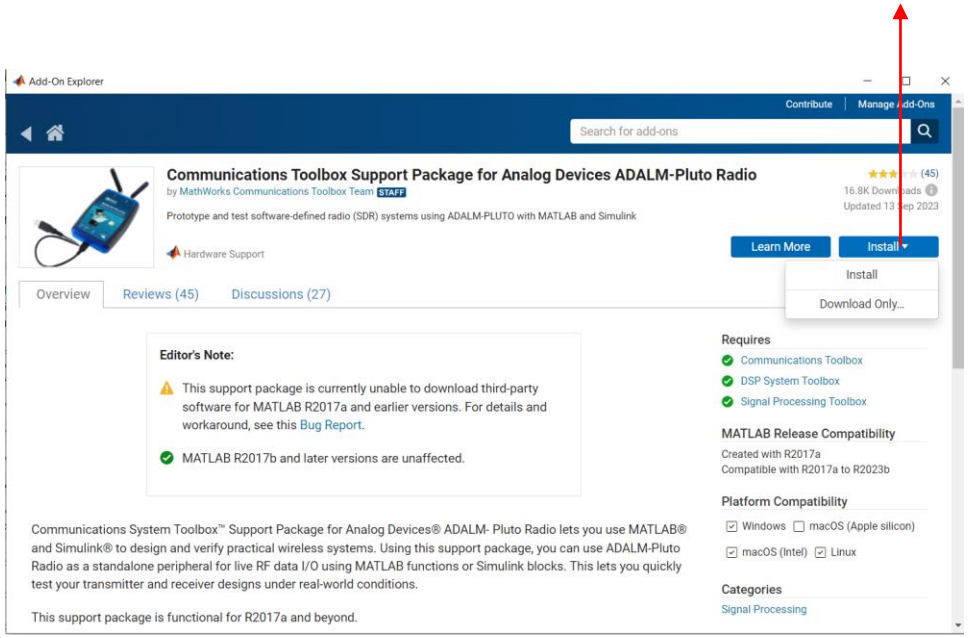
PlutoSDR and MATLAB Example

December 1, 2023

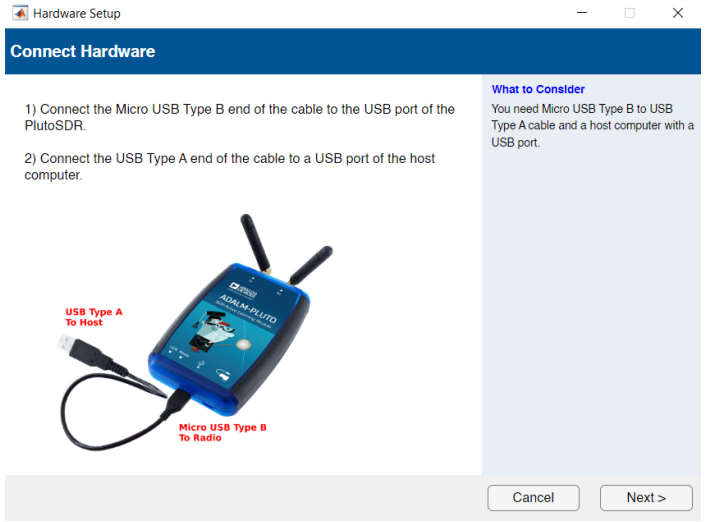
Ellarizza Fredeluces
Tokyo University of Marine Science and Technology

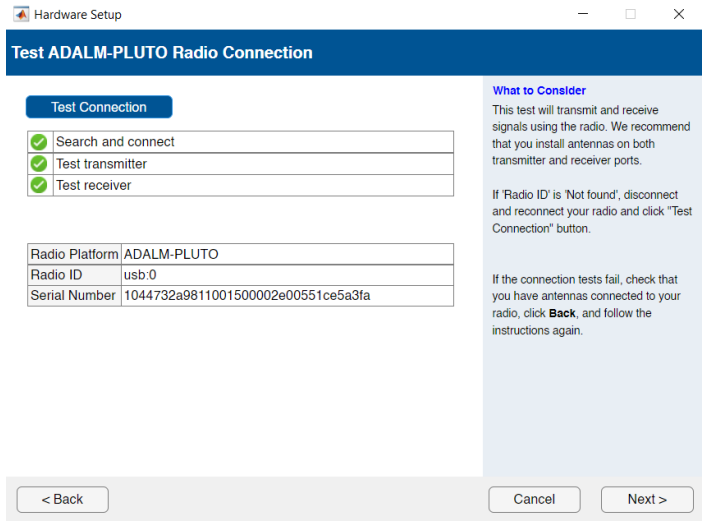
PlutoSDR installation in MATLAB

- Download MATLAB Add-on for PlutoSDR



- At some point during installation, you need to plug PlutoSDR to your PC. Click Test Connection and make sure that Radio ID is found.



