ECE 6390: Homework 2

Hyeonjae Park
Georgia Institute of Technology

September 7, 2025

1 Problem Setup.

From the previous homework, for continuous time,

$$\ddot{r} = r\,\dot{\theta}^2 - \frac{\mu}{r^2}, \qquad \ddot{\theta} = -\frac{2\,\dot{r}\,\dot{\theta}}{r}.$$

Here, μ is the standard gravitational parameter, defined as the product of the universal gravitational constant (G) and the mass of the primary body (M_p), such that $\mu = GM_p$.

$$\frac{\Delta r_n}{\Delta t} = \dot{r}_n, \qquad \frac{\Delta\theta_n}{\Delta t} = \dot{\theta}_n, \qquad r_{\mathrm{mid},n} = r_n + \tfrac{1}{2}\,\Delta r_n.$$

Then the step $n \to n+1$ is

$$\begin{aligned}
r_{n+1} &= r_n + \Delta r_n,\\
\theta_{n+1} &= \theta_n + \Delta\theta_n,\\
\Delta r_{n+1} &= \Delta r_n + \Big(r_{\mathrm{mid},n}\,\Delta\theta_n^2 - \frac{\mu}{r_n^2}\,\Delta t^2\Big),\\
\Delta\theta_{n+1} &= \Delta\theta_n - \frac{2\,\Delta r_n\,\Delta\theta_n}{r_{\mathrm{mid},n}}.
\end{aligned}$$

Solar Pressure Model

Assumptions. (1) Effective reflectivity $\alpha_r = 0.5$. (2) Solar pressure acts in the equatorial plane. (3) Sun direction rotates uniformly: $\alpha(t) = \omega_{\mathrm{yr}}t + \phi_0$, with $\omega_{\mathrm{yr}} = \dfrac{2\pi}{365.25636 \times 86400}$.

Area density and magnitude.

$$\rho_A = \frac{m_s}{A_s}, \qquad F_{\mathrm{solar}} = 9.08\ \mu\mathrm{N/m^2} \times \alpha_r A_s,$$

$$a_{\mathrm{solar}} = \frac{F_{\mathrm{solar}}}{m_s} = \frac{9.08 \times 10^{-6}\,\alpha_r}{\rho_A}\ [\mathrm{m/s^2}] = \frac{9.08 \times 10^{-9}\,\alpha_r}{\rho_A}\ [\mathrm{km/s^2}].$$

Components in polar (r, θ). Let $\psi(t) = \alpha(t) - \theta(t)$. Then radial component and tangential component is,

$$a_r = a_{\mathrm{solar}}\cos\psi, \qquad a_\theta = a_{\mathrm{solar}}\sin\psi.$$

Modified update equations. The solar pressure adds acceleration components a_r and a_θ/r to the continuous-time equations,

$$\ddot{r} = r\,\dot{\theta}^2 - \frac{\mu}{r^2} + a_r, \qquad \ddot{\theta} = -\frac{2\,\dot{r}\,\dot{\theta}}{r} + \frac{a_\theta}{r}.$$

The corresponding discrete update for the step $n \to n{+}1$ becomes

$$\Delta r_{n+1} = \Delta r_n + \Big(r_{\mathrm{mid},n}\,\Delta\theta_n^2 - \frac{\mu}{r_n^2}\,\Delta t^2 + a_{\mathrm{solar}}\cos\psi_n\,\Delta t^2\Big),$$

$$\Delta\theta_{n+1} = \Delta\theta_n - \frac{2\,\Delta r_n\,\Delta\theta_n}{r_{\mathrm{mid},n}} + \frac{a_{\mathrm{solar}}\sin\psi_n}{\mathbf{r_{mid,n}}}\,\Delta t^2,$$

where $t_n = n\Delta t,\ \alpha_n = \phi_0 + \omega_{\mathrm{yr}} t_n,\ \psi_n = \alpha_n - \theta_n$.

2 Results

2.1 Minimum area density vs. target drift

Table 1 compares the minimum area density ρ_A required to produce 1°, 5°, and 15° of azimuth drift over one sidereal year for two time steps. The 1° and 5° cases agree within about 1% and 4%, respectively, while the 15° case shows strong step–size sensitivity. We therefore report the $dt = 0.5\,\mathrm{s}$ values as our final results (without extrapolation). The signed percentage indicates $(\rho_A^{1\mathrm{s}} - \rho_A^{0.5\mathrm{s}})/\rho_A^{0.5\mathrm{s}} \times 100\%$.

