I. PAC bound for Group Testing
II. SSR for OFDM channel estimation: Implementation in SDR

Monika Bansal
Advisor: Dr. Chandra R Murthy

Department of ECE, IISc

January 16, 2016

Overview

Group Testing

A set of n items with k defective items (k<<n).

- **Group test:** A group of items is tested in a group test
  - Test outcome **1** indicates presence of **defective** item(s)
  - Outcome **0** indicates all items are **non-defective** in the test
- **Main issues:**
  - Sample Complexity
  - Correctness of solution
  - Pooling Design

Group Testing Model

$$\mathbf{y} = \mathbf{A}\mathbf{x}$$
$$= \bigvee_{j:x_j=1} \mathbf{a}_{c_j}$$

$\mathbf{y} \in \{0,1\}^M$ is the binary test outcome vector
$\mathbf{a}_{c_j} \in \{0,1\}^M$ is the j^{th} column of $\mathbf{A}$
$\mathbf{a}_{r_i} \in \{0,1\}^n$ is the i^{th} row of $\mathbf{A}$
$\mathbf{A}(i,j) \sim \mathcal{B}(p)$ i.i.d.
$\mathbf{x} \in \{0,1\}^n$ is test item vector

Learning Problem

Learn unknown target function $f(\cdot) \in \mathcal{C}$

Available items to the **learner:**

- Random examples: $\mathbf{a}_i$
- Corresponding label: y_i

$$\mathbf{a}_i \in \{0,1\}^n$$
$$y_i = f(\mathbf{a}_i) \in \{0,1\}$$
$$f : \{0,1\}^n \rightarrow \{0,1\} \text{ is some boolean function}$$

How many examples do we need to output an hypothesis f^* s.t. maximum error is ϵ with confidence $1-\delta$?

PAC Learning Model

- A learning algorithm is said to be PAC-learn $\mathcal{C}$ with *approximation parameter* ϵ *and confidence parameter* δ if $\forall$ distributions $\mathcal{D}$ and all target functions $f \in \mathcal{C}$, the algorithm draws M samples, runs for time at most t and outputs a function f^* s.t.

$$Pr\left(e(f^*, f) > \epsilon\right) \leq \delta$$

$$e(f^*, f) = Pr_{\mathbf{a} \sim \mathcal{D}}\left(f^*(\mathbf{a}) \neq f(\mathbf{a})\right)$$

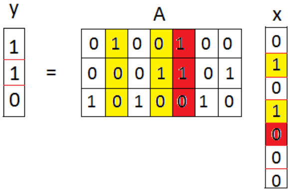
With prob. $1 - \delta$ the output hypothesis f^* will make at most ϵ error.

Group testing as a learning problem

	Group testing	**Learning model**
Target function	$\mathbf{x}$	$f(\cdot)$
Random example	$\mathbf{a}_{r_i}$	$\mathbf{a}_i$
Label	$\mathbf{y}(i)$	y_i
Distribution	$\mathcal{B}(p)$	$\mathcal{D}$
Output hypothesis	$\hat{\mathbf{x}}$	f^*

Goal: Come up with the bound on the no. of group tests (M) using PAC learning framework.

Column Matching (CoMa) Algorithm[1]



- j^{th} item is declared to be defective if: All the locations i where it has ones in the j^{th} column of $\mathbf{A}$ also corresponds to ones in the result vector y_i.
- Error event occurs: When the non-defective item is declared as the defective item. This happened because the non-defective item was **hidden**.

