

Article

Inverse Procedure to Initial Parameter Estimation for Air-Dropped Packages Using Neural Networks

Beata Potrzeszcz-Sut ^{1,*}  and Marta Grzyb ²

¹ Faculty of Civil Engineering and Architecture, Department of Structural Theory and BIM, Kielce University of Technology, al. Tysiąclecia Państwa Polskiego 7, 25-314 Kielce, Poland

² Faculty of Mechatronics and Mechanical Engineering, Department of Computer and Armament Technologies, Kielce University of Technology, al. Tysiąclecia Państwa Polskiego 7, 25-314 Kielce, Poland; mgrzyb@tu.kielce.pl

* Correspondence: beatap@tu.kielce.pl; Tel.: +48-41-34-24-801

Abstract

This paper presents a neural network-driven framework for solving the inverse problem of initial parameter estimation in air-dropped package missions. Unlike traditional analytical methods, which are computationally intensive and often impractical in real time, the proposed system leverages the flexibility of multilayer perceptrons to model both forward and inverse relationships between drop conditions and flight outcomes. In the forward stage, a trained network predicts range, flight time, and impact velocity from predefined release parameters. In the inverse stage, a deeper neural model reconstructs the required release velocity, angle, and altitude directly from the desired operational outcomes. By employing a hybrid workflow—combining physics-based simulation with neural approximation—our approach generates large, high-quality datasets at low computational cost. Results demonstrate that the inverse network achieves high accuracy across deterministic and stochastic tests, with minimal error when operating within the training domain. The study confirms the suitability of neural architectures for tackling complex, nonlinear identification tasks in precision airdrop operations. Beyond their technical efficiency, such models enable agile, GPS-independent mission planning, offering a reliable and low-cost decision support tool for humanitarian aid, scientific research, and defense logistics. This work highlights how artificial intelligence can transform conventional trajectory design into a fast, adaptive, and autonomous capability.

Keywords: air-dropped packages; relief supplies; neural network; forward mode; inverse problem; hybrid computation



Academic Editor: Wei Huang

Received: 29 August 2025

Revised: 23 September 2025

Accepted: 24 September 2025

Published: 25 September 2025

Citation: Potrzeszcz-Sut, B.; Grzyb, M. Inverse Procedure to Initial Parameter Estimation for Air-Dropped Packages Using Neural Networks. *Appl. Sci.* **2025**, *15*, 10422. <https://doi.org/10.3390/app151910422>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

1.1. Introduction to Cargo Airdrop Processes

The process of airdrops, regardless of their purpose (humanitarian aid, rescue materials, scientific research), is complex and requires strict coordination of multiple services, appropriate equipment, and adherence to required regulations. The success of such missions is influenced by many interdependent factors.

The most important of these include:

- Accuracy defining the drop zone, dependent on data from navigation systems (GPS, GLONASS, INS),

- Maneuverability, mainly dependent on the aerodynamics of the capsule, but also on the structural strength (acting g-forces),
- Achievable range, which allows for dropping the cargo from a safe distance, thereby minimizing potential losses while maintaining safety conditions,
- Achievable object speed, which in turn minimizes the time required to deliver the airdropped cargo,
- Flight profile, which allows for low-altitude flight,
- Weather conditions, mainly those whose effects cannot be predicted (atmospheric turbulence, wind shear, and gusts),
- All kinds of human factors, such as pilot or operator skills, or their availability.

Airdrops are mainly carried out using airplanes and unmanned aerial vehicles (UAV). The primary goal is to drop humanitarian aid (primarily medicines, specialized equipment, or other sensitive cargo) by quickly delivering essential resources to areas cut off from land communication routes, affected by natural disasters, armed conflict, or other crisis situations. Another specific type of mission involves the airdrop of resources such as underwater gliders, which are widely used in ocean observations [1]. In such cases, the proper water entry of the glider is crucial to prevent its destruction. For each type of these operations, methods based on airdrop dynamics modeling and control optimization are known for both cargo loading flight operations and airdrop flight operations [2,3]. Safety and risk assessment of airdrop missions are also widely studied, with approaches based on STPA-BN analysis applied to remote operations and heavy equipment transport [4,5].

1.2. Types of Airdrop Systems

Bearing in mind the methods of cargo airdrops, they can be divided into several categories:

- Non-precision cargo airdrop systems
 - They are characterized by low accuracy, meaning the cargo does not always land at the intended point,
 - Most commonly used during good visibility,
 - These systems require a low altitude and typically involve flying over the drop zone, which is not always possible and carries a high risk of mission failure, as well as danger to the pilot and aircraft.
- Precision cargo airdrop systems
 - Work on guided airdrop systems began in the early 1960s, utilizing a modified parabolic parachute [6],
 - Equipped with Autonomous Guidance Units (AGU), whose elements include: a computer for calculating flight trajectory, communication devices with antennas, a GPS receiver, temperature and pressure sensors, LIDAR radar, devices controlling steering lines, and an operating panel,
 - They use appropriate devices that detect the wind profile and speed,
 - They allow for airdropping cargo from altitudes of over 9000 m with a drop accuracy of 25 to 150 m [7],
 - They utilize advanced software (Launch Acceptability Region, LAR) that calculates the area from which a drop can be made to ensure the cargo hits the target,
 - Joint Precision Airdrop System, which is designed for conducting precise airdrops from high altitudes and comes in a wide range of versions depending on cargo weight (from 90 kg to 4500 kg). Equipped with a wing-type gliding parachute, it has the ability to fly in any direction regardless of wind and to change flight direction at any moment [8].

- Guided parachutes/parafoils
 - They are equipped with Autonomous Guidance Units (AGU), that allow for a change in flight trajectory, including: adjusting the course mid-flight, avoiding obstacles, and precise maneuvering to reach the designated drop point,
 - They have the ability to be dropped from higher altitudes and greater distances from the drop point,
 - Ram-air parachutes (wing-type) are characterized by their maneuverability and ability to fly in any direction,
 - Round parachutes (modified) are less maneuverable than ram-air, but have the advantage of being cheaper to produce; they are used in systems like AGAS, where pneumatic muscle actuators are used for steering,
 - The parachute's smart guidance System Joint Precision Airdrop System (JPADS) autonomously calculates the correct drop point. To do this, it uses data from global positioning, weather models, and advanced mathematical operations, allowing it to reach the target on its own based on the received coordinates.

1.3. Safety in Airdrop Operations

Extremely important aspect concerning the cargo airdrop process is the issue of safety. Its assessment process is complex, mainly due to the strong interconnections between human, machine, and environment, but also due to the high demands placed on such missions. One example of a method used for safety assessment is the improved System-Theoretic Process Analysis-Bayesian Network (STPA-BN) method. This is a system theory-based safety analysis method. Its purpose is to understand how a lack of control over the system can lead to unsafe situations. The STPA-BN method enables real-time decision support for pilots by estimating the probability of losses in case of a single fault or failure. The authors of article [4] propose the application of a novel risk analysis method that integrates STPA and Bayesian Networks (BN) along with elements such as Noisy-OR gates, the Parent-Divorcing Technique (PDT), and sub-modeling for remote piloting operations. For quantitative safety assessment methods, Bayesian Networks themselves are used. They model cause-and-effect relationships between events and serve to calculate probabilities.

The article [5] takes the topic of airdropping heavy equipment (over 1 ton), often used in military and humanitarian missions. The authors highlight the operational complexity of such a process, characterized by strong human-machine-environment interaction. This complexity leads to difficulties in safety assessment and numerous accidents.

In the presented article, we discuss safety, but specifically in the context of errors resulting from the improper spatial position of the cargo carrier at the moment of release. It is crucial for the aircraft or UAV carrying humanitarian aid or specialized equipment (firefighting, research) to be outside the dangerous zone. Considering all this, specifying the initial conditions V_0, h_0, θ_0 for the cargo drop is a very important issue. It frees the carrier's operator from the need to calculate, mainly the range of such cargo, thereby eliminating the so-called "human error" in this regard. This enables real-time decision-making regarding the initial conditions for the cargo drop.

1.4. Forecasting the Drop Zone and Determining the Impact Point

Determining the area where cargo should be airdropped to reach its target is crucial in precision airdrop technology and constitutes a highly complex task that requires considering many factors. The main elements influencing this process are: wind field modeling, the flight dynamics of the cargo with a parachute, algorithms for predicting the drop target, and error compensation and real-time correction. In practice, this process requires modeling the cargo's flight trajectory from the point of release to impact, taking

into account all aerodynamic forces and atmospheric conditions. This, in turn, involves advanced algorithms and measurement technologies that enable the determination of the optimal point in the air so that the cargo lands within the intended, relatively small area on the ground. In the literature [9] we can also encounter the term Calculated Aerial Release Point (CARP).

Considering all these aspects (factors), it becomes reasonable to use a system that would allow for precise determination of the initial airdrop parameters before the flight and simultaneously ensure that the cargo reaches its designated target, taking into account the current flight parameters (altitude h_0 , velocity V_0 , and possibly the angle θ_0 at which the drop is to occur). Such conditions are fulfilled by the proposed inverse algorithm, which enables the determination of the required initial parameters—including V_0 , h_0 , and θ_0 —based on the desired landing location and target parameters. This approach effectively supports pre-flight planning and can be integrated with on-board systems to enhance precision in cargo delivery under varying flight and environmental conditions. The feasibility and efficiency of solving inverse problems using neural networks—even in complex, highly nonlinear systems—has been confirmed in other fields such as robotics. For instance, in the work of [10], inverse kinematics problems in 6-DOF manipulators were successfully addressed using MLP-based networks with additional segmentation and error correction strategies, resulting in both high accuracy and low computational cost.

In order to determine the search area for a dropped cargo, mainly in conditions of limited visibility (night, fog), a multifaceted approach is used, combining precise pre-drop calculations, cargo tracking methods, and advanced post-drop search methods. The main element of this process is forecasting the drop zone. In the initial stage of this process, the probable drop zone is calculated based on precise navigation data of the aircraft (e.g., from a GNSS system) at the moment of cargo release. It is also possible to use tracking devices that can be integrated with the dropped cargo, such as: locator beacons (e.g., ELT-type devices), GPS/GNSS trackers, radio beacons, or IoT sensors. However, it is known that navigation systems are subject to various types of interference [11]. In the context of Global Navigation Satellite Systems, the main type of interference is Radio Frequency Interference (RFI), including jamming [12]. Therefore, these interferences affecting GNSS systems can cause a complete loss of signal, data distortion, or reduced positioning precision. In each of these cases, there is a huge risk of mission failure for the cargo drop. Eliminating this problem most often involves using alternative systems, such as visual odometry systems [13] or Simultaneous Localization And Mapping (SLAM) [14]. Both the visual odometry system and SLAM can operate independently of data from GNSS systems, or they can rely on them to improve their performance. Therefore, systems used to define the cargo drop zone that utilize GPS/GNSS modules [15] have their biggest disadvantage in power consumption. This can impact their functionality and operating time, and ultimately their usability. There are also places in the world that are devoid of these types of satellite signals. Therefore, it is impossible to use systems that require data from GPS. To address this, we can use the presented method for determining three parameters: range (r_k), flight time (t_k), and final impact velocity (V_k), to define the impact point of the dropped cargo. Most importantly, the process of airdropping resources using the presented system does not require satellite navigation data. It relies on data generated before the planned mission. This system can also serve as pilot or operator support and can be activated when GPS data is unavailable.

Algorithms and models predicting the cargo's flight trajectory and its potential drift also play a crucial role in forecasting the drop zone. In the presented research, only the impact point was determined, but this provides key information during a mission. For this purpose, the geometric-mass data of our capsule were used to calculate its range, flight time, and final impact velocity using artificial neural networks. This minimizes the risk of

improper carrier positioning in space (i.e., the dropped cargo failing to reach the designated location—ocean, earth).

Another aspect discussed regarding precision cargo airdrop systems is their reliability. It should be noted that these types of airdrops require incredibly advanced systems, which can be prone to failure. The article [16] focuses on the need to estimate the reliability of airdrop systems already at the design stage, which helps avoid costly and time-consuming field tests. If such an estimation is not possible, and additionally, in the absence of information from a damaged component, the presented system can be used as support. These are typically situations where urgent cargo delivery is required, and there is no time or opportunity to fix such failures.

