5. MSRI Graduate Summer School Notes
Jeff Blanchard, 2018
The Restricted Isometry Property

In the last section we saw that the we can recover a sparse vector encoded by A using one of several algorithms provided the encoder had small coherence. This is promising indeed, but we ultimately realize the coherence constraint intorduces the so called square root bottleneck in that we need $m = O(k^2)$. One way this is handled is to introduce a different property of the matrix that might be achieved by matrices with $m = O(k)$ as we wanted all along.

In this chapter, we introduce the restricted isometry property (RIP) which was first put forth by Candes and Tao in early work on compressed sensing. This is simply a linear algebra condition describing how much the measurement matrix A can distort sparse vectors. It is closely related to the singular values of the submatrices of A. We first present the simplest, and original, version of the RIP. Importantly, though omitted here, it has been shown that matrices drawn from several random matrix ensembles will satisfy various RIP conditions with high probability even when $m \sim Ck$.

5.1. Restricted Isometry Constants.

Definition 5.1. Let $A \in \mathbb{R}^{m\times n}$. A has the *restricted isometry constant* (RIC) δ_s if δ_s is the smallest value such that

$$(1-\delta_s)\|z\|_2^2 \leq \|Az\|_2^2 \leq (1+\delta_s)\|z\|_2^2$$

for every vector $z \in \chi_n(s)$. In other words, if $z \in \mathbb{R}^n$ has at most s nonzero values, then the matrix A does not decrease $\|z\|_2^2$ by more than $(1-\delta_s)$ or increase $\|z\|_2^2$ by more than a factor of $(1+\delta_s)$.

This is related to the singular values of the submatrices of A which contain exactly s columns. The lower bound in the RIP statement is equal to the minimum minimum singular value over all submatrices of A with s columns while the upper bound is the maximum maximum singular value of all submatrices with s columns. Think about this for a second. The double words "minimum minimum" and "maximum maximum" are not errors. If we fixed a submatrix with s columns, say $B = A_S$ where $|S| = s$, that would define a single support set, S for the vectors z. Then the lower bound would be the minimum singular value of B while the upper bound would be the maximum singular value of B. Now, for the matrix A to have the restricted isometry constant δ_s, it would have to be that over all possible support sets of size s, the lower bound holds. Therefore, $(1-\delta_s)$ is related to the minimum of all the minimum singular values of matrices with s columns from A. Similarly, the upper bound in the RIP statement is related to the largest of all the maximum singular values of this set of submatrices.

We define the RICs using the variable s because we will be looking for k sparse vectors. We will often need multiples of k on our RICs such as $s = ck$ for some constant c where ck is an integer.

Here's a quick exercise to remind us how to deal with norms and the benefit of squaring them in calculations.

Exercise 5.1. Suppose $\text{supp}(u) \cap \text{supp}(v) = \emptyset$. Show that $\|u \pm v\|_2^2 = \|u\|_2^2 + \|v\|_2^2$.

This is why it is standard to define the RICs with squared norms. When we impose a condition on the RICs, we call this a restricted isometry property or RIP. For example, a sufficient condition for having a unique k-sparse vector x such that $y = Ax$ is that A satisfies the RIP $\delta_{2k} < 1$. This is a nice place to start with restricted isometries.

Exercise 5.2. Let $A \in \mathbb{R}^{m\times n}$ satisfy the RIP $\delta_{2k} < 1$, let $x \in \chi_n(k)$, and suppose $y = Ax$. Prove that x is the unique, sparsest solution to $y = Ax$. (In other words, if these hypotheses are satisfied, then the ℓ_0 decoder will return the vector x.) What does the statement $\delta_{2k} < 1$ say about $\text{spark}(A)$?

Now we continue studying the RIP on its own. This lemma will prove useful later but the exercises proving this lemmas should help you understand how the RIP comes into the analysis.

