Optimization of Server Room HVAC Systems for Energy Efficiency

Leveraging CFD and AI-Driven Techniques

Master's thesis in Engineering Mathematics and Computational Sciences

John Carlsson

DEPARTMENT OF SOME SUBJECT OR TECHNOLOGY

CHALMERS UNIVERSITY OF MECHANICS AND MARITIME SCIENCES
Gothenburg, Sweden 2024
www.chalmers.se

Master's thesis 2024

Optimization of Server Room HVAC Systems for Energy Efficiency

Leveraging CFD and AI-Driven Techniques

John Carlsson

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mechanics and Maritime Sciences
Division of Fluid Dynamics
Chalmers University of Technology
Gothenburg, Sweden 2024

Optimization of Server Room HVAC Systems for Energy Efficiency
Leveraging CFD and AI-Driven Techniques
John Carlsson

Supervisor: Ola Löseth, Actemium Energy AI
Examiner: Håkan Nilsson, Chalmers

Master's Thesis 2024
Department of Mechanics and Maritime Sciences
Division of Fluid Dynamics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2024

Optimization of Server Room HVAC Systems for Energy Efficiency
Leveraging CFD and AI-Driven Techniques
John Carlsson
Department of Mechanics and Maritime Sciences
Chalmers University of Technology

Abstract

The increasing demand for energy efficiency in server rooms and Heating, Ventilation & Air Conditioning (HVAC) systems necessitates advanced optimization techniques to reduce energy consumption while maintaining thermal stability. This thesis focuses on holistic energy optimization of a server room's HVAC system by optimizing the room's geometry and HVAC parameters using computational methods such as CFD and machine learning models, including an Artificial Neural Network (ANN). The study employs a range of meta heuristic optimization algorithms, including Genetic Algorithm (GA), Simulated Annealing (SA), Particle Swarm Optimization (PSO), and Differential Evolution (DE), to identify the best configuration for reducing energy usage.

The goal of the optimization process is to maximize the inflow air temperature and minimize the mass flow while ensuring that the average room temperature does not exceed 28°C. The project was conducted at a worst case scenario, when the outside air temperature is 30 degrees Celsius and the server room is running on maximum capacity, generating waste heat of $72kW$.

A Latin Hypercube Sampling (LHS) method was employed to capture the underlying differential equations behavior throughout the high-dimensional parameter space, and a neural network was trained on this sample data to predict room temperature. Particle Swarm Optimization was used in order to find the optimal parameters for minimal energy consumption.

The results demonstrate that the proposed optimization techniques can significantly enhance the energy efficiency of HVAC systems in server rooms at a worse case scenario. By adjusting the inflow temperature and air mass flow, a reduction in cooling energy consumption of up to 22.69% was achieved. Future work could include hybrid optimization approaches to further improve system performance and the application of multi-objective optimization to accommodate varying operational phases.

Keywords: CFD, Optimization, Machine learning, Neural network, energy efficiency.

Acknowledgements

I would like to express my sincere gratitude to the Energy AI team at Actemium. Special thanks go to Leon Löwered for taking me in and believing in my potential, and to Ola Löseth for providing outstanding support in his role as supervisor. I am also incredibly grateful to the rest of the team for their support, feedback, and for making me feel included in their daily operations throughout this project.

Finally, I would like to thank my examiner, Håkan Nilsson, for his interest, understanding, and continuous encouragement during this project.

John Carlsson, Gothenburg, December 2024

Nomenclature

Below is the nomenclature of operators, indices, parameters, and variables that have been used throughout this thesis.

Operators

δ_{ij}	Kronecker delta, 1 if $i = j$ else 0
Δ	Difference operator

Indices

i, j, k	Indices
t	Index for time step

Parameters

C_p	Specific heat [J/(kg $\cdot$ K)]
d	Darcy coefficient
f	Forchheimer coefficient
h	Convective heat transfer coefficient [W/(m^2 $\cdot$ K)]
k	Thermal conductivity [W/(m $\cdot$ K)]
κ	Idelchik's pressure coefficient
μ	Dynamic viscosity [Ns/m^2]
ν	Kinematic viscosity [Ns/m^2]
ν_t	Kinematic turbulent viscosity [Ns/m^2]
P_i	Porous inertial resistance
P_v	Porous viscous resistance
ρ	Density [kg/m^3]

ζ_0	resistance coefficient of the obstruction
ζ_φ	Resistance coefficient of the obstruction
$\varepsilon_{0,Re}$	Coefficient
$\bar{f}$	The area allow for flow to pass through

Variables

A	Area [m^2]
A_p	Area allowing flow [m^2]
ΔE	Difference in objective function value
$\dot{m}$	Mass flow [kg/s]
e	Internal energy [J/kg]
γ	Specific gravity of the flowing medium
L	Length [m]
p	Pressure [Pa]
P_{accept}	Probability of acceptance
Q	Heat transfer [W]
$\mathbf{q}$	Heat flux vector [W/m^2]
T	Temperature [K]
T_∞	Temperature far from the surface [K]
T_s	Surface temperature [K]
$\mathbf{u}$	Velocity vector, (u, v, w) [m/s]
$\mathbf{x}$	Spatial vector, (x, y, z) [m]
w_1	Velocity at the obstruction [m/s]

Acronyms

HVAC	Heating, Ventilation & Air Conditioning
CFD	Computational Fluid Dynamics
RANS equations	Reynolds Average Navier Stokes equations
DNS	Direct Numerical Simulation
LES	Large Eddy Simulation
ANN	Artificial Neural Network
GA	Genetic Algorithm

PSO	Particle Swarm Optimization
SA	Simulated Annealing
DE	Differential Evolution
MSE	Mean Square Error
LEED	Leadership in Energy and Environmental Design
BREEAM	Building Research Establishment Environmental Assessment Method
ASHRAE	American Society of Heating, Refrigerating and Air-Conditioning Engineers
BEM	Building Energy Modeling
LHS	Latin Hypercube Sampling

Glossary

STAR-CCM+	A CFD software used to simulate fluid dynamics under real world conditions
Thermodynamics	A branch of physics that deals with heat, work, and temperature, and their relation to energy, entropy, and the physical properties of matter and radiation

Contents

1
Introduction

This chapter presents the history of energy optimization for industrial buildings, current solutions, and future endeavors to further increase efficiency.

1.1 Background

In recent years, the growing emphasis on energy efficiency and sustainability has led to the exploration of optimization techniques across various industries, especially in Heating, Ventilation & Air Conditioning (HVAC) systems and building design. With rising energy costs and increasing awareness of climate change, industries have turned to computational methods, such as Computational Fluid Dynamics (CFD) and Building Energy Modeling (BEM), to reduce energy consumption in both new constructions and retrofits. This shift is driven by regulatory standards like American Society of Heating Refrigerating and Air-Conditioning Engineers (ASHRAE), which encourage more energy-conscious designs.

Optimization techniques in energy systems range from material improvements to smart technologies that monitor and adjust energy use in real-time. Advances in artificial intelligence and machine learning have revolutionized this field, allowing for more dynamic and precise control over building systems. These techniques facilitate better control of airflow, temperature, and energy usage by optimizing HVAC designs and building geometries. Furthermore, older buildings, which are often inefficient due to outdated materials and poor insulation, can greatly benefit from these internal optimization methods.

1.1.1 Societal

In Sweden, in 2021, 39% of end-use energy consumption came from the housing and service sector. 90%, or 132 TWh, can be accredited to housing and non-residential buildings. The total use of energy for heating and hot water in small houses, apartment buildings, and non-residential buildings has decreased since 1983. This has occurred at the same time as the population has increased. The effect has been that while heating and hot water from biofuel, electric, and gas sources have remained the same, a small increase in district heating and a large decrease in oil usage have taken place. Thus, the decrease in energy consumption can be attributed to technological advancements and more efficient engineering techniques. As the Swedish Energy Agency puts it - "The energy required for heating and hot water in a building

depends to a large extent on the construction of the house and its shape, insulation, windows, ventilation, technical solutions, etc. The mean energy consumption per square meter for heating and hot water, therefore, often varies depending on the year of construction."[1]

Retrofitting these structures to improve energy performance can be challenging, as optimizing materials and insulation often requires significant reconstruction. This makes it more feasible to focus on internal optimizations, such as improving airflow patterns, adjusting HVAC configurations, and changing the placement of components within the room. These changes can reduce energy consumption without the need for costly structural overhauls, making them a more practical solution for enhancing energy efficiency in older buildings.

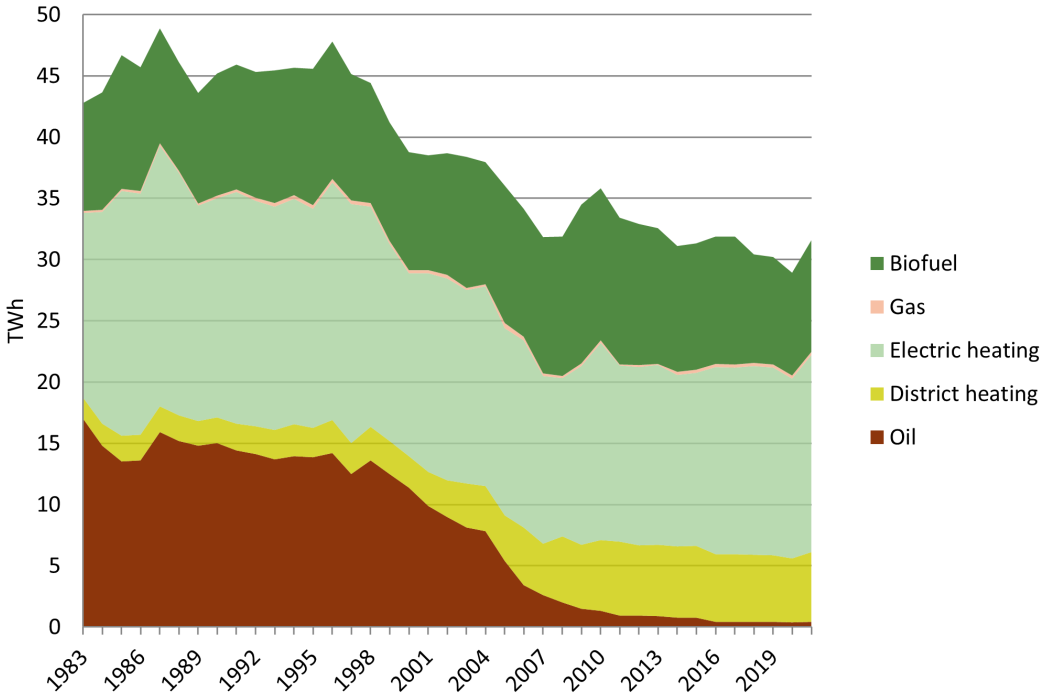

Figure 1.1: Energy consumption from heating and hot water for small houses, apartment buildings and non-residential buildings from 1983, TWh [1].

