COMPSCI 514: ALGORITHMS FOR DATA SCIENCE

Cameron Musco
University of Massachusetts Amherst. Fall 2019.
Lecture 23

LOGISTICS

- Problem Set 4 due Sunday 12/15 at 8pm.
- Exam prep materials posted under the 'Schedule' tab of the course page.
- SRTI survey is open until 12/22. Your feedback this semester has been very helpful to me, so please fill out the survey!
- `https://owl.umass.edu/partners/courseEvalSurvey/uma/`

Last Class:

- Some counterintuitive properties of high dimensional space.
- Connections to ‘curse of dimensionality’.

Last Class:

- Some counterintuitive properties of high dimensional space.
- Connections to ‘curse of dimensionality’.

This Class:

- Compressed sensing and sparse recovery.
- Applications to sparse regression, frequent elements problem, sparse Fourier transform, efficient imaging, etc.

LINEAR SYSTEMS

Consider matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ and $\mathbf{x} \in \mathbb{R}^d$. If you are given $\mathbf{b} = \mathbf{A}\mathbf{x}$, under what condition can you find $\mathbf{x}$?

$n > d$

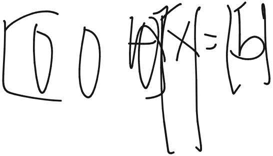
$\det(A) \neq 0$

inverse A exists

columns are linearly ind.

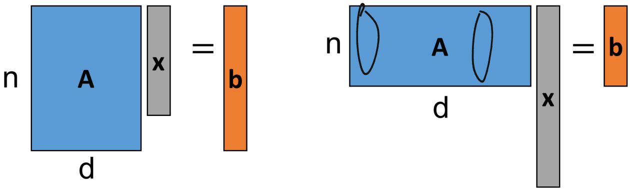
LINEAR SYSTEMS

Consider matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ and $\mathbf{x} \in \mathbb{R}^d$. If you are given $\mathbf{b} = \mathbf{A}\mathbf{x}$, under what condition can you find $\mathbf{x}$? When $\mathbf{A}$ has full column rank – i.e., all columns are linearly independent.

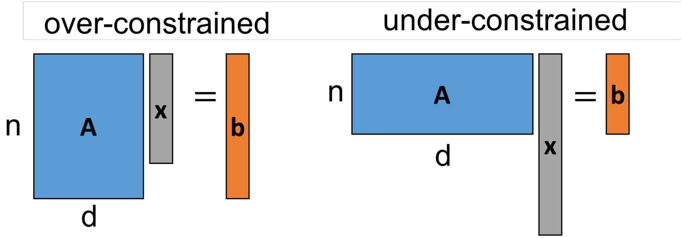


Consider matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ and $\mathbf{x} \in \mathbb{R}^d$. If you are given $\mathbf{b} = \mathbf{A}\mathbf{x}$, under what condition can you find $\mathbf{x}$? When $\mathbf{A}$ has full column rank – i.e., all columns are linearly independent.

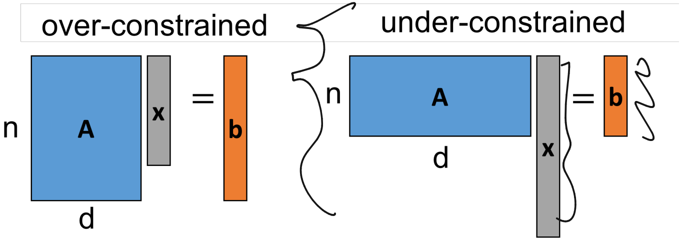
n > d



Consider matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ and $\mathbf{x} \in \mathbb{R}^{d}$. If you are given $\mathbf{b} = \mathbf{A}\mathbf{x}$, under what condition can you find $\mathbf{x}$? When $\mathbf{A}$ has full column rank – i.e., all columns are linearly independent.

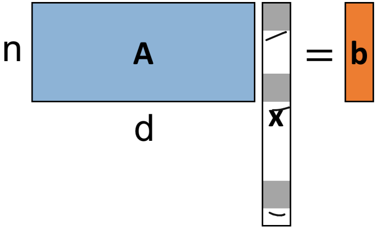


Consider matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ and $\mathbf{x} \in \mathbb{R}^{d}$. If you are given $\mathbf{b} = \mathbf{A}\mathbf{x}$, under what condition can you find $\mathbf{x}$? When $\mathbf{A}$ has full column rank – i.e., all columns are linearly independent.

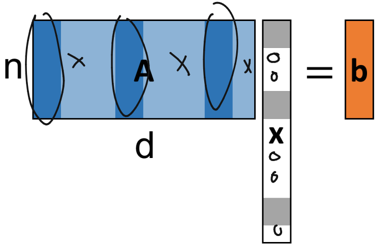


Compressed sensing: Under what assumptions we can still find $\mathbf{x}$ when the number of 'measurements' n is smaller than the number of features d (i.e., when $\mathbf{b}$ is a compression of $\mathbf{x}$)?

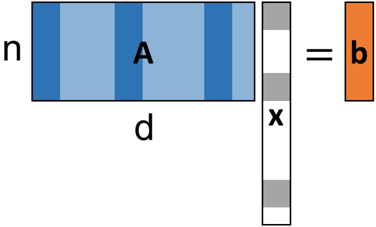
The most common assumption of compressed sensing is that $\mathbf{x}$ is *k-sparse*, i.e., has at most $k \ll d$ nonzero entries.



The most common assumption of compressed sensing is that **x** is *k*-sparse, i.e., has at most $k \ll d$ nonzero entries.

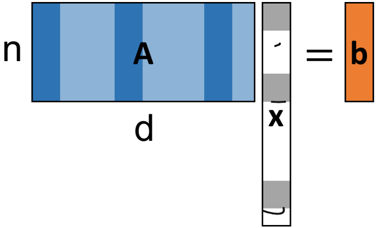


The most common assumption of compressed sensing is that $\mathbf{x}$ is *k-sparse*, i.e., has at most $k \ll d$ nonzero entries.



These types of linear systems have lots of applications.

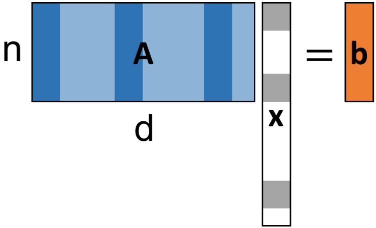
The most common assumption of compressed sensing is that $\mathbf{x}$ is *k-sparse*, i.e., has at most $k \ll d$ nonzero entries.



These types of linear systems have lots of applications.

First: Under what condition can you find $\mathbf{x}$ assuming knowledge that it is at most k-sparse?

The most common assumption of compressed sensing is that $\mathbf{x}$ is k-sparse, i.e., has at most $k \ll d$ nonzero entries.



These types of linear systems have lots of applications.

First: Under what condition can you find $\mathbf{x}$ assuming knowledge that it is at most k-sparse? When every set of $2k$ columns in $\mathbf{A}$ is linearly independent – i.e., $\mathbf{A}$ has Kruskal rank $2k$.

Sufficiency of Kruskal Rank $2k$:

all sets of 2k columns are lin. ind.

- We want to recover $\mathbf{x}$ from $\mathbf{b} = \mathbf{A}\mathbf{x}$, assuming $\mathbf{x}$ is k-sparse.

