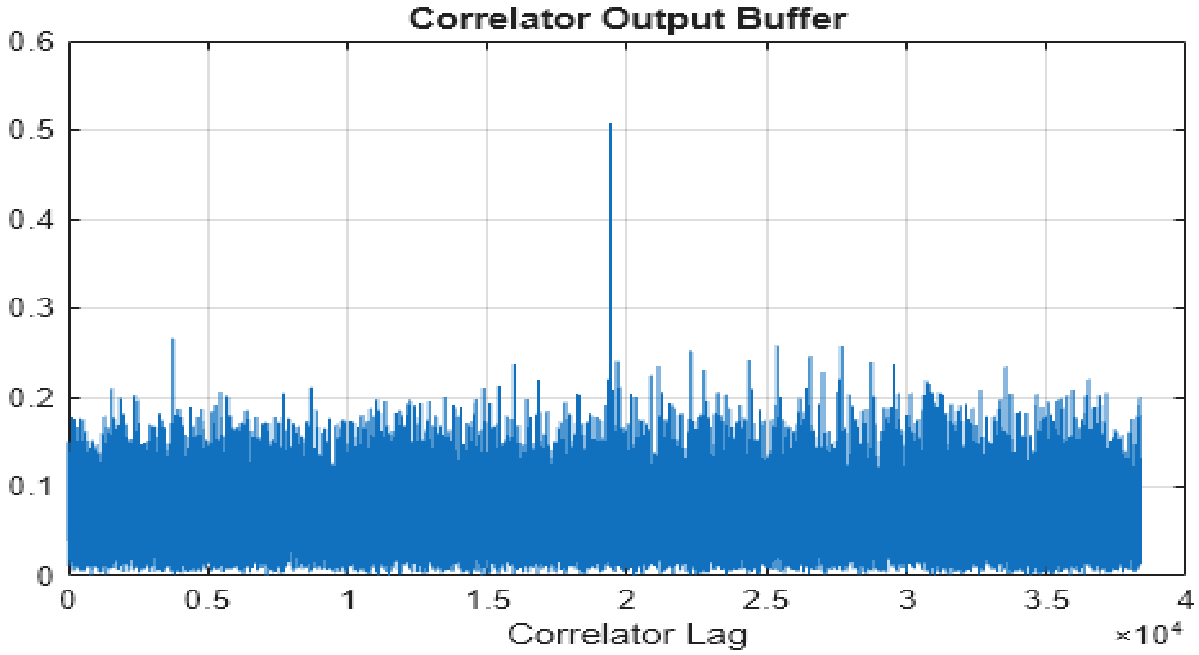
Correlate the preamble sequence against the sample buffer across different frequency offsets to detect the frequency-shifted preamble.

```
    % Increase to detect the preamble symbols after AGC convergence during the first
    % few preamble symbols
    preambleDetectionOffset = 200; % in symbols
    % Shorten the length of the preamble to correlate against to avoid frequency offset
from
    % decorrelating the reference preamble with the received preamble
    preambleDetectionLength = 175; % in symbols
    tx.preambleChips =
preambleQPSK(1+preambleDetectionOffset:preambleDetectionLength+preambleDetectionOffset);

    % Convert from OQPSK to QPSK for symbol-based preamble detection
    sampleBufferQPSK = [real(rxAGCSamples(1:end-sps/2)) +
1i*imag(rxAGCSamples(sps/2+1:end)); zeros(sps/2,1)];

    % Detect the preamble across different frequency offsets
    for preambleFreqOffset = -settings.maxDoppler:150:settings.maxDoppler
        % Add frequency offset to the preamble
        preambleShifted =
frequencyOffset(tx.preambleChips,system.chipRate,preambleFreqOffset);

        % Get the start index of the preamble. Look for a preamble only in the
        % first half of the double-buffered sample buffer.
        [startSampIdx,corrBuffer] = helperPolyphaseCorrelator( ...

sampleBufferQPSK(1:numBurstSamples),preambleShifted,sps,preambleDetectionOffset);
        if ~isempty(startSampIdx)
            break; % break out of frequency search loop
        end
    end
    if isempty(startSampIdx)
        disp(['Preamble not found after ', num2str(n), ' seconds']);
    else
        break; % break out of preamble search loop
    end

end
```

```
Preamble not found after 1 seconds
Preamble not found after 2 seconds
Preamble not found after 3 seconds
Preamble not found after 4 seconds

Found preamble at correlation buffer number 6, index 19201, sample index 153602
```

```
if isempty(startSampIdx)
    error('Preamble not detected amongst simulation samples.');
end

% Plot the correlator output
figure;
plot(abs(corrBuffer));
grid on;
title('Correlator Output Buffer');
xlabel('Correlator Lag');
```



```
% Extract the transmission burst from the sample buffer. Collect 20 more
% chips at the end for possible timing adjustments
rxBurst = sampleBuffer(startSampIdx:startSampIdx+((numChips+20)*sps)-1);
```