Target drift [deg]	$\boldsymbol{\rho_A}$ **(dt = 0.5 s) [kg/m^2]**	$\boldsymbol{\rho_A}$ **(dt = 1 s) [kg/m^2]**	**Change vs. 0.5 s [%]**
1	14.544	14.681	+0.94
5	2.784	2.675	−3.92
15	0.792	0.231	−70.83

Table 1: Minimum area density ρ_A versus time step. We use the $dt = 0.5\,\mathrm{s}$ column as the reported values. Negative percentages indicate that the $dt = 1\,\mathrm{s}$ run produced a smaller ρ_A than the $dt = 0.5\,\mathrm{s}$ run.

2.2 Minimizing 1-year azimuth deviation at $\rho_A = 0.5$

Method. With $\rho_A = 0.5$ kg/m^2 fixed, we chose a small tangential impulse Δv_t to drive the 1-year mean azimuth drift (slope) to ≈ 0 deg/day using a 4-term regression fit $[1,\ t\ (\mathrm{day}),\ \sin(2\pi t/1\mathrm{y}),\ \cos(2\pi t/1\mathrm{y})]$. Keeping Δv_t fixed, we then selected the initial longitude offset $\Delta\lambda_0$ to minimize the annual (1/year) harmonic amplitude. Dynamics were integrated in planar ECI using RK4 with a rotating solar-pressure acceleration $a_{\mathrm{solar}} = 9.08 \times 10^{-9}\alpha_r/\rho_A$ km/s^2. Look angles were computed from local ENU(East North Up) via $\mathrm{Az} = \mathrm{atan2}(E, N)$ and $\mathrm{El} = \mathrm{atan2}(U, \sqrt{E^2 + N^2})$.

Δv_t [m/s]	$\Delta\lambda_0$ [deg]	Slope [deg/day]	Annual amp [deg]	Max |Az−lin| [deg]	RMS [deg]
+0.010	−0.691	**0.001350**	**0.030**	**9.156**	**4.579**

Table 2: Summary of the minimized-azimuth design at $\rho_A = 0.5$ kg/m^2. The slope corresponds to $\approx 0.49°$ per year (viable). The annual component is strongly suppressed. The remaining ± few-degree swing is dominated by the daily geosynchronous east–west oscillation.

