POLITECNICO DI MILANO
Scuola di Ingegneria Industriale e dell'Informazione

Master thesis in Aeronautical Engineering

POLITECNICO
MILANO 1863

Methodology for adjoint-based shape optimization for noise reduction of a wheel

Advisor:
Prof. Paolo Schito

Co-Advisors:
Ing. Francesco Fabio Semeraro
Ing. Alberto Artoni

Mattia Marelli
Matr. 905747

Academic year 2020-2021

Non chi comincia
ma quel che persevera

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

New technologies and new policies have favor the increase in electric cars sales. The absence of noise generated by the heat engine, gives more importance to the noise generated by aerodynamics. In this context, reducing noise means considerably increasing comfort and therefore having more competitive cars for the market. The main goal of this project is to develop a methodology for noise reduction based on the adjoint operator for the shape optimization. This thesis can be divided into two sections. In the first, a briefly recap of CFD methods and aeroacoustic state-of-the-art is presented. Afterwards the continuous adjoint formulation for field integral objective functions used in steady-state, incompressible aerodynamic optimization is developed. The formulation includes the full differentiation of the Spalart–Allmaras turbulence model based on wall functions. In the second part, in order to reduce the noise generated, the developed adjoint method is used for optimize a cube using volumetric B-Splines as parameterization tool. Starting from literature and analyzing its reliability, an appropriate objective function to be minimized has been defined. It is expressed by the integral of the squared turbulent viscosity over a volume residing next to the geometry. It should be stressed that if the commonly used "frozen turbulence" assumption was made, the adjoint method would not have been able to provide any kind of sensitivity derivatives. This objective function, in fact, depends exclusively upon the turbulent viscosity. After successfully optimize the cube, the developed methodology is then applied to a wheel where some different settings has been tested in order to explore the formulation limits.

Keywords: CFD, adjoint method, optimization, RANS, CAA, aeroacoustics, splines, wheel

SOMMARIO

Le nuove tecnologie e le nuove politiche hanno favorito l'ascesa delle auto elettriche che annullando il rumore generato dal motore termico hanno aumentato l'importanza del rumore generato dall'aerodinamica. In questo contesto ridurre il rumore significa aumentare considerevolmente il confort e quindi avere automobili più vendibili. L'obbiettivo principale di questa tesi consiste nell'introdurre una formulazione aggiunta per l'ottimizzazione in campo aeroacustico. Questa tesi può essere divisa in due parti. Nella prima viene presentato un breve riassunto dei metodi CFD e dello stato dell'arte in campo aeroacustico. Successivamente viene sviluppata la formulazione aggiunta continua per funzioni obiettivo basate su integrali di volume, utilizzate nell'ottimizzazione aerodinamica di flussi incomprimibili e stazionari. La formulazione include la differenziazione completa del modello di turbolenza di Spalart-Allmaras basato sulle funzioni di parete. Nella seconda parte, con lo scopo di ridurre il rumore generato, il metodo aggiunto sviluppato viene utilizzato per ottimizzare un cubo utilizzando B-Spline volumetriche come strumento di parametrizzazione. Partendo dalla letteratura e analizzandone l'affidabilità, un'opportuna seppur approssimata funzione obiettivo da minimizzare, è espressa dall'integrale su un volume posto vicino alla geometria del quadrato della viscosità turbolenta. Va sottolineato che se fosse stata fatta l'assunzione comunemente usata della "turbolenza congelata", evitando così la differenziazione del modello di turbolenza, il metodo aggiunto non sarebbe stato in grado di fornire alcun tipo di derivata di sensitività, poiché tale funzione obiettivo dipende esclusivamente dalla viscosità turbolenta. Dopo aver ottimizzato con successo il cubo, la metodologia sviluppata viene quindi applicata ad una ruota in cui sono state testate alcune impostazioni diverse per esplorare i limiti della formulazione.

Parole chiave: CFD, metodologia aggiunta, ottimizzazione, RANS, CAA, aeroacustica, splines, ruota

RINGRAZIAMENTI

I would like to thank Prof. P. Schito for being my supervisor and letting me have this opportunity to work on a project which merges aeroacoustics with my interest for adjoint method. There were times I struggled with issues that looked complicated but you taught me that also a wrong result is a result from which I can learn. There is always a positive side even in the difficulties. Thank you for your extreme availability and kindness.

Thanks to Ing F. F. Semeraro who helped me with guidance on computational work. You have given me a huge support in any condition. Thanks to our meetings I have learned many things from you about aerodynamic and general CFD problems.

Thanks to Ing A. Artoni who let me gradually develop my analytical skills through our meetings and his hints. Your mathematical soul motivated me to expand the theoretical background of my work.

I would like to thank all three of you for helping a student become a better engineer.

A big thanks to my family who believed in me in any second of my academic journey. You encouraged me to overcome difficulties. You helped me to never loose the hope in dark moments and you were the first to share my joy in the happy ones.

A special thanks to my little brother Paolo who, especially during the lockdown, had to endure my nervousness, my tension and my constant request for silence to work despite he is only 6 years old.

I want to thanks also my girlfriend and all the friends that I stressed out in this last months for a tips on the work or just for a few moments of relaxation.

Things have been difficult in 2020 and 2021, but luckily I could count on all of you.

1 INTRODUCTION

1.1 ELECTRIC VEHICLES

Electric vehicles (EV) sales are growing up in the global market. In 2020 they grew up of 41% (20 millions of new ones sold) while the global automotive industry has lost the 16%. If the current trends and polices hold, it is estimated that the green vehicles will reach 145 millions sales without considering two and three wheels, which is the 7% of the road fleet. This number could be highly underestimated if the governments will commit to zero greenhouse gas emissions by 2050, reaching picks of 230 millions sales according to [1]. Road transport is one of the largest contributors to global CO2 emissions. Despite the expected rapid rise in EV sales, most countries are still not on track to bring road transport emissions to zero by mid-century. Bloomberg's new Net Zero Scenario looks at what is needed to reach 'net-zero' emissions in road transport by 2050. In this scenario, 100% of the world's road fleet runs on electricity or hydrogen by that year. As soon as 2030, nearly 60% of new car sales must be zero emissions, to stay on track for the Net Zero Scenario [1].

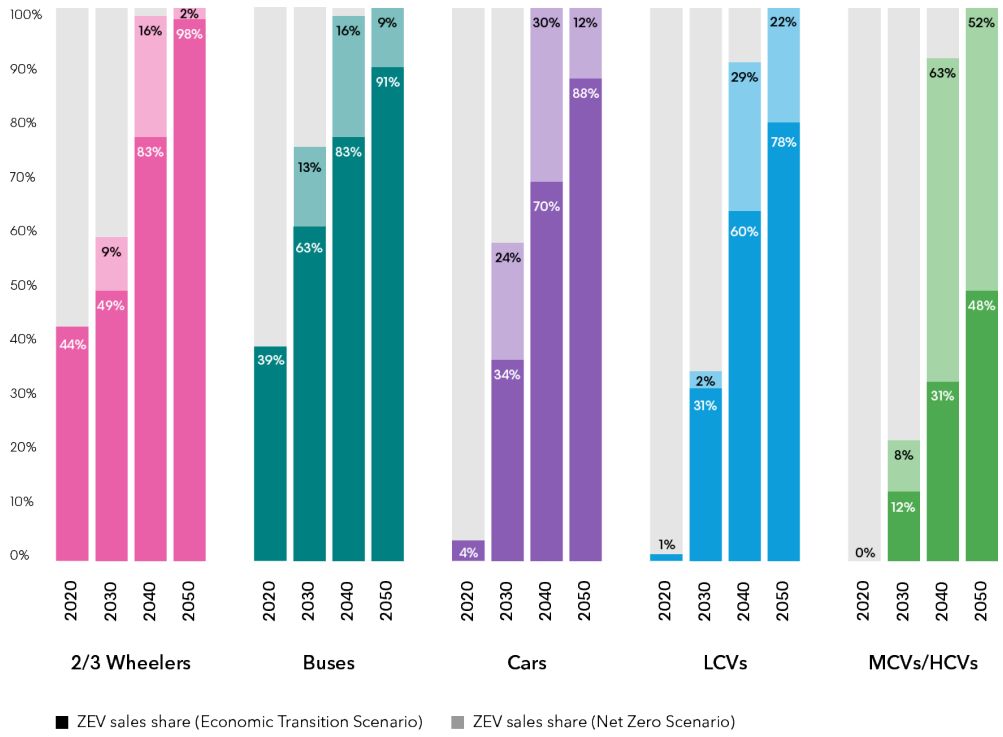


Figure 1.1: Zero-emission vehicle sales by segment: Economic Transition Scenario and Net Zero Scenarios [1]

The forecast on the trend of the electric vehicle market comes from the International Energy Agency (AIE) contained in the report "Global Electric Vehicle Outlook 2021". According to the intergovernmental organization, which deals with energy policies, the electric fleet of cars in circulation was ten millions at the end of 2020 to which must be added one million vans, trucks and buses. In the first quarter of this year, explains the IEA, the momentum of the sector continued with sales increased by 140 % compared to the same period of 2020 (500 thousand in China and 450 thousand in Europe) [2]. Electric cars, despite the need for more and more efficient charging infrastructures and more powerful batteries, are popular, regardless of state subsidies, notes the IEA.

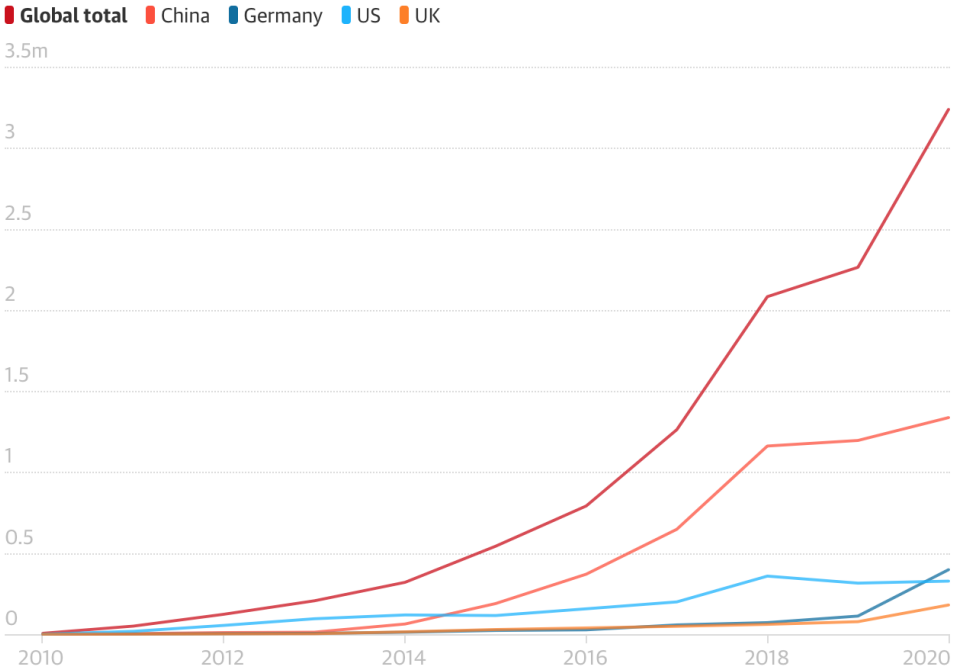


Figure 1.2: Electric vehicle sales growth from 2020 [3]

In 2020 the public columns were 1.3 million, of which 30 % fast, marking an increase of 45 % compared to 85 % in 2019, with China in the lead. Italy (with 0.13) is above the EU target for the end of 2020 which was a column for 10 vehicles (0.01 in 2020) and also exceeds the European average 0.09. The automakers, for their part, offered 370 models of electric cars in 2020, with an increase of 40 % on an annual basis, explains the AIE adding that eighteen of the 20 largest manufacturers (representing 90 % of world sales of cars) announced their intention to increase the number of models and the production of light vehicles. The shift from Internal Combustion Engines (ICE) to electrified vehicles will bring not only the challenge to the new powertrain development, but also a new cabin environment will be introduced. The passenger might receive some newly unwanted noise perceptions that was covered by the ICE one.

1.2 VEHICLE NOISE

The consideration of noise in or from a car hasn't always been as it is today. For a long time a loud engine expressed power and the speed of the car. Today, cars are widely used for commuting and long business trips and therefore a higher demand is on comfort. Too much noise is shown to cause tiredness, headaches, irritation and even if much is done to reduce known sources new ones always appear. Nowadays the focus is mainly to improve the environment for the customers inside the cars and to increase their comfort. But, there is a world outside.
In the European Union (EU) noise policy, it is stated that "65% of Europeans living in major urban areas are exposed to high noise levels" which are more than 55 dB Lden (day-evening-night level), and the main part of that noise comes from transportation [4]. At the same time World Health Organization (WHO) reports that, environmental noise causes 43 000 hospital admissions and 10 000 cases of premature deaths related to coronary heart disease every year. In addition, not only humans are affected by environmental noise but recent research shows harmful effects on wildlife as well because many animals rely on acoustic communication for e.g. mating or finding food [5].
The noise generated from vehicles can be divided into three main categories whose effect depend on the vehicle speed and type:

- drivetrain noise
- tyre-road interaction noise
- aerodynamic noise

The significance of aeroacoustics for the exterior noise of motor vehicles has been known for a long time; as early as 1983 Dobrzynskii found that from a speed of approx. 130 km/h and higher the induced aerodynamic noise was louder than all other noise sources of the vehicle combined [6]. For aerodynamic noise researchers originally attempted acoustic optimization in conventional wind tunnels. Especially with respect to the aerodynamic exterior noise this proved to be from difficult to impossible, due to the strong interference caused by the wind tunnels' inherent noise.
The use of psychoacoustic methods of assessment usually failed due to exceedingly high background noise. Normal wind tunnels are characterized by a lot of noise generators such as the fans, the circulating flow and the reflections regarding the test chamber and the relative channels; so more and more special aeroacoustic wind tunnels have been brought into operation in recent years.This has led to new discoveries in the field of aeroacoustics.

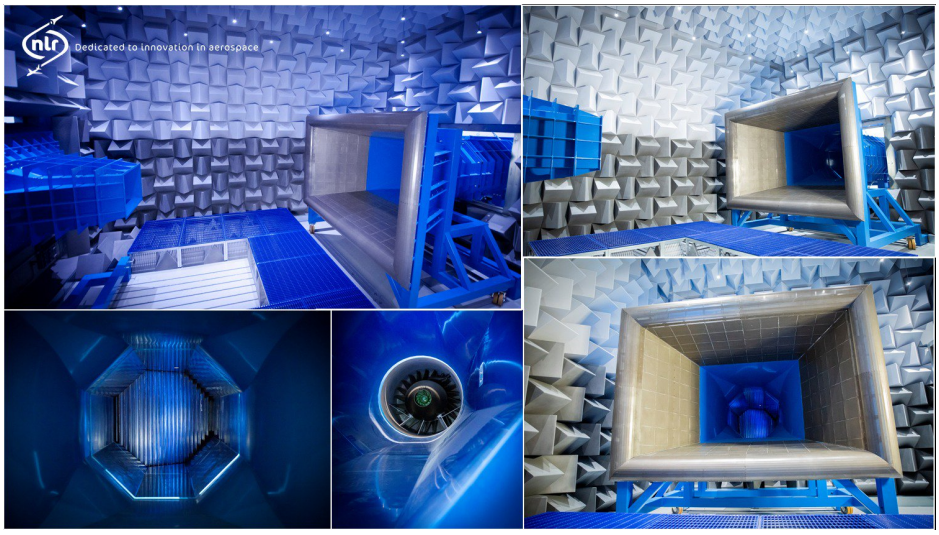


Figure 1.3: Royal NLR Anechoic wind tunnel which shows a particular use of sound absorbing materials.

Examinations considering the relevance of aerodynamic noise in relation to the overall noise have also been carried out with various vehicles and it turned out that in the case of quiet tire-road combinations the threshold velocity beyond which aerodynamic noise dominates both exterior and interior noise lies at approx. 130 km/h for passenger vehicles (Figure 1.4).

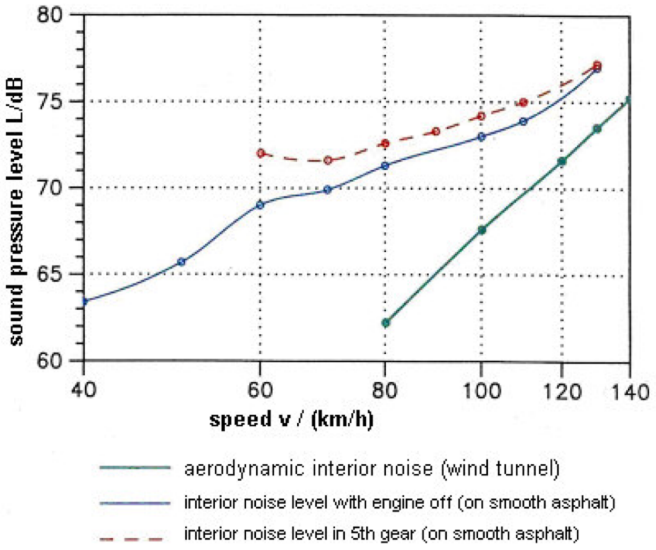


Figure 1.4: Aerodynamic interior noise measured in an aeroacoustic wind-tunnel and interior noise measured during driving; D-segment vehicle [6]

At low speeds ($< 20 km/h$) and high engine load it is the drive train that dominates the cabin noise. At low speeds and with a low engine load tire-road noise contributes the greatest part to the overall noise (Figure 1.5).

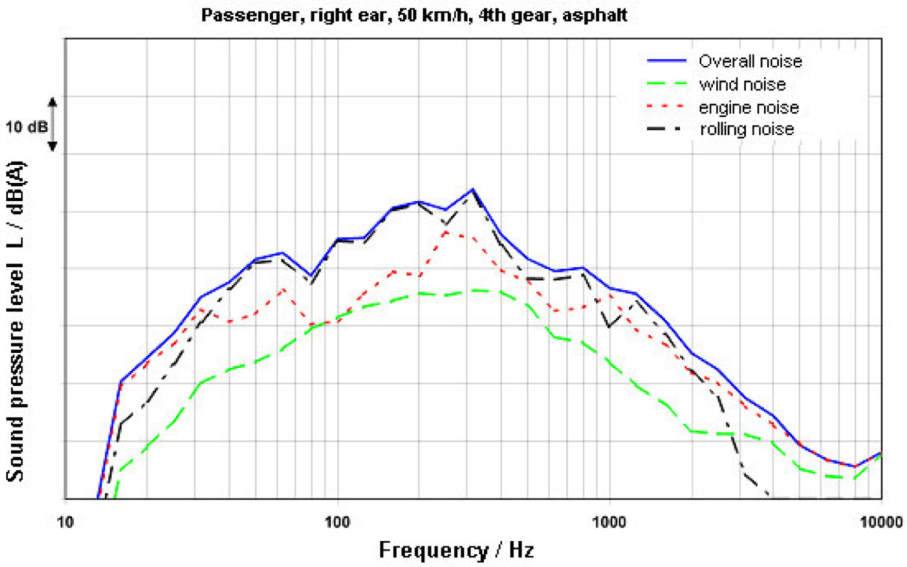


Figure 1.5: Overall noise and noise components of an upper middle-class passenger vehicle (D-segment) at $v = 50km/h$ [6]

At full-load acceleration this noise can play a dominant role due to the occasional sharp increase in generated noise caused by the high tractive forces acting on the tires. By increasing speeds the aerodynamic (Figure 1.6) noise of the vehicle becomes more significant as its acoustic power increases at about the 5th to 6th power of speed, whereas tire-road noise generally only increases at the 3rd to 4th power [6].

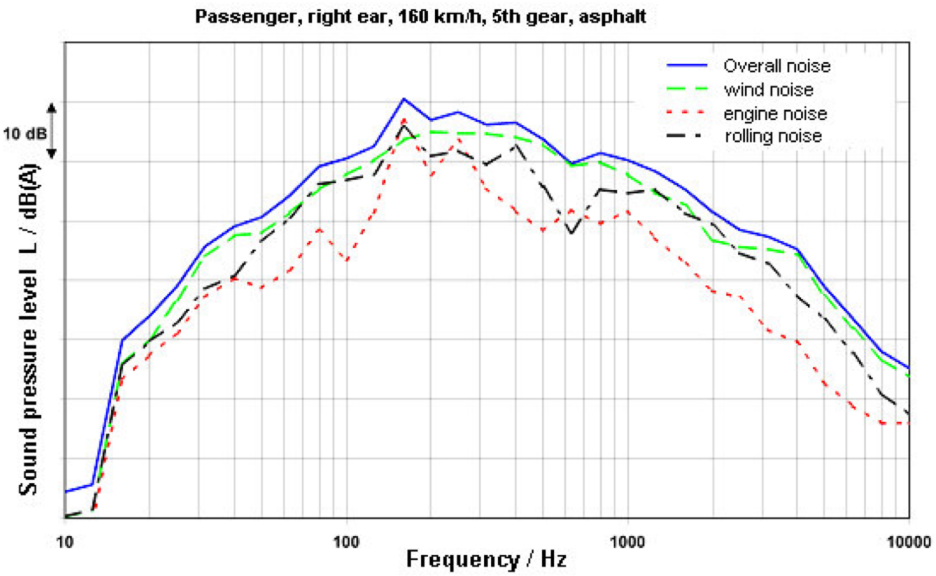


Figure 1.6: Overall noise and noise components of an upper middle-class passenger vehicle (D-segment) at $v = 160km/h$ [6]

Aeroacoustic noise in a vehicle has a lot of sources which has been studied by wind tunnel tests ([6]), by road tests ([7], [8]) or by Computational Aeroacoustic Approach ([9], [10]).

1.3 TYRE NOISE

Among all the noise sources in a vehicle (for example the side mirror [11] or the A-pillar [9]) this thesis focus is on the noise generated by aerodynamic interaction. For completeness a brief description of the different noise generation processes of a wheel will follow.

1.3.1 Air Pumping

Air pumping is widely considered one of the most critical noise mechanism contribution not only in tyre acoustic category but also in the entire tyre noise category. It happens due to groove deformation which squeezes out the air or sucks it into the tread grooves when tyre and pavement roll together. Compression and release of the air will result in a sizzling sound and the flow would accelerate due to the compressed air.

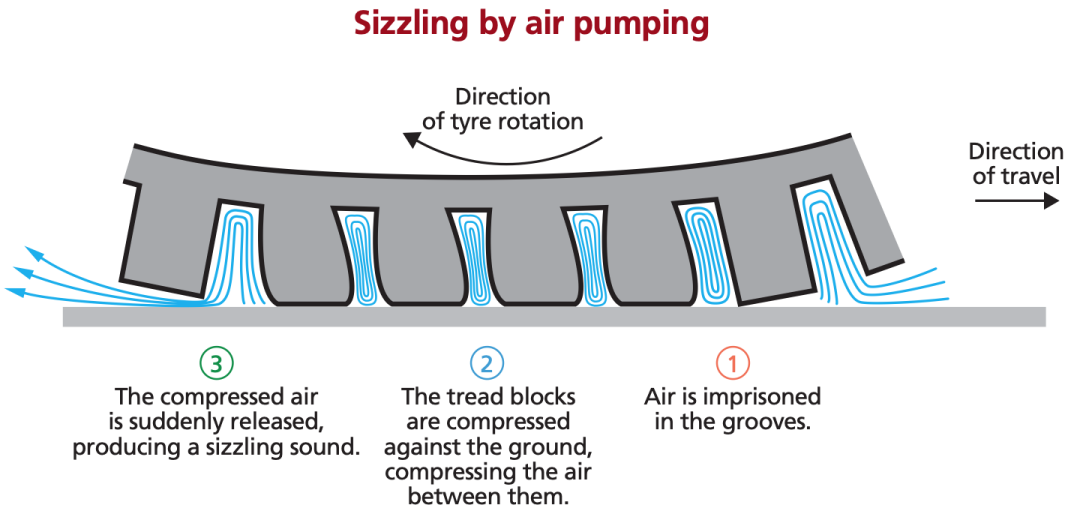


Figure 1.7: Air Pumping mechanism steps [12]

1.3.2 Pipe Resonance

Pipe resonance is considered an amplification phenomenon related to aeroacoustics of the tyre because the resonance itself doesn't generate sound until something excites the airflow in the tread voids near the resonance frequency. There are a lot of different kinds of grooved tyre: longitudinal grooves, lateral and/or diagonal (with respect to the rolling direction) and they can be open or closed end grooves.

Figure 1.8: Different tyre treads pattern [13]

These pipes have resonances as the pipes of musical instruments. In the open ones a standing wave with anti-nodes(maximum air displacement) may create at the open ends; it has a wavelength of twice the length of the pipes, which is called a half-wavelength resonator. In the most of the studies the general approach is to start from a simple definition of the half-wavelength and then an end correction of the resonance frequency is made comparing the result, as explained by Sandberg [13].

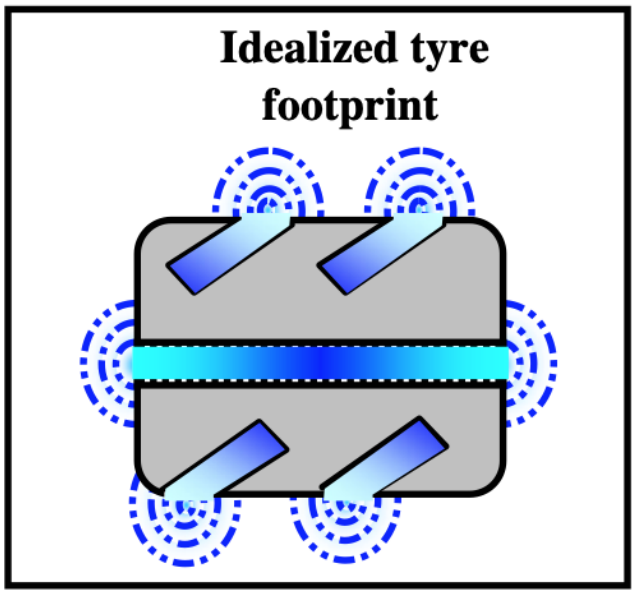


Figure 1.9: Idealized tyre footprint [13]

1.3.3 Horn effect

The geometry of tyre/road interaction is similar to a horn and this causes a bigger radiation efficiency of the tyre/road emitted noise than in the free field. It increases the coupling efficiency between the sound source and the air; in fact an impedance matching between air and the source is provided.

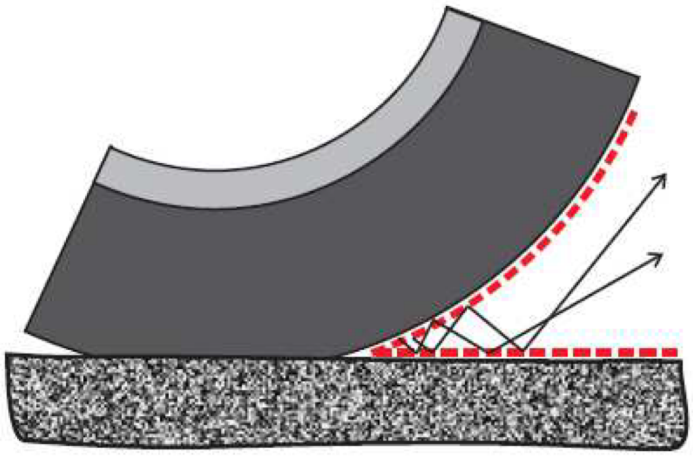


Figure 1.10: Horn amplification [14]

1.3.4 Air turbulence

Tyre noise due to air turbulence was not analyzed deeply previously because of its dependency on the each tyres' geometrical properties. The wake generated by the separation of the boundary layer from the wheel will introduce high energy of the shear layer, and two shoulder vortices are formed. The contact patch on the ground detaches two vortices but if an isolated tyre is rotating, a stagnation point horseshoe vortex cannot form, as the boundary layer near the ground does not separate in front of the wheel. By considering the aeroacoustic mechanism, these vortices are one of the primary sources contributing to tyre noise so that the wake and the turbulence will significantly contribute to tyre aeroacoustics.

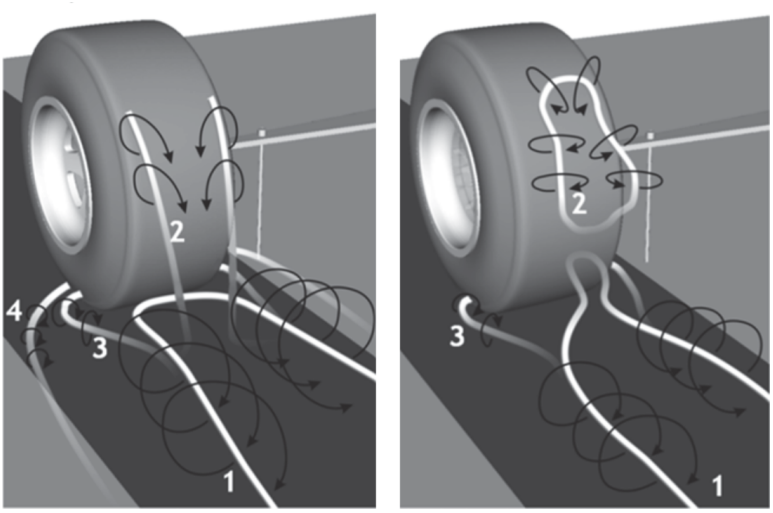


Figure 1.11: Difference between a rotating wheel and a fixed one [15]

In the context of this thesis, an isolated and not rotating wheel is considered with the aim of reduce the complexity and of mantain the focus on the optimization. Next in Ch.2 there will be a brief recall of flow simulations. Then in Ch.3 some tools for aeroacoustic will be presented while in Ch.4 the adjoint method and its formulation will be detailed. In Ch.5 it will be presented how it is implemented in OpenFoam and how I managed it while numerical tests and result will be in Ch.6, concluding in Ch.7 with the final thoughts.

2 COMPUTATIONAL FLUID DYNAMICS

There are many opportunities to observe turbulent flows in our everyday surroundings, whether it be smoke from a chimney, water in a river or waterfall, or the buffeting of a strong wind. In observing a waterfall, we immediately see that the flow is unsteady, irregular, seemingly random and chaotic, and surely the motion of every eddy or droplet is unpredictable. In engineering applications turbulent flows are prevalent, but less easily seen. In the processing of liquids or gases with pumps, compressors, pipe lines, etc., the flows are generally turbulent. Similarly the flows around vehicles such as airplanes, automobiles, ships, and submarines are turbulent. The mixing of fuel and air in engines, boilers, furnaces and the mixing of the reactants in chemical reactors take place in turbulent flows.

