ASEN 6080 HW 5

- Ian Faber, 108577813

1. a. Implement the SRIF without process noise, using Householder for the orthogonal transformation

See PDF for code!

b. Process the same data from HW2 & verify that the performance is similar to the CKF without process noise

See PDF for plots, performance is nearly identical!

c. Now, don't force $\bar{R}_i$ to be upper triangular, what happens to the filter, and why?

If I don't force $\bar{R}_i$ to be upper triangular, the filter behaves exactly the same. This makes sense, as the measurement update of the SRIF still uses a Householder transformation that ensures $\hat{R}_i$ is upper triangular. If we wanted to save $\bar{P}_i$ then, it might matter that $\bar{R}_i$ is upper triangular in general, this doesn't seem to be a requirement though.

ASEN 6080 HW 5 Problem 1 Main Script

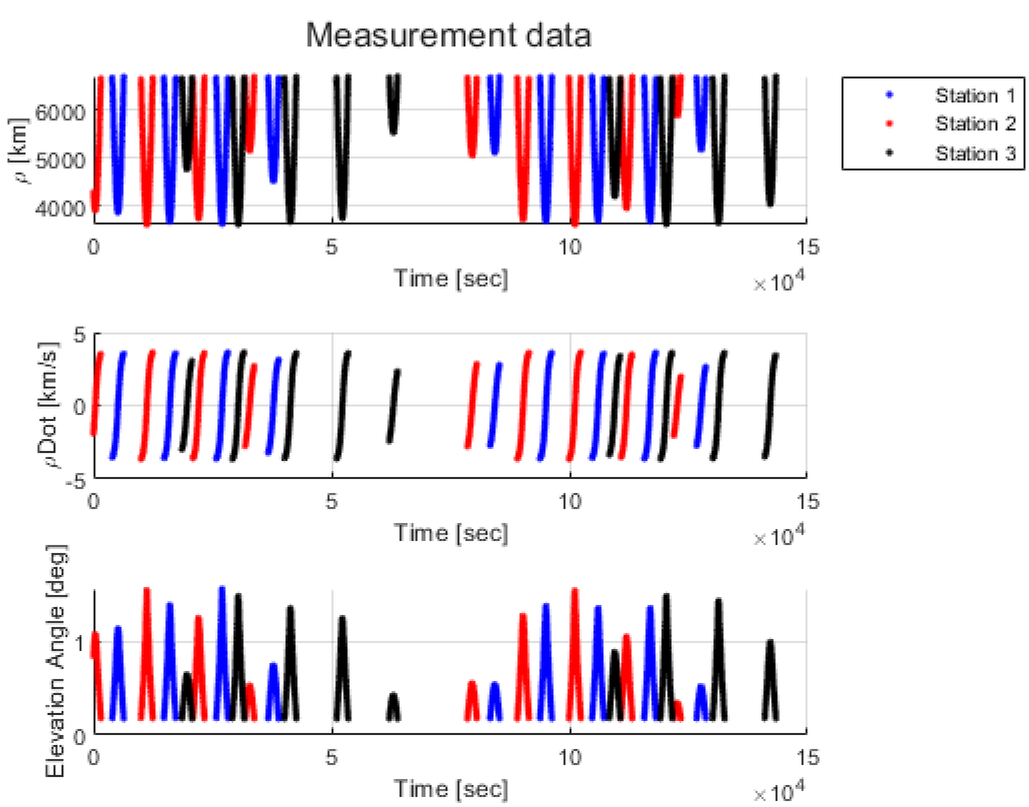
Table of Contents

By: Ian Faber

Housekeeping

Setup

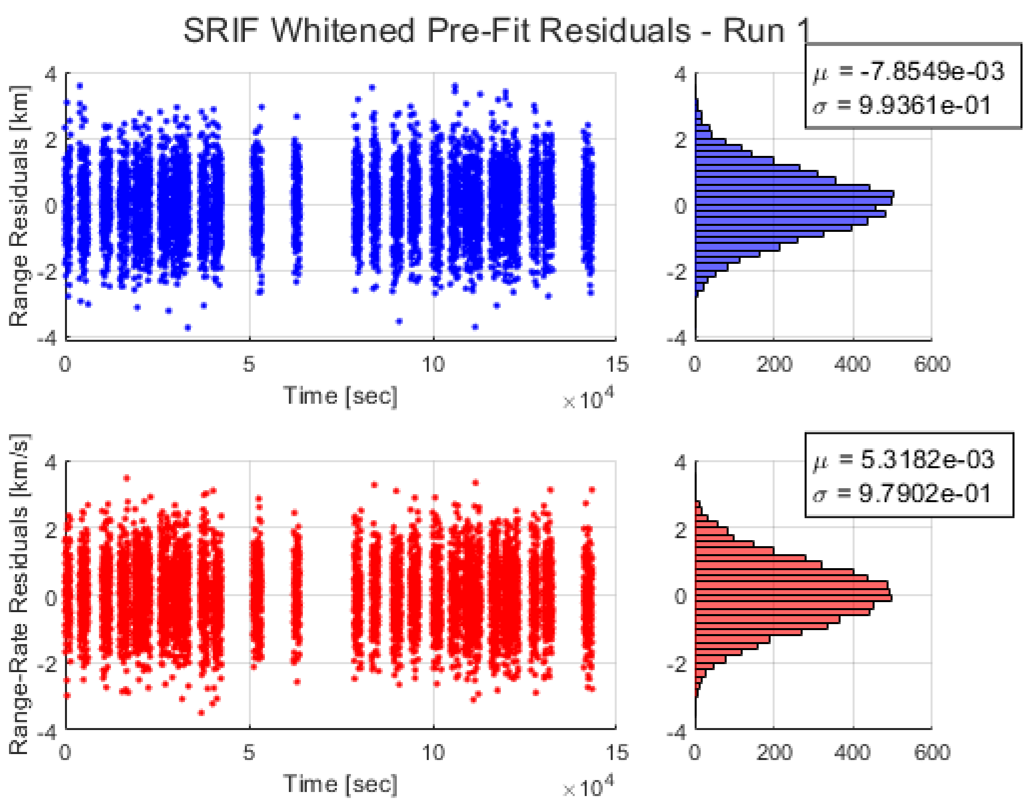
Make truth data

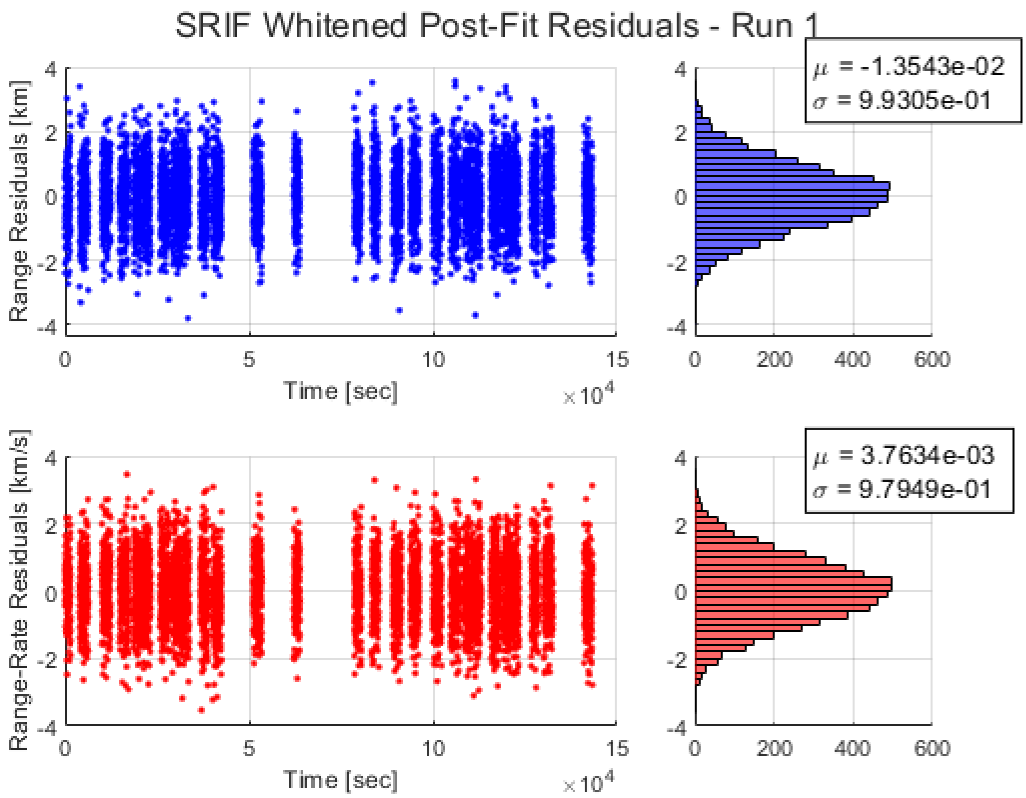


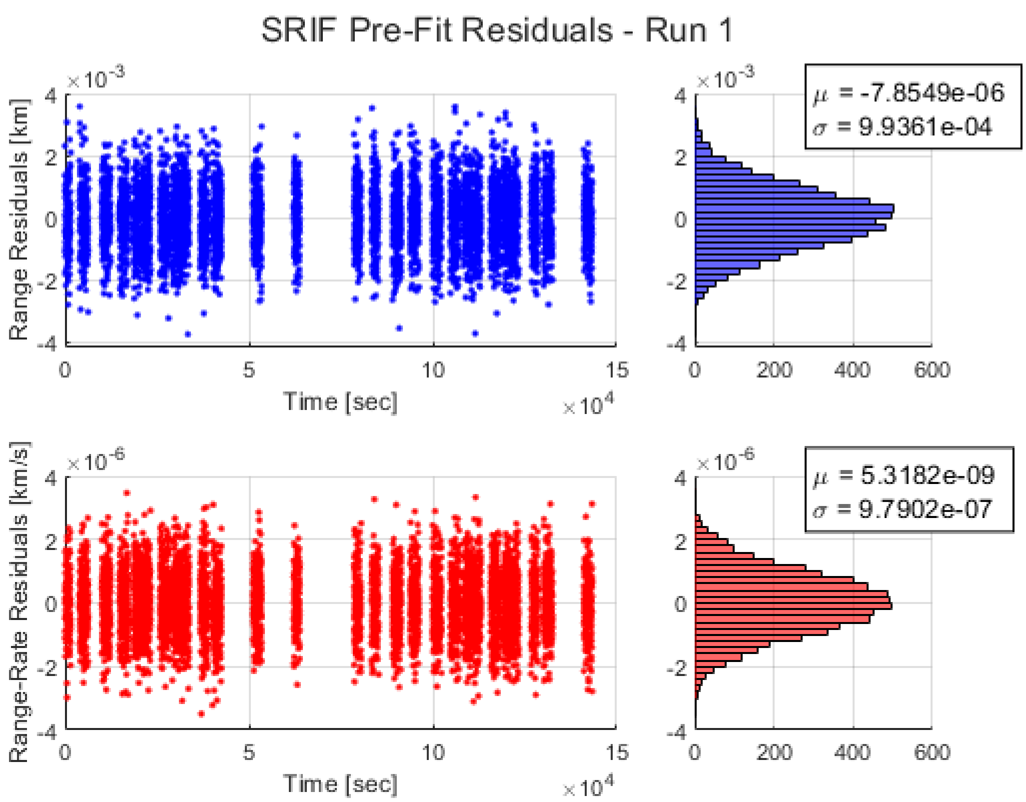
Problem 1a. Filter setup

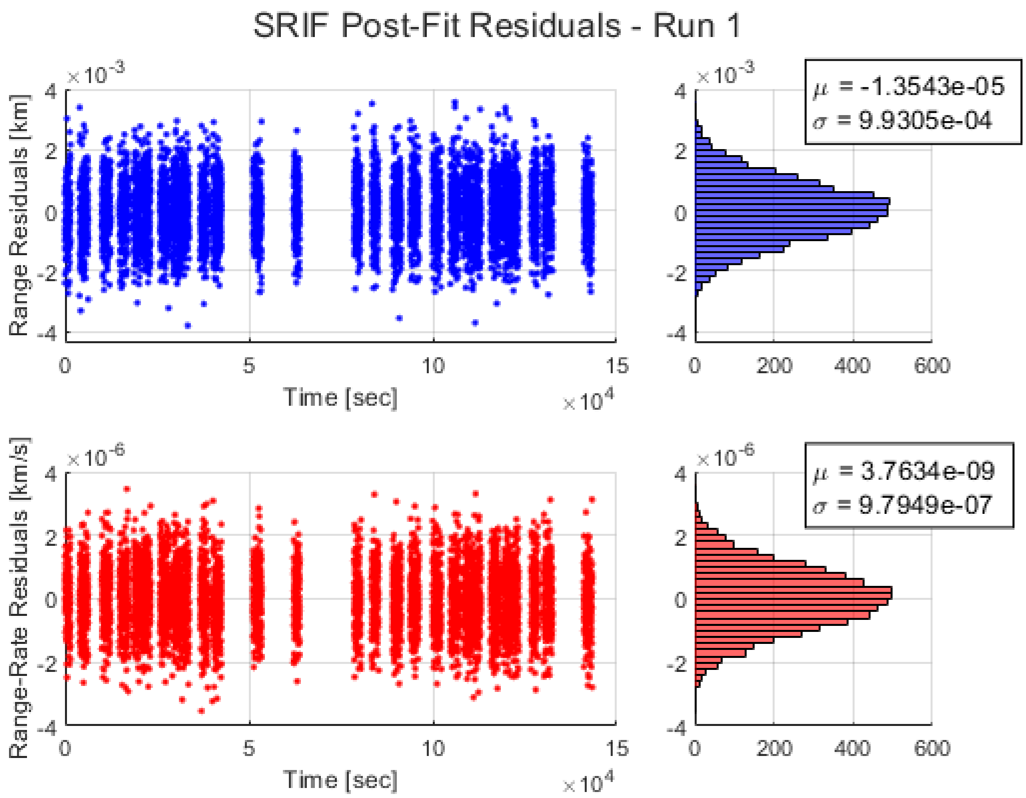
Problem 1b. Compare SRIF to LKF without process noise

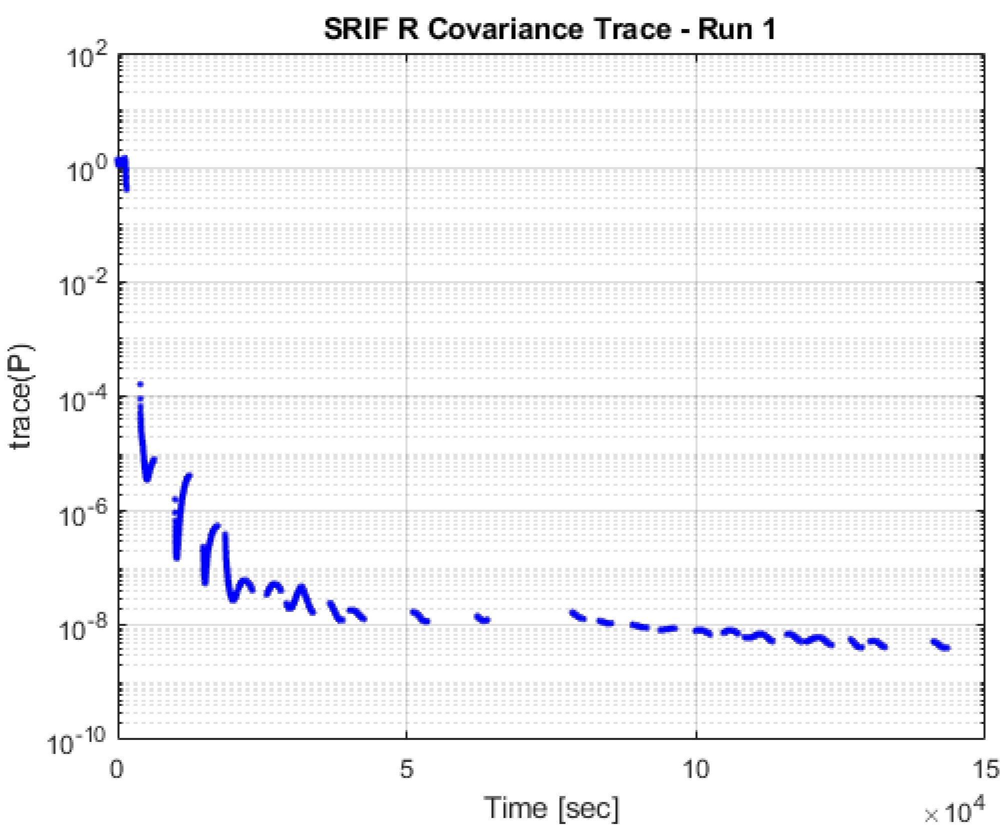
```
1b. Comparing SRIF to LKF

    Running SRIF:
Prefit RMS: 0.9863, Postfit RMS: 0.9863. Hit max SRIF iterations. Runs so
far: 1
Final whitened prefit RMS: 0.9863. Hit maximum number of 1 runs
Final whitened postfit RMS: 0.9863. Hit maximum number of 1 runs
Final prefit RMS: 0.9863. Hit maximum number of 1 runs
Final postfit RMS: 0.9863. Hit maximum number of 1 runs

    Running LKF:
Prefit RMS: 0.9863, Postfit RMS: 0.9831. Hit max LKF iterations. Runs so
far: 1
Final prefit RMS: 0.9863. Hit maximum number of 1 runs
Final postfit RMS: 0.9831. Hit maximum number of 1 runs
```

