

# HW 3 - CS754

---

Sahasra Ranjan - 190050102

Rahul Prajapat - 190050095

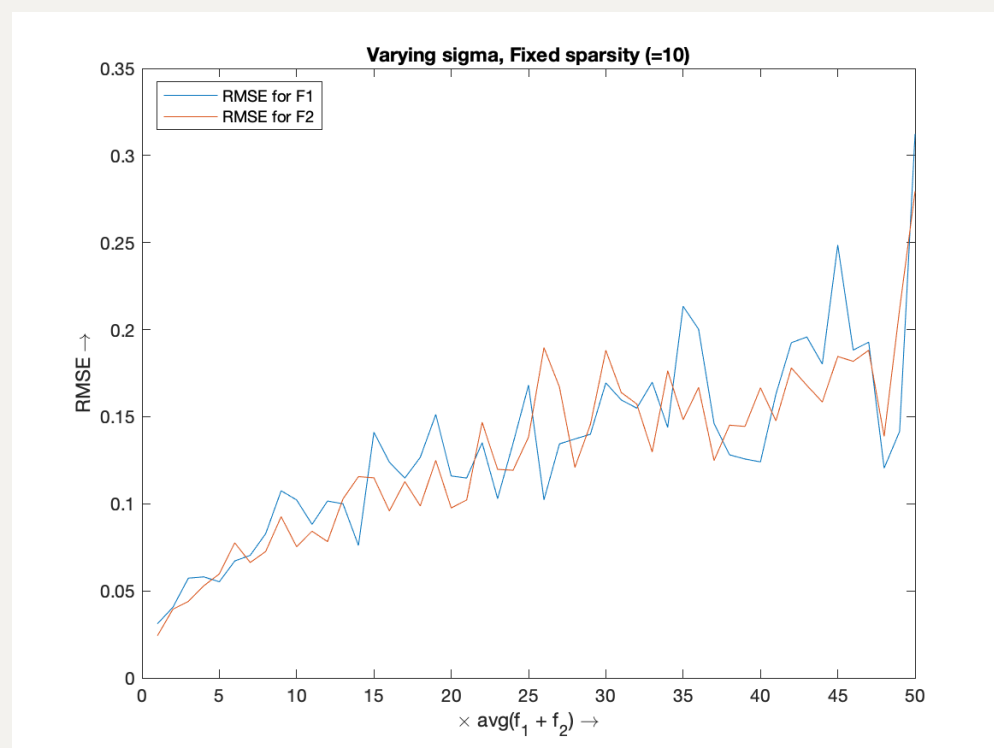
Q1

**Technique used for reconstruction:**

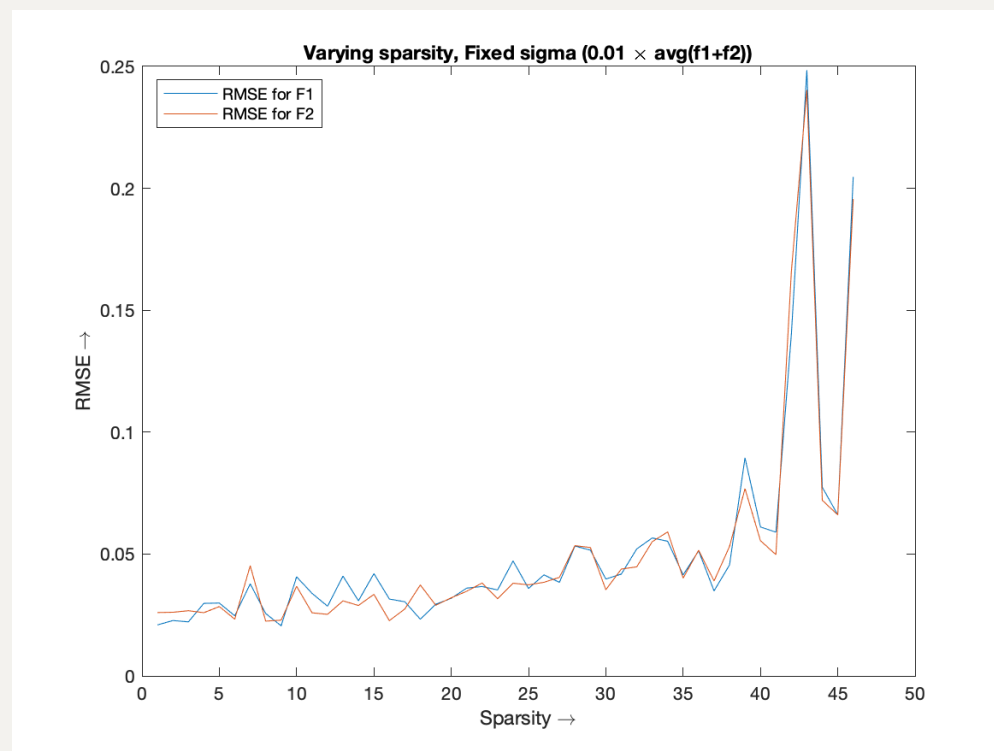
- Computed a larger basis matrix  $A = [A1 \ A2]$
- Using the  $l1\_ls$  solver in MATLAB, reconstructed the sparse signal  $f1, f2$  with basis  $A1, A2$  respectively.
