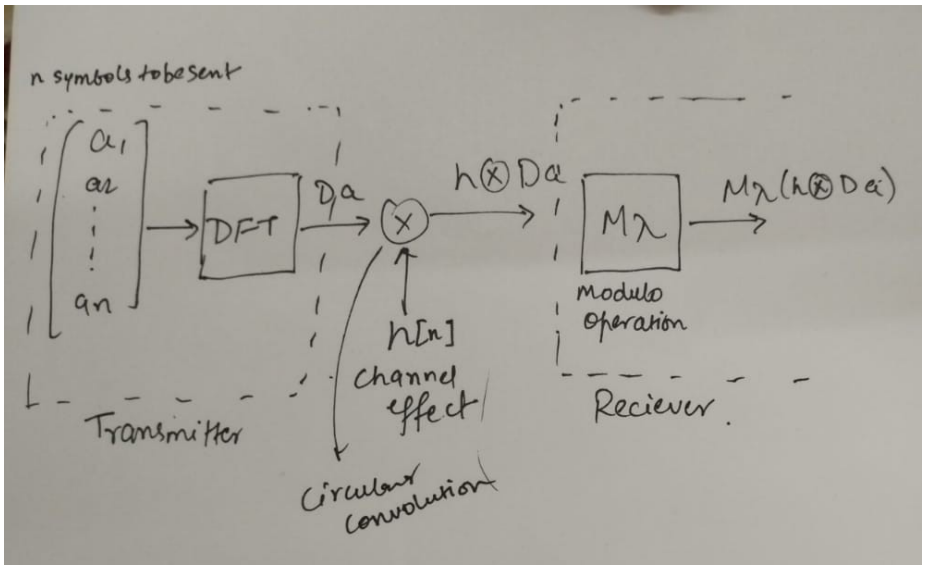

Figure 1: Block diagram

Transmitter **Receiver**

$$
\begin{aligned}
f_\lambda &= M_\lambda\big(h \otimes Da\big) \\
where, M_\lambda(a) &= \big(a+\lambda\big) mod 2\lambda - \lambda \\
f_\lambda &= h \otimes Da + z, \quad \forall\, z \in 2\lambda\mathbb{Z}^N \\
D^{-1} f_\lambda &= D^{-1}\big[h \otimes D\big]a + D^{-1}z \\
y &= D^{-1} f_\lambda = \mathrm{Diag}(Dh)\, a + D^{-1}z \\
y_{n\times 1} &= \mathrm{Diag}(Dh)\, a_{n\times 1} + D^{-1} z_{n\times 1}
\end{aligned}
$$

$$
y_{n\times 1} = \begin{bmatrix} h_1 & 0 & \dots & 0 \\ 0 & h_2 & \dots & 0 \\ \vdots & \ddots & & \vdots \\ 0 & \dots & 0 & h_n \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ 0 \\ a_i \\ \vdots \\ 0 \\ a_n \end{bmatrix} + D^{-1} z_{n\times 1}
$$

M entries of vector a are 0

$$y_{M\times 1} = \big[D\big]\, z_{n\times 1}$$

$$y_{M\times 1} = A z_{n\times 1}$$

(We are exploiting sparsity of a vector corresponding to M rows (zero ones) in vector a .)
why not perform unfolding algorithm?

- in conventional ADC distance between symbols is larger compared to that in modulo folded ADC.If modulo folded symbols are unique then symbol detection is still possible.In presence of noise, folded symbols are expected to have more probabilty of error(due to quantization). From the paper ("Low-Rate Modulo folded ADC for detecting linearly modulated communication symbol") , they have concluded that if conventional ADC and modulo ADC use same number of bits ,under high SNR probability of error for both architecture remains same.
- unfolding in presence of quantization error , introduces more error so detection is more feasible in modulo folded samples.

1 DFT matrix injectivity analyisis

	1	2	3	4	5	6	7	8
1			0	0	0	0	0	0
2	0					0		0
3		0	0	0	0	0	0	0
4					0		0	0
5				0		0	0	0
6		0	0		0	0	0	0
7				0			0	0
8					0	0	0	0