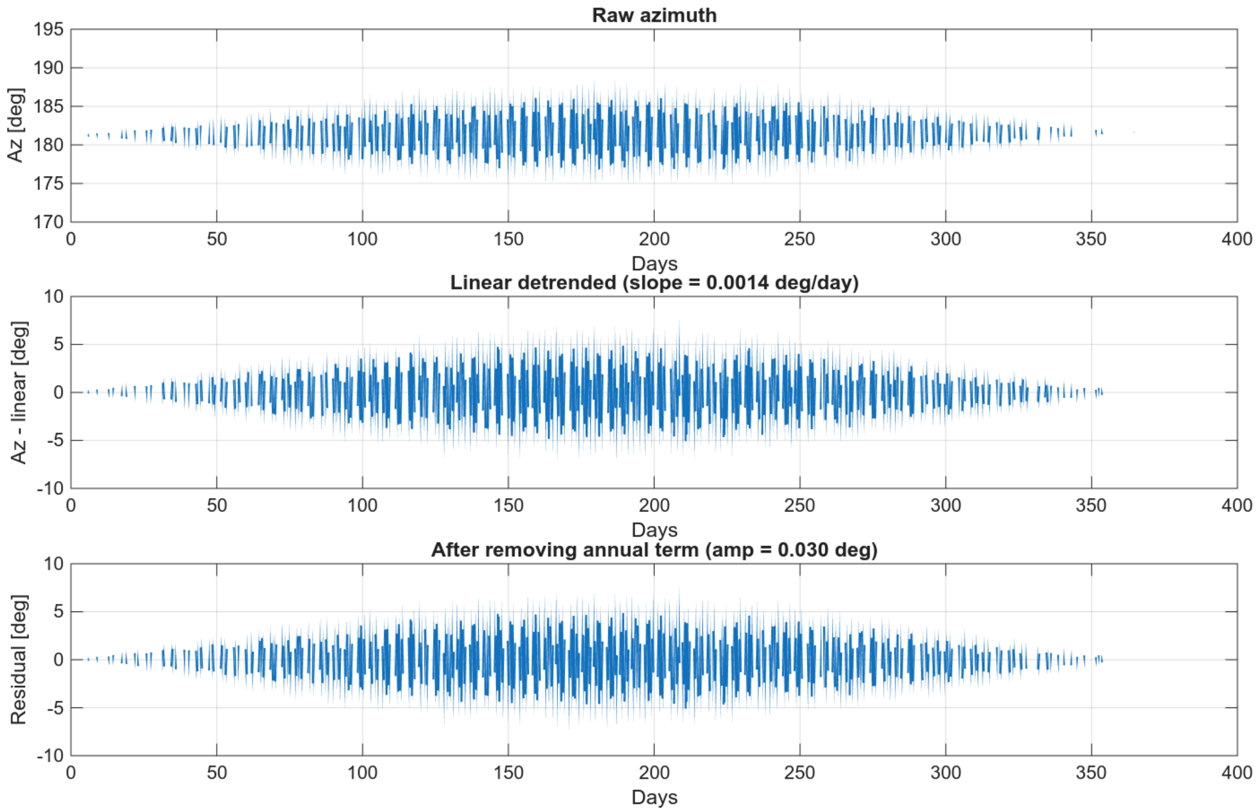

Figure 1: One-year azimuth as seen from Atlanta for the minimized-deviation design ($\rho_A = 0.5$ kg/m^2). Top: raw azimuth. Middle: azimuth after removing the linear trend (slope = 0.00135 deg/day). Bottom: residual after additionally removing the annual component (amplitude = 0.030°).

Takeaway. With $\Delta v_t = +0.010$ m/s and $\Delta\lambda_0 = -0.691°$, the 1-year mean drift is $< 1°$ and the annual wobble is $\approx 0.03°$. The residual variation is primarily the daily geosynchronous oscillation.

