Journal of High School Science



Solving the inverse problem in baseball trajectory

Kim K.B[1], Lee S[2]

Submitted: June 21, 2026, Revised: version 1, July 28, 2026, version 2, September 1, 2026
Accepted: September 17, 2026

Abstract
Inferring the release conditions required to produce a desired baseball trajectory is an ill-posed inverse problem because different combinations of release velocity, spin rate, spin efficiency, spin axis, and release direction can generate similar ball flights. We trained multilayer perceptron (MLP), convolutional neural network (CNN), and gated recurrent neural network (GRU-RNN) models on MLB Statcast data from the 2022–2024 seasons to infer plausible release conditions from scalar, complete-trajectory, and partial-trajectory information. An ensemble of these architectures achieved ~ one-inch forward trajectory error across eight pitch types. We then characterized physical non-uniqueness by sampling 10,000 plausible release vectors around representative four-seam fastball, curveball, sweeper, and splitter conditions and identifying those reproducing target trajectories within multiple tolerances. Admissible solution spaces differed substantially among the four representative pitches, with the splitter exhibiting the broadest region, indicating that release-condition identifiability may be pitch-dependent. Identifiability was also parameter-dependent: full-trajectory information substantially narrowed admissible release directions, whereas release velocity, spin rate, and spin efficiency remained compensatory, including a negative spin-rate–spin-efficiency association (r = -0.44 to -0.74). The pitch-wise ordering of physical non-identifiability was associated with ensemble uncertainty, suggesting that model disagreement may reflect underlying inverse ambiguity, although this relationship was evaluated in only four representative cases. These findings recast non-uniqueness from solely a limitation into a potential pitch-design space in which alternative release strategies may produce a desired trajectory. Thus, the study distinguishes model-dependent predictive accuracy from the inherent physical identifiability of a pitch's release conditions, providing a basis for individualized coaching, pitch development, and repertoire evaluation.

Keywords
Inverse problems, Baseball aerodynamics, Pitch design, Machine Learning, Neural Networks, Sports analytics, Trajectory prediction, Statcast, Parameter identifiability, Uncertainty quantification

[1]Corresponding author: Kyu Beom Kim, Valor International Scholars, 1, Seonggong-gil, Anseong-si, Gyeonggi-do, 17509, Republic of Korea. kimkyubeom6@gmail.com
[2]Subok Lee, Winschool, 405, Cheongho-ro, Suseong-gu, Daegu, 42093, Republic of Korea.

https://doi.org/10.64336/001c.171799

1. Introduction

The trajectory of a baseball after it has been released by a pitcher is determined by the simultaneous influence of three forces: (a) the downward gravitational force; (b) the aerodynamic drag force which opposes the motion of the ball as it moves through the air; and (c) the Magnus force which represents lift due to interaction of the spinning ball and the surrounding fluid boundary layer (1). These forces together produce a curved path as the ball travels the 60.5 feet from the pitcher's rubber to home plate. Given complete knowledge of the ball's state at release, its trajectory can be calculated by numerically solving the ordinary differential equations governing its motion—the forward problem. In professional baseball, ball-tracking systems installed in stadiums provide the measurements needed to characterize this trajectory (2).

The inverse problem is considerably more difficult than the forward problem (3): what release conditions must a pitcher impart to the ball to produce a desired trajectory? This question is central to modern pitch design, which seeks to optimize pitch movement to deceive batters and exploit their individual weaknesses (4). For example, a pitching coach may seek to design a fastball with maximum perceived rise, a sweeper with 20 inches of lateral break, or a changeup that resembles a fastball until the final 15 feet of flight. Achieving such objectives requires translating the desired ball movement into physical release parameters, including release velocity, spin characteristics, seam orientation, and release direction.

The inverse problem is complicated by three interconnected challenges. First, it is nonlinear because the Magnus force depends on both the ball's velocity and spin, preventing a straightforward analytical solution for an arbitrary target trajectory (5). Second, multiple combinations of release conditions may produce the same or very similar trajectories, making the inverse solution potentially non-unique (6). Finally, the problem is high-dimensional: the pitcher determines release velocity, spin rate, spin-axis orientation, and release direction, with the latter two each described by two angular components. Because these parameters interact through the ball's aerodynamics, small changes in one parameter may be compensated by changes in others.

Neural networks are well suited to problems involving high dimensionality and nonlinearity. Once trained, they can generate predictions in microseconds and learn the manifold of physically plausible solutions directly from data, without requiring a differentiable forward model or careful initialization for each prediction. Three principal neural-network architectures can be applied to the inverse-trajectory problem, each representing and processing trajectory information in a different way. Multilayer Perceptron (MLP) (7) maps a fixed vector of scalar trajectory descriptors directly to the release parameter vector. MLPs are suitable when the trajectory data are summarized in a compact form as is typical in broadcasting and scouting. Convolutional Neural Network (CNN) (8) treats the discretized spatial trajectory of the ball as a one-dimensional signal and applies learned local filters to extract movement features at several scales. CNNs are good choices when full ball-

flight sequences are available and spatial structure in the trajectory shape carry additional predictive information beyond what scalar summaries can provide. Recurrent Neural Networks (RNN) (9) with Gated Recurrent Units (GRU) (10) processes the trajectory as a temporal sequence of ball positions and can condition prediction on partial observation, thus allowing inversion from incomplete trajectory, as would happen in live coaching or biomechanical feedback application where the full flight was not complete. An overview of our model is shown in Figure 1.

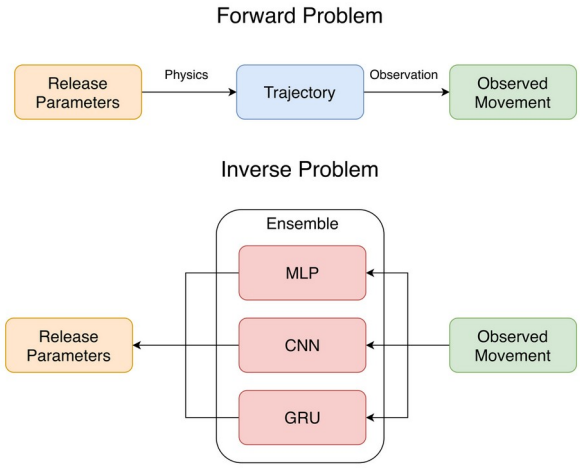


Figure 1. Overview of the forward and inverse problem and the proposed method. The forward problem maps release parameters to an observed trajectory via the aerodynamic equations of motion (top). The inverse problem infers a plausible release-parameter vector consistent with a desired trajectory using an ensemble of MLP, CNN, and GRU models trained on MLB Statcast data (bottom).

This paper presents a rigorous empirical comparison of these three architectures for the baseball inverse-trajectory problem. The models were trained on MLB Statcast data (11, 12) and evaluated using a hold-out test set of pitches from the 2024 regular season. Inversion accuracy, forward-trajectory error, computational cost, and performance under partial trajectory availability were evaluated across eight major pitch types. To directly characterize non-uniqueness, rather than infer it solely from disagreement among architectures, we additionally sampled the admissible release-parameter space for four representative pitch types using the same forward aerodynamic model used to generate the training targets. This analysis was performed across multiple trajectory-matching tolerances and under two information conditions: scalar trajectory descriptors and the full ball-flight sequence. Finally, we developed an ensemble model combining all three architectures to exploit their complementary strengths.

