order to highlight this process, consider the Fig. 34. In analogy with the j-th output neuron treatment, the chain rule it is used to find an useful expression of the gradient $\frac{\partial \mathcal{E}(t)}{\partial w_{ji}(t)}$. The local gradient can be redefined as

$$\delta_j(t) = -\frac{\partial \mathcal{E}(t)}{\partial y_j(t)} \frac{\partial y_j(t)}{\partial v_j(t)} = -\frac{\partial \mathcal{E}(t)}{\partial y_j(t)} f_j'\left(v_j(t)\right), \tag{39}$$

Since k-th neuron is an output neuron, the index j is substituted with k in (29), so differentiating with respect to $y_j(t)$, it yelds

$$\frac{\partial \mathcal{E}(t)}{\partial y_j(t)} = \sum_k e_k(t) \frac{\partial e_k(t)}{\partial y_j(t)} = \sum_k e_k(t) \frac{\partial e_k(t)}{\partial v_k(t)} \frac{\partial v_k(t)}{\partial y_j(t)}. \tag{40}$$

Considering Fig. 33, it is easy to obtain

$$e_k(t) = d_k(t) - y_k(t) = d_k(t) - f_k(v_k(t)). \tag{41}$$

Hence, differentiating with respect to $v_k(t)$, it yields

$$\frac{\partial e_k(t)}{\partial y_j(t)} = f_k'\left(v_k(t)\right). \tag{42}$$

Equation (31) can be rewritten for the k-th neuron and differentiated with respect to $y_j(t)$, in order to obtain the following expression

$$\frac{\partial v_k(t)}{\partial y_j(t)} = w_{kj}(t). \tag{43}$$

By using (43), (42) and (40), it yields

$$\frac{\partial \mathcal{E}(t)}{\partial y_j(t)} = -\sum_k e_k(t) f_k'(v_k(t)) w_{kj}(t) = -\sum_k d_k(t) w_{kj}(t) \tag{44}$$

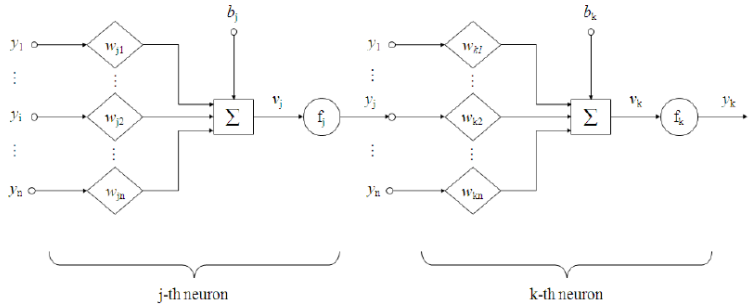


Figure 34: Signal flow scheme of the j-th hidden neuron. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

Finally, by using the (44) in (39), the back propagation formula for hidden neuron is obtained as follows

$$\delta_j(t) = f_j'(v_j(t)) \sum_k d_k(t) w_{kj}(t). \tag{45}$$

Therefore, the adjusting weight for an hidden neuron may be expressed as (38) substituting the local gradient $\delta_j(t)$ with the new expression (45). This is one of the oldest and most used weight adjustment algorithm, it recalls in such a way the gradient method used to find equation minimum and maximum points, indeed it is also known as steepest descent algorithm. The back-propagation algorithm (BP) can be only used for *on-line* training, or in other words, for real time training. The BP algorithm can be used in batch mode, as described in next section, in order to be used as *off-line* training type.

6.5.2 Batch error back-propagation algorithm

As it can be noted from (45) and (37) the correction is applied for each time step, or iteration, of the training data set that is characteristic of *on-line* training strategy. To use the error back-propagation theory, as described in previous section, in *off-line* mode, that is equivalent to use a single training run with the whole available input-output pattern and, therefore, to update the network synaptic weights after that the entire training set has been fed to the network. The gradients calculated at each training example are added together to determine the adjustment in the weight and biases. In order to consider the whole set of corresponding input output, the batch BP (BBP) error is then defined as

$$\mathcal{E}_{av} = \frac{1}{N} \sum_{t=1}^{N} \mathcal{E}(t) = \frac{1}{2N} \sum_{t=1}^{N} \sum_{j \in C} e_j^2(t) = \frac{1}{2N} \sum_{t=1}^{N} \sum_{j \in C} \left(d_j(t) - y_j(t)\right), \tag{46}$$

and a very similar procedure presented in previous section can be followed to obtain the updating weight. In order to fasten the BP algorithm, a second order training algorithm will be presented in the next section.

6.5.3 Descent Methods

Since the Levenberg–Marquardt (LM) algorithm belongs to Newton's method category, which are gradient-based methods.
As for back propagation algorithm, the focus is on minimizing the error function $\mathcal{E}(t)$, but using a second order approach. Moreover, in order to simplify the treatment, the *on-line* training method will be presented since very few modifications are required to obtain the batch, or *off-line*, mode. First a digression on gradient and Newton's methods will be presented, and then the LM algorithm will be discussed.

6.5.4 Gradient-based Methods

As shown for BP algorithm, the main objective is to minimize the performance function, $\mathcal{E}$ of (28), defined on an n-dimensional input space $\underline{w} \in \mathbb{R}^R$, where the synaptic weights and biases are arranged in one dimensional vector, $\underline{w} = [w_1, \ldots, w_R]^T$, where R is the number of all free parameters of the neural network. The gradient expression of left hand side of 30 it can be expressed as function of all weights

$$\underline{g}(t) = \frac{\partial \mathcal{E}(t)}{\partial \underline{w}(t)} = \left[\frac{\partial \mathcal{E}(t)}{\partial w_1(t)}, \ldots, \frac{\partial \mathcal{E}(t)}{\partial w_R(t)} \right]^T. \tag{47}$$

According to the gradient descent methods, the update weights, $\underline{w}$, are calculated as

$$\underline{w}(t+1) = \underline{w}(t) - \eta \underline{\underline{G}}\underline{g}(t) = \underline{w}(t) - \eta \frac{\partial \mathcal{E}(t)}{\partial \underline{w}(t)}, \tag{48}$$

where $\underline{\underline{G}}$ is a certain positive definite matrix.

6.5.5 Steepest Descent Method

The method of steepest descent is one of the oldest technique for minimizing multivariate functions. For this method the matrix $\underline{\underline{G}}$ is defined as the identity matrix, $\underline{\underline{I}}$, hence the (ErrorgradientLM) decomes the well known

$$\underline{w}(t+1) = \underline{w}(t) - \eta \underline{g}(t). \tag{49}$$

The negative search direction, $-\underline{g}$, means this method follows the *local steepest descent downhill* direction, that implies the algorithm is highly effected by initial conditions and easily *falls* in local minima, without minimizing globally the given function.

Comparing (48) and (38), it can be noted that the equations are quite familiar, indeed the gradient method, only differs from the back-propagation method for the formal analytical expressions, but they are essentially driven by the same principles: the updating weights depends on the gradient of function $\mathcal{E}$. Indeed, the BP algorithm is often identified with the steepest descent method.

6.5.6 Newton's Method

For the Newton method the descent direction is determined using the second order derivatives of the objective function, if available. Apart the mathematical concerns about the existence of derivatives, the performance function $\mathcal{E}$ can be unfolded by Taylor series and taken at second-order approximation

$$\begin{aligned}\mathcal{E}(t+1) \approx \mathcal{E}(t) & \\ &+ \underline{g}^T \left[\underline{w}(t+1) - \underline{w}(t+1) \right] \\ &+ \frac{1}{2} \left[\underline{w}(t+1) - \underline{w}(t+1) \right]^T \underline{\underline{H}} \left[\underline{w}(t+1) - \underline{w}(t+1) \right],\end{aligned} \tag{50}$$

considering that higher order terms are omitted to the assumption that $[\underline{w}(t+1) - \underline{w}(t+1)]$ is sufficiently small and where $\underline{\underline{H}}(t)$ is the Hessian matrix

$$\underline{\underline{H}}(t) = \begin{bmatrix} \dfrac{\partial^2 \mathcal{E}(t)}{\partial^2 w_1(t)} & \cdots & \dfrac{\partial^2 \mathcal{E}(t)}{\partial w_1(t) \partial w_R(t)} \\ \cdots & \cdots & \cdots \\ \dfrac{\partial^2 \mathcal{E}(t)}{\partial w_R(t) \partial w_1(t)} & \cdots & \dfrac{\partial^2 \mathcal{E}(t)}{\partial^2 w_R(t)} \end{bmatrix}. \tag{51}$$

Since Eq. 50 is a quadratic function of $\underline{w}(t)$, its minimum point (w_{MIN}) can be easily obtained differentiating (50) and making it equal to zero, and solving the following set of linear equations

$$0 = \underline{g}(t) + \underline{\underline{H}}(t) \left[\underline{w}_{MIN} - \underline{w}(t+1) \right]. \tag{52}$$

If the inverse matrix of $\underline{\underline{H}}$ exists, the Newton's method is obtained

$$\underline{w}_{MIN} = \underline{w}(t) - \underline{\underline{H}}(t)^{-1} \underline{g}(t). \tag{53}$$

It is clear that if the error function $\mathcal{E}$ is not quadratic, $[\underline{w}_{MIN} - \underline{w}(t+1)]$ represents only a first step towards the minimum point. Therefore, several steps are needed to archive the minimum point using an iterative scheme updating new weights as

$$\underline{w}(t+1) = \underline{w}(t) - \underline{\underline{H}}(t)^{-1} \underline{g}(t), \tag{54}$$

which is the general expression (48), where $\underline{\underline{G}} = -\underline{\underline{H}}^{-1}$ and $\eta = 1$.

6.5.7 Levenberg-Marquardt Algorithm

If the inverse Hessian matrix exists but is not positive definite, the Newton's method, described before, can lead to local maximum, or a saddle point [30]. To overcome this issue, Levenberg [41] and Marquardt [44] proposed to alter the Hessian matrix with a positive definite matrix $\underline{\underline{P}}$ to make $\underline{\underline{H}}$ positive definite in least square problems. Later, Goldfeld et al. [22] first applied this methodology to the Newton's method, considering

$\underline{\underline{P}} = \lambda \underline{\underline{I}}$. Therefore the update synaptic weight (54) can be re-written according to Levenberg-Marquardt as

$$\underline{w}(t+1) = \underline{w}(t) - \left(\underline{\underline{H}}(t) + \lambda \underline{\underline{I}}\right)^{-1} \underline{g}(t), \tag{55}$$

where $\underline{\underline{I}}$ is the identity matrix and λ is a certain non negative value. The Levenberg-Marquardt method transits smoothly between Newton's method as λ approaches 0 and the steepest descent method as λ grows infinitely.
In this work λ is chosen according to variation of performance index for each training iteration (t). At a large distance from the function minimum, the steepest descent method is utilized to provide steady and convergent progress toward the solution. As the solution approaches the minimum, λ is adaptively decreased, the Levenberg-Marquardt method approaches the Newton's method, and the solution typically converges rapidly to the local minima.

6.5.8 Validation

Once the neural network has been designed, trained, it needs to be validated with test data never used before in order to establish if the neural network is able to produce any input-output mapping of the system at hand, better if they differ from the examples used to train the network. The training loop of Fig. 30 can be enhanced with the validation stage as described in Fig. 35. According to [25], a neural network is said to generalize well when the input-output patterns computed by the network is correct for the test data never used. When, however a neural network learns too many, or repeated, input-output examples, the network may memorize the training data, just as a human being. Such a phenomena is referred to as overfitting or overtraining, and in this case the network looses the ability to generalize. Since, this deficiency is stored inside the synaptic weight, several technique have been developed to adjust the weights in order to make the neural network more general, such as network pruning techniques. The local minima is another issue that can be encountered when training a neural network. Consider, for example, a single layer neural network with one hidden neuron and only one free synaptic parameter. Therefore, as said before, the training can be seen as a non-linear curve-fitting problem, where the neural network error prediction, as defined in Eq. (29), depends on the value free parameter, w_{ij}, see Fig. 36. This example can be translated in multi-dimensional space and the error curve becomes a very complex hyper surface which depends on the free parameters, weights and biases. Anyway, starting from the first weight attempt, w_{init}, any deterministic training algorithms, such as BP and LM algorithms, will always find the w_{min_1} as the best weight to minimize the error, or optimize the neural network performance. Unfortunately, the performance curve, or more in general the hyper surface, look like the curve of Fig. 36, and the network training often will end up in local minima points. The local minima and overfitting issues will be treated in the next two sections.

6.5.9 Generalization - Network growing and pruning

To model real-world system using neural networks usually requires the use of complex network structures and big amount of synaptic connections, translated in computer

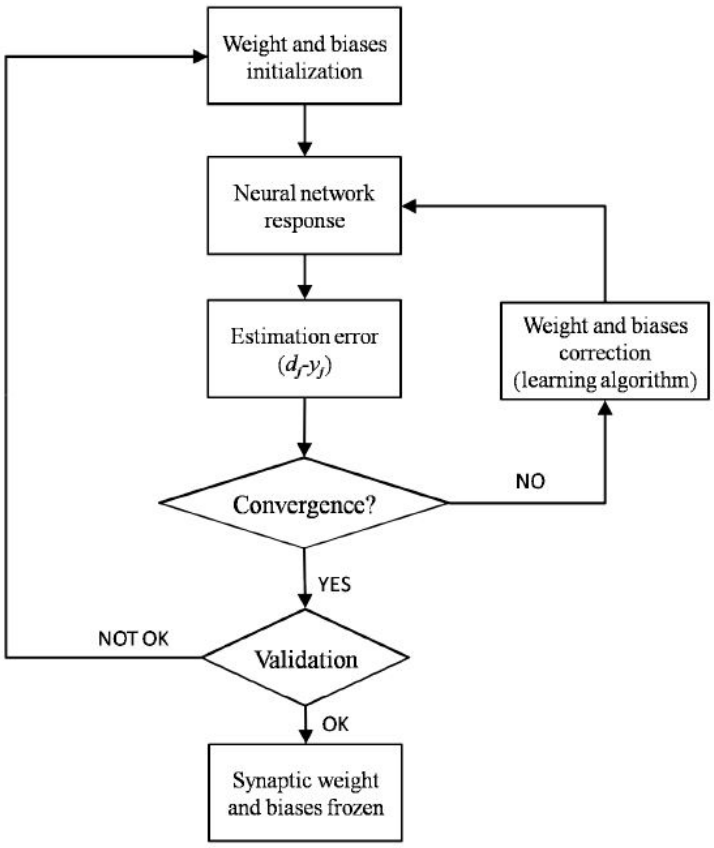


Figure 35: Flow chart of neural network training and validation process. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

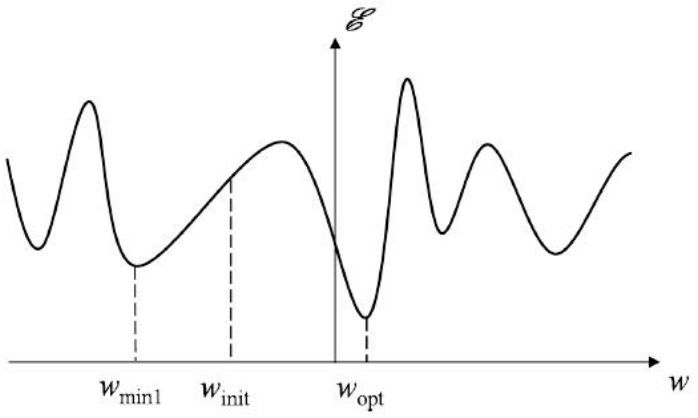


Figure 36: Non-linear curve-fitting problem with several local minima. The initial synaptic weight, w_{init}, a local minimum point, w_{min_1}, and the absolute minima, w_{opt}. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

language, high computational resources, cost and long execution time. A neural network with minimum size is less likely to learn noise in the training data and so may generalize better to new data. This can be achieved starting with the network pruning or growing techniques. In the network growing method, a small network is designed and new neuron, or a new layer, are added when the network is unable to meet the design requirements. The network pruning works in the opposite direction of the network growing. Starting from a large network with satisfactory performance for the problem at hand, some weights are weakened or deleted in a selective and orderly fashion [25, 42]. It is obvious, that anytime the neural network is modified, as when weights are deleted or neurons are added, to evaluate the new performance the network must be re-trained, taking with itself all the issues presented in the previous section. Even if the generalization, or regularization, techniques are very useful to make the neural network as more general as possible, they are really time and hardware resources consuming.

7 Neural Network for Air Data Estimation

As aerodynamic forces and moments acting on aircraft are function of interactions between the body and the external flow, the determination and monitoring of Air Data is of capital importance to prevent malfunctioning and then fatal accidents. The ADS is hence considered a safety-critical aircraft system, which currently is made of several exernal sensors.
This work deals with the estimation of the aerodynamic angles α and β (as defined in Sect. 2.6) through a Neural Network-based virtual sensor which would be highly cost and encumbrace effective with respect to the real ones.

Reminding what it has been found in Sect. 2.6, aerodynamic angles and body velocity can be expressed as:

$$\begin{cases} V = \sqrt{u^2 + v^2 + w^2} \\ \alpha = tan^{-1}\left(\frac{w}{u}\right) \\ \beta = tan^{-1}\left(\frac{v}{\sqrt{u^2+v^2}}\right) \end{cases}$$

Diffentiating these equations, the resulting system, constituted by three ordinary differential equations, needs to be solved in order to find the analytical expression for V, α and β.
For simplicity, the attention is now focused on the angle of attack: differentiating the second equation of the system above, results in:

$$\dot{\alpha} = \frac{u\dot{w} - w\dot{u}}{u^2 + w^2} \tag{56}$$

From Sect. 2.6, the body components' velocities are related to aerodynamic angles as follow

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = V \begin{bmatrix} cos\alpha\, cos\beta \\ sin\beta \\ sin\alpha\, cos\beta \end{bmatrix}$$

Replacing these last expressions into 56 yields to:

$$\dot{\alpha} = \frac{\dot{w}\, cos\alpha - \dot{u}\, sin\alpha}{V\, cos\beta} \tag{57}$$

Hence, knowing that the relationship between axial acceleration, gravity acceleration, angular velocities and the external forces ($F_i = ma_i$) in the body reference frame is:

$$\begin{cases} \dot{u} = rv - qw - g_0\, sin\theta + F_x \\ \dot{v} = -ru + pw + g_0\, sin\phi cos\theta + F_y \\ \dot{w} = qu - pv + g_0\, cos\phi\, cos\theta + F_z \end{cases} \tag{58}$$

If F_i is indicated in terms of g accelerations ($n_i = \frac{a_i}{g}$) and is introduced an expression for F_z as a function of pilot commands, or, more in general, as a function of the control surface deflections, it is possible to obtain the relationship between the normal acceleration n_z and the command actions:

$$n_z = \frac{F_z}{m} = \frac{1}{2}\rho V^2 \left(C_{z0} + C_{z,\alpha}\alpha + C_{z,\alpha^3}\alpha^3 + C_{z,q}\frac{qc}{V} + C_{z,\delta_e}\delta_e + C_{z,\delta_e\beta^2}\delta_e\beta^2\right) \tag{59}$$

Hence, substituting Eqs.58 in Eq.57 yields to:

$$\dot{\alpha} = \frac{(qu - pv + g_0\, cos\phi\, cos\theta + n_z)\, cos\alpha}{V\, cos\beta} - \frac{(rv - qw - g_0\, sin\theta + n_x)\, sin\alpha}{V\, cos\beta} \tag{60}$$

Therefore, replacing $[u, v, w]$ with their definition, finally leads to:

$$\dot{\alpha} = \frac{1}{V\, cos\,\beta}\{[V\,(q\, cos\alpha\, cos\beta - p\, sin\beta) + n_z + g_0\, cos\,\phi\, cos\theta]\, cos\alpha - [V\,(r\, sin\beta - q\, sin\alpha\, cos\beta) - g_0\, sin\theta + n_x]\, sin\alpha\} \tag{61}$$

According to Eq.61 stands out the following functional relationship

$$\alpha = f_\alpha \left(V, n_x, n_z, \beta, \theta, \phi, p, q, r \right) \tag{62}$$

Following a similar procedure, the β functional relationship is:

$$\beta = f_\beta \left(V, n_x, n_y, n_z, \alpha, \theta, \phi, p, q, r \right) \tag{63}$$

Substituting β in Eq.62 with expression written in Eq.63, collecting all the indipendent variables and explicitating the dependency on pilot demands, the angle of attack functional relationship can be further expanded as:

$$\alpha = F_\alpha \left(V, n_x, n_y, n_z, \theta, \phi, p, q, r, \delta_e, \delta_r, \delta_a \right) \tag{64}$$

Therefore, from Eq.64, comes out that the AoA depends on velocity, body axis acceleration, Euler angles and pilot commands. The real advantage of using a Neural Network for AoA/AoS estimation is actually that is not necessary an exact expression for F_α / F_β but it is sufficient the functional relationship written in Eq.64.
For all the NN's trainings characterized by the absence of external wind, a very straightforward model, suitable for real-time avionic systems, is used.
From this model, α and β can be expressed as:

$$\alpha = \hat{\alpha} + \Delta\alpha \tag{65}$$

$$\beta = \hat{\beta} + \Delta\beta \tag{66}$$

where $\hat{\alpha}$ and $\hat{\beta}$ are linear estimation obtained from flight mechanics equations while $\Delta\alpha$ and $\Delta\beta$ are the difference between the linear estimations and the true non linear angles. On the one hand, the first two terms are simply evaluated as:

$$\hat{\alpha} = \theta - \gamma \tag{67}$$

$$\hat{\beta} = K \frac{n_y}{q_c} \tag{68}$$

where γ stands for the flight path angle and K is a constant which allows to adjust the measure units that typically has a value between 20 $\frac{kg}{m^2}$ and 40 $\frac{kg}{m^2}$.
On the other hand, the other two contributes are calculated according to a patented procedure [33], based on one MLP, which process measurements obtained by the GPS, the ADS, the AHRS (Attitude and Heading Reference System) and a Pitot-tube as the only external source of data.

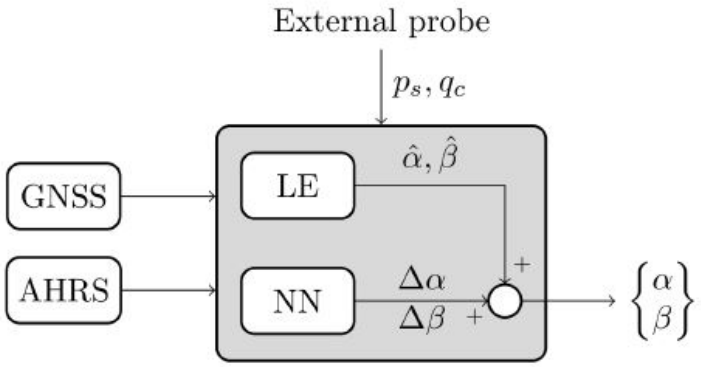


Figure 37: General schematic of Neural Network application for AoA/AoS estimation. From Angelo Lerro et al.,"Aerodynamic angle estimation: comparison between numerical results and operative environment data",[39]. From Angelo Lerro,"Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles",[38].

The analysis of the influence of the input vector on results allows the reduction of the total number of signals necessary to the NN and eventually the complexity of the itself architecture.

8 Flight data used for NN's training and test

It is commonly known that the training stage plays a crucial role in NN applications: although the training techniques are all widely experimented, documented and substantiated (see Sect. 6.5), in fact, it is not trivial to collect a training data set that is adequately representative of the model dynamics which the NN has to mimic. Training data might come from flight tests, wind tunnel data or from a mathematical model. On the other hand, despite the use of a mathematical model is very practical, only a preliminary design of the virtual sensor is possible as discrepancies between model and reality reintroduce all the disadvantages of model-based techniques. Wind tunnel data are not exploitable as well because of costs inherent to perform dynamic maneuvers. The best way to design a synthetic sensor is then through the use of real flight data. On the other hand, the main drawback is that real flight data are usually affected by several sources of disturbance, such as air turbolence, structure vibrations and electronic noise. For this reason, in order to preliminarly asses the feasibility of the integration of a NN-based synthetic sensor with real aicrafts, it has been made the choice of using, for the training stage, only flight data coming from a prototype's simulator: in this manner, all disturbances due to structure vibrations and electromagnetic compatibility can be neglected. Moreover, only maneuvers at Mach lower than 0.6 have been taken into account so that transonic effects can be neglected. The NN is feasibility with real flight

data and higher Mach number will be object of deeper future studies.

9 Realization of sync structure as input Data

The first step is to acquire, manage and convert flight data into an appropriate form to be processed by Matlab Neural Network Toolbox.
In fact, this latter requires as input a Matlab structure, called *sync*, whose fields are defined as follows:

- sync.time: Time discretization [s];
- sync.deltae: Elevator deflection [°];
- sync.deltaa: Aileron deflection [°];
- sync.deltar: Rudder deflection [°];
- sync.deltaf: Flap deflection [°];
- sync.deltaf_LE: Leading Edge flap position [°];
- sync.nx: Body longitudinal acceleration $[m/s^2]$;
- sync.ny: Body lateral acceleration $[m/s^2]$;
- sync.nz: Body normal acceleration $[m/s^2]$;
- sync.p: Roll rate $[°/s]$;
- sync.q: Pitch rate $[°/s]$;
- sync.r: Yaw rate $[°/s]$;
- sync.roll: Roll angle [°];
- sync.pitch: Pitch angle [°];
- sync.yaw: Yaw angle [°];
- sync.Vnorth: North velocity in NED reference system $[m/s]$;
- sync.Veast: East velocity in NED reference system $[m/s]$;
- sync.Vdown: Down velocity in NED reference system $[m/s]$;
- sync.qc: Dynamic pressure [Pa];
- sync.qcdot: Dynamic pressure derivative [Pa/s], computed numerically through a first order backward approximation;
- sync.gamma: Ramp angle [°];
- sync.alpha: 6 DOF angle of attack $/circ$;
- sync.beta: 6 DOF side slip angle $/circ$;

- sync.alt: Pressure altitude [m];
- sync.airsp: Calibrated Air Speed [m/s];
- sync.TAS: True Air Speed [m/s];
- sync.TASdot: True Air Speed derivative $[m/s^2]$;
- sync.IAS: Indicated Air Speed [m/s];
- sync.u: Body velocity along Xb-axis;
- sync.v: Body velocity along Yb-axis;
- sync.w: Body velocity along Zb-axis;
- sync.udot: Body acceleration along Xb-axis;
- sync.vdot: Body acceleration along Yb-axis;
- sync.wdot: Body acceleration along Zb-axis;
- sync.torqueRH:Right engine throttle position [°]
- sync.torqueLH:Left engine throttle position [°]

Flight data inputs are calibrated (at free stream condition) and their output format is at least with 4 decimal digits with the exception of latitude and longitude where at least 8 decimal digits are required. For each flight record are declared the weight and the center of gravity position.
Data from the Flight Test Acquisition System have been resampled through linear interpolation using the Matlab function *interp1* in order to be synchronized and referred to an unique reference time. This operation is necessary since data from Flight Test Acquisition System have different update frequencies and consecutive data on the Digital Serial Bus have different time stamps.
In particular, it has been chosen as sampling frequence the maximum among all temporal basis.
Moreover, in order to avoid the formation of NaN (Not a Number) during interpolation, sync.time has been initialized with the highest value between all first elements of the time vectors of each parameter and ended with the lowest last element.
Once the *sync* structures has been created, it can finally be used to train and test the NN.

10 Strategy for creating Training and Test maneuvers

10.1 Sample maneuvers

For the simulated environment, a set of sample manoeuvres must be properly defined aimed to have a suitable distribution of training points (flight test data) over speed. Each manoeuvre is performed with the objective to excite the all A/C dynamic modes, therefore the single manoeuvre is repeated several times in order to populate the training pattern with adequate data. Once all data are collected, their variables contained in the input vector are normalized between ± 1 considering minimum and maximum operative values defined by A/C manufacturer. In this work, these values are not displayed because of intellectual property reason.
The list of manoeuvers (the LAND, T/O and CLEAN configurations are considered) is reported hereafter:

1. **Steady flight condition**
   - TC1 Steady State 300 kts
   - TC2 Steady State 180 kts
   - TC3 Steady State 150 kts

   Steady flight, unaccelerated flight, or equilibrium flight is a special case in flight dynamics where the aircraft's linear and angular velocity are constant in a body-fixed reference frame. Basic aircraft maneuvers such as level flight, climbs and descents, and coordinated turns can be modeled as steady flight maneuvers. Typical aircraft flight consists of a series of steady flight maneuvers connected by brief, accelerated transitions. Steady flight states are considered as the equilibrium conditions around which flight dynamics equations are expanded.

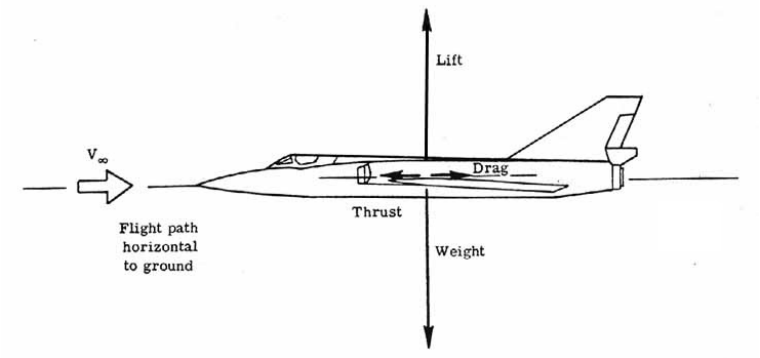


Figure 38: Steady state condition. From "SP-367 Introduction to the Aerodynamics of Flight",[3].

2. **Sawtooth glide**

   - TC4 Sawtooth glide 300 kts
   - TC5 Sawtooth glide 180 kts
   - TC6 Sawtooth glide 150 kts

   The aircraft glides completing a series of short climbs and descents where the altitude band for each maneuver is usually lesser of 1000 ft.

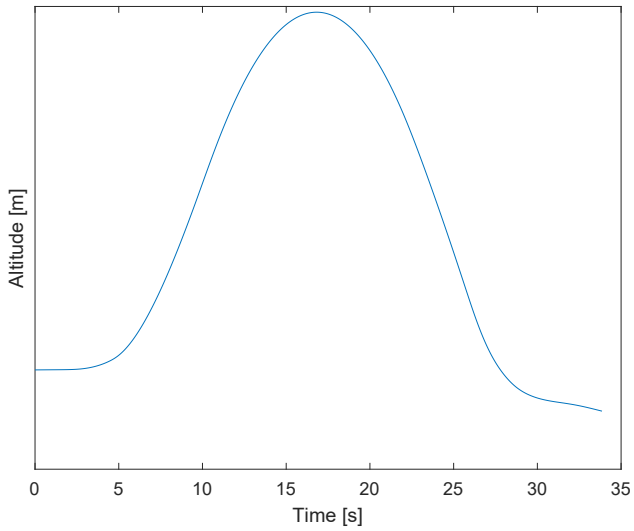


Figure 39: Sawtooth glide typical path

3. **Stall - Slow down**

   - TC7 Stall, flap UP
   - TC8 Stall, falp LAND
   - TC9 Stall, flap T/O

   Starting from 1.5 V_{stall} the airplane slows down to V_{min} where the stall occurs.
   A stall is a condition in aerodynamics and aviation such that if the angle of attack increases beyond a certain point then lift begins to decrease. The angle at which this occurs is called the critical angle of attack, is dependent upon the airfoil section or profile of the wing, its planform, its aspect ratio and other factors and eventually is the angle of attack on the lift coefficient versus angle-of-attack (C_L/α) curve at which the maximum lift coefficient occurs. Stalling is caused by flow separation which, in turn, is caused by the air flowing against a rising pressure.

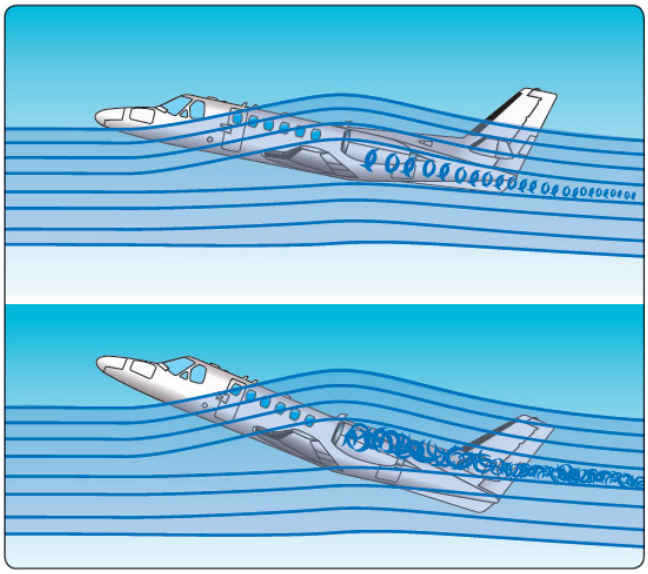


Figure 40: Stall maneuver. From "Stalls in Jet Powered Airplanes",[52].

4. **Pitch hold**
    - TC10 Pitch hold, 300 kts
    - TC11 Pitch hold, 180 kts
    - TC12 Pitch hold, 150 kts

    The airplane mantains a certain pitch angle as shown in Fig. 41.

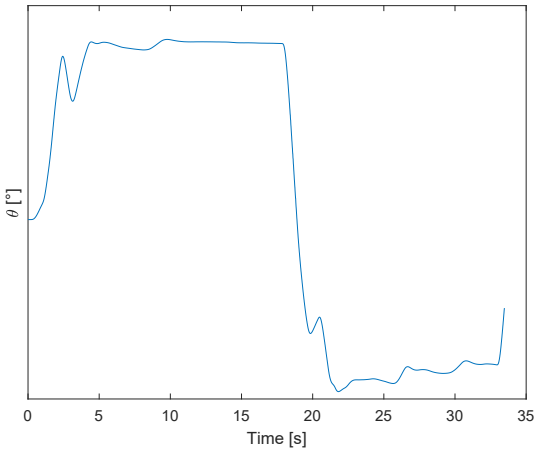


Figure 41: Pitch angle's trend during a pitch hold maneuver

5. **Pitch sweep**
    - TC13 Pitch sweep, 300 kts

- TC14 Pitch sweep, 180 kts
- TC15 Pitch sweep, 150 kts

The airplane changes continuosly its pitch angle as in Fig. 42.

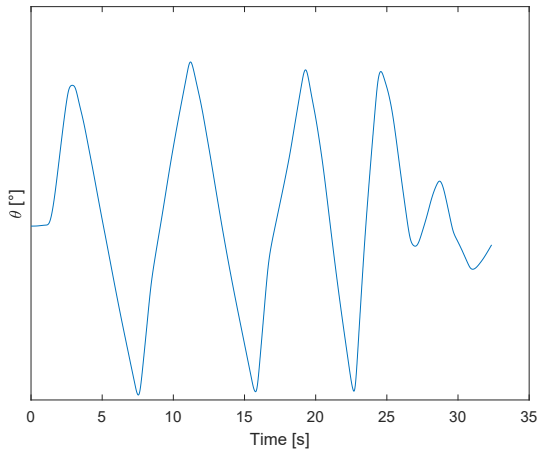
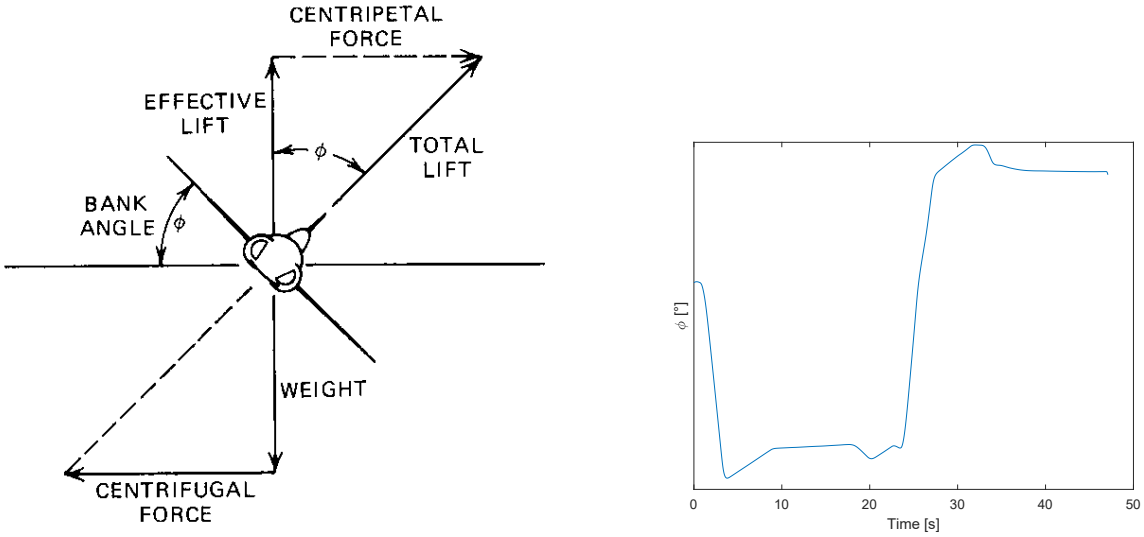


Figure 42: Pitch angle's trend during a pitch sweep maneuver

6. **Bank hold**
   - TC16 Bank hold 20°, flap UP
   - TC17 Bank hold 40°, flap UP
   - TC18 Bank hold 20°, flap LAND
   - TC19 Bank hold 40°, flap LAND
   - TC20 Bank hold 20°, flap T/O
   - TC21 Bank hold 40°, flap T/O

   The airplane mantains a certain bank angle as in Fig43.



