RBF by OMP approach

Working, Intuition and implementation

Linear Algebra

Matrix equation → $\mathbf{Ax} = \mathbf{b}$

$$\begin{array}{r} \text{Equation \# 1} \rightarrow \\ \text{Equation \# 2} \rightarrow \\ \text{Equation \# 3} \rightarrow \end{array} \begin{bmatrix} 1 & 5 & 3 \\ 2 & 8 & 6 \\ 4 & 9 & 7 \end{bmatrix} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}}_{\text{Three unknowns}} = \begin{bmatrix} 10 \\ 12 \\ 6 \end{bmatrix}$$

Linear Algebra

What if !!!

(# of equations) ≠ (# of unknowns)

$$\begin{bmatrix} 1 & 5 & 3 \\ 2 & 8 & 6 \\ 4 & 9 & 7 \\ 3 & 2 & 4 \\ 10 & 5 & 7 \\ 5 & 1 & 1 \\ 6 & 3 & 8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 10 \\ 12 \\ 6 \\ 4 \\ 24 \\ 8 \\ 20 \end{bmatrix} \Rightarrow \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1.46919879 \\ -0.16811539 \\ 1.19239733 \end{bmatrix}$$

```
x = np.linalg.lstsq(A,b)
```

Solve using Least Squares

$$\mathbf{A}^{+}\boldsymbol{x} \approx \mathbf{b}$$

Linear Algebra

What if !!!

(# of equations) ≠ (# of unknowns)

A⁺ is pseudoinverse

$$\mathbf{A}^{+}\boldsymbol{x} \approx \mathbf{b}$$

$$\begin{bmatrix} 1 & 5 & 3 \\ 2 & 8 & 6 \\ 4 & 9 & 7 \\ 3 & 2 & 4 \\ 10 & 5 & 7 \\ 5 & 1 & 1 \\ 6 & 3 & 8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 10 \\ 12 \\ 6 \\ 4 \\ 24 \\ 8 \\ 20 \end{bmatrix}$$