An important characteristic of turbulence is its ability to transport and mix fluids much more effectively than a comparable laminar flow. This is well demonstrated by an experiment first reported by Osborne Reynolds (1883). Dye is steadily injected on the centerline of a long pipe in which water is flowing. As Reynolds (1894) later established, this flow is characterized by a single non-dimensional parameter, now known as the Reynolds number Re. In general, it is defined by $Re = \frac{Ul}{\nu}$, where U and l are characteristic velocity and length scales of the flow, and ν is the kinematic viscosity of the fluid.

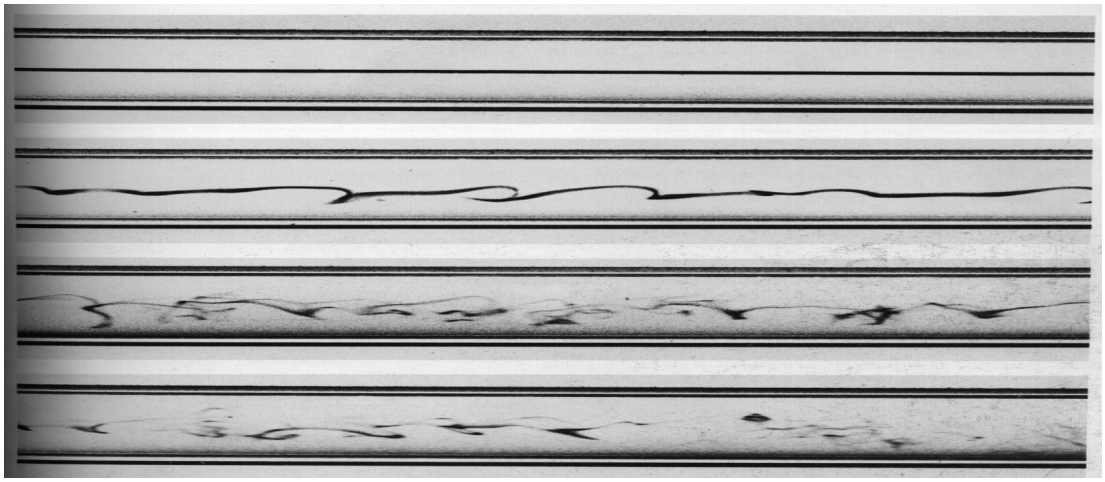


Figure 2.1: Visualization of flow at different Re number in a Reynold's experiment reptition

Turbulence is also effective at "mixing" the momentum of the fluid. As a consequence, on aircraft's wings and ships' hulls the wall shear stress (and

hence the drag) is much larger than it would be if the flow were laminar. Similarly, compared with laminar flow, rates of heat and mass transfer at solid-fluid and liquid-gas interfaces are much enhanced in turbulent flows.
The major motivation for the study of turbulent flows is the combination of the three preceding observations: the vast majority of flows is turbulent; the transport and mixing of matter, momentum, and heat in flows is of great practical importance; and turbulence greatly enhances the rates of these processes.

Many different techniques have been used to address many different questions concerning turbulence and turbulent flows. The first step toward providing a categorization of these studies is to distinguish between small-scale turbulence and the large-scale motions in turbulent flows. At high Reynolds number there is a separation of scales. The large-scale motions are strongly influenced by the geometry of the flow (i.e., by the boundary conditions), and they control the transport and mixing. The behavior of the small-scale motions, on the other hand, is determined almost entirely by the rate at which they receive energy from the large scales, and by the viscosity. Hence these small-scale motions have a universal character, independent of the flow geometry.

For fluid flows, be they laminar or turbulent, the governing laws are embodied in the Navier-Stokes equations which, despite considering the diversity and complexity of fluid flows, describe them accurately and in complete detail. The governing equations used are the fundamental equations of fluid dynamics: the continuity, momentum and the energy equation [16]. They are here written for completeness in conservative form for a compressible fluid using the index notation and Einstein convention even if the thesis will focus on incompressible flows.

- continuity equation:

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho u_i)}{\partial x_i} = 0 \tag{1}$$

- momentum equation:

$$\frac{\partial (\rho u_i)}{\partial t} + \frac{\partial (\rho u_i u_j)}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} \tag{2}$$

- energy equation:

$$\frac{\partial (\rho e_0)}{\partial t} + \frac{\partial (\rho e_0 u_i + p u_j)}{\partial x_j} = \frac{\partial (\tau_{ij} u_i - q_j)}{\partial x_j} \tag{3}$$

with

$$e_0 = e + \frac{1}{2} u_i u_j \tag{4}$$

In order to close the system there's the necessity to introduce two other equations: viscous stress tensor (τ_{ij}) and the heat flux (q_j):

$$\tau_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right) \tag{5}$$

$$q_j = -\frac{\mu c_p}{\sigma}\frac{\partial T}{\partial x_j} \tag{6}$$

where c_p is the specific heat and σ is the Prandtl number.
The equations describe every detail of the turbulent velocity field from the largest to the smallest length and time scales, so the amount of informations contained in the velocity field is vast, and as a consequence (in general) the direct approach of solving the Navier-Stokes equations is quite hard.
A powerful research tool for investigating simple turbulent flows at moderate Reynolds numbers is DNS. For the high-Reynolds-number flows that are prevalent in applications and interest, the natural alternative is to pursue a statistical approach. It consists in describing the turbulent flows, not in terms of the velocity, but in terms of some statistics as the mean velocity field. A model based on such statistics can lead to a tractable set of equations, because statistical fields vary smoothly in position and time.

2.1 TURBULENCE SIMULATIONS

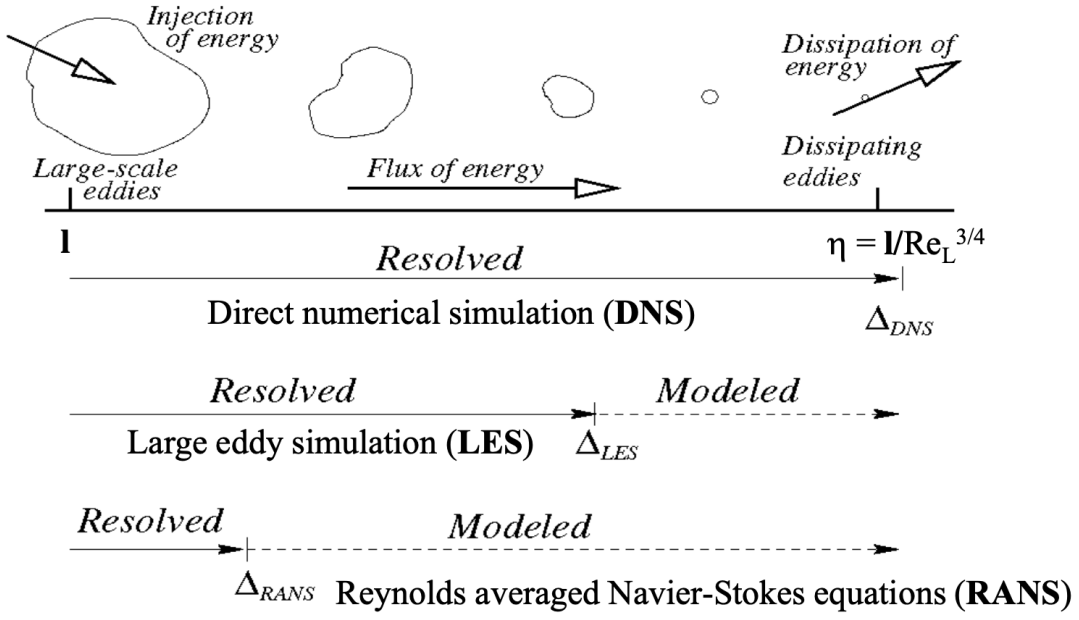


Figure 2.2: Turbulence simulation approaches [17]

2.1.1 Direct Numerical Simulation

Direct Numerical Simulation (DNS) consists in solving the Navier-Stokes equations, resolving all the scales of motion. Each simulation produces a single realization of the flow. Conceptually it is the simplest approach and, when it can be applied, it is the best in accuracy and in the level of description provided. However, it is important to understand that the cost is extremely high. The computer requirements increase so rapidly with Reynolds number so that the DNS approach was infeasible until the 1970s when computers of sufficient power became available (HPC).
In DNS the governing equations (Eq.1-3) are solved without using physical simplifications therein resolving the physical phenomena without modeling.

DNS requires a computational grid resolving the entire span from the Kolmogorov micro-scale η to the large size eddies, Λ. This presents an inherent multi scale problem where A. Johansson in [18] writes that the ratio of η the smallest to the largest scales of turbulence can be estimated as:

$$\frac{\Lambda}{\eta} \sim Re_{\Lambda}^{\frac{3}{4}} \tag{7}$$

For a grid spanning in three directions and resolved down to the Kolmogorov scale this would account in a grid-point-growth of $Re_{\Lambda}^{\frac{9}{4}}$, or taking temporal resolution into account Re_{Λ}^{3}.

2.1.2 Large Eddy Simulation

Large Eddy Simulation, or LES, is a model where the equations are filtered. This is done on the Navier-Stokes equations and the results are variables that depend on both space and time. Filtering of variables is a part of a hybrid method where large eddies are resolved and small sub-grid scale information is modeled. The modeling errors introduced are usually small. In fact, turbulence at the Kolmogorov scale is more isotropic and contains less energy than the large scale turbulence [18]. The small scales are usually modeled using the Boussinesq hypothesis (2.1.5). Although LES are less expensive than DNS, they still need to satisfy the y^+ requirement, leading to important computational costs.

2.1.3 Detached Eddy Simulation

Detached Eddy Simulation (DES) is a combination of Unsteady RANS (URANS) and LES. URANS is very similar to RANS since they both solve for the time-averaged flow but differs in the sense that URANS keeps the transient term [19]. DES models the boundary layer using RANS while the outer eddies are

resolved using LES. This method is very interesting also under a computational cost point of view because the y^+ of the LES simulation hasn't to be respected everywhere.

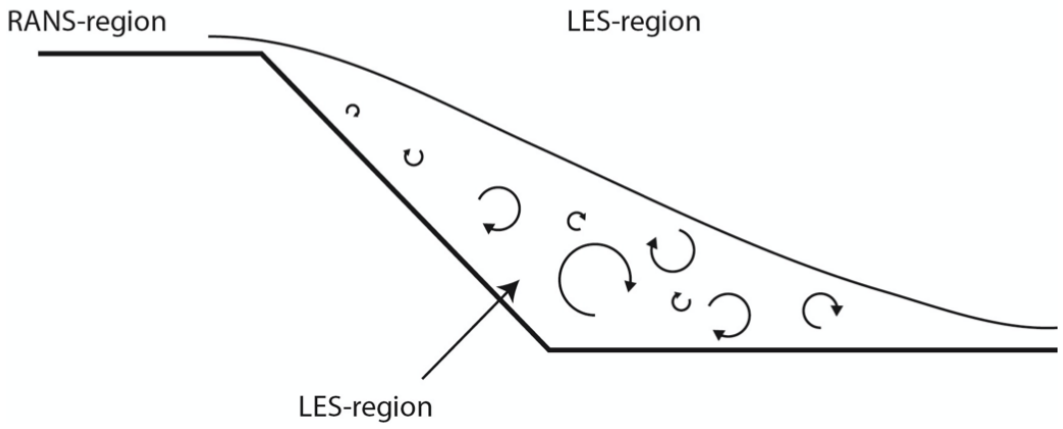


Figure 2.3: DES scheme [20]

2.1.4 Reynolds Averaged Navier Stokes

Reynolds averaged Navier-Stokes equations (RANS) are a time averaged version of the instantaneous Navier-Stokes equations. The velocity vector **u** and the pressure p are divided into a steady (time-averaged) and a fluctuating part to cut down on computational cost .

$$u_i = U_i + u_i^{'} \tag{8}$$
$$p = P + p^{'} \tag{9}$$

This decomposition is known as the Reynolds decomposition, with $U_i = \overline{u_i}$, $P = \overline{p}$ and where $\bar{\cdot}$ means averaged.

Incorporating the fluctuating velocity $u_i^{'}$ and pressure P components into the incompressible Navier-Stokes equation the resulting mean flow equation can after time averaging be written as:

$$\frac{\partial U_i}{\partial t} + U_j \frac{\partial U_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\nu \frac{\partial U_i}{\partial x_j} - \overline{u_i^{'} u_j^{'}}\right) \tag{10}$$

Finally, through Boussinesq hypothesis and adding the incompressibility equation, the RANS problem becomes:

$$\begin{cases} \nabla \cdot \mathbf{U} = 0 & in\ \ \Omega \\ (\mathbf{U} \cdot \nabla)\mathbf{U} + \nabla p = \nabla \cdot (2\nu_e \mathbb{D}(\mathbf{U})) & in\ \ \Omega \end{cases} \tag{11}$$

where

- $p = \frac{P}{\rho}$ is the ratio between pressure and density
- $\mathbf{U}$ is the flow speed
- $\mathbb{D}(\mathbf{U}) = \frac{1}{2}[\nabla\mathbf{U} + (\nabla\mathbf{U})^T]$ is the stress tensor
- $\nu_e = \nu + \nu_t$ is the effective viscosity, with ν that is the kinematic viscosity and ν_t the *turbulent viscosity*

The system of Eq.11 is solved for $\mathbf{u}$ and p but a new term $\overline{u_i' u_j'}$ is found on the right-hand-side (RHS) describing the relationship between fluctuating velocities[21]. This term is often referred to as the *Reynolds stress tensor* and introduces a closure problem where six additional unknown turbulent stresses arise.

2.1.5 Boussinesq Approximation

One of the early attempts of modeling the turbulent shear stress was made by the French 19th century scientist Boussinesq who described the *Reynolds stresses* using mean velocity gradients [18], as follows:

$$-\rho\overline{u_i' u_j'} = \rho\nu_t S_{ij} - \frac{2}{3}\rho K\delta ij \tag{12}$$

Here ν_t is the kinetic eddy viscosity, K the kinetic energy and S_{ij} the mean strain rate tensor:

$$K = \frac{1}{2}\overline{u_i' u_j'} \tag{13}$$

$$S_{ij} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) \tag{14}$$

Using the Boussinesq model the closure problem is reduced to model the eddy viscosity. Since the eddy viscosity is dominated by the length Λ and velocity V scale of the large turbulent eddies this assumption brings about a huge reduction in computational cost.

2.2 TURBULENCE MODEL FOR RANS

In order to close the set of equations a wide variety of turbulence models are available [16]:

- **Algebraic models/zero equation models:** Specific case and not very general, they work well for the scenario they were created for but need additional informations such as velocity gradients or geometry specifications.

- **One-equation models:** Like the name a one-equation model usually solves a transport equation for one turbulent variable like the *turbulent kinetic energy K*, or the *eddy viscosity* ν_t . Somewhat more general but case specific input is still required. An example of a common one-equation model is the Spalart-Allmaras model.

- **Two-equation models:** Two transport equations are solved for two different variables. Isotropic turbulence is generally assumed and no additional information is needed; hence the model is "completely formulated in terms of local quantities". Example of common two-equation models are $k-\epsilon$ and $k-\omega$.

2.2.1 Spalart-Allmaras

The Spalart-Allmaras (SA) model [22] is a one-equation model that solves for $\tilde{\nu}$, which is referred to as nuTilda in OpenFOAM but is often called the Spalart-Allmaras variable [23]. The SA model drops the last part in Eq.12 when solving for the eddy viscosity. The following relations are found in the SA model:

$$\nu_t = \tilde{\nu} f_{v1} \tag{15}$$

where the viscous damping functions are $f_{v1} = \frac{X^3}{X^3+C_{v1}^3}$ with $X = \frac{\tilde{\nu}}{\nu}$.
The transport equation is written as:

$$\frac{\partial \tilde{\nu}}{\partial t} + u_j \frac{\partial \tilde{\nu}}{\partial x_j} = c_{b1} S \tilde{\nu} + \frac{1}{\sigma}\left[\frac{\partial}{\partial x_j}\left((\nu+\tilde{\nu})\frac{\partial \tilde{\nu}}{\partial x_j}\right) + c_{b2}\frac{\partial \tilde{\nu}}{\partial x_j}\frac{\partial \tilde{\nu}}{\partial x_j}\right] - c_{w1} f_w \left(\frac{\tilde{\nu}}{d}\right)^2 \tag{16}$$

In the RHS of Eq.16 the first is the *Production* term, the second one is the *Diffusion* and the last one is the responsible of *Destruction*.
The vorticity term $\tilde{S}$ is modelled to keep its long-layer characteristics:

$$\tilde{S} = \sqrt{2\Omega_{ij}\Omega_{ij}} f_{v3} + \frac{\tilde{\nu}}{k^2 d^2} f_{v2} \tag{17}$$

where

$$\Omega_{ij} = \frac{1}{2}\left(\frac{\partial \tilde{u_i}}{\partial x_j} - \frac{\partial \tilde{u_j}}{\partial x_j}\right) \tag{18}$$

$$f_{v2} = 1 - \frac{x}{1+x f_{v1}} \tag{19}$$

$$f_{v3} = 1 \tag{20}$$

$$f_w = g\left[\frac{1+C_{w3}^6}{g^{6+C_{w3}^6}}\right]^{\frac{1}{6}} \tag{21}$$

$$g = r + C_{w2}(r^6 - r) \tag{22}$$

$$r = \frac{\tilde{\nu}}{\tilde{S}k^2d^2} \tag{23}$$

And the coefficients has the following default values:

σ	C_{b1}	C_{b2}	C_{w1}	C_{v1}
2/3	0.1355	0.622	$\frac{C_{b1}}{k^2} + \frac{1+C_{b2}}{\sigma}$	7.1

Following the recommendations in [19] for an adequate ratio of turbulent kinematic viscosity to kinematic viscosity ($\frac{\nu_t}{\nu} = 3 \div 5$), $nut(\tilde{\nu})$ is set to 7.45×10^{-5}.

2.2.2 Spalart-Allmaras DES

Detached Eddy Simulation (DES) was originally formulated for the Spalart-Allmaras model (2.2.1).

$$\frac{\partial \rho \tilde{\nu}_t}{\partial t} + \frac{\partial \rho u_j \tilde{\nu}_t}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\frac{\mu + \mu_t}{\sigma \tilde{\nu}_t}\frac{\partial \tilde{\nu}_t}{\partial x_j}\right) + \frac{C_{b2}\rho}{\sigma \tilde{\nu}_t}\frac{\partial \tilde{\nu}_t}{\partial x_j}\frac{\partial \tilde{\nu}_t}{\partial x_j} + P - \Psi \tag{24}$$

where

$$\nu_t = \tilde{\nu}_t f_i \tag{25}$$

$$P = C_{b1}\rho\left(\bar{s} + \frac{\tilde{\nu}_t}{\partial k^2 d^2 f_2}\right)\tilde{\nu}_t \tag{26}$$

$$\bar{s} = (2\overline{s_{ij}s_{ij}})^{\frac{1}{2}} \tag{27}$$

$$\Psi = C_{w1}\rho f_w\left(\frac{\tilde{\nu}_t}{d}\right)^2 \tag{28}$$

From the RANS SA model the distance, d, stems from the distance to the nearest wall, while in the DES model the distance d comes from the minimum of the cell length Δ and the turbulent length scale d. Hence

$$\tilde{d} = min(d, D_{des}\Delta) \tag{29}$$

This means that in the case where $d < C_{des}\Delta$, which would occur in the boundary layer, the DES model switches to RANS model [19].

3 AEROACOUSTICS STATE-OF-THE-ART

The compressible continuity, momentum, and energy equations describe the motion in a fluid. Hence, the solution to these equations, subjected to boundary conditions, will include not only convection and diffusion, but also acoustic wave propagation. The acoustic part of the solution can not, in general, be separated from the rest of the solution but in many cases, it is possible to view the flow and the acoustics as two different fields. Depending on the characteristics of the interaction between the flow and the acoustics, aeroacoustic problems can be divided into two main categories:

- **two-way coupled** case, if there is a mutual dependence of the flow and the acoustic (i.e. that energy is being transferred both from the flow to the acoustic and vice versa);
- **one-way coupled** case, if the acoustic part is dependent on the flow, but the flow is independent of the acoustic.

Most of the low Mach number flows exhibit a one-way coupling (e.g. vortex shedding around a cylinder at low Mach numbers), which is the basis for most aeroacoustic modeling. However, there are some flows that are exceptions. For instance, the ”booming noise” that happens on driving with an open sunroof.

In one-way coupling cases the problem can be divided in flow problem and acoustic problem with a way to transfer energy from the flow to the acoustic side. This splitting can be performed thanks to the physical difference between flow (e.g. turbulence) and acoustic (e.g. wave propagation) phenomena which is advantageous for an understanding and computational point of view. While considering the split equations, however, it is important to remember that flow and acoustics, in reality, are coupled. The “booming” noise mentioned above, for example, can not appear without the transfer of energy from the acoustics to the flow.

It’s possible to divide the flow in three main regions that can be divided looking at the acoustic wavelength [24]:

- Size of an acoustic wavelength: it’s called **flow region** and it’s dominated by hydrodynamic phenomena. Even if there are some acoustic waves in this region the pressure fluctuations are due to turbulence or larger unsteady features as separations. Here the energy of the acoustic fields is about 1% of the total energy so hydrodynamic pressure fluctuations dominate.

- Many acoustic wavelengths out: it's called **far field** and its mean flow is typically homogeneous so turbulence can be neglected. The only phenomena to be considered here are the acoustic waves propagation and so there's no generation of sound.

- **near field**: is the overlapping area between the other two and here both hydrodynamics and acoustics phenomena are important.

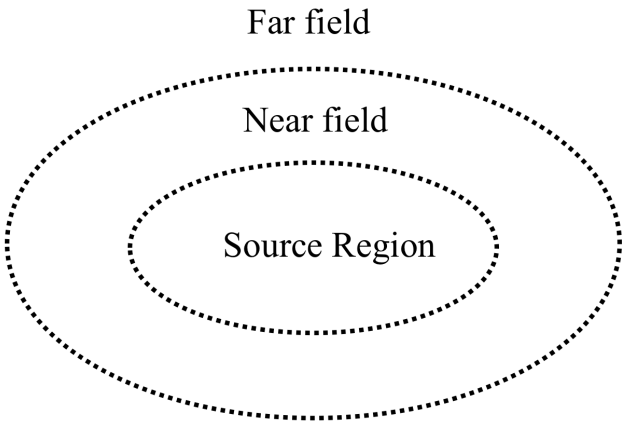


Figure 3.1: Regions in CAA [25]

Focusing on the sound generation and its propagation various methods can be used varying in computational effort and accuracy. Two main numerical Computational Aero Acoustic (CAA) approaches can be distinguished:

- *Direct Methods*: A transient solution is calculated where the source and its propagation is resolved out to the far field. In DNS the interest is on the flow but aeroacoustics studies the propagation too and the problem becomes a big multiscale problem. This method, in fact, requires a very fine grid for the spatial and temporal resolution which places stringent demands on computational resources. X. Gloerfelt in [26] writes that direct methods can only predict good acoustic results for simple configurations at moderate Re numbers.

- *Hybrid methods*: A CFD method is used to solve the flow in the source region, up to the near field region. Then, a transport problem is solved in order to compute the pressure fluctuations in the far field region. Hybrid methods significantly reduce the computational demand since only the near field needs to be spatially and temporally resolved.

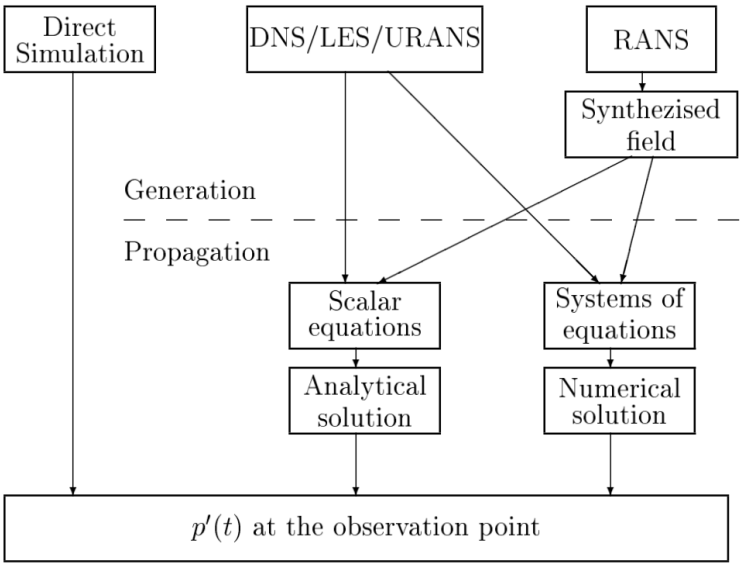


Figure 3.2: Overview of CAA method [24]

Since the sound is generated in the flow region and propagated through the far field, hybrid methods decouple the flow generation from the acoustic propagation in the far field; consequently allowing for methods adapted for the various regions.

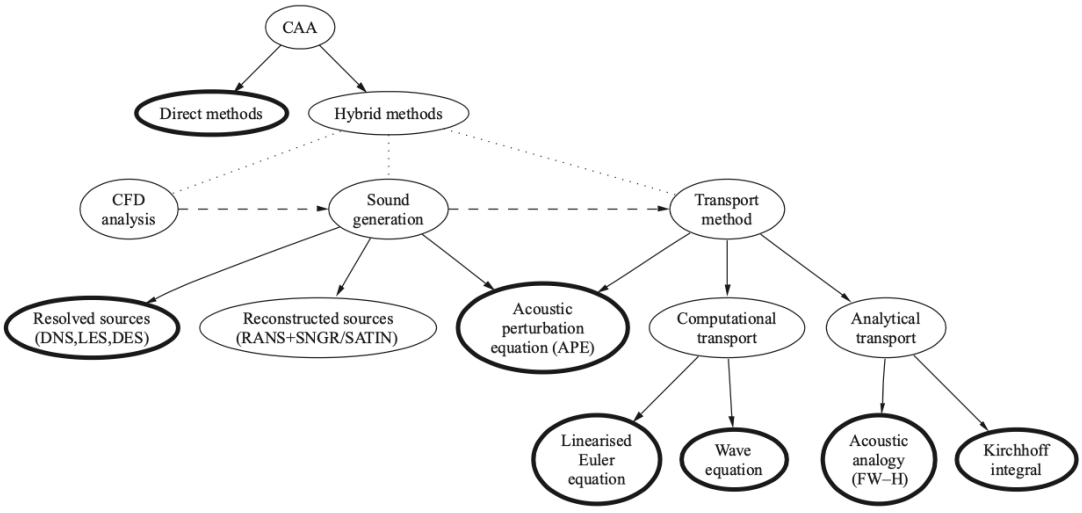


Figure 3.3: Different CAA method [27]

Finally, in Figure 3.3 there is a schematic review of several possible CAA methods; in section 3.1 there is a brief explanation of how the flow can be simulated while in section 3.2 the focus will be only on the main Hybrid methods. For further informations refer to [28] and [29] in addition to the already cited sources.

3.1 LARGE SCALE SIMULATION

A Direct Simulation (DS) solves the Navier-Stokes equations without modeling, and hence solves for the acoustic field as well as for the turbulence. If traditional DNS, which solves for the turbulence but not for the acoustics, is considered a subset of DS, one could imagine another subset that solves for the acoustics, but not for the turbulence. This subset lacks a descriptive name, but *Mankbadi et al* in [30], called it **Large Scale Simulations** (LSS). All methodologies that compute the sound field directly but that don't resolve all of the turbulence, can be considered as LSS. A compressible Unsteady RANS (URANS), for example, where the pressure fluctuations are recorded during the simulation and taken as the sound field, would belong to the LSS category of methodologies.

The main strength of LSS is, of course, that the computational cost is considerably smaller than in DS, while still computing the sound field directly. Since the small scales of the turbulence, believed to have only a very modest influence on the radiated sound, LSS is a promising approach.

3.2 HYBRID METHODS

James Lighthill laid the groundwork for the models of sound generation and is today considered the father of aeroacoustics [31]. To reduce the generated sound from jet engines, Lighthill developed analogies uncoupling the sound field from the source field. He used the fundamental equations (Eq.1, Eq.2, Eq.3) of fluid dynamics to model the source field as an inhomogeneous wave equation.

3.2.1 Lighthill's Acoustic Analogy

Differentiating the continuity equation (Eq.1) with respect to time, and the momentum equation (Eq.2) with respect to space, while subtracting the latter from the differentiated continuity equation one gets:

$$\frac{\partial^2 \rho}{\partial t^2} - \frac{\partial^2 \rho u_i u_j}{\partial x_i \partial x_j} = \frac{\partial^2 p}{\partial {x_i}^2} - \frac{\partial^2 \tau_{ij}}{\partial x_i \partial x_j} \tag{30}$$

If we subtract $c_0^2 \frac{\partial^2 \rho}{\partial {x_i}^2}$ from Eq.30 it can be re-formulated as Lighthill's wave equation:

$$\frac{\partial^2 \rho}{\partial t^2} - c_0^2 \frac{\partial^2 \rho}{\partial {x_i}^2} = \frac{\partial^2 T_{ij}}{\partial x_i \partial x_j} \tag{31}$$

where T_{ij} is defined as:

$$T_{ij} = \rho u_i u_j - \tau_{ij} + (p - c_0^2 \rho)\delta_{ij} \tag{32}$$

with

- $\rho u_i u_j$: non-linear convective forces described by the *Reynolds stress tensor*
- τ_{ij}: viscous forces
- $(p - c_0^2 \rho)$: deviation from a uniform sound velocity c_0 or the deviation from an isentropic behaviour

A distinction between the sound sources and the propagation of sound sources has been made but no assumptions so Eq.31 is exact. The left hand side is an ordinary wave operator whereas the right hand side is the acoustic source term. For Lighthill's equation to be applicable the right hand side should be known as well as decoupled from the acoustic field. By comparing the magnitude of the three terms in the stress tensor Lighthill deduced that the momentum flux tensor $\rho u_i u_j$ is the only significant contributor to sound production T_{ij} for cold jets. The acoustic wave equation can be solved analytically if the right hand side is assumed known [24].
A common approach is to integrate the sources using a free field Green function.

$$\rho(\mathbf{x}, t) - \rho_0 = \frac{1}{4\pi c_0^2} \int_\infty \frac{1}{r} \frac{\partial^2 T_{ij}}{\partial y_i \partial y_j} \partial V(\mathbf{y}) \tag{33}$$

where $\tau = t - \frac{r}{c}$, $T_{ij} = T_{ij}(\mathbf{y}, t)$ and $r = |\mathbf{x} - \mathbf{y}|$.

