Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Engineering
Department of Energy and Process Engineering

Johan Raftevold Espelund

Numerical modelling of fin side heat transfer and pressure loss for compact heat recovery steam generators

Master's thesis in Mechanical Engineering
Supervisor: Lars O. Nord
Co-supervisor: Ole H. H. Meyer and Geir Skaugen
June 2022

NTNU
Kunnskap for en bedre verden

HighEFF

Johan Raftevold Espelund

Numerical modelling of fin side heat transfer and pressure loss for compact heat recovery steam generators

Master's thesis in Mechanical Engineering
Supervisor: Lars O. Nord
Co-supervisor: Ole H. H. Meyer and Geir Skaugen
June 2022

Norwegian University of Science and Technology
Faculty of Engineering
Department of Energy and Process Engineering

NTNU
Kunnskap for en bedre verden

Acknowledgements

This master work was carried out with SINTEF Energy Research with support from the FME HighEFF project.

I would like to thank my supervisors Lars O. Nord, Ole H. H. Meyer and Geir Skaugen for their support and feedback during my thesis work, and particularly Ole for helping me out with all things OpenFOAM related, which has been of great help the last six months. Last, but not least, I also want say thank you to Christianne for being by my side and always supporting me.

Abstract

A quarter of the Norwegian CO_2 emissions in 2019 stemmed from oil and gas production, and steam bottoming cycles have been proposed as a means to reduce the emissions. Due to weight and size limitations offshore, these cycles need to be compact, and an optimization tool for bottoming cycle design has previously been developed to achieve this. This tool relies on empirical correlations to predict the pressure drop and heat transfer in the heat recovery steam generator (HRSG), which is a crucial component of the cycle. However, as these correlations are based on experiments with typical onshore designs, they are often not valid for the compact designs encountered in offshore HRSGs.

In order to extend the validity range of the optimization tool, this work will include numerical analyses of heat transfer and pressure loss in finned tube bundles by means of computational fluid dynamics (CFD). The aim has been to develop a model that is capable of simulating an optimized HRSG design, such that the performance of the empirical correlations can be evaluated. To validate the numerical model, the results were compared with available experimental data, and then the model's performance was compared with a selected empirical correlation.

First, a grid generation procedure was developed, capable of generating high quality numerical grids for a large span of parameter combinations. Grids for three fin-tube geometries were generated (two serrated and one solid), which was used in further CFD-simulations. Each geometry was simulated using steady-state solution strategies, as well as transient simulations for the solid and one of the serrated geometries. The open-source software OpenFOAM was used to set up the numerical model, which includes heat transfer in both gas and fins. It was found that most of the CFD results were within 20% of experimental data, and generally were more accurate than the empirical correlation. Though the transient solution strategy is computationally more intensive than the steady-state strategies, it is deemed to be most accurate and robust way to model the flow due to its ability to handle flow instabilities.

Samandrag

Ein fjerdedel av dei norske CO_2 utsleppa i 2019 kom from olje- og gassproduksjon. For å redusere desse utsleppa har installasjon av damp-botnsyklusar blitt føreslege. På grunn av høge krav til vekt og storleik offshore må damp-botnsyklusen ha ei kompakt utforming, og for å oppnå dette har eit optimaliseringsverkty tidlegare blitt utvikla. Dette verktyet er avhengig av å bruke empiriske korrelasjonar for å føreseie trykkfall og varmeoverføring i varmevekslaren, som er ein av dei viktigaste komponentane i syklusen. Bakdelen med dette, er at desse korrelasjonane ofte er baserte på geometriar for landbaserte varmevekslarar, og derfor ikkje er gyldige for den kompakte utforminga ein treng offshore.

For å utvide gyldigheits-området til optimaliseringsverktyet, har det i denne oppgåva blitt nytta numeriske straumingsberekningar (*Computational Fluid Dynamics*, CFD) for å føreseie trykkfall og varmeoverføring finna rør-buntar. Målet har vore å utvikle ein numerisk modell som er i stand til å simulere strøyming i dei optimaliserte varmevekslarane, slik at ein kan undersøke kor bra korrelasjonane presterer. For å validere den numeriske modellen blei resultata samanlikna med tilgjengelege eksperimentelle data, i tillegg til ein utvalgt empirisk korrelasjon.

Eit verkty for å generere numeriske gitter blei først utvikla, som kan lage numeriske gitter av høg kvalitet for eit stort spekter av geometriar. Gitter blei laga for tre finna rør (eitt med heile finnar og to med segmenterte finnar), som blei vidare brukt i CFD-simuleringar. Kvar geometri blei simulert med stasjonære løysingsmetodar, i tillegg til at transiente simuleringar blei gjennomført for geomtrien med heile finnar og ein av dei med segmenterte finnar. OpenFOAM blei brukt til å gjennomføre CFD-simuleringane og inkluderte varmeoverføring i finnane i tillegg til i gassen. Det vart funne at dei fleste CFD-resultata var innafor 20% av eksperimentelle verdiar, og var stort sett meir treffsikre enn den empiriske korrelasjonen. Sjølv om dei transiente simuleringane krev betydeleg meir reknekraft enn dei stasjonære, er det den mest nøyaktige og robuste måten å modellere strøymingar på, ettersom den er i stand til å handtere ustabile strøymingar.

Contents

NOMENCLATURE

$\dot{Q}_f$	heat transfer rate to fins [W]
$\mathbf{u}$	velocity vector [m s^{-1}], $(u\ v\ w)$
$A_{c,f}$	cross-sectional area of fin [m^2]
c_p	specific heat capacity [J kg^{-1} K^{-1}]
d_o	outer tube diameter [m]
F_{min}	minimum free flow area [m^2]
h	specific enthalpy [J kg^{-1}]
h_f	fin height [m]
h_s	serration height [m]
p	pressure [Pa]
P_l	longitudinal tube pitch [m]
P_t	transverse tube pitch [m]
q	heat flux [W m^{-2}]
s_f	fin pitch [m]
T	temperature [K]
T_f	fin temperature [K]
t_f	fin thickness [m]
T_w	tube wall temperature [K]
$u_{F_{min}}$	mean velocity in minimum free flow area [m s^{-1}]
w_s	serration width [m]

Greek letters

α	thermal diffusivity [m^2 s^{-1}]
α_o	outside heat transfer coefficient [W m^{-2} K^{-1}]
β	included angle [°]
η_f	fin efficiency [-]
$\eta_{f,\text{th}}$	theoretical fin efficiency [-]
γ	momentum source term [m s^{-2}]
κ	von Kármán constant [-]
λ	thermal conductivity [W m^{-1} K^{-1}]
μ	dynamic viscosity [kg $m^{-1}s^{-1}$]
ν	kinematic viscosity [m^2s^{-1}]
ρ	density [kg m^{-3}]
$\tilde{\nu}$	modified turbulent viscosity [m^2s^{-1}]

Dimensionless numbers

Co	Courant number, $u\Delta t/\Delta x$
Eu	Euler number, $2\Delta p/\rho u_{F_{min}}^2$
Nu	Nusselt number, $\alpha_o d_o/\lambda$
Pr	Prandtl number, α/ν
Pr_t	Turbulent Prandtl number, α_t/ν_t
Re	Reynolds number, $d_o u_{F_{\min}}/\nu$

Abbreviations

CFD	Computational Fluid Dynamics
CHT	Conjugate Heat Transfer
HRSG	Heat Recovery Steam Generator
HTE	Heat Transfer Enhancement

1 INTRODUCTION

1.1 Background

Oil and gas production contributes significantly to the global CO_2 emissions. In Norway, it is the industry that emits most greenhouse gases, accounting for 27% of CO_2 emissions from Norwegian territory in 2019. The largest contributor to these emissions (making up around 85%) are the gas turbines used for power generation offshore. [1]

The large emissions of the gas turbines makes them an attractive candidate for emission reduction, and installing steam bottoming cycles (Figure 1.1) has been proposed as a way to achieve this. Most of today's offshore power systems utilize the hot exhaust gases to some degree, e.g. for heating crude oil, but the heat loss to the atmosphere is still significant. With a steam bottoming cycle the heat is utilized for power generation in steam generators, lowering the demand for power production from the gas turbines, and can reduce the turbine CO_2 emissions with as much as 25% [2]. There is however no widespread use of offshore steam bottoming cycles today, mainly due to weight and size limitations.

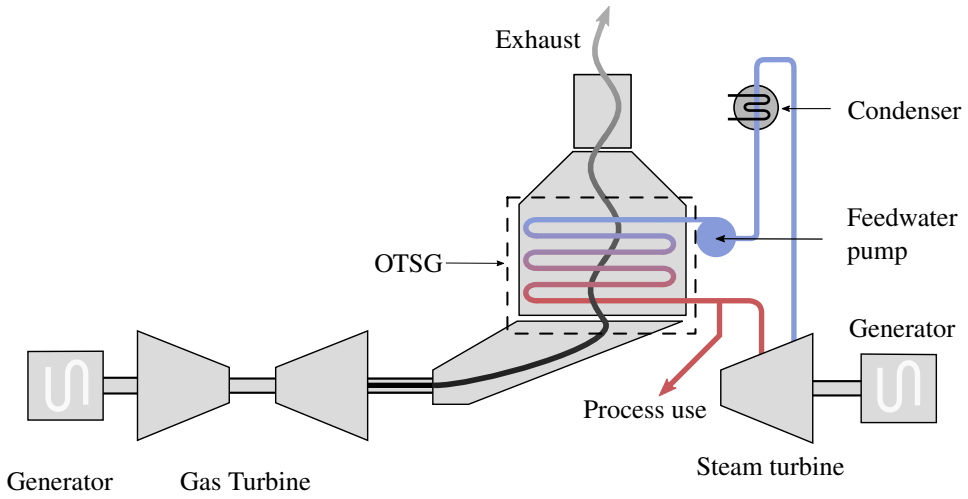


Figure 1.1 – Schematic for a steam bottoming cycle, showing the gas turbine, the waste heat recovery unit (the OTSG) and the steam turbine.

The heat recovery steam generator (HRSG) is a crucial and large component of the steam bottoming cycle. The aforementioned limitations necessitate compact designs, and the once through steam generator (OTSG) has been found to be the most suitable HRSG type for offshore steam bottoming cycles [2]. One of the key factors when optimizing OTSGs for weight and size, is heat exchanger tubes with small diameters compared with onshore equivalents [3, 4]. To improve heat transfer on the gas side, the tubes are usually enhanced with extended surfaces, most commonly solid or serrated fins (Figure 1.2).

An optimization procedure has previously been developed for use in the design process, and relies on correlations for finned tube banks in order to predict the heat transfer and pressure drop of the HRSG. Correlations are almost exclusively based on experimental data, and their region of validity is therefore limited to the range of experiments that they are based on. This has proved to be a challenge for the optimization of the compact offshore HRSG designs, which have fin and tube geometries outside the validity range of the correlations. The result is that different correlations tend to give different predictions when compared to the same experimental data sets. Holfeld [6] reported up to 77% spread between the

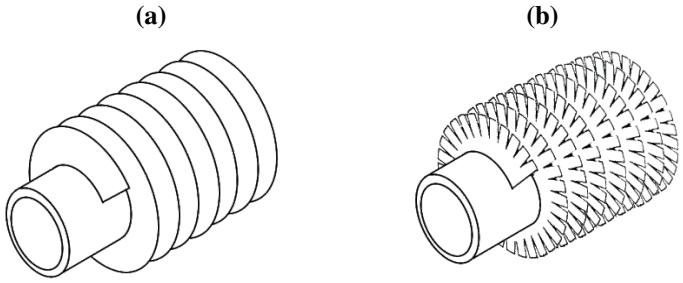


Figure 1.2 – Two types of extended surfaces, **(a)** solid fins and **(b)** serrated fins. [5]

correlations for heat transfer, and up to 410% for pressure drop when comparing different correlations to the same experimental data.

Ideally, new correlations should be developed based on experiments that are performed under conditions close to those expected for the offshore HRSGs. However, performing experiments to produce enough data for new correlation development is both costly and time-consuming, and therefore Computational Fluid Dynamics (CFD) is proposed as a way to better predict heat transfer and pressure drop in compact HRSGs, and to validate the designs produced by the optimization procedure.

As computational power available to researchers and engineers has increased, CFD has become more widespread in the engineering field. This has led to the emergence of numerical simulations of finned tube banks in the last two decades. Only relatively simple cases were investigated in the earliest studies, often only using solid fins and only including a few tube rows. The more recent studies have dealt with complex flows, also including helical serrated fins in turbulent flows. However, validation against experimental data is scarce for most studies.

1.2 Objectives

The aim of this work is to develop a numerical model that is able to predict heat transfer and pressure drop for a wide range of geometrical parameters that are relevant for compact HRSGs, particularly for the conditions in offshore steam generators. An important aspect of the model is its ability to give accurate results without being too time and resource demanding. The main objectives of this work are to:

1. Develop a grid generation procedure that is capable of generating geometries for a wide range of geometry parameters. The procedure should be able to read the output parameters from an optimized design and automatically create a numerical grid of high quality, for both solid and serrated fins.
2. Set up a numerical simulation that is able to predict both pressure drop and heat transfer, using grids from the grid generation procedure. The simulation should be easy to set up and give robust and accurate results.
3. Validate the numerical model by comparing results with available experimental data.

2 FINNED TUBE THEORY

Heat transfer enhancement is used to increase the heat transfer rate to or from a surface, e.g. the tubes in a HRSG. This is usually achieved by utilizing extended surfaces, which does not only increase the total heat transfer area, but also affects the flow field around the surface. In a HRSG, extended surfaces are mainly used on the tube outside surfaces, as the thermal resistance in a gas is orders of magnitude larger than that of the fluid on the inside of the tube. This section will give an introduction to finned tubes, one of the most common types of extended surfaces in HRSGs.

2.1 Types of finned tubes

Several variants of finned tubes exist (Figure 2.1), with the simplest being the solid fin. Alternatives to the solid fin usually aim to disrupt the boundary layer development on the fin to improve mixing and thus increase heat transfer. This is achieved by boundary layer development on repeated elements on the fin, which separates in the wake before the next segment. Types of fins are shown in Figure 2.1, and include slotted, punched triangles, serrated and wire loop fins. Out of these, the solid and serrated fins tends to be the most popular options [7], and will be the fin enhancement investigated in this work.

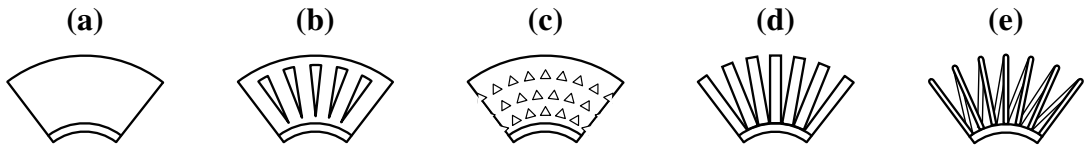


Figure 2.1 – Common enhanced fin geometries: **(a)** solid, **(b)** slotted, **(c)** punched triangles, **(d)** serrated and **(e)** wire loop fins. Adapted from Webb and Kim [7].

Solid fins have a larger total surface area compared with serrated fins, which yields a higher total heat transfer for the solid fins, assuming all other parameters to be constant. However, the mixing introduced by the serrated fins yields a higher heat transfer coefficient. This, combined with lower weight and easier manufacturing, makes serrated fins attractive for compact heat exchangers.

Helical fin attachment (Figure 2.2a) is most common, as the fins can be wrapped around and welded to the tube in a single pass for the entire tube length. Annular fins (Figure 2.2b) require each fin revolution to be individually placed and welded, making this option more costly, and thus only helical fins will be considered here.

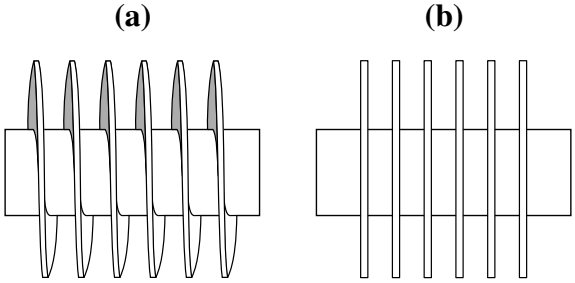


Figure 2.2 – Type of fin attachment methods, **(a)** helical and **(b)** annular, adapted from Lindqvist [8].

2.2 Geometry description

The layout of the tubes inside a heat exchanger can either be in-line or staggered, the latter is illustrated in Figure 2.3a. The center-to-center distance between the tubes in flow and transitional directions are termed longitudinal pitch P_l and transverse pitch P_t, respectively. Only staggered layouts are considered in this work, as their heat transfer performance is generally superior to that of the in-line arrangement [7].

For serrated fins, a complete set of parameters to fully describe the geometry is illustrated in Figure 2.3b. These are the tube outside diameter d_o, fin pitch s_f, fin height h_f, segment width w_s and serration height h_s. Note that a solid fin can be described as a special case of serrated fins by setting $h_s = 0$.

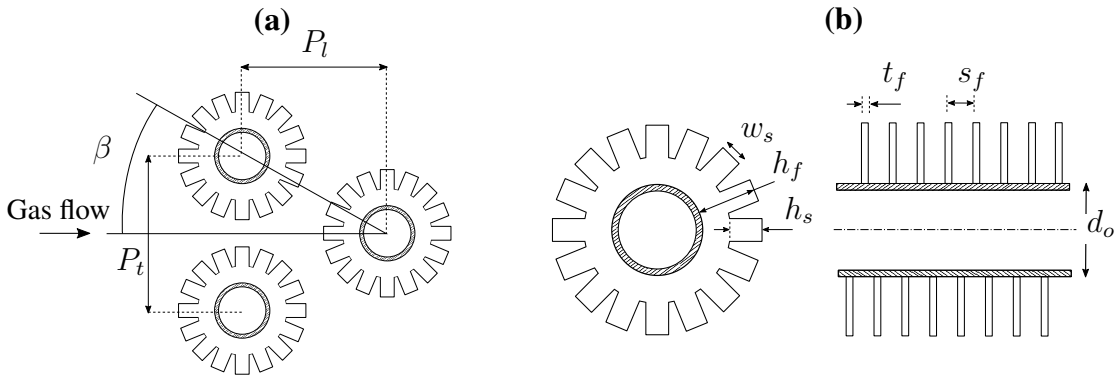


Figure 2.3 – Geometrical parameters for **(a)** tube bundle layout and **(b)** finned tube.

2.3 Flow characteristics

Experimental studies on flow conditions, heat transfer and temperature distributions have been performed for solid fins, such as Kröckels & Kottke [9] and Hu & Jacobi [10], but to the author's knowledge no corresponding studies are available for serrated fins.

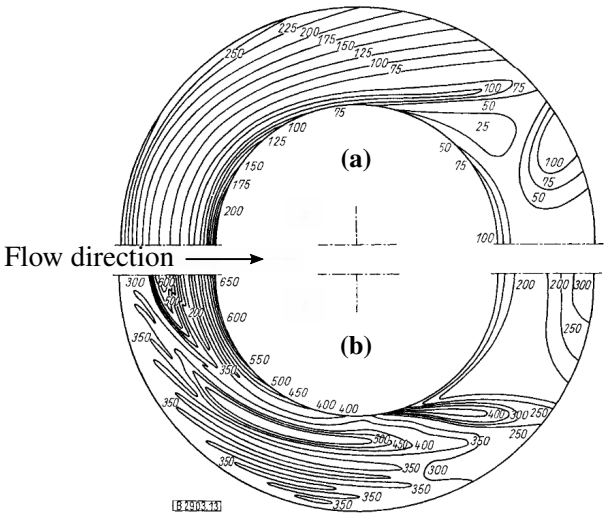


Figure 2.4 – Isolines for mass transfer-coefficient with flow direction from left to right, from Krötcke & Kottke [9]. The top half **(a)** shows the distribution for Re = 1 940, while the bottom **(b)** is for Re = 9 700.

Figure 2.4 shows the distribution of the mass transfer coefficient, which is directly proportional to the heat transfer coefficient (the heat and mass transfer analogy), for a tube with solid annular fins [9]. It is evident that the heat transfer coefficient is not uniformly distributed, and is considerably lower in the wake of the tube, due to less mixing in this region.

One of the mechanisms that yields increased heat transfer from finned tubes is the generation of *horseshoe vortices*. These have been observed in both experimental [11] and numerical studies [12], resulting from an adverse pressure gradient due to an obstacle (the fins) mounted perpendicular to a wall (tube surface) bounded flow, illustrated in Figure 2.5. Horseshoe-vortices are found in both laminar, transitional and turbulent flow, and can display both steady and transient, fluctuating behaviour. Though horseshoe-vortices contribute to improved heat transfer through mixing, increased wall friction contributes to additional pressure drop [13]. The trace of a horseshoe vortex can also be seen clearly in Figure 2.4b, at the edge of the wake.

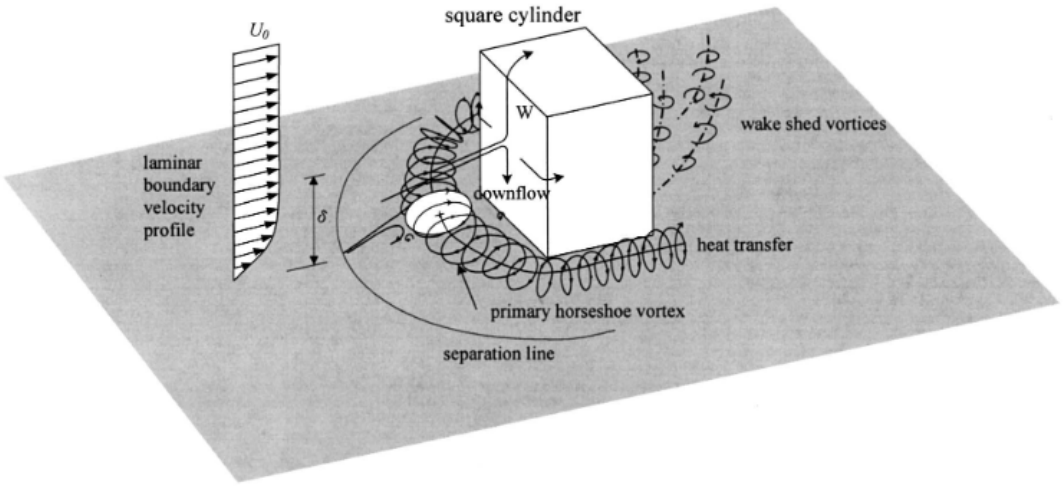


Figure 2.5 – Formation of a horseshoe vortex around a square cylinder mounted on a wall, from Lin et al. [14].

2.4 Fin efficiency

The fin efficiency η_f is a common measure for the performance of extended surfaces, and is defined as the ratio between the actual heat transfer and the maximum possible heat transfer, viz.

$$\eta_f = \frac{\dot{Q}_f}{\dot{Q}_{f,\infty}}, \tag{2.1}$$

where $\dot{Q}_f$ is the actual heat transferred from the fin, while $\dot{Q}_{f,\infty}$ is the heat transfer for the fin if it had infinite thermal conductivity, which is equivalent to the entire fin being at the tube wall temperature T_w. The use of the fin efficiency can ease the calculation of the heat transfer coefficient of a surface with fins, as the fins can be thought of as additional resistance to heat transfer. This means that knowledge of the fin surface temperature is not needed; the fin base temperature is enough to calculate the heat transfer rate.

In real-world flows, fin efficiencies are difficult to measure accurately for complex surfaces, as local variations of both temperature and heat fluxes occur. A one-dimensional conduction analysis of extended surfaces can in these cases be used to give good approximations of fin efficiencies by providing an analytical solution to the conduction equation. Assuming a fin at temperature $T_f(x)$, where x is the wall-normal direction, the 1D conduction equation reads,

$$\frac{d^2T_f}{dx^2} + \left(\frac{1}{A_{c,f}}\frac{\mathrm{d}A_{c,f}}{\mathrm{d}x}\right)\frac{\mathrm{d}T_f}{\mathrm{d}x} - m \cdot (T_f - T_\infty) = 0, \tag{2.2}$$

$$m = \sqrt{\frac{\alpha_o \cdot P}{\lambda_f \cdot A_{c,f}}}, \tag{2.3}$$

where α_o is the fin side heat transfer coefficient, P is the perimeter of the fin, λ_f is the thermal conductivity of the fin and $A_{c,f}$ is the cross-sectional area of the fin [15]. This differential equation is commonly prescribed the boundary conditions $T_f(0) = T_w$ and $\partial_x T_f(h_f) = 0$, which corresponds to wall temperature at base and insulated fin tip, respectively. By assuming a rectangular fin with constant cross-sectional area, $A_{c,f} = w_f \cdot t_f$ and $P = 2(w_f + t_f)$, the following expression for the theoretical fin efficiency is obtained,

$$\eta_{f,\mathrm{th}} = \frac{\tanh(m \cdot h_f)}{m \cdot h_f}, \tag{2.4}$$

where h_f is the fin height and m is given in Equation (2.3). As this is a solution for straight rectangular fins, it can be applied to fully serrated fins ($h_s = h_f$). For solid annular fins on circular tube, the theoretical fin efficiency can also be derived from Equation (2.2), and reads

$$\eta_{f,\mathrm{th}} = \frac{2r_o}{m\left(r_f^2 - r_o^2\right)} \frac{I_1(mr_f)\,K_1(mr_o) - I_1(mr_o)\,K_1(mr_f)}{I_0(mr_o)\,K_1(mr_f) + I_1(mr_f)\,K_0(mr_o)}, \tag{2.5}$$

where I_n and K_n are the modified Bessel functions of first and second kind, respectively. $r_o = d_o/2$ is the tube outside radius and $r_f = r_o + h_f$ is the fin radius [15].

