iMission - Leonardo Helicopters integrated performance simulation: consolidating decades of lessons learnt and keeping the door open to the lessons to be learnt

Riccardo Bianco Mengotti
Head of Flight Mechanics

Valentina Giuliani
Performance & Sizing Technical Lead

Fabio Nannoni
Head of Design Organization

Leonardo Helicopters
Cascina Costa (VA), Italy

Giuliano Prando
Technical Manager

Lorenzo Frigerio
Senior Specialist

Line Up Aviation Srl
Somma Lombardo (VA), Italy

ABSTRACT

The capability to predict of the performance of a product is the key for a number of evaluations, ranging from marketability, to requirement satisfaction, from competitive advantage to operational planning.

For this reason, from an historical perspective, the first analytical tools developed in the industry have in most cases been the performance codes. And for the very same reason performance tools are the first used in a development project or, better say, before the launch of a project.

In Leonardo Helicopters, performance software have been developed since the 70s~80s in the both former companies, Agusta and Westland, creating a solid base which incorporated over time all the experience gained during the development of products like the Lynx, the A109, the EH101, the A129 or the AB139.

In the recent years, the eco-sustainability questions posed by the CleanSky European research program and the momentum of the Company towards tiltrotors created the need for an upgrade of the performance tools. It has thus been decided to develop a common code, iMission, able to replicate the helicopter the well-established methods, allowing new configurations to be included as well as improving usability and graphing functions and including the suite for emission-related calculations.

INTRODUCTION

The possibility to assess the performance capability of rotorcraft against defined specification goals is fundamental for the preliminary optimization of the configuration during the design phase but also is required to evaluate technology improvements by translating the benefit into operating capability, to assess the bid requirement satisfaction of an existing helicopter and compare products with competitor to determine the relative strengths and weaknesses.

This principle, valid for all sorts of products, has always been clear for means of transportation and in particular for air vehicles, where a lack in this field can imply design, project, schedule or -even worse- safety risks. It is also true that the evaluation of the performance and mission profile of a product is in many situations possible with a limited knowledge of the details of the product itself: information on overall power required and available, fuel consumption, and weights are sufficient for a good part of the cases to be considered.

Therefore performance evaluation shall be regarded as a simple but critical aspect of optimization loops in the design process.

A performance code needs to allow easy and quick evaluations in particular during the initial design phases, when data availability is limited, but clear results are needed to support design decisions; moreover a performance tool shall be flexible enough to adapt to different configurations and design solutions; finally a performance tool shall allow increasing complexity levels in order to include better data definitions as the design is progressing, from initial assumptions to experimental data.

THE EVOLUTION OF PERFORMANCE EVALUATION IN AGUSTA AND WESTLAND. A BACKGROUND PERSPECTIVE

To better understand the significance of the iMission development, it is important to frame it in an historical perspective, looking at the origins of the analytical performance calculation at Leonardo Helicopters.

In particular, this section will focus on the Italian side, namely on the Agusta Preliminary Design department, where most of the core of the iMission software was born at the beginning of the 80s.

In that period Agusta had just completed the development and the certification of the A109 light twin helicopter and was busy with the A129 Mangusta military machine. Other machines were in the Company plans, but the 109 and the 129, not so distant in size and configuration, represented essentially a full dataset from development experience. This included of course the evaluation of the performance, which were extensively tested in hover and forward flight, resulting in adimensionalized power maps, used for all the necessary evaluations.

This approach, deductive rather than predictive, was of course limited in case new machines, new conditions or new type of performance had to be calculated. So in order to be prepared for such requests, in parallel with the experimental data, predictive models were developed, starting from first principles and known theories, essentially developing the helicopter momentum theory for all the conditions to be evaluated, with a code named Polar3, used until a few years ago and replicated today in iMission. This proved to be essential once the Company was addressing the needs of new markets with hot and high conditions, well beyond the limited validity of the adimensionalized data used previously.

Another important evolution that took place during the '80s was the adoption of the PC hardware and the FORTRAN code in place of minicomputers and mainframes, made possible because of the increasing power of the PCs. This step implied an easier development of the methodologies, ensuring also a portability of the source codes that in several cases has been used almost as is in today's iMission.