Sufficiency of Kruskal Rank $2k$:

- We want to recover $\mathbf{x}$ from $\mathbf{b} = \mathbf{A}\mathbf{x}$, assuming $\mathbf{x}$ is k-sparse.
- Say there was a different k-sparse $\mathbf{x}'$ with $\mathbf{A}\mathbf{x}' = \mathbf{b}$, making recovery impossible.

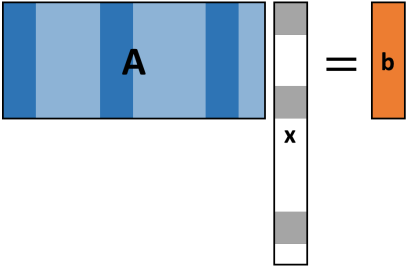
Sufficiency of Kruskal Rank $2k$:

- We want to recover $\mathbf{x}$ from $\mathbf{b} = \mathbf{A}\mathbf{x}$, assuming $\mathbf{x}$ is k-sparse.
- Say there was a different k-sparse $\mathbf{x}'$ with $\mathbf{A}\mathbf{x}' = \mathbf{b}$, making recovery impossible.
- Then $\mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{x}' = \mathbf{A}(\mathbf{x} - \mathbf{x}') = \mathbf{0}$.

  b - b

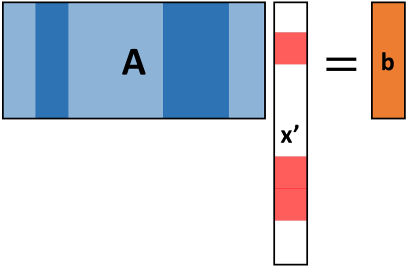
Sufficiency of Kruskal Rank $2k$:

- We want to recover $\mathbf{x}$ from $\mathbf{b} = \mathbf{A}\mathbf{x}$, assuming $\mathbf{x}$ is k-sparse.
- Say there was a different k-sparse $\mathbf{x}'$ with $\mathbf{A}\mathbf{x}' = \mathbf{b}$, making recovery impossible.
- Then $\mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{x}' = \mathbf{A}(\mathbf{x} - \mathbf{x}') = \mathbf{0}$.



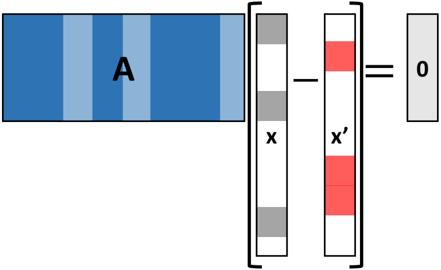
Sufficiency of Kruskal Rank $2k$:

- We want to recover $\mathbf{x}$ from $\mathbf{b} = \mathbf{A}\mathbf{x}$, assuming $\mathbf{x}$ is k-sparse.
- Say there was a different k-sparse $\mathbf{x}'$ with $\mathbf{A}\mathbf{x}' = \mathbf{b}$, making recovery impossible.
- Then $\mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{x}' = \mathbf{A}(\mathbf{x} - \mathbf{x}') = \mathbf{0}$.



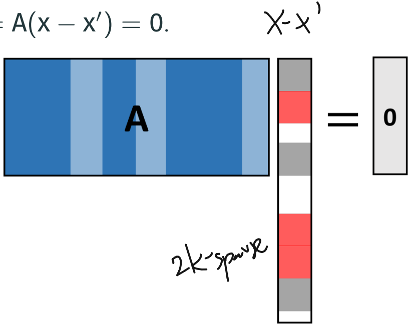
Sufficiency of Kruskal Rank $2k$:

- We want to recover $\mathbf{x}$ from $\mathbf{b} = \mathbf{A}\mathbf{x}$, assuming $\mathbf{x}$ is k-sparse.
- Say there was a different k-sparse $\mathbf{x}'$ with $\mathbf{A}\mathbf{x}' = \mathbf{b}$, making recovery impossible.
- Then $\mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{x}' = \mathbf{A}(\mathbf{x} - \mathbf{x}') = \mathbf{0}$.



Sufficiency of Kruskal Rank $2k$:

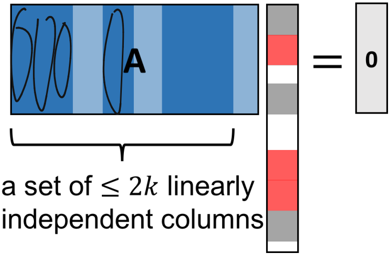
- We want to recover $\mathbf{x}$ from $\mathbf{b} = \mathbf{A}\mathbf{x}$, assuming $\mathbf{x}$ is k-sparse.
- Say there was a different k-sparse $\mathbf{x}'$ with $\mathbf{A}\mathbf{x}' = \mathbf{b}$, making recovery impossible.
- Then $\mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{x}' = \mathbf{A}(\mathbf{x} - \mathbf{x}') = \mathbf{0}$.



Sufficiency of Kruskal Rank $2k$:

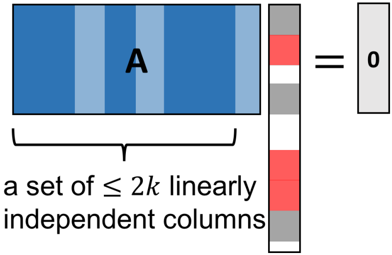
- We want to recover $\mathbf{x}$ from $\mathbf{b} = \mathbf{A}\mathbf{x}$, assuming $\mathbf{x}$ is k-sparse.
- Say there was a different k-sparse $\mathbf{x}'$ with $\mathbf{A}\mathbf{x}' = \mathbf{b}$, making recovery impossible.
- Then $\mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{x}' = \mathbf{A}(\mathbf{x} - \mathbf{x}') = \mathbf{0}$.

upto 2k sparse



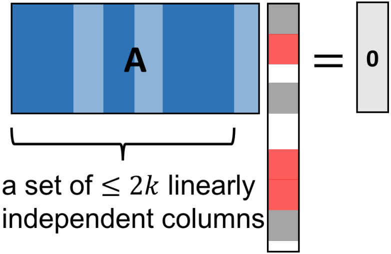
Sufficiency of Kruskal Rank $2k$:

- We want to recover $\mathbf{x}$ from $\mathbf{b} = \mathbf{A}\mathbf{x}$, assuming $\mathbf{x}$ is k-sparse.
- Say there was a different k-sparse $\mathbf{x}'$ with $\mathbf{A}\mathbf{x}' = \mathbf{b}$, making recovery impossible.
- Then $\mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{x}' = \mathbf{A}(\mathbf{x} - \mathbf{x}') = \mathbf{0}$. Violates Kruskal rank assumption.



Sufficiency of Kruskal Rank $2k$:

- We want to recover $\mathbf{x}$ from $\mathbf{b} = \mathbf{A}\mathbf{x}$, assuming $\mathbf{x}$ is k-sparse.
- Say there was a different k-sparse $\mathbf{x}'$ with $\mathbf{A}\mathbf{x}' = \mathbf{b}$, making recovery impossible.
- Then $\mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{x}' = \mathbf{A}(\mathbf{x} - \mathbf{x}') = \mathbf{0}$. Violates Kruskal rank assumption.



- Thus $\mathbf{x}$ is the unique k-sparse solution to $\mathbf{A}\mathbf{x} = \mathbf{b}$.

To satisfy the Kruskal rank $\geq 2k$ assumption $\mathbf{A}$ just needs $2k$ rows (compared with d rows to have full column rank). Can recover a d-dimensional k-sparse vector $\mathbf{x}$ from just $2k$ measurements $\mathbf{b} = \mathbf{A}\mathbf{x}$.

