

Phase Synchronization Capability of the Analog Devices FMCComms5 and DoA Estimation

Introduction

This article will provide instructions for demonstrating the phase synchronization capability of Analog Devices FMCComms5 and a direction finding implementation that exploits this functionality. FMCComms5 evaluation board can achieve a high degree of accurate phase synchronization and alignment. In this article, we will discuss how two AD9361 transceivers are phase aligned through a specific software library called *libad9361* which is built upon the *libiio* I/O infrastructure. Phase alignment is necessary for many radar applications, such as direction finding, and coherent MIMO applications.

The GNURadio source, which this example was base around, was originally developed by Srikanth Pagadarai, PhD and Travis Collins, PhD funded by Ettus Research [1]. It has been ported to the FMCComms5 platform recently, including documentation, by Dr. Collins and is currently available at github.com/tfcollins/gr-doa under the *adi* branch. This code maintains a GPL3 license. The FMCComms5 implementation maintains identical performance to the previous work performed here [1]. The whitepaper from [1] as well has been updated by the original authors here to cover FMCComms5 and its implementation strategy.

Motivation

Unlike many existing SDR solutions on the market that offer multiple receiver or transmit channels that are coherent, the FMCComms5 solutions provides true phase alignment across all RF channels without the need for additional hardware. Since the board itself utilizes a single oscillator once channels are aligned the channels will remain synchronized over time. This makes the FMCComms5 an ideal solution for development of radar and MIMO applications.

This platform also demonstrates how multiple AD9361 transceivers can be synchronized if more channels are required for a customer's own hardware solution. For convenience Analog Devices provides full schematics and layout of the FMCComms5 board to help customers implement their own designs. These are available under the hardware section at the FMCComms5 wiki page from Analog Devices as well [2].

Phase Alignment

The FMCComms5 evaluation board utilizes two AD9361 transceivers, providing a 4x4 system where all receivers and transmitters are phase aligned. This requires no external cabling or splitters since the FMCComms5 board is outfitted with a RF switches allowing phase matched analog loopback between transceivers. Since each individual AD9361 contains two transceivers, which share a local oscillator (LO) at matched lengths, it is only necessary to phase calibrate between individual chips.

From the hardware perspective, to provide this alignments there are three unique design characteristics on the FMCComms5 evaluation board. The first is that all traces to external SMA connectors are length matched between all channels. This is obvious based on the positions of the AD9361 transceivers in Figure 1. Second, ADG918 wide band (-3dB @ 4GHz) RF switches are is placed between the transceivers which are also length matched to and from the switches. Therefore, we are able to send signals from either chip to either chip by toggling switch states.

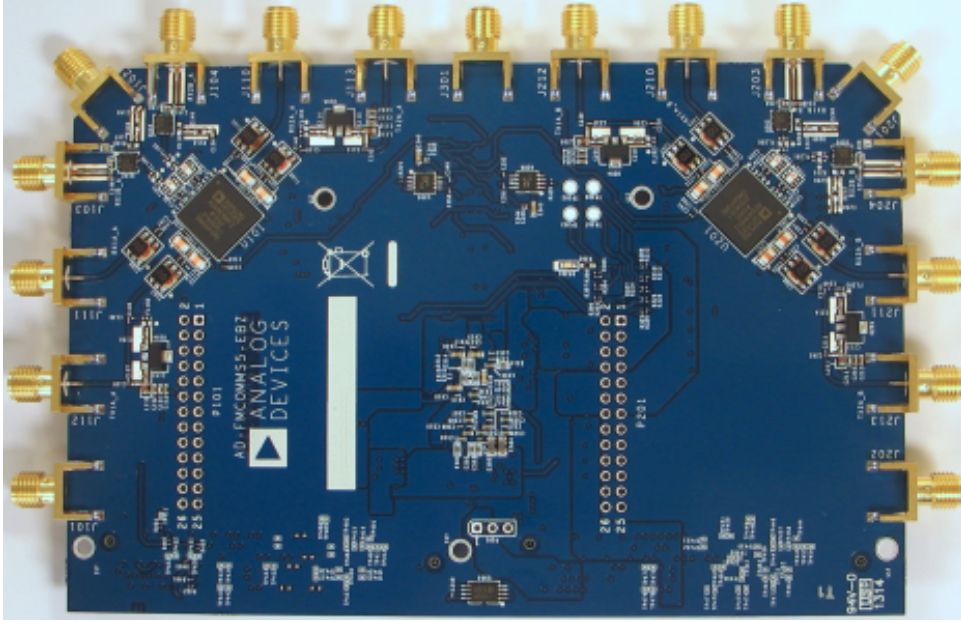


Figure 1: FMComms5 evaluation board with dual AD9361 transceivers.

Table 1: FMComms5 RF switch states input/output mapping.

Output/Input	RX Chip 1	RX Chip 2
TX Chip 1	1	3
TX Chip 2	2	4

These states and their associated input/output mappings are outlined in Table 1. The switch itself is connected to GPIO pins on the FPGA, allowing for easy programmability.

From software the entire process is managed through a function call to the **libad9361** library. This is available at github.com/analogdevicesinc/libad9361-iio. Since phase alignment calibrations are only valid for specific LO and sample rate settings, those parameters are required by the calibration routine. As long as these settings are maintained this calibration step only needs to be performance once.

During the phase alignment process the FPGA itself is used as a reference mechanism between the chip since it receive data streams from both chips. The first step of the procedure is to align the chip sampling instances, through the function call `ad9361_fmcomms5_multichip_sync`. This will force chip B to be clocked off of chip A with samples passed to the FPGA occurring at the same time instances. At this point the channels are essentially coherent at baseband but not aligned in phase from the RF perspective.

To perform the final alignment requires a three step process all which happens in the `ad9361_fmcomms5_phase_sync` function call and is visually outlined in Figures 2, 3, and 4. In step 1 we first loopback chip A's transmitter back into chip A, all occurring on channel 1.

On channel 2 of chip A we perform digital loopback, which provides our common reference. Note that this channel assigned is from the perspective of the FPGA, which we can map to be purely digital or analog loopback. This process utilizes the DDSs on the transmitter whose phases are easily manipulated in software. The difference between channel 1 (analog loopback) and channel 2 (digital loopback) is measured using an average between their instantaneous phases. This difference is used to manipulate the DDS phase until this difference equals zero. This reverse rotation can then be applied to the receivers of chip A.