- Parameter for the solver was selected using hit an trial method for many values. Out of which we selected the  $\epsilon = 10^{-3}$ .

**Varying  $\sigma$  and fixed sparsity:**

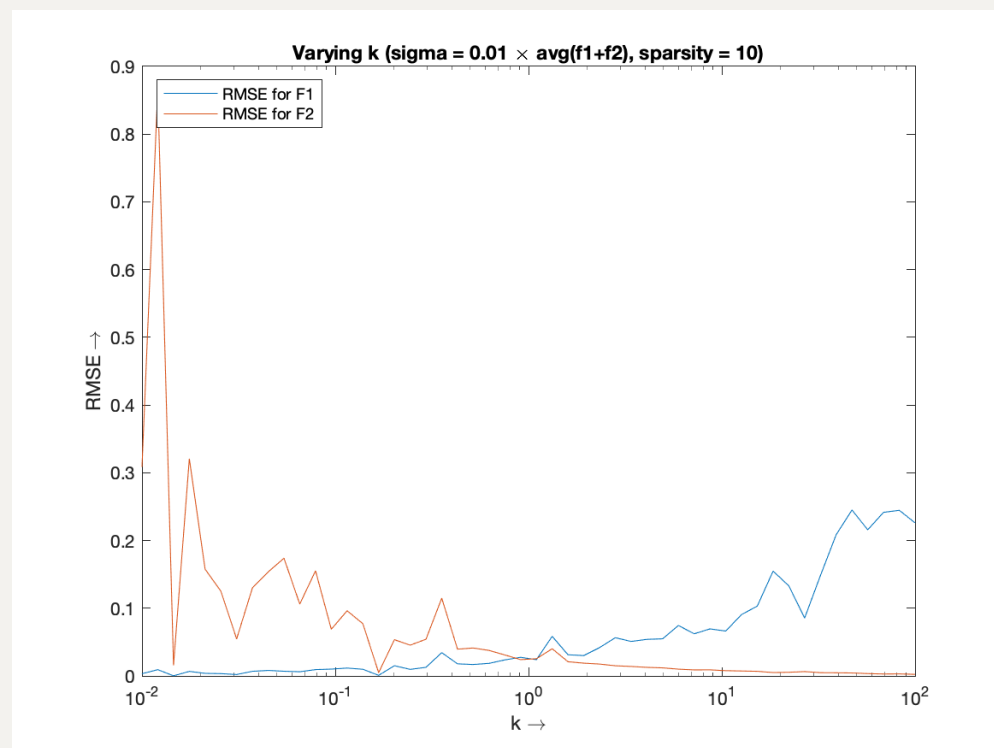
Here, x-axis has the factor  $\tau$ . Where  $\sigma = \tau \times \text{avg}(f_1 + f_2)$



