Computational Fluid Dynamics Analysis of Smart Manufacturing Oven Design

ECMM164 - MSc Dissertation
Candidate Name: Joel James Wright
Candidate Number: 630017158
Programme: MSc Mechanical Engineering
University: University of Exeter
College: CEMPS
Supervisor: Dr. Ion Sucala

21 August, 2020

Abstract

Computational Fluid Dynamics (CFD) is used in this report to help inform the company, Smart Manufacturing of the flow in their chamber design. The chamber is intended to dry 100 filters. The shape of the chamber induces a swirling flow around the filters.
OpenFOAM is the CFD software used to model and analyse the airflow in the chamber. Within the simulations, porosity effects and heat transfer are considered. Evaporation is simplified and calculated in post-processing.
An optimisation study has been conducted using one-factor-at-a-time screening technique to inform which parameters have the largest effect on the uniformity of drying in the chamber. However it is concluded that under the chamber design concept used in the study, uniformity of drying cannot be increased without the overall drying performance in the chamber decreasing.

Keywords: Computational Fluid Dynamics, OpenFOAM, Porosity, Swirling Flow, Optimisation Study.

Acknowlegment

Many thanks to Prof. Gavin Tabor for all the guidance offered, time given and attention paid to me and to my project whilst I have undertaken this research. Thank you for helping me grapple with CFD basics and for helping me better understand and use OpenFOAM.

Contents

Chapter 1

Introduction

The company Smart Manufacturing Ltd. has been awarded a tender from a company called Pall Inc. to create a machine that dries filters. Pall's filters are used in the pharmaceutical industry and little is known about these filters apart from approximate dimensions and a range of grain sizes. Smart Manufacturing Ltd have already produced a concept design for a drying chamber that circulates hot air. The design includes a cylindrical chamber, an inlet at the top of the chamber sending the flow in the radial direction and an outlet which is conical in shape that comes from the top down. The design has space for 100 filters and the filters are stacked in a circular pattern in five rows of 20.
In January 2020 when the concept was first explained to the author, a model was produced in Computer Aided Design (CAD), which has been used in the testing in this report.
It is understood that Smart Manufacturing shall have moved on from this original design, but it's evaluation can still give Smart Manufacturing intelligent design information as a proof-of-concept.
A concern of Smart Manufacturing is whether or not the filters will dry uniformly. There are certain design parameters that can be changed. For example, the dimensions and position of the inlet; the height and size of the outlet; the position in the chamber of the filters and the velocity of air entering the chamber.
In this report, the design concept will be analysed using computational fluid dynamics (CFD) to visualise and quantitatively analyse the airflow in different areas of the chamber. After the flow is understood, an optimisation exercise will take place, trying to optimise vertical uniformity in airflow through filters, in order to maximise drying uniformity and effectiveness of the design.
Chapter 2 contains a literature review of the uses of CFD in industry for swirling flow problems, the use of OpenFOAM and it's meshing utility snappyHexMesh. The literature review also summarises the background knowledge regarding the mathematics of CFD, mathematical models of flow through porous medium and heat transfer and also background to sensitivity analysis techniques.
Chapter 3 details how the knowledge collected in the literature review is applied in CFD and the methodology of how the case is set up in OpenFOAM.

In chapter 4 results of simulations and the implications of the optimisation study are discussed. Chapter 5 suggests additional work to be done in the future, in order to improve the study and finally chapter 6 offers a conclusion to the project and an outline of the findings.

Chapter 2

Literature Review

2.1 Computational Fluid Dynamics

Computational Fluid Dynamics (herein CFD) provides a way of modelling fluid flow in a system. CFD is used in the design stage of physical systemd that handles motion of fluid and/or heat transfer.
The domain of the system is specified either through a list of co-ordinates or specified using a CAD file. A mesh can then be created, the boundary conditions, simulation control parameters, turbulence model, solver and schemes may all then be specified before the simulation is run. Data from the simulation can be post-processed and then the simulation may be validated against a higher fidelity simulation or preferably, experimental results.

2.1.1 Navier-Stokes Equations

The Navier-Stokes Equations are the governing equations for fluid motion. The CFD software takes the user inputs, for example the boundary conditions and the mesh (see section 2.3.2 for more information on meshing) and applies this information to the following equations:

$$\nabla \cdot \underline{u} = 0 \tag{2.1}$$

Where:

∇ - vector differential operator

$\underline{u}$ - vector form of velocity $[ms^{-1}]$

Firstly, equation (2.1) is called the continuity equation. This equation makes sure that fluid flow cannot be created or lost; it binds whatever volume of fluid enters a cell, an equal amount of fluid must leave the cell, so what goes in must come out. $\underline{u}$ is made up of an x component, y component and z component.
The second equation is the momentum equation for incompressible flows:

$$\frac{\delta \underline{u}}{\delta t} + \underline{u}(\nabla \cdot \underline{u}) = -\frac{1}{\rho}\nabla p + \nu \nabla^2 \underline{u} + \underline{g} \tag{2.2}$$

Where:

ρ - density of fluid $[kgm^{-3}]$

p - dynamic pressure $[kgm^{-1}s^{-2}]$

$\frac{\delta \underline{u}}{\delta t}$ - variation in the flow over time

$\underline{u}(\nabla \cdot \underline{u})$ - convection

$\nu \nabla^2 \underline{u}$ - diffusion due to viscus forces (under laminar flow conditions)

$\underline{g}$ - external sources

The momentum equation (2.2) dictates the motion of the fluid. Here the momentum is written in its continuous vector format. Equation (2.2) may also be written in terms of partial derivatives and not in vector form, as shown below:

$$\rho(\frac{\delta u_x}{\delta t} + u_x \frac{\delta u_x}{\delta x} + u_y \frac{\delta u_x}{\delta y} + u_z \frac{\delta u_x}{\delta z}) = -\frac{\delta p}{\delta x} + \mu(\frac{\delta^2 u_x}{\delta^2 x} + \frac{\delta^2 u_x}{\delta y^2} + \frac{\delta^2 u_x}{\delta z^2}) + \rho g_x \tag{2.3a}$$

$$\rho(\frac{\delta u_y}{\delta t} + u_x \frac{\delta u_y}{\delta x} + u_y \frac{\delta u_y}{\delta y} + u_z \frac{\delta u_y}{\delta z}) = -\frac{\delta p}{\delta y} + \mu(\frac{\delta^2 u_y}{\delta x^2} + \frac{\delta^2 u_y}{\delta y^2} + \frac{\delta^2 u_y}{\delta z^2}) + \rho g_y \tag{2.3b}$$

$$\rho(\frac{\delta u_z}{\delta t} + u_x \frac{\delta u_z}{\delta x} + u_y \frac{\delta u_z}{\delta y} + u_z \frac{\delta u_z}{\delta z}) = -\frac{\delta p}{\delta z} + \mu(\frac{\delta^2 u_z}{\delta x^2} + \frac{\delta^2 u_z}{\delta y^2} + \frac{\delta^2 u_z}{\delta z^2}) + \rho g_z \tag{2.3c}$$

In the four equations there are four unknowns, u_x, u_y, u_z and p. However, There is a specific equation for u_x, u_y and u_z, but no specific equation to solve p directly. And the continuity equation is not an equation to solve, instead it is a restriction that the answer to the momentum equation must satisfy.

2.1.2 Solving the Navier-Stokes Equations

Therefore, solving for pressure, the three components of velocity and to satisfy the continuity equation is a challenge; this has been named the pressure-velocity coupling problem and algorithms have been written to do this.
A common approach to solve the pressure-velocity coupling problem is to predict the velocity, then solve for pressure and then correct for velocity (so it satisfies the continuity equation) in an iterative cycle. The two commonly used algorithms to do this are are the Semi Implicit Method for Pressure Linked Equations (SIMPLE) algorithm and the Pressure-Implicit with Splitting of Operators (PISO) algorithm. The main difference to note between the two algorithms is that the SIMPLE algorithm is for steady-state flows, so will not incorporate the unsteady term in the momentum equation $\frac{\delta \underline{u}}{\delta t}$. Whereas a the PISO algorithm is for transient flows, so the unsteady term of the momentum equation is retained.

In order to apply the Navier-Stokes equations as seen in this section, methods to discretize these continuous equations are required. Commonly in CFD, the Finite Volume Method takes the mesh of the geometry and is used to discretize the Navier-Stokes equations. The discretized Navier-Stokes equations then may be solved using the aforementioned algorithms. A summary of the mathematical differences between the PISO and SIMPLE algorithms can be found in Appendix A.

2.2 Modelling Turbulence

In the free stream of a flow, if the Reynolds number (defined in equation 2.5) exceeds a certain number based on the geometry, the flow may be assumed turbulent.
An eddy is a region of recirculation in turbulent flow. Within a turbulent flow there are commonly multiple eddies of different sizes. The size of an eddy can be defined by its length scale (diameter).
When a no-slip condition is applied to the walls, near the wall, the viscous forces dominate turbulent forces. Therefore, when a no-slip condition is applied, approaching a wall the length scale of eddies is reduced.
There are three main ways to model turbulence in CFD; one is through Reynolds averaged Navier-Stokes (RANS) turbulence models, another is through large eddy simulation (LES) and finally through direct eddy simulation (DES).

2.2.1 RANS

In a region of turbulent flow, the velocity through a given cell will fluctuate in a manner that is very difficult to predict, even at a very short time scale. This behaviour is shown in the figure 2.1.

Furthermore, this is happening in every cell across the region of turbulent flow. Therefore, it is not normally practical to resolve the fluctuations in velocity for most problems given computational limitation. However, a representative average can be taken over a sufficiently long time period compared to the fluctuation. RANS models average these fluctuations in velocity and pressure across each cell in a mesh, the mean velocity is shown in figure 2.1 by the orange line.
A presentation of the mathematics underpinning in RANS is held in Appendix B.

2.2.2 Note on LES

Figures 2.2 and 2.3 show a RANS and LES simulation on the same geometry with the same boundary conditions and the scale of the figures is the same. This illustrates the fundamental differences between the two approaches.
As discussed, RANS uses average velocity per cell (removing detail regarding velocity fluctuations as represented in figure 2.1) therefore averaging the turbulence through the cell. LES instead resolves the eddies with larger length scales, meaning larger eddies have a big effect

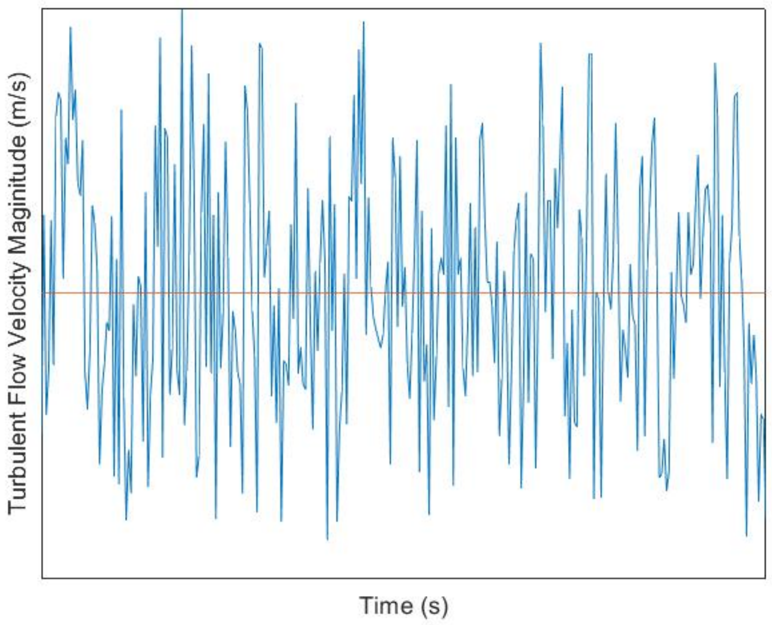


Figure 2.1: Velocity fluctuations and mean velocity

on the result of the simulations, whilst the effect of smaller eddies are ignored.
Figure 2.3 is taken at $Time = 1second$. Compared to RANS, LES offers CFD users an view of the instantaneous turbulent flow pattern.
RANS is still useful to engineers because it offers an average picture of the flow pattern. The main drawback of using LES is the computational cost compared to RANS.

2.2.3 Note on DES

DES is the purest form of CFD and by far the most expensive to compute. Pure DES resolves all fluctuations in velocity and therefore eddies of all length scales. Simulations results give a far more detail picture of instantaneous flow than even LES, and in the large majority of engineering cases, DES is too expensive to use over a reasonable time period.

2.3 OpenFOAM

2.3.1 OpenFOAM Overview

When there is a requirement to undertake CFD analysis, an engineer and a company has multiple options regarding the programme they use. Commercially, the largest software is ANSYS, which incorporates the CFD codes Fluent and CFX, a long with a host of Finite Element Analysis (FEA) codes. Another commercial option is Star-CCM+, but both these

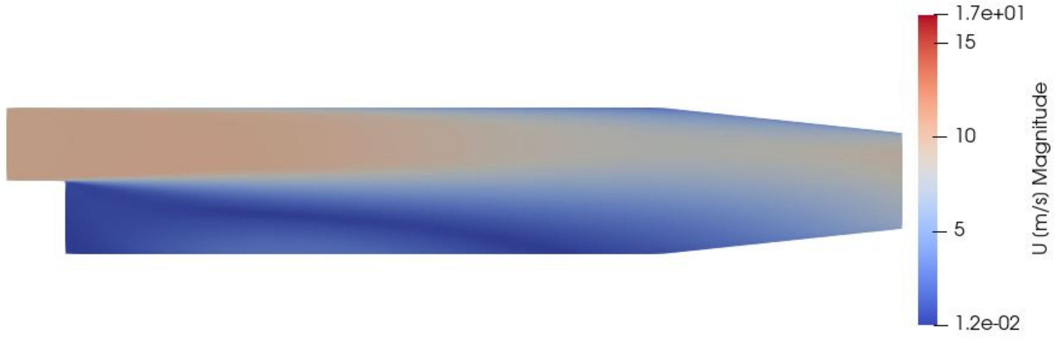


Figure 2.2: RANS simulation - pitzDaily case

commercial options require the company or the individual to pay large licence fees.
OpenFOAM which stands for "Open-source Field Operation And Manipulation" is an alternative to ANSYS codes or Star-CCM+. The first advantage of using OpenFoam over many commercial code is that the code is free to download and there are no licence costs, making CFD analysis accessible to individuals and SMEs.
FOAM was originally developed by a team at Imperial College London, headed by Prof. Henry Weller in the 90's.[1] The open-source software was released in late 2004.
OpenFOAM is a C++ library of applications, using an object-orientated structure, comprising of header and case files for its solvers and utilities.[1] OpenFOAM's main purpose is to run CFD simulations, but it also has solvers that are capable of conducting stress analysis

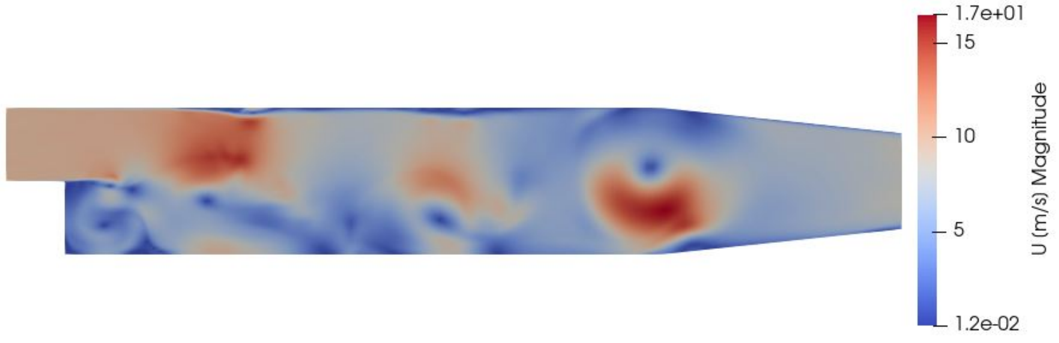


Figure 2.3: LES simulation - pitzDaily case

and also financial calculations.
As its name states, OpenFOAM is an open-source platform, it now has a large community of users. It is a relatively simple to compile a solver and add to the code. When a new version is released, it is common for the code to introduce community led innovations.
However, a downside to a large community developer and user base is agreeing on a path forward. There are two versions of OpenFOAM, stemming from a fracture in the community. Throughout this project, OpenFOAM.org version 6.0 was used.
A second downside to OpenFOAM, is that there is no guide user interface to the programme, like that of ANSYS Fluent. However, OpenFOAM is compatible with the visualisation tool ParaView, where the OpenFoam user can open and see, geometry files mesh files and view and post process simulation results. Instead of using a guide user interface, the OpenFOAM operator sets the simulation conditions in case files and dictionaries.
Within a case directory there are three subdirectories. One called `0` containing the files that set the initial conditions, one called `system` that contains files used to set-up and specify the simulation and one called `constant` that contains mesh files and other information that is constant throughout the simulation. Once the simulation parameters have been specified, the simulation is run from the case file directory in the terminal command line, by calling an in-built or user-defined solver.

2.3.2 snappyHexMesh

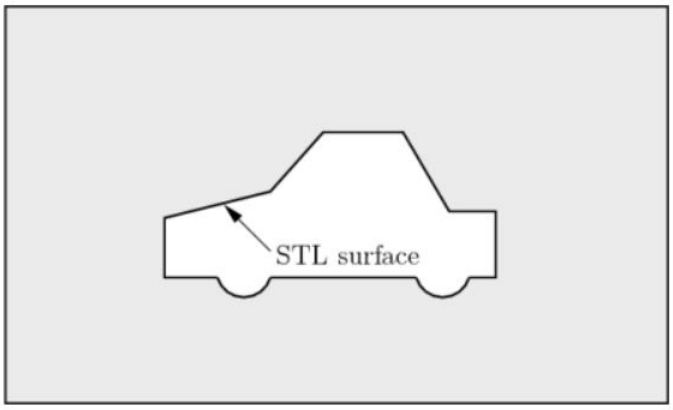


Figure 2.4: This is an example of an .stl file to be meshed

snappyHexMesh is an in-built utility for meshing a geometry in openFOAM. snappyHexMesh requires a CAD file containing the geometry of interest to be placed in `triSurface` within the `constant` directory. There are requirements on what type this CAD file can be, but it is common to use a .stl file written in ascii format.
The dictionaries `blockMeshDict`, `surfaceFeatureExtractDict` and `snappyHexMeshDict` need to be contained within `system` directory.
`blockMeshDict` specifies a simple geometry through coordinates, normally a cuboid, fully enclosing the CAD file(s), as shown in figure 2.5.