Though Equations (2.4) and (2.5) are able to estimate η_f with acceptable accuracy in many cases, care should be taken to make sure that the assumptions being made in its derivation are valid. These assumptions are,

1. Steady state
2. Homogeneous fin material
3. Uniform heat transfer coefficient
4. No temperature gradient in the transverse direction of the fin
5. Uniform bulk fluid temperature
6. Uniform tube wall temperature
7. No heat sources in the fin
8. Adiabatic fin tip
9. No contact resistance between fin and tube wall

Though most assumptions listed above are valid for many situations of interest, this is not always the case. To circumvent the adiabatic fin tip assumption (no. 8), Harper & Brown

[16] proposed a modification of the theoretical solution by adding half the fin thickness to the fin height to model fin tip heat transfer, viz.

$$\eta_{f,\text{th}} = \frac{\tanh(m \cdot (h_f + t_f/2))}{m \cdot (h_f + t_f/2)}. \tag{2.6}$$

The same correction can be applied to Equation (2.5) as well, by replacing the fin radius r_f with a corrected fin radius $r_{f,c} = r_f + t_f/2$. To account for the non-uniform distribution of the heat transfer coefficient, as discussed in Section 2.3, (which violates assumption no. 3) Weierman [17] proposed the following correction of the theoretical fin efficiency,

$$\frac{\eta_f}{\eta_{f,\text{th}}} = \begin{cases} 0.7 + 0.3 \cdot \eta_{f,\text{th}}, & \text{for solid fins } (h_s = 0), \\ 0.9 + 0.1 \cdot \eta_{f,\text{th}}, & \text{for serrated fins } (h_s > 0), \end{cases} \tag{2.7}$$

where η_f is the corrected fin efficiency. It is evident that the corrected fin efficiency is reduced compared with the theoretical, and that it is reduced most for the solid fins. This is expected, since serrated fins increase mixing in the flow, improving heat transfer compared to the solid fins.

2.5 Empirical correlations

Heat transfer and pressure drop correlations have been proposed by several authors based on experimental data, such as Weierman [17], Hashizume [18] and Kawaguchi [19]. Most correlations are formulated as functions of the Reynolds number Re and a set of geometry parameters.

The Reynolds number is used to characterize the flow rate through the tube bank, and is defined as

$$\text{Re} = \frac{d_o u_{F_{\min}}}{\nu}, \tag{2.8}$$

where $u_{F_{\min}}$ is the characteristic velocity scale, and is defined as the average velocity in the minimum free flow area $F_{\min} = \min(F_t, F_d)$, as shown in Figure 2.6. The use of d_o as length scale is the most common choice, but other scales have been used by some authors.

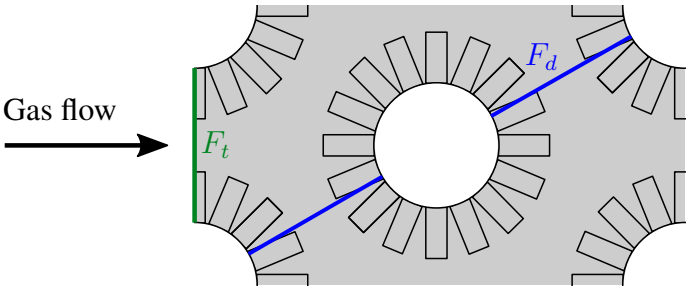


Figure 2.6 – The transversal and diagonal flow areas F_t and F_d inside a staggered tube bank.

The pressure drop is normalized by the dynamic pressure to form the Euler number,

$$\text{Eu} = \frac{\Delta p}{\frac{1}{2}\rho u_{F_{min}}^2} \tag{2.9}$$

where Δp is the pressure drop per tube row and $\frac{1}{2}\rho u^2_{F_{min}}$ is the dynamic pressure based on $u_{F_{\min}}$. Heat transfer is normalized to form the Nusselt number Nu, defined as,

$$\mathrm{Nu} = \frac{d_o \alpha_o}{\lambda}, \tag{2.10}$$

where α_o is the *outside heat transfer coefficient*, defined in Section 4.7 and λ is the thermal conductivity of the fluid. The empirical correlations are usually providing expressions for the quantities in Equations (2.9) and (2.10), being functions of Re.

The Weierman correlation

The Weierman correlation [17] will be used for comparison in this work, and is expressed both for solid and serrated fins. The expression for pressure drop is,

$$\begin{aligned}\mathrm{Eu} = &\left[0.28 + \frac{32}{\mathrm{Re}^{0.45}}\right] \cdot \sqrt{\frac{d_f}{d_o}} \cdot \\ &\left[0.11 \cdot \left(0.05 \cdot \frac{P_t}{d_0}\right)^{\left(-0.7 \cdot \left(h_f/s_f\right)^{K_1}\right)}\right] \cdot \\ &\left[1.1 + \left(1.8 - 2.1 \cdot e^{-0.15 \cdot N_l^2}\right) \cdot \left(e^{-2 \cdot P_l/P_t}\right) - \right. \\ &\left. \left(0.7 - 0.8 \cdot e^{-0.15 \cdot N_l^2}\right) \cdot \left(e^{-0.6 \cdot P_l/P_t}\right)\right],\end{aligned} \tag{2.11}$$

where N_l is the number of tube rows in the fin-tube bank and $d_f = d_o + 2h_f$ is the fin diameter. The Nusselt number is in turn expressed as,

$$\begin{aligned}\mathrm{Nu} = &\left[0.25 \cdot \mathrm{Re}^{0.65} \cdot \mathrm{Pr}^{1/3}\right] \cdot \sqrt{\frac{d_f}{d_o}} \cdot \left(\frac{T_b}{T_w}\right)^{0.25} \cdot \\ &\left[K_2 + K_3 \cdot e^{\left(-K_4 \cdot h_f/s_f\right)}\right] \cdot \\ &\left[0.7 + \left(0.7 - 0.8 \cdot e^{-0.15 \cdot N_l^2}\right) \cdot \left(\mathrm{e}^{-P_l/P_t}\right)\right],\end{aligned} \tag{2.12}$$

where $\mathrm{Pr} = \mu \cdot \lambda / c_p = \nu/\alpha$ is the Prandtl number and T_b and T_w is the temperature of the bulk fluid and the tube wall, respectively. The values for the constants K_1, K_2, K_3 and K_4 are determined by whether the fins are solid or serrated, and are listed in Table 2.1.

Table 2.1 – Values of coefficients used in the Weierman correlations, Equations (2.11) and (2.12).

Fin type	K_1	K_2	K_3	K_4
Solid	0.20	0.35	0.65	0.25
Serrated	0.23	0.55	0.45	0.35

2.6 Numerical studies

This section will present a literature review of CFD studies that have been performed for flow in fin-tube banks.

Jang et al. [20] performed both experiments and numerical modelling of laminar flow through four tube rows in a staggered layout. The numerical model was able to accurately predict pressure drop, but over-estimated heat transfer by 20 - 30%.

Mon [21] performed turbulent simulations for 23 different tube layouts, both in-line and staggered, with number of tube rows ranging from 2 to 6. The model was able to qualitatively describe the flow, but no comparison with experimental data was done. New correlations based on the numerical results and available experimental data was proposed. In a later paper, Mon & Gross [22] compared the results from selected layouts with existing correlations, where they were found to agree from $\pm 15\%$ to over 50%.

Torresi et al. [23] were among the first to model flow in tube banks with serrated fins. They simulated only one tube row, without modelling heat transfer, and then used equivalent porous medium zones in order to model the full HRSG. The results were not compared with experimental data, but showed good agreement with a proprietary 1D code.

McIlwian [24] compared the performance of solid and serrated fins in a single tube row, and gave qualitative insight into how serrated fins improve heat transfer compared with solid fins. In a later study [25], McIlwian extended the model and looked at the effects of adding a 2^{nd}, 3^{rd} and 4^{th} row. The results were compared with correlations, but neither of the studies were validated with experimental data.

As with Jang et al., laminar flow was also assumed by Lemouedda et al. [26], where fin tube bundles for Re between 600 and 2 600 were investigated. Fins with and without serration were compared, and the effect of twisting of the serrated fins was also investigated. No comparisons with experiments were made.

Hofmann & Walter [12] performed simulations and experiments for both solid and serrated fins, with both helical and angular fin attachment. Both local and overall heat transfer and pressure drop was investigated for turbulent flow with Re ranging from 3500 to 50 000. Results showed good agreement with experimental data, being within $\pm 15\%$ uncertainty.

Ó Cléirigh and Smith [27] investigated the effects of degree of serration, modelling fully serrated, partially serrated and solid fins. They found that the Nusselt number increased with 23% from partially to fully serrated fins, a distinction that is not made in most correlations. However, no validation against experimental data was performed in this study either.

Where the previous studies all have used standard inlet-outlet boundary conditions in the stream wise direction, Martinez et al. [28] utilized periodic boundary conditions also in this direction, thus assuming fully periodic flow. Local flow features were compared with experimental measurements, but global heat transfer and pressure drop were only compared with selected correlations, though with good agreement. In a consecutive paper [29], they went on to model six tube rows using standard inlet-outlet boundary conditions, and showed that the velocity, temperature and turbulence fields in fact display periodic behaviour after the third tube row.

Lindqvist & Næss [30] also used a periodic domain model, and applied it to four different cases, both serrated and solid fins. For one of them, a full domain model with eight tube rows was also considered, which was shown to match very closely with the periodic domain model. All four cases were validated against experimental data, and were found to be within 15% for both heat transfer and pressure drop. In addition, three correlations and two fin efficiency corrections were compared with the numerical and experimental results and their performances assessed. In a later study [31], the authors used the numerical model was extended to transient flows in order to investigate vortex shedding in fin-tube

banks, but heat transfer was not modelled. The model was able to predict the vortex shedding frequency with reasonable accuracy, though experimental data for comparison was scarce.

In a recent study, Salinas-Vázquez et al. [32] performed a transient LES simulation of a serrated fin-tube bank, using a periodic domain to investigate both heat transfer and pressure drop. A detailed description of both the mean and transient flow field was given. It was shown that an asymmetric vortex pattern occurred in the wake of the tube and the generation of counterrotating horseshoe vortices between the fins was also observed. The results were validated against experimental data.

3 COMPUTATIONAL FLUID DYNAMICS

The open-source CFD software OpenFOAM v2106 [33] is used in this work, and has been run simulations on the Idun-cluster [34]. OpenFOAM is a `C++` library for developing numerical solvers for continuum mechanics problems, but also provides pre-compiled solvers and utilities for many cases. Over the last years few years, OpenFOAM has become increasingly popular in the CFD field, and is continuously upgraded and expanded by developers and the user community.

OpenFOAM uses the *finite volume method*, a method where the physical domain is decomposed into small volumes called *cells*, which together makes up the *numerical grid* (sometimes referred to as the mesh). Each cell in the grid holds a value for each of the solution variables of interest, such as velocity, pressure and temperature. The governing equations are discretized using the grid and numerical schemes, and is transformed into a linear system of equations where the unknowns are the flow variables in each of the cells. This system is then solved numerically, usually in an iterative procedure, and a discrete solution is obtained.

3.1 Governing equations

To model flow and heat transfer, the governing equations of heat and fluid flow need to be solved. The equations are the continuity, momentum and energy equations, and are statements of fundamental conservation laws, and their derivation can be found in most standard fluid mechanics and CFD textbooks, such as Ferziger & Peric [35], Veersteg & Malalasekera [36] or Moukalled et. al [37]. These are partial differential equations, which are generally non-linear and with complex solution domains and boundaries. They can therefore rarely be solved analytically, but are rather solved numerically. The discretization and subsequent solution of the equations will be dealt with in the sections following this one.

Continuity equation

The continuity equation governs mass conservation for a fluid, and it simply states that the divergence of the velocity field should be zero throughout the domain, viz.

$$\nabla \cdot \mathbf{u} = 0, \tag{3.1}$$

where $\mathbf{u} = (u, v, w)$ is the velocity field and $\nabla = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right)$ is the nabla operator. Note that Equation (3.1) assumes incompressible flow, i.e. constant density, an assumption that will be discussed in Section 4.3.

Momentum equation

The momentum equation describes momentum conservation for a fluid, and is an alternative formulation of Newton's 2$^{\text{nd}}$ law. Assuming a Newtonian fluid (stresses in the fluid is proportional with the strain rate), the momentum equation is formulated as follows,

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\frac{1}{\rho}\nabla p + \nabla \cdot \left[\nu_{\text{eff}} \left(\nabla \mathbf{u} + (\nabla \mathbf{u})^{\top}\right)\right] + \mathbf{S}, \tag{3.2}$$

where p is the pressure field, $\mathbf{S}$ are momentum sources and ν_{eff} is the effective kinematic viscosity, which will be discussed in more detail in Section 3.5. In words, Equation (3.2) states that the change in momentum (the left-hand side) must equal the sum of pressure forces ($-\nabla p$), viscous forces ($\nabla \cdot \left[\nu_{\text{eff}} \left(\nabla \mathbf{u} + (\nabla \mathbf{u})^{\top}\right)\right]$) and any additional momentum forces ($\mathbf{S}$), whose role will be discussed in Section 4.6.2. In the case of steady flow, the first term in Equation (3.2) would be zero and can be omitted.

Energy equation

The energy equation imposes energy conservation in the fluid, and is an alternative formulation of the 1st law of thermodynamics. Energy can be formulated in terms of different quantities, most commonly either specific internal energy e, specific enthalpy h or temperature T. The enthalpy formulation is used in the present work, and reads as follows,

$$\frac{\partial(\rho h + e_{\text{K}})}{\partial t} + \nabla \cdot (\mathbf{u}(\rho h + e_{\text{K}})) - \frac{\text{D}p}{\text{D}t} = -\nabla \mathbf{q}, \tag{3.3}$$

where $e_{\text{K}} = \frac{1}{2}\rho|\mathbf{u}|^2$ is the specific kinetic energy and $\mathbf{q}$ is the heat flux. The heat flux is obtained using Fourier's law, i.e. the heat flux is proportional to the temperature gradient, viz.

$$\mathbf{q} = \rho \alpha_{\text{eff}} \nabla h, \tag{3.4}$$

where $\alpha_{\text{eff}} = \lambda_{\text{eff}}/(\rho c_{\text{p}})$ is the effective thermal diffusivity. The definition of the effective thermal conductivity λ_{eff}, will be given in Section 3.5. Additional heat sources can be included, e.g. viscous dissipation, but these are commonly omitted as they are negligible compared with the other terms. When the energy equation is solved, the temperature field T can be obtained using the following thermodynamic relation,

$$dh = T\, ds + \frac{1}{\rho} dp, \tag{3.5}$$

where s is the specific entropy. Assuming constant specific heat capacity c_p, at constant pressure, Equation (3.5) can be used to derive the following relation between specific enthalpy and temperature,

$$h = h_{ref} + c_p(T - T_{ref}), \tag{3.6}$$

where h_{ref} is the specific enthalpy at the reference temperature T_{ref}.

3.2 The numerical grid

In a typical CFD study, it is not uncommon that most of the time is spent on generating the numerical grid. Having a high quality grid is of great importance for the rest of the simulation, and will improve both accuracy, stability and computational time. By using a grid with more cells, the accuracy of the solution is increased, but this comes with increased computational cost. It is therefore desirable to construct a grid where the resolution is high enough to obtain solutions of acceptable accuracy, but at the same time do not exceed the computational limitations at hand.

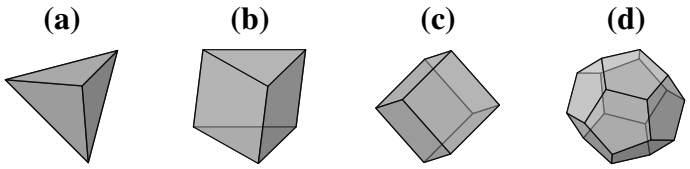


Figure 3.1 – Some of the different cell types typically used in a numerical grid, **a)** tetrahedron, **b)** prism, **c)** hexahedron and **d)** polyhedron, which is the most general cell, having an arbitrary number of faces.

OpenFOAM's grid handling is very flexible, and allows for cells ranging from tetrahedron to general polyhedral cells, as shown in Figure 3.1, and thus permits the use of both structured and unstructured grids, or a combination of the two. It should be noted that

all OpenFOAM meshes are in principle unstructured, since mesh connectivity is stored in all cases. To ensure that the grid is of sufficiently high quality, OpenFOAM provides the `checkMesh` utility in order to check that all the grid metrics are within acceptable tolerances, some of which are listed in Table 3.1.

Table 3.1 – Selected mesh check criteria in OpenFOAM.

Name	Criterion	Description
Non-orthognality	$< 70°$	Angle between face normal and vector connecting cell centre with neighbour cell center, see Figure 3.2a.
Skewness	< 4	Ratio of $\|\mathbf{e}\|$ and $\|\mathbf{d}\|$, see Figure 3.2b.

The cell non-orthogonality can play a large role in the accuracy and stability of the solution. Different CFD software use different definitions of the cell non-orthogonality, and OpenFOAM defines it as the angle θ between the vector between a cell and its neighbour and the normal of the face between them, as illustrated in Figure 3.2a. The non-orthogonality must be taken into account when discretizing diffusion terms in the transport equations, where the face normal gradient is being calculated. This gradient is decomposed into an orthogonal and non-orthogonal component, where the non-orthogonal component is added as a source term to the linear equation system. If this term gets too large, i.e. the non-orthogonal angle is too high, the solution will become unstable [37].

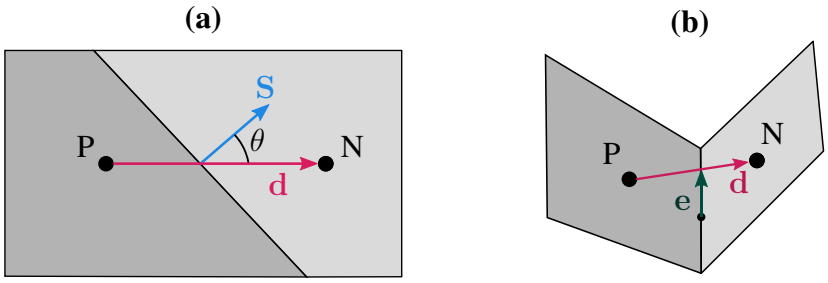


Figure 3.2 – **(a)** Cell non-orthogonality is defined by the angle θ between the face normal $\mathbf{S}$ and the vector $\mathbf{d}$ between the neighbouring cell centres. **(b)** The skewness is the length $|\mathbf{e}|$ between the face centre and the point where the vector $\mathbf{d}$ crosses the face.

Another critical factor is the skewness of the grid. In OpenFOAM this is defined as the distance between the face centre and the point where the vector between the cells that share the face cross that face, as illustrated in 3.2b. Finally, this distance is normalized by the cell connecting vector, i.e. $\mathbf{e}/|\mathbf{d}|$. When discretizing the transport equations, the transported variable (velocity, temperature, etc.) often needs to be estimated at the faces. This value should represent the average value at the face, and should therefore be located at the face centre. However, when using linear interpolation between the cells, the actual location of the intersection point is often displaced from the face centre (the vector $\mathbf{e}$ in Figure 3.2b). This lowers the accuracy of the face interpolation, and should therefore be kept at a minimum [37].

3.3 Discretization

When discretizing the governing equations, each term must be discretized with a numerical scheme. Using the numerical grid, a linear equation is obtained for each unknown variable ϕ, e.g. h in the energy Equation (3.3), in each cell. Each equation contains the unknown in the owner cell ϕ_p, as well as a combination of the unknowns in the neighbour cells ϕ_{nb}. Generally, the equation for a cell can be expressed as,

$$a_p\phi_p + \sum_{nb} a_{nb}\phi_{nb} = b_p, \tag{3.7}$$

where a_p is the coefficient for the unknown in the owner cell, ϕ_p, and the coefficients a_{nb} is are for the neighbour cells with unknowns ϕ_{nb}. The source term b_p is the sum of all constant terms left after the discretization. The choice of discretization schemes will determine the coefficients and the source term in Equation (3.7). Generally, the discretization schemes come with a trade-off between stability and accuracy, and many variants have been developed to achieve the best possible compromise between the two. The linear upwind and Crank-Nicolson schemes will be presented next.

The linear upwind scheme

The linear upwind scheme (also called second order upwind scheme) is used to discretize convective terms, i.e. terms on the form $\nabla \cdot (\phi \mathbf{u})$, where ϕ is a scalar quantity being convected in a velocity field $\mathbf{u}$. In the finite volume method, this amounts to using the cell values to estimate values of ϕ at the cell faces, ϕ_f. The scheme is illustrated in Figure 3.3 using a 1D grid, including a central (owner) cell and its upwind and downwind neighbours. The upwind direction is given by the velocity direction in the central cell, and as flow goes from left to right in Figure 3.3, the upwind cell will be to the left.

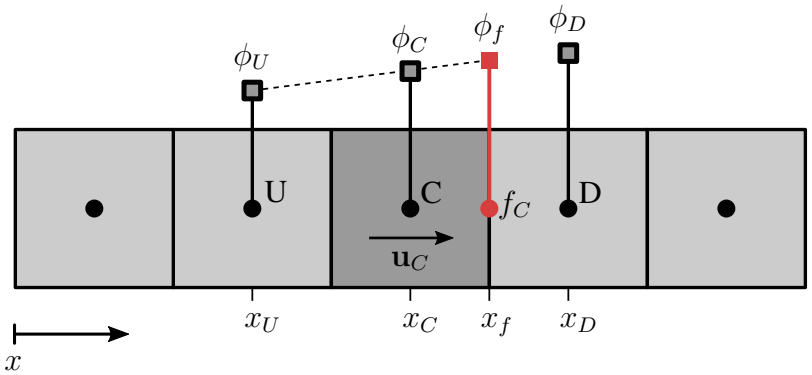


Figure 3.3 – The linear upwind interpolation profile. The scheme uses the owner (C) and upwind (U) cell values to extrapolate the value at the face shared by the owner and downwind neighbour (D), ϕ_f.

The linear upwind scheme uses the cell values in the owner cell (C) and the upwind cell (U) to extrapolate the face value ϕ_f. For a one dimensional grid, it can be expressed as,

$$\phi_f = \phi_C + \frac{\phi_C - \phi_U}{x_c - x_u}(x_f - x_c), \tag{3.8}$$

i.e. a standard linear interpolation, where the quantities are defined in Figure 3.3. It can be shown that this scheme is second order accurate, i.e. the discretization error is on the order Δx^2, where Δx is the cell size [37]. This makes it more accurate than the standard upwind scheme, which is only first order accurate.