**Varying sparsity and fixed  $\sigma$**



varying  $k$ , fixed  $\sigma$  and sparsity



Q2

Paper: [CoSaMP: Iterative signal recovery from incomplete and inaccurate samples](#)

Author: [D. Needell](#), [J. A. Tropp](#)

Journal reference: Appl. Comput. Harmon. Anal., Vol. 26, pp. 301-321, 2008

## Algorithm

**CoSaMP**( $\Phi, u, s$ )

**Input:** Sampling matrix  $\phi$ , noisy sample vector  $u$ , sparsity level  $s$ .

**Output:** An  $s$ -sparse approximation  $a$  of the target signal

```

1.  $a^0 \leftarrow 0$  // Initial approximation

2.  $v \leftarrow u$  // Current samples = input samples

3.  $k \leftarrow 0$ 

4. repeat
    a.  $k \leftarrow k + 1$ 
    b.  $y \leftarrow \Phi^* v$  // Form signal proxy
    c.  $\Omega \leftarrow \text{supp}(y_{2s})$  // Identify large components
    d.  $T \leftarrow \Omega \cup \text{supp}(a^{k-1})$  // Merge supports
    e.  $b|_T \leftarrow \Phi_T^\dagger u$  // Signal estimation by least-squares
    f.  $b|_{T^c} \leftarrow 0$ 
    g.  $a \leftarrow b_s$  // Prune to obtain next approx
    h.  $v \leftarrow u - \Phi a^k$  // Update current samples

until halting criterion true

```

---

Def. *quasi-norm*:

$$\|x\|_0 = |\text{supp}(x)| = |\{j : x_j \neq 0\}|$$

Def.

$$x|_T = x_i \text{ if } i \in T \text{ else } 0$$

$x|_T$  Is treated as an element of the vector space  $\mathbb{C}^T$ . And the restriction  $\Phi_T$  of the sampling matrix  $\Phi$  is defined as the column sub matrix whose columns are listed in the set  $T$ .

Def. Pseudo inverse:

$$A^\dagger = (A^* A)^{-1} A^*$$

**CoSaMP** uses an approach inspired by the restricted isometry property. The sampling matrix  $\Phi$  has restricted isometry constant  $\delta_s \ll 1$ . For an  $s$ -sparse signal  $x$ , the vector  $y = \Phi^* \Phi x$  serves as a proxy for the signal because the energy in each set of  $s$  components of  $y$  approximates the energy in the corresponding  $s$  components of  $x$ . In particular, the largest  $s$  entries of the proxy  $y$  point toward the largest  $s$  entries of the signal  $x$ . Since the samples have the form  $u = \Phi x$ , we can obtain the proxy just by applying the matrix  $\Phi^*$  to the samples.

---

**Theorem:** (CoSaMP). Suppose that  $\Phi$  is an  $m \times N$  sampling matrix with RIC  $\delta_{2s} \leq c$ . Let  $u = \Phi x + e$  be a vector of samples of an arbitrary signal, contaminated with arbitrary noise. For a given precision parameter  $\eta$ , the algorithm CoSaMP produces a  $2s$ -sparse approximation  $a$  that satisfies

$$\|x - a\|_2 \leq C \cdot \max \left\{ \eta, \frac{1}{\sqrt{s}} \|x - x_s\|_1 + \|e\|_2 \right\}$$

Where  $\mathbf{x}_s$  is a best  $s$ -sparse approximation to  $\mathbf{x}$ . The running time is  $\mathcal{O}(\mathcal{L} \cdot \log(\|\mathbf{x}\|_2/\eta))$ , where  $\mathcal{L}$  bound the cost of a matrix-vector multiply with  $\Phi$  or  $\Phi^*$ . The working storage use is  $\mathcal{O}(N)$

---

### Q3

Note:  $D_i$  is the  $i^{th}$  column of dictionary  $D$

- Derivative filter is applied via convolution of an image  $I$  with a derivation kernel.

