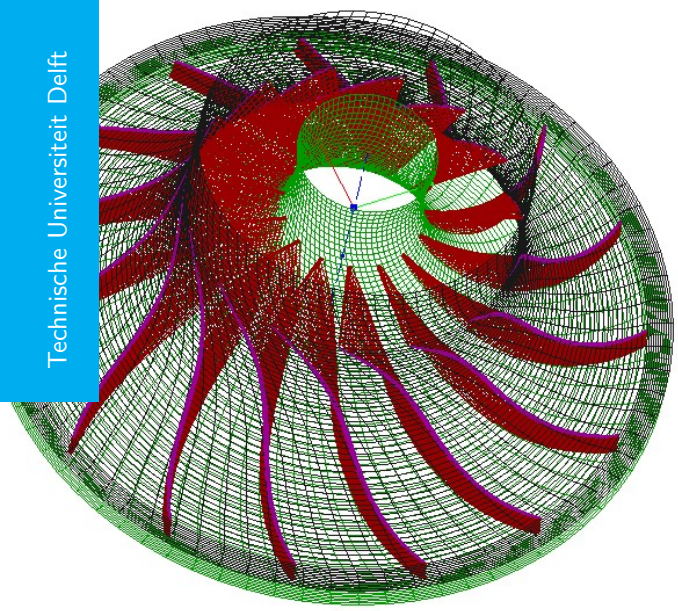
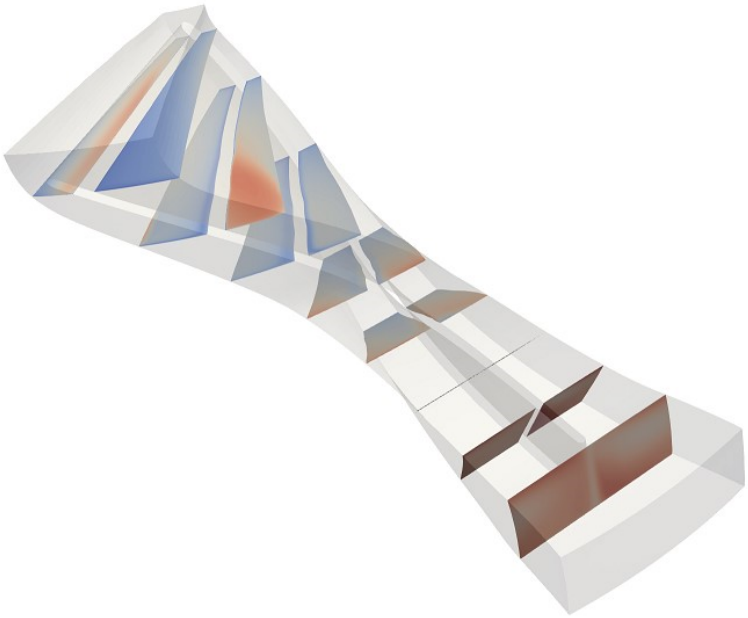
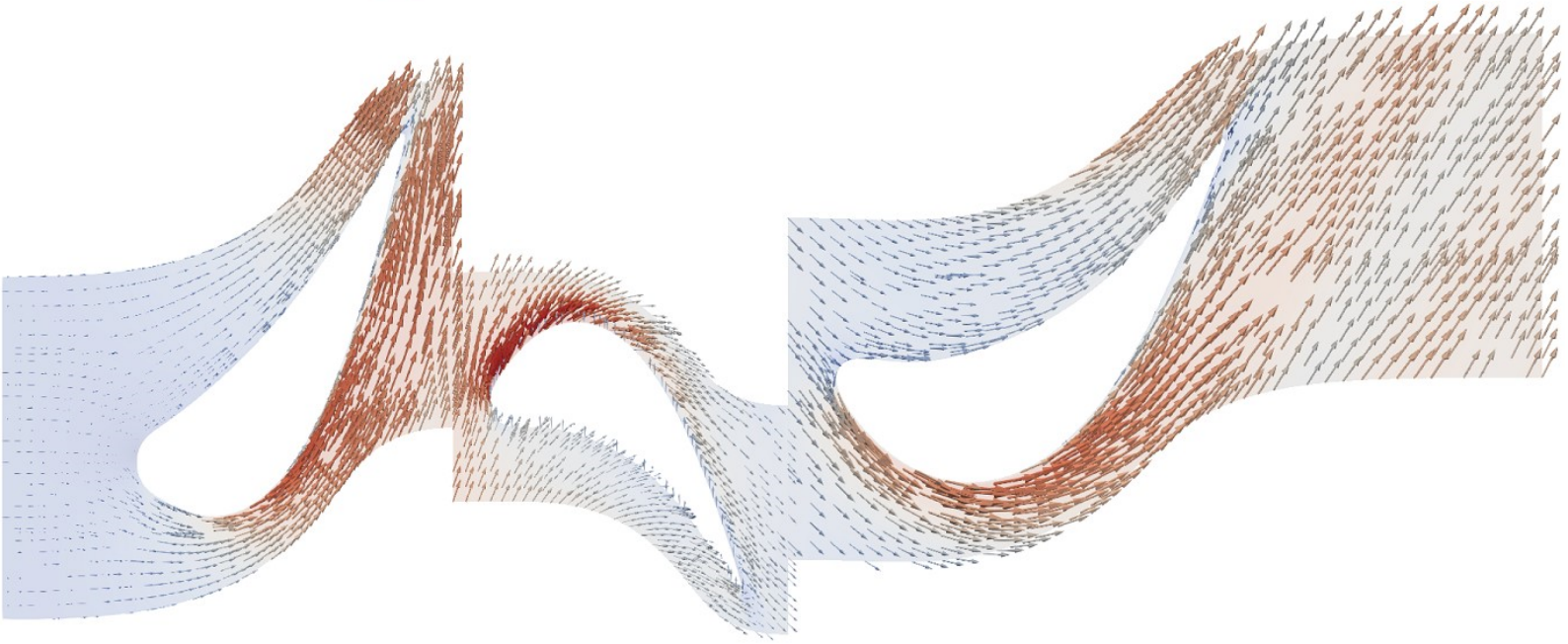
Turbomachinery flow simulations with SU2: a numerical study

M.R. Vafi

Technische Universiteit Delft







TU Delft Delft University of Technology

Challenge the future

TURBOMACHINERY FLOW SIMULATIONS WITH SU2: A NUMERICAL STUDY

by

M.R. Vafi

in partial fulfillment of the requirements for the degree of

Master of Science
in Aerospace Engineering

at the Delft University of Technology,
to be defended publicly on Tuesday March 8, 2024 at 15:00.

Supervisor:	Dr. Ir. M. Pini	
Thesis committee:	Prof. dr. Ir. P. Colonna	TU Delft
	Dr. Ir. M. Pini,	TU Delft
	Dr. Ir. D. Modesti,	TU Delft

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft Delft University of Technology

PREFACE

This thesis marks the end of my time being a student at the the TU Delft. It is the result of many years of studying and gaining knowledge, which I would like to think started during elementary school. The trajectory of this thesis has been a roller-coaster of emotions where many things did not go as planned. Many things changed since the start of this project, both involving the research plan of this thesis but also my personal situation.

During this time I have felt moments of desperation and loneliness, as well as moments of happiness and gratitude. This project has been a valuable learning experience even though it took longer then expected. Working on this thesis has taught me many new things about myself, teaching me better what I want in life. Despite the many setbacks I am happy with the final work, where I believe it provides value to the research field. I have been dreading and anticipating the moment of graduating for many years, where with the completion of this thesis this moment has finally arrived. Graduating has however only been possible due to the support of a few people, which I would like to thank first.

I would like to thank Dr. Matteo Pini for giving me this graduating opportunity and for his guidance during this thesis. The conversations we have had have always been interesting and joyful, where the feedback given during our meetings offered great value to the final work. It has been a pleasure to work with him due to his consistent kindness, where I wish we could have met more often.

I would like to thank all my study friends with whom I studied until the late hours, my roommates who made Delft truly feel like home and the people close to me who kept me sane during this whole process. Last but not least I would like to thank my parents and siblings for always supporting me and encouraging me to pursue my dreams.

M.R. Vafi
Delft, February 2024

ABSTRACT

Analysing and developing turbomachinery at off-design conditions requires the usage of robust and efficient Computational Fluid Dynamics (CFD) solvers. With the introduction of the computer, many advancements have been made in the field of numerical methods involving these flow problems. These numerical methods are used in order to solve these complex flow structures that are formed due to the interaction between the rotating machinery and the fluid. With the emergence of new design paradigms using computer-aided optimisation, novel solver methods are required which are able to deal with the increase in computational cost and the convergence difficulties following off-design conditions. One of the CFD solvers that is used for turbomachinery design is the open-source software SU2. SU2 is currently developed partially at the Delft University of Technology, where an increase in solver performance with respect to turbomachinery aerodynamics is greatly desired. The current work provides a numerical study of SU2's current performance with respect to turbomachinery analysis, as this is currently unknown.

The current performance of SU2 is to be analysed using steady RANS turbomachinery simulations. The research conducted by Xu et al. [1] will be used as reference data in order to validate SU2's performance together with data obtained from the CFD solvers CFX and Numeca. The research conducted by Xu et al. resulted in the development of their NUTSCFD solver, which showed strong performance with respect to turbomachinery analysis. Four test cases are set up and used in order to analyse SU2. The test cases that are considered for the numerical study include: the NACA 0012 airfoil, the LS89 turbine cascade, the MTU centrifugal compressor and the 1.5 stage ETH turbine. The first three test cases are validated using the results obtained by the NUTSCFD solver, where the ETH turbine is validated using CFX and Numeca. Following these test cases, SU2's performance is analysed using the residual behaviour. Xu et al. dedicate their performance increase to be the result of the Newton-Krylov method that is implemented in their NUTSCFD solver. SU2 also includes a Newton-Krylov solver, but its performance with respect to turbomachinery is also unknown. A performance assessment with respect to SU2's Newton-Krylov method involving turbomachinery analysis is therefore conducted as well.

The results obtained using the NACA 0012 and LS89 test cases show a discrepancy in solver performance involving SU2. With respect to SU2's Newton-Krylov solver, the NACA 0012 test case shows a reduction in non-linear iterations where this is not found for the LS89 test case. Both results showed however a large difference in performance when compared to the NUTSCFD solver, where SU2's standard solver was also unable to match NUTSCFD. The results obtained following these test cases have led to the development of the MTU test case, where the MTU test case was to be used in order to provide a more accurate comparison between SU2 and NUTSCFD. Instead, SU2 was unable to run the MTU test case, where the solver showed stalling behaviour. SU2's performance with respect to axial turbines is tested using the ETH test case, where the test case is validated using data obtained by the research field. This analysis provided promising results, where SU2 was able to provide results similar to that of CFX and Numeca. A comparison was also made between the results of the ETH test case obtained using SU2's standard solver and the Newton-Krylov method. Although SU2 was able to reach convergence using the standard solver, the Newton-Krylov solver showed stalling behaviour for the ETH test case as well.

The stalling behaviour of the MTU and ETH test cases did not allow for the analysis of the Newton-Krylov solver in SU2. Additional research has been conducted to investigate the reason for the observed behaviour, but this did not result in a solution. This thesis provides insight into the current strengths and weaknesses of SU2 with respect to turbomachinery analysis. It was found that the Newton-Krylov method in SU2 currently underperforms when compared to the NUTSCFD solver. In addition, it was also shown that SU2 provides similar results to CFX and Numeca with respect to axial turbomachinery analysis. Many recommendations are made with respect to the final outcome of the current work, where this thesis provides several opportunities for further research.

CONTENTS

List of Figures

List of Tables

NOMENCLATURE

PARAMETERS

Symbol	Description	Unit
$\boldsymbol{A}$	Matrix of coefficients	-
$\boldsymbol{A}_1$	First-order approximation of the coefficient matrix	-
$\boldsymbol{A}_2$	Second-order approximation of the coefficient matrix	-
$\boldsymbol{b}$	Column vector of constants (RHS)	-
e	Internal energy	J/kg
$\mathbf{e}$	Vector of the solution error	-
$\mathbf{e}_1$	Standard basis vector	-
$\mathbf{f}$	Specific net body forces	N/kg
$\mathbf{F}(\mathbf{W})$	Vector containing the governing functions	-
$\mathbf{F}_{visc}$	Vector containing the forces due to viscous effects	-
$\tilde{\boldsymbol{H}}$	Hessenberg matrix	-
$I_{k,k-1}$	Interpolation operator	-
$\boldsymbol{I}$	Identity matrix	-
k	Multigrid grid level	-
K_r, K_n	Krylov subspace	-
$\boldsymbol{L}$	Lower triangular matrix	-
$\boldsymbol{M}$	Preconditioner	-
n	Iteration	-
p	Pressure	Pa
$\dot{q}$	Rate of heat addition	W
$Q_{k,k-1}$	Transfer operator	-
r	Krylov iteration	-
$\mathbf{r}$	Residual vector	-
$\mathbf{R}(\mathbf{W})$	Steady residual vector	-
$\mathbf{R}(\mathbf{W}^n)$	Steady residual vector of the approximate solution	-
$\mathbf{R}^{uns}$	Unsteady residual vector	-
t	Time	s
$T_{k,k-1}$	Transfer operator	-
$\boldsymbol{U}$	Upper triangular matrix	-
$\boldsymbol{v}_n$	Krylov subspace vector at iteration	-
V	Velocity	m/s
$\mathbf{V}$	Vector of flow velocities	-
$\boldsymbol{V}_r, \boldsymbol{V}_n$	Orthonormal basis vector of the Krylov subspace at iteration	-
$\mathbf{W}, \mathbf{W}_c, \mathbf{w}$	Vector of conservative variables	-
$\mathbf{W}_s$	Solution vector of the conservative variables	-
$\mathbf{W}^n$	Approximation of the solution vector at iteration n	-
$\boldsymbol{x}$	Column vector of variables	-
Δt	Size of timestep	s
$\Delta\mathbf{W}$	State perturbation	-

PARAMETERS (GREEK)

Symbol	Description	Unit
α	weight parameter	-
ρ	Density	kg/m^3

1

INTRODUCTION

Computational Fluid Dynamics, also known as CFD, plays an important factor in the field of aerodynamics. Fluid flows can be described using mathematical equations, but a direct solution to these equations is yet to be found. Although obtaining a direct solution is therefore not possible, the system of equations can be solved using iterative methods. These iterative methods estimate the solution of the system, where at every iteration the error of this solution is assessed. Following this principle, it is therefore possible to reach the solution of the system of equations, thus solving the flow problem up to a certain accuracy. These problems are generally solved using a computer which is very capable of solving these iterative methods. With the introduction of the computer, many advancements have been made in the field of numerical methods involving flow problems. Although the computer might be capable of reaching the solution of the system of equations, it is generally not desired due to its increased computational cost, as a more accurate solution requires additional computation iterations. It has therefore been accepted within the field of CFD that obtaining an accurate estimate of the true solution is sufficient when analysing a flow problem [5]. Manipulating the iterative method has therefore been a relevant topic in the field of CFD, as advancements improve the convergence behaviour of the solver. Over the years this has led to the development of several iterative methods, which improved the solvers' convergence rate and stability [6]. Many different software solutions exist which make use of these numerical methods in order to simulate flow problems. One of these solutions is the open-source software called SU2, which recently improved its turbomachinery capabilities. This update should improve the turbomachinery module within SU2, but it is currently unknown to what extent. A numerical study is therefore required in order to assess SU2's current performance with respect to turbomachinery flow simulations. In addition, novel numerical methods recently studied by Xu et al. [1] showed an increase in solver performance specifically for turbomachinery flow simulations. Xu et al. developed a CFD solver, called NUTSCFD, which makes use of a Newton-Krylov method in order to solve the highly nonlinear system of flow equations. This method uses Newton iterations in order to linearize the system of nonlinear equations, which are subsequently solved by a linear generalized minimal residual (GMRES) solver. This novel method is also available within SU2, where it is currently unknown how the new turbomachinery module interacts with the Newton-Krylov solver. The current work therefore provides a numerical assessment of SU2's performance and that of the Newton-Krylov solver within SU2, using turbomachinery flow problems. The SU2 version used for this is **version 7.5** (specifically the 'restuct_singlezone' git branch).

1.1. MOTIVATION

Turbomachinery aerodynamics can be characterized by complex flow structures that are formed due to the interaction between the rotating machinery and the fluid. This generally involves complex structures which could include three-dimensional flows and other phenomena such as shock waves. These complex systems require a robust solver in order to reach convergence efficiently. Although the computational power of computers has increased significantly over the years, developing and researching efficient iterative methods is still very relevant. New design paradigms have emerged which use automatic design optimization, allowing the computer to change designs based on simulation results. This can improve the performance and efficiency of a design, but may require extensive computation power as running multiple CFD simulations is required. Inefficient and diverging simulations are therefore especially problematic, as this could have a

significant negative effect on the final result. The turbomachinery CFD solver part of SU2 has received an update where it is developed at the Power & Propulsion research group at the TU Delft. The updated turbomachinery module should improve its solver convergence and efficiency but requires a numerical study as its current performance is unknown. The goal of the thesis is therefore to perform a thorough assessment of the performance of the SU2 CFD software for turbomachinery flow analysis. In addition, the Newton-Krylov solver currently implemented in SU2 is tested as well, as Xu et al. showed promising solver performance involving turbomachinery. Xu et al. were able to increase the robustness of their solver, while also reducing the number of iterations required in order to reach convergence. Xu et al. dedicate their performance increase to be the result of the Newton-Krylov method that is implemented in their NUTSCFD solver. SU2 already includes a Newton-Krylov solver, but its performance with respect to turbomachinery analysis is also unknown. A performance assessment therefore has to be done in order to fully investigate the efficiency of this method as well. The analysis of SU2's performance with respect to turbomachinery analysis provides insight into SU2's current behaviour, which should identify its strengths and weaknesses. This thesis will therefore contribute to the further development of the SU2 CFD solver that is used in order to obtain state-of-the-art turbomachinery analysis results.

1.2. RESEARCH GOALS AND SCOPE

The objective of the research is to investigate SU2's performance with respect to turbomachinery flow analysis. This is achieved by comparing SU2's results to that of the scientific field. SU2's performance will be tested by using CFD results obtained by the CFD solvers CFX, Numeca and NUTSCFD, where additional experimental data is compared as well. Following these objectives, several research questions can be formulated which should be answered to be able to come to a comprehensive conclusion. The first research question that can be proposed, being the primary research question, is formulated as follows:

1. **To what extent can SU2 be used in order to analyse turbomachinery flow problems?** The primary research question regards to the assessment of SU2 and its turbomachinery capabilities. It is currently unknown to what extent SU2 is able to provide accurate results with respect to turbomachinery flow analysis. A numerical study is therefore to be conducted in order to analyse its current performance. In order to support the primary research question, several sub-questions can be posed with respect to the primary research question.

   (a) **To what extent does SU2 provide results that coincide with other CFD solvers involving turbomachinery flow simulations?** SU2's performance involving turbomachinery simulations is currently unknown. SU2 should provide results that are similar to that of other CFD solvers, but this is to be confirmed. Verifying the accuracy of SU2 using other CFD solvers should provide valuable insight into its current performance, allowing for the further development of SU2.

   (b) **To what extent does the Newton-Krylov solver change the performance of SU2 with respect to turbomachinery flow simulations?** A Newton-Krylov solver has been implemented in SU2, where its performance involving turbomachinery flow simulations is to be analysed. This requires an analysis of the Newton-Krylov method both involving normal and turbomachinery simulations. Analyzing the difference in performance should provide a greater assessment of the performance of the Newton-Krylov method in SU2. The performance of SU2 can be assessed by using the residual of the solver and the number of iterations required in order to reach convergence.

   (c) **To what extent does the performance of the Newton-Krylov solver in SU2 differ from the NUTSCFD solver with respect to turbomachinery applications?** Studies conducted by Xu et al. [1] showed a significant increase in convergence performance, especially with respect to turbomachinery simulations. Comparing SU2 to the NUTSCFD solver should therefore provide an indication of SU2's current performance with respect to other research in the field.

   After analysing the current performance of SU2 and the implementation of the Newton-Krylov solver, additional research is required which should help with improving the convergence efficiency of SU2. Another research question can therefore be defined with respect to this topic:

2. **What are potential areas of the SU2 code that could be improved in order to increase the Newton-Krylov solver performance with respect to turbomachinery applications?** This research question

involves the identification of areas of interest with respect to the SU2 codebase that are relevant for increasing its convergence performance. The primary research question focuses on investigating SU2's performance using comparative analysis, where the secondary research question focuses on how this performance might be improved.

(a) **What algorithms and settings in SU2 can be identified which have a large effect on the rate of convergence of the Newton-Krylov solver?** Previous literature research conducted on Newton-Krylov solvers should provide an overview of the areas which play a crucial role in the performance of the solver. By comparing the current SU2 code to these relevant algorithms, areas of interest can be identified within the SU2 codebase which might be improved.

(b) **To what extent does changing the relevant algorithms and settings which affect the Newton-Krylov solver within SU2 have an effect on the convergence rate?** Investigating the difference in convergence behaviour with respect to different solver settings can give an indication of algorithms that play a crucial role in the convergence efficiency. Investigating these settings could discover hidden performance of SU2.

Following the defined questions, the primary research question aims to determine the current performance of SU2, while the secondary research question focuses on how this performance might be improved using the Newton-Krylov method. The thesis will therefore set out to develop different test cases which are to test SU2's performance with respect to turbomachinery. These results are compared to the results following CFX, Numeca, the NUTSCFD solver, and experimental data, where additional research is conducted with respect to the solver framework of SU2. The scope of the thesis will therefore focus on the performance of SU2 with respect to turbomachinery. Both axial and radial flow machines are to be tested, where the number of stages is limited in order to obtain results efficiently. The thesis will focus mainly on internal flows, where a singular test case is used to test SU2's external flow performance. The codebase of SU2 is analysed as well, where suggestions will be made with respect to relevant algorithms that might require further research. The relevant algorithms that influence the Newton-Krylov method are analysed and explained, but no new code additions are made within the current work with respect to SU2. This is considered to be outside the scope of this thesis.

1.3. THESIS OUTLINE

The theoretical background supporting the thesis can be found in chapter 2. In this chapter relevant concepts involving SU2 and the Newton-Krylov solver are discussed. The governing equations are derived in terms of the residual, where the linearization following Newton's method is explained. After that, the GMRES linear solver is described which is used in order to solve the linear system obtained from Newton's Method. Both SU2 and the challenges found for turbomachinery problems are discussed, where the chapter is concluded by an evaluation of additional relevant solving methods. Chapter 3 discusses the development of the test cases that have been set up in order to test SU2's current performance. A detailed explanation is given following the creation of these test cases, where the chapter also contains a discussion evaluating the validity of the configurations. The results obtained using these test cases are shown in chapter 4, where these results are interpreted and compared to the reference literature. The validity of these results is discussed, where the observed data is commented upon. Based on the research conducted within this thesis, answers are provided to the research questions in chapter 5. Finally, topics of interest that can be considered for further research are discussed in chapter 6. These topics are obtained resulting from the conclusions found within this thesis, where a motivation is provided with respect to their relevance to the research field.

2

THEORETICAL BACKGROUND

This chapter contains the theoretical background which supports the work done within this thesis. The literature review therefore provides an overview of the advancements and challenges in the field of Computational Fluid Dynamics. The focus of the thesis is to investigate SU2's turbomachinery performance and its Newton-Krylov solver. SU2 is therefore compared to the solver developed by Xu et al. [1], and the CFD solvers CFX and Numeca. Xu et al. have developed a steady computational fluid dynamics solver based on the Reynolds-averaged Navier-Stokes equations. RANS solvers provide good accuracy of the flow at low computational costs, but behave relatively poorly when dealing with challenging off-design conditions [7]. This is especially relevant with respect to turbomachinery since turbomachines operating at off-design conditions create complex flow structures. As mentioned before, it is therefore desirable to develop robust and efficient solvers when analysing turbomachinery problems [8]. This thesis includes an analysis of the methods that are used within the NUTSCFD solver, where this chapter provides the theoretical background for these methods. Section 2.1 provides the basis on which these methods are built, explaining the derivation of the residual from the governing flow equations. This section is followed by section 2.2 which explains how the obtained non-linear system of equations is solved using Newton's method. The procedure that is used in order to solve the linear system of equations obtained from Newton's method is explained in section 2.3. This section is followed by section 2.4, which provides an analysis of SU2's framework and the challenges commonly encountered when analysing turbomachinery problems. The chapter is concluded with section 2.5 which analyses additional relevant methods which could be considered in order to improve SU2's performance.

2.1. FUNDAMENTALS OF CFD

The current thesis conducts solver performance analysis by simulating steady turbomachinery problems. These turbomachinery problems are simulated by solving the flow equations that represent these problems. The current section contains an explanation of the governing flow equations, where the residual equation is derived as well. The residual is fundamental when simulating flow problems as it provides an indication of the performance of the solver and the accuracy of the calculated solution. This section therefore starts with the definition of the governing equations (section 2.1.1), followed by the derivation of the steady residual (section 2.1.2) and is concluded with the derivation of the unsteady residual (section 2.1.3).

2.1.1. GOVERNING EQUATIONS

Within fluid mechanics the governing equations for fluid motion follow from the conservation equations. These equations can be derived from the Navier-Stokes equations, which results in the equations for conservation of mass, momentum and energy. The differential form of these equations is defined following equation 2.1.

$$\frac{\partial \mathbf{W}_c}{\partial t} + \nabla \cdot (\mathbf{W}_c \mathbf{V}) = -\nabla \cdot \mathbf{F}_p + \rho\,(\mathbf{F}_a) + \mathbf{F}_{visc}$$
$$with\ \ \mathbf{W}_c = \begin{pmatrix} \rho \\ \rho \mathbf{V} \\ E_t \end{pmatrix},\ \ \mathbf{F}_p = \begin{pmatrix} 0 \\ p \\ p \cdot \mathbf{V} \end{pmatrix},\ \ \mathbf{F}_a = \begin{pmatrix} 0 \\ \mathbf{f} \\ \dot{q} + \mathbf{f} \cdot \mathbf{V} \end{pmatrix},\ \ E_t = \rho\,(e + \frac{V^2}{2}) \tag{2.1}$$

Within equation 2.1 $\mathbf{V}$ is the vector of the flow velocities, p the pressure, $\mathbf{f}$ the specific net body forces, $\dot{q}$ the rate of heat addition and e the internal energy. In addition $\mathbf{F}_{visc}$ represents the forces and rate of work done on the fluid due to viscous effects. A more detailed derivation of the conservation equations is given by Anderson [5], which is beyond the scope of this thesis. Equation 2.1 is a system of non linear equations which is used in order to model and simulate flow behaviour. Solving this system of equations is therefore essential when solving flow problems. At the time of writing no analytical solution for the nonlinear conservation equations has been obtained, where generally an approximate solution is calculated instead. One way of solving this system of equations is by making use of Computational Fluid Dynamics. The governing flow equations are discretized, where a grid of discrete points is obtained. This results in a system of equations relating the different flow field properties at each grid point to the other grid points. By solving these relations a web of points is obtained which indicate the behaviour of the flow at each point. By increasing the number of points, the 'resolution' of the flow is increased, where a more accurate result is obtained. Increasing the number of points to infinity would therefore result in an approximate solution which approaches the analytical solution. Increasing the resolution to infinity is however often not required nor computationally viable, where a solution with a small error can provide sufficiently deemed results.

The principle of solving the nonlinear flow equations using discretization is the basis for CFD. Different techniques are developed over the years where generally an iterative method is used in order to obtain the solution. These methods calculate a solution iteratively, by refining an approximate solution. Every iteration a new estimation of the final solution is done, where the accuracy of this solution is determined. If the solution is sufficiently accurate the solver is regarded to be converged. Generally CFD problems are solved iteratively, where an estimated guess is done with respect to the solution. This is an iterative procedure where for every iteration the solution is assessed. In order to determine whether the solver has converged, a measure is required which is able to test the estimate solution. Within CFD this is done by calculating the residual of the solution, which determines the error between the estimate solution and the real solution. The average of the residual over all equations of all control volumes in the computational domain is then used as an indicator whether the iterative solver has converged for a given problem. In essence the residual is the difference between the left and right side of the problem, also called the solution dis-balance. It should be noted that within the current work the RANS equations are used in order to simulate the flow problems. The Reynolds-Averaged Navier-Stokes equations use the concept of mean fluid flow variables together with fluctuating components in order to model the flow behaviour. Full derivation of these equations is outside the scope of the current work.

2.1.2. THE STEADY RESIDUAL

From section 2.1.1 it can be concluded that the governing nonlinear system of equations (equation 2.1) could potentially be solved using an iterative method. In order to dictate whether the problem has converged to a solution, a method is required which is able to assess the accuracy of the solution at the given iteration. This is done using the residual. Let the nonlinear flow equations mentioned in section 2.1.1 be approximated using equation 2.2, where equation 2.2 involves an unsteady flow problem. Within equation 2.2 $\boldsymbol{F}(\boldsymbol{W})$ is a vector containing the governing functions, and $\boldsymbol{W}$ represents the vector containing the unknown flow variables $\boldsymbol{W} = (\rho, \rho\boldsymbol{V}, E_t)^T$.

$$\frac{\partial \mathbf{W}}{\partial t} + \boldsymbol{F}(\boldsymbol{W}) = 0 \tag{2.2}$$

Solving equation 2.2 with $\frac{\partial \mathbf{W}}{\partial t} = 0$ would therefore result in the steady solution $\boldsymbol{W}_s$. In order to obtain solution $\boldsymbol{W}_s$, let $\boldsymbol{W}^n$ be the computed approximation of the solution at given iteration n using an iterative method. By comparing both $\boldsymbol{W}_s$ and $\boldsymbol{W}^n$ an indication can be obtained of the current error of the current approximated solution (equation 2.3).

$$\boldsymbol{e} = \boldsymbol{W}_s - \boldsymbol{W}^n \tag{2.3}$$

Since solution $\boldsymbol{W}_s$ is unknown, it is however not possible to compute the error $\boldsymbol{e}$. $\boldsymbol{W}_s$ and $\boldsymbol{W}^n$ can however be related using the vector of the governing functions, giving the residual $\boldsymbol{r}$ (equation 2.4). By substituting the steady formulation of equation 2.2 into equation 2.4, the steady residual can be obtained where it is independent of solution $\boldsymbol{W}_s$ (equation 2.5).

$$\boldsymbol{r} = \boldsymbol{F}(\boldsymbol{W}_s) - \boldsymbol{F}(\boldsymbol{W}^n) \tag{2.4}$$

$$\boldsymbol{R}(\boldsymbol{W}^n) = 0 - \boldsymbol{F}(\boldsymbol{W}^n) \tag{2.5}$$

Since both the error $\boldsymbol{e}$ and the residual $\boldsymbol{r}$ represent the comparison between the current approximation and final solution, the iterative method converges if $\boldsymbol{r}$ is equal to 0. This is due to the comparative nature of both such that if $\boldsymbol{r}$ is equal to 0, $\boldsymbol{e}$ has to be equal to 0 [9]. Rewriting equation 2.5 finally results in the steady residual at every iteration n following equation 2.6.

$$\boldsymbol{R}(\boldsymbol{W}^n) = -\boldsymbol{F}(\boldsymbol{W}^n) = 0 \tag{2.6}$$

2.1.3. THE UNSTEADY RESIDUAL

The residual mentioned in section 2.1.2 involves the derivation of the residual for a steady flow problem. Kamenetskiy et al. [10] indicate the relevance of using an unsteady residual as opposed to a steady residual when solving for a steady-state solutions. By using the unsteady residual in combination with Newton's method, the nonlinear iterations are able to get around local extrema, improving the stability of the solver. Since the steady residual is defined following equation 2.6 and since $\boldsymbol{F}(\boldsymbol{W})$ represents the steady governing equations, the unsteady governing equations could be rewritten to the unsteady residual following equation 2.7. Within equation 2.7 $\boldsymbol{R(W)}$ represents the steady residual and $\frac{\partial \boldsymbol{W}}{\partial t}$ the time derivative of the flow variables.

$$\boldsymbol{R}^{uns} \equiv \frac{\partial \boldsymbol{W}}{\partial t} + \boldsymbol{R}(\boldsymbol{W}) = \boldsymbol{0} \tag{2.7}$$

Finding the steady solution for the unsteady residual using an iterative method requires the discretization of equation 2.7. Equation 2.8 can be derived by discretizing equation 2.7 implicitly. Within equation 2.8 Δt is the size of the timestep and $\boldsymbol{W}^n$ is the estimate flow solution at timestep n.

$$\boldsymbol{R}^{uns} \equiv \frac{\boldsymbol{W}^{n+1} - \boldsymbol{W}^n}{\Delta t} + \boldsymbol{R}(\boldsymbol{W}^{n+1}) = \boldsymbol{0} \tag{2.8}$$

In order to solve equation 2.8, the governing equation can be rewritten to a linear system of equations. This can be done by creating a linear approximation of the nonlinear term $\boldsymbol{R}(\boldsymbol{W})$ using the first-order Taylor polynomial (equation 2.9).

$$\boldsymbol{R}(\boldsymbol{W}^{n+1}) \approx \boldsymbol{R}(\boldsymbol{W}^n) + \frac{d\boldsymbol{R}(\boldsymbol{W}^n)}{d\boldsymbol{W}^n}(\boldsymbol{W}^{n+1} - \boldsymbol{W}^n) \tag{2.9}$$

By substituting equation 2.9 into equation 2.8, the unsteady residual can be rewritten to the standard matrix-vector notation for a linear system of equations. By defining $\Delta \boldsymbol{W} = \boldsymbol{W}^{n+1} - \boldsymbol{W}^n$ the governing equation becomes equation 2.10, which can finally be rewritten to equation 2.11.

$$\boldsymbol{R}^{uns} \equiv \frac{\Delta \boldsymbol{W}}{\Delta t} + \boldsymbol{R}(\boldsymbol{W}^n) + \frac{\partial \boldsymbol{R}(\boldsymbol{W}^n)}{\partial \boldsymbol{W}^n}\Delta \boldsymbol{W} = 0 \tag{2.10}$$

$$\boldsymbol{R}^{uns} \equiv \left(\frac{\boldsymbol{I}}{\Delta t} + \frac{\partial \boldsymbol{R}(\boldsymbol{W}^n)}{\partial \boldsymbol{W}^n}\right)\Delta \boldsymbol{W} = -\boldsymbol{R}(\boldsymbol{W}^n) \tag{2.11}$$

Within equation 2.11, $\boldsymbol{I}$ is the identity matrix and $\boldsymbol{R}(\boldsymbol{W}^n)$ is the steady residual which is the flux term. Equation 2.11 represents the linear approximation of the temporal discretized unsteady residual. The form of equation 2.11 is analogous to that of a linear matrix problem $\boldsymbol{Ax} = \boldsymbol{b}$, where the system could therefore be solved using matrix inversion. Solution $\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$ is for example calculated by obtaining the inverse of matrix $\boldsymbol{A}$. Solving the linear system of equations following matrix inversion is however only to be considered for small systems, as this would otherwise be computationally unviable due to time and memory constraints. Since typically flow simulations involve large systems of equations due to grid sizing, it is therefore not possible to solve the system following matrix inversion. Instead iterative solvers such as GMRES are used, which solve the linear system of equations by calculating an approximate solution.