2. Background

2.1 Coordinate system and ball properties

We work in a right-handed Cartesian coordinate system with origin at the center of home plate. The x-axis points horizontally toward the third-

base side as viewed from the catcher's perspective, the y-axis points from home plate toward the pitcher so that a pitched ball travels in the negative y direction, and the z-axis points vertically upward. A standard MLB baseball has mass $m = 141.7 \pm 2.0\ g$, radius $r = 36.8 \pm 0.4\ mm$, and cross-sectional area $A = \pi r^2 \approx 4.26 \times 10^{-3}\ m^2$. Its surface consists of two figure-eight leather panels stitched together by 108 red cotton stitches, and the seam geometry plays a critical aerodynamic role through its effect on boundary-layer separation (1).

The state of the ball at any instant t is described by its position vector $x(t) = (x, y, z)$ and velocity vector $v(t) = (v_x, v_y, v_z)$. The angular velocity is represented as $\omega = \omega \cdot \hat{\omega}$, where ω (rad/s) is the scalar spin rate and $\hat{\omega}$ is the unit vector along the spin axis. The spin rate is frequently expressed in Revolutions Per Minute: 1 RPM = $\pi/30$ rad/s. A typical MLB fastball has $\omega \approx 2{,}200\text{-}2{,}600$ RPM, breaking balls range from 2,400–2,800 RPM, and changeups and splitters are lower at 1,400–1,800 RPM.

2.2 Forces acting on the ball

Three forces govern the ball's flight through still air (1). In practice, stadium wind can contribute an additional ~ constant wind force, but we treat the air as quiescent and note that wind corrections are straightforward additive modifications. Gravity acts purely in the negative z-direction with magnitude mg, where $g = 9.807\ m/s^2$ at sea level. Its contribution to the acceleration vector is $a_g = (0, 0, -g)$ (eq.1).

For a pitch traveling at 40.2 m/s, with a flight time of ~ 0.43 seconds, gravity alone would deflect the ball ~ 0.91 meters, downward from its release direction. This is by far the largest single perturbation to the straight-line path, and it is constant and perfectly predictable, allowing aerodynamic effects to be isolated as deviations from the gravity-only trajectory. As the ball moves through air, it experiences a drag force opposing its velocity which can be described by $F_D = -\frac{1}{2}\ \rho\ C_D(v, \omega)\ A\ |v|^2\ \hat{v}$ (eq.2), where ρ is the air density (kg/m^3), C_D is the dimensionless drag coefficient, and $\hat{v} = v / |v|$ is the unit velocity vector. Air density varies significantly across MLB stadiums. At sea level, as at Fenway Park, $\rho \approx 1.225\ kg/m^3$, while at Coors Field in Denver, which sits at an elevation of 1,580 m, $\rho \approx 1.038\ kg/m^3$, a 15% reduction that substantially affects both drag and Magnus force.

The drag coefficient C_D for a baseball is not a simple constant. It depends on speed through the drag crisis, where the boundary layer shifts from laminar to turbulent, causing a sharp drop in C_D. For a smooth sphere this transition is affected by the seams on a baseball tripping the boundary layer, placing the transition in the range 60–80 mph. The empirical drag coefficient for a rotating baseball follows ~ $C_D \approx 0.33 + 0.06 \cdot \exp(-0.044\ v_{mph})$ for $v_{mph} \in [60, 105]$ (eq.3), yielding $C_D \approx 0.36$ at 95 mph and $C_D \approx 0.40$ at 70 mph. The seam orientation also influences drag. Four-seam orientation, in which the seams rotate in a plane perpendicular to the velocity, produces slightly lower drag than in a two-seam orientation, in

which the seams rotate in a tilted plane, an effect of the order 5% in C_D.

2.2.1 *Magnus force*

The Magnus force is the aerodynamic lift generated when a spinning ball moves through a fluid (13). The rotation of the ball entrains the surrounding air, creating higher flow velocity on one side and lower on the other. By Bernoulli's principle, pressure is lower on the high-velocity side, generating a net force. For a baseball, the Magnus force takes the form $F_M = ½\ \rho\ C_L(S)\ A\ |v|^2\ (\hat{\omega} \times \hat{v})$ (eq.4), where C_L is the dimensionless lift coefficient and $S = r\ \omega\ /\ |v|$ is the dimensionless spin parameter. The cross product $\hat{\omega} \times \hat{v}$ gives the direction of the Magnus force. It is perpendicular to both the spin axis and the velocity, so a ball with pure backspin, where $\hat{\omega}$ points horizontally opposite to the direction of travel for a right-hander, receives an upward Magnus push, while topspin produces downward Magnus, and sidespin produces lateral Magnus. The lift coefficient C_L has been measured extensively in wind-tunnel experiments. A widely used empirical fit is $C_L \approx 1\ /\ (2.32 + 0.4\ /\ S) \approx 0.62$ S for small S (eq.5).

Typical values are $C_L \approx 0.15 \sim 0.30$ for MLB pitches, producing Magnus forces of 0.4 ~ 1.2 N, comparable to a substantial fraction of the gravitational force of 1.39 N. An important subtlety is that not all spin contributes to Magnus force. Gyrospin, in which $\hat{\omega}$ is aligned with $\hat{v}$ such that the ball spins like a bullet, produces zero Magnus force regardless of magnitude. The effective Magnus contribution depends only on the component of spin perpendicular to the velocity, quantified by the spin efficiency $\eta = |\hat{\omega} \times \hat{v}|$, which equals 1.0 for purely transverse spin and 0.0 for pure gyrospin.

2.3 Full equation of motion

Combining the three force contributions, Newton's second law for the ball gives $m\ \ddot{x} = F_D + F_M + m\ a_g$ (eq.6). Expanding into components gives,

$$\ddot{x} = -½\left(\frac{\rho}{m}\right) C_D\ A\ |v|\ vx + ½\left(\frac{\rho}{m}\right) C_L\ A\ |v|^2 (\hat{\omega} \times \hat{v})_x \quad \text{(eq.7)}$$

$$\ddot{y} = -½\left(\frac{\rho}{m}\right) C_D\ A\ |v|\ vy + ½\left(\frac{\rho}{m}\right) C_L\ A\ |v|^2 (\hat{\omega} \times \hat{v})_y \quad \text{(eq.8)}$$

$$\ddot{z} = -g - ½\left(\frac{\rho}{m}\right) C_D\ A\ |v|\ vz + ½\left(\frac{\rho}{m}\right) C_L\ A\ |v|^2 (\hat{\omega} \times \hat{v})_z \quad \text{(eq.9)}$$

These three equations describe how the ball accelerates in each direction at every instant. Because the drag and Magnus forces both depend on the ball's current speed and direction, the equations are coupled, meaning we cannot solve one without knowing the others (13). There is no algebraic shortcut to find the full trajectory. Instead, we solve them numerically by breaking the flight into tiny time steps. At each step, we use the current velocity to compute the forces, update the velocity, and advance the position. We use a method called

fourth-order Runge-Kutta, a standard technique for solving this kind of problem accurately and efficiently (14). For a typical pitch at 90 mph, a step size of 2 ms provides trajectory accuracy better than 1 mm, far beyond measurement precision, at the cost of ~ 250 integration steps per pitch.

The ball's speed decreases monotonically during flight due to drag. For a 95 mph fastball, the speed at home plate is typically 87–88 mph, a loss of ~ 8–9%. The Magnus force direction rotates slightly as the velocity vector tilts downward under gravity and drag, but the spin axis $\hat{\omega}$ remains essentially fixed in inertial space throughout the flight due to gyroscopic rigidity, hence the Magnus force direction is ~ constant over the pitch.

