December 2018

Lecture 2 - MIMO Detection

Lecturer: Haim Permuter *Scribe: Asaf Lavi*

Detection for MIMO has been an active field for more than twenty years and provide a powerful tool for enhancing the wireless link with an emphasis on increased spectral efficiency. This lecture is a continuation of the first lecture - Introduction to wireless communication and OFDM, and the goal of this lecture has been to provide an overview of five MIMO detection algorithms. First algorithm will be the Maximum Likelihood (ML) algorithm, optimal detector which has large complexity when number of antennas gets larger. The problem above leaded to find other detectors with much lower complexity.

One kind of detectors is the **Linear Detection**. The complexity of linear detectors is the same as the complexity of inverting or factorizing a matrix of dimensions $r \times t$, hence the name. They work by spatially decoupling the effects of the channel by a process known as MIMO equalization. This involves multiplying Y with a MIMO equalization matrix $A \in \mathbb{C}^{t\times r}$ to get $\tilde{X}(Y) \in \mathbb{C}^t$. To get the estimator $\hat{X}(Y)$ we perform coordinatewise decoding:

$$\hat{X}_i(Y) = \underset{X_i \in \mathcal{X}}{\operatorname{argmin}} |\tilde{X}_i(Y) - X_i| \tag{1}$$

for all i (i.e. it maps each coordinate to the closest constellation point).

At this lecture we will discuss three linear detectors - Zero Forcing (ZL), Best Linear Estimator (BLE) and Successive Interference Cancellation (SIC). In the end of the lecture we will meet with an iterative algorithm named Sphere Decoding (SD). A part of this lecture is taken from the lecture notes of D. Ezri[1].

I. MIMO DETECTION

Reminder : as we know from the end of the last lecture, scalar signals are not used in real world. Assuming we have t antennas at base station, and r antennas at terminal we can define the Multiple Input Multiple Output detection problem with the following equation:

$$Y = HX + W \tag{2}$$

Where $H \in \mathbb{C}^{r\times t}$ is a matrix that represent the channels between each of the antennas, $X \in \mathbb{C}^{t}$ is a vector of symbols from some constellation, e.g. 4-QAM, 16-QAM etc. and $W \in \mathbb{C}^{r}$ is a complex noise.

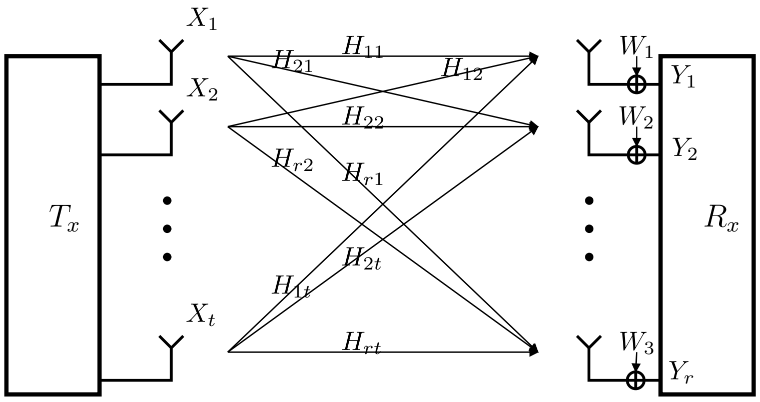

Fig. 1. MIMO configuration with t and r antennas.

A. Spatial multiplexing

We consider a spatial multiplexing MIMO system where several layers or data streams are simultaneously transmitted (i.e. technique to transmit independent and separately encoded data signals).

For a SIMO (Single Input Multiple Output) system, the receiver combines data streams from multiple transmit antennas using maximum ratio combining methods to achieve diversity gain. For multiple transmit antennas, the channel becomes more complicated, and there is interference between different transmitted streams. When the transmitter has no channel knowledge (i.e. H is unknown to the encoder), the receiver is alone in exploiting MIMO capacity, which usually means that a complicated algorithm is required.