1.1.2 Technical

HVAC systems are critical for maintaining controlled environments, particularly in industrial and commercial settings. Historically, industrial buildings have been optimized through architectural and mechanical solutions to reduce energy consumption. In the early days, improvements largely relied on manual design changes and basic Thermodynamics principles, such as improving insulation capacity and managing airflow. However, as technology progressed, more sophisticated methods, such as CFD, were introduced to model airflow and heat transfer with greater accuracy.

In recent decades, optimization of HVAC systems has evolved to include algorithmic approaches like Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and Simulated Annealing (SA). These methods allow for automated and dynamic solutions that optimize variables such as airflow, mass flow, and energy usage in real-time. Today, the focus is on applying machine learning techniques, such as Artificial Neural Network (ANN), to further improve the energy efficiency of industrial buildings while lowering costs. Neural networks can analyze vast amounts of data generated by simulations, allowing for more precise optimization than ever before.

The complexity of HVAC systems in industrial buildings necessitates a holistic approach to energy optimization. This involves fine-tuning individual subsystems, thermodynamic processes, filtration mechanisms, heaters, coolers, and steam humidifiers. Furthermore, optimization efforts extend to automatic control devices, ensuring seamless integration and efficiency across the entire HVAC system. Given the nature of these systems operating in a changing environment, achieving energy efficiency in HVAC operations is difficult but necessary for reducing energy consumption.

1.2 Previous work

In the field of building optimization, several approaches have been proposed to enhance energy efficiency. A notable example is the research conducted by Long et al. (2023), where an AI-driven model was developed to predict and optimize energy-efficient building envelopes. This study focused on material optimization, integrating machine learning algorithms, such as Gradient Boosting and ANN, with optimization techniques to improve both cost and energy savings by up to 21.17% and 8.48%, respectively.[2]

While optimizing the building envelope is critical, geometry optimization within the building for individual rooms presents the next logical step in energy reduction efforts. By focusing on the layout of the geometry and positioning of HVAC components, as in this thesis, we can target further energy consumption improvements. Thus, combining material and geometry optimization offers a more comprehensive approach to energy-efficient building design.

1.3 Aim & Scope

This thesis focuses on optimizing the geometry of a room within a HVAC system to reduce energy consumption. Using advanced simulation software, such as STAR-CCM+, and training ANNs on sample data, with a comprehensive understanding of CFD and Thermodynamics principles, this project aims to demonstrate how modifying room components and applying modern machine learning techniques can lead to significant energy savings. The optimization is carried out by modifying room components, such as ventilation inlets and outlet placements, while respecting stake-

holder constraints and thermal requirements. The project is confined to a worst case scenario, when the outside temperature is 30 degrees Celsius and heat generation inside the room is $72kW$.

The scope of this project includes:

- Developing and running CFD simulations in STAR-CCM+ to model airflow and temperature gradients within a server room.
- Training an ANN to predict the average room temperature of different geometries.
- Applying optimization algorithms to identify the best geometry for energy efficiency at a worst case scenario.
- Evaluating the energy savings achieved through the optimal geometry.

2
Theory

In the following sections, the theory regarding CFD and Thermodynamics are presented, as well as the optimization algorithms used in this thesis along with a basic background of neural networks.

2.1 Computational fluid dynamics

Computational Fluid Dynamics (CFD) is when a computer is used to solve the governing equations for fluid problems. It is a powerful tool commonly used in industrial application and academia. Continuous development of CFD has been done since its emergence in the 20th century.

2.1.1 Navier-Stokes equations

Fluid motion and its properties are governed by the Navier-Stokes equations.

$$\frac{\partial v_i}{\partial t} + \frac{\partial v_i v_j}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu \frac{\partial^2 v_i}{\partial x_j \partial x_j} \tag{2.1}$$

The continuity equation, describes the conservation of mass in a system.

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho v_i)}{\partial x_i} = 0 \tag{2.2}$$

The momentum equation, describes the conservation of momentum within the system.

$$\frac{\partial (\rho v_j)}{\partial t} + \frac{\partial (\rho v_i v_j)}{\partial x_i} = -\frac{\partial p}{\partial x_j} + \rho g_j + \frac{\partial}{\partial x_i}\left[\mu\left(\frac{\partial v_j}{\partial x_i} + \frac{\partial v_i}{\partial x_j}\right) - \frac{2}{3}\mu \frac{\partial v_k}{\partial x_k}\delta_{ij}\right] \tag{2.3}$$

The energy equation describes the conservation of energy in the system. Where the stress tensor τ is described by Equation 2.5.

$$\frac{\partial}{\partial t}\left(\rho\left[e + \frac{1}{2}v_j^2\right]\right) + \frac{\partial}{\partial x_i}\left(\rho\left[e + \frac{1}{2}v_j^2\right]v_i\right) = \rho g_i v_i - p\frac{\partial v_j}{\partial x_j} + \tau_{ij}\frac{\partial v_j}{\partial x_i} + v_j\left(-\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_i}\right) - \frac{\partial q_i}{\partial x_i} \tag{2.4}$$

$$\tau_{ij} = \mu\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i}\right) - \frac{2}{3}\mu\frac{\partial v_k}{\partial x_k}\delta_{ij} \tag{2.5}$$

These equations enable the prediction of fluid behavior.

2.1.2 Turbulence modelling

The Reynolds Average Navier Stokes equations (RANS equations) are the time averaged equations of motion for fluids. They are based on the assumption that all velocity variables can be separated into one mean and one fluctuating component.

$$u(\mathbf{x}, t) = \bar{u}(\mathbf{x}) + u'(\mathbf{x}, t) \tag{2.6}$$

A particular property of this is that the average value of the fluctuating component is equal to zero.

$$\bar{u'} = 0 \tag{2.7}$$

Inserting this into the Navier-Stokes equations, the equation can be written in the following way.

$$\frac{\partial \langle v_i \rangle \langle v_j \rangle}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \langle p \rangle}{\partial x_i} + \nu \frac{\partial^2 \langle v_i \rangle}{\partial x_j \partial x_j} - \frac{\partial \langle v_i' v_j' \rangle}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \langle p \rangle}{\partial x_i} + \frac{\partial}{\partial x_j}\left[(\nu + \nu_t)\frac{\partial \langle v_i \rangle}{\partial x_j}\right] \tag{2.8}$$

Here the term $\frac{\partial \langle v_i' v_j' \rangle}{\partial x_j}$ is called the Reynolds stress, it is unknown and must be modeled. For example, here it is done by a turbulence model in which a new unknown variable is introduced which is called the turbulent viscosity, ν_t.[3]

The $k - \epsilon$ model is one of the most widely used turbulence models in engineering applications due to its balance between computational efficiency and reasonable accuracy for a variety of flow problems. It solves two separate transport equations: one for the turbulent kinetic energy (k) and another for the dissipation rate (ϵ).

- Turbulent kinetic energy (k): Represents the energy contained in turbulent eddies.
- Dissipation rate (ϵ): Represents the rate at which this kinetic energy is converted into heat due to viscous forces.

These two equations allow for the modeling of turbulence effects in flows without directly simulating the smallest scales, making the $k - \epsilon$ model computationally efficient for large domains.

The standard $k - \epsilon$ model is based on the assumption of isotropic turbulence, which means it performs well in flows that do not have significant swirling or re-circulating behavior. For cases with complex recirculation or near-wall flows, modifications to the standard model, such as the Realizable k-epsilon or RNG k-epsilon models, may improve accuracy.

2.1.3 Convergence

As the solution converges, residuals should decrease, indicating that the governing equations are being satisfied more accurately with each iteration. A simulation is

typically considered converged when residuals drop below a predefined threshold (often around 10^{-4} or lower), though this can vary depending on the model and solver settings.

Residuals represent the difference between the left and right-hand sides of the governing equations (such as the Navier-Stokes or energy equations) at each iteration. They are calculated as the solver iterates, attempting to satisfy the governing equations. High residuals indicate large discrepancies between the calculated and actual values, while decreasing residuals suggest that the solution is approaching convergence.

The energy residual measures the imbalance between the energy entering and leaving each cell, accounting for conduction, convection, radiation, and source terms. In simulations using turbulence models, additional residuals are used to track turbulent quantities, such as turbulent kinetic energy and dissipation rate.

In addition to monitoring residuals, key physical quantities such as mass flow rate, temperature, and pressure should also be observed to ensure that they stabilize. A steady and converged state is achieved for this case when the residuals are stable, the average room temperature remains constant, and no further significant changes are observed in these monitored quantities.

2.1.4 Porosity Modelling

At the border between the inlets and the room, perforated plates are used. There are generally two ways to model this: either explicitly creating the perforated plate in a 3D modeling program or treating it as a porous zone. A porous zone is an approximation used when explicitly modeling the geometry of the perforated plate would be too computationally expensive. Instead a resistance model is used to introduce a resistance to the flow.