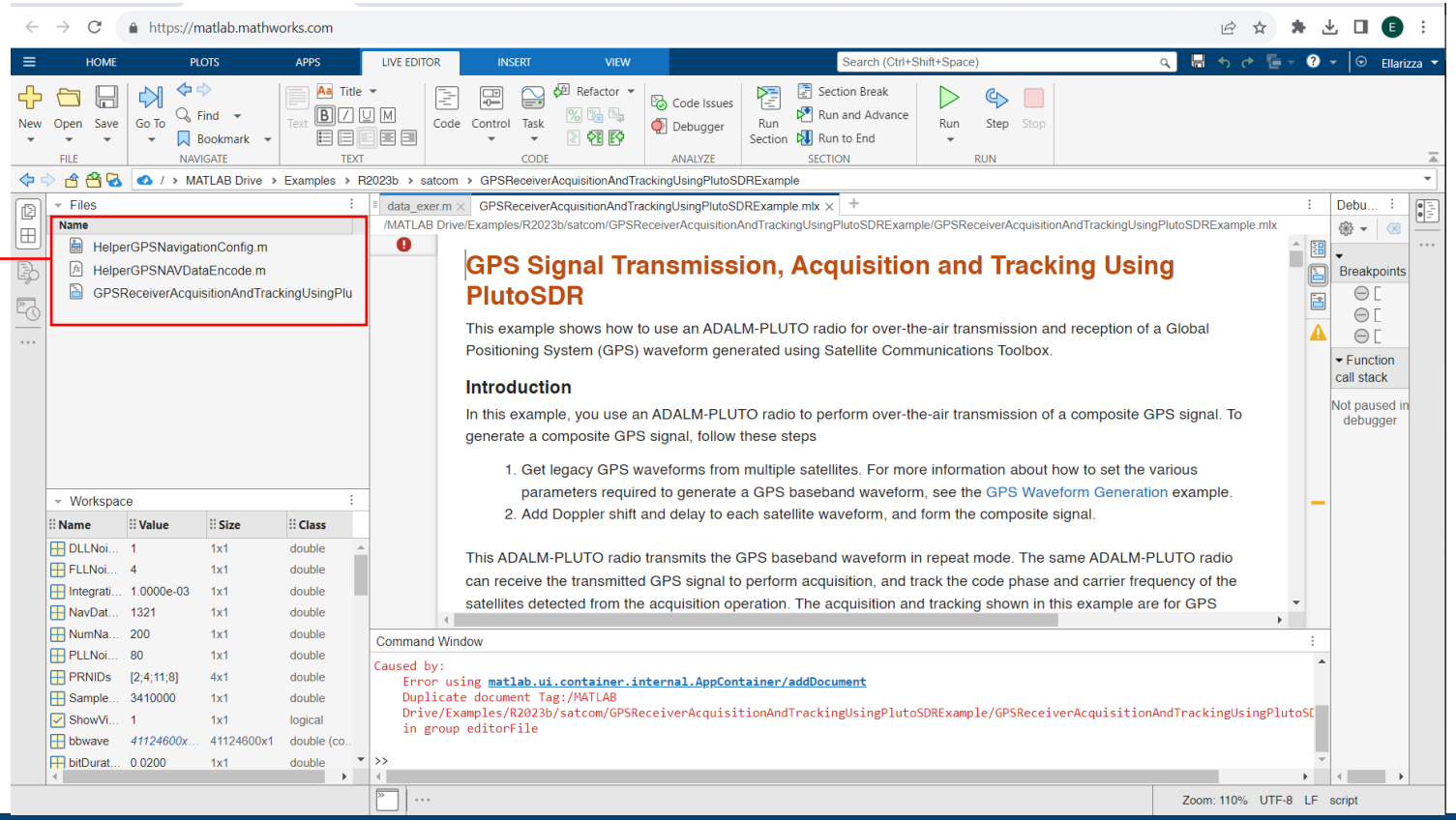


MATLAB Example

- You can make a copy of the MATLAB example by yourself or simply download it. I downloaded a copy and necessary classes.
- Needs new MATLAB version (2023b) for signal acquisition and tracking objects.

https://jp.mathworks.com/help/satcom/ug/gps-receiver-acquisition-and-tracking-using-pluto-sdr.html

Download these.



