CS168: The Modern Algorithmic Toolbox
Lecture #18: Compressive Sensing

Tim Roughgarden & Gregory Valiant[*]

May 31, 2024

1 Sparsity

Data analysis is only interesting when the data has structure — there's not much you can do with random noise. Ideally, when data has nice structure, it can be both detected and exploited algorithmically.

"Structure" can mean many things. One major theme in data analysis, implicit in many recent lectures, is *sparsity.* In the simplest case, the raw data — vectors in $\mathbb{R}^n$, say — are sparse, meaning that the vectors consist mostly of zeroes. For example, in Homework #2, when you looked at terms used in various newsgroups, you noticed that rather than explicitly storing all of the zeroes of a sparse vector, it's more efficient to store just the non-zero values and their coordinates.

In recent lectures, we've been thinking about data that becomes sparse after a suitable change of basis, meaning that in the new basis most of the coefficients of most vectors are zero. For example, last week we discussed looking at data in the Fourier basis (or "frequency domain"), and we saw that an audio recording of whistling a single note for 3 seconds was dense in the time domain but sparse (just 1-2 spikes) in the frequency domain. We knew to check the representation in the Fourier domain because of the physics of sound. Similarly, properties of real-world images suggest re-representing them in the wavelet basis (a basis similar to the Fourier basis but where the element of basis are localized in both frequency and time...this is mentioned briefly in the lecture for last week), where they are typically sparse.

Data is often only *approximately* sparse. Rather than being mostly zeroes, the data (perhaps after a change of basis) consist mostly of relatively small numbers. Such almost sparse data admit good compression and good de-noising — after expressing the data so that they are almost sparse, one can just zero out the small coefficients and remember only the large ones to get a compressed or de-noised version of the data. We've already seen a couple of examples of this paradigm.

For example, in principal components analysis (PCA), one doesn't have advance knowledge of the basis in which the data is sparse — the goal of the method is to examine the data set and identify the "best" basis for approximately representing it as linear combinations of a small set of k orthonormal vectors. The projection of the original data points onto the top k principal components can be be thought of as a sparse approximation (with only k non-zeros per data point in the new basis) of the original data set.

Low-rank matrix approximations — computing the singular value decomposition (SVD) of a matrix and retaining only its top k singular vectors and values — can also be viewed in this way. Here, matrix rank playing the role analogous to vector sparsity (recall rank k means only k non-zero singular values).

2 The Idea and Applications of Compressive Sensing

The usual approach to compressing approximately sparse data is to first collect the raw data, and then to compress it in software. For example, your smartphone initially captures an image in the standard pixel basis (with red, green, and blue intensities for each pixel). Subsequently, using a standard image compression algorithm (e.g., JPEG), the raw image is compressed, which typically reduces the image size by a large factor (15+). At a high level, such compression algorithms follow the paradigm outlined in Section 1 — expressing the image in a suitable basis (like a wavelet basis) so that it is almost sparse, and then retaining only the components that have sufficiently large coefficients. This two-step approach works fine in many applications, but it's hard not to wonder if the first step can be made more efficient. If most of the raw data is going to be thrown out immediately, before any processing, why did we bother to collect it all in the first place?

The main idea of *compressive sensing*, also called "compressed sensing," is to directly capture data in a compressed form. The theory of compressive sensing is less than fifteen years old, so the engineering viability of this idea is still being worked out in different application domains. At present, the key ideas seem well on their way to practical use and commercialization.

Here are some potential applications of compressive sensing — of collecting less information in cases where you know that the data is sparse in a suitable basis.

1. *Cameras that use less power.* The power used by your smartphone to capture an image, or a frame of a video, is driven by the number of analog-to-digital conversions that it has to do, which in turn is proportional to the number of pixels it uses. Compressive sensing suggests the possibilities of using fewer pixels without degrading image quality, resulting in qualitatively less battery drain for a given number of photos or videos. There was an early (meaning 2007) proof-of-concept prototype of a "single-pixel" camera built at Rice [4]; recently, new developments in chip fabrication suggest that commercialization of these ideas could occur in the near future.