At the end of the '80s and at the beginning of the '90s, the development of new analytical capabilities was again anticipating the incoming needs related to the new development of the EH101 and of the NH90, creating in fact the full set of tools for the performance prediction of conventional helicopters (named PERFORM, NFCONS, NFCONH).

This set of tools was not limited to the performance evaluation and evolved also to trim routines that make use of algebraic rotor

(NFTRIM), maximum thrust determination (MAXCTIG), blade element models to evaluate rigid rotor dynamics and control loads (NFLOAD), transient simulations to calculate helicopter response in the time domain and inverse simulation to address prescribed trajectories (NFPATH), for example in the flyaway or Cat A take-off manoeuvers.

These analytical tools added to the ones developed at Westland in United Kingdom represent a unique engineering value, so one of the initial cornerstones of the new performance tool was to preserve the value and the experience of the existing performance evaluation methods, including from a coding standpoint, the same calculation routines and, from a testing standpoint, ensuring the one to one consistency with the old methods and previous performance evaluations.

In this framework, in the last years, some concepts and prototypes of new or unified performance codes had been developed: in the UK a full code, named eMission, with high focus on the GUI aspects while in Italy a performance pre and post-processor, intended to provide one single environment to launch legacy executables. All these developments provided useful experience for the creation of the new iMission code.

THE OPPORTUNITY FOR A NEW PERFORMANCE EVALUATION CODE

In a scenario characterized by the availability of several proven Company codes but a growing request for stronger integration of tools and design process, the request for estimation of the environmental impact related to the CleanSky program and the Company interest into tiltrotors, created the need and opportunity for an upgrade of the performance tools. Of course the straightforward approach could have been to create new dedicated programs to calculate emissions and to replicate the tools for the tiltrotors, but this would have implied a proliferation of codes and routines to do essentially the same evaluations just on different rotorcraft categories, requiring to repeat the same effort every time a new configuration was to be considered.

It was the right time to restructure, with the help of Leonardo specialists and Line-Up external consultants, the performance codes preserving for the helicopter the well-established methods and allowing new configurations and new evaluation to be included. The approach of one integrated but modular code brought immediately advantages, from the unification of the input and output models/files to the separation of the calculation functions from the interface, which in this way was easily moved in a graphical format.

Main top level requirements established for the development of iMission were:

- leverage on well-known and proven existing Company methods, by including/porting as much as possible the well trusted core calculation routines, and by planning extensive regression testing to ensure consistency of results and approach;

- unification of a number of executable and formats, by means of a single code fulfilling all the calculation needs and as a consequence adopting on single input format;

- code modularity, isolating trim and power calculation routines, integrating with generic performance evaluation functions, allowing addition of new configuration by simply defining new trim and power definition modules;

- open/expandable input file formats, allowing easy reading and introduction of additional parameters as needed

- traceability of configurations and modifications, allowing local modifications preserving baseline models, with clear documentation of the changes;

- eco-compatibility aspects, introducing in the code the evaluation of pollutant information;

- mission optimization, with the aim of minimizing the fuel consumption and pollutant emission or increase the mission capability acting on the mission profiles;

- full input and output awareness, by means of advanced user interface and essential built-in plotting capability;

- data exportability, by means of known and accessible formats as well as possibility to select the amount of results to be produced.

IMPLEMENTATION ASPECTS

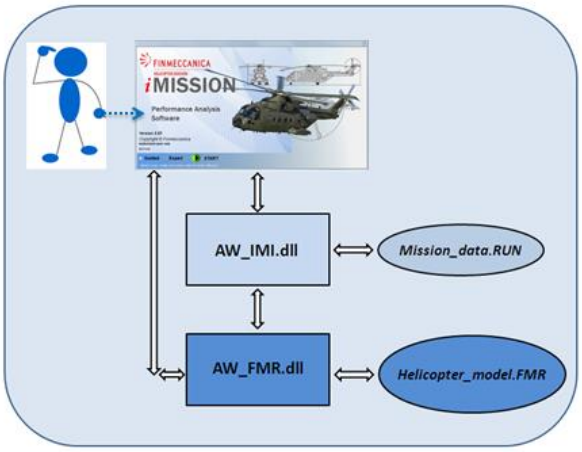
iMission framework development began in 2010 and is currently in use at Leonardo helicopter as standard performance tool.