Overdetermined system

Orthogonal Matching Pursuit algorithm

$$\mathbf{A}^{+}\boldsymbol{x} \approx \mathbf{b}$$

$$y \approx \mathbf{A}^{+}\boldsymbol{x}$$

Column signal or Signal

Dictionary or measuring matrix or sensing matrix

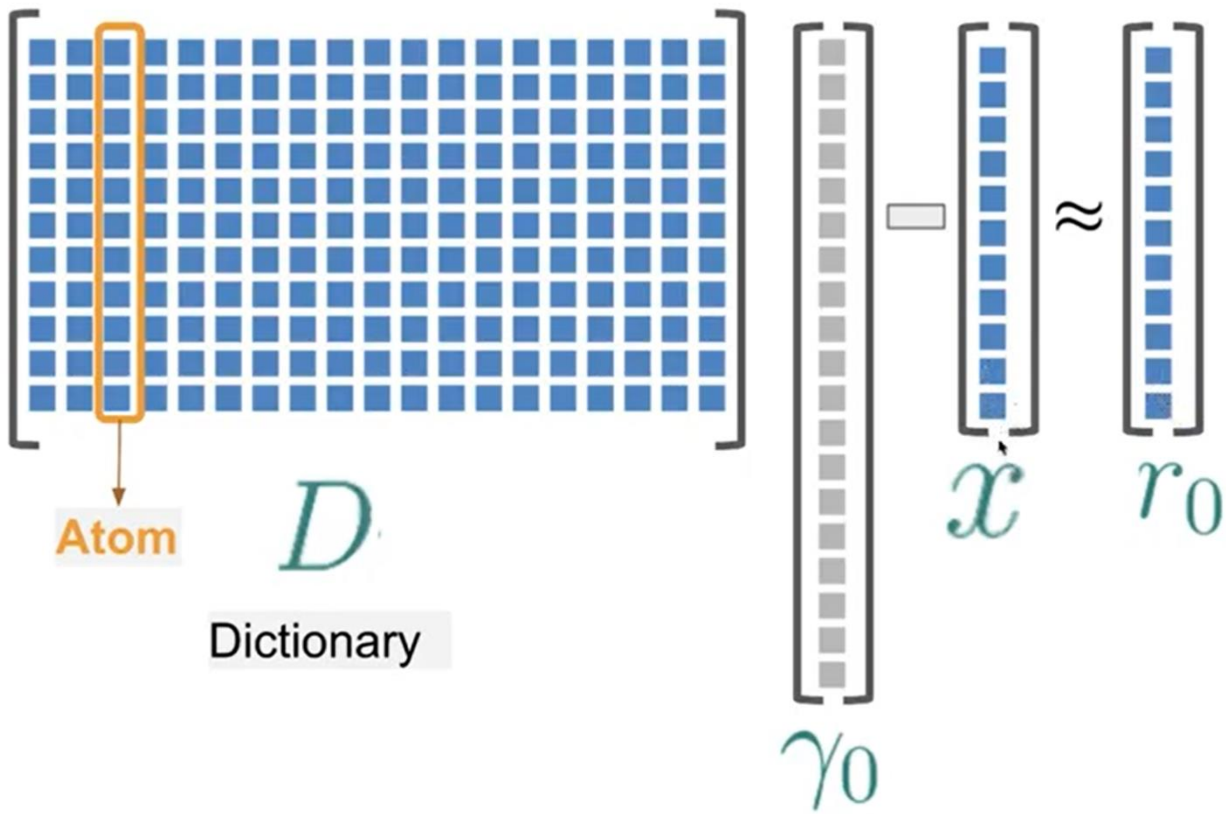
Paper notation → $x \approx D\gamma$

Sparse approximation or sparse representation

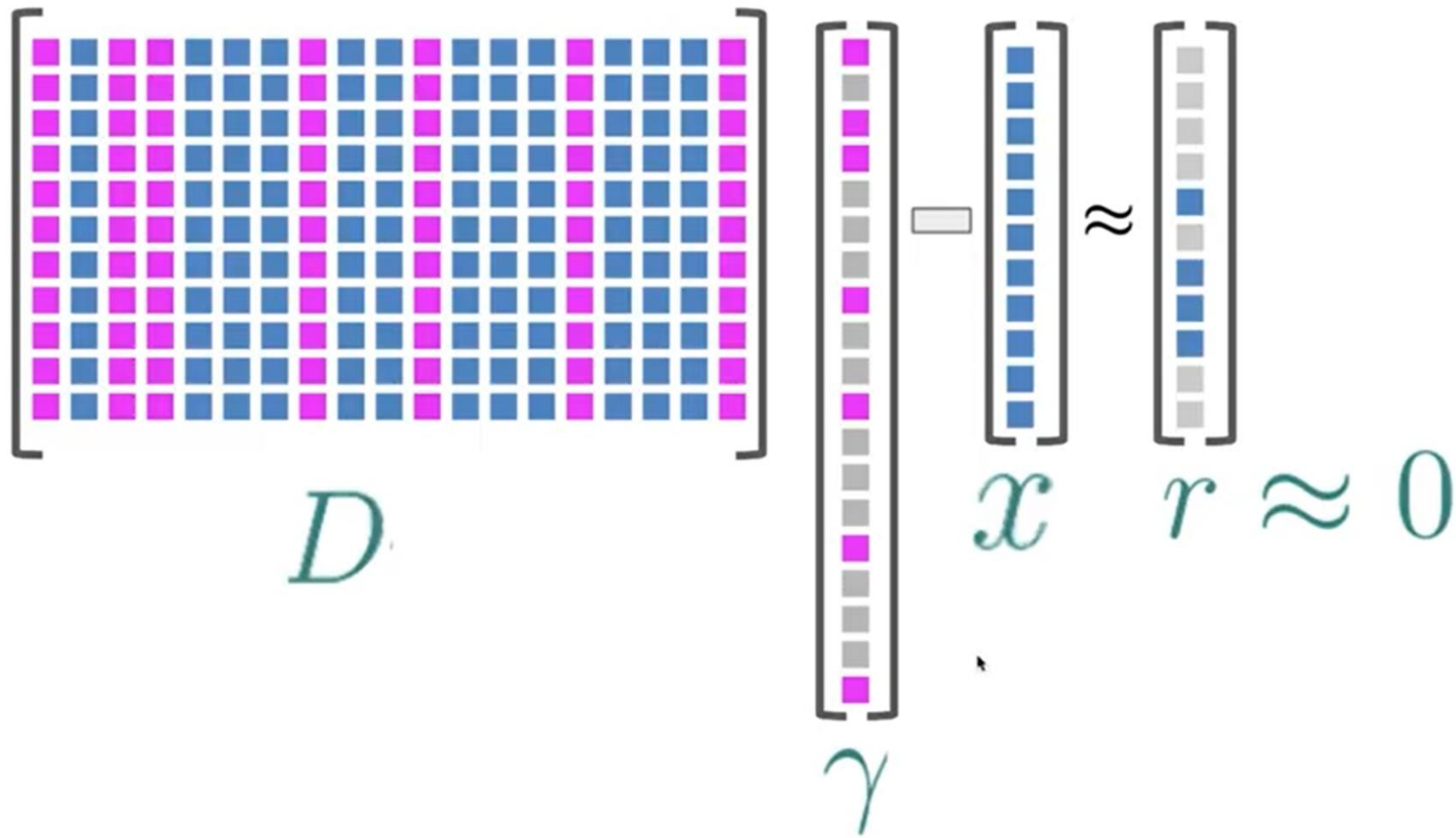
Scikit-learn notation → $y \approx Xw$

Ron Rubinstein et.al. Efficient Implementation of the K-SVD Algorithm using Batch Orthogonal Matching Pursuit. Technion
Source: https://www.cs.technion.ac.il/~ronrubin/Publications/KSVD-OMP-v2.pdf

Orthogonal Matching Pursuit algorithm



Orthogonal Matching Pursuit algorithm



Orthogonal Matching Pursuit algorithm

Input: Dictionary $\mathbf{D}$, signal x, target sparsity K or target error ϵ
Output: Sparse representation γ such that $x \approx D\gamma$

