CHALMERS

Detached Eddy Simulation for Aerospace Applications

GONZALO MONTERO VILLAR

Department of Mechanics and Maritime Sciences
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden November 2021

THESIS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY IN THERMO AND FLUID DYNAMICS

Detached Eddy Simulation for Aerospace Applications

GONZALO MONTERO VILLAR

Department of Mechanics and Maritime Sciences
CHALMERS UNIVERSITY OF TECHNOLOGY

Göteborg, Sweden November 2021

Detached Eddy Simulation for Aerospace Applications
GONZALO MONTERO VILLAR
ISBN 978-91-7905-574-5

Doktorsavhandlingar vid Chalmers tekniska högskola
Ny serie nr. 5041
ISSN 0346-718X
Department of Mechanics and Maritime Sciences
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone: +46 (0)31-772 1000

Chalmers Reproservice
Göteborg, Sweden November 2021

Detached Eddy Simulation for Aerospace Applications
GONZALO MONTERO VILLAR
Department of Mechanics and Maritime Sciences
Chalmers University of Technology

Abstract

With the continuous growth in air traffic that we see nowadays, comes an increase in the requirements needed to be satisfied in order to certify an aircraft for operation. These stricter regulations affect aspects such as CO_2 emissions, sound pollution and so on, pushing manufacturers to aim for lighter, more efficient, more robust designs. These improvements might be achieved in two different ways; by improving/optimizing existing technology, or by developing new technological concepts. In either of the two scenarios, numerical tools, such as optimization methods or reliable fluid flow simulations play a paramount role.

In this thesis, new functionalities implemented into the in-house compressible Computational Fluid Dynamics (CFD) solver, G3D::Flow, are described. These new additions have been put in place with the objective of performing turbomachinery simulations using hybrid RANS/LES methods as well as nozzle flow simulations. Some of the additions to G3D::Flow include: phase-lagged pitch-wise and rotor-stator interfaces based on the chorochronic method as well as a method based on Proper Orthogonal Decomposition (POD), sliding grid interface and synthetic turbulence injection. The added capabilities, enable G3D::Flow to perform high-fidelity turbomachinery CFD simulations, which were not affordable before due to their high computational cost, since truncated domains can be used.

A hybrid RANS/LES simulation of the VOLVO S6 nozzle contour operating under overexpanded conditions is performed. This same geometry, under the same conditions, was previously simulated and reported using a different hybrid RANS/LES methodology. A reduction of over 50% in the difference between the predicted standard deviation of the side loads and those measured in a previous experimental is observed in the current simulation.

In this work, an optimization framework called HAMON is also presented, which is based on evolutionary algorithms. In cases where the optimization is based on computationally heavy tasks, such as 3D CFD simulations, meta-modeling techniques can be used to speed up the optimization processes. HAMON can be used to fine tune an existing design, or as it has been used here, as black-box approach. It has been able to design counter rotating open rotors with more than acceptable performance where no knowledge about propeller aerodynamics was assumed, giving all the design variables more freedom than probably needed. This black-box approach might be specially useful when optimizing new technologies for which no prior knowledge exist, allowing not only to, hopefully, find good designs but also to show the trends of what a good design should be like.

Keywords: CFD, Sliding grid, Chorochronic, Phase-lagged boundary conditions, LODI, Synthetic turbulence, Stochastic optimization, Evolutionary algorithms, Nozzle flow separation, Hybrid RANS/LES, POD based phase-lag

To my family...

Acknowledgements

First of all, I would like to thank Niklas Andersson, not only for giving me the opportunity to embark on this adventure, but also for sharing so much of his knowledge with me. Where would I be without all your G3D::Flow/CFD knowledge?

I would like to express my gratitude to all my colleagues at the Division of Fluid Dynamics for all the fun times and the amazing working environment that have made me feel like home despite being several thousands kilometers away. Special mention goes to the G3D::Flow mafia; to Daniel for all your help, advice, fun that we had working, kebab evenings and also those infinite discussions on whose coding style was better, your perfectly order and fully commented one, or my ordered chaos with no white spaces (I still have not made up my mind). To Elias, who was the first person I met and shared office with when I came into the division for my Master's thesis, who made me feel welcome from the first moment (despite reminding me more often than probably needed that in Iceland killing Basque people was not totally illegal). Also for all your help (not at all with g3dmesh), advice, and fun that I have had working with you. Also Magnus, with whom I have had endless discussion about all kind of topics not only CFD related. I think Ananda and Sudharsan also deserve a special mention as both had to suffer me during the teaching of the CFD course. I hope you had as much fun sharing that course with me as I had with you.

I would also like to thank my friends outside Chalmers (the Spanish/Latino mafia) who have made it quite easy for me to occupy my mind with non-work related things, and have kept me distracted as well as positive when things were not going very well. Thanks for all the fun times we have had!

Last but not least, to my family. To my parents, Susana and Rafael, for always supporting me in my decisions, including the one that meant moving away from Spain. To my sister, Berta, who is one of the most determined persons I know, for always giving me words of encouragement and believing in me. And finally, to Naia, my niece, you have been one of the best things that ever happened to our family!

This work has been carried out within the project TurboNoiseBB, which is funded by European Union's Horizon 2020 research and innovation program under grant agreement No. 682 690714, and by the Swedish National Space Agency, through the NRFP program supported by GKN Aerospace Sweden AB. The computational resources used have been provided by the Swedish National Infrastructure for Computing (SNIC) and the Chalmers Centre for Computational Science and Engineering (C^3SE) in Sweden.

Gonzalo Montero Villar
Göteborg, November 2021

NOMENCLATURE

Greek letters

ρ	density
τ_{ij}	viscous stress tensor
σ_{ij}	turbulent stress tensor
ρe_0	total energy density
μ	dynamic viscosity
μ_t	turbulent or eddy viscosity
ν	kinematic viscosity
ϵ	dissipation rate
λ_i	characteristic speed of the i^{th} wave, eigenvalue of R_{ij}
σ	coupling parameter in LODI outlet, length scale in SEM, wave number direction, Spalart-Allmaras model constant
Ω_i	rotational speed of the i^{th} row
β_i	phase shift of the i^{th} row
Δt	simulation time step
ϵ^k	intensities of the k^{th} eddy
$\phi^n,\ \psi^n,\ \theta^n,\ \alpha^n$	random angles used in Fourier based synthetic turbulence
κ	wave number, von Kármán constant
Δx	grid size
δ_{ij}	Kronecker delta
γ	heat capacity ratio
η	Kolmogorov length scale
τ	Kolmogorov time scale
ϕ	generic flow variable
$\tilde{\nu}$	Spalart-Allmaras viscosity
Δ	LES filter width
ΔV	cell volume
$\mathscr{L}_i$	i^{th} characteristic wave amplitude
$\mathcal{F}$	convective flux operator
$\omega_n,\ \omega_k$	spinning mode frequency
$\mathbf{\Sigma}$	singular values

Roman letters

u_i	i^{th} component of the velocity vector
x_i	i^{th} component of the position vector
p	pressure
k	thermal conductivity, turbulent kinetic energy
T	temperature, period

C_p	specific heat
P_r	Prandtl number
P_{r_t}	turbulent Prandtl number
c	speed of sound
t	time
$c_{b1},\ c_{b2},\ c_{w1}$	Spalart-Allmaras model constant
$f_w,\ f_{t2},\ f_{v1}$	Spalart-Allmaras model functions
$\tilde{d}$	(D)DES modified length scale
d	wall distance
C_s	Smagorinsky model constant
C_{DES}	(D)DES model constant
r_d	DDES model function
f_d	shielding function
q_j^t	turbulent heat flux
p_∞	target pressure in LODI based outlet
NO	number of overlapping cells in GGI/sliding grid
$A_{in}^{intersec}$	intersection area in GGI/sliding grid
A_i	face area in GGI/sliding grid
$q(i)$	flow state of cell i
f	relative blade passing frequency
N_i	number of blades of i^{th} row
t_{shift_i}	time shift of i^{th} row
N_h	number of harmonics to consider on truncated Fourier series
$\hat{q}_n$	n^{th} temporal Fourier coefficient
t_s	simulation time
j	imaginary unit
$q(t)$	flow variables at time t
$Q(t)$	flow variable reconstructed using Fourier coefficients at time t
N_{tspp}	number of time steps per period
$\hat{q}_{n,k}$	time azimuthal Fourier coefficient
$\mathbf{U}$	eddy convective velocity in SEM
R_{ij}	Reynolds stress tensor
f_i	eddy shape function in SEM
$P,\ Q,\ PQ$	populations in NSGA-II
$\mathbf{X_{i,n}}$	i^{th} parameter vector of the n^{th} population in DE
$F,\ C$	user defined parameter in DE
N_k	number of eddies
V_{BOX}	volume of the box of eddies
$\mathbf{x}^k$	position of the k^{th} eddy
$\mathbf{u'}$	synthetic velocity fluctuation
a_{ij}	Cholesky decomposition of R_{ij}
$\hat{u}^n$	amplitude of the n^{th} Fourier coefficient on synthetic turbulence
$E(\kappa)$	energy contained at κ
$a,\ b$	time correlation filter variables
T_{int}	integral time scale
N	number of objective functions

o_m	value of the m^{th} objective function
cd	crowded distance
$\mathbf{W}$	flow state in SVD interface
$\mathbf{W^r}$	flow state reconstructed in SVD interface
$\mathbf{S}$	flow state history matrix in SVD interface
$\mathbf{U}$	left singular vectors
$\mathbf{V}$	right singular vectors
n_c	number of cells in SVD interface
n_s	number of time samples to store in SVD interface
n_θ	number of singular values to keep in SVD interface

Abbreviations

CFD	Computational Fluid Dynamics
(U)RANS	(Unsteady) Reynolds Averaged Navier-Stokes
LES	Large Eddy Simulation
DNS	Direct Numerical Simulation
(D)DES	(Delayed) Detached Eddy Simulation
SA	Spalart-Allmaras
SGS	Sub-Grid-Scale
NSCBC	Navier-Stokes Characteristic Boundary Condition
LODI	Local One Dimensional Inviscid
GGI	General Grid Interface
POD	Proper Orthogonal Decomposition
SEM	Synthetic Eddy Method
EA	Evolutionary Algorithm
GA	Genetic Algorithm
DE	Differential Evolution
RBF	Radial Basis Function
CROR	Counter Rotating Open Rotor
LHS	Latin Hypercube Sampling
NSGA-II	Non-dominated Sorting Genetic Algorithm II

Thesis

This thesis consists of an extended summary and the following appended papers:

Paper A D. Lindblad, G. Montero Villar, N. Andersson, A. Capitao Patrao, S. Courty-Audren, and G. Napias. "Aeroacoustic Analysis of a Counter Rotating Open Rotor Based on the Harmonic Balance Method". *2018 AIAA Aerospace Sciences Meeting.* 2018

Paper B G. Montero Villar, D. Lindblad, and N. Andersson. "Multi-Objective Optimization of an Counter Rotating Open Rotor using Evolutionary Algorithms". *2018 Multidisciplinary Analysis and Optimization Conference.* 2018

Paper C G. Montero Villar, D. Lindblad, and N. Andersson. "Effect of Airfoil Parametrization on the Optimization of Counter Rotating Open Rotors". *AIAA Scitech 2019 Forum.* 2019

Paper D G. Montero Villar, D. Lindblad, and N. Andersson. "Investigation of Phase-Lagged Boundary Conditions for Turbulence Resolving Turbomachinery Simulations". *AIAA AVIATION 2020 FORUM.* 2020

Paper E G. Montero Villar and N. Andersson. "Initial test of SVD based Phase-Lagged Boundary Conditionsfor Turbomachinery Simulations in the G3D::Flow Solver". *Technical report, Chalmers University of Technology.* 2021

Paper F G. Montero Villar, N. Andersson, and Östlund J. "Nozzle Side Loads Prediction using a Hybrid RANS/LES Method". *Technical report, Chalmers University of Technology.* 2021

Other publications:

Paper G D. Lindblad, N. A. Wukie, G. Montero Villar, and N.s Andersson. "Implementation of a Quasi-Three-Dimensional Nonreflecting Blade Row Interface for Steady and Unsteady Analysis of Axial Turbomachines". *2018 AIAA/CEAS Aeroacoustics Conference.* 2018

Paper H D. Lindblad, N. Wukie, G. Montero Villar, and N. Andersson. "A Nonreflecting Formulation for Turbomachinery Boundaries and Blade Row Interfaces". *AIAA Scitech 2019 Forum.* 2019

Paper I A. Capitao Patrao, T. Grönstedt, A. Lundbladh, and G. Montero Villar. Wake Analysis of an Aerodynamically Optimized Boxprop High-Speed Propeller. *Journal of Turbomachinery* (2019)

Paper J C. Kissner, S. Guérin, P. Seeler, M. Billson, P. Chaitanya, P. Carrasco Laraña, H. de Laborderie, Benjamin F., et al. "ACAT1 Benchmark of RANS-Informed Analytical Methods for Fan Broadband Noise Prediction—Part I—Influence of the RANS Simulation". *Acoustics*. Vol. 2. 3. Multidisciplinary Digital Publishing Institute. 2020, pp. 539–578

Paper K S. Vasudevan, S. Etemad, L. Davidson, and G. Montero Villar. Numerical model to estimate subcooled flow boiling heat flux and to indicate vapor bubble interaction. *International Journal of Heat and Mass Transfer* **170** (2021), 121038

Contents

Part I
Extended Summary

Chapter 1
Introduction

Predicting the future with absolute certainty is more often than not, impossible. Nevertheless, when it comes to the evolution of air traffic, one thing seems to be clear regardless of which study is analyzed, it is growing and will continue to grow in the foreseeable future. For instance, according to Airbus' global market forecast, the passenger fleet is expected to double by year 2035 in comparison to 2015, and the passenger traffic is expected to grow at a rate of 4.5% CAGR over the same period of time [1]. This increase in air traffic is expected to have considerable environmental and economic impact.

With these predictions in mind, the European Commission has set some goals for the technologies that should be available by the year 2050. Some of these goals focus on the environmental impact of aviation, such as having air vehicles which design and manufacturing allows them to be recycled, a 75% reduction of CO_2 emissions per passenger kilometer, a 65% reduction in perceived noise emissions and a 90% reduction in NO_x emissions with respect to the year 2000 [18]. On the other hand, taking a look at the economic side of it, the International Air Transport Association (IATA) estimated in their annual report, that 24% of all airlines operating costs in the year 2018 were due to jet fuel [27].

Taking these goals and predictions into consideration, it comes as no surprise that the aviation industry is putting some major efforts into improving their products. These efforts have yielded significant improvements over the past decades, much of it thanks to the advances in numerical simulation techniques, such as more reliable and accurate Computational Fluid Dynamics (CFD), as well as a huge increase in computational power. In order to be able to keep up with the regulations, further improvements are needed, which might be achieved in two different ways, by continuously improving designs of existing technologies (which is becoming harder and harder) or by developing new technologies that will enable radical new designs.

Regardless of which of the two ways improvements come, the use and development of computational tools seems to be a key aspect. For instance, more accurate CFD tools which better predict the flow over a certain turbomachinery component, can be helpful not only in the design and improvement process of that particular component, but also in the understanding and learning about the complex fluid physics phenomena that take place, which might lead to the development of a new concept. Moreover, design and optimization tools that do not rely on analytical/empirical methods come in handy, specially when new concepts are being developed. One example of the application of such methods where a new concept could not be designed using analytical methods, as they were not existing, can be seen with the Boxprop [4]. In this case, a fully automated optimization framework that assumed almost no existing knowledge about aerodynamics was able to produce satisfactory designs [9].

1.1 Background

When it comes to CFD simulations, steady state RANS or URANS have been the most common approach adopted by industry over the past several decades. These have proven to be very successful for a wide range of applications but not enough in several others, i.e. when large flow separation occurs or the transient turbulent content of the flow field is of great importance. In the past recent years, due to the increase in computational power, scale resolving simulations, such as hybrid RANS/LES or even LES, are becoming computationally more affordable and are starting to be used in those cases where RANS/URANS is not enough or where greater accuracy is sought for.

In turbomachinery simulations where blade rows are rotating relative to one another, and steady state computations are performed, the most common practice is to use a so called mixing-plane interface and standard periodicity together with a RANS turbulence model. For unsteady computations the set up complexity increases, replacing the mixing-plane with a sliding-grid interface when the circumferential sector covered in both domains is the same. Another layer of complexity is added when the same circumferential sector on both blade rows can not be achieved. This can happen, when for instance, one of the blade rows has a prime number of blades, which are often used to avoid certain phenomena such as fatigue or resonance, therefore requiring of a full annulus simulation [58]. In order to avoid the computationally expensive and often prohibitive full 360° simulations, Erdos et al. [16] introduced a phase-lag approach that allowed single blade passage simulations in 1977. This approach is based on full storage of the flow field at the boundaries, and its computational requirements grow very rapidly with increase in mesh cell count and time step reduction.

