Institute of Geodesy and Geoinformation für Geodäsie und Geoinformation
University of Bonn

December 22, 2025

Course: Coordinate Systems
Homework Assignment 3 - Radar Altimeter

Sandesh Pokhrel

---

1 Task 3.1

1.1 Task 3.1 Q.No: 1

Given are the Cartesian coordinates of a satellite at time t,

$$\mathbf{x}_{\text{sat}} = \begin{bmatrix} x_{\text{sat}} \\ y_{\text{sat}} \\ z_{\text{sat}} \end{bmatrix} = \begin{bmatrix} 4\,831\,342.4634 \text{ m} \\ 2\,833\,965.0779 \text{ m} \\ 5\,289\,590.6351 \text{ m} \end{bmatrix}.$$

To convert Cartesian Coordinates into Ellipsoidal coordinates, we know;

Algorithm for Conversion of Cartesian Coordinates into Ellipsoidal Coordinates

Given: Cartesian coordinates (x, y, z) and ellipsoid parameters (a, b).

Step 1: Longitude

$$\lambda = \arctan 2(y, x)$$

Step 2: First eccentricity squared

$$e^2 = \frac{a^2 - b^2}{a^2}$$

Step 3: Auxiliary quantity

$$p = \sqrt{x^2 + y^2}$$

Step 4: Initial values

$$h^{(0)} = 0, \qquad \varphi^{(0)} = \arctan\left(\frac{z}{p(1 - e^2)}\right)$$

Step 5: Iteration
For $k = 1, 2, \dots$ repeat until convergence:

$$N^{(k)} = \frac{a}{\sqrt{1 - e^2 \sin^2 \varphi^{(k-1)}}}$$

$$h^{(k)} = \frac{p}{\cos \varphi^{(k-1)}} - N^{(k)}$$

$$\varphi^{(k)} = \arctan\left(\frac{z}{p\left(1 - \frac{e^2 N^{(k)}}{N^{(k)} + h^{(k)}}\right)}\right)$$

Output: Ellipsoidal longitude λ, latitude φ, and height h.

Answer :
For Topex/Poseidon Ellipsoid, the parameters are as follows:

Topex/Poseidon Ellipsoid

The parameters of the Topex/Poseidon (T/P) ellipsoid are given by

$$a = 6\,378\,136.3 \text{ m}, \qquad f = \frac{1}{298.257}.$$

The semi-minor axis is computed as

$$b = a(1 - f).$$

The first eccentricity squared is defined as

$$e^2 = 1 - \frac{b^2}{a^2}.$$

Now; Computing the Ellipsoidal Coordinates;

Step 1: Longitude

$$\lambda = arctan(\frac{y}{x})$$

$$\lambda = arctan(\frac{2833965.0779}{4831342.4634})$$

$$\lambda = 30.395°$$

Step 2: First eccentricity squared

$$e^2 = \frac{a^2 - b^2}{a^2}$$

$For, Topex/PoseidonEllipsoid,$

$$a = 6378136.3m(Semi - MajorAxis)$$

$$f = \frac{1}{298.257}(Flattening)$$

$$b = a(1 - f) = 6378136.3 * (1 - \frac{1}{298.257})(Semi - MinorAxis)$$

$$b = 6356751.601m$$

$Hence;$

$$e^2 = \frac{6378136.3^2 - 6356751.601^2}{6378136.3^2}$$

$$e^2 = 0.00669438$$

Step 3 : Auxiliary quantity

$$p = \sqrt{x^2 + y^2}$$

$$p = \sqrt{4831342.4634^2 + 2833965.0779^2}$$

$$p = 5601180.952m$$

Step 4: Initial values

$$h^{(0)} = 0, \qquad \varphi^{(0)} = \arctan\left(\frac{z}{p(1-e^2)}\right)$$

$$\varphi^{(0)} = \arctan\left(\frac{5289590.6351}{5601180.952(1-0.00669438)}\right)$$

$$\varphi^{(0)} = 43.55333^\circ$$

Step 5: Iteration

Table 1: Iteration results for Cartesian to ellipsoidal coordinate conversion (five iterations)

