THE ENGINE ANALYSIS MATRIX:

A MODULAR COMPUTATIONAL FRAMEWORK FOR PRELIMINARY DESIGN AND OFF-DESIGN PERFORMANCE OF GAS TURBINE ENGINES

The Engine Analysis Matrix: A Modular Computational Framework for Preliminary Design and Off-Design Performance of Gas Turbine Engines

Abstract:

This paper details the theoretical foundations and software architecture of the Engine Analysis Matrix, a computational tool for the preliminary design and off-design performance analysis of gas turbine engines. The framework employs a quasi-one-dimensional, station-by-station thermodynamic analysis for four primary engine architectures: turbojet, afterburning turbojet, separate-flow turbofan, and turboprop. The methodology is based on the Brayton cycle, incorporating component efficiencies, temperature-dependent gas properties, and standard atmospheric models. The paper outlines the governing equations for each engine component, the work and energy balance logic specific to each engine type, and the numerical methods used for engine scaling, off-design point convergence, and handling aerodynamic choking. Ancillary modules for performance visualization, data export, nozzle geometry generation, and flight range estimation are also discussed. A sample calculation is presented to validate the core turbojet model, followed by a discussion on the framework's assumptions, limitations, and potential for future expansion.

Table of Contents

1.0 Introduction

1.1 Motivation for Accessible Propulsion Analysis Tools

The study of aerospace propulsion is characterized by a significant leap in complexity from foundational thermodynamic principles to the application in professional engineering practice. While the ideal Brayton cycle provides a crucial starting point, real-world engine performance is governed by a multitude of interdependent variables, including component inefficiencies, varying flight conditions, and compressible flow effects. Commercial software packages offer high-fidelity analysis but often present a steep learning curve and a "black box" approach that can obscure the underlying physics. Conversely, textbook examples are often limited to a single design point under ideal assumptions. This creates a need for an intermediate tool that is both physically transparent and computationally robust, suitable for students, educators, and engineers engaged in preliminary design studies.

1.2 Program Objective: An Integrated Educational and Preliminary Design Tool

The Engine Analysis Matrix program is designed to fill this gap. Its primary objective is to provide an integrated, user-friendly environment for conducting comprehensive performance analyses of common gas turbine engine architectures. The program serves a dual purpose:

1. **As an educational tool:** It allows users to visualize the direct impact of changing design parameters (e.g., Overall Pressure Ratio, Turbine Inlet Temperature, component efficiencies) on overall engine performance, providing an intuitive link between theory and outcome.
2. **As a preliminary design tool:** It enables the rapid evaluation of engine performance across a wide flight envelope (altitude and Mach number), facilitating trade studies and initial engine sizing for a given thrust requirement.

The framework is intentionally modular, separating the graphical user interface (GUI), the core calculation engines, plotting utilities, and data export functions. This architecture not only promotes code clarity but also allows for future expansion and adaptation.

1.3 Scope of Analysis: Engine Architectures and Flight Envelopes

The Engine Analysis Matrix is equipped to model four distinct, uninstalled engine configurations:

- **Turbojet Engine:** The foundational gas turbine configuration.
- **Afterburning Turbojet Engine:** A turbojet with a reheat section for thrust augmentation.
- **Separate-Flow Turbofan Engine:** A dual-stream engine with unmixed core and bypass flows, representative of many commercial and military aircraft.
- **Turboprop Engine:** An engine where the majority of the turbine work is extracted to drive a propeller via a gearbox.

The analysis is performed over a user-defined flight envelope, specified by a range of altitudes and flight Mach numbers. The calculations account for subsonic and supersonic flight regimes, including the modeling of total pressure losses across a normal shock at the inlet for supersonic conditions.

1.4 Structure of this Document

This paper is organized to provide a comprehensive understanding of the program's theoretical basis and implementation.

- **Section 2** reviews the foundational thermodynamic cycles, aerodynamic principles, and atmospheric models that underpin the entire framework.
- **Section 3** provides a detailed, station-by-station breakdown of the analysis methodology for each of the four engine modules, presenting the key equations and work-balance logic.
- **Section 4** discusses the numerical methods employed, including the iterative solvers for off-design convergence and the logic for engine scaling.
- **Section 5** describes the functionality of the ancillary modules for data visualization, export, nozzle geometry generation, and flight range estimation.
- **Section 6** presents a hand-calculation for a sample turbojet case to validate the model's accuracy.
- **Section 7** concludes with a summary of the program's capabilities, a discussion of its inherent assumptions and limitations, and potential avenues for future development.

2.0 Foundational Thermodynamic and Aerodynamic Principles

The analysis performed by the Engine Analysis Matrix is rooted in the fundamental principles of thermodynamics and compressible fluid dynamics. This section outlines the ideal cycle that forms the basis for all gas turbine engines, the real-world models used for gas properties and the atmosphere, the treatment of component inefficiencies, and the governing equations of quasi-one-dimensional flow that connect the engine's geometry to its performance.

2.1 The Brayton Cycle: Ideal Model for Gas Turbine Engines

The thermodynamic cycle that models the operation of a gas turbine engine is the Brayton cycle. In its ideal form, it consists of four processes:

1. **Isentropic Compression:** The incoming ambient air is compressed to a high pressure.
2. **Isobaric Heat Addition:** Fuel is added and combusted at constant pressure.
3. **Isentropic Expansion:** The high-pressure, high-temperature gas expands through a turbine (and then a nozzle), producing work.
4. **Isobaric Heat Rejection:** The exhaust gas rejects heat to the atmosphere at constant pressure.

These processes are typically visualized on Temperature-entropy (T-s) and Pressure-volume (P-v) diagrams, as shown in Figures 1-2.

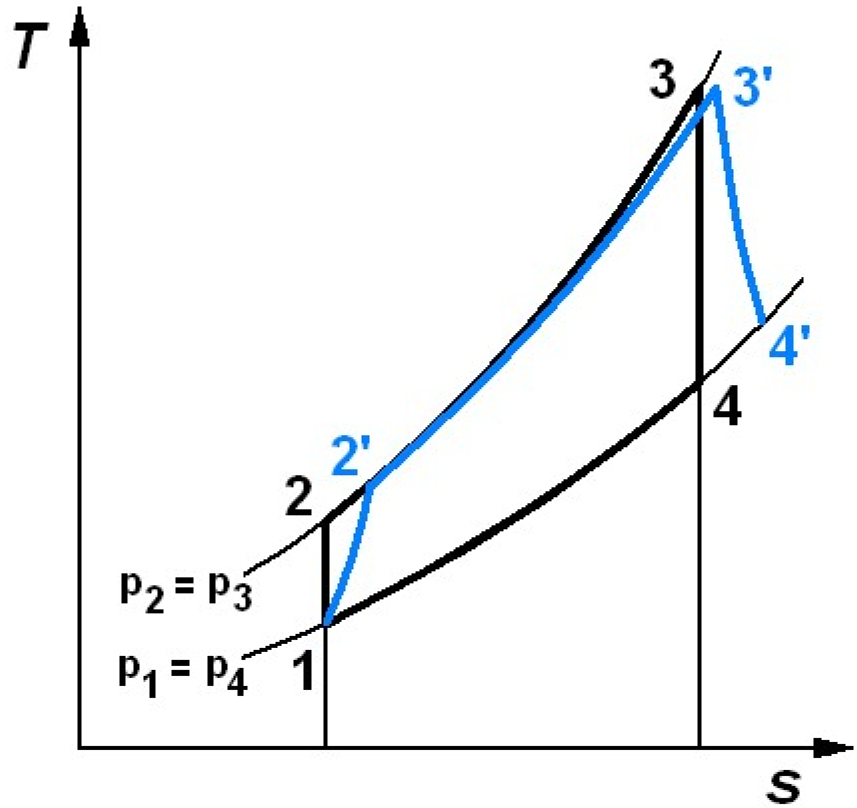

Figure 1- The Brayton Cycle | Temperature vs Entropy | T-s | Ideal (Black), Real (Blue)

