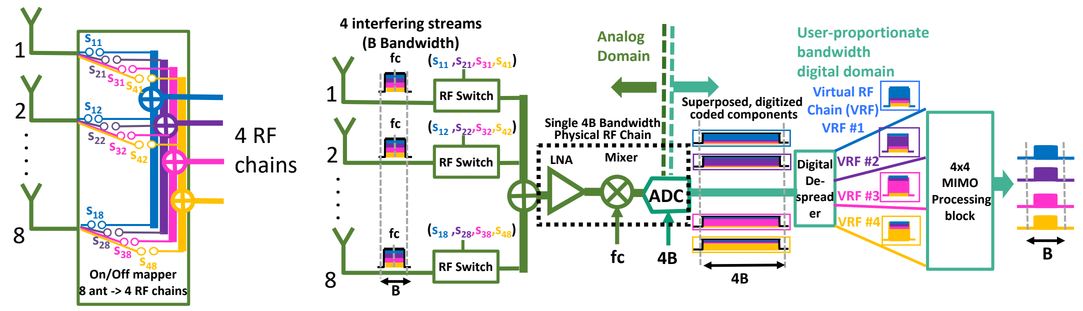
GreenMO Math Explanation

1 GreenMO equivalence to switched-Hybrid Beamformer

An equivalent system to GreenMO, however implemented with virtual RF chains is that of a fully connected, switched hybrid beamformer (Fig. 1 a). By that, we mean, an independently controlled $s_{rm} \in \{0, 1\}$ switch connected to each m-th antenna, each r-th RF chain. We will first show how we model this equivalent system so that the notation is clear with this known example, and then show how we adapt this for GreenMO albeit with single wider-bandwidth RF chain.



(a) Equivalent switched-hybrid array

(b) GreenMO with RF switches and 4 times bandwidth

Figure 1: (a) Shows an equivalent Hybrid Beamformer with physical RF chains, s_{rm} denoted if antenna m is on for r-th RF chain (b) Shows GreenMO architecture which implements similar system albeit with one RF chain by using wider bandwidth

We have $i \in \{0, 1, \dots M-1\}$ M antennas, $j \in \{0, 1, \dots R-1\}$ RF Chains, to connect $k \in 0, 1, \dots K-1$ K users, each having a single antenna each. Hence in the combined network, the user's see a $M * K$ wireless channel, with channel between k-th user, m-th antenna given by $\mathbf{H}(t) = [h_{mk}(t)]$. Each user's signal is represented by $x_k(t)$. The analog signal received at m-th antenna, $y_m(t)$ can be written as ($*$ represents convolution):

$$y_m(t) = \sum_{k=1}^{K} (x_k(t) * h_{mk}(t)) \tag{1}$$

and hence, the analog signal at r-th RF chain can be written as summation of all antenna signals y_m multiplied by on-off variable s_{rm}:

$$y_r(t) = \sum_{m=1}^{M} y_m(t) s_{rm} = \sum_{m=1}^{M} s_{rm} \sum_{k=1}^{K} (x_k(t) * h_{mk}(t)) = \sum_{m=1}^{M} \sum_{k=1}^{K} x_k(t) * (s_{rm} h_{mk}(t)) \tag{2}$$

The last step holds since s_{rm} doesn't depend on k. After digitization at $t = NT_s$, we change $(t) \rightarrow [n]$, and we get

$$y_r(nT_s) \doteq y_r[n] = \sum_{m=1}^{M}\sum_{k=1}^{K} x_k[n] * (s_{rm}h_{mk}[n]) \tag{3}$$

In frequency domain, we would get, assuming

$$Y_r[f] = \sum_{m=1}^{M}\sum_{k=1}^{K} X_k[f](S_{rm}H_{mk}[f]) = \sum_{m=1}^{M}\sum_{k=1}^{K}(S_{rm}H_{mk}[f])X_k[f] \tag{4}$$