Iteration k	Longitude (deg)	Latitude (deg)	Height (m)
1	30.395000005	43.519912315	1 340 313.531
2	30.395000005	43.520000256	1 336 043.804
3	30.395000005	43.520000024	1 336 055.029
4	30.395000005	43.520000025	1 336 055.000
5	30.395000005	43.520000025	1 336 055.000

Hence, the final results after the 5 iterations are as follows;

$$Longitude(\lambda) = 30.395000005^\circ$$

$$Latitude(\phi) = 43.520000025^\circ$$

$$Height(H) = 1\,336\,055.000m$$

1.1 MATLAB Code for Task 3.1. Q.No:1

```
% Cartesian to Ellipsoidal Coordinates
% Topex/Poseidon Ellipsoid

clc; clear;

% Given Cartesian coordinates
x = 4831342.4634;   % m
y = 2833965.0779;   % m
z = 5289590.6351;   % m

% Ellipsoid parameters
```

```
a = 6378136.3;
f = 1 / 298.257000;
b = a * (1 - f);
e2 = 1 - (b^2 / a^2);

% Longitude
lambda = atan2(y, x);
lambda_deg = rad2deg(lambda);

% Auxiliary quantity
p = sqrt(x^2 + y^2);

% Initial values
phi = atan(z / (p * (1 - e2)));
h = 0;

% Print header
fprintf('\nIter | Longitude (deg) | Latitude (deg) | Height (m)\n');
fprintf('------------------------------------------------------\n');

% For 5 iterations
for k = 1:5

    % Radius of curvature
    N = a / sqrt(1 - e2 * sin(phi)^2);

    % Height update
    h = p / cos(phi) - N;

    % Latitude update
    phi = atan( z / ( p * (1 - (e2 * N) / (N + h)) ) );

    % Print iteration result
    fprintf('%4d | %15.9f | %14.9f | %10.3f\n', ...
        k, lambda_deg, rad2deg(phi), h);
end

% Final result after 5 iterations
fprintf('\n============= FINAL RESULT (k = 5) =============\n');
fprintf('Longitude  = %.9f deg\n', lambda_deg);
fprintf('Latitude   = %.9f deg\n', rad2deg(phi));
fprintf('Height     = %.3f m\n', h);
```

1.2 Task 3.1. Q.No: 2

In this task, the ellipsoidal height of the satellite is recomputed using the GRS80 ellipsoid and compared with the result obtained using the Topex/Poseidon (T/P) ellipsoid. The Cartesian coordinates of the satellite remain unchanged; only the reference ellipsoid parameters are modified.

Ellipsoid parameters

- **Topex/Poseidon ellipsoid:**

$$a_{TP} = 6\,378\,136.3 \text{ m}, \qquad f_{TP} = \frac{1}{298.257}$$

- **GRS80 ellipsoid:**

$$a_{GRS80} = 6\,378\,137.0 \text{ m}, \qquad f_{GRS80} = \frac{1}{298.257222101}$$

Iterative computation of ellipsoidal height

The ellipsoidal latitude and height are computed iteratively for both ellipsoids using the same Cartesian satellite coordinates. The iteration procedure converges rapidly, as shown by the intermediate results.

Table 2: Iteration results for the Topex/Poseidon ellipsoid

Iteration k	Latitude (deg)	Height (m)
1	43.519912315	1 340 313.531
2	43.520000256	1 336 043.804
3	43.520000024	1 336 055.029
4	43.520000025	1 336 055.000
5	43.520000025	1 336 055.000

Table 3: Iteration results for the GRS80 ellipsoid

Iteration k	Latitude (deg)	Height (m)
1	43.519912214	1 340 312.819
2	43.520000155	1 336 043.097
3	43.519999923	1 336 054.323
4	43.519999924	1 336 054.294
5	43.519999924	1 336 054.294

Final comparison

After five iterations, the converged ellipsoidal heights are:

$$h_{TP} = 1\,336\,055.000 \text{ m}, \qquad h_{GRS80} = 1\,336\,054.294 \text{ m}.$$

The difference in ellipsoidal height is therefore

$$\Delta h = h_{GRS80} - h_{TP} = -0.706 \text{ m}.$$

Final Result:

Using the GRS80 ellipsoid instead of the Topex/Poseidon ellipsoid reduces the computed ellipsoidal height of the satellite by approximately 0.706 m.