Contents of MATLAB Example

- Configure Simulation Parameters
- Generate GPS Waveform
- Configure PlutoSDR
- Acquisition and Tracking

Configure Simulation Parameters

```
%----------------------------------------
% CONFIGURE SIMULATION PARAMETERS

ShowVisualizations = true;
% Transmitter Configuration
% Specify the number of GPS satellites in the waveform
numSat = 4;

% Specify PRN IDs
PRNIDs = [2; 4; 11; 8];

% Set this value to control the number of navigation data bits in the -
% generated waveform
NumNavDataBits = 200;

%Set bit starting index
NavdataBitStartIndex = 1321;

%Sample Rate
SampleRate = 3.41e6 % samples/s

% Signal delay
sigdelay = [300.34; 587.21; 425.89; 312.88]; % number of C/A code chips delay

% Initialize peak Doppler shift and rate for each satellite
% This example can track Doppler shift from -10kHz to +10kHz
peakDoppler = [3589; 4256; 8596; 9568];
dopplerRate = [1000; 500; 700; 500];

%----------------------------------------
% RECEIVER CONFIGURATION

PLLNoiseBandwidth = 80; % Hz
FLLNoiseBandwidth = 4; % Hz
DLLNoiseBandwidth = 1; % Hz

% PLL integration time
IntegrationTime = 1e-3; % seconds
```

For Generating IF data later

For Acquisition and Tracking later

Generating GPS Waveform

```
%----------------------------------------
% GENERATE GPS WAVEFORM
% transmitting frequency will be at 2.41 GHz
% to avoid interference with real signals

numBitsForDelay = 1;

% Initialize output waveform
resultsig = 0;

% Generate waveform from each satellite
for isat = 1:numSat
    % Create the legacy navigation (LNAV) configuration object
    lnavConfig = HelperGPSNavigationConfig("SignalType","LNAV","PRNID",PRNIDs(isat));

    % Generate the navigation data bits from the configuration object
    lnavData = HelperGPSNAVDataEncode(lnavConfig);

    % Configure the GPS waveform generation properties
    t = lnavConfig.HOWTOW*6; % First get the initial time
    % HOWTOW is an indication of the next subframe starting point. Because
    % each subframe is 300 bits long, you must subtract 300 bits from the
    % initial value to get the starting value for the first subframe. This
    % value can be negative as well. Because bit is of a 20 millisecond
    % duration, to get the time elapsed for bits, you must multiply the bit
    % index by 20e-3;
    bitDuration = 20e-3; % seconds
    pCodeRate = 10.23e6; % Hz
    numPChipsPerNavBit = bitDuration*pCodeRate;
    navdatalen = length(lnavData);
    offsetTime = mod(NavdataBitStartIndex-301, navdatalen)*bitDuration;
    inittime = t + offsetTime;

    % To model delay, get one extra navigation bit from the previous bit
    navBitIndices = mod(NavdataBitStartIndex+(-1*numBitsForDelay:(NumNavDataBits-1)), navdatalen);
    navBitIndices(navBitIndices==0) = navdatalen;
    navbits = lnavData(navBitIndices);
    navdata = 1-2*navbits;
    upSampledNavData = repelem(navdata, numPChipsPerNavBit, 1);
```

→ Generate Navigation data object

→ Create navigation data bit streams

Generating GPS Waveform

```
        % Generate P-Code and C/A code
        pgen = gpsPCode("PRNID", PRNIDs(isat), "InitialTime", inittime, ...
            "OutputCodeLength", (NumNavDataBits+numBitsForDelay)*numPChipsPerNavBit);
        pcode = 1 - 2*double(pgen());

        % Reduce the power of the I-branch signal by 3 dB, per IS-GPS-200 [1].
        % See table 3-Va in [1].
        isig = pcode/sqrt(2);

        cacode = 1 - 2*double(gnssCACode(PRNIDs(isat), "GPS"));

        numCACodeBlocks = (NumNavDataBits + numBitsForDelay)*bitDuration*1e3;
        caCodeBlocks = repmat(cacode(:), int64(numCACodeBlocks), 1);

        % Because C/A code is 10 times slower than P-code, repeat each sample
        % of C.A code 10 times
        qsig = repelem(caCodeBlocks, 10, 1);

        % Generate the baseband waveform
        gpsBBWaveform = (isig + 1j*qsig).*upSampledNavData;

        % Initialize the number of samples per bit
        numSamplesPerBit = SampleRate*bitDuration;

        % Introduce Doppler
        numSamplesGPSBB = length(gpsBBWaveform);
        sampleIndices = (0:(numSamplesGPSBB-1));
        ph = sin(dopplerRate(isat)*sampleIndices/(peakDoppler(isat)*10.23e6));
        phase = 2*pi*(peakDoppler(isat)^2)/dopplerRate(isat)*ph;
        bbwave = gpsBBWaveform(:).*exp(1j*phase(:));

        % Rate match the generateed signal to the radio sample rate
        [upfac, downfac] = rat(SampleRate/10.23e6);
        upgcode = repelem(bbwave,upfac,1);
        gpsWaveform = upgcode(1:downfac:end);

        % Get the number of samples for delay
        caCodeRate = 1.023e6;
        numDelaySamples = floor(sigdelay(isat)*SampleRate/caCodeRate);
```

```
        % Add delay to the signal by keeping samples of the previous bit at the
        % beginning of the signal
        delayedSig = gpsWaveform(numSamplesPerBit-numDelaySamples+1:end);

        % Remove the final samples to make all signals of equal length
        delayedSig = delayedSig(1:end-numDelaySamples);

        % Get the composite signal by adding the current satellite signal
        resultsig = resultsig +delayedSig;
    end
```