It has been developed with the aim to give the User an instrument for:

- Visualize and manage vehicle model and configuration data (File FMR);
- Prepare or modify launch parameters;
- Launch performance and mission calculations;
- Perform optimization on mission;
- Visualize results of calculations;
- Prepare Plots;
- Compare results of different calculations;

iMission is a GUI (Graphical User Interface) developed in vb.net language that uses:

- the external library AW_IMI.dll to compute aircraft performances,
- the AW_FMR.dll library to manage a common data structure representing and handling rotorcraft models for numerical calculation.



Since May 2013 the OPT library provides coupling with the in-House genetic program DESPOTAX 6.0 that allows the User to automatically optimize mission profiles.

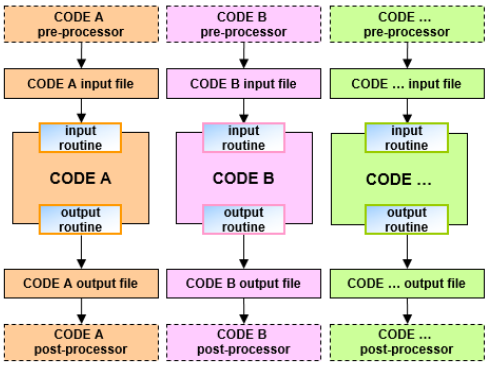
AW_FMR LIBRARY

First user requirement for iMission framework was defining a common model representing helicopter characteristics, harmonizing both Italian and United Kingdom parameters nomenclature and reference used in several software codes, not only performance calculation related.

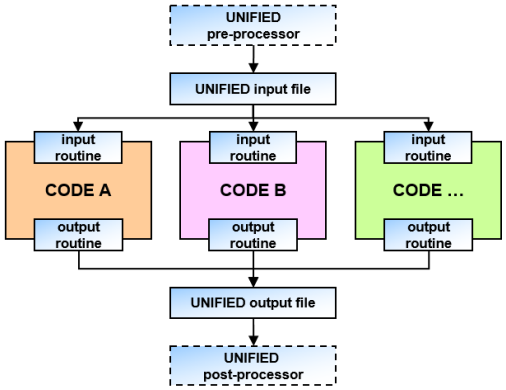
The harmonization procedure led to Agusta Westland Flight Mechanics Rotorcraft model file format (AW_FMR file).

Rather than prepare an input files pre-processor for each code, as well as a postprocessor for outputs, Leonardo Helicopters Flight Mechanics decided to undertake a deeper review of all input models used.

So even for future Flight mechanics application specialists have at their disposal developing methodologies with a reliable, powerful and documented module, easing the software development work, allowing to focus on higher added value development tasks, ensuring at the same time that new programs will automatically comply with a common format.



× POTENTIAL DIFFERENCE AMONG TOOLS
× PAINFUL INPUT/OUTPUT CONVERSION
× WRITE NEW CODE FOR EACH TOOL
× DIFFICULT LEARNING
× NIGHTMARE FOR PRE/POSTPROCESSOR



✓ CONSISTENCY AMONG CODES
✓ IMMEDIATE DATA EXCHANGE
✓ I/O ROUTINES RE-USABILITY
✓ EASIER TO LEARN
✓ ALLOWS PRE/POSTPROCESSOR

The format complies with the following guidelines:

1. Multi-code data repository, i.e. include parameters used by various tools.

2. Separation: model file only includes aircraft data not condition of analysis;
3. Hierarchically organized;
4. Expandable: allowing addition of new parameters;
5. Flexible: supporting encryption, crc-based protection, etc.

The model parameters are clustered hierarchically (one hierarchical level). Each parameter is identified by group (also called Namelist) and parameter's name (i.e. radius of first rotor is identified by ROTOR1 RADIUS). In this way it is possible to duplicate a group ensuring flexibility (for example in the current version 4 rotor groups with the same variables are provided and it is possible to add a new group i.e. ROTOR5 with minimal changes to the framework). Each parameter has associated metadata: unit of measure, description, list of possible values it can take, level of visibility.