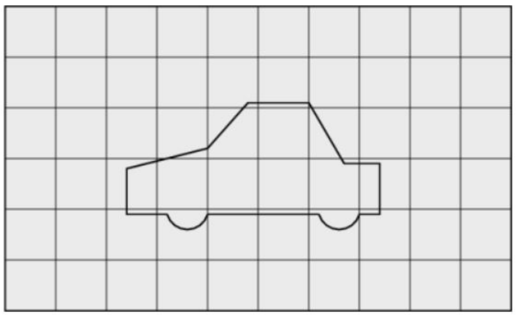


Figure 2.5: blockMesh surrounding .stl file

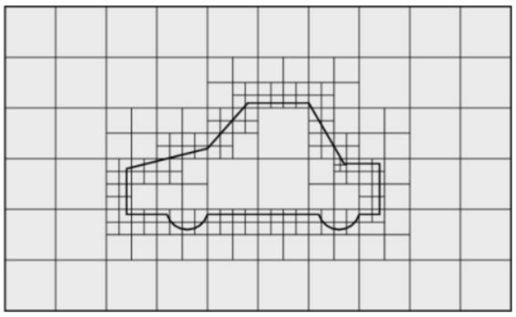


Figure 2.6: snappyHexMesh refining using levels

`surfaceFeatureExtractDict` species the CAD file(s) in use. The surfaceFeatureExtract command creates new .eMesh files that are used in `snappyHexMeshDict` to refine the edges around the geometry.
`snappyHexMeshDict` contains four sections; geometry, castellatedMeshControls, snapControls and addLayersControls.
Firstly, the Geometry specifies the geometry files and the refinement regions that need to be considered. castellatedMeshControls contains a lot of the information regarding refinement - the level of refinement for the refinement surfaces, feature edges and refinement regions are all specified here.
Levels are the way of controlling how refined the mesh becomes in the given region of interest. In the refinement process, when the surface of the geometry intersects a cell, the cell is split. In three dimensions, when one cell is split evenly, one cell becomes eight cells per level of refinement. This is shown in 2D in figure 2.6.

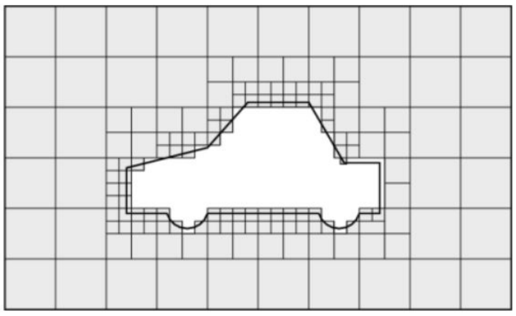


Figure 2.7: Removal of cells inside the geometry

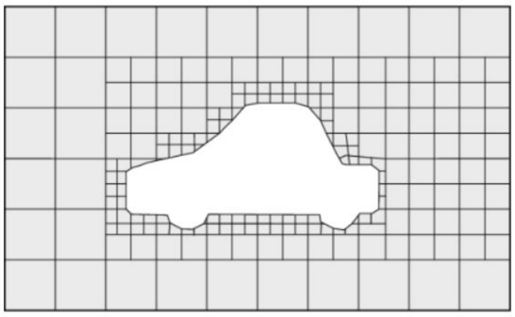


Figure 2.8: Snapping cells to the geometry boundary

The higher the level of refinement, the more accurate the mesh but the larger number of cells there will be in the mesh the more computationally expensive it will be to solve.
A very important input is the locationInMesh co-ordinate. This tells snappyHexMesh whether to keep the mesh internal or external to the CAD file(s) when cells are removed, as shown in figure 2.7. In this example, the locationInMesh cordinate is outside of the car, so

the cells inside of the car are deleted. If the locationInMesh coordinate was inside the car, snappyHexMesh would mesh the inside of the car and delete the area between the blockMesh boundary and the outside of the car.
After cells are removed, there will be some cells that overlap with the outer bounds of the geometry. The snapControls give the user an opportunity to change the default settings regarding snapping to the surface of geometry, as shown in figure 2.8.
Finally, layers around the boundary of the geometry can be added using the addLayersControls. This is especially helpful if a wall is a particular area of interest. This is shown in figure 2.9.

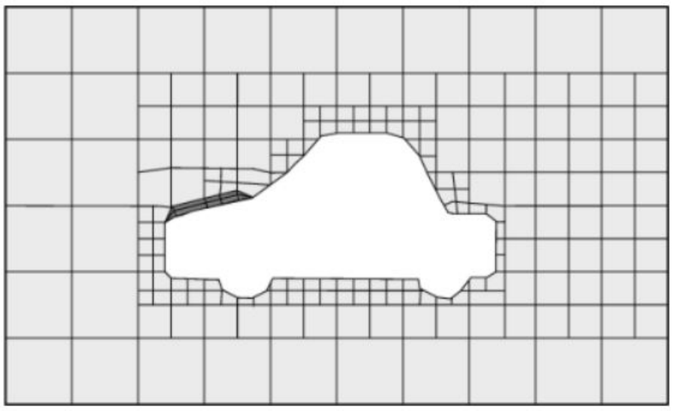


Figure 2.9: This gives an example of snappyHexMesh adding layers to a part of the mesh

Figures in this section were taken from cfd.direct's OpenFOAM user guide, section 5.4 Mesh generation with snappyHexMesh.[2]

2.4 Swirling Flows

The Smart Manufacturing design, will include a cylindrical chamber and the original design intent was for this geometry to encourage swirling flow. Two geometries that have been widely studied that also encourage swirling flow are that of a vortex flow separator and a hollow cylindrical heat exchanger.

2.4.1 Investigations into vortex flow separators

The geometrical proportions defined by coefficiants of the Lapple cyclone (as shown in figure 2.10) are commonly used in literature for vortex flow separators, otherwise called gas cyclone separators. Vortex flow separators are commonly used to separate solids particles of different sizes using the two outlets. Wang et al. (2006) present a numerical study, using the Lapple geometry; presenting different turbulence models for a range of particle sizes, comparing them to experimental results. Factors that affect the simulations were also identified. A particular observation of interest is that the separation efficiency of particles entering the vortex flow separator at the bottom of the inlet was weaker than of particles entering at the top.[3] Therefore there was a distribution of performance across the inlet and this is likely to be analogous. The authors observe areas of recirculation, and comment on where sediment may build up in the chamber, as well as suggesting that decreasing particle collisions is the key to maximising separation efficiency in cyclone design. For the Smart Manufacturing chamber design optimisation, the distribution of flow on all vertical levels needs similar to achieve uniform drying results.

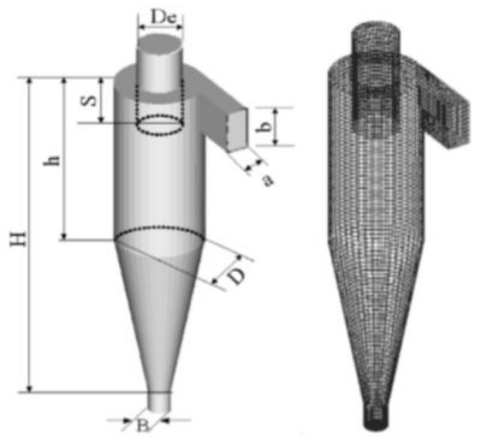


Figure 2.10: Lapple vortex flow separator and mesh[3]

Dhakal, Walters and Strasser (2014) also take geometry of the Lapple cyclone as well as a variation of the Lapple cyclone, in order to investigate several turbulence modelling approaches. The approaches used are the $k - \omega - \nu^2$ model, SST $k - \omega$ model and differential Reynolds stress transport model (DRSM). These approaches are also compared to LES and experimental data. The report concludes that the $k - \omega - \nu^2$ model, which is a non-linear eddy viscosity model that Dhakal and Walters produced[4], outperforms linear viscosity models (specifically in this instance the SST $k - \omega$ model). This was validated on both geometrical configurations. It is therefore suggested that this model provides a cost-effective alternative to rigorous modelling away from RANS.[5] This conclusion is supported by other literature.[6][7]
Azadi, Azadi and Mohebbi (2010) conducted a study using three sizes of the Lapple cyclone proportions. By varying inlet velocities on the three models, the authors compared performance parameters. Their simulation results and experimental data suggest that when increasing the cyclone size, pressure drop and the cut-off diameter increase. And as inlet velocity increases, cut-off size decrease.[8]
However the validity of experimentation and the authors conclusion have been criticised by Zhao and Su (2015). Zhao and Su point to more factors than size affecting particle cut-off diameter and suggest the authors should have used a higher fidelity method to verify the accuracy of their CFD. [9]

2.4.2 Flow in hollow-cylindrical heat exchanger

Saqr et al.(2012) model the decaying annular vortex flow in a hollow-cylindrical heat exchanger.[10] In this report, there is a comparison between the standard $k-\epsilon$ turbulence model and an $R_\epsilon/k-\epsilon$ turbulence model. The simulation results are compared to experimental data. The result of this suggests the $R_\epsilon/k-\epsilon$ model is the more accurate of the two turbulence models for these flow conditions. Coefficients for an equation to describe the decay of swirl intensity were derived from the experimental data. From the computational and experimental analysis, it was concluded there were multiple forced vortex regions multiple recirculation zones near the inner wall, as well at the free vortex near the outer wall. [10] This report by Saqr et al. (2012) offers an insight of how to measure and document the levels of swirl across a domain and a guide to presenting those results.

2.5 Porosity

Fluid flow is modelled mathematically by the Naiver-Stokes equations. CFD software uses the Naiver-Stokes equations as the basis of all solvers, and source terms may be added to the equations.
Flow through a porous medium can be modelled in one of two ways: macroscopically, where volume averaging takes place and flow is calculated arithmetically, or secondly microscopically, where the individual structural elements are modelled within the porous medium. Examples are modelling flow through foams of different materials. [11] [12].
The application in this report will be a macroscopic approach. Rather than through using a very fine mesh, a simpler way of modelling flow through porous region is by using an equation and adding it as a Source term within the CFD software. This is the same for modelling the Pall filters within the CFD model. In this section mathematics behind this approach used is discussed.

2.5.1 Darcy Law of Permeability

Darcy's law (1856) was the first of its kind to link a pressure drop to characteristics of fluid flow in a porous medium through experimental observation.

$$\frac{\Delta p}{L} = \frac{\mu}{\kappa}U \tag{2.4}$$

Where:

$\frac{\Delta p}{L}$ - pressure drop per unit length

μ - dynamic viscosity $[kgm^{-1}s^{-1}]$

U - magnitude of velocity of the fluid through the porous medium $[ms^{-1}]$

κ - permeability coefficient $[m^{-2}]$

Hellstrom and Lundstrom 2006 [13] show Darcy's law correctly describes the flow in porous materials for low flow velocities. Another way of expressing the velocity of the fluid relative to properties of the fluid, and categorising the validity of Darcy's Law, is through the Reynold's Number.

$$Re = \frac{\rho U L}{\mu} \tag{2.5}$$

Where:

ρ - density of fluid flowing through the porous medium $[kgm^{-3}]$

L - characteristic length scale of the porous medium $[m]$

When the Reynolds number reaches an upper limit, discrepancies can show in the experimental data. Experimental literature disagrees on the upper limit of the Reynolds number. Nevertheless to synthesise the research; the validity of the liner relationship between velocity and pressure drop seen in the Darcy Equation (2.4) may be called into question when $Re \geq 1 - 10$ [14][15][16] [17]. What the studies all agree on is that there is an upper limit of applicability of the Reynolds for the Darcy equation to yield accurate and valid results.[18][19]

2.5.2 Darcy-Forcheimer

2.5.2.1 Darcy-Forcheimer model

When fluid passes through a porous medium, there is a pressure drop. This can be mathematically modelled using the Darcy-Forchheimer equation:

$$\frac{dp}{dx} = \frac{\mu}{\kappa}U + \frac{1}{2}\beta\rho U^2 \tag{2.6}$$

Where:

$\frac{dp}{dx}$ - pressure drop per unit length derivative $[kgm^{-2}s^{-2}]$

β - Forchheimer coefficient $[m^{-1}]$

It can be clearly seen that the Forchheimer part of equation (2.6) introduces a variation on the kinetic energy of the fluid. At low velocities the Darcy part of the equation is dominant and at higher velocities, the Forcheimer part is dominant. Bear observes that at low velocities, viscous forces that resist flow are dominant. However at higher velocities inertial forces dominate. This means that at low Reynolds numbers the porous forces are linear, then there is a transition between linear and exponential before the exponential dominance takes over. Bear suggests that exponential dominance starts at around $Re = 100$.[19]

2.5.2.2 Darcy-Forcheimer experiment

Experimentally, the Darcy coefficient $1/\kappa$ and the Forchheimer coefficient β can be found.[18] [20]
The experimental set-up requires the velocity through a porous medium and the pressure difference across a porous medium to be measured. The velocity is varied and different readings of pressure drop as a function of velocity are plotted. The resulting Data can be curve fit to equation (2.6), and the Darcy coefficient $1/\kappa$ and the Forchheimer coefficient β may be found. [18] [20]

2.5.3 Calculating the Darcy-Forchheimer coefficients theoretically

2.5.3.1 Deriving the Kozeny-Carman equation

In 1927 Kozeny derived an equation linking the fluid flow through a porus medium with the quantities permeability κ and porosity ϕ. Kozeny origianals equation was:

$$U = \frac{\phi^3}{k\mu S^2}\frac{\Delta p}{L} \tag{2.7}$$

Where:

ϕ - Porosity [Unitless]

S - Surface Area per unit volume [m^{-1}]

k - Kozeny's constant [Unitless]

After Experimentation, Kozeny originally set the constant is name now bears at 5.0, but this number is no longer seen as valid[21].
The porosity of the material is given by the void volume left by the pores divided by the total volume of the porous material, which can be written mathematically as:

$$\phi = \frac{V_{viod}}{V_{total}} \tag{2.8}$$

In Kozey's original equation, it is assumed that pores in the material are are perfect sphere's. The ratio of material in the porous medium to empty space is given by $1-\phi$. This needs to be taken into account when calculating S. The surface area of a sphere is given by πD_g^2 and volume is given by $\frac{1}{6}\pi D_g^3$ in terms of D_g, the average pore grain diameter.

$$S = (1-\phi)\frac{\pi D_g^2}{\frac{1}{6}\pi D_g^3} \tag{2.9}$$

And this simplifies to:

$$S = \frac{6(1-\phi)}{D_g} \tag{2.10}$$

Equation (2.10) may now be integrated into to (2.7).

$$U = \frac{\phi^3}{36k\mu(1-\phi)^2} D_g^2 \frac{\Delta p}{L} \tag{2.11}$$

Equation (2.11) may be substituted into the Darcy Law (2.4) simplify the equation futher:

$$\kappa = \frac{\phi^3}{36k(1-\phi)^2} D_g^2 \tag{2.12}$$

In 1939, Carman published experimental work showing a more accurate Kozeny constant k, and equation (2.13), named the Kozeny-Carman equation is often regarded as the best theoretical approximation for permeability, when assuming ideal spherical pores.

$$\kappa = \frac{\phi^3}{150(1-\phi)^2} D_g^2 \tag{2.13}$$

This value of permeability κ can be used in equation (2.6) to the Darcy part of the Darcy-Forchheimer equation:

$$\frac{\Delta p}{L} = 150\mu \frac{(1-\phi)^2}{\phi^3} \frac{U}{D_g^2} \tag{2.14}$$

2.5.3.2 Ergun Equation

Work done by Ergun (1952) affirms the Kozeny-Carman equation and builds upon it, providing a theoretical method for calculating the Forchheimer coefficient as well.
Ergun's methodology was different to approach shown previously. Firstly Ergun was working with data regarding fixed beds - cylindrical chambers containing sediment or another material that fluid must pass through. Ergun noticed that the total energy losses in fixed beds can be treated as the sum of viscous losses and the sum of kinetic energy losses. Viscus energy losses are proportional to $\frac{(1-\phi)^2}{\phi^3}$ and the kinetic energy losses are proportional to $\frac{(1-\phi)}{\phi^3}$ [22]. Irmay(1958) also reached this conclusion [23].
Therefore Ergun writes the equation of energy loss across the pours medium as:

$$\frac{\Delta p}{L} g_c = k_1 \mu \frac{(1-\phi)^2}{\phi^3} \frac{U}{D_g^2} + k_2 \frac{(1-\phi)}{\phi^3} \frac{GU}{D_g} \tag{2.15}$$

Where:

g_c is the gravitational constant

$G = \rho U$

k_1 and k_2 are numerical constants

Ergun linearises the right hand side of equation (2.15) then simplifies:

$$\frac{\Delta p}{L} g_c \frac{D_g^2}{\mu U} \frac{\phi^3}{(1-\phi)^2} = k_1 + k_2 \frac{N_{Re}}{(1-\phi)} \tag{2.16a}$$

$$\frac{\Delta p}{L} g_c \frac{D_g^2}{\mu U} \frac{\phi^3}{(1-\phi)^2} = f_v \tag{2.16b}$$

$$f_v = k_1 + k_2 \frac{N_{Re}}{(1-\phi)} \tag{2.16c}$$

Where:

$$N_{Re} = \frac{D_p G}{\mu}$$

Equation 2.16c shows a linier relationship between f_v and $\frac{N_{Re}}{(1-\phi)}$. Ergun used this liner relationship, 640 pieces of data from fixed bed experiments and the statistical method of least squares to find the coefficients from (2.16c) as $k_1 = 150$ and $k_2 = 1.75$.[22] The Ergun equation,in terms of porosity and grain diameter, may therefore be written as: [22]

$$\frac{\Delta p}{L} = 150\mu \frac{(1-\phi)^2}{\phi^3} \frac{U}{D_g^2} + 1.75\rho \frac{(1-\phi)}{\phi^3} \frac{U^2}{D_g} \tag{2.17}$$

Where the Kozey-Carman equation (2.14) offers coefficients for the Darcy equation only, the Ergun Equation (2.17) offers a theoretical approximation of the Darcy-Forchheimer coefficients based on only the filters grain size and porosity. The Ergun equation has been validated many time [24] [25] through other experimentation and is widely accepted to give theoretical approximation of the Darcy-Forchheimer coefficients[19] with the assumption that the porous medium has constant porosity and has spherical pores.
When comparing (2.6) and (2.17), the Darcy coefficient is given by:

$$\frac{1}{\kappa} = 150 \frac{(1-\phi)^2}{\phi^3} \frac{1}{D_g^2} \tag{2.18}$$

And the Forchheimer coefficient is given by:

$$\beta = 3.5 \frac{(1-\phi)}{\phi^3} \frac{1}{D_g} \tag{2.19}$$

2.5.4 Application to case

To summarise section 2.5, much of the mathematics behind modelling flow through porous media was defined and established by the mid 20th century. For the case at hand, flow through cylindrical filters, there will be equal pressure drop in two directions and it is assumed no flow can travel in the third direction, due to the plastic casing at the top and the bottom of the filter.

2.5.4.1 Theoretical Validity

The coefficients seen in the Darcy-Forchheimer equation are theoretically calculated using the Ergun equation.
Rather than calculating coefficient theoretically it would give far more confidence to measure these coefficients experimentally, however circumstances shall not permit experimentation taking place. Therefore, the numbers calculated theoretically offer a ballpark figure to use in the simulation.
Flow through the filters is a small part of the case as a whole. Therefore, using ball-park figures and modelling filters as porous zones, but should not be far from an accurate overall picture of flow in the chamber.