Configure PlutoSDR

```
%-----------------------------------------------
% CONFIGURE PLUTOSDR

% Configure Pluto radio transmitter
fs = SampleRate;
fc = 2.41e9;
tx = sdrtx('Pluto');
tx.CenterFrequency = fc;
tx.BasebandSampleRate = fs;
tx.Gain = -33;
transmitRepeat(tx, resultsig);

% Configure Pluto radio receiver
rx = sdrrx("Pluto");
rx.CenterFrequency = fc;
rx.BasebandSampleRate = fs;
rx.SamplesPerFrame = 102300;
rx.OutputDataType = "single";
recordDuration = 0.7; % time duration for receiving data, in seconds
rxwaveform = [];
ovrflw_Cnt = 0; % count number of overflows to check discontinuities in reception
loopCnt = round(recordDuration/(rx.SamplesPerFrame/fs));

for i = loopCnt
    [y1,~,of] = rx();
    ovrflw_Cnt = of+ovrflw_Cnt;
    rxwaveform = [rxwaveform; y1];
end

release(tx);
release(rx);
```

Acquisition and Tracking

```
%-----------------------------------------------
% ACQUISITION AND TRACKING

initialsync = gnssSignalAcquirer;
initialsync.SampleRate = SampleRate;

% Consider data that is 1 millisecond long.
numSamples = ceil(SampleRate*IntegrationTime);
[allRxInput,prevSamples] = buffer(rxwaveform,numSamples);
nFrames = size(allRxInput,2);
numdetectsat = 0;
PRNIDsToSearch = 1:32;

for iBuffer = 1:nFrames
    bufferWave = allRxInput(:,iBuffer);

    if iBuffer == 1
        % This example assumes a hot start for all the satellites. Hence,
        % acquisition performed only once in this example. When decoding
        % the almanac data, based on the available satellites, you can
        % perform acquisition for the visible satellites only.
        numSamplesForInitSync = SampleRate*1e-3; % 1 milliseccond
        [y,corrval] = initialsync(bufferWave(1:numSamplesForInitSync),1:32);
        PRNIDsToSearch = (y(y(:,4).IsDetected,1).PRNID).';
        doppleroffsets = (y(y(:,4).IsDetected,2).FrequencyOffset).';
        codephoffsets = (y(y(:,4).IsDetected,3).CodePhaseOffset).';

        % In general, almanac files offer information about available
        % satellites. Because this example does not perform data decoding,
        % it depends on the acquisition results for satellite detection.
        numdetectsat = length(PRNIDsToSearch);
```

Only 1 ms of data is used for acquisition

```
        disp(['The detected satellite PRN IDs: ' num2str(PRNIDsToSearch)])
        if(numdetectsat~=0)
            if ShowVisualizations == 1
                figure;
                mesh(initialsync.FrequencyRange(1):initialsync.FrequencyResolution:initialsync.FrequencyRange(2),
                    0:size(corrval,1)-1,corrval(:,:,1));
                xlabel("Doppler Offset")
                ylabel("Code Phase Offset")
                zlabel("Correlation")
                msg = "Correlation Plot for PRN ID: " + PRNIDsToSearch(1);
                title(msg)
            end
        end

        % Initialize all the properties which must be accumulated.
        accuph = zeros(nFrames,numdetectsat); % Each column represents data from a satellite
        accufqy = zeros(nFrames,numdetectsat);
        accufqyerr = zeros(nFrames,numdetectsat);
        accupherr = zeros(nFrames,numdetectsat);
        accuintegwave = zeros(nFrames,numdetectsat);
        accudelay = zeros(nFrames,numdetectsat);
        accudelayerr = zeros(nFrames,numdetectsat);

        % Create the signal tracker object that tracks phase, frequency,
        % and delay in the signal
        carrierCodeTrack = gnssSignalTracker;
        carrierCodeTrack.SampleRate = SampleRate;
        carrierCodeTrack.IntermediateFrequency = 0;
        carrierCodeTrack.PLLNoiseBandwidth = PLLNoiseBandwidth;
        carrierCodeTrack.FLLNoiseBandwidth = FLLNoiseBandwidth;
        carrierCodeTrack.DLLNoiseBandwidth = DLLNoiseBandwidth;
        carrierCodeTrack.IntegrationTime = IntegrationTime;
        carrierCodeTrack.PRNID = PRNIDsToSearch;
        carrierCodeTrack.InitialFrequencyOffset = doppleroffsets;
        carrierCodeTrack.InitialCodePhaseOffset = codephoffsets;
    end
```