3.2.2 Curle's Analogy

Lighthill's theory was further extended by Curle to incorporate the presence of solid boundaries upon the aerodynamic sound. Curle's approach was to find a solution to the inhomogeneous wave equation where the double divergence of Eq.31 can be taken outside the integral sign. The derivation is carried out in a similar way as Lighthill's original work [31] but with two additional steps. To account for solid boundaries a surface integral is added through the Kirchoff-Helmholtz formula [32], then a coordinates transformation from source to observer ones is done. Starting with the general solution of the inhomogeneous wave equation previously mentioned but this time on a bounded domain

$$\begin{aligned} \rho(\mathbf{x}, t) - \rho_0 = & \frac{1}{4\pi c_0^2} \int_V \frac{1}{r} \frac{\partial^2 T_{ij}}{\partial y_i \partial y_j} \partial V(\mathbf{y}) \\ & - \frac{1}{4\pi} \int_S \left(\frac{1}{r} \frac{\partial \rho}{\partial n} + \frac{1}{r^2} \frac{\partial r}{\partial n} \rho + \frac{1}{c_0 r} \frac{\partial r}{\partial n} \frac{\partial \rho}{\partial t} \right) \partial S(\mathbf{y}) \end{aligned} \tag{34}$$

Using partial integration and symmetrical properties from the Green's function G_0 [33]:

$$\frac{\partial G_0}{\partial x_i} = \frac{\partial G_0}{\partial y_i} \qquad \frac{\partial G_0}{\partial \tau} = -\frac{\partial G_0}{\partial t} \tag{35}$$

it becomes:

$$\begin{aligned} \rho(\mathbf{x},t) - \rho_0 = & \frac{1}{4\pi c_0^2}\frac{\partial^2}{\partial x_i \partial x_j}\int_V \frac{T_{ij}}{r} dV(\mathbf{y}) \\ & -\frac{1}{4\pi c_0^2}\frac{\partial}{\partial x_i}\int_S \frac{n_j}{r}[\rho u_i u_j + p\delta_{ij} - \tau_{ij}]dS(\mathbf{y}) \\ & +\frac{1}{4\pi c_0^2}\frac{\partial}{\partial t}\int_S \frac{\rho u_i n_i}{r} dS(\mathbf{y}) \end{aligned} \tag{36}$$

The second surface integral in Eq.36 characterizes the monopole field created by fluid vibrations of the body which in many cases can be neglected. So in the case of solid surfaces where the velocity on the surface is zero Curle's final equation reads:

$$\begin{aligned} \rho(\mathbf{x},t) - \rho_0 = & \frac{1}{4\pi c_0^2}\frac{\partial^2}{\partial x_i \partial x_j}\int_V \frac{T_{ij}}{r} dV(\mathbf{y}) \\ & -\frac{1}{4\pi c_0^2}\frac{\partial}{\partial x_i}\int_S \frac{n_j}{r}(p\delta_{ij} - \tau_{ij})dS(\mathbf{y}) \end{aligned} \tag{37}$$

3.2.3 Ffowcs-Williams and Hawking Analogy

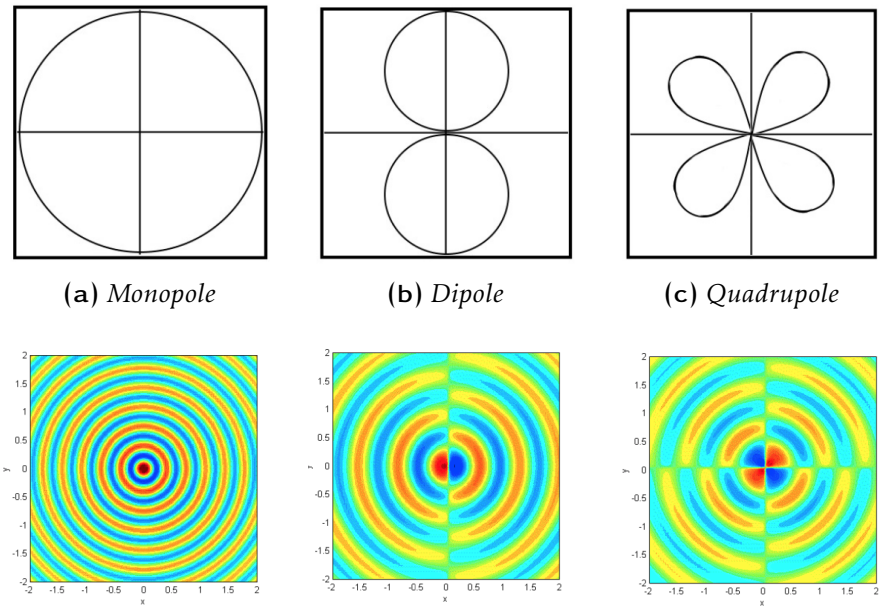
Ffowcs-Williams and Hawkings (FW-H) extended the work that Curle had published by taking into account the sound generation from arbitrary motion of a body in a turbulent flow. FW-H equation is a generalization of Curle's analogy where the governing equations are rewritten in such a way that the source terms will account for boundary effects and the result is an equation valid for a continuous infinite space.

$$\begin{aligned} \rho(\mathbf{x},t) - \rho_0 = & \frac{1}{4\pi c_0^2}\frac{\partial^2}{\partial x_i \partial x_j}\int_V \frac{T_{ij}^{'*}}{r(1-\frac{l_j v_j}{c_0})} dV(\mathbf{y}^*) \\ & -\frac{1}{4\pi c_0^2}\frac{\partial}{\partial x_i}\int_S \frac{F_j^*}{r(1-\frac{l_j v_j}{c_0})} dS(\mathbf{y}^*) \\ & +\frac{1}{4\pi c_0^2}\frac{\partial}{\partial t}\int_S \frac{Q^*}{r(1-\frac{l_j v_j}{c_0})} dS(\mathbf{y}^*) \end{aligned} \tag{38}$$

where the source terms in a moving reference frame are:

$$\begin{aligned} T_{ij}^{'*} &= \rho(u_i^* + v_i)(u_j^* + v_j) - \tau_{ij}^* + (p - c_0^2(\rho - \rho_0))\delta_{ij} \\ F_i^* &= (\rho(u_i^* + v_i)u_j^* + p\delta ij - \tau ij^*)n_j \\ Q^* &= (\rho_0 v_i + \rho u_i^*)n_j \end{aligned} \tag{39}$$

The three source terms $T^{'*}, F^*, Q^*$ are associated with *quadrupole*, *dipole*, and *monopole* source mechanisms respectively. The quadrupole source mechanism is due to fluctuating stresses in the fluid (unsteady Reynolds stresses) while dipole sources are created by external unsteady forces, or fluid pressure on a solid boundary and monopole sources are due to volume flow or fluctuating mass injection.



(a) *Monopole* (b) *Dipole* (c) *Quadrupole*

Figure 3.4: The most prominent sources within a fluid flow and the propagation pattern [34][35]

In the case of an impermeable surface simplifications can be made; focusing on the dipole term which is of most importance to this work, u_i^* is by definition equal to zero on the surface, and F_i^* can therefore be written as:

$$F_i^* = (p\delta_{ij} - \tau_{ij}^*)n_j \tag{40}$$

Numerical implementation of Eq.38 can be difficult due to the combination of spatial and temporal derivatives with respect to the observer frame of reference. Later formulations by Bretner and Farassat [36] resolve the Ffowcs-Williams and Hawkings analogy in the frequency domain to circumvent the problem of emission time. With the implementation of the FW-H analogy isotropic

wave propagation is considered, and can hence only be expected to provide good results in flows with zero or low mean motion. To incorporate mean flow the convected FW-H analogy can be used where *Gloerfelt et al.* introduces the concept of having the observer move with the mean flow. The derivation can be found in [26] and the convected FW-H equation for the frequency domain is written as following:

$$\begin{aligned}&\left\{\frac{\partial^2}{\partial {x_i}^2}+k^2-2iM_ik\frac{\partial}{\partial x_i}-M_iM_j\frac{\partial^2}{\partial x_i\partial x_j}\right\}[\mathbf{H}(f)c_0^2\rho^{'}(\mathbf{x},\omega)]\\ =&-\frac{\partial^2}{\partial x_i\partial x_j}[T_{ij}(\mathbf{x},\omega)\mathbf{H}(f)]-\frac{\partial^2}{\partial {x_i}^2}[F_i(\mathbf{x},\omega)\delta(f)]-i\omega Q(x,\omega)\delta(f)\end{aligned} \tag{41}$$

From the integral solution in [26] the source terms can be rewritten as:

$$\begin{aligned}T_{ij}=&\rho(u_i-U_i^\infty)(u_j-U_j^\infty)+(p-c_0^2\rho)\delta_{ij}-\tau_{ij}\\ F_i=&-[\rho(u_{i-2U_i^\infty})u_j+p\delta_{ij}-\tau_{ij}^*]n_j\\ Q=&\rho u_iu_j\end{aligned} \tag{42}$$

It should be noted that here, instead of using the free-space Green's function, the convective Green's function should be used.

3.3 PROUDMAN NOISE SOURCE MODEL

To estimate the level of acoustic noise within a turbulent area Proudman based his work [37] on the methods developed by Lighthill combined with techniques from the statistical theory of isotropic turbulence. He also proposed that the turbulent eddies generating noise are mainly the ones that do not dissipate energy. The noise source model derived defines the Intensity of sound

$$I(\mathbf{x},t)=\frac{c_0^3}{\rho_0}\overline{(\rho-\overline{\rho})^2} \tag{43}$$

where overbars denote mean values.

The Acoustic Power per unit volume is derived from the Acoustic Intensity after that the Lighthill's formula (Eq.33) has been substituted in and can be written as:

$$AP=\alpha\rho_0\frac{u_{rms}^3}{l}\frac{u_{rms}^5}{c_0^5} \tag{44}$$

where α was proposed to be approximately 38.

In OpenFoam it is implemented in `proudmanAcousticPower` post process function as:

$$PA = \alpha_\epsilon \rho \epsilon M_t^5 \tag{45}$$

where $\alpha_\epsilon = 0.1$, $M_t = \frac{\sqrt{2k}}{c_0}$.

The dB output instead is:

$$LP = 10 \log \frac{PA}{Pref} \tag{46}$$

where $P_{ref} = 1 \times e^{-12} [W/m^3]$ is a constant.

4 ADJOINT METHOD BACKGROUND

The use of CFD tools in aerodynamic design optimization has grown in importance within the last decade. In gradient-based optimization techniques, the goal is to minimize a suitable cost or objective function (drag coefficient, deviation from a prescribed surface pressure distribution, etc.) with respect to a set of design variables (defining, for example, an airfoil profile or aircraft surface). Minimization is achieved by means of an iterative process which requires the computation of the gradients or sensitivity derivatives of the cost function with respect to the design variables. Gradients can be computed in a variety of ways, the most actively pursued recently being adjoint methods. These allow the solution of general sensitivity analysis problems governed by fluid dynamics models ranging from the full potential equation to the full compressible Reynolds-averaged Navier-Stokes equations.

4.1 ADJOINT EQUATIONS

The mathematical formulation of this problem is part of the Operative non linear Research problems; in fact there is a certain objective function $F(\beta, \mathbf{u}, p)$ which has to be optimized but under the limit that the RANS equations (considered as R) have to be satisfied. The mathematical problem can be formulated as:

$$\begin{aligned} &minimize && F(\beta, u, p) \quad w.r.t.\ \beta \\ &s.t && R(\beta, \mathbf{u}, p) = 0 \end{aligned} \tag{47}$$

To simplify the problem let's consider the case with displacements parallel to the local normal direction

$$\mathbf{b} = \beta \mathbf{n} \tag{48}$$

where $\mathbf{n}$ is the local normal.

Adjoint methods are conventionally divided into continuous and discrete. In the former, the adjoint equations are derived from the governing PDEs and then subsequently discretized. In the latter, the adjoint equations are directly derived from the discrete governing equations. While, the discrete adjoint method should give gradients which are closer in value to exact finite-difference gradients, the continuous adjoint method has the advantage that the adjoint system

has a unique form independent of the scheme used to solve the flow-field system. Numerical studies have shown that in typical shape optimization problems in transonic flow the differences are small enough that they have no significant effect on the final result [38].

4.1.1 Objective function

The continuous adjoint method, including the adjoint wall functions technique, for objective functions defined as integrals over the boundary S of the computational domain is presented in detail in [39], while here the focus will be on functions defined as integral over the computational domain Ω. A general objective function F consisting of a volume integral can be expressed as:

$$F = \int_{\Omega} F_{\Omega} d\Omega \tag{49}$$

Differentiating F with respect to the N design variables b_n, after taking into consideration the Leibniz theorem for volume integrals with variable boundaries, yields

$$\frac{\partial F}{\partial b_n} = \int_{\Omega} \frac{\partial F_{\Omega}}{\partial b_n} d\Omega + \int_{S} F_{\Omega} n_k \frac{\delta x_k}{\delta b_n} dS \tag{50}$$

In Eq.50, the symbol $\frac{\delta(.)}{\delta b_n}$ denotes the total (or material) derivative with respect to the design variables and represents the total change caused by variations of b_n. The partial derivative $\frac{\partial(.)}{\partial b_n}$ represents the variation caused purely due to changes in the flow variables (in turn, caused by the geometry deformation) without considering space deformations.
The partial and total derivatives of an arbitrary variable Φ are linked through:

$$\frac{\delta \Phi}{\delta b_n} = \frac{\partial \Phi}{\partial b_n} + \frac{\Phi}{x_k} \frac{\delta x_k}{\delta b_n} \tag{51}$$

It is possible to expand Eq.50 by taking into consideration the dependency of F on the flow variables:

$$\begin{aligned} \frac{\delta F}{\delta b_n} = & \int_{\Omega} \acute{F}^{v}_{\Omega,i} \frac{\partial v_i}{\partial b_n} d\Omega + \int_{\Omega} \acute{F}^{p}_{\Omega} \frac{\partial p}{\partial b_n} d\Omega + \int_{\Omega} \acute{F}^{\tilde{\nu}}_{\Omega} \frac{\partial \tilde{\nu}}{\partial b_n} d\Omega \\ & + \int_{S} F_{\Omega} n_k \frac{\delta x_k}{\delta b_n} dS + \int_{S} \acute{F}^{v}_{S,i} \frac{\partial v_i}{\partial b_n} dS + \int_{S} \acute{F}^{p}_{S} \frac{\partial p}{\partial b_n} dS + \int_{S} \acute{F}^{\tilde{\nu}}_{S} \frac{\partial \tilde{\nu}}{\partial b_n} dS \end{aligned} \tag{52}$$

where $\acute{F}^{\Phi}_{\Omega}$ includes the partial derivative $\frac{\partial F_{\Omega}}{\partial \Phi}$ plus any term that might result from the use of the *Gauss divergence theorem* for integrals of the form $\int_{\Omega} \frac{\partial}{\partial b_n} (\frac{\partial \Phi}{\partial x_j}) d\Omega$. The last three integrals on the right hand side of Eq.52 are non zero only if F_{Ω} includes differential operators of the flow variables. It should

be noted that since F might depend on $\tilde{\nu}$ the equation includes also integrals containing $\frac{\partial \tilde{\nu}}{\partial b_n}$.

When dealing with turbulent flow problems, the state equations comprise the mean-flow and turbulence model equations. Computing accurate sensitivity derivatives requires the differentiation of all state PDEs. However, in the majority of articles based on the continuous adjoint method, it is a common practice to avoid the differentiation of turbulence models [40],[41], [42], [43]. This simplification is often referred to as the "frozen-turbulence" assumption. The differentiation of the turbulence model equations using the continuous adjoint method was initially addressed in [44], for the low turbulence Reynolds number variant of the Spalart–Allmaras model for incompressible flows. Later on, the continuous adjoint approach with the same turbulence model for compressible flows was also presented in [45].

4.1.2 Continuous adjoint method development

The starting point for the derivation of the adjoint equations is the augmented objective function, defined by expanding F with the volume integral of the state equations multiplied by the corresponding adjoint variables:

$$F_{aug} = F + \int_{\Omega} u_i R_i^v d\Omega + \int_{\Omega} q R^p d\Omega + \int_{\Omega} \tilde{\nu}_a R^{\tilde{\nu}} d\Omega + \int_{\Omega} \Delta_a R^{\Delta} d\Omega \tag{53}$$

where u_i is the adjoint velocity component, q is the adjoint pressure and $\tilde{\nu}_a$ is the adjoint to $\tilde{\nu}$. The latter is neglected in the commonly used "frozen turbulence" assumption. In addition, following [45] and [46], Eq.53 also includes the adjoint to the distance field Δ_a.

The last integral is included in order to make the final sensitivity expression independent of variations in the distance field. Differentiating this equation with respect to b_n and applying the *Leibnitz theorem*:

$$\begin{aligned}\frac{\delta F_{aug}}{\delta b_n} =& \frac{\delta F}{\delta b_n} + \int_{\Omega} u_i \frac{\partial R_i^v}{\partial b_n} d\Omega + \int_{\Omega} q \frac{\partial R^p}{\partial b_n} d\Omega + \int_{\Omega} \tilde{\nu}_a \frac{\partial R^{\tilde{\nu}}}{\partial b_n} d\Omega + \int_{\Omega} \Delta_a \frac{\partial R^{\Delta}}{\partial b_n} d\Omega \\ &+ \int_{S_w} (u_i R_i^v + q R^p + \tilde{\nu}_a R^{\tilde{\nu}} + \Delta_a R^{\Delta}) n_k \frac{\delta x_k}{\delta b_n} dS\end{aligned} \tag{54}$$

where:

- $R^p = -\frac{\partial v_j}{\partial x_j} = 0$

- $R_i^v = v_j \frac{\partial v_i}{\partial x_j} - \frac{\partial \tau_{ij}}{x_j} + \frac{\partial p}{\partial x_i} = 0$
  with $i = 1, 2(, 3)$ and $\tau_{ij} = (\nu + \nu_t)(\frac{\partial v_i}{\partial v_j} + \frac{\partial v_j}{x_i})$ that is the *stress tensor*

- $R^{\tilde{\nu}} = v_j \frac{\partial \tilde{\nu}}{\partial x_j} - \frac{\partial}{\partial x_j}\left[\left(\nu + \frac{\tilde{\nu}}{\sigma}\right)\frac{\partial \tilde{\nu}}{\partial x_j}\right] - \frac{c_{b2}}{\sigma}\left(\frac{\tilde{\nu}}{\partial x_j}\right)^2 - \tilde{\nu}P + \tilde{\nu}D = 0$
  with $\tilde{\nu}$ is the turbulence state variable (see section 2.2.1).

- $R^{\Delta} = \frac{\partial (c_j \Delta)}{\partial x_j} - \Delta \frac{\partial^2 \Delta}{\partial {x_j}^2} = 0$
  it is the so-called Hamilton-Jacobi equation where $c_j = \frac{\partial \Delta}{\partial x_j}$, is solved to provide the distance to the wall field (Δ) at all interior cell-centers.

After some passages that can be found in [47], it is possible to obtain the following expression for the gradient of F:

$$\begin{aligned}\frac{\delta F_{aug}}{\delta b_n} = &\int_S \mathcal{BC}_i^u \frac{\partial v_i}{\partial b_n} dS + \int_S (u_j n_j + \acute{F}_S^p)\frac{\partial p}{\partial b_n} dS + \int_S \mathcal{BC}^{\tilde{\nu}_a}\frac{\partial \tilde{\nu}}{\partial b_n} dS \\ &+ \int_S 2\Delta_a \frac{\partial \Delta}{\partial x_j} n_j \frac{\partial \Delta}{\partial b_n} dS - \int_S u_i n_j \frac{\partial \tau_{ij}}{\partial b_n} dS - \int_S \tilde{\nu_a}\left(\nu + \frac{\tilde{\nu}}{\sigma}\right)\frac{\partial}{\partial b_n}\left(\frac{\partial \tilde{\nu}}{\partial x_j}\right) n_j dS \\ &+ \int_\Omega R_i^u \frac{\partial v_i}{\partial b_n} d\Omega + \int_\Omega R^q \frac{\partial p}{\partial b_n} d\Omega + \int_\Omega R^{\tilde{\nu}_a}\frac{\partial \tilde{\nu}}{\partial b_n} d\Omega + \int_\Omega R^{\Delta_a}\frac{\partial \Delta}{\partial b_n} d\Omega \\ &+ \int_{S_w} (u_i R_i^v + qR^p + \tilde{\nu}_a R^{\tilde{\nu}} + \Delta_a R^{\Delta} + F_\Omega)\frac{\delta x_k}{\delta b_n} n_k dS\end{aligned} \tag{55}$$

where

$$\mathcal{BC}_i^u = u_i v_j n_j + \tau_{ij}^a n_j - q n_i + \tilde{\nu_a}\tilde{\nu}\frac{C_Y}{Y} e_{mjk}\frac{\partial v_k}{x_j} e_{mli} n_l + \acute{F}_{S,i}^v \tag{56}$$

$$\mathcal{BC}^{\tilde{\nu}_a} = \tilde{\nu}_a v_j n_j + (\nu + \frac{\tilde{\nu}}{\sigma})\frac{\partial \tilde{\nu_a}}{\partial x_j} n_j - \frac{\tilde{\nu_a}}{\sigma}(1 + 2c_{b2})\frac{\partial \tilde{\nu}}{\partial x_j} n_j + \acute{F}_S^{\tilde{\nu}} \tag{57}$$

and $\tau_{ij}^a = (\nu + \nu_t)(\frac{\partial u_i}{\partial v_j} + \frac{\partial u_j}{x_i})$ are the components of the *adjoint stress tensor*.

Setting to zero the multipliers of $\frac{\partial v_i}{\partial b_n}, \frac{\partial p}{\partial b_n}, \frac{\partial \tilde{\nu}}{\partial b_n}, \frac{\partial \Delta}{\partial b_n}$ in eq.55 we obtain:

$$R^q = -\frac{\partial u_j}{\partial x_j} + \acute{F}^p_\Omega = 0 \tag{58}$$

$$\begin{aligned} R^u_i &= u_j \frac{\partial v_j}{\partial x_i} - \frac{\partial v_j u_i}{\partial x_j} - \frac{\partial \tau^a_{ij}}{\partial x_j} + \frac{\partial q}{\partial x_i} + \acute{F}^v_{\Omega,i} \\ &+ \underbrace{\tilde{v}_a \frac{\partial \tilde{v}}{\partial x_i} - \frac{\partial}{\partial x_l}(\tilde{v}_a \tilde{v} \frac{C_Y}{Y} e_{mjk} \frac{\partial v_k}{\partial x_j} e_{mli})}_{A1} = 0 \end{aligned} \tag{59}$$

$$\begin{aligned} R^{\tilde{v}_a} &= -\frac{\partial v_{j\tilde{v}_a}}{\partial x_j} - \frac{\partial}{\partial x_j}\left[\left(\nu + \frac{\tilde{v}}{\sigma}\right)\frac{\partial \tilde{v}_a}{\partial x_j}\right] + \frac{1}{\sigma}\frac{\partial \tilde{v}_a}{\partial x_j}\frac{\tilde{v}}{\partial x_j} \\ &+ 2\frac{C_{b2}}{\sigma}\frac{\partial}{\partial x_j}\left(\tilde{v}_a \frac{\partial \tilde{v}}{\partial x_j}\right) + \tilde{v}_a \tilde{v} \mathcal{C}_{\tilde{v}} + \underbrace{\frac{\partial \nu_t}{\partial \tilde{v}}\frac{\partial u_i}{\partial x_j}\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i}\right)}_{P_a} \\ &+ (-P + D)\tilde{v}_a + \acute{F}^{\tilde{v}}_\Omega = 0 \end{aligned} \tag{60}$$

$$R^{\Delta_a} = -2\frac{\partial}{\partial x_j}\left(\Delta_a \frac{\partial \Delta}{\partial x_j}\right) + \tilde{v}\tilde{v}_a \mathcal{C}_\Delta = 0 \tag{61}$$

The adjoint momentum equation (Eq.59) includes terms resulting from the differentiation of the turbulence model equation marked as term *A1*. These terms depend on the adjoint turbulence variable $\tilde{v}_a$, computed by solving the adjoint turbulence model PDE (Eq.60). The latter is coupled to (Eq.59) through the adjoint turbulence production term marked as P_a, while the last term in Eq.60 summarizes the contribution of the objective function differentiation to the adjoint turbulence model. In the end Eq.61 is the adjoint Hamilton–Jacobi equation which is solved at a post-processing step, after solving the coupled system of Eq.58-61. [39]

After satisfying the adjoint equations (Eq.58, 59, 60, 61):

$$\begin{aligned} \frac{\delta F_{aug}}{\delta b_n} &= \int_S \mathcal{BC}^u_i \frac{\partial v_i}{\partial b_n} dS + \int_S (u_j n_j + \acute{F}^p_S)\frac{\partial p}{\partial b_n} dS + \int_S \mathcal{BC}^{\tilde{v}_a}\frac{\partial \tilde{v}}{\partial b_n} dS \\ &+ \int_S 2\Delta_a \frac{\partial \Delta}{\partial x_j} n_j \frac{\partial \Delta}{\partial b_n} dS - \int_S u_i n_j \frac{\partial \tau_{ij}}{\partial b_n} dS - \int_S \tilde{v}_a \left(\nu + \frac{\tilde{v}}{\sigma}\right)\frac{\partial}{\partial b_n}\left(\frac{\partial \tilde{v}}{\partial x_j}\right) n_j dS \\ &+ \int_{S_{wp}} (u_i R^v_i + q R^p + \tilde{v}_a R^{\tilde{v}} + \Delta_a R^\Delta + F_\Omega)\frac{\delta x_k}{\delta b_n} n_k dS \end{aligned} \tag{62}$$

4.1.3 Sensitivity derivatives

Once the field adjoint equations and their boundary conditions are satisfied, the remaining terms concern the sensitivity derivatives expression:

$$\begin{aligned}\frac{\delta F_{aug}}{\delta b_n} = T_{SD}^{WF} &+ \int_{S_{Wp}} u^I_{\langle t\rangle}\frac{\partial \tau_{ij}}{\partial x_k} n_j t_i^I \frac{\delta x_k}{\delta b_n} dS + \int_{S_{Wp}} u^I_{\langle t\rangle}\tau_{ij}\frac{\delta (n_j t_i^I)}{\delta b_n}\frac{\delta x_k}{\delta b_n} dS \\ &+ \int_{S_{Wp}} \mathcal{SD}_i v^I_{\langle t\rangle}\frac{\delta t_i^I}{\delta b_n} dS - \int_{S_{Wp}} \mathcal{SD}_i \frac{\partial v_i}{\partial x_k}\frac{\delta x_k}{\delta b_n} dS \\ &- \int_{S_{Wp}} \left[\nu \frac{\partial \tilde{\nu}_a}{\partial x_j} n_j + \acute{F}_S^{\tilde{\nu}}\right]\frac{\partial \tilde{\nu}}{\partial x_k}\frac{\delta x_k}{\delta b_n} dS - \int_{S_{Wp}} 2\Delta_a \frac{\partial \Delta}{\partial x_j} n_j \frac{\partial \Delta}{\partial x_k}\frac{\delta x_k}{\delta b_n} dS \\ &+ \int_{S_{Wp}} u_{\langle n\rangle}\left(\tau_{ij}\frac{\delta (n_i n_j)}{\delta b_n} + \frac{\partial \tau_{ij}}{\partial x_k}\frac{\delta x_k}{\delta b_n} n_i n_j\right) dS \\ &+ \int_{S_{Wp}} (u_i R_i^v + q R^p + \tilde{\nu}_a R^{\tilde{\nu}} + \Delta_a R^{\Delta} + F_\Omega)\frac{\delta x_k}{\delta b_n} n_k dS\end{aligned} \tag{63}$$

where

- $\mathcal{SD}_i = \tau_{ij}^a - q n_i + \acute{F}^v_{S_{Wp,i}}$

- T_{SD}^{WF} results from the differentiation of the law of the wall and the resulting correlation between the variation of the *friction velocity*, the *velocity magnitude* and distance of the first cell off the wall.

4.2 ADJOINT IMPLEMENTATION

The OpenFOAM executable `adjointOptimisationFoam` performs an optimization cycle whose main steps can be summarized in Fig.4.1 and they will be further explained in the following paragraph. The first step consists in defining the design variables and so running the *primal* solver in order to solve the flow equations. The objective (or cost) function value is calculated for each step of the *primal* solver, the values are averaged and subsequently is time for the adjoint equations to be solved. Once the adjoint equations are solved, the distance sensitivities are calculated and, basing on them, the design variables and the mesh are updated.
Obviously the executable can be set also to do not perform an optimization loop but only a single run in order to obtain just the *maps of sensitivities* without moving the mesh.

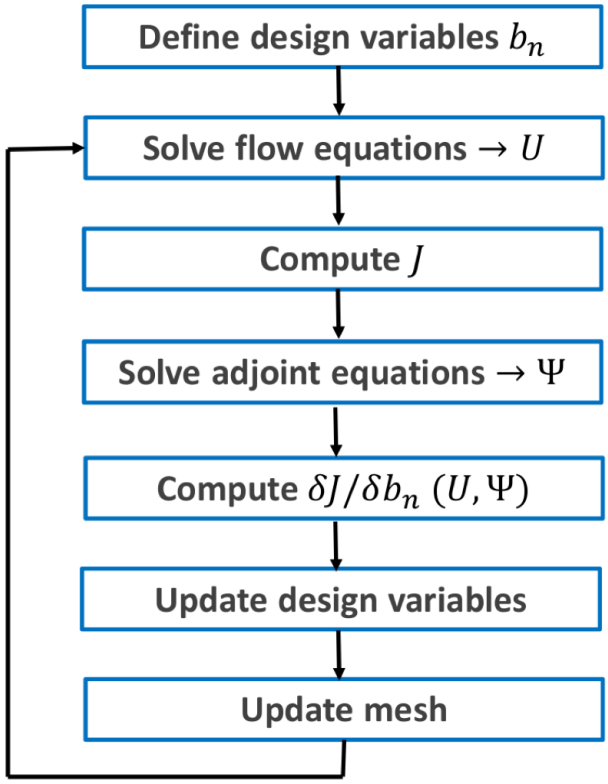


Figure 4.1: The adjoint-based shape optimisation loop executed by `adjointOptimisationFoam` when run in `steadyOptimisation`. [48]

In this thesis has been performed a shape optimization based on noise reduction. The study starts with a simple geometry, a **cube**, ending up with different settings of a **wheel**. The iterative procedure has been useful, at the beginning, to understand the `adjointOptimisationFoam` code which is implemented in OpenFOAM and the sensibility of the results on different parameters:

- *Primal* solver (section 4.2.1)
- *Adjoint* solver (section 4.2.2)
- Objective function definition (section 4.2.3)
- *ATC Model* (section 4.2.4)
- Sensitivities calculation method (section 4.2.5)
- Controlpoints (section 4.2.6)
- Mesh update method (section 4.2.7)

4.2.1 *primal solver*

Solution algorithm used to solve the primal equations which is based on `simpleFoam`. It is a steady-state solver for incompressible, turbulent flows, using the SIMPLE (Semi-Implicit Method for Pressure Linked Equations) algorithm.