When the thickness of the perforated plate is large relative to the porosity (defined $\frac{A_p}{A}$ where A_p is the area allowing flow and A is the total cross-sectional area of the plate), it's recommended to model it explicitly. In this case, with a plate thickness of 1 mm and porosity of 0.45, a porous zone is chosen. Meaning the flow can be modeled accurately by using a resistance model instead of explicitly resolving every hole in the plate. This simplifies the computation while still capturing the essential characteristics of flow through the perforated plate.

To initialize this shear-stress-based treatment, two constants are needed: P_i (porous inertial resistance) and P_v (porous viscous resistance). They come from the pressure-drop equation, Equation 2.9, which can be linked to the Darcy-Forchheimer equation for pressure-drop as shown in Equation 2.10. The expressions for P_i and P_v are given in Equation 2.11, where f and d are the Forchheimer and Darcy coefficients, typically determined experimentally but here they are calculated numerically through an empirical approach.

$$\Delta p = -\rho(P_i|u_{s,n}| + P_v)u_{s,n} \tag{2.9}$$

$$\Delta p = \mu \cdot d \cdot L \cdot u + \frac{\rho}{2} \cdot f \cdot L \cdot u^2 \tag{2.10}$$

$$P_i = \frac{f \cdot L}{2}, \quad P_v = \frac{\mu \cdot d \cdot L}{\rho} \tag{2.11}$$

The empirical studies by Idelchik [4] are used to assess the coefficients. The case used here is for a thin plate with a Reynolds number $R_e < 10^5$, based on Diagram 8-2[4]. Idelchik calculates the pressure drop using Equation 2.12, which can be simplified by assuming the mean velocity at the obstruction, w_1, is equal to the velocity u, meaning no change in velocity occurs over the plate. The formula for calculating k is defined in Equation 2.13, where $\bar{f}$ is the area which allows flow, ζ_{1quad} and ζ_φ are resistance coefficients, $\bar{\varepsilon}_{0,Re}$ is determined from the tables at diagram 8-2. [4].

$$\Delta p = \kappa \cdot \frac{\rho w_1^2}{2} \tag{2.12}$$

$$\kappa \widetilde{=} \left(\zeta_\varphi + \bar{\varepsilon}_{0,Re} \cdot (\zeta_{1quad} - \bar{f})^2\right) \frac{1}{\bar{f}^2} \tag{2.13}$$

Since Idelchik's pressure coefficient κ is quadratic, we can assume the Darcy coefficient is zero, and the viscous resistance acting upon the fluid will be negliable compared to the inertial resistance. Setting Equation 2.10 equal to Equation 2.12, results in the expression for Forchheimer's coefficient in Equation 2.14.

$$f = \frac{\kappa}{L} \tag{2.14}$$

2.1.5 Mesh

When solving CFD problems within a software such as STAR-CCM+, it is done numerically throughout the computational domain. The computational domain is in turn divided into small control volumes, in which the governing equations are discretized and solved for iteratively. Together, these control volumes make up the mesh.

As the equations need to be solved in each of these control volumes, both the computational cost and quality of the solution are strongly correlated with the number of control volumes. Generally, a finer mesh with small control volumes is needed where the gradients of flow properties are large. And larger control volumes can be used when the gradients are small. Due to these correlations, it is of great importance to achieve a good mesh in order to reduce computational cost and increase the quality of the solution.

2.2 Thermodynamics

Thermodynamics is a branch of physics which oversees the relationship between energy, heat, work and temperature. The main focus being the energy, and to predict the exchange and flow of it throughout a system, and its surroundings. A thermodynamic system can generally be described in a few ways, seen in Table 2.1. An open system allows both mass and energy to cross its boundaries. Mass can flow in and out of the system, work can be done by the system on the surroundings or vice versa, and heat can be exchanged with the surroundings.

System	Mass flow	Work	Heat
Open	✓	✓	✓
Closed	×	✓	✓
Thermally isolated	×	✓	×
Mechanically isolated	×	×	✓

Table 2.1: Various kinds of thermodynamical systems.

A system is said to have an internal energy U. The internal energy is a relative measure of a systems current state compared to its standard internal state. Depending on what kind of system, the equation for the internal energy will change.

2.2.1 Conduction

This mode of energy transfer occurs because the energy contained in molecules arises from their random translational, rotational, and vibrational motions. The process of thermal conduction involves the transfer of energy from higher energy particles to those with lower energy within the same substance, due to particle interactions. When these molecules collide, the molecule with higher energy transfers some of its energy to the molecule with less energy.

Consider the refrigeration cycle in Figure 2.1, where at the evaporator coil there is cold refrigerant in the pipes and on the other side there is warm air. This creates a temperature gradient through the pipe walls, where the molecules near the air are warmer than the ones near the refrigerant. Due to the constant collisions of molecules, energy is transferred from the the hotter to the cooler molecules. Resulting in a cooling effect of the air.

The rate at which heat is transferred via conduction is quantified by Fourier's law. In one dimension, this law is expressed in Equation 2.15.

$$Q_x = -k \cdot A \frac{dT}{dx} \tag{2.15}$$

Here, Q_x represents the heat transfer rate in the x-direction, where the area is perpendicular to the direction of heat transfer. The unit for heat flux, Q, is watts (W).

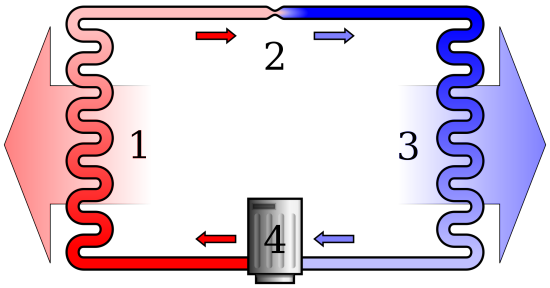

Figure 2.1: A refrigeration cycle, where: 1) condensing coil, 2) expansion valve, 3) evaporating coil, 4) compressor.[5]

This relationship indicates that heat flows in the direction of decreasing temperature, guided by the thermal conductivity k of the material and the temperature gradient across it.

2.2.2 Convection

Thermal convection occurs through the movement of fluids driven either by external forces or by buoyancy forces resulting from density differences induced by temperature variations within the fluid.

Relating this to the cooling cycle in Figure 2.1, convection is taking place in both the evaporator and condenser coils. The primary role of the evaporator in an air conditioning system is to absorb heat from the warmer environment. It does this through the evaporation of the refrigerant. The condenser's role is to expel the absorbed heat to the cooler environment. It does this by condensing the vaporized refrigerant.

This process is described with Newton's law of cooling in Equation 2.16.

$$Q = A \cdot h(T_s - T_\infty) \tag{2.16}$$

Where h is the convective heat transfer coefficient, T_s is the temperature on the surface, and T_∞ is the temperature far from the surface. In the evaporator this means that T_s is the temperature on the evaporator coils and T_∞ is the temperature at the inlet to the evaporator. The same goes for a condenser. Just as with conduction we get the unit W.

2.3 Optimization

Optimization problems can be approached using various algorithms, each suited to different types of functions and solution spaces. In this problem, the objective function is complex, unknown, and may or may not have a clear global minimum. Because the function is likely to contain multiple local minima, traditional deterministic optimization methods (e.g., gradient descent, linear programming) are not

ideal; they are typically more effective on simpler, convex problems with well-defined solution spaces.

Instead, metaheuristic optimization algorithms are used, which are designed to navigate complex, high-dimensional landscapes and locate solutions close to optimal. These methods do not guarantee a global minimum but are valuable in exploring a solution space thoroughly, making them suitable for non-linear, multi-variable problems like this one.

2.3.1 Genetic algorithm

A Genetic Algorithm is an algorithm made to mimic the evolutionary effect in biology. Where individuals of a population compete to spread their attributes to the next generation. Each individual have a set of genes, attributes, which impacts its fitness. Fitness is the creatures ability to survive, higher fitness means a higher chance of surviving. One way of preserving the fitness of a population is *Elitism*, where the fittest individual is copied to the next generation. Thus ensuring that the maximum fitness is never decreasing. An outline for the steps of a genetic algorithm can be seen in Figure 2.2. Where the process is repeated until convergence, which is when the fitness doesn't improve further.

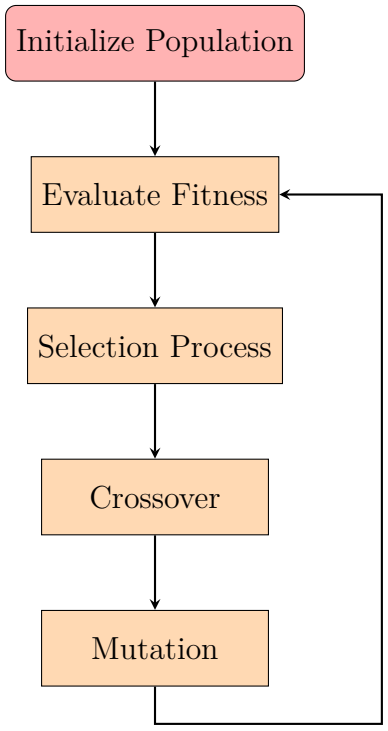

Figure 2.2: Process for a Genetic Algorithm

2.3.1.1 Mutation

Mutation is a random process which changes the current value of a gene by chance. Showcased in Figure 2.3

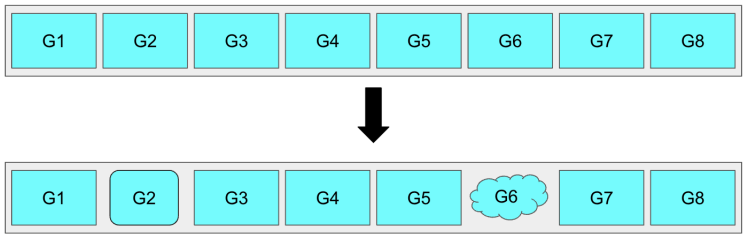

Figure 2.3: Genes 2 and 6 undergoes mutation, resulting in different values for those genes.