(a) (b)

Figure 43: Bank hold maneuver. From "Turn Performance",[56]

7. **Bank sweep**

   - TC22 Bank sweep 20°, flap UP
   - TC23 Bank sweep 40°, flap UP
   - TC24 Bank sweep 20°, flap LAND
   - TC25 Bank sweep 40°, flap LAND
   - TC26 Bank sweep 20°, flap T/O
   - TC27 Bank sweep 40°, flap T/O

   The airplane changes continuosly its bank angle for the entire duration of the maneuver as in Fig. 44

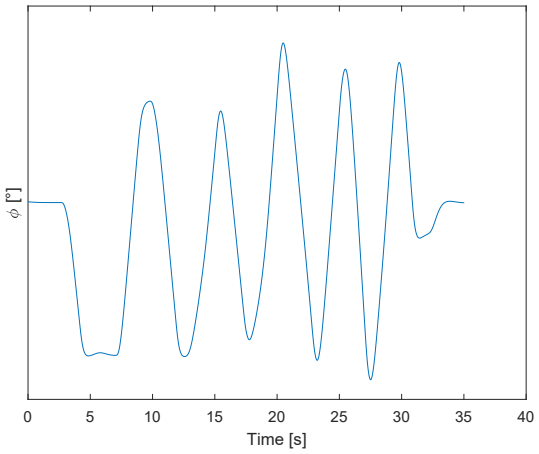


Figure 44: Bank angle's trend during a bank sweep maneuver

8. **Steady Heading Sideslip (SHSS)**

   - TC28 SHSS 5°, flap UP
   - TC29 SHSS 10°, flap UP
   - TC30 SHSS 15°, flap UP
   - TC31 SHSS 5°, flap LAND
   - TC32 SHSS 10°, flap LAND
   - TC33 SHSS 15°, flap LAND
   - TC34 SHSS 5°, flap T/O
   - TC35 SHSS 10°, flap T/O
   - TC36 SHSS 15°, flap T/O

   The maneuver is carried out in six steps:

   (a) The maneuver starts with the aircraft trimmed in straight and level flight (Fig. 45(a));

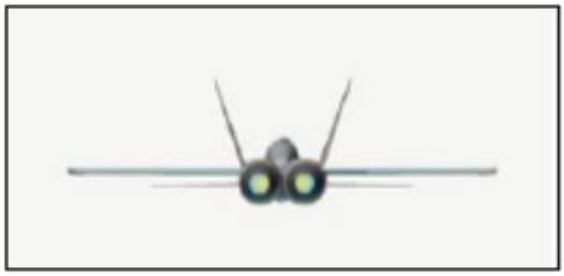
   (b) The pilot applies a constant force on the rudder pedal (Fig. 45(b));

   (c) The aircraft yaws in response to the rudder input (Fig. 45(c));

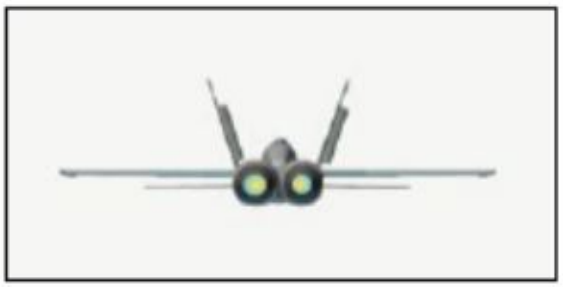
   (d) The pilot banked the aircraft in the opposite direction of the yaw in order to stabilize it at a constant heading (Fig. 45(d));

   (e) The sideslip and bank angle were incrementally increased until reaching the desired angle of sideslip;

   (f) Finally the pilot repeated the maneuver in the opposite direction returning to trim conditions.



(a) SHSS - Part I



(b) SHSS - Part II

(c) SHSS - Part III

(d) SHSS - Part IV

Figure 45: Steady Heading SideSlip maneuver. From Marie-Michèle Siu et al.,"Flight Test Results of an Angle of Attack and Angle of Sideslip Calibration Method Using Output-Error Optimization",[51].

9. **Dutch roll**

   - TC37 Dutch roll, flap UP
   - TC38 Dutch roll, flap UP
   - TC39 Dutch roll, flap UP

   This maneuver consists in stimolating a Dutch roll motion characterized by an out-of-phase combination of "tail-wagging" (yaw) and rocking from side to side (roll). This yaw-roll coupling is one of the basic flight dynamic modes (others include phugoid, short period, and spiral divergence), is normally well damped in most light aircrafts, though some aircrafts with well-damped Dutch roll modes can experience a degradation in damping as airspeed decreases and altitude increases. Dutch roll stability can be artificially increased by the installation of a yaw damper.

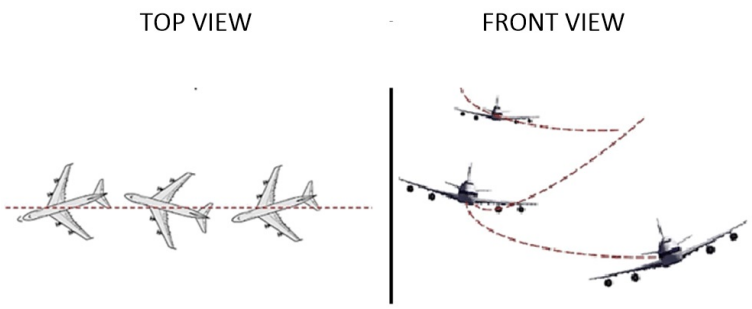


Figure 46: Dutch roll maneuver. From "Dutch Roll",[20]

10.2 Inputs' hypercube analysis

A graphical interpretation of the ability of a training set to cover the entire flight envelope of the aircraft can be conducted using the box and whiskers plots [8]. The normalized input signals used for a typical training in the simulated scenario can be seen in Fig. 47. Middle lines represent the medians and the boxes delimit the regions between the 25th and the 75th quartiles. Whiskers (dashed vertical lines) extend for 1.5 times the difference between the 75th and the 25th quartiles before and after the 25th and the 75th quartiles themselves. The remaining data are considered outliers and are individually highlighted using red plus signs. The hypercube analysis is a straightforward instrument to understand the correctness of the flown manoeuvres. When the training set of manoeuvres is defined, the optimal is to have the boxes that cover the largest region possible in the interval $[-1, 1]^n$. On the other hand, in those regions where there are outliers, the training performance will be degraded and it is recommended to add new manoeuvres to thicken the coverage on training.
It is important to remark that, in order to have good performance, the test hypercube must necessarily be a subset of the training hypercube. If this condition is not verified, AoA and AoS should be estimated in a dynamical situation that the Neural Network can not mimic.

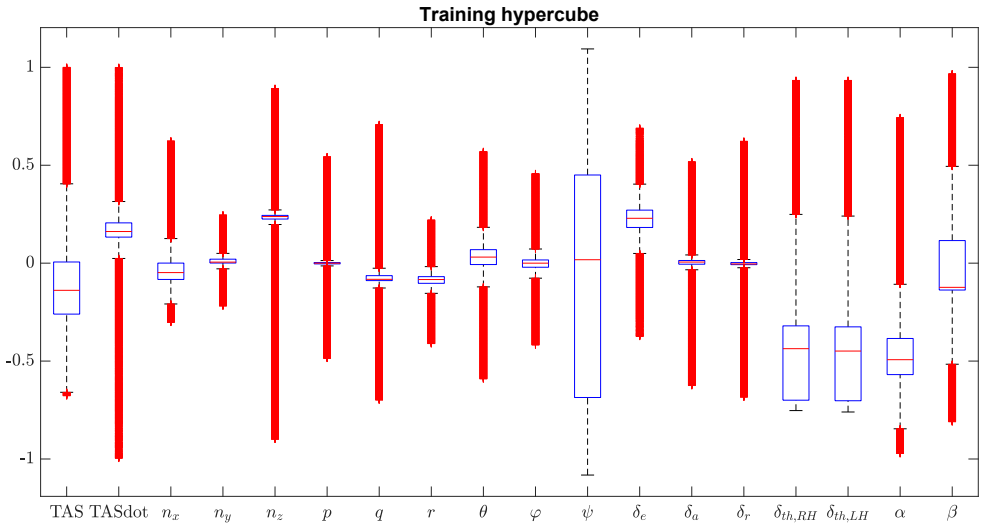


Figure 47: Graphical visualization of the normalized input range for $NN_{pat,nw}$ training (Sect.11.2)

11 Definition of the Neural Network architecture using the patented procedure

In order to define the best Neural Network architecture, several attempts have been carried out manipulating the set of training maneuvers, the input vector, the output vector, the number of neurons and layers. Once the best Neural Network architecture is found, its limits are explored by varying the testing set with the purpose of including all possibile situations that the synthetic sensor could encounter during a real flight. In fact, only if the NN eventually works efficiently in all possible different scenarios can be feasibile for real flight application.
In Tab. 1 are outlined all the attempts made throughout the current Section.

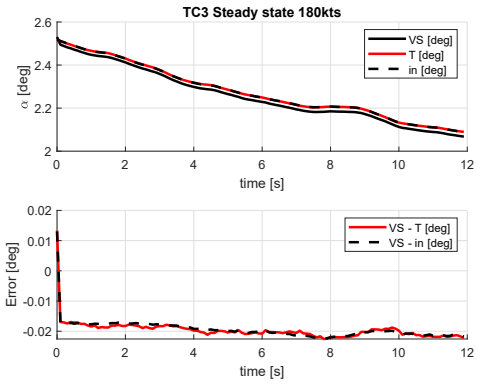
Attempt	Outputs	Long.	Latero.	Wind	Gust	CG	Train/Test
1		yes	no	no	no	intermediate	**Training**
1	$\Delta\alpha$	yes	yes	no	no	intermediate	**Test**
2		yes	yes	no	no	intermediate	**Training**
2	$\Delta\alpha$	yes	yes	no	no	intermediate	**Test**
3		yes	yes	no	no	intermediate	**Training**
3	$\Delta\alpha$,$\Delta\beta$	yes	yes	no	no	intermediate	**Test**
4		yes	yes	no	no	intermediate	**Training**
4	$\Delta\alpha$,$\Delta\beta$	yes	yes	constant	yes	intermediate	**Test**
5		yes	yes	no	no	intermediate	**Training**
5	$\Delta\alpha$,$\Delta\beta$	yes	yes	sinusoidal	no	intermediate	**Test**
6		yes	yes	no	no	intermediate	**Training**
6	$\Delta\alpha$,$\Delta\beta$	yes	yes	no	no	AFT	**Test**

Table 1: Simulations carried out throughout Sect. 11

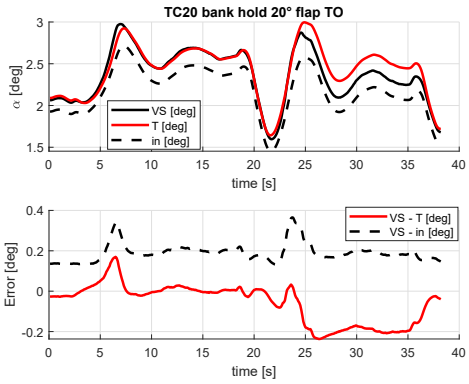
11.1 Without wind, AoA estimation - No GPS data

11.1.1 Longitudinal training

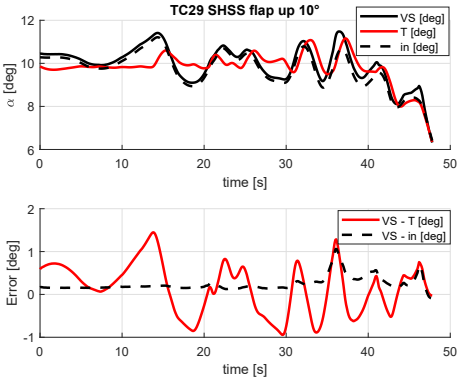
In first instance, the Neural Network is trained with longitudinal manoeuvres and tested for both longitudinal and latero-directional maneuvers. Several attempts are carried out at varying of the vector of inputs, the number of neurons and layers.
Chosen as initial vector of input $\mathbf{v} = \left[TAS, \dot{TAS}, n_x, n_y, n_z, \theta, q, \alpha_{in}\right]$ and a straightforward single layer architecture with 15 neurons, new parameters will be progressively added to $\mathbf{v}$ in order to assess if changing the NN's architecture would eventually lead to improvements in performances or not.
The first training set contains only *flap-up* maneuvers, then maneuvers with *flap-land* and *flap-TO* are added to understand if the NN would be able to work efficiently independently of the flaps configuration.
Results are reported in Figs.48, 49, 50, 51, 52, 53.



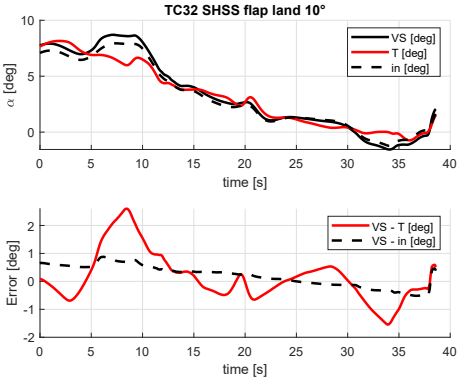
(a) TC3 Steady state 180kts



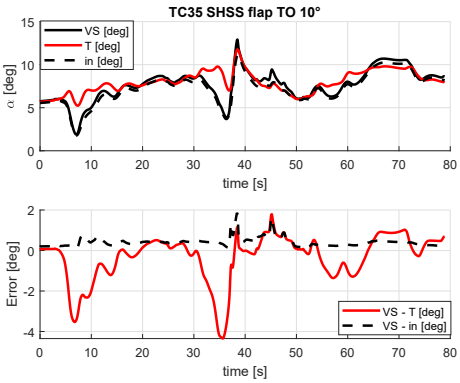
(b) TC20 bank hold 20° flap TO



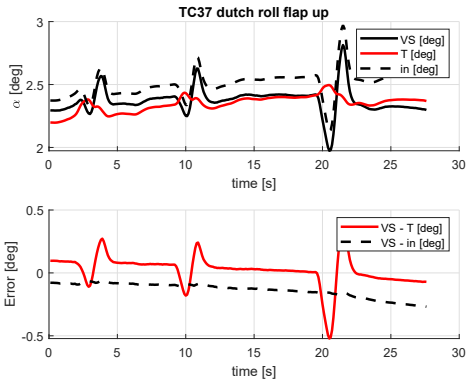
(c) TC29 SHSS flap UP 10°



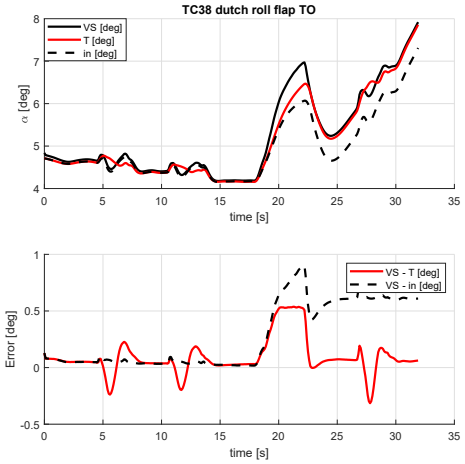
(d) TC32 SHSS flap LAND 20°



(e) TC35 SHSS flap TO 10°

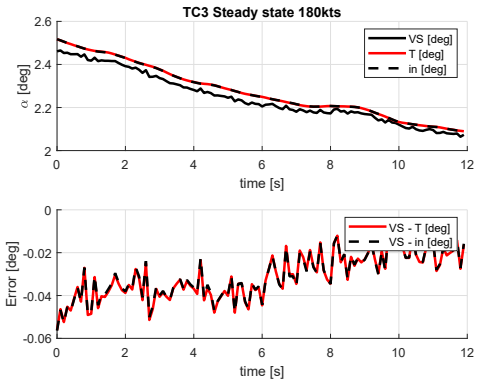


(f) TC37 dutch roll flap UP

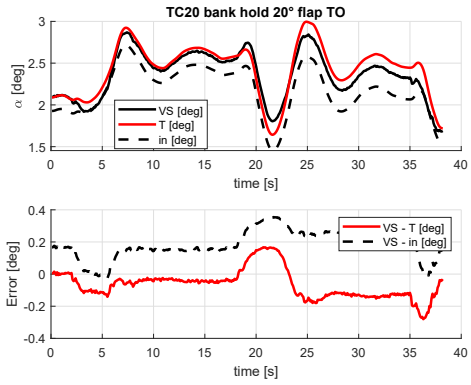


(g) TC38 dutch roll flap TO

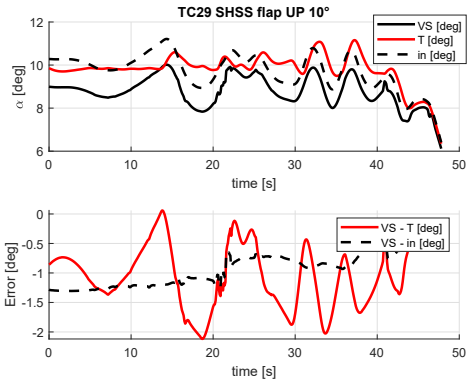
Figure 48: $\left[TAS, \dot{TAS}, n_x, n_z, \theta, q, \alpha_{in}\right]$, 15 neurons, AoA as delta, training set: no wind longitudinal flap up manoeuvres



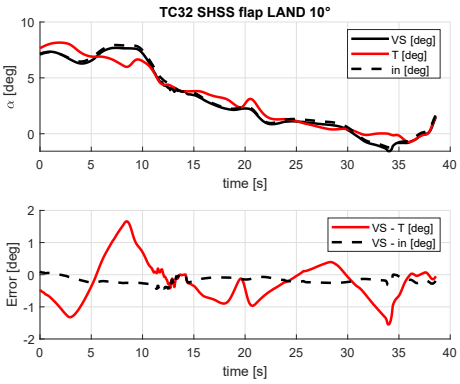
(a) TC3 Steady state 180kts



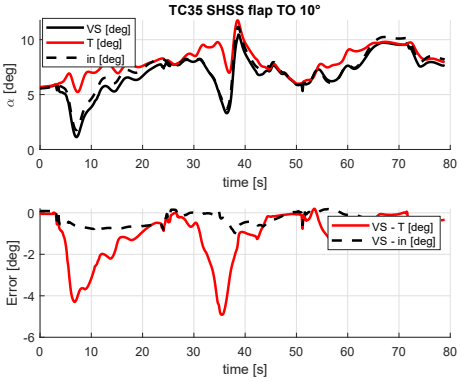
(b) TC20 bank hold 20° flap TO



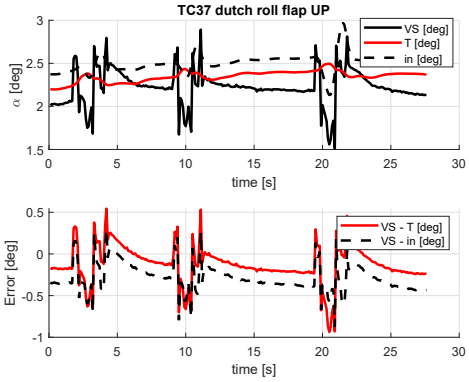
(c) TC29 SHSS flap UP 10°



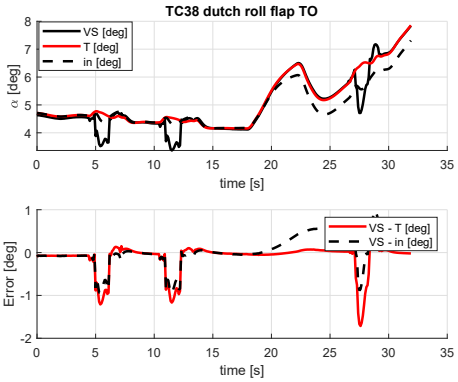
(d) TC32 SHSS flap LAND 20°



(e) TC35 SHSS flap TO 10°

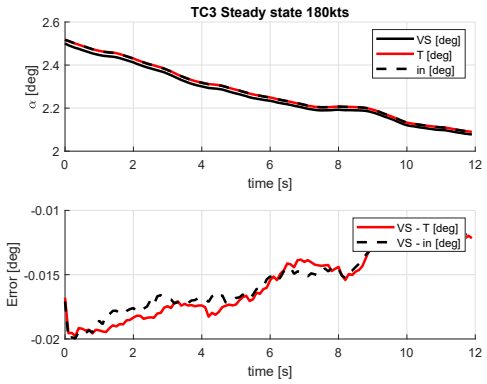


(f) TC37 dutch roll flap UP

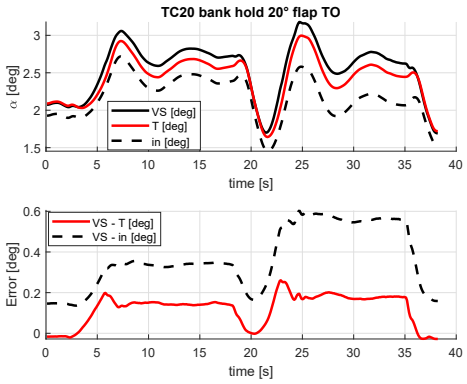


(g) TC38 dutch roll flap TO

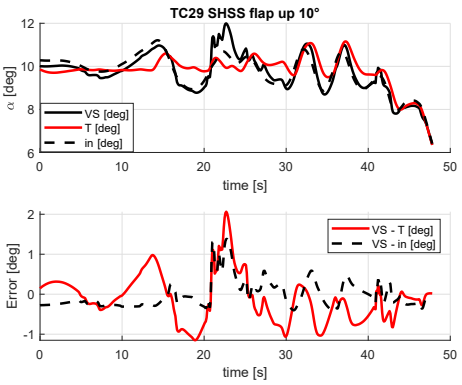
Figure 49: $\left[TAS, T\dot{A}S, n_x, n_y, n_z, \theta, q, \alpha_{in}\right]$, 15 neurons, AoA as delta, training set: no wind longitudinal flap up manoeuvres



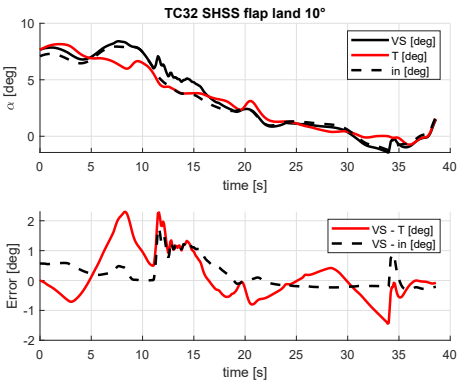
(a) TC3 Steady state 180kts



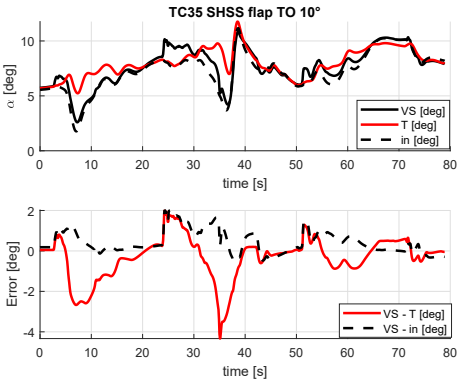
(b) TC20 bank hold 20° flap TO



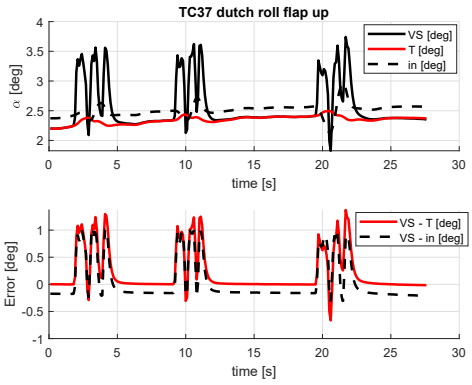
(c) TC29 SHSS flap UP 10°



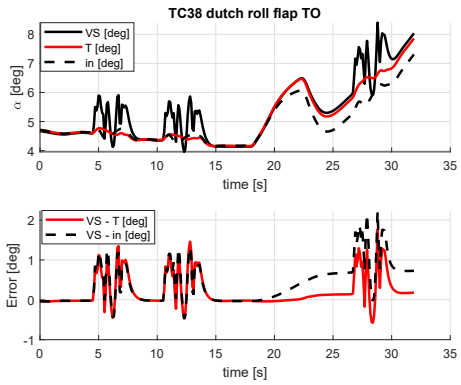
(d) TC32 SHSS flap LAND 20°



(e) TC35 SHSS flap TO 10°

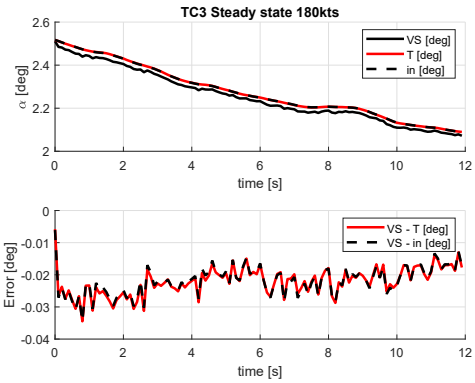


(f) TC37 dutch roll flap UP

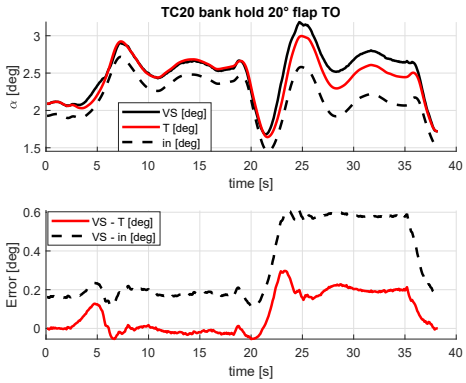


(g) TC38 dutch roll flap TO

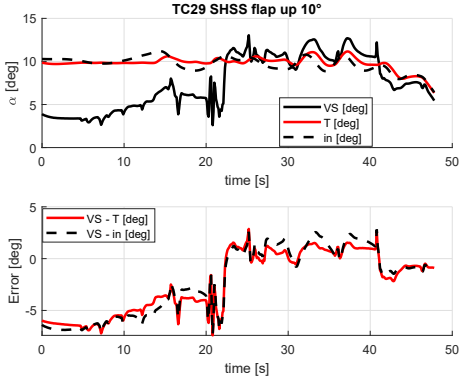
Figure 50: $\left[TAS, T\dot{A}S, n_x, n_z, \theta, q, r, \alpha_{in}\right]$, 15 neurons, AoA as delta, training set: no wind longitudinal flap up manoeuvres



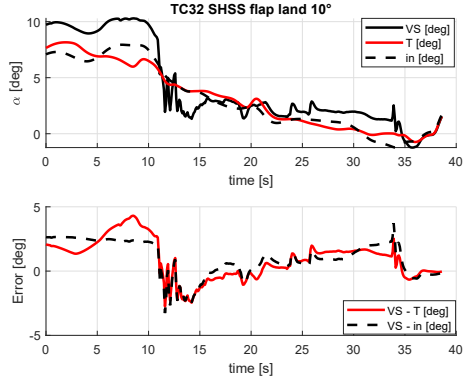
(a) TC3 Steady state 180kts



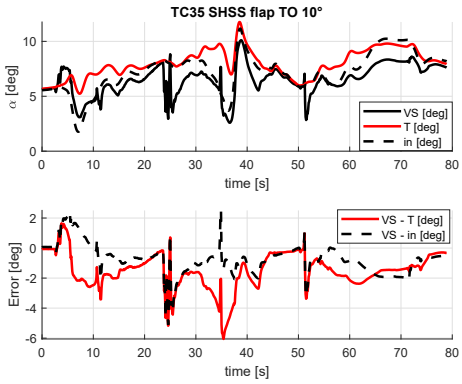
(b) TC20 bank hold 20° flap TO



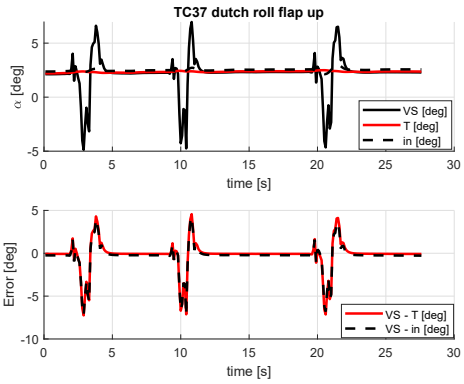
(c) TC29 SHSS flap UP 10°



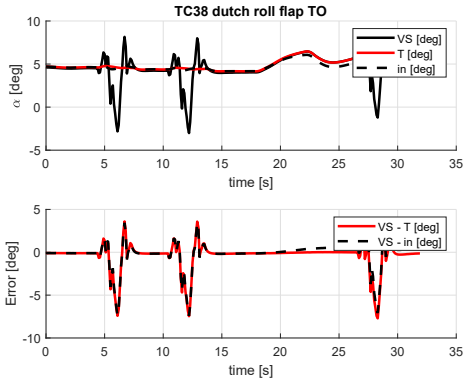
(d) TC32 SHSS flap LAND 20°



(e) TC35 SHSS flap TO 10°

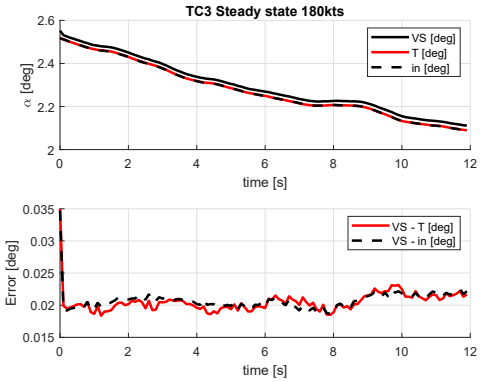


(f) TC37 dutch roll flap UP

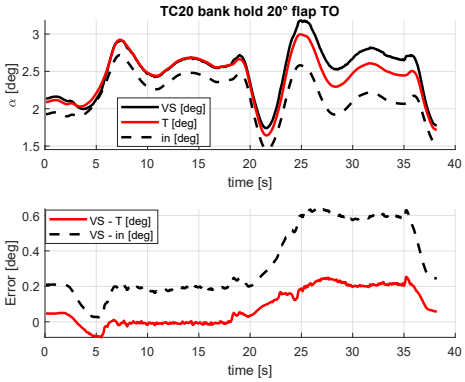


(g) TC38 dutch roll flap TO

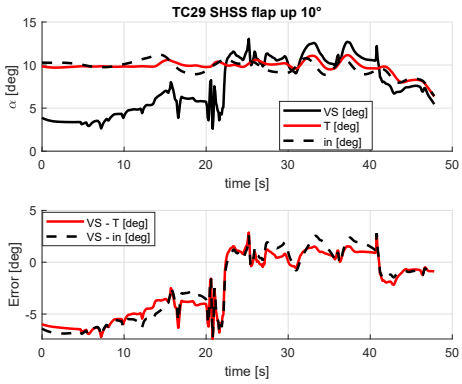
Figure 51: $\left[TAS, T\dot{A}S, n_x, n_y, n_z, \theta, q, r, \alpha_{in}\right]$, 15 neurons, AoA as delta, training set: no wind longitudinal flap up manoeuvres



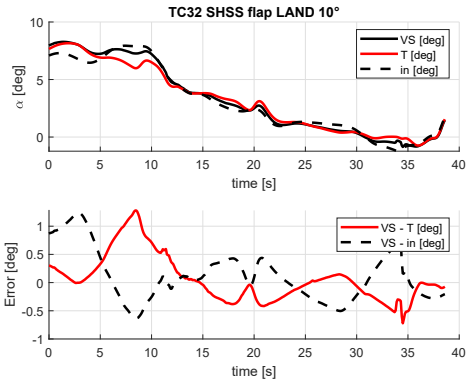
(a) TC3 Steady state 180kts



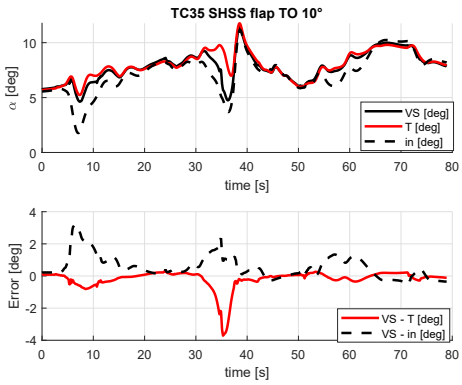
(b) TC20 bank hold 20° flap TO



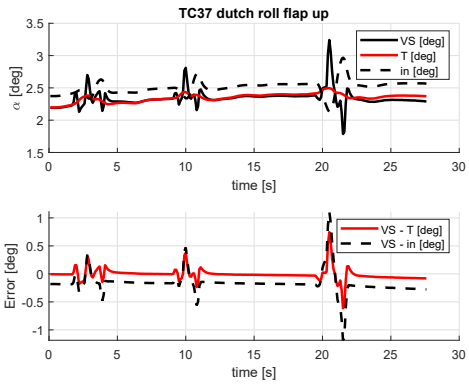
(c) TC29 SHSS flap UP 10°



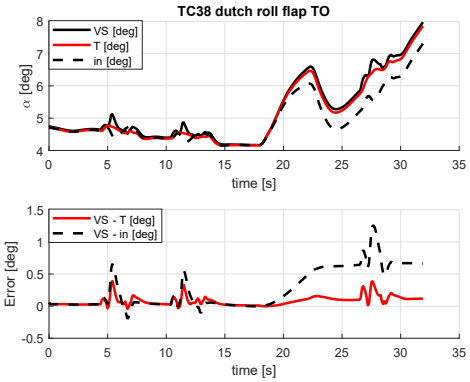
(d) TC32 SHSS flap LAND 20°



(e) TC35 SHSS flap TO 10°

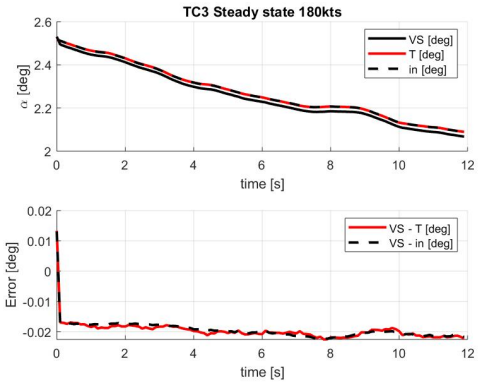


(f) TC37 dutch roll flap UP

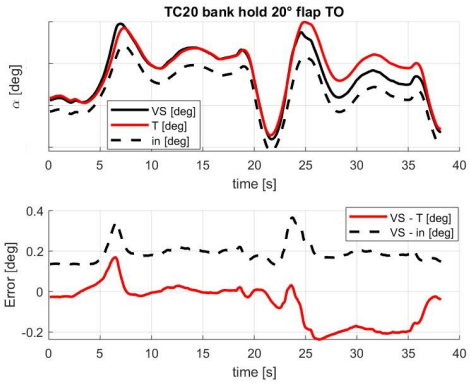


(g) TC38 dutch roll flap TO

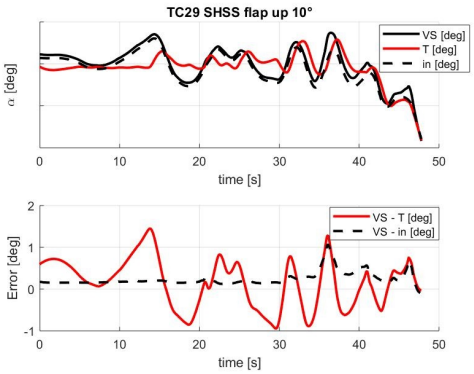
Figure 52: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \theta, q, r, \alpha_{in}\right]$, 20 neurons, AoA as delta, training set: no wind longitudinal flap up manoeuvres



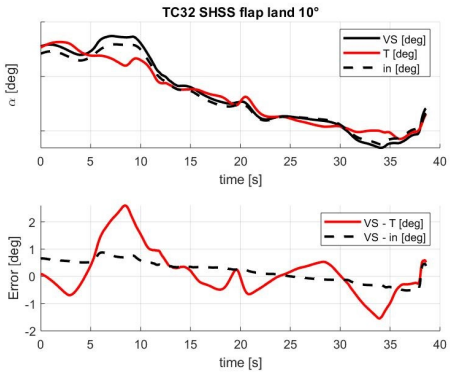
(a) TC3 Steady state 180kts



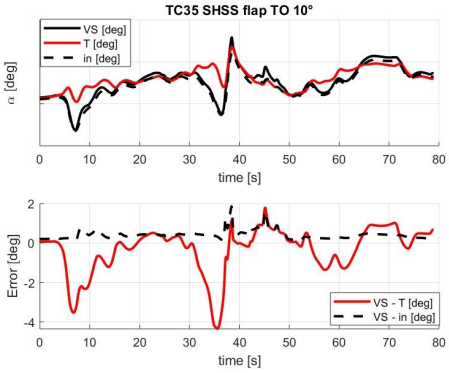
(b) TC20 bank hold 20° flap TO



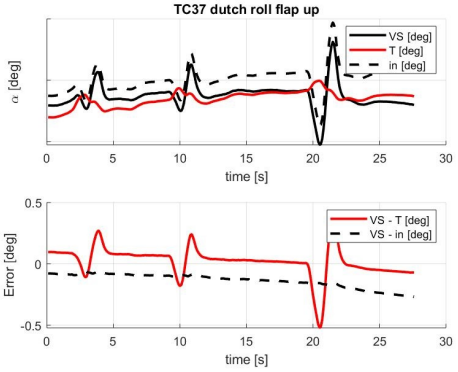
(c) TC29 SHSS flap UP 10°



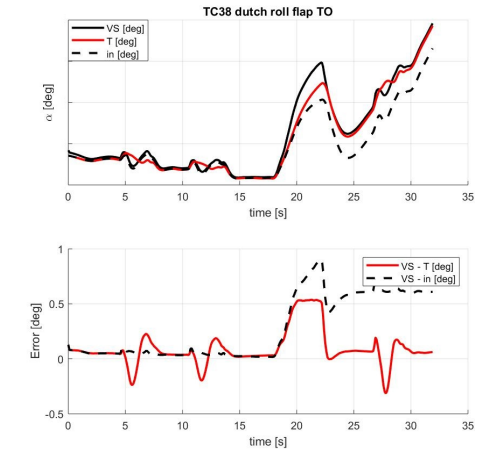
(d) TC32 SHSS flap LAND 20°



(e) TC35 SHSS flap TO 10°



(f) TC37 dutch roll flap UP



(g) TC38 dutch roll flap TO

Figure 53: $\left[TAS, T\dot{A}S, n_x, n_z, \theta, q, \alpha_{in}\right]$, 15 neurons, AoA as delta, training set: no wind longitudinal flap up and land manoeuvres

The first result that comes out is that changing the vector of input, the number of neurons or the NN's layers results ineffective on latero-directional manoeuvers if the training set only includes the longitudinal ones: in fact, in order to have good results in all the possible maneuvers, is necessary to have a complete training set that cover the largest possible part of flight evelope and includes both latero-directional both longitudinal manoeuvers. This confirms what previously said in Sect. 10 that the test set must be a subset of the training set.
The second result that comes out is that the NN is able to work efficiently independently of flap configuration.
An other interesting outcome is that if n_y or r are added to the basic vector of input $\mathbf{v}$, it can be observed, in all laterodirection maneuvers, the birth of oscillating instabilities that are due to sudden rudder variations. For example, if the rudder deflection evolution is plotted for *TC38 dutch roll flap TO* (Fig. 54), it can be remarked as the instabilities in VS solution arise exactly where the rudder is activated while the result is good where δ_r is in null position.