coloumns correspond to vector a elements and rows correspond to M

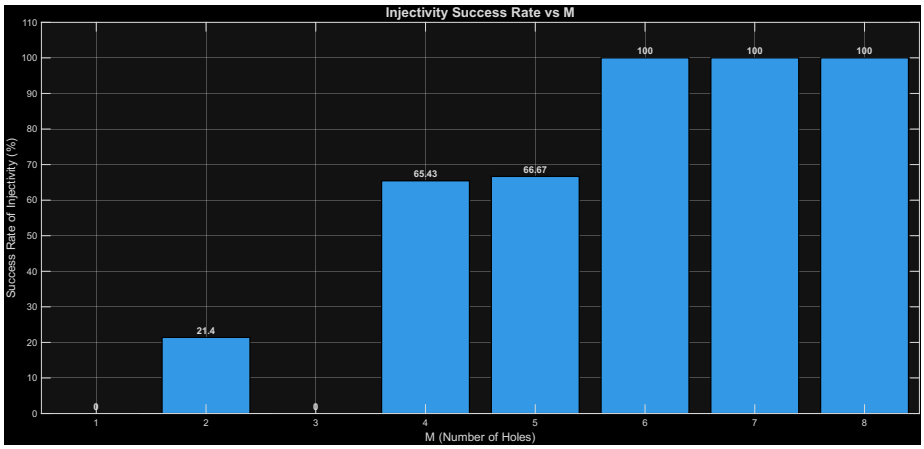

Figure 2: for hole position shown in table

2 inspecting all combinations for particular M

Since we know the hole position ,the exhaustive search space of z for y=Az, correspond to those z's for which vector a has holes at those positions. for M=8 only 1 z vector possible.

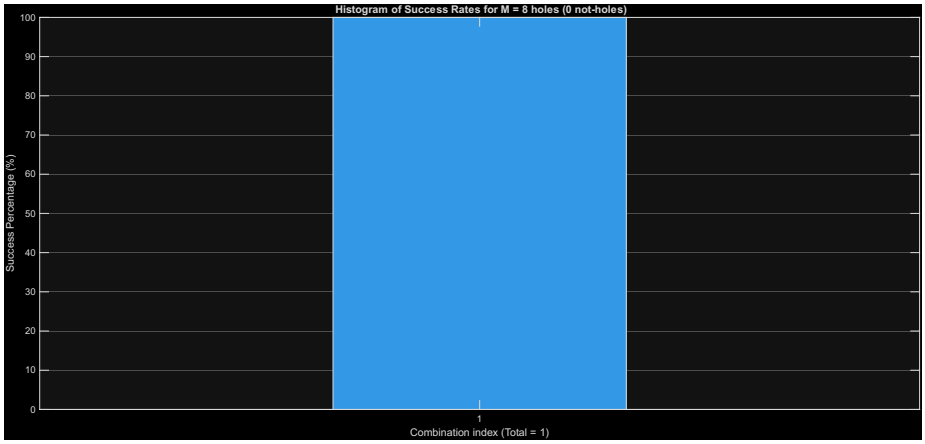

Figure 3: M=8

for M=7 only 8 z vector possible(non-zero vector element at any of 8 positions).

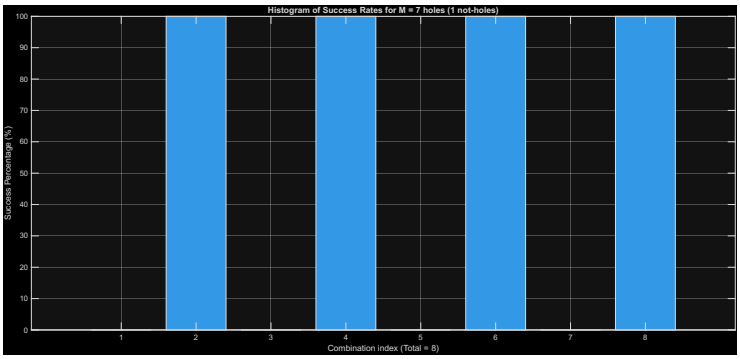

Figure 4: M=7

for M=6 only 28 z vector possible.

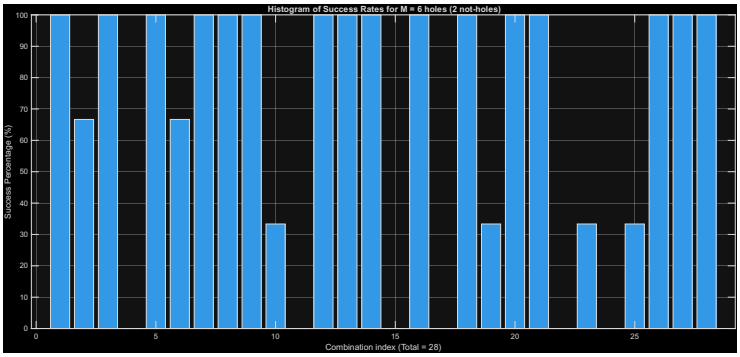

Figure 5: M=6

for M=5 only 54 z vector possible.