Each parameter can be numeric, textual or the path to an external file; in order to ensure compatibility with legacy codes all the previous external file formats were inherited. These files have typically data in tabular form.

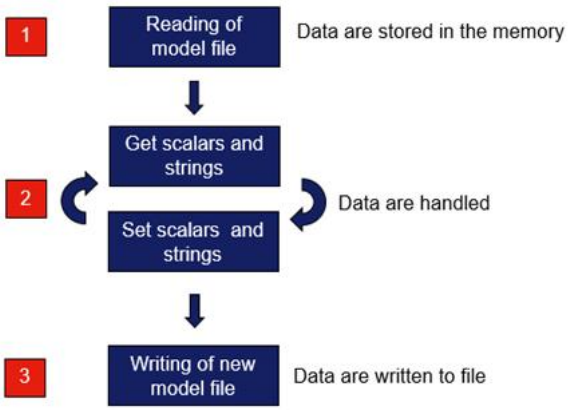
The first FMR file format was issued on July 2010, in that version (release 0.9) 1300 parameters divided into 52 groups were planned, in current version (release 61.0) over 2300 parameters divided into 54 groups are embedded. This list of parameters allows characterizing any vehicle configuration. The Namelists embedded in the file format describe: AFCS, airframe, controls, engine, environment, landing gear, horizontal tail, aircraft inertia and mass, optional kit, nacelle, rotors, sensors, slung load, transmission, vertical tail, wings and general information related to the model.

iMission framework use about 140 parameters over the 2300 available in the AW_FMR to calculate the performance of an aircraft. The large number of parameters embedded in the FMR format indicates what has been the initial effort to create a general input file for any possible application.

FMR library deploys a function to verify the model data integrity. Each model parameter has a level of visibility related to four authentication levels. AW_FMR library calculates an hash code for the parameters that have the same visibility level (4 hash codes) and store results in a model parameter. Only level 1 users are allowed to seal a model saving in the model a proper hash code. An iMission user can check integrity of a FMR model file comparing this parameter with hashcodes calculated on the fly. FMR model may be also encrypted in order to protect the data from disclosure and modification.

iMission user's levels	
1	system administrator
2	specialist
3	advanced user
4	basic user

The AW_FMR.dll library aims to read data of a model and make them available to any model data consumer, hence this library has no dependencies from other libraries or executables, but others iMission framework modules are tight coupled with this library.



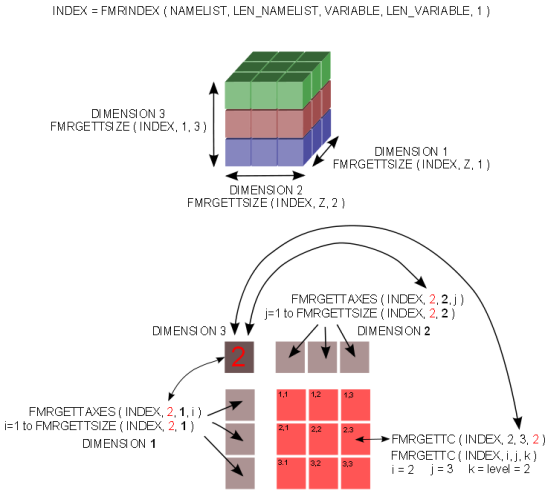
The library interface exposes several functions. Here are listed the key features guaranteed by the interface:

- read a complete aircraft model file (also crypt file) and load it into memory for further use;
- change single parameter, accessing with a unique identifier (namelist/group and parameter names);
- classify each model parameter;
- save a model in ASCII file (AgustaWestland Flight Mechanics Rotorcraft format);
- Grant files integrity.

AW_FMR library is written in FORTRAN90, first releases compiled with COMPAQ Fortran 6.6 , latest with INTEL Fortran Composer and tested on windows 32/64 bit.

To grant simulations repeatability, inside library a set of commonly used constants are defined: e.g. gravity acceleration and sea level temperature) the library is also able to automatically update a group of special parameters defined as “derived” (e.g. rotor tip speed is calculated from RPM and rotor radius, if one of these parameters changes also tip speed is updated accordingly, derived type parameter access is read only)

Internally all parameters are stored in array accessed via unique index. This index is associated to the group name and parameter name.