2.6 Heat Transfer

In CFD, two common methods of simulating heat transfer are by using compressibility and by using the boussinesq approximation.

2.6.1 Compressibility

In the Navier-Stokes momentum equation for compressible flow is shown below:

$$\frac{\delta \rho \underline{u}}{\delta t} + \underline{u}(\nabla \cdot \underline{u}) = -\frac{1}{\rho}\nabla p + \nu \nabla^2 \underline{u} + \underline{g} \tag{2.20}$$

Compared to the incompressible momentum equation (equation 2.2) an extra field term of density needs to be solved as well. Compressible simulations can be less stable and non-compressible simulations. Compressible simulation should be used if the velocity of fluid exceed 0.3 Mach or if there are significant changes in fluid density.

2.6.2 Boussinesq Approximation of Buoyancy

The Boussinesq approximation for buoyancy, takes advantage of the coefficient of thermal expansion β.

β is defined as the rate of change of density with respect to temperature at a constant pressure.[26]

$$\beta = -\frac{1}{\rho_0}(\frac{\delta \rho}{\delta T})_p \tag{2.21}$$

A linear approximation can be made for this partial derivative:

$$\beta \approx -\frac{1}{\rho_0}\frac{\rho - \rho_0}{T - T_0} \tag{2.22a}$$

$$\rho - \rho_0 \approx -\rho_0 \beta (T - T_0) \tag{2.22b}$$

$$\rho \approx \rho_0(1 - \rho_0 \beta (T - T_0)) \tag{2.22c}$$

The Buoyancy term may be written as:

$$\frac{\rho g}{\rho_0} \approx (1 - \beta(T - T_0))g \tag{2.23}$$

Where:

g - gravitational constant $[ms^{-2}]$

Therefore, the Boussinesq approximation for buoyancy may be written as a source term on Navier-Stokes momentum equation.

$$\frac{\delta \underline{u}}{\delta t} + \underline{u}(\nabla \cdot \underline{u}) = -\frac{1}{\rho}\nabla p + \nu \nabla^2 \underline{u} + (1 - \beta(T - T_0))g \tag{2.24}$$

This takes the form of incompressable flow, so is easier to solve. As shown in section 2.2, the laminar term in equation 2.24 may be replaced to take into account turbulence.
The Boussinesq approximation for buoyancy is a cheaper and therefore more efficient way of modelling heat transfer for small changes in temperature rather than via compressibility[26]. Furthermore this practice is commonly seen in literature.
Validity criteria are set out experimentally by Gray and Giorgini (1976) and computationally by Bückle and Peric (1991). These works both state that for air, the Boussinesq approximation for Buoyancy is valid for at temperature changes of 15°Kelvin starting at room temperature. [27] [28]

2.7 Evaporation

2.7.1 Offering a definition of Evaporation

Evaporation is the process of a fluid being vaporised from a liquid to a gaseous state. The aim of the Smart Manufacturing machine is to facilitate evaporation of moisture in the filters through a process of forced convection.
Evaporation takes energy, the two mechanisms involved are molecular motion (diffusion) and motion of fluid within the environment around the surface of the evaporating fluid (advection). In the case of a still lake, advection will have little direct effect causing evaporation, because near the water's surface, the velocity of the air is low due to a boundary layer that forms. Therefore, molecular motion/diffusion in this case has largest affect on evaporation. However, in the layer of air adjacent to the surface of the water, there will be a increased level of vapour, reducing the rate at which molecular diffusion and therefore evaporation may take place. [29]
In forced convection, evaporation is therefore a combination of advection with molecular diffusion. The more heat, the higher the diffusion rate; the higher the gross air flow through and around the domain of the water, the higher the level of advection. And according to Li and Heiselberg (2005), in forced convections scinarios, advection has the dominant affect. So the rate of evaporation is dependant on a number of physical factors: surface area of the water, water temperature, air temperature humidity and most importantly air velocity. [29]

2.7.2 Analytical Methods of estimating Evaporation

The law of partial pressure Dalton (1802) states that evaporation is proportional to the vapour pressure difference, at the surface of the evaporating liquid and in the air. Dalton goes on to include a relationship to velocity of the air. There has been more empirical and experimental work to find coefficents. Li and Heiselberg note that majority of the general evaporation rate formulee coined in the body of research takes the form:

$$E = \frac{(a + bV)(\Delta P)}{h_e} = \frac{h_w \Delta P}{h_e} \tag{2.25}$$

Where:

E - The rate of Evaporation $[kgm^{-2}s^{-1}]$

a - A coefficient

b - A coefficient

V - Velocity of air parallel to water surface

ΔP - Pressure difference between the surface of the water and the water vapour $[Pa]$

h_e -Evaporative heat transfer coefficient $[Wm^{-2}Pa^{-1}]$

h_w - Latent heat of vaporization of water $[Jkg^{-1}]$

2.7.3 Applying Analytical Methods to CFD

There are two main ways of modelling evaporation in CFD. One way is to run a multi-phase simulation and use a specific evaporation sub-model, to simulate the interaction between the fluids. An example of this is presented by Potham (2017) using the Volume of Fluid Method with an evaporation model they produced[30]. A second way is to run the simulation in single-phase and calculate evaporation analytically in post-processing in the regions of interest.

2.8 Systems Analysis: Optimisation

When designing a methodology to optimise the air flow in the chamber, it will be useful to consider existing techniques used in systems analysis.

2.8.1 One-Factor-At-A-Time Method

The one factor at time method is often used for open-loop sensitivity analysis in systems engineering. It offers statistical analysis for the effect of changing single variables on the system output. The methodology requires the system input and design limits of the system to be varied, and for an output of the system to be evaluated.
As the name suggests, one factor/variable (either a system input or design limit) is changed to it's highest and lowest limit whilst the other factors are at their default level, and the effect on the output of the system is measured. Once all the variables have been assessed for their affect on the output, the variable that has the single most dominant effect on the system output is known[31].
The major disadvantage of this methodology is that it is simply a screening technique, therefore it is not known how factors interact when changing multiple parameters at a time.[31]

2.8.2 Latin Hypercube Sampling Method

The Latin Hypercube Sampling Method is often used for closed-loop sensitivity analysis in systems engineering. Rather than being a screening technique, it is a global method, meaning the full range of system configurations are considered.
The steps to implement the Latin Hypercube Sampling are as follows:[32] [33]

1. Choose the number of sampling points, then list the system input and design variables and split the variables across their range into smaller ranges, based on the number of sampling points.

2. Randomly choose a number within the range for each sampling point for each variable.

3. Shuffle the variables to create a list of random simulation parameters.

4. Conduct each simulation and record each output for the number of sampling points.

5. Analyse the results.

Analysis of results can take place using mean and variance analysis, or through observation. By plotting the output against the range of inputs for each variable, some plots may show a correlation whilst other will appear random (no correlation). The plots showing correlation will be the dominating parameters and design decisions can me made accordingly.[32]
Latin Hypercube sampling is perfectly suited to analysing control systems but is not well suited to CFD because of the length of time each simulation takes to set up and complete.
The sensitivity analysis that uses global approach is widely regarded as more valuable than screening techniques[31], however, any sensitivity analysis is useful for making informed design and optimisation decisions.

2.8.3 Examples in CFD

An example of sensitivity analysis using CFD for each method is given below:
Aelaei, Karimian and Ommi (2019), used a variation of one factor at a time, taking six values and the default value across six design parameters to analyse the aerodynamic performance of a Delta Wing, which is a wing used on aircraft and missiles. The six design parameters used were wing semi-span, tip chord length, thickness, leading edge radius, front diagonal edge length and rear diagonal edge length. The output measures used to analyse their results were drag coefficient and lift coefficient from which the authors calculated an aerodynamic efficiency to plot their results against.[34]
Benson et al. Performed global sensitivity analysis on a model of a street canyon in the British city of York, not for optimisation purposes but to simply see the effect the parameters had on the model and compare that to recorded results in the city. Benson et al. studied four input parameters - the surface roughness, background roughness, wall roughness and background wind directions and used the measure normalised turbulent kinetic energy, at a certain location, as an output.[35] Benson et al. used over 10000 simulations in their sensitivity analysis, each simulation taking around 15 minutes to complete.[35].

Chapter 3

Methodology

3.1 Use of CAD and in the CFD Process

Throughout the optimisation process, multiple CAD files were created. Solidworks was used to produce the original CAD files and the CFD software OpenFOAM was used to set-up and run the CFD simulations. For each iteration of the optimisation process, when a CAD file was ready to be analysed in CFD, it was saved in .stl file format.

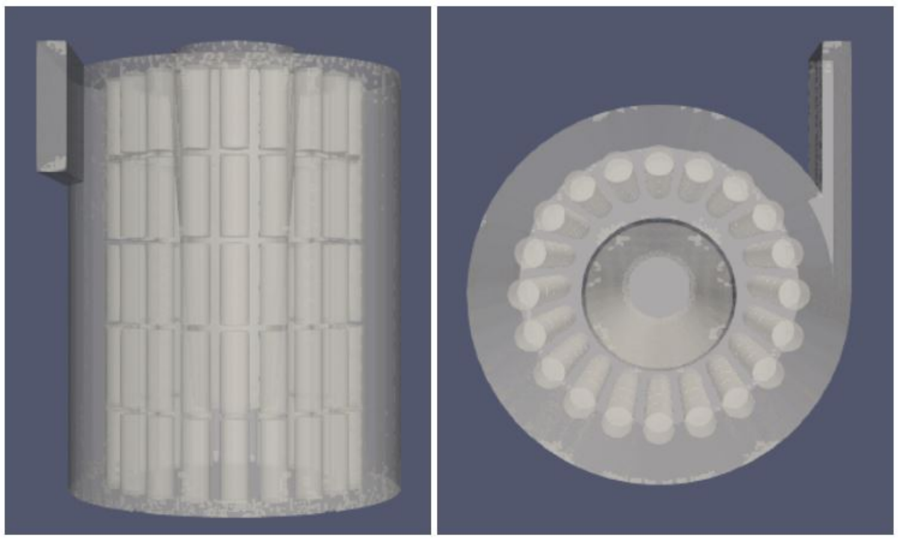


Figure 3.1: The default .stl file arrangement visualised in ParaView

The large majority of the simulations were hosted on the University of Exeter's Blade Server machines. These are high performance computing servers that are headless and may be accessed remotely, via the university's virtual private network and through using secure shell (ssh). Using the blade servers and running the simulations in parallel enabled the simulations to be completed in a shorter time than would otherwise be possible.

Once the geometry .stl file was ready, it was copied into the correct location within the case directory `case/constant/triSurface` along with an .stl file representing a single filter. A combination of python and bash scripts were used to print the locations of the 100 .stl filter files. The pattern of the filters within the chamber is shown in Figure 3.1. Then the files could be meshed (which is discussed in section 3.5). After meshing, the boundary conditions and the simulations conditions were set in the `0` and `system` directories respectively. When everything was set, the simulation was run in parallel.

3.2 Optimisation Methodology

The optimisation methodology used is one factor at a time sensitivity analysis. In order to implement one factor at a time optimisation, a functional output must be clearly defined.
In Smart Manufacturings design, the filters are fitted to a turntable, therefore radially, the filters in each row will receive the same average amount of air flow through them. However, there will be a distribution for how much air is received by each row of filters. Smart Manufacturing ltd. require a solution whereby filters are dried within the chamber uniformly. Therefore, the goal of the optimisation study is to increase uniformity of the drying of the filters in the chamber.
Modelling this as a dynamic case and using an evaporation model to estimate drying time for each of the filters is out of the scope of this study. However, evaporation is a function of the energy, and energy is a function and air flow through the filters.
There will be a mean air flow through each level of the filters. The level of air flow to and through each filter in the chamber, is defined by a functional unit.

3.2.1 Defining the Functional Units

A faceZone is the cells that make up the surface area of an object and a cellZone are the cells that make up the entire volume of the object. In order to monitor the airflow arriving at the surface of the filters and flowing through the filters, two functional units were used. The first was mass flow rate through the faceZone of each filter. The second functional unit was the velocity integrated by volume through the cellZone of each filter. The velocity through each filter is of course subject to porosity effects and this is accounted for in the simulations. The mass flow rate through the faceZones shall offer insight to the distribution of air arriving at the different filters at therefore fluid flow in the chamber, but this measure does not offer direct information about drying performance of the filters. On the other hand, as discussed in 2.7.1, velocity of air through the filter cellZones does offer insight into drying performance, as velocity is the most important indicator to the level of advection, which has a dominant effect in forced convection. Therefore, the sum of velocity through the filter volume will be the key functional unit in the optimisation process.

3.2.2 Analysing the Functional Units to Gain Insight

In the default design, logic dictates that the filters on the top row of the chamber nearest the inlet shall dry after a certain amount of time, however the filters on the bottom of the chamber may take much longer to dry.
Once the average value for air flow to and through the filters on each row is taken, there will be five values per functional unit per simulation. In order to assess uniformity, the range of the row average values can be taken.

3.2.3 Parameters to be varied in the Optimisation Study

The parameters of the chamber geometry that will be changed during the one factor at a time sensitivity analysis are: outlet height (position), outlet diameter, inlet height (position whilst keeping cross-section area constant) and filter radius. Each parameter will be varied in a way that is reasonable and an equidistant change from the default value.

Table 3.1: The limits of much how each parameter shall be varied in the optimisation study

Case:	Default Value	Percentage Varied	Comment
Outlet Height	$1295mm$	$\pm 22.78\%$	The outlet height will be varied by the length of half a filter.
Outlet Diameter	$250mm$	$\pm 20\%$	The top dimensions and the bottom dimensions of the outlet will vary the same amount.
Inlet Legnth	$400mm$	$\pm 25\%$	The cross sectional area of the inlet will be kept constant. The default dimensions are $400mm \times 90mm$.
Filter Radius Pattern	$425mm$	$\pm 6\%$	Filter radius pattern's movement is limited as the filters cannot touch
Inlet Velocity	$16.67ms^{-1}$	$\pm 25\%$	The Default volumetric flow rate is $0.6m^3s^{-1}$

So, there are five parameters that will be changed higher and lower than a default parameter - initially eleven simulations will be run, then ten mid-point simulations will take place, in order to check optimisation patterns. The post processing work to collate the measures of uniformity shall be completed via use of a python script. By the end of the analysis, the parameter with the largest effect on the system shall be highlighted.

3.3 Validation of Simulations

3.3.1 Statement Regarding the Nature of the Report

At the time of writing, Smart Manufacturing, their Customer Pall inc. and the University of Exeter have been unable to complete the signing of a non-disclosure agreement (NDA). The geometry used in the default simulation is therefore not a true reflection of the design owned by Smart Manufacturing. The default design created for the simulations was based off a discription of the January 2020 version of Smart Manufacturing's design.
The results of the optimisation study are therefore a 'proof of concept' and should therefore still be useful to Smart Manufacturing and if retained, should help to inform future design decisions they will undertake.

3.3.2 Providing Validation

The default simulation in the optimisation study would ideally be an exact replica of Smart Manufacturing's realised design, so that the simulation may be directly compared against a experimental results, in order to provide validation for the computational results. Given the lack of an NDA and the time-frame of the project, this cannot take place.
The default simulation geometry, with out the filters, will instead be validated against existing data provided for swirling flows in similar geometry, as discussed in section 2.4. It is assumed that if the flow in the chamber is valid without filters, then the flow in the chamber will be valid when the simulation contains the filters as well.

3.4 Creating The Solver

The solvers used throughout the optimisation process are called 'bouPisoFoam' and 'bouSimpleFoam. As their names suggest, they were compiled from pisoFoam and simpleFoam respectively, with the Boussinesq Approximation of Buoyancy added within the momentum equation then compiled.
The use of bouPisoFoam enabled the analysis of air flow and heat transfer, over a specified time period and can be used from when air is first enters the chamber to a specified time, ideally when the flow reaches steady-state. In order to resolve the swirling flow, the time step used in the PISO algorithm needs to be very small, therefore simulation shall take a long time to run as many calculations as are required. The SIMPLE algorithm assumes flow reaches steady-state. As made clear in section 3.2 the rotation of the filters is not modelled. This means that the flow in the chamber and around the filters will eventually reach a steady state.
The use of bouSimpleFoam is a much time more-efficient algorithm to see the flow in a steady-state form. Therefore the bouSimpleFoam will be used to conduct the optimisation study when multiple simulation shall be run.

3.5 Mesh Generation

As discussed in section 2.3.2 the features within snappyHexMesh are used to create a mesh for the chamber and a mesh for the filters that shall work in all configurations used in the optimisation study.
The chamber geometry is originally drawn in Solidworks and then saved as an .stl file, as is a single filter. Solidworks has a systematic error saving .stl files by scaling them by a factor of 1000, but this may easily be overcome using the OpenFoam command `surfaceTransformPoints`. Using the same command, the filter may be translated and written into each of the 100 different positions inside the chamber as shown in Figure 3.1. This process is automated using a bash script that reads a python script that prints a list of coordinates for the filter.
Once there are the 101 necessary .stl files in the `constant/triSurface` directory, the OpenFOAM commands blockMesh, surfaceFeatureExtract and snappyHexMesh are used in that order to complete the meshing process. As discussed in section 2.3.2 these commands are controlled by their own dictionaries.
blockMeshDict records the geometry of a volume around all the .stl files, and is made up from a certain amount of cells. surfaceFeatureExtractDict is used to list all the .stl files, from which surfaceFeatureExtract writes edge files. These edge files are then used by snappyHexMesh. snappyHexMeshDict is where the majority of the work to create a mesh takes place.

3.5.1 Content of snappyHexMeshDict

3.5.1.1 geometry

Under geometry, the .stl mesh files are listed as features and named, as well as refinement regions defined and named. A box shaped refinement region is defined around the inlet's join with the chamber and a long cylindrical refinement region is defined around the outlet.

3.5.1.2 castellatedMeshControls

Explicit feature edge refinement is required to create sharp edges in the mesh. It is done by listing the .extendedFeatureEdgeMesh files created by surfaceFeatureExtract, then by listing these files under features and also giving a level that snappyHexMesh will use to know how refined the edge of the feature needs to be.
Surface based refinement then takes place. Under refinementSurfaces, the name of each feature and levels for refinement is given. For the filters a faceZone and a cellZone inside the filter is defined here. This is important for section 3.2.1.
Once the refinement at surfaces has taken place, refinement inside regions may also take place under refinementRegions. Again, the name of the features is listed, along with refinement levels. Here, the inside of the region must be specified, as snappyHexMesh can refine inside or outside of a feature. This is also where the other refinement regions specified in the geometry section are included and given a refinement level.

Finally, locationInMesh is specified. In this case, snappyHexMesh needs to be told to mesh the inside the chamber, therefore a location inside the chamber is selected.

3.5.1.3 snapControls

After the basic mesh has been formed, the cells that intersect the geometry are snapped to the geometry in a process called snapping. The snapControls were largely increased the default levels, in order to help snapping process.
The addLayers feature of snappyHexMesh was not utilised.

