



HAL
open science

High-altitude free fall and parameter estimation for undergraduate numerical techniques laboratory.

Francois Lehmann

► **To cite this version:**

Francois Lehmann. High-altitude free fall and parameter estimation for undergraduate numerical techniques laboratory.. European Journal of Physics, 2020, 41 (5), pp.055803. 10.1088/1361-6404/ab9dc4 . hal-03015626

HAL Id: hal-03015626

<https://hal.science/hal-03015626>

Submitted on 19 Nov 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23

High-altitude free fall and parameter estimation for undergraduate numerical techniques laboratory.

François LEHMANN

Laboratoire d'Hydrologie et G ochimie de Strasbourg, Universit  de Strasbourg/EOST/ENGEEES, CNRS,
1 rue Blessig 67084 Strasbourg, France.

First draft April - Mai 2020,

Accepted Manuscript online 17 June 2020, DOI <https://doi.org/10.1088/1361-6404/ab9dc4>

Abstract

In this article, we present the development and treatment of an inverse problem applied to data from the Felix Baumgartner stratospheric jump. This jump is well documented with a lot of data. Therefore, it makes a particularly well-suited example for teaching in a numerical techniques laboratory. The major aim of the article is to give guidelines in order to construct a simple, but not simplistic, inverse problem with real data for junior undergraduate students. Students should master classical mechanics and have some skills in numerical modelling. We use the programming language Python and various libraries in order to build a model and solve the entire problem. This programming language is increasingly used and understood by students, which allows them to focus on the physical and numerical aspects of the involved problem. The fairly new strategy presented in this article is an attempt to estimate the angle of attack from acceleration measurements, and to give an uncertainty estimation of Baumgartner's free-fall speed.

24 I. Introduction

25 “Oh no, not the free-fall again!” Most first-ever physics courses start with classical mechanics, and
26 especially with the study of the motion of solids. On 14 October 2012, the Red Bull Stratos team [1]
27 shared with the world an extraordinary experiment, a leap from a capsule suspended 40km above
28 Earth, over Roswell, New Mexico. During this jump, the skydiver Felix Baumgartner was the first human
29 to break the sound barrier without any thrust force. He maintained for a total of 30 seconds a speed
30 greater than the speed of sound in air.

31 Even many years later, the extraordinary experimental conditions of this free fall still make it a
32 veritable ‘playground’ for any physicist who wants to teach classical mechanics and numerical
33 modelling. Like any physics problem under extreme conditions, this experiment is more complex than
34 it seems. It is a problem involving several scientific fields: mechanics, fluid dynamics, thermodynamics,
35 numerical method and many others, such as pathophysiology or data acquisition. Anyone can
36 understand it very quickly by reading the Red Bull report [2]. This is probably the reason why it makes
37 a particularly well-suited example for teaching. It allows students to make connections between
38 different scientific fields, to break the disciplinary boundaries and to learn to work as a team on a small
39 project. This example could be realized by undergraduate students after a set of laboratory classes,
40 such as that proposed by Samsonau [3]. To carry out this work in their third year, sophomore college
41 students should have followed an introductory course in Python language, with an emphasis on
42 applications in the physical sciences and engineering, on basic problem solving, programming
43 techniques, and fundamental algorithms. Students who have experience in programming with Python
44 of about 7 hours per week (1:30 lectures, 4h computer laboratory time and 1:30 personal work) for 6
45 to 7 weeks can easily tackle this problem.

46 Modelling such an experiment is difficult given the non-linearity of the processes in an environment
47 where air pressure, air temperature and air density are interrelated, and change rapidly with altitude.
48 The mechanistic approach, which consists of describing phenomena in the context of the laws of
49 conservation, is the most commonly used approach. It constitutes a powerful tool for understanding
50 and seems best suited for predictive simulations. However, we should also not forget that the key to
51 the modelling approach is to build a model that is simple, realistic and feasible. The most commonly
52 used approach for modelling such a high-altitude free fall is to use Newton's second law to consider
53 gravitational force, and an aerodynamic force that opposes the skydiver's motion through the air. This
54 model has been used by many authors [4 - 8] in order to estimate the position and velocity of the
55 skydiver. An excellent introduction for students to this kind of problem can be found in the book
56 written by Barger and Olsson [9]. This paper can be seen as a continuation of the earlier works of Colino
57 and Barbero [6] and Guerster and Walter [7].

58 In order to build the model, we need to completely define the forces included in the equation of
 59 motion, consisting of all the parametric functions and the initial conditions. Most of the time, not all
 60 the parameters are well known. Sometimes, we don't know the exact value of a specific parameter.
 61 Some of these parameters cannot be directly measured, so in such conditions we can use parameter
 62 estimation techniques in order to estimate the parameter values and also, importantly, the parameter
 63 uncertainty and ultimately, calculating the associated uncertainty of the computed position and
 64 velocity of the skydiver, whilst taking into account the available measures.

65 This paper follows a three-fold structure: first, building the direct problem in which the physical system
 66 of the free fall is modeled, with all parameters and with the initial conditions. Second, the inverse
 67 problem is merged with the direct problem and the data. Finally, the last section illustrates the global
 68 model application to determine an estimation of the parameters and their associated uncertainties.