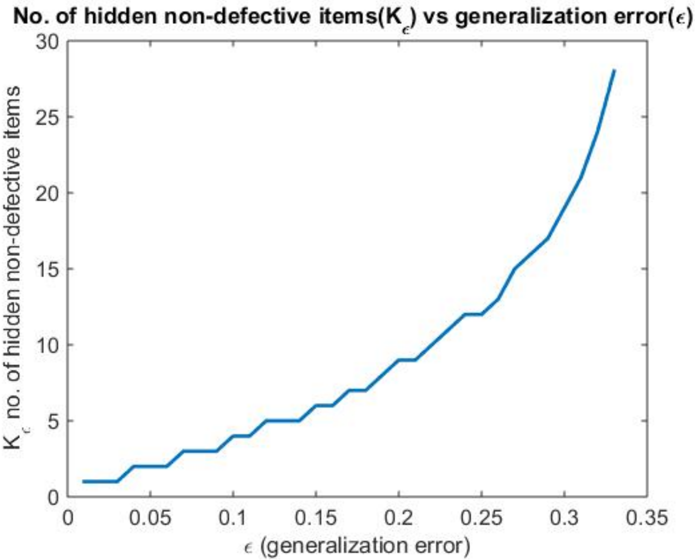
[1] Non-adaptive Group Testing: Explicit Bounds and Novel Algorithms

PAC analysis for CoMa Algorithm

$$\underbrace{\begin{bmatrix}1\\1\\0\end{bmatrix}}_{y} = \underbrace{\begin{bmatrix}0&1&0&0&1&0&0\\0&0&0&1&1&0&1\\1&0&1&0&0&1&0\end{bmatrix}}_{A} \underbrace{\begin{bmatrix}0\\1\\0\\1\\0\\0\\0\end{bmatrix}}_{x}$$

$$\begin{aligned} e(f^*, f) &= Pr_{\mathbf{a}\sim\mathcal{D}}(f^*(\mathbf{a}) \neq f(\mathbf{a})) \\ &= (1-p)^k(1-(1-p)^{K(M)}) \\ \text{For } \epsilon \text{ error,} \qquad K_\epsilon &= \left\lceil \frac{ln\left(1-\frac{\epsilon}{(1-p)^k}\right)}{ln(1-p)} \right\rceil \end{aligned} \tag{1}$$

is the **maximum** no. of unidentified (hidden) non-defective items allowed.


No. of hidden non-defective items(K$_\epsilon$) vs generalization error(ϵ)
K$_\epsilon$ no. of hidden non-defective items
ϵ (generalization error)
0
5
10
15
20
25
30
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35

PAC Bound

$$
\begin{aligned}
&Pr(e(f^*, f) > \epsilon) \leq \delta \\
&Pr(K(M) > K_\epsilon) \leq \delta \\
&Pr(K(M) \leq K_\epsilon) \geq 1 - \delta
\end{aligned} \tag{2}
$$

where, $K(M)$ is the no. of hidden non-defective items after M tests.

Probability for exactly one item to be remain hidden

Step 1:

$$\begin{aligned} P_i &= \Pr(i^{th} \text{ item is hidden in a test}) \\ &= \Pr(i^{th} \text{ item participates}) \times \Pr(\text{atleast one of the defective} \\ &\qquad \text{items also participates}) + \Pr(i^{th} \text{ item does not participate}) \\ &= p(1-p)^k + (1-p) \end{aligned} \tag{3}$$

Step 2:

$$\begin{aligned} Ph_1(M) &= \Pr(i^{th} \text{ item remains hidden after } M \text{ tests}) \\ &= \Pr(i^{th} \text{ item remains hidden in all } M \text{ tests}) \\ &= (P_i)^M = (1 - p(1-p)^k)^M \end{aligned} \tag{4}$$

Probability for exactly one item to be remain hidden

Step 3:

$$P_1(M) = \Pr(\text{one item remains hidden after } M \text{ tests})$$
$$= \binom{n-k}{1} Ph_1(M)\left(1 - Ph_1(M)\right)^{n-k-1} \quad (5)$$

Probability for exactly two items to be remain hidden

Step 1:

$$\begin{aligned} P_{ij} &= \Pr(i^{th} \text{ and } j^{th} \text{ items are hidden}) \\ &= p^2\left(1-(1-p)^k\right) + 2p(1-p)\left(1-(1-p)^k\right) + (1-p)^2 \\ &= 1-\left(1-(1-p)^2\right)(1-p)^k \end{aligned} \tag{6}$$