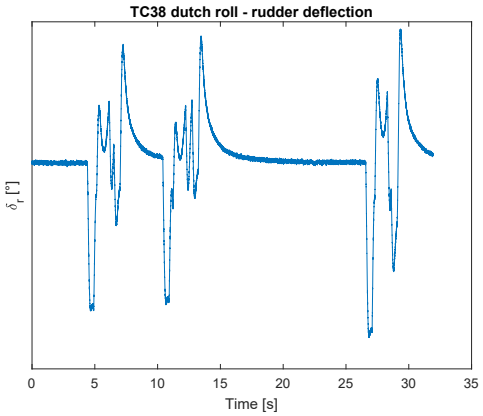
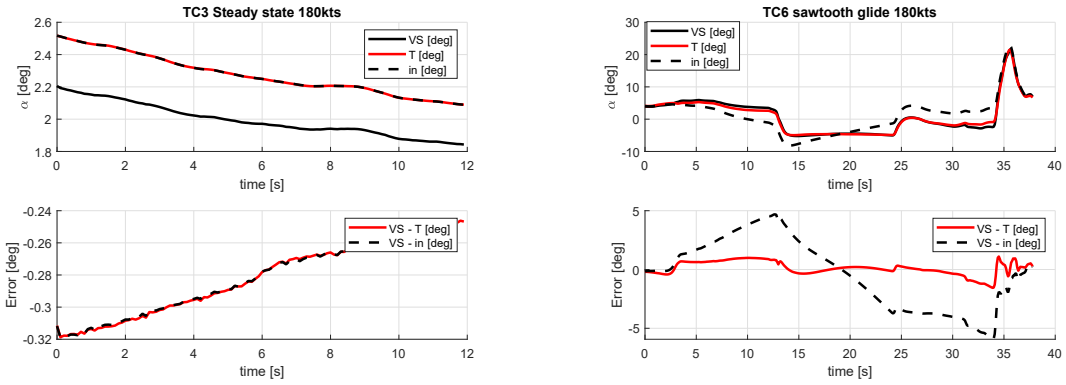


Figure 54: Rudder deflection in TC38 dutch roll manoeuvre

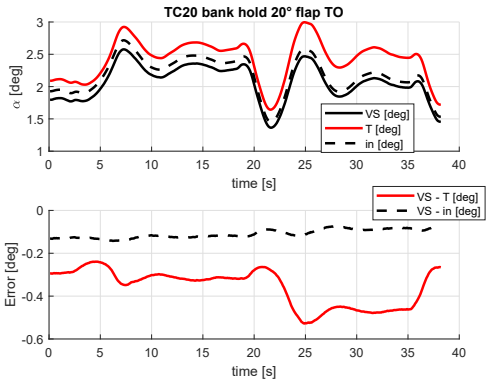
11.1.2 Longitudinal and latero-directional training

The next step consists in including both longitudinal and latero-directional manoeuvers in the training set. Once again, several attempts are carried out aimed to understand the minimum vector of input necessary to the VS to follow correctly the AoA trend provided by the real sensor.
Results of tests are reported hereafter.

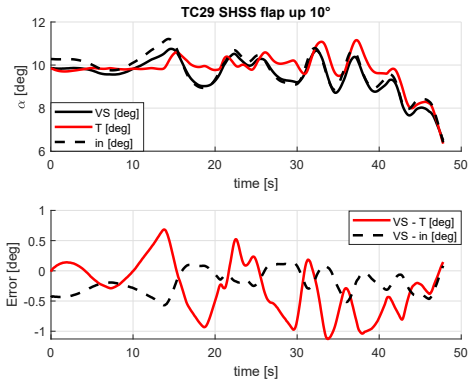


(a) TC3 Steady state 180kts

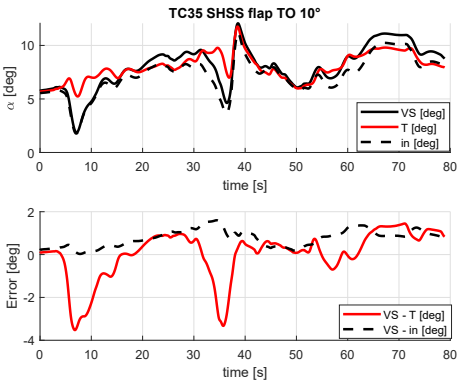
(b) TC6 Sawtooth glide 180kts



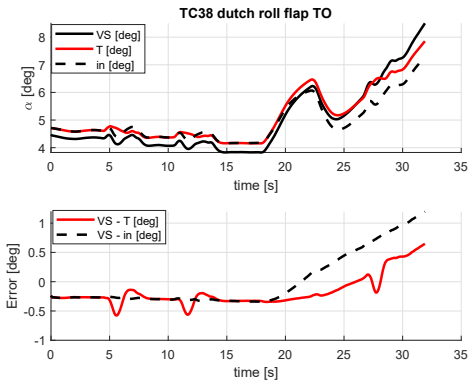
(c) TC20 bank hold 20° flap TO



(d) TC29 SHSS flap UP 10°

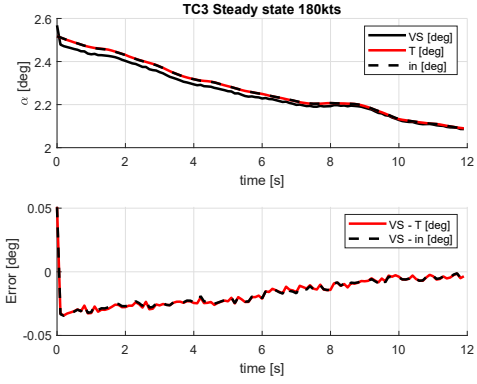


(e) TC35 SHSS flap TO 10°

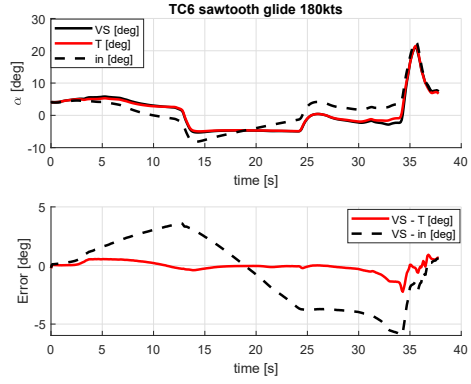


(f) TC38 dutch roll flap TO

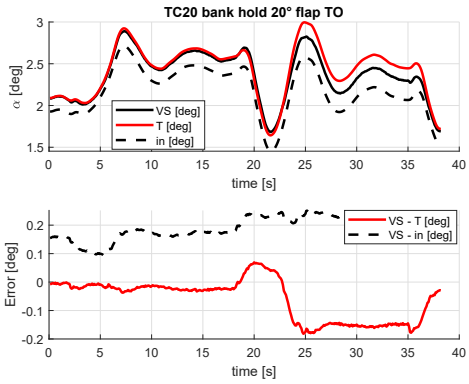
Figure 55: $\left[TAS, \dot{TAS}, n_x, n_z, \theta, q, \alpha_{in}\right]$, 15 neurons, AoA as delta, training set: no wind manoeuvers



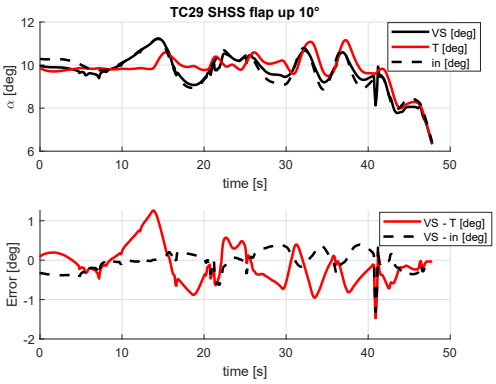
(a) TC3 Steady state 180kts



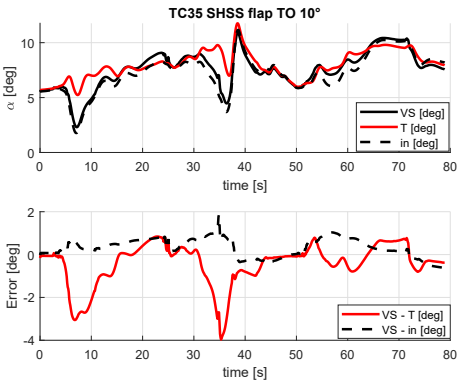
(b) TC6 Sawtooth glide 180kts



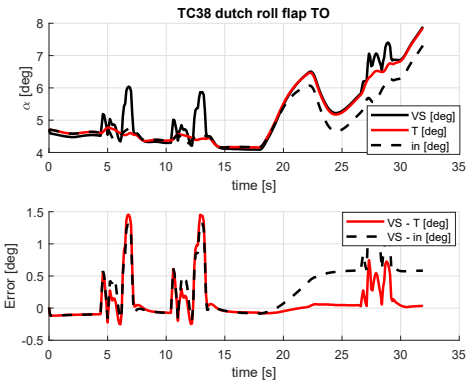
(c) TC20 bank hold 20° flap TO



(d) TC29 SHSS flap UP 10°

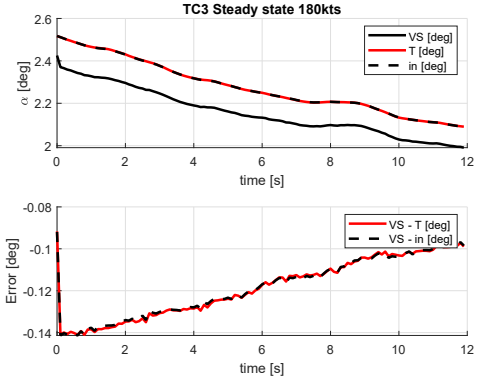


(e) TC35 SHSS flap TO 10°

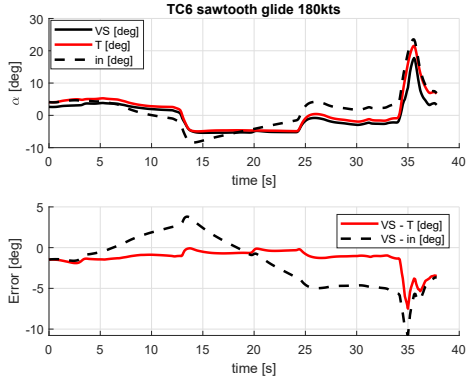


(f) TC38 dutch roll flap TO

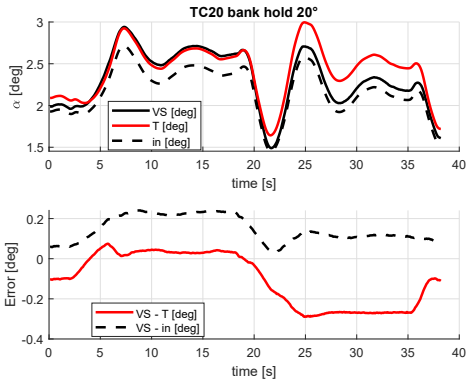
Figure 56: $\left[TAS, T\dot{A}S, n_x, n_y, n_z, \theta, q, \alpha_{in}\right]$, 15 neurons, AoA as delta, training set: no wind manoeuvers



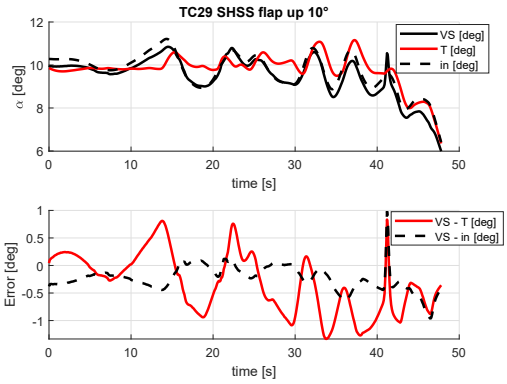
(a) TC3 Steady state 180kts



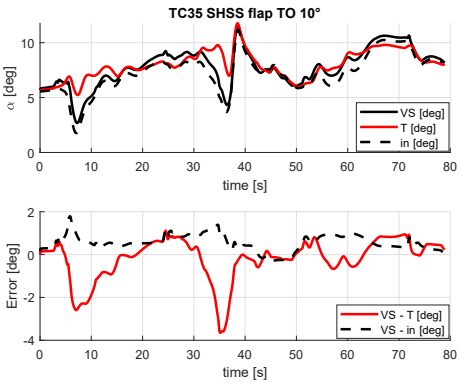
(b) TC6 Sawtooth glide 180kts



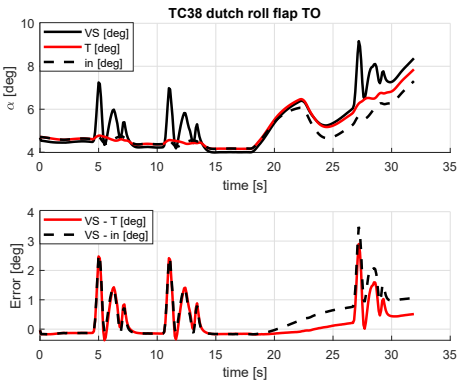
(c) TC20 bank hold 20° flap TO



(d) TC29 SHSS flap UP 10°

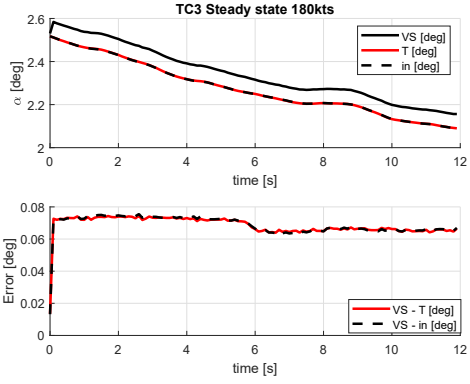


(e) TC35 SHSS flap TO 10°

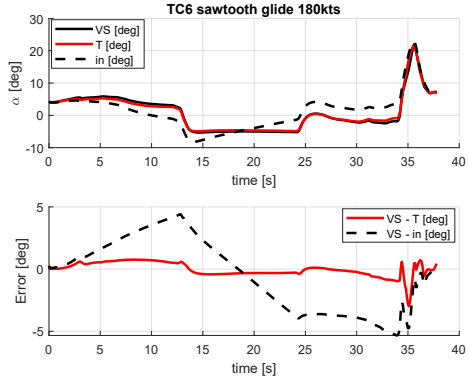


(f) TC38 dutch roll flap TO

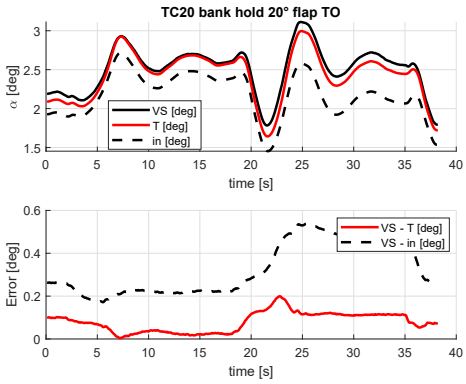
Figure 57: $\left[TAS, \dot{TAS}, n_x, n_z, \theta, q, r, \alpha_{in}\right]$, 15 neurons, AoA as delta, training set: no wind manoeuvers



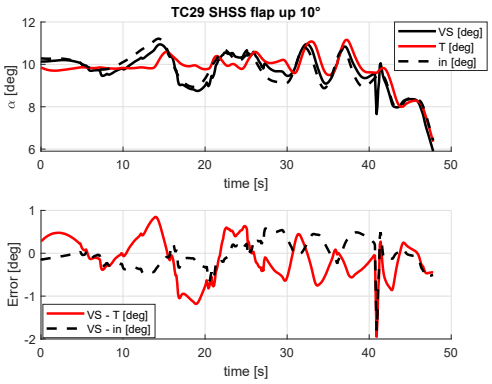
(a) TC3 Steady state 180kts



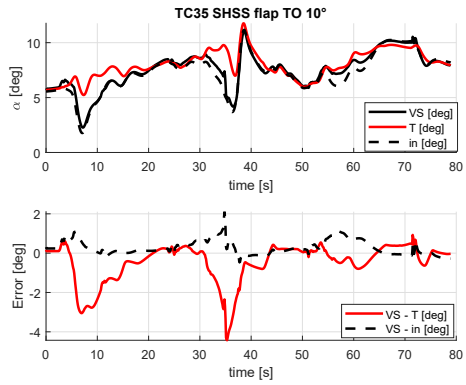
(b) TC6 Sawtooth glide 180kts



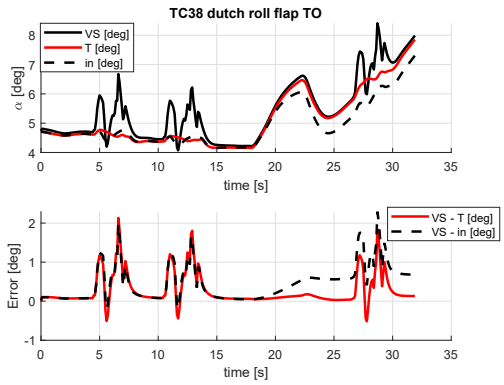
(c) TC20 bank hold 20° flap TO



(d) TC29 SHSS flap UP 10°

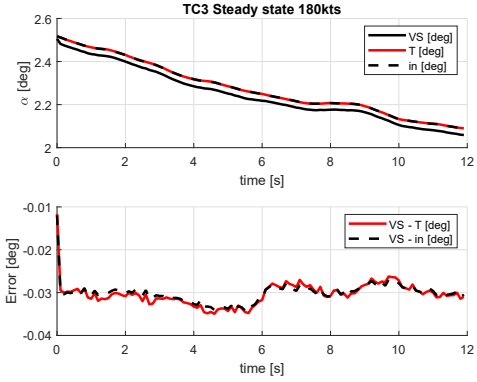


(e) TC35 SHSS flap TO 10°

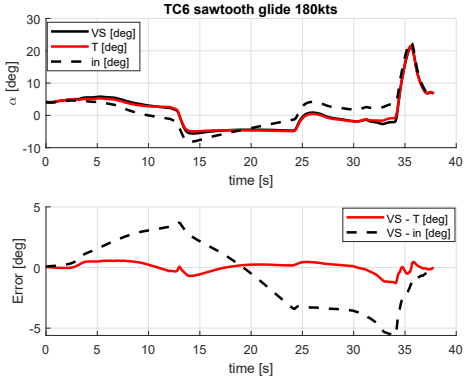


(f) TC38 dutch roll flap TO

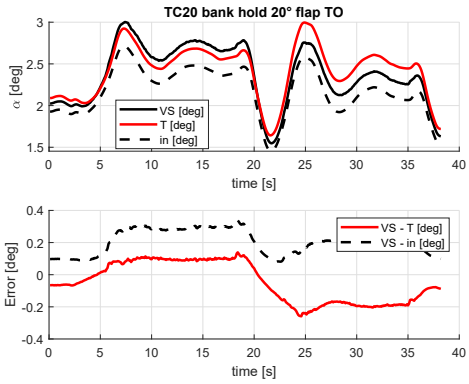
Figure 58: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \theta, q, r, \alpha_{in}\right]$, 15 neurons, AoA as delta, training set: no wind manoeuvers



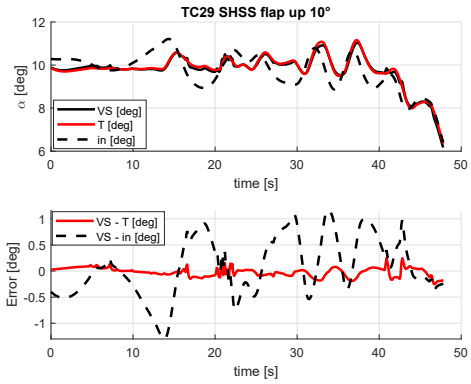
(a) TC3 Steady state 180kts



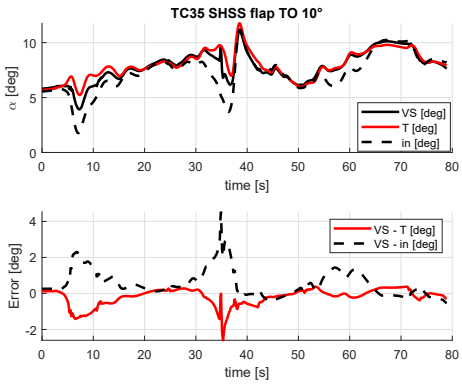
(b) TC6 Sawtooth glide 180kts



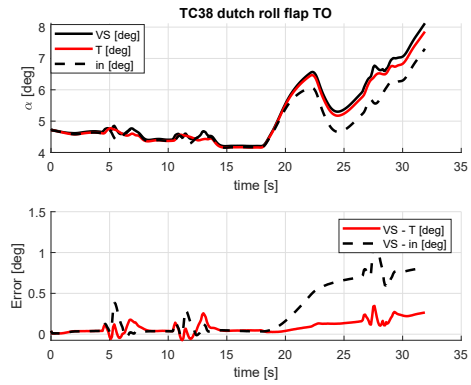
(c) TC20 bank hold 20° flap TO



(d) TC29 SHSS flap UP 10°

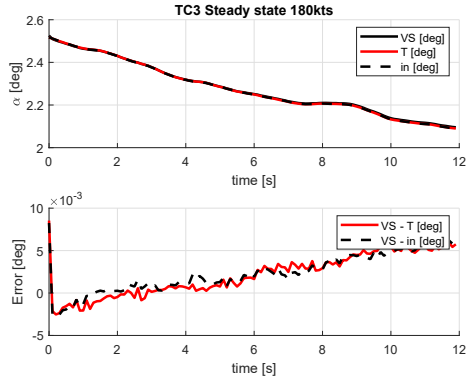


(e) TC35 SHSS flap TO 10°

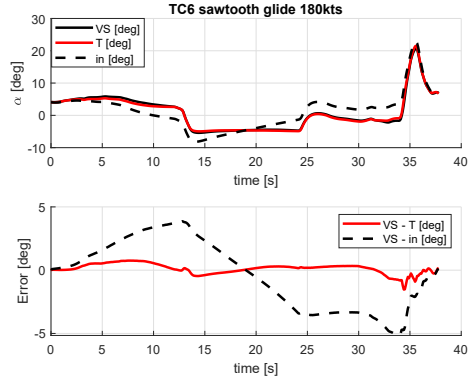


(f) TC38 dutch roll flap TO

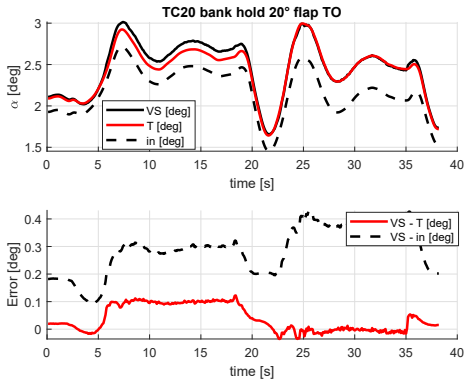
Figure 59: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, q, \alpha_{in}\right]$, 15 neurons, AoA as delta, training set: no wind manoeuvers



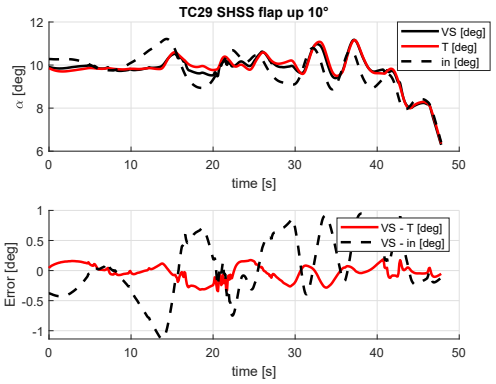
(a) TC3 Steady state 180kts



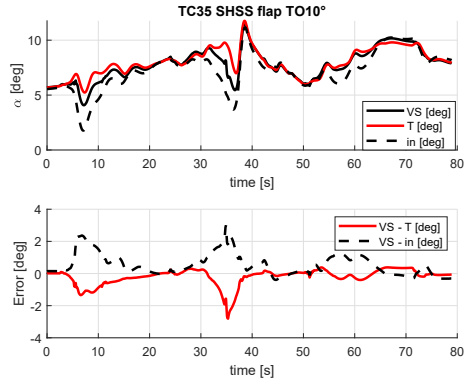
(b) TC6 Sawtooth glide 180kts



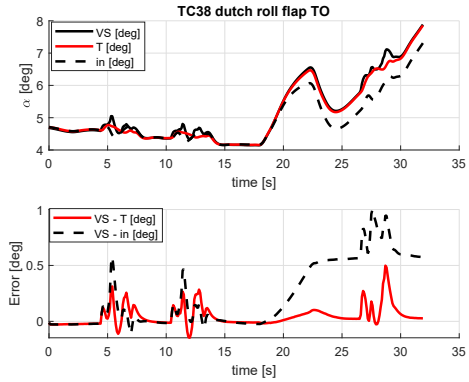
(c) TC20 bank hold 20° flap TO



(d) TC29 SHSS flap UP 10°



(e) TC35 SHSS flap TO 10°



(f) TC38 dutch roll flap TO

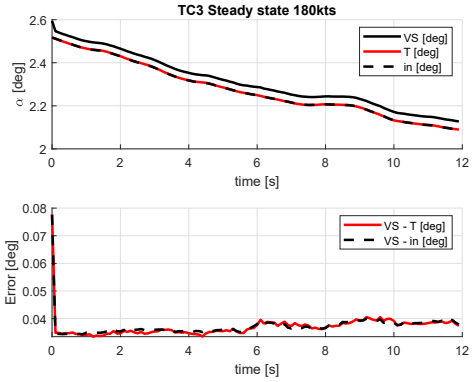
Figure 60: $\left[TAS, T\dot{A}S, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}\right]$, 15 neurons, AoA as delta, training set: no wind manoeuvers

The inclusion of latero-directional maneuvers in the training set actually leads to improvements in NN's performances. It can be observed that $\left[TAS, T\dot{A}S, n_x, n_y, n_z, \phi, \theta, q, \alpha_{in}\right]$ is the minimum optimum set of parameter which returns an error on AoA estimate well confined between $\pm 1.5°$ in all the manoeuvers. It is also remarkable that adding to the vector of inputs more parameters than necessary is counterproductive: for example, adding the yaw rate r results in oscillations and deterioration of the solution.

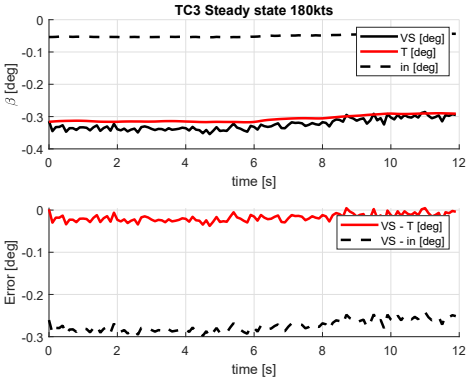
11.2 Without wind, AoA and AoS estimation - No GPS data

Once determined the minimum optimum vector of input for AoA estimation, the next step consists in understanding which parameters, typical of latero-directional motion, are

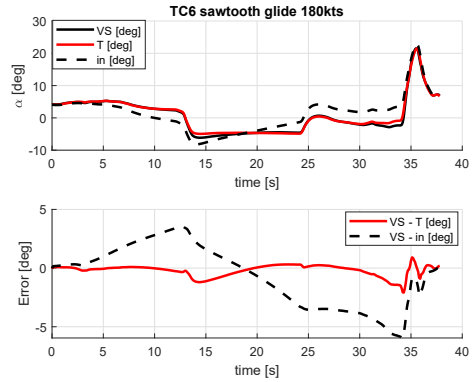
needed to be added to the previous vector in order to make the NN able to correctly estimate AoS as well. Given that the yaw rate r is one of these parameters, commands like aileron, rudder, elevator and throttle deflections have been added to evaluate their importance for NN training. The NN is trained using a complete balanced set of both longitudinal both latero-directional maneuvers.



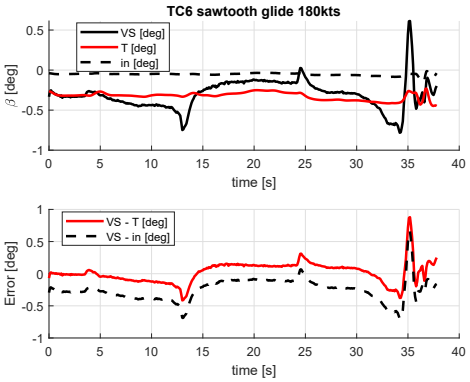
(a) TC3 Steady state 180kts



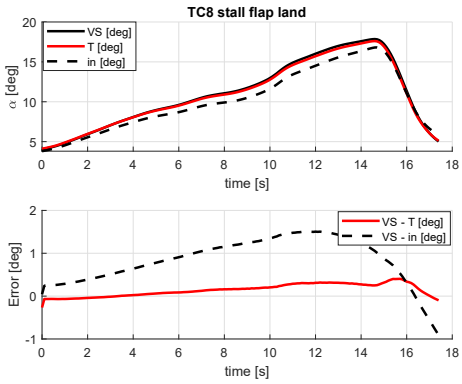
(b) TC3 Steady state 180kts



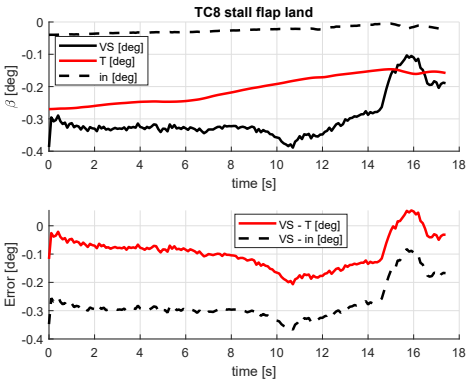
(c) TC6 Sawtooth glide 180kts



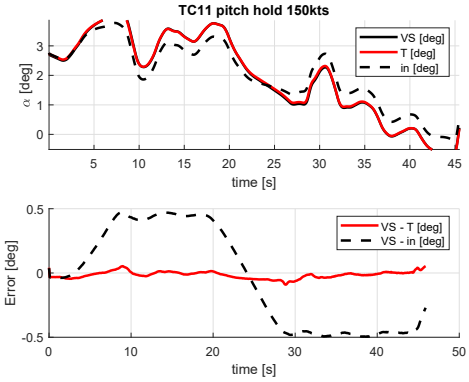
(d) TC6 Sawtooth glide 180kts



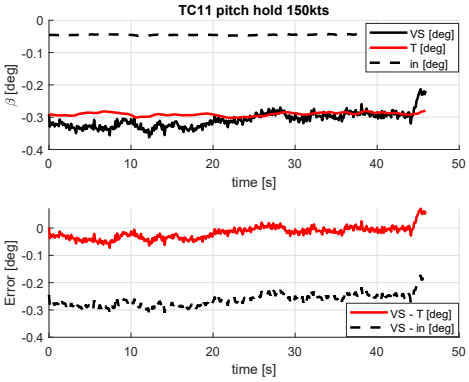
(e) TC8 stall flap LAND



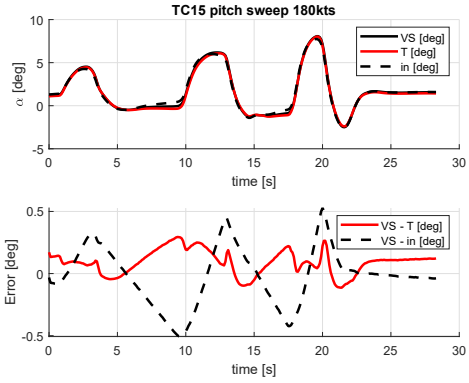
(f) TC8 stall flap LAND



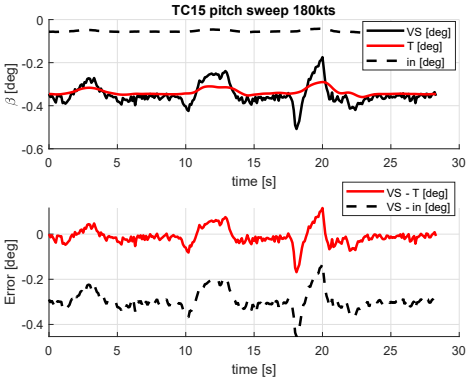
(g) TC11 pitch hold 150kts



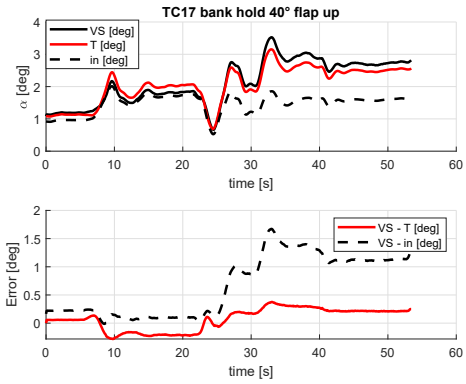
(h) TC11 pitch hold 150kts



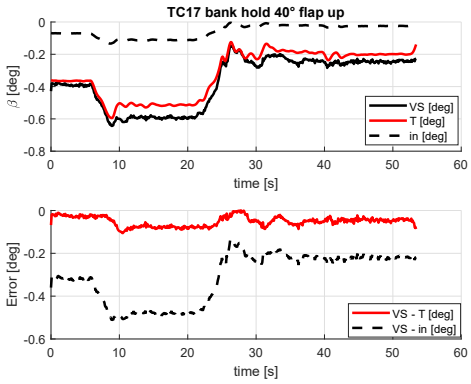
(i) TC15 pitch sweep 180kts



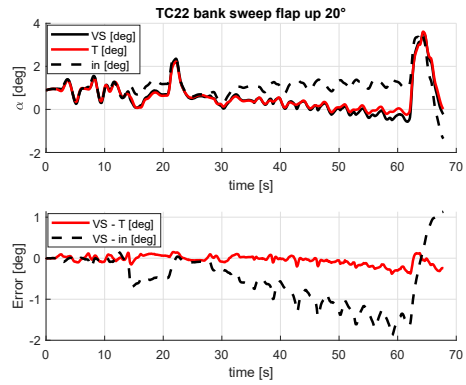
(j) TC15 pitch sweep 180kts



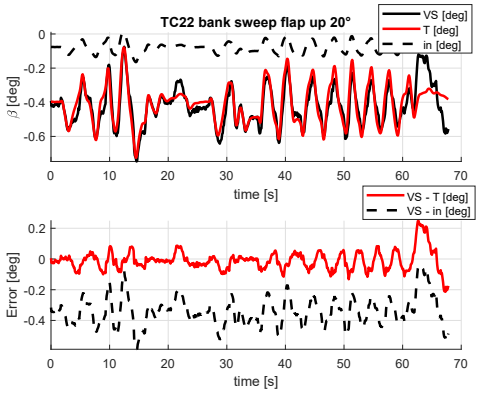
(k) TC17 bank hold flap UP 40°



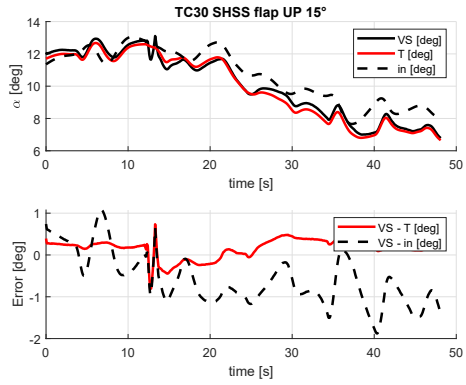
(l) TC17 bank hold flap UP 40°



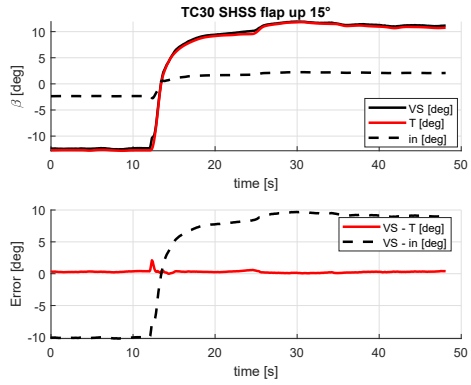
(m) TC22 bank sweep flap UP 20°



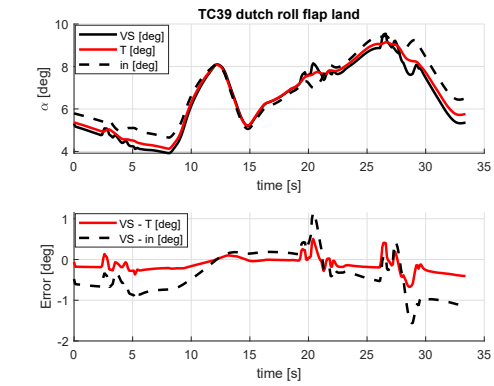
(n) TC22 bank sweep flap UP 20°



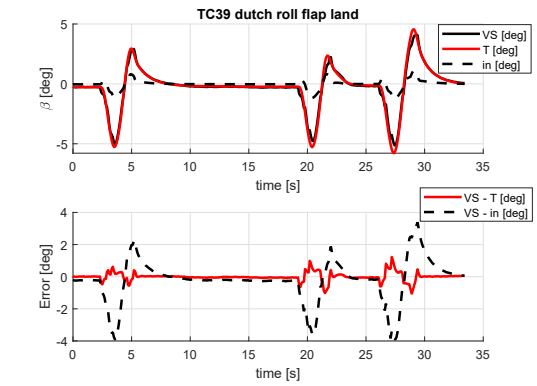
(o) TC30 SHSS flap UP 15°



(p) TC30 SHSS flap UP 15°

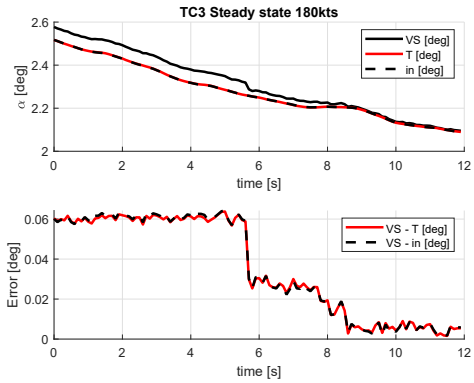


(q) TC39 dutch roll flap LAND

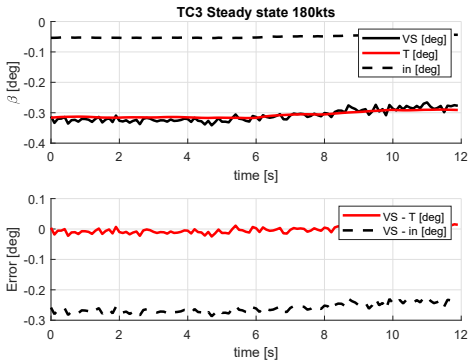


(r) TC39 dutch roll flap LAND

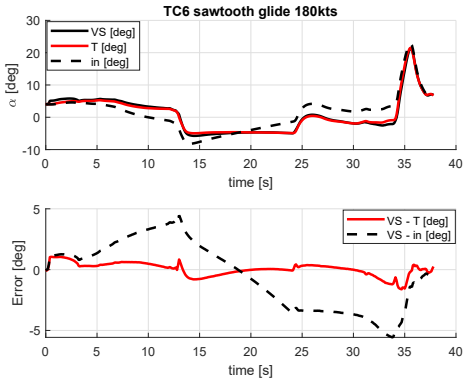
Figure 61: $\left[TAS, T\dot{A}S, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}, \beta_{in}, \delta_a, \delta_r\right]$, 15 neurons, AoA and AoS as delta, training set: no wind maneuvers



