1 Preliminaries

A signal or data point will always be a vector $\vec{x} \in \mathbb{R}^D$, where $D$ is usually large!

2 Overview

In this lecture we introduce the technique of Compressive Sensing, and motivate it by means of concrete examples.

We will focus on two problems:

1. Given a huge set of data points $X = \{\vec{x}_1, \ldots, \vec{x}_N\} \subseteq \mathbb{R}^D$, where $N$ and $D$ are very large, how can we accurately and efficiently summarize $X$?

   By summarizing we generally mean approximating by some fit model. In simple cases, this could be regression, interpolation by a smooth function, or a manifold model. This problem is very related to data compression.

2. Given a manifold model - possibly some model from (1) we learned from a training set - how can we efficiently and accurately project a given vector $\vec{x} \in \mathbb{R}^D$ onto the model, using “reduced information”?

Here’s an example for the second problem.

Example 1. Suppose we can gather only a small number, $m$, of inner products of $\vec{x} \in \mathbb{R}^D$, where $m \ll D$. That is, for fixed measurement vectors $\vec{a}_1, \ldots, \vec{a}_m \in \mathbb{R}^D$, we get to see only

   $\langle \vec{a}_1, \vec{x} \rangle, \ldots, \langle \vec{a}_m, \vec{x} \rangle$.

How well can we possibly approximate $\vec{x}$ given that it sits on a given manifold model? In other words, given the information $\{\langle \vec{a}_i, \vec{x} \rangle\}_{i=1}^m$, find a point on the manifold model which best approximates $\vec{x}$.

This is a slight generalization of compressive sensing!

3 Manifold Models: from Simple to more Complex

3.1 Affine linear subspace of $\mathbb{R}^D$ of dimension $d$

The goal is to find an affine linear subspace $\vec{a} + S$, given a set of data points $X$, then use it find a representation of a new point not in $X$. In more details,
1. Find $S$, the “best fit” $d$-dimensional subspace, $S$, for $X = \frac{1}{N} \sum_{j=1}^{N} \vec{x}_j$, then let
\[
\vec{a} = \Pi_{S^\perp} \left( \frac{1}{N} \sum_{j=1}^{N} \vec{x}_j \right),
\]
where $\Pi_K$ is the operator that projects onto the subset $K$. (Here $\vec{a} \in \mathbb{R}^D \cap S^\perp$ is a shift.)

2. Projection problem: given $\vec{y} \in \mathbb{R}^D$, not in $X$, we project it onto $\vec{a} + S$ by
\[
\Pi_S (\vec{y} - \vec{a}) + \vec{a}.
\]

3.2 Smooth $d$-dimensional submanifold of $\mathbb{R}^D$

Again, this is a two-step process: manifold learning, and finding an efficient way to project a vector $\vec{x}$ onto the manifold.

3.3 Sparsity

Denote $[D] := \{1, 2, \ldots, D\}$. For a given set $S \subseteq [D]$, with cardinality $|S| = d$, let
\[
A_S = \text{Span}\{\vec{e}_j | j \in S\},
\]
where $\vec{e}_j$ is a canonical basis vector. Thus a vector in the subspace $A_S$ has at most $d$ nonzero entries, indexed by $S$. Now let
\[
\mathcal{M} = \bigcup_{S \subseteq [D], |S| = d} A_S \subseteq \mathbb{R}^D
\]
The set $\mathcal{M}$, which contains $\binom{D}{d}$ $d$-dimensional subspaces, is the set of all possible vectors with at most $d$ nonzero entries.

The compressive sensing problem is to determine how to project $\vec{x}$ onto the manifold $\mathcal{M}$, given a few inner products. This naively exponentially hard problem can be remarkably be solved in only polynomial time.

**Example 2. Sparse Interpolation of a periodic function**

Suppose
\[
f(x) = \sum_{w \in S} C_w \cdot e^{iwx}
\]
for some subset $S \subseteq [D]$ and large $D$, where $|S| = d \ll D$. The small number $d$ could correspond to the number of transmitters. Here $C_w \in \mathbb{C}$.

How many samples, or function evaluations $f(x_1), \ldots, f(x_m)$ do we need to learn $f$? Surely, we would like to use as few samples as possible. It turns out we can use radically fewer samples than $D$, and still learn $f$.

It is clear that we learn $f$ if and only if we learn all $C_w$’s, and $S \subseteq [D]$. Every function evaluation $f(x_j)$ is a linear combination of the constants $C_w$. Say
\[
f(x_j) = \langle \vec{C}, \vec{F}_{x_j} \rangle
\]
where $\vec{C} \in \mathbb{C}^D$ has $d$ nonzero entries equal to the $C_w$'s, in positions indexed by $S \subseteq D$, and $F_{x_j}$ is the $x_j^{th}$ column of an inverse Fourier transform matrix.

We get linear samples of $\vec{C} \in \mathbb{C}^D$ (by sampling), and we know that $\vec{C} \in \mathcal{M}$. The compressive sensing problem, in this noiseless setting, is to project $\vec{C}$ onto $\mathcal{M}$.

We will learn a lower bound on the number of samples needed to learn $f$. Moreover, we will learn how to deal with noise; that is, when

$$f(x) = \sum_{w \in S} C_w \cdot e^{iwx} + g(x),$$

where $g(x)$ is small in comparison to $f(x)$.

**Example 3. Sales Model (Heavy Hitters)**

Imagine we collect global sales information from all Walmart stores, and get updates such as

$$(-2 \text{ bubble gums}, -1 \text{ sodas}, \ldots)$$

when two bubble gums and one soda bottle are sold, and other updates such as

$$(+2000 \text{ bubble gums}, \ldots)$$

when a new shipment is received from a supplier to one of our many warehouses.

Let $D$ be the number of all products sold in any store, anywhere. $D$ is obviously large. Let $\vec{x} \in \mathbb{Z}_+^D$ represent the sum of all updates, sent to corporate headquarters, on a minute-by-minute basis.

**Goal:** In the first five seconds of each minute, we would like to identify the top one hundred selling items, and then raise their price by 1 cent. It is clear that we need to identify these one hundred items very fast!

In other words, we need to project $\vec{x}$ onto $\mathcal{M}$ quickly. In general, it is too slow, if $D$ is large enough, to update all of $\vec{x}$ and then use it to project onto $\mathcal{M}$ in a trivial way (for physical reasons, such as slowly spinning hard disks, etc., etc...).

To overcome this problem, we design a linear map $M \in \mathbb{R}^{m \times D}$, where $d < m \ll D$, and only store $M\vec{x} \in \mathbb{R}^m$. Then,

$$M(\vec{x} \text{ + update}) = M\vec{x} + M(\text{update}) \in \mathbb{R}^m.$$ 

We can use $M\vec{x}$ (as inner products) to project onto $\mathcal{M}$. For efficiency, we design $M$ so that $M(\text{update})$ is computed fast, and so that $M\vec{x}$ supports super fast projections onto $\mathcal{M}$, our manifold of all 100 sparse vectors.