RECOVERY PROCEDURE

To satisfy the Kruskal rank $\geq 2k$ assumption $\mathbf{A}$ just needs $2k$ rows (compared with d rows to have full column rank). Can recover a d-dimensional k-sparse vector $\mathbf{x}$ from just $2k$ measurements $\mathbf{b} = \mathbf{A}\mathbf{x}$.

Assuming that $\mathbf{A}$ has Kruskal rank $\geq 2k$, how do we actually find the unique k-sparse solution $\mathbf{x}$ to $\mathbf{A}\mathbf{x} = \mathbf{b}$?

To satisfy the Kruskal rank $\geq 2k$ assumption $\mathbf{A}$ just needs $2k$ rows (compared with d rows to have full column rank). Can recover a d-dimensional k-sparse vector $\mathbf{x}$ from just $2k$ measurements $\mathbf{b} = \mathbf{A}\mathbf{x}$.

Assuming that $\mathbf{A}$ has Kruskal rank $\geq 2k$, how do we actually find the unique k-sparse solution $\mathbf{x}$ to $\mathbf{A}\mathbf{x} = \mathbf{b}$?

$$\mathbf{x} = \underset{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}}{\arg\min} \|\mathbf{z}\|_0,$$

$Ax' = b$

x' $\leq$ k-sparse

where $\|\mathbf{z}\|_0$ is the number of non-zero entries in $\mathbf{z}$.

To satisfy the Kruskal rank $\geq 2k$ assumption $\mathbf{A}$ just needs $2k$ rows (compared with d rows to have full column rank). Can recover a d-dimensional k-sparse vector $\mathbf{x}$ from just $2k$ measurements $\mathbf{b} = \mathbf{A}\mathbf{x}$.

Assuming that $\mathbf{A}$ has Kruskal rank $\geq 2k$, how do we actually find the unique k-sparse solution $\mathbf{x}$ to $\mathbf{A}\mathbf{x} = \mathbf{b}$?

$$\mathbf{x} = \arg\min_{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}} \|\mathbf{z}\|_0,$$

$\|\lambda z_1 + (1-\lambda) z_2\|_0 \leq \lambda \|z_1\|_0 + (1-\lambda)\|z_2\|_0$

$\geq k$ k

where $\|\mathbf{z}\|_0$ is the number of non-zero entries in $\mathbf{z}$.

This problem seems very difficult to solve. Why?

To satisfy the Kruskal rank $\geq 2k$ assumption $\mathbf{A}$ just needs $2k$ rows (compared with d rows to have full column rank). Can recover a d-dimensional k-sparse vector $\mathbf{x}$ from just $2k$ measurements $\mathbf{b} = \mathbf{A}\mathbf{x}$.

Assuming that $\mathbf{A}$ has Kruskal rank $\geq 2k$, how do we actually find the unique k-sparse solution $\mathbf{x}$ to $\mathbf{A}\mathbf{x} = \mathbf{b}$?

$$\mathbf{x} = \arg\min_{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}} \|\mathbf{z}\|_0,$$

where $\|\mathbf{z}\|_0$ is the number of non-zero entries in $\mathbf{z}$.

This problem seems very difficult to solve. Why? Non-convex.

To satisfy the Kruskal rank $\geq 2k$ assumption $\mathbf{A}$ just needs $2k$ rows (compared with d rows to have full column rank). Can recover a d-dimensional k-sparse vector $\mathbf{x}$ from just $2k$ measurements $\mathbf{b} = \mathbf{A}\mathbf{x}$.

Assuming that $\mathbf{A}$ has Kruskal rank $\geq 2k$, how do we actually find the unique k-sparse solution $\mathbf{x}$ to $\mathbf{A}\mathbf{x} = \mathbf{b}$?

$$\mathbf{x} = \arg\min_{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}} \|\mathbf{z}\|_0,$$

where $\|\mathbf{z}\|_0$ is the number of non-zero entries in $\mathbf{z}$.

This problem seems very difficult to solve. Why? Non-convex.

Exponential Time Algorithm:

RECOVERY PROCEDURE

To satisfy the Kruskal rank $\geq 2k$ assumption $\mathbf{A}$ just needs $2k$ rows (compared with d rows to have full column rank). Can recover a d-dimensional k-sparse vector $\mathbf{x}$ from just $2k$ measurements $\mathbf{b} = \mathbf{A}\mathbf{x}$.

Assuming that $\mathbf{A}$ has Kruskal rank $\geq 2k$, how do we actually find the unique k-sparse solution $\mathbf{x}$ to $\mathbf{A}\mathbf{x} = \mathbf{b}$?

$$\mathbf{x} = \arg\min_{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}} \|\mathbf{z}\|_0,$$

where $\|\mathbf{z}\|_0$ is the number of non-zero entries in $\mathbf{z}$.

This problem seems very difficult to solve. Why? Non-convex.

Exponential Time Algorithm: Loop through all $\binom{d}{k} = O(d^k)$ sparsity patterns and find the best $\mathbf{z}$ with the given sparsity pattern by solving a normal linear regression problem.

RECOVERY PROCEDURE

To satisfy the Kruskal rank $\geq 2k$ assumption $\mathbf{A}$ just needs $2k$ rows (compared with d rows to have full column rank). Can recover a d-dimensional k-sparse vector $\mathbf{x}$ from just $2k$ measurements $\mathbf{b} = \mathbf{A}\mathbf{x}$.

Assuming that $\mathbf{A}$ has Kruskal rank $\geq 2k$, how do we actually find the unique k-sparse solution $\mathbf{x}$ to $\mathbf{A}\mathbf{x} = \mathbf{b}$?

$$\mathbf{x} = \arg\min_{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}} \|\mathbf{z}\|_0,$$

where $\|\mathbf{z}\|_0$ is the number of non-zero entries in $\mathbf{z}$.

This problem seems very difficult to solve. Why? Non-convex.

Exponential Time Algorithm: Loop through all $\binom{d}{k} = O(d^k)$ sparsity patterns and find the best $\mathbf{z}$ with the given sparsity pattern by solving a normal linear regression problem.

A major accomplishment of compressed sensing/sparse recovery is to make the above procedure efficient and noise robust.

Example Applications

Example Applications

Caviat: Today we will only talk about sparse recovery without noise when $\mathbf{A}\mathbf{x} = \mathbf{b}$. In applications, it is important to be able to recover $\mathbf{x}$ from $\mathbf{b}$ with $\mathbf{A}\mathbf{x} = \mathbf{b} + \mathbf{n}$ for some small noise $\mathbf{n}$.

Example Applications

Caviat: Today we will only talk about sparse recovery without noise when $\mathbf{A}\mathbf{x} = \mathbf{b}$. In applications, it is important to be able to recover $\mathbf{x}$ from $\mathbf{b}$ with $\mathbf{A}\mathbf{x} = \mathbf{b} + \mathbf{n}$ for some small noise $\mathbf{n}$.

- The techniques discussed carry over to the noisy setting.
- Generally won't find $\mathbf{x}$ exactly, but up to some good approximation.

SPARSE REGRESSION

In high-dimensional data analysis, you often have a huge number of variables (genetic markers, characteristics of a user, etc.), possibly more than the number of data points.

In high-dimensional data analysis, you often have a huge number of variables (genetic markers, characteristics of a user, etc.), possibly more than the number of data points.