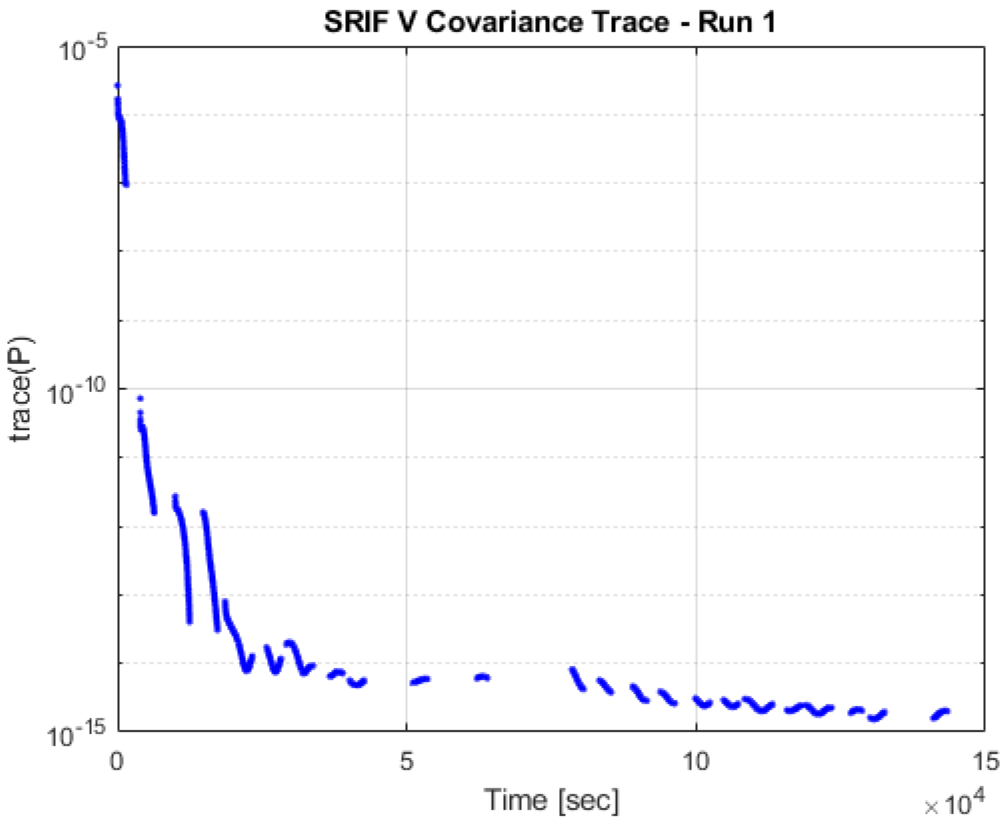


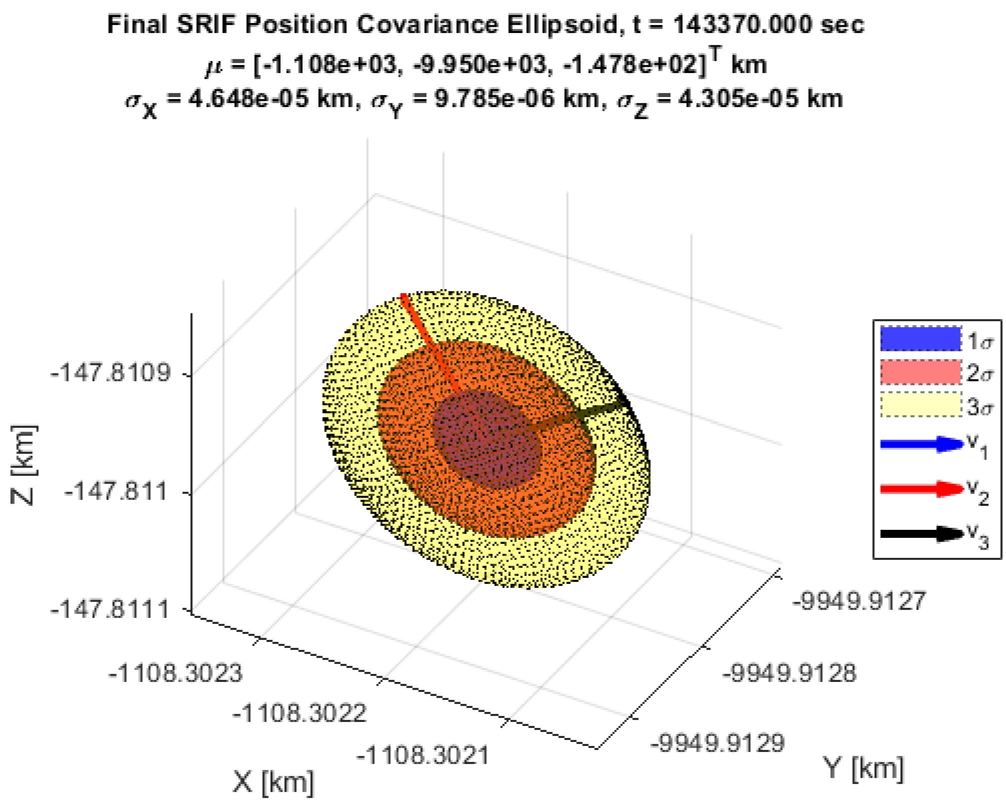

SRIF Whitened Post-Fit Residuals - Run 1
Range Residuals [km]
Time [sec]
$\times 10^4$
μ = -1.3543e-02
σ = 9.9305e-01
Range-Rate Residuals [km/s]
Time [sec]
$\times 10^4$
μ = 3.7634e-03
σ = 9.7949e-01

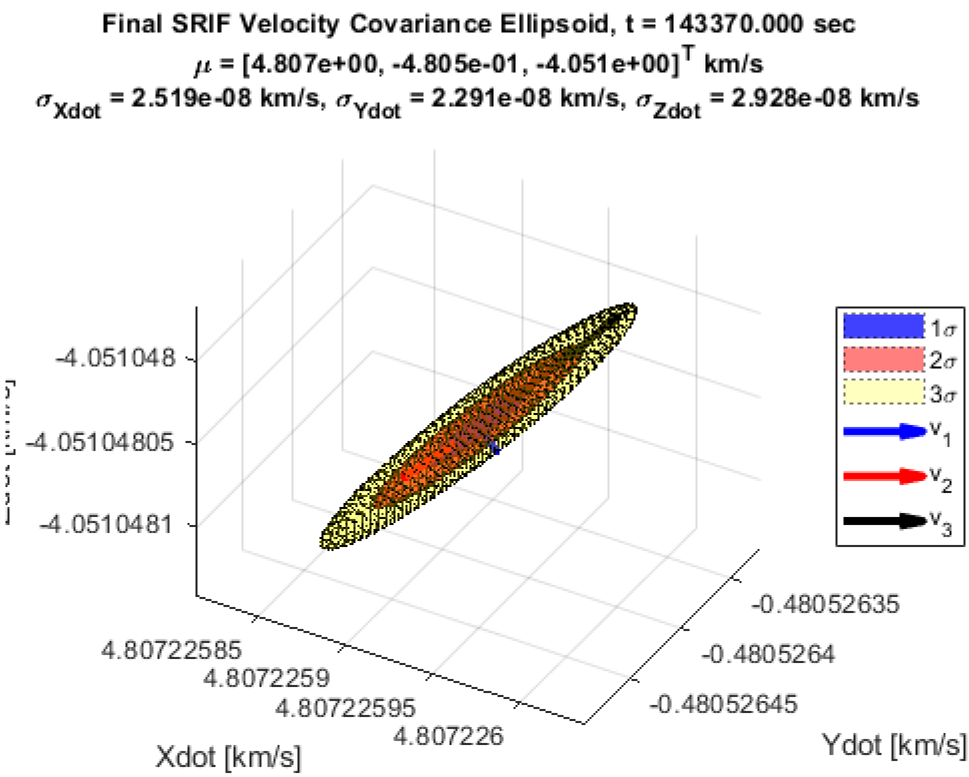

SRIF Pre-Fit Residuals - Run 1
Range Residuals [km]
$\times 10^{-3}$
Time [sec]
$\times 10^4$
$\times 10^{-3}$
μ = -7.8549e-06
σ = 9.9361e-04
Range-Rate Residuals [km/s]
$\times 10^{-6}$
Time [sec]
$\times 10^4$
$\times 10^{-6}$
μ = 5.3182e-09
σ = 9.7902e-07

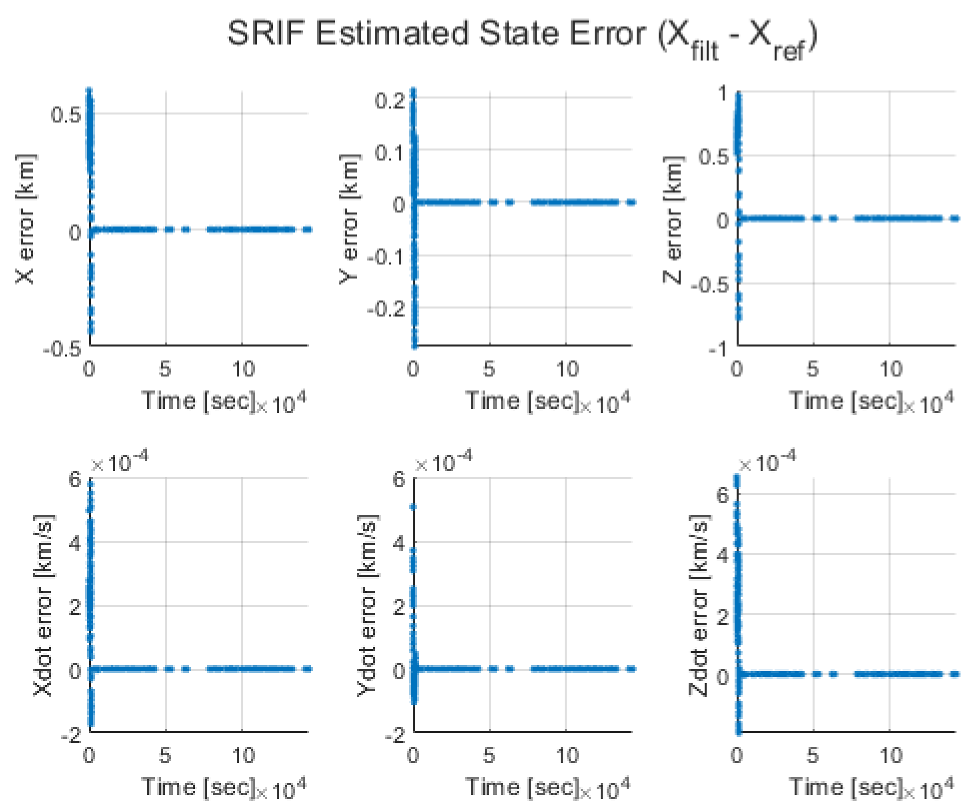

SRIF Post-Fit Residuals - Run 1
Range Residuals [km]
×10^-3
Time [sec]
×10^4
μ = -1.3543e-05
σ = 9.9305e-04
Range-Rate Residuals [km/s]
×10^-6
μ = 3.7634e-09
σ = 9.7949e-07

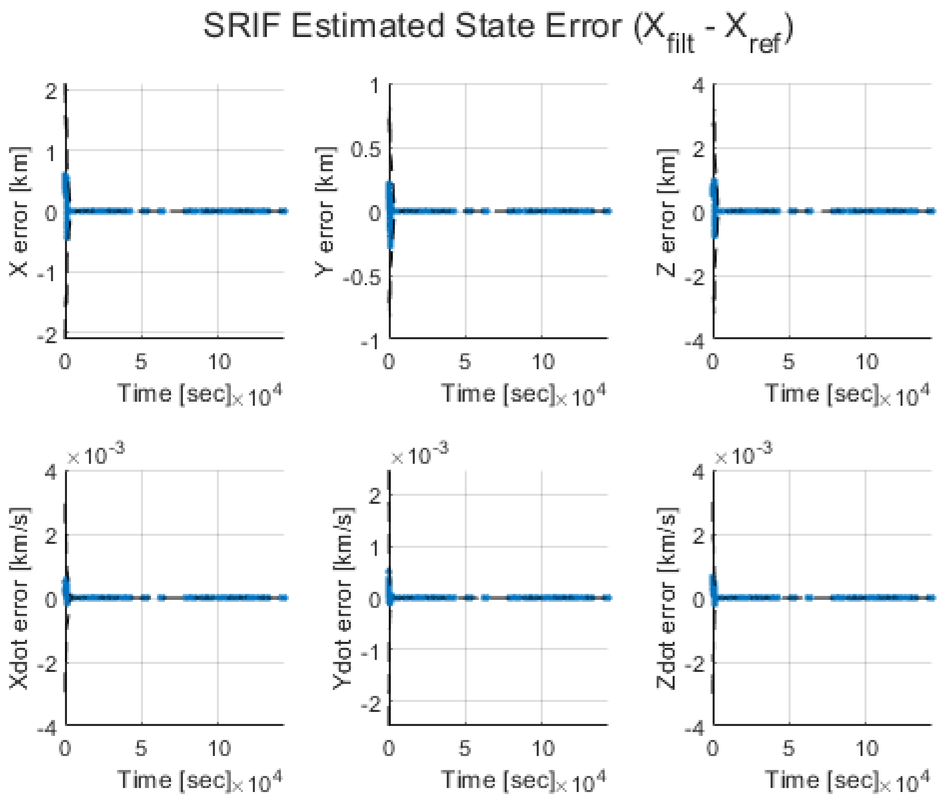

SRIF R Covariance Trace - Run 1
trace(P)
Time [sec]
×10^4

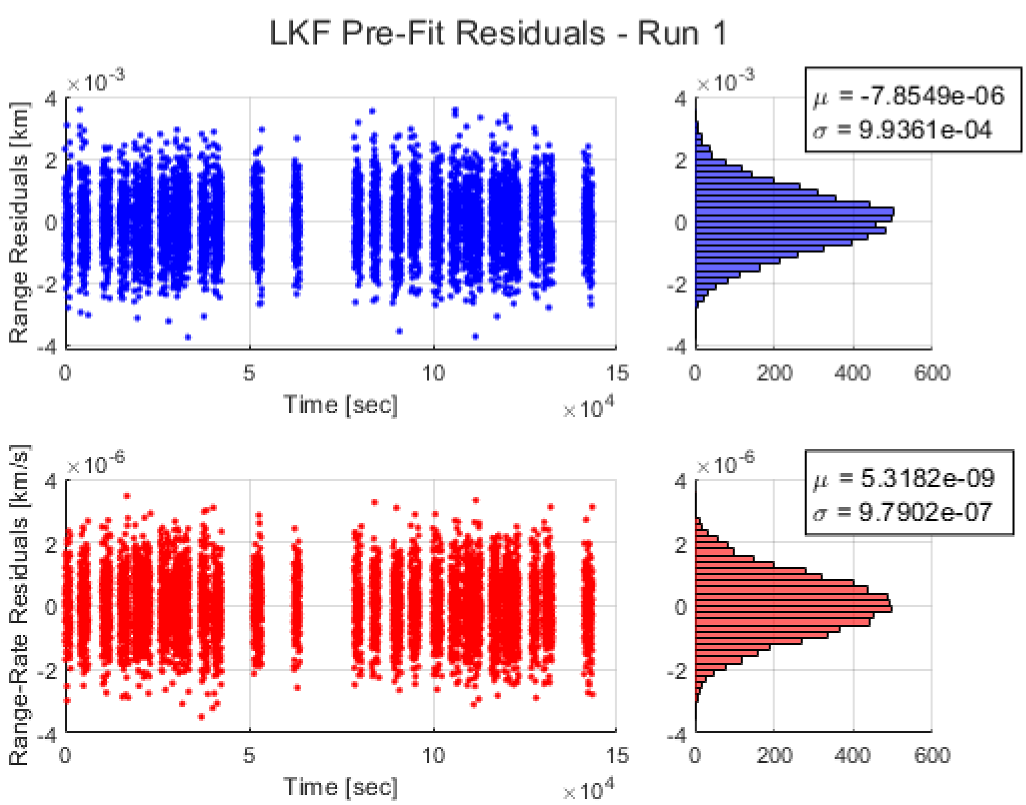

SRIF V Covariance Trace - Run 1
trace(P)
Time [sec]
$\times 10^4$
10^{-5}
10^{-10}
10^{-15}
0
5
10
15

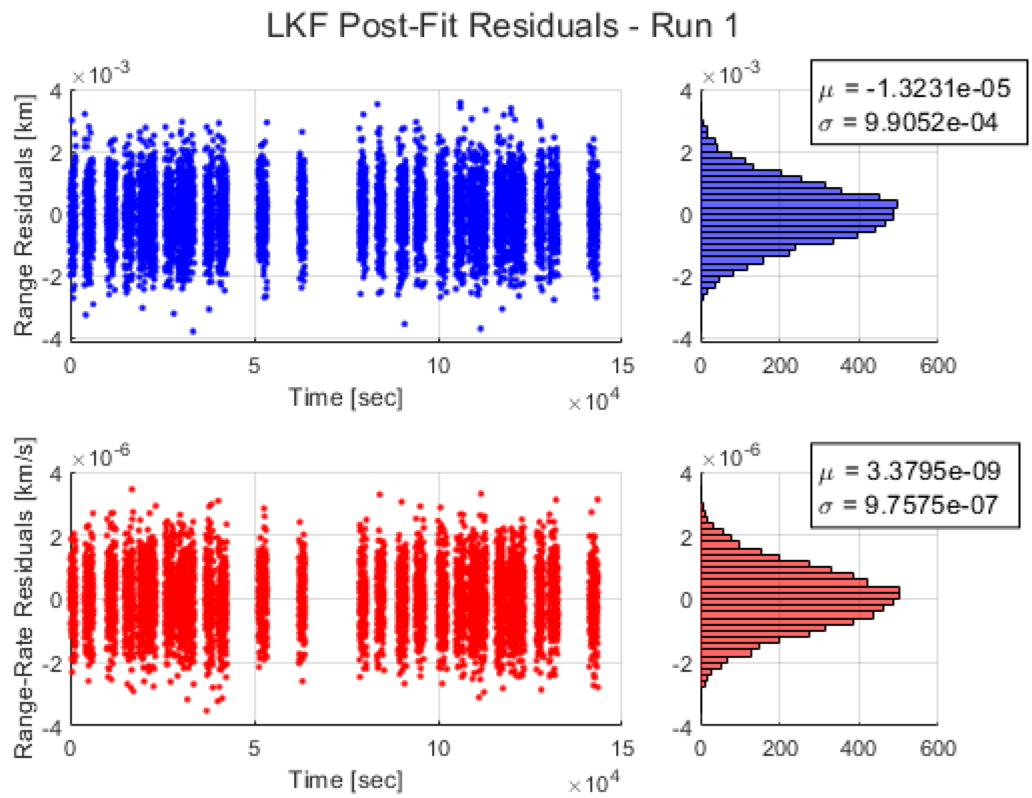

Final SRIF Position Covariance Ellipsoid, t = 143370.000 sec
μ = [-1.108e+03, -9.950e+03, -1.478e+02]^T km
σ_X = 4.648e-05 km, σ_Y = 9.785e-06 km, σ_Z = 4.305e-05 km
1σ
2σ
3σ
v_1
v_2
v_3
Z [km]
-147.8109
-147.811
-147.8111
X [km]
-1108.3023
-1108.3022
-1108.3021
Y [km]
-9949.9127
-9949.9128
-9949.9129

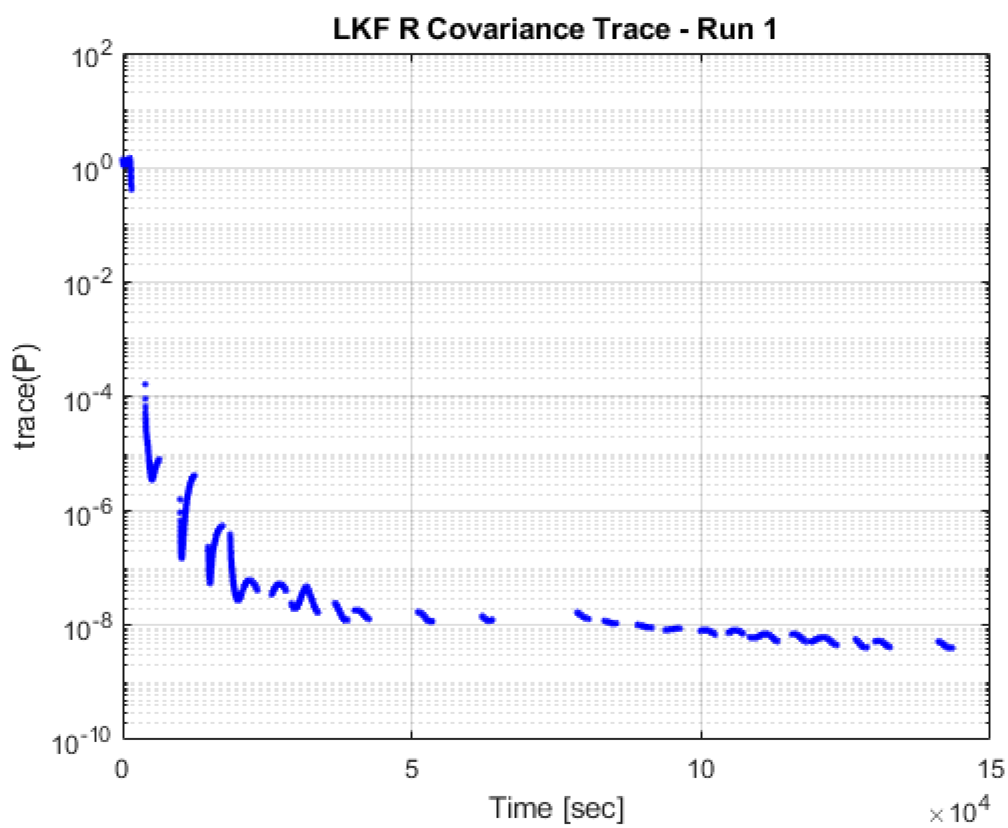

Final SRIF Velocity Covariance Ellipsoid, t = 143370.000 sec
μ = [4.807e+00, -4.805e-01, -4.051e+00]^T km/s
σXdot = 2.519e-08 km/s, σYdot = 2.291e-08 km/s, σZdot = 2.928e-08 km/s
-4.051048
-4.05104805
-4.0510481
4.80722585
4.8072259
4.80722595
4.807226
Xdot [km/s]
-0.48052635
-0.4805264
-0.48052645
Ydot [km/s]
1σ
2σ
3σ
v1
v2
v3