3.5.1.4 Note on Levels and BlockMesh

The total amount of cells in the Mesh is dependant on the levels used to refine features within snappyHexMeshDict, and also the original level of refinement set in blockMeshDict. Changing the refinement of the blockMesh by a small amount can change the total number of cells in the mesh by a large amount, due to the nature of the levels of refinement.

3.6 Boundary Conditions

The boundary conditions give the starting criteria for a simulation. These are the initial velocity, the initial pressure, the initial turbulent viscosity ν_t, the initial turbulent kinetic energy k and the turbulent energy dissipation rate ϵ. It was assumed that the outlet of the chamber was at atmospheric pressure, so there were no exhaust fans applying negative pressure at the outlet. When the initial velocity is known, the equations for calculating the initial values of k and ϵ are detailed below in section 3.6.2. The boundary condition dictionaries list values for these coefficients at the patches. There is a patch for the inlet, a patch for the outlet and a patch for the walls.

3.6.1 Creating the Patches

After the mesh is created, the patches are created using a combination of topoSet and createPatch functions. Both of these functions are controlled by their own dictionary in the `system` directory. The user specifies the coordinates for three faceSets in topoSetDict, one faceSet for the inlet, one for the outlet and one for the other walls. These faceSets are listed in the createPatchDict.

3.6.2 Additional Equations for Boundary Conditions

The equation to calculate turbulent viscosity is in equation B.3 and in order to calculate an initial value for k and ϵ the following equations are used:

$$k = \frac{3}{2}\left(U_{ref}T_i\right)^2 \tag{3.1}$$

$$\epsilon = C_\mu^{\frac{3}{4}} \frac{k^{\frac{3}{2}}}{l} \tag{3.2}$$

$$l = 0.07L \tag{3.3}$$

Where:

L - Characteristic length scale of the inlet

U_{ref} - The velocity of the fluid at the inlet

T_i - The turbulent intensity of the fluid

C_μ - A coefficient from the $k - \epsilon$ turbulence model

3.6.3 Note on Turbulent Intensity

The turbulent intensity defines how turbulent the flow is. For a flow that is not at all turbulent, the average velocity particles in the free stream of flow (away from the effects of walls) would be similar to the fully developed flow velocity. Whereas in highly turbulent flow, due to eddies that form, there is a larger reduction in the average velocity in the free stream, compared to fully developed flow. The maximum turbulent intensity for incompressible flow is around 20%. Physically, the turbulent intensity is affected by the Reynolds number of the flow and the geometry by which the flow is constrained. In these cases the turbulent flow has been set at 10%. This is an arbitrary value that could be improved upon with an experimental reading. However there has been no access to this data, nor the design of the ducting inlet into the chamber.

3.7 Settings within the ControlDict

The file `system/controlDict` is used to set simulation controls and to hold function objects. The controlDict needs to be edited when running the simulation in transient (bouPisoFoam) or in steady state (bouSimpleFoam), due to how the algorithms use the this dictionary, in particular the Time Step and the End Time values.

3.7.1 Running Simulation in Transient and in Steady-State

For a transient simulation the settings are self-explanatory; the Time Step is set at the interval between the calculations and the End Time will be the time of the last calculation. In a transient simulation, the courant number - which is a tracking measure for partials moving through cells, should remain less than 1 for fidelity. If the courant number rises to a level greater than one then the time step should be reduced.
For a steady-state simulation using the SIMPLE algorithm, the Time Step is set to 1, then the maximum number of iterations is set using the End Time setting. The convergence

criteria is set within `fvSolution` dictionary and a good steady-state simulation shall be successful if the simulation converges to the given tolerances before the maximum number of iterations is reached.

3.7.2 Functional Objects

The function objects are bolt-ons to the simulation, they specify additional data to be recorded and are commonly used for simulation post processing.
In this case, the function objects used are porosity, surfaceFieldValue and volFieldValue. surfaceFieldValue will write a script containing the data of mass flow rate at faceZones of the filters. The volFieldValue specifies the mass integrated by volume across the cellZone of each filter and writes for the three components of velocity. These two measures show the amount of air flowing to the filter over a unit of time and the amount of air flowing through the filters.

3.7.3 Porosity

A third functional object is set up, which is required for the code to take into account the porosity effects. This also enables the force of the porosity effects to be measured. This functional objects links to another file, `fvOptions` in the `system` directory. Within `fvOptions`, the Darcy-Forcheimer coefficients for porosity are specified for each filter cellZone.
As discussed in depth in section 2.5, as the filter grain size is known, the Darcy-Forchheimer coefficients for porosity of the filters may be calculated theoretically. And the Ergun Equation is used. Porosity is described through a cellZone in the three Cartesian coordinates. In fvOptions directory the porosity coefficients is set for each filter (at each cellZone) therefore it would be possible to set up a case with different filters within the chamber. However, in the optimisation study, porosity coefficients are set for a grain size of $2\mu m$.

3.8 Post-Processing Methodology

In order to estimate the level of drying of the filters, the airflow arriving at the filters and the airflow through the filters is used. The fucntional objects write numbers into a .dat format, python scripts are used to extract the data from these files and then plot the data in the specified format. As the data will be instantaneous to that point in time, there will be a radial distribution of air flow through the filters which is of little interest - in Smart Manfacturing's machine, the filters will be rotating, therefore each filter shall experience the position of every other filter in their row. However the vertical distribution is of interest. Smart Manufacturing want to increase the uniformity of drying, therefore it is key to optimise the uniformity of the airflow through the filters.

The python script is applied after the simulation has concluded and it takes the printed data organises it into twenty columns of five, just as the filters are laid out. The difference between the highest and lowest mass flow rate in every column can be easily stored and visualised. There will be several columns of particularly great difference and it is these columns that will be reduced in the optimisation process.

3.9 Simulation Setup

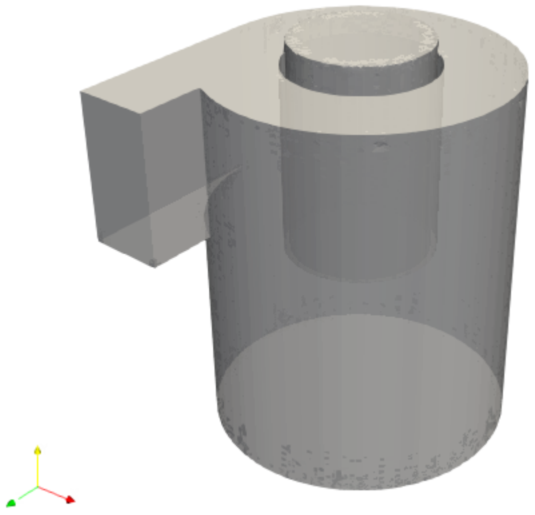
Before the simulations for the optimisation study were run, it was import to set universal conditions that all simulations were governed by. This was for accuracy and validity of the optimisation study. All of the boundary conditions were the same as the default simulation, apart from the one factor that changed per case. Away from boundary conditions, all of the case settings held in dictionaries in the `system` directory were the same for every simulation. A problem arose as follows:
Mesh refinement study took place under steady-state conditions and under the assumption that all the simulations would resolve at certain tolerances. A mesh was chosen that has a little over 1.13×10^6 cells for the default set-up. Cases were originally run to residuals of 10^{-3} for pressure and 10^{-4} for the other residues. Unfortunately, when completing the mid-point simulations, a few would not converge without lowering the residuals. All simulations were re-run with the same low solver residuals. Even though the convergence residuals are not good, this should not affect the overall result of the one-factor-at-a-time study. This is because as all the results are judged relative to the default result. One would not expect the results of the study to be vastly different if all the simulations were resolved at improved convergence criteria.

Chapter 4

Results and Analysis

4.1 Simulation Validation

In this section, CFD turbulence model and mesh used to model the flow in the chamber, will be compared to experimental data and the validity will be assessed.
The shape of the chamber, with an inlet, a cylindrical body and single outlet coming down into the chamber, is relatively unique. However, the flow between the bottom of the chamber and the bottom of the outlet, may be expected to behave like the flow near the top outlet of a vortex flow separator. And flow in the upper part of the chamber may be expected to behave like flow in a hollow cylindrical heat exchanger. Therefore, experimental data from two different peer-reviewed studies was used to validate the case.
The RNG $k - \epsilon$ turbulence model was chosen for it's stability, cost-effectiveness and reputation for good performance under swirling flow conditions. Its choice shall be assessed in this section. The default case was run with the same mesh snappyHexMeshDict, boundary conditions and solver parameters, but without the filters, so the chamber was empty in order to gain more of a direct comparison between the cases in literature. After the simulation of the empty chamber was run, in order to assess the turbulence model and the mesh under swirling flow conditions, velocity had to be converted from Cartesian coordinates to cylindrical (polar) coordinates. This was done in post processing using the software ParaView's Calculator function. The Plot Over Line function in ParaView was then used to extract the data of interest into .csv files and then these were plotted using Matplotlib in python.
The chamber was edited in both cases to better mimic the cases in the literature.



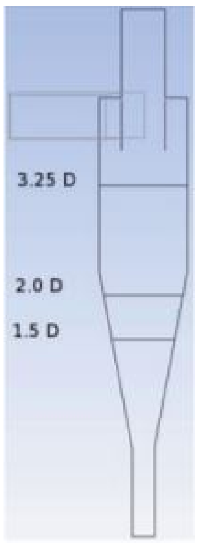
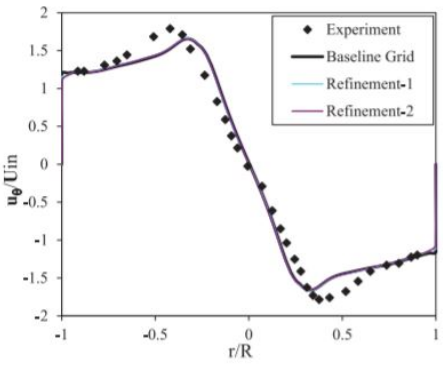


Figure 4.1: Left shows the edits made to an empty chamber (figure 3.1), using the same geometry as the upper part of the VFS (right)[5], in order to simulate flow in the equivalent position along the VFS's 3.25D line.



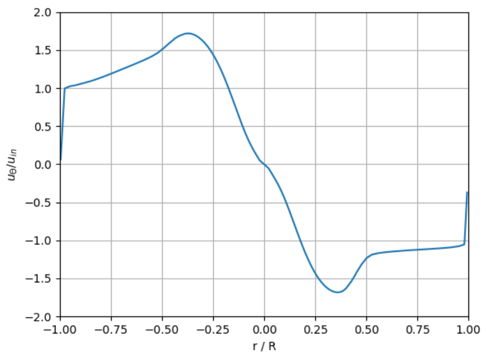
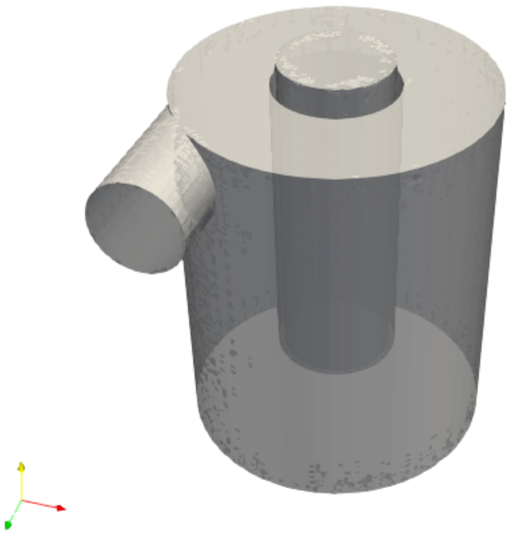


Figure 4.2: Image from [5] of experimental data and a mesh used for research

Figure 4.3: Flow profile at height 0.595m, equivalent to 3.25D on the VFS.

Figure 4.2 shows the experimental results used in the literature and figure 4.3 shows the results produced using the simulation. Overall the shape is very similar to that of the literature. The literature provides ratios in the geometry all based from a diameter of the chamber. The litrature does not provide the Reynolds number of the inlet fluid. This means that the boundary conditions will be different in the two simulations. But as the plot in figures 4.2 and 4.3 are relative to inlet velocity and radius, the variations in the data are therefore down to solver, mesh and turbulence model.



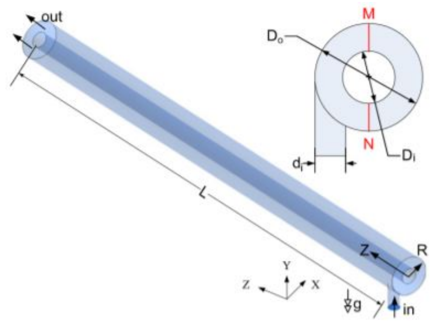
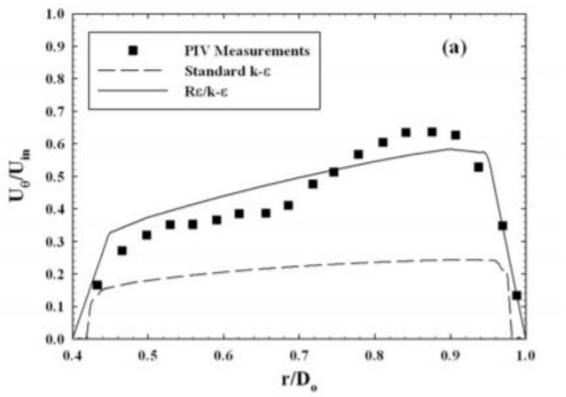


Figure 4.4: Left shows the edits made to an empty chamber (figure 3.1) in order to simulate flow between the inner and outer wall near the inlet with the same geometry as the heat exchanger model. The original heat exchanger geometry from the literature is shown (right)[10].



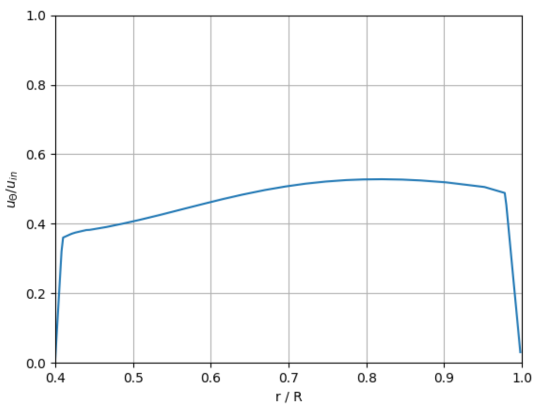


Figure 4.5: Image from [10] of experimental data and a mesh used for research

Figure 4.6: Flow profile near the inlet of chamber between an inner and outer wall

Figures 4.5 and 4.6 show a comparison between experimental data of a hollow cylindrical heat exchanger and the flow behaviour near the inlet of the chamber shown in figure 4.1. The geometry and the Reynolds number could be translated across to the model from the literature; however the biggest difference is that no relative position is given for where figure 4.5 is taken on the model. It states an exact distance from the inlet plane, but the author omits the diameter used in the model.
Nevertheless, the data from the RNG $k-\epsilon$ model again resembles the mesh used in literature, particularly in the swirling stream. Figure 4.5 contains three plots, one of experimental measurements, one of standard $k-\epsilon$ turbulence model and one of $R\epsilon/k-\epsilon$ turbulence model. It is easy to see that the $R\epsilon/k-\epsilon$ turbulence model maps more closely to the experimental data plots than the $k-\epsilon$ turbulance model plot. Data shown in figure 4.6 suggest the RNG $k-\epsilon$ turbulence model performs similarly to the $R\epsilon/k-\epsilon$ turbulence model in the free stream, but performs poorly, like the $k-\epsilon$ turbulence model at the walls. As the lines are nearly vertical, this shows that the RNG k-epsilon model and k-epsilon model struggle to model swirling flow, in the flow's boundary layer near a wall.
To summarise, there is room for improvement in the mesh, turbulence model and also solver residuals. But as shown by the resemblance to experimental data, the computational set-up is of suitable quality to give results of the desired validity, and show the behaviour inside the chamber in the free stream, which is where the large majority of air moves around the chamber. The turbulence model is particularly weak near the chamber walls, but this is a concern that shall be overlooked to accept the benefits use of the RNG $k-\epsilon$ offers; cost-effectiveness, stability and performance computing swirling flow in the stream.

4.2 Default Simulation Result

4.2.1 Comparison Between Running the Simulation in Transient and Steady-State Conditions

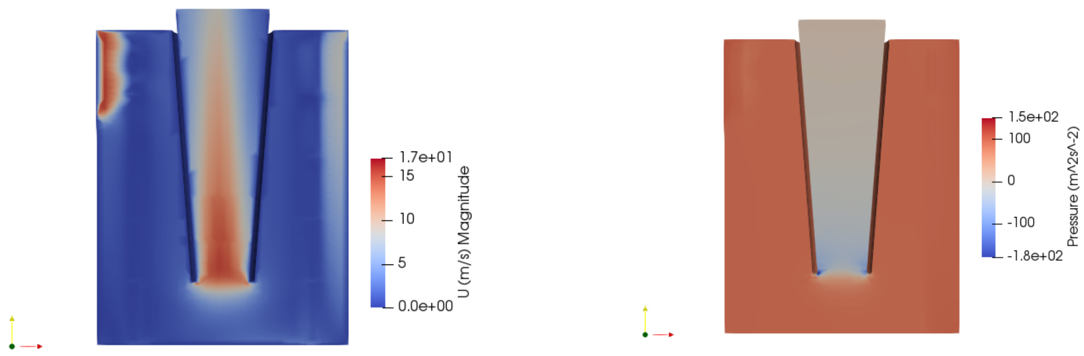


Figure 4.7: Two images taken from the transient simulation after 1 second. On the left a section clip of Velocity field and on the right a section clip of the pressure field.

The difference between running simulations in transient and steady-state conditions was outlined in section 3.4. For this case, with swirling flow, running the simulation in the PISO algorithm comes with a high computational cost. Running in parallel over four cores, the transient simulation took 29 hours to complete for 1 second of fluid flow. Whereas, on the same machinery and using a more refined mesh, the steady state simulation took around 20 minutes. This is down to the number of iterations completed by the algorithm at the requested level of tolerance. The transient simulation was completed in 100,000 iterations, the steady state simulation was complete in under 700 iterations.
The images of the results of the transient simulation run are shown in figure 4.7. And the thermal distribution in the chamber after 1 second is shown in figure 4.8. An issue was discovered and not resolved before the time of writing regarding how temperature field data was processed by the bouSimpleFoam solver (the complied steady-state solver). As shown by figure 4.8, the temperature profile makes its way to most areas of the chamber within 1 second. Under steady state conditions, it is probable that the large majority of the chamber will be near the air inlet temperature. Because of this, the analysis will focus on the flow of air, rather than the flow of thermal distribution and the effects of diffusion in evaporation.

