Linear Regression: Reading Material

AERO 689: Introduction to Machine Learning for Aerospace Engineers

Dr. Raktim Bhattacharya

Table of contents

1 Introduction

Linear regression is one of the most fundamental and widely used techniques in machine learning and statistical modeling. In aerospace engineering, it plays a crucial role in predicting aircraft performance, analyzing wind tunnel data, and optimizing flight operations. This reading material provides detailed explanations of the concepts presented in the lecture slides, with a focus on both mathematical rigor and practical aerospace applications.

1.1 Learning Objectives

By the end of this module, you should be able to:

1. **Apply linear regression to aerospace performance prediction**: Understand how to use linear regression to model relationships between flight parameters (e.g., angle of attack, Mach number) and performance metrics (e.g., drag coefficient, fuel consumption).
2. **Understand least squares method and gradient descent**: Master both the analytical (closed-form) and iterative (gradient descent) approaches to solving linear regression problems.
3. **Implement drag coefficient prediction from wind tunnel data**: Learn to process real experimental data and build predictive models that account for measurement noise and physical constraints.
4. **Validate models using aerospace-specific metrics**: Understand how to assess model quality using appropriate statistical measures and ensure predictions meet safety and certification requirements.

2 Motivation: The Fuel Crisis Challenge

2.1 The $100 Million Question

Consider a real-world scenario that aerospace engineers face daily: an airline operates a fleet of 200 aircraft. The fuel cost alone exceeds $50 million annually per aircraft type. With such enormous operational costs, even small improvements in fuel efficiency can translate to massive savings.

The Challenge: Airlines need to accurately predict fuel consumption for flight planning. Currently, most operations rely on simplified performance charts that were developed under idealized conditions. These charts provide conservative estimates but may not capture the full complexity of real-world flight operations.

The Machine Learning Opportunity: By using actual flight data collected from thousands of flights, we can build precise models that account for:

- Actual weather conditions encountered
- Real payload and weight variations
- Engine performance degradation over time
- Pilot technique variations
- Air traffic control routing constraints

2.2 Real Impact

The potential benefits of improved fuel consumption models are substantial:

- **1% fuel savings** = Over $100 million annually across the industry
- **Better range predictions** = More efficient route planning and optimization
- **Accurate payload calculations** = Improved safety margins while maximizing revenue cargo capacity

Discussion Question: What factors do you think affect aircraft fuel consumption? Consider environmental factors (weather, altitude, temperature), operational factors (weight, speed, routing), and mechanical factors (engine condition, aerodynamic efficiency).

3 From Wind Tunnel to Flight: The Data Challenge

3.1 Traditional Approach: Empirical Models

Aerospace engineers have long used empirical models to characterize aircraft performance. One classic example is the **parabolic drag polar**:

$$C_D = C_{D_0} + KC_L^2$$

where:

- C_D is the drag coefficient
- C_{D_0} is the zero-lift drag coefficient (parasitic drag)
- K is the induced drag factor
- C_L is the lift coefficient

The Problem: This model assumes perfectly controlled conditions—smooth flow, steady state, clean configuration. In reality:

- Real flights encounter turbulence, wind shear, and varying atmospheric conditions
- Aircraft weight changes continuously as fuel is consumed
- Engines degrade over time, affecting performance
- Manufacturing tolerances mean each aircraft is slightly different

The Solution: Machine learning allows us to learn patterns directly from operational data, capturing complexities that simplified analytical models may miss.

3.2 Available Data Sources

Aerospace engineers can draw from three main sources of data:

3.2.1 1. Wind Tunnel Data

Characteristics:

- Highly controlled environment
- Precise measurements
- Limited range of conditions
- Expensive to collect
- May not capture full-scale Reynolds number effects

Best for: Understanding fundamental aerodynamic behavior, validating CFD simulations

3.2.2 2. Flight Test Data

Characteristics:

- Real flight conditions
- Instrumented aircraft
- Very expensive to collect (dedicated test aircraft, crew, facilities)
- Limited sample size
- High quality, well-documented

Best for: Aircraft certification, validating performance predictions, boundary exploration

3.2.3 3. Operational Data

Characteristics:

- Massive scale (thousands or millions of flights)
- Representative of actual operations
- Noisy (sensor errors, environmental variations)
- May lack detailed instrumentation
- Continuously collected

Best for: Statistical modeling, fleet-wide trends, operational optimization

3.3 The Linear Regression Framework

For this module, we'll focus on a specific problem: predicting the drag coefficient from flight parameters.

Goal: Predict C_D (drag coefficient) accurately

Input Features:

- α (angle of attack)
- M (Mach number)
- Re (Reynolds number)
- Potentially interaction terms and polynomial features

Output: Drag coefficient value for performance calculations

Why is this important? Accurate drag prediction is essential for:

- Fuel consumption estimation
- Range calculations
- Climb performance
- Flight envelope determination
- Control system design

4 Mathematical Foundation: The Linear Model

4.1 General Form

Linear regression models the relationship between input features and a continuous output variable. For a dataset with n samples and d features, the model is:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_d x_{id} + \epsilon_i$$

Notation Explained:

- y_i: The response variable (dependent variable) for sample i. In aerospace: this could be drag coefficient, fuel flow rate, or any measurable output.
- x_{ij}: The j-th feature (independent variable) of the i-th sample. Examples: Mach number, angle of attack, altitude, etc.
- β_j: Regression coefficients (parameters) that we need to learn from data. These quantify how each feature affects the response.
- β_0: The intercept term. This represents the baseline value when all features are zero.
- ϵ_i: The error term (residual). This captures:
  - Measurement noise
  - Unmodeled physics
  - Random variations
  - Model approximation errors

4.2 Vector Notation

To work with all samples simultaneously, we use matrix-vector notation:

$$y = X\beta + \epsilon$$

where:

- $y \in \mathbb{R}^n$: Vector of all n response values
- $X \in \mathbb{R}^{n \times (d+1)}$: Design matrix (also called feature matrix)
- $\beta \in \mathbb{R}^{d+1}$: Vector of true parameters (including intercept)
- $\epsilon \in \mathbb{R}^n$: Vector of all error terms

The design matrix X is augmented with a column of ones to account for the intercept term.

5 Matrix Formulation

5.1 Design Matrix Structure

The design matrix organizes all our data into a structured format:

$$X = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1d} \\ 1 & x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix}$$

Understanding the structure:

- Each **row** represents one data sample (one observation, one flight, one wind tunnel measurement)
- Each **column** (except the first) represents one feature across all samples
- The **first column** of all ones corresponds to the intercept term β_0
- Dimensions: n rows $\times$ $(d+1)$ columns

The parameter vector and response vector are:

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_d \end{bmatrix}, \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

5.2 Aerospace Example: Drag Prediction

Let's make this concrete with a drag coefficient prediction problem.

Scenario: We have wind tunnel data measuring drag coefficient at various angles of attack and Mach numbers. We want to build a model that includes: - Linear terms: α, M, Re - Quadratic terms: α^2, M^2 - Interaction terms: αM, αRe, etc.

Design Matrix:

$$X = \begin{bmatrix} 1 & \alpha_1 & M_1 & \text{Re}_1 & \alpha_1^2 & M_1^2 & \alpha_1 M_1 & \cdots \\ 1 & \alpha_2 & M_2 & \text{Re}_2 & \alpha_2^2 & M_2^2 & \alpha_2 M_2 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Parameter Vector:

$$\beta = \begin{bmatrix} C_{D_0} \\ k_\alpha \\ k_M \\ k_{\text{Re}} \\ k_{\alpha^2} \\ k_{M^2} \\ k_{\alpha M} \\ \cdots \end{bmatrix}^T$$

Target Vector:

$$y = \begin{bmatrix} C_{D_1} \\ C_{D_2} \\ \cdots \\ C_{D_n} \end{bmatrix}^T$$

This formulation allows us to capture complex aerodynamic behavior while maintaining the linear regression framework.

6 Key Insight: What "Linear" Means

6.1 Linear in Parameters, Not Features

A common source of confusion: "linear regression" refers to linearity **in the parameters** β, **not** in the input features x.

The General Form:

$$y = \beta_0\phi_0(x) + \beta_1\phi_1(x) + \cdots + \beta_d\phi_d(x)$$

where $\phi_j(x)$ are **basis functions** that can be: - Linear: $\phi_1(x) = x$ - Polynomial: $\phi_2(x) = x^2$, $\phi_3(x) = x^3$ - Trigonometric: $\phi_4(x) = \sin(x)$, $\phi_5(x) = \cos(x)$ - Exponential: $\phi_6(x) = e^x$ - Interactions: $\phi_7(x_1, x_2) = x_1x_2$ - Any other nonlinear transformation

Key Properties:

1. **Basis functions** $\phi_j(x)$ are **fixed** (you choose them before fitting)
2. **Coefficients** β_j are **what we solve for** (learned from data)
3. The model is **linear** in β_j because each β_j appears with power 1 and no products of different β_j terms
4. The model can be **nonlinear** in x because the basis functions can transform inputs arbitrarily

6.2 Why Linearity in Parameters Matters

Mathematical Benefit: Linearity in parameters means:

- The optimization problem is **convex** (has exactly one global minimum, no local minima)
- A **closed-form solution** exists (we can write down the exact answer)
- The solution is **unique** (if $X^T X$ is invertible)
- We can use **efficient linear algebra** algorithms

Practical Benefit: We can model complex, nonlinear physical phenomena while maintaining: - Computational efficiency - Guaranteed convergence - Interpretable coefficients - Statistical properties (confidence intervals, hypothesis tests)

6.3 Aerospace Example

Consider modeling the drag coefficient with this equation:

$$C_D = \beta_0 + \beta_1 \alpha + \beta_2 \alpha^2 + \beta_3 M^2 + \beta_4 (\alpha M)$$

Analysis:

- **Nonlinear in physical variables**: The relationship between C_D and (α, M) is nonlinear due to the α^2, M^2, and αM terms. This is a parabolic surface in the (α, M) space.
- **Linear in parameters**: The equation is a weighted sum of basis functions:

$$C_D = \beta_0 \cdot 1 + \beta_1 \cdot \alpha + \beta_2 \cdot \alpha^2 + \beta_3 \cdot M^2 + \beta_4 \cdot (\alpha M)$$

- **Effect of doubling** β_2: If we double β_2, the contribution of the α^2 term exactly doubles. This linear relationship in the parameters is what makes the problem tractable.

Matrix Representation:

$$X = \begin{bmatrix} 1 & \alpha_1 & \alpha_1^2 & M_1^2 & \alpha_1 M_1 \\ 1 & \alpha_2 & \alpha_2^2 & M_2^2 & \alpha_2 M_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

Each row processes one data point through all basis functions, creating a standard linear regression problem.

7 Real Data: Noisy Measurements

7.1 Understanding Measurement Noise

In practice, all experimental data contains measurement errors. For aerospace applications, these errors arise from:

1. **Sensor limitations**: Finite precision, drift, calibration errors
2. **Environmental factors**: Temperature variations, atmospheric turbulence
3. **Flow unsteadiness**: Turbulent fluctuations, vortex shedding
4. **Model simplification**: Real physics is more complex than our model
5. **Human factors**: Data recording errors, experimental setup variations

7.2 Characteristics of Real Wind Tunnel Data

When measuring drag coefficients in a wind tunnel, we typically observe:

- **Data scatter** around the true relationship
- **Heteroscedastic noise**: Measurement uncertainty often increases with angle of attack (larger forces $\rightarrow$ larger absolute errors)
- **Systematic biases**: Wall effects, support interference
- **Outliers**: Occasionally anomalous measurements due to flow separation or experimental issues

7.3 Why the Error Term ϵ_i is Essential

The error term in our model $y_i = x_i^T \beta^* + \epsilon_i$ is not just a mathematical convenience—it's a fundamental recognition that:

1. **No model is perfect**: Even the best physical model cannot capture every detail
2. **Measurements are imperfect**: Sensors have inherent limitations
3. **Random variations exist**: Physical processes have inherent stochasticity
4. **We seek expected behavior**: Our goal is to find the average relationship, not fit every noise fluctuation

Important: We want our model to capture the **signal** (true underlying relationship) without overfitting to the **noise** (random fluctuations). This is the essence of good statistical modeling.

8 The Optimization Problem

8.1 Objective: Minimize Squared Error

Given data $\{(x_i, y_i)\}_{i=1}^n$, we want to find parameters β that make our predictions $\hat{y}_i = x_i^T\beta$ as close as possible to the observed values y_i.

Residual for sample i:

$$r_i = y_i - \hat{y}_i = y_i - x_i^T\beta$$

Residual Sum of Squares (RSS):

$$\text{RSS}(\beta) = \sum_{i=1}^{n} r_i^2 = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}(y_i - x_i^T\beta)^2$$

8.2 Why Square the Errors?

We could use different error metrics. Why do we square the errors?

1. **Penalizes large errors more**: An error of 2 contributes 4 to RSS, while two errors of 1 each contribute only 2. This makes the model sensitive to outliers.
2. **Mathematical tractability**: Squared errors lead to a smooth, differentiable objective function with a unique minimum.
3. **Statistical optimality**: Under Gaussian noise, least squares is the maximum likelihood estimator.
4. **Geometric interpretation**: Minimizing RSS is equivalent to finding the projection onto the column space of X.
5. **No sign cancellation**: Unlike simple sum of errors (where positive and negative errors cancel), squaring ensures all errors contribute positively.

8.3 Matrix Form

In matrix-vector notation:

$$\text{RSS}(\beta) = \|y - X\beta\|^2 = (y - X\beta)^T(y - X\beta)$$

This is the squared Euclidean norm of the residual vector $r = y - X\beta$.

8.4 The Optimization Goal

Our objective is to find the parameters that minimize RSS:

$$\beta^* = \arg\min_{\beta} \text{RSS}(\beta) = \arg\min_{\beta} \|y - X\beta\|^2$$

Physical Interpretation: In the context of aircraft performance modeling, we're finding the model parameters that minimize the total squared difference between predicted and actual drag coefficients across all wind tunnel measurements. This gives us the "best fit" in a least-squares sense.

9 Derivation: Normal Equations

We can derive the optimal solution using calculus. The approach is to expand the objective function, take the derivative with respect to β, set it to zero, and solve.

9.1 Step 1: Expand the Objective Function

Starting with:

$$\text{RSS}(\beta) = (y - X\beta)^T(y - X\beta)$$

Expand:

$$\text{RSS}(\beta) = y^Ty - y^TX\beta - \beta^TX^Ty + \beta^TX^TX\beta$$

Since $y^TX\beta$ is a scalar, it equals its transpose: $y^TX\beta = \beta^TX^Ty$

Therefore:

$$\text{RSS}(\beta) = y^Ty - 2\beta^TX^Ty + \beta^TX^TX\beta$$

9.2 Step 2: Take Derivative with Respect to β

Using matrix calculus identities: - $\frac{\partial}{\partial\beta}(a^T\beta) = a$ - $\frac{\partial}{\partial\beta}(\beta^TA\beta) = 2A\beta$ (when A is symmetric)

We get:

$$\frac{\partial \text{RSS}}{\partial\beta} = -2X^Ty + 2X^TX\beta$$

9.3 Step 3: Set to Zero and Solve

At the minimum, the gradient must be zero:

$$-2X^T y + 2X^T X\beta^* = 0$$

Dividing by 2 and rearranging:

$$X^T X\beta^* = X^T y$$

These are called the **Normal Equations**.

9.4 Step 4: Closed-Form Solution

If $X^T X$ is invertible (which requires that the columns of X are linearly independent), we can solve for β^*:

$$\beta^* = (X^T X)^{-1} X^T y$$

This is the **ordinary least squares (OLS) solution**.

Verification that this is a minimum: The second derivative (Hessian matrix) is:

$$\frac{\partial^2 \text{RSS}}{\partial \beta^2} = 2X^T X$$

This is positive definite (assuming X has full column rank), confirming that we have found a minimum, not a maximum or saddle point.

10 Geometric Interpretation: Understanding the Error

10.1 What is the Residual Vector?

The residual (error) vector is defined as:

$$r = y - \hat{y} = y - X\beta$$

This is the vector of differences between: - **Observed data** y: What we actually measured - **Predictions** $\hat{y} = X\beta$: What our model predicts

10.2 Goal: Minimize the Length of the Error Vector

Our optimization problem can be stated as:

$$\min_{\beta} \|r\|^2 = \min_{\beta} \|y - X\beta\|^2$$

We want to make the residual vector as short as possible.

10.3 Column Space of X

Definition: The column space of X, denoted $\text{col}(X)$, is the set of all possible linear combinations of the columns of X:

$$\text{col}(X) = \{X\beta : \beta \in \mathbb{R}^{d+1}\}$$

Key Insight: - Every prediction $\hat{y} = X\beta$ **must lie in** $\text{col}(X)$ - The observed data y typically does **not** lie in $\text{col}(X)$ (due to measurement noise) - We seek the **best approximation** to y within $\text{col}(X)$

10.4 The Fundamental Geometric Principle

Question: What is the closest point in $\text{col}(X)$ to y?

Answer: The **orthogonal projection** of y onto $\text{col}(X)$.

Why? The **Projection Theorem** from linear algebra states: The shortest distance from a point to a subspace is achieved by the perpendicular projection onto that subspace.

Mathematical Statement: The error is minimized when it is orthogonal to the column space:

$$r \perp \text{col}(X) \quad \iff \quad X^T r = 0$$

This orthogonality condition is the **geometric essence** of least squares.

11 Why Projection Minimizes Error

11.1 Visualizing the Projection

Imagine a 2D plane (representing col(X)) embedded in 3D space (representing $\mathbb{R}^n$). The data vector y is a point not on this plane.

Key Observations:

1. Any prediction $\hat{y}$ must lie on the plane (in col(X))
2. The error $r = y - \hat{y}$ is the vector from $\hat{y}$ to y
3. We want to minimize $\|r\|$ (the length of this vector)

11.2 Why Perpendicular is Optimal

Consider any two points in col(X): - $\hat{y}_\perp$: The perpendicular projection - $\hat{y}_{\text{other}}$: Any other point

By the **Pythagorean theorem**:

$$\|y - \hat{y}_{\text{other}}\|^2 = \|y - \hat{y}_\perp\|^2 + \|\hat{y}_\perp - \hat{y}_{\text{other}}\|^2$$

Since $\|\hat{y}_\perp - \hat{y}_{\text{other}}\|^2 \geq 0$, we have:

$$\|y - \hat{y}_{\text{other}}\|^2 \geq \|y - \hat{y}_\perp\|^2$$

Therefore, the perpendicular projection gives the smallest error.

11.3 Mathematical Statement of Orthogonality

The optimal prediction $\hat{y}^*$ satisfies:

$$(y - \hat{y}^*) \perp \text{col}(X)$$

In matrix form:

$$X^T(y - \hat{y}^*) = 0$$

Understanding $X^T r = 0$:

- $X^T r$ is a vector of dot products: $[x_1^T r, x_2^T r, \ldots, x_{d+1}^T r]^T$
- Each dot product equals zero: $x_j^T r = 0$
- This means r is perpendicular to **every column** of X
- Therefore, r is perpendicular to the **entire column space**

12 Deriving the Optimal Solution via Projection

Rather than using calculus, we can derive the OLS solution directly from the geometric principle of orthogonal projection.

12.1 Step 1: State the Orthogonality Condition

From geometry, the error must be perpendicular to $\text{col}(X)$:

$$r \perp \text{col}(X) \quad \Longrightarrow \quad X^T r = 0$$

This is **the fundamental condition** for least squares optimality.

12.2 Step 2: Express in Terms of β

The optimal prediction is $\hat{y}^* = X\beta^*$ and the optimal residual is $r^* = y - \hat{y}^*$.

Substituting into the orthogonality condition:

$$X^T r^* = 0$$

$$X^T(y - \hat{y}^*) = 0$$

$$X^T(y - X\beta^*) = 0$$

12.3 Step 3: Derive the Normal Equations

Expanding:

$$X^T y - X^T X\beta^* = 0$$

Rearranging:

$$X^T X\beta^* = X^T y$$

These are the **Normal Equations**—exactly the same result we obtained using calculus!

12.4 Step 4: Solve for β^*

Assuming X^TX is invertible:

$$\beta^* = (X^TX)^{-1}X^Ty$$

Key Insight: We derived the OLS solution using **projection geometry** (orthogonality condition) instead of **calculus** (setting derivative to zero). Both paths lead to the same answer, providing two complementary perspectives:

- **Calculus view**: β^* minimizes the cost function
- **Geometric view**: β^* gives the perpendicular projection

13 The Projection Matrix

13.1 Definition

The **projection matrix** is defined as:

$$P = X(X^TX)^{-1}X^T$$

This matrix projects any vector in $\mathbb{R}^n$ onto $\text{col}(X)$.

13.2 Computing Predictions

Using the projection matrix, we can write the predicted values as:

$$\hat{y} = X\beta^* = X(X^TX)^{-1}X^Ty = Py$$

Interpretation: P directly maps observed data y to predictions $\hat{y}$, bypassing the need to explicitly compute β^*.

13.3 Key Properties

13.3.1 1. Idempotent (Projecting Twice = Projecting Once)

$$P^2 = P$$

Proof:

$$P^2 = X(X^TX)^{-1}X^TX(X^TX)^{-1}X^T = X(X^TX)^{-1}X^T = P$$

Interpretation: If a vector is already in $\text{col}(X)$, projecting it again doesn't change it.

13.3.2 2. Symmetric

$$P^T = P$$

Proof: Each factor is symmetric:

$$P^T = (X(X^TX)^{-1}X^T)^T = X((X^TX)^{-1})^TX^T = X(X^TX)^{-1}X^T = P$$

13.3.3 3. Projects onto col(X)

$$PX = X$$

Proof:

$$PX = X(X^TX)^{-1}X^TX = X$$

Interpretation: Any vector in col(X) is unchanged by the projection.

13.3.4 4. Residual Matrix

$$I - P$$

This matrix projects onto the orthogonal complement of col(X), giving the residuals:

$$r = (I - P)y$$

13.4 Aerospace Interpretation

In the context of drag coefficient prediction:

- Py: The component of observed drag that **can be explained** by our aerodynamic features (angle of attack, Mach number, etc.)
- $(I-P)y$: The component that **cannot be explained** (residual variance due to unmodeled effects, measurement noise, etc.)

The projection matrix decomposes the total variance into: - **Explained variance** (signal captured by the model) - **Unexplained variance** (noise or missing features)

14 When Direct Solution Fails

While the closed-form solution $\beta^* = (X^TX)^{-1}X^Ty$ is elegant, it faces several practical challenges.

14.1 Problem 1: Singular Matrix (Non-Invertibility)

When it occurs:

1. **More features than samples**: $n < d + 1$
    - Not enough data to uniquely determine all parameters
    - Infinitely many solutions exist
    - Example: 50 wind tunnel measurements but 100 polynomial features
2. **Multicollinearity**: Highly correlated features
    - Example: Including both velocity and Mach number when temperature is constant (they're perfectly correlated)
    - $X^T X$ becomes nearly singular (very small eigenvalues)
    - Small numerical errors can cause huge changes in solution

Consequences:

- Cannot compute $(X^T X)^{-1}$
- Solution is non-unique or unstable

14.2 Problem 2: Computational Cost

Matrix inversion complexity: $O(d^3)$ operations

For high-dimensional problems: - $d = 1000$ features: ~1 billion operations - $d = 10000$ features: ~1 trillion operations

Memory requirements: Storing $X^T X$ requires $O(d^2)$ memory

When it matters: - Large-scale machine learning (millions of features) - Online learning (real-time updates) - Embedded systems (limited computational resources)

14.3 Problem 3: Numerical Stability

Condition number: $\kappa(X^T X) = \frac{\sigma_{\max}}{\sigma_{\min}}$

- Large condition number $\rightarrow$ small perturbations in data cause large changes in solution
- Floating-point arithmetic errors accumulate
- Results become unreliable

Example: If $\kappa = 10^{10}$ and we use 64-bit floating point (16 digits precision), we may lose all significant digits in the solution.

14.4 Solutions

14.4.1 1. Regularization

Add a penalty term to prevent overfitting and improve conditioning:

Ridge Regression (L2):

$$\beta^* = (X^T X + \lambda I)^{-1} X^T y$$

Lasso (L1):

$$\min_{\beta} \|y - X\beta\|^2 + \lambda \|\beta\|_1$$

14.4.2 2. Gradient Descent

Iteratively update parameters without matrix inversion: - Complexity per iteration: $O(nd)$ instead of $O(d^3)$ - Can handle very large d - Works for any differentiable loss function

14.4.3 3. QR Decomposition

Numerically stable direct method:

$$X = QR$$

- More stable than normal equations - Still $O(nd^2)$ complexity

14.4.4 4. Singular Value Decomposition (SVD)

Most numerically stable method:

$$X = U\Sigma V^T$$

- Handles rank deficiency gracefully - Reveals multicollinearity - Gold standard for numerical stability

15 Gradient Descent: Iterative Approach

When direct solution is impractical, we can use iterative optimization methods. Gradient descent is the foundation of modern machine learning.

15.1 The Algorithm

Goal: Minimize $\text{RSS}(\beta) = \|y - X\beta\|^2$

Strategy: Start with an initial guess and repeatedly move in the direction of steepest descent.

15.1.1 Steps

1. **Initialize**: Choose starting parameters $\beta^{(0)}$ (typically random or zeros)
2. **Iterate**: For $t = 0, 1, 2, ...$ until convergence:

$$\beta^{(t+1)} = \beta^{(t)} - \eta \nabla_\beta \text{RSS}(\beta^{(t)})$$

   where $\eta > 0$ is the **learning rate** (step size)

3. **Stop**: When change is small enough or maximum iterations reached

15.2 Computing the Gradient

The gradient of RSS is:

$$\nabla_\beta \text{RSS} = \frac{\partial}{\partial \beta} \|y - X\beta\|^2 = -2X^T(y - X\beta)$$

Update Rule:

$$\beta^{(t+1)} = \beta^{(t)} + 2\eta X^T(y - X\beta^{(t)})$$

Intuition: - The gradient ∇RSS points in the direction of steepest **increase** - We move in the **opposite direction** (negative gradient) to decrease the error - The learning rate η controls how big a step we take

15.3 Convergence

Under appropriate conditions (convexity, suitable learning rate):

$$\beta^{(t)} \to \beta^* \quad \text{as} \quad t \to \infty$$

The method converges to the same solution as the direct method, but arrives there iteratively.

16 Gradient Descent Variants

Different variants of gradient descent trade off between convergence speed and computational cost per iteration.

16.1 Batch Gradient Descent

Use all n samples in each iteration:

$$\beta^{(t+1)} = \beta^{(t)} - \eta \nabla_\beta \text{RSS}(\beta^{(t)})$$

where the gradient uses all data:

$$\nabla_\beta \text{RSS} = -2X^T(y - X\beta^{(t)}) = -2\sum_{i=1}^{n} x_i(y_i - x_i^T\beta^{(t)})$$

Advantages:

- Stable, smooth convergence
- Gradient is exact (no sampling noise)
- Can use optimized linear algebra libraries

Disadvantages:

- Each iteration costs $O(nd)$
- Slow for very large datasets (n in millions)
- Entire dataset must fit in memory

When to use: Small to medium datasets, when you need stable convergence

16.2 Stochastic Gradient Descent (SGD)

Use one random sample i per iteration:

$$\beta^{(t+1)} = \beta^{(t)} + 2\eta x_i(y_i - x_i^T\beta^{(t)})$$

Advantages:

- Very fast updates: $O(d)$ per iteration
- Can process data online (streaming)
- May escape shallow local minima (for non-convex problems)
- Naturally handles huge datasets

Disadvantages:

- Noisy updates (high variance)
- Oscillates around minimum (never truly converges)
- Requires careful learning rate tuning
- May need learning rate decay

When to use: Very large datasets, online learning, when fast iteration is more important than stable convergence

16.3 Mini-Batch Gradient Descent

Use a random subset of b samples per iteration (typical: $b = 32, 64, 128, 256$):

$$\beta^{(t+1)} = \beta^{(t)} - \eta \frac{1}{b} \sum_{i \in \mathcal{B}_t} -2x_i(y_i - x_i^T \beta^{(t)})$$

where $\mathcal{B}_t$ is a random mini-batch at iteration t.

Advantages:

- **Best balance** between speed and stability
- Vectorized operations (GPU-friendly)
- Reduced variance compared to SGD
- Faster than batch for large n

Disadvantages:

- One more hyperparameter (batch size)
- Still some oscillation

When to use: **Default choice for modern machine learning**—combines benefits of both batch and stochastic methods

16.4 Comparison Summary

Method	Samples per Iteration	Cost per Iteration	Convergence	Best For
Batch	All (n)	$O(nd)$	Smooth, stable	Small datasets
Stochastic	1	$O(d)$	Noisy, fast	Huge datasets, online

Method	Samples per Iteration	Cost per Iteration	Convergence	Best For
Mini-batch	b (32-256)	$O(bd)$	Balanced	**Most applications**

17 Learning Rate Selection

The learning rate η is one of the most critical hyperparameters in gradient descent.

17.1 The Role of Learning Rate

In the update rule:

$$\beta^{(t+1)} = \beta^{(t)} - \eta \nabla_\beta \text{RSS}(\beta^{(t)})$$

η controls **how far** we move in the direction of the negative gradient.

17.2 Too Small Learning Rate ($\eta \ll 1$)

Symptoms:

- Very slow progress toward minimum
- Requires many iterations to converge
- May time out before reaching optimum

Cost: Wasted computation time

Example: If optimal $\eta = 0.01$ but we use $\eta = 0.0001$, we need $100\times$ more iterations.

17.3 Too Large Learning Rate ($\eta \gg 1$)

Symptoms:

- Overshooting the minimum
- Oscillation around optimum
- Divergence (error increases instead of decreasing)
- Instability

Cost: Never converges, wasted effort

Example: Parameters "bounce" past the minimum on each side, never settling down.

17.4 Finding the Right Learning Rate

Rule of thumb: Start with $\eta \in \{0.001, 0.01, 0.1, 1.0\}$ and tune based on training curves.

Grid search: Try multiple values, plot RSS vs. iteration, choose the largest η that converges smoothly.

Diagnostic: Plot $\text{RSS}^{(t)}$ vs. t: - Decreasing smoothly $\rightarrow$ good - Decreasing then oscillating $\rightarrow$ too large - Flat or barely decreasing $\rightarrow$ too small - Increasing $\rightarrow$ way too large

17.5 Adaptive Learning Rates

Modern optimization uses sophisticated learning rate strategies:

17.5.1 1. Learning Rate Decay

Decrease η over time:

$$\eta_t = \frac{\eta_0}{1 + kt}$$

or exponential decay:

$$\eta_t = \eta_0 e^{-kt}$$

Rationale: Start with large steps (fast progress), then small steps (fine-tuning near minimum)

17.5.2 2. Momentum

Use exponentially weighted moving average of gradients:

$$v^{(t)} = \gamma v^{(t-1)} + \eta \nabla_\beta \text{RSS}(\beta^{(t)})$$

$$\beta^{(t+1)} = \beta^{(t)} - v^{(t)}$$

Benefits:

- Accelerates convergence in relevant directions
- Dampens oscillations
- Helps escape plateaus

17.5.3 3. Adam (Adaptive Moment Estimation)

Most popular modern optimizer. Combines: - Momentum (first moment) - Adaptive learning rates per parameter (second moment)

Update rule (simplified):

$$m^{(t)} = \beta_1 m^{(t-1)} + (1-\beta_1) g^{(t)}$$
$$v^{(t)} = \beta_2 v^{(t-1)} + (1-\beta_2)(g^{(t)})^2$$
$$\beta^{(t+1)} = \beta^{(t)} - \eta \frac{m^{(t)}}{\sqrt{v^{(t)}} + \epsilon}$$

where $g^{(t)} = \nabla_\beta \text{RSS}(\beta^{(t)})$

Advantages:

- Works well with default parameters ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\eta = 0.001$)
- Adapts learning rate for each parameter
- Widely used in deep learning

17.5.4 4. Line Search

At each iteration, optimize the learning rate:

$$\eta_t = \arg\min_\eta \text{RSS}(\beta^{(t)} - \eta \nabla_\beta \text{RSS}(\beta^{(t)}))$$

Benefits: Guaranteed improvement at each step

Cost: Additional computation per iteration

18 Statistical Properties: Assumptions

The theoretical properties of linear regression rely on several key assumptions. Understanding these helps us know when the method will work well and when it might fail.

18.1 Classical Linear Regression Assumptions

18.1.1 1. Linearity

Assumption: The true relationship is linear in parameters:

$$y = X\beta_{\text{true}} + \epsilon$$

What it means: There exist true parameters β_{true} such that the data-generating process follows this form.

If violated:

- OLS estimates are biased
- Predictions may be systematically wrong
- **Solution**: Transform features, add polynomial or interaction terms, use nonlinear models

18.1.2 2. Independence

Assumption: Samples (x_i, y_i) are independent and identically distributed (i.i.d.).

What it means: - Each observation is drawn independently from the same distribution - Knowing one sample doesn't tell you about others

If violated:

- Time series data (autocorrelation)
- Spatial data (neighboring points are similar)
- Clustered data (measurements from same aircraft)

Consequences:

- Standard errors are wrong
- Confidence intervals unreliable
- **Solution**: Use time series models, spatial models, clustered standard errors

18.1.3 3. Homoscedasticity (Constant Variance)

Assumption: Error variance is constant across all observations:

$$\text{Var}(\epsilon_i) = \sigma^2 \quad \text{for all } i$$

What it means: The "noise level" is the same regardless of feature values.

Heteroscedasticity (violation): Variance depends on features

$$\text{Var}(\epsilon_i) = \sigma_i^2 \quad \text{(different for each } i\text{)}$$

Aerospace example:

- Sensor noise may increase with dynamic pressure
- Measurement error in drag coefficient may depend on angle of attack
- Flow unsteadiness increases near stall

If violated:

- OLS is still unbiased but not optimal (not minimum variance)
- Standard errors are wrong
- **Solution**: Weighted least squares, robust standard errors, transform the response

18.1.4 4. Normality

Assumption: Errors are normally distributed:

$$\epsilon_i \sim \mathcal{N}(0, \sigma^2) \quad \text{independently}$$

What it means: Random fluctuations follow a bell curve (Gaussian distribution).

Importance:

- Required for exact hypothesis tests and confidence intervals
- **Not required** for OLS estimation itself
- By Central Limit Theorem, normality becomes less critical for large n

If violated:

- OLS estimates still unbiased
- Hypothesis tests may be unreliable (especially for small n)
- **Solution**: Use robust inference, bootstrap, or transform data

18.1.5 5. No Perfect Multicollinearity

Assumption: Columns of X are linearly independent (full rank):

$$\text{rank}(X) = d + 1$$

What it means: No feature can be perfectly predicted by other features.

Perfect multicollinearity examples:

- Including both Celsius and Fahrenheit temperature
- Including velocity twice
- Sum of dummy variables equals 1 (dummy variable trap)

Near multicollinearity: Features are highly (but not perfectly) correlated

- Mach number and true airspeed (at constant altitude/temperature)
- Multiple polynomial terms of the same variable
- Multiple similar wind tunnel configurations

Consequences:

- $X^T X$ is singular or nearly singular
- Cannot compute $(X^T X)^{-1}$ or it's numerically unstable
- Coefficients have huge standard errors
- Small data changes cause large coefficient changes

Solutions:

- Remove redundant features
- Regularization (Ridge, Lasso)
- Principal Component Analysis (PCA)
- Domain knowledge to select meaningful features

19 Gauss-Markov Theorem: Implications for Practice

19.1 Statement of the Theorem

Gauss-Markov Theorem: Under assumptions 1-3 (linearity, independence, homoscedasticity), the OLS estimator $\beta^* = (X^T X)^{-1} X^T y$ is **BLUE** (Best Linear Unbiased Estimator).

19.2 What Does BLUE Mean?

19.2.1 Best

Minimum variance among all linear unbiased estimators.

What it means:

- OLS gives the most precise estimates possible (smallest uncertainty)
- No other linear unbiased method produces tighter confidence intervals
- In the space of linear unbiased estimators, OLS is optimal

Why it matters: For safety-critical aerospace applications, we want the most precise performance predictions possible.

19.2.2 Linear

Estimator is a linear function of the observations y:

$$\beta^* = Cy$$

for some matrix C (that may depend on X but not on y).

For OLS: $C = (X^T X)^{-1} X^T$

Why restrict to linear estimators?:

- Computationally tractable
- Well-understood statistical properties
- Note: Nonlinear estimators might have lower variance, but they're typically biased

19.2.3 Unbiased

Expected value equals true parameter:

$$E[\beta^*] = \beta_{\text{true}}$$

What it means:

- On average (over many datasets), estimates equal true values
- No systematic over- or under-estimation
- If we repeat the experiment many times, the average of our estimates converges to the truth

Why it matters: We want methods that are correct on average, not systematically biased.

19.3 Aerospace Context

Critical for certification:

Flight envelopes must be determined with: - **Minimal uncertainty** (Best) - **No systematic bias** (Unbiased)

The BLUE property ensures:

- Tightest possible bounds on performance predictions
- Maximum confidence in safety margins
- Regulatory compliance (FAA, EASA require unbiased, minimum-variance estimates)

Example: When certifying maximum operating Mach number, we need the most precise drag predictions possible. Using OLS (under appropriate assumptions) guarantees we're using the optimal estimator.

19.4 When BLUE Doesn't Apply

BLUE only applies under assumptions 1-3. If:

- **Nonlinearity**: True relationship is nonlinear → OLS is biased
- **Heteroscedasticity**: Variance is not constant → OLS is unbiased but not minimum variance (use weighted least squares)
- **Dependence**: Samples are correlated → Standard errors are wrong (use robust methods)

Additionally, BLUE is restricted to **linear unbiased** estimators. In modern machine learning:

- We often accept **bias** for reduced **variance** (bias-variance tradeoff)
- Regularization (Ridge, Lasso) introduces bias but reduces variance
- For prediction, biased estimators can outperform OLS

20 Statistical Properties: Distribution

Understanding the distribution of our parameter estimates allows us to quantify uncertainty and perform statistical inference.

20.1 Distribution of the OLS Estimator

Under the normality assumption ($\epsilon_i \sim \mathcal{N}(0, \sigma^2)$ independently), the OLS estimator has a multivariate normal distribution:

$$\beta^* \sim \mathcal{N}\left(\beta_{\text{true}}, \sigma^2 (X^T X)^{-1}\right)$$

20.2 Unpacking This Result

20.2.1 Mean (Expected Value)

$$E[\beta^*] = \beta_{\text{true}}$$

Unbiasedness: On average, our estimates equal the true parameters.

20.2.2 Covariance Matrix

$$\text{Cov}(\beta^*) = \sigma^2 (X^T X)^{-1}$$

What this tells us:

1. **Diagonal elements** give the variance of each coefficient:

$$\text{Var}(\beta_j^*) = \sigma^2 [(X^T X)^{-1}]_{jj}$$

2. **Off-diagonal elements** show correlations between coefficient estimates:

$$\text{Cov}(\beta_j^*, \beta_k^*) = \sigma^2 [(X^T X)^{-1}]_{jk}$$

3. **Depends on** X: The design of our experiment (which features, which samples) affects estimation precision
4. **Depends on** σ^2: More noise $\rightarrow$ more uncertainty

20.3 Practical Implications

This distribution allows us to:

1. **Construct confidence intervals** for each coefficient
2. **Perform hypothesis tests** about parameters
3. **Quantify uncertainty** in predictions
4. **Design experiments** to minimize variance (optimal experimental design)

20.4 What If Normality Doesn't Hold?

Good news: Even without normality:

- OLS estimates remain **unbiased** (under linearity, independence)
- OLS is still **BLUE** (under homoscedasticity)
- **Asymptotically** (large n), the Central Limit Theorem implies approximate normality

When it matters:

- Small samples ($n < 30$): normality important for exact inference
- Large samples ($n > 100$): approximate normality holds regardless
- **Solution for small samples**: Use robust inference methods, bootstrap

21 Estimating the Noise Variance

In practice, we don't know the true noise variance σ^2. We must estimate it from the data.

21.1 Residual Variance Estimator

$$\hat{\sigma}^2 = \frac{1}{n-d-1}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \frac{\text{RSS}}{n-d-1}$$

where: - $\text{RSS} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ is the residual sum of squares - n is the number of samples - d is the number of features (not counting intercept) - $d+1$ is the total number of parameters (including intercept)

21.2 Why Divide by $n - d - 1$?

21.2.1 Degrees of Freedom

Definition: Degrees of freedom (df) = number of independent pieces of information available for estimating variance.

Calculation: - Start with n observations - Estimate $d+1$ parameters ($\beta_0, \beta_1, \ldots, \beta_d$) - Each parameter estimate "uses up" one degree of freedom - Remaining df: $n - (d+1) = n - d - 1$

21.2.2 Why Not Divide by n?

If we divided by n, the estimator would be **biased**:

$$E\left[\frac{1}{n}\sum_{i=1}^{n} r_i^2\right] < \sigma^2$$

This **underestimates** the true variance because the residuals are constrained to satisfy the normal equations (they're not truly independent).

Dividing by $n - d - 1$ corrects this bias:

$$E[\hat{\sigma}^2] = E\left[\frac{\text{RSS}}{n-d-1}\right] = \sigma^2$$

This makes $\hat{\sigma}^2$ an **unbiased estimator** of σ^2.

21.3 Aerospace Example

Scenario: Fit drag coefficient model:

$$C_D = \beta_0 + \beta_1\alpha + \beta_2\alpha^2$$

using $n = 20$ wind tunnel data points.

Parameters: - Number of features: $d = 2$ (we have α and α^2) - Total parameters: $d + 1 = 3$ (including intercept β_0) - Degrees of freedom: $n - d - 1 = 20 - 2 - 1 = 17$

Variance estimate:

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^{20}(C_{D_i} - \hat{C}_{D_i})^2}{17}$$

21.4 Relationship to Model Fit

Small $\hat{\sigma}^2$: - Residuals are small - Model fits data well - Low prediction uncertainty

Large $\hat{\sigma}^2$: - Residuals are large - Model doesn't fit well or data is very noisy - High prediction uncertainty

22 Confidence Intervals for Coefficients

22.1 Understanding Coefficient Uncertainty

Each estimated coefficient β_j^* is a **random variable**—it depends on the random sample we collected. If we repeated the experiment, we'd get different data and different estimates.

Key question: How uncertain are we about each coefficient?

22.2 Standard Error

The **standard error** of β_j^* measures its sampling variability:

$$\text{SE}(\beta_j^*) = \sqrt{\hat{\sigma}^2 \left[(X^T X)^{-1}\right]_{jj}}$$

Interpretation: - Small SE → precise estimate (high confidence) - Large SE → imprecise estimate (low confidence)

Factors affecting SE:

1. **Data noise** ($\hat{\sigma}^2$): More noise → larger SE
2. **Sample size** (n): More data → smaller SE (typically SE $\propto 1/\sqrt{n}$)
3. **Feature correlation**: Correlated features → larger SE
4. **Feature variance**: Features with more spread → smaller SE

22.3 Confidence Interval Formula

A $100(1-\gamma)\%$ **confidence interval** for β_j is:

$$\beta_j^* \pm t_{\gamma/2, n-d-1} \cdot \text{SE}(\beta_j^*)$$

where $t_{\gamma/2,n-d-1}$ is the critical value from the **t-distribution** with $n-d-1$ degrees of freedom.

Common confidence levels:

- 95% confidence: $\gamma = 0.05$, so $t_{0.025,n-d-1}$
- 90% confidence: $\gamma = 0.10$, so $t_{0.05,n-d-1}$
- 99% confidence: $\gamma = 0.01$, so $t_{0.005,n-d-1}$

For large n: t-distribution approaches normal, $t_{\gamma/2,n-d-1} \approx z_{\gamma/2}$

- 95% CI: $z_{0.025} \approx 1.96 \approx 2$
- 90% CI: $z_{0.05} \approx 1.645$

22.4 Interpretation

What a 95% CI means:

> If we repeated the experiment many times and computed a 95% CI each time, approximately 95% of these intervals would contain the true parameter value β_j.

What it does NOT mean:

- The probability that β_j is in this interval is 95% (Bayesian interpretation)
- We are 95% sure our estimate is correct

22.5 Aerospace Example

Scenario: Fitting lift coefficient model:

$$C_L = \beta_0 + \beta_1 \alpha$$

Results: - $\beta_1^* = 0.105$ (per degree) - $\text{SE}(\beta_1^*) = 0.004$ - $n = 30$, $d = 1$, so df $= 28$ - $t_{0.025,28} \approx 2.048$

95% Confidence Interval:

$$0.105 \pm 2.048 \times 0.004 = 0.105 \pm 0.008 = [0.097, 0.113]$$

Interpretation: We are 95% confident the true lift slope is between 0.097 and 0.113 per degree.

Engineering decision: This precision allows us to set conservative performance bounds for flight envelope calculations.

23 Hypothesis Testing for Individual Coefficients

23.1 The Central Question

After estimating coefficients from data, we ask: **Is this coefficient significantly different from zero, or could it just be noise?**

This is crucial for:

- **Feature selection**: Which features actually matter?
- **Model simplification**: Can we remove unimportant features?
- **Physical interpretation**: Is this effect real or spurious?

23.2 Setting Up the Test

23.2.1 Hypotheses

Null hypothesis H_0:

$$\beta_j = 0$$

Interpretation: Feature j has **no true effect** on the response. Any non-zero estimate we observed is just random sampling variation.

Alternative hypothesis H_1:

$$\beta_j \neq 0$$

Interpretation: Feature j **does affect** the response. The estimated effect is real, not just noise.

23.3 The Test Statistic

We construct a **t-statistic**:

$$t_j = \frac{\beta_j^*}{\text{SE}(\beta_j^*)} = \frac{\text{Estimated coefficient}}{\text{Standard error of estimate}}$$

Interpretation: - Measures **how many standard errors** the estimate is away from zero - If $\beta_j = 0$ truly, we expect $\beta_j^* \approx 0$ (within sampling error) - Large $|t_j|$ means the estimate is far from zero $\rightarrow$ unlikely if H_0 is true

23.4 Distribution Under the Null

If H_0 is true ($\beta_j = 0$), then:

$$t_j \sim t_{n-d-1}$$

The test statistic follows a **t-distribution** with $n - d - 1$ degrees of freedom.

This allows us to compute **p-values** and make decisions.

23.5 Making the Decision

23.5.1 Approach 1: P-value

P-value: The probability of observing a test statistic as extreme as (or more extreme than) what we got, if the null hypothesis were true.

$$p\text{-value} = P(|t| \geq |t_j| \mid H_0 \text{ is true})$$

Decision rule: - If $p < \alpha$ (significance level, typically 0.05): **Reject** H_0 (coefficient is significant) - If $p \geq \alpha$: **Fail to reject** H_0 (insufficient evidence that coefficient differs from zero)

Interpretation of p-values:

- $p < 0.001$: Very strong evidence against H_0 (highly significant)
- $p < 0.01$: Strong evidence
- $p < 0.05$: Moderate evidence (conventional cutoff)
- $p > 0.05$: Weak or no evidence

23.5.2 Approach 2: Critical Value

Critical value: The threshold $t_{\text{crit}} = t_{\alpha/2, n-d-1}$ such that:

$$P(|t| > t_{\text{crit}} \mid H_0) = \alpha$$

Decision rule: - If $|t_j| > t_{\text{crit}}$: **Reject** H_0 - If $|t_j| \leq t_{\text{crit}}$: **Fail to reject** H_0

Equivalence: Both approaches give the same conclusion. The p-value approach is more informative because it quantifies the strength of evidence.

23.6 Aerospace Example

Scenario: Testing whether Reynolds number affects drag coefficient in our model:

$$C_D = \beta_0 + \beta_1 \alpha + \beta_2 \alpha^2 + \beta_3 \log(\text{Re})$$

Results for β_3:

- $\beta_3^* = -0.0015$
- $\text{SE}(\beta_3^*) = 0.0008$

- $n = 50$, $d = 3$, df $= 46$

Test statistic:

$$t_3 = \frac{-0.0015}{0.0008} = -1.875$$

P-value (two-tailed, df $= 46$):

$$p \approx 0.067$$

Decision at $\alpha = 0.05$:

- $p = 0.067 > 0.05$: Fail to reject H_0
- **Conclusion**: Insufficient evidence that Reynolds number significantly affects drag (at 5% significance level)

Engineering interpretation:

- We might consider **removing** log(Re) from the model to simplify it
- Or acknowledge that Reynolds number effects are weak in our data range
- Or collect more data to increase statistical power

23.7 Important Caveats

1. **Statistical significance practical significance**: A tiny effect can be statistically significant with enough data
2. **Multiple testing**: Testing many coefficients increases false positive rate (use Bonferroni correction or similar)
3. **Correlation**: In models with many correlated features, individual t-tests can be misleading
4. **Model assumptions**: These tests assume normality, homoscedasticity, etc.

24 Summary

Linear regression is a powerful and versatile tool for aerospace engineering applications. Key takeaways:

24.1 Mathematical Foundations

- Linear regression models relationships as $y = X\beta + \epsilon$
- "Linear" refers to parameters, not features—we can model nonlinear relationships
- The optimal solution minimizes squared error: $\min \|y - X\beta\|^2$

24.2 Two Solution Approaches

1. **Closed-form** (Normal Equations): $\beta^* = (X^T X)^{-1} X^T y$
    - Exact, one-step solution
    - Requires matrix inversion
    - Can fail for large d or singular $X^T X$
2. **Gradient Descent**: Iterative optimization
    - Works for any problem size
    - Requires tuning (learning rate, iterations)
    - Foundation of modern machine learning

24.3 Geometric Insight

- OLS finds the **orthogonal projection** of y onto $\text{col}(X)$
- Residuals are **perpendicular** to feature space: $X^T r = 0$
- Projection matrix $P = X(X^T X)^{-1} X^T$ separates signal from noise

24.4 Statistical Properties

- Under standard assumptions, OLS is **BLUE** (Best Linear Unbiased Estimator)
- Coefficient distribution: $\beta^* \sim \mathcal{N}(\beta_{\text{true}}, \sigma^2 (X^T X)^{-1})$
- Enables **confidence intervals** and **hypothesis tests**

24.5 Practical Considerations

- Check assumptions (linearity, independence, homoscedasticity, normality)
- Watch for multicollinearity
- Use appropriate validation metrics
- Consider regularization for high-dimensional problems

24.6 Aerospace Applications

Linear regression is essential for:

- Performance prediction (drag, lift, fuel consumption)
- Wind tunnel data analysis
- Flight test analysis
- System identification
- Model validation and uncertainty quantification

The techniques you've learned form the foundation for more advanced machine learning methods used throughout aerospace engineering.