$\hat{I} = K * I$ , we use vectorized  $I$  for our dictionary representation .

This convolution can be represented by a matrix  $A$ , s.t.,  $vec(\hat{I}) = A \cdot vec(I)$

$$vec(I) = DS \Rightarrow A \cdot vec(I) = ADS$$

$$AD = [AD_1 | AD_2 | \dots | AD_n]$$

$$\therefore \hat{D} = [AD_1 | AD_2 | \dots | AD_n]$$

So our new dictionary  $\hat{D}$  is obtained by applying the same derivative filter on the columns of the old dictionary  $D$ .

- Rotating an image means re-positioning the cells of that image.

So a rotated version of a vectorized image  $I$  can be computed by multiplying it by a certain matrix,  $rotated(I) = \hat{I} = RI$ , where each row and column in  $R$  have at most 1 non-zero element whose value is 1.

Rotation by  $\alpha$  can be denoted as  $I_\alpha = R_\alpha I$ , similarly rotation by  $\beta$  can be denoted as  $I_\beta = R_\beta I$

$$I = DS$$

$$I_\alpha = R_\alpha I = R_\alpha DS$$

$$R_\alpha D = [R_\alpha D_1 | R_\alpha D_2 | \dots | R_\alpha D_n]$$

$$\therefore D^\alpha = [R_\alpha D_1 | R_\alpha D_2 | \dots | R_\alpha D_n]$$

$$\text{Similarly, } D^\beta = [R_\beta D_1 | R_\beta D_2 | \dots | R_\beta D_n]$$

As some images are rotated by  $\alpha$  while some by  $\beta$ , we need both  $D^\alpha$  and  $D^\beta$ .

$$\therefore \hat{D} = [D^\alpha | D^\beta]$$

So our new dictionary  $\hat{D}$  is obtained by column concatenation of two dictionaries which are obtained by rotating each column of the old dictionary  $D$  (considering it as an image) by  $\alpha$  and by  $\beta$  respectively.

- $I_{new}^i(x, y) = \alpha(I_{old}^i(x, y))^2 + \beta(I_{old}^i(x, y)) + \gamma$

Let the vectorized form of our image be  $X$ .

$$X = DS$$

$$X_i = D^i \cdot S, \text{ where } D^i \text{ is the } i^{th} \text{ row of } D.$$

$$X_i^2 = (\sum_j D_{ji} S_j)^2$$

$$X_i^2 = \sum_j D_{ji}^2 S_j^2 + 2 \sum_{x,y, x>y} D_{xi} D_{yi} S_x S_y$$

$\therefore$  Dictionary  $D^{sq}$  for squared signal:

$$D^{sq} = [D^s | D^q], \text{ where}$$

$D^s = [D_1^2 | D_2^2 | \dots | D_n^2]$ ,  $D_i^2$  is the column vector obtained by squaring elements of  $D_i$

$$D^q = 2[D_1 D_2 | D_1 D_3 | \dots | D_1 D_n | D_2 D_3 | D_2 D_4 | \dots | D_2 D_n | \dots | D_{n-1} D_n],$$

$D_i D_j$  is the column vector obtained by element wise product of  $D_i$  and  $D_j$

$$aX + b = aDS + b$$

$aDS + b = [aD|b\mathbf{1}]S'$ ,  $\mathbf{1}$  is the column vector with every element being 1 and  $S'$  is  $S$  row concatenated with 1.