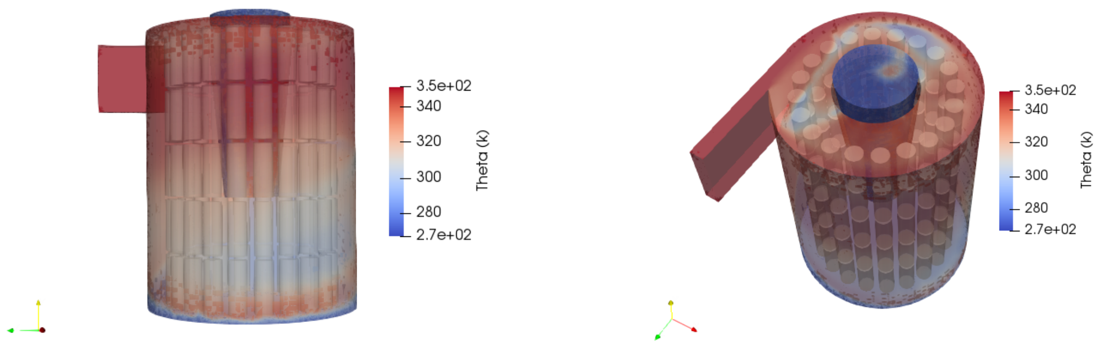


Figure 4.8: Two images of the temperature field taken from the transient simulation after 1 second.

4.2.2 Simulation Visual Results

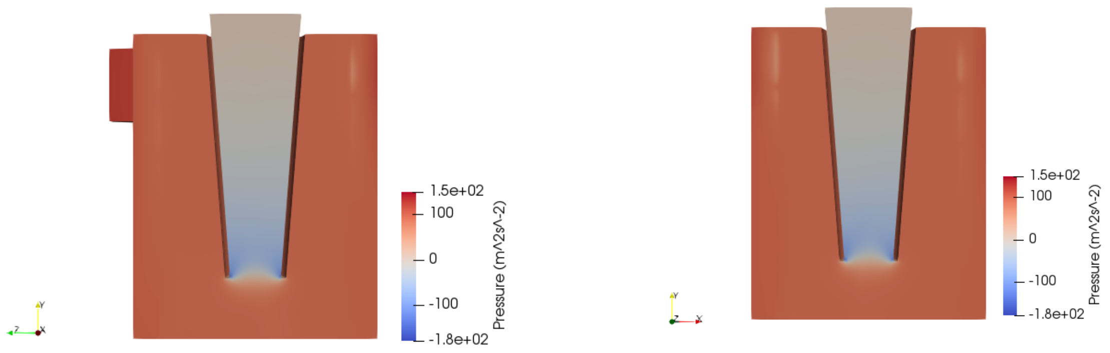


Figure 4.9: Pressure field distribution normal to the X-plane (left) and normal to the Z-Plane (right)

From the top of the chamber down, pressure in the chamber remains high, until the entry to the outlet near the bottom of the chamber, where the pressure is negative. This negative pressure has the effect of pulling air particles from the flow stream around and through the filters.

Also, near the inlet of the chamber, regions of lower pressure around the filters can be seen.

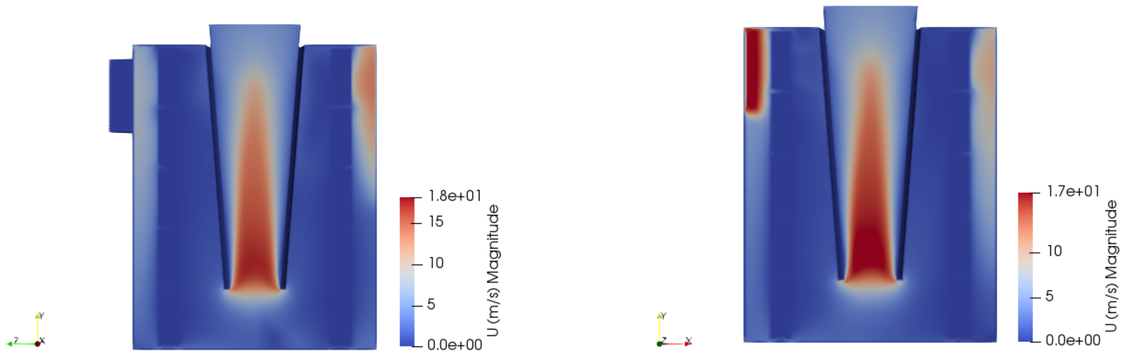


Figure 4.10: Velocity field distribution normal to the X-plane (left) and normal to the Z-Plane (right)

The velocity distribution is shown in figure 4.10. The figures show a decrease of velocity down the sides of the chamber and a large increase in velocity as particles are accelerated out of the chamber by the pressure of the outlet. The swirling flow is more clearly displayed in figure 4.11 by the used of streamlines.

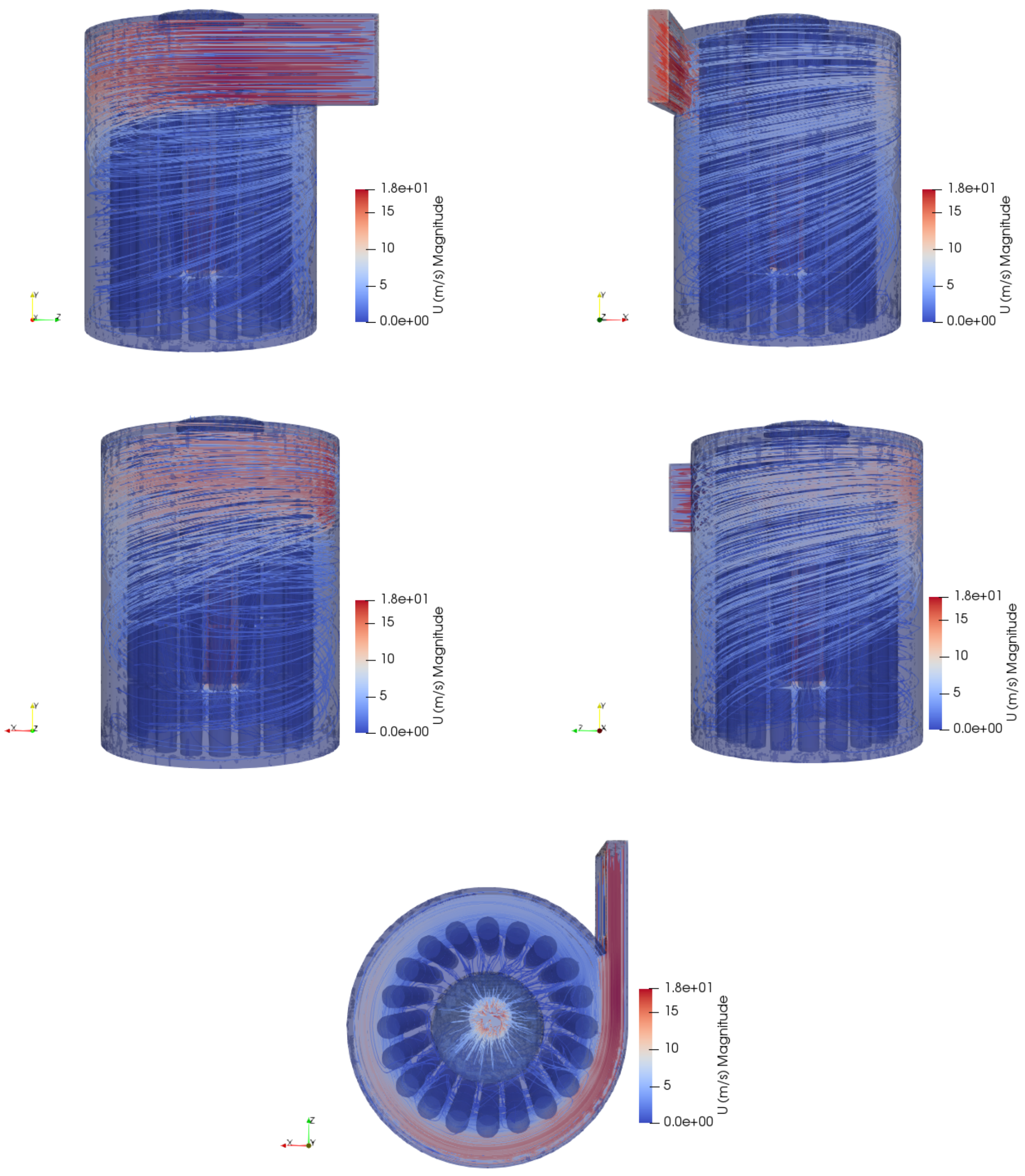


Figure 4.11: Five views of the streamlines within the chamber, clearly displaying the swirling motion

Finally, below are figures of section views normal to the y-plane throughout the domain of the chamber:

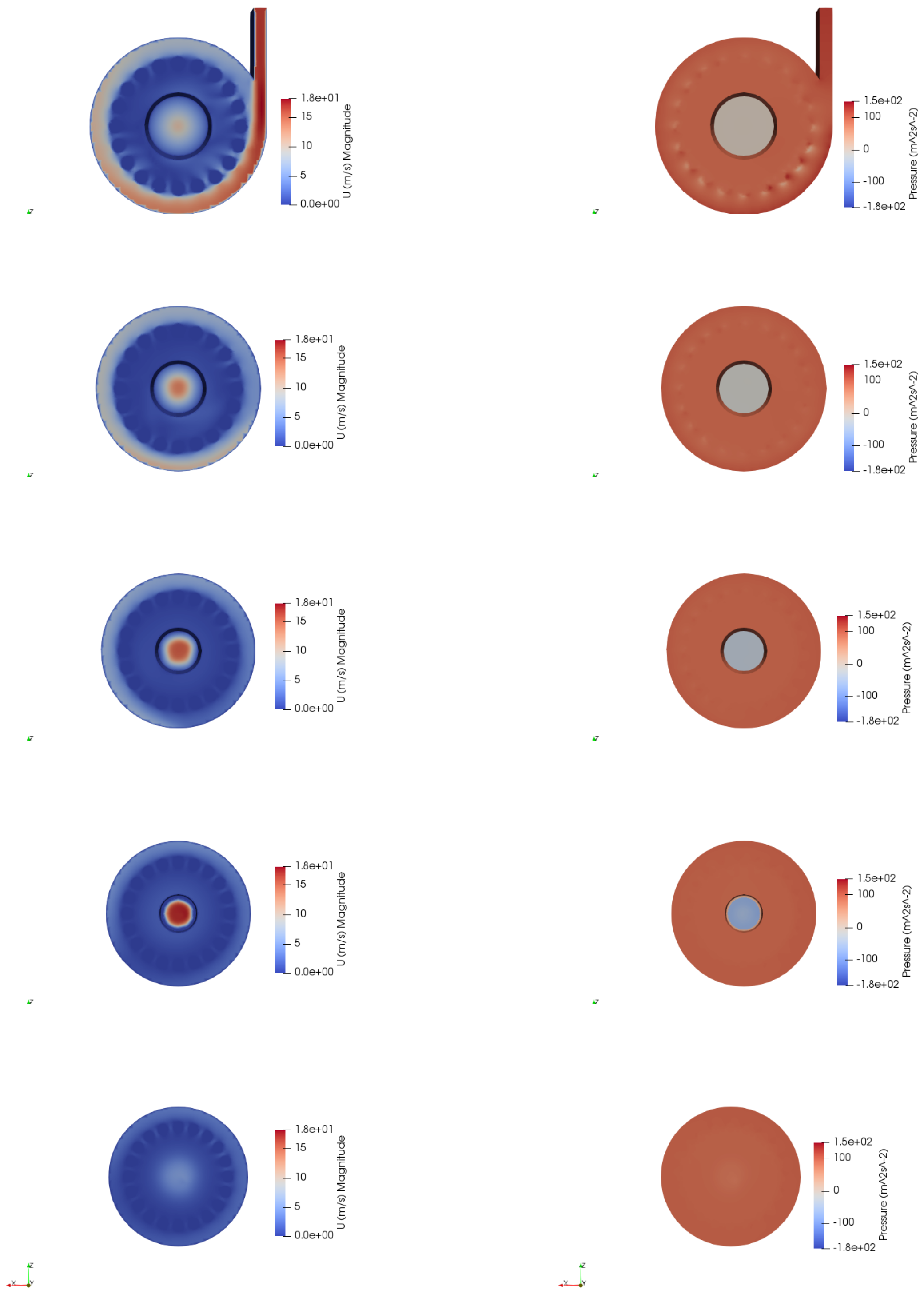


Figure 4.12: Each of the five rows represents a mid-filter section view of velocity and pressure in the chamber.

4.2.3 Functional unit comparison

In this section, the data from the steady-state simulation of the default geometry is studied. As discussed in section 3.2.1, the functional units provide key insight and are the foundation of the optimisation study. Mass flux through the faceZone of the filters provides an insight into the amount of air arriving at the filters. Perfecting the optimisation of uniformity for this measure would mean that all filters in the chamber would receive the same amount of air. This measure can be used to track airflow in the chamber, for example highlighting dead spots in the chamber but is not used to approximate drying.
Velocity of air through the filter cellZone (velocity integrated by volume) offers enough detail for an approximation of drying performance. This functional unit is therefore of a higher importance in the optimisation study.
Figure 4.13 and 4.14 contain data plotting the range flow rates to and through the filters respectively. The filter column numbering is started arbitrarily and the filter rows are plotted from 0 to 19 along the x-axis to show the distribution in the chamber.

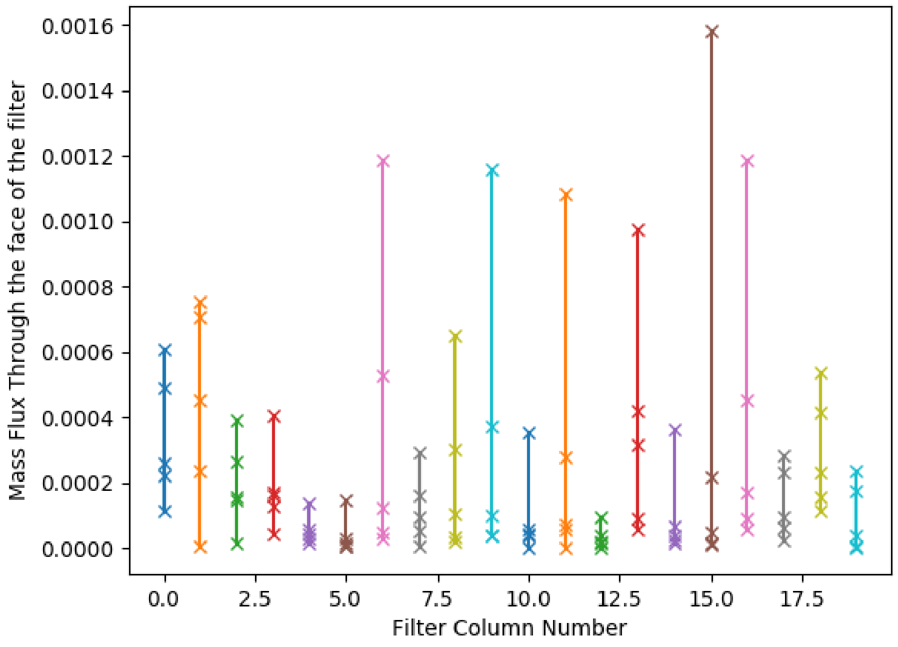


Figure 4.13: The plot for range of mass flux through the filters per row

Analysing figure 4.13 there seems to be a sequence of variation between filter columns 5 and 15 as the largest data points fluctuate a great amount from one column to the next. It is possible in the flow that one filter receives flow and in turn blocks its neighbour from receiving flow. On the other hand, there is far less fluctuation in filters receiving the least mass flux per column. This means that multiple columns through out the domain of the chamber in the default simulation have a large range, showing the lack of uniformity in airflow. The data in this figure suggests columns 4, 5 and 12 are all dead spots where there is little air flow.

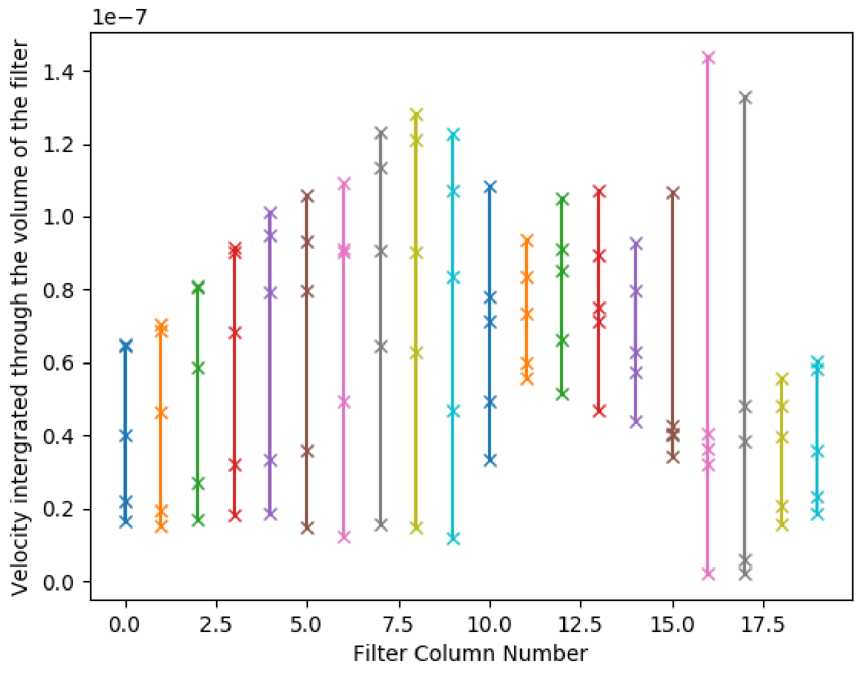


Figure 4.14: The plot for range of velocity through the filters per row

Figure 4.14 can be interpreted to approximate drying performance in the chamber. Unlike figure 4.13, there is far less saw-tooth shaped variation in the data from one column to the next. From column 18 to column 9, the minimum velocity through the filters per column appears to be a consistent non-zero value. The maximum velocity through the filters per column varies to a greater degree and peaks twice in column 8 and column 16. Between these columns, there are five columns that appear to have fairly high drying performance with low ranges. This specific behaviour is ideal for uniform drying performance.
There is an unanswered question when taking figures 4.13 and 4.14 hand in hand: why does the rate of evaporation appear to still be pretty good when in columns 4, 5 and 12 when figure 4.13 suggests there will be dead spots there? This also begs the question, are the rate of velocity integrated by volume, high or not? In order to answer this question, the definition of dead spot must be challenged and the effects of porosity must be considered.
Firstly, relative to the other columns, the mass flux is low in column 4, 5 and 12. Looking at the simulation data, the minimum value for mass flux in each of these cases is $1.7 \times 10^{-5} kgs^{-1}m^{-2}$, $5.3 \times 10^{-6} kgs^{-1}m^{-2}$ and $9.6 \times 10^{-6} kgs^{-1}m^{-2}$ respectively. Therefore the flow rate at these faceZones is low in comparison to the other data, but there is still

flow to travel through the filter. Secondly, only a very small proportion of air entering the chamber travels through the filters in this simulation. Most air goes around the filters and this is shown by the streamline plots in figures 4.11 and in the pressure and velocity section views in figure 4.12.

