JAVA FRAMEWORK FOR PARAMETRIC AIRCRAFT DESIGN PERFORMANCE ANALYSIS MODULE

Vittorio Trifari
Dept. of Industrial Engineering,
University of Naples "Federico II"
21 Via Claudio 80125Naples (Italy)
vittoriotrifari90@gmail.com

Manuela Ruocco
Dept. of Industrial Engineering,
University of Naples "Federico II"
21 Via Claudio 80125Naples (Italy)
manuela.ruocco@aol.com

Vincenzo Cusati
Dept. of Industrial Engineering,
University of Naples "Federico II"
21 Via Claudio 80125 Naples (Italy)
vincenzo.cusati@gmail.com

Abstract. The aim of this paper is to provide a comprehensive overview about the development in Java of a library, named JPAD, dedicated to the preliminary design of an aircraft, focusing on the take-off and landing performance analysis modules. To validate the results of each module, two parametric aircraft models have been taken into account: one related to a regional turboprop similar to the ATR-72 and another related to a transport jet similar to a B747-100B. Tests upon each of these models have been performed in order to compare the obtained output with the performance data of the two aircrafts retrieved from the related flight manuals or public brochures.

Keywords. Aircraft design, Java framework, performance analysis.

1 Introduction

Take-off and landing are two performance characteristics of primary importance during the preliminary design phase of any aircraft since they usually provide critical design constrains; for example, if the required runway length is too long, the aircraft cannot take-off with full fuel or full payload and its economics are compromised. So take-off and landing performance play a significant role in both conceptual and preliminary design phases of an aircraft because they may be both design requirements to be fulfilled, specified by the Federal Aviation Regulations (FAR) and by the customer, both driving parameters in the definition of the design point. In particular, the take-off performance provides a limitation both in the wing loading W/S and in the T/W (or the W/P) ratio; while the landing performance imposes a limitation only on the wing loading.

This paper deals with the description of the development of a versatile and rapid calculation tool for estimating these performance. The latter is part of more complex Java library, named JPAD, designed as a fast, reliable and user friendly computational aid for aircraft designers in the conceptual and preliminary design phases. The ultimate goal of such a tool is to perform a parametric and multi-disciplinary analyses of an aircraft and then search for an optimized configuration. An important design requirement is related to its interoperability with other engineering analysis tools; in fact, the application can be easily integrated into a comprehensive aircraft optimization cycle as shown in Figure 1.1.

The choice of the Java language derives from the following key points. This language is widely supported by Oracle and a huge community of developers so that the problem of having an obsolete library due to aging is avoided. The Java language promotes the use of open source libraries which provide a very simple management of input and output tasks as well as complex mathematical operations; furthermore, the language and the companion IDE provide a widely supported GUI framework and a GUI visual builder. Finally, the language promotes modularity so that is easier to work with an ever changing team.

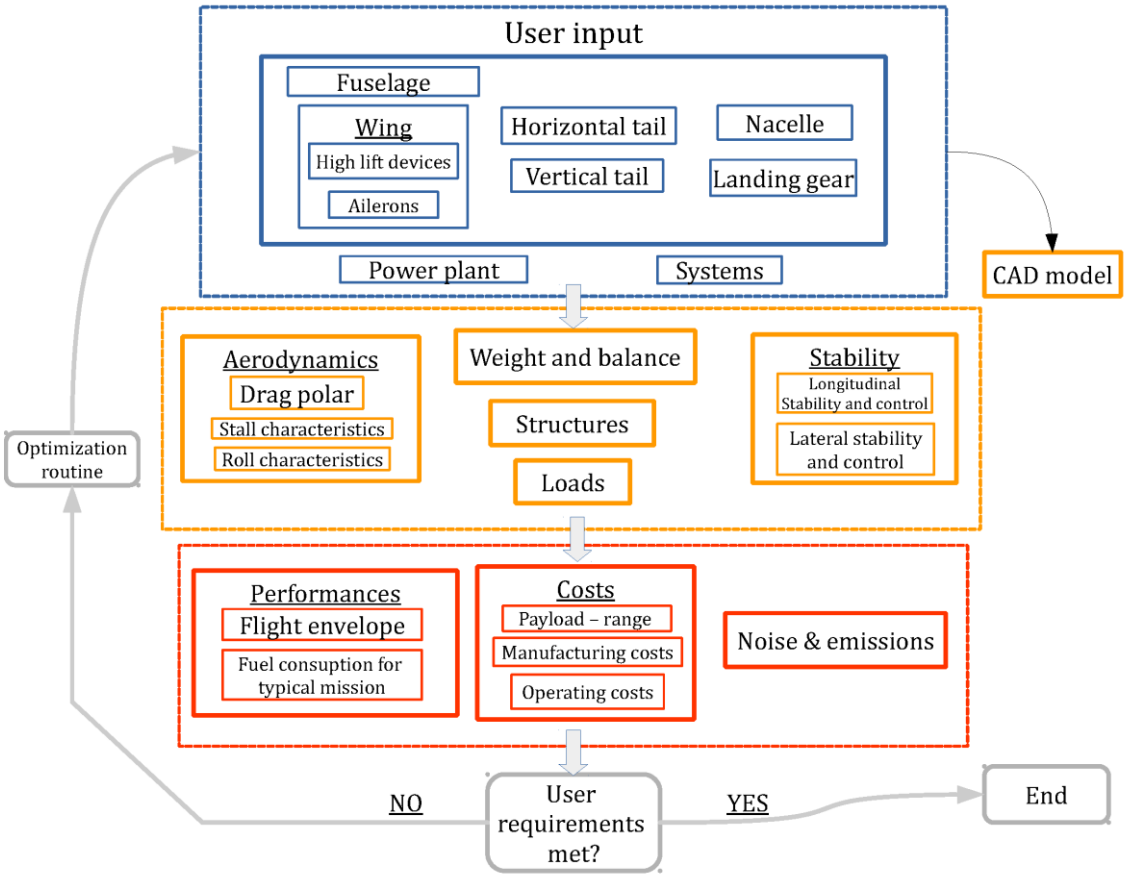

Figure 1.1: JPAD flowchart.

2 Take-off and landing performance calculation

The calculation of the take-off and landing distances have been implemented using a simulation-based approach. This expects to solve an appropriate set of ordinary differential equations (ODE), which describes the aircraft equations of motion during all the take-off and landing phases. In this case the use of the Java language has proven to be very performing as it allowed to use dedicated external libraries from the Internet which provided a fast and accurate tool to solve a complex mathematical problem.

2.1 Take-off

In order to deal with the calculation of the take-off distance, a smart strategy is to find out all the fundamentals variables, which describes completely the aircraft state during this phase, and then study the dynamic system in exam in a state-space representation. These latter can be resumed in the followings.

- Position (s)
- Speed (V)
- Ramp angle (γ)
- Altitude (h)

The set of Ordinary Differential Equations (ODE) that models the take-off run may be written in the following form:

$$\begin{Bmatrix} \dot{s} \\ \dot{V} \\ \dot{\gamma} \\ \dot{h} \end{Bmatrix} = \begin{Bmatrix} f_1(s,\ V,\ \gamma,\ h;\ \alpha) \\ f_2(s,\ V,\ \gamma,\ h;\ \alpha) \\ f_3(s,\ V,\ \gamma,\ h;\ \alpha) \\ f_4(s,\ V,\ \gamma,\ h;\ \alpha) \end{Bmatrix} \quad \text{with} \quad \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix} = \begin{Bmatrix} s \\ V \\ \gamma \\ h \end{Bmatrix} \quad \text{and} \quad u = \alpha \tag{2.1}$$

These equations can be also written in a more concise way as shown below. The unknown $\mathbf{x} = [x_1,\ x_2,\ x_3,\ x_4]^{\mathrm{T}}$ is the vector of state variables.

$$\dot{\mathbf{x}} = f(\mathbf{x}\ ;\ u) \tag{2.2}$$

The input $u(t)$ is a given function of time, for $0 \le t \le t_{final}$, that corresponds to an assumed time history of the angle of attack during the take-off rotation and airborne phases. The right-hand sides of system (2.1) are defined by the following functions:

$$f_1(\mathbf{x},u) = x_2 \tag{2.3a}$$

$$f_2(\mathbf{x},u) = \frac{g}{W}\begin{cases} T(x_2) - D(x_2,u) - \mu\left[W - L(x_2,u)\right], & \text{if } n < 1 \\ T(x_2)\cos(u) - D(x_2,u) - W\sin(x_3), & \text{if } n \ge 1 \end{cases} \tag{2.3b}$$

$$f_3(\mathbf{x},u) = \frac{g}{W\ x_2}\begin{cases} 0, & \text{if } n < 1 \\ L(x_2,u) + T(x_2)\sin(u) - W\cos(x_3), & \text{if } n \ge 1 \end{cases} \tag{2.3c}$$

$$f_4(\mathbf{x},u) = x_2\sin(x_3) \tag{2.3d}$$

The thrust $T(x_2)$ is calculated by means of the interpolating function $T_{tab}(V_a)$ based on a table lookup algorithm, where $V_a = V + V_w$ is the airspeed and V_w is the wind speed (horizontal component, positive if opposite to the aircraft motion).