In contrast to the full storage of the flow signal, some other methods have been proposed by different authors. In 1988, Giles proposed the so called "time-inclining" method, where a time coordinate transformation is carried out in azimuhtal direction to ensure that, at the pitch-wise boundaries, standard periodic boundary conditions can be applied [22]. A different approach of that of Giles is the one proposed by Gerolymos et al. [21] in 2002, where the concept of flow storage proposed by Erdos et al. [16] is revisited, but this time, Fourier series are used as a data compression mechanism. More recently, in 2016, Mouret et al. used Proper Orthogonal Decomposition (POD) to compress and store the data [40]. The latter aims at improving the performance of the Fourier-based approaches as no assumption regarding the frequency content of the flow signal is done when compressing the data, which for instance will be of a broadband nature when dealing with scale resolving computations.

Another area in which advances in CFD simulations techniques can be applied is that of rocket nozzles. Being able to both predict and understand the loads that the nozzle will be withstanding during its operation is of great importance. A proper load prediction will aid in the optimization of the design, where for instance, the weight will be tried to be reduced to a minimum since it has a great impact on the amount of energy required to send payload to space, and thus on the cost. Not only the thrust generated by the nozzle needs to be taken into account, but also other type of loads such as side loads, which can be caused by flow instabilities, and result in failure if not accounted for properly.

When nozzles are not operated under ideal conditions, the risk of having the flow

detached from the wall exist. This can happen, for instance, during the start up sequence, where the ambient pressure at sea level is high enough so that the flow can not withstand the pressure gradient and separates from the nozzle wall. Depending on the type of nozzle, different kind of separations exist, one of them being the so called free-shock separation. When this kind of phenomena occurs, the flow is separated form the wall forming an oblique shock and never attaches again. If this flow separation happens in a non-axisymmetric manner, there will be an imbalance on the wall pressure along the circumference of the nozzle, resulting in side loads. In order to predict such phenomena higher fidelity CFD simulations than those provided by RANS simulations are sought for, such as hybrid RANS/LES. More information on phenomena responsible for nozzle side loads can be found in [43].

1.2 Aim

The objective of this work has been to further develop the in-house compressible CFD solver, G3D::Flow[1], with the ultimate aim of improving and expanding its capabilities in relation to turbomachinery simulations as well as nozzle flow simulations. The added functionality to G3D::Flow include; phase-lagged periodicity and rotor-stator interfaces based on the chorochronic method as well as POD based method, and synthetic turbulence methods among others. These new additions enable to use G3D::Flow to perform scale resolving turbomachinery simulations which would not have been possible without the implemented support for truncated domains.

An optimization platform, HAMON[2], has also been developed. It is based on evolutionary algorithms that carry out the optimization process, with the capability of using meta-modelling for speeding up computationally demanding problems. This tool has been used to optimize several turbomachinery components within the Division of Fluid Dynamics at Chalmers University of Technology.

[1] Webpage: `https://nikander.github.io/g3dflow`
[2] Available at `https://github.com/gmonterovillar/HAMON`

Chapter 2
Methodology

In order to perform fluid flow simulations, the in-house solver G3D::Flow is used, which is based on the family of codes developed by Eriksson [17]. G3D::Flow is a finite volume compressible structured CFD solver, developed and maintained at the division of Fluid Dynamics at Chalmers University of Technology. It uses a three-stage second-order accurate Runge-Kutta algorithm for advancing the solution in time when the explicit solver is to be used, or a second order backward Euler scheme with dual time stepping. Regarding the spatial discretization, an upwind-biased third-order accurate low dissipation scheme is used for the convective fluxes and a second-order accurate central differencing scheme for the diffusive ones. More information on G3D::Flow can be found at [20, 3, 49].

2.1 Governing equations

The equations that dictate the behaviour of fluid flows are usually referred to as Navier-Stokes equations. Here they are presented in their viscous compressible form, for a Newtonian fluid, using tensor notation, in Cartesian coordinates and neglecting body forces. These equations are individually referred to as continuity (2.1), momentum (2.2) and energy (2.3) equations.

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho u_i)}{\partial x_i} = 0 \tag{2.1}$$

$$\frac{\partial(\rho u_i)}{\partial t} + \frac{\partial(\rho u_i u_j)}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} \tag{2.2}$$

$$\frac{\partial(\rho e_0)}{\partial t} + \frac{\partial[(\rho e_0 + p)u_j]}{\partial x_j} = \frac{\partial(u_i \tau_{ij})}{\partial x_j} + \frac{\partial}{\partial x_j}\Big(k\frac{\partial T}{\partial x_j}\Big) \tag{2.3}$$

where τ_{ij} is the viscous stress tensor, ρe_0 is the total energy density and k is the thermal conductivity, which are defined as

$$\tau_{ij} = \mu\Big(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\Big) \tag{2.4}$$

$$\rho e_0 = \frac{p}{\gamma - 1} + \frac{1}{2}\rho u_k u_k \tag{2.5}$$

$$k = \frac{\mu C_p}{P_r} \tag{2.6}$$

where μ is the dynamic viscosity, C_p is the specific heat, P_r is the Prandtl number and γ the heat capacity ratio.

Direct Numerical Simulation (DNS) allows to directly numerically solve Eqs. 2.1 - 2.3 without any approximation, but comes at the expense of having to resolve all spatial and

temporal scales contained in the flow field. This means that enough grid resolution is needed in order to be able to capture the Kolmogorov length scales, $\eta = (\nu^3/\epsilon)^{1/4}$, as well as a sufficiently small time step for the Kolmogorov time scales to be resolved, $\tau = \sqrt{\nu/\epsilon}$. Where ν and ϵ are the kinematic viscosity and dissipation rate respectively. These grid and time step requirements make DNS non-feasible for most practical applications with the computing power available today, and also the one that will be available in the foreseeable future, therefore some modelling is required.

2.2 Turbulence modelling

Turbulence modelling is a field of CFD that emerged due to the need of alleviating the spatio-temporal constrains imposed by DNS simulations in order to make fluid flow simulations of engineering and industrial interest computationally affordable. Different levels of modelling exist, coming at different computational costs. The most common approach over the past decades, and the most affordable one, is the Reynolds Averaged Navier-Stokes (RANS), and its unsteady version Unsteady Reynolds Averaged Navier-Stokes (URANS). In the former all turbulence is modelled, whereas in the latter, due to transient flow being allowed, the very large scales can be resolved. An intermediate step in between RANS/URANS and DNS would be Large Eddy Simulation (LES), where the largest structures of the flow are resolved and the smallest ones modelled, and finally the aforementioned DNS. LES is still quite computationally expensive nowadays, which has led to the emergence of the field of hybrid RANS/LES modelling.

2.2.1 Unsteady Reynolds Averaged Navier-Stokes equations

Decomposing a generic flow variable ϕ into its time averaged plus fluctuating parts can be expressed as,

$$\phi = \overline{\phi} + \phi', \quad \overline{\phi} = \frac{1}{T}\int_T \phi dt \tag{2.7}$$

where $\overline{\phi}$ represents the time average of ϕ, and ϕ' is a fluctuation. This time averaging technique can be applied on Eqs. 2.1-2.3 to obtain the RANS equations, but when dealing with the compressible form of the equations, some extra terms that would require modelling appear in comparison to the incompressible counterpart. Therefore, in order to avoid that extra modelling, Favre averaging is used,

$$\widetilde{\phi} = \frac{\overline{\rho\phi}}{\overline{\rho}}, \quad \phi = \widetilde{\phi} + \phi'' \tag{2.8}$$

Applying Favre averaging into the governing equations, Eqs. 2.1 - 2.3, yields the Unsteady Favre-averaged Navier Stokes equations,

$$\frac{\partial\overline{\rho}}{\partial t} + \frac{\partial(\overline{\rho}\widetilde{u_i})}{\partial x_i} = 0 \tag{2.9}$$

$$\frac{\partial(\overline{\rho}\widetilde{u_i})}{\partial t} + \frac{\partial(\overline{\rho}\widetilde{u_i}\widetilde{u_j})}{\partial x_j} = -\frac{\partial\overline{p}}{\partial x_i} + \frac{\partial\overline{\tau_{ij}}}{\partial x_j} + \frac{\partial\sigma_{ij}}{\partial x_j} \tag{2.10}$$

$$\frac{\partial(\overline{\rho}\widetilde{e_0})}{\partial t}+\frac{\partial[(\overline{\rho}\widetilde{e_0}+\overline{p})\widetilde{u_j}]}{\partial x_j}=\frac{\partial}{\partial x_j}\Big[\widetilde{u_i}(\overline{\tau_{ij}}+\sigma_{ij})\Big]+\frac{\partial}{\partial x_j}\Big(k\frac{\partial\widetilde{T}}{\partial x_j}+q_j^t\Big)-\frac{1}{2}\frac{\partial}{\partial x_j}\Big[\overline{\rho}(\widetilde{u_iu_iu_j}-\widetilde{u_iu_i}\widetilde{u_j})\Big] \tag{2.11}$$

with $\overline{\tau_{ij}}$ being the Favre-averaged viscous stress tensor,

$$\overline{\tau_{ij}}=\mu\Big(\frac{\partial\widetilde{u_i}}{\partial x_j}+\frac{\partial\widetilde{u_j}}{\partial x_i}-\frac{2}{3}\frac{\partial\widetilde{u_k}}{\partial x_k}\delta_{ij}\Big) \tag{2.12}$$

Due to the Favre averaging procedure, two new terms have appeared, σ_{ij} and q_j^t, namely turbulent stress tensor and turbulent heat flux. The turbulent stress tensor is defined as,

$$\sigma_{ij}=-\overline{\rho}\Big[\underbrace{(\widetilde{\widetilde{u_i}\widetilde{u_j}}-\widetilde{u_i}\widetilde{u_j})}_{I}+\underbrace{(\widetilde{u_i''\widetilde{u_j}}+\widetilde{\widetilde{u_i}u_j''})}_{II}+\underbrace{\widetilde{u_i''u_j''}}_{III}\Big] \tag{2.13}$$

where the terms in under-braces are called the Leonard stresses (I), cross stresses (II) and Reynolds stresses (III). Moreover, the turbulent heat flux is defined as,

$$q_j^t=-C_p\overline{\rho}(\widetilde{T\widetilde{u_j}}-\widetilde{T}\widetilde{u_j}) \tag{2.14}$$

Finally the last term in the energy equation, the term consisting on triple velocity products, is considered negligible and therefore dropped.

Note that the original governing equations presented in Eqs. 2.1 - 2.3 form a closed system of equations and therefore can be solved numerically without any modelling using DNS. Due to the averaging procedure followed to obtain the URANS equations presented in this section, new terms have appeared, and as a result, Eqs. 2.9 - 2.11 no longer form a closed system. This creates the need of turbulence modelling.

Many different types of turbulence models exist varying in complexity, number of transport equations solved, and modelling assumptions made. The most commonly used turbulent models are those that rely on the Boussinesq assumption, which approximate the turbulent stress tensor and the turbulent heat flux based on mean flow quantities. An eddy or turbulent viscosity, μ_t, is introduced, and the turbulent stress tensor and turbulent heat flux are approximated as,

$$\sigma_{ij}=\mu_t\Big(\frac{\partial\widetilde{u_i}}{\partial x_j}+\frac{\partial\widetilde{u_j}}{\partial x_i}-\frac{2}{3}\frac{\partial\widetilde{u_k}}{\partial x_k}\Big)-\frac{2}{3}\overline{\rho}k\delta_{ij} \tag{2.15}$$

and

$$q_j^t=C_p\frac{\mu_t}{P_{r_t}}\frac{\partial\widetilde{T}}{\partial x_j} \tag{2.16}$$

where P_{r_t} is the turbulent Prandtl number and k refers to the turbulent kinetic energy.

The Spalart-Allmaras turbulence model

In the Spalart-Allmaras (SA) one equation turbulence model [53], an additional transport equation for a new turbulent variable, $\tilde{\nu}$, is solved (note that for this particular variable,

($\tilde{\cdot}$) does not represent Favre averaging). In this model Eq. 2.15 is used to model the turbulent stress tensor, but the last term, $\frac{2}{3}\overline{\rho}k\delta_{ij}$, is dropped as the turbulent kinetic energy is not readily available. The transport equation in conservative and compressible form, as well as the relation to the turbulent viscosity can be written as [53, 2]

$$\frac{\partial\overline{\rho}\tilde{\nu}}{\partial t}+\frac{\partial(\overline{\rho}\tilde{\nu}\widetilde{u_j})}{\partial x_j}=\tilde{S}\tilde{\nu}(1-f_{t2})c_{b1}\overline{\rho}+\frac{1}{\sigma}\frac{\partial}{\partial x_j}\Big(\overline{\rho}(\nu+\tilde{\nu})\frac{\partial\tilde{\nu}}{\partial x_j}\Big) + \frac{c_{b2}}{\sigma}\overline{\rho}\Big(\frac{\partial\tilde{\nu}}{\partial x_j}\frac{\partial\tilde{\nu}}{\partial x_j}\Big)-\overline{\rho}\Big(c_{w1}f_w-\frac{c_{b1}}{\kappa^2}f_{t2}\Big)\Big(\frac{\tilde{\nu}}{d}\Big)^2-\frac{1}{\sigma}(\nu+\tilde{\nu})\frac{\partial\overline{\rho}}{\partial x_j}\frac{\partial\tilde{\nu}}{\partial x_j} \quad (2.17)$$

$$\mu_t=\overline{\rho}\tilde{\nu}f_{v1} \quad (2.18)$$

where σ, c_{b1}, c_{b2} and c_{w1} are constants of the model, f_w is a function designed to take care of the damping in the vicinity of walls, f_{t2} is a trip function (note that the f_{t1} trip term is not included in the current implementation), $\tilde{S}$ is a modified vorticity magnitude, κ is von Kármán's constant and finally d is the distance to the nearest wall. In order to see the value of the constants and the definition of the extra functions, the interested reader is referred to the original references [53, 2] or to the reference of the current implementation in G3D::Flow [49].

2.2.2 Large Eddy Simulation

When performing LES simulation a spatial filter is used instead. For an generic variable, on one dimension, the filtering is applied as,

$$\overline{\phi}(x,t)=\frac{1}{\Delta x}\int_{x-0.5\Delta x}^{x+0.5\Delta x}\phi(\xi,t)d\xi dt \quad \phi=\overline{\phi}+\phi' \quad (2.19)$$

where Δx represents the size of the control volume. If this spatial filtering is applied to the governing equations (Eqs. 2.1 - 2.3), one would arrive to the same expressions as previously presented for URANS (Eqs. 2.9 - 2.11), with the only difference being in the meaning of ($\bar{\cdot}$), which in the URANS equations means time averaged quantity and in the LES equations it means spatially filtered variable. Once again, after the filtering procedure, the resulting system of equations is a non-closed one, thus requiring of some level of modelling. When using LES, the larger scales appearing on the turbulent spectrum are resolved, and only those which are too small to be captured by the grid are modelled. That is the reason why often, when talking about turbulence models in LES, they are referred to as Sub-Grid-Scale (SGS) models, and the notation in the equations is changed from μ_t to μ_{SGS}.

The Smagorinsky model

One of the simplest model to close the LES system of equations is the zero equation Smagorinsky model [50]. Using this model, the sub-grid-scale viscosity is computed as,

$$\mu_{SGS}=\overline{\rho}(C_s\Delta)^2|\widetilde{S}| \quad (2.20)$$

$$|\widetilde{S}| = \sqrt{2\widetilde{S_{ij}}\widetilde{S_{ij}}} \tag{2.21}$$

where $\widetilde{S_{ij}}$ is the strain rate tensor, C_s is a model constant and the filter width, Δ, is the cubic root of the cell volume ($\Delta = (\Delta V)^{1/3}$ with ΔV being the cell volume). Having the filter width being mesh dependant means that the finer the mesh the more content is resolved and therefore less needs to be modelled. This property of LES makes performing a conventional grid dependency study rather difficult.

2.2.3 Hybrid RANS/LES

As the name suggests, this family of turbulence modelling methods combine both RANS and LES methodologies, and have arised due to the fact, that for many practical applications, LES or DNS are computationally prohibitive. This type of simulations are gaining popularity in the past recent years as they allow to simulate complex flow with higher fidelity than URANS simulations, but at a lower computational cost than LES.

One commonly used hybrid RANS/LES method is the Detached Eddy Simulation (DES) [52]. The aim of DES is to present a unique formulation that allows the flow solver to use an underlying RANS model on the attached boundary layer (where they have been proven to be quite successful) and LES on the rest of the domain. This means that areas where great separation occurs (such as in bluff-body simulations) or the wakes generated by turbomachinery blades, will be resolved using LES capabilities. Using RANS in the near wall region allows for less stringent requirements when it comes to grid resolution and time step as compared to LES, thus making them more affordable. In order to solve some issues related to grid sensitivity found in the DES method, such as grid induced separation [54], a further improved method, Delayed Detached Eddy Simulation (DDES), was introduced in [54].

Delayed Detached Eddy Simulation

Several turbulence models can be used as the underlying RANS in DES and DDES formulations. In this work, the one that is based on the one equation Spalart-Allmaras turbulence model described in the previous section is used. In order to transform the original SA model into its DES or DDES version, the only change required is to replace the wall distance, d, appearing in the destruction term of Eq. 2.17, and also in the expression of $\tilde{S}$ (not shown here), with a new length scale $\tilde{d}$.