2.4 Movement parameterization

From the full trajectory, two scalar quantities are extracted that are the primary targets of pitch design and the primary observable inputs to the inverse problem. First, the Induced Vertical Break (IVB) is the vertical deviation of the ball's true path from a hypothetical trajectory with zero aerodynamics, measured at home plate as $IVB = z_{actual}(t_{plate}) - z_{gravity}(t_{plate})$ (eq.10), where $z_{gravity}(t) = z_0 + vz_0\, t - \tfrac{1}{2}\, g\, t^2$ is the gravity-only vertical position. IVB isolates the net vertical Magnus contribution: a positive IVB means the ball 'rides up' because backspin lift exceeds the gravity perturbation, while a negative IVB indicates a downward aerodynamic contribution relative to the gravity-only trajectory, as can occur with topspin. Horizontal Break (HB) is the lateral deviation from a gravity-and-drag trajectory in which the Magnus force is set to zero as $HB = x_{actual}(t_{plate}) - x_{nodrift}(t_{plate})$ (eq.11). HB quantifies the side-to-side movement contributed purely by the horizontal component of the Magnus force. Combined with IVB, the pair of HB and IVB values provides a distinct two-dimensional movement fingerprint that characterizes the aerodynamic signature of any pitch type. The combination of these two scalar descriptors with release speed, plate-crossing speed, and plate location constitutes the trajectory descriptor vector used as the input to the inverse neural-network models.

2.5 Problem formulation

The release parameter vector θ encodes the physical state of the ball at the moment it leaves the pitcher's fingertips as $\theta = (v_{rel},\ \omega,\ \eta,\ \varphi_{spin},\ \varphi_{rel},\ \theta_{rel})^T \in \mathbb{R}^6$ (eq.12). The first component is vrel, the release speed in mph. The second is ω, the total spin rate in RPM. The third is $\eta \in [0,\ 1]$, which is the spin efficiency representing the fraction of spin that actively produces Magnus force. The fourth is $\varphi_{spin} \in [0°,\ 360°)$, the spin-axis clock-face angle, where 0° corresponds to pure topspin, 90° to arm-side sidespin for a right-hander, 180° to pure backspin, and 270° to glove-side sidespin. Together, spin efficiency and spin-axis angle specify the magnitude and direction of the spin component perpendicular to the ball's velocity: spin efficiency determines the fraction of total spin contributing to the Magnus force, while the spin-axis angle determines the direction of that contribution. The fifth is $\varphi_{rel} \in [-10°,\ 10°]$, the horizontal release angle measured from the pitcher-plate line. The sixth is $\theta_{rel} \in [-5°,\ 5°]$, the vertical release angle. The release position with coordinates $x_0,\ y_0,\ z_0$ is treated as a nuisance parameter,

standardized across pitchers by centering relative to the pitcher's historical average. The trajectory descriptor vector τ encodes the observable outcome of the pitch flight as $\tau = (IVB, HB, v_{plate}, x_{plate}, z_{plate})^T \in \mathbb{R}^5$ (eq.13). For the CNN and RNN models, we additionally use the full discretized trajectory sequence $X = \{x(t_0), x(t_1), \ldots, x(t_N)\}$, where positions are sampled at $N = 50$ equal-interval time steps from release to home plate. Each sample point is a three-dimensional vector (x, y, z), giving a sequence of shape $(50, 3)$. The scalar descriptor τ is always computable from X by evaluation of Eqs. (10) - (11) and measurement of plate-crossing position and speed; the reverse is not true, making X a strictly richer input representation.

The forward map $F: \mathbb{R}^6 \rightarrow \mathbb{R}^5$ takes a release parameter vector to a scalar trajectory descriptor. For the sequence-based models, the extended forward map $G: \mathbb{R}^6 \rightarrow \mathbb{R}^{50\times 3}$ maps to the full discretized trajectory. The inverse problem is: Given τ^*, find $\hat{\theta}$ such that $||F(\hat{\theta}) - \tau^*||$ is minimized (14). The inverse problem is potentially non-unique because the six-dimensional release-parameter space is mapped to a five-dimensional trajectory-descriptor space, allowing multiple release conditions to produce similar trajectory descriptors, as examined empirically in Section 4.3. We address ill-posedness through the ensemble architecture described in the Experiments section, which provides uncertainty estimates that flag ambiguous targets.

3. Methods

3.1 Multilayer perceptron

The MLP is the natural architecture when the trajectory is summarized as a compact scalar feature vector. Its inductive bias is minimal: it makes no structural assumptions about the trajectory representation and learns an arbitrary continuous mapping from descriptors to release parameters through a hierarchy of learned affine transformations and nonlinearities.

The MLP architecture consists of an input layer receiving the 5-dimensional normalized descriptor, followed by seven fully connected hidden layers with widths $[256, 512, 1024, 1024, 512, 256, 128]$, each succeeded by layer normalization and the GELU (16) activation function. Residual connections, also called skip connections, bypass every two hidden layers, improving gradient flow through the 7-layer depth. (15) The output layer has 6 linear neurons corresponding to the six release parameters. The loss function is a weighted mean squared error over the six output dimensions as $L_{MLP} = \left(\frac{1}{N}\right)\Sigma_n \Sigma_k w_k (\theta_k^n - \hat{\theta}_k^n)^2$ (eq.15). We chose weights $w = (1.0, 0.8, 0.5, 0.5, 0.3, 0.3)$ that emphasize the first 2 parameters which represent speed and spin rate as they are the most important ones among all 6 parameters. We trained our model using AdamW optimizer (17) with initial learning rate equal to $3\text{x}10^{-4}$, cosine annealing schedule for learning rate reduction during training, weight decay set at $1\text{x}10^{-4}$, batch size of 4096 samples. Training was performed for 100 epochs.

3.2 Convolutional Neural Network

CNN takes the full discrete (discretized) trajectory $X \in \mathbb{R}^{50\times 3}$, as a one-dimensional signal with three channels, for each of the x, y, and z coordinates. Its inductive bias is local spatial equivariance, meaning that the CNN learns to identify shape features of the trajectory (e.g., changes in curvature, acceleration profiles, etc.) at multiple scales, without regard to their position on the flight path. This is suitable for baseball trajectories since the most informative part of the trajectory regarding a specific type of pitch, namely the late break spanning the final 10-15 feet of flight, may begin anywhere within this region.

The CNN has an architecture based on a multi-scale feature extraction backbone. Early flight stage covered early flight with 3 convolutional layers with kernel size = 5 and widths = [64, 128, 128]. This captured coarse velocity and direction features across the entire length of the first 30% of the trajectory. Mid-flight stage covered mid-flight with 3 convolutional layers with kernel size = 3 and widths = [256, 256, 512]. This captured the onset of Magnus force and the development of horizontal break characteristics. Late flight stage covered late flight with 3 convolutional layers with kernel size = 3 and widths = [512, 512, 256]. This captured the late break and deceleration profile characteristics that differentiate pitches. After these three stages, average pooling and max pooling operations were applied across all spatial dimensions and combined by concatenating them together. This preserved both mean trajectory shape and peak-feature information from the input trajectory. Then the combined results passed through 3 fully connected layers with widths = [256, 128, 64] and GELU activation functions. This produced a 6-dimensional output identical in structure to the MLP. Skip connections were created between corresponding width stages. It was trained using the same MSE loss function as the MLP; however, it was trained for 80 epochs instead of the MLP due to convergence being faster when training on a more detailed representation of the input data.

3.3 Recurrent Neural Network

The RNN treats the trajectory as a temporal sequence and uses one observation from a position at a time. It also maintains a hidden state of the flight process. Thus, it can work with two different types of trajectories. First with complete trajectory sequences, and second with partial trajectories, to allow inversion from incomplete flight data. Therefore, the architecture is suitable for use in real-time applications where the full trajectory has not been generated.