Parameters related to external files require special handling. A typical external file (e.g. C81 file) consists of a table with the variation of a parameter as a function of other control parameters (actual library version can read 16 different table-file types including a generic table-file format, it handles at most three control parameters for C81file-table control parameters are angle of incidence and Mach number). AW_FMR library is able to parse a table-file and store its content in a special three-dimensional matrix. In this matrix are saved also control parameter conditions (e.g. for a C81 file type, aerodynamic coefficient are saved as well as related angles of attack and Mach numbers).

Library grants with a simple interface to:

- access to single cell value;
- access to the values of control parameters (e.g. angle of attack for a C81 file);
- interpolate table in a control point given by user (for a C81 file user will specify angle of attack and Mach number);
- extrapolate table, it is possible to extrapolate linearly with the data available, or use the latest available data.

AW_FMR library exposes simple interface functions for the principal programming languages. Simple testing programs in .NET framework, C, C++, Delphi, VBA (Excel), and FORTRAN were issued during the validation phase of the library. After the validation of AW_FMR library a long-term phase to adjust old performance codes to the new file format has risen.

AW_IMI LIBRARY

Before the development phase of the iMission code, a prototype has been written in Visual Basic (VB), uncovering some computing issues related to coding language itself. VB was not originally designed as a calculation language, but, instead, for ease of use in user-interaction and controlling processes. Subsequently it has been decided to have a clear separation between data view management and business logic. Business Logic subroutines are written in FORTRAN, packed and saved as library (Fortran language is particularly suitable for realizing algorithms). During library development several visualization prototypes written in different languages (C, Fortran, VBA, Delphi) were deployed for debugging and testing purpose.

IMI library initial development activity was:

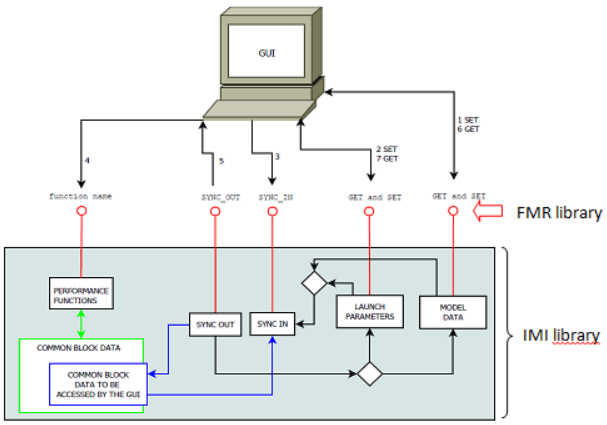
- identify the most suitable calculation algorithms for a particular vehicle configuration;
- identify the necessary inputs to these algorithms, dividing them between model parameters (updating AW_FMR format file, if needed) and launch parameters (added to the list of IMI launch parameters). Data from input files of several analytical tools have been hence collected into a unique data structure (IMI Launch Parameters). First release of IMI format did not provide parameters hierarchical organization, however soon the number of algorithms and related parameters pushed to adopt the same hierarchical structure used in the FMR file format.

IMI Launch Parameters are then hierarchically organized in two levels:

- first level is a uniquely named namelist that group parameters of the same kind;
- second level is the list of parameter data associated to each namelist. Each parameter is characterized by:
  - Namelist, (first level);
  - Name, (uniquely parameter identification inside Namelist);
  - Value, (numerical or alphanumeric parameter value);
  - Type, (to specify if the parameter is a number, a string or an array);
  - Class (a further clustering criterion);
  - A set of auxiliary information such as parameter's description and unit of measure.

AW_IMI library is written in FORTRAN90, first releases compiled with COMPAQ Fortran 6.6 , latest with INTEL Fortran Composer and tested on windows 32/64 bit.

Internally all parameters defined in IMI file format are stored in array or matrix accessed via unique index. This index is associated to the group name and parameter name as for FMR library.