The drag $D(x_2,u)$ and lift $L(x_2,u)$, as functions of airspeed V_a and angle of attack, are given by the following conventional formulas.

$$D(x_2,u) = \frac{1}{2}\rho\left[x_2 + V_w\cos(x_3)\right]^2 S\ C_D(u) \tag{2.3e}$$

$$L(x_2,u) = \frac{1}{2}\rho\left[x_2 + V_w\cos(x_3)\right]^2 S\ C_L(u) \tag{2.3f}$$

The switching function n of aircraft velocity and angle of attack represents the load factor and is defined as follows:

$$n(x_2,u) = \frac{L(x_2,u)}{W\cos(x_3)} \tag{2.3g}$$

The formulas (2.3) make the system (2.2) a closed set of ODE. When the function $u(t)$ is assigned and the system is associated to a set of initial conditions, in this particular case equal to $\mathbf{x_0} = [0,\ 0,\ 0,\ 0]^{\mathrm{T}}$, a well-posed Initial Value Problem (IVP) is formed, which can be solved numerically.

It has to be highlighted that the lift coefficient $C_L(u)$ is the one from the lift curve with flaps, and eventually slats, deflected; while the drag coefficient $C_D(u)$ that appears in (2.3e) can be modeled as follows.

$$C_D(u) = C_{D0} + (\Delta C_{D0})_{\substack{\text{flap,}\\ \text{landing gear}}} + K_g\left(\frac{C_L^{\,2}(u)}{\pi AR\,\mathrm{e}}\right) \tag{2.4}$$

The term K_g in (2.4) incorporates the ground effect and it is calculated from [1] using the (2.5) which is a fifth order interpolating function of the graph in Figure 2.1, where the ratio h_w/b is obtained dividing the height of wing above the ground by the wing span, usually between 0.1 and 0.2 when the aircraft is on the ground and assumed as $h_w = h$ during the airborne.

$$K_g = -622.44x^5 + 624.46x^4 - 255.24x^3 + 47.105x^2 - 0.6378x + 0.0055 \tag{2.5}$$

This polynomial equation has a coefficient of determination R^2 of 0.9999 which justifies the approximation.

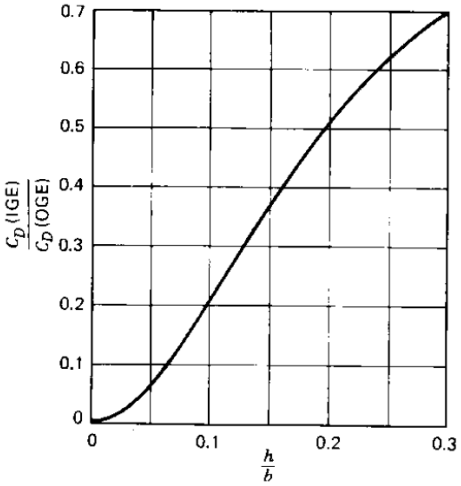

Figure 2.1: Ground effect parameter K_g as function of the h_w/b ratio.

The function $u(t)$, which represents the input law of the angle of attack, can be constructed by picking the time t_{Rot} when the rotation speed V_{Rot} is reached along the ground roll; thus the $u(t)$ function can be defined as follows.

$$u(t) = \begin{cases} \alpha_g & ,\ if\ \ n < 1 \\ \alpha_1(t) & ,\ if\ \ n \geq 1 \end{cases} \tag{2.6}$$

In the (2.6) a constant α_g during the ground roll phase up to the rotation speed, and a given non-zero law $\alpha_1(t)$ for the post-rotation angle of attack time history are assumed.

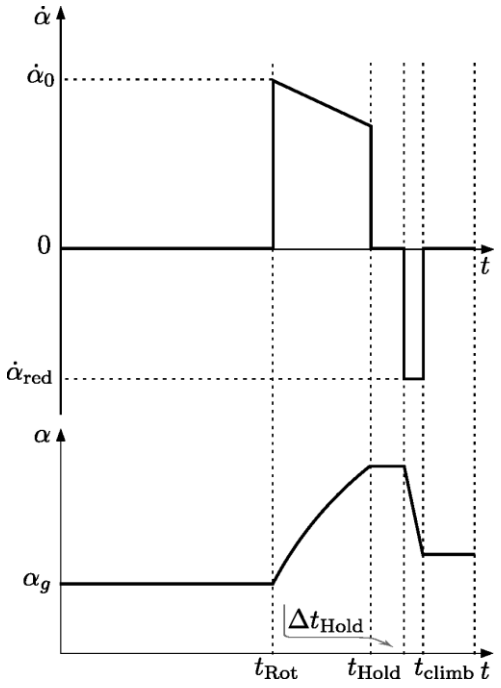

Figure 2.2: Qualitative representation of the angle of attack input law.

Figure 2.2 shows a qualitative representation of the $\alpha_1(t)$ law. As can be seen, after t_{Rot} the pilot applies an initial angular velocity $\dot{\alpha}_0$, which decreases with time, according to the law written in (2.7) as function of α, until the time t_{Hold} has been reached; this particular instant is related to the achievement of the maximum admitted lift coefficient in take-off configuration, which is set at 90% of the $C_{Lmax,\,TO}$.

$$\dot{\alpha} = \dot{\alpha}_0 (1 - k_\alpha \alpha) \tag{2.7}$$

In equation (2.7), the k_α slope is assigned (expressed in 1/° and dependent on the aircraft in exam), while the initial angular velocity $\dot{\alpha}_0$ is calculated as follows.

$$\dot{\alpha}_0 = \frac{\Delta\alpha}{\Delta t_{Rot}} = \frac{\alpha_{LO} - \alpha_g}{\Delta t_{Rot}} \tag{2.8}$$

where α_{LO} can be obtained from the lift curve of the wing, with flaps deflected in take-off configuration, by assigning the value of the $C_{L,\,LO}$; this can be derived from the $C_{Lmax,\,TO}$ dividing it by the parameter K^2_{LO}, which represents the quantity that has to be multiplied by the stall speed in take-off configuration in order to obtain V_{LO} (for example 1.1 with reference to the FAR-25 requirements).

From this point on the pilot stops the pitching maneuver and keeps the angle of attack constant for an assigned Δt_{Hold}. During this time interval, the lift coefficient is high and, as a result, also the induced drag is high so that aircraft acceleration will reduce.

After this short time interval, the pilot has to reduce the angle of attack in order to avoid the acceleration to decrease too much and so an assigned negative angular velocity $\dot{\alpha}_{\text{red}}$ is applied; the latter assumed to be constant for simplicity.

Finally, since the decrease of α determines also a reduction in C_L, the time t_{climb} will be reached when the load factor is reduced to 1; this means that a balance of the forces, perpendicular to the flight path, has been achieved and so the climb phase, at constant γ, can begin leaving α constant and equal to last value reached.

Moreover, from this time on, the lift value is constant and equal to $W\cos(\gamma)$ in order to maintain the load factor equal to 1; while the C_L is derived from the lift value using the (2.9).

$$V = \sqrt{\frac{2\,W}{\rho\,S\,C_L}} \tag{2.9}$$

2.1.1 OEI take-off distance and balanced field length (BFL)

The calculation of the take-off distance in OEI condition is quite the same as the one explained previously, with the difference that now there is a discontinuity in thrust due to the broken engine.

In particular, the thrust $T(x_2)$ will still be read from the database but considering a number of engines reduced by one from the time t_{ef} at which the engine failure occurs.

In case the take-off have to be aborted due to an engine failure that occurs at low speeds, the calculation of the aborted take-off length follows a different path.

The portion of the aborted take-off run up to the engine failure velocity V_{ef} is calculated in the same way as that for the continued take-off, so that the distance is the same in both cases. From this point on, until the pilot reacts by activating brakes, there is only a discontinuity in thrust due to the failed engine; while, after the time interval in which the pilot decides to abort the take-off, the thrust is set to idle (ideally zero) and the brakes action provides a higher friction coefficient. During this last phase, the equation (2.3b) changes in the following.

$$f_2(\mathbf{x},u) = \frac{g}{W}\left\{-D(x_2,u) - \mu_{\text{brakes}}\left[W - L(x_2,u)\right]\right\} \tag{2.10}$$

Here μ_{brakes} is bigger than μ and it is usually about 0.3; furthermore, it has to be noted that, even if the aircraft in exam is supplied with a reverse thrust device, this effect has not to be taken into account for a more conservative result.

