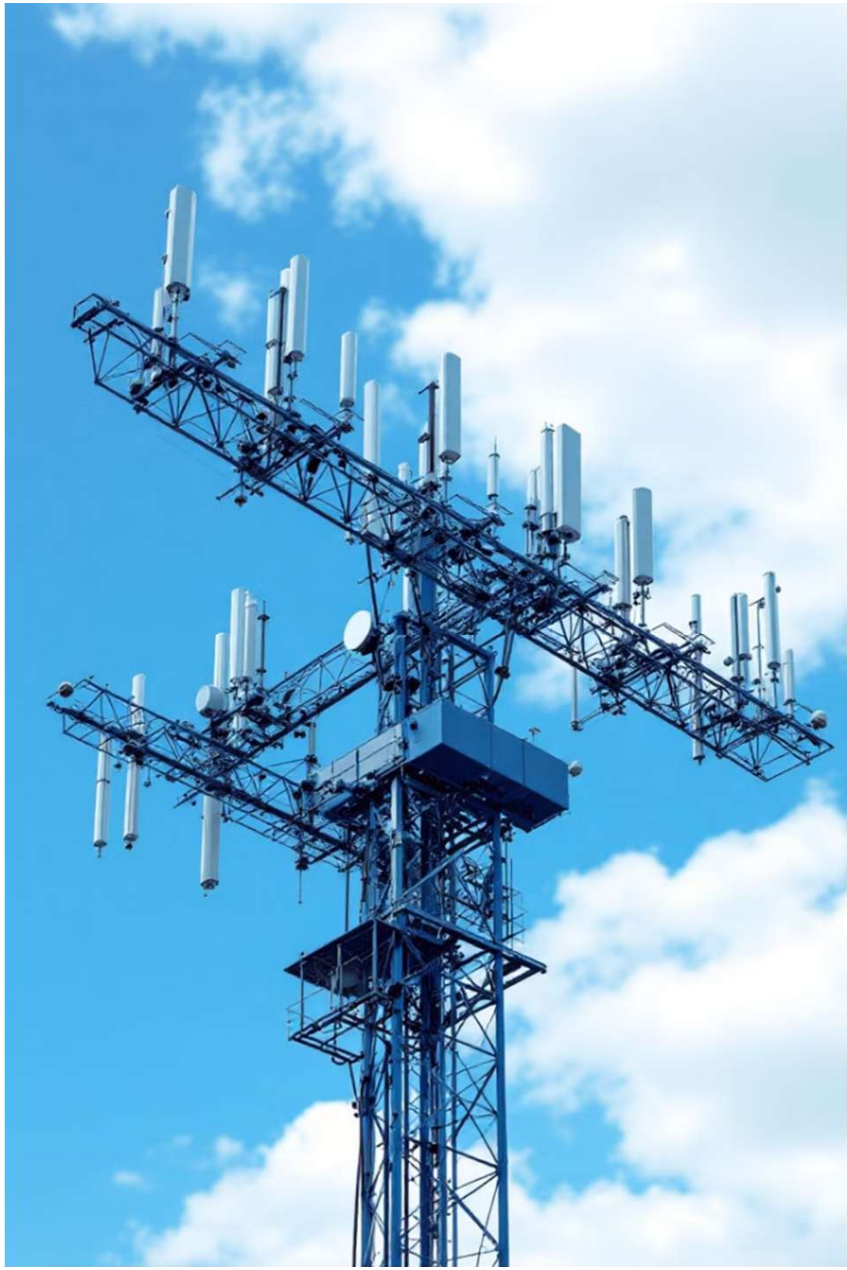
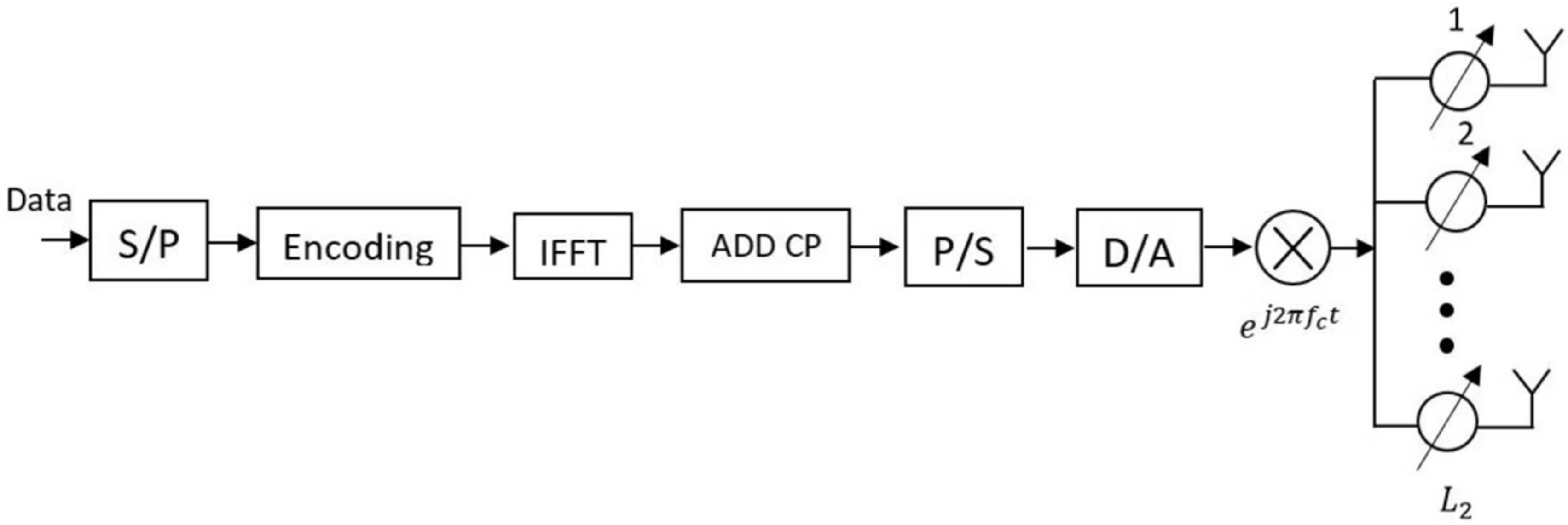
Optimizing Beamforming Vectors for Uniform Linear Arrays