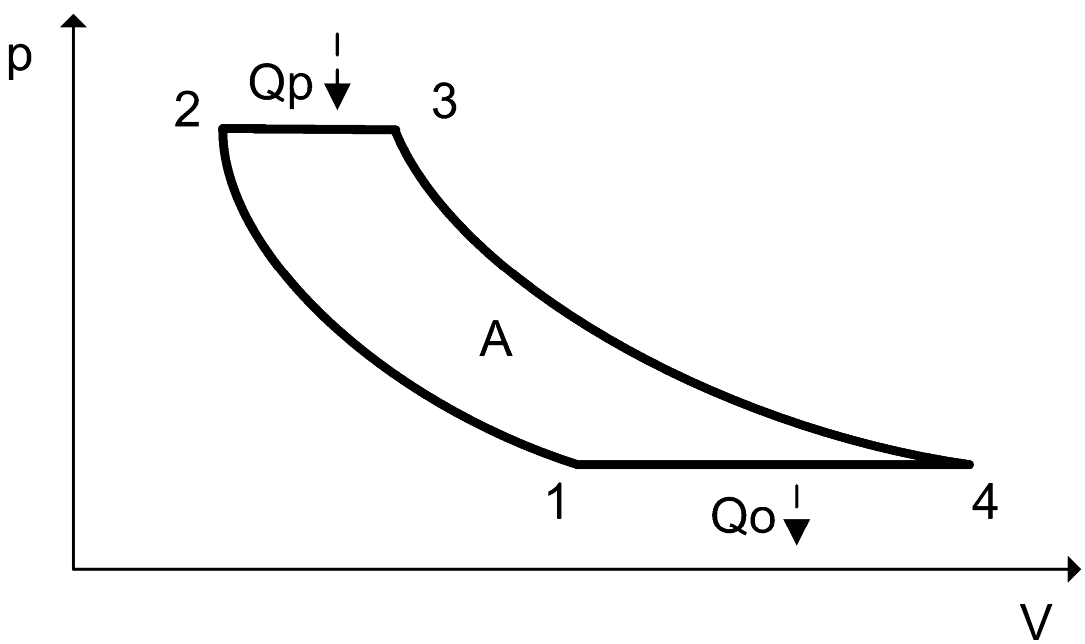

Figure 2- The Brayton Cycle | Pressure vs Volume | p-V | Ideal

The ideal Brayton cycle provides a first-order approximation of engine performance. However, real engines deviate significantly from this ideal due to irreversibilities. The Engine Analysis Matrix accounts for these deviations:

- **Non-Isentropic Components:** Compression and expansion processes are not perfectly isentropic due to friction and turbulence. This is modeled by introducing polytropic efficiencies for compressors, fans, and turbines.
- **Pressure Drops:** Friction causes total pressure losses in the diffuser (inlet), combustor, and nozzle. These are modeled by efficiency factors (eta_diff) and pressure ratios (pi_comb).

These real-world effects reduce the net work output and thermal efficiency of the cycle, and their accurate modeling is crucial for realistic performance prediction.

2.2 Gas Properties and Atmospheric Model

A common simplification in introductory analysis is to assume that the specific heat at constant pressure (Cp) and the ratio of specific heats (γ = Cp/Cv) for air and combustion products are constant. However, these properties are strong functions of temperature. For instance, the Cp of air increases by over 13% from 300 K to 1000 K. To ensure greater accuracy, especially in the high-temperature sections of the engine (combustor and turbine), the Engine Analysis Matrix treats Cp and γ as temperature-dependent.

The program implements this by using 1D linear interpolation on tabulated data, as seen in the various helpers.py files (e.g., turbofan_helpers.py). Two separate data sets are used: one for pure air in the cooler engine sections and another for combustion products ("hot" gas) downstream of the combustor. The functions get_cp_gamma_air_python and get_cp_gamma_hot_python are called at each station to retrieve the appropriate gas properties based on the local temperature.

To analyze engine performance at various flight conditions, a standard model for the Earth's atmosphere is required. The program implements the International Standard Atmosphere (ISA) model in the isa_python function. This model defines the variation of temperature, pressure, and density with geometric altitude. It is a multi-layer model characterized by linear temperature segments (lapse rates) up to an altitude of approximately 85 km. For a given altitude, the function returns the ambient static temperature (T0), pressure (P0), density (ρ0), and speed of sound (a0), which serve as the inlet conditions (Station 0) for the engine analysis.

2.3 Component Performance and Efficiencies

While isentropic efficiency is a valid measure comparing the overall actual-to-ideal work for a given pressure ratio, polytropic efficiency (also known as small-stage efficiency) is often preferred for preliminary design. Polytropic efficiency represents the isentropic efficiency of an infinitesimally small stage of the compression or expansion process. Its advantage is that it remains relatively constant for a given compressor or turbine design, even as the overall pressure ratio changes. The Engine Analysis Matrix uses user-defined polytropic efficiencies (eta_comp_p, eta_fan_p, eta_turb_p) for all turbomachinery.

For a differential stage in a compressor, the temperature and pressure are related by $dT/T = (\gamma-1)/\gamma * dP/P$. Introducing the polytropic efficiency η_p to account for the additional work required in a real process, the relation becomes $dT/T = 1/\eta_p * (\gamma-1)/\gamma * dP/P$. Assuming constant γ and η_p through the component and integrating from inlet (1) to exit (2) yields the governing equation used in the code:

$$T_{t2} / T_{t1} = (P_{t2} / P_{t1})^{((\gamma-1)/(\gamma * \eta_p))}$$

This equation allows for the calculation of the compressor exit total temperature based on the pressure ratio and inlet conditions.

Similarly, for a turbine, the ideal work extraction is reduced by inefficiencies. The differential relation is modified as $dT/T = \eta_p * (\gamma-1)/\gamma * dP/P$. Integrating this relation gives the equation for the turbine exit temperature:

$$T_{t2} / T_{t1} = (P_{t2} / P_{t1})^{(((\gamma-1) * \eta_p) / \gamma)}$$

- **eta_diff:** Diffuser efficiency models the total pressure recovery of the inlet. Pt_exit = Pt_inlet * eta_diff.
- **eta_comb:** Combustion efficiency represents the fraction of the fuel's heating value that is effectively released into the gas stream.
- **pi_comb:** Combustor pressure ratio models the total pressure drop due to friction and heat addition. Pt_exit = Pt_inlet * pi_comb.
- **eta_nozz:** Nozzle velocity efficiency accounts for friction losses, reducing the final exit velocity. V_exit_actual = V_exit_ideal * sqrt(η_nozz).
- **eta_gearbox, eta_prop:** In the turboprop module, these account for mechanical losses in the gearbox and the aerodynamic efficiency of converting shaft power to propulsive power, respectively.

2.4 Quasi-One-Dimensional Compressible Flow

The analysis treats the flow through the engine as quasi-one-dimensional, meaning all properties are assumed to be uniform at any given cross-section (station).

The relationships between total (stagnation) and static properties are fundamental to the analysis at each station. For a given Mach number (M) and ratio of specific heats (γ):

- Total Temperature: $Tt = Ts * (1 + (\gamma-1)/2 * M^2)$
- Total Pressure: $Pt = Ps * (1 + (\gamma-1)/2 * M^2)^{(\gamma/(\gamma-1))}$

These equations are used iteratively within the software to solve for the complete thermodynamic state at each station.