3 Code.

Minimum area density

```
function minimum_area_density()
% Finds the minimum area density rho_A that yields a target azimuth drift
% (1  , 5  , 15  ) over one sidereal year as seen from Atlanta (Van Leer).

%% Constants
mu = 398600.4418; %   =GM [km^3/s^2]
r_earth = 6378.137; % [km]
w_earth = 2*pi/86164.0905; % Earth sidereal rotation [rad/s]
sidereal_year = 365.25636*86400; % Sidereal year [s]
w_yr = 2*pi/sidereal_year; % Sun-direction rotation [rad/s]
alpha_r = 0.5; % Effective reflectivity (Given in problem)

% Station (Van Leer, Atlanta) latitute, longitude [rad]
station_lat = deg2rad(33.7758);
station_lon = deg2rad(-84.39738);

% GEO initial conditions (circular, equatorial)
r0 = (mu / w_earth^2)^(1/3); % [km]
vtheta0 = r0 * w_earth;
theta0 = 0.0;
```

```matlab
lon0 = station_lon;
vr0 = 0.0;

% Integrator
% Test for 1, 0.5
dt = 0.5; % [s]

%% Solve for rho_A at three drift targets
targets = [1, 5, 15]; % degrees
rho_out = zeros(size(targets));
for k = 1:numel(targets)
    rho_out(k) = rho_for_target_deg(targets(k));
end

fprintf('\nMinimum area density over 1 sidereal year\n');
for k = 1:numel(targets)
    fprintf('Target drift %2d   : rho_A = %.3f kg/m^2\n', targets(k), rho_out(k));
end

%%
    function rho = rho_for_target_deg(target_deg)
        % Find rho_A such that | Az |      target_deg after one sidereal year.
        % Drift decreases as rho_A increases.

        % Initial bracket [rho_lo, rho_hi] where drift_lo > target > drift_hi

        rho_lo = 0.5;
        rho_hi = 200; % broad bracket [kg/m^2]

        drift_lo = drift_for_rho(rho_lo);
        drift_hi = drift_for_rho(rho_hi);

        it = 0;
        while ~(drift_lo>target_deg && drift_hi<target_deg) && it < 50
            if drift_lo <= target_deg, rho_lo = rho_lo/2; drift_lo = drift_for_rho(rho_lo); end
            if drift_hi >= target_deg, rho_hi = rho_hi*2; drift_hi = drift_for_rho(rho_hi); end
            it = it + 1;
        end
        % Bisection
        for i=1:50
            mid = 0.5*(rho_lo+rho_hi);
            d = drift_for_rho(mid);
            if d > target_deg
                rho_lo = mid;
            else
                rho_hi = mid;
            end
        end
        rho = 0.5*(rho_lo+rho_hi);
    end

    function drift_deg = drift_for_rho(rho_A)
        [r_hist,theta_hist,tt] = propagate(r0,theta0,vr0,vtheta0, ...
            mu,alpha_r,rho_A,w_yr,dt,sidereal_year);
        lsh = wrapToPi( (theta_hist - w_earth*tt) + lon0 ); % ECI->ECEF subpoint longitude
        azimuth_0 = az_from_station(station_lat,station_lon,0,lsh(1),r_earth);
        azimuth_f = az_from_station(station_lat,station_lon,0,lsh(end),r_earth);
        drift_deg = abs(rad2deg(angdiff(azimuth_0,azimuth_f)));
    end
end

%% Propagator with solar-pressure
function [r_hist,theta_hist,t_hist] = propagate(r0,theta0,vr0,vtheta0,...
    mu,alpha_r,rho_A,w_yr,dt,simTime)

n = ceil(simTime/dt);
r_hist = zeros(n+1,1);
t_hist = (0:n)'*dt;
```

```
r_hist(1) = r0;
theta_hist = zeros(n+1,1);
theta_hist(1) = theta0;

% Delta variables store    over one step
delta_r = vr0*dt;
delta_theta = (vtheta0/r0)*dt;

% Solar acceleration magnitude [km/s^2]
a_solar = (9.08e-6*alpha_r/rho_A)/1000;  % m/s^2 to km/s^2

r = r0; theta = theta0;
for k = 1:n
    t = (k-1)*dt;

    % Position update
    r_next = r + delta_r;
    theta_next = theta + delta_theta;

    % Mid-step radius
    r_mid = r + 0.5*delta_r;
    theta_mid = theta + 0.5*delta_theta;

    % Sun direction and components
    alpha = w_yr*(t + 0.5*dt); % Sun angle (equatorial)
    psi = alpha - theta_mid; % Relative angle
    a_r = a_solar*cos(psi);
    a_theta = a_solar*sin(psi);

    % Modified updates
    dr_next  = delta_r + ( r_mid*(delta_theta^2) - (mu/(r_mid^2))*dt^2 + a_r*dt^2 );
    dth_next = delta_theta - (2*delta_r*delta_theta)/r_mid + (a_theta/r_mid)*dt^2;

    % Roll
    r = r_next;
    theta = theta_next;
    delta_r = dr_next;
    delta_theta = dth_next;

    r_hist(k+1) = r;
    theta_hist(k+1) = theta;
end
end

%% Azimuth from station (station_lat, station_lon) to sub-satellite point (target_lat, target_lon)
function azimuth = az_from_station(station_lat, station_lon, target_lat, target_lon, earth_radius)

    % Spherical law of cosines for central angle between station and target
    cos_gamma = sin(target_lat)*sin(station_lat) + ...
                cos(target_lat)*cos(station_lat)*cos(target_lon - station_lon);
    central_angle = acos( min(1, max(-1, cos_gamma)) );

    % Magnitude of the azimuth offset (an acute angle in [0,   /2])
    sin_azimuth_abs = abs( sin(abs(station_lon - target_lon)) * cos(target_lat) / ...
                           max(1e-12, sin(central_angle)) );
    azimuth_offset = asin( min(1, max(0, sin_azimuth_abs)) ); % in [0,   /2]

    % Quadrant resolution (for GEO with target_lat = 0 and station_lat > 0, target is to the south)
    if target_lon > station_lon % South & East
        azimuth = pi - azimuth_offset;
    else % South & West
        azimuth = pi + azimuth_offset;
    end

    azimuth = mod(azimuth, 2*pi);
end

%%
```

```
function x = wrapToPi(a)
    x = mod(a+pi, 2*pi) - pi;
end

function d = angdiff(a,b)
% minimal signed difference b-a in [-pi,pi]
    d = atan2(sin(b-a), cos(b-a));
end
```