Initialize: Set $I := ()$, $r := x$, $\gamma := 0$
while (stopping criterion not met) do

$\hat{k} := \underset{k}{argmax} \left| d_k^T r \right|$

Maximize correlation of residual vector r with columns of dictionary D

$x \approx D_I \gamma_I$
$\gamma_I \approx D_I^+ x$

$I := \left(I, \hat{k} \right)$

Indices of selected atoms or columns in D that maximize k above

$\gamma_I := (D_I)^+ x$

γ_I sparse representation of x

$\underset{\gamma}{argmin} \|x - D\gamma\|_2^2$

$r := x - D_I \gamma_I$
end while

What is Orthogonal Matching Pursuit?

Target fixed number of non-zero elements

$$\underset{\underset{\gamma}{w}}{\arg\min}\, ||\underset{x}{y} - \underset{D}{X}\underset{\gamma}{w}||_2^2$$

$$\text{subject to } ||\underset{\gamma}{w}||_0 \leq n_{\text{nonzero\_coefs}}$$

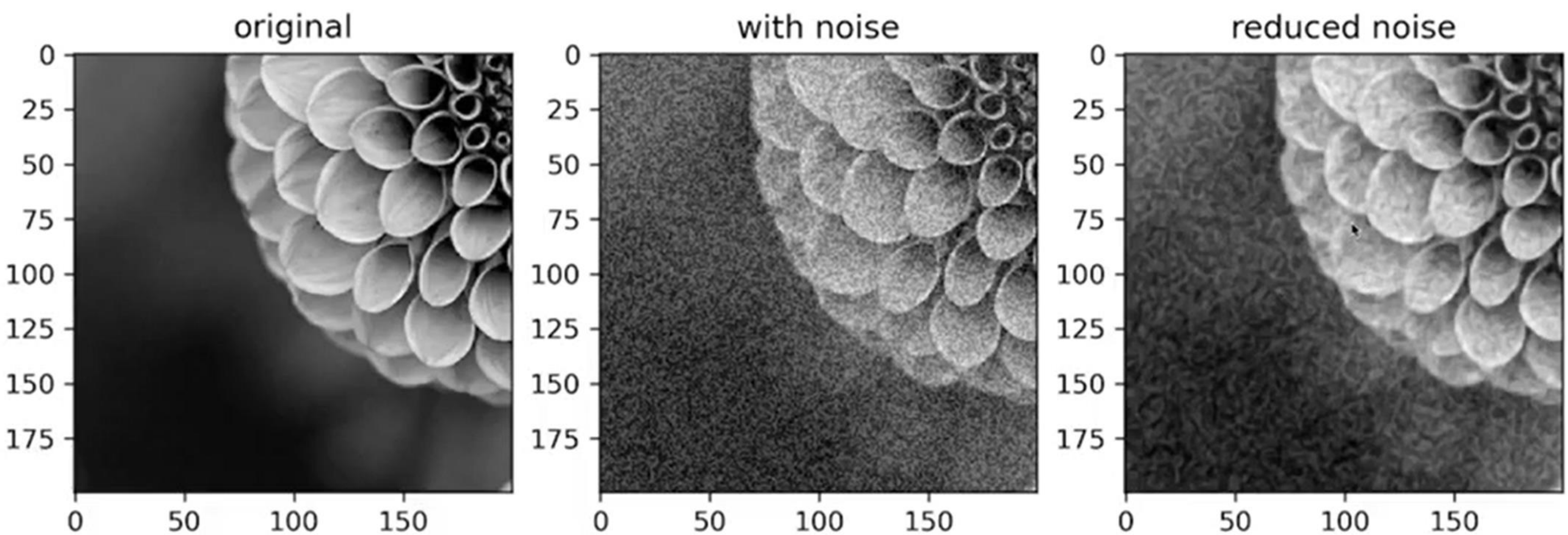
Target error or tolerance (tol)