Tracking part

Acquisition and Tracking

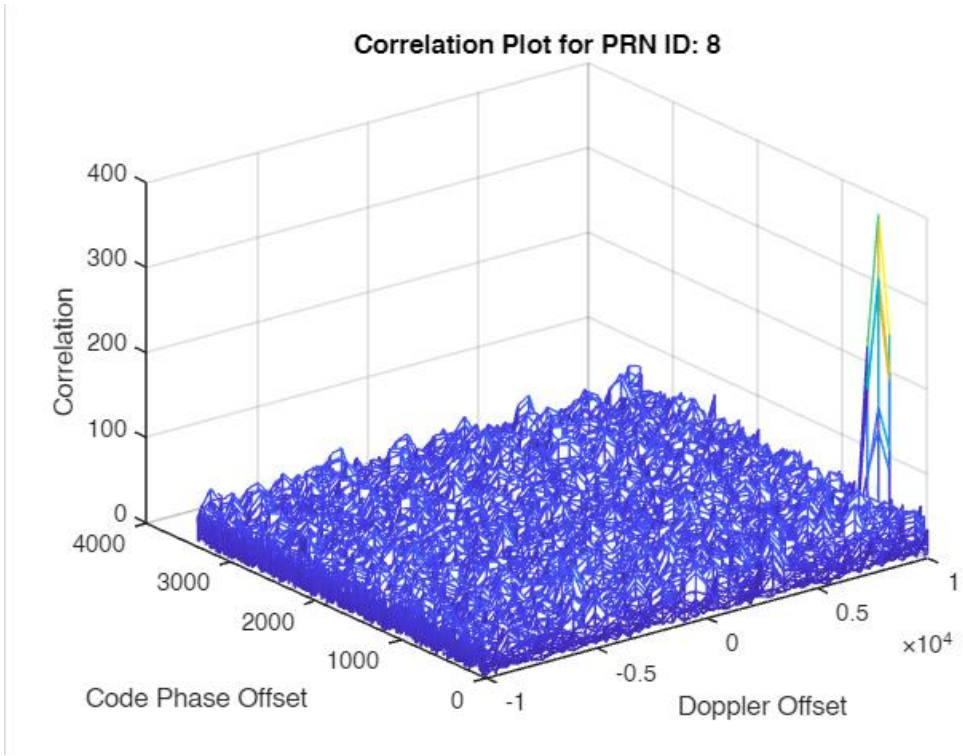
```
    [integwave,trackInfo] = carrierCodeTrack(bufferWave);

    % Accumulate the values to see the results at the end
    accuintegwave(iBuffer,:) = integwave;
    accufqyerr(iBuffer,:) = trackInfo.FrequencyError;
    accufqy(iBuffer,:) = trackInfo.FrequencyEstimate;
    accupherr(iBuffer,:) = trackInfo.PhaseError;
    accuph(iBuffer,:) = trackInfo.PhaseEstimate;
    accudelayerr(iBuffer,:) = trackInfo.DelayError;
    accudelay(iBuffer,:) = trackInfo.DelayEstimate;
end
```