Lemma 5.1. *Suppose $A \in \mathbb{R}^{m\times n}$ satisfies the RIP condition $\delta_{2k} < 1$ and let $u, v \in \chi_n(k)$ with $\text{supp}(u) = S$, $\text{supp}(v) = T$, $|S| = |T| = k$, and $S \cap T = \emptyset$. Then*

(i) $(1-\delta_{2k})\left(\|u\|_2^2 + \|v\|_2^2\right) \leq \|A(u \pm v)\|_2^2 \leq (1+\delta_{2k})\left(\|u\|_2^2 + \|v\|_2^2\right)$;
(ii) $|(Au)^*(Av)| = |\langle Av, Au\rangle| \leq \delta_{2k}\|u\|_2\|v\|_2$.

The proof of this lemma is the result of the following exercises where we always assume the hypotheses of Lem 5.1, namely $\text{supp}(u) = S$, $\text{supp}(v) = T$, $|S| = |T| = k$, and $S \cap T = \emptyset$.

Exercise 5.3. If A satisfies the RIP condition $\delta_{2k} < 1$, prove that

$$(1 - \delta_{2k}) \left(\|u\|_2^2 + \|v\|_2^2 \right) \leq \|A(u \pm v)\|_2^2 \leq (1 + \delta_{2k}) \left(\|u\|_2^2 + \|v\|_2^2 \right).$$

Exercise 5.4. Prove that $\|Au + Av\|_2^2 - \|Au - Av\|_2^2 = 2\langle Au, Av \rangle + 2\langle Av, Au \rangle$.

Exercise 5.5. Using the previous exercise, conclude that

$$|(Au)^*(Av)| = |\langle Av, Au \rangle| = \frac{1}{4} \left| \|A(u+v)\|_2^2 - \|A(u-v)\|_2^2 \right|.$$

Exercise 5.6. Now define

$$\tilde{u} = \frac{u}{\|u\|_2} \quad \text{and} \quad \tilde{v} = \frac{v}{\|v\|_2}$$

so that $\|\tilde{u}\|_2 = \|\tilde{v}\|_2 = 1$. Use Exer. 5.5 and Exer. 5.3 to show that

$$|(A\tilde{u})^*(A\tilde{v})| = |\langle A\tilde{v}, A\tilde{u} \rangle| \leq \delta_{2k}.$$

(Hint: From Exer. 5.1, what is $\|\tilde{u} \pm \tilde{v}\|_2^2$?)

Exercise 5.7. Employ the previous result to show that

$$|(Au)^*(Av)| = |\langle Av, Au \rangle| \leq \delta_{2k} \|u\|_2 \|v\|_2.$$

(Hint: Rewrite the inner product in terms of $\tilde{u}$ and $\tilde{v}$ by scaling.)

Then Exers. 5.3 and 5.7 establish Lem. 5.1.

5.2. **Iterative Greedy Algorithms.** There are many things one can prove about the RICs and we won't actually employ Lemma 5.1 until a proof about the ℓ_1 decoder. First, we introduce a new greedy algorithm, *Iterative Hard Thresholding* (IHT). The basic idea of this algorithm is that we wanted the Thresholding Decoder to decode our vector x from y and A. If it got the support of x correct, it would return the correct values as well. If it did not get the support correct, we should try to reduce the size of the residual by moving in the best direction. If x^j is our current approximation and the residual is r^j, the best direction is the gradient direction and the gradient of our optimization problem is $A^* r^j = A^*(y - Ax^j)$. So, we move in that direction and take a new guess at what the support set should be. We continue in this manner until we do in fact have the correct support set. At that point, the algorithm is simply a subspace restricted steepest descent hoping to converge to the correct values on the support. (Of course, we could also fail.)

