# ADVANCED AIRCRAFT PERFORMANCE ANALYSES

**Marc Immer**
ALR Aerospace
Gotthardstrasse 52
CH-8002 Zürich
marc.immer@alr-aerospace.ch

**Philipp Juretzko**
ALR Aerospace
Gotthardstrasse 52
CH-8002 Zürich
philipp.juretzko@alr-aerospace.ch

**Abstract.** A part of the preliminary aircraft design process is the evaluation of the design mission performance. This paper presents a combination of a scripting environment (Python) with a point-mass, segment based integration method C++ code (Aircraft Performance Program APP). By using the example of a CAS-Mach climb schedule for a commercial airliner, the possibilities for rapid parameter analyses are demonstrated and the influence of the choice for the top-of-climb condition is highlighted. These capabilities of the framework are further shown by computing a complex design mission for a small propeller UAV usable for railway inspection. Overall mission profile optimization is achieved by a combination of a Python based optimizer and the APP built-in optimizer. Furthermore, the Python-APP framework is capable of computing and analyzing mission performance for hybrid-electric propulsion aircraft.

**Keywords**. Aircraft Performance, Mission Analysis, Electric Propulsion

## 1 Introduction

In preliminary aircraft design, a certain aircraft is sought that fulfills defined requirements. Usually, such requirements contain flight performance targets, e.g. payload, range endurance and flight speed. After each iteration in the design process that produces a viable design, the flight performance is checked against the performance requirements. If the result is not satisfactory, the design is changed and re-evaluated.

The preliminary aircraft design process comprises multiple engineering disciplines involving e.g. aerodynamics, propulsion, structure, loads, stability & control and systems. As the design matures, the methods employed change from simple handbook methods (analytic or statistical nature) to more complex computer based simulations (e.g. CFD, FEM). Past and current European research efforts such as SIMSAC[1] and AGILE[2] focus on integrating the tools of the different disciplines and fidelity into a common software environment, such as CEASIOM and the Reconfigurable Computing Environment (RCE). The need for a common file format to exchange information between tools was identified in [1], and an XML schema was proposed (CPACS[3]). More complex methods increase the fidelity of the model, and therefore influence the applicability and suitability of methods for the computation of aircraft performance.

Aircraft performance can be divided into three principal categories: field performance, point-performance and mission-performance. The first category, field performance, includes take-off, rejected take-off, landing and balanced field length computations. These computations are complex,

[1] http://www.simsacdesign.org
[2] http://www.agile-project.eu
[3] https://github.com/DLR-LY/CPACS

not only due to the kinematics but also due to certification requirements that have to be take into account. Field performance can usually be considered separately from other performance evaluations, as it makes up a smaller fraction of the overall flight (with respect to time, distance and fuel consumption). The second category, point-performance, describes all computations that only depend on the state of the aircraft and not the time evolution. Examples of these include maneuvering performance (e.g. turn-rate, load factor) or Specific Excess Power (SEP). Point-performance can be evaluated in a straight forward manner using flight mechanics equations. The third category of aircraft performance, mission-performance, presents the most involved category. Mission performance looks at the entire flight and results in the range and endurance capability for a given payload. A mission is primarily composed of climb, cruise and descent stages. Due to the large range of possible missions, complex mission profiles have to be evaluated. During mission computations, the aircraft undergoes constant changes in flight state, therefore requiring an appropriate aircraft model.

Mission performance evaluation is used to check if a design fulfills requirements (e.g. payload-range chart), to obtain operating points to design and size the propulsion system [2], and even as a target function (objective function) in Multidisciplinary Design Optimization (MDO), e.g. [3]. Computing the mission performance of an aircraft is therefore an integral part of aircraft design and is reflected by the definition of a *design mission*.

Design missions for conventional civil subsonic transport aircraft and business jets are relatively simple, usually consisting of a taxi&takeoff, climb, cruise, descent, landing and alternate&reserves sequence. From an optimization standpoint, such a mission is comparatively simple due to the high degree of standardization and the similarity between aircraft performing such missions. Especially the dominating role of the cruise segment simplifies the analyses: Optimizing the cruise segment's condition (altitude and Mach number) already provides a good estimate maximum range. However, more complex modelling of take-off and climb-out, approach and reserves leads to numerous segments and conditions (Figure 1). Implicit definitions of fuel reserves require iterations in order to find the maximum range. For example, at the end of the cruise phase, an additional fuel allowance is defined as a percentage of the cruise range and for reserve fuel there is a 5% flight fuel allowance.

Design mission for military aircraft are generally more complex due to different mission requirements (e.g. interception, time-on-station, low-level high speed), return journeys instead of point-to-point flights (radius of action instead of range), more than one engine operating mode (dry and reheat) and potential large mass changes (air-drop cargo, droppable fuel tanks, external store releases). Optimization of such design mission under operational constrains can be challenging.

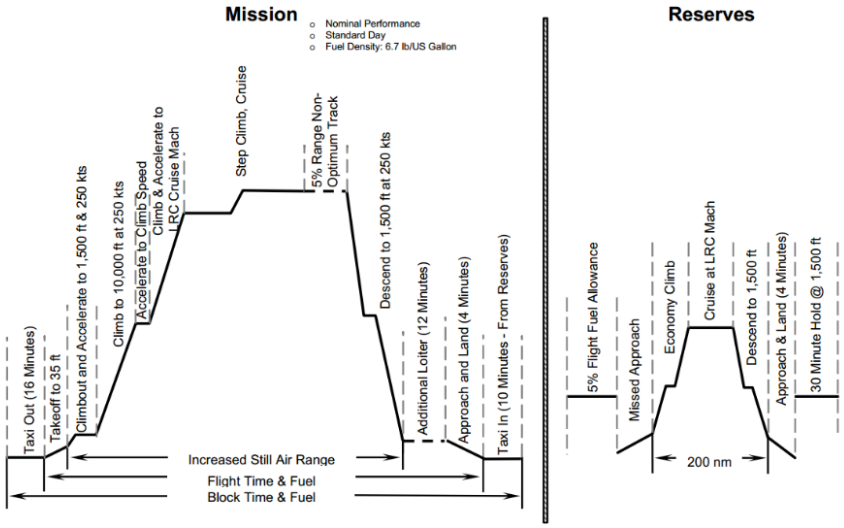


Figure 1: Typical civil mission profile [4]