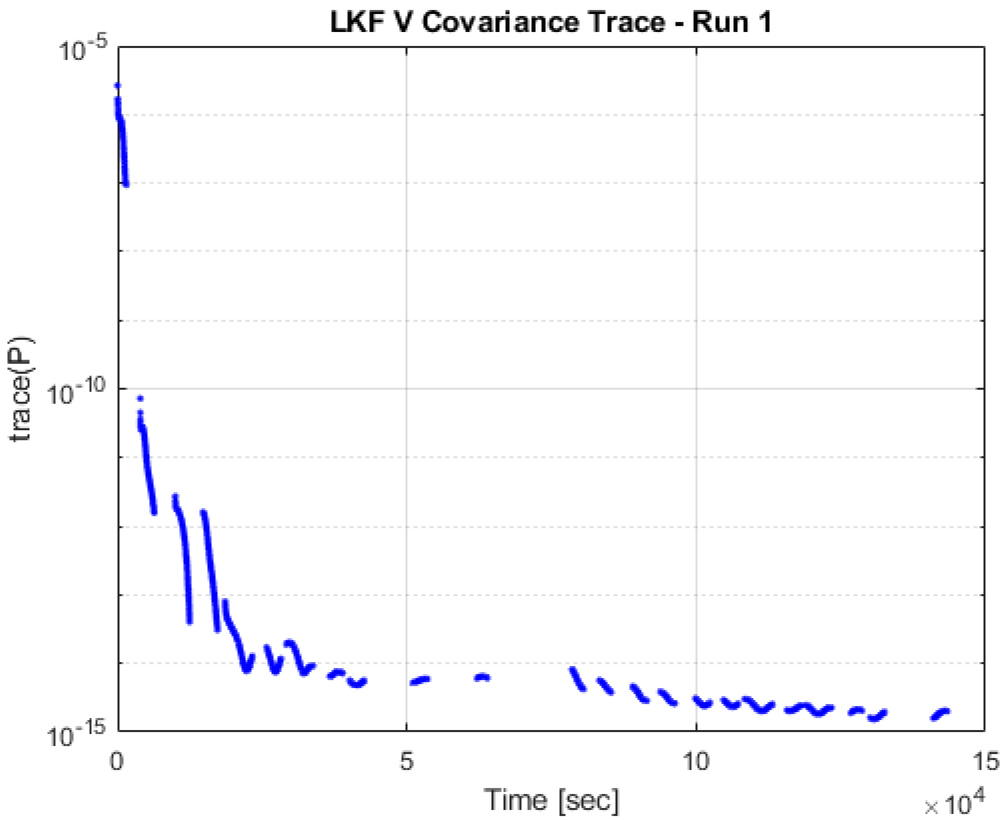

SRIF Estimated State Error (Xfilt - Xref)
X error [km]
Y error [km]
Z error [km]
Xdot error [km/s]
Ydot error [km/s]
Zdot error [km/s]
Time [sec]
×10^4
×10^-4

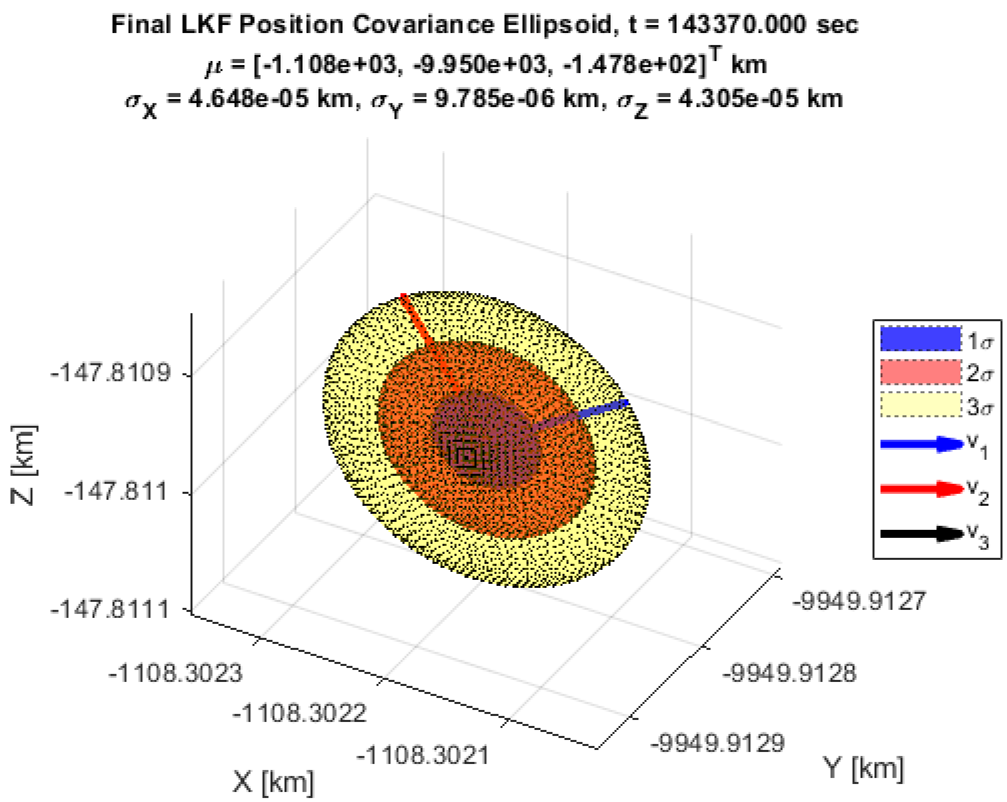

SRIF Estimated State Error (X_{filt} - X_{ref})
X error [km]
Y error [km]
Z error [km]
Xdot error [km/s]
Ydot error [km/s]
Zdot error [km/s]
Time [sec]
$\times 10^4$
$\times 10^{-3}$

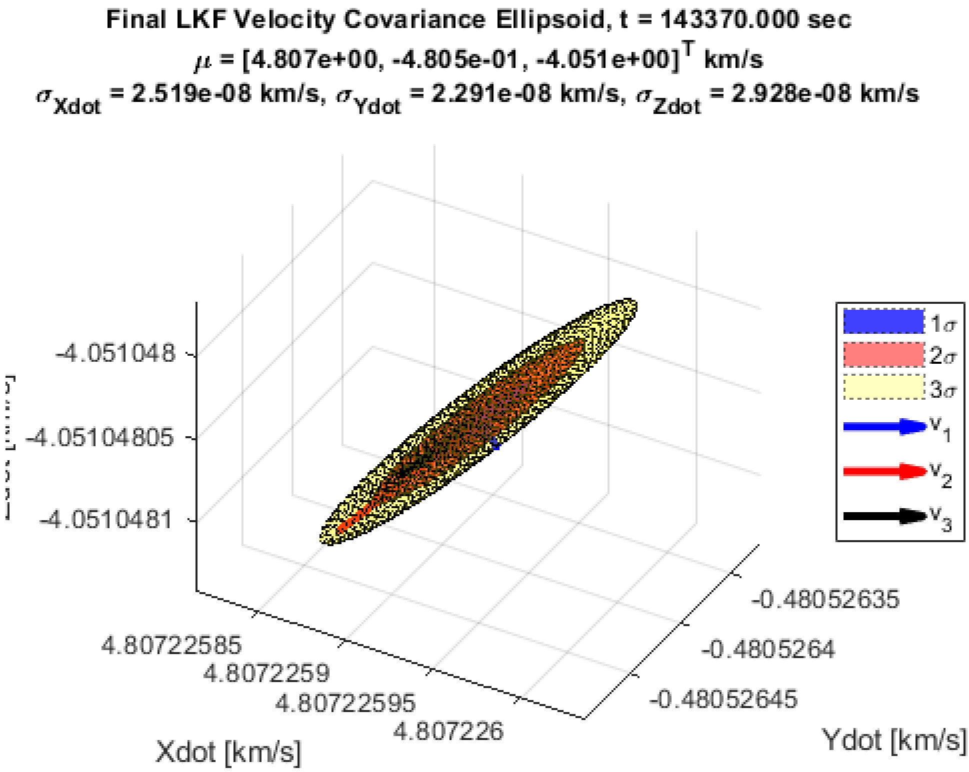

LKF Pre-Fit Residuals - Run 1
Range Residuals [km]
Range-Rate Residuals [km/s]
Time [sec]
$\times 10^4$
$\times 10^{-3}$
$\times 10^{-6}$
μ = -7.8549e-06
σ = 9.9361e-04
μ = 5.3182e-09
σ = 9.7902e-07

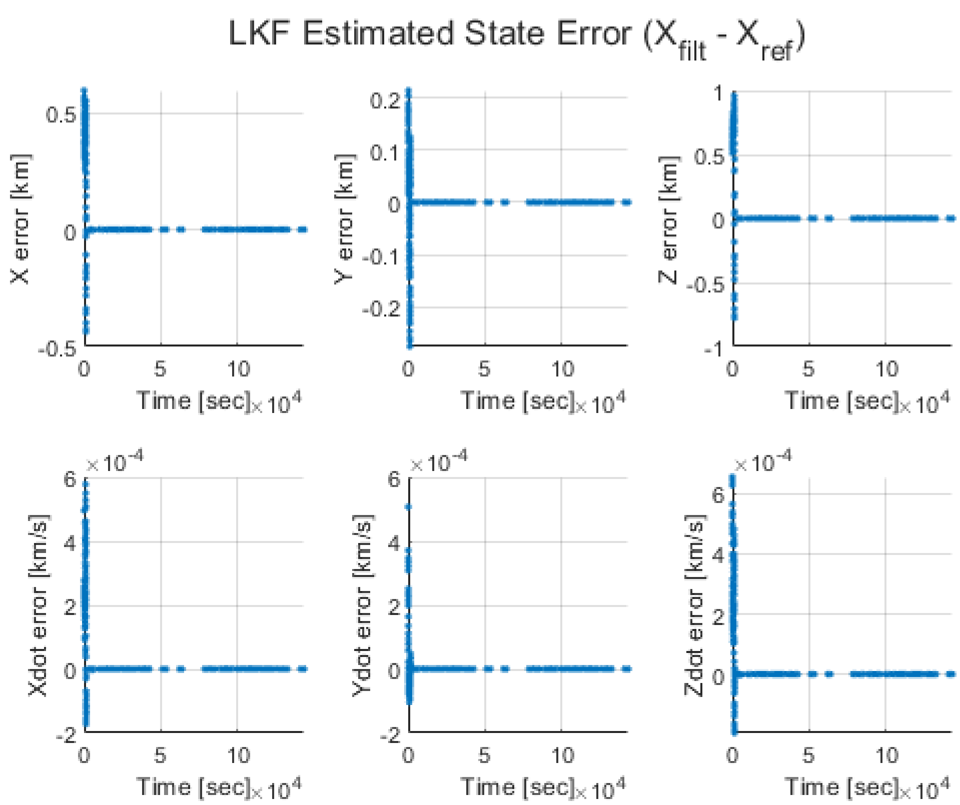

LKF Post-Fit Residuals - Run 1
Range Residuals [km]
Range-Rate Residuals [km/s]
Time [sec]
μ = -1.3231e-05
σ = 9.9052e-04
μ = 3.3795e-09
σ = 9.7575e-07

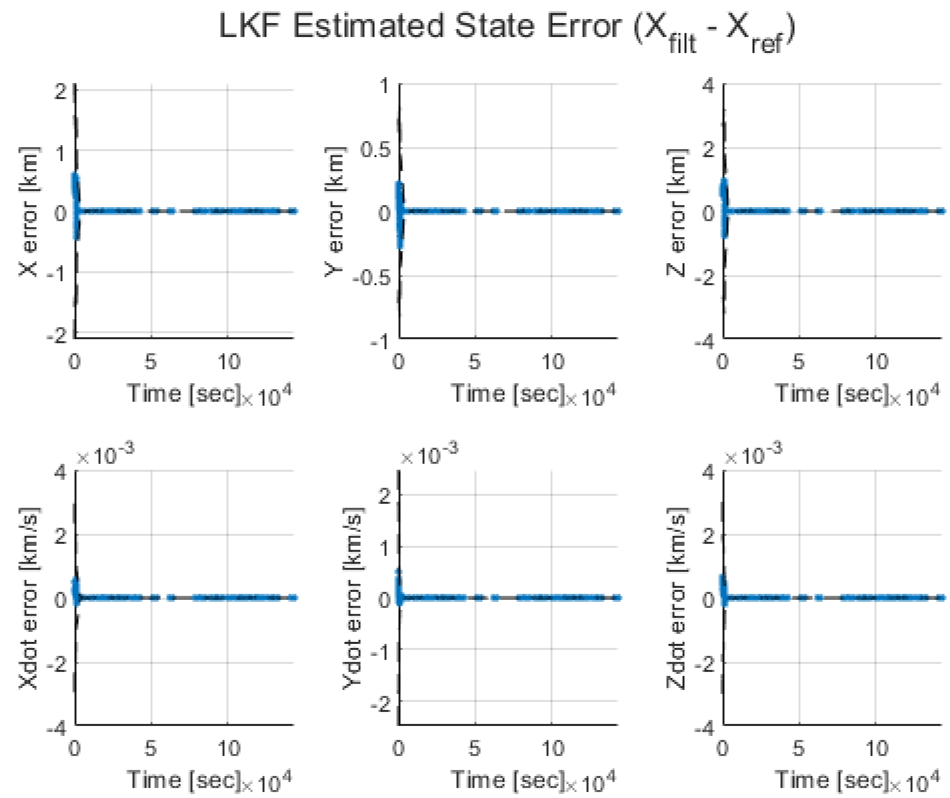

LKF R Covariance Trace - Run 1
trace(P)
Time [sec]

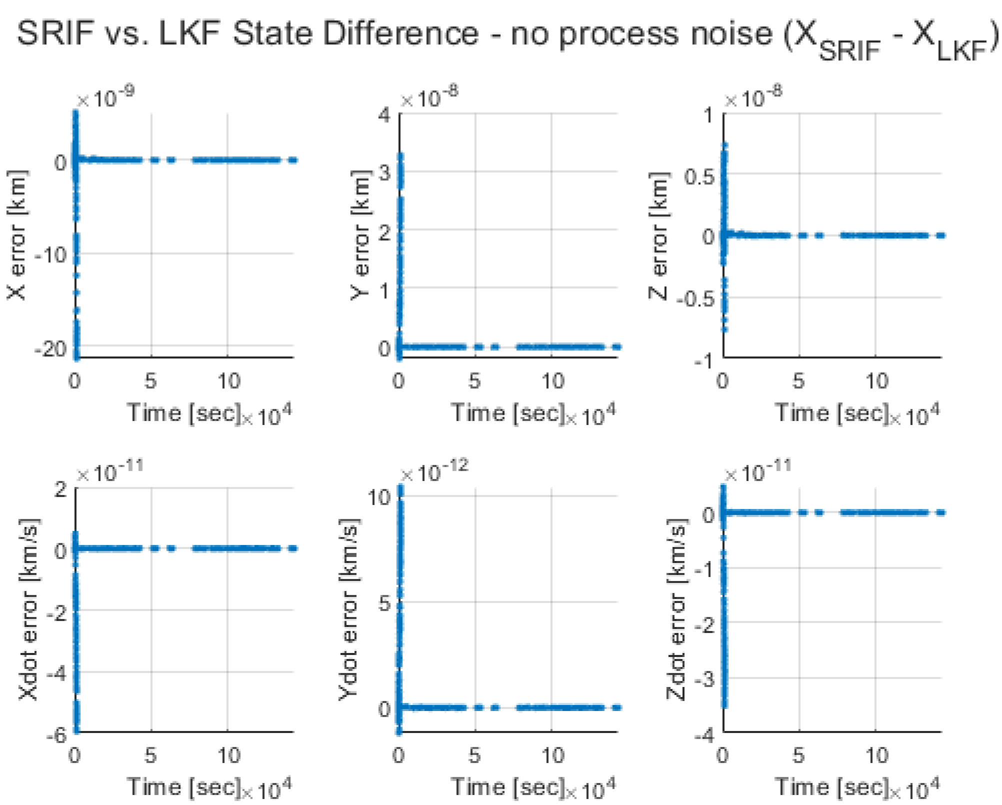

LKF V Covariance Trace - Run 1
trace(P)
Time [sec]
$\times 10^4$


Final LKF Position Covariance Ellipsoid, t = 143370.000 sec
μ = [-1.108e+03, -9.950e+03, -1.478e+02]^T km
σ_X = 4.648e-05 km, σ_Y = 9.785e-06 km, σ_Z = 4.305e-05 km
Z [km]
X [km]
Y [km]
1σ
2σ
3σ
v_1
v_2
v_3


Final LKF Velocity Covariance Ellipsoid, t = 143370.000 sec
μ = [4.807e+00, -4.805e-01, -4.051e+00]^T km/s
σ_{Xdot} = 2.519e-08 km/s, σ_{Ydot} = 2.291e-08 km/s, σ_{Zdot} = 2.928e-08 km/s
1σ
2σ
3σ
v_1
v_2
v_3
-4.051048
-4.05104805
-4.0510481
4.80722585
4.8072259
4.80722595
4.807226
Xdot [km/s]
-0.48052635
-0.4805264
-0.48052645
Ydot [km/s]

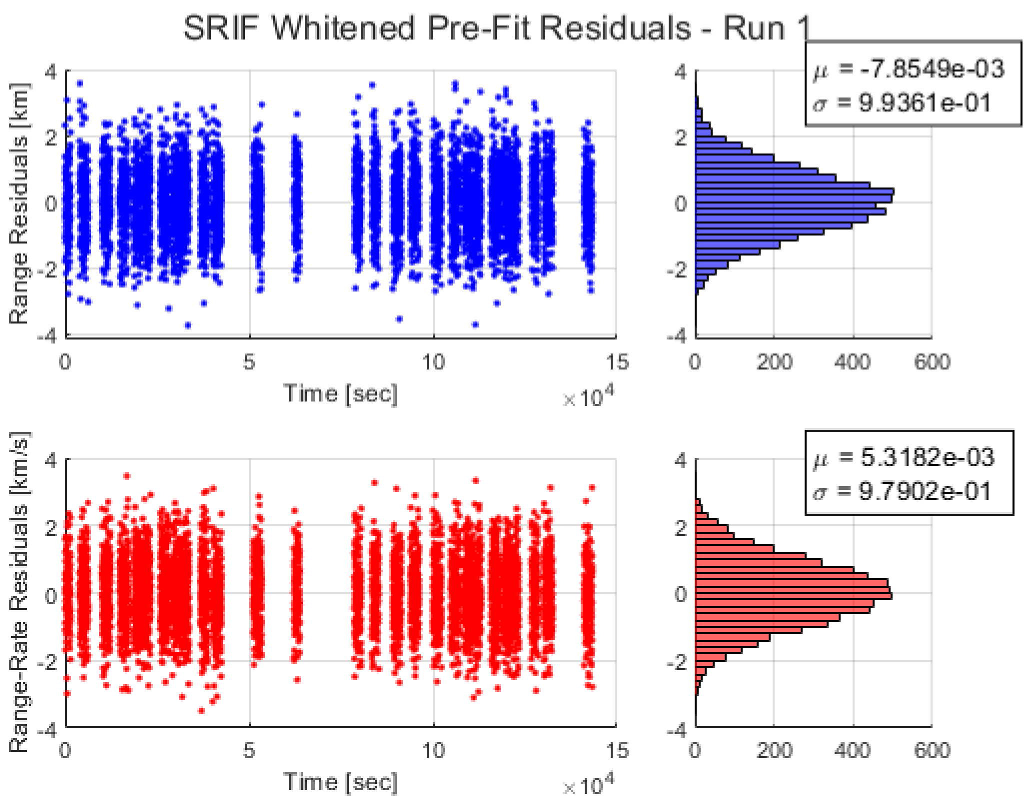

LKF Estimated State Error (X_{filt} - X_{ref})
X error [km]
Y error [km]
Z error [km]
Xdot error [km/s]
Ydot error [km/s]
Zdot error [km/s]
Time [sec]
$\times 10^4$
$\times 10^{-4}$