The solver follows a segregated solution strategy which means that the equations for each variable characterizing the system (the velocity $\mathbf{u}$, the pressure p and the variables characterizing turbulence) are solved sequentially and the solution of the preceding equations is inserted in the subsequent equation. The non-linearity appearing in the momentum equation (the face flux ϕ_f which is a function of the velocity) is resolved by computing it from the velocity and pressure values of the preceding iteration. The dependence from the pressure is introduced to avoid a decoupling between the momentum and pressure equations and hence the appearance of high frequency oscillation in the solution (check board effect). The first equation to be solved is the momentum equation. It delivers a velocity field $\mathbf{u}^*$ which is in general not divergence free, so it does not satisfy the continuity equation. After that the momentum and the continuity equations are used to construct an equation for the pressure. The aim is to obtain a pressure field p^n , which, if inserted in the momentum equation, delivers a divergence free velocity field $\mathbf{u}$. After correcting the velocity field, the equations for turbulence are solved. The above iterative solution procedure is repeated until convergence.

4.2.2 *adjoint solver*

Solution algorithm, used to solve the adjoint equations, has been constructed for managing multiple objective functions and sensitivities calculation instead of a straight line solver.

It starts by defining the time step calculation followed by taking the results of the state equations variables and calling them ϕ. After that, the adjoint variables to be used are defined and the new term, the adjoint turbulence $\tilde{\tau}_a$ appears due to the differentiation of the turbulence [39]. So after adding the source term to the adjoint equation it ends up with a proposed "adjoint" Pressure-Velocity SIMPLE corrector[49]. All the steps can be found and explored in the *adjointSimple.C* file [50].

4.2.3 Objective function

As already seen in section 3.3 Proudman defines a formulation for the Acoustic Power generated by isotropic turbulence, starting from the Acoustic Intensity, which is based on Lighthill's results.

Linking Eq.33 and Eq.43 there is a direct connection between the Acoustic Power and the Lighthill's tensor T_{ij}.

$$T_{ij} = \rho u_i u_j - \tau_{ij} + (p - c_0^2 \rho)\delta_{ij} \tag{64}$$

where, as explained in [28], the last term can be neglected for isoentropic flows. Applying the Reynold's decomposition to ρ and u

$$\rho = \overline{\rho} + \rho' \tag{65}$$
$$u = \overline{u} + u' \tag{66}$$

a reasonable estimation of the quadrupole source can be

$$T_{ij} = \overline{\rho u_i u_j} + \overline{\rho u_i} u_j' + \overline{\rho} u_i' \overline{u_j} + \overline{\rho} u_i' u_j' \tag{67}$$

where the first term does not contribute to the production of sound, while the second and third terms will contribute only if there is a shear layer.
So

$$T_{ij} \approx \overline{\rho} u_i' u_j' \tag{68}$$

This formulation of T_{ij} is correlated to the Reynold's stress tensor which appears on the left hand side of the Boussinnesq equation (Eq.12) linking it with *turbulent viscosity* and *turbulent kinetic energy* as visible in Fig.4.3.

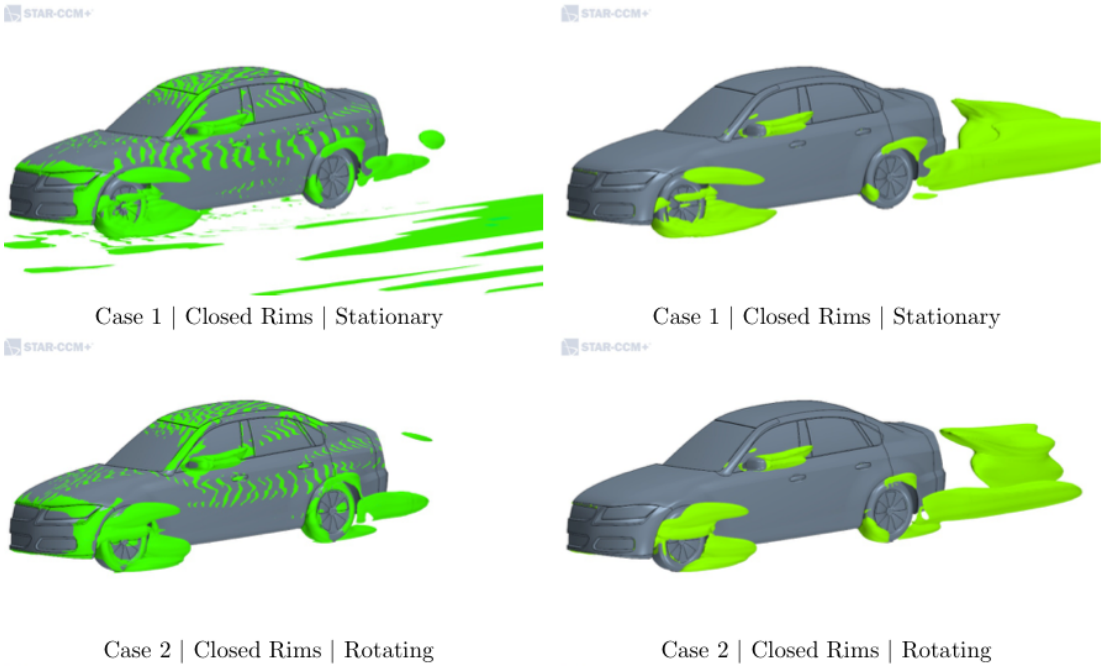


Figure 4.2: Left figures representing iso-surfaces of Proudman Acoustic Power at 70dB and the right ones showing iso-surfaces of Turbulent Kinetic Energy at 40 J/kg for closed rims. [34]

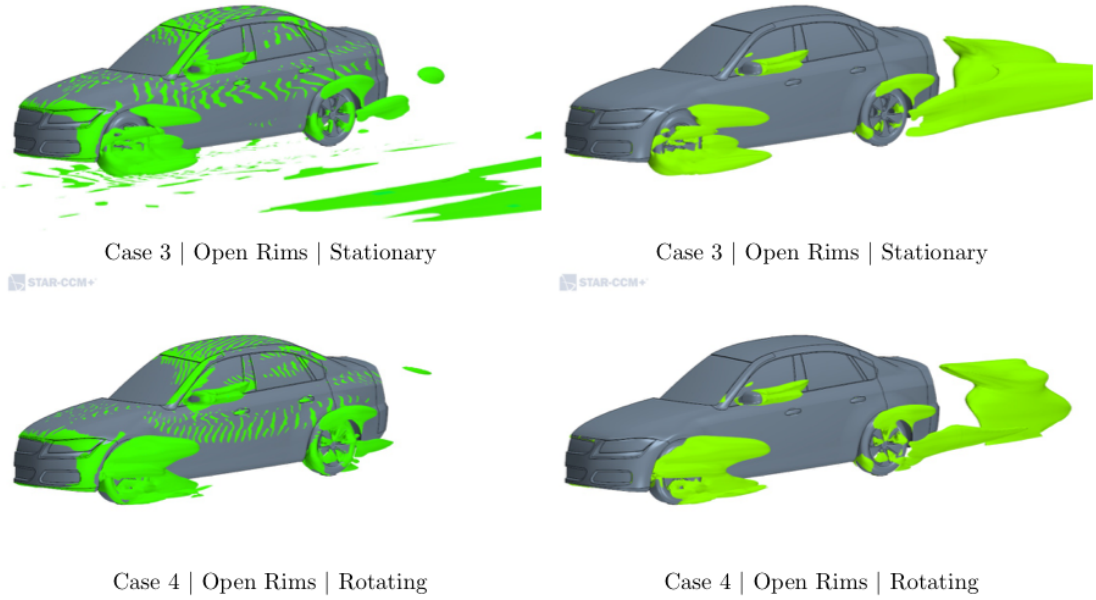


Figure 4.3: Left figures representing iso-surfaces of Proudman Acoustic Power at 70dB and the right ones showing iso-surfaces of Turbulent Kinetic Energy at 40 J/kg for open rims. [34]

Cost function correlations

To verify the correlations mentioned above I performed an unsteady 2D simulation of the flow around a cylinder with the following settings:

- Domain dimensions: 0.8×0.6 m ;
- Number of cells: 123200;
- Cylinder diameter: 0.04 m
- Re: 100
- Turbulence model: $k - \epsilon$;

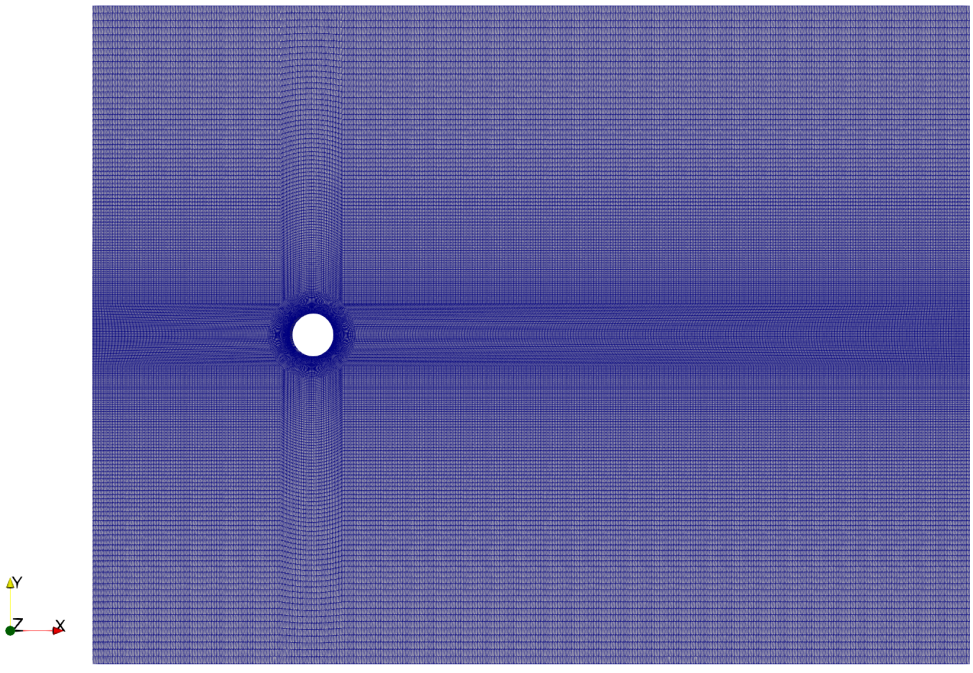
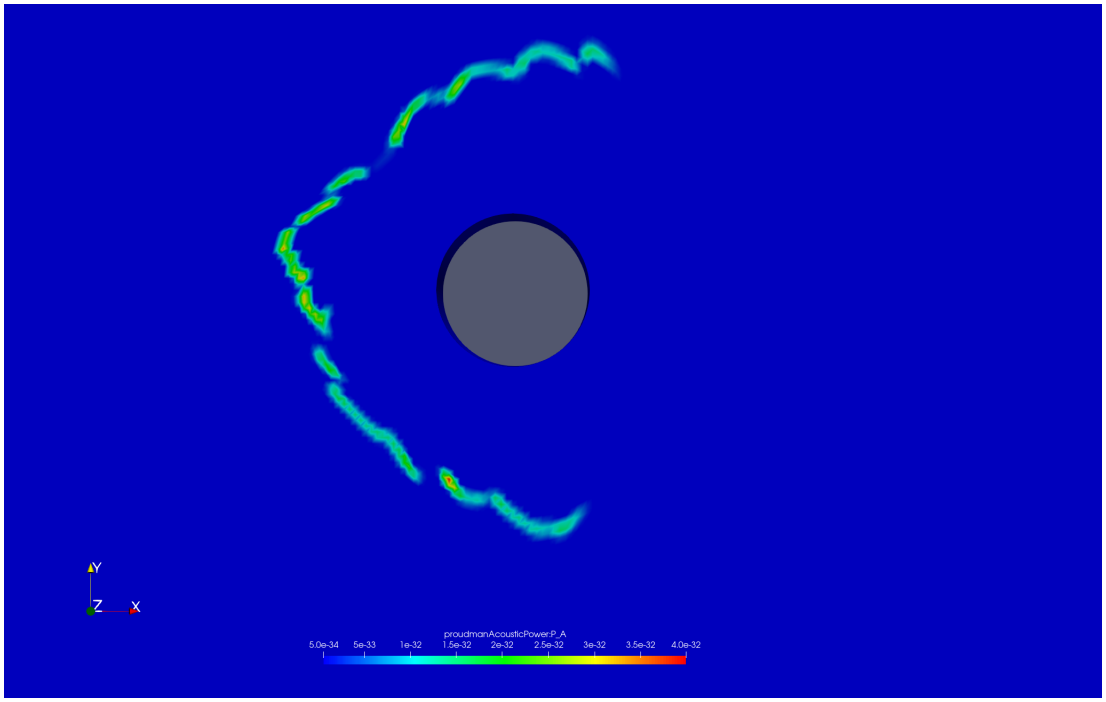
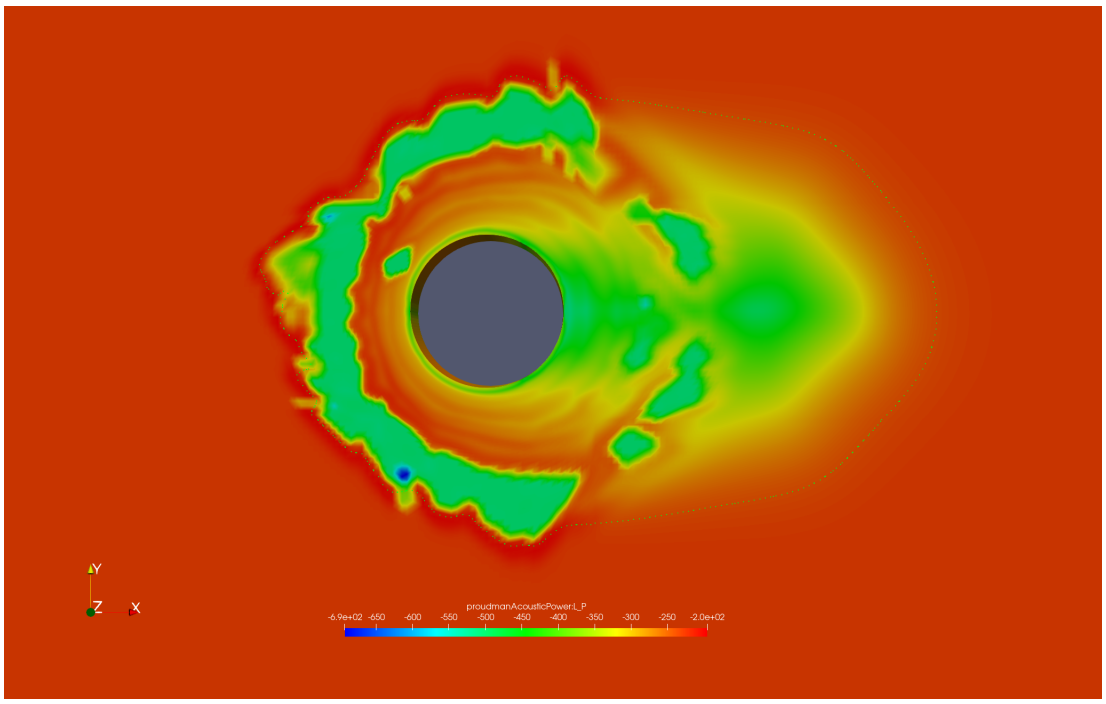


Figure 4.4: 2D cylinder computational domain

I computed the Proudman Acoustic Power through the `proudmanAcousticPower` explained in 3.3. The correlation is easily visible comparing the trend of PA or LP with the ones of turbulent kinetic energy K and the turbulent viscosity nut, as in the following figures.

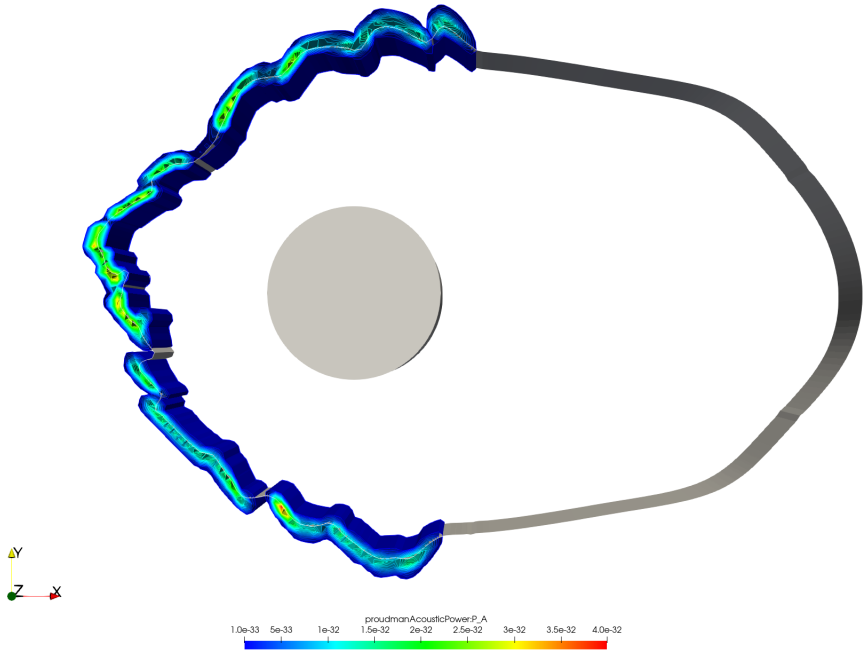


(a) *Proudman Acoustic Power (PA)*

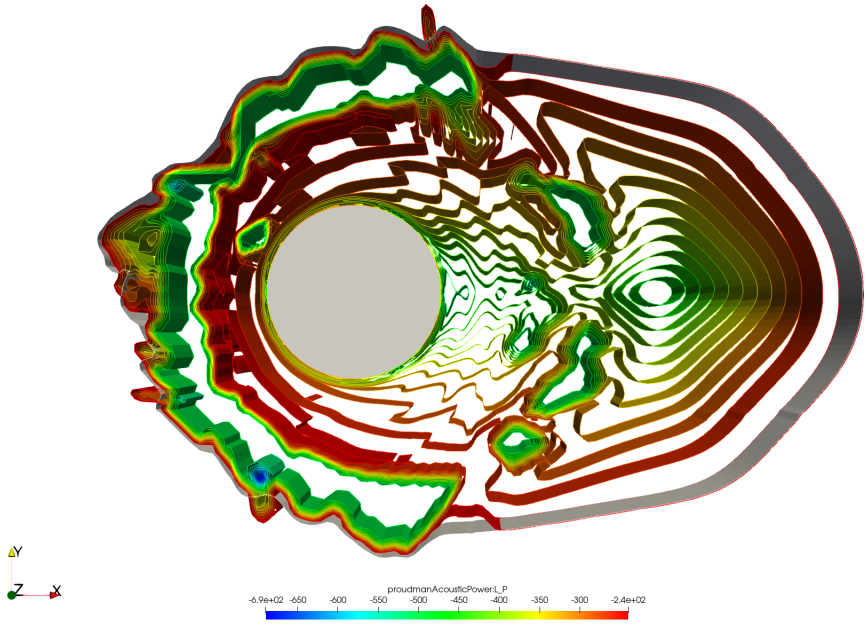


(b) *Proudman Acoustic Power in dB (LP)*

Figure 4.5: PA and LP trends in the domain

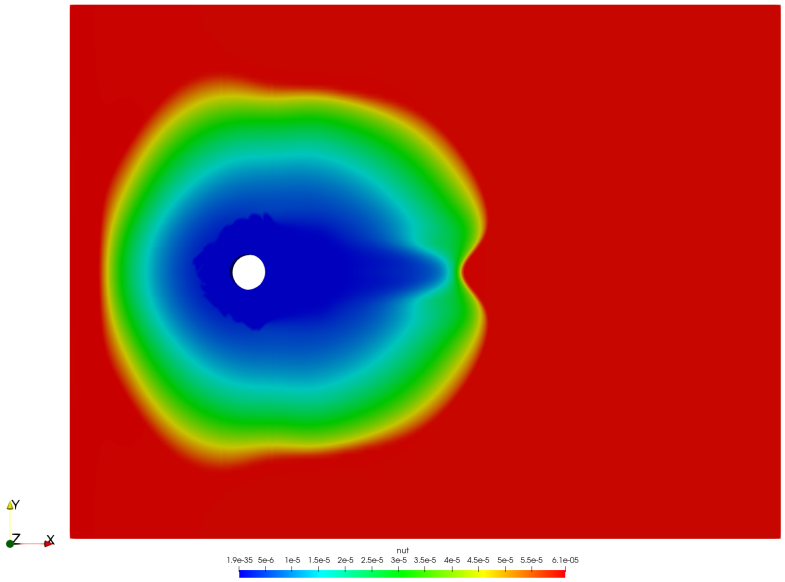


(a) *Comparison between the isoline of* $k = 1.2e^{-6}$ *and a range of isolines for PA*

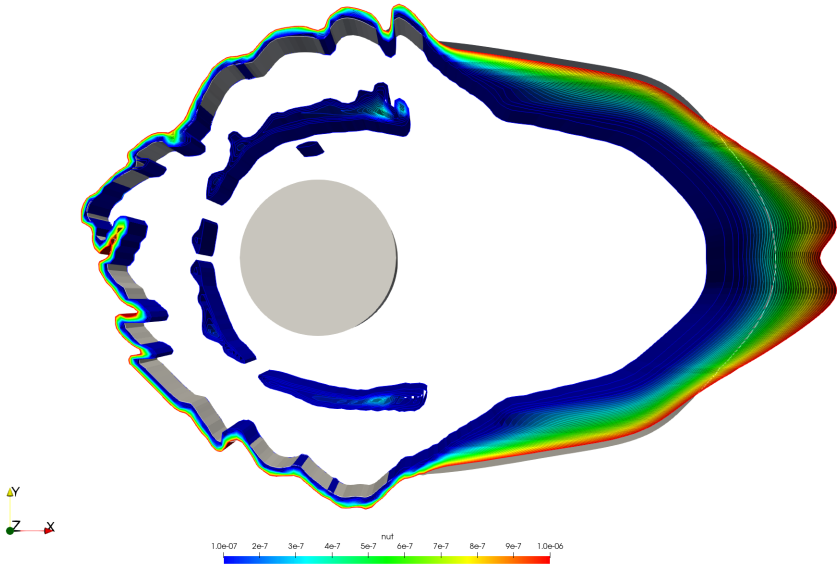


(b) *Comparison between the isoline of* $k = 1.2e^{-6}$ *and a range of isolines for LP*

Figure 4.6: The correlation between PA, LP and k



(a) *Comparison between the isoline of* $k = 1.2e^{-6}$ *and nut*



(b) *Comparison between the isoline of* $k = 1.2e^{-6}$ *and a range of isolines of nut*

Figure 4.7: The correlation between k and nut

In order to capture the noise generated by the wheel, the objective function is formulated as the volume integral of the square of the turbulent viscosity over an arbitrary domain according to [47]. Since the highest levels of turbulence appear at the lower frequencies of the energy spectrum, this is a good surrogate model for low frequency noise.

$$F = \int_{\Omega'} \nu_t^2 d\Omega \tag{69}$$

where $\Omega^{'}$ is a volume defined by the user.

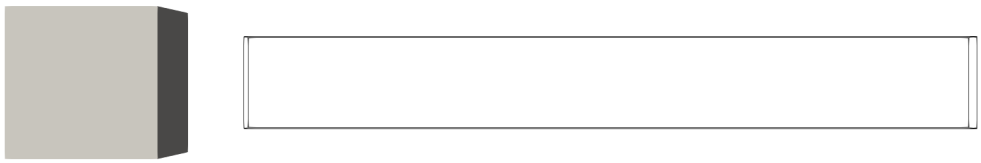
The differentiation gives:

$$\frac{\delta F}{\delta b_n} = \int_{\Omega'} 2\nu_t \frac{\partial \nu_t}{\partial \tilde{\nu}} \frac{\partial \tilde{\nu}}{\partial b_n} d\Omega + \int_{S'} \nu_t^2 n_k \frac{\delta x_k}{\delta b_n} dS \tag{70}$$

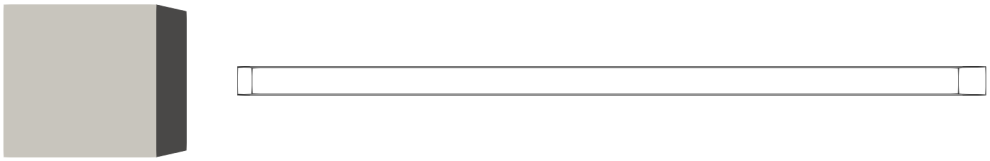
stating that an extra source term($\acute{F}_{\Omega}^{\tilde{\nu}} = 2\nu_t \frac{\partial \nu_t}{\partial \tilde{\nu}}$) must be added to the turbulence model equation (Eq.60), in the cells contained in $\Omega^{'}$. This term, in fact, triggers the development of the adjoint flow. If the *frozen turbulence* assumption would have been made, the adjoint method would not have been able to provide any kind of sensitivity information, since the objective depends exclusively on turbulence.

nutSqrZone

As just explained the cost function is defined over a user defined domain. Its dimensions are inserted in `topoSetDict` file which is in the `system` folder and through the `setsToZones` function the zone is created. In this case the domain Ω is represented by a box behind the geometry as visible in Fig.4.8 for the cube and in Fig.4.9 for the wheel.



(a) *xz plane*



(b) *xy plane*

Figure 4.8: nutSqrZone for the cube of dimension $5 \times 0.19 \times 0.62$ *m*

(a) *xz plane*

(b) *xy plane*

Figure 4.9: nutSqrZone for the wheel of dimension $5 \times 2 \times 2$ m

4.2.4 ATC Model

The ATCModel dictionary provides the available options for the so-called Adjoint Transpose Convection (ATC) term, existing in the adjoint momentum equations. It is numerically stiff and can often cause convergence difficulties for the adjoint equations because the gradient has to be expressed in explicit form but the adjoint velocity is an unknown.
The ATCModel dictionary provides some options to smooth it in order to facilitate convergence in industrial cases.

In order of decreasing robustness they are:

- `cancel`: This option excludes the ATC term from the adjoint momentum equations during the solution of the adjoint PDEs (at the same time, of course, loosing some accuracy depending on the case);

- `standard`: It is formulated by differentiating the non-conservative form of the convection term in the primal momentum equations. ATC term is computed as $u_{a_j} \frac{\partial u_j}{\partial x_i}$;

- `UaGradU`: It is formulated by differentiating the conservative form of the convection term in the primal momentum equations. The ATC term is computed as $u_j \frac{\partial u_{a_j}}{\partial x_i}$.

In the following simulations the `standard` option has been selected because it uses the correct explicit discretization of the gradient which is on the velocity calculated with RANS so it neither looses accuracy, nor increases the convergence difficulties.

4.2.5 Sensitivities calculation method

Sensitivities will be computed after all adjoint PDEs are solved. If the interest is on the sensitivity maps `surface` or `surfacePoints` has been selected, while for the optimization followed by the mesh morphing has been used `volumetricBSplinesFI`:

- `surface`: It is used to compute the so called sensitivity maps, i.e. the derivative of the objective function with respect to the normal displacement of the boundary wall faces;
- `surfacePoints`: Same as surface, but sensitivities are computed with respect to the normal displacement of boundary points, not faces. This option should be preferred to `surface` since some of the terms included in the computations (e.g. variation in the normal vector) are better posed when differentiating with respect to points;
- `volumetricBSplinesFI`: This option computes sensitivity derivatives with respect to the control points of a volumetric B-splines morpher. Sensitivities are computed using the chain rule

$$\frac{\delta J}{\delta b_n} = \frac{\delta J}{\delta x_i} \frac{\delta x_i}{\delta b_n} \tag{71}$$

  when $\frac{\delta J}{\delta x_i}$ is the sensitivity map and $\frac{\delta x_i}{\delta b_n}$ is computed analytically on the surface, by differentiating the volumetric B-splines morpher. For further informations [51] and [52].

They also compute and write the displacement of all mesh points for each geometry generated by the optimization loop, from the initial geometry. The vectorial difference of all mesh points $(x_{new} - x_{old})$ is written in a *pointVectorField* named `displacement` whereas the projection of this difference to the normal vector of the boundary mesh points in the initial geometry $((x_{new} - x_{old})n_{old})$ is written in a *pointScalarField* named `normalDisplacement`. Keeping in mind the convention for the surface normal unit vector, facing from the fluid to the solid boundaries, positive normal displacements indicate a movement aligned to the geometry normal ("inwards" or "outwards", for external or

internal aerodynamics, respectively). Negative normal displacements indicate a movement opposite to the geometry normal ("outwards" or "inwards" for external or internal aerodynamics, respectively).

4.2.6 Controlpoints

A box of n control points in each direction is placed where the shape optimization has to be performed and the mesh is then parametrized using B-splines. When the control box separates the mesh in parameterized and non-parameterized regions, the boundary control points of the control grid have to be fixed in order to ensure C_0 continuity at the interface of the two regions.

Definition

Let b_m^{ijk} be the Cartesian coordinates of the i, j and k-th control points of the 3D structured control grid. I, J and K are the number of control points per control grid direction. The Cartesian coordinates $\mathbf{x} = [x_1, x_2, x_3]^T = [x, y, z]^T$ of a CFD mesh point residing within the boundaries defined by the control grid are given by

$$x_m(u, v, w) = U_{i,pu}(u) V_{j,pv}(v) W_{k,pw}(w) b_m^{ijk} \tag{72}$$

Here $\mathbf{u} = [u_1, u_2, u_3]^T = [u, v, w]^T$ are the mesh point parametric coordinates, U, V and W are the B-Splines basis functions and pu, pv and pw their respective degrees, which may be different per control grid direction. Computing the Cartesian coordinates of any parameterized mesh point is straightforward, at a negligible computational cost, as long as its parametric coordinates $\mathbf{u}$ are known. Mesh parametric coordinates can be computed with accuracy, since a mapping from $\mathcal{R}^3(x, y, z) \rightarrow \mathcal{R}^3(u, v, w)$ is required. This means that volumetric B-Splines can reproduce any geometry to machine accuracy. Given the control points position, the knot vectors and the basis functions degrees, the parametric coordinates (u, v, w) of a point with Cartesian coordinates $\mathbf{r} = [x_r, y_r, z_r]^T$ can be computed by solving the system of equations

$$\mathbf{R}(u, v, w) = \begin{bmatrix} x(u, v, w) - x_r = 0 \\ y(u, v, w) - y_r = 0 \\ w(u, v, w) - w_r = 0 \end{bmatrix} \tag{73}$$

where $x_m = (u, v, w)$ are computed through Eq.72, based on the known $\mathbf{b}$ values.

