# Fast Non-Negative Orthogonal Matching Pursuit

Authors: Mehrdad Yaghoobi, Di Wu, and Mike E. Davies

Publication: IEEE SIGNAL PROCESSING LETTERS, 2015

Speaker: Haeung Choi

**Short summary: Non-negative signal is an important class of sparse signals. Many algorithms have already been proposed to recover such non-negative sparse representations. One of such modification is Non-Negative Orthogonal Matching Pursuit(NNOMP), which satisfies non-negative constraint by selecting positive coefficients and using a non-negative optimization technique. However, since it consumes the extra computational costs, the most significant benefit of OMP—fast implementation—cannot be exhibited. Therefore, author proposed a fast implementation of NNOMP similar with canonical fast OMP based on QR decomposition method, called Fast Non-Negative Orthogonal Matching Pursuit(FNNOMP).**

## I. INTRODUCTION

*A. Sparse Representation of Signals and OMP Algorithm.*

Using an overcomplete dictionary matrix $\mathbf{\Phi} \in \mathcal{R}^{M \times N}$ that contains $N$ prototype signal-atoms as columns $\phi_i$'s, a measurement $\mathbf{y} \in \mathcal{R}^{M}$ can be approximated as a sparse linear combination of these atoms.

$$\mathbf{y} \approx \mathbf{\Phi x}$$

The greedy sparse approximation algorithms are generally computationally low cost. One simple greedy algorithm is Matching Pursuit(MP). It iteratively adds the most correlated atom of dictionary to set of selected elements. However, it does not give the best representation for chosen atoms and may re-select atom already selected in the last iteration.

Orthogonal Matching Pursuit(OMP) is an algorithm that compensate these drawbacks. It performs the orthogonal projection of measurement $\mathbf{y}$ onto the support set $s$ for each iteration. i.e.,

$$\tilde{\mathbf{x}}_s := \arg\min_{\mathbf{x}_s} \left\| \mathbf{y} - \mathbf{\Phi}_s \mathbf{x}_s \right\|_2$$

where $\mathbf{x}_s$ and $\mathbf{\Phi}_s$ are coefficient vector and sub-dictionary for the support set $s$, respectively. From this extra computation, sparse representation of every iteration becomes the best representation using current support set $s_i$. In the other hands, since the next residual $\mathbf{r}_i := \mathbf{y} - \mathbf{\Phi}_{s_i} \mathbf{x}_{s_i}$ is orthogonal to $s_i$, atoms which are already selected cannot be re-selected at later iterations.

*B. Non-Negative OMP(NNOMP)*

Canonical NNOMP has two different points from OMP, which are that it selects the most positively correlated atom and it performs Non-Negative Least Square(NNLS) program instead of Least Square(LS) program for best representation.

Table 1. Difference between OMP and Canonical NNOMP

| | OMP | Canonical NNOMP |
|---|---|---|
| Atom Selection | $s_i \leftarrow s_{i-1} \bigcup k$ s.t. $k = \arg\max_{k \in \{1,2,\ldots N\}} \left| \boldsymbol{\phi}_k^T \mathbf{r}_{i-1} \right|$ | $s_i \leftarrow s_{i-1} \bigcup k$ s.t. $k = \arg\max_{k \in \{1,2,\ldots N\}} \boldsymbol{\phi}_k^T \mathbf{r}_{i-1}$ and $\boldsymbol{\phi}_k^T \mathbf{r}_{i-1} > 0$ |
| LS program | $\tilde{\mathbf{x}}_s = \arg\min_{\mathbf{x}_s} \left\| \mathbf{y} - \mathbf{\Phi}_s \mathbf{x}_s \right\|_2$ | $\tilde{\mathbf{x}}_s = \arg\min_{\mathbf{x}_s \geq 0} \left\| \mathbf{y} - \mathbf{\Phi}_s \mathbf{x}_s \right\|_2$ |

However, in this implementation, selected positively correlated atom in an iteration may force the coefficient in last iteration becomes zero although that atom still remains in support set. Such algorithm efficiency-reducing problem is caused by dividing the atom selection and NNLS as separate tasks.