The Crank-Nicolson method

The Crank-Nicolson method is used to model the transient terms in partial differential equations, such as the governing equations of fluid flow. Consider for now an ordinary differential equation with initial condition, viz.

$$\frac{\mathrm{d}\phi}{\mathrm{d}t} = f(t, \phi(t)), \quad \phi(t_0) = \phi_0, \tag{3.9}$$

where ϕ_0 is the value at the initial time t_0. To approximate the solution in the next time step $\phi(t_{n+1}) = \phi^{n+1}$, Equation (3.9) can be integrated from t_n to $t_{n+1} = t_n + \Delta t$. The following relation is then obtained,

$$\phi^{n+1} = \phi^n + \int_{t_n}^{t_{n+1}} f(t, \phi(t))dt, \tag{3.10}$$

where the integral on the right side of the equation needs to be approximated numerically. Evaluating f at the current time step t_n results in the *explicit Euler scheme*,

$$\phi^{n+1} = \phi^n + f(t_n, \phi_n)\Delta t. \tag{3.11}$$

Here, the unknown value at the new time is given explicitly, and can be calculated directly with values from the known time t_n. This method is first order accurate, but stable if the CFL-condition is fulfilled when applied to partial differential equations in flows dominated by convection,

$$\mathrm{Co} = \frac{U\Delta t}{\delta x}, \tag{3.12}$$

where Co is the Courant number, U the convective velocity and Δx is the spacial discretization size. By choosing another approximation of the integral in Equation (3.9) a second order method is obtained,

$$\phi^{n+1} = \phi^n + \frac{1}{2}\left[f(t_n, \phi_n) + f(t_{n+1}, \phi_{n+1})\right]\Delta t, \tag{3.13}$$

which is the *Crank-Nicolson method* when applied to partial differential equations. This method evaluates f in both the old and the new time step, which makes this an implicit method. The method can be shown to be unconditionally stable, but oscillations may still arise in the solution for large time steps [35]. OpenFOAM provides a linear blend between the explicit Euler and Crank-Nicolson to adjust the blend between accuracy and stability.

3.4 Solution algorithms

Since the momentum equation, Equation (3.2), contains a non-linear advective term, the governing equations need to be solved iteratively. Another complicating factor is the pressure field p, whose gradient is included in the momentum equation, but there is no separate equation for p. Therefore, special treatment for the coupling between the velocity field and pressure field is required. This is achieved by the widely used SIMPLE and PISO algorithms, which are used for steady and transient flows, respectively. Detailed explanations and their derivations can be found in most CFD-textbooks, but an overview of the two algorithms is given next.

3.4.1 The SIMPLE algorithm

The SIMPLE (Semi-Implicit Method for Pressure Linked Equations) algorithm was proposed by Patankar & Spalding [38] in 1972 for steady and incompressible flows, and is still widely used today. The algorithm, shown in Figure 3.4, starts by solving the discretized momentum equation using the initial fields (box A). Note that this velocity field does not satisfy continuity yet, and is therefore referred to as the *momentum predictor* step.

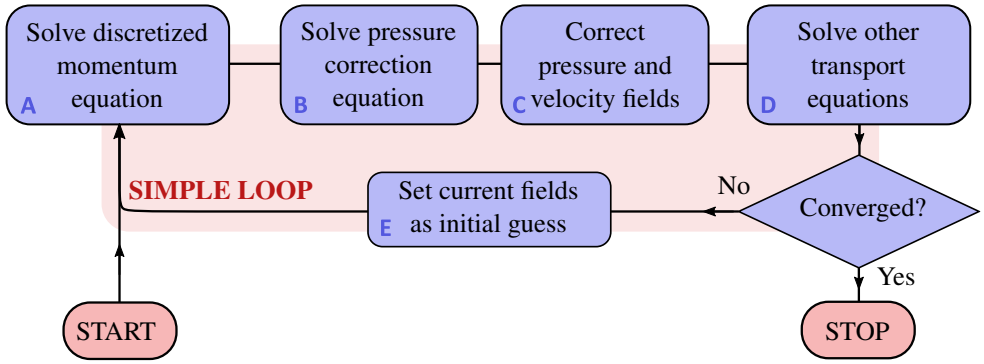


Figure 3.4 – The main steps of the SIMPLE algorithm.

To circumvent the problem of the pressure-velocity coupling, the discretized continuity equation is combined with the discretized momentum equation to create a pressure correction equation. The pressure correction equation is solved to obtain a correction of the pressure field (box B), and used to correct the pressure *and* the velocity fields such that continuity is satisfied (box C). After this, other transport equations, such as the energy equation and turbulent transport equations, are solved (box D). Last, if the solution is not converged, the current fields are used as the initial guesses (box E) and the loop is repeated. To avoid stabilities when iterating, the fields are often under-relaxed between the loops, either explicitly or implicitly. [35]

3.4.2 The PISO algorithm

The PISO algorithm was proposed by Issa [39] to handle transient flows, both compressible and in-compressible. The algorithm is illustrated in Figure 3.5, and it is evident that the structure is similar to that of the SIMPLE-algorithm in Figure 3.4. One of the main differences is the PISO-loop (or pressure corrector loop), where the pressure correction equation is solved and $\mathbf{u}$ and p is corrected repeatedly. Usually, if the time step is small enough, two or three corrector steps are sufficient. The other transport can then be solved after (box D), and the solution is advanced to the next time step without having to do any outer loops, as in the SIMPLE algorithm. The PISO algorithm is not as prone to instabilities as steady simulations performed with the SIMPLE algorithm, since the transient terms contributes to stabilize the linear equation systems, and under-relaxation is therefore often not needed. [35]

3.5 Turbulence modelling

Though the governing Equations (3.1 - 3.3) fully describe flow and heat transfer, the turbulent nature of the flow requires additional transport equations to be solved. In

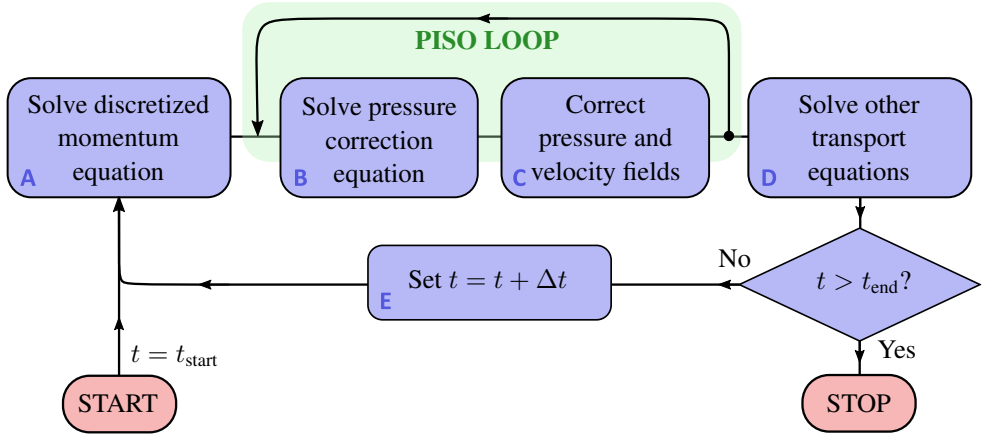


Figure 3.5 – The main steps of the PISO algorithm, highlighting the inner (PISO).

principle, the momentum and continuity equations would yield a turbulent solution, but to fully resolve the turbulent features of the flow is for all practical purposes not achievable, due to the large range of scales that the numerical domain would have to resolve, both spatially and temporally. Such simulations are termed direct numerical simulations (DNS), and require immense computational resources and/or countless CPU-hours.

Instead, the turbulent features are either modelled completely (Reynolds-Averaged Navier-Stokes, RANS) or partially (Large-Eddy Simulation, LES). The former will be used in this work, as this requires the least amount of computational resources and have been shown to yield reasonable accuracy for flow in fin-tube banks [8]. However, care should always be taken when using turbulence models, or as George Box put it [40]; *All models are wrong, but some are useful.*

The complete derivation of the RANS equations will not be presented her, but can be found in detail in turbulence text books, such as Pope [41]. The main idea is to decompose the flow variables, such as velocity, pressure and temperature, into a steady and a fluctuating component, i.e. $\mathbf{u}(x, y, z, t) = \bar{\mathbf{u}}(x, y, z) + \mathbf{u}'(x, y, z, t)$, and correspondingly for pressure and temperature[1]. When inserting this decomposition into the governing equations (3.1 - 3.3) we introduce more unknowns than we have equations. However, only the average fields are of interest, and the full equations are therefore averaged, which leads to most of the fluctuating terms being dropped from the equations. The continuity equation (3.1) remains the same, but the momentum and energy equations are left with one additional diffusive term each on the left-hand side of the equation, $-\nabla(\rho\overline{\mathbf{u}'\mathbf{u}'})$ and $-\nabla(\rho\overline{\mathbf{u}'h'})$, respectively. The tensor $-\rho\overline{\mathbf{u}'\mathbf{u}'}$ is commonly referred to as the Reynolds (or turbulent) stress, as it is analogous to the viscous stress tensor in the momentum equation. The Boussinesq hypothesis is used to couple the Reynolds stress with strain rate of the flow field, in analogy with the viscous stresses in a Newtonian fluid, viz.

$$-\rho\overline{\mathbf{u}'\mathbf{u}'} = \rho\nu_{\text{t}}\left[\nabla\mathbf{u} + (\nabla\mathbf{u})^{\top}\right] - \frac{2}{3}\rho\mathbf{I}k, \tag{3.14}$$

where ν_{t} is the *turbulent kinematic viscosity*, $\mathbf{I}$ is the identity matrix and k is the turbulent kinetic energy. The second term on the right-hand side in Equation (3.14) is either

[1]This notation is only used in this section to clarify the decomposition into a mean and fluctuating components. In remainder of the text, $\mathbf{u}$ refers to the mean velocity.

incorporated into the pressure field or neglected in some turbulence models [37]. The problem has now been reduced to finding the quantity ν_t. Note that even though Equation (3.14) is analogous to viscous stresses, ν_t is a property of the flow field, whereas the molecular viscosity ν is a property of the fluid. Several turbulence models are available to obtain ν_t, each with their strengths and weaknesses. The model selected in this work is the *Spalart-Allmaras* turbulence model. This is a one-equation model, i.e. it adds one additional transport equation to the governing equations, given as,

$$\frac{D}{Dt}(\rho\tilde{\nu}) = \nabla\cdot(\rho D_{\tilde{\nu}}\tilde{\nu}) + \frac{C_{b2}}{\sigma_{\nu_t}}\rho|\nabla\tilde{\nu}|^2 + C_{b1}\rho\tilde{S}\tilde{\nu}\,(1-f_{t2}) - \left(C_{w1}f_w - \frac{C_{b1}}{\kappa^2}f_{t2}\right)\rho\frac{\tilde{\nu}^2}{\tilde{d}^2} + S_{\tilde{\nu}}, \tag{3.15}$$

where the *modified turbulent viscosity* $\tilde{\nu}$ is transported quantity. The turbulent viscosity is then obtained as $\nu_t = \tilde{\nu}f_{v1}$, where f_{v1} is a model function. All model constants and functions in are available in the original paper by Spalart & Allmaras [42], or in the OpenFOAM documentation[2]. Earlier turbulence models, such as $k-\epsilon$ and $k-\omega$ were found to have shortcomings with the presence of adverse pressure gradients, and the Spalart-Allmaras model was developed to alleviate this problem. The $k-\omega$ SST model has also shown good performance for flow over finned tubes, but as this is a two-equation model it is computationally more demanding than the Spalart-Allmaras model.

The turbulent heat flux $-\rho\overline{\mathbf{u}'h'}$ can be modelled similar to the turbulent viscosity, namely analogous with Fourier's law in Equation (3.4), viz.

$$-\rho\overline{\mathbf{u}'h'} = \alpha_\mathrm{t}\nabla h, \tag{3.16}$$

where $\alpha_\mathrm{t} = \lambda_\mathrm{t}/(\rho c_\mathrm{p})$ is the *turbulent thermal diffusivity* and λ_t is the turbulent thermal conductivity. This is usually obtained by assuming that the turbulent Prandtl number, defined as

$$\mathrm{Pr}_\mathrm{t} = \frac{\nu_\mathrm{t}}{\alpha_\mathrm{t}}, \tag{3.17}$$

is constant. Thus, when ν_t is computed, the turbulent thermal diffusivity is simply obtained as $\alpha_\mathrm{t} = \mathrm{Pr}_\mathrm{t}\cdot\nu_\mathrm{t}$. For this work, a turbulent Prandtl number of $\mathrm{Pr}_\mathrm{t} = 0.85$ is used, as have been done in previous studies [31]. The sum of the molecular and turbulent thermal diffusivities is the effective thermal diffusivity α_eff, which was introduced in Equation (3.4).

3.6 Wall treatment

An in-depth discussion of the behaviour of turbulent wall-bounded flows is outside the scope of this thesis, but a brief background is needed in order to justify the treatment of walls in this work. The turbulent boundary layer is usually divided into several sub-layers, distinguished by one or more defining properties. The wall distance, which is by convention named y, is normalized by the *viscous length scale* $\delta_\nu = \nu\sqrt{\rho/\tau_w}$, where τ_w is the wall shear stress. The wall distance is then measured in *wall units*, defined as

$$y^+ = \frac{y}{\delta_\nu} = \frac{u_\tau y}{\nu}, \tag{3.18}$$

[2] https://www.openfoam.com/documentation/guides/latest/doc/guide-turbulence-ras-spalart-allmaras.html

where $u_\tau = \sqrt{\tau_w/\rho}$ is the friction velocity. Similarly, a normalized velocity is formed by scaling with the friction velocity, viz.

$$u^+ = \frac{u}{u_\tau}. \tag{3.19}$$

Very close to the wall, $y^+ < 5$, the turbulent stresses disappear due to the no-slip condition, and thus the molecular viscous stress dominates. This is the viscous sublayer, as seen in Figure 3.6.

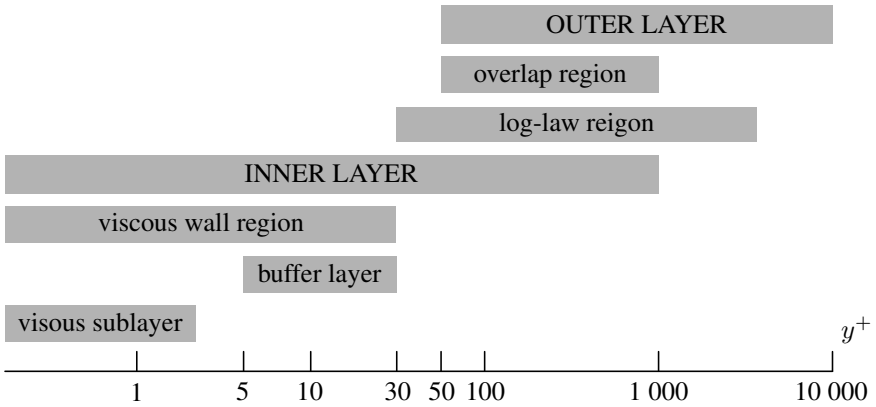


Figure 3.6 – The sub-layers and wall regions for turbulent channel flow, adapted from [41].

It can be shown [41] that in the viscous sublayer, u^+ is a closely approximated as a linear function of the wall unit y^+,

$$u^+ = y^+, \quad \text{for } y^+ < 5. \tag{3.20}$$

In the log-law region, where the turbulent stresses dominate, u^+ can be shown to have a logarithmic dependency of y^+,

$$u^+ = \frac{1}{\kappa} \ln y^+ + B, \quad \text{for } y^+ > 30, \tag{3.21}$$

where the precise values of the constants κ (the *von Kármán* constant) and B are debated in literature, but are generally within 5% of $\kappa = 0.41$ and $B = 5.2$ [41]. These expressions have been compared with both experimental and DNS data (Figure 3.8), and shows good agreement. In the buffer layer ($5 < y^+ < 30$) however, both viscous and turbulent stresses are of the same order of magnitude, and finding general theoretical solutions as in Equations (3.20) and (3.21) is difficult.

There are two main strategies when modelling turbulent boundary layers in CFD: 1) resolve the boundary layer or 2) model it using wall functions. The first is achieved by constructing a fine mesh close to the wall (Figure 3.7a), where y^+ of the first cell should be less than 1, and the cell growth rate should be moderate, ideally not growing more than 20% in the wall normal direction. This ensures that the numerical grid is fine enough to represent the boundary layer profile in the viscous sublayer, and the shear stress and heat transfer at the wall is obtained with good accuracy.

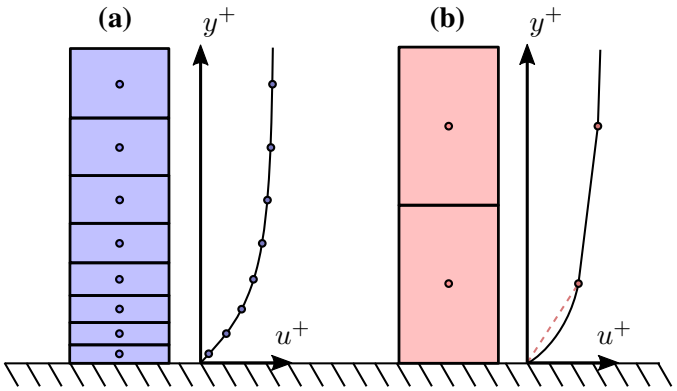


Figure 3.7 – The two approaches for dealing with turbulent boundary layers in CFD. **(a)** Resolution of boundary layer, and **(b)** modelled boundary layer using wall functions. The dashed pink line indicates the linear profile that would be used if wall functions were not adapted.

Modelling using wall functions usually require the first cell to be located outside the buffer layer ($y^+ > 30$), as modelling the flow in that region is difficult. This is however difficult to enforce in practical CFD simulations, and to circumvent this limitation, Spalding [43] fitted the following curve as a *unified wall function*,

$$y^+ = u^+ + \frac{1}{E}\left[\exp\left(\kappa u^+\right) - 1 - \kappa u^+ - \frac{1}{2}\left(\kappa u^+\right)^2 - \frac{1}{6}\left(\kappa u^+\right)^3\right] \tag{3.22}$$

where $E = 9.1$. Equation (3.22) is implicit in u^+, and thus needs to be solved iteratively. This profile however, is valid in both the viscous layer, buffer layer and the log-law region, and there is no restriction on the location of the first cell centre. This profile is strictly only valid for zero pressure gradient conditions, and is therefore not as accurate under other conditions.

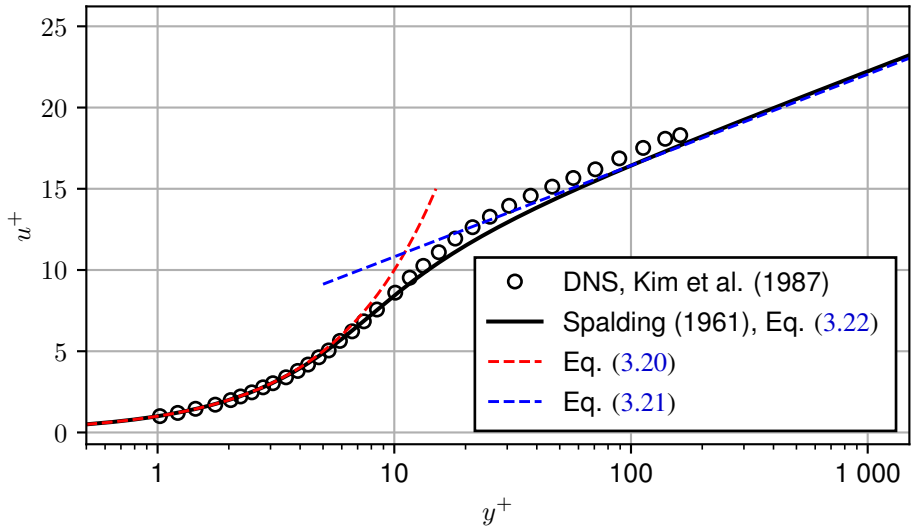


Figure 3.8 – Turbulent boundary layer profile, comparing the DNS solution from Kim et al. [44] with Spalding's unified wall function [43] and Eqs. (3.20) and (3.21). Note that the y^+-axis is logarithmic.

3.7 Errors in CFD

To access and identify the uncertainties and errors in CFD results can be challenging due to the complex solution process. The sources of errors in CFD can be classified in several ways, but they are usually divided into three main groups:

1. The *modelling error* is the error that comes from the formulation of the mathematical model, e.g. the energy and momentum equations or the boundary conditions that are imposed. For example, if a non-negligible term is neglected, it would lead to a model error. It is the error that would have been observed if the exact solution of the governing equations was compared with the actual flow that was modelled. This process is termed *validation*.

2. The *discretization error* is the error that stems from the transformation from continuous differential equation to linear equation systems with the domain divided into a finite grid and time steps. This is the error that would have been observed when comparing the solution of the discretized equations with the exact solution of the (continuous) governing equations. This process is termed *verification*.

3. The *iterative error* is the error that comes from the iterative solution procedure. In practice, the iterative process is terminated when the residuals of the linear equation system is deemed sufficiently small, as time and resources are often limited. This error is the difference between the iterative solution and the exact solution of the linear equation system.

An estimate of the spatial discretization error will be given in Section 5, using the Grid Convergence Index proposed by Roache et al. [45]. The method requires numerical solutions to be obtained using grids of different refinements, and are then used to give a quantitative estimate of the spatial discretization error.

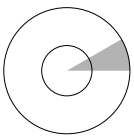
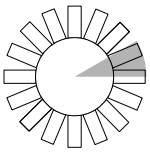
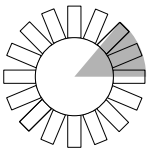
4 NUMERICAL METHOD

This section will describe the implementation of the numerical model, which includes both the grid generation procedure and the setup of the CFD simulation.

4.1 Selected geometries

The three tube and fin geometries considered in this study, listed in Table 4.1, are selected in order to span a sufficiently wide range of parameters to represent the possible designs encountered in compact heat exchangers. The selected geometries are retrieved from experimental studies in order to validate the numerical results. The first two geometries, N1 and N3, are geometries 1 and 3 from the experimental study by Næss [46], respectively. The two geometries from Næss are using the same serrated fin tube geometry, but with different layout angles, 30° and 50°, respectively. The third geometry, named H8, is geometry 8 from the experimental study by Holfeld [6], and is a solid finned tube with the same layout angle as N1.

Table 4.1 – Layout, fin and tube geometries that are modelled. The geometries are illustrated in the final row scaled 1:2, with the grey arcs indicating the layout angle β.

Name	H8	N1	N3
Type	Solid	Serrated	Serrated
P_t [mm]	38.7	46.1	70.6
P_l [mm]	33.5	39.9	29.6
d_o [mm]	13.5	20.89	20.89
h_f [mm]	10	8.61	8.61
h_s [mm]	-	8.61	8.61
s_f [mm]	2.81	5.08	5.08
t_f [mm]	0.50	0.91	0.91
w_f [mm]	-	3.97	3.97
β [°]	30	30	50
			

4.2 Numerical domain

This work will model gas flow over finned tubes including the heat transfer from the gas to the fins and the heat flow inside the fins themselves, i.e. conjugate heat transfer (CHT). This means that the numerical domain consists of two physical regions: a fluid region (the gas) and a solid region (the fins).

The earliest numerical simulations of flow over finned tube banks tended to model several tube rows in the flow direction, both for in-line and staggered layouts, while using periodic boundary conditions in the transverse direction [20, 22, 26, 27]. These setups have imposed standard inlet and outlet boundary conditions, with uniform profiles at the inlet and a fixed outlet pressure, with additional exit lengths to avoid backflow at the outlet.

Recent works by Maritnez et al. [28] and Lindqvist [8] have used periodic domains in order to reduce the computational cost of running the simulations. In addition to impose periodic boundary conditions in the transverse direction, the flow is also cyclic in the stream wise

direction. The domain can then be thought of as a ”unit cell” in an infinite tube bank, as illustrated in Figure 4.1. This is the approach used in this work, and the construction of the domain and its boundaries is based on the work of Lindqvist & Næss [30].

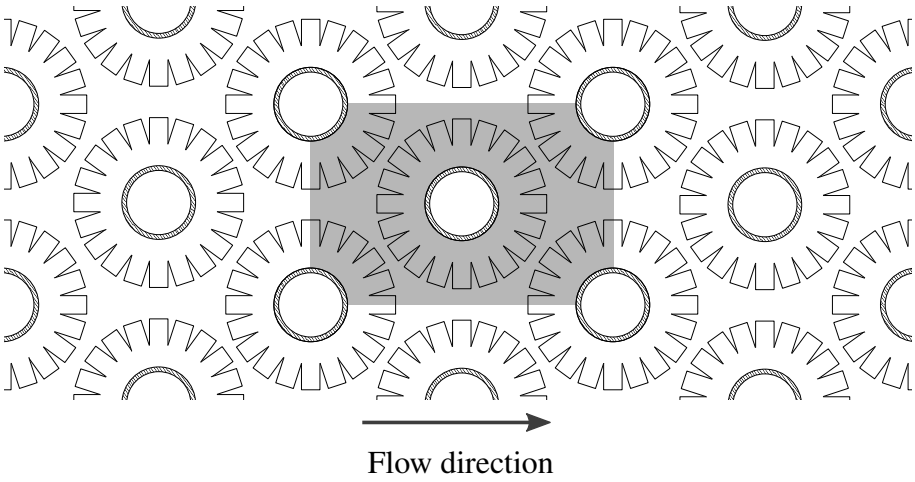


Figure 4.1 – The numerical domain (shaded in grey) inside an infinite tube bank, with flow direction from left to right.