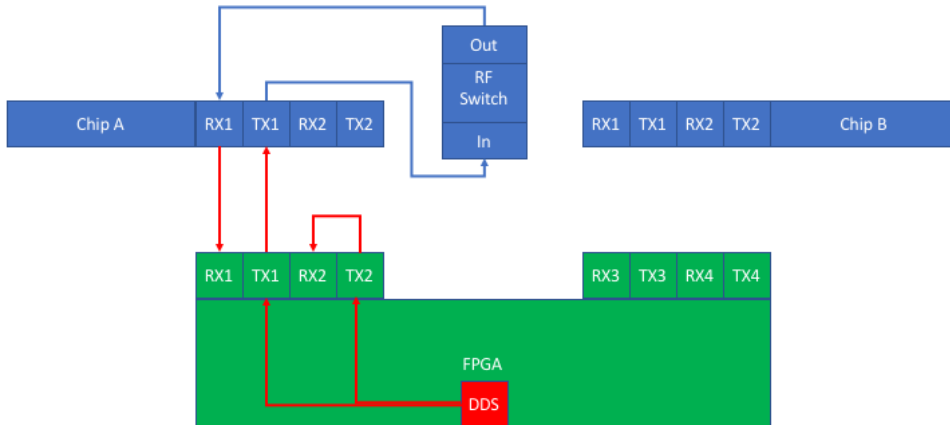


Figure 2: Step 1 of phase calibration where FPGA digital loopback offset is measured against Chip A’s analog loopback through itself.

Next we change the RF switch to loopback the DDS signal transmitted out channel 1 of Chip B, outlined in Figure 3. Then we perform the same process of modifying the DDS phase of the analog channel in reference to the digital loopback. Note that the digital loopback of any of the receive channels will be identical since they are sample synced from the `ad9361_fmcomms5_multichip_sync` function call and we are only in the digital domain.

The final step here is to align the transmitters between the chip A and chip B. To do this all we need to do is repeat step 1, but on chip B. As shown in Figure 4 we loopback chip B’s transmitter back to itself, then drive the difference between the digital and analog references to zero with the DDS phase on channel 1.

Once complete all transmitters will be phase aligned and all receivers will be aligned as well. Using an external power splitter with matched length cables we can view a transmitted sinusoid completely aligned as in Figure 5.

The output of the FMComms5 with compensated relative phase offsets can be observed by running `view_FMComms5_op_with_corrected_phase_offsets.grc` flowgraph in the `apps` directory. Ensure that the parameters selected are the same as those used for measuring the offsets. Observe that the time-scope display is similar to that shown in Figure 5. The accuracy of the phase offset compensation can also be saved in a numerical format by executing the flowgraph, `calculate_FMComms5_phase_sync_accuracy.grc`.

We must note, this calibration is valid for a particular choice of center frequency and sample rate only. The phase offset measurement and correction step needs to be repeated when these operating parameters are modified. As a result of this calibration step, the relative phase offsets observed between the four streams when receiving signals across an antenna-array will be a function of the *array-manifold* alone. Any user-developed array signal processing application

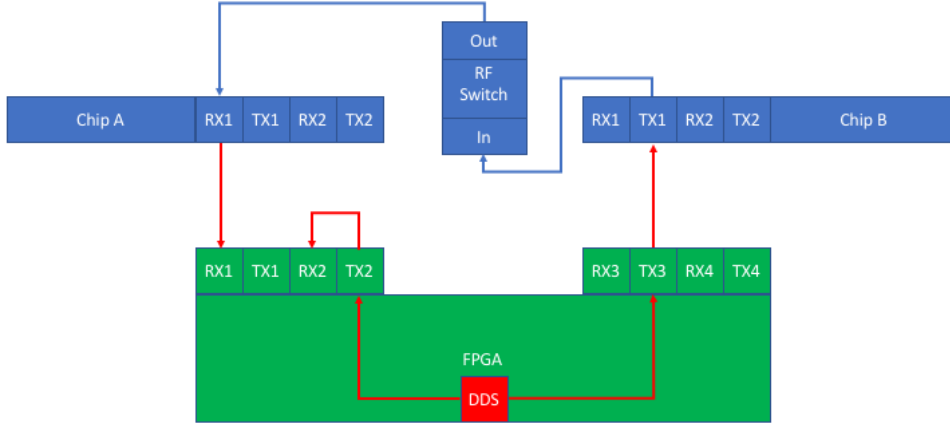


Figure 3: Step 2 of phase calibration where FPGA digital loopback offset is measured against Chip A's analog loopback through Chip B.

that requires accurate phase synchronization between receive channels will need to incorporate this compensation step as the first stage of processing. To demonstrate how this is done, we will show two direction-of-arrival (DoA) estimation algorithms in action.

Signal and Noise Subspaces

First, we look at a brief theoretical overview of signal and noise subspaces. Subspace methods are widely used for DoA estimation and rely on exploiting the orthogonality between signal and noise subspaces. For detailed background, see [3]. Suppose that an antenna array is composed of N isotropic sensors which are located at positions, p_n for $n \in \{0, 1, \dots, N-1\}$. Suppose that $x(t)$ is the signal received by the antenna array and it consists of D directional plane-wave processes plus white sensor noise. That is,

$$x(t) = \sum_{d=1}^D \tilde{f}_d(t) \mathbf{v}(\mathbf{k}_d) + w(t).$$

where, $\mathbf{v}(\mathbf{k})$ is the *array manifold vector*,

$$\mathbf{v}(\mathbf{k}) = \left[e^{-j\mathbf{k}^T p_0} \quad e^{-j\mathbf{k}^T p_1} \quad \dots \quad e^{-j\mathbf{k}^T p_{N-1}} \right]^T \quad (1)$$

evaluated for the specific *wave number*¹, \mathbf{k}_d associated with the d th target. Suppose that $\tilde{f}_d(t)$, for $d = 1, 2, \dots, D$, is bandlimited to W and we sample $x(t)$ every $1/W$ seconds to obtain a discrete-time domain snapshot model. By denoting the snapshots as $\mathbf{x}(k)$, for $k = 1, 2, \dots, K$, we can write the correlation matrix of the input streams as follows:

$$\mathbf{R}_x(k) = E [\mathbf{x}(k) \mathbf{x}^H(k)] = \mathbf{V} \mathbf{R}_f \mathbf{V}^H + \sigma_w^2 \mathbf{I}. \quad (2)$$

¹The **wavenumber**, \mathbf{k} for a plane wave as:

$$\mathbf{k} = -\frac{\Omega}{c} \mathbf{u} = -\frac{2\pi}{\lambda} \mathbf{u} = -\frac{2\pi}{\lambda} [\sin(\theta) \cos(\phi) \quad \sin(\theta) \sin(\phi) \quad \cos(\theta)]^T,$$

where, \mathbf{u} is the directional cosines vector.