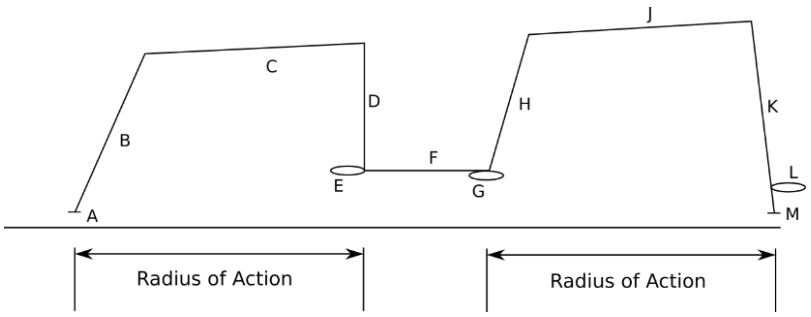


Figure 2: Exemplary BVLOS UAV railway inspection mission profile

Mission profiles of civil Unmanned Aerial Vehicles (UAVs) resemble more the military type of mission profile than the traditional civil missions. This stems from the wide range of applications of UAVs, such as imaging, mapping, inspection, remote sensing, wildlife tracking, Search and Rescue (SAR) and meteorology. To illustrate, Figure 2 shows an exemplary mission profile for a hypothetical Beyond Visual Line of Sight (BVLOS) railway inspection mission. The inspection part of the mission is flown along a railway line at low altitude (segment F) and at a certain distance (radius of action) from the launch and recovery point. Prior to the inspection, a holding might be needed, in order to wait for the railway segment to be clear of trains. After the mapping, a second holding might be necessary, if the flight back passes through controlled airspace and a clearance has to be obtained. Together with the two altitudes for the cruise segments (C and J), the two holding times (E and G) and the reserves (L), the goal is to maximize inspection distance for a specified radius of action or vice versa.

Different methods exist to evaluate flight performance. In early stages of the design and sizing process, where only few data points of the design are known, simple analytic equations such as the Breguet range equation can give a first quick mission performance evaluation [5]. However, for complex design missions, such as the one specified in Figure 2, it can be easily recognized both that using only the range equation can quickly become an involved task, and that the mission cannot be fully represented. In contrast, a complete 6-DOF trajectory simulation could be conducted. Such a simulation provides full detail about each point in a mission. However, besides the higher computational cost, a 6-DOF simulation requires a high fidelity aircraft model. The model has to include moments of inertia, aerodynamic derivatives and implement control-laws. Such information is usually not available during conceptual design, and the method is therefore unsuitable. A simplification of the trajectory simulation is the integration method. The equations of motion are simplified to two dimensions and the rotational equations are neglected (point mass). These 2D point-mass based equations of motions are integrated over time for different segments of a mission. This provides a suitable complexity for the mission definition [6] and is applicable for a wide range of model fidelities.

Existing software tools to calculate aircraft performance are NASA's Flight Optimization System (FLOPS) [7], Boeing's Mission Analysis Program (BMAP), Lissys's Piano, PACE's Pacelab APD, DLR's Simple Mission Simulator (SMS) [8] and ALR's Aircraft Performance Program (APP) [9].

Recent developments in electric and hybrid-electric propulsion have shown the need for modern performance computation tools to aid the design and sizing process [10]. In addition, advances in UAV technology together with new propulsion types lead to novel applications of aircraft use. These can lead to the rise of new design requirements. For example, the low level segments could be flown with electric power only to reduce noise emissions. Therefore, not only the sizing of a hybrid propulsion system (fuel mass, battery mass, generator power, electric motor power) becomes non-trivial, but also the evaluation of the design mission, should it serve as an objective function for the optimization. This requires great flexibility of the performance computation tool.

The current work aims to explore the possibilities presented by combining an existing mission simulation software based on the computationally efficient integration method with a powerful and flexible scripting environment. Additionally, the choice of the interface of the performance software and the suitability of integration into a CPACS based environment is discussed.

Chapter 2 describes the performance model, solution strategy and software tools. Chapter 3 presents results generated with the performance simulation environment, followed by a discussion and outlook in Chapter 4.

# 2 Methodology

This section describes the methodology used to computed and evaluate mission performance.

To compute mission performance, the integration method is used. This method uses numerical integration of the two-dimensional equations of motion split into different mission segments. The mass of the aircraft is modelled as a 2D point mass (no rotational dynamics). This places the method between the simple handbook methods (analytic integrals, i.e Breguet range equation) and full 6DOF simulations.

## 2.1 Performance Equations

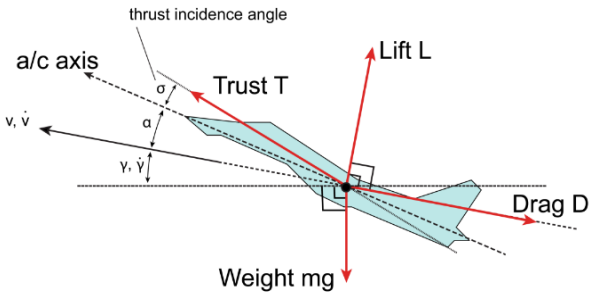


Figure 3: Forces acting on the point mass in the vertical plane

The derived performance equations are based on Newton's second law, which states that the rate of change of momentum is proportional to the applied forces

$$\frac{d\boldsymbol{p}}{dt} = \sum \boldsymbol{F}_{ext} \quad (1)$$

where p is the momentum defined as $p = mv$ and $F_{ext}$ the external forces. By neglecting the rotational dynamics (point mass assumption), the quasi-stationary performance equations can be derived. Using the assumption that the rate of change of the external forces is small (e.g. $\frac{d(mg)}{dt} \ll 1$) and the motion is restricted to the vertical plane, the equations of motion can be formulated in the flight-path angle and radial direction (airplane fixed coordinate system) as

$$m\dot{v} + D - T\ \cos(\alpha + \sigma) + mg\sin(\gamma) = 0 \quad (2)$$

and

$$-mv\dot{\gamma} + L + T\sin(\alpha + \sigma) - mg\cos(\gamma) = 0 \quad (3)$$

respectively, where T is the thrust force, D the drag force and L the lift force (Figure 3). The mass of a conventionally propelled airplane only changes by the fuel flow $\dot{m}_f$

$$\dot{m} = -\dot{m}_f. \quad (4)$$

For electric airplanes, instead of Eq. 4 an energy equation for the battery is required,

$$\dot{E_{el}} = -P_{req}\eta_D \quad (5)$$

where $\dot{E_{el}}$ is the rate of change of the electric energy stored in the battery, $P_{req}$ the net power required by the airplane (propulsion, systems, etc.) and $\eta_D$ the discharge efficiency of the battery. $\eta_D$ is usually not a constant but a function of multiple parameters (efficiency map). This simplified model for electric propulsion can easily be extended to model hybrid-electric propulsion by including an electricity generating term in Eq. 5. and use Eq. 4 to model the generator fuel flow.