The $3x3$ system of Eq.73 can be solved independently for each parameterized mesh point using the Newton–Raphson method, after computing and inverting the Jacobian $\frac{\partial x_m}{\partial u_j}, m, j \in [1, 3]$. The Jacobian matrix has a closed form expression resulting by differentiating Eq.72 with respect to the components

of **u**. Since the evaluation of the parametric coordinates of each point is independent from any other mesh point, these computations may run efficiently in parallel. The aforementioned process has to be done only once and can be seen as the "training phase" of the method. Then, after moving the control points **b**, the Cartesian coordinates of each (internal or boundary) mesh point residing within the control grid can be computed through Eq.72 at a very low cost, making volumetric B-Splines a powerful surface parameterization and mesh displacement tool.

In addition, since x_m depends only on (u,v,w) (which remain unchanged whatever the change in **b** might be) and **b**, the deformed meshes are step-independent. This facilitates the mapping of a geometry parameterized using volumetric B-Splines from one CFD mesh to another. Provided that the initial control grid for the two CFD meshes is the same, the control points displacement computed over the first mesh can be applied to the second too. Then the CFD mesh point positions of the second mesh can easily be computed through Eq.72.

The control point box is created starting from a *Python* script developed for this thesis. Defining the box extremes in cartesian coordinates (U,V,W) and the number of layers inside the box, it generates a file called `boxcpsBsplines0`. This file is read by the `dynamicMeshDict` which contains the dictionary for the control points management and it is located in the `constant` folder of OpenFoam.
Herein it is possible to define the number of points in each direction *nCPs* (the same as the layers of the `boxcpsBsplines0`), the basis function degree for each direction (the max is $nCPs-1$) and which movements are allowed.

The control points confinement has different options:

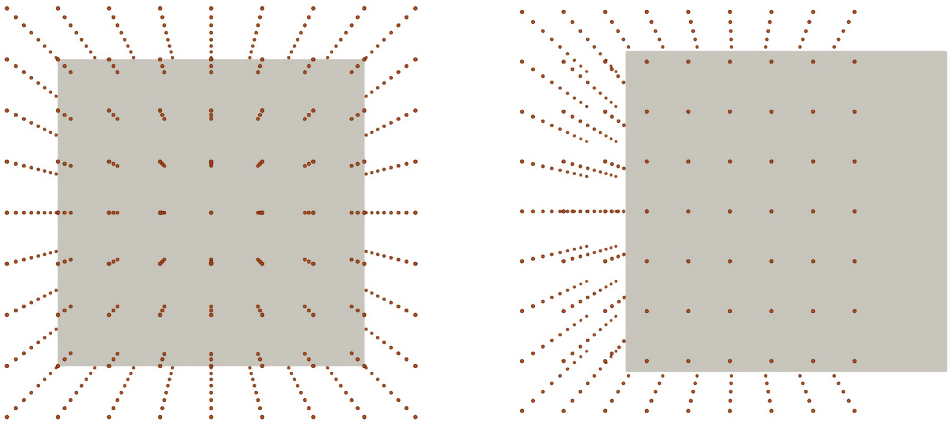
- to confine just the movement in one of the three cartesian directions;
- to confine all the boundary points to ensure that mesh elements will not overlap in the boundaries of the control grid (even if gradient and curvature continuity might not be guaranteed);
- to define more layers of control points to be kept fixed starting from the confine.

Cube

```
box
{
    type    cartesian;
    nCPsU   9;
    nCPsV   9;
    nCPsW   9;
    degreeU 3;
    degreeV 3;
    degreeW 3;

    controlPointsDefinition fromFile;

    confineUMovement false;
    confineVMovement false;
    confineWMovement false;
    confineBoundaryControlPoints false;

    confineUMinCPs ( (true true true) (true true true) );
    confineUMaxCPs ( (true true true) );
    confineVMinCPs ( (true true true) (true true true) );
    confineVMaxCPs ( (true true true) (true true true) );
    confineWMinCPs ( (true true true) );
    confineWMaxCPs ( (true true true) (true true true) );
}
```

Figure 4.10: `dynamicMeshDict` settings for cube simulation

For the cube optimization, as visible in Fig.4.10 the basis function degree has been set to 3 and the box is of 9 points in each direction. The points are confined so that the first two rows corresponding to U_{min}, V_{min}, V_{max}, W_{max} and the first row of U_{max} and W_{min} are locked.



(a) *yz plane* (b) *xz plane*

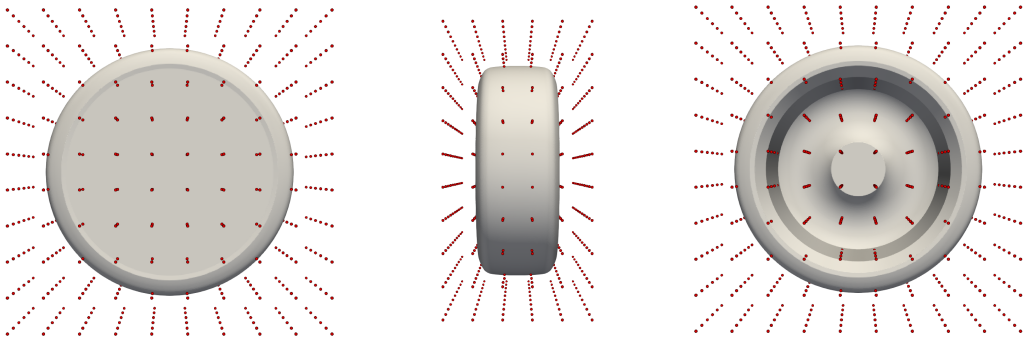
Figure 4.11: Control points for cube

Wheel

```
box
{
    type    cartesian;
    nCPsU   10;
    nCPsV   6;
    nCPsW   10;
    degreeU 3;
    degreeV 3;
    degreeW 3;

    controlPointsDefinition fromFile;

    confineUMovement true;
    confineVMovement false;
    confineWMovement true;
    confineBoundaryControlPoints false;

    confineUMinCPs ( (true true true) (true true true) );
    confineUMaxCPs ( (true true true) );
    confineVMinCPs ( (true true true) (true true true) );
    confineVMaxCPs ( (true true true) (true true true) );
    confineWMinCPs ( (true true true) );
    confineWMaxCPs ( (true true true) (true true true) );
}
```

Figure 4.12: `dynamicMeshDict` settings for wheel simulation

For the wheel optimization, as visible in Fig.4.12 the basis function degree has been set to 3 and the box is of $10 \times 6 \times 10$ points. The points are confined so that the first two rows corresponding to U_{min}, V_{min}, V_{max}, W_{max} and the first row of U_{max} and W_{min} are locked. Moreover the movement of all the points in U and W directions are confined in order to obtain a more "realistic" configuration.



(a) *front* (b) *xz plane* (c) *back*

Figure 4.13: Control points for wheel

4.2.7 Mesh update method

The mesh update method follows the scheme

$$b_i^{new} = b_i^{old} + \eta p_i \tag{74}$$

where **b** are the design variables, **p** the update direction and η a user-defined step. It can be performed with different method:

- `steepestDescent`: the simplest and robust method but also the slowest in convergence. The update vector is computed as $p_i = -\frac{\delta J}{\delta b_i}$;
- `conjugateGradient`: much faster than the previous one and can tolerate discrepancies in the sensitivities, so it should be chosen when a balance between accuracy and stability is required;
- `BFGS`: is a quasi-Newton method and it is probably the most widely used method to update the design variables in general optimization problems. The update is done with

$$\frac{\delta^2 \tilde{J}}{\delta b_i \delta b_j} p_j = -\eta_H \frac{\delta J}{\delta b_i} \tag{75}$$

  where $\frac{\delta^2 \tilde{J}}{\delta b_i \delta b_j}$ is an approximation of the objective function Hessian and η_H is a user defined constant. Its convergence is significantly faster than the others two but it requires highly accurate sensitivity derivatives.

For this thesis the `maxAllowedDisplacement`, (`/system/optimisationDict`), has been set equal to $2e^{-3}$. As the comparison between `conjugateGradient` and `BFGS` has shown no significative differences, the last one has been selected in order to achieve a faster convergence.

4.2.8 Boundary conditions

Boundary conditions (*BC*) are formulated by appropriately treating the surface integral of Eq.62 that contains variations in the flow variables.

Inlet boundaries (S_I)

Imposing *Dirichlet conditions* on v_i and $\tilde{\nu}$ along S_I, makes the first and third integral of Eq.62 vanish. To make it independent of $\frac{\partial}{\partial bn}$, $\frac{\partial \tau_{ij}}{\partial bn}$ and $\frac{\partial}{\partial b_n}(\frac{\partial \tilde{\nu}}{\partial x_j})n_j$ along S_I the following *BC* have to be imposed:

- $u_j n_j = u_{\langle n \rangle} = -\acute{F}^p_{S_I}$
- $u^I_{\langle t \rangle} = 0$

- $u^{II}_{\langle t \rangle} = 0$
- $\tilde{v}_a = 0$

where $u_{\langle n \rangle}$, $u^{I}_{\langle t \rangle}$ and $u^{II}_{\langle t \rangle}$ are respectively u_a, v_a, w_a, the adjoint velocity components.

Outlet boundaries (S_O)

Zero Neumann conditions are imposed on v_i and $\tilde{\nu}$, with a *zero Dirichlet condition* on p along S_O, so that the second and sixth integral of Eq.62 vanish.
The fifth integral can be considered to be negligible by assuming a developed velocity profile at S_O. After zeroing the multiplier of $\frac{\partial \tilde{\nu}}{\partial b_n}$ in the third integral, a *Robin-type* BC is imposed for $\tilde{v}_a$

$$\mathcal{BC}^{\tilde{v}_a} = \tilde{v}_a v_j n_j + \left(\nu + \frac{\tilde{\nu}}{\sigma}\right)\frac{\partial \tilde{v}_a}{\partial x_j} n_j + \acute{F}^{v}_{S_O} = 0 \tag{76}$$

The first integral must be eliminated by satisfying the following equation in order to make Eq.62 independent from $\frac{\partial v_i}{\partial b_n}$

$$\mathcal{BC}^{u}_{i} = u_i v_j n_j + \tau^{a}_{ij} n_j - q n_i + \acute{F}^{\nu}_{S_{O,i}} + \tilde{v}_a \tilde{\nu} \frac{\mathcal{C}_Y}{Y} e_{mjk} \frac{\partial v_k}{\partial x_j} e_{mli} n_l = 0 \tag{77}$$

In order to formulate the BC for the outlet adjoint pressure and the outlet adjoint tangential velocity, Eq.77 is decomposed into its normal and tangential components:

$$q = u_{\langle n \rangle} v_{\langle n \rangle} + 2(\nu + \nu_t)\frac{\partial u_{\langle n \rangle}}{\partial n} + \acute{F}^{\nu}_{S_{O,i}} n_i + \tilde{v}_a \tilde{\nu} \frac{\mathcal{C}_Y}{Y} e_{mjk} \frac{\partial v_k}{\partial x_j} e_{mli} n_l n_i \tag{78}$$

$$\begin{aligned} 0 &= \nu_n u^{l}_{\langle t \rangle} + (\nu + \nu_t)\left(\frac{\partial u^{l}_{\langle t \rangle}}{\partial n} + \frac{\partial u_{\langle n \rangle}}{\partial t^{l}}\right) + \acute{F}^{v}_{S_{O,i}} t^{l}_{i} \\ &\quad - \tilde{v}_a \tilde{\nu} \frac{\mathcal{C}_Y}{Y} e_{mjk} \frac{\partial v_k}{\partial x_j} e_{mzi} n_z t^{l}_{i}, \qquad l = I, II \end{aligned} \tag{79}$$

At the end to close the adjoint system a *zero Neumann* BC is imposed on $u_{\langle n \rangle}$.

Fixed(S_W) and controlled(S_{W_p}) wall boundaries

- Along S_W, due to the *Dirichlet condition* imposed on $\tilde{\nu}$, the third integral in Eq.62, vanishes, while a zero *Dirichlet condition* is imposed on $\tilde{v}_a$ to make it independent of $\frac{\partial}{\partial b_n}(\frac{\partial \tilde{\nu}}{\partial x_j})n_j$.

$$\tilde{v}_a = 0 \tag{80}$$

- To eliminate the dependency of $\frac{\delta F_{aug}}{\delta b_n}$ on $\frac{\partial p}{\partial b_n}$ the normal adjoint velocity has to be equal to

$$u_{\langle n \rangle} = -\acute{F}^{p}_{S_W} \tag{81}$$

- Further developing the first and fifth integral, it gives a zero *Dirichlet condition* for $u^{II}_{\langle t \rangle}$ and an expression for u_τ which is the *adjoint friction velocity*

$$u^{II}_{\langle t \rangle} = 0 \tag{82}$$

$$u^{2}_{\tau} = -\tau^{a}_{ij} n_j t^{I}_i = \acute{F}^{v}_{S_{W,i}} t^{I}_i \tag{83}$$

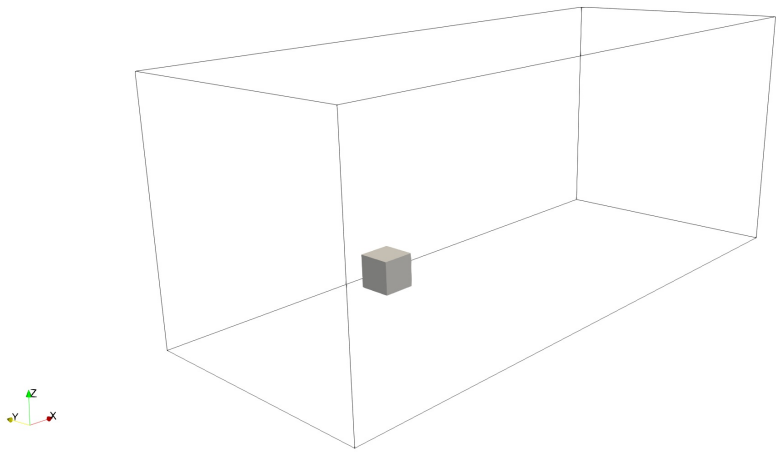
where the *adjoint friction velocity* is an indispensable part of the adjoint system of equations to the Spalart–Allmaras model with wall functions. In fact, due to the long distance between f and P, differentiating normal to the wall is prone to important errors. For further informations about the effects of neglecting the *adjoint law of the wall* see [47].

5 NUMERICAL MODEL

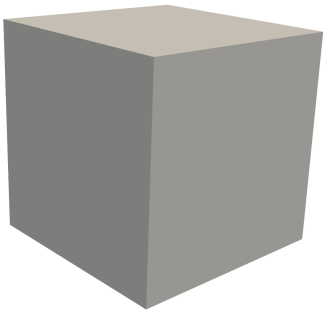
This chapter shows how the problem has been modeled and set for the simulations. For both the cube and the wheel cases there will be the explanation of the meshing strategy, of the grid convergence analysis and of the procedure used to set the *primal* and *adjoint* solvers. At the end the boundary conditions, which have been defined in section 4.2.8, will be presented in detail.

5.1 CUBE

5.1.1 Domain and geometry



(a) *Computational domain*



(b) *Geometry*

Figure 5.1: Computational domain and geometry for the cube

The computational domain has the following dimensions: $20 \times 8 \times 8$ m. The cube, instead, is characterized by a 1 m side.

5.1.2 Mesh

In order to analyze the grid convergence three simulations have been performed with three different meshes. The solver selected is `simpleFoam` with 18000 iterations, which is the primal solver of the `adjointOptimisationFoam`.

	$Ncells$	F_x	F_y	F_z
$Mesh_A$	1103149	219.257	-0.091	-0.7506
$Mesh_B$	2120186	220.510	-0.028	-1.001
$Mesh_C$	3888029	216.926	0.047	-1.042

Table 5.1: Comparison of the mesh

Observing the values of the force decomposed in the three cartesian directions in Tab.5.1 and plotted in Fig.5.2, it is easy to note how the differences are so small to be negligible. The counterpart of increasing the mesh resolution is the increase of computational cost. Thus considering the various cycles to be performed, the computational requirements would be too high.

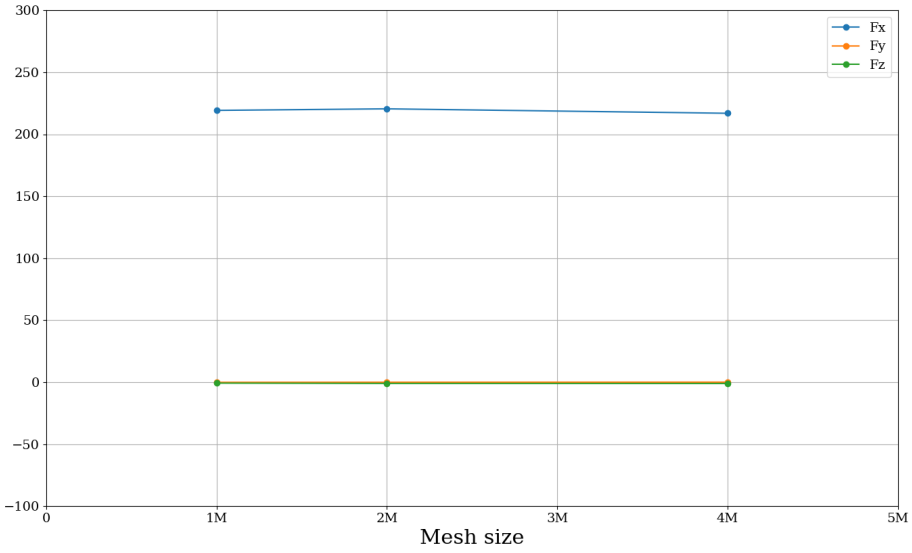
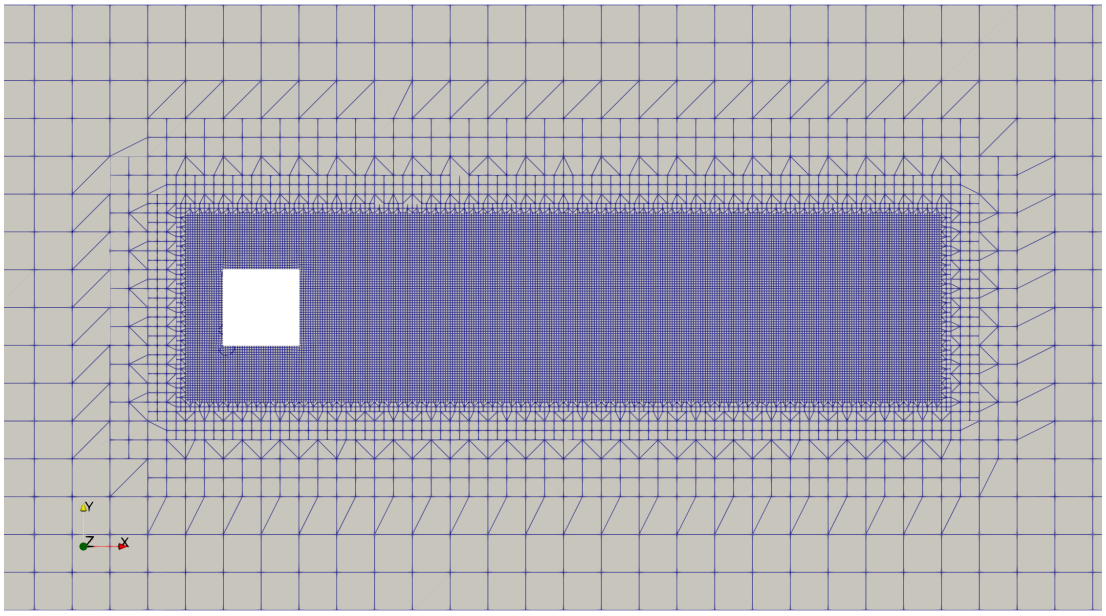


Figure 5.2: Comparison of the forces between 1M, 2M and 4M meshes

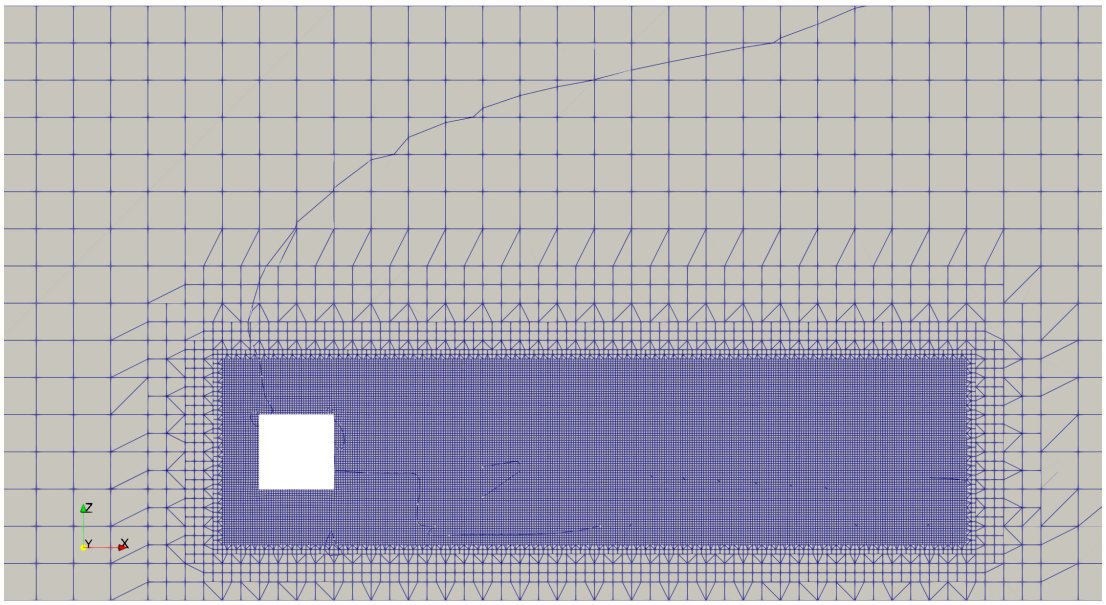
The optimization loop has been developed on $Mesh_B$ because, as already seen in the grid convergence analysis, the differences among the calculated

forces are negligible. Moreover, as suggested in [47], it is better to use a coarse mesh in order to reduce the computational cost and time. In this way only geometries which provide remarkable results are tested with a finer mesh.

The mesh has been developed using the OpenFoam tools `BlockMesh` and `SnappyHexMesh`. The refinement has been performed through some `refinementBox` so that the resolution is increased without getting worse the scaling of the cells.



(a) *Mesh slice xy plane*



(b) *Mesh slice xz plane*

Figure 5.3: Slice of the meshed domain

The quality parameters of the mesh are:

Aspect ratio	max	1.23925
Non-orthogonality	max	25.267
	average	3.99447
Skewness	max	0.333363

Table 5.2: Cube mesh quality parameters before optimization

5.1.3 Solvers setting

A crucial parameter to set was the max number of iterations for the primal solver. They had to be enough to reach a stable solution but not so much to spread up the computational cost. This last aspect is of fundamental importance because the max number of iterations, in the worst case, is repeated for each cycle. In order to define how many iterations set for the primal solver, I analized the residuals of the simulation (Fig.5.4) performed for the grid convergence on $Mesh_B$. Taking care of the balance between computational time and stability of the initial residuals, I defined the max number of iterations at 7000 with the averaging starting from iteration number 1000 for both the *primal* and *adjoint* solvers.

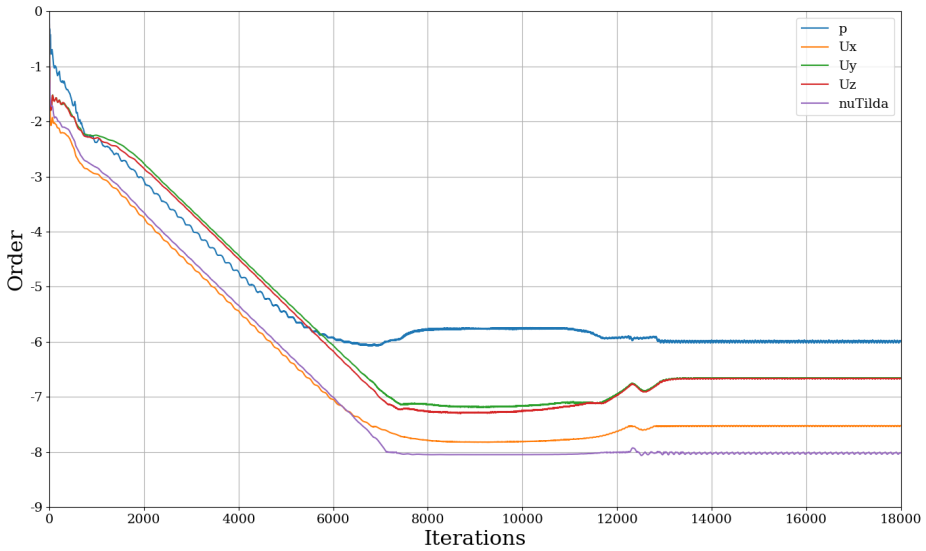
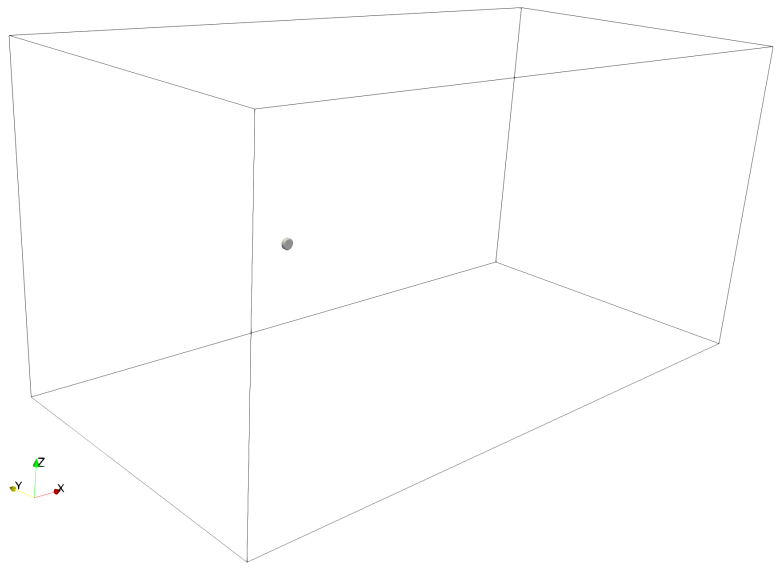


Figure 5.4: *simpleFoam* residuals on a 2M mesh

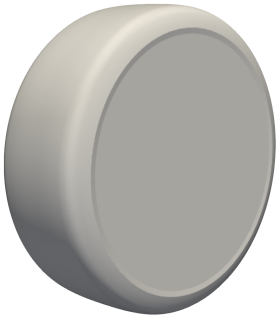
The stopping criteria for the *primal* and *adjoint* solvers have been imposed on the `residualControl`. For the former they are $1e^{-5}$ both for P and $\mathbf{u}$. For the latter the same value but for P_a, $\mathbf{u_a}$ and `nuaTilda`.

5.2 WHEEL

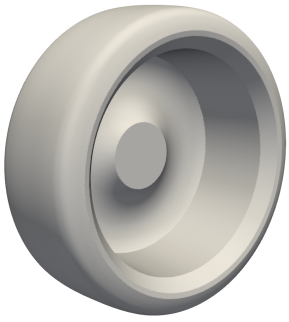
5.2.1 Domain and geometry



(a) *Computational domain*



(b) *Geometry-Front*



(c) *Geometry-Back*

Figure 5.5: Computational domain and geometry for the wheel

The computational domain has the following dimensions: $40 \times 16 \times 16$ m. The wheel, instead, is characterized by an external diameter of 0.65 m, a width of 0.24 m and a height of 0.06 m.

5.2.2 Mesh

In order to analyze the grid convergence three simulations have been performed with three different meshes as for the cube. The solver selected is `simpleFoam` with 18000 iterations, which is the primal solver of the `adjointOptimisationFoam`).

	$Ncells$	F_x	F_y	F_z
$Mesh_A$	4228526	7.358	-4.218	0.274
$Mesh_B$	8449427	7.320	-3.644	0.344
$Mesh_C$	16449918	7.182	-2.712	0.247

Table 5.3: Comparison of the mesh

Observing the values of the force decomposed in the three cartesian directions in Tab.5.3 and plotted in Fig.5.6, it is easy to note how the differences are so small to be negligible. The advantage of increasing the mesh resolution is cancelled by the increase of computational cost also in this case.

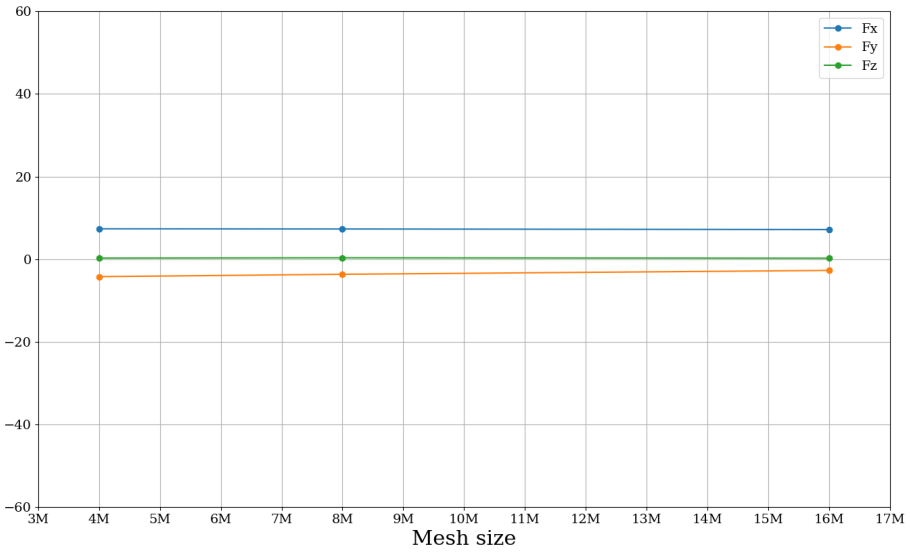


Figure 5.6: Comparison of the forces between 4M, 8M and 16M meshes

The optimization loop has been developed on $Mesh_B$ because, as already seen in the grid convergence analysis, the differences among the calculated forces are negligible.

The mesh has been developed using the OpenFoam tools `BlockMesh` and `SnappyHexMesh`. The refinement has been performed, as for the cube, through some `refinementBox` so that the resolution is increased without getting worse the scaling of the cells.