2. *MRI and tomography.* In magnetic resonance imaging (MRI), a powerful magnet and radio waves are used to image parts of the human body (especially tendons, cartilage,

joints, etc.). The scan time is proportional to the number of measurements (or "slices") taken, and can easily be 30 minutes or more. Compressive sensing techniques have been tried out in some hospitals, and they have sped up scan times by a constant factor. This may not matter much for an MRI on an athlete's knee, but when the patient is a child and/or is not allowed to breathe during the scan (e.g., for a lung scan), a constant factor speed-up can be a big deal.

3. *Demystifying undersampling.* There have been examples in different fields where accurate inference is possible with far less information that would seem to be necessary. One example is in geophysics, in the business of drilling for oil. In the late 1970s and 1980s, geophysicists started trying to figure out what layers of rock strata looked like by sending seismic waves into the ground, at various frequencies.[1] Such seismic waves pass through homogeneous rock without much ado, but when they hit a discontinuity between different layers of rock — also where oil deposits are more likely to be — there is a reflection back to the surface.[2] Even with very few samples — meaning sending waves at just a few different frequencies — the geophysicists obtained quite accurate reconstructions of the rock layers (which could be confirmed during later drilling). They even came up with the linear programming-based algorithm that we'll discuss later! The working hypothesis for the efficacy of using few samples was that the structure of the rock layers was typically "simple;" compressive sensing provides a mathematical formulation and proof of this intuition.

3 The Setup

We give a mathematical description of the compressive sensing problem. At a high level, the goal is to design a small number of "linear measurements" such that their results enable the unique and efficient reconstruction of an unknown sparse signal. After this description, we'll relate it back to the applications in Section 2.

Step 1: Design m "linear measurements" $\mathbf{a}_1, \ldots, \mathbf{a}_m \in \mathbb{R}^n$.

Step 2: "Nature" picks an unknown "signal," meaning an n-vector $\mathbf{z} \in \mathbb{R}^n$.

Step 3: Receive the measurement results $b_1 = \langle \mathbf{a}_1, \mathbf{z} \rangle, b_2 = \langle \mathbf{a}_2, \mathbf{z} \rangle, \ldots b_m = \langle \mathbf{a}_m, \mathbf{z} \rangle$.

Step 4: Recover the n-vector $\mathbf{z}$ from the m-vector $\mathbf{b}$.[3]

Note that the goal is to design a *single* set of measurements $\mathbf{a}_1, \ldots, \mathbf{a}_m$ that "works" for *all* signals $\mathbf{z}$, in the sense that Step 4 should be possible.

[1]For example, by setting off some dynamite!

[2]Think of light going from air to water — some of it reflects back, some of it refracts in the new medium.

[3]There is a loose analogy between this setup and that of error-correcting codes. With codes, one designs an encoding function (analogous to our mapping $\mathbf{z} \to \mathbf{b}$) such that, given any codeword that hasn't been corrupted too much, one can uniquely and computationally efficiently decode (for us, map $\mathbf{b}$ back to $\mathbf{z}$).

For instance, in the camera example, we think of $\mathbf{z}$ as the raw image, in the standard pixel basis. Then, each $\mathbf{a}_i$ specifies a linear combination of the pixels. Various approaches are being explored for implementing such linear combinations in hardware. In Rice's single-pixel camera, they were implemented using tiny mirrors [4]. More recently, researchers have devised on-chip hardware tailored for such linear combinations [9].

In the tomography example, $\mathbf{z}$ is the "true image" of whatever's being scanned. Medical imaging devices already report what are essentially linear measurements of $\mathbf{z}$ — this is in the spirit of the linearity of the Fourier transform discussed last week, with a different but analogous transform. (We'll return to the geophysicists in a bit.)