Together with the kinematic equations

$$\dot{x} = v\,cos(\gamma) \quad (5)$$

and

$$z = -v\sin(\gamma), (6)$$

the resulting set of (first- and second order) ordinary differential equations (ODEs) can be readily solved by any modern numerical mathematics package. A formal derivation can be found e.g. in [11],[12],[13].

## 2.2 Airplane Model

The equations of motion (Eq. 2-3), the mass equation (Eq. 4) and the electric energy equation (Eq. 5) all contain airplane parameters that have to be modelled. These usually depend on the state of the aircraft (e.g. velocity, flight altitude and throttle setting).

Lift and drag forces (L, D in Eq. 3 and 4) are commonly modelled in non-dimensional form:

$$L = C_L \frac{1}{2}\rho v^2 S_{ref} \qquad (7)$$

$$D = C_D \frac{1}{2}\rho v^2 S_{ref} \qquad (8)$$

with $\frac{1}{2}\rho v^2$ being the dynamic pressure, $S_{ref}$ the wing reference area and $C_L$ and $C_D$ the lift- and drag coefficients, respectively. The lift coefficient can be written as a function of angle-of-attack $\alpha$ and the Mach number

$$C_L = f(\alpha, M) \qquad (9)$$

and the drag coefficient can be split into a lift-independent and lift dependent part

$$C_D = C_{D0}(M,h) + C_{Di}(C_L, M), \quad (10)$$

with dependencies on Mach number M and altitude h.

The modelling of the thrust force T depends on the propulsion type. For jet propulsion, thrust can be represented as a function of Mach number, altitude and power setting. For propeller propulsion, thrust can be written in non-dimensional form as

$$T = C_T \rho n^2 D_P^4 \quad (11)$$

where $C_T$ is the propeller thrust coefficient, n is the propeller speed (revolutions) and $D_P$ the propeller diameter. Analogously, the power required by the propeller can be written in non-dimensional form

$$P = C_P \rho n^3 D_P^5. (12)$$

The fuel flow $\dot{m}_f$ (Eq. 4) is expressed as a function of thrust (for jet engines) or power required (propeller).

## 2.3 Aircraft Performance Program APP

The Aircraft Performance Program (APP) [9] implements the point-mass based performance equations (Section 2.1) and the airplane model (Section 2.2). The parameters of the airplane model are directly entered in tabulated form. The tabulated form allows arbitrary curves (e.g. for the drag polar or thrust characteristic) and makes the airplane model independent of the airplane type. The data tables are interpolated by the software during computations. Interpolation is performed either linearly or by a

cubic spline interpolation. Details on the implemented equations and interpolation methods are found in [14].

APP provides a layer-based atmospheric model defined by a custom temperature profile. The profiles of density and pressure are determined through the hydrostatic equations and the gas laws. The International Standard Atmosphere (ISA) is used by default. The unit system of APP is based on SI-units, with the option to use British units.

The mission performance mode of APP uses a fourth-order Runge-Kutta method to integrate the equations of motion and the kinematic equations. This presents an initial value problem: APP starts with the initial conditions of the aircraft (velocity, altitude, fuel mass) and integrates each segment of the mission. The state of the aircraft at the start of a segment is equal to the state at the end of the previous segment. The integration of a segment is terminated when a defined end-condition is reached, and the computation switches to the next segment. The mission is completed when all segments have been successfully evaluated. APP provides segments to change the state of the aircraft, such as climb/descent segments and acceleration/deceleration segments. Climb/descent can be computed at various speeds, such as at constant CAS, Mach or at the velocity for best climb rate. Cruising flight is evaluated at the flight speed defined by the previous segment or at the velocity for optimum specific range or fuel consumption. The optimum speed is found using Brent's algorithm [15]. In addition, each segment allows for the selection of a different power setting, propulsion and aerodynamic configuration and the choice of crediting the distance. Loitering flight is therefore simply a cruise segment with no distance credit. The full list of supported segments is detailed in [9].

In order to maximize the range/endurance of a mission, APP computes the end-condition of a cruise or loiter segment in a way that will result in a desired reserve fuel at the end of the mission. Additionally, the radius of action can be computed by APP. This is achieved by finding the end-conditions of two cruise segments in a way that will result in equal distances. This mission optimizer mode uses an adapted Newton method.

APP provides a user interface to access, create and modify aircraft data, create mission profiles, and conduct mission and point performance computations. In addition, a Windows command line interface (CLI) using file-based communication is provided to interface programmatically with APP. The APP executable is called through the CLI with a flag specifying the type of computation (point performance or mission performance) and the location of the input file. The input file holds the location of the aircraft data file, the atmospheric conditions and specifies the desired results. The results are written to a text file at a specified path.

### 2.4 Scripting Environment

A CLI/file-based interface is highly flexible. The main advantage lies in its tool-agnostic nature. The only requirements to a tool to interface with a CLI/file based interface is to read and write the input files and to call an executable.

For APP, this functionality is implemented in the *Python* [16] module *pyAPP6*[4]. *pyAPP6* uses the *numpy* [17] package. The *pyAPP6* module can read and write the APP native files (ASCII), as well as execute the APP.exe using a system call. By reading the result file (ASCII format) after the computation has finished, a closed-loop can be established. Graphical output is provided by the *matplotlib* [18] charting package. The APP/Python interface is depicted in Figure 4a. An analogous implementation for other scripting environments (e.g. MATLAB) or programming languages is therefore straightforward.

[4] http://aircraftperformance.software/pyapp6/

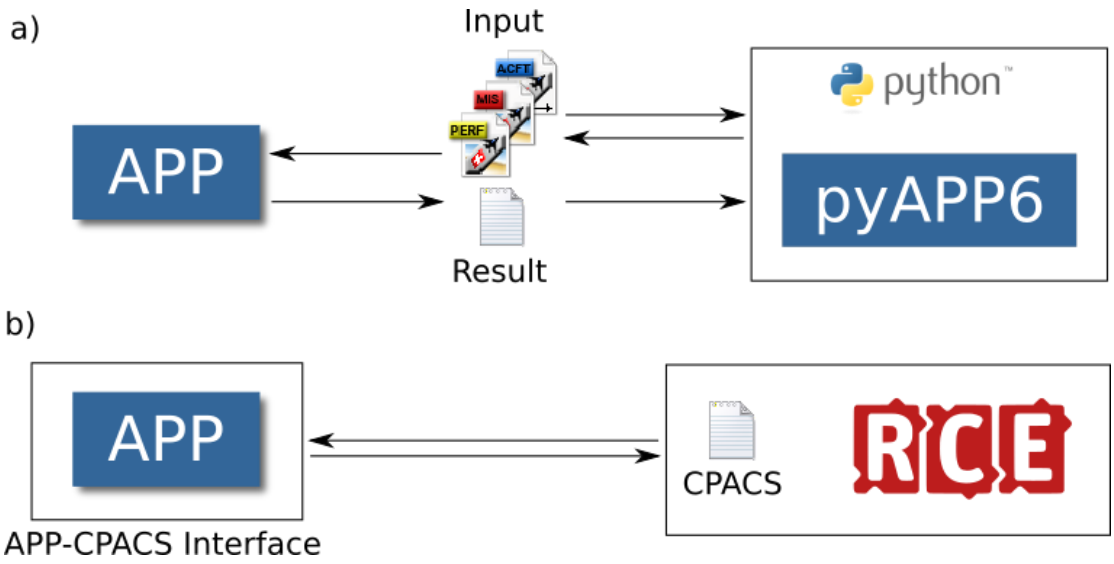


Figure 4: Interoperability of APP using a) the native file format of APP together with python scripting and b) using the CPACS format within the RCE environment