1.5. Application of Neural Networks in Airdrop Research

Artificial neural networks (ANNs) serve as powerful tools for modeling complex input-output relationships, making them highly effective in engineering applications, including precision airdrop systems. ANNs, inspired by biological neural systems, map inputs to outputs as $\mathbf{X} \rightarrow \mathbf{Y}(\mathbf{X}, \mathbf{w})$, where $\mathbf{X}$ and $\mathbf{Y}$ are input and output vectors, and $(\mathbf{w})$ represents network parameters [17–21]. In airdrop research, ANNs enhance precision, autonomy, reliability, and safety, despite limited scientific literature compared to commercial advancements [22].

A review of AI applications in airdrop systems highlights their use in intelligent control, trajectory prediction, reliability analysis, and multi-sensor data fusion. For instance, reinforcement learning (RL) with an Actor-Critic (AC) architecture improves precision in wind-affected environments [23]. Backpropagation neural networks (BPNNs), optimized with genetic algorithms (GAs) and validated by Kane's Equation, achieve high landing accuracy [24]. Deep RL, such as the Adaptive Priority Experience Replay Deep Double Q-Network (APER-DDQN), enables autonomous UAV maneuvers for airdrop missions [25,26]. Deep RL also enhances precision landing under random initial conditions [27]. Other AI methods, including GAs [28,29], particle swarm optimization [30,31], and Bayesian networks [16], further support trajectory optimization and system reliability. These approaches demonstrate ANNs' versatility in addressing complex airdrop challenges.

1.6. Forward and Inverse Neural Models for Airdrop Dynamics

Identification problems in airdrop systems can be addressed through forward and inverse analysis approaches. The forward mode defines an error function based on differences between numerical model outputs and experimental (numerical or measured) data, minimizing this function to refine parameters. This method is stable and widely used for numerical validation. A neural network (NN) approximates the forward relationship between input vector ($\mathbf{X}$) (e.g., initial conditions like velocity V_0 , altitude h_0 , and angle θ_0 , and output vector ($\mathbf{Y}$) (e.g., system response) as: $[\mathbf{Y} = f(\mathbf{X}, \mathbf{w})]$, where $f(\cdot)$ is the approximated function [17–21].

Conversely, the inverse mode assumes a mathematically consistent inverse relationship, enabling systematic retrieval of input parameters from desired outputs. For airdrop systems, this involves determining initial conditions V_0 , θ_0 , h_0 from target outcomes, such as final range r_k , time t_k , or velocity V_k , expressed as: $[\mathbf{X} = g(\mathbf{Y}, \mathbf{w})]$, where $g(\cdot)$ is the inverse function approximated by an NN [32–35].

The motivation for using NNs in inverse problems lies in their ability to model complex, nonlinear dynamics without reliance on real-time external data, such as GNSS, which may be limited by environmental or operational constraints. This enhances precision and reliability in trajectory planning. NNs' applicability extends to other domains, such as robotics and system control, where they effectively map desired outcomes to input

configurations, supporting robust parameter estimation for airdrop applications [32–35]. In airdrop research, the inverse problem involves determining initial conditions (velocity, altitude, and release angle) from predefined final outcomes, such as range, flight time, or impact velocity. These challenges are ill-posed and potentially ambiguous, as one final state can correspond to multiple initial sets. Classical analytical or iterative approaches are computationally intensive and sensitive to disturbances, limiting their use in dynamic scenarios. By approximating intricate input–output mappings, neural networks offer an efficient alternative, enabling rapid parameter estimation essential for mission accuracy and safety, especially in disturbed or GPS-denied environments.

1.7. Scope of the Present Study

This study focuses on an imprecise cargo airdrop system, constrained by low operational cost and the ability to operate without satellite navigation signals. The research proposes a computational framework capable of predicting and reconstructing drop parameters, supporting reliable logistical planning and reducing human error.

Two computational approaches were used: forward analysis and inverse analysis, integrated into a four-stage framework (Figure 1).

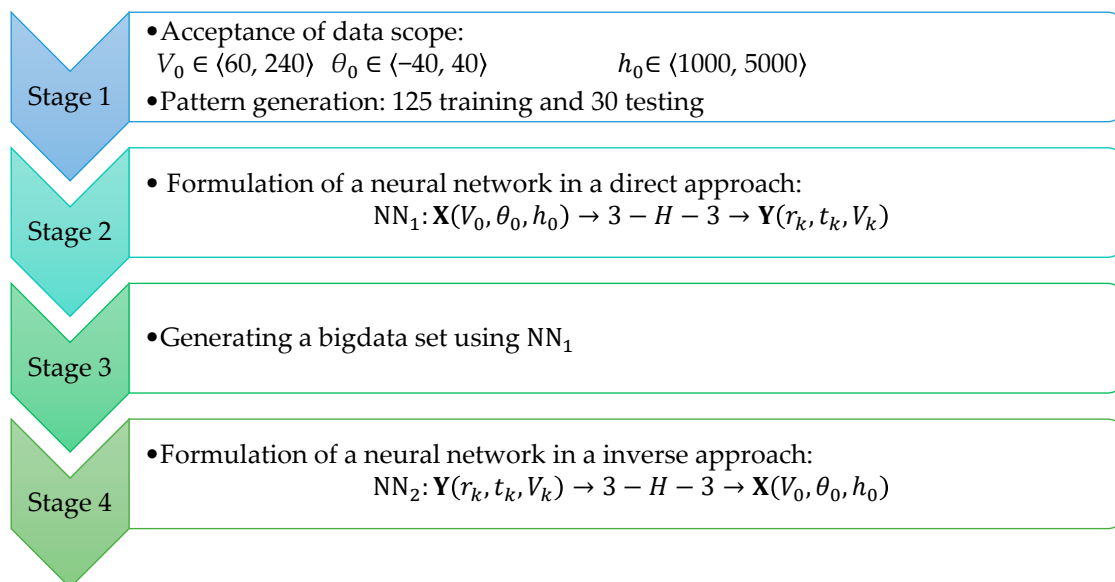


Figure 1. Diagram of the hybrid computational framework for forward and inverse analysis in precision airdrop systems, illustrating the four stages: (1) numerical simulation of capsule trajectory; (2) forward neural network for prediction; (3) dataset expansion; and (4) inverse neural network for reconstructing initial conditions.

Stage 1 involved forward numerical simulation of cargo drops, assuming initial conditions: drop velocity V_0 , θ_0 and altitude h_0 . Key outcomes—horizontal range r_k , t_k , V_k , flight time t_k , and impact velocity V_k —were calculated using a detailed trajectory model with aerodynamic forces. 125 base datasets plus 30 test sets were generated.

Stage 2 developed a neural network for forward prediction, using a simple multilayer perceptron (MLP) with one hidden layer and backpropagation, to quickly estimate r_k , t_k and V_k for arbitrary input values.

Stage 3 expanded the dataset to 8800 samples using the Stage 2 network, reducing computation time to approximately 2.3 s.

Stage 4 introduced an inverse neural network, trained on the large dataset, to infer initial conditions (V_0, θ_0, h_0) from outcome parameters (r_k, t_k, V_k) . System validation

confirmed close agreement with reference numerical simulations, showing that only the trained inverse network is needed for practical use.

In summary, while previous studies have addressed cargo airdrop safety, trajectory shaping, and neural networks applications, inverse determination of initial drop parameters remains underexplored. This work partially fills this gap, providing a neural network-based framework that supports reliable planning in GPS-denied conditions and situates the results within the context of existing research limitations.

2. Research Assumptions and Methods

2.1. Stage 1: Flight Assumptions and Capsule's Mathematical Model

For the presented research, we adopt an aerodynamic capsule shape that allows for:

- Reduced air resistance—this advantage is extremely important because it minimizes friction and air turbulence around the capsule, allowing for faster and more controlled descent,
- Improved precision and trajectory prediction—aerodynamically shaped capsules are less susceptible to the influence of crosswinds and other atmospheric disturbances. Their flight path is more stable and easier to predict, which increases the chances of a precise airdrop,
- Increased stability during descent—allows the capsule to maintain a constant orientation in flight, reducing the risk of uncontrolled spinning, swaying, or tumbling. This is particularly important to avoid cargo damage,
- Reduced loads on the capsule's structure and its contents. Laminar airflow around an aerodynamic shape minimizes the dynamic forces acting on the capsule, which reduces the risk of damage to its structure and contents, especially sensitive items such as precision measuring equipment or shock-sensitive packaged medications,
- More stable parachute opening and deployment if one is used. In the presented research, the use of a parachute was not considered. The capsule is dropped directly, e.g., from an airplane or an unmanned aerial vehicle,
- Additionally, if an increased descent speed is desired when rapid cargo delivery is a priority.

In summary, an aerodynamic shape is crucial for ensuring the safe, precise, and effective delivery of cargo in airdrop operations. This translates into minimizing the risk of damage and increasing the chances of mission success.

It should also be emphasized that damage to specialized measuring equipment or special medications during an airdrop involves high costs. These losses include not only the value of the equipment or medications themselves, but also potential delays in mission execution, loss of important research results, or the inability to provide immediate assistance in crisis situations. Therefore, ensuring the reliability and safety of airdrops is a priority, and the aerodynamic shape of the capsule is one of the factors that contributes to this.

To determine the range parameter, a mathematical model for the capsule under consideration had to be developed. To make this possible, several assumptions needed to be made:

- The capsule is a rigid solid, made of resistant materials that are not easily damaged,
- The mass of the capsule does not change with time,
- The capsule is an axisymmetric solid,
- The planes of geometric, mass, and aerodynamic symmetry are the planes Sx_vz_v ,
- There is no wind speed.

The motion of the center of mass of the capsule will be described by equations of translational and rotational motion, well-known from mechanics [36]. Newton's law was

used to derive the equations of motion for a capsule, according to which the sum of external forces acting on an object in a chosen direction is equal to the change in momentum in that direction per unit of time. Thus, the vector equation of translational motion for the capsule's center of mass can be written in the form [37]

$$F_c = m \frac{\delta V_c}{dt} + \Omega \times (m V_c), \quad (1)$$

where F_c represents the sum of all external forces in the body frame, V_c represents the velocity of the capsule, expressed in body coordinates, Ω represents the angular velocity vector of the body frame with respect to the inertial frame, also expressed in body coordinates.

Newton's second law for the rotational motion of the capsule is given by:

$$M_c = I \dot{\Omega} + \Omega \times I \Omega, \quad (2)$$

where M_c represents the sum of all external moments, expressed in the capsule body frame and I represents the moment of inertia matrix.

Most often, cargo is placed on standard aerial pallets such as ECDS (Enhanced Container Delivery System), Type V, or 463L and secured with nets. In our research, we assume that the cargo is airdropped in a special capsule. Figure 2 shows the references system and the external forces that act on the capsule during the flight.

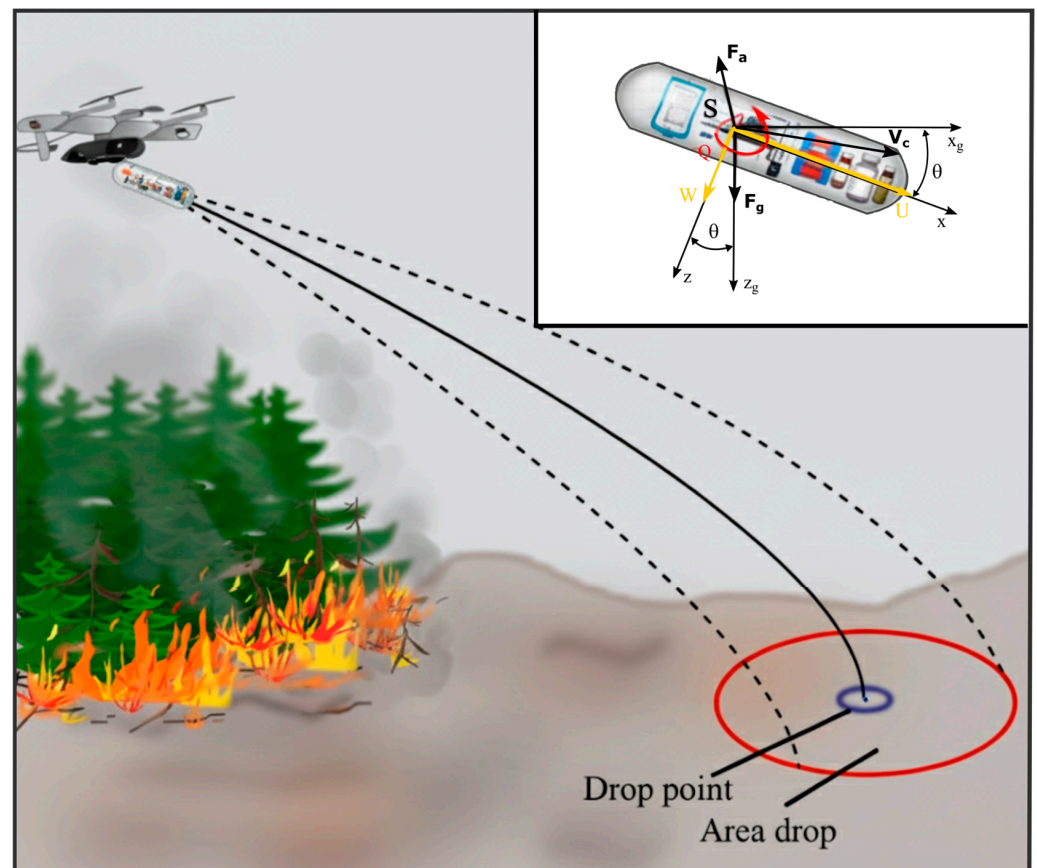


Figure 2. Trajectories, references systems and external forces acting on the air-dropped capsule.