In order to consider what is happening around the filters in the default simulation, comparison with an exaggerated simulation is useful. Appendix C shows what happens when the filters are placed right in the centre of the stream of flow. As can be seen by figure C.3, areas of high pressure build where the porosity effects from the filters effectively block flow, and areas of low pressure build around the side of the filters, pulling flow around, not through the filter. The effect of this pressure distribution around the filters can be seen in figures C.1 and C.2. Again, there will still be a limited amount of flow that does pass through the filters. The maximum mass flux in C.4 is $0.007 kgs^{-1}m^{-2}$ and the maximum flow through the filter in C.5 is around $4.5 \times 10^{-7} m^4 s^{-1}$, only and increase of 4.4 and 3.2 times respectively, even though the filters are directly in the stream here. Therefore, due to the pressure distribution that builds around the filers in fast moving air, airflow to and through the filter is restricted. At lower velocities and therefore lower flow rates, there is likely more flow to the filters due to the lack of higher pressure regions around the filter.

4.3 Optimisation Study Results

The full results of the one-factor-at-a-time simulation analysis is held in Appendix D. Tables D.1 and D.2 contain all the simulation conditions and results. This data is interpreted in terms of percentage change to uniformity, for each parameter, in tables 4.1 and 4.2 below:

Table 4.1: Percentage change of mass flux through the filters for the default simulation

Simulation:	**1-Min**	**2**	**3-Datum**	**4**	**5-Max**
Outlet Height (mm)	1147.5	1221.25	1295	1368.75	1442.5
	-2.08%	0.20%	Default	1.01%	36.79%
Outlet Diamter (mm)	200	225	250	275	300
	27.55%	-0.52%	Default	0.36%	31.52%
Inlet Length (mm)	300	350	400	450	500
	49.26%	8.08%	Default	-4.94%	27.78%
Filter Pattern Radius (mm)	400	412.5	425	437.5	450
	-22.74%	-42.00%	Default	4.75%	42.00%
Inlet Velocity (m/s)	12.500	14.583	16.667	18.750	20.833
	3.04%	-12.62%	Default	12.60%	72.14%

Table 4.2: Percentage change of Velocity through the volume of the filters for the default simulation

Simulation:	**1-Min**	**2**	**3-Datum**	**4**	**5-Max**
Outlet Height (mm)	1147.5	1221.25	1295	1368.75	1442.5
	-27.06%	-26.19%	Default	14.34%	29.66%
Outlet Diamter (mm)	200	225	250	275	300
	-23.75%	-9.83%	Default	9.90%	28.23%
Inlet Length (mm)	300	350	400	450	500
	-14.58%	-6.68%	Default	8.71%	14.21%
Filter Pattern Radius (mm)	400	412.5	425	437.5	450
	25.49%	-11.54%	Default	13.90%	9.38%
Inlet Velocity(m/s)	12.500	14.583	16.667	18.750	20.833
	-39.20%	-25.29%	Default	26.32%	65.40%

4.4 Discussion of Optimisation Study Results

4.4.1 Overview

In order to maximise uniformity, the percentage difference between the datum and the result should be negative. A positive percentage difference indicates a change that would reduce uniformity. Before the simulations were run, it was expected that varying a parameter higher and lower than a default would result in positive uniformity difference in one direction and negative uniformity in the other direction.
After running the simulations of the optimisation study, it is clear to see that this hypothesis is largely null for the uniformity of the mass flux to the filters. As shown in table 4.1, when the five parameters are varied, the results compared to the datum for the mass flux give no consistent indication as to which direction of optimisation is best, for any of the parameters. However the hypothesis is largely correct for velocity through the volume of the filters.
Table 4.2 suggests that reducing the outlet height, reducing the outlet diameter, reducing the inlet height and reducing the inlet velocity shall lead to improved uniformity of drying in the chamber. The results of the filter pattern radius screening are not conclusive.

4.4.2 Most and Least Sensitive Parameters

Furthermore, according to table 4.2, the parameter whose change (within the set limits) has the largest effect on system uniformity is the Inlet Velocity and the systems least sensitive parameter is Inlet length.

4.4.3 Inlet Length and Uniformity

The one surprising result in this analysis is for the inlet length: it was expected that increasing the inlet length would increase uniformity of drying, as the initial inlet flow velocity would be spread lower down the chamber. This was built on the assumption that the filters at the top of the chamber and therefore nearest the inlet would have the best drying performance. Further analysis of the default simulation reveals that the second row of filters, not the first row of filters, has the highest level of velocity through the filters. This helps to make sense of why the simulation results suggest uniformity of drying would decrease when the inlet length is extended downwards and its cross-sectional area remains constant; a proportion of the air that was reaching the first row of filters is being directed lower in the chamber and is more likely to travel through filters in the second row, therefore decreasing the uniformity. However, if the inlet was extended through a majority of the length of the chamber, one would not expect to see this phenomena repeated and one would expect the uniformity of drying to increase.

4.4.4 Uniformity and Overall Drying Performance

With the current set up, optimising uniformity may be to the detriment of overall evaporation performance. This can most clearly be seen in the system's most sensitive parameter, the Inlet Velocity. The uniformity study suggests the inlet velocity should be reduced but the study does not comment on overall drying performance.

Table 4.3: Additional detail into the maximum and minimum distribution of velocity through the filters in the results of the optimisation study.

Case:	Inlet Velocity (ms^{-1})	Average Row Min $\int u$ dV (m^4s^{-1})	Average Row Max $\int u$ dV (m^4s^{-1})	Average Row Range $\int u$ dV (m^4s^{-1})
Case 9	12.5	1.716-08	4.627e-08	2.911e-08
Case 19	14.58	2.583e-08	6.160e-08	3.577e-08
Default	16.67	3.358e-08	8.146e-08	4.788e-08
Case 20	18.75	4.242e-08	1.029e-07	6.048e-08
Case 10	20.83	4.675e-08	1.259e-07	7.920e-08

The data contained in table 4.3 suggests that as the inlet velocity increases, the minimum average row drying performance and the maximum row drying performance also increases. But as velocity increases the range in row drying performance in the chamber also gets larger, therefore uniformity decreases. So to summarise, reducing the inlet velocity will increase drying uniformity, but slow the rate of drying overall as the minimum average row drying performance will also decrease.

Therefore, with this design used for the default simulation,when adjusting the inlet velocity, there will be a trade-off between uniformity and overall drying performance.

4.5 Additional Design Consideration

As shown, there is a flaw in the design. The simulations show when one wants to increase uniformity, it is to the detriment of the drying performance, according to the velocity integrated with respect to filter volume measure.
Two designs, outside the parameters of the one-factor-at-a-time sensitivity analysis have been briefly considered in this section. One includes an additional outlet at the bottom of the chamber and the other extended the cross-sectional area and length of the inlet to 90% of the total chamber height. Before the simulations were run, it was hypothesised that chamber with the second outlet would bring a region of low or negative pressure to the floor of the chamber, so pulling more air particles lower down the chamber. The thought behind the chamber with the much larger inlet was that it would more evenly distribute airflow between the filters and simultaneously reduce the inlet velocity into the chamber, which was previously seen to have a positive affect on uniformity. Figure 4.15 shows images of the design of chamber with a second outlet and 4.16 shows the concept used for the Large Inlet simulation.

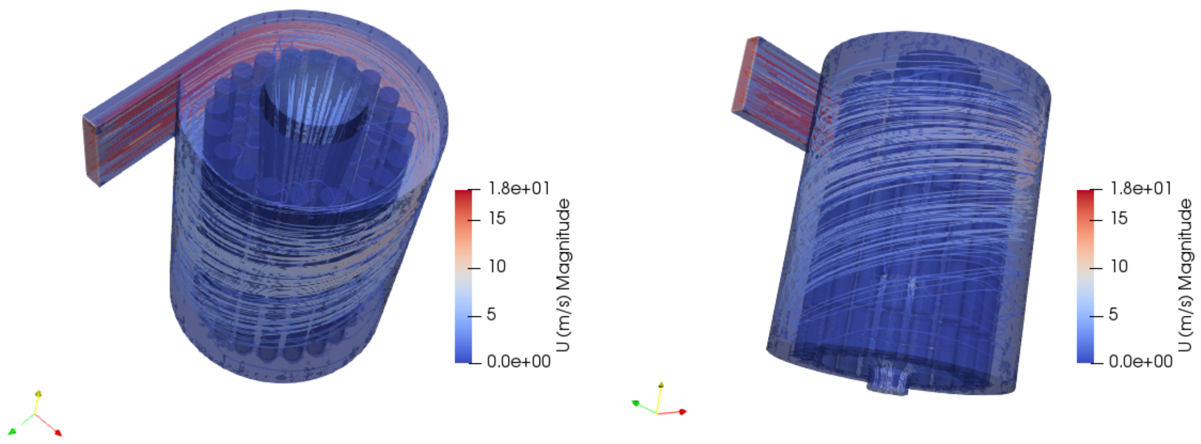


Figure 4.15: An above and below isometric view of the Second Outlet simulation with velocity streamlines.

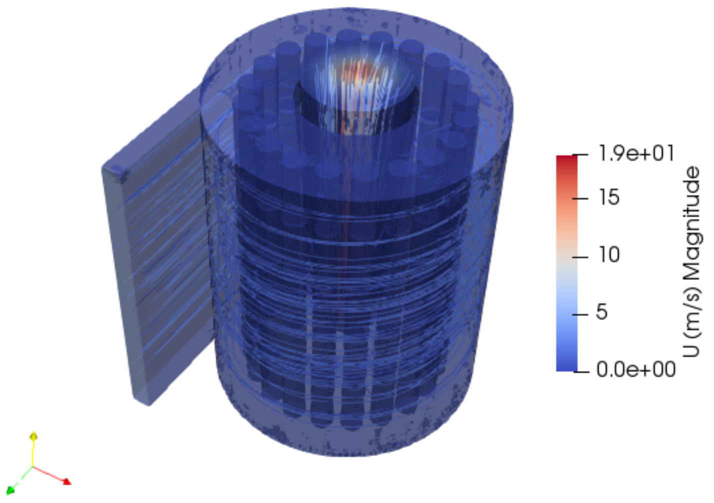


Figure 4.16: An isometric view of the Large Inlet simulation with velocity streamlines.

Once the simulations had been run (using the same bouSimpleFoam solver parameters as the optimisation study), a quick analyse of the results may take place. Table 4.4 compares the results of the row average performance against the default simulation and figure 4.17 compares the column performance.

Table 4.4: Additional detail into the maximum and minimum distribution of velocity through the filters in the results of the optimisation study.

Case:	Inlet Velocity (ms^{-1})	Average Row Min $\int u$ dV $(m^4 s^{-1})$	Average Row Max $\int u$ dV $(m^4 s^{-1})$	Average Row Range $\int u$ dV $(m^4 s^{-1})$
Second Outlet	16.67	3.050e-08	6.704e-08	3.654e-08
Default	16.67	3.358e-08	8.146e-08	4.788e-08
Large Inlet	4.95	7.631e-09	4.604e-08	3.841e-08

Overall, Table 4.4 states that both the Second Outlet and Large inlet have better drying uniformity than the Default, however the default simulation has a higher row average minimum and row average maximum drying performance. What can also be seen in the table is that the Large inlet simulation, comparatively has a very low row average minimum drying performance - 77% lower than Default simulation.

Figure 4.17 compares the column average performance across the 20 columns and also

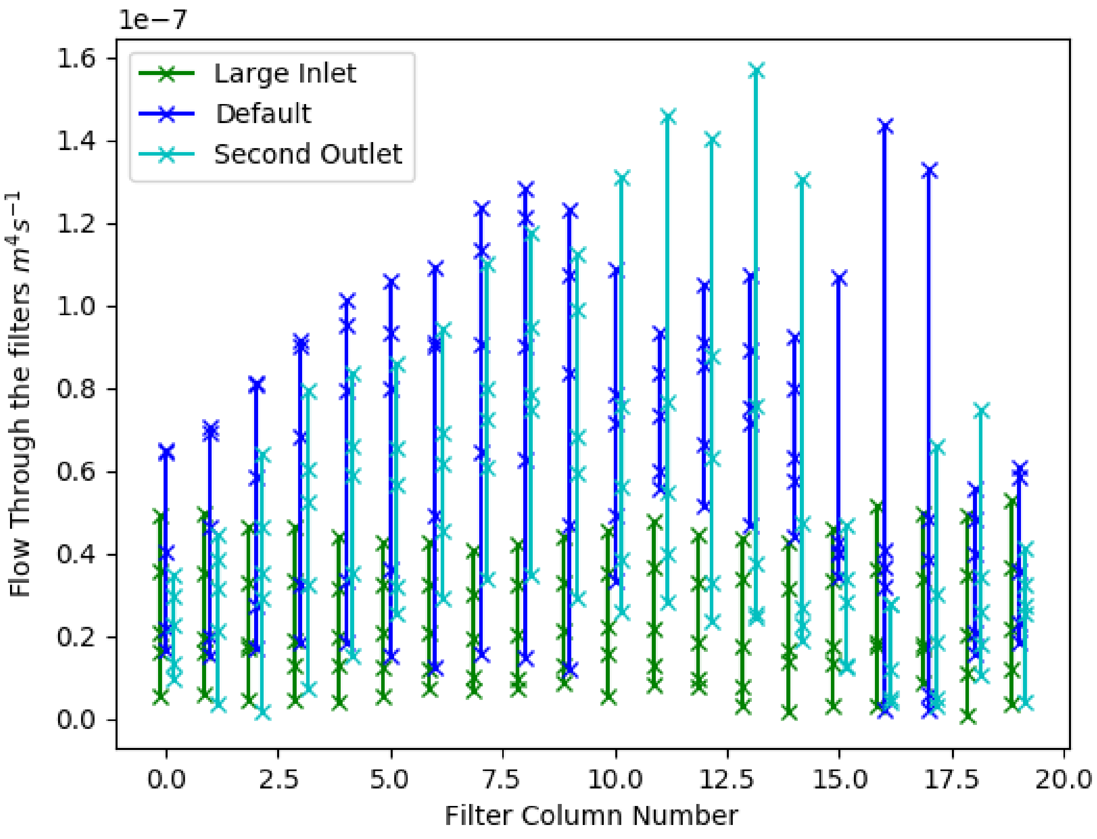


Figure 4.17: Column by column comparison between the three simulations

shows each filter with a marker. Compared to the Default and Second Outlet simulations, figure 4.17 shows the Large Inlet simulation has very consistent column to column performance. It is also interesting to see that though the Default simulation has a higher maximum row average drying performance, the Second Outlet results peak highest.

To conclude, these two simulations are by no means optimised, but they show different design concepts and different ways to improve the uniformity of drying in the chamber, outside of the bounds of the one-factor-at-a-time study undertaken in this report.
It may well be that when parameters for these designs, like the outlet height, outlet diameter(s), inlet length, filter pattern radius and inlet velocity are varied, it may be possible to optimise overall drying performance and drying uniformity, in order to ultimately minimise drying time.

Chapter 5

Future Work

In many ways, the screening technique of one-factor-at-a-time, only scratches the surface of what can be learned in an optimisation study. Of course, one-factor-at-a-time enables vision of the effect a parameter change has on the rest of the system, but it does not comment on how the overall system changes when multiple parameters are changed, and in that case, what parameter is the most dominant. In order to do this, as briefly discussed in section 3.2, global Monte-Carlo techniques may be used such as the Latin Hypercube Sample Method. For a simulation such as this, even with relatively simple optimisation methodology and solver, the simulations required to complete a global optimisation study may run into thousands of [processors] core hours.

Another improvement to this simulation would be to implement a dynamic mesh, in order to model the movement of the rotation of the filters around the chamber, just like the Smart Manufacturing design concept. Instead of taking the mean velocity through the filters in each row, this would give a much more accurate measure of evaporation over time. This may be done using a single reference frame or a multiple reference frame solver in OpenFOAM, such as SRFSimpleFoam.

Another improvement would be to better model the Pall Filters. This can be done through having exact measurements, and through measuring the Darcy-Forchheimer coefficients experimentally rather than theoretically using the Ergun equation. Within the study, it would be of interest to rotate the filters at an angle to the vertical, to see how that affects airflow inside the chamber. Another similar improvement to better the accuracy of the simulation is to model the ducting that the air travels through before it gets into the chamber, instead of relying on an estimate of turbulent intensity, as discussed in section 3.6.3.

The Boussinesq approximation for buoyancy (used to model heat transfer) is convenient to use as it means the non-compressible form of the Navier-Stokes equations can be used. It is a pity that it could not be implemented on simpleFoam in this work, but it was implemented on pisoFoam. A key assumption of the boussinesq approximation for buoyancy is that the fluctuations in density are much less than the value of density. Mathematically $\Delta\rho << \rho_0$ where ρ_0 is the density of the air at the beginning of the simulation.

As the temperature increases the density will decrease, therefore there is a limit on how much the temperature can increase before the Boussinesq approximation for buoyancy is no longer valid. The temperature change simulated in this case is around 40 deg Kelvin, as the room temperature is around 20 deg Celsius and the inlet temperature is around 60 deg Celsius. Therefore there is around a 2% fluctuation in density.

By re-running the Default simulation using a solver that solves for the compressible Navier-Stokes equations, a fair comparison may be made as to whether or not Boussinesq approximation is valid for the range of temperatures seen in this study.

Building a more comprehensive evaporation model, would certainly increase the value of the optimisation study to Smart Manufacturing, but it would undoubtedly increase the complexity of the simulations. One route to getting started running a simulation with an evaporation model is by starting in multi-phase, using Volume of Fluid method and building from there. This has been completed before and documented by Potham where they modelled the evaporation of droplets[30].

The turbulence model used throughout the simulations documented in this report was RNG $k - \epsilon$. As discussed in section 4.1, the turbulence model was primarily chosen for its cost-effectiveness and its satisfactory performance in a swirling free stream. The solver lacks ability to accurately resolve near a wall or in a boundary layer. Literature regarding swirling flows suggested the use of other turbulence models such as $k - \omega - \nu^2$ model[4] or the $R_\epsilon / k - \epsilon$ model[10]. A study into the different turbulence models that are more accurate than the RNG $k - \epsilon$ model and but also cost effective should take place to improve the quality of further optimisation studies.

And finally, it would also be prudent to complete an LES simulation, at least for the Default case. LES inherently has a much higher fidelity than RANS. LES would offer a great insight into the behaviour of the air inside the chamber and around and through the porous filters.