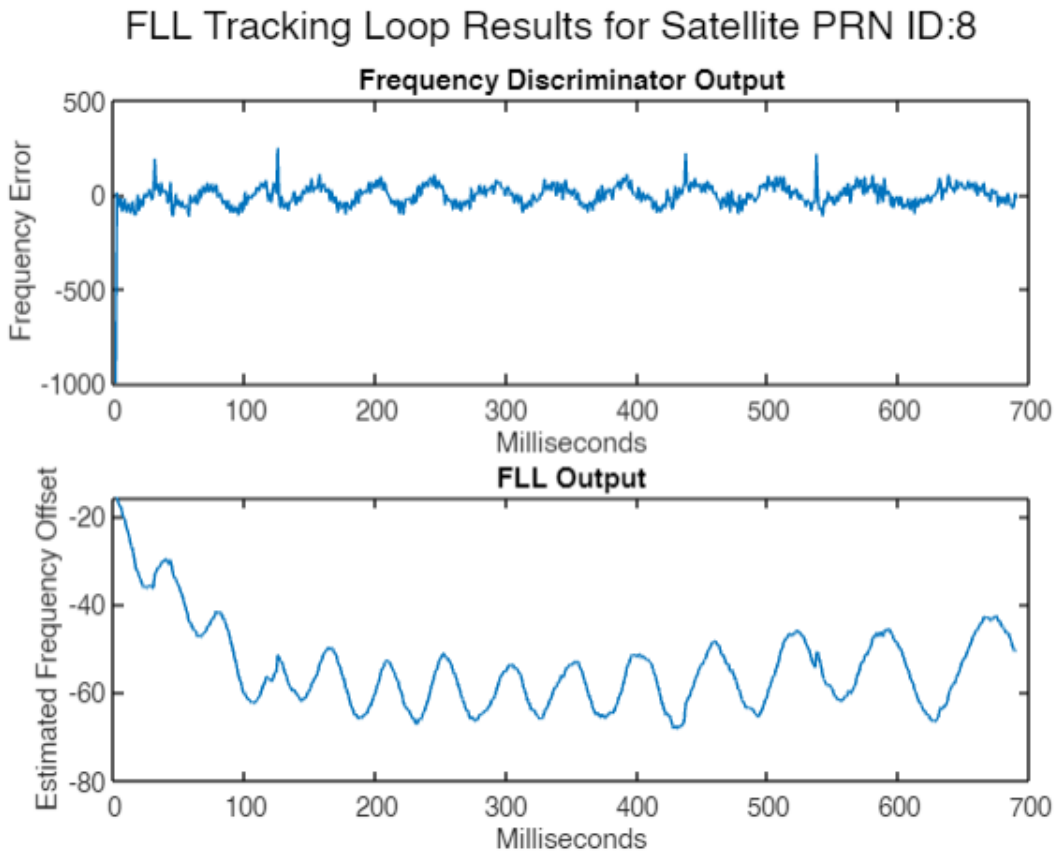
Dump of tracking results

```
if ShowVisualizations == 1
    for isat = 1 % To see tracking results for all the detected satellites by using above line
        groupTitle = ['Tracking Loop Results for Satellite PRN ID:', ...
            num2str(PRNIDsToSearch(isat))];

        figure

        % Plot the frequency discriminator output
        subplot(2,1,1)
        plot(accufqyerr(:,isat))
        xlabel('Milliseconds')
        ylabel('Frequency Error')
        title('Frequency Discriminator Output')

        % Plot the FLL output
        subplot(2,1,2)
        plot(accufqy(:,isat))
        xlabel('Milliseconds')
        ylabel('Estimated Frequency Offset')
        title('FLL Output')
        sgtitle(['FLL ' groupTitle])

        figure

        % Plot the phase discriminator output
        subplot(2,1,1)
        plot(accupherr(:,isat))
        xlabel('Milliseconds')
        ylabel('Phase Error')
        title('Phase Discriminator Output')

        % Plot the PLL output
        subplot(2,1,2)
        plot(accuph(:,isat))
        xlabel('Milliseconds')
        ylabel('Estimated Phase')
        title('PLL Output')
```