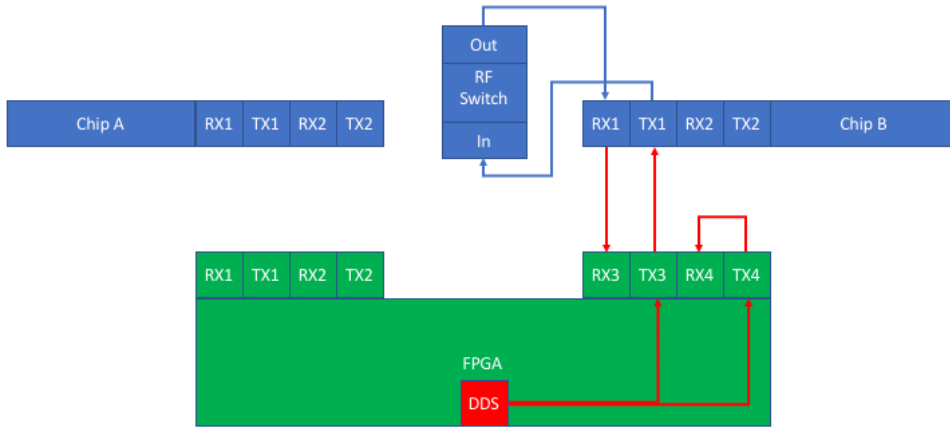


Figure 4: Step 3 of phase calibration where FPGA digital loopback offset is measured against Chip B's analog loopback through itself.

In practice, we construct a *sample* correlation matrix as an estimate for the true autocorrelation matrix. The method chosen for constructing the sample correlation matrix influences the performance of the DoA estimators. One approach to constructing a sample correlation matrix is to collect a sample during each time snapshot, across N array elements into a $N \times 1$ vector $\mathbf{x}(k)$, and perform the following operation:

$$\mathbf{C}_x = \frac{1}{K} \sum_{k=1}^K \mathbf{x}(k) \mathbf{x}^H(k). \quad (3)$$

where, K is the number of snapshots. An alternative way of computing this sample correlation matrix is to collect samples across several time snapshots, into a $N \times K$ matrix \mathbf{X}_K , and perform the following operation:

$$\mathbf{C}_x = \frac{1}{K} \mathbf{X}_K \mathbf{X}_K^H. \quad (4)$$

It has been argued that performing an additional Forward-Backward Averaging step in determining the sample correlation matrix will result in superior DoA estimator performance [3]. It can be written as follows,

$$\mathbf{C}_x = \frac{1}{2K} \mathbf{X}_K \mathbf{X}_K^H + \frac{1}{2K} \mathbf{J} \mathbf{X}_K^* \mathbf{X}_K^T \mathbf{J}. \quad (5)$$

where, \mathbf{J} is a square $N \times N$ *reflection matrix* whose elements on the cross diagonal are unity and all other elements are zero. Similar to (2), we can write the sample correlation matrix as follows:

$$\mathbf{C}_x = \mathbf{V} \mathbf{R}_f \mathbf{V}^H + \sigma_w^2 \mathbf{I}. \quad (6)$$

An important assumption implicit in the subspace methods for DoA estimation is that no two of the D signals are coherent, *i.e.*, \mathbf{R}_f is positive definite. In (2) and (6), we defined the $N \times D$

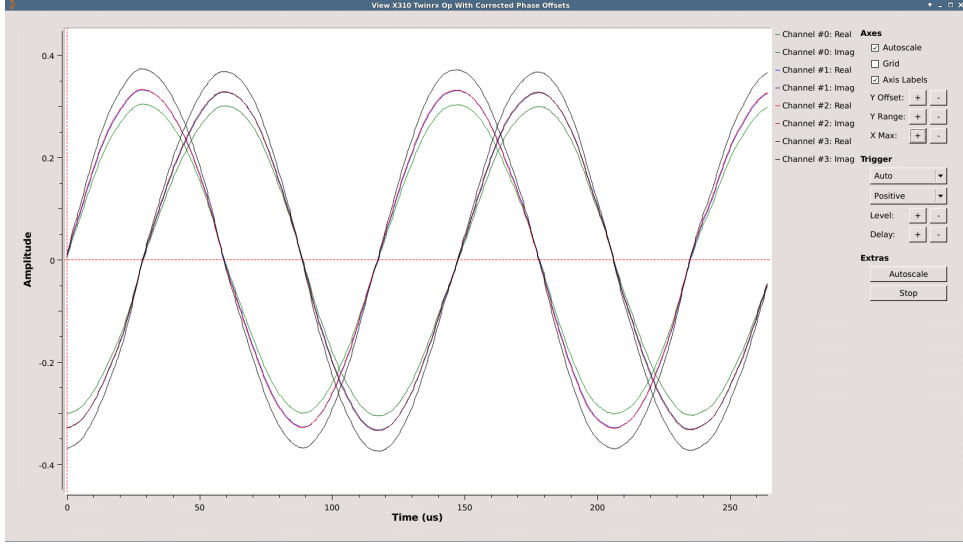


Figure 5: Time Domain Display Showing Received Signal with Phase Compensation.

array manifold matrix, \mathbf{V}^2 as follows:

$$\mathbf{V} := [\mathbf{v}(\mathbf{k}_1) \quad \mathbf{v}(\mathbf{k}_2) \quad \dots \quad \mathbf{v}(\mathbf{k}_D)]^T. \quad (7)$$

The columns of \mathbf{V} span a D -dimensional subspace that contains all of the signal energy. We can obtain the orthogonal basis of this subspace, $\{\Phi_d\}$ for $d = 1, 2, \dots, D$ as follows. The first step is to recognize that Φ_d is the linear transformation, $\Phi_d = \mathbf{V}\mathbf{c}_d$ for some $D \times 1$ -vector, \mathbf{c}_d . Now, the eigenvalue decomposition (EVD) of the first term in (6) is:

$$\begin{aligned} \lambda \Phi_d &= \mathbf{V}\mathbf{R}_f \mathbf{V}^H \Phi_d \\ \Rightarrow \lambda \mathbf{V}\mathbf{c}_d &= \mathbf{V}\mathbf{R}_f \mathbf{V}^H \mathbf{V}\mathbf{c}_d \\ \Rightarrow \mathbf{V} [\lambda \mathbf{I} - \mathbf{R}_f \mathbf{V}^H \mathbf{V}] \mathbf{c}_d &= \mathbf{0}. \end{aligned}$$