Chapter 6

Conclusion

The purpose of this proof-of-concept project report is to offer Smart Manufacturing an additional way to better comprehend the airflow within a chamber using CFD. The aim is to help make intelligent and analytically-informed design decisions.
The focus of the study was to optimise uniformity of drying within the chamber.
Evaporation is a complex process that is difficult to model computationally within CFD. In this report, drying rate has been simplified to the velocity integrated by the volume of the filters, which is the sum of velocity through the cells of the filters and is proportional to advection, the most dominant component in forced convection[29].
An optimisation study took place using a one-factor-at-a-time screening technique. In Smart Manufacturing's design, the 100 filters are on a bed that is rotating in inside chamber, this was not modelled but the concept was utilised in the optimisation study, as a row average of drying performance was used. The range of drying performance between the five rows was calculated as that value indicated the uniformity of drying within the chamber.
The optimisation study revealed (within the bounds of the study), reducing the inlet velocity had the largest effect to better the uniformity of drying in the chamber. Uniformity increased by 39.2% by reducing the inlet velocity by 25%. The other factors that the study suggested would increase drying performance were reducing the outlet height (raising the outlet) and reducing the outlet diameter. However it was concluded that under the design concept used in the study, uniformity could not be increased without being a detriment of overall drying performance.
Two additional unoptimised design concepts were drawn and tested to show different approaches to offer the same solution. The base level of drying performance and uniformity of the two additional concepts were compared to the performance of the default case.
If Smart Manufacturing wish to optimise the design of future iterations their chamber to provide a product for Pall, it is highly recommended to invest in CFD to inform design decisions. The work done in the study recorded is very limited in comparison to the work that can be achieved by a consultancy. The lengthy list of things that could be changed to better this work include running a global optimisation rather than using a local screening technique for the optimisation study; secondly incorporating an evaporation model, run in

multi-phase, that simulates water molecules being vaporised and wicked away by air flow through the filters and finally, using a dynamic mesh to simulate the movement of the filters on the turn table within the chamber.
Smart Manufacturing may also want to consider multi objective optimisation to both maximise the uniformity of drying and maximise the minimum flow through the filters. This would lead to minimum drying time and maximum energy efficiency.

Bibliography

[1] Weller, H., Tabor, G., Jasak, H., and Fureby, C., “A tensorial approach to computational continuum mechanics using object-oriented techniques,” *Computers in Physics*, Vol. 12, No. 6, 1998, pp. 630–641.

[2] cfd.direct, “Mesh generation with the snappyHexMesh utility,” `https://cfd.direct/openfoam/user-guide/v7-snappyhexmesh/`, 2019, Date Accessed: July 9, 2020.

[3] Wang, B., Xu, D., Chu, K., and AB, Y., “Numerical study of gas-solid flow in a cyclone separator,” *Applied Mathematical Modelling*, Vol. 30, 2006, pp. 1326–1342.

[4] Dhakal, T. and Walters, D., “A Three-Equation Variant of the SST k- ω Model Sensitized to Rotation and Curvature Effects,” *ASME Journal of Fluids Engineering*, Vol. 133, No. 11, 2011, pp. 1–9.

[5] Dhakal, T., Walters, D., and Strasser, W., “Numerical study of gas-cyclone airflow: an investigation of turbulence modelling approaches,” *International Journal of Computational Fluid Dynamics*, Vol. 28, No. 1-2, 2014, pp. 1–15.

[6] Arolla, S. and Durbin, P., ““Modeling Rotation and Curvature Effects Within Scalar Eddy Viscosity Framework,” *International Journal of Heat and Fluid Flow*, Vol. 39, 2013, pp. 78–89.

[7] Stephens, D. and Mohanarangam, K., “Turbulence Model Analysis of Flow Inside a Hydrocyclone,” *Progress in Computational Fluid Dynamics*, Vol. 10, No. 5-6, 2010, pp. 366–373.

[8] Azadi, M., Azadi, M., and Mohebbi, A., “A CFD study of the eddects of cyclone size on its perfomance parameters,” *Journal of Hazardous Materials*, Vol. 182, 2010, pp. 835–841.

[9] Zhao, B. and Su, Y., “Cyclone perfomance depends on multiple factors: comments on ”A CFD study of the eddects of cyclone size on its perfomance parameters” by Mehdi Azadi et al. (2010),” *Journal of Hazardous Materials*, Vol. 303, 2016, pp. 174–176.

[10] Saqr, K., Kassem, H., Aly, H., and Wahid, M., “Computational study of decaying annular vortex flow using the R_ϵ / $k - \epsilon$ turbulence model,” *Applied Mathematical Modelling*, Vol. 36, 2012, pp. 4652–4664.

[11] Boomsma, K.; Poulikakos, D. and Ventikos, Y., "Simulations of flow through open cell metal foams using an idealized periodic cell structure," *International Journal of Heat and Fluid Flow*, Vol. 24, 2003, pp. 825–834.

[12] Della Torre, A., Montenegro, G., Onorati, A., and Tabor, G., "CFD Characterization of Pressure Drop and Heat Transfer Inside Porous Substrates," *Energy Procedia*, Vol. 81, 2015, pp. 836–845.

[13] Hellström, J. G. I. and Lundström, T. S., "Flow through Porous Media at Moderate Reynolds Number," *4th International Scientific Colloquium: Modelling for Material Processing. University of Latvia, Riga, Latvia*, June 8-9 2006.

[14] Bear, J., *Dynamics of Fluids in Porous Media*, Courier Corporation, New York, 1988.

[15] Alabi, O., "Validity of Darcy's Law in Laminar Regime," *TheElectronicJournal ofGeotechnical Engineering*, Vol. 16, 2011, pp. 27–40.

[16] Chapman, R., *Geology and Water - An introduction to fluid mechanics for geologists*, Springer Netherlands, The Hague, 1981.

[17] Hassanizadeh, S. and Gray, W., "High Velocity Flow in Porous Media," *Transport in Porous Media*, Vol. 6, 1987, pp. 521–531.

[18] Sobieski, W. and Trykozko, A., "Darcy's and Forchheimer's laws in practice. Part 1. The Experiment," *Technical Sciences*, Vol. 17, No. 4, 2014, pp. 321–335.

[19] Bear, J., *Modeling Phenomena of Flow and Transport in Porous Media*, Springer International Publishing AG, Cham, Switzerland, 2018.

[20] Lee, K. and Howell, J., "Theoretical and experimental heat and mass transfer in hightly porous media," *International Journal of Heat and Mass Transfer*, Vol. 34, No. 8, 1991, pp. 2123–2132.

[21] Macdonald, F., El-Sayed, M., Mow, K., and Dullien, F., "Flow through Porous Media -the Ergun Equation Revisited," *Industrial & Engineering Chemistry Fundamentals*, Vol. 18, No. 3, 1979, pp. 199–208.

[22] Ergun, S., "Fluid Flow Through Packed Columns," *Chemical Engineering Progress*, Vol. 48, No. 2, 1952, pp. 89–94.

[23] Irmay, S., "On the theoretical derivation of Darcy's and Forchheimer's formulas. Part 1. The Experiment," *Trans Am Geophys Union*, Vol. 39, 1958, pp. 702–707.

[24] Chandrasekhara, B. C. and Vortmeyer, D., "Flow model for velocity distribution in fixed porous beds under isothermal conditions," *Heat and Mass Transfer*, Vol. 12, No. 2, 1979, pp. 105–111.

[25] Jones, W., "The flow of dilute aqueous solutions of macromolecules in various geometries. IV. The Ergun and Jones equations for flow through consolidated beds," *Journal of Physics D: Applied Physics*, Vol. 9, No. 5, 1976, pp. 771–772.

[26] Ferziger, J. and Peric, M., "1.7.5 Boussinesq Approximation," *Computational Methods for Fluid Dynamics*, Springer, 2002, pp. 14–15.

[27] Gray, D. D. and Giorgini, A., "The validity of the boussinesq approximation for liquids and gases." *International Journal of Heat and Mass Transfer*, Vol. 19, No. 5, 1976, pp. 545 – 551.

[28] Bückle, U. and Peric, M., "Numerical simulation of buoyant and thermocapillary convection in a square cavity," *Numerical Heat Transfer: Part A: Applications*, Vol. 21, No. 2, 1991, pp. 121 – 141.

[29] Li, Z. and Heiselberg, P., "CFD Simulation for Water Evaporation and Airflow Movement in Swimming Baths," Tech. rep., Aalborg Universitet Denmark, 2005.

[30] Potham, S. P., *Development of an evaporation sub-model and simulation of multiple droplet impingement in volume of fluid method*, Master's thesis, Michigan Technological University, 2017.

[31] Saltelli, A., "Sensitivity analysis: Could better methods be used?" *Journal of Geographical Research*, Vol. 104, No. 3, 1999, pp. 3789–3793.

[32] Morio, J. and Balesdent, M., "Chapter 7 - Reliability based approaches," *Estimation of Rare Event Probabilities in Complex Aerospace and Other Systems*, Woodhead Publishing, 2016, pp. 87 – 108.

[33] Helton, J., Davis, F., and Johnson, J., "A comparison of uncertainty and sensitivity analysis results obtained with random and Latin hypercube sampling," *Reliability Engineering and System Safety*, Vol. 89, 2005, pp. 305–330.

[34] Aelaei, M., Karimian, S., and Ommi, F., "Sensitivity Analysis and Optimization of Delta Wing Design Parameters using CFD-Based Response Surface Method." *Journal of Applied Fluid Mechanics*, Vol. 12, No. 6, 2019, pp. 1885 – 1903.

[35] Benson, J., Ziehn, T., Dixon, N. S., and Tomlin, A. S., "Global sensitivity analysis of a 3D street canyon model—Part II: Application and physical insight using sensitivity analysis." *Atmospheric Environment*, Vol. 42, No. 8, 2008, pp. 1874 – 1891.

[36] Jasak, H., *Error Analysis and Estimation for the Finite Volume Method with Applications to Fluid Flows*, Ph.D. thesis, Imperial College London, 1996.

[37] Patankar, S., *Numerical heat transfer and fluid flow*, Hemisphere Publish ing Corporation, New York, 1981.

[38] Issa, R., "Solution of the implicitly discretized fluid-flow equations by operator-splitting," *Journal of Computational Physics*, Vol. 62, 1986, pp. 40–65.

[39] Openfoamwiki.net, "The SIMPLE Algorithm In Openfoam," `http://openfoamwiki.net/index.php/OpenFOAM_guide/The_SIMPLE_algorithm_in_OpenFOAM`, 2013, Date Accessed: July 5, 2020.

[40] Openfoamwiki.net, "The PISO Algorithm In Openfoam," `http://openfoamwiki.net/index.php/OpenFOAM_guide/The_PISO_algorithm_in_OpenFOAM`, 2013, Date Accessed: July 5, 2020.

[41] Jones, W. and Launder, B., "The prediction of laminarization with a two equation model of turbulence," *International Journal of Heat and Mass Transfer*, Vol. 15, 1972, pp. 301–314.

[42] Launder, B. and Sharma, B., "Application of the energy-dissipation model of turbulence to the calculation of flow near a spinning disc," *Letters in Heat and Mass Transfer*, Vol. 1, 1974, pp. 131–138.

Appendices

Appendix A

The Mathematics of the PISO and the SIMPLE Algorithms

Starting with the momentum equation (2.2), after the finite volume method has been used to discretize acording to the specifics of the mesh, a matrix $\underline{\mathcal{M}}$ of the coefficients may be derived by decomposing the differential terms. [36] Continuous nomenclature ∇p is used below for simplisity, but this refers to the derivative of the pressure field, and a discretized value will be calculated for every cell in the mesh.

$$\underline{\mathcal{M}} \cdot \underline{u} = -\nabla p \tag{A.1}$$

The diagonal components of matrix $\underline{M}$ are extracted into matrix $\underline{\mathcal{A}}$, where the rest of the cells are 0. The momentum predictor equation in equation A.1 may be re-written as:

$$\underline{\mathcal{M}} \cdot \underline{u} = \underline{\mathcal{A}} \cdot \underline{u} - \underline{\mathcal{H}} \tag{A.2}$$

Where the $\underline{\mathcal{H}}$ matrix contains the residual that is left after extracting the diagonal, so the non-diagonal content of the $\underline{\mathcal{M}}$ matrix, after the $\underline{\mathcal{M}}$ matrix and $\underline{\mathcal{A}}$ matrix have been multiplied by the fluid velocity $\underline{u}$.
Equation A.2 can be rearranged for $\underline{\mathcal{H}}$, giving the final from of the momentum predictor equation:

$$\underline{\mathcal{H}} = \underline{\mathcal{A}} \cdot \underline{u} - \underline{\mathcal{M}} \cdot \underline{u} \tag{A.3}$$

The $\underline{\mathcal{H}}$ matrix is used to calculate the pressure source term. It is known from equation A.1 and equation A.3 that:

$$-\nabla p = \underline{\mathcal{A}} \cdot \underline{u} - \underline{\mathcal{H}} \tag{A.4}$$

This is how pressure is implicitly derived - here pressure is dependant on velocity field $\underline{u}$. However, at this point, it is unknown whether $\underline{u}$ satisfies the continuity equation.
Matrix $\underline{\mathcal{A}}$ is simple and computationally cheap to invert to $\underline{\mathcal{A}}^{-1}$ because it is just a diagonal matrix. Rearranging for $\underline{u}$ and multiplying by $\underline{\mathcal{A}}^{-1}$ gives:

$$\underline{u} = \underline{\mathcal{A}}^{-1} \cdot \underline{\mathcal{H}} - \underline{A}^{-1} \nabla p \tag{A.5}$$

Now there is a equation to calculate velocity field, the continuity equation (equation 2.1) may be considered as a restraint. To incorporate the continuity equation, multiply each term by ∇, recall the continuity equation $\nabla \cdot \underline{u} = 0$. Apply the continuity equation and rearrange:

$$\nabla \cdot (\underline{\mathcal{A}}^{-1} \nabla p) = \nabla \cdot (\underline{\mathcal{A}}^{-1} \underline{\mathcal{H}}) \tag{A.6}$$

In equation A.6, $\underline{\mathcal{A}}^{-1}$ and $\underline{\mathcal{H}}$ are known; the only unknown is the pressure, so pressure may be calculated. Once there is a value for pressure, the algorithm can recalculate $\underline{u}$ using equation A.5. However, when the new velocity is calculated, this invalidates $\underline{\mathcal{M}}$, therefore H needs to be updated before ∇p is recalculated. Therefore the process in full is:

$$\underline{\mathcal{M}} \cdot \underline{u} = -\nabla p \tag{A.7a}$$
$$\underline{\mathcal{H}} = \underline{\mathcal{A}} \cdot \underline{u} - \underline{\mathcal{M}} \cdot \underline{u} \tag{A.7b}$$
$$\nabla \cdot (\underline{\mathcal{A}}^{-1} \nabla p) = \nabla \cdot (\underline{\mathcal{A}}^{-1} \underline{\mathcal{H}}) \tag{A.7c}$$
$$\underline{u} = \underline{\mathcal{A}}^{-1} \cdot \underline{\mathcal{H}} - \underline{\mathcal{A}}^{-1} \nabla p \tag{A.7d}$$

In the first step equation (A.7a) matrix $\underline{\mathcal{M}}$ is created; then in the second step equation (A.7b) the momentum is predicted; in the third step equation (A.7c) a solution for pressure is found and in the final step equation (A.7d) a corrected solution for the velocity field is found. This process is commonly referred to as the Outer Corrector Loop in the algorithms. Both the SIMPLE and PISO use this architecture, albeit with slightly different structure of equation (A.7a) as PISO includes a time dependant term in matrix $\underline{\mathcal{M}}$. Where the algorithms differ, is as follows:
When the SIMPLE algorithm (Patankar 1981)[37] reaches the end of the outer corrector loop, it recalculates matrix $\underline{\mathrm{M}}$ from the momentum predictor equation, so the SIMPLE algorithm restarts and repeats the outer corrector loop. It must be noted that the SIMPLE algorithm also includes a relaxation factor to help convergence.
The PISO algorithm (Issa 1986)[38] first runs the outer corrector loop, then it continues to run an inner loop, consisting of equations (A.7b) to (A.7d). Therefore the inner corrector loop in the PISO algorithm does not recalculate matrix $\underline{\mathrm{M}}$.
Both algorithms end either when specified tolerances for residuals have been met, or after a user-defined maximum number of iterations. [36] [39] [40]

Appendix B

The Mathematics of RANS

The momentum equation for incompressible laminar flow has been shown in equation 2.2. When introducing turbulence, a source term for turbulence needs to be added and the term for the diffusion of viscous forces also will change.

$$\frac{\delta \underline{u}}{\delta t} + \underline{u}(\nabla \cdot \underline{u}) = -\frac{1}{\rho}\nabla p + \nu \nabla^2 \underline{u} + \underline{g} \tag{B.1}$$

becomes

$$\frac{\delta \underline{u}}{\delta t} + \underline{u}(\nabla \cdot \underline{u}) = -\frac{1}{\rho}\nabla p + \nabla \cdot (\mu(\nabla \underline{u} + \nabla \underline{u}^T)) - \nabla(\frac{2}{3}\mu(\nabla \cdot \underline{u})) + \underline{f} + \underline{g} \tag{B.2}$$

where:

$\underline{f}$ - the Reynolds stress $\nabla \cdot (\overline{\rho \underline{u}' \underline{u}'})$

$(\nabla \underline{u} + \nabla \underline{u}^T)$ - mean velocity gradients

The most common approach to calculate Reynolds stress is by using the Boussinesq hypothesis (1877), where the Reynolds stresses are related to the mean velocity gradients.

$$\underline{f} = \mu_t(\nabla \underline{u} + \nabla \underline{u}^T) - \frac{2}{3}\rho k \underline{I} - \frac{2}{3}(\nabla \cdot \underline{u})\underline{I} \tag{B.3}$$

where:

μ_t - the turbulent viscosity

k - the turbulent kinetic energy $[m^2 s^{-2}]$

$\underline{I}$ - Identity matrix size $\underline{u}$

μ_t is unknown, and the RANS models are used to solve for it. RANS models use a transport equation to solve turbulent kinetic energy k and second transport equation to solve turbulent dissipation rate ϵ. Different RANS models have certain advantages and disadvantages. The $k - \epsilon$ model [41] [42]

Transport Equation for turbulent kinetic energy k:

$$\frac{\delta(\rho k)}{\delta t} = \nabla \cdot (\rho D_k \nabla k) + P - \rho\epsilon \tag{B.4}$$

Where:

$\frac{\delta(\rho k)}{\delta t}$ - Rate of change of k over time.