$\therefore$  Our new Dictionary  $\hat{D}$  is:

$$\hat{D} = [\alpha D^{sq} | \beta D | \gamma \mathbf{1}]$$

- A blur kernel is applied to the image, so as seen in the first part to obtain the new dictionary  $\hat{D}$  we have to apply the same blur kernel to the columns of the old dictionary  $D$ .

- $I$  is our image on which convolution is to be applied

$$I' = K * I, K = \sum_i \alpha_i K_i, K_i \text{ is the } i^{th} \text{ kernel in the set}$$

$$K * I = \sum_i \alpha_i K_i * I$$

$X$  is the vectorized form of  $I$

$$vec(K_i * I) = A^i \cdot X, \text{ for some matrix } A^i$$

$$X = DS$$

$$A^i \cdot X = A^i \cdot DS$$

$$A^i D = [A^i D_1 | A^i D_2 | \dots | A^i D_n]$$

$$D^i = A^i D$$

$$\sum_i \alpha_i K_i * I = \sum_i \alpha_i A^i X = \sum_i \alpha_i A^i DS = \sum_i \alpha_i D^i S$$

$$\sum_i \alpha_i D^i S = \sum_i D^i (\alpha_i S)$$

$$\therefore vec(I') = \sum_i D^i (\alpha_i S)$$

$$\therefore \hat{D} = [D^1 | D^2 | \dots | D^C], \text{ where } C \text{ is the cardinality of the set of kernels}$$

So our new Dictionary  $\hat{D}$  is concatenation of dictionaries each obtained as described in the previous part for each kernel in the set.

#### Q4

- The solution to the following optimization problem:

$$\min_{A_r} ||A - A_r||_F^2 \text{ where } \text{rank}(A_r) = r, r \leq \min(m, n), A \in \mathbb{R}^{m \times n}$$

is given using SVD of  $A$  as follows:

$$A_r = \sum_{i=1}^r S_{ii} u_i v_i^t, \text{ where } A = USV^T$$

- This optimization problem occurs in Image Compression.
- Instead of storing  $mn$  intensity values, we store  $(n + m + 1)r$  intensity values where  $r$  is the number of stored singular values (or singular vectors). The remaining  $m-r$  singular values (and hence their singular vectors) are effectively set to 0.
- This is called as storing a low-rank (rank  $r$ ) approximation for an image.

- SVD gives the best possible rank- $r$  approximation of any matrix.

•

$$\begin{aligned}
 R^* &= \min_R \|A - RB\| \text{ s.t. } R^T R = I // R \text{ is orthonormal} \\
 \min_R \|A - RB\|_F^2 &= \min_A \text{trace}((A - RB)^T (A - RB)) \\
 &= \min_R \text{trace}(A^T A - 2A^T R B + B^T B) \\
 &= \max_R \text{trace}(A^T R B) \\
 &= \max_R \text{trace}(R B A^T) // \text{trace}(FG) = \text{trace}(GF) \\
 \text{--Let } B A^T &= Q, \text{ SVD of } Q \text{ gives } Q = U D V^T \\
 &= \max_R \text{trace}(R U D V^T) \\
 &= \max_R \text{trace}(V^T R U D) \\
 &= \max_R \text{trace}(Z(R) D) \text{ where } Z(R) = V^T R U \\
 &= \max_R \sum_i z_{ii} d_{ii} \leq \sum_i d_{ii} // Z(R)^T Z(R) = I \\
 &\text{The maximum is achieved for } Z(R) = I, \text{ i.e.,} \\
 &V^T R U = I \Rightarrow R = V U^T
 \end{aligned}$$

- This optimization problem arises in Bases learning using Union of Ortho-normal Bases.
- We represent a signal in the following way:

$$\begin{aligned}
 X &= A S + \epsilon \\
 (A, S) &= \min_{A, S} \|X - A S\|^2 + \lambda \|S\|_1
 \end{aligned}$$

- $A$  is an over-complete dictionary, which is assumed to be a union of ortho-normal bases, in the form

$$\begin{aligned}
 A &= [A_1 | A_2 | \dots | A_M] \\
 \forall i, 1 \leq i \leq M, A_i A_i^T &= I
 \end{aligned}$$

- This application of SVD is called orthogonal Procrustes problem.