Figure 4.2 shows the boundaries of the numerical domain, but note that it does not include the interface between the fluid and solid regions. Figure 4.2a shows the boundaries in the (transverse) y direction, and is constitutes by three pairs of cyclic boundaries (matched by colour). The triangular elements of these boundaries are a consequence of the helical fin geometry, which requires the domain to employ cut faces downstream of the tube, which are displaced by the fin pitch s_f distance.

4.3 Thermal properties

The unit cell has other benefits besides the reduced computational domain, namely the use of constant thermal properties. This is a reasonable assumption, as the temperature difference between inlet and outlet will be moderate, compared to the change in temperature over multiple tube rows. This allows for the use of the incompressible assumption in the governing equations, which makes them easier to solve, which further contributes to reduced computational cost. It also makes the velocity field decoupled from the temperature field, since neither ρ, c_p, λ nor ν will vary with temperature, which essentially makes T a passive scalar.

Table 4.2 – Thermal properties for dry air, carbon steel and aluminium 6060 at atmospheric pressure and 300 K [15, 47].

Property	Dry air	Carbon Steel	Aluminium	Unit
Thermal conductivity λ	$26.3 \cdot 10^{-3}$	48.5	210	W/(m·K)
Specific heat capacity c_p	1007	434	898	J/(kg·K)
Density ρ	1.1614	7854	2700	kg/m^3
Viscosity ν	$15.89 \cdot 10^{-6}$	-	-	kg/(m·s)

Both the fluid and solid regions are modelled using constant thermal properties. The fluid is modelled as dry air at atmospheric pressure and 300 K, which matches the experimental

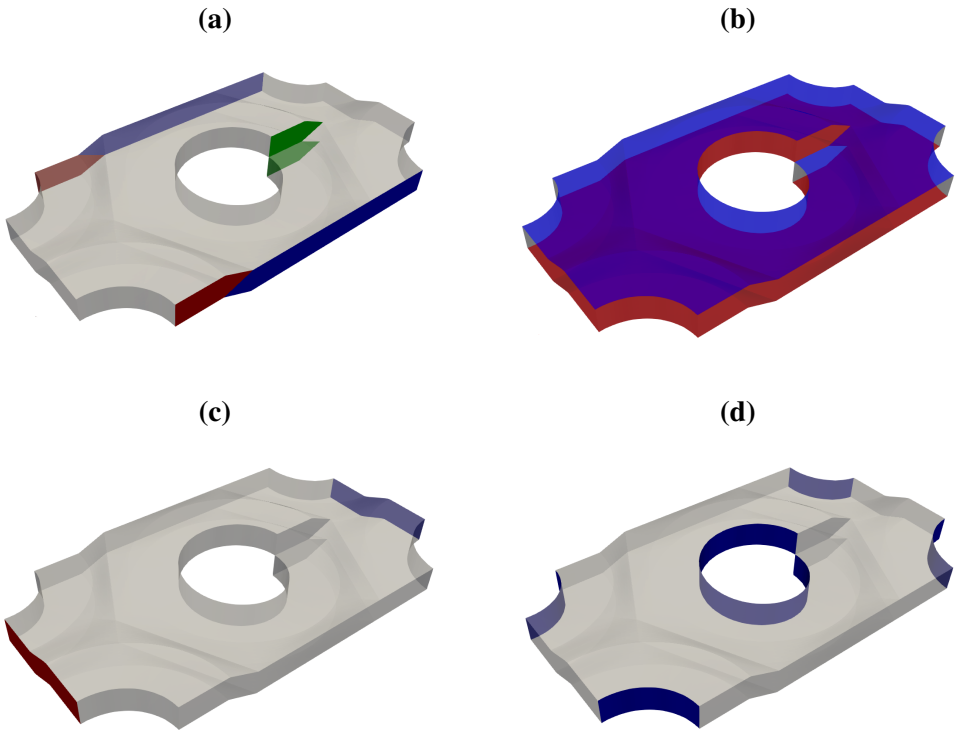


Figure 4.2 – The patches of the numerical domain. **(a)** Pairs of cyclic boundaries in the (transverse) y direction, matched by colour. **(b)** The top (■) and bottom (■) boundaries, which are the boundaries in the (transverse) z direction and constitute a cyclic pair. **(c)** The inlet (■) and outlet (■) boundaries **(d)** Wall boundaries ■. Note that the interface between the gas and fin regions are not shown here, but are also modelled as solid walls.

conditions closely, and the properties are listed in Table 4.2. The fins are modelled as carbon steel and aluminium (A6060) for the Næss (N1 and N2) and Holfeld (H8) cases, respectively.

4.4 Grid generation

As stated in Section 1.2, developing a parameterized grid generation procedure has been one of the main objectives of this work. Considerable time has been spent on developing a flexible, but yet robust, method for readily creating grids of high quality. This section will outline the structure of the grid generation procedure and highlight the most important features of the resulting grids.

4.4.1 Structure of the code

The open-source pre-processing software SALOME[3] was used for both geometry and grid generation. The software provides a Python-interface, which makes scripting of the entire grid generation process possible. The object-oriented capabilities of Python also allows for structuring the code hierarchically and minimizing code duplication. This enables completely parameterized grid generation, which is of great convenience when altering the geometric parameters for different fin geometries and tube layouts. An example of a geometry specification is given in Appendix B.1, which was used to generate the grid for

[3]The version used in this work is SALOME 9.7.0, and is available from https://www.salome-platform.org/.

geometry N1. After the grid is exported from SALOME in `unv`-format, it is converted to OpenFOAM format using the `ideasUnvToFoam` utility. One of the disadvantages with SALOME is the fact that polyhedral cells (the cells at the interface shown in box B in Figure 4.4) can not be exported using the `unv`-format, and therefore the mesh is exported with multiple distinct regions. To merge these regions, a custom OpenFOAM utility was written.

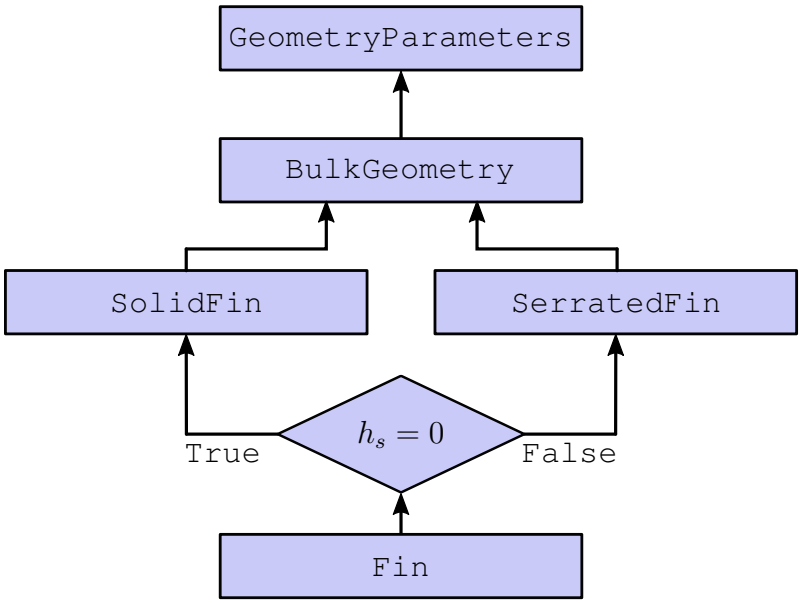


Figure 4.3 – Overall strucutre of the classes used in the grid generation procedure.

The main structure of the Salome Python classes are shown in Figure 4.3. The `GeometryParameters` class is used to read in the geometric parameters from an input file, and then calculate a number of derived parameters, such as layout angle and minimum free flow area. This class holds all geometrical information of the fin-tube geometry, which gives the possibility of implementing empirical correlations that can be called as methods using, using only the Reynolds number as an input. Note that this class does not depend on any of the Salome Python-libraries, and derived parameters and correlations can therefore be obtained without having Salome installed.

In the `BulkMesh` class, which inherits from `GeometryParameters`, the bulk-flow geometry and mesh is created. This class therefore requires installation of Salome. This class is further inherited by the `SolidFin` and `SerratedFin` classes, which generates the mesh for the actual fin and inter-fin geometries. Finally, the top-level class `Fin` is used to determine whether to invoke the `SolidFin` or `SerratedFin` class, based on the value of h_s.

4.4.2 Grid features

A grid generated from geometry N1 is showed in Figure 4.4. The grid is block structured and dominated by hexagonal cells wherever possible, but this is not achieved in the serrated regions, where wedge-shaped prism cells also are included (box A in Figure 4.4). To enable cyclic boundaries, the number of serrated elements need to be multiple of 4. This is enforced by adjusting the serration width w_s accordingly, but also minimizing the distance between the fins. Note also that the fins are in fact not fully serrated, but that a small height of fin height is added to the base between the fin segments.

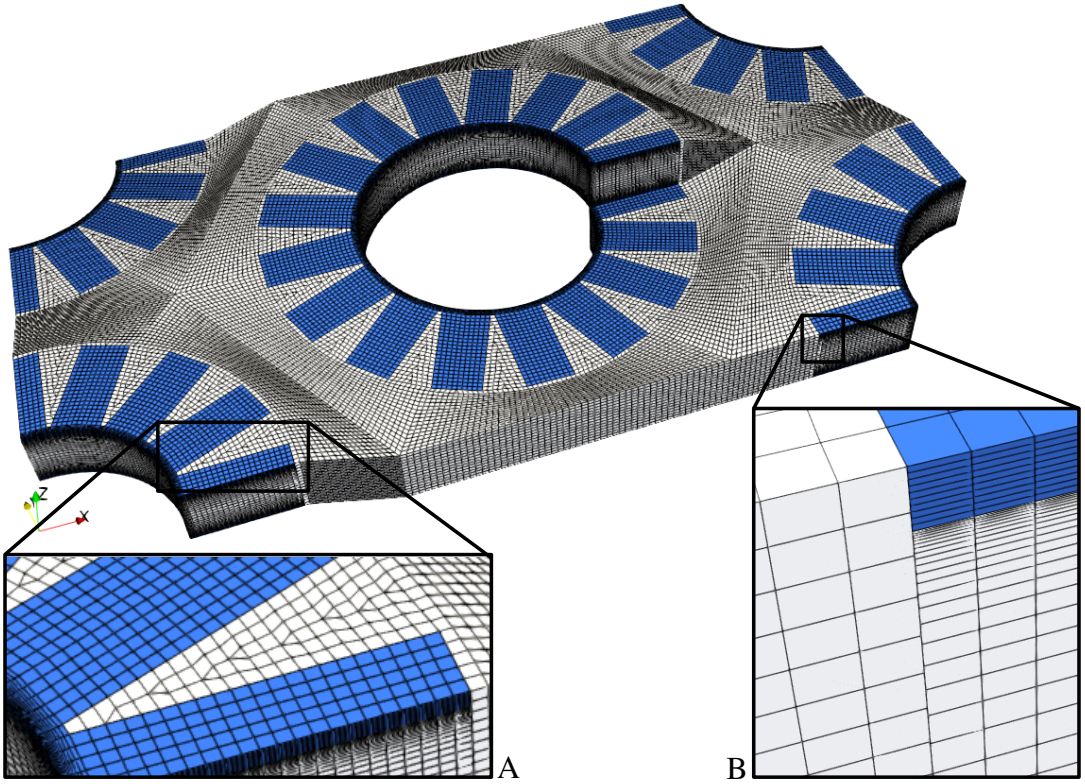


Figure 4.4 – Grid from geometry N1, with gas (■) and fin (■) regions. The unstructured mesh is highlighted in box A. The polyhedral cells connecting the inter-fin and bulk flow regions are displayed in box B.

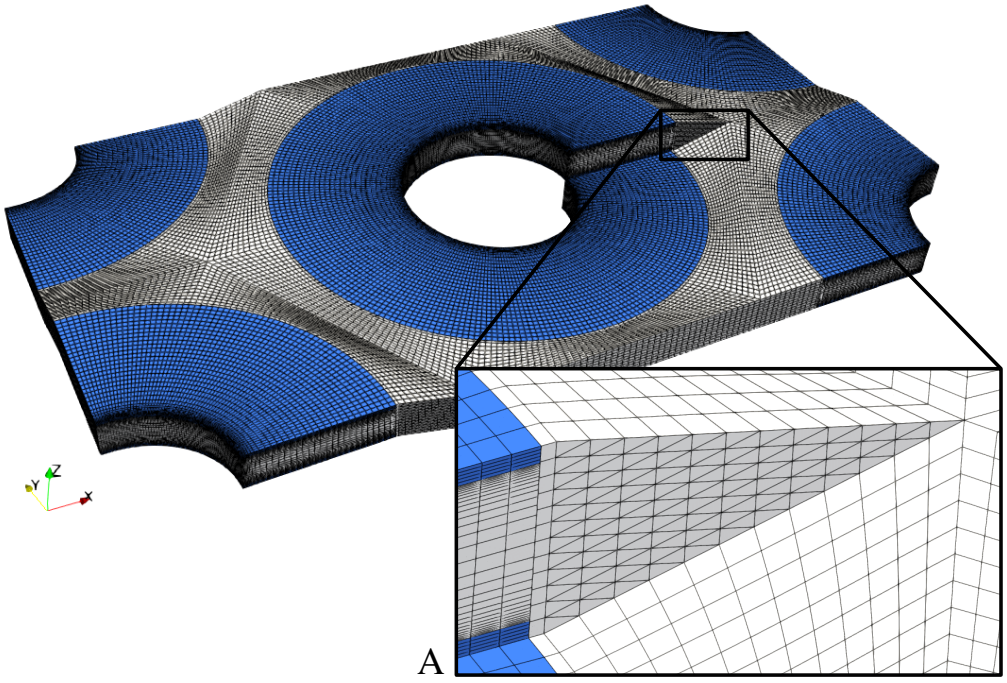


Figure 4.5 – Grid from geometry H8, with gas (■) and fin (■) regions. Box A shows the wedge-shaped faces in the cut-plane downstream of the central tube.

A mesh of geometry H8 is shown in Figure 4.5, where all cells are hexagonal as there are no serrated fins. Due to the helical attachment of the fins, a perfect alignment of hexagonal cells is not possible in the ”cut plane” downstream of the central tube (the green boundaries in Figure 4.2a), as seen in box A. The cell faces at these boundaries are split on the diagonals to generate wedge-shaped faces in order to maintain perfectly matching cyclic boundaries.

The boundary layer was resolved in the inter-fin regions, and ensuring a first cell height of $y^+ < 1$ yields a complete resolution of the turbulent boundary layer (Section 3.6). The boundary layer cells are set to a growth rate of 20 %, which is commonly used in boundary layer meshes, with at least 15 cells across the boundary layer and a smooth transition to the coarser cells in the inter-fin region. On the fins sides and ends however, the boundary layer was not resolved, and thus the turbulent boundary layer is modelled with wall functions. This choice was necessary because of the transition to the bulk mesh, and is one of the main weaknesses of this grid. The transition between the inter-fin grid and the bulk flow grid is shown in box B in Figure 4.4, and shows how polyhedral cells are used to merge the two mesh regions. This transition can cause highly non-orthogonal and skew faces, but care is taken to ensure that they are always within the OpenFOAM grid-criteria, given in Table 3.1 in Section 3.2.

4.5 Boundary conditions

Most boundaries of the domain are cyclic for all flow variables, except for the tube and fin surfaces which obviously are modelled as walls. The general boundary conditions will be explained here, but the treatment of the inlet and outlet boundary conditions for velocity and pressure will be treated in the next section.

4.5.1 Walls

When solving the continuity and momentum, the fin and tube surfaces are modelled using the no-slip and no penetration boundary conditions, i.e. $\mathbf{u}_w = \mathbf{0}$ where the subscript w denotes the wall. The wall-normal pressure gradient is set to zero, i.e. $\partial p/\partial n|_w = 0$.

For the modified turbulent viscosity $\tilde{\nu}$, the wall boundary conditions should also be zero, i.e. $\tilde{\nu}_{\text{t,w}} = 0$. This is done on all walls except for the fins sides, where the grid doesn't resolve the boundary layer. Here, the Spalding wall function is used for ν_{t} directly to impose the theoretical turbulent boundary layer profiles onto the flow field, as discussed in Section 3.6. This does not yield as accurate results as the fully resolved boundary layers do, but as these surfaces constitute only a minor part of the total wall area (5 - 17 %), the use of wall functions here is deemed acceptable.

The temperature at the walls is set to a uniformly fixed temperature T_w at the tube walls and the base of the fins. At the interface between the gas and fins, the boundary conditions are defined such that conserved heat flux through the interface as well as identical temperature at the interface for both regions are achieved, viz.

$$T_{w,\text{fluid}} = T_{w,\text{solid}}, \quad \mathbf{q}_{w,\text{fluid}} = -\mathbf{q}_{w,\text{solid}}. \tag{4.1}$$

4.5.2 Inlet and outlet for temperature

When solving the energy equation in a cyclic domain, adding a source term to the equation is possible. However, the formulation of the source term depends on the wall boundary conditions, and is rather involved for constant wall temperature boundary conditions

compared with constant heat flux boundary condition [48]. In addition, the presence of a solid-fluid interface (CHT) complicates matters more. The method employed here utilizes a cyclic boundary condition, but with an offset between the inlet and outlet. This allows for using constant wall temperature boundary conditions on the tube wall and cyclic inlet and outlet, *without* having to add source terms in the energy equation.

Due to heat being transferred from the fluid to the fin and tube surfaces, the temperature at the outlet will be lower than that of the inlet. Therefore, the offset between the inlet and outlet is defined such that the inlet always is kept at a constant average temperature T_{in} while at the same time maintaining the constant wall temperature T_w, viz.

$$T_{\text{inlet}} = T_w + \left(\frac{T_{in} - T_w}{\overline{T_{\text{outlet}}} - T_w} \right) (T_{\text{outlet}} - T_w), \tag{4.2}$$

where T_{inlet} and T_{outlet} are the inlet and outlet temperature fields, respectively, and $\overline{T_{\text{outlet}}}$ is the average outlet temperature. The implementation of this boundary condition is done using the boundary condition `groovyBCJump` from the third-party OpenFOAM library `swak4foam`[4]. This boundary condition allows for the addition of a "jump" field between the boundaries, while the boundaries still remain cyclic.

4.6 Solution method

As will be evident when presenting the results in Section 6, stable solutions proved difficult to obtain. For this reason, three different solution strategies have been tested and evaluated, as described next. The main differences between the strategies are steady or transient solution and the implementation of the stream wise cyclic boundary condition. The linear solvers and under-relaxation factors used in the simulations are found in Appendix A.

4.6.1 Steady state simulation with mapped inlet and outlet

The first strategy is based on the method used by Lindqvist & Næss [30] and is a steady state approach with mapping between the inlet and outlet boundaries, rather than an actual cyclic boundary condition. The inlet is prescribed a fixed velocity profile and the outlet with a fixed pressure, i.e. standard inlet-outlet boundary conditions. The outlet for velocity is using the `inletOutlet` boundary condition to force $\mathbf{u} = \mathbf{0}$, which is commonly imposed on outflow boundaries to stabilize in case of backflow. After a certain number of iterations, the velocity profile from the outlet is mapped onto the inlet profile and scaled to a desired average value. The same is done with the pressure, only it is mapped from the inlet to the outlet. To improve stability this is done in a staggered fashion, i.e. if velocity is mapped at iteration 1000, 2000, 3000, and so on, the pressure will be mapped at 1500, 2500, 3500, and onwards, as illustrated in Figure 4.6. The period between mapping was found through trial and error, and solution process was not found to be overly sensitive to choice of period.

The other fields are mapped from outlet to inlet after each iteration, and the temperature field is scaled to ensure a prescribed average inlet temperature, as described in Section 4.5.2. This procedure is then repeated until the fields converge. The SIMPLE-algorithm (Section 3.4) was used to obtain the steady solution.

4.6.2 Steady state with momentum source term

The second strategy was to use a source term in the momentum Equation (3.2), such that a true cyclic boundary condition can be used also in the stream wise direction. The cyclic

[4] https://github.com/wyldckat/swak4foam.

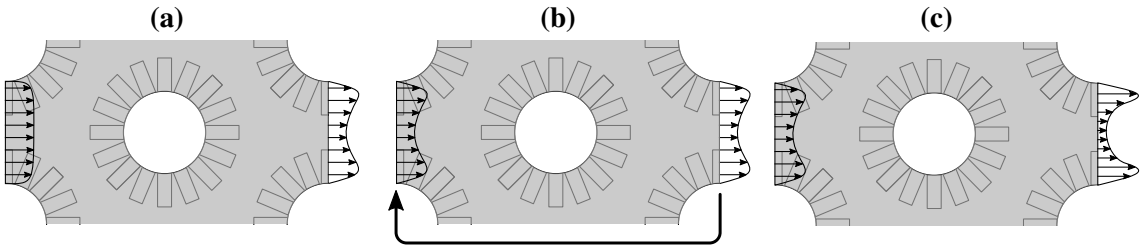


Figure 4.6 – Mapping between inlet and outlet, showing profiles **(a)** before mapping, **(b)** mapping and **(c)** after mapping. If the number of iterations i is a multiple of the period, $\mathbf{u}$ is mapped from outlet to inlet, while T and $\tilde{\nu}$ are mapped each iteration. Between the mapping of velocity, the pressure is mapped from inlet to outlet (the opposite direction of what is shown in **(b)**).

stream wise boundary conditions need additional treatment compared with the transverse boundary conditions, as there is no longer a prescribed inlet velocity and pressure outlet to drive the flow through the domain. The assumption that the flow field is cyclic does however introduce a source term in the momentum equations, as shown by Patankar & Liu [48]. Defining x to be the stream wise direction, the source term in Equation (3.2) takes the form

$$\mathbf{S} = \gamma \cdot \frac{\mathbf{u}_{\text{avg}}}{|\mathbf{u}_{\text{avg}}|} = (\gamma\ 0\ 0), \tag{4.3}$$

where γ is the imposed pressure gradient that drives the flow and $\mathbf{u}_s$ is the volumetric average velocity vector, which in this case is $\mathbf{u}_{\text{avg}} = (u_{\text{avg}}\ 0\ 0)$. γ can therefore be regarded as an assignable parameter to control the flow rate over the fins, or equivalently, the Reynolds number. However, as the pressure gradient is not known a priori, γ needs to be updated iteratively in order to achieve the desired flow rate. OpenFOAM provides this functionality through the `meanVelocityForce`[5] source. The cyclic boundary condition for the temperature field is scaled, as described in Section 4.5.2.

4.6.3 Transient with momentum source term

Finally, a transient simulation strategy was explored, in order to resolve temporal instabilities in the flow field. The stream wise cyclic boundary conditions were implemented using the previous method, i.e. with the momentum source term given in Equation (4.3). The initial condition was taken from the final iteration of the precious strategy, and the PISO algorithm was used with two inner corrector steps (PISO loops). The transient terms were discretized using a linear blend between Euler and Crank-Nicolson schemes, with Crank-Nicolson having a weight of 0.7 (Section 3.3). Using an adaptive time step, the maximum Courant number was always kept below 0.5.

To ensure fully developed conditions, a simulation time of at least 20 fluid exchanges is used, where the time of a fluid exchange t_{FE} is defined as,

$$t_{\text{FE}} = \frac{2 \cdot P_l}{u_{F_{\text{min}}}}. \tag{4.4}$$

4.7 Data reduction

To allow for direct comparison with experimental results, non-dimensional parameters are calculated using the numerical results. The Reynolds number is computed directly using

[5] https://www.openfoam.com/documentation/guides/latest/doc/guide-fvoptions-sources-mean-velocity-force.html

Equation (2.8).

Euler number

The pressure drop is normalized by the dynamic pressure to form the Euler number, given in Equation (2.9). Δp is the pressure drop per tube row, which in the mapped case is calculated as the difference between average pressure at the two cut planes showed in Figure 4.7, i.e. $\Delta p = p_1 - p_2$. If a momentum source term is used, Δp evaluated by multiplying the pressure gradient from Equation (4.3) with the longitudinal tube pitch P_l, i.e. $\Delta p = \gamma \cdot P_l$.