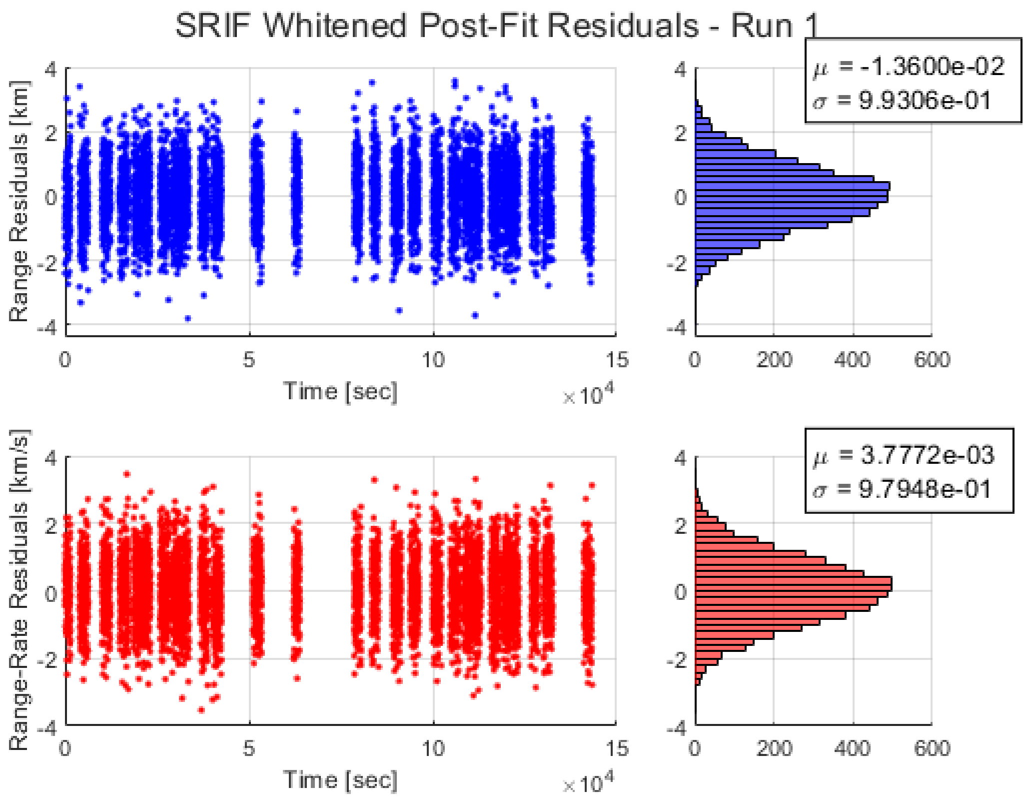

LKF Estimated State Error (X_filt - X_ref)
X error [km]
Y error [km]
Z error [km]
Xdot error [km/s]
Ydot error [km/s]
Zdot error [km/s]
Time [sec]
×10^4
×10^-3

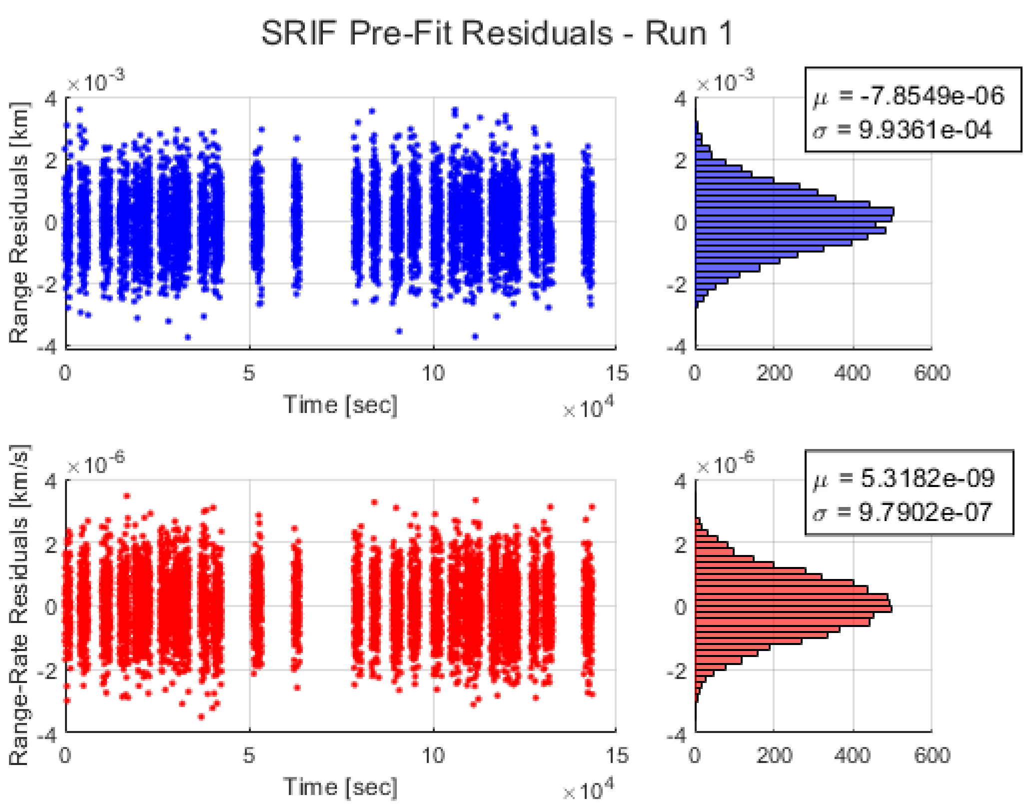

SRIF vs. LKF State Difference - no process noise (X_SRIF - X_LKF)
X error [km]
Y error [km]
Z error [km]
Xdot error [km/s]
Ydot error [km/s]
Zdot error [km/s]
Time [sec]
×10^4
×10^-9
×10^-8
×10^-11
×10^-12

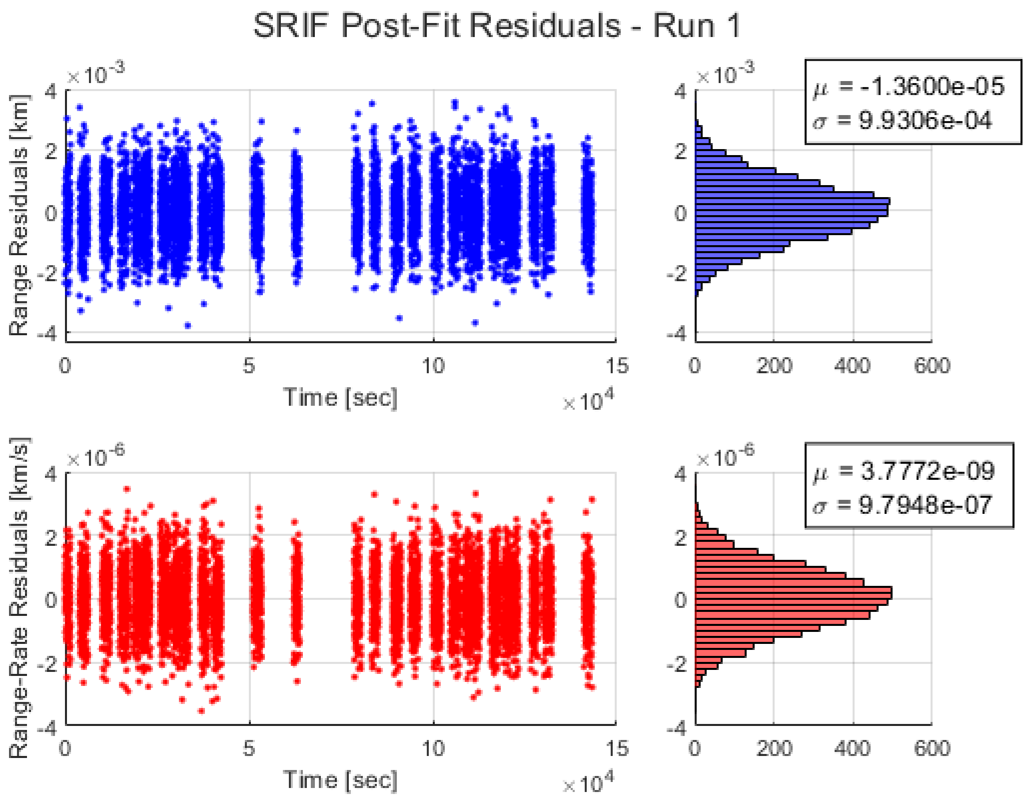
Problem 1c. Run SRIF without forcing Rbar to be upper triangular

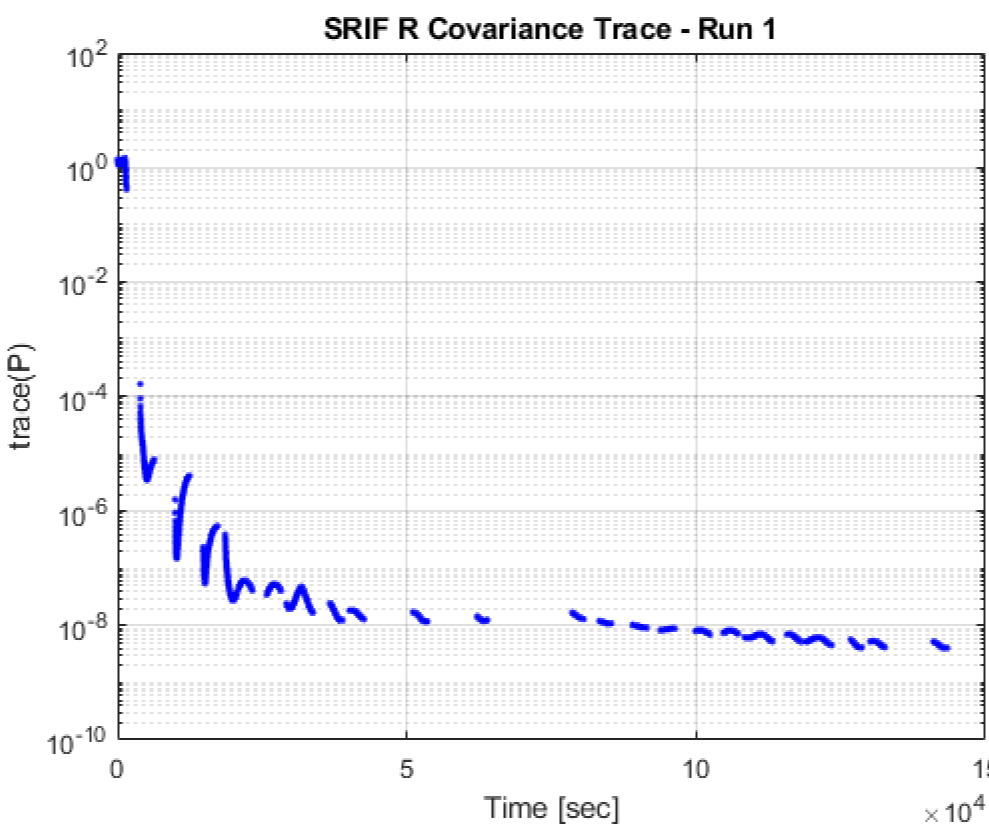
```
1c. Running SRIF without forcing Rbar to be upper triangular

    Running SRIF:
Prefit RMS: 0.9863, Postfit RMS: 0.9863. Hit max SRIF iterations. Runs so
far: 1
Final whitened prefit RMS: 0.9863. Hit maximum number of 1 runs
Final whitened postfit RMS: 0.9863. Hit maximum number of 1 runs
Final prefit RMS: 0.9863. Hit maximum number of 1 runs
Final postfit RMS: 0.9863. Hit maximum number of 1 runs
```

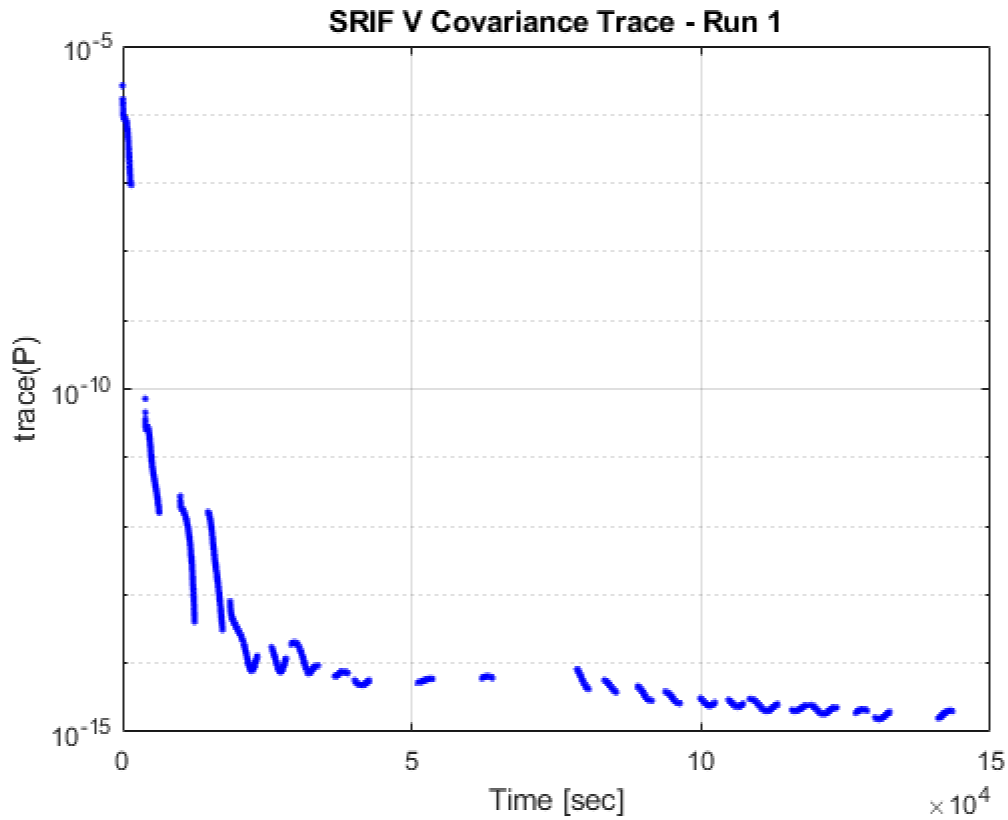


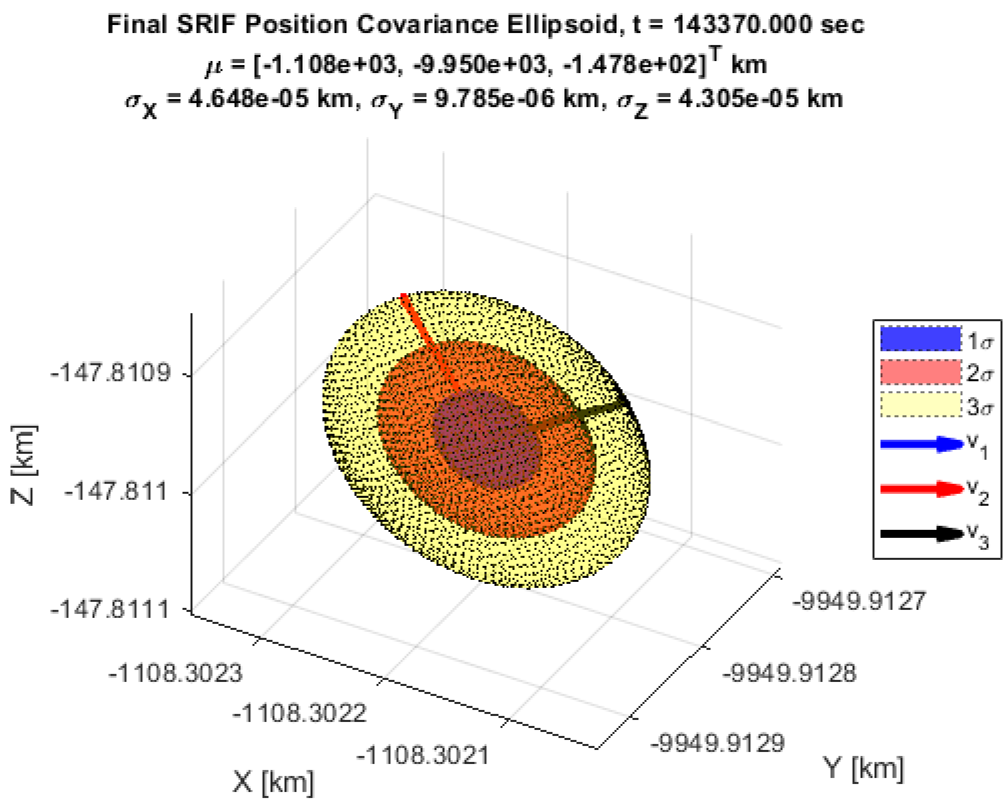

SRIF Whitened Post-Fit Residuals - Run 1
Range Residuals [km]
Time [sec]
$\times 10^4$
μ = -1.3600e-02
σ = 9.9306e-01
Range-Rate Residuals [km/s]
Time [sec]
$\times 10^4$
μ = 3.7772e-03
σ = 9.7948e-01

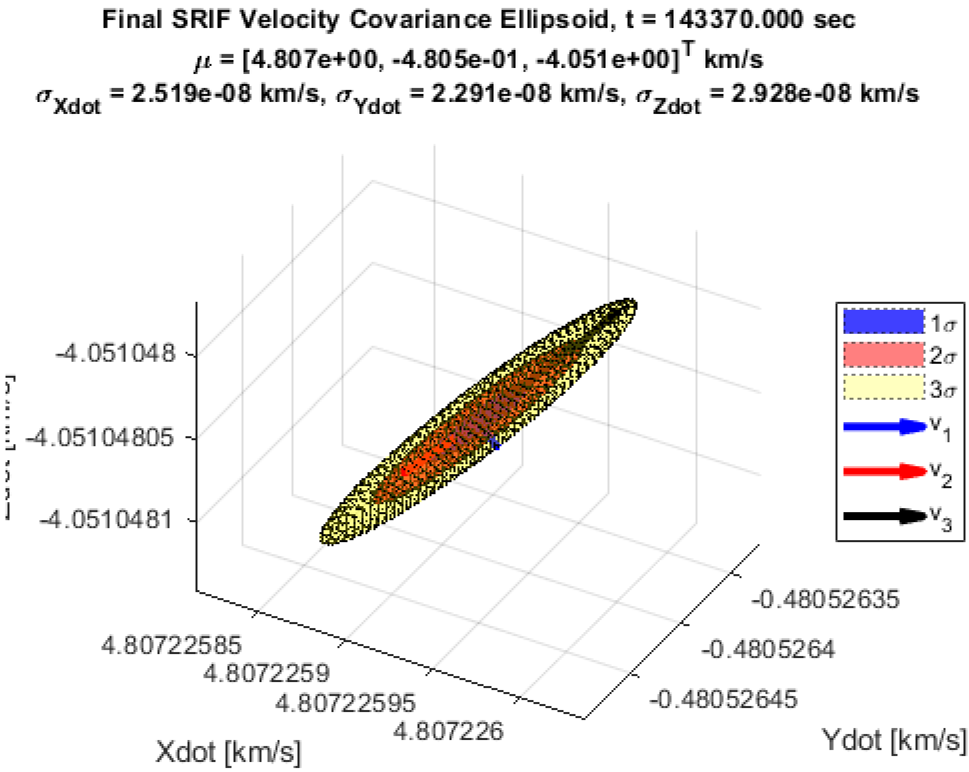

SRIF Pre-Fit Residuals - Run 1
$\times 10^{-3}$
Range Residuals [km]
Time [sec]
$\times 10^4$
μ = -7.8549e-06
σ = 9.9361e-04
$\times 10^{-6}$
Range-Rate Residuals [km/s]
Time [sec]
$\times 10^4$
μ = 5.3182e-09
σ = 9.7902e-07