## II. FAST ORTHOGONAL MATCHING PURSUIT (FOMP)

### *A. Expensive Computational Cost of Standard OMP*

In the standard OMP, we solve the LS at each iteration, which can be solved by Moore-Penrose pseudo-inverse of $\mathbf{\Phi}_s$. i.e.,

$$\tilde{\mathbf{x}}_s = \mathbf{\Phi}_s^{\dagger}\mathbf{y} = (\mathbf{\Phi}_s^T\mathbf{\Phi}_s)^{-1}\mathbf{\Phi}_s^T\mathbf{y}$$

At $k$-th iteration, $|s| = k$ and calculation of $\mathbf{\Phi}_s^{\dagger}$ needs a matrix inversion of $\mathbf{\Phi}_s^T\mathbf{\Phi}_s \in \mathcal{R}^{k\times k}$, which costs $\mathcal{O}(k^3)$ of computation. To reduce this computational cost, we can use the QR factorization.

### *B. Fast Implementation using QR Factorization*

Let $\mathbf{\Phi}_s = \mathbf{\Psi}_s\mathbf{R}_s$ be the QR factorization where $\mathbf{\Psi}_s$ is column orthonormal and $\mathbf{R}_s$ is upper-triangle with positive diagonal elements. Then we can solve

$$\tilde{\mathbf{z}}_s = \arg\min_{\mathbf{z}_s} \left\|\mathbf{y} - \mathbf{\Psi}_s\mathbf{z}_s\right\|_2$$

*where*

$$\mathbf{z}_s = \mathbf{R}_s\mathbf{x}_s$$

instead of solving $\tilde{\mathbf{x}}_s = \arg\min_{\mathbf{x}_s} \left\|\mathbf{y} - \mathbf{\Phi}_s\mathbf{x}_s\right\|_2$. Since $\mathbf{\Psi}_s$ is orthonormal and $\mathbf{R}_s$ is upper-triangle, computation of $\tilde{\mathbf{z}}_s = \mathbf{\Psi}_s^T\mathbf{y}$ and $\tilde{\mathbf{x}}_s = \mathbf{R}_s^{-1}\tilde{\mathbf{z}}_s$ are much simpler than that of $\tilde{\mathbf{x}}_s = \mathbf{\Phi}_s^{\dagger}\mathbf{y}$.

But the algorithm would not be fast if the calculation of $\mathbf{\Psi}$, $\mathbf{R}$, and $\mathbf{R}^{-1}$ could not be done efficiently. Here, by using the Graham-Schmidt (GS) procedure, we can quickly calculate $\mathbf{\Psi}$ and $\mathbf{R}$ based on iterative update. At the $k$+1-th iteration,

$$\mathbf{\Psi}_{k+1} = \begin{bmatrix}\mathbf{\Psi}_k & \boldsymbol{\psi}_{k+1}\end{bmatrix},\ \mathbf{R}_{k+1} = \begin{bmatrix}\mathbf{R}_k & \mathbf{v} \\ 0 & \boldsymbol{\mu}\end{bmatrix} \text{ and } \mathbf{R}_{k+1}^{-1} = \begin{bmatrix}\mathbf{R}_k^{-1} & -\dfrac{\mathbf{R}_k^{-1}\mathbf{v}}{\mu} \\ 0 & \dfrac{1}{\mu}\end{bmatrix}$$

where $\boldsymbol{\psi}_{k+1} = \dfrac{\mathbf{q}_{k+1}}{\left\|\mathbf{q}_{k+1}\right\|}$, $\mathbf{q}_{k+1} = (\mathbf{I} - \mathbf{\Psi}_k\mathbf{\Psi}_k^T)\boldsymbol{\phi}_{k+1}$, $\mathbf{v} = \mathbf{\Psi}_k^T\boldsymbol{\phi}_{k+1}$ and $\mu = \left\|\mathbf{q}_{k+1}\right\|$. In this setting, we do not need to keep track $\mathbf{x}_k$ and $\mathbf{R}_k$ in the intermediate iteration. By updating $\mathbf{\Psi}_k$, $\mathbf{R}_k^{-1}$, and

$\mathbf{z}_k$ at each iteration and calculating $\mathbf{x}_K = \mathbf{R}_K^{-1}\mathbf{z}_K$ at the last iteration $K$, we can make OMP faster for the cases that $K$ is large.