We used Gated Recurrent Units (GRU) instead of using vanilla RNNs or LSTMs. GRUs employ reset and update gates to regulate the flow of information through the following gates.

$$z_t = \sigma(W_z\, x_t + U_z\, h_{t-1} + b_z) \quad [update\ gate] \quad \text{(eq.16)}$$

$$r_t = \sigma(W_r\, x_t + U_r\, h_{t-1} + b_r) \quad [reset\ gate] \quad \text{(eq.17)}$$

$$\tilde{h}_t = \tanh(W_h\, x_t + U_h\, (r_t \odot h_{t-1}) + b_h) \quad [candidate] \quad \text{(eq.18)}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \quad [new\ state] \quad \text{(eq.19)}$$

GRUs contain fewer parameters than LSTMs but produce similar results for sequential data of length $T = 50$ steps. Additionally, they provide stable training for regression problems. A bidirectional GRU stack was employed in this study. The input sequence first passed through an embedding layer that applied a linear transformation from a 3-dimensional space to a 64-dimensional vector at each time-step; allowing the network to learn a task specific feature representation of position. Following this, a bidirectional GRU stack of four layers were stacked with a hidden dimension of 256 units in each direction and therefore 512 units per layer. The sequence was processed in both the forward direction from release to plate and the backward direction from plate to release. Their outputs were then concatenated together. Subsequently, an attention pooling layer (18) applied a learned attention mechanism over the $T = 50$ hidden states. The network could thus weigh the most informative flight phases based upon the learned weights and not simply rely upon the last hidden state. Lastly, the same 3-layer MLP-head as the CNN produced a 6-dimensional output. As the network used a bi-directional architecture it was able to incorporate both early-flight velocity establishing information and late-flight braking profile information into its predictions.

For all partial-trajectory experiments, the backward GRU received a zero-padded version of the unobserved part of the trajectory, and a binary observation mask was concatenated to the embedding at each step to denote whether each position represented a valid measurement or padding. This masking allowed a single trained model to be capable of processing trajectories of varying lengths without requiring re-training.

3.4 Ensemble model

The three architectures calculated the inverse problem from complementary viewpoints. The MLP uses summary statistics as input; the CNN makes use of the spatial characteristics; and the RNN uses both the temporal order of data and can handle partially observed time series. Combining the predictions of these architectures may improve predictive robustness while allowing disagreement among ensemble members to provide an empirical measure of predictive uncertainty (19).

We trained five separate models of each architecture culminating in fifteen total models with varying random seed values, and we combined their outputs (predictions) by manually assigning weights to each model. We assigned a larger weight to the CNN for the IVB/HB outputs due to their dependence upon the full trajectory shape; assigned larger weights to the MLP for the release speed prediction since vplate is very informative for this type of output; and distributed weights equally among the various architectures for predicting spin axis as this output dimension is the most ambiguous. The uncertainty estimate of the combined ensemble is the standard deviation of the 15 individual predictions.

3.5 Experiments

3.5.1 *Dataset*

We retrieved all the regular season Major League Baseball pitches that were thrown during 2022-2024 using the pybaseball library. The library uses the Baseball Savant Statcast

search API. We removed all the pitches that had no tracking information available for Statcast because they were either partially blocked from view by a camera or could not be tracked at all. Also, we removed all the pitches that were identified as being very low confidence according to the HawkEye tracking system (20). There were 2,073,446 pitches left after those exclusions.

The trajectory descriptor vector τ; the Statcast clock face angle φspin of the spin axis; the release speed vrel; the spin rate ω; the spin efficiency η (which was calculated as the ratio of active spin to total spin based on the relationship between the movements and spins of Statcast); and finally, the release angles φrel and θrel derived from the first velocity vector components, were all extracted from each pitch.

For both our CNN and RNN models, we reconstructed the full 50-point trajectory sequence X by numerically integrating the equations of motion from the provided initial conditions forward in time. The Statcast-fitted constant accelerations were used to verify that the numerical integration produced accurate results. Therefore, we obtained a per-pitch trajectory sequence of size (50, 3) with subcentimeter level of accuracy compared to the measurement of the HawkEye system. Note that because this process for reconstructing trajectories is based on a CNN or RNN being trained to reverse-engineer the trajectory output of an internally developed forward aerodynamic model as opposed to a fully independent measurement of all flight position data; the simplifications in the internally developed model (e.g. how it treats the orientation of seams, the atmosphere, etc. as well as the effects due to the shift of the wake along with the seam), are therefore not represented within the training of either sequence-based model, and can lead the models to learn how to reverse-engineer what they were told about physics as opposed to all the complexities of actual flight.

Our dataset was divided into three sets temporally. The combination of the pitches from 2022 and 2023 totaled 1,791,204 and formed our training set. The combination of the pitches from April through July of 2024 totaling 183,441 formed our validation set for hyperparameter tuning and early stopping. Finally, the combination of the pitches from August through October of 2024 totaling 98,801 formed our test set.

Therefore, we prevented a leak of knowledge between the different years. This temporal split reduced direct temporal leakage and evaluated generalization to pitches recorded later in time, including potential changes in pitcher mechanics, repertoire, and environmental conditions.

3.5.2 *Evaluation metrics*

We evaluated each model architecture at three levels. First, MAE for parameter space evaluated the average error among predictions and true values of release parameters in every dimension (speed in mph, spin rate in RPM, spin axis angle in degrees, and release angles in degrees). Forward error evaluated the MAE between the trajectory from numerically integrating the predicted parameters, $f(\theta^*)$, and the target trajectory, τ^*, where the latter is an operational measure to quantify if a pitcher executing this prediction can produce the desired movement. Finally, we assessed

calibration through calculating the Spearman rank correlation coefficient between ensemble standard deviation and actual prediction errors across test pitches. Because the target metrics have different units and scales, each target was standardized to zero mean and unit variance using the mean and standard deviation calculated from the training set.

The three neural network architectures received data in structurally different forms. The MLP accepts a five-dimensional scalar trajectory descriptor, τ, made up of IVB, HB, vplate, xplate, and zplate as a fixed length vector with mean and variance equal to zero using training set statistics. The CNN and RNN accept the full 50-point trajectory sequence $x \in \mathbb{R}^{50x3}$ with each coordinate dimension normalized independently. For the RNN partial-trajectory experiments, sequences were truncated at the 30th, 40th, and 50th time step corresponding to ~ 60%, 80%, and 100% of the flight from rubber to plate respectively simulating a range of degrees of trajectory completeness.

4. Results

4.1 Prediction model comparison

Model comparison results are shown in Table 1. In addition to fitting release parameters as an inverse-optimization baseline using the nonlinear least-squares method (Levenberg-Marquardt), we also minimized the mean squared difference in predicted and measured forward trajectories for each pitch angle. The results were a 2.89 mph Mean Absolute Error (MAE) speed, a 267 RPM MAE in spin rate, a 6.5 degrees MAE in spin axis, and an average forward error of 9.7 inches (IVB) and 7.2 inches (HB). This performance was inferior to that of all neural architectures tested as well as to the Gradient Boosting baseline on all metrics. This result is consistent with our discussion of the ill-posed nature of the problem of optimizing release parameters per pitch via nonlinear least-squares minimization of the mean squared difference in forward-trajectory predictions made by a rank-deficient, non-injective model of the aerodynamics. In contrast, the neural networks are able to leverage their global, data-driven priors of the joint probability distribution of trajectories and release parameters for all pitches during training. The NLS baseline used bounded Levenberg-Marquardt optimization initialized at the population-mean release vector for the target pitch type, with parameters scaled to unit range and convergence declared at a relative step tolerance of $1x10^{-8}$; no multistart restarts were used. Because performance of nonlinear least-squares on an ill-conditioned inverse problem can depend strongly on initialization, scaling, and multistart strategy, this result should be read as showing that the neural architectures outperform this specific NLS configuration rather than as a general claim that learned inversion is inherently superior to classical inverse optimization. We also note that treating deviation from the recorded Statcast release vector as "parameter error" presumes that vector is the unique solution of the inverse problem; given the non-uniqueness discussed throughout this paper, the recorded release state is more accurately described as the supervised training target and an appropriate practical reference, rather than as the sole physically valid solution, and a prediction that differs from it while reproducing the target trajectory within tolerance is not thereby incorrect.

