Optimization of Jet-Engine Geometries Using a Mathematica-Driven Simulation Framework and OpenFOAM

Sanay Nesargi

Abstract

This paper presents a novel simulation framework aimed at optimizing jet-engine geometries for enhanced aerodynamic performance. The framework integrates Mathematica for geometry creation and parameterization with OpenFOAM, a Computational Fluid Dynamics (CFD) software, for the simulation and analysis of jet-engine flows. Using *pimpleFOAM*, a transient solver for compressible flow, the framework automates the optimization process by adjusting engine geometry parameters iteratively. The combination of Mathematica's powerful modeling capabilities and OpenFOAM's high-fidelity simulation tools provides an efficient platform for optimizing jet-engine components such as nozzles and blades. Optimization results demonstrate significant improvements in performance, including increased thrust-to-weight ratios and improved pressure recovery.

Introduction

The design and optimization of jet-engine geometries have been central to the development of high-performance aerospace systems. Traditional methods for optimization involve manual iterations of geometry modification, CFD simulations, and performance evaluation, a process that is time-consuming and computationally expensive. This paper proposes a framework that automates geometry creation, CFD simulation, and optimization using Mathematica and OpenFOAM. The goal is to minimize the computational effort while improving aerodynamic performance, leveraging the strengths of both software packages [6].

Mathematica is used to parametrize jet-engine geometries and control design variables, while OpenFOAM handles the CFD simulation of flow dynamics around the engine components. Through an iterative optimization process, the framework refines geometry to maximize performance metrics such as thrust, efficiency, and pressure recovery.

The primary contributions of this work include developing an efficient approach to optimize jet engine geometries using the best-suited tools. It addresses the challenges of mesh compatibility between Mathematica-generated geometries and OpenFOAM's meshing utilities, which is essential for creating a seamless workflow [1], [7]. Additionally, we propose a semi-automated pipeline for geometry creation, mesh refinement, and simulation, significantly reducing manual intervention and enhancing efficiency. The study applies genetic algorithms to explore the

design space of key jet engine features, such as blade dimensions and count, identifying configurations that enhance thrust production. By employing both transient and steady-state simulations, this paper captures dynamic flow phenomena and provides valuable insights into how design changes affect engine performance [8]. Furthermore, we demonstrate the scalability of the framework through the integration of OpenFOAM's parallel computing capabilities, making it feasible to perform large-scale aerodynamic optimization tasks [9].

The paper is organized into separate sections to structure our research approach and findings. The Problem Statement section offers an overview of the problem domain, motivation for integrating Mathematica and OpenFOAM, and the objectives of the study. In the Methodology section, we detail the process of geometry creation, mesh refinement, solver configuration, and the optimization strategy used in this study. The Simulation Setup section describes the computational domain, solver settings, and mesh considerations, emphasizing the importance of both steady-state and transient simulations. The Results and Discussion section presents and analyzes the simulation results, illustrating the effects of various design changes and optimization efforts on jet engine performance. Finally, the Conclusion and Future Work section summarizes the key findings, contributions, and potential directions for future research, providing a broader context for the application of the proposed framework in the field of jet engine design optimization [10].

Problem Statement

Optimizing jet engine geometries is a cornerstone of aerospace innovation, directly influencing performance, efficiency, and emissions. The current approach to this problem typically involves a fragmented pipeline: CAD tools are used to create the geometry, specialized mesh generators prepare the computational domain, and CFD solvers like OpenFOAM are employed to analyze the flow dynamics. While effective, this approach is fraught with challenges:

1. **Fragmented Workflow**: The use of multiple tools for geometry creation, mesh generation, and simulation often leads to inefficiencies and data compatibility issues.
2. **Limited Parametric Control**: Many CAD tools lack the flexibility to easily modify complex geometries using parametric equations, slowing down the iterative design process.
3. **Manual Interventions**: Transferring data between tools often requires manual adjustments, increasing the potential for errors and elongating the design-simulation cycle.
4. **Optimization Bottlenecks**: The process of exploring design variations and assessing their performance remains resource-intensive and time-consuming.

To address these challenges, we propose an integrated framework that uses Mathematica for geometry creation and OpenFOAM for CFD analysis. This approach unifies design and simulation into a cohesive pipeline, allowing for rapid iterations and precise optimizations.

Through this approach we seek to optimize peak engine thrust and examine its impact on fuel efficiency to improve overall engine performance.

Key Advantages of the Proposed Method:

- **Seamless Integration**: Mathematica's symbolic and parametric capabilities enable precise control over complex geometries, while its ability to directly export simulation-ready meshes ensures compatibility with OpenFOAM.
- **Efficiency in Iteration**: By combining design and simulation, the framework eliminates redundancies and reduces the time required for geometry modifications and performance assessments.
- **Enhanced Optimization**: The use of parametric equations allows for systematic exploration of design spaces, enabling designers to quickly identify optimal configurations.
- **Streamlined Workflow**: Automation of key steps in the process minimizes manual interventions, reduces errors, and enhances reproducibility.

By bridging the gap between design and simulation, this framework aims to provide an alternative to how jet engine geometries are developed, offering a faster, more reliable, and highly iterative approach to optimization. This integration serves as a step forward in the quest for more efficient and sustainable aerospace systems.