Another state is when the MIMO system utilizes Channel State Information (CSI) at the transmitter. In most cases, only partial CSI is available at the transmitter because of the limitations of the feedback channel. The input-output relationship can be described as

$$Y = HEX + W \tag{3}$$

Where H, Y, X, W defined the same as before and $E \in \mathbb{C}^{t \times s}$ is a linear precoding matrix. A precoding matrix E is used to precode the symbols in the vector to enhance the performance. The column dimension of E can be selected smaller than t which is useful if $t \neq s$ because of several reasons. Examples of the reasons are as follows: either the rank of the MIMO channel or the number of receiver antennas is smaller than the number of transmit antennas.

II. **Algorithm 1:** Maximum Likelihood

This is the optimal detector from the point of view of minimizing the probability of error. The ML receiver computes the most probable symbol X given the measurements Y. The maximum likelihood detector with some kind of noise at the receiver antennas:

$$\hat{X}(Y) = \underset{X \in \mathcal{X}^t}{\text{argmax}} P_r(Y|X) \tag{4}$$

$$= \underset{X \in \mathcal{X}^t}{\text{argmax}} P_r(W = Y - HX|X) \tag{5}$$

when W is an $i.i.d$ Gaussian noise vector and independent of the input X then:

$$\underset{X \in \mathcal{X}^t}{\text{argmax}} P_r(W = Y - HX|X) \tag{6}$$

$$= \underset{X \in \mathcal{X}^t}{\text{argmax}} \frac{1}{\sqrt{2\pi}\sigma} \mathrm{e}^{\frac{-||Y-HX||^2}{2\sigma^2}} \tag{7}$$

$$= \underset{X \in \mathcal{X}^t}{\text{argmin}} ||Y - HX||^2 \tag{8}$$

The minimization is over all possible transmitted vectors, Hence, solving this problem involves computing the objective function for all $\mathcal{X}^t$ potential values of X. Therefore the ML detector has prohibitive (exponential in t) complexity (e.g. for 64-QAM and 8 antennas the complexity is 64^8).

III. Algorithm 2: Zero Forcing

The algorithm solves the following problem:

$$\tilde{X}(Y) = \underset{X \in \mathcal{X}^t}{\operatorname{argmin}} ||Y - HX||^2 \tag{9}$$

We can find the minimum of the expression $||Y - HX||^2$ by the derivation with respect to X and compare it to zero:

$$\begin{aligned} \frac{\partial}{\partial X}||Y - HX||^2 &= \frac{\partial}{\partial X}(HX - Y)^*(HX - Y) && (10)\\ &= 2H^*(HX - Y) && (11)\\ &= 0 && (12) \end{aligned}$$

Hence,

$$\tilde{X}(Y) = H^{\dagger}Y \tag{13}$$

if H^{-1} exists then:

$$H^{\dagger} = H^{-1} \tag{14}$$

In case that H^{-1} doesn't exists, the generic expression for $H^{\dagger}$ is:

$$H^{\dagger} = (H^H H)^{-1} H^H \tag{15}$$

and finally, to get the estimator $\hat{X}(Y)$

$$\begin{aligned} \hat{X}_i(Y) &= \underset{X_i \in \mathcal{X}}{\operatorname{argmin}} |\tilde{X}_i(Y) - X_i| && (16)\\ &= \underset{X_i \in \mathcal{X}}{\operatorname{argmin}} |(H_i^{\dagger}Y) - X_i| && (17) \end{aligned}$$

for all i. The complexity of obtaining $H^{\dagger}$ from H is $\sim O(t^3)$ for a square matrix. However obtaining $\hat{X}(Y)$ from $\tilde{X}(Y)$ is done in a time linear in t i.e. $\sim O(t)$. We can see that the complexity is much lower compare to ML algorithm complexity.