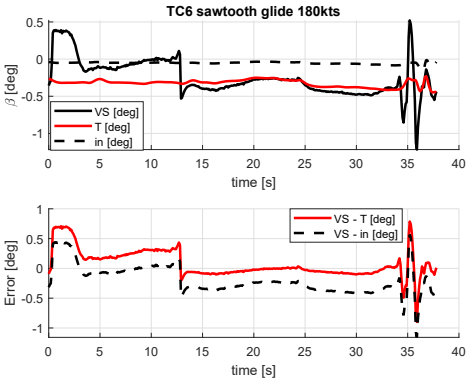
(a) TC3 Steady state 180kts



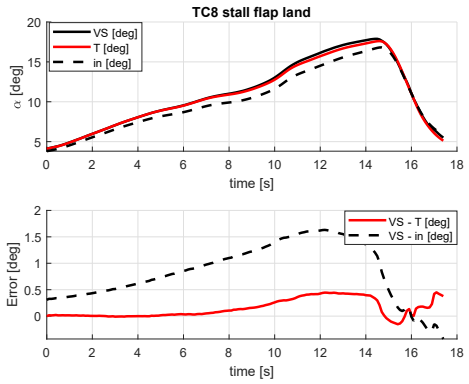
(b) TC3 Steady state 180kts



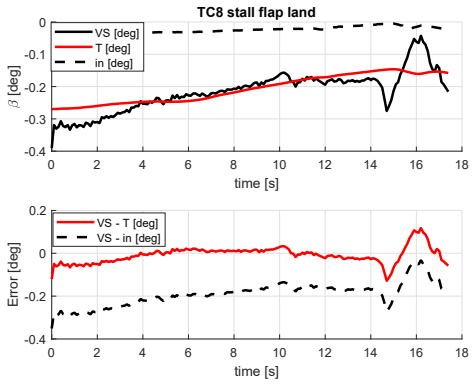
(c) TC6 Sawtooth glide 180kts



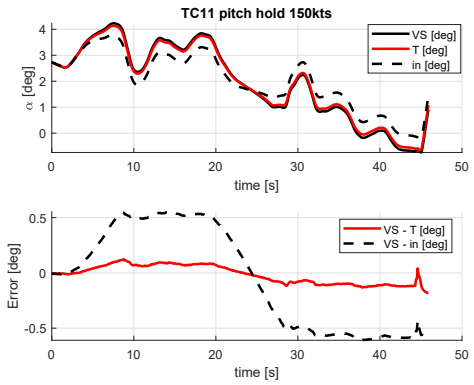
(d) TC6 Sawtooth glide 180kts



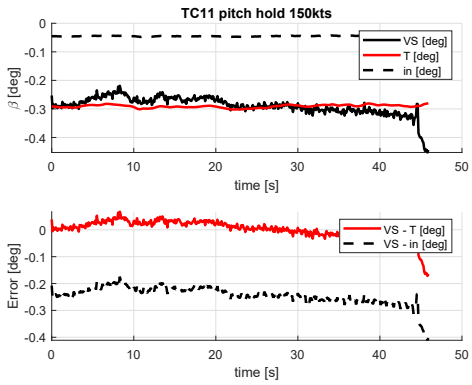
(e) TC8 stall flap LAND



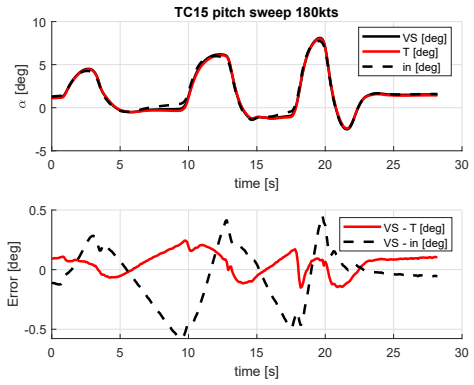
(f) TC8 stall flap LAND



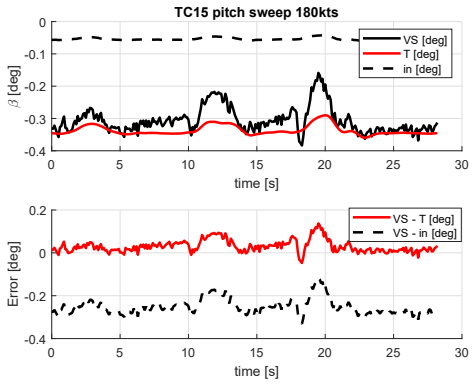
(g) TC11 pitch hold 150kts



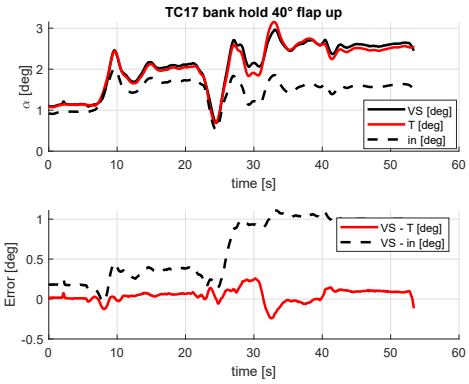
(h) TC11 pitch hold 150kts



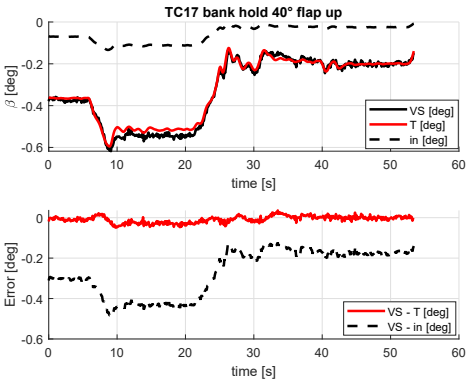
(i) TC15 pitch sweep 180kts



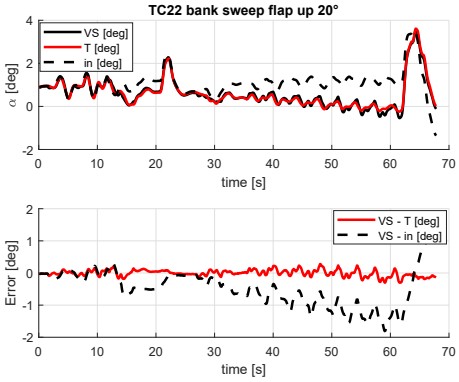
(j) TC15 pitch sweep 180kts



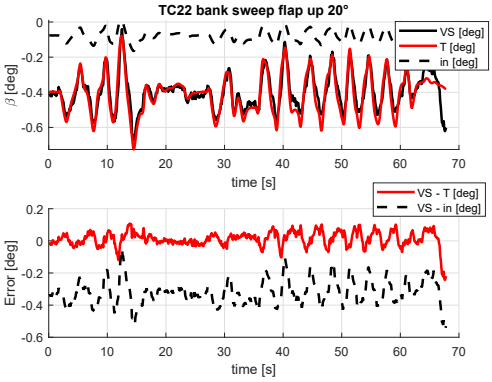
(k) TC17 bank hold flap UP 40°



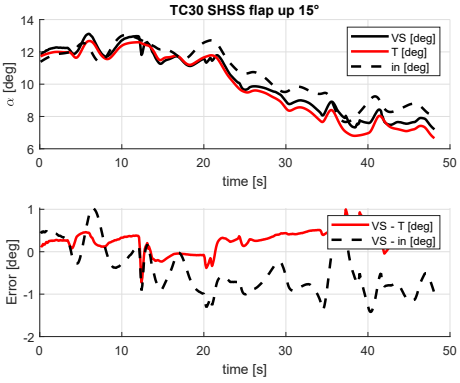
(l) TC17 bank hold flap UP 40°



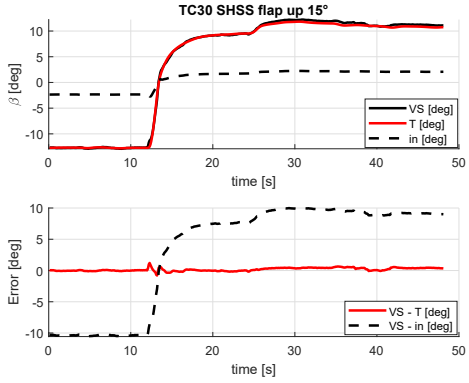
(m) TC22 bank sweep flap UP 20°



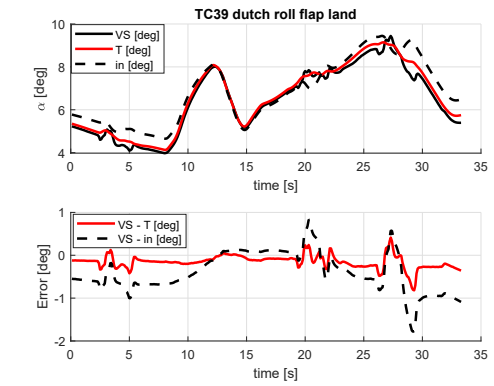
(n) TC22 bank sweep flap UP 20°



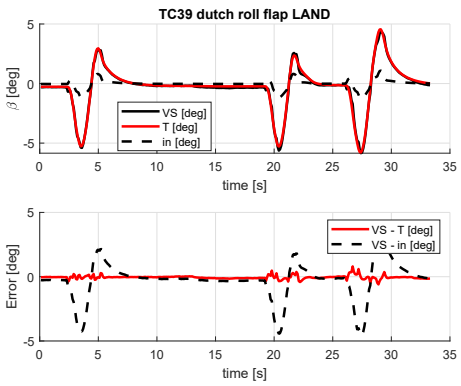
(o) TC30 SHSS flap UP 15°



(p) TC30 SHSS flap UP 15°



(q) TC39 dutch roll flap LAND



(r) TC39 dutch roll flap LAND

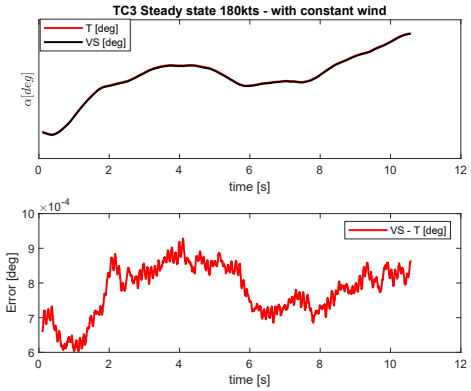
Figure 62: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}, \beta_{in}, \delta_a, \delta_r, \delta_e, \delta_{th,LH}, \delta_{th,RH}\right]$, 15 neurons, AoA and AoS as delta, training set: no wind maneuvers

As can be observed from the results in Figs. 61, 62, there is a good correspondence between the solution estimated by the real sensor and that given by the VS: the absolute error is overall near to zero with only some peaks that are, on the other hand, always lower than 1.5°. The VS shows the worst behaviour when it has to estimate AoS in logitudinal maneuvers. On the other hand, as the sideslip angle is generally small throughout all these maneuvers, even if the trends of VS and True curves are very different, the absolute value of AoS error remains always very low ($< 1°$). Another outcome is that the only commands necessary for the VS estimation are the ailerons and rudder deflections; on the other hand the addition of left and right throttles and the elevator deflections does not lead to improvements.
It is relevant to remark once more that the training set has been well balancend between longitudinal and latero-directional maneuvers. An unbalanced set in one sense or in the other would have bring to unacceptable results. Moreover it is worths to be noticed that adding latero-directional maneuvers to training set or adding latero-directional parameter to the vector of inputs does not deteriorate the AoA extimation carried out before.
Thus, on the basis of these consideration, it can be generally stated that the Neural Network works successfully if trained and tested with no-wind simulator's data and it is GPS-free.

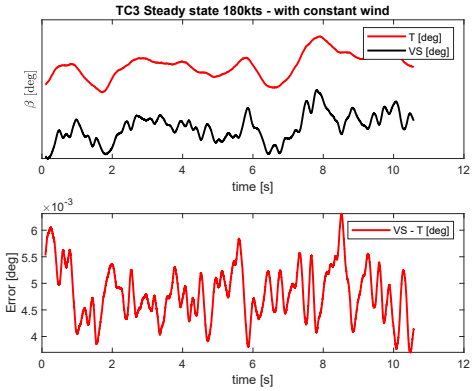
11.3 With constant wind

The following step consists in testing the NN in presence of constant cross wind and gust. In particular, for TC1-TC9 and TC16-TC39 is imposed the disturbance of a contrary constant cross-wind (that is with no component along z_w direction) with an intensity of 35 kts and direction 37.2° with respect to the geographic North; on the other hand, for TC10-TC15, is set the superimposition of a constant cross-wind with an intensity of 10.2 kts, direction $-137.7°$ N and a gust speed of 3.71 kts equal in all the direction (that is

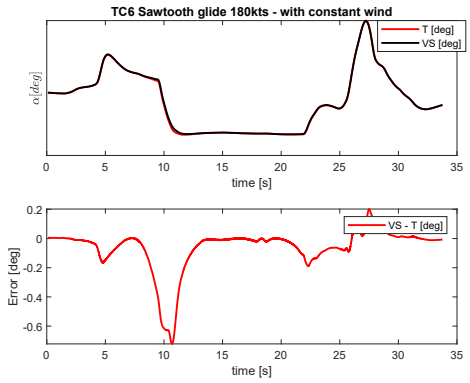
with both in the $x_w - y_w$ plane components both off the plane).
Named $\text{NN}_{pat,nw}$ the Neural Network that has been found to work in Sect. 11.2, it is convenient to verify if this NN shows good performances in presence of external constant wind as well.



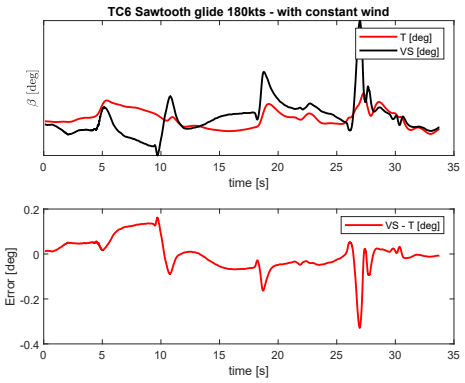
(a) TC3 Steady state 180kts



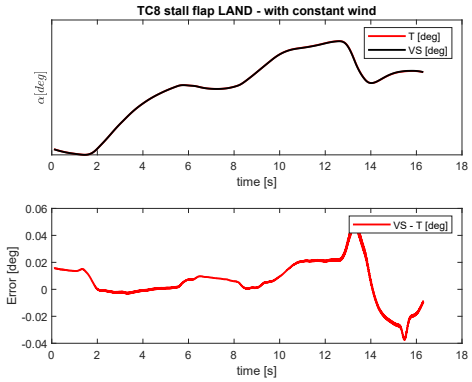
(b) TC3 Steady state 180kts



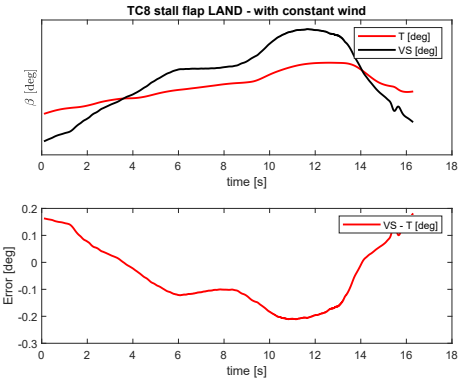
(c) TC6 Sawtooth glide 180kts



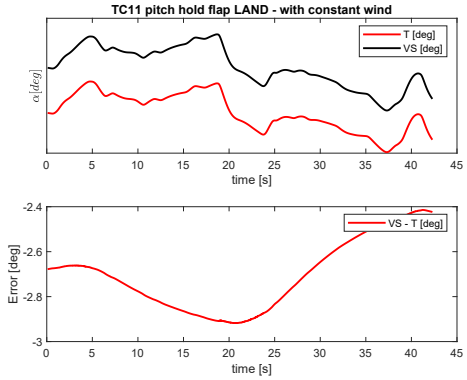
(d) TC6 Sawtooth glide 180kts



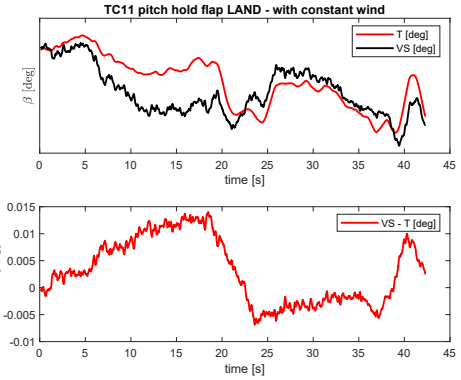
(e) TC8 stall flap LAND



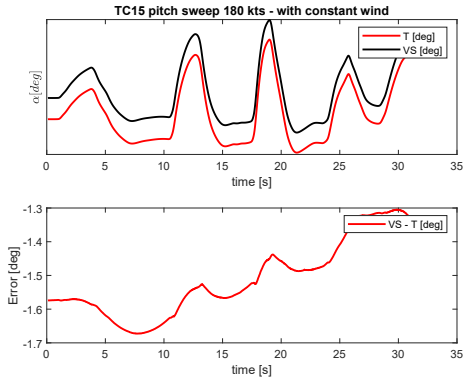
(f) TC8 stall flap LAND



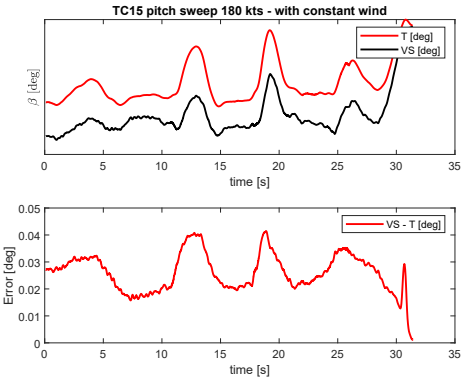
(g) TC11 pitch hold 150kts



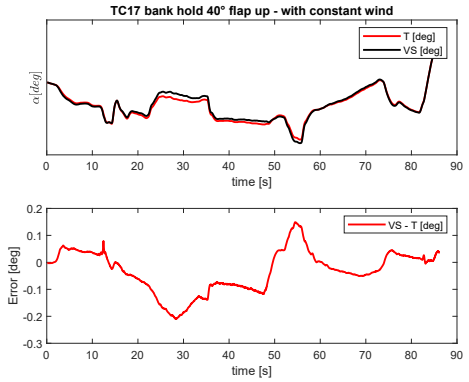
(h) TC11 pitch hold 150kts



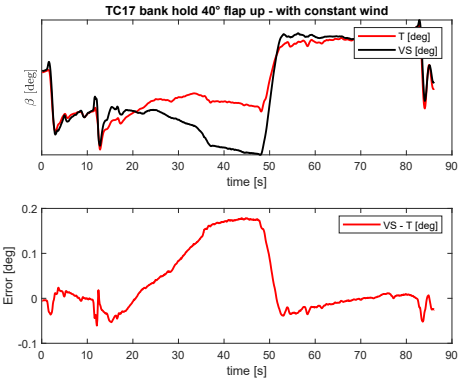
(i) TC15 pitch sweep 180kts



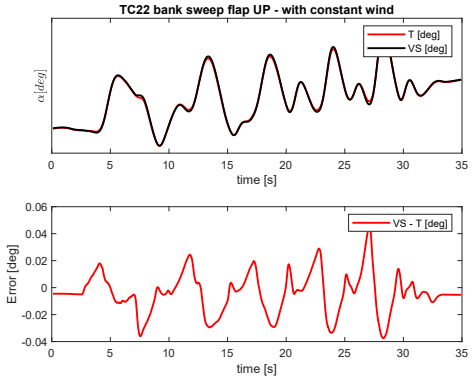
(j) TC15 pitch sweep 180kts



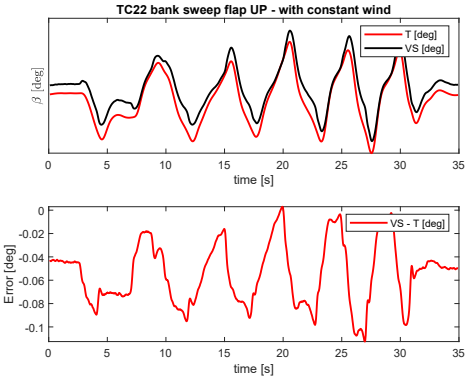
(k) TC17 bank hold flap UP 40°



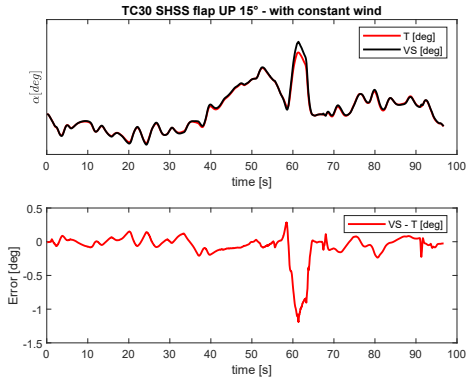
(l) TC17 bank hold flap UP 40°



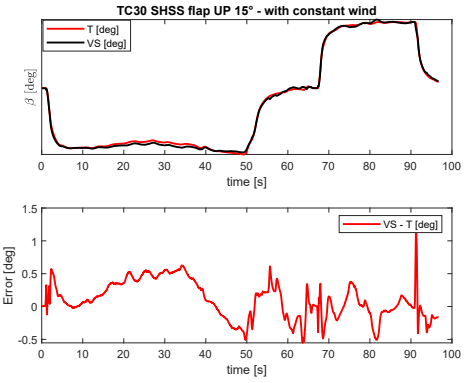
(m) TC22 bank sweep flap UP 20°



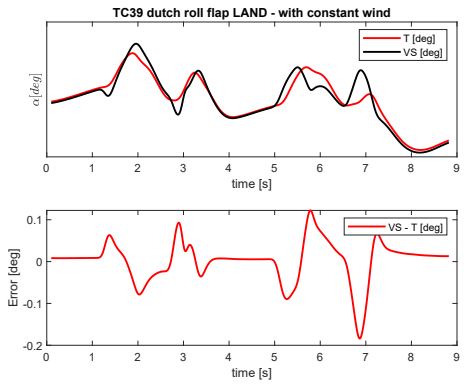
(n) TC22 bank sweep flap UP 20°



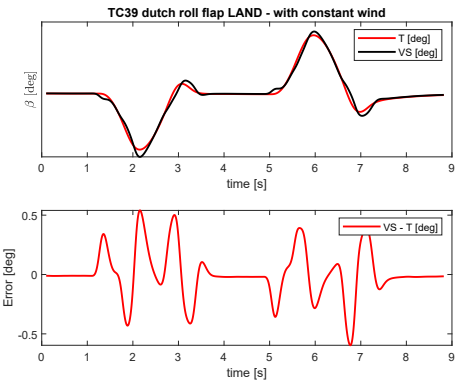
(o) TC30 SHSS flap UP 15°



(p) TC30 SHSS flap UP 15°



(q) TC39 dutch roll flap LAND



(r) TC39 dutch roll flap LAND

Figure 63: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}, \beta_{in}, \delta_a, \delta_r\right]$, 15 neurons, AoA as delta, training set: no wind maneuvers, testing set: with constant wind maneuvers

Even if in some cases (as can be seen in see Figs.63(i), 63(g)) the error's absolute value is quite large, in general it can be stated that the NN is able to capture the maneuver's dynamic successfully. What comes out from these last tests is that the largest absolute errors actually occur in those maneuvers where there is the superimposition of cross-wind and gust speed (Fig.s 63(g), 63(i)): in particular, this only results in an offset of the VS estimation whereas the AoA/AoS trend is mostly well respected. On the other hand, for what concerns tests where only constant cross-wind is imposed, it stands out as the insertion of with constant cross-wind maneuvers (instead of no-wind maneuvers only) within the training set is irrelevant for the NN's accuracy.

11.4 With sinusoidal wind

Known that the NN works successfully if tested on maneuvers with constant wind, it is now interesting to evaluate if good performance are achieved in other more complex wind situations as well.
TC3 Steady state, TC6 Sawtooth glide, TC13 Pitch sweep flap up, TC18 Bank hold 20° flap land, TC33 SHSS 15° flap land and TC39 Dutch roll have been repeated with the superimposition of sinusoidal wind (see Fig. 64) and used to test the previous NN that was trained with only no-wind maneuvers. Results are reported in Fig. 65.

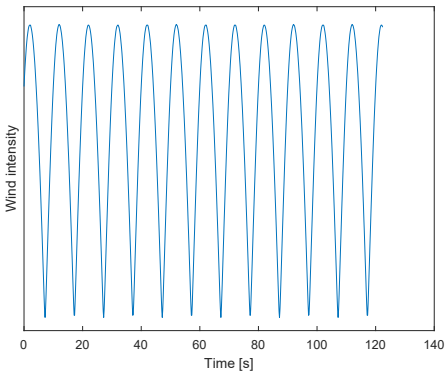
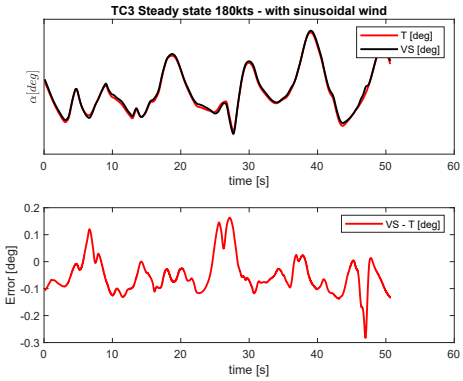
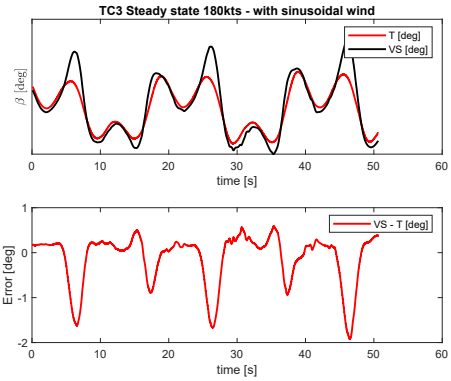


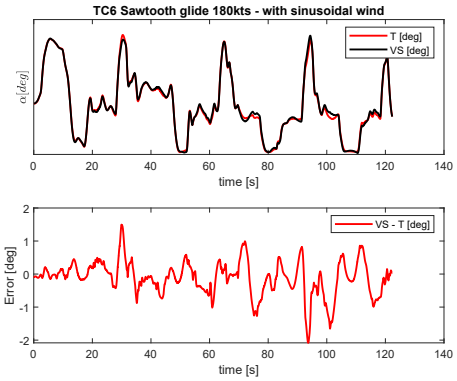
Figure 64: Sinusoidal wind intensity



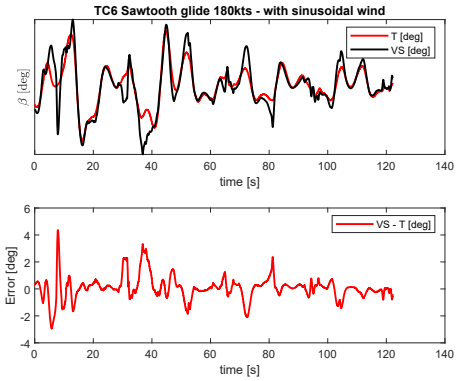
(a) TC3 Steady state 180kts



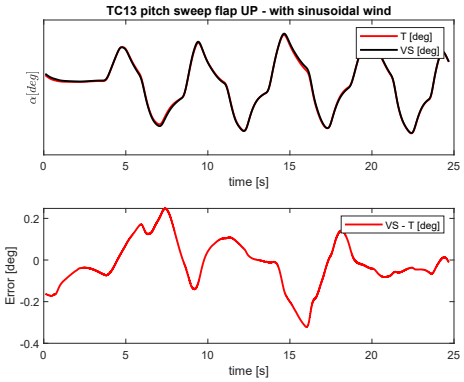
(b) TC3 Steady state 180kts



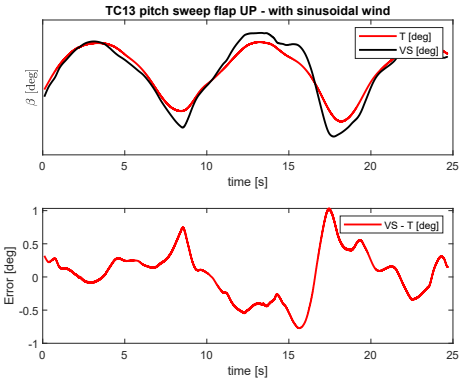
(c) TC6 Sawtooth glide 180kts



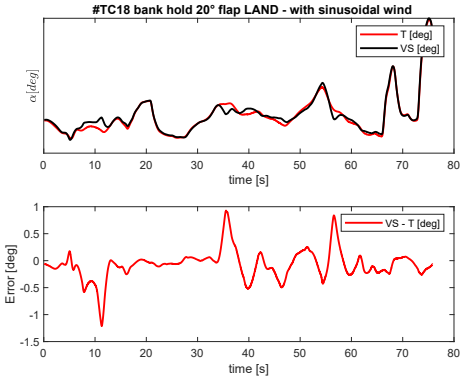
(d) TC6 Sawtooth glide 180kts



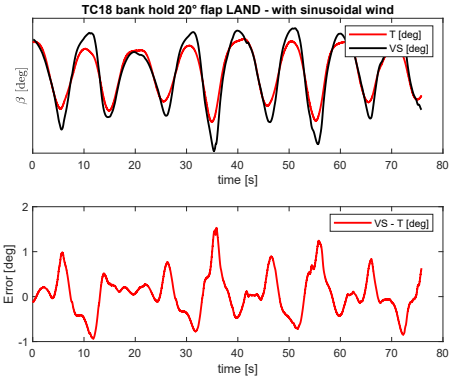
(e) TC13 pitch sweep flap UP



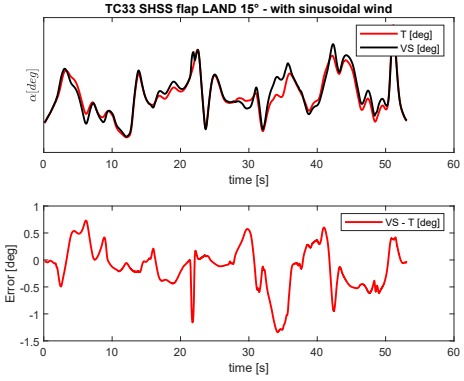
(f) TC13 pitch sweep flap UP



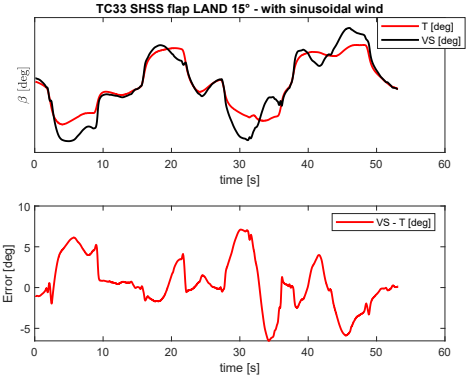
(g) TC18 bank hold flap LAND 20°



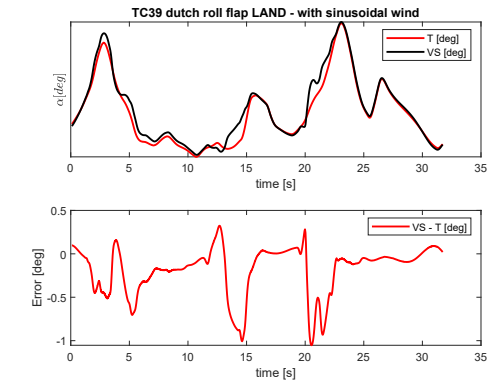
(h) TC18 bank hold flap LAND 20°



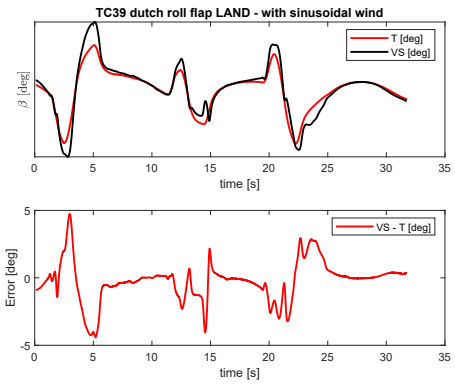
(i) TC33 SHSS flap LAND 15°



(j) TC33 SHSS flap LAND 15°



(k) TC39 dutch roll flap LAND



(l) TC39 dutch roll flap LAND

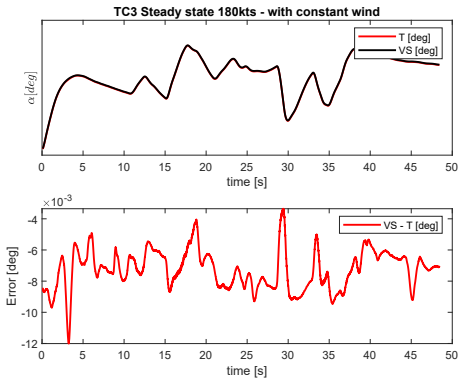
Figure 65: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}, \beta_{in}, \delta_a, \delta_r\right]$, 15 neurons, AoA as delta, training set: no wind maneuvers, testing set: sinusoidal wind maneuvers

From Fig. 65 emerges as the NN's estimation generally follows successfully the real aerodynamic angle's dynamics. On the other hand, in some maneuvers the error's absolute value is quite large as the VS tends to under/overestimate its solution in proximity of AOA/AOS's peaks (see Fig.s 65(l), 65(j), 65(d)). This behaviour suggests the necessity to deepen the analysis and provide new maneuvers aimed to make the NN learn more about transients.