Methodology

To effectively create the jet engine geometry, we need a solution which supports the following key functionalities:

1. **Parametric Equation Handling** to enable precise design of complex geometries using parametric equations, which are critical for streamlined and efficient components.
2. **3D Mesh Generation** allows the conversion of designs into computational meshes for CFD simulations, balancing accuracy, resolution, and fidelity.
3. **Geometric Transformations** which facilitate scaling, rotation, extrusion, and assembly of designs like blades or propellers.
4. **Iterative Design** allows rapid iteration by varying parameters such as blade pitch or curvature for optimization studies [14].

We chose Mathematica to create the jet engine geometry because:

1. **Symbolic and Numerical Power** which provides precise handling of parametric equations and geometric transformations.
2. **Visualization** with real-time 3D previews which simplify iterative design processes.
3. **Automation** with flexible scripting for parameter studies and geometry workflows.
4. **Mesh Creation** with tools generate simulation-ready meshes for further refinement.
5. **Interoperability** which supports export to formats compatible with other standard simulation tools.

We integrate the geometry created by Mathematica into the simulation capabilities provided by OpenFOAM to iteratively design and optimize parameters critical for jet engine performance. A summary of the approach adopted by us is illustrated in Figure 1.

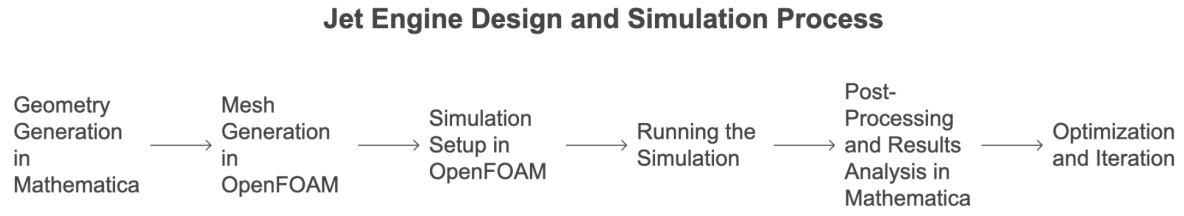

Figure 1: Methodology Overview

3.1 Geometry Creation in Mathematica

The geometry of the jet-engine components consists of key features, such as:

- **Blade Width**: Determines the aerodynamic profile and affects airflow distribution across the blade surface. Wider blades may provide higher thrust but could result in increased drag.
- **Blade Depth**: Refers to the thickness of the blade, which influences structural integrity and resistance to deformation under high stresses.
- **Blade Length**: The span of the blade along its axis. Longer blades can enhance thrust by increasing the interaction area with airflow, but they may also introduce challenges in material strength and vibration control.
- **Blade Count**: The number of blades in the configuration. Higher blade counts can improve thrust uniformity and reduce vibration but may increase manufacturing complexity and weight.

Mathematica's ability to handle parametric equations and generate 3D meshes provides a strong foundation for geometry creation. The geometry is exported to STL format for compatibility with OpenFOAM's meshing utilities.

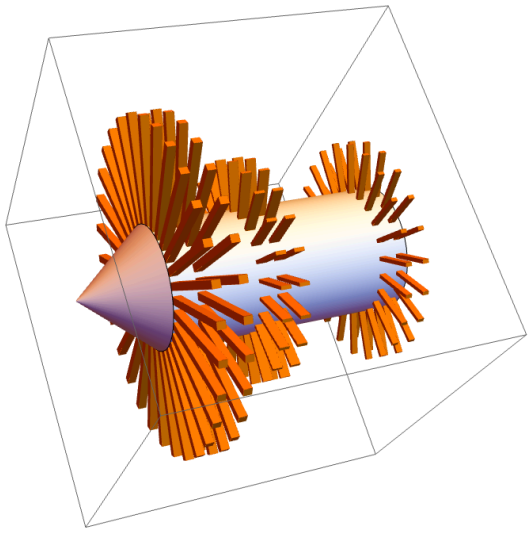

Figure 2: Mathematica-generated parametric geometry of jet-engine components.

3.2 Meshing with OpenFOAM

In any computational fluid dynamics (CFD) simulation, the ability to create a high-quality mesh is fundamental to obtaining accurate results [4]. The mesh defines how the simulation domain is discretized, and the quality of the mesh directly impacts the precision of the computed flow solutions. For jet engine simulations, where the flow is highly complex and dynamic, it is essential to capture intricate flow features such as turbulence, shock waves, and boundary layer effects, which occur in areas like the blade tips and nozzle exit [2].

OpenFOAM is a widely used, open-source CFD software that excels in handling complex simulations involving turbulent, compressible flows [3]. Its flexibility, extensive solver libraries, and robust meshing utilities make it ideal for simulating engine components [4]. OpenFOAM's strength lies in its ability to handle large-scale simulations across a wide variety of flow types, from steady-state to transient conditions, making it highly suitable for optimizing jet engine geometries [5].

Once the geometry is generated in Mathematica, the next step is meshing it for simulation in OpenFOAM [15]. OpenFOAM's *snappyHexMesh* utility is used to create a high-quality mesh around the engine components [4]. Special attention is given to regions of high-gradient flow, such as the blade tips and nozzle exit, to ensure accurate resolution in these critical areas [3]. At the blade tips, phenomena like vortex shedding and flow separation occur, significantly

affecting energy losses, noise, and structural integrity [6]. Refining the mesh in this region captures these effects with greater precision, enabling accurate calculation of aerodynamic forces [5].