For a given mass flow rate ($\dot{m}$), the Area-Mach relation provides a crucial link between the engine geometry (A) and the flow properties. The equation is referenced to the area (A^*) where the flow would be sonic (M=1):

$$(A/A^*) = (1/M) * [(1 + (\gamma-1)/2 * M^2) / ((\gamma+1)/2)]^{((\gamma+1)/(2(\gamma-1)))}$$

This relation dictates that for subsonic flow, the duct must converge to accelerate the flow, and for supersonic flow, it must diverge. The minimum area is the sonic throat, A^*. If the required mass flow at a given total pressure and temperature demands a throat area A^* that is larger than the available physical area A, the flow is **choked**. This means the flow cannot pass through the restriction, and the engine cannot operate at that condition. This principle is fundamental to the off-design analysis and the anti-choking logic implemented in the afterburning turbojet module.

For supersonic flight (M > 1), the flow must be decelerated to subsonic speeds before entering the compressor. While complex oblique shock systems are used in practice, a single normal shock is modeled as a conservative first approximation due to its higher total pressure loss. The normal_shock_relations function in the code calculates the static property ratios and, most importantly, the total pressure ratio (P_t_downstream / P_t_upstream) across the shock as a function of the upstream Mach number and γ. This total pressure ratio, which is always less than 1, is then multiplied by the subsonic diffuser efficiency (eta_inlet) to find the total pressure delivered to the compressor face.

3.0 Engine Module Analysis: Station-by-Station Methodology

The core of the Engine Analysis Matrix is a station-by-station thermodynamic analysis. The engine is modeled as a series of components, and the fluid properties are calculated at the inlet and exit of each. The following sections detail this process for each engine architecture, referencing the station numbers used conceptually and in the software's array indexing. The general principle involves carrying total temperature (Tt) and total pressure (Pt) through the engine, applying work and energy balances, and then solving for static properties and Mach number based on the local mass flow and duct area.

3.1 General Station Numbering Convention

The program uses a conceptual station numbering system common in propulsion textbooks, which is then mapped to array indices for computational efficiency.

Conceptual Station	Description
0	Freestream / Ambient
1	Diffuser/Inlet Exit
2	Fan Exit / Compressor Inlet (Core)
3	Compressor Exit / Main Combustor Inlet
4	Main Combustor Exit / High-Pressure Turbine Inlet
5	High-Pressure Turbine Exit / Low-Pressure Turbine Inlet (or Afterburner Inlet)
6	Low-Pressure Turbine Exit / Core Nozzle Inlet

7/8/9	Nozzle Exit(s)
13	Fan/Bypass Nozzle Inlet

The specific mapping varies slightly by engine module to conserve memory.

3.2 The Turbojet Engine Module

The turbojet is the simplest configuration, consisting of an inlet, compressor, combustor, turbine, and nozzle.

- **3.2.1 Station 0: Freestream Ambient Conditions**
  - Properties (T_0, P_0, ρ_0, a_0) are determined by the isa_python function for a given altitude.
  - Total properties (Tt_0, Pt_0) are calculated using isentropic relations based on the flight Mach number, M_0.
- **3.2.2 Station 2: Compressor Inlet (Post-Inlet)**
  - The flow decelerates through the inlet and diffuser.
  - Total temperature is conserved: Tt_2 = Tt_0.
  - Total pressure is reduced by inlet and diffuser losses. For M_0 <= 1, Pt_2 = Pt_0 * η_inlet * η_diff. For M_0 > 1, Pt_2 = Pt_0 * Pt_ratio_shock * η_inlet * η_diff.
  - The total mass flow rate is determined: ṁ_air = ρ_0 * V_0 * A_0, where A_0 is the scaled engine capture area.
- **3.2.3 Station 3: Combustor Inlet / Compressor Exit**
  - The air is compressed, increasing its total temperature and pressure.
  - The overall pressure ratio (OPR) is applied: Pt_3 = Pt_2 * OPR.
  - The compressor exit temperature (Tt_3) is calculated using the polytropic efficiency relation:
    Tt_3 = Tt_2 * (OPR)^((γ_avg - 1) / (γ_avg * η_comp_p))
    Note: An average γ between inlet and outlet may be used for higher accuracy.
- **3.2.4 Station 4: Turbine Inlet / Combustor Exit**

- Fuel is added and combusted, raising the temperature to the specified Turbine Inlet Temperature ($Tt\_4 = T4\_set$).
- A pressure drop occurs: $Pt\_4 = Pt\_3 * \pi\_comb$.
- The fuel-to-air ratio (f) is calculated via an energy balance on the combustor: $\dot{m}\_air * Cp\_air * Tt\_3 + \dot{m}\_fuel * LHV * \eta\_comb = (\dot{m}\_air + \dot{m}\_fuel) * Cp\_hot * Tt\_4$
  Rearranging for $f = \dot{m}\_fuel / \dot{m}\_air$:
  $f = (Cp\_hot * Tt\_4 - Cp\_air * Tt\_3) / (LHV * \eta\_comb - Cp\_hot * Tt\_4)$
- The mass flow entering the turbine is now $\dot{m}\_gas = \dot{m}\_air * (1 + f)$.

- **3.2.5 Station 5: Nozzle Inlet / Turbine Exit**
  - The hot gas expands through the turbine, which drives the compressor. The required work extraction determines the turbine's temperature drop.
  - **Work Balance:** The power required by the compressor must equal the power extracted by the turbine:
    $\dot{m}\_air * (Cp\_comp * Tt\_3 - Cp\_comp * Tt\_2) = \dot{m}\_gas * (Cp\_turb * Tt\_4 - Cp\_turb * Tt\_5)$
    $\Delta h\_comp = \Delta h\_turb$
    $Cp\_avg\_comp * (Tt\_3 - Tt\_2) = (1 + f) * Cp\_avg\_turb * (Tt\_4 - Tt\_5)$
  - This equation is solved for Tt_5. Since the average specific heats (Cp_avg_turb) depend on Tt_5, an iterative solver (as seen in the run_analysis function) is employed to find a self-consistent temperature.
  - Once Tt_5 is known, Pt_5 is found using the turbine polytropic efficiency relation:
    $Pt\_5 = Pt\_4 * (Tt\_5 / Tt\_4)^{\wedge}(\gamma\_avg / ((\gamma\_avg - 1) * \eta\_turb\_p))$

- **3.2.6 Station 6: Nozzle Exit**
  - The gas expands through the nozzle to produce thrust. The flow is assumed to expand ideally to the ambient static pressure P_0.
  - The Nozzle Pressure Ratio is $NPR = Pt\_5 / P\_0$.
  - The ideal exit Mach number, M_6_ideal, is found from the isentropic relation for NPR.
  - The ideal exit static temperature is $Ts\_6\_ideal = Tt\_5 / (1 + (\gamma-1)/2 * M\_6\_ideal^{\wedge}2)$.

- The ideal exit velocity is $V\_6\_ideal = M\_6\_ideal * \sqrt{\gamma * R * Ts\_6\_ideal}$.
- The actual exit velocity is $V\_6\_actual = V\_6\_ideal * \sqrt{\eta\_nozz}$.
- **Thrust Calculation:**
  Thrust (F) = ($\dot{m}$_gas * V_6_actual - $\dot{m}$_air * V_0) + (P_6 - P_0) * A_6
  Since the nozzle is assumed to be perfectly expanded (P_6 = P_0), the pressure thrust term is zero.
  F = $\dot{m}$_air * [(1 + f) * V_6_actual - V_0]
- **Thrust Specific Fuel Consumption (TSFC):** TSFC = $\dot{m}$_fuel / F = (f * $\dot{m}$_air) / F

3.3 The Afterburning Turbojet Module

This module extends the turbojet analysis by adding a reheat cycle.