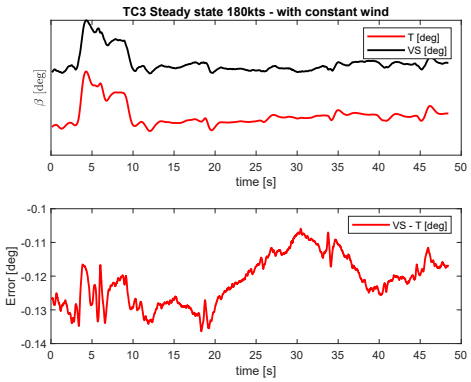
11.5 Without wind, influence of center of mass

The Balance point (Centre of Gravity - CG) position is very important during flight because of its effect on the aircraft's stability and performance. As the flight progress, the CG position moves when the distribution of the load varies (e.g. by passengers moving about or by transferring fuel from one tank to another) and as the weight changes by fuel burning off or by parachutists leaping out. The ideal location of the center of gravity (CG) is very carefully determined by the designers and it must remain within carefully defined limits throughout all stages of the flight. The pilot in command of the aircraft has the responsibility on every flight to know the maximum allowable gross weight of the aircraft and its CG limits. This allows the pilot to determine, on the preflight inspection, that the aircraft is loaded in such a way that the CG is within the allowable limits. Weight and balance is one of the most important factors affecting safety of flight and an overweight aircraft, or one whose center of gravity is outside the allowable limits, is not airworthy.
In this subsection, the influence of CG position will be explored testing $\text{NN}_{pat,nw}$ (trained with no-wind maneuvers and CG in intermediate position) with the same type of maneuvers included in the training set but now carried out with constant wind and CG in AFT position.

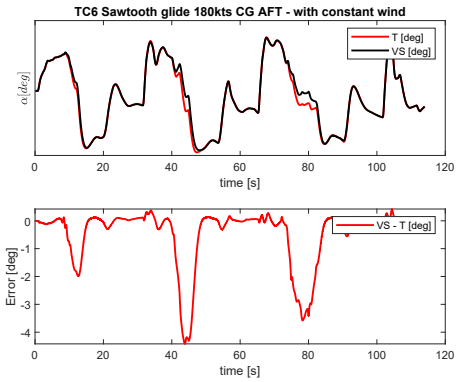
Results are reported in Fig. 66.



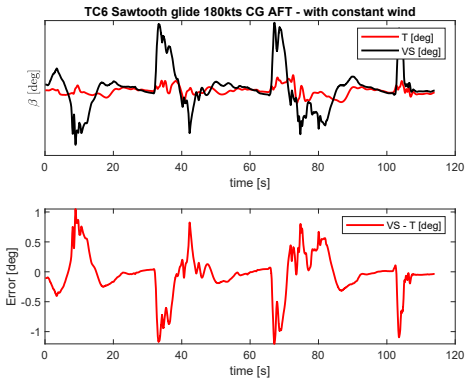
(a) TC3 Steady state 180kts



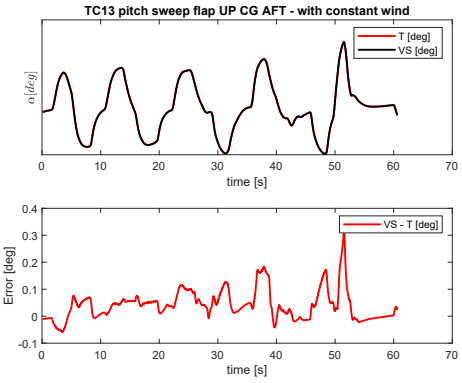
(b) TC3 Steady state 180kts



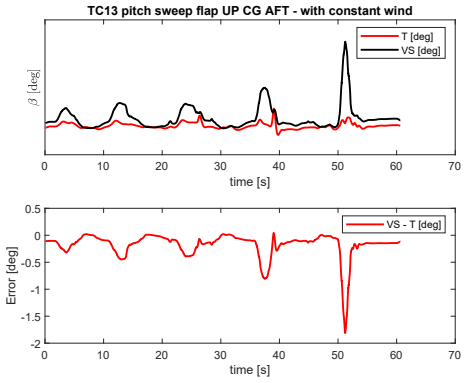
(c) TC6 Sawtooth glide 180kts



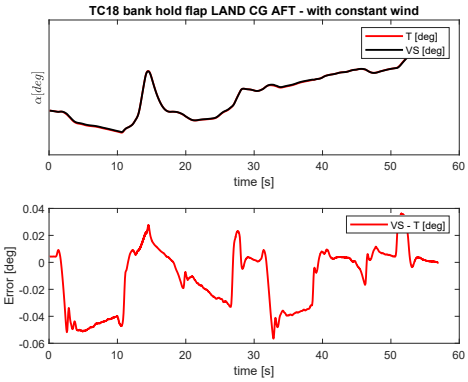
(d) TC6 Sawtooth glide 180kts



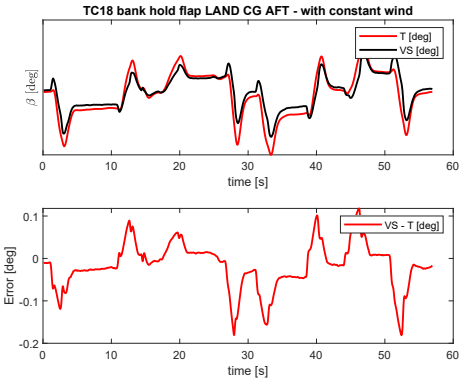
(e) TC13 pitch sweep flap UP



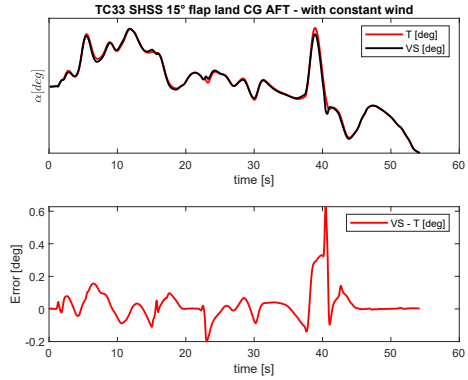
(f) TC13 pitch sweep flap UP



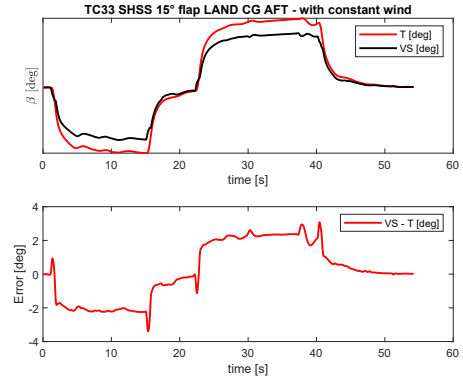
(g) TC18 bank hold flap LAND 20°



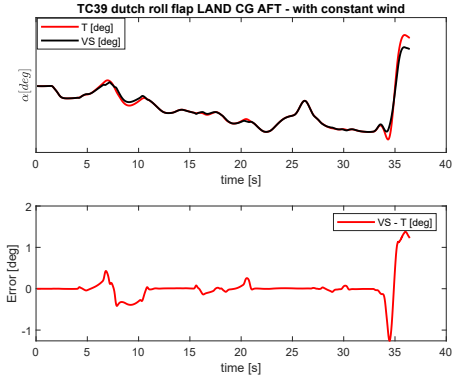
(h) TC18 bank hold flap LAND 20°



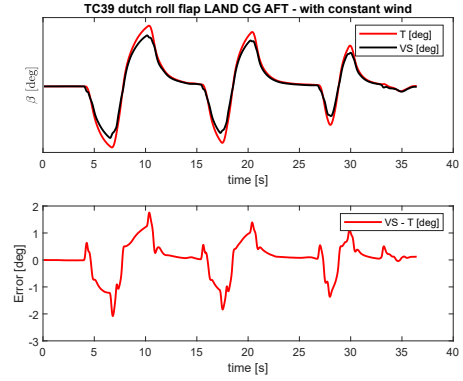
(i) TC33 SHSS flap LAND 15°



(j) TC33 SHSS flap LAND 15°



(k) TC39 dutch roll flap LAND



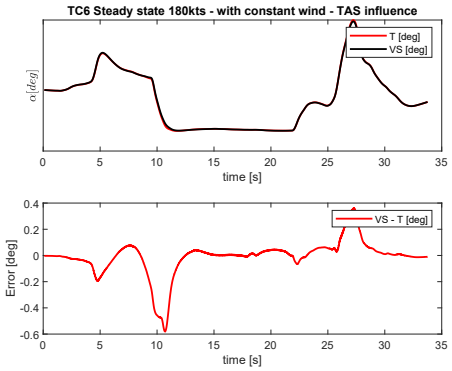
(l) TC39 dutch roll flap LAND

Figure 66: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}, \beta_{in}, \delta_a, \delta_r\right]$, 15 neurons, AoA as delta, training set: no wind maneuvers, testing set: with constant wind and AFT CG maneuvers

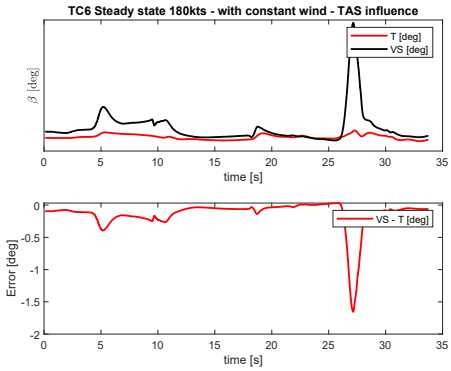
The AoA/AoS's dynamics is generally well captured by NN's estimation but, as in some maneuvers (see Figs. 66(j)-66(l)) the error's absolute value reaches peaks of 3.8°, it can be generally stated that the CG position has a certain influence in VS's performance and suggest to deepen the relative analysis.

11.6 Sensitivity analysis

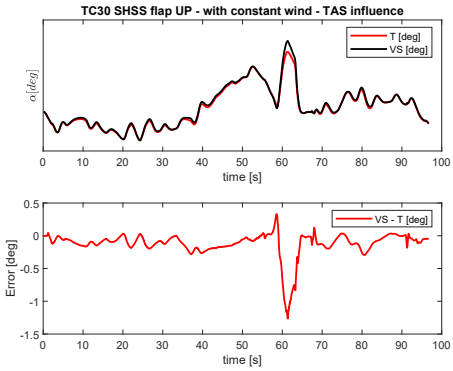
As already exposed in Sect. 6, a Neural Network has one weight for every input-to-neuron connection between the layers awn bias constant for each neuron in the hidden layer. In general, several techniques exist to and its osses if weights related to a certain type of input are low enough that the NN's output can basically be considered independent on that input signal.
Through new tests on $NN_{pat,nw}$, in this subsection will be basically evaluated the effects of introducing an error of 20% on each input at a time on the AoA/AoS estimation.



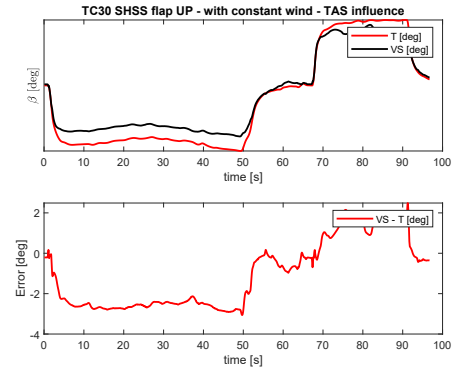
(a) TC6 Sawtooth glide 180kts



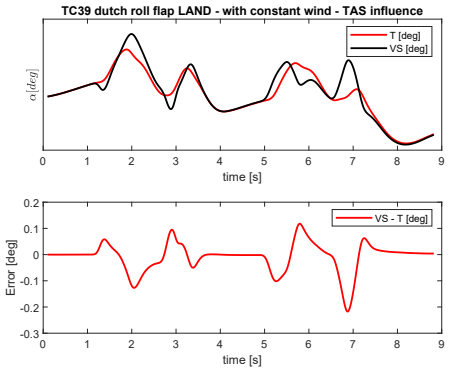
(b) TC6 Sawtooth glide 180kts



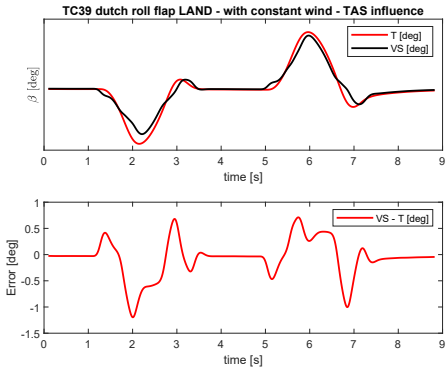
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°



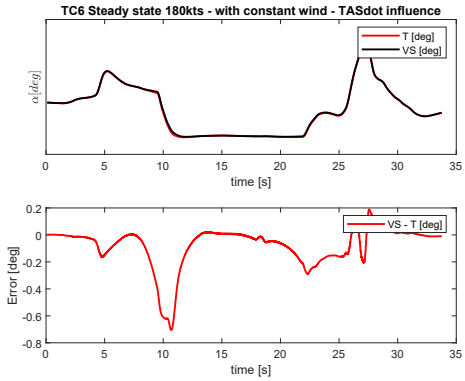
(e) TC39 dutch roll flap LAND



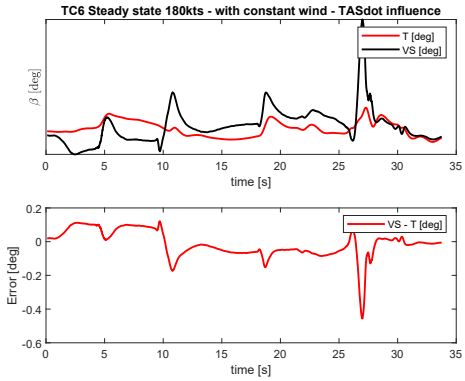
(f) TC39 dutch roll flap LAND

Figure 67: $\left[TAS, T\dot{A}S, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}, \beta_{in}, \delta_a, \delta_r\right]$, 15 neurons, AoA as delta, training set: no wind maneuvers, testing set: with constant wind and intermediate CG maneuvers, influence of a 20% error on TAS

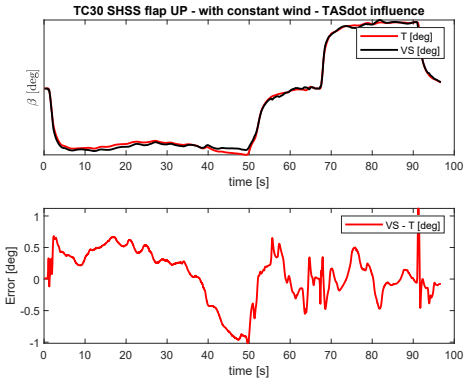
As shown in Fig. 67(d), a 20% error on TAS results in an maximum error that grows up to 330% (from 1.5° to 4°). From this consideration it can be generally stated that the TAS has a key role in AoS estimation and cannot be removed from the input vector. It is moreover important to remark as the error on TAS essentially reflects on AoS estimation whereas the AoA seems to be not to much affected.



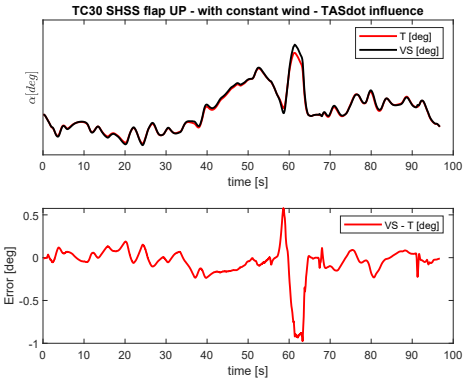
(a) TC6 Sawtooth glide 180kts



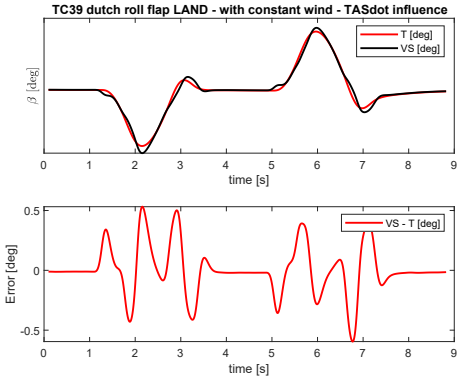
(b) TC6 Sawtooth glide 180kts



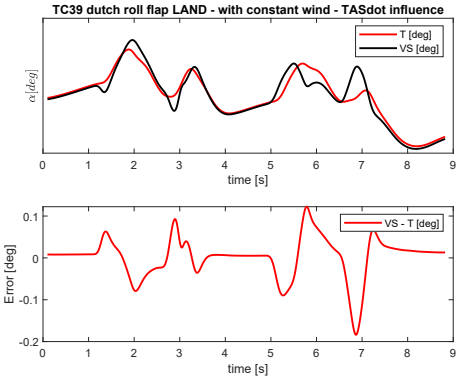
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°



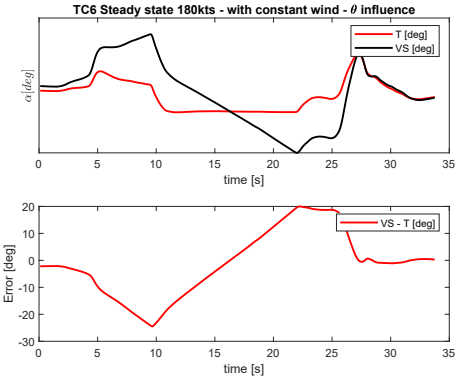
(e) TC39 dutch roll flap LAND



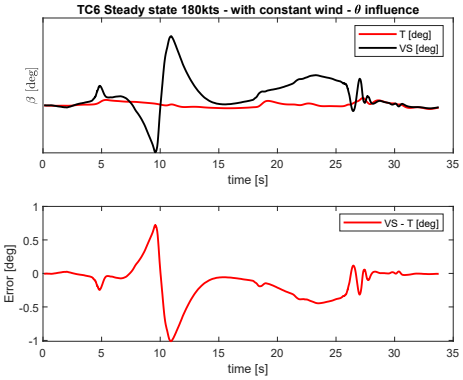
(f) TC39 dutch roll flap LAND

Figure 68: $\left[TAS, T\dot{A}S, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}, \beta_{in}, \delta_a, \delta_r\right]$, 15 neurons, AoA as delta, training set: no wind maneuvers, testing set: with constant wind and intermediate CG maneuvers, influence of a 20% error on $T\dot{A}S$

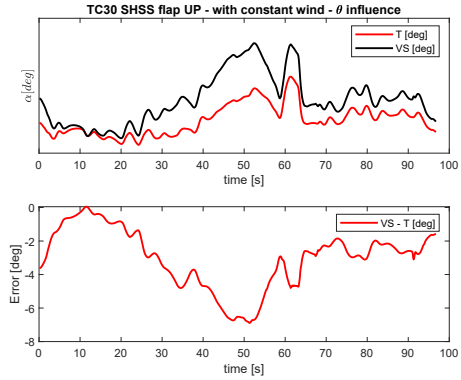
The introduction of a 20% uncertainty in $T\dot{A}S$ input signal does not lead to an increase in AoA/AoS estimation's error: this result suggest that the presence of TASdot in the input vector is unnecessary and therefore can be neglected.



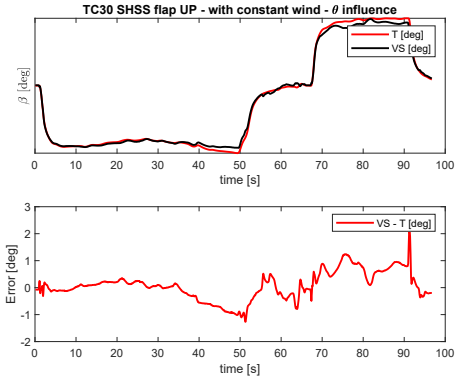
(a) TC6 Sawtooth glide 180kts



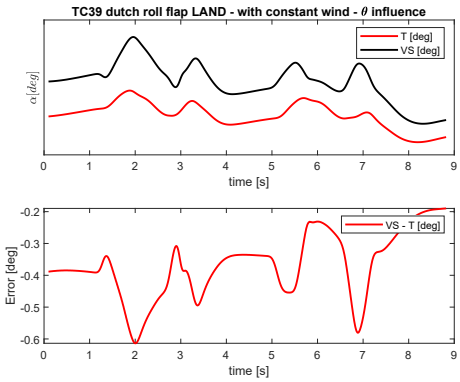
(b) TC6 Sawtooth glide 180kts



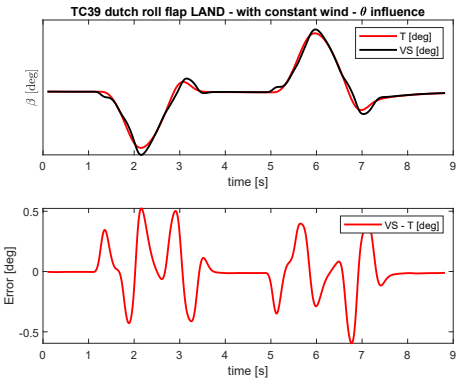
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°



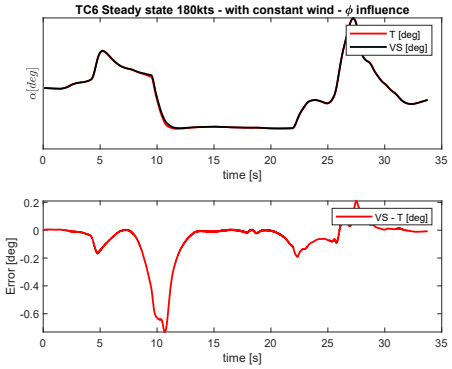
(e) TC39 dutch roll flap LAND



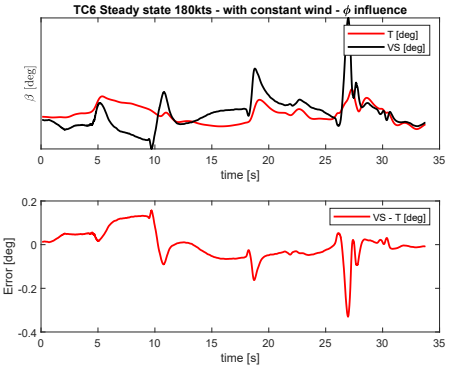
(f) TC39 dutch roll flap LAND

Figure 69: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}, \beta_{in}, \delta_a, \delta_r\right]$, 15 neurons, AoA as delta, training set: no wind maneuvers, testing set: with constant wind and intermediate CG maneuvers, influence of a 20% error on θ

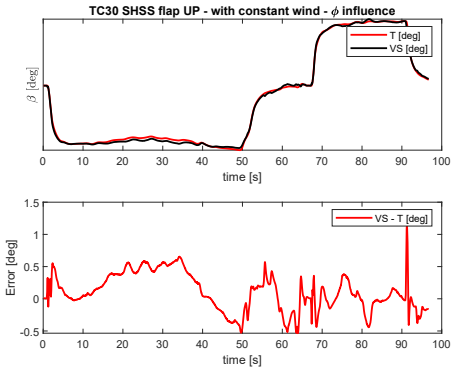
A 20% error in θ input signal results in a very large error both in AoA both in AoS estimation: the presence of pitch angle signal in the input vector is thus necessary for the NN's accuracy.



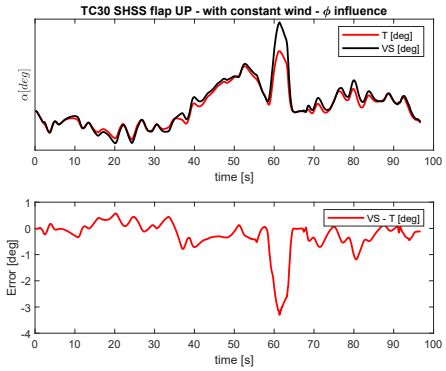
(a) TC6 Sawtooth glide 180kts



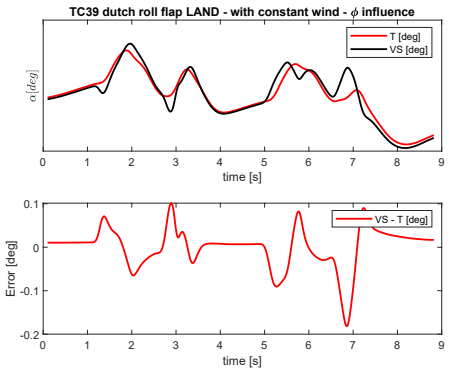
(b) TC6 Sawtooth glide 180kts



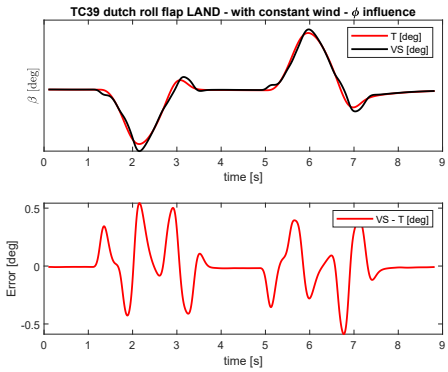
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°



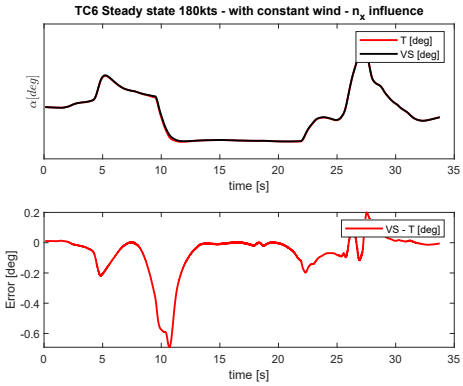
(e) TC39 dutch roll flap LAND



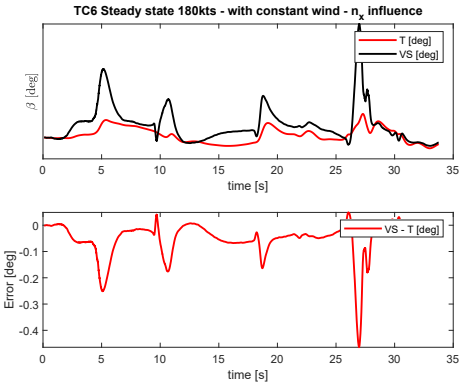
(f) TC39 dutch roll flap LAND

Figure 70: $\left[TAS, T\dot{A}S, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}, \beta_{in}, \delta_a, \delta_r\right]$, 15 neurons, AoA as delta, training set: no wind maneuvers, testing set: with constant wind and intermediate CG maneuvers, influence of a 20% error on ϕ

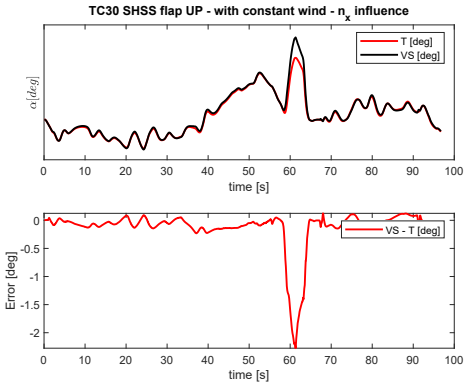
The most evident consequence of the introduction of a 20% error in roll angle is highlighted in Fig. 70(c): the error in AoA became large and the NN loses accuracy. It can be generally stated that the roll angle signal can not be taken off from the input vector.



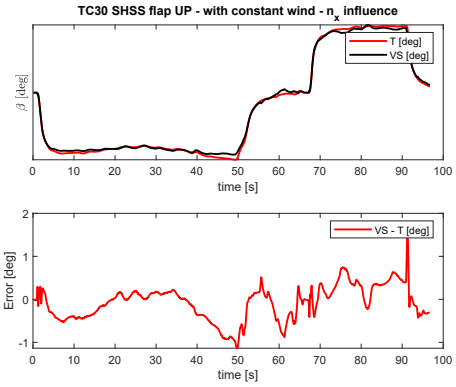
(a) TC6 Sawtooth glide 180kts



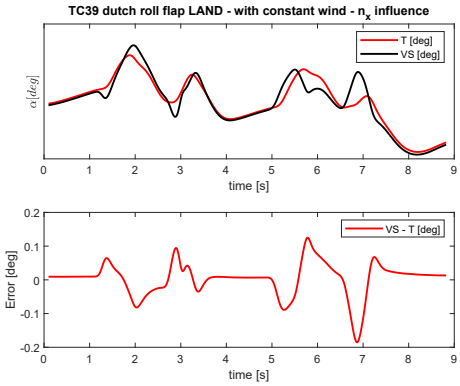
(b) TC6 Sawtooth glide 180kts



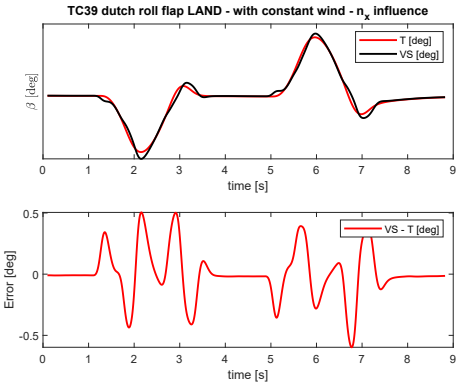
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°

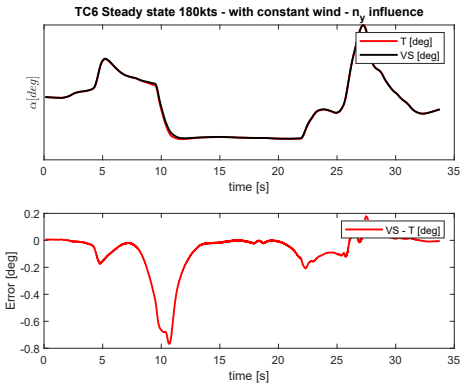


(e) TC39 dutch roll flap LAND

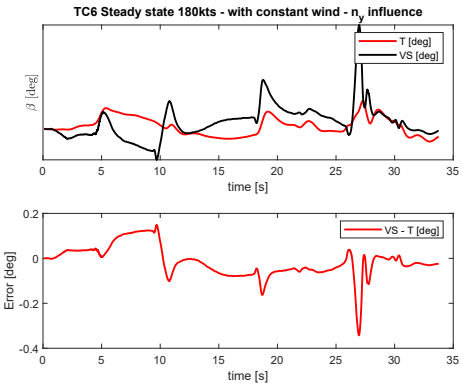


(f) TC39 dutch roll flap LAND

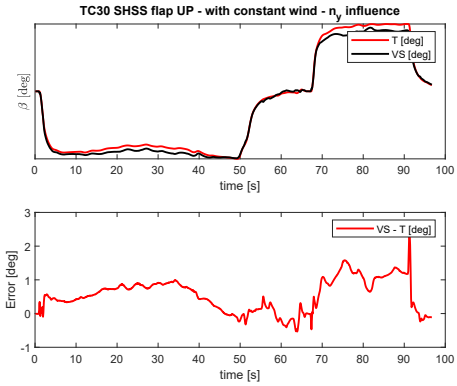
Figure 71: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}, \beta_{in}, \delta_a, \delta_r\right]$, 15 neurons, AoA as delta, training set: no wind maneuvers, testing set: with constant wind and intermediate CG maneuvers, influence of a 20% error on n_x



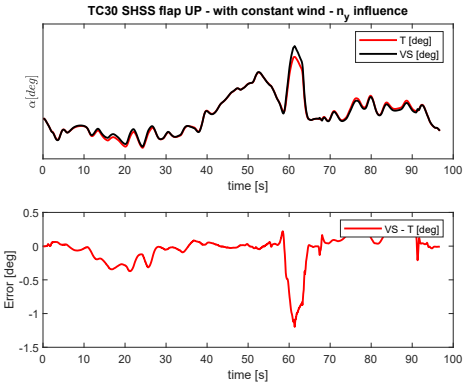
(a) TC6 Sawtooth glide 180kts



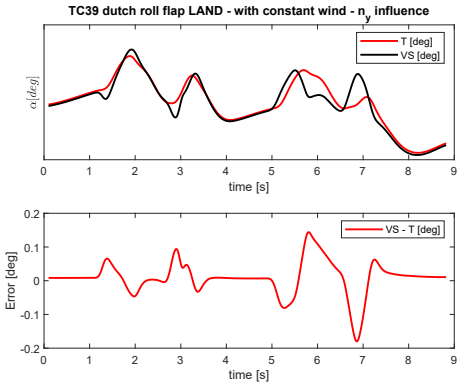
(b) TC6 Sawtooth glide 180kts



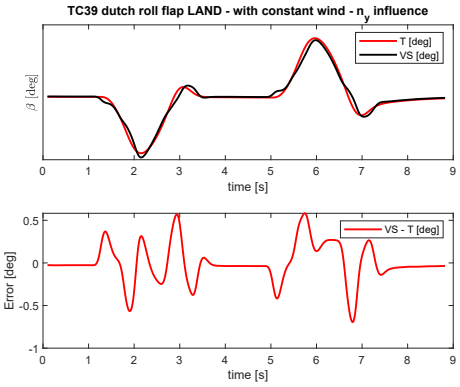
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°

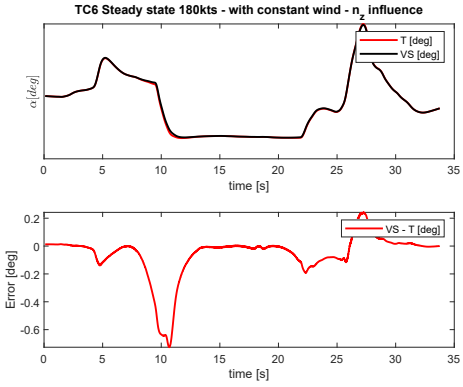


(e) TC39 dutch roll flap LAND

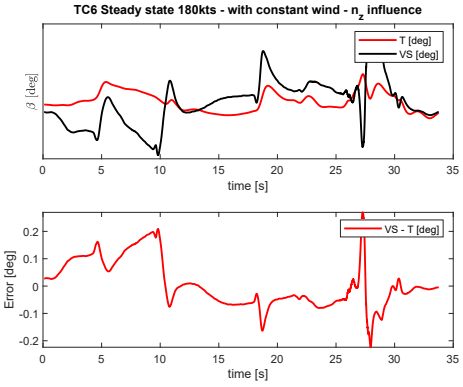


(f) TC39 dutch roll flap LAND

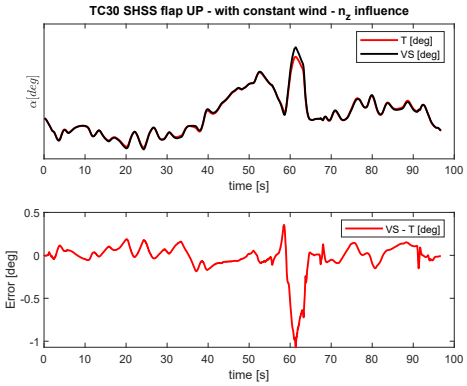
Figure 72: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}, \beta_{in}, \delta_a, \delta_r\right]$, 15 neurons, AoA as delta, training set: no wind maneuvers, testing set: with constant wind and intermediate CG maneuvers, influence of a 20% error on n_y



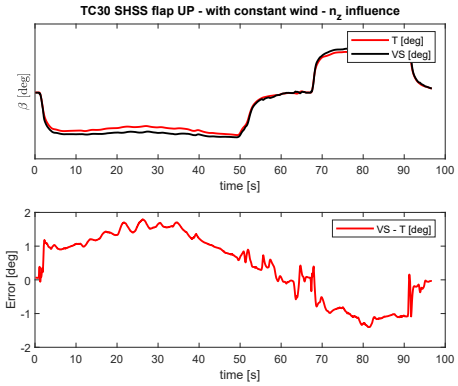
(a) TC6 Sawtooth glide 180kts



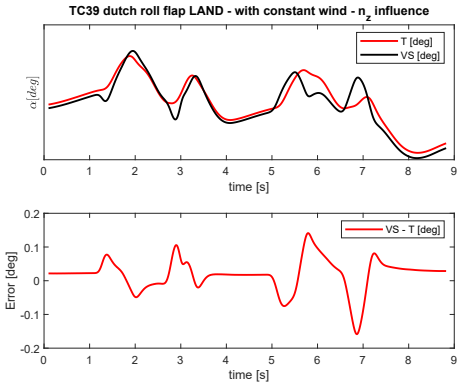
(b) TC6 Sawtooth glide 180kts



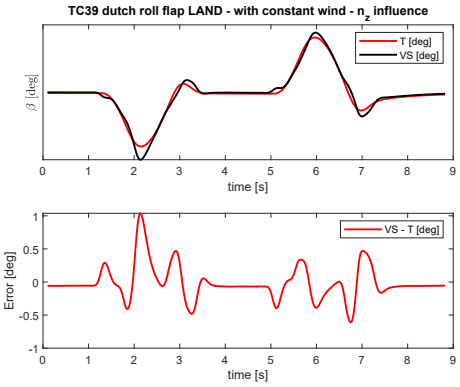
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°



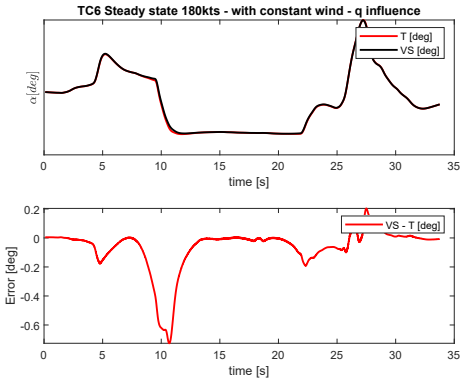
(e) TC39 dutch roll flap LAND



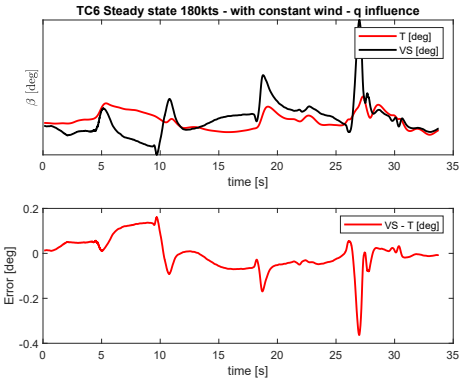
(f) TC39 dutch roll flap LAND

Figure 73: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}, \beta_{in}, \delta_a, \delta_r\right]$, 15 neurons, AoA as delta, training set: no wind maneuvers, testing set: with constant wind and intermediate CG maneuvers, influence of a 20% error on n_z

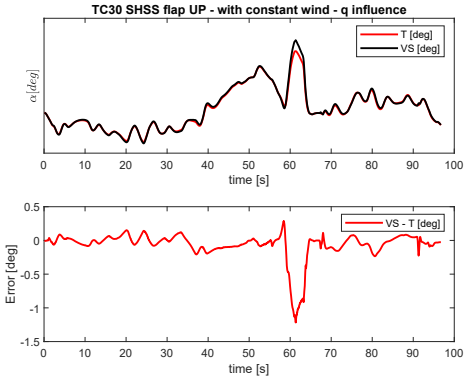
Uncertainties in n_x, n_y, n_z result in a degradation of solution (see Fig. 71(c)) therefore the inertial accelerations along each body axis are relevant for the NN's accuracy.