- Believe that just a few important features explain some phenomena (e.g., if a patient is likely to have a certain disease).

SPARSE REGRESSION

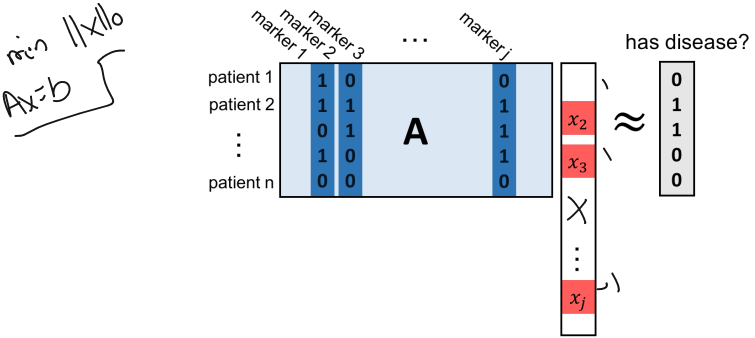
In high-dimensional data analysis, you often have a huge number of variables (genetic markers, characteristics of a user, etc.), possibly more than the number of data points.

- Believe that just a few important features explain some phenomena (e.g., if a patient is likely to have a certain disease).
- Want to find a linear regression model $\mathbf{A}\mathbf{x} \approx \mathbf{b}$ that only uses a small number of features ($\mathbf{x}$ is sparse).

SPARSE REGRESSION

In high-dimensional data analysis, you often have a huge number of variables (genetic markers, characteristics of a user, etc.), possibly more than the number of data points.

- Believe that just a few important features explain some phenomena (e.g., if a patient is likely to have a certain disease).
- Want to find a linear regression model $\mathbf{Ax} \approx \mathbf{b}$ that only uses a small number of features ($\mathbf{x}$ is sparse).



SPARSE REGRESSION

In high-dimensional data analysis, you often have a huge number of variables (genetic markers, characteristics of a user, etc.), possibly more than the number of data points.

- Believe that just a few important features explain some phenomena (e.g., if a patient is likely to have a certain disease).
- Want to find a linear regression model $Ax \approx b$ that only uses a small number of features (x is sparse).
- Interesting even in the over-constrained case. Often talked about as a different problem than compressed sensing, but very related.

Recall: The frequent elements problem asks us to return the k most frequent elements seen in a stream of items.

Recall: The frequent elements problem asks us to return the k most frequent elements seen in a stream of items.

- We saw how to (approximately) solve in $O(k)$ space using Misra-Gries or Count-Min Sketch.

Recall: The frequent elements problem asks us to return the k most frequent elements seen in a stream of items.

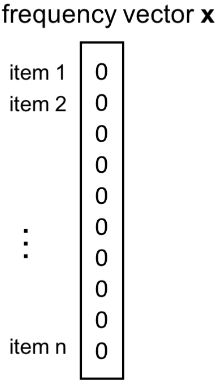
- We saw how to (approximately) solve in $O(k)$ space using Misra-Gries or Count-Min Sketch.
- Only work when frequencies are constantly incremented (we see more items over time). But what about when frequencies can be decremented?

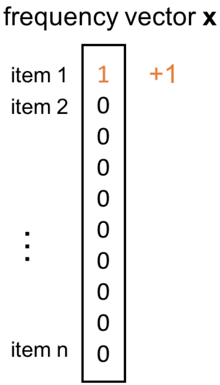
Recall: The frequent elements problem asks us to return the k most frequent elements seen in a stream of items.

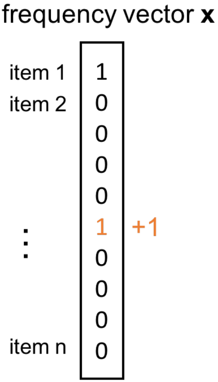
- We saw how to (approximately) solve in $O(k)$ space using Misra-Gries or Count-Min Sketch.
- Only work when frequencies are constantly incremented (we see more items over time). But what about when frequencies can be decremented?
- E.g., Amazon is monitoring what products people add to their "wishlist" and wants a list of most tagged products. Wishlists can be change over time, and items can be removed, decreasing their frequencies.

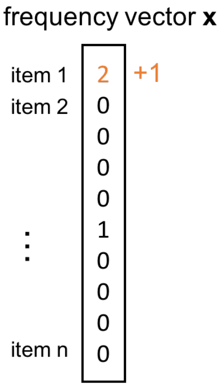
Recall: The frequent elements problem asks us to return the k most frequent elements seen in a stream of items.

- We saw how to (approximately) solve in $O(k)$ space using Misra-Gries or Count-Min Sketch.
- Only work when frequencies are constantly incremented (we see more items over time). But what about when frequencies can be decremented?
- E.g., Amazon is monitoring what products people add to their "wishlist" and wants a list of most tagged products. Wishlists can be change over time, and items can be removed, decreasing their frequencies.
- In this setting, the problem is solved with sparse recovery techniques.

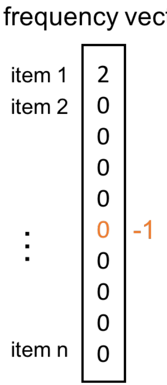

frequency vector **x**
item 1
item 2
⋮
item n
0
0
0
0
0
0
0
0
0
0

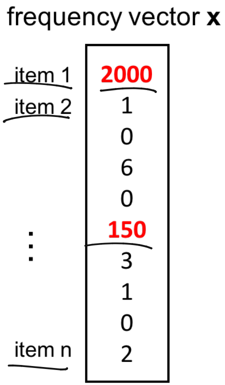

frequency vector x
item 1
item 2
item n
1
0
0
0
0
0
0
0
0
0
+1

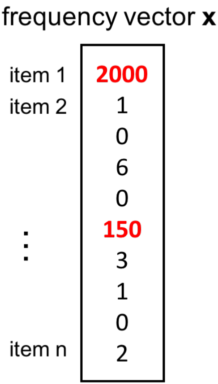

frequency vector **x**
item 1
item 2
⋮
item n
1
0
0
0
0
1
+1
0
0
0
0


frequency vector x
item 1
item 2
⋮
item n
2
0
0
0
0
1
0
0
0
0
+1

FREQUENT ITEMS COUNTING

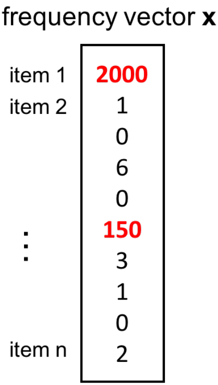



frequency vector **x**
item 1
item 2
item n
2000
1
0
6
0
150
3
1
0
2

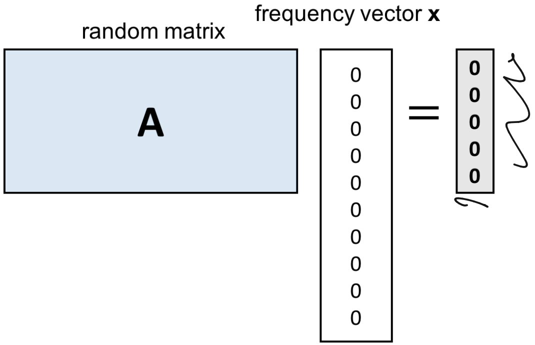


- Storing **x** requires $O(n)$ space. Will instead store **Ax** where **A** $\in \mathbb{R}^{O(k)\times n}$ is a random matrix.

