Chapter 3 Inference in the Sparse Land

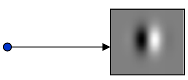
In this chapter, we introduce three algorithms in sparse coding inference.

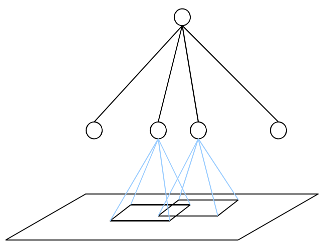
- Matching pursuit for image coding;
- Basis pursuit and Lasso
  --- Lasso is for “Least absolute shrinkage and selection operator”
- Least angle regression

1: Matching pursuit for image reconstruction

At Stat232A, we have mentioned that an image base is in fact similar to an attribute production rule. Therefore the sparse coding model can be viewed as a context free grammar. The only difference is that it adopts a linear additive model instead of an occlusion model in text.

An image base with attributes (x,y), θ, σ, α
is similar to a production rule A→ aBcDe



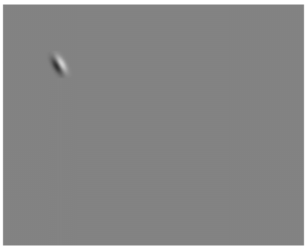


Example of Sparse Coding

$$I = \sum_{i=1}^{n} \alpha_i b_i + \epsilon, \quad b_i \in \Delta$$

Δ: a dictionary of over-complete basis

input image



matching pursuit

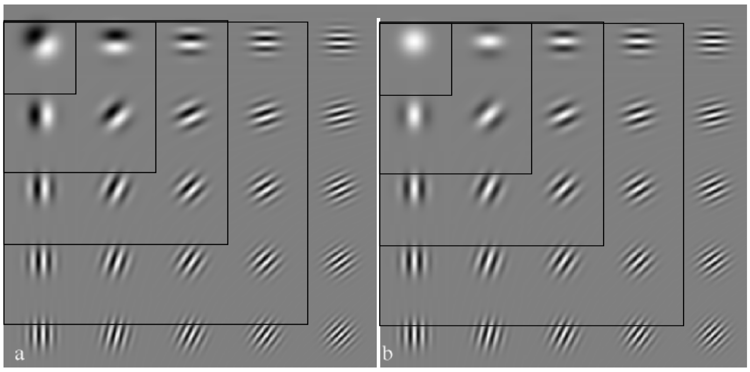
(Mallat & Zhang 93)

Generative models: Decompose the original signal into a set of components.

Example of over-complete basis: Gabor Wavelets

Sine-components at different scale and orientations

Cosine components



From T. S. Lee PAMI Oct. 1996

Matching pursuit algorithm

The algorithm minimizes the squared-error in a step-wised way.

Fixing the first m basis functions $\{(\alpha_i, b_i),\ i = 1,2,\dots,m\}$

The next one is found by minimizing the residual error,

$$(\alpha_m, b_m)^* = \text{argmin} \left(I - \sum_{i=1}^{m-1} \alpha_i b_i - \alpha_m b_m \right)^2$$

Take the derivative w.r.t α_m, and set it to 0, we have $< I - \sum_{i=1}^{m-1} \alpha_i b_i - \alpha_m b_m,\ b_m > = 0$

So,

$$\alpha_m = < I - \sum_{i=1}^{m-1} \alpha_i b_i, b_m >$$ is the projection of the residual image on b_m

Plug in, we have the squared-error,

$$\begin{aligned} Err_m &= \left(I - \sum_{i=1}^{m-1} \alpha_i b_i - \alpha_m b_m \right)^2 \\ &= \left(I - \sum_{i=1}^{m-1} \alpha_i b_i \right)^2 - 2\alpha_m < I - \sum_{i=1}^{m-1} \alpha_i b_i, b_m > + \alpha_m^2 < b_m, b_m > \\ &= Err_{m-1} - \alpha_m^2 \end{aligned}$$

Therefore, we choose b_m that has the largest α_m^2.

Matching pursuit algorithm

The data structure of the algorithm is like an OpenList in the heuristic search.

The bases from an over-complete dictionary are sorted in decreasing order by the projection lengths of the image I on them.

b_1	b_2	b_3							b_m	

$\alpha_i = <I, b_i>$

Each time, it chooses the base b_1 with highest project length, remove it from the OpenList, Adjust the projection length of any other bases that overlaps with b_1. These updated bases will be re-sorted in the OpenList.

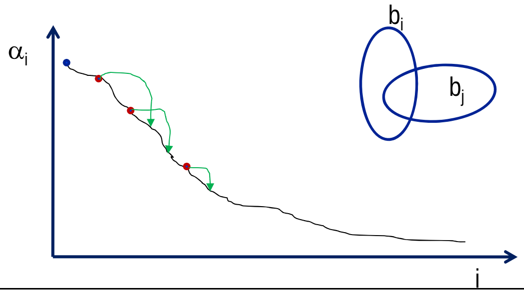
$I^{new} = I - \alpha_1 b_1$ The residue image is orthogonal to the projection

$\alpha_i^{new} = <I^{new}, b_i> = \alpha_i - \alpha_1 c_{1i}$ $C_{1i} = <b_1, b_i>$ is computed once off-line.

Matching pursuit algorithm

$\alpha_i = <I, b_i>$

The base functions { b } are sorted according the projection of the residue images, thus each time a base function b with the largest projection α is selected so that the reconstruction error (residue) is maximally reduced.



Lasso, LARS and group Lasso
are in the three papers and discussed on boad.