Instead of considering the limiting cases of aborting take-off at low speeds and continued take-off at high speeds, it is useful to determine the critical velocity at which the distance required to continue the take-off is equal to the distance required to safely abort it. This particular velocity is called decision speed V_1, while the related distance is called balanced field length (BFL).

In order to calculate the BFL, and the related velocity V_1, it is possible to evaluate at different failure speeds both the continued take-off distance with one inoperative engine, both the aborted take-off distance. Each couple of speed and distance can then be plotted with the result of building the

curves of Figure 2.3. The intersection of these latter, at which the two distances are the same, defines the balanced field length and the decision speed V_1.

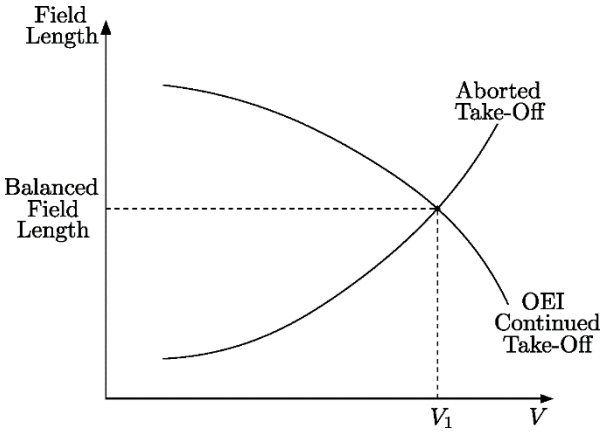

Figure 2.3: Qualitative representation of the field distances required to continue a take-off or to abort it when one engine fails as a function of aircraft speed.

2.1.2 Take-off Java class overview

In order to implement these calculation methods inside the Java library, a class named `CalcTakeOff` has been created. The most important method of the class is `calculateTakeOffDistanceODE` which is in charge of the resolution of the ODE set presented in (2.2). This method accepts as input the following parameters.

- `vFailure`, a `Double` value representing the failure speed in ms^{-1}. Can be set to `null` if the user doesn't want to calculate the OEI take-off distance.
- `isAborted`, a `boolean` flag which is true if the method has to calculate the distance related to an aborted take-off.

After performing a check upon these two variables, the method knows which case has to be analyzed and proceeds with the calculation; in particular, it creates the integrator object of the class `HighamHall54Integrator`, which implements the interface `FirstOrderIntegrator`. For more information, the reader can refer to [2].

The `HighamHall54Integrator` class implements a fifth order Higham and Hall integrator which uses seven functions evaluations per step and is supplied with step size control, automatic step initialization and continuous output. This integrator has proven to be the best choice since it provides the best compromise between calculation time and accuracy using the settings of Listing 2.1.

```
FirstOrderIntegrator theIntegrator = new HighamHall54Integrator(
                    1e-6,     // minimal step
                    1,        // maximal step
                    1e-17,    // allowed absolute error
                    1e-17     // allowed relative error
                    );
```

Listing 2.1: `HighamHall54Integrator` class object creation.

Beside the integrator, the method needs the set of equation to integrate; these are passed to it through the object of a dedicated inner class, named `DynamicsEquationsTakeOff`, which implements the interface `FirstOrderDifferentialEquations` [3].

In order to take into account of particular events which can happen during the take-off phase, the method `calculateTakeOffDistanceODE` is supplied with several implementations of the interface `EventHandler` [2]. The latter, through the definition of a specific function, can determine the occurrence of the wanted event by monitoring whether the sign of the defined function changes.

In the case in exam, the following six events are monitored.

`ehCheckFailure`	It checks when the speed x_2 becomes greater than the input `vFailure` determining, in this way, the instant of the engine failure occurrence.
`ehCheckVRot`	It checks when the speed x_2 becomes greater than the rotation speed V_{Rot} determining, in this way, the instant at which the ground roll ends and the rotation phase begins.
`ehEndConstantCL`	It checks when the time t becomes greater than the sum of t_{Hold} and of the given time interval Δt_{Hold} determining, in this way, the instant at which the angle of attack, and the related C_L, stops to be kept constant. (only if `isAborted` is false)
`ehCheckObstacle`	It checks when the altitude x_4 becomes greater than the given obstacle height (35 ft) determining, in this way, the instant at which the airborne phase, and so the entire take-off, ends. (only if `isAborted` is false)
`ehCheckBrakes`	It checks when the time t becomes greater than the sum of t_{Failure} and of the given time interval Δt_{Rec}, required to the pilot to recognize the failure, determining, in this way, the instant at which the pilot, who has decided to abort the take-off, activates the brakes in order to stop the aircraft. (only if `isAborted` is true)
`ehCheckStop`	It checks when the speed, x_2, becomes lower than zero determining, in this way, the instant at which the aircraft has stopped. (only if `isAborted` is true)

Table 2.1.: `EventHandler` implementation inside the method `calculateTakeOffDistanceODE`.

Each event of the Table 2.1 defines a time instant usable, by the class `DynamicsEquationsTakeOff`, to determine when the derivatives, or the calculation of the related physical quantities, have to switch from an equation to another. This allows to manage in a very easy way the definition of the profile of $\dot{\alpha}$ and α, as well as the derivatives change shown in (2.3b) and (2.3c).

Another important feature that the ODE package provides is the possibility to manage each time step, even if no event is triggered; in this way the developer can, for example, store data into an output file, or manage some events, which are independent from the time or the state vector, as they were in the `EventHandler` interface.

The tool which allows all these feature is the `StepHandler` interface [2]; in this particular case, this interface has only one implementation, added to the integrator, which is in charge of store the state vector, the time and all the related physical quantities, into their related `Lists` [4], at every time step in order to make them usable outside this method. Moreover, it has a key role in managing three events, to be observed only if the variable `isAborted` is false, that could not be handled well by the `EventHandler` interface.

- A check upon the load factor to catch the instant at which, for the first time, it reaches a value of 1; this instant is t_{EndRot} and determines the beginning of the airborne phase together with the changes in the derivatives shown in (2.3b) and (2.3c).

- A check upon the C_L in order to determine when it reaches the threshold value defined by $K_{CL\max}$ (usually 90%) multiplied for the $C_{L\max,\,TO}$. The related instant is the t_{Hold} of the beginning of the constant α and C_L phase.
- A second check on the load factor in order to define the instant at which its value is reduced to 1 after having applied the constant $\dot{\alpha}_{\text{Red}}$ angular velocity. This instant defines t_{climb}.

In conclusion, having a method for calculating the take-off distance in every case, it is possible to use it iteratively in order to determine the balanced field length (BFL) and the related decision speed V_1 giving as input to `calculateTakeOffDistanceODE` an array of failure speeds (from $2ms^{-1}$ to the lift-off speed) and computing both the case of aborted and continued take-off.

2.2 Landing

Similar to the take-off, the landing phase can be considered as made up of two main components: an air run and a ground run.

The air run may be considered to start at the an obstacle height of 50 ft and at an approach speed V_a which, according to FAR-25, must be at least 1.3 times the stall speed in landing configuration (V_s). The trajectory of the approach phase is supposed to be linear and when the speed is reduced to a value of $1.2\div1.25V_s$, named V_{Flare}, the aircraft begins a gradual flare rotation of large radius R turning the pitch angle ϑ from 2°÷3° down to 0° and reducing the speed to V_{TD} at which the wheels touches the runway. The total air run is the sum of the approach and the flare distances as shown in Figure 2.4.

To calculate this distance the simplest way is to assume that the flare rotation describes a circular trajectory and that ϑ_a, during approach, is small and almost equal to ϑ_f. By using these hypotheses the two distances S_a and S_{Flare} can be calculated through a geometrical approach which follows the steps below.