Frequency and Phase Offset Compensation

Frequency and timing offset will cause the constellation of the detected signal to rotate and display nonconstant envelope characteristics. The downstream receiver processing detects the offsets and corrects them to produce a stable signal suitable for demodulation and decoding. Use a `comm.CoarseFrequencyCompensator` System object to estimate the carrier frequency offset over the received burst and apply the frequency offset to the sample buffer.

```
% Estimate and correct carrier offset
coarse = comm.CoarseFrequencyCompensator( ...
    Modulation =             "OQPSK", ...
    SamplesPerSymbol =       sps, ...
    SampleRate =             fs, ...
    FrequencyResolution =    1);
[coarseSyncOut,coarseEst] = coarse(rxBurst);
fprintf('Estimated coarse frequency offset = %6.3f kHz\n',coarseEst/1000);
```

```
Estimated coarse frequency offset = -8.281 kHz
```

```
fprintf('Estimation error = %6.3f Hz\n',coarseEst-
(dopplerShift+impairments.rxFreqOffset));
```

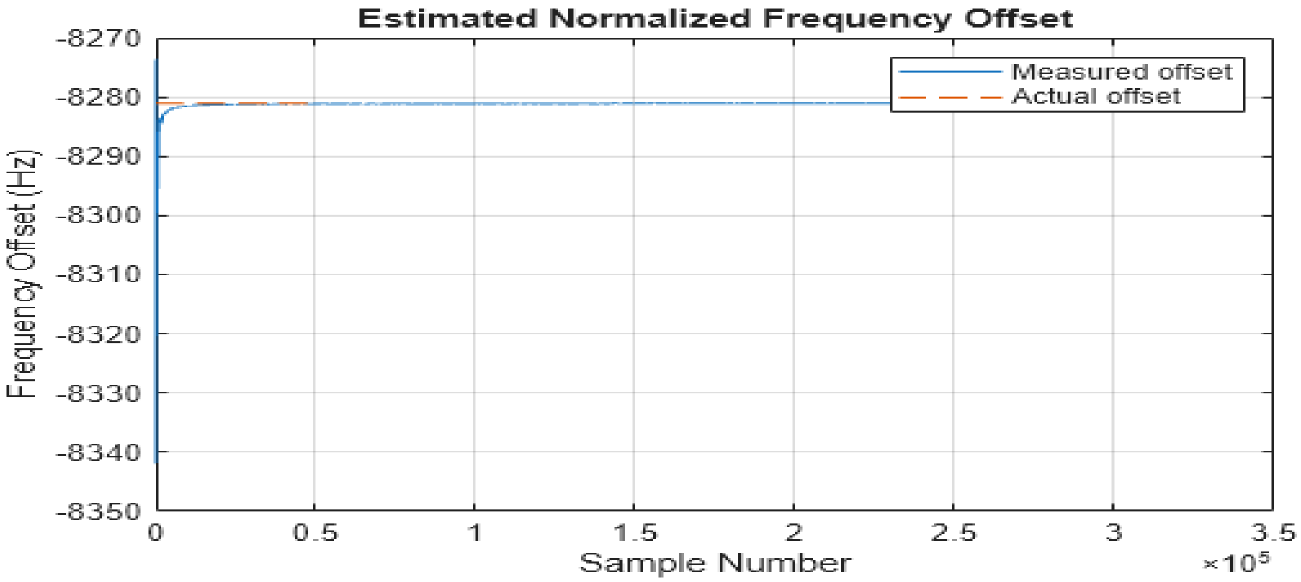
```
Estimation error =  0.004 Hz
```

Use a `comm.CarrierSynchronizer` System object to detect and correct smaller carrier frequency and phase offsets. This object aligns the received constellation to the QPSK reference points to compensate for phase offset.

```
carrierSync = comm.CarrierSynchronizer( ...
    Modulation =                "OQPSK", ...
    NormalizedLoopBandwidth=    0.01, ...
    DampingFactor =             0.707, ...
    SamplesPerSymbol =          sps);
[carrierSyncOut,phErr] = carrierSync(coarseSyncOut);
```

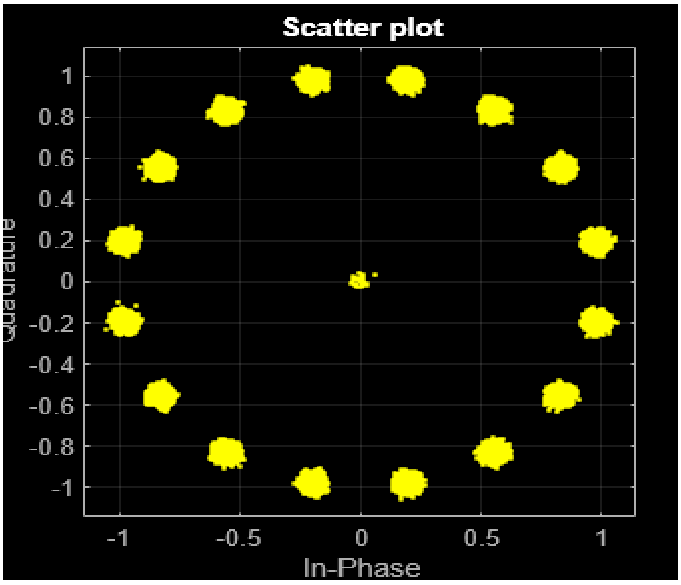
Show that the frequency offset estimation converges to the actual frequency offset at the end of the preamble and prior to the start of the payload data.

```
estFreqOffset = diff(phErr) * fs / (2*pi);
rmean = cumsum(estFreqOffset)./(1:length(estFreqOffset))';
figure;
plot(coarseEst+rmean);
title('Estimated Normalized Frequency Offset');
xlabel('Sample Number');
ylabel('Frequency Offset (Hz)');
hold on;
plot((dopplerShift+impairments.rxFreqOffset)*ones(fs*(system.preambleLength/
packetLength),1),'--');
legend({'Measured offset','Actual offset'});
grid;
```