An example of a different interface is shown in Figure 4b. Instead of using the APP native files, the CPACS file format is used. For the execution environment, RCE is used. Since CPACS also presents a file based communication, and RCE has a built-in Python scripting, there is virtually no difference between the APP-Python (Fig. 4a) and the APP-RCE (Fig. 4b) environment. Since this paper focuses on the performance evaluation part rather than the complete design and sizing cycle, no coupling to other tools is needed and the APP-Python environment is used.

# 3 Results

The method described in Section 2 is applied to two cases: optimization of a climb schedule for a medium-range narrow body commercial twin-engine jet (Airbus A320 class) and a hypothetical design mission of a railway inspection UAV.

## 3.1 CAS-Mach Climb Schedule

A possible climb schedule for a commercial airliner is a CAS-Mach climb [19]. This schedule composes of five main phases. Below 10’000 ft, the speed is limited by air traffic control to 250 KCAS, therefore the first climb is executed at 250 KCAS. When reaching 10’000 ft, the aircraft accelerates to a higher climb speed, followed by a climb at constant CAS. At the crossover altitude, the climb is continued with a constant Mach number until the initial cruise altitude (top of climb, TOC) is reached. A final acceleration (deceleration) at cruise altitude is added if the climb Mach number speed is below (above) the cruise Mach number. An exemplary climb schedule to 36’090 ft and Mach 0.78 is shown in Figure 5 for a 250 KCAS / 300 KCAS / M 0.74 climb.

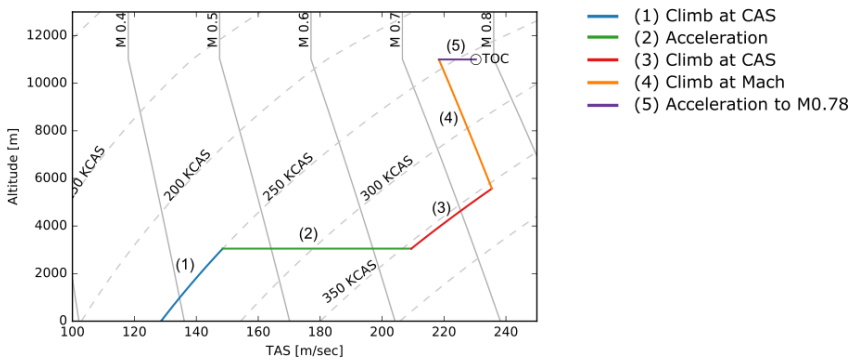


Figure 5 CAS-Mach Climb Schedule

A parameter study over the two variables (CAS and Mach) was conducted, over a range for 250 KCAS to 300 KCAS and Mach 0.6 to Mach 0.78. The parameter study was scripted in python and the missions were executed in parallel on one CPU with 4 cores (2014 Intel i7). Acceleration segments were computed with a 1 second timestep and climb segments with a 5 second timestep. Overall computing time for 1'000 missions was only 5 minutes.

For the first parameter study, the top of climb is defined as the altitude when the climb speed decreases to a speed lower than 100 ft/min (SEP < 100 ft/min). The initial altitude is 0 ft and the initial mass of the aircraft is 73'000 kg. The result of the parameter study is shown in Figure 6 for the time-to-climb, fuel-to-climb and distance-to-climb.

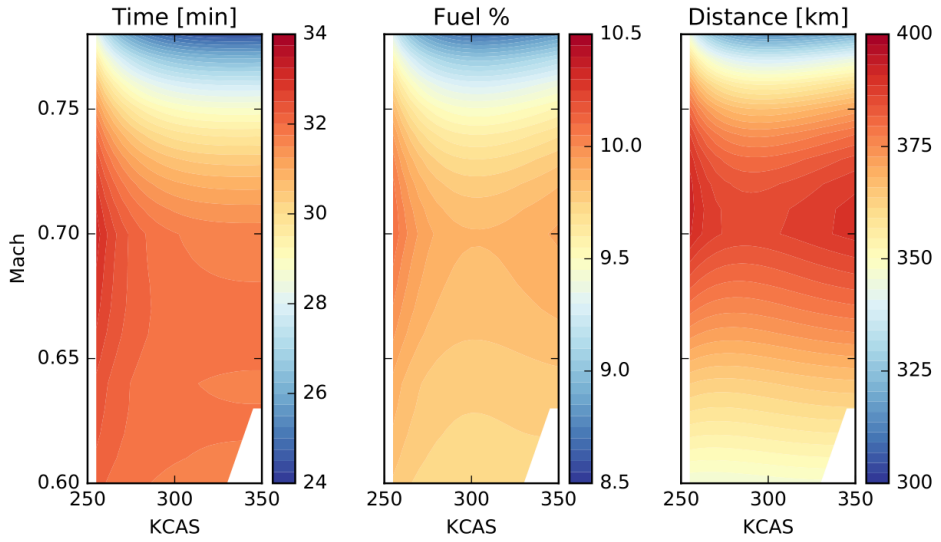


Figure 6 Climb parameter study for a SEP=100 ft/min ceiling and M=0.78 cruise speed

All fields show minima at Mach numbers equal to the cruise Mach number. At that Mach number, an optimal CAS speed at around 300 kts is visible for the fuel and distance. For the time, the optimum speed is higher and lies outside the chart. Maximal values are found for intermediate Mach numbers and the fields show a saddle-like shape. In order to investigate this behavior, an additional computation was performed. Instead of a climb-rate limited altitude, a climb to a fixed altitude of 36'090 ft was defined. The resulting fields are presented in Figure 7. All fields show similar locations for the minima, however the saddle type field is not visible anymore. Instead, a clear vertical gradient to higher values with lower Mach numbers is apparent. This can be explained when comparing two climb schedules for CAS = 300 kts and Mach 0.65 and Mach 0.78 (Figure 8). In the first schedule, the

climb Mach number is lower than the cruise Mach number and an acceleration is performed at 36’090 ft (segment 5). In the second schedule, no acceleration is needed, since the acceleration is performed during the CAS climb segment (segment 3), which makes the CAS segment slightly longer than for the first schedule. Since the acceleration performance is lower at higher altitudes, due to the lower available thrust (lower SEP), the acceleration (segment 5) therefore takes considerable more time, and therefore fuel and distance.