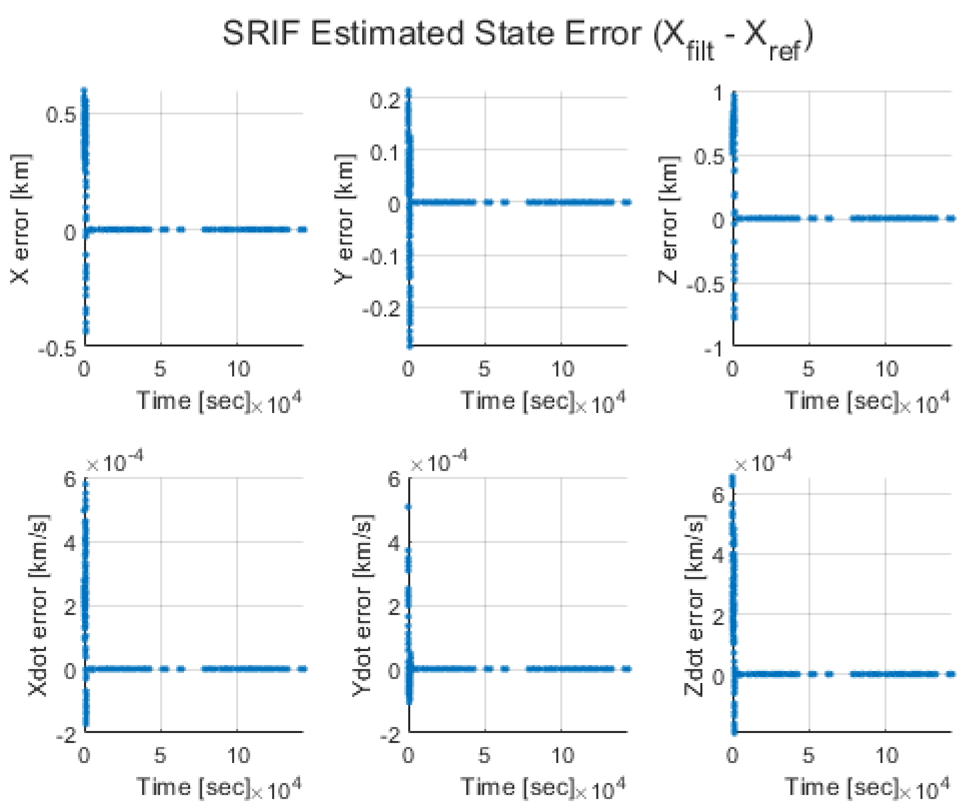

SRIF Post-Fit Residuals - Run 1
Range Residuals [km]
×10^-3
Time [sec]
×10^4
μ = -1.3600e-05
σ = 9.9306e-04
Range-Rate Residuals [km/s]
×10^-6
μ = 3.7772e-09
σ = 9.7948e-07

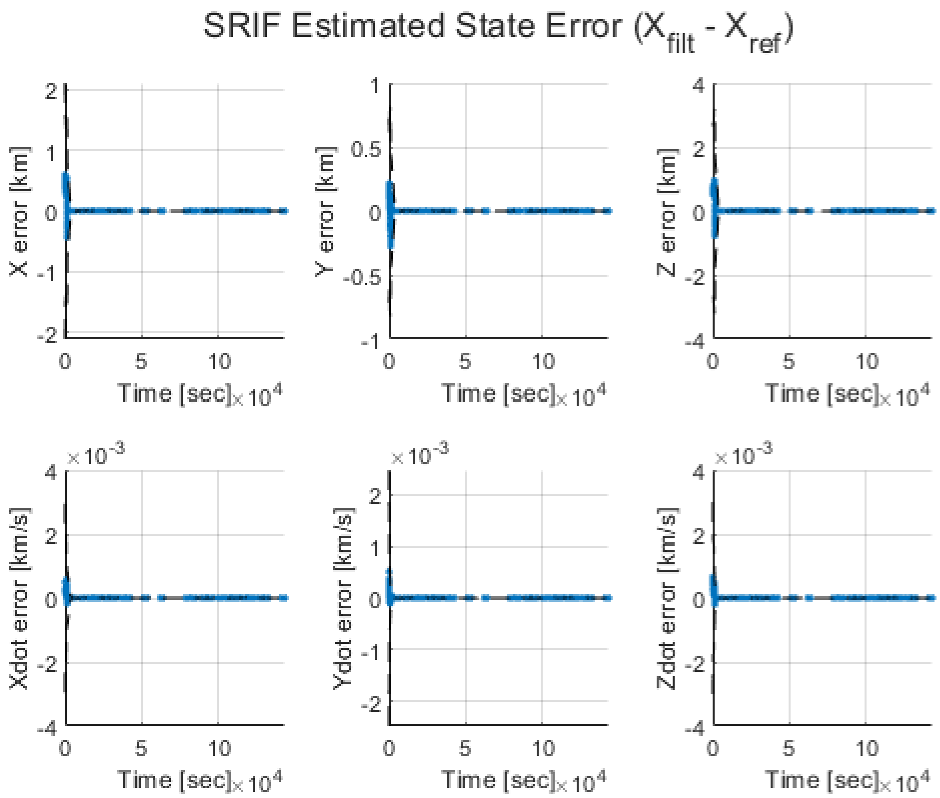

SRIF R Covariance Trace - Run 1
trace(P)
Time [sec]
×10^4


SRIF V Covariance Trace - Run 1
trace(P)
10^-5
10^-10
10^-15
0
5
10
15
Time [sec]
×10^4


Final SRIF Position Covariance Ellipsoid, t = 143370.000 sec
μ = [-1.108e+03, -9.950e+03, -1.478e+02]^T km
σX = 4.648e-05 km, σY = 9.785e-06 km, σZ = 4.305e-05 km
Z [km]
-147.8109
-147.811
-147.8111
X [km]
-1108.3023
-1108.3022
-1108.3021
Y [km]
-9949.9127
-9949.9128
-9949.9129
1σ
2σ
3σ
v1
v2
v3


Final SRIF Velocity Covariance Ellipsoid, t = 143370.000 sec
μ = [4.807e+00, -4.805e-01, -4.051e+00]^T km/s
σ_{Xdot} = 2.519e-08 km/s, σ_{Ydot} = 2.291e-08 km/s, σ_{Zdot} = 2.928e-08 km/s
-4.051048
-4.05104805
-4.0510481
4.80722585
4.8072259
4.80722595
4.807226
Xdot [km/s]
-0.48052635
-0.4805264
-0.48052645
Ydot [km/s]
1σ
2σ
3σ
v_1
v_2
v_3


SRIF Estimated State Error (X_{filt} - X_{ref})
X error [km]
Y error [km]
Z error [km]
Xdot error [km/s]
Ydot error [km/s]
Zdot error [km/s]
Time [sec]
$\times 10^4$
$\times 10^{-4}$


SRIF Estimated State Error (X_{filt} - X_{ref})
X error [km]
Y error [km]
Z error [km]
Xdot error [km/s]
Ydot error [km/s]
Zdot error [km/s]
Time [sec]
$\times 10^{4}$
$\times 10^{-3}$

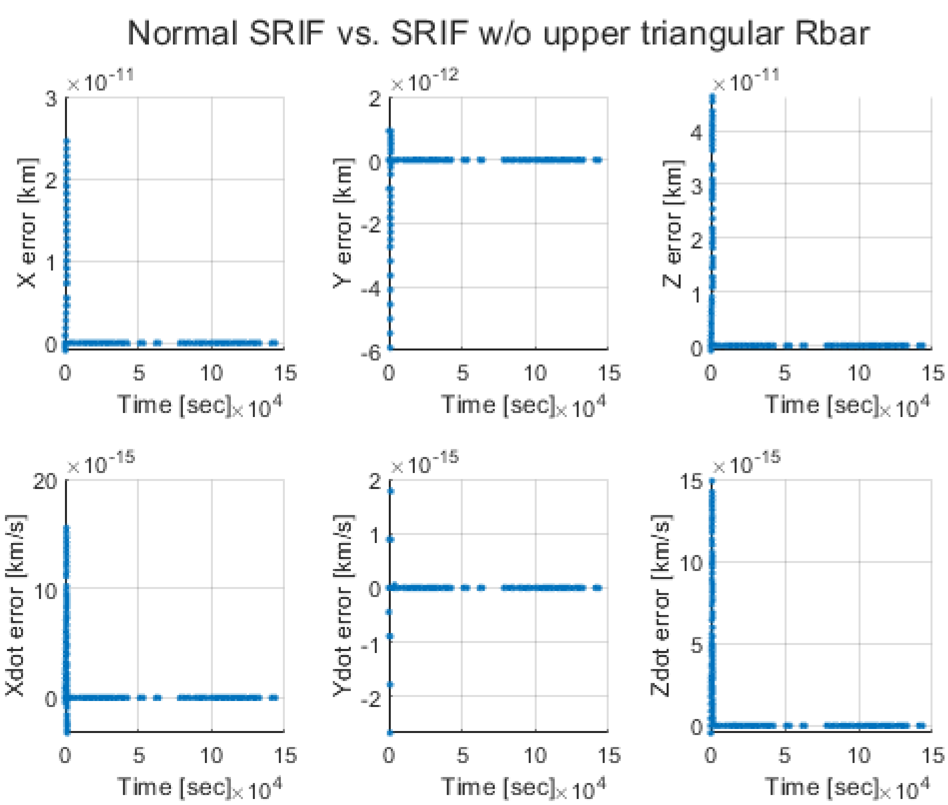

Normal SRIF vs. SRIF w/o upper triangular Rbar
X error [km]
Y error [km]
Z error [km]
Xdot error [km/s]
Ydot error [km/s]
Zdot error [km/s]
Time [sec]
$\times 10^{4}$
$\times 10^{-11}$
$\times 10^{-12}$
$\times 10^{-15}$

Table of Contents


```
function filterOut = SRIF(Xstar0, stations, pConst, P0, x0, Q0, uBar,
forceUpperTriangular)

% Function that implements an SRIF for stat OD problems
%   Inputs:
%       - Xstar0: Initial value of the reference trajectory, organized as
%                 follows:
%                 [X0; Y0; Z0; Xdot0; Ydot0; Zdot0]
%       - stations: Stations struct as defined by makeStations.m. Must have
%                   propagated station states! To propagate states, see
%                   generateTruthData.m.
%       - pConst: Planetary constant structure as formatted by
%                 getPlanetConst.m
%       - P0: Initial state covariance estimate
%       - x0: Initial state deviation estimate
%       - Q0: Initial process noise covariance matrix
%       - uBar: Mean noise vector, generally zeros(3,1) but not always!
%       - forceUpperTriangular: Boolean indicating whether the time updated
%                               R matrix is forced to be upper triangular
%                               or not. Nominally, this should be true.
%   Outputs:
%       - filterOut: Filter output structure with the following fields:
%           - xEst: Estimated state deviation at each time processed from
%                   the station measurements in "stations" (t), organized
%                   as follows:
%                   [xEst_1, xEst_2, ..., xEst_t], where
%                   xEst = [x; y; z; xDot; yDot; zDot]
%           - PEst: Estimated state covariance at each time in t, organized
%                   as follows:
%                   [{P_1}, {P_2}, ..., {P_t}]
%           - PBarEst: Estimated time update state covariance at each time
%                      in t, organized as follows:
%                      [{PBar_1}, {PBar_2}, ..., {PBar_t}];
%           - Ru: Square root noise covariance matrix used for smoothing:
%                 [{Ru_1}, {Ru_2}, ... {Ru_t}]
%           - Rux: Square root state noise covariance matrix for smoothing:
%                  [{Rux_1}, {Rux_2}, ..., {Rux_t}]
%           - bTildeu: Noise information vector for smoothing:
%                      [bTildeu_1, bTildeu_2, ..., bTildeu_t]
%           - uHat: Process noise vector at each time in t until t_tm1:
%                   [uHat_0, uHat_1, ..., uHat_tm1]
```

```
%           - prefit_res_whitened: Whitened pre-fit residuals (y_i) at each
%                                  time in t:
%                                  [y_1, y_2, ..., y_t]
%           - postfit_res_whitened: Whitened postfit residuals at each time
%                                   in t:
%                                   [e_1, e_2, ..., e_t]
%           - prefit_res: Un-whitened pre-fit residuals (V_i*y_i) at each
%                         time in t:
%                         [V_1*y_1, V_2*y_2, ..., V_t*y_t]
%           - postfit_res: Un-whitened postfit residuals at each time in t:
%                          [V_1*e_1, V_2*e_2, ..., V_t*e_t]
%           - t: Measurement time vector for the LKF filter
%           - statVis: Station visibility vector
%           - XStar: Nominal full state at each time in t:
%                    [XStar_1, XStar_2, ..., XStar_t]
%           - XEst: Estimated full state at each time in t, computed as
%                   XEst = XNom + xEst:
%                   [XEst_1, XEst_2, ..., XEst_t], where
%                   XEst = [X; Y; Z; XDot; YDot; ZDot]
%           - Phi_total: Cell array of STMs from t0 to each t_i in t:
%                        [{Phi(t_1, t0)}; {Phi(t_2,t0)}; ...; {Phi(t_f,t_0)}]
%           - Phi: Cell array of STMs from t_im1 to t_i:
%                  [{Phi(t_1, t_0)}; {Phi(t_2,t_1)}; ...; {Phi(t_i,t_im1}]
%
%   By: Ian Faber, 03/07/2025
%
```

Initialize settings

```
    % Default to forcing an upper triangular R
if isempty(forceUpperTriangular)
    forceUpperTriangular = true;
end

    % Format ode45
opt = odeset('RelTol',1e-12,'AbsTol',1e-12);

    % Find state and noise sizes
n = length(Xstar0); % Length of state
q = length(uBar); % Length of noise

    % Preallocate filter outputs
xEst = [];
PEst = [];
PBarEst = [];
Ru = [];
Rux = [];
bTildeu = [];
uHat = [];
prefit_res_whitened = [];
postfit_res_whitened = [];
prefit_res = [];
postfit_res = [];
```

```
XStar = [];
XEst = [];
Phi_total = [];
Phi = [];
```

Define helper function

```
GammaFunc = @(dt) [(dt/2)*eye(3); eye(3)];
```

Process station data into a usable form

```
[t, Y, measCov, Xs, vis] = processStations(stations); % measCov = R in a
normal Kalman Filter!
```

Whiten observations

```
for k = 1:length(Y)
    V = chol(measCov{k}, 'lower'); % measCov = V*V'

    Y{k} = (V^-1)*Y{k};
end
```

Loop through each observation

```
t_im1 = t(1);
Xstar_im1 = Xstar0;
x_im1 = x0;
R_im1 = chol(P0^-1,"upper"); % Find Rbar_0 -> P0^-1 = Lambda = R'*R
if any(Q0 > 0)
    Ru_im1 = chol(Q0^-1,"upper"); % Find Ru_0 -> Q = (Ru'*Ru)^-1
    bu_im1 = Ru_im1*uBar;
end
b_im1 = R_im1*x_im1; % Find bbar_0
Phi_full = eye(n);
for k = 2:length(Y)
        % Read next time, measurement, and measurement covariance
    t_i = t(k);
    Y_i = Y{k};
    V_i = chol(measCov{k},'lower');

        % Continue to integrate Phi(t0, tf) for iteration purposes
    XPhi_full = [Xstar_im1; reshape(Phi_full,n^2,1)];
    [~, XPhi_full] = ode45(@(t,XPhi)STMEOM_J2(t,XPhi,pConst.mu, pConst.J2,
pConst.Ri), [t_im1 t_i], XPhi_full, opt);
    Phi_full = reshape(XPhi_full(end,n+1:end), n, n);

    Phi_total = [Phi_total; {Phi_full}];

        % Integrate Xstar and Phi from t_im1 to t_i
    Phi_im1 = eye(n);
    XPhi_im1 = [Xstar_im1; reshape(Phi_im1,n^2,1)];
```

```
    [~, XPhi_i] = ode45(@(t,XPhi)STMEOM_J2(t,XPhi,pConst.mu, pConst.J2,
pConst.Ri), [t_im1 t_i], XPhi_im1, opt);
    Xstar_i = XPhi_i(end,1:n)';
    Phi_i = reshape(XPhi_i(end,n+1:end),size(Phi_im1)); % Phi(t_i, t_im1)

    Phi = [Phi; {Phi_i}];

        % Time update
    delT = t_i - t_im1;
    if any(any(Q0 > 0) & (delT <= 10)) % Time update with process noise
            % Make Gamma and Q
        Gamma_i = GammaFunc(delT);
        Q_i = Q0;

            % Set up noise variables
        Ru_k = chol(Q_i^-1,"upper"); % Q = (Ru'*Ru)^-1

            % Set up Rtilde
        Rtilde = R_im1*(Phi_i^-1);

        mat = [
                Ru_k, zeros(q,n), bu_im1;
                -Rtilde*Gamma_i, Rtilde, b_im1
              ];

            % Run Householder
        out = Householder(mat);

            % Extract time update outputs
        R_i = out(q+1:end, q+1:q+n);
        b_i = out(q+1:end, q+n+1);

            % Extract process noise outputs
        Ru_k = out(1:q,1:q);
        Rux_k = out(1:q,q+1:q+n);
        bTildeu_k = out(1:q,q+n+1);

            % Propagate noise information vector
        bu_im1 = Ru_k*uBar;

            % Accumulate process noise outputs
        Ru = [Ru, {Ru_k}];
        Rux = [Rux, {Rux_k}];
        bTildeu = [bTildeu, bTildeu_k];

    else  % Time update without process noise
        x_i = Phi_i*x_im1;
        R_i = R_im1*(Phi_i^-1);
        % b_i = R_i*x_i; % This should be constant across a time update! b_i
= R_i*x_i = R_im1*(Phi_i^-1)*Phi_i*x_im1 = R_im1*x_im1 = b_im1
        b_i = b_im1;

            % Force R_i to be upper triangular with Householder
Transformation
```

```
        if forceUpperTriangular
            out = Householder([R_i,b_i]);
            R_i = out(1:n,1:n);
            b_i = out(1:n,n+1);
        end
    end

        % Reconstruct PBar
    PBarEst = [PBarEst, {(R_i'*R_i)^-1}]; % P = (Lambda)^-1 = (R'*R)^-1

        % Get number of measurements in Y, station states, and station
        % visibility at this time
    meas = length(Y_i)/2; % Assuming 2 data points per measurement: range
and range-rate
    Xstat = Xs{k}'; % Extract station state(s) at the time of measurement
    statVis = vis{k}; % Extract the stations that were visible at the time
of measurement

        % Build y_i
    yExp = [];
    for kk = 1:meas
        genMeas = generateRngRngRate(Xstar_i, Xstat(:,meas),
stations(statVis(kk)).elMask, true); % Ignore elevation mask
        yExp = [yExp; (V_i^-1)*genMeas(1:2)]; % Whiten the expected
measurement
    end

    y_i = Y_i - yExp;

        % Build Htilde_i
    Htilde_i = [];
    for kk = 1:meas
        Htilde = MeasurementPartials_RngRngRate_sc(Xstar_i, Xstat(:,meas));
        Htilde_i = [Htilde_i; (V_i^-1)*Htilde]; % Whiten the Htilde matrix
    end

        % Measurement update
    mat = [R_i, b_i; Htilde_i, y_i];
    out = Householder(mat);

    R_i = out(1:n,1:n);
    b_i = out(1:n,n+1);
    e = out(n+1:end,n+1);

        % Intermediate variable for xHat
    xHat = (R_i^-1)*b_i;

    if any(any(Q0>0) & (delT <= 10))
            % Calculate uHat_im1 and accumulate it
        u_im1 = (Ru_k^-1)*(bTildeu_k - Rux_k*xHat);
        uHat = [uHat, u_im1];
    end

        % Accumulate data to save
```

```
    xEst = [xEst, xHat]; % xHat = R^-1*b
    PEst = [PEst, {(R_i'*R_i)^-1}]; % P = (Lambda)^-1 = (R'*R)^-1
    prefit_res_whitened = [prefit_res_whitened, y_i];
    postfit_res_whitened = [postfit_res_whitened, e];
    prefit_res = [prefit_res, V_i*y_i];
    postfit_res = [postfit_res, V_i*e];
    XStar = [XStar, Xstar_i];
    XEst = [XEst, Xstar_i + xHat];

        % Update for next run
    t_im1 = t_i;
    Xstar_im1 = Xstar_i;
    x_im1 = xHat;
    R_im1 = R_i;
    b_im1 = b_i;

end
```

Assign outputs

```
filterOut.xEst = xEst;
filterOut.PEst = PEst;
filterOut.PBarEst = PBarEst;
filterOut.Ru = Ru;
filterOut.Rux = Rux;
filterOut.bTildeu = bTildeu;
filterOut.uHat = uHat;
filterOut.prefit_res_whitened = prefit_res_whitened;
filterOut.postfit_res_whitened = postfit_res_whitened;
filterOut.prefit_res = prefit_res;
filterOut.postfit_res = postfit_res;
filterOut.t = t(2:end); % t_0 not included in estimate
filterOut.statVis = vis;
filterOut.XStar = XStar;
filterOut.XEst = XEst;
filterOut.Phi_total = Phi_total;
filterOut.Phi = Phi;

end
```

Published with MATLAB® R2023b

2. a. Implement the SRIF with process noise.

see PDF for code

b. Process the data from HW 3 and compare to the LKF with SNC.