FREQUENT ITEMS COUNTING

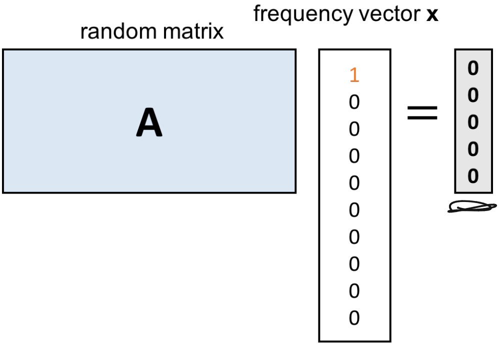


- Storing **x** requires $O(n)$ space. Will instead store **Ax** where $\mathbf{A} \in \mathbb{R}^{O(k)\times n}$ is a random matrix.
- **Ax** can be efficiently updated in a data stream.



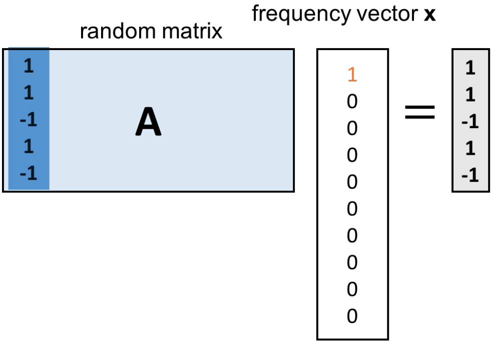
- Storing x requires $O(n)$ space. Will instead store Ax where $A \in \mathbb{R}^{O(k)\times n}$ is a random matrix.
- Ax can be efficiently updated in a data stream.

FREQUENT ITEMS COUNTING

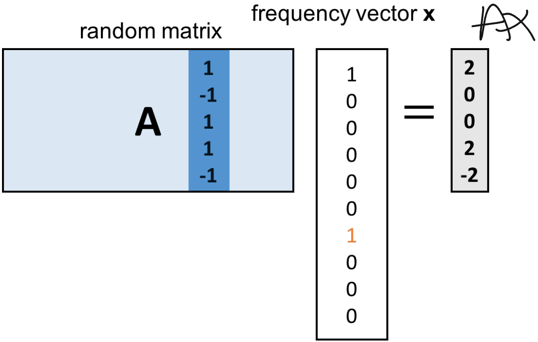


- Storing **x** requires $O(n)$ space. Will instead store **Ax** where **A** $\in \mathbb{R}^{O(k)\times n}$ is a random matrix.
- **Ax** can be efficiently updated in a data stream.

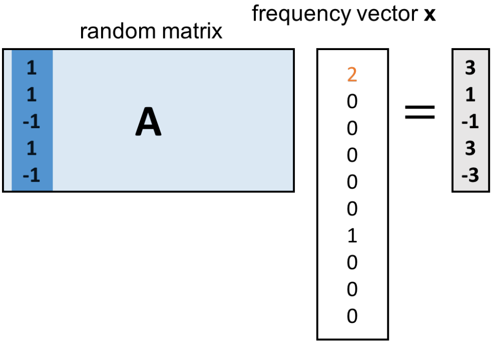
FREQUENT ITEMS COUNTING



- Storing **x** requires $O(n)$ space. Will instead store **Ax** where $\mathbf{A} \in \mathbb{R}^{O(k)\times n}$ is a random matrix.
- **Ax** can be efficiently updated in a data stream.

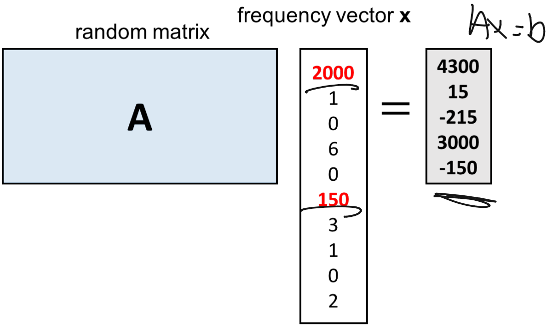


- Storing $\mathbf{x}$ requires $O(n)$ space. Will instead store $\mathbf{A}\mathbf{x}$ where $\mathbf{A} \in \mathbb{R}^{O(k)\times n}$ is a random matrix.
- $\mathbf{A}\mathbf{x}$ can be efficiently updated in a data stream.



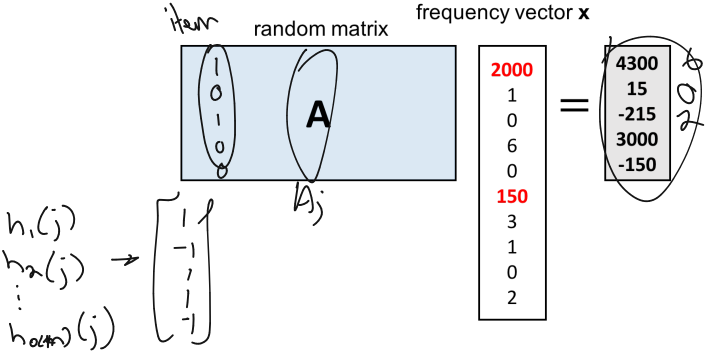
- Storing **x** requires $O(n)$ space. Will instead store **Ax** where **A** $\in \mathbb{R}^{O(k)\times n}$ is a random matrix.
- **Ax** can be efficiently updated in a data stream.

FREQUENT ITEMS COUNTING



- Storing **x** requires $O(n)$ space. Will instead store **Ax** where **A** $\in \mathbb{R}^{O(k)\times n}$ is a random matrix.
- **Ax** can be efficiently updated in a data stream.

FREQUENT ITEMS COUNTING



- Storing **x** requires $O(n)$ space. Will instead store **Ax** where $\mathbf{A} \in \mathbb{R}^{O(k)\times n}$ is a random matrix.
- **Ax** can be efficiently updated in a data stream.
- If there are a k heavy items, **x** is approximately k-sparse.
- Estimating the large entries of **x** (the counts of the most frequent items) from the compression **Ax** is exactly sparse recovery.

Many of the most important applications of sparse recovery are in imaging and signal processing.

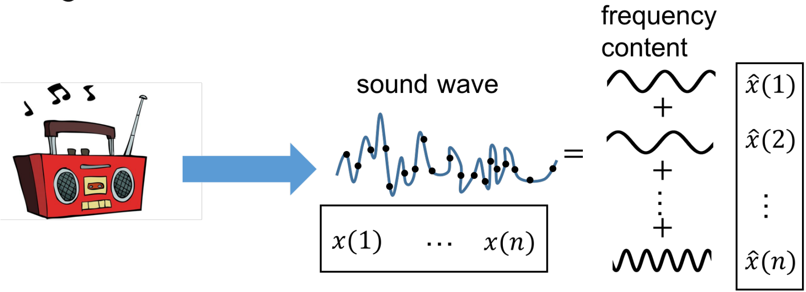
- Many signals are sparse in some basis (Fourier, wavelet, etc.).
- Using sparse recovery techniques, an n pixel image/n point signal can thus be recovered from many fewer than n measurements.
- Efficient MRI imaging, remote sensing for oil exploration, GPS synchronization, power efficient cameras, etc.

Many of the most important applications of sparse recovery are in imaging and signal processing.