4 Recovering Sparse Signals: Main Result and Interpretations

In the fourth and final step of the setup, the goal translates to solving a linear system $\mathbf{Ax} = \mathbf{b}$ in the variables $\mathbf{x}$, where the $m \times n$ constraint matrix

$$\mathbf{A} = \begin{bmatrix} — & \mathbf{a}_1 & — \\ — & \mathbf{a}_2 & — \\ & \vdots & \\ — & \mathbf{a}_m & — \end{bmatrix}.$$

was designed (or hard-coded) in advance, and the right-hand size $\mathbf{b} = \mathbf{Az}$ is given. Solving linear systems is a very well-studied problem and you've likely seen algorithms for it, for example Gaussian elimination.[4]

Can we design $\mathbf{A}$ so that it's easy to recover $\mathbf{z}$ from $\mathbf{b}$? Without further restrictions, this is a silly question, and the answer is obviously yes. Taking $\mathbf{A} = \mathbf{I}$, the $n \times n$ identity matrix, $\mathbf{b} = \mathbf{Az}$ will simply be $\mathbf{z}$ — the unknown signal is handed to us on a silver platter. In this solution, the ith linear measurement $\langle \mathbf{a}_i, \mathbf{z} \rangle$ returns the ith coordinate of $\mathbf{z}$, so n linear measurements are used.

Recall that our goal is to minimize the number of measurements used — corresponding to the power drain in a smartphone camera or the scan time of a MRI machine. We've seen that n linear measurements are sufficient; can we get away with fewer? The issue with taking $m < n$ is that the linear system $\mathbf{Ax} = \mathbf{b}$ becomes underdetermined. While $\mathbf{z}$ remains a feasible solution of the linear system (by the definition of $\mathbf{b}$), every vector $\mathbf{z} + \mathbf{w}$ with $\mathbf{w}$ in the $(n - m)$-dimensional kernel of $\mathbf{A}$ is also a solution.[5] This means we can't reconstruct $\mathbf{z}$, since we have no idea which of the infinitely many solutions to $\mathbf{Ax} = \mathbf{b}$ is the real unknown signal.

Summarizing, to reconstruct arbitrary unknown signals $\mathbf{z} \in \mathbb{R}^n$, n linear measurements are both necessary and sufficient. But recall the discussion in Section 1 — many signals of

[4]The singular value decomposition (SVD) also leads to a good algorithm for solving linear systems; see e.g. [5].

[5]Recall that the kernel of a matrix $\mathbf{A}$ is the set of vectors $\mathbf{w}$ such that $\mathbf{Aw} = \mathbf{0}$.

interest are (approximately) sparse in a suitable basis. Perhaps we can get away with fewer than n measurements if we only care about reconstructing sparse signals?

For the rest of lecture, we assume that the unknown signal $\mathbf{z}$ is *k-sparse* in the standard basis, meaning that it has at most k non-zero coordinates. For example, $\mathbf{z}$ could be a black and white image, with a black background (of 0-pixels) with lines of white (1) pixels. The parameter k could be anything between 0 and n, but you should think of it as much less than n, for example $\sqrt{n}$ or even $\log_2 n$. We focus on sparsity in the standard basis for simplicity only. Importantly, everything we'll say below also applies when $\mathbf{z}$ is k-sparse in a different basis, for example the Fourier basis (as with the audio of the whistle). The only difference is that one needs to throw in an extra change-of-basis matrix into some of the statements below.

The main theorem in compressive sensing, at least for the purposes of this lecture, provides a resounding "yes" to the question of whether or not signal sparsity allows for fewer linear measurements.