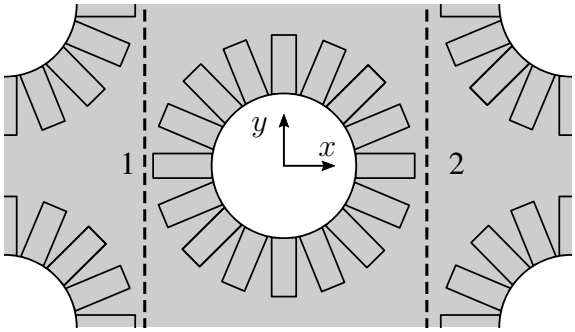


Figure 4.7 – The two cut planes (1 & 2) used to sample pressure and bulk temperature, located at $x = \mp\frac{1}{2}P_l$, respectively.

As the thermal properties are independent of temperature, the pressure drop corresponds to the *adiabatic pressure drop*. If the effect of temperature drop/increase was modelled, the change in density would contribute to additional pressure drop. However, most experiments studies perform pressure drop test under adiabatic conditions to account for this [6] or provides estimates for the adiabatic pressure drop [49], and the assumption of constant thermal properties is therefore suitable for comparison with experiments in this regard.

Nusselt number

The outside heat transfer coefficient α_o is calculated using the same procedure as Lindqvist [8], viz.

$$\alpha_o = \frac{\dot{Q}_{\text{tot}}}{[\eta_f A_f + A_t]\,\Delta T}, \tag{4.5}$$

where $\dot{Q}_{\text{tot}}$ is the total heat transferred to both the fin and tube surface, η_f is the fin efficiency and A_f and A_t is the surface area of the fin and tube, respectively. ΔT is the average temperature difference that drives the heat transfer between the gas and the surface of the finned tube. For multiple rows in a cross-flow configuration, the logarithmic mean temperature difference (LMTD) is an appropriate choice of mean temperature, but for a cyclic domain with only one tube row, a local arithmetic average temperature is used,

$$\Delta T = \frac{1}{2}\left[(T_1 - T_w) + (T_2 - T_w)\right], \tag{4.6}$$

where T_1 and T_2 is the average gas temperature in cut planes 1 and 2 (Figure 4.7), respectively. In experimental studies, the fin efficiency η_f is commonly estimated using theoretical and corrected predictions, and therefore the calculated heat transfer coefficient will depend on the chosen fin efficiency calculation. For this reason, the fin efficiency is

calculated using the same method as done in the experiments that are used for validation. For N1 and N3, Næss [46] used the theoretical fin efficiency in Equation (2.6) and the Weierman correction in Equation (2.7). Holfeld [6] used the theoretical fin efficiencies for solid angular fins in Equation (2.5), also with a corrected fin height, but did not use the Weierman correction. Since the expressions for η_{th} are functions of α_o itself, they need to be found iteratively.

Since the CFD results provide a full description of the temperature field of the fin surface, they allow for direct computation of the fin efficiency, as described by Ó Cléirigh and Smith [27],

$$\eta_f = \frac{\dot{Q}_{\text{actual}}}{\dot{Q}_{\text{ideal}}} = \frac{\alpha_o \int_{\text{fin}} (T_b - T_f)\,\mathrm{dA}}{\alpha_o A_f\,(T_b - T_t)} = \frac{T_b - T_{f,\text{avg}}}{T_b - T_t}. \tag{4.7}$$

The fin efficiency calculated with Equation (4.7) will be used to compare the theoretical and correected fin efficiencies to see how well they perform. The implementation can be found in Appendix B.2.

Finally, after obtaining η_f, the heat transfer coefficient α_o is scaled by d_o and λ to yield the Nusselt number, which was defined in Equation (2.10). Note that the results for heat transfer coefficients are presented using $\text{NuPr}^{-1/3}$, where $\text{Pr} = \alpha/\nu$ is the Prandtl number of the fluid. This is a common way of presenting the Nusselt number since $\text{Pr}^{1/3}$ is one of the factors in most of the empirical correlations.

5 GRID REFINEMENT STUDY

Ideally, grid refinement studies should be performed for all geometries at each Reynolds numbers in order to estimate the discretization error and ensure sufficient grid resolution. However, as this is too costly in regard to both time and computational resources, the procedure has been exemplified by performing a grid convergence study for geometry H8 at Re = 5 000, using three different grid resolutions with solutions from the second strategy from Section 4.6 (steady-state simulation with momentum source term). The grid convergence index (GCI) [50] was calculated using both the Eu and $\mathrm{NuPr}^{-1/3}$ as integral parameters. The Richardson extrapolates are also presented for both quantities.

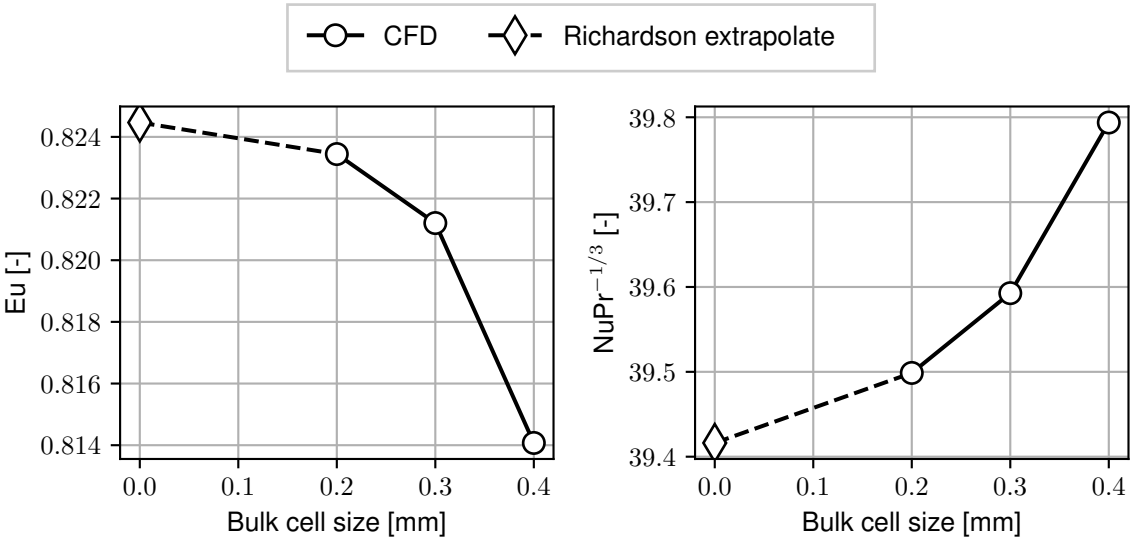


Figure 5.1 – Grid convergence for case H8 at Re = 5 000, with Richardson extrapolated values for Eu = 0.825 and $\mathrm{NuPr}^{-1/3} = 39.42$.

The results from the grid refinement study are summarized in Figure 5.1, and yielded an estimate of the spatial discretization error of 0.82% and 2.13% for Eu at the finest and medium grids, respectively. For Nu, the error is estimated to 0.96% and 2.89% for the finest and medium grids, respectively. This suggested that the medium grid, i.e. a bulk cell size of 0.3 mm, should be sufficient, which is used for all geometries in this work.

6 NUMERICAL RESULTS

Results from the three solution strategies described in Section 4.6 are presented in this section. The two steady-state strategies were run with all geometries (Table 4.1), while the transient strategy was only run with geometries H8 and N1. Several simulations were run using all three strategies, from which a representative selection are described in detail. The results of all cases will however be compared with experimental data and correlations, and the general performance of the different strategies for all geometries will be discussed in Section 7.

6.1 Steady-state simulations with mapped inlet and outlet

The iteration evolution from geometry N1 is shown in Figure 6.1, where a grid resolution of 0.3 mm is used for Re = 7 500. The period of velocity and pressure profile updates is set to 750 iterations, as is apparent from the peaks of the scaled residuals. It is evident that after a sufficient number of iterations, the pressure drop Eu and heat transfer $\mathrm{NuPr}^{-1/3}$ converges to fixed values. After the seven first profile updates the residuals are monotonically decreasing, but after this (around iteration 10 000) the residuals are observed to increase by several orders of magnitude and stabilizes on a high level. The same convergence behaviour was observed for the other investigated Re.

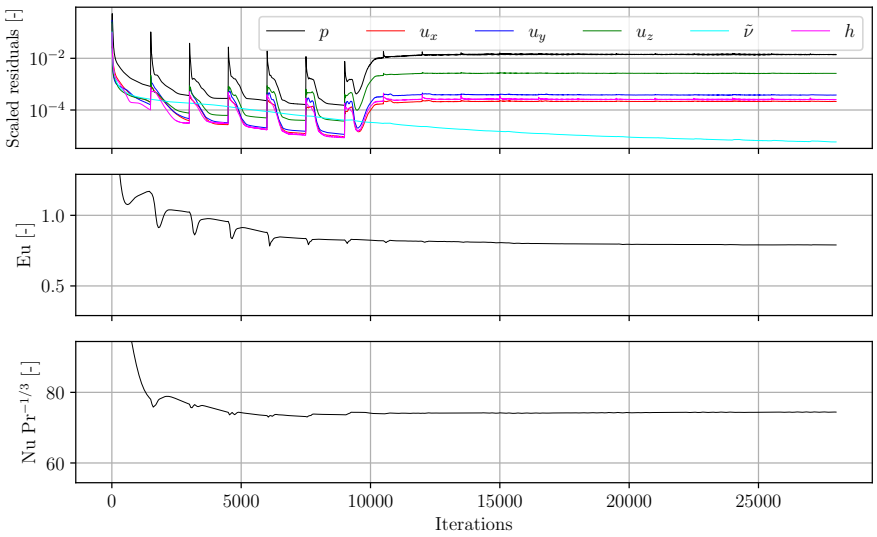


Figure 6.1 – Iteration evolution of scaled residuals, Eu and $\mathrm{NuPr}^{-1/3}$ for case N1 and Re = 7 500, using mapped inlet and outlet. At the final iteration, Eu = 0.79 and $\mathrm{NuPr}^{-1/3}$ = 74.2.

The iteration evolution from geometry H8 at Re = 7 500 with a period of 1 000 iterations is shown in Figure 6.2. As is clear from both the monotonically decreasing residuals and the stabilized Eu and $\mathrm{NuPr}^{-1/3}$, the solution is fully converged. The same convergence behaviour is observed for all investigated Re for H8. Similar behaviour is also displayed by geometry N3 (Figure 6.3), both for residuals, Eu and $\mathrm{NuPr}^{-1/3}$. The same trends are observed for all investigated Re as well.

A comparison of the numerical results with experimental data for all three cases is shown in Figure 6.4. Generally, the results are agreeing to within 20% for most cases, but the degree of accuracy is varying. For N1, the $\mathrm{NuPr}^{-1/3}$ (Figure 6.4a) is close to the experimental

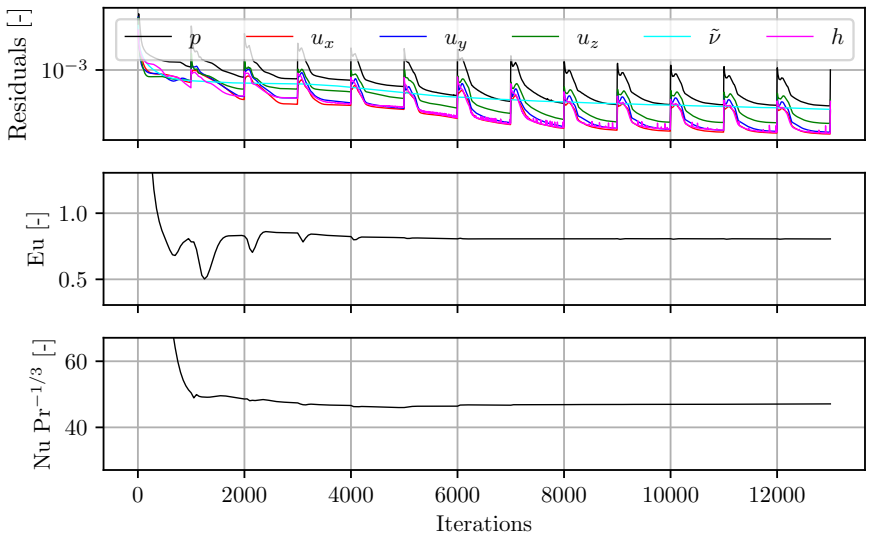


Figure 6.2 – Iteration evolution of scaled residuals, Eu and $\mathrm{NuPr}^{-1/3}$ for case H8 and Re = 7 500, using mapped inlet and outlet. At the final iteration, Eu = 0.81 and $\mathrm{NuPr}^{-1/3}$ = 47.66.

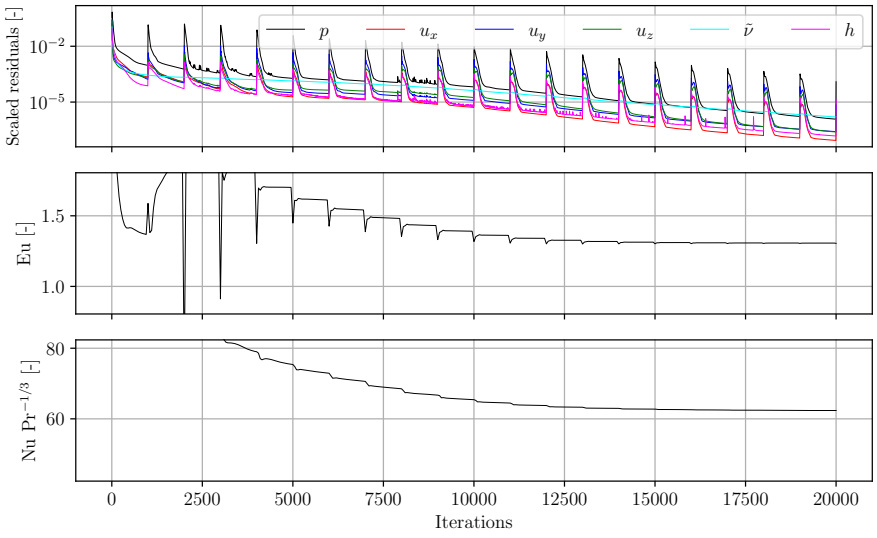


Figure 6.3 – Iteration evolution of scaled residuals, Eu and $\mathrm{NuPr}^{-1/3}$ for case N3 and Re = 7 500, using mapped inlet and outlet. At the final iteration, Eu = 1.31 and $\mathrm{NuPr}^{-1/3}$ = 62.36.

results and follows their trend, but Eu (Figure 6.4b) shows more deviation and does not seem to follow the experimental trend as clearly. However, both are predicting equally well as or better than the Weierman correlation. For N3, it is evident that $\mathrm{NuPr}^{-1/3}$ is predicted fairly well (Figure 6.4c), within 15% of the experimental data, in strong contrast to Eu (Figure 6.4d) which deviates as much as 78% from experimental values, and does not seem to follow the same trend. For H8, $\mathrm{NuPr}^{-1/3}$ (Figure 6.4e) is also very close to the experimental results but shows slight deviation from the trend. Eu (Figure 6.4f) shows more deviation from the experimental results, but are still providing better predictions than the Weierman correlation.

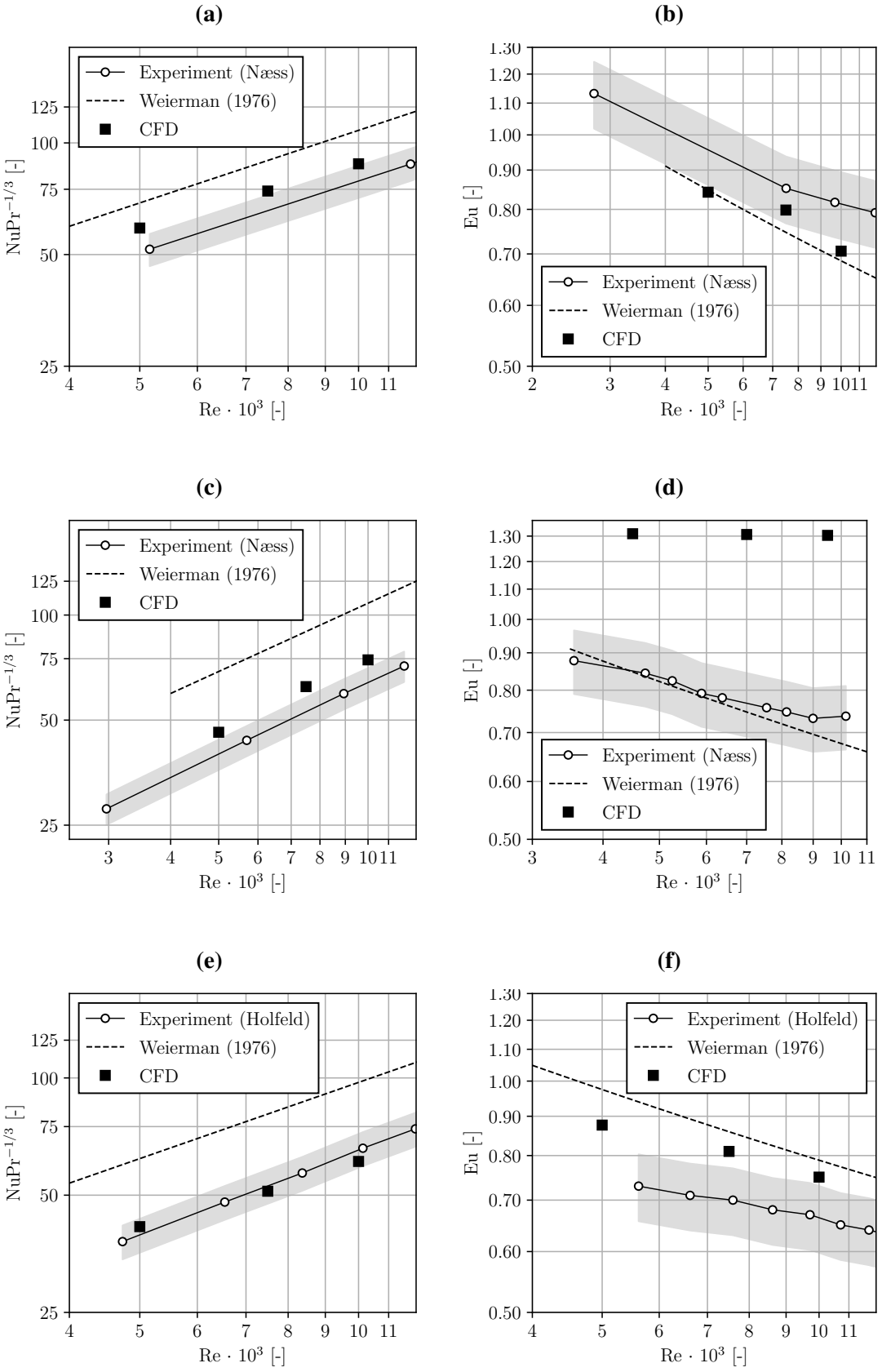


Figure 6.4 – Results from the mapped inlet/outlet strategy for $\mathrm{NuPr}^{-1/3}$ (a, c, e) and Eu (b, d, f) for geometries N1 (a, b), N3 (c, d) and H8 (e, f). The shaded areas are representing a $\pm 10\%$ range of the experimental data.

Streamlines coloured by temperature of geometries N1 and N3 are displayed in Figure 6.5 and 6.6, respectively. The flow direction is from left to right, and both are sampled at the final iteration using the mapped inlet and outlet strategy. It is evident that the wake from the central tube in geometry N3 is divided into two pairs of circulating regions. This stands in contrast to the wake in geometry N1, which consists of only one pair of circulating flow regions, which was also observed for geometry H8. It is also evident that for N1 (Figure 6.5) the wake temperature is significantly lower than for the upstream flow, whereas for N3 (Figure 6.5) difference is not as large.

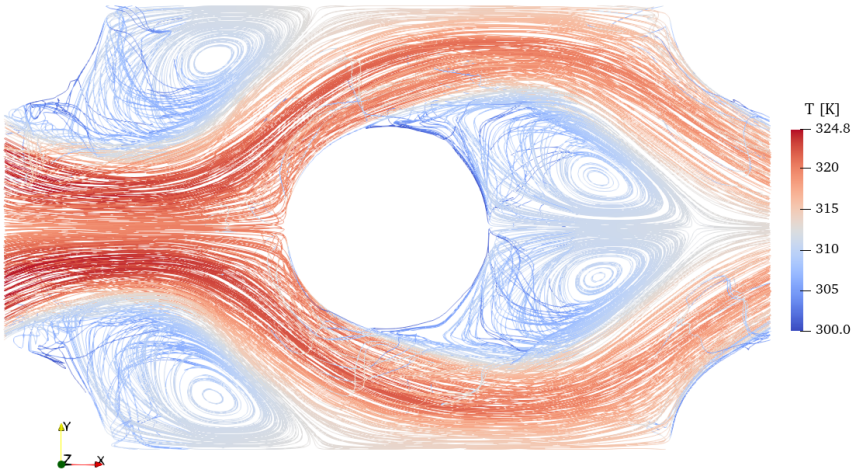


Figure 6.5 – Streamlines coloured by temperature at Re = 10 000 for geometry N1, from steady-state simulation with mapped inlet and outlet.

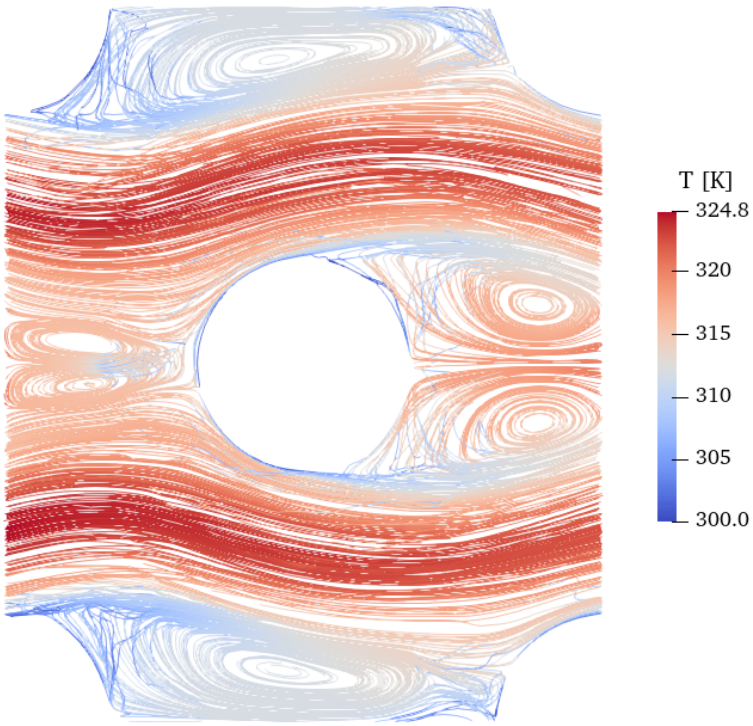


Figure 6.6 – Streamlines coloured by temperature at Re = 10 000 for geometry N3, from steady-state simulation with mapped inlet and outlet.

Figure 6.7 shows the velocity field in the plane directly upstream of the central tube of geometry H8, with the plane normal pointing in the transverse direction. At the fin base, the formation of a single pair counter-rotating horseshoe vortices is clearly visible.

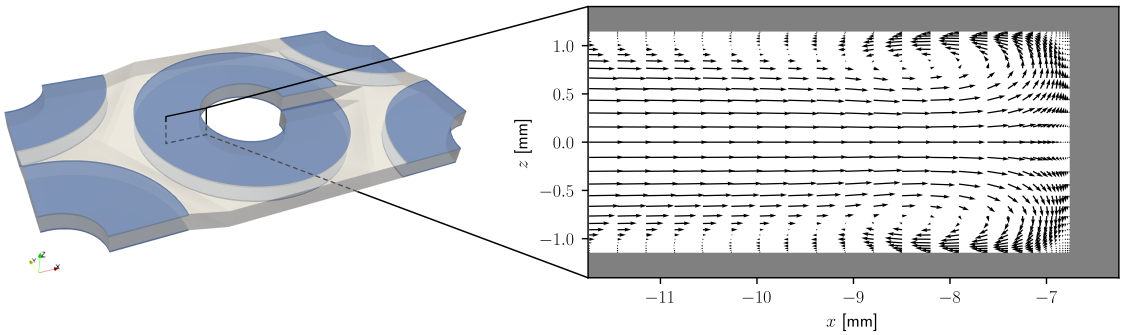


Figure 6.7 – Velocity field upstream of the central tube in case H8 at Re = 10 000, showing the formation of horseshoe vortices in the inter-fin region.

6.2 Steady-state simulations with momentum source term

The results from geometry H8 at Re = 5 000, using a momentum source term instead of mapping, is shown in Figure 6.8. After around 6 000 iterations, both Eu and $\mathrm{NuPr}^{-1/3}$ stabilize and convergence is reached. Eu and $\mathrm{NuPr}^{-1/3}$ are found to be 0.81 and 39.77, respectively, which is an 8.64% and 2.90% difference compared with the mapped inlet/outlet strategy (Figure 6.2). For higher Re, convergence was not achieved, displaying the same behaviour as geometry N1, which will be presented next.