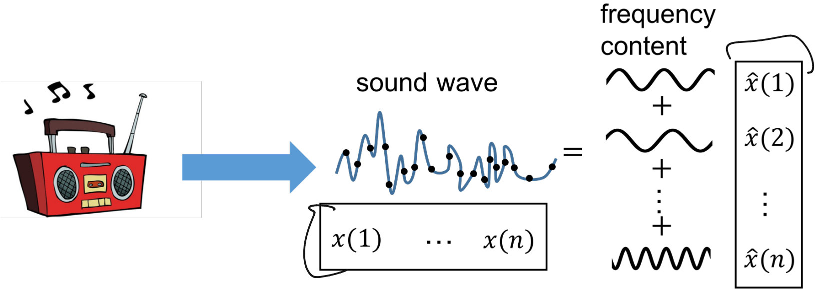
- Many signals are sparse in some basis (Fourier, wavelet, etc.).
- Using sparse recovery techniques, an n pixel image/n point signal can thus be recovered from many fewer than n measurements.
- Efficient MRI imaging, remote sensing for oil exploration, GPS synchronization, power efficient cameras, etc.

In general, there are a lot of practical complexities here. So everything I say is a major oversimplification.

Discrete Fourier Transform: For a discrete signal (aka a vector) $\mathbf{x} \in \mathbb{R}^n$, its discrete Fourier transform is denoted $\widehat{\mathbf{x}} \in \mathbb{C}^n$ and given by $\widehat{\mathbf{x}} = \mathbf{F}\mathbf{x}$, where $\mathbf{F} \in \mathbb{C}^{n \times n}$ is the discrete Fourier transform matrix.



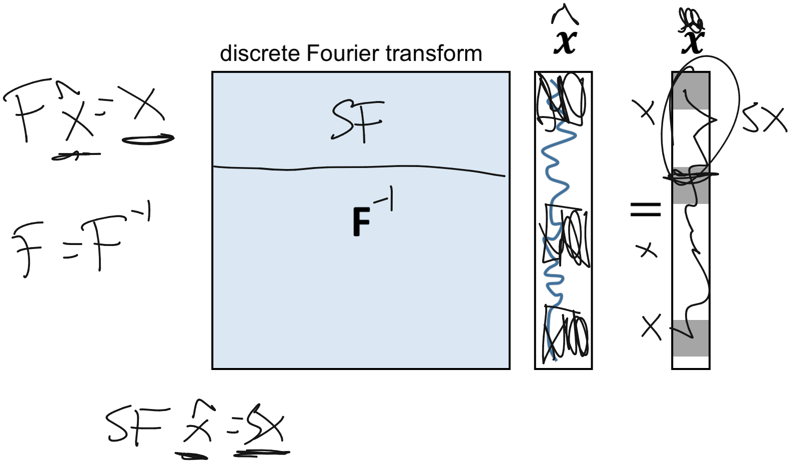
Discrete Fourier Transform: For a discrete signal (aka a vector) $\mathbf{x} \in \mathbb{R}^n$, its discrete Fourier transform is denoted $\widehat{\mathbf{x}} \in \mathbb{C}^n$ and given by $\widehat{\mathbf{x}} = \mathbf{F}\mathbf{x}$, where $\mathbf{F} \in \mathbb{C}^{n \times n}$ is the discrete Fourier transform matrix.



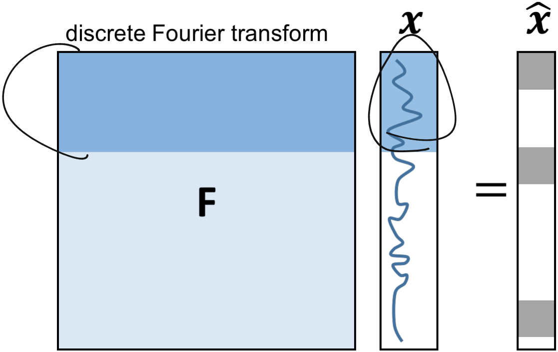
For many natural signals $\widehat{\mathbf{x}}$ is approximately sparse: a few dominant frequencies in a recording, superposition of a few radio transmitters sending at different frequencies, etc.

When the Fourier transform $\widehat{x}$ is sparse, can recover x from few measurements using sparse recovery.

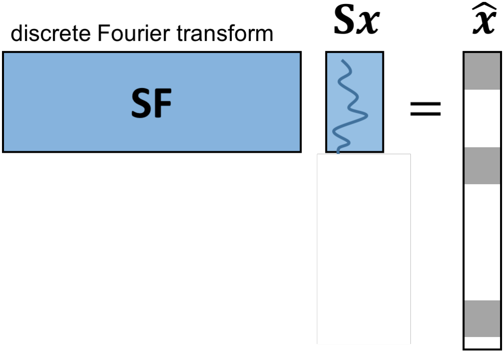
When the Fourier transform $\hat{\mathbf{x}}$ is sparse, can recover $\mathbf{x}$ from few measurements using sparse recovery.



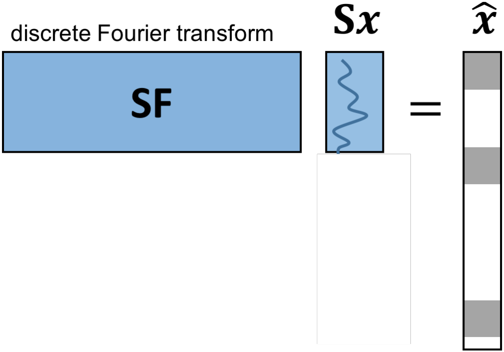
When the Fourier transform $\hat{\mathbf{x}}$ is sparse, can recover $\mathbf{x}$ from few measurements using sparse recovery.



When the Fourier transform $\widehat{x}$ is sparse, can recover x from few measurements using sparse recovery.



When the Fourier transform $\hat{x}$ is sparse, can recover x from few measurements using sparse recovery.



Translates to big savings in acquisition costs, the number of sensors required, etc.

Back to Algorithms

We would like to recover k-sparse $\mathbf{x}$ from measurements $\mathbf{b} = \mathbf{A}\mathbf{x}$ by solving the non-convex optimization problem:

$$\mathbf{x} = \arg\min_{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}} \|\mathbf{z}\|_0$$

Works if $\mathbf{A}$ has Kruskal rank $\geq 2k$, but very hard computationally.

We would like to recover k-sparse $\mathbf{x}$ from measurements $\mathbf{b} = \mathbf{A}\mathbf{x}$ by solving the non-convex optimization problem:

$$\mathbf{x} = \underset{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}}{\arg\min} \|\mathbf{z}\|_0$$

Works if $\mathbf{A}$ has Kruskal rank $\geq 2k$, but very hard computationally.

Convex Relaxation: A very common technique. Just ‘relax’ the problem to be convex.

$$\mathbf{x} = \underset{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}}{\arg\min} \|\mathbf{z}\|_1 \quad \text{where } \|\mathbf{z}\|_1 = \sum_{i=1}^{d} |\mathbf{z}(i)|.$$

We would like to recover k-sparse $\mathbf{x}$ from measurements $\mathbf{b} = \mathbf{A}\mathbf{x}$ by solving the non-convex optimization problem:

$$\mathbf{x} = \underset{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}}{\arg\min} \|\mathbf{z}\|_0$$

Works if $\mathbf{A}$ has Kruskal rank $\geq 2k$, but very hard computationally.