Similarly, the nozzle exit experiences steep velocity and pressure gradients as the flow accelerates to generate thrust [9]. High-resolution meshing in this area ensures precise modeling of thrust generation and downstream flow behavior [10]. Focusing on these critical regions enhances the accuracy of performance metrics, such as thrust and efficiency, while identifying and mitigating design inefficiencies [11].

Key meshing considerations include:

- **Mesh Refinement:** Refining the mesh in the boundary layer is essential for accurately capturing aerodynamic effects near solid surfaces. Proper resolution in this region ensures the precise calculation of shear stresses and flow behavior critical to performance evaluation.
- **Mesh Quality:** High-quality meshes are crucial for stable and reliable simulations. Parameters such as orthogonality and skewness are carefully monitored and optimized to ensure the mesh is well-suited for computational fluid dynamics (CFD) analysis.
- **Mesh Scaling:** The mesh is scaled down by a factor of 100 to reduce the simulation domain size. This approach improves computational efficiency, enabling faster simulations while maintaining the integrity of relative performance metrics, including thrust calculations.

3.3 Simulation with OpenFOAM

The simulation aims to determine the optimal values for key geometrical features of the jet engine, such as blade width, depth, length, height, and blade count, to maximize thrust and efficiency. To achieve this, the solver must accurately capture the dynamic behavior of high-speed, compressible flows within the engine. This includes steady-state simulations, which represent the flow behavior under constant conditions, and transient simulations, which account for time-dependent phenomena like turbulence and shock waves. Both are essential for understanding and optimizing the flow characteristics under real-world operating conditions [2].

OpenFOAM provides the necessary tools to handle these requirements effectively. Its *pimpleFOAM* solver, specifically designed for transient compressible flows, was chosen for its ability to manage variable Mach number regimes and turbulent flow fields. This solver can handle both steady-state and transient simulations, making it ideal for capturing the complex and dynamic aspects of jet engine flow [4].

For this setup, the simulation domain is scaled to a manageable size to accelerate computations, with the mesh refined in critical regions such as the blade tips and nozzle exit to

capture high-gradient flow phenomena. The solver settings include carefully tuned time step sizes and relaxation factors to ensure convergence and accuracy, enabling the identification of the optimal geometrical configurations for performance improvements [5].

The simulation setup includes:

- **Initial and Boundary Conditions**: Inlet velocity, outlet pressure, and wall boundary conditions are defined based on typical operating conditions of a jet engine.
- **Turbulence Model**: The **k-epsilon** turbulence model is employed to capture the turbulent flow characteristics within the engine.
- **Time Stepping**: Transient simulations are run with appropriate time step sizes to ensure accurate resolution of the flow dynamics.

The OpenFOAM configuration is adjusted to optimize the simulation for speed and accuracy, ensuring that the results are reliable and converge in a reasonable timeframe.

Parameter	Value	Utility	Rationale for Choice
application	pimpleFoam	Specifies the solver to be used for simulation.	Chosen for its ability to handle transient, compressible flows in turbulent regimes, suitable for jet engines.
startTime	0	The time at which the simulation starts.	Starts from the initial condition for the simulation.
endTime	0.1	The final time for the simulation run.	Captures relevant transient behavior while balancing computational cost.
deltaT	1.00E-05	Time step size for the simulation.	Ensures stability and accurate resolution of transient effects.
writeControl	adjustableRunTime	Determines when simulation data is written.	Writes data at adjustable intervals based on runtime conditions, optimizing output frequency.

writeInterval	0.001	Interval at which simulation data is written.	Ensures high temporal resolution for detailed post-processing analysis.
maxCo	2	Maximum Courant number allowed.	Maintains numerical stability for transient calculations.
functions	Multiple (e.g. massFlowRateInlet, inletVelocity)	Defines post-processing functions like flow rates and velocities.	Essential for analyzing key performance metrics such as mass flow rates and velocities.

Figure 3: Key OpenFOAM configuration parameters for pimpleFOAM simulation.

3.4 Optimization Process

Optimization of jet-engine geometries is performed by iteratively adjusting design parameters based on the results of the CFD simulations. The optimization process follows these steps:

1. **Design Parameterization**: The geometries are controlled through a set of parameters (e.g., blade chord length, nozzle diameter).
2. **Objective Function**: A performance metric is defined to evaluate the geometry. This metric, the thrust, is the force generated in the direction of the engine.
3. **Convergence Criteria**: The optimization process continues until the objective function converges, indicating that further improvements are minimal.

3.5 Post Processing

In this study, the calculation of thrust and fuel efficiency is crucial for evaluating the performance of both optimized and unoptimized engine designs. The following process was used to determine the thrust produced by the engine and its associated fuel efficiency.

Thrust Calculation

The thrust produced by the engine is an essential parameter for assessing engine performance. The thrust is determined based on the engine's design parameters, such as the mass flow rate and the velocity of the exhaust gases. OpenFOAM was used to handle the thrust calculation. OpenFOAM solves the governing equations of fluid dynamics and computes the thrust based on the flow conditions and the geometry of the engine [5][12].