MATLAB Code for Task 3.1. Q.No:2

```
% Task 3.1.2
% Height difference using Topex/Poseidon and GRS80 ellipsoids

clc; clear;

% -------------------------------------------------
% Given Cartesian coordinates of the satellite
% -------------------------------------------------
x = 4831342.4634;   % m
y = 2833965.0779;   % m
z = 5289590.6351;   % m

p = sqrt(x^2 + y^2);

% -------------------------------------------------
% Ellipsoid parameters
% -------------------------------------------------
% Topex/Poseidon
a_TP = 6378136.3;
f_TP = 1/298.257;

% GRS80
a_GRS = 6378137.0;
f_GRS = 1/298.257222101;

% -------------------------------------------------
% Run iterations for both ellipsoids
% -------------------------------------------------
fprintf('\n=========== TOPEX / POSEIDON ELLIPSOID ===========\n');
[h_TP, phi_TP] = iterate_height(a_TP, f_TP, x, y, z);

fprintf('\n================== GRS80 ELLIPSOID =================\n');
[h_GRS, phi_GRS] = iterate_height(a_GRS, f_GRS, x, y, z);

% -------------------------------------------------
% Final comparison
% -------------------------------------------------
fprintf('\n=============== FINAL COMPARISON ==================\n');
fprintf('Final height (T/P)    = %.3f m\n', h_TP);
fprintf('Final height (GRS80)  = %.3f m\n', h_GRS);
fprintf('Height difference    = %.3f m\n', h_GRS - h_TP);

% -------------------------------------------------
% Function performing iterations and printing results
% -------------------------------------------------
function [h_final, phi_final] = iterate_height(a, f, x, y, z)

    e2 = 2*f - f^2;
    p  = sqrt(x^2 + y^2);

    % Initial latitude
```

```
    phi = atan( z / (p*(1 - e2)) );

    fprintf('Iter | Latitude (deg) | Height (m)\n');
    fprintf('----------------------------------\n');

    for k = 1:5
        N = a / sqrt(1 - e2*sin(phi)^2);
        h = p/cos(phi) - N;
        phi = atan( z / ( p*(1 - (e2*N)/(N + h)) ) );

        fprintf('%4d | %14.9f | %10.3f\n', k, rad2deg(phi), h);
    end

    h_final  = h;
    phi_final = phi;
end
```

1.3 Task 3.1 Q.No:3

Geocentric Latitude is given by;

$$\psi = tan^{-}1(\frac{Z}{p})$$

$$\psi = tan^{-}1(\frac{Z}{\sqrt{x^2+y^2}})$$

$$\psi = 43.3611867^{\circ}$$

From Task 3.1. Q.No.1., we have

Geodetic Latitude

$$\varphi = 43.520000025^{\circ}$$

Now, **the Difference in Latitude is given by:**

$$Diff = \varphi - \psi = 0.158813325^{\circ}$$

Similarly, **Error Calculation in Meters**

$$\Delta S \approx Radius\_of\_Earth * \frac{Difference\_in\_Latitude * \pi}{180}$$