Convex Relaxation: A very common technique. Just 'relax' the problem to be convex. **Basis Pursuit:**

$$\mathbf{x} = \underset{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}}{\arg\min} \|\mathbf{z}\|_1 \quad \text{where } \|\mathbf{z}\|_1 = \sum_{i=1}^{d} |\mathbf{z}(i)|.$$

We would like to recover k-sparse $\mathbf{x}$ from measurements $\mathbf{b} = \mathbf{A}\mathbf{x}$ by solving the non-convex optimization problem:

$$\mathbf{x} = \arg\min_{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}} \|\mathbf{z}\|_0$$

Works if $\mathbf{A}$ has Kruskal rank $\geq 2k$, but very hard computationally.

Convex Relaxation: A very common technique. Just 'relax' the problem to be convex. **Basis Pursuit:**

$$\mathbf{x} = \arg\min_{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}} \|\mathbf{z}\|_1 \quad \text{where } \|\mathbf{z}\|_1 = \sum_{i=1}^{d} |\mathbf{z}(i)|.$$

What is one algorithm we have learned for solving this problem?

We would like to recover k-sparse $\mathbf{x}$ from measurements $\mathbf{b} = \mathbf{A}\mathbf{x}$ by solving the non-convex optimization problem:

$$\mathbf{x} = \arg\min_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_0$$

Works if $\mathbf{A}$ has Kruskal rank $\geq 2k$, but very hard computationally.

Convex Relaxation: A very common technique. Just 'relax' the problem to be convex. **Basis Pursuit:**

$$\mathbf{x} = \arg\min_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_1 \quad \text{where } \|\mathbf{z}\|_1 = \sum_{i=1}^{d} |\mathbf{z}(i)|.$$

What is one algorithm we have learned for solving this problem?

- Projected gradient descent – convex objective function and convex constraint set.

We would like to recover k-sparse $\mathbf{x}$ from measurements $\mathbf{b} = \mathbf{A}\mathbf{x}$ by solving the non-convex optimization problem:

$$\mathbf{x} = \arg\min_{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}} \|\mathbf{z}\|_0$$

Works if $\mathbf{A}$ has Kruskal rank $\geq 2k$, but very hard computationally.

Convex Relaxation: A very common technique. Just 'relax' the problem to be convex. **Basis Pursuit:**

$$\mathbf{x} = \arg\min_{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}} \|\mathbf{z}\|_1 \quad \text{where } \|\mathbf{z}\|_1 = \sum_{i=1}^{d} |\mathbf{z}(i)|.$$

What is one algorithm we have learned for solving this problem?

- Projected gradient descent – convex objective function and convex constraint set.
- An instance of linear programming, so typically faster to solve with a linear programming algorithm.

Why should we hope that the basis pursuit solution returns the unique k-sparse $\mathbf{x}$ with $\mathbf{A}\mathbf{x} = \mathbf{b}$? The minimizer $\mathbf{z}^*$ will have small ℓ_1 norm but why would it even be sparse?

$$\operatorname*{arg\,min}_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_1 \quad \text{vs.} \quad \operatorname*{arg\,min}_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_0$$

BASIS PURSUIT

Why should we hope that the basis pursuit solution returns the unique k-sparse $\mathbf{x}$ with $\mathbf{A}\mathbf{x} = \mathbf{b}$? The minimizer $\mathbf{z}^*$ will have small ℓ_1 norm but why would it even be sparse?

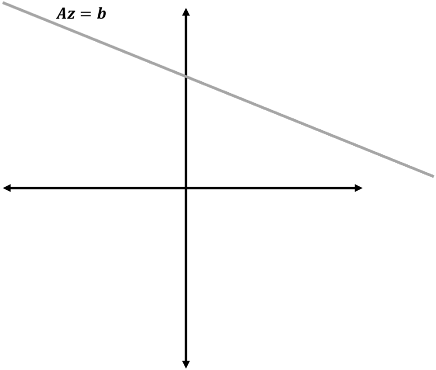
$$\arg\min_{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}} \|\mathbf{z}\|_1 \quad \text{vs.} \quad \arg\min_{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}} \|\mathbf{z}\|_0$$

Assume that $n = 1, d = 2, k = 1$. So $\mathbf{A} \in \mathbb{R}^{1 \times 2}$ and $\mathbf{x} \in \mathbb{R}^2$ is 1-sparse.

Why should we hope that the basis pursuit solution returns the unique k-sparse $\mathbf{x}$ with $\mathbf{A}\mathbf{x} = \mathbf{b}$? The minimizer $\mathbf{z}^*$ will have small ℓ_1 norm but why would it even be sparse?

$$\arg\min_{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}} \|\mathbf{z}\|_1 \quad \text{vs.} \quad \arg\min_{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}} \|\mathbf{z}\|_0$$

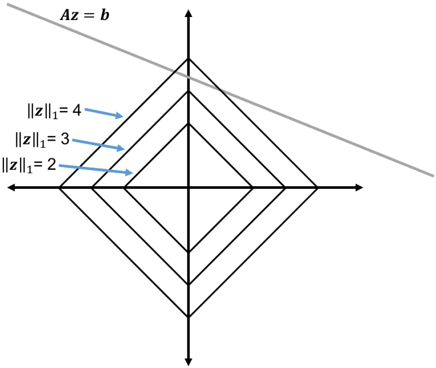
Assume that $n = 1, d = 2, k = 1$. So $\mathbf{A} \in \mathbb{R}^{1\times 2}$ and $\mathbf{x} \in \mathbb{R}^2$ is 1-sparse.



Why should we hope that the basis pursuit solution returns the unique k-sparse $\mathbf{x}$ with $\mathbf{A}\mathbf{x} = \mathbf{b}$? The minimizer $\mathbf{z}^*$ will have small ℓ_1 norm but why would it even be sparse?

$$\arg\min_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_1 \quad \text{vs.} \quad \arg\min_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_0$$

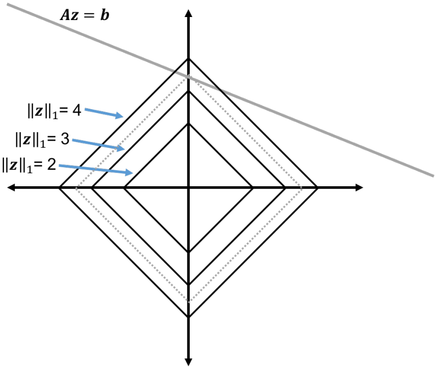
Assume that $n = 1, d = 2, k = 1$. So $\mathbf{A} \in \mathbb{R}^{1\times 2}$ and $\mathbf{x} \in \mathbb{R}^2$ is 1-sparse.



Why should we hope that the basis pursuit solution returns the unique k-sparse $\mathbf{x}$ with $\mathbf{A}\mathbf{x} = \mathbf{b}$? The minimizer $\mathbf{z}^*$ will have small ℓ_1 norm but why would it even be sparse?

$$\arg\min_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_1 \quad \text{vs.} \quad \arg\min_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_0$$

Assume that $n = 1, d = 2, k = 1$. So $\mathbf{A} \in \mathbb{R}^{1\times 2}$ and $\mathbf{x} \in \mathbb{R}^2$ is 1-sparse.