## Q5

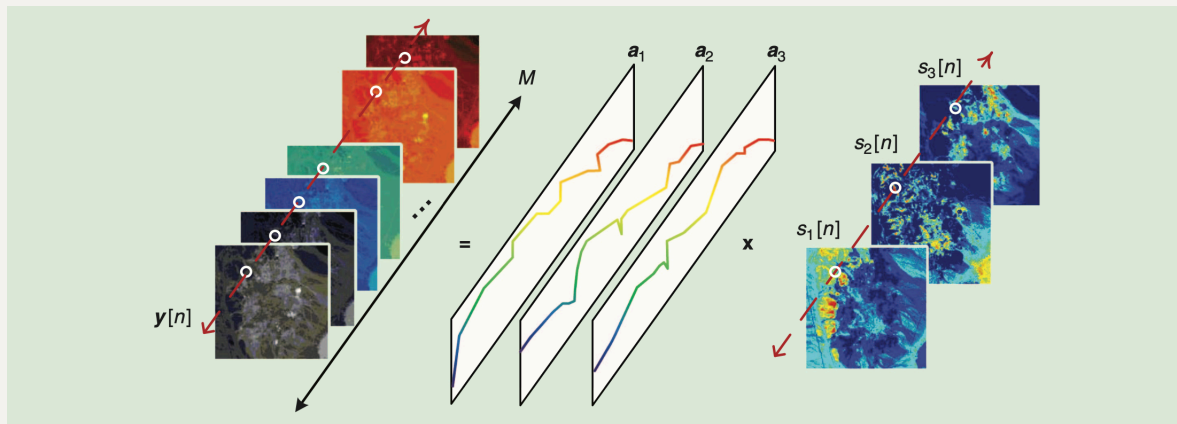
Hyperspectral Unmixing is a procedure that decomposes the measured pixel spectrum of hyperspectral data into a collection of constituent spectral signals and a set of corresponding fractional abundances.

In general, the linear mixture model (LMM) is recognised as an acceptable model for Hyperspectral Unmixing. The LMM is describes as follows. Let  $y_m[n]$  denote the hyperspectral camera's measurement at spectral band  $m$  and pixel  $n$ . Let,  $y[n] = [y_1[n], y_2[n], \dots, y_M[n]]^T \in \mathbb{R}^M$ , where  $M$  is the number of spectral bands. The LMM is given by:

$$y[n] = \sum_{i=1}^N \mathbf{a}_i s_i + \mathbf{v}[n] = \mathbf{A} \mathbf{s}[n] + \mathbf{v}[n]$$

for  $n = 1, \dots, L$ , where,

- each  $\mathbf{a}_i \in \mathbb{R}^M, i = 1, \dots, N$  is called an *endmember signature vector*, which contains the spectral components of a specific material (indexed by i) in the scene.
- $N$  Is the number of endmembers, or material, in the scene.  
 $\mathbf{A} = [\mathbf{a}_1, \dots, \mathbf{a}_N] \in \mathbb{R}^{M \times N}$  Is called the *endmember matrix*.
- $s_i[n]$  describes the contribution of material  $i$  at pixel  $n$ .  
 $\mathbf{s}[n] = [s_1[n], \dots, s_N[n]] \in \mathbb{R}^N$  Is called the *abundance vector* at pixel  $n$ .
- $L$  Is the number of pixels
- $\mathbf{v}[n] \in \mathbb{R}^M$  is noise.