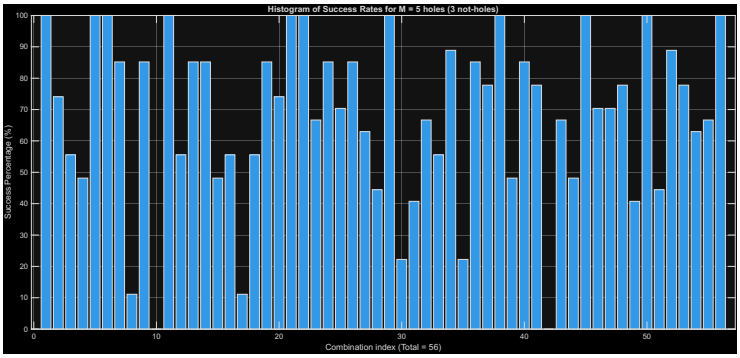

Figure 6: M=5

for M=4 only 70 z vector possible.

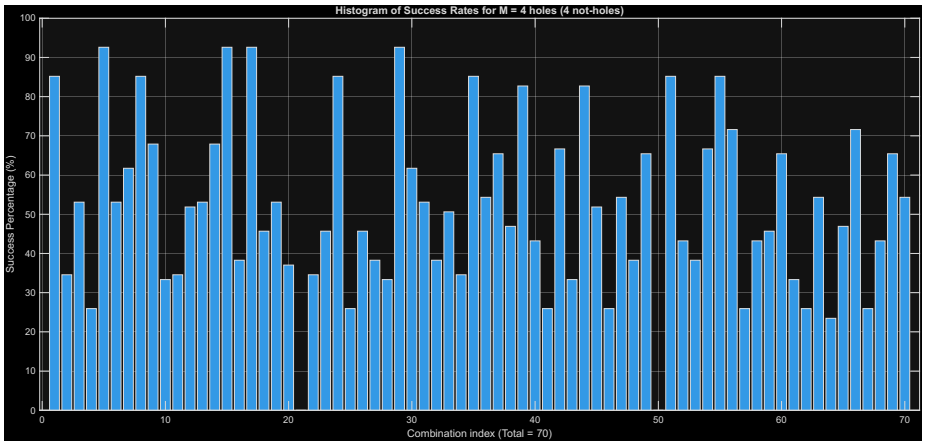

Figure 7: M=4

for M=3 only 54 z vector possible.

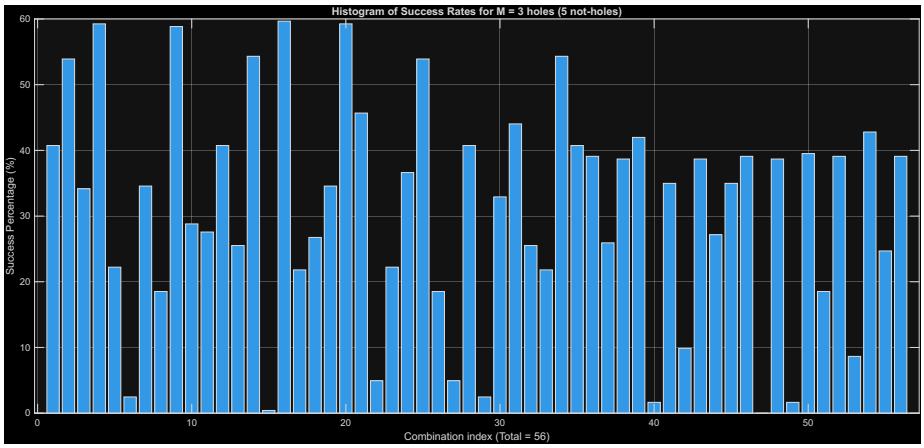

Figure 8: M=3

for M=1 only 8 z vector possible.