This presentation explores methods for calculating optimal beamforming vectors for uniform linear arrays.

Ivan Kakurin



Optimal Beamforming Vector Calculation



Objective

Maximize signal strength in desired direction

Minimize interference from other directions

Constraints

Multi-dimensional problem

Non-convex, no closed-form solution

Solutions

Problem Formulation

$$\max_{u_l \forall l} \sum_{n=1}^{N} \sum_{k=1}^{K} \log_2 \left(1 + \frac{\left|\mathbf{u}^T \mathbf{h}_n(k)\right|^2}{\xi_1}\right)$$

$$S.T. \quad |u_l| = 1,$$

We're maximizing Sound-to-Noise ratio of signal using b-bit phase shifters with phases u from P:

$$\mathcal{P} = \left\{0,\ \frac{2\pi}{2^b},\ 2 \times \frac{2\pi}{2^b},\ 3 \times \frac{2\pi}{2^b}, \cdots,\ (2^b - 1) \times \frac{2\pi}{2^b}\right\}.$$

Successive Convex Optimization

$$\max_{u_l \forall l} \sum_{n=1}^{N} \sum_{k=1}^{K} \log_2 \left(1 + c_n(k)\right)$$

$$S.T.\ C_1 : |u_l| \leq 1 \quad \forall l$$

$$C_2 : a_n(k) = \Re\left\{\frac{\mathbf{u}^T \mathbf{h}_n(k)}{\xi_1}\right\} \quad \forall n$$

$$C_3 : b_n(k) = \Im\left\{\frac{\mathbf{u}^T \mathbf{h}_n(k)}{\xi_1}\right\} \quad \forall n$$

$$C_4 : c_n(k) \leq f(a_n(k), b_n(k)) \quad \forall n,$$

$$f(a_n(k), b_n(k)) = \tilde{a}_n(k)^2 + \tilde{b}_n(k)^2 + 2\tilde{a}_n(k)\left(a_n(k) - \tilde{a}_n(k)\right) + 2\tilde{b}_n(k)\left(b_n(k) - \tilde{b}_n(k)\right),$$