We assume that the capsule's motion occurs exclusively in the vertical plane. For such a case, considering Equations (3) and (4), we can finally write the differential equations for longitudinal dynamics of the capsule [38]:

$$\begin{aligned}\dot{U} &= \frac{F_x}{m} - QW \\ \dot{W} &= \frac{F_z}{m} + QU \\ \dot{q} &= \frac{M_c}{I_y} \\ \dot{\theta} &= Q\end{aligned}\quad (3)$$

where F_x, F_z are the resultant forces along x and z body axes, M is the total pitching moment acting on the capsule, I_y is the moment of inertia about the pitch axis, m is capsule mass, U, W is the components of the velocity vector of the capsule in relation to the air in the boundary system $Sxyz$, Q is the component of the angular velocity vector of the capsule body and $\Omega = Q$, $V_c = [U, W]^T$.

Ultimately, the forces F_x and F_z and moment M_c acting on the capsule appearing in Equation (3) can be written in the form:

$$F_x = -mg\sin\theta - C_{aX}\frac{\rho|V_c|^2}{2}S_c \quad (4)$$

$$F_z = mg\cos\theta + \frac{\rho|V_c|^2}{2}S_c\left(-C_{aN}\left(\frac{W}{|V_c|}\right) - C_{aNr}\left(\frac{Qd}{2|V_c|}\right)\right) \quad (5)$$

$$M_c = \frac{\rho|V_c|^2}{2}S_c\left(dC_m\left(\frac{W}{|V_c|}\right)dC_q\left(\frac{Qd}{2|V_c|}\right)\right) \quad (6)$$

where g is the acceleration of gravity, ρ is the air density, d is the diameter of the capsule body, S_c is the characteristic surface (cross-sectional area of the capsule), V_c represents the velocity vector of the centre of capsule mass in relation to the air, C_{aX} is the coefficient of the aerodynamic axial force, C_{aN} is the coefficient of the aerodynamic normal force, C_{aNr} is the coefficient of the aerodynamic damping force, C_m is the coefficient of the aerodynamic tilting moment, C_q is the coefficient of the damping tilting moment.

The trajectory describes the position of the capsule's center of mass S as a function of time and external forces. Besides the gravitational force, the axial aerodynamic force F_{aX} has a significant impact on the shape of the capsule's flight trajectory. It acts along the longitudinal axis of the capsule, but opposite to its axis, which has a measurable impact on its range. The second component of the resultant aerodynamic force is the normal force F_{aN} , which is perpendicular to the Sx axis.

The primary source of difficulty in determining the aerodynamic forces and moments of a capsule is the determination of its aerodynamic characteristics. This concept refers to the coefficients of aerodynamic forces and moments acting on a capsule moving through the Earth's atmosphere [39]. These coefficients, due to their dimensionless nature, allow for the comparison and evaluation of the aerodynamic properties of flight objects of different sizes [40]. The coefficient C_{aX} of the axial force depends on the nutation angle α_t :

$$C_{aX} = C_{aX0} + C_{aX\alpha2}\sin^2\alpha_t \quad (7)$$

where C_{aX0} represents zero pitch coefficient, $C_{aX\alpha2}$ is pitch drag coefficient, α_t is the nutation angle. However, the normal force coefficient C_{aN} mainly depends on the angle of attack α .

The aerodynamic characteristics of the considered capsule were determined experimentally, in a wind tunnel.

The trajectory of the capsule's center of mass in the Earth-fixed coordinate system is obtained using appropriate transformations. Finally, for a capsule moving in the vertical plane, we obtain the following:

$$\begin{aligned}\dot{x}_g &= U\cos\theta + W\sin\theta \\ \dot{z}_g &= -U\sin\theta + W\cos\theta\end{aligned}\quad (8)$$

Numerical integration of the capsule's flight trajectory was performed using the fourth-order Runge–Kutta algorithm. We thus obtained the capsule's position for each moment in time.

In summary, the proposed system for determining initial parameters for the pilot or operator can be used as a supporting system for the precise cargo airdrop process. Incorrect determination of the initial airdrop conditions can result in mission failure or difficulties in its execution. Most importantly, the proposed system takes into account the dynamics of the dropped capsule, which reflects its actual behavior.

For the assumptions made, the range of the capsule can be determined in the form of the following:

$$r_k = \sqrt{(x_k - x_g)^2 + (z_k - z_g)^2} \quad (9)$$

where x_k, z_k are the coordinates of the end point (cargo drop), x_g, z_g are the coordinates of the initial point.

It should be noted that the available literature lacks specific data on airdrop speeds for packages containing medicines, firefighting equipment, or measurement capsules. Therefore, a rather broad range of drop speeds for the carrier was adopted in the research, falling within the interval of $V_0 \in \langle 60, 240 \rangle$ m/s. Additionally, in the research, an airdrop angle for such cargo was adopted, which directly refers to the pitch angle of the object that will be transporting this cargo.

The total mass of the capsule, including its contents, was assumed not to exceed 15 kg. Meteorological conditions and the necessity of defining aerodynamic characteristics for each falling object type further complicate accurate delivery. From the standpoint of precision and reducing the risk of uncontrolled cargo damage upon impact, the drop altitude should be kept as low as possible. As indicated in Table 1, the height of the cargo drop is within the range $h_0 \in \langle 1000, 5000 \rangle$. Moreover, the UAV's velocity at the moment of release is a key factor affecting delivery accuracy. The presented studies assumed that $V_0 \in \langle 60, 240 \rangle$ [41].

Table 1. Reference set of 125 data.

Adopted Values	Range	Computed Values	Range
V_0 [m/s]	$\langle 60, 240 \rangle$	r_k [m]	$\langle 492.8, 7346.4 \rangle$
θ_0 [deg]	$\langle -40, 40 \rangle$	t_k [s]	$\langle 5.75, 52.68 \rangle$
h_0 [m]	$\langle 1000, 5000 \rangle$	V_k [m/s]	$\langle 145.62, 291.38 \rangle$

Table 1 presents the reference dataset consisting of 125 data samples used in the research process. It includes both the adopted input parameters—initial velocity, angle, and altitude—as well as the corresponding computed output values, namely horizontal range, flight time, and impact velocity, as shown in Supplementary Materials. The input parameters were divided into four distinct ranges, which allowed for their structured distribution in preparation for analysis and neural network training conducted in Stage 2.

Figure 3 shows scatter plots illustrating the relationships between the three input variables of the model and their corresponding output parameters. The 3×3 grid layout enables a clear analysis of how each input variable influences the computed results. Distinct

patterns are visible—for example, the horizontal range r_k increases with both the initial velocity V_0 and drop altitude h_0 , while the effect of the drop angle θ_0 appears more complex. The color scheme was selected to highlight each output variable: blue for r_k , magenta for t_k , and green for V_k .

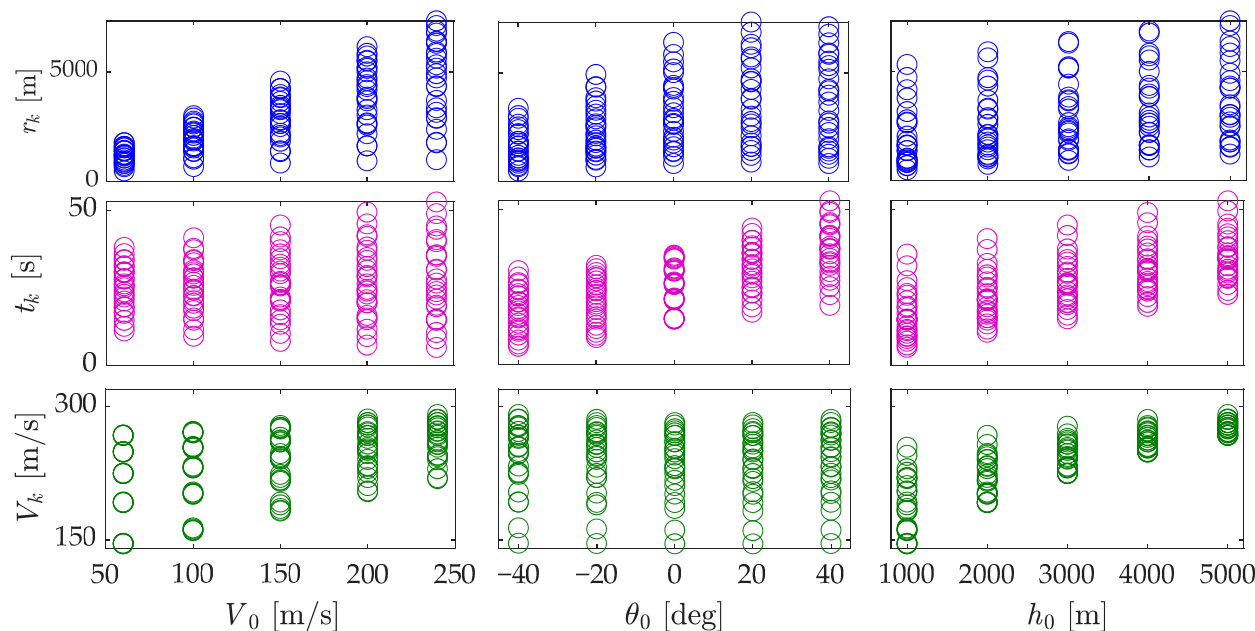


Figure 3. Visualization of the structure of the reference dataset: Point distributions illustrating the dependencies of output variables (r_k , t_k , V_k) on input parameters (V_0 , θ_0 , h_0), with clear dependency patterns for range (blue), flight time (purple), and final velocity (green).

2.2. Stage 2: Simple Neural Network: NN_1 in a Direct Approach

In this study, the same type of neural network—a backpropagation neural network—was applied in both Stage 2 and Stage 4 computations. Due to the structure of the training data, a $3 - H - 3$ architecture was adopted in Stage 2, consisting of three neurons in the input layer, H neurons in the hidden layer, and three neurons in the output layer (Figure 3). In contrast, Stage 4 employed a deeper network with two hidden layers, following a $3 - H - H - 3$ architecture (refer to Figure 4). The neurons in the input layer receive the input signals and transmit them to the hidden layers through synaptic weights. The hidden layer performs a nonlinear transformation of the input data, which enables the network to model complex nonlinear relationships between inputs and outputs. The output from the hidden layer is passed to the output layer, which generates the final result vector. The learning process is carried out using the backpropagation method, which minimizes the error function, typically by means of the gradient descent algorithm. During training, the derivatives of the error with respect to the weights are computed, and the weights are subsequently updated in the direction opposite to the gradient in order to reduce the prediction error. Neural networks of this type are commonly used for classification, regression, and pattern recognition tasks. A critical aspect of their design is the selection of the number of neurons in the hidden layer (H), which directly affects the network's ability to generalize and to avoid overfitting. A network with a $3 - H - 3$ architecture is an example of a multilayer perceptron (MLP). In general terms, the operation of the network can be expressed as the following formula:

$$\mathbf{Y} = F^{out}(\mathbf{w}^{out} \cdot F^h(\mathbf{w}^h \cdot \mathbf{X} + \mathbf{b}^h) + \mathbf{b}^{out}) \quad (10)$$

where $\mathbf{X}$ —input vector (e.g., control parameters, material parameters), $\mathbf{Y}$ —output vector (e.g., observed system response, simulation result), $\mathbf{w}$, $\mathbf{b}$ —set of neural network parameters (weights and biases), $F^{out}(\cdot)$, $F^h(\cdot)$ —activation functions of output and hidden layer, respectively.