Theorem 4.1 (Main Theorem [2, 3]) *Fix a signal length n and a sparsity level k. Let $\mathbf{A}$ be an $m \times n$ matrix with $m = \Theta(k \log \frac{n}{k})$ rows, with each of its mn entries chosen independently from the standard Gaussian distribution.*[6] *With high probability over the choice of $\mathbf{A}$, every k-sparse signal $\mathbf{z}$ can be efficiently recovered from $\mathbf{b} = \mathbf{A}\mathbf{z}$.*

In Theorem 4.1, a single matrix $\mathbf{A}$ — a fixed set of $m = \Theta(k \log \frac{n}{k})$ linear measurements — "works" simultaneously for all k-sparse signals $\mathbf{z}$, allowing efficient recovery of $\mathbf{z}$ from $\mathbf{b}$. (Even though there's zillions of different solutions to $\mathbf{Ax} = \mathbf{b}$!) Note that for all interesting values of k, the number m of measurements used is way less than n — so signal sparsity indeed radically reduces the number measurements needed. For example, if n is a million, with $k = \sqrt{n}$ only tens of thousands of measurements are needed, and with $k = \log_2 n$ only hundreds are needed.

The proof of Theorem 4.1 is beyond the scope of this course; we'll focus instead on interpreting it and discussing the algorithms used for recovery. Some important comments:

1. Theorem 4.1 extends to "almost k-sparse" signals $\mathbf{z}$. This is crucial for practical applications — approximately sparse signals are far more common than exactly sparse ones. By "approximately k-sparse," we mean that the sum of the magnitudes of the k largest coordinates of $\mathbf{z}$ is much larger than the sum of the magnitudes of the rest of the coordinates of $\mathbf{z}$. The recovery guarantee changes, as you might expect: if $\mathbf{z}$ is only approximately k-sparse, then the algorithm returns a vector $\mathbf{z}'$ that is only approximately equal to $\mathbf{z}$. (If you're lucky, you might even get back a sparser, "de-noised" version of $\mathbf{z}$.) The guarantee degrades gracefully: for $\mathbf{z}$'s further and further away from sparse vectors, the guarantee for the returned $\mathbf{z}'$ grows weaker and weaker. In the extreme case, where $\mathbf{z}$ isn't sparse at all, then since $m < n$ our previous discussion implies that the returned vector $\mathbf{z}'$ might have nothing to do with $\mathbf{z}$.

[6]You've seen this matrix before, in a quite different context — the Johnson-Lindenstrauss (JL) Lemma for Euclidean distance-preserving dimensionality reduction (Lecture #4). While compressive sensing and dimensionality reduction seems like quite different goals, there are deep connections between them (see [1, 7]).

2. The number $m = \Theta(k \log \frac{n}{k})$ of linear measurements used in Theorem 4.1 is optimal up to a constant factor, at least if one also wants the approximate recovery of approximately sparse vectors.[7]

   For some intuition about where the mysterious $\Theta(k \log \frac{n}{k})$ term comes from, recall the proof that every comparison-based sorting algorithm requires $\Omega(n \log n)$ comparisons: a correct algorithm distinguishes between the $n!$ possible relative orderings of the n input elements, and each comparison might be resolved in a way that only reduces the number of remaining possibilities by a factor of 2 (or less). Thus, at least $\log_2(n!) = \Theta(n \log n)$ comparisons are needed in the worst case. In the present context, there are an infinite number of possibilities for the unknown k-sparse vector $\mathbf{z}$, but if we bucket these by support (i.e., the set of non-zero coordinates) then there are $\binom{n}{k}$ different possibilities that our linear measurements must differentiate. Thinking of a linear measurement as analogous to a comparison, we might guess that at least $\log_2 \binom{n}{k}$ measurements are required to correctly reconstruct all possible k-sparse signals. (You can use Stirling's approximation to verify that $\log_2 \binom{n}{k} = \Theta(k \log \frac{n}{k})$.) This analogy is not entirely accurate, as each linear measurement returns a real value, not just a bit. (This is why smaller values of m, even $m = 2k$, are sufficient for recovering exactly sparse vectors.) Still, a non-trivial proof shows that for recovering approximately sparse vectors, $m = \Theta(\log_2 \binom{n}{k})$ is indeed the correct answer.