IV. **Algorithm 3:** Best Linear Estimator

The BLE is another example for linear detection algorithm (which is very similar to ZF). In BLE we construct the best linear estimator of X conditioned on Y as follows:

$$\tilde{X}_{BLE}(Y) = E[X] + \Sigma_{XY}\Sigma_{YY}^{-1}(Y - E[Y]) \tag{18}$$

Using the fact that $E[X] = E[Y] = \bar{0}$ and $\Sigma_{XX} = c\mathbf{I}$ where c is some constant that usually normalize to one, we can calculate the rest of the elements on equation 18 as follows:

$$\begin{aligned} \Sigma_{XY} &= E[XY^*] & (19)\\ &= E[X(Hx + W)^*] & (20)\\ &= \Sigma_{XX}H^* & (21)\\ &= H^* & (22) \end{aligned}$$

$$\begin{aligned} \Sigma_{YY} &= E[(Hx + W)(Hx + W)^*] & (23)\\ &= HH^* + \sigma^2\mathbf{I} & (24) \end{aligned}$$

Hence,

$$\tilde{X}_{BLE}(Y) = H^*(HH^* + \sigma^2\mathbf{I})^{-1}Y \tag{25}$$

We can notice the similarity between the BLE and ZF estimators - both created by almost the same matrix multiplications, except the component $\sigma^2 I$ that has been added to the inverse matrix. Hence, the noisy components in Y will reduced at the matrix multiplication and by that, reduce the noise affect on the estimator.

The estimator $\hat{X}(Y)$ obtained by:

$$\begin{aligned} \hat{X}_i(Y) &= \operatorname*{argmin}_{X_i \in \mathcal{X}} |\tilde{X}_{iBLE}(Y) - X_i| & (26)\\ &= \operatorname*{argmin}_{X_i \in \mathcal{X}} |(H^*(HH^* + \sigma^2\mathbf{I})^{-1}Y)_i - X_i| & (27) \end{aligned}$$

for all i. The complexity of achieving $\tilde{X}_{BLE}(Y)$ is similar to the complexity of achieving $\tilde{X}_{ZL}(Y) \sim O(t^3)$.

Additionally, we can notice that in a limit of high SNR:

$$\lim_{\sigma^2 \Rightarrow \infty} \tilde{X}_{BLE}(Y) = \tilde{X}_{ZF}(Y) \tag{28}$$

V. **ALGORITHM 4:** SUCCESSIVE INTERFERENCE CANCELLATION (V-BLAST)

The Successive Interference Cancellation (SIC) method, or Vertical Bell Laboratories Layered Space-Time(V-BLAST) who invented by P.W. Wolniansky, G.J. Foschini, G.D. Golden and R.A. Valenzuela from Bell Laboratories (hence the name), is the last example of linear detection that we will discuss at this lecture, borrows concepts from decision feedback equalization and synchronization[2]. The algorithm begins with linear processing, similarly to equation 13:

$$\tilde{X} = (H^*H)^{-1}H^*Y \tag{29}$$

$$= (H^*H)^{-1}H^*(HX + W) \tag{30}$$

$$\triangleq G(HX + W) \tag{31}$$

$$= X + GW \tag{32}$$

At this point, we look for the index k_1 featuring the largest per stream post processing SNR associated with equation 29

$$k_1 = \underset{j=0,1,\ldots,t-1}{\operatorname{argmin}} ||G(j,:)||^2 \tag{33}$$

The symbol X_{k_1} which is the most robust, is decoded linearly, as implied by (26)

$$\tilde{X}_{k_1} = G(k_1,:)Y \tag{34}$$

$$\tag{35}$$

and then sliced to obtain $\hat{X}_{k_1} \in QAM$. After we estimated $\hat{X}_{k_1}$ of X_{k_1}, the algorithm proceeds to cancel out the estimated contribution of X_{k_1} in the original received vector Y

$$Y_2 = Y - H_{k_1}\hat{X}_{k_1} \tag{36}$$

s.t. now we have new problem to solve, similar to before but with $t-1$ symbols to detect

$$Y_2 = H_{k_1}^{-} X_{k_1}^{-} + W \tag{37}$$

where $H_{k_1}^{-}$ is H with the k_1^{th} column eliminated, and $X_{k_1}^{-}$ is X with the k_1^{th} element eliminated. The algorithm continues recursively with the computation of the pseudo inverse G_2 of $H_{k_1}^{-}$, the determination of

$$k_2 = \underset{j=0,1,\ldots,t-1, j\neq k_1}{\operatorname{argmin}} ||G_2(j,:)||^2 \tag{38}$$

and so on, until all symbols are decoded.