In the original DES formulation $\tilde{d}$ was defined as [52],

$$\tilde{d} = min(d, C_{DES}\Delta) \tag{2.22}$$

where C_{DES} is a turbulence model constant. This expression makes $\tilde{d}$ uniquely grid size and wall distance dependent, without taking into account the flow. This caused different problems, such as grid induced separation or allowing the solver to go into LES mode inside the boundary layer, where the grid did not have enough resolution. In order to alleviate those problems, a shielding function, f_d, is introduced in the DDES model [54] as,

$$f_d = 1 - \tanh[(8r_d)^3] \tag{2.23}$$

this shielding function should take the value of zero inside the boundary layer where RANS is to be used and the value of one where LES is to be used. As previously mentioned the original DES had some issues due to the choice of RANS or LES being solely grid and wall distant dependent. Thus, the function r_d in Eq. 2.23 makes sure that the flow field is also accounted for, being defined as,

$$r_d = \frac{\nu + \nu_t}{\sqrt{U_{i,j}U_{i,j}}\kappa^2 d^2} \tag{2.24}$$

where $U_{i,j} = \partial u_i / \partial x_j$. Finally the length scale $\tilde{d}$ in the DDES model is defined as,

$$\tilde{d} = d - f_d \max(0, d - C_{DES}\Delta) \tag{2.25}$$

as it can clearly be seen from Eq. 2.25, when $f_d = 0$, $\tilde{d} = d$ which is the RANS formulation of the SA model and when $f_d = 1$, $\tilde{d} = C_{DES}\Delta$ which is the desired output of the original DES method.

Note that in the current work, the constant in Eq. 2.23 is changed from the original value of 8, to 16, as proposed in [45]. Finally, Δ in Eq. 2.25, is not taken as $\Delta = (\Delta V)^{1/3}$ or $\Delta = max(\Delta x, \Delta y, \Delta z)$, which are quite popular options, but instead, the length scale proposed in [48] is used which accounts for the direction of the vorticity vector.

2.3 Boundary treatment

When performing a CFD simulation, the domain is usually trimmed to a region of space which is of special interest. This results in having to apply boundary conditions in the artificially created boundaries of the domain, which will differ depending on the nature of the flow and the conditions under which it is simulated. In order to save computational power, the boundaries should be placed as close as possible to the area of interest without compromising the results. Correct implementation and use of the boundary conditions is fundamental in order not to contaminate the numerical solution obtained inside of the domain, for instance, by generating nonphysical upstream traveling pressure waves on an outlet that may destroy the solution.

2.3.1 Navier-Stokes Characteristic Boundary Conditions

In this section, the method of imposing boundary conditions referred to as Navier-Stokes Characteristic Boundary Conditions (NSCBC) by Poinsot and Lele[44] is outlined. As the names suggests, the boundary conditions are imposed using the method of characteristics. The different waves travelling across the boundary are assumed to be the same ones that would occur on an Euler simulation (i. e. waves related to viscous phenomenon are discarded). The description is made assuming boundaries normal to the x_1 direction, being the application to x_2 and x_3 direct.

Starting from the governing equations previously presented (Eqs. 2.1, 2.2 and 2.3), one can write an equivalent set of equations in which the terms involving the waves

propagating though the boundary can be identified (note that Eq. 2.2, the momentum equation, has been expanded into its three components)

$$\frac{\partial \rho}{\partial t} + d_1 + \frac{\partial(\rho u_2)}{\partial x_2} + \frac{\partial(\rho u_3)}{\partial x_3} = 0 \tag{2.26}$$

$$\frac{\partial \rho u_1}{\partial t} + u_1 d_1 + \rho d_3 + \frac{\partial(\rho u_1 u_2)}{\partial x_2} + \frac{\partial(\rho u_1 u_3)}{\partial x_3} = \frac{\partial \tau_{1j}}{\partial x_j} \tag{2.27}$$

$$\frac{\partial \rho u_2}{\partial t} + u_2 d_1 + \rho d_4 + \frac{\partial(\rho u_2 u_2)}{\partial x_2} + \frac{\partial(\rho u_2 u_3)}{\partial x_3} = -\frac{\partial p}{\partial x_2} + \frac{\partial \tau_{2j}}{\partial x_j} \tag{2.28}$$

$$\frac{\partial \rho u_3}{\partial t} + u_3 d_1 + \rho d_5 + \frac{\partial(\rho u_3 u_2)}{\partial x_2} + \frac{\partial(\rho u_3 u_3)}{\partial x_3} = -\frac{\partial p}{\partial x_3} + \frac{\partial \tau_{3j}}{\partial x_j} \tag{2.29}$$

$$\begin{aligned} &\frac{\partial(\rho e_0)}{\partial t} + \frac{1}{2} u_k u_k d_1 + \frac{d_2}{\gamma - 1} + \rho u_1 d_3 + \rho u_2 d_4 + \rho u_3 d_5 + \\ &\frac{\partial[(\rho e_0 + p) u_2)]}{\partial x_2} + \frac{\partial[(\rho e_0 + p) u_3)]}{\partial x_3} = \frac{\partial(u_i \tau_{ij})}{\partial x_j} + \frac{\partial}{\partial x_j}\Big(k \frac{\partial T}{\partial x_j}\Big) \end{aligned} \tag{2.30}$$

Contributions from variations in direction normal to the boundary have been replaced by terms that depend on the vector d_i, which can be expressed using characteristic analysis [57] as

$$\begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{bmatrix} = \begin{bmatrix} \frac{1}{c^2}\Big[\mathscr{L}_2 + \frac{1}{2}(\mathscr{L}_1 + \mathscr{L}_5)\Big] \\ \frac{1}{2}(\mathscr{L}_1 + \mathscr{L}_5) \\ \frac{1}{2\rho c}(\mathscr{L}_5 - \mathscr{L}_1) \\ \mathscr{L}_3 \\ \mathscr{L}_4 \end{bmatrix} = \begin{bmatrix} \frac{\partial(\rho u_1)}{\partial x_1} \\ \frac{\partial(c^2 \rho u_1)}{\partial x_1} + (1-\gamma)\mu \frac{\partial p}{\partial x_1} \\ u_1 \frac{\partial u_1}{\partial x_1} + \frac{1}{\rho}\frac{\partial p}{\partial x_1} \\ u_1 \frac{\partial u_2}{\partial x_1} \\ u_1 \frac{\partial u_3}{\partial x_1} \end{bmatrix} \tag{2.31}$$

where c is the speed of sound and $\mathscr{L}_i$ are the characteristic wave amplitudes defined by

$$\begin{bmatrix} \mathscr{L}_1 \\ \mathscr{L}_2 \\ \mathscr{L}_3 \\ \mathscr{L}_4 \\ \mathscr{L}_5 \end{bmatrix} = \begin{bmatrix} \lambda_1\Big(\frac{\partial p}{\partial x_1} - \rho c \frac{\partial u_1}{\partial x_1}\Big) \\ \lambda_2\Big(c^2 \frac{\partial \rho}{\partial x_1} - \frac{\partial p}{\partial x_1}\Big) \\ \lambda_3 \frac{\partial u_2}{\partial x_1} \\ \lambda_4 \frac{\partial u_3}{\partial x_1} \\ \lambda_5\Big(\frac{\partial p}{\partial x_1} + \rho c \frac{\partial u_1}{\partial x_1}\Big) \end{bmatrix} \tag{2.32}$$

where λ_i are the characteristic speeds of the waves with amplitude $\mathscr{L}_i$. These five waves can be interpreted as two acoustic waves (one travelling upstream with λ_1 and one downstream with λ_5) and two vorticity and an entropy wave (these three travelling at $\lambda_2 = \lambda_3 = \lambda_4$).

$$\lambda_1 = u_1 - c; \qquad \lambda_2 = \lambda_3 = \lambda_4 = u_1; \qquad \lambda_5 = u_1 + c \tag{2.33}$$

Based on these characteristic speeds one can determine whether a wave is leaving or entering the domain. For instance, at a subsonic inlet were the velocity is in the positive x_1 direction, there are four incoming waves and one outgoing with speed λ_1. This information can be used to formulate boundary conditions, and the solution can be advanced in time using Eqs. 2.26-2.30 if the value of the $\mathscr{L}$'s can be determined. As suggested in [44] and [62] the characteristic amplitudes corresponding to outgoing waves are determined from one-sided derivatives using interior points.

2.3.2 Local One Dimensional Inviscid relations

Due to the lack of a method to compute the exact values of the incoming $\mathscr{L}$'s in the general case, the problem is reduced to considering Local One Dimensional Inviscid (LODI) relations. As the name implies, these relations can be obtained by dumping every term related to the x_2 and x_3 dimensions, as well as viscosity from Eqs. 2.26-2.30, as well as substituting the d's according to Eq. 2.31. This results in the following LODI relations, here expressed as a function of the primitive variables

$$\frac{\partial \rho}{\partial t} + \frac{1}{c^2}\left[\mathscr{L}_2 + \frac{1}{2}(\mathscr{L}_1 + \mathscr{L}_5)\right] = 0 \tag{2.34}$$

$$\frac{\partial u_1}{\partial t} + \frac{1}{2\rho c}(\mathscr{L}_5 - \mathscr{L}_1) = 0 \tag{2.35}$$

$$\frac{\partial u_2}{\partial t} + \mathscr{L}_3 = 0 \tag{2.36}$$

$$\frac{\partial u_3}{\partial t} + \mathscr{L}_4 = 0 \tag{2.37}$$

$$\frac{\partial p}{\partial t} + \frac{1}{2}(\mathscr{L}_1 + \mathscr{L}_5) = 0 \tag{2.38}$$

These relations can also be combined to express time or spatial derivatives of other quantities in order to simplify the imposition of boundary conditions.

Subsonic outlet

In the case of a subsonic outlet, there is only one wave going into the domain with an amplitude $\mathscr{L}_1$, the rest of the amplitudes can be determined from interior points using one-sided derivatives as aforementioned. Once all the $\mathscr{L}$'s have been determined, Eqs. 2.26-2.30 are used to perform the iteration. Different types of subsonic outlets can be obtained:

1. **Reflecting outlet**: one way of obtaining an exact reflection of the waves at the outlet is to keep the pressure at the outlet constant ($\partial p/\partial t$=0). This together with one of the LODI relations, Eq. 2.38, allows to calculate the amplitude of the incoming wave

   $$\mathscr{L}_1 = -\mathscr{L}_5 \tag{2.39}$$

   where $\mathscr{L}_5$ as well as $\mathscr{L}_2$, $\mathscr{L}_3$ and $\mathscr{L}_4$ has been computed from the interior.

2. **Perfectly non-reflecting outlet**: in order to avoid reflections one can set the amplitude of the incoming wave to zero, $\mathscr{L}_1 = 0$. The problem with this implementation, is that since no information is travelling inside the domain, it prevents from setting the right pressure level.

3. **Partially-reflecting outlet**: due to the problem described with the perfectly non-reflecting outlet, some small reflections inside the domain are allowed in order to be able to set the correct pressure level. This is achieved by computing the amplitude of the entering wave as [44]

$$\mathscr{L}_1 = K(p - p_\infty) \tag{2.40}$$

p_∞ being the desired pressure. The used form of K is as described in [47]

$$K = \frac{\sigma(1 - M^2)c}{L} \tag{2.41}$$

where the coupling parameter σ is set to a default value of 0.58, M is the Mach number of the flow and L a representative length of the domain.

Implementation

In order to implement the subsonic outlet boundary condition just described into G3D::Flow, the fluxes normal to the outlet boundary are not computed and instead, they are substituted with a source term for each of the conservation equations as:

$$S = \begin{bmatrix} d_1 \\ u_1 d_1 + \rho d_3 \\ u_2 d_1 + \rho d_4 \\ u_3 d_1 + \rho d_5 \\ \frac{1}{2} u_k u_k d_1 + \frac{d_2}{\gamma - 1} + \rho u_1 d_3 + \rho u_2 d_4 + \rho u_3 d_5 \end{bmatrix} \Delta V \tag{2.42}$$

where ΔV is the cell volume. The five source terms in Eq. 2.42 are introduced in the continuity, three momentum and energy equations respectively.

2.4 Turbomachinery simulations

When performing turbomachinery-like CFD simulations, it is quite common to encounter a situation where two or more blade rows need to be simulated, which might or might not be in relative motion to one another. The most widely used approach to perform these computations is to use a mixing-plane interface together with rotational periodic boundary conditions for steady state problems. If this is the case, the simulation set up is the same independently of the blade count in each of the rows and the mesh structure at the interface in between them, as a mixing-plane will perform an azimuthal average of the flow properties in order to transfer information between domains (in the case where the blade rows are in relative motion, a change in frame of reference is also performed). By using this approach, one major drawback is the fact that a lot of the information

contained in one domain is lost when transferred to the adjacent one. This is illustrated in Fig. 2.1, where a) represent the flow field right upstream of the mixing-plane, and b) what the downstream domain receives after the flow field has gone through the averaging process.

a) *b)*

Figure 2.1: *Generic wake on a RANS turbomachinery CFD simulation; a) flow upstream of the mixing-plane, b) information transferred to the downstream domain by the azimuthal average performed by the mixing-plane.*

Looking at Figure 2.1, one can clearly see that even though this approach might be useful to perform steady state computations, it is not desirable on a transient simulation, where for instance the impact of the wake impinging a second blade row is sought for, as that information is lost. When these type of simulations are to be carried out, one set up that is always valid, is to simulate the entire 360° wheel with a sliding-grid instead of the mixing-plane interface. More often than not, this becomes prohibitive due to computational resources. If the full wheel simulation is to be avoided, two different scenarios exist. In the first one, both domains cover the same pitch, this can be achieved by using one blade passage in each domain if the blade count is the same, or by using a different number of blade passages on each domain which add up to covering the same tangential sector (for instance one rotor and two stator blades in a configuration with 20 by 40 blades). In the second scenario, the blade count is different and using a combination of blade passages that cover the same tangential sector is not possible (either because there is not an existing configuration, for instance, if one of the blade counts is a prime number, or because it requires too many blades to be simulated, making computational resources a limiting factor).

In the first scenario described (same tangential sectors), a solution not to lose the information via the averaging procedure is to use a General Grid Interface (GGI) when the blade rows are not in relative motion, or a sliding-grid interface when a relative motion exist. This allows for the use of standard rotational periodicity in the pitch-wise boundaries. In the second scenario (different tangential sectors), the so called phase-lagged boundary conditions can be used to avoid the full wheel simulation. In the following subsections these two approaches are outlined.

2.4.1 General grid interface, sliding grid

When the tangential sector covered by two domains is the same, one can use a GGI or sliding grid (depending on whether they are in relative motion or not) to transfer the information between them. One can think about the GGI as an interpolator which connects cells on both sides of the interface in order to perform the flux calculation. Figure 2.2 shows an schematic on how the mesh on two sides of a GGI might look. As it can be seen, there is no need to have matching nodes, or the same cell count, and the domain can be shifted by an angle. The sliding-grid interface follows the same concept, with the difference that both the upstream and downstream domains are in relative motion to one another, therefore every time step the overlap between cells needs to be recomputed (to account for the rotations) as well as a change in frame of reference of the flow state for the flux calculation.

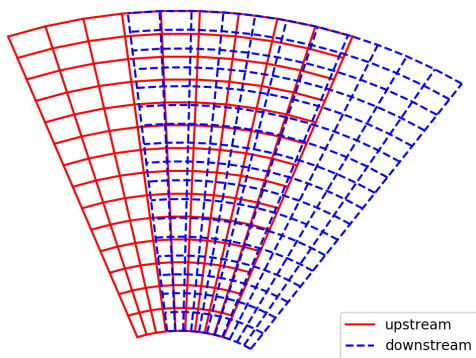
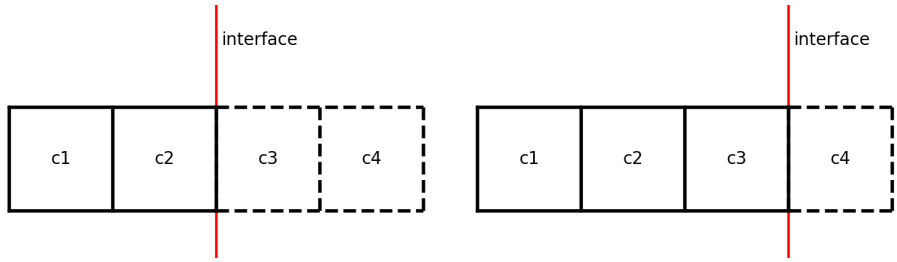


Figure 2.2: *Mesh distribution schematic on both sides of a GGI.*

Implementation

G3D::Flow uses a ghost cell approach to treat interfaces, and when using a third order accurate upwind-biased scheme for convective fluxes, two cells layers of ghost cells on each side of the interface are utilized (as depicted in 2.3 a)). The first step is to find all the cells on the other side of the interface that overlap with a given cell (both layers, i.e. $c3$ and $c4$ in Fig. 2.3 a)) and the overlapping area is calculated. This is done only once as a preprocessing step in case of a GGI being used. On the other hand, if a siding grid-interface is used, the overlap is checked for every iteration as the domains are fictionally rotated with different angular speeds. When computing the fluxes on the upstream domain (left of the interface in Fig. 2.3) two different strategies are adopted; one for the face which is located at the interface (Fig. 2.3 a)) and another one for the face on step inside the domain (Fig. 2.3 b)).