2.3.1.2 Crossover

Crossover is when two individuals exchange one or multiple genes with each other. Showcased in Figure 2.4

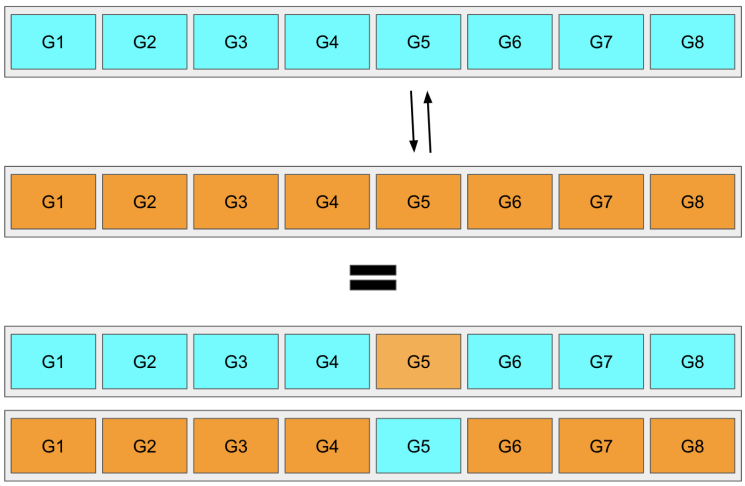

Figure 2.4: Single gene crossover between individuals, resulting in offspring with genes from both parents.

2.3.2 Particle Swarm Optimisation

Particle Swarm Optimization (PSO) is an optimization algorithm first described by Kennedy and Eberhart in 1995.[6] Designed to mimic social behaviour it was later discovered to perform optimization. The algorithm solves the problem by taking particles as solution and moves them over the search space. The position of the particles are influenced by the swarms best known position and its own best known position.
The particles updates their position with Equation 2.17. Where the position at t, the current time step, is updated using the velocity of the next time step, $t + 1$.

$$x_i(t + 1) = x_i(t) + v_i(t + 1) \tag{2.17}$$

For a simple PSO, the velocity of the particle is calculated using Equation 2.18. c_1 and c_2 are social parameters, which determine if the particles should follow their own best solution or the swarms best solution. w is the inertia of the swarm, determining how much of the previous time step is carried over to the next step. r_p and r_g are random numbers that impact how much the particle should follow the swarms best solution or it's own. $\hat{y}_j$ is the global best position and y_{ij} is the best location in dimension j for particle i.

$$v_{ij}(t + 1) = wv_{ij}(t) + c_1r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + c_2r_{2j}(t)[\hat{y}_j(t) - x_{ij}(t)] \tag{2.18}$$

2.3.3 Simulated Annealing

Simulated Annealing (SA) is an optimization algorithm first introduced by Kirkpatrick et al. in 1983, inspired by the annealing process in metallurgy, where controlled cooling of a material allows it to reach a stable state with minimal internal energy.[7] The algorithm mimics this physical process to solve optimization problems by gradually exploring the solution space. The concept of temperature is used to probabilistically accept solutions that are worse than the current solution, with the goal of escaping local minima early in the search process. As the algorithm progresses, the temperature decreases, reducing the probability of accepting worse solutions and allowing the search to focus on fine-tuning around the current best solution.

The theoretical foundation for Simulated Annealing is linked to the Metropolis algorithm, where a new solution is accepted with a probability governed by the Boltzmann distribution. The acceptance probability is calculated using Equation 2.19, where ΔE is the difference in objective function values between the current solution and the new solution, T is the current temperature, and P_{accept} is the acceptance probability:

$$P_{\text{accept}} = \exp\left(\frac{-\Delta E}{T}\right) \tag{2.19}$$

As the temperature T decreases over time, the algorithm transitions from exploration to exploitation, eventually converging towards an optimal or near-optimal

solution. In this way, Simulated Annealing can efficiently navigate complex search spaces with multiple local minima, a feature that has been analyzed in depth by Romeo and Sangiovanni-Vincentelli. [8]

An extension of this approach, known as dual annealing, incorporates both global search and local search phases, refining solutions by combining the traditional simulated annealing framework with a local optimization procedure. This hybridization improves convergence in challenging problems with complex landscapes, as shown in the work of Xiang et al. [9].

2.3.4 Differential Evolution

Differential Evolution (DE) is an optimization algorithm first introduced by Storn and Price in 1997. [10] It is designed to optimize continuous functions by maintaining a population of candidate solutions, which evolve over generations using a mutation and crossover mechanism. Unlike traditional genetic algorithms, DE uses vector differences to perturb the population, ensuring an effective search of the solution space.

Each candidate solution, or individual, is updated by creating a trial vector. The trial vector, v_i, is formed by adding the weighted difference of two randomly chosen individuals to a third individual, as shown in Equation 2.20. This mutation strategy allows DE to explore the search space by directing the perturbation towards better areas.

$$v_i = x_{r1} + F \cdot (x_{r2} - x_{r3}) \tag{2.20}$$

Here, x_{r1}, x_{r2}, and x_{r3} are three randomly selected individuals from the population, and F is the mutation factor, which controls the amplification of the differential variation.

DE also employs a crossover operation, where the trial vector competes with the original individual in a one-to-one competition. The individual is replaced or combined with the trial vector only if the trial vector has a better fitness, allowing DE to retain superior solutions as the generations progress. This selection mechanism is simple yet effective in maintaining diversity and focusing the search, making DE a robust algorithm for continuous optimization problems. A comprehensive survey of DE's state-of-the-art advancements was provided by Das and Suganthan. [11]

2.4 Neural Network

Artificial Neural Networks are inspired to mimic the structure and function of the human brain. The human brain is composed of billions of neurons connected by synapses. Each neuron receives inputs, processes them, and transmits the output to other neurons. In ANN's this is represented by the following instances:

- Neurons (Nodes): Units that receive input, apply a transformation, and pass the output to the next layer.

- Synapses (Weights): Connections between neurons that have associated weights, which adjust during training to minimize error.
- Layers: Consists of nodes, a ANN has an input layer and an output layer with a number of hidden layers in between.

A basic representation of a ANN can be seen in Figure 2.5.

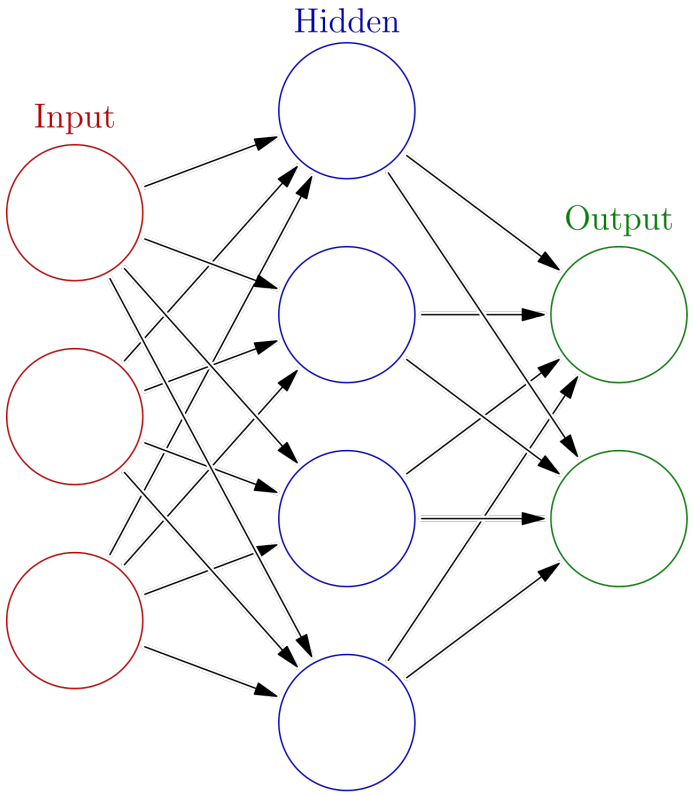

Figure 2.5: A simple representation of a Neural Network[12]

2.4.1 Universal approximation theorem

The Universal Approximation Theorem is a fundamental result in the field of Artificial Neural Network, which states that a feedforward neural network with at least one hidden layer can approximate any continuous function to any desired degree of accuracy, given sufficient neurons in the hidden layer and appropriate weights.[13]

3
Methods

This chapter introduces the methods and approaches utilized in this project towards optimization. The CFD domain and its geometry is presented, followed by the modeling of certain components. Then the process of optimization.

3.1 CFD Domain

The CFD domain is the parts of the system modelled in STAR-CCM+. Here that would be the "server room". The room is created as a simple block with another block on top, to represent the outlet. The other components are placed within the room.

3.1.1 Geometry

The room is designed to represent a server room with cabinets generating large amounts of energy. The dimensions can be seen in Figure 3.1. The geometry was created in STAR-CCM+'s own 3D modeling tool. The room, the inlets, the cabinets and the maintenance locker were all modelled in separate files to increase modularity and easily make small changes to certain parts without impacting the total assembly.

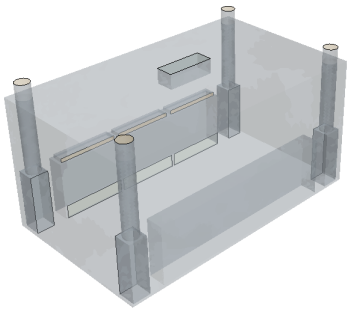

Figure 3.1: Baseline geometry of the room

3.1.1.1 Inlets

The room is cooled by transporting cold air in using four ventilation inlets, which together can deliver 6 kg/s of air into the room. The inlets are placed at the corners, and the outlet is positioned in the center. See Figure 3.2 for the geometry of the inlets.

Figure 3.2: Ventilation inlet with inlet at the top and outlet at the bottom, both highlighted.

When modeling the porous region in STAR-CCM+, this can be done in two ways, depending on the flow characteristics. If the flow is perpendicular to the perforated plate on the inlets, a resistance-based treatment of the baffle interface is advised. For predominantly tangential flow, a shear-stress-based treatment is used. Examining the inlets' flow behavior (without a baffled area) in Figure 3.3 and Figure 3.4, the flow appears to be a mix of tangential and normal components. The flow is more tangential at the top and more normal near the bottom.