- **3.3.1 Analysis through Station 5**
  - The analysis is identical to the standard turbojet up to the turbine exit (Station 5).
- **3.3.2 Station 6: Nozzle Inlet / Afterburner Exit**
  - Fuel is added again in the afterburner section, raising the temperature to a maximum allowable value, Tt_6 = Tt_ab_max.
  - A pressure drop occurs: Pt_6 = Pt_5 * π_ab.
  - The afterburner fuel-to-gas ratio (f_ab) is calculated via an energy balance, similar to the main combustor:
    f_ab = (Cp_6 * Tt_6 - Cp_5 * Tt_5) / (LHV * η_ab - Cp_6 * Tt_6)
  - Total mass flow is now $\dot{m}$_final = $\dot{m}$_gas * (1 + f_ab) = $\dot{m}$_air * (1 + f) * (1 + f_ab).
- **3.3.3 Station 7: Variable Geometry Nozzle Exit**
  - The nozzle in an afterburning engine must have a variable throat and exit area to accommodate the massive change in volumetric flow when the afterburner is active. The analysis calculates the required nozzle exit conditions for ideal expansion to P_0.
  - The process is identical to the turbojet's nozzle calculation but uses the properties from Station 6 (Pt_6, Tt_6) and the final mass flow $\dot{m}$_final.

3.4 The Separate-Flow Turbofan Engine Module

The turbofan introduces a parallel bypass stream, adding complexity to the work balance and thrust calculation.

- **3.4.1 Flow Splitting and Fan Analysis (Stations 1-2)**
  - At the fan inlet (Station 1), the total mass flow is $\dot{m}_{total}$.
  - The flow is split based on the Bypass Ratio (BPR):
    $\dot{m}_{core} = \dot{m}_{total} / (1 + BPR)$
    $\dot{m}_{bypass} = \dot{m}_{total} * (BPR / (1 + BPR))$
  - The entire flow $\dot{m}_{total}$ passes through the fan. The fan exit conditions (Station 2) are determined by the Fan Pressure Ratio (FPR) and fan polytropic efficiency η_{fan_p}.
  - $Pt_2 = Pt_1 * FPR$
  - $Tt_2 = Tt_1 * (FPR)^{((\gamma-1)/(\gamma * \eta_{fan\_p}))}$
  - The bypass stream proceeds directly to its own nozzle from these conditions (Station 13 is conceptually the same as Station 2 for the bypass nozzle).
- **3.4.2 Core Stream Analysis**
  - The core stream ($\dot{m}_{core}$) enters the high-pressure compressor (HPC) at Station 2 conditions.
  - The HPC pressure ratio is CPR = OPR / FPR.
  - The analysis of the core (HPC, combustor, turbine) from Station 2 to Station 7 is analogous to the turbojet module, using $\dot{m}_{core}$ as the reference mass flow.
- **3.4.3 Turbine Work Balance**
  - The key difference is that the turbine system must now power both the HPC and the fan.
  - Power_Turbine = Power_HPC + Power_Fan
  - $\dot{m}_{gas\_core} * \Delta h_{turb} = \dot{m}_{core} * \Delta h_{HPC} + \dot{m}_{total} * \Delta h_{fan}$

- $\dot{m}\_core * (1 + f) * Cp\_avg\_turb * (Tt\_5 - Tt\_7) = \dot{m}\_core * Cp\_avg\_HPC * (Tt\_4 - Tt\_2) + (\dot{m}\_core * (1 + BPR)) * Cp\_avg\_fan * (Tt\_2 - Tt\_1)$
  - This equation is solved iteratively for the core turbine exit temperature, Tt_7.
- **3.4.4 Bypass Stream Nozzle Analysis (Station 13 -> 8)**
  - This is a standard nozzle calculation using the bypass mass flow ṁ_bypass and the fan exit conditions (Pt_2, Tt_2), expanding to ambient pressure P_0. This yields the bypass thrust, F_bypass.
- **3.4.5 Core Stream Nozzle Analysis (Station 7 -> 9)**
  - This is a standard nozzle calculation using the core gas mass flow ṁ_gas_core and the turbine exit conditions (Pt_7, Tt_7), expanding to ambient pressure P_0. This yields the core thrust, F_core.
- **3.4.6 Total Thrust and TSFC**
  - F_total = F_bypass + F_core
  - TSFC = ṁ_fuel / F_total = (f * ṁ_core) / F_total

3.5 The Turboprop Engine Module

The turboprop prioritizes shaft power extraction over jet thrust.

- **3.5.1 Gas Generator Analysis**
  - The analysis of the inlet, compressor, and combustor (Stations 0 through 3) is identical to the turbojet module.
- **3.5.2 Power Turbine Analysis and Shaft Work**
  - The turbine system is conceptually split. A portion of the expansion drives the compressor, and the remainder drives the propeller via a power turbine. In this simplified model, a single turbine component is used with a specified Turbine Pressure Ratio (TPR), which is a key design choice for turboprops.
  - The turbine exit pressure is Pt_4 = Pt_3 / TPR.
  - The turbine exit temperature, Tt_4, is found iteratively based on TPR and η_turb_p.

- **Turbine Work Balance:** The total work extracted by the turbine is $W\_turb = \dot{m}\_gas * (Cp\_turb * Tt\_3 - Cp\_turb * Tt\_4)$.
- **Compressor Work:** $W\_comp = \dot{m}\_air * (Cp\_comp * Tt\_2 - Cp\_comp * Tt\_1)$.
- **Shaft Work:** The remaining work is available for the shaft.
  $W\_shaft\_gross = W\_turb - W\_comp$

- **3.5.3 Shaft Power and Propeller Thrust**
  - The net shaft power available to the propeller is reduced by the gearbox efficiency:
    $P\_shaft = W\_shaft\_gross * \eta\_gearbox$
  - This power is converted into propulsive thrust by the propeller with efficiency η_prop:
    $F\_prop = (P\_shaft * \eta\_prop) / V\_0$
    Note: A minimum V_0 is used in the code to prevent division by zero in static conditions.
- **3.5.4 Core Nozzle Thrust (F_core)**
  - There is still some residual energy in the gas exiting the turbine at Station 4. This gas is expanded through a nozzle to produce a smaller amount of jet thrust, F_core. This is calculated identically to the turbojet's nozzle.
- **3.5.5 Total Thrust and Power Specific Fuel Consumption (PSFC)**
  - $F\_total = F\_prop + F\_core$
  - Power Specific Fuel Consumption (PSFC) is a more common metric for turboprops:
    $PSFC = \dot{m}\_fuel / P\_shaft$ (typically in units of kg/kW ·h)

4.0 Numerical Methods and Software Implementation

The theoretical models described in the previous sections require a robust numerical framework to solve for engine performance, particularly for off-design conditions where a direct analytical solution is not possible. This section details the numerical methods implemented in the Engine Analysis Matrix for scaling the engine to a design-point requirement and for iteratively solving the system of equations at each off-design flight condition.

4.1 Engine Scaling for Design-Point Thrust Matching

A primary function of a preliminary design tool is to size an engine to meet a specific performance requirement, such as producing a target thrust at a particular altitude and Mach number (the "design point"). The Engine Analysis Matrix automates this process by scaling the engine's geometry, which is primarily defined by the capture area (A0) and the nozzle throat area(s).