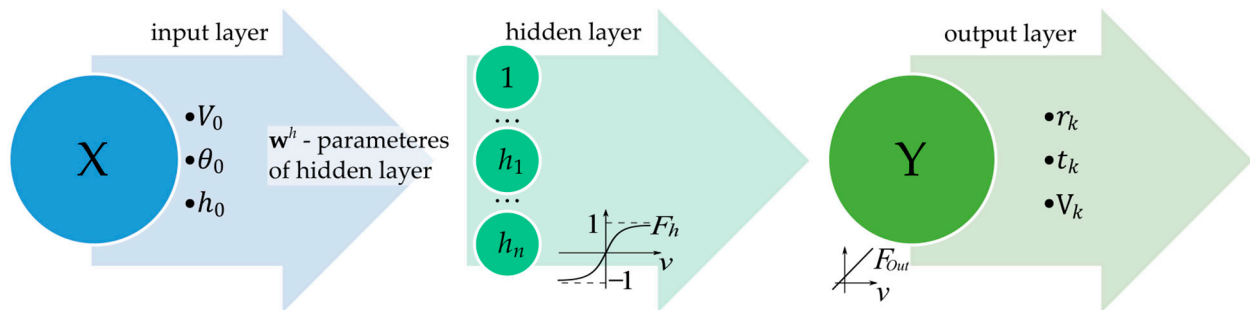


Figure 4. Structure of the NN_1 network with a 3 – H – 3 architecture, designed to predict flight parameters for airdrops.

At the beginning of Stage 2, the data computed in Stage 1 had to be appropriately preprocessed. Due to the activation function used in the hidden layer ($F^{out}(\cdot) = \tanh(\cdot)$), which operates within the range $(-1, 1)$, the input data also needed to be scaled to this interval. The tanh function, defined by the formula:

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (11)$$

is commonly used in neural networks due to its normalization properties and its ability to model nonlinear dependencies. The choice of tanh was motivated by the need for symmetric output normalization, which was particularly relevant for our dataset due to its input characteristics. Compared to the ReLU function, which excels in deep networks by mitigating the vanishing gradient problem, tanh provides better normalization for our moderately deep model. Meanwhile, the Universal Activation Function (UAF) [42], while more flexible for complex problems, was less suitable due to its higher computational complexity and the lack of necessity for such an advanced activation in our task. The scaling factors and adjusted data ranges are presented in Table 2.

Table 2. Prepared learning and testing data.

Values	Range	Scale Factor	New Range
V_0 [m/s]	$\langle 60, 240 \rangle$	252.0	$\langle 0.2381, 0.9524 \rangle$
θ_0 [deg]	$\langle -40, 40 \rangle$	42.0	$\langle -0.9524, 0.9524 \rangle$
h_0 [m]	$\langle 1000, 5000 \rangle$	5250.0	$\langle 0.1905, 0.9524 \rangle$
r_k [m]	$\langle 492.8, 734.4 \rangle$	7713.7	$\langle 0.0639, 0.9524 \rangle$
t_k [s]	$\langle 5.75, 52.68 \rangle$	55.3	$\langle 0.1039, 0.9524 \rangle$
V_k [m/s]	$\langle 145.62, 291.38 \rangle$	305.9	$\langle 0.4759, 0.9524 \rangle$

For the development of the resulting free-flight parameters of the air-dropped packages, a backpropagation neural network with a 3 – H – 3 architecture was implemented (Figure 4). This network is referred to as NN_1 throughout the study.

The input layer consisted of a three-element vector representing the predefined flight assumptions: V_0 , θ_0 , h_0 . The number of neurons H in the hidden layer was determined adaptively. During the preliminary computational phase, multiple network configurations were evaluated, with H varying from 4 to 16. It was observed that for $H \geq 7$, the learning and testing errors, measured using Mean Squared Error (MSE), satisfied the predefined

accuracy threshold set by the authors. Consequently, this network architecture was selected for further analysis (see Figure 3). The MSE formula is expressed as follows:

$$MSE = \frac{1}{P} \sum_{p=1}^P \sum_{i=1}^M \left(t_i^{(p)} - y_i^{(p)} \right)^2 \quad (12)$$

where $p = 1, \dots, P$ —pairs of data, $y_i^{(p)}$ —reference data, $t_i^{(p)}$ —computed values.

Table 3 presents the Mean Percent Error (MPE) and Mean Squared Error (MSE, see Equation (12)) for both the training and testing phases of the adopted network, evaluated in the normalized input space scaled to the range $[-1, 1]$. The results indicate a high-quality fit to the training dataset, confirming the network's ability to accurately model the underlying functional relationships.

Table 3. MPE and MSE in learning and testing phase of NN₁.

Values	Learning Phase		Testing Phase	
	MPE [%]	MSE	MPE [%]	MSE
r_k [m]	2.03	0.0000560	2.01	0.0000518
t_k [s]	1.14	0.0000331	1.30	0.0324275
V_k [m/s]	0.68	0.0000421	0.63	0.1926063

The trained neural network (NN₁) serves a crucial role in the subsequent analytical phase, where it will be efficiently utilized to generate a comprehensive dataset. This dataset will then constitute the training set for NN₂, which is designed to solve the inverse problem.

2.3. Stage 3: Generating a Big Dataset Using NN₁

At this Stage, the formulated neural network NN1 was employed to solve the direct problem. To generate the dataset, the input variables V_0 , θ_0 , and h_0 were once again used to compute the corresponding output set r_k , t_k , and V_k by means of the NN₁ network. The adopted values were divided into 20 or 22 ranges.

In total, 8800 reference datasets were generated in this manner, with the entire computational process taking 0.218889 s. The range of the input values used, as well as the resulting output values, was consistent with those presented in Table 1. Figure 5 presents an example of the data structure for the variable r_k , shown as a function of the input parameters h_0 . The plot illustrates how the generated dataset is distributed with respect to the original reference data in the input space. As shown, both datasets span the same input domain; however, the network-generated data exhibits a significantly higher density in the input space. This increased granularity provides a richer representation of the feature space, enabling more effective training of the inverse model (NN₂). The clear continuity and smoothness of the generated surface also confirm the stability and generalization capability of the NN₁ network across the considered input range.

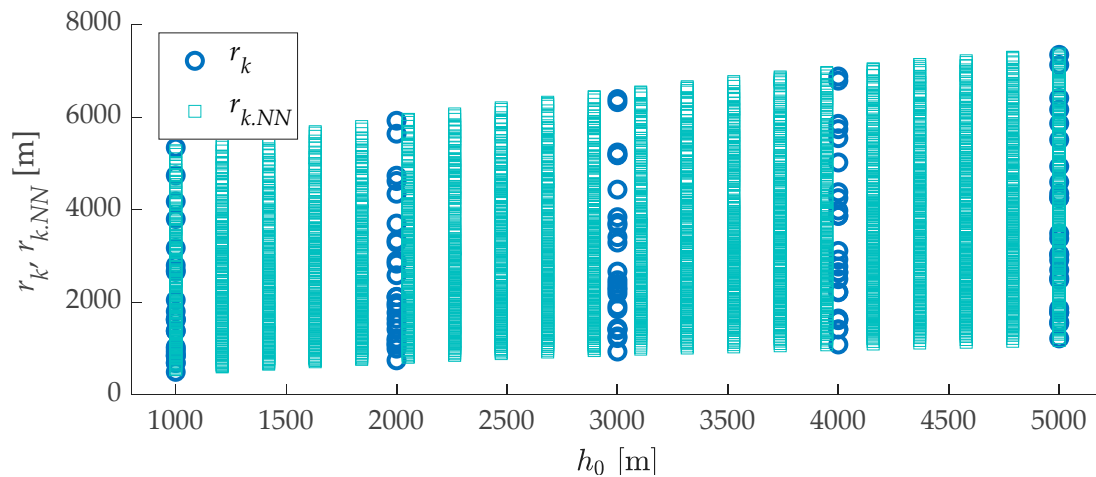


Figure 5. Example mapping of the output variable r_k as a function of h_0 , comparing reference data and NN_1 -generated data.

2.4. Stage 4: Formulation of a Neural Network in an Inverse Approach NN_2

Similarly to the second Stage, a backpropagation neural network was also employed in this case. In this study, a neural network with two hidden layers was utilized to enable more effective modeling of nonlinear relationships between input features and output. Although neural networks with a single hidden layer are universal approximators of continuous functions, architectures with two hidden layers can often achieve better generalization performance and require fewer neurons to reach comparable accuracy levels [43–45]. In particular, when analyzing drop trajectories and performing inverse estimation of initial conditions, the use of two layers allows for capturing more complex and hierarchical data patterns [46]. Herein, a neural network featuring two hidden layers was implemented (cf. Figure 6). The training dataset for this fourth Stage comprised 8800 patterns, which were generated in the third Stage. In contrast to the preceding Stage 2, this neural network was specifically designed to perform the inverse task, i.e., computing the values of parameters that were originally considered as inputs in the direct problem — V_0 , θ_0 , h_0 . Such a computational procedure is essential, as reversing the process of determining the initial parameters based on the object's range r_k , along with auxiliary parameters t_k and V_k , would be infeasible using classical analytical approaches, commonly referred to as “hard computing” methods also known as a classical numerical method. Accordingly, the neural network functions as an approximative model that enables the reconstruction of input values based on the observed output parameters. In practical applications, the parameter r_k does not necessarily need to be derived from GPS data. In GPS-denied environments, it can be obtained using hybrid approaches such as inertial navigation systems (INS) combined with visual odometry. In this way, the proposed model remains applicable and reliable even when satellite positioning is unavailable. This approach allows for the effective solution of the inverse identification problem under conditions of high nonlinearity and potentially ambiguous relationships between variables.

A functional representation of the NN_2 network is given by:

$$NN_2: X(r_k, t_k, V_k) \rightarrow 3-H1-H2-3 \rightarrow Y(V_0, \theta_0, h_0) \quad (13)$$

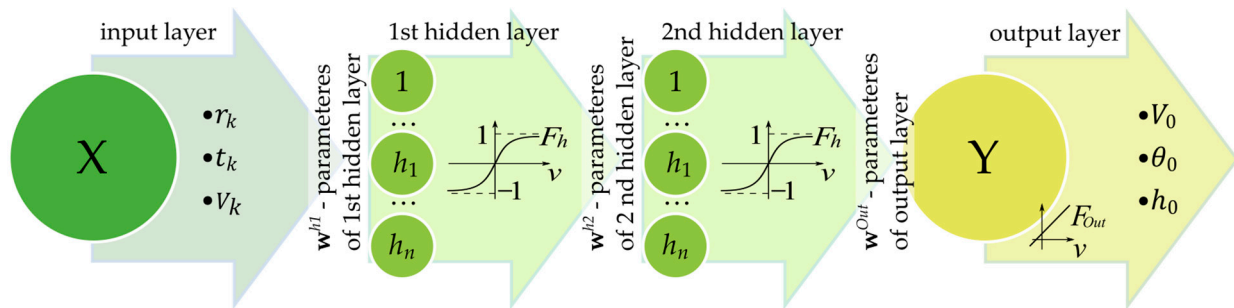


Figure 6. Structure of neural network (NN₂) used for the inverse modeling task. Structure of NN₂.

At this Stage, the developed neural network employed a 3–H1–H2–3 architecture. The training process was completed after 457 iterations (epochs), with $H1 = 6$ neurons in the first hidden layer and $H2 = 6$ neurons in the second hidden layer. Preliminary tests with alternative configurations (e.g., 3 – 12 – 3 and 3 – 8 – 8 – 3) showed that, although all networks achieved acceptable accuracy, the two-layer 3 – 6 – 6 – 3 architecture provided slightly lower errors and more stable convergence. Therefore, it was selected as the baseline inverse model. A more extensive ablation study is left for future work. For the NN₂ network, the Mean Squared Error (MSE), Mean Percent Error (MPE), and Maximum Percent Error (maxPE) were calculated to evaluate training performance. The results are summarized in Table 4.

Table 4. MPE, maxPE and MSE in Stage 4.

Values	Learning Phase		
	MPE [%]	maxPE [%]	MSE
V_0 [m/s]	0.96	7.67	0.00002
θ_0 [deg]	2.68	75.01	0.00006
h_0 [m]	0.88	10.62	0.00001

Based on the data presented in the table, the highest mean percentage error was observed for the release angle θ_0 , amounting to 2.68%, while the lowest error was recorded for the release height h_0 at 0.88%. The maximum percentage error also occurred for the release angle, reaching 75.01%. The low mean squared error (MSE) values, on the order of 10^{-5} for all output variables, indicate high accuracy in reproducing the training data and a proper fit of the model to the training set.