For the flux calculation corresponding to the face at the interface (Fig. 2.3 a)) several



a) *b)*

Figure 2.3: *Schematic of the cells used when computing fluxes across an interface in G3D::Flow. Cells drawn with dashed lines represent ghost cells. a) interface face and b) face one step inside the domain. Note that for simplicity here the ghost cells have been represented as fully connecting across the interface, which is not necessarily the case when using a GGI or sliding-grid interface.*

flux calculations are performed to ensure that the interface is conservative as,

$$F_{c2i} = \sum_{n=1}^{NO_i} \mathcal{F}\Big(q(c1_i), q(c2_i), q(c3_{in}), q(c4_{in})\Big) \frac{A_{in}^{intersec}}{A_i} \tag{2.43}$$

where the subscript i refers to the face in question, F_{c2i} the resultant flux onto cell $c2$ though face i, NO_i the amount of neighbouring cells overlapping, $c3_{in}$ and $c4_{in}$ are the n^{th} cell overlapping face i closer to the interface and one step in the neighbouring domain respectively (see Fig. 2.3 a)), $A_{in}^{intersec}$ is the overlapping area between face i and cell $c3_{in}$, A_i is the area of face i, $\mathcal{F}$ is the convective flux operator which takes four cells as input and $q(j)$ is the flow state of cell j.

For the flux computation one step inside the domain (see Fig. 2.3 b)) a unique flux calculation is performed by introducing an average flow state of all the overlapping cells in the ghost cell as,

$$F_{c3i} = \mathcal{F}\Big(q(c1_i), q(c2_i), q(c3_i), \sum_{n=1}^{NO_i} q(c4_{in}) \frac{A_{in}^{intersec}}{A_i}\Big) \tag{2.44}$$

where $A_{in}^{intersec}$ now refers to the overlapping area between face i and cell $c4_{in}$ instead.

If the sliding-grid is used, due to the relative motion between both sides of the interface, the flow state introduced into the ghost cells are transformed to the frame of reference of the cell where the flux will be added (cells represented by solid lines in Fig. 2.3).

2.4.2 Chorochronic method

When the tangential sector covered is not the same, one can use the so called phase-lagged boundary conditions. One such method is the chorochronic one, for which an overview is given here. For more details the interested reader is referred to [21]. Two different treatments have to be performed with this method in order to be applicable

to turbomachinery simulations, namely; the pitch-wise boundaries and the blade row interface. The notation used here mostly follows that in [41].

Pitch-wise boundaries

One can compute the relative blade passing frequency for the first blade row as

$$f_1 = \frac{N_2|\Omega_1 - \Omega_2|}{2\pi} \tag{2.45}$$

where Ω_i and N_i represent the rotational speed and the number of blades of the i^{th} blade row respectively. Indexes 1 and 2 must be swapped in Eq. 2.45 in order to compute the relative blade passing frequency for the second blade row, f_2. This is the frequency at which a given blade sees the opposite row's blades, and hence the frequency related to the periodicity of the flow properties on the former blade row (leaving aside turbulent flow instabilities which might have their own frequency). Due to the difference in pitch between the blade rows, two continuous blades see the same flow field with a phase shift, β, which can be computed as [21],

$$\beta_1 = -\frac{2\pi(N_1 - N_2)\text{sign}(\Omega_1 - \Omega_2)}{N_1} \tag{2.46}$$

Again, swapping indexes allows for the calculation of the phase shift for the second blade row. This phase shift can be converted into a time shift as

$$t_{\text{shift}_i} = \pm\frac{\beta_i}{2\pi f_i} \tag{2.47}$$

where the use of the positive or negative sign in Eq. 2.47 is determined based on the tangential direction in which the phase shift is applied.

Due to the time periodic nature of the flow, a truncated temporal Fourier series can be used to efficiently store it, where only N_h harmonics are considered. For every cell belonging to the pitch-wise boundary (two cell layers on each side as illustrated in Fig. 2.3), the n^{th} temporal Fourier coefficient is calculated as ,

$$\begin{aligned} \hat{q}_n &= f_1 \int_{t_s - \frac{1}{f_1}}^{t_s} q(t) e^{-2j\pi n t f_1} dt \\ \hat{q}_n &= f_2 \int_{t_s - \frac{1}{f_2}}^{t_s} q(t) e^{-2j\pi n t f_2} dt \end{aligned} \tag{2.48}$$

where t_s is the current simulation time, j is the imaginary unit, $q(t)$ are the flow variables at time t for a given cell, and depending which blade row the cell belongs to, the first or the second equation in Eq. 2.48 will be used. Due to the discrete nature of CFD methods,

Eq. 2.48 can be rewritten as,

$$\begin{aligned}\hat{q}_n &= f_1 \sum_{i=it-N_{tspp1}}^{it} q(i\Delta t) e^{-2j\pi n i \Delta t f_1} \Delta t \\ \hat{q}_n &= f_2 \sum_{i=it-N_{tspp2}}^{it} q(i\Delta t) e^{-2j\pi n i \Delta t f_2} \Delta t\end{aligned} \tag{2.49}$$

where N_{tsppi} is the number of time steps required to complete a period associated to the frequency computed for the i^{th} blade row with Eq. 2.45, Δt is the simulation time step and it is the current iteration of the simulation.

In order to avoid storing the time signal over one full time period before the temporal Fourier coefficients in Eq. 2.49 can be computed, a moving average technique is used to update them [21]. This technique updates the Fourier coefficients by adding the contribution of the current flow solution to all coefficients and subtracting the value one period back in time obtained using the coefficients. By using this updating technique, one can clearly see that when it has fully converged, due to the periodicity of the signal, the reconstructed value one period back in time and the current value should be the same, hence the Fourier coefficients remain unchanged. This moving average based technique can be expressed as,

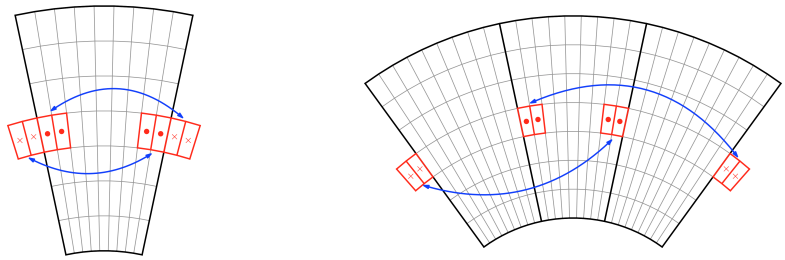
$$\begin{aligned}\Delta\hat{q}_n &= \frac{1}{N_{tspp1}} \Big(q(t_s) - Q(t_s - \frac{1}{f_1}) \Big) e^{-2j\pi n t_s f_1} \\ \Delta\hat{q}_n &= \frac{1}{N_{tspp2}} \Big(q(t_s) - Q(t_s - \frac{1}{f_2}) \Big) e^{-2j\pi n t_s f_2}\end{aligned} \tag{2.50}$$

where $\Delta\hat{q}_n$ is the update of the n^{th} temporal Fourier coefficient, and $Q(t)$ is the value obtained when reconstructing the flow properties at time t using the temporal Fourier coefficients computed with Eq. 2.49. Note that the subtraction in Eq. 2.50 is only done when the simulation time exceeds one time period associated with the relative blade passing frequency of the corresponding blade row computed with Eq. 2.45, or if a the beginning of the simulation ($t_S = 0$), $\hat{q}_0$ for the different flow variables are set equal to the initial solution values (typically obtained from a previous mixing-plane simulation).

Finally, when values are needed in order to transfer information between both sides of the pitch-wise boundaries, the temporal Fourier coefficients are used to reconstructed the desired flow properties with the corresponding time shift computed according to Eq. 2.47 as,

$$Q(t_s + t_{\text{shift}_i}) = \sum_{n=-N_h}^{N_h} \hat{q}_n e^{2j\pi n f_i (t_s + t_{\text{shift}_i})} \tag{2.51}$$

When using the periodic chorochronic boundary condition, the standard setup is the same as the one when using a standard periodic boundary condition (see Fig. 2.4 a)). In a normal periodic boundary condition this configuration generates no problems, but it might generate convergence issues when using the chorochronic one due to an underlying error feedback loop [38].



a) *b)*

Figure 2.4: *Pitch-wise boundary setup strategy. a) single sector with coupled periodic boundaries, b) multiple sectors with decoupled periodic boundaries.* • *represents the cells where the temporal Fourier coefficients are sampled and X the ghost cells where the phase shifted values are introduced. The blue arrows show which Fourier coefficients are used for phase shifting which values.*

As the values need to be phase-lagged, one has to wait until the moving average of the temporal Fourier coefficients is somehow converged in order to be able to reconstruct the phase shifted values correctly. In figure 2.4, the cells represented with a full dot are the cells where the temporal Fourier coefficients are computed, and the cells drawn with a cross, the ghost cells where the phase-shifted values are introduced. In the standard single sector set up (see Fig. 2.4 a)), before the temporal Fourier coefficients are properly converged, an error introduced by the wrong reconstruction of the phase shifted values on the left boundary (crossed cells on the left), is going to be propagated via the flux calculation to the cells where the coefficients are calculated (dotted cells on the left). This newly introduced error on the Fourier coefficients on the left side is going to be propagated to the right side boundary when the ghost cells values on the right are computed from the coefficients on the left. One can see how this is an ongoing error feedback loop that deteriorates the convergence of the moving average. In order to try to alleviate this, the setup illustrated in Fig. 2.4 b) can be used. Here more than one sector is used (in this case three, but it could also be done with two, or any number greater than one) and as can be seen from the figure, the feedback loop is broken. Now the cells where the Fourier coefficients are sampled and the ones where the phase shifted values are introduced, are not next to each other (this meaning that the error is not directly propagated via the flux calculation). In this case where the periodic boundaries have been decoupled, the algorithm used is exactly the same as described above, with the only difference that β computed with Eq. 2.46 is doubled to account for the extra sector, and therefore the time shift (see Eq. 2.47) as well. A double passage strategy was used in [5, 12] where sampled flow signal improvements was reported and [42] used three sectors in rotor-wake stator interaction computations. In the former two works whether they were simply adding sectors by still having the boundaries coupled or whether they were decoupled to break the feedback loop was not specified. In [38], decoupling the boundaries as in Fig. 2.4 b) showed improved convergence rate and stability in comparison to the standard setup.

Blade-row interface

In a turbomachinery blade-row interface, the occurring interacting spinning modes are described by the well known Tyler-Sofrin modes [59],

$$m_{n,k} = nN_1 + kN_2 \tag{2.52}$$

Each of these modes spins at the following frequencies in the frame of reference of the first and second blade row respectively,

$$\begin{aligned} \omega_n &= nN_2(\Omega_1 - \Omega_2) \\ \omega_k &= kN_1(\Omega_2 - \Omega_1) \end{aligned} \tag{2.53}$$

The flow signal at the interface is considered to have a double periodic nature, in time and space, at least for the deterministic part of it. Therefore a truncated double Fourier series is used to represent and store it, where only the first N_h harmonics are considered. A first step required in order to compute the time-azimuthal coefficients for every radial position, $\hat{q}_{n,k}(r)$, is to compute the temporal Fourier coefficients for every cell belonging to the interface in the same manner as described for the pitch-wise boundaries in Sec. 2.4.2. Once the temporal coefficients are computed, the time-azimuthal ones for every radial span can be constructed as,

$$\begin{aligned} \hat{q}_{n,k}(r) &= \frac{N_1}{2\pi} \int_0^{\frac{2\pi}{N_1}} \hat{q}_n(\theta, r) e^{j m_{n,k} \theta} d\theta \\ \hat{q}_{n,k}(r) &= \frac{N_2}{2\pi} \int_0^{\frac{2\pi}{N_2}} \hat{q}_k(\theta, r) e^{j m_{n,k} \theta} d\theta \end{aligned} \tag{2.54}$$

for each of the blade rows respectively. Finally, when the reconstructed values are needed in order to transfer information between blade rows, the flow state can be obtained using the time-azimuthal Fourier coefficients computed using Eq. 2.54 as,

$$\begin{aligned} Q(r, \theta, t) &= \sum_{n=-N_h}^{N_h} \sum_{k=-N_h}^{N_h} \hat{q}_{n,k}(r) e^{j\omega_n t - j m_{n,k} \theta} \\ Q(r, \theta, t) &= \sum_{n=-N_h}^{N_h} \sum_{k=-N_h}^{N_h} \hat{q}_{n,k}(r) e^{j\omega_k t - j m_{n,k} \theta} \end{aligned} \tag{2.55}$$

2.4.3 Singular value decomposition method

In this section a short overview of a different phase-lagged method is presented. This method uses as a truncated SVD in order to stored and compressed the flow data [40]. For greater detail, the interested reader is referred to the original work of Mouret et al., or to Paper E, where an extensive detailed explanation of the methodology used is presented. In this method the flow signal in the entire interface is sampled and stored in $\mathbf{W}$. This is a column vector of size n_c, where n_c is the number of cells in the interface. One can

construct a matrix $\mathbf{S} \in \mathbb{R}^{n_c x n_s}$, where n_s is the amount of samples to store, by appending contiguous $\mathbf{W}$ column vectors together so that,

$$\mathbf{S} = [\mathbf{W_1} ... \mathbf{W_{n_s}}] \tag{2.56}$$

If this is done, in order to apply the time shift of the pitchwise boundary computed by Eq. 2.47, one could search in $\mathbf{S}$ for the $\mathbf{W_k}$ column vector that correspond to the proper time and use that to transfer the information between sides. This is what the flow storage approach proposed by Erdos et a. [16] would look like, where the computational resources required increase very rapidly with reduction in time step and increased cell count. Therefore, in order to compress the data at the interfaces, SVD is applied,

$$\mathbf{S} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{2.57}$$

where, $\mathbf{U} \in \mathbb{R}^{n_c x n_s}$, $\mathbf{\Sigma} \in \mathbb{R}^{n_s x n_s}$ and $\mathbf{V} \in \mathbb{R}^{n_s x n_s}$ represent the left singular vectors, diagonal matrix containing the singular values and the right singular vectors respectively, where n_s is the amount of time steps required to simulate one time period, T, associated with the frequencies computed in Eq. 2.45. This would be not only equivalent to a direct storage method, since theoretically there is no loss of information when performing a singular value decomposition, but also more computationally demanding, as $\mathbf{U}$, $\mathbf{\Sigma}$ and $\mathbf{V}$ require more storing space than $\mathbf{S}$. The data compression is done by choosing not to keep all the singular values and thus discarding some rows and columns from the matrices. If one chooses to keep only the largest n_θ singular values, the new matrix dimensions become: $\mathbf{U} \in \mathbb{R}^{n_c x n_\theta}$, $\mathbf{\Sigma} \in \mathbb{R}^{n_\theta x n_\theta}$ and $\mathbf{V} \in \mathbb{R}^{n_s x n_\theta}$. This way, only the most energy containing singular values and their corresponding right and left singular vectors are kept. Once some of the content from the matrices has been removed, the desired time step can be reconstructed in order to apply the phase-lagged boundary condition as,

$$\mathbf{W}_k^r = \mathbf{U}\mathbf{\Sigma}\mathbf{V}_k^T \tag{2.58}$$

where W_k^r is the reconstructed interface flow signal at time $t_s - (n_s - k)\Delta t$. If k is chosen so that $k\Delta t = t_{shift}$ or $k\Delta t = T - t_{shift}$ (depending on which tangential direction the flow is phase shifted), one can use those values to update the ghost cells needed for the phase-lagged flux computation on the pith-wise boundaries. In the case of the blade row interface, the same procedure is followed, where now $\mathbf{W}_k^r$ contains the flow variables at the interface for the adjacent blade row. This reconstructed interface values are used to populate the ghost cells, and the flux computation is performed in the same way as it would be done for a sliding grid (see Sec. 2.4.1).

In the original work by Mouret et al[40] a method for being able to update the decomposition of $\mathbf{S}$ without having to compute $\mathbf{S}$ is shown and in Paper E the complete process is described in detail.

One difference that can be observed between the SVD method and the chorochronic one, is that the latter one compresses and transfers information based on it's frequency, whereas in the former, the aim is to retain the largest energy content possible. Despite this difference, the same error-feedback loop occurs at the pitch-wise boundaries as described in Sec. 2.4.2 when the SVD method is used, therefore the same decoupled setup can be used to obtain similar convergence and stability benefits.

2.5 Synthetic turbulence

When performing turbulence resolving simulations such as hybrid RANS/LES, LES or DNS, the inflow boundary conditions are not as straight forward as when performing (U)RANS simulations. In the latter case, it is enough with specifying the mean profiles of the flow quantities, whereas on the former case, a time dependent turbulent flow field needs to be provided. This is done to avoid having to deal with a large inlet so that the simulation can generate the turbulent flow field by the time the flow reaches the area of interest. In order to tackle this issue, several methods for introducing these fluctuations exist, some of them being; using data from a precursor simulation, Fourier based methods, synthetic eddy methods and so on. The interested reader is referred to a comprehensive review of this methods by [56].