Table 1. Prediction accuracy of all models across evaluation metrics. Parameter-space MAE columns report mean absolute error for release speed (mph), spin rate (RPM), and spin-axis angle (degrees). Forward (Fwd) columns report trajectory error at home plate for induced vertical break (IVB) and horizontal break (HB) in inches. Calibration indicates the correlation between ensemble uncertainty and actual error where higher is better. Dashes indicate metrics not applicable to baseline models.

Model	Speed (mph)	Spin (RPM)	Spin Axis (degrees)	Fwd IVB (in)	Fwd HB (in)	Uncertainty–error Spearman ρ
NLS (Levenberg-Marquardt)	2.89	267	6.5	9.7	7.2	—
Linear Regression	3.42	312	10.1	6.2	5.9	—
Gradient Boosting	1.76	149	5.1	2.6	2.4	—
MLP	1.04	77	2.9	1.4	1.3	0.81
CNN	1.21	84	3.1	1.1	**1.0**	0.84
RNN/GRU (full trajectory)	1.18	81	3.0	1.2	1.1	0.82
RNN/GRU (80% trajectory)	1.31	91	3.5	1.4	1.3	0.80
RNN/GRU (60% trajectory)	1.57	108	4.2	1.7	1.6	0.76
Ensemble	**0.92**	**68**	**2.5**	**1.1**	**1.0**	**0.91**

A few findings stood out. The Multi-Layer Perceptron (MLP) achieved the least amount of mean absolute error in terms of the number of parameters required to describe the scalar trajectory features vplate and IVB/HB for the speed of the ball upon release at 1.04 mph and the spin rate of the ball at 77 RPM. The Convolutional Neural Network (CNN) achieved the least amount of forward error in terms of IVB at 1.1 inches and HB at 1.0 inches; this demonstrates that the entire spatial trajectory shape contains information regarding how the Magnus force develops during a pitch which the scalar endpoints cannot obtain. However this does not necessarily imply the recovery of the true physical release state due to the non-unique solution of the inverse trajectories. Instead, low forward error demonstrates that the trajectory produced by the solution is similar to the desired trajectory.

The Recurrent Neural Networks (RNN) were able to achieve performance very similar to the CNN, and demonstrated that recognizing temporal ordering of position observations contained information similar to that of spatial pattern recognition. The RNN was also found to degrade gracefully as trajectory completeness was decreased: at an 80% complete trajectory (~ the position 12 feet from the plate) forward error increased from 1.2 to 1.4 inches, however, was still competitive with the MLP. At a 60% complete trajectory (~ 24 feet from the plate) the error increased to 1.7 inches. In practical terms, a real-time pitching coach application could continually update their release parameter predictions based on ball flight observation data over the first half of a pitch's flight path, thus allowing for progressively more accurate coaching as additional observation data become available.

The multi-architecture ensemble produced the best results across all measured parameters; achieving forward IVB and HB errors of 1.1 and 1.0 inches respectively, both less than the ~ 2-inch perceptual discrimination threshold for Major League Baseball batters. The ensemble uncertainty–error correlation ($\rho = 0.91$) was higher than that of any individual architecture, consistent with prior work demonstrating the utility of deep ensembles for predictive uncertainty estimation (19).

A comparative examination of where each architecture failed most distinctly illustrates complementary strength and weakness among architectures. For example, the MLP showed significantly elevated errors for pitches characterized as having high break, where non-linear relationship between spin-axis and movement is highest, and scalar endpoint descriptors contain inadequate constraints to provide precise location information. The CNN displayed elevated errors for pitches characterized by extreme release angle values, where ball-flight trajectories have been shifted systematically away from those seen during training and therefore fail to generalize because of fixed length convolution kernel processing. The RNN exhibited elevated errors for pitches exhibiting abrupt late-break onset, where the attention mechanism assigns excessive weight to early-flight observation data prior to the break occurring.

The error profiles indicate a division-of-labor. The MLP should be used when only scalar trajectory feature descriptors are available such as when using broadcast data. The CNN is preferable when full ball-flight sequence is available and either pitch-type or break-magnitude is predicted to be extreme. The RNN is preferable for partial-trajectory based applications or real-time streaming analyses. The ensemble achieves the lowest errors observed in this study by combining predictions from all three architectures using output-specific weights intended to exploit their complementary strengths.

4.2 Pitch type comparison

Table 2 shows that in addition to being more accurate than the MLP and RNN on all 8 pitch types, the CNN also had larger improvement margins over the MLP and RNN for certain pitches. Those two pitches were the change-up (CNN: 1.52, MLP: 1.74), and the sweeper (CNN: 1.41, MLP: 1.62). This may be attributed to the fact that both of those pitches have a middle-of-the-air shape that does not lend itself as well to being described using scalar summary measures.

A change-up is pitched to act like a fastball until about 40 ft. down the track at which time it drops off quickly, creating a unique spatial curvature profile. Therefore, although the vertical movement, as measured by IVB, and plate-crossing speed may appear similar to those of a fastball, these scalar measures do not fully represent the trajectory characteristics of the pitch.

On the other hand, a sweeper has a gradual buildup of horizontal Magnus force as it travels down the track. By the time the ball reaches about 20 ft., there is enough horizontal force generated to create a distinct lateral acceleration signature. Since the CNN uses spatial filters to analyze the trajectory of the ball, it can identify

these unique shapes as opposed to analyzing only end-point measurements.

Table 2. Combined forward trajectory error (inches) by pitch type for each model architecture. Values represent mean absolute error averaged over induced vertical break (IVB) and horizontal break (HB) on the held-out 2024 test set.

Pitch Type	MLP	CNN	RNN (full)	Ensemble	Calib. ρ
Four-Seam Fastball	1.18	0.98	1.04	0.91	**0.93**
Sinker	1.31	1.14	1.21	1.09	**0.90**
Sweeper	1.62	1.41	1.53	1.34	**0.82**
Slider	1.48	1.29	1.39	1.22	**0.86**
Curveball	1.22	1.07	1.13	1.01	**0.92**
Changeup	1.74	1.52	1.61	1.44	**0.79**
Cutter	1.29	1.14	1.21	1.06	**0.88**
Splitter	1.97	1.71	1.83	1.61	**0.74**

The differences in performance between architectures become less pronounced when moving to pitch types that have aerodynamic profiles that are less complex. For example, four-seam fastballs have very simple aerodynamic profiles. They are primarily influenced by backspin driven backspin Magnus forces acting in essentially one direction along the entire path of flight. As a result, they are the easiest pitch type to invert using either model. The CNN was able to achieve a forward error of 0.98 inches on average for four-seam fastballs, and the ensemble reduced this further to 0.91 inches.

Curveballs have similar conditions. Their high levels of topspin cause a dominant and predictable downward Magnus force. In addition, the scalar measure IVB provides an efficient representation of this signal, thus explaining why the MLP's 1.22 inch error is so much closer to the CNN's 1.07 inch error.

Sinkers and cutters fall somewhere in between the curveball and splitter in terms of the complexity of their aerodynamic profiles. Sinkers and cutters typically experience moderate amounts of horizontal and vertical break. The CNN is slightly better at resolving this than the MLP as sinker and cutter break is dependent upon the orientation of the seam on the baseball.