Step 2:

$$\begin{aligned} Ph_2(M) &= \Pr(i^{th} \text{ and } j^{th} \text{ items remain hidden after } M \text{ tests}) \\ &= (P_{ij})^M = \left(1-\left(1-(1-p)^2\right)(1-p)^k\right)^M \end{aligned} \tag{7}$$

Probability for exactly two items to be remain hidden

Step 3:

$$\begin{aligned} P_2(M) = & \ \Pr(\text{two items remain hidden after } M \text{ tests}) \\ = & \binom{\binom{n-k}{2}}{1} Ph_2(M) \left(1 - Ph_2(M)\right)^{\binom{n-k}{2}-1} \\ = & \binom{n-k}{2} Ph_2(M) \left(1 - Ph_2(M)\right)^{\binom{n-k}{2}-1} \end{aligned} \quad (8)$$

Probability for exactly q items to be remain hidden

$$P_q(M) = \binom{n-k}{q} Ph_q(M) \left(1 - Ph_q(M)\right)^{\binom{n-k}{q}-1} \quad \text{and}$$
$$Ph_q(M) = \left(1 - \left(1 - (1-p)^q\right)(1-p)^k\right)^M \quad (9)$$

Lower Bound

$$
\begin{aligned}
Pr(K(M) \leq K_\epsilon) &= \sum_{q=0}^{K_\epsilon} P_q(M) \\
&= (1 - Ph_1(M))^{n-k} + (n-k)Ph_1(M)\,(1 - Ph_1(M))^{(n-k)-1} \\
&\quad + \ldots + \binom{n-k}{K_\epsilon} Ph_{K_\epsilon}(M)\,(1 - Ph_{K_\epsilon}(M))^{\binom{n-k}{K_\epsilon}-1} \\
&\geq (1 - Ph_1(M))^{n-k} + (n-k)Ph_2(M)\,(1 - Ph_2(M))^{(n-k)-1}
\end{aligned}
$$

Bound on the Number of Tests (M)

$$
\begin{aligned}
Pr(K(M) \leq K_{\epsilon}) &\geq 1-\delta \\
(1-Ph_1(M))^{n-k} + (n-k)Ph_1(M)\,(1-Ph_1(M))^{(n-k)-1} &\geq 1-\delta \\
(1-Ph_1(M))^{n-k-1} &\geq 1-\delta \\
M &\geq \frac{ln\left(1-(1-\delta)^{\frac{1}{n-k-1}}\right)}{ln\left(1-p(1-p)^k\right)}
\end{aligned} \tag{10}
$$

Further Simplification

By using the inequality $(1-\delta)^{\frac{1}{n-k-1}} \leq (1-\frac{1}{n-k-1}\delta)$ in eq (10).

$$M \geq \frac{ln\left(\frac{n-k-1}{\delta}\right)}{ln\left(\frac{1}{1-p(1-p)^k}\right)} \tag{11}$$

Comparison with the Group testing Literature Bound

Bounding the denominator,

$$\begin{aligned} ln\left((1-p(1-p)^k)^{-1}\right) &\geq ln\left(1+p(1-p)^k\right) \\ &\geq p(1-p)^k \end{aligned} \tag{12}$$

If $p = \frac{1}{k}$, then by applying the inequality $(1-\frac{1}{k})^k \geq \frac{1}{exp(1)}$ we get,

$$M \geq exp(1)k\left[ln(n-k-1) + ln(\frac{1}{\delta})\right]$$

Group Testing literature bound:

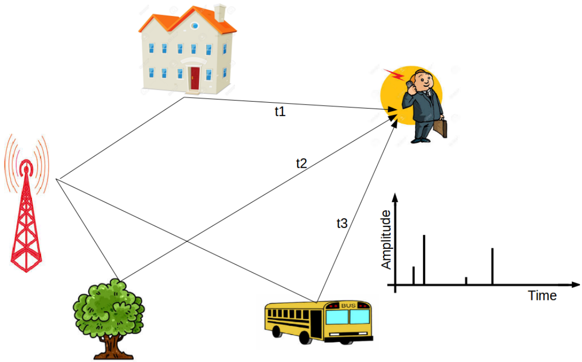
$$M \geq 2(1+\Delta)exp(1)k \times ln(n)$$

for probability of error at most $n^{-\Delta}$.