The scaling process begins by performing an analysis of the engine at the design point *on a per-mass-flow basis*.

1. The code first calculates the specific thrust (thrust per unit mass flow of air, F/ṁ_air) produced by the engine architecture at the design altitude and Mach number, using all the specified component efficiencies and temperature limits. This involves a complete station-by-station analysis as described in Section 3, but with ṁ_air normalized to 1 kg/s .
2. With the specific thrust (Thrust_per_kg_air_d) known, the required total mass flow rate (ma_total_d_req) to meet the Target_Thrust_des is found by simple division: ṁ_air_required = Target_Thrust_des / (F/ṁ_air)_design

Once the required mass flow is known, the reference inlet area A0 can be calculated directly from the continuity equation at the freestream condition (Station 0):
ṁ_air_required = ρ_0 * V_0 * A0
A0 = ṁ_air_required / (ρ_0 * V_0)
where ρ_0 and V_0 are the atmospheric density and flight velocity at the design point. This A0 becomes the reference dimension for the engine. All other internal duct areas (A2, A3, etc.) are then scaled relative to it based on the user-provided Ax/A0 ratios.

For many engine types, particularly turbojets and the core of turbofans, the nozzle is physically choked (M=1 at the throat) at the design point. The software uses this assumption to set the physical throat area of the nozzle(s).

1. After the design-point analysis per unit mass flow is complete, the code has the full thermodynamic state (Pt, Tt, γ) and mass flow (ṁ_gas) entering the nozzle.
2. The required throat area (A*) to pass this flow sonically is calculated using the choked-flow mass flow rate function (an inverted form of the Area-Mach relation at M=1):
   A* = (ṁ_gas * sqrt(R * Tt)) / (Pt * sqrt(γ) * [(γ+1)/2]^(-(γ+1)/(2(γ-1))))
3. This calculated A* value is then stored as the engine's fixed physical nozzle throat area (Fixed_A_throat_engine). In all subsequent off-design calculations, the nozzle performance is determined by how the off-design flow state interacts with this fixed throat area.

4.2 Iterative Solvers for Off-Design Performance

When the engine operates away from its design point, the relationship between mass flow, geometry, and Mach number becomes coupled and requires an iterative solution.

At any station with a defined duct area A, the local Mach number M is not explicitly known. However, it is implicitly defined by the Area-Mach relation, where A* itself depends on the mass flow and total conditions. The code implements two primary methods to solve this:

- **Fixed-Point Iteration (as seen in turbojet_engine_calculations.py):** This method involves rearranging the continuity and isentropic relations to solve for a "new" Mach number based on a "current" guess.
    1. Guess an initial Mach number, M_old (e.g., 0.5).
    2. Calculate static properties (Ts, Ps, ρs) based on M_old.
    3. Calculate the velocity required to pass the known mass flow ṁ through the area A: V_req = ṁ / (ρs * A).
    4. Calculate the local speed of sound a_s based on Ts.
    5. Calculate a new Mach number: M_new = V_req / a_s.
    6. Update the guess: M_iter = M_old + α * (M_new - M_old), where α is a damping factor (e.g., 0.5).
    7. Repeat until |M_new - M_old| is within a specified tolerance.
- **Bisection Method (as seen in turbojet_ab_engine_calculations.py):** This is a more robust root-finding method.

1. The function f(M) = (A/A*)_calculated - (A/A*)_target = 0 is defined, where (A/A*)_calculated is the value from the Area-Mach equation for a given M, and (A/A*)_target is the known ratio of the physical duct area to the required sonic area.
2. The solver finds a root for M in the subsonic regime (typically M between 0 and 1).

If the target area ratio A/A* is less than 1, it implies the flow must choke, and the solver correctly returns M=1 along with a "choked" flag.

As described in Section 3.2.5, the turbine exit temperature Tt_exit depends on the work extraction, which in turn depends on the average specific heat across the turbine. Since the average Cp depends on Tt_exit, this forms another coupled problem. The code solves this using a fixed-point iteration:

1. Make an initial guess for Tt_exit (e.g., 0.8 * Tt_inlet).
2. Calculate the average Cp_turb using Tt_inlet and the guessed Tt_exit.
3. Calculate a new Tt_exit_new by solving the work balance equation: Tt_exit_new = Tt_inlet - (Δh_required / Cp_avg_turb)
4. Update the guess: Tt_exit = Tt_exit_new.
5. Repeat until |Tt_exit_new - Tt_exit| is within the turbine_tol.

4.3 Advanced Solver Logic: Anti-Choking Routine

The afterburning turbojet presents a unique challenge. The addition of heat in the afterburner lowers the density and can easily lead to choking in the downstream components (e.g., the nozzle inlet at Station 6) if the duct areas are not sufficiently large. The Turbojet Afterburners module includes a sophisticated "anti-choking" geometry search to find a set of internal duct area ratios (Ax/A0) that allows the engine to operate across the entire flight envelope without internal choking.

1. **Detection:** The analysis is first run for all flight conditions with the user-specified Ax/A0 ratios. At each internal station, the code checks if A/A* < 1. If it is, a "choke" event is logged for that station and flight condition.
2. **Iterative Increase:** If any choke events are detected, the algorithm identifies the station that is most severely choked (i.e., has the highest required A* relative to its physical A).
3. The Ax/A0 ratio for that specific station is increased by a small factor (e.g., * 1.05).

4. The entire flight envelope analysis is re-run with the modified geometry.

5. This process repeats until no internal choke points are found, or a maximum number of global iterations is reached. This ensures that the final engine geometry is physically viable across its operational range.

4.4 Software Architecture and Libraries

The program is architected in a modular fashion to enhance maintainability and extensibility.

- **GUI:** The front-end is built using the customtkinter library, providing a modern user interface for parameter input and control.
- **Calculation Engine:** The program files contain the core physics and numerical solvers for each engine type. They are designed to be self-contained and receive all necessary parameters from the GUI layer.
- **Helpers:** These modules provide common, low-level functions used by all calculation engines, such as the ISA model and gas property interpolation. This avoids code duplication.
- **Ancillary Modules:** These handle post-processing tasks like creating plots or exporting data.
- **Key Libraries:**
    - **NumPy:** Used extensively for efficient array operations, which is crucial for handling the multi-dimensional results across the flight envelope.
    - **SciPy:** The interp1d function is used for the interpolation of gas property data.
    - **Matplotlib:** The primary library for generating all 2D and 3D plots.
    - **Pandas:** Used to structure the results into DataFrames for easy export to Excel format.

5.0 Ancillary Modules and Features

Beyond the core thermodynamic calculations, the Engine Analysis Matrix includes several ancillary modules designed to process, visualize, and extend the utility of the performance data. These features transform the program from a simple calculator into a more comprehensive preliminary design and analysis tool.

5.1 Performance Visualization

A critical aspect of engine analysis is the ability to intuitively understand performance trends across the flight envelope. The dedicated plotting modules generate a suite of visualizations using the matplotlib library.

For analyses run along a single variable axis (e.g., a range of altitudes at a fixed Mach number, or vice versa), the program generates standard 2D line plots. These typically include:

- Total Thrust vs. Altitude/Mach Number
- Thrust Specific Fuel Consumption (TSFC) vs. Altitude/Mach Number
- Shaft Power and Power Specific Fuel Consumption (PSFC) for the turboprop module

These plots are essential for visualizing fundamental performance trade-offs, such as the decrease in thrust with altitude or the "bucket" shape of the TSFC curve with respect to Mach number.