Finally, splitters prove to be the most difficult pitch type to invert regardless of the architecture used. Even though the ensemble performs significantly better than any individual model for all pitch types except splitters (reducing errors by up to 0.10 inches), none of the models perform exceptionally well on splitters. The best performing model, the CNN, performed with an average error of 1.71 inches. However, even with the use of ensembling techniques, the ensemble was unable to reduce this error below 1.61 inches. There are several reasons why splitters pose such a challenge to inversion.

This can be confirmed by analyzing the uncertainty in the form of calibration. The relationship between the accuracy of each type of pitch in the ensemble's calibration, and how well the ensemble can accurately predict what will happen next when it attempts to use that information follows a very consistent pattern. Pitches with simple, low dimensional aerodynamics and thus easy to understand and model, are those for which the ensemble has been able to make an accurate assessment of how much "uncertainty" there is about that pitch. Fastballs thrown from over the top (Four-Seam Fastball) and curveballs have a primary direction of force due to the Magnus effect; therefore these two types of pitches provide the best estimates of their corresponding trajectories based on the ensemble's inversion. Therefore, they have the highest calibration level ($\rho = 0.93$ for four-seam fastballs and $\rho = 0.92$ for curveballs). The splitter and the changeup, both of which are characterized by complex, compensatory relationships between the amount of spin imparted onto the ball and the speed at which it was released, were found to have the lowest calibration levels ($\rho = 0.74$ for splitters and $\rho = 0.79$ for changeups). Sweeper balls, which have characteristics similar to those of the splitter/changeup in terms of developing a horizontal force component gradually during flight, fall somewhere in the middle ($\rho = 0.82$). These results suggest that rather than simply being random, the degree of agreement among members of the ensemble provides a qualitative measure of the difficulty associated with converting observed motion into predicted future motion.

4.3 Admissible solution space analysis

To show the lack of uniqueness in the inverse map's solutions and to examine the admissible solution set, we used the exact same forward aerodynamics model (Eqs. 1–11) that generated our training targets based on sequences. Four representative pitch types from the entire scope of aerodynamically complex pitches listed in Table 2 — four-seam fastball, curveball, sweeper, and splitter — each had a representative release vector θ^* characteristic of that pitch type defined along with its corresponding target trajectory descriptor τ^* and complete trajectory X^*. Then we randomly sampled 10,000 possible release vectors θ from a box centered around θ^* ($v_{rel} \pm 8$ mph, $\omega \pm 600$ RPM, $\eta \pm 0.30$, $\varphi_{spin} \pm 60°$, $\varphi_{rel} \pm 6°$, $\theta_{rel} \pm 4°$) and used these candidate release vectors to propagate through the forward model and retain them as valid candidates if their computed trajectories fit the specified tolerance ε to either the target trajectory descriptor τ = (IVB, HB) or the full 50-point trajectory X. The sampling process was repeated under two different criteria for admissibility - matching only the scalar descriptor τ = (IVB, HB), as accessible to the MLP, and matching the full 50-point trajectory X, as accessible to the CNN/RNN - and at four different levels of tolerance per criterion (ε = 0.25, 0.5, 1.0, 2.0 in for the scalar criterion; ε = 1.5, 2.5, 4.0, 6.0 in for the full-trajectory criterion). The intent was to evaluate how sensitive the shape of the admissible region was to varying tolerance.

The results showed three main features. Firstly, regardless of pitch type or whether the scalar descriptor or full trajectory was being used as a basis for determining admissibility, the admissible parameter-space fraction, of the admissible region increased smoothly and continuously with increasing tolerance for every

pitch type examined (see Figure 2a-b), while the ranking of pitch types by admissible parameter-space fraction, of admissible region remained constant across all values of tolerance. Hence, none of the subsequent findings depended on an arbitrary value of tolerance having been selected.

Secondly, under the scalar descriptor criterion alone, the splitter consistently produced significantly larger regions of admissible release vectors than do all of the other three pitch types tested (at a tolerance level of $\varepsilon = 2.0$ in, for example, about 12.0% of the sample box produced valid trajectories for the splitter whereas about 4.8-5.4% did for each of the three remaining types). Further, the order of relative sizes of admissible region - splitter (largest) > sweeper > curveball ≈ four-seam fastball (smallest) - was opposite to that found by comparing the ensemble calibration accuracy reported in Table 2 ($\rho = 0.74, 0.82, 0.92, 0.93$ for each respective pitch type); although no generalization beyond these four specific pitches can be made without additional testing, this represents the first empirical comparison here in the manuscript between ensemble disagreement and physical non-identifiability.

Lastly, and most relevant to understanding the relative performance results shown in Table 1, the structure of admissible release conditions differs depending upon which criterion is applied. Specifically, for the case where admissibility is determined solely by matching a scalar descriptor τ = (IVB, HB) available only at home plate, there appear to be few constraints on the angle of release (φ_rel) and elevation angle (θ_rel) even at the smallest tolerance tested (the range of angles permitted by these criteria typically spans nearly the entire sampled range; i.e., for each pitch type examined in Table 2, φ_rel and θ_rel have relatively little effect on IVB and HB when evaluated at home plate). By way of contrast, when evaluating admissibility by requiring matches of the full trajectory X available to both CNN/RNN models and not available to the MLP models, φ_rel and θ_rel are constrained ~ 4-6 times tighter (e.g., at comparable fractions of accepted candidate releases as measured by φ_rel and θ_rel narrowed from ~ 11.6° to 3.3° and from ~ 7.9° to 2.0° for the four-seam fastball). This demonstrates that full-trajectory information can constrain release direction substantially more strongly than IVB and HB alone. Because the MLP uses three additional scalar descriptors, however, this analysis should not be interpreted as a direct information-matched explanation of the CNN/RNN performance advantage.

Release velocity, spin rate and spin efficiency are also generally degenerate under both criteria (Figure 3) as evidenced by a physical compensation: The Magnus force term in Equation (4) is proportional to $v^2 \cdot C_L(S)$ such that a fast release with inefficient spin and slow release with efficient spin produce similar trajectories. In fact, this compensation is evident in the sampled admissible cloud as a negative correlation between ω and η for each of the four-pitch types tested (Pearson r = -0.44 to -0.74; Figure 3).

We stress again that we are limited by the nature of this study. It uses the same simple forward model that was used to create our training data for sequencing, so it does not consider possible degeneracies due to omitted aerodynamics (e.g., seam orientation, shift in wake of seams,

variations in atmosphere), nor does it evaluate all the pitches in our test set; instead, it evaluates four representative ones. The sampling-box is a modeling choice and its bounds affect the absolute admissible fractions. The robustness of the observed pitch-type ordering to alternative sampling domains remains to be established. Thus, this analysis should be viewed as a necessary initial, reproducible step toward understanding the structure of the ill-posedness of the baseball inverse problem rather than a comprehensive solution to it.