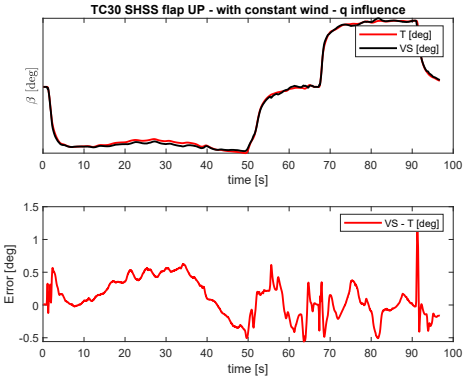
(a) TC6 Sawtooth glide 180kts



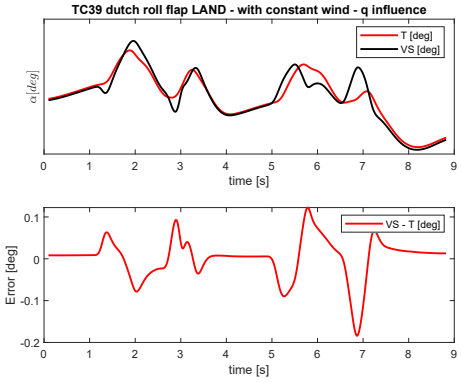
(b) TC6 Sawtooth glide 180kts



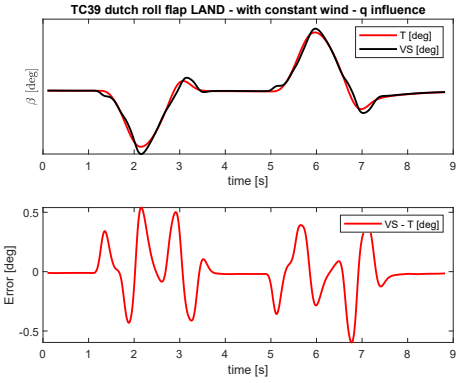
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°

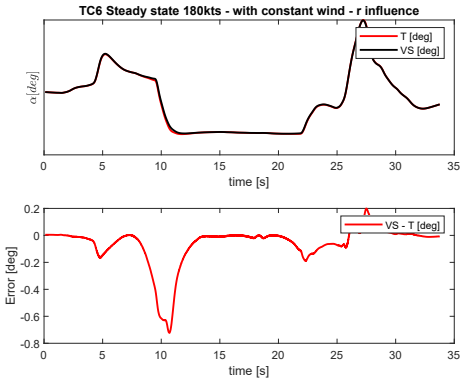


(e) TC39 dutch roll flap LAND

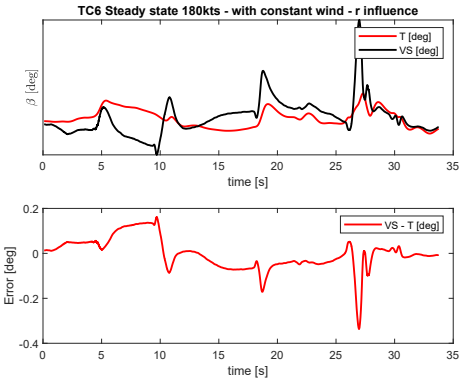


(f) TC39 dutch roll flap LAND

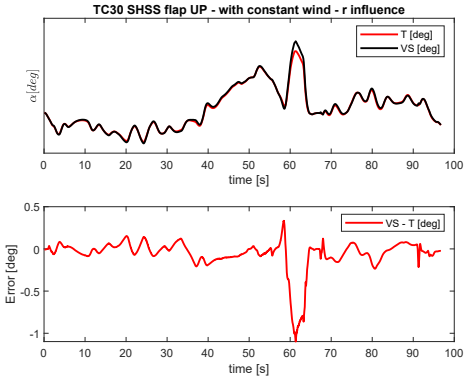
Figure 74: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}, \beta_{in}, \delta_a, \delta_r\right]$, 15 neurons, AoA as delta, training set: no wind maneuvers, testing set: with constant wind and intermediate CG maneuvers, influence of a 20% error on q



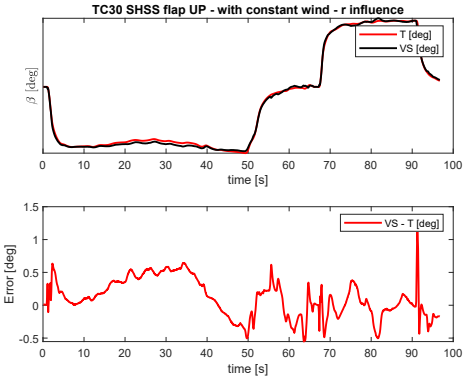
(a) TC6 Sawtooth glide 180kts



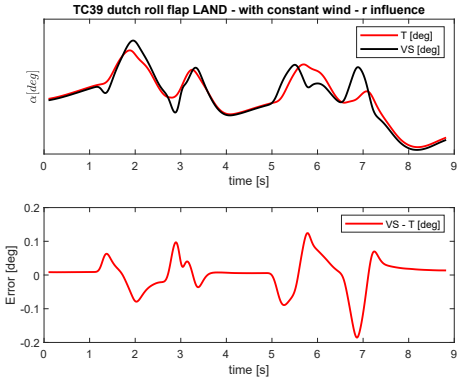
(b) TC6 Sawtooth glide 180kts



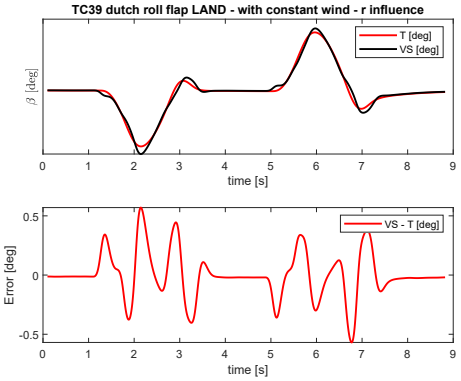
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°

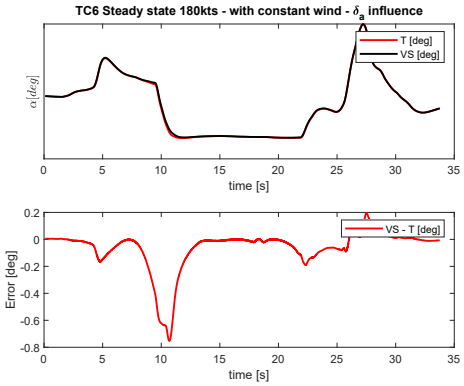


(e) TC39 dutch roll flap LAND

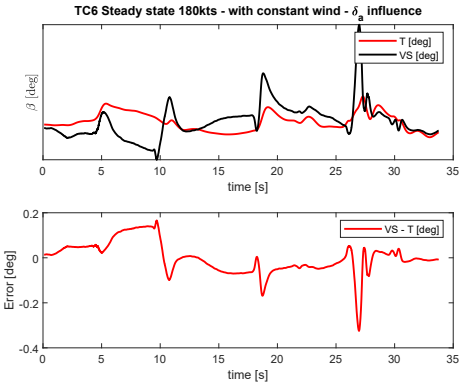


(f) TC39 dutch roll flap LAND

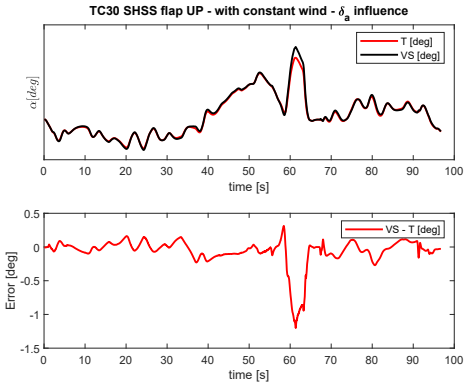
Figure 75: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}, \beta_{in}, \delta_a, \delta_r\right]$, 15 neurons, AoA as delta, training set: no wind maneuvers, testing set: with constant wind and intermediate CG maneuvers, influence of a 20% error on r



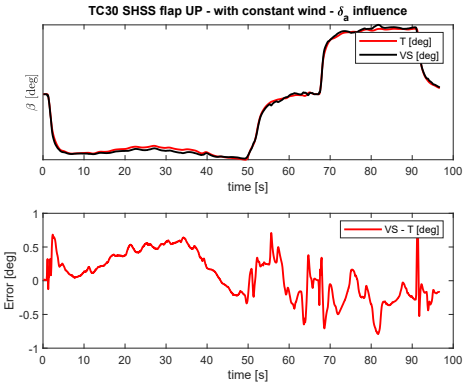
(a) TC6 Sawtooth glide 180kts



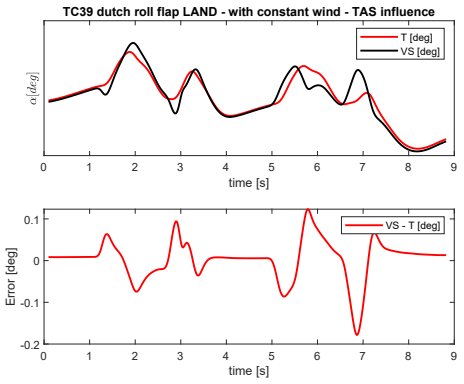
(b) TC6 Sawtooth glide 180kts



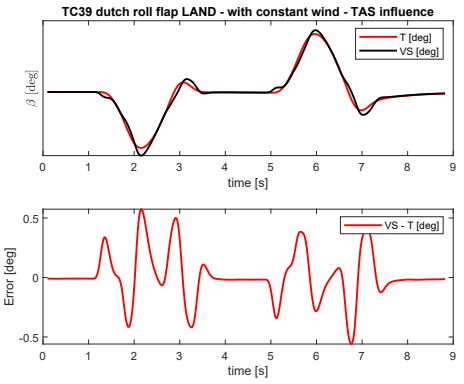
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°



(e) TC39 dutch roll flap LAND

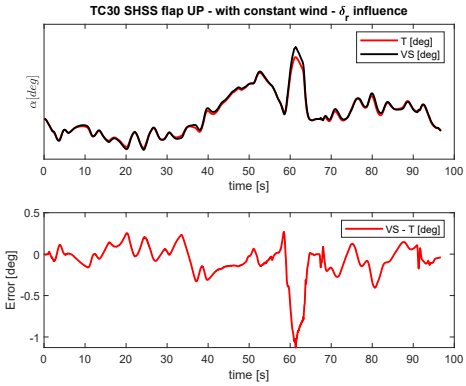


(f) TC39 dutch roll flap LAND

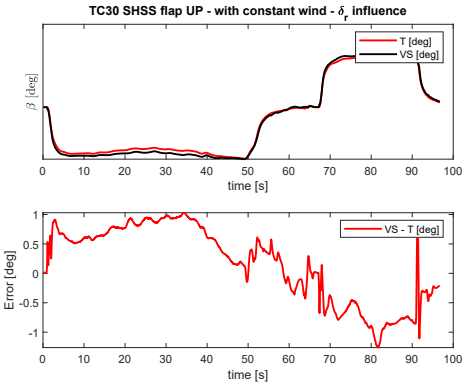
Figure 76: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}, \beta_{in}, \delta_a, \delta_r\right]$, 15 neurons, AoA as delta, training set: no wind maneuvers, testing set: with constant wind and intermediate CG maneuvers, influence of a 20% error on δ_a

From Figs.74,75 it seems that all angular velocities do not play a really important role

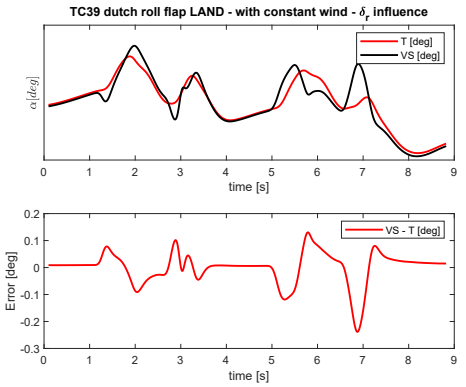
for the NN training and thus could be removed streamlining the vector of inputs.



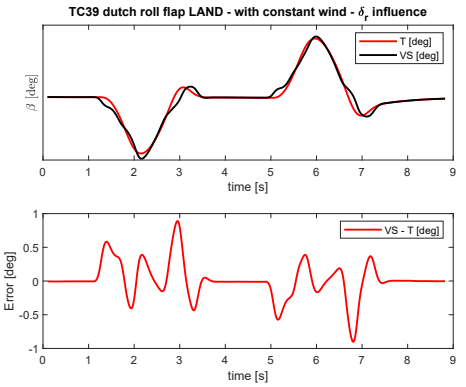
(a) TC30 SHSS flap UP 15°



(b) TC30 SHSS flap UP 15°



(c) TC39 dutch roll flap LAND



(d) TC39 dutch roll flap LAND

Figure 77: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, q, r, \alpha_{in}, \beta_{in}, \delta_a, \delta_r\right]$, 15 neurons, AoA as delta, training set: no wind maneuvers, testing set: with constant wind and intermediate CG maneuvers, influence of a 20% error on δ_r

The presence of α_{in} and β_{in} seems to well replace δ_a and δ_r for AoA/AoS determination and thus could be removed from the input vector.
Results obtained through the sensitivy analysis on $\text{NN}_{pat,nw}$ are highlighted in Tab. 2.

Input Parameter	Influence on NN's outputs
TAS	high
$\dot{TAS}$	very low
θ	very high
ϕ	very high
n_x	high
n_y	high
n_z	high
p	low
q	low
r	low
δ_a	very low
δ_r	very low

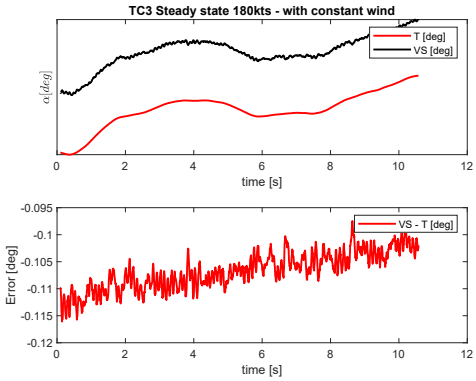
Table 2: Results of sensitivity analysis on $\text{NN}_{pat,nw}$

12 AoA/AoS estimation computing their absolute value through NN

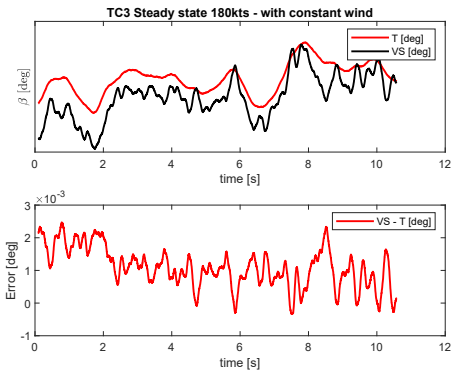
12.1 With constant wind

According to the patented procedure illustrated so far, aerodynamic angles are calculated starting from a linear initial estimation of α_{in} (Eq.67)/β_{in} (Eq.68) given by mechanics of flight equations and then, an additional contribute $\Delta\alpha/\Delta\beta$ is calculated by a MLP such that AoA/AoS's final NN estimation can be expressed as $\alpha = \alpha_{in} + \Delta\alpha$ and $\beta = \beta_{in} + \Delta\beta$. On the other hand Eq.67 and Eq.68 are no longer valid in presence of external wind hence, if with-wind maneuvers are introduced in the training set, both the angle of attack both the sideslip angle must be computed by the NN only through their absolute value $(\alpha_{NN}, \beta_{NN})$. In order to accomplish this procedure is necessary to provide the NN with new input signals which do the same work of α_{in} and β_{in}. In particular, a first attempt is carried out with the input vector which contains all commands, all angular velocities and all inertial accelerations then, a sensitivity analysis is made to asses which of these input signals could be neglected.

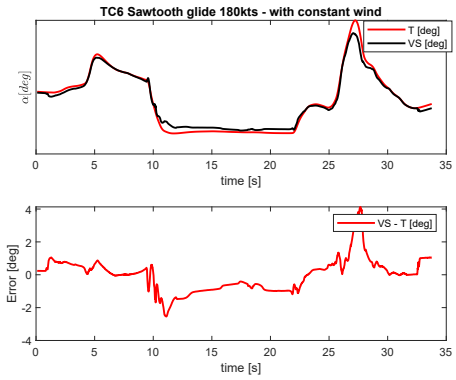
In Fig. 81 are reported tests on with-wind maneuvers where the training set contains only without-wind maneuvers and aerodynamic angles are computed without any preliminary linear estimation of themselves (resulting in a Neural Network that from now on will be referred as $\mathrm{NN}_{ass,nw}$). Besides, in Fig. 79 is shown an attempt where the training set includes a balanced set of both withouth both with wind maneuvers (resulting in a Neural Network labelled as $\mathrm{NN}_{ass,hyb}$).



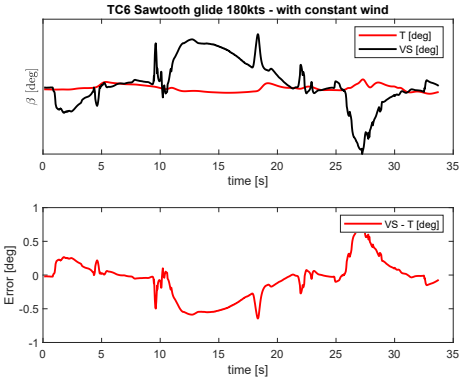
(a) TC3 Steady state 180kts



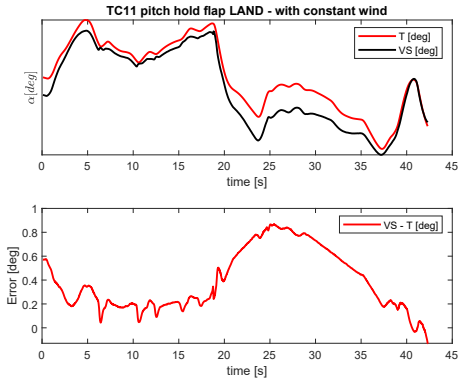
(b) TC3 Steady state 180kts



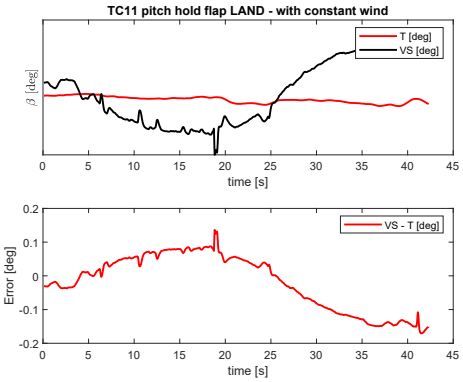
(c) TC6 Sawtooth glide 180kts



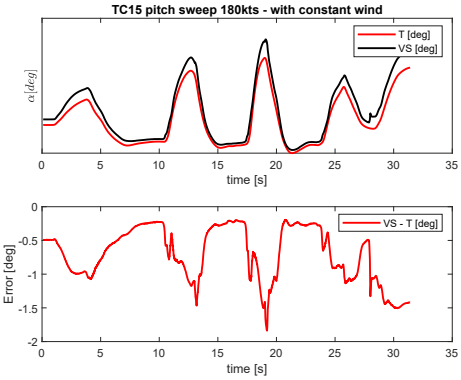
(d) TC6 Sawtooth glide 180kts



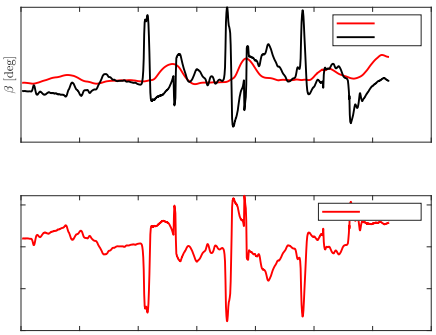
(e) TC11 pitch hold 150kts



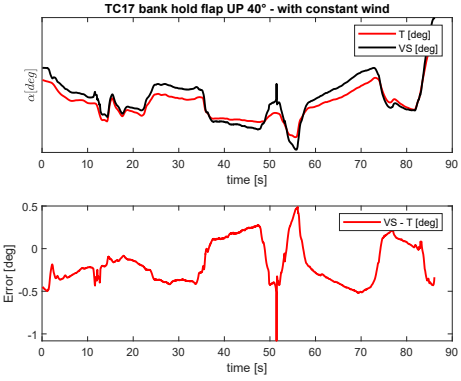
(f) TC11 pitch hold 150kts



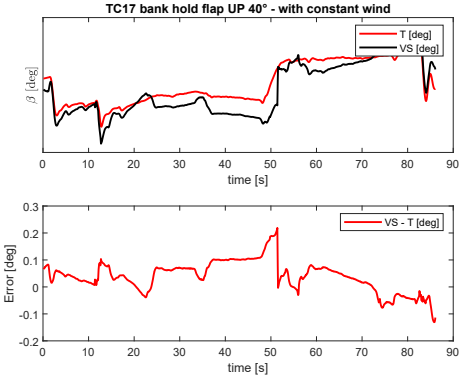
(g) TC15 pitch sweep 180kts



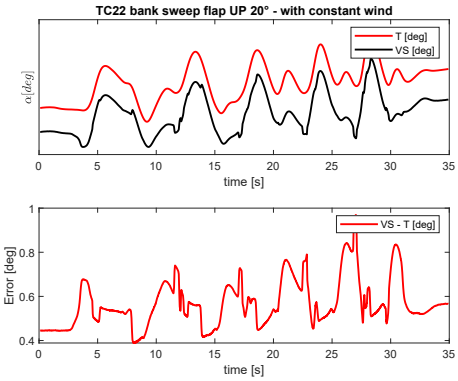
(h) TC15 pitch sweep 180kts



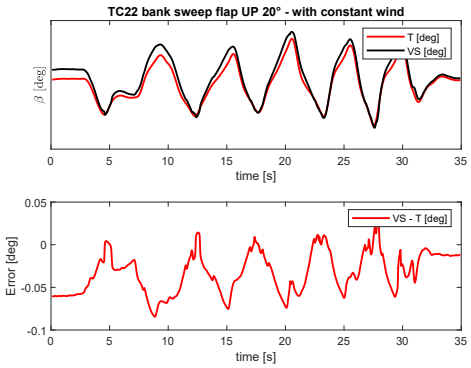
(i) TC17 bank hold flap UP 40°



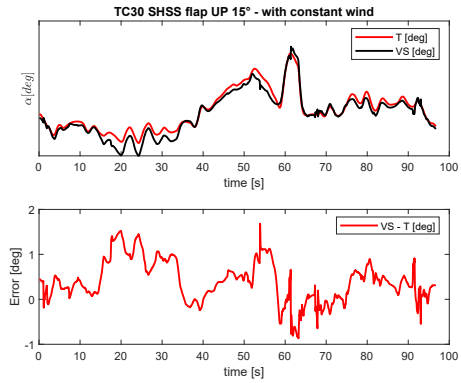
(j) TC17 bank hold flap UP 40°



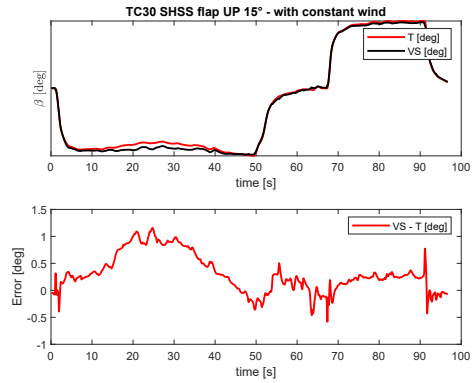
(k) TC22 bank sweep flap UP 20°



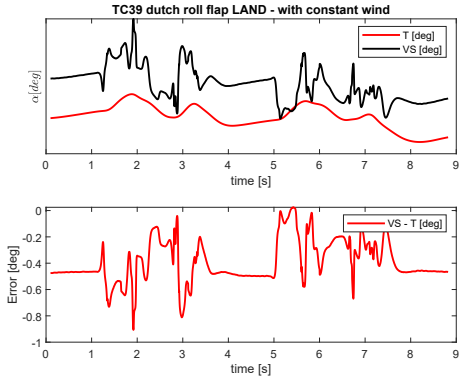
(l) TC22 bank sweep flap UP 20°



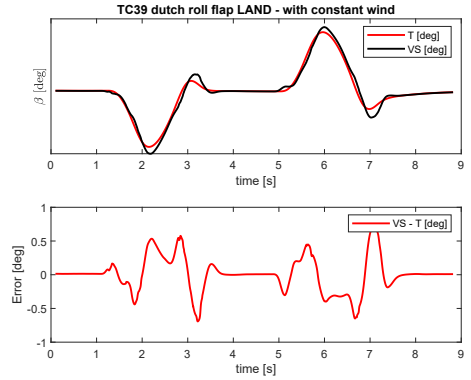
(m) TC30 SHSS flap UP 15°



(n) TC30 SHSS flap UP 15°

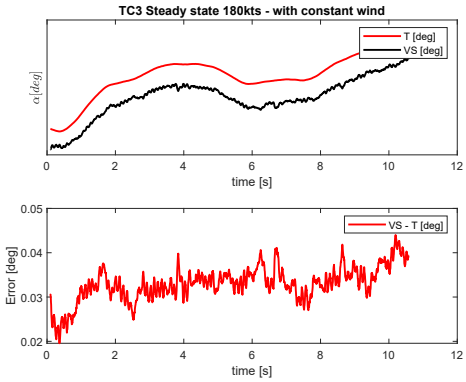


(o) TC39 dutch roll flap LAND

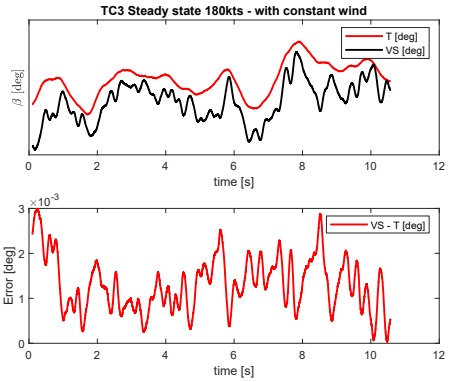


(p) TC39 dutch roll flap LAND

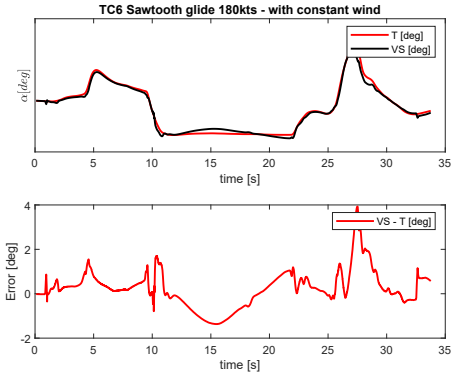
Figure 78: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, p, q, r, \delta_e, \delta_f, \delta_{f,LE}, \delta_a, \delta_r, \frac{1}{2}\left(\delta_{th,RH} + \delta_{th,LH}\right)\right]$, 30 neurons, AoA/AoS as absolute value, training set: no wind maneuvers, testing set: with constant wind maneuvers



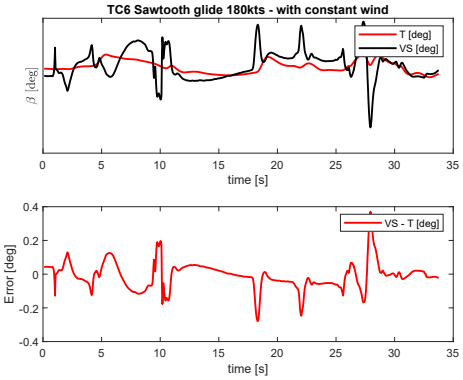
(a) TC3 Steady state 180kts



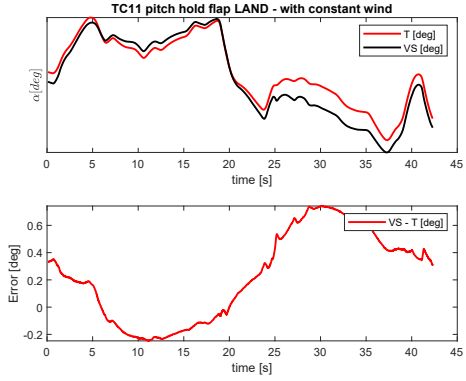
(b) TC3 Steady state 180kts



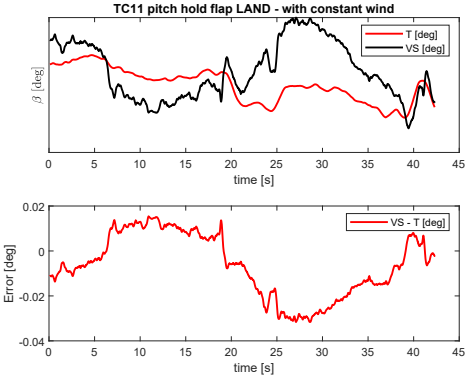
(c) TC6 Sawtooth glide 180kts



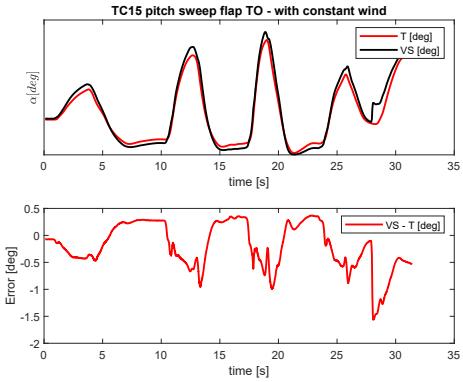
(d) TC6 Sawtooth glide 180kts



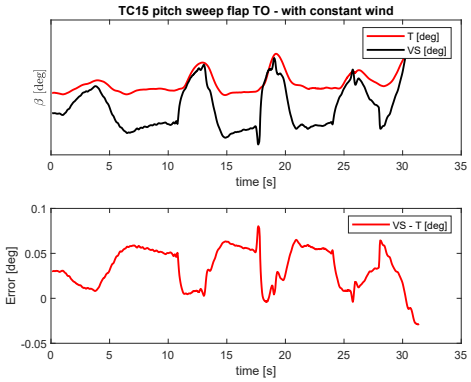
(e) TC11 pitch hold 150kts



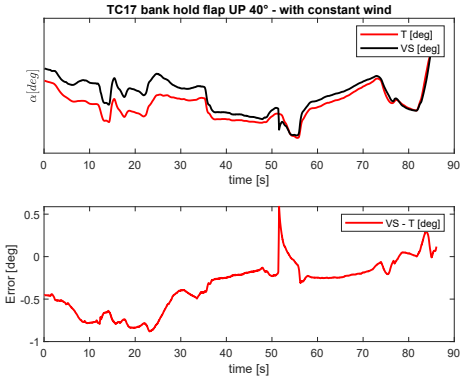
(f) TC11 pitch hold 150kts



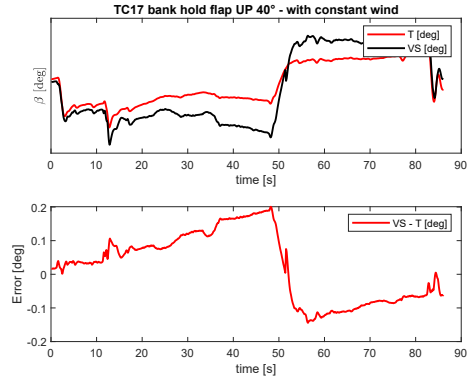
(g) TC15 pitch sweep 180kts



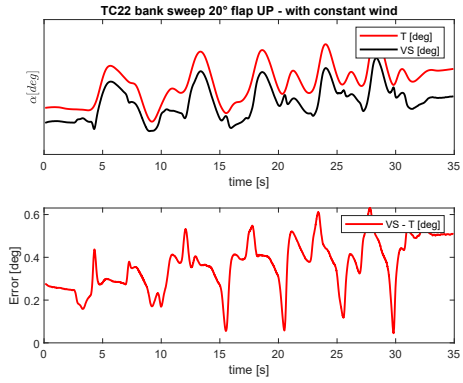
(h) TC15 pitch sweep 180kts



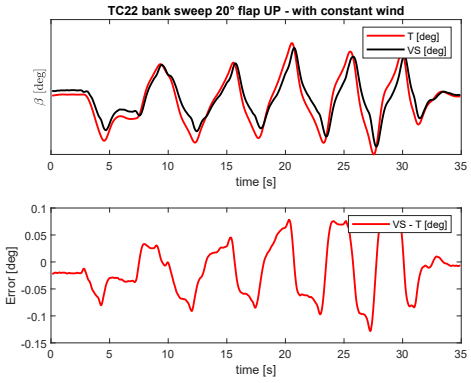
(i) TC17 bank hold flap UP 40°



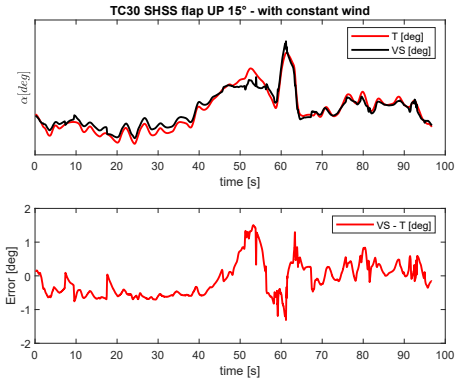
(j) TC17 bank hold flap UP 40°



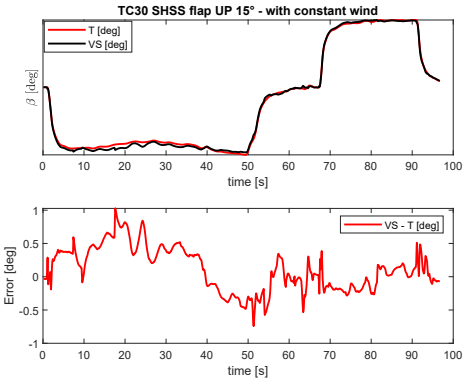
(k) TC22 bank sweep flap UP 20°



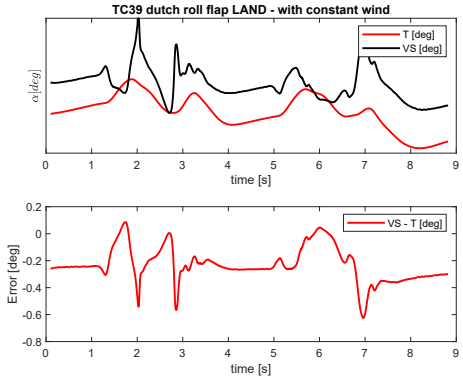
(l) TC22 bank sweep flap UP 20°



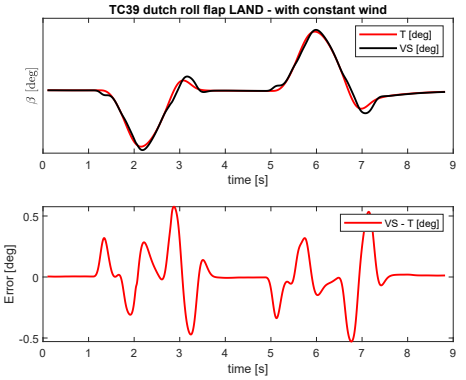
(m) TC30 SHSS flap UP 15°



(n) TC30 SHSS flap UP 15°



(o) TC39 dutch roll flap LAND



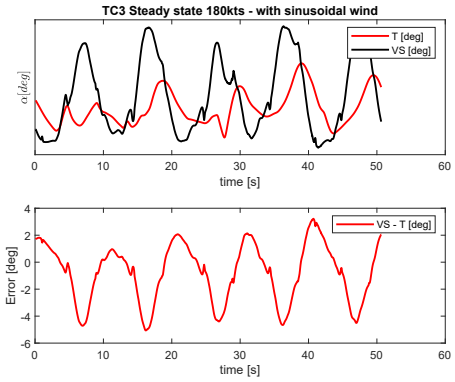
(p) TC39 dutch roll flap LAND

Figure 79: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, p, q, r, \delta_e, \delta_f, \delta_{f,LE}, \delta_a, \delta_r, \frac{1}{2}\left(\delta_{th,RH} + \delta_{th,LH}\right)\right]$, 30 neurons, AoA/AoS as absolute value, training set: both no wind both with wind maneuvers, testing set: with constant wind maneuvers

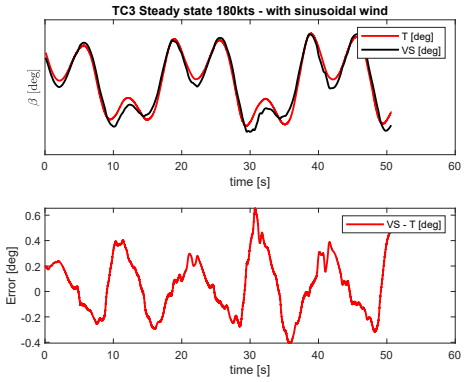
Even if in those maneuvers where a costant gust was imposed (see Fig. 78(e) and Fig. 79(e)) the offset between the real and the estimated solution for AoA decreases considerably, NN's results seems to be degraded with respect to that obtained in Sect. 11.3 reaching a peak of 4° error in 78(c): it can be generally stated that the patented procedure represents a better solution in case of constant wind maneuvers whereas the estimation of AoA/AoS through their absolute value works better in that maneuvers where there are gusts as well. One other results that comes out is that an hybrid training set that contains both with constant wind both without wind maneuvers does not lead to remarkable improvements with respect the solution obtained training the NN only on without-wind maneuvers.