The above condition is met by the D solutions, $\lambda_1^s \geq \lambda_2^s \geq \dots \geq \lambda_D^s$ of

$$\det(\lambda \mathbf{I} - \mathbf{R}_f \mathbf{V}^H \mathbf{V}) = 0.$$

That is, we perform the EVD of $\mathbf{R}_f \mathbf{V}^H \mathbf{V}$. The corresponding eigenvectors, \mathbf{c}_d are used to obtain the orthogonal basis, $\{\Phi_d\}$ for $d = 1, 2, \dots, D$. We can now form the $N \times D$ matrix,

$$\mathbf{U}_S := [\Phi_1 \quad \Phi_2 \quad \dots \quad \Phi_D]. \quad (8)$$

The range space of \mathbf{U}_S is referred to as the **signal subspace**. On the other hand, referring to (6), we notice that the noise component spans the entire N -dimensional space. Consequently, we need $(N - D)$ additional orthogonal vectors to represent it. Any arbitrary set of vectors

²For a uniform linear array, assuming the locations of the antenna elements to be:

$$p_{x_n} = 0, \quad p_{y_n} = 0, \quad p_{z_n} = \left(n - \frac{N-1}{2}\right) d, \quad n = 0, 1, \dots, N-1,$$

we can show that

$$\mathbf{v}(k_z) = \left[e^{j\left(\frac{N-1}{2}\right)k_z d} \quad e^{j\left(\frac{N-3}{2}\right)k_z d} \quad \dots \quad e^{-j\left(\frac{N-1}{2}\right)k_z d} \right]^T,$$

where, $k_z = -\frac{2\pi}{\lambda} \cos(\theta)$. It is helpful to also define, $\psi = -k_z d$.

that are orthogonal to each other and orthogonal to \mathbf{U}_S meet the requirement. We now form the $N \times (N - D)$ matrix,

$$\mathbf{U}_N := [\Phi_{D+1} \quad \Phi_{D+2} \quad \dots \quad \Phi_N]. \quad (9)$$

The range space of \mathbf{U}_N is referred to as the **noise subspace**. In summary,

$$\mathbf{C}_x = \sum_{n=1}^N \lambda_n \Phi_n \Phi_n^H, \quad (10)$$

where,

$$\lambda_n = \begin{cases} \lambda_n^s + \sigma_w^2, & \text{if } n = 1, 2, \dots, D \\ \sigma_w^2, & \text{if } n = D + 1, D + 2, \dots, N. \end{cases} \quad (11)$$

Alternatively, by defining the $D \times D$ diagonal matrix, $\Lambda_S := \text{diag}[\lambda_1, \lambda_2, \dots, \lambda_D]$ and the $(N - D) \times (N - D)$ diagonal matrix, $\Lambda_N = \sigma_w^2 \mathbf{I}$, we can write,

$$\mathbf{C}_x = \mathbf{U}_S \Lambda_S \mathbf{U}_S^H + \mathbf{U}_N \Lambda_N \mathbf{U}_N^H. \quad (12)$$

This idea of decoupling the signal subspace from its orthogonal noise subspace is fundamental to many array processing methods.

MUSIC and Root-MUSIC Algorithms

The discussion we have seen so far is the basis for several subspace-based methods. One of the earliest algorithms within the family of subspace methods for DoA estimation is Multiple Signal Classification (MUSIC). MUSIC requires that the following assumptions are satisfied: (1) the received waveform is narrowband, (2) the received waveform consists of D plane-wave signals plus *uncorrelated* noise, (3) the *number* of plane-wave signals, D is known (4) the number of antenna elements, N is at least equal to $D + 1$. A summary of MUSIC for linear arrays is as follows:

Algorithm MUSIC

- 1: Construct the *sample* correlation matrix, \mathbf{C}_x .
 - 2: Determine \mathbf{U}_N , the noise subspace from \mathbf{C}_x using Eigen-Value Decomposition (EVD).
 - 3: **for** $0 \leq \theta \leq \pi$ (alternatively, $-\frac{2\pi d}{\lambda} \leq \psi \leq \frac{2\pi d}{\lambda}$) **do**
 - 4: Generate the corresponding array manifold vector, $\mathbf{v}(\psi)$.
 - 5: Compute the *null-spectrum*,

$$Q(\psi) := \|\mathbf{v}^H(\psi) \mathbf{U}_N\|^2 = \mathbf{v}^H(\psi) \mathbf{U}_N \mathbf{U}_N^H \mathbf{v}(\psi).$$
 - 6: Choose the D minima of $Q(\psi)$. The corresponding values of ψ are the D angles of arrival.
-

It can be noticed from line 5 of the above algorithm that whenever ψ equals the true angle of arrival, $Q(\psi)$ equals zero since $\mathbf{v}(\psi)$ will be a basis vector of the signal subspace and is therefore, orthogonal to the noise subspace. This is the underlying principle behind MUSIC algorithm. For visualization purposes, it is a common practice to plot $1/Q(\psi)$, termed *pseudo-spectrum*.

Root MUSIC is a straight-forward variant of MUSIC algorithm which involves finding the roots of a polynomial instead of plotting the pseudospectrum across all values of the polar angle, θ and searching for the peaks. Consider a Uniform Linear Array whose array manifold vector is written as,

$$\mathbf{v}(\psi) = e^{-j(\frac{N-1}{2})\psi} \begin{bmatrix} 1 & e^{j\psi} & \dots & e^{j(N-1)\psi} \end{bmatrix}^T. \quad (13)$$

An equivalent phase-shifted representation is,

$$\mathbf{v}(z) = [1 \quad z \quad \dots \quad z^{N-1}]^T. \quad (14)$$

It is easy to verify that, $\mathbf{v}(z)|_{z=e^{j\psi}} = e^{j(\frac{N-1}{2}\psi)} \mathbf{v}(\psi)$. Now, a polynomial representation of the null-spectrum is,

$$\begin{aligned} Q(z) &= \mathbf{v}^T(1/z) \mathbf{U}_N \mathbf{U}_N^H \mathbf{v}(z) \\ &\stackrel{(a)}{=} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} z^{-m} \mathbf{U}_N^2(m, n) z^n \\ &\stackrel{(b)}{=} \sum_{-N+1}^{N-1} u_l z^l. \end{aligned} \quad (15)$$