Summary

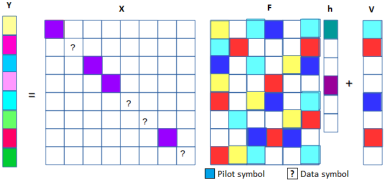
- Group testing as a learning problem
- PAC analysis for the CoMa algorithm
- Bound on number of tests (M)

Wireless Communication Channel



Sparse in lag domain

OFDM Channel Model



$$\mathbf{y} = \mathbf{XFh} + \mathbf{v}$$

$\mathbf{y} \in \mathbb{C}^{N \times 1}$ is the received vector after FFT

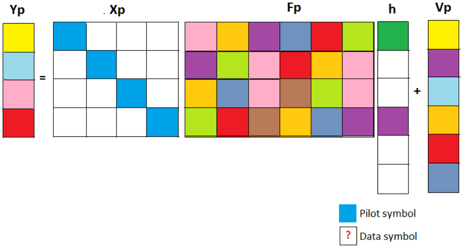
$\mathbf{X} \in \mathbb{C}^{N \times N}$ contains data symbol and pilot symbols along the diagonal

$\mathbf{F} \in \mathbb{C}^{N \times L} (N > L)$ contains the first L columns of $N \times N$ DFT matrix

$\mathbf{v} \in \mathbb{C}^{N \times 1} \sim \mathcal{CN}(0, \sigma^2 \mathrm{I})$ is the AWGN noise

$\mathbf{h} \in \mathbb{C}^{L \times 1}$ is the time domain channel response

Channel Model using *Pilots* only:



$$\mathbf{y}_P = \mathbf{X}_p\mathbf{F}_p\mathbf{h} + \mathbf{v}_p, \qquad (P < L)$$
$$= \phi_p\mathbf{h} + \mathbf{v}_p$$

SBL Framework

$$\mathbf{h} \sim \mathcal{CN}(0, \mathbf{\Gamma}), \qquad \mathbf{\Gamma} = \text{diag}(\gamma(1), ..., \gamma(L))$$

ML estimation problem:

$$\begin{aligned}\hat{\mathbf{h}} &= \underset{\mathbf{h}, \gamma \in \mathbb{R}_{+}^{L \times 1}}{\arg\max}\ p(\mathbf{y}_p | \mathbf{h}; \gamma) p(\mathbf{h}; \gamma) \\ &= \underset{\mathbf{h}, \gamma \in \mathbb{R}_{+}^{L \times 1}}{\arg\min} \frac{\|\mathbf{y}_p - \mathbf{X}_p \mathbf{F}_p \mathbf{h}\|_2^2}{\sigma^2} + \log|\mathbf{\Gamma}| + \mathbf{h}^H \mathbf{\Gamma}^{-1} \mathbf{h}\end{aligned}$$

Instead of estimating **h** directly, we estimate γ first

$$\hat{\gamma}_{ML} = \underset{\gamma \in \mathbb{R}_{+}^{L \times 1}}{\arg\max}\ p(\mathbf{y}_p; \gamma)$$