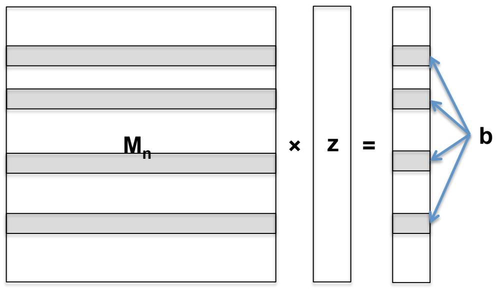
3. The assumption that the entries of $\mathbf{A}$ are i.i.d. Gaussians is not crucial — many matrices $\mathbf{A}$ work, and there is now a reasonable understanding of such matrices. Indeed, if the linear measurements are implemented in hardware, then it may be infeasible to implement Gaussian coefficients. In the camera application, it makes sense to use coefficients with fewer bits, ideally just in $\{-1, 0, 1\}$. In the MRI example, one doesn't even have the luxury of using random linear measurements — instead one is stuck with the measurements performed by the MRI machine, which are similar to the rows of the Fourier matrix $\mathbf{M}_n$ discussed last week (Figure 1(a)). Fortunately, there are slightly weaker versions of Theorem 4.1 that also apply to these sorts of linear measurements.[8]

   One type of matrix $\mathbf{A}$ that certainly *doesn't* work is a sparse matrix. For example, imagine that $\mathbf{A}$ is a random subset of not too many rows of the identity matrix $\mathbf{I}_n$ (Figure 1(b)). $\mathbf{Az}$ is then just the projection onto the coordinates of $\mathbf{z}$ corresponding to the chosen rows of $\mathbf{I}_n$. Plenty of sparse vectors $\mathbf{z}$ will be 0 in every one of these coordinates, in which case $\mathbf{b} = 0$. There is no hope of reconstructing $\mathbf{z}$ from $\mathbf{b}$ in this case.

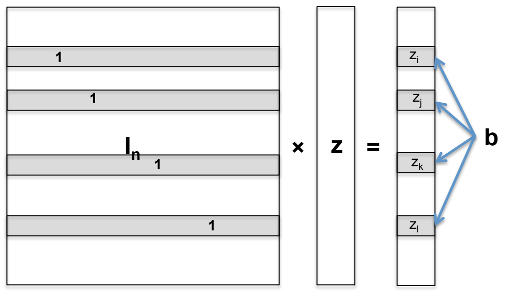
   Roughly, generalized versions of Theorem 4.1 say that as long as $\mathbf{A}$ is sufficiently dense and not pathological, it enables the recovery of sparse signals. More generally, if $\mathbf{z}$ is

[7]For the weaker and less practically relevant goal of reconstructing only exactly k-sparse vectors, it turns out that $m = 2k$ measurements are necessary and sufficient (see [8, Section 4.4])!

[8]Precisely, randomly choosing m rows of the Fourier matrix (with m a bit larger than k, but much less than n) yields, with high probability, a matrix $\mathbf{A}$ such that every k-sparse vector $\mathbf{z}$ can be recovered from $\mathbf{b} = \mathbf{Az}$.



(a) Rows of the Fourier matrix



(b) Rows of the identity matrix

Figure 1: Randomly sampling a small number of rows from the (dense) Fourier matrix M_n enables the recovery of all sparse signals; not so for the (sparse) identity matrix I_n.

a signal that is sparse in some other basis, and $\mathbf{A}$ is dense and non-pathological after changing to the relevant basis, then $\mathbf{z}$ can be recovered from the linear measurements in $\mathbf{A}$.

A related concept is the *uncertainty principle*, which states that no signal $\mathbf{z}$ can be "localized" (i.e., made sparse) simultaneously in both the time and frequency domains. We saw a vivid illustration of this last week: the whistle audio signal was sparse in the frequency domain, but dense in the time domain; while the series of percussive noises was sparse in the time domain (with one spike for each noise) but dense in the frequency domain. The uncertainty principle is a rigorous result stating that *every* signal is dense in the time domain or in the frequency domain (or both).