In (a), we defined the ‘‘square’’ of \mathbf{U}_N as $\mathbf{U}_N^2 := \mathbf{U}_N \mathbf{U}_N^H$. In (b), for $l > 0$, u_l is the sum of elements along the l th super-diagonal and for $l < 0$, u_l is the sum of elements along the l th sub-diagonal. Clearly, \mathbf{U}_N^2 is a Hermitian non-negative definite matrix and the relation, $u_l = u_{-l}^*$ holds true. Due to this result, we need to compute only $N - 1$ polynomial coefficients. Notice that plotting the null-spectrum, $Q(\psi)$ and choosing D minima is equivalent to finding D out of $2(N - 1)$ roots of $Q(z)$ that lie on the unit circle. In practice, due to the presence of noise, the roots will not necessarily be on the unit circle. So, we choose the D roots that are inside the unit circle and closest to the unit circle. We can now summarize Root-MUSIC as follows:

Algorithm Root MUSIC

- 1: Construct the *sample* correlation matrix, \mathbf{C}_x .
 - 2: Determine \mathbf{U}_N^2 , the ‘‘square’’ of noise subspace from \mathbf{C}_x using Eigen-Value Decomposition (EVD).
 - 3: Determine the roots of the polynomial, $Q(z) = \sum_{-N+1}^{N-1} u_l z^l$. For $l > 0$, u_l is the sum of elements of \mathbf{U}_N^2 along the l th super-diagonal and for $l < 0$, u_l is the sum of elements of \mathbf{U}_N^2 along the l th sub-diagonal.
 - 4: Choose D roots, $\{z_d\}$ for $d = 1, 2, \dots, D$ that are inside the unit circle and closest to the unit circle.
 - 5: Obtain $\{\theta_d\}$ for $d = 1, 2, \dots, D$ using the relation,
$$\theta_d = \text{acos}(\arg(z_d) \times \lambda / (2\pi d)).$$
-

It must be noted that unlike MUSIC algorithm, Root-MUSIC is not generalizable for arbitrary array geometries. Consequently, the current implementation of Root-MUSIC in `gr-doa` supports linear array geometry only.

Simulation Examples

A GNU Octave based simulation testbench is available in the `@doa_testbench` directory under `gr-doa/examples` that implements MUSIC algorithm for DoA estimation. To list the arguments needed to create the simulation testbench, open Octave and at the command-prompt, type `help doa_testbench_create`. For a demonstration of DoA estimation using MUSIC algorithm for linear arrays, run the script, `estimate_doa_MUSIC_linear_array.m`. A Pseudo-Spectrum plot is shown in Figure 6 for a simulation setting involving targets present at angles, 30° and 125° . Simulation flowgraphs that utilize blocks developed using GNU Radio framework for implementing MUSIC are available in `gr-doa/apps`.

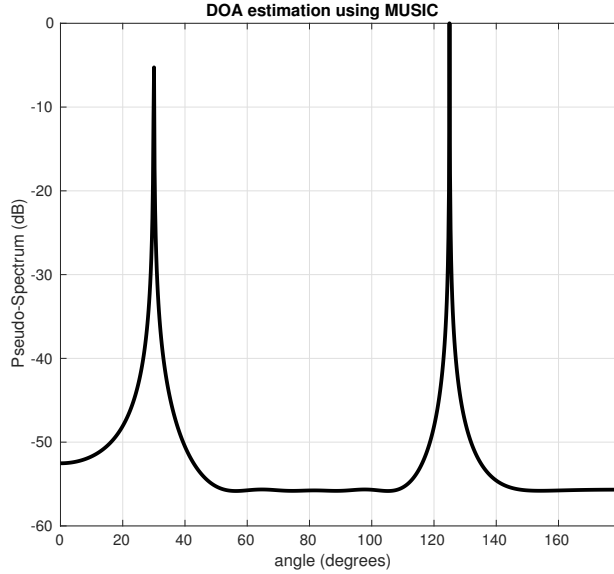


Figure 6: Pseudo-Spectrum of a Simulated 2-target DoA Estimation Scenario.

The GNU Octave based simulation testbench, `doa_testbench` also contains an implementation of Root-MUSIC algorithm for DoA estimation. To run a simulation example that demonstrates DoA estimation using Root-MUSIC algorithm for linear arrays, run the script, `estimate_doa_RootMUSIC_linear_array.m`. Since Root-MUSIC algorithm generates roots of a polynomial, it does not have an associated visual representation. However, by running the simulation flowgraph, `gr-doa/apps/run_RootMUSIC_lin_array_simulation.grc`, the resultant polynomial roots can be seen on a compass display.

Physical Experiments

We now discuss the methodology involved in performing DoA estimation with the blocks developed using GNU Radio framework. In addition to the list of items indicated previously, we also require:

- 4 monopole antennas (or other antennas of your choice which have a good gain-phase relationship in your frequency band of choice).
- (Optional) An array fixture with variable antenna element spacing and 4 equal length SMA-M to SMA-M cables to connect the antennas held in place by the array fixture to the SDR board.

NOTE: It is strongly recommended that the antenna element separation is half-wavelength or lower depending on the tone center-frequency. Arranging antenna elements at larger distances leads to an aliasing effect and compromises the angle resolution capability of the DoA algorithms.

Position the transmitter at a distance in the far-field from the receiver antenna array. Note the geometrical angle that is to be expected. At this point, we assume that the user has performed the relative phase offset measurement and correction stage as outlined previously. In the host PC that is connected to the transmitter, navigate to the `gr-doa/apps` directory and open the flowgraph titled, `run_DoA_transmitter.grc`. The user-defined variables are contained in the `struct` variable titled, `input_variables`. Here, select appropriate values for parameters

such as tone frequency, sample rate, center frequency, ip-address of the device and transmit gain based on your transmitter.

Now, assuming that a linear antenna array is being used for DoA estimation, navigate to the `gr-doa/apps` directory in the host PC that is connected to the receiver and open the flowgraph titled, `run_MUSIC_lin_array_FMComms5.grc`. Again, select appropriate values for the ip-address of the receiver, receive gain, number of array elements, normalized spacing of the antenna elements etc. Ensure that parameters such as tone frequency, sample rate and center frequency are the same as those selected at the transmitter. Ensure that the correct config file for phase offset correction is selected in the `Phase Correct` block. A screenshot of the this flowgraph is shown in Figure 7. Execute this flowgraph to see DoA estimation performance over a linear array.