We can notice that error propagation is one of the main problem of the SIC - if we estimated wrong one of the symbols it will affect all other estimations along the process. In the 2×2 case, the performance is very much similar to ZF, however, in the case of $M > 2$, the SIC algorithm exhibits somewhat superior performance[3].

VI. **Algorithm 5:** Sphere Decoding

Sphere decoding is an iterative method for the computation of the ML solution, and trades off performance versus complexity by controlling a parameter r. By choosing a large enough r, the performance of SD approaches that of the ML detector. For small r, the search space (and hence complexity) of the SD is much smaller than that of the ML detector, but it suffers a performance degradation as a result [4].

Remember that

$$Y = HX + W \tag{39}$$

define:

$$H = QR \tag{40}$$

where

- Q - unitary matrix
- R - upper triangle matrix

The decomposition at (40) is always possible, and since the squared-distance norm does not change under multiplication by a unitary matrix Q^H, we have

$$||Y - HX||^2 = ||Q^H Y - Q^H QRX||^2 \tag{41}$$

$$= ||\tilde{Y} - RX||^2 \tag{42}$$

$$= \sum_{i=t}^{1} |\tilde{Y}_i - \sum_{j=i}^{t} R_{i,j} X_j|^2 \tag{43}$$

Hence, the sphere decoder solves the following problem:

$$\hat{X}(Y) = \underset{X \in \mathcal{X}^t, ||\tilde{Y} - RX||^2 \leq r}{\operatorname{argmin}} ||\tilde{Y} - RX||^2 \tag{44}$$

The basic premise in sphere decoding is rather simple: we attempt to search over only lattice points $X \in \mathcal{X}^t$ that lie in a certain sphere of radius r around the given vector X, thereby reducing the search space and hence the required computations (see figure above). Clearly, the closest lattice point inside the sphere will also be the closest lattice point for the whole lattice. However, close scrutiny of this basic idea leads to two key questions.

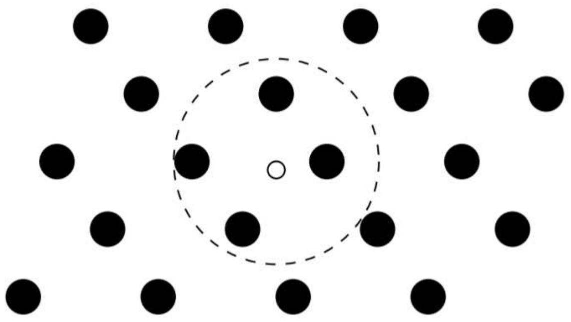

Fig. 2. Idea behind the sphere decoder.

1. $How\ to\ choose\ r$? Clearly, if r is too large, we obtain too many points and the search remains exponential in size, whereas if r is too small, we obtain no points inside the sphere. A natural candidate for r is the covering radius of the lattice, defined to be the smallest radius of spheres centered at the lattice points that cover the entire space. This

is clearly the smallest radius that guarantees the existence of a point inside the sphere for any vector X. The problem with this choice of r is that determining the covering radius for a given lattice is itself NP hard [5].

2. *How can we tell which lattice points are inside the sphere?* If this requires testing the distance of each lattice point from X (to determine whether it is less than r), then there is no point in sphere decoding as we will still need an exhaustive search.

Sphere decoding does not really address the first question. However, it does propose an efficient way to answer the second, and more pressing, one. The basic observation is the following. Although it is difficult to determine the lattice points inside a general t-dimensional sphere, it is trivial to do so in the (one-dimensional) case of $t = 1$. The reason is that a one-dimensional sphere is simply an interval and so the desired lattice points will be the integer values that lie in this interval. We can use this observation to go from dimension k to dimension $k+1$. Suppose we have determined all k-dimensional lattice points that lie in a sphere of radius r. Then for any such k-dimensional point, the set of admissible values of the $(k+1)^{th}$ dimensional coordinate that lie in the higher dimensional sphere of the same radius r forms an interval. The above means that we can determine all lattice points in a sphere of dimension t and radius r by successively determining all lattice points in spheres of lower dimensions $1, 2, ..., t$ and the same radius r. Such an algorithm for determining the lattice points in an t-dimensional sphere essentially constructs a tree where the branches in the k^{th} level of the tree correspond to the lattice points inside the sphere of radius r and dimension k (see figure above). Moreover, the complexity of such an algorithm will depend on the size of the tree, i.e., on the number of lattice points visited by the algorithm in different dimensions.