69

70 II. The direct problem

71 In order to establish the equation of motion, Newton's second law is applied to a body of mass m
 72 subject to two forces, gravitational force \vec{P} and drag force \vec{F}_D :

$$73 \quad m\vec{a} = \vec{P} + \vec{F}_D \quad (1)$$

74 \vec{a} is the body acceleration in m s^{-2} . If the free fall is strictly vertical and the positive z direction is chosen
 75 to be up, the equation of motion, as described in detail by Guerster and Walter [7], is given by:

$$76 \quad m \frac{dv_z}{dt} = -mg(z) + \frac{1}{2} C_D(Ma) A_{\perp}(\beta) \rho(z, T) v_z^2 \quad (2)$$

77 where $C_D(Ma)$ is the drag coefficient, which depends on the Mach number Ma , A_{\perp} is the projected
 78 area depending on the angle of attack β (the angle between the z axis reversed and a longitudinal
 79 reference line on the body, head first), $\rho(z, T)$ the air density depending on altitude z and air
 80 temperature T , $g(z)$ the acceleration due to gravity depending on z . For this equation of motion, the
 81 state variables are the altitude z and the speed v_z . The aim of the direct problem is to compute the
 82 state variables, which are dependent variables, against time (the independent variable). This can be
 83 done by rewriting the second-order differential equation (2) as a system of two first-order equations:

$$84 \quad \begin{cases} \frac{dz}{dt} = v_z \\ \frac{dv_z}{dt} = -g(z) + \frac{1}{2m} C_D(Ma) A_{\perp}(\beta) \rho(z, T) v_z^2 \end{cases} \quad (3)$$

85 To determine the evolution of the body, it is necessary to define its initial state, that is the initial
 86 position and the initial velocity. At this step, it is also necessary to define all the parametric functions
 87 which complete the model. The same relations as those used by Guerster and Walter [7] are included
 88 within the model.

89

90

91 *The parametric functions and the data*

92 Before defining explicitly all the parametric functions, it is important to fix the geographical coordinate
93 system in which the trajectory is studied. The initial GPS position (φ :latitude, λ :longitude, z :altitude),
94 when Baumgartner left the capsule on 14 October 2012, at 18:06:32.0 UTC, was ($\varphi = 33.3408417^\circ$,
95 $\lambda = -103.7679067^\circ$, $z = 38969.4$ m). To study the trajectory and the speed of Baumgartner during
96 his fall, we choose a local tangent plane coordinates system with an origin given by ($\varphi =$
97 33.3408417° , $\lambda = -103.7679067^\circ$, $z = 0.0$ m); by convention the east axis is labeled x_{East} , the
98 north y_{North} and the up z_{Up} . In this local ENU (East, North, Up) coordinate system, the trajectory
99 versus time is depicted in figure 1. This representation has not been used by other authors who studied
100 the free fall, but it is very interesting, more intuitive and practical. In figure (1), we can clearly identify
101 two stages of the leap's evolution. The first stage, from exit time $t=0$ to 75 s, is a quasi-perfect vertical
102 one dimensional movement. The second stage from $t=75$ to $t=260$ s (main chute deployment, at
103 18:10:52.0 UTC) looks more like a classic free fall of a skydiver who does not try to increase his speed
104 but instead tries to fall slowly and steadily in a nice regular "belly-to-earth" position.
105 In most papers [4,5 and 8], the acceleration due to gravity is taken as constant, but the change of g can
106 be easily considered by using the world geodetic system ellipsoidal gravity formula (WGS 1984) with
107 the first-order free-air correction factor:

108
$$g(z, \varphi) = 9.7803 \frac{1+0.00193 \cdot \sin^2(\varphi)}{\sqrt{1-0.00669 \cdot \sin^2(\varphi)}} - 3.086 \times 10^{-6} \cdot z \quad (4)$$

109 at the origin of local ENU system $\varphi = 33.3408417^\circ$

110
$$g(z, \varphi) = 9.7959 - 3.086 \times 10^{-6} \cdot z \quad (5)$$

111 For z varying from the initial position to the main chute deployment, $2566.8 \leq z \leq 38969.4$ m, the
112 acceleration due to gravity is $9.6756 \leq g \leq 9.7879$ m/s^2 , that is a relative error of 1% if g is taken
113 as a constant. The total mass of Baumgartner taken for the simulations is $m = 121.2$ kg [7].

114 For this experience, the Red Bull Stratos team did everything to ensure the safety of Baumgartner's
115 jump [1; 10]. Various measures have been taken, analyzed and processed by the team to validate
116 several world records, and now they are available to us. Baumgartner was equipped with several GPS
117 apparatuses, and in particular a Garmin 18X-5 WAAS GPS with a sampling frequency of 5 position
118 measurements per second. The data files made available also contain temperature, air pressure, speed
119 of sound and triaxial acceleration data. Baumgartner was equipped with a triaxial accelerometer
120 positioned at chest level in order to understand and analyze the environmental stressors experienced
121 [11].

122 Atmospheric density and speed of sound ($Mach_1$) profiles are estimated based on the pressure $P(z)$
123 and temperature $T(z)$ measurements with:

124
$$\rho(z) = \frac{P(z)}{R_s T(z)} \text{ and } Mach_1(z) = \sqrt{1.40 \cdot R_s T(z)} \quad (6 \text{ and } 7)$$

125 with $R_s = 287.0 \text{ J}/(\text{K kg})$. The profiles of temperature, pressure, density and speed of sound are
 126 depicted in figure 2 (red curve) and are compared to two classical models, the US Standard
 127 atmosphere, 1976 [12] and NRLMSISE-00 Atmosphere Model [13, 14], and to the sounding operated
 128 by the Red Bull Stratos team at the Santa Teresa observatory and Roswell airport. The most important
 129 data for studying the free fall is for $20 \leq z \leq 40 \text{ km}$. In this part, we have good correlation between
 130 the data and the NRLMSISE-00 Atmosphere Model. In the later simulations we will use the data
 131 depicted on the red curve. To our knowledge, the data was obtained by the Red Bull Stratos team with
 132 a numerical weather forecast model.