$$\Delta S \approx 6378137 * \frac{0.158813325 * \pi}{180}$$

$$\boxed{\Delta S \approx 17679.02m}$$

MATLAB Code for Task 3.1 Q.No:3

```
% Question 3: Geocentric vs Geodetic Latitude
% MATLAB verification code

clc; clear;

% Given Cartesian coordinates
x = 4831342.4634;   % m
y = 2833965.0779;   % m
z = 5289590.6351;   % m

% Geocentric latitude
p = sqrt(x^2 + y^2);
psi = atan(z / p);           % radians
psi_deg = rad2deg(psi);      % degrees

% Geodetic latitude (from Task 1.1)
phi_deg = 43.520000025;      % degrees
phi = deg2rad(phi_deg);      % radians

% Difference in latitude
diff_deg = phi_deg - psi_deg;
diff_rad = deg2rad(diff_deg);

% Error in meters
R = 6378137;                 % Earth radius [m]
```

```
delta_S = R * diff_rad;

% Display results
fprintf('Geocentric latitude = %.9f deg\n', psi_deg);
fprintf('Geodetic latitude   = %.9f deg\n', phi_deg);
fprintf('Difference = %.9f deg\n', diff_deg);
fprintf(' Error   = %.2f m\n', delta_S);
```

1.4 Task 3.1 Q.No:4

Altimeter footprint or Sub-satellite Point is the point in the reference ellipsoid directly below the satellite. It is located on the ellipsoid. It is also known as the shadow of the satellite on the ellipsoid. The height at this point is zero.
Solution:
Given, Satellite Coordinates:

$$\begin{bmatrix} x_{sat} \\ y_{sat} \\ z_{sat} \end{bmatrix} = \begin{bmatrix} 4\,831\,342.4634 \\ 2\,833\,965.0779 \\ 5\,289\,590.6351 \end{bmatrix} \text{ m}$$

Then, the coordinates of the Altimeter Footprint;

$$\begin{aligned} \text{Latitude:} \quad & \varphi_{foot} = \varphi_{sat} = 43.520000025°(Given\_in\_Question) \\ \text{Longitude:} \quad & \lambda_{foot} = \lambda_{sat} = 30.395000005°(Given\_in\_Question) \\ \text{Height:} \quad & h_{foot} = 0 \text{ m} \end{aligned}$$

Then,
Conversion of Latitude and Longitude into Cartesian Coordinates;

$$N_f = \frac{a}{\sqrt{1 - e^2 \sin^2 \varphi_f}} = 6388283.676m$$

$$\begin{aligned} X_f &= N_f \cos \varphi_f \cos \lambda_f = 3995680.082m \\ Y_f &= N_f \cos \varphi_f \sin \lambda_f = 2343782.893m \\ Z_f &= N_f(1 - e^2) \sin \varphi_f = 4369572.824m \end{aligned}$$

Similarly, as given; The ellipsoidal longitude, latitude, and height of a nearby tide gauge station w.r.t. to the Topex/Poseidon ellipsoid

$$\begin{aligned} \lambda_{tg}^{T/P} &= 30.329\,000\,100° \\ \varphi_{tg}^{T/P} &= 43.592\,000\,088° \\ h_{tg}^{T/P} &= 30.888 \text{ m} \end{aligned}$$

Now, Conversion of tide gauge coordinates into Cartesian Coordinates;

$$N_{tg} = \frac{a}{\sqrt{1 - e^2 \sin^2 \varphi_{tg}}} = 6388310.598m \tag{1}$$

$$\begin{aligned} X_{tg} &= (N_{tg} + h_{tg}) \cos \varphi_{tg} \cos \lambda_{tg} = 3993638.834m \\ Y_{tg} &= (N_{tg} + h_{tg}) \cos \varphi_{tg} \sin \lambda_{tg} = 2336406.516m \\ Z_{tg} &= (N_{tg}(1 - e^2) + h_{tg}) \sin \varphi_{tg} = 4375391.343m \end{aligned}$$

Hence, the difference between the coordinates is:

$$\begin{aligned} \Delta X &= 2041.247 \text{ m} \\ \Delta Y &= 7376.377 \text{ m} \\ \Delta Z &= -5818.519 \text{ m} \end{aligned}$$