Now, back to the geophysicists. Their unknown signal $\mathbf{z}$ was the structure of rock strata, and it was typically sparse in the spatial domain (analogous to percussive noises) — mostly chunks of homogeneous rock, separated from each other by the occasional discontinuity (Figure 2). Their measurements — seismic waves — were sparse in the frequency domain. By the uncertainty principle, when these measurements are re-represented in the spatial domain — the basis in which the signal is sparse — they are dense (and presumably not pathological). Theorem 4.1 and its generalizations suggest that relatively few measurements should suffice for recovering the signal, corroborating what was observed empirically.

5 Recovery via ℓ_1 Minimization

5.1 Recovery via Optimization

Next, we study how to recover a k-sparse signal $\mathbf{z}$ from $\mathbf{b} = \mathbf{A}\mathbf{z}$, where $\mathbf{A}$ is chosen as in Theorem 4.1. Since $\mathbf{A}$ has many fewer rows than columns (recall $m = \Theta(k \log \frac{n}{k})$), $\mathbf{A}\mathbf{x} = \mathbf{b}$ is an underdetermined linear system with tons of solutions ($\mathbf{z}$, and lots of others). We need some principled way to select one feasible solution out of the many — hopefully it will be $\mathbf{z}$.

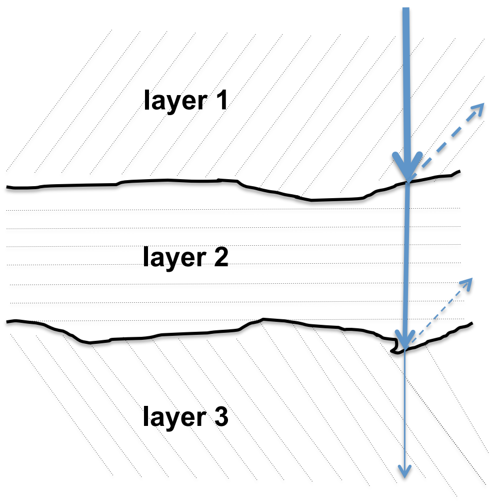


Figure 2: The structure of rock strata is often pretty simple. When a seismic wave hits a discontinuity, there is both a reflection and a refraction.

A natural and flexible way to do this is via optimization — to posit an objective function defined on the feasible solutions to $\mathbf{Ax} = \mathbf{b}$ and single out the "best" one.

More precisely, for some real-valued function $f : \mathbb{R}^n \to \mathbb{R}$, we consider optimization problems of the form

$$\min f(\mathbf{x})$$

subject to

$$\mathbf{Ax} = \mathbf{b}.$$

(We could also maximize a function f, but this is equivalent to minimizing $-f$.) We talked a bit about optimization before, in Week 3 in the context of generalization. There, we only considered *unconstrained* optimization (like finding regression coefficients), whereas here we need to constrain the recovered signal $\mathbf{x}$ to be consistent with the observed measurements $\mathbf{b}$.[9]

What objective function f should we use? All we know about the unknown signal $\mathbf{z}$ is that it satisfies $\mathbf{Az} = \mathbf{b}$ and that $\mathbf{z}$ is k-sparse. This suggests that we want to look for the sparsest solution to $\mathbf{Ax} = \mathbf{b}$:

$$\min |\text{supp}(\mathbf{x})|$$

subject to

$$\mathbf{Ax} = \mathbf{b},$$

where $\text{supp}(\mathbf{x})$ denotes the *support* of $\mathbf{x}$, meaning the set of coordinates for which it is non-zero. The quantity $|\text{supp}(\mathbf{x})|$ is sometimes called the ℓ_0 norm of $\mathbf{x}$, so we can call this problem *ℓ_0-minimization*.