The SRIF with process noise and LKF with SNC give very similar results, with state estimates agreeing to within 20 m in position and 20 cm/s in velocity.

I was expecting these to match exactly, but I suspect there's a small numerical bug in my code that's causing the mismatch.

Further, my state errors don't lie solely within the 3σ bounds, which could just be due to the presence of the unmodeled J3 dynamics and suggests that the σ I chose isn't optimal.

However, the two filters are fairly close and work great when using the right dynamics.

ASEN 6080 HW 5 Problem 2 Main Script

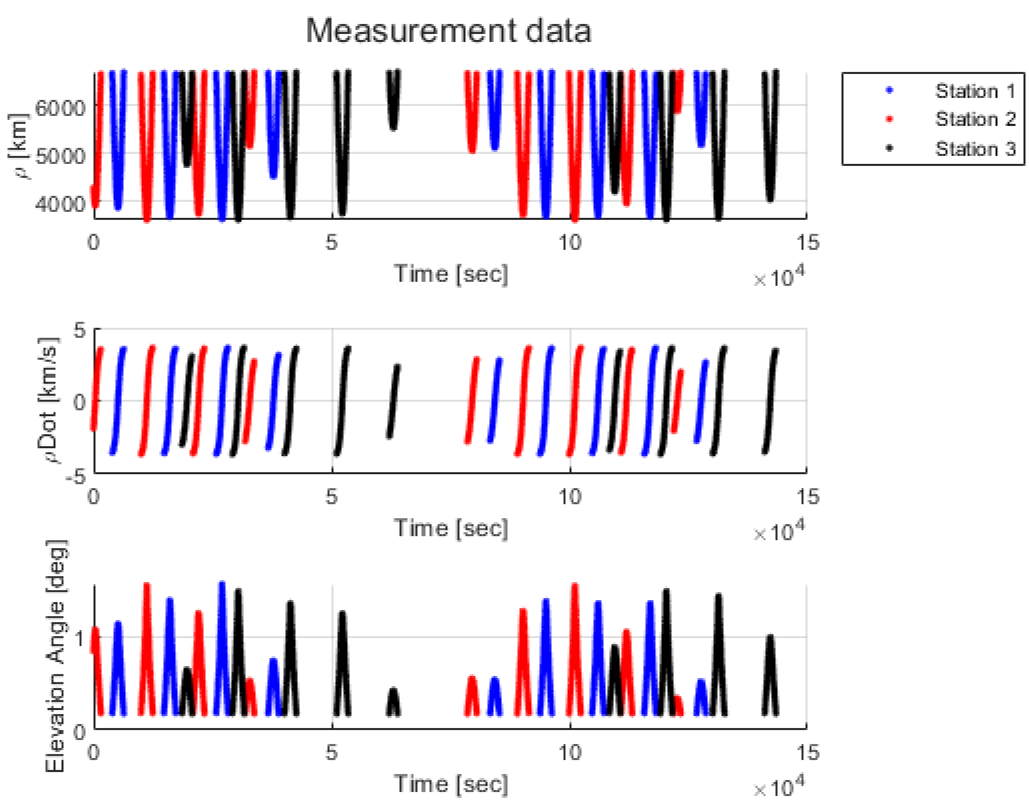
Table of Contents

By: Ian Faber

Housekeeping

Setup

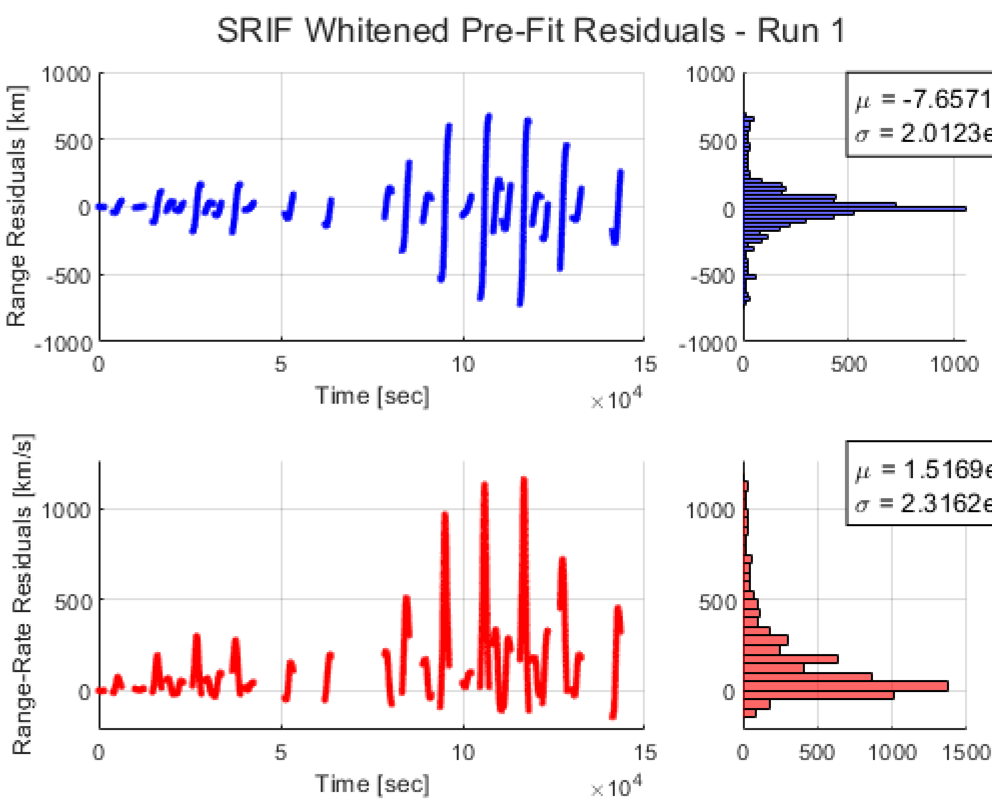
Make truth data

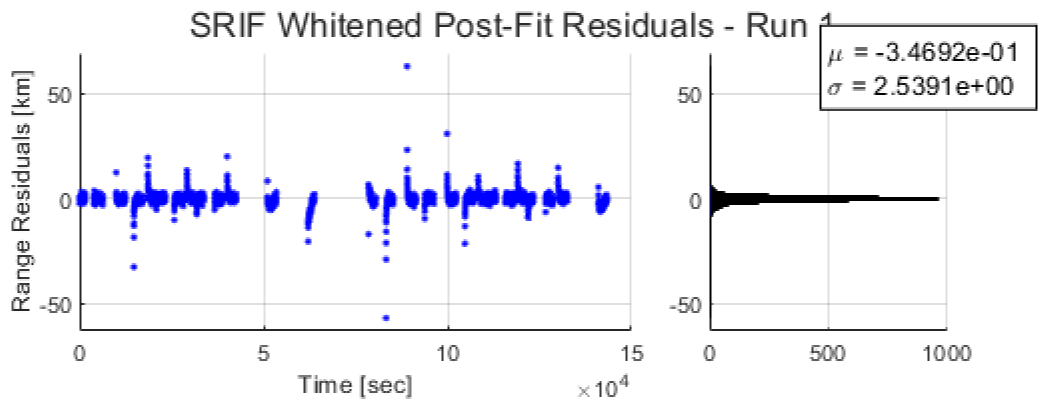


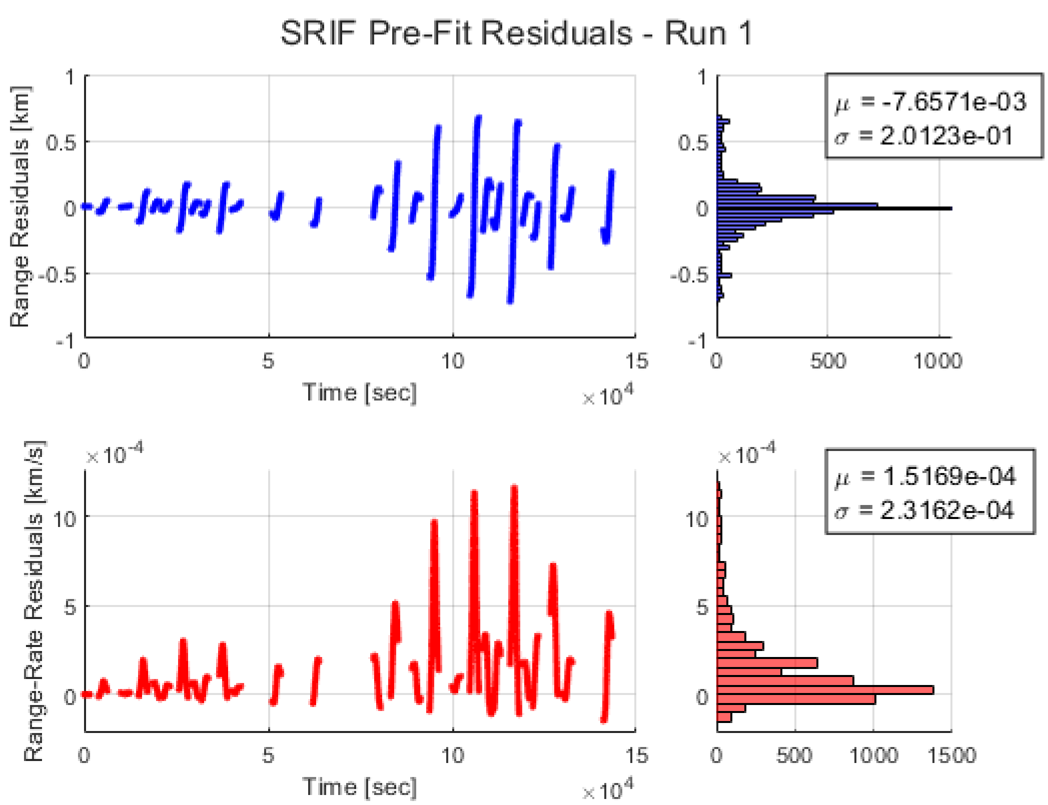
Problem 2a. Filter setup

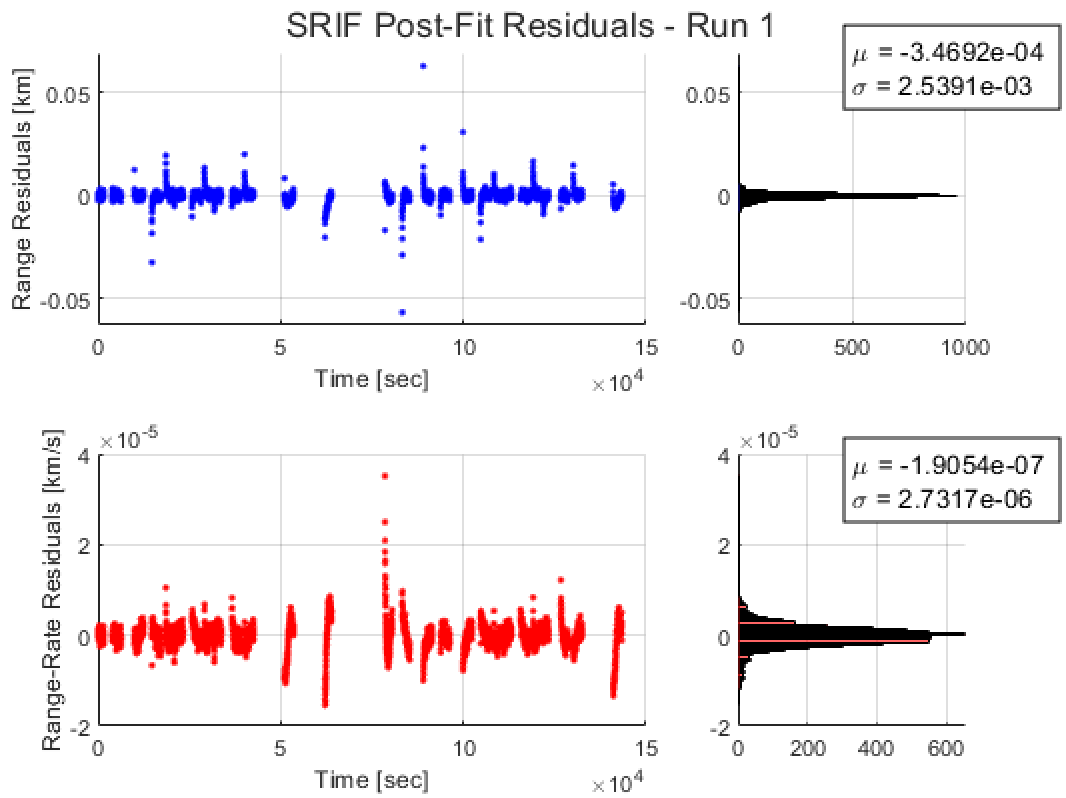
Problem 2b. Compare SRIF to LKF with process noise

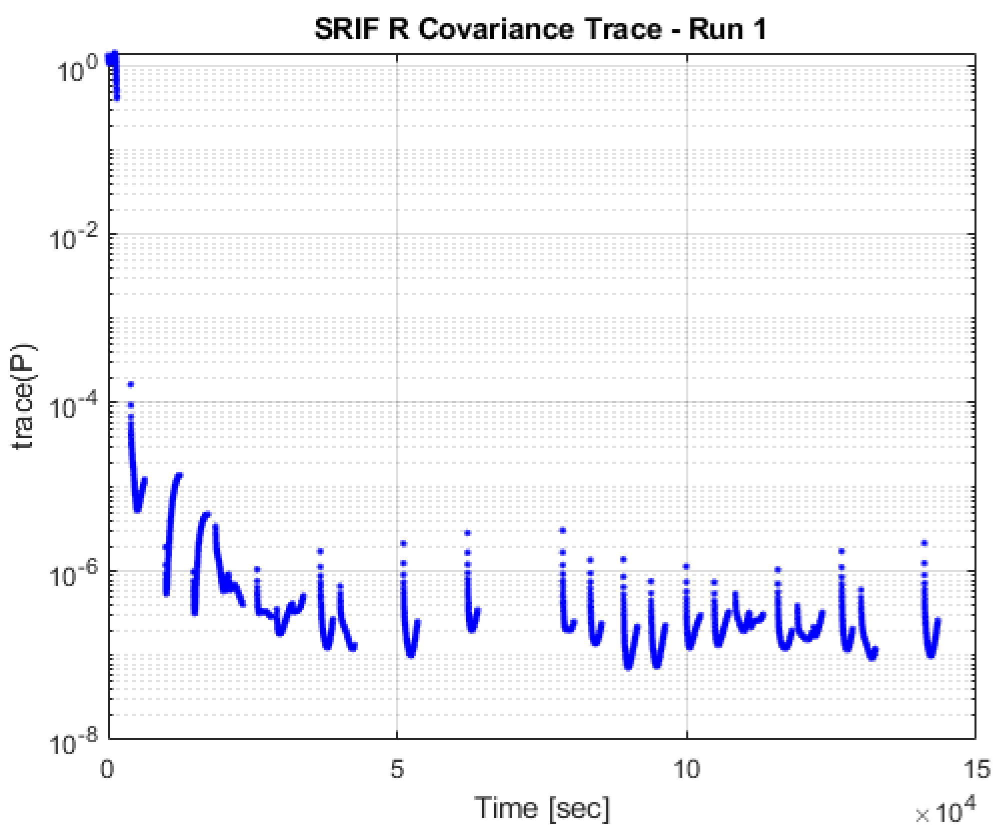
```
2b. Comparing SRIF to LKF

    Running SRIF:
Prefit RMS: 242.0656, Postfit RMS: 2.6517. Hit max SRIF iterations. Runs so
far: 1
Final whitened prefit RMS: 242.0656. Hit maximum number of 1 runs
Final whitened postfit RMS: 2.6517. Hit maximum number of 1 runs
Final prefit RMS: 242.0656. Hit maximum number of 1 runs
Final postfit RMS: 2.6517. Hit maximum number of 1 runs

    Running LKF:
Prefit RMS: 242.0656, Postfit RMS: 0.9973. Hit max LKF iterations. Runs so
far: 1
Final prefit RMS: 242.0656. Hit maximum number of 1 runs
Final postfit RMS: 0.9973. Hit maximum number of 1 runs
```