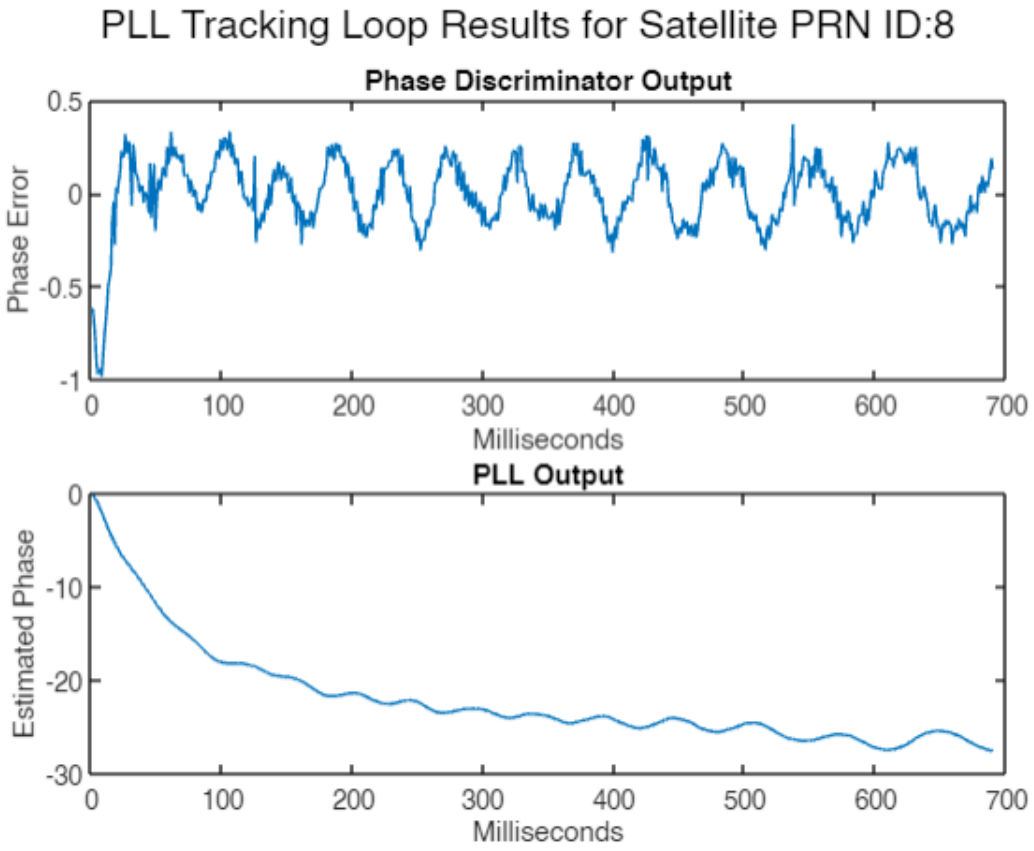
Plot of Tracking results

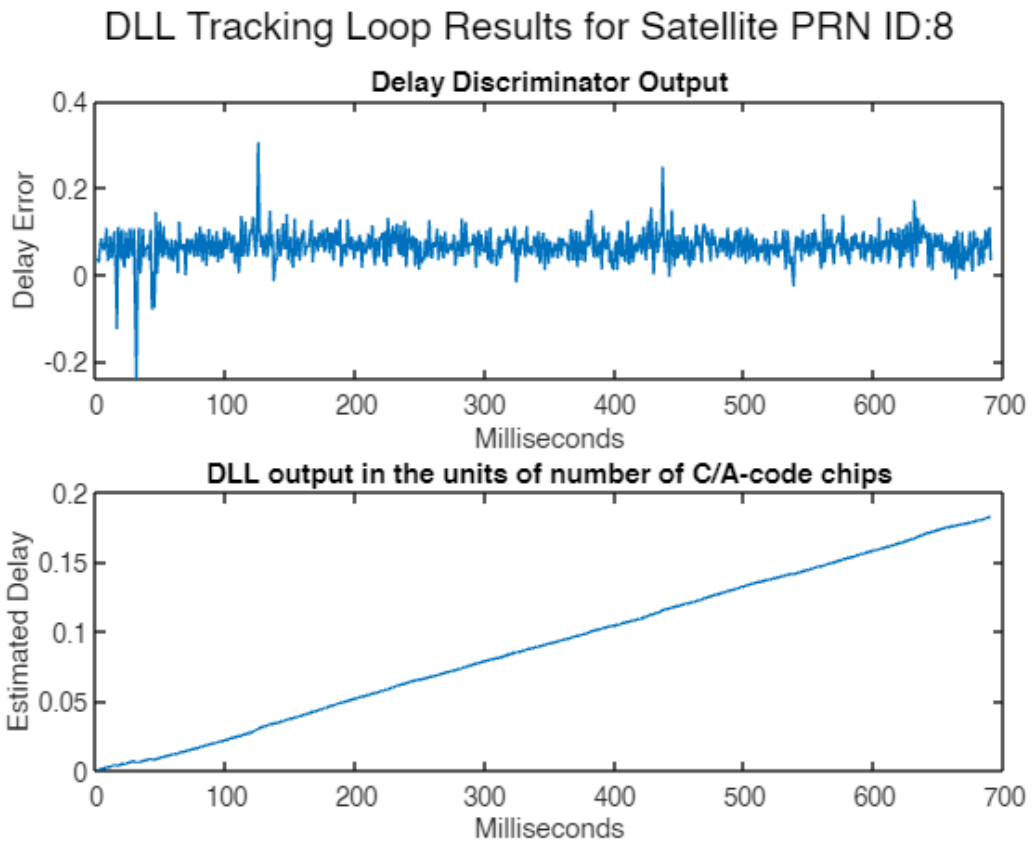
Results


Correlation Plot for PRN ID: 8
Correlation
400
300
200
100
0
4000
3000
2000
1000
0
Code Phase Offset
-1
-0.5
0
0.5
1
×10^4
Doppler Offset