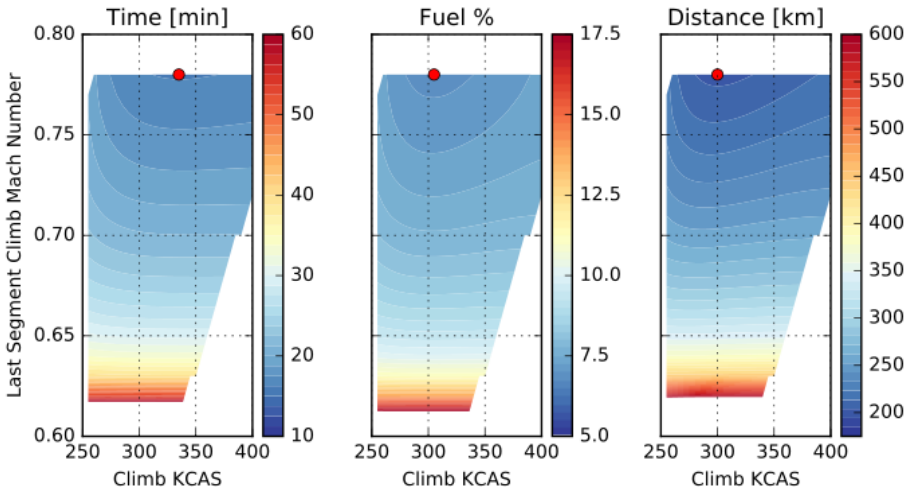


Figure 7 Climb parameter study to h=36’090 ft and M=0.78 cruise speed. The red dots mark the minima.

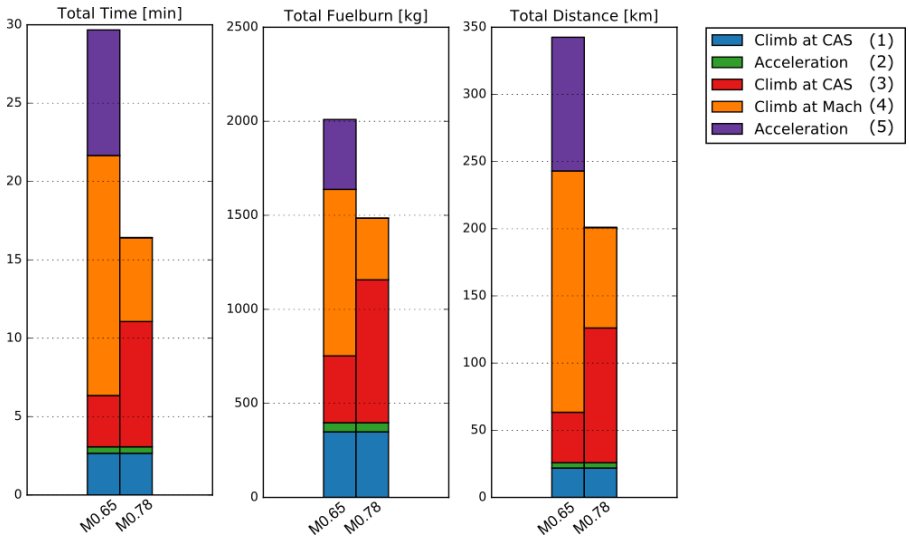


Figure 8 Contribution per segment to the overall time, fuel and distance to climb for a 300KCAS/M0.65 (left bars) and a 300KCAS/M0.78 (right bars) climb schedule to h=36’090 ft and M0.78

Even if the acceleration/deceleration to the cruise Mach number is neglected completely, the overall climb performance is still worse for the low Mach number climb, due to the inefficient climb of the segment 4. The results for the entire field without the acceleration (segment 5) is presented in Figure 9.

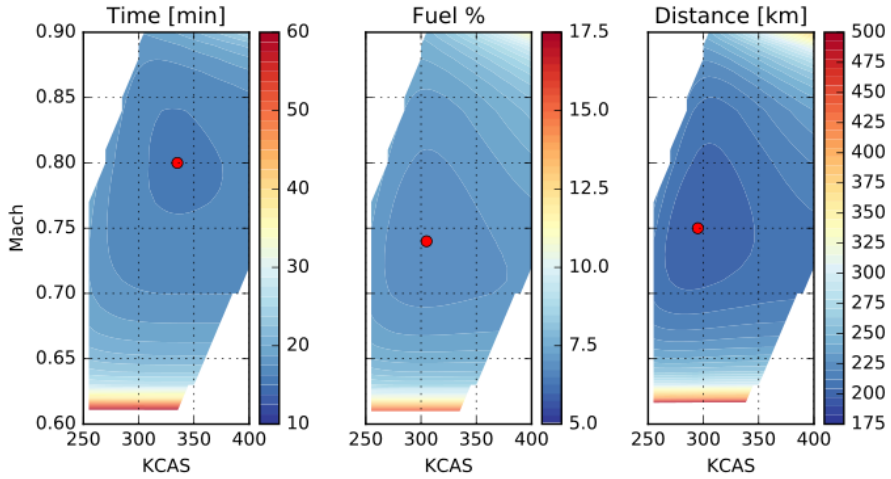


Figure 9 Climb parameter study to h = 11'000 ft without acceleration or deceleration to cruise speed (without aircraft speed limits). The red dots mark the minima.

A clear minimum is visible. For the time-to-climb, a faster CAS and Mach result in the shortest time, whereas for lowest fuel burn and lowest distance, a slower CAS and Mach speed are desired. The optima are relatively flat. Note that no aircraft speed limits (maximum CAS or Mach number) were imposed for this chart.

### 3.2 Radius of Action Optimization

The second example computes the mission performance of the UAV railway inspection mission described in Figure 2 (Section 1) using a generic piston engine UAV model. The UAV has a wingspan of 5.7 m, a length of 4.6 m and a wing area of 3.4 $m^2$. The takeoff mass with 42 kg of fuel and 45 kg of payload is 275 kg. The two-stroke engine has a maximum power of 30.8 kW at sea level and a fixed pitch propeller with a diameter of 0.9 m.