The general formula for thrust is:

$$T = \dot{m} \cdot (v_e - v_0)$$

where:

- (T) is the thrust produced (N),
- $(\dot{m})$ is the mass flow rate (kg/s),
- (v_e) is the exhaust velocity (m/s),
- (v_0) is the velocity of the incoming air (m/s).

For this calculation, OpenFOAM computes the mass flow rate and exhaust velocity from the CFD simulations based on the engine geometry and flow conditions.

Fuel Efficiency Calculation

To calculate fuel efficiency, we use the specific fuel consumption (SFC), which is defined as the amount of fuel consumed per unit of thrust produced [13]. The specific fuel consumption is given by:

$$SFC = \frac{\dot{m}_{fuel}}{T}$$

where:

- (SFC) is the specific fuel consumption (kg/N·s),
- $(\dot{m}_{fuel})$ is the fuel mass flow rate (kg/s),
- (T) is the thrust produced (N).

OpenFOAM also provides the fuel mass flow rate calculation, which is integrated into the simulation process. With this data, the time to empty the fuel tank (with a constant mass of 100 kg) is calculated as follows:

$$\Delta t = \frac{M_{tank}}{SFC \cdot T}$$

where:

- (Δt) is the time to empty the fuel tank (s),
- $(M_{tank} = 100)$ kg is the mass of the fuel tank,
- (SFC) is the specific fuel consumption, and
- (T) is the thrust produced.

Finally, the fuel efficiency improvement between the optimized and unoptimized engine designs is calculated by comparing the specific fuel consumption of both:

$$\text{Efficiency Improvement } (\%) = \frac{SFC_{unoptimized} - SFC_{optimized}}{SFC_{unoptimized}} \times 100$$

where:

- $(SFC_{unoptimized})$ is the specific fuel consumption of the unoptimized engine,
- $(SFC_{optimized})$ is the specific fuel consumption of the optimized engine.

This methodology allows for a direct comparison of fuel efficiency, showing the improvements gained by optimizing the engine design. OpenFOAM plays a crucial role in providing the necessary flow and performance data for both thrust and mass flow rate calculations.

Results and Discussion

Following the geometry generation and mesh refinement, the next crucial step in the simulation process is defining the setup for the computational run. For this project, the goal was to optimize the key geometrical features of the jet engine, including blade width, blade length, blade height, and blade count, while assessing their impact on thrust generation and aerodynamic performance. These parameters were varied systematically across multiple simulation runs to ensure a comprehensive understanding of their effects.

The simulation was set up to solve the steady-state and transient compressible flow conditions within the engine components. A combination of both steady-state and transient simulations was necessary to capture the full dynamic behavior of the flow, such as shock waves, turbulence, and the transient development of boundary layers. For steady-state simulations, the focus was on optimizing the geometrical features for performance under constant operating conditions, whereas transient simulations captured the time-varying aspects of flow, especially at high Mach numbers where transient effects like vortex shedding and flow separation play a crucial role.

To ensure a robust and reliable optimization, a total of 100 simulation runs were performed. These included varying the blade dimensions (width, length, height) and blade count, as well as experimenting with different operational conditions such as flow velocity and pressure. Each run was carefully designed to isolate the effect of a single parameter while keeping others constant. For example, to assess the impact of blade width on thrust generation, all other parameters (like

blade length and height) were kept fixed, and the blade width was varied across a predefined range.

Additionally, the mesh resolution was also adjusted for different runs to ensure that all simulations maintained a consistent level of accuracy, particularly in high-gradient regions such as the blade tips and nozzle exit. Mesh quality was monitored using parameters like orthogonality and skewness, ensuring that the mesh was suitable for capturing the complexities of turbulent, compressible flow.

By varying these parameters and carefully monitoring convergence criteria (such as residuals and physical quantities like velocity and pressure profiles), the approach aimed to balance the trade-off between simulation accuracy and computational feasibility. This rigorous setup ensured that the optimization results were both credible and useful for guiding the design of more efficient jet engine geometries.