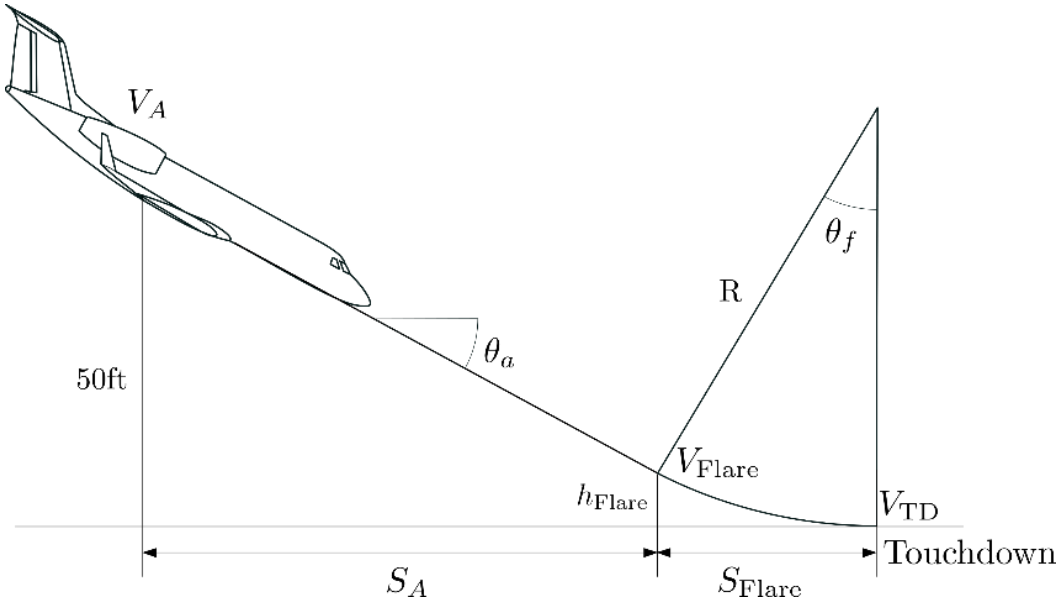

Figure 2.4: Scheme of the air run of the landing phase.

- Assuming a load factor n =1.2 and a V_{Flare} equals to the mean value between $V_a = 1.3\,V_s$ and $V_{TD} = 1.15\,V_s$, the radius R can be calculated as follows.

$$R = \frac{V^2}{g(n-1)} = \frac{V^2_{\text{Flare}}}{0.2\ g} \tag{2.11}$$

- With ϑ_a around 2°÷3° is possible to calculate the altitude at the beginning of the flare rotation.

$$h_{\text{Flare}} = R\left[1-\cos(\vartheta_a)\right] \tag{2.12}$$

- At this point, the two distances S_{A} and S_{Flare} can be calculated using the following expressions.

$$S_{\text{A}} = \frac{50-h_{\text{Flare}}}{\tan(\vartheta_a)}; \qquad S_{\text{Flare}} = R\ \sin(\vartheta_a) \tag{2.13}$$

The ground run begins when the aircraft touches the ground and ends when the speed is equal to zero. In order to calculate this distance, the same approach used for the take-off may be applied with the difference that now only two state variables are taken into account: the position and the speed.

$$\begin{Bmatrix} \dot{s} \\ \dot{V} \end{Bmatrix} = \begin{Bmatrix} f_1(s,V) \\ f_2(s,V) \end{Bmatrix} \quad \text{with} \quad \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} = \begin{Bmatrix} s \\ V \end{Bmatrix} \tag{2.14}$$

The right-hand side of the ODE set can be modeled as described in the (2.15).

$$f_1(\mathbf{x}) = x_2 \tag{2.15a}$$

$$f_2(\mathbf{x}) = \frac{g}{W}\begin{cases} -D(x_2) - \mu\left[W - L(x_2)\right] & ,\ if\ \ t \le t_{\text{free roll}} \\ T_{\text{Rev}}(x_2) - D(x_2) - \mu_{\text{brakes}}\left[W - L(x_2)\right] & ,\ if\ \ t > t_{\text{free roll}} \end{cases} \tag{2.15b}$$

The drag and lift values are calculated in the same way of the take-off ground roll with the difference that now also the spoilers contribute is added (this can be calculated as explained in [5]), while T_{Rev} is a reverse thrust produced by thrust reversers, for jet engine, or by a particular regulation of the blade pass, for a propeller engine. The latter is usually neglected for the ground distance calculation because it helps the deceleration of the aircraft, but for more clarity, it has been considered in the equations.

Furthermore, the instant $t_{\text{free roll}}$ is related to the end of the free-roll distance and it is decide by the user. During this short time, the thrust is not really zero but it is the one calculated with engines in idle setting; however, since this little contribution is usually of the same magnitude of the rolling resistance of the landing gears, these two terms have both been neglected.

When both the ground distance, both the airborne phase distance are calculated, it is finally possible to determine the total landing distance by summing these latter.

2.2.1 Landing Java class overview

The guideline followed for implementing the Java class in charge of the landing distance calculation is very similar to the one used in building the take-off class.

In order to manage the only significant event, which is when the aircraft reaches a speed equal to zero, only one `EventHandler` has been implemented following the structure of `ehCheckStop` defined for the aborted take-off run. Moreover, all the physical quantities related to the ground run are stored in the related `Lists` [4] at every integration step using the an instance of the interface `StepHandler`.

The peculiarity of this case, unlike the take-off, is that the integration starts from the end of the flare phase, with an initial position and a speed different from zero. This means that this method requires having previously carried out the calculation of the airborne distance, or having previously assigned values to S_A and S_{Flare}.

3 Case study: ATR-72 and B747-100B

In order to validate the calculation methods for both the take-off and landing performance, two parametric aircraft models have been defined inside JPAD; the first one similar to the regional turboprop ATR-72 and the second one similar to the jet airliner B747-100B.

The results obtained for the FAR-25 take-off and landing field lengths have, firstly, been compared with the flight manuals data of the two analyzed aircrafts in order to evaluate the calculation accuracy.

Furthermore, a sensibility analysis has been carried out, using the B747-100B model, with the aim of comparing the JPAD output with the statistical field lengths at different values of two main design parameters: the wing loading W/S, for both take-off and landing, and the thrust ratio T/W, only for the take-off.

3.1 ATR-72

Concerning the take-off analysis of the ATR-72 model, the following input data have been assigned in order to describe the maneuver parameters and to allow the Java class to perform the calculation of some important quantities like the stall speed, all the characteristic speeds of the take-off phase, the ground effect parameter and the effects of high lift devices on the drag coefficient and lift coefficient in take-off configuration.

Focusing on these latter, the Java class in charge of the take-off calculation uses another class of the JPAD library which allows to evaluate the high lift devices effects on a wing using semi-empirical methods from literature. In particular, for this aircraft two single slotted flaps have been chosen, both with a deflection of 20° leading to a $C_{Lmax,\,TO}$ of 2.05.

`dtRot`	The duration of the rotation phase.	3 s
`dtHold`	The duration of the constant C_L phase.	0.5 s
`kRot`	The coefficient by which the stall speed in take-off configuration has to be multiplied in order to obtain the rotation speed.	1.05
`kAphaDot`	The slope of the $\dot{\alpha}$ curve described in the (2.7).	0.05
`kcLMax`	The percentage of the $C_{Lmax,\,TO}$ that must not be surpassed.	0.85
`kLO`	The coefficient by which the stall speed in take-off configuration has to be multiplied in order to obtain the lift-off speed.	1.13
`kFailure`	The drag increment due the failed engine.	1.1
`alphaReductionrRate`	The negative $\dot{\alpha}_{red}$ described in Figure 2.2.	-3 deg/s
`mu`	The friction coefficient between wheels and runway.	0.03
`muBrake`	The friction coefficient when the brakes are activated.	0.5

(continued on next page)
(continued from previous page)

`phi`	The throttle setting.	1.0
`deltaCD0LandingGear`	The C_{D0} contribution of the landing gears.	0.014
`wingToGroundDistance`	The wing distance from ground.	4 m
`obstacle`	The obstacle height to be surpassed.	35 ft
`vWind`	The wind velocity along the runway.	0.0 m/s
`alphaGround`	The fuselage angle of attack when it is on the runway.	0 °
`iw`	The wing angle of incidence referred to the fuselage.	1.5 °

Table 3.1.: ATR-72 input data of the take-off analysis.

Using the input data of Table 3.1, speeds and distances of Table 3.2 have been calculated. Furthermore, the charts from Figure 3.1 to Figure 3.6 describe the evolution of some remarkable physical quantities during the take-off considering the AOE condition.

Stall speed in take-off configuration (V_s)	53.73 m/s
Rotation speed (V_{rot})	56.41 m/s
Decision speed (V_1)	58.73 m/s
Lift-off speed (V_{LO})	60.71 m/s
Take-off safety speed (V_2)	65.40 m/s
Ground roll distance	766.4 m
Rotation distance	200.04 m
Airborne distance	245.8 m
Take-off distance in AOE condition	1221.25 m
FAR-25 take-off field length (Take-off distance AOE * 1.15)	1404.44 m
Balanced field length	1532.15 m

Table 3.2.: ATR-72 output data of the take-off analysis.