The mission parameters are chosen as follows: The inspection part of the mission is conducted at 500 ft and 70 KCAS over a fixed distance of 108 nm (200 km). At the beginning and the end of the inspection segment, a loiter of 15 minutes is included. Fuel reserves are considered by a 20 min loiter at sea level at the end of the mission and 5% of the total fuel mass. The cruise flight to and back from the inspection (S1 and S2 in Figure 10a) is performed at 8000 ft and at optimal speed.

The procedure shown in Figure 11 was used to compute the optimum mission. The two flight speeds V1 and V2 of the cruise segments S1 and S2 were computed using the Nelder-Mead simplex-downhill optimizer included in the *scipy* [20] python package. For each optimization step APP is called to compute the flight performance. The APP internal optimizer adjusts the distance of the two cruise segments S1 and S2 to compute the maximum radius-of-action in such a way that 5% fuel is left at the end of the mission.

The resulting flight profile is shown in Figure 10a and the time evolution of the fuel mass in Figure 10b. The optimum speeds for the two cruise segments were evaluated by the python script as 74.1 KCAS and 72.1 KCAS, respectively. This results in an overall maximum radius-of-action of 198 nm (366 km).

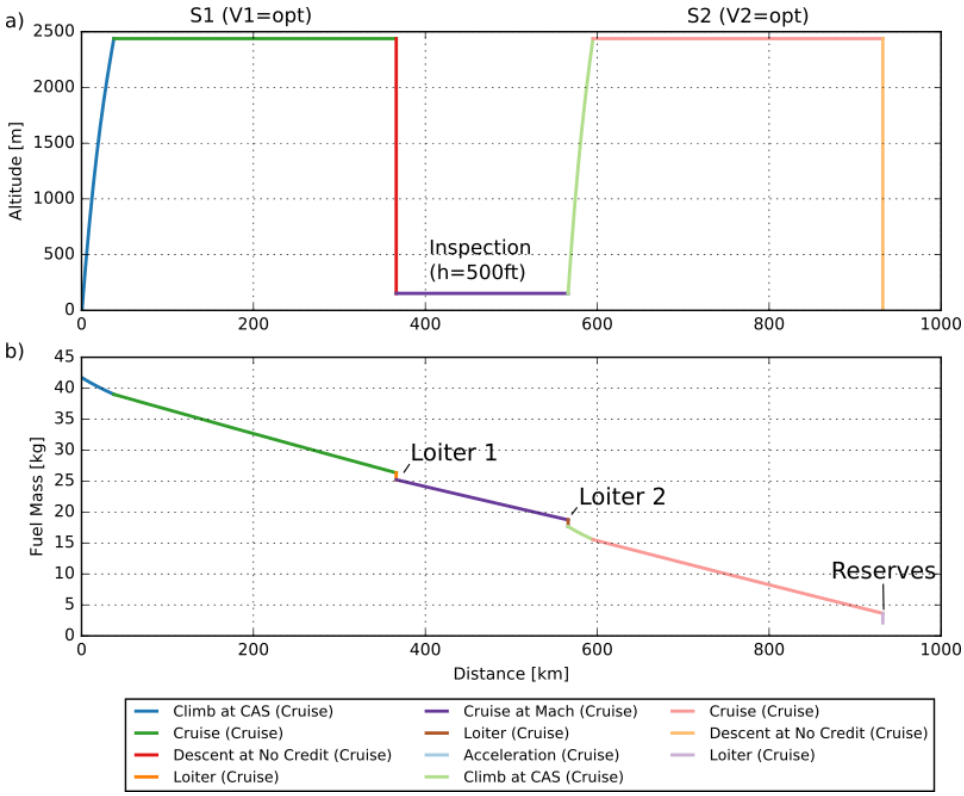


Figure 10 Railway inspection mission showing a) the flight profile and b) the time evolution of the fuel mass

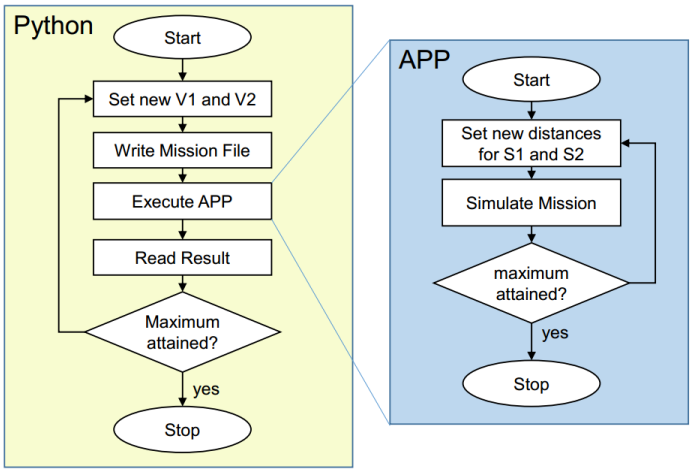


Figure 11 Simulation flow chart for the UAV mission optimization

Adding a trade- and parameter study to the performance analyses is now straight forward. Additional loops for each parameter are added around the python optimization loop depicted in Figure 11. The result is shown in Figure 12. The figure shows that the radius-of-action and the inspection distance trades almost 1:2. This is expected, since the specific range of the cruise and the inspection segments is similar (the slope of the curve in Figure 10b, kg/km, is the inverse of the specific range). The parameter study for the loiter time of each loiter segment results only in a vertical shift of the curve and is mostly independent (slope does not change).

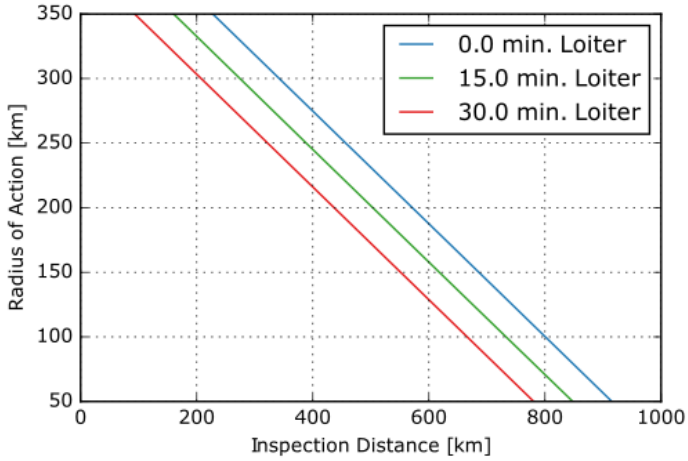


Figure 12 Trade study between radius-of-action and inspection distance, for three different loiter times.