These results confirm that the proposed neural network architecture NN₂: 3 – 6 – 6 – 3 effectively models the relationship between the projection parameters and the corresponding initial values. To further illustrate the performance of the NN₂ network, Figure 7 presents bar plots of the percentage error distributions for each output variable. Analysis of these plots suggests that the largest prediction errors are limited to only a few individual patterns, whereas for the majority of the data, the relative error remains low and within an acceptable range.

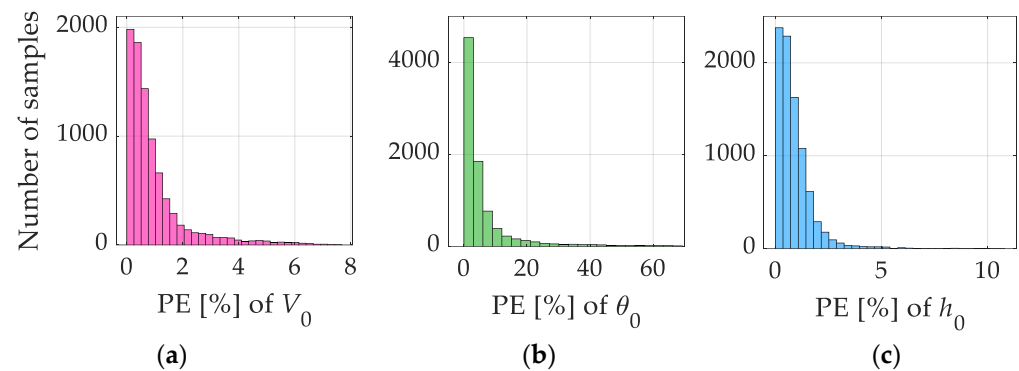


Figure 7. Bar plots of the percentage error of NN_2 , for the individual outputs: (a) initial speed V_0 , (b) drop angle θ_0 , (c) drop height h_0 .

3. Results

The primary objective of the analysis was to determine whether it is possible to reliably estimate the corresponding initial conditions (initial velocity, release angle, and drop height) based on the known and expected range of a dropped object. Two test scenarios were conducted, differing in their approach to defining the input data for the neural network NN_2 .

- **Case 1—deterministic verification:** in this scenario, all input variables of the network (range, flight time, and final velocity) are taken from the reference dataset (Stage 1), and the outputs generated by NN_2 ($V_{0.NN}$, $\theta_{0.NN}$, $h_{0.NN}$) are compared to the corresponding values from the reference set to assess their accuracy. The errors are calculated by comparing the original data (Stage 1) with the outputs of the neural network NN_2 .
- **Case 2—verification using random datasets:** the input data for the network are sampled from predefined intervals based on a discrete uniform distribution. The results obtained from NN_2 are verified by analyzing the final parameters after conducting simulations using the estimated initial conditions.

In all cases, the results are compared in the physical (unscaled) domain, which allows for a direct assessment of their practical applicability. It is assumed that, under real-world conditions, the range value is determined based on data acquired from a GPS system.

3.1. Case 1

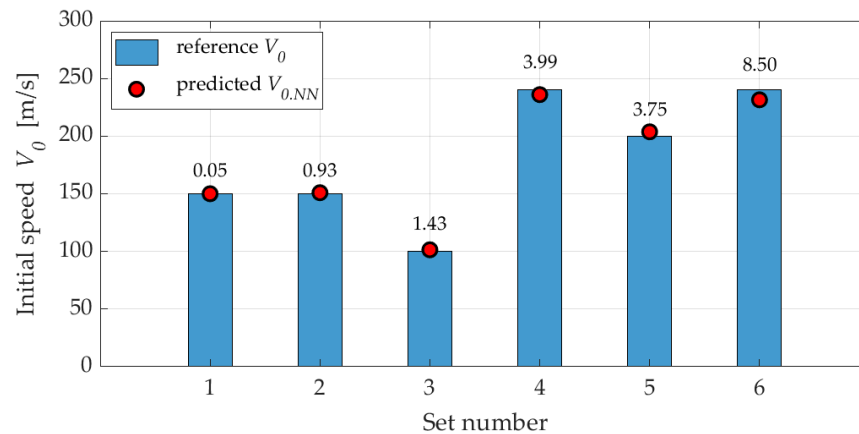
In this test, six datasets were selected from the training set generated in Stage 1 for verification using the NN_2 : 3–6–6–3 network, which performs the inverse task. It should be noted that these datasets overlap with those used for training the NN_1 network. As shown in Table 5, this approach allows for achieving relatively low errors. In the selected set, the highest relative percentage error (EP) was 14.47% and occurred in only one case, concerning the release angle.

The results presented in Table 5 were grouped according to three analyzed parameters: initial speed, drop angle, and drop height. To further illustrate the neural network's performance, a separate plot was generated for each of these groups, comparing the reference values with the network's predicted values.

For the initial speed plot (Figure 8), the discrepancies observed between the actual data and the neural network predictions are minimal, not exceeding a few units. This demonstrates the model's high accuracy in estimating the initial parameters: $V_{0.NN}$, $\theta_{0.NN}$, $h_{0.NN}$.

Table 5. Calculation procedure and error analysis for Case 1.

Set No.	Variable (A)	Stage 1 Output	Stage 4 Output (B)	Percent Error (A–B) [%]
	V_0 [m/s] θ_0 [deg] h_0 [m]	Input to Stage 4 r_k [m] t_k [s] V_k [m/s]	$V_{0,NN}$ [m/s] $\theta_{0,NN}$ [deg] $h_{0,NN}$ [m]	
1 (7)	150.00	1450.20	150.05	0.03
	−30.00	11.89	−29.07	3.10
	1500.00	207.85	1440.27	3.98
2 (39)	150.00	1413.30	150.93	0.62
	−40.00	13.23	−39.66	0.85
	2000.00	223.83	1943.17	2.84
3 (59)	100.00	1538.50	101.43	1.43
	−20.00	17.61	−22.89	14.47
	2000.00	202.70	2102.53	5.13
4 (77)	240.00	4934.00	236.01	1.66
	−20.00	27.83	−20.71	3.53
	5000.00	285.84	4992.56	0.15
5 (95)	200.00	4432.50	203.75	1.88
	0	26.25	1.65	-
	3000.00	252.51	2851.01	4.97
6 (123)	240.00	4738.80	231.50	3.54
	20.00	24.86	20.84	4.19
	1000.00	219.08	1061.39	6.14

**Figure 8.** Initial speed V_0 reference and predicted values.

For the drop angle plot (Figure 9), only minor discrepancies are observed. Despite the greater variability of this parameter in the dataset, the predicted values remain close to the reference data, indicating good generalization capability of the model.

The drop height plot (Figure 10), on the other hand, shows that even for larger magnitudes (on the order of thousands of meters), the absolute errors remain relatively small. This indicates that the network can effectively reproduce the data across the entire range of drop heights.

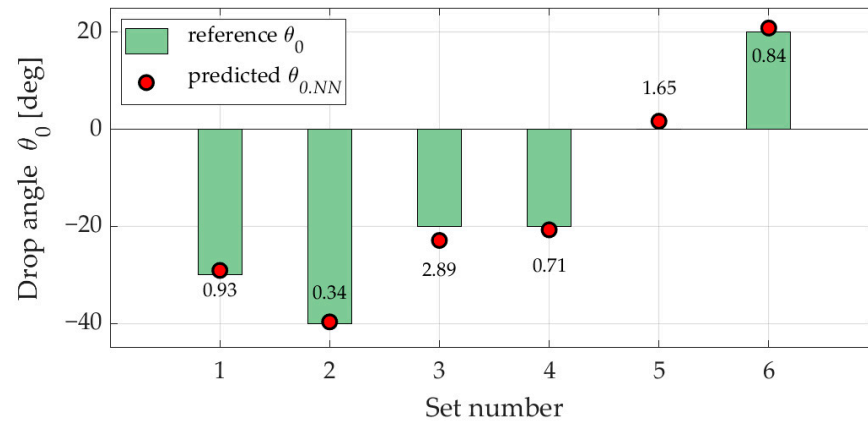


Figure 9. Drop angle θ_0 reference and predicted values.

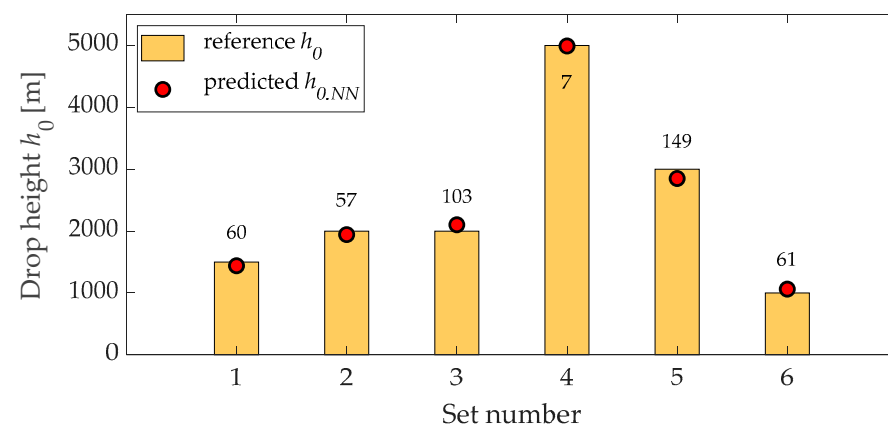


Figure 10. Drop height h_0 reference and predicted values.

In summary, the differences between the reference data and the predictions are minor in the context of the overall scale of the respective parameters, which confirms the high quality of the predictions generated by the applied neural network NN₂.

3.2. Case 2

In the Type II test scenario (Case 2), 18 input datasets were generated for the inverse neural network (NN₂). Each set included three key final parameters describing the course of an airdrop operation: the delivery range $r_{k.rand}$, the fall time $t_{k.rand}$, and the final velocity at touchdown $V_{k.rand}$. These data were generated randomly based on a discrete uniform distribution, meaning that each integer within a given interval had an equal probability of being selected. The applied intervals were as follows:

- $r_{k.rand} \in \langle 492, 7347 \rangle$ [m],
- $t_{k.rand} \in \langle 5, 53 \rangle$ [s],
- $V_{k.rand} \in \langle 145, 292 \rangle$ [m/s].

The use of a uniform distribution ensures the absence of any preference toward specific regions of the input space, which is particularly important in analyses aimed at evaluating the model's response across the full spectrum of possible operational scenarios—without implicit assumptions regarding their likelihood [47].

Inverse problems—where the final parameters are known and the initial conditions are sought—are generally more challenging to solve than problems based on direct cause-effect relationships. In the classical forward approach (e.g., $V_0, \theta_0, h_0 \rightarrow r_k, t_k, V_k$), a single set of input values typically leads to a unique solution. In the inverse direction (i.e., $r_k, t_k, V_k \rightarrow$

V_0, θ_0, h_0), ambiguities may arise, as different initial conditions can produce similar final outcomes. Moreover, for certain combinations of final values, a physically feasible solution may not exist at all (e.g., an unrealistically short time relative to the range, or a negative drop height).

The data used in the training process (for both NN_1 and NN_2) do not uniformly cover the entire space of possible final parameters. This implies the existence of substantial regions within the input space for which the neural network was not previously exposed to reference data. Random sampling may inadvertently target such “gaps,” increasing the risk of extrapolation beyond known regions, which can lead to inaccurate or unreliable results.

For each test sample, the minimum absolute difference (distance) between the randomly generated value and the closest value in the training dataset was calculated—separately for each parameter: r_k, t_k and V_k . This enabled an estimation of how “close” the test data were to the known (training) data.

The computational scheme illustrating the data flow in the analyzed scenario was as follows:

$$\left. \begin{matrix} r_{k.rand} \\ t_{k.rand} \\ V_{k.rand} \end{matrix} \right\} \xrightarrow{NN_2} \left. \begin{matrix} V_{0.NN} \\ \theta_{0.NN} \\ h_{0.NN} \end{matrix} \right\} \xrightarrow{\text{analytical method}} \left. \begin{matrix} r_{k.new} \\ t_{k.new} \\ V_{k.new} \end{matrix} \right\} \quad (14)$$

The initial parameters predicted by the inverse neural network— $V_{0.NN}, \theta_{0.NN}, h_{0.NN}$ —did not have assigned reference values. As a result, it was not possible to directly compare them with the real reference values. Their quality was assessed indirectly—by analyzing the resulting final values $r_{k.new}, t_{k.new}, V_{k.new}$ obtained from a simulation based on the initial parameters returned by NN_2 .