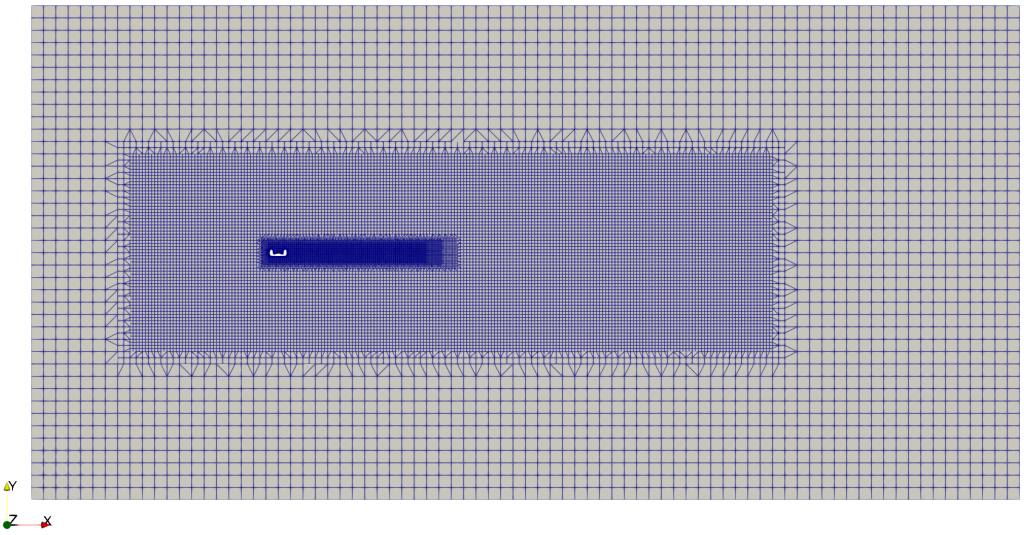


Figure 5.7: Wheel mesh slice *xy plane*

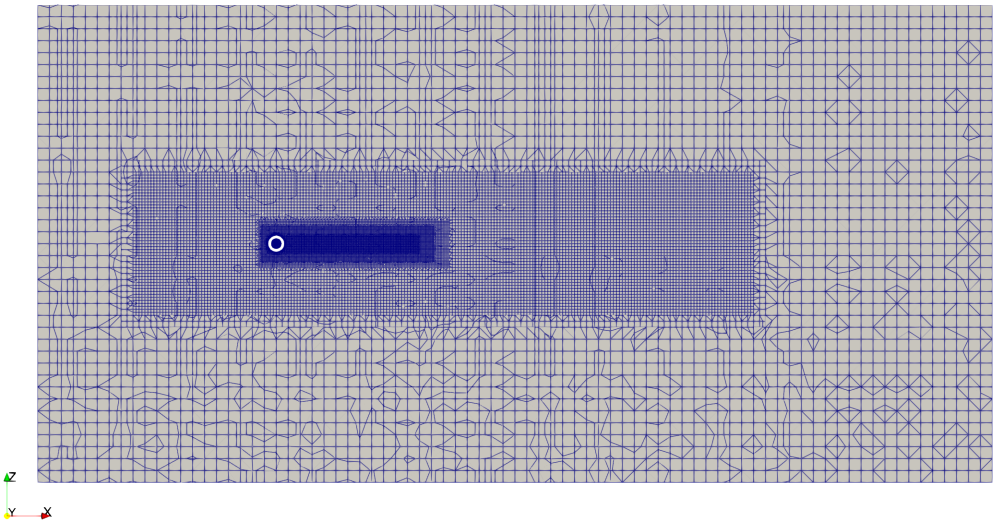


Figure 5.8: Wheel mesh slice *xz plane*

In Fig.5.9 and Fig.5.10 there is a focus on the cells near the wheel. This area is very critical in order to obtain a good mesh because of the edges and grooves. A few amount of cells here would not reproduce the exact geometry but too much cells would rise up the computational cost.

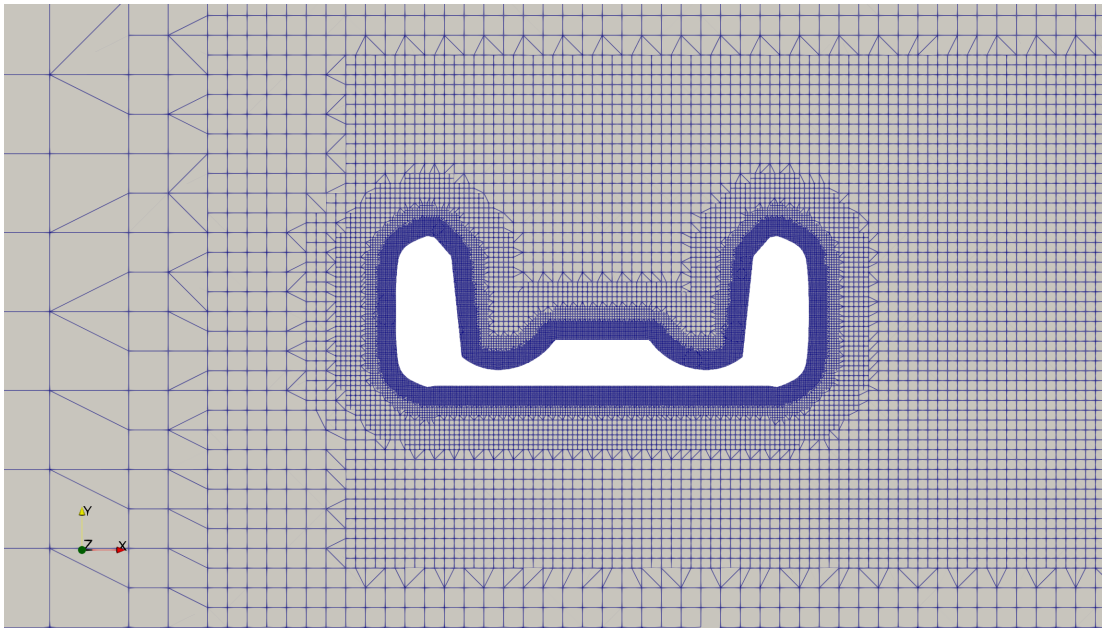


Figure 5.9: Detail in the mesh slice *xy plane*

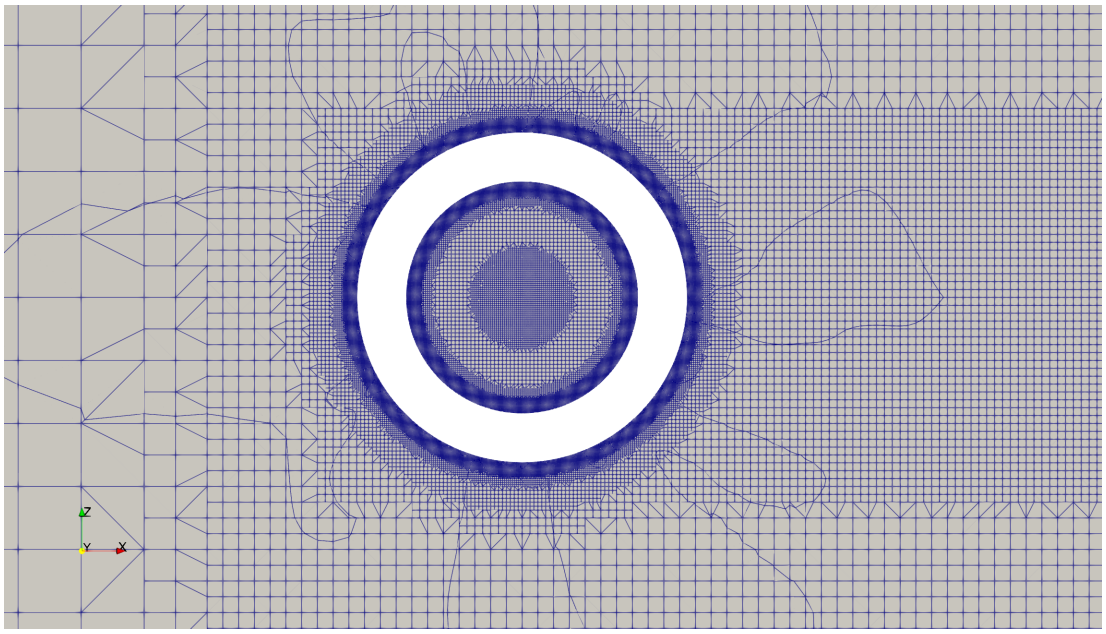
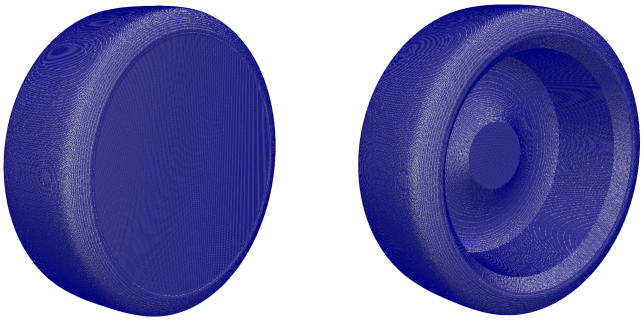


Figure 5.10: Detail in the mesh slice *xz plane*

The wheel geometry has been refined a lot in order to help the convergence and increase the sensitivities quality. Here I imposed a `level 7` which means that the smallest cell on the wheel is 3.9×10^{-3} m.



(a) *Front* (b) *Back*

Figure 5.11: Wheel geometry

The quality parameters of the mesh are:

Aspect ratio	max	4.47395
Non-orthogonality	max	39.7099
	average	5.19576
Skewness	max	1.0613

Table 5.4: Wheel mesh quality parameters before optimization

5.2.3 Solvers setting

In order to define how many iterations set for the primal solver, a simulation of the wheel baseline running a `simpleFoam` up to 18000 iterations has been performed as for the cube.

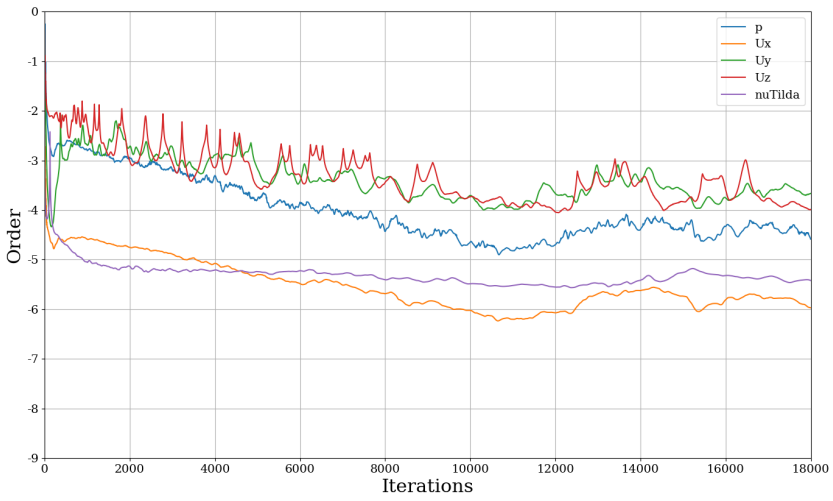


Figure 5.12: *simpleFoam* final residuals

Analyzing the initial residuals in Fig.5.12 can be observed a big difference with respect to Fig.5.4. Here residuals are less smooth and it is possible to see how, if in the case of the cube after about 7000 iterations they go to convergence, here they continue to oscillate around a value.
In order to mitigate this effect an average of the flow parameters has been performed during both the *primal* and *adjoint* simulations.

The stopping criteria for the *primal* and *adjoint* solvers have been imposed, as for the cube, on the `residualControl`. For the former they are $1e^{-5}$ both for P and $\mathbf{u}$. For the latter the same value but for P_a, $\mathbf{u_a}$ and `nuaTilda`. The *primal* solver has been set to perform 9000 iterations at the first cycle and 5000 for the following. The *adjoint* solver has been set to perform 9000 iterations in each cycle.

5.3 BOUNDARY CONDITIONS

5.3.1 Velocity u

Inlet	type	fixedValue
	value	(41.67, 0, 0)
Outlet	type	inletOutlet
	value	(41.67, 0, 0)
UpperWall	type	slip
LowerWall	type	fixedValue
	value	(41.67, 0, 0)
FrontAndBack	type	slip
Wheeel	type	noSlip

Table 5.5: *BC* for u

5.3.2 Pressure p

Inlet	type	zeroGradient
Outlet	type	fixedValue
	value	0
UpperWall	type	slip
LowerWall	type	zeroGradient
FrontAndBack	type	slip
Wheeel	type	zeroGradient

Table 5.6: *BC* for p

5.3.3 Turbulent viscosity nut

Inlet	type	calculated
	value	uniform 0
Outlet	type	calculated
	value	uniform 0
UpperWall	type	calculated
	value	uniform 0
LowerWall	type	nutSpaldingWallFunction
	value	uniform 0
FrontAndBack	type	calculated
	value	uniform 0
Wheeel	type	nutSpaldingWallFunction
	value	uniform 0

Table 5.7: BC for nut

5.3.4 Spalart-Allmaras variable $nuTilda$

Inlet	type	fixedValue
	value	1.5e-04
Outlet	type	inletOutlet
	inletValue	1.5e-04
	value	1.5e-04
UpperWall	type	slip
LowerWall	type	fixedValue
	value	uniform 0
FrontAndBack	type	slip
Wheeel	type	fixedValue
	value	uniform 0

Table 5.8: BC for $nuTilda$

5.3.5 Adjoint velocity u_a

Inlet	type	adjointInletVelocity
	value	uniform (0 0 0)
Outlet	type	adjointOutletVelocity
	inletValue	uniform (0 0 0)
	value	uniform (0 0 0)
UpperWall	type	slip
LowerWall	type	adjointWallVelocity
	value	uniform (0 0 0)
FrontAndBack	type	slip
Wheeel	type	adjointWallVelocity
	value	uniform (0 0 0)

Table 5.9: *BC* for u_a

5.3.6 Adjoint pressure p_a

Inlet	type	zeroGradient
Outlet	type	adjointFarFiedlPressure
	inletValue	uniform (0)
UpperWall	type	slip
LowerWall	type	zeroGradient
FrontAndBack	type	slip
Wheeel	type	zeroGradient

Table 5.10: *BC* for p_a

5.3.7 Adjoint Spalart-Allmaras variable *nuaTilda*

Inlet	type	adjointInletNuaTilda
	value	uniform 0
Outlet	type	adjointOutletNuaTilda
	inletValue	uniform 0
UpperWall	type	slip
LowerWall	type	fixedValue
	value	uniform 0
FrontAndBack	type	slip
Wheeel	type	zeroGradient
	value	uniform 0

Table 5.11: *BC* for *nuaTilda*

For further details on the different Boundary Condition type listed refer to [23], [53] and [39].

6 RESULTS

Firstly a cube has been successfully optimized with a loop of 10 cycles. Subsequently the focus moved on an isolated, non rotating road car wheel. The first, among the wheel simulations, has been done with the setting used for the cube while in the following something has changed.

The analysis has been developed initially on *CINECA HPC Galileo (1022 compute nodes, 2x18-core Intel(R) Xeon E5-2697 v4 @2.3GHz and 128GB RAM)* and at the end on *Marconi (288 compute nodes, 68-core Intel(R) Knights Landing @1.4GHz, 96 GB RAM and 6GB MCDRAM)*.

6.1 CUBE

6.1.1 Primal and Adjoint solvers

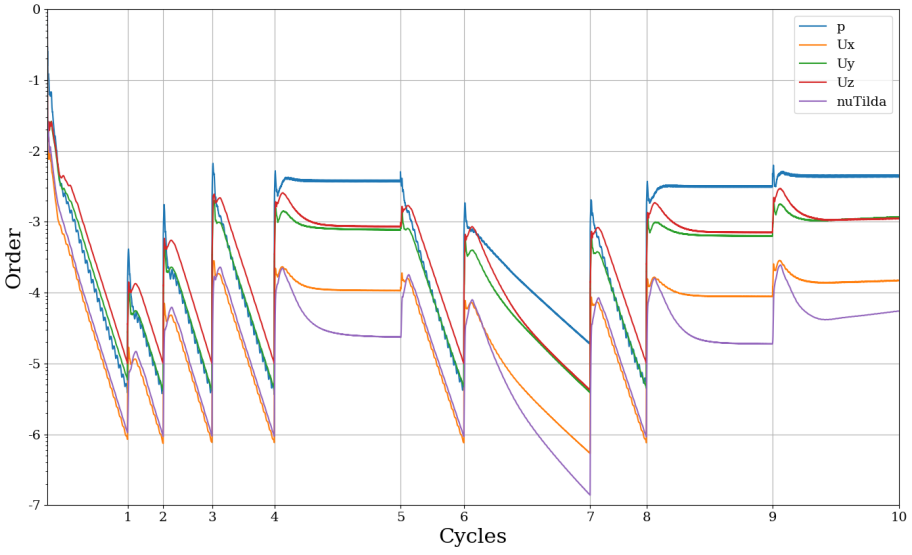


Figure 6.1: Cube *primal* solver initial residuals

Cycle	1	2	3	4	5	6	7	8	9	10
Iterations (x/7000)	4449	1981	2710	3454	7000	3522	7000	3119	7000	7000

Table 6.12: Number of iterations of the *primal* solver for each optimization cycle

Comparing Tab.6.12 with Fig.6.1 a difference among the different steps pops up. Some steps last all the 7000 iterations while other reach the stopping criteria before. However the fundamental aspect is that in each step the trend of the inistial residuals are, for the most, linear.

The "stability" and magnitude of the primal solver initial residuals are fundamental for the adjoint solver which is based on the nutSqr value calculated by the primal. Analyzing Fig.6.1 and Fig.6.2 can be noted how starting from the end of the third cycle, as the residuals of the prima solver increase, the residuals for the adjoint solver increase too.

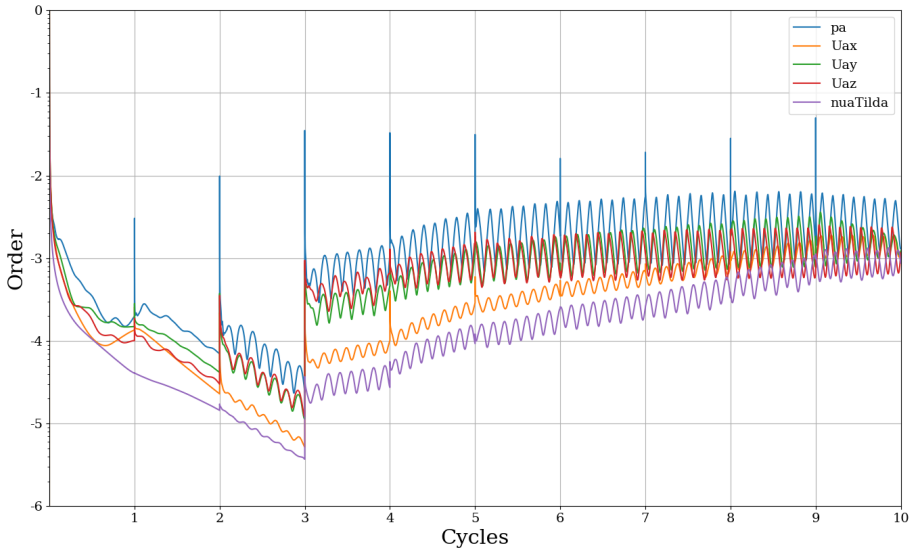


Figure 6.2: Cube *adjoint* solver initial residuals

As the cycles proceed and the geometry is modified, it's easy to understand the strong dependency of the two solvers residuals on the geometry. This is a consequence of the fact that the mesh has been developed and customized on the initial geometry so modifying the geometry also the mesh is modified but, at the moment, there's not an active control on the final mesh parameters. The simulation gives quality mesh parameters at the beginning of each cycle but a way to directly "control" them hasn't been implemented yet. As will be better explained in 6.2 in simple cases this isn't such a big problem because, even after the morphing the mesh, it maintains a good quality. In complex ones, instead,

this represents a big problem for the accuracy of the sensitivities and so for the mesh updating.

The computational time has been of $\approx 12\ h$ using 136 cores with a parallel calculation.

6.1.2 nutSqr

The objective function, which has been described in 4.2.3 by formulas is here presented numerically.

Opt. cycle	1	2	3	4	5	6	7	8	9	10
$nutSqr$	8.22	8.25	8.36	7.94	7.54	7.88	7.74	7.90	7.62	7.12
$nutSqr_{cycle}$	8.20	8.25	8.36	7.94	7.54	7.88	7.74	7.90	7.62	7.12

Table 6.13: Values of $nutSqr$ and $nutSqr_{cycle}$ to be multiplied by e^{-2}

In Tab.6.13 there are $nutSqr$, which is the value of the cost function in the last iteration of each cycle and $nutSqr_{cycle}$, which is the mean value among all the iteration of that cycle. A more panoramic view of its value can be seen in Fig.6.3 where the value in all the iteration of each cycle is presented. Observing the trend it's possible to see some positive or negative peaks at the beginning of each cycle which explain the choice of averaging nutSqr value for each cycle.

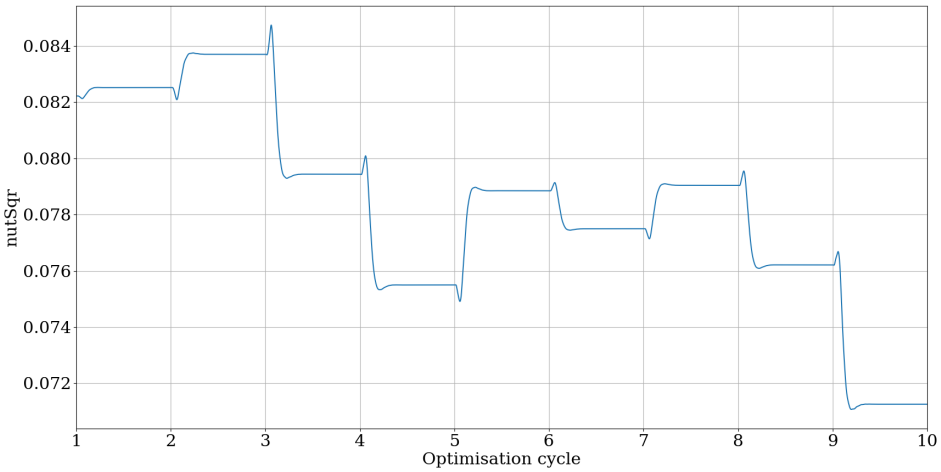


Figure 6.3: Objective function $nutSqr$

Even if in some steps the nutSqr increases because of the already mentioned limits of the mesh morphing, a decreasing trend is denoted. The initial value of 0.0820142 is at the end reduced to 0.0712472, so it means a reduction of 13,13% as visible in Fig.6.4.

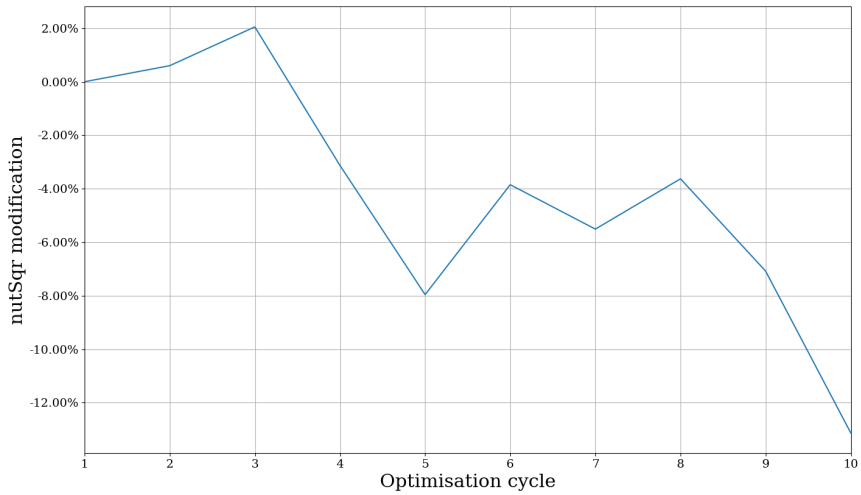


Figure 6.4: Objective function *nutSqr* reductions

6.1.3 Geometry upgrade

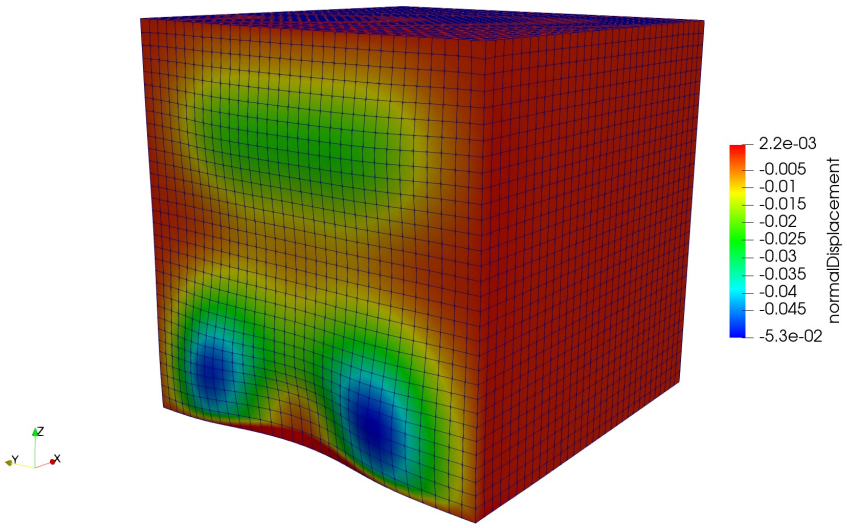
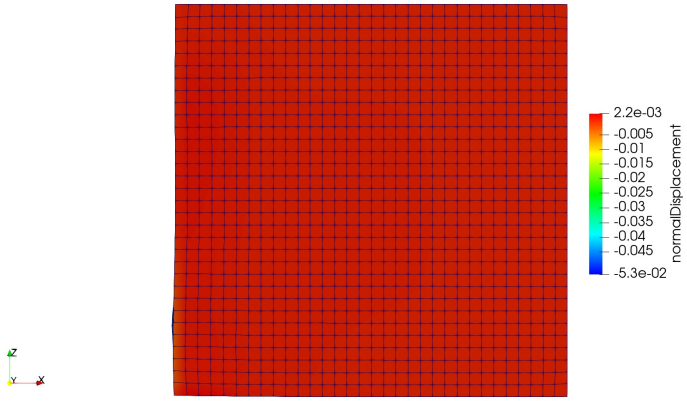
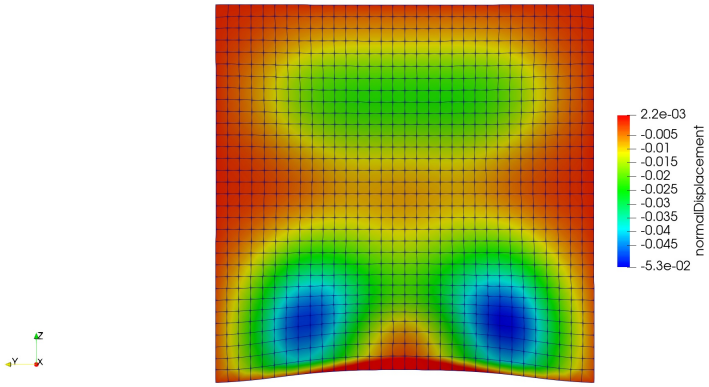


Figure 6.5: Cube optimisation results at the end of the 10^{th} cycle

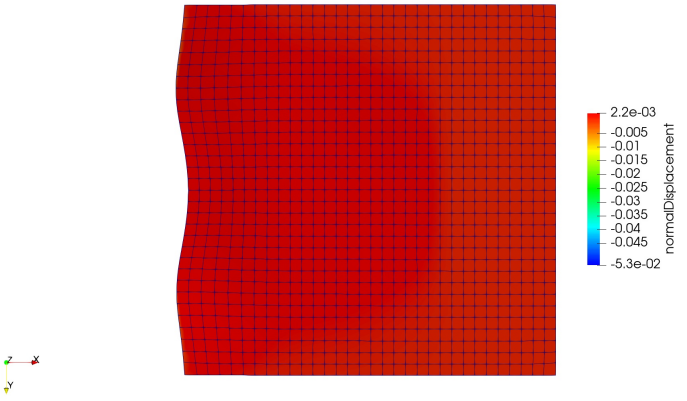
In Fig. 6.6 there will be different views of the morphed geometry at the 10^{th} cycle and in particular it will be shown the `normalDisplacement`. Geometry of cycle n means geometry modified at the beginning of cycle n basing the mesh movement on the values calculated in cycle $n-1$.



(a) *normalDisplacement xz plane*



(b) *normalDisplacement yz plane*



(c) *normalDisplacement xy plane*

Figure 6.6: Cube optimisation results at the end of the 10^{th} cycle, *plane*

6.2 WHEEL

The wheel has been studied with different settings:

- $u = (20\ \ 0\ \ 0)$, which corresponds to a suburban speed of 72 km/h (Section 6.2.1);
- $u = (41.67\ \ 0\ \ 0)$, which corresponds to a motorway speed of 150 km/h (Section 6.2.2);
- Different `nutSqrZone` with respect to the one presented in section 4.2.3 (Section 6.2.3);
- The same `nutSqrZone` as the previous case but decreasing the limitations on `controlpoints` movement (Section 6.2.4).

6.2.1 Wheel $u = (20\ \ 0\ \ 0)$

Primal and Adjoint solvers

As visible in Tab.6.14 the cycles performed all the iterations. In Fig.6.7 the trend of each cycle's initial residuals is no more linear as in the cube (Fig.6.1).

Cycle	1	2	3
Iterations	9000	5000	5000

Table 6.14: Number of iterations of the *primal* solver for each optimization cycle

Figure 6.7: *Primal* solver initial residuals

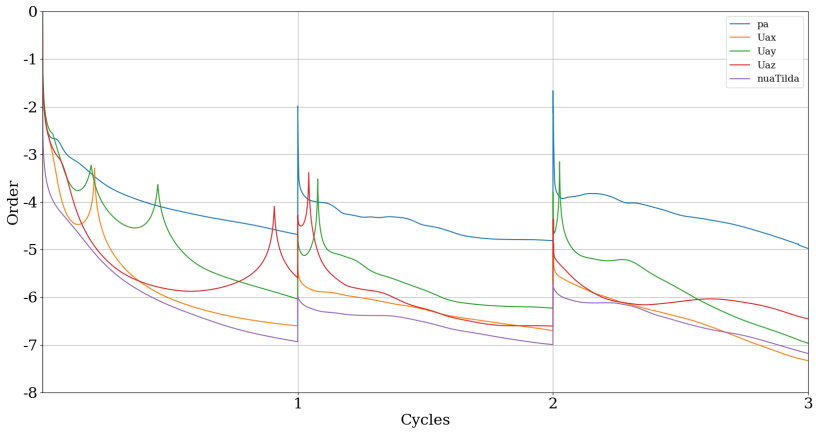


Figure 6.8: *Adjoint* solver initial residuals

The computational time has been of $\approx 14\ h$ using 136 cores with a parallel calculation.

nutSqr

Opt. cycle	1	2	3
$nutSqr$	5.90	6.51	8.34
$nutSqr_{cycle}$	2.78	6.59	8.29

Table 6.15: Values of $nutSqr$ and $nutSqr_{cycle}$ to be multiplied by e^{-4}

As show in Fig.6.9 the objective function increases its value instead of reducing it.