Hybrid electric propulsion opens new possibilities, but also creates additional requirements for a performance tool and presents the aircraft designer with a larger design space. The following performance analyses example is a feasibility study for the small UAV retrofitted with a parallel hybrid electric propulsion. The goal is to minimize noise emissions by flying the inspection part of the mission using only electric propulsion. For the rest of the mission, the internal combustion engine (ICE) is used. The payload is reduced from 45 kg to 15 kg (assuming a more modern sensor package). An initial battery energy density of 200 Wh/kg was selected. The result of interest is the trade between radius-of-action (ICE part) and the inspection distance (electric part). The battery mass is varied between 30 kg and 52 kg and the fuel mass between 20 kg and 42 kg. The sum of the fuel mass and battery mass is always 72 kg. The result is presented in Figure 13a.

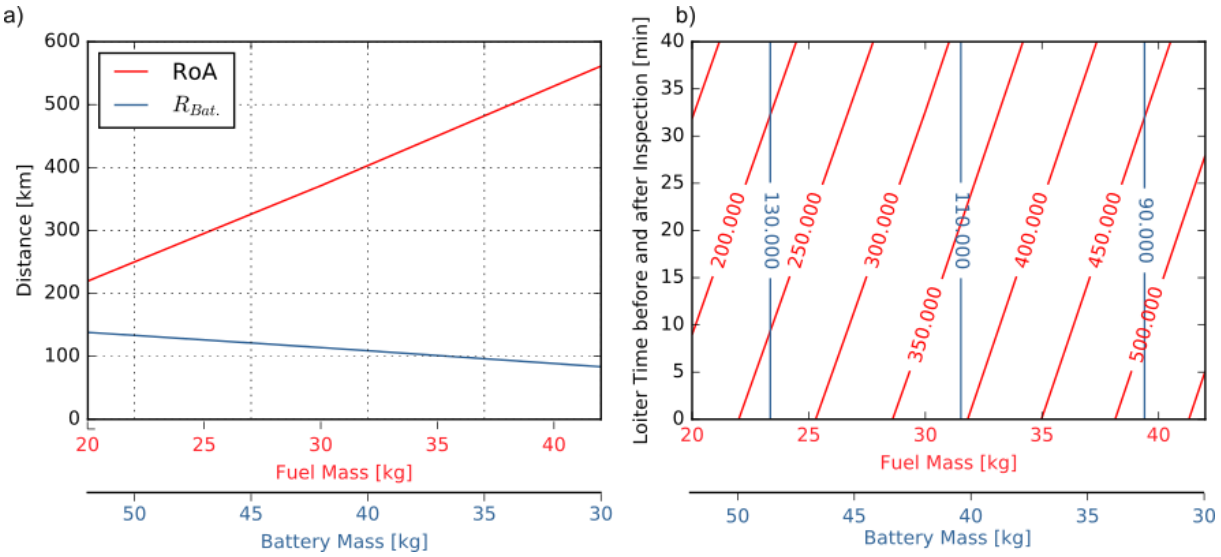


Figure 13 Trade study between battery mass and fuel mass showing a) the radius-of-action (RoA) and the distance of the inspection and with b) loiter time parameter study.

The slope of the energy curves clearly show the effect of the high energy density of the fuel. The radius-of-action increases by 15.5 km per kg fuel mass, and the inspection range by 2.5 km per kg battery mass. This is a ratio of around 6.2. If the ratio for the RoA is counted twice, this results in a 12.4 times better performance per kilogram for the combustion engine. This is not as high as expected when comparing the energy densities (200 Wh/kg for the battery vs ~12'000 Wh/kg for gasoline) and is due to the high efficiency of the electric motor in comparison to the ICE. Figure 13b shows the same loiter time parameter study as conducted in Figure 12, but with the trade between battery- and fuel mass. The chart shows, that only the radius-of-action is influenced by the loiter time, as the loitering is flown with the ICE. The inspection range is therefore independent. Note that this chart changes when a different hybridization strategy is used.

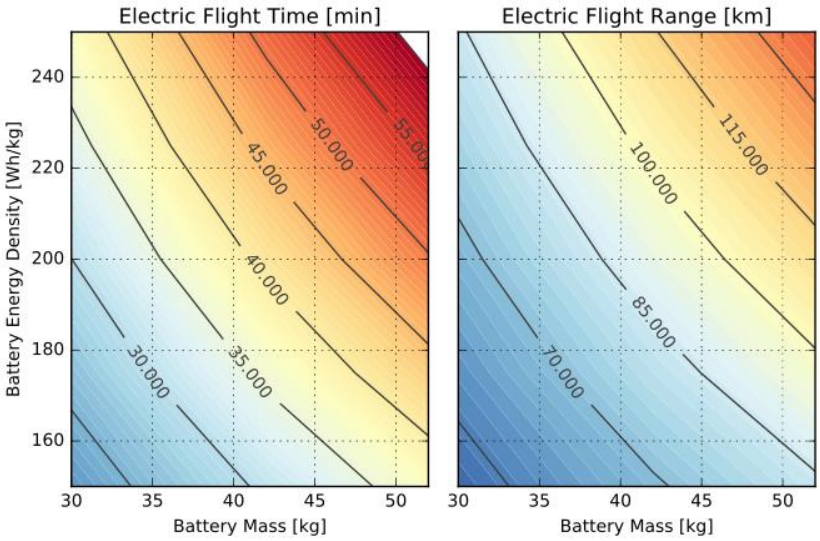


Figure 14 Parameter study for the battery energy density

In addition to the choice of mission parameter or the battery mass, an additional design parameter is the battery energy density. The scripting environment can also change airplane parameters, and therefore also vary the energy density. The result is presented in Figure 14, showing the flight time and range with increasing energy density for the same trade of battery mass vs. fuel mass. The known and significant influence of this parameter is immediately apparent.

# 4 Discussion and Outlook

This work demonstrated some of the possibilities when combining a point-mass, segment based mission performance simulation (APP) with a scripting environment (Python). The scripting environment is used to create, launch and post-process mission simulations. The mission simulation is performed by a computationally efficient and proven C++ code, by using a command line interface. This approach allows to conduct parameter studies and overall mission optimizations.

An exemplary A320 CAS-Mach climb schedule parameter study was performed. Even though the problem definition is relatively simple and well known, a complex interplay between the definition of the top of climb and the climb parameters was revealed. The resulting fields for the time-to-climb, fuel-to-climb and distance-to-climb showed differing shape depending on the definition of the top-of-climb. Especially the definition for a 100 ft/min climb ceiling (Figure 6) showed unexpected results when compared with a fixed altitude ceiling (Figure 7). The distinct saddle shape in Figure 6 is therefore presumed due to the definition of the top-of-climb. Since the available SEP depends on the

speed, slower climb Mach climb speeds reach a lower TOC altitude, making the climb shorter (and therefore faster and with less fuel consumed).

