Capon beamforming: Algorithm and Implementation

Algorithm descriptions

Let s(t) be the incoming waves after mixing to baseband, the sensor array signal to be processed is given by

$X(t) = A(\theta)s(t) + n(t)$

where $A(\theta) = (a(\theta_1), \ldots a(\theta_M))$ is the steering matrix
$a(\theta) = (e^{j2\pi y_1 \sin(\theta)}, \ldots, e^{j2\pi y_N \sin(\theta)})$ is the steering vector
M is number of angle bins
y_n is the sensor position normalized by wavelength

Capon BF approach is
$\theta_{capon} = \text{argmin}_{\theta}\{\text{trace}(A(\theta)^* R_n^{-1} {}^* A(\theta)^H\}$
where R_n is the spatial covariance matrix

Implementation details

- Assumptions:
  - Receive antennas are equally spaced with distance λ/2
- Assuming slow motion scenario, Rn can be constructed using multiple chirps within a frame.
- Calculation of the solution can be significantly reduced with a(θ) constructed for equally spaced antenna, combining with the fact that Rn is an Hermitian matrix that is also persymmetric.
- Current implementation of the module consists of the following sub blocks:
  - Static clutter removal is option is added before Rn matrix generation by removing DC components per range bin. Input assumes 16-bit I/Q, and output is also in 16-bit I/Q format (trade precision for memory and cycles).
  - Per range bin, construct Rn using multiple chirps within a frame. Then Rn is inverted and the upper diagonal of the R_n^{-1} is stored in memory for each range bin.
  - Per range bin, calculate the Capon BF solution and store the angle spectrum in memory to construct the range-azimuth heatmap.
    - Since conventional BF using covariance matrix has very similar solution $\theta_{conventional} = \text{argmax}_\theta\{\text{trace}(A(\theta)*R_n*A(\theta)^H\}$, we have a fallback flag to use the same code to calculate conventional BF.
  - After detection, per detected point, estimate Doppler using the Capon beamweights and Doppler FFT.
  - Hardcoded for 4 and 8 antennas, and focused on 8-antenna test

Initial Benchmarks for Capon BF (C674x)

- **Loop summary**:
  - **Range-Azimuth heatmap generation, per range bin**
    - Clutter removal: mean calc: ii = 5 per 8 samples (chirp samples), remove mean: ii = 2 per 4 samples (chirp samples)
      total: (5/8+2/4)*8*128 = 1152
    - Rn calculation: diagonal: ii = 6 for 8 samples (chirp samples) off-diagonal: ii = 6 for 4 samples (chirp samples)
      total: [8 * (6/8) + 28 * (6/4)] * 128 = 6144
    - Rn inverse: total 4000 (measured by Cesar)
    - Heatmap calc: ii = 15 per angle bin
      total 15*64 = 960
    - Heatmap total: 12256 per range bin
  - **Doppler Est, per detected Obj:**
    - beamforming output: ii = 17 per chirp samples for all antennas
      total: 17*128 = 2176
    - 128 size FFT :
      total 1400 cycles per DSPLIB formula
    - peak search: ii = 2 per Doppler bin
      total 2* 128 = 256
    - Doppler Est total: 3832 per detected Obj
- **Measurement from 6745 Cycle-Accurate Simulator**
  - **Flat memory CPU cycles**:
    - Heatmap: 12910 per range bin
    - DopplerEst: 4606 per detected Obj
  - **Cycle accurate cycle measurement with cache misses simulated**
    - Heatmap: 22122 per range bin
    - DopplerEst: 7792 per detected Obj

Memory Usage

- Data memory usage:
  - Range-Azimuth heatmap: Nrange * Nangle * 4bytes/float
  - Inv(Rn) for all range bins: Nrange * [Nant * (Nant+1)/2] * 8bytes/complex-float
  - Memory for local instances (twiddle etc): DopplerFFTsize*8bytes/complex-float + 48 * 4bytes
  - Scratch memory: 8-antenna:max{2*DopplerFFTsize*8bytes, 360*4bytes }, 4-antenna:max{2*DopplerFFTsize*8bytes, 82*4bytes }
- Code size:
  - Signal processing blocks: ~21000 bytes
  - Module Initialization functions: 1100 bytes