However, this FOMP implementation cannot be directly applied for non-negative problem because we may get negative $\mathbf{x}_K$ from positive $\mathbf{z}_K$

## III. FAST NON-NEGATIVE ORTHOGONAL MATCHING PURSUIT (FNNOMP)

Let $\mathbf{\Phi}_k\mathbf{x}_k = \mathbf{\Psi}_k\mathbf{z}_k$ be the best approximation of $\mathbf{y}$ with the non-negative coefficient in $k$-th iteration. In the $k+1$th iteration, we have

$$\begin{aligned}\mathbf{\Psi}_{k+1}\mathbf{z}_{k+1} &= \mathbf{\Psi}_k\mathbf{z}_k + z_{k+1}\boldsymbol{\psi}_{k+1} \\ &= \mathbf{\Phi}_k\mathbf{x}_k + z_{k+1}\boldsymbol{\psi}_{k+1}\end{aligned}.$$

From $\mathbf{\Psi}_{k+1}\mathbf{R}_{k+1} = \mathbf{\Phi}_{k+1}$, we can get

$$\begin{aligned}\mathbf{\Psi}_{k+1} &= \mathbf{\Phi}_{k+1}\mathbf{R}_{k+1}^{-1} \\ \boldsymbol{\psi}_{k+1} &= \mathbf{\Phi}_{k+1}\boldsymbol{\gamma}_{k+1}\end{aligned}$$

where $\boldsymbol{\gamma}_{k+1} = \begin{pmatrix}\gamma_1 & \gamma_2 & \cdots & \gamma_{k+1}\end{pmatrix}^T$ is $k+1$ column of $\mathbf{R}_{k+1}^{-1}$. Then

$$\begin{aligned}\mathbf{\Psi}_{k+1}\mathbf{z}_{k+1} &= \mathbf{\Phi}_k\mathbf{x}_k + \boldsymbol{\psi}_{k+1}z_{k+1} \\ &= \mathbf{\Phi}_k\mathbf{x}_k + \mathbf{\Phi}_{k+1}\boldsymbol{\gamma}_{k+1}z_{k+1} \\ &= \mathbf{\Phi}_k(\mathbf{x}_k + \boldsymbol{\gamma}_k z_{k+1}) + \cancel{\boldsymbol{\phi}_{k+1}}z_{k+1}\gamma_{k+1} \\ &= \mathbf{\Phi}_{k+1}\mathbf{x}'_{k+1}\end{aligned},$$

where $\boldsymbol{\gamma}_k = \begin{pmatrix}\gamma_1 & \gamma_2 & \cdots & \gamma_k\end{pmatrix}^T$. Here, we can see that

$$x'_i = x_i + \gamma_i z_{k+1} \geq 0 \quad \text{for all} \quad i \in \{1, 2, \ldots, k\}$$

and

$$x'_{k+1} = z_{k+1}\gamma_{k+1} \geq 0$$

to keep the non-negative restriction. From the update rule of $\mathbf{R}_{k+1}^{-1}$, $\gamma_{k+1} = \frac{1}{\mu} = \frac{1}{\|\mathbf{q}_{k+1}\|}$ is always positive. $x_i$'s are also positive at all times and $\gamma_i$ may be positive, may be negative. Appling these, we can specify the range of $z_{k+1}$ to get non-negative representation as

$$0 \le z_{k+1} \le z^{th} := \begin{cases} +\infty & \text{where } \forall \gamma_i \ge 0 \text{ for } i \in \{1,2,\ldots,k\} \\ \min\limits_{\gamma_i<0} \left| \dfrac{x_i}{\gamma_i} \right| & \text{where } \exists \gamma_i < 0 \end{cases} .$$

For finding the most positively correlated atoms, we should calculate $\zeta_i = \min\{\boldsymbol{\psi}_i^T \mathbf{r}, z_i^{th}\}$ and select $z_{k+1} = \max\{\zeta_1, \zeta_2, \ldots\}$ for all $i \in \{1,2,\cdots,N\} \setminus s_k$, where $s_k$ is the support set of last iteration. In fact, since $\boldsymbol{\psi}_i^T \mathbf{r} \ge \boldsymbol{\psi}_j^T \mathbf{r}$ where $\boldsymbol{\phi}_i^T \mathbf{r} \ge \boldsymbol{\phi}_j^T \mathbf{r}$, we do not need to calculate $\zeta_i$ for all $i$. Instead, we can investigate from the index which has largest $\boldsymbol{\phi}_i^T \mathbf{r}$ value to descent order, while keep tracking the index and value of shrunk by $z_i^{th}$. Here is the pseudo code and decision rule of FNNOMP.

