A tutorial on sparse modeling.

Outline:

1. Why?

2. What?

3. How.

4. no really, why?

Sparse modeling is a component in many state of the art signal processing and machine learning tasks.

- image processing (denoising, inpainting, superresolution): [Yu, Mallat, Sapiro], [Mairal, Elad, Sapiro].

- Object recognition: [Yang, Yu, Gong, Huang], [Boureau, La Roux, Bach, Ponce, LeCun].

- general supervised learning: [Mairal, Bach, Ponce, Sapiro, Zisserman].

- Building graphs for large scale semi-supervised learning: [Liu, Wang, Kumar, Chang].

Sparse modeling and matrix factorizations

Given a $d \times n$ matrix X of n points in $\mathbb{R}^d$.

- Want to factor $X \approx WZ$, where W is $d \times K$, and Z is $K \times N$.

- W is a dictionary, Z are the coefficients.

- We need to choose an appropriate notion of "close" and conditions on Z to force the decomposition to be parsimonious

- If we restrict the size of $K < \min(d, N)$, and "close" is operator or Frobenious norm, we get PCA.

- If we restrict $Z_{ij} \in \{0, 1\}$, and $\sum_i Z_{ij} = 1$ (i.e. Z_j is really sparse!), we get K-means

- Everything in between (including the endpoints): dictionary learning. E.g.

$$\arg\min_{Z \in \mathbb{R}^{K \times n}, W \in \mathbb{S}^{(d-1) \times K}} \sum_{j=1}^{n} ||Wz_j - x_j||^2, \ ||z_j||_0 \leq q,$$

- or the Z coordinate convexification:

$$\arg\min_{Z \in \mathbb{R}^{K \times n}, W \in \mathbb{S}^{(d-1) \times K}} \sum_{j=1} ||Wz_j - x_j||^2 + \lambda ||z_j||_1.$$

Structured sparsity/Group sparsity

Coefficents lie in specified groups; constraints on or penalties for non-sparse group activations rather than non-sparse elementwise activations

- A simple "manifold" model: non-overlapping groups and 1-sparse group activations.

- If the groups overlap, can encourage trees, grids, etc.

Manifold learning

- manifold=locally well approximated by affine spaces (for a true manifold, the tangent spaces).

- It may be impractical to work with the tangent planes at every point in X.

- if the "curvature" of X is not excessive, it may be possible to find a set of l good q dimensional "secant" planes so that every point is close to its secant plane.

Choosing the $q-$planes that minimize the average distance from each point in X to its plane is minimizing

$$||WZ - X||_F^2,$$

such that

$$W = d\Big\{ \begin{pmatrix} \overbrace{W_1}^{q} & \overbrace{W_2}^{q} & \dots & \overbrace{W_l}^{q} \end{pmatrix}, \quad Z = \begin{array}{c c} & \begin{array}{ccccc} \overbrace{}^{|X_1|} & \overbrace{}^{|X_2|} & & & \overbrace{}^{|X_l|} \end{array} \\ \begin{array}{c} q\{ \\ q\{ \\ q\{ \\ \\ q\{ \end{array} & \begin{pmatrix} Z_1 & 0 & 0 & \cdots & 0 \\ 0 & Z_2 & 0 & \cdots & 0 \\ 0 & 0 & Z_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & 0 & Z_l \end{pmatrix} \end{array},$$

where X_i is the set of points whose nearest plane is the span of W_i

Thus we can interpret approximating the data set by l q-planes as a "structured" sparse dictionary design problem with $K = lq$.

In fact, all the previous models are “manifold” models. For each of the previous models:

- The analysis map from x to z with W fixed is a piecewise affine.

- The reconstruction map $y = Wz$ is linear.

For example: for the map

$$z_* = z_*(x, W) = \arg\min_z ||Wz - x||^2 + \lambda||z||_1,$$

- under mild regularity conditions on W, the solution z_* is unique, and has explicit solution once its sign is fixed:

$$z_*|\Omega = (W_\Omega^T W_\Omega)^{-1}(W_\Omega^T x - \lambda\epsilon),$$

where $\epsilon = \text{sign}(z)$, and Ω is the set of nonzero entries in ϵ.

Sparse coding vs. compressive sensing

Compressive sensing: $$\arg\min_{z \in \mathbb{R}^K} ||Wz - x||^2 + \lambda ||z||_1.$$

Here, z is the data, and x is the code. Encoding is trivial (multiplication by W), decoding requires an optimization. W is *universal*.

Sparse coding: $$\arg\min_{z \in \mathbb{R}^K} ||Wz - x||^2 + \lambda ||z||_1.$$

Here, z is the code, and x is the data. Decoding is trivial (multiplication by W), encoding requires an optimization. W is *adapted*.