To better understand which component of the velocity vector dominates, we refer to Figure 3.5. It becomes clear that the tangential velocity is dominant, so we proceed with the shear-stress-based treatment.

For a higher Reynolds number or a thicker plate, Idelchik recommends using Diagram 8-5[4] for plates where $\frac{L}{d_h} \leq 0.015$, where L is the plate thickness and d_h is the hole diameter. We performed calculations for both thicker and thinner plates under varying Reynolds numbers, since inlet mass flow variations impact velocity and, consequently, the Reynolds number. Using Diagram 8-3[4] for a thick plate and Diagram 8-1[4] for a thin plate with Reynolds number $R_e > 10^5$ [4], and comparing these to a thin plate with $R_e < 10^5$, we observe small differences, as shown in Table 3.1.

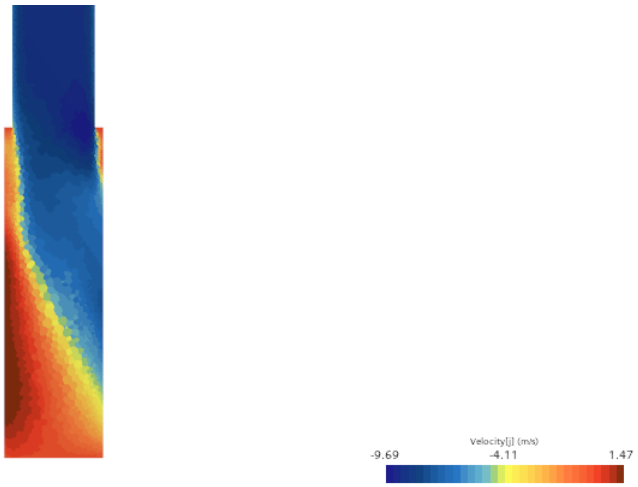

Figure 3.3: Velocity tangential to the porous outlet. Outlet to the right.

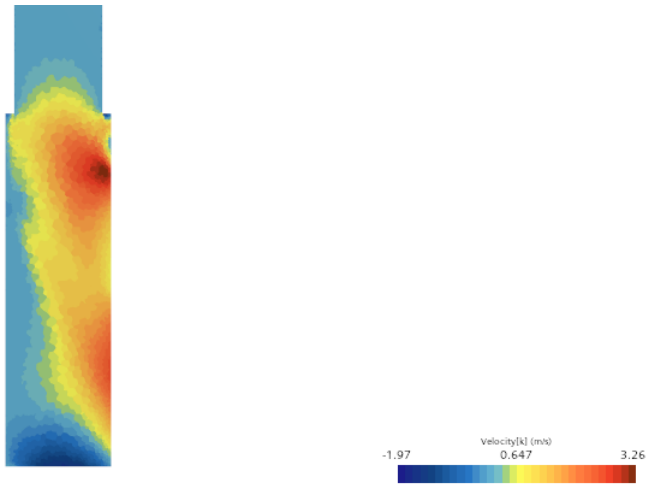

Figure 3.4: Velocity normal to the porous outlet. Outlet to the right.

	Thin $R_e < 10^5$	Thin $R_e > 10^5$	Thick
P_i	2.76	3.07	2.96

Table 3.1: Comparison of the coefficients for thin vs. thicker plates.

Since $\frac{L}{d_h} > 0.015$, the value for the thick plate is chosen.

3.1.1.2 Cabinets

In this model there are 3 cabinets, each capable of generating $26kW$ of waste heat. The cabinets have an air intake in the bottom and an outlet at the top which can transfer a maximum $6m^3/s$ of air. To accurately mimic the behaviour of the cabinets, Equation 3.1 was used to set the outgoing temperature, transferring the energy generated to the air flowing through. See Figure 3.6 for the cabinet model.

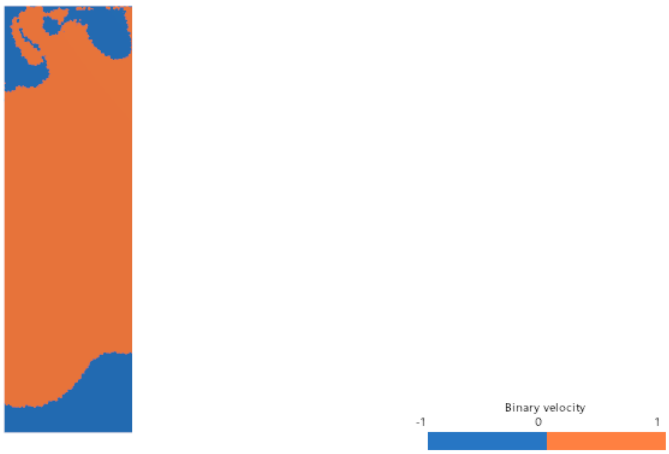

Figure 3.5: Binary velocity profile at the outlet, where 1(orange) indicates a larger tangential component and -1(blue) a larger normal component. Outlet in the plane.

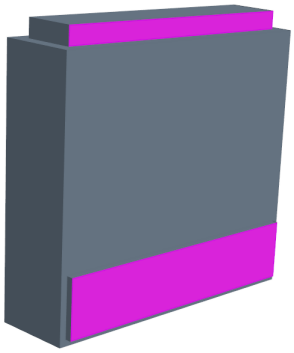

Figure 3.6: Cabinet model with outlet at the top and inlet at the bottom highlighted

$$Q = \dot{m} C_p (T_{out} - T_{in}) \tag{3.1}$$

3.1.2 Meshing

In this project the polyhedral volume mesher provided in STAR-CCM+ is used. It has a good balance between speed and efficiency and enables a high-quality mesh. The mesher starts by building a tetrahedral mesh, then combines cells to generate the poly mesh with a lower number of cells, though each is slightly larger. It is recommended to use for flow simulations.[14] A finer mesh in areas of sharp velocity or temperature changes is necessary for accurate results but comes at the cost of increased computational time. Conversely, coarser meshes reduce computational cost but can introduce inaccuracies, particularly in regions of complex flow dynamics. Therefore, where the inlets and room connects the mesh is refined further. This is done by specifying these regions and setting a smaller base size for the mesh.

A mesh convergence study was conducted in order to assess the the significance of lowering the size of cells. The results and plot of the study can be seen in Figure 3.7 and Figure 3.8.

By lowering the mesh size, the effect on the room temperature average is negligible in Figure 3.7, while the time it takes to complete one simulation almost doubles, as seen in Figure 3.8.

The mesh refinement strategy in this thesis balances computational cost with solution accuracy. A base mesh size of 1m was selected after a convergence study showed that further refinement led to diminishing returns in temperature accuracy but greatly increased computational time.

In regions where flow gradients are sharp, such as around inlets and outlets, a finer mesh was used to capture the detailed behavior of the flow. This refinement was necessary to ensure accurate predictions of local phenomena without increasing the overall computational load. This hybrid approach to mesh refinement, coarser in regions with less variation and finer in critical zones, allowed for efficient simulations while maintaining high solution quality.

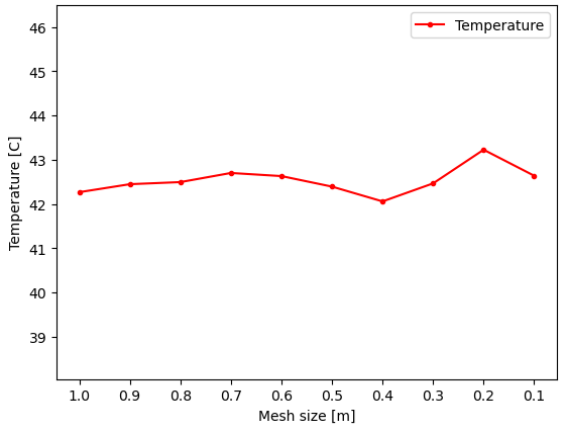

Figure 3.7: Mesh convergence study, until continuity convergence at 10^{-4} and temperature asymptotic convergence.

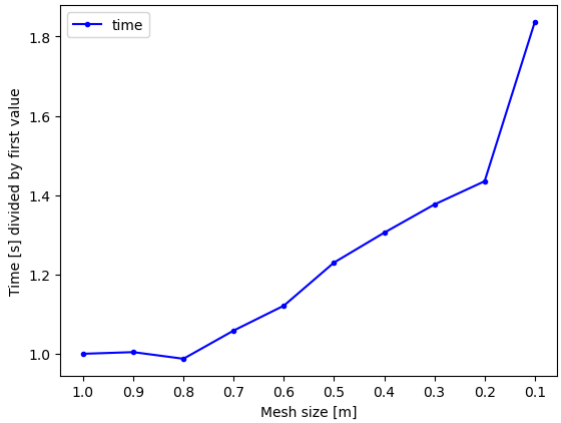

Figure 3.8: Mesh convergence study, until continuity convergence at 10^{-4} and temperature asymptotic convergence. Time normalized with the first simulation.

3.1.3 Turbulence Model: $k - \epsilon$

While the $k-\epsilon$ model is less accurate than high-fidelity methods like Direct Numerical Simulation (DNS) or Large Eddy Simulation (LES), it provides a good trade-off for many industrial applications where computational resources are limited. These high-fidelity methods, while more accurate, are often too computationally expensive to be applied in large-scale simulations, such as the server room modeled in this thesis. For example, DNS resolves all scales of turbulence, requiring extremely fine meshes, whereas LES only models the largest eddies, with smaller scales being resolved using subgrid models. In contrast, the $k-\epsilon$ model offers a practical approach for simulating turbulent flows in large domains where computational cost is a concern, such as in HVAC systems or industrial buildings. Especially when a large number of simulations are required as well.