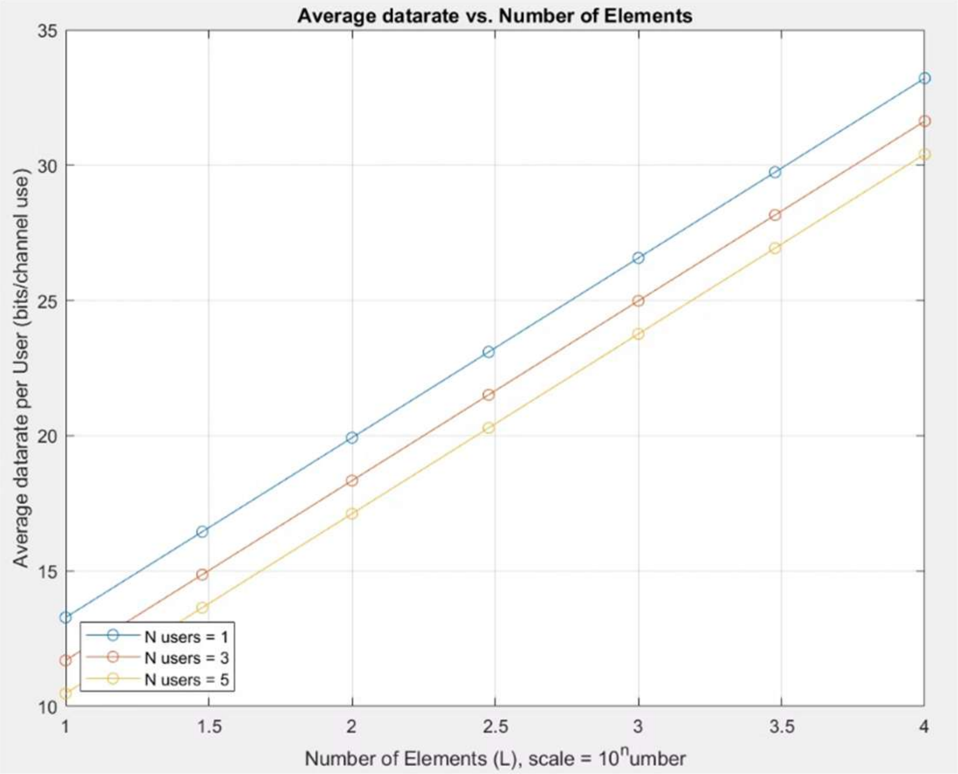
Brute-force search

We have finite number of possible beamforming vectors for b-bit phase shifters, so we can use brute-force search using cupy

$$\max_{u_l \forall l} \sum_{n=1}^{N} \sum_{k=1}^{K} \log_2 \left(1 + \frac{\left|\mathbf{u}^T \mathbf{h}_n(k)\right|^2}{\xi_1}\right)$$

$$S.T. \quad |u_l| = 1,$$

Convex Optimization Approach



I've used MATLAB CVX library for convex optimization

Computation time: 13882 s

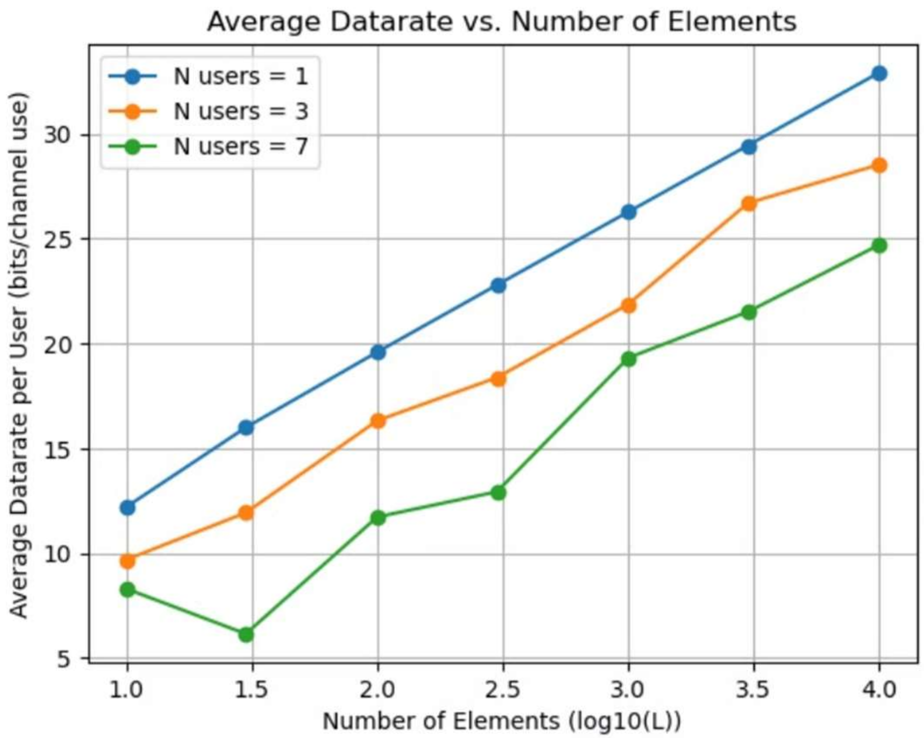
Computation complexity: O(K^4.5; N^3.5; L)