1. Set MODEL DATA variables via a FMR library SET interface;
2. Set LAUNCH PARAMETERS variables via IMI library SET interface;
3. Call the INPUT SYNCRONIZATION IMI library function;
4. Call the IMI library performance function;
5. Call the OUTPUT SYNCRONIZATION IMI library function;
6. Get results via FMR library GET interface;

Other internal data (scalar, array and matrix values), shared by all performances routines, are saved in a COMMON BLOCK data. These COMMON BLOCK variables are internal to the library and not directly accessible for the user by a call to the library. No metadata (as description, unit of measure, …) are provided for these data, however to ensure code's maintainability variable names are meaningful and unit of measure is SI. To batch update these variables from library caller, a unique input synchronization interface function is provided. Input Synchronization function copy MODEL DATA and LAUNCH PARAMETERS variables in the COMMON BLOCK variables, using hard-coded link rules. To retrieve calculation results a unique interface output synchronization function is provided as well. Output synchronization interface copy COMMON BLOCK variables back to LAUNCH PARAMETERS and MODEL DATA, using hard-coded link rules.

The IMI library calculates general performance and mission performance where

- general performance can be either spot-point performance (for a given flight condition: power required, maximum speed, max hover mass, …) or plots of spot-point performance to create performance charts (maximum mass to hover, chart, maximum speed, …);
- mission performance can be single calculations of maximum range or radius of action with a given payload, or payload-range charts.

The limiting performance of an aircraft can generally be determined by matching the power required to achieve a desired flight condition with the power available from the engines and transmission.

Power required and available calculation algorithm varies with the configuration and resolution method chosen by user, while the algorithms to evaluate a single performance type (e.g. speed of best range) starting from Power surplus are fixed.

Currently four methods to define the power required are implemented two for the conventional helicopter configuration and two for the tiltrotors:

- ENERGY Method for Helicopter. This simple, preliminary theoretical method is based on the combined Momentum Theory (also known as Disk Actuator Theory) and simplified blade element theory. It allows obtaining, quickly and with a good approximation, a first evaluation of the power required for an helicopter. The basic energetic method is not able to take into account the phenomena related to the rotor stall and the fluid compressibility, but in the HEL - ENRG formulation there are some empirical corrections for the evaluation of

the average blade drag coefficient that permit more accurate calculations.

- POWER CARPET Method for Helicopter. The Power Carpet method utilizes the experimental results given by flight tests. These are generally a table of power data for given speeds and weights, normalized with respect to σ (air density ratio) and to n (the rotor speed ratio wrt the nominal).
- TILTROTOR SIMPLIFIED Method for Tiltrotor. The Tiltrotor simplified method is a preliminary theoretical approach that allows to obtain, quickly and with a good approximation, a first evaluation of the power required for a tiltrotor only in helicopter or aircraft mode (nacelle angle =90 or 0deg).
- TILTROTOR BLADE ELEMENT Method for Tiltrotor. Tilt Rotor Blade Element methods is able to obtain the performance and the required power estimation for a tiltrotor at every nacelle angle because is based on a more extensive computational analysis that includes the aircraft longitudinal trim and the proprotor power definition based the blade element theory and uniform inflow model.

All the four methods determine the power required of the aircraft having in input a dedicate aircraft model (FMR model define the characteristic of the vehicle) and the flight condition to be analyzed.

Power Available is obtained from the comparison of the engine data and transmission limits.

In general it is defined a set of power ratings represented by the maximum power in output, the maximum duration and the condition (All Engine Operative or One Engine Inoperative)

The engine data are collected into a linked file which describes the max engine power for the relevant ratings and atmospheric conditions in tabular form (engine look up table) or with a mathematical formulation (so called "rubber engine").

The development of IMI computing library took a long time, during this time several visualization prototypes written in different languages (C, Fortran, VBA, Delphi, .Net, Matlab) and different capabilities (basic GUI, advanced GUI, console program, Excel Macro) were developed for debugging and testing purpose.

GRAPHICAL USER INTERFACE

The iMission program (or in principle a Graphical User Interface or a Console program), allows to run different types of calculations using the new file format for air vehicle configuration and launch parameters. Caller will use the AW_FMR library interface to access to helicopter configuration and the AW_IMI library interface to perform the calculations.

iMission replaces the codes accessing to the same functionalities of the old codes via AW_IMI library interface, adopting a loose coupling with data model and so far ensuring a better code maintainability.