Table 6 presents the data used in the Type II test scenario, in which the inverse neural network (NN_2) receives randomly assigned values of the range variable ($r_{k.rand}$) and is tasked with determining the initial conditions that lead to similar outcomes. The table compares these target values ($r_{k.rand}$) with the corresponding values ($r_{k.new}$) obtained through simulation, following the complete inverse prediction chain. Additionally, the table includes the percentage error, absolute error, and the minimum distance of each point from the nearest training sample (Min. Dev. of r_k). The general formula for calculating the minimum deviation (Min. Dev.) of any variable x from the training set is given by:

$$\Delta x_{k_i} = \min_j (x_{k.rand_j} - x_{k_i}) \quad (15)$$

where $x_{k.rand_j}$ denotes the set of training samples for the variable x .

Table 6. Range: random values vs. new reference values and deviation analysis.

Draw No.	Input to Stage 4 (C): $r_{k.rand}$ [m]	Stage 1 Result (D): $r_{k.new}$ [m]	Percent Error (C–D) [%]	Absolute Error (C–D) [m]	Min. Dev. of Δr_k [m]
1	4213.00	4205.10	0.19	7.90	35.30
2	2692.00	2569.70	4.76	122.30	4.10
3	3504.00	3436.00	1.98	68.00	17.40
4	2257.00	2319.0	2.67	62.00	7.20
5	3567.00	3907.30	8.71	340.30	80.40
6	2966.00	3064.90	3.23	98.90	44.00
7	3353.00	3344.80	0.25	8.20	21.00
8	1925.00	2108.80	8.72	183.80	6.00

Table 6. Cont.

Draw No.	Input to Stage 4 (C): $r_{k.rand}$ [m]	Stage 1 Result (D): $r_{k.new}$ [m]	Percent Error (C–D) [%]	Absolute Error (C–D) [m]	Min. Dev. of Δr_k [m]
9	755.00	797.27	5.30	42.27	11.37
10	3067.00	3132.80	2.10	65.80	24.80
11	6647.00	6731.70	1.26	84.67	139.90
12	3309.00	3238.90	2.16	70.10	11.50
13	2636.00	2460.00	7.15	176.00	1.30
14	705.00	776.87	9.25	71.87	12.99
15	1453.00	1431.80	1.48	21.20	2.80
16	5191.00	5185.10	0.11	5.90	6.90
17	5921.00	5988.70	1.13	67.70	0.50
18	6334.00	5597.10	13.17	736.90	11.70

Table 6 presents the data used in the Type II test scenario, in which the inverse neural network (NN₂) receives randomly generated reference values of the range variable ($r_{k.rand}$). Its task is to determine the initial conditions that lead to similar outcomes. The table compares these reference values ($r_{k.rand}$) with the values $r_{k.new}$, which were obtained through simulation after executing the complete inverse prediction chain. Additionally, the table reports the percentage error, absolute error, and the minimum distance of the given point from the nearest training sample (Min. Dev. of Δr_k).

The data analysis indicates that the inverse neural network (NN₂) accurately reconstructs the parameters leading to the desired range in most cases, particularly in regions well-represented by the training data. The average absolute error for the entire dataset is approximately 124.6 m, with an average relative error of around 4.14%, which is satisfactory given the wide variability in range values (from several hundred to over 6000 m). The best results were obtained for test cases where the minimum distance from the training data (Min. Dev., Section 3.2) was small—below 10 m (e.g., samples 1, 7, and 16).

However, in data-sparse regions, such as sample 18 (13.17% error, 736.9 m, Table 6) and sample 5 (8.71% error, 340.3 m), significant deviations were observed. These errors arise due to the model's extrapolation beyond well-represented regions of the training data, highlighting its fragility in such scenarios. Additionally, some larger errors occurred in samples with low Min. Dev. values, suggesting potential local discontinuities or ambiguities inherent to the inverse problem. To improve the model's performance in future iterations, we are considering techniques such as data augmentation (e.g., generating synthetic samples in sparse regions) and regularization methods (e.g., dropout) to enhance robustness during extrapolation. Further research will focus on increasing the density of training data in underrepresented regions to reduce errors in such cases.

Figure 11 presents a graphical analysis of the performance accuracy of the inverse neural network in the Type II test scenario (Case 2). The lower plot illustrates a comparison between the randomly generated range values ($r_{k.rand}$, red dots) and the corresponding $r_{k.new}$ values (bars) obtained through the inverse prediction chain. The horizontal axis represents the sample (draw) number, while the vertical axis shows the range value in meters.

The upper part of the plot illustrates the minimum distance (in meters) between each random sample $r_{k.rand}$ and the nearest training point from the original dataset. This provides insight into how far a given sample is from known training data, which may affect the prediction quality.

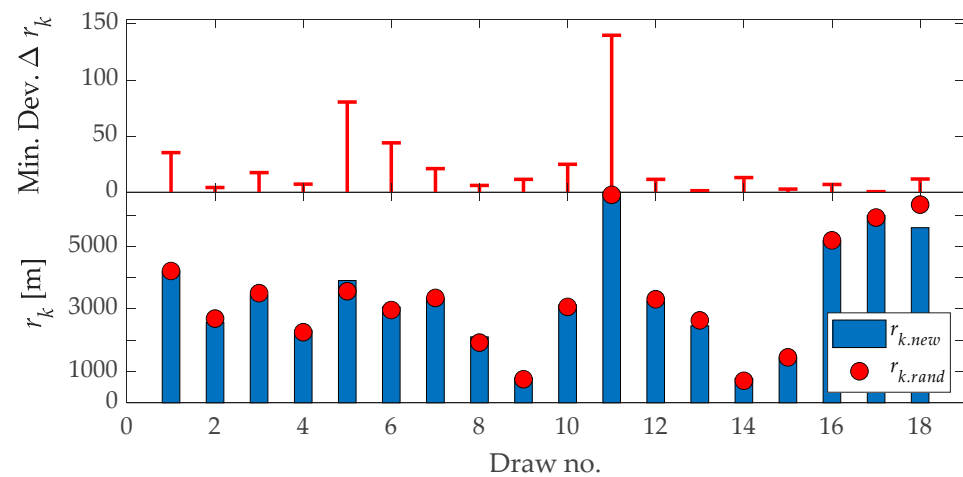


Figure 11. Flight range analysis: comparison of random values $r_{k.rand}$ with reference data $r_{k.new}$ and visualization of minimum deviation.

Table 7 presents a comparison between the randomly generated time-of-flight values ($t_{k.rand}$), which served as input for Stage 4 of the inverse model, and the new reference values ($t_{k.new}$) obtained through a two-step predictive process. The table also includes the absolute and percentage errors, as well as the minimum deviation of each sample from the nearest element in the training dataset (Δt_k), allowing for an assessment of how far a given sample is from known cases within the network.

Table 7. Flight time: random values vs. new reference values and deviation analysis.

Draw No.	Input to Stage 4 (C): $t_{k.rand}$ [s]	Stage 1 Result (D): $t_{k.new}$ [s]	Percent Error (C–D) [%]	Absolute Error (C–D) [s]	Min. Dev. of Δt_k [s]
1	26.00	25.86	0.54	0.14	0.010
2	18.00	23.35	22.91	5.35	0.145
3	33.00	33.11	0.33	0.11	0.048
4	15.00	15.00	0	0	0.114
5	48.00	44.94	6.81	3.06	0.924
6	41.00	41.13	0.32	0.13	0.133
7	38.00	37.65	0.93	0.35	0.001
8	42.00	29.28	43.44	12.72	0.121
9	14.00	14.26	1.82	0.26	0.241
10	36.00	36.37	1.02	0.37	0.018
11	41.00	40.96	0.10	0.04	0.133
12	17.00	16.45	3.34	0.55	0.080
13	31.00	30.20	2.65	0.80	0.005
14	17.00	17.23	1.33	0.23	0.080
15	12.00	11.90	0.84	0.10	0.114
16	33.00	34.05	3.08	1.05	0.048
17	38.00	38.99	2.54	0.99	0.001
18	48.00	38.93	23.30	9.07	0.924

The results show that, for the majority of samples, the absolute differences between $t_{k.rand}$ and $t_{k.new}$ are small—typically below 1 s—indicating the model’s strong ability to reconstruct temporal parameters when the test data fall within the range represented by the training set. Exceptions to this pattern occur for samples located outside the main region of the training space, as confirmed by higher Δt_k values. In such cases

(e.g., samples 2, 8, and 18), the percentage errors exceeded 20%, with the maximum absolute deviation reaching 9.07 s (sample 18).

These results confirm that the inverse model exhibits high predictive accuracy when performing interpolation within well-represented regions of the training data space. However, in situations that require extrapolation into sparsely represented areas of the input space, the accuracy declines significantly, leading to increased prediction errors.

Figure 12 provides a graphical interpretation of the results presented in Table 7. The lower plot displays a comparison between the randomly generated time-of-flight values ($t_{k.rand}$ —red points) and the new reference values ($t_{k.new}$ —blue bars), obtained by the inverse model based on the predicted initial conditions. The horizontal axis corresponds to the sample (draw) number. The upper plot indicates the minimum deviation of each $t_{k.rand}$ value from the nearest training sample (Δt_k), allowing for an estimation of how far each test sample deviates from the known range of the training data.

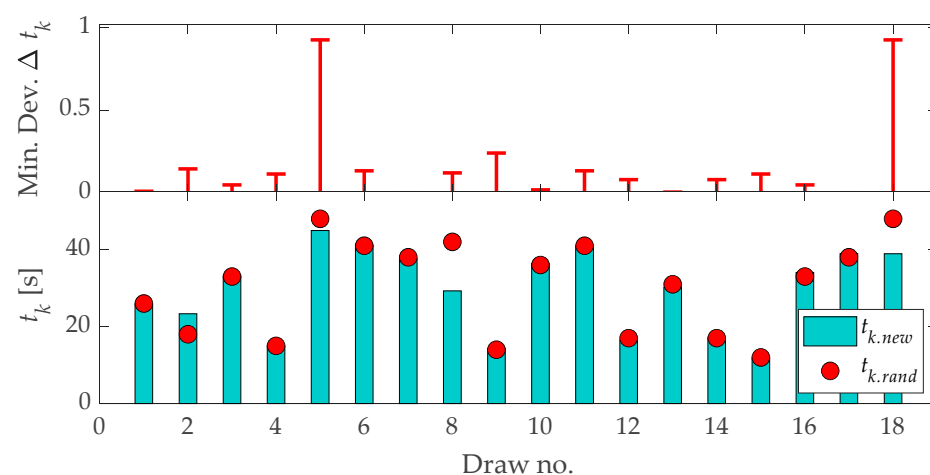


Figure 12. Flight time analysis: comparison of random values $t_{k.rand}$ with reference data $t_{k.new}$ and visualization of minimum deviation.

The plot reveals that, in most cases, the differences between $t_{k.rand}$ and $t_{k.new}$ are relatively small, indicating high-quality inverse prediction in the context of interpolation. However, in cases where $t_{k.rand}$ is located farther from the training points (e.g., samples 5, 8, and 18), significant prediction errors emerge—both in terms of absolute deviation and elevated Δt_k values. This relationship confirms the strong influence of the representativeness of the training data on the effectiveness of inverse prediction.

Table 8 presents the results of a comparison between the randomly assigned values of the final contact velocity ($V_{k.rand}$) and their corresponding reference values ($V_{k.new}$), as determined by the inverse model. For each pair, the table also includes the relative and absolute errors, as well as the minimum deviation (Min. Dev. ΔV_k) of the given random sample from the nearest value in the training dataset, providing a basis for assessing interpolation quality.