133 For the drag coefficient and the projected area, we use the relations proposed by Guerster and
 134 Walter [7]:

$$135 \quad C_D = A \quad \text{for } Ma = \frac{v_z}{Mach_1} \leq 0.6 \quad (8)$$

$$136 \quad C_D = A + B(Ma - 0.6)^2 \quad \text{for } 0.6 \leq Ma \leq 1.1 \quad (9)$$

$$137 \quad C_D = A + 0.25 \cdot B - C(Ma - 1.1) \quad \text{for } 1.1 \leq Ma \quad (10)$$

138 and

$$139 \quad A_{\perp}(\beta) = A_x \sin(\beta) + A_z \cos(\beta) \quad (11)$$

140 with $A_x = 1.19 \text{ m}^2$ the effective front area and $A_z = 0.525 \text{ m}^2$ the effective top area [7], A, B and C
 141 three parameters that should be estimated in the inverse problem.

142 The last needed information, and probably the most difficult to obtain is the angle of attack β during
 143 the leap. Guerster and Walter [7], estimate an angle of attack $\alpha = \frac{\pi}{2} - \beta$ with the full video from
 144 Baumgartner's point of view during the leap. In this study, the angle of attack is estimated based on
 145 the triaxial acceleration measurements (fig.3). Accelerometers are sensitive to both linear acceleration
 146 and the local gravitational field. In the absence of linear acceleration, the accelerometer output is a
 147 measurement of the rotated gravitational field vector and can be used to determine the accelerometer
 148 orientation angles. At the start of a fall, the vector sum of acceleration will tend toward 0 g (fig 3) -
 149 this is called the phenomenon of weightlessness. Until $t=25 \text{ s}$, the longitudinal and lateral accelerations
 150 do not change, after that time and until $t=75 \text{ s}$, these accelerations fluctuate a lot. During this time,
 151 Baumgartner was in a dangerous situation of spinning, which has been studied in detail by Garbino et
 152 al. [11]. The vertical acceleration has a different signal: it changes gradually, probably due to the drag
 153 force's module and direction acting on the body (fig. 3). Even if we aren't in a perfectly static situation,
 154 and we also neglect the lateral acceleration A_y , and if we suppose likewise that Baumgartner didn't
 155 rotate around his body's z-axis (head first), the angle of attack is estimated with [15, eq. 55 page 19]:

$$156 \quad \beta = \cos^{-1} \left(\frac{A_z}{\sqrt{A_x^2 + A_z^2}} \right) \quad (12)$$

157 with A_z and A_x representing the vertical and longitudinal acceleration respectively. In order to smooth
158 the signal and remove potential linear acceleration, a rolling mean of 2.5 s is applied to the angle and
159 used later in a linear interpolation lookup table for the simulations (fig. 4).

160 *The numerical implementation*

161 The programming language we use is Python, a language which gives readable code that closely
162 resembles the equations. The equation (3) is a system of two first-order equations that can be solved
163 by numerical integration. There are a lot of numerical tools called “ODE solver” (ordinary differential
164 equation) to solve this kind of equation [16]. Here, we have chosen to work with the SciPy ecosystem
165 [17], and in particular with the module `scipy.integrate.solve_ivp`. This module allows us to choose
166 between all the classical numerical integration methods, and trigger events to stop the integration
167 time if needed. By default, we select the Runge-Kutta method. We can also set the maximum allowed
168 time step size during integration, which allows us to properly control any variations in the parametric
169 functions, like the angle of attack. At this stage, we have all the necessary information allowing us to
170 program and to solve the direct problem, apart from the parameters A, B and C appearing in the drag
171 force. A useful solution is to generate uniformly distributed random parameters between bounds
172 and/or to take some values from literature and to build the code and do tests. This is an important
173 step before tackling the inverse problem, since the direct code must be solved in as little CPU time as
174 possible, and be reliable and robust regardless of the parameters used.

175

176 **III. The inverse problem**

177 The inverse problem covers a very wide range of formulations and fields of application. In this part, we
178 will present an application of a classical non-linear inverse problem. Here, the aim is to find the model
179 parameters we don't know that produce the data we have measured. From a mathematical point of
180 view, we formulate this with the help of a classical cost function or objective function, i.e. the mean
181 sum square error between model output and the measures:

$$182 \quad O(p; y, y_{mes}) = \frac{1}{nm-np} \sum_{i=1}^{nm} (y_i(p) - y_i^*)^2 \quad (13)$$

183 with $p = (A, B, C)$ the vector of parameters, nm and np the number of measurements and parameters
184 respectively, $y_i(p)$ the model output, e.g. the speed and y_i^* the measurements. The aim is to find the
185 parameters that minimize the objective function, which is done iteratively. Generally, the algorithm
186 starts with a set of initial parameters and calculates the corrections to be made to the parameters, so
187 that the objective function decreases until a criterion of the objective function is reached and/or until
188 the parameter values don't change anymore. There are plenty of algorithms that can be used to
189 minimize a function. The most widely used is probably the Levenberg-Marquardt algorithm. The choice
190 of a good algorithm is problem dependent, on the number of parameters, the bounds on the