**Algorithm 2** Fast Non-Negative Orthogonal Matching Pursuit.

**initialisation:** $s = z_0 = \emptyset$, $k = 0$ and $\mathbf{r}_0 = \mathbf{y}$
**while** $k < K$ & $\max(\boldsymbol{\Phi}^T \mathbf{r}_k) > 0$ **do**
  $(\boldsymbol{\zeta}, \boldsymbol{\iota}) \leftarrow \text{sort}_{\downarrow}(\boldsymbol{\Phi}^T \mathbf{r}_k)$
  $p \leftarrow 1$
  $p^c \leftarrow 1$
  $z^c \leftarrow 0$
  **while** $\sim Terminate \& p < N$ **do**
    $z^t$ from (5)
    $z \leftarrow \boldsymbol{\psi}_{\iota(p)}^T \mathbf{r}_k : \boldsymbol{\psi}_{\iota(p)} = \mathbf{q}/\|\mathbf{q}\|_2$, $\mathbf{q} = (\mathbf{I} - \boldsymbol{\Psi}_k \boldsymbol{\Psi}_k^T)\boldsymbol{\phi}_{\iota(p)}$
    Update based on Table I
  **end while**
  $s \leftarrow s \cup \iota(p)$
  Update $\boldsymbol{\Psi}$ and $\mathbf{R}^{-1}$
  $\mathbf{z}_{k+1} \leftarrow [\mathbf{z}_k; z_{k+1}]$
  $\mathbf{r}_{k+1} \leftarrow \mathbf{r}_k - z_{k+1}\boldsymbol{\psi}_{k+1}$
  $k \leftarrow k + 1$
**end while**
**output:** $\mathbf{x}|_s \leftarrow \mathbf{R}^{-1}\mathbf{z}_K$

TABLE I
DECISION RULES TO GUARANTEE POSITIVITY OF THE COEFFICIENTS

| If | Then |
|---|---|
| $0 < z \le z^t,\ z > z^c$ | $z_{k+1} \leftarrow z$, Terminate |
| $0 < z \le z^t,\ z \le z^c$ | $z_{k+1} \leftarrow z^c, p \leftarrow p^c$, Terminate |
| $z > z^c \ge z^t$ | $p \leftarrow p + 1$ |
| $z^c \ge z > z^t$ | $z_{k+1} \leftarrow z^c, p \leftarrow p^c$, Terminate |
| $z > z^t > z^c$ | $z^c \leftarrow z^t, p^c \leftarrow p,\ p \leftarrow p + 1$ |
| $z < 0$ | Terminate |

*A. Computational Complexity*

**i. CNNOMP with positively correlated atom selection and NNLS**

Atom selection: $\mathcal{O}(MN)$, each iteration.

NNLS: $\mathcal{O}(LMk^2)$ for $k$-th iteration.

**Overall complexity:** $\mathcal{O}(NMK + LMK^3)$

where $\mathcal{O}(L) \approx \mathcal{O}(k)$ and $K$ is the maximum number of iteration.

**ii. FNNOMP**

Atom selection and sorting: $\mathcal{O}(MN + N \log N)$, worst case.

Calculating threshold $z_i^{th}$: $\mathcal{O}(Mk + k^2)$, worst case.

Calculating $\boldsymbol{\psi}_i^T \mathbf{r}$: $\mathcal{O}(Mk)$

Overall complexity for updating $z_{k+1}$: $\mathcal{O}(PMk + Pk^2)$ where inner loop terminates at $P$-th iteration

Finding $\mathbf{x}_K$ from $\mathbf{R}_K^{-1}\mathbf{z}_K$: $\mathcal{O}(K^2)$

**Overall complexity:** $\mathcal{O}(MNK + NK \log N + PMK^2 + PK^3)$**.**

Assuming that $K$ is large so that $\mathcal{O}(M) \approx \mathcal{O}(K)$, the complexity of FNNOMP becomes $\mathcal{O}(NK^2 + NK \log N + PK^3)$, roughly $\mathcal{O}(K^3)$ for small $N$ and $K$, while that of CNNOMP becomes $\mathcal{O}(K^5)$.

## IV. SIMULATIONS

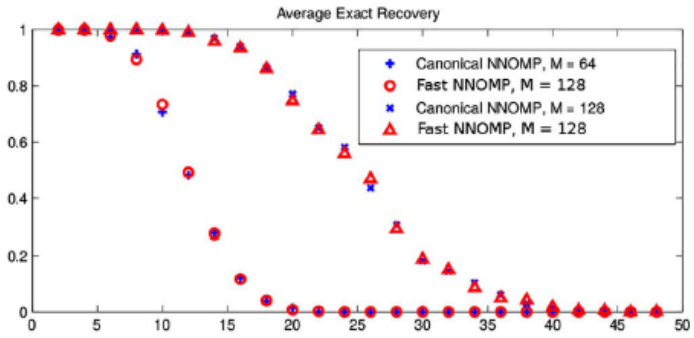
Simulation Setting:

$\boldsymbol{\Phi} \in \mathcal{R}^{M \times N}$ is i.i.d. Gaussian random matrix.