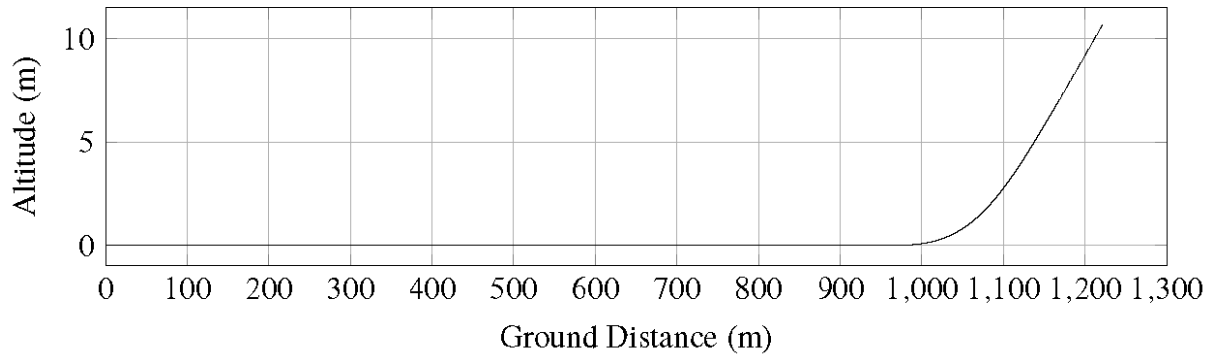

Figure 3.1: Take-off trajectory in AOE condition - ATR-72.

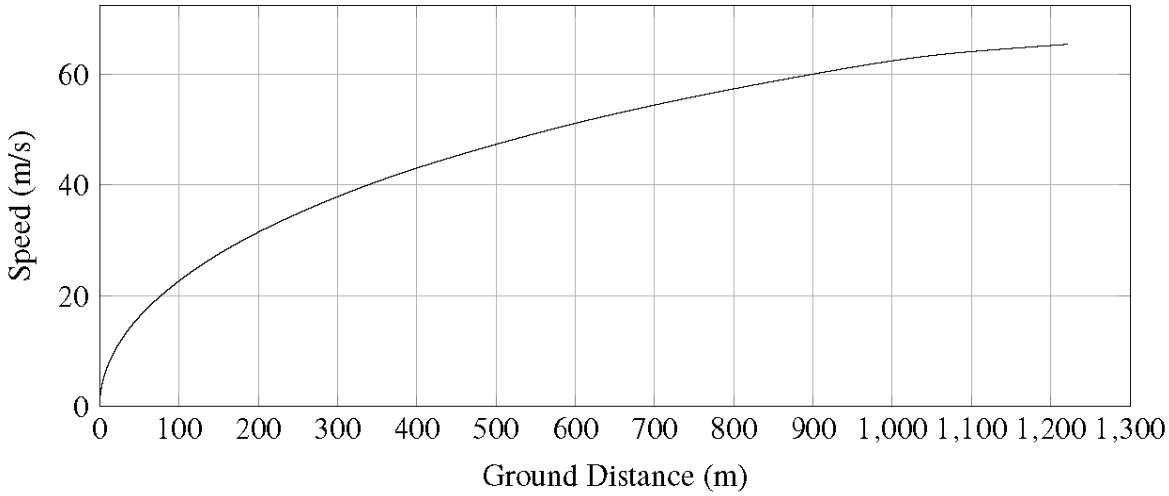

Figure 3.2: Speed v.s. ground distance in AOE condition - ATR-72.

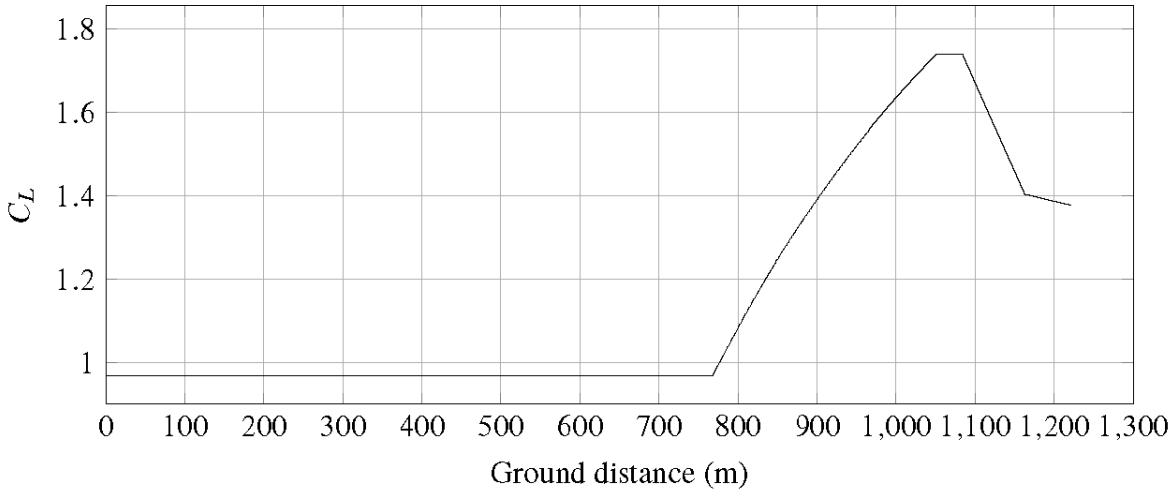

Figure 3.3: Lift coefficient v.s. ground distance in AOE condition - ATR-72.

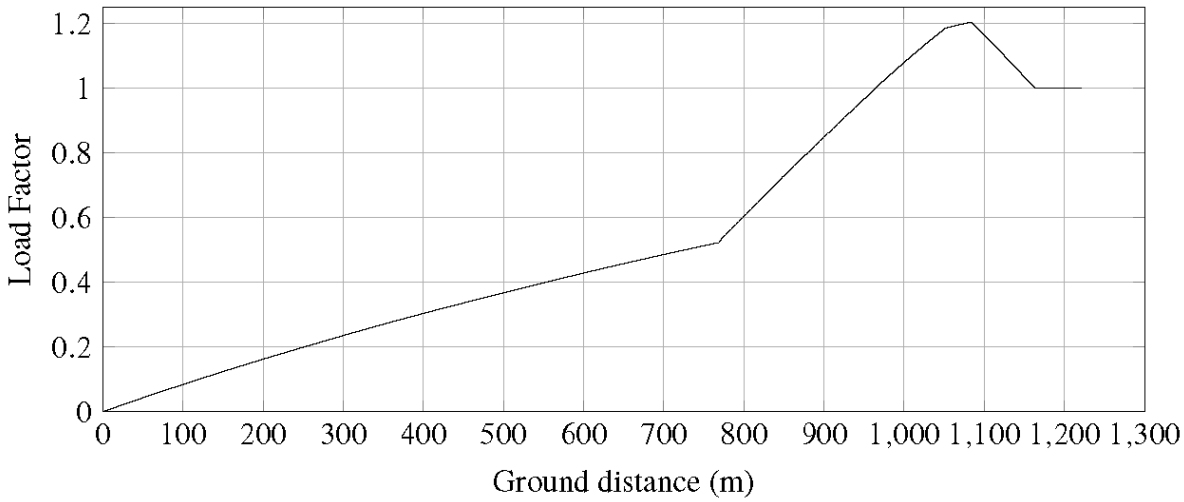

Figure 3.4: Load factor v.s. ground distance in AOE condition - ATR-72.

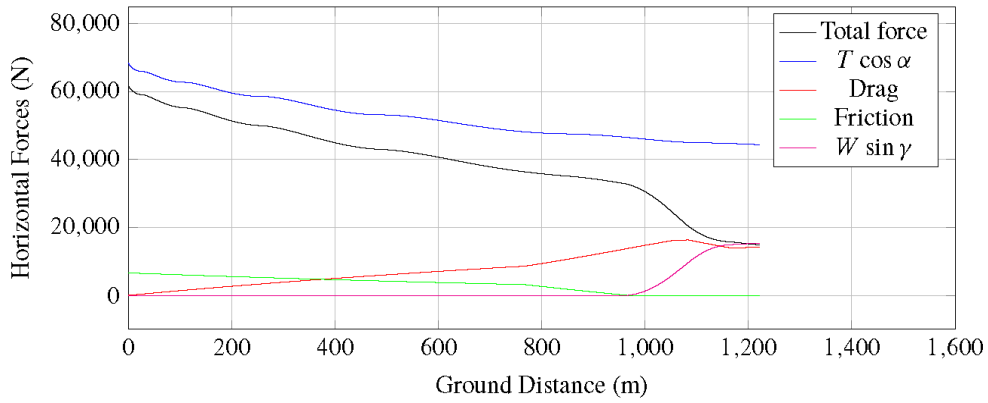

Figure 3.5: Forces v.s. ground distance in AOE condition - ATR-72.