Greedy methods for the forward l_0 problem with W fixed

$$\min_z ||Wz - x||^2,$$

$$||z||_0 \le q,$$

where the $d \times K$ matrix W is the dictionary, the $K \times 1$ z is the code, and x is an $d \times 1$ data vector.

- matching pursuit, orthogonal matching pursuit, order recursive matching pursuit

- CoSaMP [Needell and Tropp].

(O)MP:

1. **Initialize:** coefficients $z = 0$, residual $r = x$, active set $\Omega = \emptyset$.

2. $j = \arg\max_i |W_i^T r|$

3. $\Omega = \Omega \bigcup j$

4. For MP $z_j = W_\Omega^T r$
   For OMP $z = \left(W_\Omega^T W_\Omega\right)^{-1} W_\Omega^T X$

5. $r = x - Wz$. Goto 2 until q iterations.

Note that with a bit of bookkeeping, it is only necessary to multiply W against x once, instead of q times. This at a cost of an extra $O(k^2)$ storage for the Gram matrix Q of W. We can also keep a running update of $Q_\Omega^{-1} = \left(W_\Omega^T W_\Omega\right)^{-1}$ using a Cholesky factorization, and the submatrix $\overline{Q}_\Omega = W^T W_\Omega$ of Q. Critical for many inferences with a fixed dictionary.

1. **Initialize:** $t = s = W^T x$, active set $\Omega = \emptyset$.

2. $j = \arg\max_i |t_i|$

3. $\Omega = \Omega \bigcup j$, update Q_Ω^{-1}

4. $t = s_\Omega - \overline{Q}_\Omega Q_\Omega^{-1} s_\Omega$

5. goto 2 until q iterations.

when to use what method?

- ORMP>OMP>>MP, in terms of accuracy. Exactly opposite in terms of runtime.
- don't use MP unless you have to (need every cpu cycle, or in convolutional problems).
- if problem is large, and only being done once, solution is not very sparse, and dictionary is well conditioned, use CoSaMP.

In general, greedy methods good when you expect/will enforce extreme sparsity. Computation time is roughly on the order of one multiplication of the data by the dictionary, assuming you have stored the Q.

methods for the forward relaxed problem with W fixed:

$$\arg\min_z ||Wz - x||^2 + \lambda||z||_1$$

too many methods to discuss. Will focus on two good ones.

LARS [Efron, Hastie, Johnstone, Tibshirani] uses the explicit solution once the active set is fixed to generate a path in solution space parameterized by the regularity. As before, can store $W^T W$ and keep running updates of all variables in compact form for large speedup.

1. set $\Omega = \arg\max |W_j^T x|$, $\lambda = |W_\Omega^T x|$,

2. choose the next smallest λ such that with
   $z|\Omega = (W_\Omega^T W_\Omega)^{-1}(W_\Omega^T x - \lambda \epsilon_\Omega)$, $z_{\Omega^c} = 0$,

   (a) $\exists\ i \in \Omega^c$ such that $|W_i^T (Wz - x)| = \lambda$; in this case, $\Omega = \Omega \bigcup i$.

   (b) $\exists\ i \in \Omega$ such that $z_i = 0$; in this case, $\Omega = \Omega - i$.

3. Update ϵ

ISTA: Iterated Shrinkage Thresholding Algorithm or proximal gradient descent:

1. **Initialize:** $z = 0$.

2. $y = z - \eta W^T(Wz - x)$
   (gradient step with respect to the smooth part).

3. $z = \arg\min_p ||p - y||^2 + \eta\lambda|p|_1$
   $= \mathrm{shrink}(y, \eta\lambda)$
   $= (|y| - \eta\lambda)_+ \,\mathrm{sign}(y)$
   (optimize the nonsmooth part with a penalty for straying too far from smooth update).

4. goto 2 until stopping criteria.

As before, we can precompute things and make the algorithm a little faster. Set $Q = W^T W$, $b = W^T x$.

1. **Initialize:** $z = 0$, $t_1 = 1$.

2. $x_k = \text{shrink}((I - Q)z - b, \eta\lambda)$

3. $t_k + 1$

4. repeat until stopping criteria.

Notice: linear map, followed by offset, followed by nonlinearity. Repeat.

Using a clever (magic) momentum term convergence can be greatly sped up! [Nesterov 1983, Beck and Teboulle 2009]

1. $y_k = \text{shrink}((I - Q)z_k - b, \eta\lambda)$

2. $t_{k+1} = \left(1 + \sqrt{1 + t_{k+1}^2}\right)/2$

3. $z_{k+1} = y_k + \frac{t_k - 1}{t_{k+1}}(y_k - y_{k-1})$

when to use what method?