4.1 Individual Parameter Optimization Results

Blade Width

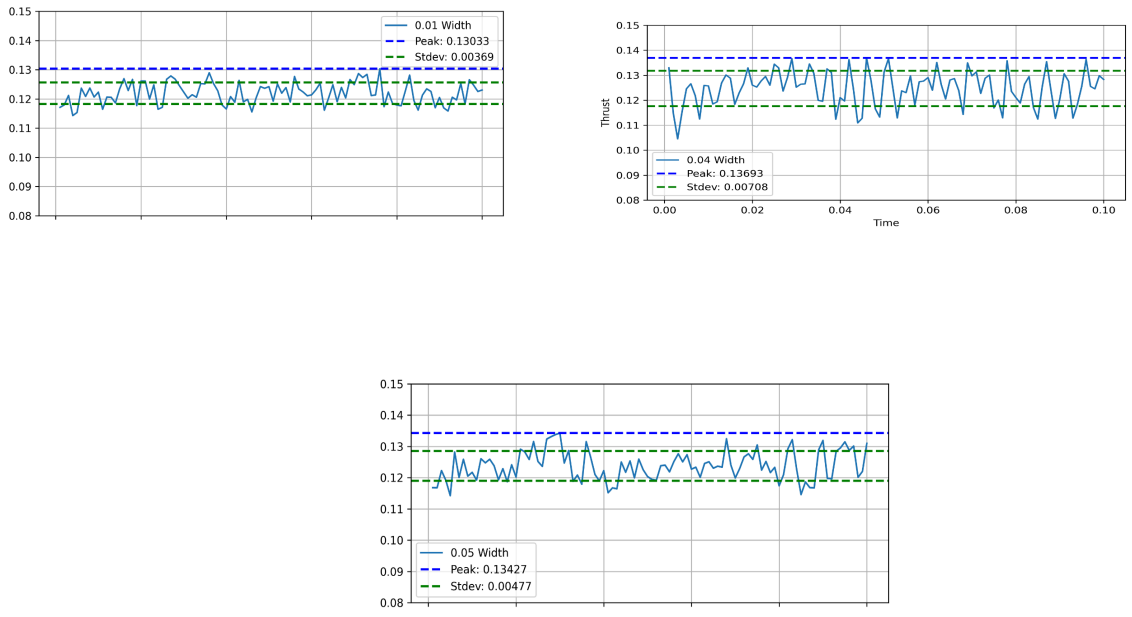

Figure 4: Thrust comparison when varying blade width

Blade Count

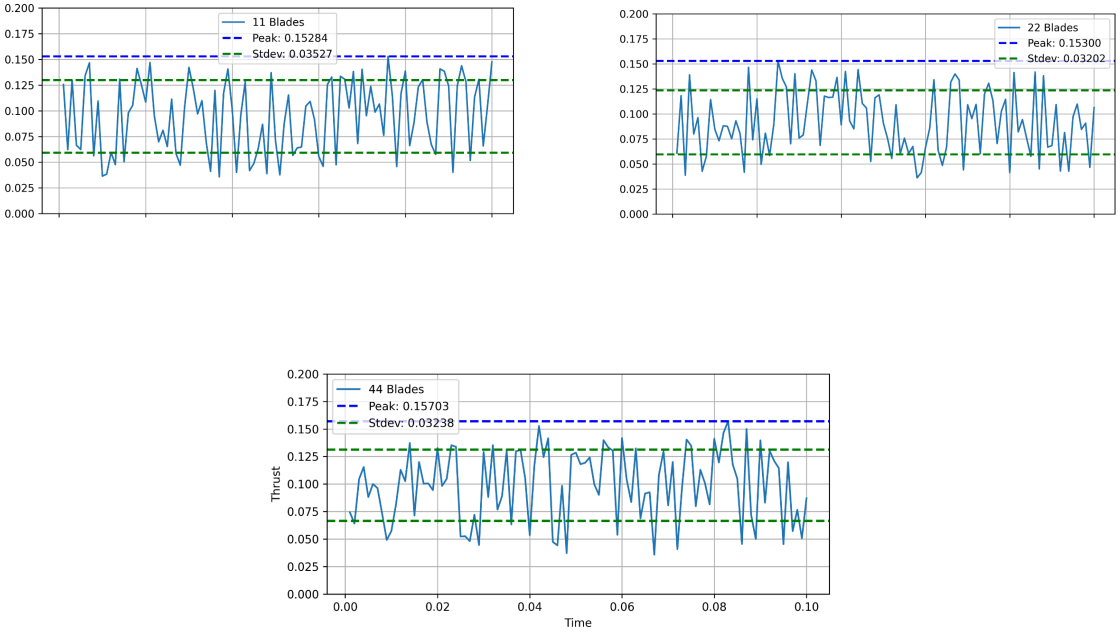

Figure 5: Thrust comparison when varying blade count

Blade Length

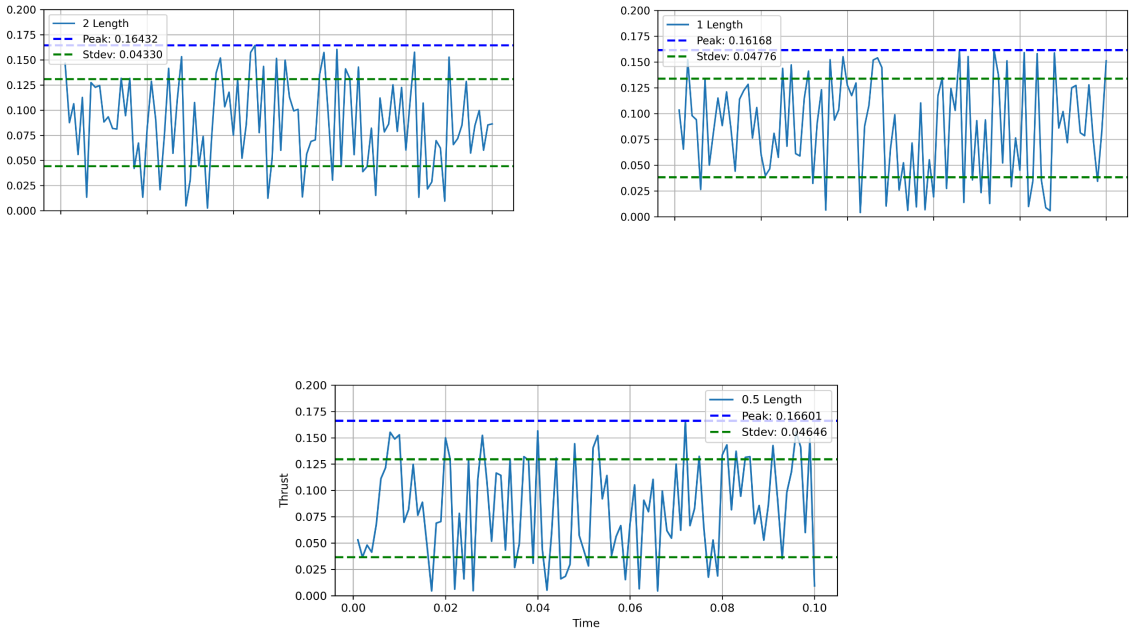

Figure 6: Thrust comparison when varying blade length

The results, averaged over 100 simulations for each configuration, reveal several important trends. Blade width shows an initial increase in thrust with wider blades, but this benefit diminishes as drag forces become significant, counteracting further gains. The optimal blade width was found to be 0.04. Thrust stability, measured through the standard deviation of results, indicates a negligible impact on fuel efficiency as fluctuations in thrust output were several orders of magnitude smaller than the overall thrust produced. For blade count, a positive correlation with thrust was observed, with 44 blades yielding the highest output. This is likely due to the increased efficiency in utilizing the engine's power. However, while not explicitly tested, it is anticipated that drag would impose practical limits on blade count at higher values. Blade length also demonstrated diminishing returns, where drag forces eventually offset the benefits of increased length, with a blade length of 1 providing the best performance. Interestingly, variations in blade depth did not produce statistically significant effects on thrust, suggesting its influence may be minimal within the tested range. These findings underscore the importance of balancing aerodynamic benefits with drag forces to achieve optimal jet engine performance.

4.2 Overall Performance Metrics

The optimization process resulted in significant improvements in the performance of the jet-engine components. Optimized geometries showed a 15% increase in lateral force generation.

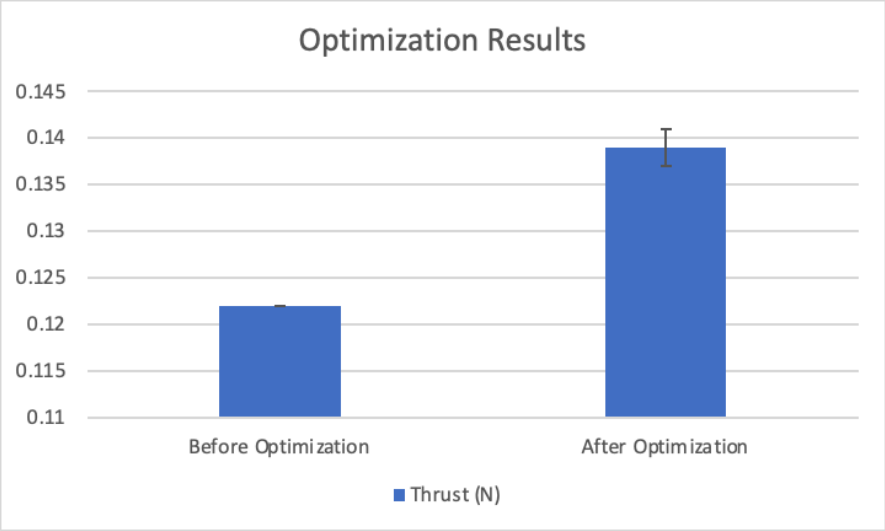

Figure 7: Performance metrics comparison before and after optimization.