2.5.1 Synthetic Eddy Method

Here, a short overview of the Synthetic Eddy Method (SEM) introduced in [28] is given, for the full details the interested reader is referred to the original work. SEM represents the turbulent fluctuations at the inlet as a contribution from different discrete eddies that are convected with the mean flow.

For simplicity, let the dimensions of the inlet plane be defined as $[0,0],[0,L_y],[0,L_z]$ in x, y and z direction respectively so that it is a plane orthogonal to the x axis , $\sigma(\mathbf{x})$ the prescribed turbulence length scales, $\mathbf{U}$ the average convection velocity for the eddies and R_{ij} the desired prescribed Reynolds stress tensor. Moreover let σ_i^{max} be the equal to $\max(\sigma_i(\mathbf{x}))$ over the entire inlet plane. First step is to create a box surrounding the inlet plane with dimensions:

$$[-\sigma_x^{max},\sigma_x^{max}]x[-\sigma_y^{max},L_y+\sigma_y^{max}]x[-\sigma_z^{max},L_z+\sigma_z^{max}]$$

This is the box inside which turbulent eddies are allowed to exist and to be convected. In order to ensure that the inlet plane is statistically covered by eddies, one can compute the total number of eddies, N_k, as,

$$N_k = \frac{V_{BOX}}{min(\sigma_x(\mathbf{x})\sigma_y(\mathbf{x})\sigma_z(\mathbf{x}))} \tag{2.59}$$

where V_{BOX} is the volume of the box of eddies just computed.

Once the number of eddies and the size of the box have been determined, the algorithm for computing the velocity fluctuations at every point of the inlet is the following. First, initialize random position for every eddy inside the boundaries of the box, $\mathbf{x}^k$, and assign random intensities to each of them ϵ^k, which take discrete values of -1 or 1. Then repeat the following steps for each physical time step:

1. Convect the eddies through the box using the convection velocity $\mathbf{U}$ as $\mathbf{x}^k = \mathbf{x}^k + \mathbf{U}\Delta t$, where Δt is the physical time step. If any eddy steps out of the box boundaries, it is regenerated randomly at the opposite face of the box with new random intensities ϵ.

2. Compute velocity fluctuations $\mathbf{u}'(\mathbf{x})$ according to Eq. 2.61.

In order to compute the fluctuations $\mathbf{u}'(\mathbf{x})$, first an initial fluctuation, $\mathbf{u}''(\mathbf{x})$, is computed and then scaled according to the prescribed Rij. The contribution of all the eddies needs to be accounted for when computing the initial fluctuation at an inlet cell located at $\mathbf{x}$ as,

$$u_i''(\mathbf{x}) = \frac{1}{\sqrt{N_k}} \sum_{k=1}^{N_k} \epsilon_j^k f_j(\mathbf{x} - \mathbf{x}^k) \tag{2.60}$$

where $\mathbf{x}$ is the position of the inlet point for which the fluctuations are being computed and $f_i(\mathbf{x})$ is a shape function used to represent the fluctuations induced by the turbulent eddies. Once this fluctuation is been computed it is scaled as,

$$u_i'(\mathbf{x}) = a_{ij} u_j''(\mathbf{x}) \tag{2.61}$$

where a_{ij} is the Cholesky decomposition of the prescribed Reynolds stress tensor and reads as follows,

$$a_{ij} = \begin{pmatrix} \sqrt{R_{11}} & 0 & 0 \\ R_{21}/a_{11} & \sqrt{R_{22} - a_{21}^2} & 0 \\ R_{31}/a_{11} & (R_{32} - a_{21}a_{31})/a_{22} & \sqrt{R_{33} - a_{31}^2 - a_{32}^2} \end{pmatrix} \tag{2.62}$$

In this work the used shape function reads,

$$f_i(\mathbf{x} - \mathbf{x}^k) = \sqrt{\frac{V_{BOX}}{\prod_{i=1}^{3} \sigma_i}} \prod_{i=1}^{3} f\Big(\frac{x_i - x_i^k}{\sigma_i}\Big) \tag{2.63}$$

and

$$f(x) = \begin{cases} \sqrt{1.5}(1 - |x|) & \text{if } |x| < 1 \\ 0 & \text{else} \end{cases} \tag{2.64}$$

There might be situations where synthetic fluctuations are imposed in which the simulation also contains periodic boundaries that are in contact with the inlet or RANS/LES interface, such as a turbulent channel flow. For those simulations, the SEM method does not provide a periodic velocity fluctuation field, which might results in reflections emanating from the periodic boundaries which can become of importance in, for instance, aeroacoustic simulations. In order to address this issue, whenever periodic boundaries are used, the eddies are allowed to induce a perturbation across the boundaries by creating copies of them, as illustrated in Fig. 2.5. When this is done, the size of the box of eddies is shrunken in the direction in which the periodicity applies so that the box will perfectly fit the computational domain in that direction. This treatment ensures that the fluctuating velocity field will be fully periodic.

2.5.2 Fourier based synthetic fluctuations

An overview of a method to imposed synthetic fluctuations as a superimposition of random Fourier modes is described [8, 14, 6, 7]. In order to prescribe anisotropic fluctuations, this method generates an isotropic turbulent field on the principal components of the desired Reynolds stress tensor and then transforms them to the computational reference system. Here an overview of the method is presented.

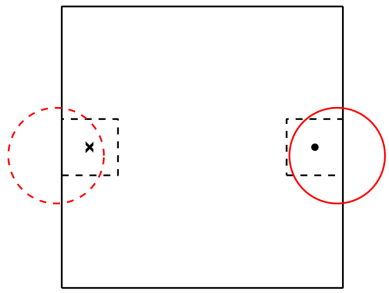


Figure 2.5: *Schematic of the treatment of periodic boundaries when SEM is utilized. The full dot and the cross represent cells that are affected by the presence of the original eddy (solid line circle) and the periodic copy of it (dashed line circle) respectively.*

Isotropic fluctuations

The turbulent fluctuations are expressed as contributions from different random Fourier modes as,

$$\mathbf{u'}_{iso}(\mathbf{x}) = 2\sum_{n=1}^{N} \hat{u}^n \cos(\boldsymbol{\kappa}^n \mathbf{x} + \psi^n)\boldsymbol{\sigma}^n \tag{2.65}$$

where N is the number of Fourier modes used, $\hat{u}^n$, $\boldsymbol{\kappa}^n$, $\boldsymbol{\sigma}^n$ and ψ^n are the amplitude, wave number vector, direction and phase of the n^{th} mode, and $\mathbf{x}$ is the coordinate vector of the grid node where fluctuations are to be introduced. To be able to compute $\boldsymbol{\kappa}^n$, $\boldsymbol{\sigma}^n$ with Eq. 2.66 three random angles are drawn, namely α^n, ϕ^n and θ^n. The random numbers here needed with their probability distributions and allowed ranges are shown in Tab. 2.1.

$$\begin{aligned} \kappa_1^n &= \sin(\theta^n)\cos(\phi^n) & \sigma_1^n &= \cos(\phi^n)\cos(\theta^n)\cos(\alpha^n) - \sin(\phi^n)\sin(\alpha^n) \\ \kappa_2^n &= \sin(\theta^n)\sin(\phi^n) & \sigma_2^n &= \sin(\phi^n)\cos(\theta^n)\cos(\alpha^n) + \cos(\phi^n)\sin(\alpha^n) \\ \kappa_3^n &= \cos(\theta^n) & \sigma_3^n &= -\sin(\theta^n)\cos(\alpha^n) \end{aligned} \tag{2.66}$$

Table 2.1: Probability distributions and allowed ranges for random variables.

variable	prob.	range
ψ^n	$1/(2\pi)$	$0 < \psi^n < 2\pi$
ϕ^n	$1/(2\pi)$	$0 < \phi^n < 2\pi$
θ^n	$1/(2\sin(\pi))$	$0 < \theta^n < \pi$
α^n	$1/(2\pi)$	$0 < \alpha^n < 2\pi$

For more datailed information on the geometrical and physical meaning of the different random variables, the reader is referred to [14, 13].

In order to compute the amplitude of the Fourier modes, $\hat{u}^n$, the range of wave numbers is first defined. The highest wave number is determined from the grid size and the smallest wave number from the turbulent length scale [13]. Once the range is determined, it is distributed among the N modes on equally large sectors $\Delta\kappa$, and the energy contained by each wave number, $E(\kappa)$, is taken from a modified von Kármán spectrum [14, 13]. The amplitude is finally computed as,

$$\hat{u}^n = \sqrt{E(\kappa)\Delta\kappa} \tag{2.67}$$

Anisotropic fluctuations

The approach presented above has been extended to be able to impose anisotropic fluctuations [51, 6, 8], and used and described in [14, 13]. As previously mentioned the main idea behind this method is to generate isotropic fluctuations on the principal directions of the prescribed Reynolds stress tensor, and then transform them to the computational reference system.

In order to achieve this, Eq. 2.65 is transformed into,

$$\mathbf{u}'_{aniso}(\mathbf{x}^*) = 2\sum_{n=1}^{N} \hat{u}^n cos(\boldsymbol{\kappa}^{*n}\mathbf{x}^* + \psi^n)\boldsymbol{\sigma}^{*n} \tag{2.68}$$

where the superscript * denotes principal coordinates and the wave number vector and direction are scaled according to

$$\boldsymbol{\kappa}^{*n} = \kappa_i^{*n} = \sqrt{\frac{1}{\lambda_i}} R_{ji}\kappa_j^n \quad ; \quad \boldsymbol{\sigma}^{*n} = \sigma_i^{*n} = \sqrt{\lambda_i} R_{ji}\sigma_j^n \tag{2.69}$$

where $\boldsymbol{\lambda}$ are the eigenvalues of the desired Reynolds stress tensor. Finally $\mathbf{u}'_{aniso}(\mathbf{x}^*)$ is transformed to the computational frame of reference, this transformation matrix can be obtained from the eigenvectors of R_{ij}. A complete description of the transformation procedures can be found in [8].

Time correlation

Both the isotropic an anisotropic synthetic generation methods described above use Fourier modes in order to provide the turbulent field with a spatial correlation. In order to ensure that the computed perturbations also have a temporal correlation, a time filter is applied as follows,

$$(\mathbf{U}')^k = a(\mathbf{U}')^{k-1} + b(\mathbf{u}')^k \tag{2.70}$$

where k denotes the simulation time step, $\mathbf{U}'$ the actual fluctuation that is introduced in the computational domain or RANS/LES interface, $\mathbf{u}'$ is taken from either the isotropic or anisotropic version of the method and a and b are defined as,

$$a = e^{-\Delta t/T_{int}} \qquad b = \sqrt{1-a^2} \tag{2.71}$$

T_{int} being the integral time scale and Δt the simulation time step.

2.6 HAMON - an optimization platform

HAMON is an optimization platform developed in Python whose optimization mechanisms are based on Evolutionary Algorithms (EAs) [23]. It can be used to deal with single- and multi- objective constrained and unconstrained optimization problems. In cases where the objective function is too expensive to evaluate (such as in optimization where the objective function values are obtained from a 3D CFD simulation), it offers the possibility to use Radial Basis Functions (RBFs) to try and speed up the optimization process.

The procedure implemented when using HAMON as an optimization tool, together with CFD simulations and RBFs is illustrated in Fig. 2.6. In this case, it is applied to the multi-objective aerodynamic optimization of a Counter Rotating Open Rotor (CROR).

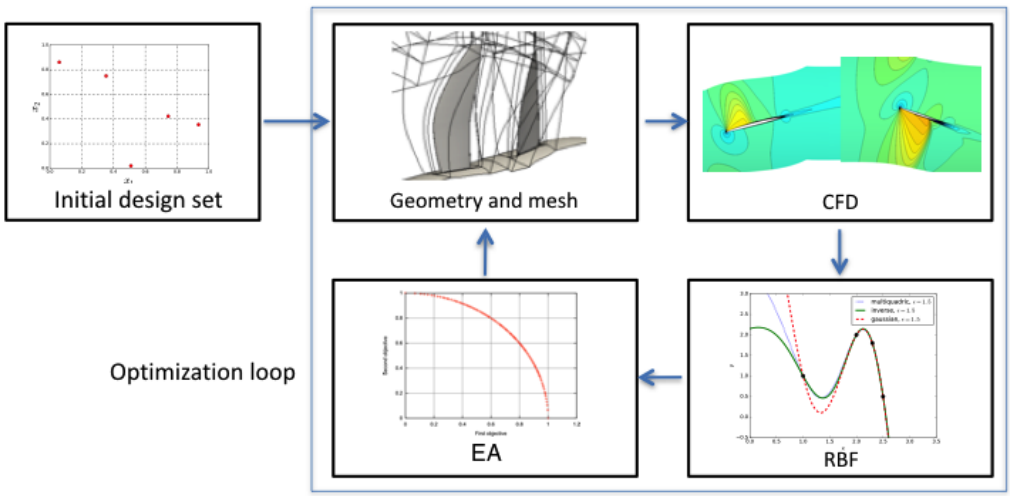


Figure 2.6: *Optimization procedure implemented in HAMON when RBFs are used.*

This entire procedure can be broken down into the following steps, once the design parameters have been defined;

1. **Initial design set:** an initial random sampling of the allowed design space is carried out, in this case, by means of Latin Hypercube Sampling (LHS).

2. **Geometry and mesh:** for a given number of designs (coming from the LHS in the first iteration loop, or from the EA in the following ones) an automated process generates the corresponding geometries and meshes.

3. **CFD:** the generated designs in the previous step are simulated using 3D CFD simulations and the objective function values post-processed in order to assess their performance.

4. **RBF:** based on all the CFD evaluations performed, a response surface is fitted to allow to estimate the performance of a new design as an algebraic expression (which

is way faster than the CFD evaluation).

5. **EA:** the generated RBF is passed to the evolutionary algorithm to optimize based on it. A set of optimal designs are found by the optimization process (this is based on RBF prediction not CFD evaluation).

6. **Iterate:** the newly found designs by the EA are put through steps two and three in order to check their "true" performance based on CFD. With the newly obtained information on the designs, the RBF is updated aiming at improving its prediction capabilities (step four). The updated RBF is passed to the EA to optimize (step five). This process is repeated until a satisfactory design is found or the RBF shows no further improvement in its prediction capabilities despite being given information about new designs.

2.6.1 Evolutionary algorithms

HAMON uses evolutionary algorithms as its optimization method, Genetic Algorithms (GAs) and Differential Evolution (DE) being implemented. Both EA methods can handle single- and multi-objective, as well as constrained and unconstrained optimization problems. These two types of evolutionary algorithms are stochastic optimization methods inspired by biological evolution which are population-based algorithms. Both methods start with a pool of candidates or individuals that conform the population (in HAMON it is randomly initialized), which is advanced through generations through selection, mutation and recombination processes [61].

Single-objective genetic algorithm

In HAMON a standard genetic algorithm with binary encoding (meaning that the design variables are encoded in an array of ones and zeros called chromosome) is implemented for single-objective problems. The processes used for advancing the population from one generation to the next are the following ones;

1. **Selection** (tournament selection of size two): two individuals are randomly chosen from the population and compared against each other. The one with better performance will move on with a user defined probability (typically between 0.7 and 1). This selection process resembles the "survival of the strongest" seeing in nature.

2. **Crossover** (one-point crossover): after two individuals have come out of tournament selection, they are given the chance (with a user defined probability) of generating an offspring by combining their chromosomes (see Fig. 2.7). If the offspring is successful the two children move on, otherwise the parents do. This process mirrors the reproduction seen in nature.

3. **Mutation**: the individuals coming out of the crossover process (either the parents or the children) get each of their chromosomes swapped with a user-defined probability.

4. **Elitism**: before putting a population through the three processes aforementioned, a copy of the best performing individual is stored to ensure that it is not lost. After

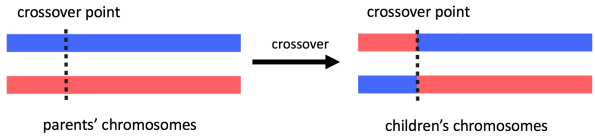


Figure 2.7: *Illustration of one-point crossover.*

the population has gone through selection, crossover and mutation, the saved copy is swapped by a random individual of the population provided the stored copy performed better.

Once these four processes are performed on a population, the population is considered to have advanced one generation. This is performed iteratively until a stopping criteria is being satisfied. This criteria is usually a maximum number of generations, or a relative value of the objective function.

Multi-objective genetic algorithm

The main difference when dealing with multi-objective problems as opposed to single-objective ones, is that comparing two individuals is not always as direct. In situations where one of the individuals performs better on one objective function, but worse on another one, choosing which individual to keep is not straightforward (this direct comparison is for instance used in the selection or elitism processes). Due to these situations where a clear best individual can not be chosen, the solution to a multi-objective problem is no longer an individual, but instead a group of them, called pareto-front. In order to deal with multi-objective problems, the NSGA-II algorithm proposed by Deb et al. [15] has been implemented in HAMON.