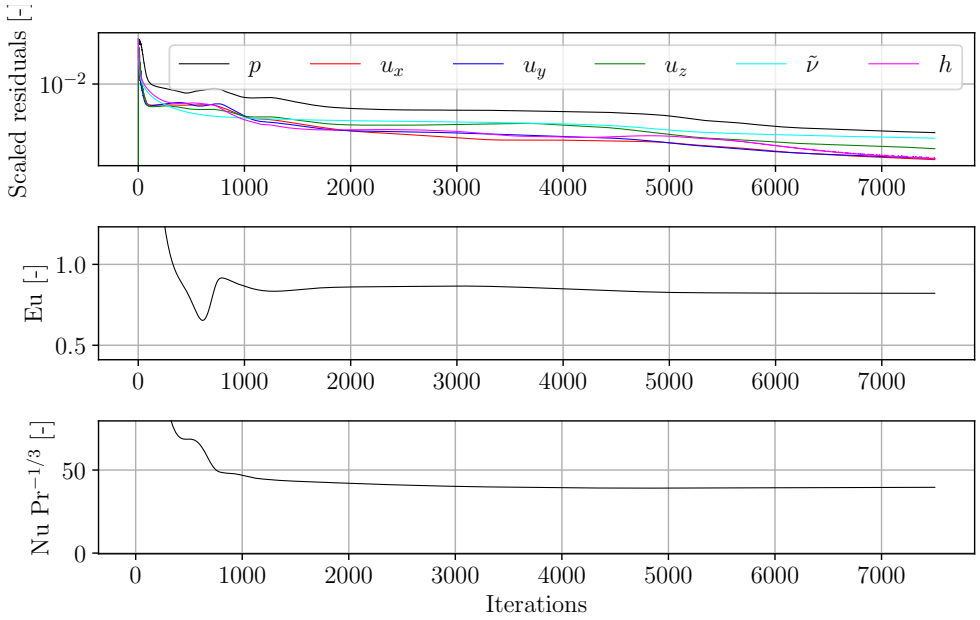


Figure 6.8 – Iteration evolution of scaled residuals, Eu and $\mathrm{NuPr}^{-1/3}$ for case H8 at Re = 5 000, using momentum source term. At the final iteration, Eu = 0.81 and $\mathrm{NuPr}^{-1/3}$ = 39.77.

The results from the case N1 at Re = 5 000 is shown in Figure 6.9. After a monotonically decreasing trend for both Eu and $\mathrm{NuPr}^{-1/3}$, fluctuations starts to occur after around 8500 iterations, accompanied by rising residuals. Though the residuals are displaying the same trend as in the mapped strategy (Figure 6.1), neither Eu nor $\mathrm{NuPr}^{-1/3}$ converges to fixed values, nor do they reach a steady oscillatory state. In other words, no converged solution is found, which was also the case for the other investigated Reynolds numbers.

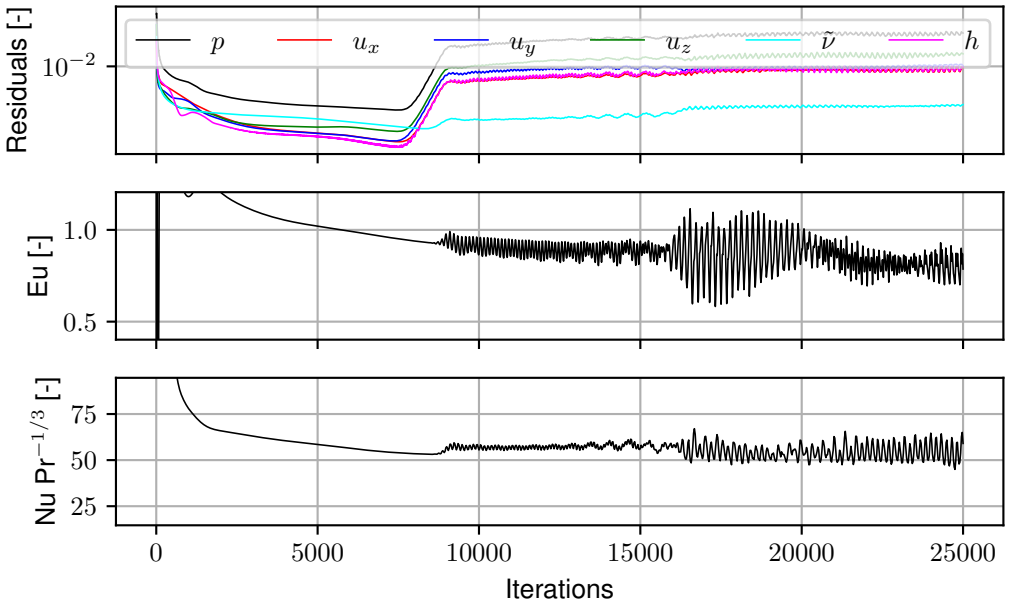


Figure 6.9 – Iteration evolution of scaled residuals for geometry N1 at Re = 5 000, using momentum source term in steady-state simulation.

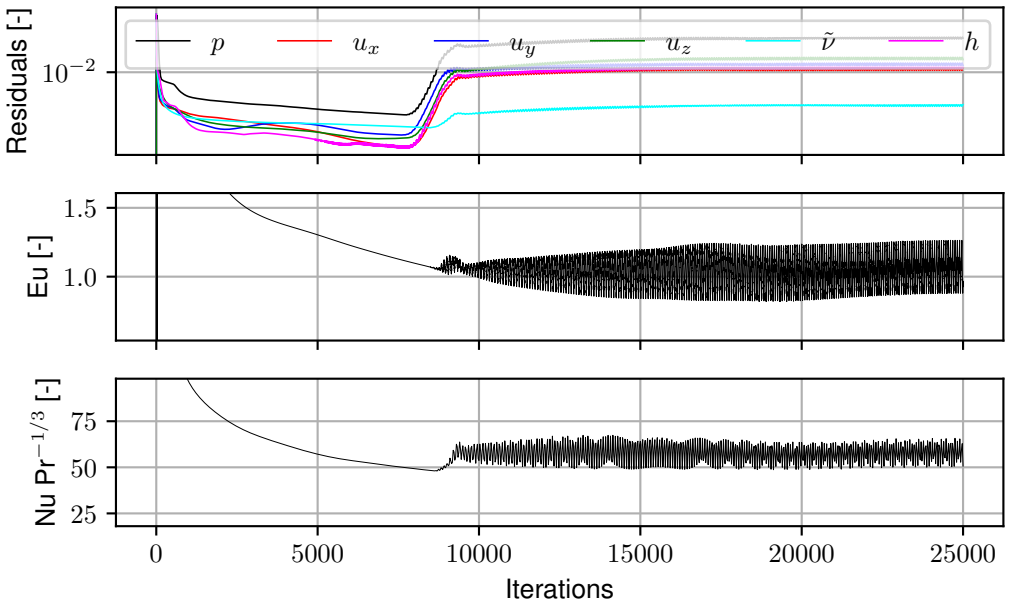


Figure 6.10 – Iteration evolution of scaled residuals for geometry N3 at Re = 5 000, using momentum source term in steady-state simulation.

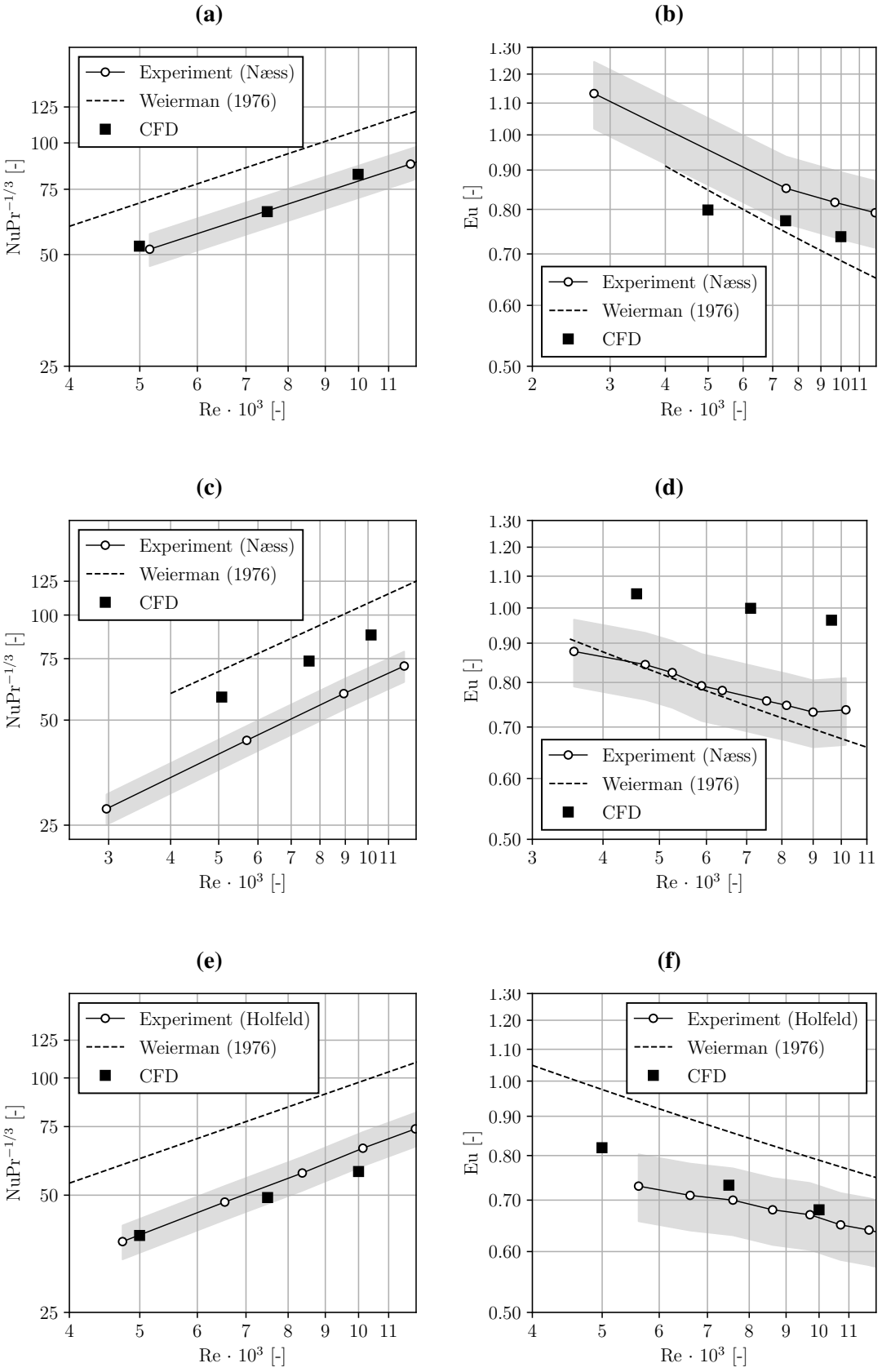


Figure 6.11 – Results from the steady source term strategy for $\mathrm{NuPr}^{-1/3}$ (a, c, e) and Eu (b, d, f) for geometries N1 (a, b), N3 (c, d) and H8 (e, f). The shaded areas are representing a $\pm 10\%$ range of the experimental data.

The results from geometry N3 at Re = 5 000 is shown in Figure 6.10. The same behaviour is as N1 is displayed also here, where from around iteration 8 000 the residuals are increasing. In contrast to N1, however, Eu and $\mathrm{NuPr}^{-1/3}$ are not fluctuating as randomly and seem to approach more stable oscillatory states.

A comparison between results from the steady source term strategy and experimental data is shown in Figure 6.11, for all three cases. As stated, only one of the simulations fully converged (H8 at Re = 5 000), and the values for Eu and $\mathrm{NuPr}^{-1/3}$ presented in Figure 6.11 are the averages sampled over the last 2 000 iterations. For case N1, $\mathrm{NuPr}^{-1/3}$ agrees well with experimental data, within 10% for all Reynolds numbers, as seen in Figure 6.11a. As seen in Figure 6.11b, Eu has an error of more than 15% for Re = 5 000, while for higher Re it is within 10% of experimental values. As with the mapped strategy, N3 is largely offset from the experimental data, with deviations as high as 31% and 43% for Eu (6.11d) and $\mathrm{NuPr}^{-1/3}$ (6.11c), respectively. However, Eu seems to follow the decaying trend more closely than in the mapped strategy (Figure 6.4d). $\mathrm{NuPr}^{-1/3}$ is also predicting more closely than the Weierman correlation, and also follows the same trend as the experimental data. The results from case H8 is within 10% for both Eu and $\mathrm{NuPr}^{-1/3}$ (though no experimental data for Eu is available at Re = 5 000), see Figures 6.11f and 6.11e, respectively. It is also evident that the predictions are more accurate than the Weierman in both cases.

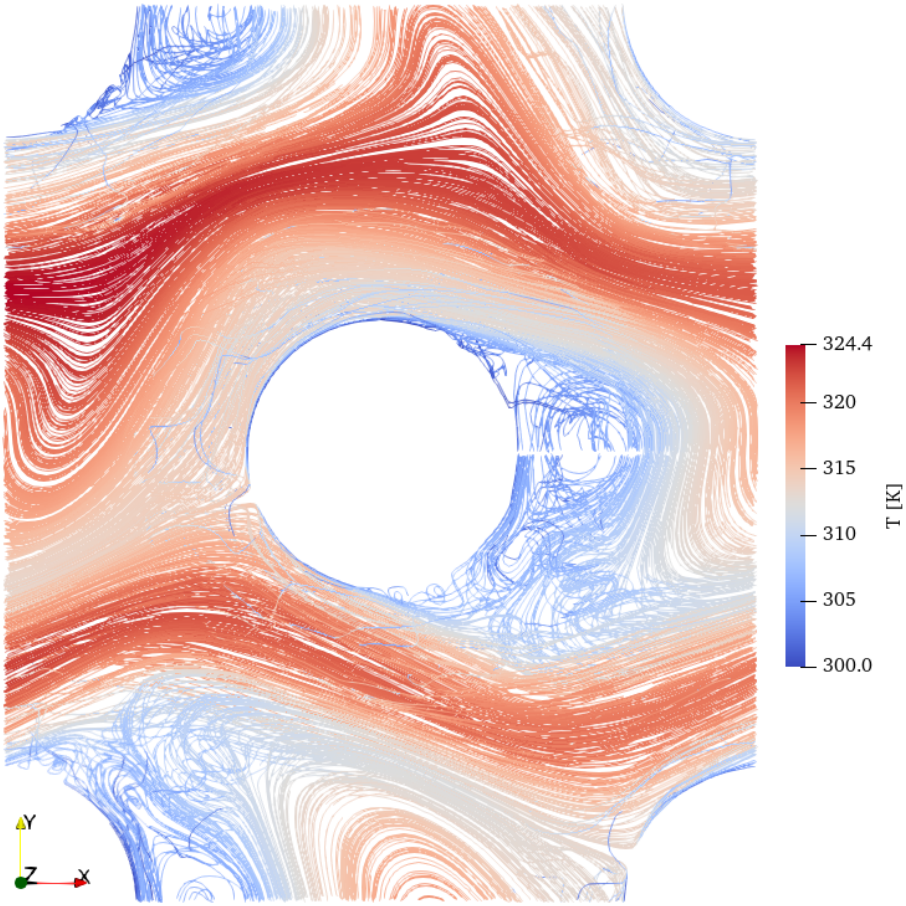


Figure 6.12 – Streamlines coloured by temperature at Re = 10 000 for geometry N3, from the final iteration of the steady-state simulation with momentum source term.

Figure 6.12 shows streamlines coloured by temperature for geometry N3 from the final iteration of the steady-state simulation with momentum source term at Re = 10 000. Unstable vortices occur in the wake of the tube, and the temperature in these regions are significantly lower than outside the wake. The wake is not split at the outlet boundary, in contrast to Figure 6.6.

6.3 Transient simulations with momentum source term

Due to limited time and computational resources, transient simulations was not be performed to the same extent as the steady-state simulations, and only geometries N1 and H8 were simulated.

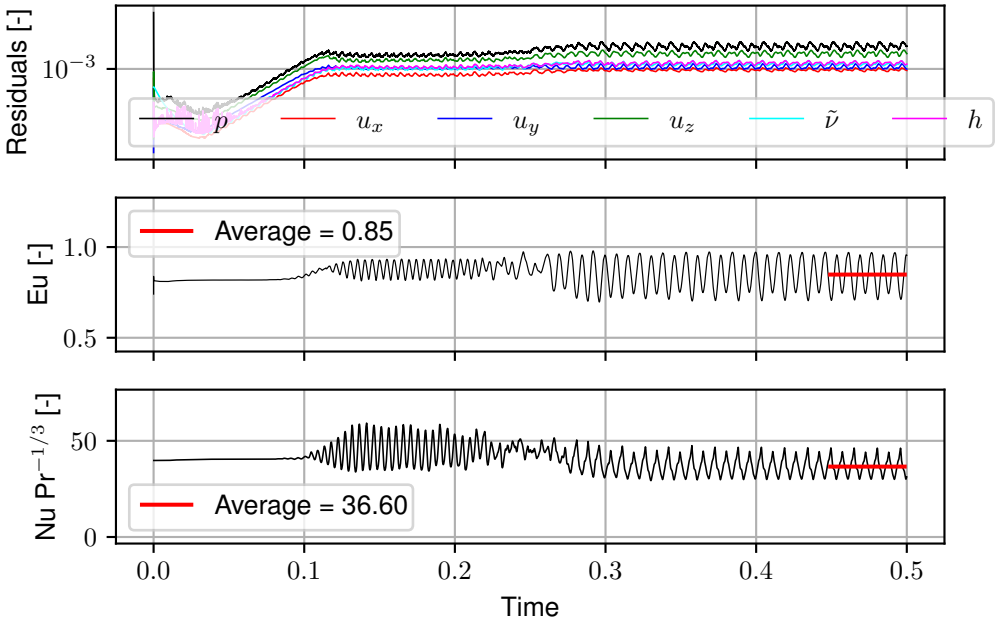


Figure 6.13 – Iteration evolution of scaled residuals, Eu and $NuPr^{-1/3}$ for case H8 at Re = 5 000, using the transient solution strategy. The average values (—) are, Eu = 0.85 and $NuPr^{-1/3}$ = 36.60.

The time development for geometry H8 at Re = 5 000 is shown Figure 6.13. After around 0.3 seconds (which corresponds to ca. 15 exchanges of fluid in the domain), the flow reaches a stable oscillatory pattern for both Eu and $NuPr^{-1/3}$, with average values of 0.73 and 44.50, respectively.

The time development for geometry N1 at Re = 5 000 is shown Figure 6.14. After around 0.12 seconds (which corresponds to ca. 10 exchanges of fluid in the domain), the flow seems to reach a stable oscillatory pattern for both Eu and $NuPr^{-1/3}$, with average values of 0.92 and 57.62, respectively. Note that this simulation was terminated due to time limitation on the computational cluster, and was intended to be around twice as long.

The results from the transient simulations are compared with experimental data and the Weierman correlation in Figure 6.15. For N1, both Eu (Figure 6.15b) and $NuPr^{-1/3}$ (Figure 6.15a) matches closely with experimental data, both being within 10%, and thus performs better than the Weierman correlation. The transient results from geometry H8 are underpredicting $NuPr^{-1/3}$, with the largest deviation being 20% at Re = 7 500. The

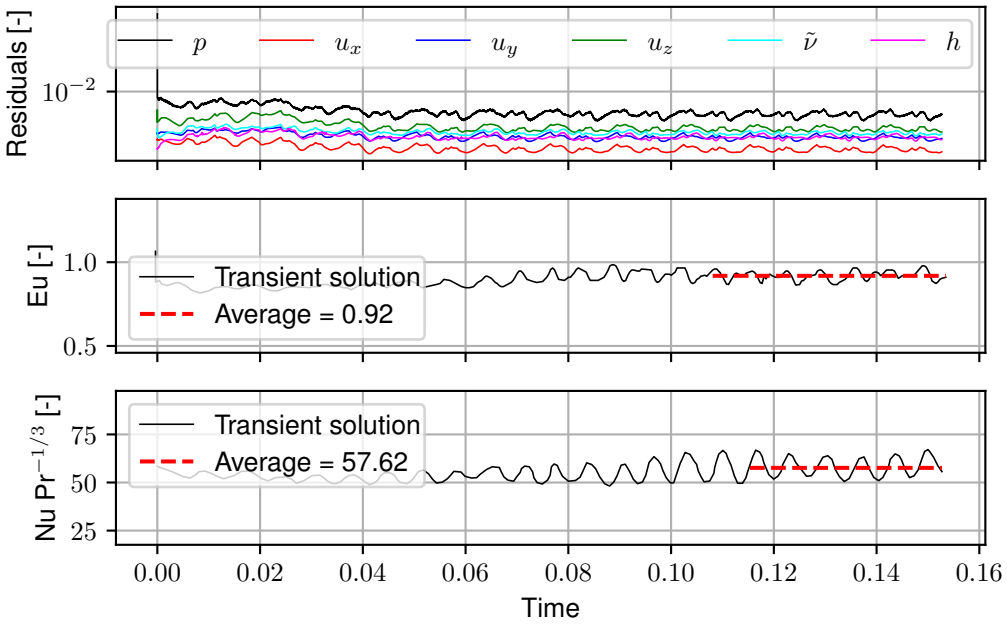


Figure 6.14 – Iteration evolution of scaled residuals, Eu and $NuPr^{-1/3}$ for case N1 at Re = 5 000, using the transient solution strategy. The average values (—) are, Eu = 0.92 and $NuPr^{-1/3}$ = 57.62.

prediction for Eu is very close to the experimental value, whereas it seems to slightly overpredict for Re = 5 000, though no experimental data points are available for Re $< 5\,600$.

Figure 6.16 shows the flow field from the transient simulation of geometry N1 from the final time step, at Re = 5 000. The streamlines are coloured by temperature and show the shedding of vortices behind the tubes, where both vortices are located at the top part of the tube. The which are at lower temperatures compared with upstream flow (as seen also in Figure 6.5), but is not only restricted to directly downstream of tube, but is also dispersed around the tube.

6.4 Fin efficiencies

A comparison of the theoretical (Equations (2.5) and (2.6) for H8 and N1, respectively) and corrected (Equation 2.7) fin efficiencies are presented in Figure 6.17, where they are given as the deviation in percent from the fin efficiency calculated using CFD data directly, i.e. Equation (4.7). For H8 (Figure 6.17a), the theoretical fin efficiency deviates with less than 0.5% for all Re, whereas the maximum deviation for the corrected fin efficiency is 5%. For N3 (Figure 6.17b), the theoretical fin efficiency deviates with less than 2%, while the largest deviation for the corrected fin efficiency is 3.7%.