$\mathbf{x} \in \mathcal{R}^N$ is generated using Gaussian-Bernoulli model, i.e., uniform random support and Normal distributed non-zero coefficients.

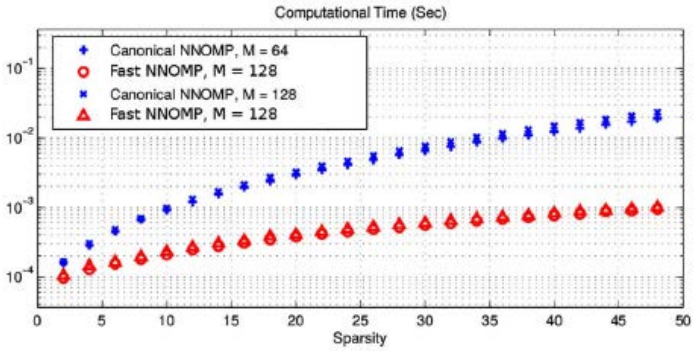
1000 times simulation

*A. Probability of Exact Recovery*



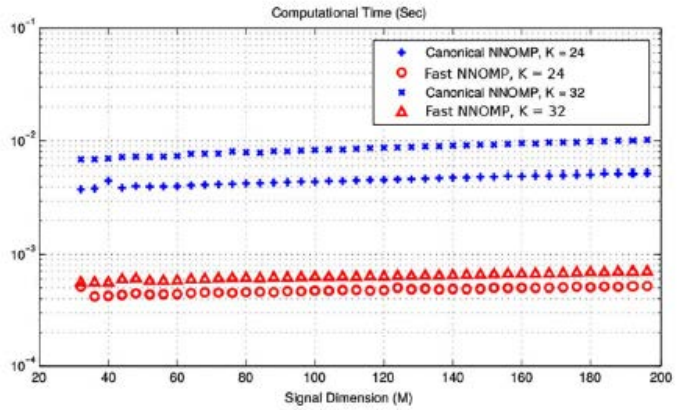

For fixed $N = 256$, exact recovery probability is almost same.

*B. Computational Time versus Sparsity*



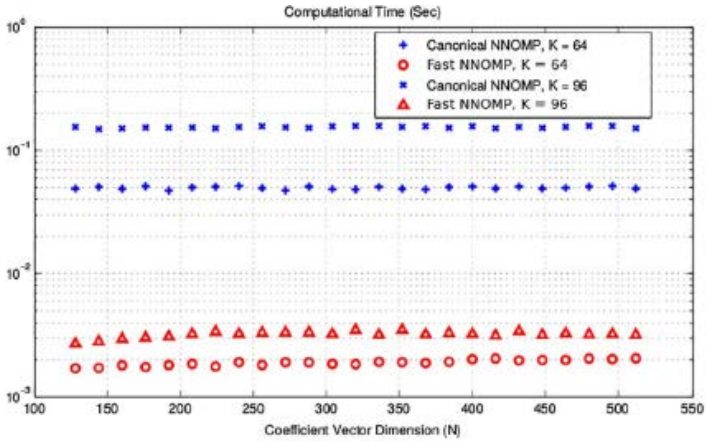

For fixed $N = 256$, computational time of FNNOMP is much shorter as sparsity grows.

*C. Computational Time versus Length of Measurement*




For fixed $N = 256$ and $K = 24$ and 32, computational time of both algorithms is slowly increasing with $M$. But still FNNOMP much faster than CNNOMP.

### *D. Computational Time versus Signal Length*




For fixed $M = 128$ and $K = 64$ and 96, the computational time of CNNOMP is almost constant, while that of FNNOMP increases very slowly with $N$ .

## V. CONCULSION

By using QR factorization, author proposed FNNOMP. FNNOMP and CNNOMP has similar recovery performance but computational cost of FNNOMP is significantly low than CNNOMP, especially under the large sparsity and small signal dimension condition.

## VI. DISCUSSION

## Appendix