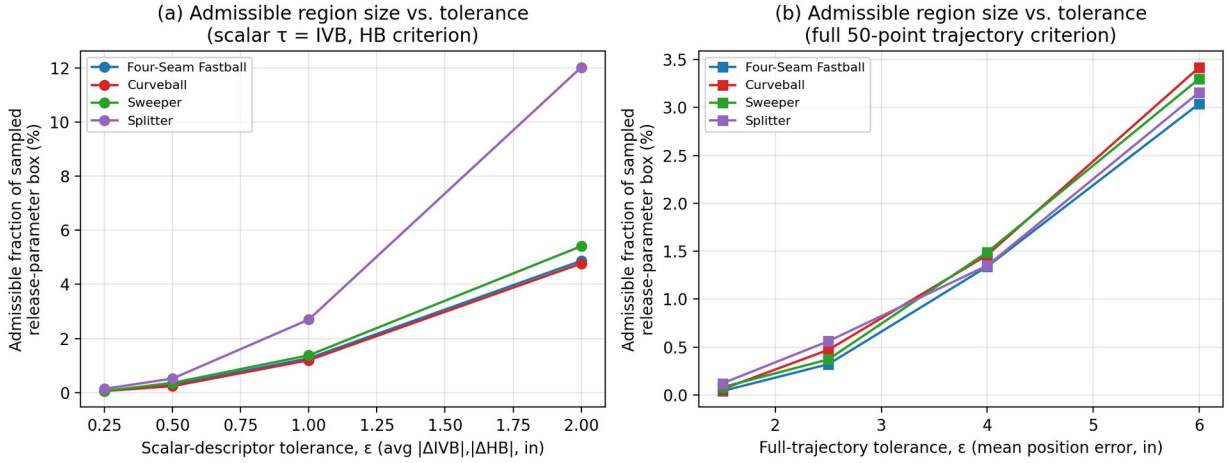


Figure 2. (a) Fraction of the sampled release-parameter box accepted as admissible under the scalar-descriptor criterion (matching IVB and HB only), as a function of tolerance ε, for four representative pitch types. (b) Same, under the full 50-point trajectory-matching criterion. The two horizontal axes are not directly comparable in scale (see text).

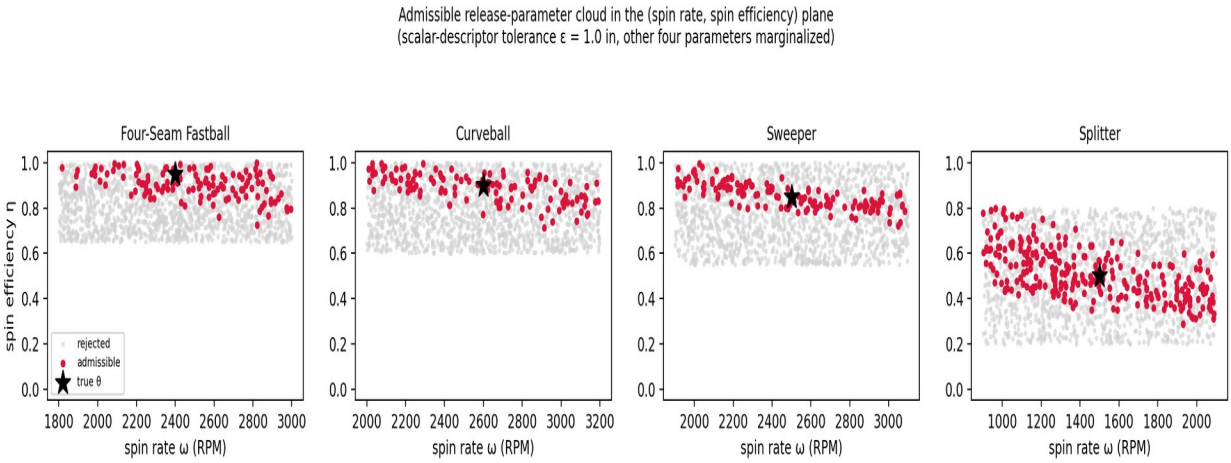


Figure 3. Admissible release-parameter cloud in the (spin rate ω, spin efficiency η) plane for each representative pitch type, at a scalar-descriptor tolerance of ε = 1.0 in (other four release parameters marginalized). Gray points are rejected candidates; red points are admissible; the black star marks the representative release vector θ* for that pitch type. The negative ω-η association visible in each panel reflects the compensation in the Magnus force term (Eq. 4).

A further limitation concerns the comparison between the scalar and full-trajectory admissible solution spaces. The scalar admissibility analysis was defined using IVB and HB, whereas the MLP received five trajectory descriptors (IVB, HB, plate velocity, and horizontal and vertical plate location). Consequently, the broader admissible regions obtained under the IVB/HB criterion cannot be interpreted as a direct information-matched explanation for the performance difference between the MLP and sequence-based CNN/RNN models. The additional MLP inputs may themselves constrain release velocity, release direction, or other components of the admissible parameter space. The present analysis therefore demonstrates that full-trajectory information can reduce degeneracy relative to IVB/HB alone, but does not establish that this reduction accounts for the observed CNN/RNN advantage over the MLP. Future work should construct an admissible-space criterion incorporating all five MLP inputs and compare it directly with the full-trajectory criterion.

4.4 Ablations

Table 3 shows the effect of eliminating or reducing one architectural element at a time from each model. With respect to the MLP, depth was the most important variable: the 4-layer shallow version only reached 1.9 inch forward RMSE and 3.8° spin axis MAE; the 7-layer version without skip connections improved to 1.7 inch and 3.4°. When adding skip connections to the 7-layer version resulted in a full MLP result of 1.6 inch and 2.9°. This pattern follows what is already well-known to be difficult for training large networks without paths for residual gradients: gradient strength decreases as they travel back through multiple non-linear layers, and skip connections provide the required gradient path without needing larger layers or less learning rate.

Table 3. Ablation study results. Forward RMSE (inches) and spin-axis MAE (degrees) are reported for progressively complete versions of each architecture and for the final ensemble. Each row (except that labeled ‘full’) disables one architectural component relative to the full configuration.

Configuration	Forward RMSE (in)	Spin Axis MAE (°)
MLP, 4 hidden layers (shallow)	1.9	3.8
MLP, 7 layers, no skip connections	1.7	3.4
MLP, 7 layers + skip (full)	1.6	2.9
CNN, no multi-scale (single stage)	1.6	3.3
CNN, no skip connections	1.4	3.2
CNN, multi-scale + skip (full)	1.2	3.1
RNN, vanilla GRU (unidirectional)	1.5	3.4
RNN, bidir. GRU, no attention	1.3	3.1
RNN, bidir. GRU + attention (full)	1.2	3.0
Ensemble (5 × each architecture)	**1.1**	**2.5**

The CNN’s multi-scale backbone was the dominant architectural contributor. Replacing the three-stage architecture with a single convolutional stage of equivalent total capacity

increased the forward RMSE from 1.2 to 1.6 inches. Because the single-stage network processes the entire trajectory using only one receptive-field scale, it is less able to capture both the coarse velocity characteristics of early flight and the finer curvature associated with late break. Removing the skip connections while retaining the multi-scale architecture increased the error to 1.4 inches. Thus, both multi-scale processing and skip connections contributed to CNN performance, although through different mechanisms: multi-scale processing captures trajectory features across different spatial scales and phases of flight, whereas skip connections preserve representations learned in earlier stages as information propagates through deeper layers.

Bidirectionality and attention made ~ equal contributions to the RNN. The unidirectional baseline produced a forward RMSE of 1.5 inches. Because it processes the trajectory sequentially from release toward home plate, information about late break becomes available only after the later portions of the sequence have been processed, making the final representation strongly dependent on the last hidden state. Adding a backward GRU stream reduced the error to 1.3 inches by allowing the trajectory to be processed in both directions, thereby incorporating information from late flight into the sequence representation. Attention pooling across the 50 hidden states further reduced the error to 1.2 inches by allowing the network to assign greater weight to the most informative portions of the trajectory rather than treating all time steps equally. Finally, the ensemble, which stacked 15 trained models across the three architectures, reduced the forward error to 1.1 inches and achieved a calibration of 0.91, providing both greater predictive accuracy and stronger uncertainty calibration than the individual architectures.