191 parameters, the number of available measurements, the correlations between parameters, the
 192 structure of the model and so forth. We can distinguish two main classes of algorithms: global and
 193 local algorithms. According to the author's own experience, the most efficient strategy when solving a
 194 minimization problem is to be able to use several kind of algorithms and analyze the different results.
 195 Here, we have chosen to use LMFIT: Non-Linear Least-Squares Minimization and Curve-Fitting for
 196 Python [18]. This library allows us to use more than twenty methods of minimization, from the local
 197 Levenberg-Marquardt least squares method to the Markov Chain Monte Carlo (MCMC) algorithm,
 198 amongst others. A complete discussion of inverse methods is beyond the scope of this document; the
 199 aim here is simply to give students access to up-to-date methods and algorithms as "gray-box", and
 200 not completely "black-box".

201 The use of `lmfit.minimize()` module is straightforward, as long as the user starts with one of the
 202 numerous examples available on the GitHub repository ([https://lmfit.github.io/lmfit-
 203 py/examples/index.html](https://lmfit.github.io/lmfit-py/examples/index.html)) and adapts it to their problem. Principally, the user has to build a callable
 204 function that corresponds to the direct problem. This function should return the difference between
 205 model and measure, i.e. the residual vector:

$$\varepsilon_i = y_i(p) - y_i^* \quad (14)$$

207 In our example, the state variable of interest is the speed. $y_i(p)$ is the solution of the system of
 208 equation (3), i.e. $v_z(t = t_i)$, with t_i the discrete time observations for $i = 1 \dots nm$. In the local ENU
 209 coordinate system, we can compute the speed by using the evolution of the altitude z over time with
 210 a first order finite difference equation:

$$y_i^* \equiv v_i \simeq \frac{z_{i+1}(t_{i+1}) - z_i(t_i)}{t_{i+1} - t_i} \quad (15)$$

212 The GPS used has a sampling frequency of 5 position measurements per second; for the whole duration
 213 of the leap (from $t=0$ to 260s), this gives us a total number of available measurements of 1300. In figure
 214 (1), we have seen that the vertical free fall is well approximated during only the first part of the leap,
 215 that is for a period from $t=0$ to 75 s. For the parameter estimation, we only use the data obtained
 216 during this first part of the leap and with one measurement per second, that is a total number of
 217 measurements $nm = 75$, for a total number of parameters to be estimated $np = 3$.

218

219 **IV. Results and discussion**

220 In order to start the inverse problem, it is often necessary to have some prior knowledge of the
 221 parameters we want to estimate. Sometimes, these parameters have no physical meaning, but even
 222 in this case we should give an initial set of parameters and some lower and upper bound values (Table
 223 1). For this simulation, we use the trust region reflective method to minimize the objective function

224 [18]. At the end of the estimation, after 11 iterations, the final set of parameters and the 95%
225 confidence interval on the parameters are:

$$226 \quad A = 0.617 \pm 0.035 \text{ (5.7\%)} \quad (16)$$

$$227 \quad B = 0.840 \pm 0.259 \text{ (30.8\%)} \quad (17)$$

$$228 \quad C = 1.425 \pm 0.344 \text{ (24.2\%)} \quad (18)$$

229 The results differ from those of Guerster and Walter [7] for the parameters B and C. We find slightly
230 larger values for almost identical confidence intervals. It should be noted that the parameters A and B
231 have a correlation of -0.92 and for B and C of 0.89. Since we didn't use exactly the same angle of attack
232 as Guerster and Walter [7], it is possible to find different parameters that give almost identical results
233 for the speed. This is a key point for parameter estimation, as there is not always a unique solution to
234 the inverse problem, but instead depends greatly on the model structure and on available data. The
235 drag force in the model depends on the product of the drag coefficient and the projected area, so this
236 creates a perfect correlation between them. Colino and Barbero's [6] strategy was to use the product
237 and take different values of this product depending on the stages of the leap. The results of the inverse
238 problem for the altitude, speed and Mach number are depicted in figure (5). We can observe a fairly
239 good match between the measurements and the modelling. The gray region (fig. 5) represents the 95%
240 confidence interval of the estimated speed. In the second stage, from $t=75$ to $t=260$ s, we observe
241 larger differences between the measurements and the model, probably due to a 3D effect on the
242 trajectory. A more detailed graphic concerning the supersonic stage is given in figure (6). We found a
243 total supersonic time of 30s, a maximum speed, reached at $t=50$ s, $v = 375.7 \pm 10.8$ m/s and a
244 Mach number $Ma = 1.25 \pm 0.03$. We can also see the impact of variations over time of the angle of
245 attack on the speed, for $t=40$ s and $t=74$ s.

246 Now that the whole model has been built (direct and inverse), it is relatively simple to carry out
247 additional tests if needed, for example to test other sets of initial parameters, or to assume that g is
248 constant, or to propose a different model for the drag coefficient or for the air density equation, or to
249 modify the time parametric function for the angle of attack.

250 If one wishes to have more reliable information on the parameters and their uncertainties, it is possible
251 with LMFIT to use an MCMC algorithm to determine the posterior distributions for the parameters,
252 and not only a local uncertainty estimation.