The Euclidean distance between altimeter footprint and tide gauge is given by:

$$d = \sqrt{(X_f - X_{tg})^2 + (Y_f - Y_{tg})^2 + (Z_f - Z_{tg})^2}$$

$$\boxed{d = 9614.917m = 9.615km}$$

MATLAB Code: Task 3.1. Q.No:4

```
% ---------------------------------------------
% Distance between altimeter footprint and tide gauge
% ---------------------------------------------

clc; clear;

fprintf('--- Ellipsoid parameters (Topex/Poseidon) ---\n');

% Ellipsoid parameters
a  = 6378136.3;                % semi-major axis [m]
f  = 1/298.257000;
e2 = 2*f - f^2;

fprintf('a  = %.4f m\n', a);
fprintf('f  = %.12f\n', f);
fprintf('e^2 = %.12f\n\n', e2);

% ---------------------------------------------
% Satellite geodetic coordinates (from Q1)
% ---------------------------------------------
phi_s = deg2rad(43.520000025);  % latitude [rad]
lam_s = deg2rad(30.395000005);  % longitude [rad]

fprintf('--- Satellite geodetic coordinates ---\n');
fprintf('Latitude   = %.9f deg\n', rad2deg(phi_s));
fprintf('Longitude  = %.9f deg\n\n', rad2deg(lam_s));

% ---------------------------------------------
% Altimeter footprint (h = 0)
% ---------------------------------------------
Nf = a / sqrt(1 - e2*sin(phi_s)^2);

Xf = Nf * cos(phi_s) * cos(lam_s);
Yf = Nf * cos(phi_s) * sin(lam_s);
Zf = Nf * (1 - e2) * sin(phi_s);

fprintf('--- Altimeter footprint (ECEF) ---\n');
fprintf('Prime vertical radius N_f = %.3f m\n', Nf);
fprintf('X_f = %.3f m\n', Xf);
fprintf('Y_f = %.3f m\n', Yf);
fprintf('Z_f = %.3f m\n\n', Zf);

% ---------------------------------------------
% Tide gauge coordinates
% ---------------------------------------------
```

```
phi_tg = deg2rad(43.592000088);
lam_tg = deg2rad(30.329000100);
h_tg   = 30.888;

Nt = a / sqrt(1 - e2*sin(phi_tg)^2);

Xt = (Nt + h_tg) * cos(phi_tg) * cos(lam_tg);
Yt = (Nt + h_tg) * cos(phi_tg) * sin(lam_tg);
Zt = (Nt*(1 - e2) + h_tg) * sin(phi_tg);

fprintf('--- Tide gauge coordinates (ECEF) ---\n');
fprintf('Latitude   = %.9f deg\n', rad2deg(phi_tg));
fprintf('Longitude  = %.9f deg\n', rad2deg(lam_tg));
fprintf('Height h_tg    = %.3f m\n', h_tg);
fprintf('Prime vertical radius N_tg = %.3f m\n', Nt);
fprintf('X_tg = %.3f m\n', Xt);
fprintf('Y_tg = %.3f m\n', Yt);
fprintf('Z_tg = %.3f m\n\n', Zt);

% ---------------------------------------------
% Euclidean distance
% ---------------------------------------------
d = sqrt((Xf - Xt)^2 + (Yf - Yt)^2 + (Zf - Zt)^2);

fprintf('--- Distance computation ---\n');
fprintf('Difference in X  %.3f m\n', Xf - Xt);
fprintf('Difference in Y = %.3f m\n', Yf - Yt);
fprintf('Difference in Z = %.3f m\n', Zf - Zt);
fprintf('Distance footprint{tide gauge = %.3f m\n', d);
```

2 Task 3.2.

Task 3.2 Q.No: 1 – Satellite footprint coordinates and ground track

Given Inclination $i = 66.036006500°$, RAAN $\Omega = 335.188990200°$, argument of perigee $\omega = 289.450123600°$. Circular orbit with period $T = 110.0000\,\text{min}$ and sampling $\Delta t = 1\,\text{s}$.
We use the TOPEX/Poseidon ellipsoid with semi-major axis $a = 6378136.3\,\text{m}$ and flattening $f = 1/298.257$, therefore $b = a(1-f)$.
Earth constants:

$$\mu = 3.986004418 \times 10^{14}\,\text{m}^3\text{s}^{-2}, \qquad \omega_E = 7.292115 \times 10^{-5}\,\text{rad s}^{-1}.$$

