Sparse Solutions to Linear Inverse Problems

Yuzhe Jin
Outline

- Intro/Background
- Two types of algorithms
  - Forward Sequential Selection Methods
  - Diversity Minimization Methods
- Experimental results
- Potential application to Speech
Background: Sparseness

- A large vector with only a very small # of non-zero entries
Why sparse? In what scenarios?

- Bio-magnetic inverse problem
- Band-limited extrapolation, especially for Speech signals
- Direction-of-arrival estimation
- Channel equalization, Echo cancellation
- Image restoration
- ...

- Represent a signal of interest using the minimum number of vectors from an over-completed dictionary.
Problem Description

A measurement

A “dictionary” with each column as a codeword

A sparse source, only a few of the entries are non-0
Problem Description

\[
\begin{bmatrix}
    b
\end{bmatrix} = \begin{bmatrix}
    a_1 & a_2 & a_3 & \ldots & a_{n-2} & a_{n-1} & a_n
\end{bmatrix} \begin{bmatrix}
    x_1 \\
    x_2 \\
    x_3 \\
    \cdot \\
    \cdot \\
    x_{n-2} \\
    x_{n-1} \\
    x_n
\end{bmatrix}
\]

A measurement  
A “dictionary” with each column as a codeword  

\[
b = \sum x_i \cdot a_i
\]

Sparse source, only a few of the entries are non-0
With multiple measurements

\[ B = AX \]

\[ \begin{bmatrix} b_1 & \cdots & b_m \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 & \cdots & a_{n-2} & a_{n-1} & a_n \end{bmatrix} \]

Multiple sources with same sparsity profile
With multiple measurements

\[ \begin{bmatrix} b_1 & \ldots & b_m \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 & \ldots & a_{n-2} & a_{n-1} & a_n \end{bmatrix} \]

Multiple measurements

\[ \mathbf{B} = \mathbf{A} \mathbf{X} \]

Multiple sources with same sparsity profile
Then, add noise to the observations

- New Model:

\[ B = AX + N \]

- Tradeoff between fit and sparsity of solution

- Modeling error
- Noise present
Then, add noise to the observations

- **New Model:**
  
  \[ B = AX + N \]

- **Tradeoff between fit and sparsity of solution**
  
  \[ \|AX - B\| \]

- Modeling error
- Noise present
Type I: Forward Sequential Selection

\[
\begin{bmatrix}
  b \\
  \vdots \\
\end{bmatrix} = \begin{bmatrix}
  a_1 & a_2 & a_3 & \cdots & a_{n-2} & a_{n-1} & a_n \\
  \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
\end{bmatrix} \begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3 \\
  \vdots \\
  x_{n-2} \\
  x_{n-1} \\
  x_n \\
\end{bmatrix}
\]

Known       Known

Unknown
Type I: Forward Sequential Selection

\[
\begin{bmatrix}
    b \\
\end{bmatrix} = \begin{bmatrix}
    a_1 & a_2 & a_3 & \cdots & a_{n-2} & a_{n-1} & a_n \\
\end{bmatrix} \begin{bmatrix}
    x_1 \\
    x_2 \\
    x_3 \\
    \vdots \\
    x_{n-2} \\
    x_{n-1} \\
    x_n \\
\end{bmatrix}
\]
Type I: Forward Sequential Selection

\[
\begin{bmatrix}
0.1 & 0.01 & 10 & 0.02 & 0.005 & 0.3 \\
\end{bmatrix}
\]

\[
b = \begin{bmatrix}
a_1 & a_2 & a_3 & \cdots & a_{n-2} & a_{n-1} & a_n \\
\end{bmatrix}
\]

\[
x_1
\]

\[
x_2 \\
x_3
\]

\[
x_{n-2}
\]

\[
x_{n-1}
\]

\[
x_n
\]
How to compute residual?

- **Basic Matching Pursuit**
  - Remove contribution from the selected vector
  \[
P_{a_{kp}}^{\perp} \mathbf{b}_{p-1}
\]
  Project onto one direction

- **Orthogonal Matching Pursuit**
  - Remove contribution from the selected subspace
  \[
P_{S_{p}}^{\perp} \mathbf{b}_{p-1} \quad S_{p} = [S_{p-1}, \mathbf{a}_{kp}]
\]
  Project onto a sub-space
Another variation: Order Recursive Matching Pursuit

- Most like Orthogonal Matching Pursuit
- Main Difference:
  - The vector is selected by

\[
  k_p = \arg \max_k \frac{<b, a_k>}{\|a_k^{(p-1)}\|^2}
\]

Normalization term

Correction:

- Remove the contribution from vectors found previously
- Normalized inner product (since the codewords are not orthonormal)
Type II: Diversity Minimization

- Metric of Sparseness
  \[ E^{(p)}(x) = \sum_{i=1}^{n} |x[i]|^p \]

  - \( p = 2 \): Like 2-norm
    - Commonly used in engineering solutions
    - Nothing about sparsity
  - \( p = 0 \): a count on the non-zero entries in \( x \)
    - Direct measurement on sparseness
    - Very hard to solve, exhaustive search, NP-hard, combinatorial…
  - **What if \( p \) takes some value in between?**
    - In practice, setting \( p \) to some value between 0.8 ~ 1.0 gives good tradeoff between computational complexity and quality of sparse solution
Now, we have a cost function. What’s Next?

- Gradient Descent
  - FOCUSS-class Algorithms
    - FOCUSS
    - Regularized FOCUSS
      - Add in a regularization term to improve matrix condition for computing its inverse
      - Introduce bias, tradeoff between bias and quality of convergence
Experiments Setup

- Different number of measurements
- Different SNR
- Compare performances across 4 algorithms
  - M-OMP, M-ORMP, M-FOCUSS, R-M-FOCUSS
- Test:
  - If an algorithm correctly identifies the non-zero positions, **Percentage Success**
  - The MSE between the recovered vector and the ground truth, **MSE**
Experimental results

- SNR=20dB
- SNR=30dB
- SNR=40dB
- SNR=50dB

MSE

Comparison of M-OMP, M-ORMP, M-FOCUSS, and R-M-FOCUSS algorithms.
Experimental results

MSE vs SNR for different values of L:
- L=1
- L=3
- L=5

% Success vs SNR for different values of L:
- L=1
- L=3
- L=5

Methods compared:
- M-OMP
- M-ORMP
- M-FOCUSS
- R-M-FOCUSS
Applications to Speech

- Speech: Sparse in frequency domain

- Extrapolation

- Compression
Conclusion

- These algorithms are able to explore sparse linear inverse problems efficiently. They are robust to signal contaminated by noise.

- Speech signal is sparse in certain transform domain. We can apply this technique to speech.