$$\underset{\underset{\gamma}{w}}{\arg\min}\, ||\underset{\gamma}{w}||_0$$

$$\text{subject to } ||\underset{x}{y} - \underset{D}{X}\underset{\gamma}{w}||_2^2 \leq \text{tol}$$

Image denoising

Adapted from code at:
https://scikit-learn.org/stable/auto_examples/decomposition/plot_image_denoising.html



WHY COMPRESSIVE SENSING ??

- ✓ A random picture with
  - # resolution : 1920 x 1080
  - # pixel count : 20,73,600

- ✓ Image vector $x \in R^n$ where n = 20,73,600

$y \in R^m \quad m << n$

Gray scale image

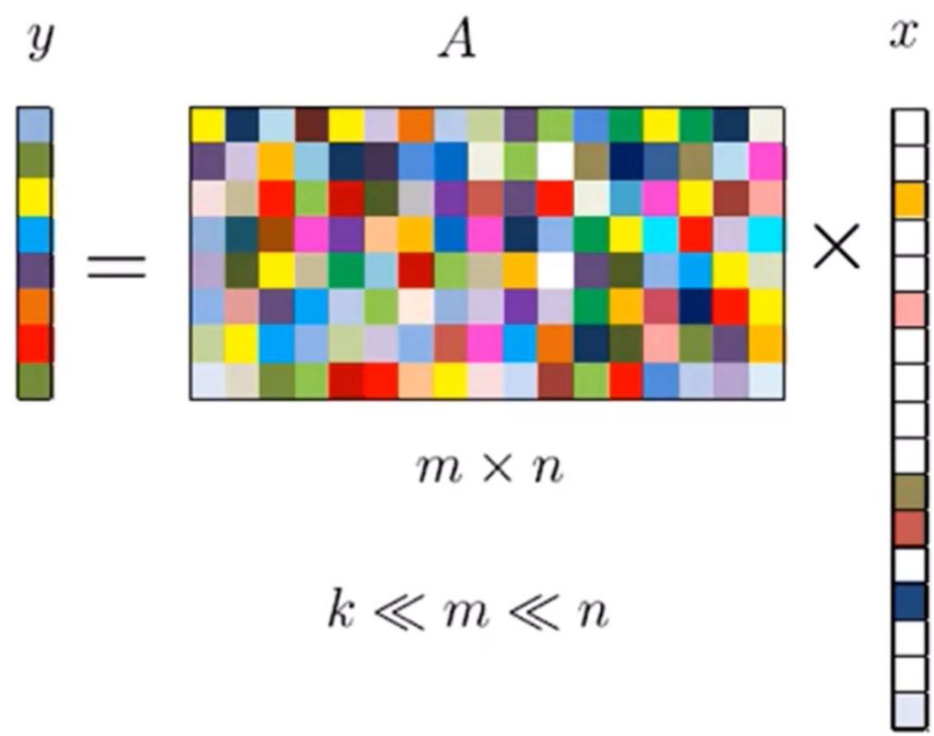
PROBLEM FORMULATION

- ✓ **Given an observation vector $y \in R^m$ and a matrix $A \in R^{m \times n}$, find $x \in R^n$ such that**

$$\boxed{y = Ax}$$

and x is k-sparse.