In order to address the issue of comparing two individuals, the dominance operator ($\prec$) was introduced in the NSGA-II algorithm. Let N be the number of objective functions, i_1 and i_2 the two individuals to be compared and $o_m(j)$ the value of the mth objective function of the j^{th} individual. Assuming all the objective functions are to be minimized (if this is not the case, the corresponding greater than or lower than sign is used for each objective function) then individual i_1 dominates individual i_2 if:

$$i1 \prec i2 \leftrightarrow o_m(i_1) \leq o_m(i_2),\ \forall m \in \{1,...,N\} \ \wedge\ \exists k \in \{1,...,N\} \ :\ o_k(i_1) < o_k(i_2) \tag{2.72}$$

The dominance operator is used to classify the individuals of a population into ranks, being rank one the best. Individuals that are not dominated by any of the individuals of the population belong to rank one, individuals which are dominated by only rank one individuals are rank two and so on.

The crowded distance parameter (cd) is another concept introduced in NSGA-II. It aims to quantify the density of the surroundings of an individual in the objective functions space. Let $\mathcal{A}_m$ be the set containing individuals with a common rank sorted according to the value of their objective function m, M the size of $\mathcal{A}_m$ and o_m^{max} and o_m^{min} the maximum and minimum values of the objective function m in that set respectively. The

crowded distance for each of the individuals i , $cd(i)$, belonging to the set is calculated as follows,

$$
\begin{aligned}
&\text{for each i, set } cd(i) = 0 \\
&\text{for } m = 1 \text{ to } N \\
&\quad cd\big(\mathcal{A}_m(1)\big) = cd\big(\mathcal{A}_m(M)\big) = \infty \\
&\quad \text{for } i = 2 \text{ to } (M-1) \\
&\qquad cd\big(\mathcal{A}_m(i)\big) = cd\big(\mathcal{A}_m(i)\big) + \Big[o_m\big(\mathcal{A}_m(i+1)\big) - o_m\big(\mathcal{A}_m(i-1)\big)\Big] / \big(o_m^{max} - o_m^{min}\big)
\end{aligned}
\tag{2.73}
$$

Taking the concepts of crowded distance and rank into account, when comparing two individuals the one with lower rank is considered the better performing one. In the case where they belong to the same rank, the one with the largest crowded distance is chosen, thus aiming at exploring in the objective function space.

The NSGA-II implemented in HAMON still uses tournament selection (of size two), crossover and mutation processes. These are done in the same manner as for the single-objective case (except from the difference on how individuals are compared in tournament selection). The elitism process is the one that differs the most. NSGA-II applies an elitism process that involves the newly generated population through mutation, crossover and selection (here called Q), and the previous population (here called P). Both populations are combined into a new one with twice the size, PQ, and their individuals are ranked and the crowded distance computed. The population P of the new generation (which is of the same size as the original one) is obtained by the process illustrated in Fig. 2.8. First, rank one individuals of population PQ are introduced in the new population P. Later on, the individuals of rank two are added and so on. This process proceeds until adding all the individuals of a certain rank will max out the size of the new population P of the next generation. Then, only the individuals with larger crowded distance are added until the new population is of the right size, and the rest of the individuals are discarded.

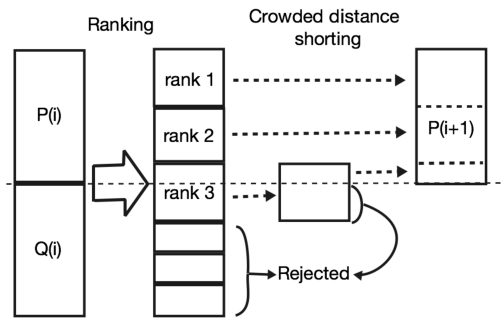


Figure 2.8: *Elitism process in the NSGA-II algorithm. Here i represents the generation number of the population.*

Differential evolution

As well as genetic algorithms, differential evolution is a stochastic optimization method that falls under the evolutionary algorithms category [55]. In HAMON's implementation of the DE, the general algorithm for both single- and multi-objective problems is the same as that of the GA; but mutation, recombination (analogous to crossover) and selection are performed differently. The main processes in DE are explained here with a notation similar to the one used by Rai [46]. As previously mentioned, each individual's design variables are encoded in their chromosome when using GA. On the other hand, on DE each individual can be considered a parameter vector containing its design variables:

$$\mathbf{X_{i,n}} = \left[x_{1,i,n}, x_{2,i,n}, ..., x_{D,i,n}\right]$$

where $\mathbf{X_{i,n}}$ is the parameter vector of the i^{th} individual of n^{th} generation's population, D is the number of variables and $x_{j,i,n}$ is the j^{th} design variable of that individual.

The following three processes are used to evolve $\mathbf{X_{i,n}}$ to the next generation's $\mathbf{X_{i,n+1}}$:

1. **Mutation**: three individuals are randomly selected from the population, $\mathbf{X_{a,n}}$, $\mathbf{X_{b,n}}$ and $\mathbf{X_{c,n}}$, with $a \neq b \neq c \neq i$. Then a new individual $\mathbf{W}$ is created as:

$$\mathbf{W} = \left[w_1, w_2, ..., w_D\right] = \mathbf{X_{a,n}} + F(\mathbf{X_{b,n}} - \mathbf{X_{c,n}}) \tag{2.74}$$

   where $0 < F < 1$ is a user define parameter. If any of the variables of $\mathbf{W}$ was to fall outside the allowed design range, they are properly adjusted.

2. **Recombination**: the newly created individual ($\mathbf{W}$) is given the chance to combine itself with the original one ($\mathbf{X_{i,n}}$) to generate a new individual ($\mathbf{Z}$) as:

$$\mathbf{Z} = \left[z_1, z_2, ..., z_D\right], \text{ where } z_j = \begin{cases} w_j & \text{if } \ r_j \leq C \\ x_{j,i,n} & \text{if } \ r_j > C \end{cases} \tag{2.75}$$

   where r_j is uniformly distributed random number, and $0 < C < 1$ a user defined parameter.

3. **Selection**: the last step is to make $\mathbf{X_{i,n+1}}$ a copy of the best performing individual when comparing $\mathbf{Z}$ and $\mathbf{X_{i,n}}$. Selection process is only done in single-objective optimization, as it can be interpreted as an elitism process as well, since it ensures that one generation is never worse than the previous one. There is no need to do this in multi-objective optimization, as the same process as in the NSGA-II algorithm is used (see Fig. 2.8) and it already compares the old and the new generation in order to form the best performing possible one.

Chapter 3
Vaidations and results

In this section some validation cases used to verify the correct implementation of the different methods described in chapter 2, as well as some results are presented.

3.1 LODI based outlet

Here the behaviour of the three different types of pressure outlets described in Sec 2.3.2 are tested, namely; perfectly-reflecting, perfectly non-reflecting and partially-reflecting outlet. In order to test the response, a half sine wave velocity pulse is excited at the inlet and convected through the domain. The results at three different time instances can be seen in Fig. 3.1 in the form of pressure fluctuations along the domain.

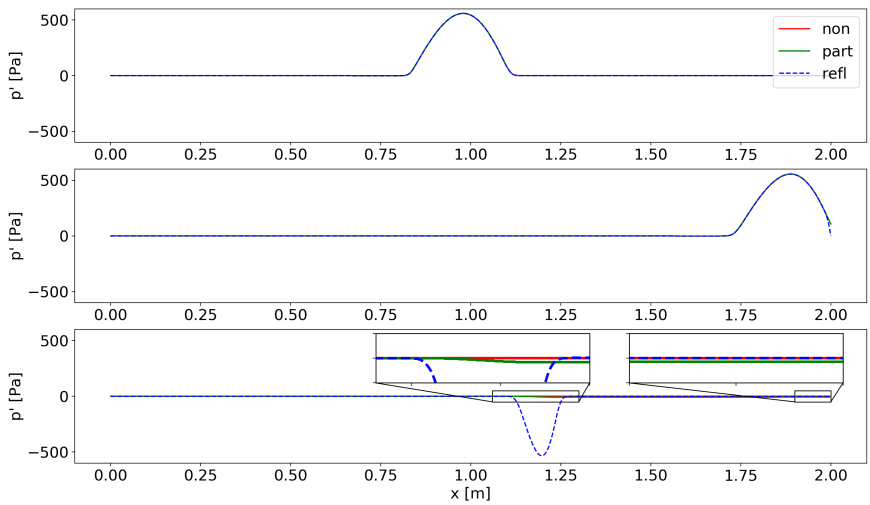


Figure 3.1: *Pressure fluctuation comparison between the three LODI based outlets; perfectly non-reflective, partially-reflective and perfectly-reflective labeled "non", "part" and "refl" respectively in the figure. Three different time instances; top: wave travelling from left to right in the middle of the domain, middle: wave reaching the outlet and bottom: original wave has exited the domain, reflections might have bounced back.*

At the time of the top figure, the wave is still travelling inside the computational domain and since the pressure used to initialize the solution is the same as the one specified for the outlet boundary, the three solutions are identical. In the middle figure, the wave is reaching the location of the outlet and a minor difference, almost negligible,

can be observed. In the bottom one, the original wave has already left the computational domain, therefore whatever pressure fluctuation is remaining inside the domain is non physical and has been caused by the outlet boundary condition. It can clearly be seen in the middle of the domain how the perfectly reflecting formulation has left an almost undamped wave with an amplitude reaching almost 500 Pa, same as the original wave. By taking a closer look at the area near the outlet where the zoom in is shown, one can see that the partially-reflecting formulation struggles a little bit to keep the exact pressure level at the outlet. Even though this difference is not of great importance (6 Pa deviation out of the around 500 Pa of the incoming wave), it can also be appreciated in the second zoomed in figure closer to the center of the domain. The best performing formulation in this case, as expected, is the perfectly non-reflective one, as it is able to maintain the right pressure level with zero reflections emanating from the outlet. As previously mentioned in Sec. 2.3.2, the problem with this formulation is that since no information from the outside is allowed to travel inside the domain, setting the right pressure level is not possible (except in a case like this one, where the pressure level was right from the initialization), therefore the partially-reflective one is been set as default in G3D::Flow.

3.2 Sliding grid

To test the implementation of the GGI and the sliding-grid a URANS simulation of a slightly modified version of the VINK compressor [30, 24] is performed. The blade count of the compressor has been modified to have 50 rotor and 90 stator blades instead of the 51 and 88 of the original design. This change allows to perform a simulation with sliding-grid and standard periodic boundary conditions, where 5 rotors and 9 stators are considered, as both cover a 36° sector. Furthermore, in order to reduce the computational resources, only a sector that covers around 15% of the whole blade span is used (same as used in Paper D). To validate the correct implementation the flow properties right before and right after the interface are sampled along a constant radius line and compared. Results for the axial component of the velocity can be seen in Fig. 3.2.

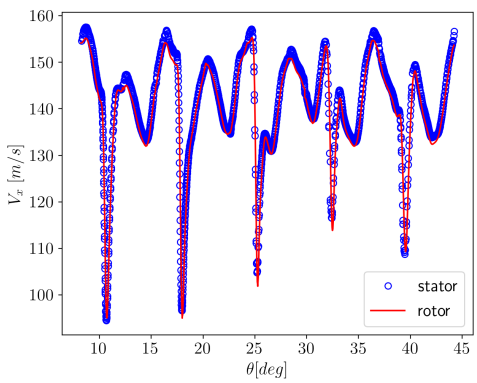


Figure 3.2: *Axial velocity comparison before and after the sliding grid interface.*

It can clearly be seen from the figure, how the five wakes emanating from the five rotors appear and that the agreement between the rotor and stator domain is good. Axial velocity is only shown here for brevity, but it has been checked that the other two velocity components (accounting for change in frame of reference) as well as thermodynamic properties, such as temperature, density and pressure have the same level of agreement.

3.3 Chorochronic periodicity and boundary decoupling

The correct implementation of the chorochronic pitch-wise boundaries, as well as the assumption that the decoupling of the boundaries (achieved by introducing extra sectors) improves the convergence of the temporal Fourier coefficients' moving average is verified here. To do so, a test case where a vortical gust is introduced is used, and three simulations are performed; standard periodic (this is used as benchmark), standard chorochronic set up and decoupled chorchronic set up (see Fig. 2.4). The axial contours of the three simulations is shown in Fig. 3.3, where the phase shift applied in the pitch-wise boundaries can clearly be seen.

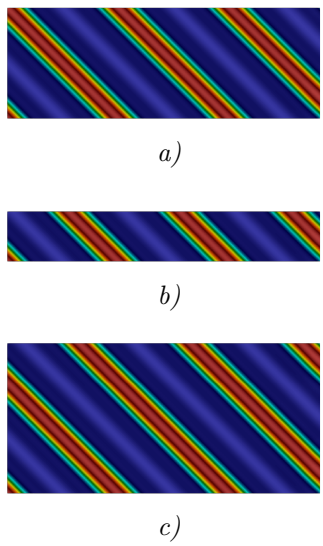


Figure 3.3: *Pitch-wise chorochronic and decoupling assessment. Axial velocity contours; a) standard periodic, b) standard chorochronic and c) decoupled chorochronic. Same colorbar in all three contour plots.*

To perform a more quantitative comparison the flow properties are extracted over a line that goes through the middle of the domain and the results are shown in Fig. 3.4 where an excellent agreement can be seen.

To test the boundary decoupling previously described, the convergence rate of the

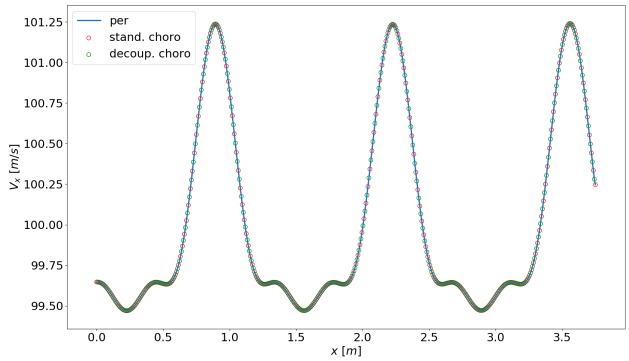


Figure 3.4: *Axial velocity comparison. For visibility enhancement not all the points have been plotted for standard and decoupled chorochronic, and they have been alternated.*

moving average for these two setups are compared. In order to measure the convergence rate, ϵ, as defined in Eq. 3.1 is used,

$$\epsilon = \frac{q(t) - Q(t-T)}{q(t)} \tag{3.1}$$

where $q(t)$ is the flow variable at time t, T is the time period of the signal and $Q(t-T)$ is the reconstructed variable using the temporal Fourier coefficients one period back in time. The comparison of the convergence rate is shown in Fig. 3.5 where it can be clearly seen that the decoupled set up converges not only faster, but also to a lower residual value.

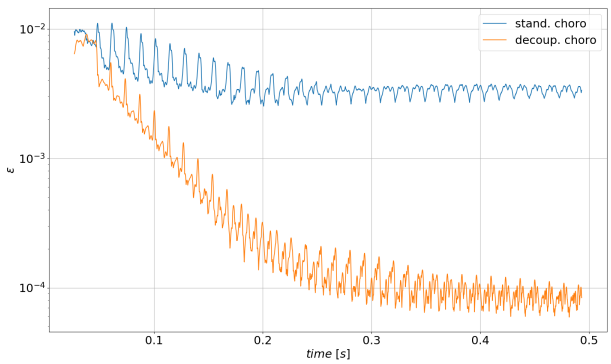


Figure 3.5: *Comparison of the convergence of the moving average between both chorochronic pitch-wise set up strategies.*

A more thorough validation of the implementation of the entire chorochronic method (including the rotor-stator interface) is presented in Paper D.

3.4 Synthetic turbulence

Here a validation test for the two methods for generating synthetic fluctuations at the inlet is presented, namely; the synthetic eddy method and the method based on Fourier coefficients. In this case the homogeneous decaying isotropic turbulence test case has been chosen, where the normal components of the Reynolds stress tensor are equal to one. A domain of sizes 6π by 2π by 2π in the x, y and z directions respectively is used which is discretized using 384 cells in the stream-wise direction and 128 in the y and z directions. A mean velocity of 20 m/s, integral length scale of 0.5 m and the Smagorinsky model is used as a subgrid scale model. Periodic boundary conditions are used on the y and z directions and the flow statistics are averaged over three through flows.