253

254 **V. Conclusion**

255 The good thing about building a full model, i.e. direct and inverse, is that it gives students the
256 opportunity to learn multiple skills by discussing the problem on the conceptual level, together with
257 mathematical formulation, finding numerical methods, appropriate algorithms and writing the
258 computer code, and testing the programs and using them to perform numerical experiments, to

259 analyze the problem, and to compare the results to real data. All of this is now possible, especially with
260 Python if we agree to use some libraries as a "gray-box" and explain the principles of these algorithms
261 to students. Additionally, representing the computational results graphically is always very important,
262 and this is facilitated for students with Python [19]. Finally, teaching the building of an inverse problem
263 has a lot of similarities with creating an Investigative Science Learning Environment [20, figure 1.3 page
264 1-7].

265

266 **Acknowledgments**

267 A special thanks to Jonathan Clark and Alex Garbino, Department of Medicine, Section of Emergency
268 Medicine and Center for Space Medicine, Baylor College of Medicine, Houston - TX, for support and
269 making the flight data available to us. A sincere thank you to our colleague, Rob Simmons, from Institut
270 Universitaire de Technologie Louis Pasteur, for his diligent proofreading of the article.

271

272 **References:**

273 [1] Red Bull Stratos www.redbullstratos.com.

274

275 [2] Red Bull Stratos: Full Scientific Data Report, Findings of the Red Bull Stratos Scientific Summit,
276 California Science Center, Los Angeles, California, USA January 23, 2013,
277 https://issuu.com/redbullstratos/docs/red_bull_stratos_summit_report_final_050213.

278

279 [3] Samsonau S. V., 2018. Computer simulations combined with experiments for a calculus-based
280 physics laboratory course, Phys. Educ. 53 055013.

281

282 [4] Benacka J., 2010. High-altitude free fall revised. American Journal of Physics 78(6)616-619; doi:
283 10.1119/1.3298375.

284

285 [5] Theilmann F. and Apolin M., 2013. Supersonic freefall—a modern adventure as a topic for the
286 physics class. Phys. Educ. 48(2)150-158.

287

288 [6] Colino J. M. and Barbero A. J., 2013. Quantitative model of record stratospheric freefall. Eur. J. Phys.
289 (34)841–848, doi:10.1088/0143-0807/34/4/841.

290

291 [7] Guerster M. and Walter U., 2017. Aerodynamics of a highly irregular body at transonic speeds—
292 Analysis of STRATOS flight data. PLoS ONE 12(12): e0187798; doi: 10.1371/journal.pone.0187798.

293

294 [8] Corvo T., 2019. An Analytical Solution to the Extreme Skydiver Problem. The Physics Teacher
295 57:287-289; doi: 10.1119/1.5098912.
296

297 [9] Barger V. and Olsson M., 1995. Classical Mechanics: A Modern Perspective, Second Edition,
298 McGraw-Hill, New York, 418pp.
299

300 [10] Blue R.S., Law J., Norton S.C., Garbino A., Pattarini J. M., Turney M. W., Clark J.B., 2013. Overview
301 of medical operations for a manned stratospheric balloon flight. Aviat Space Environ Med (84)237-41.
302

303 [11] Garbino A., Blue R.S., Pattarini J. M., Law J., Clark J.B., 2014. Physiological monitoring and analysis
304 of a manned stratospheric balloon test program. Aviat Space Environ Med (85)177–82.
305

306 [12] U.S. standard atmosphere, 1976, Washington, D.C., NOAA--S/T 76-1562,241pp.
307

308 [13] J.M. Picone, A.E. Hedin, D.P. Drob, and A.C. Aikin, 2002. NRLMSISE-00 empirical model of the
309 atmosphere: Statistical comparisons and scientific issues, J. Geophys. Res., 107(A12), 1468,
310 doi:10.1029/2002JA009430.
311

312 [14] Papitashvili N. 2001. Web interface for NRLMSISE-00 Atmosphere Model, The Community
313 Coordinated Modeling Center, <https://ccmc.gsfc.nasa.gov/modelweb/models/nrlmsise00.php>.
314

315 [15] Pedley M., 2013. Tilt Sensing Using a Three-Axis Accelerometer. Freescale Semiconductor, Inc.
316 AN3461 Application Note Rev. 6, 22pp.
317

318 [16] Linge S. and Langtangen H. S., 2020. Programming for Computations – Python : A Gentle
319 Introduction to Numerical Simulations with Python 3.6, Second Edition, Springer Open, 350pp.
320 <https://doi.org/10.1007/978-3-030-16877-3>.
321

322 [17] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau,
323 Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew
324 Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern,
325 Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef
326 Perktold, Robert Cimrman, Ian Henriksen, E.A. Quintero, Charles R Harris, Anne M. Archibald, Antônio
327 H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. (2020) SciPy 1.0:

328 Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*,
329 <https://doi.org/10.1038/s41592-019-0686-2>.

330

331 [18] Newville M., Stensitzki T., Allen D. B. and Ingargiola A., 2014. LMFIT: Non-Linear Least-Square
332 Minimization and Curve-Fitting for Python (Version 1.0.1). Zenodo.
333 <http://doi.org/10.5281/zenodo.11813>.