Minimum azimuth look angles

```
function result = minimum_azimuth_lookangles()
% Choose  v_t  so that the 1-year mean azimuth drift (slope)     0.
% Choose   0   to minimize the 1-year (solar-pressure) harmonic amplitude.
% Run a high-fidelity simulation and plot 1-year look angles/metrics.
%
% Output:
% Console:  v_t ,   0  , slope, annual amplitude, detrended/annual-removed Max/RMS
% Figure: (1) Raw azimuth, (2) Linear detrended, (3) After removing annual term
% result struct: dv_t_mps, dlambda0_deg, metrics

%% Constants
mu = 398600.4418; %   =GM [km^3/s^2]
r_earth = 6378.137; % Earth radius [km]
w_earth = 2*pi/86164.0905; % Earth sidereal rotation [rad/s]
sidereal_year = 365.25636*86400; % Sidereal year [s]
sidereal_days = 365.25636; % Sidereal year [day]
w_yr = 2*pi/sidereal_year; % Sun-direction rotation [rad/s]
alpha_r = 0.5; % Effective reflectivity (given)
rho_A = 0.5; % Area density [kg/m^2] (fixed)
r0 = 42164.17; % GEO radius [km]

% Station (Van Leer, Atlanta)     latitude, longitude [rad]
station_lat = deg2rad(33.7758);
station_lon = deg2rad(-84.39738);

%% Find  v_t  such that mean drift slope     0
dt_A = 120; % [s] 2 min (tighter slope estimation)

slope_fn = @(dv_t) slope_deg_per_day( ...
    dv_t, 0.0, dt_A, mu, r_earth, w_earth, sidereal_year, w_yr, ...
    alpha_r, rho_A, r0, station_lat, station_lon, sidereal_days);

% Grid to bracket a sign change for fzero
cand = linspace(-1, 1, 81); %  v_t  in [-1, +1] m/s
svals = arrayfun(slope_fn, cand);
ix = find(sign(svals(1:end-1)).*sign(svals(2:end)) <= 0, 1, 'first');

if ~isempty(ix)
    bracket = [cand(ix), cand(ix+1)];
else
    cand2 = linspace(-5, 5, 81);
    svals2 = arrayfun(slope_fn, cand2);
    ix2 = find(sign(svals2(1:end-1)).*sign(svals2(2:end)) <= 0, 1, 'first');
    if ~isempty(ix2)
        bracket = [cand2(ix2), cand2(ix2+1)];
    else
        bracket = [-5, 5];
    end
end

dv_t_star = fzero(slope_fn, bracket); % [m/s]

%% Minimize 1-year amplitude by setting   0
dt_B = dt_A;
amp_fn = @(d_lambda0) annual_amp_deg( ...
```

```
    dv_t_star, d_lambda0, dt_B, mu, r_earth, w_earth, sidereal_year, w_yr, ...
    alpha_r, rho_A, r0, station_lat, station_lon, sidereal_days);

% Search range  15  deg
d_lambda0_star = fminbnd(amp_fn, deg2rad(-15), deg2rad(15));

%% Simulation & plots
dt_F = 60; % [s]

[t_hist, az_deg_raw, az_deg_detr_lin, resid_noannual_deg, metrics] = ...
    sim_and_metrics( ...
        dv_t_star, d_lambda0_star, dt_F, mu, r_earth, w_earth, ...
        sidereal_year, w_yr, alpha_r, rho_A, r0, station_lat, station_lon, sidereal_days);

fprintf('\nMin-azimuth design (rho_A = %.3f kg/m^2)\n', rho_A);
fprintf(' v_t  (slope0) = %+7.3f m/s\n', dv_t_star);
fprintf('  0   (min annual amp) = %+7.3f deg\n', rad2deg(d_lambda0_star));
fprintf('Mean drift slope = %10.6f deg/day\n', metrics.slope_deg_per_day);
fprintf('Annual amp (Az) = %7.3f deg (peak-to-peak     %7.3f deg)\n', ...
        metrics.annual_amp_deg, 2*metrics.annual_amp_deg);
fprintf('Max |Az detrended| = %7.3f deg; RMS = %7.3f deg\n', ...
        metrics.max_abs_detrended_deg, metrics.rms_detrended_deg);