(1) Footprint coordinates for one revolution

Step 1: Converting period to seconds and compute mean motion

$$T = 110\,\text{min} \times 60 = 6600\,\text{s}.$$

For a circular orbit, mean motion

$$n = \frac{2\pi}{T} = \frac{2\pi}{6600} = 9.519977738 \times 10^{-4}\,\text{rad s}^{-1}.$$

Step 2: Orbital radius from Kepler's third law For a circular orbit, $r = a_{\text{orb}}$ and

$$a_{\text{orb}} = \left(\mu\left(\frac{T}{2\pi}\right)^2\right)^{1/3}.$$

Numerically:

$$\frac{T}{2\pi} = \frac{6600}{2\pi} = 1050.422\ \text{s},$$

$$\mu\left(\frac{T}{2\pi}\right)^2 = (3.986004418 \times 10^{14})(1050.422)^2 \approx 4.399 \times 10^{20},$$

$$a_{\text{orb}} \approx (4.399 \times 10^{20})^{1/3} = 7.604814705 \times 10^6\,\text{m}.$$

Hence $r = 7.604814705 \times 10^6\,\text{m}$.

Step 3: True anomaly at each measurement epoch Let the first measurement be at time $t_0 = t$ and assuming $\nu(t_0) = 0$. For a circular orbit:

$$\nu(t_k) = n\,(t_k - t_0) = n\,k\Delta t, \qquad k = 0, 1, \ldots, N-1$$

with $N = T/\Delta t = 6600$ samples. Example:

$$\nu(t_1) = n(1) = 9.51998 \times 10^{-4}\,\text{rad} = 0.05456°.$$

Step 4: Position in the orbital plane

$$\mathbf{r}_{PQW}(t_k) = \begin{bmatrix} r\cos\nu(t_k) \\ r\sin\nu(t_k) \\ 0 \end{bmatrix}.$$

Step 5: Rotate into the inertial frame (ECI) Using the standard rotation sequence:

$$\mathbf{r}_{ECI} = \mathbf{R}_3(\Omega)\,\mathbf{R}_1(i)\,\mathbf{R}_3(\omega)\,\mathbf{r}_{PQW}.$$

where

$$\mathbf{R}_1(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}, \quad \mathbf{R}_3(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Step 6: Converting ECI to Earth-fixed (ECEF) Neglecting polar motion and precession-nutation, the Earth-fixed rotation is

$$\theta(t_k) = \omega_E(t_k - t_0) = \omega_E k\Delta t, \quad \mathbf{r}_{ECEF}(t_k) = \mathbf{R}_3(-\theta(t_k))\,\mathbf{r}_{ECI}(t_k).$$

Example Earth rotation in 1 s:

$$\theta(1) = \omega_E(1) = 7.292115 \times 10^{-5}\,\text{rad} = 0.004178^\circ.$$

Step 7: Nadir footprint (intersection with the ellipsoid) Let $\mathbf{u} = \mathbf{r}_{ECEF}/\|\mathbf{r}_{ECEF}\|$ be the direction. The footprint point is $\mathbf{p} = s\,\mathbf{r}_{ECEF}$ where s is chosen so that $\mathbf{p}$ lies on the ellipsoid:

$$\frac{p_x^2 + p_y^2}{a^2} + \frac{p_z^2}{b^2} = 1.$$

With $\mathbf{p} = s\mathbf{r}_{ECEF}$:

$$s = \left(\frac{x^2 + y^2}{a^2} + \frac{z^2}{b^2}\right)^{-1/2}, \quad \mathbf{p} = s\begin{bmatrix} x \\ y \\ z \end{bmatrix}.$$