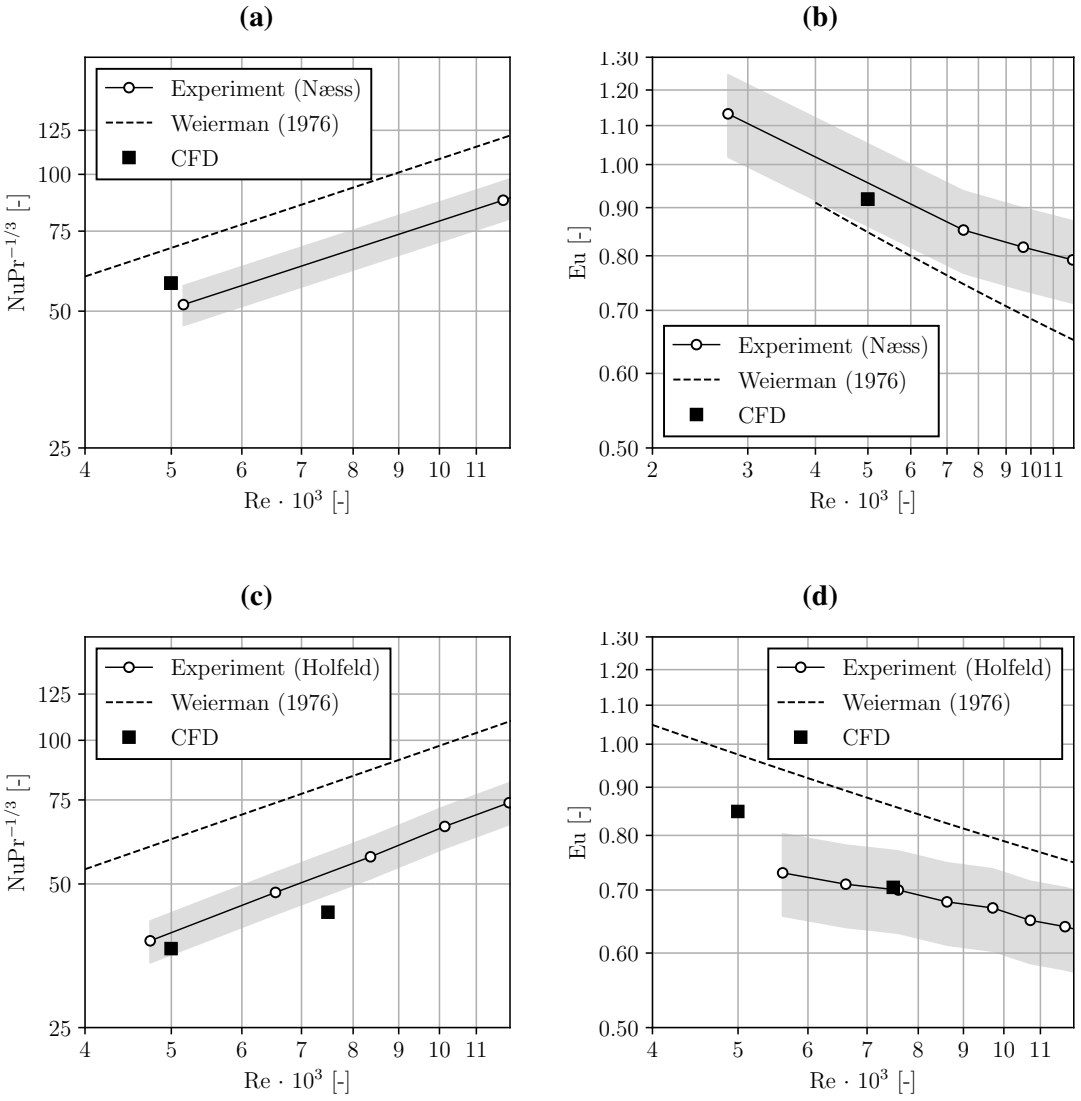


Figure 6.15 – Results from the transient strategy for $\mathrm{NuPr}^{-1/3}$ (a, c) and Eu (b, d) for geometries N1 (a, b) and H8 (c, d). The shaded areas are representing a $\pm 10\%$ range of the experimental data.

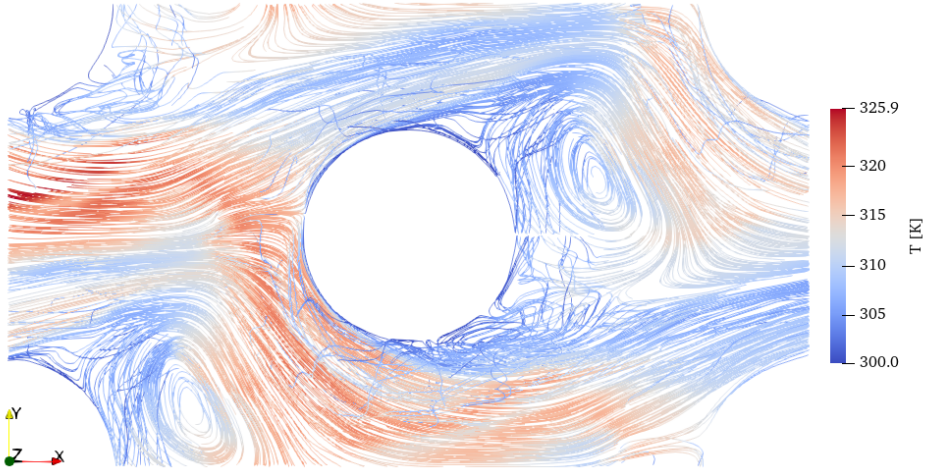


Figure 6.16 – Streamlines coloured by temperature at Re = 10 000 for geometry N1, from transient simulation with momentum source term.

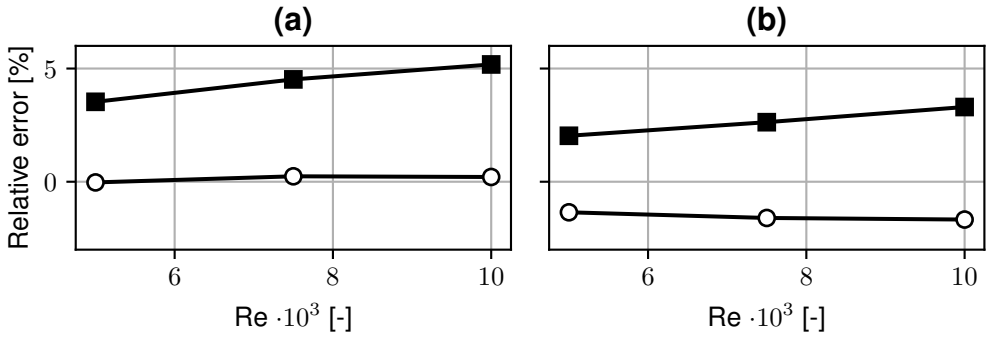


Figure 6.17 – Percentage deviation of theoretical (◯) and corrected (■) fin efficiencies from the fin efficiencies based on the CFD computations for geometry **(a)** H8 and **(b)** N1, based on results from the steady-state simulation with mapped inlet and outlet (strategy 1).

7 DISCUSSION

This section will include a discussion of the results that were presented in Section 6, and will be structured correspondingly, i.e. each solution strategy will be discussed independently, before being compared with each other.

7.1 Steady-state simulations with mapped inlet and outlet

The fact that the residuals from geometry N1 (Figure 6.1) are not monotonically decreasing, but rather increasing to high levels, indicates that there are instabilities in the flow field, which in turn makes it challenging for the linear solvers to find a steady solution. Instabilities such as these may arise from numerous sources, such as unstable numerical schemes or poor mesh quality, e.g. too much non-orthogonality and/or skewness, as discussed in Section 3.2. Both coarse and fine grids were tested, as well as more stable numerical schemes (such as standard upwind) and gradient limiters, but the same unstable behaviour was observed nevertheless. This indicates that the instabilities arise due to transient behaviour of the flow field, which the steady SIMPLE-algorithm is incapable of finding a steady solution to.

For geometry H8 on the other hand (Figure 6.2), it is clear that a stable flow field is indeed found, as evident from the monotonically decreasing residuals and the convergence of both Eu and $\mathrm{NuPr}^{-1/3}$. This suggests that instabilities are less prominent in solid fins compared with serrated fins, which is expected due to the high degree of mixing in flow over serrated fins. Another factor that may contribute to more stable behaviour for H8 is the fin pitch s_f, which is almost twice as much for N1 compared with H8. Thus, it could be possible to also find stable solutions for serrated fins if s_f were sufficiently small to dampen the instabilities.

The iteration history for N3 (Figure 6.3) indicates that the flow field is stabilized and a converged solution is found, and seemingly follows the same behaviour as H8. However, when comparing the streamlines of N3 to N1, Figures 6.6 and 6.5, respectively, there is a clear difference in the structure of the wake at the outlet. For N1, it is evident that the wake does not extend beyond the outlet, and therefore the velocity flows out of the domain at the entire outlet boundary. For N3 on the other hand, the layout angle is so large that the wake can be expected to extend further than the outlet, which implies that there will be backflow into the domain at some regions in the outlet boundary. This results in unphysical behaviour of the simulation, as the inlet-outlet boundary condition sets $u_{\text{outlet}} = 0$ if backflow occurs. The consequence is that the wake is limited by the outlet boundary, and a second recirculating zone forms after the inlet, due to the mapping. This is not the expected behaviour, and can be compared with the wake at the top and bottom of the domain, where the recirculating zone extends all the way between the two tube rows. The mapping is also the cause of the relatively high temperatures in the wake of geometry N3 (Figure 6.6), as there is no information of the temperature coming in from backflow, and a fixed temperature must be prescribed in these cases. It is therefore evident the mapping strategy is not able to represent the flow or temperature field for large layout angles β, and which is likely to be the reason for the behaviour observed in Figure 6.4d, where Eu in particular shows large deviations from the experimental values and does not follow the same trends.

Another reason for the large deviations from experimental data for N3, may be attributed to uncertainties in the data reduction procedure by Næss [46] for this geometry. Inconsistency

has been found when F_{min} was reported for N3, which propagates to the calculation of both Re and Eu. However, this does not affect the fact that the mapped strategy is not appropriate for geometries with high β.

The horseshoe vortices observed in Figure 6.7 are expected from observations made in previous studies (Section 2.3). No more than one pair of horsehoe vortices were observed in this work, which can be attributed to insufficiently large Reynolds numbers. Another explanation could be that the model does not resolve the smallest vortices due to the turbulence modelling chosen, and could have been captured by using LES, as was done by Salinas-Vázquez et al. [32]. For accessing only overall heat transfer and pressure drop however, an overly highly resolved flow is not necessary as long as the mean effects are captured, though it would be of interest to investigate this further to study the mechanics of flow mixing closer.

The results indicate that if the layout angle is fixed at small enough values, the mapped inlet and outlet strategy could be used to get tolerable steady-state predictions for both heat transfer and pressure drop without too much computational effort, as shown for N1 and H8. Other layout angles should be investigated to establish the limits of this strategy, but for general cases it should not be used, due to its disability to model flows with large layout angles.

7.2 Steady-state simulations with momentum source term

The steady-state simulations with momentum source term were tested to avoid the unphysical behaviour at the wake, as this ensures that the inlet and outlet are actual cyclic boundaries, which can handle backflow at the outlets. For Re = 5 000, H8 (Figure 6.8) reached full convergence after approximately 6000 iterations, whereas around 12 000 iterations were needed when using the mapped strategy. This speed-up is expected, since the mapped strategy can be thought of as a series of standard inlet/outlet simulations, while the strategy with a momentum source term is solved in one go. This is a significant advantage when simulation time is of importance, which is a key factor of the second objective of this work (Section 1.2).

The weaknesses of this method is, however, apparent when looking at the iteration history of geometry N1 (Figure 6.9) and N3 (6.10). Behaviour similar to that of H8 is observed until around iteration 8000, but after this, residuals are increasing and both Eu and $\mathrm{NuPr}^{-1/3}$ start to fluctuate. In contrast to the mapped strategy, no convergence for neither Eu nor $\mathrm{NuPr}^{-1/3}$ is achieved, and no stable oscillatory solution is found for N1. This unstable behaviour is likely due to same reasons as discussed in Section 7.1, but the violent fluctuations that are observed in Figure 6.9 is probably due to the update of the momentum source term, which is performed each iteration. When the mapped strategy was used, the inlet was fixed between iterations (except for the period updates), which contributes to stabilizing the flow. Similar behaviour is seen for geometry N3 in Figure 6.10, though a more regular oscillatory pattern is displayed for both Eu and $\mathrm{NuPr}^{-1/3}$, compared with N1. Though several attempts were made to stabilize the method with both lower order schemes and several grid refinements, this was not achieved, and it should therefore be concluded that this method can not be used to get reliable estimates for pressure drop and heat transfer.

The temperature-coloured streamlines (Figure 6.12), shows a fluctuating flow field with vortex shedding in the wake of the tubes. It is worth noting the outlet boundary is no

longer a barrier for the wake, as the flow is now actually periodic across the inlet and outlet boundaries.

7.3 Transient simulations with momentum source term

The transient simulations were run in order to resolve the instabilities that were observed in two preceding steady-state strategies, and is in principle an extension of the second strategy. The limited number of simulations that were performed does not give the same basis of comparison as the steady-state simulations, but still yield both quantitative and qualitative results that can be used to access its performance.

For H8 (Figure 6.13) the time evolution shows that a stable oscillatory state is reached after approximately 15 fluid domain exchanges, and matches experimental results closely at both Re investigated. N1 (Figure 6.13) was not simulated until the end time, but the same stable oscillatory state seemed to be reached for this geometry as well, which indicates that the instabilities discussed in the two previous sections are at least partially caused by transient flow phenomena, such as vortex shedding (Figure 6.16). The residual behaviour for H8, which are still relatively high, may indicate that some numerical instabilities still are present. Due to the limited time frame, different setups could not be tested, but some possible explanations are discussed next.

The transient simulations require significantly longer computational time than the steady state, which is an immediate drawback of this method. A way to alleviate this problem could be to employ the PIMPLE-algorithm in OpenFOAM, which can be thought of as a combination of the PISO- and SIMPLE-algorithms (using both inner and outer corrector loops). The PIMPLE-algorithm can be used with larger time steps than the PISO-algorithm and still maintain stability and accuracy, which can reduce computational time significantly. The reason for why this was not used in this work, is that the PIMPLE-algorithm is not fully implemented for CHT simulations in the OpenFOAM version (v2106) used here, and porting the case to a version were this is implemented (e.g. OpenFOAM 9) was not achieved, as the transient simulations were performed towards the end of the time frame for this work.

The original setup of the simulation was not intended for transient simulation, but was adapted to use with a transient solver after the instabilities of the steady-state strategies became apparent. One concern with the unit cell in transient flow conditions is that the extent of the domain may be too small to resolve larger flow features that arise when the flow becomes transient, and that it should be extended in both stream wise and transverse directions to allow for these features to be resolved. Another concern is that the vortices observed in Figure 6.16 are being forced to be periodic, when in fact they may have been expected to be alternating between tube rows, as was reported by Salinas-Vázquez et al. [32]. This could be investigated further by adding another tube row to the domain, but an alternative cyclic boundary condition could also be employed. The proposed boundary condition would avoid doubling the domain size by instead rotating the inlet and outlet boundaries 180° to account for the alternating vortex shedding, as illustrated in Figure 7.1. If no asymmetry is present, the boundary condition would not affect the flow, and could therefore also be used in cases were vortex-shedding does not occur.

7.4 Comparison of the strategies

The results from the different strategies for geometries N1 and H8 are summarized in Table 7.1 at Re = 5 000. N3 was not included due to the uncertainties in the experimental values

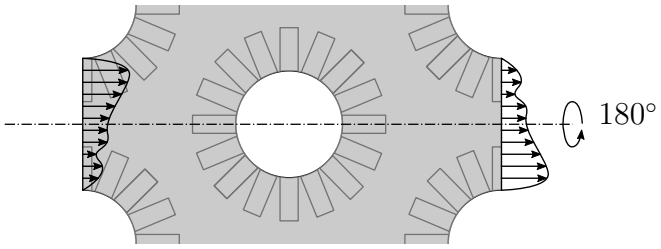


Figure 7.1 – Proposed rotated cyclic boundary condition.

and as no transient simulation was performed using this geometry.

Table 7.1 – Summary of results from the different strategies for geometries N1 and H8 at Re = 5 000, presented as percentage error from the experimental results. Non-converged solutions are marked ×.

Geometry	Strategy 1		Strategy 2		Strategy 3	
	Eu	$NuPr^{-1/3}$	Eu	$NuPr^{-1/3}$	Eu	$NuPr^{-1/3}$
N1	11.5	15.2	×	×	2.3	15.2
H8	15.3	5.1	6.6	2.0	10.5	6.2

Though all strategies were able to yield predictions for N1 and H8 within 20% of the experimental results, there is some variation between Eu and $NuPr^{-1/3}$ within each strategy. There is no clear preference, as both the strategy 1 and 3 reports max. errors of the same size, though strategy 3 is generally closer to the experiments. Strategy 2 performs well for H8 at Re = 5 000, being within 10% for both Eu and $NuPr^{-1/3}$, but the severe convergence issues that are displayed for other geometries and Re means that this method should not be used in general. The numerical results were able to predict better than the Weierman correlation in most all cases, which is promising for the use of the numerical model for validating compact HRSG design.

It concluded that the transient solution strategy is the most suitable method of modelling flow and heat transfer in finned tube banks, as this is able to handle transient instabilities which. This method will also be able to handle large layout angles, as this uses periodic boundary condition, which was shown to be able to represent periodic flow also for these cases (Figure 6.12). A natural extension of this work would be to investigate the cyclic boundary conditions, as suggested in Section 7.3. In addition to perform grid convergence studies for all geometries, a temporal convergence study should be conducted for the transient simulations. Another advantage is that the transient results could be used directly to conduct vibrational analyses, as was done by Lindqvist % Næss [31], though this has not been the objective of the present work.

7.5 Fin efficiencies

The performance of the theoretical and corrected fin efficiencies (Figure 6.17) shows that for Reynolds number in the range investigated in this work, the deviations are minimal, being 5% or less for all cases. It is however interesting to note that the corrected fin efficiencies perform *poorer* than theoretical fin efficiencies. This suggests that the Weierman correction should be used with care. The same conclusion was also drawn by Lindqvist & Næss [30], who had the same finding for some of their investigated geometries.

8 CONCLUSIONS

A novel numerical modelling framework has been developed. The model can predict heat transfer and pressure drop in compact heat recovery steam generators by using computational fluid dynamics (CFD). A parameterized grid generation procedure was developed, which is able to generate numerical grids to be used in CFD simulations for both solid and serrated fins. The procedure allows for the generation of a wide range of geometries in an automated manner, and in this work three different fin-tube geometries were created for further CFD-simulations. The grid constitutes a periodic domain where flow around a single tube row was modelled. The CFD simulation was set up using the open-source software OpenFOAM, and three main solution strategies were implemented and tested: steady state simulation with mapping between inlet and outlet, steady state with momentum source term and, finally, a transient simulation with momentum source term. The converged numerical results were validated with available experimental data for all three geometries, with most results agreeing within 20% of experiments values. The results were also compared with a selected empirical correlation, and out-performed the correlation in most cases.

The first strategy (steady with mapped inlet/outlet) gave stability issues for the serrated geometries, but convergence was achieved for both pressure drop and heat transfer. For large layout angles, however, this strategy was not able to predict the wake behind the finned tube, and should therefore only be used for sufficiently small layout angles.

The second strategy (steady with momentum source term) was found to be more unstable than the first strategy, also for solid fins (except for low Re). In the cases where convergence was reached, this strategy required fewer iterations than the first strategy, but in most cases it was observed to not converge at all.

To resolve the flow instabilities, transient simulations were performed as a third strategy. Due to the computational demand of the transient simulations, only a few cases were run, and only two of the geometries were simulated. The transient simulations were able to resolve the instabilities that were observed in the steady-state simulations, and showed good agreement with experimental data. However, there are concerns that the cyclic stream wise boundary condition might not be applicable when the flow field is transient, and further work is needed in this area. The results do however indicate that transient simulations are the most robust way of modelling flow in finned tube bundles, though they are computationally more demanding than the steady-state simulations.

REFERENCES

[1] Klima- og miljørapport, Tech. rep., Norsk Olje og Gass (2021).
URL https://klimamiljorapport.norskoljeoggass.no/NOROG%20Klima%20og%20Miljorapport%202021.pdf

[2] L. Nord, O. Bolland, Steam bottoming cycles offshore – Challenges and possibilities, Journal of Power Technologies 92 (2012) 201–207.

[3] R. M. Montañés, G. Skaugen, B. Hagen, D. Rohde, Compact Steam Bottoming Cycles: Minimum Weight Design Optimization and Transient Response of Once-Through Steam Generators, Frontiers in Energy Research 9 (2021) 687248. doi:10.3389/fenrg.2021.687248.

[4] M. J. Mazzetti, B. A. L. Hagen, G. Skaugen, K. Lindqvist, S. Lundberg, O. A. Kristensen, Achieving 50% weight reduction of offshore steam bottoming cycles, Energy 230 (2021) 120634. doi:10.1016/j.energy.2021.120634.
URL https://www.sciencedirect.com/science/article/pii/S0360544221008835

[5] glorytube, Finned tube manufacturer from China- Glorytubetech.
URL https://www.glorytubetech.com/products/finned-tube/

[6] A. Holfeld, Experimental investigation of heat transfer and pressure drop in compact waste heat recovery units, Doctoral theses, Norwegian University of Science and Technology, Trondheim (Jun. 2016).
URL https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2432512

[7] R. Webb, Principles of Enhanced Heat Transfer, 2nd Edition, Taylor & Francis, 2005.
URL https://books.google.no/books?id=Yn1gQgAACAAJ&dq=Principles+of+Enhanced+Heat+Transfer&hl=no&sa=X&redir_esc=y

[8] K. Lindqvist, Computational Fluid Dynamics Modeling of Flow and Heat Transfer in Fin-Tube Bundles, Doctoral theses, Norwegian University of Science and Technology, Trondheim (Jan. 2019).
URL https://ntnuopen.ntnu.no/ntnu-xmlui/bitstream/handle/11250/2583704/Karl%20Lindqvist_fulltext.pdf?sequence=5&isAllowed=y

[9] W. Krückels, V. Kottke, Untersuchung über die Verteilung des Wärmeübergangs an Rippen und Rippenrohr-Modellen, Chemie Ingenieur Technik 42 (6) (1970) 355–362, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cite.330420607. doi:10.1002/cite.330420607.
URL https://onlinelibrary.wiley.com/doi/abs/10.1002/cite.330420607

[10] X. Hu, A. M. Jacobi, Local Heat Transfer Behavior and Its Impact on a Single-Row, Annularly Finned Tube Heat Exchanger, Journal of Heat Transfer 115 (1) (1993) 66–74. doi:10.1115/1.2910671.
URL https://doi.org/10.1115/1.2910671

[11] J. V. Simo Tala, S. Russeil, D. Bougeard, J. L. Harion, Investigation of the flow characteristics in a multirow finned-tube heat exchanger model by means of PIV measurements, Experimental Thermal and Fluid Science 50 (2013) 45–53. doi:10.1016/j.expthermflusci.2013.05.003.
URL https://www.sciencedirect.com/science/article/pii/S0894177713001131

[12] R. Hofmann, H. Walter, Experimental and Numerical Investigation of the Gas Side Heat Transfer and Pressure Drop of Finned Tubes—Part II: Numerical Analysis, Journal of Thermal Science and Engineering Applications 4 (4) (2012) 041008. doi:10.1115/1.4007125.
URL https://asmedigitalcollection.asme.org/thermalscienceapplication/article/doi/10.1115/1.4007125/378390/Experimental-and-Numerical-Investigation-of-the

[13] C. J. Baker, The laminar horseshoe vortex, Journal of Fluid Mechanics 95 (2) (1979) 347–367, publisher: Cambridge University Press. doi:10.1017/S0022112079001506.
URL https://www.cambridge.org/core/journals/journal-of-fluid-mechanics/article/laminar-horseshoe-vortex/DEE27F181D2A82AC96642213E6C9D2A2

[14] C. Lin, T. C. Ho, S. Dey, Characteristics of Steady Horseshoe Vortex System near Junction of Square Cylinder and Base Plate, Journal of Engineering Mechanics 134 (2) (2008) 184–197, publisher: American Society of Civil Engineers. doi:10.1061/(ASCE)0733-9399(2008)134:2(184).
URL https://ascelibrary.org/doi/full/10.1061/%28ASCE%290733-9399%282008%29134%3A2%28184%29

[15] F. Incropera, D. DeWitt, T. Bergman, A. Lavine, Principles of Heat and Mass Transfer, John Wiley & Sons Singapore Pte. Limited, 2013.
URL https://books.google.no/books?id=UTJ0CgAAQBAJ

[16] D. Harper, W. Brown, Mathematical Equations for Heat Conduction in the Fins of Air-Cooled Engines, Tech. Rep. 158, USE (1922).

[17] C. Weierman, Correlations Ease the Selection of Finned Tubes, Oil and Gas Journal (1976) 94–100.

[18] K. Hashizume, R. Morikawa, T. Koyama, T. Matsue, Fin Efficiency of Serrated Fins, Heat Transfer Engineering 23 (2) (2002) 6–14, publisher: Taylor & Francis _eprint: https://doi.org/10.1080/01457630252800386. doi:10.1080/01457630252800386.
URL https://doi.org/10.1080/01457630252800386

[19] K. Kawaguchi, K. Okui, T. Asai, Y. Hasegawa, The heat transfer and pressure drop characteristics of finned tube banks in forced convection (effects of fin height on pressure drop characteristics), Heat Transfer—Asian Research 35 (3) (2006) 179–193, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/htj.20112. doi:10.1002/htj.20112.
URL https://onlinelibrary.wiley.com/doi/abs/10.1002/htj.20112

[20] J.-Y. Jang, J.-T. Lai, L.-C. Liu, The thermal-hydraulic characteristics of staggered circular finned-tube heat exchangers under dry and dehumidifying conditions, International Journal of Heat and Mass Transfer 41 (21) (1998) 3321–3337. doi:10.1016/S0017-9310(98)00037-4.
URL https://www.sciencedirect.com/science/article/pii/S0017931098000374

[21] M. S. Mon, Numerical Investigation of Air-Side Heat Transfer and Pressure Drop in Circular Finned-Tube Heat Exchangers, Ph.D. thesis, Technischen Universität Bergakademie Freiberg, Freiberg, Germany (Feb. 2003).