2.2. SOLVING THE NONLINEAR FLOW PROBLEM

Section 2.1.3 resulted in the reformulation of the governing equations into the unsteady residual of the flow problem. Equation 2.11 can be rewritten to $\boldsymbol{P}\Delta\boldsymbol{W} = -\boldsymbol{R}(\boldsymbol{W}^n)$, where the solution is found if the nonlinear equations $\boldsymbol{R}(\boldsymbol{W}) = 0$. The procedure of approximating the nonlinear residual using the first-order Taylor polynomial and solving for 0 iteratively is called the Newton-Raphson method. In order to solve the nonlinear equation $\boldsymbol{R}(\boldsymbol{W})$ Newton's method uses consecutive guesses which are calculated using iterations. Every iteration a new guess is done based on the previous iteration, resulting in the approximate solution to the nonlinear equations when using sufficient amount of iterations. This method does not solve the system directly, where therefore the exact solution is not found. Instead the residual of the final guess is determined which should meet the predetermined imposed accuracy of the solution. An example of Newton's method can be seen in equation 2.12. In order to solve for x in $f(x) = 0$, $f(x)$ is rewritten resulting in the iterative method. An illustration of this method can be seen in figure 2.1, using an arbitrary function $f(x)$. Within figure 2.1 the dashed lines represent the tangent at every step n. It is shown that using a sufficient number of iterations n would result in x being equal to x^*, with x^* being the root of $f(x)$, thus being the solution to $f(x) = 0$.

$$f(x) = 0 \Rightarrow x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{2.12}$$

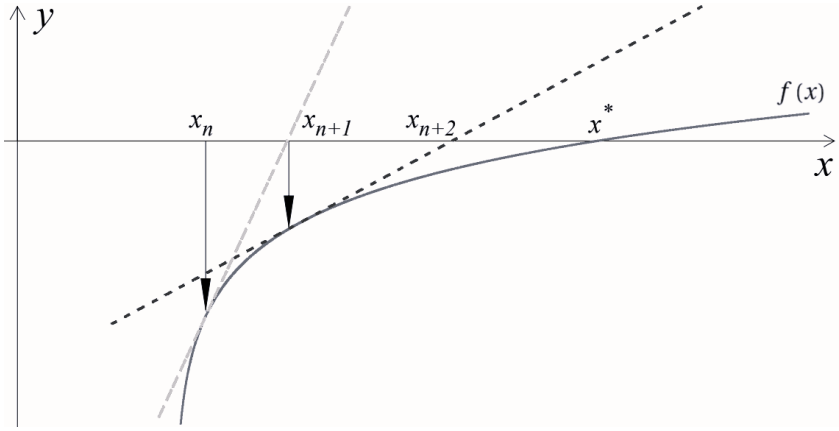


Figure 2.1: Example of Newton's method for function $f(x)$ [2]

Newton's method can therefore provide a sufficiently accurate result to a nonlinear equation while not solving the problem directly. It is often not necessary for the final solution to be equal to x^*, as this level of accuracy is often considered to be trivial. This core principle is especially relevant when using complex nonlinear systems that are difficult to solve, such as systems involving the nonlinear flow models. In essence equation 2.12 and equation 2.11 are the same problem. When starting the solving procedure of the nonlinear system, an initial guess $\boldsymbol{W}^0$ is used which results in the linear problem $\boldsymbol{P}\Delta\boldsymbol{W} = -\boldsymbol{R}(\boldsymbol{W}^0)$. This linear problem is to be solved in order to determine $\Delta\boldsymbol{W}$ and thus $\boldsymbol{W}^1$. Solving for $\boldsymbol{W}^1$ results in one Newton iteration, where for the next iteration $\boldsymbol{W}^1$ is used as the initial guess. Newton's method therefore results in a linear problem which is to be solved in order to do a single Newton iteration. It has been shown that Newton's method can provide a successful strategy for solving complex nonlinear systems by creating a linear problem. Different methods can be considered to solve the nonlinear problem, but Newton's method has been chosen due to its strong error-damping capability compared to other schemes [2]. The linear problem that is obtained using Newton's method is to be solved using a linear solver.

2.3. SOLVING THE LINEAR SYSTEM OF EQUATIONS

Following section 2.2 a linear system of equations is obtained which should be solved in order to complete the Newton iteration. Different linear solvers are available, where within the current work the generalized minimal residual (GMRES) solver is analysed. The combination of Newton's method with GMRES results in the Newton-Krylov solver, which is the main subject of interest within this thesis. Other solvers such as BiCGSTAB use the Krylov subspace as well, but are not analysed due to the reference paper using GMRES.

This section first starts with an explanation of the GMRES algorithm in section 2.3.1 and is followed by an explanation of the FGMRES algorithm (section 2.3.2) which is currently available in SU2.

2.3.1. GMRES

From section 2.2 a large sparse linear system of equations is obtained which is derived from the discretized governing equations. This linear system is to be solved in order to solve the nonlinear flow problem. Different methods can be considered, where for the current study the generalized minimal residual is analyzed. Generalized minimal residual (GMRES) is a residual-driven method where the solver focuses on minimizing the residual vector. GMRES is an iterative method where for the current study every (outer) nonlinear iteration functions as the starting conditions for the linear GMRES solver. The combination of both iterative methods therefore results in a levelled system of inner (GMRES) and outer (Newton) iterations.

GMRES [11] is a generalized version of the MINRES algorithm, which focuses on solving nonsymmetric linear systems. By using Arnoldi's method, the GMRES algorithm is able to construct an l_2-orthogonal basis of the Krylov subspaces. These Krylov subspaces are then used to obtain the approximate solution to the linear system. As with the Newton iterations, it is generally accepted for the system to have a small error, thus not requiring the direct solution to the linear system. Although it is possible to solve the linear system to a certain extent using GMRES, an error tolerance is specified allowing the solver to converge to an approximate solution. Converging to an approximation therefore reduces the total number of GMRES iterations, saving computational power and time.

KRYLOV SUBSPACE

The GMRES algorithm can be explained by evaluating the requirements needed in order to reach convergence. Since GMRES is an iterative method and given its convergence characteristics, an approximation of the solution should be derived (using some variables) such that it is able to meet the GMRES convergence criteria. This approximation has to be calculated for every iteration until the approximate solution is reached. A strategy for obtaining this iterative approximation is by using the coefficient matrix and the right-hand side vector of the linear problem $\boldsymbol{Ax} = \boldsymbol{b}$. Calculating the approximate solution vector using this iterative procedure would therefore result in a span of vectors, which combined give a vector subspace. In theory, with enough iterations, this subspace of vectors could contain a solution vector which approaches the solution to the linear problem. One way of obtaining this theoretical subspace is by generating the Krylov subspace (equation 2.13), where the right-hand side vector $\boldsymbol{b}$ is multiplied with the coefficient matrix $\boldsymbol{A}$. This subspace is expanded by multiplying the last vector of the subspace with the coefficient matrix $\boldsymbol{A}$. The size of the subspace is therefore determined by the number of matrix multiplications done in order to create the Krylov subspace.

$$K_r(\boldsymbol{A}, \boldsymbol{b}) = \text{span}\left\{\boldsymbol{b}, \boldsymbol{Ab}, \boldsymbol{A}^2\boldsymbol{b}, \dots, \boldsymbol{A}^{r-1}\boldsymbol{b}\right\} \tag{2.13}$$

An algorithm can be used in order to generate the Krylov subspace, where for the current study this algorithm is Arnoldi's method. Arnoldi's method is an eigenvalue algorithm, where it is able to provide an approximation of the eigenvalues and eigenvectors to the linear system [11]. Arnoldi's method also provides the orthonormal basis of the Krylov subspace. Creating the orthonormal basis of the subspace is desired as this improves the robustness of the solver by improving the accuracy of the approximate solution. This is achieved by projecting every newly computed Krylov vector onto each of the previously computed Krylov vectors, subtracting the projections from the new Krylov vector. In addition to the Krylov subspace, Arnodli's method also provides the upper Hessenberg matrix of the linear system. This matrix has the dimensions of $r+1$ by r, where r represents the number of Krylov iterations and thus the size of the Krylov subspace. The Hessenberg matrix is computed using the Krylov products, where it is obtained by projecting the new Krylov vector upon the Krylov subspace of the previous iterations. The Hessenberg matrix is therefore a projection of the original coefficient matrix $\boldsymbol{A}$ upon the orthonormal Krylov subspace [12], providing the approximate eigenvalues to the coefficient matrix. The coefficient matrix $\boldsymbol{A}$ can be related to the Hessenberg matrix following equation 2.14, with $\bar{\boldsymbol{H}}$ being the Hessenberg matrix and $\boldsymbol{V}_r$ being the orthonormal basis vector of the Krylov subspace.

$$\boldsymbol{AV}_r = \boldsymbol{V}_{r+1}\bar{\boldsymbol{H}}_r \tag{2.14}$$

LEAST SQUARES PROBLEM

As mentioned before, GMRES is a residual-driven method where a residual has to be defined in order to determine whether the iterative algorithm has converged to a solution. Similar to section 2.1.2 the residual for the linear problem can be derived resulting in $\boldsymbol{R}(\boldsymbol{x}) = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}$. Solving $\boldsymbol{R}(\boldsymbol{x})$ results in a minimization problem as the main goal of the solver is to find x for which the residual is the smallest. This minimization problem becomes a least squares problem when using the Euclidean norm on the residual (equation 2.15), as is done in order to improve stability.

$$\boldsymbol{R}(\boldsymbol{x}) = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x} \Leftrightarrow \|\boldsymbol{r}_n\| = \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_n\| \tag{2.15}$$

Instead of creating the Krylov subspace of the linear problem $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, it is also possible to setup the Krylov subspace in terms of the residual (equation 2.16). Within equation 2.16 $\boldsymbol{r}_0$ represents the initial error for $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$ where the initial guess for $\boldsymbol{x}_0$ is not the zero vector.

$$K_n(\boldsymbol{A}, \boldsymbol{r}_0) = \text{span}\left\{\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{r}_0, \boldsymbol{A}^2\boldsymbol{r}_0, \dots, \boldsymbol{A}^{n-1}\boldsymbol{r}_0\right\} \tag{2.16}$$

Since K_n is created using the linear residual equation, it can be stated that $\boldsymbol{x}_n \in K_n$. The goal of the least squares problem is to find $\boldsymbol{x}_n$, where $\boldsymbol{x}_n$ can also be defined in terms of the orthonormal basis of the Krylov subspace created by Arnoldi's method. This results in the formulation of a new problem: $\boldsymbol{x}_n = \boldsymbol{x}_0 + \boldsymbol{V}_n\boldsymbol{y}_n$, with $\boldsymbol{V}_n$ being the orthonormal basis of the Krylov subspace and $\boldsymbol{y}_n$ a new variable which should be determined in order to obtain $\boldsymbol{x}_n$. $\boldsymbol{V}_n$ is defined as $[\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_n]$, with $\boldsymbol{v}_1 = \boldsymbol{r}_0 / \|\boldsymbol{r}_0\|$ and $\boldsymbol{v}_n$ being calculated using Arnoldi's method. Substituting this into equation 2.15 results in equation 2.17.

$$\|\boldsymbol{r}_n\| = \left\|\boldsymbol{b} - \boldsymbol{A}(\boldsymbol{x}_0 + \boldsymbol{V}_n\boldsymbol{y}_n)\right\| = \left\|\boldsymbol{r}_0 - \boldsymbol{A}\boldsymbol{V}_n\boldsymbol{y}_n\right\| = \left\|\|\boldsymbol{r}_0\|\,\boldsymbol{v}_1 - \boldsymbol{A}\boldsymbol{V}_n\boldsymbol{y}_n\right\| \tag{2.17}$$

As mentioned before, the Hessenberg matrix could be used instead of the original coefficient matrix in order to obtain an approximate solution. By defining $\beta = \|\boldsymbol{r}_0\|$ and by substituting the Hessenberg matrix $\tilde{\boldsymbol{H}}$ (equation 2.14) into equation 2.17, equation 2.18 is obtained.

$$\|\boldsymbol{r}_n\| = \left\|\beta\boldsymbol{v}_1 - \boldsymbol{V}_{n+1}\tilde{\boldsymbol{H}}_n\boldsymbol{y}_n\right\| \tag{2.18}$$

Equation 2.18 can be rewritten further by making use of the standard basis vector $\boldsymbol{e}_1$. The standard basis vector is defined as $\boldsymbol{e}_1 = [1, 0, \dots, 0]^T$ where together with the orthonormal basis $\boldsymbol{V_n}$ it follows that $\boldsymbol{V_n} \cdot \boldsymbol{e}_1 = \boldsymbol{v}_1$. Since the norm of the orthonormal basis is equal to 1, equation 2.18 can be rewritten to equation 2.19.

$$\|\boldsymbol{r}_n\| = \left\|\boldsymbol{V}_{n+1}(\beta\boldsymbol{e}_1 - \tilde{\boldsymbol{H}}_n\boldsymbol{y}_n)\right\| = \left\|\beta\boldsymbol{e}_1 - \tilde{\boldsymbol{H}}_n\boldsymbol{y}_n\right\| \tag{2.19}$$

Equation 2.19 is to be used in order to find x_n. Minimizing equation 2.19 results in a value for $\boldsymbol{y}_n$, which gives x_n following $\boldsymbol{x}_n = \boldsymbol{x}_0 + \boldsymbol{V}_n\boldsymbol{y}_n$. The derivation described within the current section allows for the residual to be minimized using the Hessenberg matrix, as is typically done within the GMRES method due to Arnoldi's method being required. The resultant method is a method which is limited by the complexity of the matrix-vector product $\boldsymbol{A} \cdot \boldsymbol{v}_n$ which is much more computationally efficient compared to the direct methods used in order to solve a linear system. The resultant GMRES iterative algorithm can be described as follows:

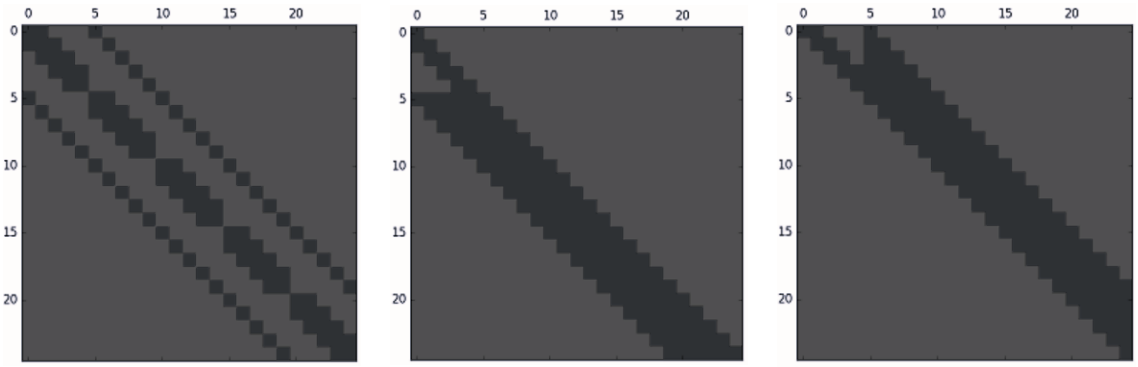
1. Calculate the new orthonormal vector $\boldsymbol{v}_n$ using Arnoldi's method
2. Find the value for $\boldsymbol{y}_n$ by minimizing equation 2.19
3. Calculate the new approximate solution vector using $\boldsymbol{x}_n = \boldsymbol{x}_0 + \boldsymbol{V}_n\boldsymbol{y}_n$
4. Reiterate the procedure if the residual does not meet the criteria

When minimizing equation 2.19, solution $\boldsymbol{x}_n$ provides an approximate solution to the linear problem. The solver has converged if the residual meets the solution criteria. If this is not the case, a new Krylov vector is calculated and the subsequent residual is tested until convergence of the linear system to the approximate solution is reached. A key advantage of GMRES is that it uses matrix-vector multiplication rather than matrix-matrix operations. This reduces the computational cost needed for solving the linear system of equations. Since for CFD the coefficient matrix can become large and sparse, the effect of using matrix-vector multiplications can become rather significant. The efficiency of GMRES is however linked to the number of iterations, as every iteration increases the size of the matrices and the number of vector multiplications needed in order to reach convergence. The performance of the solver can be further improved by transforming the linear system of equations by considering the usage of a preconditioner.

2.3.2. PRECONDITIONED GMRES

Computational costs can be further reduced by influencing the sparsity of the coefficient matrix that is used for the GMRES algorithm. A preconditioner is therefore used in combination with the GMRES algorithm in order to improve the convergence of the linear system. A preconditioner transforms the linear problem into an equivalent system that is easier to solve. Saad [13] showed the relevance of using a preconditioner in combination with GMRES, where he developed the Flexible GMRES algorithm (FGMRES). This algorithm uses GMRES in combination with a changing preconditioner, which reduced the number of iterations needed in order to reach convergence. It should be noted that the NUTSCFD solver developed by Xu et al. uses an Incomplete LU factorization (ILU) preconditioner, but instead of a variable preconditioner they used a fixed right preconditioner based on the Jacobian matrix.

Incomplete LU factorization approximates the coefficient matrix of a linear system into two triangular matrices such that $\boldsymbol{A} \approx \boldsymbol{LU}$. $\boldsymbol{U}$ and $\boldsymbol{L}$ are the upper and lower triangular matrix respectively, which are obtained following LU-decomposition given by $\boldsymbol{A} = \boldsymbol{LU}$. Since $\boldsymbol{LU}$ could approximate $\boldsymbol{A}$, it is possible to alter these matrices such that it benefits the solver. This would prevent the solver to calculate the exact solution to $\boldsymbol{Ax} = \mathbf{b}$, but it has been stated before that this is not always necessary. Xu et al. conducted several studies which showed that using an incomplete version of the decomposition matrices $\boldsymbol{L}$ and $\boldsymbol{U}$ is a highly effective method of preconditioning GMRES [14]. It was however shown that results may vary depending on the setup of the solver, indicating that choosing the right ILU preconditioner is problem specific. Xu et al. showed high performance following an incomplete LU preconditioner with a fill-in factor of 0. The fill-in factor is used in order to manage the sparsity of the decomposition matrices. The effect of the fill-in factor on the ILU preconditioner can be explained using figures 2.2 and 2.3.



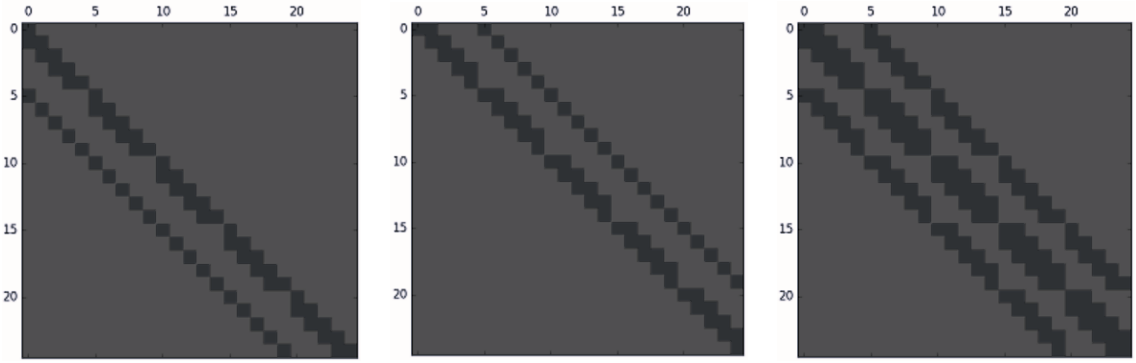
(a) Sparsity pattern of matrix $\boldsymbol{A}$ (b) Sparsity pattern of matrix $\boldsymbol{L}$ (c) Sparsity pattern of matrix $\boldsymbol{U}$

Figure 2.2: Sparsity pattern of matrices following $\boldsymbol{A} = \boldsymbol{LU}$

Let $\boldsymbol{A}$ be a sparse matrix, where figure 2.2a shows the structure of the matrix. The sparsity pattern of the matrix is indicated by the black elements, where the grey elements represent the zeros. Since $\boldsymbol{A} = \mathbf{LU}$, $\mathbf{L}$ and $\mathbf{U}$ can be decompositioned into figures 2.2b and 2.2c, where the fill-in factor will change the sparsity structure of these matrices. A fill-in factor of 0 uses the sparsity pattern of the original coefficient matrix $\boldsymbol{A}$ and applies it to matrices $\mathbf{L}$ and $\mathbf{U}$, such that if entry $\mathbf{A}_{i,j}$ is zero, entry $\mathbf{L}_{i,j}$ and $\mathbf{U}_{i,j}$ are zero. This results in figures 2.3a and 2.3b, which represent the matrices $\mathbf{L}_0$ and $\mathbf{U}_0$. By using the fill-in factor, the preconditioner can be expressed following $\boldsymbol{A} \approx \boldsymbol{A}_0 = \mathbf{L}_0\mathbf{U}_0 = \boldsymbol{M}$. For the sake of completeness, a higher fill-in factor uses the sparsity pattern of the product of the previous decomposition matrices $\mathbf{A}_{FF-1}$, where FF is the current fill-in factor. By using right preconditioning, the linear system of equations can be rewritten into equation 2.20.

$$\mathbf{AM}^{-1}(\mathbf{Mx}) = \mathbf{b} \tag{2.20}$$

Instead of using the standard formulation for a linear system of equations, the derivation involving GMRES (section 2.3.1) is applied on equation 2.20. Instead of using the coefficient matrix $\boldsymbol{A}$, the preconditioned coefficient matrix $\boldsymbol{AM}^{-1}$ is used, influencing both the Hessenberg matrix and the orthonormal basis. It is possible for this preconditioner to vary for every Arnoldi iteration, resulting in the FGMRES algorithm. It should be noted that currently SU2 uses FGMRES instead of GMRES. Detailed description for the FGMRES algorithm can be found in [13]. Advantages of FGMRES is that it could reduce the number of iterations that



(a) Sparsity pattern of matrix $\boldsymbol{L}_0$ (b) Sparsity pattern of matrix $\boldsymbol{U}_0$ (c) Sparsity pattern of matrix $\boldsymbol{A}_0$

Figure 2.3: Sparsity pattern of matrices $\boldsymbol{L}_0$ and $\boldsymbol{U}_0$ where $\boldsymbol{L}_0\boldsymbol{U}_0 = \boldsymbol{A}_0$

are required for the iterative method to find the solution to the linear system. FGMRES does however simultaneously introduce a preconditioning step which could be computationally expensive if not chosen correctly. Pueyo et al. [15] have found that a flexible preconditioner does not necessarily improve solver speed, as it is also possible that a fixed preconditioner does not affect convergence negatively. Using the correct fixed preconditioner can therefore reduce CPU costs, as the factorization of the preconditioner is only computed once for the linear problem.

2.4. CFD FRAMEWORK

The sections above described how flow problems can be solved mathematically using models. These models are used in Computation Fluid Dynamics, where the solution is obtained iteratively. The current section describes the frameworks that are used in order to realise these iterative solvers. CFD tools such as SU2 are used to simulate flow problems, where the current thesis is focused on Turbomachinery. Section 2.4.1 therefore contains a description of the challenges commonly found within Turbomachinery. The CFD software that is used and analysed within the current thesis is SU2, where therefore section 2.4.2 contains an explanation of the SU2 solver structure.

2.4.1. CFD AND TURBOMACHINERY

Solver robustness and solution accuracy are some of the major aspects that dictate the performance of a CFD code. These aspects are generally the center of attention when it comes to further development of the codebase. Since aerodynamic simulations can vary widely in conditions, it is often difficult to develop code which is able to perform optimally for every simulation setup. Simulations at design conditions for industrial applications can for example be characterized by favorable flow conditions such as largely attached flows. The combination of explicit and implicit solvers with multigrid acceleration has shown good performance for these types of flow conditions [2]. Challenges arise however when designing for off-design conditions, as these conditions generally introduce less favorable flow structures. This is especially the case for turbomachines as these operate over a wide range of conditions. Instead of designing a turbomachine to operate at a single design point, it is often desirable for the machine to be efficient over a wide range of conditions. Together with the complex thee-dimensional flow structures that are often encountered for turbomachinery, it can be concluded that the resultant flow structures that are formed at off-design conditions would be less then favourable. Unfortunately it has also been shown that steady solvers lack numerical stability for these flow regimes, resulting in convergence slowdown. It has been hypothesised by Xu et al. [1] that this degradation in performance is commonly encountered, but generally accepted as machine-zero convergence of the simulation is often deemed unnecessary. In combination with the limited group of users that is affected by these convergence issues, studies resolving these numerical instabilities has received little attention. Nonetheless turbomachinery simulations are often plagued by these numerical instabilities, as has also been encountered within the current work. In order to improve machine design at off-design conditions, obtaining the machine-zero solution is therefore critical for the aerodynamic analysis of turbomachinery.

As has been mentioned before, Xu et al. propose the usage of a Newton-Krylov method in order to over-

come these numerical instabilities. Their research lead to the development of their Newton-Krylov RANS solver, which showed improved robustness over a wide range of operating conditions. This NUTSCFD solver features a nonlinear flow solver which combines exact Jacobian matrix forming together with straightforward parallelization and a reliable globalization strategy. Independent of starting conditions, the solver showed reliable performance with respect to achieving machine-zero convergence. Xu et al. were therefore able to obtain a substantial increase in solver performance, especially with respect to turbomachinery.

2.4.2. SU2

SU2 is the CFD solver that is used and analysed within the current work. SU2 is an open-source computational toolbox which can be used to solve mathematical problems involving partial differential equations (PDE) [16]. It uses numerical methods which have been developed specifically for aerospace applications. SU2 is therefore able to solve complex multiphysics problems, where it uses unstructured mesh topologies and is developed using a modular structure. The code is structured in an objected oriented philosophy where it is written in C++ and Python. This allows for high code reusability, where code can be shared between modules. This provides a powerful structure, allowing modules to be used in a cohesive manner. The main module that is used for solving CFD problems is the SU2_CFD module. An overview of the code structure can be seen in figure 2.4, displaying a basic flow diagram of SU2_CFD. Below the diagram are examples of classes displayed which could be used during the simulation cycle. SU2's code is partitioned in classes such that it allows for standalone development of the classes.

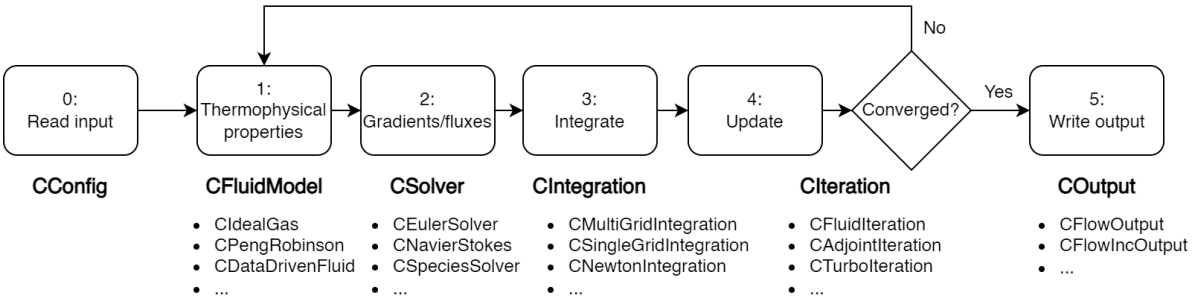


Figure 2.4: SU2_CFD flow diagram for an arbitrary flow simulation

As mentioned before, SU2's turbomachinery capabilities are currently developed at the TU Delft. The challenges described in section 2.4.1 have been encountered before, where SU2 also contains a Newton-Krylov solver. This Newton-Krylov solver uses FGMRES where different kinds of preconditioners can be used. Previous work has shown that SU2 is unlikely to match the performance of the NUTSCFD solver, where therefore further research is required. In order to analyse SU2's performance with respect to the NUTSCFD solver and the open literature, section 2.5 contains additional information with respect to relevant models which could improve SU2's performance.

2.5. ADDITIONAL CFD ACCELERATION TECHNIQUES

Sections 2.1, 2.2 and 2.3 describe the problems and solutions that are formulated when setting up a CFD flow problem. The schemes and models that are used for these types of problems can be computationally costly, which increases when increasing the complexity of the problem. This section contains an analysis of the solver acceleration techniques that have been mentioned within literature, and that have been used by Xu et al. in order to improve their solver performance. It has been mentioned in section 2.4.1 that multigrid can accelerate the solver, where therefore the current section starts with an analysis of multigrid (section 2.5.1). Increasing the accuracy of the solution increases the complexity of the system of equations, where multigrid might be able to have an effect on the convergence behaviour. This section is followed by an examination of how the formation of the Jacobian matrix might impact the solver's convergence time (section 2.5.2). It was found that the Jacobian can have a significant effect on convergence behaviour, where this forming process can especially be computationally expensive. The current section is concluded with an explanation of the blended Jacobian (section 2.5.3), which is used for forming the preconditioner within the NUTSCFD solver. It has been stated in section 2.3.2 that using a different preconditioner can have a large effect on the solver performance, where the formation of this preconditioner is therefore analysed as well.

2.5.1. MULTIGRID

The multigrid method is an approach which alters the solving process of a computational problem. The problem is projected upon multiple scales in order to accelerate convergence. The multigrid algorithm, also referred to as multilevel methods, can be applied to multiple mathematical fields allowing for geometric and algebraic manipulation. Following the resemblance to the computational problems commonly found in CFD, only geometric multigrid is analysed and considered within this thesis.

As mention in the sections above, CFD requires an iterative method in order to converge to a solution. Estimates are obtained for every iteration which can become costly with the increase of the number of cells, and thus with the increase in flow resolution. Clearly a finer high element mesh takes more computational time compared to a coarser mesh. This core principle is the basis for the multigrid method. Multigrid uses different mesh resolutions, where the solver is able to obtain a rough estimate of the final solution much quicker. A solution is obtained where additional refinement is required in order to converge to the finer (more accurate) final solution. Fundamentally the coarse grid allows for rapid wave propagation of flow variables which reduces convergence time, where the finer grid ensures greater accuracy. The multigrid method therefore switches between grid resolutions in order to solve the CFD simulation. The procedure of accelerating convergence following modeling strategies is also known as relaxation. It was found that SU2 contains multigrid, but it is currently unable to combine this with the Newton-Krylov solver.

GRID FORMATION AND INTERPOLATION

In order to solve the CFD simulation using multigrid, cells of the simulation grid are combined in order to form a coarser mesh (figure 2.5). These new coarser cells effectively represent the combined finer cells, where the flux balances of the finer cells are summed. By applying this technique to the whole grid, the original mesh is converted to a coarser mesh. In general this process is reiterated depending on the requirements of the simulation. Coarsening the mesh multiple times reduces computation effort, but can also introduce stability issues [17].

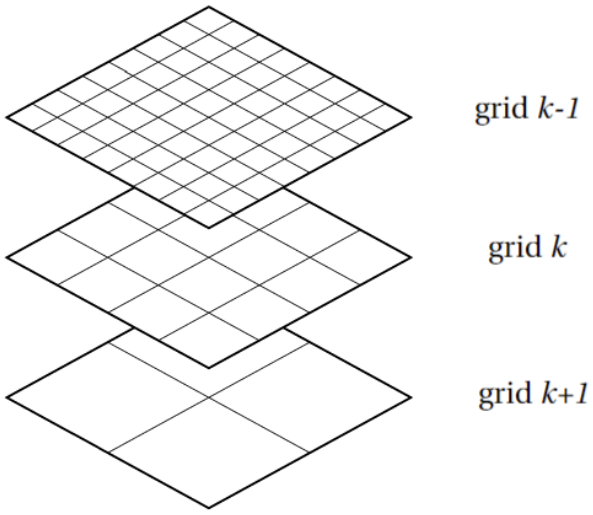


Figure 2.5: Example of multiple mesh resolutions following the same mesh, with decreasing number of cells [3]

The newly generated coarser meshes are considered to be independent from each other, where they do not influence each other directly. The process of reducing the number of cells to create a new coarser mesh is called restriction. Data transfer between the different mesh scales is however still required. This is possible following interpolation. Results that are found on the coarser levels are transferred upwards following corrections, accelerating convergence on the finer levels.

MULTIGRID FORMULATION

A formulation of the multigrid method can be given using subscripts. Different techniques are available as multigrid has developed over time, but the essence of the method will be explained within this section using Jamesons formulation [17]. In order to make a distinction between the different resolutions, grid levels will

be indicated using the subscript *k*. Grid *k-1* will then be the finer grid compared to grid *k* (see figure 2.5). In order to use the coarser grid *k*, the solution vector **w** must be initialized. This is achieved using transfer operation $T_{k,k-1}$.

$$\mathbf{w}_\mathbf{k}^{(0)} = T_{k,k-1}\mathbf{w}_{\mathbf{k}-1} \tag{2.21}$$

Within equation 2.21, $\mathbf{w}_\mathbf{k}^{(0)}$ then represent the initial solution vector on grid *k*. In addition to the solution vector, a residual forcing function has to be transferred as well. This due to the solution on grid *k* requiring the residual of grid *k-1*.

$$\mathbf{P_k} = Q_{k,k-1}\mathbf{R}_{\mathrm{k}-1}\left(\mathbf{w}_{\mathrm{k}-1}\right) - \mathbf{R}_\mathrm{k}(\mathbf{w}_\mathbf{k}^{(0)}) \tag{2.22}$$

Equation 2.22 shows the residual forcing function for grid *k*, containing the residual of grid *k* ($\mathbf{R}_\mathrm{k}(\mathbf{w}_\mathbf{k}^{(0)})$) as well as the residual of grid *k-1* ($\mathbf{R}_{\mathrm{k}-1}\left(\mathbf{w}_{\mathrm{k}-1}\right)$). $Q_{k,k-1}$ is another transfer operator allowing the usage of the finer grid residual. By combining the residual forcing function together with a time stepping schema, equation 2.23 can be setup which gives the general form using the solution vector at the next time step $q+1$.

$$\mathbf{w}_\mathrm{k}^{(q+1)} = \mathbf{w}_\mathbf{k}^{(0)} - \alpha_{q+1}\Delta t_k\left(\mathbf{R}_\mathrm{k}^{(q)} + \mathbf{P}_\mathrm{k}\right) \tag{2.23}$$

Within equation 2.23 Δt_k represents the timestep and α_{q+1} depends on the time stepping schema chosen. Depending on the setup of the multigrid schema, the solution for $\mathbf{w}_\mathrm{k}$ could be used as the initial data for grid *k+1*. If the solution $\mathbf{w}_\mathrm{k}$ is deemed to be sufficient, the data could also be transferred back to grid *k-1* following a correction. Let the superscript $^+$ indicate the final calculated value of the solution vector on a given grid level, then for grid *k-1* the corrected solution can be defined following equation 2.24.

$$\mathbf{w}_{\mathrm{k}-1}^+ = \mathbf{w}_{\mathrm{k}-1} + I_{k-1,k}\left(\mathbf{w}_\mathrm{k}^+ - \mathbf{w}_\mathrm{k}^0\right) \tag{2.24}$$

Within equation 2.24 $\mathbf{w}_{\mathrm{k}-1}$ represents the solution vector following the timestep (equation 2.23), $I_{k-1,k}$ the interpolation operator, $\mathbf{w}_\mathrm{k}^0$ the initialized solution vector on grid *k* (equation 2.21) and $\mathbf{w}_\mathrm{k}^+$ the final corrected value of the solution vector on grid *k*. Depending on the final number of grid levels, the grid with the least number of cells does not require a correction as there is no level below it. The transfer operators depend on the formulation that is used in order to represent the conservation laws on the mesh. When using a cell centered scheme, the operator $T_{k,k-1}$ is defined following equation 2.25, with *V* being the cell area or volume of the overlapping cells between the mesh levels. Similarly the residual follows from equation 2.26, being the sum of the integral cells on the finer mesh level *k-1*. Finally the interpolation operator $I_{k-1,k}$ represents the bilinear or trilinear interpolation. A detailed explanation for a vertex-centered schema is given by Jameson [17].

$$T_{k,k-1}\mathbf{w}_{k-1} = \frac{\sum V_{k-1}\mathbf{w}_{k-1}}{V_k} \tag{2.25}$$

$$Q_{k,k-1}\mathbf{R}_{\mathrm{k}-1} = \sum \mathbf{R}_{\mathrm{k}-1} \tag{2.26}$$

2.5.2. JACOBIAN MATRIX FORMATION

One of the aspects that plays a large role in the convergence time of the CFD solver is the formation of the Jacobian matrix. Equation 2.11 defines the governing equations in terms of the residual, where the term $\frac{\partial \boldsymbol{R}(\boldsymbol{W}^n)}{\partial \boldsymbol{W}^n}$ represents the Jacobian matrix. The dimensions of the Jacobian matrix depend on the size of the flow problem, where the size increases significantly with the increase in mesh accuracy. The Jacobian matrix is part of the coefficient matrix within the linear problem formulation $\boldsymbol{Ax} = \boldsymbol{b}$, where matrix inversion is therefore unviable due to its enormous computational cost. Instead the Jacobian matrix is used for the formation of the Hessenberg matrix with respect to the GMRES algorithm. This reduces the computational needs of the solver, as matrix-vector multiplications is used instead. Even though matrix inversion can be disregarded, the formation of the Jacobian matrix itself still requires significant computational power. Jalali et al. [18] showed that the time required for calculating the Jacobian matrix scales significantly with the increase in mesh refinement, where Xu et al. [1] showed that large percentages of the solver time are dedicated to the formation of the Jacobian matrix. Different algorithms exist which accelerate the formation of the Jacobian matrix. These algorithms include methods such as graph coloring, as has been used within the NUTSCFD solver. SU2 does currently not use an algorithm of graph coloring for forming the Jacobian matrix, where this might be relevant

depending on solver performance. Improving the convergence of the linear solver should however reduce the number of iterations that require Jacobian formation (as has been found by Xu et al.). This should therefore reduce the total number of nonlinear iterations, which has a significant effect on solver speed. It could therefore be considered either to accelerate the formation of the Jacobian matrix, or to improve the linear solvers in SU2. Looking into the effectiveness of the current FGMRES algorithm within SU2 is therefore relevant, as it influences the total number of costly Jacobian formations required during the simulation run.