Step 8: Converting ECEF footprint to geodetic latitude/longitude Longitude:

$$\lambda = \operatorname{atan2}(p_y, p_x).$$

Latitude ϕ is geodetic and is obtained iteratively, using $e^2 = f(2-f)$ and $N(\phi) = \frac{a}{\sqrt{1-e^2\sin^2\phi}}$. A robust iteration is:

$$\phi^{(0)} = \operatorname{atan2}(p_z, \sqrt{p_x^2 + p_y^2}(1 - e^2)),$$

then repeat

$$N^{(j)} = \frac{a}{\sqrt{1 - e^2\sin^2\phi^{(j)}}}, \quad h^{(j)} = \frac{\sqrt{p_x^2 + p_y^2}}{\cos\phi^{(j)}} - N^{(j)},$$

$$\phi^{(j+1)} = \operatorname{atan2}\left(p_z, \sqrt{p_x^2 + p_y^2}\left(1 - \frac{e^2 N^{(j)}}{N^{(j)} + h^{(j)}}\right)\right).$$

Finally $\phi = \phi^{(j)}$ when converged.

Reported range for one revolution (1 Hz sampling) Using $N = 6600$ samples:

$$\boxed{\phi \in [-66.1785^\circ,\ +66.1785^\circ], \qquad \lambda \in [-179.9866^\circ,\ +179.9308^\circ]}$$

NOTE: MATLAB Code of this question is written below together with Q.No: 2

Task 3.2.Q.No: 2 Groundtrack plot

Coastline maps were added to give a geographic reference for the satellite ground track. The computed latitude and longitude points were plotted on a global map, and coastline data were overlaid so that continents and oceans are visible.In MATLAB, coastline data are provided by mapping tools. Including coastlines makes the ground track plot clearer, more interpretable, and scientifically meaningful.

Plotting φ_k vs λ_k globally, overlaying coastlines, and labeling axes.