EM algorithm

$$p(\mathbf{h};\gamma) = \prod_{i=1}^{L} (\pi\gamma(i))^{-1} exp\left(-\frac{|h(i)|^2}{\gamma(i)}\right)$$

$$\textbf{E-step: } Q\left(\gamma|\gamma^{(r)}\right) = \mathbb{E}_{\mathbf{h}|\mathbf{y}_p;\gamma^{(r)}}\left[\log\, p(\mathbf{y}_p, \mathbf{h};\gamma)\right]$$

$$\textbf{M-step: } \gamma^{(r+1)} = \underset{\gamma\in\mathbb{R}_{+}^{L\times 1}}{\arg\max}\; Q\left(\gamma|\gamma^{(r)}\right)$$

$$\gamma^{(r+1)}(i) = \Sigma(i,i) + |\mu(i)|^2$$

Probability densities:

$$p\left(\mathbf{h}|\mathbf{y}_p;\gamma^{(r)}\right) = \mathcal{CN}(\mu,\Sigma)$$

$$\Sigma = \Gamma^{(r)} - \Gamma^{(r)}\phi_p^H(\sigma^2\mathrm{I}_{P_b} + \phi_p\Gamma^{(r)}\phi_p^H)^{-1}\phi_p\Gamma^{(r)}, \;\; \mu = \sigma^{-2}\Sigma\phi_p^H\mathbf{y}_p$$

Algorithm 1 SBL for estimating time domain channel taps

Input: $\mathbf{y}_p, \phi_p, r_{max}$ and ϵ.

Initialize $\Gamma^{(0)} = \mathrm{I}_L$, Set difference $= 1$, $r = 0$

while (difference$> \epsilon$ and $r < r_{max}$)

E-step: $\mu = \sigma^{-2}\Sigma\phi_p^H\mathbf{y}_p$

$\Sigma = \Gamma^{(r)} - \Gamma^{(r)}\phi_p^H\left(\sigma^2\mathrm{I}_{P_b} + \phi_p\Gamma^{(r)}\phi_p^H\right)^{-1}\phi_p\Gamma^{(r)}$

M-step: $\gamma^{(r+1)}(i) = \Sigma(i,i) + |\mu|^2$ for $i = 1, 2, \ldots, L$

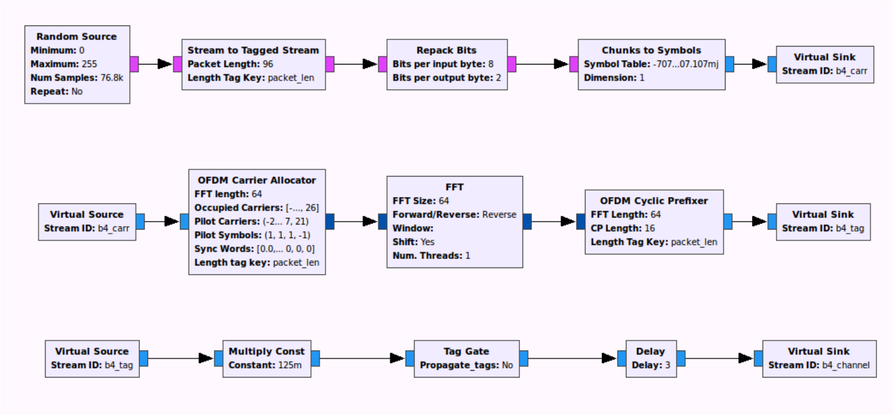
difference $\triangleq \|\gamma^{(r+1)} - \gamma^{(r)}\|_2^2$, $r \leftarrow r + 1$ **end**