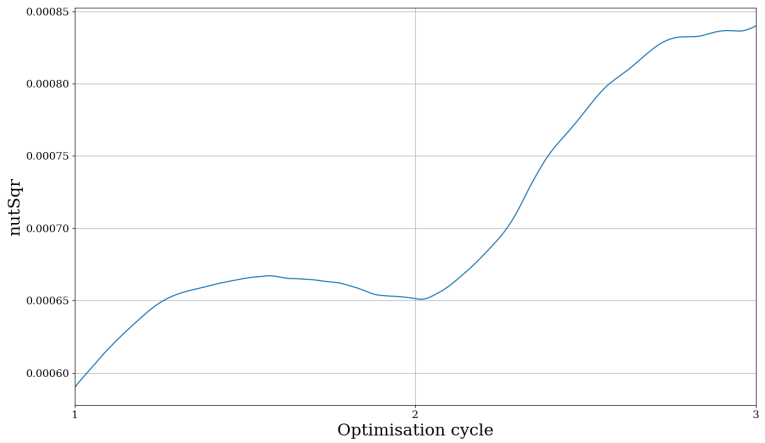
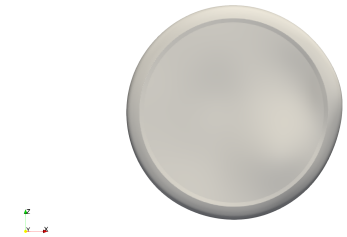
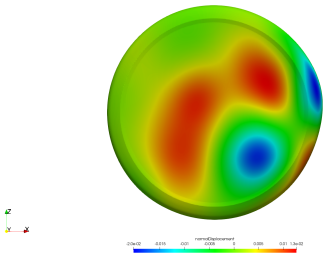


Figure 6.9: Objective function *nutSqr*

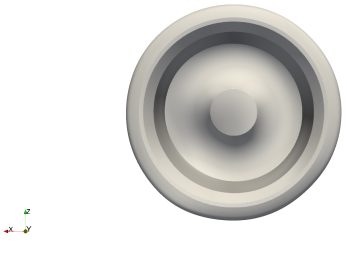
Geometry upgrade



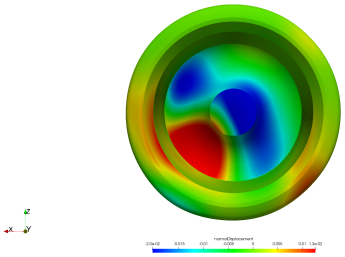
(a) *front view, solid*



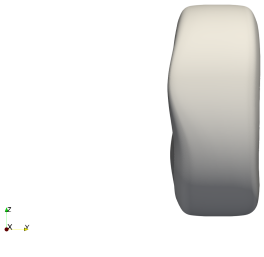
(b) *front view, normalDisplacement*



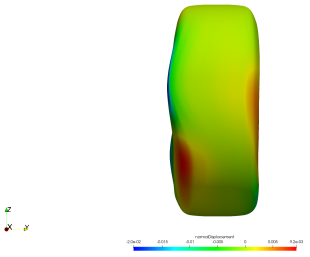
(c) *back view, solid*



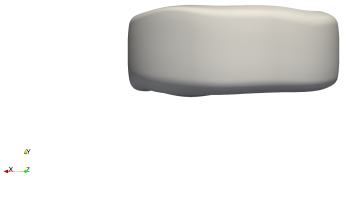
(d) *back view, normalDisplacement*



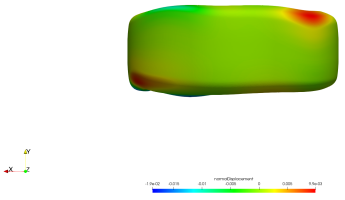
(e) *yz plane, solid*



(f) *yz plane, normalDisplacement*



(g) *xy plane, solid*

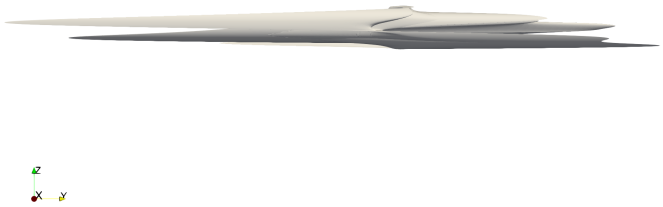


(h) *xy plane, normalDisplacement*

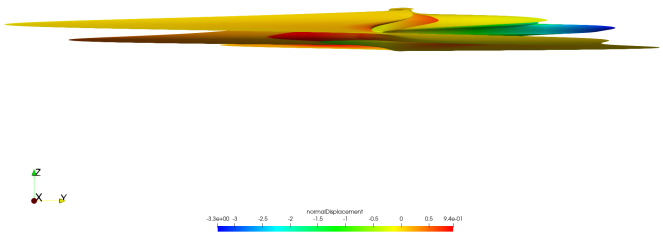
Figure 6.10: New geometry for the 3^{rd} optimisation cycle. On the left there is the geometry view while on the right side the `normalDisplacement` is visualized

(a) *solid* (b) *normalDisplacement*

Figure 6.11: New geometry for the 4^{th} optimisation cycle, xz plane. On the left there is the geometry view while on the right side the `normalDisplacement` is visualized



(a) *solid*



(b) *normaldisplacement*

Figure 6.12: New geometry for the 4^{th} optimisation cycle, yz plane. On the left there is the geometry view while on the right side the `normalDisplacement` is visualized

Fig.6.10 clearly shows how the wheel starts to be deformed in one direction. In this case, as already explained in section 6.1.3, 3^{rd} cycle geometries means the geometry modified at the beginning of *cycle 3* basing the mesh movement on the values calculated in *cycle 2* and so on. In Fig.6.11 and Fig.6.12 there is the geometry obtained by the 4^{th} cycle where how it "explodes" giving something no more linked to a conventional wheel is clearly visible.

As already mentioned at the beginning of each cycle an evaluation of the mesh quality parameters is performed and, in this case, the "extreme" mesh movement of cycle 4^{th} is denoted by the following parameters:

Aspect ratio	max	$7.502e^{+97}$
Non-orthogonality	max	179.996
	average	67.601
Skewness	max	22654.043

Table 6.16: Wheel mesh quality parameters at 4^{th} optimisation cycle

6.2.2 Wheel $u = (41.67 \;\; 0 \;\; 0)$

Primal and Adjoint solvers

As visible in Tab.6.17 the cycles performed all the iterations in this case too. In Fig.6.13 the trend of each cycle's final residuals is no more linear as in the previous case.

Cycle	1	2	3
Iterations	9000	5000	5000

Table 6.17: Number of iterations of the *primal* solver for each optimization cycle

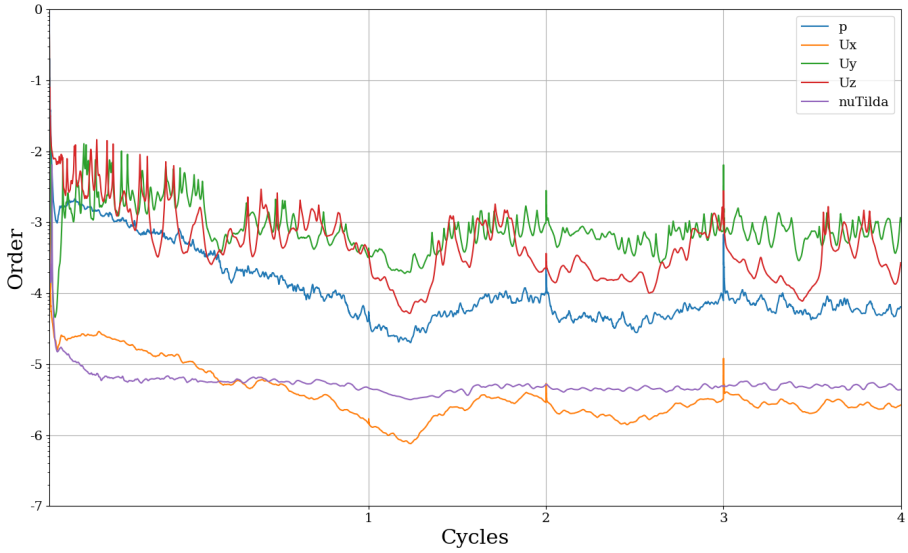


Figure 6.13: *Primal* solver initial residuals

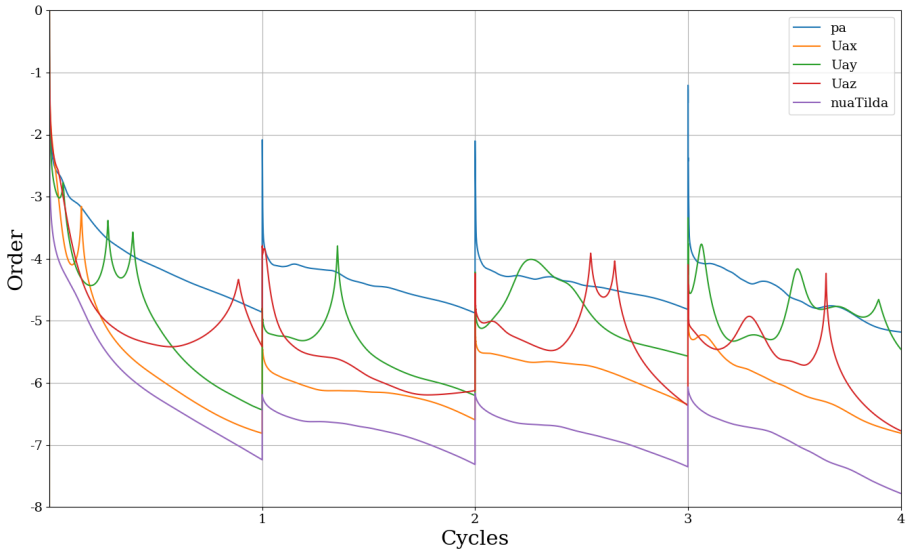


Figure 6.14: *Adjoint* solver initial residuals

The computational time has been of $\approx 18\ h$ using 136 cores with a parallel calculation.

nutSqr

The objective function, which has been described in section 4.2.3 by formulas is here presented numerically.

Opt. cycle	1	2	3	4
$nutSqr$	1.78	2.51	2.60	2.48
$nutSqr_{cycle}$	0.85	2.36	2.57	2.54

Table 6.18: Values of $nutSqr$ and $nutSqr_{cycle}$ to be multiplied by e^{-3}

From the theoretical part I understood that the aerodynamic noise enhances if speed increases.(Fig.1.6). Comparing Tab.6.18 with Tab.6.15 it has to be noted that nutSqr increases of one order of magnitude, according to theory. Observing Tab.6.18 and the nutSqr trend in Fig.6.15 can be seen that after reaching a maximum level during the third cycle the cost function start decreasing.

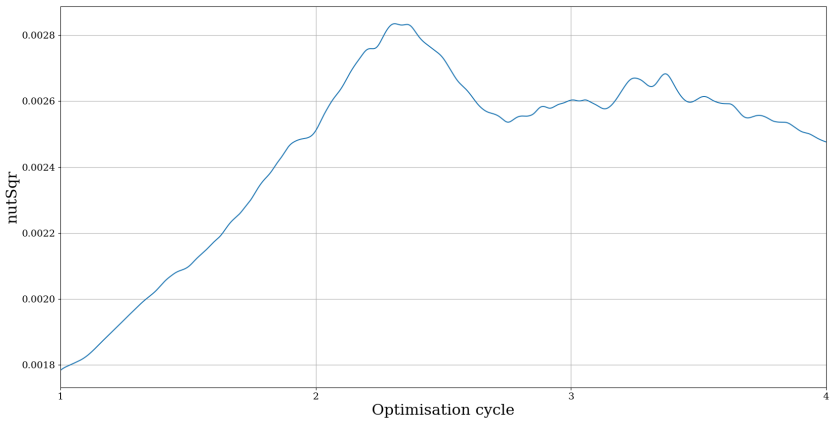
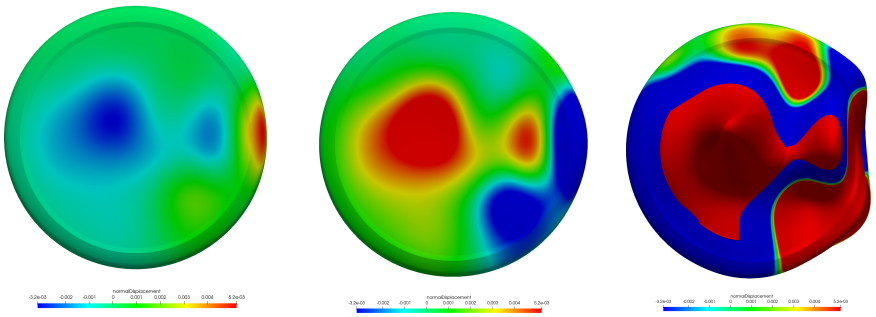


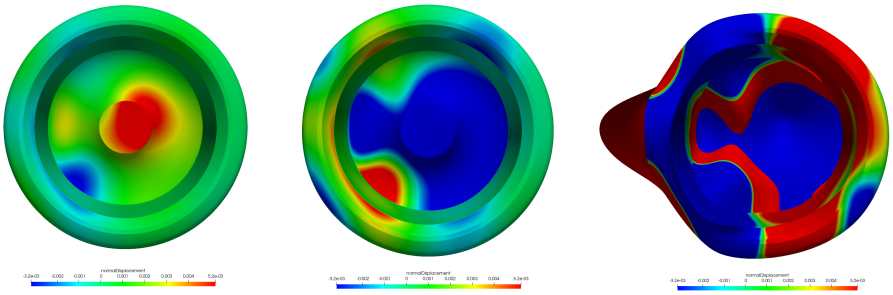
Figure 6.15: Objective function $nutSqr$

Geometry upgrade



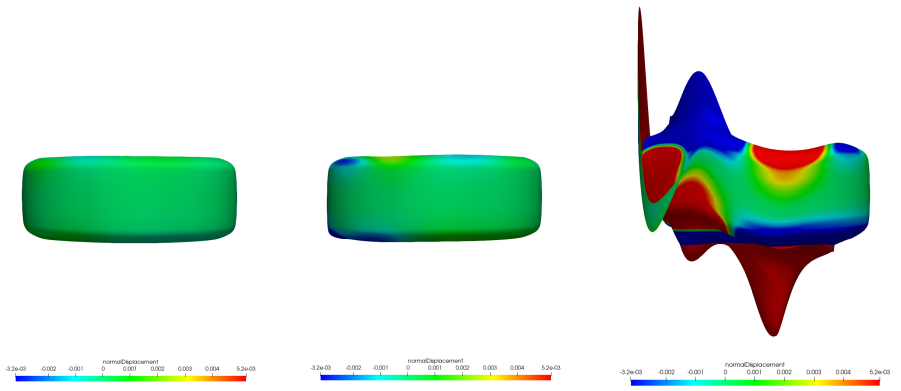
(a) *front view, 3rd cycle* (b) *front view, 4th cycle* (c) *front view, 5th cycle*

Figure 6.16: New geometry for the 3^{trd}, 4^{th} and 5^{th} optimisation cycle, front view of `normalDisplacement`



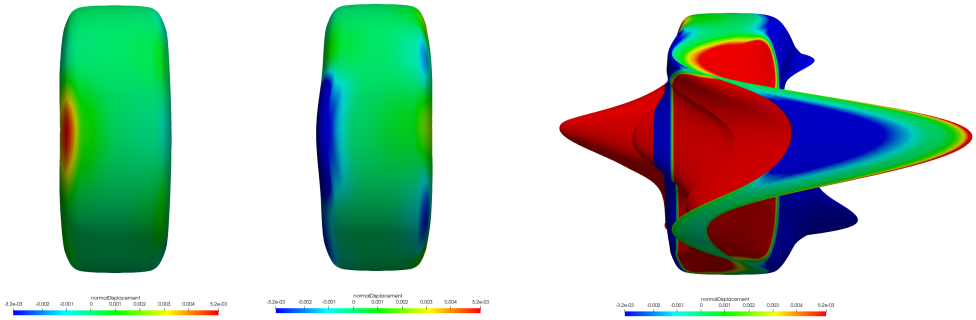
(a) *front view, 3^{rd} cycle* (b) *front view, 4^{th} cycle* (c) *front view, 5^{th} cycle*

Figure 6.17: New geometry for the 3^{rd}, 4^{th} and 5^{th} optimisation cycle, back view of `normalDisplacement`



(a) *xy plane view, 3^{rd} cycle* (b) *xy plane view, 4^{th} cycle* (c) *xy plane view, 5^{th} cycle*

Figure 6.18: New geometry for the 3^{rd}, 4^{th} and 5^{th} optimisation cycle, xy plane view of `normalDisplacement`



(a) *yz plane view, 3^{rd} cycle* (b) *yz plane view, 4^{th} cycle* (c) *yz plane view, 5^{th} cycle*

Figure 6.19: New geometry for the 3^{rd}, 4^{th} and 5^{th} optimisation cycle, yz plane view of `normalDisplacement`

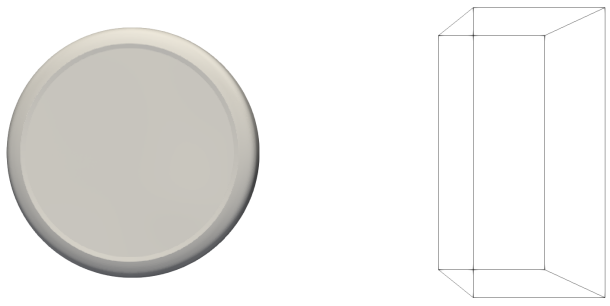
Looking at the previous figures (Fig.6.16 - 6.19) has to be noted that there is an alternation between inwards and outwards movement. The zones that in the 3^{rd} cycle has been moved in one direction, in the 4^{th} has been moved in the opposite. It's the same passing from 4^{th} to 5^{th} cycle. The last cycle, as in the previous case for the 4^{th}, hasn't been performed because of the worst geometry update. The 5^{th} cycle mesh quality parameters, in fact, are the follows:

Aspect ratio	max	$3.833e^{+96}$
Non-orthogonality	max	179.999
	average	43.897
Skewness	max	273283.056

Table 6.19: Wheel mesh quality parameters at 5^{th} optimisation cycle

6.2.3 Wheel - nutSqrZone

This case has the same setting of the baseline (section 6.2.1) but I modified the nutSqrZone with a smaller and closer one to test its influence on the results. The new nutSqrZone is $0.3 \times 0.6 \times 0.7$ m.



(a) *xz plane*

(b) *xy plane*

Figure 6.20: New nutSqrZone for the wheel

Primal and Adjoint solvers

Cycle	1	2	3
Iterations	9000	5000	5000

Table 6.20: Number of iterations of the *primal* solver for each optimization cycle

Figure 6.21: *Primal* solver initial residuals

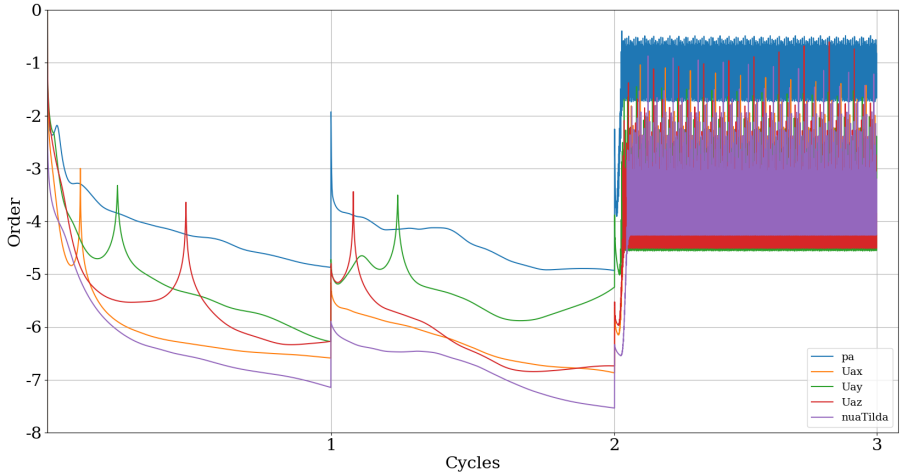


Figure 6.22: *Adjoint* solver initial residuals

Observing Fig.6.22 the non convergence of the *adjoint* solver is explained by the instability of its initial residuals.

The computational time has been of $\approx 14\ h$ using 136 cores with a parallel calculation.

nutSqr

Opt. cycle	1	2	3
$nutSqr$	1.52	2.01	1.60
$nutSqr_{cycle}$	1.41	1.87	1.61

Table 6.21: Values of $nutSqr$ and $nutSqr_{cycle}$ to be multiplied by e^{-5}

Comparing the values in Tab.6.21 with the baseline ones in Tab.6.14 has to be noted how the reduction of the nutSqrZone reduce the nutSqr values of one order of magnitude.

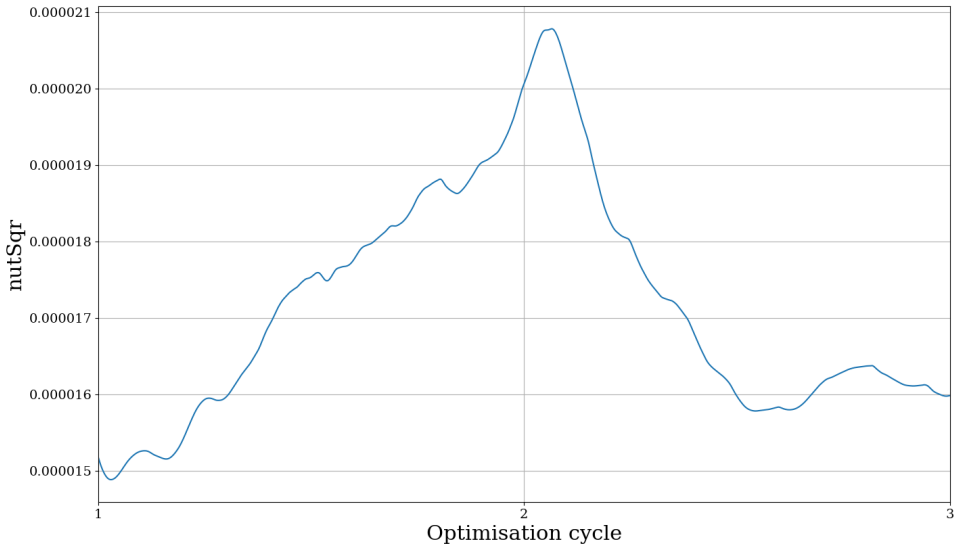
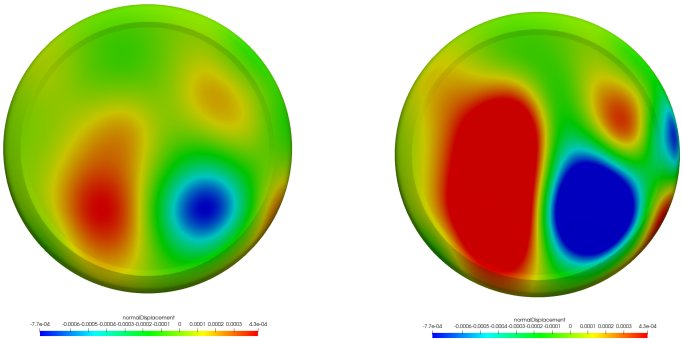


Figure 6.23: Objective function *nutSqr*

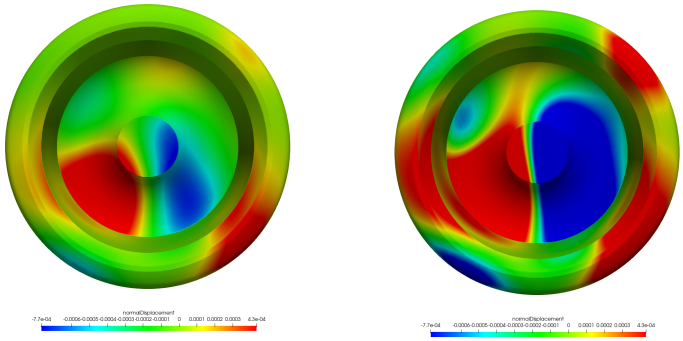
Geometry upgrade

In this case only the front and back views will be presented because the geometry modifications are so small that can't be seen in pictures.



(a) *front view, 2^{nd} cycle* (b) *front view, 3^{rd} cycle*

Figure 6.24: New geometry for the 2^{nd} and 3^{th} optimisation cycle, front view of `normalDisplacement`



(a) *front view, 2^{nd} cycle* (b) *front view, 3^{rd} cycle*

Figure 6.25: New geometry for the 2^{nd} and 3^{th} optimisation cycle, back view of `normalDisplacement`

Comparing this sensitivity map evolutions (Fig.6.24, 6.25) with the previous case ones (Fig.6.16 - 6.19) has to be noted how the `normalDisplacement` has the same trend passing from a cycle to the next. Areas in blue/red remain in blue/red there, is no more the alternation seen before but a gradual increasing.

6.2.4 Wheel - free `controlpoints`

After obtaining a unidirectional geometry modification with the previous setting, the idea has been to correct what was gone wrong maintaining the new `nutSqrZone`. Considering that the `maxAllowedDisplacement` was maintained equal to $2e^{-3}$ as for the first two simulations, the small movement of the geometry for the previous case let me think to relax the constraints of the points as in Fig.6.26.

```
box
{
    type    cartesian;
    nCPsU   10;
    nCPsV   6;
    nCPsW   10;
    degreeU 3;
    degreeV 3;
    degreeW 3;

    controlPointsDefinition fromFile;

    confineUMovement false;
    confineVMovement false;
    confineWMovement false;
    confineBoundaryControlPoints false;

    confineUMinCPs ( (true true true) (true true true) );
    confineUMaxCPs ( (true true true) );
    confineVMinCPs ( (true true true) (true true true) );
    confineVMaxCPs ( (true true true) (true true true) );
    confineWMinCPs ( (true true true) );
    confineWMaxCPs ( (true true true) (true true true) );
}
```

Figure 6.26: `dynamicMeshDict` settings

Primal and Adjoint solvers

Cycle	1	2	3	4	5	6
Iterations	9000	5000	5000	5000	5000	5000

Table 6.22: Number of iterations of the *primal* solver for each optimization cycle

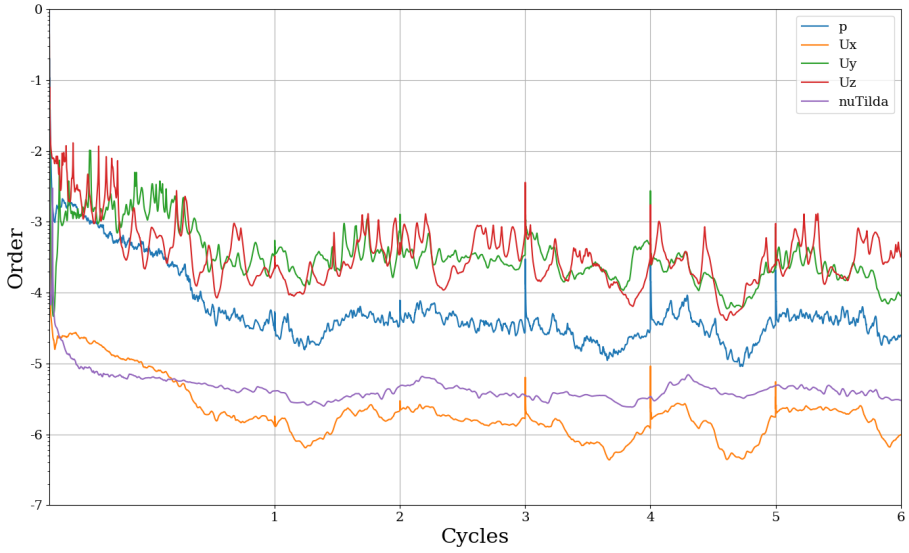


Figure 6.27: *Primal* solver final residuals

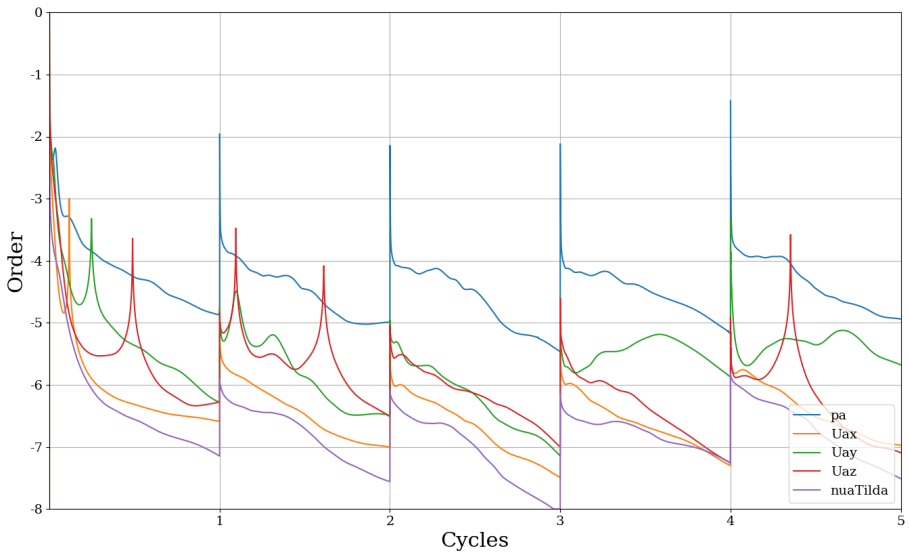


Figure 6.28: *Adjoint* solver final residuals

Using 136 cores with a parallel calculation the computational time reached the daily limit of CINECA fixed at 24 h .

nutSqr

Opt. cycle	1	2	3	4	5	6
$nutSqr$	1.52	1.78	1.62	1.98	1,95	1.92
$nutSqr_{cycle}$	1.41	1.78	1.58	1.84	1.79	1.84

Table 6.23: Values of $nutSqr$ and $nutSqr_{cycle}$ to be multiplied by e^{-5}

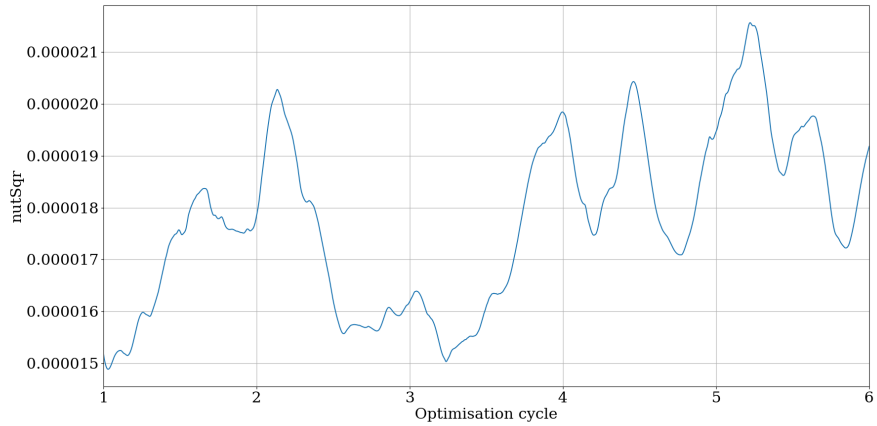


Figure 6.29: Objective function $nutSqr$.