2.5.3. BLENDED JACOBIAN MATRIX

It has been noted in section 2.3.2 that using an incorrect preconditioner can have a significant effect on the performance of the linear solver. Xu et al. use incomplete LU factorization with zero fill-in in order to define a preconditioner. Their preconditioner is based on a blended Jacobian matrix, where it is used as a right preconditioner. The blended Jacobian matrix is formed by combining Jacobian matrices based on second- and first-order-accurate spatial discretization. The assumption that a preconditioner based on the exact Jacobian matrix would be more efficient compared to an approximation of the Jacobian has therefore been shown to be invalid. Similar research conducted by Pueyo et al. [15] showed that using an approximation of the Jacobian matrix with improved diagonal dominance reduces solver stability issues. Pueyo et al. therefore introduced a variable which controls the artificial dissipation within the Jacobian matrix, where the resultant matrix is used in order to form the preconditioner. This variable controls the amount of second difference dissipation that is added to the first-order approximation of the Jacobian matrix, which therefore affects the formation of the preconditioner. The value of this variable and the effectiveness of the preconditioner based on the artificially introduced dissipation depends however on the structure of the functional Jacobian matrix (that is used in order to solve the linear system). McCracken et al. [19] showed that a preconditioner based only on the second-order spatial discretisation gives a solution with oscillatory artifacts, where these artifacts are not encountered when using the first-order approximation. The first-order preconditioner therefore introduces less stability problems for the solver, but also reduces the accuracy of the approximation that the preconditioner is based on. Instead, a weighted preconditioner matrix is constructed through the use of a weighting function to approximate the Jacobian matrix (equation 2.27). Within equation 2.27, $\mathbf{A}_2$ represents the second-order approximation of the coefficient matrix, $\mathbf{A}_1$ the first-order approximation and α is the weight parameter that determines the amount of second-order dissipation added to the final approximation of $\mathbf{A}$.

$$\mathbf{A}_\alpha = \alpha\mathbf{A}_2 + (1-\alpha)\mathbf{A}_1 \tag{2.27}$$

Equation 2.27 is a variation on the approach used by Pueyo et al, where equation 2.27 is also used by Xu et al. within their Newton-Krylov solver. Langer [2] underlined the importance of the preconditioner when using a Newton-Krylov solver, where the correct preconditioner reduces the total number of GMRES steps. By using a weighted approximation of the Jacobian matrix the benefits of the better conditioned first-order Jacobian matrix are combined with the better approximated second-order Jacobian matrix. This combination of matrices should improve the overall effectiveness of the preconditioner. SU2 currently allows for the usage of the ILU(0) preconditioner, where this preconditioner is not based on a blended representation of the first- and second-order-accurate spatial discretization of the Jacobian. Whether a blended preconditioner has a positive effect on SU2 is however uncertain. Within the current literature study it has been stated that the effectiveness of preconditioners can be situation specific, where SU2 currently differs to NUTSCFD on multiple aspects.

3

CASE STUDIES

Chapter 2 discussed the theoretical background of this thesis. A gap has been identified with respect to SU2's performance, where a detailed analysis of SU2's current behaviour is to be obtained. Chapter 2 discovered the different algorithms that have been used by Xu et al. in order to develop their NUTSCFD solver. From the literature study it was found that SU2 differs from NUTSCFD with respect to the implementation of the Newton-Krylov solver. The current chapter discusses the steps that have been taken in order to conduct a numerical study of SU2's current performance, which also allowed for the analysis of its Newton-Krylov solver.

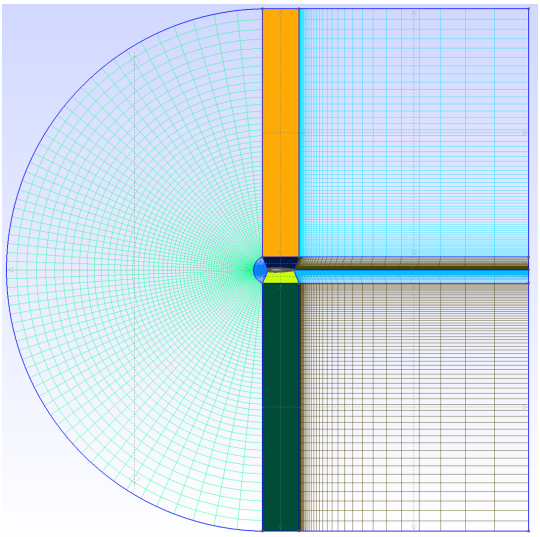
Chapter 3 discusses the development of the test cases that have been set up in order to test the performance of SU2. These test cases are run using SU2's RANS solver, where the results are obtained for steady simulations. The current chapter therefore outlines the procedures that have been used in order to test the goals of the thesis. Data obtained by the CFD solvers CFX, Numeca and NUTSCFD will be used to test SU2, where the test cases used for obtaining this data are recreated.

SU2's performance is tested using four different test cases. These test cases include models based on a NACA 0012 airfoil, a LS89 turbine stage, the MTU radiver compressor and the 1.5 stage ETH turbine. Both the development of the meshes and set up of the configuration files for these test cases are discussed within this chapter. The current chapter starts with section 3.1 which explains the development of the NACA 0012 test case. This test case will be used in order to test SU2's performance with respect to external flows. It should also be noted that this test case does not contain the turbomachinery module, thus providing a greater analysis of the code. This section is followed by section 3.2, discussing the development of the LS89 test case. Both the NACA 0012 and the LS89 test cases use 2D meshes and have been developed using a single-zone setup. The chapter continues with the development of the MTU test case in section 3.3 and the ETH test case in section 3.4. Both the MTU and ETH test cases are set up using the turbomachinery module and are created using a multizone configuration. All sections contain an explanation of the motivation behind the selected setup of the test cases, where the goal is to match the test cases to the reference literature. The chapter is concluded with section 3.5, containing the discussion of the development process. This section evaluates the validity of the test cases developed, where it also discusses additional changes that are made in order to obtain the results of chapter 4.

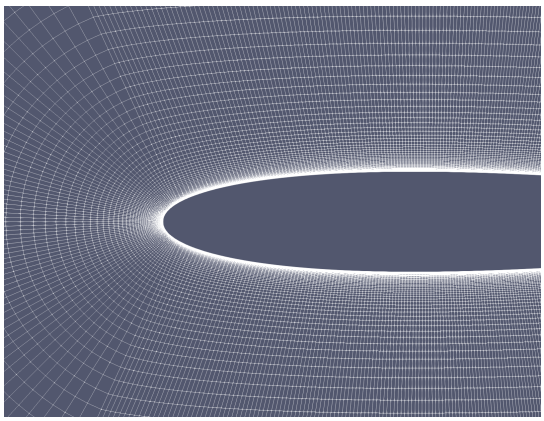
3.1. NACA 0012 AIRFOIL

The NACA 0012 airfoil is used in order to test the computational performance of SU2 on simple external flow problems. Different flow structures are created depending on the mesh, where the flow around a NACA 0012 airfoil differs greatly from the internal flow resulting from turbomachinery. The NACA 0012 test case is therefore set up in order to assess the performance of the SU2 solver for a larger range of flow regimes. The geometric data has been obtained online, where the mesh has been created using Gmsh (figure 3.1a). The mesh has been created using a C-grid that follows the contour of the airfoil, where it has been developed such that it contains smooth transitions in order for the C-grid to transition into the rectangular section. Vertex boundaries are used to match the element spacing between the sections, where exponential distributions

are used in order to refine the mesh near the boundary layer of the airfoil (figure 3.1b). In addition, it was also made sure that the mesh does not contain any elements with negative volume. The resultant number of grid points is matched to that of the mesh used by Xu et al. using the options of Gmsh. Xu et al. created a mesh with 134,976 grid-points, where the developed mesh contains 134,588 points. The NACA 0012 mesh is validated by comparing the results obtained by SU2 to the results obtained by NUTSCFD. Details involving the analysis of the data can be found in section 4.1.



(a) View of the NACA 0012 mesh



(b) Close-up view of the NACA 0012 mesh

Figure 3.1: Images of the NACA 0012 mesh

The NACA 0012 test case is set up such that it matches the reference literature. Xu et al. have provided data for subsonic and transonic conditions at different angles of attack. It should however be noted that Xu et al. did not provide any mesh data. 3 meshes have been used, where the mesh with the Level_0 grid density has been selected for the analysis of SU2. Details about the solver setup for the NACA 0012 test case can be found in table 3.1, where additional information involving the setup conditions can be found in table 3.2. The linear solver has been matched to the reference literature by allowing a maximum number of 500 inner iterations. It was also tried to use a maximum tolerance of 0.1, but this causes SU2 to crash.

Table 3.1: NACA 0012 test case solver settings

Settings	Type	Values
Turbulence model	SA Negative	
Flow numerical method	JST	Sensor coefficients (0.5, 0.01)
Linear solver	FGMRES	Error (1e-2) max iterations (500)
Spatial gradient model	Weighted least squares	
Fluid model	Standard air	R = 287.058 J/kg*K γ = 1.4
Viscosity model	Sutherland	μ_{ref} = 1.716e-5 kg/m*s T_{ref} = 273.15 K C_{sut} = 110.4
Conductivity model	Constant Prandtl number	C_{KT} = 0.0257
Turbulent numerical method	Scalar upwind	

Table 3.2: NACA 0012 test case conditions

Parameter	Mach number [-]	Angle of attack [deg]	Reynolds number [-]	Number of grid points
Value	0.76	0	$15 \cdot 10^6$	134,976

3.2. LS89 TURBINE CASCADE

The second test case that is used in order to test SU2 is based on the LS89 turbine cascade. The LS89 cascade has been designed and tested at the von Kármán Institute of Fluid Dynamics [20]. As opposed to the NACA 0012 airfoil test case, the LS89 test case requires the SU2 turbomachinery module where this test case is therefore created in order to analyse SU2's turbomachinery capabilities. The LS89 test case is analysed using the data obtained by Xu et al. and their NUTSCFD solver. A 2D test case for the LS89 has been developed by the SU2 community, where the mesh is available on the test case Github repo of the SU2 project. This 2D mesh is used for the test case that is developed within the current section. Xu et al. have developed multiple mesh levels for their LS89 test case, where the coarsest mesh has an element count of 14520 (called Level_5). The mesh developed by the SU2 community has 15568 elements (figure 3.2), where this mesh is therefore compared to the level_5 mesh. Xu et al. provide limited information about their mesh characteristics, where only the first layer of the cell height is documented. The configuration file that has been developed by the SU2 community is used. This configuration file has been edited in order to match the conditions of the reference literature.

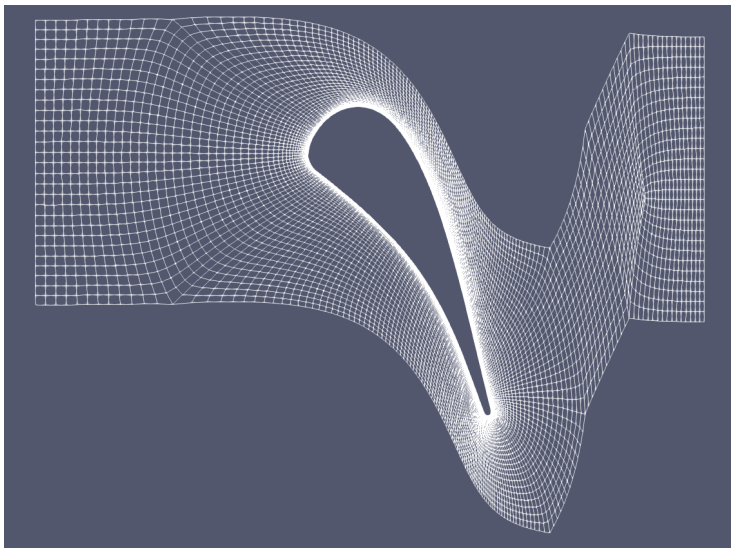


Figure 3.2: Wireframe view of the LS89 mesh developed by the SU2 community

The test conditions numbered "MUR43" are used in order to investigate the axial stage. Specifics for these conditions can be found in table 3.3. The test case has been setup such that it resembles the standard turbomachinery configuration used within the current work. Specifics for the solver setup can be found within table 3.4. It should be noted that using the 'MARKER_GILES' setting for the boundary conditions led the solver to crash. Instead separate inlet and outlet markers are used following 'MARKER_INLET' and 'MARKER_OUTLET'. In addition the constant viscosity model also crashed the solver, where this has been changed to the Sutherland model. It was tried to match the linear solver to the reference literature by using a maximum number of 500 iterations. Increasing the number of iterations resulted however in a simulation crash. The same was tried for the error tolerance of the linear solver, but this also resulted in SU2 to crash.

Table 3.3: Parameters for the MUR43 conditions for the LS89 test case

Parameters	Value
Mach number	0.84
Inlet total temperature	420 K
Inlet total pressure	143.5 kPa
Outlet static pressure	90.4 kPa
Reynolds number	$1 \cdot 10^6$

Table 3.4: Solver setup for the LS89 test case

Settings	Type	Values
Turbulence model	SA Negative	
Flow numerical method	JST	Sensor coefficients (0.5, 0.03125)
Linear solver	FGMRES	Error (1e-3) max iterations (2)
Spatial gradient model	Weighted least squares	
Fluid model	Standard air	R = 287.058 J/kg*K γ = 1.4
Viscosity model	Sutherland	μ_{ref} = 1.716e-5 kg/m*s T_{ref} = 273.15 K C_{sut} = 110.4
Conductivity model	Constant Prandtl number	C_{KT} = 0.0257
Turbulent numerical method	Scalar upwind	

3.3. MTU RADIVER COMPRESSOR

In order to assess the performance of the SU2 toolbox, a test case is developed based on the MTU compressor impeller (figure 3.3). This test case will be used in order to validate SU2's current performance with respect to radial compressors. The results obtained by Xu et al. will be used in order to validate the test case. The MTU compressor is an open test case which is provided by MTU Aero Engines [4], where it has been researched extensively within literature providing significant data for validation. The setup that is used within the current study is a configuration involving a vaneless diffuser. Motivation for the development of the MTU test case follows from the NACA 0012 and LS89 test cases. The goal for the MTU test case is to provide a more accurate assessment of SU2, where the accuracy of the analyses using the NACA 0012 and LS89 test cases is discussed in section 3.5.

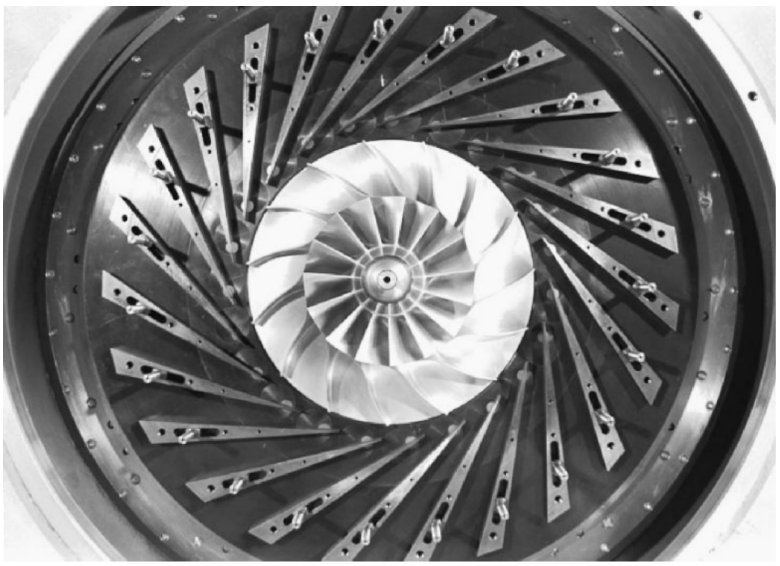


Figure 3.3: Top view showing the MTU impeller (light grey) with the outer diffuser vanes (dark grey) [4]

3.3.1. MESH GENERATION

The MTU compressor has been remodelled using the coordinate files of the blade profile. 11 blade profile sections have been used in order to remodel the MTU compressor, where the sections cover the impeller from hub to tip. ANSYS BladeGen is used in order to construct the complete impeller. The coordinate files have been manipulated such that they can be imported into BladeGen using the Data Import Wizard. Subsequently BladeGen creates curve files, which can be imported into Turbogrid. By orienting the coordinates of the blade correctly during the setup of the wizard, and by finishing the BladeGen Wizard, the design of the

MTU compressor is obtained within BladeGen (figure 3.4). Figure 3.4 contains the blade, hub and shroud profile of the MTU impeller, being the geometric model of the compressor. BladeGen does not construct a mesh of the impeller, where the mesh is created using Turbogrid. Relevant characteristics involving the MTU impeller are shown within table 3.5.

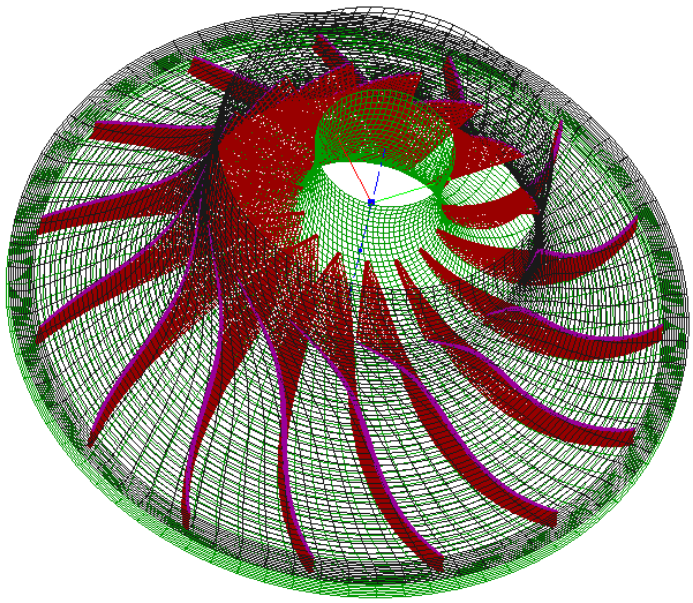


Figure 3.4: BladeGen model of the MTU compressor

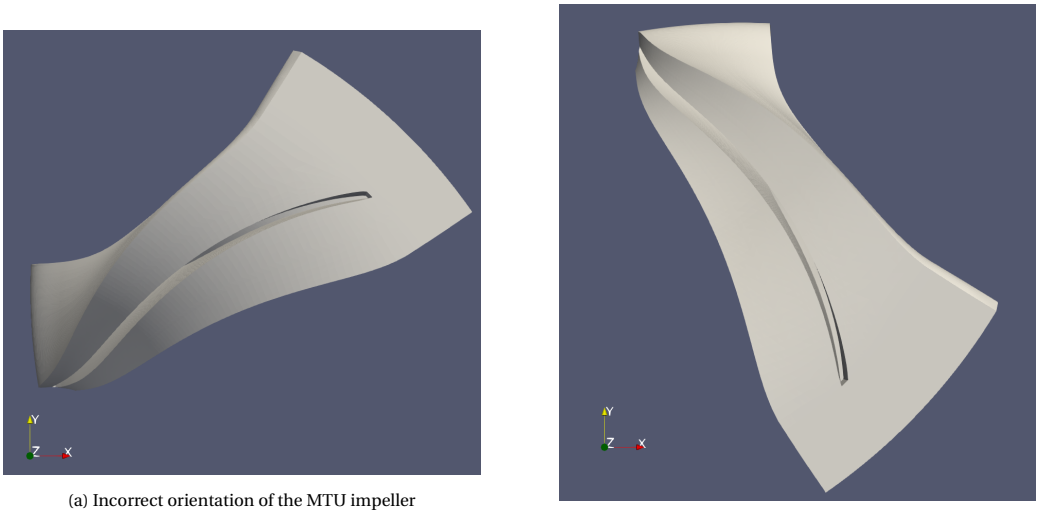
Table 3.5: Characteristics of the MTU impeller test case

Characteristics	Value
Tip radius	135 mm
Number of blades	15
Blade backsweep angle at impeller exit	38 deg
Shaft speed	3686.14 rad/s
Inlet total pressure	0.6 bar
Inlet total temperature	296 K
Outlet static pressure	1.013 bar
Impeller tip gap	0.7 mm

The BladeGen model is imported into Turbogrid using ANSYS Workbench. Using Turbogrid a mesh is created using the geometric model of the MTU impeller. Since the exact mesh of the MTU mesh used by Xu et al. is not available, a mesh of the MTU has to be created which mimics the mesh from the reference literature. The goal of the MTU study is to compare the performance of the CFD solvers, where therefore the number of mesh elements is matched. The resultant MTU mesh was chosen to have 858,102 elements and 907,308 nodes. The mesh developed by Xu et al. contains 905,953 grid points with a first layer viscous wall height of 10^{-6} m satisfying $y^+ \approx 1$. A second mesh of the MTU has been created containing 181.360 elements and 197.694 nodes. During the current work a large amount of errors have been encountered, where the smaller MTU mesh has been used in order to debug the runner. Both meshes contain no negative volume elements, where the standard Turbogrid analysis tools have been used in order to assess the quality of the mesh. In addition the mesh is forced to satisfy $y^+ \approx 1$ using Turbogrid.

Turbogrid does not export mesh files which are readily usable by SU2. The MTU meshes therefore have to be modified before they can be used within SU2. SU2 allows for multizone physical problems, where the MTU mesh was exported following 3 separate CGNS files. The configuration file of the MTU is based on a previously developed test case. This test case, called the Eckardt test case, uses multiple zones in order to

simulate the turbomachinery problem. Like the Eckardt test case, the MTU is divided into three zones: the inlet, blade passage and outlet section. These zones are exported separately using Turbogrid, where they are manually combined into a single mesh file following the SU2 multizone convention. The CGNS files also have to be re-orientated as otherwise SU2 is unable to solve the Turbomachinery problem: orienting the mesh incorrectly results in an error in the Turbomachinery Preprocessor, which computes a negative number of blades. Running the simulation with this negative blade count causes SU2 to crash. At the moment of writing the SU2 Turbomachinery Preprocessor therefore requires a specific orientation of the impeller in order to be able to start the solver. It is therefore strongly advised in order to match the turbomachinery mesh orientation to that of the example problems provided by the SU2 codebase. Similar issues have been encountered with the ETH axial stage turbine, where more information is provided in section 3.4.2. Figure 3.5 shows the difference in orientation, where the model in figure 3.5a has been rotated 90° around the Z-axis in order to obtain figure 3.5b. Despite its apparent simplicity, without the correct mesh orientation the solver will not run. Similarly the axis of rotation has to be the Z-axis in order to prevent SU2 from crashing. The mesh geometry has been rotated using Ansys ICEM, where it has been exported using the CGNS format.



(a) Incorrect orientation of the MTU impeller

(b) Correct orientation of the MTU impeller

Figure 3.5: MTU simulation orientation

The separate CGNS files that were exported from Turbogrid and ICEM are to be combined into one mesh in order to create a multizone mesh. This is done by using the SU2_DEF runner in SU2, which is able to convert the CGNS files into the .SU2 format. SU2_DEF is a mesh deformation tool, where it is able to interpret the CGNS files and export a deformed mesh. Without deforming the mesh, SU2_DEF exports the three meshes in the correct format, converting the files from CGNS to SU2. This results in three separate meshes: inlet, outlet and blade-section, which have to be combined in order to be used for a multizone simulation. Combining the separate sections can be done using a simple text-editor, where additional lines are required to indicate the different zones. This is done while making sure to follow the multizone format for SU2 meshes. With the mesh combined and converted to the SU2 format, the mesh can be used within the SU2_CFD runner.

3.3.2. NUMERICAL SETTINGS

The configuration file for the MTU test case is developed by using the old configuration file for the Eckardt test case. Numerical settings are copied and changed such that the new configuration file is compatible with the MTU test case. Different configurations have been used in order to test and analyse the performance of the developed MTU mesh. Settings are changed within the configuration file depending on the goal of the analysis. Several boundary conditions are however kept constant in order to make sure that the MTU test cases match the setup of the open literature. Detailed information with respect to the boundary conditions used by Xu et al. have been mentioned in table 3.5, where the total pressure and temperature are used for the inlet, and the static pressure for the outlet conditions. These conditions are specified using the Giles marker. The MTU mesh has been divided into three separate domains: the inlet, the blade section and the outlet (figure 3.6). These domains are connected following the multizone setup as has been mentioned in section 3.3.1. The different domains can be manipulated separately following the multizone setup, as is required in

order to enforce a rotation upon the blade section. Rotating the domain enforces the rotation of the hub and blade, which following table 3.5 rotate at 3686.14 rad/s. Important to note is that an additional boundary condition is required which fixes the shroud, as otherwise the shroud would not be stationary following the frame rotation. For SU2 this can be done using the *"MARKER_SHROUD"* configuration option. When setting up SU2 test cases it is advisable to use example configuration files of previous test cases. Although the donor configuration file might differ in its setup, it is advisable to consult previous test cases as SU2 could behave unexpectedly due to missing settings (as has been encountered with the mesh orientation in section 3.3.1). Additionally, table 3.6 contains the settings for the solver of the MTU test case.

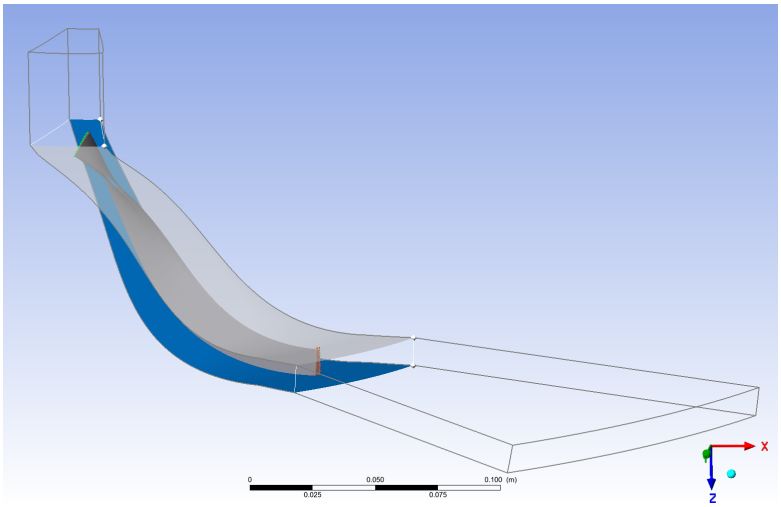


Figure 3.6: Turbogrid view of the three separate MTU domains: inlet, blade section and outlet

Table 3.6: MTU test case solver settings

Settings	Type	Values
Turbulence model	SA	
Flow numerical method	JST	Sensor coefficients (0.5, 0.03125)
Linear solver	FGMRES	Error (1e-4) max iterations (10)
Spatial gradient model	Weighted least squares	
Fluid model	Standard air	R = 287.058 J/kg*K γ = 1.4
Viscosity model	Constant Viscosity	μ_{ref} = 1.716e-5 kg/m*s
Conductivity model	Constant Prandtl number	C_{KT} = 0.0257
Turbulent numerical method	Scalar upwind	

3.4. ETH 1.5 STAGE TURBINE

In addition to the NACA 0012, LS89 and MTU test cases, the ETH 1.5 stage turbine is analysed using SU2. This test case is developed in order to validate SU2's performance with respect to axial turbines. Data obtained by CFX and Numeca is used in order to validate SU2's performance. The ETH turbine has been chosen due to its relevance within the turbomachinery field and due to the large amount of experimental data available. The ETH test case is based on the axial 1.5 stage turbine test rig that has been developed at the research turbine facility 'LISA' at the Turbomachinery Laboratory of ETH Zurich. This test rig has been developed in order to investigate models of steam and gas turbine stage configurations at different operating conditions [21]. For the current work the turbine test case developed for the investigation by Behr [21] is used, hereafter referred to as the ETH 1.5 stage turbine. The ETH 1.5 stage turbine is a 3-row turbine, consisting of 2 stators and a rotor. The turbine has been developed specifically to analyse the effects of tip leakage and secondary flows in unshrouded high-pressure axial turbines.

Two distinct meshes have been developed within the current work with respect to the ETH 1.5 stage turbine. Section 3.4.1 describes the development of the first ETH mesh, which was initially developed in order to compare SU2's performance to that of the second mesh. The first mesh has been created in-house at the university of the TU Delft, where this mesh is therefore denoted with ETH TUD. It should be noted that this mesh does not have a tip-gap region. The second mesh has been created externally. This mesh has been created by the Northwestern Polytechnical University in China, where their study analyses the performance of CFX and Numeca. Both CFX and Numeca are CFD software similar to SU2, where their accuracy is evaluated using the ETH test case. The mesh created by the Northwestern Polytechnical University is to be converted in order to be SU2 compatible, where the conversion process is described in section 3.4.2. The configuration of the test case for the ETH meshes is presented in section 3.4.3. The data that is obtained by Behr, CFX and Numeca is used in order to validate SU2's performance.

Behr used four separate measurement planes in order to collect experimental data from the ETH test setup. These four planes are used in order to compare the experimental data to the CFD results. Figure 3.7 shows the 2D meridional flow path of the ETH turbine, where the four planes are located at the inlet, at the outlet and in-between the stages of the 1.5 stage turbine. The experimental data is to be compared to the data obtained following these planes using span-wise averaging. The SU2 data-points are extracted using Paraview, where digital planes are created filtering only the relevant data-points that are positioned on the planes (section 3.4.4). These data-points are post-processed where the area average for every span-wise location is calculated.

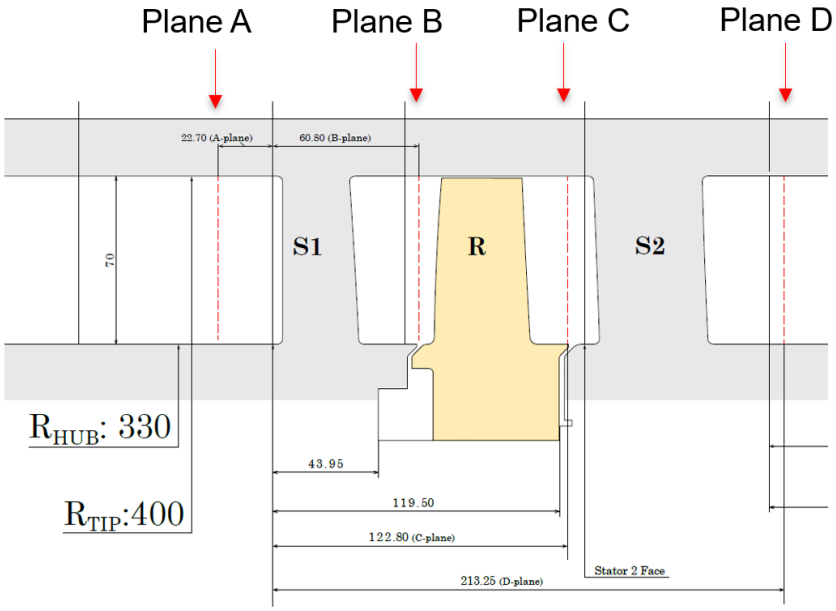


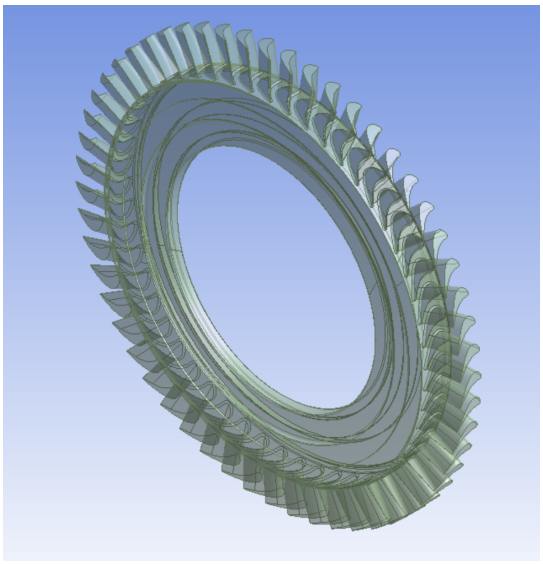
Figure 3.7: 2D meridional flow path of the ETH Turbine showing the measurement planes

The development of the TUD mesh and modification of the Northwestern Polytechnical University mesh resulted however in many complications. These complications are discussed extensively in section 3.5.4 and section 3.5.5. The final result is two TUD meshes, where instead of converting the Northwestern Polytechnical University, the first TUD mesh is altered in order to mimic the mesh created by the Northwestern Polytechnical University group. The final result of this section is therefore the 225k (no tip-gap) TUD mesh and the 900k (tip-gap containing) TUD mesh. It has to be noted that the goal of the study is to compare SU2 to CFX and Numeca, where the (first) tip-gap lacking TUD mesh is therefore redundant after creating the second TUD mesh. Nonetheless the results of the study involving the first TUD mesh are discussed within this work as they show SU2's performance for a different configuration of the ETH turbine.

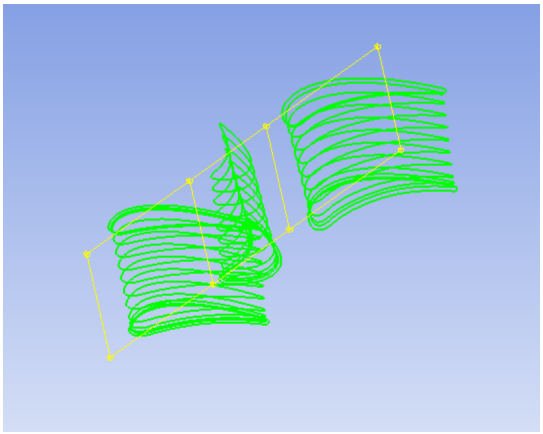
3.4.1. GENERATION OF THE TUD MESH

The TUD ETH mesh is developed using the CAD geometry of the ETH 1.5 stage turbine. A 3D CAD model has been developed, where it has been used together with 2D technical drawings in order to create the ETH test case. Unlike the procedure used in section 3.3.1 for developing the MTU mesh, Ansys DesignModeler (figure 3.8a) is used in order to develop the ETH mesh. This procedure was chosen due to the 3D CAD model, allowing for a more accurate mesh of the 3 rows. The geometry of the whole turbine has been imported, where the flow path is remodelled using the 2D design of the meridional view (figure 3.8b). Cross sections of the blades are created in order for Turbogrid to be able to import the blade model. An approximation of the blade is created within Turbogrid based on the blade profiles, where a sufficient amount of cross-sections is required in order to obtain an accurate representation of the blade. It was tried to import the whole blade volume into Turbogrid in order to obtain an exact representation of the blades, but due to unknown reasons this was not possible (DesignModeler giving an error). The CAD geometry of the ETH turbine contains fillets which are speculated to be the cause of this error.