[9]There are various ways to turn constrained optimization problems into unconstrained ones (like Lagrangian relaxation), but more direct solutions are preferred for most sparse recovery problems.

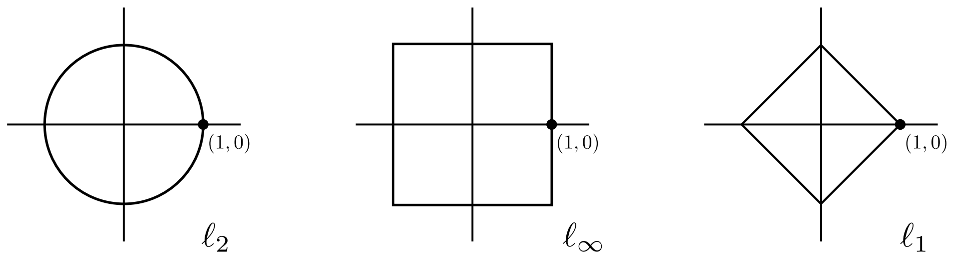


Figure 3: The unit balls in $\mathbb{R}^2$ under the ℓ_2, ℓ_∞, and ℓ_1 norms.

Unfortunately, ℓ_0-minimization is NP-hard [6]. So the next idea is to replace the ℓ_0 norm with a more tractable norm. An important lesson of this lecture is that minimizing the ℓ_1 norm, $\|\mathbf{x}\|_1 = \sum_{j=1}^n |x_j|$, has two non-obvious and very useful properties:

(1) Minimizing the ℓ_1 norm of a solution promotes sparsity.

(2) The ℓ_1 norm can be minimized by computationally efficient algorithms.

In comparison, the ℓ_0 norm satisfies property (1) but not (2). Minimizing the ℓ_2 norm satisfies property (2) — the singular value decomposition (SVD) discussed in Lecture #9 can be used to give a slick solution — but as we'll see shortly, this does not lead to property (1).

5.2 Geometry with the ℓ_1 Norm

We next explain property (1), that ℓ_1 minimization tends to promote sparse solutions. First we recall an exercise we did way back in Lecture #3, comparing the unit balls under various norms. See Figure 3 for an illustration in 2 dimensions.

Imagine that you tied a dog to a pole at the origin using a leash with unit ℓ_p norm. With $p = 2$, the dog can roam the same amount in every direction. With $p = \infty$, the dog can travel the farthest (in ℓ_2 distance) in the northwest, northeast, southwest, and southeast directions. With $p = 1$, the dog can stray farthest (in ℓ_2 distance) along the standard basis vectors.

Now consider the set $\{\mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{b}\}$ of feasible solutions to a linear system. In our context, this set is the $(n - m)$-dimensional kernel of $\mathbf{A}$, shifted by $\mathbf{z}$ (an affine subspace). See Figure 4. Let's examine the optimal solution of the optimization problem

$$\min \|\mathbf{x}\|_p$$

subject to

$$\mathbf{A}\mathbf{x} = \mathbf{b},$$

as a function of the choice of norm p. Geometrically, we'll think about blowing up a balloon centered at the origin, where the shape of the balloon is that of the unit ball in the given norm. If we've blown up the balloon to a radius (in ℓ_p norm) of r, then the intersection of