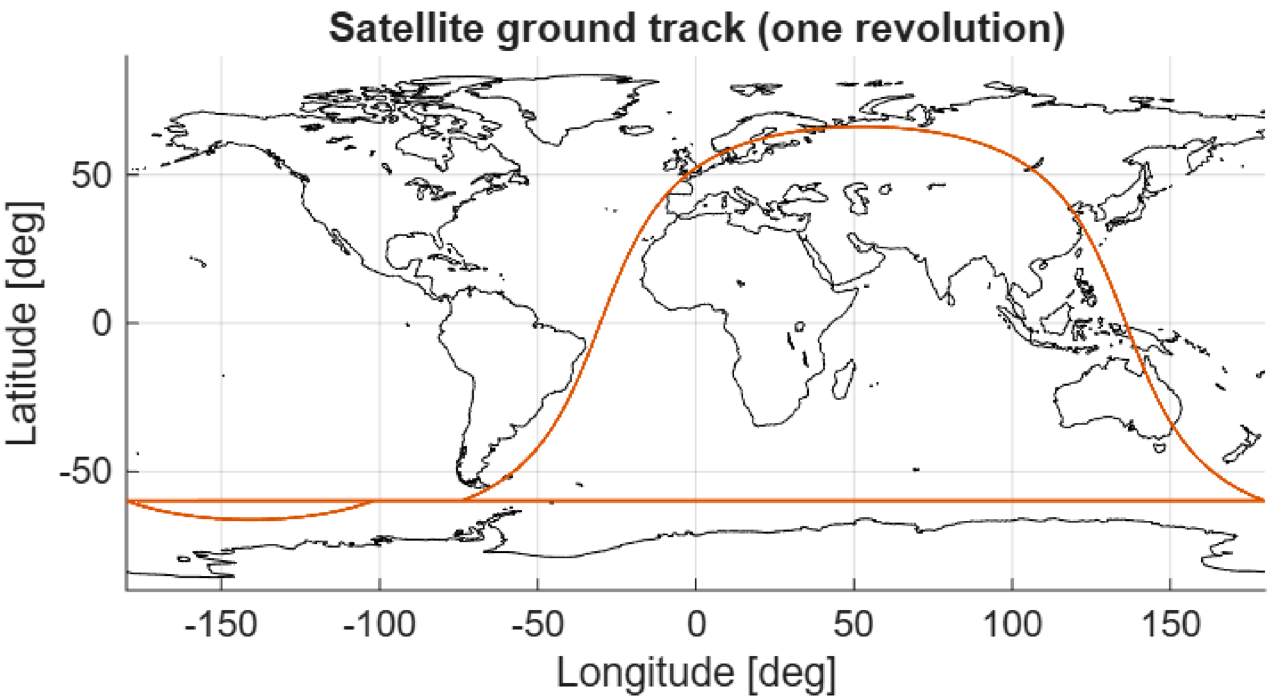

Figure 1: Satellite ground track for one revolution ($T = 110$ min, $\Delta t = 1$ s) with coastlines.

MATLAB Code (For both Task 3.2. Q.No 1 and 2)

```
%% Task 3.2 -- Satellite footprint coordinates + ground track map
clear; clc;

%% Given orbital elements (deg)
i_deg     = 66.036006500;
Omega_deg= 335.188990200;     % RAAN
omega_deg= 289.450123600;     % argument of perigee

T_min = 110.0;
dt    = 1.0;

%% Constants
mu        = 3.986004418e14;      % [m^3/s^2]
omegaE    = 7.2921150e-5;        % [rad/s]

% TOPEX/Poseidon ellipsoid
a = 6378136.3;                   % [m]
f = 1/298.257;
b = a*(1-f);
e2 = f*(2-f);
```

```
%% Step 1: Period [s], mean motion n
T = T_min*60;                   % 6600 s
n = 2*pi/T;

%% Step 2: Orbital radius from Kepler (circular)
r = (mu*(T/(2*pi))^2)^(1/3);

%% Time grid for one revolution
N = round(T/dt);
t = (0:N-1)'*dt;

%% Step 3: True anomaly
nu = n*t;

%% Rotation matrices
R1 = @(x)[1 0 0; 0 cos(x) -sin(x); 0 sin(x) cos(x)];
R3 = @(x)[cos(x) -sin(x) 0; sin(x) cos(x) 0; 0 0 1];

i     = deg2rad(i_deg);
Omega = deg2rad(Omega_deg);
omega = deg2rad(omega_deg);

Q = R3(Omega)*R1(i)*R3(omega);

%% Position in PQW then ECI
rPQW = [r*cos(nu)'; r*sin(nu)'; zeros(1,N)];
rECI = Q*rPQW;

%% ECI -> ECEF
theta = omegaE*t';
rECEF = zeros(3,N);
for k=1:N
    rECEF(:,k) = R3(-theta(k))*rECI(:,k);
end

%% Footprint intersection with ellipsoid
x = rECEF(1,:); y = rECEF(2,:); z = rECEF(3,:);
den = (x.^2 + y.^2)/a^2 + (z.^2)/b^2;
s   = 1./sqrt(den);

xf = s.*x; yf = s.*y; zf = s.*z;

%% Convert ECEF to geodetic lat/lon
lon = atan2(yf, xf);
p   = sqrt(xf.^2 + yf.^2);
lat = atan2(zf, p*(1-e2));

for it=1:10
    sinlat = sin(lat);
    Nphi = a ./ sqrt(1 - e2*sinlat.^2);
    h = p./cos(lat) - Nphi;
```

```
    lat = atan2(zf, p .* (1 - (e2.*Nphi)./(Nphi + h)));
end

lat_deg = rad2deg(lat);
lon_deg = mod(rad2deg(lon) + 180, 360) - 180;

%% Plot ground track
figure; hold on; grid on;
load coastlines
plot(coastlon, coastlat, 'k');
plot(lon_deg, lat_deg, 'b');
xlabel('Longitude [deg]');
ylabel('Latitude [deg]');
title('Satellite ground track (one revolution)');
```