In this manner both business logic (algorithm details for calculating performance rating) and data access layer (routines and format details of files describing model configuration) do not reside in GUI, which acts as a simple viewer.

Total separation of model structure (AW_FMR model) from its visualization (Graphical Interface) is not possible. Data model replication inside GUI is tolerated to assure a clearer interface (i.e. to group input data, display meaningful names for labels, etc.).

The comprehensive GUI designed to manage IMI and FMR libraries is iMission.

MISSION OPTIMIZATION LIBRARY: AW_IMI_OPT.DLL

The iMission performance optimization capability, introduced in order to allow reduction of pollutants and fuel consumptions, was added to the baseline code using an external library providing state of the art optimization functionality.

In fact AW_IMI library is coupled with the genetic optimizer in-house program DESPOTAX. A specific interface library between DESPOTAX and AW_IMI library it has been developed: AW_IMI_OPT.library. This library allows initializing the interface, to pass data from Despotax to AW_IMI library and to perform mission calculation with AW_IMI library and pass back to Despotax the results.

For a given mission profile is possible to evaluate the best optimized values for this step variables:

- Altitude variation;
- Distance;
- Horizontal speed;
- Nacelle Angle;
- RPM;
- Step Duration.

iMission user can specify constrains for genetic algorithm. Input which do not grant constrains will be discharged automatically by the genetic program. User can define mission constrain (mission duration, mission distance, total fuel consumed) or single mission step constrain (step speed, step fuel consumption, …).

iMission user can select, as optimization target, one or more among these variables:

- total distance;
- total time;
- total fuel consumed;
- Pollutant emission (CO_2, CO, H_2O, SO_4 , NO_X, PM, UHC)

AW_IMI_OPT library is written in FORTRAN90, compiled with COMPAQ FORTRAN 6.6 and INTEL Fortran Composer, and tested on windows 32/64 bits environments. It is possible with minor source modifications, to compile library also on LINUX system.

Mission optimizer will rewrite some steps input to assure overall distance and altitudes consistency between optimized step and surrounding fixed steps. The program will apply default input corrections on fixed step input parameters. User will be able to overrides default correction setting a set of constraints. Available constraints will depend on variables optimized and type of current step.

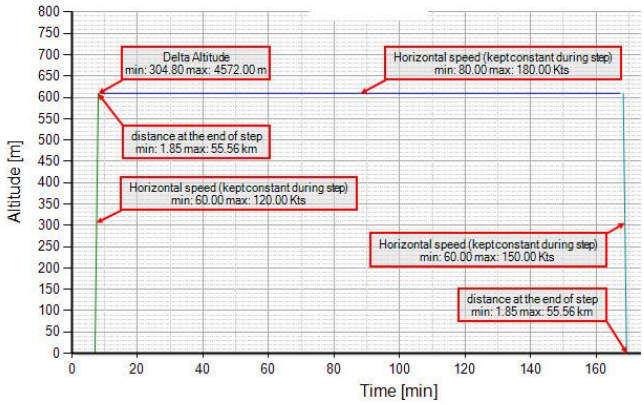
Optimization example

iMission framework with optimization library was adopted as Technology Evaluator (TE) in Clean Sky (CS) project.

In the following there is the test result for a Passenger Transport / Corporate mission reported in a final Clean Sky deliverable.

<u>Passenger Transport / Corporate Mission</u>

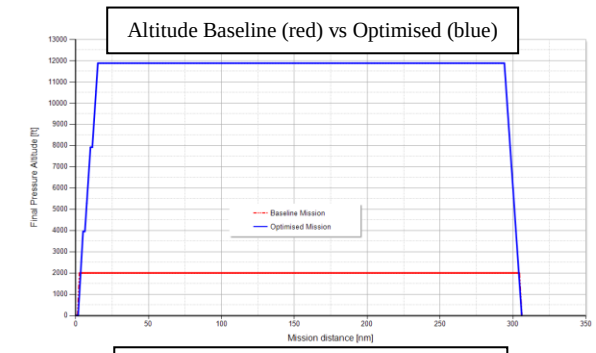
The figure below shows the free variables of optimization process and their change domain.