3.1.4 Parameters

In the CFD model, there are 12 different parameters that are allowed to be changed. 13 including the width of the outlet, however that is set as a function of the length to always maximize the area. The parameters can be seen in Table 3.2 with the reference values. By reference values, it means the values which would be applied if no optimization would be done. As well as a non adaptive cooling system, always aiming at providing the same temperature inflow, 14°C, at a worst case scenario. Providing a level of over dimensioning to ensure that the requirements are satisfied.

Parameter	Value	Unit
Mass flow inlet 1	1.5	kg/s
Mass flow inlet 2	1.5	kg/s
Mass flow inlet 3	1.5	kg/s
Mass flow inlet 4	1.5	kg/s
Inlet temperature	14	$°C$
Rotation ventilation 1	0	°
Rotation ventilation 2	0	°
Rotation ventilation 3	0	°
Rotation ventilation 4	0	°
Length of outlet	1.6	m
Translation of outlet X	0	m
Translation of outlet Y	0	m

Table 3.2: Baseline parameter values

3.1.5 Parameter Constraints

The parameters all have minimum and maximum value shown in Table 3.3.

Parameter	min	max	Unit
Mass flow inlet 1	0	2	kg/s
Mass flow inlet 2	0	2	kg/s
Mass flow inlet 3	0	2	kg/s
Mass flow inlet 4	0	2	kg/s
Inlet temperature	14	28	$°C$
Rotation ventilation 1	0	90	°
Rotation ventilation 2	0	90	°
Rotation ventilation 3	0	90	°
Rotation ventilation 4	0	90	°
Length of outlet	0.48	1.6	m
Translation of outlet X	-2	2	m
Translation of outlet Y	-2	2	m

Table 3.3: The parameters constraints

3.1.6 Stakeholder Constraints

There are also a set of constraints supplied by different stakeholders which need to be accommodated when optimizing, these can be seen in Table 3.4. Then there are the common sense constraints, such as the translation of the outlet can't be so large that the outlet becomes separated from the room.

Constraint	Type of constraint	Value	Stakeholder
Average room temperature	Maximum	$28°C$	Client
Total mass inflow	Maximum	6 m^3/s	Constructor
Outlet area	Maximum	0.96 m^2	Constructor

Table 3.4: Constraints imposed by stakeholders

3.2 Energy efficiency metrics

In this study, energy efficiency is measured based on two key variables: the mass flow of air through the HVAC system and the temperature of that air. After determining the optimal geometry, these two factors become the primary controllable

elements within the system.

- Mass Flow: The amount of air circulated by the HVAC system, expressed in kg/s. By increasing the mass flow, more air is introduced into the system, which can reduce the average temperature of the room.
- Inflow Temperature: The temperature of the air entering the system, which directly impacts the cooling load.

The goal is to minimize the energy required to maintain the server room temperature below a specified threshold (28°C) by adjusting these two factors. It is well established that cooling air requires significantly more energy than simply increasing the mass flow of air at a higher temperature. This is because the energy required for cooling is proportional to the specific heat capacity of air and the temperature difference ($Q = \dot{m}C_p\Delta T$), whereas increasing the mass flow alone is proportional to the pressure drop over the system. In this study, the duct is assumed rather short and thus a very small pressure drop is expected. Whereas the outside temperature at the worst case scenario is 30°C, resulting in a large need for cooling.

3.3 Data Sampling

The data sampling was conducted in STAR-CCM+ using a Java macro that generated a Latin hypercube sample space. A simulation was run for each sample, iterating up to 2000 steps or until convergence.

3.3.1 Latin Hypercube Sampling (LHS)

Latin Hypercube Sampling (LHS) was chosen for this study to efficiently explore the high-dimensional parameter space of the server room's geometry and HVAC settings. LHS is a statistical method used for generating a distribution of plausible collections of parameter values from a multidimensional space. It divides each input variable into equally probable intervals and then ensures that each interval is sampled exactly once, which leads to a more uniform and comprehensive exploration of the parameter space compared to simple random sampling.

LHS was particularly suitable for this project because:

- Efficiency in High Dimensions: In contrast to random or grid sampling, LHS allows for a better coverage of the input space with fewer sample points. This is crucial given the high number of parameters (e.g., mass flow rates, inlet temperatures, outlet positions) that affect the performance of the system. It ensures that the sample space is thoroughly explored without needing an excessive number of simulations.

- Stratification: LHS ensures that the entire range of each parameter is sampled, preventing clustering of sample points that could occur with other methods,

like random sampling. This stratification minimizes the risk of missing important areas of the input space, especially in complex, multi-variable problems.

- Computational Efficiency: The uniformity of LHS allows for capturing variations across the parameter space while keeping the computational cost relatively low. This method provides a good balance between precision and the number of samples required, which is important in reducing the overall runtime of the simulations in STAR-CCM+.

By using LHS, this study was able to explore the impact of various geometric and operational parameters on the room's energy efficiency, ensuring a robust set of training data for the neural network model used later in the optimization process.

3.3.2 Sampling robustness

In terms of convergence, the residuals for continuity, momentum, and energy were monitored closely to ensure the stability of the simulation results. The continuity residuals represent the error in the conservation of mass across the simulation, while the momentum residuals track the conservation of forces acting within the flow. High residuals would indicate significant imbalances in the forces or mass within the system. Convergence was declared when the continuity residuals dropped below a threshold of 10^{-4}, and the average room temperature stabilized within 0.1°C over the final 75 iterations. This was critical, as any significant fluctuations in temperature or residuals indicated that the system had not reached a steady-state solution.

If the simulation would diverge, the script would refine the mesh by halving the base size and run it again. if it would diverge again, it would be halved once more. If the simulation still diverged, the sample point was discarded.

3.4 The optimal geometry

When designing the system, this is done against the worst case scenario. This is done by fitting an Artificial Neural Network against the sample data. This neural network is then used in an objective function in order to minimize the energy usage. Which in this case means maximize the inflow temperature and minimize the mass-flow so that the average temperature never exceeds 28°C.

When loading the data into the model it's important to have reliable data that won't skew the results. Thus outliers were removed, here defined as when the average temperature reached over 100°C. This threshold was chosen in order to avoid using data points where the input mass flow is low and inlet temperature is high. Which may result in a steadily increasing temperature and unrealistic operation setup.
The Artificial Neural Network is then used to approximate the room, it is trained on the remaining sample points and optimized using Optuna hyperparameter optimization.

Below in Table 3.5 the parameter space can be seen. Except for the parameters "Optimizer" and "Activation function" which were categorical values. For optimizers "Adam" and "SGD" were considered and for activation functions: "ReLu", "Sigmoid" and "Tanh".

Parameter	Min value	Max Value
Learning rate	0.001	0.1
Batch size	12	124
Train fraction	0.5	0.95
Number of epochs	100	4000
Number of hidden layers	1	6
Neurons in each hidden layer	24	512
Early stop	25	250

Table 3.5: The parameter space for the neural network hyperparameters

3.5 Optimisation methods

Several metaheuristic optimization methods were tested and compared to identify a near-optimal solution. These methods, outlined below, are each adapted for the type of irregular, multi-modal solution space encountered here. Metaheuristics increase the likelihood of locating a solution close to the optimal by exploring the global solution space and reducing the chance of becoming trapped in local minima. Furthermore, using multiple metaheuristic algorithms helps leverage the strengths of each, as certain algorithms may perform better on this specific function due to

its unique characteristics.
The choice of optimization algorithms, including Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Differential Evolution (DE) and Simulated Annealing (SA), was motivated by their ability to navigate complex, multi-dimensional search spaces. Each algorithm was tested to evaluate its performance in finding the optimal geometry for the server room. All algorithms were used in Python with appropriate libraries.

3.5.1 Genetic Algorithm

The initial population is randomly initialized and a tournament selection is used together with an adaptive mutation. A common problem with a "classic" genetic algorithm which uses a constant mutation rate for all chromosomes irrespective of their fitness is detailed in.[15] Where a very good individual is equally likely to be disrupted by mutation as a bad one. With an adaptive mutation, two probabilities are applied depending on what fitness the individual have. If the individual has a fitness below the average of the population, a high mutation rate is applied. If the individual has higher fitness than the population average a lower mutation rate is applied in order to keep the good genes. The rates chosen are 25% and 2% respectively. The Genetic Algorithm ran for 10000 generations with a population size of 1000, with 600 parents creating the offspring. Elitism was also used to conserve the 10 top performing individuals for each generation.

3.5.2 Particle Swarm Optimization

The Particle Swarm Optimization algorithm in this study begins by initializing a population of 1000 particles, each representing a potential solution within the search space. The particles adjust their positions iteratively based on two primary influences: their personal best position and the best-known position found by the swarm. The algorithm parameters were set as follows: $c_1 = 0.5$ and $c_2 = 0.3$, controlling the relative influence of each particle's personal experience and the swarm's collective experience.

To enhance the balance between exploration and exploitation, an inertia weight $w = 0.9$ was applied to each particle's velocity, retaining a significant portion of previous movement to allow exploration of the solution space. Additionally, the acceleration constants $k = 2$ and $p = 2$ were used to tune the velocity update, maintaining control over the search dynamics.

A linearly decreasing inertia weight is implemented to gradually reduce w over time, guiding the swarm from broad exploration to more focused exploitation as it converges. To address premature convergence, a small random perturbation is added to the velocity updates, ensuring diversity in particle movement and reducing the chance of stagnation in local optima. This setup, with 1000 particles and the selected parameter values, is designed to enhance PSO's robustness in navigating the

complex, multi-dimensional objective function space.

3.5.3 Simulated Annealing

The SA algorithm in this study uses dual annealing, and was implemented with SciPy. The process begins with an initial temperature of 5230, allowing the algorithm to accept both improving and worse solutions, thus promoting escape from local minima in the early stages.

The temperature cooling schedule is managed by an adaptive scheme, with a restart temperature rate set to 0.00002. This low rate gradually cools the system, balancing global exploration with local fine-tuning as the algorithm progresses. Additionally, the accept parameter is set to -5, controlling the likelihood of accepting worse solutions. A higher accept parameter would result in more aggressive exploration.