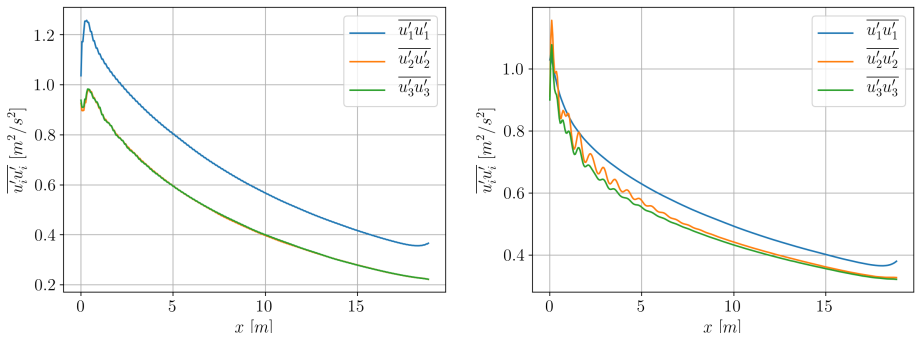


Figure 3.6: *Normal Reynolds stress components along the length of the domain. a) SEM and b) Fourier based synthetic turbulence.*

Figure 3.6 shows the results obtained with both methods for the normal stresses (the off-diagonal stresses were below 10^{-2} for both cases and are not shown here for brevity). Taking a look at the results obtained with SEM (Fig. 3.6 a)) one can observe that the values start relatively close to the desired ones but there is a spike close to the inlet. The reason of why this is happening is still unclear, but similar behaviours have been reported by other authors. In the original work by Jarrin et al. [28], the individual stresses are not reported, but a similar kink can be seen in the turbulent kinetic energy. In the work by Matha et al. [34], this kink appears again even though the synthetic turbulence generation method is different. A very similar behaviour is seen in the results obtained with the Fourier based method, but in this case some oscillations of the $\overline{u_2'u_2'}$ and $\overline{u_3'u_3'}$ stresses is seen close to the inlet, one possible explanation being that this method does not generate a periodic inlet perturbation field.

3.5 Supersonic base flow of a cylindrical aferbody

In this section the supersonic base flow experimentally tested in [26], where the incoming Mach number is 2.46, is simulated using the G3D::Flow solver. This flow is characterized by an expansion fan emanating from the edge of the cylinder that leads to a recirculation bubble. The shear layer generated at the sharp end of the cylinder contains the circulation bubble and then, an oblique shock shock is formed due to the recompression. This phenomena can be seen in the instantaneous Mach number field depicted in Fig. 3.7.

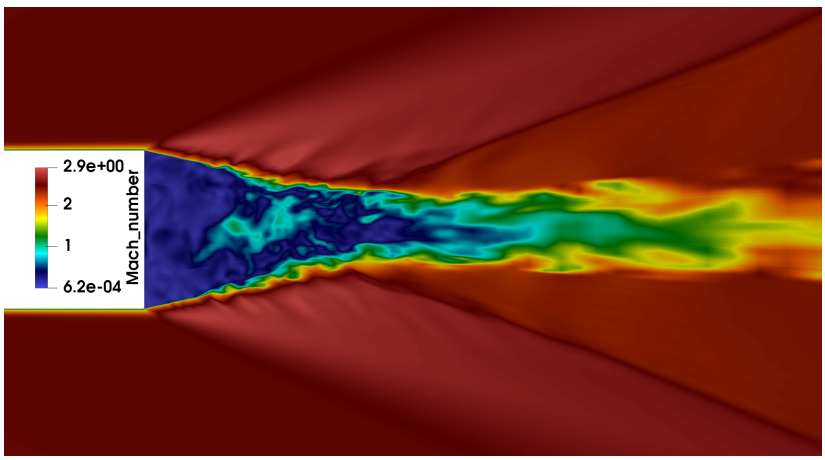


Figure 3.7: *Instantaneous Mach number contour of the DDES simulation of the supersonic base flow.*

The simulation is performed using the Spalart-Allmaras based DDES turbulence model with $\Delta = \Delta_{SLA}$ [48]. Moreover the dual time stepping method that uses a second order backward Euler temporal discretization is used [49], which allows to set the time step $\Delta t = 5.6 \cdot 10^{-6}$ s. A third order accurate upwind-biased low dissipation scheme is used to perform the convective fluxes and a second order central differencing one for the diffusive fluxes.

One important aspect to predict in this kind of flow, is the low pressure in the base of the cylinder, as this can have a major contribution on the total drag. Figure 3.8 shows the values of C_p obtained along the base. As it can be seen, some discrepancies are found. Guseva et al. [25] reported a very similar C_p profile when using DDES together with $\Delta = \Delta_{SLA}$. Forsythe et al. [19], found similar deviations while performing several computations, where they used different cylinder lengths, and several different values of the C_{DES} constant in the DDES model. Throughout this section, markers will represent experimental values and the solid line the values obtained from the CFD simulation.

In order to see how well the length of the recirculation bubble is predicted, Fig. 3.9 compares time averaged axial velocity along the center line of the cylinder, with that measured on the experiments. The velocity has been normalized by the free-stream velocity, U_0. As can be seen there is a good agreement on the prediction of the length

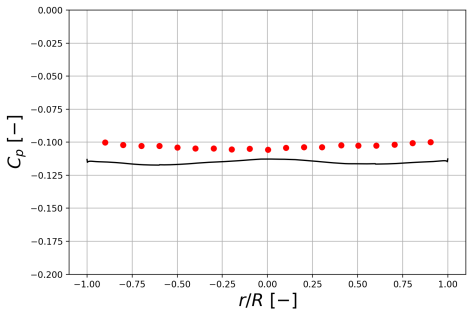


Figure 3.8: *Cp on the base of the cylinder. r represent the radial position and R the total radius of the cylinder.*

of the recirculation bubble. but the deep is overpredicted. Once again, very similar behaviour has been found in literature [19, 25]. As shown by Guseva et al. [25] and Carlsson [11], using $\Delta = \Delta_{max}$ instead of $\Delta = \Delta_{SLA}$, seems to better predict the deep found in the recirculation, but on the other hand, it predicts a much longer bubble length.

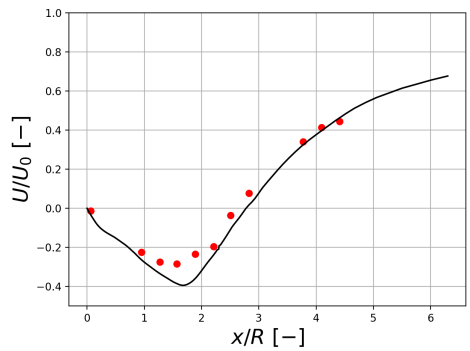


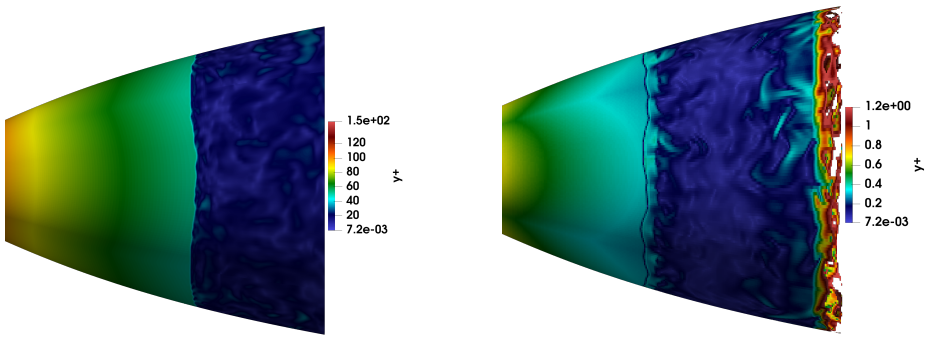
Figure 3.9: *Normalized time averaged axial velocity along the center line of the cylinder. The base is located at* $x = 0$.

The rest of the available experimental data has been compared to the one obtained with G3D::Flow and it is not show here for brevity purposes. The results, despite finding some differences with the experiments, show very similar trend to that reported in the literature when similar methods are used.

3.6 Overexpanded nozzle flow simulations

Two transient flow simulations were performed using the Spalart-Allmaras based DDES on the VOLVO S6 nozzle geometry operating under overexpanded conditions. The mesh

for the first simulation was prepared with the objective in mind of having a resolved boundary layer ($y^+ < 1$), here referred to as WR, whereas for the second one wall functions were to be used, referred to as WF. When generating the mesh for the WF, only the cell distribution and cell count in the wall normal direction were modified, leaving the tangential and axial resolutions unchanged. Both the convective and diffusive schemes used were the same in both cases. Figure 3.10 shows the y^+ values along the divergent part of the nozzle for a given time instance, where it can be seen that the objective values set during the meshing design process are accomplished.



a) Wall function, WF *b) Wall resolved, WR*

Figure 3.10: y^+ *values across the nozzle wall.*

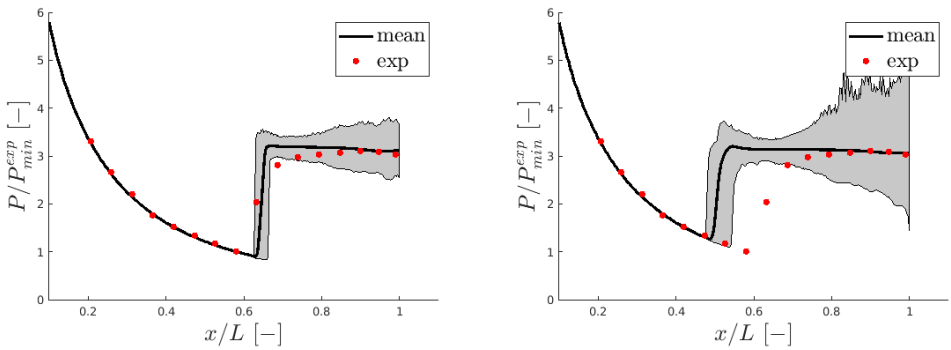
Due to the difference in cell sizes, two solver setups were used for the simulations. This was done because using a three stage explicit Runge-Kutta solver on the wall resolved case became unfeasible, as it required a time step in the order 10^{-10} seconds to get CFL values which would keep the solver stable. Therefore a dual time stepping method with second order backward Euler temporal discretization and a five stage Runge-Kutta scheme for the inner iterations was used in the wall resolved case. More information on the dual time stepping implementation in G3D::Flow can be found in [49]. Some of the differences concerning both simulation setups are outlined in Tab. 3.1.

	WR	WF
millions of cells	56.6	36.7
max local CFL	~ 600	~ 0.9
temp. scheme	Dual-time stepping (~ 30 subiters per time step)	3 stage Runge-Kutta
y^+	$y^+ < 1$ in divergent area	$y^+ < 150$ over the whole nozzle

Table 3.1: Differences in the setups of the VOLVO S6 simulations.

In order to see if the wall pressure is correctly predicted by the simulations, the azimuthal-temporal averages at every axial location are plotted against the experimental data in Fig. 3.11. Here L represents the length of the divergent part of the nozzle, the

throat is located at $x = 0$ and P_{min}^{exp} is the minimum wall pressure of the experimental data. The shadowed area represents the maximum and minimum values over the whole simulated time and over the whole circumference of the nozzle. This shadowed area can be used to visualize the shock wave movement as well as the more unstable behaviour of the flow after the separation. The results appear quite surprising as the wall function simulation nicely predicts the separation, whereas the wall resolved one predicts a way earlier separation point. This can also be observed in the Mach number field contour shown in Fig. 3.12.



a) Wall function, WF *b) Wall resolved, WR*

Figure 3.11: *Nozzle wall pressure, DDES vs experiment.*

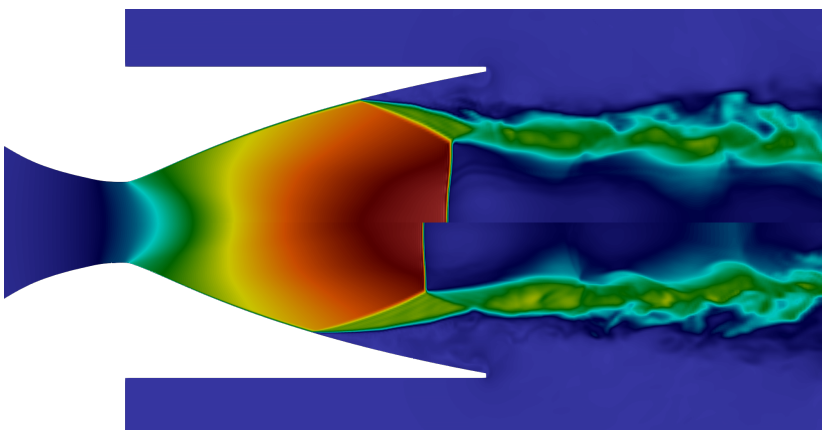


Figure 3.12: *Instantaneous Mach number contour of both DDES simulations. Top, wall function and bottom, wall resolved simulation.*

The reason of why such different results are obtained is still unclear. Despite the time step used in the wall resolved simulation being approximately two to three times as large as the one used in the wall function simulation, the dual time stepping solver showed nice

convergence properties, with residuals continuously decreasing, dropping four orders of magnitude every time step. More details on the wall function simulation can be found in Paper F.

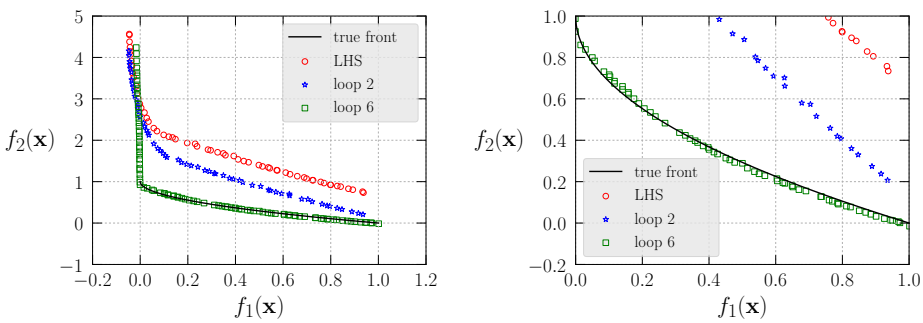
3.7 HAMON

In this section a validation case where HAMON is used together with RBFs is presented (see Fig. 2.6). The chosen test case is a well known function that is often used as a benchmark for multi-objective unconstrained problems as it has an analytical solution. The function is often referred to as the ZDT 1 function [63], which definition is given in Table 3.2.

Table 3.2: Definition of ZDT 1.

Function	variables, m	Variable range	Functions to minimize ($f_1(\mathbf{x})$ & $f_2(\mathbf{x})$)
ZDT 1	30	$[0,\ 1]$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x})\left[1 - \sqrt{x_1/g(\mathbf{x})}\right]$ $g(\mathbf{x}) = 1 + 9\left(\sum_{i=2}^{m} x_i\right)/(m-1)$

In this case the optimization has been performed using the multi-objective DE in HAMON, but very similar results were also observed using the NSGA-II algorithm. The results for the optimization can be seen in Fig. 3.13 where the evolution of the predictive capabilities of the RBF are shown. It can clearly be seen that when the RBF is fed only the designs from the LHS, the pareto-front obtained using the RBF is very far from the true one. The more information is given to the RBF, the closer it gets to the true front, being really close by loop number 6.



a) Entire pareto-front *b) Close up view*

Figure 3.13: *Evolution of the DE pareto-front over several optimization loops.*

A more extensive validation of the different possibilities that HAMON offers can be found in Papers B and C.

Chapter 4
Division of work

4.1 Paper A

D. Lindblad, G. Montero Villar, N. Andersson, A. Capitao Patrao, S. Courty-Audren, and G. Napias. "Aeroacoustic Analysis of a Counter Rotating Open Rotor Based on the Harmonic Balance Method". *2018 AIAA Aerospace Sciences Meeting*. 2018

The geometry was designed by Alexandre and it was later meshed by Gael and Suk-kee. The implementation of the rotor-stator interface as well as the pitch-wise boundary conditions used into the G3D::Flow solver was done by both Daniel and Gonzalo with support from Niklas. The simulation, post-processing and writing of the article was done by Daniel.

4.2 Paper B

G. Montero Villar, D. Lindblad, and N. Andersson. "Multi-Objective Optimization of an Counter Rotating Open Rotor using Evolutionary Algorithms". *2018 Multidisciplinary Analysis and Optimization Conference*. 2018

The decision of which design variables to use as well as the ranges allowed for them was decided by Gonzalo and Daniel. The set up of the CFD simulation in STAR-CCM+ was done by Daniel. The development of the optimization framework, the automation of geometry generation and meshing, performing the optimization, post-processing and writing of the article was done by Gonzalo. Niklas provided support along the way.

4.3 Paper C

G. Montero Villar, D. Lindblad, and N. Andersson. "Effect of Airfoil Parametrization on the Optimization of Counter Rotating Open Rotors". *AIAA Scitech 2019 Forum*. 2019

The design variables used as well as the ranges allowed for them was decided by Gonzalo and Daniel. The set up of the CFD simulation in STAR-CCM+ was done by Daniel. The development of the optimization framework, the automation of geometry generation (when using NACA profiles) and meshing, performing both optimizations, post-processing and writing of the article was done by Gonzalo. Daniel extended the existing geometry generation code to allow for CST profiles. Niklas provided support along the way.

4.4 Paper D

G. Montero Villar, D. Lindblad, and N. Andersson. "Investigation of Phase-Lagged Boundary Conditions for Turbulence Resolving Turbomachinery Simulations". *AIAA AVIATION 2020 FORUM*. 2020

The mesh was generated by Gonzalo. The rotor-stator interface as well as the pitch-wise boundary conditions used were implemented into G3D::Flow by Gonzalo and Daniel with support from Niklas. The simulation set up, running and post-processing as well as writing the article was done by Gonzalo.