- for many small, very sparse problems use LARS (almost as fast as OMP there).

- if problem is large, and only being done once, solution is not very sparse, and dictionary is well conditioned, use Nesterov accelerated proximal gradient descent.

Note: an introduction to methods for basis pursuit could easily be a weeklong affair.

Learning the W

General good practice: some version of stochastic gradient descent.

- Gradient w.r.t. W:

$$\nabla W = (Wz - x)x^T.$$

Can sometimes do better with averaging type sgd. e.g. [Mairal, Bach, Ponce, Sapiro].

Batch: alternate between updating the codes and updating the filters, as in K-SVD [Aharon el. al]:

1. Initialize W.

2. Solve for Z as above.

3. For each W_j,

   - find all x where W_j is activated

   - for each such x_p, find e_p by removing the contribution of W_j (that is $e_p = x_p - W_j z_{jp}$).

   - update $W_j \leftarrow \mathrm{PCA}(E_p)$

What do we know theoretically?:

About the compressed sensing problem, Lots!

- if W is sufficiently regular (e.g. incoherent), and z is sufficiently sparse, both greedy methods and l_1 relaxations are guaranteed to recover the true z

- Mutual coherence: $\mu(W) = \text{max}_{i \neq j}(| < W_j, W_i > |)$

- Problem: dictionaries we train will often be coherent.

What do we know theoretically about dictionary learning (that is, when does it work?):

Very little!

Dictionary identification:

- If enough data is sampled i.i.d. from distribution built from an incoherent dictionary, then w.h.p. the "true solution" is a local minimum for the dictionary learning problem [Gribonval and Schnass], [Geng and Wright].

- These works are for the constrained problem

$$\min |z|_1 \text{ s.t. } Wz = x.$$

Generalization bounds [Maurer and Pontil]:

Define

$$z(x, W) = \operatorname*{arg\,min}_{|z|_1 = 1} ||Wz - x||^2$$

suppose the n point set $X \subset \mathbb{R}^d$ is generated i.i.d. from μ, and W_* is the minimizer of

$$\sum_{x \in X} ||W_* z(x, W_*) - x||^2,$$

and $\hat{W}$ is the minimizer of

$$\mathbb{E}_{\mu(x)} ||\hat{W} z(x, \hat{W}) - x||^2,$$

and B is the value of that expression at the minimizer. Then

$$\mathbb{E}_{\mu(x)} ||W_* z(x, W_*) - x||^2 < B + O\left(K\sqrt{\frac{\ln m}{m}} + \sqrt{\frac{\log(1/\delta}{m}}\right)$$

with probability δ. (see also [Vainsencher, Mannor, Bruckstein])

- But all of these discuss our ability to succesfully use the model. They do not give much insight to *when* the model makes sense and should be used.

- Can we look at a set of data points, extract some geometric statistics, and then decide sparse modeling is a reasonable approach for that data? and estimate the correct method and parameters for maximum generalization?

- even in simple cases?

let $R(X, K, q)$ be the minimal error $||W_* Z_* - X||^2_{\text{FRO}}$ for a given K, q, and X in the pure sparse coding model.

- Question 1: What is the worst possible reconstruction error for a data set with n points? In equations, the problem is to describe

  $$f(K, q, n) = \max_{X \in \mathbb{S}^{d-1}, |X|=n} R(X, K, q).$$

  Here X is constrained to the unit sphere to avoid a trivial answer via scaling, and $|X|$ is the number of elements in X.

- Question 2: Suppose that we know X is actually close to a given set of q'-planes in $\mathbb{R}^d$, that is, there exist orthogonal matrices $P_1, ..., P_m$ of size $d \times q'$

$$\sum_j \min_i ||x_j - P_i P_i^T x_j||^2 \leq \epsilon.$$

Then describe

$$f(K, q, n) = \min_{X \in \mathbb{S}^{d-1}, |X|=n} R(X, K, q).$$

Also: how to get from a representation of X via the P to a representation via W and Z?

- Question 3: More generally, what kinds of geometries (if not locally approximated by planes) allow for good representations via the various sparse coding models? In other words, given a data set, how can we decide which (if any) of the models are appropriate?
  - Not completely trivial/nontrivial even for PCA, depending on the (kind of) noise in the data
  - Or even: how can we decide on the parameters of the model if we know the correct one?!
  - Just deciding q is a serious issue (even in the PCA case, with certain kinds of noise)....

- What about the relationship between sparse modeling and pooling?