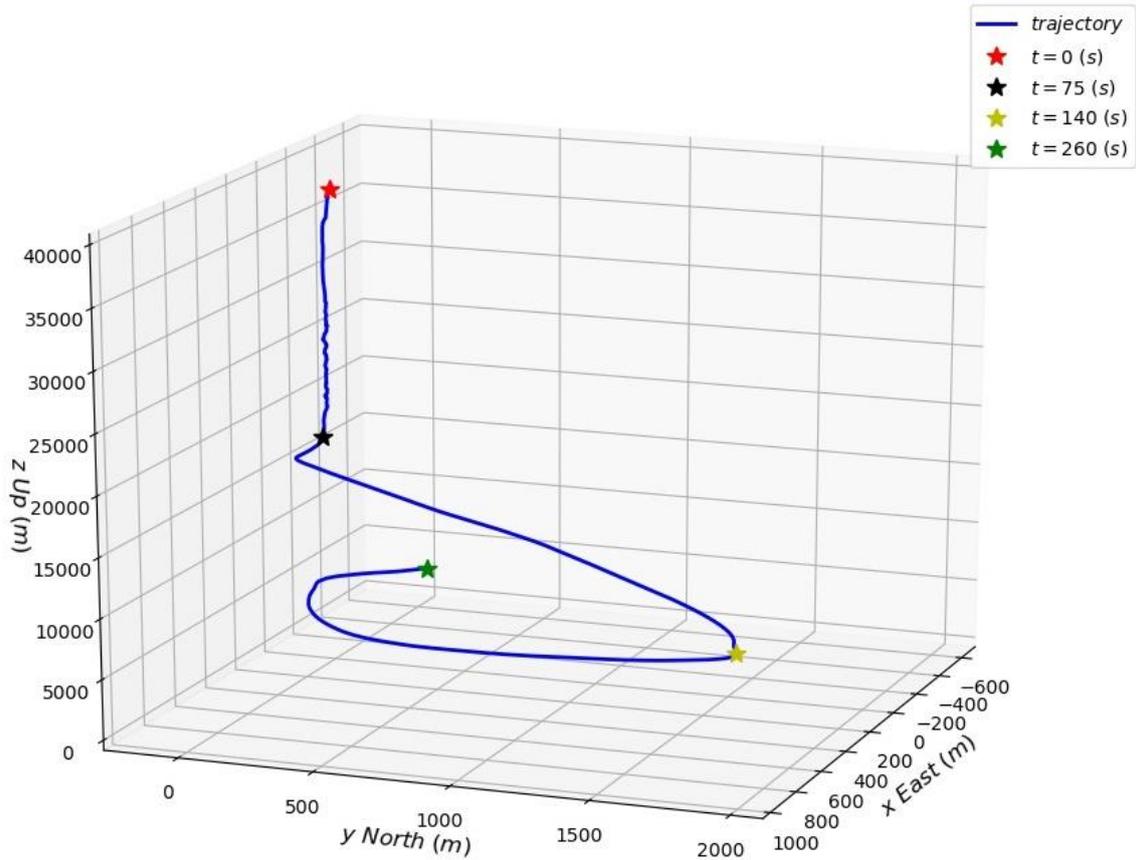
334

335 [19] Weber J. and Wilhelm Th., 2020. The benefit of computational modelling in physics teaching: a
336 historical overview, *Eur. J. Phys.*41 034003.

337

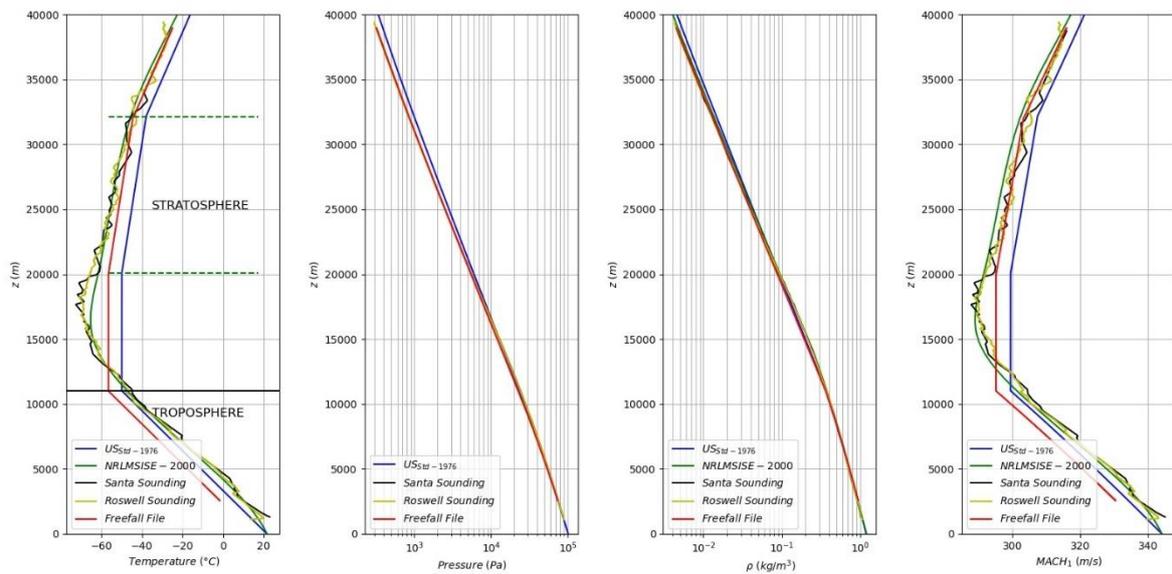
338 [20] Etkina E., Brookes, D. T. and Planinsic G., 2019. Investigative Science Learning Environment. When
339 learning physics mirrors doing physics. Morgan & Claypool Publishers, 138pp.
340 <http://dx.doi.org/10.1088/2053-2571/ab3ebd>.