- ✓ **Here A is a fat matrix ($m \ll n$).**



SUMMARY

- OMP is an effective method to find the sparse approximation.
- It takes only k number of iterations to find the sparse approximation.
- OMP demands sparsity level of the original vector.

RBF Formulation

data-driven approach [3]. In the RBFs
solve a system of linear equations: f
corresponds to the measured force vec
RBF matrix defined over measured po
formation and $\mathbf{w} \in \mathbb{R}^N$ is a solution ve
weights of radial basis functions (kern
struct the RBF matrix, we implement
in [3], known as randomly selected- ort
suit (RS-OMP). This method first rand
N sample values to define an underd
$A \in \mathbb{R}^{K \times N}$ and a new force vector $\mathbf{f} \in$
matching pursuit (OMP), a widely kn
pressive sensing, is employed to find th
we define $p = (K/N) * 100$, then simil
RBF matrix is also trained on $p\%$ of the
criterion is defined by the number of rac

Since there are 100 decision trees in ou
we also train the RBF matrix A with 1
pares our random forest (RF) approach

gions of similar material behavior in the examined deformable objects. This is done by using a k-means clustering algorithm on the force-displacement curves obtained during quick exploratory manipulation. The method yields a set of exemplary contact locations that are representative for regions of similar mechanical behavior. The number of these positions can be adjusted according to desired error ranges. Then, in the second step, for each of these extracted locations an independent RBF reconstruction of the force response

tion times are too high. Creating an RBF representation of a less constrained interaction could take several hours. In addition, the previous methods also exhibit increased errors when only a small number of data samples are used (which is typical for greedy-like algorithms). In order to overcome these problems, we use a different formulation inspired by recent work in Compressive Sensing (see e.g. [4, 8]). We combine an ℓ^1-minimization with a random selection strategy. The new formulation is more robust to undersampling, as compared to similar methods. It also provides good performance in the construction of the RBF models, without any significant reduction in accuracy. The mentioned improvements are also demonstrated on test cases.

Thank you

RECOVERING SPARSE SIGNAL

- ✓ Recovering x given the observation y and the measurement matrix A.
- ✓ But A has non zero null space
- ✓ If x^* is a solution to this equation then any vector from $x^* + N(A)$ will also be the solution.
- ✓ Hence, recovering a unique x from A is impossible without any prior knowledge on x.
- ✓ Knowing that x is sparse, we try to recover the sparse solution by solving

$$\boxed{\begin{array}{c} x = \arg\min \ \|x`\|_0 \\ y = Ax` \end{array}}$$

where $\|x`\|_0$ is the number of nonzero entries of x.

APPROACH

One of the well known approach is GREEDY APPROXIMATION that generates an approximate solution to SMV by recursively building an estimate x^.

- **Orthogonal Matching Pursuit (OMP) is one such iterative algorithm that finds the sub-optimal solution**

 # Resolves big problem to small optimisation problems

 # In each iteration one non zero location of the sparse solution is found and the residual gets updated.

 # Uses k iterations to find k non zero locations of the sparse signal.

 # Finally computes the sparse signal approximation.

FINDING THE NON-ZERO INDEX OF 1-SPARSE SOLUTION

$$y = [c_1\, c_2\, c_3\, c_4\, c_5\, c_6\, c_7]\, [1\,0\,0\,4\,0\,0\,7]^T$$

$$y = (1.\, c_1 + 4.\, c_4) + (7.\, c_7)$$

$$\boxed{\gamma_i = \arg\max_j \frac{yTa_j}{\| a_j \|_2}}$$

a_j : j^{th} column of the matrix

λ_i : position of the non-zero entry in the 1-sparse x

The non zero location of x corresponds to that column of A which is highly correlated with y.