All three rows are exported separately as Turbogrid does not allow for simultaneous meshing of the whole 1.5 stage. Transferring the geometric data and the flowpaths is done using the information flow of the Ansys Workbench. Workbench is a project platform which facilitates communication and project management between the different design components of the Ansys toolbox. Ansys Workbench allows for data propagation between the different components, where additional accuracy of the rows is obtained if desired. It was therefore chosen to use a minimum of 12 blade profiles for every blade in order for Turbogrid to generate an accurate representation of the CAD model. More layers are used near the shroud and hub of the stages in order to model the fillets correctly, where an even distribution of profiles has been created for the centre section of the blade. Important to note is that the axis of rotation should be around the Z-axis, as otherwise Turbogrid is unable to import the blade models and flowpaths.



(a) Rotor geometry of the 1.5 stage ETH turbine



(b) Geometric export of the geometry and the flowpath of the ETH turbine

Figure 3.8: DesignModeler data

The geometric data for the stages and the flowpaths is imported into Turbogrid, where Turbogrid is used in order to create the mesh (figure 3.9). Turbogrid provides tools which allow the possibility of mesh refinement and quality assessment. Both tools are used in order to refine the mesh while also reaching quality standards, where the default Turbogrid settings are used to analyse the mesh quality. It was made sure that all mesh volumes are positive, where 0.5% of the cells are affected by a skewness factor that exceeds the threshold of the default quality settings. At the moment of creation, the ETH mesh did not have a tip-gap due to complications with the Turbogrid software. The effect of this is discussed in section 3.5.4. Following these principles, three separate meshes for every row are created, where these have to be converted to a SU2 compatible format. Similar to section 3.3.1 the rows are exported from Turbogrid using the CGNS format, allowing for further processing.

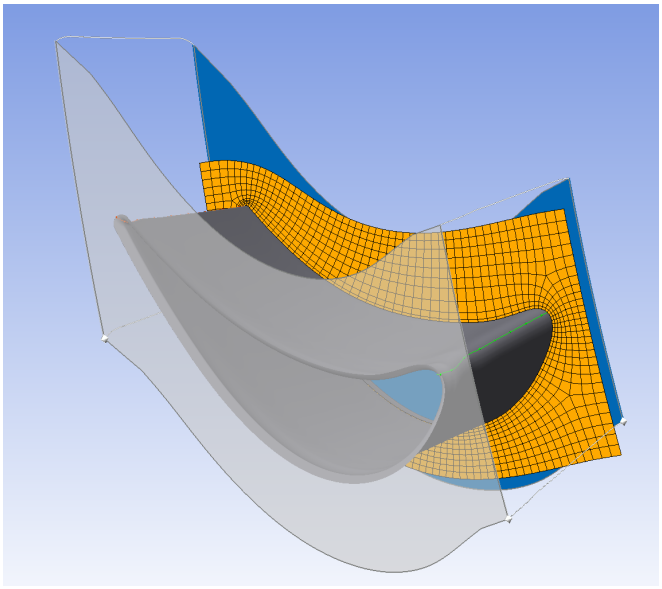
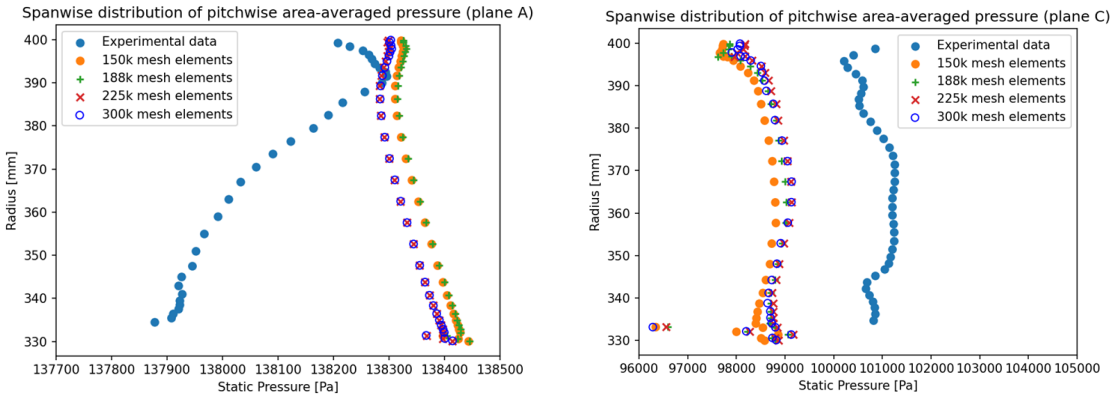


Figure 3.9: Turbogrid view of the second stator of the 1.5 stage ETH turbine

The CGNS files that are exported using Turbogrid are converted to a single multizone SU2 mesh by using the SU2_DEF module within SU2. Similar to section 3.3.1 the CGNS files are converted to the .SU2 format, where the rows are combined into a single mesh using the text editor and the multizone file style. No further translations or rotations of the mesh are required as the exported default orientation for the axial turbine is compatible with SU2. Since the ETH test case is to be compared to the experimental data by Behr, no previous indication of the number of cell elements required is available, thus requiring a grid independence study. Multiple meshes are created in order to ensure the cell count does not have an effect on the SU2 output. The results of the grid independence study are discussed within the current section, as they dictate the final element count that is chosen for the ETH TUD mesh. Different element counts have been considered, ranging from 150k to 900k elements. All simulations involving the grid independence study have reached iteration 2000, where different measurement planes (figure 3.7) have been used in order to assess the performance of the mesh.



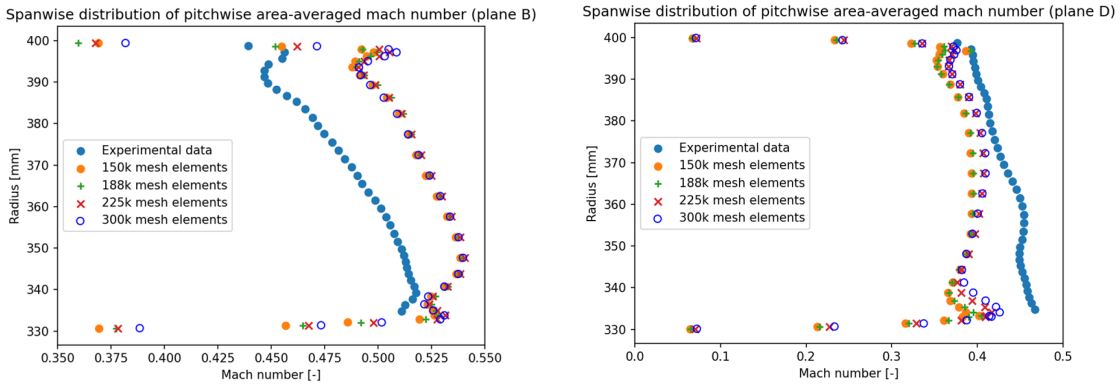
(a) Area-averaged spanwise static pressure at plane A

(b) Area-averaged spanwise static pressure at plane C

Figure 3.10: Grid independence study involving the TUD mesh with respect to the static pressure at 2000 iterations

Figures 3.10 and 3.11 show the results involving the grid independence study for the static pressure and Mach number for the different planes. All figures show the spanwise area-averaged distribution of the results, which are compared to the experimental data of Behr [21]. It can be concluded from these figures that the trends involving the different mesh sizes does not change drastically with respect to the number of elements. This is especially true for the results involving smooth behaviour with little fluctuations in values

(figure 3.10a), where greater deviation can be found near fluctuating conditions such as the hub and tip region (figure 3.11a). It can also be seen that the data-points differ more significantly downstream of the turbine (figure 3.10b and 3.11b) showing greater deviation in trends, indicating that the effect of the different element counts can have a cascading effect on the downstream results of the turbine. From the grid independence study it has been concluded that the 225k mesh should provide sufficient accuracy in order to observe trends involving the area-averaged results. This element count should provide a good indication of whether the results from SU2 are comparable to those of Numeca and CFX. The chosen element count could however be insufficient if more accurate results are desired. As mentioned before, additional simulations have been run using finer meshes. The results involving these finer meshes showed similar trends to that of figures 3.10 and 3.11 but differed slightly in values.



(a) Area-averaged spanwise Mach number at plane B

(b) Area-averaged spanwise Mach number at plane D

Figure 3.11: Grid independence study involving the TUD mesh with respect to the Mach number at 2000 iterations

3.4.2. GENERATION OF THE NPU MESH

The second mesh that is created with respect to the ETH 1.5 stage turbine has been developed at the Northwestern Polytechnical University (NPU). The NPU mesh consists of approximately 5590k elements and is used by the Northwestern Polytechnical University group in order to analyse the ETH test case with respect to their solver performance. The NPU group uses both Numeca and CFX where a comparison is made to the experimental data of Behr [21]. This mesh has been created using AutoGrid, where the mesh has been exported directly to the .SU2 mesh format. The three different rows are supplied separately where these have to be combined following the SU2 multizone setup. Several issues emerged when converting the mesh to an SU2 turbomachinery compatible mesh. These issues involved the axis of rotation and the orientation of the blades, where it is advised to use the same orientation as the example test cases developed by the SU2 community. Wrong configurations can lead to negative blade counts and the SU2 solver to crash. Issues with the NPU mesh have therefore been solved by making sure that the axis of rotation is the Z-axis, and by making sure the blades are positioned within the positive X-Y domain (figure 3.12).

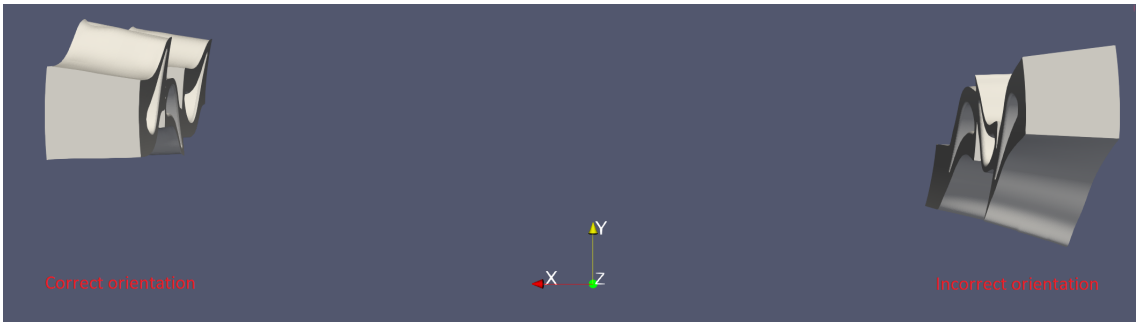


Figure 3.12: Difference in compatible SU2 orientation involving the ETH mesh

After converting the mesh to be SU2 compatible, it was however still found that SU2 was unable to run the NPU mesh. During simulating the ETH test case SU2 exited randomly at the start of the simulation iterations. This resulted in the NPU mesh being unusable and the development of another TUD mesh. The procedure of the NPU mesh being deemed unusable and the subsequent development of the second (900k) TUD mesh is discussed extensively in section 3.5.5.

3.4.3. ETH NUMERICAL SETTINGS

The ETH test case has specifically been developed to model the test rig that has been used by Behr [21]. The research turbine facility 'LISA' is capable of multiple configurations, where the experimental data by Behr is to be used in order to evaluate SU2's output. The relevant characteristics of the test rig following Behrs setup can be found in table 3.7. The configuration of the ETH test case is analogous to that of the MTU, such that similar boundary markers are used. Both meshes created, involving section 3.4.1 and 3.4.2, use the same test case configuration. Similar to the MTU test case, both the pressure and temperature are used in order to set the boundary conditions. This is achieved using the total pressure and temperature at the inlet, and the static pressure at the outlet. The inlet and outlet sections of the rows are connected using interfaces, allowing for data transfer between the zones. This is required as a multizone configuration is being used. The rotating frame is used in order to rotate the rotor section at 282.74 rad/s around the Z-axis, where the shroud of the rotor mesh is locked using the shroud marker (similar to section 3.3.2). In addition, table 3.8 shows the solver settings that are used for the ETH test case. It should be noted that these settings are identical to the MTU test case.

Table 3.7: Characteristics of the ETH turbine test case

Characteristics	Value
Stator 1 number of blades	36
Rotor 1 number of blades	54
Stator 2 number of blades	36
Shaft speed	282.74 rad/s
Inlet total pressure	1.4 bar
Inlet total Temperature	328.15 K
Outlet static pressure	0.9 bar
Mass flow	11.7 kg/s

Table 3.8: ETH test case solver settings

Settings	Type	Values
Turbulence model	SA	
Flow numerical method	JST	Sensor coefficients (0.5, 0.03125)
Linear solver	FGMRES	Error (1e-4) max iterations (10)
Spatial gradient model	Weighted least squares	
Fluid model	Standard air	R = 287.058 J/kg*K γ = 1.4
Viscosity model	Constant Viscosity	μ_{ref} = 1.716e-5 kg/m*s
Conductivity model	Constant Prandtl number	C_{KT} = 0.0257
Turbulent numerical method	Scalar upwind	

3.4.4. ETH SU2 DATA POST PROCESSING

Data is being extracted from SU2 using Paraview, where Paraview can be used in order to plot the data. In order for the SU2 data to be compared to the data obtained by CFX, Numeca and Behr, additional post-processing is required. Paraview is used in order to visualize and post-process the results that are obtained by SU2. Within Paraview digital planes can be created at the locations of the measurement planes in order to extract the relevant data. These planes filter out the data points that are located on these planes. Figure 3.13 shows the measurement planes A, B, C and D in white, where the rotor stage is visible as well. Using

these planes, CSV data is filtered and extracted where additional post-processing is done using Python. The results that have been obtained by CFX, Numeca and Behr are area-averaged where a spanwise distribution is created using the averaged data. The CSV results extracted using Paraview are therefore averaged in order to allow for a valid comparison. Python code has been written in order to calculate the spanwise distribution using the radius location of the data points, where Python has also been used in order to plot the data.

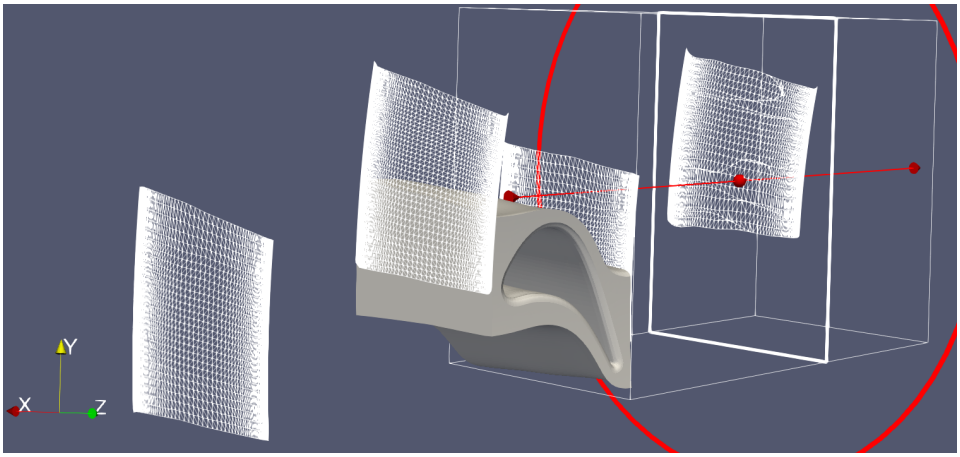


Figure 3.13: Measurement planes shown in white within Paraview

3.5. CASE STUDIES SUMMARY

This section provides an overview of the test cases that have been developed, where it also evaluates the validity of these test cases. Many issues emerged during the analysis of these test cases, which had an effect on the final outcome of the work. The current section therefore also discusses the adjustments that were made to the test cases following these issues. This section starts with a discussion of the NACA 0012 (section 3.5.1) and LS89 test cases (section 3.5.2). These sections are followed by a discussion on the validity of the MTU mesh (section 3.5.3) and the ETH TUD mesh (section 3.5.4). The current section is concluded by section 3.5.5 which discusses the further development of the NPU mesh in order to obtain a valid test case. This section outlines the creation process of the 900k TUD mesh, which will be used as a substitute to the NPU mesh, as the NPU mesh was found to be unusable. The 900k mesh is not to be confused with the original TUD mesh created in section 3.4.1. Due to complications with the NPU mesh, both TUD meshes will be used in order to study the ETH test case. The difference between these meshes is the element count and the tip-gap. For sake of clarity the original TUD mesh is called the 225k ETH mesh, where the new TUD mesh is called the 900k ETH mesh. The reason for the absence of a tip-gap in the 225k mesh can be attributed to the lack of experience with Turbogrid for tip-gap creation during the development of the original 225k test case. The 900k test case has been developed at a later period compared to the 225k test case. The absence of the tip-gap involving the 225k test case is discussed in section 3.5.4 as well.

3.5.1. NACA 0012 TEST CASE

The results obtained by the NUTSCFD solver will be used as reference data in order to analyse the performance of SU2 with respect to the NACA 0012 airfoil test case. This data will be used in order to verify whether the currently developed test case gives correct results, while also using the data in order to assess the convergence behaviour of SU2. Little information is provided with respect to the mesh used by Xu et al. where therefore it is not possible to do an exact comparison between the literature and the obtained results. Efforts are however being made to make sure both meshes contain the same number of elements. The mesh developed for SU2 was created using Gmsh, where Gmsh does not have specific options handling y^+. It is therefore currently unsure whether the new mesh meets the condition of $y^+ < 1$, where the reference mesh does. Due to these limitations, the results that could be obtained following this analysis are limited. Instead the results from the NACA 0012 test case will be used to give a broader indication of the current performance of SU2, where it also provides an indication of solver performance with respect to external flow conditions. In addition, these results will be used in order to compare SU2's performance to that of a turbomachinery test case. The NACA 0012 test case therefore provides a broader analysis of SU2 as it does not use the turbomachinery

module. The results for the NACA 0012 test cases therefore also serve as a basis for a comparison between different SU2 test cases, especially in combination with the LS89 turbine cascade. A direct comparison involving SU2 is however not possible, as the mesh does not match the reference literature.

3.5.2. LS89 TEST CASE

The LS89 turbine test case has been developed using the mesh created by the SU2 community. Like the NACA 0012 test case it is therefore likely that the mesh differs from the reference literature. Xu et al. have developed multiple mesh levels for the LS89 test case, where the mesh level closest to the current LS89 element count is chosen. As the mesh used for the SU2 LS89 test case has been created by the SU2 community, it is currently unsure whether the current mesh meets the conditions set by the reference literature. Xu et al. state that the first layer of cell height is $1.7 \cdot 10^{-5}$ m, where this is unknown for the SU2 mesh. It is likely for this difference in mesh geometry to have an effect on the accuracy of the solver analysis, but the outcome will also be used in order to compare SU2's performance to that of the NACA 0012 test case. Despite these discrepancies the outcome of the LS89 test case is therefore still relevant to the current work.

3.5.3. MTU TEST CASE

An MTU test case has been developed based on the MTU radiver compressor. This model of the MTU impeller has been created using BladeGen, where a more efficient technique of creating the impeller model might be achieved by using Ansys DesignModeler. Small anomalies are created when using the BladeGen Wizard (figure 3.14), which deviates the digital model from the geometric data. Figure 3.14 shows the exact profile of the blade using the pink line, where the interpreted model is created using the grey line (mimicking the blade profile). This import step is required by BladeGen in order to obtain correct leading- and trailing-edge behaviour. It is possible to match the leading- and trailing-edge of the profile, but due to its parabolic nature of the geometric approximation, changing the approximation results in a deviation in blade profile near the centre of the blade.

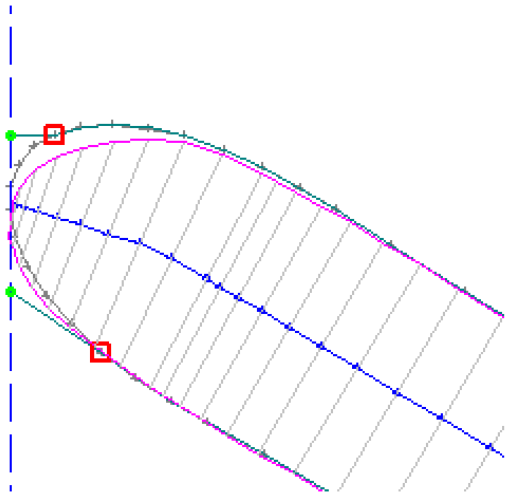


Figure 3.14: Deviation in leading-edge tip geometry showing the blade geometry (pink line) and the BladeGen approximation (grey/green line)

Given this deviation in interpretation, it might be proposed to model the impeller using the procedure mentioned in section 3.4.1. This procedure uses Ansys DesignModeler instead of BladeGen. Although this has not been considered in the current study, the data required for Turbogrid has been summarized in figure 3.8b, showing the blade profiles and the flowpath of the ETH turbine stage. Data for both the blade profiles and flowpath of the MTU impeller have been obtained (figure 3.15), where importing the MTU into Turbogrid using DesignModeler is expected to be achievable. The mesh used in the current study is however based on the model created using BladeGen. In the current work, a significant number of errors have been encountered with respect to the MTU test case. It is unlikely for the currently chosen meshing procedure to have an effect on the solver, as the developed SU2 mesh is also used within the CFD solver CFX. The results following CFX are not discussed within this thesis, as it is not part of the current work done, but this does indicate the validity of the developed mesh since CFX is able to converge using this mesh.

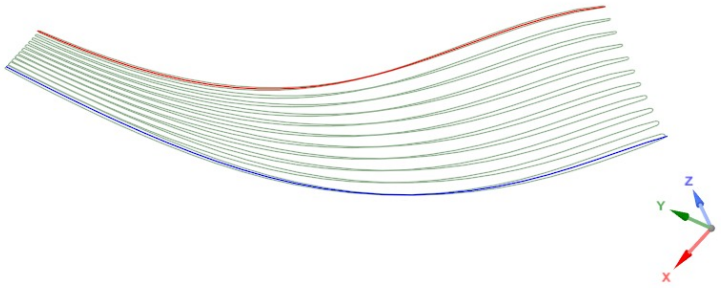


Figure 3.15: Geometric data of the MTU impeller, showing the blade sections (green), shroud (red) and the hub (blue)

It should be noted that the currently developed mesh of the MTU contains the problematic tip-gap interfaces 'SHROUD_TIP_GGI_SIDE_1' and 2. It was found during the development of the ETH test case that these interfaces should not be exported from Turbogrid. Detailed explanation for this, and the handling of these interfaces can be found in section 3.5.4. The problematic nature of these tip-gap interfaces was only realised during the development of the ETH test case, meaning that the results obtained for the MTU impeller (section 4.3) are obtained using these interfaces. It should however be noted that section 4.3.1 also contains results following an MTU test case that does not have a tip-gap region. The tip-gap interfaces are not created if the tip-gap region is excluded, therefore allowing for analysis of the effect of the tip-gap interfaces on the solver performance.

Several notes have to be made with respect to the reference file on which the MTU test case is based. The Eckardt reference configuration file has been altered in order for it to be usable for the MTU test case. It was found during development that the mixingplane caused issues, where this has been replaced by a combination of the 'MARKER_ZONE_INTERFACE' and the 'MARKER_FLUID_INTERFACE'. Similar findings have been found with respect to the ETH test case, where this is discussed in section 3.5.4. In addition, it should be noted that the settings of the linear solver were not matched to that of the NUTSCFD solver. This was not done as convergence of the MTU test case was prioritised, where the default settings of the Eckardt file have been used instead.

3.5.4. 225K ETH TUD TEST CASE

The first ETH mesh that has been developed in-house at the TU Delft does not have a tip-gap. The goal of the ETH study is to develop a mesh that is comparable to the mesh developed by the Northwestern Polytechnical University group. The absence of the tip-gap region therefore provides a large difference in performance, where it is likely for the simulation results to differ from the reference case. Nonetheless the results are presented, as the initial TUD mesh provides insight into the performance of SU2 at different operating conditions. From the results (section 4.4.1) it was however later concluded that the absence of the tip-gap region provides inaccurate results with respect to the reference test case, which led to the development of the NPU mesh (section 3.4.2) and subsequently the second (900k) TUD mesh. Following new Turbogrid features previously unknown to the author, it was possible to create a mesh which contains the tip-gap region. This knowledge has been used in order to create the second (900k) TUD mesh, which is discussed in section 3.5.5. A distinction is therefore made between the first (225k) and second (900k) TUD mesh, where the first TUD mesh has not been changed involving the tip-gap region. The first TUD mesh is therefore not sufficient in order to do a direct comparison to the results obtained by the Northwestern Polytechnical University group.

3.5.5. 900K ETH TUD TEST CASE

This section provides a summary of the work that has been done in order to obtain the 900k ETH TUD test case. This test case is created by modifying the 225k TUD mesh such that it includes a tip-gap region. The 900k TUD mesh has been developed due to the NPU mesh being too fine for SU2. The assessment of the NPU is discussed within the current section as well.

TURBOGRID TIP-GAP PROCESSING

Due to its relevance to the final result of the ETH study, the Turbogrid procedure that was used in order to create the valid tip-gap region is discussed here. It should be noted that this procedure was not used for creating the 225k test case. Turbogrid allows for the creation of the tip-gap region by specifying the blade length. When using the default Turbogrid settings, creating a tip-gap region results in the creation of 2 surfaces in-between the shroud and blade (figure 3.16). Figure 3.16 shows several interfaces that are created when exporting the ETH mesh using Turbogrid. These interfaces are to be assigned the correct boundary conditions in order to make sure the simulation provides accurate results. Figure 3.16 also shows the formation of the two interfaces 'SHROUD_TIP_GGI_SIDE_1' and 2, which are indicated with the purple surfaces. These interfaces are created in-between the blade and the shroud, where these are internal interfaces. At the moment of writing, SU2 is currently unable to assign the correct boundary conditions to these internal interfaces (SU2 version 7.5). The tip-gap interfaces therefore have to be altered in order for SU2 to be able to simulate a test case using a tip-gap region.

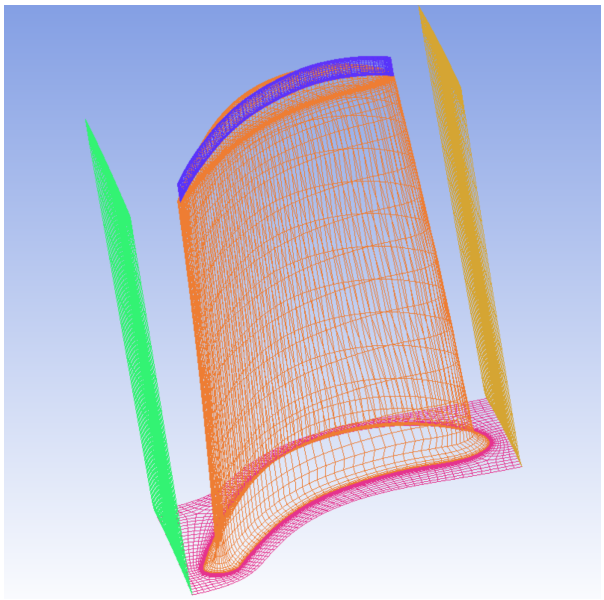


Figure 3.16: View of several mesh interfaces that are created when exporting the rotor ETH mesh

Turbogrid has been used in order to prevent the creation of the SHROUD_TIP_GGI_SIDE interfaces when exporting the mesh. At the moment of writing this has been achieved by using the Conformal Tip setting that is available in Turbogrid. This setting can be enabled when using the 'Beta Features'. Enabling these settings allows Turbogrid to use a mixture of structured hexahedra elements at the leading and trailing edge of the blade, where it uses wedge elements in-between (figure 3.17). This prevents the creation of the SHROUD_TIP_GGI_SIDE interfaces while maintaining the tip-gap region. Since this mesh does not contain the internal interfaces, the exported mesh is SU2 compatible.

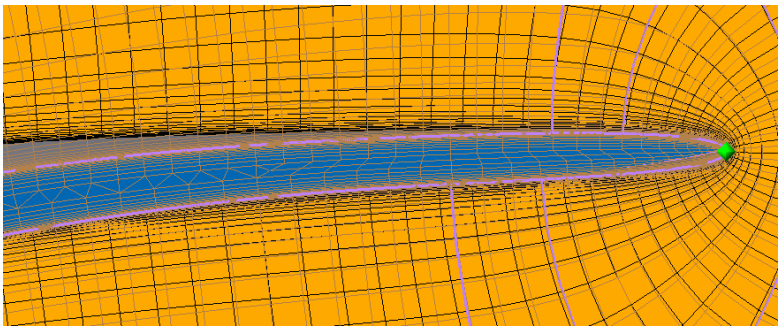


Figure 3.17: Turbogrid meshing at the blade tip when using the Conformal Tip setting

During the development of the ETH mesh it was made sure that the y^+ number is below 1. This was done in order to make sure that the near-wall resolution of the mesh is accurate enough such that near-wall flow phenomena can be modelled accurately. This y^+ condition has been enforced by using Turbogrid, where this has been met with respect to the mesh of rotor 1 and stator 2. It was however not possible to achieve this same resolution with respect to the mesh of stator 1. Reducing the y^+ number to below 1 involving the mesh of stator 1 resulted in the formation of negative volume elements. Since maintaining positive volume elements is considered to be of greater significance, the condition of $y^+ < 1$ is not reached. Instead the y^+ number of the mesh of stator 1 is approximately 4.5. It was however found that by changing the inlet channel length of stator 1, Turbogrid allowed both conditions to be met. By manipulating the initial flow path it might therefore be possible to create a mesh that meets both conditions, providing a more accurate representation of the turbine. This was however not considered within the current work as this outcome is uncertain. The mesh that has been obtained by the Northwestern Polytechnical University group has a y^+ of approximately 3, where since the condition of $y^+ < 1$ is met regarding rotor 1 and stator 2, the in-house developed ETH mesh is deemed sufficient. It should be noted that these same conclusions involving the y^+ number have been reached for the 900k TUD mesh that is discussed in section 3.5.5.

Finally, it should be noted that the setup of the ETH configuration differs from the reference test case involving the mixingplane. It was found that the mixingplane marker caused SU2 to diverge. Similar to the MTU test case, the mixingplane markers have been replaced by a combination of the fluid and zone markers. It should be noted that 'MARKER_GILES' should also be altered, where the mixout and mixin planes are omitted. The default settings for the linear solver have been used as well, since the ETH test case is not used in order to test the NUTSCFD solver.

NPU MESH ASSESSMENT

The mesh that has been supplied by the faculty of Northwestern Polytechnical University (NPU) was unusable in combination with SU2. The mesh that has been provided contains 5590k elements, where different setups have been tested in order to investigate the issue. An overview of the different scenarios can be found in table 3.9. Table 3.9 shows different combinations of the mesh, where the configuration file has been kept constant (excluding the setup of the boundary conditions since these depend on the mesh).

Table 3.9: SU2 solver result for different configurations following the ETH test case created using the NPU mesh

Setup	SU2 Outcome
Row 1 + 2 + 3	Simulation killed
Row 1 or Row 2 or Row 3	Runs normally
Row 1 + Row 3	Runs normally, crash at exit
Row 1 + Row 2 or Row 2 + Row 3	Simulation killed

Since SU2 does not provide an error output involving these configurations, several topics have been investigated in order to find the cause of the simulation crashes. These topics involve the configuration file, the tip-gap region, the mixing plane and the rotor mesh. It was first made sure that the configuration file is correct, by using the mesh created in section 3.4.1. This setup was able to converge to a solution, where the issue is therefore unlikely to be due to the configuration file. The tip-gap region was considered due to its significant effect on the test case, and due to prior issues involving the MTU mesh. A new ETH NPU mesh was therefore created excluding the blade tip-gap. Removing the tip-gap did not have an effect on the solver, where the same result was obtained following table 3.9. Different mixing plane settings have been considered as issues emerged when combining the different rows (table 3.9). It was found that this also did not have an effect on the final result. Finally, different meshes have been combined in order to investigate the issue. Table 3.10 shows different setups involving the mesh created by the faculty of Northwestern Polytechnical University (NPU) and the mesh created in section 3.4.1 at the university of Delft (TUD). This table shows the combination of different rows in order to further investigate the cause of the simulation crashes.

Table 3.10 indicates that there might be a problem with the NPU mesh itself. Due to time constrains it was therefore considered to alter the TUD mesh such that it could mimic the performance of the NPU mesh. When modifying the TUD mesh it was however observed that SU2 was unable to run the altered TUD mesh. Similar errors were obtained where SU2 did not provide an error message. Reducing the element count of the TUD mesh would eventually result in a successful SU2 simulation, where it is therefore expected that the

Table 3.10: SU2 solver result for different mesh configurations following the ETH test case created using the NPU mesh and the TUD mesh

Setup	SU2 Outcome
Row 1 (NPU) + Row 2 (NPU) + Row 3 (NPU)	Simulation killed
Row 1 (NPU) + Row 2 (TUD) + Row 3 (NPU)	Simulation killed
Row 1 (NPU) + Row 2 (TUD)	Simulation killed
Row 2 (TUD) + Row 3 (NPU)	Simulation killed
Row 1 (TUD) + Row 3 (NPU)	Runs normally
Row 1 (NPU) + Row 3 (TUD)	Runs normally
Row 1 (NPU) + Row 3 (NPU)	Runs normally, crash at exit

large size of the NPU mesh is the issue that is causing SU2 to crash. It might be possible for SU2 to be able to handle large size meshes, where the issue could be raised by the Turbomachinery module. It was therefore not possible to test the performance of SU2 with respect to the supplied NPU mesh, where instead the modified TUD mesh is used.

Different mesh sizes have been considered for the modified TUD mesh, where the NPU group has used meshes with 5590k, 3340k and 1930k elements. All three meshes have been recreated, where only the 1930k mesh did not result in the SU2 solver to crash. The 1930k TUD mesh was therefore initially chosen as a substitute for the NPU mesh. Since only the recreated 1930k TUD mesh is able to run using SU2, it is therefore also expected that the reason for the NPU mesh to crash is due to its element count and not the setup of the test case (as has been hypothesized above). Following figure 3.18 the 1930k TUD mesh showed however stalling behaviour, where it was not feasible to run the mesh indefinitely due to time and performance constraints. The peaks in figure 3.18 are the result of the simulation restarting, which therefore can be ignored. A new mesh of 900k elements has been created instead and is used as the substitute for the NPU mesh. It has been shown before following the grid independence study that a mesh of 225k elements should be sufficient when simulating the ETH test case (section 3.4.1). The grid independence study showed small deviations in trends, where a 900k element mesh had been considered as well. Although the results for this finer mesh following the grid independence study showed similar results, the mesh documented within the current section is chosen to have 900k elements. This was done so since the goal of this section was to mimic the NPU mesh, where the 900k element mesh was the first coarser setting that prevented SU2 from crashing. It is therefore likely for this mesh to have an inefficient element count. Reaching convergence with this new mesh did however require approximately a day of running on the current hardware, where this was therefore deemed acceptable.