Note that choosing $r = \infty$ gives us the ML decoder. For a smaller r, the solver can exploit the upper triangular nature of r to prune many candidate solutions, thereby reducing the detection complexity significantly. One surprising property of the SD is that if it finds a valid solution, it is the same solution that the ML detector would have returned.

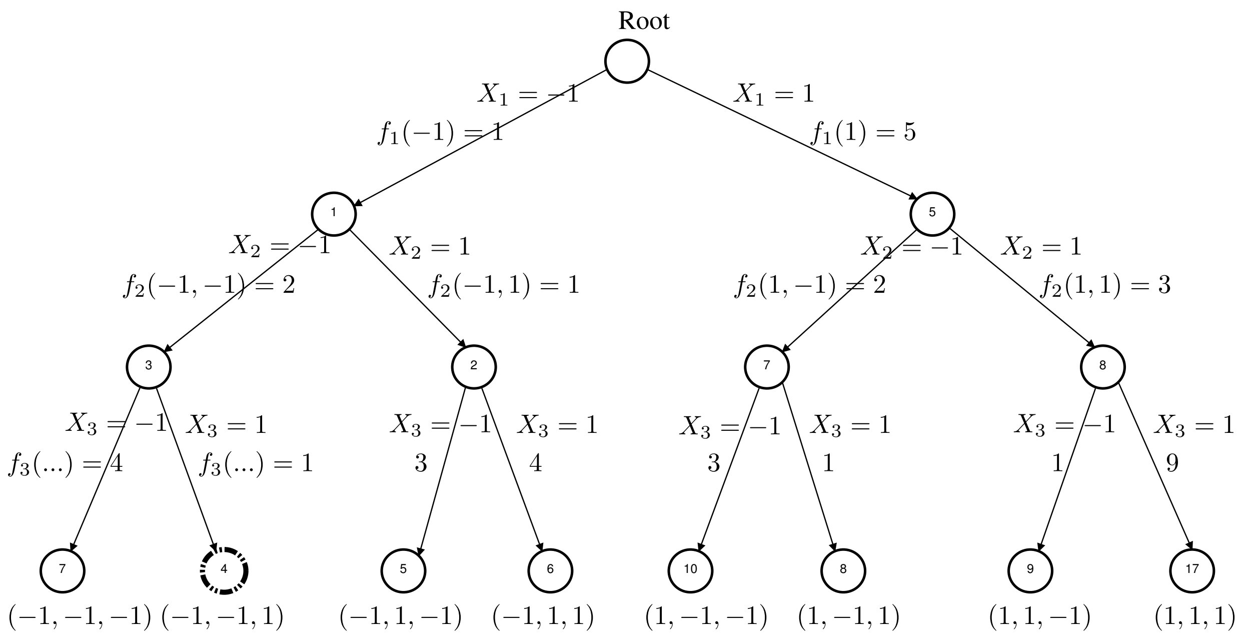

Fig. 3. Sample tree generated to determine lattice points in a 3-dimensional sphere. $\hat{X}$ will be $(-1,-1,1)$

REFERENCES

[1] D. Ezri,*MIMO-OFDM Lecture Notes*, Tel Aviv and Ben Gurion Universities

[2] P.W. Wolniansky, G.J. Foschini, G.D. Golden and R.A. Valenzuela *"V-BLAST: an architecture for realizing very high data rates over the rich-scattering wireless channel"*

[3] Loyka S. and Gagnon F., *"Performance analysis of the V-BLAST algorithm: an analytical approach"*, IEEE Trans. Wireless Comm.,3(4):13261337, 2004.

[4] E. G. Larsson, *"MIMO Detection Methods: How They Work, lecture notes"*

[5] B. Hassibi and H. Vikalo *"On the Sphere Decoding Algorithm. I. Expected Complexity"*, Department of Electrical Engineering California Institute of Technology, Pasadena, CA 91125