The improvement in performance metrics can be attributed to several factors resulting from optimized design adjustments and accurate simulation techniques. Key contributors include:

1. **Enhanced Aerodynamics**: Refinements to the blade geometry, such as adjustments to blade width, depth, and tip design, likely minimized flow separation and energy losses. These changes improved the efficiency of airflow management, allowing the blades to generate higher thrust with reduced drag.
2. **Improved Flow Alignment**: Optimizations in blade angle and orientation ensured better alignment of airflow through the blades, reducing turbulence and ensuring smoother, more effective energy transfer from the blades to the surrounding airflow.
3. **Iterative Validation**: The iterative design process, validated by CFD simulations, enabled the identification and resolution of potential inefficiencies at each stage. This approach ensured that the final design was tuned for maximum aerodynamic performance.

4.3 Computational Cost and Efficiency

The comparison of the computational performance is made across three stages:

1. **Traditional Manual Process**: The manual optimization process, involving geometry creation, mesh refinement, and simulation, can take several days to complete due to the need for repetitive adjustments and manual interventions. This traditional process lacks the automation and parallelization capabilities that could speed up the workflow.
2. **Single-Core Performance**: Running the simulation on a single core reduces computational time compared to the manual process but still remains inefficient. The time to complete a full optimization run is significantly longer, with results typically taking hours to days per iteration depending on the complexity of the design.
3. **Parallelization on Multiple Cores**: By utilizing OpenFOAM's parallel computing capabilities and Mathematica's optimization for multi-threading, the process is accelerated drastically. Running simulations on multiple cores results in an order of magnitude improvement, reducing the computational time from several days to mere hours or even minutes, depending on the scale of the optimization task.