12.2 With sinusoidal wind

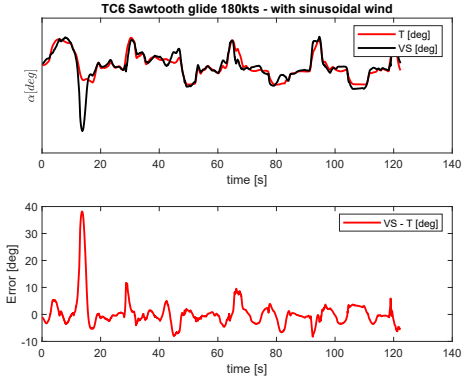
It is now interesting to evaluate the behaviour of the virtual sensor based on $\text{NN}_{ass,hyb}$ in presence of sinusoidal wind.
From Fig. 80, immediatly comes out as the α and β estimations result to be very degraded with respect to those obtained in Sect. 11.4: besides some maneuvers where the error's absolute value reaches peaks very large and is absolutly unacceptable (see Fig.s 80(c), 80(k) where the error even reaches 40° and 10° respectively) it always results larger than that obtained using the patented-based NN.
It can generally be stated that the virtual sensor, in presence of external sinusoidal wind, does not work efficiently with none of the two Neural Networks developed (and in general all other situation with variable external wind) and therefore, its feasability under variable wind conditions should be object of further studies.



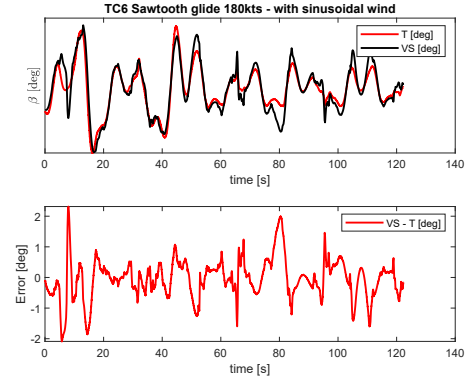
(a) TC3 Steady state 180kts



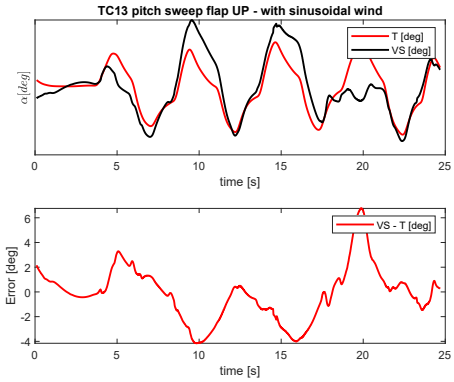
(b) TC3 Steady state 180kts



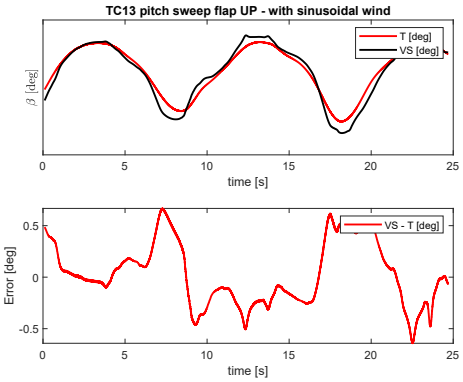
(c) TC6 Sawtooth glide 180kts



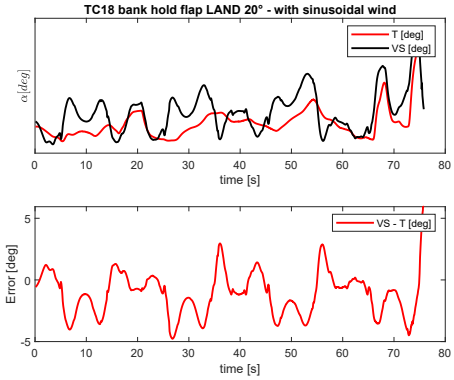
(d) TC6 Sawtooth glide 180kts



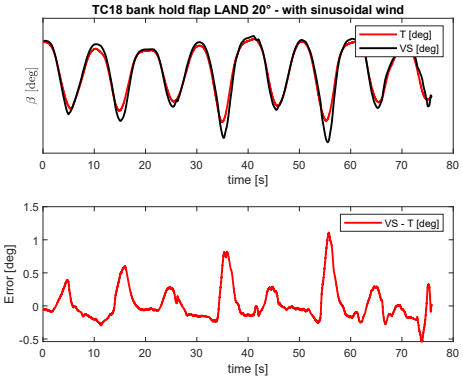
(e) TC13 pitch sweep flap UP



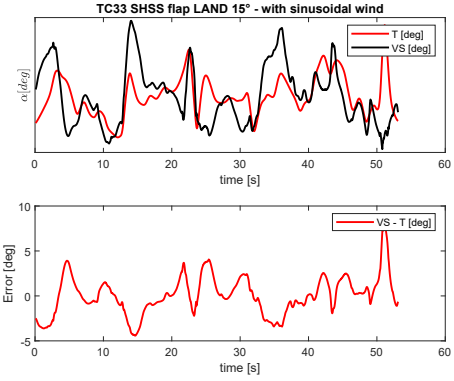
(f) TC13 pitch sweep flap UP



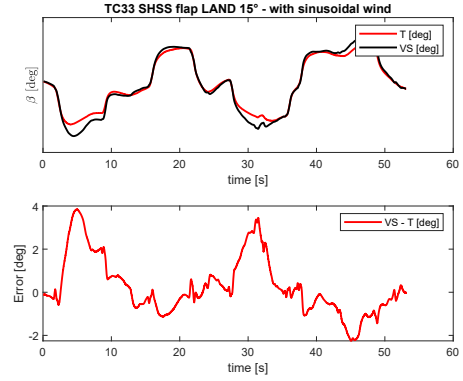
(g) TC18 bank hold flap LAND 20°



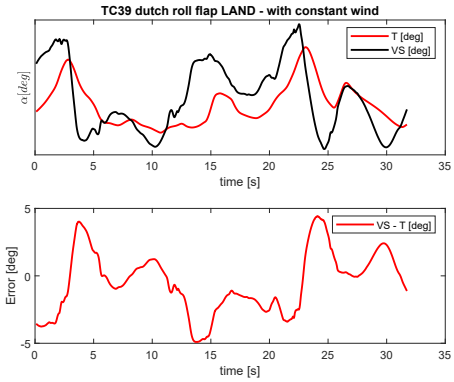
(h) TC18 bank hold flap LAND 20°



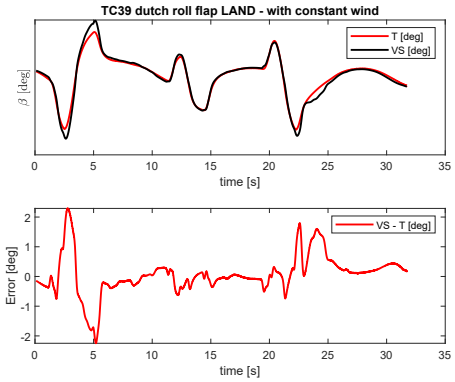
(i) TC33 SHSS flap LAND 15°



(j) TC33 SHSS flap LAND 15°



(k) TC39 dutch roll flap LAND

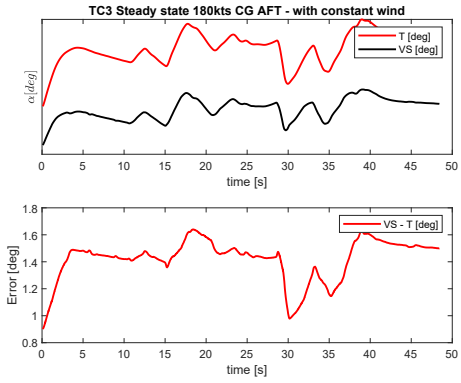


(l) TC39 dutch roll flap LAND

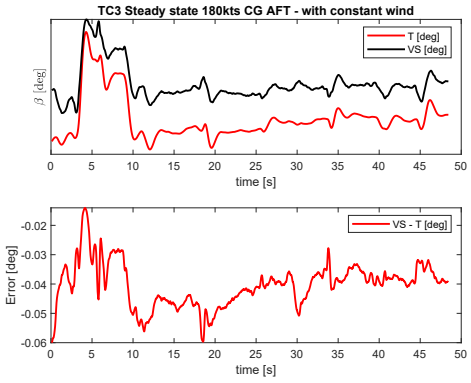
Figure 80: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, p, q, r, \delta_e, \delta_f, \delta_{f,LE}, \delta_a, \delta_r, \frac{1}{2}\left(\delta_{th,RH} + \delta_{th,LH}\right)\right]$, 30 neurons, AoA/AoS as absolute value, training set: both no-wind both with wind maneuvers, testing set: with sinusoidal wind maneuvers

12.3 Without wind, influence of center of mass

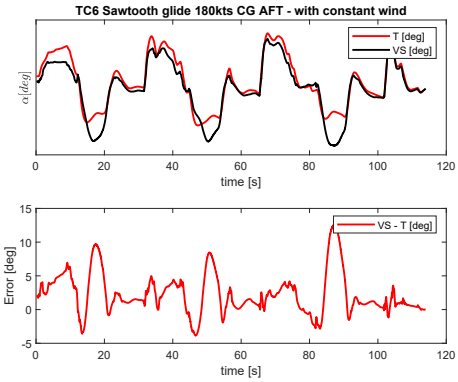
The same analysis carried out in Sect. 12.2 for the sinusoidal wind, is now pursued for evaluate the influence of the center of mass position. As comes out from Fig. 81, in this case as well it can be noticed a clear worsening of the α and β predictions with respect those obtained in Sect. 11.5 showing a peak that, in Fig. 81(c), even reaches 14°. These results suggest that also the center of mass position represents a key parameter for the NN training and its influence should be object of deeper studies in this regards.



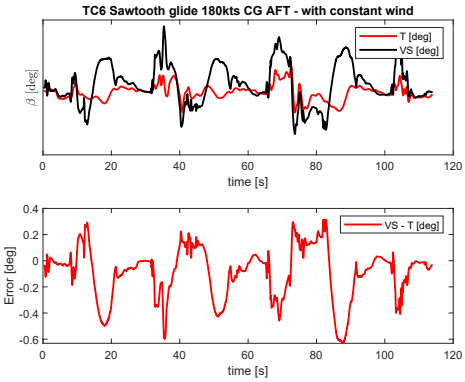
(a) TC3 Steady state 180kts



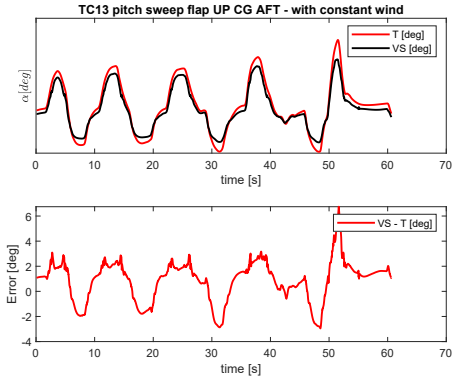
(b) TC3 Steady state 180kts



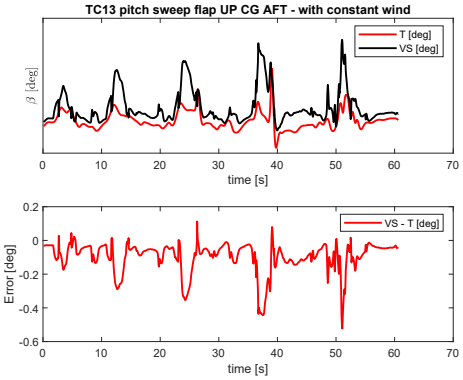
(c) TC6 Sawtooth glide 180kts



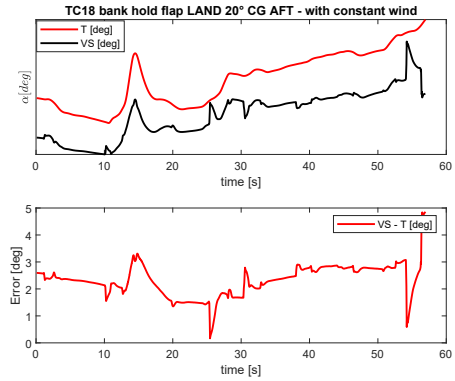
(d) TC6 Sawtooth glide 180kts



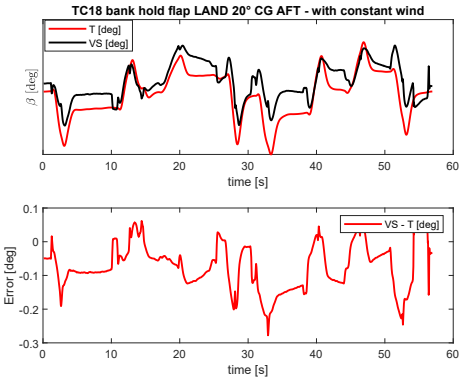
(e) TC13 pitch sweep flap UP



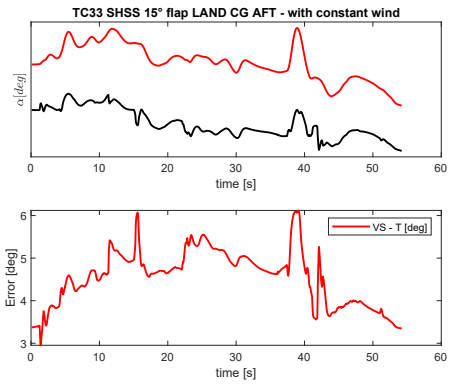
(f) TC13 pitch sweep flap UP



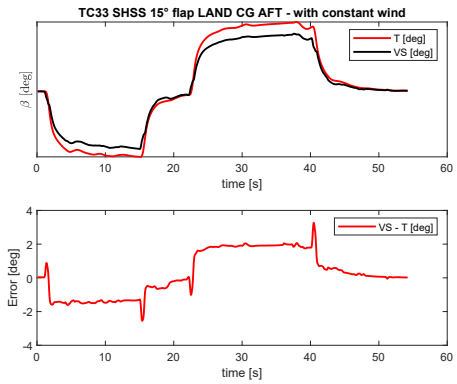
(g) TC18 bank hold flap LAND 20°



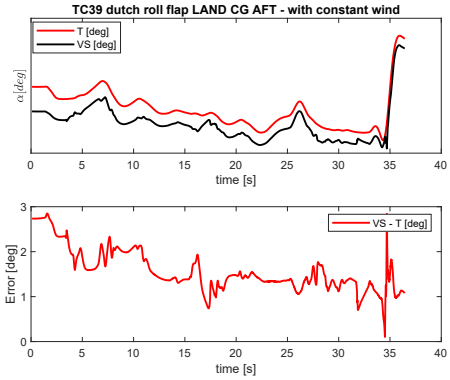
(h) TC18 bank hold flap LAND 20°



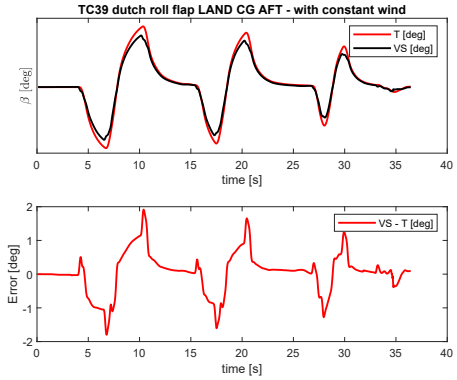
(i) TC33 SHSS flap LAND 15°



(j) TC33 SHSS flap LAND 15°



(k) TC39 dutch roll flap LAND

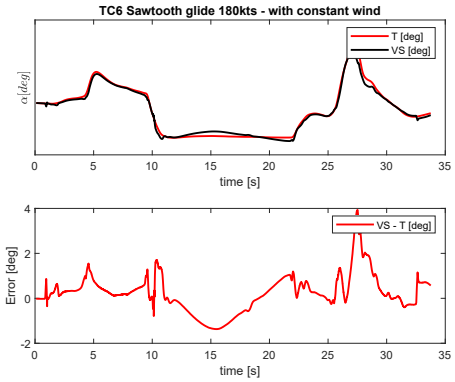


(l) TC39 dutch roll flap LAND

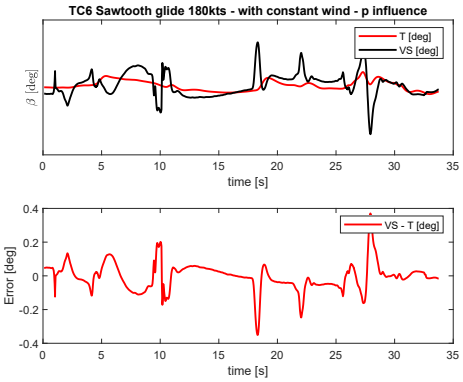
Figure 81: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, p, q, r, \delta_e, \delta_f, \delta_{f,LE}, \delta_a, \delta_r, \frac{1}{2}\left(\delta_{th,RH} + \delta_{th,LH}\right)\right]$, 30 neurons, AoA/AoS as absolute value, training set: both no wind both with wind maneuvers, testing set: no wind, CG in AFT position maneuvers

12.4 Sensitivity analysis

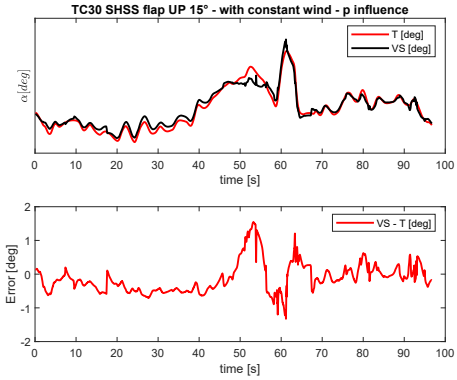
Standing the considerations and results obtained in Sect. 11.6, it will be now carried out a sensitivity analysis on signals contained within the input vector used to train $\text{NN}_{ass,hyb}$. As the importance of $[TAS, n_x, n_y, n_z, \phi, \theta]$ it's already been verified in Sect. 11.6, the attention will be now focused on angular velocities and commands. As previously discovered in 11.6, from Fig.s 82, 83, 84 comes out that angular velocities does not play a key role for NN's training in this case as well.



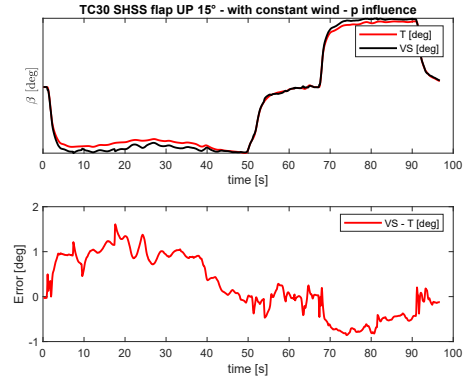
(a) TC6 Sawtooth glide 180kts



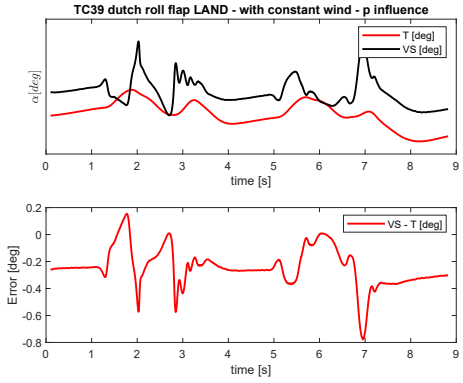
(b) TC6 Sawtooth glide 180kts



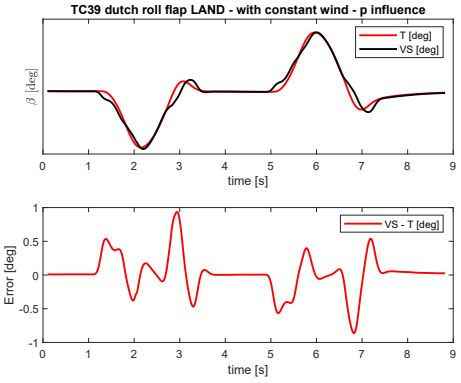
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°

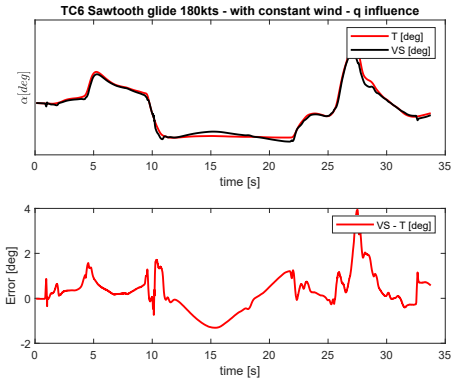


(e) TC39 dutch roll flap LAND

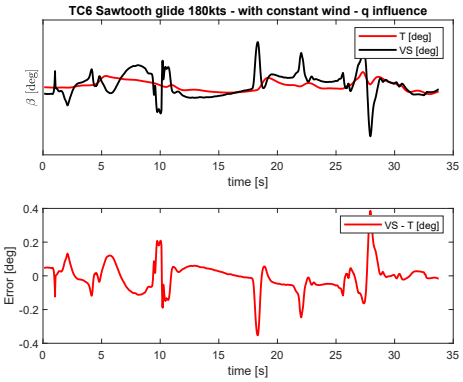


(f) TC39 dutch roll flap LAND

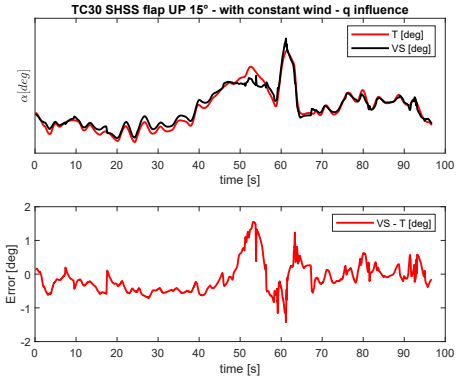
Figure 82: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, p, q, r, \delta_e, \delta_f, \delta_{f,LE}, \delta_a, \delta_r, \frac{1}{2}\left(\delta_{th,RH} + \delta_{th,LH}\right)\right]$, 30 neurons, AoA/AoS as absolute value, training set: both no wind both with wind maneuvers, testing set: with constant wind maneuvers, influence of a 20% error on p



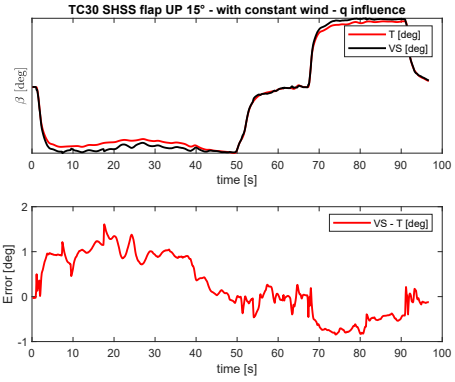
(a) TC6 Sawtooth glide 180kts



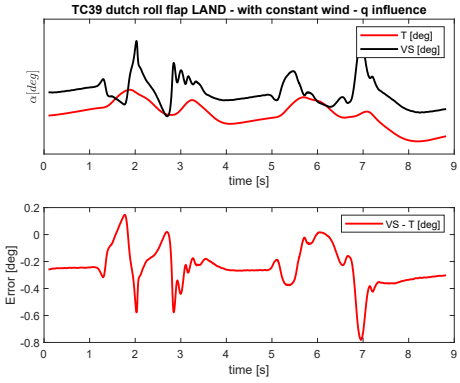
(b) TC6 Sawtooth glide 180kts



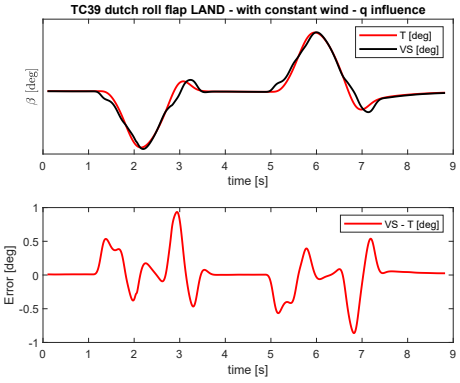
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°

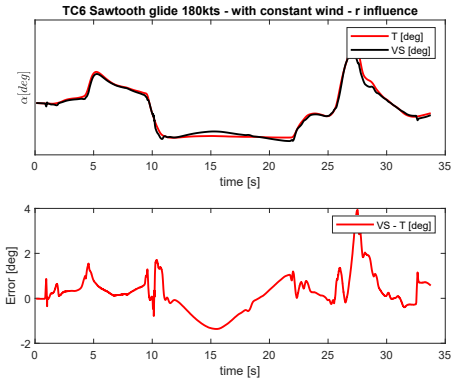


(e) TC39 dutch roll flap LAND

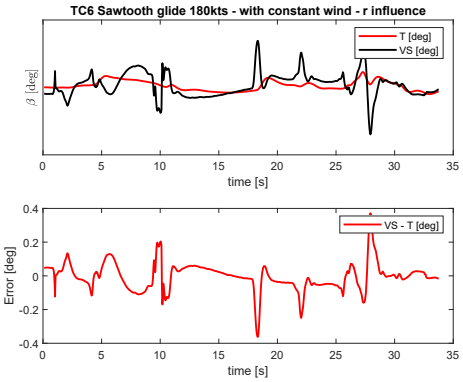


(f) TC39 dutch roll flap LAND

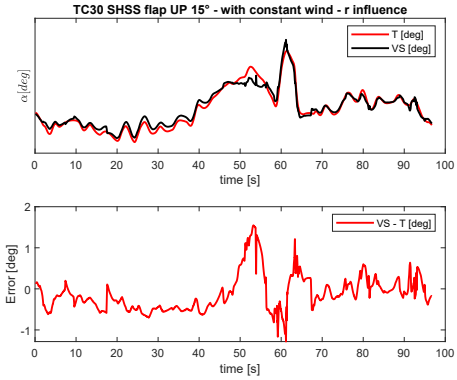
Figure 83: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, p, q, r, \delta_e, \delta_f, \delta_{f,LE}, \delta_a, \delta_r, \frac{1}{2}\left(\delta_{th,RH} + \delta_{th,LH}\right)\right]$, 30 neurons, AoA/AoS as absolute value, training set: both no wind both with wind maneuvers, testing set: with constant wind maneuvers, influence of a 20% error on q



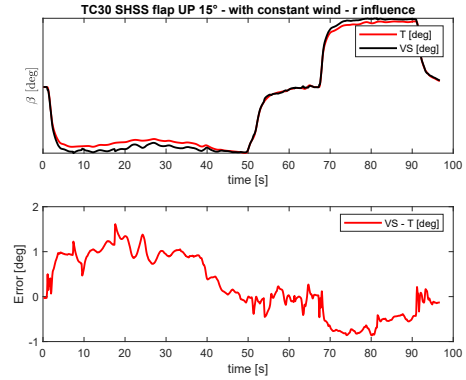
(a) TC6 Sawtooth glide 180kts



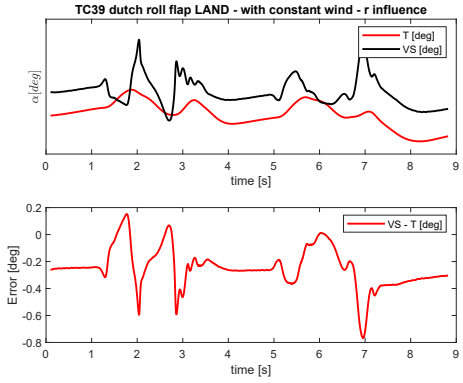
(b) TC6 Sawtooth glide 180kts



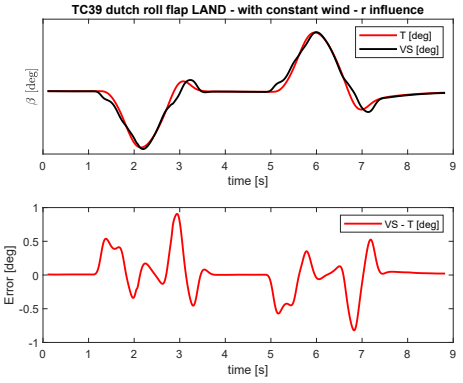
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°

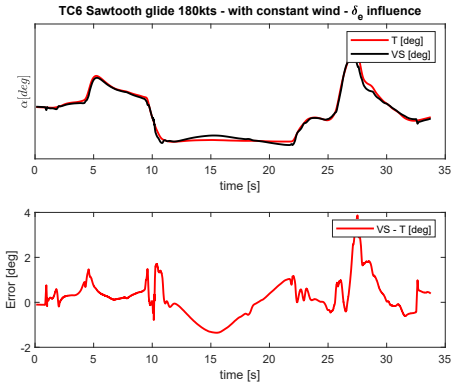


(e) TC39 dutch roll flap LAND

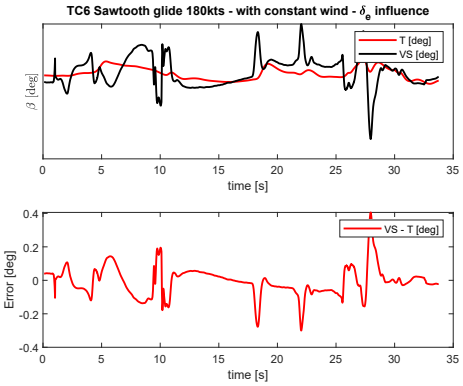


(f) TC39 dutch roll flap LAND

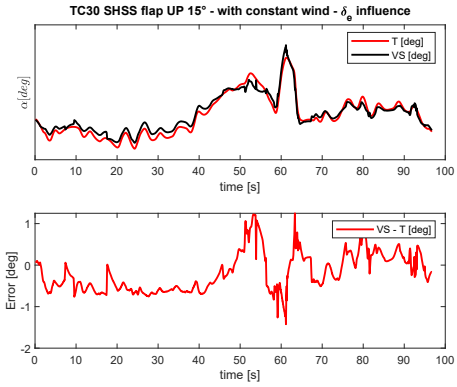
Figure 84: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, p, q, r, \delta_e, \delta_f, \delta_{f,LE}, \delta_a, \delta_r, \frac{1}{2}\left(\delta_{th,RH} + \delta_{th,LH}\right)\right]$, 30 neurons, AoA/AoS as absolute value, training set: both no wind both with wind maneuvers, testing set: with constant wind maneuvers, influence of a 20% error on r



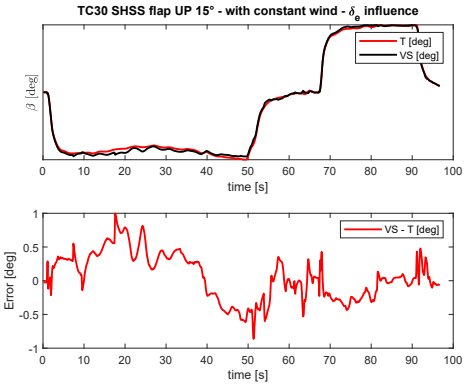
(a) TC6 Sawtooth glide 180kts



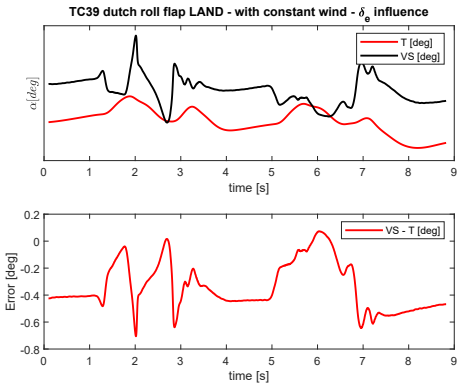
(b) TC6 Sawtooth glide 180kts



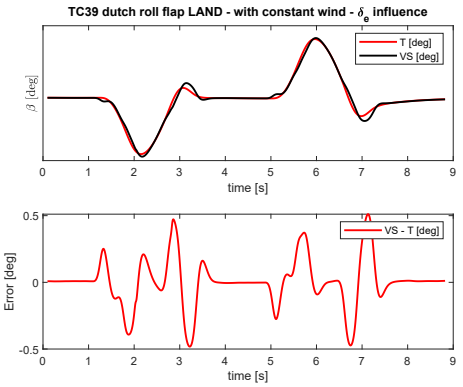
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°