Algorithm 3 Iterative Hard Thresholding (IHT)

Input: A, y, k, and an error tolerance *tol*
Output: A k-sparse approximation $\hat{x}$ of the target signal x

Initialization: **Set** $j = 0; x^0 = 0, r^0 = y$**;**

Iteration:

1: **while** $\|r\|_2 > tol$ **do**
2: $w^j = x^{(j-1)} + A^* r^{(j-1)}$; (identify column of A most correlated to residual)
3: $T^j = \text{PrincipalSupport}_k(w^j)$; (estimate for support set)
4: $x^j_{T^j} = w^j_{T^j}$ and $x^j_{(T^j)^c} = 0$; (projection onto k subspace defined by T^j)
5: $r^j = y - Ax^j$; (update the residual)
6: $j = j + 1$; (increase the iteration counter)
7: **end while**
8: **return** $\hat{x} = x^k$ (return the k-sparse vector x^k)

The flexibility of this algorithm to change the support set differs from both Thresholding and OMP and allows it to succeed more frequently. The algorithm also has computational advantages over OMP when the problem sizes get larger and the sparsity is large. Moreover, the following theorem states that if A is an encoder satisfying a particular RIP condition, then A with IHT is a successful endcoder-decoder pair. This

is our first result that achieves our desired outcome where the number of measurements can be proportional to the sparsity ($m \approx Ck$).

Theorem 5.2.1. *Suppose $A \in \mathbb{R}^{m\times n}$ with restricted isometry constant δ_{3k}. Let $x \in \chi_n(k)$, $y = Ax$, and let $\{x^j\}$ be the sequence of vectors generated by the iterations of IHT (Alg. 3) when given (y, A, k). If*

$$\delta_{3k} < \frac{1}{2}$$

then $\{x^j\}$ converges to x so that the IHT decoder can return an arbitrarily accurate approximation to x.

This is an iterative algorithm, much like Newton's method is an iterative method. IHT will create a sequence of vectors $\{x^j : j = 1, \ldots, \infty\}$ and we want this sequence of vectors to converge to the original target vector x, i.e. we want $\lim_{j\to\infty} \|x - x^j\|_2^2 = 0$. The great thing is that we need only show that after an iteration we are a little bit closer to x than we were in the previous iteration.

Exercise 5.8. Suppose $x \in \mathbb{R}^n$ and $\{x^j\}$ is a sequence of vectors in $\mathbb{R}^n$. Suppose that for all $j \in \mathbb{N}$

$$\|x - x^j\|_2 \leq \rho\|x - x^{j-1}\|_2.$$

Prove that if $\rho < 1$ then $\{x^j\}$ converges to x.

For our proof of IHT convergence to x, we need a particular implicatoin of the restricted isometry constants.

Lemma 5.2. *Suppose $u, v \in \mathbb{R}^n$ with $S = supp(u)$, $T = supp(v)$, and $|S \cup T| = s$. Suppose $A \in \mathbb{R}^{m\times n}$ has the restricted isometry constant δ_s. Then,*

$$|\langle (I - A^*A)u, v\rangle| \leq \delta_s\|u\|_2\|v\|_2.$$

Proof. Let $Q = S \cup T$ so that $|Q| = s$. Since $S, T \subset Q$ note that $\langle u, v\rangle = \langle u_Q, v_Q\rangle$, $\|u_Q\|_2 = \|u\|_2$, $\|v_Q\|_2 = \|v\|_2$, $Au = A_Q u_Q$, and $Av = A_Q v_Q$. These facts allow the following computations

$$\begin{aligned}
\langle (I - A^*A)u, v\rangle = \langle u, v\rangle - \langle A^*Au, v\rangle &= \langle u, v\rangle - \langle Au, Av\rangle \\
&= \langle u_Q, v_Q\rangle - \langle A_Q u_Q, A_Q v_Q\rangle \\
&= \langle u_Q, v_Q\rangle - \langle A_Q^* A_Q u_Q, v_Q\rangle \\
&= \langle (I - A_Q^* A_Q)u_Q, v_Q\rangle \\
&\leq \|(I - A_Q^* A_Q)u_Q\|_2\|v_Q\|_2 \\
&\leq \delta_s\|u_Q\|_2\|v_Q\|_2 \\
&= \delta_s\|u\|_2\|v\|_2
\end{aligned}$$

where the first inequality is an application of the Cauchy-Shwartz inequality and the second inequality is an application of Lemma 6.2 which is proven in Exercises 6.4-6.7. □