The visit parameter of 2.62 helps determine how the search moves within the solution space. Higher values promote more extensive exploration, while lower values focus on local adjustments. With this value, the algorithm maintains sufficient exploration without deviating excessively from promising areas.

The algorithm runs for a maximum of 1000 iterations, during which the temperature decreases, shifting from a broad, exploratory search to a narrower focus on exploitation.

3.5.4 Differential Evolution

The Differential Evolution algorithm in this study utilizes a Sobol sequence to initialize a population of 100 candidate solutions, providing a quasi-random sampling that maximizes initial coverage of the solution space. For each candidate solution, the algorithm randomly selects three other individuals from the population to generate a trial vector. The trial vector is created by mutating the difference between two of these individuals, scaled by a mutation factor F that is uniformly selected between 0.5 and 1 at each iteration, and adding the result to the third individual. After mutation, a recombination constant C =0.7 is applied. This parameter, also known as the crossover probability, controls how much of the trial vector's information is retained when blended with the current candidate solution. A higher recombination value like 0.7 allows a larger portion of the mutated vector to influence the solution, encouraging exploration of new areas in the search space. However, it also risks reducing population stability.
The maximum number of iterations is set to 1000, allowing sufficient time for convergence. At each iteration, if the trial vector outperforms the candidate solution in fitness, it replaces the candidate in the population.

4

Results

Here the performance of the Artificial Neural Network along with the optimal geometry will be presented and compared to the baseline case.

4.1 Neural network

The result of the hyper parameter optimisation can be seen in Table 4.1, where the early stop value is the number of epochs to wait for improvement before stopping the training. These parameters were then used when training the model.

Parameter	Value
Learning rate	0.015
Optimiser	Adam
Batch size	42
Train fraction	0.85
Number of epochs	2000
Number of hidden layers	2
Neurons in each hidden layer	255,236
Activation function	Relu
Early stop	50

Table 4.1: Hyper parameters to train the network.

The Mean Square Error (MSE) was used as loss function when training the model. The results during training can be seen in Figure 4.1. T

he performance of the neural network model to determine the geometry can be seen in Table 4.2. The R^2 value, the proportion of the variation which the model can explain, shows that the model predictions are close to the actual data points. The Mean Absolute Error (MAE) shows that on average a data point will be 1.04 degrees of from the actual simulated value.

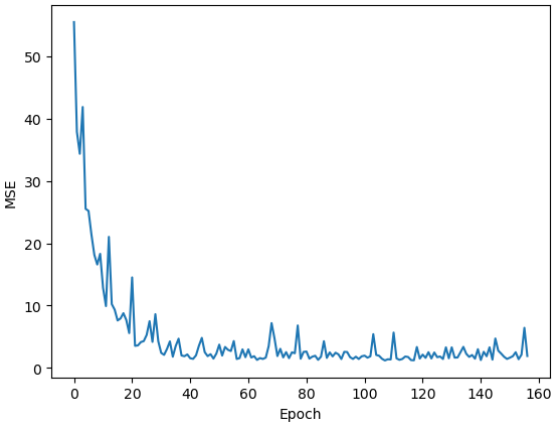

Figure 4.1: Mean squared error compared to the epoch

Criterion	value
MSE	1.86
MAE	1.04
R^2	0.974

Table 4.2: Performance of the neural network model on the test set.

4.2 Optimal geometry

The room with no optimizations is used as a baseline. The results of all models can be seen in Table 4.3, the best geometry over all was found by Particle Swarm Optimization. A comparison of the parameters between the best performing geometry and the baseline is seen in Table 4.4 together with the optimal geometry in Figure 4.2.

Model	Predicted Average room temperature [$°C$]	Actual Average room temperature [$°C$]	Total mass-flow [kg/s]	Inflow temperature [$°C$]
Baseline	–	26	6	14
Genetic Algorithm	27.99	27.835	6	17.2
Simulated Annealing	27.99	28.9	6	17.75
Particle Swarm Optimization	27.99	27.76	6	17.6318
Differential Evolution	27.76	27.77	6	17.52

Table 4.3: Comparison of optimization methods

Parameter	Baseline	Optimal	Unit
Mass flow inlet 1	1.5	1.47	kg/s
Mass flow inlet 2	1.5	0.79	kg/s
Mass flow inlet 3	1.5	1.80	kg/s
Mass flow inlet 4	1.5	1.92	kg/s
Inlet temperature	14	17.63	$°C$
Rotation ventilation 1	0	87.90	°
Rotation ventilation 2	0	58.34	°
Rotation ventilation 3	0	46.15	°
Rotation ventilation 4	0	89.33	°
Length of outlet	1.60	1.40	m
Translation of outlet X	0	-1.87	m
Translation of outlet Y	0	0.74	m

Table 4.4: Comparison of parameters between baseline and optimal geometry

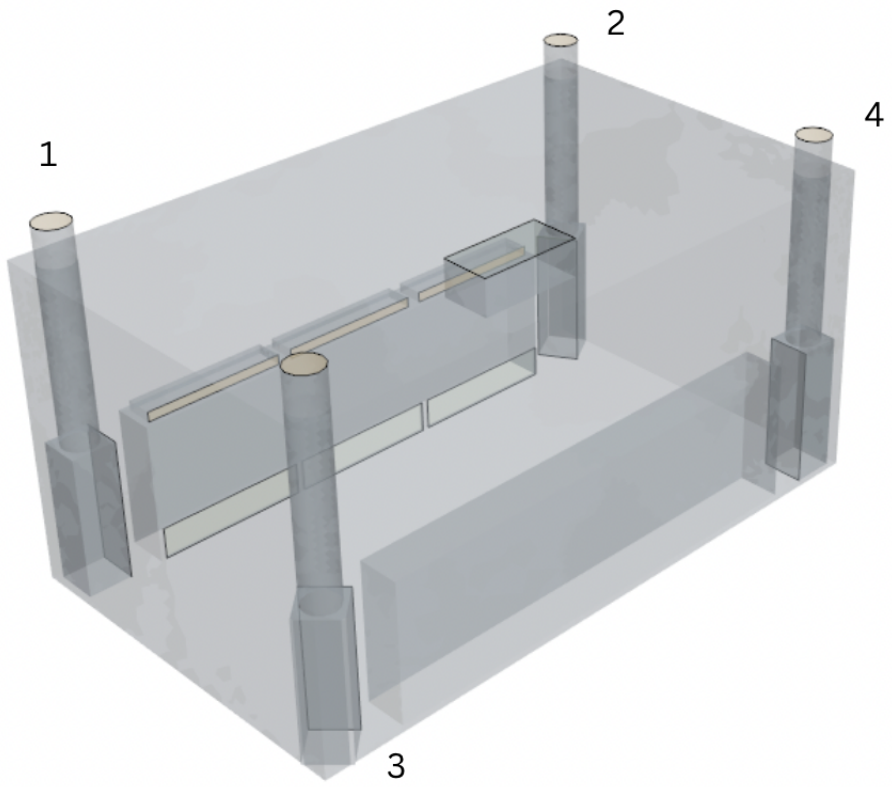

Figure 4.2: Optimal geometry with inlets specified

5
Discussion

In this chapter, the results and methods are evaluated and discussed. Potential improvements and future prospects for making the methods more rigorous are also considered, as well as the impact of the results.

5.1 Results

The results show that by allowing for changes in geometry, an increase of 3.63° C in the flow temperature can be achieved without violating any constraints. In a worst-case scenario of 30°C external temperature, this corresponds to a 22.69% reduction in energy used for cooling.

When looking at the other solutions, they share no obvious similar pattern, but are rather scattered. Which is not too unexpected with a problem that can have many local minima where the parameters influence each other in a way that can be hard to predict. Resulting in many different, close to optimal solutions. It is however noted that some solutions were mirrored, which in this case where there is a symmetry, implies that there are some underlying correlation that affects the airflow in a meaningful way.

5.2 Geometry

In this project, only the existing parts were modified. New components could be designed and added to alter flow characteristics, potentially further reducing cooling requirements. However, this step is more exploratory and difficult to achieve with machine learning, so it was excluded from this project. Nevertheless, it should be considered in real-world applications.

The optimal geometry may not be ideal from an engineering standpoint. As seen in Figure 4.2, two inlets direct their flow into adjacent structures, which may work for worst-case scenarios but might not be suitable for regular operation. This highlights the importance of understanding the operational phase. For instance, if only two cabinets are running regularly with the third on standby, a worst-case scenario may never occur. In such cases, a multi-objective optimization targeting typical operational phases would be more appropriate.

5.3 Sensitivity Analysis

The sensitivity analysis highlights how various parameters influence the average room temperature. As expected, the results presented in Table 5.1 reveal a strong correlation between inflow temperature and average room temperature. Intuitively, increasing the inflow temperature leads to a rise in the average room temperature. Similarly, the mass inflow shows a negative correlation: as mass inflow increases, circulation improves, pushing hot air out of the room, thereby lowering the temperature.

Other parameters, such as rotational angles (RA1, RA2, RA3, RA4) and translations, exhibit relatively weaker direct effects on room temperature, suggesting that they may have more subtle or indirect influences when combined with other variables. However, looking at the optimal geometry the angles are oddly specific, this might indicate that there is a flow pattern that arises with the angles at this exact combination. And that the combination of all inlets play a crucial role in creating said airflow.

Parameter	Correlation with Room Temperature
Length	0.00
RA1	-0.03
RA2	0.01
RA3	-0.01
RA4	0.01
Translation X	0.05
Translation Y	0.03
Inflow temperature	0.48
Mass flow 1	-0.37
Mass flow 2	-0.41
Mass flow 3	-0.36
Mass flow 4	-0.40

Table 5.1: Correlation between parameters and average room temperature.

When analyzing the combined effects of certain parameters, a more complex and nuanced behavior is revealed. In Figure 5.1, the interaction between $RA1$ (the rotational angle of inlet 1) and the mass flow from inlet 1 is explored, with all other parameters held at baseline values. The results indicate that the relationship between these two parameters is not linear, which is seen by the heat in the heat-map going diagonally across the figure. Demonstrating a significant level of interaction.