We now discuss our DoA estimation results using the flowgraph mentioned above. The experiments were conducted in an anechoic chamber of approximate dimensions, 3 m(L) \times 2 m(W) \times 3 m(H). We conducted this experiment for a range of transmitter positions with respect to the receiver. In Figure 8, we show the performance of MUSIC in terms of the Pseudo-Spectrum for four directions of arrival. Despite the fact that we conducted measurements in a reasonably controlled environment, as shown in this figure, we occasionally observed strong secondary peaks which were nevertheless weaker compared to the peak corresponding to the true direction of arrival. In these experiments, the spacing between the antenna elements was 1 in and the center frequency was selected such that this spacing corresponded to half-wavelength of the carrier. The sample rate chosen was 1 MS/s.

Effect of Antenna Calibration on DoA Estimation

Based on Figure 8, we noticed that the SNR of the pseudo-spectrum is rather low despite conducting experiments in a reasonably ideal environment. While relative phase offset correction across the receive streams is one aspect of the overall calibration process, we demonstrate in this subsection that calibrating the antenna element gains and phases also has a noticeable effect. To do so, we provide a brief overview of the approach adopted based on [4].

Suppose that $\Gamma := \text{diag}\{1, \alpha_1 e^{-j\theta_1}, \dots, \alpha_{N-1} e^{-j\theta_{N-1}}\}$ represents the diagonal matrix consisting of the non-uniform antenna gains and phases along its principal diagonal. It can be

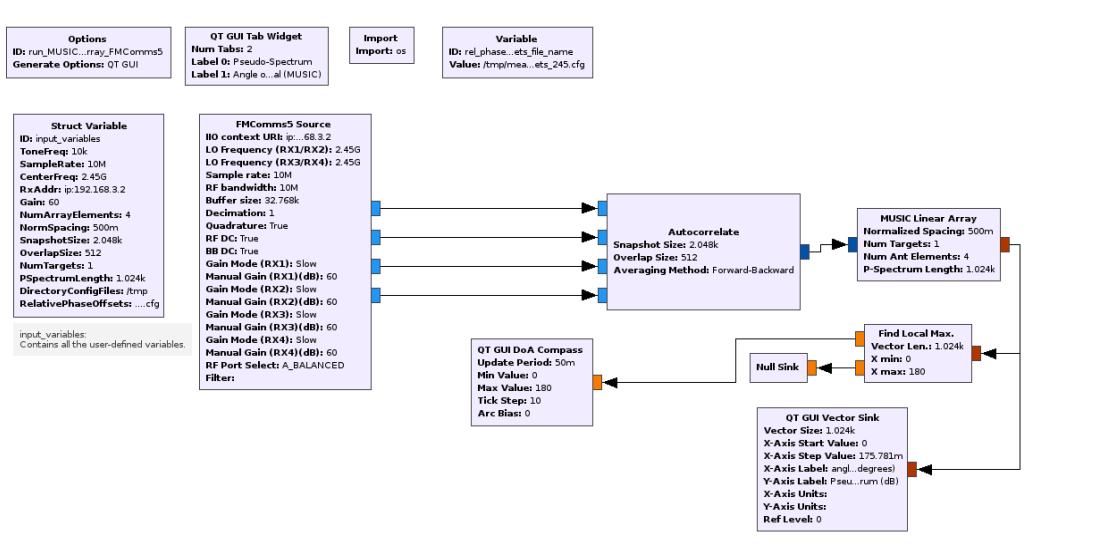


Figure 7: MUSIC DoA Estimation Flowgraph for Linear Antenna Arrays.

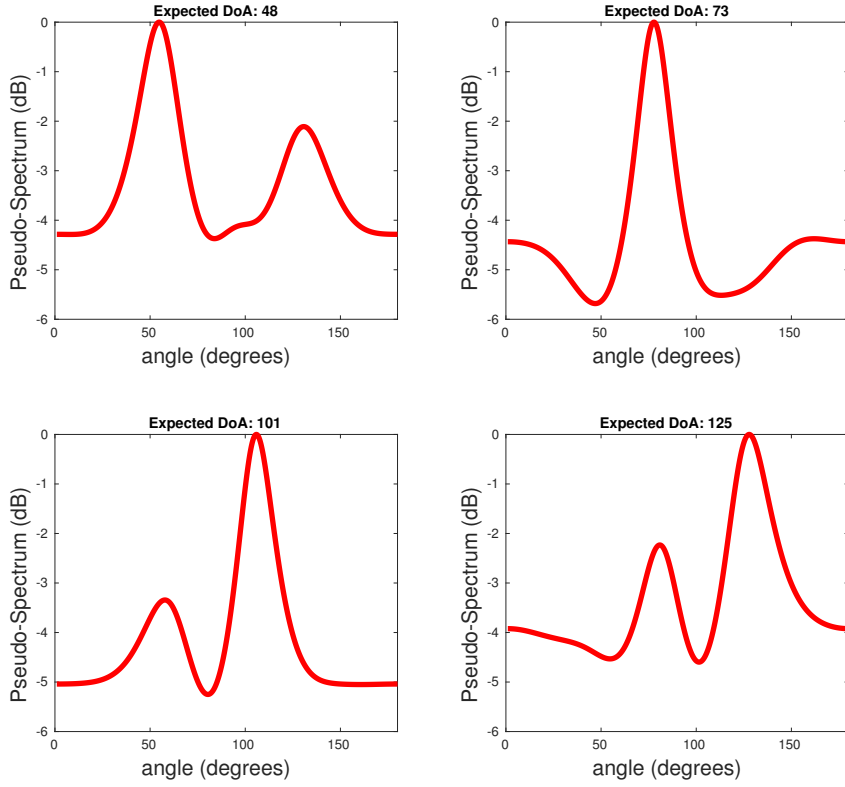


Figure 8: Pseudo-Spectrum of a 1-target DoA Estimation Scenario using a Linear Array. Experiments Conducted in an Anechoic Chamber.

showed that this effect can be captured by updating (6) as shown below:

$$\mathbf{C}_x = \Gamma \mathbf{V} \mathbf{R}_f \mathbf{V}^H \Gamma^H + \sigma_w^2 \mathbf{I}. \quad (16)$$

Now, taking the EVD of \mathbf{C}_x gives,

$$\mathbf{C}_x = \mathbf{E}_S \Lambda_S \mathbf{E}_S^H + \sigma_w^2 \mathbf{E}_N \mathbf{E}_N^H. \quad (17)$$

where, \mathbf{E}_S denotes the eigenvector matrix of \mathbf{C}_x associated with the signal subspace spanned by $\Gamma \mathbf{V}$. Thus, the solution set of sensor gains and phases and directions of arrival are constrained such that

$$\mathbf{E}_S \mathbf{E}_S^H \Gamma \mathbf{V} = \Gamma \mathbf{V}.$$

Equivalently,

$$\mathbf{E}_S \mathbf{E}_S^H \Gamma \mathbf{v}(\mathbf{k}_d) = \Gamma \mathbf{v}(\mathbf{k}_d), \quad d = 1, \dots, D.$$