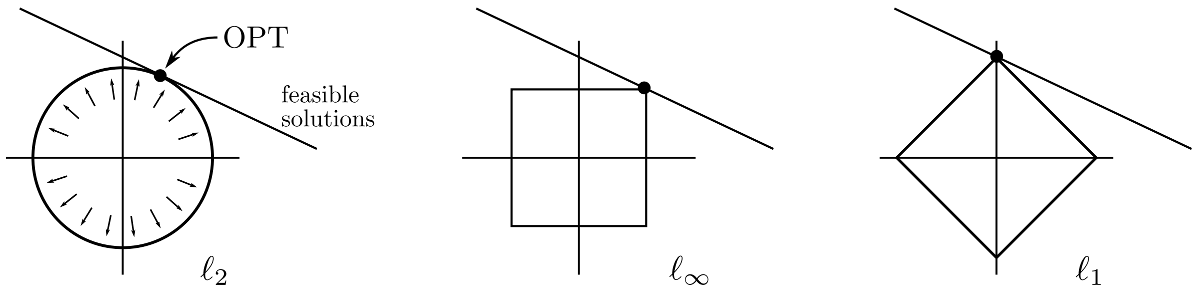


Figure 4: For a given choice of norm, the minimum-norm feasible solution is the first point of contact between a blown-up balloon and the affine subspace of feasible solutions.

the balloon and $\{\mathbf{x} : \mathbf{Ax} = \mathbf{b}\}$ is exactly the set of norm-r solutions to $\mathbf{Ax} = \mathbf{b}$. Thus, the minimum-norm solution to $\mathbf{Ax} = \mathbf{b}$ is the first point of contact between the balloon, as we blow it up, and the affine subspace of feasible solutions.

For example, consider the case depicted in Figure 4, where points live in two dimensions and the feasible region is one-dimensional (a line). With $p = 2$, where the balloon is expanding at the same rate in all directions, the minimum-norm feasible solution is the point that forms a line with the origin perpendicular to the feasible region. With $p = \infty$, the relevant direction in which the balloon is expanding the quickest is northeast, so the optimal point is the intersection of this direction with the feasible region. With $p = 1$, the ball is expanding the quickest along the standard basis vectors, so the first intersection occurs along the y-axis. The optimal solution under the $p = 2, \infty$ norms have full support (non-zero x- and y-coordinates), while the optimal solution under the ℓ_1 norm has a support size of only 1. The reader is encouraged to repeat this thought experiment in three dimensions — where the ℓ_1 ball is an octahedron — with one- and two-dimensional subspaces of feasible solutions.

Summarizing, because the ℓ_1 ball expands more quickly in directions with smaller support than in directions with larger support, it makes sense that minimizing the ℓ_1 norm generally favors sparser solutions. Indeed, ℓ_1 minimization is precisely the algorithm used for the guarantee in Theorem 4.1. We conclude the lecture by sharpening the theorem statement to take this into account.

Theorem 5.1 (Main Theorem, Sharpened [2, 3]) *Fix a signal length n and a sparsity level k. Let $\mathbf{A}$ be an $m \times n$ matrix with $m = \Theta(k \log \frac{n}{k})$ rows, with each of its mn entries chosen independently from the standard Gaussian distribution. With high probability over the choice of $\mathbf{A}$, for every k-sparse signal $\mathbf{z}$, the unique optimal solution to the ℓ_1-minimization problem*

$$\min \|\mathbf{x}\|_1$$

subject to

$$\mathbf{Ax} = \mathbf{b}$$

is $\mathbf{z}$.

References

[1] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28(3):253–263, 2008.

[2] E. J. Candes, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.

[3] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

[4] M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, T. Sun, K. F. Kelly, and R. G. Baraniuk. Single-pixel imaging via compressive sampling. *IEEE Signal Processing Magazine*, 25(2):83–91, 2008.

[5] G. H. Golub and C. F. van Loan. *Matrix Computations*. Johns Hopkins University Press, 1996. Third Edition.

[6] L. Khachiyan. On the complexity of approximating extremal determinants in matrices. *Journal of Complexity*, 11(1):138–153, 1995.

[7] F. Krahmer and R. Ward. New and improved johnson-lindenstrauss embeddings via the restricted isometry property. *SIAM Journal on Mathematical Analysis*, 43(3):1269–1281, 2011.

[8] A. Moitra. Algorithmic aspects of machine learning. Lecture notes, 2014.

[9] Y. Oike and A. El Gamal. CMOS Image Sensor With Per-Column ADC and Programmable Compressed Sensing. *IEEE Journal of Solid-State Circuits*, 48(1):318–328, 2012.