Now we are ready to prove Thm. 5.2.1. We will do so with an observation followed by a sequence of exercises. Throughout the exercises, we are assuming the hypotheses of Thm. 5.2.1 and using the notation from Algorithm 3. First, we observe that for any vector z, the best k term approximation of that vector is the one where we set all entries to zero other than those in $\text{PrincipalSupport}_k(z)$. So, we know that in each iteration, x^j is a better k-sparse approximation of w^j than is the vector x, namely

$$\|w^j - x^j\|_2^2 \leq \|w^j - x\|_2^2.$$

Exercise 5.9. Show that $\|w^j - x^j\|_2^2 = \|w^j - x\|_2^2 + \|x - x^j\|_2^2 + 2\langle w^j - x, x - x^j\rangle$.

Exercise 5.10. Use the previous exercise and the observation $\|w^j - x^j\|_2^2 \leq \|w^j - x\|_2^2$ to argue that $\|x - x^j\|_2^2 \leq 2\langle x - w^j, x - x^j\rangle$. (Hint[4].)

Exercise 5.11. Show that $x - w^j = (I - A^*A)(x - x^{j-1})$.

Exercise 5.12. Combine the three previous exercises with Exercise 5.8 and Lemma 5.2 to complete the proof of Theorem 5.2.1.

[4]Note that the inner product involves $x - w^j$ rather than $w^j - x$ in the previous exercise.

There are many things to say about iterative greedy algorithms in compressed sensing and sparse approximation, but I will restrict my comments to practical guidance on algorithm selection. The algorithms we studied here are incredibly important in terms of our understanding of how they work and our introduction to compressed sensing, and historically to the development of other algorithms. However, compressed sensing has seen major advances in algorithm development, theory, and implementation. The two best greedy algorithms for compressed sensing are *Conjugate Gradient Iterative Hard Thresholding*[5,6] (CGIHT) and *Compressive Sampling Matching Pursuit* [7] (CoSaMP).

- Thresholding is not a good choice, because there are other algorithms, like CoSaMP and CGIHT that will always do at least as well as Thresholding.
- OMP might be a good choice. It is a good choice when your problem is small so that k is small. Remember that you must solve k least squares problems.
- Solving ℓ_1 minimization is not an algorithm, it's an optimization problem. You need an algorithm to solve ℓ_1-minimization. Often these algorithms have many tuning parameters and require extensive tuning.
- If you want to solve a compressed sensing or sparse approximation problem I recommend CGIHT or CoSaMP described below.

CGIHT is an algorithm developed by combining the advantageous features of many other algorithms including Normalized Iterative Hard Thresholding, Hard Thresholding Pursuit, and CoSaMP. It turns that a nice unifying theory acompanies the algorithm design, but the main feature is that it is very fast and able to solve problems with relatively large sparsity levels k for a given encoder $A \in \mathbb{R}^{m\times n}$. CoSaMP has a higher computational cost (and is therefore slower) but can solve problems with somewhat larger sparsity levels that can CGIHT. When a set of problem dimensions (k, m, n) indicate CGIHT will be successful, I recommend using CGIHT. When (k, m, n) are such that CGIHT will likely fail but CoSaMP will succeed, then certainly use CoSaMP. A detailed discussion can be found in the papers on CGIHT.

I encourage to implement both of these algorithms and test them just as you did with Thresholding and OMP.

Exercise 5.13. Implement and test IHT. (If it is not working to well, change step 2 to $w^j = x^{(j-1)} + \frac{1}{2}A^*r^{(j-1)}$.)

Exercise 5.14. Implement and test CGIHT.

Exercise 5.15. Implement and test CoSaMP.