By defining $\mathbf{V}_d = \text{diag}(\mathbf{v}(\mathbf{k}_d))$ and $\gamma = [1, \alpha_1 e^{-j\theta_1}, \dots, \alpha_{N-1} e^{-j\theta_{N-1}}]^T$, the above equation becomes,

$$\mathbf{E}_S \mathbf{E}_S^H \mathbf{V}_d \gamma = \gamma \mathbf{V}_d \quad d = 1, \dots, D.$$

Since \mathbf{V}_d is a diagonal matrix composed entirely of the array-manifold vector, it is also unitary. Hence,

$$\mathbf{V}_d^H \mathbf{E}_S \mathbf{E}_S^H \mathbf{V}_d \gamma = \gamma \quad d = 1, \dots, D.$$

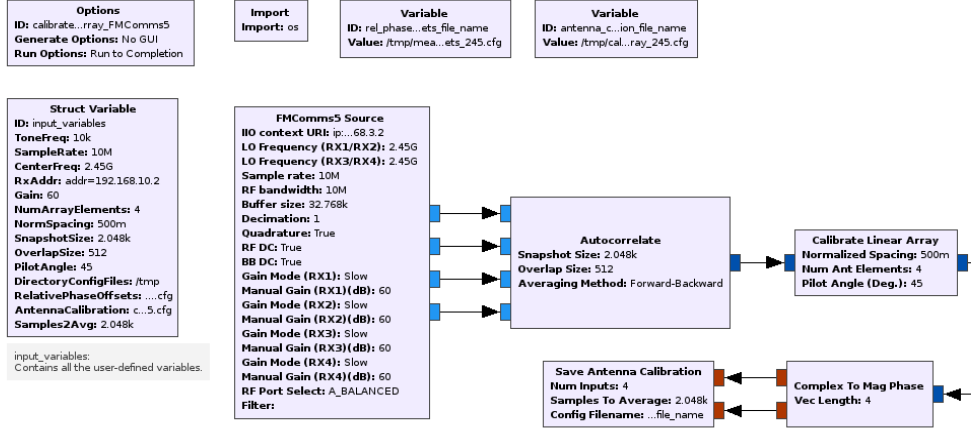


Figure 9: Flowgraph for Antenna Calibration of a Linear Array.

Therefore, if we perform EVD of the matrix, $\mathbf{V}_d^H \mathbf{E}_S \mathbf{E}_S^H \mathbf{V}_d$, the eigenvector that corresponds to a value of unity is the estimated antenna coefficient vector. Notice that we require accurate knowledge of \mathbf{V}_d with respect to D pilot targets with known DoA. A standard practice is to position one pilot transmitter and calibrate for antenna coefficients with respect to its DoA.

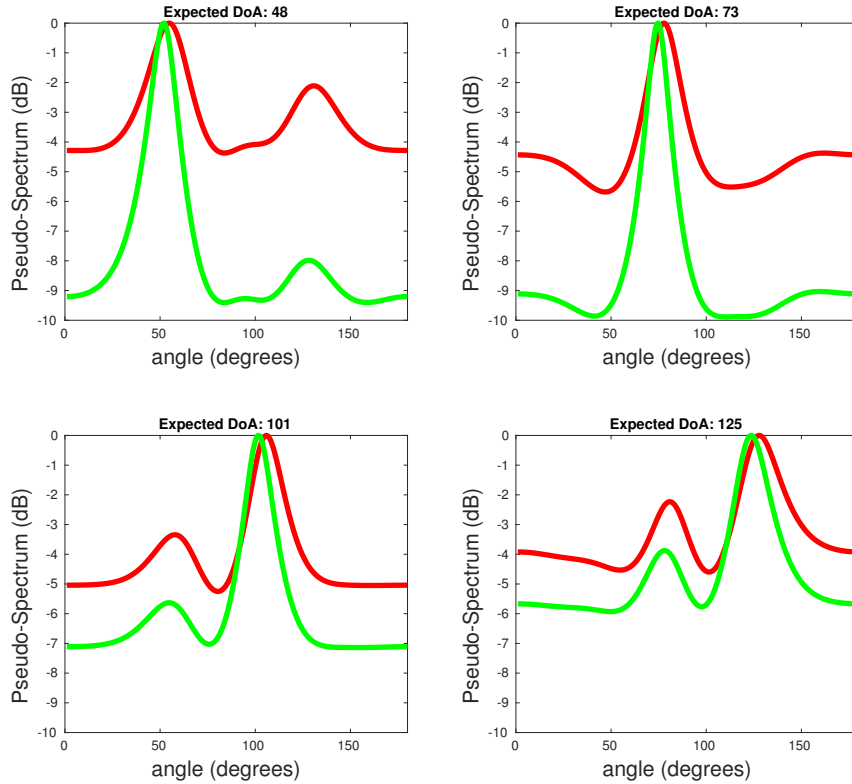


Figure 10: Pseudo-Spectrum of a 1-target DoA Estimation Scenario using a Calibrated Linear Array. Experiments Conducted in an Anechoic Chamber.

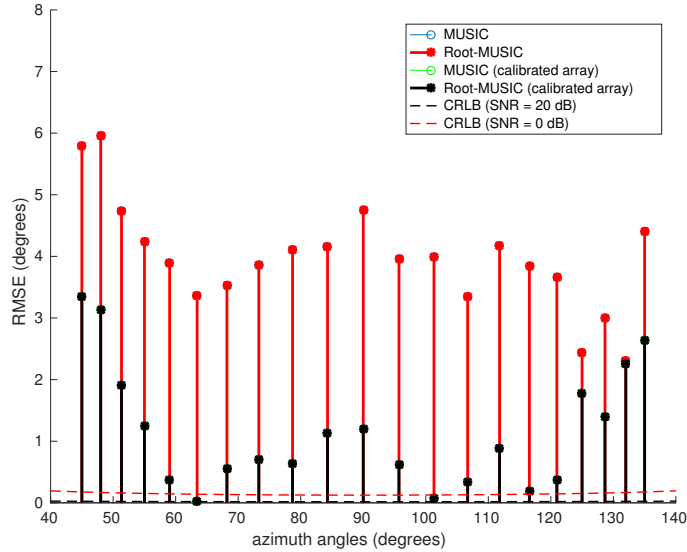


Figure 11: RMSE of MUSIC DoA Estimation using a Linear Array. Experiments Conducted in an Anechoic Chamber.