UPDATING RESIDUAL

$$\boxed{r_i = y - A_{\Lambda i} A_{\Lambda i}^{+} y}$$

where $A_{\Lambda i}$: Matrix containing columns specified by Λi

$$A_{\Lambda i}^{+} = [A_{\Lambda i}^{T} A_{\Lambda i}]^{-1} A_{\Lambda i}^{T}$$

$P_{A_{\Lambda i}} = A_{\Lambda i} A_{\Lambda i}^{+}$: projection matrix onto the column space of $A_{\Lambda i}$

$$\boxed{r_i = [\, I - P_{A_{\Lambda i}} \,]\, y}$$

Projects y onto the space which is orthogonal to the column space of $A_{\Lambda i}$

UPDATING RESIDUAL Contd.

$$y = [c_1\, c_2\, c_3\, c_4\, c_5\, c_6\, c_7]\ [1\ 0\ 0\ 4\ 0\ 0\ 7]^T$$

$$y = (1.\, c_1 + 4.\, c_4) + (7.\, c_7)$$

If $\Lambda i = (7,4)$

$r_i = (I - P_{A \Lambda i})\, y$

$= (I - P_{A \Lambda i})((1.\, c_1 + 4.\, c_4 + 7.\, c_7))$

$r_i = (I - P_{A \Lambda i})\, 1.\, c_1$

The effect of c_7 and c_4 has been removed from the vector y.

OMP ALGORITHM

- **Input**

  # An m x n measurement matrix A

  # An m dimensional data vector y

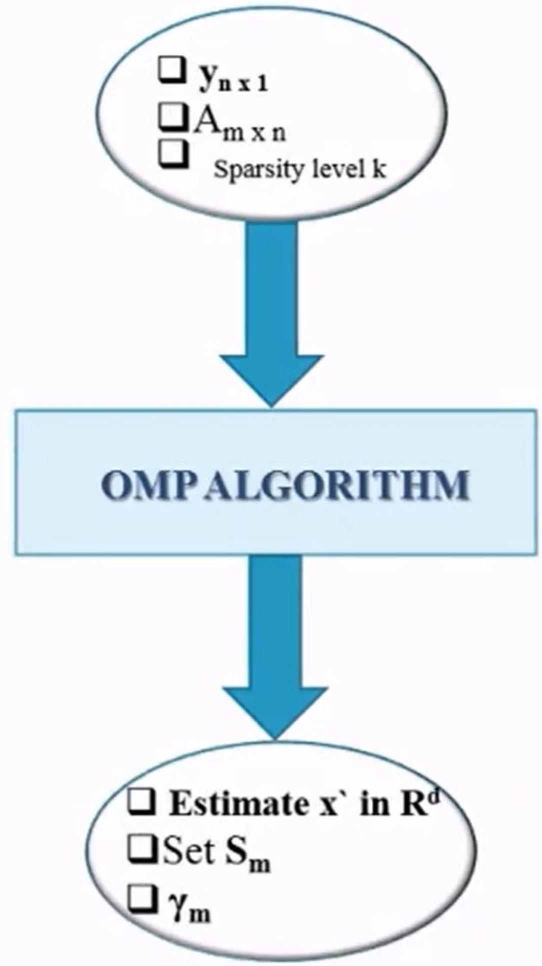
  # The sparsity level k of the signal x

- **Output**

  # An estimate x` in R^n for the signal x

  # The set Λ_k containing k elements from {1,2,3,..........n}

  # The n dimensional residual r_k



OMP ALGORITHM : PROCEDURE

- **Step 1: Initialize**

  # Iteration counter $i = 1$.

  # Residual $r_o = y$

  # Index set $\Lambda_o = \Phi$

- **Step 2: Find the index λ_i**

  # $\lambda_i = \arg\max_{j} \; r_{k-1}^{T} a_j$

  If the maximum occurs for multiple indices ,break the tie deterministically

- **Step 3: Augment the index set**

  # $\Lambda_i = \Lambda_i \cup \lambda_i$

OMP PROCEDURE Contd.

- **Step 4:**

  # Remove the effect of columns specified by Λi from the measurement vector

  # Update the residual $r_i = y - A_{\Lambda i} A_{\Lambda i}^{+} y$

- **Step 5:**

  #Increment i and repeat step 2 if $i < k$

- **Step 6:**

  # Calculate the sparse approximation

  # The estimate x` will have non zero indices at the components listed in Λ_k

  # The value of the estimate in the component λ_j equals j^{th} entry in $A_{\Lambda k}^{+} y$