Which in matrix form, by collecting all Y_r across R RF chains, becomes:

$$\mathbf{Y}[f] = \mathbf{SH}[f]X[f] \tag{5}$$

where $\mathbf{S} = [s_{rm}]$ represents a $R \times M$ binary matrix, that maps the $M \times K$ wireless channel matrix to an equivalent $R \times K$ matrix, that gets digitized per-user by the R RF chains. To get back the user's transmitted symbols, $X[f]$ from these R RF chains, the processing can pseudo-invert $\mathbf{SH}$, i.e. one can get an estimate of $X[f]$, denoted as $\hat{X}[f]$,

$$\hat{X}[f] = (\mathbf{SH[f]})^{\dagger}Y[f] \tag{6}$$

2 How does GreenMO implement this with a single RF chain?

Now that we presented how the GreenMO equivalent system looks like with R physically different RF chains, we will go over how GreenMO creates the same system model but with R virtual RF chains spawned out from a single $R\times$ bandwidth RF chain.

First, observe that till now s_{rm}, that denotes switch value for r-th RF chain and m-th antenna did not have a time-dependence in the previous analysis. This is typical with Hybrid beamforming literature, since the analog beamforming weights (0/1 here in case of s_{rm}, or phase weights ϕ_{rm} in case of phased array) are typically treated as static. GreenMO uses a single physical RF chain 1, and hence, we get with some time-variability switching represented by $s_{1m}(t)$:

$$y^1(t) = \sum_{m=1}^{M}\sum_{k=1}^{K} x_k(t) * (s_{1m}(t)h_{mk}(t)) \tag{7}$$

We sample this single RF chain with $n\frac{T_s}{R}$, since we have $R\times$ bandwidth at disposal with the single RF chain:

$$y^1[n] = y^1(n\frac{T_s}{R}) = \sum_{m=1}^{M}\sum_{k=1}^{K} x_k(n\frac{T_s}{R}) * (s_{1m}(n\frac{T_s}{R})h_{mk}(n\frac{T_s}{R})) \tag{8}$$

Now, let's define $y_r[i] = y^1[Ri{+}r]$, i.e. $y_0[n]$ is a collection of $y^1[0], y^1[R], y^1[2R], \ldots$, $y^1[n]$ is a collection of $y^1[1], y^1[R+1], y^1[2R+1], \ldots$ and so on. Thus, we have,

$$y_r[i] := y^1((Ri+r)\frac{T_s}{R}) = \sum_{m=1}^{M}\sum_{k=1}^{K} x_k(iT_s + \frac{rT_s}{R}) * (s_{1m}(iT_s + \frac{rT_s}{R})h_{mk}(iT_s + \frac{rT_s}{R})) \tag{9}$$

Here, we make one assumption, that the channel $h_{mk}(iT_s + \frac{rT_s}{R})$ does not depend on r, but changes only with i (with T_s rate, and not $\frac{T_s}{R}$ rate). Hence, $h_{mk}(iT_s) \approx h_{mk}(iT_s + \frac{T_s}{R}) \approx h_{mk}(iT_s + \frac{2T_s}{R}) \ldots \approx h_{mk}(iT_s + \frac{(R-1)T_s}{R})$. Thus, we can re-write, $h_{mk}(iT_s + \frac{rT_s}{R}) = h_{mk}(iT_s) = h_{mk}[i]$.

Further, we design codes such that the codes repeat every T_s time slots, that is $s_{1m}(iT_s + \frac{rT_s}{R}) = s_{1m}((i-1)T_s + \frac{rT_s}{R}) = s_{1m}((i+1)T_s + \frac{rT_s}{R}), \forall i$, since the code design is in our hands and we can force this with RF switches. Hence, we can re-write $s_{1m}(iT_s + \frac{rT_s}{R}) = s_{1m}(\frac{rT_s}{R}) := s_{rm}$ where the last step is just to simplify the description, since it only depends on r. As an example, when $R = 4$, one possible code design could be $s_{1m} = 1, 0, 0, 0, 1, 0, 0, 0 \ldots$, $s_{2m} = 0, 1, 0, 0, 0, 1, 0, 0 \ldots$, $s_{3m} = 0, 0, 1, 0, 0, 0, 1, 0 \ldots$, $s_{4m} = 0, 0, 0, 1, 0, 0, 0, 1 \ldots$. Basically, for each s_{rm} code, it should repeat every R samples, since the sampling rate is $R\times$. This code design can also be looked alternately as a frequency domain spreader (since this on-off nature creates harmonics), and the discussion is detailed in our main paper. Hence, we can rewrite the above equation more simply as:

$$y_r[i] = \sum_{m=1}^{M} \sum_{k=1}^{K} x_k(iT_s + \frac{rT_s}{R}) * (s_{rm} h_{mk}[i]) \tag{10}$$