[5] *Conjugate Gradient Iterative Hard Thresholding for Compressed Sensing and Matrix Competion*, J.D. Blanchard, J. Tanner, K. Wei, **Information and Inference: a journal of the IMA**, 4(4): 289-327, 2015

[6] *Conjugate Gradient Iterative Hard Thresholding: Observed Noise Stability in Compressed Sensing*, J.D. Blanchard, J. Tanner, K. Wei, **IEEE Transactions on Signal Processing**, 63(2): 528-537, 2015

[7] *CoSaMP: Iterative Signal Recovery from Incomplete and Inaccurate Samples*, D. Needell, J.A. Tropp, **Applied and Computational Harmonic Analysis**, 26, 301-321, 2009

Algorithm 4 Conjugate Gradient Iterative Hard Thresholding Restarted (CGIHT)

Input: A, y, k, and an error tolerance tol
Output: A k-sparse approximation $\hat{x}$ of the target signal x

Initialization: Do a single step of thresholding and set $T^{-1} = \emptyset$ and $p^0 = 0$;

$w^0 = A^* y$; (compute column correlations with y)
$T^0 = \text{PrincipalSupport}_k(w^0)$; (estimate for support set)
$x^0 = w^0_{T^0}$ and $x^0_{(T^0)^c} = 0$; (threshold to k-sparse vector with support T^0)
$r^0 = y - Ax^0$; (set the initial residual)
Set $j = 1$;

Iteration:

while $\|r\|_2 > tol$ **do**
  $g^j = A^* r^{j-1}$; (construct the gradient direction)
  if $T^{j-1} \neq T^{j-2}$; **then**
    $\beta^j = 0$; (if support changed restart with gradient direction)
  else
    $\beta^j = \frac{\|g^j_{T^{j-1}}\|_2^2}{\|g^{j-1}_{T^{j-1}}\|_2^2}$; (if support did not change set orthogonalization parameter)
  end if
  $p^j = g^j + \beta^j p^{j-1}$; (create conjugate gradient direction)
  $\alpha^j = \frac{\|g^j_{T^{j-1}}\|_2^2}{\|A_{T^{j-1}} p^j_{T^{j-1}}\|_2^2}$; (compute subspace restricted optimal step size)
  $w^j = x^{(j-1)} + \alpha^j p^j$; (update current approximation)
  $T^j = \text{PrincipalSupport}_k(w^j)$; (estimate for support set)
  $x^j_{T^j} = w^j_{T^j}$ and $x^j_{(T^j)^c} = 0$; (threshold to k-sparse vector with support T^j)
  $r^j = y - Ax^j$; (update the residual)
  $j = j + 1$; (increase the iteration counter)
end while
return $\hat{x} = x^k$ (return the k-sparse vector x^k)

Algorithm 5 Compressive Sampling Matching Pursuit (CoSaMP)

Input: A, y, k, and an error tolerance *tol*
Output: A k-sparse approximation $\hat{x}$ of the target signal x

Initialization: Do a single step of thresholding: set $j = 0$;

$w^0 = A^* y$; (compute column correlations with y)
$T^0 = \text{PrincipalSupport}_k(w^0)$; (estimate for support set)
$x^0 = w^0_{T^0}$; (threshold to k-sparse vector with support T^0)
$r^0 = y - Ax^0$; (set the initial residual)

Iteration:

while $\|r\|_2 > tol$ **do**
  $S^j = \text{PrincipalSupport}_k(A^* r^{(j-1)})$; (identify columns of A most correlated to residual)
  $\Lambda^j = T^{j-1} \cup S^j$; (expanded estimate for set containing support)
  $w^j_{\Lambda^j} = A^\dagger_{\Lambda^j} y$ and $w^j_{(\Lambda^j)^c} = 0$; (projection onto $2k$ subspace defined by Λ^j)
  $T^j = \text{PrincipalSupport}_k(w^j)$; (estimate for support set)
  $x^j_{T^j} = w^j_{T^j}$ and $x^j_{(T^j)^c} = 0$; (threshold to k-sparse vector with support T^j)
  $r^j = y - Ax^j$; (update the residual)
  $j = j + 1$; (increase the iteration counter)
end while
return $\hat{x} = x^k$ (return the k-sparse vector x^k)