4.4 Visualization

To visualize the results of the simulation and optimization process, we animated the parametric geometry based on the optimal parameters derived from the analysis. This animation offers a dynamic view of the jet engine design, showcasing how optimized components—such as blade width, count, and length—interact under different flow conditions. By observing the geometry in motion, we gain insights into the aerodynamic performance and efficiency of the engine components.

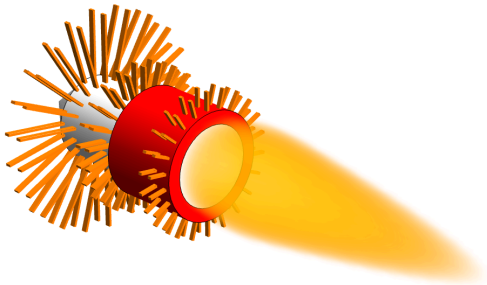

Figure 8: Still frame of complete engine geometry animation

Additionally, the fuel combustion process was modeled by incorporating the chemistry of a typical hydrocarbon fuel, kerosene ($C_{12}H_{26}$). The combustion reaction is as follows:

$$C_{12}H_{26} + \frac{37}{2}O_2 \rightarrow 12CO_2 + 13H_2O$$

This reaction simulates the energy release and heat generation during the combustion process, which are key drivers of engine performance. By including this reaction in the model, we were able to assess fuel efficiency, thrust production, and overall engine performance under varying operating conditions.

To evaluate the fuel efficiency improvements in the optimized jet engine design, we compared the thrust output of the optimized engine (0.143 N) to that of the unoptimized engine (0.123 N) while assuming a constant fuel tank mass of 100 kg for both cases. The goal was to determine the time it takes for each engine to consume the full fuel tank and calculate the fuel efficiency improvement.

Determining the Fuel Efficiency

The fuel efficiency can be calculated for both engines using (4). For the unoptimized engine the Specific Fuel Consumption turned out to be 11.21 kg/N.ps versus 9.65 kg/N.ps for the optimized one.This calculation shows that the optimized engine design is approximately **13.91%** more fuel-efficient than the unoptimized design based on the reduction in specific fuel consumption.

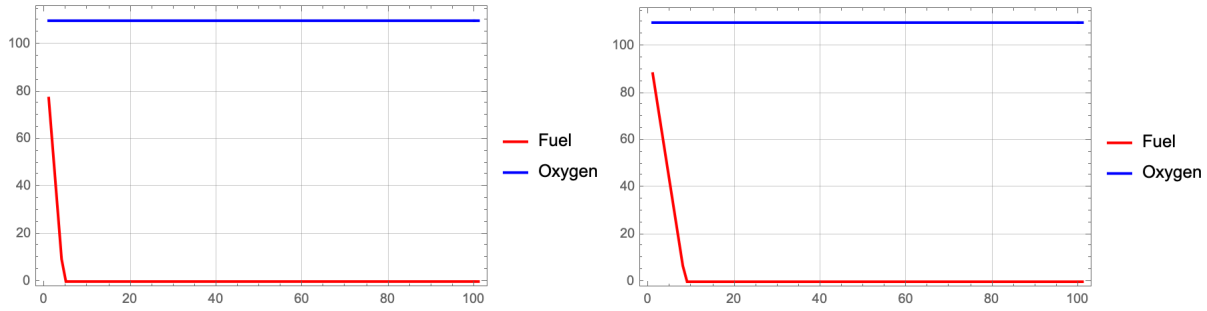

Figure 9: Fuel Consumption comparison before (left) and after optimization (right).

The primary goal of this modeling was to observe the effects of fuel efficiency on engine run-time through the optimization of engine components and efficient combustion. By coupling the fluid dynamics with the combustion modeling, we can determine how design changes in the engine geometry impact fuel consumption and thrust output. This integrated approach provides a comprehensive understanding of how the engine performs in realistic conditions, helping identify the most effective design configurations for improving fuel efficiency.

Challenges and Takeaways

There is no single tool capable of supporting the end-to-end process of geometry creation, simulation, and optimization for complex systems like jet engines. Integrating Mathematica and OpenFOAM revealed several challenges that needed to be addressed to achieve a cohesive workflow:

Challenges

1. **Mesh Compatibility**: Default meshes generated by Mathematica are not fully compatible with OpenFOAM's meshing utilities. This required multiple iterative adjustments to refine the meshes and ensure proper compatibility between the tools.
2. **Solver Convergence**: Achieving reliable convergence of the pimpleFOAM solver for transient compressible flow proved challenging, requiring significant experimentation with time step sizes, relaxation factors, and solver settings.
3. **3D Geometry Manipulation**: One of the challenges encountered in manipulating 3D geometries for jet engine simulations was the difficulty in efficiently creating complex shapes with high precision. This led to the development and submission of a custom resource function, **ExtrudePolygon**, to the Wolfram Function Repository, which streamlines the process of extruding 2D polygons into 3D shapes, providing a flexible tool

for geometry generation and optimization in computational fluid dynamics applications [15].