Plot the frequency-corrected constellation of the payload data. The constellation should now have a near-constant envelope shape.

```
figure;
scatterplot(carrierSyncOut(numPreambleChips+1:end))
grid on
```



Path Detection

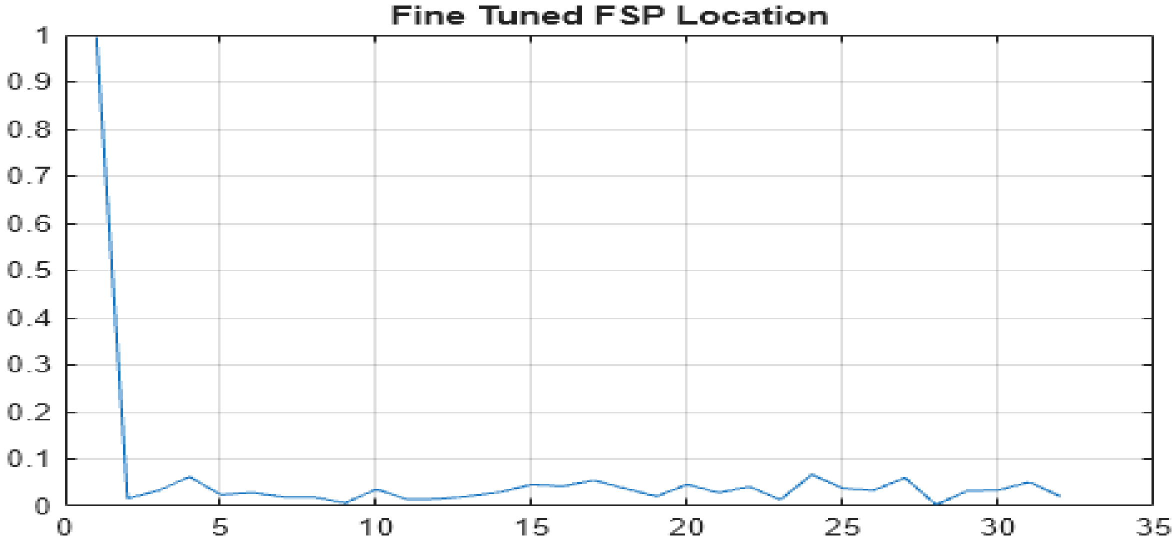
The location of the preamble start might have shifted slightly after frequency offset correction. Repeat preamble detection using the entire preamble for maximum resolution.

```
% Convert from OQPSK to QPSK
```

```
carrierSyncOutQPSK = [real(carrierSyncOut(1:end-sps/2)) +
1i*imag(carrierSyncOut(sps/2+1:end)); zeros(sps/2,1)];

preambleOffsetChips = 5000; % preamble correlation offset in chips
preambleOffset = preambleOffsetChips / 2;
preamble0 = preambleQPSK(preambleOffset+1:end);
[FSPSampIdx,corrBuffer2] =
helperPolyphaseCorrelator(carrierSyncOutQPSK,preamble0,sps,preambleOffset);
```

```
Found preamble at correlation buffer number 3, index 1, sample index 1
```

```
if isempty(FSPSampIdx)
    error('Preamble lost. Check frequency tracking.');
end

% Plot the correlator output
figure;
plot(abs(corrBuffer2(1+preambleOffset:32+preambleOffset)));
grid on;
title('Fine Tuned FSP Location');
```



Timing Recovery, Demodulation, and Phase Resolution

After frequency and phase offset compensation, perform timing recovery on the signal. During OQPSK timing recovery, a `comm.SymbolSynchronizer` System object delays the quadrature component, performs synchronization, and converts the oversampled OQPSK signal to symbol-rate QPSK such that the output is a conventional QPSK constellation.

```
% Timing recovery of OQPSK signal, via its QPSK-equivalent version
symbolSynchronizer = comm.SymbolSynchronizer( ...
    Modulation =       'OQPSK', ...
    NormalizedLoopBandwidth = 0.001, ...
    DetectorGain =     4, ...
    DampingFactor =    2, ...
    SamplesPerSymbol = sps);
[syncedQPSK,timingError] = symbolSynchronizer(carrierSyncOut(FSPSampIdx:end));

figure;
scatterplot(syncedQPSK(numPreambleChips+1:end)); % plot the payload constellation
title('Synchronized Payload Constellation');
```