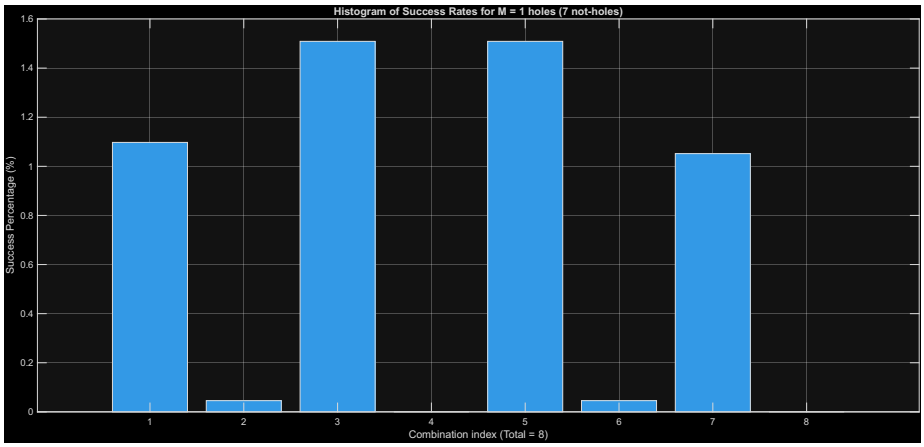

Figure 9: M =1

FOR each M finding mean success rate and variance

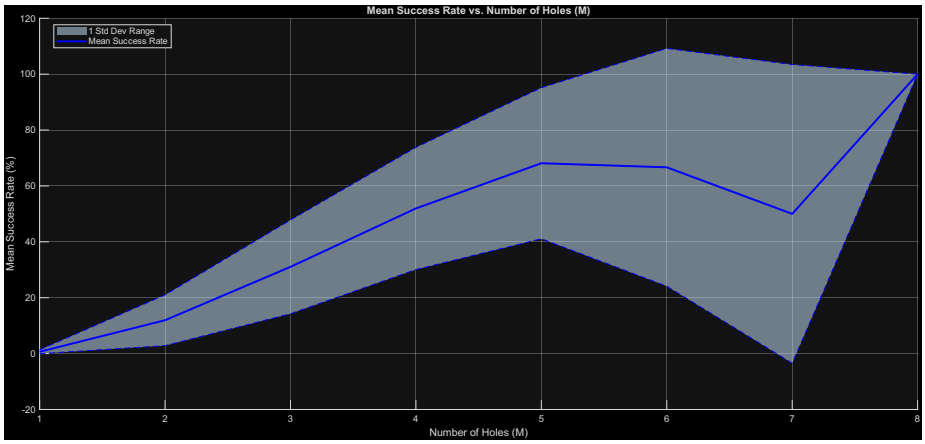

Figure 10: summarised plot

3 minimum distance condition

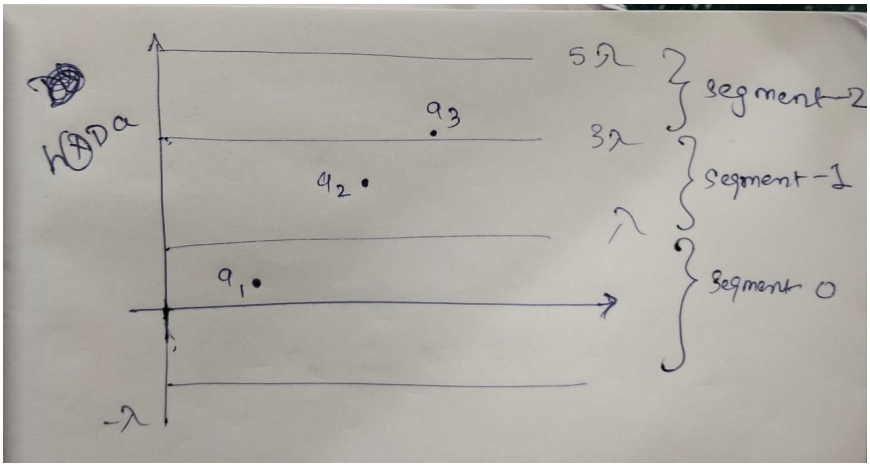

Figure 11: visualisation

For each a_i to have unique z_i corresponding to $hxDa_i$. Every a_i should be in different segment. Equivalently minimum distance between a_i's should be greater than 2λ
$\mathbf{d}_{\min} \geq 2\lambda$

4 Error analyis

The following outlines the structure and logic of the simulation code:

- **Initialization of Variables:**
  - $S = [1, 2, 3]$**: Set of symbol indices.**
  - $a_{\mathbf{true}} = [0; 3; 0; 1; 0; 0; 3; 0]$**: True amplitude vector, length** $N = 8$**.**
  - $N = 8$**: Signal or measurement length.**
  - $\lambda = 1$**: Modulo parameter.**
  - $h = [2, 1, -1, 5, 0, 0, 0, 0]^T$**: Filter or channel vector.**
  - **holes_location** $= [1, 3, 5, 6, 8]$**: Index positions counted as measurement "holes".**
  - $D =$ **dftmtx**$(N)/\sqrt{N}$**: Normalized DFT matrix.**
- **Main Monte Carlo Loop for Each Noise Level (σ):**
  - **For each σ (noise std), repeat for $r = 1$ to** `numRealizations`**:**