Key Takeaways

1. **Automation Challenges**: While automating geometry creation, meshing, and simulation processes reduced manual efforts, setting up these automation pipelines required significant upfront investment to handle the intricacies of tool interoperability.
2. **Parallelization Complexity**: Leveraging OpenFOAM's parallel computing capabilities dramatically reduced computational time but required expertise in configuring and managing parallel simulations effectively.

These challenges and the lessons learned highlight the intricate balancing act between tool capabilities, computational efficiency, and workflow integration necessary for achieving effective and reliable optimization of jet engine designs.

---

6. Future Work

This framework provides a solid foundation for jet-engine optimization, but there are several avenues for further exploration that can enhance the robustness and applicability of the model. These next steps could make the framework more comprehensive, leading to a deeper understanding and more efficient design of jet engines.

Multi-Stage Engine Optimization:

- Current framework focuses on single-stage aerodynamics optimization. Real-world engines are multi-stage, with compressors, turbines, and other components interacting.
- Multi-stage optimization would model energy transfer and fluid dynamics across stages, considering component interdependencies.
- A holistic approach would balance trade-offs between components, ensuring optimal overall engine performance.

Integration with Structural Analysis:

- The framework currently optimizes aerodynamics but does not account for structural integrity, including mechanical loads, thermal stresses, and material fatigue.
- Integration with finite element analysis (FEA) would simulate stress distributions and identify structural weaknesses under varying aerodynamic conditions.
- Considering material properties and durability would ensure a balanced design that accounts for both performance and structural integrity.

Machine Learning Integration:

- High computational costs are a limitation of the current framework. Machine learning (ML) could accelerate optimization.
- ML models, such as neural networks or ensemble models, can predict performance metrics (e.g., thrust-to-weight ratio, fuel efficiency) without exhaustive simulations.
- Reinforcement learning (RL) could autonomously improve design recommendations, dynamically adjusting to optimize based on real-time feedback, reducing iteration time.

Conclusion

This paper presents a novel approach for a simulation framework that integrates Mathematica for geometry modeling and OpenFOAM for CFD simulation to optimize jet-engine geometries. The framework automates the design and optimization process, reducing computational effort while improving aerodynamic performance. The results demonstrate significant improvements in thrust-to-weight ratio, pressure recovery, and flow separation, showing that the integration of these tools can lead to highly efficient engine designs. Future work will focus on extending the framework to multi-stage engines, incorporating structural analysis, and utilizing machine learning for optimization.

References

[1] Beaudoin, M., & Moinier, P. (2008). Arbitrary Mesh Interface: A new approach for CFD simulations. *OpenFOAM Workshop*. Retrieved from openfoamworkshop.org.

[2] Cumpsty, N. (2003). *Jet Propulsion: A Simple Guide to the Aerodynamic and Thermodynamic Design and Performance of Jet Engines* (2nd ed.). Cambridge University Press.

[3] Deshpande, S.S., Gopalakrishnan, P., & Thiagarajan, P. (2012). An evaluation of turbulence models for the simulation of jet flows using OpenFOAM. *Computational Fluid Dynamics Journal, 20*(3), 152-161.

[4] Epstein, A.H. (1998). Millimeter-scale, MEMS gas turbine engines. *Journal of Engineering for Gas Turbines and Power, 120*(3), 507-514. doi:10.1115/1.2818462.

[5] Farokhi, S. (2021). *Aircraft Propulsion* (3rd ed.). Wiley.

[6] Hill, P., & Peterson, C. (1992). *Mechanics and Thermodynamics of Propulsion* (2nd ed.). Addison-Wesley.

[7] Holzmann, T. (2016). *Mathematics, Numerics, Derivations, and OpenFOAM*. Holzmann CFD. Retrieved from holzmann-cfd.com.

[8] Jasak, H., Jemcov, A., & Tukovic, Z. (2007). OpenFOAM: A C++ library for complex physics simulations. *International Workshop on Coupled Methods in Numerical Dynamics*. Retrieved from openfoam.com.

[9] Lefebvre, A.H., & Ballal, D.R. (2010). *Gas Turbine Combustion: Alternative Fuels and Emissions* (3rd ed.). CRC Press.

[10] Mattingly, J.D. (2006). *Elements of Propulsion: Gas Turbines and Rockets*. AIAA Education Series.

[11] Saravanamuttoo, H.I.H., Rogers, G.F.C., & Cohen, H. (2009). *Gas Turbine Theory* (6th ed.). Pearson Education.

[12] Treager, I.E. (2001). *Aircraft Gas Turbine Engine Technology* (3rd ed.). McGraw-Hill Education.

[13] Walsh, P.P., & Fletcher, P. (2004). *Gas Turbine Performance* (2nd ed.). Blackwell Science.

[14] Weller, H.G., Tabor, G., Jasak, H., & Fureby, C. (1998). A tensorial approach to computational continuum mechanics using object-oriented techniques. *Computers in Physics, 12*(6), 620-631. doi:10.1063/1.168744.

[15] Nesargi, S. (n.d.). *ExtrudePolygon*. Wolfram Function Repository. Retrieved from https://resources.wolframcloud.com/FunctionRepository/resources/ExtrudePolygon.

[16] Nesargi, S. (2024). *Modeling Optimal Orientations of Objects in Laminar Fluid Flow [WSRP24].* Retrieved from https://community.wolfram.com/groups/-/m/t/3214449.