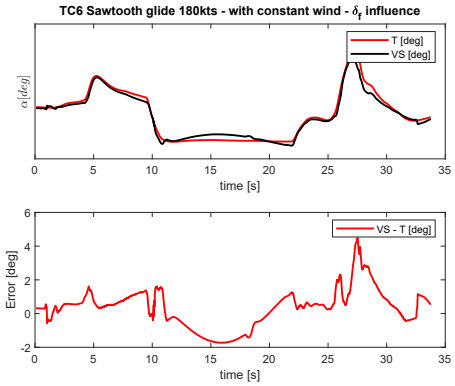


(e) TC39 dutch roll flap LAND

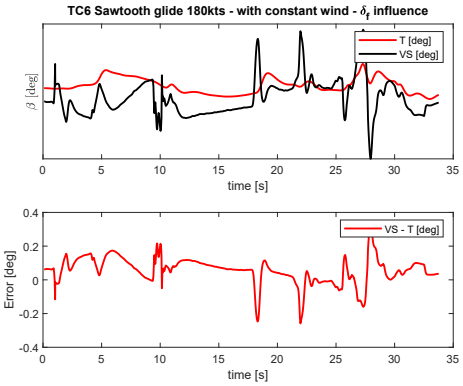


(f) TC39 dutch roll flap LAND

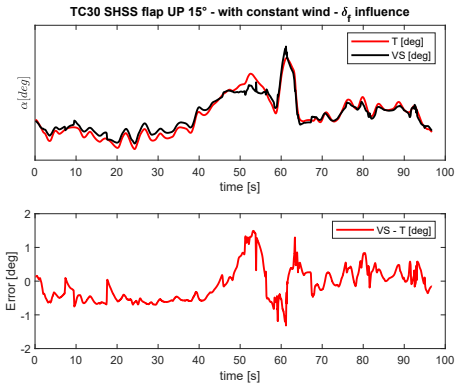
Figure 85: $\left[TAS, T\dot{A}S, n_x, n_y, n_z, \phi, \theta, p, q, r, \delta_e, \delta_f, \delta_{f,LE}, \delta_a, \delta_r, \frac{1}{2}\left(\delta_{th,RH} + \delta_{th,LH}\right)\right]$, 30 neurons, AoA/AoS as absolute value, training set: both no wind both with wind maneuvers, testing set: with constant wind maneuvers, influence of a 20% error on δ_e



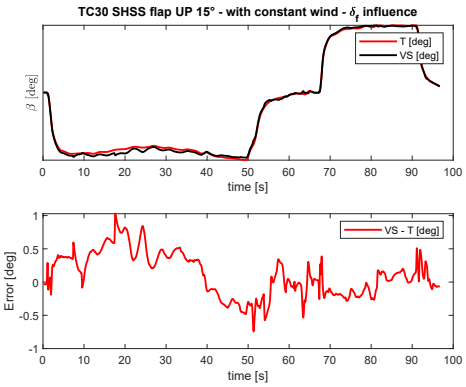
(a) TC6 Sawtooth glide 180kts



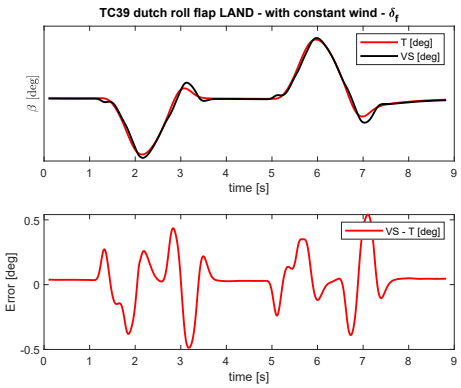
(b) TC6 Sawtooth glide 180kts



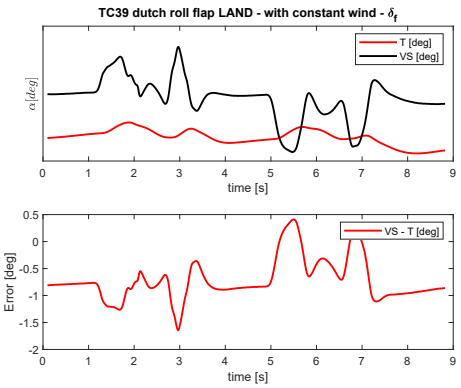
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°

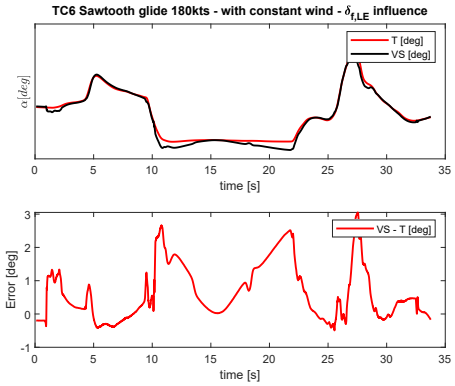


(e) TC39 dutch roll flap LAND

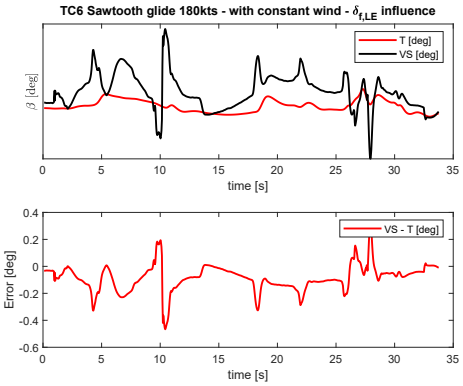


(f) TC39 dutch roll flap LAND

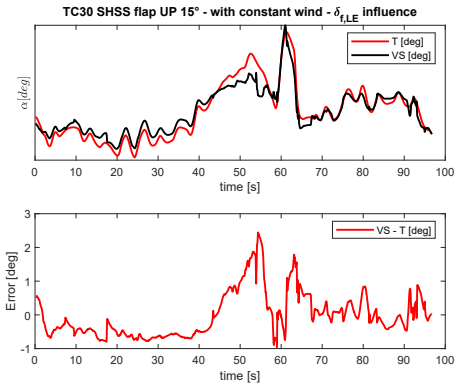
Figure 86: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, p, q, r, \delta_e, \delta_f, \delta_{f,LE}, \delta_a, \delta_r, \frac{1}{2}\left(\delta_{th,RH} + \delta_{th,LH}\right)\right]$, 30 neurons, AoA/AoS as absolute value, training set: both no wind both with wind maneuvers, testing set: with constant wind maneuvers, influence of a 20% error on δ_f



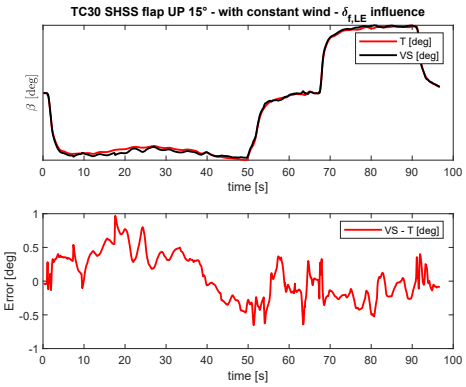
(a) TC6 Sawtooth glide 180kts



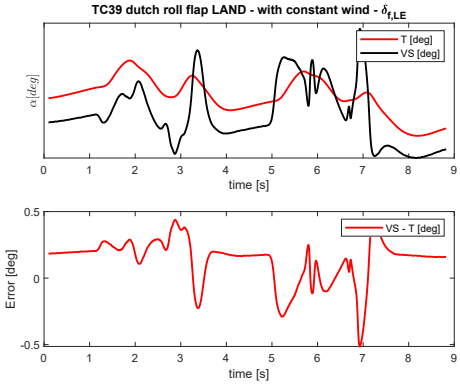
(b) TC6 Sawtooth glide 180kts



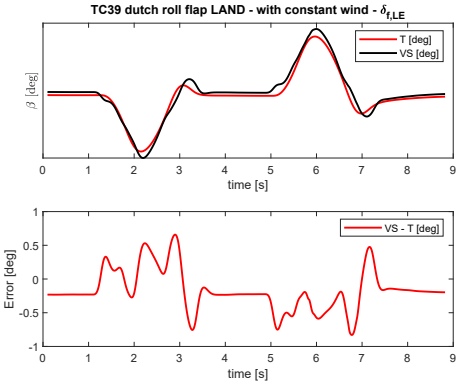
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°

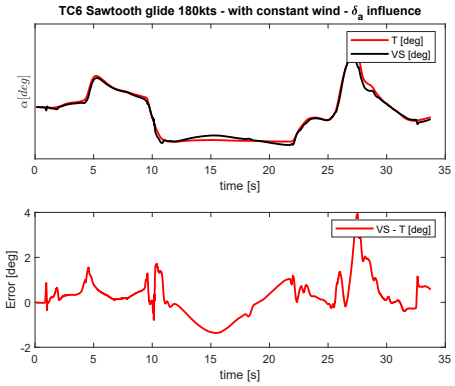


(e) TC39 dutch roll flap LAND

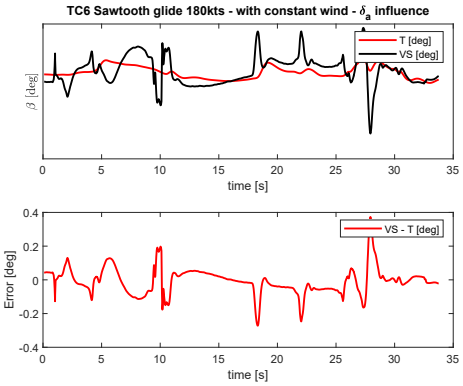


(f) TC39 dutch roll flap LAND

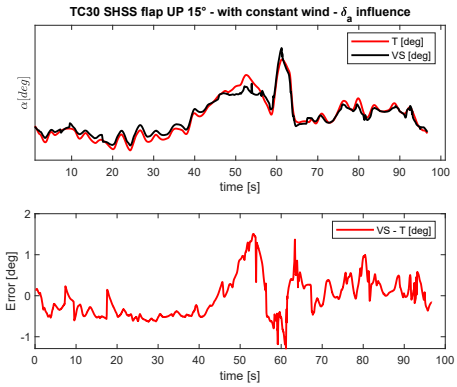
Figure 87: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, p, q, r, \delta_e, \delta_f, \delta_{f,LE}, \delta_a, \delta_r, \frac{1}{2}\left(\delta_{th,RH} + \delta_{th,LH}\right)\right]$, 30 neurons, AoA/AoS as absolute value, training set: both no wind both with wind maneuvers, testing set: with constant wind maneuvers, influence of a 20% error on $\delta_{f,LE}$



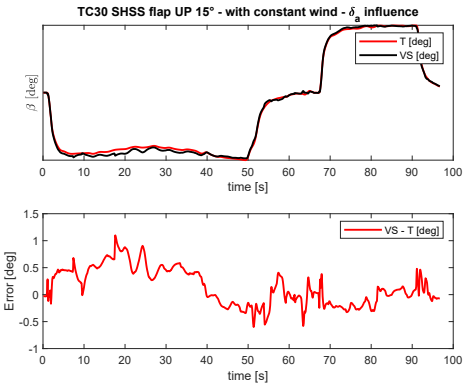
(a) TC6 Sawtooth glide 180kts



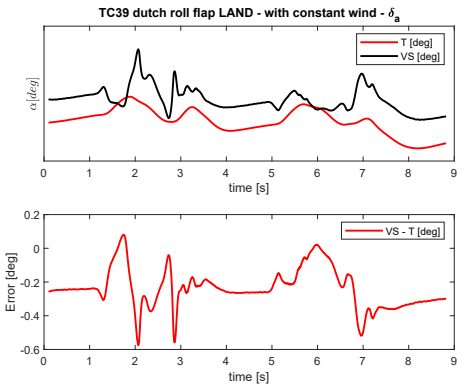
(b) TC6 Sawtooth glide 180kts



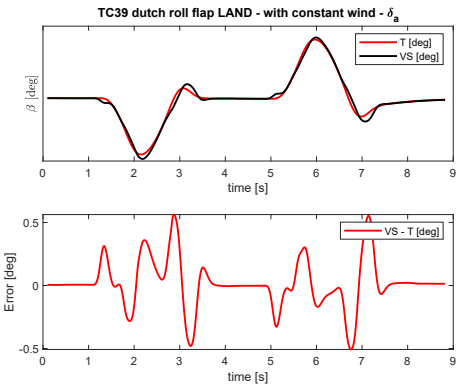
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°

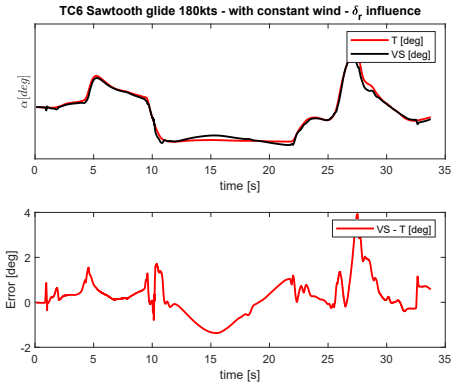


(e) TC39 dutch roll flap LAND

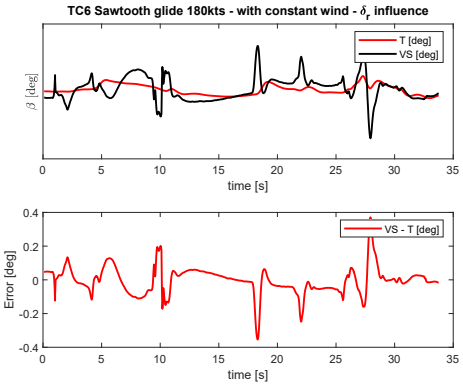


(f) TC39 dutch roll flap LAND

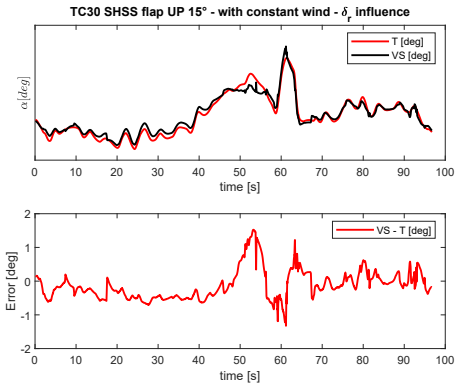
Figure 88: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, p, q, r, \delta_e, \delta_f, \delta_{f,LE}, \delta_a, \delta_r, \frac{1}{2}\left(\delta_{th,RH} + \delta_{th,LH}\right)\right]$, 30 neurons, AoA/AoS as absolute value, training set: both no wind both with wind maneuvers, testing set: with constant wind maneuvers, influence of a 20% error on δ_a



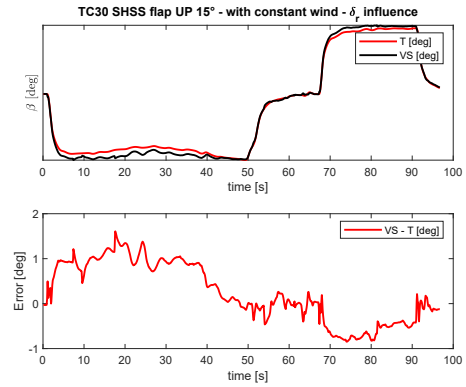
(a) TC6 Sawtooth glide 180kts



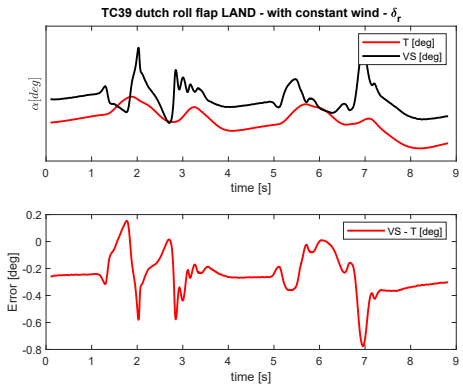
(b) TC6 Sawtooth glide 180kts



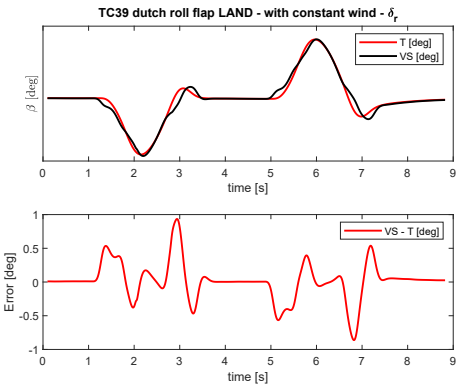
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°

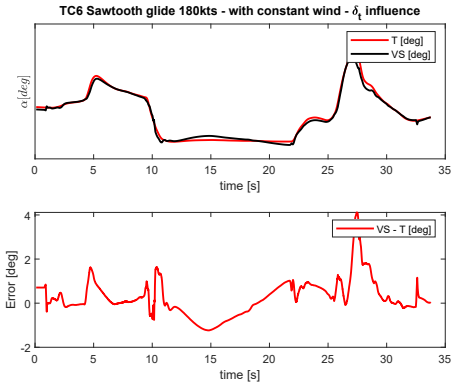


(e) TC39 dutch roll flap LAND

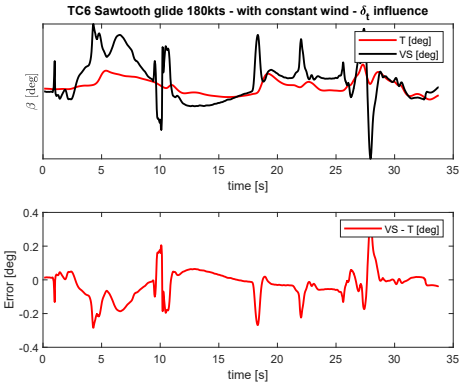


(f) TC39 dutch roll flap LAND

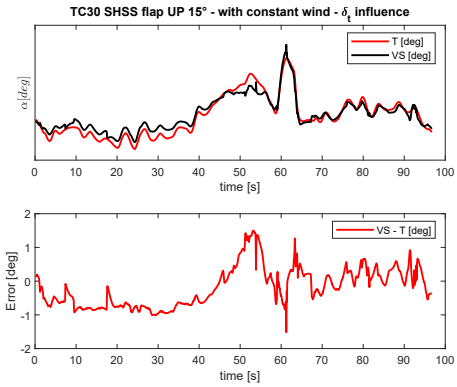
Figure 89: $\left[TAS, \dot{TAS}, n_x, n_y, n_z, \phi, \theta, p, q, r, \delta_e, \delta_f, \delta_{f,LE}, \delta_a, \delta_r, \frac{1}{2}\left(\delta_{th,RH} + \delta_{th,LH}\right)\right]$, 30 neurons, AoA/AoS as absolute value, training set: both no wind both with wind maneuvers, testing set: with constant wind maneuvers, influence of a 20% error on δ_r



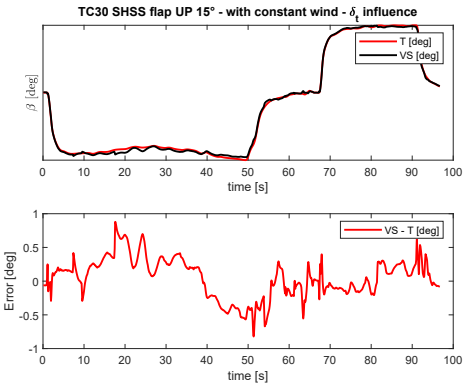
(a) TC6 Sawtooth glide 180kts



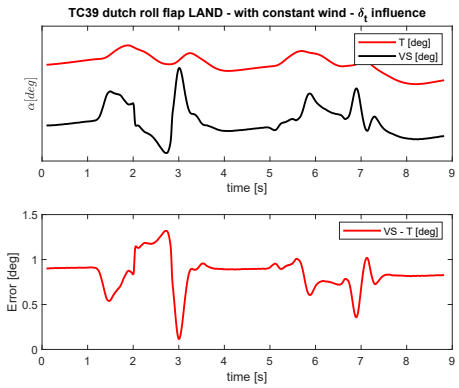
(b) TC6 Sawtooth glide 180kts



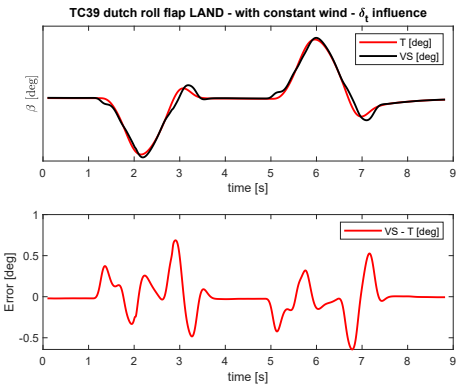
(c) TC30 SHSS flap UP 15°



(d) TC30 SHSS flap UP 15°



(e) TC39 dutch roll flap LAND



(f) TC39 dutch roll flap LAND

Figure 90: $\left[TAS, T\dot{A}S, n_x, n_y, n_z, \phi, \theta, p, q, r, \delta_e, \delta_f, \delta_a, \delta_r, \frac{1}{2}\left(\delta_{th,RH} + \delta_{th,LH}\right)\right]$, 30 neurons, AoA/AoS as absolute value, training set: both no wind both with wind maneuvers, testing set: with constant wind maneuvers, influence of a 20% error on $\frac{1}{2}\left(\delta_{t,LH} + \delta_{t,RH}\right)$

As it can be noticed in Figs.86, 87, 90 a 20% error in δ_f, $\delta_{f,LE}$ and δ_{th} is reflected in a deterioration of AoA/AoS estimation: it follows that these signals play a key role in the NN training for the AoA/AoS determination as their absolute values. On the other hand, from Fig. 85 it seems that, for these particular type of maneuvers, the angular deflections δ_e, δ_r and δ_a have a lower weight on AoA/AoS estimation and thus may be removed from the input vector.
Results obtained through the sensitivy analysis on $\text{NN}_{ass,nw}$ are summarized in Tab. 3.

Input Parameter	**Influence on NN's outputs**
TAS	high
$\dot{TAS}$	very low
θ	very high
ϕ	very high
n_x	high
n_y	high
n_z	high
p	low
q	low
r	low
δ_f	high
$\delta_{f,LE}$	high
δ_{th}	high
δ_e	very low
δ_a	very low
δ_r	very low

Table 3: Results of sensitivity analysis on $\text{NN}_{ass,hyb}$

13 Conclusion

This work deals with the development and application of an innovative solution for the estimation of aerodynamic angles on aircrafts using synthetic sensors based on Neural

Network techniques. Two type of Neural Networks were developed with the aim of predicting simultaneously both angle of attack both angle of sideslipe by processing inertial data, command surface positions, flap positions and dynamic pressure: the first NN was built estimating aerodynamic angles using a patented procedure [33] whereas the second one was developed evaluating α and β as their absolute value. From both virtual sensors came out that the flap configuration (i.e. UP/TO/LAND) is irrelevant for the Neural Network training, whilst a correct coverage of the input signals' hypercube with a sufficient number of both longitudinal, both latero-direction maneuvers is a necessary condition.
The first patented NN-based virtual sensor was trained using only without-wind maneuvers and was demonstrated to be very accurate if tested on maneuvers in absence of wind (AoA error always within $\pm 1.5°$ and AoS error always within $\pm 1.0°$) and in presence of constant wind without gusts (AoA/AoS error always within $\pm 1.5°$). On the other hand, the same virtual sensor showed degraded performance when tested on maneuvers in presence of constant wind and constant gusts (AoA error's absolute value that reached peaks of $2.8°$ despite the AoS error remained very low). The superimposition of external sinusoidal wind resulted to be an issue: even if the AoA/AoS trend was well captured and the error's absolute value in estimation of α always stayed within $\pm 2°$, the prediction of β results inaccurate and the error's absolute value reached peaks of the order of $5 - 6°$. Furthermore, tests on maneuvers in absence of wind with center of mass in forward and aft position showed a syntetic sensor that, in most cases, seemed to work efficently but, in some situations, provided a poor estimation of both aerodynamic angles with an error's absolute value that reached $4°$.
As the training set for the patented procedure was confined to maneuvers in absence of external wind and gusts, the second virtual sensor was trained, giving in output the respective aerodynamic angles absolute values, firstly with the same training set previously adopted and secondly with a balanced complete set of maneuvers with constant wind and without. The first result that came out was that the introduction of constant wind maneuvers in the training set does not lead to remarkable differences or improvements with respect to α and β prediction obtained with only no-wind maneuvers in the training set. This suggests that maneuvers without and with constant wind have the same influence on the Neural Network training set. For each of the two training set, the virtual sensor developed estimating α and β as their absolute value was tested on the same maneuvers with constant wind and gusts as before. Results demonstrated a better prediction for those maneuvers where external gusts were imposed and a degraded performance for those maneuvers where only a costant wind was present. On the other hand, tests both with sinusoidal wind both by varying the position of the center of mass, showed a clear worsening in α and β prediction with respect to those obtained using the NN obtained according to the patented procedure.
In light of these considerations it can be generally stated that an all weather, GPS-free and Neural Network-based synthetic sensor may currently represent a valid support to physical sensors only in absence of external wind or in presence of constant wind. In these conditions, the virtual sensor has always an accuracy of $\pm 1.5°$ and complies with certification requirements. On the other hand, to be feasibile in real flight situation, this work has demostrated as it is necessary a further study on transients due to variable external wind and on the influence of the location of the center of gravity. In fact, even

if the general trend of the aerodynamic angles' dynamics is always well captured, errors of the order of $5 - 6^\circ$ are too large with respect to the bandwidth of $\pm 1.5^\circ$ that was targeted.

Thanks

"*All people mentioned in this page have given a large contribute for the completion of this dissertation, but I want to point out that every eventual mistake or inaccuracy has to be charged only on me.*"

"*First of all I want to thank that person without whom this whole work would not exist: Angelo. More than a simply academic advisor, a brilliant professor who knows how to remind what is like to be a student. I will always admire you for both your professional both your human side.*"

"*The second person I want to mention is my Leonardo advisor, Massimo. Thanks to your spontaneity, friendliness and helpfulness you always made me feel like a colleague. I promise I will make good use of all those constructive life advices you gave me.*"

"*I proceed by thanking my friends and faithful companions who have always been next to me in this long journey: Francesco, Jacopo, Pietro and Tommaso. The jokes, fears and aspirations we shared are the best memories of these years and will remain indelible in my mind.*"

"*Next, a special thought to the dearest person to me: Noemi. Your love, joy and comprehension gave me all the support I needed to pursue my objectives everytime I thought that difficulties were bigger than me. Knowing you are by my side make me confident to overcome every single challenge that life will present us.*"

"*In conclusion, my heartfelt thanks to my family, without whose sacrifices I would never have arrived to this important point of my life. I hope I have made you proud of the man I have become, all my present and future achievements will be always yours.*"

References

[1] URL: https://en.wikipedia.org/wiki/World_Geodetic_System.

[2] URL: https://www.physik.uzh.ch/local/teaching/SPI301/LV-2015-Help/gmath.chm/3D_Cartesian_Coordinate_Rotation_Euler.html.

[3] URL: https://history.nasa.gov/SP-367/f117.htm.

[4] *AeroSmart S.r.l.* http://www.aerosmartsrl.it/. May 2019. URL: http://www.aerosmartsrl.it/.

[5] Andrew J. Meade Ankur Srivastava and Kurtis R. Long. "Learning Air-Data Parameters for Flush Air Data Sensing Systems". In: *IEE Transactions On Systems, Man, and Cybernetics* (2010).

[6] Fabio Balzano et al. "Air Data Sensor Fault Detection with an Augmented Floating Limiter". In: *Hindawi* (2018).

[7] M. Battipede, P. Gili, and A. Lerro. "Neural Networks for Air Data Estimation: Test of Neural Network Simulating Real Flight Instruments". In: *Engineering Applications of Neural Networks*. Ed. by Chrisina Jayne, Shigang Yue, and Lazaros Iliadis. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 282–294. ISBN: 978-3-642-32909-8.

[8] M. Battipede et al. "Development of Neural Networks for Air Data Estimation: Training of Neural Network Using Noise-Corrupted Data". In: *Proceedings of CEAS 2011 - The international Conference of the European Aerospace Societies*. 2011. ISBN: 9788896427187.

[9] Manuela Battipede et al. "Novel Neural Architecture for Air Data Angle Estimation". In: *Engineering Applications of Neural Networks: 14th International Conference, EANN 2013, Halkidiki, Greece, September 13-16, 2013 Proceedings, Part I*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 313–322. ISBN: 978-3-642-41013-0. DOI: 10.1007/978-3-642-41013-0_32. URL: http://dx.doi.org/10.1007/978-3-642-41013-0%5C_32.

[10] A. Brandl et al. “Sensitivity analysis of a neural network based avionic system by simulated fault and noise injection”. In: *AIAA Modeling and Simulation Technologies Conference, 2018* 209959 (2018). DOI: 10.2514/6.2018-0122.

[11] Alberto Brandl et al. “Air Data Virtual Sensor: a Data-Driven Approach to Identify Flight Test Data Suitable for the Learning Process”. In: *5th CEAS Specialist Conference on Guidance, Navigation and Control*. Apr. 2019.

[12] Kenneth R. Britting. *Inertial Navigation Systems Analysis*. Artech House, 2010, pp. 11–64.

[13] Jr. Arthur E. Bryson and Yu-Chi Ho. *Applied Optimal Control*. Taylor & Francis, 1969.

[14] Bureau d’enquêtes et d’Analyses pour la sécurité de l’aviation civile. *Report of accident on 27 November 2008 off the coast of Canet-Plage (66) to the Airbus A320-232 registered D-AXLA operated by XL Airways Germany*. URL: https://www.bea.aero/.

[15] R. P. Collinson. *Introduction to Avionics Systems*. Springer Science and Business, 2013.

[16] G. E. Hinton D. E. Rumelhart and R. J. Williams. *Parallel distributed processing: explorations in the microstructure of cognition*. Vol. 1. MIT Press, 1986. Chap. Learning internal representation by error propagation, pp. 318–362.

[17] Agostino De Marco and Domenico P. Coiro. “Quaderno 1, Terne di riferimento”. In: *Elementi di Dinamica e simulazione di volo*. 2017.

[18] Xiaoping Du et al. “A Prediction Model for Vehicle Sideslip Angle based on Neural Network”. In: (Sept. 2010). DOI: 10.1109/ICIFE.2010.5609398.

[19] E. Dubrova. *Fault-Tolerant Design*. Springer, 2013.

[20] *Dutch Roll*. URL: http://avidic.ir/public/show-word-single/1742/dutch-roll.

[21] "Final report On the accident on 1st June 2009 to the Airbus A330-203 registered F-GZCP operated by Air France flight AF 447 Rio de Janeiro – Paris". In: *BEA* (2012).

[22] Stephen M. Goldfeld, Richard E. Quandt, and Hale F. Trotter. "Maximization by Quadratic Hill-Climbing". In: *Econometrica* 34.3 (1966), pp. 541–551. ISSN: 00129682, 14680262. URL: http://www.jstor.org/stable/1909768.

[23] W. Gracey. *Measurement of Aircraft Speed and Altitude*. United States, 1980.

[24] W. Gracey. *Measurement of Aircraft Speed and Altitude*. United States, 1980.

[25] Simon Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 1999.

[26] "Indonesia Report: Pilots, Ground Crew Share Blame With Boeing For Lion Air Crash". In: *NPR.org* (2019).

[27] *Insect in pitot tube causes accident*. URL: https://generalaviationnews.com/2019/05/10/insect-in-pitot-tube-causes-accident/.

[28] A. S. Krogh J. A. Hertz and R. G. Palmer. *Introduction To The Theory Of Neural Computation*. Westview Press, 1991.

[29] A. S. Krogh J. A. Hertz and R. G. Palmer. *Introduction To The Theory Of Neural Computation*. Westview Press, 1991.

[30] C.T Sun J.S.R. Jang and E. Mizutani. *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. MATLAB curriculum series. Prentice Hall, 1997.

[31] C. S. Lee and N. J. Wood. "Calibration and Data Reduction for a Five-hole Probe". In: *Joint intitute for aeronautics and acoustics* (1986).

[32] Leonardo di Lena. *Meccanica del volo - parte 1 di 2*. URL: https://issuu.com/fotter/docs/meccanica_del_volo_-_parte_1_di_2_-_v3.57.

[33] A. Lerro, M. Battipede, and P. Gili. "System and process for measuring and evaluating air and inertial data". EP3022565A2. 2013.

[34] A. Lerro et al. "Advantages of Neural Network Based Air Data Estimation for Unmanned Aerial Vehicles". In: *International Journal of Mechanical, Aerospace,*

Industrial, Mechatronic and Manufacturing Engineering 19 (2017), pp. 1196–1205. DOI: 10.5281/zenodo.1130557.

[35] A. Lerro et al. "Comparison Between Numerical Results and Operative Environment Data on Neural Network for Air Data Estimation". In: *PROCEEDINGS of the 6th CEAS Air and Space Conference Aerospace Europe 2017*. 2017.

[36] A. Lerro et al. "Survey on a neural network for non linear estimation of aerodynamic angles". In: *2017 Intelligent Systems Conference, IntelliSys 2017* 2018-January (2018), pp. 929–935. DOI: 10.1109/IntelliSys.2017.8324240.

[37] A. Lerro et al. "Test in Operative Environment of an Artificial Neural Network for Aerodynamic Angles Estimation". In: *28th Society of Flight Test Engineers European Chapter Symposium (SFTE-EC 2017)*. 2017.

[38] Angelo Lerro. "Development and Evaluation of Neural Network-Based Virtual Air Data Sensor for Estimation of Aerodynamic Angles". Politecnico di Torino, 2012. DOI: 10.6092/polito/porto/2518884.

[39] Angelo Lerro et al. "Aerodynamic angle estimation: comparison between numerical results and operative environment data". In: *CEAS Aeronautical Journal* (Sept. 2019). ISSN: 1869-5590. DOI: 10.1007/s13272-019-00417-x. URL: https://doi.org/10.1007/s13272-019-00417-x.

[40] Angelo Lerro et al. "The Clean Sky 2 MIDAS Project - an Innovative Modular, Digital and Integrated Air Data System for Fly-by-Wire Applications". In: *2019 IEEE 5th International Workshop on Metrology for AeroSpace (MetroAeroSpace)*. June 2019, pp. 714–719. DOI: 10.1109/MetroAeroSpace.2019.8869602.

[41] KENNETH LEVENBERG. "A METHOD FOR THE SOLUTION OF CERTAIN NON-LINEAR PROBLEMS IN LEAST SQUARES". In: *Quarterly of Applied Mathematics* 2.2 (1944), pp. 164–168. ISSN: 0033569X, 15524485. URL: http://www.jstor.org/stable/43633451.

[42] N.K. Poulsen M. Norgaard O. Ravn and L.K. Hansen. *Neural Networks for Modelling and Control of Dynamic Systems*. Advanced Textbooks in Control and Signal Processing. Springer-Verlag London, 2000.

[43] Author links open overlay panelPaul M.Frank. *Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy: A survey and some new results*. 3 vols. Automatica, 1990, pp. 459–474.

[44] Donald W. Marquardt. "An Algorithm for Least-Squares Estimation of Nonlinear Parameters". In: *Journal of the Society for Industrial and Applied Mathematics* 11.2 (1963), pp. 431–441. ISSN: 03684245. URL: http://www.jstor.org/stable/2098941.

[45] J.T. McKenna. "Blocked static ports eyed in Aeroperu 757 crash". In: *Aviation Week & Space Technology* 145.20 (1996).

[46] J. M. Mendel and R. W. McLaren. *A prelude to neural networks*. Prentice Hall Press, 1994. Chap. Reinforcement-learning control and pattern recognition systems, pp. 287–318.

[47] Magnus Nørgaard, Charles Jorgensen, and Jim Ross. "Neural Network Prediction of New Aircraft Design Coefficients." NASA TM-112197". In: (June 1997).

[48] M. Oosterom and R. Babuska. "Virtual Sensor for the Angle-of-Attack Signal in Small Commercial Aircraft". In: (2006). Ed. by IEEE International Conference on Fuzzy Systems, pp. 1396–1403.

[49] Matthew M. Peet. *Spacecraft and Aircraft Dynamics. Lecture 9: 6DOF Equations of Motion*. URL: http://control.asu.edu/Classes/MMAE441/Aircraft/441Lecture9.pdf.

[50] F. Rosenblatt. "The perceptron: A probabilistic model for information storage and organization in the brain". In: *Psychological Review* 65 (1958), pp. 386–408.

[51] Marie-Michèle Siu, Borja Martos, and John V. Foster. "Flight Test Results of an Angle of Attack and Angle of Sideslip Calibration Method Using Output-Error Optimization". In: 2013.

[52] *Stalls in Jet Powered Airplanes*. URL: https://www.flightliteracy.com/stalls-in-jet-powered-airplanes/.

[53] Renee Swischuk and Douglas Allaire. "A Machine Learning Approach to Aircraft Sensor Error Detection and Correction". In: Jan. 2018. DOI: 10.2514/6.2018-1164.

[54] *The Different Frames and the Keplerian Elements*. URL: https://adcsforbeginners.wordpress.com/tag/earth-centred-inertial-frame/.

[55] J.E. Tomayk. *Computers Take Flight: A History of NASA's Pioneering Digital Fly-By-Wire Project*. CreateSpace Independent Publishing Platform, 2013.

[56] *Turn Performance*. URL: http://code7700.com/aero_turn_performance.htm.

[57] P.J. Werbos. "Beyound Regression: New Tools for Prediction and Analysis in the Behavioral Sciences". PhD thesis. 1974.

[58] H. White. "Connectionist nonparametric regression: multiplayer feedforward network can learn arbitrary mappings". In: *Neural Networks*. Vol. 3. 1990.