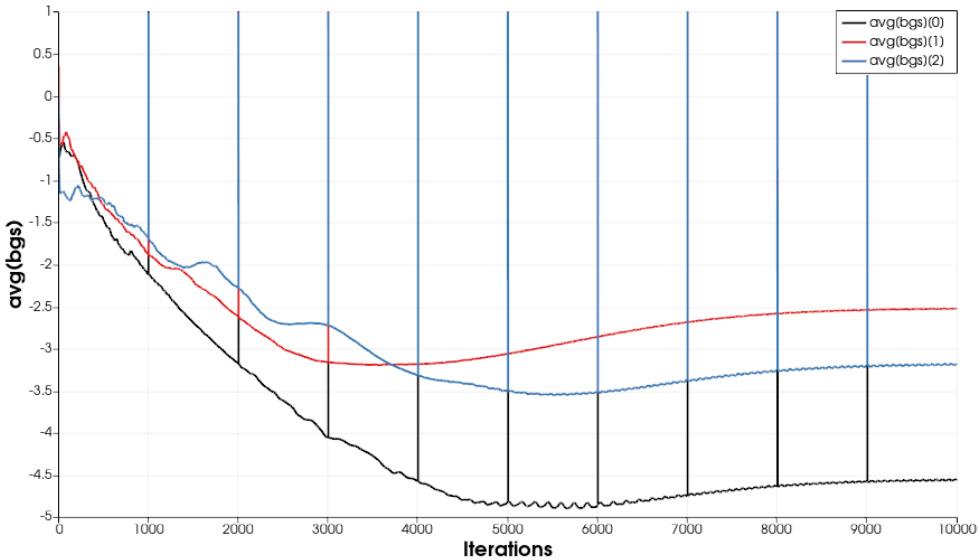


Figure 3.18: Residual behaviour of the ETH 1930k TUD mesh for 10000 iterations

Although the 900k mesh mimics the NPU mesh, it is possible for the simulation results to differ from the results obtained by the NPU group. Alghouth both meshes have a tip-gap (unlike the mesh created in section 3.4.1), the 900k mesh differs in the number of elements used. It is however unlikely for the difference in element count to impact the final results significantly. This is concluded following the results of the grid independence study done in section 3.4.1. This grid independence study was however conducted using the mesh without the tip-gap region. It should also be noted that there is a slight deviation in geometry compared to the CAD model following the meshing procedure mentioned in section 3.4.1. Instead of using the exact CAD model, the geometric export uses sections which results in a different blade profile. These effects should affect the final simulation results, but their significance is expected to be minimal.

4

RESULTS

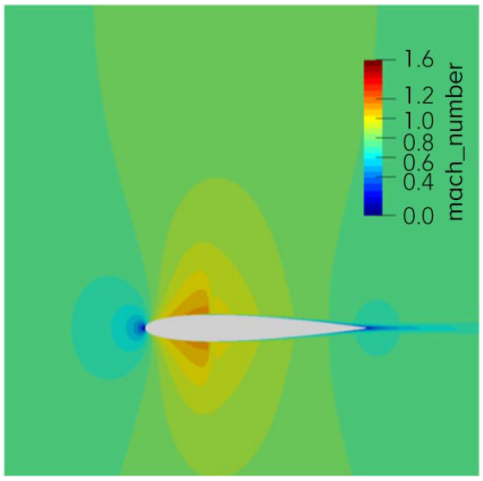
The previous chapter covered the development of the test cases. These test cases are used in order to assess the performance of SU2. It was explained how these models have been set up, where the validity of these models has been discussed. The current chapter contains the SU2 results following these test cases, where the performance of the solver is analysed using the residual and simulation results. The residual is used to give an indication of the error of the approximate solution, where plotting the whole residual over the span of the simulation gives an idea of the solver's performance.

The current chapter is divided into sections which discuss the results that have been obtained. This chapter starts with the analysis of the single zone test cases using the NACA 0012 airfoil (section 4.1) and the LS89 turbine stage (section 4.2). Both test cases are used in order to compare the current result obtained by SU2 to that of Xu et al. [1] and their NUTSCFD solver. The NACA 0012 test case serves as a basis for which the performance of SU2 can be compared with respect to external flow. These flow scenarios involve relatively simple flow structures with respect to turbomachinery. The solver performance involving a single zone turbine stage will be tested using the LS89 test case. This scenario results in more complex flow structures, where the test case is more representative of the subject of interest. Both test cases are also used in order to test the performance of the Newton-Krylov solver in SU2. The chapter continues with the results for the MTU test case in section 4.3. It was found that the MTU test case is unable to reach convergence, where a detailed analysis is conducted in order to identify potential problems created by the test case. This section is followed by the analyses of the results obtained by the ETH test case (section 4.4). The development of the ETH test case has resulted in two ETH meshes, where a distinction is made between the 225k and 900k TUD meshes. Section 4.4 starts with the validation of the results obtained by the ETH test case by comparing the data to that of other solvers. These results are initially obtained using the standard solver, where this section continues with the performance analysis of the ETH test case with respect to the Newton-Krylov solver. Finally, the current chapter is concluded with the discussion of the results, which can be found in section 4.5.

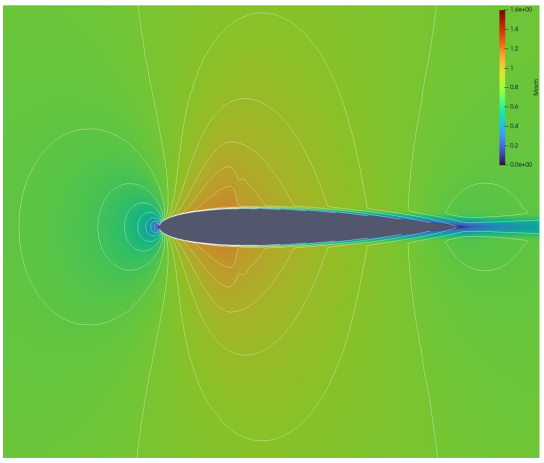
4.1. NACA 0012 RESULTS

The NACA 0012 test case is validated using the Mach data obtained from the literature. The results obtained by Xu et al. also allow for the convergence assessment of SU2. Figure 4.1 shows both the Mach contours obtained by the NUTSCFD solver (figure 4.1a) and SU2 (figure 4.1b). Both figures show identical results with respect to the Mach contour, where it is concluded that the NACA 0012 test case has been set up properly. The convergence behaviour of the test case is analysed using the nonlinear steps of the solvers. The maximum number of iterations allowed for the linear solver is set to 500, similar to that of the NUTSCFD solver. A comparison is made in figure 4.2, where both the results for SU2 and NUTSCFD are plotted. Following figure 4.2a, the convergence behaviour of 'd' is to be compared to that of SU2. The other results have been obtained for different angles of attack, which are not analysed within the current work. Figure 4.2b shows the convergence behaviour of SU2. Figure 4.2b contains two results, where the convergence of a normal setup is compared to that of the Newton-Krylov solver using SU2. From the figure it can be seen that the Newton-Krylov solver shows better performance compared to the normal setup, where the Newton-Krylov solver is able to approach convergence at approximately 930 iterations. The NUTSCFD results show however that the

Newton-Krylov solver is able to reach convergence at approximately 200 iterations, performing better than SU2. It is clear from figure 4.2b that the Newton-Krylov solver within SU2 is able to have a positive effect on the number of non-linear iterations that are required in order to reach convergence, but 4.2a shows that there is still room for improvement. The difference between the standard and Newton-Krylov solver is that only the NEWTON_KRYLOV option is turned off for the former.

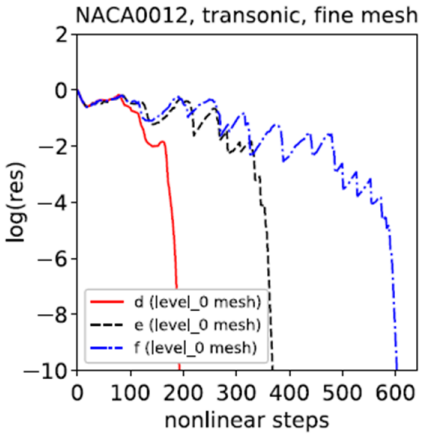


(a) NUTSCFD result following the NACA 0012 test case [1]

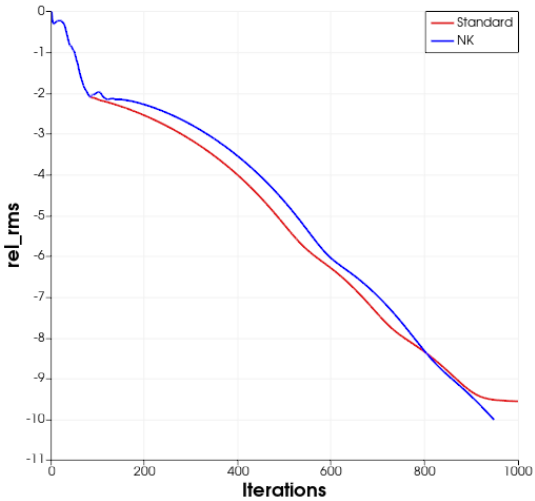


(b) SU2 result following the NACA 0012 test case

Figure 4.1: NACA 0012 Mach contour



(a) NUTSCFD convergence result following the NACA 0012 test case [1]



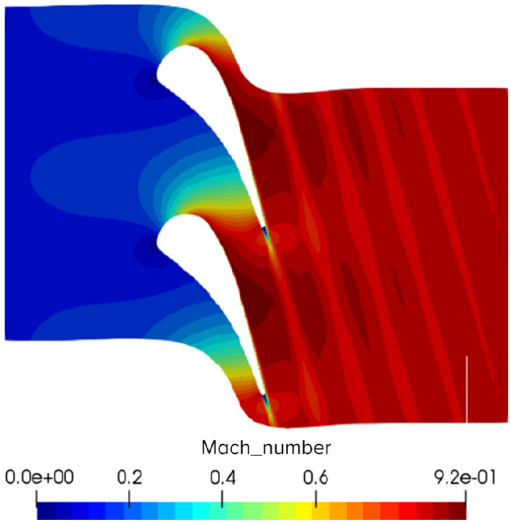
(b) SU2 convergence result following the NACA 0012 test case using the density residual

Figure 4.2: NACA 0012 convergence behaviour

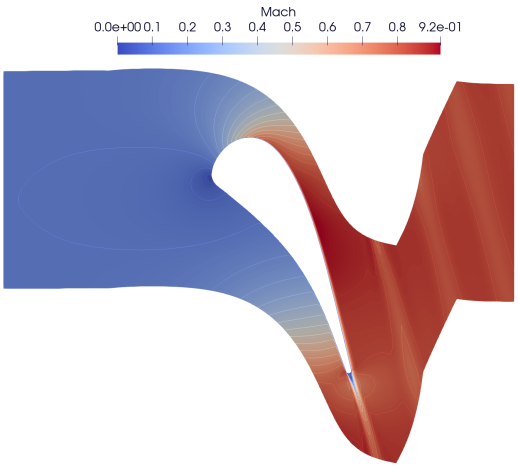
4.2. LS89 RESULTS

This section contains the analysis results of the LS89 test case. Figure 4.3 shows the results obtained for the LS89 turbine. It was found that SU2 provides the same Mach range as the NUTSCFD solver as both figures 4.3a and 4.3b show a maximum Mach number of 0.92 Mach. Similar contours have been found and plotted, where occasionally differences in contour locations can be seen. The results are deemed to be sufficiently similar, where the residual behaviour of SU2 is analysed and compared to that of the NUTSCFD solver.

Figures 4.4 and 4.5 show the residual behaviour of both SU2 and NUTSCFD. Figure 4.4 shows both the residual involving a normal setup, and by making use of the Newton-Krylov solver. In contrast to the results



(a) NUTSCFD convergence result following the LS89 test case [1]



(b) SU2 convergence result following the LS89 test case

Figure 4.3: LS89 Mach contour

found in section 4.1, the LS89 test case requires more iterations when using the Newton-Krylov solver. When comparing the results obtained by Xu et al. to that of SU2, a notable difference in convergence behaviour can be found. The data obtained by the NUTSCFD solver is generally plotted using non-linear steps, where the number of linear steps is not documented. Xu et al. do however document the total number of work units, which is defined as "the CPU time required for a single residual evaluation" [1]. Both the non-linear steps and the work units are visible in figure 4.5, where level_5 represents the relevant convergence data. SU2 does currently not provide an option in order to measure the CPU time required in order to obtain a single residual evaluation. It is therefore not possible to make a comparison based on this aspect. The iterations that have been visualized in figure 4.4 represent the SU2 'inner iterations' as the LS89 test case does not use a multizone configuration. It is however expected that these inner iterations do not account for the separate FGMRES linear solver iterations.

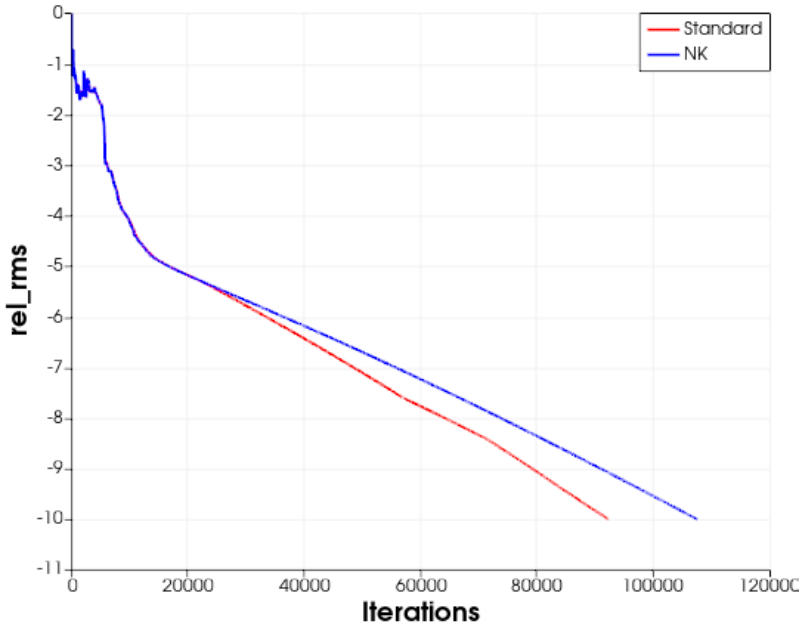


Figure 4.4: Convergence behaviour of the residual of the LS89 test case following SU2 using the density residual

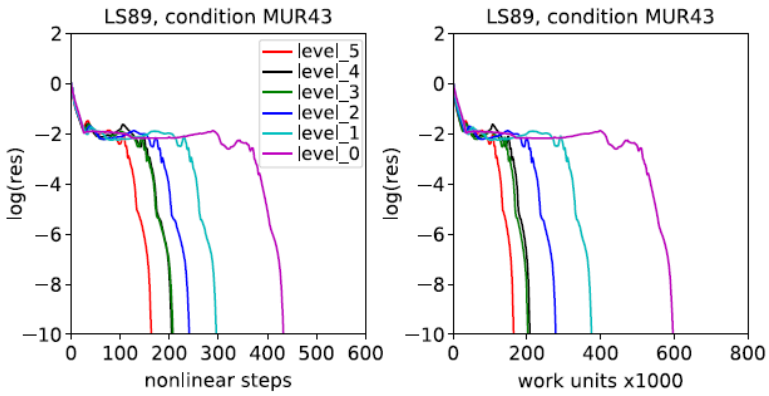


Figure 4.5: Convergence behaviour of the residual of the LS89 test case following NUTSCFD [1]

In order to analyse whether the number of linear solver iterations has an effect on the total number of SU2 Newton-Krylov 'inner iterations', a new simulation has been run where the max number of linear solver iterations has been set to 1. It was found that test case LS89 diverges if the maximum number of linear iterations is set at 3 or higher. It was tried to set the maximum number of iterations at 500, as this is the case for the NUTSCFD solver, but this resulted in a crash at iteration number 23. Figure 4.6 shows the results following a maximum of 1 linear iteration. It can be seen that the standard solver matches the number of iterations required for the Newton-Krylov solver to reach convergence. This suggests that the Newton-Krylov solver is unable to calculate multiple linear iterations, as limiting the maximum number of linear iterations to 1 results in the same convergence behaviour. When comparing the results obtained for the NACA 0012 airfoil (section 4.1), it can be seen that the Newton-Krylov solver seems to perform worse for the LS89 test case. Both LS89 simulations converge at approximately 108000 iterations, which is close to the maximum number of possible NUTSCFD iterations (if limited to 1 linear iteration). If the NUTSCFD solver would require all linear 500 FGMRES iterations, a theoretical maximum of approximately 90000 iterations would be required. It is however unclear whether the NUTSCFD solver uses more iterative techniques which could influence its final performance. A difference in performance is to be expected due to the difference in meshes used, but it is also currently unclear whether an accurate comparison can be made involving the LS89 test case. Section 4.5.2 discusses these conclusions, where solutions to these problems are proposed.

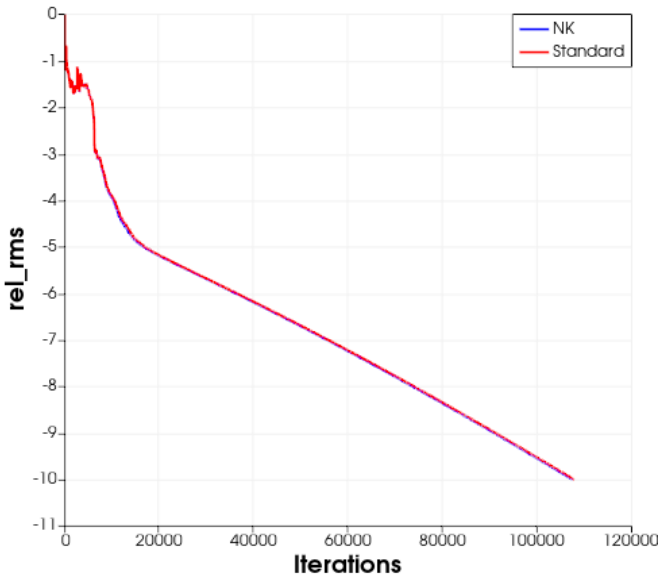


Figure 4.6: SU2 convergence behaviour of the residual of the LS89 test case with limited (1) number of linear iterations using the density residual

4.3. MTU RADIVER RESULTS

From the results of the NACA 0012 and LS89 test case and the discussion in section 4.5.2 it has been concluded that a new test case is required in order to accurately analyse SU2. The previous test cases are unable to provide a reliable analysis, where the MTU test case has been developed to solve these issues. This section contains the results for this test case, where the development of the test case has been documented in section 3.3. The results have been divided into different sections, where section 4.3.1 presents the initial findings for the MTU test case. The performance of the test cases is tested by plotting the average Block-Gauss Seidel residual. It was found that the test case stalls, where this section therefore also contains the initial analysis of this behaviour. Section 4.3.2 investigates whether potential problematic regions can be identified by plotting the residual on the mesh, where finally the different Newton-Krylov setups are investigated in section 4.3.3. The results found within section 4.3 are discussed in section 4.5.3.

4.3.1. MTU PERFORMANCE ANALYSIS

This section contains the initial results for the MTU test case. The creation of this test case has been documented in section 3.3, where the results have been analysed extensively due to poor performance. It should be noted that the results within this section are obtained using the standard solver. The Newton-Krylov solver is analysed in section 4.3.3. Different solver setups have been used in order to identify the cause of the poor behaviour. The 200k element mesh of the MTU test case has been used for these simulations, as the accuracy of the mesh is of less importance due to the solver not converging to a solution. Currently the MTU shows stalling behaviour, where the test case does not reach convergence. Applying a rotation rate to the rotating frame of the impeller blade passage resulted in SU2 to crash at low solver iterations, where therefore the frozen rotor setup has been analysed first. Figure 4.7 shows the convergence result for the MTU test case where the configuration has been set up using a frozen rotor. Figure 4.7 shows the residual versus the iterations of the SU2 solver. The 3 zones of the MTU multizone simulation have been plotted, where the residual for the inlet, blade passage and outlet are visualized using black, red and blue respectively. The plot shows oscillating behaviour for both the inlet and outlet residual, where the simulation stalls and does not improve any further past iteration 2000.

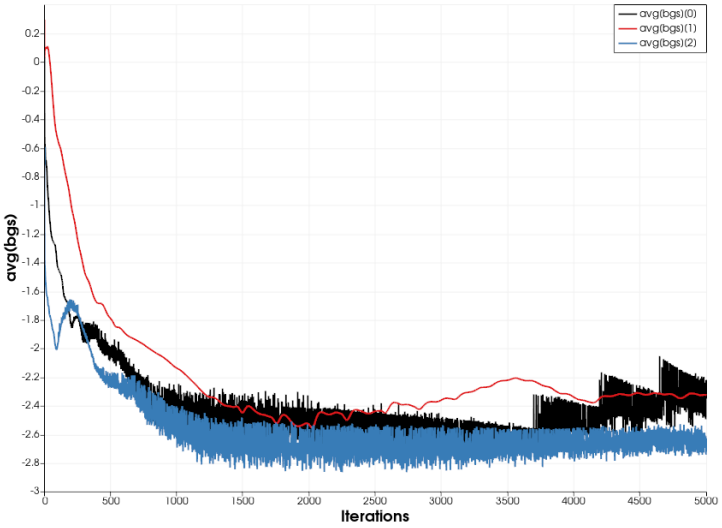
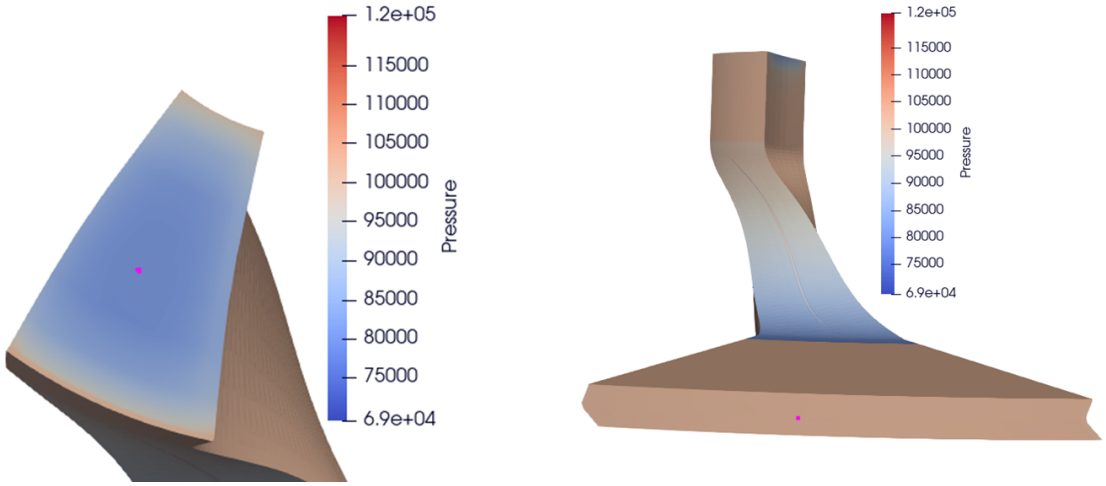


Figure 4.7: Convergence behaviour of the residual of the MTU test case following the frozen rotor setup

Following these results, the boundary conditions of the simulation have been analysed first. This was done by running the frozen rotor simulation for one iteration, making sure the inlet boundary conditions are correct. Figure 4.8 shows the pressure contours of the inlet and outlet boundary surfaces. Using Paraview static pressure measurements have been done, where the measurement location is visualized using a small purple dot. From these points it was found that the inlet value is 0.86 bar and the outlet 1.025 bar. The inlet and outlet conditions have been set following the conditions mentioned in table 3.5, setting the inlet total pressure at 0.6 bar and the outlet static pressure at 1.013 bar. The measured static pressure at the inlet is relatively high compared to the set boundary condition, but from the configuration file there is no indication that the test case has not been setup properly.



(a) Static pressure inlet distribution with measurement location

(b) Static pressure outlet distribution with measurement location

Figure 4.8: Static pressure distributions for the MTU test case at 1 iteration

The observed behaviour of the MTU has lead to the analysis of the test case on which the MTU test case has been based. The MTU test case has been created by making use of the Eckardt test case which has been developed at the TU Delft. Figure 4.9 shows the residual behaviour of the solver involving the Eckardt test case. Figure 4.9 clearly shows stalling behaviour where all residuals for the three zones oscillate. The setup that was chosen for this test case was the default setup (provided by the TU Delft), where the default mesh was used. It was expected for this simulation to reach convergence, but this is clearly not the case. This test case has been used in prior studies, where it is likely for the SU2 code base to have changed such that the Eckardt test case does no longer converge. During the current work it was initially not expected for the default Eckardt test case to stall. Realizing this fact has therefore been a time-consuming process, as the MTU test case was analyzed first, thus initially overlooking the possibility of a greater underlying issue causing this behaviour. The result of the Eckardt analysis is therefore discussed first.

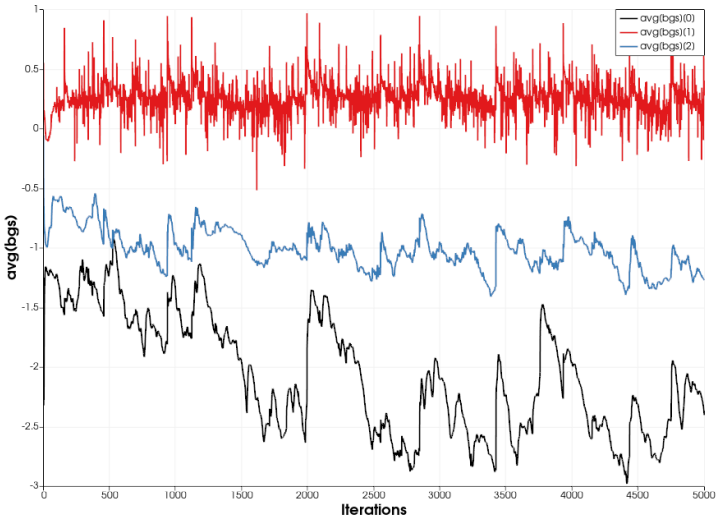


Figure 4.9: Convergence behaviour of the residual of the Eckardt test case following the default setup

Research has been done with respect to the performance of the Eckardt test case. Different setups have been tested, where the results are visible in table 4.1. The default Eckardt test case uses the JST numerical scheme in combination with the BCGSTAB linear solver. Different schemes and setups are tested in order to identify the issue currently causing the stalling behaviour. The default test case is altered, where an overview of the change in simulation setup can be seen in table 4.1.

FLUID INTERFACE + ZONE INTERFACE SETUP

The first changes that were made to the default Eckardt setup involve the boundary conditions and thus the markers in the configuration file. Different fluid interfaces have been tried, which alters the behaviour of the data that is transferred between the zones. Changes were made to the fluid interface and the marker zone interface, as the default test case made use of the mixing plane interface. This setup initially showed better performance (figure 4.10a) compared to the default test case (figure 4.9), but again showed stalling behaviour after approximately 400 iterations.

Table 4.1: Different Eckardt test case setups that have been analysed with the solver result

Simulation type Eckardt	Result
Default (JST and BCGSTAB) setup	Oscillating residual
Fluid interface + zone interface setup	Oscillating residual
Frozen rotor setup	Oscillating residual
Massflow setup	Oscillating residual
FGMRES setup	Simulation crash
ROE numerical setup	Normal behaviour

FROZEN ROTOR SETUP

A setup was tested which made use of the frozen rotor setup. The default configuration was used, where the rotating frame of the impeller was fixed. Fixing the rotor should provide a more stable simulation, but this did not improve the convergence of the Eckardt test case.

MASSFLOW SETUP

Another setup has been tested by changing the boundary conditions with respect to the massflow. The massflow setup used instead of the exit static pressure the massflow at the outlet in order to close the system of equations. This is a relatively new exit boundary condition within SU2, where this did not improve the solver convergence.

FGMRES SETUP

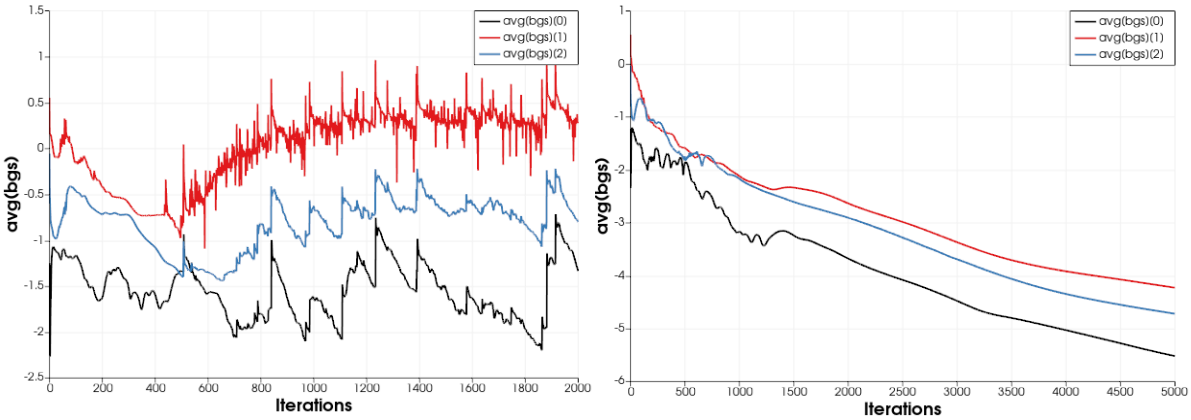
In addition to boundary condition changes, solver changes have been done as well where the FGMRES linear solver was tested. This setup was achieved by changing the default configuration, which uses the BCGSTAB linear solver, into the FGMRES solver. This did however worsen the convergence behaviour as this resulted in the simulation to crash at iteration number 409.

ROE NUMERICAL SETUP

Finally the numerical scheme was changed in order to test the convergence behaviour of the Eckardt test case. The default configuration was altered by changing the JST scheme into the Roe scheme. Changing the numerical scheme improved the convergence behaviour of the solver, where the Roe setup results in normal solver performance. The residual for the Roe setup can be seen in figure 4.10b.

The boundary conditions of the Eckardt test case are analysed similarly to the MTU. Figure 4.11 shows the inlet and outlet static pressure after one iteration of the Eckardt test case. The inlet and outlet static pressure are measured at 1.01 and 1.21 bar respectively. The inlet total pressure was set at 1.01 bar, where the outlet static pressure was set at 1.51 bar. A difference in measured pressures was therefore found, where these measurements were done using the default Eckardt configuration. Both figures 4.11a and 4.11b differ from figures 4.8a and 4.8b showing different pressure distributions, where this is expected to be caused by the difference in scale. Since the Roe setup is able to show normal solver behaviour (figure 4.10b), it is unlikely that the current setup for the boundary conditions is causing the oscillating residual behaviour for the other setups.

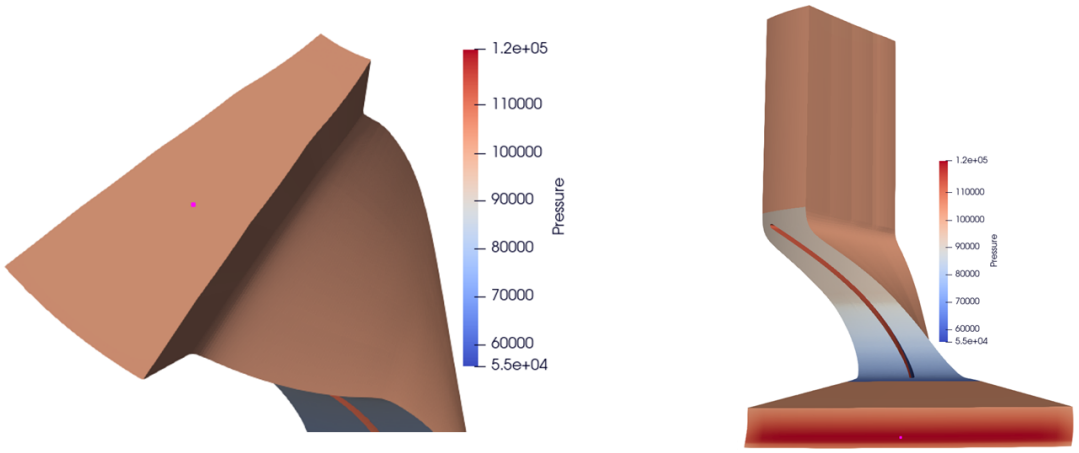
Currently the only setup that is able to converge is the setup involving the ROE numerical scheme. Due to the sake of completion the MTU has been simulated using the ROE scheme, which resulted in a diverged simulation. Further analysis has been conducted for the MTU test case, showing a similar outcome to that of the Eckardt setup analysis (table 4.1). It was found from the setup analysis that no MTU test case was able to reach convergence. Additional tests have been conducted by investigating the local Mach number on the mesh. Since Paraview is unable to plot the Mach distribution along the meridional view of the impeller, slices



(a) Convergence behaviour of the residual of the Eckardt test case following the fluid interface setup

(b) Convergence behaviour of the residual of the Eckardt test case following the Roe setup

Figure 4.10: Convergence behaviour of the residual of the Eckardt test case for two setups following table 4.1



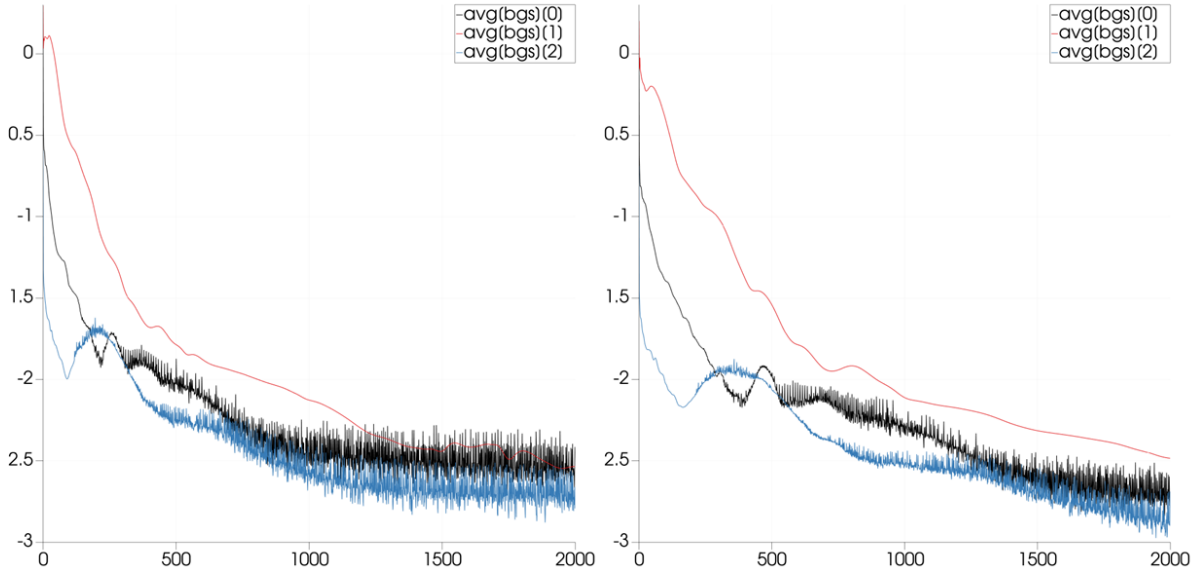
(a) Static pressure inlet distribution with measurement location

(b) Static pressure outlet distribution with measurement location

Figure 4.11: Static pressure distributions for the Eckardt test case at 1 iteration

of the Mach distribution have been inspected instead. Although not ideal, these slices did not show unusual Mach behaviour of the impeller except for high Mach numbers being present at the tip-gap of the blade. The Mach results that have been inspected involve the stalling results obtained for the default MTU test case at iteration 2000. The CFL number has also been inspected in order to test whether this had an effect on the magnitude of the oscillating residual. The result of this analysis can be found in figure 4.12. Comparing figures 4.12a and 4.12b shows that reducing the CFL number resulted in less oscillatory behaviour. The solver has been configured such that the CFL number is non-adaptive.