Non-negative matrix factorisation is posed as a low-rank matrix approximation problem where, given a data matrix  $\mathbf{Y} \in \mathbb{R}^{M \times L}$ , the task is to find a pair of non-negative matrices  $\mathbf{A} \in \mathbb{R}^{M \times N}, \mathbf{S} \in \mathbb{R}^{N \times L}$ , with  $N < \min\{M, L\}$ , that solves

$$\min_{\mathbf{A} \geq 0, \mathbf{S} \geq 0} \|\mathbf{Y} - \mathbf{AS}\|_F^2$$

In blind HU the connection is that the NMF factors obtained,  $\mathbf{A}$  and  $\mathbf{S}$ , can serve as estimates of the endmembers and abundances.

$$\begin{aligned} \mathbf{A} &= [\mathbf{a}_1, \dots, \mathbf{a}_N] \in \mathbb{R}^{M \times N} \\ \mathbf{s}[n] &= [s_1[n], \dots, s_N[n]] \in \mathbb{R}^N \end{aligned}$$

In the HU equation mentioned above,  $y[n], s[n]$  are for a pixel at position  $n$ . In the NMF equation  $\mathbf{S}$  and  $\mathbf{Y}$  will be concatenated  $s[n]$  and  $y[n]$ . And so we can reframe hyperspectral unmixing problem into non-negative matrix factorisation problem.

$$\min_{\mathbf{A} \geq 0, \mathbf{S} \in S^L} \|\mathbf{Y} - \mathbf{AS}\|_F^2 + \lambda \cdot g(\mathbf{A}) + \mu \cdot h(\mathbf{S})$$

Where,  $\mathbf{S}^l = \{\mathbf{S} | s[n] \geq \mathbf{0}, \mathbf{1}^T \mathbf{s}[n] = 1, 1 \leq n \leq L\}$ ,  $g$  and  $h$  are regularisers, which vary from one work to another and  $\lambda, \mu > 0$  are some constants.

### MVC-NMF

$$\min_{\mathbf{A} \geq 0, \mathbf{S} \in S^L} \|\mathbf{Y} - \mathbf{AS}\|_F^2 + \lambda \cdot (\text{vol}(B))^2$$

Where,  $\text{vol}(B)$  is the simplex volume corresponding to  $\mathbf{A}$ , in which  $b_i = C^\dagger(a_i - d)$  for all  $i$ . This is essentially a variation of the VolMin formulation.

### ICE

Iterated constrained endmember (ICE) and sparsity promoting ICE (SPICE) avoid this issue by replacing  $(\text{vol}(B))^2$  with a convex surrogate, specifically,  
 $g(\mathbf{A}) = \sum_{i=1}^{N-1} \sum_{j=i+1}^N \|a_i - a_j\|_2^2$ , which is the sum of differences between vertices.

$$\min_{A \geq 0, S \in S^L} \|\mathbf{Y} - \mathbf{AS}\|_F^2 + \sum_{I=1}^{N-1} \sum_{j=i+1}^N \|a_i - a_j\|_2^2$$

## Dictionary Learning

For the abundance regulariser  $h$ , the design principle usually follows that of sparsity.

$$\min_{A \geq 0, S \in S^L} \|\mathbf{Y} - \mathbf{AS}\|_F^2 + \mu \cdot \|\mathbf{S}\|_{1,1}$$

Where,  $\|\mathbf{S}\|_{1,1} = \sum_{n=1}^L \sum_{I=1}^N |s_i[n]|$ . The idea is to learn the dictionary  $\mathbf{A}$  by joint dictionary and sparse signal optimization.

## $L_{1/2}$ -NMF

Similar to the DL optimisation techniques,  $L_{1/2}$ -NMF uses a non-convex, but stronger sparsity-promoting regulariser based on the  $l_{1/2}$  quasi norm. Apart from sparsity, exploitation of spatial contextual information via TV regularisation may be used.

$$\min_{A \geq 0, S \in S^L} \|\mathbf{Y} - \mathbf{AS}\|_F^2 + \mu \cdot \|\mathbf{S}\|_{1/2,1/2}^{1/2}$$