output: $\mu, \gamma^{(r)}$

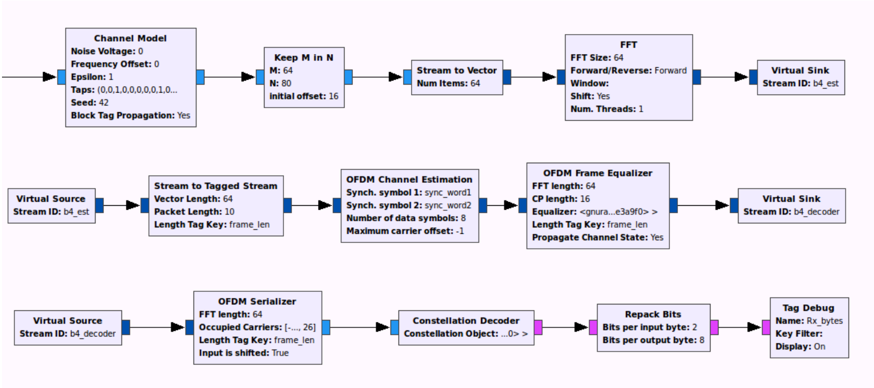
Introduction to GNU Radio

- A software development tool kit with **signal processing** blocks written in C++/Python.
- GRC (GNU Radio Companion) is the user interface for GNU Radio.
- Can be used with external RF hardware (such as USRP N210) to create Software Defined Radio(SDR).

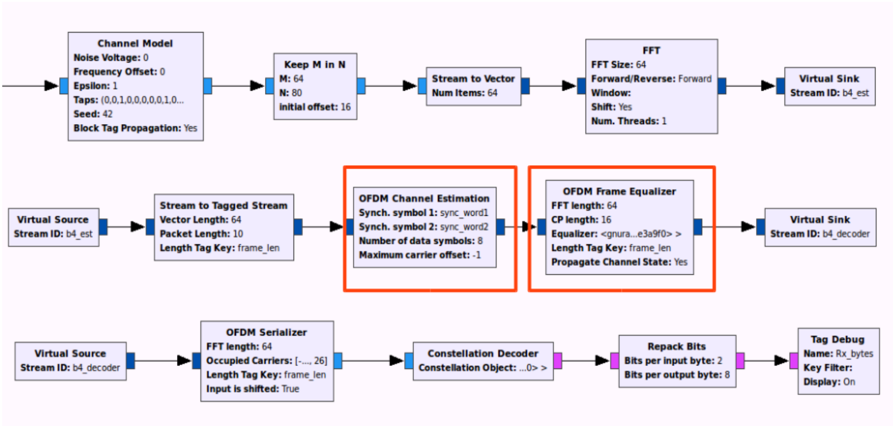
OFDM Transmitter in GNU Radio



OFDM Receiver in GNU Radio

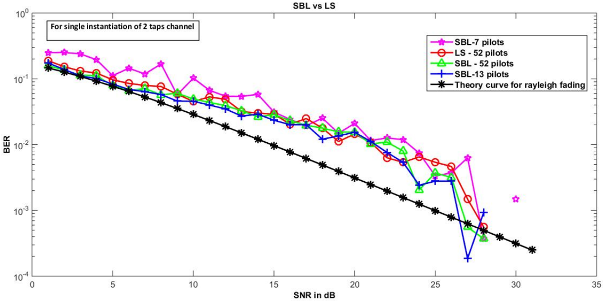


OFDM Receiver in GNU Radio

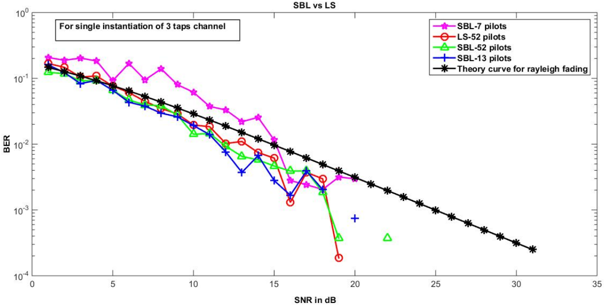


Our Contribution

Implementation of SBL algorithm in place of existing Channel Estimation block.



Comparing Plot



Summary

- Implemented SBL based channel estimation algorithm in GNU radio.
- Simulation curves verify the advantages of SSR based algorithm.

Future Work

- Implementation of joint channel estimation and data detection SBL and other SSR based algorithms in simulation set up.
- Real time performance evaluation of SSR based channel estimation algorithms using RF hardware such as USRP N210.
- Analyse other group testing algorithms in PAC learning model and give bounds on the no. of tests.