fprintf('After removing annual: Max = %7.3f deg; RMS = %7.3f deg\n', ...
        metrics.max_abs_after_annual_deg, metrics.rms_after_annual_deg);

% Plots
days = t_hist/86400;
figure('Name','Az over sidereal year','Color','w');
subplot(3,1,1);
plot(days, az_deg_raw, 'LineWidth', 1); grid on;
xlabel('Days'); ylabel('Az [deg]');
title('Raw azimuth');

subplot(3,1,2);
plot(days, az_deg_detr_lin, 'LineWidth', 1); grid on;
xlabel('Days'); ylabel('Az - linear [deg]');
title(sprintf('Linear detrended (slope = %.4f deg/day)', metrics.slope_deg_per_day));

subplot(3,1,3);
plot(days, resid_noannual_deg, 'LineWidth', 1); grid on;
xlabel('Days'); ylabel('Residual [deg]');
title(sprintf('After removing annual term (amp = %.3f deg)', metrics.annual_amp_deg));

result = struct('dv_t_mps', dv_t_star, ...
                'dlambda0_deg', rad2deg(d_lambda0_star), ...
                'metrics', metrics);
end

%%
function slope = slope_deg_per_day(dv_t_mps, d_lambda0, dt, ...
        mu, r_earth, w_earth, T, w_yr, alpha_r, rho_A, r0, station_lat, station_lon, sidereal_days)
% Estimate mean azimuth drift slope (deg/day) via 4-term regression:
% [1, t(day), sin(2 t /1y), cos(2 t /1y)] to separate the annual component.

[t_hist, az_deg] = forward_az( ...
    dv_t_mps, d_lambda0, dt, mu, r_earth, w_earth, T, w_yr, ...
    alpha_r, rho_A, r0, station_lat, station_lon);

tt_day = t_hist/86400;
y_deg = unwrap(deg2rad(az_deg))*180/pi; % unwrap in rad, then convert to deg
X = [ones(size(tt_day)), tt_day, ...
      sin(2*pi*tt_day/ sidereal_days), cos(2*pi*tt_day/ sidereal_days)];
b = X\y_deg;
slope = b(2); % deg/day
end

function A_deg = annual_amp_deg(dv_t_mps, d_lambda0, dt, ...
        mu, r_earth, w_earth, T, w_yr, alpha_r, rho_A, r0, station_lat, station_lon, sidereal_days)
```

```
% Return annual amplitude (deg) from the sin/cos coefficients of the 4-term fit.

[t_hist, az_deg] = forward_az( ...
    dv_t_mps, d_lambda0, dt, mu, r_earth, w_earth, T, w_yr, ...
    alpha_r, rho_A, r0, station_lat, station_lon);

tt_day = t_hist/86400;
y_deg = unwrap(deg2rad(az_deg))*180/pi;
X = [ones(size(tt_day)), tt_day, ...
          sin(2*pi*tt_day/ sidereal_days), cos(2*pi*tt_day/ sidereal_days)];
b = X\y_deg;
A_deg = hypot(b(3), b(4));
end

function [t_hist, az_deg, az_deg_detr_lin, resid_noannual_deg, m] = ...
    sim_and_metrics(dv_t_mps, d_lambda0, dt, ...
        mu, r_earth, w_earth, T, w_yr, alpha_r, rho_A, r0, station_lat, station_lon, sidereal_days)
% Final simulation and metric computation.

[t_hist, az_deg] = forward_az( ...
    dv_t_mps, d_lambda0, dt, mu, r_earth, w_earth, T, w_yr, ...
    alpha_r, rho_A, r0, station_lat, station_lon);

tt_day = t_hist/86400;
y_deg  = unwrap(deg2rad(az_deg))*180/pi;

% Remove linear trend
P = polyfit(tt_day, y_deg, 1);
trend = polyval(P, tt_day);
az_deg_detr_lin = y_deg - trend;

% Remove annual term (sin/cos at 1/year)
X = [sin(2*pi*tt_day/ sidereal_days), cos(2*pi*tt_day/ sidereal_days)];
ab = X\az_deg_detr_lin;
annual = X*ab;
resid = az_deg_detr_lin - annual;

m = struct();
m.slope_deg_per_day = P(1);
m.annual_amp_deg = hypot(ab(1), ab(2));
m.max_abs_detrended_deg = max(abs(az_deg_detr_lin));