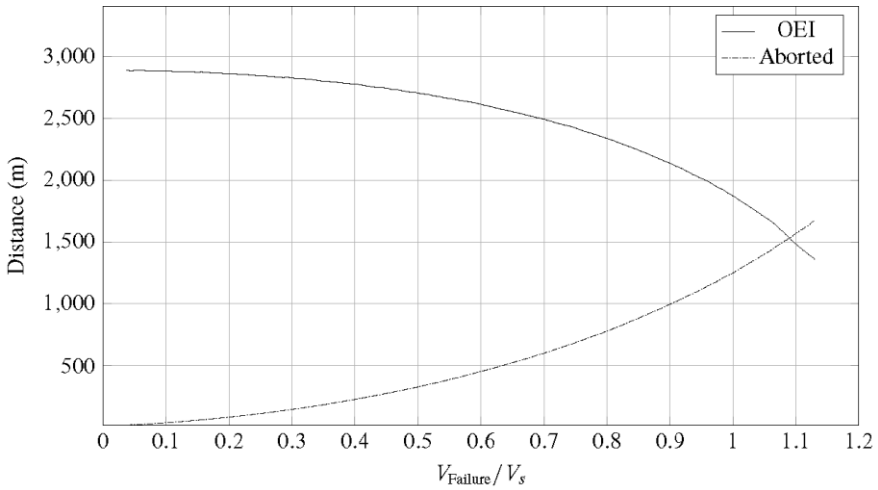

Figure 3.6: Balanced field length chart - ATR-72.

Concerning the landing analysis of the ATR-72, the input data of Table 3.3 have been assumed. Furthermore the flaps deflection has been set to 40° obtaining a $C_{Lmax,\,LND}$ of 2.63.

`kA`	The coefficient by which the stall speed in landing configuration has to be multiplied in order to obtain the approach speed.	1.3
`kFlare`	The coefficient by which the stall speed in landing configuration has to be multiplied in order to obtain the flare speed.	1.23
`kTD`	The coefficient by which the stall speed in landing configuration has to be multiplied in order to obtain the touchdown speed.	1.15
`mu`	The friction coefficient between wheels and runway.	0.03
`muBrake`	The friction coefficient when the brakes are activated.	0.5
`phiRev`	The reverse thrust throttle setting.	0.25
`deltaCD0LandingGear`	The C_{D0} contribution of the landing gears.	0.014
`deltaCD0Spoiler`	The C_{D0} contribution of the spoilers.	0.011
`wingToGroundDistance`	The wing distance from ground.	4 m
`obstacle`	The obstacle height from which the approach phase starts.	50 ft
`vWind`	The wind velocity along the runway.	0.0 m/s
`alphaGround`	The fuselage angle of attack when it is on the runway.	0 °
`iw`	The wing angle of incidence referred to the fuselage.	1.5 °
`nFreeRoll`	The duration of the free-roll phase.	2 s
`thetaApproach`	The pitch angle during the approach phase.	4 °

Table 3.3.: ATR-72 input data of the landing analysis.

As for the take-off, the Table 3.4 shows the results of the analysis, while the charts from Figure 3.7 to Figure 3.8 describe the evolution of some important physical quantities during the landing phase.

Stall speed in landing configuration (V_s)	45.52 m/s
Approach speed (V_A)	59.17 m/s
Flare speed (V_{Flare})	55.99 m/s
Touchdown speed (V_{TD})	52.35 m/s
Approach distance	162.27 m
Flare distance	111.49 m
Ground roll distance	414.13 m
Landing distance	687.88 m
FAR-25 landing field length (Landing distance/0.6)	1146.47 m

Table 3.4.: ATR-72 output data of the landing analysis.

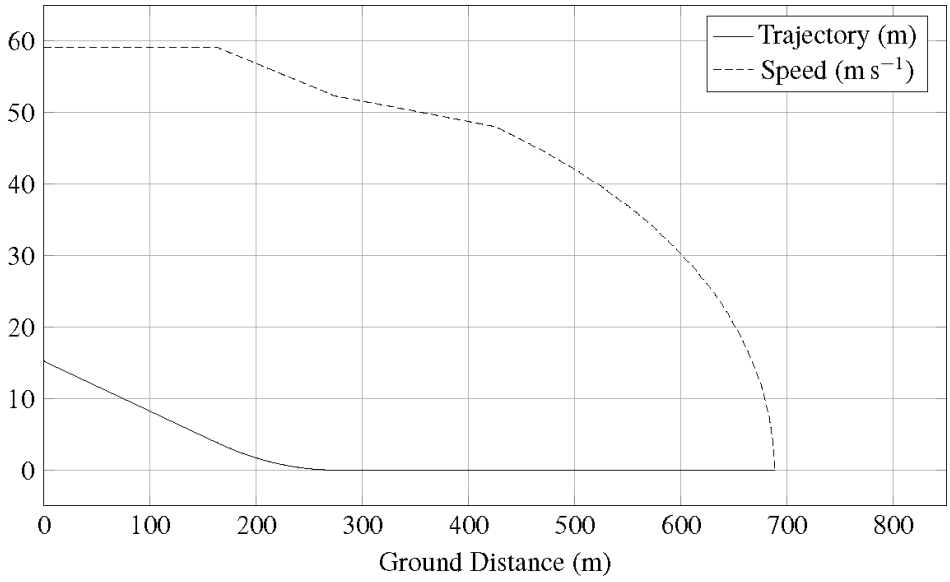

Figure 3.7: Landing trajectory and speed evolution - ATR-72.

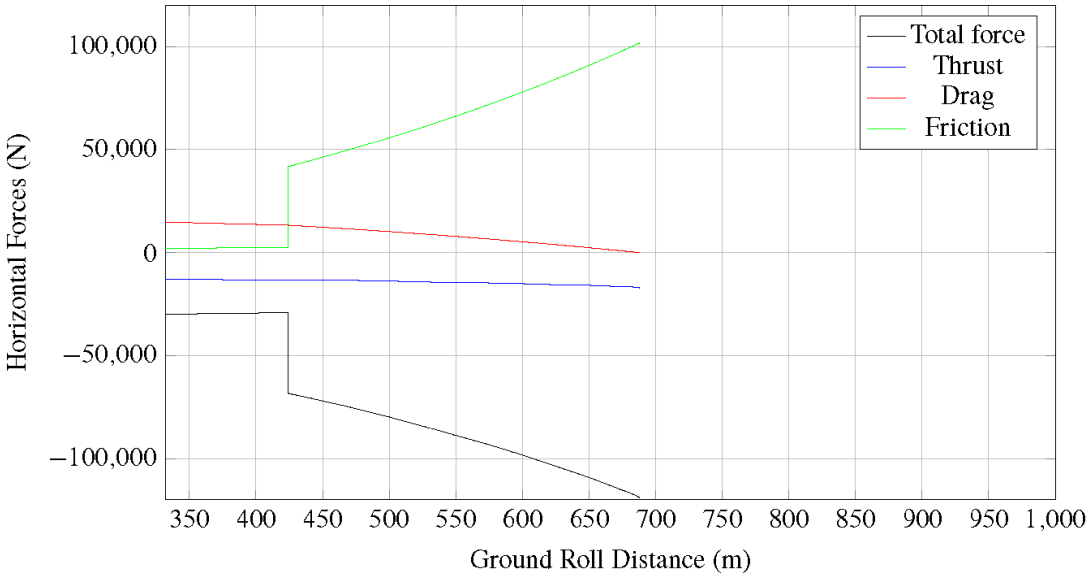

Figure 3.8: Forces evolution during landing phase - ATR-72.

In conclusion, with the purpose of validating the results obtained, the Table 3.5 shows the comparison between the JPAD output and public domain data from the ATR-72 brochure concerning the FAR-25 take-off and landing field length.

ISA CONDITION / SEA LEVEL MTOM = 22500 kg / MLM = 22350 kg	**Public data**	**JPAD**	**Difference (%)**
FAR-25 take-off field length	1300 m	1404 m	≈ 7%
FAR-25 landing field length	1067 m	1146 m	≈ 7%

Table 3.5.: JPAD results comparison with known data.

3.2 B747-100B

The analysis that will be shown and commented in this subsection will regard a sensibility analysis of the FAR-25 take-off and landing field lengths, calculated using JPAD, with respect the wing loading W/S, for both take-off and landing, and the thrust ratio T/W, for the take-off only.

Starting from the flight manual data regarding the maximum take-off and landing weights, and knowing the wing geometry, the two wing loadings, in both take-off and landing conditions, have been derived. At this point, a variation of these latter from -10% to +20% (for the take-off) and from -10% to +10% (for the landing) has been assigned obtaining the results of Table 3.6.