To observe the effect of antenna calibration, first navigate to the `gr-doa/apps` directory and open the flowgraph titled, `calibrate_lin_array_FMComms5.grc`. Select appropriate values for the parameters in `input_variables`. A screenshot of the this flowgraph is shown in Figure 9. Position the transmitter at a known angle and execute this flowgraph to calibrate the antenna array. Then, execute the flowgraph titled, `run_MUSIC_calib_lin_array_FMComms5.grc` to observe the improvement in the DoA estimated by MUSIC algorithm. Figure 10 provides some example results of this implementation, we noticed not only an improvement in the SNR of the pseudo-spectrum and a general sharpening of the peak around the true DoA but also an apparent shift of the peak towards the true DoA. This overall improvement in the estimation performance is further illustrated by Figure 11. The dashed lines in this figure represent the Cramer-Rao Lower Bound as reported in [5].

Benchmark Tests

In order to determine the fastest rate at which data can be streamed from an X310 to a host PC to run our DoA implementation, we first conducted a preliminary simulation-based study to understand the effect of *Snapshot Size* parameter on estimation performance. As shown in Figure 12, we noticed that the root-mean-squared error (RMSE) progressively decays with increasing snapshot size. As a consequence, we were encouraged to use a sufficiently large value for this parameter such that not only does it help in achieving a reasonable amount of spatial averaging effect when computing the *sample* autocorrelation matrix, but also reduces the RMSE in DoA estimation.

The fundamental reason for focusing on the `Autocorrelate` block and using its performance as an indicator of the fastest overall rate is due to its computationally-intensive nature. Furthermore, all other operations are performed downstream from the perspective of this *rate-conversion* block except for the `Phase Correct` block. Hence, this block acts as a rate-limiting block for all practical purposes. In other words, while it can be argued that one of the downstream blocks, such as `MUSIC`, might operate at a slower rate than `Autocorrelate` for smaller snapshot sizes and consequently, when more items are made available in the output buffer of

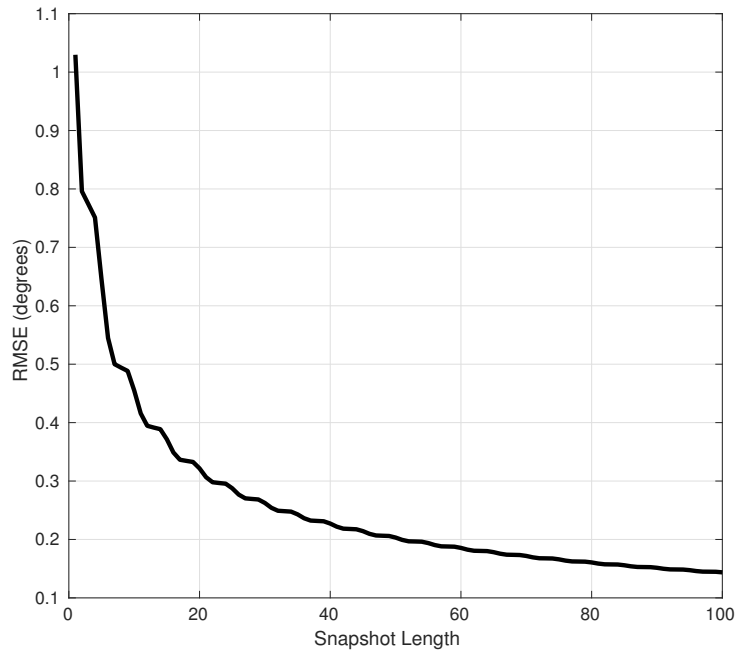


Figure 12: RMSE of MUSIC DoA Estimation as a Function of Snapshot Length.

`Autocorrelate` block, smaller snapshot sizes defeat the purpose of accurate DoA estimation as demonstrated by Figure 12.

In our experiments, we used both a laptop and a desktop computer acting as the host PC for DoA estimation. The laptop consisted of 8-cores running at a clock frequency of 1.87 GHz whereas the desktop machine is a 48-core computer with a clock frequency of 2.6 GHz. With the laptop connected to the SDR over a 1 Gigabit Ethernet interface, we were able to operate at the fastest possible rate, *i.e.*, 6.25 MS/s per stream without buffer overflow except for snapshot sizes of 32 and below. With the desktop machine acting as the host PC and with a 1 Gigabit Ethernet link as the interface, we achieved the fastest possible rate, *i.e.*, 6.25 MS/s per stream, for smaller snapshot sizes also.

Conclusion

This article discusses the phase synchronization feature available by using the FMComms5 evaluation board. Using a simple calibration procedure, we showed a way to measure and save the fixed repeatable relative phase offsets. Any user-developed software running on the host PC that requires accurate phase synchronization will need to incorporate a phase offset correction step, as we demonstrated in the case of MUSIC-based DoA estimation. The physical experimental results we provided illustrate the performance observed using our lab setup.

Links

- libiio: <https://github.com/analogdevicesinc/libiio>
- libad9361-iio: <https://github.com/analogdevicesinc/libad9361-iio>

- gr-doa: <https://github.com/tfcollins/gr-doa>

References

- [1] Srikanth Pagadarai, Travis Collins, Neel Pandeya *Phase Synchronization Capability of TwinRX Daughterboards and DoA Estimation*. Available from World Wide Web: (https://github.com/tfcollins/gr-doa/blob/master/docs/whitepaper/doa_whitepaper.pdf).
- [2] Analog Devices Inc. *AD-FMCOMMS5-EBZ Hardware*. Available from World Wide Web: (<https://wiki.analog.com/resources/eval/user-guides/ad-fmcomms5-ebz/hardware>).
- [3] Harry L. Van Trees, "Optimum Array Processing: Part IV of Detection, Estimation, and Modulation Theory", Wiley-Interscience, New York, 2002.
- [4] V. C. Soon, L. Tong, Y. F. Huang and R. Liu, "A Subspace Method for Estimating Sensor Gains and Phases," in IEEE Transactions on Signal Processing, vol. 42, no. 4, pp. 973-976, Apr 1994.
- [5] P. Stoica and A. Nehorai, "MUSIC, Maximum Likelihood, and Cramer-Rao Bound," in IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 37, no. 5, pp. 720-741, May 1989.