where N is the number of users,

K is the number of OFDMA sub-carriers per user

L is the number of antennas

Brute-force search



I've used Python with cupy library

Computation time: **1034 s**

Computation complexity: O(K; N; L, 2^b)

where N is the number of users,

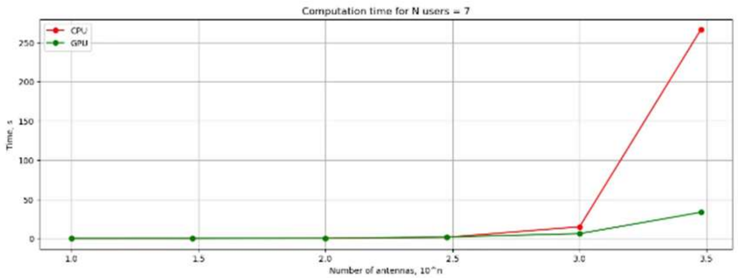
K is the number of OFDMA sub-carriers per user

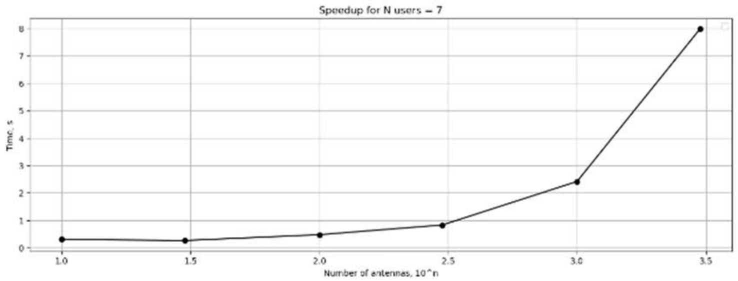
L is the number of antennas

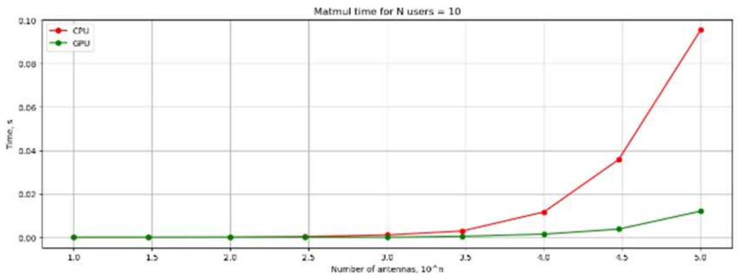
2^b is the number of possible phase-shifts

Results are almost the same as with CVX

Results Comparison







Comparing njit and cupy,

Brute-force search is 8 times faster using GPU for large antenna arrays (L = 10^4). It's because most computation time is matrix multiplication, which is 8 times faster.

Comparing cupy and CVX,

cupy is 13 times faster for large antenna arrays (L = 10^4)

Method	Computation Time (s)
njit	9231 s
cupy	1034 s
CVX	13882 s

Conclusion

Convex Optimization

Effective for calculating optimal beamforming vectors for small antenna arrays, provides optimal solution for continious phase-shifts

Cupy

Accelerates computation and improves scalability for big antenna arrays, provides optimal solution for discrete phase-shifts

Future Work

Explore combination of CVX and GPU-accelerated computations