D_k - Effective diffusivity of k

P - Turbulent Energy Production rate $[m^2 s^{-3}]$

There are variants of the $k - \epsilon$ model. The equation for turbulent kinetic energy k is the same for each model, however the transport equation for the turbulent kinetic energy dissipation rate ϵ differs through model coefficients, distinguishing the model variants and give them different characteristics. The transport equation for turbulent kinetic energy dissipation rate ϵ is as follows:

$$\frac{\delta(\rho\epsilon)}{\delta t} = \nabla \cdot (\rho D_\epsilon \nabla \epsilon) + \frac{C_1 \epsilon}{k}(f_1 P + C_3 \frac{2}{3} k \nabla \cdot \underline{u}) - f_2 C_2 \rho \frac{\epsilon^2}{k} \tag{B.5}$$

Where:

D_ϵ - Effective diffusivity of ϵ

C_1 - Model coefficient

C_2 - Model coefficient

C_3 - Model coefficient

f_1 - Damping function coefficient

f_2 - Damping function coefficient

Once k and ϵ have been calculated, the turbulent viscosity μ_t (that appears in equation B.3) may be calculated using the equation:

$$\mu_t = f_\mu C_\mu \frac{\rho k^2}{\epsilon} \tag{B.6}$$

Where:

C_μ - Model coefficient

f_μ - Damping function coefficient

Damping Functions

In order to calculate dissipation rate ϵ and turbulent viscosity $\mu_(t)$, damping functions must be considered.
As previously mentioned, approaching a non-slip wall the length scale of turbulent eddies will reduce as the viscous forces in the flow start to dominate turbulent forces, in the viscous sub-layer turbulent kinetic energy k will be small and dissipation rate ϵ will be high. The turbulent Reynolds number (Re_T) characterises the strength of turbulence near a wall and is defined mathematically by the following equation:

$$Re_T = \frac{\rho k^2}{\mu \epsilon} \tag{B.7}$$

There is a value for Re_T in every cell in the mesh. Near a wall it is expected that Re_T is small, therefore viscosity dominates.
Damping functions are driven by the value of Re_T. Below are the example of damping functions for the $k - \epsilon$ turbulence model.[42]

$$f_1 = 1 \tag{B.8a}$$
$$f_2 = 1 - 0.3\exp(-Re_T^2) \tag{B.8b}$$
$$f_\mu = \exp\left(\frac{-3.4}{\left(1 + (\frac{Re_T}{50})\right)^2}\right) \tag{B.8c}$$

It can be seen for cells near the wall, when Re_T tends to 0, the damping functions will tend to 1. Whereas in the free turbulent stream, where Re_T is large, damping coefficients f_2 and f_μ will tend to 0, therefore minimising the value of turbulent viscosity μ_t and reducing turbulent dissipation rate ϵ.
After the damping functions and turbulent viscosity have been found, the transport equations for k and ϵ can be solved then finally the solutions to the Naiver-Stokes equations may be found, which include transport equations for turbulence.

Appendix C

Filter Directly in the Inlet Stream

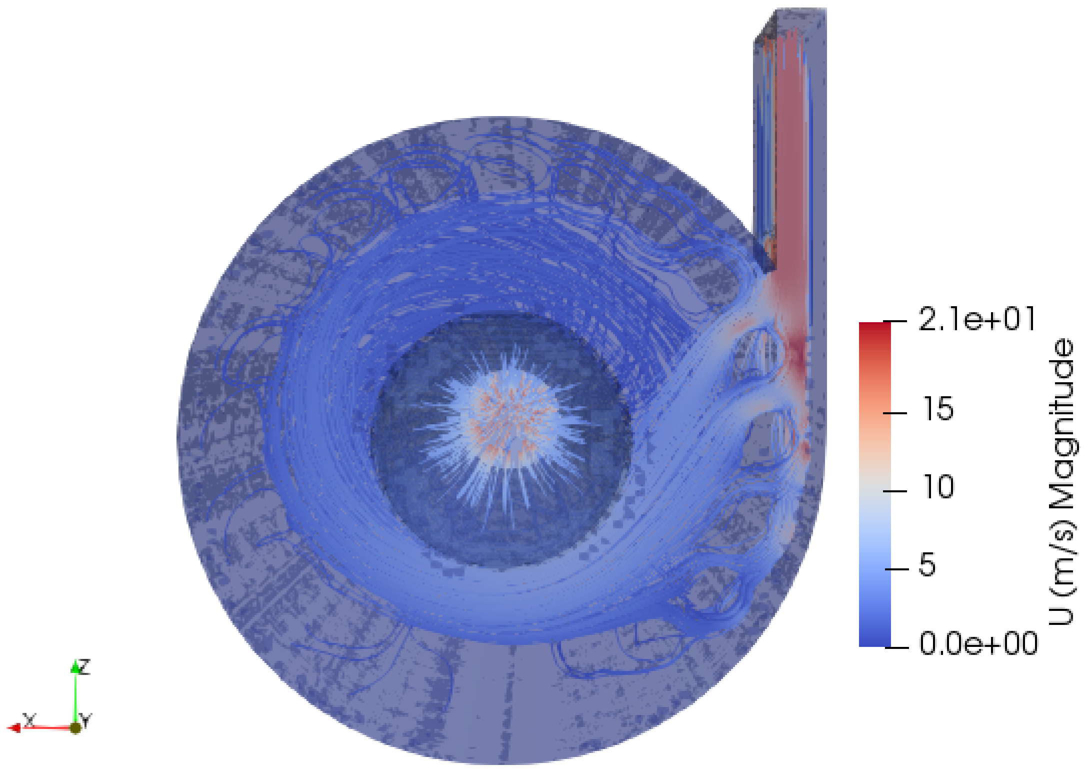


Figure C.1: Velocity streamlines around the filters in the way of the inlet

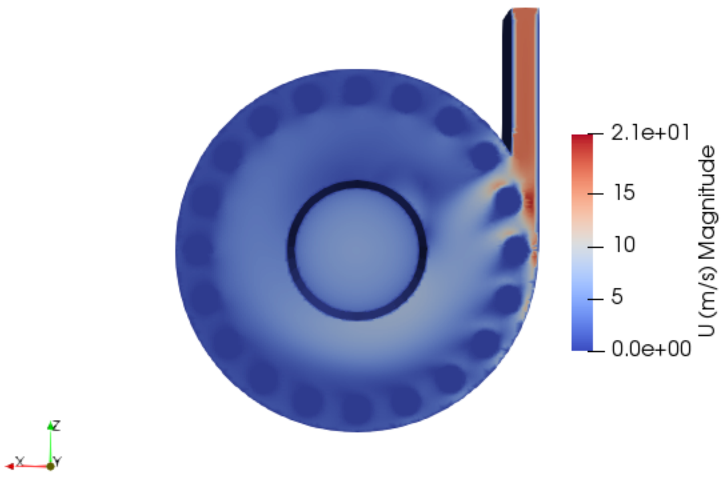


Figure C.2: The velocity distribution mid-plane of the top filters

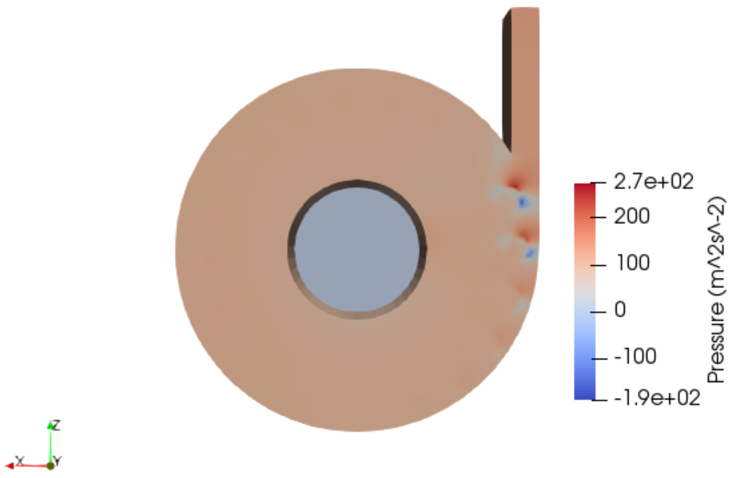


Figure C.3: The pressure distribution mid-plane of the top filters

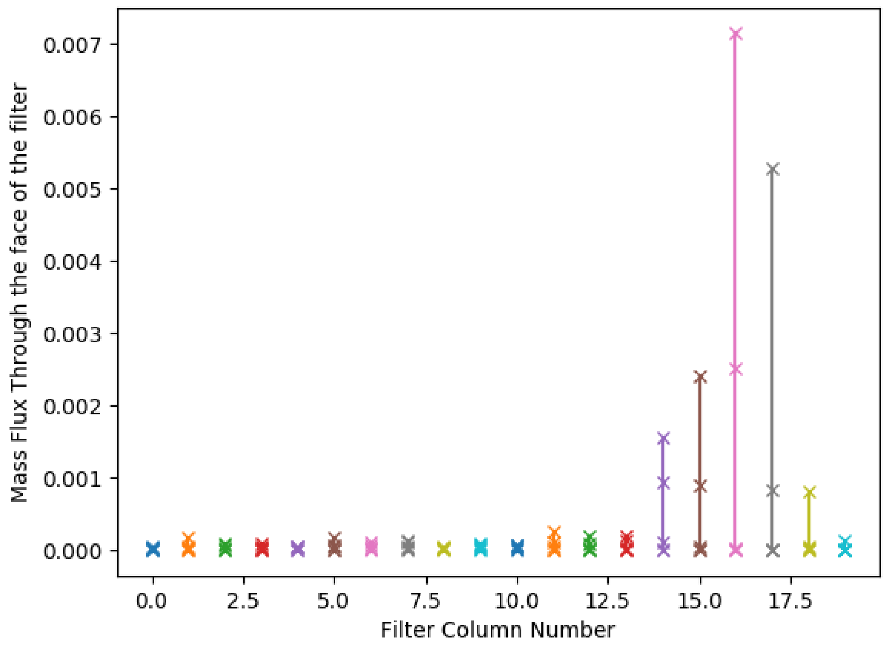


Figure C.4: mass flux to the filters across the columns

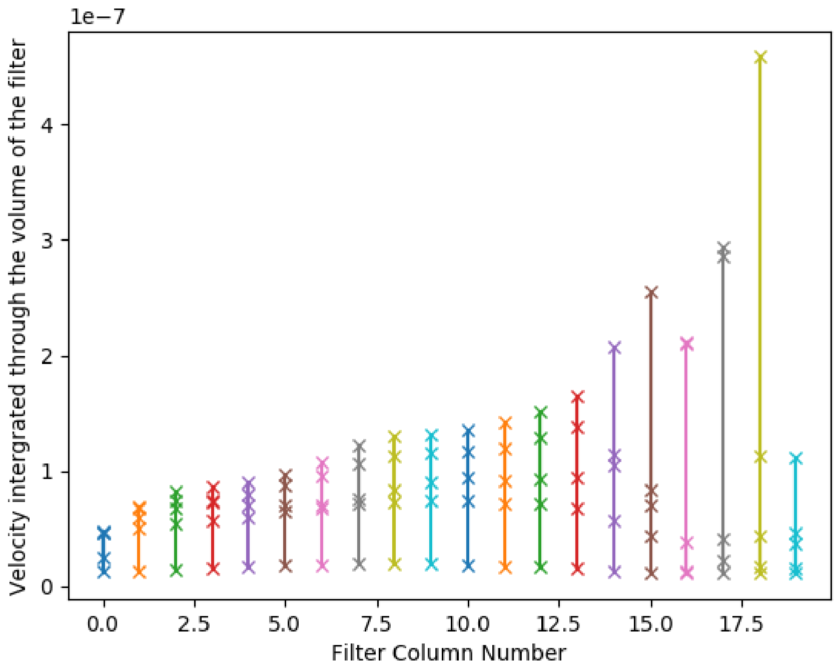
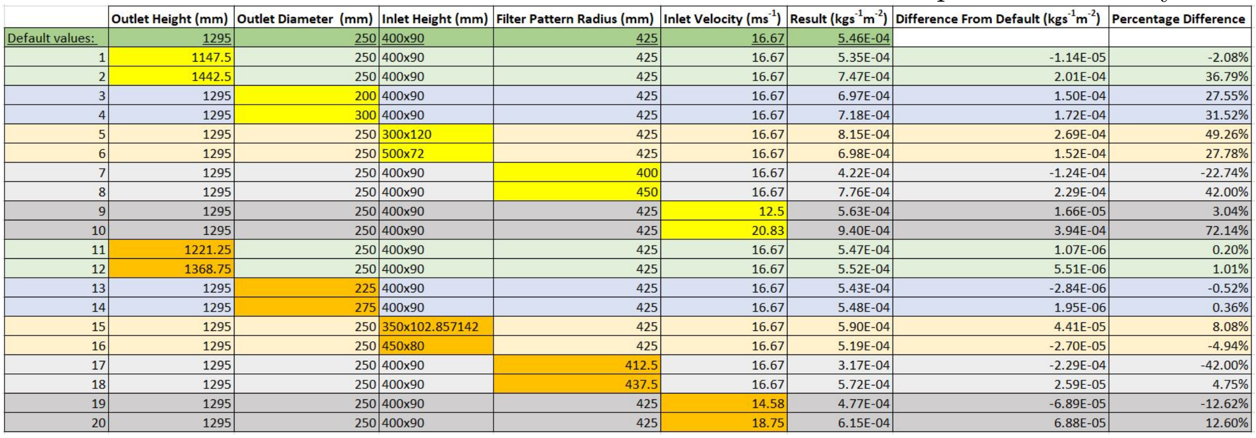


Figure C.5: Velocity through the filter across the columns

Appendix D

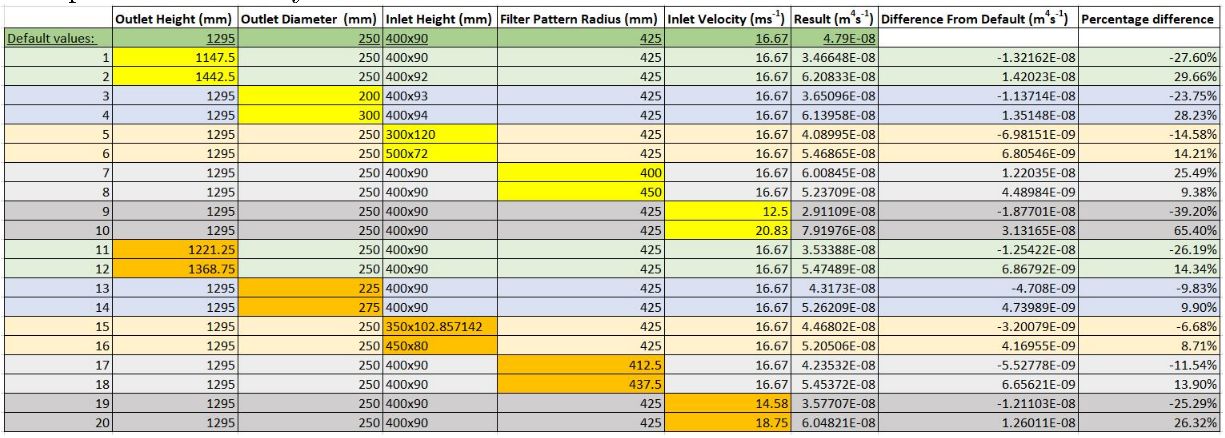
Full Optimisation Simulation Data Set

Table D.1: The full data set for one factor at a time the mass flux optimisation study



	Outlet Height (mm)	Outlet Diameter (mm)	Inlet Height (mm)	Filter Pattern Radius (mm)	Inlet Velocity (ms^{-1})	Result ($kgs^{-1}m^{-2}$)	Difference From Default ($kgs^{-1}m^{-2}$)	Percentage Difference
Default values:	1295	250	400x90	425	16.67	5.46E-04		
1	1147.5	250	400x90	425	16.67	5.35E-04	-1.14E-05	-2.08%
2	1442.5	250	400x90	425	16.67	7.47E-04	2.01E-04	36.79%
3	1295	200	400x90	425	16.67	6.97E-04	1.50E-04	27.55%
4	1295	300	400x90	425	16.67	7.18E-04	1.72E-04	31.52%
5	1295	250	300x120	425	16.67	8.15E-04	2.69E-04	49.26%
6	1295	250	500x72	425	16.67	6.98E-04	1.52E-04	27.78%
7	1295	250	400x90	400	16.67	4.22E-04	-1.24E-04	-22.74%
8	1295	250	400x90	450	16.67	7.76E-04	2.29E-04	42.00%
9	1295	250	400x90	425	12.5	5.63E-04	1.66E-05	3.04%
10	1295	250	400x90	425	20.83	9.40E-04	3.94E-04	72.14%
11	1221.25	250	400x90	425	16.67	5.47E-04	1.07E-06	0.20%
12	1368.75	250	400x90	425	16.67	5.52E-04	5.51E-06	1.01%
13	1295	225	400x90	425	16.67	5.43E-04	-2.84E-06	-0.52%
14	1295	275	400x90	425	16.67	5.48E-04	1.95E-06	0.36%
15	1295	250	350x102.857142	425	16.67	5.90E-04	4.41E-05	8.08%
16	1295	250	450x80	425	16.67	5.19E-04	-2.70E-05	-4.94%
17	1295	250	400x90	412.5	16.67	3.17E-04	-2.29E-04	-42.00%
18	1295	250	400x90	437.5	16.67	5.72E-04	2.59E-05	4.75%
19	1295	250	400x90	425	14.58	4.77E-04	-6.89E-05	-12.62%
20	1295	250	400x90	425	18.75	6.15E-04	6.88E-05	12.60%

Table D.2: The full data set for one factor at a time the velocity through the volume of the filters optimisation study



	Outlet Height (mm)	Outlet Diameter (mm)	Inlet Height (mm)	Filter Pattern Radius (mm)	Inlet Velocity (ms^{-1})	Result (m^4s^{-1})	Difference From Default (m^4s^{-1})	Percentage difference
Default values:	1295	250	400x90	425	16.67	4.79E-08		
1	1147.5	250	400x90	425	16.67	3.46648E-08	-1.32162E-08	-27.60%
2	1442.5	250	400x92	425	16.67	6.20833E-08	1.42023E-08	29.66%
3	1295	200	400x93	425	16.67	3.65096E-08	-1.13714E-08	-23.75%
4	1295	300	400x94	425	16.67	6.13958E-08	1.35148E-08	28.23%
5	1295	250	300x120	425	16.67	4.08995E-08	-6.98151E-09	-14.58%
6	1295	250	500x72	425	16.67	5.46865E-08	6.80546E-09	14.21%
7	1295	250	400x90	400	16.67	6.00845E-08	1.22035E-08	25.49%
8	1295	250	400x90	450	16.67	5.23709E-08	4.48984E-09	9.38%
9	1295	250	400x90	425	12.5	2.91109E-08	-1.87701E-08	-39.20%
10	1295	250	400x90	425	20.83	7.91976E-08	3.13165E-08	65.40%
11	1221.25	250	400x90	425	16.67	3.53388E-08	-1.25422E-08	-26.19%
12	1368.75	250	400x90	425	16.67	5.47489E-08	6.86792E-09	14.34%
13	1295	225	400x90	425	16.67	4.3173E-08	-4.708E-09	-9.83%
14	1295	275	400x90	425	16.67	5.26209E-08	4.73989E-09	9.90%
15	1295	250	350x102.857142	425	16.67	4.46802E-08	-3.20079E-09	-6.68%
16	1295	250	450x80	425	16.67	5.20506E-08	4.16955E-09	8.71%
17	1295	250	400x90	412.5	16.67	4.23532E-08	-5.52778E-09	-11.54%
18	1295	250	400x90	437.5	16.67	5.45372E-08	6.65621E-09	13.90%
19	1295	250	400x90	425	14.58	3.57707E-08	-1.21103E-08	-25.29%
20	1295	250	400x90	425	18.75	6.04821E-08	1.26011E-08	26.32%