341



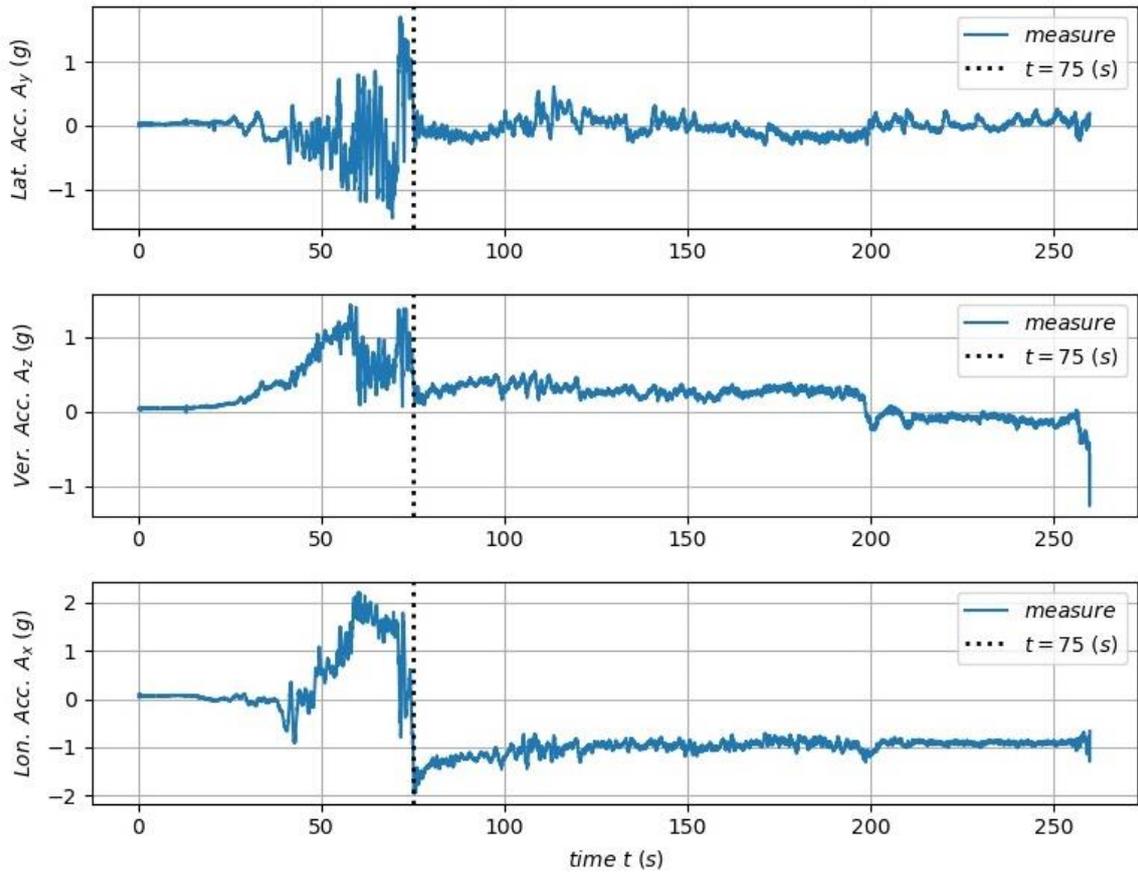
342
343
344
345

Figure 1: Skydiver position evolution through the leap, expressed in a local tangent plane coordinates system, as origin ($\varphi = 33.3408417^\circ$, $\lambda = -103.7679067^\circ$, $z = 0.0$ m)



346
347

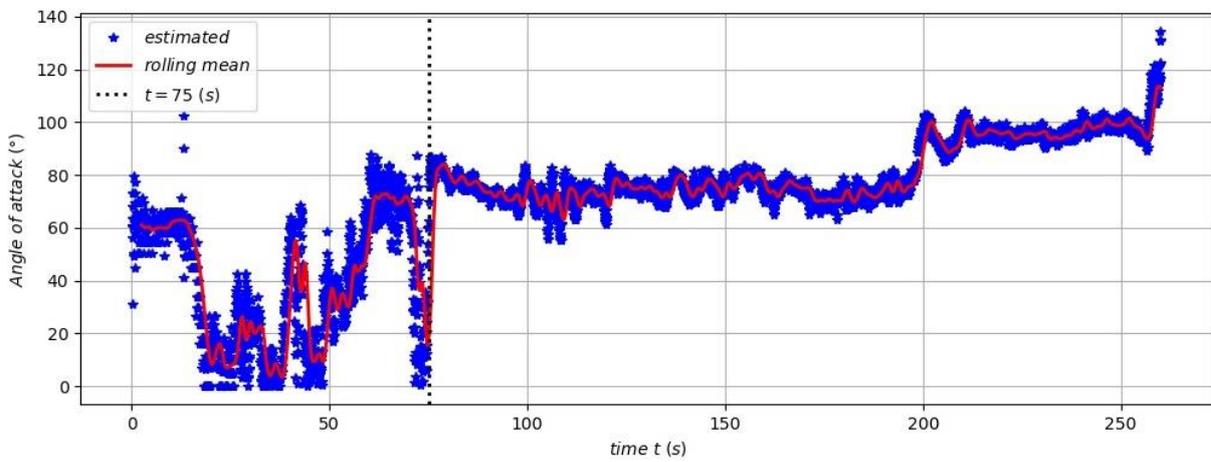
Figure 2: Comparison of the different atmospheric data; the red curves are used for the simulations.