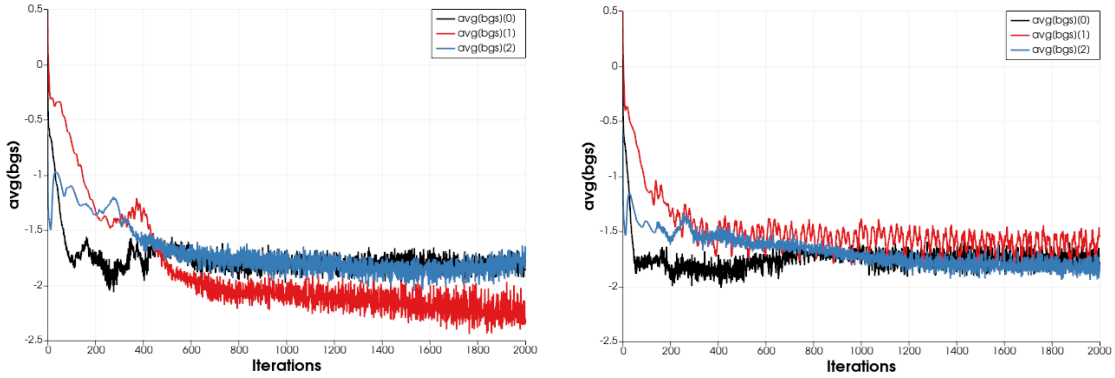
Finally an analysis has been done with respect to a no tip-gap setup. The default Eckardt mesh does not contain a tip-gap region, and due to the analysis of the Mach contours, it was decided to investigate the behaviour of the MTU without a tip-gap region. The mesh has been altered in Turbogrid, where the default MTU setup has been used. Results of the analysis can be seen in figure 4.13. Due to the sake of completion the ROE setup has been tested as well. Figure 4.13a shows the residuals with respect to the default test case, where again the MTU shows stalling behaviour. Figure 4.13b shows the residual plot with respect to the ROE setup. Unlike figure 4.10b, combining the MTU test case with the ROE scheme results in the simulation stalling. It was expected for the MTU to show similar behaviour to that of the Eckardt compressor, but this is not the case. Instead the MTU simulation stalls at approximately iteration 100.



(a) Residual behaviour for the MTU test case with a CFL number of 10

(b) Residual behaviour for the MTU test case with a CFL number of 5

Figure 4.12: Analysis of the residual with respect to the CFL number
(using the fluid interface + zone interface setup with a frozen rotor)

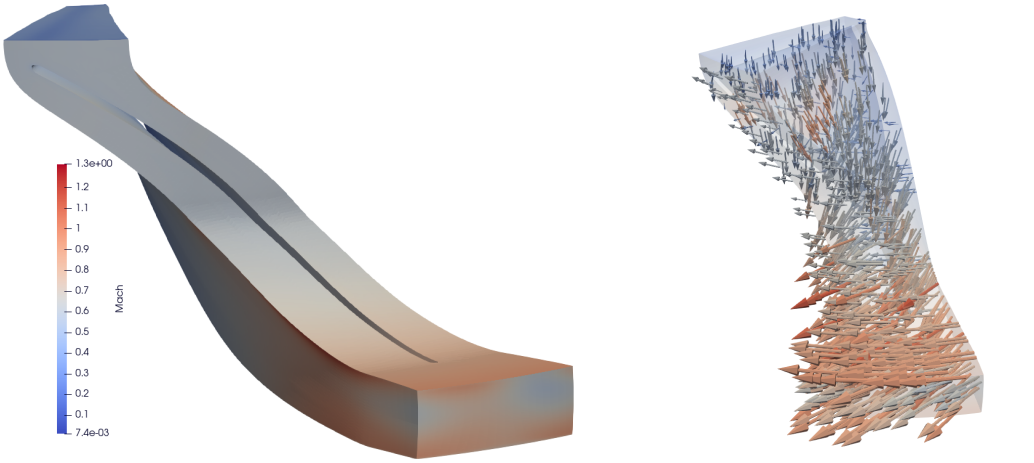


(a) Residual behaviour with respect to the default test case

(b) Residual behaviour with respect to a setup using the ROE scheme

Figure 4.13: Analysis of the residual for the no tip-gap MTU test case

Within the current work it is concluded that the test case developed for the MTU impeller provides stalling results. Analysing the results provided insight into the fact that the reference test case of the Eckardt compressor, on which the MTU test case is based, also provides stalling results. Research has been conducted on both test cases, where an explanation for the current behaviour was not found. This outcome has therefore compromised the ability of the current study to analyse the performance of the Newton Krylov solver in SU2. The only centrifugal test case that was able to reach converges is the Eckardt test case following the Roe setup (figure 4.10b), where the results of the test case can be seen in figure 4.14. Paraview has been used in order to post-process the data, where Paraview is currently unable to plot the meridional view of the impeller. Figure 4.14a shows the Mach contour, where figure 4.14b shows the velocity field of the test case. The scale of the arrows is proportional to the momentum of the flow, where the colour of the arrows indicates the Mach number. Additional studies have been conducted with respect to the mesh quality of the MTU. The results of these studies can be found in section 4.3.2. In addition, the results found within the current section are discussed in section 4.5.3.



(a) Mach contour of the Eckardt test case

(b) Velocity field of the Eckardt test case

Figure 4.14: Eckardt results following the Roe setup

4.3.2. ANALYSIS OF THE MTU RESIDUAL ON THE GRID

A grid analysis has been done with respect to the MTU test case in order to determine why the MTU test case did not reach full convergence. This section contains the results for this analysis where the study has been conducted using the residual of the results. This section is divided into 4 sections which discuss the residual with respect to the density, energy, momentum and nu tilde. The results have been plotted on the mesh (figure 4.15), visualizing the top 100 highest residual locations. In addition the residual distribution of the mesh is evaluated, where the largest deviating values are analyzed. This has been done in order to better identify problematic zones, providing a broader view of the performance of the mesh. All results have been obtained at iteration 2000 since the MTU test case is unable to reach convergence (figure 4.7). The study has been conducted using the frozen rotor setup.

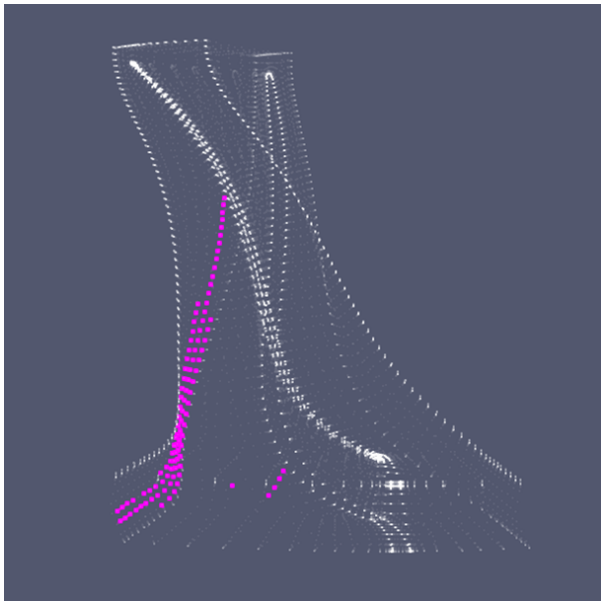
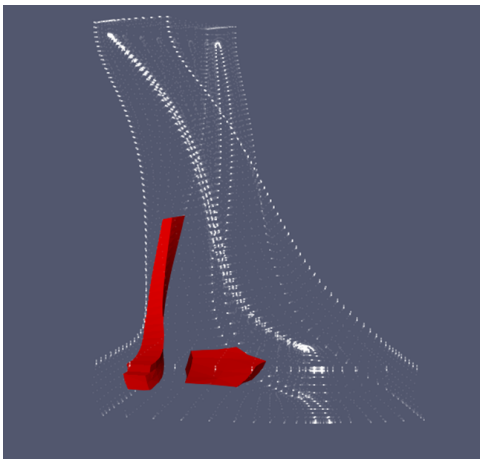


Figure 4.15: Top 100 points with the highest density residual

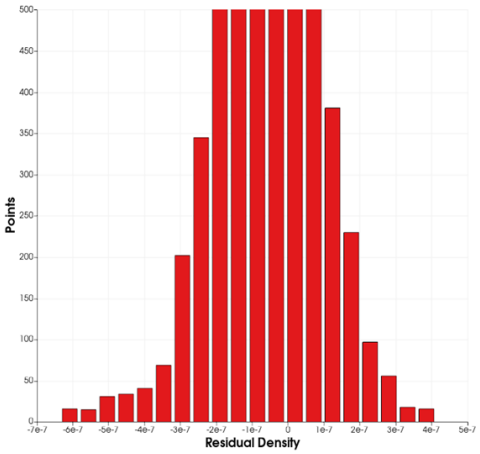
DENSITY RESIDUAL

Figure 4.15 shows the top 100 points with the highest density residual for the MTU test case. The gridpoints have been visualized in white, where the top 100 points are indicated using purple. The results range from

$-6.36e^{-7}$ to $-4.20e^{-7}$, where the points are mostly located near the left periodic boundary condition of the mesh. Figure 4.16 shows the results for the residual distribution analysis. Figure 4.16a visualizes the most extreme residual values, where the residual distribution versus the number of points has been plotted in figure 4.16b. From the residual distribution (figure 4.16b) it can be seen that a relatively low number of points is located at the edges of the distribution, where the two columns deviating the most have been visualized in figure 4.16a. It can be seen that the location of these deviating values matches the locations in figure 4.15. It should be noted that the residual distribution (figure 4.16b) ranges only up to 500 points. This was done for the sake of clarity, where most of the points are located around 0. The bar graphs around 0 therefore extend past the maximum value of 500.



(a) Location of the most extreme density residuals



(b) Distribution of the density residual

Figure 4.16: Results for the density residual distribution analysis for the MTU test case

MOMENTUM RESIDUAL

The top 100 points with the largest momentum residual have been plotted in figure 4.17. The top 100 points range from $1.47e^{-3}$ to $0.57e{-}3$, where they have been located at the surface of the mesh (at the shroud). The error of these points is significantly greater compared to that of the density residual but differs in location.

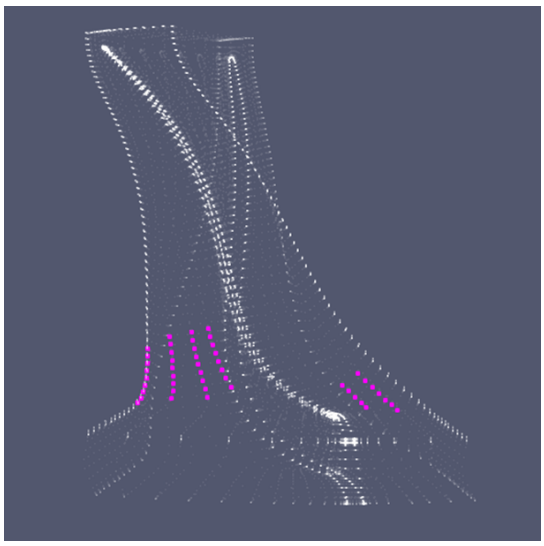
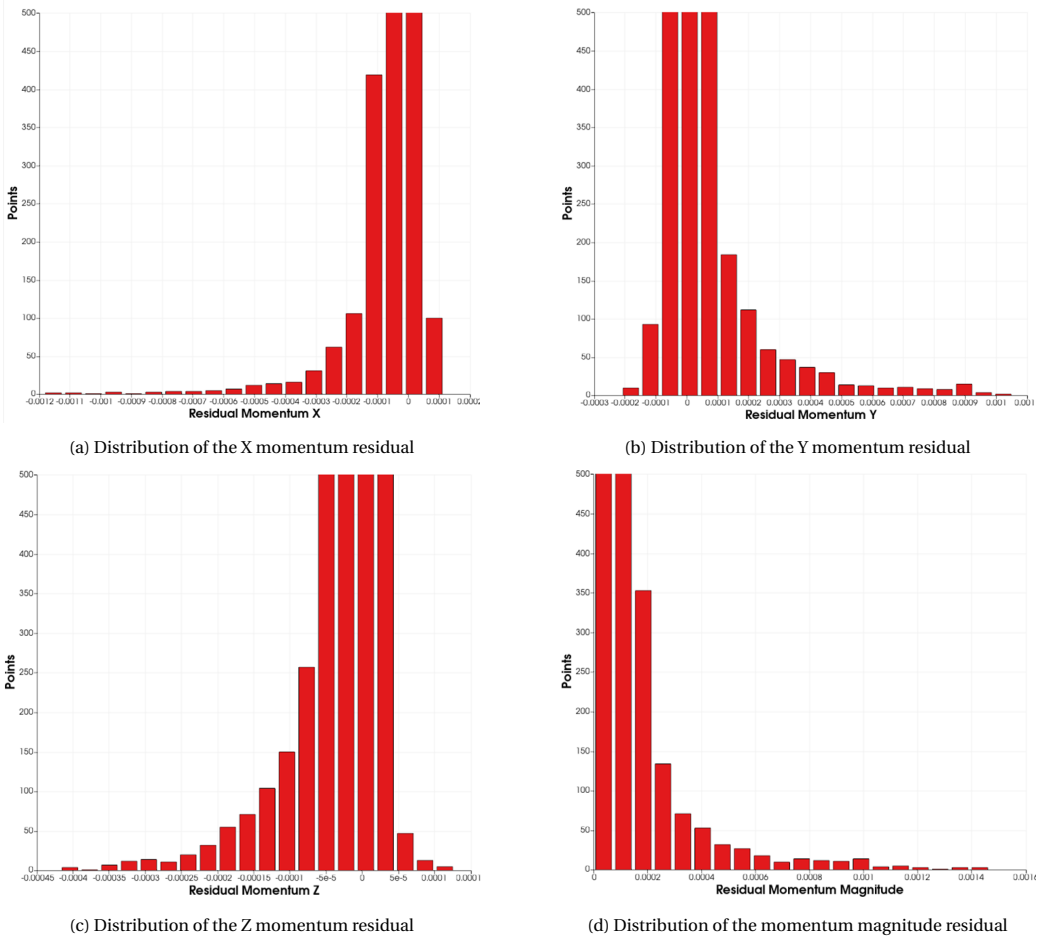


Figure 4.17: Top 100 points with the highest momentum residual

The residual distributions have been plotted in figure 4.18. From these distributions it can be seen that the X momentum provides the largest deviating residual. Although the number of points that deviate from 0 is relatively low, the scale of these distributions is significantly greater compared to the distribution of the

density residual (figure 4.16b). The effect of the momentum residual is therefore likely to affect the SU2 simulation more greatly.



(a) Distribution of the X momentum residual

(b) Distribution of the Y momentum residual

(c) Distribution of the Z momentum residual

(d) Distribution of the momentum magnitude residual

Figure 4.18: Results for the X, Y, Z and magnitude momentum residual distribution analysis for the MTU test case

ENERGY RESIDUAL

Figure 4.19 shows the top 100 points with the largest energy residual. These points range from 0.11 to 0.05, indicating relatively large error values compared to that of the density and the momentum. The top 100 points for the energy residual are more spread out compared to that of the density, where it is similar to that of the momentum, being positioned at the top surface of the grid. Figure 4.20 shows the results for the residual distribution. Figure 4.20b shows a low number of points that deviate from the value of 0. It should be noted that these values differ more significantly due to the increase in scale, compared to the previously plotted residual distributions. The strongest outliers following this distribution have been plotted in figure 4.20a.

NU TILDE RESIDUAL

The final residual that is analysed involves the Nu tilde. The top 100 points involving the nu tilde residual have been plotted in figure 4.21. The figure shows the points being positioned at the shroud near the trailing edge of the blade. This location differs significantly compared to the previous results that have been plotted for the density, momentum and energy residual. Figure 4.22 shows the results involving the residual distribution. Figure 4.22b shows little deviating residual values, where the largest deviating points are located at a similar location to that of the top 100 points (figure 4.22a).

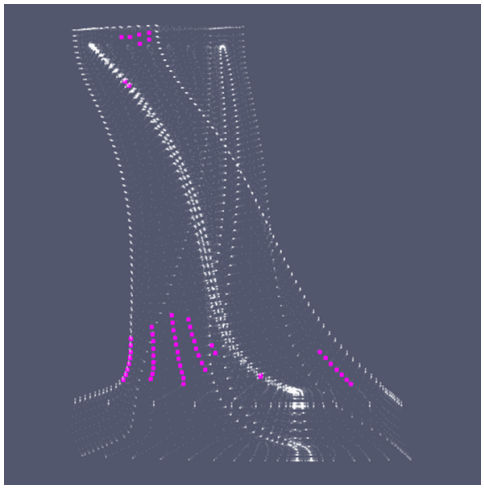
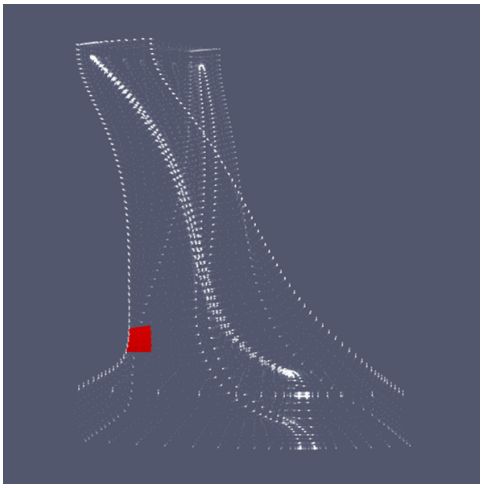
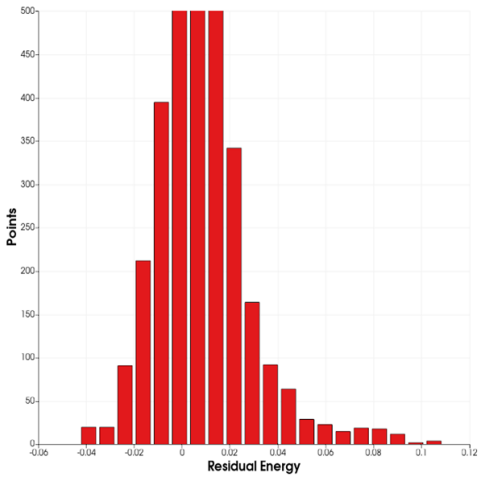


Figure 4.19: Top 100 points with the highest energy residual



(a) Location of the most extreme energy residuals



(b) Distribution of the energy residual

Figure 4.20: Results for the energy residual distribution analysis for the MTU test case

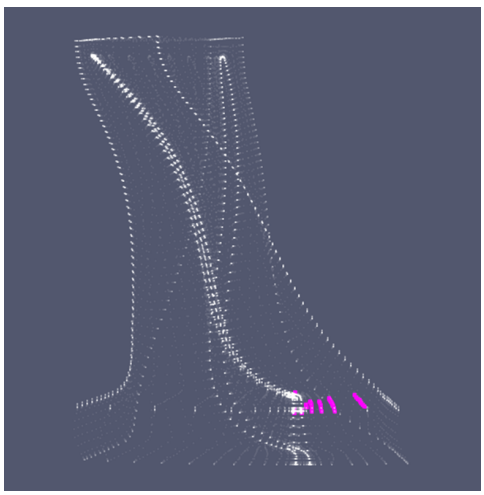
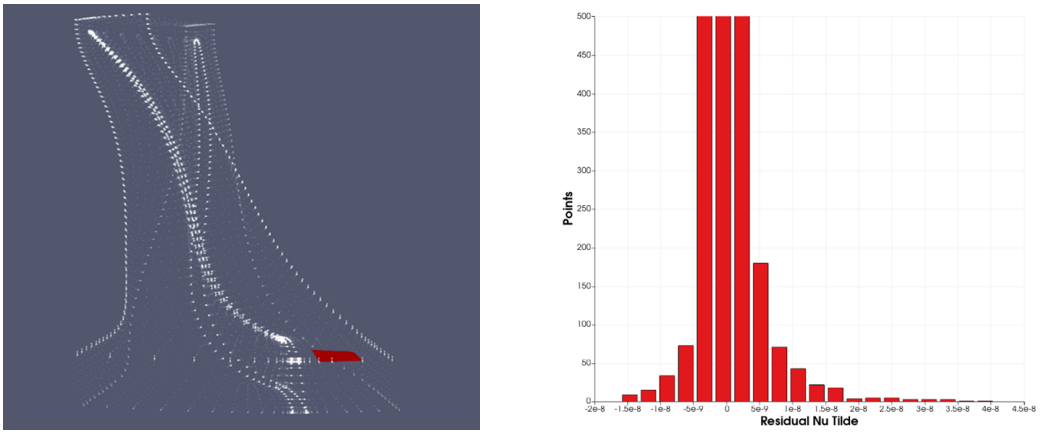


Figure 4.21: Top 100 points with the highest nu tilde residual



(a) Location of the most extreme nu tilde residuals

(b) Distribution of the nu tilde residual

Figure 4.22: Results for the nu tilde residual distribution analysis for the MTU test case

Overall it can be concluded that the largest deviating residuals can be found for the energy residual. This residual is orders of magnitude higher compared to the other residuals. The analysis does however not provide a clear indication of a problematic zone. The top 100 points for the energy residual are spread out, where the other residuals do not provide a clear matching region. Although the location of the highest momentum residual points is similar to that of the energy residual, for there to be a problematic region it would be expected for these points to be more concentrated. It would also be expected that the other density and nu tilde residuals would provide similar results for there to be a problematic region. Following these results, a clear reason for the current stalling behaviour of the MTU was therefore not identified.

4.3.3. MTU NEWTON-KRYLOV PERFORMANCE STUDY

It was found in section 4.3.1 that the MTU test case is unable to reach convergence following the standard solver setup. The results showed that the MTU test case stalls, which complicates the analysis of SU2. Following these results it is therefore not possible to compare SU2 to NUTSCFD using the MTU test case. Nonetheless the current section provides an analysis of the Newton-Krylov solver with respect to different solver setups. Following the theoretical background (chapter 2) it was found that the Newton-Krylov method is impacted by the preconditioner and the formation of the Jacobian matrix. The current section therefore provides an analysis of SU2's performance using different preconditioners, and different solver settings affecting the Jacobian matrix. It should be noted that the results within the current section are obtained using the Newton-Krylov solver, which was not the case for the preceding MTU sections. The setup that was chosen for the current section involves the default MTU configuration (table 3.6). This configuration makes use of the zone + fluid interface.

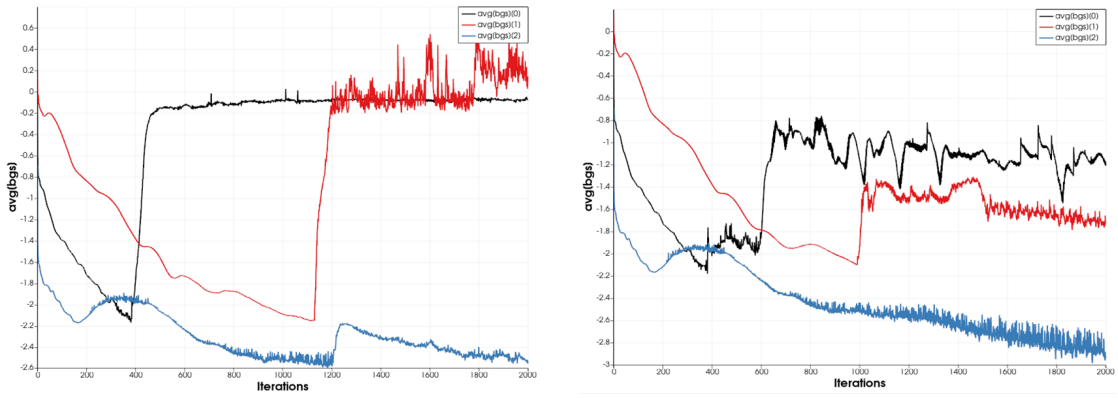
PRECONDITIONER STUDY

This section contains the results for the preconditioner study for the MTU test case. Following the literature study (section 2.3.2) it was determined that the preconditioner can have a significant effect on the performance of the Newton-Krylov solver. Different preconditioners have therefore been analysed, where the results of this study are presented within this section. SU2 allows for the selection of different preconditioners, where the different types of preconditioners are described in table 4.2. The MTU test case has been used with the Newton-Krylov solver being activated, where the results within the current section are obtained at iteration 2000. This was chosen as previous simulations showed the MTU test case to stall.

Table 4.2: Different preconditioner options in SU2 and their description

Preconditioner type	Description
JACOBI	Block Jacobi preconditioner
LU_SGS	Lower-Upper Symmetric Gauss-Seidel
ILU	Incomplete Lower Upper factorization with connectivity-based sparse pattern
LINELET	Line-implicit Jacobi preconditioner

Figure 4.23 shows the convergence behaviour of the SU2 Newton-Krylov solver when using a LUSGS and Jacobi-based preconditioner. Within these figures the residuals of the inlet, passage and outlet section of the multizone simulations are visualised using black, red and blue respectively. Both simulations show a sudden sharp increase in residual error for both the inlet and passage section. From the figures it is concluded that the Jacobi-based preconditioner (figure 4.23b) performed slightly better compared to the LUSGS preconditioner (figure 4.23a), where both simulations stall. Figure 4.24 shows the performance of SU2 using the Linelet preconditioner. This figure shows better performance for the blade passage section, with a sharp increase in residual error for the inlet section at iteration 400. It can also be seen that the outlet section performed similar to that of the Jacobi preconditioner.



(a) Residual behaviour with the LUSGS preconditioner

(b) Residual behaviour with the Jacobi preconditioner

Figure 4.23: MTU convergence behaviour of the Newton-Krylov solver using a LUSGS and Jacobi preconditioner

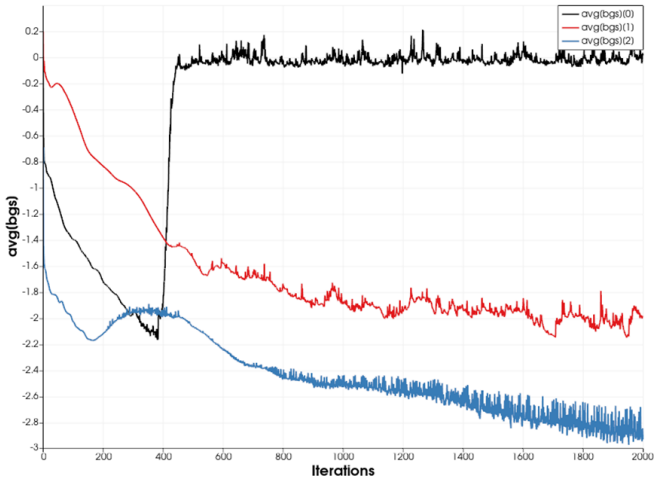
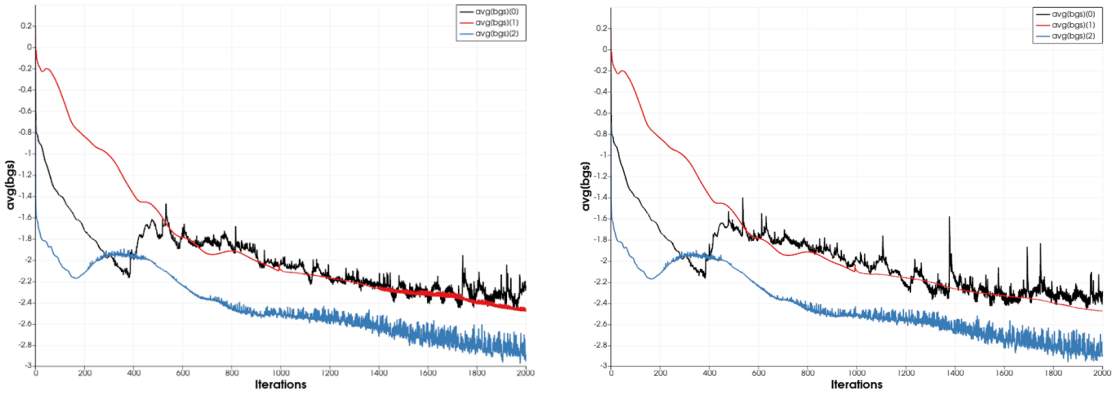


Figure 4.24: MTU convergence behaviour of the Newton-Krylov solver using a Linelet preconditioner

Figure 4.25 shows the performance of the SU2 Newton-Krylov solver when using the ILU preconditioners. Both ILU preconditioners show better performance compared to the results of the LUSGS, Jacobi and Linelet preconditioners. It should be noted that Xu et al. [1] also used an ILU preconditioner for their Newton-Krylov solver (section 2.3.2). Both figures 4.25a and 4.25b show no sudden increases in residual error, unlike seen in figures 4.23 and 4.24. The ILU(0) preconditioner seems to cause less fluctuations in residual error involving the inlet zone compared to the ILU(1) preconditioner. In contrast an oscillation can be seen for the passage section, which is not present for the ILU(1) result. The residual for the exit zone behaves similar for both ILU preconditioners. From the preconditioner study it has been found that both ILU preconditioners provide the best result in combination with the Newton-Krylov solver. It should be noted that all simulations stall, where the selection of the preconditioner did not have an effect on the final stalling result. The ILU preconditioners showed a significant difference in performance, where the LUSGS, Jacobi and Linelet preconditioners showed

large sudden increases in residual error. For the current test case it would therefore be recommended to use the ILU preconditioner when using the Newton-Krylov solver in SU2.



(a) MTU residual behaviour with the ILU(0) preconditioner

(b) MTU residual behaviour with the ILU(1) preconditioner

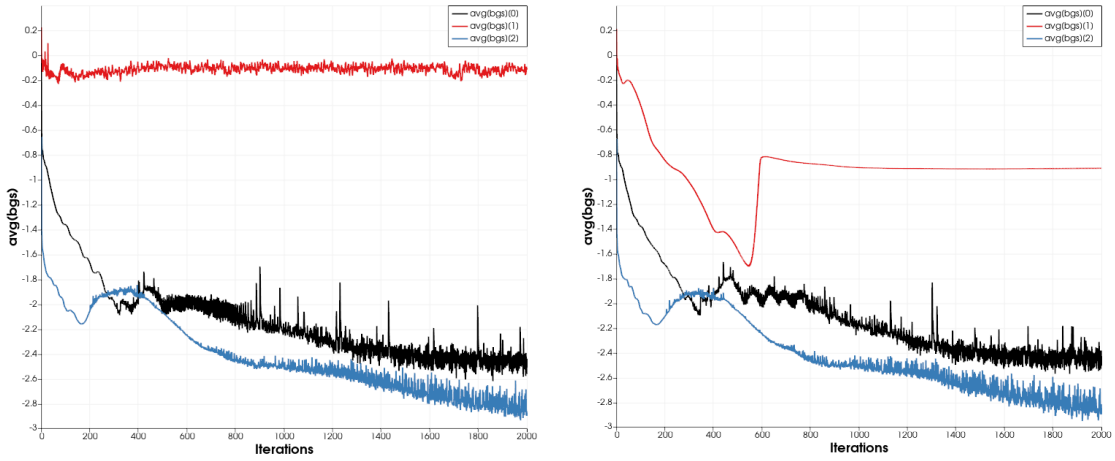
Figure 4.25: MTU convergence behaviour of the Newton-Krylov solver using a ILU(0) and ILU(1) preconditioner

JACOBIAN MATRIX SETTINGS ANALYSIS

SU2 provides additional settings which might affect the performance of the Newton Krylov solver. The effect of these settings on the residual with respect to the MTU test case is presented within the current section. The settings are related to the Jacobian matrix, which plays an important role within the Newton-Krylov solver (section 2.5.2). Xu et al. use a graph coloring algorithm, which is not present in SU2. Instead the effect of the settings '*USE_ACCURATE_FLUX_JACOBIANS*' and '*CENTRAL_JACOBIAN_FIX_FACTOR*' are analysed. The setting *USE_ACCURATE_FLUX_JACOBIANS* affects the AUSM+up(2) and SLAU(2) convective schemes that can be used in order to calculate the Jacobian matrix. Analysing the residual resulted in a plot which did not differ from the original setup. The setup case for the ILU(0) analysis has been used as the original setup (figure 4.25a), where the setting of the *USE_ACCURATE_FLUX_JACOBIANS* had no effect on the behaviour of the solver. The default setting is 'NO', where 'YES' was used in order to activate this component for the current analysis.

The Jacobian fix factor affects the numerical properties (diagonal dominance) of the global Jacobian matrix. It has been stated that the optimum is from 3 to 4 (central schemes), where for the current analysis 2, 3, 4 and 5 are considered. Figure 4.26 shows the results for the different fix factors. Figure 4.26a shows the poorest performance, where the residual of the passage section of the impeller does not improve over time. In contrast figure 4.26b shows a decreasing passage residual, but a sudden increase in error at approximately 600 iterations. It should however be noticed that this passage residual does not show any oscillatory behaviour, where this is commonly found for the other figures plotted within the current section. Figure 4.26c shows the convergence behaviour of the solver for a fix factor of 4. This result is identical to that of the result found for the ILU(0) analysis (figure 4.25a). Figure 4.26d shows the result for the fix factor of 5. It shows initially better performance with respect to the inlet section, where the residual increases at approximately 800 iterations.

From the fix factor analysis is can be concluded that the fix factor has very little effect on the outlet section of the MTU test case. All figures show nearly identical behaviour following this outlet section. The fix factor mainly affects the inlet and passage sections of the mesh, where the fix factor of 4 is able to provide the best result. It should be noted that 4 is also the default value for the fix factor in SU2, where all previous results documented within the current work are obtained using this fix factor. The passage section of the test case is strongest influenced by the fix factor, where lower factors result in poorer solver performance. The opposite is true for the inlet section, where a lower fix factor obtains better results with respect to the inlet residual. All results obtained show stalling behaviour, where the fix factor did not improve the oscillatory behaviour of the solver.



(a) Residual behaviour with the fix factor at 2

(b) Residual behaviour with the fix factor at 3

(c) Residual behaviour with the fix factor at 4

(d) Residual behaviour with the fix factor at 5

Figure 4.26: MTU convergence behaviour of the Newton-Krylov solver with respect to a Jacobian fix factor of 2, 3, 4 and 5

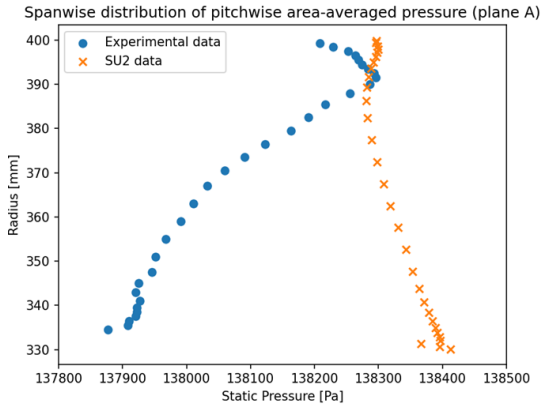
4.4. ETH 1.5 STAGE TURBINE RESULTS

This section contains the results for the ETH 1.5 stage turbine test case described in section 3.4. The results obtained involve both the 225k and 900k TUD meshes that have been the result of the test case development process. It was not possible to use the NPU mesh, where the 900k TUD mesh was created instead. The 225k and 900k TUD meshes differ in the number of elements used and the presence of the tip-gap. It should be noted that the 225k TUD mesh does not contain the tip-gap, where the 900k TUD mesh does.

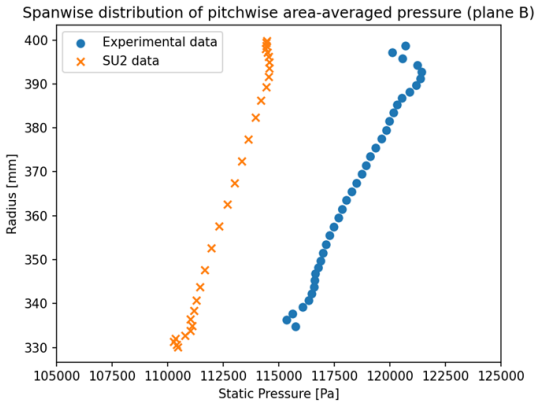
The section starts with the analysis of the 225k TUD test case using the standard SU2 solver (section 4.4.1). The results are obtained using SU2, where the exported values are area-averaged using the span-wise location of the data points. All data is compared to the experimental results obtained by Behr [21]. This section is therefore used in order to validate SU2's output, analysing whether the test case has been set up properly. This section is followed by section 4.4.2 showing the results for the 900k TUD mesh. Similar to section 4.4.1 the results are compared to the experimental data, where additional data has been obtained following research conducted by the Northwestern Polytechnical University. Both data sets allow for the comparison of SU2 to that of CFX, Numeca and the open literature, where again the test case is validated using the standard solver. The section concludes with section 4.4.3 which analyses the ETH test case using the Newton-Krylov solver. The convergence behaviour of the test case is shown, where additional tests are conducted using different preconditioners. Similar to section 4.3.3, the effect of the Jacobian configuration is evaluated as well. The results found within this section are discussed in sections 4.5.4.