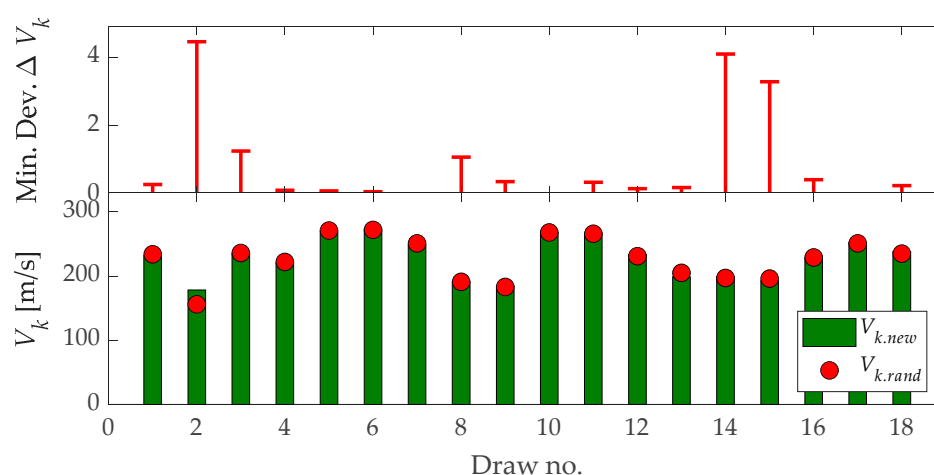
The analysis indicates that, for the vast majority of cases, the differences between $V_{k.rand}$ and $V_{k.new}$ are minimal—both the relative and absolute errors remain within a few percent. The largest deviations were observed in samples 2, 13, 14, and 15, where the relative error exceeded 3%, and $V_{k.rand}$ was located farther from the reference data (e.g., Min. Dev. $\Delta V_k = 4.45$ m/s in row 2). In the remaining cases, the discrepancies fall within the range of measurement error or are negligible for practical applications.

Table 8. Final descent velocity: random values vs. new reference values and deviation analysis.

Draw No.	Input to Stage 4 (C): $V_{k.rand}$ [m/s]	Stage 1 Result (D): $V_{k.new}$ [m/s]	Percent Error (C–D) [%]	Absolute Error (C–D) [m/s]	Min. Dev. of ΔV_k [m/s]
1	234.00	231.88	0.91	2.12	0.30
2	156.00	178.19	12.45	22.19	4.45
3	236.00	235.15	0.36	0.85	1.27
4	222.00	220.97	0.47	1.03	0.13
5	271.00	270.37	0.23	0.63	0.11
6	272.00	272.33	0.12	0.33	0.09
7	251.00	248.95	0.82	2.05	0.03
8	191.00	191.81	0.42	0.81	1.09
9	183.00	184.06	0.58	1.06	0.38
10	268.00	268.18	0.07	0.18	0.02
11	266.00	267.33	0.50	1.33	0.36
12	231.00	233.47	1.06	2.47	0.18
13	205.00	199.00	3.02	6.00	0.21
14	197.00	198.95	0.98	1.95	4.09
15	196.00	192.70	1.71	3.30	3.28
16	229.00	228.33	0.29	0.67	0.44
17	251.00	252.26	0.50	1.26	0.03
18	235.00	238.14	1.32	3.14	0.27

The conclusions drawn from the table confirm the effectiveness of the inverse model's predictions, provided that the training data space is sufficiently covered. The smaller the deviation from the existing training set, the higher the accuracy of the resulting predictions.

Figure 13 presents a comparison between the randomly generated final descent velocity values ($V_{k.rand}$ —red dots) and their corresponding reference values ($V_{k.new}$ —green bars), determined based on the predictions of the inverse model. The lower panel displays the values of both sets across 18 random samples, while the upper panel shows the minimum deviation of $V_{k.rand}$ from the nearest point in the training dataset (ΔV_k), indicating the degree of similarity between each test sample and the training data.

**Figure 13.** Fall speed analysis: comparison of random values $V_{k.rand}$ with reference data $V_{k.new}$ and visualization of minimum deviation.

The plot indicates that, in most cases, the inverse model accurately reproduced V_k —the $V_{k.new}$ values closely match $V_{k.rand}$, particularly where ΔV_k was small. Noticeably larger

prediction errors occurred only in a few instances (e.g., samples 1, 5, and 15), where the deviation from the training data was relatively substantial.

Table 9 presents the values of the initial conditions predicted by the inverse model, i.e., the initial velocity $V_{0,NN}$, launch angle $\theta_{0,NN}$, and release altitude $h_{0,NN}$, for each tested case. Since no direct reference data were available for these parameters, no quantitative error metrics are provided at this Stage. Nevertheless, these results constitute the core output of the neural prediction process and form the basis for the subsequent forward simulation and verification Stages.

Table 9. Baseline flight parameters determined by the NN₂ network.

Draw No.	$V_{0,NN}$ [m/s]	$\theta_{0,NN}$ [°]	$h_{0,NN}$ [m]
1	192.53	11.22	2126.03
2	140.17	30.44	1000.00
3	138.71	30.62	2816.64
4	174.76	−15.08	1663.18
5	153.44	46.80	4399.42
6	108.77	36.14	4993.07
7	133.79	39.46	3442.67
8	119.64	48.61	1540.09
9	78.36	−41.24	1656.65
10	105.05	16.01	4797.22
11	237.73	26.25	3582.56
12	222.36	−4.50	1480.92
13	130.14	45.96	1610.60
14	66.26	−43.64	2127.58
15	138.07	−22.62	1263.42
16	220.27	33.28	1625.57
17	226.62	31.75	2703.74
18	231.41	39.59	1847.37

Model NN₂ demonstrates good performance in interpolation within regions that are well represented by the training data. However, the prediction quality significantly deteriorates for inputs located far from the training data space. The results presented in Case 2 confirm that the sampling density of the training set is crucial for the effectiveness of the inverse model.

4. Discussion

The conducted research confirmed that the developed inverse model based on the NN₂ neural network is capable of effectively determining the initial release conditions (initial velocity, release angle, and altitude) using only the final data such as range, flight time, and final velocity. In both the deterministic scenario (Case 1) and the stochastic scenario (Case 2), the results were satisfactory, although the representativeness of the training data in the vicinity of a given test sample proved to be essential.

In Case 1, where the test data overlapped with the training data, the model exhibited very high prediction accuracy—the errors were minimal, and the largest observed discrepancy (14.47%) occurred in only one case and concerned the release angle. This confirms that the inverse neural network accurately reproduces known relationships when an appropriately selected training dataset is used.

In the Case 2 scenario, where final parameters were randomly generated within broad input intervals, the model's performance remained high but exhibited greater variability. The mean absolute error for range was approximately 124.6 m, with a relative error of 4.14%, which is acceptable given the large variability range (exceeding 7000 m). Flight time

and final velocity were also accurately reconstructed in most cases; however, larger errors occurred when test data fell outside the well-represented domain of the training set.

The analysis showed that the key factor influencing the quality of the inverse prediction is the distance of the input data from the training set. In interpolation scenarios (i.e., when test data fall within the range of the training set), the NN₂ model demonstrates high accuracy. In contrast, when extrapolation is required (i.e., when test samples are significantly distant from known data), the prediction accuracy deteriorates considerably, and the inverse mapping may become ambiguous or fail to yield a physically meaningful solution. Additionally, our results indicate that the release angle (θ_0) and flight time (t_k) are particularly sensitive parameters. This behavior cannot be explained by extrapolation alone but points to identifiability challenges of the inverse problem. In some regions, different initial conditions may yield nearly indistinguishable outputs, which makes θ_0 and t_k harder to predict with a deterministic network. Future work will address this issue by applying local sensitivity analysis (e.g., Jacobian conditioning) and exploring probabilistic models such as Mixture Density Networks.

It should be emphasized that the presented model is preliminary and subject to limitations. The analysis was carried out under simplified conditions without accounting for turbulence or wind profiles, which will be addressed in future studies using Dryden and von Kármán models. The applied neural networks require large datasets and are prone to overfitting; cross-validation and regularization were employed here, and further work will expand the dataset to include cases with atmospheric disturbances. The current results are based on numerical simulations, and experimental validation (e.g., UAV-based capsule drops) is planned as the next stage of method development.

An advantage of the proposed approach is the use of a relatively small training dataset and low computational requirements, which makes the model well-suited for practical engineering applications, including real-time implementation.

5. Conclusions

In this study, a two-step neural network-based inverse modeling approach was developed and validated for estimating the initial release conditions of an airdropped payload (initial velocity, release angle, and altitude) using only final trajectory data such as range, flight time, and final velocity. The model achieved the primary research objective of providing an accurate and efficient tool for predicting drop parameters. The results, characterized by low relative errors in most cases, demonstrated that the NN₂ architecture can effectively reconstruct the underlying relationships when the training dataset sufficiently represents the test domain. The analysis further revealed that the release angle (θ_0) and flight time (t_k) are the most sensitive parameters, which reflects the intrinsic identifiability challenges of the inverse problem.

The proposed procedure offers significant potential for practical engineering use. Its relatively small training dataset requirements and low computational cost make it suitable for integration into real-time or near real-time mission planning systems. The methodology may enhance the precision of aerial delivery operations, with applications in both military contexts (e.g., JPADSs) and civilian missions (e.g., humanitarian aid, rescue equipment drops, or sensor payload deployment). Importantly, the approach provides a data-driven complement to classical numerical trajectory reconstruction methods.

The present work has limitations, as it was carried out under simplified conditions without accounting for atmospheric disturbances such as turbulence and wind. Future research will therefore focus on expanding the dataset to include a wider range of environmental conditions, incorporating turbulence models (Dryden and von Kármán), and applying hybrid strategies that combine neural networks with optimization tech-

niques. Additional studies will investigate probabilistic approaches such as Mixture Density Networks to address multi-modality in the inverse mapping. Finally, experimental validation (e.g., UAV-based capsule drops) is planned to confirm the model's performance in real-world conditions and to explore integration with autonomous navigation and control systems.

Supplementary Materials: The following supporting information can be downloaded at: <https://www.mdpi.com/article/10.3390/app151910422/s1>, Table S1: Data for Capsule Free Flight Simulation (learning set); Table S2: Data for Capsule Free Flight Simulation (testing set).

Author Contributions: Conceptualization, B.P.-S. and M.G.; methodology, B.P.-S. and M.G.; software, B.P.-S.; validation, B.P.-S. and M.G.; formal analysis, M.G.; investigation, M.G.; resources, B.P.-S. and M.G.; data curation, B.P.-S.; writing—original draft preparation, B.P.-S. and M.G.; writing—review and editing, B.P.-S. and M.G.; visualization, B.P.-S.; supervision, B.P.-S. and M.G.; project administration, B.P.-S.; funding acquisition, B.P.-S. and M.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are openly available in zenodo.org at Data for Capsule Free Flight Simulation, 125 training samples and 27 testing samples, DOI: <https://doi.org/10.5281/zenodo.16789234> (accessed on 28 August 2025).

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AGU	Autonomous Guidance Unit;
APER-DDQN	Adaptive Priority Experience Replay Deep Double Q-Network;
BN	Bayesian Network;
BPNN	Backpropagation Neural Network;
CARP	Calculated Aerial Release Point;
DDQN	Deep Double Q-Network;
GA	Genetic Algorithm;
GLONASS	Globalnaya Navigatsionnaya Sputnikovaya Sistema (Global Navigation Satellite System);
GNSS	Global Navigation Satellite System;
GPS	Global Positioning System;
INS	Inertial Navigation System;
JPADS	Joint Precision Airdrop System;
KE	Kane's Equation (model equation mentioned);
LIDAR	Light Detection and Ranging,
MLP	Multilayer Perceptron;
MPE	Mean Percent Error;
MSE	Mean Squared Error;
NN ₁	Neural Network (direct analysis)
NN ₂	Neural Network (inverse analysis)
NNs	Neural Networks;
PADS	Precision Airdrop Systems;
PDT	Parent-Divorcing Technique;

PER	Prioritized Experience Replay;
RFI	Radio Frequency Interference;
SLAM	Simultaneous Localization and Mapping;
STPA-BN	System-Theoretic Process Analysis-Bayesian Network;
UAF	Universal Activation Function;
UAV	Unmanned Aerial Vehicle;
WSHA	Whale-Swarm Hybrid Algorithm;
List of Symbols	
α	Angle of attack
α_t	Nutation angle
C_{aN}	Coefficient of the aerodynamic normal force
C_{aNr}	Coefficient of the aerodynamic damping force
C_{aX}	Coefficient of the aerodynamic axial force
C_{aX0}	Zero pitch coefficient
$C_{aX\alpha 2}$	Pitch drag coefficient
C_m	Coefficient of the aerodynamic tilting moment
C_q	Coefficient of the damping tilting moment
d	Diameter of the capsule body
F_{aX}	Axial aerodynamic force
F_{aN}	Normal force
$F_h(\cdot)$	Activation function of the hidden layer
$F_{out}(\cdot)$	Activation function of the output layer
F_x	Sum of all external forces along x body axes
F_x	Resultant force along x body axis
F_z	Sum of all external forces along z body axes
F_z	Resultant force along z body axis
g	Acceleration of gravity
H	Number of neurons in the hidden layer
h_0	Initial height
I	Moment of inertia matrix
I_y	Moment of inertia about the pitch axis
m	Capsule mass
M	Total pitching moment acting on the capsule
M_c	Sum of all external moments, expressed in the capsule body frame
Ω	Angular velocity vector of the body frame with respect to the inertial frame, also expressed in body coordinates
p, P	Pairs of data
ρ	Air density
Q	Component of the angular velocity vector of the capsule body
r_k	Range
S_b	Characteristic surface (cross-sectional area of the capsule)
θ_0	Initial angle of pitch / angle of release
t_k	Flight time
U	Component of the velocity vector of the capsule in relation to the air in the boundary system $Sxyz$ (along x -axis)
V_0	Initial velocity
V_c	Velocity of the capsule, expressed in body coordinates
V_k	Impact velocity