1. **Generate clean wrapped measurements:**

$$f_\lambda^{\mathbf{clean}} = \mathbf{generate\_f\_lam}(N, \lambda, a_{\mathbf{true}}, h)$$

2. **Add Gaussian noise:**

$$f_\lambda^{\mathbf{noisy}} = f_\lambda^{\mathbf{clean}} + \sigma \cdot \mathbf{randn}(N, 1)$$

3. **z-based estimate:**
   For each candidate in $Z_{\mathbf{all}}$, compute $y_{\mathbf{est}} = D_h^{\mathbf{reduced}} Z_{\mathbf{all}}(:, i)$, pick the best match by minimizing $\|y_{\mathbf{est}} - y\|^2$.
4. **Classification error:**
   For each method, compute 0-1 vector misclassification rate versus the true $a_{\mathbf{true}}$,N=8(length of vector):

$$\mathbf{error}_z = \frac{1}{N} \sum (a_{\mathbf{true}}(i) \neq \mathbf{bestEstimate}_{a,z}(i))$$

$$\mathbf{error}_a = \frac{1}{N} \sum (a_{\mathbf{true}}(i) \neq \mathbf{bestEstimate}_{a,f}(i))$$

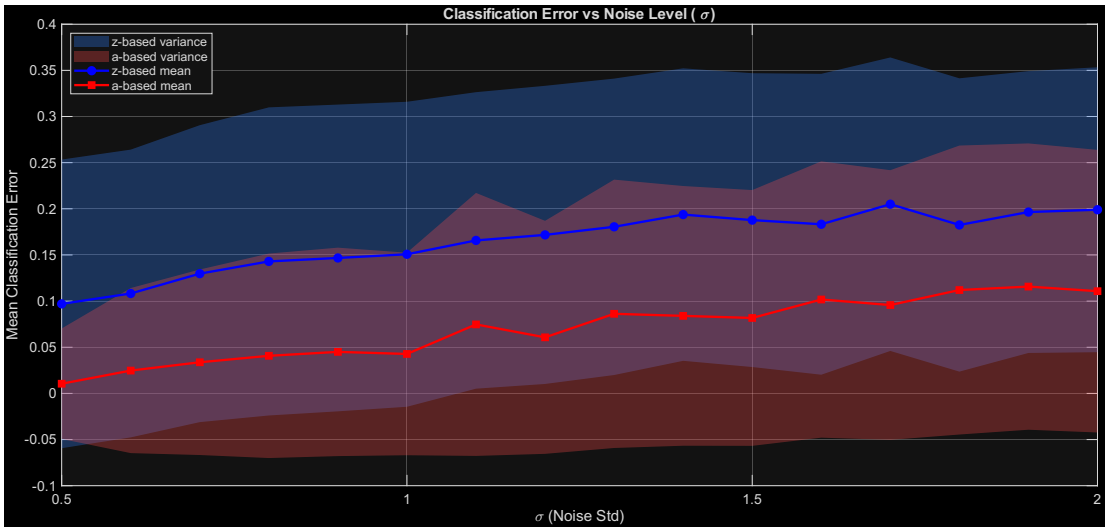

Figure 12: Error analysis

5 why high errors??

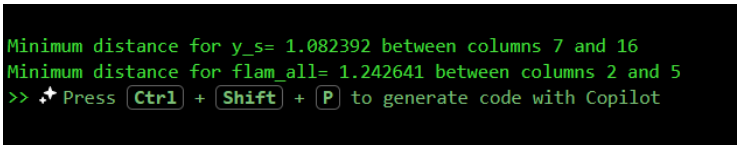

Figure 13: screenshot

Minimum distance between $y = D^h z$,with removed rows of Dh matrix is lesser than Minimum distance between f_λ.

6 Proof for distance reduction if some entry of vector is removed

Let the two 3D vectors be

$$\mathbf{a} = (x_1, y_1, z_1), \qquad \mathbf{b} = (x_2, y_2, z_2)$$

The Euclidean distance between them is

$$d_{3D} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}$$

Now, if we set $z = 0$ for both vectors, we get

$$\mathbf{a}' = (x_1, y_1, 0), \qquad \mathbf{b}' = (x_2, y_2, 0)$$

and the new distance becomes

$$d_{2D} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Comparing the two distances:

$$d_{3D} = \sqrt{d_{2D}^2 + (z_1 - z_2)^2}$$

Since $(z_1 - z_2)^2 \geq 0$, we have

$$d_{3D} \geq d_{2D}$$

Hence, setting $z = 0$ for both vectors will **decrease** (or keep the same) the distance between them. The distance can never increase.