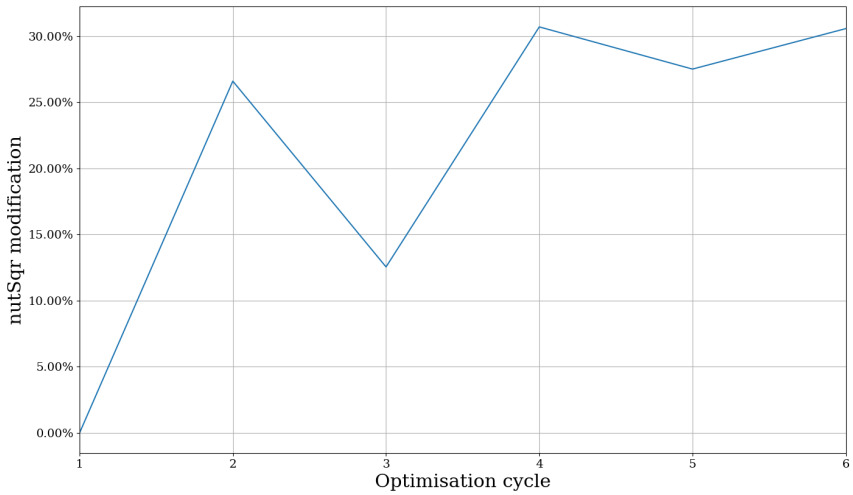
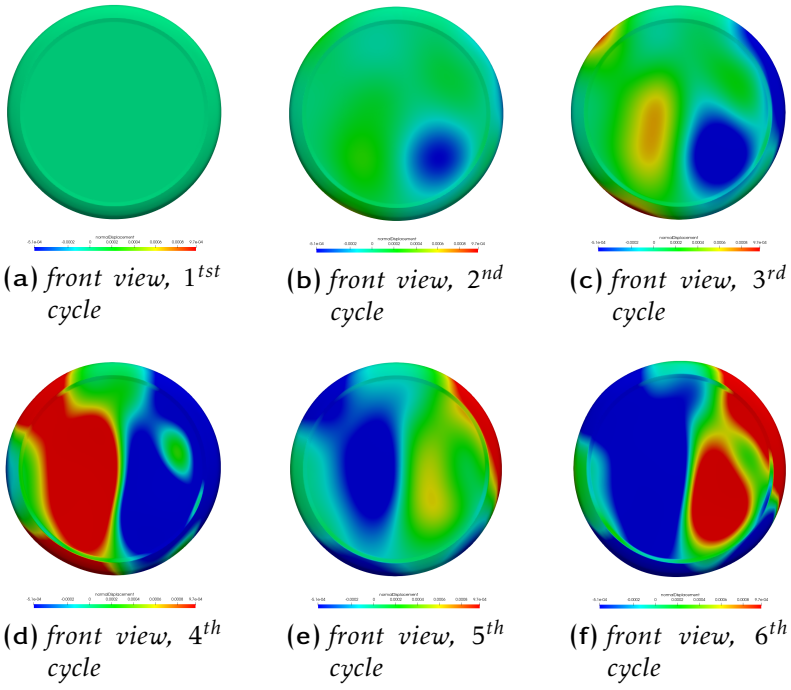


Figure 6.30: Objective function *nutSqr* percentage of modification

In Fig.6.29 and Fig.6.30 has to be noted, once again, how the nutSqr value oscillates cycle by cycle. It increases up to 30% more of the initial configuration.

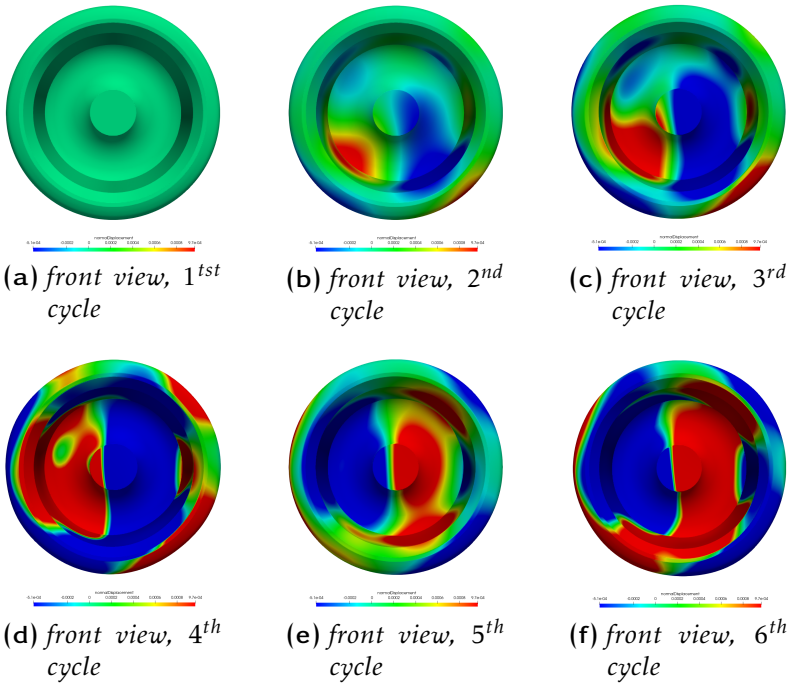
Geometry upgrade



(a) *front view, 1^{tst} cycle* (b) *front view, 2nd cycle* (c) *front view, 3rd cycle*

(d) *front view, 4th cycle* (e) *front view, 5th cycle* (f) *front view, 6th cycle*

Figure 6.31: New geometry for the 6 optimisation cycles, front view of `normalDisplacement`



(a) *front view, 1^{tst} cycle*

(b) *front view, 2^{nd} cycle*

(c) *front view, 3^{rd} cycle*

(d) *front view, 4^{th} cycle*

(e) *front view, 5^{th} cycle*

(f) *front view, 6^{th} cycle*

Figure 6.32: New geometry for the 6 optimisation cycles, back view of `normalDisplacement`

6.3 FINAL REMARKS

Analizing the results of the different simulations for the wheel I noted that they have in common the oscillation of `nutSqr` and its non-reduction. The oscillatory behaviour has as consequence the movement , among the cycles, in one sense and in the opposite. This phenomena which is particularly visible in the *Geometry upgrade* sections of 6.2.2 and 6.2.4 where, in the `normalDisplacement` maps, same areas have opposite values going from one cycle to the other.

Comparing all the results obtained in the wheel and the cube simulations there is a substantial difference in the *primal* solver stability convergence. The *primal* solver initial residuals in the cube are more smooth and linear with respect all the wheel ones.

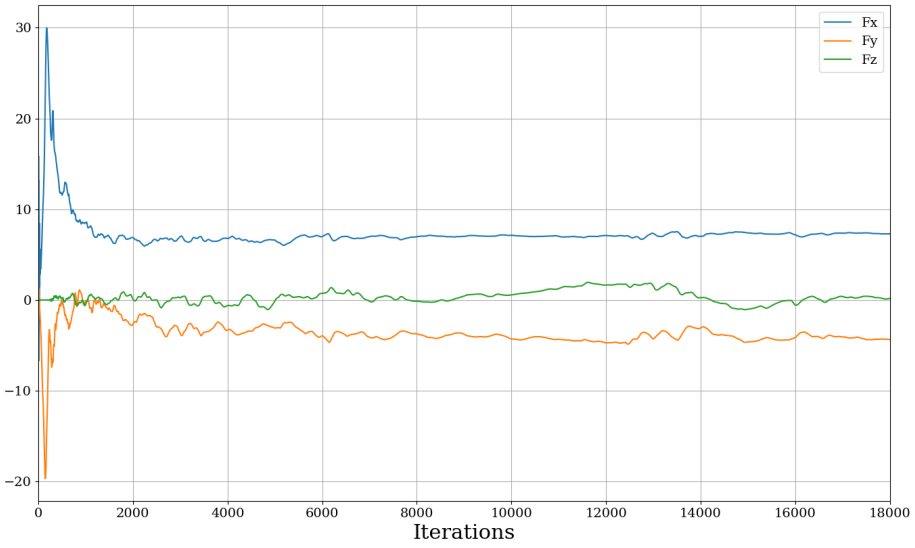
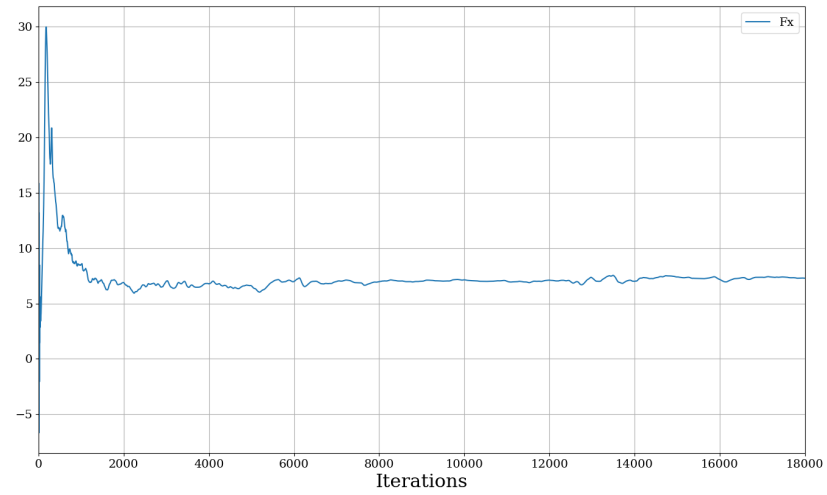
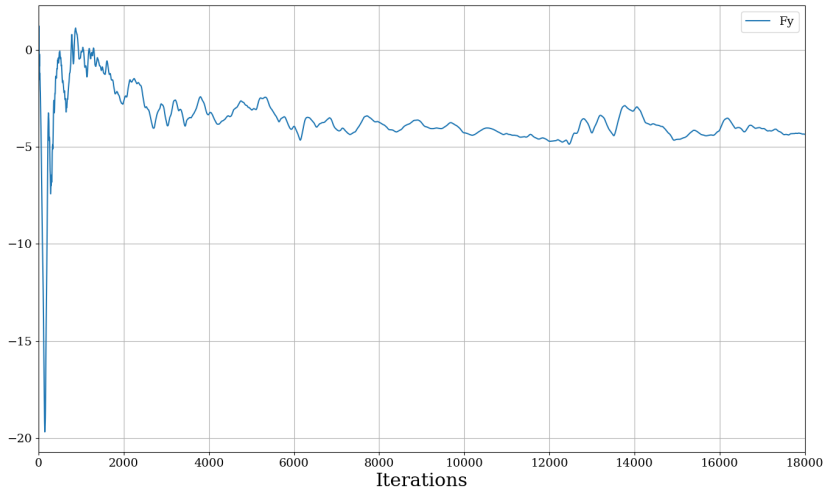


Figure 6.33: Forces trends of flow around the wheel.

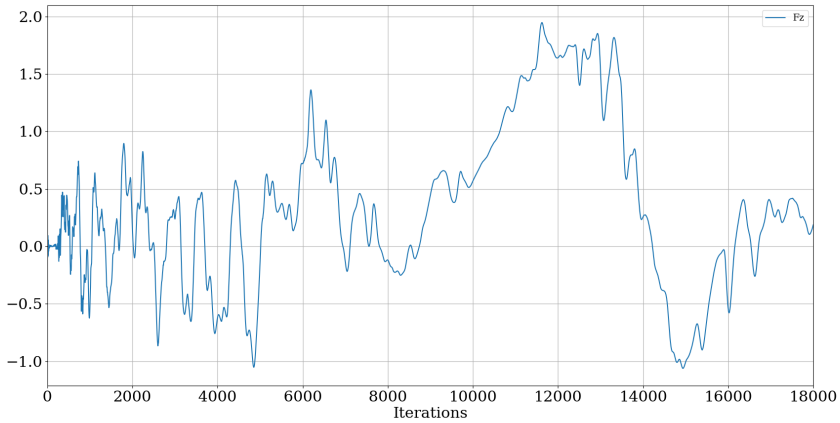
Observing the forces sampling for the wheel in Fig.6.33 is clearly visible how their trend is not linear and far from convergence. This observation suggested me that a problem was occurring with the use of `simpleFoam` and matching Fig.6.34 and Fig.6.35 is clear how it worked better for the cube than for the wheel.



(a) F_x

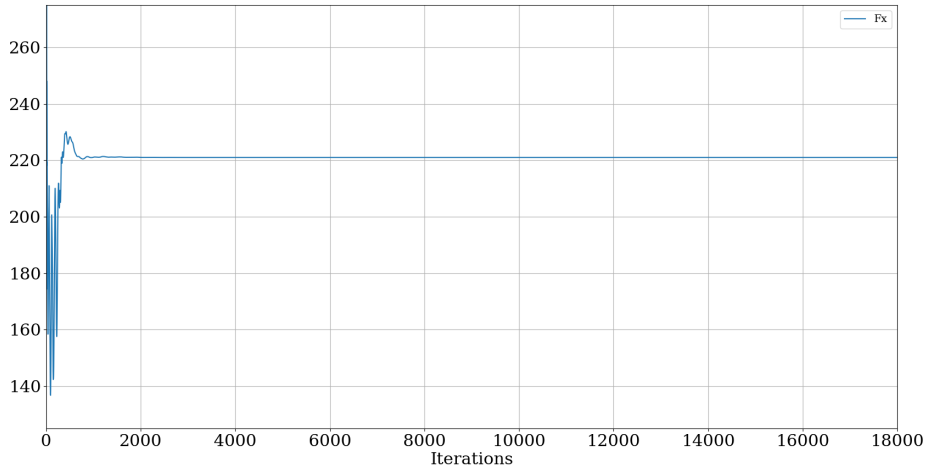


(b) F_y

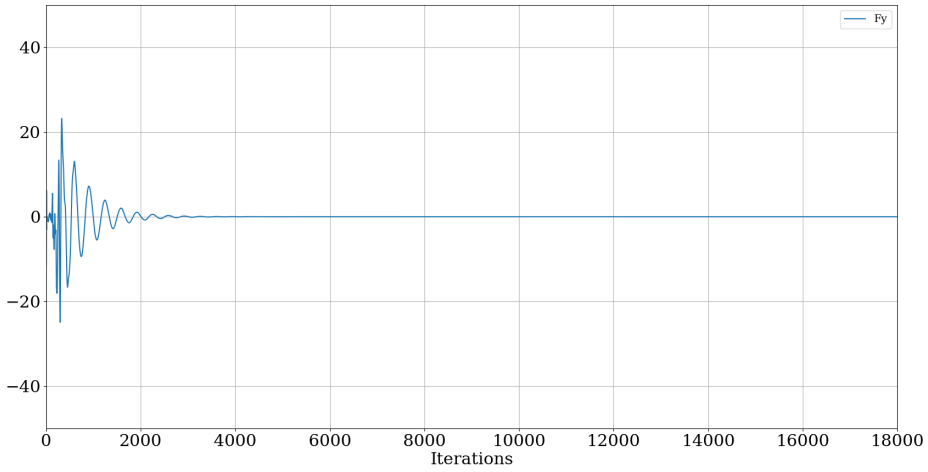


(c) F_z

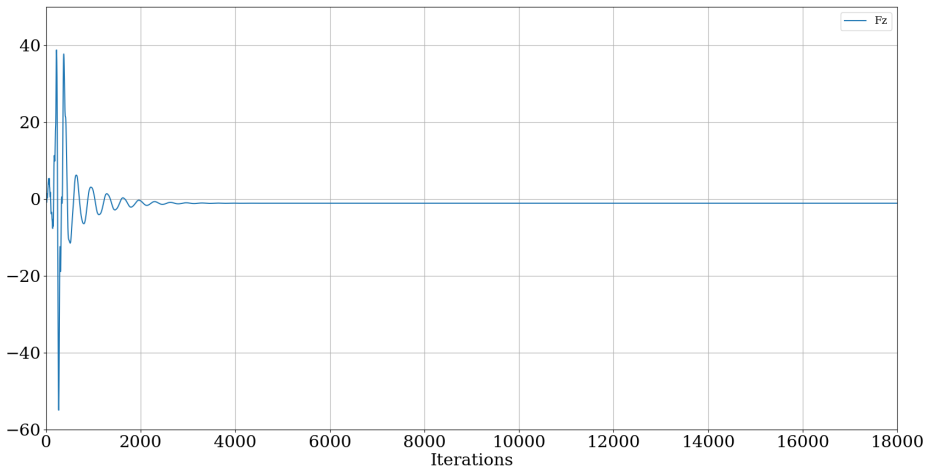
Figure 6.34: Force components of flow around the wheel



(a) F_x



(b) F_y



(c) F_z

Figure 6.35: Force components of flow around the cube

Considering that `simpleFoam` has been set in the same way in both the simulations the difference must be in the flows.
The cube has edges that impose the separations points so the flow is non-stationary but more stable than the one around the wheel. Performing a simulation with a stationary solver is enough for the cube but in the wheel case the simulation can't converge. This is also the answer to the non optimization of the cost function. If the *primal* solver, that calculates the cost function, doesn't converge, it gives to the *adjoint* solver wrong values and so this results in wrong sensitivities.

The difference in the two flows is understandable also looking at the effect of reducing the volume of `nutSqrZone` in sections 6.2.3 and 6.2.4. Speed and the other parameters are the same of section 6.2.1 but the cost function is calculated on a smaller and nearer domain so that a minor part of the wake is caught.

7 CONCLUSIONS

The main objective of this thesis was the developing and testing of a continuous adjoint method for shape optimization applied to noise reduction. This methodology not only provides a sensitivity maps but it foresees the mesh morphing too. The mesh movement refers on the sensitivities calculated and it is based on a previous mesh parametrization through *B-splines*. All the simulations has been performed in OpenFoam and post processed with Matlab and Python scripts. This analysis was heavy for computational cost but it has been possible thanks to CINECA's HPC. The objective function based on the *turbulent viscosity* found in literature was defined through some experimental tests. Thus I analyzed its theoretical background and performed a CFD simulation to show its correlation with *kinetic energy* and *Proudman acoustic power*. A cube has been successfully optimized with a loop of 10 cycles. Considering the maximum allowed displacement of the mesh, the noise reduction of 13% with respect to the initial geometry is remarkable. The same setting has been applied to a wheel without a satisfactory result. Subsequently I performed three other simulations with three different settings. I tested the effects of increasing the free velocity and it resulted in an overall minor cost function values but without an effective optimization. After noticing the oscillatory movement of the mesh, the next step has been to modify the cost function domain of integration (`nutSqrZone`) which gave a monotone mesh modification but it performed only 3 cycles. In the end the `controlpoints` were let free to move. The *adjoint* solver, thanks to the constraints relax, resulted to be more stable and so more cycles has been performed. From the 4^{th} to the 5^{th} cycle the mesh movement direction changed representing the oscillatory movement. This kind of recurring mesh movement let me think about the complexity of a wheel wake with respect to the cube. As already state at the end of chapter 6, I end up stating that `simpleFoam` can't converge for the wheel due to its wake's high non-stationary behavior.

The approach followed in this thesis is very worthwhile for optimizations. It reduces the computational cost and the total time with respect to a classic optimization process based on the running of several simulations interspersed by manual modifications of the geometry. Thus, the mesh morphing process is remarkable because not only the simulation gives a sensitivity map but also a new geometry based on it. In this thesis I highlighted the difficulties of dealing with aeroacoustic optimization based on the adjoint method, providing a methodology to approach the problem. The results are not satisfactory at all for complex flows, but I believe that this approach could improve our design knowledge on aeroacoustic optimization.

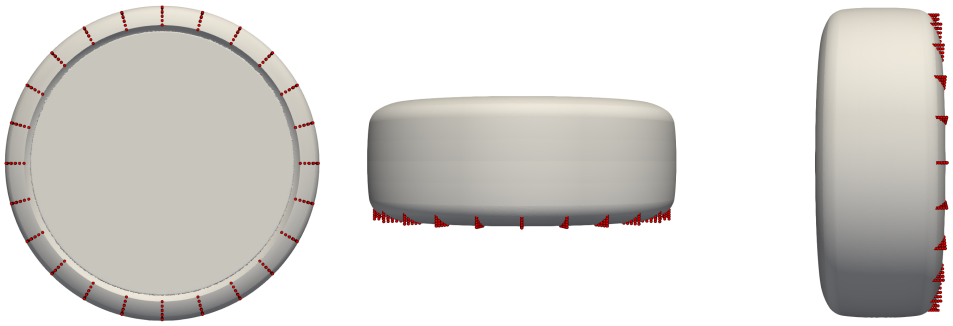
7.1 FUTURE DEVELOPMENT

Starting from my work there could be a lot of developments and improvements. For instance performing a non-stationary simulation on the wheel to better understand its wake complexity.

As already inquired by [34], other steps could be to apply this method to an open rim wheel, a grounded wheel or a rotating wheel and study how these modifications affect the optimization process.

Further interesting work could be the idea of inserting a vehicle geometry like *DriveAer* and to check how the wheel arch presence influences the hypothesis done on the wheel wake. In chapter 3 I briefly explained the basis of computational aeroacoustics, so could be newsworthy to perform a CAA simulation of the first and optimized geometries; or even better to directly interface the two kinds of simulations.

Focusing more on the code itself I think that the updating mesh control could be improved. In order to have a better control and a more realistic mesh modification, I developed a *Python* code to generate control points in a circular crown (Fig.7.1) instead of a box. The next step could be to find a way to let this new definition of points link with the points management in the `meshDict` file.



(a) *front view* (b) *xy plane* (c) *yz plane*

Figure 7.1: Circular crown of control points

This thesis has not the aim to be a turnkey work but a starting point for aeroacoustic optimization based on the adjoint method.

BIBLIOGRAPHY

[1] BloombergNEF. *Electric Vehicle Outlook 2021*. 2021. URL: https://www.theguardian.com/environment/2021/jan/19/global-sales-of-electric-cars-accelerate-fast-in-2020-despite-covid-pandemic.

[2] Ansa. *Boom vendite auto elettriche, fino a 230 milioni nel 2030*. 2021. URL: https://www.ansa.it/canale_motori/notizie/eco_mobilita/2021/04/29/boom-vendite-auto-elettriche-fino-a-230-milioni-nel-2030_2224f1ea-fe86-4e9a-9287-132451b9a162.html.

[3] TheGuardian. *Global sales of electric cars accelerate fast in 2020 despite pandemic*. 2021. URL: https://www.theguardian.com/environment/2021/jan/19/global-sales-of-electric-cars-accelerate-fast-in-2020-despite-covid-pandemic.

[4] Council of European Union, ed. *Environmental noise*. 2020. URL: https://ec.europa.eu/environment/noise/index_en.htm.

[5] European Environment Agency, ed. *Noise in Europe 2014*.

[6] Helfer, M. "General aspects of vehicle aeroacoustics". In: Lecture series-Von Karman Institute for Fluyd Dynamics. Von Karman Institute. 2005.

[7] Haider, M., Conter, M., and Wehr, R. *Tyre/road noise at low speeds*. FEHRL.

[8] *Final Report SI2.408210 Tyre/Road Noise – Volume 2*.

[9] Shetty, Neeti. "Analysis of the A-Pillar". MA thesis. Linköping University Division for Applied Thermodynamics and Fluid Mechanics, 2018.

[10] MA thesis.

[11] Ask, J. and Davidson, L. "The Sub-critical Flow past a Generic Side Mirror and its Impact on Sound Generation and Propagation". In: 12th AIAA/CEAS Aeroacoustics Conference. Cambridge, Massachusetts, May 2006.

[12] Michelin. *The tyre Mechanical and acoustic comfort*. Societe de Technologie Michelin, 2002.

[13] Sandberg, U. "A Closer Look at the Tread Groove Resonance in Tyre/Road Noise". In: International congress on acoustics. 2004.

[14] Kindt, P. *Tire-Road Noise, Basics and an outlook to further reductions of road traffic noise*.

[15] Wäschle, A. *Numerische und experimentelle Untersuchung des Einflusses von drehenden Rädern auf die Fahrzeugaerodynamik*. 2006. Chap. 2.

[16] Pope, S.B. *Turbulent Flows*. Cambridge University Press, 2000.

[17] Bakker, A. *Computational Fluid Dynamics*. 2008.

[18] Johansson, A. and Wallin, S. *An introduction to turbulence*. Royal Institute of Technology, 2013.

[19] Spalart P., R. and Rumsey C., L. "Effective Inflow Conditions for Turbulence Models in Aerodynamic Calculations". In: *AIAA Journal* (2007).

[20] Johansson, E. "Aeroacoustic study on the roofbar of a track using CFD". MA thesis. Chalmers University Of Technology, 2013.

[21] Jones, D.A. et al. *RANS Simulations using OpenFOAM Software*. 2016.

[22] Spalart P., R. and Allmaras S., R., eds. *A one-eqution turbulence model for aerodynamic flows*. Vol. 1. 1994, pp. 5–21.

[23] Group, ESI. *User Guide*. 2021. URL: https://www.openfoam.com/documentation/user-guide.

[24] Larsson, J. "Computational Aeroacoustics for Vehicle Applications". MA thesis. Chalmers University of Technology, 2002.

[25] Sjöberg, E. "Implementation of Aeroacoustic Methods in OpenFOAM". MA thesis. KTH Kungliga Tekniska Högskolan, 2016.

[26] Gloerfelt, X., Juvé, D., and Bailly, C. "Direct computation of the noise radiated by a subsonic cavity flow and application of integral methods". In: *Journal of Sound and Vibration* 266 (2003), pp. 119–146.

[27] Wagner, C., Hütti, T., and Sagaut, P. *Large-eddy simulation for acoustics*. Cambridge Aerospace Series. 2007.

[28] Bose, T. *Aerodynamic Noise, An Introduction for Physicists and Engineers*. Ed. by Springer. Ret., Indian Institute of Technology Madras, 2013.

[29] Carley, M. *Turbulence and noise*. University of Bath, 2011.

[30] Mankbadi R., R. et al. "Direct Computation of Jet Noise Produced by Large-Scale Axisymmetric Structures". In: *Journal of Propulsion and Power* 16 (2000), pp. 207–215.

[31] Lighthill M., J. "On sound generated aerodynamically I. General theory". In: 211 (1952). Ed. by Society, Royal, pp. 564–587.

[32] Ask, J. *Predictions of aerodynamically induced wind noise around ground vehicles.* 2008.

[33] Hirscheberg, A. and Rienstra, S. *An introduction to acoustics*. Eindhoven University of Technology, 2021.

[34] Ringwall, E. "Aeroacoustic sound sources around the wheels of a passenger car". MA thesis. Chalmers University Of Technology, 2017.

[35] Korchagova, V., Kraposhin, M., and Strizhak, S. *Computational Aeroacoustics Methods with OpenFOAM v. 4.1*. Institute for System Programming of Russian Academy of Sciences, Web-laboratory UniCFD, 2017.

[36] Farassat, F. *Derivation of Formulations 1 and 1A of Farassat*. NASA Langley Research Center, Hampton, Virginia, 2007.

[37] Proudman, I., ed. *The Generation of Noise by Isotropic Turbulence*. Vol. 214. Mathematical and Physical Sciences. Royal Society, 1952, pp. 119–132.

[38] Nadarajah S., K. and Jameson, A. "A comparison of the continuous and discrete adjoint approach to automatic aerodynamic optimization". In: *Aiaa-2000-0667* (2000).

[39] Papoutsis-Kiachagias E., M. and Giannakoglou K., C. "Continuous Adjoint Methods for Turbulent Flows, Applied to Shape and Topology Optimization: Industrial Applications". In: *Arch Computat Methods Eng* 23 (2014), pp. 255–299.

[40] Anderson W., K. and Venkatakrishnan, V. "Aerodynamic design optimization on unstructured grids with a continuous adjoint formulation". In: *Computers Fluids* 28.4 (1999), pp. 443–480.

[41] Papadimitriou D., I. and Giannakoglou K., C. "A continuous adjoint method with objective function derivatives based on boundary integrals, for inviscid and viscous flows". In: *Computers Fluids* 36.2 (2007), pp. 325–341.

[42] Jameson, A., Martinelli, L., and Pierce, N. A. "Optimum aerodynamic design using the Navier-Stokes equations". In: *Theoretical and Computational Fluid Dynamics* 10.1-4 (1998), pp. 213–237.

[43] Castro, C. et al. "A Systematic Continuous Adjoint Approach to Viscous Aerodynamic Design on Unstructured Grids". In: *AIAA Journal* 45 (2007), pp. 2125–2150.

[44] Zymaris A., S. et al. "Continuous adjoint approach to the Spalart–Allmaras turbulence model for incompressible flows". In: *Computers Fluids* 38.8 (2009), pp. 1528–1538.

[45] Bueno-Orovio, A. et al. "Continuous adjoint approach for the Spalart-Sllmaras model in aerodynamic optimization". In: *AIAA Journal* 50.3 (2012), pp. 631–646.

[46] Papoutsis-Kiachagias E., M. and Giannakoglou K., C. "Continuous adjoint methods for turbulent flows, applied to shape and topology optimization: Industrial applications". In: *Archives of Computational Methods in Engineering* (2014), pp. 1–45.

[47] Papoutsis-Kiachagias E., M. et al. "Noise reduction in car aerodynamics using a surrogate objective function and the continuous adjoint method with wall functions". In: *Computers Fluids* 122 (2015), pp. 223–232.

[48] Giannakoglou K., C., Papoutsis-Kiachagias E., M., and Gkaragkounis K., T. *adjointOptimisationFoam, an OpenFOAM-based optimisation tool*. Parallel CFD Optimization Unit, School of Mechanical Engineering, National Technical University of Athens. 2020.

[49] Moukalled, F., Mangani, L., and Darwish, M. *The Finite Volume Method in Computational Fluid Dynamics, An Advanced Introduction with OpenFOAM® and Matlab®*. Ed. by Springer. 2015.

[50] Garcia Rodriguez L., F. "The Continuous Adjoint Turbulent Solver for Topology Optimization "adjointOptimisationFoam", Industrial Approach". In: Chalmers University, Sweden. 2019.

[51] Martin J., M. et al. "Volumetric b-splines shape parametrization for aerodynamic shape design". In: *Aerospace Science and Technology* 37 (2014), pp. 26–36.

[52] Piegl, L and Tiller, W. *The NURBS Book*. Ed. by Springer. 1997.

[53] Greenshields C., J. *OpenFoam user guide*. The OpenFoam Foundation. 2020.