4.4.1. 225K TUD MESH RESULTS

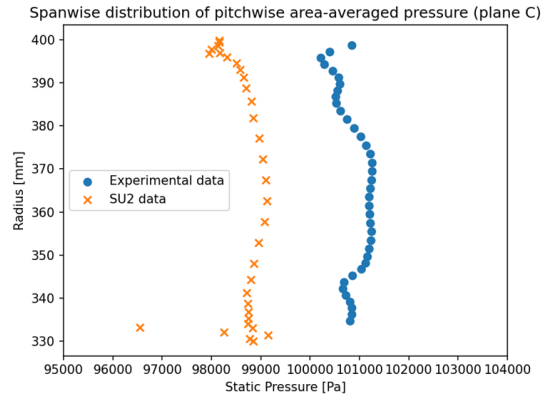
The results that have been obtained using the 225k TUD mesh are presented within this section. Results have been obtained for both the static pressure and the Mach number at the measurement plane locations (figure 3.7). The obtained results are compared to the experimental data obtained by Behr in order to verify the results obtained by SU2. Figure 4.27 shows the performance of the 225k mesh with respect to the static pressure for all 4 measurement planes. All figures show a strong deviation with respect to the experimental data. Only figures 4.27b and 4.27c show similar trends to that of the open literature, but differ in magnitude. SU2 was able to reach convergence, where the results are obtained following a residual smaller than 10^{-8}.



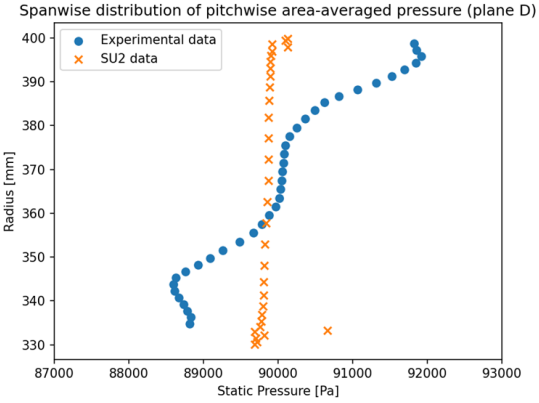
(a) Static pressure distribution at Plane A



(b) Static pressure distribution at Plane B



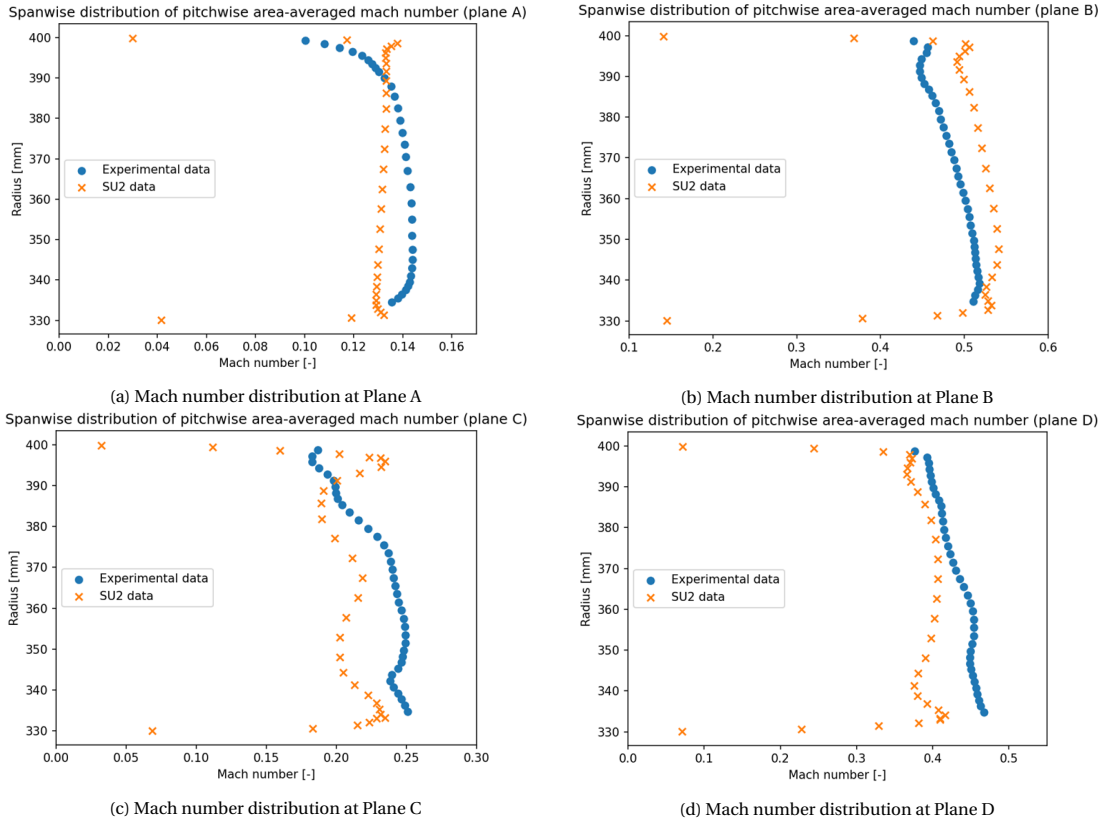
(c) Static pressure distribution at Plane C



(d) Static pressure distribution at Plane D

Figure 4.27: SU2 static pressure results for the 225k TUD test case for Plane A, B, C and D

Figure 4.28 shows the Mach distributions of the results for all 4 measurement planes. Unlike the static pressure, the Mach number shows more similarities with respect to the magnitude of the data. The values of the Mach number deviate however from the experimental data with respect to all figures. Figure 4.28c shows a strong deviation in trends near the shroud region. This strong difference in trend is expected to be the result of the missing tip-gap region. All results obtained within this section show little similarities to that of the experimental data obtained by Behr. It was expected for the data to show similar trends, where instead strong deviations are found. Interesting to notice is the difference with respect to the data involving measurement plane A (especially figure 4.27a). The SU2 results show different behaviour where these trends are unlikely to be caused by the missing tip-gap. From the results it can be concluded that the created mesh is not accurate enough in order to compare the ETH performance, especially with respect to the experimental data. The strongest indication for this is the result obtained involving the Mach number at plane C (figure 4.28c). This plot shows a slightly similar trend with respect to the spanwise distribution, where a strong deviation is seen near the shroud of the turbine. This is likely to be the result of the deviating tip-gap region, where the effect of this is travelling downstream of the rotor section. The results obtained within the current section strongly suggest the creation of a new test case which has a tip-gap. This has therefore led to the development of the second (900k) ETH mesh. The procedure of which is described and discussed within section 3.5.5.



(a) Mach number distribution at Plane A

(b) Mach number distribution at Plane B

(c) Mach number distribution at Plane C

(d) Mach number distribution at Plane D

Figure 4.28: SU2 Mach results for the 225k TUD test case for Plane A, B, C and D

4.4.2. 900K TUD MESH RESULTS

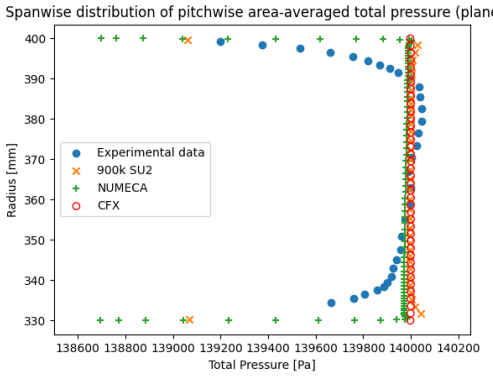
This section contains the ETH results that are obtained using the 900k TUD mesh. From section 4.4.1 it has been concluded that a mesh without the tip-gap region is not sufficient in order to study the ETH test case. The 900k TUD mesh has therefore been generated in order to obtain more accurate results. This section shows all results for the 900k mesh with respect to the total pressure, static pressure, the Mach number and the relative flow velocity. All values have been area-averaged and distributed along the span of the blade, where the results are compared to the results obtained by CFX, Numeca and the experimental data by Behr. No convergence data was provided with respect to residual behaviour, where the convergence behaviour of SU2 is therefore not analysed within the current section.

PLANE A

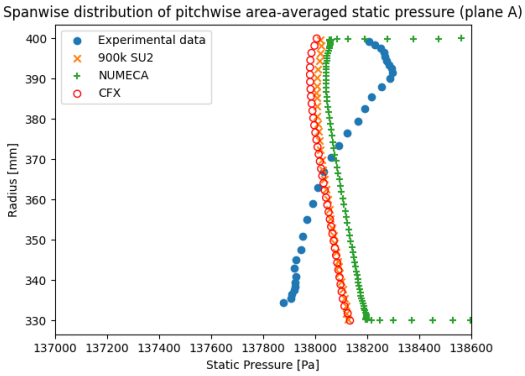
The results obtained for measurement plane A are plotted in figure 4.29. All plots show strong similarities between the CFD data. Both SU2, CFX and Numeca show similar trends, where SU2 matches the results obtained by CFX. Interesting to notice is the significant difference in performance with respect to the experimental data. Although figures 4.29a, 4.29c and 4.29d show similar trends, the plots show a difference in the exact value. Figure 4.29b shows an even larger difference in values, where the trend of the experimental data differs greatly from the CFD results.

PLANE B

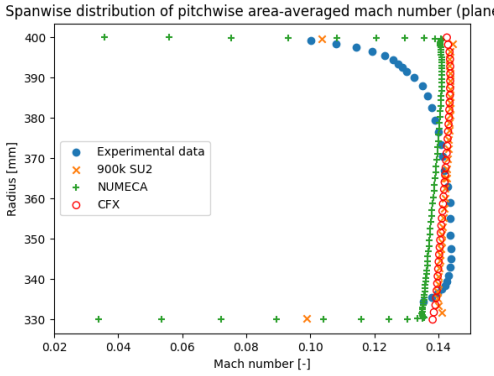
Figures 4.30 and 4.31 show the results that have been obtained for measurement plane B. With respect to the results obtained for plane A, the SU2 results differ more significantly from the CFX data. From figures 4.31a and 4.31b it can be found that again SU2 is able to show similar trends with respect to the data obtained by CFX, where figure 4.31b shows a small anomaly near the radius of 385 mm. This bump is however also seen in the experimental data near a 385 mm radius. Figure 4.30a shows a strong loss in total pressure, but a similar trend with respect to the other results. A slight drop in total pressure over stator 1 is to be expected when compared to figure 4.29a, but the pressure drop shown in figure 4.30a for SU2 is rather significant. Figure 4.30b shows SU2 outputting similar results to that of CFX, where a stronger static pressure loss is found near the tip of the blade compared to CFX.



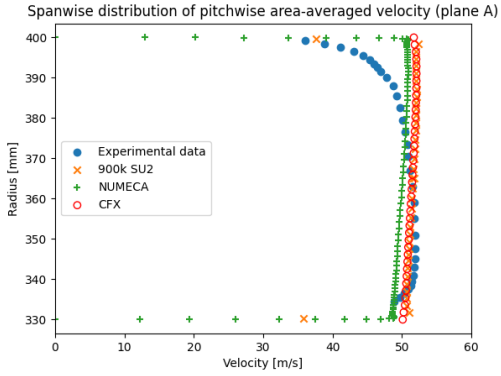
(a) Total pressure distribution at Plane A



(b) Static pressure distribution at Plane A

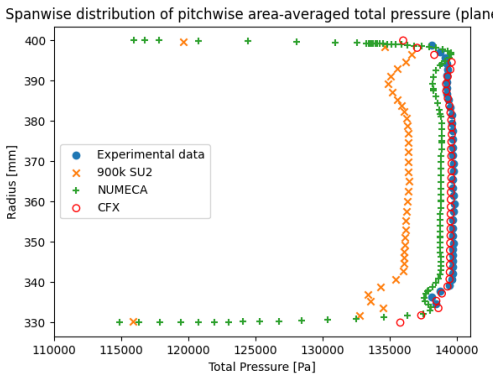


(c) Mach number distribution at Plane A

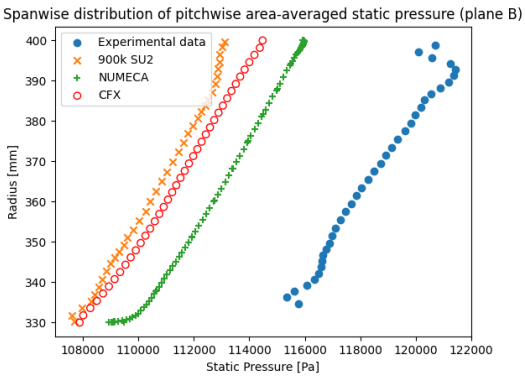


(d) Relative velocity distribution at Plane A

Figure 4.29: SU2 pressure, Mach number and velocity results for the 900k TUD test case for Plane A



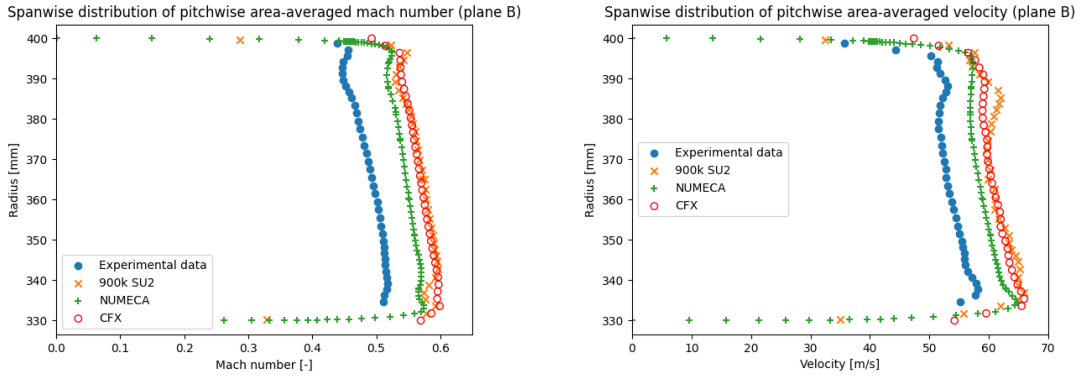
(a) Total pressure distribution at Plane B



(b) Static pressure distribution at Plane B

Figure 4.30: SU2 pressure results for the 900k TUD test case for Plane B

All plots involving measurement Plane B show again deviating results with respect to the experimental data. Only the results following the total pressure show similar values compared to CFX and Numeca. The other figures show similar trends when compared to the experimental data, but differ in exact values. It should be noted that these trends for measurement plane B follow the experimental data more closely, where this was not the case for plane A.



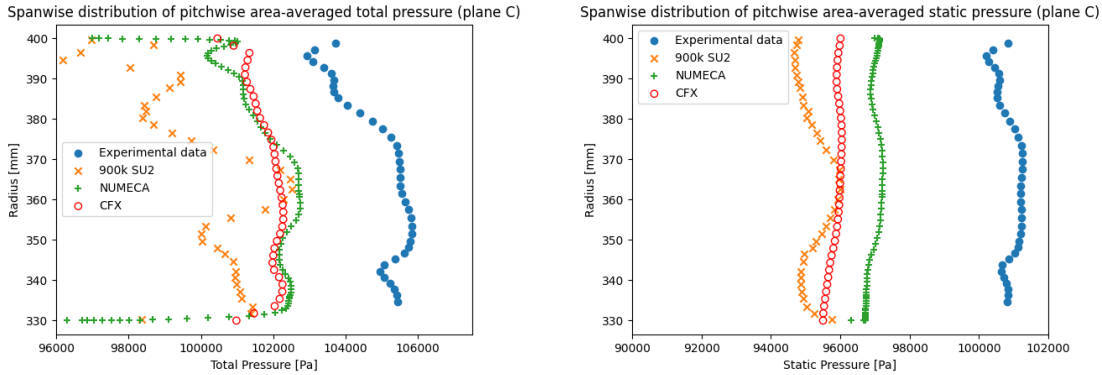
(a) Mach number distribution at Plane B

(b) Relative velocity distribution at Plane B

Figure 4.31: SU2 Mach number and velocity results for the 900k TUD test case for Plane B

PLANE C

Figures 4.32 and 4.33 show the results obtained for measurement Plane C. Measurement plane C is located at the exit of the rotor section, where these results are therefore strongly influenced by the performance of the rotor. Figures 4.32a, 4.32b and 4.33a show large fluctuations in SU2 behaviour compared to the other results. It can be seen that SU2 follows the trends of the experimental data slightly, as an increase in experimental values also translates roughly to an increase in SU2 (local) overall value. SU2 also shows however large fluctuations which is neither found in the experimental data nor the CFX and Numeca results. Figure 4.33b shows more similarities between SU2 and the other results. Similar trends are for example found near the hub and shroud of the turbine following SU2, CFX, Numeca and the experimental data. It is currently unknown why SU2 provides such different data compared to the other data sets, especially when looking at the results found for measurement plane D. SU2 does however provide similar results in the sense that the values are within range of the other data sets. Interesting to notice is that also for Plane C, the CFD results do not match the experimental data.



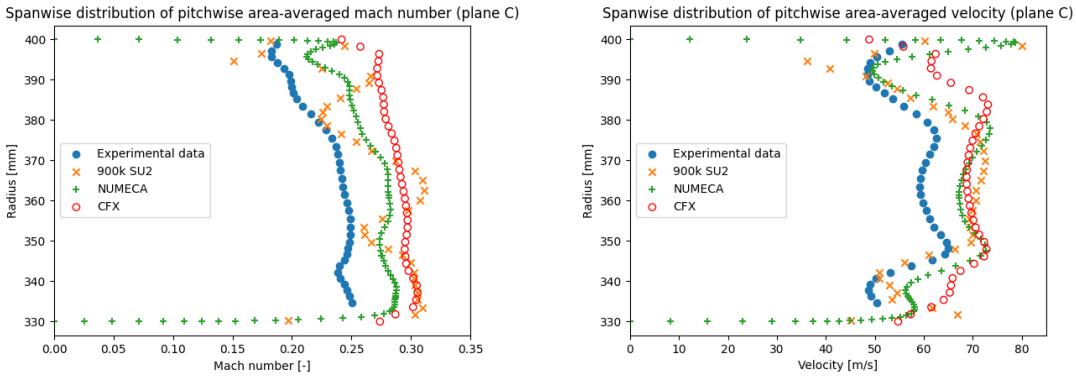
(a) Total pressure distribution at Plane C

(b) Static pressure distribution at Plane C

Figure 4.32: SU2 pressure results for the 900k TUD test case for Plane C

PLANE D

Figures 4.34 and 4.35 show the spanwise distribution results obtained for Plane D. Similar to the results found for plane B, figures 4.34a, 4.34b and 4.35a show that SU2 obtained similar results to that of CFX and Numeca,

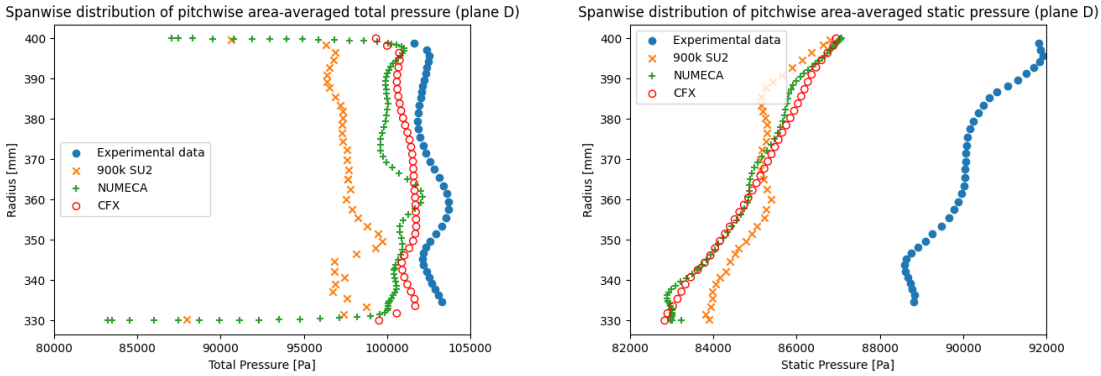


(a) Mach number distribution at Plane C

(b) Relative velocity distribution at Plane C

Figure 4.33: SU2 Mach number and velocity results for the 900k TUD test case for Plane C

again showing a greater total pressure loss. In contrast, the results obtained for Plane D show a greater deviation in trends and a lower Mach number (figure 4.35a). Figure 4.35b shows large fluctuations in data, where this is also shown by Numeca. Overall SU2 shows similar results to that of CFX and Numeca, where local deviations in trends can be found. Following the results for Plane D, SU2 over-predicts the fluid dynamic losses, showing a greater total pressure loss compared to the other data. It has also been found that the large deviation in results following Plane C does not affect the results found at Plane D significantly, as the data found by SU2 nears the data obtained by CFX and Numeca.

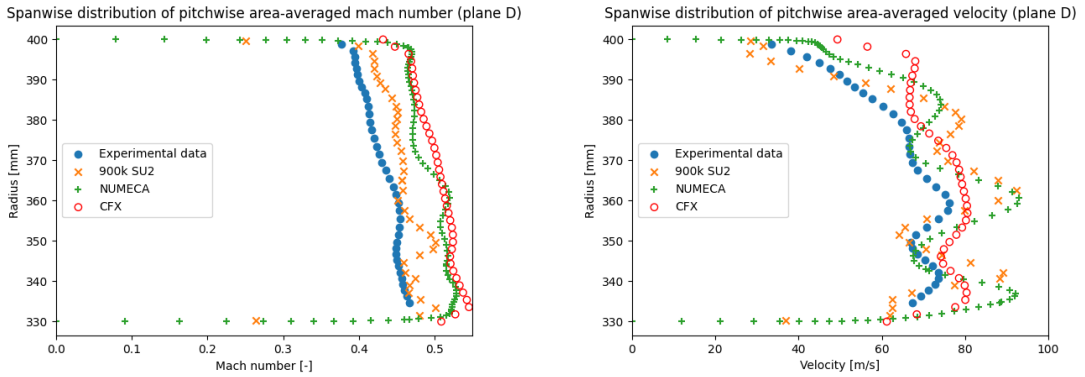


(a) Total pressure distribution at Plane D

(b) Static pressure distribution at Plane D

Figure 4.34: SU2 pressure results for the 900k TUD test case for Plane D

Overall it can be concluded that SU2 provides data that is reasonably comparable to that of CFX and Numeca. SU2 provides similar trends and results, where these can differ more significantly depending on the measurement plane and the flow variable. Especially for plane A and B SU2 provides results that match the other CFD data more closely. A greater deviation is found at plane C, where the deviating effects seem to have been reduced at plane D. Following all planes it has been noticed that nearly no CFD results match the experimental data. The pressure distribution of the converged ETH test case can be seen in figure 4.36.



(a) Mach number distribution at Plane D

(b) Relative velocity distribution at Plane D

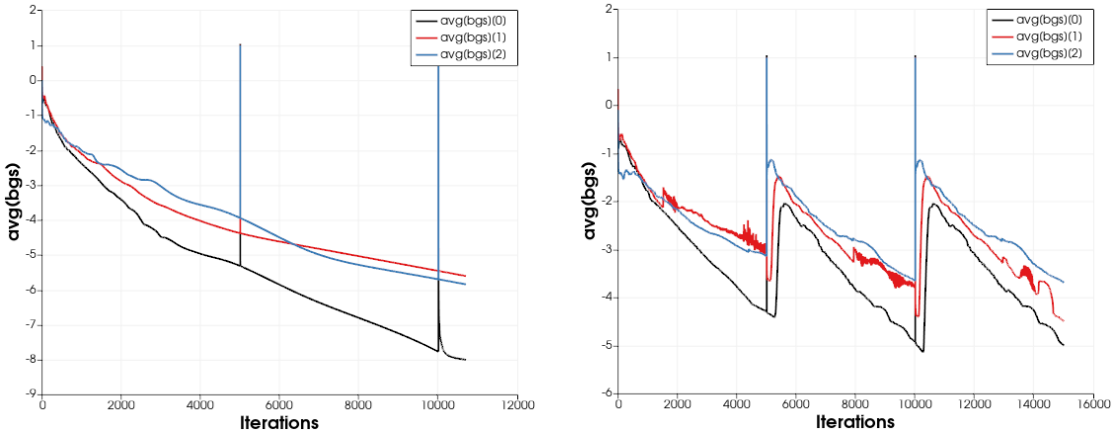
Figure 4.35: SU2 Mach number and velocity results for the 900k TUD test case for Plane D

Figure 4.36: Pressure distribution of the ETH test case at midplane in [Pa]

4.4.3. ETH NEWTON-KRYLOV PERFORMANCE STUDY

The Newton-Krylov solver in SU2 is also tested using the ETH test case. It was chosen to use the 900k TUD mesh for this analysis. Different results are expected as the ETH test case is an axial turbine, where the MTU test case is a radial compressor. Different flow phenomena could therefore have different effects on the Newton-Krylov solver, but this is to be tested within the current section. Identical to section 4.3.3, the performance of the Newton-Krylov solver following the ETH test case is tested by investigating the effect of the preconditioner and by varying the configurations of the solver. The convergence behaviour of the ETH test case is however to be investigated first, as this has not been done prior to the current section. The ETH test case is therefore re-run using a Newton-Krylov setup, where the results are compared to the standard solver (figure 4.37).

Figure 4.37 shows a comparison between the convergence behaviour of the normal solver and the Newton-Krylov solver. Following the standard setup, SU2 is able to reach convergence after approximately 10500 iterations. Both figures show spikes at 5000 and 10000 iterations, which are the result of the solver being restarted. Due to a lack of computing power, the test case was simulated in a segregated manner, where the solution was restarted after 5000 and 10000 iterations. The simulations have been run using a single CPU core, where every 5000 iterations took approximately 20 hours of computing time. In addition, it was unclear at what iteration the test case would converge, where restarting the simulation after 5000 iterations would therefore prevent the possibility of redoing the whole simulation upon a computer or solver crash. These small spikes at 5000 and 10000 iterations in residual values can therefore be ignored. The simulations have been set up such that the solver converges if one of the zones is able to reach a residual of 10^{-8}. Following figure 4.37a this is the case for the inlet section, where the residual for the passage and outlet are approximately $10^{-5.5}$ and 10^{-6}



(a) Residual behaviour following the standard setup

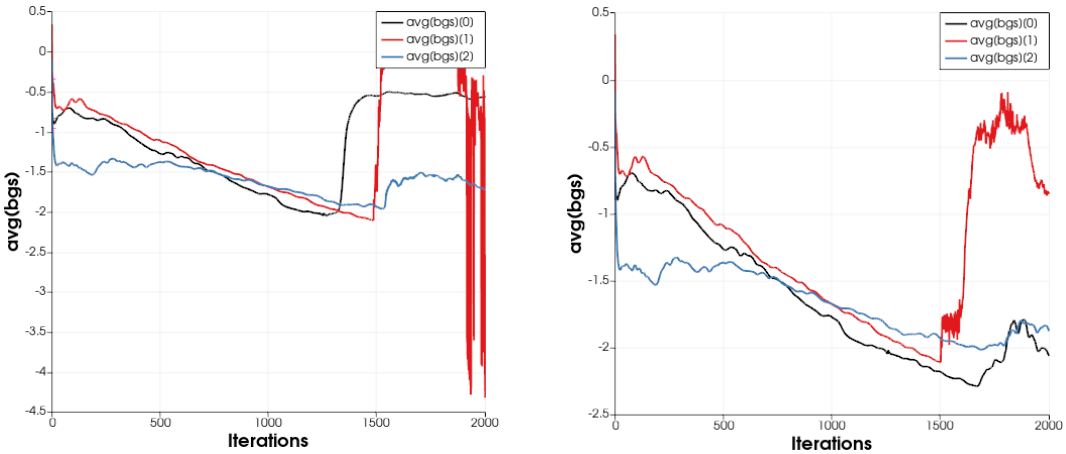
(b) Residual behaviour following the Newton-Krylov setup

Figure 4.37: Residual behaviour of the ETH test case for both the standard and the Newton-Krylov setup

respectively. It is likely for these sections to reach a residual of 10^{-8} as well, as no stalling behaviour is observed. Figure 4.37b shows the residual for the Newton-Krylov setup. It is clear that the solver does not reach the convergence value of 10^{-8}, where an oscillating residual for the blade section can be seen. Restarting the simulation shows to have a negative effect on the solver, where this should not affect the residual performance significantly. The oscillatory residual behaviour can be seen in all three (5000 iteration) sections. It should be noted that a longer test case has been run, where the solver restarted after 7500 iterations. This Newton-Krylov test case did however show similar behaviour to that of figure 4.37b. Due to the oscillatory behaviour of the residual being observed at approximately 1500 iterations (figure 4.37b), the subsequent test cases involving the preconditioner study are run using 2000 iterations.

PRECONDITIONER STUDY

The preconditioner study has been done using the 900k ETH mesh in combination with a Newton-Krylov solver configuration. The importance of using the correct preconditioner has been stated before, where the effect of different preconditioners on the ETH Newton-Krylov configuration is analysed within the current section. Similar to section 4.3.3 all different preconditioner options within SU2 (table 4.2) are analysed, where their effect on the solver is to be observed. Figure 4.38 shows the result for both the LUSGS and Jacobi-based preconditioners. The residuals for the different multizone sections have been visualised using black, red and blue which represent the inlet, blade passage and outlet section respectively. Both figures show a sharp increase in passage section residual at approximately 1500 iterations.



(a) Residual behaviour with the LUSGS preconditioner

(b) Residual behaviour with the Jacobi preconditioner

Figure 4.38: ETH convergence behaviour of the Newton-Krylov solver using a LUSGS and Jacobi preconditioner

When comparing figures 4.38a and 4.38b it can be found that the Jacobi preconditioner performs better compared to the LUSGS preconditioner, where the LUSGS preconditioner shows large fluctuations in residual behaviour involving the blade passage section. The residual behaviour of the solver following a Linelet preconditioner can be seen in figure 4.39. Different from figure 4.38, figure 4.39 shows normal behaviour involving the passage section where instead a significant increase in residual involving the inlet section can be seen.

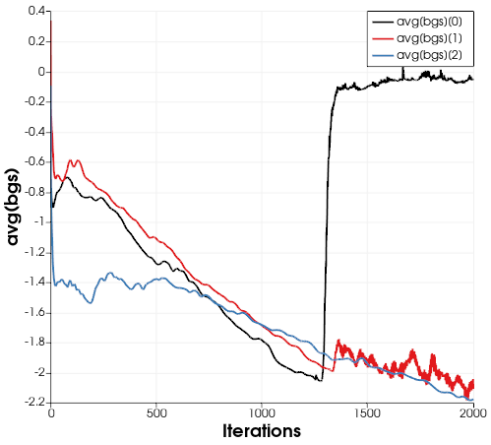
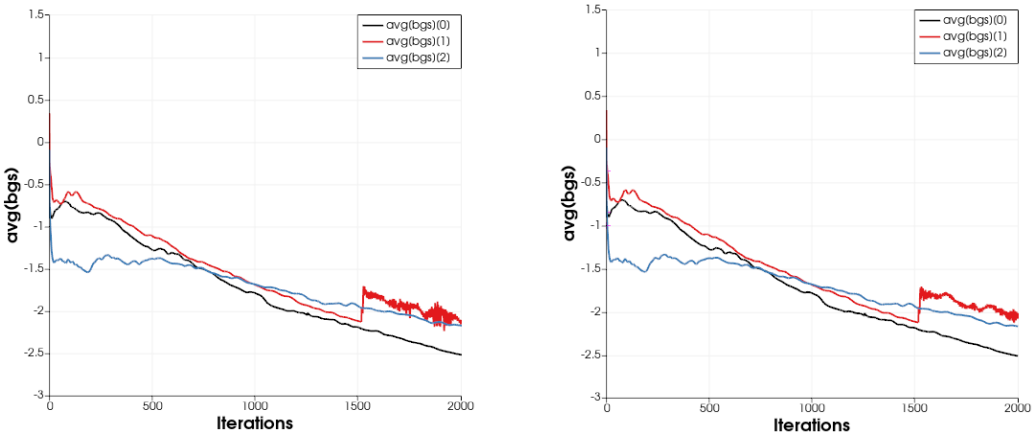


Figure 4.39: ETH convergence behaviour of the Newton-Krylov solver using a Linelet preconditioner

Figure 4.40 shows the performance of the SU2 Newton-Krylov solver when using the ILU-based preconditioners. Similar to the MTU test case, both ILU preconditioners show better performance compared to the results of the LUSGS, Jacobi and Linelet preconditioners. Interesting to notice is that following figure 4.40b the ILU(1) preconditioner seems to perform slightly better compared to the ILU(0) preconditioner. The oscillating residual found after 1500 iterations is more stable for the ILU(1) preconditioner compared to that of the ILU(0) preconditioner. Both preconditioners show however similar behaviour, where the residual of the passage section starts showing unusual performance at approximately 1500 iterations. Interesting to notice is that all residuals following figures 4.38, 4.39 and 4.40 show near identical performance following the first 1200 iterations. The solvers start to deviate either at 1300 or 1500 iterations, where sudden spikes in residual values can be found. Due to the resemblance in residual behaviour, it might be possible that a common cause is currently causing the simulations to diverge. The residual performance following a preconditioner based on Incomplete LU factorization showed the best result, where the residual was able to show more stable behaviour. Since the ILU(1) preconditioner provides the best results following the preconditioner study, the ILU(1) preconditioner is used in order to evaluate the effect of the flux Jacobian and the fix factor settings in the following section.



(a) Residual behaviour with the ILU(0) preconditioner

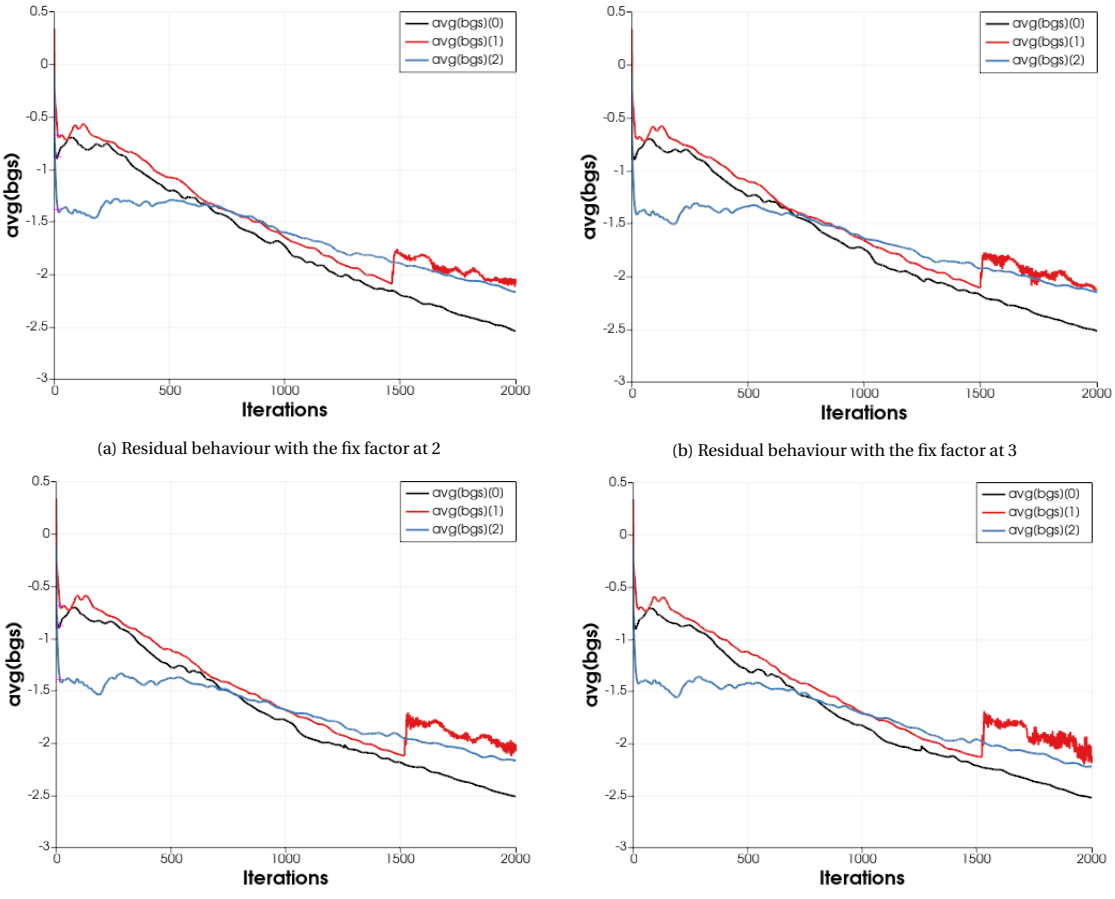
(b) Residual behaviour with the ILU(1) preconditioner

Figure 4.40: ETH convergence behaviour of the Newton-Krylov solver using a ILU(0) and ILU(1) preconditoner

JACOBIAN MATRIX SETTINGS ANALYSIS

Similar to section 4.3.3, the settings affecting the Jacobian matrix are investigated using the ETH test case. This involves the USE_ACCURATE_FLUX_JACOBIANS and CENTRAL_JACOBIAN_FIX_FACTOR settings, where their effect on the Newton-Krylov solver is investigated. The default configuration that has been used for this analysis is the IL(1) ETH test case, where the results following figure 4.40b are used in order to assess the behaviour of the different configurations. Following the USE_ACCURATE_FLUX_JACOBIANS setting analysis it was again found that the residual is not affected by this setting. This is concluded as the result obtained is identical to that of figure 4.40b.