[22] M. S. Mon, U. Gross, Numerical study of fin-spacing effects in annular-finned tube heat exchangers, International Journal of Heat and Mass Transfer 47 (8) (2004) 1953–1964. doi:10.1016/j.ijheatmasstransfer.2003.09.034.
URL https://www.sciencedirect.com/science/article/pii/S0017931003005829

[23] M. Torresi, A. Saponaro, S. Camporeale, B. Fortunato, CFD Analysis of the Flow Through Tube Banks of HRSG, Berlin, Germany, 2008, pp. 1–11. doi:10.1115/GT2008-51300.

[24] S. R. McIlwain, A Comparison of Heat Transfer Around a Single Serrated Finned Tube and a Plain Finned Tube, International Journal of Research and Reviews in Applied Sciences 2 (2) (2010) 88–94.

[25] S. R. Mcilwain, A CFD Comparison of Heat Transfer and Pressure Drop Across Inline Arrangement Serrated Finned Tube Heat Exchangers With an Increasing Number of Rows, International Journal of Research and Reviews in Applied Sciences 4 (2) (2010) 162–169.

[26] A. Lemouedda, A. Schmid, E. Franz, M. Breuer, A. Delgado, Numerical investigations for the optimization of serrated finned-tube heat exchangers, Applied Thermal Engineering 31 (8) (2011) 1393–1401. doi:10.1016/j.applthermaleng.2010.12.035.
URL https://www.sciencedirect.com/science/article/pii/S1359431111000044

[27] C. T. Ó Cléirigh, W. Smith, Can CFD accurately predict the heat-transfer and pressure-drop performance of finned-tube bundles?Publisher: Elsevier (Dec. 2014). doi:10.1016/j.applthermaleng.2014.08.019.

[28] E. Martinez, W. Vicente, M. Salinas-Vazquez, I. Carvajal, M. Alvarez, Numerical simulation of turbulent air flow on a single isolated finned tube module with periodic boundary conditions, International Journal of Thermal Sciences 92 (2015) 58–71. doi:10.1016/j.ijthermalsci.2015.01.024.
URL https://www.sciencedirect.com/science/article/pii/S1290072915000277

[29] E. Martinez-Espinosa, W. Vicente, M. Salinas-Vazquez, I. Carvajal-Mariscal, Numerical Analysis of Turbulent Flow in a Small Helically Segmented Finned Tube Bank, Heat Transfer Engineering 38 (1) (2017) 47–62, publisher: Taylor & Francis _eprint: https://doi.org/10.1080/01457632.2016.1156396. doi:10.1080/01457632.2016.1156396.
URL https://doi.org/10.1080/01457632.2016.1156396

[30] K. Lindqvist, E. Næss, A validated CFD model of plain and serrated fin-tube bundles, Applied Thermal Engineering 143 (2018) 72–79. doi:10.1016/j.applthermaleng.2018.07.060.
URL https://www.sciencedirect.com/science/article/pii/S1359431117359598

[31] K. Lindqvist, E. Naess, NUMERICAL MODELING OF VORTEX SHEDDING IN HELICALLY WOUND FINNED TUBE BUNDLES IN CROSS FLOW, in: International Heat Transfer Conference 16, Begellhouse, Beijing, China, 2018, pp. 1843–1850. doi:10.1615/IHTC16.cms.024036.
URL http://ihtcdigitallibrary.com/conferences/ihtc16,1b62c5720219fa67,4ffa35c077d9305a.html

[32] M. Salinas-Vázquez, J. Ramírez-Cruz, W. Vicente, E. Martínez-Espinosa, H. M. Aviña-Jiménezez, C. Lagarza-Cortés, Large eddy simulation of fully-developed flow in a helical segmented-fin tube bundle, Applied Mathematical Modelling 98 (2021) 595–610. doi:10.1016/j.apm.2021.05.027.
URL https://www.sciencedirect.com/science/article/pii/S0307904X21002808

[33] H. G. Weller, G. Tabor, H. Jasak, C. Fureby, A tensorial approach to computational continuum mechanics using object-oriented techniques, Computers in Physics 12 (6) (1998) 620–631, publisher: American Institute of Physics. doi:10.1063/1.168744.
URL https://aip.scitation.org/doi/10.1063/1.168744

[34] M. Själander, M. Jahre, G. Tufte, N. Reissmann, EPIC: An Energy-Efficient, High-Performance GPGPU Computing Research Infrastructure, arXiv:1912.05848 [cs]ArXiv: 1912.05848 (Sep. 2021).

[35] J. Ferziger, M. Peric, Computational Methods for Fluid Dynamics, Springer Berlin Heidelberg, 2012.
URL https://books.google.no/books?id=ye3tCAAAQBAJ

[36] H. Versteeg, W. Malalasekera, An Introduction to Computational Fluid Dynamics: The Finite Volume Method, New York, 1995.
URL https://books.google.no/books?id=uZBpQgAACAAJ

[37] F. Moukalled, L. Mangani, M. Darwish, The Finite Volume Method in Computational Fluid Dynamics: An Advanced Introduction with OpenFOAM® and Matlab, Fluid Mechanics and Its Applications, Springer International Publishing, 2015.
URL https://books.google.no/books?id=GYRgCgAAQBAJ

[38] S. V. Patankar, D. B. Spalding, A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows, International Journal of Heat and Mass Transfer 15 (10) (1972) 1787–1806. doi:10.1016/0017-9310(72)90054-3.
URL https://www.sciencedirect.com/science/article/pii/0017931072900543

[39] R. I. Issa, A. D. Gosman, A. P. Watkins, The computation of compressible and incompressible recirculating flows by a non-iterative implicit scheme, Journal of Computational Physics 62 (1) (1986) 66–82. doi:10.1016/0021-9991(86)90100-2.
URL https://www.sciencedirect.com/science/article/pii/0021999186901002

[40] G. E. P. Box, Science and Statistics, Journal of the American Statistical Association 71 (356) (1976) 791–799, publisher: Taylor & Francis _eprint: https://www.tandfonline.com/doi/pdf/10.1080/01621459.1976.10480949. doi:10.1080/01621459.1976.10480949.
URL https://www.tandfonline.com/doi/abs/10.1080/01621459.1976.10480949

[41] S. Pope, P. Eccles, S. Pope, Cambridge University Press, Turbulent Flows, Cambridge University Press, 2000.
URL https://books.google.no/books?id=HZsTw9SMx-0C

[42] P. Spalart, S. Allmaras, A one-equation turbulence model for aerodynamic flows, in: 30th Aerospace Sciences Meeting and Exhibit, American Institute of Aeronautics

and Astronautics, Reno,NV,U.S.A., 1992. doi:10.2514/6.1992-439.
URL https://arc.aiaa.org/doi/10.2514/6.1992-439

[43] D. B. Spalding, A Single Formula for the "Law of the Wall", Journal of Applied Mechanics 28 (3) (1961) 455–458. doi:10.1115/1.3641728.
URL https://doi.org/10.1115/1.3641728

[44] J. Kim, P. Moin, R. Moser, Turbulence statistics in fully developed channel flow at low Reynolds number, Journal of Fluid Mechanics 177 (1987) 133–166, publisher: Cambridge University Press. doi:10.1017/S0022112087000892.
URL https://www.cambridge.org/core/journals/journal-of-fluid-mechanics/article/turbulence-statistics-in-fully-developed-channel-flow-at-low-reynolds-number/308DCF387F4488D6A0FB189D8206DF7B#article

[45] P. J. Roache, K. N. Ghia, F. M. White, Editorial Policy Statement on the Control of Numerical Accuracy, Journal of Fluids Engineering 108 (1) (1986) 2. doi:10.1115/1.3242537.
URL https://doi.org/10.1115/1.3242537

[46] E. Næss, Experimental investigation of heat transfer and pressure drop in serrated-fin tube bundles with staggered tube layouts, Applied Thermal Engineering 30 (13) (2010) 1531–1537. doi:10.1016/j.applthermaleng.2010.02.019.
URL https://www.sciencedirect.com/science/article/pii/S1359431110000785

[47] Information on Aluminium 6060 - Thyssenkrupp Materials (UK).
URL https://www.thyssenkrupp-materials.co.uk/aluminium-6060.html

[48] S. V. Patankar, C. H. Liu, E. M. Sparrow, Fully Developed Flow and Heat Transfer in Ducts Having Streamwise-Periodic Variations of Cross-Sectional Area, Journal of Heat Transfer 99 (2) (1977) 180–186. doi:10.1115/1.3450666.
URL https://asmedigitalcollection.asme.org/heattransfer/article/99/2/180/382469/Fully-Developed-Flow-and-Heat-Transfer-in-Ducts

[49] Y. Ma, Y. Yuan, Y. Liu, X. Hu, Y. Huang, Experimental investigation of heat transfer and pressure drop in serrated finned tube banks with staggered layouts, Applied Thermal Engineering 37 (2012) 314–323. doi:10.1016/j.applthermaleng.2011.11.037.
URL https://www.sciencedirect.com/science/article/pii/S135943111100665X

[50] P. J. Roache, Perspective: A Method for Uniform Reporting of Grid Refinement Studies, Journal of Fluids Engineering 116 (3) (1994) 405–413. doi:10.1115/1.2910291.
URL https://doi.org/10.1115/1.2910291

A OPENFOAM CASE FILES

A.1 fvSchemes for gas region

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  v2106                                 |
|   \\  /    A nd           | Website:  www.openfoam.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/

FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
    default     steadyState; // CrankNicolson 0.7; (used for transient simulation)
}

gradSchemes
{
    default     Gauss linear 1.0;
    grad(U)     cellLimited Gauss linear 1.0;
}

divSchemes
{
    default             none;
    div(phi,U)          bounded Gauss linearUpwindV grad(U);
    div(phi,h)          bounded Gauss linearUpwind grad(U);
    div(phi,nuTilda)    bounded Gauss linearUpwind grad(U);
    div(phi,K)                                  Gauss linear;
    div(((rho*nuEff)*dev2(T(grad(U)))))         Gauss linear;
    div((nuEff*dev2(T(grad(U)))))               Gauss linear;
    div(div(phi,U))                             Gauss linear;
    div((nuEff*dev(T(grad(U)))))                Gauss linear;
}

laplacianSchemes
{
    default        Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
    default         corrected;
}
```

```
wallDist
{
    method          meshWave;
}
// ************************************************************************* //
```

A.2 fvSchemes for fin region

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  v2106                                 |
|   \\  /    A nd           | Website:  www.openfoam.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
    default     steadyState; // CrankNicolson 0.7; (used for transient simulation)
}

gradSchemes
{
    default         Gauss linear;
}

divSchemes
{
    default         none;
}

laplacianSchemes
{
    default         Gauss linear uncorrected;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
    default         uncorrected;
}

// ************************************************************************* //
```

A.3 fvSolution for gas region (steady-state)

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  v2106                                 |
|   \\  /    A nd           | Website:  www.openfoam.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
solvers
{
    "(p_rgh|p).*"
    {
        solver          GAMG;
        tolerance       1e-5;
        relTol          0.001;
        minIter         3;
        maxIter         300;
        nPreSweeps      1;
        nPostSweeps     2;
        nFinestSweeps   4;
        scaleCorrection true;
        directSolveCoarsestLevel false;
        cacheAgglomeration true;
        nCellsInCoarsestLevel 300;
        agglomerator    faceAreaPair;
        mergeLevels     1;

        smoother        DICGaussSeidel;

    }
    "U.*"
    {
        solver          PBiCGStab;
        preconditioner  DILU;
        tolerance       1e-8;
        relTol          0.0;
        maxIter         300;
    }

    "(h|k|epsilon|nuTilda).*"
    {
        solver          PBiCGStab;
        preconditioner  DILU;
        tolerance       1e-8;
        relTol          0.01;
        maxIter         300;
    }

    "omega.*"
    {
        solver          PBiCGStab;
```

```
        preconditioner  DILU;
        tolerance       1e-9;
        relTol          0.001;
        maxIter         300;
        minIter         1;
    }

    Phi
    {
        solver          GAMG;
        smoother        DIC;

        tolerance       1e-06;
        relTol          0.01;
    }

}

SIMPLE
{
    momentumPredictor yes;
    nNonOrthogonalCorrectors 1;
    pRefCell        0;
    pRefValue       0;
}

relaxationFactors
{
    fields
    {
        "rho.*"         1;
        "p_rgh.*"       0.3;
    }
    equations
    {
        "U.*"           0.7;
        "(h|e).*"       0.8;
        "(k|epsilon|omega|nuTilda).*" 0.3;
    }
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
```

A.4 fvSolution for gas region (transient)

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  v2106                                 |
|   \\  /    A nd           | Website:  www.openfoam.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSolution;
}
solvers
{
    rho
    {
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-7;
        relTol          0.1;
    }

    rhoFinal
    {
        $rho;
        tolerance       1e-7;
        relTol          0;
    }

    p_rgh
    {
        solver           GAMG;
        tolerance        1e-12;
        relTol           0.001;
        smoother         GaussSeidel;
    }

    p_rghFinal
    {
        $p_rgh;
        tolerance        1e-12;
        relTol           0;
    }

    "(U|h|k|epsilon|omega|R|nuTilda)"
    {
        solver           PBiCGStab;
        preconditioner   DILU;
        tolerance        1e-7;
        relTol           0.1;
    }

    "(U|h|k|epsilon|omega|R|nuTilda)Final"
    {
        $U;
        tolerance        1e-7;
```

```
        relTol           0;
    }
}

PIMPLE
{
    momentumPredictor   yes;
    nCorrectors         2;
    nNonOrthogonalCorrectors 1;
    pRefCell        0;
    pRefValue       0;

}

relaxationFactors
{
    equations
    {
        ".*"            1;
    }
}
// ************************************************************************* //
```

A.5 fvSolution for fin region (steady-state)

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  v2106                                 |
|   \\  /    A nd           | Website:  www.openfoam.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
{
    "(e|h)"
    {
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-06;
        relTol          0.1;
    }

    eFinal
    {
        $e;
        relTol          0;
    }
}

SIMPLE
{
    nNonOrthogonalCorrectors 0;
}
residualControl
    {

        "(h|e)" 1e-5;
    }

relaxationFactors
{
    equations
    {
        ".*" 1;
    }
}

// ************************************************************************* //
```

A.6 fvSolution for fin region (transient)

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  v2106                                 |
|   \\  /    A nd           | Website:  www.openfoam.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
{
    "(e|h)"
    {
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-06;
        relTol          0.1;
    }

    "(e|h)Final"
    {
        $e;
        relTol          0;
    }
}

PIMPLE
{
    nNonOrthogonalCorrectors 1;
    pRefCell        0;
    pRefValue       0;
    nCorrectors          1;
}

relaxationFactors
{
    equations
    {
        ".*" 1;
    }
}

// ************************************************************************* //
```

B PYTHON SCRIPTS

B.1 Mesh generation specification

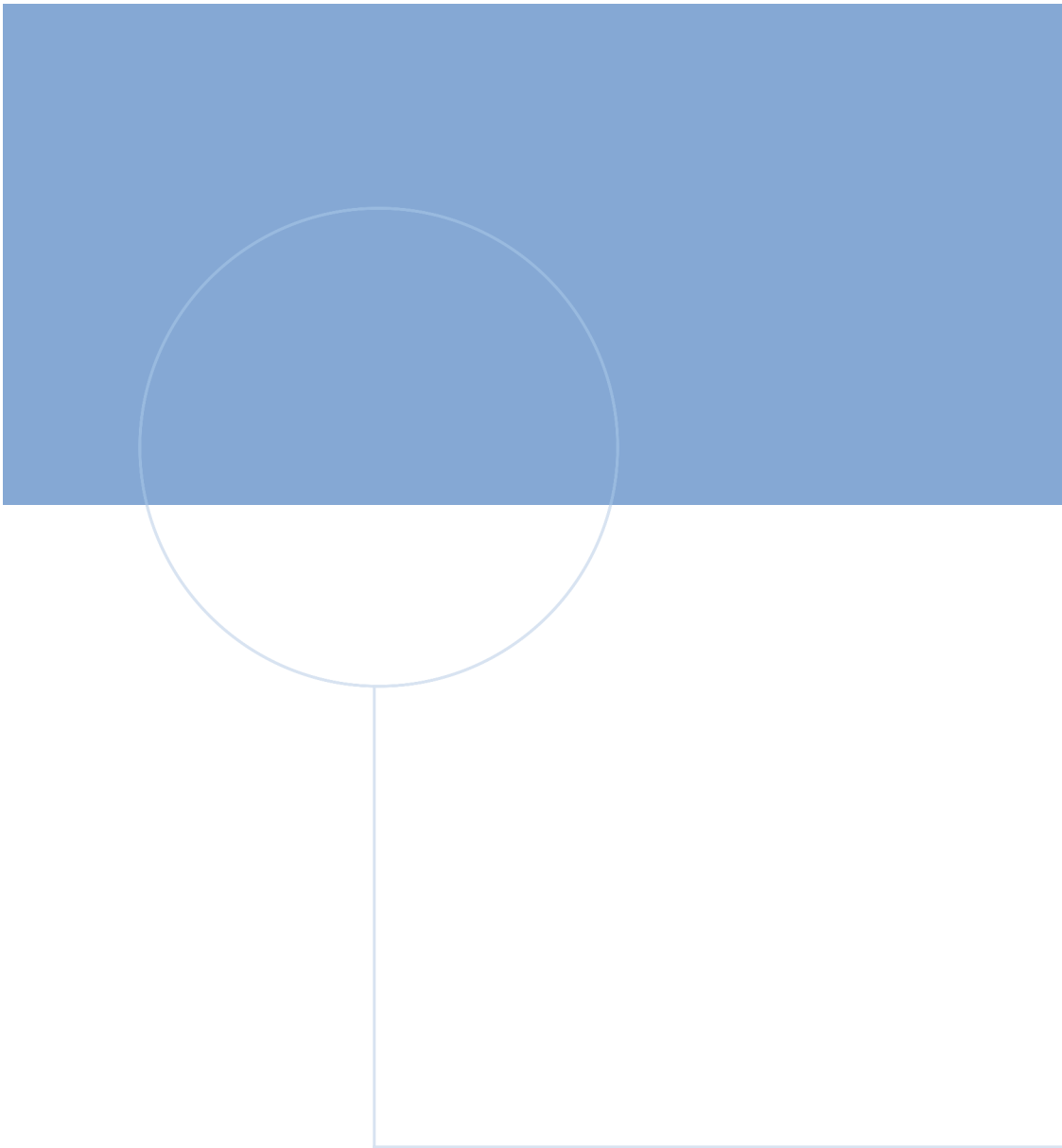
```
import HelicalFin

mesh = HelicalFin.HelicalFin(
    # Geometry no. 1 from Næss (2010), 0.3 mm cell size
    name = "N1_03",

    # longitudal tube pitch [mm]
    pl = 39.9,

    # transverse tube pitch [mm]
    pt = 46.1,

    # height of fin that is not serrated    [mm]
    h_f = 8.61,

    # height of fin that is serrated   [mm]
    h_s = 8.01,

    # outer diameter   [mm]
    d_o = 20.89,

    # width of serration    [mm]
    w_s = 3.97,

    # axial distance between fins   [mm]
    s_f = 5.08,

    # fin thickness [mm]
    t_f = 0.91,

    ## Grid parameters:

    # target average bulk cell size [mm]
    bulkCellSize = 0.3,

    # target thickness of cells in fins [mm]
    finCellThickness = 0.05,

    # BL cell growth ratio [-]
    r = 1.2,

    # first BL cell thickness [mm]
    startHeightBL = 0.01,

    # number of BL cells [-]
    numberOfBL = 15
)
```

B.2 Fin efficiency calculation

```
from scipy.special import kv, iv    # Modified Bessel functions
from numpy import tanh, sqrt
# Import mesh-object and thermal propoerties
from setBCs import mesh, solidKappa, kappa, Pr

# Values from simulation
deltaT = 17.595256250000006
A_fins = 0.0015302352766
A_tube = 9.79689072e-05
Q = 1.766859129

# Values calculated from the CFD simulation
alpahNum = 76.3715
eta_fNum = 0.7399
NuPr13Num = 38.75

def eta_th(mesh, kappa_f, htc):
    if mesh.isSolid():
        m = (2*htc/(kappa_f*mesh.t_f*1e-3))**0.5
        r1 = (mesh.d_o*1e-3)/2
        r2 = r1 + mesh.h_f*1e-3 + (mesh.t_f*1e-3)/2
        C = 2*r1/(m*((r2**2) - (r1**2)))
        D = iv(1,m*r2)*kv(1, m*r1) - iv(1,m*r1)*kv(1, m*r2)
        E = iv(0,m*r1)*kv(1, m*r2) + iv(1,m*r2)*kv(0, m*r1)
        return C*D/E
    else:
        he = (mesh.h_f + mesh.t_f/2)*1e-3
        m = sqrt(2*htc*((mesh.t_f+mesh.w_s)*1e-3)/
         ↪  (kappa_f*mesh.t_f*1e-3*mesh.w_s*1e-3))
        return tanh(m*he)/(m*he)

def eta_corr(mesh, kappa, htc):
    eta_t = eta_th(mesh, kappa, htc)
    if mesh.isSolid():
        return eta_t*(0.9 + 0.1*eta_t)
    else:
        return eta_t*(0.7 + 0.3*eta_t)

def htc(eta, deltaT, Af, At, Q):
    return Q/((eta*Af + At)*deltaT)

def diff(a, b):
    # Relative error [%]
    return 100*(a-b)/a

def calc(eta_func):
```

```
    # eta_func: function to calulcate eta (theoretical or corrected)
    eta = 0.9       # initial guess
    alpha = 1       # initial guess
    looping = True  # used to break loop
    i = 0           # iteration counter
    while looping:
        i += 1
        # calulate htc based on previous eta-value
        newAlpha = htc(eta, deltaT, A_fins, A_tube, Q)

        # update eta based on new htc
        newEta = eta_func(mesh, solidKappa, newAlpha)

        # check for convergence or terminate if i > 100
        if abs(newEta - eta)/eta < 1e-8 and abs(newAlpha -
        ↪  alpha)/alpha < 1e-8:
            print(f"Converged after {i} iterations using eta function:
            ↪  {eta_func.__name__}")
            return newEta, newAlpha
        elif i >= 100:
            looping = False
            print("Reached max. number of iterations, terminating.")
            return None, None
        eta, alpha = newEta, newAlpha

thEta, thAlpha = calc(eta_th)
corrEta, corrAlpha = calc(eta_corr)

thNuPr13 = (mesh.d_o*1e-3*thAlpha/kappa)*Pr**(-1/3)
corrNuPr13 = (mesh.d_o*1e-3*corrAlpha/kappa)*Pr**(-1/3)

# Print results
print("-"*76)
print(f"Parameter".ljust(15),
        "CFD".ljust(10),"|",
        "Theoretical".ljust(10),
        "Diff. [%]".ljust(9),"|",
        " Corrected".ljust(10),
        " Diff. [%]".ljust(9),"|")
print("-"*76)
print(f"alpha".ljust(15),
        f"{alpahNum}" .ljust(10),"|",
        f"{thAlpha: .2f}".ljust(10),
        f"{diff(alpahNum,thAlpha): .2f}".ljust(10),"|",
        f"{corrAlpha: .2f}".ljust(10),
        f"{diff(alpahNum,corrAlpha): .2f}".ljust(10),"|")
print(f"eta_f".ljust(15),
        f"{eta_fNum}".ljust(10),"|",
```

```
92          f"{thEta: .2f}".ljust(10),
93          f"{diff(eta_fNum,thEta): .2f}".ljust(10),"|",
94          f"{corrEta: .2f}".ljust(10),
95          f"{diff(eta_fNum,corrEta): .2f}".ljust(10),"|")
96  print(f"NuPr^(-1/3)".ljust(15),
97          f"{NuPr13Num}".ljust(10),"|",
98          f"{thNuPr13: .2f}".ljust(10),
99          f"{diff(NuPr13Num,thNuPr13): .2f}".ljust(10),"|",
100         f"{corrNuPr13: .2f}".ljust(10),
101         f"{diff(NuPr13Num,corrNuPr13): .2f}".ljust(10),"|")
102 print("-"*76)
```



NTNU
Kunnskap for en bedre verden

HighEFF