W	Component of the velocity vector of the capsule in relation to the air in the boundary system S_{xyz} (along z-axis)
$\mathbf{w}, \mathbf{b}$	Set of neural network parameters (weights and biases)
$\mathbf{X}$	Input vector (e.g., control parameters, material parameters)
x_g	x-coordinate of the initial point
x_k	x-coordinate of the end point (cargo drop)
$\mathbf{Y}$	Output vector (e.g., observed system response, simulation result)
$y_i^{(p)}$	Reference data
z_g	z-coordinate of the initial point
z_k	z-coordinate of the end point (cargo drop)
$t_i^{(p)}$	Computed values

References

- Wu, Q.; Wu, H.; Jiang, Z.; Tan, L.; Yang, Y.; Yan, S. Multi-objective optimization and driving mechanism design for controllable wings of underwater gliders. *Ocean Eng.* **2023**, *286*, 115534. [CrossRef]
- Li, G.; Cao, Y.; Wang, M. Modeling and Analysis of a Generic Internal Cargo Airdrop System for a Tandem Helicopter. *Appl. Sci.* **2021**, *11*, 5109. [CrossRef]
- Xu, B.; Chen, J. Review of modeling and control during transport airdrop process. *Int. J. Adv. Robot. Syst.* **2016**, *13*, 1729881416678142. [CrossRef]
- Basnet, S.; Toroody, A.B.; Chahal, M.; Lahtinen, J.; Bolbot, V.; Banda, O.A.V. Risk analysis methodology using STPA-based Bayesian network-applied to remote pilotage operation. *Ocean Eng.* **2023**, *270*, 113569. [CrossRef]
- Xu, J.; Tian, W.; Kan, L.; Chen, Y. Safety Assessment of Transport Aircraft Heavy Equipment Airdrop: An Improved STPA-BN Mechanism. *IEEE Access* **2022**, *10*, 87522–87534. [CrossRef]
- Kane, R.; Dicken, M.T.; Buehler, R.C. *A Homing Parachute System Technical Report*; Sandia Corporation: Albuquerque, NM, USA, 1961.
- Wegereef, J.W. Precision airdrop system. *Aircr. Eng. Aerosp. Technol. Int. J.* **2007**, *79*, 12–17. [CrossRef]
- Jóźwiak, A.; Kurzawiński, S. The concept of using the Joint Precision Airdrop System in the process of supply in combat actions. *Mil. Logist. Syst.* **2019**, *51*, 27–42. [CrossRef]
- Mathisen, S.H.; Grindheim, V.; Johansen, T.A. Approach Methods for Autonomous Precision Aerial Drop from a Small Unmanned Aerial Vehicle. *IFAC-PapersOnLine* **2017**, *50*, 3566–3573. [CrossRef]
- Lu, J.; Zou, T.; Jiang, X. A Neural Network Based Approach to Inverse Kinematics Problem for General Six-Axis Robots. *Sensors* **2022**, *22*, 8909. [CrossRef]
- Dever, C.; Dyer, T.; Hamilton, L.; Lommel, P.; Mohiuddin, S.; Reiter, A.; Singh, N.; Truax, R.; Wholey, L.; Bergeron, K.; et al. Guided-Airdrop Vision-Based Navigation. In Proceedings of the 24th AIAA Aerodynamic Decelerator Systems Technology Conference, Denver, CO, USA, 5–9 June 2017. [CrossRef]
- Felux, M.; Fol, P.; Fiquet, B.; Waltert, M.; Olive, X. Impacts of Global Navigation Satellite System Jamming on Aviation. *Navigation* **2023**, *71*, navi.657. [CrossRef]
- Mateos-Ramirez, P.; Gomez-Avila, J.; Villaseñor, C.; Arana-Daniel, N. Visual Odometry in GPS-Denied Zones for Fixed-Wing Unmanned Aerial Vehicle with Reduced Accumulative Error Based on Satellite Imagery. *Appl. Sci.* **2024**, *14*, 7420. [CrossRef]
- Li, D.; Zhang, F.; Feng, J.; Wang, Z.; Fan, J.; Li, Y.; Li, J.; Yang, T. LD-SLAM: A Robust and Accurate GNSS-Aided Multi-Map Method for Long-Distance Visual SLAM. *Remote Sens.* **2023**, *15*, 4442. [CrossRef]
- Pramod, A.; Shankaranarayanan, H.; Raj, A.A. A Precision Airdrop System for Cargo Loads Delivery Applications. In Proceedings of the International Conference on System, Computation, Automation and Networking (ICSCAN), Puducherry, India, 30–31 July 2021. [CrossRef]
- Cheng, W.; Yang, C.; Ke, P. Landing Reliability Assessment of Airdrop System Based on Vine-Bayesian Network. *Int. J. Aerosp. Eng.* **2023**, *23*, 1773841. [CrossRef]
- Siwek, M.; Baranowski, L.; Ładyżyńska-Kozdraś, E. The Application and Optimisation of a Neural Network PID Controller for Trajectory Tracking Using UAVs. *Sensors* **2024**, *24*, 8072. [CrossRef] [PubMed]
- Yong, L.; Qidan, Z.; Ahsan, E. Quadcopter Trajectory Tracking Based on Model Predictive Path Integral Control and Neural Network. *Drones* **2025**, *9*, 9. [CrossRef]
- Hertz, J.; Krogh, A.; Palmer, R. *Introduction to the Theory of Neural Computation*, 2nd ed.; CRC Press: Boca Raton, FL, USA, 1995.
- Haykin, S. *Neural Networks-A Comprehensive Foundation*; Prentice Hall: New York, NY, USA, 1999.
- Truong Pham, D.; Xing, L. *Neural Networks for Identification, Prediction and Control*, 1st ed.; Springer: London, UK, 2012.
- ParaZero's DropAir: Precision Airdrop for Contested Environments. Available online: <https://www.autonomyglobal.co/parazeros-dropair-precision-airdrop-for-contested-environments/> (accessed on 10 August 2025).

23. Xu, R.; Yu, G. Research on intelligent control technology for enhancing precision airdrop system autonomy. In Proceedings of the International Conference on Machine Learning and Computer Application (ICMLCA), Hangzhou, China, 27–29 October 2023.
24. Wang, Y.; Yang, C.; Yang, H. Neural network-based simulation and prediction of precise airdrop trajectory planning. *Aerosp. Sci. Technol.* **2022**, *120*, 107302. [\[CrossRef\]](#)
25. Ouyang, Y.; Wang, X.; Hu, R.; Xu, H. APER-DDQN: UAV Precise Airdrop Method Based on Deep Reinforcement Learning. *IEEE Access* **2022**, *10*, 50878–50891. [\[CrossRef\]](#)
26. Li, K.; Zhang, K.; Zhang, Z.; Liu, Z.; Hua, S.; He, J. A UAV Maneuver Decision-Making Algorithm for Autonomous Airdrop Based on Deep Reinforcement Learning. *Sensors* **2021**, *21*, 2233. [\[CrossRef\]](#)
27. Wei, Z.; Shao, Z. Precision landing of autonomous parafoil system via deep reinforcement learning. In Proceedings of the IEEE Aerospace Conference, Big Sky, MT, USA, 2–9 March 2024. [\[CrossRef\]](#)
28. Qi, C.; Min, Z.; Yanhua, J.; Min, Y. Multi-Objective Cooperative Paths Planning for Multiple Parafoils System Using a Genetic Algorithm. *J. Aerosp. Technol. Manag.* **2019**, *11*, e0419. [\[CrossRef\]](#)
29. Tao, J.; Sun, Q.L.; Zhu, E.L.; Chen, Z.Q.; He, Y.P. Genetic algorithm based homing trajectory planning of parafoil system with constraints. *J. Cent. South Univ. Technol.* **2017**, *48*, 404–410. [\[CrossRef\]](#)
30. Zhang, A.; Xu, H.; Bi, W.; Xu, S. Adaptive mutant particle swarm optimization based precise cargo airdrop of unmanned aerial vehicles. *Appl. Soft Comput.* **2022**, *130*, 109657. [\[CrossRef\]](#)
31. Wu, Y.; Wei, Z.; Liu, H.; Qi, J.; Su, X.; Yang, J.; Wu, Q. Advanced UAV Material Transportation and Precision Delivery Utilizing the Whale-Swarm Hybrid Algorithm (WSHA) and APCR-YOLOv8 Model. *Appl. Sci.* **2024**, *14*, 6621. [\[CrossRef\]](#)
32. Aster, R.; Borchers, B.; Thurber, C. *Parameter Estimation and Inverse Problems*; Elsevier: Amsterdam, The Netherlands, 2003.
33. Bolzon, G.; Maier, G.; Panico, M. Material Model Calibration by Indentation, Imprint Mapping and Inverse Analysis. *Int. J. Solids Struct.* **2004**, *41*, 2957–2975. [\[CrossRef\]](#)
34. Potrzeszcz-Sut, B.; Dudzik, A. The Application of a Hybrid Method for the Identification of Elastic–Plastic Material Parameters. *Materials* **2022**, *15*, 4139. [\[CrossRef\]](#) [\[PubMed\]](#)
35. Potrzeszcz-Sut, B.; Pabisek, E. ANN Constitutive Material Model in the Shakedown Analysis of an Aluminum Structure. *Comput. Assist. Methods Eng. Sci.* **2017**, *21*, 49–58. [\[CrossRef\]](#)
36. Etkin, B. *Dynamics of Atmosphere Flight*; John Wiley & Sons, Inc.: New York, NY, USA, 1972.
37. Blakelock, J.H. *Automatic Control of Aircraft and Missiles*; John Wiley & Sons, Inc.: New York, NY, USA, 1991.
38. Kowaleczko, G.; Klemba, T.; Pietraszek, M. Stability of a Bomb with a Wind-Stabilised-Seeker. *Probl. Mechatron. Armament Aviat. Saf.* **2022**, *13*, 43–66. [\[CrossRef\]](#)
39. Baranowski, L.; Frant, M. Calculation of aerodynamic characteristics of flying objects using Prodas and Fluent software. *Mechanik* **2017**, *7*, 591–593. [\[CrossRef\]](#)
40. Grzyb, M.; Koruba, Z. Comparative Analysis of the Guided Bomb Flight Control System for Different Initial Conditions. *Meas. Autom. Robot.* **2024**, *3*, 41–52. [\[CrossRef\]](#)
41. Humennyi, A.; Oleynick, S.; Malashta, P.; Pidlisnyi, O.; Aleinikov, V. Construction of a ballistic model of the motion of uncontrolled cargo during its autonomous high-precision drop from a fixed-wing unmanned aerial vehicle. *Appl. Mech.* **2024**, *5*, 25–33. [\[CrossRef\]](#)
42. Yuen, B.; Tu Hoang, M.; Dong, X.; Lu, T. Universal Activation Function for Machine Learning. *Sci. Rep.* **2021**, *11*, 18757. [\[CrossRef\]](#) [\[PubMed\]](#)
43. Cybenko, G. Approximation by superpositions of a sigmoidal function. *Math. Control Signals Syst.* **1989**, *2*, 303–314. [\[CrossRef\]](#)
44. Hornik, K.; Stinchcombe, M.; White, H. Multilayer feedforward networks are universal approximators. *Neural Netw.* **1989**, *2*, 359–366. [\[CrossRef\]](#)
45. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
46. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [\[CrossRef\]](#) [\[PubMed\]](#)
47. Feller, W. *An Introduction to Probability Theory and Its Applications*; Wiley: New York, NY, USA, 1968; Volume 1, 2.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.