The Jacobian fix factors have been analysed using the values 2, 3, 4 and 5. Figure 4.41 shows the results following the fix factor study. Interesting to notice is that nearly all results show similar performance, unlike the results found in section 4.3.3 for the MTU test case. Small differences in oscillatory behaviour can be observed, where the fix factor of 2 provides the smallest oscillations. Changing the fix factor did however not change the overall stalling performance of the ETH test case when using the Newton-Krylov solver. It was concluded in section 4.3.3 that the default fix factor of 4 provides the best result. Within the current section it has been concluded that this is not the case. The observed behaviour of the solver is however very similar, where using a fix factor of 4 for the ETH test case should provide sufficient results.



(a) Residual behaviour with the fix factor at 2

(b) Residual behaviour with the fix factor at 3

(c) Residual behaviour with the fix factor at 4

(d) Residual behaviour with the fix factor at 5

Figure 4.41: MTU convergence behaviour of the Newton-Krylov solver with respect to a Jacobian fix factor of 2, 3, 4 and 5

4.5. RESULTS SUMMARY

This section provides a summary of the results that have been obtained within this chapter. It discusses the results, providing additional commentary and reasoning. The current section has been divided into the discussion of the NACA 0012 test case (section 4.5.1), the LS89 test case (section 4.5.2), the MTU test case (section 4.5.3) and the ETH test case (section 4.5.4).

4.5.1. NACA 0012 ANALYSIS

Section 4.1 showed that the NUTSCFD solver is able to reach convergence for the NACA 0012 test case using less nonlinear iterations compared to SU2. Several possible reasons for this are discussed within the current section. SU2 showed for the NACA 0012 test case that the current Newton-Krylov solver can have a positive effect on the number of iterations it takes for the solver to reach convergence. It should however be noted that this does not mean that the Newton-Krylov setup is always quicker compared to the regular solver. It is still possible for the linear solver to require more iterations, where (re-)calculating the vector-matrix product can be a time-consuming process. Since both solvers use a maximum of 500 linear iterations it could however be concluded that the solver by Xu et al. provides better performance, as it required significantly fewer non-linear iterations.

Section 4.1 showed that the NUTSCFD solver by Xu et al. required approximately 800 non-linear iterations less in order to reach convergence. It is however unlikely for the SU2 mesh to be the same to that of the reference paper. Xu et al. give little information about the mesh, where only information is given about the farfield and y^+. It has only been stated that the condition of $y^+ < 1$ is being met. The currently used and developed NACA 0012 test case has been meshed using Gmsh, where currently Gmsh is unable to set specific values for y^+. In combination with the limited information for the farfield, it is possible for the developed NACA 0012 mesh to be more accurate compared to the reference data. A more accurate mesh generally requires more iterations for the solver to reach convergence, which could be the reason for the current result. It is therefore unclear whether the current analysis provides an accurate comparison between SU2 and NUTSCFD. The results from section 4.1 do however provide insight into the current performance of the Newton-Krylov solver in SU2 for external flow situations.

4.5.2. LS89 ANALYSIS

From the results it is clear that there is a very large difference in solver performance. Xu et al. are able to reach convergence with approximately 180 non-linear iterations, where this is not the case for SU2. With respect to SU2's Newton-Krylov solver, it was found that SU2 is unable to reach large numbers of linear iterations where the maximum has to be set at 2 iterations. Allowing more maximum linear iterations currently causes the FGMRES solver to diverge, thus limiting the number of maximum linear iterations allowed per non-linear iteration. It was found for the NACA 0012 test case that the maximum number of iterations could be set at 500, therefore indicating the FGMRES solver might have an issue with either the turbomachinery module or the LS89 test case. It is known for the LS89 test case to be challenging, as generally poorer solver performance is found due to the highly loaded design [1].

Instead of using the nonlinear steps in order to compare SU2 to the NUTSCFD solver, it might be considered to use the number of CPU work units. Xu et al. define the work units as the time it takes for the CPU to evaluate a single residual iteration. Currently SU2 does not provide timing mechanisms in the code which provides an indication of the time it takes in order for the CPU to evaluate a single residual iteration. Timing mechanisms are however typically not very complex code, where implementation of this could be relatively straightforward. Calculating the work units could provide valuable insight into the performance of the solvers, as comparing both solvers using only the non-linear steps is likely to provide a poor comparison. Xu et al. do not provide clear data on the total number of linear iterations next to the work units, unlike other similar CFD performance research conducted by Jalali et al. [18] (using airfoils). It is therefore currently not possible to use the work units or the non-linear steps in order to do an accurate analysis of the performance of SU2 to that of the NUTSCFD solver involving the LS89 test case. A more viable comparison could however be made between SU2 and NUTSCFD if the maximum number of 500 linear iterations is possible.

It was found that, with respect to the LS89 test case, SU2 is unable to reach higher levels of linear iterations. It is currently expected that the Newton-Krylov solver only does single iterations of the linear solver (when comparing figure 4.4 and 4.6), but this result is to be verified. The reason for this is currently unknown. It might be possible that the solver is able to reach the tolerance value, although this would not explain why FGMRES diverges when allowing for a higher maximum number of linear iterations. Next to the deviation in FGMRES performance, it is also expected that the mesh design has an effect on the results found in section 4.2. The LS89 mesh used is the default mesh created by the SU2 community, where no modifications are made. Xu et al. indicate that their first layer of cell height is $1.7 \cdot 10^{-5}$ m, where this is not known for the SU2 mesh. It is currently therefore unsure whether both LS89 meshes are similar enough, such that an accurate

comparison can be made. It is therefore recommended to set up a new turbomachinery test case which matches the setup of the reference data in order to do an accurate analysis of the Newton-Krylov solver. This has led to the development of the MTU test case.

4.5.3. MTU ANALYSIS

This section contains a discussion of the MTU results. It has been divided into different subsections which discuss the different studies of the MTU test case. The MTU test case was unable to reach convergences which complicated the overall performance analysis of SU2.

MTU PERFORMANCE ANALYSIS

Section 4.3.1 showed stalling results for both the MTU and the Eckardt test case. Within the current work it was not possible to reach convergence for the MTU impeller, which complicated the analysis of SU2's performance. The analysis of section 4.3.1 was unable to identify the reason for the stalling behaviour of the MTU, where these results are therefore discussed within the current section. Several aspects of the test case are evaluated in order to assess their effect on the current performance of the MTU impeller. It should be noted that the current section discusses the results that are obtained using the standard solver setup.

The MTU test case has been developed using the Ansys toolbox (section 3.3) where the mesh has been created in Turbogrid. The MTU test case has been developed for two studies, where next to the current work, it is used in order to model the slip effect in high-speed centrifugal compressors. This second research uses the MTU mesh in combination with CFX, where CFX is able to use the MTU mesh successfully. The chance of the MTU mesh causing the current stalling behaviour in combination with SU2 is therefore slim, as is also indicated by the stalling Eckardt test case. The MTU test case has been created by making use of the Eckardt test case. The setup configuration for the Eckardt impeller has been re-used and altered in order to comply with the conditions of the MTU. Changes are therefore only made to the boundary conditions of the configuration file. The Eckardt test case has been developed prior to the current thesis, where it had been assumed that this test case is able to reach convergence. Within the current work it was however found that this is not the case. Since the Eckardt test case has been used in prior work, and since the MTU mesh is compatible with CFX, it is likely that the root cause of the stalling behaviour is present within SU2. The default Eckardt test case showed a stalling residual, thus indicating that the code of SU2 has changed causing the Eckardt test case to no longer converge. The current work has been conducted using the turbo branch (specifically the 'restuct_singlezone' git branch) of SU2, which focuses on the development of the turbomachinery module part of SU2. Since the turbo branch is a separate Github branch, it is possible that the code base has changed as it is not part of the stable main release of SU2. Unfortunately no clear other reason for the stalling behaviour has been identified or provided by SU2. In addition since the ETH test case is able to reach convergence, an error being present in the configuration file of the MTU is also expected to be slim.

In retrospect it was expected for the tip-gap lacking MTU test case to perform better compared to the standard MTU test case. The MTU test case has been developed and researched prior to the ETH test case, where during the development of the ETH test case the problems involving the tip-gap interfaces were identified (section 3.5.4). The result of this is that the tip-gap interfaces of the MTU (SHROUDTIPGGISIDE1 and 2) are configured as internal markers, which is incorrect (see section 3.5.4). Results from the tip-gap lacking MTU and the Eckardt test case showed however no difference in behaviour with respect to stall performance. Future MTU test cases which have a tip-gap region should however be developed similar to that of the ETH turbine, such that the SHROUDTIPGGISIDE1 and 2 internal interfaces are not created.

Recently it has been stated in the SU2 developers group that the turbomachinery module contains a bug regarding the mixing plane. It was found that the mixing plane works well with a certain number of CPUs, where it performed poorly for other numbers. This resulted in discrepancies in mass flow rate and discontinuities in the flow field. A fix has been proposed, but this fix has not yet been implemented in the currently used 'restuct_singlezone' SU2 branch. Since the ETH test case is able to converge, the behaviour of the mixing plane is not necessarily expected to be the cause of the current stalling behaviour. The Eckardt test case using the fluid interface setup (table 4.1) did not provide better results (figure 4.10a), suggesting that the mixing plane might not be the root cause of the issue. The mixing plane performance might however be taken into consideration for future research.

Due to the MTU test case not converging, it is difficult to compare the performance of SU2 to that of the NUTSCFD solver. No comparison has therefore been made between the data obtained by SU2 and to that of Xu et al. [1] with respect to the MTU. Comparing the current data would result in an invalid analysis of SU2, where it is currently recommended to reach convergence before further research can be conducted involving the solver performance.

RESIDUAL ON GRID

Following the results of section 4.3.1 involving the analysis of the Mach contours, it was expected that the problematic residual zones would be near the tip-gap region. During the development of the ETH test case it was found that a mistake was made during the setup of the MTU test case. The MTU contains a tip-gap region, where the test case contains the SHROUDTIPGGISIDE1 and 2 markers. The effect of these markers has been discussed in section 3.5.5, where the creation of these markers should be avoided in order for SU2 to be able to simulate the test case correctly. In combination with the Mach plot observation it was therefore expected for section 4.3.2 to provide more results related to the tip-gap region, but this result was not obtained. Following section 4.3.2 no clear problematic zone was found, where plotting the residuals on the grid therefore did not provide a clear indication of why the MTU test case is currently stalling.

NEWTON-KRYLOV CONFIGURATION ANALYSIS

The results and conclusions that have been obtained in section 4.3.3 are with respect to the developed MTU test case in combination with the Newton-Krylov solver. Within section 2.3.2 it has been stated that the effectiveness of a preconditioner can be situation-specific. Changing the MTU test case in order to make sure the simulation converges could therefore behave differently compared to the current stalling MTU test case. Although it is likely for the ILU preconditioner to outperform the other preconditioners when using a new Newton-Krylov SU2 MTU test case, a new analysis has to be done in order to fully verify this result. Nonetheless it is recommended to use the ILU preconditioner when using the Newton-Krylov solver in SU2 due to the results found in section 4.3.3 and the fact that the NUTSCFD solver also uses an ILU preconditioner.

4.5.4. ETH ANALYSIS

This section contains the discussion of the ETH results. It has been divided into different subsections which discuss the different studies of the ETH test case. The ETH test case has been analysed, which showed to have similar results to that of CFX and Numeca. The residual performance of the ETH test case was not evaluated as no data with respect to convergence behaviour was available involving CFX and Numeca.

225K MESH PERFORMANCE ANALYSIS

The results obtained in section 4.4.1 show large differences in results when compared to the experimental data by Behr. The same results have been found in section 4.4.2 for CFX and Numeca, where no CFD solver was able to match the experimental data. No conclusion was therefore made with respect to the solver performance in section 4.4.1. It was stated that the 225k TUD mesh was not sufficient when simulating the ETH test case due to the lacking tip-gap region, especially following figure 4.28c. When comparing the 225k mesh results to that of the 900k mesh, it can be concluded that the 900k mesh results match CFX and Numeca more closely. The decision made in section 4.4.1 to develop a mesh containing a tip-gap region was therefore the correct conclusion.

900K MESH PERFORMANCE ANALYSIS

From section 4.4.2 it has been concluded that SU2 provides similar results to that of CFX and Numeca. SU2 provides data with trends showing similar behaviour and values, where these results deviate more strongly for plane C and D. The reason for the deviation in results is discussed within this section, where different topics are discussed. The first reason for the deviation in results might be due to SU2. SU2 showed near identical performance to CFX and Numeca for plane A, where the results deviate more strongly when travelling downstream. This could be due to SU2 over-predicting fluid dynamic losses, or the usage of different models which influence the final result. SU2, CFX and Numeca all use the SA-turbulence model, where no additional information with respect to the solver setup was provided for the reference data.

Another reason that could influence the final results of SU2 could be related to the mesh. It has been mentioned in section 3.5.5 that the TUD mesh is likely to differ in blade geometry compared to the NPU mesh. It has been tried to recreate the NPU mesh as accurate as possible, where it was not possible to use the exact

CAD geometry. It has been stated that the effect of this deviation in blade geometry is expected to be of little influence on the final result. Nonetheless this slight difference in geometry should be noted. In addition, the mesh element count is also not likely to have a significant effect on the SU2 results. From section 3.4.1 it has been concluded that a mesh count of 225k elements should be sufficient when simulating the ETH turbine test case. This section contains the grid independence study for the 225k TUD mesh, which does not have a tip-gap. It is therefore expected that 225k elements are not enough for the TUD mesh when it does contain a tip-gap. This section does however provide results for the ETH test case for different element counts, showing no large differences in fluctuating data when increasing the element count of the mesh. It is therefore unlikely for the element count of the mesh to be the reason behind the large fluctuating data found for plane C in section 4.4.2, but this is to be verified.

The final reason that is identified that might have an effect on the SU2 results is potential errors in the post-processing setup. From all spanwise results obtained in section 4.4 it has been tried to match the measurement planes to that of the open literature, where this is therefore not expected to be of influence. The results obtained in section 4.4.2 showed more deviations in trends and values when plotting stronger fluctuating results (for example with respect to the results found at plane C). It might therefore be possible that an error has been made in the averaging process, since the results for the low-fluctuating results (plane A and B) showed more similarities to the data from CFX and Numeca. It could therefore be considered to use mass-averaging, where these results could be compared to the mass-averaged data obtained by Behr.

NEWTON-KRYLOV CONFIGURATION ANALYSIS

The previous sections discussed the ETH test case involving the standard SU2 setup. The current section discusses SU2's performance with respect to the Newton-Krylov solver. It was found that the ETH test case was unable to reach convergence using the Newton-Krylov solver, where this is not the case for the standard setup. The simulation shows stalling behaviour, where the reason for this is unclear. The LS89 test case showed that SU2 should be able to reach convergences using the Newton-Krylov solver (figure 4.6) when simulating an axial turbine. It should however be noted that the LS89 test case is a single zone simulation, where no frame rotation is enforced.

Following the results of section 4.4.3 it was found that for the 2000 first iterations the ILU(1) preconditioner provides the most stable residual behaviour. This therefore validates the fact that the preconditioner can be test case specific, as has been mentioned in section 4.5.3. It was again found that the Incomplete LU factorization based preconditioners provide the best result, where the recommendation of using an ILU preconditioner when using the Newton-Krylov solver holds. In addition it was found in section 4.4.3 that using a fix factor of 2 might be more optimal over the default value of 4. It should however be noted that the default value provided similar results, indicating that a fix factor of 2 is not required.

5

CONCLUSION

The goal of this thesis was to analyse the solver performance of SU2 using turbomachinery analysis. Test cases have been developed which are used to analyse the residual behaviour of the solver and the simulation results. These test cases analyse the SU2 solver following different conditions, including axial and radial turbomachinery designs. A numerical study has been conducted in order to provide insight into SU2's current performance, which helps further development of its turbomachinery capabilities. Following the analysis conducted within the current work it has been found that the performance of the turbomachinery module within SU2 is inconsistent, and therefore requires improvement.

This thesis set-out to test SU2 by comparing its performance to that of the open literature. For this, the CFD solvers NUTSCFD, CFX and Numeca were to be used. SU2 was initially analysed using two test cases that have also been analysed using the NUTSCFD solver. These test cases include the analysis of a NACA 0012 airfoil and an LS89 turbine stage. The initial SU2 results showed that the NUTSCFD solver, developed by Xu et al. [1], was able to reach convergence with lower numbers of non-linear iterations. Following these results it was concluded that a test case had to be developed which could be used in order to provide a more accurate comparison between SU2 and NUTSCFD. This has led to the development of the MTU radiver compressor test case, which is based on the open MTU test case provided by MTU Aero Engines [4]. In addition to the MTU test case, a complementary 3D test case has been developed resulting in the creation of the ETH test case. The ETH test case is based on the axial 1.5 stage turbine developed at ETH Zurich. The ETH turbine has been investigated by Behr [21], where his experimental data was used in order to test SU2. During the thesis it was quickly realised that the MTU test case stalls when simulated using SU2. Many attempts have been made in order to identify the problem that causes the stalling behaviour, but this was ultimately not found. Nonetheless studies have been conducted with respect to different configuration settings, where their effect on the residual has been analysed. The ETH test case has also been used in order to verify SU2's performance to that of CFX and Numeca, where additional research has been conducted in combination with the Newton-Krylov solver. The ETH test case showed similar performance to that of CFX and Numeca, where it was unable to converge when using the Newton-Krylov solver. Interesting to notice is that the MTU test case did not converge when using the standard solver, where this is the case for the ETH turbine. The work that has been done for the current thesis provides answers to the research questions formulated in section 1.2. The answers to these questions are formulated below, where a summary is provided of the completed work that led to these answers.

1. **To what extent can SU2 be used in order to analyse turbomachinery flow problems?**

   (a) **To what extent does SU2 provide results that coincide with other CFD solvers involving turbomachinery flow simulations?**

   The results from the ETH test case showed that SU2 is able to obtain results similar to that of CFX and Numeca. SU2 showed similar trends, where a slight deviation in final value was found. From the current work it is therefore concluded that SU2 is able to produce relatively accurate results when simulating multistage axial turbines. Similarly SU2 was able to provide results that match the reference literature with respect to the NACA 0012 and LS89 test case. It was however noted

that SU2 was unable to match the residual of the NUTSCFD solver, where Xu et al. were able to obtain a converged simulation for less non-linear iterations. With respect to radial compressors, SU2 was unable to reach convergence following the MTU and Eckardt test cases. Both the MTU and Eckardt test case showed stalling behaviour. The stalling behaviour of the Eckardt test case indicates a greater problem within SU2, as the Eckardt test case has been used in prior research. From these results it has therefore been concluded that SU2 is currently unable to provide accurate results involving the analysis of radial compressors.

(b) **To what extent does the Newton-Krylov solver change the performance of SU2 with respect to turbomachinery flow simulations?**

SU2 has been analysed using test cases which involve both the standard turbomachinery solver and the Newton-Krylov solver. When comparing the results between the NACA 0012 airfoil and LS89 turbine stage it was found that the Newton-Krylov solver improved convergence with respect to the NACA 0012 test case. In contrast, the LS89 test case showed a decrease in convergence behaviour, where the Newton-Krylov solver increased the number of iterations required in order to reach convergence. A similar result was found for the ETH test case, which showed that the Newton-Krylov solver causes the simulation to stall, which was not the case for the normal solver. From the NACA 0012 result it is therefore clear that it is possible to increase the solver performance, where these benefits are currently unobtainable for turbomachinery simulations. The main difference between these problems is the turbomachinery module which is used for the turbomachinery test cases. Further research is therefore required in order to assess whether the turbomachinery module is the main cause of the reduced solver performance.

(c) **To what extent does the performance of the Newton-Krylov solver in SU2 differ from the NUTSCFD solver with respect to turbomachinery applications?**

The current work set out to answer this question using the MTU test case. It was found during the thesis that deviations in mesh designs provide a skewed representation of SU2's performance with respect to that of the NUTSCFD solver. Xu et al. provided limited information with respect to the NACA 0012 test case, where the LS89 mesh used within the current work could not be altered in order to meet the boundary layer cell dimensions. The MTU mesh was therefore to be developed where it should provide a more accurate comparison between SU2 and the reference data. Unfortunately the MTU compressor showed stalling behaviour, where it was not possible to reach convergence using SU2. An analysis was done with respect to the quality of the mesh using the residual, but this did not result in a solution for the currently observed behaviour. It was also found that the configuration file on which the MTU test case is based also did not converge (the Eckardt test case), which significantly reduced the chance of convergence for the MTU impeller. When comparing the MTU test case to the setup of the converging ETH test case, it is unlikely that a mistake was made in the configuration file, thus indicating that SU2 might be the problem. In order to assess SU2's performance it is therefore advisable to first make sure a valid radial turbomachinery test case is able to reach convergence before the Newton-Krylov solver can be analysed. This thesis was therefore unable to provide an answer to the research question involving the comparison between SU2 and NUTSCFD with respect to the Newton-Krylov solver.

2. **What are potential areas of the SU2 code that could be improved in order to increase the Newton-Krylov solver performance with respect to turbomachinery applications?**

(a) **What algorithms and settings in SU2 can be identified which have a large effect on the rate of convergence of the Newton-Krylov solver?**

Following the literature study it was found that both GMRES and the preconditioner are likely to have a significant effect on the convergence behaviour of the Newton-Krylov method. It should be noted that the ILU(0) preconditioner used by Xu et al. is based on a blended approximate Jacobian matrix which is created by combining first- and second-order-accurate spatial discretization. In contrast SU2 does not use a blended approximation of the Jacobian matrix when using an ILU(0) preconditioner. SU2 currently also uses FGMRES where Xu et al. use right preconditioned GMRES. If desired, it is expected that the implementation of right preconditioned GMRES is more straightforward compared to the implementation of the blended ILU(0) preconditioner. Future research could therefore be conducted with respect to SU2's current code structure, assessing the

possibility of implementing these changes. In addition, Xu et al. also use different Jacobian formation tools compared to SU2. It is possible that the current Jacobian forming procedure of SU2 is limiting the performance of the Newton-Krylov method, but this is not expected as the NACA 0012 test case is able to reach convergence more efficiently compared to the standard solver. Instead it is proposed to investigate why the LS89 test case is unable to reach more than 2 linear FGMRES iterations when using the Newton-Krylov solver, as this is more likely to give an indication of the limiting factor currently influencing the rate of convergence.

(b) **To what extent does changing the relevant algorithms and settings which affect the Newton-Krylov solver within SU2 have an effect on the convergence rate?**

The effect of different preconditioners on SU2's Newton-Krylov method has been analysed in sections 4.3.3 and 4.4.3, showing a difference in the performance of the solver when using different preconditioners. It has been concluded that the ILU preconditioners provide the best results, where Xu et al. chose to use an ILU(0) preconditioner. Other settings have been evaluated involving the 'USE_ACCURATE_FLUX_JACOBIANS' and 'CENTRAL_JACOBIAN_FIX_FACTOR' settings, where only the fix factor influenced the convergence behaviour of the solvers. It was found that the correct fix factor can be test case specific, where for both the MTU and ETH test cases the default value of 4 provided good results. SU2 did not provide other options which affect the Newton-Krylov method following the example configuration template on GitHub.

The current work has provided a numerical assessment of SU2's performance involving turbomachinery flow simulations. It was found that SU2 provides similar results to that of CFX and Numeca, but also showed many issues involving its Newton-Krylov solver. Due to the stalling behaviour of the MTU test case it was not possible to compare SU2's Newton-Krylov performance to that of NUTSCFD accurately. The current thesis has analysed SU2 using different test cases, where the data obtained using these test cases has been evaluated and compared. This thesis therefore provides insight into the current strengths and weaknesses of SU2. Research has shown that SU2 is currently unable to simulate the radial compressors test cases, but does provide relatively accurate results with respect to the axial turbines. It was also found that the Newton-Krylov solver currently underperforms when it is used for turbomachinery analysis, where the standard solver is often able to obtain better results. Improving SU2's Newton-Krylov method therefore requires further research, where suggestions resulting from the current thesis are summarized in chapter 6.

6

RECOMMENDATIONS

Many suggestions for future research can be done based on this thesis. This section contains a summary of the recommendations that can be made with respect to the problems encountered during the current work. Many issues have been encountered involving stalling simulations, where the reason for this behaviour is currently unknown. The analysis of the SU2 solver has also led to several new questions, which should help further analyse SU2's current behaviour. Topics that could be considered for further research are given below, accompanied by a motivation of why this might be considered.

- **Investigate why the LS89 test case is limited to 2 linear iterations**

  The LS89 test case currently crashes when increasing the maximum number of allowed linear iterations. Xu et al. were able to increase their maximum number of linear iterations to 500, where this is expected to have a significant effect on the final number of non-linear iterations required in order to reach convergence. It was found that the standard SU2 solver matches the Newton-Krylov solver in performance when limited to 1 linear iteration. It is therefore currently hypothesised that the Newton-Krylov solver limits the linear iterations significantly, which hampers the effectiveness of the method. Finding out why the linear iterations are limited might give a good indication of why the current Newton-Krylov solver behaves poorly when analysing turbomachinery.

- **Investigate why the Eckardt test case is not converging**

  The MTU test case is based on the Eckardt test case. During the thesis it was found that the default Eckardt configuration causes the solver to stall. It is therefore recommended to first run the reference test case using SU2, before using the configuration file for a newly developed test case. It was initially assumed that the Eckardt test case was able to reach convergence, but it was later discovered that this was not the case. It is therefore advisable to first make sure the old test case is able to run before a new test case can be developed.

- **Investigate the implementation of GMRES and/or the blended Jacobian preconditioner**

  The solver developed by Xu et al. uses a combination of GMRES with a blended Jacobian preconditioner. Since SU2 does not have either of those it might be considered to conduct further research into the implementation of these models. SU2 uses FGMRES, where in contrast to NUTSCFD, this preconditioner is flexible. It should be noted that Xu et al. use a preconditioner, but this preconditioner is fixed for every inner linear iteration. The implementation of this right preconditioned GMRES into SU2 might therefore be straightforward, such that FGMRES is altered where it does not update the preconditioner for every inner iteration. Whether this is the case is however to be determined. Implementing the blended Jacobian is less trivial, as it uses both Jacobian matrices approximated following first- and second-order-accurate spatial discretization in order to form the preconditioner.

- **Investigate into the implementation of CPU work units into SU2**

  Xu et al. provide convergence data using the non-linear iterations of the solver and the work units of the CPU. It has been discussed within this thesis that only comparing the non-linear iterations might not give an accurate representation of the solver's performance. Comparing only the non-linear iterations

involving SU2 would therefore provide a skewed analysis. This comparison might only be valid if SU2 is able to reach the maximum number of 500 iterations for the linear solver, while also using the Newton-Krylov solver. This comparison would therefore not be fair if the standard SU2 solver is used. It should be noted that less non-linear iterations does not automatically mean faster convergence. It was found during the current work that generally the Newton-Krylov solver requires more time in order to calculate a single non-linear iteration. Since SU2 currently does not output the CPU work units, it might be valuable to implement code which allows for the calculation of these units. This should provide a more accurate representation of the time required in order for the solvers to converge.

- **Investigate the effect of running multi-core simulations on the Newton-Krylov solver**

  All results obtained within the current work have been run using a single CPU core. It was found that the mixing plane within the Turbomachinery module might provide different results based on the number of cores used in order to solve the problem. A fix for this problem has been proposed during the timeline of the thesis, where it is possible that this fix was not yet implemented in the solver used within the current work. Additional research might therefore be needed in order to assess whether this issue has had an effect on the results found within this thesis.

- **Investigate the effect of the Krylov methods within SU2**

  SU2 contains multiple Krylov methods which can be used in order to solve the linear system of equations obtained from Newton's method. The current thesis does not consider the different linear solvers within SU2, where the FGMRES solver is chosen due to Xu et al. using the GMRES algorithm. The BCGSTAB algorithm has been considered in section 4.3.1, but this is not in combination with Newton's method thus not forming a Newton-Krylov solver (since BCGSTAB is also a Krylov method). It is unlikely for these linear solvers to provide different results compared to FGMRES, as the MTU test case was unable to reach converges using BCGSTAB. Nonetheless these solvers might be considered to be of interest as the combination of these solvers and Newton's method also form a Newton-Krylov solver.

- **Investigate whether the turbomachinery module influences the Newton-Krylov solver**

  Due to the Newton-Krylov solver diverging for the ETH test case it is possible that the turbomachinery module is causing some of the issues. It should however be noted that SU2 is able to run the LS89 test case, while using the Newton-Krylov method, thus indicating that the turbomachinery module might not be the root cause. It might however still be valuable to investigate whether the turbomachinery module is causing complications when being used in combination with the Newton-Krylov solver. A suggestion is therefore made to set up a test case which can be run both with and without the turbomachinery module. This test case should use the Newton-Krylov solver, where analysing the residual should provide insight into whether the turbomachinery module is currently causing the problems. The question of whether the turbomachinery module is interfering is raised due to the fact that the Newton-Krylov method works efficiently for the NACA 0012 airfoil, where this is not the case for the LS89 turbine stage. The efficiency of the method might however also be affected by the LS89 test case being challenging [1], where using a different test case would provide greater insight.

In addition several remarks can be made with respect to the development of new SU2 test cases. These remarks could be considered as good practices for future work where they should prevent problems when setting up a new test case:

- When developing a new test case it is advisable to match the mesh orientation to that of the reference case. Both the MTU and ETH test cases have shown to have preferred mesh orientation and location. Using the wrong orientation results in a negative blade count in the Turbomachinery Preprocessor.
- When getting the 'killed' error it is possible for SU2 to lack computational power. It was sometimes found that running multiple SU2 simulations at the same time resulted in some of the solvers being 'killed'. The same error was given for the fine ETH mesh (using 5.5 million elements), where a coarser mesh was able to run. This issue might be solvable by running simulations using multiple cores, or by using more memory.
- It is advised to match the reference test case closely when creating a new test case. It is also advisable to make sure that the reference test case is able to converge before developing/analysing the new test case.

BIBLIOGRAPHY

[1] S. Xu, P. Mohanamuraly, D. Wang, and J.-D. Müller, *Newton–krylov solver for robust turbomachinery aerodynamic analysis,* AIAA Journal **58**, 1320 (2020).

[2] S. Langer, *Preconditioned newton methods to approximate solutions of the reynolds averaged navier-stokes equations,* DLR Deutsches Zentrum fur Luft- und Raumfahrt e.V. - Forschungsberichte **2018**, 1 (2018).

[3] H. Ibeid, L. Olson, and W. Gropp, *Fft, fmm, and multigrid on the road to exascale: Performance challenges and opportunities,* Journal of Parallel and Distributed Computing **136** (2019), 10.1016/j.jpdc.2019.09.014.

[4] K. U. Ziegler, H. E. Gallus, and R. Niehuis, *A Study on Impeller-Diffuser Interaction—Part I: Influence on the Performance ,* Journal of Turbomachinery **125**, 173 (2003).

[5] J. D. Anderson, *Fundamentals of aerodynamics*, 6th ed. (McGraw-Hill, 2017).

[6] J. D. Anderson, *Computational Fluid Dynamics*, 1st ed. (McGraw-Hill, 1995).

[7] M. S. Campobasso and M. B. Giles, *Effects of flow instabilities on the linear analysis of turbomachinery aeroelasticity,* Journal of Propulsion and Power **19**, 250 (2003).

[8] Johnson, F. T., Kamenetskiy, D. S., Melvin, R. G., Venkatakrishnan, V., Wigton, L. B., Young, D. P., Allmaras, S. R., Bussoletti, J. E., and Hilmes, C. L., *Observations regarding algorithms required for robust cfd codes,* Math. Model. Nat. Phenom. **6**, 2 (2011).

[9] W. Briggs, V. Henson, and S. McCormick, *A Multigrid Tutorial, 2nd Edition* (2000).

[10] D. S. Kamenetskiy, J. E. Bussoletti, C. L. Hilmes, V. Venkatakrishnan, L. B. Wigton, and F. T. Johnson, *Numerical evidence of multiple solutions for the reynolds-averaged navier–stokes equations,* AIAA Journal **52**, 1686 (2014).

[11] Y. Saad and M. H. Schultz, *Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems,* SIAM Journal on Scientific and Statistical Computing **7**, 856 (1986).

[12] J. G. L. Booten, P. M. Meijer, H. J. J. te Riele, and H. A. van der Vorst, *Parallel arnoldi method for the construction of a krylov subspace basis: An application in magnetohydrodynamics, High-Performance Computing and Networking*, (1994).

[13] Y. Saad, *A flexible inner-outer preconditioned gmres algorithm,* SIAM Journal on Scientific Computing **14**, 461 (1993).

[14] S. Xu and S. Timme, *Robust and efficient adjoint solver for complex flow conditions,* Computers Fluids **148**, 26 (2017).

[15] A. Pueyo and D. W. Zingg, *Efficient newton-krylov solver for aerodynamic computations,* AIAA Journal **36**, 1991 (1998).

[16] T. D. Economon, F. Palacios, S. R. Copeland, T. W. Lukaczyk, and J. J. Alonso, *Su2: An open-source suite for multiphysics simulation and design,* AIAA Journal **54**, 828 (2016).

[17] A. Jameson, *Multigrid algorithm for compressible flow calculations,* Multigrid Method II **1228**, 166 (November 2006).

[18] A. Jalali and C. Ollivier-Gooch, *Higher-order unstructured finite volume rans solution of turbulent compressible flows,* Computers Fluids **143**, 32 (2017).

[19] A. McCracken, A. D. Ronch, S. Timme, and K. Badcock, *Solution of linear systems in fourier-based methods for aircraft applications,* International Journal of Computational Fluid Dynamics **27**, 79 (2013).

[20] T. Arts and M. Lambert de Rouvroit, *Aero-Thermal Performance of a Two-Dimensional Highly Loaded Transonic Turbine Nozzle Guide Vane: A Test Case for Inviscid and Viscous Flow Computations,* Journal of Turbomachinery **114**, 147 (1992).

[21] T. Behr, *Control of rotor tip leakage and secondary flow by casing air injection in unshrouded axial turbines,* (2007), 10.3929/ethz-a-005478482.