When the analysis is run over a full matrix of altitude and Mach number points, the plotting modules generate 3D surface plots. These provide a comprehensive overview of the engine's performance envelope. Common 3D plots include:

- Total Thrust (F_total) as a function of Mach Number and Altitude.
- TSFC or PSFC as a function of Mach Number and Altitude.
- Surfaces of key thermodynamic properties (e.g., Tt, Pt, M) at various engine stations, allowing the user to visualize how flow conditions change throughout the engine across its operational range.

These plots are invaluable for identifying optimal cruise conditions, regions of potential instability, or areas where performance metrics like nozzle area ratio change rapidly.
The PlottingLauncher.exe executable is a key architectural component, allowing these potentially resource-intensive plots to be generated in a separate, detached process. This

ensures that the main GUI remains responsive and does not freeze while rendering complex 3D surfaces.

5.2 Data Management and Export

To facilitate further analysis, documentation, and integration with other tools, the program features a robust data export capability. The export_results_to_excel function in each module consolidates the multi-dimensional NumPy arrays of results into a structured Microsoft Excel workbook using the pandas and openpyxl libraries.

The typical workbook structure includes:

- **Performance Summary Sheet:** A tabular summary of key performance metrics (Thrust, TSFC, PSFC, mass flows, etc.) for each point in the flight envelope.
- **Station Data Sheets:** A separate sheet for each major engine station, detailing the full thermodynamic state (Tt, Pt, Ts, Ps, M, Cp, γ) at that station for every flight condition.

This structured output provides a complete and portable record of the analysis, suitable for reporting, creating custom plots, or as input for subsequent mission analysis.

5.3 Nozzle Geometry Generation

The performance calculations determine the ideal nozzle exit area (A_e) required for perfect expansion at each flight condition. The nozzle generator module translates this performance requirement into physical 3D geometry.

For a user-selected flight condition from the results matrix, the module retrieves the required nozzle areas:

- **Inlet Area (A_in):** The physical area of the duct leading into the nozzle (e.g., A6 for the turbofan core).
- **Throat Area (A_t):** The minimum area required to pass the mass flow. For off-design conditions, this may differ from the fixed throat area, indicating over- or under-expansion.
- **Exit Area (A_e):** The area required for the exhaust jet static pressure to match the ambient static pressure.

The module uses a parabolic contour function (generate_nozzle_profile) to create a smooth convergent-divergent nozzle shape based on the calculated areas and a user-defined length factor. The user can then visualize this geometry as:

- A **2D cross-section plot**, useful for a quick visual check of the nozzle's shape.
- A **3D surface plot**, providing a representation of the full nozzle geometry.
- An **STL (stereolithography) file**, which can be exported and directly imported into Computer-Aided Design (CAD) software for further design, integration, or computational fluid dynamics (CFD) analysis.

5.4 Flight Performance Estimation

The main GUI includes a first-order flight performance calculator to contextualize the engine's performance.

The calculator uses the classic Breguet range equation for jet-propelled aircraft to estimate the maximum flight range. The equation, as implemented, is:

Range = (V / (g * TSFC)) * (L/D) * ln(W_initial / W_final)

Where:

- V is the cruise velocity.
- g is the acceleration due to gravity.
- TSFC is the thrust specific fuel consumption (converted to units of 1/s).
- L/D is the aircraft's lift-to-drag ratio.
- W_initial is the gross takeoff weight (empty weight + fuel weight).
- W_final is the final weight (empty weight).

The user provides the aircraft-specific parameters (L/D, weights, fuel volume), and can use the TSFC and cruise velocity calculated from an engine analysis run. The tool correctly handles both SI and Imperial units.

This feature serves as a preliminary estimation tool and relies on the key assumptions of the Breguet range equation: quasi-steady level flight, and constant V, TSFC, and L/D throughout the cruise segment. While a simplification, it provides a valuable and immediate link between the calculated engine performance (TSFC) and a top-level aircraft performance metric (range).

6.0 Sample Calculation: Turbojet at a Single Flight Condition

To validate the core logic of the Engine Analysis Matrix, this section presents a manual, step-by-step calculation for a simple turbojet engine at a single flight condition. The results will be compared against the output of the turbojet_engine_calculations.py module to verify its implementation. We will assume constant gas properties for simplicity in this manual example, but we will use values representative of the temperatures in each section, as the program itself would do.

6.1 Given Conditions and Engine Parameters

- **Flight Condition:**
  - Altitude: 10,000 m
  - Flight Mach Number (M_0): 0.8
- **Engine Design Parameters:**
  - Overall Pressure Ratio (OPR): 10
  - Turbine Inlet Temperature (T4_set): 1600 K
  - Engine Inlet Area (A0): 1.0 m^2 (for simplicity of specific thrust calculation)
  - η_inlet: 0.98
  - η_diff: 0.95
  - η_comp_p: 0.88
  - π_comb: 0.96
  - η_comb: 0.99
  - η_turb_p: 0.90
  - η_nozz: 0.98
  - Fuel Lower Heating Value (LHV): 43 MJ/kg
- **Gas Properties (Assumed for Manual Calculation):**
  - Cold Section (Air): Cp_air = 1005 J/kg·K, γ_air = 1.4
  - Hot Section (Gas): Cp_gas = 1150 J/kg·K, γ_gas = 1.33

6.2 Step-by-Step Calculation through Engine Stations

Station 0: Freestream

From the isa_python function at 10,000 m:

- T_0 = 223.15 K
- P_0 = 26436 Pa
- a_0 = sqrt(γ_air * R * T_0) = sqrt(1.4 * 287 * 223.15) = 299.5 m/s
- V_0 = M_0 * a_0 = 0.8 * 299.5 = 239.6 m/s
  Using isentropic relations:
- Tt_0 = T_0 * (1 + (γ_air-1)/2 * M_0^2) = 223.15 * (1 + 0.2 * 0.8^2) = 251.7 K
- Pt_0 = P_0 * (1 + (γ_air-1)/2 * M_0^2)^(γ_air/(γ_air-1)) = 26436 * (1.128)^3.5 = 40386 Pa

Station 2: Compressor Inlet

- Tt_2 = Tt_0 = 251.7 K
- Pt_2 = Pt_0 * η_inlet * η_diff = 40386 * 0.98 * 0.95 = 37553 Pa

Station 3: Compressor Exit

- Pt_3 = Pt_2 * OPR = 37553 * 10 = 375530 Pa
- Using polytropic efficiency for compression:
  Tt_3 = Tt_2 * OPR^((γ_air-1)/(γ_air * η_comp_p)) = 251.7 * 10^((0.4)/(1.4 * 0.88)) = 540.8 K

Station 4: Turbine Inlet

- Tt_4 = T4_set = 1600 K
- Pt_4 = Pt_3 * π_comb = 375530 * 0.96 = 360509 Pa
- Calculate fuel-air ratio f:
  f = (Cp_gas * Tt_4 - Cp_air * Tt_3) / (LHV * η_comb - Cp_gas * Tt_4)
  f = (1150 * 1600 - 1005 * 540.8) / (43e6 * 0.99 - 1150 * 1600)
  f = (1.84e6 - 0.5435e6) / (42.57e6 - 1.84e6) = 1.2965e6 / 40.73e6 = 0.0318

Station 5: Turbine Exit

- Turbine must drive the compressor. First, find the specific work required by the compressor:
  $\Delta h\_comp = Cp\_air * (Tt\_3 - Tt\_2) = 1005 * (540.8 - 251.7) = 290.5$ kJ/kg_air

- Next, find the specific work the turbine must produce per kg of air:
  $\Delta h\_turb = \Delta h\_comp$