Model data		**W/S variation range in take-off**	**W/S variation range in landing**
$\mathbf{W_{TO}}$	750000 lb	123 lb/ft^2	97 lb/ft^2
$\mathbf{W_{LND}}$	590654 lb	136 lb/ft^2	107 lb/ft^2
S	5500 ft^2	150 lb/ft^2	118 lb/ft^2
$\mathbf{(W/S)_{TO}}$	136 lb/ft^2	164 lb/ft^2	
$\mathbf{(W/S)_{LND}}$	107 lb/ft^2		

Table 3.6.: Wing loading variation in take-off and landing starting from the B747-100B data.

Regarding the thrust ratio T/W, the public data for the B747-100B power plant suggest a static thrust of 204000N per engine which, compared with the maximum take-off weight, leads to a thrust ratio of 0.272N/lb. From this value the same variation made for the wing loading has been imposed obtaining the results of Table 3.7.

Model data		**T/W variation range in take-off**	**T_0 from the T/W ratio**
$\mathbf{W_{TO}}$	750000 lb	0.245 N/lb	183600 N
T_0 (single engine)	204000 N	0.272 N/lb	204000 N
$\mathbf{(T/W)_{TO}}$	0.272 N/lb	0.299 N/lb	224400 N
		0.326 N/lb	244800 N

Table 3.7.: Thrust ratio variation in take-off starting from the B747-100B data.

At this point, the methods of the class `CalcTakeOff` are invoked for each of the previous scenarios allowing to determine, in each case, the value of the take-off field length and of the related FAR-25 take-off field length; the latter obtained by increasing the calculated take-off distance by 15%.

These values have then been compared with the statistical field length, calculated using the preliminary design equation (3.1) that is function of the statistical parameter named TOP25.

$$TOP_{25} = \frac{\left(\frac{W}{S}\right)_{TO}}{\sigma\, C_{L\max,TO} \left(\frac{T}{W}\right)_{TO}}; \qquad S_{TO,FL} = 37.5 \cdot TOP_{25} \tag{3.1}$$

The analysis results of Table 3.8 highlight a very little difference (less than the 5%) between the `CalcTakeOff` Java class output and the statistical data, validating this way the calculation implemented inside JPAD. Moreover, as a second check, the FAR-25 take-off field length, obtained using the wing loading and the thrust ratio of the baseline model, have been compared with the take-off field length shown upon the B747-100B flight manual; also in this case the difference between the two data is small (about the 10%) proving the good quality of the calculations made.

Take-off analysis at sea level						
W/S	**T/W**	**JPAD TO distance**	**FAR-25 TO FL**	**TOP 25**	**Statistical FL**	**Difference**
123 lb/ft^2	0.245 N/lb	2418.82 m	2781.64 m	251.28	2872.14 m	3.15%
136 lb/ft^2	0.245 N/lb	2704.07 m	3109.68 m	279.20	3191.26 m	2.56%
150 lb/ft^2	0.245 N/lb	2986.81 m	3434.83 m	307.12	3510.39 m	2.15%
164 lb/ft^2	0.245 N/lb	3278.05 m	3769.76 m	335.04	3829.52 m	1.56%
123 lb/ft^2	0.272 N/lb	2153.99 m	2477.09 m	226.15	2584.92 m	4.17%
136 lb/ft^2	**0.272 N/lb**	**2403.19 m**	**2763.67 m**	**251.28**	**2872.14 m**	**3.78%**
150 lb/ft^2	0.272 N/lb	2652.01 m	3049.81 m	276.41	3159.35 m	3.47%
164 lb/ft^2	0.272 N/lb	2906.85 m	3342.88 m	301.54	3446.57 m	3.01%
123 lb/ft^2	0.299 N/lb	1948.37 m	2240.62 m	205.59	2349.93 m	4.65%
136 lb/ft^2	0.299 N/lb	2169.48 m	2494.91 m	228.44	2611.03 m	4.45%
150 lb/ft^2	0.299 N/lb	2390.3 m	2748.87 m	251.28	2872.14 m	4.29%
164 lb/ft^2	0.299 N/lb	2618.57 m	3011.36 m	274.12	3133.24 m	3.89%
123 lb/ft^2	0.326 N/lb	1784.68 m	2052.38 m	188.46	2154.10 m	4.72%
136 lb/ft^2	0.326 N/lb	1983.98 m	2281.57 m	209.40	2393.45 m	4.67%
150 lb/ft^2	0.326 N/lb	2183.03 m	2510.48 m	230.34	2632.79 m	4.65%
164 lb/ft^2	0.326 N/lb	2387.18 m	2745.25 m	251.28	2872.14 m	4.42%

FAR-25 FL @ baseline W/S and T/W	**Flight manual**	**Difference**
2763.67 m	**3080 m**	**10.27%**

Table 3.8.: Take-off sensibility analysis results.

The same analysis has been led for the landing field length, with the following differences.

- The FAR-25 landing field length is calculated as the total landing distance divided by 0.6.
- The statistical field length is now obtained using the preliminary design equation (3.2) with the approach speed V_A in kts and statistical landing field length $S_{LND,FL}$ in ft.

$$S_{TO,FL} = 0.3 \cdot V_A^2 \quad \text{where} \quad V_A = 1.3 \cdot V_{S,LND} \tag{3.2}$$

Also in this case, a second comparison with the flight manual data has been performed considering a flap deflection of 30°.

Landing analysis at sea level					
W/S	**JPAD LND distance**	**FAR-25 LND FL**	**V_A**	**Statistical FL**	**Difference**
96 lb/ft^2	1161.35 m	1935.58 m	73.28 m/s	1855.19 m	4.33%
107 lb/ft^2	**1248.11 m**	**2080.18 m**	**77.24 m/s**	**2061.32 m**	**0.91%**
118 lb/ft^2	1334.92 m	2224.86 m	81.04 m/s	2269.22 m	1.95%

FAR-25 FL @ baseline W/S and T/W	**Flight manual**	**Difference**
2080.18 m	**1930 m**	**7.78%**

Table 3.9.: Landing sensibility analysis results.

The analysis results of Table 3.9 show an excellent accordance with the statistical data denoting a maximum difference less than the 5%. Concerning the comparison with the flight manual data, considering the case of 30° of flaps deflection, the difference is a little bit higher but still under the 10%.

Finally, the charts from Figure 3.9 to Figure 3.13 show the evolution of the main physical quantities related to the B747 take-off at different altitude. This in order to prove that the Java class is able also to compute the effect of altitude upon the take-off performance.

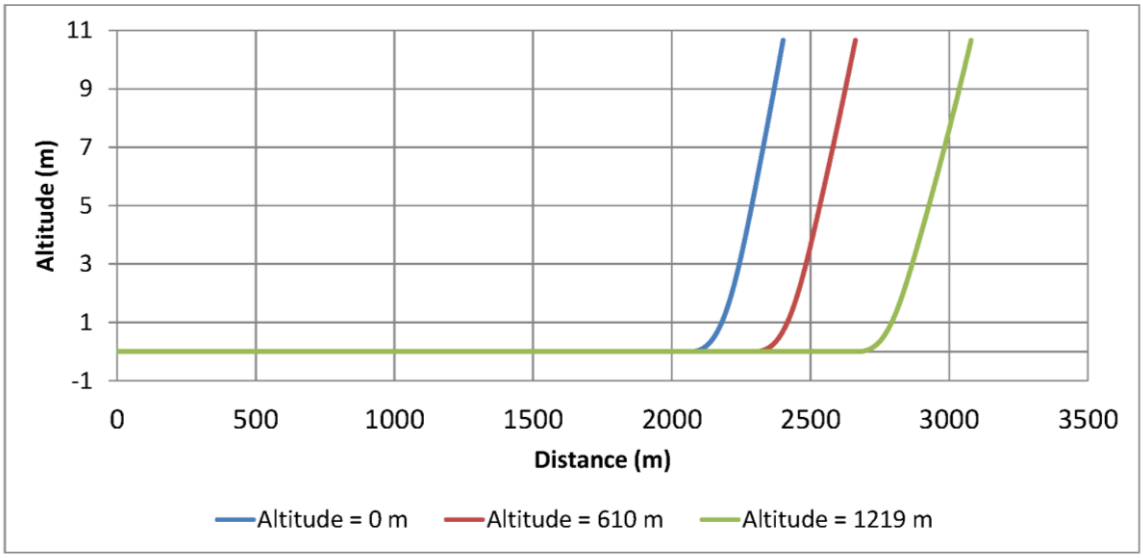

Figure 3.9: Take-off trajectory B747-100B - SL, 2000 ft ,4000 ft.