This indicates that parameters are not working in isolation but rather affect each

other in ways that are not immediately intuitive. As the complexity of the system increases, such as when the workload of the server cabinets varies, understanding these interactions becomes increasingly challenging. This underscores the need for advanced machine learning models capable of capturing these intricate dynamics and optimizing the system efficiently.

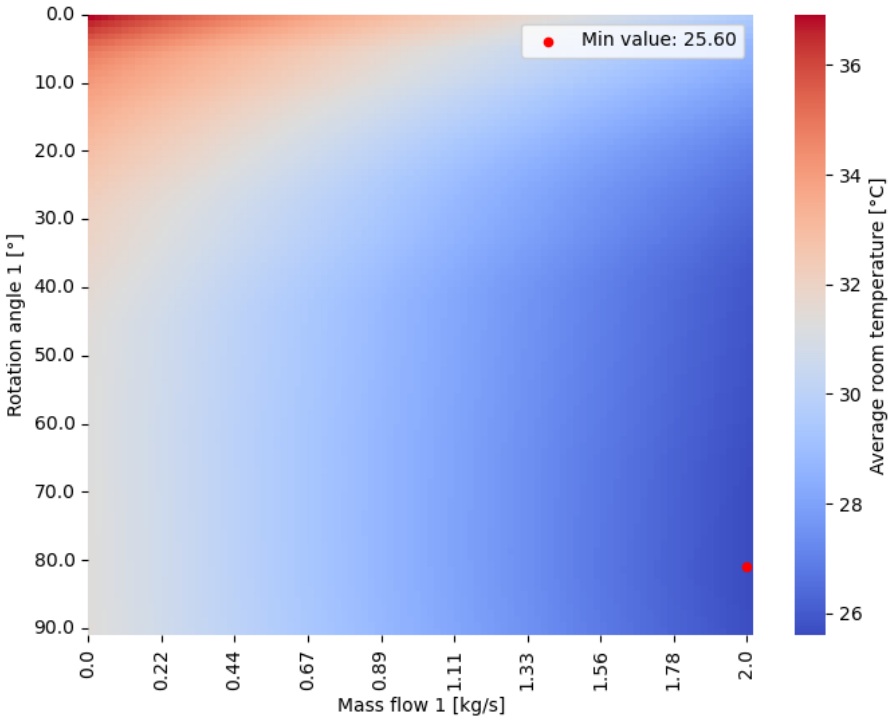

Figure 5.1: Multivariate sensitivity analysis of rotational angle ($RA1$) and mass flow (Inlet 1), heat-map showing the average room temperature.

5.4 Sampling

The response variable chosen for sampling was the average room temperature, however, a more thorough analysis could include additional points, such as temperature and velocity at various locations in the room or various server loads. This would require more from the Neural Network but could provide deeper insight into the factors affecting room temperature. The constraint where able to hold never the less which was sufficient for this scope.

Data points were discarded when the net energy input caused continuous heating or when the inlets pointed to a wall with high mass flow, resulting in diverging gradients. While these samples were few, they could be avoided by either creating a sample space that excludes such cases or by having the simulation adjust more conservatively with each iteration. This would increase simulation runtime but could reduce divergence, leading to a more complete sample space.

The optimal geometry also has two inlets nearly at 90 degrees, which is a boundary case for these parameters as can be seen in Table 3.3. More simulations at the sample space limits may be necessary, as the Latin hypercube method, while good at spreading the sample points within the space, may miss edge cases.

5.5 Neural Network

The neural network used to approximate the room's behavior was trained on a sample set of 4681 points, with 247 points reserved for testing. The network performed well, achieving a Mean Square Error of 1.86 and an R-squared value of 0.974, indicating that the model captured the relationship between input parameters and room temperature.

However, further analysis of the MSE values is needed to assess how the model performs across different regions of the input space. Simulations with divergent results were observed to be mostly caused by low mass flow combined with high inflow temperature, which limited sampling. Also, MSE of 1.86 shows that there appear to be some points in the test data that are not accurately represented in the training data, this can be seen in Figure 4.1 where there are some spikes, skewing the results.

5.6 Optimization

It is hard to say if the solution by the Particle Swarm Optimization actually is the best one, there is a difference in results when comparing the predicted temperature by the neural network, compared to the simulated value. There is therefore no guarantee that the solutions found, which all balance the line of 28°C, are the best. To actually find the optimal solution, the simulation software would have to be integrated directly into the algorithms. That would result in a more accurate result,

without relying on the neural network performance. However, it would be much more computationally expensive. The solution used in this project could be regarded as a good trade-off, sacrificing accuracy for less computational power.

5.7 Limitations

It is important to note that the optimization algorithms used in this study are inherently stochastic. This means that their performance can vary between runs due to random initializations and probabilistic operations. As a result, an algorithm that performed best in this experiment may not consistently outperform others in every run; it could perform differently if the experiment is repeated.

To mitigate this variability, one could run each algorithm multiple times and average the results. However, given that the objective function is unknown and has several local minima, averaging the parameters from different runs may not yield meaningful solutions. In fact, this approach could produce an average parameter set that corresponds to a solution worse than any of the individual local minima found. Therefore, while multiple runs can provide insight into the variability of algorithms, averaging results does not necessarily lead to better optimization results in this context.

Furthermore, the study was limited by computational resources, which limited the number of simulations and the complexity of the neural network model. Simplifications and assumptions in the CFD simulations, such as steady-state conditions and neglecting certain physical phenomena, may also affect the generalization of the results.

5.8 Future Work

Future iterations of this project could consider additional factors to further optimize energy usage. This study focused on geometry optimization under worst-case conditions, with all cabinets running at full power. Future work could explore adaptive control strategies, where the HVAC system dynamically adjusts to real-time conditions, such as fluctuating server loads and external temperatures. Analyzing the most common operational phases and adjusting the geometry accordingly would also be valuable. In addition, hybrid approaches combining the global search capabilities of Simulated Annealing or Particle Swarm Optimization with the fine-tuning of Genetic Algorithm could create a more robust optimization framework, particularly for complex objective functions with multiple local minima.

6
Conclusion

The objective of this thesis was to optimize the energy efficiency of a server room cooling system using machine learning techniques and Computational Fluid Dynamics, CFD, simulations. By applying Artificial Neural Networks ANN trained on sample data from Star-CCM+, this study identified the relationship between geometric configurations and energy consumption. The optimization process, which aimed to reduce energy usage, was conducted using four different algorithms: Genetic Algorithm (GA), Simulated Annealing (SA), Particle Swarm Optimization (PSO), and Differential Evolution (DE). This integrated approach bridges the gap between CFD simulations and machine learning, providing a novel methodology for optimizing cooling systems.

6.1 Key Findings

The results of this study demonstrate that allowing geometric adjustments in the server room can lead to significant energy savings. Specifically, allowing an increase of 3.63°C, from 14 to 17.63 °C, of the inflow temperature was achieved without violating operational constraints. This translates to a 22.69% reduction in cooling energy consumption during a worst-case scenario where the external temperature reaches 30°C.

Of the four optimization algorithms tested, PSO proved to be the most effective in balancing exploration and exploitation, thereby avoiding local minima and yielding the most optimal solutions. While the Genetic Algorithm and Simulated Annealing performed well, they showed limitations in escaping local minima, particularly due to early convergence and over-exploitation of fit solutions. Differential Evolution demonstrated a robust performance but did not outperform PSO in the final analysis.

6.2 Geometric and Sampling Considerations

A multi-objective optimization approach that simultaneously considers energy consumption, temperature uniformity, and equipment reliability during both worst-case and typical operations could provide more practical and universally applicable solutions.

Regarding sampling, focusing solely on average room temperature may overlook critical hotspots or areas with inadequate cooling. Incorporating spatially resolved temperature and velocity data would enable the ANN to capture complex thermal behaviors, leading to more accurate predictions and effective optimizations.

Bibliography

[1] Energimyndigheten. Energiläget i siffror 2023. https://www.energimyndigheten.se/nyhetsarkiv/2023/energilaget-i-siffror-2023/, 2023. Accessed: 2024-04-03.

[2] Luong Duc Long. An ai-driven model for predicting and optimizing energy-efficient building envelopes. *Alexandria Engineering Journal*, 79:480–501, 2023.

[3] Lars Davidson. *Fluid mechanics, turbulent flow and turbulence modeling*. Mar 2024.

[4] I E Idelchik. *Handbook of Hydraulic Resistance*. 1984.

[5] Wikipedia. Heating, ventilation, and air conditioning. *Wikipedia*, 2024. Accessed: 2024-04-03.

[6] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, pages 1942–1948 vol.4, 1995.

[7] Scott Kirkpatrick, C Daniel Gelatt, and Mario P Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[8] Fabio Romeo and Alberto Sangiovanni-Vincentelli. A theoretical framework for simulated annealing. *Algorithmica*, 6:302–345, 1991.

[9] Yudong Xiang, Duan Sun, and Wei Fan. Generalized simulated annealing algorithm and its application to the thomson model. *Physics Letters A*, 233(3):216–220, 1997.

[10] Rainer Storn and Kenneth Price. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359, 1997.

[11] Swagatam Das and Ponnuthurai Nagaratnam Suganthan. Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, 15(1):4–31, 2011.

[12] Bidragsgivare till Wikimedia-projekten. Artificiellt neural network, May 2023.

[13] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1989.

[14] Siemens Digital Industries Software. A New User's Guide to STAR-CCM+ Simulation - Part 3/5: Meshing. `https://community.sw.siemens.com/s/article/A-new-user-s-guide-to-STAR-CCM-simulation-Part-3-5-Meshing`, Year. Accessed on April 11, 2024.

[15] S. Marsili Libelli and P. Alba. Adaptive mutation in genetic algorithms. *Soft Computing*, 4(2):76–80, Jul 2000.

DEPARTMENT OF SOME SUBJECT OR TECHNOLOGY
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY