Project: Ballistic Delivery of Humanitarian Aid

Can be completed in groups of up to 4 people. Each part is worth 25%.

Motivation

The idea of using ballistic projectiles to deliver humanitarian aid has been suggested at various points in the last 60 years. Here are some examples:

- Project HARP (High Altitude Research Project) in the 1960s explored high-altitude projectile launches, although it was mainly for scientific purposes.
- SpaceX's Starship: SpaceX has suggested that Starship could be used for rapid point-to-point Earth transport, potentially applicable for emergency aid delivery.
- Rocket Lab: Proposed concepts for delivering small payloads using rockets to hard-to-reach areas.

In this project you will model ballistic trajectories on Earth, and use your numerical models to analyze several situations.

Objectives

This project involves predicting and optimizing projectile trajectories launched from Earth's surface. The assignment is divided into three distinct parts:

1. **Numerical simulation of projectile trajectory**
2. **Trajectory optimization**
3. **Predictive modeling**

Each part will build upon the previous one, using computational methods relevant to aerospace and mechanical engineering.

You will turn in a written report (think of it as a lab report), detailing how you approached the problem, and your results from the assigned tasks.

Your group will also produce a video of less than 5 minutes describing the project and your results. The video has the goal of convincing someone to trust your results and to use your approach when they are cooking. It could include a demonstration of using the approach.

This project is purposely less structured that a typical homework assignment. Where the problem statement is ambiguous, you should use your judgment.

Objectives

This project involves predicting and optimizing projectile trajectories launched from Earth's surface. The assignment is divided into five distinct parts:

1. **Numerical simulation of projectile trajectory**
2. **Trajectory optimization**

3. **Predictive modeling**
4. **Radar Coverage and Ballistic projectile Detection Modeling**
5. **Visualization of Projectile and Radar Detection Volume**

Each part will build upon the previous one, using computational methods relevant to aerospace engineering.

Notation and Definitions

- Latitude ϕ [degrees]
- Longitude λ [degrees]
- Altitude h [m]
- Launch velocity v_0 [m/s]
- Launch elevation angle θ [degrees above horizontal]
- Launch azimuth angle ψ [degrees from geographic north, clockwise]
- Earth radius (WGS84 standard): $R_{Earth} = 6378137$ [m]
- Gravitational acceleration: $g \approx 9.81$ [m/s^2], adjusted for altitude as needed
- Earth rotation rate: $\omega_{Earth} = 7.2921159 \times 10^{-5}$ [rad/s]

Coordinate systems:

- Earth-Centered Earth-Fixed (ECEF) coordinates: Cartesian coordinates centered at Earth's center.
- East-North-Up (ENU): Local tangent plane coordinates at launch site.

Part 1: Numerical Simulation of Projectile Trajectory

Goal: Numerically integrate the equations of motion to predict projectile landing location given initial conditions.

Tasks:

- Convert launch latitude/longitude/altitude to ECEF coordinates.
- Transform initial velocity from ENU coordinates to ECEF coordinates.
- Include gravitational, drag, Coriolis, and centrifugal forces in the equations of motion.
- Numerically integrate using Python (`solve_ivp`).
- Identify the projectile impact point (ECEF coordinates) and convert back to latitude/longitude.

Deliverable: Python code implementing these steps, verified by test cases using the great circle trajectory. Grading will take into account how realistic the density and gravitational acceleration varies along the trajectory.

Part 2: Trajectory Optimization

Goal: Using optimization, find the launch angles (θ, ψ) and minimal initial speed v_0 required for the shortest path to land exactly at a specified target latitude/longitude.

Tasks:

- Formulate optimization problem clearly, defining objective function (minimum flight path length).
- Use numerical optimization methods (e.g., `scipy.optimize.minimize`).
- Clearly define constraints (impact latitude/longitude accuracy within $\pm 0.001°$).

Deliverable: Optimization code, results for the target locations in Table 1. The projectiles should be launched from locations in Table 2 so that the total time the projectiles are in the air is minimized. Each launch location can launch up to 4 projectiles.

Aid delivery location	**Latitude (N)**	**Longitude (E)**
Dombarovsky (Yasny, Orenburg Oblast)	51.1100°	59.8400°
Kozelsk (Kaluga Oblast)	54.0340°	35.7800°
Tatishchevo (Saratov Oblast)	51.6670°	45.1170°
Uzhur (Krasnoyarsk Krai)	55.3330°	89.8170°
Teykovo (Ivanovo Oblast)	56.8510°	40.5320°
Yoshkar-Ola (Mari El Republic)	56.6330°	47.8710°
Nizhny Tagil (Sverdlovsk Oblast)	58.0830°	60.0500°
Novosibirsk (Novosibirsk Oblast)	55.3330°	83.0000°
Irkutsk (Irkutsk Oblast)	52.3170°	104.2330°

Table 1: Latitude and Longitude Coordinates of Aid Delivery Locations.

Missile Base	**Latitude (N)**	**Longitude (W)**
Francis E. Warren AFB (Wyoming)	41.1265°	-104.8668°
Minot AFB (North Dakota)	48.41583°	-101.35806°
Malmstrom AFB (Montana)	47.5028°	-111.1857°

Table 2: Latitude and Longitude Coordinates of U.S. Aid Launch Silos.

Part 3: Predictive Modeling

Goal: Develop a predictive model (machine learning or physics-informed model) that estimates projectile landing position based on given velocity and location measurements at intermediate flight times.

Tasks:

- Generate training data by varying launch conditions and numerically integrating trajectories.
- Train a machine learning model (e.g., neural network) or develop a simplified physical model.
- Validate your approach using trajectories in Part 2

Deliverable: Predictive model implementation, performance analysis, and validation report.

Part 4: Radar Coverage and Ballistic projectile Detection Modeling

Goal: Model the detection volume and coverage of a military-grade radar station in Siberia, designed to detect ballistic projectiles trajectories. The objective is to determine the time of detection for an incoming trajectory launched from Minot AFB and estimate the position of the projectile when its trajectory is confirmed as a ballistic projectile.

This will utilize your model from Part 3.

Tasks:

- Define Radar Characteristics:
  - Radar Location: Choose two realistic radar locations in Siberia, such as near Norilsk (69.3558° N, 88.1893° E) or Yakutsk (62.0355° N, 129.6755° E).
  - Radar Range: Assume a maximum detection range of R_{max}, typically 300–400 km for long-range ballistic missile defense radars.
  - Beamwidth and Coverage:
    - ∗ Azimuth coverage: 360° for full spherical detection or defined sector coverage.
    - ∗ Elevation coverage: Up to 90° for high-altitude objects.
- Simulate Trajectory from Minot AFB:
  - Define launch location: Minot AFB (48.4158° N, 101.3584° W) with a target location of (55.3330° N, 83.0000° E.)
  - Simulate an incoming projectile trajectory using the equations of motion from Part 1.
  - Track the projectile's position relative to the radar location in Siberia.
- Determine Time of Detection and Confirmation:
  - Model the radar detection volume and calculate the point at which the projectile enters the radar's detection range.
  - Estimate the time required to classify the incoming object as a ballistic projectile and determine its landing location.
  - Compute where the projectile is located at the time of confirmation.
- Calculate Detection Area and Volume:
  - Calculate 2D detection area:
$$A = \pi R_{max}^2$$
  - Calculate 3D detection volume:
$$V = \frac{4}{3}\pi R_{max}^3 \times \frac{\alpha}{360^\circ} \times \frac{\beta}{360^\circ}$$
- Estimate Horizon Range and Earth Curvature Effects:
  - Account for the curvature of the Earth using the horizon range equation:
$$R_h = \sqrt{2R_{Earth}h} + h$$

Deliverable:

- Python code that simulates radar detection volume, models detection and confirmation of an incoming trajectory from Minot AFB, and estimates the projectile's location at confirmation.
- Visualization of radar coverage and incoming projectile trajectory.
- Analysis report detailing detection time, classification duration, and estimated projectile position.
- If the projectile were headed to 39.6°N, 124.7°E, would the radar detect that projectile. If so, how long would it take to confirm that the landing site is south of 40º N and east of 130ºE?

Visualization Requirements

Goal: In each part you should visualize trajectories of the projectile flying around the globe, including a map overlay, and visualize the radar detection volume in a similar 3D representation.

Tasks:

- 3D Visualization of Projectile Trajectory:
  - Plot the projectile trajectory on a 3D globe.
  - Overlay a world map on the globe to show realistic geography.
  - Show launch and impact points with markers.
- Visualize Radar Detection Volume:
  - Display the 3D radar detection volume using a spherical sector.
  - Include radar beamwidth, azimuth, and elevation coverage.
  - Show radar location and visualize the detected trajectory relative to the radar's position.
- Bonus Task:
  - Add animation to show the projectile's flight path dynamically over time.

Deliverable:

- Python code that generates 3D visualizations of projectile trajectory and radar detection volume.

In short, you be judged on the quality of visualizations and animations illustrating the flight path and radar coverage, as necessary.

Submission Requirements

- Organized Python scripts or notebooks for each part.
- Detailed report including mathematical formulations, assumptions, code documentation, and results discussion.
- A 5 minute video presenting your results discussing specifically how you do the detection and determination of projectile's landing location.

Appendix: Equations and Coordinate Transformations

Equations of Motion

The position vector $\mathbf{r}$ is defined in ECEF coordinates as:

$$\mathbf{r} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

where X, Y, Z are the Cartesian coordinates in the Earth-Centered Earth-Fixed (ECEF) system.

$$m\frac{d^2\mathbf{r}}{dt^2} = m\mathbf{g}(\mathbf{r}) + \mathbf{F}_{drag}(\mathbf{v}) - 2m\omega_{Earth} \times \frac{d\mathbf{r}}{dt} - m\omega_{Earth} \times (\omega_{Earth} \times \mathbf{r})$$

Drag Force Model

The drag force is given by:

$$\mathbf{F}_{drag}(\mathbf{v}) = -\frac{1}{2}C_D \rho A v^2 \hat{v}$$

where:

- C_D: Drag coefficient (depends on projectile shape and Mach number)
- ρ: Air density, modeled as a function of altitude
- A: Cross-sectional area of the projectile
- v: Magnitude of the velocity vector
- $\hat{v}$: Unit vector in the direction of velocity

Drag Force on a Conical Projectile

For a conical projectile, the drag coefficient C_D is approximated by:

$$C_D = C_{D0} + \frac{C_{D1}}{\sqrt{M^2 - 1}} \text{ for supersonic flow}$$

where:

- C_{D0} and C_{D1} are empirical coefficients
- M is the Mach number, defined as $M = \frac{v}{c}$, where c is the speed of sound

Reasonable numbers are $C_{D0} \approx 0.5$ to 0.7 and $C_{D1} = 0.7$ to 1.1. The cross-sectional area should be in the range 1.5 - 3 m^2.

Coordinate Transformations

Geodetic to ECEF:

$$X = (N + h)\cos\phi\cos\lambda, \quad Y = (N + h)\cos\phi\sin\lambda, \quad Z = \big(N(1 - e^2) + h\big)\sin\phi$$

where $N = \frac{a}{\sqrt{1-e^2\sin^2\phi}}$, $a = 6378137$ m, and $e = 0.08181919$.

ECEF to Geodetic: To convert ECEF coordinates back to geodetic coordinates:

$$\phi = \tan^{-1}\left(\frac{Z(1 - e^2 N)}{\sqrt{X^2 + Y^2}}\right), \quad \lambda = \tan^{-1}\left(\frac{Y}{X}\right), \quad h = \frac{\sqrt{X^2 + Y^2}}{\cos\phi} - N$$

Local Rotation Matrices

To transform from ENU to ECEF:

$$\begin{bmatrix} X_{ECEF} \\ Y_{ECEF} \\ Z_{ECEF} \end{bmatrix} = \begin{bmatrix} -\sin\lambda & -\sin\phi\cos\lambda & \cos\phi\cos\lambda \\ \cos\lambda & -\sin\phi\sin\lambda & \cos\phi\sin\lambda \\ 0 & \cos\phi & \sin\phi \end{bmatrix} \begin{bmatrix} E \\ N \\ U \end{bmatrix}$$

where E, N, U are the local ENU velocities.

ECEF to ENU:

$$\begin{bmatrix} E \\ N \\ U \end{bmatrix} = \begin{bmatrix} -\sin\lambda & \cos\lambda & 0 \\ -\sin\phi\cos\lambda & -\sin\phi\sin\lambda & \cos\phi \\ \cos\phi\cos\lambda & \cos\phi\sin\lambda & \sin\phi \end{bmatrix} \begin{bmatrix} X_{ECEF} - X_0 \\ Y_{ECEF} - Y_0 \\ Z_{ECEF} - Z_0 \end{bmatrix}$$

where X_0, Y_0, Z_0 are the ECEF coordinates of the origin.

Earth Rotational Velocity Vector

The rotational velocity of the Earth in ECEF coordinates is given by:

$$\omega_{Earth} = \begin{bmatrix} 0 \\ 0 \\ \omega_{Earth} \end{bmatrix}$$

where $\omega_{Earth} = 7.2921159 \times 10^{-5}$ rad/s is the Earth's rotation rate.

Radar Detection and Tracking Parameters

To accurately define the detection process:

- **Azimuth Angle** α: The angle measured clockwise from geographic north to the projection of the radar beam on the ground plane.
- **Elevation Angle** β: The angle between the radar beam and the local horizontal plane, ranging from 0° to 90°.
- **Range** R: The distance from the radar to the target.
- **Altitude** h: The height of the target above the Earth's surface.
- **Horizon Range** R_h: The distance to the horizon from the radar, accounting for Earth's curvature.

Transforming Initial Velocity from ENU to ECEF

Coordinate Systems Overview

- **ENU (East-North-Up) Coordinate System:**
  - Origin: Located at the launch site (latitude ϕ and longitude λ).
  - **East (E):** Tangent to the Earth's surface, points toward increasing longitude.
  - **North (N):** Tangent to the Earth's surface, points toward increasing latitude.
  - **Up (U):** Normal to the surface, points radially outward from the Earth.
- **ECEF (Earth-Centered Earth-Fixed) Coordinate System:**
  - Origin: The center of the Earth.
  - X-axis: Points toward the intersection of the equator and the prime meridian.
  - Y-axis: Perpendicular to the X-axis, lies in the equatorial plane.
  - Z-axis: Aligned with the Earth's rotation axis, pointing toward the North Pole.

Define Initial ENU Velocity

The initial velocity in ENU coordinates is given by:

$$\mathbf{v}_{ENU} = \begin{bmatrix} v_E \\ v_N \\ v_U \end{bmatrix}$$

where:

- v_E is the velocity component along the eastward direction.
- v_N is the velocity component along the northward direction.
- v_U is the velocity component in the upward direction.

Rotation Matrix: ENU to ECEF

To transform the velocity vector from ENU to ECEF coordinates, apply the following rotation matrix:

$$\mathbf{R}_{ENU \to ECEF} = \begin{bmatrix} -\sin\lambda & -\sin\phi\cos\lambda & \cos\phi\cos\lambda \\ \cos\lambda & -\sin\phi\sin\lambda & \cos\phi\sin\lambda \\ 0 & \cos\phi & \sin\phi \end{bmatrix}$$

where:

- ϕ is the **latitude** of the launch site.
- λ is the **longitude** of the launch site.

Transformed Velocity Vector in ECEF

The initial velocity in ECEF coordinates is computed as:

$$\mathbf{v}_{ECEF} = \mathbf{R}_{ENU \to ECEF}\mathbf{v}_{ENU}$$

$$\mathbf{v}_{ECEF} = \begin{bmatrix} -\sin\lambda & -\sin\phi\cos\lambda & \cos\phi\cos\lambda \\ \cos\lambda & -\sin\phi\sin\lambda & \cos\phi\sin\lambda \\ 0 & \cos\phi & \sin\phi \end{bmatrix} \begin{bmatrix} v_E \\ v_N \\ v_U \end{bmatrix}$$

Expanded Form of ECEF Velocity Components

$$\begin{aligned} v_X &= -\sin\lambda \cdot v_E - \sin\phi\cos\lambda \cdot v_N + \cos\phi\cos\lambda \cdot v_U \\ v_Y &= \cos\lambda \cdot v_E - \sin\phi\sin\lambda \cdot v_N + \cos\phi\sin\lambda \cdot v_U \\ v_Z &= \cos\phi \cdot v_N + \sin\phi \cdot v_U \end{aligned}$$

Example Calculation

Consider a projectile launched from:

- Latitude: $\phi = 45°$
- Longitude: $\lambda = 90°$
- Initial velocity in ENU: $\mathbf{v}_{ENU} = [500, 0, 100]$ m/s

This corresponds to:

- 500 m/s due east.

- 0 m/s due north.
- 100 m/s upward.

The transformed ECEF velocity vector is obtained using the rotation matrix:

$$\mathbf{v}_{ECEF} = \mathbf{R}_{ENU\to ECEF}\mathbf{v}_{ENU}$$

$$\mathbf{R}_{ENU\to ECEF} = \begin{bmatrix} -\sin 90^\circ & -\sin 45^\circ \cos 90^\circ & \cos 45^\circ \cos 90^\circ \\ \cos 90^\circ & -\sin 45^\circ \sin 90^\circ & \cos 45^\circ \sin 90^\circ \\ 0 & \cos 45^\circ & \sin 45^\circ \end{bmatrix}$$

Summary of the ENU to ECEF Transformation

To define the initial velocity correctly in ECEF coordinates:

- Apply the rotation matrix $\mathbf{R}_{ENU\to ECEF}$.
- Ensure that the initial position and velocity are expressed in the ECEF frame before integrating the equations of motion.