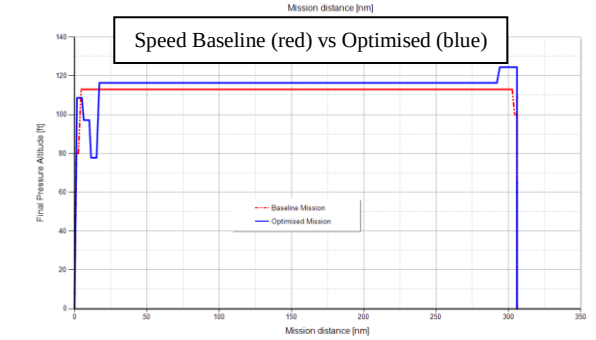


The parameter involved in optimization process are step speed, step altitude, step distance. The total mission distance is constant and equal to reference mission, while the total time could change.

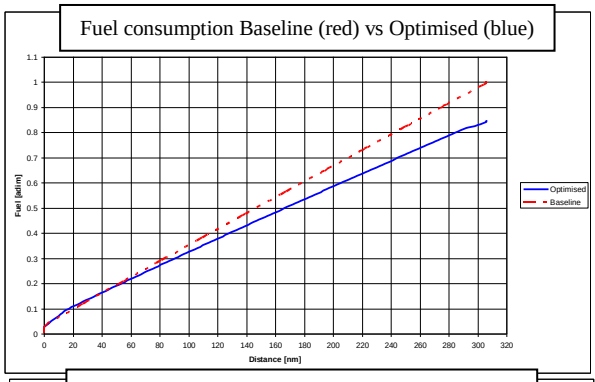
The iMission outputs, target of optimization process, are pollutant and fuel consumption.

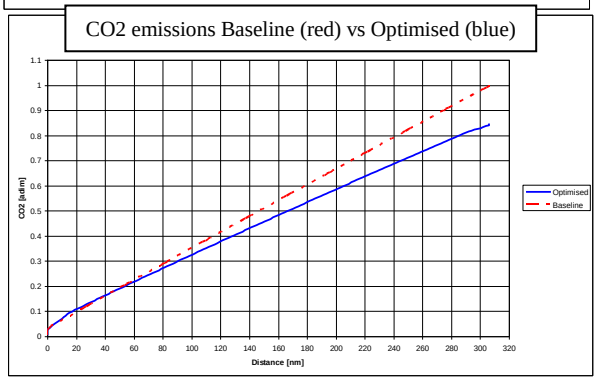
Next figures show the altitude and horizontal speed of the optimized trajectory compared with baseline mission.





The following figures and table highlight the reduction of fuel consumption and pollutant obtained by mission optimization.





	FC	CO2	CO	SO4	NOX
Difference wrt baseline mission	-15%	-15%	-11%	-15%	-17%

THE IMISSION CODE AT WORK

iMission has quickly been introduced at Leonardo Helicopters in the day by day jobs for a limited number of cases, to check both results accuracy and user experience: once validated and proven effective, all products have been migrated to the code, which is now the standard for all Flight Mechanics performance activities.

In the turn of almost five years, more than 80 qualified and documented models have been produced. The code has been used for more than 400 design or calculation tasks by a pool of 20 specialists, providing in the initial 2~3 years validation experience and valuable improvement requests, addressed by the consultants of Line-Up.

The most effective characteristics of the code considered by the users are the flexibility of the possible evaluations, the modularity allowing continuous introduction of improvements or even possibility of full new configurations, as well as the steep learning curve allowed by the advanced GUI and the completeness of the documentation. An improvement area has been also identified in the possibility to allow some kind of automation or scheduling of routine calculations for standard assessments.

One of the most complete examples of this successful development has been the optimization of a full set of missions for pollutants and fuel consumption reduction performed in the framework of CleanSky1 GRC research project. Finally another success case has been the implementation of a full tiltrotor modelling, used for current and future company products of this class.

ACKNOWLEDGMENTS

The Authors want to thank the Company and the Clean Sky Project for the recognition of the innovative value of this work, as well as all the Colleagues and who contributed to this achievement.