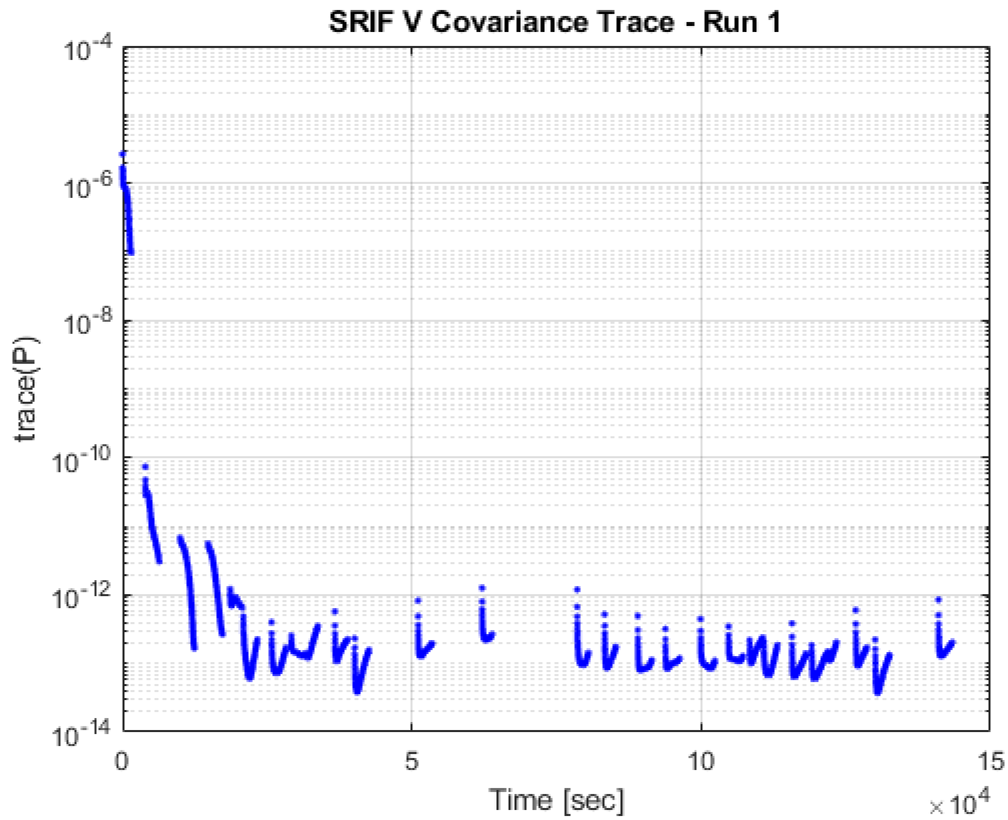


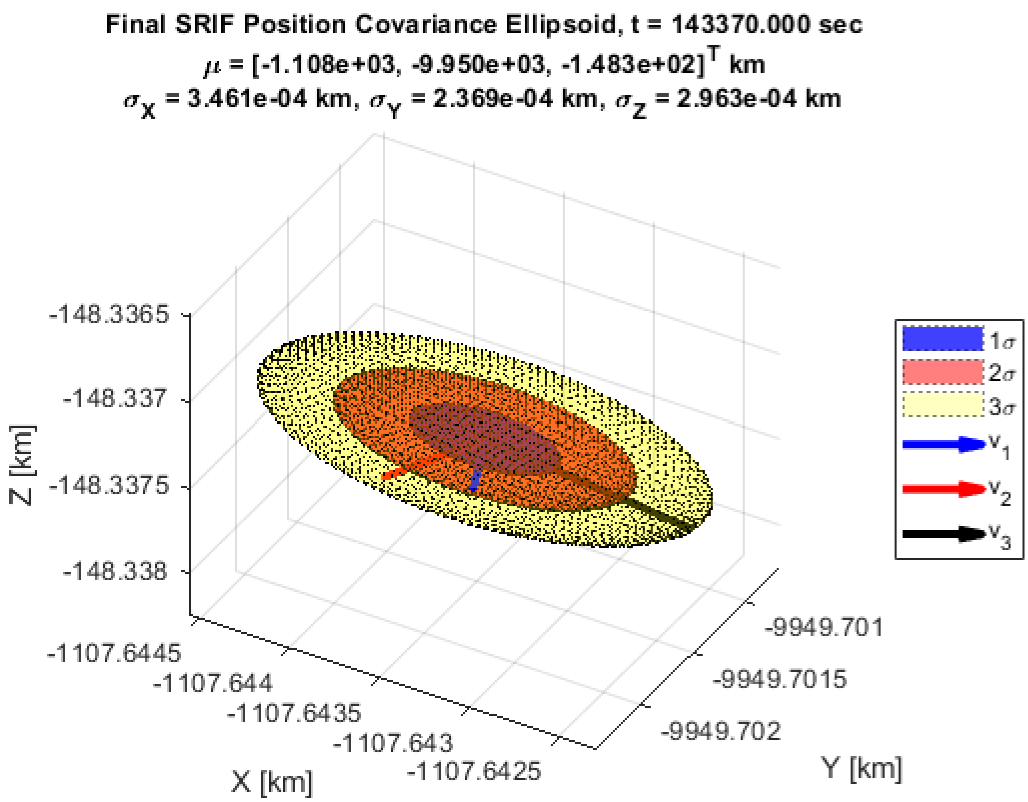

SRIF Whitened Post-Fit Residuals - Run 1
μ = -3.4692e-01
σ = 2.5391e+00
Range Residuals [km]
Time [sec]
×10^4
μ = -1.9054e-01
σ = 2.7317e+00
Range-Rate Residuals [km/s]
Time [sec]
×10^4

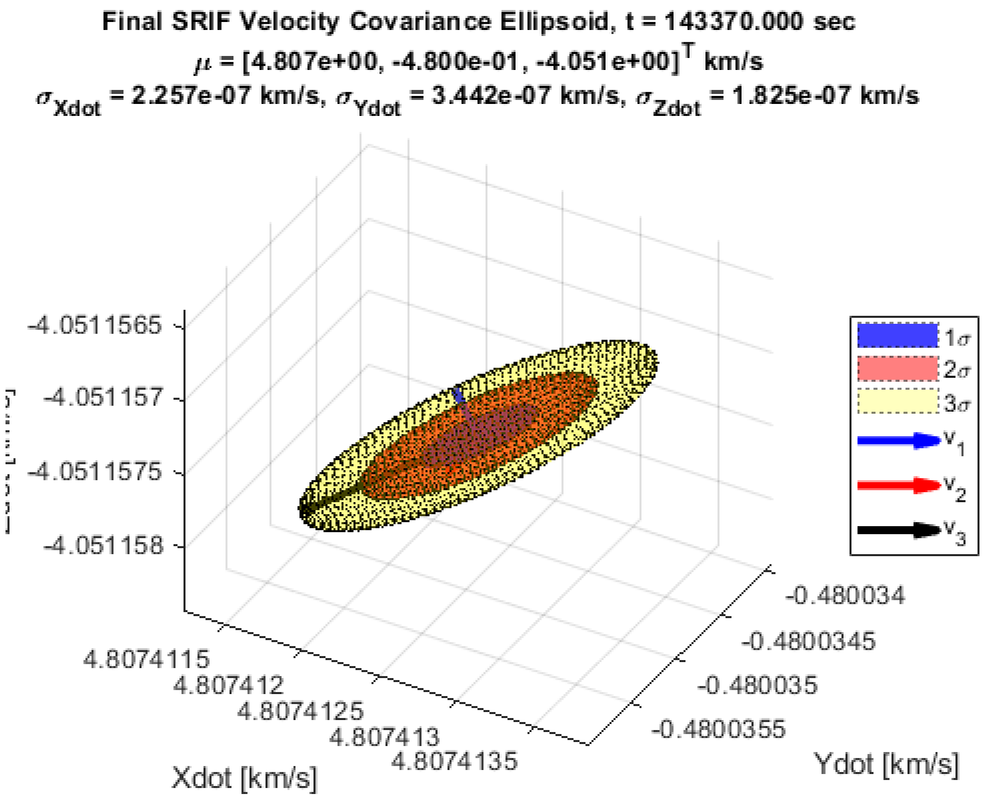

SRIF Pre-Fit Residuals - Run 1
μ = -7.6571e-03
σ = 2.0123e-01
Range Residuals [km]
Time [sec]
×10^4
×10^-4
μ = 1.5169e-04
σ = 2.3162e-04
Range-Rate Residuals [km/s]
Time [sec]
×10^4

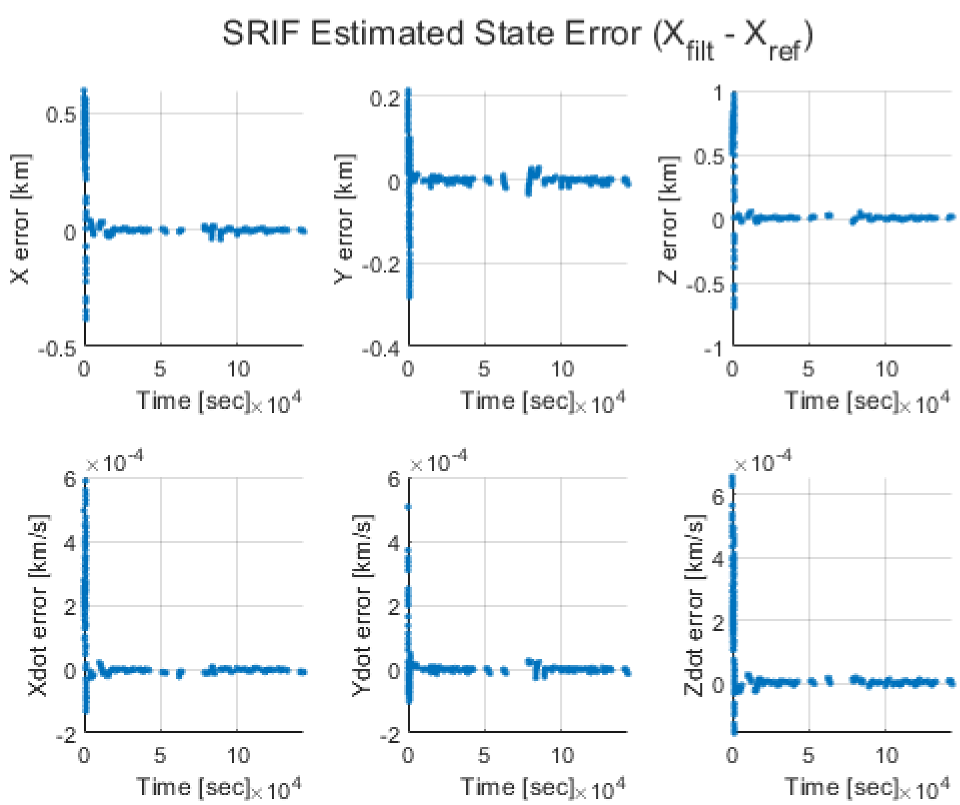

SRIF Post-Fit Residuals - Run 1
μ = -3.4692e-04
σ = 2.5391e-03
Range Residuals [km]
0.05
0
-0.05
0
5
10
15
Time [sec]
×10^4
0
500
1000
Range-Rate Residuals [km/s]
×10^-5
4
2
0
-2
μ = -1.9054e-07
σ = 2.7317e-06
200
400
600

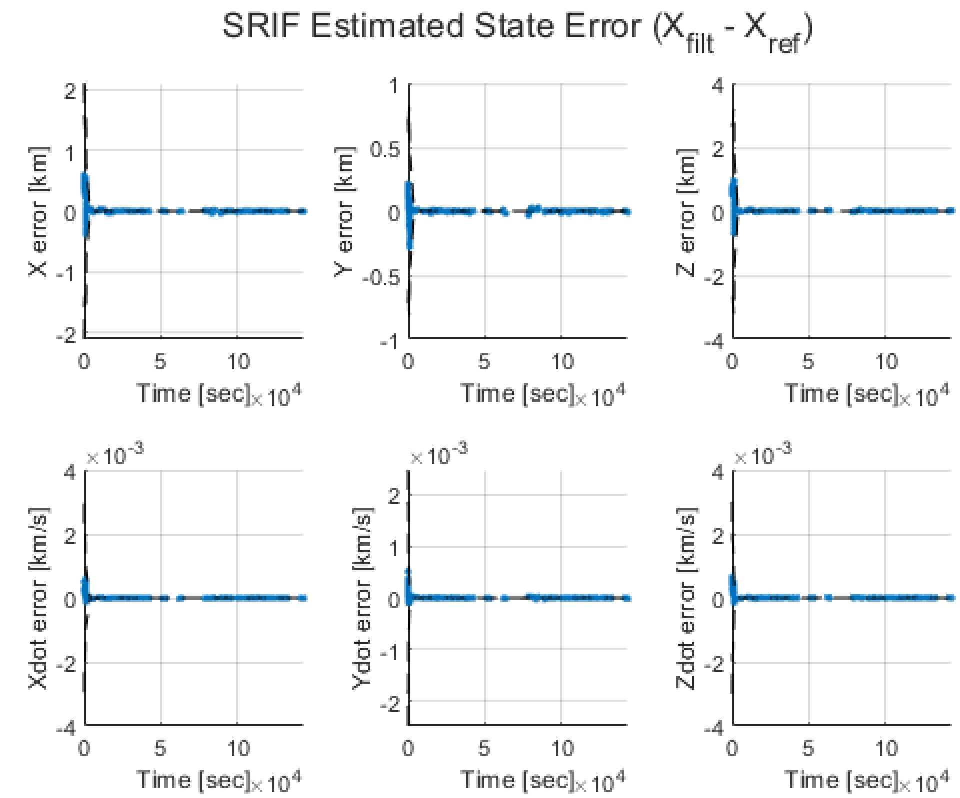

SRIF R Covariance Trace - Run 1
trace(P)
10^0
10^-2
10^-4
10^-6
10^-8
0
5
10
15
Time [sec]
×10^4

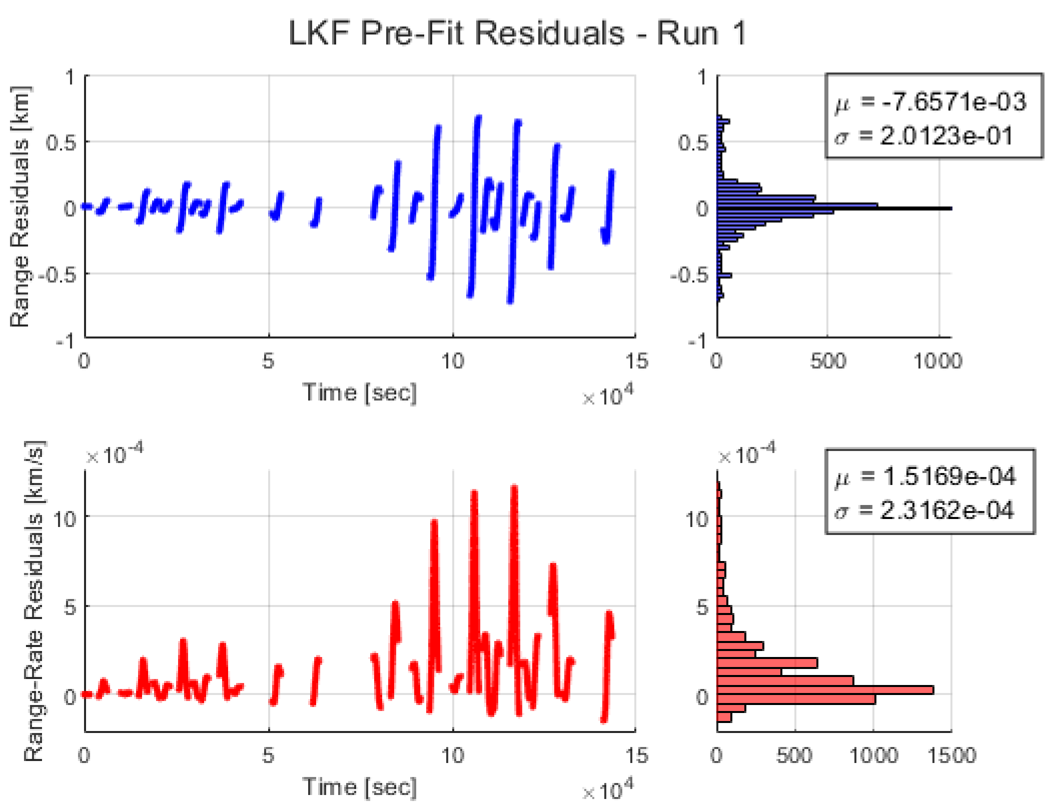

SRIF V Covariance Trace - Run 1
10^-4
10^-6
10^-8
10^-10
10^-12
10^-14
trace(P)
0
5
10
15
Time [sec]
×10^4

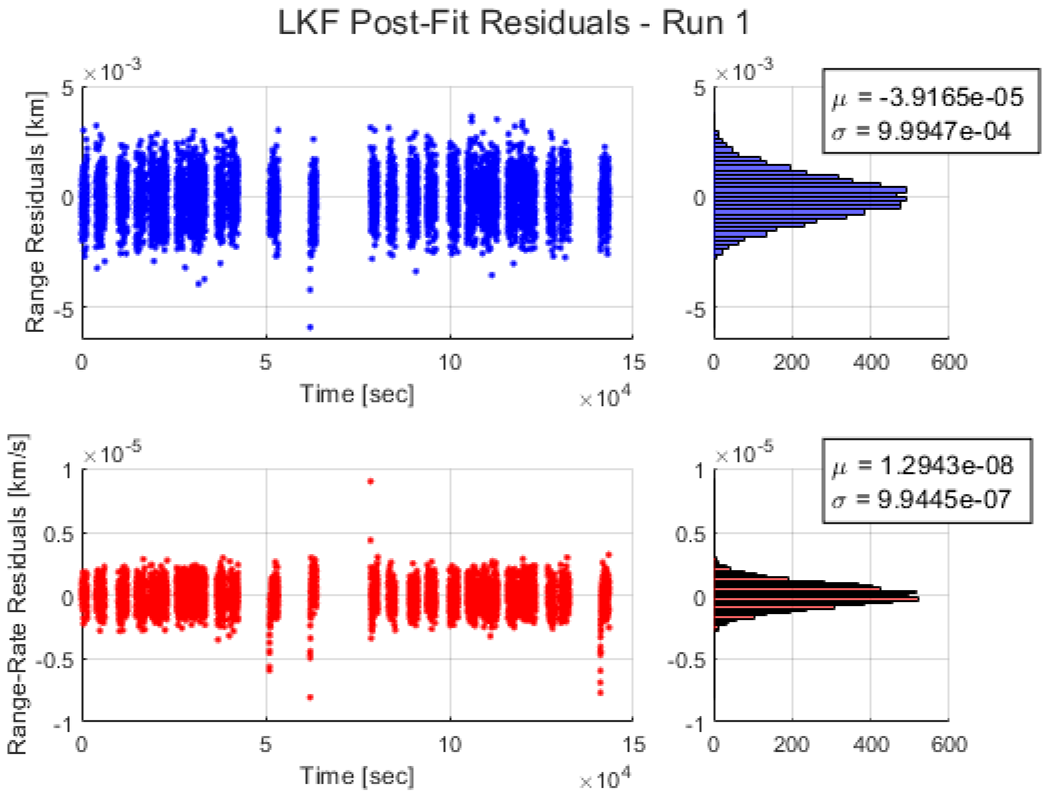

Final SRIF Position Covariance Ellipsoid, t = 143370.000 sec
μ = [-1.108e+03, -9.950e+03, -1.483e+02]^T km
σX = 3.461e-04 km, σY = 2.369e-04 km, σZ = 2.963e-04 km
-148.3365
-148.337
-148.3375
-148.338
Z [km]
-1107.6445
-1107.644
-1107.6435
-1107.643
-1107.6425
X [km]
-9949.701
-9949.7015
-9949.702
Y [km]
1σ
2σ
3σ
v1
v2
v3

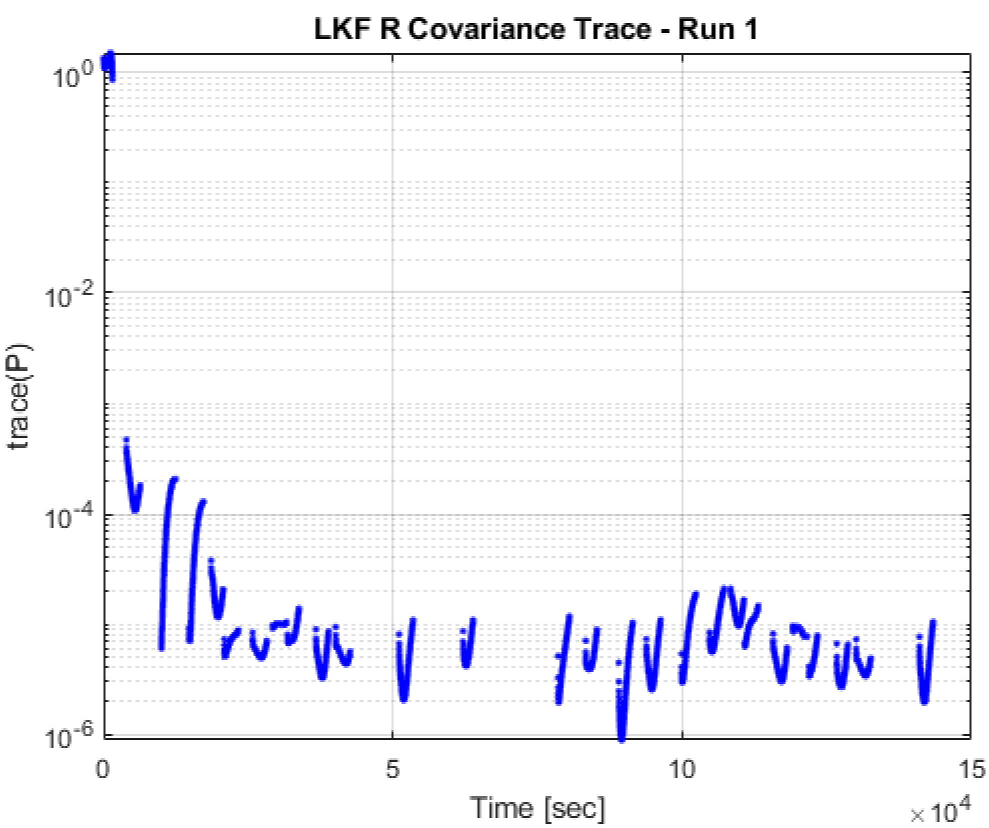

Final SRIF Velocity Covariance Ellipsoid, t = 143370.000 sec
μ = [4.807e+00, -4.800e-01, -4.051e+00]^T km/s
σXdot = 2.257e-07 km/s, σYdot = 3.442e-07 km/s, σZdot = 1.825e-07 km/s
-4.0511565
-4.051157
-4.0511575
-4.051158
4.8074115
4.807412
4.8074125
4.807413
4.8074135
Xdot [km/s]
-0.480034
-0.4800345
-0.480035
-0.4800355
Ydot [km/s]
1σ
2σ
3σ
v1
v2
v3