m.rms_detrended_deg = sqrt(mean(az_deg_detr_lin.^2));
m.max_abs_after_annual_deg = max(abs(resid));
m.rms_after_annual_deg = sqrt(mean(resid.^2));

resid_noannual_deg = resid;
end

function [t_hist, az_deg] = forward_az( ...
        dv_t_mps, d_lambda0, dt, mu, r_earth, w_earth, T, w_yr, ...
        alpha_r, rho_A, r0, station_lat, station_lon)
% Earth-centered initial (ECI)
% Earth-centered - Earth-fixed (ECEF)
% Planar ECI -> ECEF -> East North Up -> azimuth time-series over T seconds.
% East North Up (ENU)

% Propagate in Cartesian with RK4 (planar, solar pressure rotating with sun)
[s_hist, t_hist] = propagate_rk4( ...
    init_state(dv_t_mps, d_lambda0, r0, mu, station_lon), ...
    dt, T, mu, alpha_r, rho_A, w_yr);

% ECI -> ECEF
x = s_hist(:,1);  y = s_hist(:,2);
theta_e = w_earth * t_hist;
c = cos(theta_e); s = sin(theta_e);
xe = c.*x + s.*y;
ye = -s.*x + c.*y;
ze = zeros(size(xe)); % planar z = 0
```

```
% Station ECEF
[xs, ys, zs] = station_ecef(r_earth, station_lat, station_lon);

% Line-of-sight in ECEF
dx = xe - xs; dy = ye - ys; dz = ze - zs;

% ECEF -> East North Up
[E, N, U] = ecef_to_enu(dx, dy, dz, station_lat, station_lon); %#ok<ASGLU>

% Look angles
azimuth = atan2(E, N);            % [rad], 0..2
azimuth = mod(azimuth, 2*pi);

az_deg = rad2deg(azimuth);
end

function s0 = init_state(dv_t_mps, d_lambda0, r0, mu, station_lon)
% Initial planar ECI state from  v_t  and   0  .

v_circ = sqrt(mu/r0); % [km/s]
v_t = v_circ + dv_t_mps/1000; % [km/s]
theta0 = station_lon + d_lambda0; % start longitude offset

x0 = r0*cos(theta0);
y0 = r0*sin(theta0);
vx0 = -v_t*sin(theta0);
vy0 =  v_t*cos(theta0);

s0 = [x0; y0; vx0; vy0];
end

function [S, t_hist] = propagate_rk4(s0, dt, T, mu, alpha_r, rho_A, w_yr)
% 4th-order  Runge Kutta  propagation (planar 2D + rotating solar-pressure accel).

n = floor(T/dt);
t_hist = (0:n)'*dt;

S = zeros(n+1, 4);
S(1,:) = s0.';
a_solar = 9.08e-9 * alpha_r / rho_A; % [km/s^2] (from 9.08e-6 m/s^2)

for k = 1:n
    tk = t_hist(k);
    s = S(k,:).';

    k1 = f(tk,           s);
    k2 = f(tk + 0.5*dt,  s + 0.5*dt*k1);
    k3 = f(tk + 0.5*dt,  s + 0.5*dt*k2);
    k4 = f(tk + dt,      s + dt*k3);

    S(k+1,:) = (s + (dt/6)*(k1 + 2*k2 + 2*k3 + k4)).';
end

    function ds = f(ti, s)
        x=s(1); y=s(2); vx=s(3); vy=s(4);
        r = hypot(x,y);
        aG = -mu/r^3 * [x; y]; % gravity
        alpha = w_yr * ti; % sun direction angle
        aS = a_solar * [cos(alpha); sin(alpha)]; % solar-pressure accel
        ds = [vx; vy; aG(1)+aS(1); aG(2)+aS(2)];
    end
end

function [xs, ys, zs] = station_ecef(r_earth, station_lat, station_lon)
% Spherical Earth station to ECEF.
xs = r_earth*cos(station_lat)*cos(station_lon);
ys = r_earth*cos(station_lat)*sin(station_lon);
```

```
zs = r_earth*sin(station_lat);
end

function [E, N, U] = ecef_to_enu(dx, dy, dz, station_lat, station_lon)
% ECEF delta -> local East North Up.
sl = sin(station_lon);
cl = cos(station_lon);
sp = sin(station_lat);
cp = cos(station_lat);
E = -sl.*dx +  cl.*dy;
N = -sp.*cl.*dx - sp.*sl.*dy + cp.*dz;
U = cp.*cl.*dx + cp.*sl.*dy + sp.*dz;
end
```