348

349

Figure 3: Measured triaxial acceleration



350

351

Figure 4: Estimated angle of attack based on the acceleration A_x and A_z

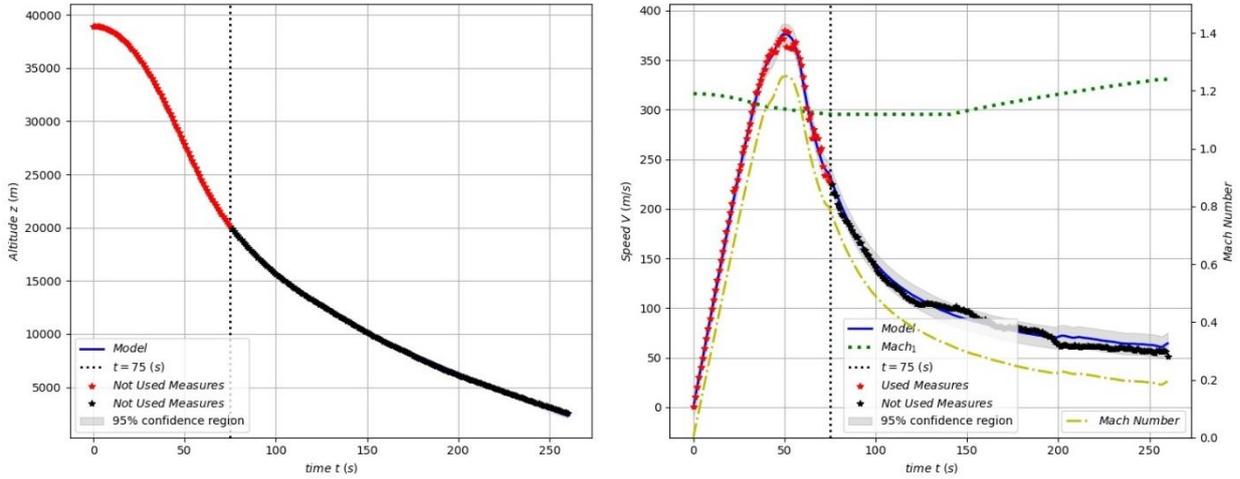
352

353 Table 1: Initial set of parameters with lower and upper bounds.

Parameters p_k	Min value	Initial value	Max value
A	0.25	0.6	2
B	0.1	0.6	2
C	0	0.3	5

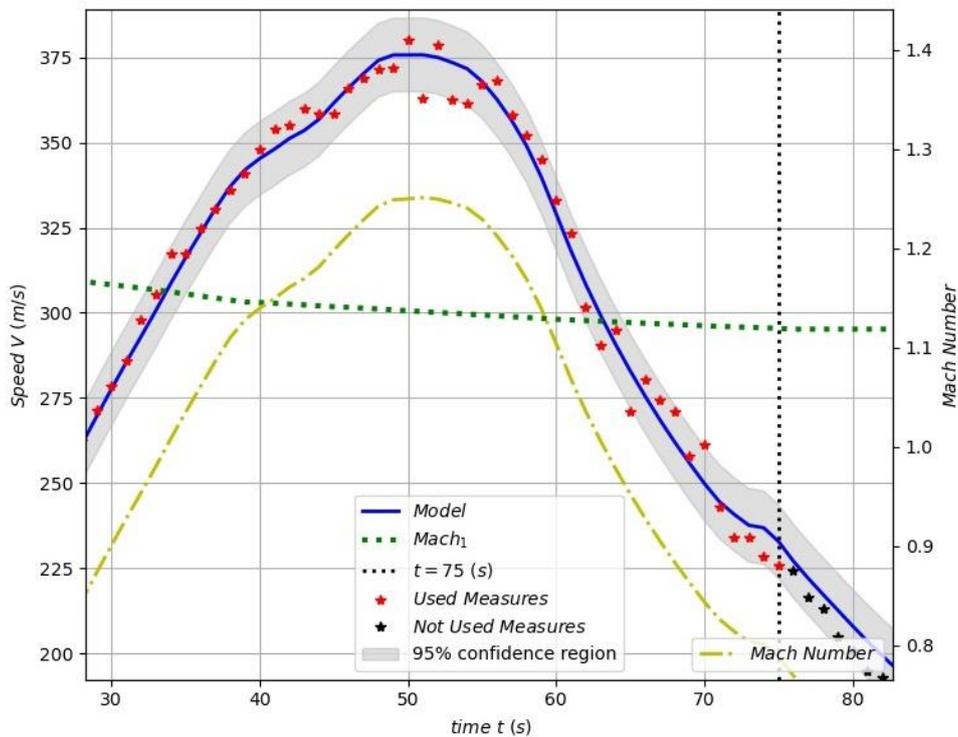
354

355



356

357 Figure 5: Measured and predicted altitude, speed and Mach number with the final set of parameters.



358

359 Figure 6: Measured and predicted altitude, speed and Mach number with the final set of parameters
 360 for the supersonic stage.