4.5 Paper E

G. Montero Villar and N. Andersson. "Initial test of SVD based Phase-Lagged Boundary Conditionsfor Turbomachinery Simulations in the G3D::Flow Solver". *Technical report, Chalmers University of Technology*. 2021

The mesh was generated by Gonzalo. The rotor-stator interface as well as the pitch-wise boundary conditions used were implemented into G3D::Flow by Gonzalo with support from Niklas. The simulation set up, running and post-processing as well as writing the article was done by Gonzalo.

4.6 Paper F

G. Montero Villar, N. Andersson, and Östlund J. "Nozzle Side Loads Prediction using a Hybrid RANS/LES Method". *Technical report, Chalmers University of Technology*. 2021

The mesh, the simulation set up, running and post-processing as well as writing the article was done by Gonzalo. Niklas and Jan provided a great deal of support when discussing results.

Chapter 5
Concluding remarks

An optimization platform based on evolutionary algorithms called HAMON has been developed and validated against several well known benchmark functions where satisfactory results have been obtained. The possibility of coupling it with meta-modeling techniques to speed up the optimization process when necessary, has also been validated by benchmark functions. HAMON has been used to aerodynamically optimize a counter-rotating open rotor configuration in two different scenarios. In the first one, insights into what a good design might look like were used to facilitate the optimization, whereas on the second one, almost zero previous knowledge was assumed. In both cases satisfactory results were obtained.

A sliding grid interface has been implemented into G3D::Flow and validated, which allows to perform transient turbomachinery simulations where blade rows in relative motion are allowed to be coupled. Moreover, phase-lagged boundary conditions for the pitch-wise boundaries, as well as the blade-row interface have been implemented. These are used to reduce the computational domain in simulations where the difference in blade count between adjacent blade rows makes it computationally expensive to find a common tangential sector. For this set up, two different types of methodologies have been implemented. The first one relies on decomposing the flow field into Fourier coefficients in order to couple the interfaces (i.e. chorochronic method) and the second one uses singular value decomposition of the flow variables. The latter method is a relatively new one, introduced in 2016, and to the author's knowledge it is available in only one other CFD code around the world (based on published literature). In order to alleviate an existing error feedback loop that emerges from the use of phase-lagged boundary conditions, a decoupling method has been implemented which has shown great convergence and stability enhancements when using together with both aforementioned phase-lag methodologies.

A transient Hybrid RANS/LES simulation based on the Spalart-Allmaras turbulence model has been performed on a overexpanded nozzle configuration. A nozzle pressure ratio that would result in internal free-shock separation was chosen with the aim of being able to predict the side loads acting on the nozzle structure. In this simulation, the use of wall-function was chosen, which allowed the use of an explicit solver with an affordable time step. The used CFD set up, wall functions included, has shown to be a suitable choice as a very good agreement is found on the predicted time averaged nozzle wall pressure when compared against available experimental data. The simulation also predicted side loads that are in very good agreement with data obtained in a previous experimental campaign, reducing the difference by over 50% when compared to previously reported results in the same geometry under the same conditions.

Finally, some extra additions have been introduced to G3D::Flow which concern boundary treatment, such as NSCBC or synthetic turbulence injection methods.

5.1 Suggested future work

Due to the novelty and scarcity of codes in which the phase-lagged boundary conditions that use singular value decomposition are implemented, the amount of literature describing and applying the method is really limited. Therefore, a study where full 3D turbomachinery simulations are performed using different phase-lagging methods to directly compare their performance would be of great value and interest, both using URANS and hybrid RANS/LES or LES.

Very good agreement predicting side loads has been found for the chosen nozzle pressure ratio on the performed hybrid RANS/LES computation. Whether this level of agreement can be maintained during the entire start-up sequence is not known yet, therefore, performing a full nozzle start-up sequence simulation, where the nozzle pressure ratio is driven all the way up to the full throttle condition using the same CFD set up would be of great interest.

References

[1] Airbus. *Global Market Forecast 2016-2035*. 2016.

[2] S. R. Allmaras and F. T. Johnson. "Modifications and clarifications for the implementation of the Spalart-Allmaras turbulence model". *Seventh International Conference on Computational Fluid Dynamics (ICCFD7)*. Vol. 1902. Big Island, HI. 2012.

[3] N. Andersson. *A Study of Subsonic Turbulent Jets and Their Radiated Sound Using Large-Eddy Simulation*. PhD thesis, Chalmers University of Technology, 2005.

[4] R. Avellan and A. Lundbladh. *Air propeller arrangement and aircraft*. US Patent App. 13/519,588. 2009.

[5] T. Biesinger, C. Cornelius, C. Rube, A. Braune, R. Campregher, P. G. Godin, G. Schmid, and L. Zori. "Unsteady CFD methods in a commercial solver for turbomachinery applications". *ASME Turbo Expo 2010: Power for Land, Sea, and Air*. American Society of Mechanical Engineers Digital Collection. 2010, pp. 2441–2452.

[6] M. Billson. *Computational techniques for turbulence generated noise*. PhD thesis, Chalmers University of Technology, 2004.

[7] M. Billson, L.-E. Eriksson, and L. Davidson. "Jet noise prediction using stochastic turbulence modeling". *9th AIAA/CEAS aeroacoustics conference and exhibit*. 2003, p. 3282.

[8] M. Billson, L.-E. Eriksson, L. Davidson, and P. Jordan. "Modeling of synthetic anisotropic turbulence and its sound emission". *10th AIAA/CEAS aeroacoustics conference*. 2004, p. 2857.

[9] A. Capitao Patrao. *On the Aerodynamic Design of the Boxprop*. PhD thesis, Chalmers University of Technology, 218.

[10] A. Capitao Patrao, T. Grönstedt, A. Lundbladh, and G. Montero Villar. Wake Analysis of an Aerodynamically Optimized Boxprop High-Speed Propeller. *Journal of Turbomachinery* (2019).

[11] M. Carlsson. *Development of Numerical Methods for Accurate and Efficient Scale-Resolving Simulations*. Licenciate thesis, Chalmers University of Technology, 2021.

[12] S. Connell, M. Braaten, L. Zori, R. Steed, B. Hutchinson, and G. Cox. "A Comparison of Advanced Numerical Techniques to Model Transient Flow in Turbomachinery Blade Rows". *ASME 2011 Turbo Expo: Turbine Technical Conference and Exposition*. American Society of Mechanical Engineers. 2011, pp. 1241–1250.

[13] L. Davidson. *Fluid mechanics, turbulent flow and turbulence modeling*. 2015.

[14] L. Davidson and M. Billson. Hybrid LES-RANS using synthesized turbulent fluctuations for forcing in the interface region. *International journal of heat and fluid flow* **27**.6 (2006), 1028–1042.

[15] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *Evolutionary Computation, IEEE Transactions on* **6**.2 (2002), 182–197.

[16] JI Erdos, EM Alzner, and W McNally. Numerical solution of periodic transonic flow through a fan stage. *AIAA Journal* **15**.11 (1977), 1559–1568.

[17] L. E. Eriksson. Development and validation of highly modular flow solver versions in g2dflow and g3dflow. *Volvo Aero Corporation, Trollhättan, Sweden, Technical Report* 9970-1162 (1995).

[18] *Flightpath 2050, Europe's Vision for Aviation*. https://ec.europa.eu/transport/sites/transport/files/modes/air/doc/flightpath2050.pdf. Accessed: August 2020.

[19] James Forsythe, Klaus Hoffmann, and Jean-Francois Dietiker. "Detached-eddy simulation of a supersonic axisymmetric base flow with an unstructured solver". *Fluids 2000 Conference and Exhibit*. 2000, p. 2410.

[20] G3D::Flow homepage (accessed 2020). `https://nikander.github.io/g3dflow/`.

[21] G. A. Gerolymos, G. J. Michon, and J. Neubauer. Analysis and Application of Chorochronic Periodicity in Turbomachinery Rotor/Stator Interaction Computations. *Journal of propulsion and power* **18**.6 (2002), 1139–1152.

[22] M. B. Giles. Calculation of unsteady wake/rotor interaction. *Journal of Propulsion and Power* **4**.4 (1988), 356–362.

[23] GitHub repository for HAMON (2020). `https://github.com/gmonterovillar/HAMON`.

[24] GitHub repository for VINK (2017). `https://github.com/nikander/VINK`.

[25] E. K. Guseva, A. V. Garbaruk, and M. Kh. Strelets. Assessment of delayed DES and improved delayed DES combined with a shear-layer-adapted subgrid length-scale in separated flows. *Flow, Turbulence and Combustion* **98**.2 (2017), 481–502.

[26] JL Herrin and J Craig Dutton. Supersonic base flow experiments in the near wake of a cylindrical afterbody. *AIAA journal* **32**.1 (1994), 77–83.

[27] *IATA Annual Review 2019*. https://www.iata.org/contentassets/c81222d96c9a4e0bb4ff6ced0126f0bb/iata-annual-review-2019.pdf. Accessed: August 2020.

[28] N. Jarrin, S. Benhamadouche, D. Laurence, and R. Prosser. A synthetic-eddy-method for generating inflow conditions for large-eddy simulations. *International Journal of Heat and Fluid Flow* **27**.4 (2006), 585–593.

[29] C. Kissner, S. Guérin, P. Seeler, M. Billson, P. Chaitanya, P. Carrasco Laraña, H. de Laborderie, Benjamin F., et al. "ACAT1 Benchmark of RANS-Informed Analytical Methods for Fan Broadband Noise Prediction—Part I—Influence of the RANS Simulation". *Acoustics*. Vol. 2. 3. Multidisciplinary Digital Publishing Institute. 2020, pp. 539–578.

[30] M. Lejon, T. Grönstedt, N. Glodic, P. Petrie-Repar, M. Genrup, and A. Mann. "Multidisciplinary design of a three stage high speed booster". *ASME Turbo Expo 2017: Turbomachinery Technical Conference and Exposition*. American Society of Mechanical Engineers Digital Collection. 2017.

[31] D. Lindblad, G. Montero Villar, N. Andersson, A. Capitao Patrao, S. Courty-Audren, and G. Napias. "Aeroacoustic Analysis of a Counter Rotating Open Rotor Based on the Harmonic Balance Method". *2018 AIAA Aerospace Sciences Meeting*. 2018.

[32] D. Lindblad, N. Wukie, G. Montero Villar, and N. Andersson. "A Nonreflecting Formulation for Turbomachinery Boundaries and Blade Row Interfaces". *AIAA Scitech 2019 Forum*. 2019.

[33] D. Lindblad, N. A. Wukie, G. Montero Villar, and N.s Andersson. "Implementation of a Quasi-Three-Dimensional Nonreflecting Blade Row Interface for Steady and

Unsteady Analysis of Axial Turbomachines". *2018 AIAA/CEAS Aeroacoustics Conference*. 2018.

[34] M. Matha, C. Morsbach, and M. Bergmann. "A comparison of methods for introducing synthetic turbulence". *ECCOMAS–ECFD 2018-6th European Conference on Computational Mechanics (Solids, Structures and Coupled Problems)/7th European Conference on Computational Fluid Dynamics*. 2018.

[35] G. Montero Villar and N. Andersson. "Initial test of SVD based Phase-Lagged Boundary Conditionsfor Turbomachinery Simulations in the G3D::Flow Solver". *Technical report, Chalmers University of Technology*. 2021.

[36] G. Montero Villar, N. Andersson, and Östlund J. "Nozzle Side Loads Prediction using a Hybrid RANS/LES Method". *Technical report, Chalmers University of Technology*. 2021.

[37] G. Montero Villar, D. Lindblad, and N. Andersson. "Effect of Airfoil Parametrization on the Optimization of Counter Rotating Open Rotors". *AIAA Scitech 2019 Forum*. 2019.

[38] G. Montero Villar, D. Lindblad, and N. Andersson. "Investigation of Phase-Lagged Boundary Conditions for Turbulence Resolving Turbomachinery Simulations". *AIAA AVIATION 2020 FORUM*. 2020.

[39] G. Montero Villar, D. Lindblad, and N. Andersson. "Multi-Objective Optimization of an Counter Rotating Open Rotor using Evolutionary Algorithms". *2018 Multidisciplinary Analysis and Optimization Conference*. 2018.

[40] G. Mouret, N. Gourdain, and L. Castillon. Adaptation of Phase-Lagged Boundary Conditions to Large Eddy Simulation in Turbomachinery Configurations. *Journal of Turbomachinery* **138**.4 (2016), 041003.

[41] M. Olausson. *Turbomachinery aeroacoustic calculations using nonlinear methods*. PhD thesis, Chalmers University of Technology, 2011.

[42] M. Olausson and L.-E. Eriksson. "Rotor wake/stator broadband noise calculations using hybrid RANS/LES and chorochronic buffer zones". *15th AIAA/CEAS Aeroacoustics Conference (30th AIAA Aeroacoustics Conference)*. 2009, p. 3338.

[43] J. Östlund, T. Damgaard, and M. Frey. Side-load phenomena in highly overexpanded rocket nozzles. *Journal of Propulsion and Power* **20**.4 (2004), 695–704.

[44] T. J. Poinsot and S. K. Lele. Boundary Conditions for Direct Simulations of Compressible Viscous Flows. *Journal of Computational Physics* **101**.1 (1992), 104–129.

[45] A. Probst, C. Wolf, R. Radespiel, T. Knopp, D. Schwamborn, and R. Radespiel. "A comparison of detached-eddy simulation and reynolds-stress modeling applied to the flow over a backward-facing step and an airfoil at stall". *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*. 2010, p. 920.

[46] M. M. Rai. Single-and Multiple-Objective Optimization with Differential Evolution and Neural Networks. *VKI lecture series: introduction to optimization and multidisciplinary design* **58** (2006).

[47] D. H. Rudy and J. C. Strikwerda. Boundary conditions for subsonic compressible Navier-Stokes calculations. *Computers & Fluids* **9**.3 (1981), 327–338.

[48] M. L. Shur, P. R. Spalart, M. Kh. Strelets, and A. K. Travin. An enhanced version of DES with rapid transition from RANS to LES in separated flows. *Flow, turbulence and combustion* **95**.4 (2015), 709–737.

[49] E. M. V. Siggeirsson. *Aerodynamics of an Aeroengine Intermediate Compressor Duct: Effects from an Integrated Bleed System*. PhD thesis, Chalmers University of Technology, 2020.

[50] J. Smagorinsky. General circulation experiments with the primitive equations: I. The basic experiment. *Monthly weather review* **91**.3 (1963), 99–164.

[51] A. Smirnov, S. Shi, and I. Celik. Random flow generation technique for large eddy simulations and particle-dynamics modeling. *J. Fluids Eng.* **123**.2 (2001), 359–371.

[52] P. R. Spalart. "Comments on the Feasibility of LES for Wings, and on Hybrid RANS/LES Approach". *Proceedings of First AFOSR International Conference on DNS/LES, 1997*. 1997.

[53] P. R. Spalart and S. R. Allmaras. "A one-equation turbulence model for aerodynamic flows". *30th Aerospace Sciences Meeting and Exhibit*. 1992, p. 439.

[54] P. R. Spalart, S. Deck, M. L. Shur, K. D. Squires, M. Kh. Strelets, and A. Travin. A New Version of Detached-Eddy Simulation, Resistant to Ambiguous Grid Densities. *Theoretical and computational fluid dynamics* **20**.3 (2006), 181.

[55] R. Storn and K. Price. Differential Evolution–a Simple and Efficient Heuristic for Global Optimization Over Continuous Spaces. *Journal of Global Optimization* **11**.4 (1997), 341–359.

[56] G. R Tabor and MH Baba-Ahmadi. Inlet conditions for large eddy simulation: A review. *Computers & Fluids* **39**.4 (2010), 553–567.

[57] K. W. Thompson. Time dependent Boundary Conditions for Hyperbolic systems. *Journal of Computational Physics* **68**.1 (1987), 1–24.

[58] J. Tyacke, N. R. Vadlamani, W. Trojak, R. Watson, Y. Ma, and P. G. Tucker. Turbomachinery simulation challenges and the future. *Progress in Aerospace Sciences* **110** (2019), 100554.

[59] J. M. Tyler and T. G. Sofrin. *Axial flow compressor noise studies*. Tech. rep. SAE Technical Paper, 1962.

[60] S. Vasudevan, S. Etemad, L. Davidson, and G. Montero Villar. Numerical model to estimate subcooled flow boiling heat flux and to indicate vapor bubble interaction. *International Journal of Heat and Mass Transfer* **170** (2021), 121038.

[61] M. Wahde. "Biologically Inspired Optimization Methods: an Introduction". WIT Press, 2008. Chap. 3, pp. 35–67.

[62] A. Widenhorn, B. Noll, and M. Aigner. "Accurate boundary conditions for the numerical simulation of thermoacoustic phenomena in gas-turbine combustion chambers". *ASME Turbo Expo 2006: Power for Land, Sea, and Air*. American Society of Mechanical Engineers Digital Collection. 2006, pp. 347–356.

[63] E. Zitzler, K. Deb, and L. Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation* **8**.2 (2000), 173–195.