5. Perspectives and future application

The non-uniqueness of the inverse problem may have practical implications beyond limiting the recovery of a unique release state. If multiple combinations of release speed, spin rate, spin efficiency, and release direction can produce essentially the same trajectory, the admissible solution space can instead be regarded as a pitch-design space. Importantly, the present results suggest that the extent of this flexibility may differ among pitch types. Among the representative pitches examined here, the splitter exhibited a substantially larger admissible release-parameter region than the sweeper, curveball, and four-seam fastball, suggesting that different pitch types may impose different degrees of constraint on how their characteristic trajectories can be produced. A pitch with a broad admissible region may permit substantial compensation among release parameters, whereas a more tightly constrained pitch may require a narrower combination of release conditions. This distinction could have direct implications for coaching and individualized pitch development. Rather than prescribing an idealized release condition, a coach could potentially identify the range of release states capable of producing the desired pitch and select the combination most compatible with an individual pitcher's velocity, spin-generation ability, spin efficiency, release mechanics, and reproducibility. The same framework could eventually provide an additional dimension for player evaluation and valuation. Expertise in a tightly constrained pitch may represent a different technical

capability from proficiency in a pitch possessing a broad admissible solution space, while a pitcher capable of reliably producing several pitch types with differently constrained release requirements may possess greater adaptable pitching capacity. Conversely, a player's apparent inability to reproduce a particular pitch need not imply a general limitation if alternative admissible release strategies exist that better match that player's capabilities. Thus, inverse-space characterization could ultimately complement conventional outcome-based statistics by distinguishing not only what pitches a player can produce, but also the physical constraints, flexibility, and reproducibility underlying that repertoire. These coaching and player-evaluation applications remain hypotheses rather than demonstrated consequences of the present study. Future work should therefore establish whether pitch-type-specific admissible spaces persist across larger populations, determine whether pitchers can deliberately exploit predicted compensatory release strategies while preserving target trajectories, and test whether the size and structure of these spaces are associated with pitch learnability, reproducibility, performance, or player development outcomes.

6. Conclusion

This study demonstrates that the baseball inverse-trajectory problem can be approached effectively using neural-network architectures that exploit different representations of ball-flight information. Across eight major pitch types, the MLP, CNN, and GRU-RNN provided complementary capabilities for inferring aerodynamic release parameters from scalar, complete-trajectory, and partial-trajectory information, respectively. Their combination in an ensemble produced the lowest overall prediction error in this study, with forward-simulated trajectories reproducing observed pitch movement to ~ one inch. The ability of the recurrent model to retain useful predictive performance from incomplete trajectories further suggests that release conditions can be estimated before the complete ball flight is available.

Prediction accuracy, however, represents only one aspect of the inverse problem. A central finding of this study is that an accurately reconstructed trajectory does not necessarily imply that the underlying release conditions have been uniquely identified. Direct sampling of the admissible release-parameter space demonstrated that multiple release states can reproduce essentially the same target trajectory. Moreover, this non-uniqueness was structured rather than uniform. Among the four representative pitches examined, the extent of the admissible region differed by pitch type, while identifiability also differed substantially among release parameters. Full ball-flight information constrained release-direction angles much more strongly than scalar trajectory descriptors, whereas release velocity, spin rate, and spin efficiency retained substantial compensatory freedom. The negative association between admissible spin rate and spin efficiency further illustrates how different physical release conditions can compensate for one another while preserving similar ball flight.

These results establish an important distinction between model-dependent predictive accuracy and the inherent physical identifiability of release conditions. A neural network may accurately predict one plausible release state

without that state being the unique physical solution. Conversely, the size and structure of the admissible solution space are properties of the inverse problem under the aerodynamic model and available trajectory information, rather than consequences of the particular neural-network architecture used to solve it. Across the four representative pitches examined, ensemble uncertainty covaried with the independently estimated extent of physical non-identifiability. This finding suggests that model disagreement may contain information about inverse ambiguity, although broader validation is required before ensemble uncertainty can be interpreted as a general proxy for physical non-identifiability.

This distinction also changes the practical interpretation of the inverse problem. Non-uniqueness need not be regarded solely as a limitation: the admissible release region can instead be viewed as a pitch-design space. If several combinations of velocity, spin, efficiency, and release direction can produce a desired trajectory, pitch design need not seek a single theoretically optimal release condition. Future systems could instead identify alternative admissible solutions and determine which are most compatible with an individual pitcher's physical capabilities and reproducibility. The observation that the representative pitch types examined here differed in the size of their admissible spaces further raises the possibility that pitches differ intrinsically in the flexibility with which their characteristic trajectories can be produced.

The principal contribution of this work is therefore not simply the application of neural networks to infer baseball release parameters. Rather, it provides a framework for separating how accurately an inverse model can recover a plausible solution from how strongly the observed trajectory constrains the underlying physical solution space. Extending the admissible-space analysis across larger numbers of pitches and pitchers, incorporating all available trajectory descriptors and more complete aerodynamic effects, and experimentally testing whether pitchers can exploit predicted compensatory release strategies will determine whether this framework can ultimately support individualized coaching, pitch development, and repertoire evaluation.

7. References

1. Nathan, A. M. (2003). The physics of baseball. Physics Today, 56(10), 32–37.

2. Bahill, A. T., Baldwin, D. G., & Venkateswaran, J. (2005). Predicting a baseball's path. American Scientist, 93(3), 218–225. https://doi.org/10.1511/2005.53.218

3. Tikhonov, A. N., & Arsenin, V. Y. (1977). Solutions of ill-posed problems. Winston.

4. Driveline Baseball. (2022). Modern pitch design: Translating movement goals into mechanical instructions. Driveline Baseball Research Blog.

5. Seydel, R. (2010). Practical bifurcation and stability analysis (3rd ed.). Springer. https://doi.org/10.1007/978-1-4419-1740-9

6. Hadamard, J. (1902). Sur les problèmes aux dérivées partielles et leur signification physique. Bulletin de l'Université de Princeton, 13, 49–52.

7. Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. Nature, 323(6088), 533–536. https://doi.org/10.1038/323533a0

8. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444. https://doi.org/10.1038/nature14539

9. Elman, J. L. (1990). Finding structure in time. Cognitive Science, 14(2), 179–211. https://doi.org/10.1207/s15516709cog1402_1

10. Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling [Preprint]. arXiv. https://doi.org/10.48550/arXiv.1412.3555

11. Baseball Savant. (n.d.). Baseball Savant: Statcast, trending MLB players and visualizations. Major League Baseball. Retrieved July 28, 2026, from https://baseballsavant.mlb.com/

12. Petti, J., & Gilani, S. (2021). pybaseball: A Python package for baseball data. Journal of Open Source Software, 6(59), 3234.

13. Nathan, A. M. (2008). The effect of spin on the flight of a baseball. American Journal of Physics, 76(2), 119–124. https://doi.org/10.1119/1.2805242

14. Burden, R. L., Faires, J. D., & Burden, A. M. (2019). Numerical analysis (11th ed.). Cengage Learning.

15. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 770–778). IEEE. https://doi.org/10.1109/CVPR.2016.90

16. Hendrycks, D., & Gimpel, K. (2016). Gaussian error linear units (GELUs) [Preprint]. arXiv. https://doi.org/10.48550/arXiv.1606.08415

17. Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. In International Conference on Learning Representations (ICLR). https://doi.org/10.48550/arXiv.1711.05101

18. dos Santos, C. N., Tan, M., Xiang, B., & Zhou, B. (2016). Attentive pooling networks [Preprint]. arXiv. https://doi.org/10.48550/arXiv.1602.03609

19. Lakshminarayanan, B., Pritzel, A., & Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. Advances in Neural Information Processing Systems, 30, 6402–6413.

20. Owens, N., Harris, C., & Stennett, C. (2003). Hawk-Eye tennis system. In 2003 International Conference on Visual Information Engineering (VIE 2003) (pp. 182–185). Institution of Electrical Engineers. https://doi.org/10.1049/cp:20030517