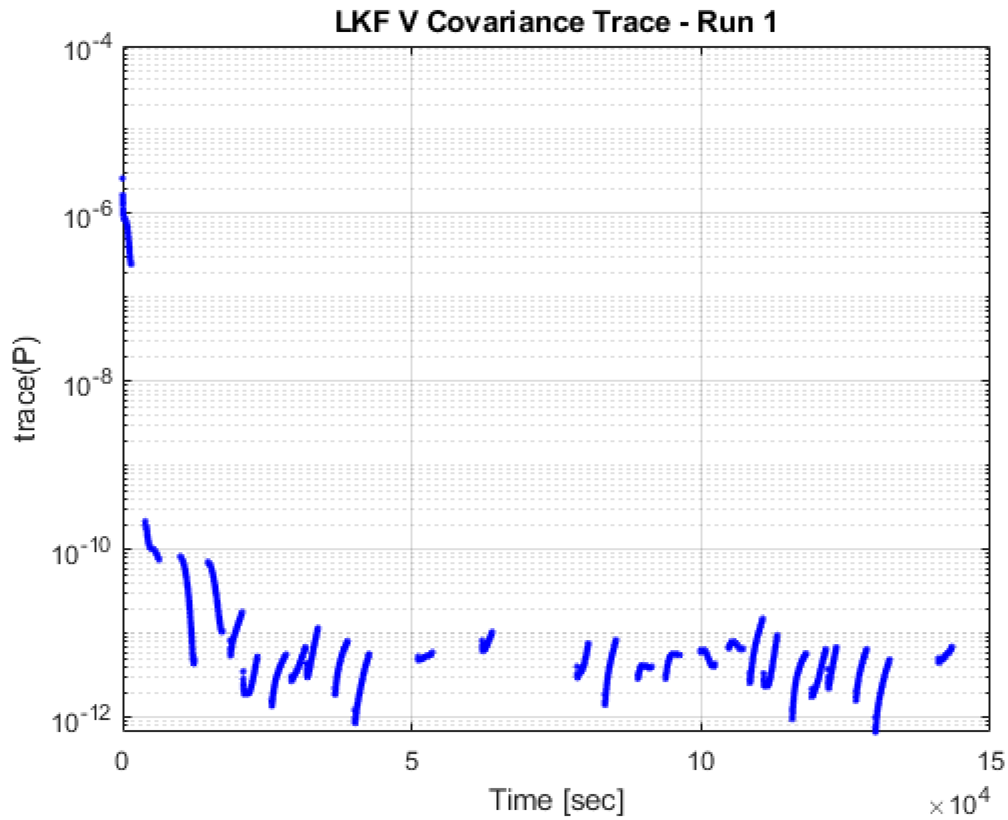

SRIF Estimated State Error (Xfilt - Xref)
X error [km]
Y error [km]
Z error [km]
Xdot error [km/s]
Ydot error [km/s]
Zdot error [km/s]
Time [sec]
×10^4
×10^-4

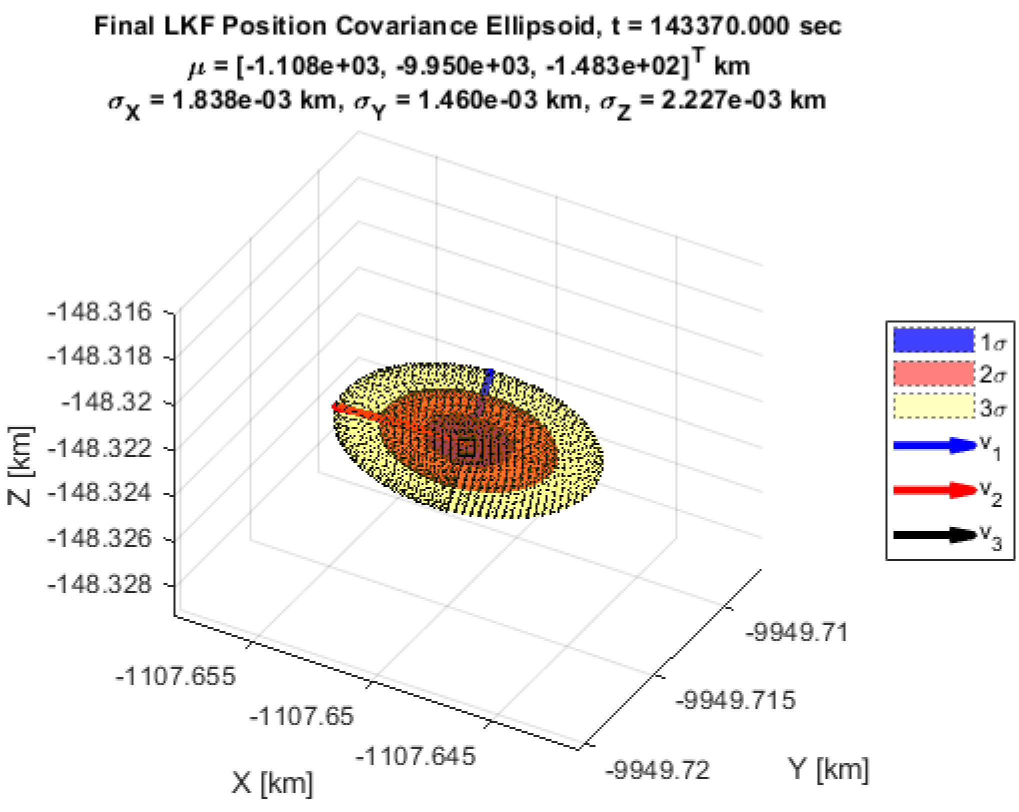

SRIF Estimated State Error (X_{filt} - X_{ref})
X error [km]
Y error [km]
Z error [km]
Xdot error [km/s]
Ydot error [km/s]
Zdot error [km/s]
Time [sec]
×10^4
×10^-3

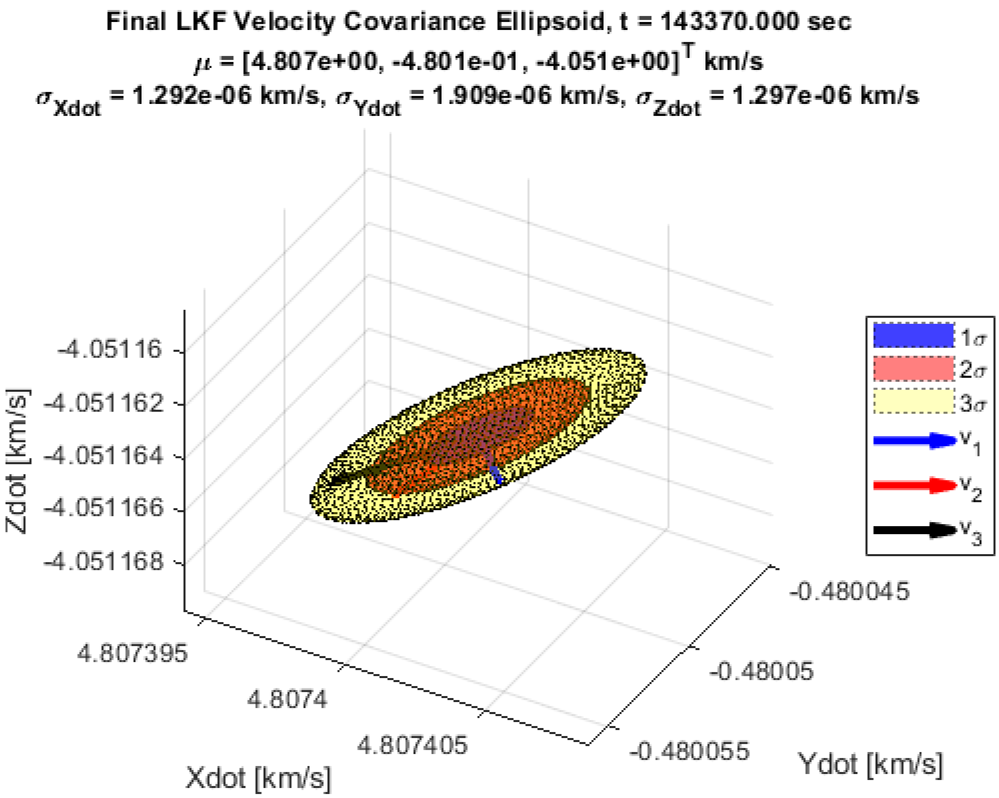

LKF Pre-Fit Residuals - Run 1
Range Residuals [km]
Range-Rate Residuals [km/s]
Time [sec]
×10^4
×10^-4
μ = -7.6571e-03
σ = 2.0123e-01
μ = 1.5169e-04
σ = 2.3162e-04

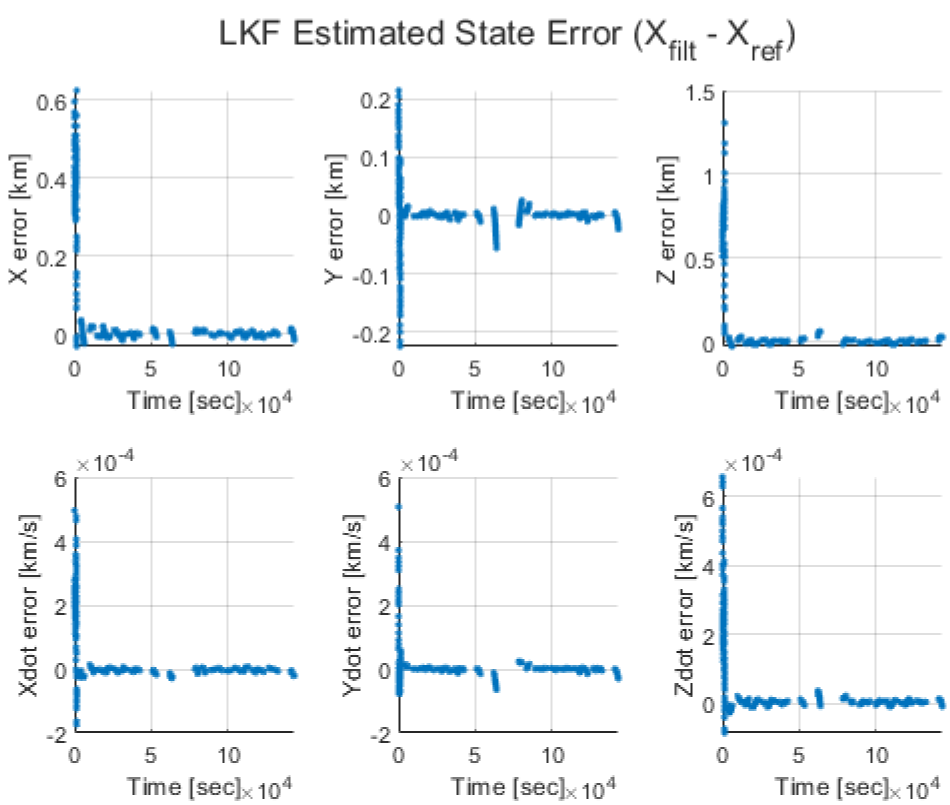

LKF Post-Fit Residuals - Run 1
Range Residuals [km]
×10^-3
Time [sec]
×10^4
μ = -3.9165e-05
σ = 9.9947e-04
Range-Rate Residuals [km/s]
×10^-5
μ = 1.2943e-08
σ = 9.9445e-07

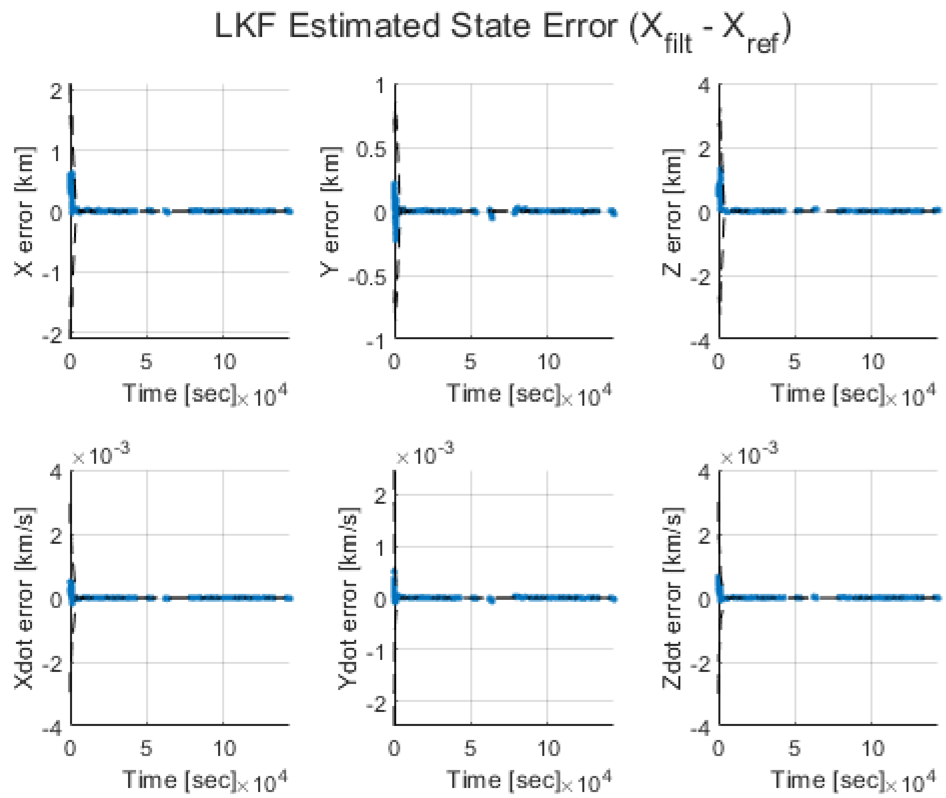

LKF R Covariance Trace - Run 1
trace(P)
Time [sec]
×10^4

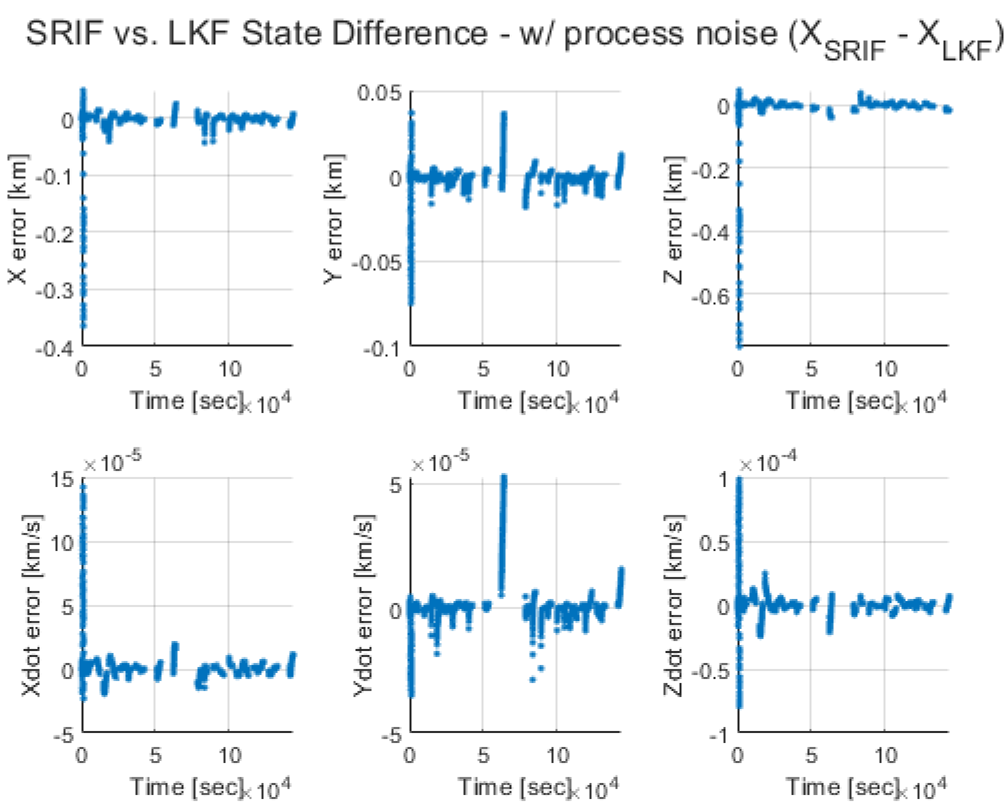

LKF V Covariance Trace - Run 1
trace(P)
Time [sec]
$\times 10^4$
0
5
10
15
10^{-4}
10^{-6}
10^{-8}
10^{-10}
10^{-12}


Final LKF Position Covariance Ellipsoid, t = 143370.000 sec
μ = [-1.108e+03, -9.950e+03, -1.483e+02]^T km
σ_X = 1.838e-03 km, σ_Y = 1.460e-03 km, σ_Z = 2.227e-03 km
Z [km]
X [km]
Y [km]
-148.316
-148.318
-148.32
-148.322
-148.324
-148.326
-148.328
-1107.655
-1107.65
-1107.645
-9949.71
-9949.715
-9949.72
1σ
2σ
3σ
v_1
v_2
v_3


Final LKF Velocity Covariance Ellipsoid, t = 143370.000 sec
μ = [4.807e+00, -4.801e-01, -4.051e+00]^T km/s
σXdot = 1.292e-06 km/s, σYdot = 1.909e-06 km/s, σZdot = 1.297e-06 km/s
Zdot [km/s]
-4.05116
-4.051162
-4.051164
-4.051166
-4.051168
4.807395
4.8074
4.807405
Xdot [km/s]
-0.480045
-0.48005
-0.480055
Ydot [km/s]
1σ
2σ
3σ
v1
v2
v3


LKF Estimated State Error (Xfilt - Xref)
X error [km]
Y error [km]
Z error [km]
Xdot error [km/s]
Ydot error [km/s]
Zdot error [km/s]
Time [sec]
×10^4
×10^-4


LKF Estimated State Error (X_filt - X_ref)
X error [km]
Y error [km]
Z error [km]
Xdot error [km/s]
Ydot error [km/s]
Zdot error [km/s]
×10^-3
Time [sec] ×10^4


SRIF vs. LKF State Difference - w/ process noise (X_SRIF - X_LKF)
X error [km]
Y error [km]
Z error [km]
Xdot error [km/s]
Ydot error [km/s]
Zdot error [km/s]
×10^-5
×10^-4
Time [sec] ×10^4