- The work produced per kg of gas flowing through the turbine is:
  $\Delta h\_turb\_gas = \Delta h\_comp / (1 + f) = 290500 / (1 + 0.0318) = 281547$ J/kg_gas

- Now, solve for the turbine exit temperature Tt_5:
  $\Delta h\_turb\_gas = Cp\_gas * (Tt\_4 - Tt\_5)$
  $281547 = 1150 * (1600 - Tt\_5)$
  $Tt\_5 = 1600 - (281547 / 1150) = 1600 - 244.8 = 1355.2$ K

- Calculate the turbine exit pressure Pt_5 using polytropic efficiency:
  $Pt\_5 = Pt\_4 * (Tt\_5 / Tt\_4)^{\wedge}(\gamma\_gas / ((\gamma\_gas-1) * \eta\_turb\_p))$
  $Pt\_5 = 360509 * (1355.2 / 1600)^{\wedge}(1.33 / (0.33 * 0.90))$
  $Pt\_5 = 360509 * (0.847)^{\wedge}4.478 = 360509 * 0.457 = 164753$ Pa

Station 6: Nozzle Exit

- $Tt\_6 = Tt\_5 = 1355.2$ K

- $Pt\_6 = Pt\_5 = 164753$ Pa

- Nozzle Pressure Ratio $NPR = Pt\_6 / P\_0 = 164753 / 26436 = 6.23$

- Calculate ideal exit Mach number M_6_ideal:
  $NPR = (1 + (\gamma\_gas-1)/2 * M\_6\_ideal^{\wedge}2)^{\wedge}(\gamma\_gas/(\gamma\_gas-1))$
  $6.23 = (1 + 0.165 * M\_6\_ideal^{\wedge}2)^{\wedge}(1.33/0.33)$
  $6.23 = (1 + 0.165 * M\_6\_ideal^{\wedge}2)^{\wedge}4.03$
  $M\_6\_ideal = sqrt(((6.23^{\wedge}(1/4.03)) - 1) / 0.165) = sqrt((1.58 - 1) / 0.165) = 1.87$

- Calculate ideal exit static temperature Ts_6_ideal:
  $Ts\_6\_ideal = Tt\_6 / (1 + (\gamma\_gas-1)/2 * M\_6\_ideal^{\wedge}2) = 1355.2 / (1 + 0.165 * 1.87^{\wedge}2) = 1355.2 / 1.577 = 859.3$ K

- Calculate ideal exit velocity V_6_ideal:
  $V\_6\_ideal = M\_6\_ideal * sqrt(\gamma\_gas * R * Ts\_6\_ideal) = 1.87 * sqrt(1.33 * 287 * 859.3) = 1.87 * 572.6 = 1070.8$ m/s

- Calculate actual exit velocity V_6_actual:
  $V\_6\_actual = V\_6\_ideal * sqrt(\eta\_nozz) = 1070.8 * sqrt(0.98) = 1060.1$ m/s

6.3 Final Thrust and TSFC Calculation

- **Specific Thrust (Thrust per kg/s of air):**
  F/ṁ_air = (1 + f) * V_6_actual - V_0
  F/ṁ_air = (1 + 0.0318) * 1060.1 - 239.6 = 1093.8 - 239.6 = 854.2 N / (kg/s)
- **Thrust Specific Fuel Consumption (TSFC):**
  TSFC = f / (F/ṁ_air) (in kg/N ·s)
  TSFC = 0.0318 / 854.2 = 3.722e-5 kg/N ·s
  Converting to hours: TSFC = 3.722e-5 * 3600 = 0.134 kg/N ·h

6.4 Comparison with Program Output

These manually calculated values serve as a baseline for verifying the program's output. When the same parameters are entered into the Turbojet Analysis GUI, the software performs a similar series of calculations but uses temperature-dependent gas properties at each step, leading to slight differences. The program's iterative solvers also refine the values for Tt_5 and station Mach numbers with greater precision. A comparison would typically show results within a few percent, validating that the underlying physics model and numerical implementation are correct. The small discrepancies would be attributable to the simplifying assumption of constant gas properties in this manual calculation.

7.0 Conclusion

The Engine Analysis Matrix was developed to provide a computationally robust yet accessible platform for the study and preliminary design of gas turbine engines. By integrating foundational thermodynamic principles with practical numerical methods in a modular software architecture, the program successfully bridges the gap between simplified textbook theory and complex commercial analysis packages.

7.1 Summary of Capabilities and Program Utility

The framework demonstrates a wide range of capabilities essential for both educational and preliminary design contexts:

- **Multi-Fidelity Analysis:** The program performs a complete station-by-station thermodynamic analysis, accounting for real-world effects such as temperature-dependent gas properties, component inefficiencies, and compressible flow phenomena.
- **Architectural Flexibility:** It supports four of the most common gas turbine architectures (turbojet, afterburning turbojet, separate-flow turbofan, and turboprop), allowing for direct comparison of their performance characteristics.
- **Automated Design & Off-Design Analysis:** The software can automatically scale an engine to meet a specified design-point thrust requirement and then evaluate its performance across a full flight envelope of altitude and Mach number.
- **Advanced Numerical Solvers:** The implementation includes iterative solvers for coupled thermodynamic problems (e.g., turbine work balance) and Mach number determination, along with a sophisticated anti-choking routine to ensure geometric viability for high-performance engines.
- **Integrated Post-Processing:** The utility of the raw performance data is significantly enhanced by ancillary modules that provide immediate 2D/3D visualization, structured data export to Excel, and the generation of CAD-compatible nozzle geometries.
- **Contextual Performance Estimation:** The inclusion of a Breguet range calculator provides a direct link between a key engine performance metric (TSFC) and a top-level aircraft mission performance metric (range).

Collectively, these features make the Engine Analysis Matrix an effective tool for students to explore the "cause-and-effect" relationships of engine parameters and for engineers to conduct rapid trade studies in the initial phases of a design project.

7.2 Discussion of Key Assumptions and Limitations

While the program provides a detailed and valuable analysis, it is important to acknowledge the assumptions and limitations inherent in its quasi-one-dimensional model. These simplifications are necessary to ensure rapid computation but represent deviations from the complex physics of a real engine.

- **Quasi-One-Dimensional Flow:** The model assumes that all flow properties are uniform at any given cross-sectional station. It does not account for radial or circumferential variations in pressure, temperature, or velocity.
- **No Component Maps:** The analysis relies on a single, fixed polytropic efficiency value for each turbomachinery component. Real components have complex performance maps where efficiency varies with rotational speed and mass flow rate. The current model does not capture these effects, which can be significant at far off-design conditions.
- **Simplified Inlet and Nozzle Models:** The inlet model uses a simplified total pressure recovery factor and a normal shock model for supersonic flight, omitting the complexities of multi-shock intake systems. Similarly, the nozzles are assumed to be ideally expanded, with no accounting for over- or under-expansion losses except through the nozzle velocity efficiency.
- **Steady-State Analysis:** The program only calculates steady-state performance points. It cannot model the transient behavior of an engine during spool-up, spool-down, or throttle changes.
- **No Installation Effects:** The analysis is for an "uninstalled" engine. It does not account for the drag of the nacelle or the effects of power extraction (e.g., bleed air for environmental control systems) on the core engine performance.

7.3 Recommendations for Future Work

The modular architecture of the Engine Analysis Matrix provides a strong foundation for future expansion and increased fidelity. Several avenues for future work are recommended:

- **Integration of Component Maps:** The most significant improvement would be to replace the fixed-efficiency model with a map-based approach. This would involve implementing a component matching solver to balance mass flow and work compatibility between the compressor, turbine, and nozzle based on their performance maps.