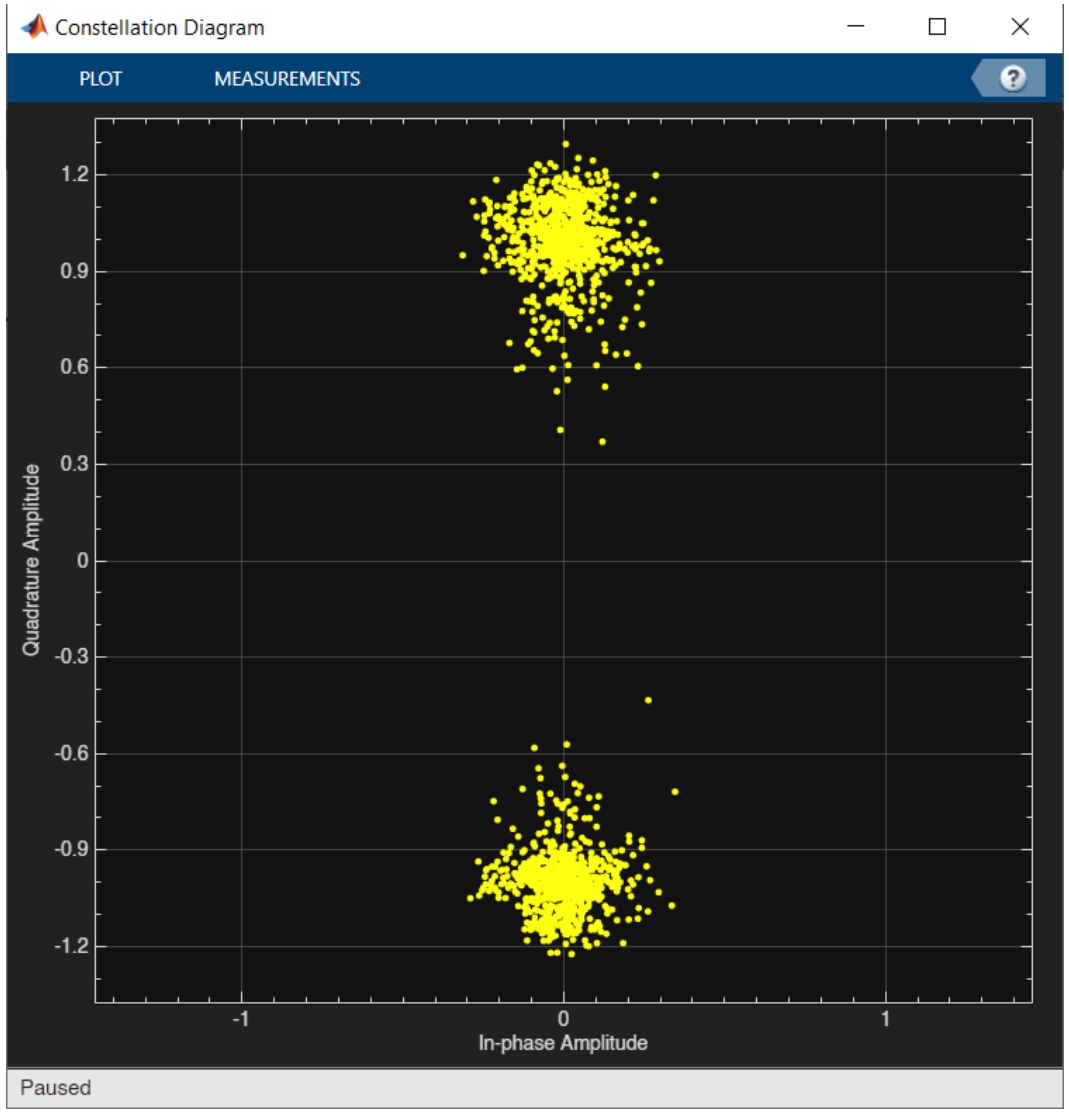

FLL Tracking Loop Results for Satellite PRN ID:8
Frequency Discriminator Output
Frequency Error
500
0
-500
-1000
0
100
200
300
400
500
600
700
Milliseconds
FLL Output
Estimated Frequency Offset
-20
-40
-60
-80
0
100
200
300
400
500
600
700
Milliseconds

Results


PLL Tracking Loop Results for Satellite PRN ID:8
Phase Discriminator Output
Phase Error
0.5
0
-0.5
-1
0
100
200
300
400
500
600
700
Milliseconds
PLL Output
Estimated Phase
0
-10
-20
-30
0
100
200
300
400
500
600
700
Milliseconds


DLL Tracking Loop Results for Satellite PRN ID:8
Delay Discriminator Output
Delay Error
0.4
0.2
0
-0.2
0
100
200
300
400
500
600
700
Milliseconds
DLL output in the units of number of C/A-code chips
Estimated Delay
0.2
0.15
0.1
0.05
0
0
100
200
300
400
500
600
700
Milliseconds

Results


Constellation Diagram
PLOT
MEASUREMENTS
1.2
0.9
0.6
0.3
0
-0.3
-0.6
-0.9
-1.2
Quadrature Amplitude
-1
0
1
In-phase Amplitude
Paused