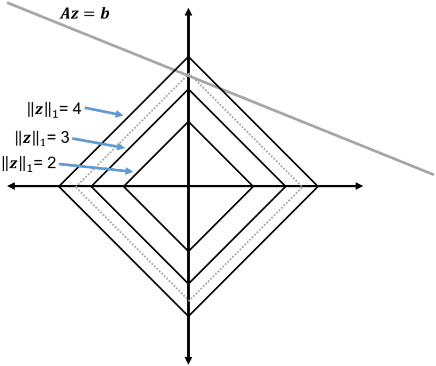


BASIS PURSUIT

Why should we hope that the basis pursuit solution returns the unique k-sparse $\mathbf{x}$ with $\mathbf{A}\mathbf{x} = \mathbf{b}$? The minimizer $\mathbf{z}^*$ will have small ℓ_1 norm but why would it even be sparse?

$$\arg\min_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_1 \quad \text{vs.} \quad \arg\min_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_0$$

Assume that $n = 1, d = 2, k = 1$. So $\mathbf{A} \in \mathbb{R}^{1\times 2}$ and $\mathbf{x} \in \mathbb{R}^2$ is 1-sparse.

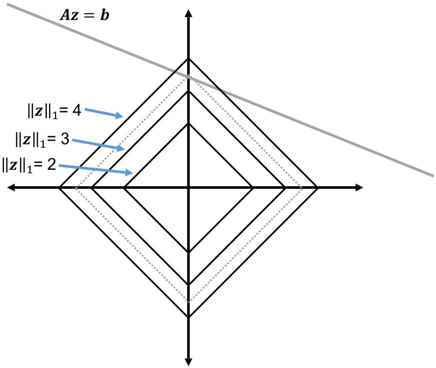


- Optimization solution will be on a corner (i.e., sparse), unless $\mathbf{A}\mathbf{z} = \mathbf{b}$ has slope 1.

Why should we hope that the basis pursuit solution returns the unique k-sparse $\mathbf{x}$ with $\mathbf{A}\mathbf{x} = \mathbf{b}$? The minimizer $\mathbf{z}^*$ will have small ℓ_1 norm but why would it even be sparse?

$$\arg\min_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_1 \quad \text{vs.} \quad \arg\min_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_0$$

Assume that $n = 1, d = 2, k = 1$. So $\mathbf{A} \in \mathbb{R}^{1\times 2}$ and $\mathbf{x} \in \mathbb{R}^2$ is 1-sparse.



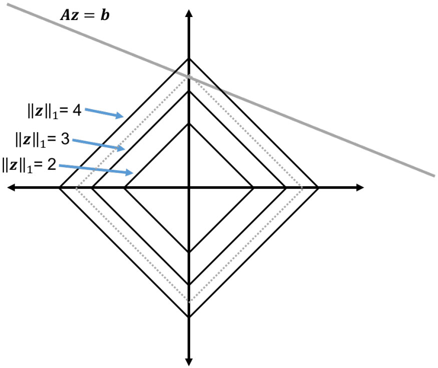
- Optimization solution will be on a corner (i.e., sparse), unless $\mathbf{A}\mathbf{z} = \mathbf{b}$ has slope 1.
- Similar intuition to the LASSO method.

BASIS PURSUIT

Why should we hope that the basis pursuit solution returns the unique k-sparse $\mathbf{x}$ with $\mathbf{A}\mathbf{x} = \mathbf{b}$? The minimizer $\mathbf{z}^*$ will have small ℓ_1 norm but why would it even be sparse?

$$\arg\min_{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}} \|\mathbf{z}\|_1 \quad \text{vs.} \quad \arg\min_{\mathbf{z} \in \mathbb{R}^d : \mathbf{A}\mathbf{z} = \mathbf{b}} \|\mathbf{z}\|_0$$

Assume that $n = 1, d = 2, k = 1$. So $\mathbf{A} \in \mathbb{R}^{1\times 2}$ and $\mathbf{x} \in \mathbb{R}^2$ is 1-sparse.



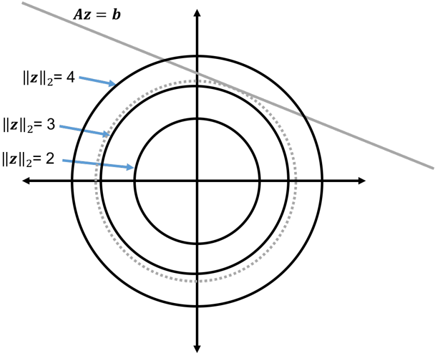
- Optimization solution will be on a corner (i.e., sparse), unless $\mathbf{A}\mathbf{z} = \mathbf{b}$ has slope 1.
- Similar intuition to the LASSO method.
- Does not hold if e.g., the ℓ_2 norm is used.

BASIS PURSUIT

Why should we hope that the basis pursuit solution returns the unique k-sparse $\mathbf{x}$ with $\mathbf{A}\mathbf{x} = \mathbf{b}$? The minimizer $\mathbf{z}^*$ will have small ℓ_1 norm but why would it even be sparse?

$$\arg\min_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_1 \quad \text{vs.} \quad \arg\min_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_0$$

Assume that $n = 1, d = 2, k = 1$. So $\mathbf{A} \in \mathbb{R}^{1\times 2}$ and $\mathbf{x} \in \mathbb{R}^2$ is 1-sparse.



- Optimization solution will be on a corner (i.e., sparse), unless $\mathbf{A}\mathbf{z} = \mathbf{b}$ has slope 1.
- Similar intuition to the LASSO method.
- Does not hold if e.g., the ℓ_2 norm is used.

BASIS PURSUIT THEOREM

Can prove that basis pursuit outputs the exact k-sparse solution $\mathbf{x}$ with $\mathbf{A}\mathbf{x} = \mathbf{b}$ (same as $\arg\min_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_0$)

- Requires a strengthening of the Kruskal rank $\geq 2k$ assumption (that still holds in all the applications discussed).

Can prove that basis pursuit outputs the exact k-sparse solution $\mathbf{x}$ with $\mathbf{A}\mathbf{x} = \mathbf{b}$ (same as $\arg\min_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_0$)

- Requires a strengthening of the Kruskal rank $\geq 2k$ assumption (that still holds in all the applications discussed).

Definition: $\mathbf{A} \in \mathbb{R}^{n\times d}$ has the (m, ϵ) restricted isometry property (is (m, ϵ)-RIP) if for all m-sparse vectors $\mathbf{x}$:

$$(1-\epsilon)\|\mathbf{x}\|_2 \leq \|\mathbf{A}\mathbf{x}\|_2 \leq (1+\epsilon)\|\mathbf{x}\|_2$$

Can prove that basis pursuit outputs the exact k-sparse solution $\mathbf{x}$ with $\mathbf{A}\mathbf{x} = \mathbf{b}$ (same as $\arg\min_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_0$)

- Requires a strengthening of the Kruskal rank $\geq 2k$ assumption (that still holds in all the applications discussed).

Definition: $\mathbf{A} \in \mathbb{R}^{n\times d}$ has the (m, ϵ) restricted isometry property (is (m, ϵ)-RIP) if for all m-sparse vectors $\mathbf{x}$:

$$(1-\epsilon)\|\mathbf{x}\|_2 \leq \|\mathbf{A}\mathbf{x}\|_2 \leq (1+\epsilon)\|\mathbf{x}\|_2$$

Theorem: If $\mathbf{A}$ is $(3k, \epsilon)$-RIP for small enough constant ϵ, then $\mathbf{z}^\star = \arg\min_{\mathbf{z}\in\mathbb{R}^d:\mathbf{A}\mathbf{z}=\mathbf{b}} \|\mathbf{z}\|_1$ is equal to the unique k-sparse $\mathbf{x}$ with $\mathbf{A}\mathbf{x} = \mathbf{b}$ (i.e., basis pursuit solves the sparse recovery problem).

Questions?