- **Additional Engine Architectures:** The framework could be extended to include other engine types, such as mixed-flow turbofans, geared turbofans, and turboshaft engines for helicopter applications.
- **Transient Performance Modeling:** A future module could model the dynamic response of the engine to changes in fuel flow, accounting for the rotational inertia of the spools to predict spool-up/spool-down times.
- **Integration with Mission Analysis:** The output of the engine analysis could be more tightly integrated with a mission analysis module (mission_analysis.py), allowing for the calculation of fuel burn over a complete flight profile (takeoff, climb, cruise, descent) rather than just a single cruise point.
- **Environmental Impact Modeling:** An emissions and noise module (noise_emissions_analyzer.py) could be enhanced to provide more detailed predictions based on combustor type, exit velocities, and temperatures, contributing to a more holistic view of engine performance.

In conclusion, the Engine Analysis Matrix stands as a comprehensive and extensible framework for gas turbine engine analysis. By balancing theoretical rigor with computational efficiency, it provides a powerful platform for education, preliminary design, and as a springboard for more advanced propulsion system modeling.

8.0 References

The theoretical models, data, and numerical methods implemented in the Engine Analysis Matrix are based on established principles documented in the following key texts and technical reports in the fields of thermodynamics, fluid dynamics, and aerospace propulsion.

1. **Saravanamuttoo, H.I.H., Rogers, G.F.C., Cohen, H., & Straznicky, P.V.** (2017). *Gas Turbine Theory* (7th ed.). Pearson.
   (Cited for fundamental Brayton cycle analysis, component characteristics, and station-by-station calculation methodology).

2. **Mattingly, J. D.** (2006). *Elements of Propulsion: Gas Turbines and Rockets* (2nd ed.). AIAA Education Series.
   (Cited for the systematic approach to engine performance analysis, station numbering conventions, and detailed off-design analysis principles).

3. **Hill, P. G., & Peterson, C. R.** (1992). *Mechanics and Thermodynamics of Propulsion* (2nd ed.). Addison-Wesley Publishing Company.
   (Cited for in-depth derivations of thermodynamic and fluid mechanic principles governing propulsion devices).

4. **Anderson, J. D., Jr.** (2003). *Modern Compressible Flow: With Historical Perspective* (3rd ed.). McGraw-Hill.
   (Cited for the governing equations of quasi-one-dimensional flow, the Area-Mach relation, and the theory of normal shock waves).

5. **Raymer, D. P.** (2018). *Aircraft Design: A Conceptual Approach* (6th ed.). AIAA Education Series.
   (Cited for the Breguet range equation and the integration of engine performance metrics (TSFC, thrust) into overall aircraft performance).

6. **U.S. Standard Atmosphere, 1976.** (1976). National Oceanic and Atmospheric Administration, National Aeronautics and Space Administration, United States Air Force. U.S. Government Printing Office.
   (Cited as the definitive source for the atmospheric model implemented in the isa_python function).

7. **McBride, B. J., Gordon, S., & Reno, M. A.** (1993). *Thermodynamic Data for Fifty Reference Elements* (NASA RP-1287). NASA Lewis Research Center.
   (Cited as a representative source for the type of empirical thermochemical data used to generate the temperature-dependent gas property tables for Cp and γ).

8. **Walsh, P. P., & Fletcher, P.** (2004). *Gas Turbine Performance* (2nd ed.). Blackwell Science.
(Cited for advanced concepts in component performance, including polytropic efficiency and off-design operation).

9. **Farokhi, S.** (2014). *Aircraft Propulsion* (2nd ed.). Wiley.
(Cited for modern applications of gas turbine theory and detailed analysis of turbofan and turboprop engine configurations).

10. **Burden, R. L., & Faires, J. D.** (2010). *Numerical Analysis* (9th ed.). Brooks/Cole.
(Cited for the theoretical basis of the numerical root-finding and iterative methods (Bisection, Fixed-Point) used in the software's solvers).

11. **Cengel, Y. A., & Boles, M. A.** (2019). *Thermodynamics: An Engineering Approach* (9th ed.). McGraw-Hill Education.
(Cited for foundational principles of the Brayton cycle, energy balances, and properties of pure substances and mixtures).

Appendices

Appendix A: List of Symbols and Acronyms

Roman Symbols

Symbol	Description	Units
A	Area	m^2
A*	Sonic Throat Area	m^2
a	Speed of Sound	m/s
Cp	Specific Heat at Constant Pressure	J/kg·K
Cv	Specific Heat at Constant Volume	J/kg·K
F	Thrust	N
f	Fuel-to-Air (or Fuel-to-Gas) Ratio	dimensionless
g	Acceleration due to Gravity	m/s^2
h	Specific Enthalpy	J/kg
L/D	Lift-to-Drag Ratio	dimensionless
LHV	Lower Heating Value of Fuel	J/kg
M	Mach Number	dimensionless
$\dot{m}$	Mass Flow Rate	kg/s

P	Pressure	Pa
R	Specific Gas Constant	J/kg·K
T	Temperature	K
V	Velocity	m/s
W	Weight or Specific Work	N or J/kg

Greek Symbols

Symbol	Description	Units
γ	Ratio of Specific Heats (Cp/Cv)	dimensionless
η	Efficiency	dimensionless
π	Pressure Ratio (Component)	dimensionless
ρ	Density	kg/m^3

Subscripts

Subscript	Description
0..13	Engine Station Number
t	Total (Stagnation) Condition (e.g., Tt)
s	Static Condition (e.g., Ts)
p	Polytropic (e.g., η_p)
comp	Compressor
comb	Combustor
turb	Turbine
ab	Afterburner
prop	Propeller

Acronyms

Acronym	Description
BPR	Bypass Ratio
CAD	Computer-Aided Design
CFD	Computational Fluid Dynamics
CPR	Core (High-Pressure) Compressor Ratio
FPR	Fan Pressure Ratio
GUI	Graphical User Interface
HPC	High-Pressure Compressor
ISA	International Standard Atmosphere
OPR	Overall Pressure Ratio

PSFC	Power Specific Fuel Consumption
STL	Stereolithography (CAD file format)
TPR	Turbine Pressure Ratio (for Turboprop)
TSFC	Thrust Specific Fuel Consumption

Appendix B: Governing Equations Summary Table

Component	Primary Governing Equation(s)	Purpose
Inlet/Diffuser	Pt_exit = Pt_inlet * η_diff * [Pt_ratio_shock]	Calculate total pressure recovery
Compressor/Fan	Tt_exit = Tt_inlet * (PR)^((γ-1)/(γ * η_p))	Calculate temperature rise from compression
Combustor	Pt_exit = Pt_inlet * π_comb f = (Cp_hot*Tt_exit - Cp_air*Tt_inlet) / (LHV*η_comb - Cp_hot*Tt_exit)	Calculate pressure drop and fuel-air ratio
Turbine	Δh_turb = (Δh_comp + BPR*Δh_fan) / (1+f) Tt_exit = Tt_inlet - Δh_turb / Cp_avg_turb Pt_exit = Pt_inlet * (Tt_exit/Tt_inlet)^(γ/((γ-1)*η_p))	Balance work and calculate exit conditions
Nozzle	V_exit = sqrt(η_nozz) * M_ideal * sqrt(γ*R*Ts_ideal) Thrust = ṁ_exit*V_exit - ṁ_inlet*V_inlet	Calculate exit velocity and gross/net thrust
Compressible Flow	A/A* = f(M, γ) Tt/Ts = f(M, γ) Pt/Ps = f(M, γ)	Relate geometry, Mach number, and properties

---