The same computations were performed using an optimizer (Nelder-Mead simplex-downhill) and the same minima were found. Instead of the approximately 1000 computations for the parameter study, only about 30 optimization steps were performed. The time-gain for this example was not found to be very significant, since each step could only be executed in series. More importantly however, a single optimized computation does not reveal the influence of the choice of end condition for the climb schedule or shows the off-optimum behavior. Especially the field visualizations give the aircraft designer useful insights and help to interpret and understand the overall performance characteristics.

The second example shows the computation of a complex design mission for a propeller driven small UAV. The performance simulation code APP is capable of computing the radius of action, in order to match a defined reserve fuel and can handle very complex mission definitions. It was demonstrated that when using a scripting environment as an outer loop, an optimizer package can be used to optimize any desired value, in this case the two cruise speeds that optimize the overall radius of action were evaluated. On top of this, a parameter study can be conducted, to vary any parameter of the mission or the aircraft. This was shown for the loiter time and the battery energy density. This example demonstrates how a powerful scripting environment can be used efficiently and the actual performance simulation can be performed by a fast and established code. All examples were conduction on an actual complex design mission, instead of a simplified, more academic example (e.g cruise only).

The limitations of the chosen coupling between Python and APP (the CLI interface) can be best shown using the climb schedule example. We have demonstrated, that a predefined CAS-Mach schedule can easily be optimized by varying the CAS and Mach speeds. The overall climb schedule is however constraint by the choice of segments made prior to executing the mission computation. Previous work (e.g. [21], [22]) showed the limitations of simplified climb schedules and proposed an overall optimization. Since the scripting environment can't access the mission computation during its execution, it can't influence the segments (e.g. number of segments, or the velocity target during a segment). Two solutions can be proposed: integrating the desired functionality into the performance code itself, or implement a "plugin"-segment, that provides a callback function for the scripting environment. The callback function is executed at each timestep of the mission simulation, providing the scripting environment with the state of the aircraft and hooks to update the flight parameters (e.g. power setting, AoA, climb angle). This enables a feedback-loop not only over entire mission segments, but also for each timestep. The downsides of such an approach is that the execution performance is likely to suffer and the complexity to set up a computation increases, reducing the potential by the synergies between the high speed performance routines and the flexible but slower scripting. Given the relatively flat minima (e.g. Figure 9), and the difficulties of flying such speed profiles in reality, the benefit of this for commercial jets is questionable. Plugins do however provide an advanced user with the option to analyze very specific, but uncommon scenarios.

When integrating performance evaluations into MDO environments, such as RCE, the results in this paper have shown that even simpler missions can lead to complex and sometimes unintuitive results. This should be considered when adding an integration based mission evaluation tool into such an environment. Providing enough visual feedback for performance evaluations should also be take into account when working with novel propulsion types (e.g. hybrid electric) and uncommon design missions & requirements. The present definition of the CPACS schema (version 2.3) offers only some fields suitable for mission definition and performance evaluation and could be extended towards including a design mission definition, and additional input fields for electric and hybrid electric propulsion.

# References


[1] Nagel, B., Böhnke, D., Gollnick, V., Schmollgruber, P., Rizzi, A., La Rocca, G., & Alonso, J. J. (2012). Communication in Aircraft Design: Can We Establish a Common Language? 28th International Congress of the Aeronautical Sciences, 1–13.

[2] Mattingly, J. D. (2002). Aircraft engine design. AIAA Education Series.

[3] Morrisey, B., & McDonald, R. (2009). Multidisciplinary Design Optimization of an Extreme Aspect Ratio HALE UAV. In 9th AIAA Aviation Technology, Integration, and Operations Conference (ATIO) (pp. 1–13). Reston, Virigina: American Institute of Aeronautics and Astronautics

[4] Bradley, M. K. and Droney, K. D. (2011). Subsonic Ultra Green Aircraft Research: Phase I Final Report, NASA/CR–2011-216847

[5] McDonald, R. A. (2011). Mission Performance Considered as Point Performance in Aircraft Design. Journal of Aircraft, 48(5), 1576–1587

[6] Anemaat, W. A. J., Po, K., & Kaushik, B. (2008). Aircraft Performance Prediction: Comparison of Classical Handbook Methods to Detailed Time Integration Computer-Aided Methods. SAE International Journal of Aerospace, 1(1), 2008–01–2253

[7] McCullers, L. A. (1984). Aircraft configuration optimization including optimized flight profiles.

[8] Moerland, E., Zill, T., Nagel, B., Spangenberg, H., Schumann, H., & Zamov, P. (2012). Application of a distributed MDAO framework to the design of a short-to medium-range aircraft.

[9] APP 6.0 User Manual (2016), ALR Aerospace, Zurich

[10] Oliviero, F. and Cipolla, V. (2016). Hybrid Propulsion System and Aircraft-level Performance Simulation, E2 - Fliegen Symposium: Flugzeuge mit elektrischem Antrieb - Aufbruch in die emissionsfreie Zukunft Stuttgart, 18-19th February 2016

[11] Ruijgrok, G. J. (1990). Elements of airplane performance (Vol. 2600). Delft, Netherlands: Delft university press.

[12] Ojha, S. K. (1995). Flight Performance of Aircraft, American Institute of Aeronautics and Astronautics. Inc., Washington, DC.

[13] Eshelby, M. E. (2000). Aircraft Performance, Theory and Practice.

[14] APP 6.0 Technical Reference Manual (2016), ALR Aerospace, Zurich

[15] Press, W. H. (2007). Numerical recipes 3rd edition: The art of scientific computing. Cambridge university press.

[16] Python 2.7, Python Software Foundation, Retrieved from https://www.python.org/

[17] Stéfan van der Walt, S. Chris Colbert and Gaël Varoquaux. The NumPy Array: A Structure for Efficient Numerical Computation, Computing in Science & Engineering, 13, 22-30 (2011)

[18] John D. Hunter. Matplotlib: A 2D Graphics Environment, Computing in Science & Engineering, 9, 90-95 (2007), DOI:10.1109/MCSE.2007.55

[19] Getting to Grips with Aircraft Performance (2002), Airbus, Flight Operations Support & Line Assistance

[20] Jones E, Oliphant E, Peterson P, et al. SciPy: Open Source Scientific Tools for Python, 2001-, http://www.scipy.org/ [Online; accessed 2016-08-10].

[21] Johnson, D. T. (1972). Evaluation of Energy Maneuverability Procedures in Aircraft Flight Path Optimization and Performance Estimation (No. AFFDL-TR-72-58). Air Force Flight Dynamics Lab Wright-Patterson AFB OH.

[22] Pierson, B. L., & Ong, S. Y. (1989). Minimum-Fuel Aircraft Transition Trajectories. Mathematical and Computer Modelling, 12(8), 925-934.