This equation has already started to take shape of a traditional hybrid beamformer, and this becomes more apparent as we take the frequency transform:

$$Y_r[f] = \sum_{m=1}^{M} \sum_{k=1}^{K} \mathcal{F}(x_k(iT_s + \frac{rT_s}{R}))(S_{rm} H_{mk}[f]) \tag{11}$$

Now, $\mathcal{F}(x_k(iT_s + \frac{rT_s}{R}))$ is just $\frac{rT_s}{R}$ delayed form of $x_k(iT_s)$, hence, $\mathcal{F}(x_k(iT_s + \frac{rT_s}{R})) = e^{-j2\pi f \frac{rT_s}{R}} \mathcal{F}(x_k(iT_s)) = e^{-j2\pi f \frac{rT_s}{R}} X[f]$. Hence, the extra delay just acts as a phase-varying exponential, and thus, we get

$$Y_r[f] = \sum_{m=1}^{M} \sum_{k=1}^{K} X_k[f] e^{-j2\pi f \frac{rT_s}{R}} S_{rm} H_{mk}[f] \tag{12}$$

We can absorb the $e^{-j2\pi f \frac{rT_s}{R}} S_{rm} = \tilde{S}_{rm}[f]$, to further simplify:

$$Y_r[f] = \sum_{m=1}^{M} \sum_{k=1}^{K} X_k[f] \tilde{S}_{rm}[f] H_{mk}[f] \tag{13}$$

Which in matrix form, by collecting all Y_r across R 'virtual' RF chains (Remember, to get y_r from y^1 we spliced across the time samples, we did not have R physical RF chains), becomes:

$$\mathbf{Y}[f] = \tilde{\mathbf{S}}[f] \mathbf{H}[f] \mathbf{X}[f] \tag{14}$$

Hence, now to get back $X[f]$, we can pseudo invert $\tilde{\mathbf{S}}[f]\mathbf{H}[f]$ per subcarrier.

The pseudo-inversion performance does depend on choice of the matrix $\tilde{\mathbf{S}}[f]$. Empirically, we have found choice of $\tilde{\mathbf{S}}[f]$ that does binarized analog beamforming per-user and makes the equivalent channel $\tilde{\mathbf{S}}[f]\mathbf{H}[f]$ diagonal heavy gives the best estimate. More details on this can be found in our main paper.

3 Hardware implications

The text here explains the theory behind how GreenMO creates the virtual RF chains, by using wider bandwidth RF chains and periodically repeating codes. The key implications on hardware are:

- To implement this in hardware, the switches need to toggle with 0-1 codes that repeat every $T = 1/B$ to support bandwidth B, hence have a fundamental frequency of B. Further the duty cycle needs to be $1/R$ to support total R virtual chains. Hence, the rise/fall time of switched need to be $<< 1/(RB)$. For example, to support 4 users with $B = 10$ MHz each, the rise time for the switches we used in hardware was 10 nano seconds, less than $1/(40 * 10^6) = 25$ns. To support say, 10 users with 100 MHz bandwidth, we would need faster than 1ns switches. This hardware already exists in form of high-speed diodes/mixers, or very fast analog voltage level changing TTDs which will be used in future versions of GreenMO.
- Next, we need a high sampling rate ADC. Again to support 4 virtual chains with 10 MHz each, we could get away with a standard WARP SDR that has 40 MHz sampling rate. In the next version of GreenMO, we can use the ADI FMCDAQ2 that support 1 GHz ADC/DAC, and we have already started integrating this with RF components.
- Other important hardware are a linear, maybe $\approx 4\times$ RB bandwidth RF chain (so that all the harmonics created by switching are carried across, with high fidelity). This requires a nice PCB design, well simulated in ADS/HFSS and ensure that PCB losses are minimized.