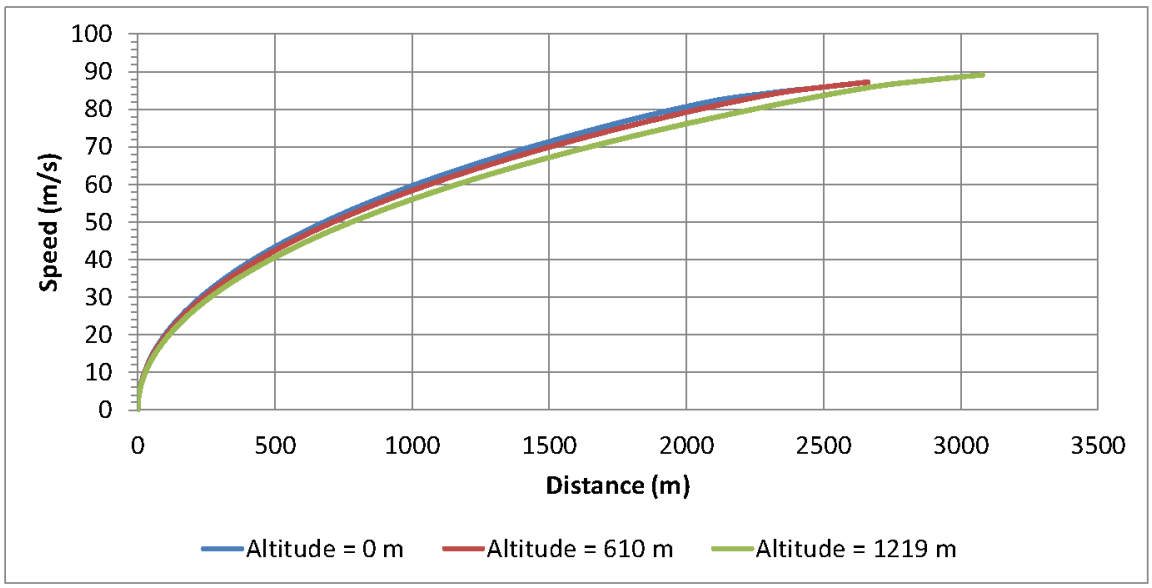

Figure 3.10: Speed evolution B747-100B - SL, 2000 ft ,4000 ft.

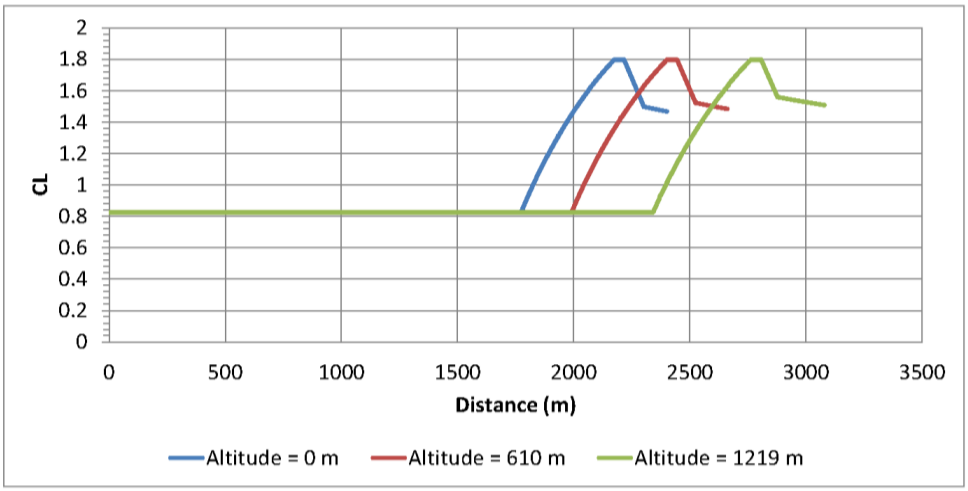

Figure 3.11: Lift coefficient evolution B747-100B - SL, 2000 ft ,4000 ft.

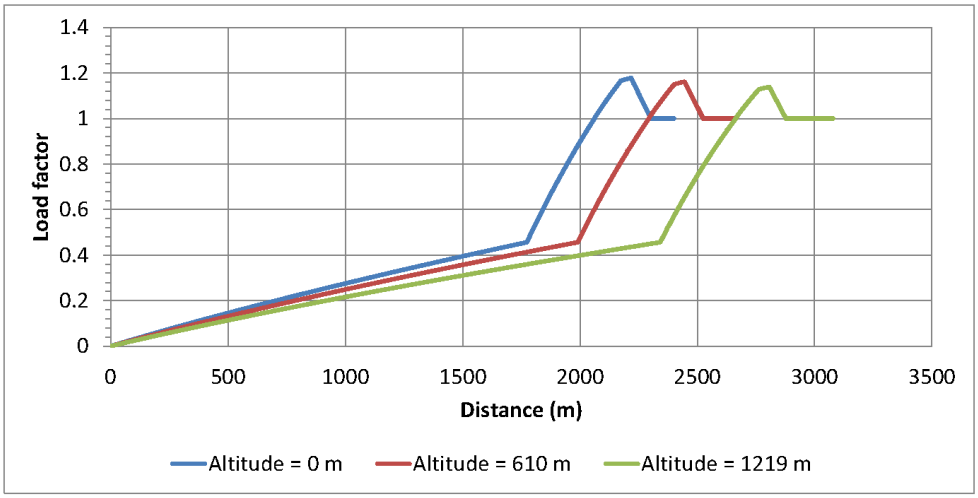

Figure 3.12: Load factor evolution B747-100B - SL, 2000 ft ,4000 ft.

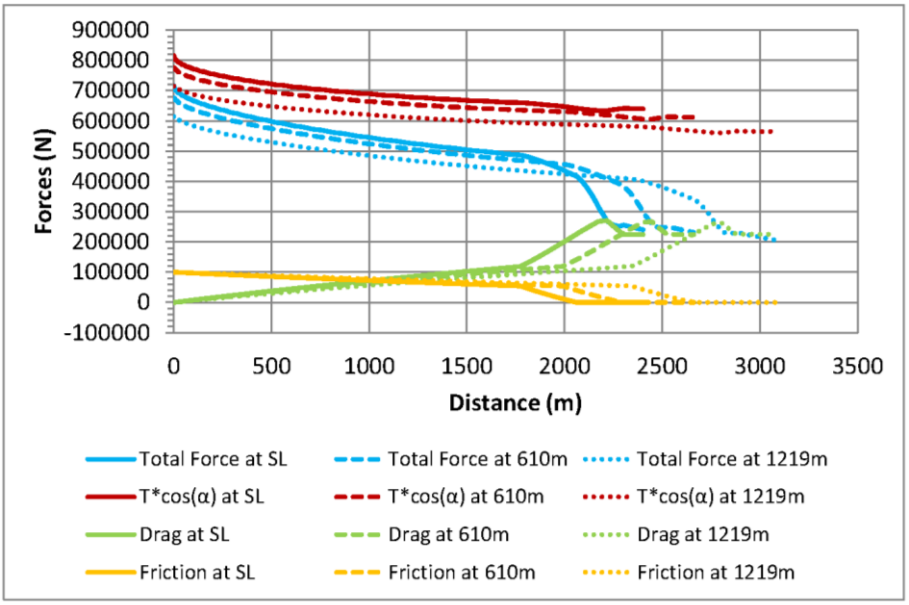

Figure 3.13: Forces evolution B747-100B - SL, 2000 ft ,4000 ft.

4 Conclusion

This work has shown the possibility to evaluate the take-off and landing performance of a parametric aircraft model with a simulation-based approach. The tool developed has proven to be very reliable and versatile as it performs the calculation of the required performance with a little computational effort and with a good accuracy, providing a difference from the statistical trend less than the 5% and a difference from the flight manual or public brochure data around the 10%.

Acknowledgment

The author wants to thank all the UniNa research group for all the support and the precious advices given during the development of this tool.

References

[1] Barnes, W., McCormick. *Aerodynamics, Aeronautics, and Flight Mechanics*. John Wiley & Sons, 1979, ISBN 0471575062.

[2] The Apache Software Foundation. *Ordinary Differential Equations Integration*. <https://commons.apache.org/proper/commons-math/userguide/ode.html>

[3] The Apache Software Foundation. *Interface FirstOrderDifferentialEquations.* <https://commons.apache.org/proper/commons-math/apidocs/org/apache/commons/math3/ode/FirstOrderDifferentialEquations.html>

[4] Oracle Corporation. *List.* < https://docs.oracle.com/javase/8/docs/api/java/util/List.html>

[5] Lausetti, A. *Decollo e atterramento di aeroplani, idrovolanti trasportati.* Levrotto & Bella Editrice s.a.s., 1992.